Abschlussbericht

der Projektgruppe

Themensteller:	Prof. Dr.-Ing. habil. Jorge Marx Gómez, Dr. Joachim Kurzhöfer
Betreuer:	Stefan Wunderlich M. Sc., Dipl.-Math. Jens Siewert
Vorgelegt von:	Daniel Martin Ahlers, Artjom Baranow, Sebastian Beckmann, Vanessa Dering, Holger Eichholz, Jan Fischer, Tobias Kromke, Jessica Schulte, Patrick Smit, Dirk Tesche, Jan Wiesemann
Abgabetermin:	8. April 2016

Abstract

In vielen deutschen und europäischen Unternehmen existiert ein bislang wenig genutztes Potenzial: die Kreativität der Mitarbeiter. Mit der in diesem Projekt entstandenen Plattform IMPACT wird eine Möglichkeit geschaffen, genau dieses Potenzial effektiv nutzen zu können. Dabei werden eine Reihe von Aspekten betrachtet, insbesondere den der Mitarbeitermotivation und des kollaborativen Arbeitens. Mittels eines fest vorgegebenen Ablaufes werden Mitarbeiter aktiv dabei unterstützt, innovative Ideen zu entwickeln, zu optimieren, und bis hin zur Umsetzung zu gestalten. Auch das oft vernachlässigte Potenzial am Anschluss an eine Umsetzung wird unterstützt. Es werden Rückmeldungen über abgeschlossene Entwicklungen verfasst, sodass nachfolgende Projekte auf Basis von Erfahrungen effektiver umgesetzt und Fehler vermieden werden können. Damit kann IMPACT den Innovationsprozess innerhalb eines Unternehmens nachhaltig optimieren.

Inhaltsverzeichnis

Abkürzungsverzeichnis

API	Application Programming Interface
App	Application
CDI	Contexts and Dependency Injection
CeBIT	Centrum für Büroautomation, Informationstechnologie und Telekomunikation
CI	Continuous Integration
EPK	Ereignisgesteuerte Prozessketten
FAQ	Frequently Asked Questions
IMPACT	Innovation Management Platform to Activate Creative Thoughts
JPQL	Java Persistence Query Language
KMU	Klein- und Mittelständische Unternehmen
LIS	Lufthansa Industry Solutions
MVC	Modell-View-Controller
ORM	Objektrelationales Mapping
SCSS	Syntactically Awesome Style
SQL	Structured Query Language
UI	User Interface
UML	Unified Modeling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VLBA	Very Large Business Applications

Abbildungsverzeichnis

Tabellenverzeichnis

Quelltextverzeichnis

Glossar

Brainstorming Das Brainstorming ist eine Methode zur Findung von neuen und kreativen Ideen.. 37, 56

Crowdfunding Das Crowdfunding oder auch Schwarmfinanzierung ist ein Finanzierungsmethode zur Geldbeschaffung für Projekte oder Produkte. XXVII, 39, 241

CSS Cascading Style Sheets (CSS) ist eine Sprache zur grafischen Gestaltung von elektronischen Dokumenten. Mit CSS werden Gestaltungsanweisungen für Oberflächen erstellt, z.B. für HTML-Elemente. 72, 81, 82

Entwicklungsmodell Das Entwicklungsmodell wird in der Softwareentwicklung auch als Vorgehensmodell bezeichnet. Es dient dazu die Softwareentwicklung übersichtlich zu gestalten und in der Komplexität beherrschbar zu machen. 6

Ereignissgesteuerte Prozessketten Die Ereignisgesteuerte Prozesskette (EPK) ist eine grafische Modellierungssprache zur Darstellung von Geschäftsprozessen. 19

Feedback Das Feedback bezeichnet eine Rückmeldung oder Rückinformation in der Kommunikation von Menschen. Bei diesen Informationen meldet der Empfänger dem Sender, was dieser wahrgenommen hat und ermöglicht so eine etwaige Korrektur und Reaktion. XXVIII

Framework Ein Framework stellt eine wiederverwendbare, gemeinsame Struktur zur Verfügung, mit der ein Programmierer seine Anwendung entwickelt. Dies wird insbesondere im Rahmen der objektorientierten Programmierung eingesetzt. 76, 81, 93, 119

GIT GIT ist eine freie Software für die verteilte Versionsverwaltung von Daten. 18–20

Java Persistence API Die Java Persistence API (JPA) ist eine Spezifikation für Java-Anwendungen, dessen Implementierungen das automatische Mapping von Objekten zu relationalen Datenbanken ermöglicht. 76, 77

JUnit JUnit bezeichnet ein Framework für Modul-Tests für Java-Programmen. 20, 214

Objektrelationales Mapping Das Objektrelationales Mapping ist eine Technik der Softwareentwicklung, die automatisches Mapping von Objekten zu relationalen Datenbanken ermöglicht. 76, 214

Roadmap Die Roadmap stellt einen Projektplan dar, welcher eine grobe Planung über einen längeren Zeitraum umfasst und aufzeigt, wie sich ein Produkt über einen strategischen Zeitraum entwickelt. 12

Selenium Selenium ist eine Testumgebung für Benutzerschnittstellen von Webanwendungen. Selenium ermöglicht dabei einen automatisierten Test mit beliebig vielen Wiederholungen. 214

UI UI steht für User Interface und bezeichnet die Schnittstelle zwischen der Anwendung und dem Anwender, damit ist üblicherweise eine grafische Oberfläche gemeint. 188, 195, 197

Wireframe Das Wireframe wird in der Web-Entwicklung genutzt, um in einer sehr frühen Phase der Entwicklung die Oberfläche der Website oder Software darzustellen. Dabei geht es nicht um die Funktionen, sondern um die Anordnung der Elemente. XXXIX, XL

1. Einleitung

Die Innovationsmanagementplattform IMPACT stellt einen neuen, transparenten und einheitlichen Ansatz für innerbetriebliches Innovationsmanagement bereit. Der eigens entwickelte Prozess ermöglicht es, Ideen, Probleme und Vorschläge im innerbetrieblichen Kontext zu identifizieren. Im weiteren Prozessverlauf werden diese gemeinsam bearbeitet und eine Lösung entwickelt. Hinter dem Namen IMPACT (Innovation Management to Activate Creative Thoughts) verbirgt sich zum einen der Wunsch, dass jeder Mitarbeiter einen positiven Einfluss (impact) in seinem Unternehmen hinterlassen kann und zum anderen, dass vorhandenes Wissen genutzt wird, um dem Unternehmen einen langfristigen Mehrwert zu generieren.

1.1. Motivation

Viele Unternehmen sind sich der Problematik bewusst, dass Innovationen, für die Entwicklung ihres Unternehmens, erforderlich sind. Eine Innovation kann als unkonventionelles Produkt oder Verfahren verstanden werden, welches sich deutlich von dem bisherigen unterscheidet (vgl. [HS11], S.4). In der Praxis verlieren Unternehmen oftmals den Innovationsgedanken und fokussieren sich zu sehr auf den täglichen Betrieb. Dabei geht enormes Entwicklungs- und Innovationspotenzial verloren, wodurch es dem Unternehmen erschwert wird, konkurrenzfähig zu bleiben (vgl. [SJ10], S.4). Folgende Probleme können durch dieses Verhalten entstehen:

- Für die strategische Planung von innerbetrieblichen Optimierungen bleibt im Alltagsgeschäft keine Zeit.
- Es existieren Ideen, bei denen nicht sichergestellt werden kann, ob diese erfolgreich umgesetzt werden können.
- Neuerungen werden nur langsam umgesetzt.

Die aufgelisteten Probleme können durch ein innerbetriebliches Innovationsmanagement aufgefangen und zum Teil gelöst werden. Das Innovationsmanagement kann die Entstehung, Verarbeitung und Gewinnung von neuen Ideen fördern, welche einen essenziellen Faktor für den Fortbestand und die Wertschöpfung eines Unternehmens darstellen. Um dies zu erreichen, setzt ein Innovationsmanagament folgende Ziele (vgl. [SJ10], S.8):

- Gewinnerzielung,
- Wachstum,
- Verbesserung der Wettbewerbsposition,

- Steuerung des Produktportfolios,
- Vermarktung von Erfindungen,
- Anpassung an veränderte Kundenwünsche,
- Sicherung von Arbeitsplätzen,
- Imagepflege und
- Förderung des Gemeinwohls und Verbesserung des Umweltschutzes.

Unternehmen müssen sich einer Vielzahl neuer Herausforderungen stellen. Die fortschreitende Globalisierung sorgt für internationale Konkurrenz und die Erhöhung des Wettbewerbsdrucks. Hinzu kommen die technischen Möglichkeiten, insbesondere im Bereich der Informations- und Kommunikationstechnik. Kürzere Produktlebenszyklen und die Erwartungshaltung der Konsumenten verlangen nach kreativen Ideen und innovativen Produkten. Zudem steht den Unternehmen eine größere Menge an Wissen zur Verfügung. Alles führt dazu, dass sich die Unternehmen in einem weltweiten Wissens- und Zeitwettbewerb befinden. Die dabei entstehende Konkurrenzsituation führt zu dem Zwang, den Innovationsprozess nachhaltig zu fördern (vgl. [SJ10]).
Für Unternehmen heißt dies nicht nur, dass diese ihr Angebot entsprechend optimieren müssen, sondern auch intern neue Lösungen finden müssen, um Fertigungs- und Geschäftsprozesse effizienter zu gestalten, sowie Arbeitsweisen zu optimieren. Die Schaffung eines innovationsfördernden Arbeitsklimas kann einer der Wege sein, sich den verschärften Wettbewerbsbedingungen erfolgreich zu stellen. Diese Aussage unterstützt Bill Fischer, Autor bei *forbes.com*, indem er erwähnt, dass Innovationen und das Entwickeln dieser erst zustande kommen können, wenn das Unternehmen sich dementsprechend öffnet und die Möglichkeit bietet, Innovationen bzw. Ideen zu fördern (vgl. [Fis11]). Um sich den Herausforderungen zu stellen und wettbewerbsfähig zu bleiben, ist es notwendig, dass ein Unternehmen innovationsfähig ist (vgl. [SJ10], S.4).
Dabei kann die Forschungs- und Entwicklungsabteilung nur einen kleinen Teil zur Innovationsgenerierung leisten. Das meiste Potenzial verbirgt sich in dem Wissen der Mitarbeiter und deren Expertise (vgl. [SRU10]). Für Unternehmen ist es unumgänglich, dieses Wissen zu extrahieren und daraus Gewinn zu generieren. Während das systematische und planmäßige Ideenmanagement in großen Unternehmen eine Verbreitung von 80% erreicht hat, ist im Bereich von kleinen und mittelständischen Unternehmen (im Folgenden KMU) nur eine Verbreitung von 7% zu erkennen (vgl.[Rid00], S.17). Dies zeigt, dass dort ein großes Potential für die Entwicklung einer Innovationsmanagementplattform besteht. IMPACT setzt an dieser Stelle an und stellt Unternehmen eine Plattform bereit, um aus Problemen Lösungen entstehen zu lassen.

1.2. Problemstellung

Viele Innovationsprozesse werden in der Regel auf Wunsch der Kunden vorangetrieben, selten jedoch von den internen Mitarbeitern. Auch fehlen oft die Zeit oder die methodischen Kenntnisse, um innovative Lösungen umzusetzen (vgl. [RB08], S.25). Ein weiteres Problem, welches in Bezug auf den Praxispartner (Lufthansa Industrie Solutions) ermittelt wurde, besteht darin, dass oftmals kein richtiger Innovationsprozess existiert oder implementiert wurde. Innerhalb der Lufthansa Industrie Solutions ist lediglich eine *Ideenbox* vorhanden, in die Mitarbeiter ihre Verbesserungsvorschläge einwerfen können. Die abgegebenen Vorschläge und Ideen geraten dabei in Vergessenheit und werden nur von einer Person bewertet, und damit nur subjektiv wahrgenommen (vgl. Protokoll A.7.3). Durch eine *Ideenbox* werden die Ziele eines Innovationsmanagements nicht oder nur in geringem Maße erreicht. Rüggeberg und Burmeister haben ermittelt, dass es besonders förderlich ist, Mitarbeiter früh in den Innovationsprozess einzubinden und diesen softwareunterstützend darzustellen (vgl. [RB08], S.32). Diese beiden Anforderungen werden von einer *Ideenbox* nicht umgesetzt. Zum einen werden die Mitarbeiter in den Prozess der Entscheidung nicht eingebunden, zum anderen ist dieser Ansatz nicht standortübergreifend einsetzbar. Zusätzlich wird der Weg zur Entscheidung nicht transparent und somit auch nicht nachvollziehbar dargestellt.
Das Ziel der *Förderung des Gemeinwohls* eines Innovationsmanagement kann ebenfalls nur schwer erreicht werden. Durch die Intransparenz des Prozesses und der Entscheidung werden zum einen weniger Vorschläge eingereicht und zum anderen wird die Akzeptanz der *Ideenbox* vermindert. Mitarbeiter werden frustriert und das Gemeinwohl nicht gefördert.
Zusätzlich ist zu erwähnen, dass Vorschläge zur Optimierung von betriebsinternen Prozesses nicht umgesetzt werden, da ein alleiniger Mitarbeiter über die Umsetzung eines Vorschlages entscheidet. Damit kann das Potenzial und der resultierende Mehrwert vieler Vorschläge unerkannt bleiben.
Der bisher umgesetzte Innovationsprozess ist nicht zufriedenstellend und soll zukünftig die generellen Ziele eines Innovationsprozesses erfüllen. Darüber hinaus soll der Innovationsprozess in mehrere Stufen unterteilt werden. Einige dieser Angaben wurden bereits in dem Kick-off-Meeting definiert.

1.3. Zielsetzung

Das Ziel des Projektes besteht darin, eine Innovationsmanagementplattform umzusetzen, die einen softwareunterstützenden Innovationsprozess abbildet und es den Mitarbeitern ermöglicht, daran teilzunehmen. Dabei wird der Fokus vor allem auf einen innovativen, stufenbasierten und transparenten Prozess gelegt. Insbesondere werden die Mitarbeiter

von Beginn an in den Prozess integriert. Dadurch soll eine höhere Akzeptanz und Motivation geschaffen werden, die es den Mitarbeitern ermöglicht, Ideen, Vorschläge und Probleme einzureichen. Andere Mitarbeiter sollen in der Lage sein, diese Ideen Vorschläge oder Probleme zu bewerten und zu kommentieren. Infolgedessen sollen die Ideen insoweit konkretisiert und optimiert werden, dass das Unternehmen diese umsetzen kann. Durch die Optimierung von Unternehmensprozessen soll eine Verbesserung der Wettbewerbsposition, der Gewinne sowie des Wachstums erfolgen.
Darüber hinaus sollen Methoden zur kreativen Umsetzung von Ideen, und Kommunikationsmöglichkeiten innerhalb des Systems, welche standortübergreifend bereitgestellt werden können, eingesetzt werden. Es wird ebenso gewährleistet, dass anderen Anwendern generiertes Wissen zur Verfügung steht und nicht verloren geht.

1.4. Projektgruppe

Die Projektgruppe ist ein einjähriges Projekt im Rahmen des Masterstudiums im Department für Informatik der Carl von Ossietzky Universität Oldenburg. Bei der Projektgruppe Innovation Management Platform to Activate Creative Thoughts (IMPACT) handelt es sich um eine Gruppe aus elf Studenten. Ziel der einjährigen Zusammenarbeit in der Abteilung Very Large Business Applications (VLBA) ist es, gemeinsam als Auftragnehmer eines Softwareentwicklungsprojektes aufzutreten und gemäß der ermittelten Anforderungen einen Prototypen zu erstellen.
Der Wissenstransfer untereinander, der Aufbau von Know-how und das Sammeln neuer Erfahrungen im Bereich Softwareentwicklung und Projektmanagement stehen dabei im Vordergrund. Dazu werden, neben einer Seminarphase zum Aufbau von Grundlagen und Wissensangleichung zu Thematiken im Softwareprojektkontext, die bei einem Softwareentwicklungsprojekt typischen Phasen durchlaufen. Während der gesamten Dauer steht das Projektteam in intensivem Kontakt mit den Auftraggebern. Auftraggeber der Projektgruppe IMPACT ist zum einen die Lufthansa Industry Solutions, repräsentiert durch Herrn Dr. Joachim Kurzhöfer. Die Lufthansa Industry Solutions beschäftigt deutschlandweit mehr als 1.200 Mitarbeiter, davon ca. 80 Mitarbeiter am Standort Oldenburg und agiert innerhalb der Deutschen Lufthansa AG als eigenständiger IT-Dienstleister. Zum anderen fungieren die Betreuer Stefan Wunderlich und Jens Siewert, aus der Universistätsabteilung VLBA, als Auftraggeber.

1.5. Aufbau des Abschlussberichtes

Dieser Abschlussbericht gliedert sich in 12 Kapitel. In der Einleitung wird auf die Problemstellung und die Motivation eingegangen, eine Innovationsmanagementplattform zu entwickeln. Darüber hinaus werden weitere Informationen zur Projektgruppe geliefert.

In Kapitel 2 wird die Organisation innerhalb der Projektgruppe beschrieben. Dazu wird auf die verschiedenen Rollen und Aufgabenbereiche näher eingegangen und die Zeitplanung dargestellt. Außerdem werden die entwickelten Arbeitsgrundlagen und getroffenen Absprachen näher erläutert.
Nach der Seminarphase startete das eigentliche Projekt mit der Anforderungsanalyse. Die dabei angewandten Methoden und Werkzeuge sind in Kapitel 3 genauer beschrieben. Außerdem werden die daraus abgeleiteten funktionalen und nicht-funktionalen Anforderungen aufgelistet.
Die darauf folgende Konzeptionsphase wird in Kapitel 4 beschrieben. Dazu werden die getroffenen Entscheidungen hinsichtlich des Prozessmodells, der verwendeten Technologien und des UI-Designs dargestellt und begründet.
Die beiden darauf folgenden Kapitel gehen näher auf die Realisierung ein. Dabei liegt der Fokus des Kapitels 5 auf dem zugrundeliegenden, technischen Architekturmodell hinsichtlich der verschiedenen Schichten und Module. Weiterhin wird die Datenbank und das Datenmodell vorgestellt.
In Kapitel 6 wird die Realisierung der einzelnen Systemkomponenten beschrieben. Dies beinhaltet sowohl das UI-Design, die einzelnen Schritte des Innovationsprozesses, sowie zusätzliche Funktionen wie der Activity Stream, die Bewertungsfunktion oder den Chat. Nicht umgesetzte Anforderungen sind ebenfalls mit entsprechender Begründung aufgelistet.
In Kapitel 7 wird der Projektfortschritt mit den einzelnen Phasen und den im Rahmen des SCRUM-Prozesses durchgeführten Sprints, dokumentiert. Außerdem wird näher auf den Projektabschluss eingegangen.
Um die Qualität der entwickelten Software sicherzustellen, wurden verschiedene Tests durchgeführt. Die angewendeten Verfahren, sowie deren Planung, Durchführung und Auswertung sind in Kapitel 8 beschrieben.
Das Projektergebnis stellt derzeit einen Prototypen dar und bietet noch Entwicklungspotenzial. Die in Zukunft möglichen Erweiterungen sind in Kapitel 9 beschrieben.
Wie bei den meisten Prototypen, beinhaltet das Projektergebnis einige Fehler. Eine Auflistung von bekannten Fehlern, welche im Rahmen des Projekts noch nicht behoben werden konnten, wird in Kapitel 10 gezeigt.
Ein abschließendes Fazit über den gesamten Projektzeitrum wird in Kapitel 11 gezeigt. Dazu werden der gesammelten Erfahrungen der Projektgruppe präsentiert.
In Kapitel 12 wird schließlich in Aussicht gestellt, wie die Software nach dem offiziellen Abschluss weiterentwickelt und zum Einsatz kommen kann.

2. Projektmanagement

Im folgenden Kapitel werden die Elemente der Organisation innerhalb der Projektgruppe näher erläutert. Dazu zählen die Rollenverteilung, Workshops und Regeln zur Mitarbeit.

2.1. Rollenaufteilung

Die Durchführung eines Projekts mit elf internen Beteiligten und über den Zeitraum von einem Jahr macht es erforderlich, dass klare Rollen definiert werden. Dabei ergeben sich einige Rollen, durch die Auswahl des Entwicklungsmodells und weitere Rollen durch die Vorgaben seitens der Aufgabensteller bzw. Betreuer.
Die Festlegung der Rollen soll dabei bereits zu Beginn eine strukturierte und effiziente Arbeitsweise der Projektphase ermöglichen, dazu wurden interne Verantwortliche und Aufgabenbereiche definiert. Damit wurde sichergestellt, dass für alle Fragen und Probleme ein Ansprechpartner zur Verfügung steht.
Dabei wurden die folgenden Rollen innerhalb von IMPACT definiert:

- Projektleiter,
- Scrum-Master,
- Serveradministrator,
- Websitebeauftragter,
- Konfliktmanager,
- Product Owner und
- Entwickler

Projektleiter

Der Projektleiter spielt dabei eine besonders zentrale Rolle, da er für die Steuerung und auch für die operative Planung des Projektes zuständig ist. Dies umfasst z. B. die Abstimmung von Terminen, Gespräche mit den Betreuern und dem Praxispartner sowie die grundlegende Kontrolle, ob gesteckte Ziele erreicht werden. Die Steuerung des Projektes erfolgt in Zusammenarbeit mit dem Scrum-Master. Durch die Vorgaben der Projektbetreuer, wird diese Rolle gewechselt, sodass verschiedene Projektbeteiligte diese zentrale Rolle in einem Projekt begleiten können. **Scrum-Master** Durch die Wahl des agilen Entwicklungsvorgehens mittels Scrum, und dessen hoher Dynamik während des Entwicklungsprozesses, nimmt der Scrum-Master, neben dem Projektleiter, eine weitere kontrollierende Rolle ein. Der Scrum-Master soll dafür sorgen, dass aus der hohen Dynamik keine unkontrollierte Desorganisation entsteht. Er hat primär die Aufgabe, dafür zu sorgen, dass die Regeln

und Vorgehenweisen des Scrum-Prozesses eingehalten werden. Der Scrum-Master soll die nachfolgenden Aufgaben einnehmen (vgl. [Kri16]):

- Der Scrum-Master trägt die Verantwortung für die korrekte Implementation des Scrum-Prozesses,
- Er ist ein Vermittler und sorgt für Informationsfluss zwischen Product Owner und dem Team,
- Er hat die Aktualität der Scrum-Artefakte (Product Backlog, Sprint Backlog, Burndown Chats) im Blick und
- Er moderiert die Scrum-Meetings

Serveradministrator

Neben den organisatorischen Rollen existieren technische Rollen, dazu zählt der Serveradministrator. Dieser ist für die Installation, der Einrichtung, sowie der Wartung der benötigten Server zuständig. Im laufenden Betrieb spielt zudem die Optimierung der Komponenten eine wichtige Rolle.

Websitebeauftragter

Um die Projektgruppe auch nach außen hin zu präsentieren, ist die Entscheidung gefallen, eine Wordpress-Seite[1] einzurichten, um bspw. über besondere Vorkommnisse, technologische Entscheidungen, Aktivitäten und Meilensteine zu berichten. Die Wartung und Pflege ist damit die Aufgabe des Webseitenbeauftragten.

Konfliktmanager

Bei einer Gruppenarbeit über einen längeren Zeitraum, ist es häufig nicht zu vermeiden, dass es zu Konflikten kommt. Um mögliche Eskalationen zu entschärfen, wurde frühzeitig ein Konfliktmanager benannt. Die Aufgabe ist dabei, als Vermittler zwischen den Parteien zu agieren und mit ihnen eine einvernehmliche Lösung zu finden.

Product Owner

Aufgrund der Aufgabenstellung wurde die Rolle des Product Owners definiert. Dieser fungiert als Schnittstelle zwischen dem Projektteam und dem Auftraggeber. Gerade bei Projekten mit externen Auftraggebern ist es essenziell, dass ein interner Ansprechpartner für Fragen innerhalb der Entwicklung vorhanden ist. Der Product Owner nimmt dabei die Position des internen Auftraggebers ein und stimmt sich dabei mit dem externen Auftraggeber ab.

[1] https://de.wordpress.org/

Entwickler

Eine Rolle, die jedes Mitglied von IMPACT begleitet, ist die des Entwicklers. Dies steht im Gegensatz zur Praxis, in der lediglich ein Teil des Projektteams die Rolle des Entwicklers einnimmt.

2.2. Aufgabenbereiche

Zur besseren Einteilung und Strukturierung des gesamten Projektablaufes, wurden eine Reihe von Aufgabenbereichen definiert. Diese werden im Folgenden detaillierter beschrieben.

Anforderungsdefinition

Zu den Aufgabenbereichen, gerade zu Beginn der Projektgruppe, zählen die Anforderungsdefinition und der Entwurf. Diese Phase legt dabei die Grundlage, um ein Innovationsmanagementsystem zielgerichtet entwickeln zu können. Dazu ist es essenziell, eine möglichst vollständige und verbindliche Sammlung von Anforderungen zu formulieren. Die genaue Vorgehensweise der Anforderungsermittlung ist unter Kapitel *Anforderungen* 3 nachzulesen.
Aus den festgelegten Anforderungen wurden die ersten Entwürfe eines Prozessmodelles erstellt. Dieses wurden im weiteren Verlauf, entsprechend weiterer Anforderungen, angepasst.

Konzeption

In dem zweiten Aufgabenbereich wird die Konzeption fokussiert. Dabei wird das im vorherigen Schritt erstellte Prozessmodell in einzelne, detaillierte Teilkonzepte überführt. Diese umfassen, neben der Struktur, auch Informationen für die technische Umsetzung.

Entwicklung

Der Großteil der Projektphase bestand aus der Entwicklung der Plattform. Dabei wurden die, in der Anforderungsdefinition festgelegten, Funktionen mittels Scrum-Prozess / Product Backlog implementiert. Durch den Einsatz agiler Prozesse verläuft die Entwicklung häufig parallel zu dem Entwurf. Neu aufgenommene Anforderungen können den Entwurf und das Konzept verändern, bevor sie als Funktion im Product Backlog erscheinen.

Tests

Mit Hilfe von Softwaretests wird die Qualität der entwickelten Lösung sichergestellt. Die Software wird auf die im Vorfeld definierten Anforderungen überprüft, sowie deren korrekte Umsetzung überwacht. Im Rahmen der Softwaretests können zudem Fehler auffallen, die anschließend behoben werden können. Die Durchführung der Softwaretests ist im Kapitel 8 nachzulesen.

Design

Der Aufgabenbereich des Designs umfasst die optische Anpassung der Innovationsmanagementplattform. Dabei werden bspw. die in der Entwurfsphase festgelegten Farbcodes und Kombinationen umgesetzt. Zudem trägt dieser Bereich für die Userbility bei, indem dort bspw. die Anordnung von Buttons, Tabellen sowie Vereinheitlichung von Ausdrücken durchgeführt wird.

Benutzerdokumentation

Mittels der Benutzerdokumentation wird dem Anwender die Funktionsweise der Software gezeigt, wie diese verwendet werden kann und welche Daten erfasst und verarbeitet werden. Eine vollständige Dokumentation umfasst hingegen mehrere Bereiche, die auf verschiedene Stakeholder ausgelegt sind (vgl. [Hei16]):

- Benutzerhandbuch
- Installationsdokumente A.1
- Spezifikation 3
- Projektdokumentation
- Testdokumentation 8
- Entwicklungsdokumentationen

2.3. SCRUM

Mit Scrum wurde ein Vorgehensmodell der agilen Softwareentwicklung gewählt, welches im Rahmen der Projektarbeit angewendet wird. Innerhalb dieses Kapitels wird dem Leser ein Überblick über Scrum vermittelt. Des Weiteren wird die Entscheidung der Projektgruppe für dieses Vorgehensmodell erläutert und begründet. Abschließend werden die im Rahmen dieser Dokumentation notwendigen geplanten Modifikationen des Prozesses vorgestellt.

Einführung in das Vorgehensmodell Scrum

Scrum basiert auf der allgemeinen, agilen Vorgehensweise und legt dabei den Schwerpunkt auf die Kommunikation und Selbstorganisation des Teams. Es ist ein Framework, welches aus Rollen, Meetings sowie Artefakten besteht und alle Aktivitäten der Produktentwicklung abläuft (vgl. [Glo13], S.9).
Scrum legt Wert auf eine klare Rollenverteilung innerhalb des Prozesses. Rollen, die in jedem Scrum-Projekt belegt sein müssen, sind der *Product Owner*, das *Entwicklungsteam* und der *Scrum Master*. Darüber hinaus existieren weitere Rollen, wie Manager, Anwender und Kunde. Diese können sich in Projekten überschneiden oder auch fehlen. Der *Product Owner* wird entweder vom Kunden oder aus dem eigenen Unternehmen gestellt und vertritt die Wünsche des Kunden gegenüber dem Entwicklerteam. Er ist verantwortlich dafür, dass das Team die Funktionalitäten in richtiger Reihenfolge entwickelt und arbeitet kontinuierlich an dem *Product Backlog* (vgl. [Glo13], S.12). Das *Entwicklungsteam* entwickelt das Produkt und ist selbstständig organisiert. Es besteht meistens aus 5-10 Mitgliedern. Der *Scrum Master* ist verantwortlich für die Einhaltung des Scrum-Prozesses und greift nicht direkt in das Geschehen ein. Er ist ebenfalls dafür zuständig, Probleme zu beseitigen, die das Team aufhalten könnten (vgl. [Glo13], S.12).
Artefakte, die innerhalb von Scrum erstellt und bearbeitet werden, sind z.B. das *Product Backlog*, das eine priorisierte Liste der zu liefernden Funktionalitäten darstellt. Diese werden auch *Product Backlog Items* genannt und als User Stories formuliert. Oftmals werden *Product Backlog Items* und Anforderungen synonym verwendet. Es muss aber darauf hingewiesen werden, dass Anforderungen mit vielen Informationen, z. B. der Entwicklung, angereichert sind. Diese Informationen werden erst in dem *Sprint Planning* und den weiteren Meetings hinzugefügt (vgl.[Glo13], S.81). Der *Sprint Backlog* enthält alle Items aus dem *Product Backlog*, welche innerhalb eines Sprints bearbeitet werden sollen. Diese werden vom *Entwicklerteam* mit konkreten Entwicklungstätigkeiten ergänzt. Das *Impediment Backlog* ist eine Liste, welche alle Blocker (Impediments), die das Team an einer effektiven Arbeitsweise hindern, beinhaltet. Es muss dann versucht werden, diese Blocker zu beseitigen. Wenn diese prozessual bedingt sind, ist der *Scrum Master* für die Beseitigung dieser Hindernisse verantwortlich. Ein weiteres Artefakt ist das Produkt-Inkrement, welches als eine Art Teilsystem aufgefasst werden kann. Es beinhaltet Funktionen, die dem Kunden präsentiert und ausgeliefert werden könnten. Innerhalb des Prozesses finden vor, nach und während eines Sprints Meetings statt, die dem gesamten Team zum Fortschritt des Projektes verhelfen sollen.
Zu Beginn eines jenen Sprints steht das *Sprint Planning*. In diesem Meeting wird das Ziel eines Sprints definiert, die dazu passenden *Product Backlog Items* ausgewählt und Möglichkeiten zum Erreichen des Ziels besprochen. Dazu zählen Auswahl der Architektur der Applikation, Test Cases und Interfaces (vgl. [Glo13], Abschnitt B, S.13). Wenn alle

notwendigen Aufgaben notiert sind, werden diese in das *Sprint Backlog* überführt. Anwesend ist das gesamte Scrum-Team. Danach startet der zeitlich begrenzte Sprint, innerhalb dessen täglich das *Daily Scrum* stattfindet, bei dem jedes Mitglied des *Entwicklerteams* kurz erklärt, welche Aufgaben erledigt sind, aufgenommen werden oder Probleme bereiten. Geleitet wird das Meeting vom *Scrum Master*. Ist der Sprint nach der vorgegeben Zeit erledigt und ein lauffähiges System bereitgestellt, findet zum einen das *Sprint Review* und zum anderen die *Sprint Retrospektive* statt. In dem *Sprint Review* werden die erstellten Funktionalitäten präsentiert und besprochen. Es ist möglich, Feedback zu den lauffähigen Funktionalitäten zu geben und damit die Produktqualität zu erhöhen. Eventuelle daraus entstandene Anforderungen können in das *Product Backlog* als *Product Backlog Items* eingefügt werden. Die *Sprint Retrospektive* setzt auf die Prozessverbesserung. Hierbei werden Arbeitsabläufe analysiert und festgestellt, welche optimiert werden können, damit das *Entwicklerteam* produktiver arbeiten kann. Die Ergebnisse werden in dem *Impediment Backlog* festgehalten und im nächsten *Sprint Planning* als Verbesserungsvorschläge eingebracht (vgl. [Glo13],Abschnitt B, S.13).

Entscheidung für Scrum

Scrum bietet vor allem Vorteile in der Organisationsstruktur und besitzt im Verhältnis zu anderen Vorgehensmodellen weniger Risiken. Durch die Visualisierung des Fortschritts innerhalb eines Sprints soll gewährleistet werden, dass alle Teilnehmer fokussiert und motiviert an dem Projekt arbeiten. Ebenso soll durch eine klare Verantwortlichkeitsstruktur gesichert werden, dass das Team zielstrebig und konsequent an dem Projekt arbeitet. Darüber hinaus bietet Scrum die Möglichkeit, die selbstständige Arbeit der Projektgruppe zu fördern. Jedes Mitglied kann neue Bereiche kennenlernen und seine fachlichen Kompetenzen ausbauen.
Diese Besonderheiten ermöglichen es der Projektgruppe IMPACT, koordiniert, strukturiert und produktiv zu arbeiten. Die Wahl auf das Vorgehensmodell Scrum fiel nach dem Abwägen der Vor- und Nachteile gegenüber anderen Vorgehensmodellen. In der Seminararbeit zum Thema *Softwareentwicklungs-Vorgehensmodell* ist der detaillierte Entscheidungsweg beschrieben.

Modifikation des Vorgehens

Wie bereits erwähnt, müssen Anpassungen vorgenommen werden, damit der Prozess für die Projektgruppe einsetzbar ist. Zum einen wird das *Daily Scrum* durch das *Weekly Scrum* ersetzt, da ein tägliches Treffen im Rahmen des Projektes nicht möglich ist. Zum anderen wurde die *Sprint Retrospektive* nur nach Bedarf, und das *Sprint Review* in dem wöchentlichen Treffen mit den Betreuen abgehalten. Ebenso wurden die *Estimation Meetings* eins und zwei zusammengelegt und im Laufe der Projektphase verkürzt. Es wurden

die Methoden des *Magic Estimation* und des *Planning Poker* eingesetzt, um den Aufwand der *Product Backlog Items* einschätzen zu können. Im Verlauf des Projektes hat sich jedoch herausgestellt, dass *Magic Estimation* für die Projektgruppe besser geeignet ist. Auch wurde im weiteren Verlauf des Projektes auf die Ausformulierung von User Stories und das Pflegen und Erstellen eines *Product Backlogs* verzichtet. Beide Elemente bieten der Projektgruppe keine Vorteile und behinderten die Entwicklung. Der Einsatz von Jira als *Scrumboard* wurde hingegen genutzt und stellte sich als geeignet heraus.

2.4. Projekt-Roadmap

Die Projekt-Roadmap ist ein Plan, der ähnlich wie eine Karte aufgebaut ist und das Projekt in die wichtigsten, strategischen Schritte unterteilt. Mit Hilfe der Roadmap soll grafisch dargestellt werden, wie ein Projekt vom Start bis zum Ende umgesetzt wird.

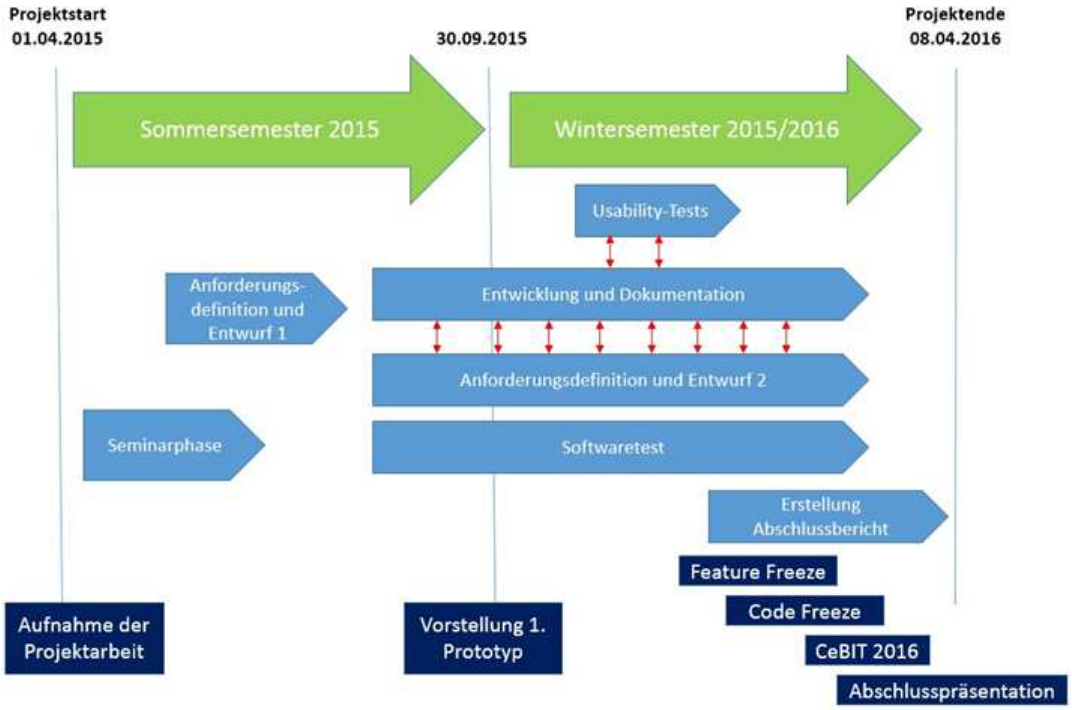


Abbildung 1: Projekt-Roadmap

Mit Beginn des *Sommersemesters 2015 (April 2015)* startete die Projektgruppe IMPACT. Zunächst wurden in einer Seminarphase wichtige Themen rund um das Innovationsmanagement, Vorgehensmodelle, Motivation, Konfliktmanagement oder Programmierkonventionen, erarbeitet, um eine gemeinsame Wissensgrundlage zu schaffen. In der darauffolgenden Phase wurden die ersten erforderlichen Anforderungen und Entwürfe erstellt. Auf dieser

Grundlage konnten die fünf großen Bereiche Entwicklung, Dokumentation, Anforderungsdefinition, weitere Entwürfe sowie die Softwaretests gestartet werden. Durch den agilen Prozess von Scrum, war es laufend möglich, neue Anforderungen sowie Ergebnisse der Usability-Tests, die in die Entwicklung integriert werden, einzubeziehen. Der erste Prototyp konnte somit bereits im *Herbst 2015* vorgestellt werden. Der zweite Prototyps wurde im *Dezember 2015* vorgestellt.
Anfang Februar begannen die ersten Vorbereitungen für die Abschlussdokumentationen, dazu wurden alle relevanten Themen gesammelt, diese entsprechenden Personen zugewiesen und daraus eine Gliederung erstellt.
Für den *26. Februar 201*6 wurde mit dem Feature Freeze ein weiterer großer Meilenstein erreicht. Ab diesem Zeitpunkt wurden keine neuen Funktionen mehr in das Product-Backlog geschoben, sondern nur noch bereits angefangene Implementierungen fertiggestellt.
Einen weiteren Meilenstein bildete der Code Freeze. Dieser wurde auf den *11. März 2016* datiert. Ab diesem Zeitpunkt wurden keine Veränderungen mehr am Quellcode vorgenommen. Mit der lauffähigen Prototypversion wurde ein Produktvideo für die CeBIT 2016 erstellt, welche in der folgenden Woche vom *14.-18. März*in Hannover stattfand. Weitere Informationen rund um den Messeauftritt sind im Anhang unter dem Kapitel CeBIT A.3.2 einzusehen.

2.5. Projekttagebuch

Um die Projektfortschritte, das Projektteam, die Ziele und Visionen, eine externe Präsentation und die Kontaktmöglichkeit zu bieten, wurde eine Website eingerichtet. Diese ist unter der URL *impact.informatik.uni-oldenburg.de* zu erreichen. Ein zentrales Element stellt dabei das Projekttagebuch dar, welches als Blog geführt wird. In diesem Blog wird über die wichtigsten Ergebnisse/Entscheidungen, besondere Termine oder Gruppenevents berichtet, um damit den Projektfortschritt zu dokumentieren und um für externe Interessenten die Möglichkeit zu verschaffen, sich über die Projektgruppe und Neuigkeiten zu informieren. Zusätzlich können die Externen über die Website mit der Projektgruppe in Kontakt treten.

2.6. Arbeitsgrundlagen

Für die reibungslose Zusammenarbeit in einem Projektteam mit elf Mitgliedern ist es erforderlich, dass eine Arbeitsgrundlage definiert wird, an die sich jedes Teammitglied halten muss. In diesem Kapitel werden dafür Regeln für die Zusammenarbeit (vgl. Abschnitt 2.6.1), Workshops zum Wissenaufbau (vgl. Abschnitt 2.6.2), einheitliche Programmierkonventionen (vgl. Abschnitt 2.6.3) und die einzusetzenden Werkzeuge (vgl. Abschnitt 2.6.4) festgelegt, an die sich die Mitglieder von IMPACT halten müssen.

2.6.1. Regeln für die Zusammenarbeit

Um mögliche Konflikte der Teammitglieder des Projektes auf ein Minimum zu reduzieren, wurden einige Regeln aufgestellt. Diese sind im Folgenden definiert: *Organisatorische Regeln:*

- Die Auswahl des Moderators der Treffen erfolgt in alphabetischer Reihenfolge. Das Protokoll wird dabei immer vom Moderator der darauffolgenden Woche angefertigt.
- Der Moderationsplan für das Gruppentreffen muss spätestens einen Tag vor dem Treffen, bis 16:00 Uhr, im Confluence zur Verfügung stehen. Dieser kann von allen Gruppenmitgliedern ergänzt werden.
- Das Protokoll der Gruppensitzung muss bis zum darauffolgenden Samstag um 18:00 Uhr im Confluence zur Verfügung gestellt werden.
- Ist ein Teilnehmer für ein Treffen verhindert, so kann sich dieser am Vortag des Treffens bis 18:00 Uhr abmelden (dies gilt nicht für kurzfristige Krankheitsfälle).
- Monatlich erhält der Kassenwart von jedem Gruppenmitglied einen Mitgliedsbeitrag in Höhe von 5 Euro, welcher für gemeinsame Aktivitäten und Anschaffungen eingesetzt wird.
- Urlaube müssen in den Terminkalender im Confluence eingetragen werden.

Diskussionsregeln:

- Die Diskussion zu bestimmten Punkten der Tagesordnung soll vom jeweiligen Thema nicht zu sehr abgleiten.
- Es kann jedes Thema diskutiert werden, welches von einem Gruppenmitglied vorgeschlagen wird.
- Alle Meinungen und Vorschläge der Gruppenmitglieder müssen akzeptiert werden.
- Es können alle Fragen gestellt werden.
- Die Diskussion soll immer sachlich bleiben und nicht persönlich werden.
- Der Redner einer Diskussion wird in seinem Wort nicht unterbrochen.

Allgemeine Regeln:

- Im Rahmen der Treffen sind Laptops/Notebooks nicht erlaubt. Ausnahmen bildet ein Vortragender, der Moderator der Sitzung, sowie der Protokollant.
- Smartphones haben ausgeschaltet bzw. lautlos eingestellt zu sein.

- Die Treffen werden pünktlich gestartet, sodass ein pünktliches Erscheinen aller Mitglieder vorausgesetzt wird. Eine Verspätung wird mit einer Geldstrafe von 1 Euro pro Minute belegt (bis max. 5 Euro).
- Bei Problemen jeglicher Art wird sich gegeneinander geholfen, um das Gruppenziel zu erreichen.

Auf Basis der Seminarbeit *Kommunikation und Konfliktmanagement* wurde zusätzlich ein Konfliktmanager bestimmt, der bei Schwierigkeiten unter den Teammitgliedern vermitteln kann (vgl. Seminararbeit *Kommunikation und Konfliktmanagement*). Da der Konfliktmanager aus den Reihen der Projektgruppe kommt, wurde zusätzlich der Scrum-Master als Stellvertreter eingesetzt.

2.6.2. Workshops zum Wissensaufbau

In der Projektgruppe befinden sich Teilnehmer mit diversen Erfahrungsstufen und verschiedenen Schwerpunkten im Studium. Um einen einheitlichen Wissensstand zu schaffen, wurde, vor der eigentlichen Umsetzung, eine Seminarphase durchgeführt. Innerhalb dieser Phase hat sich jeder Teilnehmer der Projektgruppe einem Thema gewidmet, welches im späteren Verlauf von Bedeutung sein könnte. Darunter waren Themen wie Gamification, Anreizsysteme, Programmierkonventionen, Entwicklungsstufen, Testmanagement, Dokumentation oder Oberflächendesign. Jedes der Themen wurde in einer jeweils 20-minütigen Präsentation vorgestellt und einer 10 bis 15-seitigen Seminararbeit festgehalten.
Für den Aufbau eines gleichen Wissensstandes, und als Basis für die spätere Programmierung, fand am 10. Juni 2015 ein Workshop statt, in dem die grundlegende Einrichtung der Software gezeigt und die geplanten Technologien, Vaadin und Maven, vorgestellt. Auf die einzelnen Technologien wird in Kapitel 4.4 im Detail eingegangen. Zudem wurden die Entwurfsmuster (Model-View-Controller bzw. Model-View-Presenter) wieder in Erinnerung gerufen. Das gemeinsame Einrichten hatte zum Vorteil, dass die Teilnehmer sich bei auftretenden Fehlermeldungen und Problemen gegenseitig unterstützen konnten. Am Ende des Workshops hatten alle Teilnehmer die Entwicklungsumgebung Eclipse (Version Luna) mit einem Vaadin-Projekt über Maven eingerichtet und die Verbindung zum Versionsverwaltungsprogrammes hergestellt. Mit dieser einheitlichen Basis konnte mit der technischen Umsetzung des Projektes begonnen werden.

2.6.3. Programmierkonventionen

Mit zunehmender Verbreitung von Software in alltägliche Anwendungsbereiche nimmt auch die Wichtigkeit ihrer zuverlässigen Funktionsweise immer weiter zu (vgl. [Lig01], S.2). Somit ist es auch für den Erfolg von IMPACT entscheidend, dass die Plattform

reibungslos funktioniert. Andernfalls sorgt Frust für sinkende Motivation und Vertrauensverlust auf Seite der Anwender.
Es ist deshalb wichtig, dass schon bei der Entwicklung des Quelltextes Mechanismen zum Einsatz kommen, die eine hohe Qualität gewährleisten. Je früher Fehler erkannt werden, desto geringer ist der Beseitigungsaufwand und die damit entstehenden Kosten(vgl. [Sch12], S.16).
Einen weiteren Grund stellt die Arbeitsteilung und Zusammenarbeit im Projektteam dar. Jeder hat sich seinen eigenen Programmierstil angeeignet. Damit spielt es keine Rolle, wie programmiert wird. Beim Aufeinandertreffen mehrerer Philosophien treten immer Probleme auf (vgl. [PJ14], S.23). Ein gemeinsamer Stil dagegen sorgt für konsistentere Quelltexte und bessere Verständlichkeit. Für den Leser sollte es keine Rolle spielen, wer den Quelltext verfasst hat (vgl. [Mar09], S.90).
Aus diesen Gründen wurden bereits zu Beginn des Projektes Regeln und Programmierpraktiken aufgestellt, die es beim Schreiben des Quelltextes zu beachten galt. Im Folgenden werden einige dieser Regeln näher erläutert. Detailliertere Informationen sind der Seminararbeit Programmierkonventionen zu entnehmen.

Sprache

Der Quelltext ist durchgehend auf Englisch verfasst, da Programmiersprachen meist auf englischen Begriffen beruhen und somit die Lesbarkeit erhöht wird (vgl. [PJ14], S.20ff.). Bei Kommentaren wird dagegen Deutsch verwendet.

Vertikale Formatierung

Der Einsatz von Leerzeilen kann die Lesbarkeit des Quelltext erheblich beeinflussen. Deshalb sollten logisch zusammengehörige Stellen nicht voneinander getrennt und gedanklich voneinander abzugrenzende Blöcke und Konzepte alleine stehen. Ebenso sollten Variablen nahe ihrer Verwendung deklariert werden. Gleiches gilt für sich gegenseitig aufrufende Funktionen. (vgl. [Mar09], S.76ff.)

Horizontale Formatierung

Zu lange Zeilen sollten vermieden werden, da sie die Lesbarkeit einschränken. 120 Zeichen pro Zeile sind das empfohlene Maximum. Leerzeichen sollten verwendet werden, um Argumente voneinander abzugrenzen, die nicht zusammen gehören. Dies sind z. B. die von Operatoren zu verarbeitenden Argumente. Bei einer starken Zusammengehörigkeit, wie z. B. den in Klammern stehenden Parametern eines Methodenaufrufes, macht es Sinn, keinen Zwischenraum zu verwenden. (vgl. [Mar09], S.85ff.)

Namensgebung

Namen für Variablen, Funktionen, Klassen etc. sollten verständlich, unterscheidbar, einheitlich, sprechbar, suchbar und direkt sein(vgl. [Mar09], S.18ff.) (vgl. [PJ14], S.33ff.). Notwendige Zusatzinformationen, wie bspw. die Einheit einer physikalischen Größe, sollten in den Namen einer Variable aufgenommen werden. Namen von Konstanten werden in nahezu allen Programmiersprachen in Großbuchstaben geschrieben.(vgl. [PJ14], S.38) Funktionsnamen sollten beschreiben, was eine Funktion tut und nicht, wie sie es tut. Dabei sollte der Name mit einem Verb beginnen, an das sich das behandelte Objekt anschließt(vgl. [PJ14], S.37ff.) Zu lange Funktionsnamen deuten darauf hin, dass die Funktion besser in weitere Unterfunktionen aufgeteilt werden sollte. Es sollte nicht versucht werden einen schöneren, abstrakteren oder einfacheren Namen zu finden, ohne den Inhalt der Funktion entsprechend anzupassen.(vgl. [PJ14], S.33f.)
In der objektorientierten Programmierung sollten Klassen immer mit Substantiven bezeichnet werden. Dabei hilft es, an Berufe wie zum Beispiel `Worker` oder `Writer` und Geräte wie zum Beispiel `Converter` oder `Adapter` zu denken. In den Namen sollte nicht mit aufgenommen werden, dass es sich um eine Klasse handelt. Bei der Benennung von Interfaces macht es Sinn, von Verben abgeleitete Adjektive einzusetzen, da alle Objekte, die ein Interface implementieren, bestimmte Handlungen durchführen können. Beispiele dafür sind `Comparable` oder `Runnable`.(vgl. [PJ14], S.52f.)

Kommentare

Kommentare sollten den Quelltext nicht eins zu eins wiedergeben, sondern den Gesamtkontext erklären und über Ideen, Gründe und Absichten aufklären (vgl. [PJ14], S.62ff.) Kommentare sollten direkt zusammen mit dem Quelltext und nicht erst zeitlich am Ende eingefügt werden. Noch hilfreicher ist es, sie vorher zu schreiben, um sich klarzumachen, was zu tun ist. Dabei geht es darum, warum der Quelltext etwas tut und nicht, was dieser tut.(vgl. [PJ14], S.67f.)
Gute Kommentare liefern zusätzliche Informationen über Dokumente, Copyrights oder Autorennamen. Sie beschreiben Absichten sowie bestehende Probleme oder erklären komplizierte und unklare Ausdrücke. Außerdem können sie Warnungen aufzeigen, auf Einschränkungen hinweisen oder zukünftige Pläne beschreiben.(vgl. [Mar09], S.55ff.) Viele Kommentare sind ein Zeichen für schlechten Quelltext. Ein gut geschriebenes Programm macht Kommentare überflüssig(vgl. [Fow05], S.82). Um die Einhaltung der festgelegten Regeln zu überprüfen, wurde eine Checkliste für regelmäßige Code-Reviews erarbeitet. Dieses Vorgehen hat außerdem den Vorteil, dass das Wissen im Projektteam weiter verbreitet wird. Die gemeinsame Überprüfung sorgt für Klarheit über den Quelltext und gibt anderen Mitgliedern die Gelegenheit, Vorschläge und Lösungsmöglichkeiten zu äußern (vgl. [Fow05], S.47).

2.6.4. Eingesetzte Tools

Im Folgendem werden die verwendeten Tools aufgezählt und erläutert.

Entwicklungsumgebung

Integrierte Entwicklungsumgebungen unterstützen den Anwender bei der Softwareentwicklung in unterschiedlichen Programmiersprachen. Sie vereinen eine Vielzahl von Werkzeugen, wodurch es ermöglicht wird, die einzelnen Tätigkeiten der Softwareerstellung wie Editieren, Übersetzen, Ausführen usw., miteinander und ohne einen Medienbruch auszuführen. So können bspw. zum Teil unvollständige Programme ausgeführt werden, um die Funktionalität zu testen oder den Programmcode während der Eingabe auf syntaktische Richtigkeit zu überprüfen. (vgl. [ES13], S.7f)
Für die Projektgruppe standen die drei Entwicklungsumgebungen Eclipse IDE [2], NetBeans IDE [3] und IntelliJ IDEA [4] zur Auswahl. Dies war der Fall, da Eclipse und NetBeans die am weit verbreitetsten Entwicklungsumgebungen für Java darstellen und ein Teammitglied über positive Erfahrungen mit IntelliJ berichtete (vgl. [CS14], S.642). Gegen NetBeans und IntelliJ sprach, dass bisher der Großteil der Projektgruppe über keine Erfahrungen mit den Programmen verfügte und somit eine längere Einarbeitungsphase daraus resultieren würde. Ferner sprach sich zudem ein Teammitglied gegen NetBeans aus, da es negative Erfahrungen mit dem Programm gesammelt hatte. So wurde schlussendlich Eclipse gewählt, da im Rahmen des Studiums schon jedes Teammitglied mit dieser Entwicklungsumgebung gearbeitet hatte und somit die Grundkenntnisse vorhanden waren.

Versionsverwaltung

Im Rahmen Softwareentwicklung entschließen sich die Teammitglieder der Projektgruppe dazu, ein Tool zur Versionsverwaltung einzusetzen. Hierbei ist eine Auswahl zwischen Apache Subversion (SVN)[5] und GIT[6] zu treffen. Weitere Alternativen zur Versionsverwaltung werden nicht berücksichtigt, da die erwähnten Tools zu den am weitesten verbreiteten Varianten zählen(vgl. [CS14], S.641). Der Unterschied zwischen den Werkzeugen besteht dabei im Umgang der verwalteten Daten. SVN speichert dabei mit jeder neuen Version lediglich die Änderungen an der entsprechenden Datei ab, wohingegen GIT den kompletten Zustand der Dokumente im Projekt abbildet (vgl. [BTH14] S.6). Aufgrund der unterschiedlichen Herangehensweisen ergeben sich aus Sicht der Projektgruppe unterschiedliche Vor- und Nachteile.

[2] https://eclipse.org
[3] https://netbeans.org
[4] https://www.jetbrains.com/idea/
[5] https://subversion.apache.org
[6] https://git-scm.com

Für den Einsatz von SVN spricht dabei, dass es innerhalb des Teams bekannt ist, mit TortoiseSVN[7] eine vertraute Benutzeroberfläche bietet und durch die Anzahl der gebotenen Funktionen weniger komplex erscheint.
GIT hingegen bietet mit seinem größeren Funktionsumfang mehr Operationen, wodurch es jedoch eine höhere Lernkurve besitzt. Dabei ist gerade die Möglichkeit des Erstellens und parallelen Bearbeitens auf unterschiedlichen Zweigen (so genannten Branches) für die Projektgruppe relevant. Ein Branch ermöglicht es dabei, verschiedene Funktionen unabhängig voneinander zu entwickeln (vgl. [CS14], S.43). Ein weiterer Vorteil von Git ist, dass sich die Dateien dezentral auf den Systemen der einzelnen Entwickler befinden (vgl. [CS14], S.4). Somit kann zum Einen offline weiterentwickelt werden, andererseits ist die Projektgruppe beim Ausfall des Servers davor geschützt, sämtliche Daten zu verlieren.
In einer Diskussion innerhalb der Gruppensitzung vom *16. April 2015* entschließt sich das Team dazu, Git für die Versionsverwaltung einzusetzen (vgl. Protokoll A.7.2). Zentrales Argument für diese Entscheidung ist dabei die Entwicklung verschiedener Funktionen auf unterschiedlichen Zweigen.

Modellierungstools

Ein Modellierungstool wird für die Modellierung von verschiedenen Diagrammen verwendet. Für die Softwareentwicklung werden zur Darstellung von Softwarebestandteilen und -zusammenhängen meist UML-Diagramme eingesetzt. Darüber hinaus kann es sinnvoll sein den Ablauf von Prozessen in ereignisgesteuerten Prozessketten (Ereignissgesteuerte Prozessketten) darzustellen.
Für die Modellierung von UML-Diagrammen, wie z. B. Sequenzdiagramme, wird *Visual Paradigm* verwendet. Es unterstützt alle gängigen UML-Diagrammelemente. Einige Projektmitglieder haben damit bereits in vorherigen Modulen ihres Studiums Erfahrungen gesammelt. Außerdem bestand die Möglichkeit eine Academic Licence zu erhalten, wodurch das Tool in vollem Funktionsumfang genutzt werden konnte.
Für die anfängliche Modellierung von Prozessen in der Konzeptionsphase wurde *Microsoft Visio* verwendet. Visio enthält die Elemente zur Erstellung von EPKs und erweiterten Ereignissgesteuerte Prozesskettens. Dadurch, dass sich die Bedienung zu bekannten anderen Microsoft Office Produkten nicht wesentlich unterscheidet und einige Projektmitglieder bereits mit diesem Werkzeug gearbeitet hatten, wurde die Entscheidung zur Verwendung von *Microsoft Visio* getroffen.

Continuous Integration

Unter Continuous Integration (CI) wird die regelmäßige Integration von Änderungen in

[7] https://tortoisesvn.net/index.de.html

eine Software verstanden. Bei der CI wird sowohl die Anwendung neu gebaut als auch, wenn eingerichtet, automatisiert getestet. Um die Anwendung auf Integrationsprobleme testen zu können, sollten die Änderungen funktionstüchtig sein. Dadurch wird gewährleistet, dass jederzeit eine lauffähige Version verfügbar ist. Der häufigste Anwendungsfall in CI sind die sogenannten Nightly Builds. Dabei wird der aktuelle Programmcode zu einer vorgegebenen Uhrzeit, häufig nachts, übersetzt und automatisiert getestet (vgl.[Beh11], S.13).

Jenkins als CI-Werkzeug Jenkins (ehemals Hudson) ist ein webbasiertes Open Source Continuous Integration System. Es wurde in Java entwickelt und ist somit plattformunabhängig. Jenkins unterstützt im Standard bereits zahlreiche andere Werkzeuge wie JUnit, GIT und Maven. Auch für weitere Programmiersprachen, wie PHP, Ruby oder .NET, ist Jenkins geeignet. Zudem lassen sich Testwerkzeuge und Plugins intuitiv über die Benutzeroberfläche integrieren (vgl.[Beh11], S.14).

3. Anforderungen

Das Kapitel der Anforderungen umfasst zunächst eine Beschreibung, wie die Analyse der Anforderungen aussieht. Dazu werden zunächst die eingesetzten Kreativmethoden vorgestellt und auf die Form der Anforderungserhebung eingegangen, bevor festgelegt wird, wie der Aufbau der Anforderungen im Abschnitt 3.1.5 aussehen soll. Anschließend erfolgt die Festlegung der funktionalen und nichtfunktionalen Anforderungen als Basis für die anschließende Entwicklung.

3.1. Anforderungsanalyse

Die Anforderungsermittlung ist ein Verfahren, um eine möglichst vollständige Sammlung von Anforderungen zu erstellen. Anforderungen sind Aussagen über Funktionen und Eigenschaften, die ein System beschreiben. Der Erfolg eines Softwareprojektes ist stark davon abhängig, in welchem Umfang die Kundenerwartungen erfüllt werden. (vgl. [Six16])
Bei der Anforderungsermittlung treten häufig folgende Probleme auf (vgl. [Hue16]):

- Heterogene Gruppen von Projektbeteiligten (Kunden, Spezialisten, Marketing, Management),
- Kunden/Benutzer können Anforderungen nicht genau spezifizieren,
- Verwendung unterschiedlicher Begrifflichkeiten,
- Begriffe werden unterschiedlich interpretiert und
- Neue und veränderte Anforderungen während der Analyse und Entwicklung.

Von dem Kunden wurden nur wenige Anforderungen im Vorfeld festgelegt. Aus diesem Grund hat die Projektgruppe sowohl die Rolle des Entwicklers, als auch die des Product Owners eingenommen. Durch festgelegte Begriffe und entsprechende Definitionen konnten Missverständnisse weitestgehend vermieden werden. In teaminternen Sitzungen wurden mit Hilfe von Kreativmethoden Funktionen und Eigenschaften des Systems ermittelt, siehe 3.1.1. Zusätzlich wurde ein Fragebogen konzipiert, um weitere Anforderungen von den Mitarbeitern der Lufthansa Industry Solutions zu identifizieren, siehe 3.1.2. In unregelmäßigen Abständen erfolgten weitere Interviews 3.1.3 mit dem Praxispartner, indem zusätzliche Anforderungen festgehalten wurden. Alle gesammelten Anforderungen wurden zunächst in den User Stories, inklusive Akzeptanzkriterien, festgehalten, siehe 3.1.4. Durch den agilen Entwicklungsprozess konnten, auch während der Entwicklung, zusätzliche Anforderungen in den Product Backlog aufgenommen werden. Im Kapitel der Anforderungen werden alle funktionalen (vgl. Kapitel 3.2) und nicht funktionalen Anforderungen (vgl. Kapitel 3.3) festgehalten.

3.1.1. Kreativmethoden

Als wesentliche Kreativmethode wurde das Brainstorming eingesetzt. Dabei werden (initiale) Ideen gesammelt und nach sachlichen oder qualitativen Anforderungen gegliedert. Diese Methodik bietet sich vor allem bei komplexen Strukturen oder Dienstleistungseinheiten an. Kurz nach Abschluss der Seminarphase wurde ein erstes Brainstorming vorgenommen. Dabei lag das Hauptaugenmerk auf die möglichen Funktionen und Bereiche, die in der Innovationsmanagementplattform umgesetzt werden sollen. Als informelle Technik hat sich die Projektgruppe für das Mind Mapping entschieden. Hierbei werden, von der Mitte aus, die Hauptäste zu den übergeordneten Themen gezeichnet. Innerhalb dieser Hauptäste können weitere Abzweigungen zu den Unterthemen erfolgen. Zu einzelnen Themengebieten, wie z.B. dem gemeinsamen Arbeiten (vgl. Abbildung 2), wurden im weiteren Verlauf der Projektzeit detailliertere Mind Maps erstellt. Über die Umsetzung der Unterpunkte wurde innerhalb der Projektgruppe diskutiert. Die ausgewählten Anforderungen wurden zunächst als User Stories 3.1.4 für den Scrum-Prozess formuliert.

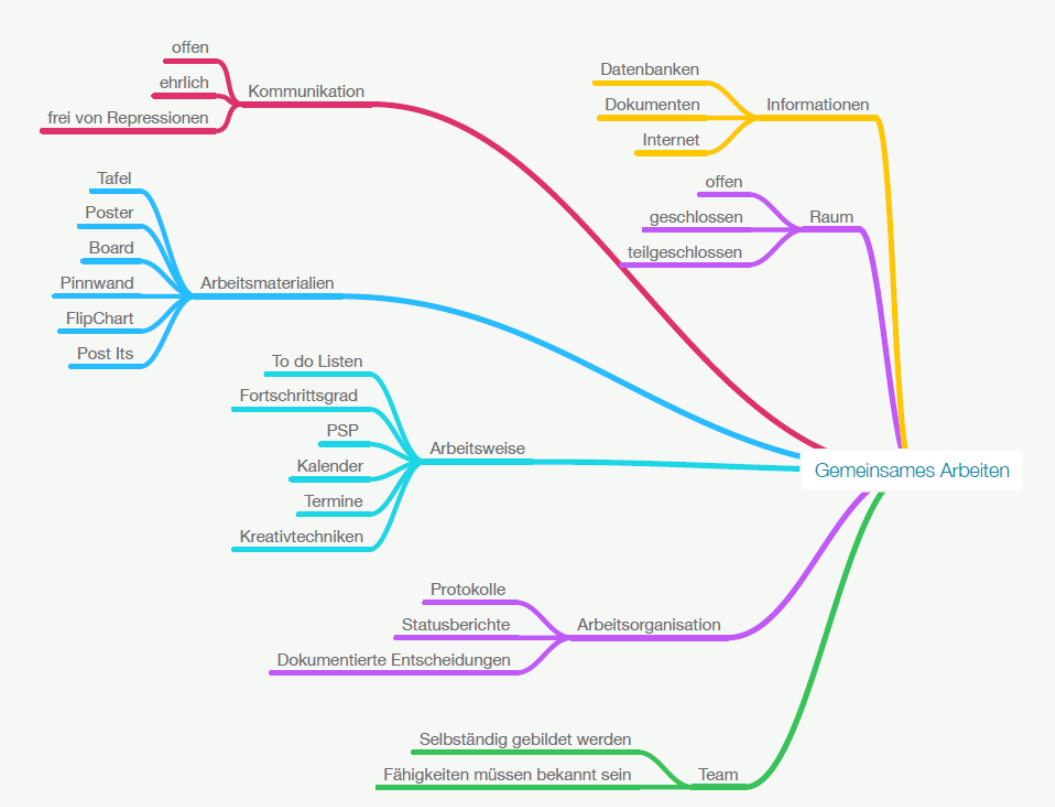


Abbildung 2: Mind Map zum Thema Gemeinsames Arbeiten

3.1.2. Fragebogen

Im Rahmen der Anforderungsanalyse wurde eine Online-Fragebogen konzipiert. Die Fragebogenmethode ist ein Untersuchungsmittel, das seinen Ursprung in den Sozialwissenschaften hat. Dieses Mittel dient dazu, dass durch die Beantwortung von offenen und geschlossenen Fragen neue Erkenntnisse gewonnen werden können (vgl. [MG14], S.15ff.) Die Umsetzung der Fragebogenmethode in Form einer Online-Umfrage wurde aus Gründen der Auswertbarkeit, Schnelligkeit und maximalen Reichweite gewählt. Im Rahmen der Vorbereiteten Maßnahmen wurden vier Themenkomplexe identifiziert. Der erste Komplex beinhaltet allgemeine Einführungsfragen. Diese Fragen wurden verwendet, um einen leichten Einstieg in die Umfrage zu gewährleisten. Im zweiten und dem dritten Komplex wurden Fragen zu funktionalen und nicht nicht funktionalen Anforderungen gestellt. Dieser Abschnitt war das Kernelement der Online-Umfrage, aus dessen Antworten ein Großteil der Anforderungen generiert werden sollten. Im letzten Themenkomplex wurden persönliche Fragen gestellt. Zusätzlich wurde nach der Bereitschaft zu einem persönlichen Interview gefragt. Insgesamt wurden 20 Fragen konzipiert, welche in einem Zeitraum von ungefähr fünf bis zehn Minuten beantwortet werden konnten. Umfrage wurde im Zeitraum vom *13. Juli 2015* bis zum *27. Juli 2015* durchgeführt. Insgesamt wurden neun vollständige Antworten registriert, was eine Mitarbeiterbeteiligung von ca. *12%* darstellte. Die geringe Anzahl an Antworten führte dazu, dass die Ergebnisse für die Anforderungsanalyse nicht nutzbar waren. Aus ihnen konnten lediglich Tendenzen abgeleitet werden. Die wesentlichen Ergebnisse der Umfrage werden im Folgenden kurz beschrieben.
Drei entschiedene Punkte konnten der Online-Umfrage entnommen werden. In Abbildung 3 wird die Bedeutung ausgewählter Funktionalitäten für ein Innovationsmanagementsystem bewertet. Das Ergebnis zeigt, dass insbesondere ein intuitives Oberflächendesign, die Nachvollziebarkeit der Entscheidungen, Diskussionsmöglichkeiten, das Versenden von Privatnachrichten und ein kollaborativer Raum eine hohe Bedeutung zu teil wird.

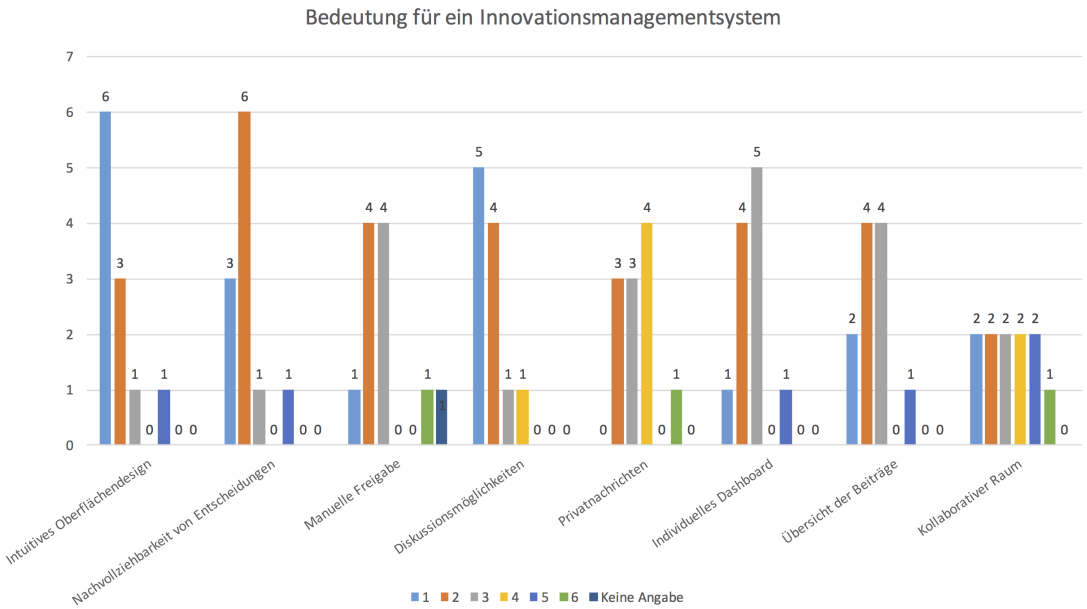


Abbildung 3: Bedeutung der Funktionalitäten

Ein weiterer entscheidender Punkt, war die Frage nach der Möglichkeit Beiträge anonym bewerten und kommentieren zu können. Die Antworten auf diese beiden Fragen können in der Abbildung 4 betrachtet werden. Ein großer teil der Befragten sprach sich für die Möglichkeit der anonymen Abgabe aus.

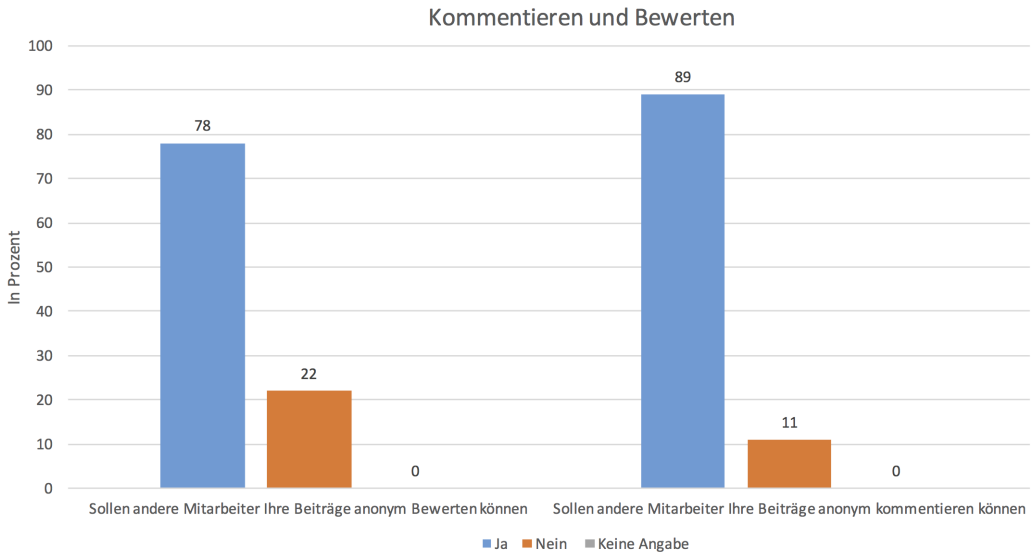


Abbildung 4: Kommentieren und Bewerten

Der letzte Punkt, welcher anhand der Umfrage beantwortet werden konnte. War die Frage nach einem gemeinsamen Arbeiten an einer Idee. Die überwiegende Meinung war, dass ein kollaboratives arbeiten gewünscht ist. Die Ausprägungen der jeweiligen Fragen können der Abbildung 5 entnommen werden.

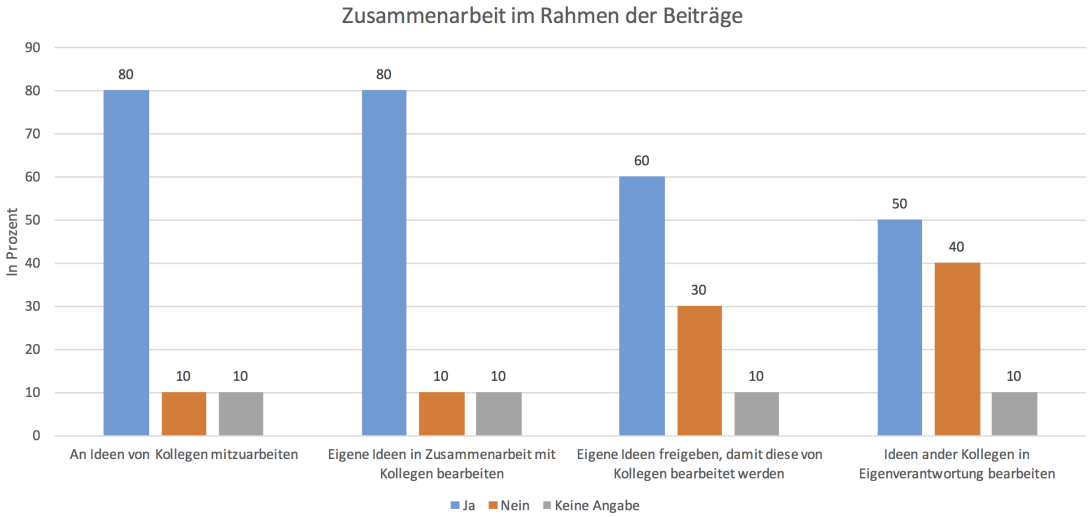


Abbildung 5: Arbeiten im Team

Die Ergebnisse der Umfrage waren aufgrund der geringen Teilnahme nur von zweitrangiger Bedeutung. Jedoch konnten einige Einblicke gewonnen werden, welche sich in den nachfolgenden Anforderungen Wiederspiegeln. Im nachfolgenden Kapitel 3.1.3 wird eine weiter Methode beschrieben mit der im Rahmen dieses Projektes Anforderungen ermittelt wurden.

3.1.3. Interviews

Als Product Owner und Ansprechpartner auf Seiten der Lufthansa Industry Solutions wurden auch einige Gespräch mit Dr. Joachim Kurzhöfer bezüglich der Anforderungen und Funktionalitäten geführt. Die Gespräche folgten in der Regel immer folgendem Aufbau.

- Aktueller Stand der Projektgruppe,
- Fragen aus der Projektgruppe,
- Intentionen aus Sicht der Lufthansa und
- Priorisierung zukünftiger Aufgaben.

Vor allem die Denkprozesse und Handlungsweisen aus Sicht der Lufthansa haben den Projektverlauf beeinflusst. Die Intentionen hinter den jeweiligen Entscheidungen wurden klar kommuniziert, ohne die Projektgruppe jedoch bei der Umsetzung einzuschränken. So war jederzeit das Ziel deutlich, den Weg dorthin konnten wir aber selber bestimmen.

3.1.4. User Stories

In sogenannten User Stories werden die Anforderungen an ein System aus Nutzersicht dokumentiert. Eine User Story besteht dabei aus drei grundlegenden Elementen (vgl. [HvB16]):

- Einem kurzen und eindeutigen Namen,
- Einer Beschreibung der Anforderung und
- Mehrere Akzeptanzkriterien, die sowohl Details beinhalten als auch bei der Klärung helfen, ob eine User Story abgeschlossen ist.

Als Grundgerüst einer User Story hat sich der folgende Aufbau in der Praxis bewährt:
Als *Anwendertyp* möchte ich folgende *Aktion* durchführen, um dieses *Ziel* zu erreichen.
Eine gute User Story erfüllt zudem die sogenannten INVEST-Kriterien (vgl. [LPKS14]):

- Independent – Eine User Story ist unabhängig von anderen User Stories.
- Negotiable – User Stories sind kein in Stein gemeißeltes Gesetz, sondern Kunde und Entwickler besprechen und präzisieren diese gemeinsam.
- Valuable – Die User Story sollte einen erkennbaren Mehrwert für das Produkt haben.
- Estimatable – Eine User Story muss im zeitlichen Kontext schätzbar sein (Aufwand).
- Small – Über den konkreten Umfang einer User Story entscheidet letztlich das Team. Als Richtlinie sollte eine User Story mindestens einen halben Personentag und maximal zehn Personentage erfordern.
- Testable – Der Abschluss einer User Story muss überprüfbar sein, um beurteilen zu können ob sie erfolgreich abgeschlossen wurde.

Anhand der beiden exemplarischen User Stories für die Challenge-Erfassung 1 und dem Dateiupload 2 soll der Aufbau verdeutlicht werden. Der vollständige Anforderungskatalog ist dem Anhang A.5 zu entnehmen.

Kritierium	Ausführung
Name der Story	Challenge erfassen
Beschreibung	Als Benutzer möchte ich auf der Plattform meine Challenge erfassen, um diese anderen Benutzern zugänglich zu machen
Akzeptanzkriterien:	Ich kann meine Challenge auf der Plattform mit Hilfe eines Formulars eintragen Meine Challenge wird gespeichert Andere Benutzer können meine Challenge aufrufen

Tabelle 1: Userstory für „Challenge erfassen“

Kritierium	Ausführung
Name der Story	Datei-Upload
Beschreibung	Als Benutzer möchte ich Dateien hochladen, um diese anderen Benutzern zur Verfügung zu stellen.
Akzeptanzkriterien:	Ich kann verschiedene Datei-Typen hinzufügen (*.pdf, *.docx, *.xlsx) Ich kann mehrere Dateien hochladen Ich kann mehrere Dateien zeitgleich hochladen (Multi-Upload) Die Dateien werden gespeichert und sind auch für andere Benutzer abrufbar

Tabelle 2: Userstory für „Dateiupload“

3.1.5. Anforderungsaufbau

Der Aufbau der funktionalen und nicht-funktionalen Anforderungen ist an [Rup14] angelehnt. Die wichtigsten Elemente dieser Vorlage sind der folgenden Abbildung 6 zu entnehmen.

Abbildung 6: Beispielsatz für Formulierung der Anforderungen

Bezüglich der Formulierung sind vor allem die rechtlichen Verbindlichkeiten von besonderer Bedeutung.

- Attribut „muss“: Anforderungen müssen verpflichtend umgesetzt werden und können rechtlich eingefordert werden.
- Attribut „wird“: Anforderungen müssen in der Zukunft verpflichtend umgesetzt werden und können rechtlich eingefordert werden.
- Attribut „sollte“: Anforderungen müssen nicht verpflichtend umgesetzt werden und können rechtlich nicht eingefordert werden. Eine Erfüllung dieser Anforderungen erhöht die Zufriedenheit der Auftraggeber.

Der Akteur steht für die entsprechende Rolle, für die die Anforderung festgelegt wird (z.B. Anwender, Administrator, etc.). Im Objekt wird der Gegenstand der Anforderungen beschrieben und das Prozesswort ist ein Verb. Die Verwendung dieser Schablone wird an Hand folgendem Beispiels 7 verdeutlicht.

Abbildung 7: Beispielsatz für Formulierung der Anforderungen

3.2. Funktionale Anforderungen

Funktionale Anforderungen beschreiben was ein System leisten soll. Sie beschreiben u. a. Funktionen, Ein- und Ausgaben, sowie Verhalten bei bestimmten Situationen. (vgl. [Sch16])

3.2.1. Systemadministration

Die Anforderungen im Bereich der Systemadministration befassen sich mit der Installation, der Systempflege (Betrieb, Wartungen und Support), die technische Dokumentation, Konfiguration- und Erweiterungsmöglichkeiten (Skalierbarkeit, Portabilität etc.), sowie die Anforderungen an die Datensicherheit (vgl. [BRS06], S.157). Dazu sind die folgenden Anforderungen an IMPACT gestellt:

- Der Anwender muss das System ohne eine Installation auf dem Endgerät nutzen können.

- Das System muss eine Installationsanweisung und ein Benutzerhandbuch bereitstellen.
- Die Datenbank auf dem Server sollte einer täglichen Datensicherung unterzogen werden.
- Das System muss die Möglichkeit bieten verlorene Daten wiederherzustellen.

Rollenkonzept

Die zentrale Anforderungen des Rollenkonzeptes besteht darin, die Anwender zu verwalten. Dazu können die folgenden Anforderungen aufgestellt werden:

- Das System darf nicht allen Anwendern die kompletten Umfang bereitstellen.
- Das System muss raumselektiv gestaltet werden.
- Das Anwender muss sich einen Basisnutzer selbst erstellen können.
- Das System sollte ein Backend für die Verwaltung von Anwendern zu Verfügung stellen.
- Die Anwender müssen in Anwendergruppen unterteilt werden können.
- Das System muss unterschiedliche Rechte in einem Bereich ermöglichen.

Registrierungsprozess

Die Registrierung zählt zu den notwendigen Kernfunktionalitäten des Systems und muss gewissen Anforderungen entsprechen. Die folgenden Anforderungen sind von dem System umzusetzen:

- Der Anwender muss sich im System registrieren können.
- Der Anwender muss die Registrierung nur einmalig durchführen können.
- Der Anwender muss Vornamen, Namen, E-Mailadresse und ein Passwort für die Registrierung eingeben.
- Der Anwender sollte weitere Angaben, wie Tätigkeitsbeschreibung und Abteilung, bei der Registrierung eingeben können.
- Einem Administrator sollte es möglich sein neue Anwender in einem Backend einzutragen.
- Sollte ein Backend umgesetzt werden, soll der Anwender nicht mehr in der Lage sein, sich selbst in dem System zu registrieren.

Umgang mit Passwörtern

Die Sicherheit von personenbezogenen Daten ist eine der wichtigsten Funktionen in einer Software-Lösung. In diesem Kontext ist es für die Benutzer von besonderer Bedeutung, dass ihre Passwörter sicher übertragen und gespeichert werden. Folgende Anforderungen wurden ermittelt.

- Das System muss fähig sein das Passwort verschlüsselt zu übertragen.
- Das System muss fähig sein das Passwort verschlüsselt zu speichern.
- Das System muss die Möglichkeit bieten das Passwort zu ändern.
- Falls der Benutzer sein Passwort vergisst, muss das System fähig sein das Passwort zurückzusetzen.

Protokollierung (log-file)

Auch wenn das System ausgiebig getestet wird, lassen sich Systemstörungen nicht vollständig verhindern. Um diese möglichst gut beheben zu können, ist zunächst eine vollständige Dokumentation in Form von Protokollierung notwendig. Folgenden Anforderungen wurden für die Protokollierung ermittelt.

- Das System muss fähig sein Systemstörungen zu protokollieren.
- Das System wird fähig sein eine individuelle Auswertung der Systemstörungen zu erstellen.
- Das System soll fähig sein bei festgelegten Systemstörungen automatisch Folgeaktionen auszuführen.

Navigation und URL-Parameter

Mit Hilfe von URL-Parametern werden die verschiedenen Ansichten gesteuert. Um diese Ansichten auch direkt im Browser aufrufen zu können, ist eine Eingabe von URL-Parametern notwendig.

- Das System muss die Möglichkeiten bieten Ansichten per URL-Parameter aufzurufen.
- Das System muss dem Anwender eine zentrales Menü mit den wichtigsten Funktionen bereitstellen.
- Der Anwender muss die Challenge von einem zentralen Erreichen können.
- Das System muss über Verlinkungen verfügen, damit der Anwender einfach zwischen Punkten springen kann.
- Das System muss Übersichten bereitstellen, um Eintragungen zusammenzufassen.

Validierung von Benutzereingaben

„Die Validierung zielt darauf ab, die richtigen Dinge zu tun (doing the things right). Die Validierung prüft die Ergebnisse daraufhin, ob das spätere Ergebnis den angenommenen Nutzen bringt“([MR16], S.120). Für die Anforderungen an die Validierung heißt dies, dass eine Überprüfung der Anwendereingaben erfolgt. Im Bereich der Registrierung bedeutet es, dass die Eingabe der E-Mail-Adresse und des Passworts überprüft werden muss. Der Anwender muss bei der E-Mail-Adresse ein @-Zeichen eingeben. Beim Passwort muss der Anwender eine achtstellige Kombination aus Ziffern und Zahlen eingeben. Schlägt die Validierung fehl, dann muss der Anwender vom System eine begründete Rückmeldung erhalten.

Für die Validierung der Benutzereingaben sind die folgenden Anforderungen gegeben:

- Schlägt die Validierung der Eingaben fehl, so muss das System dem Anwender eine Rückmeldung geben.
- Das System muss fähig sein zu überprüfen, ob die Eingabe des Anwenders bereits im System vorhanden ist.
- Das System muss fähig sein doppelte Eingaben, wie bei dem Passwort, auf Gleichheit zu überprüfen.

Fehlerbehandlung

Neben den systemseitigen Fehlern sollte die Plattform auch die wesentlichen Fehlerquellen auf Seiten der Benutzereingaben abfangen. Dieses kann z.B. durch Überprüfung und Validierung der Eingaben erfolgen. Folgende Anforderungen wurden definiert.

- Das System sollte fähig sein Fehleingaben abzufangen.
- Bei bekannten Fehlern sollte das System fähig sein spezifische Fehlermeldungen auszugeben.
- Bei unbehandelten Fehlern sollte das System fähig sein eine vorher definierte Fehlermeldung auszugeben.
- Das System soll fähig sein bei festgelegten Fehlermeldungen automatisch Folgeaktionen auszuführen.

Cross-Plattform

Mobile Endgeräte finden in der heutigen Gesellschaft eine rasant wachsende Beliebtheit. Doch nicht nur in der Freizeit, sondern auch im Berufsalltag kommen sie zunehmend zum Einsatz. Das System sollte daher auch für die mobile Nutzung optimiert sein. Folgende Anforderungen wurden ermittelt.

- Die essentiellen Funktionen des Systems müssen auf Smartphone, Tablet und PC/Notebook lauffähig sein.
- Das System sollte auf unterschiedlichen Browsern lauffähig sein.
- Das System muss fähig sein die Elemente entsprechend der Auflösung skalieren zu können.

3.2.2. Personalisierung

Die Personalisierung des Systems umfasst die Anforderungen, die dem Anwender individuelle Einstellungen ermöglichen. Die Personalisierung der Plattform trägt dazu bei, dass sich die Nutzer mit dieser besser identifizieren und einen Bezug dazu aufbauen. Die Identifizierung hat zur Folge, dass die Plattform von den Nutzern akzeptiert wird und somit aktiver genutzt wird. Dazu soll es persönliche Designs geben, sodass sich der Nutzer ein Design nach dem persönlichen Geschmack auswählen kann. Zusätzlich soll es persönliche Nachrichten (Notifications) und einen Aktivitätsstream gegeben, der alle Änderungen an der Plattform zeigt. Auf diese Punkte wird in den folgenden Unterkapiteln genauer eingegangen. Eine weitere Form der Personalisierung umfasst ein Berechtigungskonzept, dabei erhält jeder Nutzer individuelle Berechtigungen, die es ihm ermöglicht, für ihn freigegeben Funktionen angezeigt und nutzen zu können.

- Das System muss dem Anwender individuelle Einstellungsmöglichkeiten bieten.
- Der Anwender muss das Design an seinen persönlichen Geschmack anpassen können.
- Der Anwender muss die Möglichkeit haben, persönliche Informationen einzutragen.
- Das System sollte dem Anwender ermöglichen Beträge nach persönlichen Interessen zu filtern.
- Das System sollte dem Anwender eine Ansicht bereitstellen, wo dieser seine Einträge sehen kann.

Verschiedene Designs

Eine Möglichkeit der Personalisierung liegt in der Anpassung des Designs an die persönlichen Vorlieben des Benutzers. Dazu soll dem Anwender ermöglicht werden, dass dieser Hintergrundbilder und ähnliches schnell und einfach anpassen kann. Die Anpassungsmöglichkeit trägt dazu bei, dass sich der Nutzer mit der Plattform identifiziert und somit aktiver an dieser arbeitet. Dies soll zusätzlich unterstützt werden, indem das Design der Plattform sich von den alltäglichen Programmen abhebt und somit kein direkter Bezug zu den alltäglichen Aufgaben hergestellt wird. Dies ermöglicht eine kreativere und konstruktive

Arbeitsumgebung.
Die Einstellung des individuellen Designs sollte an einer zentralen und leicht erreichbaren Stelle wie dem Benutzerprofil befinden.

- Das System sollte dem Anwender ermöglichen die Hintergrundbilder an persönliche Vorlieben anzupassen.
- Das System muss sich im Design von alltäglichen Anwendungen der Anwender unterscheiden.
- Der Anwender muss die Anpassungen des Design an einer zentralen Stelle im System vornehmen können.
- Der System muss dem Anwender min. drei Design zur Vorfügung stellen.

Activity-Stream

Der Aktivitätenstream stellt weitere zentrale Anforderungen an die Innovationsplattform dar. Folgende Inhalte sollen dabei eine essenzielle Rolle einnehmen:

- Das System muss über einen Aktivitätenstream verfügen, der Änderungen und neue Beiträge anzeigt.
- Der Aktivitätenstream muss sich an einer zentralen Stelle im System befinden.
- Die Anzeige der Aktivitäten muss dabei in chronologischer Reihenfolge erfolgen.
- Der Eintrag im Aktivitätenstream muss den Zeitpunkt, Anwender und die Stelle der Eintragung als Information enthalten.
- Der Eintrag muss mit dem jeweiligen Eintrag verlinkt werden.

Notifications

Eine weitere Anforderung im Bereich der Personalisierung, besteht in den Nachrichten (Notification). Dem Anwender wird ermöglicht individuelle Benachrichtigungen zu erhalten, die nur für den jeweiligen Anwender sichtbar sind. Dies umfasst zum einen die Veränderungen bei den eigenen Beiträgen. Zudem soll dem Nutzer auch die Möglichkeit gegeben werden, dass dieser sich bestimmt, für ihn relevante Bereiche oder Themen abonnieren kann, sodass dieser bei Veränderungen informiert wird und die Änderung einsehen kann. Dies ist besonders für Nutzer sinnvoll, die nicht jeden Tag ins System gucken und bspw. im Urlaub sind. Diese müssen dann nicht den schnellwechselnden Aktivitätenstream komplett durchsuchen, sondern erhalten die persönliche, relevanten Nachrichten an einer zentralen Stelle.

- Der Anwender soll die Möglichkeit haben, dass er für sich relevante Bereiche abonnieren kann.
- Der Anwender soll über Beträgen zu seinen Einträgen im Aktivitätenstream informiert werden.
- Das System soll die abonnierten Einträge an einer zentralen Stelle bereitstellen.
- Das System muss dem Anwender nur die individuellen Nachrichten darstellen.
- Das System soll den Anwender nicht mit Nachrichten nicht überlasten.
- Das System soll die persönlichen Einträge im Aktivitätenstream kennzeichnen.

3.2.3. Gamification

In diesem Abschnitt wird zunächst der Begriff *Gamification* im Rahmen dieser Dokumentation definiert. Im Anschluss werden die Anforderungen an unser System festgelegt, bevor weitere mögliche Elemente aus dem Gamification-Umfeld vorgestellt werden.

Definition

Gartner definiert Gamification als „[...] the use of game mechanics and experience design to digitally engage and motivate people to achieve their goals“(vgl. [Bur14]). Die Schlüsselelemente sind dabei folgende (frei übersetzt) (vgl. [Bur14]).

- Unter den *game mechanic*s werden Elemente wie z.B. Punkte, Abzeichen und Ranglisten zusammengefasst.
- *Experience design* beschreibt die Reise des Spielers, die durch Elemente beeinflusst werden wie z.B. das Gameplay und die Story-Line.
- *Digitally engage* bedeutet, dass die Spieler über Computer, Handys und sonstige digitale Geräte zugreifen und weniger mit anderen Personen direkt interagieren.
- Das Ziel der Gamification ist es die Benutzer zu motivieren ihr Verhalten zu ändern, Fähigkeiten zu entwickeln oder Innovationen voranzutreiben.
- Und schlussendlich sollen die Spieler ihre Ziele erreichen. Wenn Ziele der Organisationen mit denen der Spieler zusammenhängen, dann erreicht die Organisation ihre Ziele, wenn die Spieler ihre Ziele erreichen.

Zunächst einmal soll Gamification aus Betriebssicht den Wert eines Gesamtzieles maximieren. Das Problem im Unternehmenskontext ist, dass Personen unterschiedlich auf dieselben Elemente reagieren.

Dieses wird als Gamification Design Problem beschrieben (vgl. [MJ14]). Da es im zeitlichen Kontext der Projektgruppe nicht möglich war eine wissenschaftliche Auswertungen zu einzelnen Elementen zu erheben, werden auf der Plattform nur grundlegende Elemente verwendet. Diese werden in folgenden Anforderungen festgehalten.

- Das System muss fähig sein Bestenlisten darzustellen.
- Das System muss dem Benutzer die Möglichkeit bieten, sein Benutzerprofil individuell anzupassen.
- Das System muss fähig sein einen Fortschrittsbalken anzuzeigen.
- Das System sollte fähig sein Ränke an Hand von Erfahrungspunkten zu vergeben.
- Das System sollte fähig sein virtuelle Güter zu tauschen.
- Das System sollte dem Benutzer die Möglichkeit bieten positives Feedback zu erfassen.

Weitere mögliche Elemente

Neben den festgelegten Anforderungen gibt es noch eine Vielzahl weiterer möglicher Gamification-Elemente für das System. Um diese sinnvoll in die Plattform zu integrieren, bedarf es weiterer Untersuchungen und Auswertungen. Es folgt eine Aufzählung komplexerer Elemente, die für das System in Frage kommen könnten.

- Questsystem: Der Benutzer bekommt Aufgaben gestellt, die er in einem festgelegten zeitlichen Horizont absolvieren kann. Die Komplexität der Aufgabe, als auch der Wert der Belohnung, sollte im Verlauf steigern.
- Community Collaboration: Einige Aufgaben lassen sich nur in Zusammenarbeit mit anderen Benutzern lösen. Dieses führt dazu, dass mehr Kontakte geknüpft und gepflegt werden.
- Easter Egg: Versteckte Besonderheiten in dem System, wie z.B. Geheimaufgaben oder besondere Belohnungen, soll die Abenteuerlust der Benutzer wecken.

3.2.4. Design Thinking

In diesem Abschnitt wird zunächst der Begriff *Design Thinking* für den Rahmen dieser Arbeit definiert, und im weiteren Verlauf die möglichen Elemente erläutert und Anforderungen definiert.

Definition

Der Begriff *Design Thinking* ist nicht in der Literatur eindeutig definiert es wird als Mindset oder Prozess oder als beides verstanden (vgl. [HPI15]). Einigkeit besteht darüber, dass *Design Thinking* etwas was ist was unter Zusammenarbeit von verschiedenen Personen in einem Team um neue Ideen zu entwickeln (vgl. [HPI15]). Im Rahmen dieser Arbeit wird *Design Thinking* als Design Thinking als Vorgehensmodell zur Problemlösung angesehen, das durch verschiedene Phasen iterativ durchlaufen wird. Innerhalb der zahlreichen Ausprägungen mit unterschiedlichen Prozessschritten und Bezeichnungen ist ein wesentliches Konzept zu erkennen (vgl. [Win11], S.24). Dieses vereinfachte Prozessmodell lässt sich in zwei Bereiche aufteilen, die jeweils in konvergente und divergente Phasen geteilt werden können. Die Abbildung 8 zeigt dieses Phasenmodell, welches im Folgenden erläutert wird.

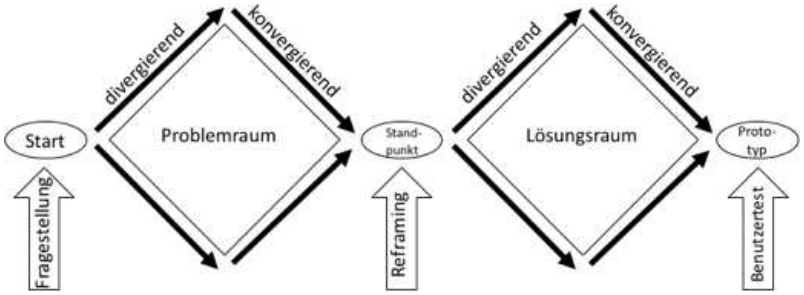


Abbildung 8: Das Phasenmodell zeigt das Zusammenspiel von divergierenden und konvergierenden Denkprozessen, adaptiert nach [Rai14], S.179

Im ersten Teilbereich dem *Problemraum explorieren* wird dabei in der divergenten Phase versucht so viele Daten wie möglich über das Problem zu erheben. In der darauffolgenden konvergenten Phase werden die erhobenen Daten auf tiefer gehende Probleme analysiert und selektiert. Nach Winkler folgt als nächstes ein *reframing* bei dem das Problem anhand der gewonnenen Informationen neu definiert wird, sowie die Aufstellung von Theorien und Hypothesen (vgl. [Win11], S.25). Ziel ist neben der Gewinnung von neuen Erkenntnissen die Festlegung der Rahmenbedingungen für die möglichen Lösungen.
Im zweiten Teilbereich dem *Lösungsraum explorieren* werden in einer weiteren divergenten Phase möglichste viele Ideen und Lösungen dazu generiert. Ziel ist es das Problem aus möglichst vielen Perspektiven zu betrachten um somit eine optimale Lösung zu finden (vgl. [Win11], S.25). In der anschließenden konvergenten Phase werden die zuvor entwickelten Ideen auf ihre Umsetzbarkeit geprüft. Verschiedene Lösungen werden dabei kombiniert um möglichst viele Aspekte abzudecken (vgl. [Win11], S.25). Es erfolgt eine Reduktion auf nur wenige Ideen, die daraufhin weiterverfolgt und getestet werden.

Mögliche Elemente

Design Thinking ist in seiner Gesamtheit nicht in der Plattform abzubilden, aber es ist möglich die wesentlichen Elemente durch die Gestaltung der Plattform und dem Einsatz von im *Design Thinking* üblichen Methoden zu unterstützen. Im Vordergrund steht dabei die Zusammenarbeit, der Wechsel von divergenten und konvergenten Phasen und letztendlich der Einsatz von Kreativ-Methoden wie Brainstorming und Mind Mapping. Als Anforderung ergeben sich daraus:

- Das System soll Anwendern die Möglichkeit bieten gemeinsam an einer Challenge zu arbeiten.
- Das System soll Anwendern die Möglichkeit bieten mehrere Lösungsvorschläge zu einer Challenge einzureichen.
- Das System soll Anwendern die Möglichkeit bieten mehrere Lösungsvorschläge als eigenständige Projekte durchzuführen.
- Das System soll Anwendern die Möglichkeit bieten mehrere Lösungsvorschläge zu einem neuen Lösungsvorschlag zu kombinieren.
- Das System soll Anwendern die Möglichkeit bieten gemeinsam an einer Projekt zu arbeiten.
- Das System soll Anwendern die Möglichkeit bieten Kreativ-Methoden durchzuführen.

3.2.5. Weitere Anforderungen

Unter weitere Anforderungen sind zusätzliche Features, die nicht zu den Hauptfunktionalitäten der Plattform gehören, zusammengefasst.

Bewertungsfunktion

Mit der Bewertungsfunktion sollen Challenges und Lösungsvorschläge bewertet werden. Diese werden somit messbar gemacht. Außerdem werden die Meinungen von anderen Mitarbeitern auf dem Portal übersichtlich dargestellt. Das hilft bei der Entscheidung, ob eine Challenge zu einem Projekt umgewandelt werden soll und welcher Lösungsvorschlag zur Bewältigung der Challenge eingereicht werden soll. Der dafür verwendete Bewertungsalgorythmus soll einfach sein und sowohl für die Challenge als auch für den Lösungsvorschlag einheitlich sein. Die abgegebene Bewertung soll begründet werden.

- Das System muss die Möglichkeit bieten Bewertungen für Challenges und Lösungsvorschläge abzugeben.

- Das System muss die Möglichkeit bieten alle Bewertungen einzusehen.
- Das System soll die Möglichkeit bieten Bewertungen anonym zu erfassen.
- Die Bewertungsfunktion soll einen nachvollziehbaren Bewertungsalgorithmus besitzen.
- Das System soll die Möglichkeit bieten Bewertungen zu begründen.

Kommentarfunktion

Angemeldete Benutzer können mit der Kommentarfunktion Challenges, Lösungsvorschläge und Projekte kommentieren. Außerdem sind diese in der Lage alle abgegebenen Kommentare von anderen Benutzern des Portals einzusehen. Challenges, Lösungsvorschläge und Projekte können mit Hilfe der Kommentarfunktion diskutiert werden. Den Challenge-, Projekt- oder Lösungsvorschlagerstellern wird auf diese Weise weitere oder ergänzende Informationen mitgeteilt. Dem Ersteller steht dann frei, ob er die Informationen übernimmt oder nicht.

- Das System muss die Möglichkeit bieten Kommentare für Challenges, Lösungsvorschläge und Projekte zu erstellen.
- Das System soll die Möglichkeit bieten Kommentare anonym zu veröffentlichen.
- Das System soll die Möglichkeit bieten alle Kommentare einzusehen.

Tag-Cloud

Beim Erstellen von Challenges und Lösungsvorschlägen können Tags angegeben werden. Tags sind kurze Schlagwörter mit denen etwas beschrieben werden kann. Tags sollen in einer Tag-Cloud angezeigt werden. Eine Tag-Cloud ist eine Stelle, in der alle Tags von Challenges und Lösungsvorschlägen zusammengefasst werden. Je nach Häufigkeit eines Tags werden diese unterschiedlich groß angezeigt. Weiter soll es die Möglichkeit geben Tags auszuwählen und sich alle Challenges und Lösungsvorschläge mit diesem Tag anzusehen. Mit Hilfe der Tags können Inhalte kategorisiert werden. Dies erleichtert die Suche. Auch bietet es eine Möglichkeit Redundanz zu vermeiden.

- Das System muss die Möglichkeit bieten Tags bei Challenges und Lösungsvorschlägen anzugeben.
- Das System wird eine Tag-Cloud anzeigen.
- Das System wird die Möglichkeit bieten Tags innerhalb der Tag-Cloud anzuklicken, um zugehörige Challenges und Lösungsvorschläge in einer Liste anzuzeigen.

Ideenbörse

Die Ideenbörse ist eine Möglichkeit Ideen innerhalb des Unternehmens zu vermarkten, sowie Interessenten für die Mitarbeit oder für die Finanzierung dieser Idee zu gewinnen. Es entsteht so eine Art Crowdfunding für die Idee. Crowdfunding ist eine Möglichkeit der Finanzierung im Internet. Hierbei kann Eigenkapital anderer für die Umsetzung einer Idee beschafft werden. Es ist eine Möglichkeit Ideen von anderen bewerten zu lassen. Wenn jemand Interesse an der Idee zeigt, wird die Idee mittels der Ideenbörse voran getrieben. Folgende Anforderungen lassen sich daraus ableiten.

- Das System wird fähig sein eine Ideenbörse zu integrieren.
- Das Innovationsmanagementportal wird weiterhin die Möglichkeit bieten Challenges und Lösungsvorschläge gemeinsam unabhängig von der Ideenbörse zu erarbeiten.

Kollaborativer Raum

Der kollaborative Raum ist ein digitaler Raum, indem mehrere Personen gemeinsam an einer Aufgabe arbeiten können. Mittels Whiteboard ersetzt dieser, zusammen mit einem VideoChat, die Zusammenarbeit in einem realen Raum. Außerdem ergibt sich daraus die Möglichkeit zu unterschiedlichen Zeiten an einem Projekt zu arbeiten. Mit dem kollaborativen Raum wird ein Gedanke des Design Thinkings umgesetzt, da es Mitarbeitern unterschiedlicher Disziplinen ermöglicht in einem kreativen Umfeld Ideen zu entwickeln. Die Anforderungen an den kollaborativen Raum sind wie folgt definiert.

- Das System soll fähig sein einen kollaborativen Raum zur Verfügung zu stellen.
- Der kollaborative Raum sollte ein Witheboard besitzen.
- Der kollaborative Raum wird die Möglichkeit haben Daten auf dem Witheboard zu sichern.

Video-Chat

Ein Video Chat ermöglicht es den Mitarbeitern einen synchronen Informationsaustausch mit Bild und Ton herzustellen ohne dazu am selben Ort zu sein.

- Das System wird fähig sein einen VideoChat anzubieten.

Stand-Alone-System

Ein Stand-Alone-System funktioniert unabhängig von anderer Software und muss nicht installiert werden. Die Anwendung beinhaltet eine unabhängige Benutzerverwaltung. So kann sie unabhängig von anderer Software des Unternehmens eingesetzt werden. Außerdem kann jeder von überall mit einer Internetverbindung darauf zugreifen. Folgende Anforderungen sind für ein Stand-Alone-System definiert.

- Das System muss eine unabhängige Benutzerverwaltung besitzen.
- Das System soll von überall zugänglich sein.

Chat

Auf einer Plattform, die die Zusammenarbeit von Mitarbeitern fördern soll, ist die Kommunikation ein wichtiger Erfolgsfaktor. Aus diesem Grund soll eine (Gruppen)-Konversation über einen Chat realisiert werden.

- Das System muss die Möglichkeit bieten, dass Benutzer miteinander chatten können.
- Das System muss fähig sein einen Einzel- als auch Gruppen-Chat zu ermöglichen.
- Das System sollte die Möglichkeit bieten die Chat-Historie aufzuzeichnen.
- Das System sollte die Möglichkeit bieten die Chat-Historie aufzurufen.
- Das System sollte die Möglichkeit bieten Dateien im Chat zu übertrage.

Dateitransfer

In mehreren Bereichen, wie z.B. bei der Challenge oder dem Projekt, ist es nötig die erfassten Inhalte mit weiteren Informationen externer Quellen anzureichern. Aus diesem Grund muss es möglich sein, Dateien hoch- und -herunterzuladen.

- Das System muss die Möglichkeit bieten Dateien hochzuladen.
- Das System muss die Möglichkeit bieten Dateien herunterzuladen.
- Das System muss unterschiedliche Dateitypen unterstützen.
- Das System sollte die Möglichkeit bieten mehrere Dateien zeitgleich hochzuladen (Multi-Upload).
- Das System sollte die Möglichkeit bieten Dateien zu löschen.

Mitarbeiterbörse

Für Projekte werden Mitarbeiter mit bestimmten Qualifikation benötigt. Häufig gestaltet es sich schwierig, vor allem in anderen Unternehmensbereichen, diese zu finden. Um eine zentrale Anlaufstelle und Suche zu ermöglichen, soll eine Mitarbeiterbörse umgesetzt werden. Dort kann ein Mitarbeiter freiwillig seine Qualifikationen und Interessen hinterlegen. In der Suche kann entsprechend dieser Kriterien gefiltert werden.

- Das System sollte die Möglichkeit bieten, dass der Benutzer seine Qualifikation und Interessen im Profil hinterlegt.
- Das System sollte fähig sein eine Übersicht aller teilnehmenden Mitarbeiter bereitzustellen.
- Das System sollte die Möglichkeit bieten nach bestimmten Kriterien zu filtern.
- Das System sollte die Möglichkeit bieten Mitarbeiter zu Projekten einzuladen.
- Das System sollte die Möglichkeit bieten entsprechend der formulierten Aufgaben passende Mitarbeiter mit freier Kapazität vorzuschlagen.

Profil bearbeiten

Unterschiedlichste Veränderungen können dazu führen, dass die Daten im Profil nicht mehr aktuell sind. Sei es der Tausch von Büroräumen oder die Hochzeit mit Namensänderung. Sofern die personenbezogenen Daten nicht zentral gepflegt werden, muss der Benutzer die Möglichkeit haben seine Angaben anzupassen.

- Das System muss die Möglichkeit bieten das Profil zu bearbeiten.
- Das System muss die Möglichkeit bieten die Änderungen zu speichern.

Profilübersicht

Um die Kontaktdaten anderer Benutzer zu lesen ist es notwendig, dass es eine Benutzerübersicht gibt. In dieser Übersicht sollten zudem Möglichkeiten der Filterung umgesetzt werden.

- Das System muss die Möglichkeit bieten alle Benutzer in einer Übersicht darzustellen.
- Das System muss die Möglichkeit bieten in der Übersicht Filterkriterien einzugeben.

3.3. Nicht-Funktionale-Anforderungen

Nicht funktionale Anforderungen beschreiben wie ein System oder einzelne Funktionen arbeiten sollen. Dazu kommen weitere Qualitätsanforderungen u. a. aus den Bereichen Sicherheit und Performance. (vgl. [Sch16])
Die Kategorien der nicht funktionalen Anforderungen orientieren sich an die ISO25010 Norm (vgl. Abbildung 9). Diese besteht aus acht Hauptbereichen, welche wiederum in einzelne Ausprägungen unterteilt sind. Die ISO Norm beschreibt somit eine Taxonomie von Softwarequalitätseigenschaften. Es muss jedoch darauf hingewiesen werden, dass es sich

um keine harmonisierte (europäische) Norm handelt, sondern eher als Checkliste geeignet ist um die Vollständigkeit der Systemanforderungen zu überprüfen.(vgl. [Joh16])

Abbildung 9: Nichtfunktionelle Anforderungen nach ISO25010, entnommen aus [Chr15]

Die von der Projektgruppe festgelegten nicht funktionalen Anforderungen sind die wesentlichen Anforderungen an das System. Für weitere und detailliertere nicht funktionale Anforderungen und Kennzahlen fehlen entsprechende Vergleichswerte. Dieses spiegelt sich vor allem in den Bereichen der Wartbarkeit und Zuverlässigkeit wieder. Auf einige weitere Faktoren, wie z.B. die Performance des Servers, hat die Projektgruppe keinen Einfluss. Dieser wird von der Universität Oldenburg gestellt. Im folgenden sind die Anforderungen entsprechend der ISO25010-Kategorien festgehalten. Der Bereich der *Übertragbarkeit* beschreibt, wie leicht sich ein System in andere Umgebungen übertragen lässt. Hier haben wir folgende Anforderungen definiert.

Anpassungsfähigkeit

- Das System muss die Elemente entsprechend der definierten Berechtigungsrollen 4.1 anpassen.
- Das System sollte dem Benutzer die Möglichkeit bieten Elemente individuell anzupassen.

Installierbarkeit

- Das System muss über eine aktuelle Installationsanleitung verfügen.
- Das System muss über eine aktuelle Update-Dokumentation verfügen.
- Die Installationsdauer des Systems muss kleiner gleich acht Stunden sein.
- Die Installationsdauer von Updates muss kleiner gleich zwei Stunden sein.
- Die Installation des Systems sollte ohne externe Unterstützung möglich sein.
- Die Installation der Updates sollte ohne externe Unterstützung möglich sein.

Austauschbarkeit

- Die Module des Systems, wie z.B. das Whiteboard, sollten mit einem Aufwand kleiner gleich drei Personentagen austauschbar sein.

Im Bereich der *Wartbarkeit* werden Anforderungen erfasst, die sich mit dem Aufwand der Durchführung von Änderungen befassen.

Modularität

- Das System sollte aus separaten Modulen bestehen.
- Der Ausfall eines Moduls sollte nicht die Funktionstüchtigkeit anderer Module beeinflussen.
- Die Module eines Systems sollten ein eindeutige Schnittstelle besitzen, in denen die Ein- und Ausgabedaten beschrieben sind.
- Die Module sollten flexibel miteinander kombinierbar sein.

Wiederverwendbarkeit

- Die Wiederverwendbarkeit von Systemkomponenten sollte größer gleich 20% sein.
- Das System sollte nach Möglichkeit standardisierte Frameworks einsetzen.

Analysierbarkeit

- Das System sollte auf die nötigsten Komponenten reduziert sein, um Fehlerquellen durch Analysen schnellstmöglich identifizieren zu können.

Modifizierbarkeit

- Die Modifizierbarkeit des Systems muss durch den Hersteller gewährleistet sein.

Prüfbarkeit

- Die Prüfbarkeit von System-Funktionen sollte größer gleich 80%.
- Die automatisierte Prüfbarkeit von System-Funktionen sollte größer gleich 65%.

Im Bereich der *Sicherheit* werden Anforderungen definiert, um den Zugriff als auch die Daten vor unbefugten Zugriff Dritter zu schützen.

Vertraulichkeit

- Das System muss die Vertraulichkeit von Nachrichten sicherstellen und vor Zugriff Dritter schützen.

Integrität

- Das System sollte die Dateiübertragungen mittels einer Prüfsumme vor Übertragungsfehlern schützen.
- Das System sollte die erfolgreiche Übertragung von Nachrichten mit Sequenznummern sicherstellen.
- Das System muss die Daten entsprechend der Integritätsbedingungen in der Datenbank speichern.

Nachweisbarkeit

- Das System sollte die wichtigsten Änderungen zwecks Nachweisbarkeit protokollieren.

Also wichtige Änderungen gelten neben den Änderungen an den inhaltlichen Elementen (Challenge, Lösungsvorschlag, Projekt, Business Case und Feedback) auch systemrelevante Änderungen wie z.B. die Änderung von Benutzerrollen und Rechten.

Verantwortlichkeit

- Das System muss über Rollen- und Berechtigungskonzepte sicherstellen, dass der Benutzer nur Zugriff auf die für ihn bestimmten Verantwortlichkeiten erhält.

Authentizität

- Das System sollte mittels digitaler Signaturen die Authentizität von Nachrichten und Dokumenten sicherstellen.

Die *Effizienz* beschreibt das Verhältnis zwischen Leistungsniveau der Plattform und der eingesetzten Ressourcen.

Antwortzeitverhalten

- Das System muss auf Eingaben mit einem Antwortzeitverhalten von kleiner gleich drei Sekunden reagieren.

Ressourcenverbrauch

- Das System sollte ressourcenschonend aufgebaut sein.
- Um Lastspitzen abfangen zu können, muss das System auf 150% der minimalen Ressourcen zurückgreifen können.

Kapazität

- Das System muss 1250 Transaktionen pro Stunde durchführen können.

Im Bereich der *Funktionalität* wird festgelegt, inwieweit die Software die geforderten Funktionalitäten umsetzt:

Vollständigkeit

- Das System muss die rechtlich verpflichtenden Anforderungen vollständig umgesetzt haben.
- Das System muss von den zusätzlichen Anforderungen mindestens 25% umgesetzt haben.

Korrektheit

- Das System muss die in den verpflichtenden Anforderungen und User Stories enthaltenen Spezifikationen korrekt umsetzen.

Die *Zuverlässigkeit* beschreibt, ob die Software ein bestimmtes Leistungsniveau unter bestimmten Bedingungen über einen bestimmten Zeitraum aufrechterhalten kann.

Ausgereiftheit

- Das System muss die wichtigsten Funktionen fehlerfrei zur Verfügung stellen.

Fehlertoleranz

- Das System sollte durch den modularen Aufbau eine Grundfunktionalität zur Verfügung stellen, selbst wenn Fehler in einem der Module existieren.

Verfügbarkeit

- Das System muss eine Verfügbarkeit von Minimum 95% erreichen.

Wiederherstellbarkeit

- Das System muss in kleiner gleich acht Stunden wiederhergestellt werden können.

Die *Benutzbarkeit/Gebrauchstauglichkeit* beschreibt, welchen (Schulungs-)Aufwand die Benutzer bei der Verwendung der Software aufbringen müssen.

Verständlichkeit

- Das System muss in der deutschen Sprache verfügbar sein.
- Das System sollte einen übersichtlichen und strukturierten Aufbau haben.
- Das System muss ein Benutzerhandbuch in deutscher Sprache bereitstellen.
- Das System sollte Feldhilfen für Eingabefelder zur Verfügung stellen.

Erlernbarkeit

- Das System muss in einem Schulungstag für den Benutzer erlernbar sein.

Bedienbarkeit

- Das System muss durch den Benutzer intuitiv bedienbar sein.
- Der Benutzer muss in 90% der Fälle die Aufgabe `Challenge erstellen` in unter zehn Minuten erfüllen.
- Das System muss mit Maus/Tastatur und Touch-Eingaben bedient werden können.

Attraktivität

- Das System sollte modern, aufgeräumt und motivierend gestaltet sein.
- Das System sollte über ein abgestimmtes Farbschema verfügen.

Barrierefreiheit

- Das System sollte Personen mit Rot-/Grün-Schwäche barrierefreie Designs bereitstellen.
- Das System sollte Personen mit Sehschwächen eine Vergrößerung der Schrift bereitstellen.
- Das System muss Hintergrund und Schrift kontrastreich voneinander trennen.
- Das System muss auf animierte Grafiken verzichten.
- Das System muss Hyperlinks von normaler Schrift deutlich abgrenzen.

Fehlertoleranz gegenüber Anwenderfehlern

- Das System sollte mögliche Fehlerquellen durch Anwenderfehler ausschließen.
- Das System sollte bei Anwenderfehlern, die nicht abgefangen wurden, eine aussagekräftige Fehlermeldung bereitstellen.

Im Bereich der *Kompatibilität* wird beschrieben, ob und wie gut das System mit anderen heterogenen und homogenen System funktioniert.

Koexistenz

- Das System muss neben einer Software mit ähnlichen oder gleichen Funktionen arbeiten können.

Interoperabilität

- Das System sollte durch standardisierte Schnittstellen mit anderen heterogenen Systemen kommunizieren können.

4. Konzeption

In diesem Kapitel werden alle notwendigen Konzepte zur Erstellung der Innovationsmanagementplattform vorgestellt und einzeln erläutert. Ebenso werden die Entscheidungen für die eingesetzten Technologien erörtert.

4.1. Berechtigungskonzept

Nachdem die Anforderung gestellt wurde, dass nicht alle Nutzer den Zugriff auf alle Funktionen erhalten sollen und es Unterschiede bspw. bei der Editierbarkeit von Einträgen geben muss, ist es erforderlich, dass ein Konzept für die Berechtigungen von Nutzern erstellt wird.
Für die Erstellung des Konzepts der Berechtigungen für die Innovationsmanagementplattform IMPACT wurden die umzusetzenden Funktionalitäten betrachtet und festgelegt, welche Nutzerrolle eine Funktionalität in welcher Form nutzen kann. Dies bildet die Basis für die Definition von Berechtigungsrollen. Für die Erstellung eines Berechtigungskonzepts wurden folgende Rollen definiert, die im weiteren Verlauf dieses Kapitels detaillierter Vorgestellt werden:

- Standarduser,
- Abteilungsleiter,
- Administrator,
- Projektleiter,
- Projektbeteiligter,
- Management,
- Controller (FiBu) und
- Eigentümer.

Beim Standarduser (ST) handelt es sich um einen Basisuser, den jeder Anwender des Systems erhält, nachdem die Registrierung erfolgreich abgeschlossen wurde. Mit dem Standarduser kann sich der Anwender am System anmelden und die Grundfunktionen der Innovationsmanagementplattform nutzen, dazu gehören Kommentare, Lösungsvorschläge abgeben und bewerten sowie Challenges und Einsehen von Projekten. Bei den Profilen anderer Nutzer kann der Basisnutzer nur ausgewählte Informationen wie E-Mailadresse und Abteilung sehen. Zudem besitzt der Standarduser die Möglichkeit eine eigene Wunschliste zu erstellen. Des Weiteren kann der Standarduser sein eigenes Nutzerprofil komplett einsehen, bearbeiten sowie Einträge in der eigenen Wunschliste hinzufügen und löschen.

Eine weitere Rolle stellt den Abteilungsleiter (AT) dar, der eine erweiterte Berechtigung gegenüber dem Standarduser bietet. Die Anwender mit der Berechtigung des AT verfügt über die Befugnis, sich die Wunschlisten aller Standarduser anzusehen. Dies bietet die Möglichkeit, dass der Leiter einer Abteilung die persönlichen Wünsche seines Mitarbeiters ansehen kann, um mit der Erfüllung eines Wunsches zur Steigerung der Motivation beizutragen. Der Administrator (AM) verfügt über die Berechtigungen des Abteilungsleiters, jedoch kann ein AM zusätzlich noch die Archivierung von Projekten oder Challenges vornehmen. Zum jetzigen Zeitpunkt ist keine gesonderte Adminstratorenansicht geplant, denn sonst bekommt ein Anwender mit der Berechtigung des Administrators das Recht die Anwender zu verwalten (neue Anwender anlegen, sperren oder freischalten). Der Administrator würde dabei als Superuser auftreten, der alle Bereiche des Systems beeinflussen kann, dazu zählt auch die Datenbank.
Der Projektleiter (PL) nimmt eine zentrale Rolle in IMPACT ein und verfügt somit über die umfassendsten Berechtigungen. Neben den Berechtigungen eines Standardusers, hat der Projektleiter die Berechtigung, eine Entscheidung zu treffen. Aus diesem Grund erhält ein Anwender nur die Berechtigung des Projektleiters, wenn dieser bereits reale Erfahrungen mit der Leitung eines Projekts gemacht hat. Durch die Berechtigung der Entscheidungsgewalt kann der Projektleiter bspw. den Übergang zwischen den Phasen Challenge und Projekt durchführen, Teile des Projekts für erfüllt erklären und das Projekt zur Entscheidung in den Business Case geben. Der Projektleiter verfügt zudem über die Berechtigung, die kompletten Nutzerprofile einzusehen, um sich sein Projektteam zusammenzustellen und nach einem erfolgreichen Projekt eine Erfolgsbeteiligung in Form eines erfüllten Anwenderwunsches zu gewähren.
Der Projektbeteiligte (PB) verfügt über die Berechtigungen des Standardusers und kann zusätzlich in der Phase des Projekts eine Bearbeitung eines Projektes vornehmen, nachdem dieser vom Projektleiter ins Projektteam aufgenommen wurde.
Der Anwender mit den Berechtigungen des Managements (M) ist ein Entscheider, der über die Berechtigungen des Standardusers verfügt und die Berechtigung hat, einen Business Case zu genehmigen, abzulehnen oder zur Überarbeitung zu geben. Zusätzlich kann dieser Anwender die Challenges, Projekte oder den Business Case archivieren und damit beenden. Ein Anwender, der über die Berechtigung des Management verfügt, kann alle Funktionen im Bereich des Business Case nutzen und im Bereich des Feedbacks ein Feedback abgeben.
Der Controller (CO) ist eine Berechtigung, die ein Anwender aus der Finanzbuchhaltung übernimmt. Dieser verfügt über die Berechtigungen des Standardusers und kann zusätzlich den Business Case einsehen. Ein Controller ist berechtigt den Business Case zu bearbeiten, um durch die finanziellen Analysen die Entscheidung des Managements zu unterstützen.
Die Berechtigungen des Eigentümers (ET) erlangt jeder Standarduser, der einen Beitrag

nicht anonym veröffentlicht bspw. durch die Erstellung einer Challenge, die Abgabe seiner Meinung in Form eines Kommentars oder einer Bewertung. Im Gegensatz zum Standarduser, kann ein Eigentümer seine Beiträge editieren.
Nach dem Abschluss der Definition der Berechtigungen wird festgelegt, bei welcher Funktion eine Rolle über welche Berechtigung verfügen muss. Dazu werden Anwendungsfälle definiert und festgelegt, welche Anwender in den Anwendungsfällen über die Berechtigung zur Ausführung der Funktion verfügen. Das Ergebnis ist eine Tabelle mit über 55 Anwendungsfällen, die zur Vereinfachung der Implementierung zusammengefasst wurde. Mit der Bezeichnung AbstractItem sind die ersten beiden Phasen (Challenge und Projekt) zusammengefasst, deren Funktionen in diesen Fällen gleich sind. Das Berechtigungskonzept wird in der Tabelle 3 abgebildet.

Anwendungsfall	**ST**	**AT**	**AD**	**PL**	**M**	**CO**	**PB**	**ET**
Nutzerprofil: Profileingaben ändern	-	-	-	-	-	-	-	X
Nutzerprofil:Profilangaben einsehen	X	X	-	X	X	-	-	X
AbstractItem: Bearbeiten	-	-	-	X	-	-	X	-
AbstractItem: Bewerten	X	X	X	X	X	X	X	X
AbstractItem: Archivieren	-	-	X	X	-	-	-	-
Challenge: Challenge veröffentlichen	-	-	-	X	-	-	-	-
AbstractItem: Einsehen	X	X	X	X	X	X	X	X
Kommentar: Kommentar hinzufügen	X	X	X	X	X	X	X	X
Kommentar: Kommentar bearbeiten	-	-	-	-	-	-	-	X
AbstractItem: Anhänge hinzufügen	-	-	-	X	-	-	X	-
AbstractItem: Übergang: Angaben bearbeiten	-	-	-	X	-	-	-	-
Business Case: BC einsehen	-	-	-	X	X	X	-	-
Business Case: BC genehmigen/ablehnen	-	-	-	-	X	-	-	-
Business Case: BC bearbeiten	-	-	-	X	X	X	-	-
Business Case: BC archivieren	-	-	-	X	X	-	-	-
Business Case: Anhänge hinzufügen	-	-	-	X	X	X	-	-
Feedback: Feedback abgeben	-	-	-	X	X	-	-	-
Feedback: Feedback einsehen	X	X	X	X	X	X	X	-
Bewertungen: Bewertungen bearbeiten	X	-	-	-	-	-	-	X

Tabelle 3: Berechtigungskonzept

4.2. Motivationskonzept

Die Mitarbeitermotivation ist ein Kernelement der Innovationsmanagementplattform. Die Anwender sollen dazu angehalten werden, die Plattform dauerhaft und langfristig zu nutzen.

4.2.1. Motivationsziele in Unternehmen

Aus der Sicht des Unternehmens können mit motivierten Angestellten die Wirtschaftlichkeit erhöht und somit die Umsätze gesteigert werden. Motivierte Mitarbeiter bringen sich eher in den Innovationsprozess ein. Ergänzend zu den wirtschaftlichen Zielen werden die Mitarbeiter durch eine entsprechende Motivation an das Unternehmen gebunden. Dadurch kann gewährleistet werden, dass Unternehmen ebenfalls in der Zukunft erfolgreich arbeiten können. Im Bereich des Marketings tragen zudem motivierte Mitarbeiter dazu bei, dass Kunden sich an das Unternehmen langfristig binden, sodass Umsätze auch in der Zukunft gesichert werden. Für die Erreichung dieser Ziele müssen dabei die Unternehmen insbesondere die intrinsische Motivation ansprechen (vgl. Seminararbeit Motivation und Anreizsysteme).

4.2.2. Motivation außerhalb von Anwendungen

Neben den Maßnahmen, die sich in der Plattform umsetzen und unterstützen lassen, sind viele Methoden zur Mitarbeitermotivation unabhängig von der Plattform zu sehen. Darunter zählen Anerkennung durch Lob und persönliche Gespräche der Vorgesetzten mit den Angestellten, die Negierung der Sprache im persönlichen Umgang, Feedback für die Mitarbeiter durch ihre Vorgesetzten, das Arbeitsumfeld oder auch regelmäßige Einführungen in die Plattform.
Für viele Angestellte stellt ein persönliches Lob und Gespräche ein wichtiger Faktor für die Motivation dar. Den Umgang der Anwender untereinander kann die Innovationsmanagementplattform IMPACT nicht beeinflussen, sondern dieses wichtige Instrument der Mitarbeitermotivation lässt sich nur durch die Unternehmenskultur umsetzen. Ebenso zählt das Arbeitsumfeld bspw. der Umgang unter den Kollegen im Team oder Bereitstellung von Ressourcen zu den Punkten, die unabhängig von der Plattform zur Motivation der Mitarbeiter beitragen. Eine weitere Methode zur Steigerung der Motivation von Angestellten zur Nutzung der Plattform besteht in der Durchführung regelmäßiger Einführungen, damit u.a. neue Mitarbeiter das System kennen lernen. Auf die Häufigkeit der Durchführungen derartiger Einführungen kann mit der Plattform keinen Einfluss genommen werden.

4.2.3. Umsetzung in IMPACT

Für die Umsetzung eines Mitarbeitermotivationskonzept kann die Plattform mit verschiedenen Möglichkeiten beitragen, dazu zählt insbesondere die Förderung der intrinsischen Motivation. Die Angestellten sollen dadurch zu einer aktiven Beteiligung aufgefordert werden und ihre Meinung vertreten können, ohne dabei negative Auswirkung auf ihre eigentliche Beschäftigung befürchten zu müssen. Dabei nimmt die persönliche Ansprache eine essenzielle Rolle ein, indem der Anwender durch das Duzen direkt angesprochen wird. Das Duzen vermittelt dem Anwender eine gewisse Nähe und Vertrautheit. Bei der Umsetzung wird dies bei allen Erklärungen, Ansprachen oder auch Rückmeldungen eingesetzt.

Abbildung 10: Begrüßungstext auf der Startseite

Die Abbildung 10 zeigt, dass dies bereits im Login umgesetzt wird. Durch die persönliche Ansprache wird dem Anwender das Gefühl vermittelt, dass dieser ein Teil des Systems ist und im Fokus steht. Zudem sind die Texte und Hinweise in einer verständlichen Weise gestaltet, sodass für den Anwender direkt ersichtlich wird, wie der weitere Prozessverlauf aussieht. Bei einer erfolgreichen Durchführung erhält der Anwender einen grünen Hinweis, bspw. mit *Das Profil wurde erfolgreich aktualisiert.* Im Fall der nicht erfolgreichen Ausführung wird der Hinweis in rot dargestellt. Der Anwender erhält eine Rückmeldung in Form einer Ursachenbeschreibung und Hinweise zur erfolgreichen Durchführung. Durch diese Hilfestellungen wird verhindert, dass der Anwender die Motivation verliert, sich aktiv an IMPACT zu beteiligen. Die persönliche Ansprache setzt sich in der spielerischen und graphische Darstellung von Elementen fort, indem dem Anwender fünf verschiedene Themes zur Verfügung gestellt werden, die der Anwender im Nutzerprofil einstellen kann. Durch diese Auswahlmöglichkeit kann sich der Anwender, das für sich ansprechendste Theme, auswählen. Zudem trägt das im Kapitel 4.5 beschriebene UI-Design zur Steigerung der Mitarbeitermotivation bei, indem dabei die Grundsätze der Benutzerfreundlichkeit beachtet werden.

Abbildung 10 zeigt, dass dies bereits beim Login umgesetzt wird. Durch die persönliche Ansprache wird dem Anwender das Gefühl vermittelt, dass dieser ein Teil des Systems abbildet und dabei im Fokus steht. Zudem sind die Texte und Hinweise in einer verständlichen Weise gestaltet, sodass für den Anwender direkt ersichtlich wird, wie das weitere Vorgehen ist. Bei einer erfolgreichen Durchführung erhält der Anwender einen grünen Hinweis, bspw. mit *Das Profil wurde erfolgreich aktualisiert.* Im Fall der nicht erfolgreichen Ausführung wird der Hinweis in rot dargestellt. Der Anwender erhält eine Rückmeldung in Form einer Ursachenbeschreibung und Hinweise zur erfolgreichen Durchführung. Der Hinweis könnte bspw. lauten *Login fehlgeschlagen, weil Nutzer nicht vorhanden.* Durch diese Hilfestellungen wird verhindert, dass der Anwender die Motivation verliert, sich aktiv an IMPACT zu beteiligen. Die persönliche Ansprache setzt sich in der spielerischen und graphische Darstellung von Elementen fort, indem dem Anwender fünf verschiedene Themes zur Verfügung gestellt werden, die der Anwender im Nutzerprofil einstellen kann. Durch diese Auswahlmöglichkeit kann sich der Anwender das für sich ansprechendste Theme aussuchen. Zudem trägt das im Kapitel 4.5 beschriebene UI-Design zur Steigerung der Mitarbeitermotivation bei, indem dabei die Grundsätze der Benutzerfreundlichkeit beachtet werden.
Neben diesen allgemeinen graphischen Elementen und Hinweisen tragen die folgenden Elemente zur Motivation der Anwender bei.

- Aktivitätenstream,
- Nutzerprofil,
- Anonyme Beiträge,
- Wunschliste,
- Rückmeldungen (Feedback),
- Checklisten und
- Anwenderschulungen.

Im Folgendem werden die Elemente in Bezug auf die Motivation der Anwender betrachtet.

Aktivitätenstream

Der Aktivitätenstream nimmt hinsichtlich der Motivation eine essenzielle Rolle ein. Mit dem Aktivitätenstream sollen die Veränderungen (z.B. Erstellen einer Challenge, Bearbeiten eines Lösungsvorschlages) auf der Plattform in einer zeitlichen Abfolge dargestellt werden. Der Nutzer bekommt dadurch die Möglichkeit, Neuigkeiten von anderen Nutzern nachzuverfolgen. Durch eine Verlinkung wird dem Anwender ermöglicht, eine direkte Navigation zu den entsprechenden Elementen (z.B. zu der jeweiligen Challenge, Lösungsvorschlag) durchzuführen, um auf Veränderungen reagieren und den aktuellen Stand nachvollziehen zu können.

Nutzerprofil

Neben der im Rahmen der Registrierung erforderlichen Angaben, besitzt der Anwender die Möglichkeit weitere freiwillige Angaben, wie z. B. Projekterfahrung, Kontaktinformationen oder besondere Fähigkeiten im Nutzerprofil einzutragen. Die Eintragung dieser Informationen geschieht auf freiwilliger Basis, da die Verbreitung der Daten von vielen Mitarbeitern unerwünscht sein könnte. Der Zwang zur Eintragung dieser Informationen würde eher zu einer Demotivation der Anwender führen. Zudem sind die freiwilligen Angaben nicht für alle Anwender sichtbar, sondern stehen in Abhängigkeit zur jeweiligen Berechtigung (vgl. Kapitel 4.1) bereit. Lediglich die Kontaktinformationen sind für alle Anwender einsehbar. Die Betrachtung der Informationen zur Projekterfahrung und besonderen Fähigkeiten sind zusätzlich für einen Projektleiter möglich, da dieser die Informationen bspw. für die Zusammenstellung des Projektteams benötigt. Durch die Einschränkung der Einsicht auf die Angaben im Nutzerprofil ist der Anwender eher bereit diese freiwilligen Informationen einzugeben. Ebenfalls hat der Anwendet die Möglichkeit, ein Profilbild einzustellen. Das Profilbild wird bei jeder Eintragung in IMPACT angezeigt und stellt somit einen persönlichen Bezug zwischen dem Eintrag und dem Anwender her.

Anonyme Beiträge

Um die Hemmschwelle für die Abgabe von außergewöhnlichen Beiträgen bei den Anwendern zu senken, gibt es die Möglichkeit, dass die Beiträge anonym abgegeben werden können. Dadurch kann der Beitrag keinem Anwender zugeordnet werden. Diese Möglichkeit der anonymen Abgabe kann die Angestellten motivieren auch außergewöhnliche Beiträge und Probleme einzutragen, ohne eine negative Reaktion seitens des Vorgesetzten oder Kollegen befürchten zu müssen. Für den Angestellten hat dieses Vorgehen nur den Nachteil, dass dieser ggf. nicht an einer Belohnung beteiligt wird.

Wunschliste

In der Wunschliste im Nutzerprofil kann der Anwender seine persönlichen Wünsche eintragen. Durch die Erfüllung persönlicher Wünsche eines Mitarbeiters kann der Entscheidungsträger, Projektleiter oder Abteilungsleiter zur Förderung der Motivation beitragen. Die vom Anwender eingetragenen Wünsche können dabei bspw. Eintrittskarten für eine Veranstaltung, Reise an eine bestimmten Ort, oder auch eine Fortbildung sein. Der Anwender kann seine Wünsche mit Links versehen, in Kategorien einteilen, aber auch Kollegen für den Wunsch einladen. Dadurch kann der Anwender bspw. mit einem bestimmten Kollegen eine Fortbildung besuchen. Die Wunschliste bietet die Möglichkeit, dass das Unternehmen die Wünsche des Mitarbeiters besser kennenlernt und ihn somit überraschen kann.

Rückmeldungen (Feedback)

Einer der wichtigsten Punkte zur Steigerung der Mitarbeitermotivation besteht, wie im Kapitel 4.2.2 beschrieben, in einem regelmäßigen Feedback. Dies bezieht sich zum einen auf ein direktes Feedback durch die Plattform selbst bspw. beim erfolgreichen Anlegen einer Challenge und zum anderen in einem indirekten Feedback bspw. bei der Abgabe einer Bewertung oder Lösungsvorschlag. Gleichzeitig wird dem Anwender die Möglichkeit gegeben seine Meinung durch die Eintragung eines Lösungsvorschlags, einer Challenge und einem Kommentar zu hinterlassen und somit eine Rückmeldung an den Ersteller zugeben. Durch diese Funktionen in IMPACT wird der Anwender aufgefordert sich aktiv am Prozess zu beteiligen.
Eine weiteres Feedback kann der Entscheider in der Phase des Business Cases geben, indem eine Begründung als Rückmeldung eingegeben wird, wenn ein Business Case abgelehnt oder überarbeitet werden muss. Zudem wird in IMPACT die Möglichkeit gegeben, in der Umsetzungsphase des Projekts ein regelmäßiges Feedback abzugeben. Dies können dabei bspw. Projektbeginn, Zwischenstände, Projektabschluss oder Erfolgsgeschichten sein.
Bei der Formulierung des Feedbacks, insbesondere der Vermeidung von Negierungen und die Häufigkeit kann IMPACT nicht zur Steigerung der Mitarbeitermotivation beitragen, da die Eintragung eines Feedbacks ausschließlich durch einen Entscheider erfolgt. Eine weitere Feedbackmöglichkeit für den Anwender besteht in der Abgabe einer Begründung bei der Erstellung einer Bewertung. Diese Möglichkeit kann im Bezug auf die Motivation aus zwei Blickwinkeln betrachtet werden. Bei einer positiven Rückmeldung führt das Feedback zu einer Steigerung der Motivation und motiviert den Ersteller weitere Beiträge abzugeben (vgl.[Rei16]). Ein wiederholt negatives Feedback für zu einer Demotivation, sodass der Anwender die aktive Anwendung einstellt und von weiteren Einträgen absieht.

Checklisten

Die Checklisten zeigen den aktuellen Bearbeitungsstand und Veränderungen an. Diese Veränderungen werden für den Anwender im Bereich des Projekts und Business Case sichtbar und tragen dazu bei, dass die Anwender motiviert werden, die jeweilige Phase abzuschließen. Die Checklisten geben Auskunft, welche der erforderlichen Bereiche bereits erfüllt wurden und welche Voraussetzungen für den Übergang in die nächste Phase noch erfüllt werden müssen. Gerade wenn nur noch wenige Punkte zu erfüllen sind, damit die Phase abgeschlossen werden kann, trägt die Checkliste zur Steigerung der Motivation bei.

Anwenderschulungen

Ein wichtiger Punkt für die Motivation neuer Anwender und als Informationsmöglichkeit bei Fragen, besteht in der Bereitstellung von Unterlagen in Form eines Benutzerhandbuchs und einer Installationsanweisung A.1. Diese werden der Plattform hinzugefügt, um den Anwender einen einfacheren Einstieg zu ermöglichen. Durch die Gestaltung eines Schulungsleitfaden kann gewährleistet werden, dass eine Einführung durchgeführt wird und immer nach dem selben Vorgehen abläuft. Zudem erhalten die Anwender einen einheitlichen Wissenstand.

4.3. Prozessmodell

Das Prozessmodell bildet das Kernelement der Webanwendung IMPACT. Um den bisherigen innerbetrieblichen Innovationsprozess zu verbessern oder vielmehr zu erstellen, ist es notwendig, alle Schritte zu betrachten, die benötigt werden, um eine Idee voranzubringen. Zunächst wird der Weg zum endgültigen Prozessmodell aufgezeigt und dabei auf die Probleme der vorherigen Versionen eingegangen. Anschließend wird die dritte Version des Prozessmodells, mit ihren einzelnen Prozessschritten, thematisiert.

4.3.1. Weg zum Prozessmodell

Um Mitarbeitern den Zugang zum Innovationsprozess zu ermöglichen, ist es zunächst notwendig grundlegende Meilensteine innerhalb des Prozesses festzulegen. Zunächst wurde mit Hilfe von Brainstorming ermittelt, welche Elemente für den Innovationsprozess notwendig sind.

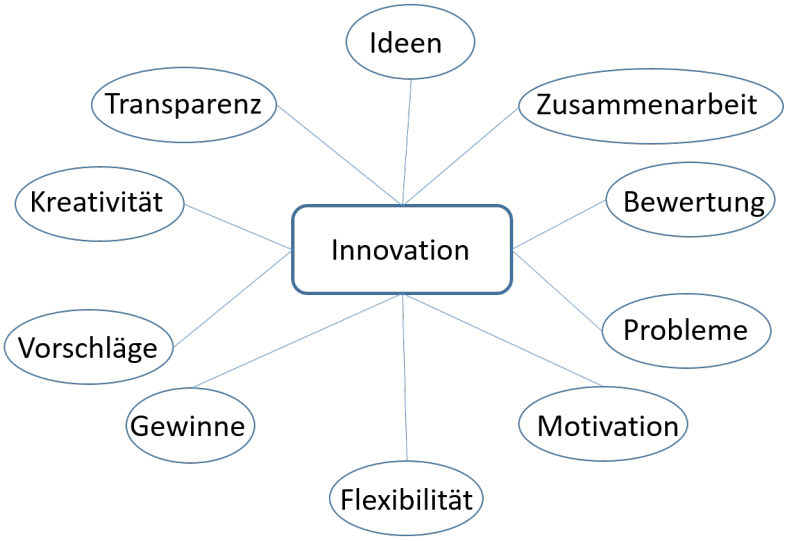


Abbildung 11: Brainstorming zum Prozessmodell

Das Brainstorming hat ergeben, dass zum einen das kreative Umsetzen von Innovationen zu einem Innovationsprozess dazu gehören, als auch die Zusammenarbeit in einem Team (vgl. Abbildung 11). Durch Zusammenfassen und Zerlegen einzelner Elemente konnten verschiedene Themenbereiche erstellt und bearbeitet werden. Dabei sollte darauf geachtet werden, wie diese einzelnen Themenbereiche zu definieren sein könnten. Mithilfe der folgenden Fragen.

- Können die einzelnen Elemente dieser Themenbereiche noch genauer definiert werden?
- Sind weitere Elemente notwendig, um diesen Themenbereich zu verstehen bzw. umzusetzen?
- Wie könnte eine mögliche Umsetzung aussehen?
- Wie kann man diesen Themenbereich in Beziehung zu den anderen Themenbereichen setzen?

wurden diese Bereiche genauer definiert. Es hat sich herausgestellt, dass in dem Brainstorming zum einen Themenbereiche gefunden wurden, die direkt vom Prozess umgesetzt werden sollen und die den Prozess beeinflussen. Indirekt beeinflussen der Themenbereich Motivation (vgl. Abschnitt 4.2) und Anforderungen wie Transparenz und Kreativität das Prozessmodell. Direkt in dem Prozess umgesetzt werden müssen die Themenbereiche Ideen, Probleme, Bewertung, Vorschläge und Gewinne.
Durch die erste Definition der einzelnen Themenbereiche wurde das erste Mal eine Art Ablauf sichtbar. Ebenso wurde durch die Seminararbeit *Entwicklungsstufen* deutlich, dass ein Prozessdurchlauf folgende Themenbereiche abdecken muss.

- Problem
- Vorschläge
- Ideen
- Vorhaben und Business Case

Nach mehrfachen Diskussionen innerhalb der Projektgruppe wurden die bereits genannten Themenbereiche als notwendig erachtet, um einen möglichst intuitiven Prozessdurchlauf zu gewährleisten. Innerhalb der Diskussionen wurde die Abfolge der einzelnen Prozessschritte verdeutlicht und auf obengenannte Reihenfolge geeinigt. Die Prozessschritte wurden in Gruppenarbeiten weiter ausgearbeitet und die Übergänge definiert.
Der allgemeine Prozessdurchlauf startet bei der Problem- oder Vorschlageingabe (vgl. Abbildung 12). Der Anwender kann entweder das Problem eingeben oder einen Vorschlag zu einem Problem. Diese Probleme und Vorschläge können kommentiert werden. Dadurch soll gewährleistet werden, dass aus ausgearbeiteten Vorschlägen eine Idee entsteht.
Eine Idee wird dann von einem Team, welches sich in der Ausarbeitung der Vorschläge gefunden hat, weiter bearbeitet. Eine ausgearbeitete Idee wird dann als Vorhaben verstanden. Innerhalb des Vorhaben können Daten und Informationen gesammelt werden, die für den Business Case notwendig sind. Falls nicht genug Informationen vorliegen oder evtl. andere Lösungsansätze betrachtet werden sollen, ist es möglich einen Prozessschritt zurück zu gehen und diese dort einzubinden.
Im Prozessschritt *Business Case* wird ein solcher durchgeführt und entschieden, ob das Vorhaben durchgeführt werden kann, das Vorhaben nochmal bearbeitet werden muss oder ob es nicht durchführbar ist. Diese Entscheidung erfolgt über die Bewertung des Business Cases. Nach dem Business Case endet der Prozess.
In den letzten drei Schritten ist es möglich, Bewertungen abzugeben. Mit dieser Funktionalität können die besten Ideen und Vorhaben gewählt und weiter entwickelt werden. Innerhalb des gesamten Prozesses soll es möglich sein zu archivieren und kommentieren.
Innerhalb der Prozesschritte *Vorhaben* und *Business Case* ist es möglich, sich als Mitarbeiter zu beteiligen.

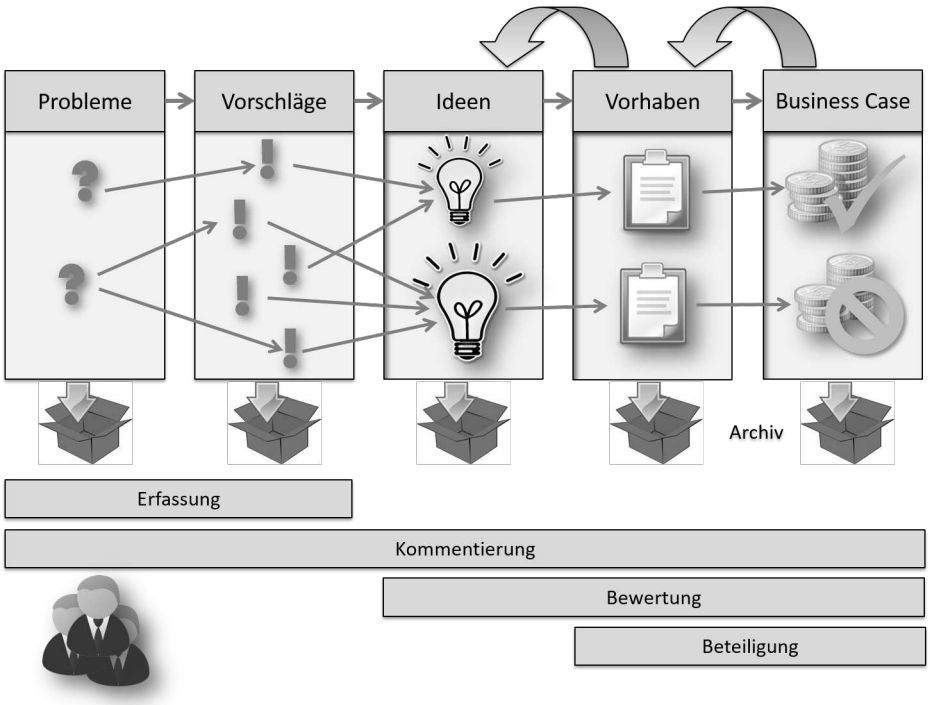


Abbildung 12: Prozessmodel Version 1.0

Die Schwachstellen dieser ersten Version des Prozessmodells liegen zum einen in dem Prozesseinstieg. Aus jedem Problem kann ein Vorschlag entstehen und jedem Vorschlag liegt ein Problem zu Grunde. Doch beide Begrifflichkeiten können im weitesten Sinne auch als Idee verstanden werden. Die klare Trennung der Begrifflichkeiten ist nicht gewährleistet und steigert die Eintrittsbarriere. Zum anderen fehlt ein Feedback zu den Projekten. Dadurch ist es möglich, dass aus den umgesetzten Projekten Wissen gezogen werden kann und mögliche Fehler in darauffolgenden Projekten zu vermieden werden können.
In der Version 2.0 (vgl. Abbildung 13) ist die Einstiegsbarriere vermindert durch das Löschen des Schrittes *Idee*. Dadurch soll verhindert werden, dass die Begrifflichkeiten zur Verwirrung führen bei dem Einstieg in den Prozess. Im Vorhaben werden weiterhin alle wichtigen Informationen für den Business Case gesammelt und für diesen bereitgestellt. Gleichzeitig wurde der Prozessschritt Feedback eingeführt und es wurden klare Beziehungen zwischen den einzelnen Schritten definiert. Es besteht eine M:N-Beziehung zwischen Problem und Vorschlag, danach wird schnellstmöglich eine 1:1-Beziehung zwischen den einzelnen Prozessschritten sichergestellt.

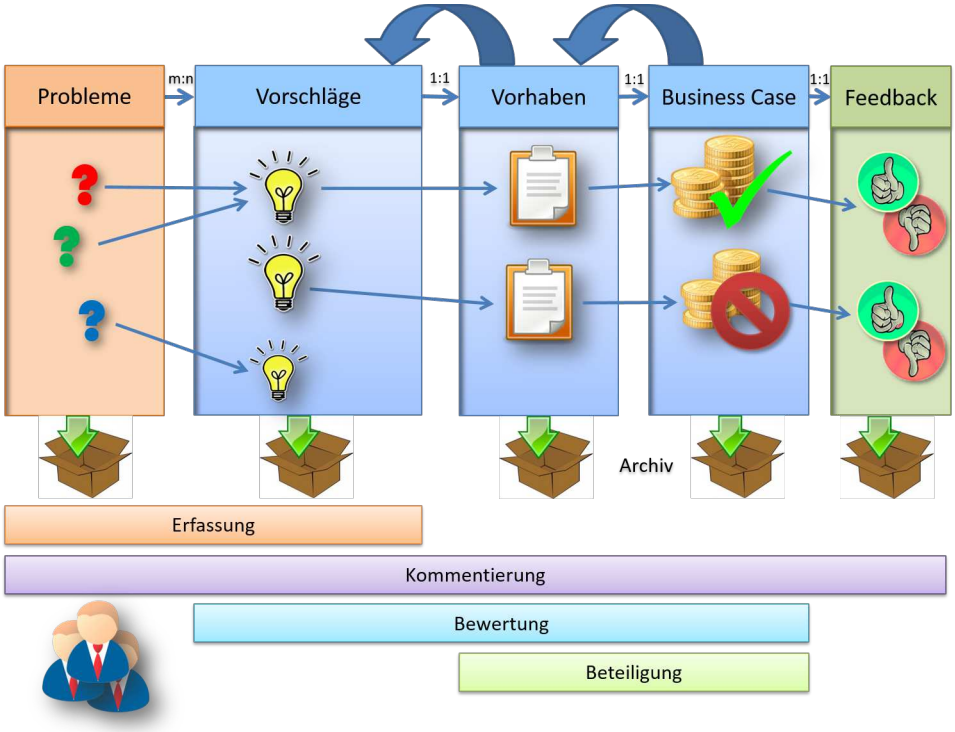


Abbildung 13: Prozessmodel Version 2.0

Während der Realisierungsphase des Projektes stellte sich heraus, dass die Umsetzung des Prozesses zu starr und wenig intuitiv verläuft. Das Unterscheiden zwischen Problem und Vorschlag während des Einstiegs in den Prozess, führte weiterhin zu vielen Problemen. Um diese Hürde zu nehmen, wurde der Einstiegsprozessschritt in *Challenge* umbenannt. Challenges können sowohl Probleme, als auch Wünsche oder auch Vorschläge sein. Ebenso wurde der Prozessschritt Vorhaben in Projekt geändert (vgl. Abbildung 14). Der Begriff Projekt ist für den Benutzer verständlicher als Vorhaben.
Der Prozessverlauf wurde ebenfalls angepasst. So ist es nicht mehr möglich, von einem Vorhaben/ Projekt zu einem Lösungsvorschlag zu gelangen. Bereits während des Erstellens erster Vorhaben hat sich herausgestellt, dass das Wechseln in den vorherigen Prozessschritt keinen Mehrwert für den Benutzer bietet und nicht benötigt wird. Die 1:1 Beziehung bleibt aber zwischen dem Lösungsvorschlag und Projekt bestehen.

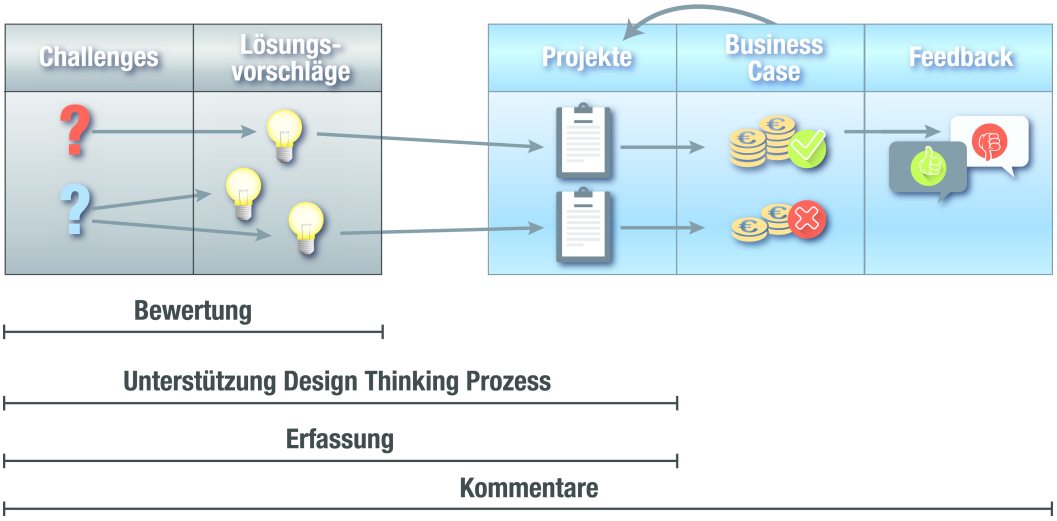


Abbildung 14: Prozessmodel Version 3.0

Die dritte Version stellt somit den endgültigen Verlaufs des Prozesses dar. Die Schritte Challenge, Lösungsvorschlag, Projekt, Business Case und Feedback sind die zu durchlaufenden Etappen innerhalb des Prozesses. Die Challenge stellt dabei den Beginn des Prozesses dar.

4.3.2. Challenge

Um einen möglichst allgemeingültigen Einstieg in den Prozess der Webanwendung IMPACT zu gewährleisten, beginnt der Prozess mit einer Challenge. Eine Challenge kann sowohl ein Problem, Idee als auch einen Vorschlag beinhalten.
Der Prozess startet mit einer Idee oder einem Problem eines Mitarbeiters. Dieses wird als eine Challenge in der Webanwendung eingetragen. In der Challenge wird zunächst zusammengefasst, um was es sich bei dieser Idee oder Problem handelt, welches Ziel verfolgt werden soll und welche Hindernisse entstehen könnten, um dieses Ziel zu erreichen. Zusätzlich können Dateien angehängt und Schlagwörter gesetzt werden sowie eine Auswahl stattfinden, ob die Challenge Anonym eingestellt werden soll. Alle Informationen und Felder, welche in einer Challenge aufgeführt werden, sind in der Tabelle 4 zu finden. Des Weiteren können Lösungsvorschläge abgegeben werden.

Bezeichnung	Pflichtfeld	Komponente	Beschreibung
Challeng-beschreibung	Ja	RichTextArea	Textuelle Erfassung und Beschreibung des Problems
Challengetitel	Ja	TextField	Textuelle Erfassung eines Titels
Erfassungs-zeitpunkt	Automatisch Timestamp	Datum und Uhrzeit der ersten Erfassung	
Anonym	Ja	CheckBox	Entscheidung, ob die Challenge anonym erfasst werden soll. Vorauswahl „false“.
Challengeziel	Ja	TextField	Textuelle Erfassung des Ziels
Challenge-hindernis	Ja	TextField	Textuelle Erfassung von Hindernissen
Schlagwörter	Nein	TokenField	Textuelle Erfassung von Schlagwörtern
Dateiupload	Nein	CollectMultiFileUp	Auswahl zum Anhängen einer Datei
Kommentare	Nein	TextArea	Textuelle Erfassung eines Kommentars
Bewertung	Ja	RatingStars	Visuelle Erfassung aller Bewertungen

Tabelle 4: Felder der Challenge

4.3.3. Lösungsvorschlag

Lösungsvorschläge bieten konkrete Ideen zur Behebung eines Problems oder zur Umsetzung einer Idee. Innerhalb eines Lösungsvorschlages findet eine genauere Beschreibung der Umsetzungsidee statt, welche von jedem Anwender einsehbar ist. Alle weiteren Informationen, die gesammelt werden, sind in der Tabelle 5 zu finden. Es ist möglich, dass andere Mitarbeiter Challenges und Lösungsvorschläge bewerten und kommentieren. Somit soll gewährleistet werden, dass Challenges und Lösungsvorschläge weiterentwickelt werden und die Meinung der anderen Mitarbeiter zusätzlich miteinbezogen wird. Darüber hinaus dienen die Lösungsvorschläge, die die beste Bewertung repräsentieren, als Grundlage für Projekte.

Bezeichnung	Pflichtfeld	Komponente	Beschreibung
Lösungsvorschlagbeschreibung	Ja	RichTextArea	Textuelle Erfassung und Beschreibung des Lösungsvorschlag
Lösungsvorschlagtitel	Ja	TextField	Textuelle Erfassung eines Titels
Erfassungszeitpunkt	Automatisch	Timestamp	Datum und Uhrzeit der ersten Erfassung
Anonym	Nein	CheckBox	Entscheidung, ob der Lösungsvorschlag anonym erfasst werden soll. Vorauswahl „false“.
Schlagwörter	Nein	TokenField	Textuelle Erfassung von Schlagwörtern
Dateiupload	Nein	CollectMultiFileUp	Auswahl zum Anhängen einer Datei
Kommentare	Nein	TextArea	Textuelle Erfassung eines Kommentars
Bewertung	Nein	RatingStars	Visuelle Erfassung aller Bewertungen

Tabelle 5: Felder des Lösungsvorschlags

4.3.4. Übergang: Lösungsvorschlag zu Projekt

Der Übergang von einer Challenge und Lösungsvorschlag zu einem Projekt wird durch den Challengeersteller bestimmt. Nachdem die Challenge komplett ausgefüllt und mindestens eine Bewertung sowie ein Lösungsvorschlag abgegeben wurde, besteht die Möglichkeit, ein Projekt zu starten. Die Basis eines Projektes bildet ein Lösungsvorschlag. Der Lösungsvorschlag beinhaltet die Idee, die auch umgesetzt und weiter verfolgt werden soll. Ebenso können verschiedene Lösungsvorschläge kombiniert und zu einem zusammengefasst werden. Am Ende des Übergangs wird ein neues Projekt erstellt.

4.3.5. Projekt

Ein Projekt besteht aus einem Projektteam und einem Teamleiter, welche sich um eine mögliche Umsetzung des Lösungsvorschlages Gedanken machen. Dabei werden notwendige Arbeitsschritte, Gewinne, Anschaffungen, Risikobetrachtungen o.ä. definiert und gesammelt. Diese Informationen stellen die Grundlage für die finanzielle Betrachtung des Projektes dar. Weitere zu sammelnde Informationen können der Tabelle 6 entnommen

werden. Innerhalb dieser Prozessphase ist es wichtig, dass das Projektteam kreativ arbeiten kann. Dafür steht in dieser Phase ein kollaborativer Raum zur Verfügung. Mit diesem besteht die Möglichkeit, dass das Team sich mit Hilfe eines Whiteboards austauscht, Ideen festhält und weiterentwickelt.
Darüber hinaus ist es wichtig, dass innerhalb des Teams eine hohe Kommunikation stattfindet. Um das zu erleichtern, ist innerhalb der Webanwendung eine Chatfunktion implementiert. Andere Mitarbeiter, welche nicht im Projektteam angesiedelt sind, sollen das Projekt einsehen und kommentieren können. Dadurch kann gewährleistet werden, dass der Fortschritt eines Projektes transparent dargestellt wird und andere Anwender eine Resonanz auf diesen geben können.
Der Teamleiter entscheidet mit dem Team zusammen, wann das Projekt abgeschlossen werden kann und alle notwendigen Informationen für einen Business Case gesammelt werden. Um den Fortschritt zu verfolgen, werden die vollständigen Informationen markiert und in der Checkliste wird dieser anschließend dargestellt.

Bezeichnung	Pflichtfeld	Komponente	Beschreibung
Projekttitel	Ja	TextField	Textuelle Erfassung eines Titels
Projekt-beschreibung	Ja	RichTextArea	Detaillierte Beschreibung des Projekts
Risiko-beschreibung	Ja	RichTextArea	ausführliche Beschreibung der Risiken eines Projektes
Nutzungsdauer	Ja	RichTextArea	Wie lange das Projektergebnis genutzt wird
Nutzungsdauer in Jahren	Ja	TextField	Textuelle Erfassung der Nutzungsdauer in Jahren
Projektteam	Ja	Tabelle	Mitarbeiter müssen mit Name, Qualifikationen und verfügbare Zeit eingetragen werden.
Nutzer-argumentation	Ja	RichTextArea	Sammeln von Nutzen und Zielen
Gewinne	Ja	RichTextArea	Gewinne und Einsparungen müssen dargestellt werden mit u.a. den betroffenen Bereichen
Tätigkeiten	Ja	RichTextArea	Aufgaben die in einem Projekt ausgeführt werden müssen
Anschaffung	Ja	RichTextArea	Benötigte Ressourcen in einem Projekt
Dateiupload	Nein	CollectMulti-FileUpload	Auswahl zum Anhängen einer Datei
Checkliste	Nein	Liste	Die Checkliste zeigt an, welche Informationen noch gesammelt werden müssen
Check-listenelemente	Ja	Checkboxen	Controlle gibt an, ob die gesammelten Informationen der Pflichtfelder ausreichen
Schlussfolgerung oder Empfehlung	Ja	RichTextArea	Controller verschriftliche mögliche Verbesserungen und Empfehlung nach dem Prüfen des Business Cases

Tabelle 6: Felder des Projektes

4.3.6. Übergang: Projekt zu Business Case

Die Überführung eines Projektes zu einem Business Case erfolgt, wie bereits erwähnt, durch den Teamleiter. Die gesammelten Daten werden direkt an den Business Case weitergereicht und dort verarbeitet.

4.3.7. Business Case

Der eigentliche Business Case wird außerhalb der Anwendung IMPACT ausgeführt. Innerhalb der Anwendung werden lediglich die Informationen gesammelt, die für die Bearbeitung notwendig sind. Es besteht die Möglichkeit von dem Prozessschritt Business Case in den vorherigen Prozessschritt Projekt zu springen. Fällt innerhalb des Business Cases auf, dass Informationen fehlen oder nur ungenügend vorliegen, können diese von dem Projektteam innerhalb des Projektes überarbeitet werden (siehe Abbildung 14). Darüber hinaus kann der Controller weitere Empfehlungen und Anmerkungen geben. Weitere zu sammelnde Informationen können Tabelle 7 entnommen werden.

Bezeichnung	Pflichtfeld	Komponente	Beschreibung
Projekttitel	Nein	TextField	Textuelle Erfassung eines Titels
Projekt-beschreibung	Nein	RichTextArea	Detaillierte Beschreibung des Projekts
Risiko-beschreibung	Ja	RichTextArea	ausführliche Beschreibung der Risiken eines Projektes
Nutzungsdauer	Ja	RichTextArea	Wie lange das Projektergebnis genutzt wird
Nutzungsdauer in Jahren	Ja	TextField	Textuelle Erfassung der Nutzungsdauer in Jahren
Projektteam	Ja	Tabelle	Mitarbeiter müssen mit Name, Qualifikationen und verfügbare Zeit eingetragen werden.
Nutzer-argumentation	Ja	RichTextArea	Sammeln von Nutzen und Zielen
Gewinne	Ja	Tabelle	Gewinne und Einsparungen müssen dargestellt werden mit u.a. den betroffenen Bereichen
Tätigkeiten	Ja	Tabelle	Aufgaben die in einem Projekt ausgeführt werden müssen
Anschaffung	Nein	Tabelle	Benötigte Ressourcen in einem Projekt
Dateiupload	Nein	CollectMulti-FileUpload	Auswahl zum Anhängen einer Datei
Kommentare	Nein	TextArea	Textuelle Erfassung eines Kommentars
Checklisten-elemente	Ja	Checkboxen	Projektleiter gibt an, ob die gesammelten Informationen der Pflichtfelder ausreichen
Checkliste	Nein	Liste	Die Checkliste zeigt an, welche Informationen noch gesammelt werden müssen

Tabelle 7: Felder des Business Cases

4.3.8. Übergang: Business Case zu Feedback

Liegen alle Pflichtinformationen vor, kann der Controller den Business Case durchführen und zur Bewertung an den Entscheider weiterreichen. Erfolgt eine positive Bewertung des Business Cases, wird das Projekt durchgeführt. Erfolgt hingegen eine negative Bewertung, wird das Projekt nicht durchgeführt oder es wird wieder zurück in die Projektphase geschoben. Nach dem Business Case erfolgt ein Feedback zu dem Business Case.

4.3.9. Feedback

Innerhalb des Prozessschrittes Feedback kann geschildert werden, welches die nächsten umzusetzenden Ziele darstellen. Ebenso ist es möglich ein Feedback zu erstellen, welches dem Anwender die Möglichkeit bietet, seine Meinung zum Business Case zu äußern. Durch diesen Prozessschritt ist es möglich aus bisherigen Fehlern innerhalb der Prozessschritte oder der Umsetzungsphase zu lernen. Gleichzeitig können Projekte, die anderen Mitarbeitern in positivem Maße gefallen haben, weiter verfolgt werden. Die dafür benötigten Felder sind in der Tabelle 8 aufgelistet.

Bezeichnung	Pflichtfel	Komponente	Beschreibung
Feedbacktitel	Nein	TextField	Textuelle Erfassung eines Titels
Feedback	Nein	TextField	Textuelle Erfassung des Feedbacks

Tabelle 8: Felder des Feedbacks

4.4. Technologieentscheidung

In diesem Kapitel werden die Entscheidungen für eingesetzte Technologien begründet und aufgeführt.

4.4.1. Webanwendung

Webanwendungen stellen dynamisch Funktionen und Inhalte zur Verfügung. Die auf dem Client anzuzeigenden Dokumente und Oberflächen werden dazu zunächst auf einem Webserver erzeugt und anschließend ausgeliefert. Der Client zeigt die übermittelten Daten mit Hilfe eines Webbrowsers an (vgl.[Bun14]).
Der Webbrowser als Laufzeitumgebung stellt eine Abstraktionsschicht zwischen der Anwendung und dem Betriebssystem dar. Ein solcher Webbrowser ist auf den meisten Clients, besonders in Unternehmen, bereits installiert. Auch mobile Endgeräte verfügen über eine solche Software (vgl.[CHC10], S.39).

Mit der genannten Unabhängigkeit vom Betriebssystem und der damit einhergehenden weiten Verbreitung sind Webanwendungen gut für eine Vielzahl an Anwendern geeignet. Außerdem spart die clientseitige Plattformunabhängigkeit Aufwand und Kosten bei der Einführung der neuen Software und macht diese damit wirtschaftlicher (vgl.[CHC10], S.40). Serverseitig hat IMPACT speziellere Anforderungen. Da der Server in der Regel im Zugriff der Entwickler und Administratoren liegt, können die entsprechenden Voraussetzungen allerdings geschaffen werden.
Weitere Vorteile ergeben sich bei der Wartung. Änderungen am Programm können einfach und kostengünstig zentral auf dem Server durchgeführt werden. Eine Verteilung der Änderungen ist nicht notwendig. Stattdessen steht die neue Version sofort allen Clients zur Verfügung. Damit lässt sich außerdem das Sicherheitsproblem nicht aktueller Software minimieren. Kritische Sicherheitslücken können zeitnah und effizient auf dem Server geschlossen werden. Zusätzlich bieten Webanwendungen eine höhere Datensicherheit, da die Daten nicht auf dem Client gespeichert werden (vgl.[CHC10], S.40 f.).
Ein Nachteil einer Webanwendung ist, dass ständig eine Verbindung zum Server bestehen muss. Bricht die Verbindung ab, ist die Anwendung nicht mehr erreichbar (vgl.[CHC10], S.41). Da IMPACT in Unternehmen eingesetzt werden soll, stellt diese Anforderung kein Problem dar, weil die Clients ins Unternehmensnetz eingebunden und somit jederzeit online sind. Die Datenrate sollte ebenfalls den Anforderungen der Webanwendung entsprechen. Da bei IMPACT nur Texte bzw. Office-Dokumente zwischen Client und Server übertragen werden, ist das Unternehmensnetz mehr als ausreichend. Anwendungen wie Videostreaming haben deutlich höhere Anforderungen und kommen bisher nur beim Tutorial zum Einsatz. Da Webanwendungen im Browser aufgerufen werden, wird ein Teil des Bildschirms durch dessen Steuerelemente eingenommen. Damit verringert sich der durch die eigentliche Anwendung nutzbarer Platz und kann die Bedienbarkeit durch den Benutzer einschränken (vgl.[CHC10], S.41f.).
Ein weiterer Nachteil ist, dass Webanwendungen durch die Verteilung auf Server und Client schwieriger zu entwickeln sind, als nicht-verteilte Anwendungen. Dadurch, dass Teile im Webbrowser und Teile auf dem Webserver ablaufen, wird außerdem die Fehlersuche erschwert (vgl.[CHC10], S.43). Aufgrund der für diesen Anwendungsfall zahlreichen Vorteile, welche die Nachteile überwiegen, wurde entschieden, IMPACT als Webanwendung zu entwickeln. Lediglich der Nachteil, dass IMPACT nicht offline genutzt werden kann, stellt ein Problem dar. Sollten sich Mitarbeiter auf Geschäftsreise befinden und keinen Zugang zum Unternehmensnetz haben, besteht keine Möglichkeit mit der Plattform zu arbeiten. Das ist besonders deshalb kritisch, da erfahrungsgemäß viele Ideen und kreative Vorschläge außerhalb des Büros entstehen können. Lösung wäre eine Erreichbarkeit der

Webanwendung über das Internet oder eine App, welche die entsprechenden Daten übermitteln kann. Darüber hinaus existieren auch für weitere angesprochene Nachteile bereits Lösungsansätze (vgl.[CHC10], S.43ff.).

4.4.2. Java

Dieser Abschnitt befasst mit der Entscheidung *Java*[8] als Programmiersprache einzusetzen und *Java Plattform, Enterprise Edition*[9] (kurz *Java EE*) zu verwenden. Aufgrund der Vorkenntnisse und Erfahrungen der Mehrheit der Gruppenmitglieder mit der Programmiersprache wurde *Java* ausgewählt, um damit die Logik zu implementieren. Mit der Entscheidung die Plattform als Webanwendung zu realisieren, fiel die Wahl auf *Java EE*. Dieses Framework stellt eine Sammlung von Spezifikationen für die Implementierung von auf Java basierenden verteilten Anwendungen dar. Im sogenannten Web-Profil sind nur ein Teil dieser Spezifikation enthalten, die für die Implementierung von Webanwendungen benötigt werden. Die wichtigsten Spezifikation betreffen die persistente Speicherung, die Kommunikation zwischen einzelnen Anwendungen auf verschiedenen Rechnern und Schnittstellen für Webanwendungen. Eine vollständige Auflistung der enthaltenen Spezifikationen findet sich in der *JavaTM Platform, Enterprise Edition 7 (Java EE 7) Web Profile Specification*[10].

4.4.3. Vaadin

Nachdem in den vorherigen Abschnitten die Entscheidung der Implementierung der Webanwendung unter der Verwendung von Java als Programmiersprache und den Java EE Spezifikation erläutert wurde, befasst sich dieser Abschnitt mit der Entscheidung, Vaadin[11] als Framework zu verwenden. Dazu wird im Folgenden zunächst erläutert, was Vaadin darstellt und im Zuge dessen ein Vergleich zu anderen üblichen Technologien gezogen.
Vaadin basiert auf dem Google Web Toolkit (GWT)[12] und stellt eine Open-Source-Framework zur Entwicklung von Rich Internet Applications (RIA) dar. Es erlaubt die vollständige Implementierung von komplexen browserbasierten Anwendung in Java und unterstützt alle geforderten Spezifikation des Java EE[13].

[8] https://www.java.com/de/download/faq/java8.xml
[9] http://www.oracle.com/technetwork/java/javaee
[10] http://download.oracle.com/otn-pub/jcp/java_ee-7-fr-eval-spec/WebProfile.pdf
[11] https://vaadin.com/home
[12] http://www.gwtproject.org
[13] https://vaadin.com/javaee

Die Entscheidung fiel auf dieses Framework, da gegenüber klassischen JSP [14] der Java-Programmcode nicht in HTML-Seiten integriert werden muss, sondern aus dem Programmcode dynamische HTML-Seiten generiert werden [15]. *Vaadin* liefert dazu eine Reihe von vorgefertigten Klassen, die klassischen HTML-Komponenten (Button, Inputs, Link) nach empfunden sind, es erlaubt die Navigation über URL-Fragmente, ein simple zu verwendendes Sessionmanagement, leicht zu integrierte Server-Push-Technologie und Lebenszyklen für *UI*. Ein weiterer Vorteil von Vaadin ist die serverseitige Datenvalidierung, wodurch der Server über möglicherweise veränderte Client-Daten informiert wird und diese Manipulation unterbindet[16]. Ein Nachteil des Frameworks sind die höhere Anzahl an Server-Aufrufen, da die meisten UI-Events serverseitig ausgeführt und verarbeitet werden. Als Ergänzung existieren sogenannte *Widgets Sets*, die clientseitig durch JavaScript im Browser ausgeführt werden und die Anzahl der Server-Aufrufe verringern.
Weiter ausschlaggebend für die Verwendung des Frameworks z. B. gegenüber GWT war die gute Dokumentation, die Vielzahl an verfügbaren client- und serverseitigen Erweiterungen, die Unterstützung von mobilen Endgeräten sowie die Möglichkeit der Verwendung des *MVP-Patterns*. Eine komplexer Vergleich des Vaadin-Frameworks gegenüber anderen Frameworks befindet sich auf der Homepage des Frameworks[17].
Im Verlauf des Projektes wurden einige Erweiterungen, sogenannte *Addons* eingebunden, die im folgenden kurz erläutert werden.

Vaadin CDI

Diese Erweiterung ermöglicht es, *CDI* (vgl. Abschnitt 4.4.7) innerhalb von *Vaadin* zu nutzen.

Vaadin Wizzard

Dieses *Addon* bietet eine Assistenten-Funktionalität, wie in Software-Installationsroutinen üblich, welcher aus mehreren Teil-Schritten besteht und ermöglicht es, dem Nutzer durch eine komplexere Aufgabenstellung in Teilschritten durch zuführen.

FilteringTable

Dieses *Widget Set* erlaubt den Nutzer Daten in einer Tabelle zu sortieren und nach bestimmten Einträgen zu filtern. Der Vorteil dieser Erweiterung ist, dass diese vollständig clientseitig ausgeführt wird und somit keine zusätzlichen Serveraufrufe verursacht.

[14] `http://www.oracle.com/technetwork/java/faq-137059.html`
[15] `https://vaadin.com/features`
[16] `https://vaadin.com/features`
[17] `https://vaadin.com/comparison`

RatingStars

Dieses *Addon* bietet eine typische Bewertungsoberfläche mit Sternen an. Diese erlaubt dem Nutzer mittels gebräuchlichen Elementen eine Bewertung abzugeben.

NumberField

Diese Erweiterung stellt ein Eingabefeld dar, das nur die Eingabe von Ziffern erlaubt. Die Validierung wird clientseitig ausgeführt und ist somit effizienter gegenüber der serverseitigen Validierung.

CollabSketch

Dieses *Addon* erweitert *Vaadin* um ein kollaboratives Whiteboard. Es besteht aus zwei Teilen: einer clientseitigen Canvas-Oberfläche, die es dem Nutzer erlaubt mittels JavaScript zu zeichnen und einem serverseitigen *Container*, der dazu dient, die Zeichnung zwischen mehreren Clients zu synchronisieren.

Stackpanel

Dieses *Widget Set* basiert auf *JavaScript* und *CSS*. Es erlaubt auf und zu klappbare *Panels*, um Informationen bei Bedarf ein und auszublenden, ohne dass dabei Kommunikation zum Server nötig ist oder die Seite neu geladen werden muss.

4.4.4. Maven

Bei Maven handelt es sich um ein Build-Automation-Tool, das hauptsächlich für Java Projekte eingesetzt wird. Mit Hilfe von Maven kann das Projekt mit allen Abhängigkeiten gebaut und die jeweiligen war- oder jar-Dateien erzeugt werden. Alle dazu notwendigen Skripte werden in einer zentralen Datei, der pom.xml, gespeichert.
Der Hauptvorteil von Maven ergibt sich aus der Komplexitätsreduzierung des Build-Prozesses, da dieser automatisiert abläuft. Abhängigkeiten werden aufgelöst, aktualisiert und bei Bedarf heruntergeladen, der Quellcode wird kompiliert und das Projekt zusammen mit anderen nicht kompilierbaren Dateien in einen Container verpackt.
Darüber hinaus bietet Maven einen einheitlichen Build-Prozess, der bei allen Projekten gleich ist. Dies spart Einarbeitungszeit beim Navigieren durch mehrere Projekte.
Maven bietet die Möglichkeit, weitere Informationen über das Projekt zu erhalten. Dazu gehören Logs oder Abhängigkeiten unter den einzelnen Programmteilen. Damit erhöht sich die Informationsqualität des erzeugten Programms.
Um ebenfalls die Qualität des Entwicklungsprozesses zu erhöhen, sind Best Practises wie Spezifikationsdefinition oder das Erzeugen von Testberichten in den Build Zyklus integriert. Darüber hinaus bietet Maven Unterstützung im Projekt Workflow, wie z. B. Release Management oder das Nachverfolgen von Änderungen (vgl.[Apa16]).

Aufgrund der Vorteile der Komplexitätsreduzierung und Qualitätserhöhung wurde entschieden, Maven zur Entwicklung von IMPACT einzusetzen. Damit wird außerdem die Entwicklung effizienter gestaltet, da alle Abhängigkeiten global bekannt gemacht werden und somit das Fehlerpotential beim Erzeugen des Programms reduziert wird.

4.4.5. Shiro

Daniel Die Wahl der Technologie zur Sicherung der entstehenden Applikation ist ebenfalls Teil des Projektes. Darunter fällt die Entscheidung, auf welche Weise User registriert und eingeloggt werden, darüber hinaus, wie das Sessionmanagement und das Rechtekonzept umgesetzt wird. Neben der Möglichkeit, sämtliche Aspekte in vollständiger Eigenentwicklung umzusetzen, existieren eine Reihe von Frameworks, durch welche der Entwickler bei der Implementierung unterstützt wird.
Gemäß den Anforderungen in Kapitel 3.2 müssen neue Nutzer bei der Registrierung angelegt und persistiert sowie Passwörter sicher gespeichert werden. Weiterhin muss gewährleistet sein, dass der Nutzer ausschließlich nur diejenigen Inhalte einsehen kann, die gemäß Kapitel 2.1 für ihn definiert wurden. Neben den globalen Rollen, welche fest deklariert werden, können Nutzer spezielle Rechte erhalten, z. B. soll beim Erstellen eines Kommentars für den Nutzer das exklusive Recht bestehen, im Anschluss Änderungen daran durchzuführen. Diese speziellen Rechte müssen berücksichtigt werden, sodass ein festes Rollenmanagement nicht ausreicht. Ein so genanntes instanzbasiertes Rechtemanagement, bei dem dies unterstützt wird, ist damit ebenfalls ein Kriterium.
Mögliche Frameworks, die die Kriterien erfüllen, sind zum einen Spring Security[18], sowie Apache Shiro[19]. Ersteres bietet durchaus mächtige Möglichkeiten zur Absicherung von Webapplikationen. Es stellt jedoch lediglich eine Erweiterung des gesamten Spring Frameworks[20] dar, welches viele weitere Anwendungsfälle ermöglicht. Damit würden zwar unter Umständen mehr Möglichkeiten entstehen, für gleichen Nutzen ist jedoch mit höherer Komplexität zu rechnen.
Apache Shiro ist eine Standalone Security Framework. Es bietet auf der Website eine umfassende Dokumentation[21] über den Leistungsumfang. Es unterstützt die genannten Anforderungen und bietet ein instanzbasiertes Rechtemanagement. Es stellt darüber hinaus eine Vielzahl von Werkzeugen zur Verfügung, mit denen die Anwendung abgesichert werden kann. Der Entwickler wird dabei effektiv unterstützt. Beispielsweise werden sichere Möglichkeiten zum Verschlüsseln der Nutzerpasswörter nicht nur geboten, sondern bei entsprechender Nichtnutzung dessen Fehlermeldungen an den Entwickler ausgegeben.

[18] http://projects.spring.io/spring-security/
[19] http://shiro.apache.org/
[20] https://spring.io/
[21] http://shiro.apache.org/reference.html

Im Vergleich zu Spring Security ist damit eine geringere Komplexität des Frameworks zu erwarten. Damit ist ein niedrigerer Aufwand bei der Einarbeitung und Implementierung verbunden, bei gleichzeitig vollständiger Unterstützung aller genannten Kriterien.

4.4.6. Architektur

Im folgenden werden die grundlegenden Begriffe in Bezug auf *MVP* (Model-View-Presenter) erklärt.

Model-View-Presenter

Bei dem Entwurfsmuster *MVP* handelt es sich um eine Erweiterung des *Model-View-Controllers (MVC)*. Die folgende Abbildung zeigt die beiden Entwurfsmuster und verdeutlicht die Beziehungen:

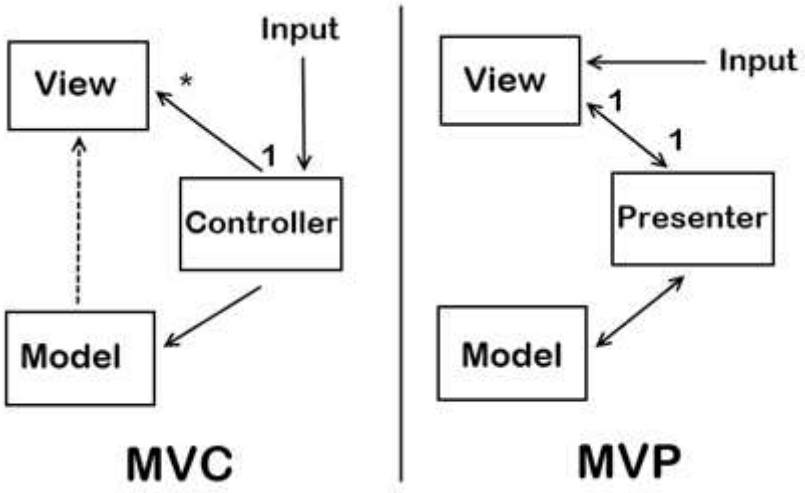


Abbildung 15: Unterschied zwischen MVP und MVC, adaptiert nach [Sta08]

Das Entwurfsmuster ist in drei Bereiche unterteilt. Das *Model* stellt dabei eine Klasse dar, in welcher die benötigten Daten und Zustände gespeichert werden.
Dabei hat das *Model* weder auf die *View* noch auf den *Presenter* direkten Zugriff. Ein direkte Kommunikation bzw. Zugriff erfolgt ausschließlich vom *Presenter* aus.
Es ist somit die Aufgabe des *Presenters*, als eine Art Vermittler zwischen der *View* und dem *Model* zu agieren. Im Detail bedeutet das, dass Daten vom *Model* durch den *Presenter* in der *View* angezeigt werden (können). Umgekehrt werden Daten durch die *View* über den *Presenter* im *Model* gespeichert. Der *Presenter* ist somit für die Steuerung logischer Abläufe zuständig. Dadurch, dass die *View* nicht an das *Model* gebunden ist und zudem keine Abhängigkeiten nach außen aufweist, lässt sie sich in anderen Kontexten problemlos wiederverwenden (vgl. [BAE+14], S.216f.).

Dieses aufgezeigte *MVP*-Entwurfsmuster unterteilt sich zusätzlich in die zwei Varianten *Passive View* und *Supvervising Controller* auf. Bei der *Supervising Controller* Variante verfügt sowohl die *View*, als auch der *Presenter* Kenntnisse über das *Model*. Hier erfolgt die Datensynchronisierung nicht direkt durch den *Presenter*, sondern über eine Datenanbindung von der *View* zum *Model*. Die Datenanbindung kümmert sich hierbei eigenständig um die Datensynchronisation, während der *Presenter* lediglich für die Verarbeitung der reinen Logik verantwortlich ist. Im Gegensatz zur *Passive View* kann die *View* hier auch Logik enthalten. Demgegenüber steht die *Passive View*. Wie der Name dieser Variante bereits darlegt, verhält sich die *View* hier in einer passiven Form. Das bedeutet, dass in diesem Fall die gesamte Logik im *Presenter* liegt. Der *Presenter* trifft dabei alle Entscheidungen und ist für die Steuerung der *View* verantwortlich. Die *View* kümmert sich anschließend lediglich um die Darstellung. Diesbezüglich hat sich die Projektgruppe für die *Passive View*-Variante entschieden. Gründe dafür waren insbesondere, dass die *Passive View* gegenüber dem *Supervising Controller* die Test- und Erweiterbarkeit verbessert, da lediglich einfachster Quellcode zur Ein- und Ausgabe in der Ansicht vorzufinden ist (vgl.[BAE+14], S.217ff.).

Humble View

In dem vorherigen Abschnitt wurde aufgezeigt, dass die Projektgruppe sich im Bereich des MVP für die *Passive View*-Variante entschieden hat und es wurde zusätzlich aufgezeigt, dass durch diese Entscheidung u.a. die Testbarkeit verbessert wird. Um diesen Vorteil weiter auszubauen, wurde die Entscheidung getroffen, dass so wenig Logik wie möglich innerhalb der Oberfläche erfolgen soll. In diesem Fall kann die sogenannte *Humble View* Abhilfe schaffen. Diese sorgt dafür, dass die Oberfläche von dem Code, der die Logik implementiert, völlig trennt. Dies hat den Vorteil, den Teil des View-Codes, der sich schwierig oder nicht testen lässt, von dem Teil, der testbar ist, zu trennen. Im Zuge dessen fiel die Entscheidung, diese Variante ebenfalls zu realisieren (vgl.[BAE+14], S.236f.).

4.4.7. CDI

Contexts and Dependency Injection (CDI) erlaubt es, das Konzept der allgemeinen *Dependency Injection* innerhalb von Java-EE-Umgebungen einzusetzen. Dabei handelt es sich um ein Prinzip, welches die Kontrolle über den Aufbau und die Verwaltung von Abhängigkeiten zwischen Objekten erlaubt. Einen essenziellen Faktor, weshalb die Entscheidung getroffen wurde, diese Technologie zu nutzen, stellt die lose Kopplung von Komponenten dar. Einen weiteren Grund bildeten die effiziente Testbarkeit, Wartbarkeit und Austauschbarkeit. Damit diese Technologie jedoch innerhalb von Vaadin eingesetzt werden kann, muss ein spezielles *Vaadin CDI* verwendet werden. Die genaue Realisierung innerhalb des Projektes kann Kapitel 5.3 entnommen werden (vgl. [BAE+14], S. 256).

4.4.8. JPA

Ein weiterer Teil zur Entscheidung einer zu nutzenden Technologieform betrifft der Zugriff und das Persistieren von anfallenden Daten. Damit sind insbesondere solche Daten gemeint, die zur Laufzeit des Projektes anfallen. Das entsprechende Datenmodel wird im Kapitel 5.4 im Detail beschrieben.
Es existieren eine Reihe von Möglichkeiten zur Umsetzung dieser Funktionalität, welche sich in verschiedenen Punkten von einander unterscheiden. Um eine sinnvolle Auswahl treffen zu können, wurden eine Reihe von Kriterien aufgestellt:

- Skalierbarkeit: Die zugrunde liegende Technologie soll im Stande sein, alle anfallenden Daten in ausreichender Geschwindigkeit zu verwalten. Insbesondere, da nur schwere Aussagen getroffen werden können, hinsichtlich der Menge der in der Zukunft anfallender Daten, muss dieser Punkt gewährleistet werden.

- Erweiterbarkeit: Sowohl im Rahmen der Projektarbeit, als auch im Ausblick nach Projektende hinaus, soll die Möglichkeit bestehen, bspw. neue Tabellen anzulegen, bestehende Tabellen um Attribute zu erweitern oder zu dezimieren.

- Plattformunabhängigkeit: Die Plattform muss darauf ausgelegt sein können, auf mehreren Plattformen lauffähig zu sein, idealerweise mit nur geringem Konfigurationsaufwand.

Neben der Möglichkeit, direkt auf die Daten über entsprechende JDBC-Treiber[22] zuzugreifen, kann der Zugriff über Objektrelationales Mapping (im Folgenden Objektrelationales Mapping), durch den Einsatz von entsprechenden Framework, realisiert werden. In Java existiert die standardisierte Schnittstelle Java Persistence API (Im Folgenden JPA, vgl. [DeM13]), welche das Objektrelationale Mapping umsetzt. Bei Letzterem werden spezielle Java-Klassen definiert, welche einem von JPA vorgegebenen Muster entsprechen müssen. Durch das Zuweisen von Daten an die Attribute einer solchen Klasse werden entsprechende Operationen an der Datenbank automatisch durchgeführt. Der Medienbruch von existierenden Java-Klassen und -Attributen zu einem Relationalen Modell entfällt damit für den Entwickler. Hinsichtlich der Kriterien werden beide Möglichkeiten zum Persistieren der anfallenden Daten verglichen.
Beim Zugriff auf die Datenbank über JDBC-Treiber werden Operationen direkt über SQL-Statements definiert. In einem ersten Schritt werden, vorab und einmalig, sämtliche Tabellen der Datenbank, inklusive aller Beziehungen der Datenbankattributen, eventuellen Bedingungen etc. über SQL-Statements, erstellt. Im Anschluss können Lese- und Schreibzugriffe auf die Tabellen durchgeführt werden. Da der Zugriff auf die Daten über

[22] http://www.oracle.com/technetwork/java/javase/jdbc/index.html

eigenhändig definierte SQL-Statements erfolgt, kann auf die Geschwindigkeit, mit der auf Daten zugegriffen wird, direkt Einfluss genommen werden. Je nach verwendetem Datenbankmanagementsystem, existieren Möglichkeiten zur Geschwindigkeitsoptimierung bei hohem Datenaufkommen. MySQL bietet bspw. durch Verwaltung der Indizes von Datenbanktabellen Möglichkeiten dazu an (vgl. [ZB05], S.65ff.). Eine hohe Skalierbarkeit ist damit durch entsprechende Implementierung und Verwaltung möglich, wird bei der Verwendung von JDBC jedoch nicht garantiert.
Der Punkt der Erweiterbarkeit ist bei dieser Zugriffsmöglichkeit jedoch problematisch: Bei Änderungen am Datenbankmodell, auch bereits bei geringfügigen Änderungen, müssen entsprechende SQL-Statements definiert werden, welche die jeweiligen Änderungen beschreiben. Auch bei Lese- und Schreibzugriffen müssen definierte Statements angepasst werden. Die Erweiterung des Datenbankmodells ist damit zwar prinzipiell möglich, jedoch mit vergleichsweise hohem Aufwand verbunden (vgl. [SM05], S.167ff.).
Es existieren unterschiedliche JDBC-Treiber, mit denen sich verschiedene Datenbankmanagementsysteme verwenden lassen. Durch den Austausch des Treibers kann entsprechend ein anderes Datenbankmanagementsystem verwendet werden (vgl. [Ree00], S.30ff.) . Veränderungen am Code sind dabei nicht zwingend erforderlich. Es existieren jedoch eine Reihe von SQL-Dialekten, sodass entsprechende Anfragen möglicherweise verändert werden müssen. Auch können einzelne Datenbankmanagementsysteme Funktionen bereitstellen, welche sich von denen der Anderen unterscheiden. Minimale Änderungen an den SQL-Statements können damit beim Wechsel nicht ausgeschlossen werden. Damit wird der Punkt der Plattformunabhängigkeit nur bedingt erfüllt.
Dem Gegenüber steht die Möglichkeit, Daten über Java Persistence API zu persistieren. Es werden so genannte Entitäten definiert, Java-Klassen, welche nach durch JPA vorgegebene Muster erstellt werden. Dies betrifft insbesondere Annotationen, mit Hilfe dessen erforderliche Informationen angegeben werden. Mittels der @Entity-Annotation über einer Klasse (Siehe Listing 1) wird bspw. festgelegt, dass der Inhalt als Entität betrachtet wird, sodass enthaltende Felder automatisch als Attribute der angegebenen Datenbank erstellt werden. Weitere Annotationen können in den JPA-Spezifikationen eingesehen werden (siehe [DeM13]). Die definierten Entitäten werden nachfolgend über einen festgelegten JPA-Provider verwaltet. Dieser enthält die Implementierung der JPA-Spezifikationen. Das bedeutet insbesondere, dass dieser die entsprechenden Tabellen in der definierten Datenbank erstellt und bei Veränderungen anpasst sowie Datensätze bei Bedarf hinzugefügt, bearbeitet oder ausliest (vgl. [MW12], S.17ff.).

```
134 @Entity
135 public class Entity implements Serializable {...}
```

Quelltext 1: Definition einer Entität

Auf das Kriterium der Skalierbarkeit kann als Entwickler nur indirekt Einfluss genommen werden, da die Datenverwaltung vollständig vom JPA-Provider übernommen wird. Man kann insofern Einfluss auf dieses Kriterium nehmen, als das verschiedene JPA-Provider, daher unterschiedliche Implementierungen der JPA-Spezifikationen, existieren. Durch Wahl einer geeigneten Implementierung, bei der das Model entsprechend umgesetzt wird, sodass Daten mit hoher Skalierbarkeit persistiert werden können, kann das Kriterium als erfüllt betrachtet werden.
Die Erweiterbarkeit gestaltet sich vergleichsweise einfach. Durch Hinzufügen oder Entfernen von neuen, in Java definierten Entitätenklassen oder entsprechenden Feldern, können Veränderungen problemlos umgesetzt werden. Anpassungen am Datenbankmodell werden anschließend vom JPA-Provider durchgeführt. Die Qualität, mit der Veränderungen am Datenmodell durchgeführt werden, hängt damit ebenfalls direkt vom gewählten JPA-Provider ab.
Das Erstellen der einzelnen Entitäten erfolgt, wie bereits beschrieben, vollständig in Java-Klassen, unabhängig von der verwendeten Datenbank. Auch an dieser Stelle hängt die Qualität, mit der das Kriterium erfüllt wird, direkt von dem verwendeten JPA-Provider ab. Durch Wahl eines Providers, welcher viele Datenbankmanagementsysteme unterstützt, ohne weitere Anpassungen in den Entitätenklassen bzw. Abfragen erforderlich zu machen, kann der Punkt der Plattformunabhängigkeit als erfüllt betrachtet werden.
Es wird deutlich, dass bei Wahl eines geeigneten JPA-Providers, das Persistieren über JPA in allen definierten Anforderungspunkten Vorteile aufweist. Populäre Provider sind u.a. EclipseLink[23] und Hibernate[24]. Beide sind zertifizierte JPA-Implementierungen, die die Spezifikationen vollständig umsetzen, verhalten sich jedoch mitunter unterschiedlich und enthalten über die JPA-Spezifikation hinaus zusätzliche Funktionen (vgl. [MW12], S.17ff.). Insbesondere existiert über die Erweiterung Hibernate Envers[25] die Möglichkeit, Daten automatisch zu historisieren, sodass Veränderungen in der entstehenden Applikation protokolliert und alte Bestände eingesehen werden können. Dies kann durchaus eine sinnvolle Erweiterung darstellen. Aus diesen Gründen wurde zunächst Hibernate zur Umsetzung des Datenmodells genutzt.
Im Laufe der Implementierung wurde jedoch deutlich, dass sich Hibernate weniger für das Projekt eignet, als zunächst erwartet. Insbesondere kam es damit zu Fehlern, wenn es darum ging, Informationen bei multidimensionalen Beziehungen bestehender Entitäten nachzuladen. Für einen fehlerfreien Ablauf wurde es damit erforderlich, sämtliche Daten, welche evtl. benötigt werden, beim ersten Zugriff auf Informationen vorzuladen. Dies wirkte sich stark auf die Geschwindigkeit aus. Derartige Probleme bestehen unter dem

[23] http://www.eclipse.org/eclipselink/
[24] http://hibernate.org/
[25] http://hibernate.org/orm/envers/

JPA-Provider EclipseLink nicht, sodass die Entscheidung gefallen ist, den Provider auszutauschen.
Durch die Wahl von JPA zur Persistierung der anfallenden Daten entfällt die Bindung an eine spezifische Datenbank. Die Anforderungen hinsichtlich Geschwindigkeit und Skalierbarkeit werden von vielen populären Datenbanken hinreichend erfüllt. Aufgrund der Quelloffenheit und der höchsten praktischen Erfahrung der Teammitglieder fiel die Wahl der verwendeten Datenbank auf MySQL[26].

4.4.9. Design

In diesem Kapitel werden Konzeptionierungstechniken sowie Technologien, die hinsichtlich des Designs innerhalb von Vaadin genutzt werden, näher erläutert. Zu Beginn werden Technologien im Bereich der Wireframes und Prototypen näher aufgezeigt. Im Anschluss erfolgt eine nähere Beschreibung von Techniken, die innerhalb von Vaadin im Bereich des Themes sowie im Bereich der Stil-Definitionen genutzt werden.

Wireframes und Prototypen

Um unterschiedliche Masken einer Webanwendung (vor der Implementierung) effektiv zu visualisieren, können sogenannte Wireframes verwendet werden. Ein Wireframe stellt dabei eine optische Darstellung der Anforderungen an eine Software dar. Die Funktionen werden dabei lediglich grafisch angedeutet. Ein Wireframe wird meist aus einer Skizze gefertigt und stellt in der Konzeption den Schritt nach der reinen Skizzierung einer Anwendung dar. Es dient dabei als saubere Darstellung der Skizzen. Gegenüber einem Mockup verfügt das Wireframe hingegen über keinerlei Designinformationen, wie die Ausrichtung nach einem Corporate Identity. Das Wireframe gibt darüber Aufschluss, wie die Anordnung der User-Interface-Elemente (UI-Elemente) angedacht ist. (vgl. [AAB10], S.139ff.)

[26] http://www.mysql.org/

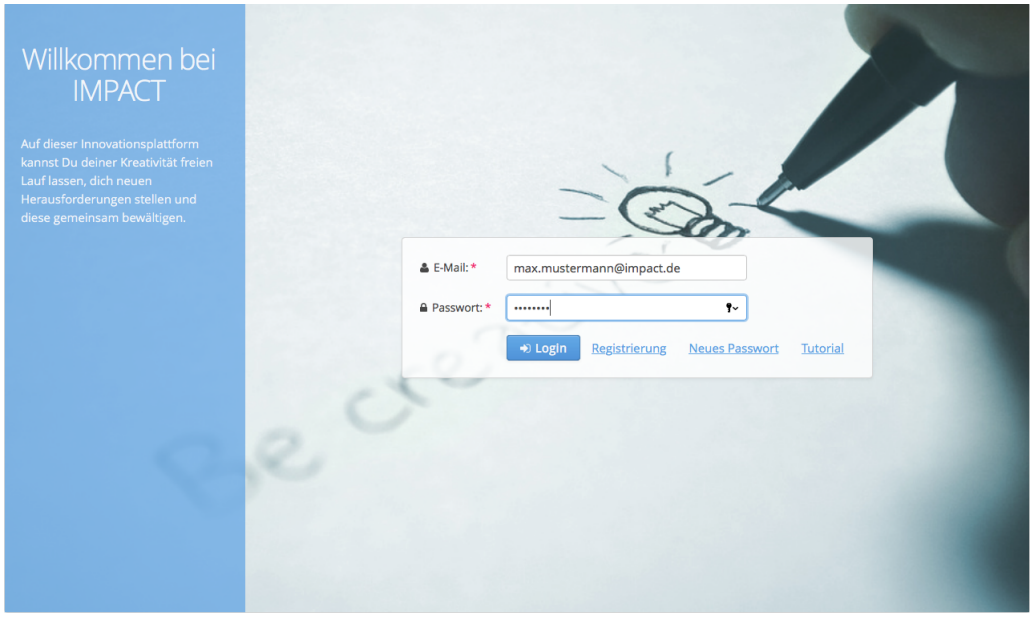


Abbildung 16: Startseite von IMPACT

Wie in Abbildung 29 veranschaulicht, werden innerhalb eines Wireframe u.a. die Ausrichtung und Platzierung von Elementen abgebildet. Darüber hinaus lässt sich erkennen, welche funktionalen Anforderungen an die Applikation gestellt werden, wie bspw. eine Loginfunktionalität. Wireframes können bereits mit vielen kostenlosen Programmen, wie z. B. Balsamiq Mockup[27], gestaltet werden. Diese Programme dienen dabei oft nicht nur zur Umsetzung von Wireframes, sondern auch direkt zur Erstellung von Mockups. Weitere Wireframes, die während der Konzeption hinsichtlich der Webanwendung zu den einzelnen Bereichen von IMPACT entstanden sind, können im Anhang unter Wireframes A.4 entnommen werden.
Um die Oberflächenprogrammierung leichter zu gestalten, können sogenannte Prototypen eingesetzt werden. Mit Hilfe der visuellen Darstellung ist es möglich Styleformate und -Strukturen in der Programmierung leichter abzubilden. Ein Oberflächenprototyp bildet im Gegensatz zum Wireframe eine konkrete Darstellung eines Systems ab. Es ist dabei möglich, logische und interaktive Elemente zu benutzen, um eine klarere Vorstellung des Produktes zu vermitteln (vgl. [AAB10], S.139f.).
Für die Auswahl einer geeigneten Software muss zuvor geklärt werden, in welcher Form die visuelle Darstellung genutzt und inwieweit diese detailliert umgesetzt werden soll. Innerhalb der Projektgruppe *IMPACT* wurden im Zuge dessen erste Ideenskizzen sowie Überlegungen in der Gruppe erarbeitet. Dies fand zunächst meist an Whiteboards oder

[27] https://balsamiq.com/

klassisch auf dem Papier statt. Gerade in einer Phase, in der sehr viele Ideen entstehen und kollaborativ weiterentwickelt werden müssen, bietet sich diese Form nach wie vor an. So können bei Meetings Ideen schnell ohne jeglichen Computer festgehalten, verdeutlicht und anderen Team-Mitgliedern präsentiert werden.
Das Tool Balsamiq Mockup für die Visualisierung von Wireframes kommt dabei u. a. zum Einsatz. Mit diesem Tool lassen sich per Drag and Drop einfache Wireframes erstellen, die eine erste digitale Visualisierung eines Produkts ergeben können. Dabei bietet Balsamiq Mockup bereits einige Bausteine für die Erstellung von Wireframes im Bereich von Webseiten an. Anhand dieser Vielzahl an Möglichkeiten, die Balsamiq Mockup bietet, fiel die Wahl der Projektgruppe auf dieses Tool. Darüber hinaus wurde das Tool Axure eingesetzt, welches eingesetzt wird, um interaktive Prototypen zu erstellen. Diese können z. B. in Form von HTML Seiten generiert werden.

Design innerhalb von Vaadin

Ein weiterer Aspekt hinsichtlich der Technologieentscheidung lässt sich im Bereich des Designs in Vaadin herauskristallisieren. Das Vaadin-Framework verfolgt dabei eine strikte Trennung zwischen dem Aufbau der Benutzeroberfläche, die durch die Komponenten definiert wird und der eigentlichen Darstellung des Designs im Webbrowser. Hierbei rückt das sogenannte Theming in den Vordergrund, welches von Vaadin eingesetzt wird, um den Komponenten ein einheitliches Design zu übergeben.
Vaadin selbst bringt von sich aus mehrere Themes mit, die entweder direkt verwendet oder als Ausgangsbasis für ein eigenes angepasstes Theme genutzt werden können. Bezüglich des Themes fiel die Wahl auf das seit Version 7.3 verfügbare *Valo*[28]. Dabei handelt es sich um das neuste Theme von Vaadin, welches die effizientesten und effektivsten Möglichkeiten hinsichtlich des Designs der Komponenten etc. bietet. Im Zuge dessen hat sich die Projektgruppe dazu entschieden, diese Technologie zu nutzen.
Einen weiteren Aspekt bildet das eigenständige Design der einzelnen Komponenten. Vaadin bietet diesbezüglich die Möglichkeit, diese Komponenten in Form von Syntactically Cascading Style Sheets (SCSS) und Cascading Style Sheets (CSS) über zwei Wege, selbstständig zu gestalten. Dieses Styling ist ebenfalls in Themes ausgelagert, die alle Styles für die komplette Komponenten- Bibliothek bereitstellen. Die Vaadin Literatur empfiehlt diesbezüglich selbst bei einfachen Themes, statt normalem CSS die Spracherweiterung SCSS zu nutzen. Bei komplexen Webanwendungen kann nämlich die Übersicht im CSS-Code schnell verloren gehen. Moderne Webanwendungen und Websites nutzen z. B. ein Farbkonzept, welches durchdacht ist und basieren auf Gestaltungsrastern, die auf sämtliche Displaygrößen interaktiv reagieren können.

[28] `https://vaadin.com/valo`

Der Aufbau ist auf den ersten Blick etwas komplexer als bei einem CSS-Theme, dies ermöglicht jedoch u.a. die spätere Verwendung eines eigenen Themes als Vorlage für weitere Themes. Anhand dieser Argumente fiel die Wahl auf SCSS gegenüber CSS, da dahingehend lediglich Vorteile generiert werden (vgl. [BAE+14], S.169ff.).

4.5. UI-Design

Dieses Kapitel beinhaltet sämtliche Erklärungen rund um das Design der Webanwendung von IMPACT und verschafft einen Überblick über die Design-Elemente, welche im späteren Verlauf innerhalb der Realisierung genutzt werden sollen. Des Weiteren werden Annäherungen aufgezeigt, inwiefern das Design der Anwendung konzipiert wurde. Für das erste Vorgehen war notwendig, Ideen hinsichtlich der Designvorstellung zu sammeln und das weitere Vorgehen zu planen. Dabei stellte insbesondere die intuitive Bedienung der einzelnen Abläufe eine essenzielle Anforderung dar. Es sollte demnach ein Weg gefunden werden, einen Nutzer möglichst frühzeitig zu motivieren, die Anwendung langfristig zu bedienen. Ebenso soll das Design so gestaltet sein, dass Fehler während der Eingabe möglichst vermieden und abgefangen werden können. Letzteres kann Kapitel 6.2 entnommen werden. Grundsätzlich wurde der Projektgruppe für die Gestaltung der Anwendung keine grundlegenden Vorgaben gegeben, sodass die grafische Darstellung komplett frei entwickelt werden konnte. Der nächste Schritt umfasst nun die Entwicklung eines geeigneten Layouts der Webanwendung. Dieses soll in erster Linie den Inhalt der Seite unterstützen, indem es den Blick des Betrachters leitet und ihm ein klares und übersichtliches Raster bietet, auf dessen Basis die gesuchten Informationen schnell zu finden sind.
Den ersten Schritt beinhaltete das Vorgehen für den Entwurf des Designs, welches mit der Erstellung mehrerer Wireframes der einzelnen Bereiche der Webanwendung (Startseite, Lobby etc.) begann. Im späteren Verlauf des Projektes wurde zusätzlich ein Prototyp erstellt, um das *Look and Feel* der Anwendung zu testen. Während des ganzen Projektes wurde der Prototyp und das letztendliche Design während des Entwicklungsprozesses stetig weiterentwickelt und verfeinert, damit die gesetzten Anforderungen fachgerecht erreicht wurden.

4.5.1. Wireframes

Das Wireframe der Webanwendung wurde in Kleingruppen zunächst auf Papier erstellt. Im weiteren Verlauf wurde, wie in Kapitel 4.4.9 erläutert, die Software Balsamiq Mockup benutzt, um diese dementsprechend digital zu erstellen und effizienter arbeiten zu können. Dies führte dazu, dass unterschiedliche Vorstellungen hinsichtlich des Designs der Wireframes durch die einzelnen Kleingruppen eingeflossen sind. Durch die unterschiedlichen Vorstellungen entstand viel Diskussionsbedarf hinsichtlich des Designs der Anwendung, wodurch ein positives Resultat erzielt werden konnte.

4.5.2. Seitenstruktur

Dieses Unterkapitel soll einen Überblick über die Seitenstruktur von IMPACT verschaffen.

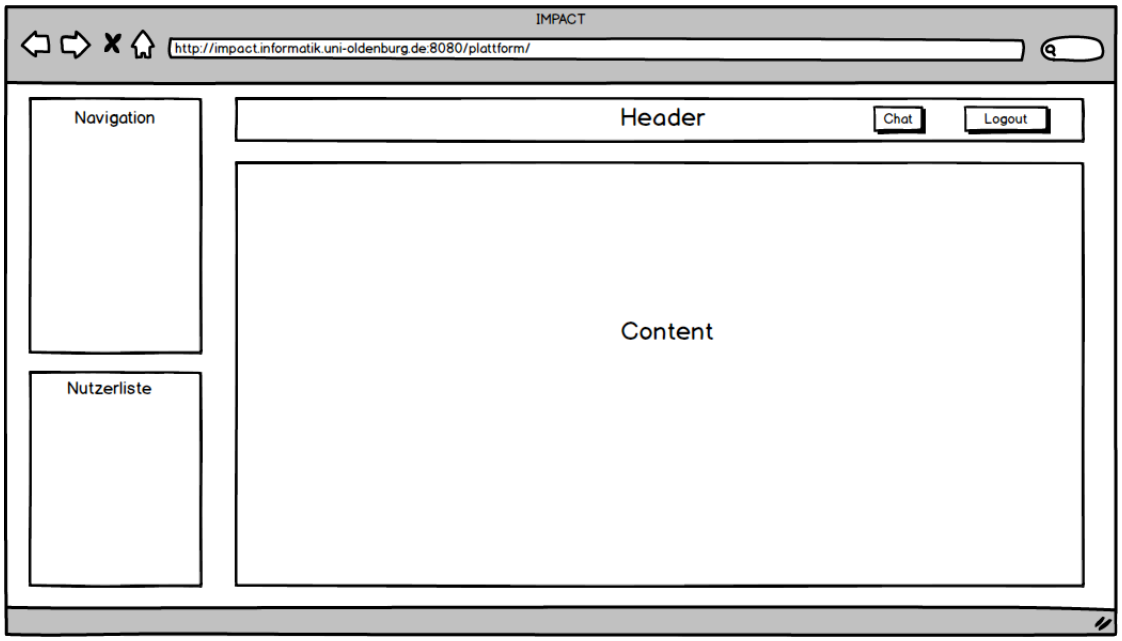


Abbildung 17: Wireframe: Bereich nach dem Einloggen

Die Abbildung 17 zeigt den vereinfachten Grundaufbau des Layouts der Webanwendung von IMPACT in Form eines Wireframes auf. Zu erkennen ist die schlichte und überschaubare Struktur sowie die strikte Trennung der primären Inhalte von den Seitenattributen. Die Navigation wird dabei auf der linken oberen Seite der Anwendung angeordnet, um ein optimales Platzverhältnis für den Kopf- und Inhaltsbereich zu gewährleisten, die innerhalb der Webanwendung als Header und Content dargestellt werden. Des Weiteren lässt sich anhand der Abbildung erkennen, dass der Platz unterhalb der Navigation für die Nutzerliste genutzt wird. Die restlichen Bereiche werden dahingehend für den Header

sowie für den Inhalt verwendet. Anhand dessen lässt sich erkennen, dass der Platz und die Anordnung der Elemente optimal ausgenutzt wird. Die nachfolgende Abbildung verdeutlicht, inwiefern dieser Grundaufbau innerhalb der Anwendung realisiert wurde. Der gesamte Bereich nach dem Einloggen wird innerhalb der Plattform auch als *Lobby* bezeichnet.

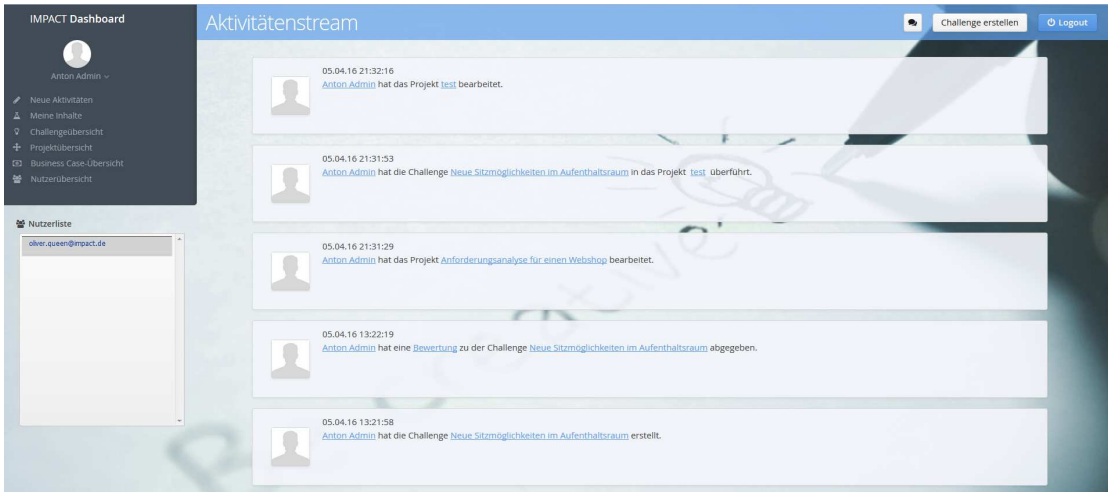


Abbildung 18: Bereich nach dem Einloggen

Die Navigation auf der Seite dient dazu, um zwischen den verschiedenen Hauptinhalten der Webanwendung zu navigieren und so letztendlich zu weiteren Informationen zu gelangen. Die Navigation der Webanwendung ist dabei in simpler Form gehalten. Die Hauptinhalte werden in vertikaler Form gegliedert. Ein weiteres Element, welches hinsichtlich der Seitenstruktur im Bereich der Navigation genutzt wird, lässt sich bei der Reduzierung der Pfade herauskristallisieren. Die Webanwendung zeichnet sich durch eine flache Seitenstruktur aus und stellt kein separates Menü zur Navigation zur Verfügung. Dies wurde verwendet, da die Strukturierung bewusst so gewählt wurde, dass die gesuchten Informationen effizient zu finden sind.

4.5.3. Farbwahl

Im Bereich der Farbwahl hat die Projektgruppe folgendes Farbschema als finale Version ausgewählt:

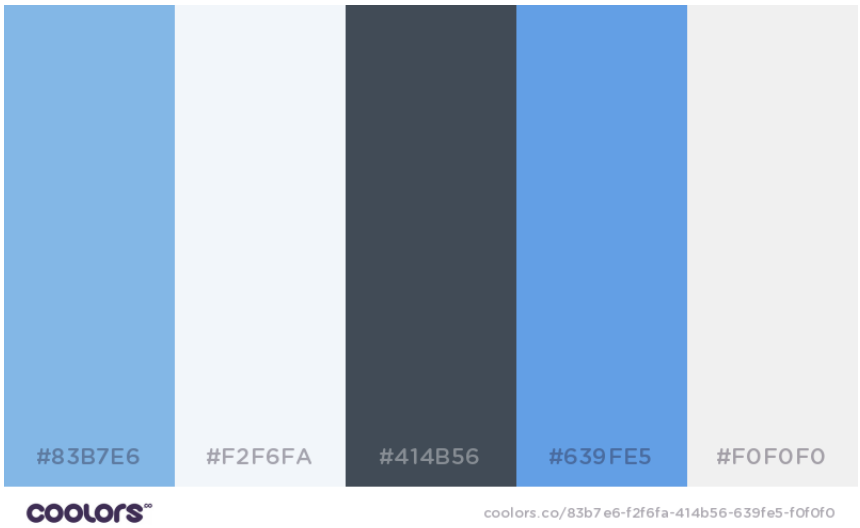


Abbildung 19: Farbpalette von IMPACT

Anhand der Grafik 19 ist zu erkennen, dass sich die Farbwahl in eine Farbpalette mit sechs Farben gliedert. Die Primärinhalte wurden klassisch in schwarzer Schrift auf weißem und transparentem Hintergrund belassen, um eine gute Lesbarkeit und Übersichtlichkeit zu gewährleisten. Im Zuge dessen wurde bewusst ein subtiler Farbton eingenommen, damit ein zusätzlicher Fokus auf das Hintergrundbild gelegt wird (vgl. Abbildung 21). Der Hintergrund der Webanwendung ist in Form einer Abbildung dargestellt. Das Hintergrundbild wurde von *CreativCommons.org* entnommen und bildet eine weitere Verbindung zu den anderen genutzten Farben. Der Header sticht in klassischem hellblau hervor. Dieser soll die Anwendung insbesondere bei hohen Auflösungen vom Hintergrund abheben.

Abbildung 20: Bereich nach dem Einloggen

Eine zusätzliche Hintergrundfarbe lässt sich im Bereich der Navigation auf der linken Seite der Webanwendung herauskristallisieren. Diese ist in einem dunklen Grauton gehalten. Dieser gedeckte Farbton tritt dezent zum Header auf, setzt sich aber gegenüber dem Hintergrundbereich deutlich ab und trennt das Menü von den Inhaltsbereichen, ohne aggressiv hervorzustechen. Der helle Weißton wurde als Hintergrund genutzt, um die Hauptinhalte

der Lobby und die Informationen, die sich hinter den Navigationspunkten befinden, darzustellen. Darauf aufbauend wurde die Schrift in einem klassischen Schwarzton angesetzt, da dies einen positiven Kontrast gegenüber dem weißen Hintergrund einnimmt und somit eine positive Lesbarkeit gewährleistet.

4.5.4. Typographie

Die Typografie wurde bei IMPACT lediglich als gestalterisches Mittel eingesetzt, indem Hervorhebungen und Auszeichnungen von Texten und Überschriften sehr dezent gehalten wurden. Vaadin bietet diesbezüglich vorgefertigte Mittel an, die den Einsatz einer optimalen Typografie zusätzlich optimieren.

Schriftgröße, Schriftart und Farbwahl

Bei der Schriftgröße und dem Zeilenabstand wurde stets darauf geachtet, dass diese stetig an die Inhalte sowie an die Hintergrundfarbe angepasst sind, um ein optimales Leseverhalten zu gewährleisten. Diesbezüglich wurde das Design der Webanwendung so optimiert, dass lediglich die Farben weiß und schwarz in Verbindung mit der serifenlosen Schrift *Open Sans* zum Einsatz kommen. Dies sorgt dafür, dass der Nutzer sich auf das wesentliche konzentriert und somit die Webanwendung optimal bedienen kann. Hauptüberschriften, wie sie z. B. auf der Startseite bei der Begrüßung oder im Header der Lobby eingesetzt werden, sind in einem dezenten Weiß dargestellt. Die Überschriften werden bewusst größer als die Texte im Content-Bereich dargestellt, damit Elemente, die größer erscheinen, beim Betrachter eine höhere Aufmerksamkeit erhalten und auf Grund der größeren Flächenbeanspruchung bevorzugt wahrgenommen werden. Da die Schriften auf höher aufgelösten Bildschirmen kleiner ausfallen, werden die Texte im Content-Bereich auf bis zu 16 Pixel vergrößert, um ein optimales Leseverständnis zu ermöglichen.

Icon-Fonts

Neben herkömmlichen Schriftarten wurden zusätzlich Icon-Fonts eingesetzt. Diese sorgen dafür, dass statt Buchstaben, Icons eingesetzt werden, die sowohl farblich, als auch in der Größe individuell angepasst werden können. Im Rahmen der Webanwendung wurden diese bei den Formularfeldern und den Buttons eingesetzt. Dieser Ansatz wurde verfolgt, um die Usability zu erhöhen und dem Nutzer die Bedienung der Webanwendung zu erleichtern.

4.5.5. Lobby als zentrales Content-Element

Zuvor wurde erläutert, wie sich die Seitenstruktur innerhalb der Webanwendung gliedert. Es wurde dabei kurzerhand aufgezeigt, dass die Lobby das zentrale Element abbildet, welches nach dem erfolgreichen Login aufgerufen wird, um Platz für die primären Inhalte der Webanwendung zu schaffen. Diese Inhalte besitzen ein hohes Volumen an überwiegend textuellen Informationen. Zur Darstellung dieser Inhalte wird der Content-Bereich der Seite (vgl. Abbildung 21) genutzt.

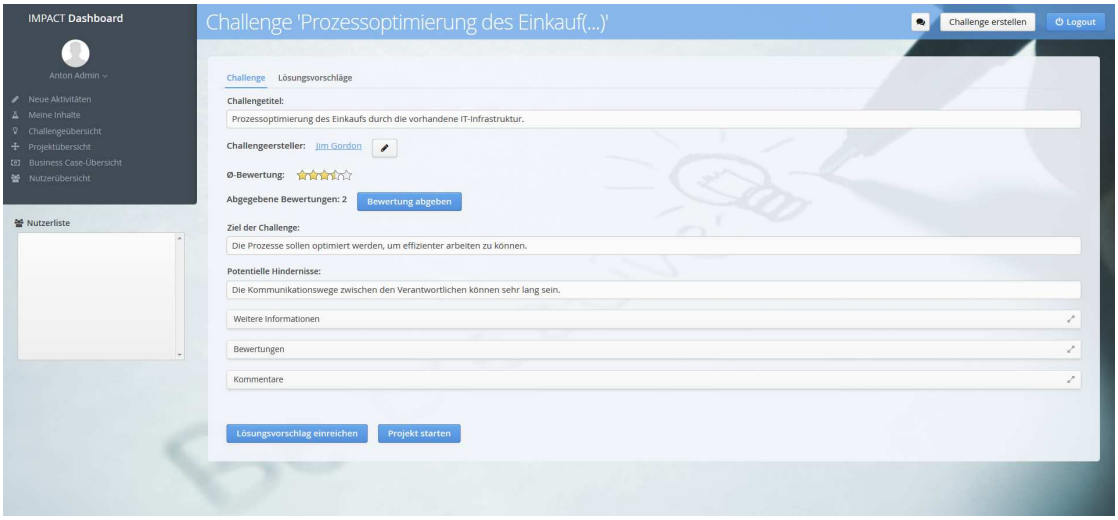


Abbildung 21: Ansicht: einzelne Challenge

Anhand der Abbildung 21 lässt sich aufzeigen, dass der Content dazu genutzt wird, um sowohl die textuellen als auch grafischen Elemente der Webanwendung darzustellen, wie bereits zuvor erwähnt wurde. In dem Content-Bereich werden die jeweiligen Ansichten geladen, die aufgerufen werden. Lediglich in diesem Bereich findet eine ständige Veränderung statt. Somit können alle Ansichten über die Menüleiste oder über die Kopfzeile aufgerufen werden. Der Aktivitätenstream 18 stellt dabei den Start-Content dar, in dem bei jedem Aufruf neue Aktivitäten innerhalb des Systems bekannt gemacht werden. Die im Content-Bereich dargebotenen Informationen sind umfangreich und werden daher klar strukturiert und gradlinig dargestellt, um die Usability zu ermöglichen und den Platz für einzelne Komponenten optimal auszunutzen. Somit bietet dieser den idealen Spielraum für die Verwendung verschiedener Designelemente, ohne die klare Struktur des Content-Bereichs der Webanwendung negativ zu beeinflussen.

4.5.6. Optimierung der Desktop-Ansicht

Hinsichtlich der Optimierung im Bereich der Desktop-Ansicht existieren eine Vielzahl an Strategien. Im Rahmen der Konzeption wurde neben den zuvor aufgezählten Designkriterien der Punkt betrachtet, die Webanwendung in einem bestimmten Maße an mehrere Auflösungen anzupassen. Es tritt immer wieder der Fall ein, dass die Nutzer über unterschiedliche Endgeräte mit unterschiedlichen Displaygrößen verfügen. Hinsichtlich dieser Problemstellung kann das sogenannte *Responsive Webdesign* eingesetzt werden, welches auch unter dem Begriff *reaktionsfähiges Design* aufgefasst wird. Mit Hilfe dieser kann das eingangs erwähnte Problem im Hinblick auf die unterschiedlichen Displaygrößen gelöst werden (vgl. [Zil12], S.10f.).
In dem Kapitel der Technologieauswahl im Bereich des Designs (vgl. Abschnitt 4.4.9) wurde aufgezeigt, dass im Bereich der Design-Entwicklung *Valo* eingesetzt wird. Im Hinblick auf das *Responsive Webdesign* bietet dieses Theme die Möglichkeit, die einzelnen Komponenten von Vaadin (Layouts, Textfelder, Buttons etc,) sowohl im Java-, als auch im SCSS-Code, einzubinden. Entsprechende Vorgehensweisen hinsichtlich der technischen Umsetzung werden in Kapitel (vgl. Abschnitt 4.4.9) genauer erläutert. Die nachfolgenden Abbildungen 22 und 23 verdeutlichen einen Überblick über die unterschiedlichen Ansichten der Lobby. In diesem lässt sich erkennen, dass in dem Bereich *Meine Inhalte* bei einer kleineren Displaygröße die Navigation sowie die Nutzerliste nach oben verschoben werden, um optimalen Platz zu schaffen. Des Weiteren werden die zentralen Elemente im Content-Bereich eingerückt und der Header nimmt eine schmalere Form ein.

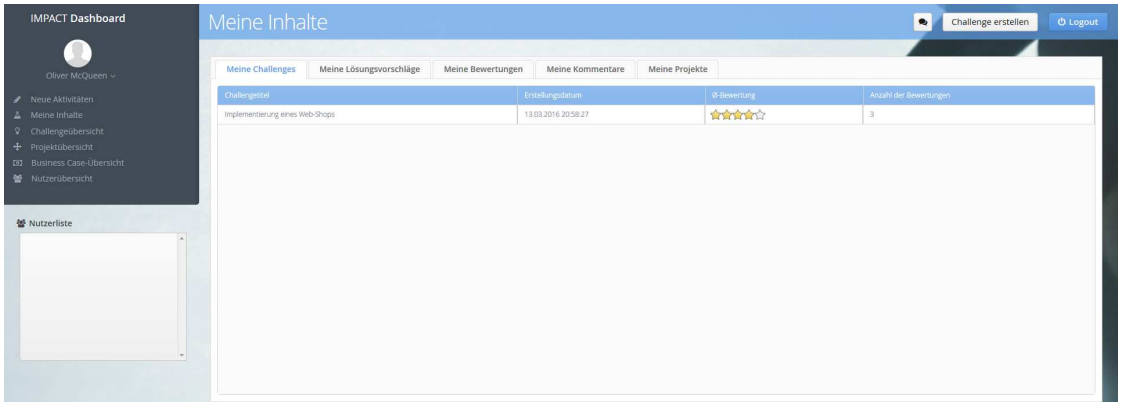


Abbildung 22: Ansicht: Meine Inhalte

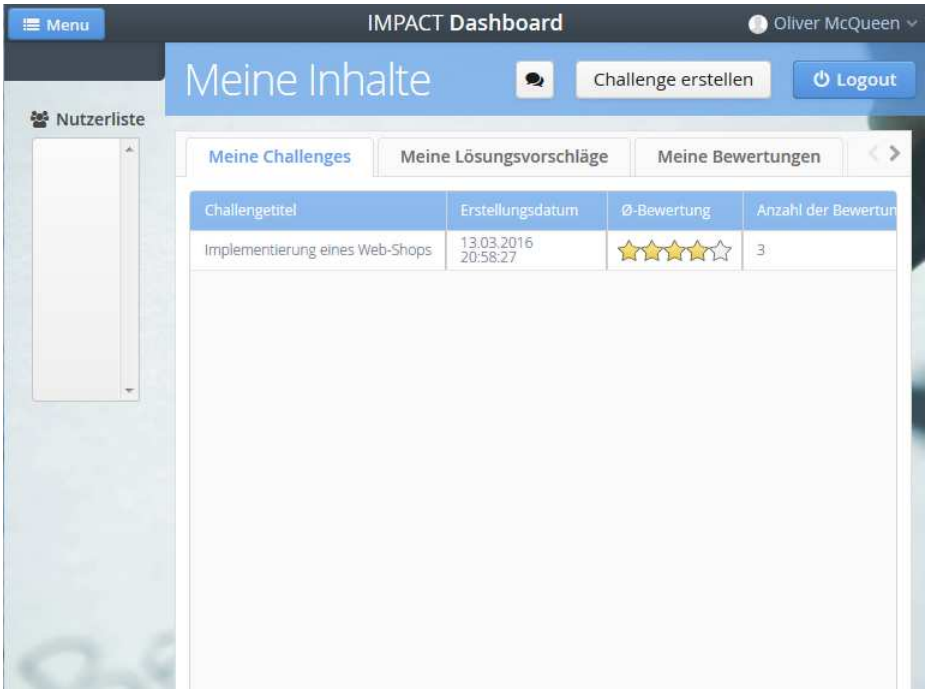


Abbildung 23: Responsive Ansicht: Meine Inhalte

4.6. Bewertungssystem

Bewertungen beschreiben ein Verfahren zur Bestimmung des Werts von Gütern oder Handlungsalternativen[Pfi15]. In dem Kontext der Projektgruppe soll der Benutzer die Möglichkeit erhalten, Challenges und Lösungsvorschläge zu bewerten. Mit Hilfe dieser Bewertung soll die Qualität und der Nutzen ermittelt werden.
Es existieren jedoch viele unterschiedliche Formen einer Bewertung. Die bekanntesten Formen sind hierbei die Sterne-Bewertung (Amazon), die Temperatur-Bewertung (mydealz) und die *Gefällt mir*-Bewertung (Facebook). Um einen Überblick über die verschiedenen Methoden und deren Vor- und Nachteile zu bekommen, wurden die wichtigsten in Form einer Präsentation aufgearbeitet. Diese werden im folgenden Abschnitt vorgestellt.

Bewertungsmethode Facebook

Facebook bietet dem Benutzer die Möglichkeit, Beiträge und Kommentare mit *Gefällt mir* zu markieren. Der Benutzer kann somit nur positives Feedback abgeben und keine Differenzierung vornehmen. Weiterhin ist es möglich, Kommentare zu Beiträgen und anderen Kommentaren zu verfassen. Dort könnte eine sachliche Argumentation einer Bewertung stattfinden. Die Bewertung mittels *Gefällt mir* steht jedoch in keinem direkten Zusammenhang mit einem Kommentar. Das bedeutet der Benutzer kann diese Möglichkeiten unabhängig voneinander verwenden.

Bewertungsmethode Axel Springer Verlag/Bild

Auf der Webseite der Bild-Zeitung hat der Benutzer zum einen die Möglichkeit seine Gefühlslage zu einem Artikel in fünf emotionalen Kategorien einzugliedern (vgl. Abbildung 24). Zum anderen ist es möglich, Kommentare zu Artikeln abzugeben, Kommentare zu verfassen und diese mit dem Facebook-*Gefällt mir* zu bewerten.

Abbildung 24: Bewertung auf der Bild Webseite

Bewertungsmethode YouTube

YouTube bietet seinen Benutzern die Möglichkeit, Videos sowohl positiv (Daumen hoch) als auch negativ (Daumen herunter) zu bewerten. Wie bei den vorherigen Methoden lassen sich auch hier Kommentare zu Videos verfassen. Diese können wiederum positiv und negativ bewertet werden.

Bewertungsmethode Amazon

Auf dem Online-Marktplatz Amazon können die Benutzer Artikel mit Sternen bewerten. Dabei stellt ein Stern das Minimum dar. In 0,5er-Schritten kann die Bewertung erhöht werden, bis zu einem Maximum von fünf Sternen (vgl. Abbildung 25).

Abbildung 25: Bewertung auf der Amazon Webseite

Auch bei Amazon ist es möglich eine Rezension in Form eines Kommentars abzugeben. Diese Rezension kann von anderen Benutzern als hilfreich oder nicht hilfreich eingestuft werden. Zusätzlich lassen sich Kommentare zu Rezensionen erfassen.

Bewertungsmethode mydealz

Die Plattform mydealz lässt die Angebote mit *heiß* oder *kalt* bewerten und ermittelt dadurch die Temperatur (in Grad). Besonders heiße Angebote werden zusätzlich durch eine Flamme hervorgehoben. Weiterhin haben die Benutzer die Möglichkeit, Kommentare zu den Artikeln und zu Kommentaren zu verfassen.

Zusammenfassung der Bewertungsmethoden

Die Methoden besitzen alle die Gemeinsamkeit, dass Kommentare verfasst werden können. In diesen Kommentaren hat der Benutzer die Möglichkeit, seine Bewertung oder aber den Inhalt des Beitrages genauer zu erläutern. Facebook und Bild-Online erlauben ihren Benutzern lediglich die positive Rückmeldung. Auf YouTube und mydealz stehen zwei Möglichkeiten (positiv und negativ) zur Verfügung. Einzig Amazon hat die Funktion einer differenzierten Bewertung. In 9 Abstufungen (ein Stern bis fünf Sterne; 0,5-Intervall) kann der Kunde das Produkt bewerten. Weiterhin bietet keine Methode eine Abhängigkeit zum zeitlichen Kontext, um Trends zu erkennen. Somit wird jede Bewertung, unabhängig vom Alter der Bewertung, gleich gewichtet.

Anforderungen an ein Bewertungssystem

Die Vor- und Nachteile der jeweiligen Systeme wurden zunächst innerhalb der Projektgruppe diskutiert. In dem Kontext, dass die Innovationsmanagement-Plattform im betrieblichen Rahmen verwendet wird, wurden folgende Anforderungen an das Bewertungssystem festgelegt.

- Das Bewertungssystem muss transparent und nachvollziehbar sein.
- Das Bewertungssystem muss optisch ansprechend sein.
- Das Bewertungssystem muss eindeutig sein.
- Das Bewertungssystem muss qualitative Bewertung ermöglichen.
- Das Bewertungssystem muss eine differenzierte Bewertung ermöglichen.
- Das Bewertungssystem sollte eine schnelle und intuitive Eingabe ermöglichen.
- Das Bewertungssystem sollte eine Bewertung im zeitlichen Kontext betrachten.

Entwicklung eines Bewertungssystems

Da keine der untersuchten Methoden den Anforderungen gerecht wurde, hat sich die Projektgruppe entschlossen, ein eigenes Bewertungssystem zu entwickeln. Der zentrale Bestandteil des Systems soll eine Formel sein, die die Bewertung anhand der zeitlichen Attribute gewichtet. Neue Bewertungen bekommen eine höhere Gewichtung, sodass aktuelle Trends leichter identifizierbar sind. Zusätzlich wird eine Abhängigkeit zu der Anzahl der registrierten Benutzer hergestellt. Je höher der prozentuale Anteil, desto höher fällt die Bewertung ins Gewicht. Das Ergebnis ist der *Impact-Score*, der in Java wie folgt berechnet wird.

*calculatedRating = Math.round(logOfBase(1+1/rateCount, givenRating) * 12/regUser * 100)/100.0*

Zurück (kaufmännisches Runden). Die Variable *rateCount* beinhaltet die Anzahl der abgegeben Bewertungen. Die Bewertung (eins bis fünf Punkte; 0,5er Intervalle) wird in der Variable *givenRating* gespeichert. Unter *regUser* wird die Anzahl der registrierten Benutzer festgehalten. Diese Formel, zunächst ohne Abhängigkeit der zeitlichen Komponente, wurde dem Praxispartner im Rahmen eines wöchentlichen Meetings vorgestellt.

Feedback Impact-Score

Der Praxispartner ist von der Idee eines eigenen Bewertungssystems grundsätzlich positiv angetan. Zu Bedenken und nicht zu unterschätzen sei aber der Faktor, dass der Impact-Score zur Verwirrung der Mitarbeiter führen könnte (vgl. Protokoll A.7.25). Dieses ist durch die gewichtete Bewertung begründet.

Beispiel: Drei von 100 Mitarbeitern bewerten eine Challenge mit 4 Sternen. Auf Grund der geringen prozentualen Anzahl der Bewertungen würde der Impact-Score kleiner als 4 Sterne sein. Dieses könnte die Mitarbeiter verwirren und fördert nicht das Vertrauen und die Transparenz einer solchen Bewertung. Dieses Feedback führte dazu, dass die zeitliche und prozentuale Komponente nicht weiter verfolgt wird. Die Projektgruppe hat sich entschieden, die Bewertungsmethode von *Amazon* zu übernehmen. Dort ist im Gegensatz zu den anderen Methoden eine differenzierte Bewertung möglich.

5. Realisierung der Systemarchitektur

In diesem Kapitel wird die realisierte Systemarchitektur vorgestellt. Besonders wird das Vaadin-Framework erörtert.

5.1. Architektur-Überblick

Das *Vaadin*-Framework besteht aus einer server- und clientseitigen API, *UI*-Komponenten/*Widgets*, *Themes*, die das Aussehen der Anwendung bestimmen und einem Datenmodel, welches direkt an die serverseitigen Komponenten gebunden werden kann.
Die clientseitige Entwicklung wird durch den *Vaadin-Compiler* unterstützt, welcher *Java*-Code in *JavaScript* übersetzt.
Die Abbildung 26 zeigt die Architektur einer *Vaadin*-Anwendung und deren Kommunikation zwischen Client (im Browser) und Server.

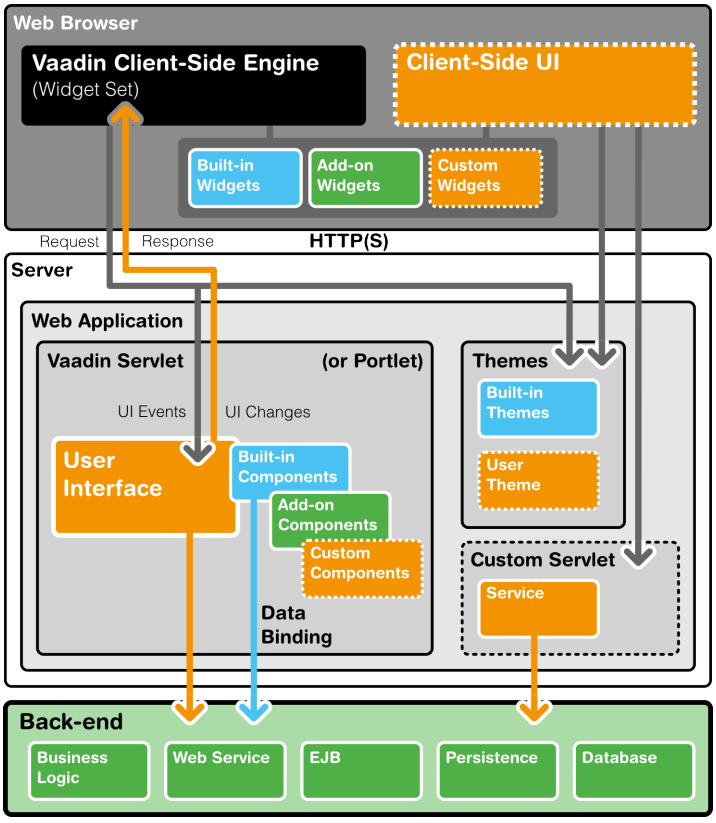


Abbildung 26: Abbildung der Architektur von Vaadin

Eine serverseitige *Vaadin*-Anwendung läuft als *Servlet* in einem Java-Web-Server, z.B. Tomcat. Das *VaadinServlet* empfängt Client-Anfragen (*requests*) und interpretiert diese als *Events* für die *Session* des Benutzers. *Events* sind Komponenten der Benutzeroberfläche

angebunden und werden an, in der Anwendungen definierte *Listener*, weitergeleitet. Wenn die *UI*-Logik Änderungen an den serverseitigen Oberflächen-Komponenten durchführt, sendet das *Servlet* diese als Antwort (*response*) an den Browser. Die *Client-Side Engine* im Browser nimmt die Antworten (*responses*) entgegen und aktualisiert dementsprechend die Webseite im *Browser*. Im Folgenden werden die wichtigsten Komponenten der serverseitigen Architektur beschrieben.

User Interface

Die Benutzeroberfläche der Anwendung stellt die Benutzerschnittelle bereit und bietet sowohl Zugriff auf die Daten, als auch die Logik der Anwendung. Technisch umgesetzt wird diese durch die Klasse `com.vaadin.ui.UI`, dessen Hauptaufgabe ist es, die anfängliche Benutzeroberfläche mittels *UI*-Komponenten zu erstellen und *Event-Listener* einzurichten, welche auf Benutzereingaben reagieren und diese verarbeiten.

User Interface Components/Widgets

Die Benutzeroberfläche besteht aus Server-*Komponenten*, und den dazugehörigen, clientseitigen *Widgets*, welche im Browser angezeigt werden und mit dem Benutzer interagieren. Die Interaktionen werden über serverseitige Komponenten an die Anwendungslogik weitergeleitet.

Client-Side Engine

Die *Client-Side Engine* verwaltet die Benutzeroberfläche und die einzelnen *Widgets*, außerdem erfolgt darüber die Kommunikation zum Server über asynchrone HTTP Anfragen.

Vaadin Servlet

Das *Vaadin Servlet* empfängt die Anfragen von verschiedenen Clients und legt fest, zur welcher Benutzersitzung diese jeweils gehören, um eine entsprechende Weiterleitung zu ermöglichen.

Themes

Vaadin trennt das Aussehen und die Struktur der Komponenten der Benutzerschnittstelle. Während die *UI*-Logik in *Java*-Code implementiert wird, wird das Aussehen durch *Themes* in *CSS* oder *Sass* definiert. Vaadin bietet eine Reihe von Standard-*Themes*.

Events

Durch Interaktionen mit der Oberfläche werden *Events* erzeugt. Diese werden zuerst auf der Clientseite von den *Widgets* verarbeitet, dann über den HTTP-Server, dem *Vaadin-Servlet*, an definierte *Listener* übergeben.

Back-end

Während *Vaadin* für den Aufbau der Benutzerschnittstelle genutzt wird, sollten Geschäftslogik und Daten von der *UI* getrennt werden. Typischerweise werden Datenspeicher per *DBMS* oder Persistenzlösungen wie *JPA* angebunden.

5.2. Datenbankmanagement

Die Umsetzung des Datenbankmanagements erfolgt, wie bereits in Kapitel 4.4.8 beschrieben, mittels der Java Persistence API. Bei dieser Form des Persistierens liegt der Fokus nicht auf dem Erstellen und Verwalten von Datenbanktabellen und -Einträgen über SQL-Befehle, sondern darauf, bestehende Java Entitätenklassen zentral zu verwalten, um damit das Speichern und Auslesen von Daten in hohem Maße effizient, als auch effektiv zu gestalten. Zunächst gliedert sich das Datenbankmanagement in mehrere Teilbereiche. Für jeden Teilbereich existiert jeweils ein Interface, siehe Tabelle 9.

Interface	Beschreibung
LoginService	Kapselt sämtliche Methoden zum Suchen der Nutzer anhand der Mailadresse
UserService	Definiert Methoden zum Anzeigen von Nutzern in der Profilübersicht sowie zum Ändern von Eigenschaften, hinzufügen von Berechtigungsrollen und dem Anzeigen anderer Nutzer, welche ebenfalls angemeldet sind.
RegistrationService	Enthält Methoden zum Registrieren neuer Nutzer
StageService	Definiert Methoden zum Anzeigen und Bearbeiten von Einträgen (z.B. Challenges), Kommentaren und Bewertungen
RatingService	Methoden zum Erstellen von Bewertungen
PermissionService	Enthält Methoden zum Laden von Rechten aus der Datenbank, sowie zum Zuweisen von Nutzerrollen an einzelne Nutzer
TopTenService	Stellt Methoden bereit, um die Liste der zehn am besten bewerteten Einträge aufzulisten.

Tabelle 9: Interfaces aller Datenbankoperationen

Sämtliche Interfaces werden, in der aktuellen Form des Projektes, über die Klasse `DatabaseService` implementiert. Über die in Kapitel 4.4.7 beschriebene Dependency Injection wird beim Datenbankzugriff, über die in Tabelle 9 beschriebenen Service-Interfaces, auf die Implementierungen dieser Klasse zugegriffen. Durch entsprechende CDI-Konfigurationen wäre ein hocheffizienter Austausch der Implementierung der einzelnen Interfaces möglich.

Der Zugriff auf die Daten erfolgt über die Klasse `EntityManager`, welche Teil des JPA-Packages ist. Nach dem einmaligen Instantiieren können die Daten in der spezifizierten Datenbank manipuliert werden. Zum Erstellen neuer Einträge in der Datenbank werden Objekte einer Entitätenklasse an den `EntityManager` übergeben. Daraufhin werden sämtliche Felder des Objektes ausgelesen und daraus ein neuer Eintrag in der entsprechenden Datenbanktabelle erstellt (vgl. Quellcode 2).

```
1 User user = new User();
2 entityManager.getTransaction().begin();
3 entityManager.persist(user);
4 entityManager.getTransaction().commit();
```

Quelltext 2: Beispielhaftes Erstellen eines neuen Nutzers in der Datenbank

Das Auslesen von bereits persistierten Objekten erfolgt über die definierten Primärschlüssel. Zum Bearbeiten muss im `EntityManager` eine Transaktion eröffnet werden. Alle Felder in entsprechenden Objekten, welche bis zum Abschluss der Transaktion bearbeitet werden, werden im Anschluss in der Datenbank editiert (vgl. Quellcode 3).

```
1 User user = entityManager.find(User.class,1);
2 entityManager.getTransaction().begin();
3 user.setName = "Neuer Name";
4 entityManager.getTransaction().commit();
```

Quelltext 3: Beispielhaftes Laden und Bearbeiten eines Nutzers mit der ID 1

Die beschriebenen Operationen werden für sämtliche Fälle ausgeführt, in denen einzelne Objekte verwaltet werden. Abschließend müssen noch diejenigen Fälle betrachtet werden, in denen der Primärschlüssel nicht sind oder mehrere Objekte in einer Abfrage verwaltet werden sollen, z. B. beim tabellarischen Anzeigen aller registrierten Nutzer. Mittels JPA kann auf diese Daten über die Java Persistence Query Language (JPQL) zugegriffen werden, einer Sprache, welcher der Structured Query Langauge (SQL) angelehnt ist, jedoch entscheidende Unterschiede aufweist. Insbesondere werden über JPQL Objekte spezifiziert, wohingegen über SQL einzelne oder mehrere Daten aus Datenbankattributen verwaltet werden. Die Ergebnisse von JPQL-Abfragen sind damit immer einzelne oder mehrere Java-Objekte. Weiterhin wird eine JPQL-Abfrage nicht direkt an das Datenbankmanagementsystem übermittelt. Anstelle dessen übersetzt der JPA-Provider die Anfrage in ein SQL-Statement, welches die spezifizierte Datenbank unterstützt.

```
1
2 TypedQuery<User> query = entityManager.createQuery("select u
     from User u where u.email = ?1", User.class);
3 query = query.setParameter(1, emailAdress);
4 User user = tq.getSingleResult();
```

Quelltext 4: Beispielhaftes Auslesen eines Benutzers mittels JPQL

Im Quellcode 4 ist das Auslesen eines Nutzers anhand seiner Mailadresse, anstelle der Identifikationsnummer, dargestellt. Sämtliche Fälle zum Manipulieren der Datenbank werden auf Basis der drei oben genannten Operationen durchgeführt. Eine vollständige Übersicht aller Modellklassen des Projektes, auf denen Manipulationen durchgeführt werden, kann im Kapitel 5.4 eingesehen werden. Das Erstellen der Datenbanktabellen, auf Basis der definierten Modellklassen, wird automatisch durch den JPA-Provider vorgenommen. Aus diesem Grund wird dieser Teil nicht weiter beschrieben.

5.3. MVP

Dieser Abschnitt beschäftigt sich mit der Implementierung des in Abschnitt 5.3 erläuterten MVP-Entwurfsmusters und Verwendung von *CDI*. Dazu werden zunächst die grundlegenden Klassen und Interfaces beschrieben und im weiteren Verlauf das MVP-Prinzip an der Implementierung des Logins verdeutlicht.

5.3.1. Grundgerüst

Das Grundgerüst der im Projekt verwendeten MVP-Triade stellt die Klasse `AbstractPresenter` (vgl. Quellcode 5) dar, die den Konstruktor und die Methoden `show` und `reset` vorgeben. Über den Konstruktor werden *Model* und *View* (bestehend aus der *HumbleView* und *ViewLogic*) geladen. Die Methode `show` wird aufgerufen, um die Daten aus dem *Model*, mittels dem *Presenter*, in die *View* zu laden. *Presenter* und *View* müssen das Interface `ViewAccessor` (vgl. Quellcode 6) implementieren, um die *HumbleView*, die reine Implementierung des Designs, ohne zusätzliche Logik, anderen Komponenten zur Verfügung zu stellen. Neben dem aus *HumbleView*, *ViewLogic*, *Presenter* und *Model* bestehenden Grundgerüst, existieren global verfügbare *Services*. Diese stellen der gesamten Anwendung Daten zur Verfügung und synchronisieren diese zwischen den Clients.

```
4  public abstract class AbstractPresenter<M, V> {
5
6      protected final M model;
7      protected final V view;
8
9
10     public AbstractPresenter(M model, V view) {
11         if (model == null) {
12             throw new NullPointerException("Undefiniertes
                   Model!");
13         }
14         if (view == null) {
15             throw new NullPointerException("Undefinierte View
                   !");
16         }
17         this.model = model;
18         this.view = view;
19     }
20
21     public abstract void reset();
22
23     public abstract void show();
24 }
```

Quelltext 5: AbstractPresenter.java

```
0  public interface ViewAccessor {
1
2      <T> T getViewAs(Class<T> type) throws
           UnsupportedViewTypeException;
3  }
```

Quelltext 6: ViewAccessor.java

5.3.2. Kopplung mittels CDI

Um eine lose Kopplung zwischen den einzelnen Komponenten zu erreichen, werden diese mittels *CDI*, abhängig vom Kontext, injiziert.
Dabei werden die von einem Modul verwendeten Abhängigkeiten von außen dem Modul bekannt gemacht und im Modul lediglich als Abhängigkeit, also in Form eines *Interfaces* oder einer Klasse definiert. Dies bietet den Vorteil, dass, je nach Kontext, unterschiedliche Abhängigkeiten bzw. konkrete Objekte (*Beans*) geladen werden. Es existieren verschiedene Möglichkeiten, Abhängigkeiten bzw. *Beans* einer Klasse zu injizieren. Die gebräuchlichsten sind die *Field Dependency Injection* und *Constructor Dependency Injection*.

Scopes

Werden *Beans* in andere *Beans* injiziert, muss dieser Zustand, solange der Benutzer mit der Applikation interagiert, gespeichert werden. *Scopes* stellen eine Möglichkeit zur Konfiguration dar. Dabei handelt es sich um den Lebenszyklus und die Sichtbarkeit der injizierten *Beans*. In diesem Projekt wurden die folgenden *Scopes* verwendet.

- `@UiScoped`: *Beans* mit diesem *Scope* existieren, solange die *UI* existiert, in die sie injiziert wurden. Alle *Presenter* verwenden diesen *Scope*, das bedeutet, dass sich innerhalb einer *UI* alle injizierten *Presenter* vom selben Typ, auf das gleiche Objekt beziehen.
- `@ApplicationScoped`: Dieser *Scope* gibt an, dass die *Beans* solange existieren, wie die Applikation in Betrieb ist, sodass für alle Anwender die gleichen Objekte bereitgestellte werden. Dieser *Scope* wird für *Services*, wie dem `DatabaseService` verwendet, damit sich alle Client-Instanzen den gleichen Datenkontext teilen.
- `@Dependent`: *Beans* ohne explizite Definition einer *Scope* erhalten automatisch diesen *Scope*. Sie existieren solange wie die *Bean*, in die diese injiziert wurden.

CDI Events

Um eine Entkopplung zwischen den einzelnen Komponenten zu gewährleisten, erfolgt die Kommunikation von der *View* zum *Presenter* über *Events*. Dabei wird das *Observer-Pattern*[29] verwendet. Alle *Events* erben von der Klasse `EventObject` und werden mittels des *Interfaces* `javax.enterprise.event.Event` injiziert. Zum Aktivieren eines *Events* wird die Methode `javax.enterprise.event.Event.fire` aufgerufen. Alle Methoden, die auf dieses *Event* reagieren sollen, erhalten den Typ des *Events* als Parameter über die Annotation `@Observes`. Für eine genauere Unterscheidung, auf welche *Events* reagieren sollen, können dem Parameter weitere Annotationen hinzugefügt werden. Die Abbildung

[29] `http://docs.oracle.com/javaee/6/tutorial/doc/gkhic.html`

27 zeigt eine abstrahierte Aufrufkette. Die Kommunikation wird von der `ViewImpl` gestartet. Üblicherweise geschieht dies durch das Betätigen eines Buttons oder ähnliches, wodurch ein *Event* abgesendet wird. Der *Presenter* reagiert darauf und greift dann per direktem Aufruf auf die `ViewImpl` zu. Diese liefert dann in diesem Beispiel über die Klasse `ViewDesign` einen Wert zurück. Der *Presenter* kann diesen dann per direktem Aufruf an das Model übergeben.

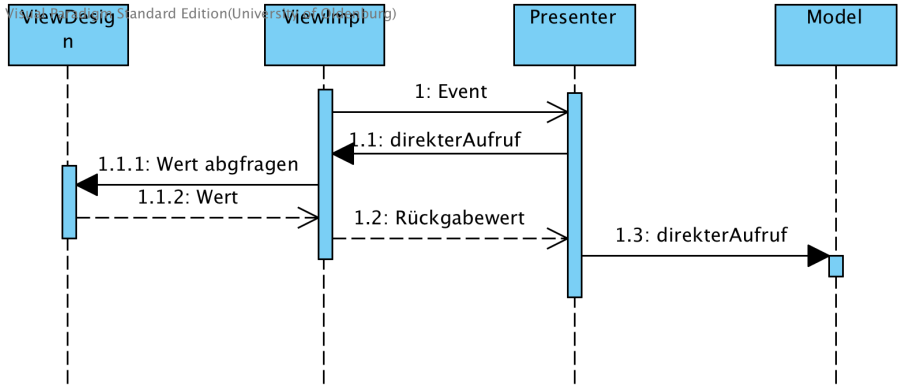


Abbildung 27: Sequenzdiagramm der MVP-Aufrufkette

5.3.3. Beispiel-Implementierung

Im Quellcode 7 wird die gekürzte Implementierung des `LoginPresenterImpl`, unter Verwendung der oben genannten Konzepte und Techniken, gezeigt. Der *Presenter* selbst wird per `@Inject` in der Klasse `WelcomeUI` injiziert. In Zeile 37 ist der *Scope* der *Bean* angegeben, über den Konstruktor in Zeile 50 werden *Model* und *View* injiziert.
Ab Zeile 44 wird ein *Event* injiziert und die Annotation `OriginalSender` definiert. Zeile 69 zeigt die Methode `onLoginAttempt`, die auf das *Event* `LoginAttemptEvent` reagiert und in der wiederum selbst das *Event* `loginSuccessEventSink` abgesendet wird.

```
37 @UIScoped
38 public class LoginPresenterImpl extends AbstractPresenter<
      LoginModel, LoginView> implements LoginPresenter {
39
40     @Inject
41     private Event<LoginSuccessEvent> loginSuccessEventSink;
42
43     @Inject
44     @OriginalSender
45     private Event<UserLoggedInEvent> userLoggedInEventSink;
```

```
46
47
48      @Inject
49      public LoginPresenterImpl(LoginModel model, LoginView
            view) {
50          super(model, view);
51      }
52
53      @Override
54      public <T> T getViewAs(Class<T> type) throws
            UnsupportedViewTypeException {
55          return view.getViewAs(type);
56      }
57
58      @Override
59      public void show() {
60
61      }
62
63      public void reset() {
64          view.reset();
65      }
66
67      @SuppressWarnings("unused")
68      private void onLoginAttempt(@Observes LoginAttemptEvent
            event) {
69          //do model.login(...)
70          //...
71          //if login sucessful
72          loginSuccessEventSink.fire(new LoginSuccessEvent(this
                , user, currentUser));
73
74      }
75
76  }
```

Quelltext 7: Auszug LoginPresenterImpl.java

5.4. Datenmodell

An dieser Stelle wird das Datenmodell beschrieben, auf das im Datenbankmanagement (vgl. Kapitel 5.2) zugegriffen wird. In der Grafik wird das entsprechende ER-Diagramm dargestellt. Wie in Kapitel 4.4.8 beschrieben, werden die Datenbanktabellen vollständig über den JPA-Provider erzeugt. Tabellen in der tatsächlichen Datenbank können daher, bei Wahl eines anderen JPA-Providers, unterschiedlich dargestellt werden. Das Diagramm spiegelt demnach die in Kapitel 4.4.8 beschriebenen Entitätenklassen wieder. Die zentrale Klasse `AbstractItem` ist die Oberklasse für alle Klassen, welche innerhalb der *Stages* dargestellt werden. Dies betrifft die Klassen `Challange`, `Suggestion`, `Project`, `Business Case`, sowie `Feedback`. Alle *Stages* sind über 1:1 Beziehungen miteinander verknüpft. `AbstractItem` beinhaltet damit sämtliche Attribute und Beziehungen zu anderen Klassen, welche von allen *Stages* genutzt werden. Die weitere wichtige Klasse ist die `User`-Klasse. Es bestehen Beziehungen zu allen anderen Klassen, da Nutzer als Ersteller oder Leiter von Elementen in allen Bereichen dargestellt werden.

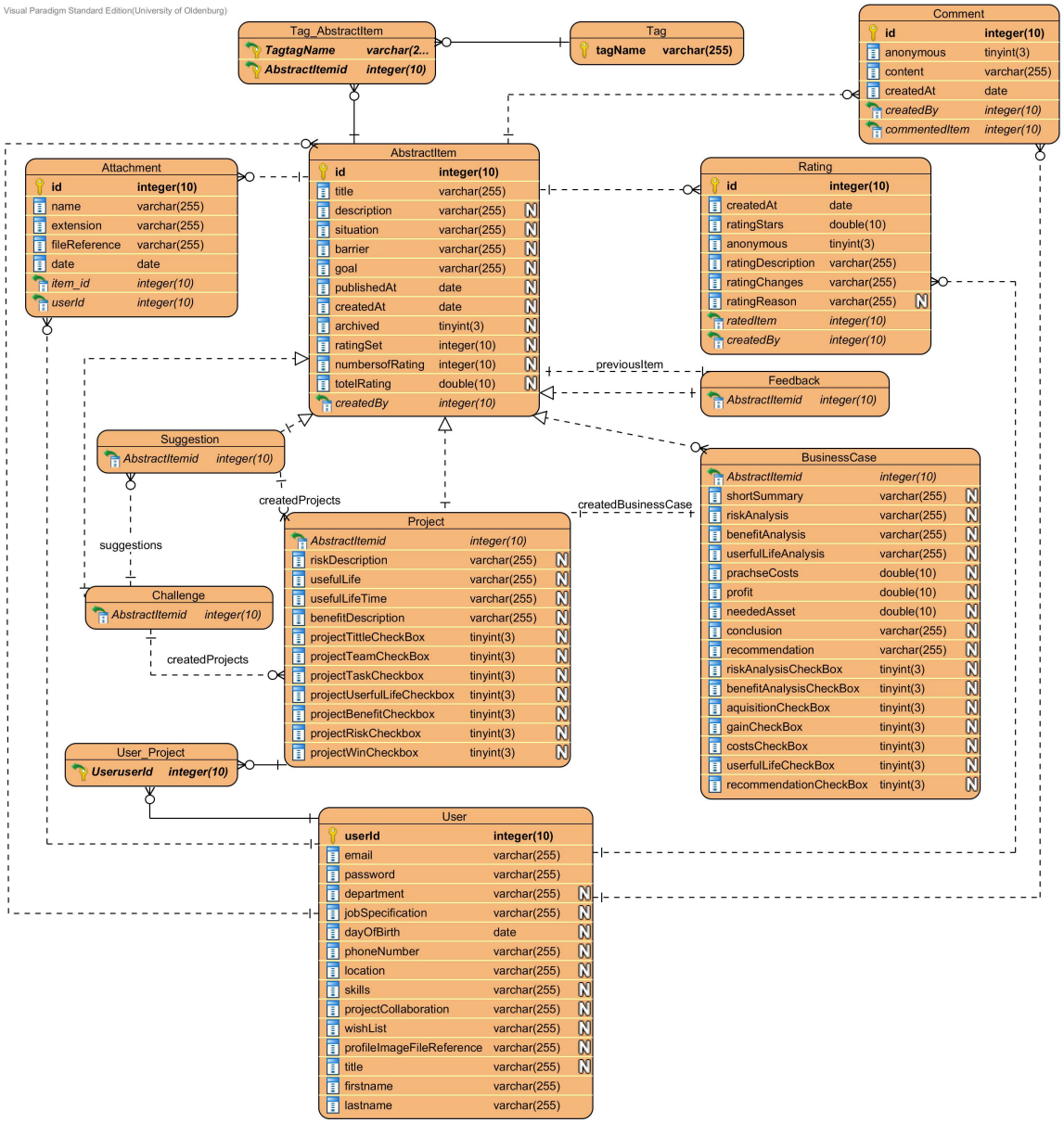


Abbildung 28: ER-Diagramm zur Darstellung des Datenmodells

6. Realisierung der Systemkomponenten

Im Folgendem werden die realisierten Komponenten vorgestellt. Dazu zählt besonders die technische Realisierung des Prozessmodels. Darüber hinaus werden die Grundfunktionalitäten und dessen Systemkomponenten genauer betrachtet.

6.1. Umsetzung UI-Design

Dieses Kapitel soll einen Überblick darüber verschaffen, wie das UI-Design innerhalb von Vaadin auf technischer Ebene realisiert wurde. Dabei wird zu Beginn die Grundstruktur des Themes erklärt und wie diese innerhalb der Entwicklung mit den einzelnen Bestandteilen aufgebaut wurde. Im nächsten Schritt wird auf wesentliche Bestandteile (Seitenstruktur, Layouts etc.), die in Kapitel 4.5 innerhalb der Konzeption des Designs aufgezeigt wurden, auf technischer Ebene in kurzer Form dargestellt.

6.1.1. Seitenstruktur

In diesem Kapitel soll aufgezeigt werden, wie die Seitenstruktur der Plattform technisch realisiert wird. Folgende Abbildung soll zunächst aufzeigen, in welche technischen Bereiche die Plattform strukturiert ist.

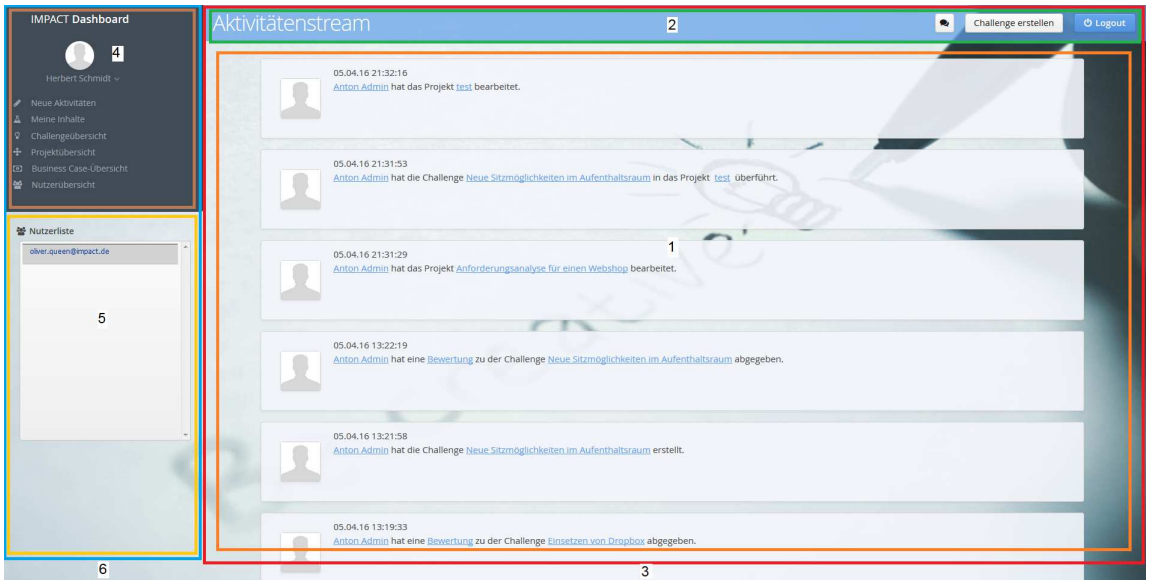


Abbildung 29: Seitenstruktur von IMPACT

Anhand dessen lässt sich erkennen, dass die Plattform im Bereich der Lobby in sechs unterschiedliche Bereiche unterteilt ist. Anzumerken ist hierbei, dass das Hauptlayout in der `LobbyNavigatorView` angesiedelt ist. Dieses unterteilt sich wiederum in andere Layouts, da die Plattform wie bereits erwähnt, in verschiedene Bereiche unterteilt ist.

```
765 private void setMainContent(Component c) {
766
767     if (mainComponent != null) {
768         rightLayout.removeComponent(mainComponent);
769     }
770     mainComponent = c;
771     rightLayout.addComponent(mainComponent, 1);
772     mainComponent.setStyleName("lobby-maincontent");
773 }
```

Quelltext 8: Seitenstruktur des Hauptlayouts

Das oben aufgezeigte Code-Beispiel verdeutlicht, welches Layout und welche Stilelemente dem Hauptbereich zugeordnet werden. In diesem Fall wird dem `rightLayout` die `mainComponent` hinzugefügt. Das `righLayout` repräsentiert dabei das Hauptlayout im rechten Bereich der Plattform, welches innerhalb der Abbildung 29 die (3) abdeckt. Die `mainComponent` stellt dabei die Hauptkomponente dar, welches die (1) abbildet. Dem `rightLayout` wird des Weiteren eine weitere Komponente hinzu geordnet, in diesem Fall stellt diese das `headerLayout` dar. Folgendes Code-Beispiel illustriert die Implementierung dieses Layouts:

```
777 private void buildHeader(String title) {
778
779     if (headerLayout == null) {
780     headerLayout = headerPresenterImpl.getViewAs(
        HorizontalLayout.class);
781     }
782     rightLayout.removeComponent(headerLayout);
783     headerPresenterImpl.reset();
784     headerPresenterImpl.setViewTitle(title);
785     rightLayout.addComponentAsFirst(headerLayout);
786     }
```

Quelltext 9: Seitenstruktur des Headers

Analog zur `mainComponent`, wird dem `rightLayout` das `headerLayout` zugeordnet, welches das Layout für den Header repräsentiert. Innerhalb der Abbildung 29 lässt sich dieser Bereich der (2) zuordnen.
Den nächsten Abschnitt stellt die Struktur das Menüs dar. Dieses ist wie die vorher aufgezeigten Layouts, in drei Bereiche aufgeteilt. Den obersten Bereich der Hierarchie bildet das `leftLayout`, welches innerhalb der Grafik 29 mit (6) gekennzeichnet ist. Diesem wird zum einen das `dashboardMenuLayout` zugeordnet, welches das Layout für die Navigation darstellt und in der Abbildung 29 als (4) ausgewiesen wird. Die letzte Komponente, welche hinzugefügt werden muss, repräsentiert die `userListComponent`, die für die Struktur der Nutzerliste verantwortlich ist (vgl. Abbildung 29 (5)). Die Implementierung dieser Struktur befindet sich in folgendem Code-Beispiel:

```
795 private void buildMenu() {
796
797     dashboardMenuLayout = dashboardMenuPresenterImpl.
           getViewAs(CustomComponent.class);
798     dashboardMenuPresenterImpl.reset();
799
800     leftLayout.addComponent(dashboardMenuLayout);
801     userlistPresenterImpl.reset();
802     userListComponent = userlistPresenterImpl.getViewAs(
           Component.class);
803     userlistPresenterImpl.show();
804     leftLayout.addComponent(userListComponent);
805
806 }
```

Quelltext 10: Seitenstruktur des Menüs

6.1.2. Vaadin-Theme

Im Kapitel 4.4.9 der Technologieentscheidungen hinsichtlich des Designs wurde erläutert, dass das Theme von Vaadin das zentrale Element abbildet, um eine Webanwendung zu gestalten. Im Zuge dessen wird das von Vaadin eingeführte Standard-Theme *Valo* verwendet, welches hauptsächlich genutzt wird hinsichtlich der Gestaltung der einzelnen Komponenten (vgl. Kapitel 4.4.9). Dieses Basis-Theme enthält eine Vielzahl grundlegender Stil-Definitionen, die für das Funktionieren der Komponenten essenziell sind. Vaadin regelt hierbei die Situation so, dass eine optimale Erweiterbarkeit mit weiteren Stil-Definitionen

möglich ist, hält das Theme dabei möglichst stabil und bietet dahingehend optimale Voraussetzungen für ein eigenes Theme. Da die Wahl auf ein SCSS basiertes Theme fiel, wurden bezüglich der eigenen Stilkomponenten die Dateien in diesem Stil angelegt und entwickelt, damit der Entwicklungsprozess dahingehend reibungslos funktioniert.

```
55 @import "../valo/valo";
56 @import "views/login";
57 @import "views/registration";
58 @import "views/dashboard";
59 @import "views/suggestionstage";
60 @import "common";
61 @import "components/tooltip";
62 @import "views/tutorial";
63 @import "views/businesscase";
64 @import "views/challenge";
65 @import "views/intention";
66 @import "views/passwordforgot";
67 @import "views/registration";
68 @import "views/tutorial";
69 @import "components/userlist";
70 @import "components/activitystream";
71 @import "components/chat";
72 @import "views/checklistBC";
73 @import "components/collaborativeRoom";
```

Quelltext 11: Einbindung von Valo und anderen Stil-Elementen in die Haupt SCSS-Datei

```
93  @mixin mytheme {
94  @include valo;
95  @include impact-login-view;
96  @include impact-registration-view;
97  @include impact-dashboard-view;
98  @include impact-challenge-view;
99  @include impact-intention-view;
100 @include impact-businesscase-view;
101 @include impact-common;
102 @include valo-tooltip;
103 @include impact-tutorial-view;
104 @include impact-userlist;
105 @include impact-activitystream;
106 @include impact-chat;
107 @include impact-checklistBC;
108 @include impact-collaborativeRoom;
109 }
```

Quelltext 12: Einbindung von Valo und anderen Stil-Elementen in das Mixin der Haupt SCSS-Datei

Die oben aufgeführten Code-Beispiele zeigen auf, welches Haupttheme innerhalb der Plattform genutzt und wie *Valo* sowie andere Stil-Elemente im Zuge dessen eingebunden werden. Bei *Valo* reicht die Einbindung über einen Import-Befehl und in ein `mixin` aus. Vaadin erkennt dabei automatisch, dass es sich um das eigene Theme handelt, sodass keine eigene Ordnerstruktur und Datei dafür angelegt werden müssen. Anders verhält sich die Situation bei den eigenen Gestaltungen. Wie zu sehen ist, werden neben *Valo* weitere Importierungen und Einbindungen in das `mixin` durchgeführt. Die `mixins` sind bei der Nutzung von SCSS essenziell, um CSS-Eigenschaften oder auch ganze Regelsätze auszulagern und somit nutzen zu können. `Mixins` stellen dabei eine Art CSS-Vorlage dar, die per `@include` an der gewünschten Stelle eingefügt wird (vgl. Code-Beispiel 12)
Diese Einbindungen unterteilen sich dabei in mehrere Bereiche, um eine bessere Übersicht zu gewährleisten und Redundanzen zu vermeiden. So werden bspw. Dateien, wo zwischen unterschiedlichen Sichten gewechselt wird (Login, Registration etc.), in den Ordnern *views* eingefügt. Wesentliche Komponenten, die innerhalb der Anwendung genutzt werden (Aktivitätenstream etc.), sind in dem Ordner *components* untergebracht. Die Ordnerstruktur

innerhalb dieses Ordners ist weitgehend frei gestaltbar. Die nachfolgende Abbildung demonstriert die Ordnerstruktur, die innerhalb der Entwicklung der Plattform eingesetzt wurde.

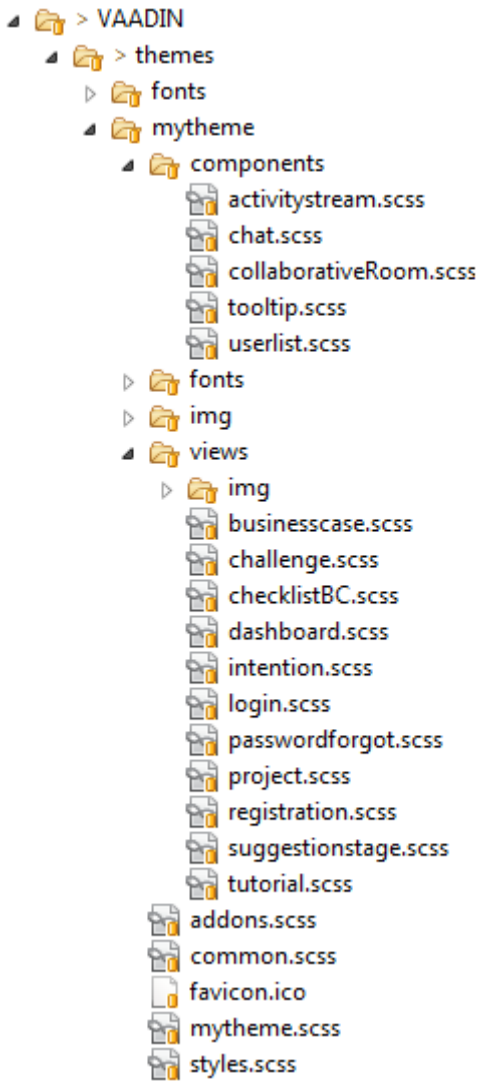


Abbildung 30: Theme Ordnerstruktur

Diese unterschiedlichen importierten Stile befinden sich innerhalb der `mytheme.scss`. Damit diese Stile in der Webanwendung angezeigt und einbezogen werden können, muss das Haupttheme in die UI-Klassen der Plattform integriert werden (vgl. Quellcode 13 und 14).

```
56 @SuppressWarnings({ "serial", "unused" })
57 @Theme("mytheme")
58 @Widgetset("de.impact.plattform.MyAppWidgetset")
59 @CDIUI("welcome")
60 public class WelcomeUI extends UI {
```

Quelltext 13: Einbindung des Hauptthemes in die WelcomeUI

```
105 @CDIUI("lobby")
106 @Theme("mytheme")
107 @Widgetset("de.impact.plattform.MyAppWidgetset")
108 @SuppressWarnings({ "serial" })
109 @Push(value = PushMode.AUTOMATIC, transport = Transport.
       WEBSOCKET)
110 public class LobbyUI extends UI {
```

Quelltext 14: Einbindung des Hauptthemes in die LobbyUI

Anhand der oben aufgezeigten Codebeispiele lässt sich erkennen, dass das anzuwendende Theme über die Annotation `@Theme` innerhalb der UI-Klasse festgelegt wird.

6.1.3. Verwendung von Stil-Definitionen

Im Zuge der Entwicklung der Webanwendung kam es in häufiger Form dazu, den Komponenten von Vaadin eigene Stil-Definition zu geben. Im Folgenden wird ein Code-Beispiel aufgezeigt, wie es innerhalb der Entwicklung eingesetzt wurde. Über die Methoden `setStyleName` bzw. `addStyleName` können einer Komponente weitere SCSS-Klassen mit zusätzlichen Stil-Elementen hinzugefügt werden. Folgende SCSS-Klasse sorgt bspw. dafür, um der Login-Form auf der Startseite, die standardmäßig mithilfe eines `FormLayouts` realisiert ist, eine zusätzliche Gestaltung zu geben.

```
20 .login-form {
21 @include valo-panel-style;
22 padding: $v-layout-margin;
23 @include valo-animate-in-fade($duration: 1s);
24 padding: round($v-unit-size / 1.5) round($v-unit-size / 1.5);
25 min-width: $v-font-size * 13;
26 max-width: 90%;
27 background: rgba(255,255,255,0.8);
```

```
28 @include animation(valo-animate-in-slide-up 800ms 10ms
       backwards, valo-animate-in-fade 600ms 100ms backwards);
29
30 > .v-spacing {
31 height: round($v-unit-size / 2);
32 }
33
34 .labels {
35 display: block;
36
37 .h4 {
38 margin: 0;
39 vertical-align: baseline;
40 }
41
42 .h3 {
43 margin: 0px auto;
44 float: right;
45 }
46 }
47
48 .fields .v-icon {
49 opacity: 0.3;
50 }
51 }
```

Quelltext 15: SCSS-Klassendefinition für die Login-Form

Wie zuvor erwähnt, müssen die Klassendefinitionen von SCSS-Dateien über die Methoden `setStyleName` bzw. `addStyleName` einer Komponente hinzugefügt werden. Im Bereich des Logins muss dabei die entsprechende SCSS-Klasse selektiert werden und der Java Designklasse `LoginViewDesign` hinzugefügt werden.

```
84 FormLayout loginForm = new FormLayout();
85 loginForm.addStyleName("login-form");
86 loginForm.setSizeUndefined();
87 loginForm.setMargin(false);
```

Quelltext 16: Einbindung der Login-Form SCSS-Klasse in eine Java-Klasse

Die zuvor erwähnte Methode sorgt hierbei dafür, dass die SCSS-Klasse der Login-Form dem `FormLayout` hinzugefügt wird. Dieses Beispiel demonstriert die Einbindung einer eigenen Stil-Definition. Es existiert daneben die Möglichkeit, bereits vordefinierte Stile von dem zuvor erwähnten Valo-Theme zu nutzen, um z. B. Überschriften oder andere Komponenten zu ändern, ohne eigene Stil-Definitionen verfassen zu müssen.

```
102 CssLayout buttons = new CssLayout();
103 buttons.setStyleName("buttons");
104 loginForm.addComponent(buttons);
105
106 buttons.addComponent(loginButton = new Button("Login"));
107 loginButton.setIcon(FontAwesome.SIGN_IN);
108 loginButton.setClickShortcut(ShortcutAction.KeyCode.ENTER);
109 loginButton.addStyleName(ValoTheme.BUTTON_PRIMARY);
```

Quelltext 17: Einbindung eines Valostils in eine die Login

Analog zur Einbindung eigener SCSS-Klassen, werden bei dem Valo-Theme ebenfalls Stil-Definitionen über die Methode `addStyleName` hinzugefügt. In dem oben genannten Code-Beispiel wird dem Login-Button ein Stil des Valo-Themes hinzugefügt, der dafür sorgt, dass dieser den primären Stil annimmt.

6.1.4. SCSS-Sprachkonstrukte und hierarchische Selektoren

Die Verwendung von SCSS ermöglicht es, die Stilelemente insbesondere in dem Punkt der Wartbarkeit aufzuwerten, indem Codeduplikationen in bestimmten Situationen vermieden werden können und die Lesbarkeit der Quelldateien erheblich verbessert wird. Die Syntax orientiert sich dabei an die CSS Syntax und erweitert diese um zusätzliche Features wie Variablen und `Mixins`, worauf im Folgenden näher eingegangen wird.

Variablen

Mittels SCSS besteht die Möglichkeit, Variablen, wie z. B. Schriften, Schriftgrößen oder Farben unter einem bestimmten Namen zu speichern. Diese werden einmalig deklariert und können anschließend innerhalb der SCSS-Klassen eingesetzt werden. Das folgende Code-Beispiel verdeutlicht, wie innerhalb der Webanwendung bestimmte Haupteigenschaften, wie z. B. die Schriftgröße oder die Hintergrundfarbe des Basis-Theme angepasst werden.

```
34 $v-focus-color: rgb(96, 160, 234) !default;
35 $v-error-indicator-color: #eb2977 !default;
36 $v-friendly-color: rgb(54, 185, 85);
37
38 $v-font-size: 15px !default;
39 $v-font-weight: 400 !default;
40 $v-unit-size: 32px !default;
```

Quelltext 18: Variablendeklaration in SCSS

Verschachtelung von Selektoren und Eigenschaften

Eine weiterer Aspekt, der bei der Verwendung von SCSS bedacht werden muss, ist die Hierarchie sowie deren Verschachtelung von Selektoren und Eigenschaften der Stil-Elemente. Somit lassen sich hierarchische Selektoren deutlich lesbarer und ohne Codeduplikation schreiben. Folgendes Code-Beispiel verdeutlicht, wie so eine Anordnung genutzt wurde, um den Titel der Navigationsleiste bei verschiedenen Pixelgrößen auf unterschiedliche Weise zu gestalten.

```
367 .v-ui[width-range~="0-800px"] {
368 .valo-menu-title {
369 @include valo-gradient($color: $valo-menu-background-color);
370 border-bottom-color: first-color(valo-border($color: $valo-
       menu-background-color));}
371 .valo-menu .v-menubar-user-menu {
372 color: inherit;} }
373 .v-ui[width-range~="801px-"] {
374 .valo-menu {
375 @include animation(valo-animate-in-slide-right 700ms 700ms
       backwards);}
376 .valo-menu-title {
377 background: transparent;
378 @include box-shadow(none);
379 border-bottom: none;}
```

Quelltext 19: Verschachtelung von Selektoren und Eigenschaften bei dem Menütitel der Navigationsleiste

6.1.5. Responsive Webdesign

Im Bereich der Design-Konzeption wurde in Kapitel 4.5 im letzten Abschnitt die Möglichkeit aufgezeigt, mit Hilfe von Vaadin und dem *Valotheme* im Bereich der Entwicklung das *Responsive Webdesign* einzubeziehen. In diesem Kapitel soll beispielhaft aufgezeigt werden, inwiefern diese Situation mit Hilfe von Vaadin technisch realisiert werden kann. Im Gegensatz zu herkömmlichen Webseiten, wo das *Responsive Webdesign* in der Regel innerhalb der CSS- bzw. SCSS-Datei umgesetzt wird, besteht bei Vaadin die Möglichkeit, einzelne Komponenten und Layouts in den einzelnen Java-Klassen direkt reaktionsfähig zu gestalten. Das nachfolgende Code-Beispiel illustriert die Situation, inwiefern eine einzelne Challenge innerhalb des Projektes ein reaktionsfähiges Design einnimmt:

```
189 VerticalLayout ChallengeForm = new VerticalLayout();
190 ChallengeForm.setSpacing(true);
191 ChallengeForm.setWidth("100%");
192 Responsive.makeResponsive(ChallengeForm);
```

Quelltext 20: Umsetzung von Responsive Webdesign innerhalb von Vaadin

Hierbei wird das Hauptlayout `Challengeform` der Erweiterung `Responsive` mit Hilfe der Methode `makeResponsive` hinzugefügt, welche letztendlich dafür sorgt, dass das Hauptlayout in relativen Abständen an die unterschiedlichen Displaygrößen angepasst wird.

6.2. Allgemeine Komponenten

In diesem Abschnitt werden die allgemeinen Komponenten der Plattform beschrieben. Zu diesen gehören genau die Komponenten, welche unabhängig von den Prozessphasen der Plattform angesiedelt sind. Zunächst werden die vorwiegend technischen Komponenten, wie z. B. *WebServlets*, Fehlerbehandlung oder die Navigation beschrieben. Im Abschnitt 6.2.1 wird die Umsetzung des Registrierungsprozesses erläutert. Nach der erfolgreichen Registration erfolgt die Anmeldung auf der Plattform. Neben der Login-Funktionalität ist zu gewährleisten, dass der Benutzer ausschließlich die Bereiche sehen kann, wofür er entsprechende Berechtigung verfügt. Letzteres wird im Abschnitt 6.2.3 dargestellt. Sollte ein Benutzer sein Passwort vergessen, muss er in der Lage sein, dieses wiederherzustellen. Gezeigt wird diese Funktionalität in Abschnitt 6.2.2. Die Ansicht und Verwaltung persönlicher Daten wird im Abschnitt 6.2.4 realisiert. Um einen schnellen Überblick über die wichtigsten Bereiche zu bekommen, gibt es diverse Übersichten im Kapitel 6.2.5. Darin hat der Benutzer die Möglichkeit, Inhalte zu sortieren und zu filtern.
Die beschriebenen Funktionen werden zudem als Komponenten im Komponentendiagramm in Abbildung 31 im Detail beschrieben. Zu sehen sind die wichtigsten Komponenten

des Projekts, darunter sämtliche *Presenter* der einzelnen Stages, des *Userprofils* und des *Rechtemanagements*, welche wiederum durch die Komponente *LobbyUI* aufgerufen und verwaltet werden. Die Komponenten zum *Login*, zur *Registrierung* sowie zum *Zurücksetzen des Passworts* werden durch die *WelcomeUI* verwaltet. Alle genannten Klassen greifen über verschiedene Interfaces, welche über die Komponente *DatabaseService* implementiert werden, auf die Datenbank zu. Dazu wird die weitere Schnittstelle *JPA* durch *EclipseLink* implementiert, sodass in der letzten Stufe der Zugriff auf das Datenbankmanagementsystem erfolgt.

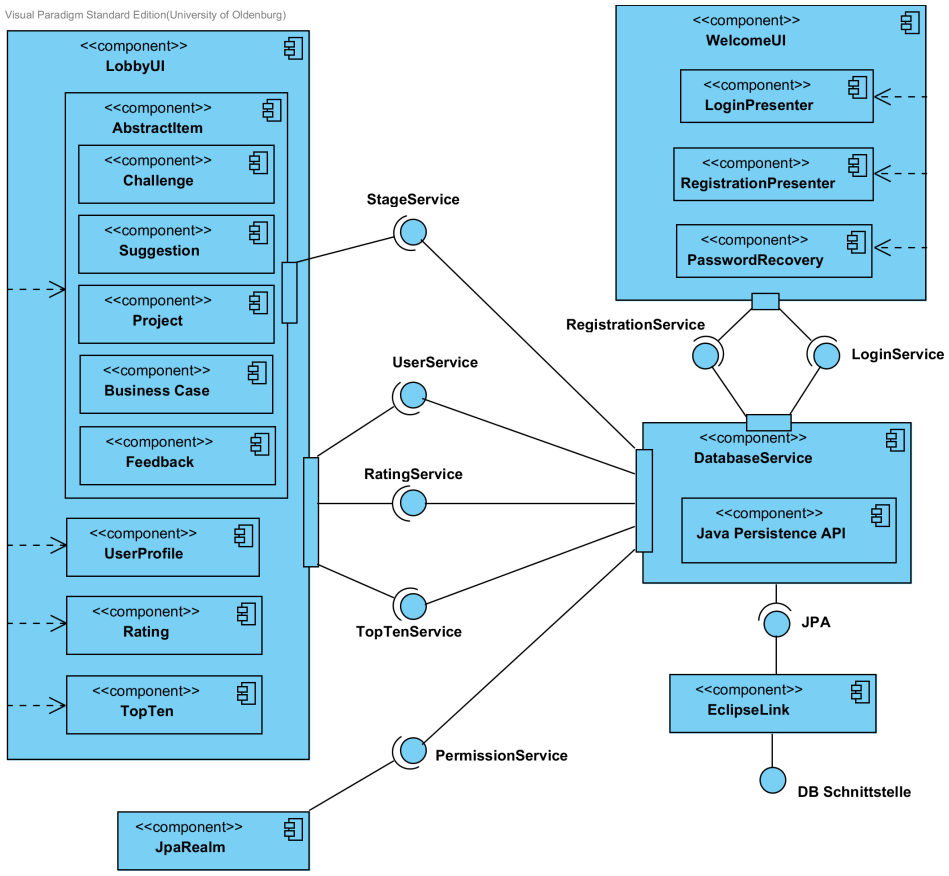


Abbildung 31: Komponentendiagramm zur Darstellung der Zugriffe auf die Datenbank

WebServlet

Die Klasse `WelcomeUIServlet` enthält die Konfiguration des *Servlet*. Als *Servlet* werden Java-Klassen bezeichnet, die die Anfragen der Clients entgegennehmen und beantworten. Innerhalb einer *Vaadin* Anwendung dient das *Servlet* zur Konfiguration (z. B. unter welcher *URI* die Anwendung erreichbar ist und wie lange eine Session ohne Interaktion eines Anwenders aufrecht erhalten wird) und Initialisierung der Webanwendung. Des Weiteren werden im *Servlet* `SessionDestroyListener` hinzugefügt, die bei dem Ablauf der *Session*, den eingeloggten Nutzer vom System abmelden.

UIs

Die Klassen `WelcomeUI` und `LobbyUI` stellen die User-Interfaces dar und stellen das Grundgerüst der Anwendung dar. Die `WelcomeUI` stellt die Oberfläche für unangemeldete Benutzer dar und die Klasse `LobbyUI`, die der angemeldeten Benutzer. Die Unterscheidung wird durch die Klasse *LoginAwareUIProvider* umgesetzt, indem geprüft wird, ob ein authentifiziertes *Subject* in der Session gespeichert ist. Der Quelltext 21 zeigt die Konfiguration der Klasse `LobbyUI` mittels Annotationen.

```
110 @Theme("mytheme")
111 @Widgetset("de.impact.plattform.MyAppWidgetset")
112 @Push(value = PushMode.AUTOMATIC, transport = Transport.
    WEBSOCKET)
```

Quelltext 21: Konfiguration LobbyUI

In der Zeile 110 wird angegeben, welches *Theme* (vgl. Abschnitt 6.1.2) für diese *UI* verwendet wird. Die darauf folgende Zeile enthält den Verweis auf das *Widget Set*, der clientseitigen Komponente, die letzte Zeile stellt die Konfiguration des *ServerPush* dar, in diesem Fall ist *Push* aktiviert und serverseitige Änderungen werden somit asynchron per *Websocket* an den Client gesendet.
Die wichtigsten Methoden der UI stellen die Methoden `init` und `detach` dar. Die `init`-Methode wird beim Aufruf der UI ausgeführt, in dieser Methode werden die *ErrorHandler* zur Fehlerbehandlung definiert und `View`-Objekte dem *Navigator* hinzugefügt. Die Methode `detach` wird aufgerufen, wenn eine *UI* von der *Session* entfernt wird. In der Klasse *LobbyUI*, dient die Methode dazu, einen Benutzer abzumelden, wenn dieser den Browser schließt. Des Weiteren werden in den Klassen *Events* abgefangen, die z. B. neue *Popup-Windows* öffnen.

Navigation

Die Navigation wird durch die Klasse `HistoryNavigator` und die dazugehörigen Implementierungen des Interfaces `com.vaadin.navigator.View` realisiert. Die Klasse `HistoryNavigator` enthält eine Liste von *Views*, die jeweils über ein bestimmtes *URI-Fragment* erreichbar sind, die mittels *#!* von der eigentlichen *URI* getrennt werden. Jede *UI* verwendet dabei einen eigenen *Navigator*. Für die Oberfläche `WelcomeUI` wird zwischen den *URI-Fragmenten login*, *registration*, *tutorial*, *passwordrecovery* unterschieden. Die Navigation erfolgt über das *URI-Fragment* in der Browserleiste durch den Nutzer oder durch den Aufruf der Methode `navigateTo` des *Navigator* über die `static`-Methode `UI.getCurrent().getNavigator()` oder innerhalb der Klasse `WelcomeUI` über `getNavigator()`. Der *Navigator* liefert für jedes der oben genanten *URI-Fragmente* die Klasse `WelcomeNavigatorView`. Der Quellcode 22 zeigt die `enter`-Methode, die von jeder Instanz des Interfaces `View` implementiert wird.

```
50 @Override
51 public void enter(ViewChangeEvent event) {
52
53     // je nach uri / navigateTo, den entsprechenden Inhalt
       setzen   switch (event.getViewName()) {
54     case WelcomeUI.LOGIN_VIEW:
55         setContent(loginPresenterImpl.getViewAs(CssLayout.
               class));
56         break;
57     case WelcomeUI.REGISTRATION_VIEW:
58         setContent(registrationPresenterImpl.getViewAs(
               VerticalLayout.class));
59     break;
60     case WelcomeUI.TUTORIAL_VIEW:
61         setContent(tutorialPresenterImpl.getViewAs(
               VerticalLayout.class));
62         break;
63     case WelcomeUI.FORGOTTENPASSWORD_VIEW:
64         setContent(passwordRecoveryPresenterImpl.getViewAs(
               VerticalLayout.class));
65     break;
66
67     default:
68         String viewName = event.getViewName();
```

```
69          String[] parameters = viewName.split("/");
70          if (Arrays.asList(LobbyUI.SECURED_VIEWS).contains(
               viewName) || Arrays.asList(LobbyUI.SECURED_VIEWS).
               contains(parameters[0])) {
71              Notification.show("Bitte einloggen", Type.
                   WARNING_MESSAGE);
72          }
73
74          else {
75              Notification.show("404 - Seite nicht gefunden",
                   Type.WARNING_MESSAGE);
76          }
77      event.getNavigator().navigateTo(WelcomeUI.LOGIN_VIEW);
78      break;
79      }
80
81  }
```

Quelltext 22: enter-Methode der Klasse WelcomeNavigatorView

Wird die Methode aufgerufen, wird anhand der `switch-case`-Anweisung des *URI-Fragments* über den *Presenter* der entsprechende Komponente, die *View* geladen. Wenn das *URI-Fragment* einer Komponente aufgerufen wird, für dessen Ansicht der Anwender sich authentifizieren muss, wird der Anwender aufgefordert sich anzumelden. Ist das *URI-Fragment* nicht bekannt, wird eine entsprechende Fehlermeldung angezeigt. In beiden Fällen wird der Anwender auf den Ansicht *Login* weitergeleitet.
In der Oberfläche *LobbyUI* wird zwischen folgenden *URI-Parametern* unterschieden: *mysubscriptions*, *mymessages*, *projects*, *challenges*, *mycontent*, *mynotifications*, *businesscases*, *userprofile*, *activitystream*. Das Verhalten der `enter`-Methode der dazugehörigen Klasse `LobbyNavigatorView` ist weitestgehend analog zu der oben gezeigten Methode. Allerdings können zur weiteren Unterscheidung von Zuständen Parameter *URI-Fragmenten* angehängt werden. Ein Beispiel stellt die *URI www.impact.de/#!challenges/ch1/s2/c1* dar. Die Parameter werden durch den Separator / kenntlich gemacht. Die Nutzung von Parameter erlaubt es Anwender Lesezeichen zu verwenden. Der Quellcode 23 zeigt einen Ausschnitt aus der `enter`-Methode der Klasse `LobbyNavigatorView`, in der Parameter von der *URI* getrennt werden.

```
185 // fetch parameters and split
186 String[] parameters = new String[0];
187 if (event.getParameters() != null && !event.getParameters().
    isEmpty()) {
188     parameters = event.getParameters().split("/");
189 }
```

Quelltext 23: Auszug der enter-Methode der Klasse LobbyNavigatorView

Je nach Anzahl der angehängten Parameter werden dementsprechend eine Challenge, ein Lösungsvorschlag oder ein dazugehöriger Kommentar angezeigt, indem die Parameter in mehreren verschachtelten *if*-Anweisungen ausgewertet werden.

Nutzerliste

In diesem Abschnitt wird die Nutzerliste beschrieben. Die Nutzerliste zeigt alle Benutzer, die gerade online sind. Über diese Liste ist es möglich, mit dem entsprechenden Benutzer einen Chat zu starten. Die Nutzerliste befindet sich unter dem Menü. In der Liste werden die Email-Adressen der Benutzer angezeigt, die gerade online sind. Mit einem Klick auf den Benutzer öffnet sich ein Chatfenster. Dieser Vorgang wird im Abschnitt 6.12 genauer erläutert.
Sobald ein Benutzer sich am System anmeldet, wird ein `UserLoggedInEvent` an alle angemeldeten *UI* versendet und der Client fordert über das `UserlistModelImpl` eine Liste mit allen angemeldeten Nutzern an. Diese werden über den `UserlistPresenterImpl` in der *View* in einer Liste angezeigt wird. Wird ein Benutzer vom System abgemeldet, erfolgt dies analog mittels einem `UserLoggedOutEvent`.

Server Push

Der *Server Push* dient dazu, eine *UI* durch einen anderen *Client* oder durch einen Hintergrund-*Thread* zu aktualisieren. Damit die Änderung nicht erst bei der nächsten Anfrage an den Server im Browser erscheint, bietet das *Vaadin*-Framework die Möglichkeit an, auf eine `UI`-Klasse mittels der Methode `access` zuzugreifen. Die Konfiguration des Übertagungsmodus wird per Annotation in der Klasse `UI` durchgeführt, wie bereits im Paragraph *UI* erläutert. Im Projekt wird *Server Push* dazu genutzt, um globale *Events* an andere Clients weiterzuleiten. Dies erfolgt in der Regel über Klasse `BroadcastServiceImpl`, die die Interfaces `BroadcastService`, `ChatService` und `ActivityService` implementiert. Dazu enthält die Klasse verschiedene Listen, in denen die Instanzen der aktiven *UIs* gespeichert wird. Je nach Verwendungszweck ist der dazugehörige Benutzer oder nur die Instanz der *UI* bekannt. Der Quellcode 24 der Methode `onUserLogin` zeigt, wie der entfernte Zugriff

auf andere *UIs* erfolgt, wenn ein Benutzer sich am System anmeldet. Ab Zeile 170 wird innerhalb einer Schleife, über eine Instanz des `ExecutorService`, für jede `UI`-Instanz ein `UserLoggedInEvent` abgesendet. Der `ExecutorService` startet für jeden Aufruf einen eigenen *Thread* und erlaubt somit die parallele Ausführung. Dies gewährleistet eine zügige Verarbeitung.

```
167 private synchronized void onUserLogin(@Observes
       @OriginalSender UserLoggedInEvent event) {
168 // für jede ui, ui.access aufrufen, in der ein
       userLoggedInEvent abgesendet wird
169     userService.addUserToOnlineList(event.getUser());
170     uis.asMap().forEach((k, v) -> v.forEach(ui ->
171         executorService.execute(() -> ui.access(() ->
172             userLoggedInEventSink.fire(event)))));
173 }
```

Quelltext 24: AbstractPresenter.java

Fehlerbehandlung

Vaadin bietet im Standard bereits eine eingebaute Fehlerbehandlung für Komponenten an (vgl. [Grö15], S.93ff.). Scheitert die Überprüfung einer solchen Komponente, wird im Tooltip eine vordefinierte oder individuelle Fehlermeldung wiedergegeben. Mittels der Funktion `setComponentError` können die Fehlermeldungen für jede Komponente individualisiert gestaltet werden. Fehlermeldungen, die nicht abgefangen werden, landen beim `DefaultErrorHandler`. Der von Vaadin vordefinierte `DefaultErrorHandler` wurde erweitert, um zwischen Produktions- und Debugmodus zu unterscheiden. Diese Anpassungen wurden in der `WelcomeUI` und `LobbyUI` vorgenommen, damit alle Bereiche der Plattform abgedeckt werden.

```
414 public void error(com.vaadin.server.ErrorEvent event) {
415 String cause = "<b>Es ist ein unerwarteter Fehler aufgetreten
       :</b><br/>";
416 String fullStack = "";
417 if (!VaadinService.getCurrent().getDeploymentConfiguration()
418 .isProductionMode()) {
419     for (Throwable t = event.getThrowable(); t != null; t = t
420     .getCause())
421         if (t.getCause() == null) // Der finale Fehler wird
            ermittelt
```

```
422                 cause += t.getClass().getName() + "<br/>";
423                 event.getThrowable().printStackTrace();
424                 StringWriter errors = new StringWriter();
425                 event.getThrowable().printStackTrace(
426                 new PrintWriter(errors));
427                 fullStack = errors.toString();
428 }
429 Notification.show("Fehler", cause, Type.ERROR_MESSAGE);
430 }
```

Quelltext 25: Erweiterung des DefaultErrorHandler in der LobbyUI

In dem oben aufgeführtem Code-Ausschnitt wurde bei `String cause` die Fehlermeldung zunächst angepasst. Sollte sich die Plattform im Debugmodus befinden, wird die darauffolgende if-Abfrage ausgeführt und der finale Grund der Fehlermeldung ermittelt. Dieser wird nun zusätzlich zu der Fehlermeldung ausgegeben.

Validierung von Eingaben

Im Rahmen sämtlicher zu treffenden Eingaben findet innerhalb der Plattform eine Validierung dieser statt. Dies dient vorwiegend zur Vermeidung von Fehleingaben. Weiterhin wird durch diese sichergestellt, dass beim Ausfüllen von Feldern genügend Informationen eingetragen werden, sodass diese für andere Anwender nachvollziehbar sind.
Vor diesem Hintergrund sind Formulare mit Pflichtfeldern versehen um einen Beiträge möglichst aussagekräftig zu gestalten. Erzielt wird dies indem die entsprechenden Felder (required) gesetzt werden. Der Quellcode 26 veranschaulicht dieses Vorgehen am Beispiel der Challenge. Bei der Erstellung einer Challenge (vgl. Abschnitt 6.3.1) ist unter anderem der Titel als Pflichtfeld gesetzt. Dazu wird es über die Methode `setRequired(true)` als Pflichtfeld gesetzt. Um den Nutzer zu verdeutlichen, dass es sich um ein Pflichtfeld handelt, und welche Bedingungen für dieses existieren ist zusätzlich eine Fehlermeldung (vgl. Quellcode 26, Zeile 135) definiert. Diese kann frei formuliert werden, sollte dabei aber prägnante Aussagen treffen.

```
134     challengeTitleTextField.setRequired(true);
135     challengeTitleTextField.setRequiredError("Bitte gib einen
            Namen für deine Challenge an.");
```

Quelltext 26: Definition von Pflichtfeldern bei der Erstellung einer Challenge

Neben Pflichtfelder sind für Eingaben, wie bei Passwörtern, weitere Bedingungen angeknüpft. Der Quellcode 27 veranschaulicht dieses Vorgehen. Dabei wird dem /emphpasswordField ein `RegexpValidator` hinzugefügt. Dieser definiert die Anforderungen, die an ein Passwort beim Registrieren gestellt werden.

```
138     passwordField.addValidator(new RegexpValidator("
            (?=.*[0-9]).(?=.*[A-z]).{7,}","Bitte überprüfe deine
            Eingabe!"));
```

Quelltext 27: Definition von Passwortbedingungen

Für den dargestellten Fall muss ein Passwort aus mindestens acht Zeichen bestehen, wobei zumindest eine Zahl und ein Buchstabe enthalten sein müssen. Sonderzeichen sind nicht zulässig. Sollte der Anwender beim Ausfüllen diese Bedingungen nicht erfüllen, wird er durch eine Fehlermeldung darauf hingewiesen seine Eingabe zu überprüfen.

Archivierung

Die Beschreibung der Realisierung der Archivierung lässt sich in zweit Bereiche aufteilen. Der erste Abschnitt beinhaltet das Archivieren eines Objekts und der zweite bezieht sich auf das neu Anlegen von archivierten Objekten.

Archivieren

Eine Archivierung kann für eine Challenge, ein Projekt oder einen Business Case vorgenommen werden. Im Folgenden wird die Archivierung der Challenge beschrieben. Die Archivierung eines Projekts oder Business Cases unterliegt demselben Prozess. Zum Archivieren muss man die zu archivierende Challenge anzeigen und kann dort unter bestimmten Voraussetzungen die Challenge über einen Button *Archivieren* archivieren. Dieses wird über die Methode `setArchiveButton()` der `ChallengeView` beim Anzeigen einer Challenge überprüft (siehe Kapitel Challenge anzeigen). Bedingung für die Archivierung von Challenges ist, dass das Veröffentlichungsdatum länger als vier Wochen zurückliegt oder die Bewertung weniger als zwei Sterne beträgt und mindestens fünf Bewertungen abgegeben wurden. Dann wird der Button *Archivieren* in der Challenge angezeigt. Beim Klick auf den *Archivieren* Button wird das Event `ArchiveChallengeEvent` gefeuert. Daraufhin wird über das `Model` die Methode `archiveChallenge(challenge)` aufgerufen, über die das Attribut `archived` auf `true` gesetzt wird. Anschließend wird die `ChallengeView` über die Methode `populateView()` aktualisiert.

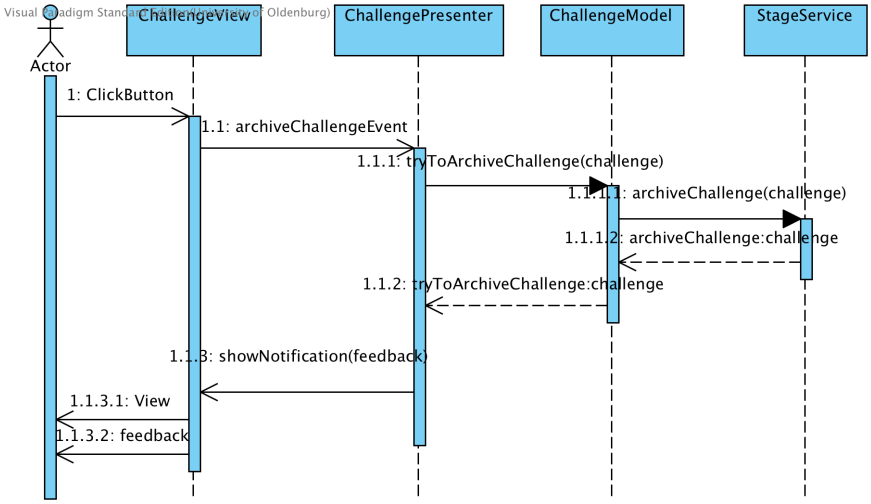


Abbildung 32: Sequenzdiagramm: Challenge archivieren

Neu Anlegen

Archivierte Challenges kann man in der Challengeübersicht im Tab *archivierte Challenges* einsehen. Hierfür wird gefiltert ob das Attribut `archived` der Challenge auf `true` gesetzt ist. Durch Doppelklick auf ein Element ist die Anzeige der Challenge möglich.
Bei dem Versuch der Anzeige einer Challenge wird vorab über das Attribut `archived` überprüft, ob es sich um eine archivierte Challenge handelt. Ist dieses der Fall, sind keine Änderungen an der Challenge möglich. Es können weder Bewertungen noch Kommentare oder Lösungsvorschläge hinzugefügt werden. Wenn der Benutzer entsprechende Berechtigungen verfügt, kann er über den Button *Neu Anlegen* die Challenge erneut anlegen und anschließend wie gewohnt bearbeiten. Beim Klick auf den Button *Neu Anlegen* wird das Event `RebuildChallengeEvent` gefeuert. Dieses wird über den `ChallengePresenter` abgefangen. Dort wird das Attribut `archived` der Challenge auf `false` gesetzt. Dieses wird über das `Model` der Methode des `StageServices` übergeben. Anschließend wird die `ChallengeView` über die Methode `populateView()` aktualisiert.

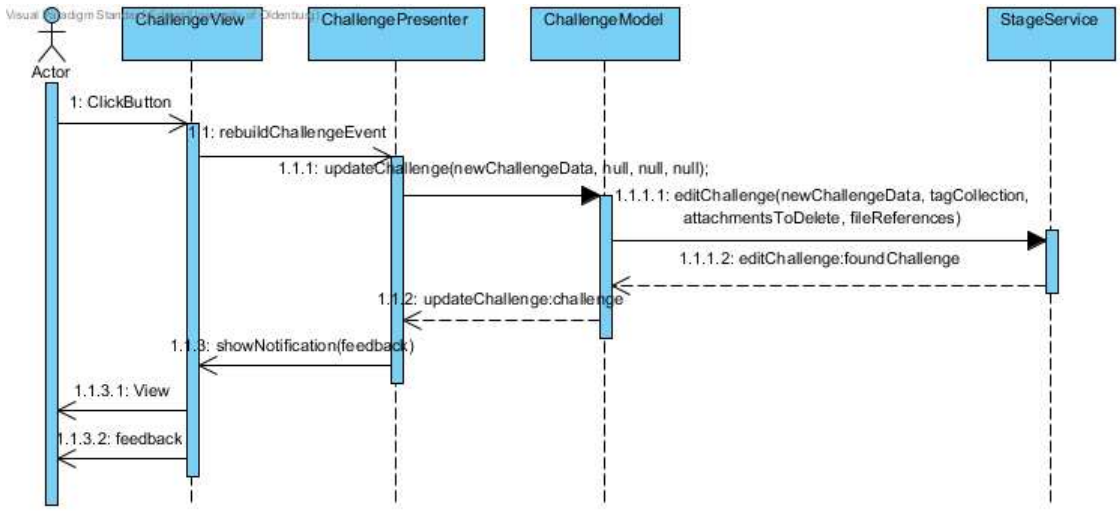


Abbildung 33: Sequenzdiagramm: Challenge neu anlegen

Gemeinsames Bearbeiten des Projekts

Dieser Abschnitt beschäftigt sich mit dem Sperren der Bearbeitungsansicht des Projekts. Das Prinzip und die technischen Umsetzung lässt sich für alle Ansichten anwenden, die durch mehrere Anwender bearbeiten werden können. Derzeitig ist diese Funktionalität nur für das Projekt umgesetzt. Um gemeinsames Arbeiten an einem Projekt zu realisieren ohne das Änderungen überschrieben werden, wird das Prinzip der exklusiven Sperre angewendet. Sobald ein Benutzer die Bearbeitungsansicht für ein Projekt aktiviert, wird für alle anderen Benutzer die Ansicht gesperrt. Nach der Bearbeitung werden andere Benutzer, die sich in der selben Ansicht befinden, über die Aktualisierung des Projektes informiert und müssen die Ansicht aktualisieren bevor sie das Projekt bearbeiten können. Realisiert wird das Sperren über die Klasse `BroadcastServiceImpl`. Innerhalb der Klasse wird für jedes Projekt eine Liste von `UI`-Objekten geführt. Sobald ein Benutzer die Bearbeitungsansicht öffnet, werden an alle *UI*s ein *Event* gesendet, durch das der Button und damit die Ansicht gesperrt wird.Die Abbildung 34 zeigt eine Nachricht welche Nutzer gerade das Projekt bearbeitet und das diese Möglichkeit haben ihm eine Nachricht zu schicken.

Abbildung 34: Rückmeldung bei gesperrten Projekt

Benutzer, welche die Projektansicht öffnen, werden der Betrachter-Liste hinzugefügt. Verlässt ein Benutzer die Ansicht, meldet sich ab oder schließt den Browser, wird die dazugehörige `UI` aus der Liste entfernt und ggf. andere *UIs* darüber informiert das die *View* wieder freigegeben ist.

```
377
378     case UNREGISTERVIEWER:
379         projectViewer.remove(event.getId(), event.getUi());
380     case UPDATED:
381     case UNLOCKED:
382     //versuchen den user zu entfernen
383         if(lockedProjects.remove(event.getUi(), event.getId()
                )) {
384             projectLockedByUser.remove(event.getId(), event.
                    getUser());
385             for (UI ui : projectViewer.get(event.getId())) {
386                 executorService.execute(() -> ui.access(() ->
                        viewStatusChangeEvent.fire(event)));
387         }
388     }
389     break;
```

Quelltext 28: Verarbeitung eines Sperrereignisses

Der Quellcode 28 zeigt einen Ausschnitt der Methode `onViewStatusChanged` in der Klasse `BroadcastServiceImpl`, die auf ein `ViewStatusChangeEvent` aus der *View* reagiert. Abhängig von dem Status des *Events*, wird die Ansicht gesperrt, entsperrt oder ggf. der Benutzer zur Aktualisierung aufgefordert. In Zeile 377 wird der Fall erörtert, wenn ein Anwender nicht mehr länger ein Projekt betrachtet. Zunächst wird die *UI* aus der Liste der Betrachter entfernt. In Zeile 382 wird geprüft, ob diese *UI* die Bearbeitungsansicht zunächst gesperrt hat. Ist dies der Fall, wird der Nutzer aus der Liste gelöscht und an alle übrigen Betrachter des Projektes das `ViewStatusChangeEvent` gesendet. Das Versenden eines Events, an eine andere *UI*, erfolgt dabei über eine Instanz des `ExecutorService`, der für jede *UI* einen eigenen Thread eröffnet, damit die Verarbeitung parallel erfolgen kann.

6.2.1. Registration

Die Nutzung der Innovationsplattform setzt eine Registrierung des Anwenders voraus. Im Rahmen dieses Prozesses müssen relevante Daten vom Anwender eingetragen werden, um sich ein Nutzerkonto zu erstellen. Diese werden im Folgenden beschrieben.

Oberfläche

Über die Startseite gelangt der Anwender zum Registrierungsmenü. Innerhalb dieses Menüs muss der Nutzer die aufgeführten Pflichtfelder befüllen:

- Vorname,
- Nachname,
- E-Mail,
- E-Mail Wiederholung,
- Passwort und
- Passwort Wiederholung

Zusätzlich kann der Anwender die Felder *Abteilung* und *Tätigkeitsbeschreibung* freiwillig befüllen (vgl. Abbildung 35).
Wie bereits im Kapitel 3.2.1 beschrieben, wird die eingegebene E-Mail und das Passwort überprüft. Zur Sicherheit wird das Passwort beim speichern (zusätzlich zur Validierung) mittels einer Hashfunktion verschlüsselt. Die Wiederholung der E-Mail und des Passwortes werden gefordert, um erste Eingabefehler zu verringern. Die technische Umsetzung wird nachfolgend beschrieben.

Abbildung 35: Registrierungsbildschirm

Technische Umsetzung

Mit dem aktivieren des Buttons zur Registrierung wird das `registrationAttemptEvent` gesendet. Anschließend wird ein Viewwechsel vollzogen, indem der `RegistrationPresenter` befüllt wird. Dem Anwender werden obligatorische und optionale `TextFields` zur Verfügung gestellt, welche im vorigen Abschnitt beschrieben wurden. Um unvollständige und falsche Eingaben zu vermeiden, wird für die Felder E-Mail und Passwort eine zweite Eingabe gefordert und der Button zum speichern erst nach vollständiger Registrierung aktiviert. Dazu wurden in der `RegistrationView` für die

Pflichtfelder `ValueChangeListener` registriert (vgl. Quellcode 34). Sind die Daten erfasst, wird die Methode `registerUser` des `RegistrationModel` aufgerufen. Von dort aus wird der implementierte `RegistrationService` ausgeführt. Der `RegistrationService` ist dabei als ein Interface realisiert, wobei die aufgerufene Funktion im `DatabaseService` implementiert ist. Anschließend erhält der Anwender bei erfolgreicher Registrierung ein Feedback in Form einer `notification` und wird durch das `changeToLoginEvent` zur Startseite geleitet, um die Registrierung mittels einer Anmeldung abzuschließen. Das Sequenzdiagramm 36 veranschaulicht diesen Vorgang.

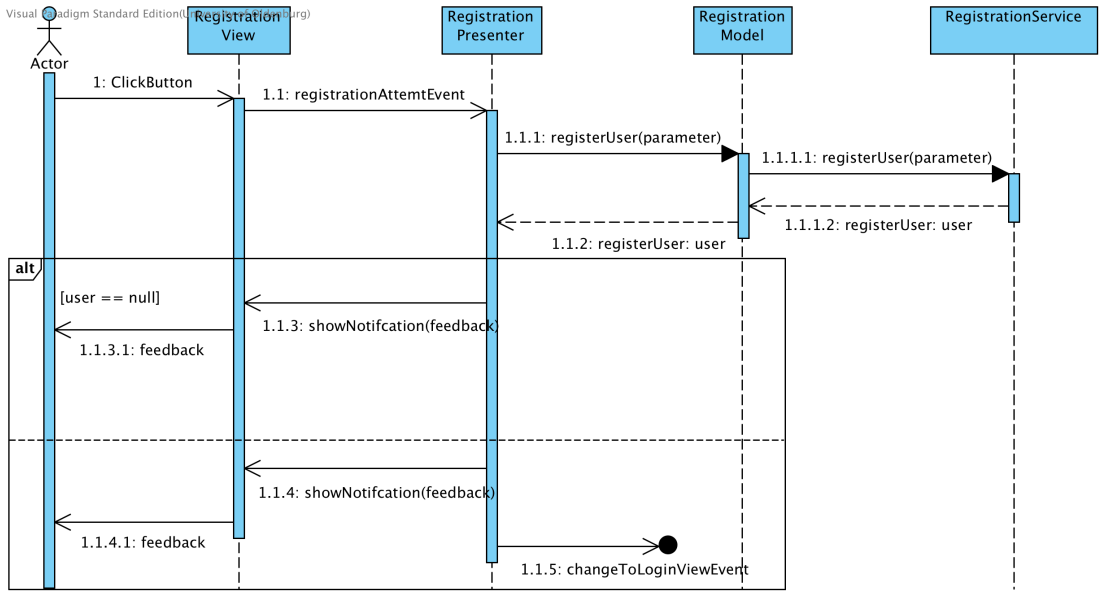


Abbildung 36: Sequenzdiagramm: Registrierung

6.2.2. Passwort Wiederherstellung

Dem Anwender wird mit der Funktionalität Passwort Wiederherstellen die Möglichkeit gegeben, dass im Falle eines vergessenen oder abgelaufenen Passworts, ein neues anzufordern.

Oberfläche

Um ein neues Passwort zu beantragen, muss der Anwender den vorgesehenen Link zur Passwort Wiederherstellung, auf der Startseite aktivieren. Innerhalb der neuen Eingabemaske muss der Anwender seine registrierte E-Mail eingeben (vgl. Abbildung 37). Anschließend erhält er eine neue Nachricht, welche ein neues Passwort und die Aufforderung zum anschließenden ändern des neuen Passworts enthält.

Abbildung 37: Passwort Wiederherstellung

Technische Umsetzung

Sobald der Anwender ein neues Passwort mit der Aktivierung des Buttons anfordert, wird ein `passwordRecoveryAttemptEvent` gesendet. Anschließend wird vom `PasswordRecoveryPresenter` die Methode `recoverPassword` aufgerufen. Die anschließende Methode `getUserByMail` des `PasswordRecoveryModels` wird aufgerufen. Dieser überprüft, ob der Anwender bereits in der Datenbank registriert wurde.
Bei erfolgreicher Prüfung werden die Services `MailService` und `PasswordGeneratorService` aufgerufen. Mittels dieser Services wird ein neues Passwort generiert und eine E-Mail an den Anwender versendet. Der Nutzer erhält anschließend ein Feedback, welches den Prozess abschließt. Das nachfolgende Sequenzdiagramm 38 veranschaulicht diesen Prozess.

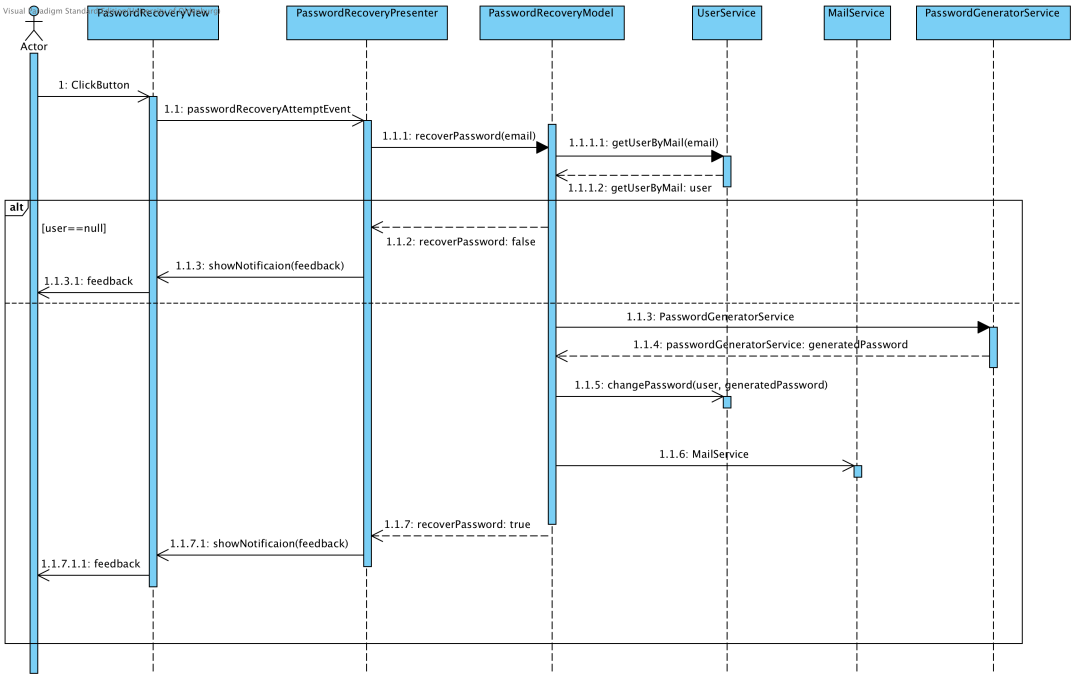


Abbildung 38: Sequenzdiagramm: Passwort Wiederherstellung

6.2.3. Login und Berechtigungskonzept

Die Anmeldung ermöglicht dem Anwendern den Zugang zur Innovationsplattform. Sie bildet den zentralen Startpunkt der Plattform und setzt eine vorangestellte Registrierung voraus, diese wird im Abschnitt 3.2 nachfolgend beschrieben.

Login-Oberfläche

Die Anmeldung stellt die zentrale Startseite der Innovationsplattform da. Sie verfügt über zwei Pflichtfelder, in denen der Benutzername und das Passwort eingegeben werden muss, sowie einen Button um die getätigten eingaben zu bestätigen. Zusätzlich verfügt die Oberfläche über drei Links, mit denen der Registrierungsprozess, die Anforderung eines neuen Passwortes und ein Tutorial gestartet werden kann (vgl. Abbildung40).

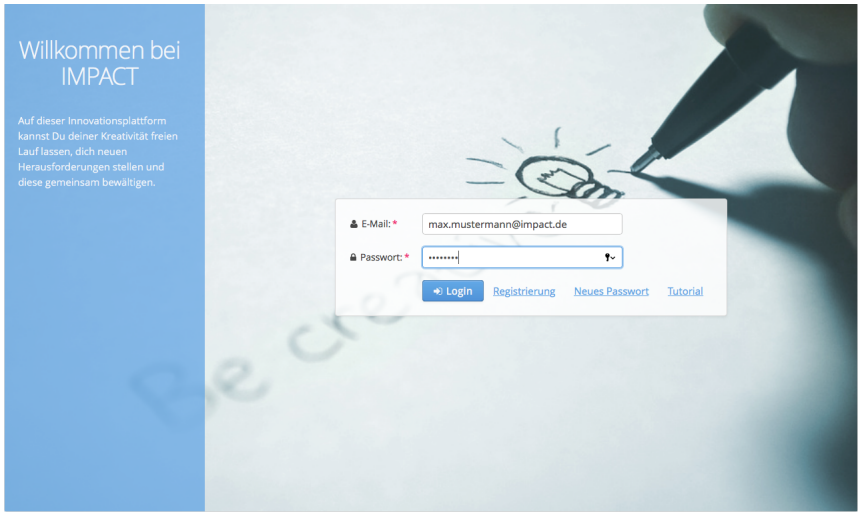


Abbildung 39: Anmeldung in der Plattform

Technische Umsetzung und Berechtigungskonzept

Die Umsetzung des Login- und Berechtigungskonzepts erfolgt, wie bereits in Kapitel 4.4.5 beschrieben, unter Verwendung des Frameworks Apache Shiro. Die Architektur des Frameworks wird in Abbildung 41 schematisch dargestellt.

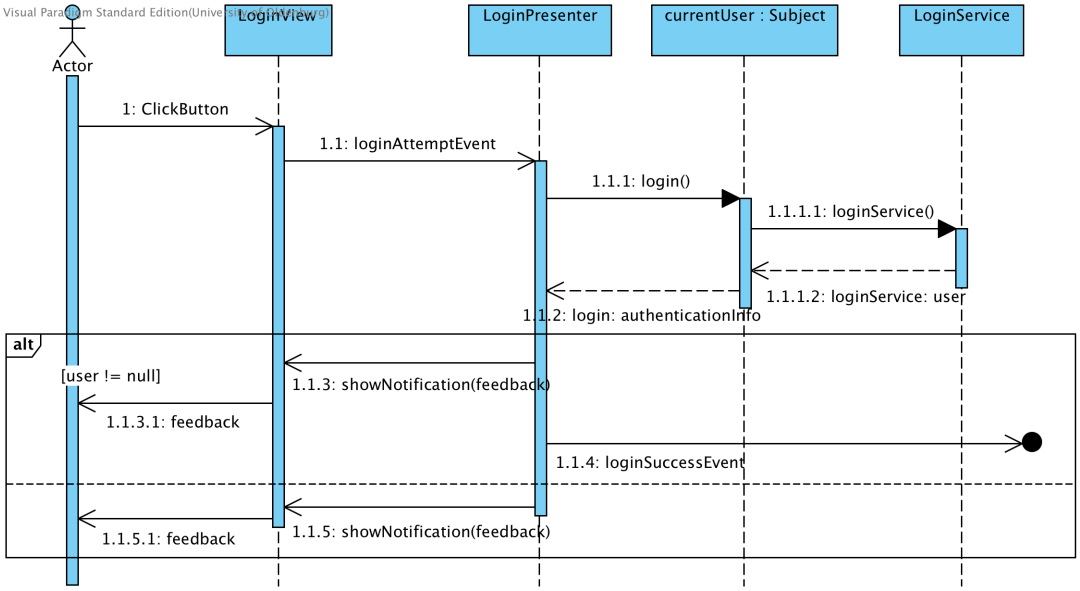


Abbildung 40: Sequenzdiagramm: Anmeldung

Das zentrale Element des Frameworks ist der Security Manager, welches direkt zum Start der Applikation instantiiert wird. Dabei werden zunächst die konfigurierten Realms geladen, Quellen, aus denen benötigte Informationen wie Benutzerdaten ausgelesen werden können. Die Konfiguration kann in der Übersicht der Rechtekonfiguration eingesehen werden. Entsprechend sind mehrere Quellen für Login-Informationen möglich. In aktuellen Form des Projekts werden Nutzerdaten in der in Kapitel 5.4 beschriebenen Datenbank verwaltet. Dazu wurde der Realm JpaRealm erstellt, welche sowohl die Authentifzierung verwaltet, als auch mögliche Authorisierungen.

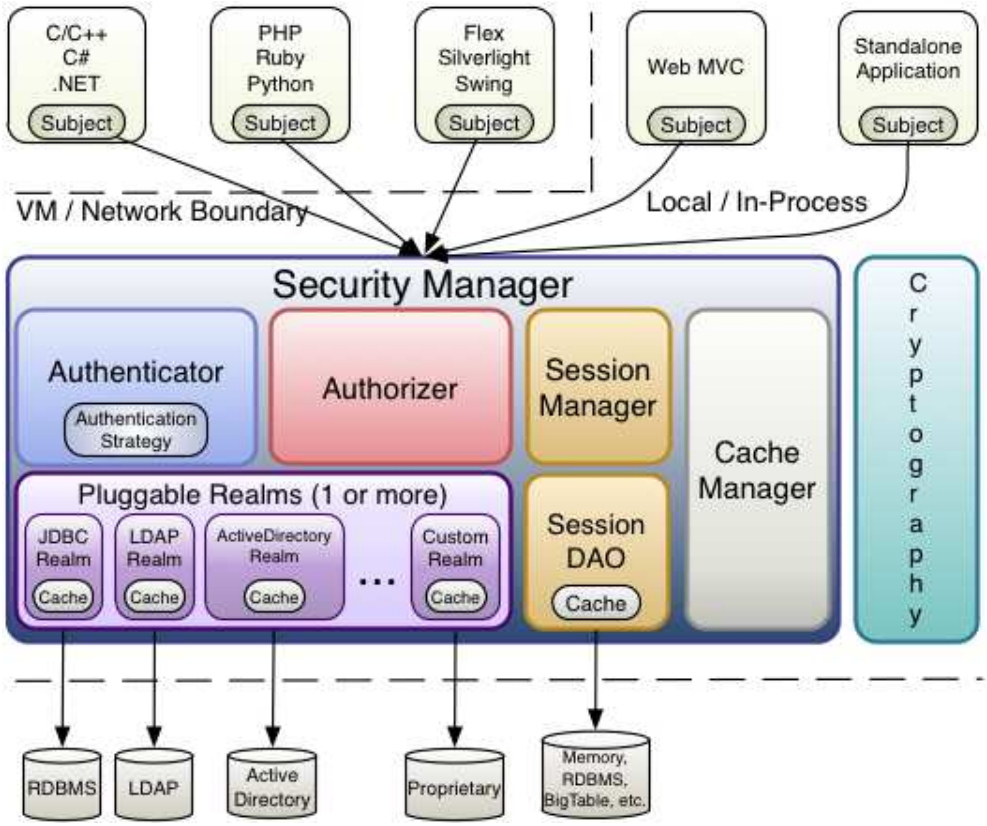


Abbildung 41: Architektur des Frameworks Apache Shiro, entnommen aus [Fou16]

Das Einloggen wird in Abbildung 41 gezeigt. Nach dem Klicken des Nutzers auf den Button zum Einloggen greift der Security Manager über den festgelegten Authenticator auf die Klasse JpaRealm zu, welche mittels der Daten aus der Datenbank die Benutzereingaben verifiziert. Sind diese Valide, gibt der Security Manager ein Subject-Objekt zurück, in der sowohl Informationen über den Nutzer enthalten sind, als auch Daten zum Validieren von möglichen Benutzerrechten.
Letzteres erfolgt in Apache Shiro über den Authorizer. Beim Zugriff auf eine Quelle, deren Zugriff eingeschränkt sein soll, wird über ein Data Access Object die Information über

die Art des Zugriffs übermittelt. In der aktuellen Form des Projekts erfolgt dies über die definierte Klasse AccessPermission. Beim Instantiieren wird, neben dem Nutzer, welcher auf ein Objekt zugreifen will, die Quelle übergeben, auf die zugegriffen werden soll, z.B. ein User-Objekt zum Einsehen des Profils eines Nutzers, als auch die Art des Zugriffs (z.B. lesend). Eine beispielhafte Validierung ist im Quelltext 29 einzusehen. worin geprüft wird, ob der eingeloggt Nutzer das Recht hat, das Profil eines anderen Nutzers zu editieren.

```
1  Subject subject = (Subject) VaadinSession.getCurrent().
       getAttribute(WelcomeUI.USER);
2  User user = (User) subject.getPrincipal();
3
4  AccessType type = AccessType.Edit;
5  AccessSource source = AccessSource.UserProfile;
6  User userToEdit = new User();
7
8  AccessPermission accessDao = new AccessPermission(user,source
       ,type,userToEdit);
9
10 boolean editPermission = subject.isPermitted(accessDao);
```

Quelltext 29: Beispielhafte Validierung des eingeloggten Nutzers auf das Recht, ein spezifisches Userprofil ändern zu dürfen

Das Validieren erfolgt ebenfalls über den in der Konfiguration festgelegten Realm, aktuell über die Klasse JpaRealm. Es wird geprüft, ob dem Nutzer entsprechende Rollen für die Operationen zugewiesen wurde. Diese sind, zusammen mit den einzelnen Berechtigungen einer Rolle, in der in Kapitel 5.4 beschriebenen Datenbank hinterlegt. Mittels des übergebenen Objektes wird zudem überprüft, ob Sonderrollen existieren, z.B. die Eigentümerrolle. Das Ergebnis wird in ein weiteres Data Access Object abgelegt, sodass dieses über den Cache Manager zwischengespeichert werden kann. Auf diese Weise können zukünftige Rechteabfragen effizienter ablaufen.

6.2.4. Profil

Alle Anwender bei IMPACT besitzen eine Nutzerprofilseite. Die Nutzerprofilseite verfügt dabei über persönliche Informationen, die den Nutzer identifizieren. Diese können dabei von dem jeweiligen Nutzer angesehen und bearbeitet werden.

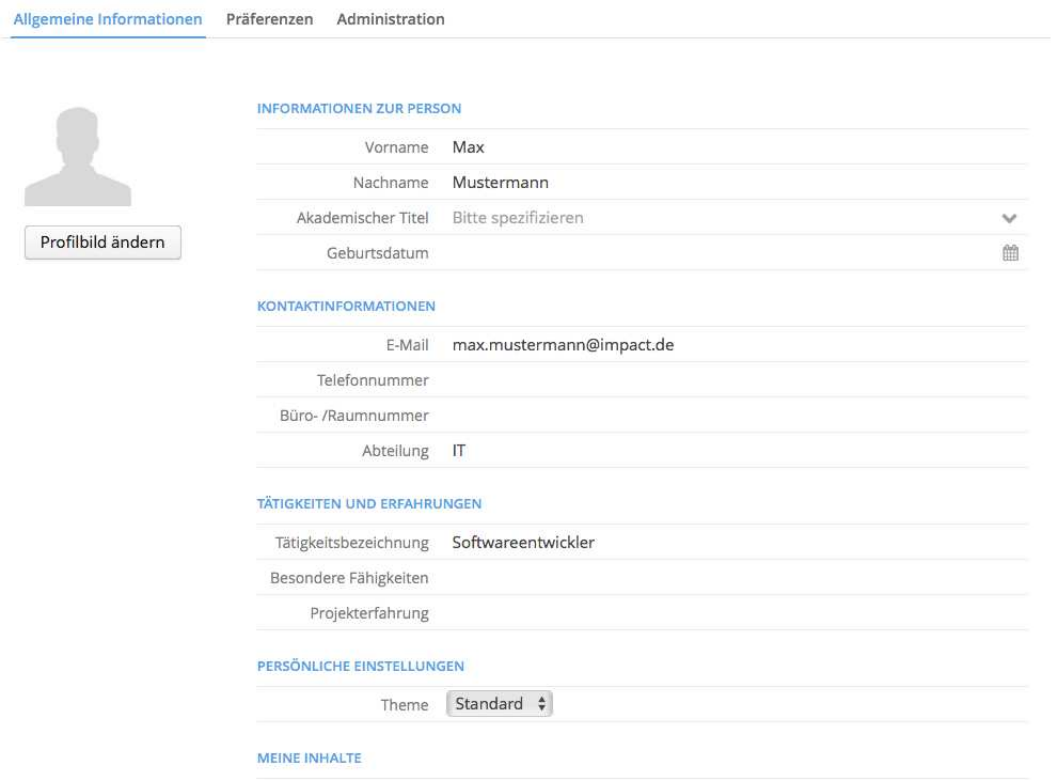


Abbildung 42: Ansicht eines Nutzerprofils

Oberfläche

Das Nutzerprofil lässt sich über die Lobby nach dem erfolgreichen Login über die Navigation erreicht werden. Dieses wird dabei in die Bereiche *Allgemeine Informationen*, *Präferenzen* sowie *Administration*, die in Form von Tabs unterteilt werden. Die Startseite der Bearbeitungssicht stellt den Bereich der Allgemeinen Informationen dar. Diese unterteilen sich jeweils in die Bereiche *Informationen zu der Person*, *Kontaktinformationen*, *Tätigkeiten und Erfahrungen*, *Persönliche Einstellungen* sowie *Meine Inhalte*. Anhand der Abbildung 42 lässt sich erkennen, dass das Profil Ähnlichkeit mit einem Kontaktformular aufgebaut ist. Dies wurde realisiert, damit der Anwender einerseits eine übersichtliche Darstellung der Nutzerinformation erhält und andererseits diese auf einfache Art und Weise bearbeiten kann. Genauere Informationen hinsichtlich der Informationen und den Möglichkeiten können im Benutzerhandbuch entnommen werden.

Technische Umsetzung

Bei der Anzeige des Nutzerprofils werden sämtliche, zuvor eingetragenen Informationen, entsprechend aufbereitet und dargestellt. Eine Visualisierung der einzelnen Felder mit den dazugehörigen Informationen, wie es im Folgenden beschrieben ist, kann der Abbildung 57 entnommen werden.

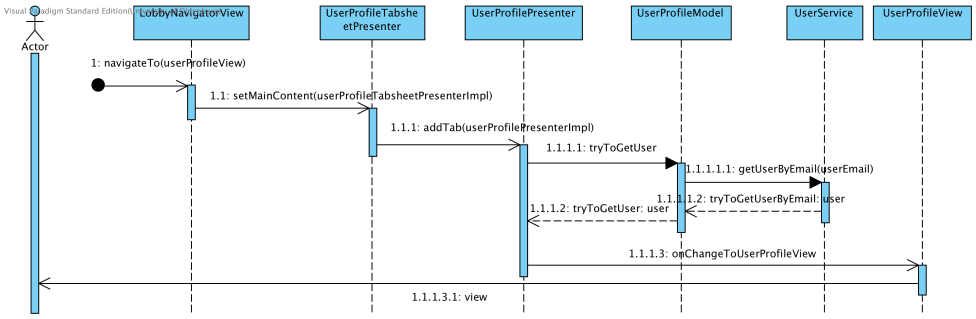


Abbildung 43: Sequenzdiagramm: Nutzerprofil anzeigen

Die Navigation zum Nutzerprofil erfolgt über die `LobbyNavigatorView`. An dieser Stelle wird zunächst das `UserProfileTabsheet` aufgerufen, da sich das Nutzerprofil als ein Tab in diesem befindet. Der zweite und der dritte Tab realisieren die Präferenzen sowie die Administration. Hinzugefügt werden diese über die `addTab()` Methode. Anschließend wird das Nutzerprofil mit Inhalten gefüllt, dazu ruft der Presenter über die Funktion `tryToGetUser(newUserData)` die entsprechenden Informationen aus der Datenbank ab. Für diesen Schritt wird die entsprechende `userID` benötigt, wobei der UserProfilePresenter auf eine entsprechende Funktion im ChallengeModel zurückgreift. Dieses nutzt dazu eine Methode des UserService, welche im DatabaseService implementiert ist. Dabei wird ein Manager genutzt, um den User anhand der userID aus der Datenbank zu lesen und liefert sie zurück. Mit den erhaltenen Informationen befüllt und aktualisiert der Presenter unter Einsatz der Methode `updateUser(newUserData)` die Felder der View mit ihren Inhalten. Die Grundlage der technische Realisierung für das Bearbeiten und somit das Aktualisieren des Profils bildet die *editUser(User newUserData)*, die im DataBaseService implementiert wird. Dabei wird zunächst das User Objekt aus der Datenbank über die userID ermittelt. Anschließend werden die einzelnen Informationen verwaltet, falls an dieser Stelle Änderungen vorgenommen wurden. Im Hintergrund sorgt die *updateBean* Methode dafür, dass die entsprechenden Feldes des Nutzerprofils mit dem User Objekt synchronisiert werden. Diese Situation wird in dem folgenden Sequenzdiagramm in übersichtlicher Form erläutert.

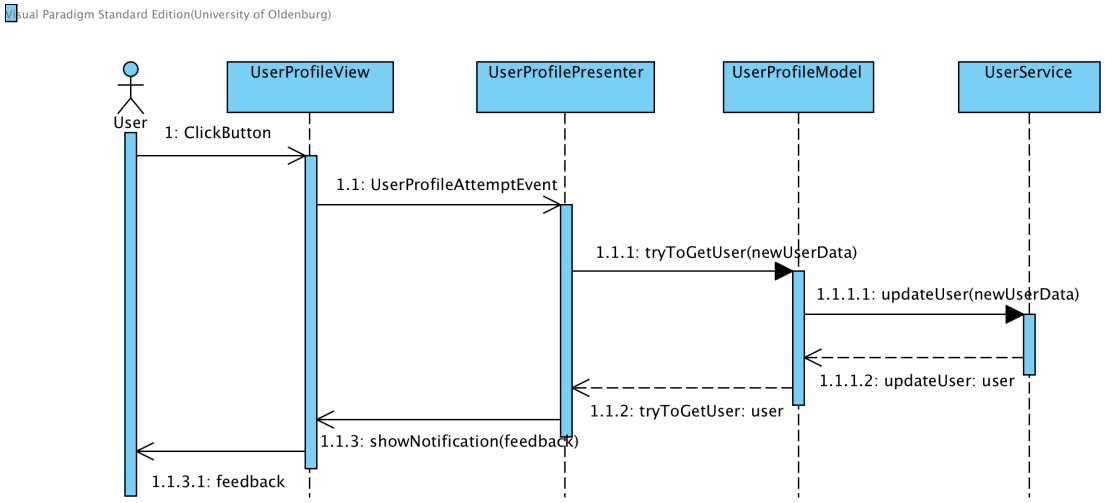


Abbildung 44: Sequenzdiagramm: Nutzerprofil bearbeiten

6.2.5. Übersichten

Die Übersichten von allen in dem System vorhandenen Challenges, Projekten, Business Cases und Nutzern sind über die Menüleiste zu erreichen (vgl. Abbildung 45). Dabei ist der generelle Ablauf zum Aufruf der einzelnen Ansichten nahezu identisch und unterscheidet sich lediglich in den Aufrufen der Klassen. Es ist zu erwähnen, dass die Business Case-Gesamtübersicht nur sichtbar ist, wenn man das Recht dazu besitzt Business Cases einzusehen. Eine weitere Besonderheit ist in der *Nutzerübersicht* zu finden. Es ist möglich diese Tabelle zu filtern durch den Einsatz einer Filtertabelle. Im weiteren Verlauf wird der Aufruf und die Anzeige der Challengeübersicht genauer betrachtet. Die Übersichten der *archivierten Objekte*, der *Top Ten* und *Meine Inhalte* werden einzeln betrachtet, da diese sich von den anderen Übersichten unterscheiden.
Die Übersichten sind über die Menüleiste zu erreichen. Über der Menüleiste ist das Profilbild und der Nutzername zu sehen.

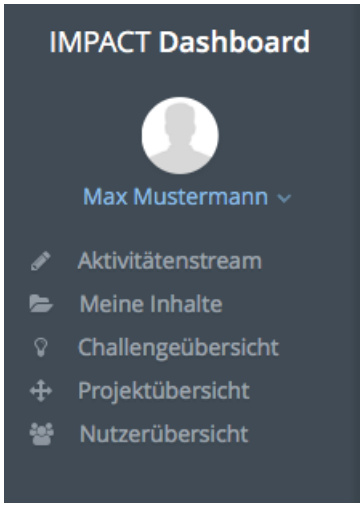


Abbildung 45: Menüleiste zur Navigation

Oberfläche

Die Challengeübersicht erfolgt in tabellarischer Form. Es ist möglich nach Aktualität oder ähnliches zu sortieren. Angezeigt werden die wichtigsten Informationen zu einer Challenge, dazu zählen Bewertungen, Titel, Beschreibung und Autor (vgl. Abbildung 46). Über die Elemente der Tabelle gelangt man zu den Detailansichten der Challenges.

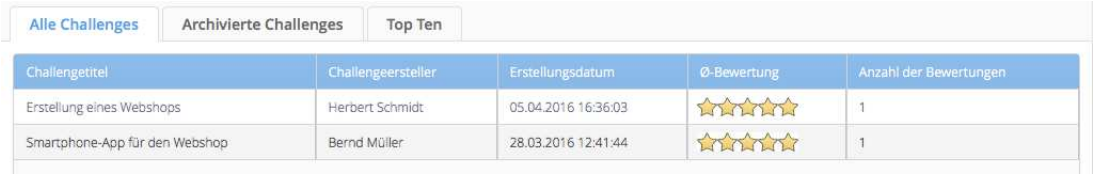


Abbildung 46: Übersicht der vorhandenen Chalenges

Technische Umsetzung

Durch das Betätigen des Menübuttons *Challengeübersicht* wird das Event `ChangeToChallengeViewEvent` aus der `DashoboardMenuView` gesendet und über die `LobbyNavigatorView` zur Gesamtübersicht geleitet. In der `LobbyNavigatorView` wird die Methode `show` des `ChallengeStageOverviewPresenter` aufgerufen und über die Methode `tryToGetAllChallenges` des *Models*, nach alle vorhandenen Challenges in dem System gefragt und anschließend als Liste zurück gegeben. Anschließend wird in der `ChallengeStageOverviewView` ein *BeanContainer* für die Challenges erstellt und anschließend mit den, in der Liste vorhandenen, Challenges befüllt. Um die Challengeübersichtstabelle mit den Challenges zu füllen, muss der *BeanContainer* `bc` als Datenquelle gesetzt werden (vgl. Quellcode 30).

```
153 public void updateChallengeTable(List<Challenge>
        challengeList) {
154
155         BeanContainer<Integer, Challenge> bc = new
                BeanContainer<Integer, Challenge>(Challenge.class)
                ;
156         bc.setBeanIdProperty(Challenge.ITEM_PROPERTY_ID);
157         bc.addAll(challengeList);
158
159         allChallengeTable.setContainerDataSource(bc);
160
161 }
```

Quelltext 30: Methode zum Befüllen der Challengeübersichttabelle

Anschließend wird die Ansicht mit der gefüllten Tabelle veröffentlicht und alle Challenges sind sichtbar und auswählbar (vgl. Abbildung 47).

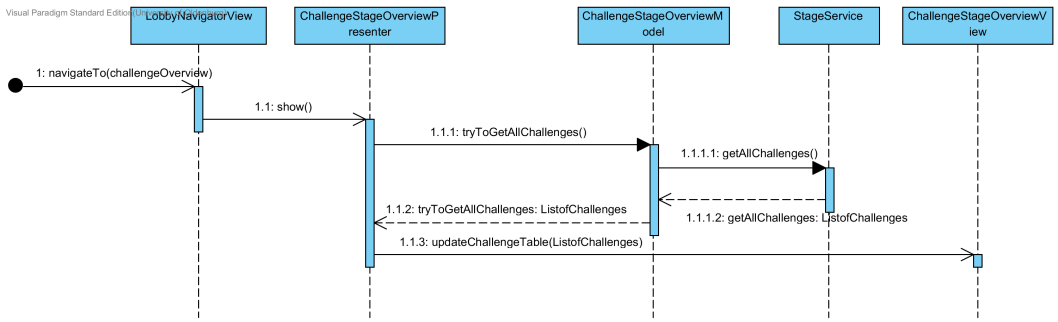


Abbildung 47: Sequenzdiagramm: Challengegesamtübersicht

Meine Inhalte

Die Übersicht *Meine Inhalte* besitzt Besonderheit, dass nur die Elemente angezeigt werden, die ein Benutzer erstellt hat oder bearbeiten darf. Dazu gehören Bewertungen, Kommentare, Lösungsvorschläge, Challenges und Projekte. Die gesamte Anzeige wird mit Hilfe eines Tabsheets umgesetzt. Jedes dieser Elemente besitzt seinen eigenen Tab. Durch das Wechseln des Tabs werden die jeweiligen Ansichten geladen. darüber hinaus, können die verschiedenen Spalten der Tabellen z.B. nach der Aktualität sortiert werden.

Abbildung 48: Übersicht: Meine Inhalte

Durch das Betätigen des Menübuttons *Meine Inhalte* wird das Event `ChangeToMyContentViewEvent` aus der `DashboardmenuView` gesendet. Die `LobbyNavigatorView` navigiert anschließend zur Gesamtübersicht der eigenen Inhalten weiter. In der `LobbyNavigatorView` wird die Methode `show` des `MyContentPresenter` aufgerufen. In dieser wird über die Methode `tryToGetAbsractItemToUser` des Models, nach allen AbstractItems gesucht, in denen der entsprechende Anwender als Ersteller eingetragen ist. Über den StageService wird diese Abfrage umgesetzt. Innerhalb der `aiList` sind alle Challenges, Lösungsvorschläge und Projekte vorhanden, in denen der Anwender als Ersteller eingetragen ist (vgl. Quellcode 31). Ebenfalls wird über die Methode `tryToGetParticipatedProjectsToUser` nach allen Projekten gesucht in denen der Anwender ein Teammitglied ist. Die beiden Ergebnisse werden in Form von Listen bereitgestellt. Kommentare und Bewertungen werden nach dem selben Prinzip gesucht und bereitgestellt. Die entsprechenden Listen werden in der `MyContentView` zu den passenden BeanContainern hinzugefügt, welche wiederum als Datenquellen in den Tabellen gesetzt werden (vgl. Abbildung 49).
Um die Projekttabelle zu befüllen, müssen die beiden Listen der gesammelten Projekte abgeglichen werden, um doppelte Einträge zu vermeiden, denn ein Projektersteller kann auch ein Teammitglied sein. Somit würde er in beiden Listen auftauchen. Die Projekte, in welchen der Anwender nicht als Teammitglied eingetragen ist, jedoch als Ersteller werden zu der Liste `projectList` hinzugefügt (vgl. Quellcode 31).

```
404 private void populateProjectTable(List<AbstractItem> aiList,
405             List<Project> pList) {
406
407         List<Project> projectList = new ArrayList();
408         for (AbstractItem ai : aiList) {
409             if (ai instanceof Project && !pList.contains((
                    Project) ai)) {
410
411                 projectList.add((Project) ai);
412             }}}
```

Quelltext 31: Vergleichen der Listen

Darüber hinaus werden alle weiteren Projekte, die in der Liste `pList` der Liste `projectList` hinzugefügt. Die Liste `pList` beinhaltet alle Projekte in denen der Anwender als Projektmitglied eingetragen ist.

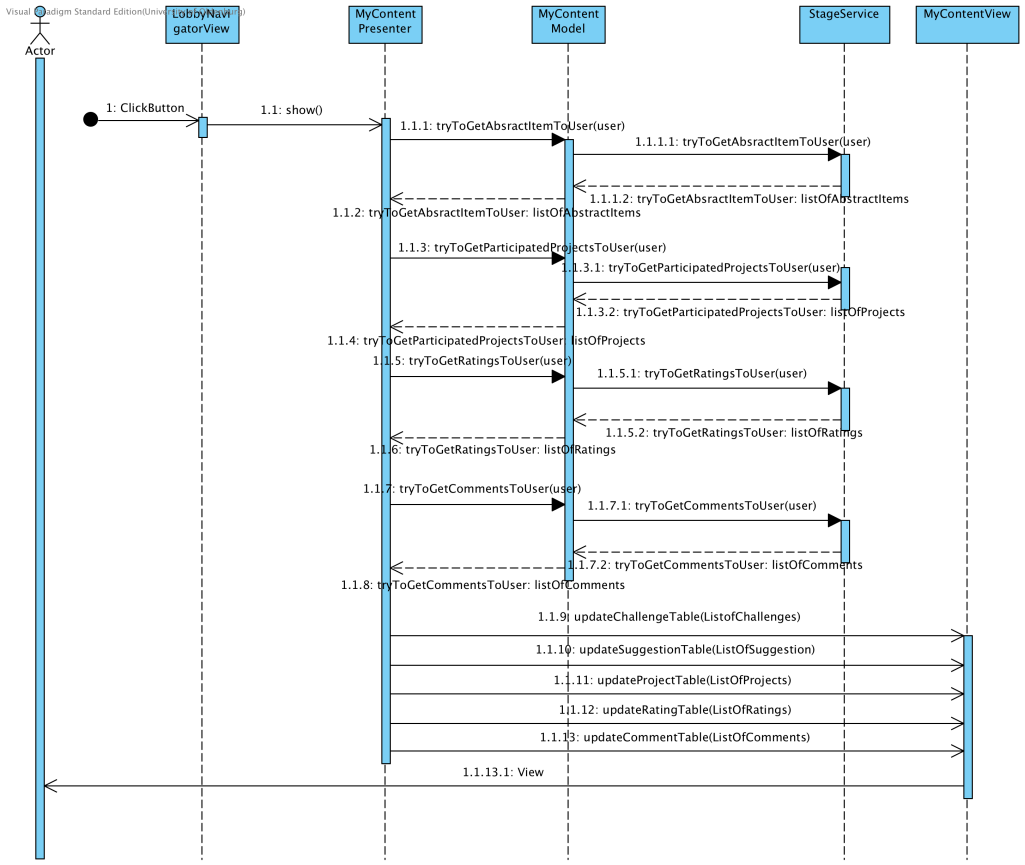


Abbildung 49: Sequenzdiagramm: Meine Inhalte-Übersicht

Übersichten über archivierte Objekte

Übersichten über archivierte Objekte werden in der *Challenge Übersicht* und der *Projekt Übersicht* angezeigt. Die Übersicht der archivierten Challenges wird unter dem Menüeintrag *Challenge Übersicht* im *Tab archivierte Challenges* angezeigt. In der *Projekt Übersicht* existiert die gleiche Übersicht für das Anzeigen von archivierten Projekten.
Die Übersichten werden in Form einer Tabelle angezeigt. Durch das Klicken auf ein Item werden entsprechende Details zum Item angezeigt, indem sich eine gesperrte *View* öffnet. Bei Klick auf den Tab *Archivierte Challenges* wird das Event `MyArchiveChallengeTableEvent` geworfen. Dieses wird im `ChallengeStageOverviewPresenter` abgefangen. Dort wird eine `List` mit allen archivierten Challenges erstellt, die über die Methode `tryToGetArchivedChallenges()` des `Models`

gefüllt wird. In der daraufhin aufgerufenen Methode `getAllArchivedChallenges()` wird nach dem Attribut `archived` der Challenge oder des Projekts gefiltert. Wenn das Attribut auf `true` gesetzt ist, wird die Challenge oder das Projekt in der Tabelle aufgelistet.

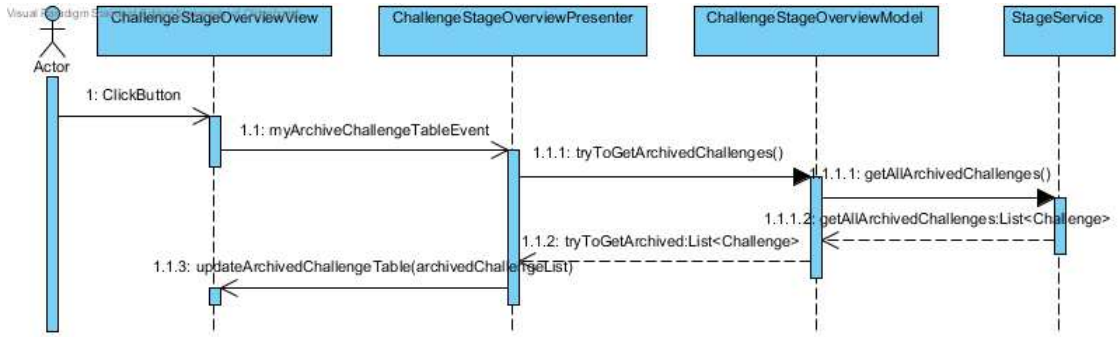


Abbildung 50: Sequenzdiagramm: Archivierte Challenges Übersicht

Top 10

Für Challenges wird in der *Challenge Übersicht* eine Top10 Liste im *Tab Top10* angezeigt. Die zehn am höchsten bewerteten Challenges werden hier absteigend nach der Bewertung angezeigt.

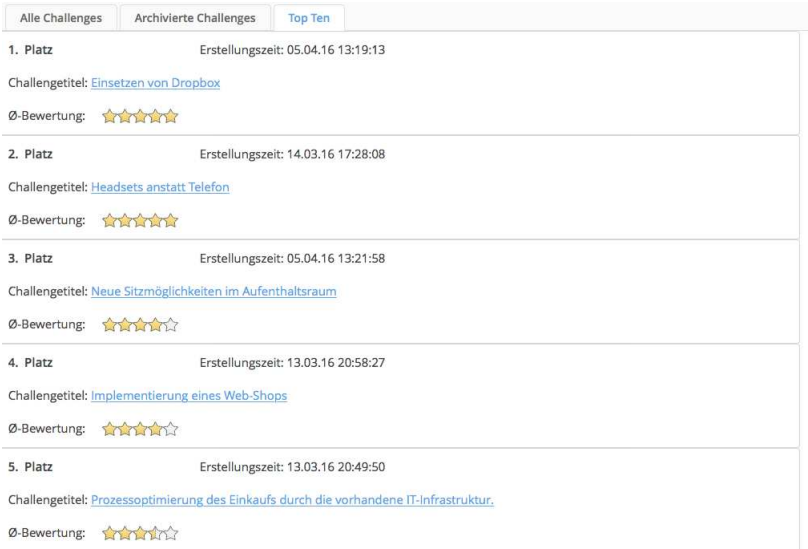


Abbildung 51: Top 10 Übersicht

Beim Klick auf *Top10* wird das Event `CreateTopTenListEvent` in der `ChallengeStageOverviewView` geworfen. Dieses wird von `ChallengeStageOverviewPresenter` abgefangen. Hier wird über den `TopTenPresenter`

die Art und die ChallengeId für die `TopTenView` gesetzt und ein `VerticalLayout` generiert, dass der `ChallengeStageOverviewView` übergeben wird.
Die Anzeige der Informationen zu einer Challenge erfolgt mit einer `TopTenComponent`. Diese werden mit der Methode `addTopTenElements( List<TopTenEvent> events)` der `TopTenView` in ein `VerticalLayout` im `TopTenViewDesign` hinzugefügt (vgl. Quellcode 32). Bei der Erstellung von *TopTenComponent*s werden zusätzlich *Listener* an die jeweilige Komponente registriert, damit bei Klick auf eine Komponente die Challenge Details angezeigt werden.

```
94          int i = events.size();
95          int x = 10;
96          while (x > 0 && i > 0) {
97              TopTenComponent p = new TopTenComponent(events.
                    get(i));
98              this.view.getStream().addComponent(p);
99              registerListeners(j);
100             i--;
101             x--;
102         }
```

Quelltext 32: Hinzufügen von Top10-Komponenten

Die Liste mit `TopTenEvents` wird über den `StageService` mit der Methode `getTopTenList( int AbstractItemId, int art)` erstellt (vgl. Quellcode 33). Ein `TopTenEvent` enthält das `AbstractItem`, die Art des Items und die Id des Items. Art und Id sind von Bedeutung, da die Top10 Liste auch für Lösungsvorschläge erstellt werden könnte.

```
1419 List<AbstractItem> abstractItemContainer = new ArrayList<
        AbstractItem>( getAllChallenges());
1420 Collections.sort( abstractItemContainer,
1421     new Comparator<AbstractItem>() {
1422
1423         @Override
1424         public int compare( AbstractItem o1, AbstractItem o2)
                {
1425             Double val1;
1426             Double val2;
1427             if (o1.getTotalRating() == null) {
1428                 val1 = 0.0;
```

```
1429                } else {
1430                    val1 = o1.getTotalRating();
1431                }
1432                if (o2.getTotalRating() == null) {
1433                    val2 = 0.0;
1434                } else {
1435                    val2 = o2.getTotalRating();
1436                }
1437                return val1.compareTo(val2);
1438            }
1439    });
1440    for (int i = 0; i < abstractItemContainer.size(); i++) {
1441        list.add( new TopTenEvent( abstractItemContainer.get(i),
                art, 0));
1442    }
```

Quelltext 33: Abfragen der Top10-Listen

Es werden zunächst alle Challenges in eine `ArrayList` geladen. Anschließend werden diese mit Hilfe des `Comperators` sortiert. In der darauffolgenden `for`-Schleife werden dann die `TopTenEvents`, die von der oben genannten Methode `addTopTenElements( List<TopTenEvent> events)` verwendet werden, generiert.

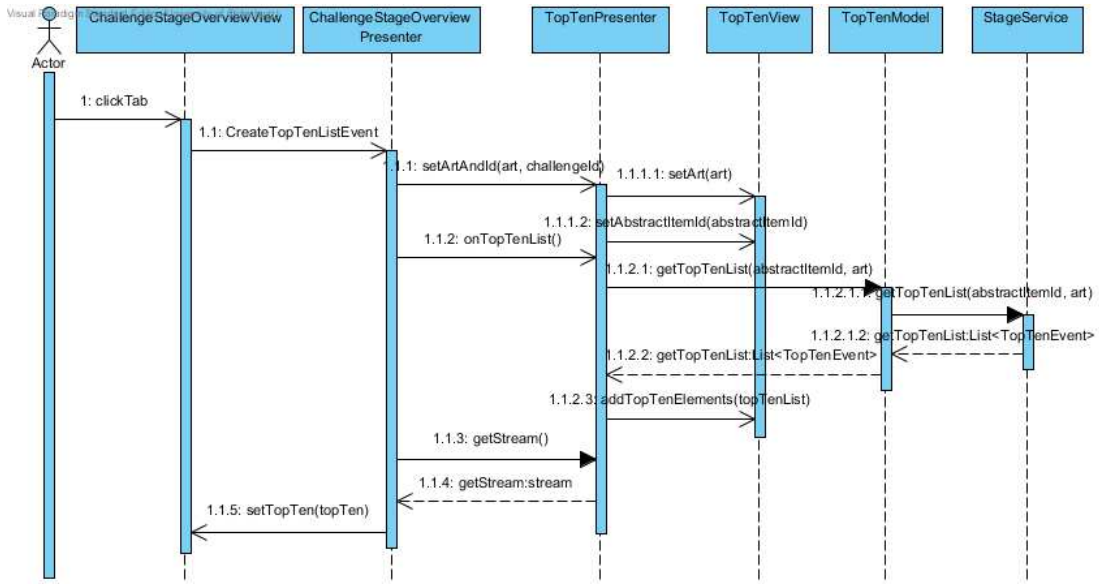


Abbildung 52: Sequenzdiagramm: Top 10 Übersicht

Nutzer-Übersicht

In diesem Abschnitt wird die Nutzer-Übersicht beschrieben. In der Nutzer-Übersicht werden alle Benutzer angezeigt, egal ob on- oder offline, die auf der Plattform registriert sind. Mit einem Klick auf die *Nutzerübersicht* in der Menüleiste wird diese Ansicht aufgerufen. In der Ansicht werden die wichtigsten öffentlichen Daten (Vorname, Nachname, E-Mail, Tätigkeitsbezeichnung und Abteilung) angezeigt. Neben der Sortierungsfunktion ist es auch hier möglich in den jeweiligen Spalten zu filtern. Durch einen Klick auf die gewünschte Zeile gelangt man zu dem vollständigen Profil des Benutzers.

6.3. Challenge

Eine Challenge ist die schriftliche Erfassung eines Problems oder einer Idee, welche von einem Mitarbeiter angelegt wird. Bei der Implementierung kann zwischen drei Kernfunktionalitäten unterschieden werden. Diese sind das Erstellen, Anzeigen und Editieren.

6.3.1. Challenge erstellen

Das erstellen einer Challenge wird in einer eigenen MVP-Triade ausgeführt. Die technische Umsetzung wird nach der Beschreibung der Oberfläche erläutert.

Oberfläche

Bevor die einzelnen sowie die gesamten Challenges angezeigt bzw. bearbeitet werden können, müssen diese im vorherigen Schritt zunächst erstellt werden. Eine Challenge kann dabei innerhalb der `LobbyUI` über den *Challenge erstellen*-Button angelegt werden. Im nächsten Schritt wird ein Popup-Fenster aufgerufen. Das Fenster rückt hierbei in den Vordergrund und der Hintergrund wird explizit dezent gehalten, damit der Nutzer auf der einen Seite nicht vom restlichen Inhalt der Webanwendung abgelenkt wird und auf der anderen Seite genau weiß, wie er eine Challenge zu erstellen hat. Folgende Abbildung demonstriert das Fenster, in dem eine Challenge eingetragen werden kann.
Hierbei ist auf den ersten Blick durch die Überschrift zu erkennen, dass der Nutzer direkt angesprochen wird. Dies soll dafür sorgen, dass die Motivation erhöht wird, die einzelnen Pflichtfelder auszufüllen. Diese setzen sich aus dem Titel, dem Ziel, den Hindernissen sowie der Beschreibung der Challenge zusammen. Die ersten drei Pflichtfelder werden dabei als *Textareas* und die detaillierte Beschreibung wird als *RichtTextArea* dargestellt. Die *Textarea* stellt ein normales Textfeld dar, wo Informationen eingetragen werden können. Die *RichtextArea* dient dazu, detailliertere Informationen einzutragen und bietet zusätzlich die Möglichkeit, Hervorhebungen und Formatierungen vorzunehmen. Um der Challenge zusätzliche Informationen und Aussagekraft zu geben, können bei Bedarf Anhänge sowie Schlagwörter in Form von Tags hinzugefügt werden. Im letzten Abschnitt der Challengeerstellung kann bei Bedarf ein Lösungsvorschlag im gleichen Schritt erstellt werden. Ein

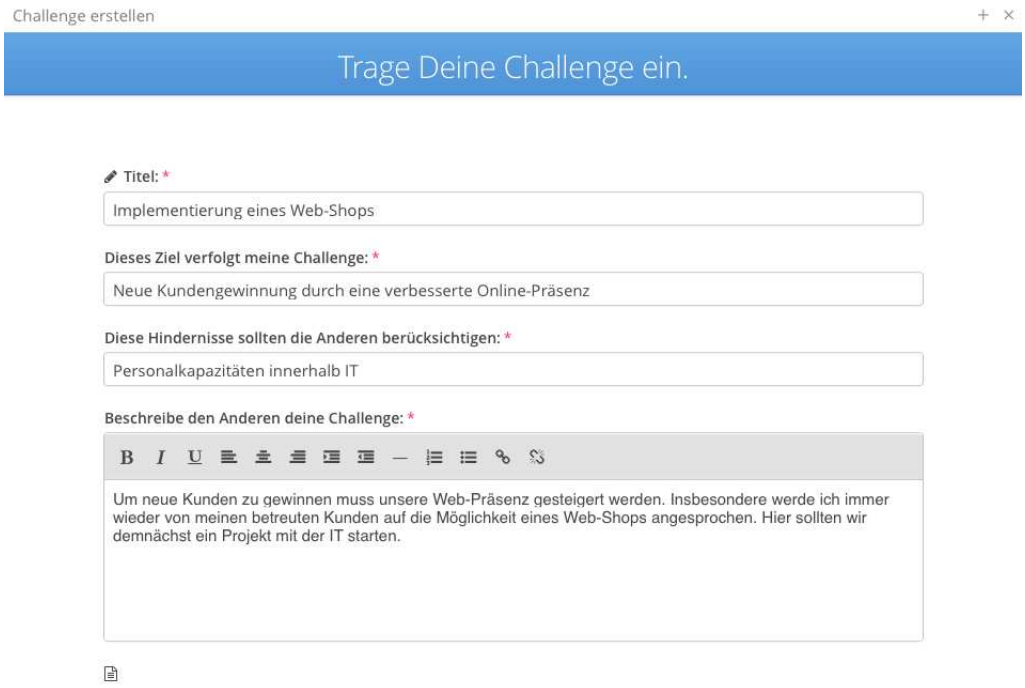


Abbildung 53: Anlegen einer Challenge

Lösungsvorschlag kann jedoch auch nachträglich hinzugefügt werden. Am Ende kann der Nutzer zusätzlich entscheiden, ob er die Challenge anonym abgeben, ob er diese speichern und erstellen oder den Prozess der Challengeerstellung abbrechen möchte. Dabei ist generell zwischen optionalen und Pflichtfeldern zu unterscheiden. Letztere müssen gesondert festgelegt werden. Eine detaillierte Umsetzung dessen ist in Abschnitt 3.2 beschrieben.

Technische Umsetzung

Geöffnet wird das Erstellen einer Challenge, sobald ein `ChangeToCreateChallengeViewEvent` gefeuert wird. Dieses wird in der `LobbyUI` abgefangen, woraufhin ein neues Popup-Fenster erzeugt und mit dem `CreateChallengePresenter` befüllt wird. Bei der Erstellung werden dem Nutzer konkret die Eingaben mittels *TextFields* und *RichTextAreas* bereitgestellt, wie innerhalb der Beschreibung der Oberfläche im vorherigen Abschnitt des Kapitels bereits aufgezeigt worden ist. Um Fehleingaben vorzubeugen wird zusätzlich der Button zum Speichern einer neuen Challenge erst aktiviert, sofern alle Pflichtfelder ausgefüllt sind. Dazu sind in der `CreateChallengeView` für die jeweiligen Pflichtfelder `ValueChangeListener` registriert. Diese führen nach jeder Veränderung des Zustandes die Methode `updateSaveAndPublishButton` aus (vgl. Quellcode 34).

```
212 public void updateSaveAndPublishButton() {
213     if (view.getChallengeTitleField().isValid()
214     && view.getChallengeDescriptionTextArea().isValid()) {
215         view.getSaveAndPublishButton().setEnabled(true);
216     } else {
217         view.getSaveAndPublishButton().setEnabled(false);}
218 }
```

Quelltext 34: Button zum Speichern der Challenge de-/aktivieren

Diese überprüft, ob alle Pflichtfelder mit Inhalt befüllt sind. Ist dies der Fall, so wird der Button zum Speichern und Veröffentlichen aktiviert, andernfalls wird er deaktiviert. Analog zu dieser Funktion existiert eine Methode zum deaktivieren des Speicher-Buttons für den Fall, dass es dem Nutzer ermöglicht wird, die Challenge zwischenzuspeichern, um sie zu einem späteren Zeitpunkt fertigstellen zu können. Diese Methode erlaubt es andere Bedingungen an den Button zum Zwischenspeichern zu definieren.
Neben dem Eintragen von Informationen ist es dem Anwender möglich, direkt einen Lösungsvorschlag zu seiner erstellten Challenge einzureichen. Dazu kann er in einer *RadioBox* wählen, ob er schon einen Ansatz besitzt oder nicht. Ist dies der Fall, werden bei Auswahl des entsprechenden Feldes weitere Textfelder für den Vorschlag angezeigt. Hierzu ist ebenfalls ein `ValueChangeListener` auf die *RadioBox* gesetzt, welcher die zusätzlichen Informationen sichtbar macht bzw. ausgeblendet.
Sofern die notwendigen Felder durch den Anwender ausgefüllt sind, kann die Challenge erstellt werden. Dabei wird durch das Betätigen eines Speicher-Buttons ein `CreateChallengeAttemptEvent` gesendet, welches im `CreateChallengePresenter` abgefangen wird. Zur Erzeugung der Challenge werden nun die entsprechend eingetragenen Informationen den Feldern entnommen. Zusätzlich wird der Nutzer aus der Session gelesen, um ihn als Ersteller zuordnen zu können. Möchte dieser nicht genannt werden, so hat er die Möglichkeit, diese anonym abzugeben, was über eine `Checkbox` bei der Erstellung realisiert ist. Sind die Daten erfasst, wird die Methode `tryToCreateChallenge` der `CreateChallengeModel` aufgerufen. Von dort aus wird `saveNewChallenge` des `StageService` ausgeführt. Der `StageService` wird dabei als ein `Interface` realisiert, wobei die aufgerufene Funktion im `DatabaseService` implementiert ist. Die Abbildung 54 veranschaulicht diesen Vorgang.
Der Quellcode 35 stellt die Umsetzung dieser Funktion dar. Dazu wird zunächst anhand der übergebenen Parameter eine neue Challenge erzeugt. Anschließend wird diese durch den `manager` persistiert.

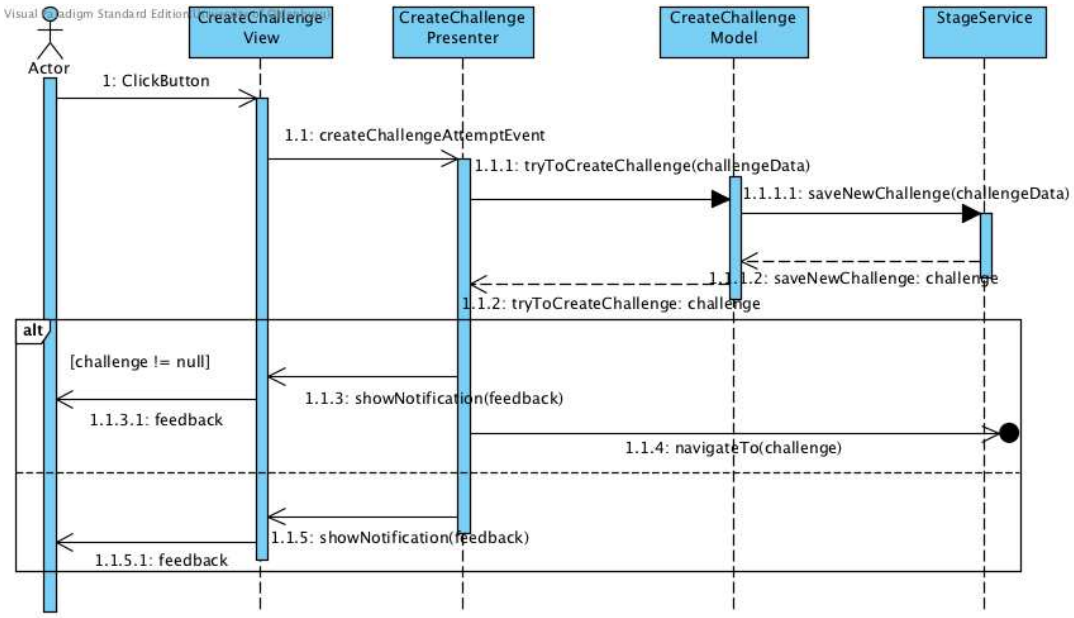


Abbildung 54: Sequenzdiagramm: Challenge erstellen

```
404     Challenge challenge = new Challenge(title, description,
            goal, barrier, user, createdAt, publishedAt, anonymous
            );
405     manager.getTransaction().begin();
406     manager.persist(challenge);
```

Quelltext 35: Funktion zum Erzeugen einer Challenge

Sofern vom Anwender Tags dem Objekt beigefügt sind, werden diese der neuen Challenge in der Datenbank angehängt (vgl. Quellcode 36). Sollten diese dort noch nicht existieren, werden sie ebenfalls durch das Persistieren vom Manager erzeugt. Analog dazu wird mit Anhängen verfahren. Abgeschlossen wird die Aktion durch ein `commit` des Managers und dem Zurückliefern des Challenge Objektes. Dieses wird über das *Model* an den *Presenter* weitergereicht. Sofern es `null` ist, wird dem Anwender ein negatives Feedback in Form einer `notification` gegeben. Andernfalls erhält dieser eine positive Benachrichtigung und wird zu seiner neu angelegten Challenge geleitet. Dies geschieht über die Navigation durch den `LobbyNavigator` (vgl. Abschnitt 3.2). Informationen zur Realisierung der Darstellung kann dem Kapitel 6.3.2 entnommen werden.

```
412 if (tags != null && !tags.isEmpty()) {
413     LinkedHashSet<Tag> tagSet = new LinkedHashSet<Tag>();
414     Iterator<Tag> it = tags.iterator();
415     while (it.hasNext()) {
416         Tag t = it.next();
417         Tag foundTag = manager.find(Tag.class, t.getTagName()
                );
418         if (foundTag != null) {
419             foundTag.getAbstractItems().add(challenge);
420             tagSet.add(foundTag);
421         } else {
422             Set<AbstractItem> as = new HashSet<>();
423             as.add(challenge);
424             t.setAbstractItems(as);
425             manager.persist(t);
426             tagSet.add(t);  }}
427     challenge.setTags(tagSet);  }
```

Quelltext 36: Funktion zum Speichern eines Tags

6.3.2. Challenge anzeigen

Die Funktionalität des Anzeigen ermöglicht allen Anwendern das lesen der detaillierten Informationen zur Challenge. Im Folgenden wird die Oberfläche und die technische Umsetzung beschrieben.

Oberfläche

Nachdem eine Challenge erstellt wurde, wird anschließend die detaillierte Sicht dieser Challenge geöffnet. Der Benutzer bekommt hierbei einen Überblick über die einzelnen Komponenten und Felder, die zu einer einzelnen Challenge gehören. Die Übersicht wird dabei in Form eines *Tabsheets* dargestellt, um eine effiziente Navigation zwischen der Challenge und dem dazugehörigen Lösungsvorschlag zu gewährleisten.
In Abbildung 57 lässt sich erkennen, dass zu Beginn der Titel sowie der Name des Erstellers eine Challenge abgebildet werden, damit sich die Challenge direkt identifizieren lässt. Den nächsten Abschnitt stellt die Bewertung der Challenge dar. Der Anwender bekommt einen Überblick über die durchschnittliche Bewertung einer Challenge und kann diese anschließend bewerten. Darüber hinaus wird Auskunft über die zuvor angelegten Ziele sowie Hindernisse gegeben.

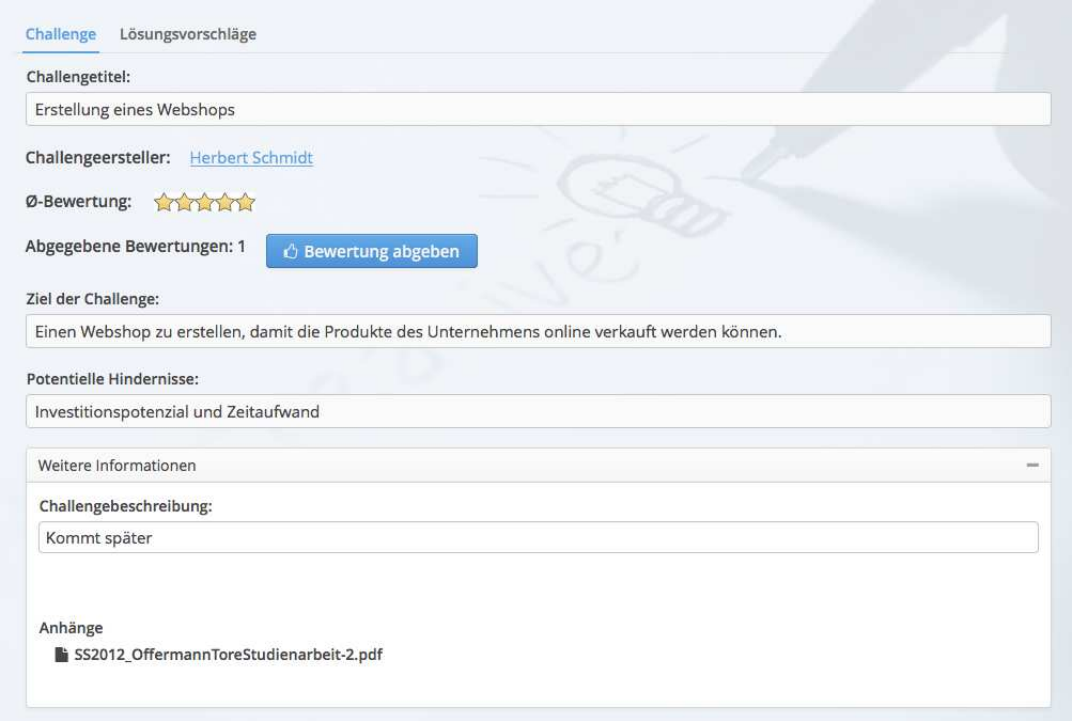


Abbildung 55: Anzeige einer Challenge

Weiterhin kommt hinzu, dass innerhalb der Anwendung durchaus viele Informationen auf einer Seite (detaillierte Beschreibungen etc.) dargestellt werden müssen. Dies kann schnell unübersichtlich werden und gestaltet das Finden der richtigen Information schwierig. Deshalb wurden sogenannte *Stackpanels* implementiert, sodass bewusst einige Informationen erst ausgeblendet werden und bei Bedarf ausgeklappt werden können. Die Abbildung 56 illustriert die *Stackpanels* innerhalb der Challenge: Im Bereich dieser Oberfläche werden *Stackpanels* eingesetzt, um die detaillierte Beschreibung, die jeweiligen Bewertungen sowie die Kommentare der einzelnen Challenge anzuzeigen. Um sich die Informationen anzuschauen, können die *StackPanels* wie zuvor beschrieben, eingeklappt und ausgeklappt werden. Bei den Kommentaren kann der Anwender nach dem ausklappen des *Panels* einen eigenen Kommentar zu der Challenge abgeben und diesen abspeichern. Im letzten Schritt kann zu der jeweiligen Challenge ein Lösungsvorschlag erstellt werden.

Technische Umsetzung

Bei der Anzeige der Challenge werden sämtliche, zuvor eingetragenen Informationen, für alle Anwender des Systems ansprechend aufbereitet und dargestellt. Eine Visualisierung des Befüllens der Felder, wie es im Folgenden beschrieben ist, kann der Abbildung 57 entnommen werden.

Abbildung 56: Stackpanel innerhalb der Challenge

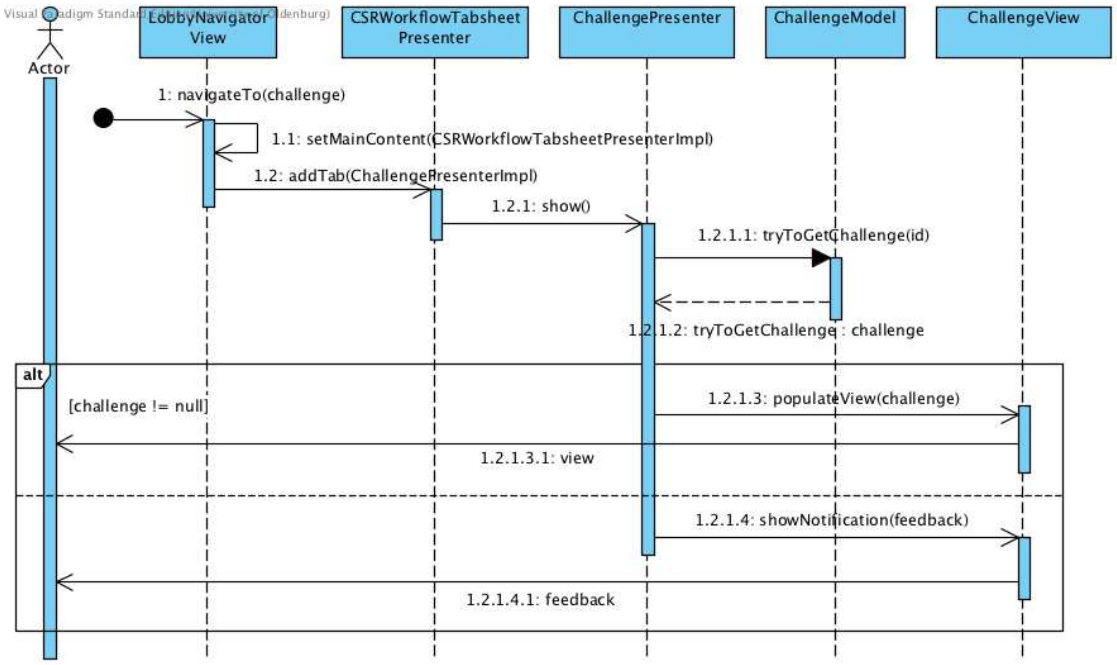


Abbildung 57: Sequenzdiagramm: Challenge anzeigen

Die Navigation zur Challenge erfolgt über die `LobbyNavigatorView`. An dieser Stelle wird zunächst das `CSRWorkflowTabsheet` aufgerufen, da sich die Challenge als ein `Tab` in diesem befindet. Der zweite `Tab` beinhaltet die zu einer Challenge gehörigen

Lösungsvorschläge. Hinzugefügt werden diese zusammengefasst über die `addTab` Methode. Anschließend wird die Challenge mit Inhalten gefüllt, dazu ruft der *Presenter* über die Funktion `tryToGetChallenge(id)` die entsprechenden Informationen aus der Datenbank ab. Für diesen Schritt wird die entsprechende `ChallengeId` benötigt, wobei der `ChallengePresenter` auf eine entsprechende Funktion im `ChallengeModel` zurückgreift. Dieses nutzt dazu eine Methode des `StageService`, welche im `DatabaseService` implementiert ist. Dieser nutzt den Manager, um die Challenge anhand ihrer id aus der Datenbank zu lesen und liefert sie zurück. Mit den erhaltenen Informationen befüllt der *Presenter* unter Einsatz der Methode `populateView(challenge)` die Felder der *View* mit ihren Inhalten.
Innerhalb der Challenge kann zwischen zwei Ansichten unterschieden werden. Die Standardsicht stellt dabei lediglich Informationen dar. Darüber hinaus ist es dem Anwender möglich diese zu bewerten (vgl. Kapitel 6.11)oder einen Kommentar zu verfassen (vgl. Abschnitt 6.10). Sofern der Nutzer über die entsprechenden Rechte verfügt (bspw. da er der Ersteller ist), wird ihm zusätzlich eine zweite Sicht, die des Editierens, bereitgestellt. Zu erreichen ist dieser Modus durch einen Klick auf den Button *Bearbeiten*, welcher einen ClickListener besitzt. Durch ihn wird die Methode `editChallenge` aufgerufen.
Neben bewerten und kommentieren kann der Nutzer innerhalb einer Challenge neue Lösungsvorschläge (vgl. 6.4) für diese erstellen. Sind genügend Informationen gesammelt, so kann durch das Betätigen des Buttons *Projekt starten* ein Projekt aus einer Challenge erstellt werden.

6.3.3. Challenge bearbeiten

Die Bearbeitung einer Challenge kann nur durch den Ersteller durchgeführt werden. Der Aufbau dieser Funktionalität wird im weiteren verlauf beschrieben.

Oberfläche

Neben dem Erstellen ist es dem Anwender möglich, seine getroffenen Angaben zu erweitern bzw. zu verändern. Dazu ist die Bearbeiten-Ansicht vorgesehen. Diese kann über einen Bearbeitungsbutton erreicht werden. Nachfolgende Abbildung stellt die Bearbeitungssicht der einzelnen Challenge dar: Die Kernfunktionalität ist es dem Anwender die Bearbeitung der Informationsfelder zu ermöglichen. Weiterhin werden die übrigen Funktionalitäten innerhalb der Challenge, wie das Kommentieren und Bewerten, ausgeblendet, um Fehler und Missverständnisse vorzubeugen. Zum Wechseln des Modus kann der Anwender abschließend seine Änderungen speichern.

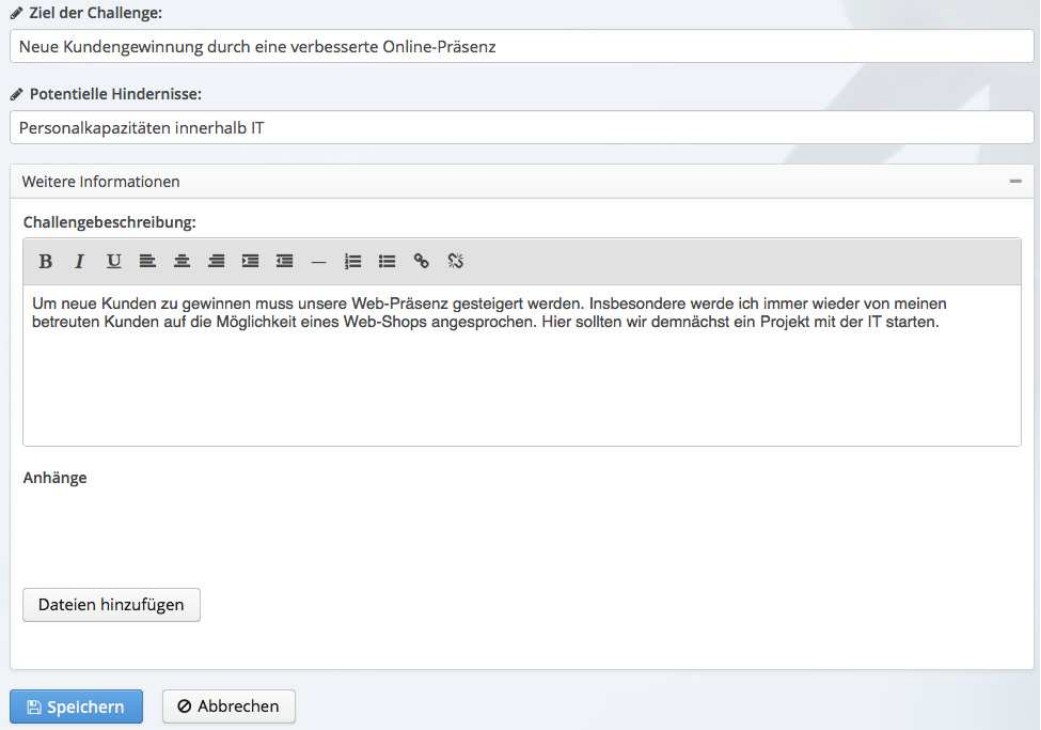


Abbildung 58: Bearbeitungsansicht der Challenge

Technische Umsetzung

Das Bearbeiten geschieht ähnlich zur Erstellung der Challenge (vgl. Absatz /ref-subsubsection:challengeerstellen) durch die Methode `editChallenge(challenge)` im `DatabaseService`. Hierbei wird zunächst das Challenge Objekt aus der Datenbank über die `Id` ermittelt. Anschließend werden Anhänge und Tags verwaltet, für den Fall, dass an dieser Stelle Änderungen vorgenommen wurden. Nach Anpassung der Informationen über die entsprechenden Methoden wird die Operation durch einen `commit` abgeschlossen und das Objekt mit den neuen Werten zurückgeliefert. Abschließend wird über `populateView(challenge)` die View mit den neuen Informationen befüllt und in den Anzeigemodus gewechselt, wobei die entsprechenden Felder ebenfalls in diesen Modus gesetzt werden. Für den Fall, dass der Nutzer sich dazu entschließt, seine Änderungen zu verwerfen, kann er dies durch die Betätigung des Abbrechen-Buttons durchführen. Dann werden die Felder mit den alten Informationen befüllt und es wird wieder auf den Anzeigemodus gewechselt. Diese beiden Alternativen sind im Sequenzdiagramm 59 erläutert.

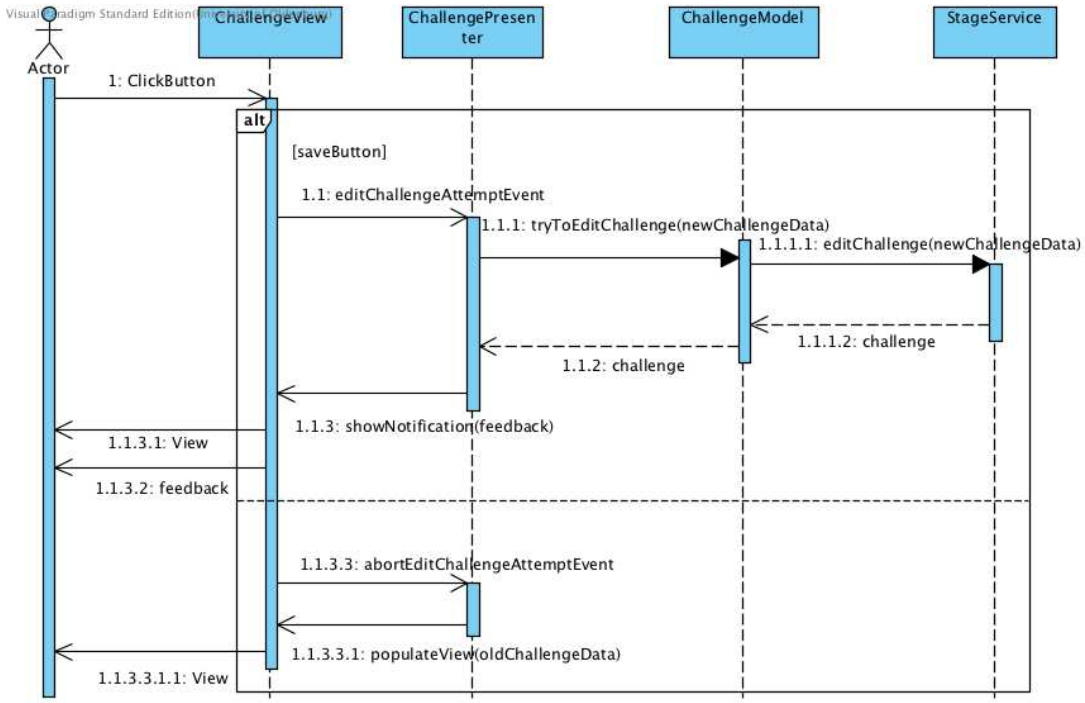


Abbildung 59: Sequenzdiagramm: Challenge bearbeiten

6.4. Lösungsvorschlag

Ein Lösungsvorschlag ist eine schriftliche Erfassung einer Lösung, die von einem Anwender verfasst wird und damit einen Vorschlag zu einer Challenge darstellt. Wie bereits in der Challenge beschrieben (vgl. Abschnitt 6.3) kann bei der Implementierung zwischen den Kernfunktionalitäten erstellen, anzeigen und editieren unterschieden werden. In den folgenden Abschnitten werden diese Funktionalitäten beschrieben.

6.4.1. Lösungsvorschlag erstellen

Das Erstellen eines Lösungsvorschlag wird in einer eigenen MVP-Triade ausgeführt. Die technische Umsetzung wird nach der allgemeinen Beschreibung der Oberfläche erläutert.

Oberfläche

Ein Lösungsvorschlag wird immer zu einer Challenge erstellt. Dies wird entweder unmittelbar beim Erstellen der Challenge durchgeführt (vgl. Abschnitt 6.3.1) oder zu einem späteren Zeitpunkt. Dazu muss diese Challenge über die im Kapitel 6.2.5 beschriebenen Übersichten aufgerufen werden. Über die *Challenge Übersicht* wird die Challenge aufgrufen und bietet dem Anwender die Möglichkeit einen Lösungsvorschlag einzureichen. Nach dem

Aktivieren des Buttons wird ein neues Popup-Fenster aufgerufen über die der Lösungsvorschlag angelegt wird. Hierbei muss der Anwender den Titel und die Beschreibung des Lösungsvorschlages angeben. Als zusätzliche Eingabemöglichkeiten stehen dem Anwender die Möglichkeit der Erstellung von Tags zur Verfügung. Der Lösungsvorschlag kann äquivalent zur Challenge anonym abgegeben werden. Dies soll unterschwellige Hemmnisse oder Vorbehalte der Anwender reduzieren. Folgende Abbildung demonstriert das Fenster, in dem ein Lösungsvorschlag eingetragen werden kann.

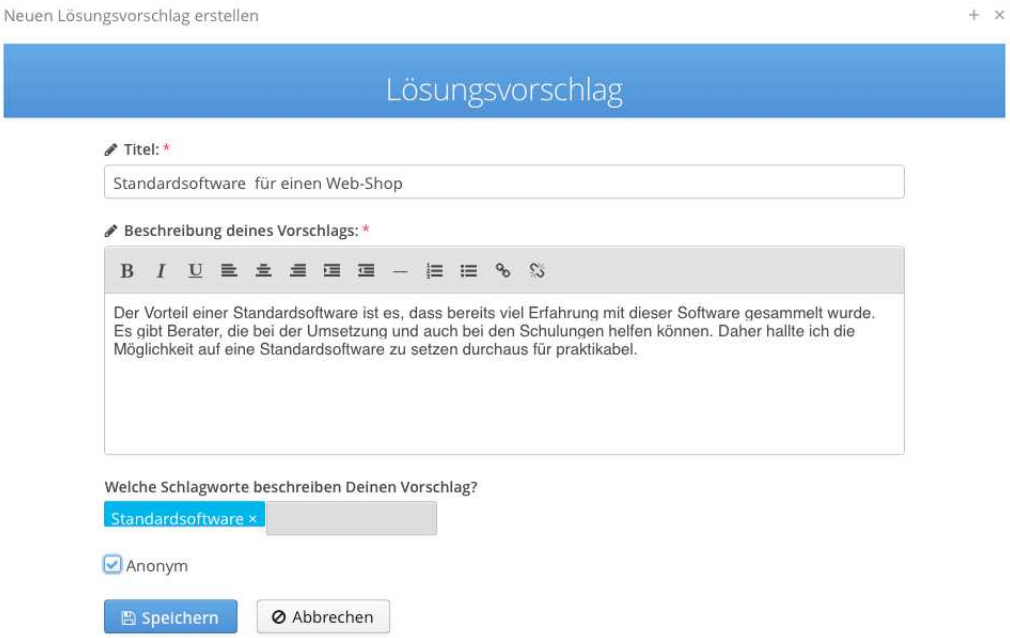


Abbildung 60: Anlegen eines Lösungsvorschlags

Technische Umsetzung

Das Erstellen eines Lösungsvorschlages beginnt mit dem Feuern des `CreateSuggestionViewEvent`. Dieses wird identisch, wie beim Erstellen der Challenge (vgl. Abschnitt 6.3.1) in der `LobbyUI` abgefangen. Um den Lösungsvorschlag der entsprechenden Challenge zuzuordnen wird die ChallengeID und der User über die Methode `tryToGetChallengeId` aus der Session ausgelesen (vgl. Quellcode 37). Anschließend wird ein neues Pop-up erzeugt, welches mit dem `CreateSuggestionPresenter` befüllt wird. Beim Erstellen des Lösungsvorschlages stehen dem Anwender ein `TextField` und eine `RichTextArea` zur Verfügung, welche als Pflichtfelder definiert wurden. Um Anwenderfehler vorzubeugen, wird der Button zum Speichern erst aktiviert, wenn alle Felder ausgefüllt sind. Um dies zu gewährleisten, wurden in der `CreateSuggestionView` für die jeweiligen Pflichtfelder `ValueChangeListener` registriert. Nach jeder Veränderung des Zustandes wird die Methode `updateSaveButton` ausgeführt, welche analog zur Methode

`updateSaveAndPublishButton` aufgebaut ist (vgl. Quellcode 34). Diese überprüft, ob alle Pflichtfelder mit Inhalt befüllt sind und aktiviert nach erfolgreicher Prüfung den Button zum speichern.

```
77 public void onSuggestionCreateAttempt(
78 @Observes CreateSuggestionAttemptEvent event) {
79 try {
80         challenge = model.tryToGetChallenge(event.challengeId
              );
81 } catch (Exception e) {
82         e.printStackTrace();
83 }
84         User user = null;
85         Date publishedAt = null;
86 try {
87         Subject subject = (Subject) VaadinSession.getCurrent
              ().getAttribute(WelcomeUI.USER);
88         user = (User) subject.getPrincipal();} }
```

Quelltext 37: Auslesen der ChallengeId aus der Session

Nachdem alle Pflichtfelder durch den Anwender befüllt sind, kann der Lösungsvorschlag erstellt werden. Dazu wird durch die Betätigung des Buttons ein `CreateSuggestionAttemptEvent` gesendet, welches im `CreateChallengesPresenter` abgefangen wird. Zur Erzeugung des Lösungsvorschlages werden die Eintragungen aus den Feldern entnommen. Sobald alle Daten erfasst wurden, wird die Methode `tryToCreateSuggestion` des `CreateSuggestionModel` aufgerufen. Nachfolgend wird die Methode `saveNewSuggestion` des `StageService` ausgeführt. Dieser wird als ein Interface realisiert, die ausgeführte Methode ist im `DatabaseService` realisiert. Die Abbildung 61 veranschaulicht diesen Vorgang.

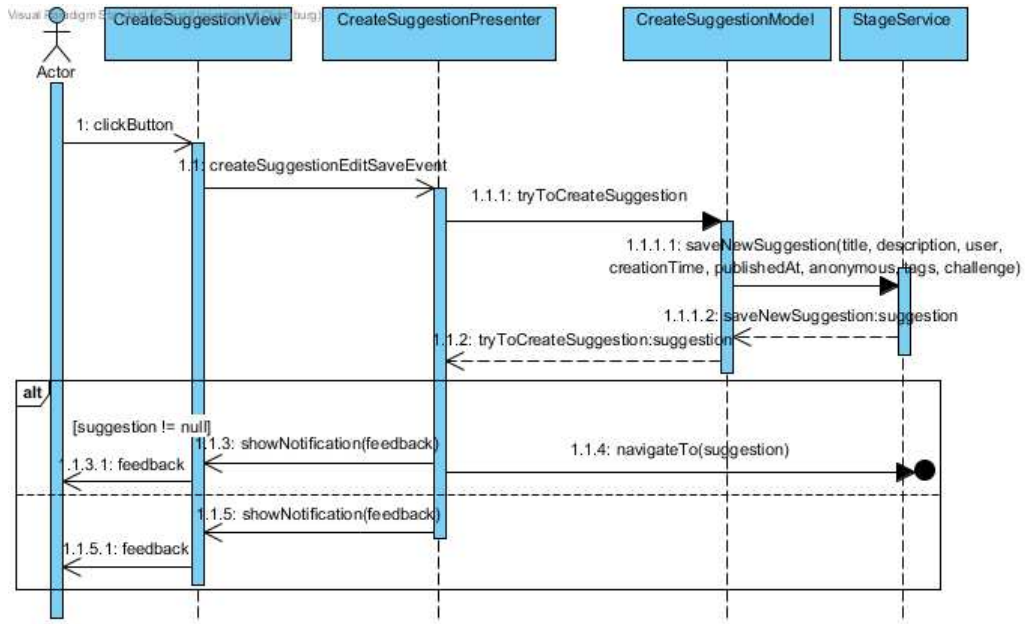


Abbildung 61: Sequenzdiagramm: Suggestion erstellen

6.4.2. Lösungsvorschlag anzeigen

Die Funktionalität des Anzeigens ermöglicht allen Anwendern das Lesen der detailierten Informationen zu einem Lösungsvorschlag. Im Folgenden wird die Oberfläche und die technische Umsetzung beschrieben.

Oberfläche

Nachdem ein Lösungsvorschlag erstellt wurde, wird dieser angezeigt. Der Benutzer bekommt hierbei einen Überblick über die einzelnen Komponenten und Felder, die zu einzelnen Lösungsvorschlägen gehören. In der selben Ansicht können auch andere Lösungsvorschläge durch Klick auf die jeweiligen Titel geöffnet werden. Die gesamte Übersicht der Lösungsvorschläge befindet sich auf der selben Ebene wie die Challengeanzeige, da durch einen Tab-Wechsel zwischen Challenge und Lösungsvorschlag gewechselt werden kann.

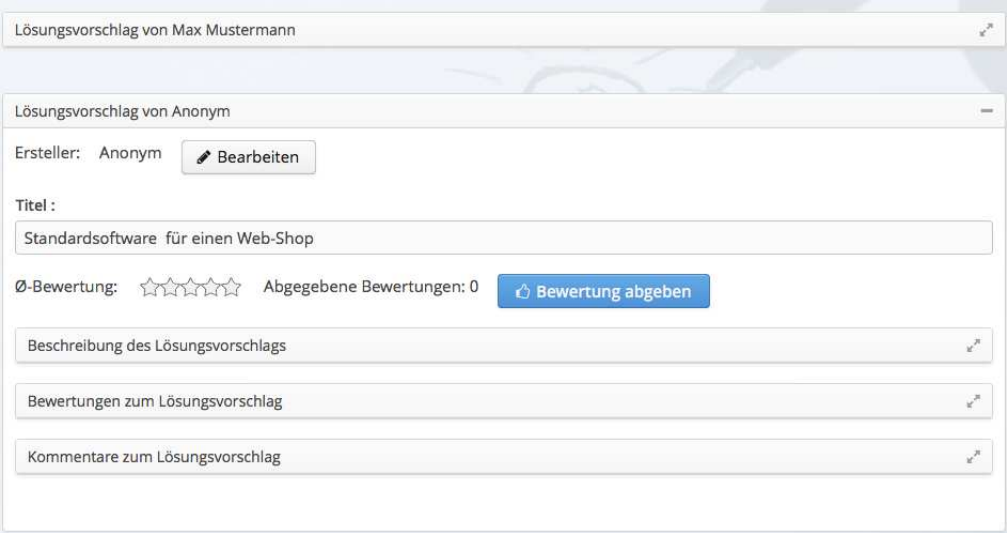


Abbildung 62: Anzeige eines Lösungsvorschlags

In der Abbildung lässt sich der Aufbau des Lösungsvorschlags erkennen. Die Lösungsvorschläge werden in `StackPanel` angezeigt. Der erste Lösungsvorschlag ist aufgeklappt. Hier steht zu erst der Ersteller, der Titel, sowie die Gesamtbewertung. Über den Button *Bewertung abgeben* kann eine neue Bewertung erstellt werden. Innerhalb des `StackPanel`s sind weitere `StackPanel`s enthalten. Das erste enthält die Lösungsvorschlagsbeschreibung, das zweite enthält bereits abgegebene Bewertungen und das dritte enthält Kommentare zum Lösungsvorschlag. Die `StackPanel` lassen sich einzeln auf-und zuklappen.

Technische Umsetzung

Bei der Anzeige der Lösungsvorschläge werden sämtliche zuvor eingetragenen Informationen für alle Anwender des Systems ansprechend aufbereitet und dargestellt. Eine Visualisierung des Befüllens der Felder, wie es im Folgenden beschrieben ist, kann der Abbildung 63 entnommen werden.

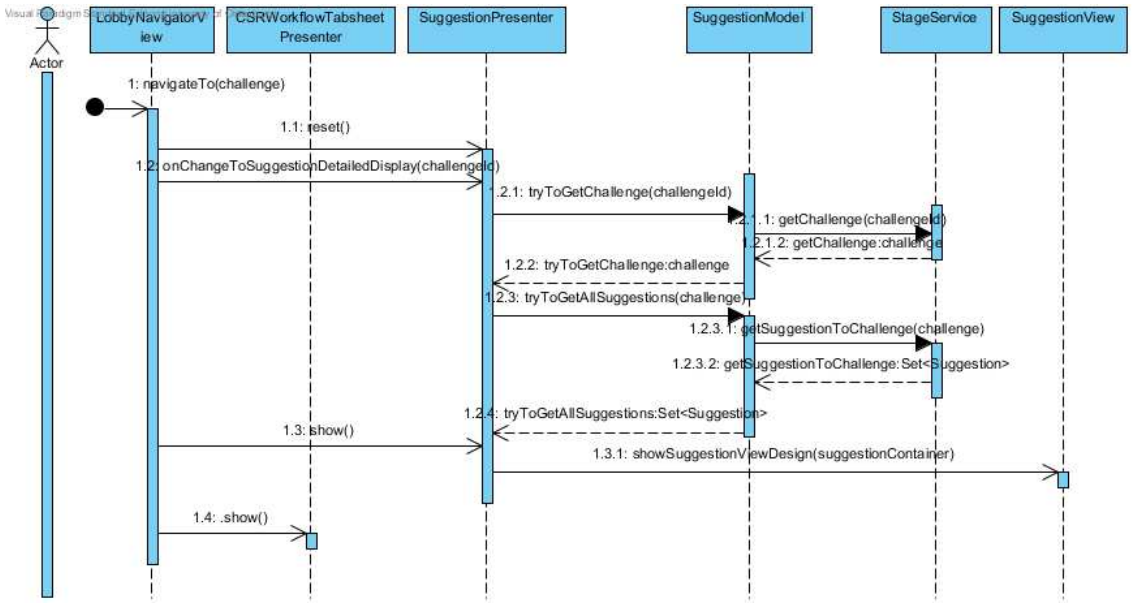


Abbildung 63: Sequenzdiagramm: Suggestion anzeigen

Die Navigation zum Lösungsvorschlag erfolgt über die `LobbyNavigatorView`. Der Tab, in dem sich die Lösungsvorschläge befinden, wurde bereits mit der Erstellung des Challenges-Tabs erstellt.

Zur Anzeige der Lösungsvorschläge wird nach der `reset` Methode des `SuggestionPresenter` die `onChangeToDetailedDisplay(challenge)` Methode aufgerufen. Innerhalb dieser wird über das `Model` auf den `StageService` zugegriffen, worüber die aktuelle Challenge geladen wird. Diese Challenge Informationen werden benötigt, um im zweiten Schritt über das `Model` und den `StageService` die Lösungsvorschläge zu dieser Challenge laden zu können.

Dannach wird `show()` vom `SuggestionPresenter` aufgerufen, die wiederum die Methode `showSuggestionViewDesign` aufruft. Hiermit werden die `SuggestionPanel` erstellt.

Abschließend wird `show()` des `CSRWorkflowPresenter` aufgerufen, in der der Tabwechsel durchgeführt wird.

Der Anwender kann über die Ansicht Lösungsvorschläge bewerten und kommentieren. Sofern der Nutzer über die entsprechenden Rechte verfügt (bspw. weil er der Ersteller ist), wird ihm zusätzlich eine zweite Sicht, die des Editierens, bereitgestellt. Zu erreichen ist dieser Modus durch einen Klick auf den Button *Bearbeiten*, welcher einen `ClickListener` besitzt. Durch ihn wird die Methode `editSuggestion` aufgerufen.

6.4.3. Lösungsvorschlag bearbeiten

Die Bearbeitung eines Lösungsvorschlags kann nur durch den Ersteller durchgeführt werden. Der Aufbau dieser Funktionalität wird in weiteren Verlauf beschrieben.

Oberfläche

Neben dem Erstellen ist es möglich getroffene Angaben zu ändern oder zu erweitern. Über den Button *Vorschlag editieren* kann der Bearbeitungsmodus erreicht werden. Nachfolgende Abbildung stellt die Bearbeitungssicht des einzelnen Lösungsvorschlags dar.

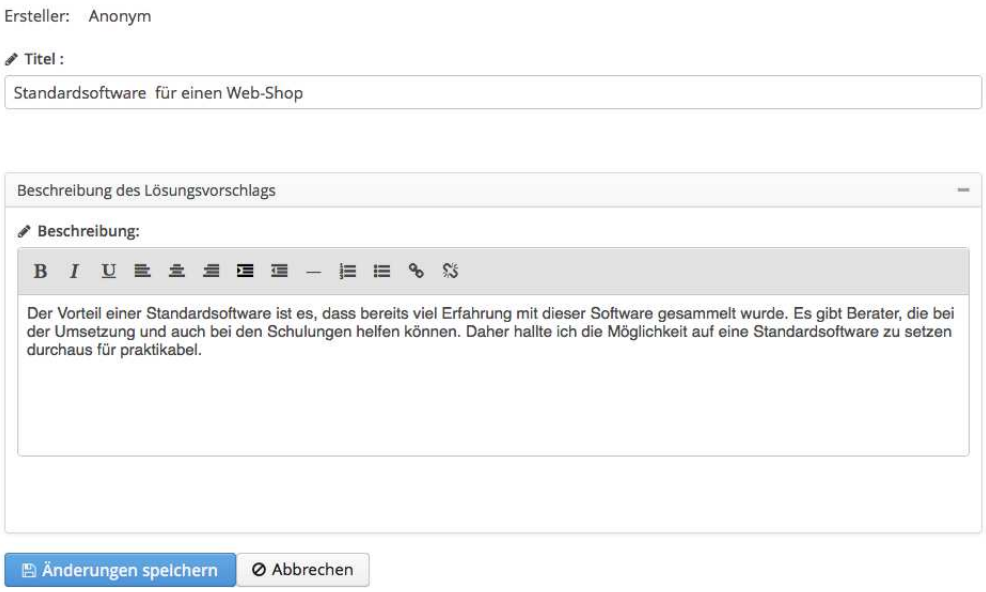


Abbildung 64: Bearbeiten eines Lösungsvorschlags

In der Abbildung ist zu erkennen, dass der Titel und die Beschreibung des Lösungsvorschlags angepasst werden kann. Die Möglichkeit Bewertungen einzusehen oder abzugeben oder zu kommentieren gibt es in dieser Ansicht nicht. Mit Betätigen des Button *Änderungen speichern* werden die Änderungen übernommen. Bei *Abbrechen* werden die Änderungen wieder rückgängig gemacht. Es findet dann ein Wechsel zum Modus der Lösungsvorschlagansicht statt.

Technische Umsetzung

Das Bearbeiten geschieht ähnlich zur Erstellung eines Lösungsvorschlags. Beim Klick auf den Button *Speichern* wird ein `editSuggestionAttemptEvent` geworfen, dass im `SuggestionPresenter` abgefangen wird und die Methode `updateSuggestion(suggestion)` des `Models` aufruft, dass über den `StageService` die Änderungen des Lösungsvorschlags durchführt. Anschließend gibt es den geänderten

Lösungsvorschlag zurück und über den `SuggestionPresenter` wird eine entsprechende Meldung ausgegeben.
Beim Abbrechen der Bearbeitung wird ein `Event` geworfen, woraufhin die Ansicht zurückgesetzt wird und wieder in den Ansichtsmodus gewechselt wird.

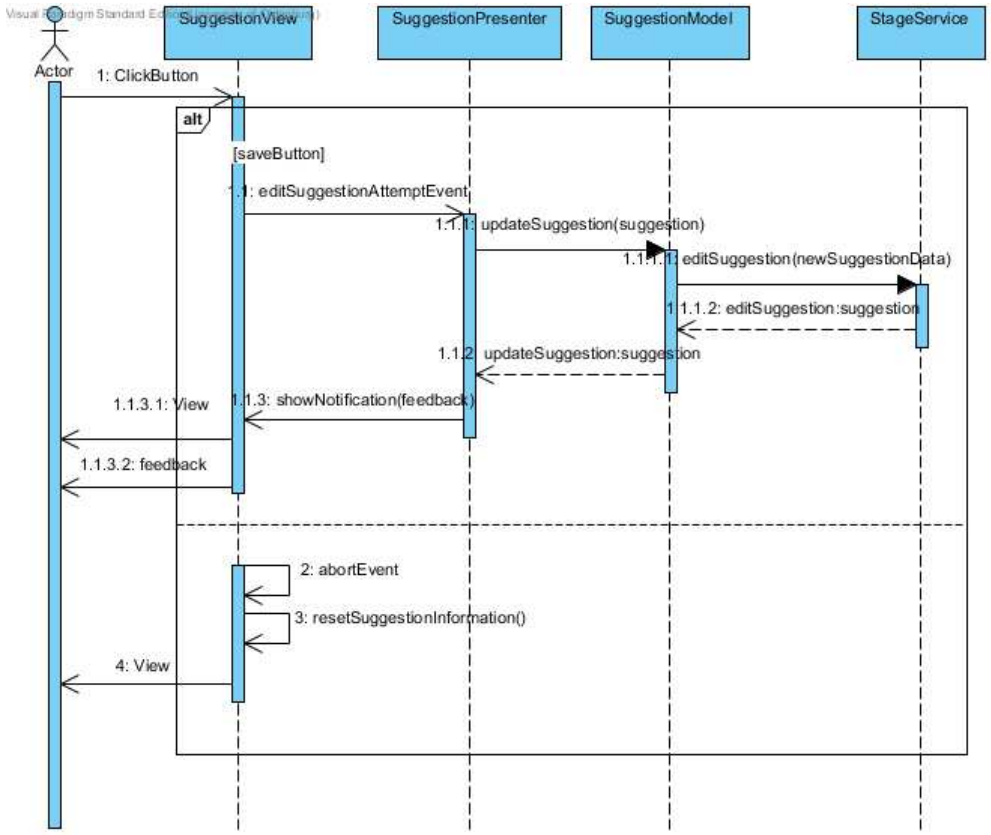


Abbildung 65: Sequenzdiagramm: Lösungsvorschlag bearbeiten

6.5. Übergang: Challenge zu Projekt

Der Übergang einer Challenge zum Projekt wird in einem *Wizard* umgesetzt. Hier kann der Benutzer Schritt für Schritt seine Challenge zum Projekt überführen. Die Erstellung von projektbezogenen, zusätzlichen Daten wird im Kapitel 6.5.1 beschrieben.

Genereller Aufbau

Der Benutzer kann aus einer Challenge heraus, über den Button *Projekt starten*, den *Wizard* zum Anlegen eines Projekts starten. Hier wird er zunächst in den Wizard geführt und kann sich im nächsten Schritt die Daten zur Challenge erneut ansehen. Im darauffolgenden Schritt kann der Benutzer einen existierenden Lösungsvorschlag für die Bewältigung seiner Challenge auswählen oder einen neuen Lösungsvorschlag definieren. Anschließend gelangt

er zum Schritt der Erstellung eines Projekts. Hier können zusätzliche, projektbezogene Daten eingegeben werden, die im Kapitel Projekt erstellen beschrieben werden (vgl. Abschnitt 6.5.1). Im letzten Schritt werden alle Eingaben überprüft und über den Buttons *Projekt starten* das Projekt erstellt. Nun ist es in der *Projektübersicht* sichtbar und kann von dort aus weiter bearbeitet werden.

Oberfläche

Auf der Startseite des *Wizards* wird der Benutzer in den Prozess der Erstellung eines neuen Projekts eingeführt. Der Benutzer kann mit den Button *Weiter* und *Zurück* zwischen den einzelnen Schritten wechseln. Über *Abbrechen* kann der *Wizard* jederzeit beendet werden. Im letzten Wizardschritt über den Button *Projekt erstellen* die finale Erstellung des neuen Projekts durchgeführt werden.

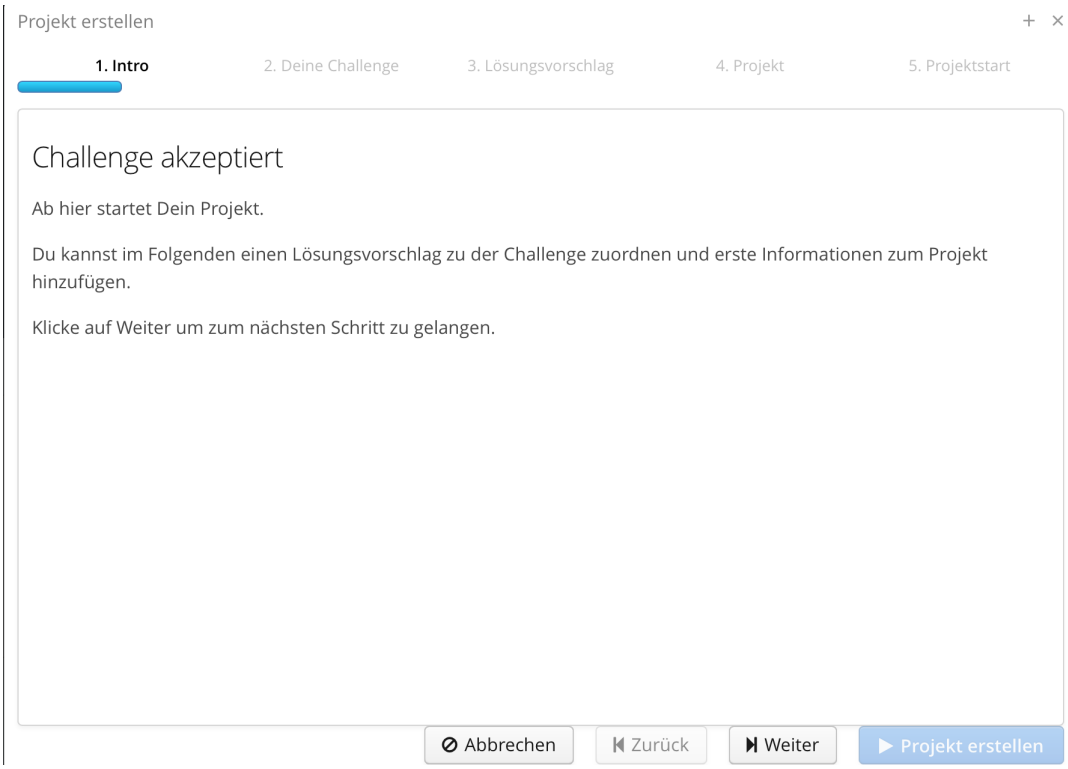


Abbildung 66: Ansicht des Wizard

Technische Umsetzung

Um eine Challenge in ein Projekt zu überführen, muss für die Challenge ein Titel, eine Beschreibung, ein Ziel, die Information über potentielle Hindernisse, mindestens eine Bewertung und mindestens ein Lösungsvorschlag hinterlegt sein. Erst

dann wird der Button *Projekt starten* aktiviert, sodass dieser betätigt werden kann. Die Klasse `TransitionChallengeProjectViewDesign` implementiert das `Interface WizardProgressListener`. Hier wird der *View* ein Objekt des *Wizards* hinzugefügt (vgl. Quellcode 38).

```
33 public Wizard wizard;
34 public TransitionChallengeProjectViewDesign() {
35     wizard = new Wizard();
36     addComponent(wizard);
37 }
```

Quelltext 38: Hinzufügen der Wizard-Komponente

Dem *Wizard* werden über die Methode `addStep` in der `show`-Methode des `TransitionChallengePresenter` einzelne Schritte zu dem *Wizard* hinzugefügt (vgl. Quellcode 39). Mit `view.getWizardFromView` erhält man das *Wizard*-Objekt, das oben beschrieben wurde.

```
33 view.getWizardFromView().addStep( new
       IntroProjectTransitionWizardStep());
34 }
```

Quelltext 39: Hinzufügen eines Schritts im Wizard

Die Schritte implementieren das Interface `WizardStep`. Die nachfolgende Methoden müssen dabei implementiert werden.
Die Methode `getCaption` erwartet als Rückgabewert einen `String` mit der Überschrift des Schritts.
Die Methode `getContent` erwartet ein `Component`, in der die einzelnen *Views* geladen werden können.
Die Methode `onBack` erwartet einen `boolean`-Wert. Bei `true` wird der Button *Zurück* aktiviert.
Die Methode `onAdvance` erwartet einen `boolean`-Wert. Bei `true` wird der Button *Weiter* aktiviert.
Im zweiten Schritt wird die Klasse `ChallengeTransitionStep` eingefügt. Hier wird als Inhalt eine *View* über die `ChallengePresenterImpl` geladen. Weitere Informationen zur Anzeige der Challenge sind im Kapitel 6.3 zu finden.
Im dritten Schritt wird die Klasse `SuggestionTransitionWizardStep` eingefügt, in dessen Inhalt eine `SuggestionSelectionView` geladen wird. Alle Lösungsvorschläge zu einer Challenge werden hier in einer `OptionGroup` dargestellt. Um zum nächsten Schritt

zu gelangen, muss ein Lösungsvorschlag ausgewählt bzw. ein neuer erstellt werden. Im vierten Schritt wird die Klasse `ProjectTransitionWizardStep` eingefügt, in der die `CreateProjectView` geladen wird, die im Kapitel 6.5.1 näher beschrieben wird. Im letzten Schritt des *Wizards* wird `LastProjectTransitionWizardStep` eingefügt. Hier wird der Button *Projekt erstellen* aktiviert. Nach dem Betätigen wird ein Event gesendet, über das die Vollständigkeit des Projekts in der `TransitionChallengeProjektPresenterImpl` überprüft wird.

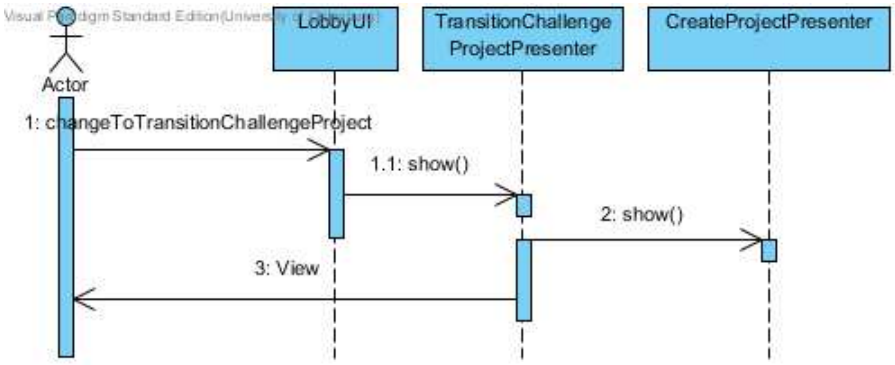


Abbildung 67: Sequenzdiagramm: Übergang von Challenge zu Projekt

6.5.1. Erstellen des Projekts

Über den Wizardschritt *Projekt* wird ein Projekt erstellt und die dafür benötigten Informationen wie Titel, Beschreibung und Projektteam gesammelt. Diese Felder müssen verpflichtend ausgefüllt werden, um den nächsten Wizardschritt zu starten. Das ausgewählte Team wird innerhalb einer Tabelle dargestellt und kann dort bearbeitet werden. Zusätzlich wird ein Projektleiter gewählt, der für den weiteren Projektverlauf eine essenzielle Rolle einnimmt.

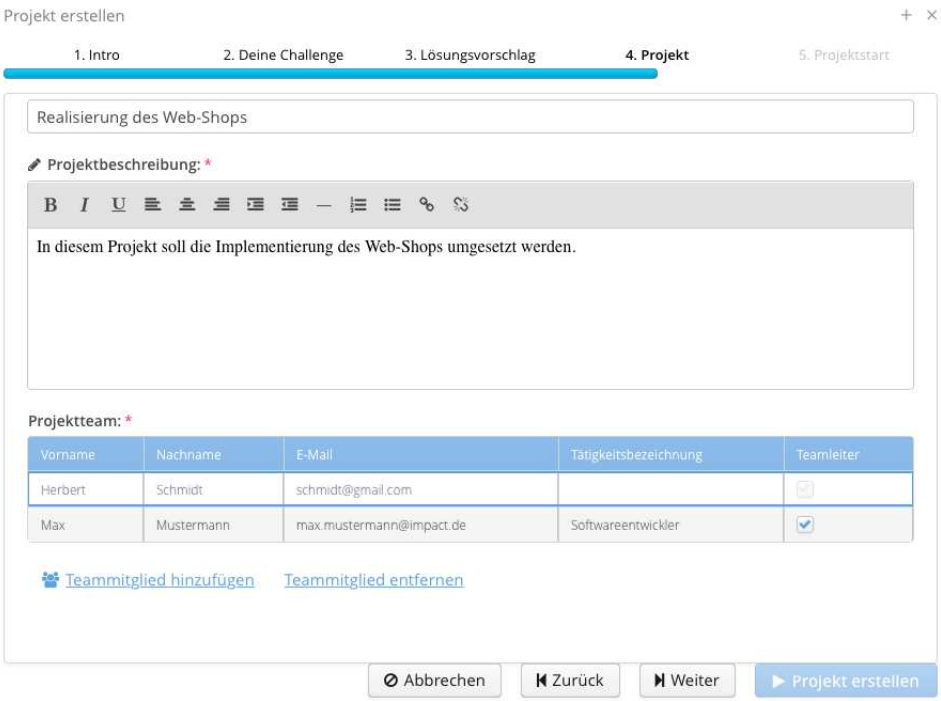


Abbildung 68: Erstellen eines Projektes über den Wizard

Technische Umsetzung

Um ein Projekt erstellen zu können, ist es zunächst notwendig, alle Pflichtfelder auszufüllen. Mit der Methode `onAdvance` wird in der Wizardklasse `ProjectTransitionWizardStep` geprüft, ob die entsprechenden Felder ausgefüllt sind und der Button aktiviert werden kann. Dazu wird die Variable `nextStep` entsprechend `true` oder `false` gesetzt (vgl. Quellcode 40). Mit der Methode `isValid` wird sicher gestellt, dass ein Projekttitel, sowie eine Beschreibung eingetragen wurden.

```
84 public void onAdvance() {
85         boolean nextStep = false;
86     if (createProjectPresenterImpl.getSelectedUser() != null
87                 && createProjectPresenterImpl.getProjectTitle
                       ().isValid()
88                 && createProjectPresenterImpl.
                       getProjectDescription().isValid()
89                 && createProjectPresenterImpl.getTeamLeader
                       () !=null) {
90             nextStep = true;}
```

Quelltext 40: Button zum Erstellen eines Projektes aktivieren

Anschießend wird die Methode `create` des `CreateProjectPresenters` aufgerufen. Innerhalb dieser Methode wird ein Projektobjekt erstellt und im `TransitionChallengeProjectPresenter` gesetzt. Weitergehende Informationen wie Ersteller, Erstelldatum, Projektleiter, vorherige Challenge und Lösungsvorschlag werden gesetzt und das Objekt erzeugt (vgl. Quellcode 41). Nachdem das Projektobjekt erzeugt wurde, wird das Team in dem Wizardschritt `ProjectTransitionWizardStep` hinzugefügt.

```
92 public void create() {
93     project = new Project(title, description, createdDate,
           false, user, "",
94                 "", "", "", challenge, suggestion, false,
                       false, false, false,
95                 false, false, false, false, this.
                       projectLeader);
96 }
```

Quelltext 41: Erstellen eines Projektobjektes in der Methode create

Durch das Betätigen des Buttons wird das Event `closeTransitionChallengeToProjectPopupEvent` ausgelöst, welches aus der Klasse `TransitionChallengeProjectView` gesendet und im `TransitionChallengeProjectPresenter` abgefangen wird. In der Methode `wizard`, welche ausgeführt wird, wenn das Event empfangen wurde, wird der Wizard geschlossen. Über das `TransitionChallengeProjectModel` wird auf den `StageService` zugegriffen, um ein Projekt zu speichern und in die Datenbank einzutragen (vgl. Abbildung 69) Bei dem erfolgreichen Beenden des Vorganges wird ein positives Feedback, in Form einer `Notification`, zurückgegeben. Andernfalls wird ein negatives Feedback ausgegeben. Anschließend wird über die `LobbyNavigatorView` zum erstellten Projekt navigiert.

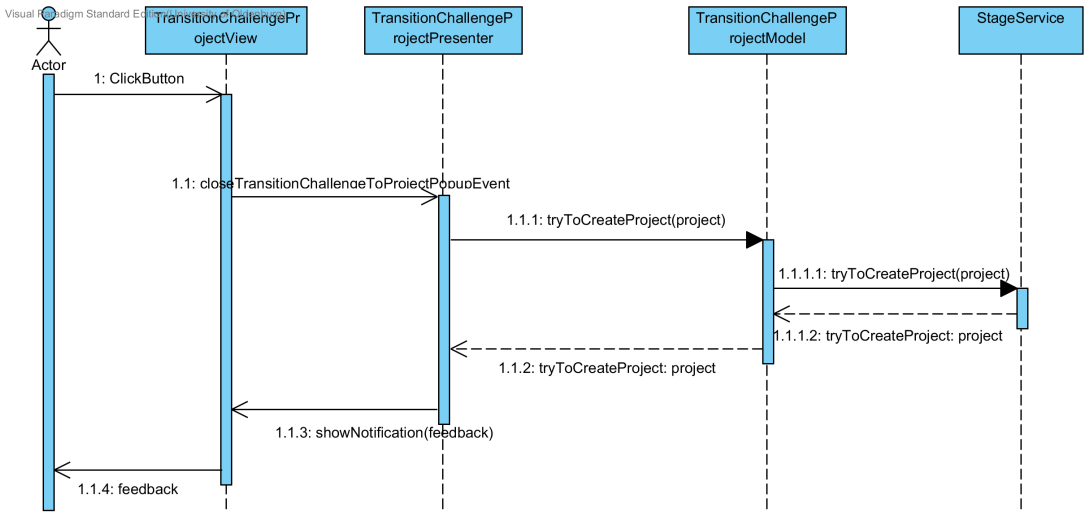


Abbildung 69: Sequenzdiagramm: Projekt erstellen

Projektteam erstellen

Das Hinzufügen eines Projektteams ist in dem Vorgang des Erstellens eines Projektes zwingend erforderlich. Durch das Betätigen des Buttons *Teammitglied hinzufügen* wird das Event `OpenAddUsersToProjectTeamWindowEvent` aus der `CreateProjectView` gesendet. Im `CreateProjectPresenter` wird dieses in der Methode `onOpenAddUsersToProjectTeamWindowAttempt` abgefangen und ein PopUp-Fenster geöffnet, welches erlaubt, alle Anwender in der Webanwendung anzuzeigen und entsprechend auszuwählen. Dazu wird zuvor ein `BeanContainer` erstellt, welcher alle Anwender aus der Anwendung enthält, die nicht in dem Projektteam vorhanden sind. Dazu wird über das Model auf den `StageService` zugegriffen, um die entsprechenden Nutzer zu erhalten (vgl. Quellcode 42). Dieser BeanContainer wird als Datenquelle genutzt, um die diversen Anwender aus einer `ComboBox` auszuwählen und anzuzeigen.

```
166 public void onOpenAddUsersToProjectTeamWindowAttempt() {
167     BeanContainer<Integer, User> bc = new BeanContainer<
        Integer, User>(
168             User.class);
169      bc.setBeanIdProperty(User.USER_PROPERTY_USERID);
170      ArrayList<User> users = model.getAllUsers();
171 }
```

Quelltext 42: Erstellen eines Projektteams

Hat der Anwender alle entsprechenden Teammitglieder ausgewählt, so kann er dieses durch Betätigen des Buttons *Speichern* abspeichern. Dabei wird das Event `SaveProjectTeamAttemptEvent` aus dem `CreateProjectPresenter` gesendet und im Zuge dessen über die Methode `onSaveProjectTeamAttempt` abgefangen. Innerhalb dieser Methode werden zunächst die ausgewählten Teammitglieder zu einem `HashSet` hinzugefügt, die `CreateProjectView` erneuert, und die hinzugefügten Teammitglieder in der Tabelle angezeigt werden(vgl. Abbildung 70).
Es besteht zudem die Möglichkeit, einen Teamleiter auszuwählen. Standardmäßig übernimmt diese Rolle der Ersteller des Projektes. Falls jedoch ein anderes Teammitglied diese Rolle bekommen soll, ist es möglich, dies in der Teamtabelle, mithilfe einer `Checkbox`, auszuwählen.
In dem `TransitionChallengeProjectPresenter` wird in der Methode `wizard` das aktuelle Team zu dem Projekt hinzugefügt, über die Methode des Models `addUserToProject` in die Datenbank eingetragen und somit gespeichert. Die Methode `wizard` wird dabei durch das Beenden des Wizards aufgerufen (vgl. Abschnitt 6.5.1).

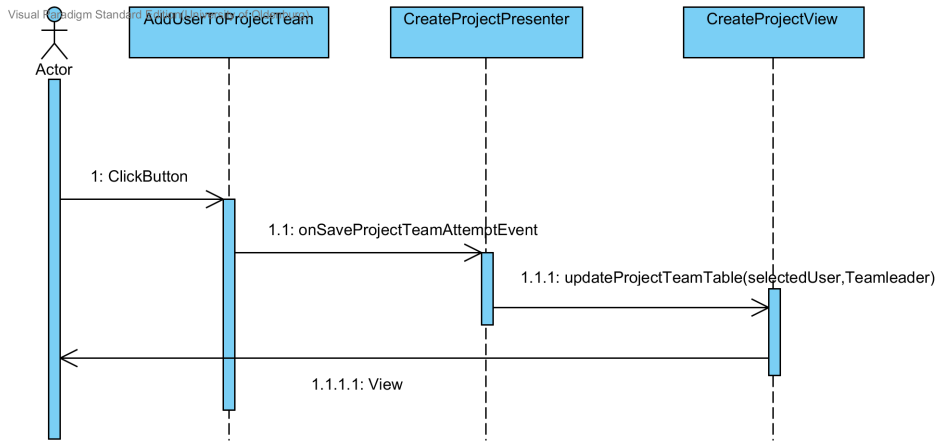


Abbildung 70: Sequenzdiagramm: Projektteam erstellen

6.5.2. Projekt anzeigen

Die Anzeige eines Projektes wird innerhalb eines *TabSheets* umgesetzt. In diesem Kapitel wird darauf eingegangen, wie das Anzeigen eines Projektes umgesetzt wurde.

Oberfläche

Nachdem ein Projekt erstellt oder ausgewählt wurde, gelangt man über die `LobbyNavigatorView` zur detaillierten Ansicht des Projektes. Diese Ansicht wird in ein `Tabsheet` geladen. In den weiteren Tabs sind die Ansichten des Business Cases bzw. Feedbacks verborgen. Die gesamte Übersicht eines Projektes ist mit Hilfe von `StackPanels`

gegliedert (vgl. Abbildung 72). Dies führt dazu, dass der Anwender nicht scrollen muss, sondern gezielt nach Informationen suchen und diese einsehen oder bearbeiten kann. Die Gruppierung der Informationen sind nach folgenden Themengebieten gegliedert:

- allgemeine Informationen,
- Projektteam,
- Arbeitsschritte,
- Business Case Grundlagen,
- Anschaffungen und Gewinne.

Hinter den `StackPanel`s verbergen sich verschiedene Eingabefelder und Tabellen. Durch das Aufklappen eines `StackPanel`s sind diese zugänglich. Hinter dem `StackPanel` *Allgemeine Informationen* sind Titel und Beschreibung zu finden. Hinter den `StackPanel`s *Projektteam, Arbeitsschritte, Anschaffungen und Gewinne* verbergen sich die entsprechenden, tabellarischen Darstellungen der Informationen. Das `StackPanel` *Business Case Grundlagen* beinhaltet Eingabefelder wie Anwendungsdauer, Nutzeranalyse und Risikobetrachtung. Permanent stehen dem Benutzer Informationen über die vorherige Challenge und dem Lösungsvorschlag zur Verfügung. Besitzt der Anwender die Rechte zur Bearbeitung eines Projektes, so werden zusätzlich die Buttons *Bearbeiten* und *kollaborativer Raum* sichtbar.
Besitzt ein Benutzer nur Leserechte, so sind die zuvor genannten Buttons nicht sichtbar. Alle anderen Informationen sind für den Benutzer einsehbar, können jedoch nicht bearbeitet werden.

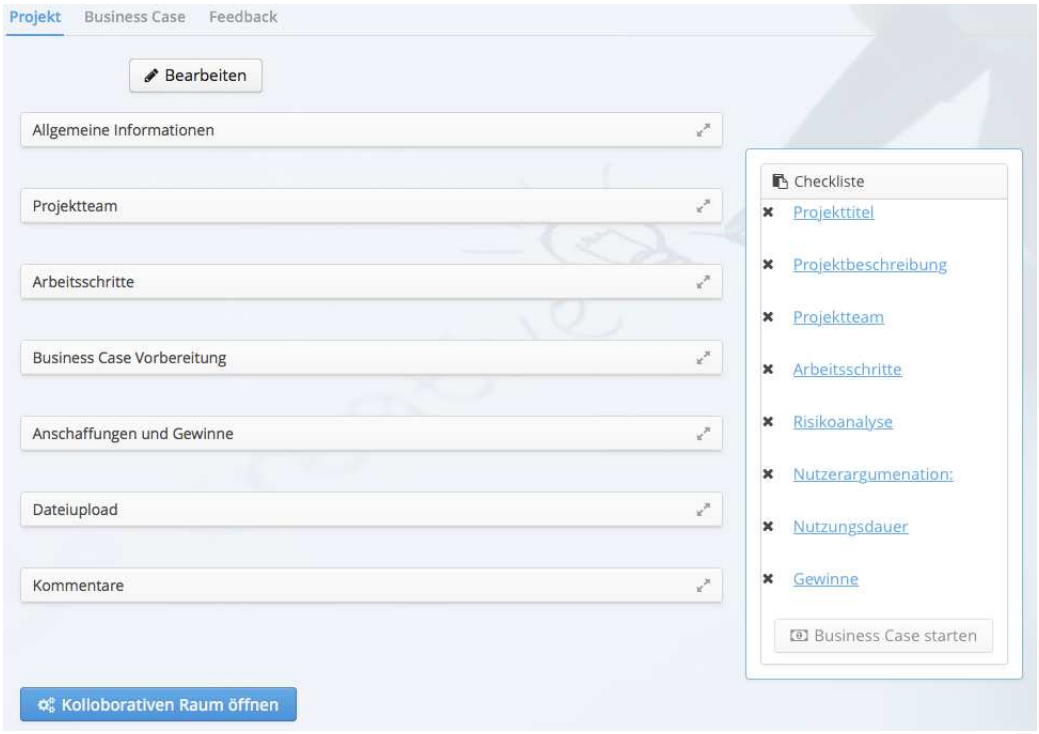


Abbildung 71: Darstellung eines Projektes

Technische Umsetzung

Bei dem Anzeigen eines Projektes werden alle Informationen zur Ansicht bereitgestellt. Je nachdem, welche Rechte der Anwender besitzt, ist es zusätzlich möglich, ein Projekt zu bearbeiten (vgl. Kapitel 4.1). Über die `LobbyNavigatorView` wird zunächst das `WorkflowTabsheet` aufgerufen. In diesem Tabsheet befinden sich die Reiter *Projekt*, *Business Case* und *Feedback*. In dem `WorkflowTabsheetPresenter` wird die Methode `onChangeToProjectDetailedDisplay` des `ProjectPresenters` aufgerufen, um die Projektansicht eines bestimmten Projektes zu befüllen und aufzurufen (vgl. Abbildung 72). Zuvor muss der `ProjectPresenter` in dem `WorkflowTabsheetPresenter` definiert werden. Zusätzlich wird der erste Tab des Tabsheets ausgewählt und angezeigt, da sich dahinter das Projekt verbirgt.

In der Methode `onChangeToProjectDetailedDisplay` wird das entsprechende Projektobjekt, anhand der `projectId`, über den `StageService`, aus der Datenbank herangezogen. Die Methode `populateView` setzt die Inhalte in die entsprechenden Felder und ermöglicht die Bearbeitung sowie die Anzeige eines Projektes. Besitzt der Anwender keine Rechte zur Bearbeitung eines Projektes, so wird die Methode `populateReadView` ausgeführt. Diese ermöglicht das Anzeigen eines Projektes. Es wird nur zur Anzeige des Projekts navigiert,

wenn ein Projekt gefunden wurde. Ansonsten wird der Anwender zur Projektübersicht geleitet.

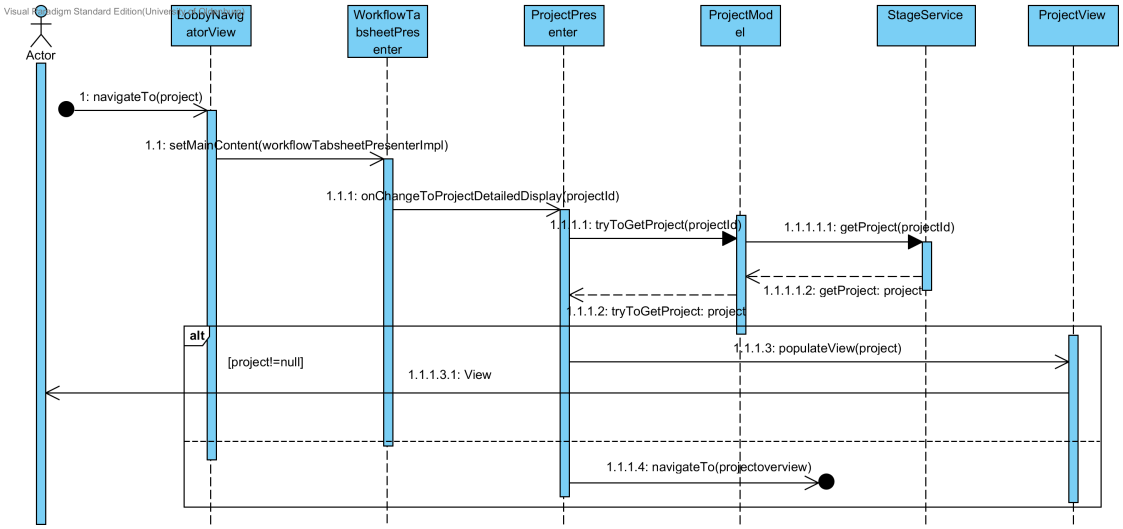


Abbildung 72: Sequenzdiagramm: Projekt anzeigen

Projektteam anzeigen

Um das Projektteam anzuzeigen, muss zunächst das entsprechende `StackPanel` geöffnet werden. Bei dem Öffnen des `StackPanels` wird das Event `FillProjectTeamTableEvent` gesendet und in der Methode `onFillProjectTeamTable` des `ProjectPresenters` abgefangen. In dieser Methode wird ein `BeanContainer` für den `User` erstellt, um anschließend über das Model und den `StageService`, das Projekt zu erlangen und darüber auf alle Teammitglieder zuzugreifen (vgl. Abbildung 73). Alle Elemente des `Teamsets` werden anschließend zu dem `BeanContainer` hinzugefügt (vgl. Quellcode 43).

```
1437 public void onFillProjectTeamTable(@Observes
        FillProjectTeamTableEvent event) {
1438
1439     BeanContainer<Integer, User> teambc = new BeanContainer<
            Integer, User>(
1440                 User.class);
1441         teambc.setBeanIdProperty(User.USER_PROPERTY_USERID);
1442         Set<User> teamSet = model.getProject().getTeamSet();
1443         teambc.addAll(teamSet);
1444 }
```

Quelltext 43: Methode zum Befüllen der Teamtabelle

Um die Mitglieder eines Projektteams einsehen zu können, muss der `BeanContainer` als Datenquelle der Teamtabelle gesetzt werden. Das setzen der Datenquelle muss bei jedem Öffnen des `StackPanel`s neu erfolgen, da es sonst zu Problemen bei der Aktualisierung der Tabelle kommt. Mit diesem Verfahren kann gewährleistet werden, dass dem Anwender die aktuellsten Daten zur Verfügung stehen.

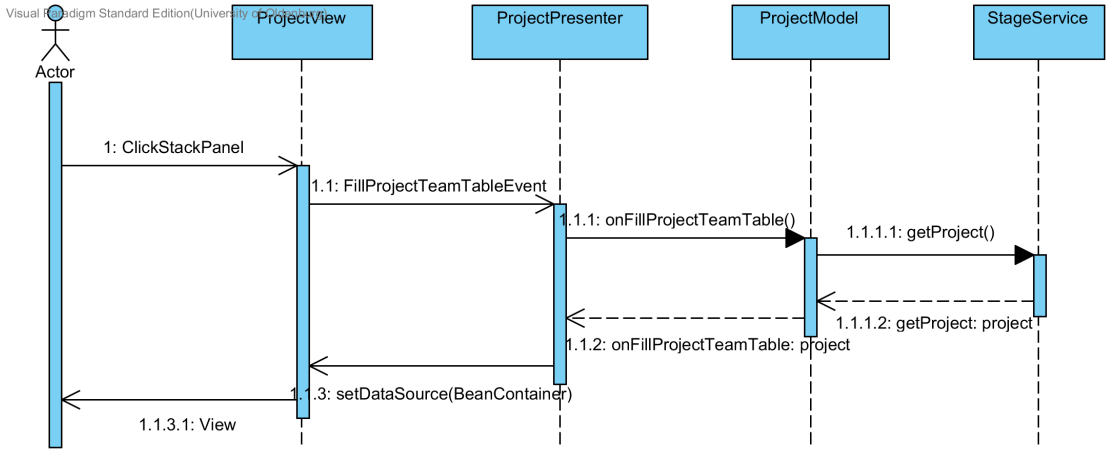


Abbildung 73: Sequenzdiagramm: Projektteam anzeigen

6.5.3. Projekt bearbeiten

Das Bearbeiten eines Projektes ist eine Kernfunktionalität, die in diesem Kapitel erklärt wird. Zunächst wird die Oberfläche beschrieben und anschließend die technische Umsetzung.

Oberfläche

Ähnlich wie bei dem Anzeigen des Projektes sind alle Felder mit den Informationen befüllt. Jedoch ist es möglich diese Werte zu bearbeiten und zu löschen. Zusätzlich werden an jedem editierbaren Eingabefeld ein Stift angezeigt, um den Bearbeitungsmodus grafisch zu unterstreichen. Ebenfalls werden ein Speicher-Button und ein Abbruch-Button angezeigt, um die Änderungen zu Speichern oder diese zu verwerfen.

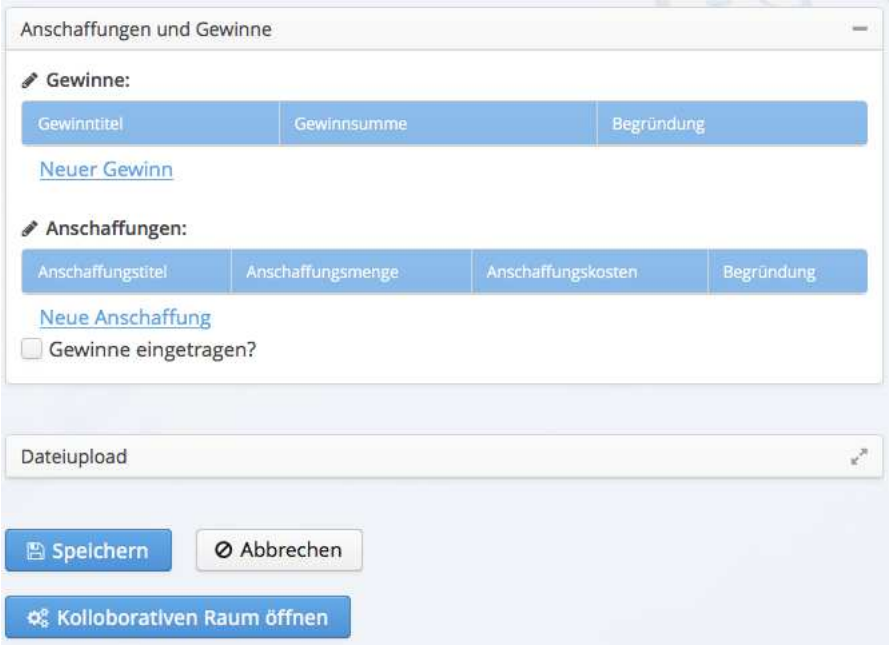


Abbildung 74: Bearbeiten eines Projektes

Technische Umsetzung

Das Bearbeiten eines Projektes ist nur möglich, wenn der Anwender ein Teammitglied oder ein Teamleiter darstellt und über die entsprechenden Rechte besitzt (vgl. 4.1). Alle anderen Anwender besitzen lediglich die Rechte zum Ansehen eines Projektes.
Um den Bearbeitungsmodus zu starten, ist es notwendig, den Button *Bearbeiten* zu betätigen. Dadurch wird die Methode `toggleDisplayMode` aufgerufen, welche zwischen dem Bearbeitungsmodus sowie dem Ansichtsmodus unterscheidet und je nach Situation, in diesen dementsprechend wechselt. In dem Bearbeitungsmodus wird die Methode `setEdit` der `ProjectView` aufgerufen. In dieser werden alle Buttons angezeigt, die dafür notwendig sind, um ein Projektteammitglied hinzuzufügen und um im nächsten Schritt einen Gewinn, eine Anschaffung und einen Arbeitsschritt zu erstellen. Ebenso wird der Speicher-Button sichtbar und die diversen `CheckBox`en für die Checkliste werden zur Bearbeitung freigegeben. Diese können jedoch nur von dem Projektleiter bearbeitet werden. Alle restlichen Textfelder werden ebenfalls in diesen Modus gesetzt.
Werden Veränderungen in den einzelnen Eingabefeldern o.ä. vorgenommen, muss anschließend der Button *Speichern* betätigt werden, um die Veränderungen zu speichern. Durch das Betätigen des Buttons wird das Event `SaveProjectAttemptEvent` aus der `ProjectView` gesendet und in der Methode `tryToSaveProject` des `ProjectPresenters` abgefangen. In dieser Methode werden alle Werte aus der `View` ausgelesen und in dem Projektobjekt gesetzt. Dazu gehört auch das Entfernen von Teammitgliedern, welches

unter *Projektteam bearbeiten* genauer erläutert wird. Das Hinzufügen von Teammitgliedern erfolgt wie beim Projektteam erstellen (siehe 6.5.1). Anschließend wird die Methode `tryToEditProject` des `ProjectModels` aufgerufen und über den `StageService` versucht das Projekt zu verändern und diese Veränderungen in der Datenbank über den `DatabaseService` einzutragen. Die Methode `editProject` des `DatabaseServices` sucht zunächst nach dem Projekt in der Datenbank, welches bearbeitet werden soll und setzt das gefundene Projekt als `foundProject` (vgl. Quellcode 44). Anschließend werden die Werte des übergebenen Projektes in dem gefundenen Projekt gesetzt und mit dem `manager` bestätigt.

```
1023 private Project editProject(Project project,
1024             List<FileReference> fileReferences,
1025             Set<Attachment> attachmentsToDelete) {
1026
1027     manager.getTransaction().begin();
1028     Project foundProject = manager.find(Project.class,
                 project.getId());
1029 }
```

Quelltext 44: Methode zum Speichern eines bearbeiteten Projektes

Ist das bearbeitete Projekt erfolgreich gespeichert, bekommt der Anwender eine Rückmeldung in Form eines Feedbacks, dass das Projekt erfolgreich gespeichert wurde.
Möchte der Anwender seine Veränderungen nicht speichern, so kann er durch den Abbruch-Button den Vorgang beenden und erhält die Ansicht des nicht bearbeiteten Projektes (vgl. Abbildung 75). Nachdem das Projekt gespeichert und der Speichern-Button betätigt wurde, wird wieder in den Ansichtsmodus navigiert, damit die Änderungen erkannt werden.

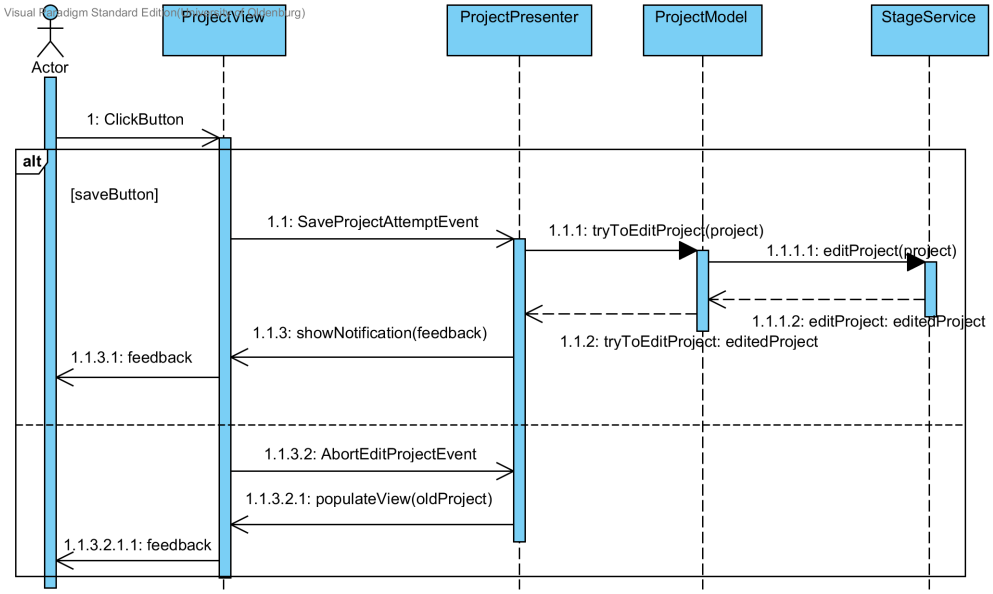


Abbildung 75: Sequenzdiagramm: Projekt bearbeiten

Projektteam bearbeiten

Zum Bearbeiten eines Projektteams stehen mehrere Funktionen zur Verfügung. Dazu gehören das Hinzufügen eines neuen Teammitglieds, das Setzen eines neuen Projektleiters und das Entfernen eines Teammitglieds. Das Hinzufügen und das Setzen eines neuen Projektleiters wird bereits in dem Kapitel 6.5.1 erklärt. Das Entfernen eines Teammitglieds wird über den Button *Teammitglied entfernen* und das auswählen eines Teammitgliedes aus der Teamtabelle geregelt. Durch das Betätigen des Buttons wird die Methode `removeUserFromProjectTeamTable` aufgerufen, in der das Löschen eines Nutzers aus dem Team vorgenommen wird. Das ausgewählte zu entfernende Teammitglied wird in der Datenquelle der Tabelle herausgesucht und entfernt. Dabei ist es wichtig zu unterscheiden, ob ein oder mehrere Teammitglieder gleichzeitig zu entfernen sind und ob generell ein Mitglied ausgewählt wurde und die Teamtabelle existiert (vgl. Quellcode 45). Durch das Speichern des Projektes werden die Änderungen des Projektteams ebenfalls gespeichert.

```
860 private void removeUserFromProjectTeamTable() {
861
862 if (projectTeamTable != null && projectTeamTable.getValue()
    != null) {
863             if (projectTeamTable.getValue() instanceof Set) {
864                 for (int i : (Set<Integer>) projectTeamTable.
                        getValue()) {
```

```
865                         projectTeamTable.getContainerDataSource()
                               .removeItem(i);
866                     }
867                 } else {
868                     int i = (int) projectTeamTable.getValue();
869                     projectTeamTable.getContainerDataSource().
                           removeItem(
870                                projectTeamTable.getValue());
871                 }}
872 }
```

Quelltext 45: Methode zum Entfernen eines Projektteammitglieds

6.6. Übergang: Projekt zu Business Case

Betätigt der Projektleiter den Button *Business Case starten*, so wird das Event `CreateNewBusinessCaseAttemptEvent` aus der `ProjectView` gesendet und in der Methode `tryToCreateNewBusinessCaseAttempt` des `ProjectPresenter` abgefangen. Innerhalb der Methode wird über die Methode `createBusinessCase` des *Models* ein neuer Business-Case erstellt aus den gesammelten Informationen des Projektes und des vorangegangen Projekt-Objektes (vgl. Quellcode 46). Wurde der Business Case erfolgreich erstellt, so bekommt der Anwender ein Feedback, dass der Business Case erfolgreich erstellt wurde (vgl. Abbildung 76). Zudem wird man über die `LobbyNavigatorView` zu dem Business Case geleitet. Wurde der Business Case nicht erfolgreich erstellt, so bekommt der Benutzer ein negatives Feedback, dass der Vorgang fehlgeschlagen ist.

```
404 BusinessCase businessCase = new BusinessCase(title, createdAt
    ,
405                 createdBy, anonymous, shortSummary,
                       conclusion, riskAnalysis,
406                 benefitAnalysis, usefulAnalysis,
                       purchaseCosts, profit,
407                 neededAsset, project, false, false, false,
                       false, false, false,
408                 false, "");
```

Quelltext 46: Erzeugen eines Business Cases

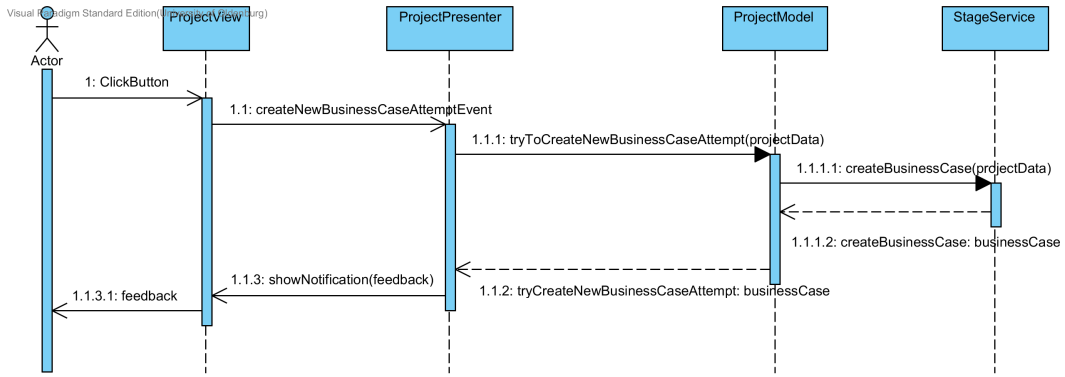


Abbildung 76: Sequenzdiagramm: Business Case erstellen

6.7. Business Case

Ein Business Case (BC) ist ein Entscheidungsunterstützungsinstrument, welches zur betriebswirtschaftlichen Beurteilung eines Projektes dient. Es wird ein Nachweis über die Wirtschaftlichkeit des Szenarios, welches vorher im Projekt genau ausgearbeitet wurde, erarbeitet. Bei der Implementierung kann zwischen den drei Kernfunktionen Erstellen, Anzeigen und Editieren unterschieden werden. Da das Erstellen des Business Case ein Teil der Projekt-Komponente ist, wir dieses schon vorher in Abschnitt 6.6 dargestellt. Zudem gibt es beim Anzeigen und Editieren die Besonderheit, dass es verschiedene Zustände gibt, die jeweils unterschiedliche Auswirkungen auf die Oberfläche haben. Dies ist der Fall, da die BC-Komponente dem Anwender die Möglichkeit bietet nach einer wirtschaftlichen Beurteilung eine Entscheidung zu dem dazugehörigen Projekt auszusprechen. Im Folgenden werden zunächst die Zustände erläutert und darauf aufbauend das Anzeigen und Editieren dargestellt.

6.7.1. Business Case Zustände

Um die unterschiedlichen Zustände des Business Case erklären zu können, müssen zunächst zwei Anwendertypen aufgezeigt werden, welche mit dieser Komponente arbeiten. Einerseits gibt es den Controller, der durch sein betriebswirtschaftliches Wissen den eigentlichen Business Case durchführt. Andererseits existiert der Entscheider, welcher schlussendlich mit Hilfe des BC das Projekt beurteilt und ein Entscheidung triff. Beide Anwender haben unterschiedliche Benutzerrechte, wodurch sie die einzelnen Zustände jeweils anders beeinflussen können.
Insgesamt gibt es sechs verschiedene Zustände, welche in dem Zustandsdiagramm 77 abgebildet sind. Sobald ein Projekt in einen Business Case überführt wird hat dieser den Zustand *INPROGRESS*. In diesem Fall kann der Controller den Business Case durchführen bzw. bearbeiten, wohingegen der Entscheider keine Rechte zur Bearbeitung sondern nur

zur Einsicht hat. Wenn der Controller alle erforderlichen Informationen zusammengestellt hat, kann er den Business Case als *fertig* markieren, wodurch er den Zustand *WAITINGREVIEW* erhält. Nun kann der Entscheider den vorgelegten Business Case beurteilen. Fällt die Entscheidung so aus, dass dem Projekt zugestimmt wird, ändert sich der Zustand auf *ACCEPTED* und das Projekt kann durchgeführt werden. Wird das Projekt jedoch abgelehnt, so wird der Zustand auf *REJECTED* gesetzt. Sofern der Entscheider eine Überarbeitung des Business Case fordert, wird der Zustand auf *REVISEBC* geändert. In diesem Zustand hat der Controller die Möglichkeit seinen Wirtschaftlichkeitsnachweis zu überarbeiten. Ist dies geschehen und dieser mit *fertig* markiert, erhält er wieder den Zustand *WAITINGREVIEW*. Für den Fall, dass der Entscheider eine Überarbeitung des Projektes fordert, wird der Zustand des Business Cases und des Projektes auf *REVISEPROJECT* gesetzt und das Projekt muss überarbeitet werden. Ist dies geschehen, kann das Projekt wieder in den Business Case überführt werden und der Prozess beginnt von vorne.

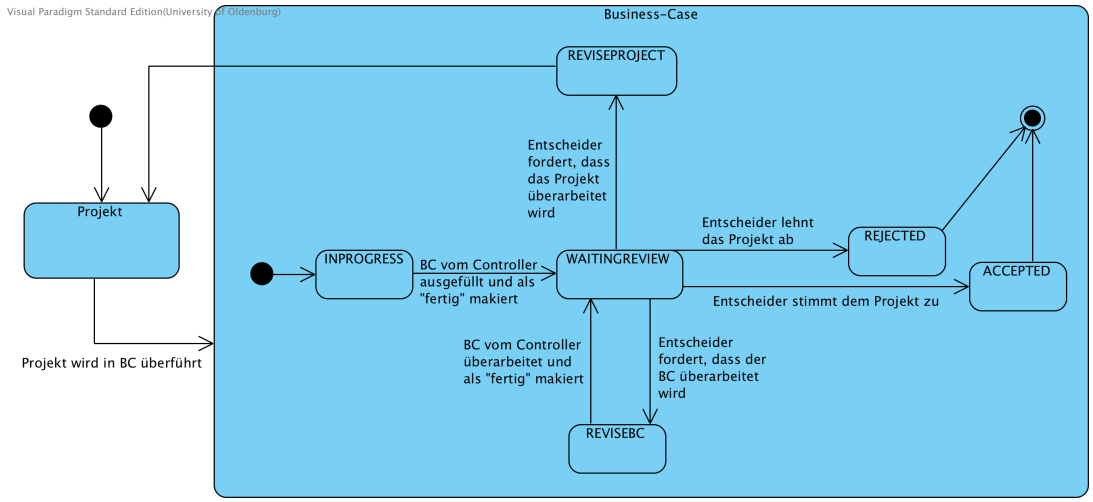


Abbildung 77: Zustandsdiagramm: Business Case

6.7.2. Business Case anzeigen

Innerhalb des Business Cases wird dem Anwender die Funktionalität geboten, das ihm alle im Projekt gesammelten Informationen, sowie die durch den Controller hinzugefügten Daten, angezeigt werden. Hierfür sind jedoch die nötigen Rechte vorhanden, ansonsten erscheint eine Ausweich-Seite. Im Folgenden wird zunächst die Oberfläche und danach die dazugehörige technische Umsetzung beschrieben.

Oberfläche

Sobald der Business Case erstellt oder ausgewählt wurde, gelangt man über die `LobbyNavigatorView` zu der detaillierten Ansicht des Business Cases. Diese Ansicht wird in ein *Tabsheet* geladen und zwischen die beiden Tabs *Projekt* und *Feedback* eingeordnet. Wie der Abbildung 78 zu entnehmen ist, wird die Gesamtübersicht mit Hilfe von `StackPanels` gegliedert.

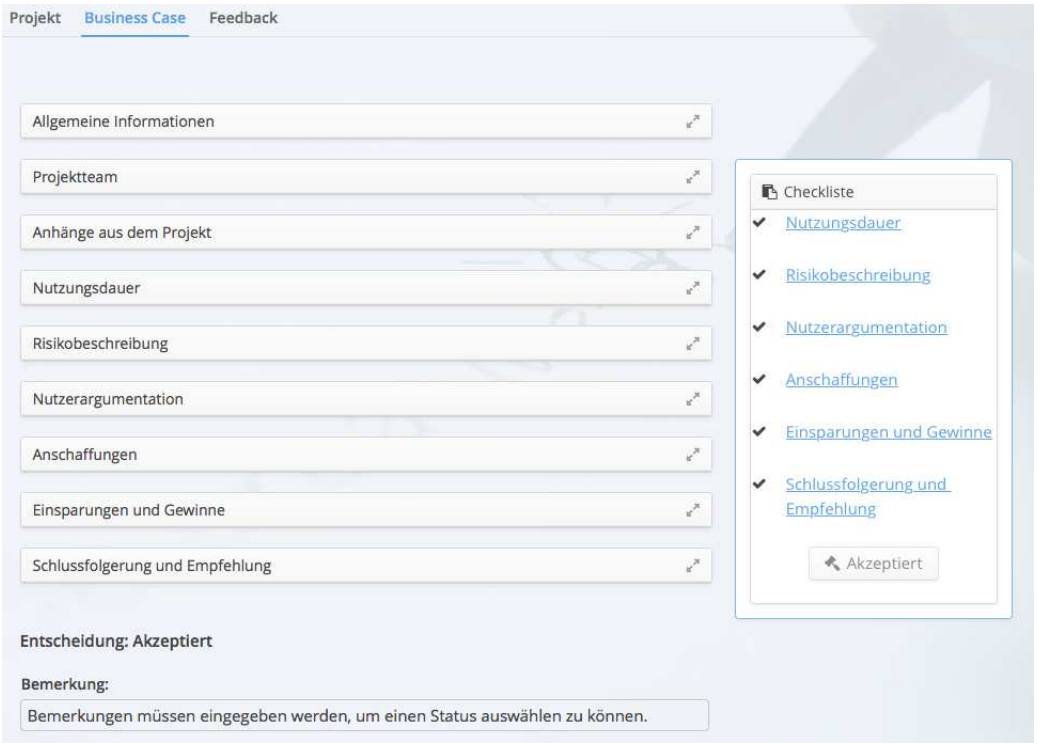


Abbildung 78: Anzeige eines Business Cases

Hierbei sind die Informationen so gegliedert, wie es in einem Business Case üblich ist:

- Allgemeine Informationen: Unter diesem Punkt sind Verlinkungen zur dazugehörigen Challenge und zum Projekt, der Projektverantwortliche und eine Projektbeschreibung aufgeführt.
- Projektteam: In diesem `StackPanel` werden alle Mitglieder des Projektes aufgelistet.
- Anhänge aus dem Projekt: Die zuvor hochgeladen Anhänge aus dem Projekt werden mittels Download zur Verfügung gestellt.
- Nutzungsdauer: Unter diesem Punkt wird die geschätzte Nutzungsdauer und eine Begründung hierfür aufgeführt. Ferner stehen die Dateien, die der Controller diesbezüglich hochgeladen hat, zum Download zur Verfügung.

- Risikobeschreibung: Dieses `StackPanel` enthält die Risikobeschreibung aus dem Projekt, sowie die Dateien, die der Controller diesbezüglich hochgeladen hat.
- Nutzerargumentation: Dieser Punkt ist analog zur Risikobeschreibung aufgebaut.
- Anschaffungen: Dieser Punkt ist analog zur Risikobeschreibung aufgebaut.
- Einsparungen und Gewinne: Dieser Punkt ist analog zur Risikobeschreibung aufgebaut.
- Schlussfolgerung und Empfehlung: Dieses `StackPanel` enthält eine vom Controller verfasste Empfehlung bezüglich der Durchführung.

Zudem ist auf der linken Seite eine Checkliste, die anzeigt, welche Bereiche der Controller bereits abgeschlossen hat. Unter dieser Liste ist eine Button, welcher sobald alle benötigten Informationen vorhanden sind, betätigt werden kann. Diese Elemente sind für alle Zustände, welche in Abschnitt 6.7.1 beschrieben werden, gleich. Im Folgenden wird je Zustand und Benutzerrolle die Elemente aufgezeigt, welche hinzukommen:

- *INPROGRESS:* Falls der Anwender die Rechte zum Bearbeiten besitzt (Controller), wird in den `StackPanels`, in denen Dateien hochgeladen werden können, die Buttons *Dateien hochladen* und *Speichern* und eine `Checkbox` zusätzlich angezeigt. Mittels der `Checkbox` wird angegeben, ob alle benötigten Dateien für diesen Informationsbereich hochgeladen wurden. Zudem gibt es einen *Speichern*-Button im `StackPanel` für die Schlussfolgerung und Empfehlung.
- *WAITINGREVIEW:* In diesem Fall werden dem Entscheider unter den `StackPanels` die Buttons *Projekt kann durchgeführt werden*, *Projekt kann NICHT durchgeführt werden*, *Business Case muss überarbeitet werden* und *Projekt muss überarbeitet werden* angezeigt. Zusätzlich ist unter diesen noch ein Textfeld für Bemerkungen.
- *ACCEPTED:* Unter den `StackPanels` befindet sich ein Schriftzug, dass das Projekt akzeptiert wurde und eine Begründung vom Entscheider.
- *REJECTED:* Aufbau ist analog zum *ACCEPTED*-Zustand.
- *REVISEBC:* Zusätzlich zu den Elementen aus dem Zustand *INPROGRESS* kommt ein Schriftzug unter die `Stackpanels`, das der Business Case überarbeitet werden muss, mit einer Begründung vom Entscheider.

Technische Umsetzung

Die Umsetzung, wie ein Business Case angezeigt wird, ist zu großen Teilen analog zu der technischen Umsetzung des Projektes, welche in Abschnitt 6.5.1 erläutert wird. Daher wird auf diesen Teil nicht genauer eingegangen. Unterschiede sind erst in der Methode `show` der Klasse `BusinessCaseStepPresenter` zu finden. In dieser wird zu Beginn die `populateView` Methode aufgerufen, wodurch die *View* mit Daten aus dem *Model* gefüllt wird. Hierfür wurde vorher das Business Case Objekt in das *Model* durch die `setBusinessCaseId(id)` Methode innerhalb der `LobbyNavigatorView` geladen. Zudem wird je nach Zustand bspw. der Button der Checkliste angepasst. Anschließend erfolgt ein Aufruf der Methode `setEditmode`, durch welche je nach Berechtigung und aktuellem Zustand der Business Case zur Bearbeitung freigegeben wird oder nicht. Danach findet der Aufruf der `setManagemode` Methode statt. Sie beeinflusst je nach Berechtigung und Zustand, ob die Sicht für den Entscheider angezeigt wird. Schlussendlich wird die Methode `setMainlayoutVisible` der `BusinessCaseStepView` innerhalb des *Presenters* aufgerufen, durch welche die Daten des Business Cases ausgeblendet werden, falls der Nutzer nicht die nötigen Rechte hat. Anstelle der Übersicht kommt dann eine Ausweich-Seite mit der Mitteilung *Sie haben keine Rechte, um diesen Bereich einzusehen.* Eine Veranschaulichung dieses Prozesses ist in dem Sequenzdiagramm 79 dargestellt.

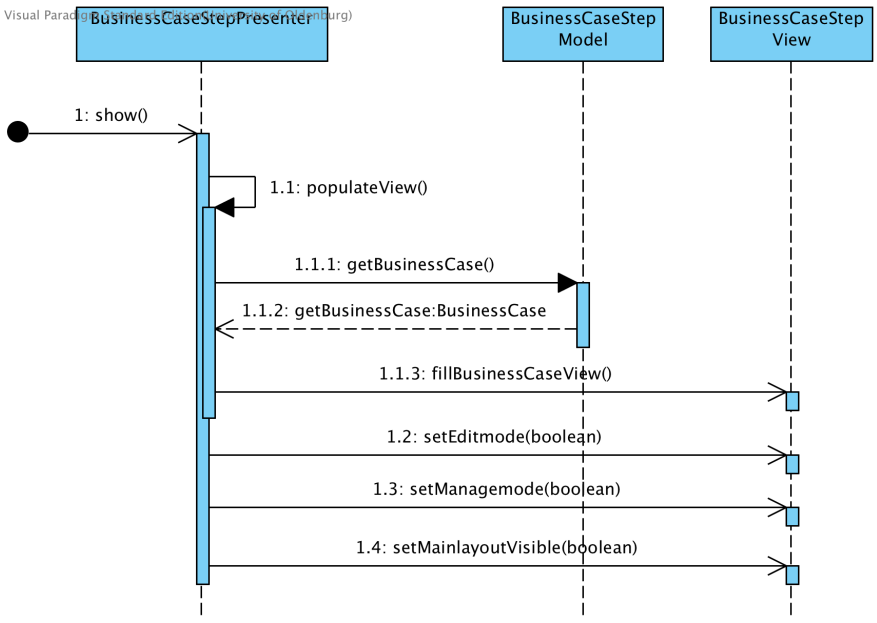


Abbildung 79: Sequenzdiagramm: Business Case anzeigen

6.7.3. Business Case bearbeiten

Durch das Bearbeiten des Business Case wird es ermöglicht alle benötigten Dateien mit diesem zu verknüpfen. Im Gegensatz zu den anderen Komponenten wie Challenge oder Projekt wird der eigentliche Business Case außerhalb der Plattform durchgeführt. Es werden lediglich die Dokumente mit den Ergebnissen für die jeweiligen Unterpunkte bereitgestellt. Dieses ist dadurch begründet, dass die Erstellung eines Business Case je nach Unternehmen variiert.

Oberfläche

Um einen Business Case zu bearbeiten muss nicht, wie bei einem Projekt auf den *Bearbeiten*-Button geklickt werden. Sondern ob ein Business Case editierbar ist, wird nur über den Zustand und die Rechte geregelt. Dies ist ein gewollter Stilbruch, da an einem Business Case nicht eine Vielzahl von Personen sondern nur jeweils ein Controller arbeitet. Somit wäre ein Button mit dem in den Bearbeiten-Modus gewechselt werden kann nur störend. Sofern die Möglichkeit besteht einen Business Case zu bearbeiten, können Dateien mit Hilfe des *Dateien hochladen*-Buttons Dateien hochgeladen und mittel *Speichern*-Button direkt gespeichert werden. Für den Fall, dass alle benötigtem Dateien eines Unterpunktes hinzugefügt wurden, kann ein Haken bei der dazugehörigen `Checkbox` gesetzt werden. Sind alle `Checkboxen` gesetzt, so kann der Button *BC-fertig* gedrückt werden. Daraufhin kann der Entscheider eine Bemerkung in einem Textfeld abgeben und entscheiden ob das Projekt angenommen oder abgelehnt wird. Falls noch Daten fehlen kann er eine Überarbeitung anfordern.

Technische Umsetzung

Da der eigentliche Business Case außerhalb der Plattform durchgeführt wird, gibt es bis auf ein Textfeld für die Schlussfolgerung und Bemerkung keine Felder, die ausgefüllt werden können. Es existiert nur die Möglichkeit zu den verschiedenen Unterpunkten Dateien unterschiedlicher Formate hochzuladen. Auf die technische Umsetzung dieser Komponente wird in Abschnitt 6.14 genauer eingegangen. Ferner können noch die *Checkboxen* der einzelnen Punkte ausgewählt werden. Geschieht dies, wird jeweils das Event `SaveCheckBoxStatusAttemptEvent` gesendet, welches im *Presenter* empfangen wird. Dort wird es verarbeitet und die Änderung weiter an das im *Model* befindliche Business Case Objekt übertragen. Anschließend wird durch die Methode `tryToEditBusinessCase` des *BusinessCaseStepModels* über den `StageService` und `DatabaseService` die Veränderung in die Datenbank übertragen. Danach wird die Methode `handleCheckboxes` der *View* aufgerufen, wodurch die Checkliste an der linken Seite aktualisiert wird. Das Speichern der Schlussfolgerung ist analog hierzu. Es wird lediglich nach dem Betätigen des *Speichern*

Buttons das Event `SaveRecommendationAttemptEvent` gesendet, welches dann im *Presenter* verarbeitet wird. Nachdem alle *Checkboxen* angewählt wurden, wird über die vorher erwähnte Methode `handleCheckboxes` der Button *BC fertig* freigegeben. Eine Betätigung dessen sendet dann das `InformDeciderAttemptEvent` Event, welches wiederum analog im *Presenter* über die `tryToEditBusinessCase` Methode den Zustand des Business Cases speichert und zudem die `show` Methode aufruft. So wird die Oberfläche entsprechend dem neuen Zustand angepasst. Fällt der Entscheider nun ein Urteil, füllt er das Textfeld für die Bemerkung aus und betätigt einen der Entscheider-Buttons, wird das Event `ProcessDecisionAttemptEven` gesendet. Die Verarbeitung dieses Events findet wieder im *Presenter* statt. Dann wird analog der Zustand und die Bemerkung gespeichert und zum Schluss die `show` Methode aufgerufen, um die Oberfläche anzupassen. Zudem wird vorher noch je nach Event ein passendes `ActivityEvent` für den *ActivityStream* gesendet.

6.8. Feedback

Mithilfe von Feedbacks kann ein Verlauf der Umsetzung eines Business Cases verfolgt werden. Es soll ein realistischer Verlauf dieses Vorgangs dargestellt werden und Meinungen zu dem Business Case und den dahinter stehenden Elementen wie Projekt, Lösungsvorschlag und Challenge, festgehalten werden. Das Bearbeiten eines Feedbacks ist in dem Prototyp nicht möglich. Somit soll vermieden werden, dass im Nachhinein Anpassungen vorgenommen werden, die den Verlauf oder die Meinungen verzerren könnten.

6.8.1. Feedback erstellen

Dieser Abschnitt beschäftigt sich mit dem Erstellen eines Feedbacks. Unterschieden wird dabei in Oberfläche und technische Umsetzung.

Oberfläche

Um ein Feedback zu erstellen, muss ein Business Case existieren. In dem *Tab Feedback* steht der *Neues Feedback erstellen* Button zur Verfügung, um dieses zu erstellen. Ebenso können hier alle bisherigen eingetragenen Feedbacks betrachtet werden. Wird diesr Button betrachtet, gelangt man zu einem Popup-Fenster. In diesem ist es möglich, dem Feedback einen Titel zu geben und das eigentliche Feedback abzugeben (vgl. Abbildung 80). Beide Eingabefelder sind Pflichtfelder und müssen ausgefüllt werden, um den *Speichern* Button zu aktivieren.

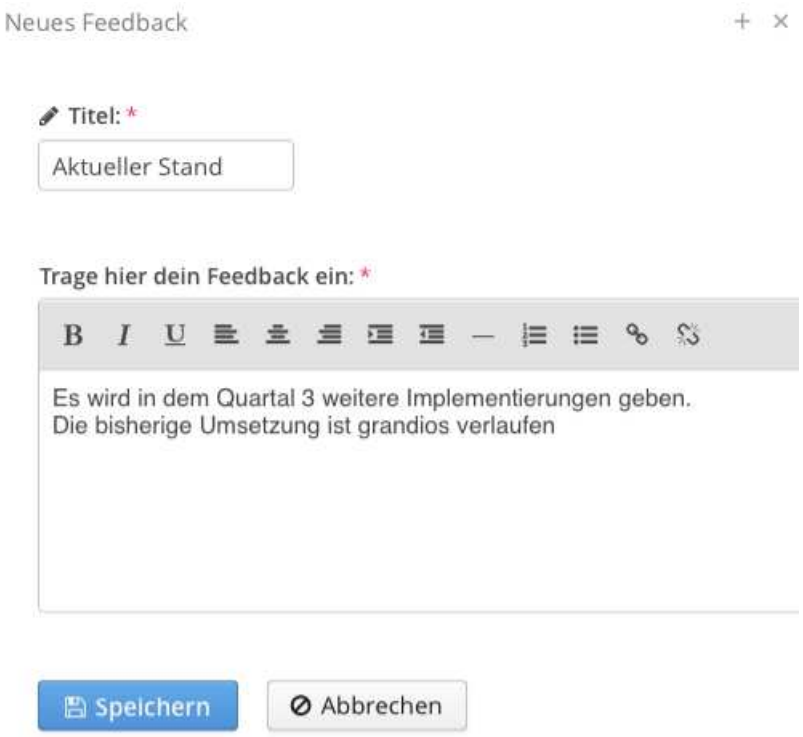


Abbildung 80: Anlegen eines Feedbacks

Technische Umsetzung

Die technische Umsetzung, um ein Feedback zu erstellen, sieht wie folgt aus. Durch das Betätigen des *Feedback erstellen* Buttons wird das Event `AddNewFeedbackEvent` aus der `FeedbackStepView` gesendet und in der Methode `onAddNewFeedback` der `LobbyUI` abgefangen. In dieser Methode wird ein Popup-Fenster, mit dem Inhalt von `AddFeedback`, erstellt. Zuvor muss jedoch der `AddFeedbackPresenter` bekannt gemacht werden, um den Inhalt und deren dahinter liegenden Funktionalitäten zu laden.
Wurden alle Pflichtfelder der Eingabemaske ausgefüllt, so wird der *Speichern* Button aktiviert. Durch das Betätigen dieses Buttons wird das Event `SaveFeedbackAttemptEvent` gesendet und in der Methode `onSaveFeedbackAttempt` des `AddFeedbackPresenter` abgefangen. Innerhalb dieser Methode wird über die Methode `tryToCreateNewFeedback` des *Models* und über den `StageService` versucht ein Feedback zu erstellen (vgl. Abbildung 81). Beim Erstellen eines Feedbacks, innerhalb des `DatabaseServices`, wird zusätzlich ein `AbstractItem`, `founditem`, benötigt (vgl. Quellcode 47). Dieses `AbstractItem` kann ein Projekt, eine Challenge oder ein Business Case sein. In der derzeitigen Umsetzung jedoch, wird nur ein Business Case übergeben, da das Feedback zu diesem abgegeben wird. Der `manager` sucht den entsprechenden Business Case bzw. das `AbstractItem` und setzt diesen als `foundItem`.

```
Feedback feedback = new Feedback(titleField, feedbackText,
    foundUser,
                    date, foundItem);
```

Quelltext 47: Erzeugen eines Feedbacks

Wenn das Feedback erfolgreich erstellt wurde, wird dem Benutzer eine positive Rückmeldung angezeigt, dass das Feedback erfolgreich gespeichert wurde. Zusätzlich wird eine Aktivität erstellt, dass ein Feedback erstellt wurde, welche in dem Aktivitätenstream zu sehen ist. Wurde das Feedback nicht erstellt, bekommt der Benutzer eine negative Rückmeldung. In beiden Fällen wird das Popup-Fenster geschlossen und es wird wieder zu der Feedbackübersicht navigiert.

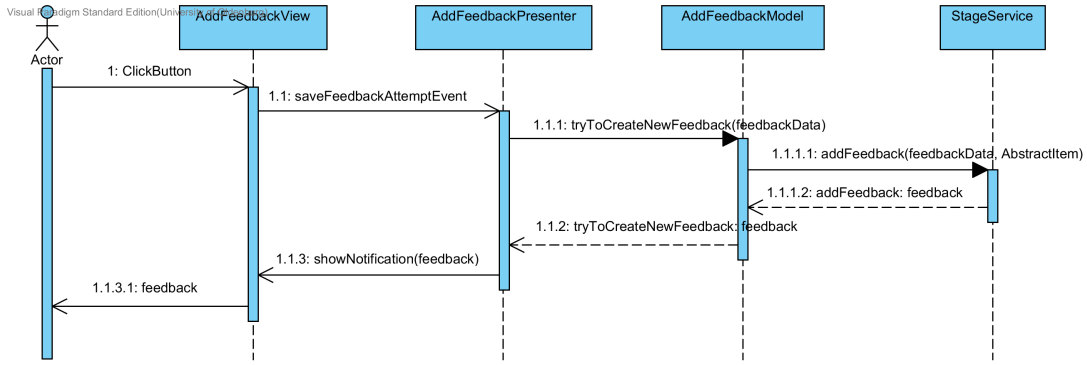


Abbildung 81: Sequenzdiagramm: Feedback erstellen

6.8.2. Feedbackübersicht

Im folgenden wird das Anzeigen eines Feedbacks genauer erläutert.

Oberfläche

Die Feedbackübersicht wird mit Hilfe von *FeedbackPanels* umgesetzt, welche in Form von *StackPanels* dargestellt werden. Für jedes Feedback wird ein neues *FeedbackPanel* erstellt und die Feedbackinformationen in diesem angezeigt. Die Informationen werden dazu in einem Layout angeordnet und mit Hilfe der Methode `setContent` als Inhalt des *FeedbackPanels* gesetzt. Dazu gehören Ersteller, Erstelldatum, Titel und das Feedback. Das neu erstellte Feedback wird unterhalb der bereits vorhandenen *FeedbackPanels* angeordnet.

Abbildung 82: Übersicht vorhandener Feedbacks zu einem Business Case

Technische Umsetzung

Bei dem Wechsel des *Tabs* des `WorkflowTabsheets` wird das Event `WorkflowTabsheetChangeEvent` gesendet. Dabei wird die Position des ausgewählten *Tabs* mit übergeben, damit anschließend ermittelt werden kann, welcher *Tab* gewählt wurde. In der Methode `onWorkflowTabsheetChange` des `WorkflowTabsheetPresenter` wird das Event abgefangen und mit Hilfe einer Switch-Case-Abfrage festgestellt, welcher *Tab* ausgewählt wurde und zu welcher Ansicht navigiert werden soll. Ist der *Tab* an der Position zwei ausgewählt, so wird der Anwender zu der Feedbackübersicht geleitet.
Dort werden alle Feedbacks zu dem entsprechenden Business Case geladen. In dem `FeedbackStepPresenter` wird mit Hilfe der Methode `buildFeedbacks` aus dem *Model*, welches über den `StageService`, alle Feedbacks zu dem Business Case aus der Datenbank abfragt. Gleichzeitig wird die Liste der Feedbacks an die `FeedbackStepView` weiter geleitet und dort gesetzt. Von dort aus wird die Liste an das `FeedbackStepViewDesign` weitergereicht. In der Methode `showAllFeedback` des `FeedbackStepViewDesign` wird für jedes Feedback ein neues *FeedbackPanel* erstellt und sichtbar gemacht (vgl. Abbildung 83). Falls keine Feedbacks vorhanden sind, wird ein Text angezeigt, dass derzeit keine Feedbacks existieren (vgl. Quellcode 48).

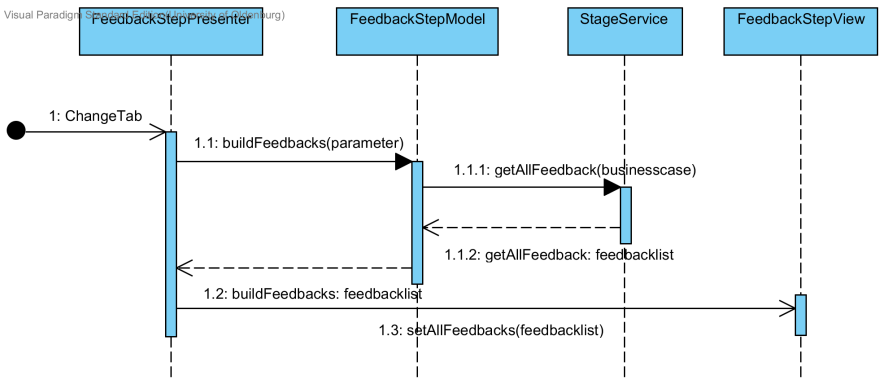


Abbildung 83: Sequenzdiagramm: Feedbackübersicht

```
404 public void showAllFeedback(List<Feedback> feedbackList){
405     if (feedbackList != null && !feedbackList.isEmpty()) {
406 
407             for (Feedback f : feedbackList) {
408                 FeedbackPanel fbPanel = createFeedBackPanel(f
                        );
409 
410                 mainContent.addComponent(fbPanel);
411             }
412         } else {
413 
414 
415             mainContent.addComponent(noFeedbackLabel);
416         }
```

Quelltext 48: Anzeigen der Feedbacks

6.9. Activitenstream

Sobald sich eine Anwender auf der Plattform anmeldet, gelangt er als erstes zum *Aktivitätenstream*. Hier werden ihm zunächst die letzten Aktivitäten angezeigt. Im Folgenden wird zu Beginn auf die Oberfläche und danach die technische Umsetzung genauer eingegangen.

Oberfläche

Der *Aktivitätenstream* ist ein Kernelement der Innovationsmanagementplattform IMPACT. Wie in der Abbildung 84 zu erkennen ist, werden alle Aktivitäten, die innerhalb der Plattform stattfinden, aufgelistet. Dazu zählen:

- Challenge: *akzeptiert*, *beendet*, *kommentiert*, *erstellt*, *bearbeitet* und *bewertet*.
- Lösungsvorschlag: *abgegeben*, *bearbeitet* und *kommentiert*.
- Projekt: *bearbeitet*, *überarbeiten* (aufgrund eines Business Cases), *akzeptiert* (aufgrund eines Business Cases) und *abgelehnt* (aufgrund eines Business Cases).
- Feedback: *abgegeben*.

Es ist möglich, zu den Elementen der Aktivitäten zu navigieren und dort die Änderungen nachzuvollziehen. Jede Aktivität wird in einem eigenen *Panel* dargestellt. Es wird zum

einen der Anwender mit Profilbild und das Datum der Aktivität dargestellt. Zum anderen wird die Aktivität mit einer Satzschablone formuliert und angezeigt. Ein Beispiel für einen solchen Satz wäre, der Anwender *X* hat eine *Bewertung*zu der Challenge *Y* abgegeben. Der Name des Anwenders, der Challenge und die Bewertung wird als *Link* dargestellt. Die aktuellste Aktivität wird an oberster Position angezeigt. Ferner werden beim Öffnen des Aktivitätenstreams die letzten 20 Aktivitäten in die Ansicht geladen.

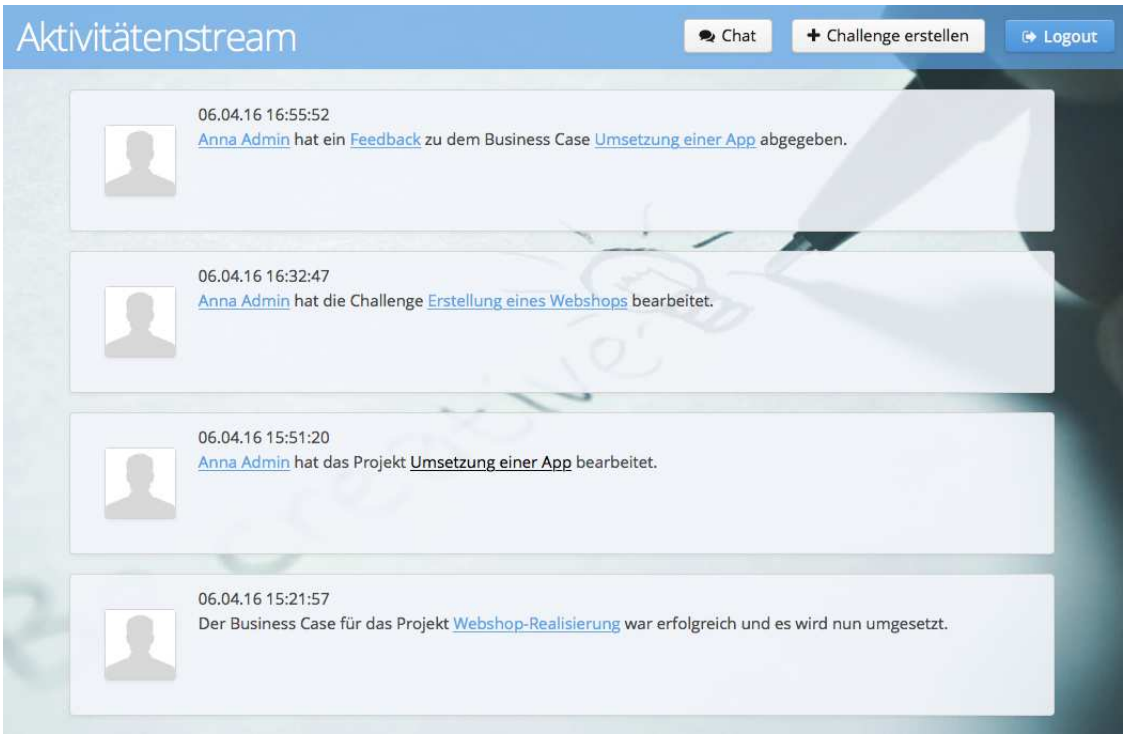


Abbildung 84: Aktivitätenstream in der Plattform

Technische Umsetzung

Im ersten Teil der technischen Umsetzung wird der Prozess erläutert, wie ein `ActivityEvent` an die eingeloggten Teilnehmer der Plattform weitergeleitet wird. Auf diesen Abschnitt folgt dann die eigentliche Verarbeitung und die Anzeige der Aktivitäten im *Aktivitätenstream*.

Wird ein `ActivityEvent` gesendet, so wird dieses durch die Methode `onNewActivityHappend` des `ActivityServices` empfangen. In dieser wird durch den Aufruf der Methode `saveActivityEvent`, welche zum `StageService` gehört, die Aktivität in die Datenbank geschrieben. Ist dies geschehen, wird das `ActivityEvent`-Objekt über die Methode `add` einer Liste hinzugefügt. Diese Liste dient dazu, dass Anwender, die die Plattform betreten, diese nachträglich abrufen können. Bei der Anmeldung am System werden bis zu 20 Aktivitäten in den *Aktivitätenstream* geladen. Danach folgt

eine Schleife, in der an jeden Anwender bzw. an jede *UI* die Aktivität weitergeleitet wird. In diesem Fall werden nur Anwender, welche gerade online sind, adressiert. Sobald sich ein Benutzer anmeldet, wird dieser in eine Liste eingetragen und dementsprechend wieder ausgetragen, falls er sich vom System abmeldet. Das Versenden eines *Events* an eine andere *UI* erfolgt über eine Instanz des `ExecutorService`, der für jede *UI* einen eigenen Thread öffnet, damit die Verarbeitung parallel erfolgen kann. Dieser beschriebene Vorgang wird in dem Sequenzdiagramm 85 zur Verdeutlichung vereinfacht dargestellt.

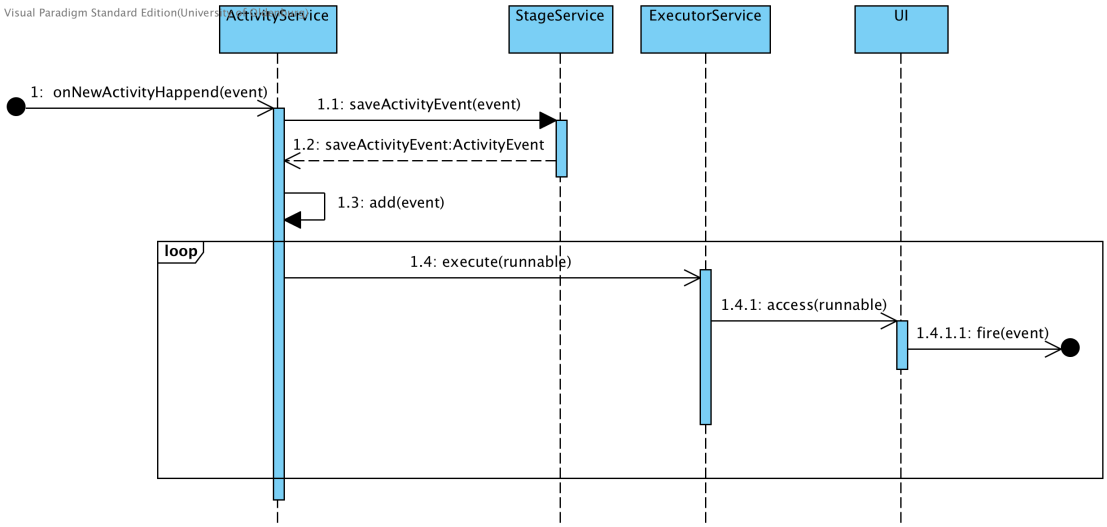


Abbildung 85: Sequenzdiagramm: ActivityEvent an UIs weiterleiten

Das `ActivityEvent` wird nach dem Senden an die *UI* von der Methode `onNewActivityFrom` innerhalb des `ActivityStreamPresenters` empfangen. Diese aktualisiert zuerst die `latestActivityId` im *Model* mittels der Methode `setLatestActivityId` in der Klasse `ActivityStreamModel`. Über die gespeicherte ID wird überprüft, ob die aktuellsten Aktivitäten geladen sind. Danach wird über die Methode `addActivityAsFirst` der `ActivityStreamView` die Aktivität in ein `ActivityPanel` überführt und dem *Aktivitätenstream* an erster Stelle hinzugefügt. Das Umwandeln eines `ActivityEvent` in die grafische Komponente *ActivtyPanel* erfolgt in einer `switch`-Anweisung: Je nach Art der Aktivität werden unterschiedliche Links, in Form von klickbaren Labels, erzeugt. Der Quellcode 49 zeigt einen Ausschnitt aus der Methode `formatActivity`, die diesen Vorgang verdeutlicht.

```
144
145 case CHALLENGE_CREATED:
146
147 if (!event.getItem().isAnonymous()) {
148
149     contentText.addComponent(createUserLink());
150     contentText.addComponent(new Label(
151     " hat die Challenge ", ContentMode.HTML));
152     contentText.addComponent(createChallengeLink());
153     contentText.addComponent(new Label(" erstellt.",
154     ContentMode.HTML));
155
156 } else {
157
158     contentText.addComponent(new Label("Die Challenge ",
            ContentMode.HTML));
159     contentText.addComponent(createChallengeLink());
160     contentText.addComponent(new Label(" wurde erstellt.
            ",ContentMode.HTML));
161     image = new Image("", new ThemeResource("img/profile-pic.
            jpg"));
162 }
163
164 break;
```

Quelltext 49: Ausschnitt aus der Methode formatActivity

Innerhalb der Methode wird unterschieden, ob ein Link zum Profil des Erstellers der Challenge erzeugt wird. Dieses hängt davon ob, ob dieser anonym bleiben möchte. Im Anschluss werden die restlichen *Label* generiert und die Weiterleitung zur erstellten Challenge mittels der Methode `createChallengeLink` hinzugefügt.

6.10. Kommentarfunktion

Die Kommentarfunktion gewährt den Anwendern die Möglichkeit, sich innerhalb der Plattform auszutauschen. Diese steht auf den Ebenen der Challenge (vgl. Abschnitt 6.3), Lösungsvoschlag (vgl. Abschnitt 6.4) und Projekt (vgl. Abschnitt 6.5.1) zur Verfügung. Die Abbildung 86 visualisiert exemplarisch den Aufbau der Kommentarfunktion in einer Challenge.

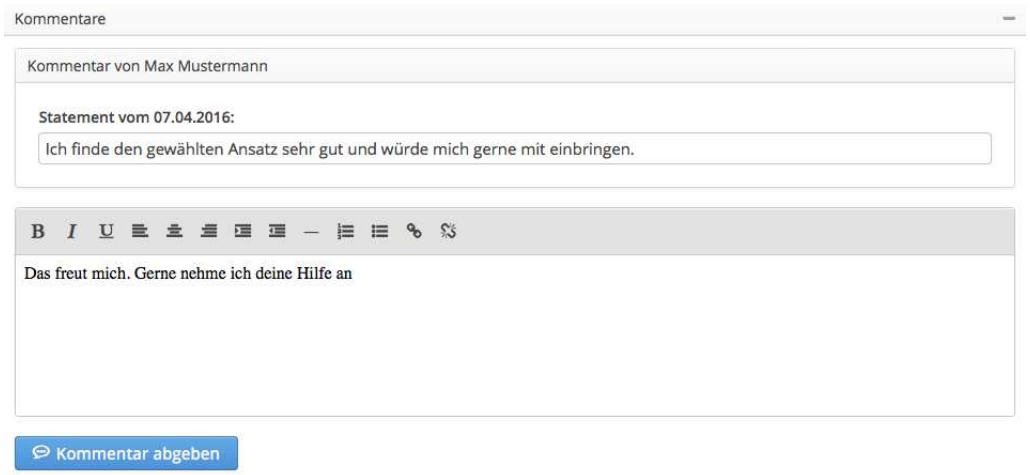


Abbildung 86: Aufbau der Kommentarfunktion

Kommentar erstellen

Verfasst werden Kommentare innerhalb einer *RichTextArea*, um so die Möglichkeit zu bieten, auf Links zu verweisen oder Auszüge besonders hervorzuheben. Über eine, wie in Abschnitt 6.2 beschriebene Validierung wird dabei sichergestellt, dass der Nutzer keine leeren Kommentare abgeben kann. Ist der Kommentar durch den Anwender verfasst, so kann dieser durch Betätigung des entsprechenden Buttons diesen speichern. Dies findet durch einen Aufruf der Methode `SaveNewComment` des `StageServices` statt. Der Quellcode 50 visualisiert dabei den Ablauf. Dabei wird zunächst ein neuer Kommentar anhand der übergebenen Werte erzeugt. Anschließend wird die Transaktion des Managers gestartet, um den Kommentar zu persistieren. Daraufhin ermittelt der Manager anhand der Id das `AbstractItem`, zu dem der Kommentar erstellt wurde und fügt diesen dem `commentSet` hinzu. Abgeschlossen wird die Aktion durch ein `commit` und dem Zurückliefern des Kommentars.

```
1545        Comment comment = new Comment(content, commentedItem,
                createdBy, createdAt, anonymous);
1546        manager.getTransaction().begin();
1547        manager.persist(comment);
1548        manager.flush();
1549        AbstractItem foundItem = manager.find(AbstractItem.
               class,
1550        commentedItem.getId());
1551        foundItem.getCommentSet().add(comment);
1552        manager.getTransaction().commit();
1553        return comment;
```

Quelltext 50: Methode zum Speichern eines Kommentars

Kommentar anzeigen

Entgegen der Speicherung wird zum Anzeigen eine Komponente, das *CommentPanel*, verwendet. Dies dient der einheitlichen Gestaltung von Kommentaren und besteht dabei jeweils aus einem *Panel* und einer *RichTextArea*. Diese erhalten Informationen wie den Namen des Anwenders, von dem der Kommentar verfasst wurde, zu welchem Zeitpunkt die Erstellung stattfand und welche Aussage getätigt wurde. Erstellt werden diese durch die Methode (showComments) (vgl. Quellcode 51), welche bei Veränderungen der Kommentare oder beim erstmaligen Anzeigen des Objektes aufgerufen wird.

```
128 public void showComments(Set<Comment> commentSet) {
129     // leeren des Layouts
130     mainCommentLayout.removeAllComponents();
131     commentsRichTextArea.clear();
132     // für jeden Kommentar zur Challenge Panel erzeugen
133     if (commentSet != null && !commentSet.isEmpty()) {
134         for (Comment c : commentSet) {
135             mainCommentLayout.addComponent(createCommentPanel
                    (c));}
136     } else {
137     // falls keine Kommentare existieren Nutzer Feedback
            geben
138     Label noCommentLabel = new Label(
```

```
139     "Zu dieser Challenge existiert bisher noch kein Kommentar
            . Also sei der Erste, der seinen Kommentar hinterlässt
            !");
140     mainCommentLayout.addComponent(noCommentLabel);}
141     // RichTextArea + Button zum Speichern neuer Kommentare
            erzeugen
142     mainCommentLayout.addComponent(commentsRichTextArea);
143     mainCommentLayout.addComponent(commentButton);}
```

Quelltext 51: Methode zum Anzeigen eines Kommentars

Die Methode setzt zunächst sämtliche Felder zurück und überprüft anschließend, ob Kommentare zu dem entsprechenden `AbstractItem` existieren. Ist das der Fall, so wird für jeden Kommentar ein neues *CommentPanel* erzeugt, welches dem `mainCommentLayout` hinzugefügt wird. Andernfalls wird ein Label gesetzt, um dem Nutzer zu verdeutlichen, dass bisher keine Kommentare (im vorliegenden Bsp. zur Challenge) existieren. Abschließend wird das zuvor erwähnte Feld samt Button zum Erstellen eines Kommentars angefügt.

6.11. Ratingfunktion

Neben Kommentare, ist es dem Anwender möglich, innerhalb der Plattform Bewertungen abzugeben. Dies ist entgegen der Kommentarfunktion (vgl. Abschnitt 6.10) nur auf den Ebenen der Challenge und des Lösungsvorschlags möglich. In den weiteren Prozessschritten ist keine Bewertung durch die Nutzer vorgesehen. Die Abbildung 87 illustriert dabei die Darstellung der Bewertungen.

Abbildung 87: Anzeige der abgegebenen Bewertungen

Bewertung erstellen

Die Abgabe bei einer Bewertung erfolgt, anders als die eines Kommentars, über ein Popup-Fenster. Dies entspricht dem Vorgehen der Erstellung einer Challenge (vgl. Abschnitt 6.3.1). Eine Besonderheit ist es, dass (wie im Sequenzdiagramm 88 dargestellt) zunächst unterschieden wird, ob der Nutzer bereits eine Bewertung zu dem entsprechenden Objekt abgegeben hat. Ist dies der Fall so werden vom `RatingModel` über den `StageService` die Informationen dieser Beurteilung aus der Datenbank abgerufen, um sie anschließend

in der Ansicht darzustellen. Dies ermöglicht es dem Anwender seine Bewertung zu einem späteren Zeitpunkt anzupassen. Sofern dieser seine Angaben editiert und speichert, werden die Daten in der Datenbank angepasst. Das Vorgehen ist analog zu der Bearbeitung einer Challenge (vgl. Abschnitt 6.3).

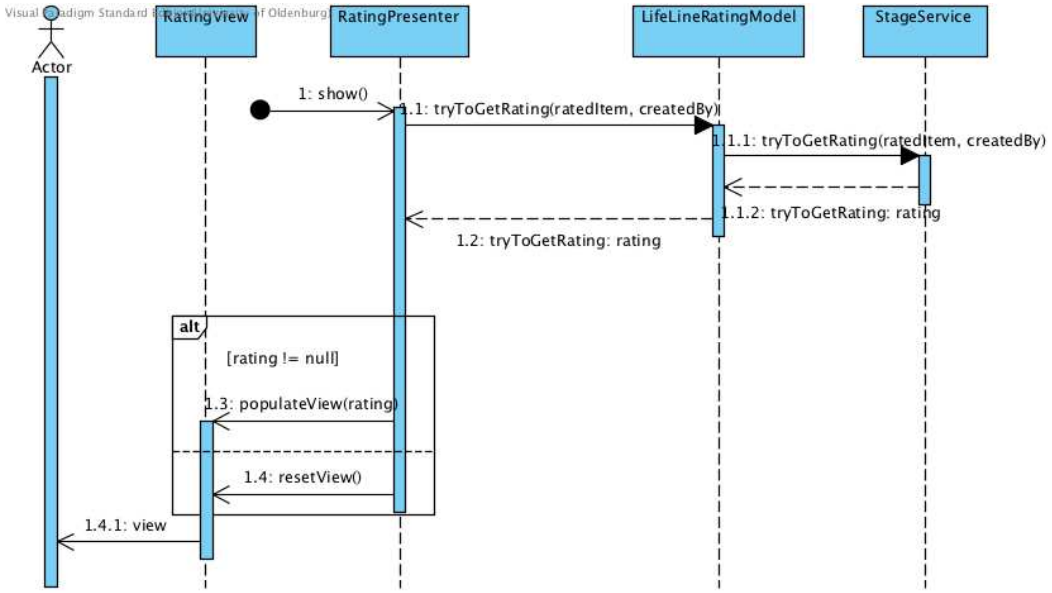


Abbildung 88: Sequenzdiagramm zum Abgeben einer Bewertungen

Sofern der Nutzer keine Bewertung zum Objekt getätigt hat, wird ihm die Möglichkeit geboten, eine neue zu erstellen. Dazu muss er zum einen eine Stern-Bewertung abgeben und diese weiterführend mit einer Begründung versehen. Dazu werden dem Nutzer drei Leitfragen gestellt, in denen er beschreiben soll, was ihm gefällt, was ihm nicht gefällt und weshalb er sich für seine Bewertung entschieden hat. Sind die Felder ausgefüllt, hat der Benutzer die Möglichkeit, seine Bewertung zu speichern. Dies ist auch anonym möglich, wodurch er nicht als Autor zugeordnet werden kann. Die Speicherung erfolgt dabei analog zum Hinzufügen eines Kommentares (vgl. Abschnitt 6.10).

Bewertung anzeigen

Die Darstellung vorhandener Bewertungen entspricht dem Vorgehen für Kommentare. Die einzige Ausnahme ist es, dass ein *RatingPanel* um ein zusätzliches Feld für die Visualisierung der entsprechenden Wertung ergänzt wurde. Dazu wird das Vaadin Addon `RatingStars` verwendet, welches auf Grundlage eines `double`-Wertes eine entsprechende Anzahl von Sternen darstellt. Ein weiterer Unterschied zu den Kommentaren zeigt sich im Aufbau des *RatingPanels*. Dieses ist als *StackPanel* realisiert, was den Vorteil hat, dass einzelne Bewertungen auf- und zugeklappt werden können. Dies verbessert die Übersicht, da gerade Bewertungen durch die Möglichkeit umfangreiche Informationen auszutauschen,

komplex werden können. Die Befüllung mit Inhalten findet analog zu den Kommentare statt, wobei entsprechend die Methode `showRating` verwendet wird. Detaillierte Informationen zum Vorgehen können daher dem Abschnitt 6.10 entnommen werden.
Neben den Einzelbewertungen wird für jedes `AbstractItem` ein Durchschnittswert gebildet. Dieser ist einerseits bei der Anzeige des Objektes im Verbund mit der Anzahl abgegebener Wertungen dargestellt. Weiterhin kann der Durchschnitt in den Übersichten der Challenges und Lösungsvorschläge verwendet werden, um diese zu sortieren.

6.12. Chat-Funktion

Dieser Abschnitt beschäftigt sich mit der Komponente *Chat*, die aus der Übersicht der laufenden Chats, der Komponente `ChatWindow` selbst, der Ansicht zum Hinzufügen eines Benutzers zum Chat und dem `ChatService`, der zur Verteilung der Nachrichten dient, besteht. Ein Chat mit einem anderem Nutzer wird durch den Klick auf die E-Mailadresse in der Nutzerliste gestartet.

Oberfläche

Das eigentliche Chatfenster wird in Abbildung 89 gezeigt. Es besteht aus dem Nachrichtenverlauf, einem Eingabefeld, dem Button *Senden* und dem Button zum Hinzufügen von weiteren Benutzern zum Chat.

Abbildung 89: Chatfenster

Weitere Benutzer werden über ein eigenes *Popup* hinzugefügt. Das Auswahlmenü enthält eine Liste der verfügbaren Nutzer, die gerade online sind. Durch einen Klick auf den Namen werden diese in die darüber liegende Liste hinzugefügt. Über den Button *Entfernen*

neben dem Namen können diese wieder entfernt werden. Der Button *Speichern* fügt die Nutzer dem Chat hinzu und schließt das Fenster. Durch das Hinzufügen einer dritten Person zu einem Chat wird ein neues Gruppenchatfenster geöffnet, zu diesem dann beliebig viele weitere Teilnehmer hinzugefügt werden können. Über den Button *Chat* im Header wird eine Übersicht der laufenden Konversationen geöffnet, mit denen das dazugehörige Chatfenster wieder in den Vordergrund gebracht wird.

Technische Umsetzung

Sobald sich ein Anwender beim System anmeldet, wird die dazugehörige *UI* in der Klasse `BroadcastServiceImpl` vermerkt. Falls sich ein Benutzer mit mehreren Browser-Instanzen anmeldet, wird für jede UI ein Eintrag erstellt. Dies gewährleistet die Weiterleitung der Nachrichten an alle Instanzen. Genauer ist dieser Vorgang im Kapitel *Server Push* 6.2 beschrieben. Wie bereits erwähnt wird ein Chat über einen Klick auf den entsprechenden Nutzer in der Nutzerliste gestartet. Das dadurch erzeugte `OpenChatEvent` wird durch den `ChatPresenterImpl` empfangen und erzeugt über die `ChatViewImpl` eine Instanz des `ChatWindow`, bzw. bringt bereits geöffnete Fenster in den Vordergrund. Alle erzeugten Fenster werden unter einem eindeutigem Schlüssel, dem *Channel* gespeichert. Der Channel setzt sich aus den teilnehmenden Nutzern und einer zufälligen Zahl zusammen. Jede gesendete und empfange Nachricht wird durch ein *ChatMessageEvent* repräsentiert, in dem die eigentliche Nachricht, der *Channel* und der Absender gespeichert wird. Es besteht die Möglichkeit einen Chat mit beliebig vielen Teilnehmen zu führen, dazu ist es möglich zu einem bestehenden Chat weitere Nutzer hinzuzufügen. Über den entsprechenden Button im *ChatWindow* wird ein weiteres PopUp geöffnet, welches über das *Model* eine Liste mit allen zu dem Zeitpunkt angemeldeten Benutzer erhält, die noch zum Chat hinzugefügt werden können. Sobald ein Nutzer ausgewählt ist und auf den *Speichern* Button geklickt wurde, wird ein `AddUsersToChatEvent` Event versendet. Durch dieses werden die Nutzer dem Chat hinzugefügt und die Aktualisierung über das Model dem Service bekannt gemacht.

Versendet ein Benutzer aus der *View* eine Nachricht, indem er auf den Button *Senden* klickt, wird dieses *Event* im *Presenter* abgefangen und die Nachricht an das *Model* weitergeleitet. Das *Model* leitet wiederum das Event an die Klasse `BroadcastServiceImpl` weiter. Über den *Channel* werden die teilnehmenden Benutzer identifiziert und das Event an die entsprechenden UI-Instanzen gesendet, wie im Abschnitt *Server Push* im Kapitel 6.2 beschrieben. Falls ein Nutzer nicht online ist, wird die Nachricht auf dem Server gespeichert. Wenn ein Nutzer sich neu am System anmeldet, werden die gespeicherten Nachrichten dann in die UI Instanz weitergeleitet. Die Abbildung 90 verdeutlicht diesen Ablauf in einem Sequenzdiagramm.

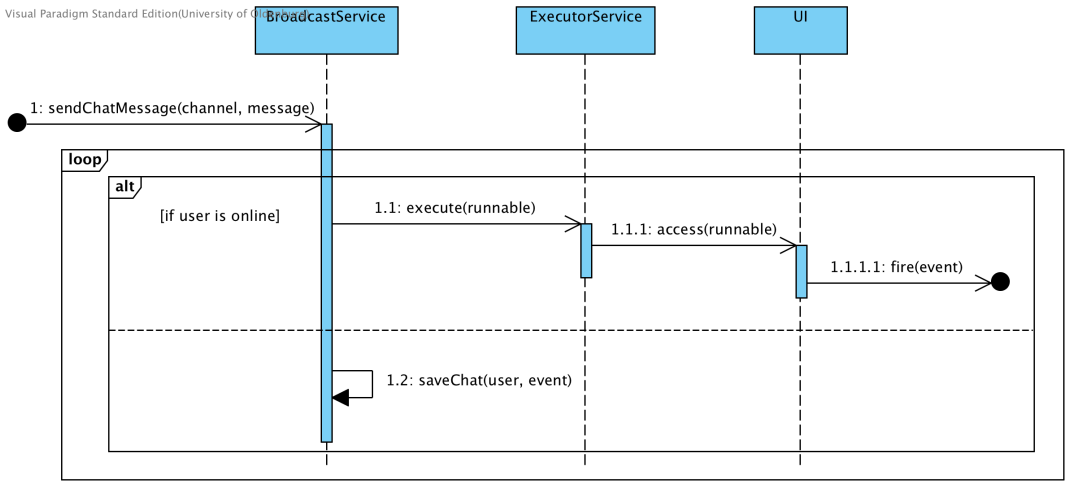


Abbildung 90: Sequenzdiagramm: Weiterleitung einer Chatnachricht

Die Anzeige der Nachrichten in der Komponente *ChatWindow* wird im folgenden beschrieben. Die im `ChatPresenterImpl` empfangenen `ChatMessageEvents` werden über die Methoden `addSentMessage` oder `addReceivedMessage` hinzugefügt. Hierbei wird anhand des Absenders zwischen eingehenden Nachrichten und ausgehenden Nachrichten unterschieden. Ferner enthält das *Model* des Chats ein Liste aller laufender Konversationen, die im Header über den entsprechenden Button angezeigt werden können. So lassen sich geschlossene Instanzen des `ChatWindow` wieder in den Vordergrund bringen.

6.13. Kollaborativer Raum

In diesem Abschnitt wird der *Kollaborative Raum* eines Projektes beschrieben. Dazu erfolgt zunächst eine allgemeine Beschreibung der Komponente. Im weiteren Verlauf wird auf die Oberfläche und die technische Umsetzung eingegangen. Der *Kollaborativer Raum* dient zur Zusammenarbeit und wird über den Button *Kollaborativen Raum öffen* als neues *Window* geöffnet.

Oberfläche

Der *Kollaborative Raum* ist in zwei Bereiche aufgeteilt, das Whiteboard und den Gruppenchat. Der Button *Bild speichern* erlaubt dem Nutzer den aktuellen Zustand des Whiteboards zu speichern. Im Gruppenchat können die Benutzer zu dem Projekt ihre Gedanken austauschen. Der Button *Whiteboard löschen* setzt den Zustand des Whiteboards zurück.

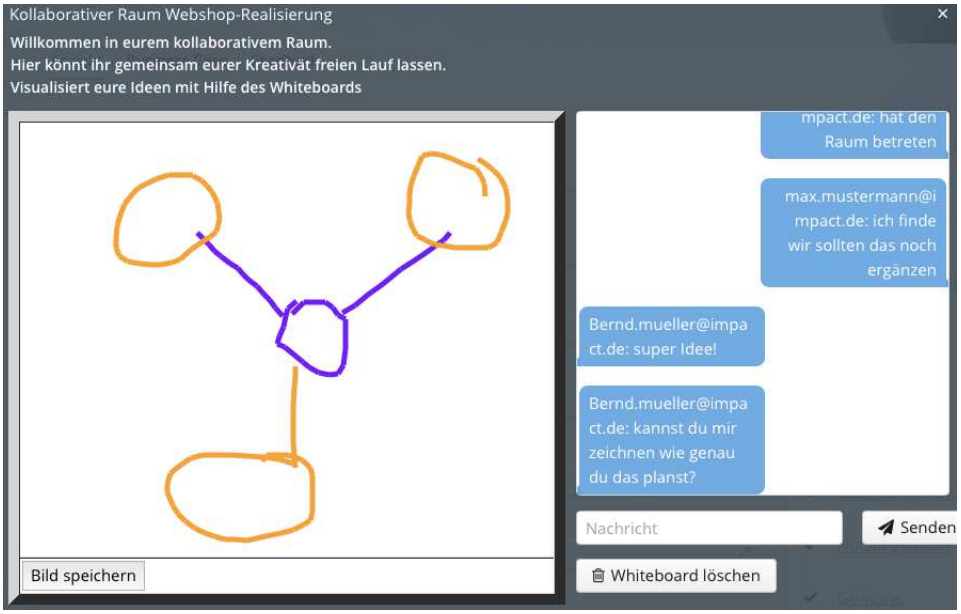


Abbildung 91: Kollaborativer Raum in der Plattform

Technische Umsetzung

Zu jedem Projekt existiert ein *Kollaborativer Raum* mit einem Whiteboard und einem Gruppenchat. Die Komponente besteht aus den Klassen `CollaborativeSpacePresenterImpl`, `CollaborativeSpaceViewImpl`, `CollaborativeSpaceViewDesign` und `CollaborativeSpaceModel`. Serverseitig werden die Daten in der Klasse `CollaborativeSpaceServiceImpl` verarbeitet. Öffnet ein Benutzer den *Kollaborativen Raum*, wird der Benutzer am Gruppenchat angemeldet, die letzten Nachrichten geladen und der aktuelle Zustand des Whitboards abgerufen. Die Anmeldung erfolgt, indem die `UI` in die zu dem Projekt gehörende Liste eingetragen wird. Das Whiteboard wird durch das *Addon CollabSketch* realisiert, die aus einer clientseitgen *Canvas*-Oberfläche und einem serverseitigem Container besteht, das den Zustand selbständig zwischen mehreren *UI*-Instanzen synchronisiert. Sobald der *Kollaborative Raum* geöffnet wird, wird die clientseitige Komponente mit dem Container, der auf dem Server gespeichert ist, verbunden. Für die Realisierung des Gruppenchats enthält die Klasse `CollaborativeSpaceServiceImpl` zu jedem Projekt eine Liste, in der die Nachrichten gespeichert werden. Die Kommunikation erfolgt über `CollaborativeSpaceMessage`-Events, die die ProjektID, den Absender und die eigentliche Nachricht enthalten. Der Quellcode 52 zeigt die Methode `onSentMessage`, die für die Verteilung der Events an die Clients zuständig ist.

```
98  private void onSentMessage(@Observes @View
        CollaborativeSpaceMessage message) {
99
100     //save message
101     messageMap.put(message.getAbstractItemId(), message);
102
103     //fire message
104     for (User user : spaceUserMap.get(message.
            getAbstractItemId())) {
105         for (UI ui : uiMap.get(user)) {
106             executorService.execute(() -> ui
107             .access(() -> collaborativeSpaceMessageSink
108             .fire(message)));
109         }
110     }
111 }
```

Quelltext 52: Verteilung der *Events* an die angemeldeten UIs

Zunächst wird in der Zeile 101 die Nachricht gespeichert, damit Benutzer, die zu einem späterem Zeitpunkt am Gruppenchat teilnehmen, der Konversation folgen können. Ab Zeile 104 erfolgt die Weiterleitung des *Events* an die bereits angemeldeten Benutzer. Das Versenden eines *Events* an eine andere *UI* erfolgt dabei über eine Instanz des `ExecutorService`, der für jede *UI* einen eigenen Thread eröffnet, damit die Verarbeitung parallel erfolgen kann. Verlässt ein Benutzer den *Kollaborativer Raum*, wird er abgemeldet, in dem er aus der Liste entfernt wird.

6.14. Datei Upload

Dieser Abschnitt beschreibt die Umsetzung des *Datei-Uploads* und geht dabei auf die Oberfläche der Komponenten sowie der technischen Umsetzung ein. Im weiteren Verlauf wird beschrieben, wie Dateien mit dem *Document-Management-System* verwaltet werden.

Oberfläche

Über den Button *Datei hinzufügen* kann der Benutzer Dateien auswählen und hochladen. Bereits hoch geladene Dateien werden unter diesem angezeigt. Über den Button *Entfernen* werden die Dateien verworfen.

Abbildung 92: Hochladen von Dateien in einer Challenge

Technische Umsetzung

Der Datei Upload wird durch die Klasse `FileReferenceUploadField`, die vom*Addon EasyUploads*[30] stammt und dessen Klasse `UploadField` implementiert, umgesetzt. Für jede hinzugefügte Datei wird eine Objekt vom Typ `FileReference` erzeugt, dieses enthält folgende Informationen: den Namen, die Dateiendung, die Dateigröße, den *Mime-Type*, den Pfad der Datei, einen Zeitstempel und Informationen über den Benutzer, der die Datei hochgeladen hat. Sobald eine Datei z. B. zu einer *Challenge* hinzugefügt werden soll, wird das Objekt `FileReference` an das *Document- Management-System*, das durch das Interface *DMSService* und die Klasse *DMSServiceImpl* umgesetzt ist, übergeben. Innerhalb des *DMSServices* wird die Datei umbenannt und unter diesem eindeutigem Schlüssel gespeichert. Mittels diesem Schlüssel werden *Attachments* erzeugt, die im Objekt *Challenge* gespeichert werden. Die Dateien werden dann in der Komponente *Dateiliste* angezeigt.

6.15. Datei Liste

Dieser Abschnitt beschreibt die Umsetzung der *Datei-Liste* und geht dabei auf die Oberfläche der Komponenten, als auch der technischen Umsetzung ein. Im weiteren Verlauf wird beschrieben, wie Dateien mit dem *Document-Management-System* heruntergeladen werden.

[30] `https://vaadin.com/directory#!addon/easyuploads`

Oberfläche

Über das Datei-Symbol *Datei hinzufügen* kann der Benutzer Dateien herunterladen. Über den Button *Entfernen* können in der Bearbeitungsansicht Dateien gelöscht werden.

Abbildung 93: Dateiliste in einer Challenge

Technische Umsetzung

Die Komponente `AttachmentListComponent` stellt die technische Umsetzung der Dateiliste dar. Sie enthält eine Liste von `Attachment`-Objekten, dessen Dateiname in der Liste angezeigt wird. Mit einem Klick auf den Namen wird eine *Methode* aufgerufen, die ein `DownloadFileAttemptEvent` absendet, welches den Namen und den eindeutigen Schlüssel der Datei enthält. Dieses *Event* wird in der Klasse `DMSServiceImpl` abgefangen, welches dann über die `static`-Methode `Page.getCurrent().open` ein neues Browser Fenster öffnet und den Download startet.

6.16. Nicht umgesetzte Anforderungen

Im Rahmen der Anforderungsanalyse, deren Erhebung im Kapitel 3 beschrieben wird, wurden alle funktionalen Anforderungen dokumentiert. Diese Anforderungen wurden im Verlauf der technischen Umsetzung weitgehend umgesetzt. Die nicht umgesetzten Anforderungen können in zwei Kategorien unterteilt werden. Zum einem gibt es Funktionalitäten, die aus den Anforderungen abgeleitet wurden, die zeitlich nicht umgesetzt werden konnten. Diese Funktionalitäten werden in dem Kapitel 9 ausführlich beschrieben.
Zum anderen gibt es Anforderungen, die nicht umgesetzt wurden, da sich der Auftraggeber dagegen entschieden hat. Sie könnten aber für die weitere Gestaltung der Anwendung sinnvoll sein. Deshalb werden diese innerhalb dieses Kapitel zusammengefasst. Außerdem wird die Entscheidung der Nichtumsetzung begründet. Damit wird eine abschließende Bewertung der Erreichung aller Anforderungen innerhalb dieses Projektes ermöglicht.

Active Directory

Die Anforderung Benutzer aus einem Active Directory zu lesen und anhand deren Anmeldedaten für die Innovationsmanagementplattform zu nutzen wurde abgelehnt. Die Begründung hierfür ist, dass die Plattform unabhängig als Stand-Alone System nutzbar sein sollte. Es sollte überall und von jedem einsetzbar sein. Daher wurde eine eigene Benutzerverwaltung mit Registrierung für IMPACT eingeführt.

Video Chat

Die Anforderung einen Video Chat in die Plattform einzubinden wurde abgelehnt. Durch die Kommunikation mittels Video Chat können wichtige Informationen verloren gehen, da sie undokumentiert bleiben. Ein asynchrones Arbeiten an einer Challenge wird so erschwert, da an einem Video Chat alle Beteiligten anwesend sein müssten, um den aktuellen Informationsstand zu erfahren. Darüber hinaus ist nicht sicher gestellt, dass alle im Video Chat diskutierten Punkte schriftlich dokumentiert werden.

Gamification Elemente

Einige grundlegenden Gamification Elemente werden umgesetzt. Es wird ein Top 10 Ranking für Challenges erstellt. Es gibt die Möglichkeit ein individuelles Benutzerprofil zu pflegen. Außerdem kann der Benutzer an allen Stellen des Entwicklungsprozesses einer Challenge Feedback in Form von Kommentaren erfassen.
Da aber eine Untersuchung von einsetzbaren Gamification Elementen zu umfangreich war und so eine sinnvolle Integration in die Plattform nicht möglich war, wurden Elemente wie Questreihen, Community Collaboration und Easter Eggs nicht umgesetzt.

7. Projektfortschritt

Im nachfolgenden Kapitel wird der Projektfortschritt des Projektes dokumentiert. Hierzu ist der Ablauf chronologisch dargestellt, Die Abbildung 94 gibt einen groben Überblick zum Verlauf des Projektes.

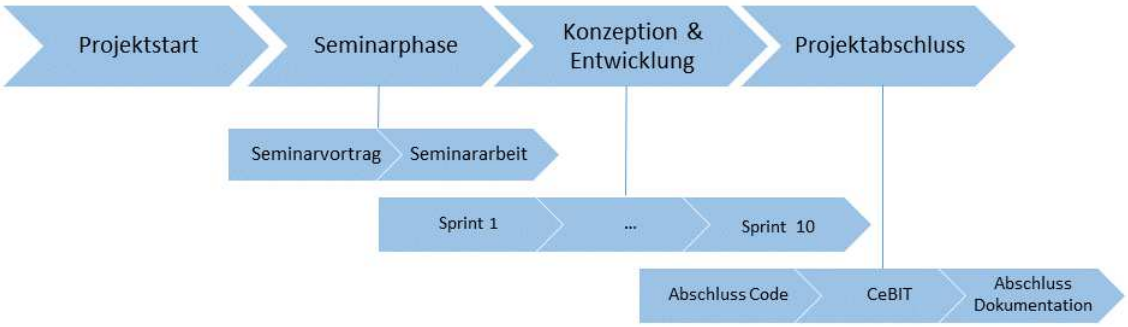


Abbildung 94: Darstellung des Projektfortschritts

Dabei kann das Projekt generell in vier Abschnitte unterteilt werden. Diese sind der Projektstart, die Seminarphase, die Konzeption und Entwicklung, sowie der Projektabschluss. Die detaillierten Inhalte können den einzelnen Kapiteln entnommen werden. Diese fassen die Kernaspekte der Phasen zusammen. Weitere Informationen zu den einzelnen Ereignissen sind in den entsprechenden Protokollen (vgl. Abschnitt A.7) zu entnehmen.

7.1. Projektstart

Den Beginn der Projektgruppe markiert das Kick-Off-Meeting am 31. März 2015. In diesem Rahmen fand eine Begrüßung durch die Betreuer statt, bei welcher das Projekt im Detail erläutert wurde. Zusätzlich wurden die zu besetzenden Rollen vorgestellt. Eine Rolle war der Projektleiter, welche in der Sitzung bestimmt wurde. (Die genaue Rollenverteilung kann im Abschnitt 2.1 eingesehen werden.) Weiterhin fand ein Kennenlernen der Projektgruppen-Mitglieder untereinander in Form einer Vorstellungsrunde statt.
Neben der Delegation erster Zuständigkeiten wurden Arbeitsstandards definiert um angemessene Arbeitsbedingungen zu ermöglichen. Zu diesen zählte u.A. die Kommunikation über E-Mails, die Verwaltung von Protokollen durch Confluence, Fristen für ihre Bereitstellung und weitere Aufgaben. Der Abschluss des Projektstarts ist durch die Vergabe der Themen für die Seminararbeiten definiert.

7.2. Seminarphase

Mit der Verteilung der Ausarbeitungen begann die Seminarphase. Ziel innerhalb dieser war es, dass jedes Gruppenmitglied zu einem Themenschwerpunkt Expertenwissen erlangt, um damit die Gruppe voranzutreiben. Sie lässt sich in zwei Abschnitte untergliedern. Der erste beschreibt dabei Aktivitäten, die im Rahmen der Erstellung der Vorträge durchgeführt wurden. Im zweiten Teil wurde die schriftliche Ausarbeitung erstellt. Zur besseren Übersichtlichkeit wird im weiteren Verlauf zwischen den beiden Phasen unterschieden.

7.2.1. Seminarvortrag (1. April bis 22. Mai 2015)

Neben der Erarbeitung der jeweiligen Präsentationen zu den entsprechenden Themenkomplexen durch die Gruppenmitglieder wurden Aufgaben in den Bereichen innere Organisation, Anforderungsanalyse, Außendarstellung und Technologieentscheidungen bearbeitet. Der letztere beinhaltet dabei u. A. die Auswahl einer Versionsverwaltung-Software, bei welcher sich die Gruppe für Git entschied, sowie die Entscheidung für Java als Programmiersprache. Zudem wurde für den Einsatz von Vaadin als Framework gestimmt. Weitere Entscheidungen sind den entsprechenden Protokollen zu entnehmen.
Hinsichtlich der Anforderungsanalyse wurde beschlossen eine Mitarbeiterbefragung durchzuführen. Diese sollte mittels einer Onlineumfrage durchgeführt werden. Zusätzlich waren Interviews mit dem Themensteller angedacht.
Im Bereich der inneren Organisation wurde eine Ordnerstruktur für die Dokumentation erstellt. Weiterhin wurden zusätzliche Rollen (wie Serveradministrator und Webseitenbeauftragter) vergeben. Gleichzeitig wurde festgelegt wie die Moderationen und das Protokollieren auf Teammitglieder verteilt ist.
Für die Außendarstellung des Projektes wurde eine Webseite angelegt. Dabei wurde beschlossen, dass das Projekttagebuch als Blog auf dieser dargestellt wird. Zusätzlich wurde der Name zu IMPACT[31] geändert, um eine bessere Identifikation mit dem Projekt zu erzeugen.

7.2.2. Seminararbeit (22. Mai bis 22. Juni 2015)

Nach der Präsentation der Vorträge wurden anhand der zugrundeliegenden Erkenntnisse von den Gruppenmitgliedern Ausarbeitungen erstellt, um dieses Wissen zu dokumentieren. Gleichzeitig fanden weiterführende Aktivitäten im Bereich der inneren Organisation, sowie die Anforderungsanalyse statt. Zudem kamen erste Ansätze zur konzeptionellen Umsetzung und Entwicklung.

[31] Innovation Management Plattform to Activate Creative Thoughts

Im Bereich der Anforderungsanalyse wurde dabei die Umfrage für die Mitarbeiterbefragung erstellt. Diese wurde innerhalb der Gruppe diskutiert und nach gewonnenen Erkenntnissen aus einem Pilottest überarbeitet. Die innere Organisation entwickelte sich in dieser Phase so weiter, dass SCRUM als Vorgehensmodell für die Arbeit ausgewählt wurde. Zusätzlich wurde mit Auslauf der Phase die Projektleitung gewechselt. Detaillierte Informationen dazu sind aus den entsprechenden Protokollen nachzuvollziehen.
Neben den zuvor bekannten Aspekten fand die erste konzeptionelle Umsetzung statt. Dies geschah unter Einzug des Wissens, dass in den einzelnen Seminarvorträgen präsentiert wurde. In gemeinsamen Sitzungen wurden sämtliche relevante Aspekte diskutiert, mit dem Ziel ein erstes Konzept zu entwickeln. Dieses stellte die Grundlage für spätere Weiterentwicklungen und ist als Leitfaden anzusehen, auf dessen Grundlage im Rahmen der Entwicklung ein erstes Datenbankmodell erstellt wurde. Dies geschah im Rahmen der Erzeugung eines ersten Testprojektes in Vaadin.
Der Abschluss der Seminarphase ist durch den Beginn des SCRUM-Prozesses definiert, welcher mit der Abgabe der Seminararbeiten startete.

7.3. Konzeption und Entwicklung (22. Juni bis 25. Februar 2016)

Mit dem Ende der Seminarphase startete die Projektgruppe mit der agilen Entwicklung. Hierzu wurde als Vorgehensmodell Scrum gewählt (vgl. Abschnitt2.3). Im weiteren Verlauf dieses Abschnittes wird der jeweilige Projektfortschritt beschrieben. Dazu werden die jeweiligen Sprints dokumentiert und zusammengefasst. Hierbei werden die Planung und Zielsetzung sowie die Ergebnisse des Sprint-Reviews und der abschließenden Retrospektive beschrieben.

7.3.1. Sprint 0

Vom *15. Juni 2015* bis zum *28. Juni 2015* führte die Projektgruppe einen ersten Sprint durch. Ziel dieses Sprints war es, dass die Teammitglieder erste Erfahrungen im Prozessablauf sammeln konnten. Dies ermöglichte allen Teilnehmern die modifizierten Anpassungen im Prozessablauf auf ihre Praxistauglichkeit zu testen. Inhalt dieses ersten Sprints war, dass jedes Projektmitglied eine MVP-Triade anlegt und einen Login nachbaut. Am Ende der zwei Wochen wurden die Ergebnisse innerhalb der Projektgruppe vorgestellt. Die Resultate waren unzureichend, da nicht alle Mitglieder die Aufgabe in Gänze bearbeiten konnten. Als Konsequenz wurde beschlossen, dass für die zukünftigen Sprints das virtuelle Scrumboard im JIRA verstärkt benutzt werden sollte, um so eine gegenseitige Kontrolle und eine allgemeine Übersicht über den Projektfortschritt zu erhalten. Die abschließende Sprintretrospektive wurde im Rahmen dieses Sprints nicht durchgeführt, da zu diesem Zeitpunkt noch kein Bedarf innerhalb der Gruppe bestand.

7.3.2. Sprint 1

Vom *29. Juni 2015* bis zum *16. Juli 2015* wurde im Rahmen der agilen Entwicklung der erste Sprint durchgeführt. Dazu wurde im Vorfeld das Sprint-Estimation-Meeting vollzogen. Dieses Meeting diente zur Festlegung der ersten Aufgaben und zur Koordinierung der einzelnen Tasks. Hierbei kam es darauf an, dass die einzelnen Tasks gemeinsam erarbeitet und die Aufwände geschätzt wurden. Im Rahmen dieses Sprints wurde die Aufwandsschätzung ausgesetzt. Lediglich die beiden Methoden Planing-Poker und Magic-Estimation wurden erläutert. Abgeschlossen wurde dieses Meeting mit der Entscheidung, dass die Aufwände in Zukunft durch Planing-Poker ermittelt werden.
Die Inhalte dieses Sprints waren zum einen konzeptionelle Aufgaben, wie z.B. die Definition der einzelnen Prozessschritte, Erarbeitung eines Konzepts des Farbschemas und die Evaluation der Software Axzure zum Erstellen von Mockups. Ergänzend wurde im Rahmen dieses Sprints die Online-Umfrage getestet und freigeschaltet, um so zusätzliche Anforderungen zu ermitteln.
Zum anderen wurde der geplante Workshop (vgl. Abschnitt 2.6.1) vorbereitet und bei allen Mitgliedern die Entwicklungsumgebung installiert.
Abschließend wurden die konzeptionellen Ergebnisse vorgestellt und zusätzliche Anpassungen für den nächsten Sprint beschlossen. Die intern durchgeführte Retrospektive lieferte ausschließlich positive Ergebnisse, hier wurde insbesondere der erfolgreich durchgeführte Workshop erwähnt. Es wurden keine negativen Punkte aufgeführt, da die Gruppe zu diesem Zeitpunkt noch keine negativen Eindrücke offen kommunizierte.

7.3.3. Sprint 2

Vom *17. Juli 2015* bis zum *30. Juli 2015* wurde der zweite Sprint durchgeführt. Die Planung der einzelnen Task wurde im Rahmen des Estimation-Meetings mit allen Projektgruppenmitgliedern durchgeführt. Die Aufwände, welche als Story Points dargestellt werden, wurden mittel Planing-Poker ermittelt. Hierbei wurde die Problematik der Semantik, in Bezug auf die Größenverhältnisse eines Story Points, allen Mitgliedern bewusst und diskutiert. Als Ergebnis wurde festgelegt, dass ein Story Point keine Zeiteinheit, sondern den benötigten Aufwand symbolisiert. Die Durchführung des Planing-Pokers war zeitintensiv und der Nutzen wurde in Frage gestellt. Daher wurde beschlossen, dass die nächste Aufwandsschätzung mittels Magic-Estimation durchgeführt wird.
Der Inhalt dieses Sprints gliederte sich in die drei Teilbereiche der Anforderungsanalyse, konzeptionellen Entwicklung und technische Umsetzung. Im Rahmen der Anforderungsanalyse wurden die Ergebnisse der Mitarbeiterbefragung aufbereitet und zusammengefasst (vgl. Abschnitt 3.1.2). Die Anpassungen der einzelnen Projektphasen, sowie die Erstellung eines Bewertungssystem bildeten die Grundlage für die konzeptionelle Entwicklung. In der

technischen Umsetzung wurde ein neues MVP-Beispiel erstellt, welches auf der Grundlage des Workshops aufbaute. Dieses sollte allen als Einstiegsorientierung dienen, um so eigene Beispiele abzuleiten.
Abschließend wurden die Ergebnisse vorgestellt, welche als Grundlage für den nächsten Sprint dienten. Die Retrospektive brachte keine neuen Erkenntnisse, welche einen Einfluss auf den nachfolgenden Sprint hatten.

7.3.4. Sprint 3

Vom *31. Juli 2015* bis zum *20. August 2015* zu Beginn des dritten Sprints wurden die ersten großflächigen Tasks zur technischen Umsetzung angelegt. Ziel dieses Estimations-Meeting war es, dass alle Gruppenmitglieder aufbauend auf ihren erlangten Wissen erste Tasks der technischen Umsetzung absolvierten. Die Aufwandsschätzungen wurden mit Magic-Estimation durchgeführt. Die Gruppe stellte hier eine deutliche Verbesserung im Vergleich zum Planing-Poker fest, dies führte zur Entscheidung die weiteren Aufwandsschätzungen mittels Magic-Estimitation zu ermitteln.
Die Inhalte des Sprints gliederten sich in die Themenschwerpunkte konzeptionelle Entwicklung und technische Umsetzung. Konzeptionell wurde das Bewertungssystem weiter entwickelt. Hier wurde sich zum Ende des Sprints für ein 5-Sterne-System entschieden, wie es z.B. bei Amazon verwendet wird (vgl. Abschnitt 4.6 und ProtokollA.7.15). Des Weiteren wurden in Vorbereitung auf ein zukünftiges Rechtemanagement erste Rollen und Aufgaben identifiziert und dokumentiert. Ein weiterer Teil der konzeptionellen Aufgaben war die Erstellung erster Oberflächenentwürfe, welche mit der Software Axure erstellt wurde. Diese Software wurde im Verlauf des Sprints als zu umfangreich bewertet und durch Balsamiq Mockups ersetzt (vgl. ProtokollA.7.16).
Im Zuge der technischen Umsetzung wurden erste eER-Modelle konzipiert, welche anschließend in ein relationales Modell überführt wurden. Zusätzlich wurde an der Umsetzung eines Sessionmanagements und an der Implementierung des Login und der Registrierung gearbeitet.
Innerhalb des Sprintreviews wurden die Ergebnisse vorgestellt und weitere Anpassungen aufgenommen und in die zusätzliche Sprintplanung eingebunden. Innerhalb der Retrospektive wurde der Beginn der umfassenden technischen Umsetzung als positiver Punkt festgestellt. Die Feingranulierung der einzelnen Tasks wurde als unzureichend beschrieben. Dieses Problem wurde in den nachfolgenden Sprints behoben, indem diese atomarer erstellt wurden.

7.3.5. Sprint 4

Vom *21. August 2015* bis zum *17. September 2015* wurde der vierte Sprint durchgeführt. Beim Erstellen der einzelnen Tasks wurde darauf geachtet, dass diese möglichst feingranular waren. Dies sollte der Projektgruppe die Möglichkeit geben den Fortschritt innerhalb des vierten Sprit mittels Burndown-Diagramm nachzuvollziehen. Die Abbildung 95 zeigt den Verlauf des vierten Sprints, welcher optimal durchgeführt wurde. Die Inhalte des Sprints waren im Kontext der technischen Umsetzung die Verbesserung des Logins und der Registrierung. Hierbei wurden die ersten Ansätze der Password und E-Mail Validierung implementiert. Des Weiteren wurden neue MVP-Triaden erstellt, um das Dashboard zu realisieren. Dies beinhaltet u.a. erste Übersichten in Form von Tabellen und Buttons. In Zuge der Tabellen wurden Übersichten der Erstellten Ideen und Probleme hinzugefügt. Die Buttons sollten den Anwender die Möglichkeit zum Anlegen und Bearbeiten der Ideen und Probleme geben.

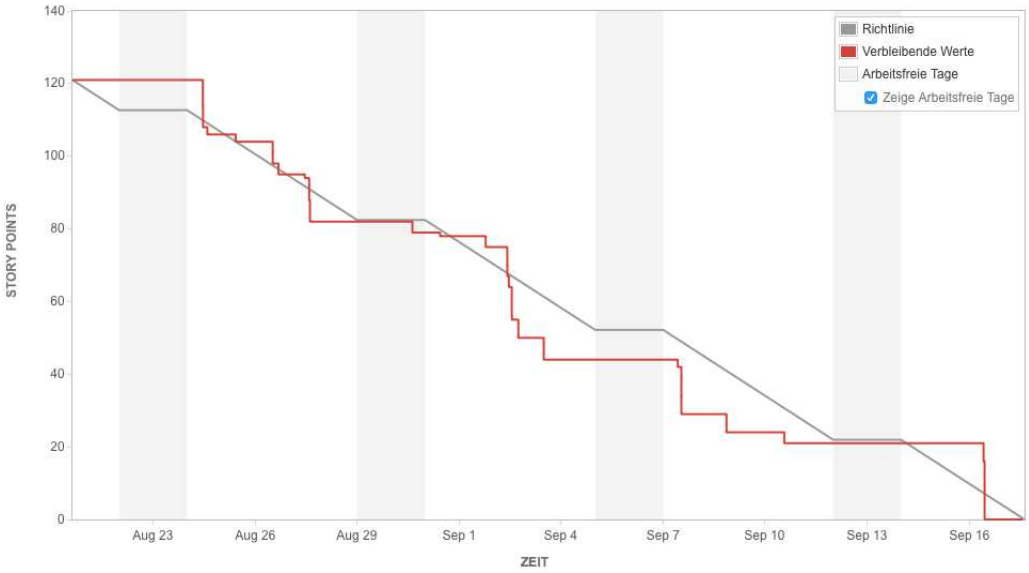


Abbildung 95: Burndown Diagramm des 4. Sprints

In der konzeptionellen Arbeit wurden weitere Mockups erstellt, anhand der die technische Umsetzung erfolgen sollte. In der Abbildung 96 wird das konzipierte Dashboard dargestellt. Ein weiterer Schwerpunkt innerhalb dieses Sprints war die Vorbereitung und Durchführung des Lufthansa Technologietag. Dort wurden zum einen die aktuellen Ergebnisse der Gruppe vorgestellt und zum anderen Anmerkungen der Teilnehmer registriert. Im Zuge des Sprint-Reviews wurden am Ende des Sprints die Ergebnisse vorgestellt. Hierbei wurde u.a. beschlossen, dass ein eigenes Rechtemanagement implementiert und nicht auf ein Active Directory zugegriffen werden soll. Im Rahmen der Retrospektive wurden das Problem

des nicht vorhandenen Gruppenarbeitsraum als größtes Problem identifiziert, sowie der schleppende Fortschritt in der technischen Entwicklung. Deshalb wurde beschlossen, dass die beiden Programmierwochen innerhalb des nächsten Sprints verpflichtend durchgeführt werden und ein Arbeitsraum gebucht wird.

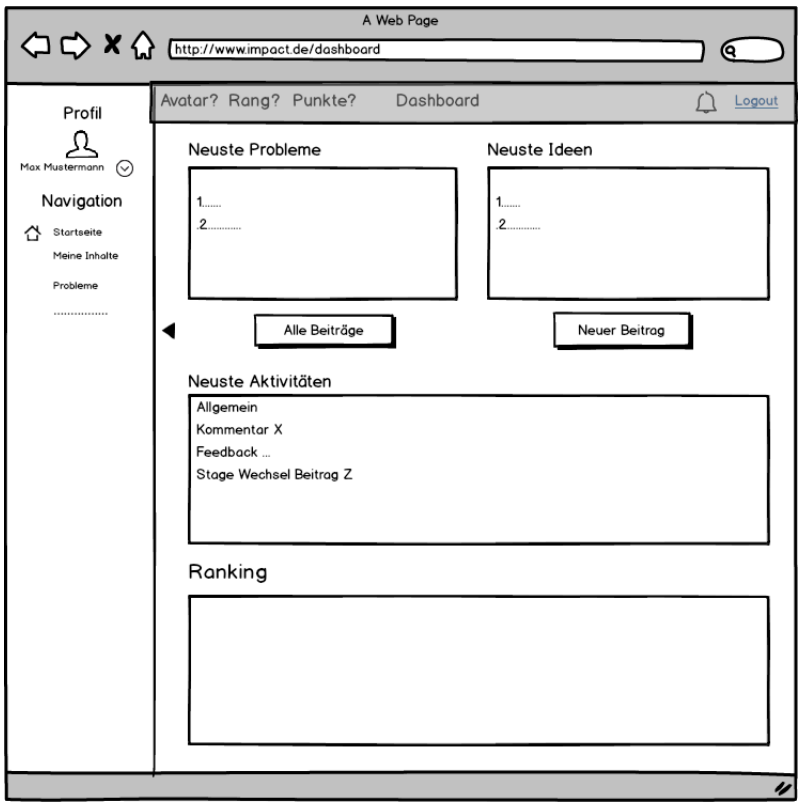


Abbildung 96: Mockup vom Dashboard

7.3.6. Sprint 5

Vom *18. September 2015* bis zum *22. Oktober 2015* wurde der fünfte Sprint durchgeführt. Dieser war durch die Durchführung von zwei obligatorischen Programmierwochen geprägt. Die Länge des Sprints wurde im Zuge dessen auf vier Wochen verlängert. Der Schwerpunkt dieses Sprints lag auf der technischen Umsetzung. Konzeptionelle Aufgaben wurden ausschließlich im Bereich der Weboberfläche durchgeführt. Hierbei wurden erste Farbkonzepte sowie die grafische Gestaltung definiert. Ein wesentlicher Bestandteil der technischen Umsetzung war die Implementierung des Dashboards, Erstellung eines Vorschlages, Umsetzung der Chatfunktion und Notifications. Am Ende der zweiten Programmierwoche wurde ein internes Gruppentreffen durchgeführt. In diesem wurde das bisherige Prozessmodell diskutiert. Am Ende des Treffens wurde das bisherige Prozessmodell grundlegend angepasst (vgl. Abschnitt 4.3.1). Als die wesentlichen Änderungen können einerseits die

Implementierung eines Aktivtätsstreams als zentrales Element der Plattform hervorgehoben werden. Andererseits wurde eine Verringerung der Einstiegshürde, indem der Nutzer ausschließlich den Prozess mit dem Erstellen einer Challenge startet, erreicht. Diese Ergebnisse wurden während des Sprint-Reviews vorgestellt und in der neuen Sprintplanung berücksichtigt. Im Rahmen der Retrospektive wurde der Fortschritt innerhalb des Sprints und die konzeptionelle Anpassung des Prozessmodells hervorgehoben. Die unentschuldigten Fehlzeiten einiger Mitglieder während der Programmierwochen wurden als störende Faktoren beschrieben, welche zukünftig durch explizite Absprachen vermieden werden sollen.

7.3.7. Sprint 6

Vom *23. Oktober 2015* bis zum *12. November 2015* wurde der sechste Sprint der Projektgruppe durchgeführt. Im Rahmen dieses Sprints wurden erstmals keine konzeptionellen Tasks erstellt. Die technische Umsetzung gliederte sich in die Punkte der Implementierung und Oberflächenanpassung. Im Zuge der Implementierung wurden die Erstellung und Bearbeitung einer Challenge, ein Filter- und Bewertungssystem, sowie die Umsetzung des Aktivitätenstreams umgesetzt. Hierbei wurde die zuvor erstellten MVP-Triaden, welche durch die Anpassungen des Prozessmodell obsolet wurden, durch eine Strukturverbesserung (Refactoring) angepasst. Im weiteren Verlauf wurde eine Datenbank aufgesetzt und die ersten Eingaben gespeichert. Die umgesetzten Aufgaben wurden anschließend grafisch bearbeitet, so dass die Oberflächen dem Design-Konzept entsprachen. Abschließend wurden die Ergebnisse vorgestellt und neue Anpassungen bzw. Erweiterungen definiert. Diese wurden in der neuen Sprintplanung berücksichtigt. Im Laufe des sechsten Sprint wurde der Projektgruppe die Möglichkeit gegeben den Projektgruppenraum zu nutzen. Davon versprach sich die Projektgruppe einen positiven Effekt in Bezug auf den weiteren Projektfortschritt.

7.3.8. Sprint 7

Vom *12. November 2015* bis zum *03. Dezember 2015* führte die Projektgruppe den siebten Sprint durch. Innerhalb dieses Sprints gliederten sich die Aufgaben in die Implementierung der einzelnen Prozessschritte. Hierbei wurden Tasks zu den Themen Challenge, Lösungsvorschlag, Aktivitätenstream, Bewertungsfunktion und der Erstellung einer Top-10-Liste festgelegt. Zusätzlich wurde die Möglichkeit der Abgabe von Schlagworten in der Challengeerstellung implementiert. Das Bewertungssystem wurde als 5-Sterne-System umgesetzt, da dieses im Rahmen einer Gruppenbesprechung als erfolgversprechendes bewertet wurde (vgl. Abschnitt 4.6). Im Rahmen dieses Sprints wurden zwei Verantwortliche für die Umsetzung der Designkonventionen benannt. Diese sollten sich mit SCSS beschäftigen

und dies im Rahmen des Projektes umsetzen. Am Ende des siebten Sprints wurden die Ergebnisse erneut vorgestellt. Hierbei stellten alle Gruppenmitglieder fest, dass die Aufgaben, welche innerhalb der Tasks definiert wurden nicht vollständig umgesetzt werden konnten, da diese Tasks zu komplex definiert waren. Die Abbildung 97 zeigt den Verlauf des Burndown-Diagramms, indem der erfolglose Sprint nachvollzogen werden kann. Dies führte zu der Entscheidung den nächsten Sprint-Zeitraum zu verlängern, weil zu diesem Zeitpunkt die Aufsplittung der einzelnen Task in weitere Sub-Tasks als nicht zielführend bewertet wurde. Alle offenen Tasks wurden nicht abgeschlossen und somit in den neuen Sprint übernommen. Im Rahmen der Retrospektive wurde der Scrum-Prozess als zu zeitaufwendig empfunden und sollte für die nachfolgenden Sprints verschlankt werden, im Rahmen der vorhandenen Möglichkeiten (vgl. Abschnitt 2.3). Abschließend wurde ein Termin zur Vorstellung des Prototyps fixiert.

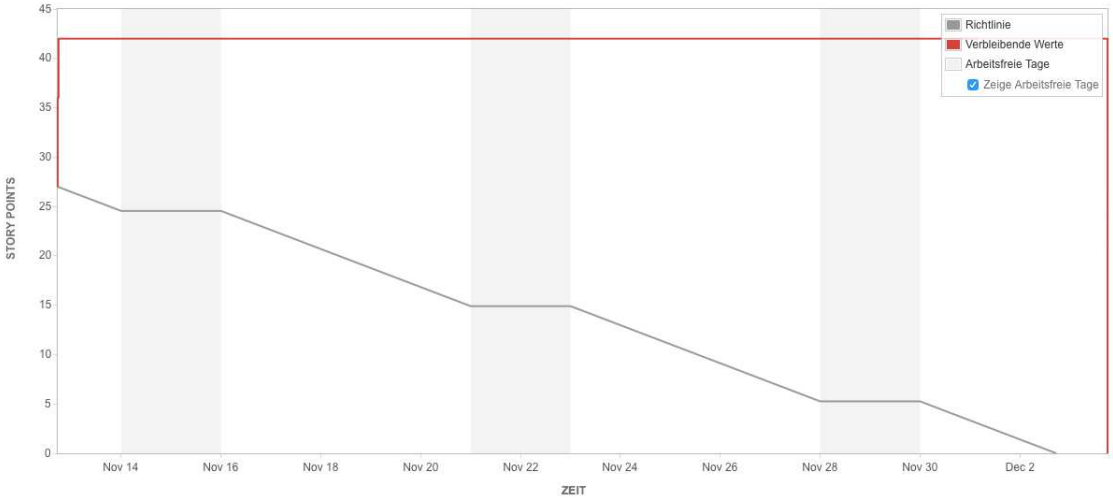


Abbildung 97: Burndown Diagramm des 7. Sprints

7.3.9. Sprint 8

Vom *03. Dezember 2015* bis zum *07. Januar 2016* wurde der achte Sprint durchgeführt. Dieser Sprint wurde durch die terminierte Vorstellung des Prototypen am *17. Dezember 2015* im wesentlichen dominiert. Eine Sprintplanung wurde im Rahmen nicht durchgeführt, da die Tasks aus dem siebten Sprint übernommen wurden. Lediglich eine Anpassung der Aufgaben wurde in einem gemeinsamen Meeting umgesetzt. Die Verantwortlichen für die jeweiligen Aufgaben waren bekannt, so dass die Aufgaben kontinuierlich weiter bearbeitet werden konnten. Im Zuge dieses Sprints war es von Bedeutung, dass die Verantwortlichen Absprachen trafen, um die Übergänge zwischen den jeweiligen Prozessschritten zu implementieren. Dies machte eine enge Zusammenarbeit unabdingbar.

Hierbei war der nun genutzte Projektgruppenraum von entscheidender Bedeutung. Dadurch konnten alle Absprachen direkt getroffen und umgesetzt werden Am *17. Dezember 2015* wurde der Prototyp erfolgreich vorgeführt. Insbesondere die positive Resonanz der Projektgruppenbetreuung und des Auftragsgebers motivierte alle Beteiligten. Anmerkungen wurden aufgenommen und für den nächsten Sprint dokumentiert. Im weiteren Verlauf des Sprints wurde im Schwerpunkt *JavaDocs* erstellt, da diese über den Jahreswechsel erstellt werden sollten. Innerhalb der ersten Jahreswoche wurden die ersten Usability-Tests vorbereitet und durchgeführt (vgl. Abschnitt 8.3). Am Ende des Sprints wurden keine weiteren Ergebnisse vorgestellt, da über die Jahreswende keine Veränderungen am Projekt vorgenommen wurden.

7.3.10. Sprint 9

Vom *07. Januar 2016* bis zum *03. Februar 2016* absolvierte die Projektgruppe den neunten Sprint. Innerhalb dieser Phase konzipierte das gesamte Team feingranulare Tasks. Hierfür wurde ein ausführliches Estimation-Meeting durchgeführt, da die Anmerkungen aus der Prototypvorstellung diskutiert und eingearbeitet werden mussten. Des Weiteren wurden die Ergebnisse der Usabilty-Test eingearbeitet. Im Rahmen dieses Sprints wurde auf der technischen Seite die Umsetzung bzw. Anpassung der Phasen des Projekts und des Business Cases implementiert. Hierbei sollten Checklisten erstellt werden, so dass die Bearbeiter eines Projektes eine direkte Anleitung zum Bearbeiten erhalten. Zusätzlich wurde der Wizard finalisiert, der den Übergang zwischen Lösungsvorschlag und Projekt ermöglicht. Des Weiteren wurde der Datei-Upload implementiert. Der zweite Schwerpunkt lag auf der Durchführung weiterer Usability-Evaluationen. Dazu wurden neue potenzielle Kandidaten eingeladen und weitere Tests durchgeführt. Das exakte Vorgehen, die neuen Aufgabenstellungen, sowie die Ergebnisse der Evaluation werden im Kapitel 8 beschrieben. Abschließend wurden alle Ergebnisse vorgestellt und letzte Anpassungen besprochen. Diese sollten im nachfolgenden finalen Sprint umgesetzt werden. Die im Anhang A.8 dokumentierten Screenshots zeigen den Projektfortschritt zum Ende des neunten Sprints.

7.3.11. Sprint 10

Vom *04. Februar 2016* bis zum *25. Februar 2016* wurde der finale zehnte Sprint durchgeführt. Im Rahmen dieses Sprints wurden die weiteren Funktionen implementiert und die Anpassungen, welche aus den Usability-Test hervorgingen, umgesetzt. Neue Funktionen waren u.a. die Implementierung eines kollaborativen Raums und exklusives sperren im Rahmen des gemeinsamen Arbeitens. Im Zuge des Sprints wurde zur finalen Implementierung mit der Erstellung der Dokumentation begonnen. Mit dem Sprintende startete die Projektabschlussphase, welche im Abschnitt 7.4 beschrieben wird. Die Ergebnisse wurden

im Rahmen des Sprint-Review vorgestellt, indem letzte Anpassungen besprochen wurden, welche zwingend für die CeBIT umgesetzt werden sollten.

7.4. Projektabschluss (25. Februar bis 8. April 2016)

Die Projektabschlussphase beinhaltet drei Kernpunkte: der Abschluss des Quellcodes, sowie Abschluss der Dokumentation und die Präsentation der Ergebnisse auf der CeBIT[32]. Die Phase startete dabei mit der Bestimmung eines neuen Projektleiters. Anschließend fand die Fertigstellung des Produktes statt. Die grundlegenden Zeitpunkte sind dabei der Feature Freeze, welcher auf den *28. Februar 2016* verschoben wurde. Ab diesem Zeitpunkt lag der Fokus darauf Fehler zu beheben um die Plattform für die CeBIT vorzubereiten. Dies endete mit dem Code Freeze am *13. März 2016*.
Während dieser Zeit wurde parallel der Auftritt der Projektgruppe auf der CeBIT vorbereitet. Dazu wurde, neben der Vorbereitung des Standes, ein Video produziert. Dieses diente der Veranschaulichung des zugrundeliegenden Konzeptes. Der Messeaufenthalt ermöglichte den Teammitgliedern durch Gespräche mit Fachbesuchern aus Forschung und Wirtschaft erweiterte Blickwinkel auf den Themenbereich des innerbetrieblichen Innovationsmanagements zu erlangen. Besonders aufschlussreich dabei war die positive Rückmeldung der Standbesucher. Die Erfahrungen wurden im Rahmen der Nachbereitung innerhalb der Gruppe ausgetauscht.
Den Abschluss des Projektes stellt die Fertigstellung der Dokumentation dar. In diesem Rahmen wurden zunächst die eingangs erarbeiteten Konventionen verfeinert. In diesem Schritt wurde zudem die Gliederung überarbeitet. Nachdem die Grundlagen bestimmt waren, wurde gemeinschaftlich das Abschlussdokument erarbeitet. Dieses wurde am *8. April 2015* finalisiert, wodurch das Projekt abgeschlossen wurde.

[32] http://www.cebit.de

8. Softwaretests

Im Rahmen des Softwareentwicklungsprozesses bilden Softwaretests einen elementaren Bestandteil dieses Prozesses. Aufbauend auf der Seminararbeit *Testmanagement* wird im Folgenden das Vorgehen der Projektgruppe, die Testarten und die umgesetzten Testverfahren beschrieben.

8.1. Allgemeines zum Testen

Softwaretests werden im Zuge der Softwareentwicklung durchgeführt, es wird im Allgemeinem zwischen Modul-, Integrations-, System- und Akzeptanztests unterschieden (vgl. [SL12], S.44). Diese vier Teststufen werden im Folgenden beschrieben.

Modultest

Mit dem Modultest wird die erste Teststufe beschrieben. Er wird je Programmiersprache auch als Komponenten-, Unit- oder Klassentest bezeichnet (vgl. [SL12], S.45ff.). Innerhalb eines Modultests werden Funktionen, Methoden und einzelne Klassen getestet (vgl. [BKP+13], S.145).

Integrationstest

Der Integrationstest wird verwendet, um die Zusammenarbeit mehrerer Systemteile zu überprüfen. Dies kann von einzelnen Modulen über Teilsysteme bis hin zum Gesamtsystem geschehen. Hierbei werden die korrekte Verwendung von Schnittstellen, Parametern und die Nutzung gemeinsamer Ressourcen getestet. Der Integrationstest wird wie der Modultest in der Entwicklungsumgebung durchgeführt (vgl. [SL12], S.52ff.).

Systemtest

Während des Systemtests wird das Gesamtsystem unter möglichst realistischen Bedingungen überprüft. Der Systemtest wird in einer Testumgebung durchgeführt, welche der späteren Produktivumgebung möglichst ähnlich ist. Auch die benötigten Testdaten werden den späteren Daten ähneln. Die Überprüfung von funktionalen und nicht funktionalen Anforderungen wird als Ziel des Systemtest definiert (vgl. [SL12], S.60ff.).

Akzeptanztest

Der Akzeptanztest wird auch als Abnahmetest bezeichnet, der das Gesamtsystem in der Produktivumgebung testet. Er bildet die Grundlage zur Abnahme des Systems. Hierbei wird der Fokus auf die Verwendbarkeit des Systems durch den Endnutzer gerichtet (vgl. [SL12], S.64ff.).

Im Zuge der Produktentwicklung werden Modul-, Integrations- und Systemtests erforderlich sein. Die Akzeptanztests werden im Rahmen dieser Entwicklung nicht weiter betrachtet, da das Ziel dieses Projektes auf eine prototypische Entwicklung beschränkt ist. Mit der Technolgieentscheidung für das Build-Automation-Tool *Maven* wird eine Testautomation unter Verwendung des Software-Framework *JUnit* ermöglicht. Im weiteren Verlauf des Projektes wird der Einsatz des Tools *Selenium* eingeplant. Dieses Tool bietet eine Testumgebung für Webanwendungen und kann ebenfalls im Zuge der Testautomation eingesetzt werden. Im Laufe der technischen Umsetzung des Projektes wurde festgelegt, dass sich die Testaktivitäten auf gegenseitige Code Reviews, exploratives Testen und Usability-Test beschränken werden. Diese Entscheidung resultiert aus der Tatsache, dass die entwickelte Software wenig bis keine Elemente besitzt, für die ein JUnit-Test erforderlich ist. Das bedeutet im einzelnen, dass die entwickelte Software über keine aufwendigen Algorithmen verfügt, welche mittels eines Unit-Test getestet werden können. Die Umgesetzten Aufrufketten, welche mittels MVP umgesetzt werden, können direkt im Rahmen der Codeausführung getestet werden. Die Plattform wird als eine Webanwendung umgesetzt und beinhaltet lediglich Felder, deren Eingabe in einer Datenbank gespeichert werden und mittels Listenausgaben visualisiert. Die Programmierung eines Unit-Test stellt im Rahmen des Projektes einen erheblichen Zeitaufwand dar, welcher das hervorgehende Ergebnis nicht rechtfertigt. Es ist nicht nötig die Datenbanklogik zu testen, da diese durch das Objektrelationales Mapping-Framework *ElicpseLink* umgesetzt wird. Auch das Tool *Selenium* wird keinen Einsatz finden, da der Programmcode mittels Vaadin umgesetzt wird. Dies hat zur Folge, dass ein spezielles Addon benötigt wird welches kostenpflichtig und deshalb im Rahmen dieses Projektes nicht eingesetzt werden kann. Diese dargelegten Gründe sollen die Entscheidung der Projektgruppe, sich ausschließlich auf die Testverfahren des Code Reviews, exploratives Testen und Usability-Tests zu fokussieren, erläutern. Die Umsetzung der Testverfahren werden im nachfolgenden Abschnitt 8.2 beschrieben.

8.2. Testverfahren

Im Zuge dieses Kapitels werden die drei Testverfahren, welche im Rahmen des Entwicklungsprozesses durchgeführt werden im Einzelnen beschrieben und die vorhandenen Testergebnisse dokumentiert.

Code-Review

Im Rahmen des Code-Reviews werden die Ergebnisse manuell geprüft. Hierbei kommt es darauf an, dass der Gegenstand der Untersuchung zuvor definiert wurde. Beim Code-Review wird der zuvor definierte Programmabschnitt von einem oder mehreren Gutachtern

gelesen, um so mögliche Fehler zu finden. Der Gutachter kann ebenfalls ein Softwareentwickler sein (vgl. [Mar14], S.124ff.). Im Verlauf der technischen Umsetzung wurden regelmäßige Code-Reviews durchgeführt, dabei ist war es wichtig, dass die entdeckten Fehler nicht nur korrigiert, sondern in Zusammenarbeit mit dem Ersteller behoben wurden. So konnte sichergestellt werden, dass der Entwickler seine Fehler kennt und diese nicht im weiteren Verlauf der Implementierung wiederholte. Das Testverfahren wurde mit dem Beginn der technischen Entwicklung eingeführt und in allen Projektphasen umgesetzt. Dieses Verfahren wurde durch den Einsatz von explorativen Testen komplementiert.

Exploratives Testen

Beim explorativen Testen wird die Software überprüft, ohne dass der Tester den programmierten Abschnitt in allen Einzelheiten kennt. Ziel dieser Methode ist es, dass ein grober Überblick über die Qualität der zu testenden Software erlangt wird. Das Vorgehen beruht auf den Erfahrungen des Testers und baut auf den zuvor erfolgten Tests auf (vgl. [BKP+13], S.117ff.). Im Rahmen der technischen Entwicklung wurde das explorative Testen als Ergänzung der Code-Reviews durchgeführt. Hierbei wurden fertige Teilbereiche von Projektgruppenmitgliedern getestet, welche an der Entwicklung nur indirekt beteiligt waren. Die Ergebnisse dieser Tests wurden den Erstellern präsentiert, wodurch diese die entdeckten Fehler unmittelbar beheben konnten. So konnte die Qualität der entstanden Software gewährleistet werden. Um die Güte des Produktes, insbesondere im Bereich der Nutzbarkeit zu steigern, wurden zusätzliche Usability-Tests durchgeführt. Diese werden im Detail nachfolgend beschrieben.

Usability-Tests

Um die Qualität der entwickelten Plattform sicherzustellen, wurden im Rahmen der Testabdeckung zwei Usability Evaluationen durchgeführt. Die dabei verfolgten Ziele lassen sich zu folgenden Punkten zusammenfassen:

1. Testen der Oberfläche
2. Aufdeckung von Eingabefehlern
3. Steigerung der Übersichtlichkeit
4. Optimierung des Arbeitsflusses
5. Erlangen weiterer Erkenntnisse

Der erste Punkt beschreibt dabei den Ansatz potentielle Fehler, die zuvor nicht bekannt sind, durch Anwendertests aufzudecken. Dazu sollen potentielle Fehleingaben durch ggf. unklare Bezeichnungen erkannt und vermieden werden. Weiterhin soll der allgemeine Arbeitsfluss beobachtet werden um ihn bei Bedarf zu optimieren. Ein Aspekt ist es dabei die

Plattform übersichtlich zu gestalten. Somit sind Probleme im Bereich der Übersichtlichkeit zu analysieren und Ansätze zur Steigerung zu erarbeiten. Neben den beschriebenen Aspekten zielte speziell die erste Evaluation darauf ab weitere Erkenntnisse bezüglich des entwickelten Systems zu generieren. Die Durchführung der ersten Evaluation fand dabei im Anschluss an die Vorstellung des Prototypen statt (vgl. Protokoll A.7.31). Da aus ihren Ergebnissen deutliche Mehrwerte generieren ließen, wurde beschlossen in Verbindung des Prototypen, der für die CeBIT weiterentwickelt wurde, eine zweite (abschließende) Evaluation durchzuführen. Ziel war es dabei zu messen inwieweit sich die Veränderungen am System auf die Erfahrungen der Anwender bei Nutzung auswirken. Das Vorgehen und die Testberichte werden im Abschnitt 8.3 dargelegt.

8.3. Testberichte

Zur Durchführung der Usability Evaluation fand zunächst für jeden Test eine Definition von Aufgaben statt. Diese wurden anschließend in einem Pilot-Test erprobt. Ziel war es unverständliche Aufgaben zu erkennen, um so eine klare Aufgabenstellung gewährleisten zu können. Der Test wurde in einem selbst eingerichteten Usability-Labor durchgeführt. Der Aufbau kann der Abbildung 98 entnommen werden.

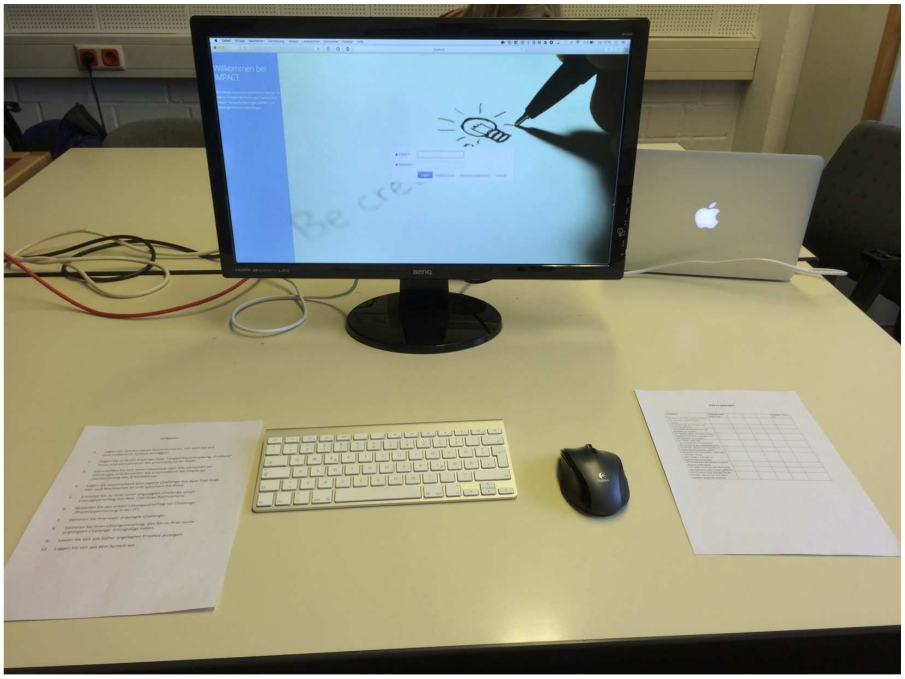


Abbildung 98: Aufbau des Usability-Labors

Der Proband nutzt einen separaten Monitor in Verbindung mit einer Tastatur und Maus als Eingabegeräte. Auf dem eingesetzten Computer wurde ein Gastkonto eingerichtet, um

sicherzustellen, dass der Nutzer nicht von früheren Eingaben oder ähnlichem abgelenkt wird. Zur Durchführung des Tests liegt dem Probanden zudem ein Aufgabenblatt vor, an welchem er sich orientieren muss. Die Usability-Tests wurden dabei jeweils von acht bzw. neun Probanden bearbeitet. Der generelle Ablauf wurde dabei vorher definiert, um sicherzustellen, dass die Ergebnisse auf einer einheitlichen Grundlage erhoben wurden (vgl. Anhang A.6.1). Generell lässt sich die Durchführung in drei Phasen untergliedern. Im ersten Schritt wird der Proband begrüßt, wobei eine Einführung stattfindet. Anschließend wird der Test durchgeführt, wobei dieser beobachtet wird. Abschließend wird durch den Probanden ein Standardfragebogen ausgefüllt. Dieses Vorgehen wurde gewählt, da zum Einen qualitative Daten durch die Beobachtung generiert werden, gleichzeitig jedoch quantitative Informationen für einen späteren Vergleich durch den Fragebogen gewonnen werden. Das genaue Vorgehen innerhalb der Phasen wird im Folgenden detailliert erläutert.

8.3.1. Einführung in den Usability-Test

Zu Beginn der Testdurchführung werden die Probanden begrüßt und in den allgemeinen Ablauf eingeführt. Hierbei werden die Projektaufgabe der Gruppe und allgemeine Informationen zum Projektablauf vorgestellt. Im Zuge dieser Vorstellung erhalten die Testpersonen eine kurze Einweisung in den Ablauf und die Ziele des Usability-Tests. Dabei wird den Probanden erläutert, dass sie während der Durchführung von zwei Personen beobachtet werden, welche ihre Beobachtungen unmittelbar dokumentieren. Zusätzlich werden die Testpersonen darauf aufmerksam gemacht, dass während des Tests eine Bildschirmaufnahme erstellt wird. Des Weiteren werden die Probanden ermutigt laut zu denken, damit die beiden Beobachter ihre Notizen umfassend dokumentieren können. Abschließend zur Einführung in den Test wird den Probanden erläutert, dass sie nach Beendigung des Tests, einen System Usability Scale (SUS)-Bogen unter der Aufsicht des dritten Beobachters, im dazu vorgesehenen Raum beantworten sollen. Nach der Einführung in den Usability-Test, werden den Probanden die wesentlichen Bestandteile der Plattform vorgestellt sowie die einzelnen Aufgaben erläutert. Nachdem alle Fragen geklärt wurden, konnte der der Usability-Test gestartet werden.

8.3.2. Beobachtungen der 1. Usability-Evaluation (Test 1)

Während des ersten Tests der ersten Usability-Evaluation wurden neun Probanden beobachtet. Die Ergebnisse dieser Beobachtung werden hierbei zusammengefasst dargestellt. Dabei ist anzumerken, dass die einzelnen Probanden jeweils zehn Aufgaben abarbeiten mussten. Die erste Spalte soll dabei eine kurze Beschreibung liefern, welche Auffälligkeiten am Verhalten der Probanden bzw. beim Benutzen der Plattform beobachtet worden sind. In der zweiten Spalte sind die Nummern der Probanden angesiedelt, bei denen dieses Verhalten aufgetreten ist. Im vorherigen Schritt wird zusätzlich die jeweilige Aufgabe, die der Proband durchführen musste, aufgezeigt und was die allgemeinen Erkenntnisse der Beobachtung der jeweiligen Aufgabe darstellen.

Erkenntnisse der Aufgabe 1: Legen Sie sich ein neues Nutzerkonto an und loggen Sie sich anschließend im System ein.
Bei den Ergebnissen dieser Aufgabe fiel vereinzelt auf, dass den Testpersonen nicht sofort deutlich war, aus welchen Zeichen sich das Passwort zusammensetzen soll. Da keine genauen Hinweise gegeben waren, die den Nutzer auf bestimmte Eingabemöglichkeiten hinweisen, sondern nur aufzeigen, dass etwas einzutragen ist. Des Weiteren fiel vereinzelt auf, dass die Rückmeldung nach der Registrierung an der falschen Position platziert ist. Die Registrierung wurde ansonsten schnell gefunden und das Konto erstellt. Das Einloggen erfolgte hinterher auch problemlos. Im Allgemeinen traten bei der Erfüllung dieser Aufgabe wenig Probleme und Rückfragen auf.

Beschreibung	**Probanden**
Registrierungslink schnell gefunden	1, 2, 3, 5, 6, 7, 8, 9
Login, bevor ein Nutzerkonto erstellt wurde	4
Feedback nach der Registrierung: • gut, jedoch schlecht positioniert, da Login-Form überlagert wird	5
Login nach Registrierung erfolgte problemlos	alle
Mouseover-Effekte: • wünscht sich schönere Mouse-Overeffekte bei Feldern und Buttons	2
Richtlinien für die Passworteingabe: • nicht eindeutig und klar, da keine einheitlichen Hinweise gegeben sind	alle

Tabelle 10: Beobachtung der 1. Evaluation (Test 1) - Aufgabe 1

Ergebnisse der Aufgabe 2: Ergänzen Sie Ihr Profil um die Tätigkeitsbeschreibung. Tragen Sie dort „Proband“ ein. Anschließend speichern Sie Ihre Änderungen.
Die Ergebnisse der zweiten Aufgabe haben ergeben, dass keiner der Probanden Probleme bei der Bearbeitung aufgezeigt hat. Das Nutzerprofil wurde im Großteil unmittelbar gefunden und die entsprechende Änderung aktualisiert.

Beschreibung	Probanden
Nutzerprofil wurde schnell gefunden und die Änderung wurde aktualisiert	alle
Rückmeldung zur erfolgreichen Aktualisierung:	
• sollte eher als Fenster oder kleinerer Balken erfolgen	5

Tabelle 11: Beobachtung der 1. Evaluation (Test 1) - Aufgabe 2

Ergebnisse der Aufgabe 3: Verschaffen Sie sich einen Überblick zu den vorhandenen Challenges und sehen Sie sich die detaillierte Beschreibung der Challenge „Verbesserung des Arbeitsklimas“ an. Bewerten Sie diese anschließend.
Zu dieser Aufgabe lässt sich sagen, dass alle Probanden keine Probleme aufwiesen, zur Gesamtübersicht aller Challenges zu gelangen. Probleme traten erst auf, wenn es darum ging, zu der einzelnen Challengeübersicht zu gelangen, da sich die Mehrheit der Probanden nicht sicher war, wohin sie mit dem Mauszeiger navigieren müssen. Hinsichtlich der Bewertung hat sich ergeben, dass die Mehrheit der Probanden nicht auf den ersten Blick wusste, wo diese zu der einzelnen Challenge abgeben werden kann, da der Button für die einzelne Bewertung sich nicht sofort als sichtbar erwies, sondern erst nach mehrmaligem scrollen zu sehen war.

Beschreibung	Probanden
Challengeübersicht wurde schnell gefunden	alle
Schnelles Gelangen zur Ansicht der einzelnen Challenge	2, 3, 5
Nicht sicher, ob die Tabelle der Challengeübersicht klickbar ist	1, 4, 6, 7, 8, 9
Nicht sofort ersichtlich, wo eine Challenge bewertet werden kann	1, 3, 4, 5, 6, 8
Rückmeldung beim Abgeben einer Bewertung fehlt	5

Tabelle 12: Beobachtung der 1. Evaluation (Test 1) - Aufgabe 3

Ergebnisse der Aufgabe 4: Legen Sie eine eigene Challenge mit dem Titel „Vorname.Nachname“ (Ihren Vor- und Nachnamen) an und füllen Sie die Pflichtfelder mit einem kurzen Satz aus. Speichern Sie diese Challenge mit dem Button „Speichern und Veröffentlichen“.

Der Button zum Anlegen einer neuen Challenge wurde von allen Probanden ohne Probleme gefunden. Probleme haben sich explizit dahingehend bei der Mehrheit ergeben, dass nicht genau ersichtlich war, wobei es sich um Pflichtfelder gehandelt hat und wobei nicht. Zusätzlich wurde von allen Probanden angemerkt, dass speziell das Titelfeld falsch angebracht ist und an oberster Stelle stehen sollte sowie dass allgemein geschaut werden soll, dass die Elemente besser positioniert werden müssen.

Beschreibung	Probanden
Challenge Erstellen-Button wurde schnell gefunden	alle
Nicht eindeutig, welche Felder als Pflichtfelder deklariert sind	1, 3, 4, 5, 6, 8
Anmerkungen, dass Titelfeld und Elemente nach oben positioniert werden müssen	alle

Tabelle 13: Beobachtung der 1. Evaluation (Test 1) - Aufgabe 4

Ergebnisse der Aufgabe 5: Erstellen Sie zu Ihrer angelegten Challenge einen Lösungsvorschlag mit dem Titel „Nachname“ (Ihr Nachname).

Bei den Ergebnissen dieser Aufgabe wurde ersichtlich, dass der Mehrheit der Testpersonen nicht sofort deutlich wurde, wo der entsprechende Lösungsvorschlag zu erstellen ist, da erst nach mehrmaligem scrollen die Position ermittelt worden ist. Bei dem Erstellen des Lösungsvorschlages selbst traten keine Probleme auf und es gab keine Anmerkungen.

Beschreibung	Probanden
Nicht eindeutig, wo ein Lösungsvorschlag erstellt werden kann	1, 3, 5, 6, 7, 8
Keine Probleme und Anmerkungen bei dem Erstellen eines Lösungsvorschlages	alle

Tabelle 14: Beobachtung der 1. Evaluation (Test 1) - Aufgabe 5

Ergebnisse der Aufgabe 6: Lesen Sie die detaillierte Beschreibung des ersten Lösungsvorschlags zur Challenge „Prozessoptimierung in der IT“ und bewerten Sie diesen.
Hierbei wurde deutlich, dass alle Probanden nicht sofort erkannt haben, wo der bereits erstellte Lösungsvorschlag zu der erstellten Challenge zu finden ist. Erst nach mehrmaligem scrollen und teilweise nach der Einholung einer Hilfestellung der Beobachter wurde der Lösungsvorschlag zu der entsprechenden Challenge gefunden. Das Bewerten hat bei allen Probanden problemlos funktioniert.

Beschreibung	Probanden
Probleme, den Lösungsvorschlag zu der entsprechenden Challenge zu finden	alle
Keine Probleme, den Lösungsvorschlag zu bewerten	alle

Tabelle 15: Beobachtung der 1. Evaluation (Test 1) - Aufgabe 6

Ergebnisse der Aufgabe 7: Editieren Sie Ihre zuvor angelegte Challenge, indem Sie den Titel in „Nachname.Vorname“ ändern.
Bei den Ergebnissen dieser Aufgabe hat sich gezeigt, dass die zuvor angelegte Challenge von allen Probanden in schneller Zeit gefunden wurde. Als es jedoch darum ging, diese bearbeiten zu müssen, so traten bei der Mehrheit Probleme auf. Zum einen wollte die Mehrheit der Probanden in der direkten Ansicht der Challenge diese direkt bearbeiten, nachdem der Button zum Editieren gefunden wurde, verlief das Ganze problemlos.

Beschreibung	Probanden
Probleme, zu der Bearbeitungsansicht der zuvor erstellten Challenge zu gelangen	1, 3, 5, 6, 7, 8
Editieren der Challenge in der Bearbeitungsansicht erfolgte problemlos	alle

Tabelle 16: Beobachtung der 1. Evaluation (Test 1) - Aufgabe 7

Ergebnisse der Aufgabe 8: Editieren Sie Ihren Lösungsvorschlag, den Sie zuvor zu Ihrer Challenge erstellt haben, indem Sie den Titel in „Vorname“ ändern.
Bei den Ergebnissen dieser Aufgabe hat sich gezeigt, dass der zuvor angelegte Lösungsvorschlag von allen Probanden in schneller Zeit gefunden wurde. Als es jedoch darum ging, diesen zu bearbeiten, so hatte die Mehrheit Probleme. Zum einen wollte die Mehrheit der Probanden in der direkten Ansicht des Lösungsvorschlages diesen bearbeiten, nachdem der Button zum Editieren gefunden wurde, verlief das Ganze problemlos.

Beschreibung	Probanden
Probleme, zu der Bearbeitungsansicht des zuvor erstellten Lösungsvorschlages zu gelangen	1, 2, 3, 5, 6, 7, 8
Editieren des Lösungsvorschlages in der Bearbeitungsansicht erfolgte problemlos	alle

Tabelle 17: Beobachtung der 1. Evaluation (Test 1) - Aufgabe 8

Ergebnisse der Aufgabe 9: Lassen Sie sich alle bisher angelegten Projekte anzeigen.
Das Lösen dieser Aufgabe hat keinerlei Probanden Probleme bereitet, da der Navigationspunkt dazu in unmittelbarer Zeit gefunden wurde.

Beschreibung	Probanden
Navigationspunkt zur Projektübersicht schnell gefunden	alle

Tabelle 18: Beobachtung der 1. Evaluation (Test 1) - Aufgabe 9

Ergebnisse der Aufgabe 10: Loggen Sie sich aus dem System aus.
Bei dieser Aufgabe entstanden keine Probleme, da alle Probanden in sehr kurzer Zeit sich aus dem System ausgeloggt haben, ohne lange suchen zu müssen.

Beschreibung	Probanden
Logout-Button oben rechts schnell gefunden und ausgeloggt	1, 2, 4, 5, 6, 7, 8, 9
Logout erfolgte über das Profil, anstatt oben rechts	3

Tabelle 19: Beobachtung der 1. Evaluation (Test 1) - Aufgabe 10

8.3.3. Beobachtungen der 2. Usability-Evaluation (Test 1)

Während des ersten Tests der zweiten Usability-Evaluation wurden ähnlich wie während des ersten Durchlaufs, acht Probanden beobachtet. Die Beobachtungsergebnisse werden im weiteren Verlauf zusammenfassend abgebildet, analog zu der ersten Usability-Evaluation. In diesem Test ging es insbesondere darum, den ersten Test mit neuen Probanden durchzuführen, um einen Einblick zu bekommen, ob die Änderungen der Anmerkungen der ersten Evaluation positive Ergebnisse erzielen oder nicht. In diesem Test mussten von den Testpersonen elf Aufgaben abgearbeitet werden.

Ergebnisse der Aufgabe 1: Legen Sie sich ein neues Nutzerkonto an und loggen Sie sich anschließend im System ein.

Bei den Ergebnissen der Aufgabe ist im Gegensatz zum ersten Test aufgefallen, dass hinsichtlich der Passworteingabe während der Registrierung keine Schwierigkeiten verzeichnet werden konnten. Das Ausfüllen des Formulars innerhalb der Registrierung sowie das anschließende Einloggen wies keine Probleme auf.

Beschreibung	Probanden
Registrierungslink schnell gefunden	alle
Login nach Registrierung erfolgte problemlos	alle
Richtlinien für die Passworteingabe: • nicht eindeutig und klar, da der Hinweis nicht direkt gefunden wurde	3

Tabelle 20: Beobachtung der 2. Evaluation (Test 1) - Aufgabe 1

Ergebnisse der Aufgabe 2: Ergänzen Sie Ihr Profil um die Tätigkeitsbeschreibung. Tragen Sie dort „Proband“ ein. Anschließend speichern Sie Ihre Änderungen.

Die Ergebnisse der zweiten Aufgabe haben aufgezeigt, dass keine Schwierigkeit hinsichtlich der Bearbeitung aufgetreten sind. Ein Unterschied wurde lediglich im Bereich des Profilaufrufs verzeichnet. Der Großteil der Probanden hat das Nutzerprofil auf dem üblichen Weg unter dem Profilbild aufgerufen. Vereinzelt wurde beobachtet, dass das eigene Profil über die Nutzerübersicht erreicht wurde.

Beschreibung	Probanden
Nutzerprofil wurde schnell gefunden und die Änderung wurde aktualisiert	alle
Aufruf des Nutzerprofils erfolgte über die Nutzerübersicht, statt unter dem Profilbild	3, 6

Tabelle 21: Beobachtung der 2. Evaluation (Test 1) - Aufgabe 2

Ergebnisse der Aufgabe 3: Verschaffen Sie sich einen Überblick zu den vorhandenen Challenges und sehen Sie sich die detaillierte Beschreibung der Challenge von Oliver Queen an „Verbesserung des Arbeitsklimas“ an. Bewerten Sie diese anschließend.
Im Bereich dieser Aufgabe konnte beobachtet werden, dass alle Probanden keine Schwierigkeiten hinsichtlich der Bearbeitung aufwiesen. Im Vergleich zu der vorherigen Evaluation konnte herauskristallisiert werden, dass die Navigation zu der einzelnen Challenge keine Probleme verzeichnen konnte. Die Bewertung dieser verlief analog zum Aufruf der Challenge und konnte im Vergleich zu der ersten Evaluation ohne Probleme gelöst werden.

Beschreibung	Probanden
Challengeübersicht wurde schnell gefunden	alle
Schnelles Gelangen zur Ansicht der einzelnen Challenge	1, 2, 3, 5, 6, 7, 8
Nicht sofort ersichtlich, wo eine Challenge bewertet werden kann	8

Tabelle 22: Beobachtung der 2. Evaluation (Test 1) - Aufgabe 3

Ergebnisse der Aufgabe 4: Legen Sie eine eigene Challenge mit dem Titel „Proband“ an und füllen Sie die Pflichtfelder mit einem kurzen Satz aus. Speichern Sie diese Challenge mit dem Button „Speichern“.
Der Button zum Anlegen einer eigenen Challenge wurde von allen Probanden ohne Probleme gefunden. Bezüglich dem Ausfüllen der einzelnen Felder und das abschließende Speichern einer Challenge sind größtenteils keine Schwierigkeiten aufgetreten.

Beschreibung	Probanden
Challenge Erstellen-Button wurde schnell gefunden	alle
Nicht eindeutig, welche Felder ausgefüllt werden müssen	5

Tabelle 23: Beobachtung der 2. Evaluation (Test 1) - Aufgabe 4

Ergebnisse der Aufgabe 5: Erstellen Sie zu Ihrer angelegten Challenge einen Lösungsvorschlag mit dem Titel „ProbandLV“.
Bei den Ergebnissen dieser Aufgabe wurde deutlich, dass im Vergleich zu dem Test der 1. Evaluation die Mehrheit der Probanden den Lösungsvorschlag ohne große Probleme finden konnte, da die Ansicht in diesem Fall so angepasst wurde, dass der Button, um diesen zu erzeugen, im Vergleich zu vorher auf den ersten Blick sichtbar war und so nicht mehr gescrollt werden musste. Das Erstellen des Lösungsvorschlag selbst und das dazugehörige Ausfüllen der Felder verlief problemlos.

Beschreibung	Probanden
Sofort erkennbar, wo ein Lösungsvorschlag erstellt werden kann	1, 2, 3, 5, 6, 7, 8
Keine Probleme und Anmerkungen bei dem Erstellen eines Lösungsvorschlages	alle

Tabelle 24: Beobachtung der 2. Evaluation (Test 1) - Aufgabe 5

Ergebnisse der Aufgabe 6: Lesen Sie die detaillierte Beschreibung des ersten Lösungsvorschlags zur Challenge „Prozessoptimierung in der IT“ und bewerten Sie diesen.
Hierbei wurde deutlich, dass der Großteil der Testpersonen das Navigieren zu dem Lösungsvorschlag ohne Probleme absolvieren konnte.

Beschreibung	Probanden
Keine Probleme, den Lösungsvorschlag zu der entsprechenden Challenge zu finden	2, 3, 5, 6, 7, 8
Keine Probleme, den Lösungsvorschlag zu bewerten	alle

Tabelle 25: Beobachtung der 2. Evaluation (Test 1) - Aufgabe 6

Ergebnisse der Aufgabe 7: Editieren Sie Ihre zuvor angelegte Challenge „Proband“, indem Sie den Titel in „Änderung Proband“ ändern.
Bei den Ergebnissen dieser Aufgabe wurde deutlich, dass die zuvor angelegte Challenge von allen Testpersonen in kurzer Zeit gefunden wurde. Die Bearbeitungssicht wurde im Vergleich zu dem vorherigen Usability-Test, in relativ kurzer Zeit vorgefunden und die Änderung des Titels stellte keine Herausforderung dar.

Beschreibung	Probanden
Probleme, zu der Bearbeitungsansicht der zuvor erstellten Challenge zu gelangen	1, 3, 5, 6, 7, 8
Editieren der Challenge in der Bearbeitungsansicht erfolgte problemlos	alle

Tabelle 26: Beobachtung der 2. Evaluation (Test 1) - Aufgabe 7

Ergebnisse der Aufgabe 8: Editieren Sie Ihren Lösungsvorschlag, den Sie zuvor zu Ihrer Challenge erstellt haben, indem Sie den Titel in „LV Final“ ändern.
Das Bearbeiten dieser Aufgabe verlief größtenteils analog zu vorherigen Aufgabe. Vereinzelt wurde beobachtet, dass die eigene Challenge über den Navigationspunkt *Meine Inhalte* erreicht wurde. Dies ist dem Umstand geschuldet, dass diese Funktion während der ersten 1. Evaluation noch nicht implementiert war.

Beschreibung	Probanden
Keine Probleme, zu der Bearbeitungsansicht des zuvor erstellten Lösungsvorschlages zu gelangen	2, 3, 4, 5, 6, 7, 8
Editieren des Lösungsvorschlages in der Bearbeitungsansicht erfolgte problemlos	alle
Navigation zur Challenge über *Meine Inhalte*	6, 8

Tabelle 27: Beobachtung der 2. Evaluation (Test 1) - Aufgabe 8

Ergebnisse der Aufgabe 9: Lassen Sie sich alle bisher angelegten Projekte anzeigen.

Diese Aufgabe hat äquivalente Beobachtungsergebnisse aufgezeigt, wie in dem vorherigen Usability-Test.

Beschreibung	Probanden
Navigationspunkt zur Projektübersicht schnell gefunden	alle

Tabelle 28: Beobachtung der 2. Evaluation (Test 1) - Aufgabe 9

Ergebnisse der Aufgabe 10: Ändern Sie im Benutzerprofil-¿Persönliche Einstellungen das Theme „Standard“ auf „Facebook“, warten Sie bis es angepasst ist und speichern Sie es anschließend.
Das Bearbeiten dieser Aufgabe hat bei allen Probanden keine Probleme hervorgerufen. Allen Testpersonen war bewusst, wie das eigene Profil und die dementsprechenden Einstellungen zu finden sind.

Beschreibung	Probanden
Nutzerprofil wurde schnell gefunden und die Änderung wurde aktualisiert	alle
Aufruf des Nutzerprofils erfolgte über die Nutzerübersicht, statt unter dem Profilbild	3, 6

Tabelle 29: Beobachtung der 2. Evaluation (Test 1) - Aufgabe 10

Ergebnisse der Aufgabe 11: Loggen Sie sich aus dem System aus.
Die Bearbeitung der elften Aufgabe verlieft analog zu der letzten Aufgabe des vorherigen Usability-Tests. Alle Probanden konnten sich in sehr kurzer Zeit aus dem System ausloggen.

Beschreibung	Probanden
Logout-Button schnell gefunden und ausgeloggt	alle
Logout erfolgte über das Profil, anstatt oben rechts	2

Tabelle 30: Beobachtung der 2. Evaluation (Test 1) - Aufgabe 11

8.3.4. Beobachtungen der 2. Usability-Evaluation (Test 2)

Der zweite Usability-Test der zweiten Usability-Evaluation stellte den letzten Teil der Beobachtungen dar. Dabei wurden acht Probanden beobachtet und die Ergebnisse zusammenfassend dargestellt. Hierbei mussten von den Probanden insgesamt zehn Hauptaufgaben durchgeführt werden. Die restliche Vorgehensweise wurde analog zu den vorherigen Beobachtungsergebnissen vorgenommen.

Ergebnisse der Aufgabe 1: Loggen Sie sich mit dem Benutzer Oliver.Queen@impact.de mit Passwort „1234rewq“ in das System ein.
Bei der Beobachtung dieser Aufgabe konnte verzeichnet werden, dass keine Testperson Probleme hinsichtlich dem Login aufwies, sodass in diesem Fall keine tabellarische Darstellung erfolgt.

Ergebnisse der Aufgabe 2: Kommentieren Sie den Lösungsvorschlag von Matt zur Challenge „Verbesserung des Arbeitsklimas“.
Der Großteil der Probanden konnte in diesem Fall keine Probleme hinsichtlich der Navigation zum Lösungsvorschlag aufweisen. Im Bereich der Kommentarabgabe konnte verzeichnet werden, dass auch hier die Mehrheit der Testpersonen keine Schwierigkeiten aufwies, die Aufgabe durchzuführen.

Beschreibung	Probanden
Lösungsvorschlag:	
• wurde schnell gefunden	1, 2, 3, 5, 7, 8
• nicht sofort klar, wo der Lösungsvorschlag zu finden ist	4, 6
Kommentarabgabe:	
• erfolgte ohne Probleme	3, 4, 5, 6, 7, 8
• Probleme, Kommentarsicht zu finden	1, 2

Tabelle 31: Beobachtung der 2. Evaluation (Test 2) - Aufgabe 2

Ergebnisse der Aufgabe 3: Überführen Sie die Challenge „Verbesserung des Arbeitsklimas“ in ein Projekt.

Bei dem Lösungen dieser Aufgabe konnte beobachtet werden, dass die Mehrheit der Probanden die Überführung der Challenge zum Projekt mit den dazugehörigen Unteraufgaben ohne große Schwierigkeiten bewältigen konnte. Vereinzelt wurde beobachtet, dass bestimmte Testpersonen den Schritt hinsichtlich des Hinzufügens von Nutzern übersprungen haben.

- Entscheiden Sie sich für den zweiten Lösungsvorschlag.
- Geben Sie dem Projekt ihren Namen (Vorname) und fügen Sie eine kurze Beschreibung hinzu.
- Fügen Sie den User Matt und Susan Ihrem Projekt hinzu.
- Starten Sie das Projekt.

Beschreibung	Probanden
Button, um das Projekt zu starten, wurde in kurzer Zeit gefunden	alle
Teilaufgaben innerhalb der Überführung der Challenge:	
• Teilaufgaben wurden ohne Probleme abgearbeitet	1, 2, 3, 5, 6, 8
• Vergessen, neue Nutzer hinzuzufügen	4, 7

Tabelle 32: Beobachtung der 2. Evaluation (Test 2) - Aufgabe 3

Ergebnisse der Aufgabe 4: Sie befinden sich jetzt in Ihrem Projekt. Aktivieren Sie die Bearbeitungssicht.

Die Ergebnisse dieser Aufgabe haben gezeigt, dass ein kleiner Teil der Probanden Probleme aufwies, als es darum ging, die Bearbeitungssicht zu finden. Dabei wurde teilweise versucht, die einzelnen Informationen direkt zu bearbeiten, ohne diese Sicht vorher zu aktivieren.

Nach der Aktivierung der Bearbeitungssicht konnten die einzelnen Teilaufgaben und das anschließende Abspeichern bei dem Großteil der Probanden ohne Probleme durchgeführt werden.

- Überprüfen Sie die Allgemeinen Informationen und bestätigen Sie, dass Projektname und -beschreibung final angelegt sind, indem Sie die CheckBox aktivieren.
- Überprüfen Sie Ihr Projektteam und entfernen Sie den User Matt.
- Legen Sie einen Arbeitsschritt mit dem Namen Anforderungsanalyse hinzu.
- Speichern Sie Ihre Eingaben und aktivieren Sie anschließend die Bearbeitungssicht.

Beschreibung	Probanden
Probleme, zu der Bearbeitungsansicht zu gelangen	1, 3, 5
Keine Probleme bei der Bearbeitung der Teilaufgaben	2, 4, 6, 7, 8

Tabelle 33: Beobachtung der 2. Evaluation (Test 2) - Aufgabe 4

Ergebnisse der Aufgabe 5: Fügen Sie folgende Gewinne dem Projekt hinzu.
Das Hinzufügen der Gewinne stellte bei allen Probanden keine Probleme dar, sodass hier keine tabellarische Aufzählung erfolgt.

- Projektgewinntitel: *ABC*
- Gewinneinsparung: *1234*
- Gewinnbegründung: *DEF*
- Speichern Sie Ihre Eingaben

Ergebnisse der Aufgabe 6: Überprüfen Sie die Checkliste. Sollten noch offene Punkte angezeigt werden, so bestätigen Sie, dass diese final angelegt sind und speichern Sie Ihre Eingaben.
Bezüglich der sechsten Aufgabe konnte beobachtet werden, dass die Hälfte der Probanden bei der Abarbeitung der Checkliste kleine Probleme aufwies. Nicht allen Probanden war sofort ersichtlich, wo die einzelnen Punkte innerhalb der Checkliste zu finden sind und wie die offenen Punkte final angelegt werden sollen.

Beschreibung	Probanden
Probleme, Checkliste abzuarbeiten und zu finalisieren	1, 2, 4, 7
Überprüfen der Checkliste und das finalisieren stellten keine Probleme dar	3, 5, 6, 8

Tabelle 34: Beobachtung der 2. Evaluation (Test 2) - Aufgabe 6

Ergebnisse der Aufgabe 7: Überführen Sie das Projekt in einen Business Case mit dem entsprechenden Button.
Das Überführen des Projekts in einen Business Case stellte bei allen Probanden keine Schwierigkeiten dar, sodass hier keine tabellarische Aufzählung aufgezeigt wird.

Ergebnisse der Aufgabe 8: Loggen Sie sich aus und melden sich als Administrator mit dem Nutzer a und dem Passwort a ein.
Das Lösen dieser Aufgabe erfolgte analog zur siebten Aufgabe, sodass hier keine detaillierte tabellarische Auflistung erfolgt.

Ergebnisse der Aufgabe 9: Suchen Sie den von Ihnen zuvor erstellten Business Case und laden Sie die Datei „UsabilityTest“ in das Feld „Nutzungsdauer“ des Business Cases hoch.
Bei dieser Aufgabe konnten bei allen Probanden ebenfalls keine Probleme verzeichnet werden.

Beschreibung	Probanden
Nutzerprofil wurde schnell gefunden und die Änderung wurde aktualisiert	alle
Rückmeldung zur erfolgreichen Aktualisierung:	
• sollte eher als Fenster oder kleinerer Balken erfolgen	5

Tabelle 35: Beobachtung der 2. Evaluation (Test 2) - Aufgabe 9

Ergebnisse der Aufgabe 10: Loggen Sie sich anschließend aus dem System aus.
Bei dieser Aufgabe entstanden keine Probleme, da alle Probanden in sehr kurzer Zeit das Ausloggen aus dem System durchgeführt haben, ohne lange suchen zu müssen.

8.3.5. System Usability Scale

Zur Erhebung quantitativer Daten im Umfang der Tests wurden die Probanden nach Beendigung der Durchführung gebeten einen Fragebogen auszufüllen. Dazu wurde der System Usability Scale (SUS) eingesetzt. Der SUS ist eine aus zehn Punkten bestehende Skala,

die dazu genutzt werden kann, eine allgemeine Einschätzung der subjektiven Usability-Wahrnehmung zu ermitteln (vgl. [Bro96] S.3). Dabei wird anhand der Antworten ein fester Punktwert ermittelt, durch welchen unterschiedliche Realisierungen vergleichbar werden. Die Ergebnisse der Befragungen sind in der Tabelle 36 entsprechend für jeden Test aufgeführt. Durch die Liste werden dabei die Gesamtergebnisse aller Fragebögen zusammengefasst, wobei das Zustandekommen den einzelnen Unterlagen im Anhang A.6 entnommen werden kann. Anzumerken ist dabei, dass beim ersten Test neun Probanden befragt wurden, mit dem Ziel, die Wahrscheinlichkeit zumindest acht dieser für einen späteren Test erneut befragen zu können, möglichst hoch zu halten. Der zusätzliche Proband sorgt damit für eine Verschiebung des Mittelwertes und Medians, wobei dieser mit ungefähr einem Punkt eine äußerst geringe Abweichung darstellt, so dass es nicht berücksichtigt wird.

Punktwert	1. Evaluation (Test 1)	2. Evaluation (Test 1)	2. Evaluation (Test 2)
Proband 1	57,50	95,00	85,00
Proband 2	97,50	97,50	97,50
Proband 3	85,00	100,00	62,50
Proband 4	95,00	90,00	95,00
Proband5	62,50	80,00	95,00
Proband 6	90,00	95,00	72,50
Proband 7	97,50	75,00	87,50
Proband 8	87,50	92,50	85,00
Proband 9	75,00	-	-
Mittelwert	83,06	90,63	85,00
Median	87,50	93,75	86,25

Tabelle 36: Ergebnisse der Evaluation

Bei Betrachtung der Werte ist festzuhalten, dass allgemein mit über 80 Punkten ein positiver Wert erreicht wird. Wobei die Diskrepanz zwischen Höchst- und Minimalwert beim ersten Durchlauf am größten ausfällt. Im Vergleich zur zweiten Evaluation wurden demnach beim ersten Test weniger Punkte erreicht. Die Abbildung 99 veranschaulicht, dass durch die getroffenen Anpassungen allgemein eine Verbesserung bei der Bearbeitung ermittelt wird.

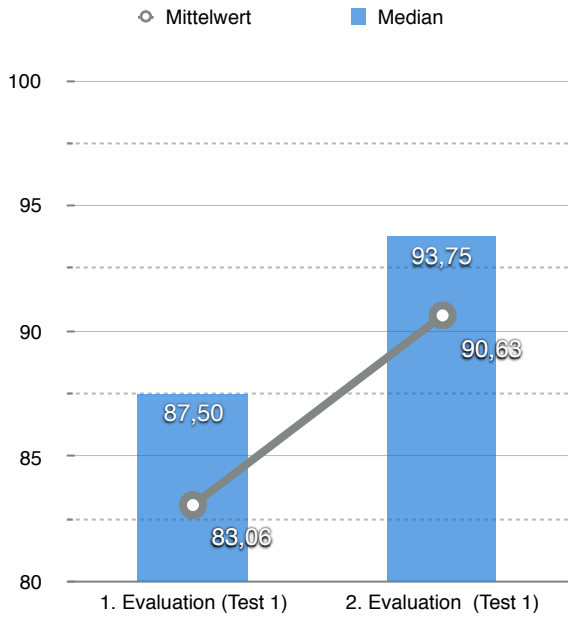


Abbildung 99: Vergleich der Ergebnisse des ersten Tests

Um genauer ermitteln zu können, welche Änderungen zu der Verbesserung führen, sind ferner die zwei Probanden mit den schlechtesten abgegebenen Bewertungen erneut dazu eingeladen worden, den ersten Test durchzuführen. Diese Ergebnisse sind in Abbildung 100 visualisiert.

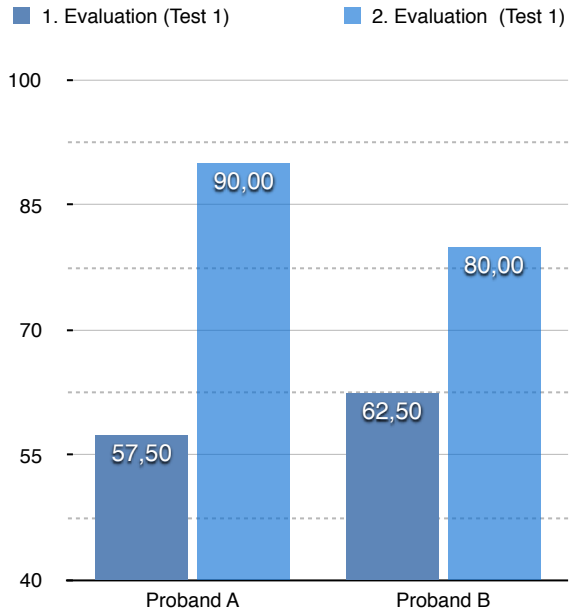


Abbildung 100: Veränderung der Ergebnisse ausgewählter Probanden zum ersten Tests

Diese veranschaulicht eine deutliche Verbesserung, was auch durch den Aussagen der Probanden bestätigt ist. So wechseln die Anmerkungen, von das System intuitiver zu gestalten bis hin zu positiven Meinungen, in denen beschrieben wird, dass die Anpassungen zu einer deutlichen Verbesserung des Arbeitsflusses geführt haben. Auch wenn zu berücksichtigen ist, dass die Anwender durch eine zweite Durchführung bereits mit dem System bekannt sind, kann behauptet werden, dass die Änderungen zu einer deutlichen Verbesserung führen.
Neben den Aufgaben des ersten Testes fand eine Untersuchung weiterführender Funktionen im System statt. Diese wurden direkt unter Berücksichtigung der Erkenntnisse der ersten Evaluation entwickelt. Diese sind als komplexer einzustufen, da sie tiefer in den zugrundeliegenden Prozess hinein greifen. Die Abbildung 101 stellt die Endergebnisse gegenüber.

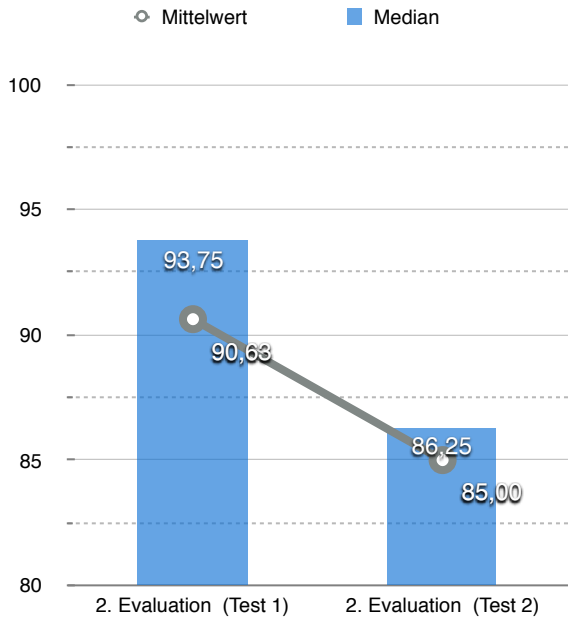


Abbildung 101: Vergleich der Ergebnisse des ersten und zweiten Tests

Dabei ist festzustellen, dass allgemein ein Rückgang der Punkte erkennbar ist, wobei mit über 80 Punkten weiterhin ein positiver Wert erreicht wird. Ursachen für diesen Verlust sind u.a. der größere Funktionsumfang, welcher zu einer beeinträchtigten Übersichtlichkeit führen kann. Zudem sind im Rahmen des zweiten Tests zusätzliche Funktionalitäten hinzugekommen. Diese lagen im ersten Test nicht in einer vergleichbaren Form vor, weshalb keine Erkenntnisse zu diesen Punkten gewonnen werden konnten.
Zusammenfassend zeigen die Werte, dass die Plattform allgemein gute Ergebnisse liefert. Die Anpassungen führen dabei zu einer Optimierung des Arbeitsflusses.

9. Future Works

Die Plattform IMPACT kann zum aktuellen Zeitpunkt bereits als fertiges Produkt betrachtet werden. Trotz dessen können noch eine Reihe von Erweiterungen hinzugefügt werden, um höhere Effektivität und Effizienz beim Arbeiten mit der Plattform zu ermöglichen. Mögliche Erweiterungen werden im folgenden aufgezählt.

Hilfestellung für den Nutzer

Eine mögliche Erweiterung stellt der Bereich der Hilfestellungen für die Nutzer dar. So können bspw. eine Reihe von Kurzvideos erstellt werden, welche einzelne Funktionen, Möglichkeiten und Abläufe zeigen können. Auf diese Weise können Funktionen gezeigt werden, welche weniger intuitiv sind.
Eine weitere Hilfestellung könnte dadurch gegeben werden, einen Bereich für Antworten auf häufig gestellte Fragen (FAQ) anzubieten. Auf diese Weise könnten Nutzer schneller mögliche Antworten auf ihre Fragen erhalten, andererseits würde der Support der Anwendung entlastet werden.
Für die Schaffung eines einheitlichen Standard bei der Einführung in das System könnte als weitere Hilfestellung standardisierte Abläufe und Schulungsunterlagen bereitgestellt werden. Dies trägt dazu bei, dass neue Anwender schnell und einfach die wichtigsten Funktionen kennenlernen. Durch die zusätzliche Schaffung einer Testumgebung könnten die neuen Anwender testen, ohne dabei ins Produktivsystem einzugreifen.

Ausbau der Kommentarfunktion

In der aktuellen Version der Anwendung besteht die Möglichkeit, Kommentare abzugeben, beispielsweise für Challenges oder Projekte. Über diese Funktion hinaus könnte die Möglichkeit geschaffen werden, bestehende Kommentare anderer Nutzer zu kommentieren. Es gibt verschiedene Darstellungsformen für diese Funktion. Zum einen könnte eine Baumstruktur geschaffen werden, sodass Antworten auf einen Kommentar direkt darunter erscheinen können. Zum anderen können die Originalkommentare als Zitat in einem eigenen Kommentar dargestellt werden. Als weiteren Ausbau der Kommentarfunktion kann die Möglichkeit implementiert werden, bestehende Kommentare zu bewerten. Auf diese Weise könnte man dem Nutzer eine sehr effiziente Möglichkeit geben, Zustimmungen bzw. Ablehnungen der Kommentare Ausdruck zu verleihen.

Backend für Administratoren

Wie bereits in Kapitel A.2 beschrieben, existieren in der aktuellen Version der Anwendung Möglichkeiten zum Administrieren. Diese können jedoch noch ausgebaut werden. Insbesondere könnten bestehende Einträge (z.B. Challenges oder Projekte) durch entsprechende

Administratoren verwaltet werden. Dazu könnte das Archivieren zählen oder das Entfernen von doppelten Einträgen. Als weitere Funktion könnte die Verwaltung der Nutzerrollen über eine grafische Oberfläche ermöglicht werden, da dies aktuell nur über Veränderungen in der Datenbank vorgenommen werden kann. Die Funktionen für ein mögliches Backend sind keineswegs abschließend, sodass zukünftig auch weitere Administrationsmöglichkeiten implementiert werden können.

Schnittstellen zu Drittsystemen

Schnittstellen zu Drittsystemen sind in der aktuellen Version des Projektes noch nicht enthalten. Es können jedoch eine Reihe von solchen Schnittstellen geschaffen werden. Beispielsweise könnten Möglichkeiten entstehen, sodass Projektpläne, welche auf der Plattform erstellt wurden, durch weitere Projektmanagementtools ausgelesen werden können. Auf diese Weise entfallen Medienbrüche, sodass sowohl Fehler vermieden werden können, als auch die Effizienz gesteigert wird. Auch an diesem Punkt sind eine Reihe von weiteren Erweiterungen vorstellbar.

PDF-Generator

Eine weitere, mögliche Erweiterung ist das Generieren von PDF-Dokumenten aus der Anwendung heraus. Damit entsteht für die Nutzer die Möglichkeit, den aktuellen Stand des Projektes abzuspeichern und in einer äquivalenten Form auszudrucken, sodass eine sinnvolle Weitergabe möglich werden kann.

Business Intelligence

Gerade in der heutigen Zeit spielt die Auswertung und Aufbereitung von Daten eine immer wichtigere Rollen. Durch die Auswertung könnte bspw. analysiert werden, welche Klickpfade die Anwender nehmen und an welchen Stellen ggf. ein Ausstieg erfolgt. Die daraus erfolgte Erkenntnisse könnten in die Verbesserung der Innovationsplattform einfließen. Durch die Auswertung der Trackingdaten könnte die Notwenigkeit einer Applikation für mobile Endgeräte ermittelt werden. Für das Management könnte die Auswertung zum Vergleich von einzelnen Projekten untereinander und der Vergleich von Soll- und Haben des Projekts interessant sein. Dazu könnten die Daten bspw. durch graphische Elemente aufbereitet werden. Neben den zu definierenden Grundauswertungen für das Management kann eine individuelle Erweiterungsmöglichkeit gegeben werden, sodass die Analysen an die Anforderungen des jeweiligen Entscheidungsträgers erfolgen kann. Dies erfolgt in einem Management-Backend.

Ressourcenmarkt

Gerade in größeren Unternehmen ist oft unklar, bspw. welche Materiellen, aber auch

menschliches Wissen vorhanden sind. Ein Ressourcenmarkt soll diesem Problem begegnen, indem nach dem Prinzip des „schwarzen Bretts “ein Marktplatz gegeben wird, um Anfragen und Angebote machen zu können. Der Ressourcenmarkt bietet den Anwendern dann die Möglichkeit mit dem Suchenden oder Bietenden in Kontakt zu treten. Durch den Austausch der vorhandenen Ressourcen könnten diese effizienter genutzt werden. Für die Anwender bietet der Ressourcenmarkt die Chance neue Erfahrungen zu machen und Erkenntnisse und Fähigkeiten aus dem privaten Bereich einzubringen.

Crowdfounding

Das Crowdfounding bezeichnet die gemeinschaftliche Investition in ein Projekt, Produkt oder Vorhaben. Dabei legen verschiedene Investoren ihre finanziellen Mittel zusammen und erhalten bei erfolgreicher Umsetzung die Finanzierungssumme zurück. Dabei erfolgt die eigentliche Umsetzung erst, wenn der erforderliche Betrag in einem definierten Zeitraum erreicht wurde. Für die Übertragung des Prinzip des Crowdfounding erhalten die Abteilungsleiter ein virtuelles Budget, das sie für die Finanzierung der Projekte einsetzen können. Dabei können Sie bestimmen, in welcher Höhe die Investition erfolgen soll. Kommt ein Projekt nicht zur Umsetzung, dann erhält der Investor seinen Einsatz zurück. Im Fall einer Umsetzung ist der Einsatz zunächst „verloren “. Der Betrag der Auszahlung richtet sich dann nach dem Erfolg der Projektumsetzung. Bei Misserfolg ist der Einsatz teilweise oder komplett verloren. Bei einem Erfolg erhält der Investor den kompletten Einsatz plus einen Bonus. Neben dieser spielerischen Variante könnte das Unternehmen ein reales Budget zur Verfügung stellen, dass die Abteilungsleiter investieren können, um ein Projekt durchzuführen.

Ideenbörse

Für die Ideenbörse bekommen die Anwender ein Budget an Punkten zur Verfügung gestellt. Dieses können sie auf Challenges setzen. Entwickelt sich die Challenge in eine Richtung, die der Anwender nicht möchte, dann kann er das Budget komplett oder zum Teil wieder abziehen. Wird die Challenge erfolgreich umgesetzt, kommt es zu einem Zuwachs von Punkten, die verteilt werden können. Wird die Challenge nicht umgesetzt, dann werden die Punkte abgezogen. Dies zwingt den Anwender dazu, dass dieser sich Gedanken machen muss, ob er seine Punkte setzt. Auf Basis der gesetzten Punkte könnten Rankingübersichten erstellt werden, die dazu animieren, dass sich andere Mitarbeiter die Challenge ansehen. Der Zugewinn an Punkten erfolgt dabei über Aktivitäten. Dabei bekommen die Anwender bspw. fürs Einloggen, Bewertung/Kommentare abgeben, Challenge einreichen, Punkte, die zur Vergrößerung des Budget führen. Die Ideenbörse könnte dazu führen, dass sich die Anwender regelmäßig mit der Plattform beschäftigen.
Die Plattform IMPACT kann auf vielfältige Weise erweitert werden, sodass die oben genannten Punkte lediglich als Anreiz zu betrachten sind und keineswegs als abschließend beschrieben werden können.

10. Known Bugs

Im Folgenden werden die bekannten Fehler der Plattform IMPACT aufgezeigt, welche im späteren Verlauf zu beheben sind:

- *Kollaborativer Raum:* Das Whiteboard kann unter Umständen nur verzögert synchronisiert werden.
- *Gewinne dem Projekt hinzufügen / Anschaffungen dem Projekt hinzufügen*: Es können bei den Feldern, in denen die Gewinn- bzw Anschaffungskosten eingetragen werden auch Sonderzeichen mit eingetragen werden, wodurch eine Exception geworfen wird. *Responsive Design:* Zu kleine Auflösungen führen zur Überlappung von Elementen.

11. Fazit und Erfahrungsberichte der PG-Mitglieder

In diesem Kapitel wird zu den einzelnen Themengebieten ein Fazit gezogen, welches die Meinung der Projektgruppe widerspiegelt.

Ablauf der Projektgruppe

Generell ist zu resümieren, dass sich der Ablauf des Projektes in drei Phasen (Seminarphase, Entwurfs-/Konzeptionsphase und Entwicklungsphase) gliedert. Im Verlauf des Projektes hat sich herausgestellt, dass die Seminar- und Konzeptionsphase zu viel Zeit in Anspruch genommen hat. Eine Verkürzung der beiden Phasen würde bedeuten, dass mehr Zeit für die Entwicklung bleibt, welche sich im Endeffekt als besonders zeitintensiv herausgestellt hat. Auch wäre eine bessere Zeitplanung innerhalb der Projektgruppe erstrebenswert gewesen. Durch den Zeitmangel in der Entwicklungsphase wurde der Codefreeze hinausgezögert und nur unter großer Anstrengung rechtzeitig eingehalten.
Positiv zu bemerken ist, dass gerade zu Beginn der Projektphase eine ständiger Austausch mit dem Praxispartner und eine sehr gute Betreuung stattgefunden hat.

Eingesetzte Technologien

Viele Entscheidungen bezüglich der eingesetzten Technologien mussten früh getroffen werden, ohne dass Projektmitglieder zuvor ausreichende Erfahrungen sammeln konnten. Glücklicherweise stellte sich heraus, dass die eingesetzte Software zur Versionsverwaltung von Dateien (Git) eindeutig die richtige Entscheidung war. Es gab zwar einige Probleme, die sich auf damals mangelnde Erfahrung zurückführen lassen, jedoch stellte sich heraus, dass SVN nicht die Ansprüche der Projektgruppe hätte erfüllen können. Der Einsatz des Vaadin-Frameworks erleichterte die Umsetzung einer Webanwendung. Durch eine große Gemeinschaft existieren vielfältige Erweiterungen und Foren, die bei Fragen weiterhelfen. Jedoch wurden anderen Tools, wie zum Beispiel Jira, nicht konsequent eingesetzt und verloren dadurch ihren Mehrwert.

Scrum

Die Entscheidung Scrum als Vorgehensmodell zu wählen wird sehr kritisch betrachtet. Rückblickend wäre ein Vorgehensmodell, welches sich durch weniger Planungsmeetings und durch einen flexibleren Entwicklungszeitraum (Sprint) auszeichnet, besser geeignet. Scrum ist aufgrund der zahlreichen Treffen und den täglichen Scrum-Meetings nur bedingt einsetzbar. Die Einhaltung der Rahmenbedingung von Scrum erschwerten oftmals den Entwicklungsprozess. Zum anderen wurden essentielle Rollen, wie der Product Owner, nicht im vollen Umfang umgesetzt und erschwerten den Umgang mit Scrum.

Des weiteren wäre ein Veröffentlichen des aktuellen Standes nach jedem Sprint eine Maßnahme gewesen, die die Motivation innerhalb des Teams gefördert hätte. Dieser Vorgang wurde aber nicht konsequent eingehalten.

Kommunikation

Im Nachhinein hat die Kommunikation innerhalb der Projektgruppe des Öfteren nur unzureichend stattgefunden. Dies ist besonders der Raumsituation zuzuschreiben. Das gemeinsame Entwickeln innerhalb eines Raumes stellte sich als enorm wichtig heraus. Leider war in dem Projektgruppenraum kaum Platz für elf Studenten, was die Kommunikation ungemein erschwerte, da zwischenzeitlich einige Mitglieder auf andere Räumlichkeiten ausweichen mussten.

Rollen

Wie bereits erwähnt, wurden die Rollen innerhalb von Scrum nicht eingehalten bzw. erfüllt. Darüber hinaus hätte die Projektgruppe in Teams eingeteilt werden müssen. Innerhalb der Projektgruppe kristallisierten sich Experten für bestimmte Themenbereiche der Innovationsmanagementplattform heraus. Dadurch kam es oftmals zum Stillstand, wenn gewisse Experten nicht anwesend waren. Darüber hinaus hätte eine Einteilung in Teams zu einer eindeutigeren Aufgabenverteilung und zu einer effizienteren Bearbeitung dieser geführt.

12. Ausblick

Ziel des Projektes war es eine innerbetriebliche Innovationsmanagementplattform und den zu Grunde liegenden stehenden Prozess zu entwickeln. Der softwareunterstützte Innovationsprozess sollte den bisherigen ablösen und es dem Unternehmen ermöglichen, Innovationen zu verwalten. Der daraus entstehende Mehrwert sollte dem Unternehmen von nutzen sein, seine Marktposition zu verbessern oder zu halten.
Das Ziel einen softwareunterstützenden Prozess zu entwickeln wurde in Form des Prototypen der Innovationsmanagementplattform IMPACT umgesetzt. Der Prototyp bietet die Möglichkeiten Probleme, Ideen oder Vorschläge zu verwalten, um daraus mögliche Lösungen entstehen zu lassen.
IMPACT bietet Entwicklungspotential für weiterführende Projektgruppen. Zum einen wäre ein notwendiger Schritt, die Plattform mit einem Praxispartner zu evaluieren. In der Evaluation soll die Auswirkungen des Portals analysiert und bewertet werden. Dadurch könnten sich weitere Entwicklungszweige kristallisieren, die derzeitig nicht beachtet werden. Erweiterungsbereiche, die zu diesem Zeitpunkt als sinnvoll erachtet werden, beziehen sich insbesondere auf eine Erweiterung der Funktionalitäten.

Das Einbinden eines *Ressourcenmarkts* könnte eine dieser Erweiterungen darstellen. Hinter dem Begriff *Ressourcenmarkt* verbirgt sich die Idee, dass Anwender ihre Expertisen, Interessen und Zeit als eine Ressource einstellen können, um an neuen Projekten teilzunehmen. Der Benutzer könnte in diesem Kontext den Wunsch äußern, in anderen Unternehmensbereichen Erfahrungen zu sammeln. Des Weiteren könnte ein Crowdfunding-Ansatz implementiert werden, indem es den Anwendern ermöglicht wird eine Art Kapital einzusetzen. Durch diesen Ansatz wäre es möglich, dass sich die Benutzer stärker mit den Ideen identifizieren und diese zum Erfolg führen. Dementsprechend könnten Challenges bzw. Ideen zeitlich begrenzt werden und erst ab einem gewissen Betrag an Kapital als Projekt gestartet werden. Dies hätte zur Folge, dass nur Projekte erstellt werden, die den meisten Zuspruch der Anwender erhalten und somit die Meinung der Anwender wiederspiegeln.
Der Bereich des Gamification könnte ebenfalls ausgebaut werden. Der Einsatz von *Easter Eggs*, Freischaltung von Minispielen oder eines Rangsystems von Benutzern könnte den Anwender dazu motivieren, mit der Plattform zu arbeiten. Dafür müsste eine Analyse des Unternehmens und dessen Spielertypen vorgenommen werden um den besten Ansatz für das Unternehmen weiter zu verfolgen (siehe Kapitel 3.2.3, Seminararbeit *Gamification*).
Die Aufbereitung und Auswertung von Daten, die in dem System entstanden sind, könnte ebenfalls ein interessantes Erweiterungspotenzial bieten. Prognosen über mögliche Problemquellen, Erweiterungsbedarf und Technologietrends anhand der Datenbasis könnten dem Unternehmen einen Mehrwert bieten. Frühzeitiges Beheben von Problemen fördert zum einen das Allgemeinwohl der Unternehmens und ermöglicht zum anderen den Mitarbeitern einen ungestörten Arbeitsablauf. Das Entdecken von Technologietrends könnte zu einem Wettbewerbsvorteil gegenüber Konkurrenten führen.
Zusammenfassend ist zu sagen, dass der umgesetzte Prototyp bereits die Grundfunktionalitäten bietet, jedoch noch in viele Richtungen erweiterbar ist. Je nach Unternehmen kann der Prozess angepasst und erweitert werden. Dadurch weist IMPACT einen großen Grad an Flexibilität auf und macht es zu einem vielseitig einsetzbaren System. Darüber hinaus ermöglicht die Innovationsmanagementplattform nicht nur die Chance aus Problemen Lösungen entstehen zu lassen, sondern geht einen Schritt weiter und lässt Platz für Innovationen.

Literatur

[AAB10] Jonathan Arnowitz, Michael Arent, and Nevin Berger. *Effective prototyping for software makers*. Elsevier, 2010.

[Apa16] Apache Maven Project. What is maven? `https://maven.apache.org/what-is-maven.html`, 2016. Zuletzt besucht am 01.04.2016.

[BAE+14] J. Baumann, D. Arndt, F. Engelen, F. Hardy, and C. Mjartan. *Vaadin: Der kompakte Einstieg für Java-Entwickler*. Dpunkt.Verlag GmbH, 2014.

[Beh11] Mario Behrendt. *Jenkins: Kurz und gut*. O-Reilly, 2011.

[BKP+13] Manfred Baumgartner, Martin Klonk, Helmut Pichler, Richard Seidl, and Siegfried Tanczos. *Agile testing : der agile Weg zur Qualität*. Hanser, München, 2013.

[Bro96] J. Brooke. SUS: A 'Quick and Dirty'Usability Scale, Usability Evaluation in Industry, 1996.

[BRS06] Kurt Badertscher, Roger Romano, and Johannes Scheuring. *Wirtschaftsinformatik: Konzeption und Planung eines Informations- und Kommunikationssystems*. Compendio Bildungsmedien, Zürich, 2006.

[BTH14] Anton Barua, Stephen W. Thomas, and Ahmed E. Hassan. What are developers talking about? an analysis of topics and trends in stack overflow. *Empirical Software Engineering*, 19(3):619–654, 2014.

[Bun14] Bundesamt für Sicherheit in der Informationstechnik. Webanwendungen. `https://www.bsi.bund.de/DE/Themen/ITGrundschutz/ITGrundschutzKataloge/Inhalt/_content/baust/b05/b05021.html`, 2014. Zuletzt besucht am 07.04.2016.

[Bur14] Brain Burke. Gartner Redefines Gamification. Gartner Redefines Gamification: `http://blogs.gartner.com/brian_burke/2014/04/04/gartner-redefines-gamification/`, 2014. Zuletzt besucht am 19.03.2016.

[CHC10] Stefan Christmann, Svenja Hagenhoff, and Thorsten Caus. Webbasierte Anwendungen als Lösungsansatz für die Heterogenität im mobilen Internet. `http://webdoc.sub.gwdg.de/ebook/serien/lm/arbeitsberichte_anwebus/2010_03.pdf`, 2010. Zuletzt besucht am 02.04.2016.

[Chr15] Christian Baranowski. SWQS Software-Qualitätssicherung. `http://sw-qs.blogspot.de/2015/10/einfuhrung-software-qualitatssicherung.html`, 2015. Zuletzt besucht am 07.04.2016.

[CS14] Scott Chacon and Ben Straub. *Pro git*. Apress, 2014.

[DeM13] Linda DeMichiel. Jsr 338: Java persistence api, version 2.1. Technical report, Oracle, 2013.

[Ebh16] Dieter Ebhart. Veranschaulichung team- internes und -externes Reporting. \errhelp{Use‘‘forasimpledoublequotecharacter.}\errmessage{ngerman: Thecommand"hisundefined}‘‘http://pi.informatik.uni-siegen.de/stt/34_1/03_Technische_Beitraege/Steuerung_final_04.pdf, 2016. Zuletzt besucht am 19.03.2016.

[ES13] G. Engels and W. Schäfer. *Programmentwicklungsumgebungen: Konzepte und Realisierung*. XLeitfäden der angewandten Informatik. Vieweg+Teubner Verlag, 2013.

[Fis11] Bill Fischer. Ceos say innovation is most important factor for growth — voxy.co.nz. http://www.forbes.com/sites/billfischer/2011/06/04/ceos-say-innovation-is-most-important-factor-for-growth-voxy-co-nz/#b59f6dc72e90, 2011. Zuletzt besucht am 07.04.2016.

[Fou16] The Apache Software Foundation. Apache shiro architecture. http://shiro.apache.org/architecture.html, 2016. Zuletzt besucht am 07.03.2016.

[Fow05] Martin Fowler. *Refactoring – Wie Sie das Design vorhandener Software verbessern*. Addison-Wesley Verlag, München, 2005.

[Glo13] Boris Gloger. *Scrum: Produkte zuverlässig und schnell Entwickeln*. Carl Hanser Verlag GmbH & Company KG, 2013.

[Grö15] Marko Grönroos. *Book of Vaadin*. Vaadin Ltd, 2015.

[Hei16] A. Hein. Hinweise zur Erstellung und Dokumentation benutzerfreundlicher und fehlertoleranter Software. https://www.in.th-nuernberg.de/professors/Hein/HINTS/Software_Development.pdf, 2016. Zuletzt besucht am 19.03.2016.

[HPI15] HPI School of Design Potsdam. Design Thinking – Mindset. http://hpi.de/school-of-design-thinking/design-thinking/mindset.html, 2015. Zuletzt besucht am 05.08.2015.

[HS11] Jürgen Hausschildt and Sören Salomo. *Innovationsmanagement*. Vahlen Verlag, München, 1. auflage edition, 2011.

[Hue16] Hueßmann. Anforderungsermittlung. `http://st.inf.tu-dresden.de/Lehre/WS00-01/st2/vorlesung/st2k2a-extra.pdf`, 2016. Zuletzt besucht am 07.04.2016.

[HvB16] Paul Herwarth von Bittenfeld. User Stories: Anforderungen aus Nutzersicht dokumentieren. `https://blog.seibert-media.net/blog/2010/11/29/user-stories-anforderungen-aus-nutzersicht-dokumentieren/`, 2016. Zuletzt besucht am 07.04.2016.

[Joh16] Christian Johner. ISO 9126 und ISO 25010. `https://www.johner-institut.de/blog/iec-62304-medizinische-software/iso-9126-und-iso-25010`, 2016. Zuletzt besucht am 07.04.2016.

[Kri16] Alexander Kriegisch. Aufgaben Scrum Master. `http://scrum-master.de/Scrum-Rollen/Scrum-Rollen_ScrumMaster`, 2016. Zuletzt besucht am 07.04.2016.

[Lig01] Peter Liggesmeyer. *Software-Qualität: Testen, Analysieren und Verifizieren von Software*. Spektrum Akademischer Verlag, Heidelberg u.a., 2001.

[LPKS14] Olga Liskin, Raphael Pham, Stephan Kiesling, and Kurt Schneider. *Agile Processes in Software Engineering and Extreme Programming - Why We Need a Granularity Concept for User Stories*. Springer, Heidelberg, 2014.

[Mar09] Robert C. Martin. *Clean Code – A Handbook of Agile Software Craftsmanship*. Pearson Education, Inc., Boston, 2009.

[Mar14] R.C. Martin. *Clean Coder: Verhaltensregeln für professionelle Programmierer*. mitp Professional. mitp/bhv, 2014.

[MG14] H.D. Mummendey and I. Grau. *Die Fragebogen-Methode: Grundlagen und Anwendung in Persönlichkeits-, Einstellungs- und Selbstkonzeptforschung*. Hogrefe, 2014.

[MJ14] Michael Meder and Brijnesh-Johannes Jain. The Gamification Design Problem. The Gamification Design Problem: `http://arxiv.org/pdf/1407.0843.pdf`, 2014. Zuletzt besucht am 05.04.2016.

[MR16] Helga Meyer and Heinz-Josef Reher. *Projektmanagement*. Springer, Wiesbaden, 1. auflage edition, 2016.

[MW12] B. Müller and H. Wehr. *Java Persistence API 2: Hibernate, EclipseLink, OpenJPA und Erweiterungen*. Carl Hanser Verlag GmbH & Company KG, 2012.

[Pfi15] Pfitzer, Norbert and Oser, Peter and Dennerlein, Birgitta and Böcking, Hans-Joachim and Weber Jürgen and Henselmann, Klaus. Bewertung. `http://wirtschaftslexikon.gabler.de/Archiv/55225/35/Archiv/55225/bewertung-v12.html`, 2015. Zuletzt besucht am 07.04.2016.

[PJ14] Kathrin Passig and Johannes Jander. *Weniger schlecht programmieren*. O'Reilly Verlag, Köln, 2014.

[Rai14] S. Raimer. *Design Thinking, Lean, Agile: Schlüssel zur Innovation*. Symposion Publishing GmbH, 2014.

[RB08] Harald Rüggeberg and Kjell Burmeister. Innovationsprozesse in kleinen und mittleren unternehmen. `http://hdl.handle.net/10419/74333`, 2008. Zuletzt besucht am 01.04.2016.

[Ree00] G. Reese. *Database Programming with JDBC and Java*. Java (o'Reilly) Series. O'Reilly, 2000.

[Rei16] Kersten Reich. Theoretische und praktische Begründung. `http://methodenpool.uni-koeln.de/feedback/feedback_begruendung.html`, 2016. Zuletzt besucht am 07.04.2016.

[Rid00] Elena Ridolfo. *Ideenmanagement*. Tectum Verlag, Marburg, 2000.

[Rup14] Chriss Rupp. *Requirements-Engineering und -Management: Aus der Praxis von klassisch bis agil*. Hanser, München, 6. edition, 2014.

[Sch12] Kurt Schneider. *Abenteuer Softwarequalität: Grundlagen und Verfahren für Qualitätssicherung und Qualitätsmanagement*. dpunkt.verlag, Heidelberg, 2. überarbeitete und erweiterte auflage edition, 2012.

[Sch16] Ina Schaefer. Anforderungsanalyse Software Engineering. `https://www.tu-braunschweig.de/Medien-DB/isf/sse/v2-re.pdf`, 2016. Zuletzt besucht am 07.04.2016.

[Six16] H.W. Six. Anforderungsermittlung. `http://www.mathematik-netz.de/pdf/Anforderungsermittlung.pdf`, 2016. Zuletzt besucht am 18.03.2016.

[SJ10] Thomas Stern and Helmut Jasberg. *Erfolgreiches Innovationsmanagement*. Gabler Verlag, Wiesbaden, 4. auflage edition, 2010.

[SL12] A. Spillner and T. Linz. *Basiswissen Softwaretest: Aus- und Weiterbildung zum Certified Tester ; Foundation Level nach ISTQB-Standard*. dpunkt-Verlag, 5 edition, 2012.

[SM05] R. Sheldon and G. Moes. *Beginning MySQL*. Programmer to Programmer. Wiley, 2005.

[SRU10] Sven Schwarz, Gilad Riedl, and Miroslav Ures. Innerbetriebliches innovationsmanagement als prozess. http://download.springer.com/static/pdf/75/art2010.

[Sta08] StackOverflow. What are mvp and mvc and what is the difference? `http://stackoverflow.com/questions/2056/what-are-mvp-and-mvc-and-what-is-the-difference/34377336`, 2008. Zuletzt besucht am 07.04.2016.

[Win11] M. Winkler. Innovative Teams im Design Thinking, 2011.

[ZB05] J.D. Zawodny and D.J. Balling. *High performance MySQL: Optimierung, Datensicherung, Replikation & Lastverteilung ; [fortgeschrittene Techniken für MySQL-Administratoren]*. O'Reilly, 2005.

[Zil12] Christoph Zillgens. *RResponsive Webdesign: reaktionsfähige Websites gestalten und umsetzen*. Carl Hanser Verlag GmbH Co KG, 2012.

A. Anhang

A.1. Installationsanweisung

Das Einrichten der Webapplikation IMPACT erfordert die Installation von zusätzliche Programmen. Zum einen ist ein Webserver für Java-EE 7 Web Profile Anwendungen erforderlich (z.B. Apache TomEE 1.7.2, welcher in diesem Projekt verwendet wurde). Zum anderen wird die Installation einer Datenbank benötigt, welche durch das Framework EclipseLink unterstützt wird. Es existieren eine Reihe von Konfigurationsdateien, nachfolgend beschrieben, anhand dessen Einstellungen vorgenommen werden können.

Datenbankkonfiguration

Über die Datei Persistence.xml müssen Einstellungen getroffen werden, um die Verbindung mit der Datenbank zu ermöglichen. Innerhalb der Datei existieren eine Reihe von Properties, deren Werte zugewiesen werden können. Die in Tabelle 37 dargestellten Properties müssen gezwungenermaßen konfiguriert werden. Es sind darüber hinaus weitere Konfigurationsmöglichkeiten vorhanden, wodurch zusätzliches Optimierungspotential besteht. Eine vollständige Übersicht der Möglichkeiten zur Konfiguration kann in den JPA-Spezifikationen eingesehen werden, (Siehe [DeM13], S. 362 ff.)

Property	Wert
javax.persistence.jdbc.url	Die URL der Datenbank, zu der die Verbindung aufgebaut wird.
javax.persistence.jdbc.user	Der Nutzer, welcher in dem Datenbankmanagementsystem existiert und von IMPACT genutzt werden soll.
javax.persistence.jdbc.password	Das zum angegebenen Nutzer vergebene Passwort.

Tabelle 37: Zwingend erforderliche Konfiguration zur Einstellung der verwendeten Datenbank

Konfiguration des Sicherheitsmanagements

Wie in Kapitel 4.4.5 beschrieben, wird Apache Shiro zur Umsetzung des Login- und Rechtemanagements genutzt. Über die Konfigurationsdatei shiro.ini lassen sich Einstellungen vornehmen, sodass andere Quellen für Login-Informationen wählbar sind. Dies kann über das Property *securityManager.realm* definiert werden. Zur aktuellen Zeit existiert jedoch noch keine Implementierung, um andere Quellen zu ermöglichen. Zukünftig, über den Abschluss des Projektes hinaus, könnten jedoch Weitere implementiert werden.

Konstante	Wert
SMTP HOST	Der Hostname bzw. die IP-Adresse des SMTP-Servers
SMTP PORT	Der Port des SMTP-Servers
SMTP USER	Der Nutzername für eine Anmeldung am Server
SMTP PASSWORD	Das Passwort des angegebenen Nutzers
SMTP FROM	Der Name, welcher in versendeten E-Mails als Absender verwendet wird.

Tabelle 38: Konstanten der Klasse MailService, zur Konfiguration des zu verwendenden Mailserver zum versenden von E-Mails

Mailkonfiguration

Die Mailkonfiguration betrifft die Funktion des Wiederherstellens von Passwörtern einzelner Nutzer, siehe Kapitel 6.2.2. Das System sendet dazu eine Mail über einen Mailserver. Zur Nutzung der Funktion muss ein valider Postausgangsserver (SMTP-Server) definiert werden. In der aktuellen Version des Projektes ist eine Konfiguration nicht über Konfigurationsdateien möglich. Stattdessen müssen entsprechende Werte in der Klasse MailService implementiert werden. Es existieren dazu entsprechende Konstanten, welche in Tabelle 38 aufgezählt und erläutert werden.

A.2. Übersicht der Rechtekonfiguration

Daniel Die Benutzerrollen und deren Rechte, wie in Kapitel 2.1 beschriebenen, und durch das Framework Apache Shiro umgesetzt wurden (vgl. 4.4.5), können auf verschiedene Weise konfiguriert werden. Zum einen besteht die Möglichkeit, bestehende Rechte einzelner Rollen zu erweitern bzw. einzuschränken, zum anderen können den Nutzern des System Rollen gegeben bzw. entzogen werden. Für die Möglichkeit, Rechte einzelner Rollen zu editieren, existieren aktuell noch keine entsprechenden Benutzeroberflächen. Die Konfiguration muss daher über das in Kapitel 5.4 beschriebene Datenmodell erfolgt. Über die Tabelle *PermissionRole* kann bestimmt werden, welche Berechtigungen einer bestimmte Rolle zugewiesen werden. Dazu wird, unter der jeweiligen Rolle, für einzelne Berechtigungen der Wert *0* gesetzt, sofern Nutzern dieser Rolle der entsprechenden Zugriff verwehrt werden soll. Der Wert *1* anstelle dessen wird zum Gewähren des Zugriff angegeben.
Die Konfiguration der Rollen einzelner Nutzer kann hingegen über eine grafische Oberfläche erfolgen. Dazu wird ein Nutzer der Plattform erfordert, welcher über die Administratorrolle verfügt. In diesem Fall erscheint in den Nutzerprofilen aller Nutzer im System der zusätzliche Reiter *Administration*. In dem entsprechenden Fenster werden die festgelegten Rollen des betrachteten Nutzer dargestellt. Über die Button *Hinzufügen* bzw. *Entfernen* können entsprechend neue Rollen hinzugefügt bzw. Bestehende entfernt werden.

A.3. Aktivitäten der PG

A.3.1. Lufthansa-Technik-Tag

Unser Betreuer auf Seiten der Lufthansa, Dr. Joachim Kurzhöfer, hat unsere Projektgruppe eingeladen auf dem diesjährigen Technologietag der Lufthansa unser Projekt/Produkt zu präsentieren. Dieser Einladung sind wir sehr gerne nachgekommen. Die Vorbereitungen begannen mit der Überlegung welche Inhalte wir in die 20-minütige Präsentation integrieren. Da wir uns als Externe von dem Kreis der üblichen Vortragenden unterscheiden, haben wir uns dazu entschieden zunächst einmal die Projektgruppe als solche (Aufbau, Struktur, Rahmen) und ihre Aufgaben und Ziele vorzustellen. Da zu dem Zeitpunkt noch kein Prototyp fertig war, wurde im mittleren Teil der Präsentation unser Prozessmodell der Innovationsmanagementplattform erläutert. Den Schlussteil bestand aus der technologischen Besonderheit in unserem Projekt, nämlich dem Vaadin Framework. Die Funktionsweise des serverseitigen Java-Frameworks wurde in den Grundzügen erklärt. Bei serverseitigen Frameworks stellt sich häufig die Frage „Does it scale?“. Aus diesem Grund haben wir zum Abschluss der Präsentation noch einige Last-Tests vorgestellt sowie die Skalierung betrachtet. Diese fällt entgegen der häufig verbreiteten Erwartung sehr gut aus.
Der Technologietag begann um 9.00 Uhr. Da wir unsere erste Präsentation um 11.15 Uhr hatten und die Anreise knapp 3,5 Stunden in Anspruch nahm, trafen wir gegen 10.30 Uhr ein. In fünf Räumen wurden parallel zueinander Vorträge zu unterschiedlichen Themen (u.a. Big Data, Windows 10, SAP Fiori, automatisierte Tests) gehalten. Dabei konnten die grob geschätzten 75 Mitarbeiter sich frei entscheiden, welche Vorträge sie sich anhören. Unsere Vorträge um 11.15 Uhr und 15.35 Uhr stießen auch auf Interesse der Mitarbeiter und es konnten im Anschluss noch einige Fragen zu der Projektgruppe und der Innovationsmanagementplattform beantwortet werden.
Für uns als Zuhörer war der Vortrag zu dem Thema „Mydea – Modernes Innovationsmanagement mit Crowdfunding und Gamification“ besonders interessant, da sich der Vortrag mit unserem Thema befasste. Es wurde eine Innovationsmanagement-Lösung basierend auf Microsoft SharePoint präsentiert, die mit Elementen aus Crowdfunding und Gamification erweitert wurde. Die wichtigste Erkenntnis für uns war, dass die Prozesse sehr ähnlich zu unseren sind und man sich somit auf Modellebene auf einem guten Weg befindet. Gegen 17.00 Uhr traten wir die Heimreise an und trafen gegen 20.30 Uhr wieder in Oldenburg ein. Ein insgesamt sehr interessanter und auch lehrreicher Tag geht somit zu Ende.

A.3.2. CeBIT 2016

Bereits im Oktober 2015 wurde die Möglichkeit eröffnet den Prototypen als Projektgruppe IMPACT auf dem niedersächsischen Gemeinschaftsstand auf der CeBIT 2016 in Hannover

zu präsentieren. Nach einer internen Abstimmung während der Projektgruppensitzung haben sich die Mitglieder der Projektgruppe für eine Bewerbung ausgesprochen und es wurde mit den ersten Vorbereitungen wurden durchgeführt. Wesentliche Inhalte der Bewerbung waren neben der Innovation am Projekt die Erwartungen, die an den Messeauftritt geknüpft sind. Die wichtigsten Erwartungen waren:

- Nachhaltige Partnerschaften zwischen Universität und Unternehmen
- Ideen und Anregungen von externen Fachexperte
- Kontakte zu Investoren für die mögliche Gründung eines Start-Up Unternehmen

Die eingereichte Bewerbungsunterlage ist unter A.3.3 einzusehen. Anfang November erhielt die Projektgruppe die positive Rückmeldung aus Hannover. Nach der Zusage begann sofort die Ausarbeitung der Inhalte für die CeBIT. In Zusammenarbeit mit der Redaktion KRISPIN Marketing Management aus Hannover wurden der Stand-Flyer A.3.4 als auch die Grafiktafel erarbeitet . Parallel dazu wurden die benötigten Anträge und Formulare ausgefüllt und bei den entsprechenden Stellen eingereicht. Um unsere Plattform auch ansprechend präsentieren zu können, wurde ein Produktvideo aufgenommen. In diesem Video wird mit Hilfe von Schaubildern das Grundkonzept vorgestellt. Die wichtigsten Standbilder des Videos können unter A.3.5 betrachtet werden. Auf der CeBIT 2016 haben sich die Erwartungen der Projektgruppe erfüllt. Täglich besuchten interessierte Fachbesucher den Stand und informierten sich über die Innovationsmanagementplattform IMPACT. Unter den Besuchern befanden sich auch potentielle Investoren, die sich über die Zukunft der Lösung erkundigten. Rückblickend hat sich der Aufwand für die CeBIT gelohnt, denn neben konstruktiven Verbesserungsvorschlägen wurde vor allem positives Feedback erhalten.

A.3.3. CeBIT Bewerbung

CeBIT 2016

Exponat-Anmeldung der Universität Oldenburg
IMPACT – Innovation Management Platform To Activate Creative Thoughts

1. Anmeldende Hochschule
   Carl von Ossietzky Universität Oldenburg

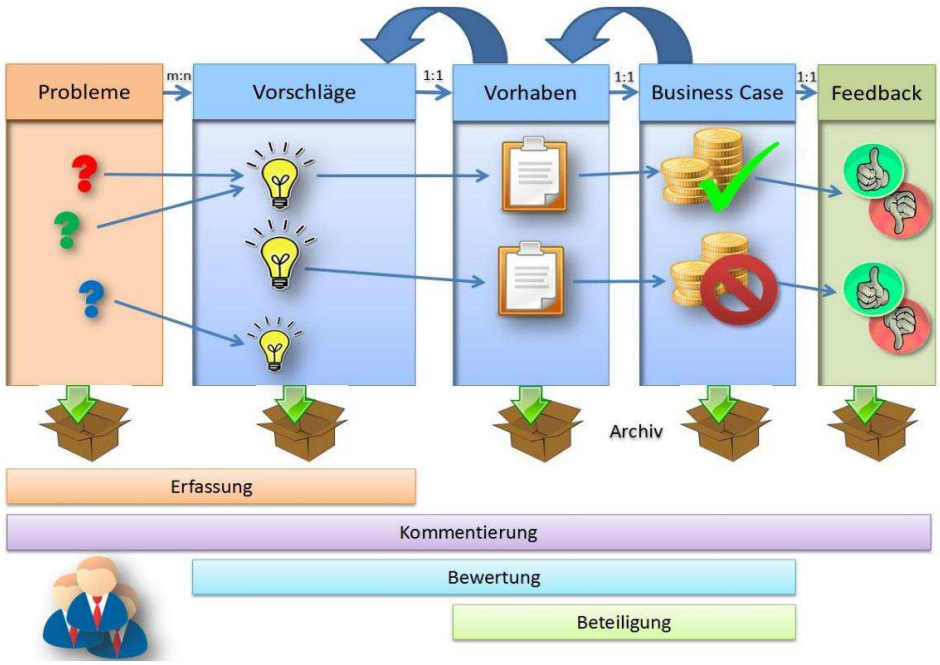
2. Titel des Projekts
   IMPACT - Innovation Management Platform To Activate Creative Thoughts

3. Betreuendes Institut/ Betreuende Professur
   Department für Informatik
   Abteilung Wirtschaftsinformatik I /VLBA
   Prof. Dr.-Ing. Jorge Marx Gómez
   jorge.marx.gomez@uni-oldenburg.de

4. Thema
   Thema des Projekts „IMPACT“ ist eine Innovationsmanagement-Plattform zur Verbesserung des innerbetrieblichen Innovationsmanagements.
   Der Innovationsprozess ist in vielen Unternehmen in Form einer physischen „Ideenbox“ umgesetzt. Die Schwächen dieses Vorgehens sind, dass die Ideen nicht effizient weiterbearbeitet und nur unzureichend im Unternehmen kommuniziert werden. Vorhandene Innovationspotenziale und Optimierungsmöglichkeiten zur Erhöhung der strategischen Wettbewerbsfähigkeit bleiben ungenutzt.
   An dieser Stelle setzt die von uns entwickelte Softwareplattform an. Ziel ist es, den unternehmensinternen Innovationsprozess
   - kollaborativ (Zusammenarbeit an Problemen, Ideen und Lösungen),
   - durchgängig (Von der Problemerfassung über Lösung und Business Case bis hin zum Feedback) und
   - transparent (Arbeitsfortschritte und Entscheidungen sind für alle Mitarbeiter jederzeit einsehbar und nachvollziehbar)

   zu gestalten.

5. Was ist das Innovative an diesem Projekt?
   Unsere Innovationsmanagement-Plattform verfolgt einen ganzheitlichen Ansatz. So werden neben der reinen Problem- und Vorschlagserfassung auch die Erstellung von Vorhaben, die Generierung eines Business Cases sowie das abschließende Feedback und somit der gesamte Innovationsprozess abgebildet (siehe Abbildung).
   Wir bieten den Anwendern einen kollaborativen Raum (u. a. mit Chat und Whiteboard) für die gemeinsame Erarbeitung von Vorschlägen und Vorhaben. So können Mitarbeiter unterschiedlicher Unternehmensbereiche ihr Wissen bündeln, Synergieeffekte nutzen und gemeinsam die Qualität der Ergebnisse steigern. Die interdisziplinäre Zusammenarbeit wird durch Ansätze aus dem Design Thinking unterstützt. Durch den damit verbundenen humanorientierten Ansatz sollen benutzerorientierte und zielgruppengerechte Lösungen entstehen.
   Um die Motivation und die Anreize zur Ideenentwicklung auch langfristig aufrecht erhalten zu können, werden Elemente aus dem Gamification-Umfeld zielgerichtet eingesetzt. Bewertungs- und Rankingsysteme sorgen für einen spielerischen Umgang mit der Software, unterstützten kreative Denkprozesse und erhöhen den Spaß an der Ideengenerierung.



6. Exponatbeschreibung
   Es sollen zwei PCs mit Präsentation und Prototyp aufgebaut werden. Diese sollen durch mindestens ein Poster ergänzt werden, sodass Inhalte des Projekts veranschaulicht dargestellt werden können. Die Präsentation beinhaltet das Prozessmodell der Plattform und weitere relevante Aspekte des Projekts.

7. Bedarf an Ausstellungsfläche
   Ein Standard-Arbeitsplatz

8. Sind Unternehmen als Kooperationspartner am Projekt beteiligt? (wenn ja, welche)
   Lufthansa Industry Solutions TS GmbH

9. Welche Erwartungen knüpfen Sie an den Messeauftrifft?

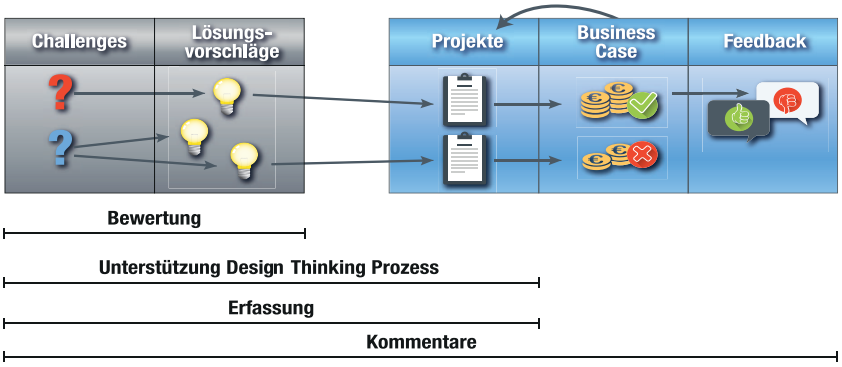
   - Aus dem Projekt heraus können nachhaltige Partnerschaften zwischen Universität und Unternehmen geknüpft werden, die wiederum einen noch besseren Praxisbezug des Studiums in Oldenburg ermöglichen.
   - Die entstehenden neuen Kontakte bringen auch Vorteile für die Studenten aus dem Projekt:
     Kontakte zu potentiellen Arbeitgebern können geknüpft werden und es erfolgt eine praxisnahe Präsentation der Studieninhalte.
   - Ideen und Anregungen sollen mit externen Fachexperten und Austellern ähnlicher Lösungen ausgetauscht werden.
   - Durch die Gespräche mit den Fachexperten erhoffen wir uns das Marktpotenzial unserer Lösung besser einschätzen zu können.
   - Unser studentisches Projekt soll einem breiten Publikum nahegebracht werden und somit auch zur Reputation der Universität beitragen.
   - Die Ausrichtung der Universität im Bereich der Wirtschaftsinformatik soll nicht nur Akademikern, sondern auch Studieninteressierten nahegebracht werden, indem eine praxisnahe Präsentation der Studieninhalte der Wirtschaftsinformatik erfolgt.

A.3.4. CeBIT Flyer

CARL VON OSSIETZKY universität OLDENBURG

IMPACT – Innovationsmanagement-Plattform fördert kreative Gedanken

IMPACT – Innovation Management Platform to Activate Creative Thoughts



Prozessmodell der Innovationsmanagement-Plattform
Process model of the innovation management platform

In vielen Unternehmen spielt das Innovationsmanagement eine untergeordnete Rolle. Die Prozesse sind oft intransparent, langwierig und bieten wenig Raum für gemeinsame Ideenfindung. Hier setzt die Innovationsmanagement-Plattform IMPACT an.

Die einzigartige Lösung bildet den vollständigen Prozess von der Problemerfassung bis hin zum Feedback ab. Die Anwender haben über kollaborative Methoden die Möglichkeit, gemeinsam an Vorschlägen und Vorhaben zu arbeiten. Dadurch wird nicht nur das Wissen unterschiedlicher Unternehmensbereiche gebündelt, sondern auch die Qualität der Ergebnisse gesteigert. Die interdisziplinäre Zusammenarbeit wird zusätzlich durch Ansätze aus dem Design-Thinking unterstützt. Durch diese humanorientierte Vorgehensweise sollen benutzerorientierte und zielgruppengerechte Lösungen entstehen. Um die Motivation und die Anreize zur Ideenentwicklung auch langfristig aufrechterhalten zu können, werden Elemente aus dem Gamification-Umfeld eingesetzt. Bewertungs- und Rankingsysteme sorgen für einen spielerischen Umgang mit der Software, unterstützen kreative Denkprozesse und erhöhen den Spaß an der Ideengenerierung. Arbeitsfortschritte und Entscheidungen sind für die Mitarbeiter jederzeit transparent gestaltet.
Zusammengefasst werden vorhandene Innovationspotenziale besser ausgeschöpft und ermöglichen dadurch eine Erhöhung der strategischen Wettbewerbsfähigkeit.

Fakultät II – Department für Informatik
Abt. Wirtschaftsinformatik / VLBA

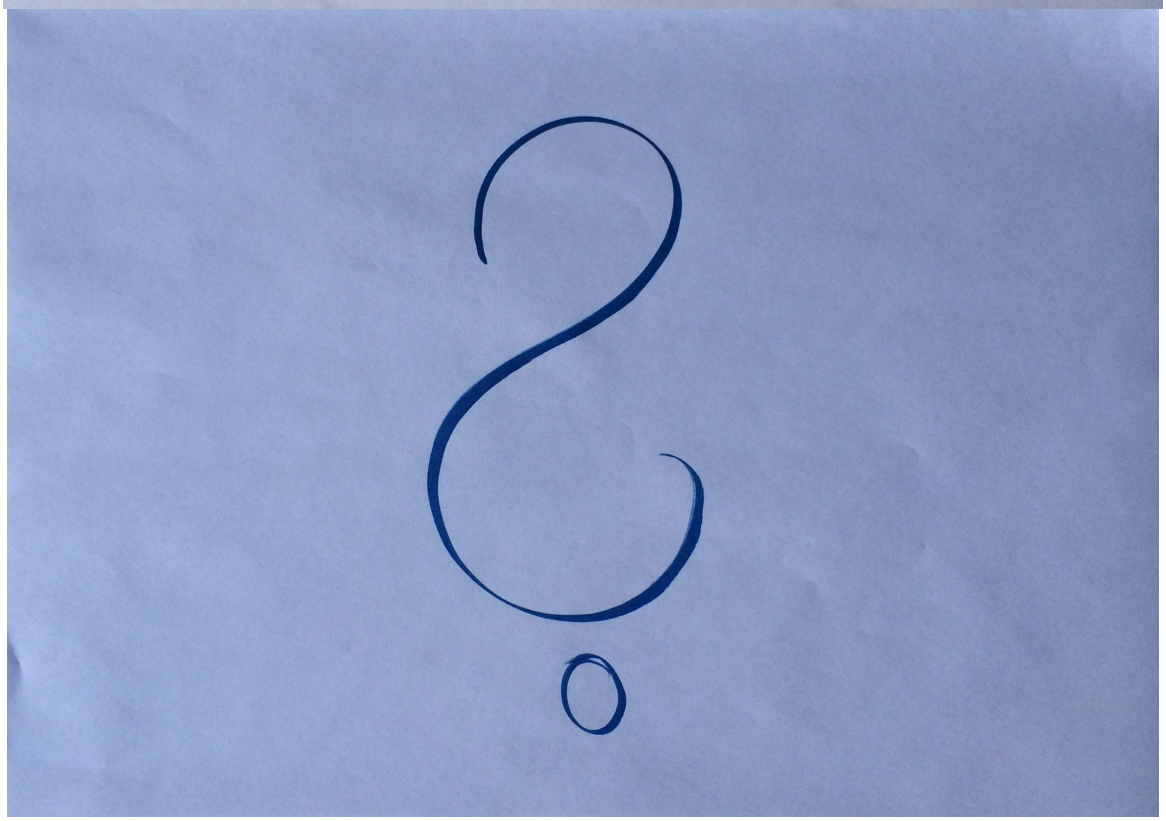
Prof. Dr.-Ing. Jorge Marx Gómez

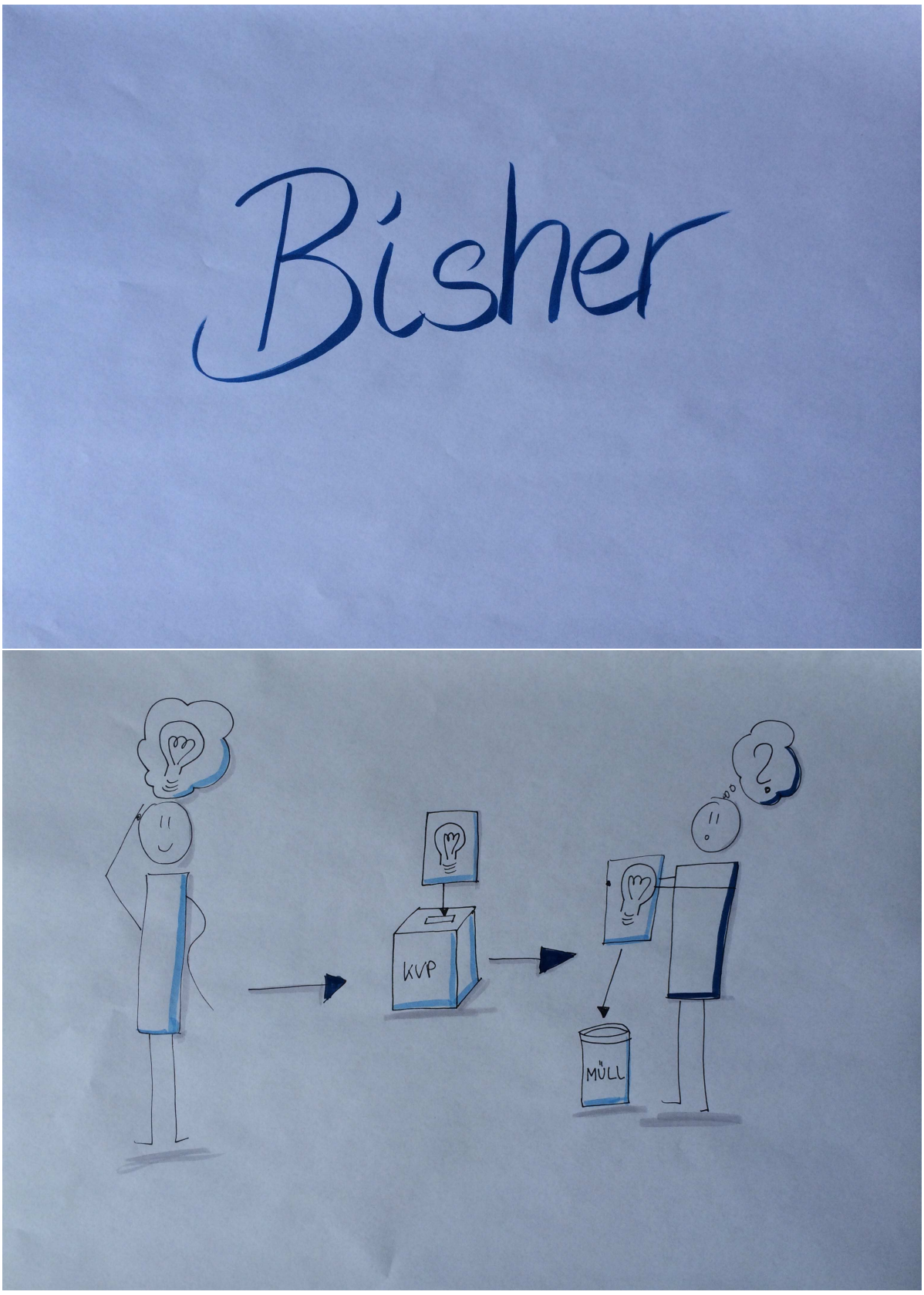
jorge.marx.gomez@uni-oldenburg.de
https://www.uni-oldenburg.de/vlba/

In many companies, innovation management plays a subordinate role. The processes are often not transparent and offer little room for collective brainstorming.
The stand-alone innovation management platform IMPACT represents the complete process from the problem identification up to the feedback. Users from different business sectors have the opportunity to work together using collaborative methods to create powerful solutions. The creative process is also supported by approaches from the design thinking and gamification area.

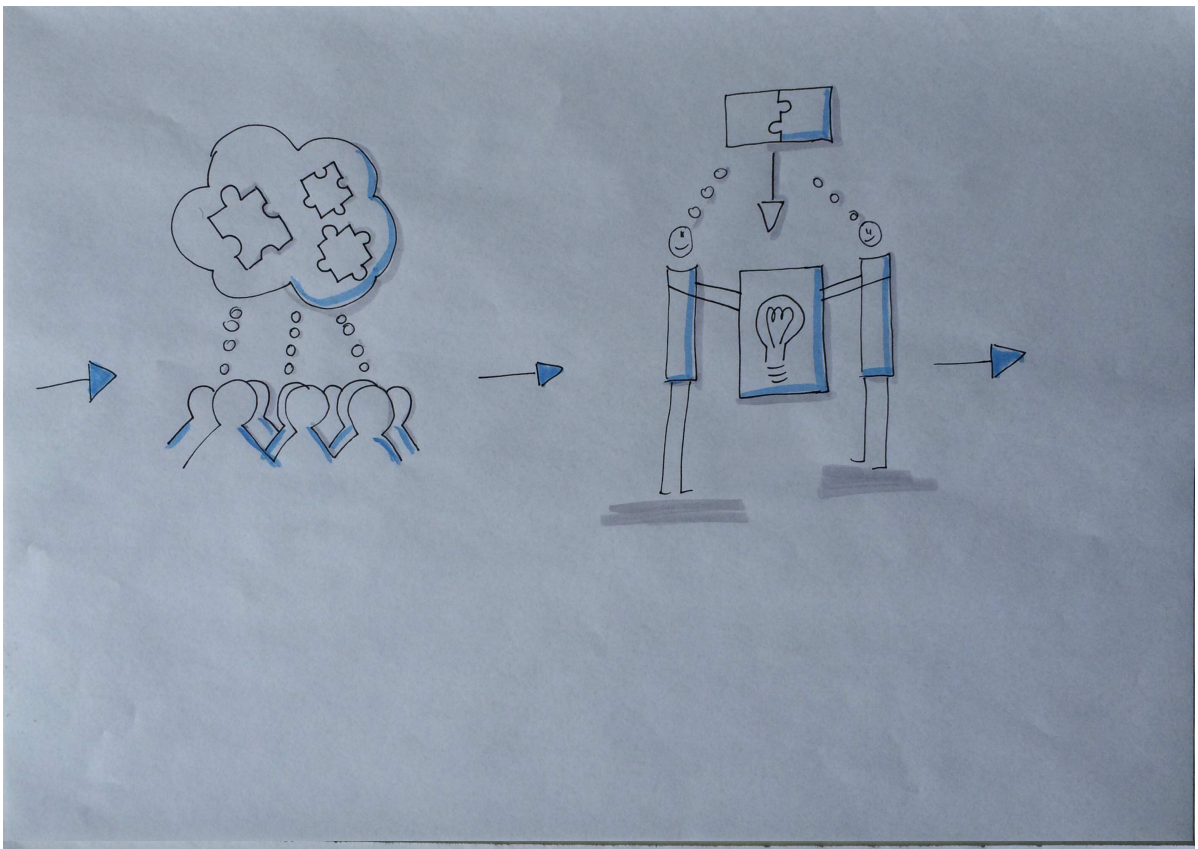
Overall, the existing innovation potentials are better scooped and thereby, the strategic competitiveness increases.

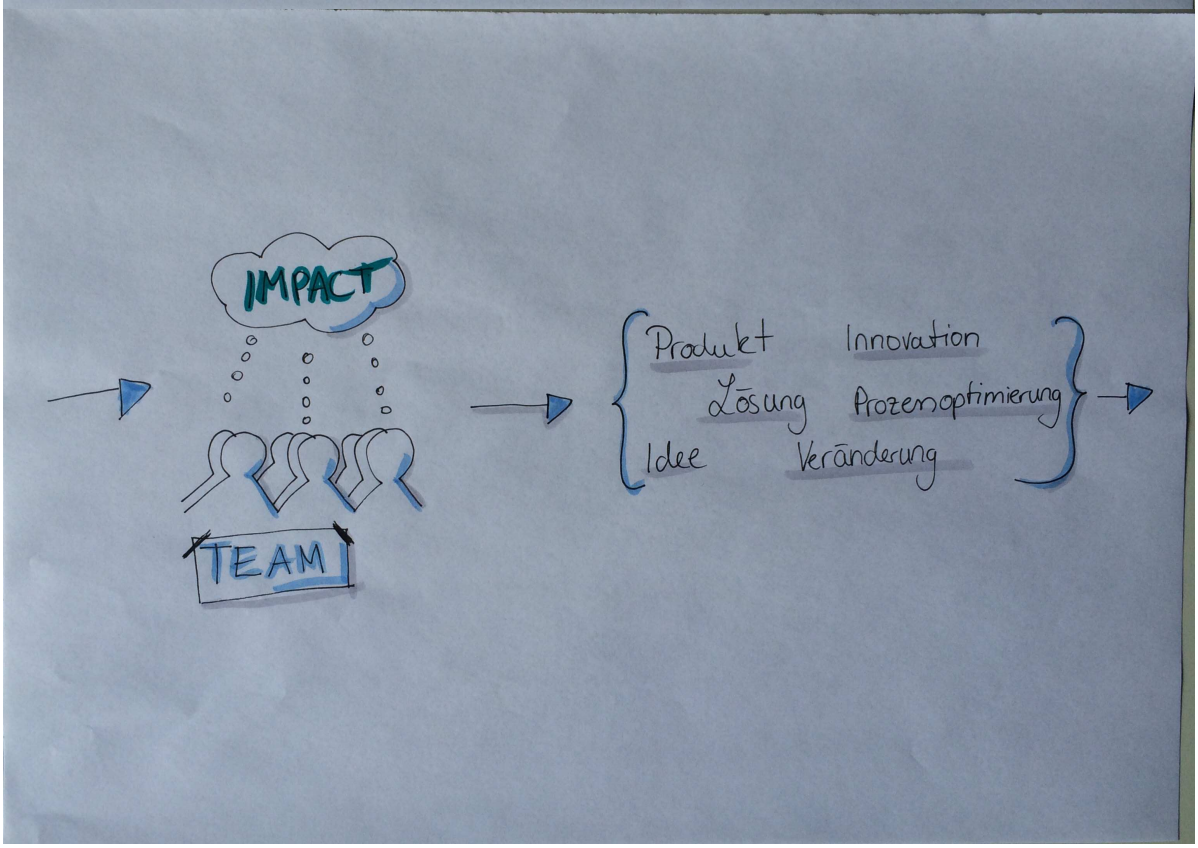
A.3.5. Erstellung des CeBIT-Teasers

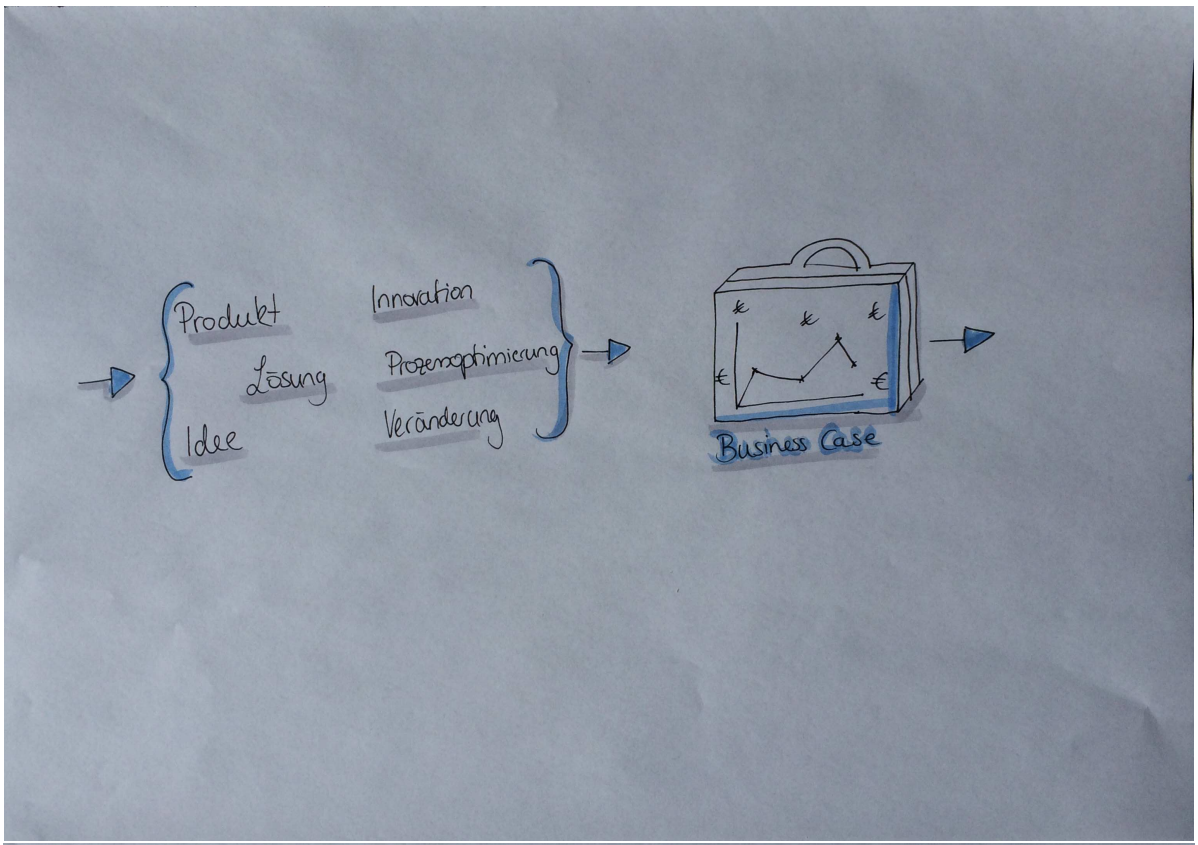


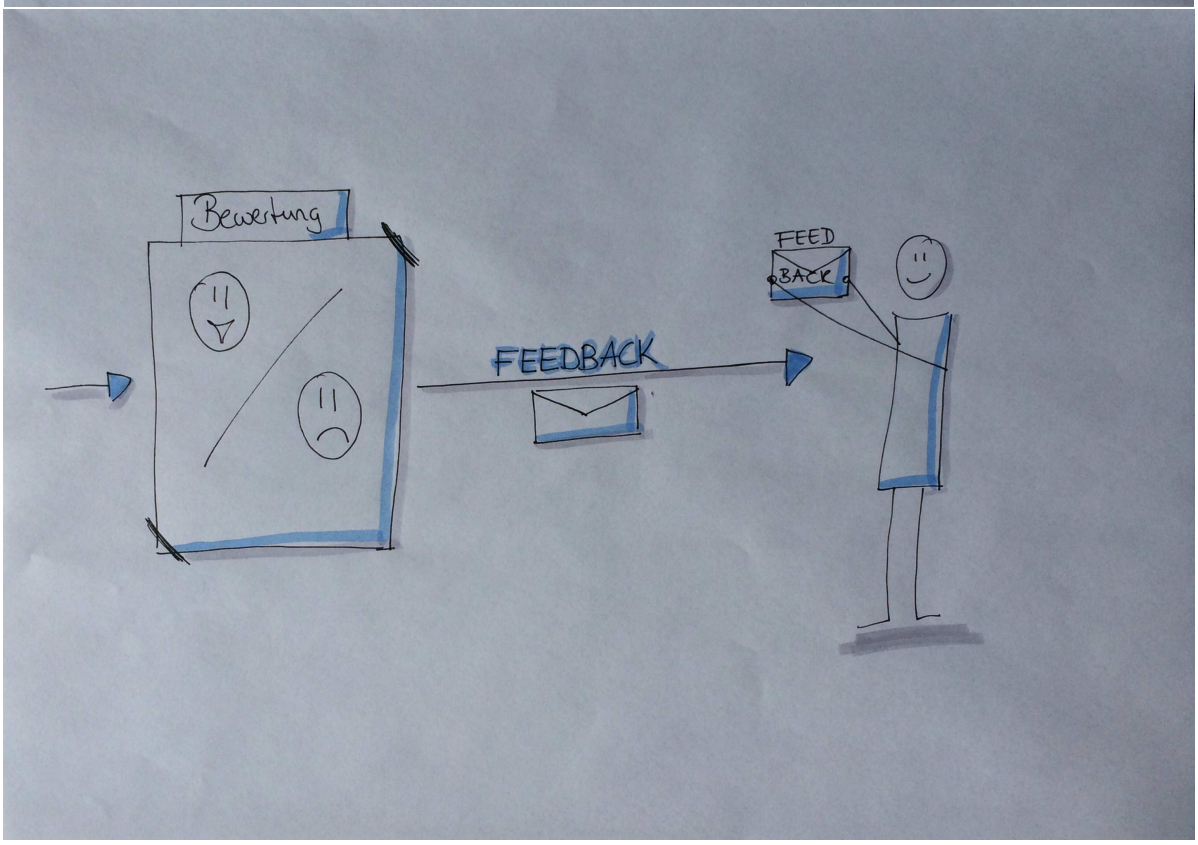

Bisher
KVP
MÜLL

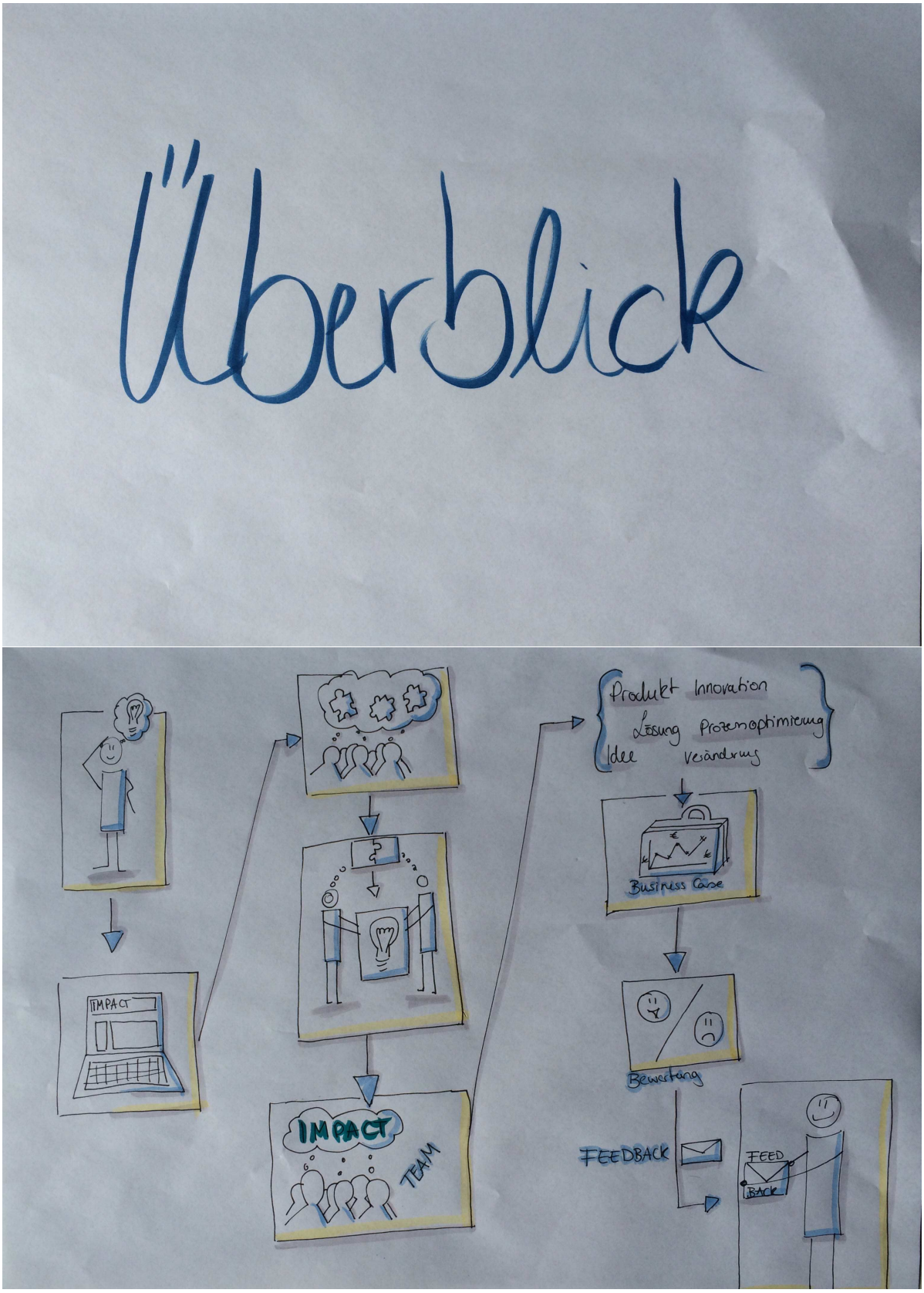
Heute
IMPACT

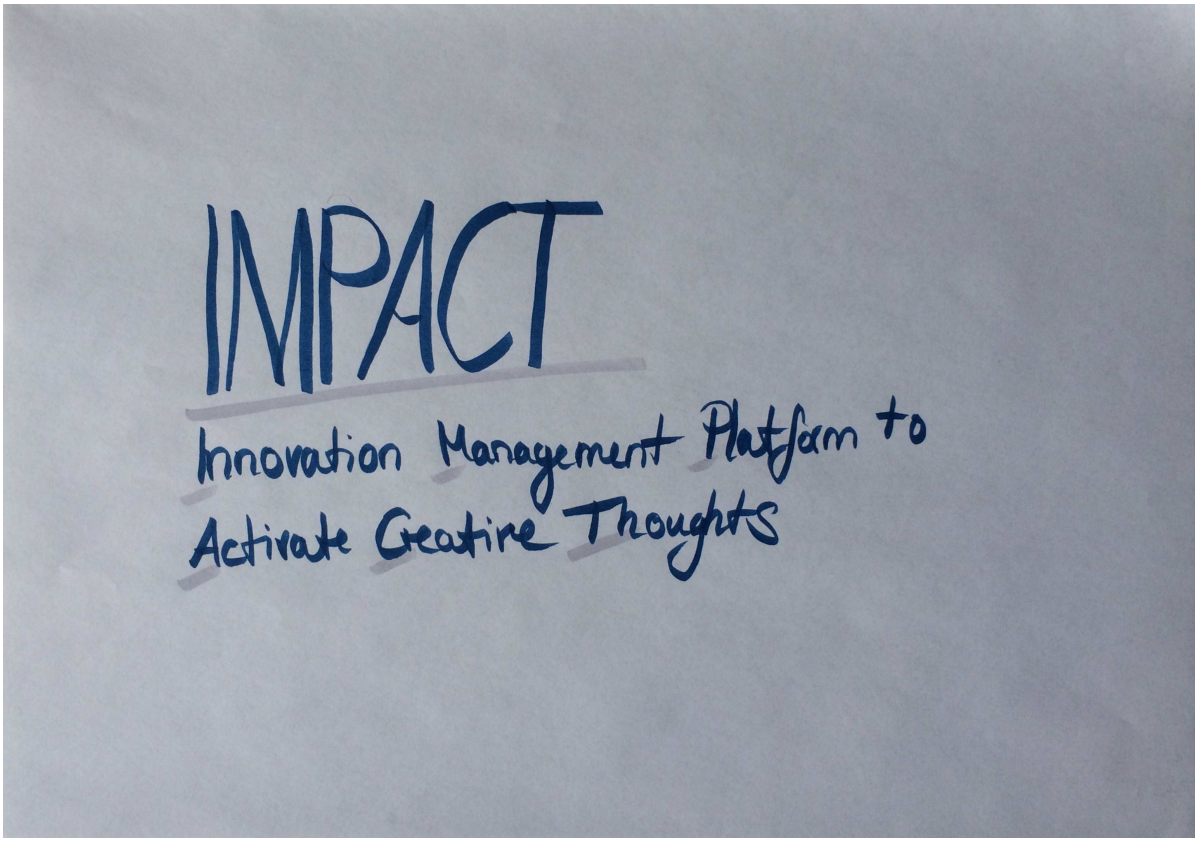



IMPACT
TEAM
Produkt
Innovation
Lösung
Prozessoptimierung
Idee
Veränderung


Produkt
Lösung
Idee
Innovation
Prozessoptimierung
Veränderung
Business Case


Bewertung
FEEDBACK
FEED
BACK


Überblick
IMPACT
IMPACT
TEAM
Produkt Innovation
Lösung Prozessoptimierung
Idee Veränderung
Business Case
Bewertung
FEEDBACK
FEED
BACK


IMPACT
Innovation Management Platform to
Activate Creative Thoughts

A.4. Wirefreames

Im Rahmen der Konzeptionierungsphase wurden eine Vielzahl von Wireframes erstellt, um einen ersten Einblick zu bekommen, wie die Plattform später aussehen könnte. Die nachfolgenden Wireframes stellen einen Auszug der angefertigen Wireframes dar.

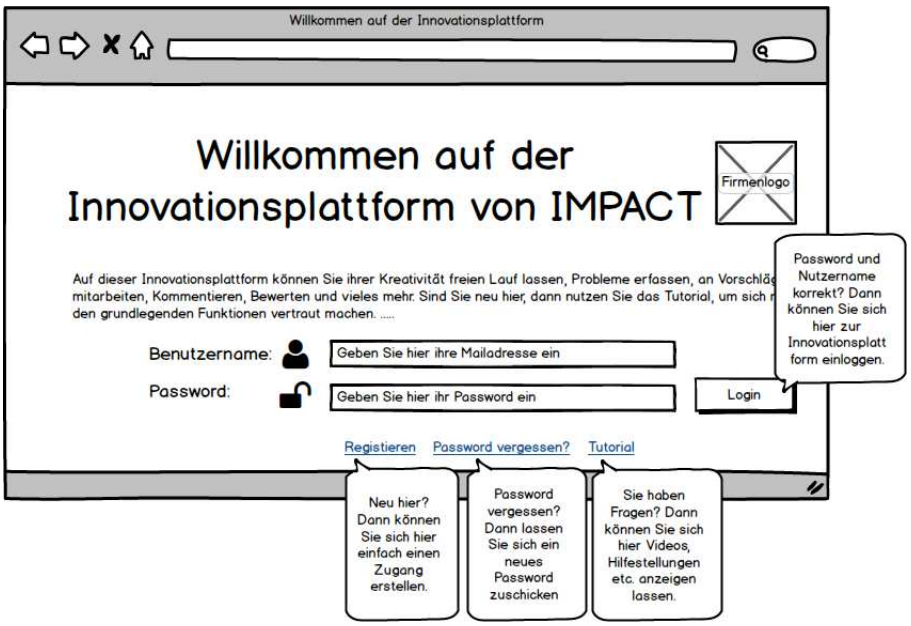


Abbildung 102: Wireframe: Startseite

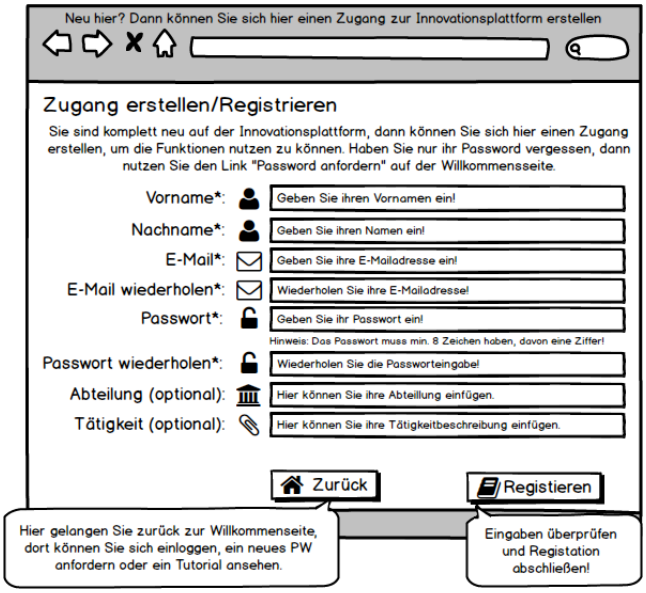


Abbildung 103: Wireframe: Registrierung

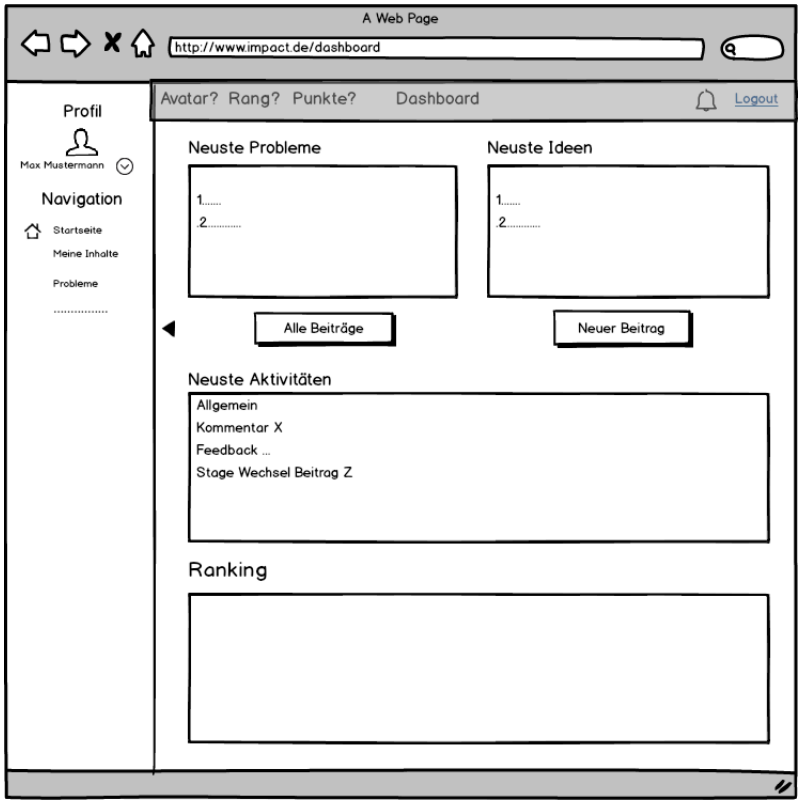


Abbildung 104: Wireframe: Lobby

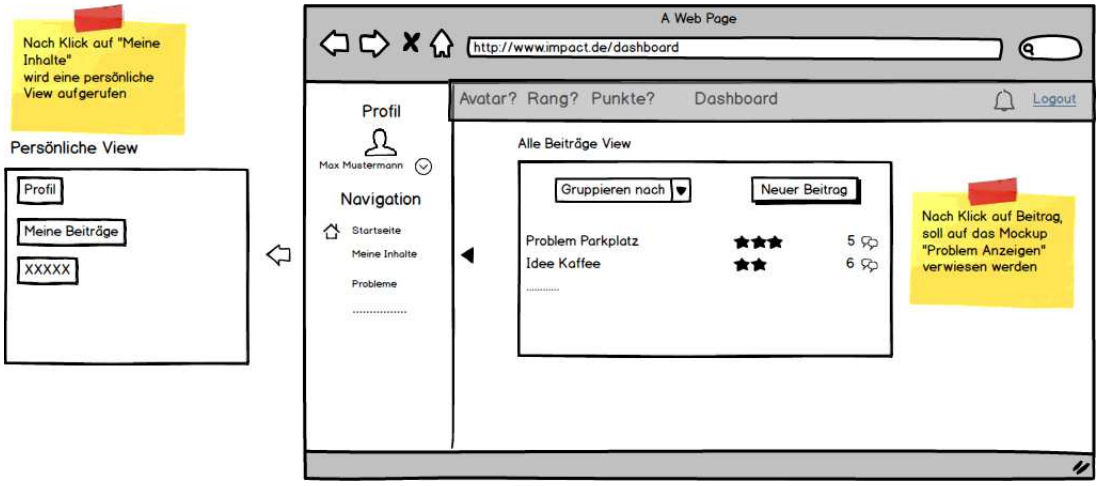


Abbildung 105: Wireframe: Beitrag

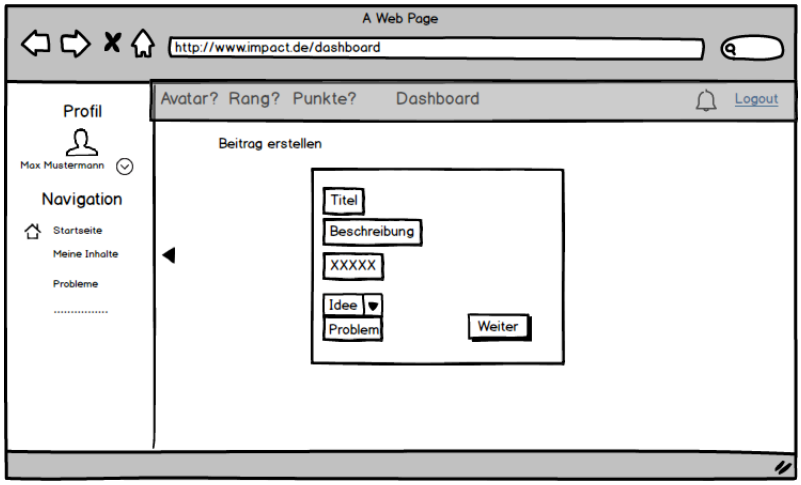


Abbildung 106: Wireframe: Beitrag erstellen

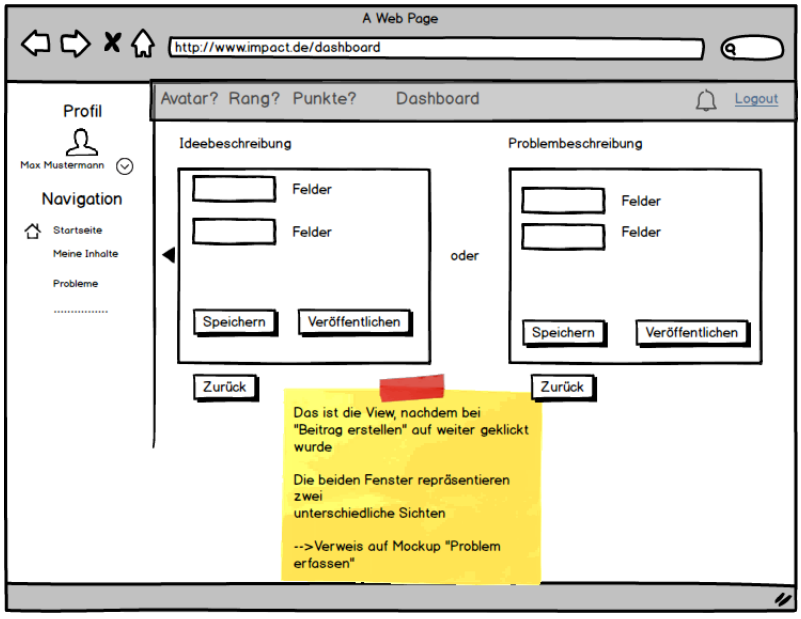


Abbildung 107: Wireframe: Idee und Problem

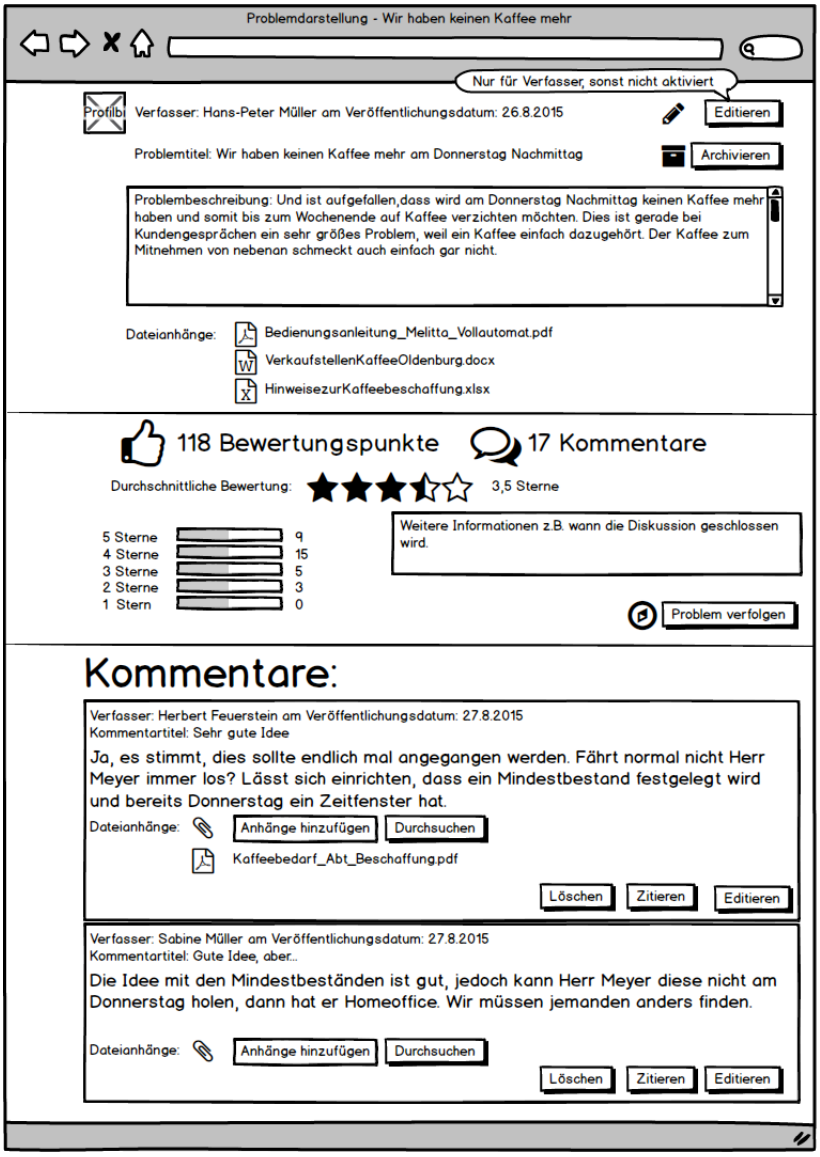


Abbildung 108: Wireframe: Problem anzeigen

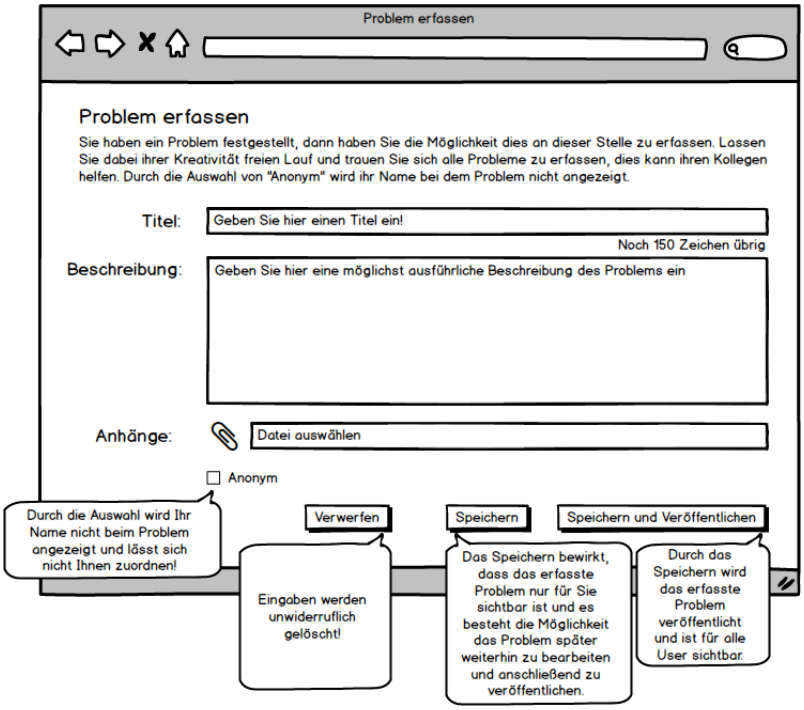


Abbildung 109: Wireframe: Problem erfassen

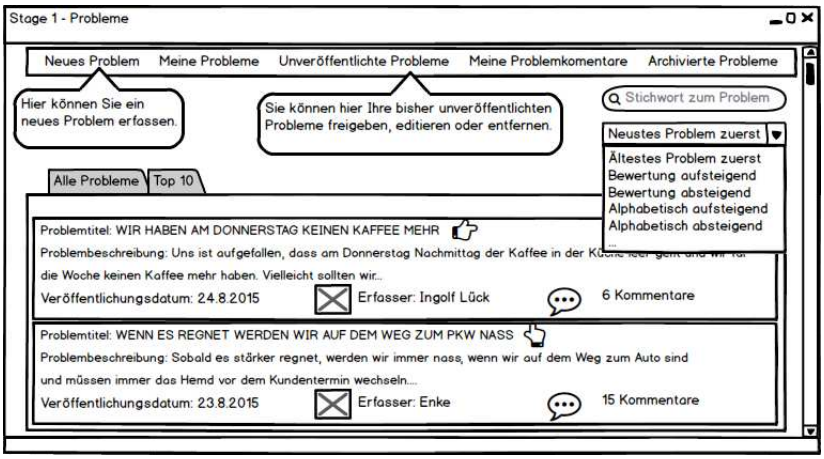


Abbildung 110: Wireframe: Problemansicht

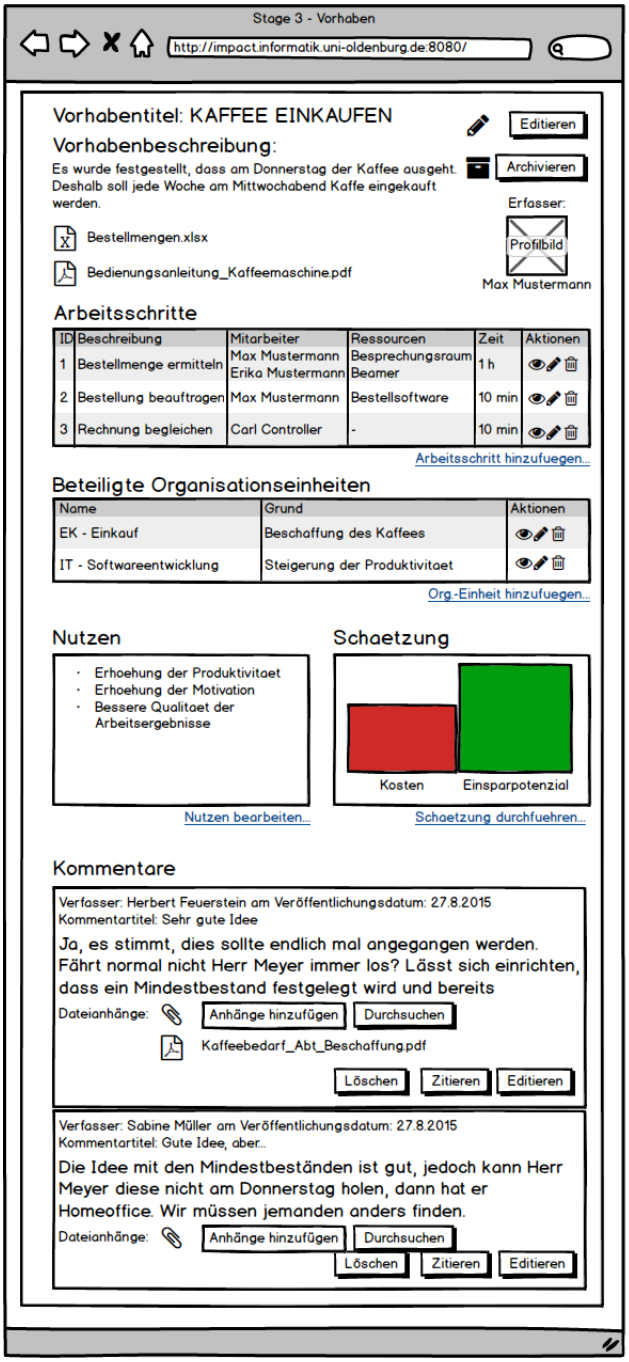


Abbildung 111: Wireframe: Vorhabendetails

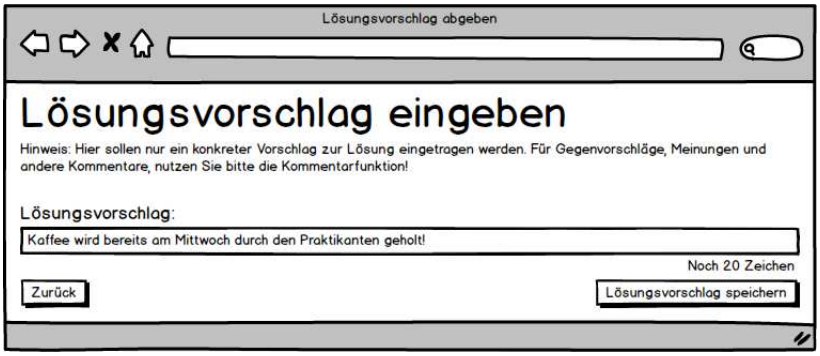


Abbildung 112: Wireframe: Lösungsvorschlag eingeben

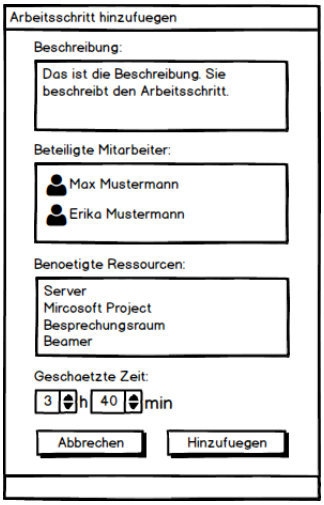


Abbildung 113: Wireframe: Arbeitsschritt anlegen

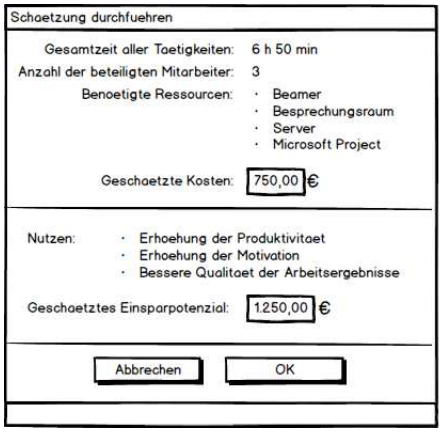


Abbildung 114: Wireframe: Schätzung durchführen

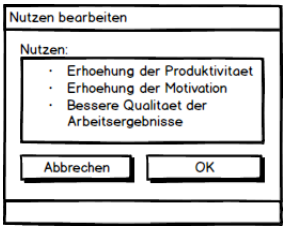


Abbildung 115: Wireframe: Nutzen bearbeiten

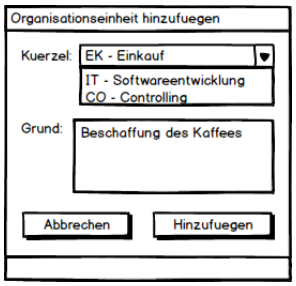


Abbildung 116: Wireframe: Organisationseinheit hinzufügen

A.5. User Stories

Kritierium	Ausführung
Name der Story	Challenge erfassen
Beschreibung	Als Benutzer möchte ich auf der Plattform meine Challenge erfassen, um diese anderen Benutzern zugänglich zu machen
Akzeptanzkriterien:	Ich kann meine Challenge auf der Plattform mit Hilfe eines Formulars eintragen Meine Challenge wird gespeichert Andere Benutzer können meine Challenge aufrufen Ich kann meine Challenge anonym veröffentlichen

Tabelle 39: Userstory für „Challenge erfassen“

Kritierium	Ausführung
Name der Story	Challenge editieren
Beschreibung	Als Benutzer möchte ich auf der Plattform meine Challenge editieren, um Änderungen vorzunehmen
Akzeptanzkriterien:	Ich kann meine Challenge auf der Plattform editieren
	Meine editierte Challenge wird gespeichert
	Andere Benutzer können meine editierte Challenge aufrufen

Tabelle 40: Userstory für „Challenge editieren“

Kritierium	Ausführung
Name der Story	Challengeübersicht
Beschreibung	Als Benutzer möchte ich auf der Plattform eine Challengeübersicht, damit ich nach anderen Challenges suchen und ansehen kann
Akzeptanzkriterien:	Ich kann in der Challengeübersicht alle Challenges sehen
	Ich kann nach wesentlichen Kriterien suchen und Filtern
	Ich kann von der Übersicht aus einzelne Challenges öffnen

Tabelle 41: Userstory für „Challengeübersicht“

Kritierium	Ausführung
Name der Story	Challengehistorie
Beschreibung	Als Benutzer möchte ich die Historie einer Challenge öffnen, damit ich alle Änderungen nachvollziehen kann
Akzeptanzkriterien:	Die Historie einer Challenge wird gespeichert
	Ich kann die Historie einer Challenge, wenn benötigte Rechte vorhanden sind, öffnen

Tabelle 42: Userstory für „Challengehistorie“

Kritierium	Ausführung
Name der Story	Lösungsvorschlag erfassen
Beschreibung	Als Benutzer möchte ich einen Lösungsvorschlag für eine Challenge erfassen, damit andere Benutzer den bewerten und kommentieren können
Akzeptanzkriterien:	Ich kann meinen Lösungsvorschlag auf der Plattform mit Hilfe eines Formulars eintragen Mein Lösungsvorschlag wird gespeichert Andere Benutzer können meinen Lösungsvorschlag aufrufen Ich kann meinen Lösungsvorschlag anonym veröffentlichen

Tabelle 43: Userstory für „Lösungsvorschlag erfassen“

Kritierium	Ausführung
Name der Story	Lösungsvorschlag editieren
Beschreibung	Als Benutzer möchte ich auf der Plattform meinen Lösungsvorschlag editieren, um Änderungen vorzunehmen
Akzeptanzkriterien:	Ich kann meinen Lösungsvorschlag auf der Plattform editieren Meine editierter Lösungsvorschlag wird gespeichert Andere Benutzer können meinen editierten Lösungsvorschlag aufrufen

Tabelle 44: Userstory für „Lösungsvorschlag editieren“

Kritierium	Ausführung
Name der Story	Projekt erfassen
Beschreibung	Als Benutzer möchte ich auf der Plattform ein Projekt erfassen, um dieses anderen Benutzern zugänglich zu machen
Akzeptanzkriterien:	Ich kann ein Projekt auf der Plattform mit Hilfe eines Formulars eintragen Mein Projekt wird gespeichert Andere Benutzer können das Projekt aufrufen

Tabelle 45: Userstory für „Projekt erfassen“

Kritierium	Ausführung
Name der Story	Projekt editieren
Beschreibung	Als Benutzer möchte ich auf der Plattform ein Projekt editieren, um Änderungen vorzunehmen
Akzeptanzkriterien:	Ich kann ein Projekt auf der Plattform editieren
	Meine editiertes Projekt wird gespeichert
	Andere Benutzer können das editierte Projekt aufrufen

Tabelle 46: Userstory für „Projekt editieren“

Kritierium	Ausführung
Name der Story	Business Case erfassen
Beschreibung	Als Benutzer möchte ich auf der Plattform einen Business Case erfassen, um dieses anderen Benutzern zugänglich zu machen
Akzeptanzkriterien:	Ich kann einen Business Case auf der Plattform mit Hilfe eines Formulars eintragen
	Mein Business Case wird gespeichert
	Andere Benutzer können den Business Case aufrufen

Tabelle 47: Userstory für „Business Case erfassen“

Kritierium	Ausführung
Name der Story	Business Case freigeben
Beschreibung	Als Benutzer möchte ich auf der Plattform einen Business Case freigeben, um diesen für die Entscheidung freizugeben
Akzeptanzkriterien:	Ich kann einen Business Case auf der Plattform freigeben
	Mein Business Case wird für die Entscheidung freigegeben
	Andere Benutzer können den Business Case aufrufen

Tabelle 48: Userstory für „Business Case freigeben“

Kritierium	Ausführung
Name der Story	Business Case Entscheidung
Beschreibung	Als Benutzer möchte ich auf der Plattform einen Business Case bewerten, um diesen an- oder ablehnen zu können
Akzeptanzkriterien:	Ich kann einen Business Case auf der Plattform an- oder ablehnen Andere Benutzer können den Business Case aufrufen

Tabelle 49: Userstory für „Business Case Entscheidung“

Kritierium	Ausführung
Name der Story	Feedback erfassen
Beschreibung	Als Benutzer möchte ich auf der Plattform ein Feedback erfassen, um dieses anderen Benutzern zugänglich zu machen
Akzeptanzkriterien:	Ich kann ein Feedback auf der Plattform mit Hilfe eines Formulars eintragen Mein Feedback wird gespeichert Andere Benutzer können das Feedback aufrufen

Tabelle 50: Userstory für „Feedback erfassen“

Kriterium	Ausführung
Name der Story	Registrierung
Beschreibung	Als Benutzer möchte ich mich auf der Plattform registrieren, um Zugriff auf die Plattform zu erhalten
Akzeptanzkriterien:	Ich kann mich auf der Plattform mit Hilfe eines Registrierungsformulars registrieren In dem Registrierungsprozess kann ich alle benötigten Daten eintragen Die eingetragenen Daten werden auf Plausibilität überprüft

Tabelle 51: Userstory für „Registrierung“

Kriterium	Ausführung
Name der Story	Profil bearbeiten
Beschreibung	Als Benutzer möchte ich mein Profil bearbeiten, damit ich selbstständig Änderungen einpflegen kann
Akzeptanzkriterien:	Ich kann mein Profil bearbeiten Ich kann die Änderungen in meinem Profil speichern

Tabelle 52: Userstory für „Profil Bearbeitung“

Kriterium	Ausführung
Name der Story	Anlegen von Benutzern
Beschreibung	Als Administrator möchte ich die Möglichkeit haben Benutzer anzulegen, damit diese Zugriff auf die Plattform erhalten
Akzeptanzkriterien:	Ich kann im Backend mittels eines Formulars neue Benutzer anlegen In dem Anlageprozess kann ich alle benötigten Daten eintragen Die eingetragenen Daten werden auf Plausibilität überprüft Für die neu angelegten Benutzer kann ich ein Initial-Passwort setzen

Tabelle 53: Userstory für „Benutzer anlegen“

Kriterium	Ausführung
Name der Story	Umgang mit Passwörtern
Beschreibung	Als Benutzer möchte ich das meine Passwörter sicher übertragen und aufbewahrt werden, damit ein Missbrauch durch Dritte im Rahmen technischer Möglichkeiten verhindert wird
Akzeptanzkriterien:	Das System muss fähig sein die Passwörter verschlüsselt zu übertragen Das System muss fähig sein die Passwörter verschlüsselt zu speichern

Tabelle 54: Userstory für „Umgang mit Passwoertern“

Kritierium	Ausführung
Name der Story	Datei-Upload
Beschreibung	Als Benutzer möchte ich Dateien hochladen, um diese anderen Benutzern zur Verfügung zu stellen.
Akzeptanzkriterien:	- Ich kann verschiedene Datei-Typen hinzufügen (*.pdf, *.docx, *.xlsx) Ich kann mehrere Dateien hochladen Ich kann mehrere Dateien zeitgleich hochladen (Multi-Upload) Die Dateien werden gespeichert und sind auch für andere Benutzer abrufbar

Tabelle 55: Userstory für „Dateiupload“

Kriterium	Ausführung
Name der Story	Kompatibilität auf mobilen Endgeräten
Beschreibung	Als Benutzer möchte ich die Plattform auf unterschiedlichen mobilen Endgeräten aufrufen, damit ein Arbeiten von unterwegs sichergestellt ist
Akzeptanzkriterien:	Die wichtigsten Funktionen müssen auf Smartphone, Tablet und Notebook lauffähig sein Die Plattform ist auf unterschiedlichen Browsern lauffähig Die Elemente auf der Plattform werden entsprechend der Auflösung skaliert

Tabelle 56: Userstory für „Kompatibilität auf mobilen Endgeräten“

Kriterium	Ausführung
Name der Story	Personalisierung
Beschreibung	Als Benutzer möchte ich die Plattform auf meine Bedürfnisse anpassen, damit ein effizientes Arbeiten möglich ist
Akzeptanzkriterien:	Die Plattform bietet im Standard heraus verschiedene Designs an Die wichtigen Elemente auf der Plattform lassen sich anpassen (Größe, Ausrichtung, Design) Der Benutzer kann Themen oder Bereiche abonnieren Der Benutzer wird über Neuigkeiten der abonnierten Themen/-Bereiche informiert

Tabelle 57: Userstory für „Personalisierung“

Kriterium	Ausführung
Name der Story	Gamification
Beschreibung	Als Benutzer möchte ich das Gamification Elemente auf der Plattform integriert werden, damit ein motivierendes und spielerisches Arbeiten möglich ist
Akzeptanzkriterien:	Es werden Elemente aus dem Gamification Umfeld in die Plattform integriert Der Benutzer kann steuern, welche persönlichen Gamification Elemente aktiviert sind

Tabelle 58: Userstory für „Gamification Elemente“

Kriterium	Ausführung
Name der Story	Design Thinking
Beschreibung	Als Benutzer möchte ich mit anderen Benutzern zusammenarbeiten, damit gemeinsam Ideen und Lösungen entwickelt werden können
Akzeptanzkriterien:	Die Plattform bietet die Möglichkeit gemeinsam an Problemen und Lösungen zu arbeiten Es werden Elemente zur kollaborativen Zusammenarbeit bereit gestellt Die Plattform bietet die Möglichkeit mehrere Lösungsvorschläge zu kombinieren

Tabelle 59: Userstory für „Design Thinking Elemente“

Kriterium	Ausführung
Name der Story	Rating Funktion
Beschreibung	Als Benutzer möchte ich Challenges und Lösungsvorschläge bewerten, damit ich meine persönliche Relevanz des Objektes ausdrücken kann
Akzeptanzkriterien:	Die Plattform bietet die Möglichkeit Bewertungen für Challenges und Lösungsvorschläge abzugeben Die Plattform bietet die Möglichkeit alle Bewertungen einzusehen Die Plattform bietet die Möglichkeit Bewertungen anonym zu erfassen Die Plattform besitzt einen nachvollziehbaren Bewertungsalgorithmus

Tabelle 60: Userstory für „Rating Funktion“

Kriterium	Ausführung
Name der Story	Kommentarfunktion
Beschreibung	Als Benutzer möchte ich Kommentare zu Challenges und Lösungsvorschläge erfassen, damit ich meine Meinung sowie nützliche Informationen ausdrücken kann
Akzeptanzkriterien:	Die Plattform bietet die Möglichkeit Kommentare zu Challenges und Lösungsvorschläge abzugeben Die Plattform bietet die Möglichkeit alle Kommentare einzusehen Die Plattform bietet die Möglichkeit Kommentare anonym zu erfassen

Tabelle 61: Userstory für „Kommentarfunktion“

Kriterium	Ausführung
Name der Story	Tag-Cloud
Beschreibung	Als Benutzer möchte ich bei dem Erstellen von Challenges und Lösungsvorschläge Tags hinzufügen, damit ich diesen bestimmte Schlagwörter zuordnen kann
Akzeptanzkriterien:	Die Plattform bietet die Möglichkeit bei dem Erstellen von Challenges und Lösungsvorschlägen Tags hinzuzufügen Die Plattform bietet die Möglichkeit Tags anzuklicken, um alle Challenges/Lösungsvorschläge zu diesem Schlagwort anzeigen zu lassen Die Plattform erstellt eine Tag-Cloud

Tabelle 62: Userstory für „Tag-Cloud“

Kriterium	Ausführung
Name der Story	kollaborativer Raum
Beschreibung	Als Benutzer möchte ich mit anderen Benutzern zusammen arbeiten, damit gemeinsam Lösungen und Konzepte entwickelt werden können
Akzeptanzkriterien:	Die Plattform bietet einen kollaborativen Raum für die Zusammenarbeit Der kollaborative Raum soll ein Whiteboard für gemeinsame Zeichnungen besitzen Das Whiteboard muss die Möglichkeit bieten Zeichnungen zu speichern

Tabelle 63: Userstory für „kollaborativen Raum“

Kriterium	Ausführung
Name der Story	Video-Telefonie
Beschreibung	Als Benutzer möchte ich mit anderen Benutzern eine Video-Telefonie führen, damit Gespräche unabhängig vom Ort geführt werden können
Akzeptanzkriterien:	Die Plattform bietet die Möglichkeit zur Video-Telefonie Die Plattform bietet die Möglichkeit eine Kontaktliste für die Video-Telefonie zu pflegen Die Plattform bietet die Möglichkeit Aufzeichnungen der Video-Telefonie zu speichern

Tabelle 64: Userstory für „Video-Telefonie“

Kriterium	Ausführung
Name der Story	Chat
Beschreibung	Als Benutzer möchte ich mit anderen Benutzern Chatten können, damit eine Plattform-interne Kommunikation möglich ist
Akzeptanzkriterien:	Die Plattform bietet die Möglichkeit zum Chatten Die Plattform bietet die Möglichkeit eine Kontaktliste für den Chat zu pflegen Die Plattform bietet die Möglichkeit einer Chat-Historie

Tabelle 65: Userstory für „Chat“

Kriterium	Ausführung
Name der Story	Ideenbörse
Beschreibung	Als Benutzer möchte ich meine Ideen bzw. Lösungsvorschläge vermarkten, damit über Crowdfunding die benötigten Ressourcen für die Umsetzung generiert werden können
Akzeptanzkriterien:	Die Plattform bietet die Möglichkeit meine Lösungsvorschläge auf einer Ideenbörse zu vermarkten Die Plattform bietet die Möglichkeit, dass Mitarbeiter eine virtuelle Währung auf Lösungsvorschläge setzen können

Tabelle 66: Userstory für „Ideenbörse“

Kriterium	Ausführung
Name der Story	Mitarbeiterbörse
Beschreibung	Als Benutzer möchte ich meine Arbeitskraft und Wissen anbieten, damit Projektverantwortliche die Möglichkeit haben entsprechendes Personal zu finden
Akzeptanzkriterien:	Die Plattform bietet die Möglichkeit meine Fähigkeiten und Interessen zu erfassen Die Plattform bietet die Möglichkeit eine Übersicht aller teilnehmenden Benutzer anzuzeigen Die Plattform bietet die Möglichkeit nach Fähigkeiten und Interessen zu suchen Die Plattform bietet die Möglichkeit Benutzer zu Projekten einzuladen

Tabelle 67: Userstory für „Mitarbeiterbörse“

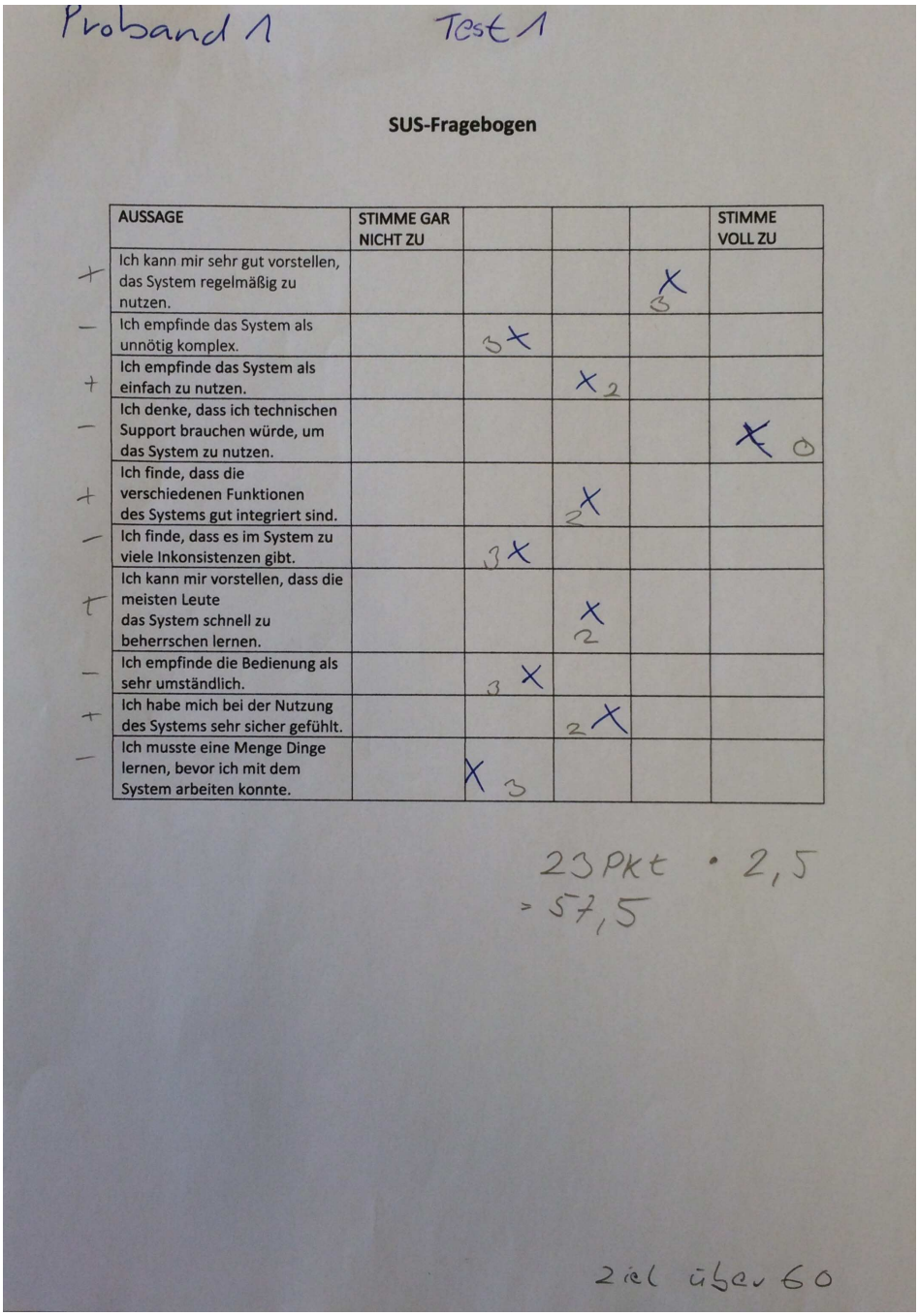
A.6. Usability-Tests

A.6.1. Leitfaden zur Einführung der Testprobanden

Einführung in das Testszenario (Usability-Test)

- Vorstellung der Durchzuführenden
  - Namen,
  - PG
- Vorstellung des PG Schwerpunktes
  - Erstellen einer Plattform zum innerbetrieblichen Innovationsmanagement in Verbindung mit der Mitarbeitermotivation
  - PG ist in Abt. VLBA verankert und führt das Projekt mit der Lufthansa Solution GmbH
- Einweisung in den Testablauf
  - Usability Test
  - Wie intuitiv lässt sich die Plattform bedient
  - Nicht fertiger Prototyp (Zu testende Funktionen sollen als bereits fertige Funktionen betrachtet werden)
  - Um neue Funktionalitäten abzuleiten
  - Produktverbesserung
- Einführung in das System
  - Challenge
  - Lösungsvorschlag
  - Projekt
  - Bewertungen (Stars und geführt)
- Ablauf
  - Bildschirmaufzeichnung
  - Laut denken
  - Sie werden beobachtet
  - Anschließend einen Fragebogen beantworten
- Danke und Auf Wiedersehen.

A.6.2. Auswertung der SUS-Fragebögen des ersten Testdurchgangs



Proband 1 Test 1

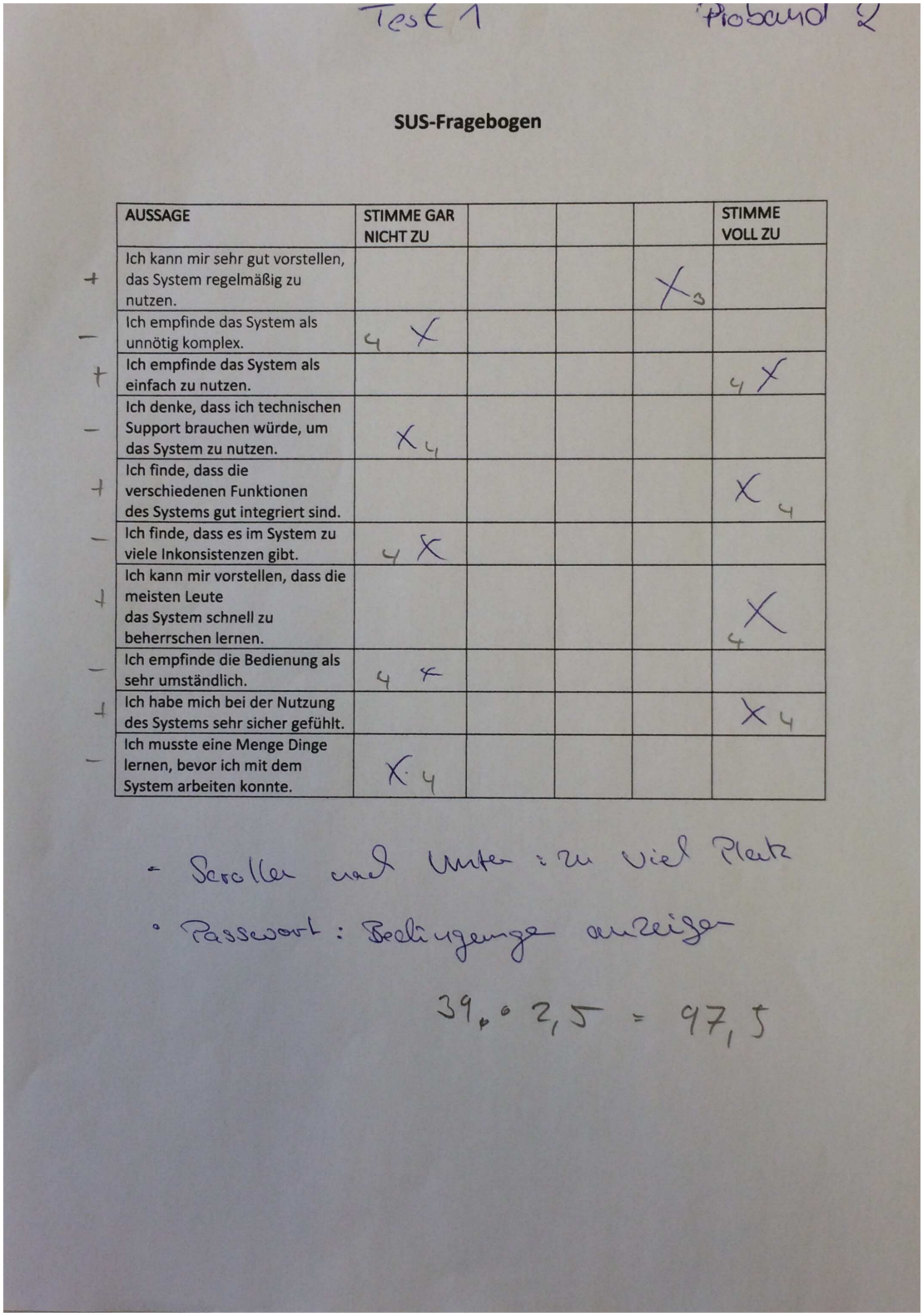
SUS-Fragebogen

	AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU
+	Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.				X 3	
−	Ich empfinde das System als unnötig komplex.		3 X			
+	Ich empfinde das System als einfach zu nutzen.			X 2		
−	Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.					X 0
+	Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.			X 2		
−	Ich finde, dass es im System zu viele Inkonsistenzen gibt.		3 X			
+	Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.			X 2		
−	Ich empfinde die Bedienung als sehr umständlich.		3 X			
+	Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.			2 X		
−	Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.		X 3			

23 Pkt · 2,5
= 57,5

Ziel über 60

Abbildung 117: Auswertung des SUS-Fragebogen von Proband 1



Test 1 Proband 2

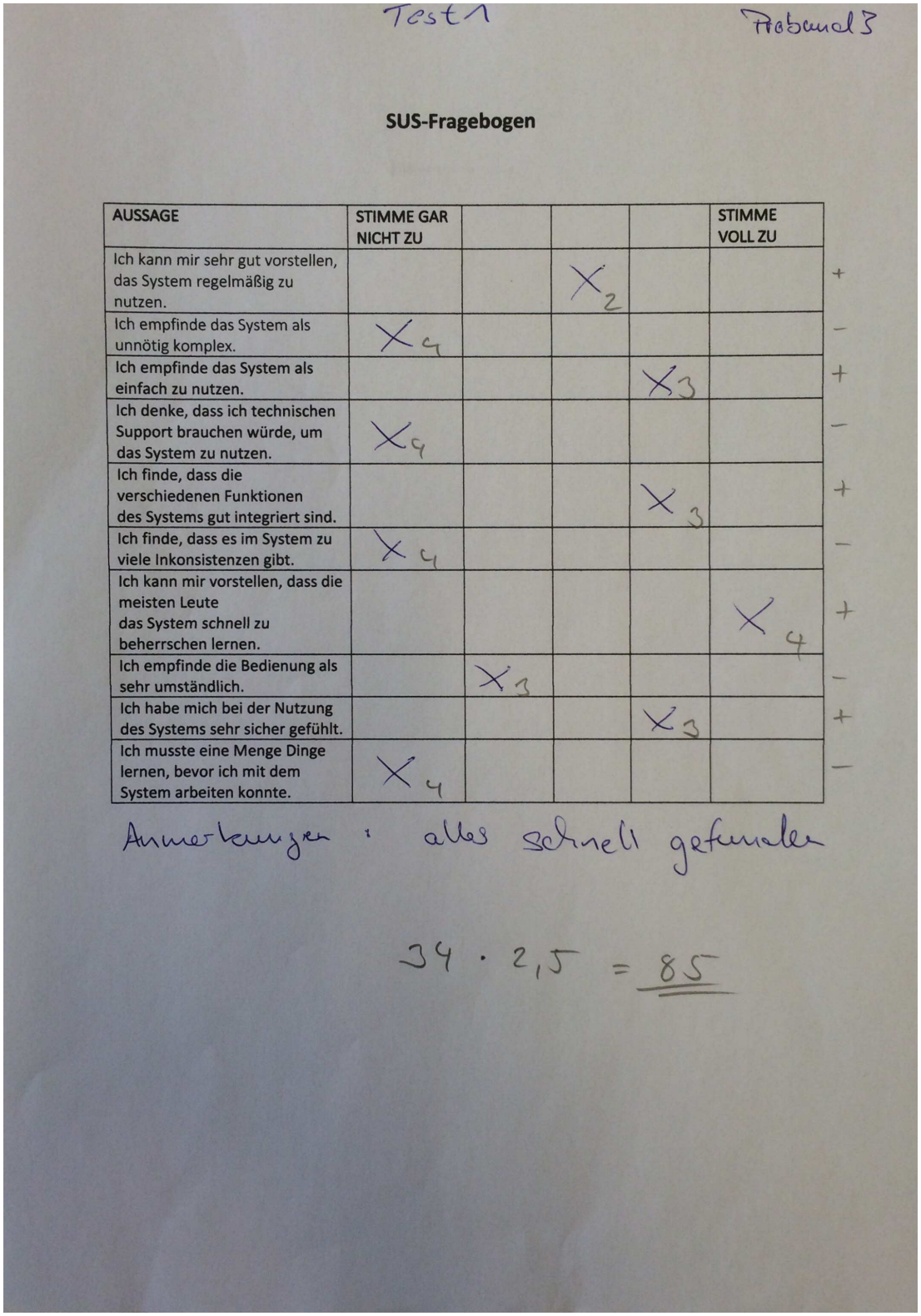
SUS-Fragebogen

	AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU
+	Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.				X 3	
−	Ich empfinde das System als unnötig komplex.	4 X				
+	Ich empfinde das System als einfach zu nutzen.					4 X
−	Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	X 4				
+	Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.					X 4
−	Ich finde, dass es im System zu viele Inkonsistenzen gibt.	4 X				
+	Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.					4 X
−	Ich empfinde die Bedienung als sehr umständlich.	4 X				
+	Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.					X 4
−	Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	X 4				

- Scrollen nach Unten : zu viel Platz
- Passwort : Bedingungen anzeigen

39 · 2,5 = 97,5

Abbildung 118: Auswertung des SUS-Fragebogen von Proband 2



Test 1 Proband 3

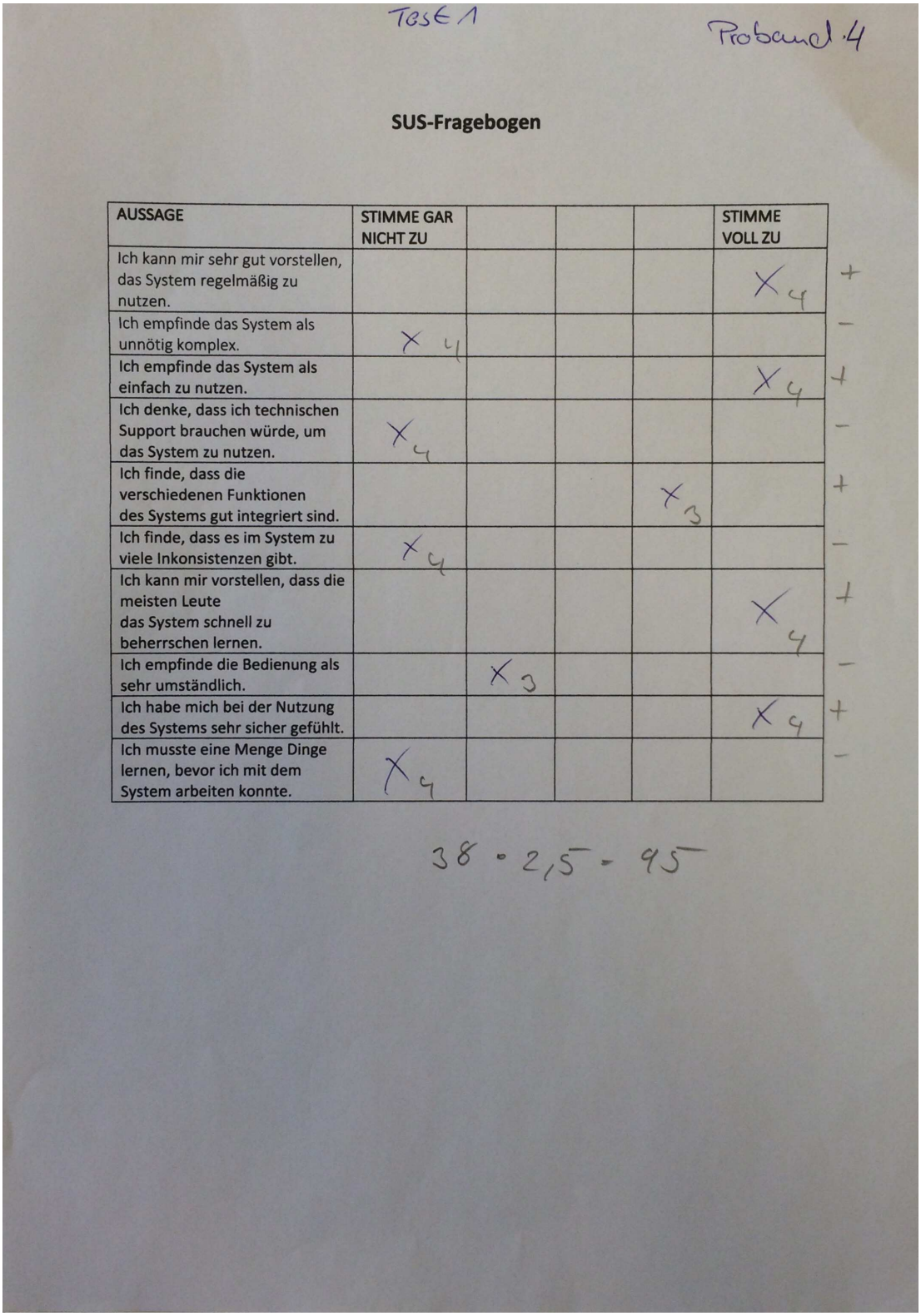
SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU	
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.			X 2			+
Ich empfinde das System als unnötig komplex.	X 4					−
Ich empfinde das System als einfach zu nutzen.				X 3		+
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	X 4					−
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.				X 3		+
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	X 4					−
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.					X 4	+
Ich empfinde die Bedienung als sehr umständlich.		X 3				−
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.				X 3		+
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	X 4					−

Anmerkungen: alles schnell gefunden

34 · 2,5 = 85

Abbildung 119: Auswertung des SUS-Fragebogen von Proband 3



Test 1

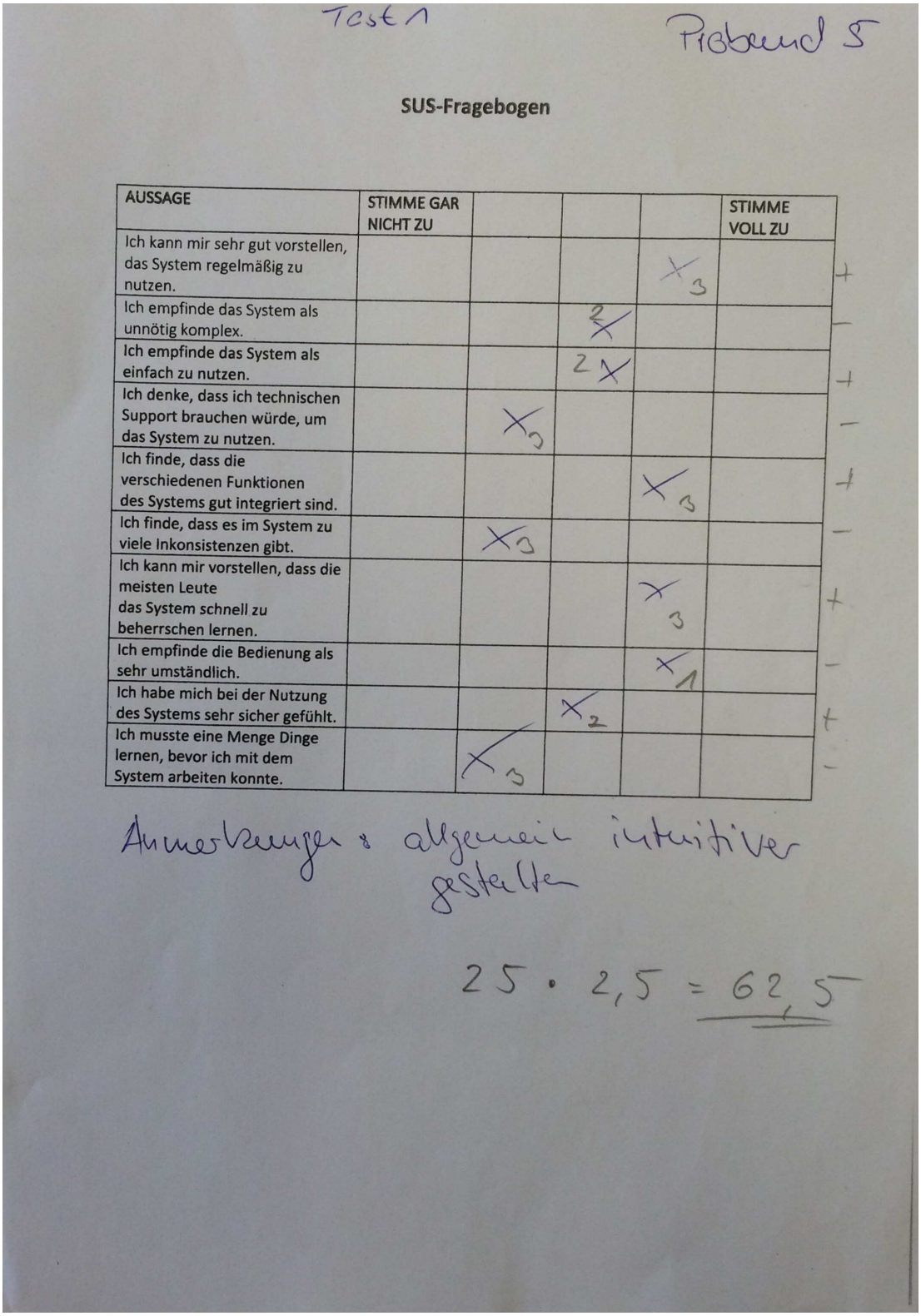
Proband 4

SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU	
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.					X 4	+
Ich empfinde das System als unnötig komplex.	X 4					−
Ich empfinde das System als einfach zu nutzen.					X 4	+
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	X 4					−
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.				X 3		+
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	X 4					−
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.					X 4	+
Ich empfinde die Bedienung als sehr umständlich.		X 3				−
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.					X 4	+
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	X 4					−

38 · 2,5 = 95

Abbildung 120: Auswertung des SUS-Fragebogen von Proband 4



Test 1

Probend 5

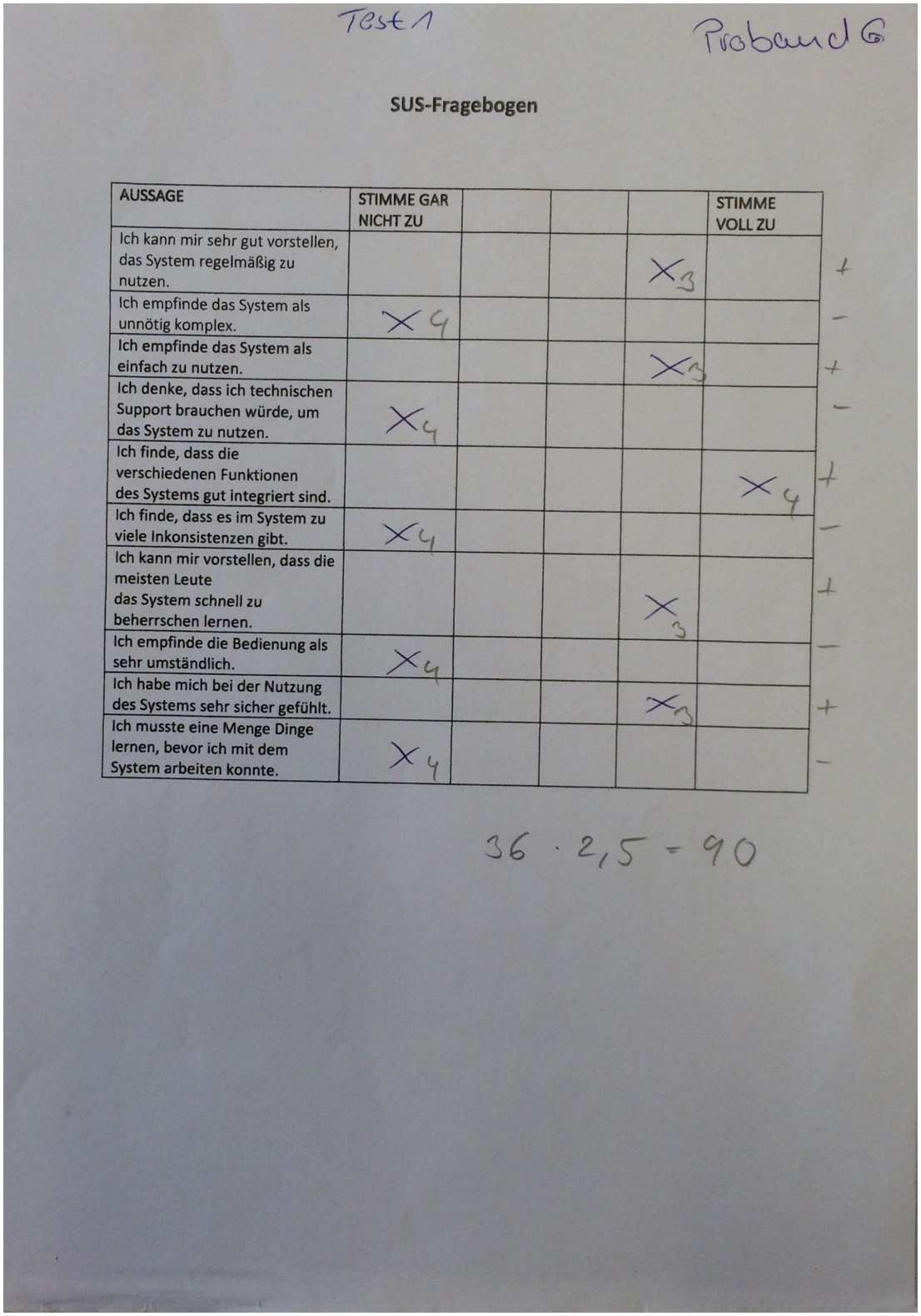
SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU	
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.				X 3		+
Ich empfinde das System als unnötig komplex.			2 X			−
Ich empfinde das System als einfach zu nutzen.			2 X			+
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.		X 3				−
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.				X 3		+
Ich finde, dass es im System zu viele Inkonsistenzen gibt.		X 3				−
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.				X 3		+
Ich empfinde die Bedienung als sehr umständlich.				X 1		−
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.			X 2			+
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.		X 3				−

Anmerkungen: allgemein intuitiver gestalten

25 · 2,5 = 62,5

Abbildung 121: Auswertung des SUS-Fragebogen von Proband 5



Test 1

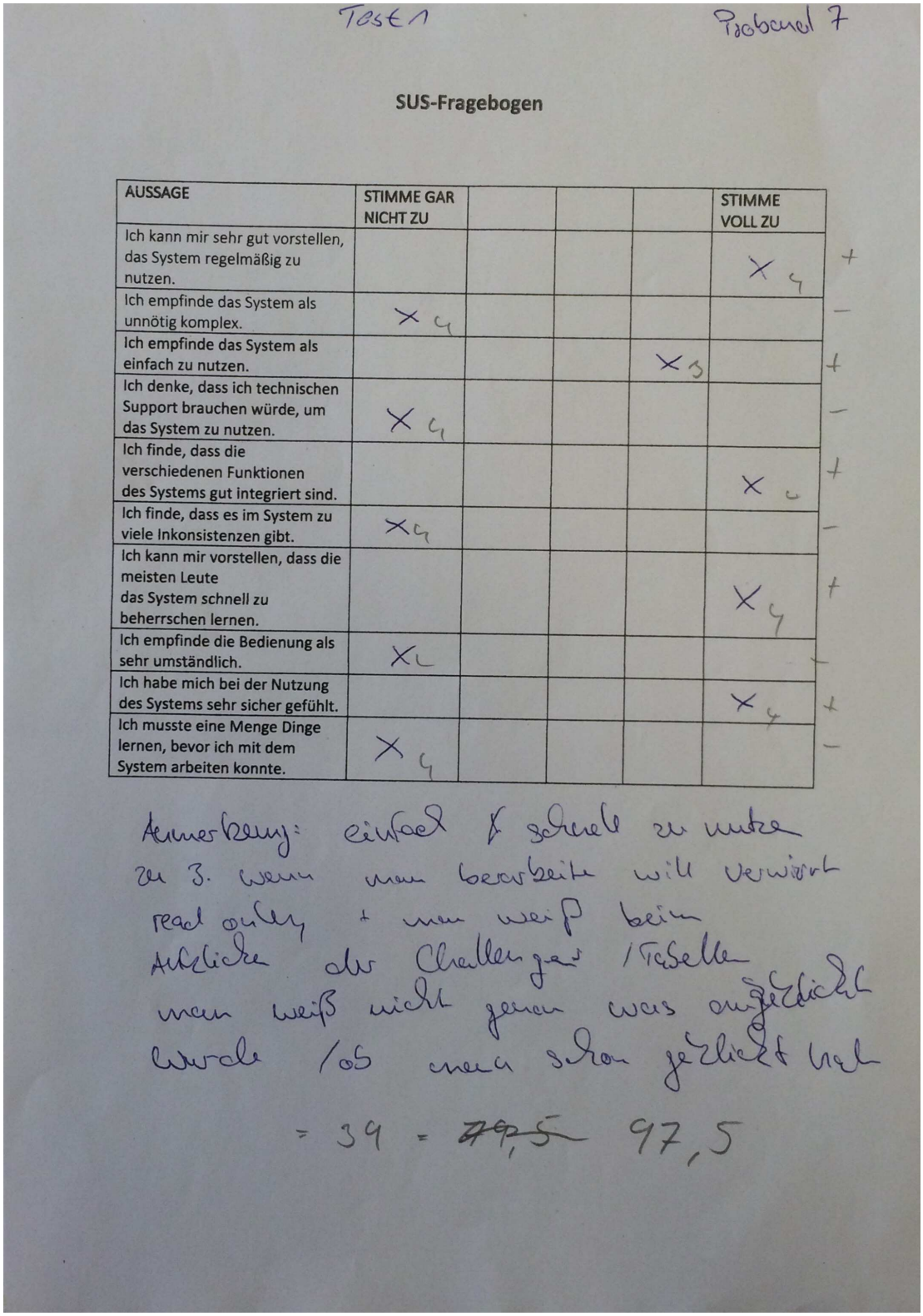
Proband 6

SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU	
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.				X 3		+
Ich empfinde das System als unnötig komplex.	X 4					−
Ich empfinde das System als einfach zu nutzen.				X 3		+
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	X 4					−
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.					X 4	+
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	X 4					−
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.				X 3		+
Ich empfinde die Bedienung als sehr umständlich.	X 4					−
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.				X 3		+
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	X 4					−

36 · 2,5 = 90

Abbildung 122: Auswertung des SUS-Fragebogen von Proband 6



Test 1 Probant 7

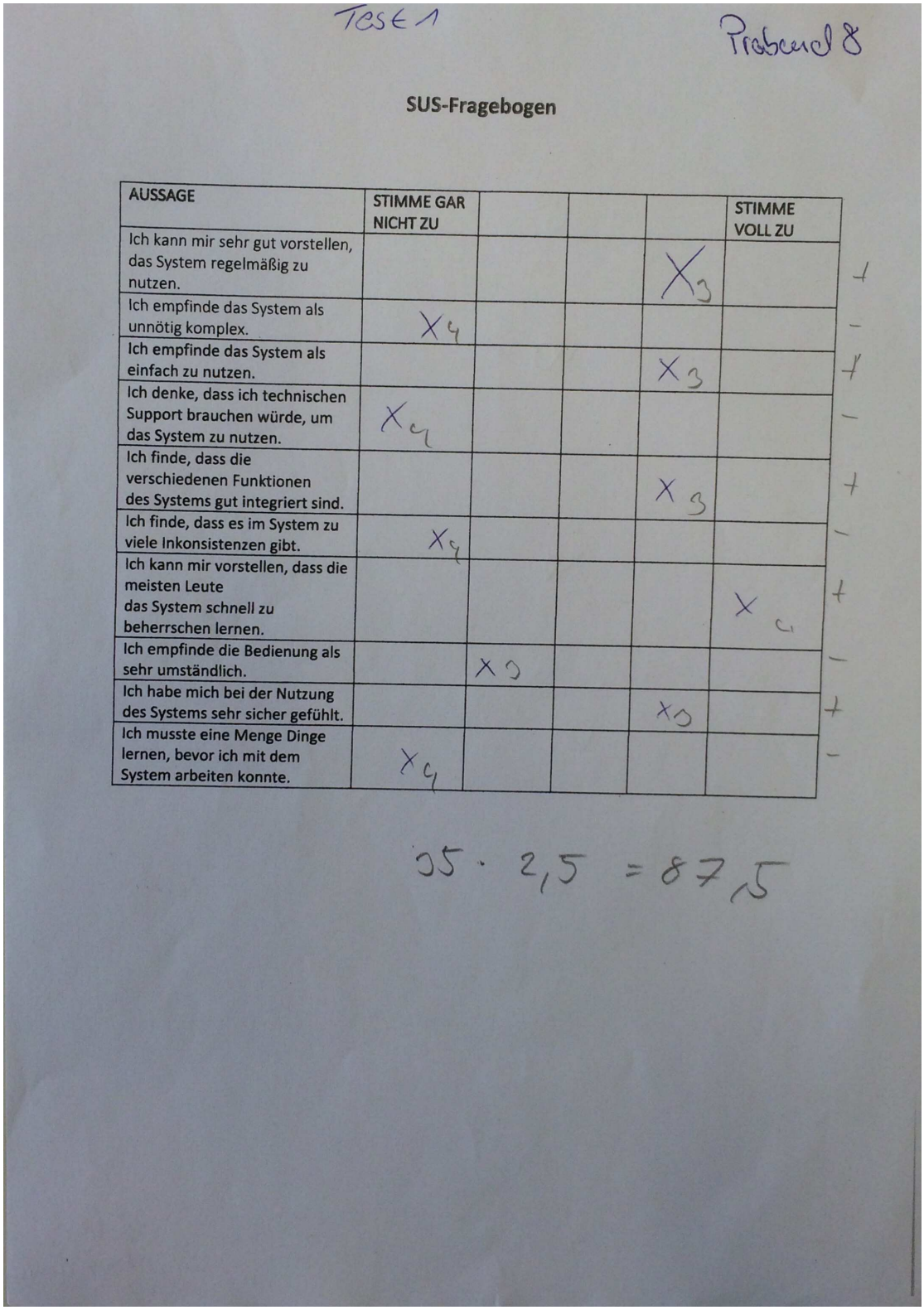
SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU	
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.					X 4	+
Ich empfinde das System als unnötig komplex.	X 4					−
Ich empfinde das System als einfach zu nutzen.				X 3		+
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	X 4					−
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.					X 4	+
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	X 4					−
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.					X 4	+
Ich empfinde die Bedienung als sehr umständlich.	X 2					−
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.					X 4	+
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	X 4					−

Anmerkung: einfach & schnell zu nutzen
zu 3. wenn man bearbeiten will verwirrt
read only + man weiß beim
Klicken der Challenge / Teilen
man weiß nicht genau was angeklickt
wurde los man schon geklickt hat

= 39 = ~~79,5~~ 97,5

Abbildung 123: Auswertung des SUS-Fragebogen von Proband 7



Test 1

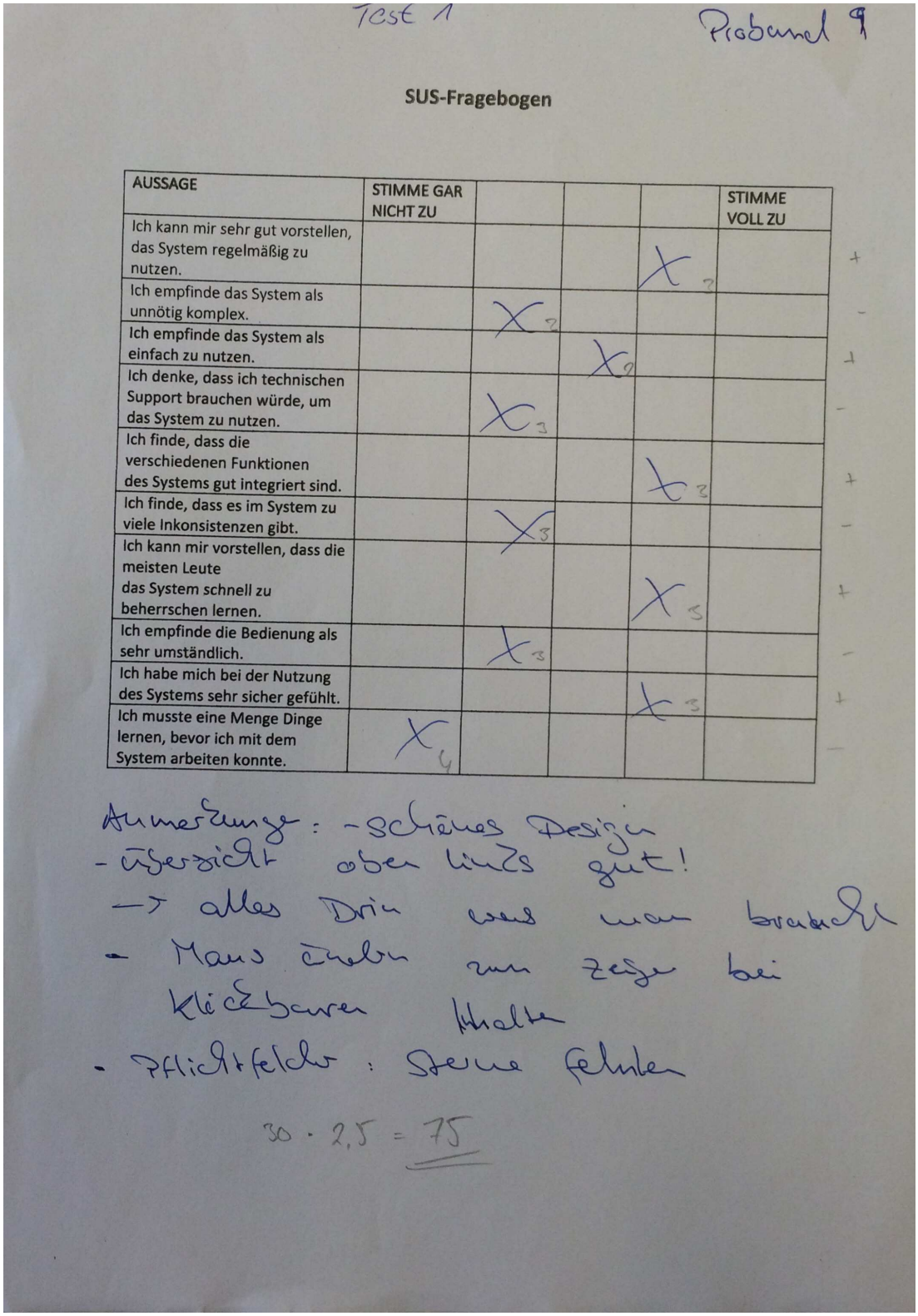
Probend 8

SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU	
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.				X 3		+
Ich empfinde das System als unnötig komplex.	X 4					−
Ich empfinde das System als einfach zu nutzen.				X 3		+
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	X 4					−
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.				X 3		+
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	X 4					−
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.					X 5	+
Ich empfinde die Bedienung als sehr umständlich.		X 2				−
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.				X 4		+
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	X 4					−

35 · 2,5 = 87,5

Abbildung 124: Auswertung des SUS-Fragebogen von Proband 8



Test 1 Proband 9

SUS-Fragebogen

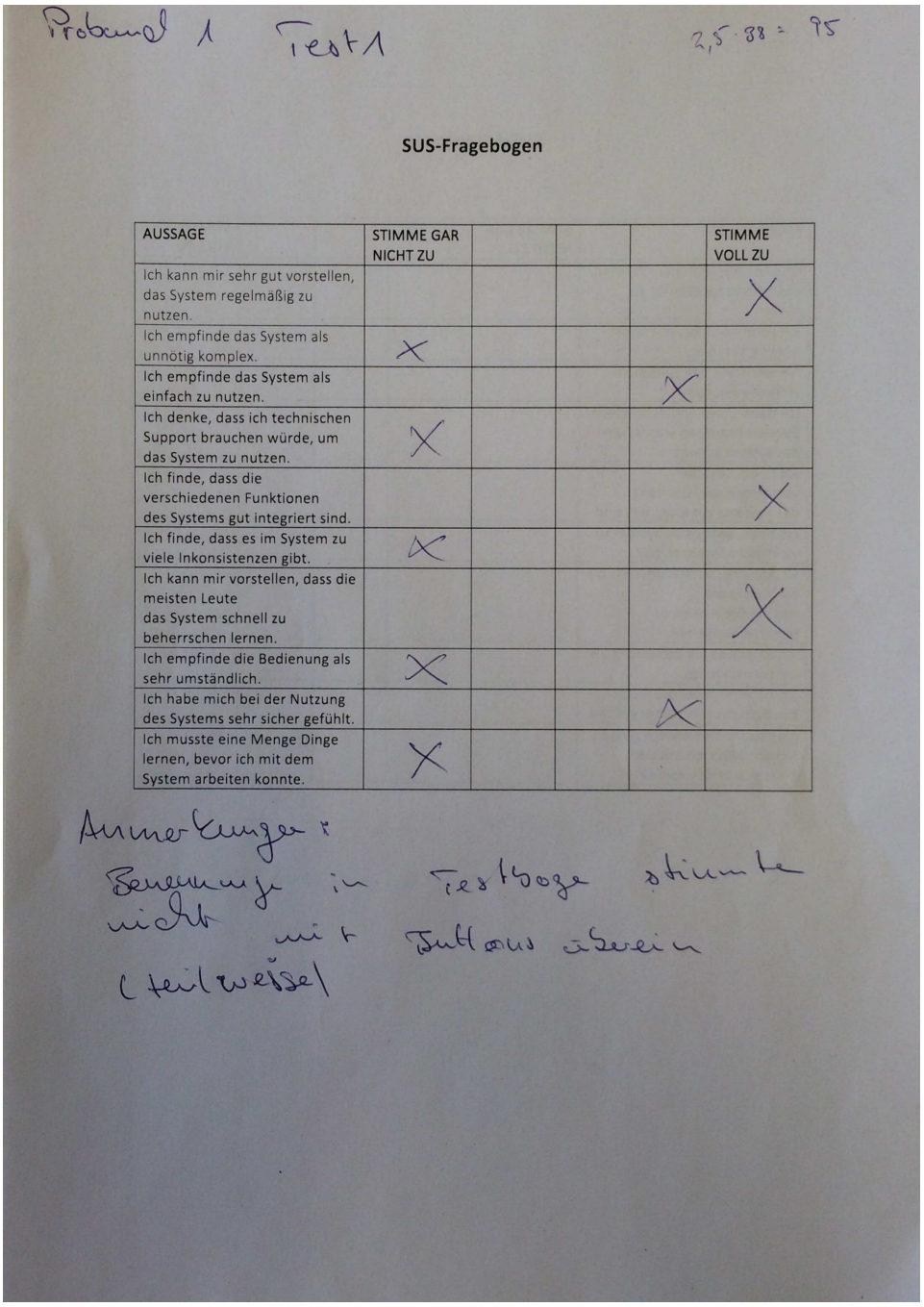
AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU	
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.				X 2		+
Ich empfinde das System als unnötig komplex.		X 2				-
Ich empfinde das System als einfach zu nutzen.			X 2			+
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.		X 3				-
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.				X 3		+
Ich finde, dass es im System zu viele Inkonsistenzen gibt.		X 3				-
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.				X 3		+
Ich empfinde die Bedienung als sehr umständlich.		X 3				-
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.				X 3		+
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	X 4					-

Anmerkungen: - schönes Design
- Übersicht oben links gut!
→ alles Drin was man braucht
- Maus öfter zum Ziehen bei Klickbaren Inhalten
- Pflichtfelder: Sterne fehlen

30 · 2,5 = 75

Abbildung 125: Auswertung des SUS-Fragebogen von Proband 9

A.6.3. Auswertung der SUS-Fragebögen des zweiten Testdurchgangs



Proband 1 Test 1 2,5 · 38 = 95

SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.					X
Ich empfinde das System als unnötig komplex.	X				
Ich empfinde das System als einfach zu nutzen.				X	
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	X				
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.					X
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	X				
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.					X
Ich empfinde die Bedienung als sehr umständlich.	X				
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.				X	
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	X				

Anmerkungen:
Bewertung im Testbogen stimmte nicht mit Tutorin überein (teilweise)

Abbildung 126: Auswertung des SUS-Fragebogen von Proband 1

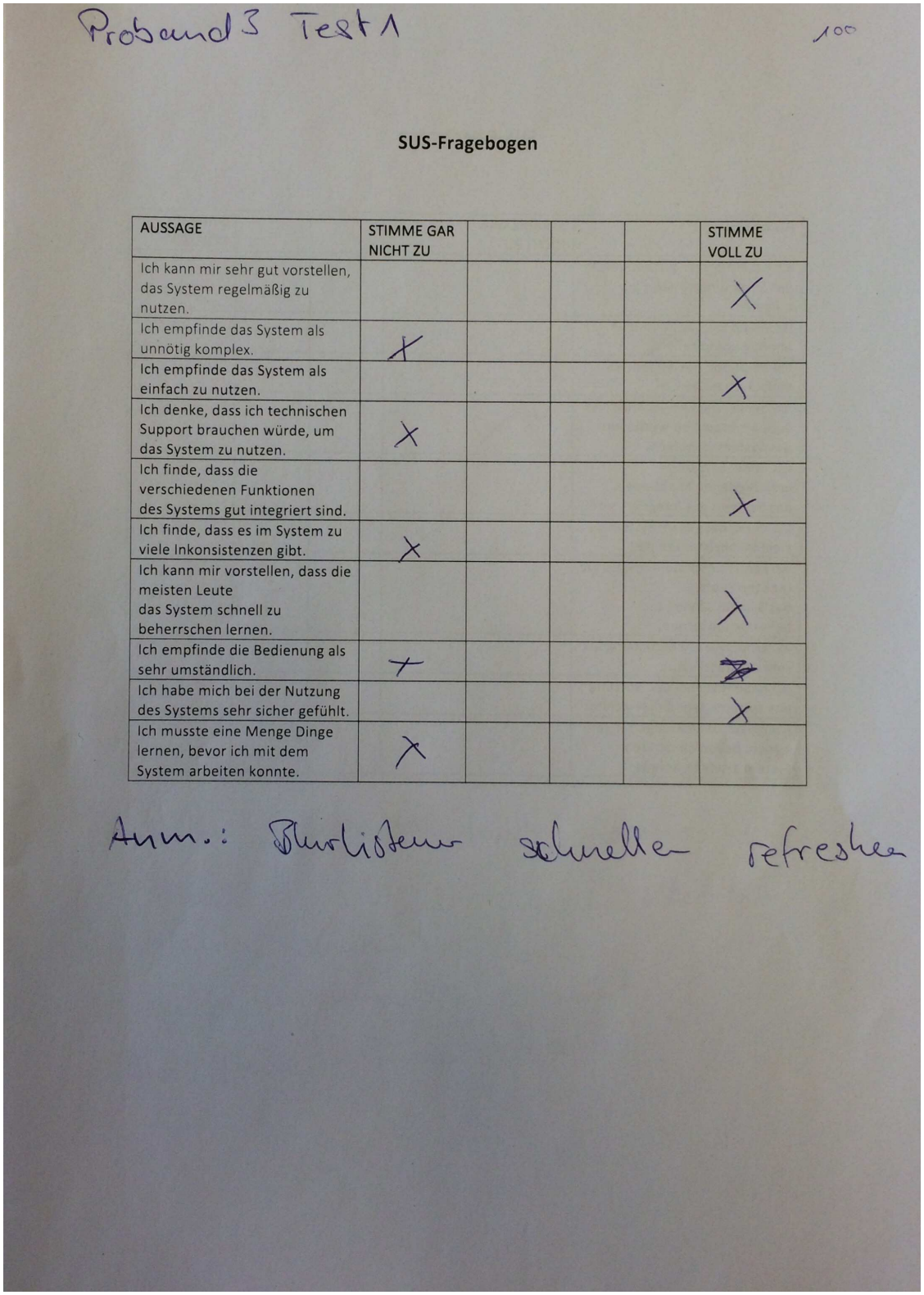
Proband 2 Test 1 2,5 - 39 - 97,5

SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.				X	
Ich empfinde das System als unnötig komplex.	X				
Ich empfinde das System als einfach zu nutzen.					X
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	X				
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.					X
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	X				
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.					X
Ich empfinde die Bedienung als sehr umständlich.	X				
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.					X
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	X				

Anm.: Langzeit motivation u. U. nicht stets ausgeprägt

Abbildung 127: Auswertung des SUS-Fragebogen von Proband 2



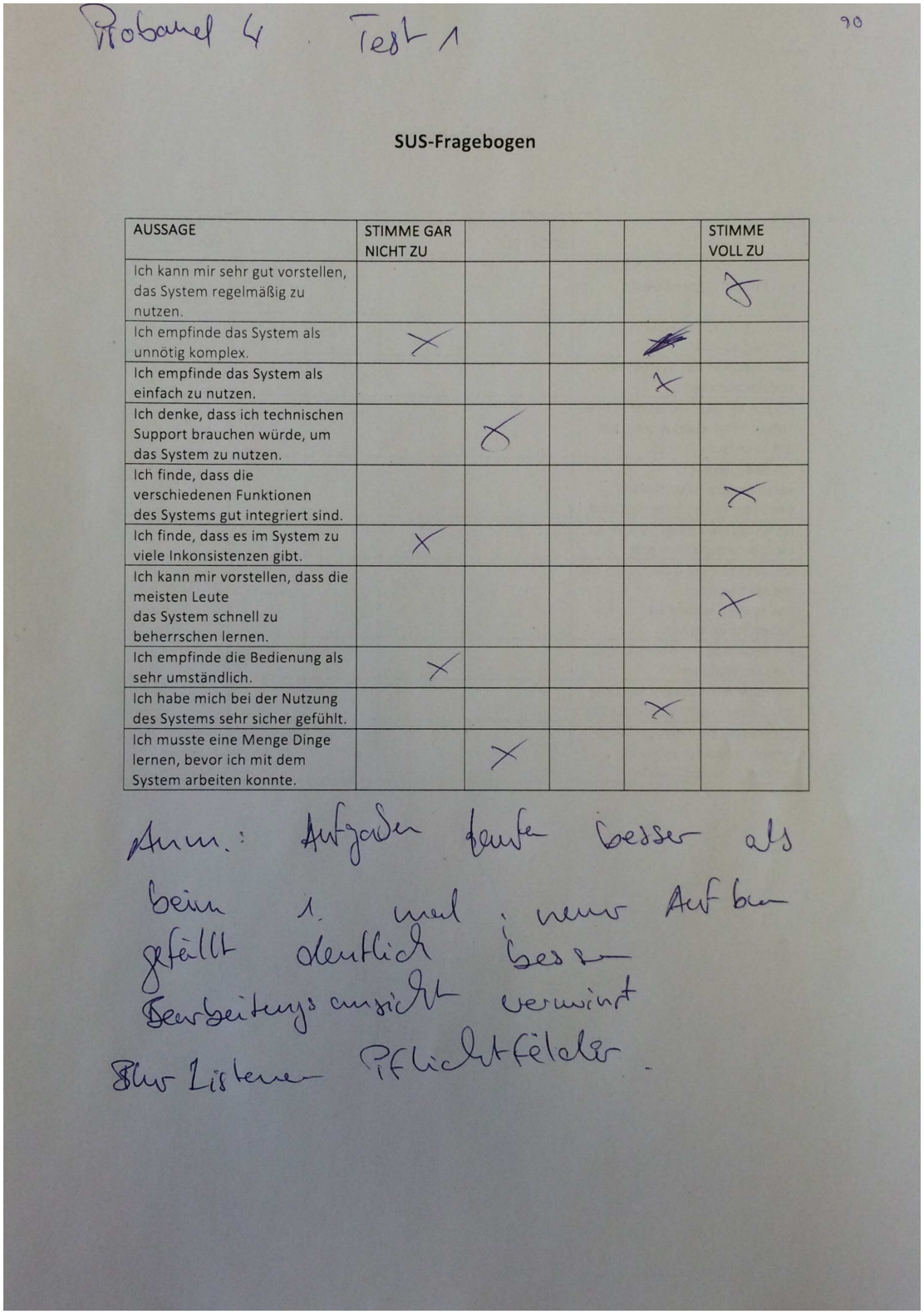
Proband 3 Test 1 100

SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.					X
Ich empfinde das System als unnötig komplex.	X				
Ich empfinde das System als einfach zu nutzen.					X
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	X				
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.					X
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	X				
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.					X
Ich empfinde die Bedienung als sehr umständlich.	X				X
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.					X
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	X				

Anm.: Blurlisten schneller refreshen

Abbildung 128: Auswertung des SUS-Fragebogen von Proband 3



Proband 4 Test 1

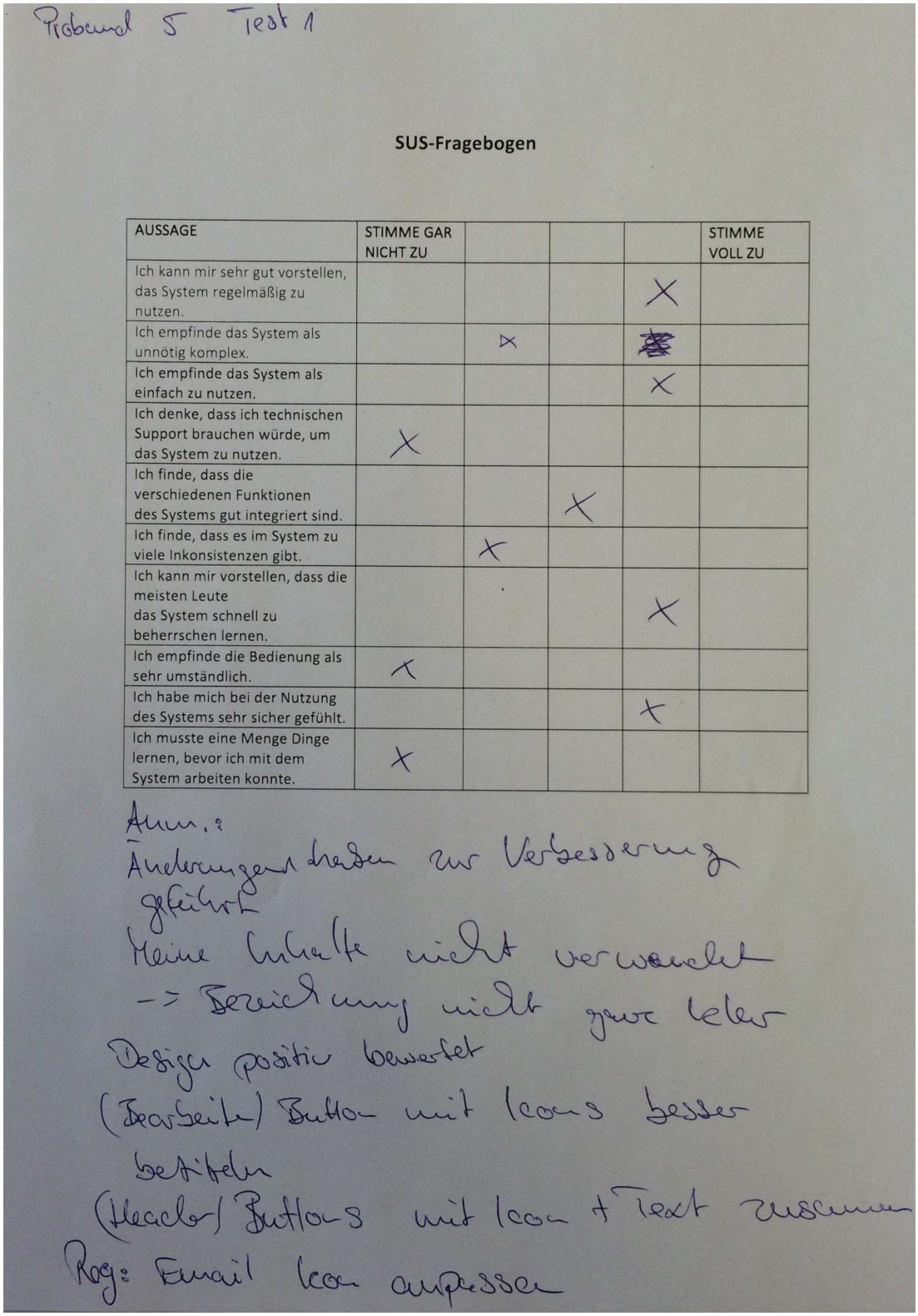
90

SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.					X
Ich empfinde das System als unnötig komplex.	X			~~X~~	
Ich empfinde das System als einfach zu nutzen.				X	
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.		X			
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.					X
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	X				
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.					X
Ich empfinde die Bedienung als sehr umständlich.	X				
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.				X	
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.		X			

Anm.: Aufgaben laufen besser als beim 1. mal. neue Aufbau gefällt deutlich besser Bearbeitungsansicht verwirrt Bsp Listenen Pflichtfelder.

Abbildung 129: Auswertung des SUS-Fragebogen von Proband 4



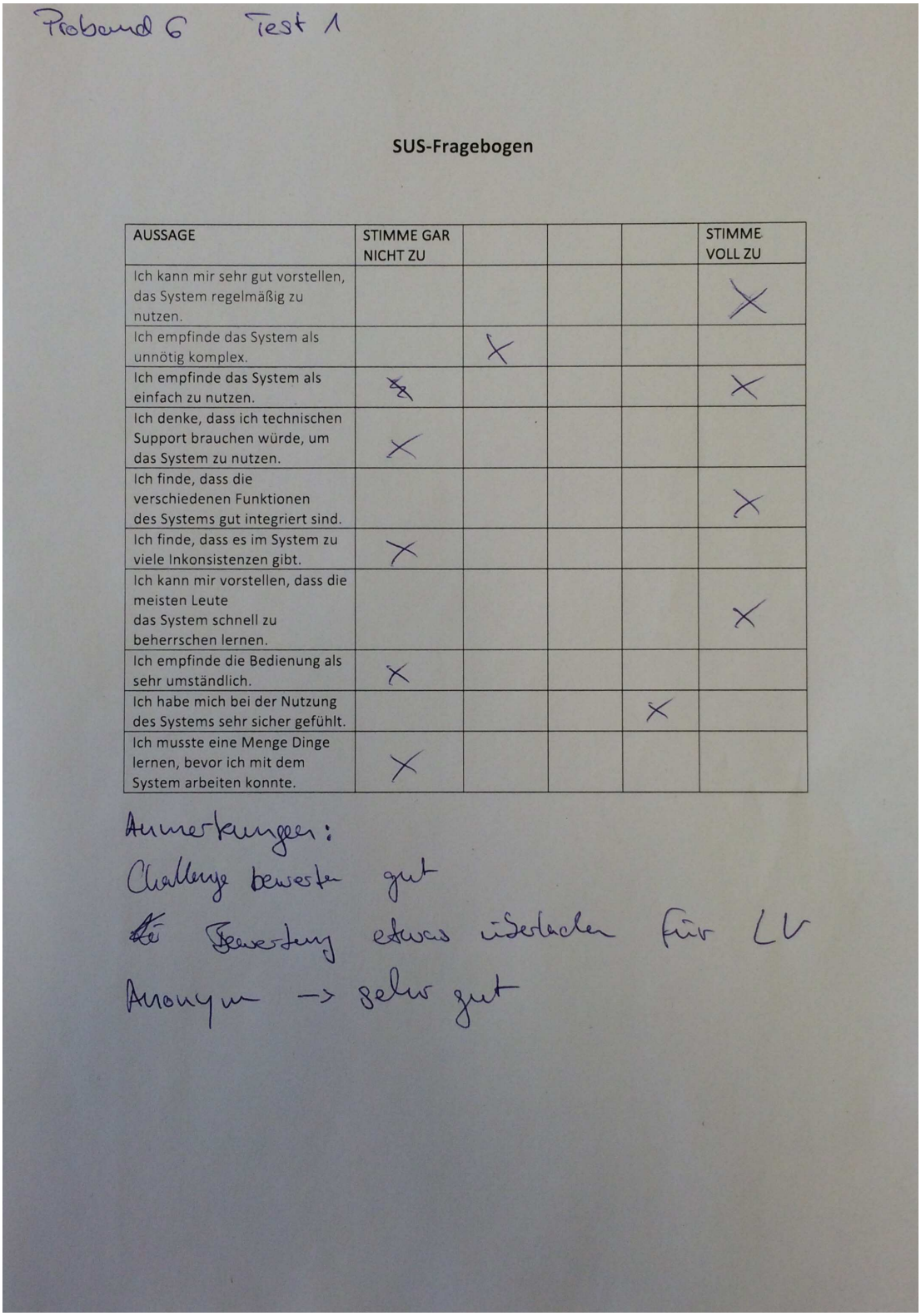
Proband 5 Test 1

SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.				X	
Ich empfinde das System als unnötig komplex.		X		X	
Ich empfinde das System als einfach zu nutzen.				X	
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	X				
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.			X		
Ich finde, dass es im System zu viele Inkonsistenzen gibt.		X			
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.				X	
Ich empfinde die Bedienung als sehr umständlich.	X				
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.				X	
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	X				

Anm.:
Änderungen haben zur Verbesserung geführt
Meine Inhalte nicht verwendet
-> Bezeichnung nicht gut gewählt
Design positiv bewertet
(Tagesseite) Buttons mit Icons besser betiteln
(Header) Buttons mit Icon + Text zusammen
Reg: Email Icon anpassen

Abbildung 130: Auswertung des SUS-Fragebogen von Proband 5



Proband 6 Test 1

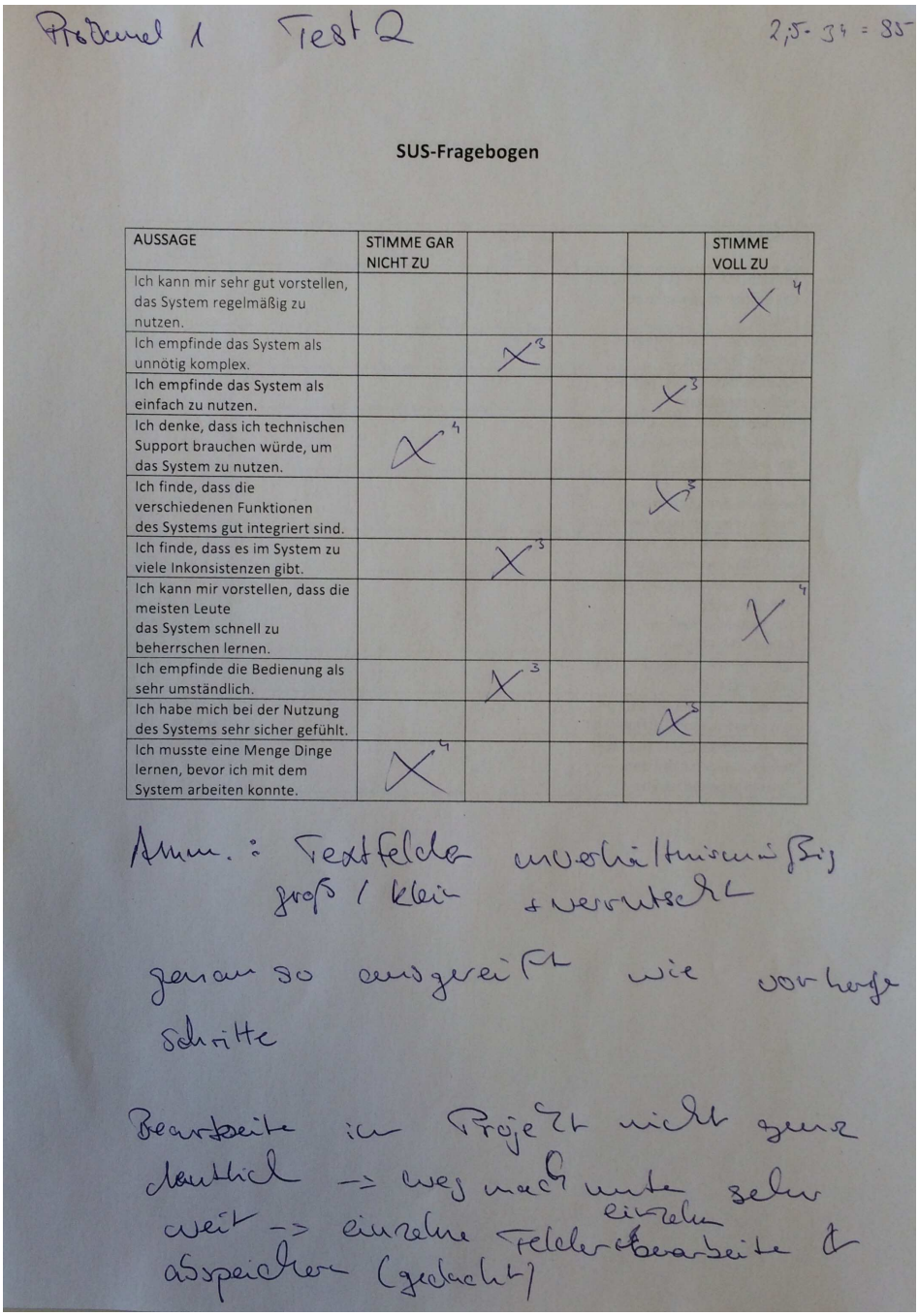
SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.					X
Ich empfinde das System als unnötig komplex.		X			
Ich empfinde das System als einfach zu nutzen.	X				X
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	X				
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.					X
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	X				
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.					X
Ich empfinde die Bedienung als sehr umständlich.	X				
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.				X	
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	X				

Anmerkungen:
Challenge bewerten gut
Bewertung etwas überdenken für LV
Anonym -> sehr gut

Abbildung 131: Auswertung des SUS-Fragebogen von Proband 6

A.6.4. Auswertung der SUS-Fragebögen des dritten Testdurchgangs



Proband 1 Test 2 2,5 · 34 = 85

SUS-Fragebogen

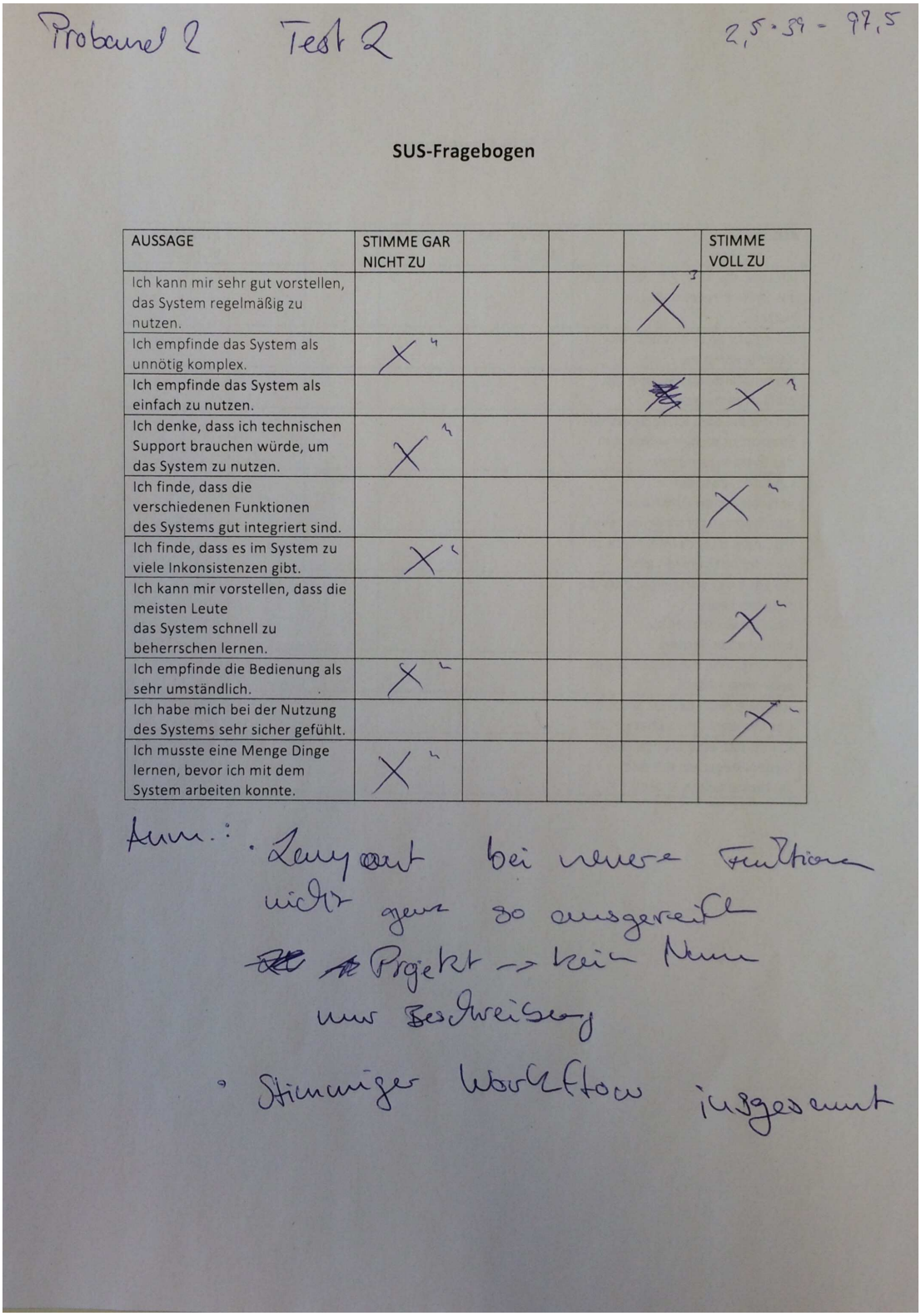
AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.					X 4
Ich empfinde das System als unnötig komplex.		X 3			
Ich empfinde das System als einfach zu nutzen.				X 3	
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	X 4				
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.				X 3	
Ich finde, dass es im System zu viele Inkonsistenzen gibt.		X 3			
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.					X 4
Ich empfinde die Bedienung als sehr umständlich.		X 3			
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.				X 3	
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	X 4				

Anm.: Textfelder unverhältnismäßig groß / klein + verrutscht

genau so ausgereift wie vorherige Schritte

Bearbeite im Projekt nicht ganz deutlich -> weg nach unten sehr weit -> einzelne Felder einzeln bearbeite & abspeichern (gedacht)

Abbildung 132: Auswertung des SUS-Fragebogen von Proband 1



Proband 2 Test 2 2,5 · 39 = 97,5

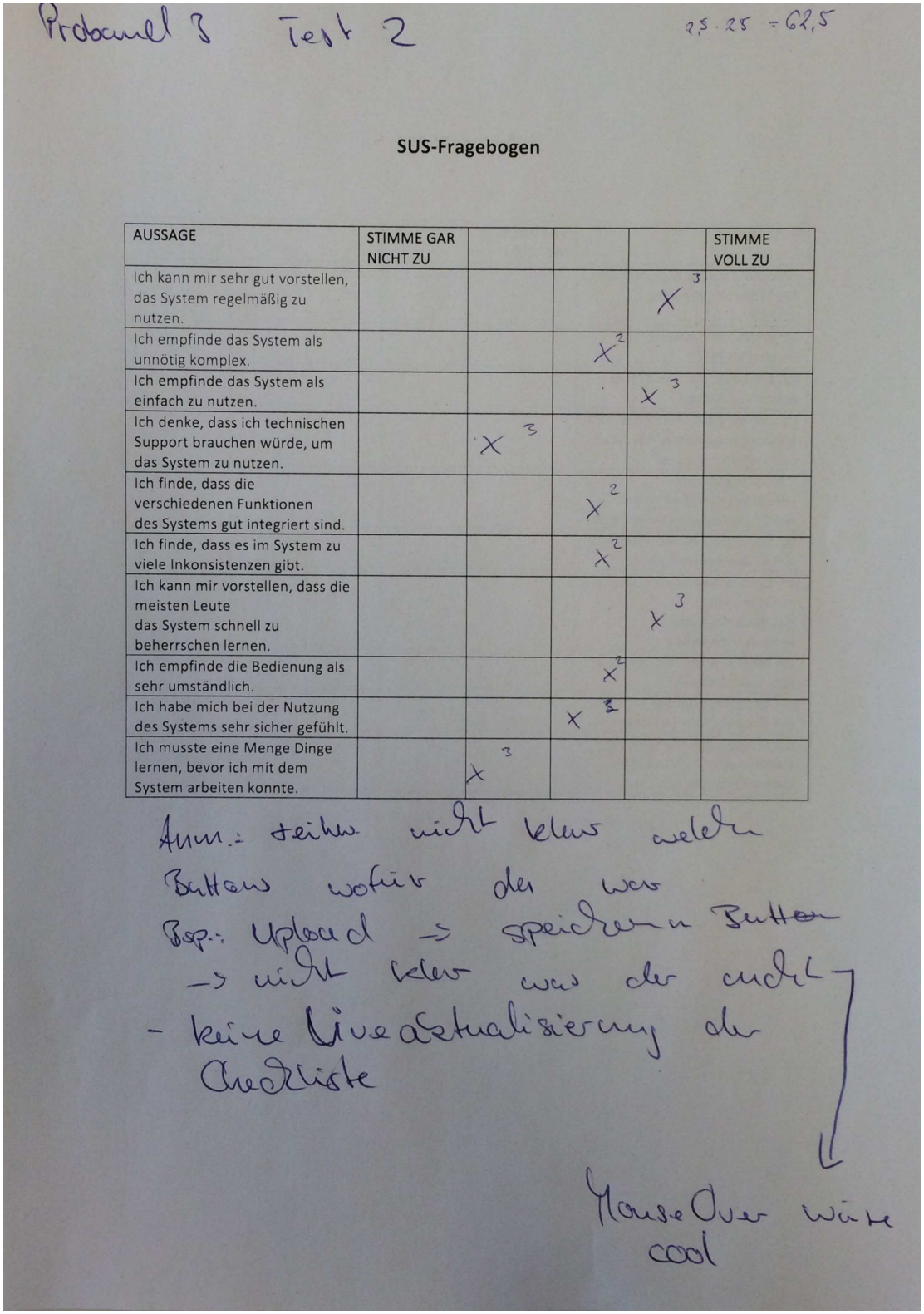
SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.				X 3	
Ich empfinde das System als unnötig komplex.	X 4				
Ich empfinde das System als einfach zu nutzen.				~~X~~	X 1
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	X 4				
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.					X 4
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	X 4				
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.					X 4
Ich empfinde die Bedienung als sehr umständlich.	X 4				
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.					X 4
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	X 4				

Anm.: · Lernkurve bei neuen Funktionen nicht ganz so ausgeprägt ~~Projekt~~ Projekt → kein Nutzen und Beschreibung

· Stimmiger Workflow insgesamt

Abbildung 133: Auswertung des SUS-Fragebogen von Proband 2



Proband 3 Test 2

22,5 · 25 = 62,5

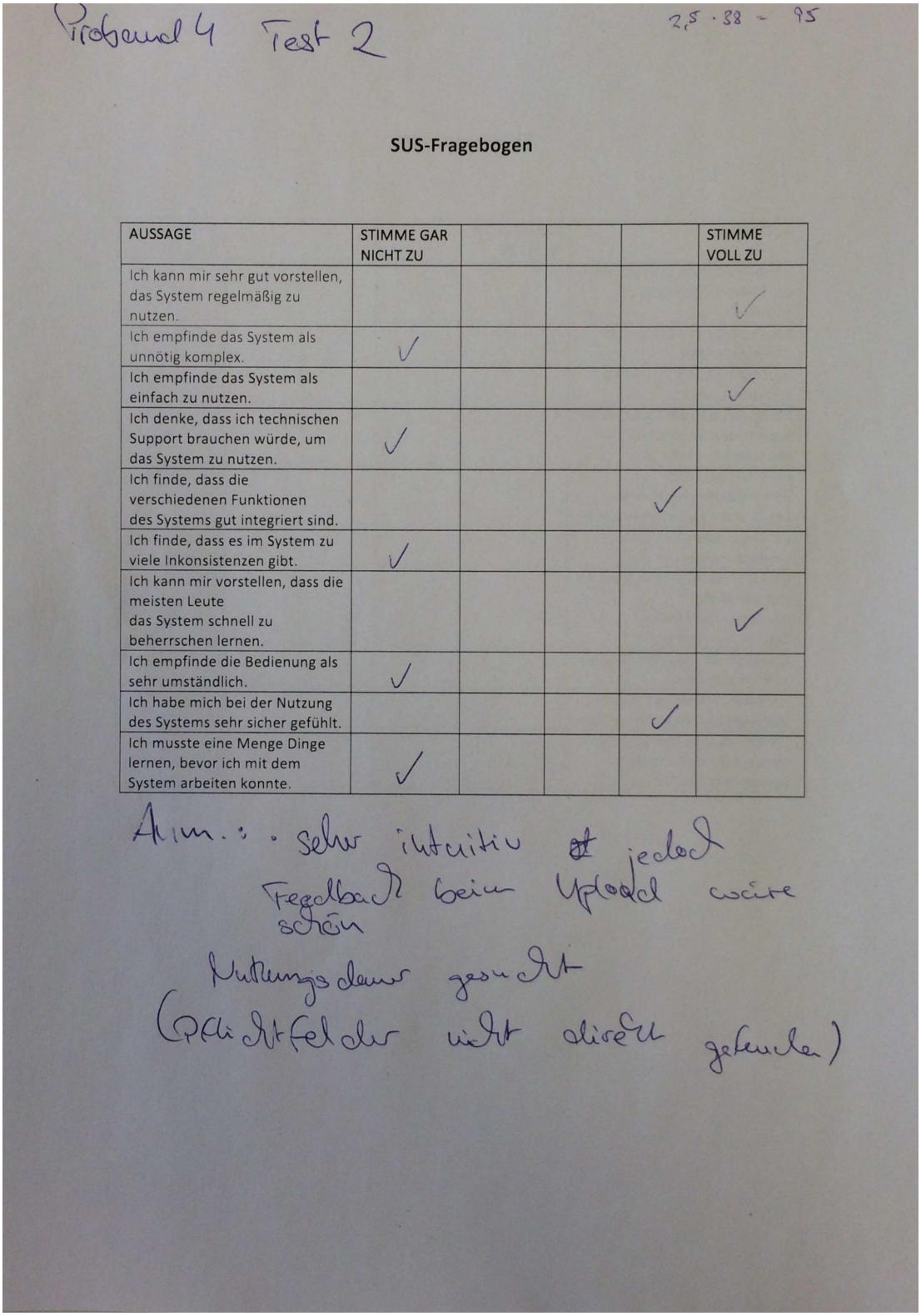
SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.				X 3	
Ich empfinde das System als unnötig komplex.			X 2		
Ich empfinde das System als einfach zu nutzen.				X 3	
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.		X 3			
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.			X 2		
Ich finde, dass es im System zu viele Inkonsistenzen gibt.			X 2		
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.				X 3	
Ich empfinde die Bedienung als sehr umständlich.			X 2		
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.			X 3		
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.		X 3			

Anm.: teils nicht klar welche Buttons wofür da war
Bsp.: Upload → speichern Button
→ nicht klar was da macht
- keine Live-Aktualisierung der Checkliste

Mouse Over wäre cool

Abbildung 134: Auswertung des SUS-Fragebogen von Proband 3



Proband 4 Test 2

2,5 · 38 = 95

SUS-Fragebogen

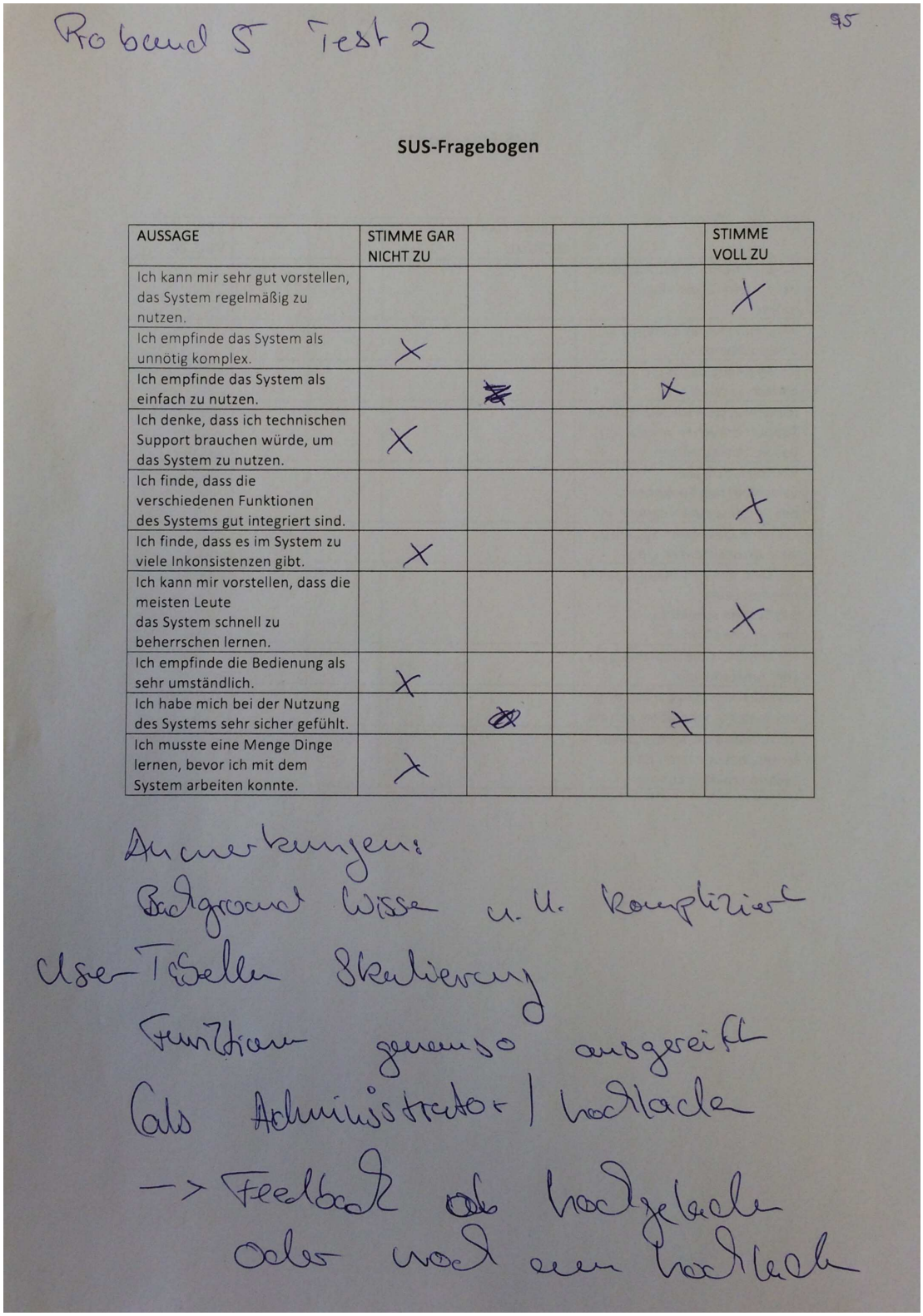
AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.					✓
Ich empfinde das System als unnötig komplex.	✓				
Ich empfinde das System als einfach zu nutzen.					✓
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	✓				
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.				✓	
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	✓				
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.					✓
Ich empfinde die Bedienung als sehr umständlich.	✓				
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.				✓	
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	✓				

Anm.: · sehr intuitiv ~~ist~~ jedoch Feedback beim Upload wäre schön

Nutzungsdauer gesucht

(Pflichtfelder nicht direkt gekennzeichnet)

Abbildung 135: Auswertung des SUS-Fragebogen von Proband 4



Proband 5 Test 2

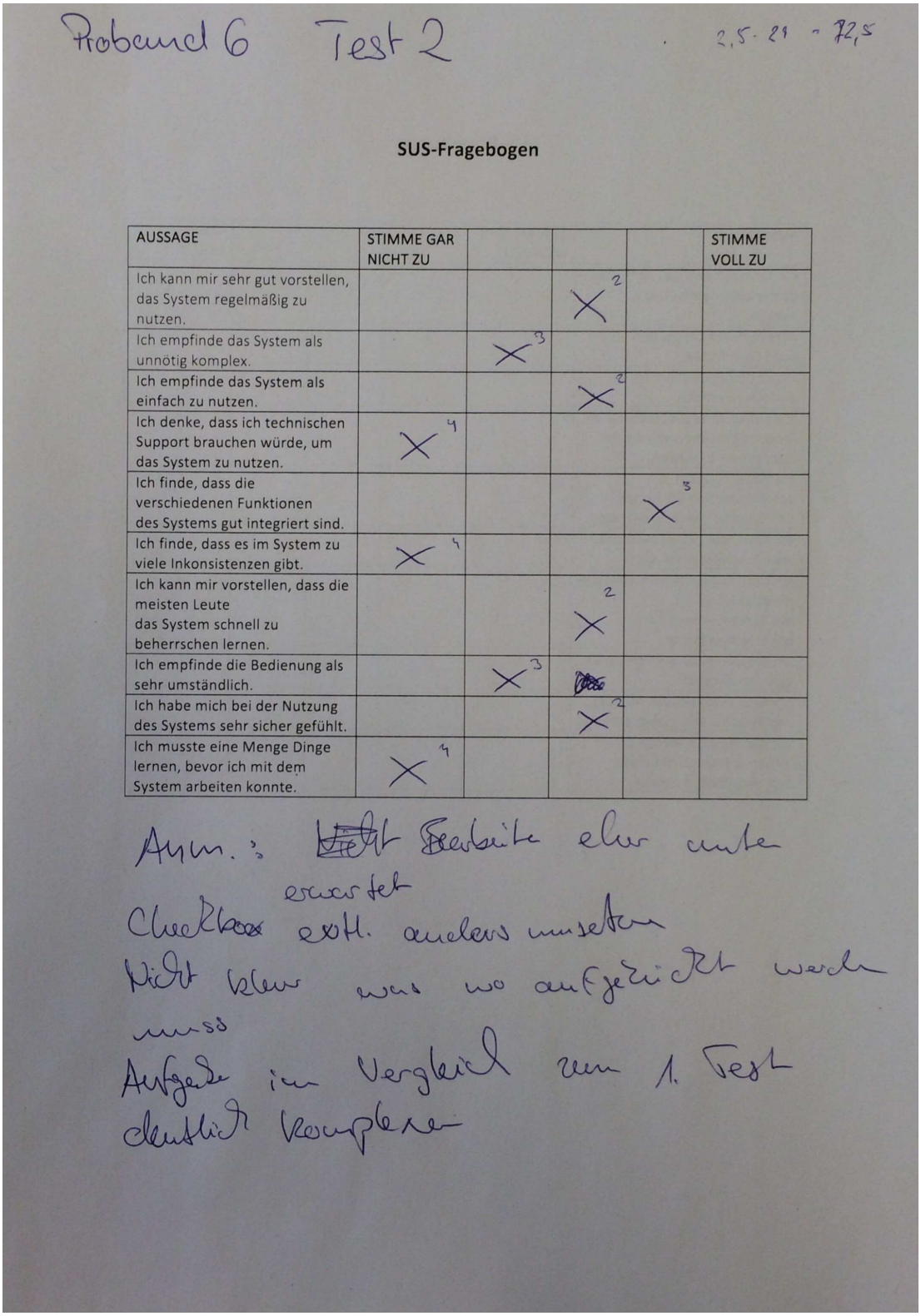
95

SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.					X
Ich empfinde das System als unnötig komplex.	X				
Ich empfinde das System als einfach zu nutzen.		X		X	
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	X				
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.					X
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	X				
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.					X
Ich empfinde die Bedienung als sehr umständlich.	X				
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.		X		X	
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	X				

Anmerkungen:
Background Wissen u.U. Kompliziert
User-Tabelle Skalierung
Funktionen genauso ausgereift
Als Administrator / Hacker
-> Feedback als Hacker
oder noch einen Hacker

Abbildung 136: Auswertung des SUS-Fragebogen von Proband 5



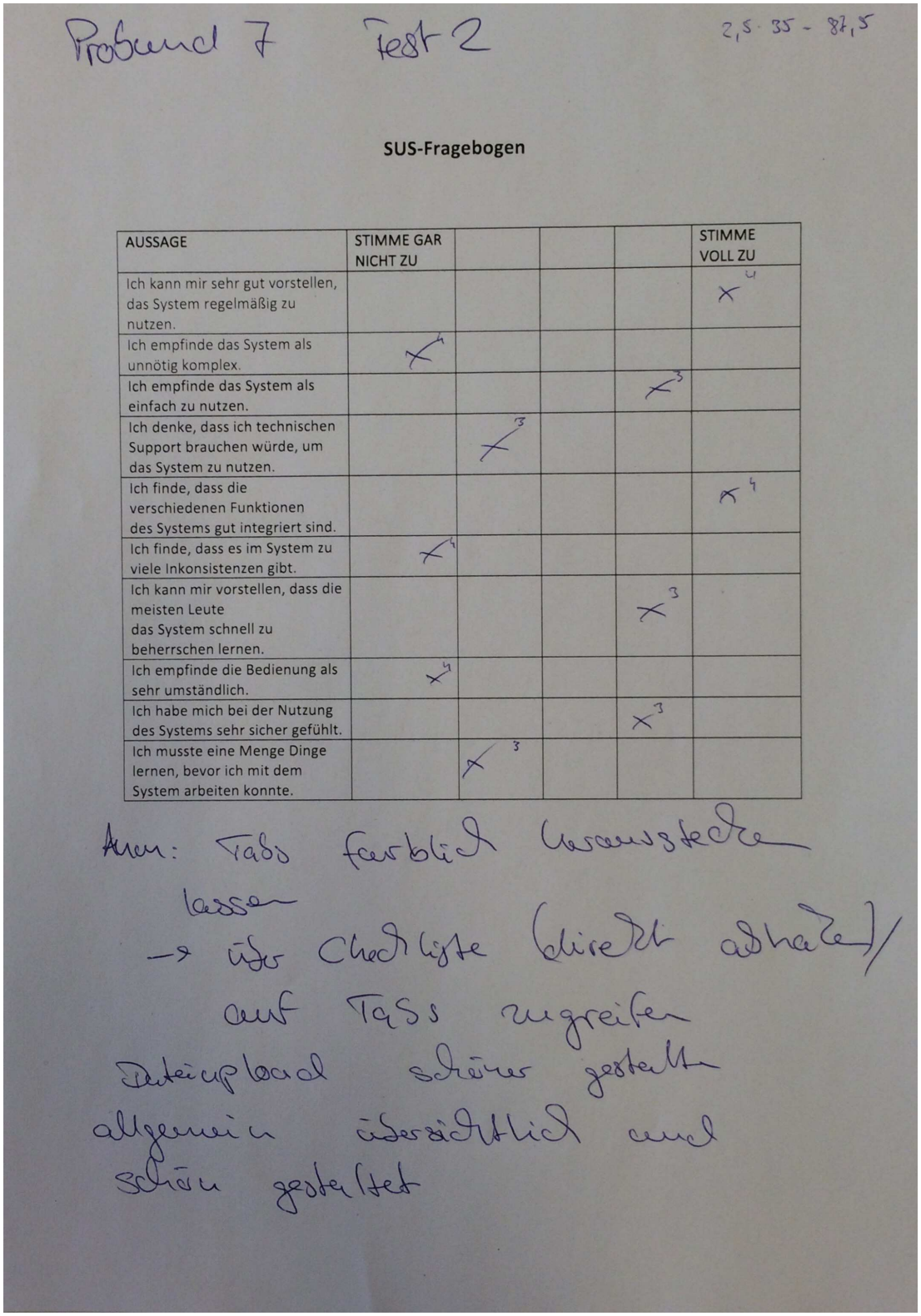
Proband 6 Test 2 2.5.21 - 72,5

SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.			X 2		
Ich empfinde das System als unnötig komplex.		X 3			
Ich empfinde das System als einfach zu nutzen.			X 2		
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	X 4				
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.				X 3	
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	X 4				
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.			X 2		
Ich empfinde die Bedienung als sehr umständlich.		X 3	[illegible]		
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.			X 2		
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	X 4				

Anm.: ~~Nicht~~ Bearbeite eher unter erwartet
Checkbox evtl. anders umsetzen
Nicht klar wann wo aufgerückt werden muss
Aufgabe im Vergleich zum 1. Test deutlich komplexer

Abbildung 137: Auswertung des SUS-Fragebogen von Proband 6



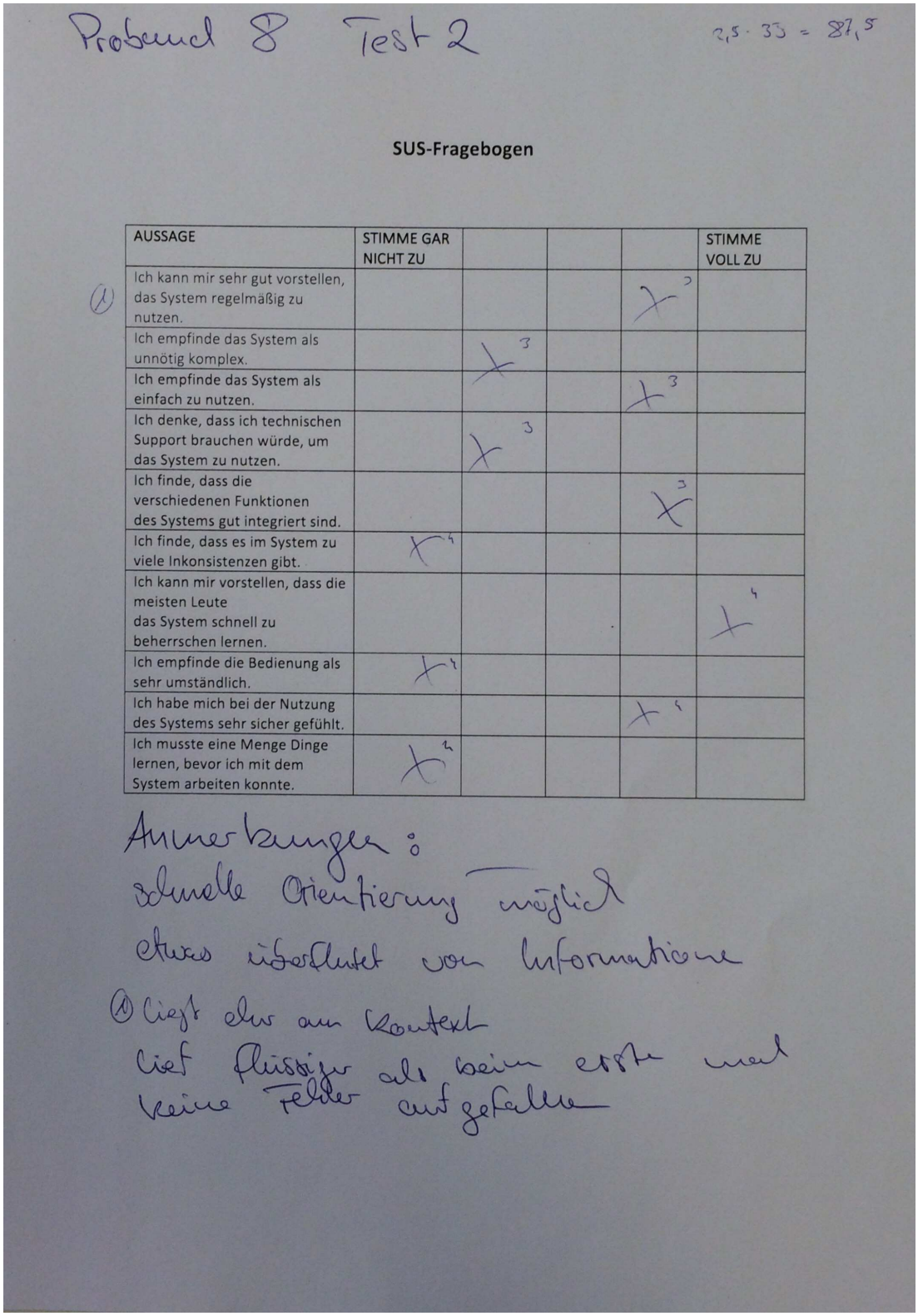
Proband 7 Test 2 2,5 · 35 = 87,5

SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.					X 4
Ich empfinde das System als unnötig komplex.	X 4				
Ich empfinde das System als einfach zu nutzen.				X 3	
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.		X 3			
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.					X 4
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	X 4				
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.				X 3	
Ich empfinde die Bedienung als sehr umständlich.	X 4				
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.				X 3	
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.		X 3			

Anm: Tabs farblich voneinander lassen
→ über Checkliste (direkt abhaken) auf Tabs zugreifen
Dateiupload schöner gestalten
allgemein übersichtlich und schön gestaltet

Abbildung 138: Auswertung des SUS-Fragebogen von Proband 7



Proband 8 Test 2

2,5 · 35 = 87,5

SUS-Fragebogen

AUSSAGE	STIMME GAR NICHT ZU				STIMME VOLL ZU
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.				X 3	
Ich empfinde das System als unnötig komplex.		X 3			
Ich empfinde das System als einfach zu nutzen.				X 3	
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.		X 3			
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.				X 3	
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	X 4				
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.					X 4
Ich empfinde die Bedienung als sehr umständlich.	X 4				
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.				X 4	
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	X 4				

Anmerkungen:
schnelle Orientierung möglich
etwas überfordert von Informationen
① liegt eher am Kontext
lief flüssiger als beim ersten mal
keine Fehler aufgefallen

Abbildung 139: Auswertung des SUS-Fragebogen von Proband 8

A.7. Protokoll

Im Rahmen der wöchentlichen Projektgruppensitzungen wurde jeweils ein Protokoll angelegt. Die Rolle des Protokollanten wurde dabei für jeden Termin getauscht. Dies geschah in einer anhand einer festgelegten Reihenfolge unter Verwendung einer Liste.
Die nachfolgenden Protokolle stellen somit eine Zusammenfassung der jeweiligen Gruppensitzung dar.

A.7.1. Protokoll vom 09.04.2015

Protokoll_150409

Datum: 09.04.15

Moderation: Artjom Baranow

Protokollant: Sebastian Beckmann

Abwesend: Yaping Lian (entschuldigt)

Top 1: Abstimmungen und organisatorische Themen

- Gruppentreffen: Donnerstag ab 12:30 Uhr bis ca. 16:00 Uhr
- Fotos: Falls alle da sind, werden beim nächsten Gruppentreffen die Einzel- und Gruppenfotos gemacht.
- Server steht ab nächster Woche zur Verfügung. Falls kostenpflichtige Erweiterungen benötigt werden, muss dies bei der Abteilung angefragt werden.
- Name der Projektgruppe: IMPACT – Innovation Management Platform To Activate Creative Thoughts
- Die Webseite wird mittels Wordpress umgesetzt.
- Folgende Standards wurden festgelegt:
    - Versionsverwaltungssoftware: SVN
    - E-Mail Verteiler: Es wird einen internen und externen E-mail Verteiler geben (voraussichtlich pg-impact-intern@.... und pg-impact-extern). Der externe Verteiler ist zusätzlich zu den Studierenden an die Modulverantwortlichen adressiert.
    - Projektmanagementsoftware: Confluence und Jira
    - Dokumentationssoftware: Latex
- Strafen:
    - Beim Zuspätkommen wird eine Pauschale von 1€ fällig, diese erhöht sich alle 5 Minuten um 1€ bis maximal 6€ pro Treffen.
    - Alle Strafen sind beim Kassenwart (Dirk Tesche) zu entrichten.
    - Am Vortrag des Treffens kann man sich bis 18:00 Uhr abmelden (dies gilt nicht für kurzfristige Krankheitsfälle).
- Monatlich erhält der Kassenwart ein Mitgliedsbeitrag von 5€ von jedem Gruppenmitglied.
- Das Protokoll einer Gruppensitzung muss bis zum nächsten Montag 24:00 Uhr im Confluence zur Verfügung stehen.
- Die Tagesordnung für das nächste Gruppentreffen muss einen Tag vor diesem, also am Mittwoch bis 18:00 Uhr, im Confluence zur Verfügung stehen.

- Urlaube müssen ins Confluence eingetragen werden

Top 2: Zuordnung zu den Seminarthemen

Name	Thema
Vanessa	SE-Vorgehensmodell
Dirk	Testmanagement
Artjom	Oberflächendesign
Jan	Programmierkonventionen
Sebastian	Aufbau eines Business Case
Jessica	Entwicklungsstufen
Patrick	Gamification
Holger	Anreizsysteme/Motivation
Tobi	Kommunikation und Konfliktmanagement
Daniel	Erfolgsbeteiligung für Mitarbeiter bei erfolgreichen Projekten
Yaping	Dokumentation

Zusätzlich wird es einen Vortrag von Christoph Schmitt und Kerem Sevlimi geben. Sie stellen ihre Masterarbeit mit dem Thema “Konzeption und prototypische Implementierung einer Enterprise Crowdfunding Plattform“ vor.

Top 3: Rollen-/Aufgabenverteilung

- Projektmanager für die Seminarphase: Patrick Smit
- Serveradministrator: Vanessa Dering
- Webseitenbeauftragter: Artjom Baranow
- Der Moderator für jede Sitzung wird alphabetisch bestimmt.
- Der Moderator muss einen Beamer organisieren.
- Der Protokollant ist jeweils die Person, die nach dem Moderator im Alphabet kommt.
- Der Protokollant ist somit der Moderator der darauffolgenden Sitzung

Top 4: Aufgaben zum nächsten Termin

- Alle Gruppenmitglieder sollen sich mit folgenden Themen vertraut machen:
    - Vorgehensmodelle
    - Projektzeitraum (z.B. wann Gespräche mit den Personal geplant sind)

- Ihr eigenes Seminarthema
- Das Logo der Projektgruppe
- Was Serverseitig gebraucht wird

Einzelaufgaben:

Stefan	Mit der Abteilung abklären, ob Lizenzen für ein UML-Werktzeug (z.B. Visual Paradigm) zur Verfügung stehen
Patrick	Mit Confluence und Projektplan beschäftigen, um ersten Gruppenabend kümmern
Sebastian	E-Mail Verteiler einrichten
Vanessa	Serverarchitektur einrichten
Tobias	Latex-Vorlage erstellen, SVN einrichten
Artjom	Gedanken zur Webseite machen

Top 5: Anmerkungen / Sonstiges

- Erster Gruppenabend: Wahrscheinlich in der Pinte42 mit Grillen
- Aufgaben eines Projektleiters (Brainstorming der Gruppe):
  - Terminverwaltung
  - Arbeitszuweisung
  - Direkter Ansprechpartner für Modulverantwortliche
  - Zusammenhalt der Gruppe stärken
  - Hat das Recht zu delegieren (muss gewisse Autorität besitzen)

A.7.2. Protokoll vom 16.04.2015

Protokoll_150416

Datum: 16.04.2015

Uhrzeit: 12:30 bis 14:00

Moderation: Sebastian Beckmann

Protokollant: Vanessa Dering

Top 1: Weekly Scrum

Name	erledigte Aufgaben
Artjom	• Literatur für das Seminar gesammelt • In Wordpress eingearbeitet und ersten Entwurf der Webseite fertig gestellt • Moderations- und Protokollübersicht • Einarbeitung in Jira+Confluence
Tobias	• LaTex Vorlage erstellt • Einarbeitung in SVN und Jira+Confluence • Seminarthema bearbeitet
Patrick	• Planung des Gruppentreffens • Einarbeitung in Seminarthema • Einarbeitung in Jira+Confluence
Holger	• Einarbeitung in Literatur zum Seminarthema und Jira+Confluence • Strukturierung des Seminarablaufes • Absprache mit Vanessa zur thematischen Abstimmung der Seminarthemen
Daniel	• Einarbeiten in das Seminarthema
Jan	• Literatur für das Seminar beschafft und angelesen • Einarbeiten in Jira+Confluence • Logovorschlag erstellt • Inhalte der Webseite zusammengesucht
Yaping	• Einarbeiten in Jira+Confluence und in das Seminarthema
Jessica	• Überlegung zum Vorgehensmodell • Planen des Projektzeitraumes • Gliederung erstellt und Quellen zusammengesucht für das Seminar • Einarbeiten in Jira+Confluence
Sebastian	• E-Mail Verteiler erstellt • Recherche zu SVN und GIT

	• Einarbeiten in die Literatur zum Seminarthema
Vanessa	• Einarbeiten in das Seminarthema und Jira+Confluence • Organisatorische Überlegungen

Top 2: Abstimmungen und organisatorische Themen

- Nach Diskussion und Abstimmung wird im weiteren Verlauf GIT zur Versionsverwaltung benutzt.
- Die Seminarthemen werden am 21.5 ab 12:30 Uhr und am 22.5 ab 9:00 Uhr vorgestellt.
- Der Tausch der Präsentation ist bis zum 15.05. mit seinem Peer Review Partner zu vollziehen. Die Gruppen werden von Patrick zusammengestellt und bekannt gegeben.
- Die Präsentationen wurden thematisch in folgende Gruppen gebündelt:
    - Gruppe 1: SE- Vorgehensmodell, Testmanagement, Programmierkonventionen, Entwicklungsstufen, Kommunikation und Konfliktmanagment, Dokumentation
    - Gruppe 2: Erfolgsbeteiligung für Mitarbeiter bei erfolgreichen Projekten, Anreizsysteme/Motivation, Gamification, Aufbau eines Business Case, Oberflächendesign
- Das Grundgerüst des Moderationsplans muss vom Moderator hochgeladen werden. Alle Mitglieder können bis Mittwoch 16 Uhr weitere Besprechungspunkte in den Moderationsplan einfügen.
- Das Protokoll muss bis Samstag 18 Uhr hochgeladen sein.
- Der Gruppentermin ohne Betreuer kann nach Bedarf Donnerstags von 14:00 - 16:00 stattfinden.
- Am 14.05 wird das Gruppentreffen ausfallen.
- Am 23.04 wird Dr. Joachim Kurzhöfer das Unternehmen AS Inpro vorstellen und anschließend für Fragen bereit stehen. Die Fragen sollen aufklären wie das Unternehmen organisiert und strukturiert ist.

Top 3: Aufgaben zum nächsten Termin

- Alle Gruppenmitglieder sollen sich mit folgenden Themen vertraut machen:
    - Fragen zum Unternehmen AS Inpro bis Dienstag Morgen erarbeiten.
    - Das Logo der Projektgruppe

Name	Aufgabe
Patrick	Kalender befüllen, Tasks erstellen, Erstellen des Dokumentes für die Sammlung der Fragen zum Unternehmen AS Inpro
Dirk	Überlegung zur Gestaltung des Projekttagebuches
Holger	Erstellen einer Regelliste, Bearbeiten der Formatvorlage für Mitarbeiterinterview, Terminabstimmung mit AS Inpro für Mitarbeiterbefragung

Top 4: Anmerkungen / Sonstiges

- Jens Siewert wird nächste Woche 23.04 nicht am Gruppentreffen teilnehmen.
- Dr. Joachim Kurzhöfer wird am 30.04 nicht am Gruppentreffen teilnehmen.

A.7.3. Protokoll vom 23.04.2015

Protokoll_150423

Datum: 23.04.2015

Uhrzeit: 12:30 bis 14:30

Moderation: Vanessa Dering

Protokollant: Holger Eichholz

Abwesend: Stefan Wunderlich (abgemeldet), Jens Siewert (abgemeldet)

Top 1: Weekly Scrum

Name	erledigte Aufgaben
Artjom	• Gliederung der Präsentation • Übertragung der Website in Wordpress auf den Server • Überlegung einer Ordnerstruktur für GIT
Tobias	• Beschäftigung und Vorbereitung der Vorstellung von GIT
Patrick	• Erstellung des Terminkalenders • Erstellung von Tasks in JIRA • Organisation der Vorträge
Holger	• Erstellen eines Vorschlag für den Regelkatalog • Gedanken zur Durchführung der Mitarbeiterbefragung • Beschäftigung mit dem Seminarthema
Daniel	• Beschäftigung mit der Seminarbeit • Einarbeitung in Confluence/JIRA • Beschäftigung mit dem GIT-Hub
Jan	• Beschäftigung mit GIT • Bearbeitung der Seminarbeit
Yaping	• Einarbeitung in Latex • Gedanken zu einem Deckblatt • Literraturrecherche zur Seminarbeit
Jessica	• Vorlagen für Protokolle erstellt • Auseinandersetzung mit GIT • Fragen für die Präsentation
Sebastian	• Aufsetzen von Systemen
Dirk	• Erstellen und Informationen zum Projekttagebuch
Vanessa	• Erstellung von Protokoll und Moderationsplan

Top 2: Abstimmungen und organisatorische Themen

- Vorstellung des Projekttagebuches
    - Projekttagebuch ist eine einheitliche Quelle für alle Dokumente, in der auch wichtige Erfahrungen eingebracht werden
    - Vorstellung der Exceltabelle durch Dirk als ersten Entwurf (-> genaue Gliederung im Lauf des Projekts erforderlich)
    - Aufbereitung der wesentlichen Punkte in einem Text für den Blog
    - Entscheidungshilfe für den Verantwortlichen
    - Ziel ist es, dass im Projekttagebuch alles eingetragen wird, was gemacht und entschieden wurde
    - Fazit: Zunächst soll geschaut werden, wie andere PG dies gemacht haben (-> Dirk)
- Vorstellung der Regelliste:
    - Bisher ein erster Vorschlag, der Ergänzt werden kann
    - Unterteilung in allgemeine, organisatorische und Diskussionsregeln
    - Zentraler Punkt für Regeln, die sonst in unterschiedlichen Protokollen stehen
- Ordnerstruktur fürs GIT
    - Nach einer kurzen Vorstellung von Artjom und Diskussion über die Ordnernamen, erfolgte eine Abstimmung mit 9 Stimmen für eine Benennung von z.B. "10 - Projektdokumentation", "20 - Quellcode"
- Sollten die Inhalte auf dem ersten Treffen ins Confluence?
    - JA
- Wie soll die Mitarbeiterbefragung aussehen? Einelne Personen....
    - Interview soll als Onlineumfrage gestaltet werden
    - Anschließend können weitere Interview mit einzelnen Personen geführt werden
    - Auch ist ein Workshop und Umstellung möglich
    - Die Fragen müssen wird Dr. Joachim Kurzhöfer abgesegnet werden
    - Die Onlineumfrage muss dabei anonym erfolgen
    - Der Fragebogen wird im Team im Anschluss an die Fragensammlung erstellt (-> Dirk, Tobias, Patrick und Holger)
- Fragen zum Seminarthema:
    - Programmierkonventionen sollen auf Java basieren (Jan)
    - Mehrere Modelle von der Idee zum Business Case (Jessica)
- Logo-Vorschläge:
    - Artom und Vanessa erstellen Vorschläge

    - Anschließend Abstimmung über Vorschläge
- Vorstellung GIT:
    - Der folgende Ablauf ist bei der Einrichtung durchzuführen
        - Installation eines SSH-Terminal (z.B. Putty)
        - Einrichtung und Password ändern
        - Installation von SourceTree/TexStudio (GIT-Programm ist egal, solange es UTF-8 einsetzt)

Top 3: Vorstellung AS Inpro durch Dr. Joachim Kurzhöfer

- AS Inpro hat über 1000 Mitarbeiter an den Standorten Wolfsburg, Berlin, Oldenburg, Köln, Bern, Basel, Miami etc.
- Ca. 75 Mitarbeiter aus den Bereichen Colsulting, Development (neue Produkte und bestehende Produkte/Team mit ca. 4-12 Personen) und Operation (Anpassungen und Betrieb) am Standord Oldenburg
- Die Mitarbeiter wissen bisher nicht von dem Projekt (Vorstellung erfolgt am 19.5.)
- Die Mitarbeiterstruktur ist recht jung (viele Master- und Bachelorabsolventen)
    - Offen für Vorschläge
    - meisten Mitarbeiter arbeiten ausschließlich am Computer
    - Als Diensthandy ein IPhone und viele haben ein IPad, aber auch andere Smartphones
        - Keine Erreichbarkeit rund um die Uhr (Ausnahme: Bereitschaftsdienst)
- Die Herachieebenen sind sehr gering:
    - GFL (Standortleiter) für Oldenburg: Dr. Joachim Kurzhöfer
    - sGFL (Bereichsleiter)
    - Consultant (Mitarbeiter)
- AP Inpro fungiert als eigenständiges Mittelstandsunternehmen mit ca. 150 Kunden aus dem Bereichen Media, Travel, Manifacturing, Healthcare, Energy, Automotive, Transport&Logistik
- Die Lösungen über Technologie (neuste Technologien durch Mobile Anwendungen/Big Data), Prozesskenntnisse (Kenntnisse über branchenspezifiische Prozesse und Methoden) stehen im Mittelpunkt
- Leistungssprektrum beinhaltet u.a. Ideen/Strategien, Prozessberatung, Konzept/Design, Entwicklung/Technologie usw.
- IST-Zustand bei Innovationen:
    - Box in der hinteren Ecke der Kaffeeecke.
    - Es gibt kaum Vorschläge und keine bekannte Innovation bisher

    - Zustand ist derzeit nicht motivierend und zielführend
    - Alternativ: Direkte Kommunikation und persönenbezogene Entscheidung durch den GFL
- Kommunikation erfolgt vorwiegend über SharePoint, Outlook oder den "Flurfunk" (persönliche Gespräche)
    - Auch nach oben erfolgt die Kommunikation auf persönliche Ebene
    - Einen Kommunikationsprozess gibt es bisher nur auf dem Papier, aber real wird dieser nicht eingesetzt
- Bei der Umsetzung der PG sind keine Widerstände zu erwarten
- Herausforderung der Innovation über alle Bereiche der AS Inpro
    - Bisher beziehen sich die Innovationen eher auf den Pausenraum&Mentoring (Ideen für den Arbeitsplatz)
- Bereich für die Vorstellung, Erarbeitung von neuen Bereichen und Technologie
- Controlling kommt vor der Wirtschaftlichkeitsrechnung
- Entscheidungen und Reaktionen sind von den Personen/Stimmung abhängig
- Es können Anzreize über Dank & Arbeitszeitfreigabe gemacht werden

Top 4: Aufgaben zum nächsten Termin

Name	Aufgabe
Alle	MIND-Map mit Ideen zu den Anforderungen füllen
Alle	Fragen für den Fragebogen (bis zum 30.4.2015)
Alle	Grober Ablaufplan und Gedanken zu Urlaub etc.
Dirk	Recherche zum Projekttagebuch anderer Projektgruppen

Top 5: Anmerkungen / Sonstiges

- Dr. Joachim Kurzhöfer wird am 30.04 nicht am Gruppentreffen teilnehmen.
- Es gibt einen Kuchen von Daniel wegen Klingeln des Smartphones. 
- Das nächste Treffen direkt bei AS Impro findet am 28.5.2015 statt
- Anschließend ging es zum Eisessen (ohne Dr. Joachim Kurzhöfer) in die Eisdiele San Marco am Westkreuz

A.7.4. Protokoll vom 30.04.2015

Protokoll_150430

Datum: 30.04.2015

Moderation: H. Eichholz

Protokollant: Tobias Kromke

Abwesend: Jan Wiesemann (abgemeldet), Vanessa Dering (abgemeldet)

Top 1: Weekly Scrum

Bearbeitung des Seminarthemas

Brainstorming Anforderungen

Name	erledigte Aufgabe
Artjom	Website auf Server gesetzt Logos für Website erstellt Recherche zu potentiellen Technologien Git Ordnerstruktur angelegt
Tobias	Recherche zu potentiellen Technologien Recherche zur Gestaltung eines Fragebogens Confluence Dateien neu strukturiert
Patrick	Seite für Plattform und Versionen erstellt Grober Projektplan erstellt Doodle Umfrage
Holger	Protokoll und Moderationsplan erstellt Liste für Fragen an Mitarbeiter erstellt
Daniel	Git angeguckt
Sebastian	Server Rechte angepasst Bilder angepasst Recherche zu potentiellen Technologien
Yaping	Git & Sourcetree angeguckt Recherche SQL vs NoSQL
Jessica	Projektplanung angeschaut
Dirk	Weiterführende Recherche zum Projekttagebuch (vergleich mit ehemaligen Gruppen)

Top 2: Abstimmung und organisatorische Themen

- Betrachtung des Fragenkatalogs für die Mitarbeiterbefragung
- Zu Interviewtechniken informieren um es später begründen zu können
- Durchführung der Umfrage gegen ende Mai
- Interviews unterstützend durch "Papier-Prototypen"
- Betrachtung und Vorstellung der MIND-Map zu den Anforderungen
- Anforderungen anhand ihres Abstraktionsgrades gliedern
- Detailierte Informationen zum Projekttagebuch
- bei alten Gruppendokumentationen Projekttagebuch nicht explizit aufgeführt
- Highlights des Projekttagebuchs in Blogform auf der Website
- Meilensteine reflektieren
- Vorstellung des Projektplans
- Sprintdauer bisher willkürlich dargestellt
- Umfrage und Interviews müssen noch eintragen werden
- Überlegung, welche Software wird benötig
- Entscheidung für mySQL
- Verwendung der selben Java und Eclipse Version aller
- UML: Visual Paradigm ; einfache Modellierung geht auch mit Visio
- Weitere Vorgehen bei der Regelliste
- Struktuierung der Dateien in Confluence
- bereits strukturiert
- Logo
- Entscheidung für Logo 1 -> Font anpassen (nicht fleckig)
- Website / Blog
- Sitemap unten entfernen
- Letzter Beitrag weiter oben auf der Startseite setzen

Top 3: Aufgaben zum nächsten Termin

Seminarthema bearbeiten

Informieren zu GWT / Vaadin

Name	Aufgabe
Dirk	ersten Eintrag in Projekttagebuch
Artjom	Logo & Website anpassen
Tobias	Fragebogen Aufbau

Top 4: Anmerkungen / Sonstiges

Abgabe der schriftlichen Ausarbeitung 22.06.2015

A.7.5. Protokoll vom 07.05.2015

Protokoll_150507

Datum: 07.05.2015

Moderation: Tobias Kromke

Protokollant: Yaping Lian

Abwesend: -

Top 1: Weekly Scrum

Name	erledigte Aufgaben
Artjom	• Anpassung der Webseite und Logo • Bearbeitung der Seminararbeit • Einrichtung des Eclipses • Recherchieren der Funktion von Vaadin und GWT
Sebastian	• Beschäftigung mit der Seminararbeit • Beschäftigung mit den Vaadin und GWT
Tobias	• Erstellung der Struktur vom Vortrag • Beschäftigung mit der Umfrage • Recherchieren der Technik für die Dokumentation • Recherchieren der Funktion von Vaadin und GWT
Jan	• Beschäftigung mit der Seminararbeit • Beschäftigung mit den Vaadin und GWT
Patrick	• Anpassung des Projektplans • Beschäftigung mit dem Seminararbeit
Holger	• Literaturrecherche zum Aufbau des Fragebogens • Beschäftigung mit der Seminararbeit
Daniel	• Beschäftigung mit der Seminararbeit • Beschäftigung mit dem Vaadin
Dirk	• Beschäftigung mit dem Tutorium Vaadin • Beschäftigung mit dem Projekttagebuch • Bearbeitung der Seminararbeit
Yaping	• Beschäftigung mit der Seminararbeit • Beschäftigung mit den Vaadin und GWT
Jessica	• Beschäftigung mit der Seminararbeit • Beschäftigung mit den Vaadin und GWT

Vanessa	• Fertigstellen vom Logo • Recherchieren allgemeines Anreizsystems • Bearbeitung der Seminararbeit • Recherchieren der Funktion von Vaadin und GWT

Top 2: Abstimmungen und organisatorische Themen

- Vorstellung des Projekttagebuches als Blog
    - Datum vor dem Titiel schreiben
    - Timeline nach Kategorie sortieren
- Diskussion über Entwicklungswerkzeug
    - Entscheidung für Vaadin
    - Eclipse Version: Eclipse 4.4.2
    - Java Version: Java 8.0
    - Kombination zwischen Buildsystem und Vaadin
- Diskussion über den Fragebogen
    - 4-5 Leute unserer Projektgruppe werden den Fragebogen testen.
    - Erinnerungsmail an Mitarbeiter schicken kurz vor dem Endtermin
    - Der Fragebogen sollte folgende Eigenschaften haben:
        - klare Instruktion
        - möglichst kurze, einfache und verständliche Fragen
        - offene Fragen sind notwendig
    - wichtig: Fragebogen müssen anonym bleiben, außer jemand seinen Name für das spätere Interview hinterlegen will.

Top 3: Aufgaben zum nächsten Termin

- Fertigstellen der Präsentation
- Informieren Buildsystem (Maven, Ant ...)

Name	Aufgabe
Vanessa	Serverarchitektur einrichten

Top 4: Anmerkungen/Sonstiges

- Da am 22.05.2015 unsere Seminararbeiten präsentiert werden, gibt es keinen regulären Moderationsplan.
  Nächster Moderationsplan sollte bis 27.05.2015 hochgeladen werden.

A.7.6. Protokoll vom 28.05.2015

Protokoll_150528

Datum: 28.05.2015

Moderation: Yaping Lian

Protokollant: Patrick Smit

Abwesend: Stefan Wunderlich

Top 1: Weekly Scrum

- Alle: Präsentation fertig gestellt, Hausarbeit, Diskussion am 27.05.2015
- Vanessa: Einrichtung LimeSurvey
- Artjom: Eingelesen in diverse Themen

Top 2: Besprechung der Entscheidungen

- Festlegung auf Scrum (Dirk Scrum Master, Patrick Product Owner)
- Testen (kontinuierliches, automatisches Testen mit JUnit, ggf. Selenium)
- Oberflächendesign (Usability Engeneering, basteln von Prototypen)
- Business Case (erst kleiner Vorab-Business-Case, danach qualifizierter Business-Case, wichtigsten Zahlen aus Business-Case erfassbar)
- Programmierkonventionen (Wird von Jan als Dokument vorbereitet)
- Entwicklungsstufen (genauer Ablauf wird in kleiner Gruppe herausgearbeitet, Gates und Zuständigkeiten, Problem+Ideenerfassung, Diskussion, Ideenprüfung, Übergang zum Business Case)
- Gamification (Elemente primär der ersten Stufe verwenden, Elemente hängen aber stark von dem System ab und kann daher noch nicht festgelegt werden)
- Kommunikation (7 Stufen werden angewendet, Projektleiter und Scrum-Master als Ansprechpersonen)
- Dokumentation (alle Dokumentationen der Präsentationen werden benötigt)

Scrum Master kümmert sich nur darum, dass der Scrum Prozess sauber durchgeführt wird. Projekt Manager um alles weitere u.a. zur Problemlösung.

- Benutzen Maven als Buildsystem

Top 3: Aufgaben zum nächsten Termin

- Alle: Hausarbeit bearbeiten

- In jeweiligen Gruppen die entsprechenden Themen vorbereiten und aufbereiten
    - Artjom, Tobi, Seb: User Management Entwicklung Login
    - Vanessa und Yaping: Oberflächendesign
    - Dirk: Testautomation
    - Jan, Jessica, Holger: Prozessentwicklung
    - Yaping: Erstellt Gesamt-Dokument, jeder einzelne begründet seine Entscheidung, sofern eine getroffen wurde
    - Interview-Gruppe: Kümmert sich darum, dass der Fragenkatalog erstellt wird

Top 4: Anmerkung / Sonstiges

- Der Technologie-Tag der Lufthansa in Norderstedt wird von der PG mit 2-3 Personen besucht
    - Die Präsentation wird ca. 3 mal a 25 Minuten + 5 Minuten Diskussion gehalten
    - Publikum sind die Mitarbeiter der Lufthansa
- Neue Fotos werden aufgenommen (11. Juni wird als Termin festgelegt)

A.7.7. Protokoll vom 04.06.2015

Protokoll_150604

Datum: 04.06.2015

Moderation: Patrik Smit

Protokollant: Dirk Tesche

Abwesend: Stefan Wunderlich, Jessica Schulte, Daniel Ahlers

Top 1: Weekly Scrum

- Seminararbeiten

Top 2: Abstimmungen und organisatorische Themen

- Hompage
    - Artjom stellte eine alternative Homepage vor, welche auf der Startseite unser Projekt visualisiert.
    - Die Startseite soll aus einer Kombination von Blogeinträgen und kurzer Projektbeschreibung bestehen.
    - Die Präsentationen auf der Webseite sollen alle in PDF umgewandelt werden und mit einer kurzen Beschreibung des Themas ergänzt werden.
- User Management Entwicklung, Login
    - ER Modell wurde erstellt,
    - Vorläufige Probleme mit dem Login wurden erläutert,
    - MVP soll als Entwurfsmuster verwendet werden,
    - Projekt wurde im GIT angelegt.
- Oberflächendesign
    - Erste Vorschläge zur Startseite und Login wurden vorgestellt.
- Testautomation
    - Testumgebung wurde erstellt,
    - erste Test wurden durchgeführt.
- Prozessentwicklung
    - 5-Phasen-Modell wurde vorgestellt,
    - Problem - Vorschläge - Ideen - Vorhaben - Business Case
    - Vorschlag wurde diskutiert und Vorschläge / Anmerkungen werden eingearbeitet.
- Interview
    - Fragenkatalog wurde vorgestellt,

    - Ziele des Fragebogens wurden erläutert,
    - Weitere Maßnahmen werden im Anschluss des Meetings durch die Interviewgruppe durchgeführt.
- Projekthandbuch
    - Erster Entwurf wurde von Yaping vorgestellt,
    - Der Aufbau des Handbuchs wird von Yaping in Latex eingepflegt.
- Kleiderordnung für das Foto-Shooting
    - Für die neuen Fotos sind Jens und Hemd festgelegt worden.
    - Die Fotos werden am 11.06.15 aufgenommen.
- Raumsituation
    - Patrik klärt mit den beiden anderen PG die Raumnutzung.
    - Nachtrag: Der PG steht bis August der Arbeitsraum am Donnerstag (ganztägig) und Freitag (08:00 - 13:00) zur verfügung.

Top 3: Aufgaben zum nächsten Termin

- Alle: Hausarbeit bearbeiten.
- In jeweiligen Gruppen die entsprechenden Themen vorbereiten und aufbereiten.

Top 4: Anmerkungen / Sonstiges

- Daniel wird die Gruppe User Management Entwicklung, Login unterstützen.

A.7.8. Protokoll vom 15.06.2015

Protokoll_150611

Datum: 11.06.2015

Moderation: Dirk Tesche

Protokollant: Jessica Schulte

Abwesend: Vanessa Dering

Top 1: Weekly Scrum

- alle haben an ihren Seminararbeiten gearbeitet
- keine weiteren Anmerkungen

Top 2: Abstimmungen und organisatorische Themen

- Festegung eines neuen Projektmanagern intern
- Festlegung auf internationalen Zitationsstil, der auch in der VLBA Vorlage verwendet wird
- Frage von Yaping bezüglich ihrer Seminararbeit
    - Aufteilung der Anforderungsdokumentation in verschiedene Bereiche
    - Testdokumentation: genaue Dokumentation der Tests notwendig
- Dateiorganisation im Comfluence/Git: Projektplanung und Kleingruppendokumente in Git hochladen
- Fragen an Joachim: Antworten auf die Fragen festhalten/verschriftlichen in einem Git-Ordner
- Projektgruppenraum: Situation wurde noch mal besprochen, Kompromiss muss noch mit anderen Projektgruppen abgeklärt werden
- für neues Gruppenmitglied (Jan Fischer) ausgesprochen
- Scrum Prozess starten:
    - bis heute sollen alle Gruppen ihr Product Backlog an Dirk schicken
    - Donnerstags Retroperspektive
    - Sprint-Dauer 2 Wochen
    - Gruppe User Management. User Klasse erstellen, Einarbeitung ins Framework, JDBC Connector, MVP Konzept aufbauen
    - Testmanagement: Zeigen wie Tests in JUnit funktionieren, Zeigen wie ein Testfall umgesetzt wird
    - Gruppe Fragebogen: Pre-Test, Vorstellung des Fragebogens vorbereiten
    - Gruppe: Prozess-Design: detailierte Beschreibung, Einarbeitung der neuen Vorschläge
    - Gruppe Oberflächen-Design: Farbdesign, Mockup für Login

- Vorstellung des erarbeiteten Prozesses der Prozzdesign-Gruppe
    - Vorschlag: Ideen werden bewertet und ab einer bestimmten Bewertung wird ein Verantwortlicher dafür gesucht.
    - Vorschlag: Ideen werden an Vorschläge gebunden, beim Erstellen einer Idee durch den Ideenersteller
    - 1:1 Beziehung zwischen Vorhaben und Business Case, weil Business Case umfangreich
    - nach Business Case Phase dokumentieren des weiteren Weges, was mit dem Business Case passiert
    - Begriffe Idee und Vorschlag müssen noch definiert werden

Top 3: Aufgaben zum nächsten Termin

- Tobi kümmert sich um die Antworten auf die Fragen an Joachim
- Dirk trägt die Tickets in JIRA ein
- Vorstellung des Fragebogens am nächsten Termin durch die Gruppe
- beim nächsten Termin besprechen, wie die Ausarbeitungen abgegeben werden sollen
- beim nächsten Termin soll der neue Projektmanager bestimmt werden

Top 4: Sonstiges/Anmerkungen

- Fragegruppe braucht Mockups Alternativen vom Hauptmenü von der Oberflächengruppe bis Sonntag

A.7.9. Protokoll vom 18.06.2015

Protokoll_150618

Datum: 18.06.2015

Moderation: Jessica Schulte

Protokollant: Jan Wiesemann

Abwesend: Jan Fischer (abgemeldet)

Top 1: Weekly Scrum

Name	erledigte Aufgaben
Alle	• Seminararbeit
Tobias	• Überarbeitung der Umfrage • Konfiguration der Datenbank • in Vaadin eingelesen
Sebastian	• Verbindung zwischen GIT und Datenbank • in Vaadin eingelesen
Artjom	• Konfiguration der Datenbank
Dirk	• Überarbeitung der Umfrage
Patrick	• Erstellung der Umfrage in LimeSurvey, Gespräch mit Joachim
Holger	• Überarbeitung des Konzeptes
Yaping	• Gestaltung des Oberflächendesigns
Jan	• Überarbeitung des Konzeptes
Vanessa	• Gestaltung des Oberflächendesigns • SCRUM Prozess
Jessica	• Überarbeitung des Konzeptes

Top 2: Abstimmungen und organisatorische Themen

- Projektmanager festlegen
    - Als Projektmanager für die 2. Projektphase wurde Jan Wiesemann festgelegt
- Vorstellung des Fragebogens und Sammlung von Verbesserungsvorschlägen
    - Kleine Einführung zum Hintergund und generell mehr Erläuterungen hinzufügen
    - Weniger spezielles Vokabular verwenden
    - Begründungen für Antwortmöglichkeiten (z.B. bei Designvorschlägen) hinzufügen
    - Mehr Freitextfelder hinzufügen
    - Bewertungsskala überarbeiten
- Abgabe der Seminararbeiten
    - bis zum 22.06.2015 um 23:59 Uhr an die Betreuer
- SCRUM in JIRA
    - Jessica stellte die Unterstützung von SCRUM in JIRA vor
    - Der Prozess muss genauer definiert werden
        - Meeting am 24.06.2015 um 10:00 Uhr im HGAS Raum (AStA)
- Prozessdesigns
    - Holger stellte das überarbeitete Konzept vor
    - Vorschlag "Ideenbörse" wurde angenommen
    - Vorschlag "Flugvokabular" wurde angenommen
- Projektglossar
    - Vorschlag wurde angenommen
    - Projektglossar wird in GIT gepflegt

- Sonstiges
    - Budget zur Beschaffung von Literatur
        - Nicht vorhanden
        - Möglichkeit 1: Anschaffungsvorschlag an Bibliothek
        - Möglichkeit 2: Finanzierung aus Beiträgen der Projektmitglieder
    - Vision formulieren
        - Welches Problem wollen wir lösen?
        - Was wollen wir erreichen?
        - Wie wollen wir das erreichen?
        - Wo wollen wir hin?
        - Welche Grundfunktionalitäten wollen wir umsetzen?
    - Raumsituation
        - HGAS-Raum
            - im AStA (zweite Tür rechts)
            - kann für spontane Treffen benutzt werden
        - PC-Schulungsraum
            - A4 2-201
            - könnte für regelmäßige Treffen (besonders im nächsten Semster) verwendetet werden
            - frühzeitige vorherige Reservierung beim Raumbüro (Tel.: 2483 oder 2545)
            - Öffnungszeiten
                - während der Vorlesungszeit: Mo - Fr, 8:00 - 20:00 Uhr
                - während vorlesungsfreier Zeit: Aufschließen bei Bedarf durch Hausmeister

- Einladung zum PG-Boule-Turnier 2015
  - Termin: 08.07.2015 von 14:00 - 18:00 Uhr auf dem Ascheplatz der Uni in Wechloy
  - Einladung wurde angenommen

Top 3: Aufgaben zum nächsten Termin

Name	Aufgabe
Alle	• Abschließen der Seminararbeit • Genauere Definition des SCRUM Prozesses (24.06.2015,10:00 Uhr, HGAS-Raum)
Artjom, Sebastian, Tobias,	• In "Book of Vaadin" einlesen
Dirk, Holger, Patrick, Tobias	• Überarbeitung des Fragebogens
Holger, Jan, Jessica	• Überarbeitung des Konzeptes
noch festzulegen	• Formulierung einer Vision • Entwicklung Meilensteinplan

Top 4: Anmerkungen / Sonstiges

- nächster Termin: 25.06.2015, 12:30 Uhr
- nächster interner Termin: 24.06.2015, 10:00 Uhr, HGAS-Raum

A.7.10. Protokoll vom 25.06.2015

Protokoll_150625

Datum: 25.06.2015

Moderation: Jan Wiesemann

Protokollant: Daniel Martin Ahlers

Abwesend: Vanessa Dering, Jan Fischer,

Top 1: Weekly Scrum

Name	erledigte Aufgaben
Alle	• Abgabe der Seminararbeit um Montag, den 22.06.15 • Genauere Definition des SCRUM Prozesses (24.06.2015,10:00 Uhr, HGAS-Raum)
Tobias	• "Book of Vaadin"
Sebastian	• "Book of Vaadin"
Artjom	• "Book of Vaadin"
Dirk	• Überarbeitung der Umfrage
Patrick	• Überarbeitung der Umfrage
Holger	• Überarbeitung der Umfrage
Yaping	• Gestaltung des Oberflächendesigns
Jan W.	• Überarbeitung des Konzeptes
Jessica	• Überarbeitung des Konzeptes

Top 2: Abstimmungen und organisatorische Themen

- **Vorstellung des aktuellen Fragebobens**
    - Anmerkungen der letzten Sitzung wurden umgesetzt
    - Zu Beginn der Umfrage sollen Begrifferklärungen durchgeführt werden

    - Generelle Information von Joachim: Der Fokus soll nicht nur innerhalb der Lufthansa Industry Solutions gelten, auch für Kunden
    - **Aufgabe bist Montag:** Die eMail mit dem Fragebogen als Verlinkung anfertigen, welche Joachim an die Mitarbeiter sendet
- **Vorstellung des erarbeiteten Workflows/ Konzeptes**
    - Anmerkunden von letzter Sitzung wurden übernommen
    - Kleine Änderungen: Einführung von Addons (z.B. Punktebörse), eigene Begriffe für Rollen innerhalb der Plattform
    - Diskussion über Bewertungssysteme von Ideen. Es wurde ein Task im Backlog erstellt
- **Vision / Mission**
    - Die am Mittwoch, den 24.06. verfasste Vision/Mission wurde angenommen - einzusehen unter Vision und Mission im Confluence

Top 3: Aufgaben zum nächsten Termin

- Die Aufgaben zum nächsten Termin leiten sich aus dem in Top 5 beschriebenen Sprint Planing ab

Top 4: Anmerkungen / Sonstiges

- **Anfrage an die Betreuer: Feedback über unser Arbeitsverhalten**
    - In jeder Woche wird es ein Feedback an zwei Leute aus dem Team geben. Entsprechend erhält jeder alle 6 Wochen ein Feedback über seine Arbeit.
    - Im Laufe der Sitzung kam es zu einem ersten sehr positives Feedback über die erste Projektphase.
- **Neue Moderations-/ und Protokollübersicht online**
    - Vanessa und Holger können an ihren Terminen leider nicht, im laufe der Woche wird die Übersicht daher aktualisiert

Top 5: Sprintplanung

- Erstellung von Tasks zu jedem Teilbereich der Gruppe. Dauer des Sprints: 2 Wochen
- **Kleines Feedback der Betreuer**: Die Sprintplanung im Detail soll zukünftig nicht zu den Meetings erfolgen, lediglich eine Diskussion darüber
- Die Tasks werden komplett ins GIT hochgeladen
- Abschluss: Planing Poker für den ersten Task "Workshop: Vollständige Entwicklungsumgebung"
- Weiteres Planing Poker ebenfalls außerhalb des Meetings

A.7.11. Protokoll vom 02.07.2015

Protokoll_150702

Datum: 02.07.2015

Moderation: Daniel Martin Ahlers

Protokollant: Jan Fischer

Abwesend: Patrick Smit

Top 1: Weekly Scrum

Name	erledigte Aufgaben
Alle	
Tobias	• "Book of Vaadin" • Korrektur des Fragebogens
Sebastian	• "Book of Vaadin"
Artjom	• Deutsches Vaadin-Buch • Tasks erstellt und ausformuliert mit Dirk
Dirk	• Fragebogen • Entwurf der Email zum Fragebogen • Jira-Tasks angelegt • Organisatorisches
Patrick	
Holger	• Template , kleine Änderung (in der Gruppe)
Yaping	• Wireframe erstellt • Tools / Artikel zu Farbschemata recherchiert
Jan W.	• Glossar angelegt • Template, kleine Änderungen (in der Gruppe)
Jessica	• Template, kleine Änderungen (in der Gruppe)
Jan F.	• "Book of Vaadin" durchgearbeitet • Design Pattern • Möglichkeiten zur CDI in Vaadin recherchiert
Daniel	• Vaadin
Vanessa	• Wireframe erstellt

Top 2: Abstimmungen und organisatorische Themen

- Vision und Mission
    - **Ergebnis:** Vison und Mission wird so übernommen und aufi der Homepage präsentiert
- Vorstellung Prozessdesign
    - **Ergebnis:** Gruppe ist mit dem Dokument zur Beschreibung einer Stage einverstanden
- Vorstellung Wireframes
    - **Ergebnis:** Die neuerstellten Wirframes werden in die Umfrage übernommen
- Vorstellungen Fragebogen:
    - **Ergebnis:** Gruppe ist damit zufrieden
- Scrum Tasks
    - Mit-Bearbeiter lassen sich nun über die Gruppe "Users" eintragen

Top 3: Aufgaben zum nächsten Termin

- An den Tasks weiter arbeiten

Top 4: Anmerkungen / Sonstiges

- Emaileinladung zum Fragebogen/Umfrage
    - **Ergebnis:** Umfrage wird 2 Wochen online sein (Start: 06.07.15), nach 1 1/2 Wochen wird eine Erinnerungsmail geschickt
- Erster Programmiersprint
    - Der erste Programmiersprint wird voraussichtlich erst in der Vorlesungsfreien beginnen, bis dahin müssten auch die ersten Userstories fertig sein
- Einzelfeedback
    - Das erste Feedback gibt es in 2 Wochen, 16.07.15, es wird dazu eine Liste von Jan W. angelegt (Termin wünsche müssen in der Gruppe besprochen werden)
- Treffen bei Lufthansasystems
    - nächstes Treffen findet 16.07.15

A.7.12. Protokoll vom 09.07.2015

Protokoll_150709

Datum: 09.07.2015

Moderation: Jan Fischer

Protokollant: Artjom Baranow

Top 01: Weekly Scrum

Name	erledigte Aufgaben
Alle	
Sebastian	• Vorbereitung auf den Vaadin-Workshop • Handout/Cheatsheet über Maven • Deutsches Vaadin-Buch
Artjom	• Vorbereitung auf den Vaadin-Workshop • Präsentation und Handout für Vaadin-Einführung • Deutsches Vaadin-Buch
Dirk	• Vorbereitung auf den Vaadin-Workshop • Deutsches Vaadin-Buch • Aktivierung und Überwachung des Fragebogens
Patrick	• Deutsches Vaadin-Buch • Gamification-Elemente untersucht und auf Nutzbarkeit innerhalb der Projektgruppe untersucht
Daniel	• Vaadin-Workshop Vorbereitung • Deutsches Vaadin-Buch • Handout für grundlegendes in Eclipse + Verknüpfung mit GIT und Tomcat
Jan W.	• Deutsches Vaadin-Buch • Grundlegende Funktionalitäten für Prozessdesign-Stages in Form einer User-Story erstellt • Use-Cases
Tobias	• Vaadin-Workshop Vorbereitung

	• Präsentation und Handout für MVP und Entwurfsmuster o Abgrenzung zu MVC und MVVM
Jessica	• Deutsches Vaadin-Buch • Grundlegende Funktionalitäten für Prozessdesign-Stages in Form einer User-Story erstellt • Use-Cases
Holger	• Deutsches Vaadin-Buch • Grundlegende Funktionalitäten für Prozessdesign-Stages in Form einer User-Story erstellt • Use-Cases
Yaping	• Erstellung mehrerer Farbschemas • Dokumentation von Entscheidungen • In das Tool "Axure" eingearbeitet
Vanessa	• Erstellung mehrerer Farbschemas • Dokumentation SE-Vorgehensmodell • In das Tool "Axure" eingearbeitet und kleinen Prototypen erstellt
Jan F.	• Vorbereitung auf den Vaadin-Workshop • Deutsches Vaadin Buch • Eclipse-Beispiel vorbereitet, indem zentrale Dinge wie MVP, RPC, Design-Abindung in Vaadin umgesetzt werden

Top 2: Sprint-Abschluss

- **Sprint-Review und Retrospektive**:
  - Sollen intern geführt werden, sofern es protokolliert ist, damit die Betreuer ggf. Einblicke über die Ergebnisse sehen können

Top 3: Abstimmung und organisatorische Themen

Vorstellung bisheriger Ergebnisse:

- Oberflächendesign (UI):
    - Es wurden 5 verschiedene Farbschemas vorgestellt
        - Mehrheitliche Entscheidung fiel auf **Farbschema 1**
    - Axure wird als Tool für Prototyp-Erstellung akzeptiert und soll in Zukunft verwendet werden

Top 4: Aufgaben zum nächsten Termin

- Alle:
    - Vorbereitung auf den Vaadin-Workshop
    - Deutsches Vaadin Buch lesen
    - Eclipse startbereit machen
- Daniel:
    - Schickt nochmal einen Download-Link für Eclipse an alle Projektmitglieder
- Vanessa:
    - Schaut nochmal nach, welche Themen bei Axure hinsichtlich der Nutzung und Weiterverbreitung
      von Prototypen beachtet werden muss

A.7.13. Protokoll vom 16.07.2015

Protokoll_150716

Datum: 16.07.2015

Moderation: Sebastian Beckmann

Protokollant: Tobias Kromke

Abwesend: Artjom Baranow, Jessica Schulte, Daniel Martin Ahlers

Top 1: Weekly Scrum

Name	erledigte Aufgabe
Tobias	Workshop vorbereitet und durchgeführt
Sebastian	Workshop durchgeführt Book of Vaadin gelesen
Jan F.	Workshop durchgeführt MVP-Framework anpassen + Realisierung für das Projekt Hibernate + Vaadin beschäftigt Beispiel zum Verständnis vorbereitet
Dirk	Workshop nachbereitet Anlegen der Tasks im Jira
Patrick	Workshop nachgearbeitet Gamification Elemente verfeinert
Holger	Anpassung der Stages Erstellung der fehlenden User Storys
Vanessa	mit Vaadin weiter beschäftigt mit Oberflächen Prototypen beschäftigt Versucht Use-Cases für Stage 1 umzusetzen
Yaping	Workshop nachbereitet Farbschema dokumentiert
Jan W.	Grobkonzept korrigiert User Storys weiter ausformuliert für 1. Stage Meilensteinplanung erstellt Dokument für Programmierkonventionen erstellt
Jessica	Konzept erneut überarbeitet User Storys "Problemerstellung" ausformuliert und abgeschlossen MVP in Vaadin Beispiel nachgearbeitet

Top 2: Sprint Planning

- Sprint endet am 30.07.2015
- Ergebnisse werden in der Sitzung am 30.07.2015 vorgestellt
- 30.07.2015 für die nächste Sprintplanung freihalten

Top 3: Abstimmungen und organisatorische Themen

- Fragebogen: bisher 16 Personen bei der Umfrage teilgenommen; 3 vollständige Antworten
- Joachim sendet Anfang der Woche eine Erinnerung zur Beantwortung des Fragebogens
- Meilensteinplan:
    - Abschluss der Erhebung von Anforderungen bis 02.08.2015
    - Beginn der Realisierung 03.08.2015
    - Feature Freeze 01.02.2016
    - Als Endergebnis wird kein fertiges Produkt erwartet; ein guter Prototyp ist ausreichend
    - Abschließende Tests ergänzen / berücksichtigen
    - Schulungen nicht benötigt
    - Projektgruppendokumentation nicht vergessen
    - CeBit (14.03.2016 - 18.03.2016) ergänzen
- Programmierkonventionen
    - Inhalte der Seminararbeit tabellarisch aufbereitet
- Joachim ist die nächsten 2 Wochen nicht da
- Definition of Done klären
- In den Semesterferien viel erarbeiten

Top 4: Aufgaben zum nächsten Termin

Name	Aufgabe
Tobias	IMPACT 47 IMPACT 63
Jan F.	IMPACT 66
Sebastian	IMPACT 69
Jan W.	IMPACT 67
Dirk	IMPACT 47 IMPACT 63
Yaping	IMPACT 68
Holger	IMPACT 70 IMPACT 67
Patrick	IMPACT 69 IMPACT 63
Vanessa	IMPACT 67
Artjom	IMPACT (70 oder 68)
Daniel	IMPACT (70 oder 68)
Jessica	IMPACT 51

A.7.14. Protokoll vom 23.07.2015

Protokoll_150723

Datum: 23.07.2015

Moderation: H. Eichholz

Protokollant: Vanessa Dering

Abwesend: Jan Wiesemann, Dirk Tesche

Top 1: Weekly Scrum

Name	erledigte Aufgabe
Tobias	Nachvollziehbarkeit des Bewertungssystems analysiert
Sebastian	Aufgrund von Klausuren keine Tätigkeiten
Jan F.	Erstellung der Seminarpräsentation
Patrick	Aufgrund von Klausuren keine Tätigkeiten
Holger	Erstellung und Bearbeitung der Stagebeschreibung Stage 5
Vanessa	Erstellung des ersten Oberflächenprototypen bearbeiten der Dokumentation Erstellung und Bearbeitung der Stagebeschreibung Stage 2
Yaping	Erstellung und Bearbeitung der Stagebeschreibung Stage 3
Jessica	Erstellen eines ER-Diagramms für Stage 1 Erstellung und Bearbeitung der Stagebeschreibung Stage 3
Daniel	Erstellung und Bearbeitung der Stagebeschreibung Stage 5
Artjom	Erstellung und Bearbeitung der Stagebeschreibung Stage 3

Top 2: Abstimmungen und organisatorische Themen

- Sitzung der PG findet in der vorlesungsfreien Zeit wöchentlich Donnerstags zwischen 12-14 statt.
- Bisher gibt es 8 vollständige und 19 unvollständige Antworten auf den Fragebogen.
- Die Umfrage wird am Montag den 27.07.2015 um 12.00 Uhr geschlossen.

Top 4: Aufgaben zum nächsten Termin

- Termin finden in den Semesterferien, für eine Woche intensive Aufgabenbearbeitung

Name	Aufgabe
Tobias	IMPACT 47 IMPACT 63
Jan F.	IMPACT 66
Sebastian	IMPACT 69
Jan W.	IMPACT 67
Dirk	IMPACT 47 IMPACT 63
Yaping	IMPACT 68
Holger	IMPACT 70 IMPACT 67
Patrick	IMPACT 69 IMPACT 63
Vanessa	IMPACT 67
Artjom	IMPACT 68
Daniel	IMPACT 70
Jessica	IMPACT 51

Top 5: Sonstiges

- Präsentieren der Semiarpräsentation von Jan Fischer

A.7.15. Protokoll vom 30.07.2015

Protokoll_150730

Datum: 30.07.2015

Moderation: Artjom Baranow

Protokollant: Sebastian Beckmann

Abwesend: Vanessa Dering, H. Eichholz, Yaping Lian

Top 1: Weekly Scrum

Name	Erledigte Aufgaben
Jan W.	Meilensteinplanung angepasst, Räume gebucht, Stage 2 abgeschlossen
Jessica	Stage1 abgeschlossen + Präsentation, Stage3 abgeschlossen
Jan F.	Minimal Beispiel mit Vaadin CDI
Dirk	Fragenbogen abgeschlossen, Bewertungssystem angeschaut aufbereitet, Organisatorisches
Daniel	Stage 5 abgeschlossen
Patrick	Stage 4 abgeschlossen
Tobias	Bewertungssystem erarbeitet
Sebastian	Stage 4 abgeschlossen
Artjom	Stage 3 abgeschlossen

Top 2: Abstimmungen und organisatorische Themen

- Mittwochs und Donnerstags in der vorlesungsfreien Zeit sind Räume gebucht. Plan ist im Confluence vorhanden. Anwesenheit
- Vor-Ort-Vertretung für Projektleitung: Tobias Kromke
- Hackathon: Termin folgt
- Tasks für JIRA müssen in Zukunft kleiner bzw. atomarer erstellt werden.
- FishEye-Zugang wird von Stefan Wunderlich besorgt.

Top 3: Berichte aus den einzelnen Konzeptionsgruppen

- Umfrage nicht repräsentativ, da zu wenig Personen dran teilgenommen haben. Zudem widersprachen sich viele Antworten. Bericht zur Auswertung der Umfrage ist im GIT.
- Keine weiteren Interviews in nächster Zeit, da sich in der Umfrage keiner freiwillig gemeldet hat.

- Für die Findung des Bewertungssystem muss eine neue Arbeitsgruppe gebildet werden, die konkrete Vorschläge ausarbeitet. (KISS-Prinzip)
- Anonyme Bewertung möglich, wobei Avatare zur Wiedererkennung genutzt werden sollten.
- Wichtig: Auswahl des Bewertungssystems mit Literatur belegen.
- Ideen sollten nicht löschbar sein, sondern nur archivierbar.
- UseCases für Administration etc. müssen noch erstellt werden.
- Es sollte bei Projekten ein Feld geben, um verwandte Projekte zu verlinken.
- Risikofaktoren werden in Stage3 erhoben.
- Pate/Projektleiter sollte beim Übergang von Stage 3 auf Stage 4 bestimmt werden.
- Usermanagement muss noch erarbeitet werden.

Top 4: Sonstiges

- Sprint war zu kurz, nächster Sprint 3 Wochen
- **Kommenden Dienstag um 14:30 treffen**

Top 5: Sprintplanung

- Konzeption eines Bewertungssystems (Tobias, Jan F. und Dirk, Sebastian)
- Allgemeines, komplettes Datenmodel
    - Hibernate + JPAContainer, Tutorial in Kombination mit Vaadin (Daniel und Artjom)
- Konzeption: User Rollen aus den Stages sammeln und definieren (Patrick und Jessica)
- Declarative UI (Yaping, Vanessa, Holger und Artjom)
- Login und Registrierung mit Vaadin implementieren, Dummy Model (Sebastian, Tobias und Dirk)
- Prototyp in Axure fortführen (Vanessa, Yaping, Holger und Artjom)
- Vaadin Beispiel zur Einführung (ALLE)

Tasks in JIRA für die Aufgaben müssen von den jeweiligen Bearbeitern erstellt werden.

A.7.16. Protokoll vom 06.08.2015

Protokoll_150806

Datum: 6.8.2015

Moderation: Vanessa Dering

Protokollant: H. Eichholz

Abwesend: Jan Wiesemann, Tobias Kromke, Jens Siewert

Top 1: Weekly Scrum

Name	Erledigte Aufgaben
Jessica	• Ausarbeitung der Rollenkonzepte für Stage 2/3
Jan F.	• Erweiterung des CDI-Beispiel • Ausarbeitung zur Bewertungssystems • Mögliches Problem: Mehrfache Frames sind in Vaddin nicht vorgesehen
Dirk	• Bearbeitung der Tasks des Sprints • Administration des Sprintmeetings • Zeitproblem beim Login
Daniel	• Bearbeitung der Tasks • Erstellen eines ersten Beispiels mit Hybernate
Patrick	• Tasks für autonome Rollen erstellt • Treffen mit der Gruppe für die Rollen pro Stage
Yaping	• Fortsetzung des UI-Design • Befassung mit CSS/SCSS
Sebastian	• Beginn der Login-Programmierung • Vaddin-Beispiel
Artjom	• Definition der Tasks zum Datenmodell • Erstellung des ER-Modell • Auftakttreffen für das Datenmodell
Holger	• Befassung mit CSS/SCSS • Auftakttreffen für das Datenmodell
Vanessa	• Erweiterung des Prototyp zum UI-Design • Ergänzung des Projekthandbuchs • Befassung mit CSS/SCSS
Stefan	• Nachfragen wegen Fisheye: o Serverantwortlichen derzeit im Urlaub

	○ Fisheye funktoniert derzeit nicht wegen dem Server
Joachim	• Drei Ansprechpartner für die Mitarbeiterbefragung gefunden (Kontaktdaten kommen später)

Top 2: Abstimmungen und organisatorische Themen

- Abstimmung zum zweiten Termin (Scrum-Prozess)
  - Am derzeit vorgeschlagenen Termin können alle (außer Patrick)
  - Das Treffen findet nun immer am Dienstag bzw. Mittwoch vor dem Sprintende statt
  - Am **Mittwoch 19.8.** findet das nächste Treffen statt (Uhrzeit wird noch diskutiert, derzeit Vorschlag um 1000)
  - Findet das Treffen an einem Dienstag statt, so beginnt dies um 1430
- Vaadin-Designer:
  - Alpha-Version: Ermöglicht nur eine bestimmte Anzahl der Frames
  - Eher schwer eine Lizenz über die Uni zu bekommen (Kein Haushalt vorhanden)
  - Einfachste Lizenz kostet 13 Dollar im Monat pro Entwickler
  - Stefan erkundigt sich bezüglich ggf. vorhandenen Lizenzen
- SCSS vs. CSS:
  - SCSS wird benutzt, da dies übersichtlicher ist für verschiedene Browser
- Axure:
  - Programm ist zu mächtig für uns, daher wird sich auf die Funktonen konzentiert, die Vaadin bietet

Top 3: Aufgaben zum nächsten Termin

- Für alle: Bearbeitung der Tasks für den Sprint

A.7.17. Protokoll vom 13.08.2015

Protokoll_150813

Datum: 12.08.2015

Moderation: Tobias Kromke

Protokollant: Yaping Lian

Abwesend: Jan Wiesemann, Vanessa Dering

Top 1: Weekly Scrum

Name	Erledigte Aufgabe
Daniel	Erstellung der eER-Modell aus Daten und Elementen, die aus den Stage-Beschreibungen extrahiert wurden. Beschäftigung mit Hibernate
Dirk	Befassung mit dem Bewertungssystem mit möglichen Funktionen und Bestandteilen Beschäftigung mit dem benötigten Feldern vom Login und der Registration Erstellung der GUI-Login (benötigte Felder)
Jan F.	Erstellung der GUI-Login (benötigte Felder) Befassung mit dem Bewertungssystem mit möglichen Funktionen und Bestandteilen Auswertung der Inhalte aus der Diskussion für das Bewertungsssystem
Sebastian	Erstellung der GUI-Login (benötigte Felder) Befassung mit dem Bewertungssystem mit möglichen Funktionen und Bestandteilen
Patrick	Bearbeitung an den Rechten und Aktivitäten von Stages
Holger	Bearbeitung am eER-Modell Erstellung des Login und Registrierungsdesign
Artjom	Bearbeitung am eER-Modell Erstellung des Login und Registrierungsdesign
Jessica	Bearbeitung an den Rechten und Aktivitäten von Stages
Yaping	Erstellung des Login und Registrierungsdesign Einarbeitung mit der Login-Erstellung im Eclipse
Tobias	Beschäftigung mit dem benötigten Feldern vom Login und der Registration Befassung mit dem Bewertungssystem mit möglichen Funktionen und Bestandteilen

Top 2: Abstimmungen und organisatorische Themen

- Programmierwoche Termine: 21.09.2015 - 25.09.2015 und 05.10.2015 - 09.10.2015
- Bewertungssystem: alle einverstanden mit dem 5-Sterne "Amazon-Bewertungssystem". Die Bewertungsgruppe entwickelt ein neues Bewertungsssystem auf Basis von diesem 5-Sterne-Bewertungssystem.
- Wegen der Blockveranstaltung Business Intelligence 2 fällt das Treffen am KW37 aus.

Top 3: Aufgaben zum nächsten Termin

- Für alle: weiter an eigenen Tasks arbeiten

Top 4: Sonstiges

- **Kommenden Mittwoch um 10:00 im A4 2-201 treffen, um nächstes Sprint zu planen**

A.7.18. Protokoll vom 22.08.2015

Protokoll_150822

Datum: 22.08.2015

Moderation: Yaping Lian

Protokollant: Jessica Schulte

Abwesend: Vanessa Dering Jan Fischer Jan Wiesemann

Top 1: Weekly Scrum

Name	Erledigte Aufgabe
Daniel	Datenmodell erstellt (aus ER Diagramm ein relationales Datenmodell) Datenmodell in Projekt integriert
Dirk	Registrierung implementiert Möglichkeiten für den Wechsel zwischen den Views herausgearbeitet
Sebastian	Möglichkeiten für den Wechsel zwischen den Views herausgearbeitet Navigator wegen MVP nicht möglich, CDIUIProvider
Patrick	Rollen und Rechtekonzept Umsetzungsmöglichkeiten in Vaadin erarbeitet Präsentation für das Userkonzept erstellt
Holger	Login Design erstellt
Artjom	Login Design erstellt Erstellungsmöglichkeiten von UIs (DeclarativeUI nicht verwenden, da zu wenig Doku) GUIs angepasst Unterschiede zwischen CSS und SCSS erarbeitet
Jessica	am Userkonzept gearbeitet/Möglichkeiten zur Umsetzung herausgesucht Präsentation für das Userkonzept erstellt
Yaping	Login Design erstellt
Tobias	Registrierung implementiert Sessionmanagement/ CDIUI Provider

Top 2: Abstimmungen und organisatorische Themen

- Login Design Gruppe:
    - myTheme.scss
    - standard Theme valo wird verwendet

- Registrierung Implementierung Gruppe
    - hat Ergebnis vorgestellt
    - Bedingungen für die Eingabefelder müssen noch angezeigt werden
- Userrollen:
    - weitere Rolle für Teammitglieder
    - wie Zusammenarbeitin den Stages ermöglichen, auch in der Vorhaben Stage soll auch Zusammenarbeit stattfinden
    - Zuordnung zu den Ressourcen (Idee, Vorhaben...)
    - LDAP?
- Datenmodell:
    - Nutzung des JPAContainers
    - persistence.xml enthälz Verbindungsdaten zum Server
- Technologietag am 11. September,
    - halten eines Vortrags á 25 min zweimal,
    - Prozess, Konzept und Vaadin vorstellen

Top 3: Aufgaben zum nächsten Termin

- Sebastian: TomEE auf Server Nutzung klären

Top 4: Anmerkungen/Sonstiges

- erstellten Tasks schätzen
- 4 Wochen Sprint

A.7.19. Protokoll vom 27.08.2015

Protokoll_150827

Datum: 27.08.2015

Moderation: Jessica Schulte

Protokollant: Patrick Smit

Abwesend: Sebastian Beckmann, Dirk Tesche, Vanessa Dering, Jan Fischer

Top 1: Weekly Scrum

- Artjom: Login/Registrierung, Übergabe der Views, Nutzer in der Datenbank
- Tobi: Login/Registrierung, Übergabe der Views, Nutzer in der Datenbank, Testfälle Eingabefelder
- Patrick: Präsentation Technologietag, Rolle Teammitglied
- Holger: Mockups Stages 1+2, View-Presenter-Model angelegt
- Yaping: MVP für Stage Vorhaben, Beispiele Dashboard Desgin (Mockup)
- Jessica: MVP Stage 1 + Dashboard, Mockup Dashboard

Top 2: Abstimmung und organisatorische Themen

- Vorstellung des Dashboard-Desgins
    - Anmerkungen: Neue Liste/Neue Probleme, Abonnierte Themen, TOP-Listen (Gamification Elemente berücksichtigen)
    - TOP-Liste als Heißluftballon sollte intensiver diskutiert werden
    - "Meine Bewertung" zum Mockup Probleme hinzufügen
    - Weiterleitung "Ideen zum Problem" anzeigen auf der Problem-Seite
- Vorstellung Präsentation Technologie-Tag
    - Ziel: 3 Punkt als 1. nehmen
    - Fazit: Hohe Einstiegshürde Vaadin
    - Letzte Seite: Kontakt
    - Grafik etwas kleiner
- Design Thinking /Feedback überlegen ob für ersten Phasen anwendbar

A.7.20. Protokoll vom 03.09.2015

Protokoll_150903

Datum: 03.09.2015

Moderation: Patrick Smit

Protokollant: Dirk Tesche

Abwesend: Vanessa Dering, Sebastian Beckmann, Daniel Martin Ahlers

Top 1: Weekly Scrum

Name	Thema
Tobi	Sessionmanagement, Inhalt Dashboard und weitere notwendige Views identifiziert, Entwicklung eines Konzeptes zur Zusammenarbeit auf der Plattform.
Dirk	Jenkins aufgesetzt, Inhalt Dashboard und weitere notwendige Views identifiziert, Entwicklung eines Konzeptes zur Zusammenarbeit auf der Plattform.
Jan F.	Sessionmanagement, User-Service.
Artjom	Design der Startseite und des Dashboard umgesetzt, Inhalt Dashboard und weitere notwendige Views identifiziert, Entwicklung eines Konzeptes zur Zusammenarbeit auf der Plattform.
Holger	Mockup Bewertung erstellt, Stage 3 gemäß Tasks angelegt (MVP), Entwicklung eines Konzeptes zur Zusammenarbeit auf der Plattform.
Jessica	Design des Dashboards umgesetzt, neue Elemente hinzugefügt.
Yaping	Design des Dashboards umgesetzt, neue Elemente hinzugefügt, Entwicklung eines Konzeptes zur Zusammenarbeit auf der Plattform.
Jan W.	Stage 4 gemäß Tasks angelegt (MVP), Entwicklung eines Konzeptes zur Zusammenarbeit auf der Plattform.
Patrick	Präsentation angepasst und fertiggestellt, an zusätzlicher notwendigen Berechtigungsrolle gearbeitet. Stage 5 - Feedback bearbeitet.

Top 2: Abstimmung und organisatorische Themen

- Vorstellung der bisherigen Arbeitsergebnisse:
    - Login und Registration wurden nicht vorgestellt, da diese bereits durch alle PG-Mitglieder
      aktiv bearbeitet wurden und daher bekannt waren.
    - Farbschema wurden im Rahmen dieses Treffen ebenfalls nicht erneut vorgestellt, da diese bekannt waren.

    - Stage 5 - Feedback
        - Feedback soll auch bei einem nicht erfolgreich umgesetzten B-C erfolgen. (Vgl. Power Point Präsentation)
- Scrum-Meeting
    - Nächstes Scrum-Meeting wird am Mittwoch den 16.09.15 um 10:00 Uhr durchgeführt.
    - Die Sprintdauer beträgt drei Wochen.

Top 3: Aufgaben zum nächsten Termin

- Patrick richtet die E-Mail : pg-impact@informatik.uni-oldenburg.de und die Umleitung auf pg-impact-intern@informatik.uni-oldenburg.de ein.
- Dirk bereitet das Scrum-Meeting vor und informiert alle PG-Teilnehmer mit einer zusätzlicher E-Mail über Zeitpunkt, Raum und Ziele.

A.7.21. Protokoll vom 17.09.2015

Protokoll_150917

Datum: 17.09.2015

Moderation: Dirk Tesche

Protokollant: Jan Wiesemann

Abwesend: Vanessa Dering, Sebastian Beckmann, Patrick Smit, Artjom Baranow

Top 1: Weekly Scrum

- Aufgrund BI2-Seminar ausgefallen

Top 2: Abstimmung und organisatorische Themen

- Kurzer Bericht zum Lufthansa Technologietag von Daniel
    - Es wurde angefragt, ob das Projekt als Startup weitergeführt werden soll
- Dokumentation
    - Einzelne Kapitel der Dokumentation werden nun per "input" ausgelagert, um Konflikte beim gemeinsamen Arbeiten zu vermeiden
    - Eine Sprint-Dokumentation wird eingeführt
- Hibernate Envers
    - Daniel schlägt eine Versionierung der Datenbankinhalte mittels Hibernate Envers vor
    - Minimalbeispiel wurde durch Daniel erstellt
    - Erhöhtes Datenvolumen durch Log-Tabelle, aber keine Performanceeinbußen
    - Das Thema wird weiterverfolgt
- Sprint Planning
    - Es wurden die Tasks für den nächsten 3-wöchigen Sprint besprochen
    - Die Planung wurde per Magic Estimation durchgeführt

A.7.22. Protokoll vom 01.10.2015

Protokoll_151001

Datum: 01.10.15

Moderation: Dirk Tesche

Protokollant: Jan Fischer

Abwesend: Daniel Martin Ahlers, H. Eichholz, Jessica Schulte, Tobias Kromke, Yaping Lian

Top 01: Weekly Scrum

- Sebastian Beckmann: Userlist, Push-Notification, Chat
- Patrick Smit: Dokumentation, Bewerbungs-Entwürfe für CeBit
- Jan Fischer: Chat Window, Userlist, Push-Notification
- Vanessa Dering: Header ausgelagert, Stage 1 Desgin, View Change Wechsel, Suggestions Stage von Tobi durchgearbeitet
- Jan Wiesemann: Administrative Dinge, Quellcode kommentiert, (Bewerbung für CeBit)
- Dirk Tesche: Nutzerprofil bearbeiten, Datenbank auslesen, Werte ändern
- Artjom Baranow: Nutzerprofil bearbeiten, Themewechsel

Top 2: Abstimmungen und organisatorische Themen

- Bewerbung CeBit 2016
    - generell: saubere Adressierung, Sorgfältigkeit
    - Inhalte:
        - Themenbeschreibung -> von der Idee bis zum Businesscase
        - Innovationsaspekte des Projektes
        - Exponat: 2 PCs (Präsentation + Prototyp)
        - Bild als Vektorgrafik (als Poster)
        - Innovationsprozess durch Grafik beschreiben
        - Besonderheit der Technologie (Vaadin?)
        - Bild in der Exponatsbeschreibung: Bezug in der Beschreibung auf das Bild im Text davor
- Estimation Meeting: Donnerstag Besprechung, Freitag Schätzung, Samstag Sprint starten
- Verfügbarkeit während der nächsten Programmierwoche: Bitte in Liste eintragen
    - Jan Wiesemann organisiert ggf. einen Schlüssel
    - Dirk Tesche organisiert ggf. Getränke (Mate + Bier)

A.7.23. Protokoll vom 22.10.2015

Protokoll_151022

Datum: 22.10.2015

Uhrzeit: 12:30 bis 14:00

Moderation: Jan Fischer

Protokollant: Artjom Baranow

Abwesend: Patrick Smit

Top 1: Weekly Scrum

- Jan Fischer: Internes Meeting bezüglich allgemeinem Konzept, Chat weiter bearbeitet insbesondere im Bereich der Offline Messages, n:m Beziehung in der Datenbank
- Dirk Tesche: Internes Meeting bezüglich allgemeinem Konzept, User Profile Passwort ändern
- Artjom Baranow: Internes Meeting bezüglich allgemeinem Konzept, User Profile Passwort ändern und kleine Änderungen, JavaDoc User Profile
- Daniel Martin Ahlers: Internes Meeting bezüglich allgemeinem Konzept, Datenbank im Bereich Hibernate Envers
- Tobias Kromke: Internes Meeting bezüglich allgemeinem Konzept, Tags und Anpassung der n:m Beziehung in der Datenbank (bei n:m Beziehung noch Probleme bei Hibernate)
- Sebastian Beckmann: Internes Meeting bezüglich allgemeinem Konzept, Chat weiter bearbeitet insbesondere im Bereich der Offline Messages
- H. Eichholz: Internes Meeting bezüglich allgemeinem Konzept, kleine Änderungen am Design von Buttons
- Jessica Schulte: Tutorial View, Design Stage 3 (jetzt Projektstage)
- Vanessa Dering: Internes Meeting bezüglich allgemeinem Konzept, Beschäftigung mit dem Rollenkonzept
- Jan Wiesemann: Internes Meeting bezüglich allgemeinem Konzept, Beschäftigung mit dem Rollenkonzept, Dokumentation

Top 2: Abstimmungen und organisatorische Themen

- Rating (Einzelnes vs. mehrere Kriterien):
    - Sternesystem soll beibehalten werden
    - Algorithmus zur Abstufung bei mehreren Bewertungen
    - Kommentare sollen nicht bewertet werden
    - Eine Kategorie im allgemeinen, keine Unterkategorien
    - halbe Sterne mit Rundungsmöglichkeit sollen möglich sein
- Terminfindung wöchentliches Treffen für das laufende Semester:
    - Donnerstag 14-16 Uhr wird als regelmäßiges Treffen geplant
- Zusammenfassung der am Mittwoch erzielten Ergebnisse:
    - Activity Stream als zentrales Element im Dashboard
    - Teilorientierung von IMPACT an Facebook und eXo Platform
    - Stage 1 (Problem) und Stage 2 (Vorschlag) werden zu Challenge zusammengefasst (Vorteile zur Erweckung erster Anreize und Motivationen und Verbesserung der Usability),
      Stage 3 (Vorhaben) wird zu Projekt zusammengefasst
        - Linearer Ablauf der Stages wirkt dadurch nicht mehr so starr
        - Progressbar innerhalb der Challenge bis zum Projekt
        - Projekte bei ähnlicher Zielsetzung werden im Business Case priorisiert
        - Tags und Suchfunktion sollen zunächst reichen, um z.B. Challenges zu finden, die ähnlich sind bzw. ähnliche Lösungsvorschläge haben
- Anpassung der Meilensteinplanung, fixer Termine und Priorisierung der nächsten Schritte / Aufgaben:
    - Arbeitspakete wurden mit der Wichtigkeit von 1-3 priorisiert
    - Meilensteinplanung muss noch angefertigt werden
- Sprint Retrospektive:
    - ist in der Sprint Dokumentation festgehalten

Top 3: Aufgaben zum nächsten Termin

- Jan W. Erstellt Doodle Umfrage bezüglich der Treffen am Wochenende kümmert sich um weitere organisatorische Sachen wie Räume etc.
- Alle: Doodle Umfrage bezüglich der Treffen am Wochenende ausfüllen

Top 4: Anmerkungen / Sonstiges

- Gruppe ist insgesamt zufrieden, Vaadin als Technologie gewählt zu haben (wichtig für die Dokumentation)
- Bezüglich des E-Mail-Systems muss Hans Hermann angefragt werden, damit ein E-Mail-Server eingerichtet wird
- Interaktive Elemente, z.B. neueste Challenges, sollen besser als Komponenten eingebunden werden, damit nicht für alle ähnlichen Elementen eine neue
  eigene MVP-Triade angelegt werden muss (Wiederverwendbarkeit soll dadurch vereinfacht und optimiert werden)
- Backlog Grooming nach der Sprintplanung soll dauerhaft eingeführt werden
- Aktueller Sprint bekommt ein Refinement
- Sprintplanung soll in Zukunft nicht direkt nach dem Meeting, sondern an einem Extra-Termin stattfinden, solange es nicht nach dem Wochenende ist

A.7.24. Protokoll vom 29.10.2015

Protokoll_151029 / Activity Stream

Datum: 29.10.2015

Moderation: Daniel Martin Ahlers

Protokollant: Sebastian Beckmann

Abwesend: Vanessa Dering

Top 1: Weekly Scrum

- Dirk Tesche: Eintragen der Challenge angepasst, aktuell beim Auslesen der Challenge
- Jan Fischer: Chat/Userservice Javadocs erstellt, Filter fürs Auslesen der Datenbank angeschaut, Rating-Formel erstellt
- Jessica Schulte: View für Lösungsvorschläge,
- H. Eichholz: Notifications UI erstellt
- Jan Wiesemann: In Sterne Plugin eingearbeitet, Rating-Formel erstellt
- Patrick Smit: Error-Handling
- Artjom Baranow: Eintragen der Challenge angepasst, aktuell beim Auslesen der Challenge, Bewertungssystem MVP-Triade
- Tobias Kromke: Eintragen der Challenge angepasst, aktuell beim Auslesen der Challenge
- Sebastian Beckmann: Chat/Userservice Javadocs erstellt, Dashboard-Sample angeschaut und Datenstruktur für Notifications
- Daniel Martin Ahlers: Mit Domain-Model beschäftigt, Minimalbeispiel many-many

Top 2: Abstimmungen und organisatorische Themen

- Code-Cleaning-Day Termin muss gefunden werden.
- Rating-Formel: möglicherweise den Faktor Zeit mit einbeziehen (in die Konstante 12), vielleicht Glühbirnen anstatt Sternen
- Bewertungen werden nicht bewertet (hilfreich/nicht hilfreich)
- Neue Startzeit des Treffens: **14:15 Uhr**
- Kommentar kann Rating enthalten (muss aber nicht)
- Bewertungen können nachträglich angepasst werden
- Sprintplanung Mittwochs ab **12:30 Uhr** (alle 3 Wochen)

Activity Stream

- Zu beginn: Ein Feld mit: "Passen Sie ihren Activity Stream an!"
- Einstellungen zum Activity Stream im Profil
- Folgende Elemente sollen anzeigbar sein:
    - Erstellen
    - Bearbeiten
    - Kommentare
    - Bewerten
    - Beendet
    - Feedback
    - Neue Lösungsvorschläge
    - Rang
- Was gehört in eine Aktivität:
    - Wer
    - Wann (Uhrzeit Datum)
    - Wo
    - Was (->Elemente)
- Design wie Facebook
- Notifications:
    - zu Beginn abonnierte Projekte
    - Einstellbar

A.7.25. Protokoll vom 05.11.2015

Protokoll_151105

Datum: 05.11.2015
Moderation: Artjom Baranow
Protokollant: Tobias Kromke
Abwesend: Jessica Schulte

Top 1: Weekly Scrum

- Dirk: Challenges auslesen und als Liste darstellen + Sortierung mit Filter, überflüssigen Code aus kommentiert
- Jan W.: Testprojekt für IMPACT Rating angelegt Vorbereitung für Rating
- Holger: mit Notifications beschäftigt, PG-Raum umgestaltet
- Daniel: Tabelle für Challenges gestaltet (Daten auslesen + filtern), Minimalbeispiel für Problem mit Tags
- Vanessa: mit Filtern & Notifications beschäftigt, mit nested properties beschäftigt
- Patrick: mit Exceptionhandling befasst
- Tobias: Challenge in DB eintragen, Tags in DB eintragen vorbereitet (Fehler in DB war Problem), Challenges auslesen, PG-Raum umgestaltet
- Jan F.: Activity-Stream (Datenstruktur, Threadhandling, Modell gebaut, Events angelegt, Aktionen identifiziert)
- Sebastian: Activity-Stream (Datenstruktur, Threadhandling, Modell gebaut, Events angelegt, Aktionen identifiziert)
- Artjom: PG-Raum umgestaltet, Challenge auslesen, Tabelle füllen + filtern

Top 2: Abstimmungen und organisatorisches

- IMPACT-Rating:
    - Problematisch, da nicht ganz Transparent ist wie Rating zustanden kommt
    - Problematisch Ansatz gesamte Anzahl registrierter Nutzer zu nehmen, besser aktive Nutzer
    - Trennung Qualität und Popularität nicht vorhanden -> irreführend, da bei neuen Ideen nicht klar ist, ob Qualität schlecht ist, oder nur geringe Anzahl an Nutzern abgestimmt haben
    - Sterne werden als reine Qualitätsbewertung verstanden, deshalb Verwendung von IMPACT-Rating bei geringer Anzahl an Bewertungen kritisch

        - evtl. IMPACT-Rating erst anzeigen, wenn gewisse Anzahl von Bewertungen vorhanden ist
    - IMPACT-Rating erst in Abhängigkeit zur Zeit nutzen
        - Gefahr der Verschlechterung über einen gewissen Zeitraum
- Filtermöglichkeiten auf Popularität setzen
- Sprintplanung und Priorisierung der Aufgaben11.11.2015 12:30 Bibliothek Raum 3.2
- 2 Bildschirme behalten, Rest zurückgeben
- Programmierwochenende 21.-22.11 & 12-13.12
- Mitte Dezember den Betreuern einen Stand präsentieren, auf Grundlage dessen fehlende Kernfunktionen identifizieren welche noch umzusetzen sind (nach der 2. Programmierwoche)

A.7.26. Protokoll vom 12.11.2015

Protokoll_151112

Datum: 12.11.2015

Moderation: Sebastian Beckmann

Protokollant: H. Eichholz

Top 1: Weekly Scrum

- Alle: Scrummeeting am 11.11.2015 in Bibliotheksraum 3.2
    - Ziele definiert bis Dezember
    - Grobe Zeitplanung
- Jan F.: Aktivity Stream ist in der letzten Phase (u.a. fehlt noch die graphische Aufarbeitung); Überlegung für den Umbau auf Links
- Tobias: Auslesen der Challenges, Anpassung von Design
- Patrick: Testszenario zu den Links
- Artjom: ItemClickListener für die Challenges, nächste Schritte auslesen
- Dirk: siehe Artjom,
- Holger: Notification z.B. abonieren von Challenges, Views für die Übersicht von Notifications
- Daniel: Anpassung der Datenbank und Problembehebung, Beteiligung beim ItemClickListener, Fehlerlösung von Exception
- Jan W.: View für die Challenge, Dateidownload
- Vanessa: Filtersystem
- Jessica: Implimentierung der Lösungsvorschläge u.a. Anzeigen auf Editierview
- Sebastian: Aktivity Stream

Top 2: Abstimmungen und organisatorisches

<u>Sprint Retrospektive:</u>

- Positiv:
    - Raumsituation (gute Gestaltung)
    - Anwesendheit (fast immer jemand da)
    - Gute Fortschritte
- Negativ:
    - Mergeprobleme
    - Task nur sehr grob (genauere Untertasks)

    - Dokumentation kommt durch Fortschritte zu kurz
    - Kommunikation etwas eingeschlafen...

Hinweis: Die Entwicklung darf nicht auf dem Master-Branch stattfinden, sondern muss über einen eigenen Branch erfolgen.

Vorstellung der Ergebnisse:

- Keine Ergebnisse, die angezeigt werden müssen, da es nur kleine Änderungen gab

Dokumentation:

- Ca. 4-5 Wochen nach der Abgabe gibt es eine abschließende Präsentation!
- Gesamtnote für die Ausarbeitung und Abgabe
- Es gibt keine genaue Einteilung
- Benutzerhandbuch ist erforderlich, da es für eine größere Gruppe an Nutzer gedacht ist
- APA für die Weiterentwicklung ist nicht erforderlich

A.7.27. Protokoll vom 19.11.2015

Protokoll_151119

Datum: 19.11.2015

Moderation: H. Eichholz

Protokollant: Jessica Schulte

Abwesend: Vanessa Dering, Dirk Tesche

Top 1: Weekly Scrum

- Jessica: Suggestions speichern
- Tobi: Challenges editieren und speichern
- Artjom: Challenges editieren und speichern
- Jan F. : Vaadin Navigation mit URLs, Error Handler angepasst, Fehler-Debugging.
- Sebastian: Activity Stream fast fertig
- Patrick: Vaadin Navigation mit URLs
- Daniel: Tags mit Item verknüpfen und speichern
- Jan W.: Wizard angesehen, um Projekt zu dokumentieren
- Holger: Button mit Optionen zum abbonieren, Tabellen für Notifications

Top 2: Abstimmung und organisatorische Themen

- Programmierwochenende: 11 Uhr Weekly Scrum
- Sonntag frühstücken
- Jan organisiert Weihnachtsmarktbesuch (an einem Donnerstag)

A.7.28. Protokoll vom 26.11.2015

Protokoll_151126

Datum: 26.11.2015

Moderation: Jessica Schulte

Protokollant: Vanessa Dering

Abwesend: Daniel Martin Ahlers

Top 1: Weekly Scrum

- Sebastian: Activity Stream eingebunden, Wechsel der Views in der LobbyUI + WelcomeUI
- Jan F. : Wechsel der Views in der LobbyUI + WelcomeUI, Tags löschen, Auslesen von Tabelle (Notifications)
- Artjom: Bewertungen in die DB schreiben
- Patrick: Wechsel der Views in der LobbyUI + WelcomeUI, Gliederung für Dokumentation erstellt
- Holger: Notification an Challenges angebunden, Auslesen von Tabelle (Notifications)
- Jan W.: Wizard implementiert, TabSheet implementiert, Business Case View erstellt, Wechsel zwischen bearbeiten und anzeigen
- Vanessa: Wizard implementiert, TabSheet implementiert, Project View erstellt, Wechsel zwischen bearbeiten und anzeigen
- Dirk: Bewertungen in die DB schreiben, Tags löschen
- Tobias: Bewertungen in die DB schreiben, Tags löschen
- Jessica: Services fürs Rating der Lösungsvorschläge erstellt, Wechsel der Views

Top 2: Abstimmung und organisatorische Themen

- Cebit:
    - Patrick erstellt die englische und deutsche Beschreibung,
    - Anreise in Eigenregie
    - Evaluation auf der CeBit
- Evaluation Prototyp (Dezember):
    - Usability-Test des Prototypen soll umgesetzt werden, mit Tutorial
- Sprintplanung:
    - Mittwoch, 02.12.2015
    - Ideen für Anreizsysteme und Gamificationelemente sammeln

- Mergeday:
    - Sonntag, 13.12.2015
- Weihnachtsmarkt:
    - Donnerstag, 03.12.2015

A.7.29. Protokoll vom 03.12.2015

Protokoll_151203

Datum: 03.12.2015

Moderation: Vanessa Dering

Protokollant: Patrick Smit

Top 1: Weekly Scrum

- Sebastian: Activity Stream (navigateTo)
- Artjom: Rating, Durchschnittsbewertung, eintragen von Objekten in Datenbank, SCSS aufgeräumt
- Patrick: CeBIT Texte, Aufbau Dokumentation
- Jan F: Activity Stream (Navigation), Chat
- Holger: SCSS (Loginseite)
- Jan: Daten in Datenbank (Business Case)
- Jessica: Rating, Bewertung in View laden
- Dirk: Rating, Bewertungen
- Daniel: Absturz Datenbanken (CDI), Datenbankabfragen erst bei Menüaufruf
- Tobi: Rating, Bugfixing, Tags
- Vanessa: Service

Top 2: Abstimmung und organisatorische Themen

- CeBIT:
    - Änderungen in aktuelle Version einpflegen
- Bewertungen der Mitarbeit

Top 3: Aufgaben zum nächsten Mal

- Abstimmung Stundenzettel (dagegen gestimmt)
    - Vorschlag eigene Einschätzung zu Beginn
    - Sprint Planung auf Donnerstag verlegen
- Feedback - Konzept Dokumentation

A.7.30. Protokoll vom 10.12.2015

Protokoll_151210

Datum: 10.12.1987

Moderation: Patrick Smit

Protokollant: Jan Wiesemann

Abwesend: Jens Siewert, Joachim Kurzhöfer

Top 1: Weekly Scrum

- Dirk: Speichern des Ratings umgesetzt, Loesungsvoschlaege integrieren
- Vanessa: TransitionWizard als Uebergang von Challenge zu Projekt erstellt, Teammitglieder zu Projekt hinzufuegen umgesetzt, Arbeitsschritte zu Projekt hinzufuegen, Merge
- Artjom: in Design eingearbeitet, SCSS Struktur ueberarbeitet (Dashboardgestaltung), Performancesteigerungen
- Tobias: Rating speichern und editieren umgesetzt, Support und KnowHow Verteilung
- Sebastian: Navigator umgeschrieben (Problem: Challengeuebersicht wurde beim Oeffnen einer Challenge aus dem Activity Stream geladen), Bugfixing und optische Anpassungen des Chat
- Jan F.: Activity Stream, Vorraussetzungen fuer Notifications geschaffen (Problem: zu viele Branches), Support und KnowHow Verteilung
- Holger: Design, SCSS, Startseite, Registrierung und Passwortrecovery angepasst, Usability, Description Feld
- Daniel: Optimierung der Datenbank, Ausfuehrung der Datenbankoperationen (z.B. JOINS) erst dann, wenn sie tatsaechlich benoetigt werden
- Jan W.: Speichern, AUslesen und Editiere von BusinessCase und Feedback, Zusammenwirken und Uebergaenge der unterschiedlichen Stages, Merge
- Jessica: Bewertung in Loesungsvorschlaegen anzeigen (Problem: Bewertung wird verspeatet angezeigt, Auslesen aus DB dauert lange), Merge, Challenge mit Loesungsvorschlag verknuepfen
- Patrick: CeBIT Texte, Prozessmodell fuer Grafiken aktualisieren, Dashboard Design

Top 2: Abstimmung und organisatorische Themen

- CeBIT 2016
  - positive Rueckmeldung zu Texten erhalten
  - naechster Schritt: Grafiken erstellen
- MKWI 2016
  - Wissenschaftliche Veroeffentlichung auf Basis der Projektdokumentation
  - Fokus: Nicht technische Loesung vorstellen, Gedanke dahinter, Warum Forschungsbedarf? Loesungsansatz vorstellen, Warum?, Plaene fuer die Zukunft
  - Paper mit ca. 12 Seiten (deutsch)
  - Bis ca. Nov 2016
- Testkonzept
  - Dirk entwickelt Testkonzept bis Januar
  - Einsatz von Selenium (Testumgebung für Webanwendungen) geplant

A.7.31. Protokoll vom 17.12.2015

Protokoll_151217

Datum: 17.12.2015

Moderation: Tobias Kromke

Protokollant: Dirk Tesche

Top 1: Weekly Scrum

- Entfällt

Top 2: Vorstellung des Prototypen

- Die Präsentation des Prototypen beginnt mit einer kurzen Einführung in den aktuellen Fortschrittsgrades durch Tobi.
  Insbesondere was bereits umgesetzt wurde und was gerade aktuell bearbeitet wird. Im weiteren Verlauf werden die zukünftig geplanten Funktionen vorgestellt.
- Vorstellung des Prototyp:
  Die einzelnen Funktionen wurden vorgestellt.
  - Anmerkungen vgl. Protokoll "Vorstellung des Prototypen"

Top 3: Retrospektive

- Vergleiche Protokoll Retrospektive vom 17.12.15

A.7.32. Protokoll vom 07.01.2016

Protokoll_160107

Datum: 07.01.2016

Moderation: Jan Wiesemann

Protokollant: Daniel Martin Ahlers

Abwesend: Jan Fischer, Vanessa Dering, Sebastian Beckmann

Top 1: Weekly Scrum

- Dirk: Chapter, Gliederung Doku, Code, Usability Tests, Fragebogen
- Tobias: Chapter, Gliederung Doku, Code, Usability Tests, Fragebogen
- Artjom: Chapter, Gliederung Doku, Code, Usability Tests, Fragebogen
- Patrick: Abstimmung CeBit
- Daniel Gliederung Doku
- Holger: Codekommentierung
- Jessica: Codekommentierung, TOP 10 Implementierung
- Jan Wiesmann: Codekommentierung

Top 2: Terminplanung

- Ostern für Urlaub wird in Frage gestellt
- Abschlusspräsi: Alle wichtigen Leute einladen, aus Abteilung, Joachim
- Programmierwoche im Februar/März - Wenn möglich komplette Raumbuchung für 2 Monate
- CeBIT Planung: Information über benötigte Peripherie muss bis 26.1. feststehen, Texte fertig, Grafiken erstellen steht noch aus
- Wichtige Info von Stefan Wunderlich: Literaturverzeichnis ist nicht Teil der Arbeit, sondern mit eigener Section im Anhang

Top 3: Sprintplanung

- Stand: Prototyp wurde vorgestellt
- Auflistung der "Kundenwünsche" befinden sich im GIT
    - Vorstellung von Dirk
- Projekt/BusinesCase und Berechtigungssystem haben Priorität
    - Projekt: Daten sammeln, nach und nach, dann, bei ausreichend Daten, Übergang zum BC
    - BusinessCase Planung aus Basis von Arbeit von Sebastian
    - Projekt: Geführter Ablauf: Was soll an dieser Stelle stattfinden
- Up und Download von Dateien Sekundär
- Kommentare usw. Tertiär
- Auslesen aller Bewertungen muss funktionieren -> Sekundär
- Designgruppe bleibt erhalten, Anhand von Feedback sollen auch Inkonsistenzen entfernt werden

A.7.33. Protokoll vom 14.01.2016

Protokoll_160114

Datum: 14.01.2016

Moderation: Daniel Martin Ahlers

Protokollant: Artjom Baranow

Abwesend: Jan Fischer, Jan Wiesemann, Jessica Schulte (alle entschuldigt)

Tagesordnung

- Top 0: Sitzungseröffnung und Begrüßung
- Top 1: Weekly Scrum
- Top 2: Abstimmungen und organisatorische Themen
- Top 3: Aufgaben zum nächsten Termin
- Top 4: Anmerkungen / Sonstiges

Top 1: Weekly Scrum

- Dirk: Usability-Test Durchführung und Evaluation, Erstellung vorläufiger Ablaufpläne für zukünftige Sprints
- Tobias: Usability-Test Durchführung und Evaluation
- Patrick: Cebit Sachen weiter bearbeitet und abgeschickt, Design-Sachen im Dashboard, Übergang vom Businesscase zum Feedback
- Daniel: Beschäftigung mit dem Berechtigungssystem aus technischer Sicht und geschaut, inwiefern Vaadin dabei Hilfestellung bietet
    - angefangen, Minimalbeispiel in Eclipse zu erstellen
- Holger: Design Guidelines und Richtlinien, die im Bereich des Designs angepasst und umgesetzt werden können
- Vanessa: Projekt-Felder bearbeitet und aufgeschlüsselt, Übergänge vom Projekt und BusinessCase konzeptionell ausgearbeitet
- Sebastian: sich darum gekümmert, dass der TomEE Server wieder funktioniert, Übergägnge vom Projekt zum BusinessCase konzeptionell ausgearbeitet
- Artjom: Usability-Test Durchführung und Evaluation, Projekt: Felder bearbeitet und aufgeschlüsselt

Top 2: Abstimmungen und organisatorisches

- Übersicht/ Vorstellung von zukünftigen Sprints/Aufgaben:
    - Sprint 10 und 11 sollen in 3-Wochen und Sprint 12 in 2 Wochen eingeteilt werden
- Vorstellung Konzept zum Design:
    - Unterscheidung zwischen Hauptbutton und Standardbutton beachten
    - Mouse-Over Effekte bei nicht eindeutigen Elementen verwenden
    - Eindeutige Bezeichnungen bei den Eingabfeldern verwenden
    - Kennzeichnung, wobei es sich um Pflichtfelder handelt
    - Vertical-Layout bei Miniaturansicht und HorizontalLayout bei größerer Ansicht benutzen
    - Placeholder bei nicht eindeutigen Feldern verwenden
    - Klickbare Elemente kennzeichnen, z.B. durch Änderung des Mauszeigers oder durch Einfärben von Flächen etc.
    - Gleiche Begriffe für gleiche Einträge verwenden, z.B. Tag vs. Schlagwort
    - Icons zu Menüpunkten und Buttons müssen aussagekräftig sein
- Vorstellung Konzept zur Archivierung:
    - wird in der darauffolgenden Woche von Jessica vorgestellt
- Vorstellung Konzept zu Projekt -> Business Case -> Feedback:
    - Projekt :
        - Statusbearbeitung, Darstellung z.B. mit Hilfe von Gamification Elementen
        - Teilt sich in Ressourcen, Tätigkeiten, Gewinne und Einsparungen und Anschaffungen auf
        - Definition von Zielen, um strategische Annahmen zu treffen, die wichtig für den BC sind
        - Nach Erfüllung einer Checkliste kommt der Übergang zum BC
        - Einbindung eines kollaborativen Raums, um Kreativität zu fördern und Klickpfade zu minimieren
    - Business Case:
        - Statusbearbeitung, ähnlich wie beim Projekt
        - Komplexe Berechnungen im Bereich Finanzkennzahlen etc. und Analysen fallen weg, erfolgen stattdessen in anderen Bereichen der Lufthansa Industry Solutions

- Vorstellung der Ergebnisse des Usability-Tests:
    - Ergebnisse:
        - Allgemein
            - Mouse-over-Effekte
            - Anordnung der Elemente
        - Registration und Login
            - Eingabemodalitäten, speziell beim Passwort
        - Challenge und Lösungsvorschläge
            - Clickbarkeit der Listen
            - Feedback nach dem anlegen
            - Kennzeichnung der Editierbarkeit
            - Inkosistender Aufbau
- Bestimmung des neuen Projektleiters:
    - Holger erklärt sich bereit, die letzte Phase des Projekts als Projektleiter zu übernehmen
- Raumbuchung:
    - A02 2-203
    - 01.02. - 31.03.2016
    - Nutzung am Wochenende: Schluessel vom Hausmeister, Chip vom Infopoint
- CeBIT 2016 - Vorstellung Grafiktafel/Einzelseite + Aussteller-/Fachbesuchertickets:
    - Patrick kümmert sich darum, die 300 Tickets (gedruckt) für die VLBA zu besorgen
    - 2 Personen sollen pro Tag anwesend sein, beim Auf- und Abbautag ggf. mehr

Top 4: Anmerkungen / Sonstiges

- Dokumentationsinhalte, die abgegeben werden müssen:
    - Softwaredokumentation
    - Benutzerhandbuch
    - Installationshandbuch
    - Abschlussbericht

A.7.34. Protokoll vom 21.01.2016

Protokoll_160121

Datum: 21.01.2016

Moderation: Artjom Baranow

Protokollant: Jan Fischer

Abwesend: Patrick Smit

Top 1: Weekly Scrum

Person	Tätigkeit
Jan Fischer	Upload / Print / Download UI im Servertime Example angelegt, Gedanken zu DMS gemacht,
Sebastian Beckmann	Business Case: Stackpanel / Tabs Recherche, im Projekt eingefügt
Tobias Kromke	Usability Anpassungen (Terminologie) (mit Dirk / Artjom), Umgestaltung der Challenge Erstellung
Dirk Tesche	Usability Tasks Anpassung (siehe Tobi / Artjom), Übersichten angepasst,
H. Eichholz	Umfrage (Doodle), Dateiupload Addon bei Vaadin angeschaut, Doku Inhaltsverzeichnis + Kapitel
Vanessa Dering	Businesscase Anordnung, Dokumentation Benutzerhandbuch Gliederung
Jan Wiesemann	Konzept Prüfung Projekt zu BC
Artjom Baranow	Usability Tasks Anpassung (siehe Tobi / Dirk), Feedbacks verschlankt, SCSS angepasst, Redundanzen beseitig
Jessica Schulte	TopTen List erstellt ind Challenge Overview implementiert, TopTen Lösungsvorschläge, Konzept für Archivierung
Daniel Martin Ahlers	Berechtigungssystem (Shiro), Entwickelt bei Minimalbeispiel, Problemlösung Rating doppelt
Patrick Smit	Dokugestaltung

Top 2: Abstimmungen und organisatorische Themen

- Cebit:
    - Anwesenheitslist / Dienstpläne
        - Montag: Tobias / Vanessa
        - Dienstag: Dirk / Artjom
        - Mittwoch: Holger / Daniel
        - Donnerstag: Jessica / Patrick
        - Freitag: Jan / Sebastian
    - Patrick Smit: Bitte Fr. Franke nach Reisekostenerstattung fragen, wie groß ist der wird der Monitor sein, was ist mit den Parktickets?
    - Inhalte zur Darstellung klären: Schaubilder / Video / Demo Access der Software?
        - Produktivideo über die Software / Prozessdiagramm
    - Interviewleitfaden für Standbesucher
        - Unternehmenskontext festhalten
- Dokumentation
    - Gliederung überarbeiten
- Vorstellung Konzept zur Archivierung
    - Archivierung möglich wenn:
        - Zeitraum: 4 Wochen nach letzter Bearbeitung
        - Bewertung: bei mehr als 5 Berwertung und 2 Sternen
    - Archivierung möglich durch
        - Challenge: Ersteller
        - BC / Projekt: durch den Ersteller / Verantwortlicher
    - Archivierung müssen rückgängig gemacht werden können
- Stackpanel oder Tabs
    - Gruppe hat sich für Stackpanel (einstimmig) entschieden

A.7.35. Protokoll vom 28.01.2016

Protokoll_160128

Datum: 28.01.2016

Moderation: Jan Fischer

Protokollant: Sebastian Beckmann

Abwesend: Vanessa Dering, Artjom Baranow

Top 1: Weekly Scrum

Person	Tätigkeit
Sebastian Beckmann	Stackpanel in BC-Übersicht einpflegt
Jan Fischer	DMS-Service, Upload-Service
Dirk Tesche	Änderungen an Challenge und Lösungsvorschlägen (auf Basis der Umfrage), Dokumentation (Fortschrittsbericht)
Tobias Kromke	Änderungen an Challenge und Lösungsvorschlägen (auf Basis der Umfrage)
Jessica Schulte	Top-10 angeschlossen, Archivierungskonzept angepasst
Jan Wiesemann	Dokukonventionen, BusinessCase GUI
Patrick Smit	gelernt, cebit emails
Daniel Martin Ahlers	Berechtigungssystem (Minimalbeispiel fertig, wird nun ins Projekt überführt)
H. Eichholz	Dokumentation (Workshops, Anforderungsanalyse)

Top 02: Abstimmungen und organisatorische Themen

- Aufgabe "Konzept zur Mitarbeitermotivation erstellen": Holger, Patrick (, Jan) . Konzept bis nächste Woche Donnerstag erarbeiten.
- Dokukonventionen im Git und Confluence
- Programmierwoche: Mo 15.02. - Fr 19.02. 11:00-16:00Uhr pflicht
- Programmierwochenende: erstmal optional
- Teambuilding: Mittwoch (04.02.16) -> muss nur noch geklärt werden, was gemacht wird
- In der Abschlussdokumentationsübersicht im Confluence Schlagworte für die jeweilgen Themen eintragen, falls einem etwas einfällt
- Usability-Studie um eine Woche nach hinten verschieben -> hierfür muss jeder an das Usability-Team herantreten und den jeweiligen Stand darlegen

A.7.36. Protokoll vom 04.02.2016

Protokoll_160204

Datum: 04.02.2016

Moderation: Sebastian Beckmann

Protokollant: Tobias Kromke

Abwesend: -

Top 1: Weekly Scrum

Dirk	**Challenge auf Stackpanel umgebaut, Struktur Lösungsvorschlag angepasst, Komponente für Lösungsvorschlag eingebaut, aktuell Ersetzung des alten Aufbaus durch neue Komponente in Bearbeitung**
Tobi	Challenge auf Stackpanel umgebaut, Struktur Lösungsvorschlag angepasst, Komponente für Lösungsvorschlag eingebaut, aktuell Ersetzung des alten Aufbaus durch neue Komponente in Bearbeitung
Jan F	Upload + Multi File Upload, Komponente gebaut zum Hochladen, Komponente zum temporären Speichern, Dokumentenmanagementsystem
Artjom	Challenge auf Stackpanel umgestellt, Usabillity: Anordnung der Elemente auf Grundlage der Ergebnisse GUI bearbeitet
Jan W	Dokukonventionen bearbeitet
Patrick	mit Responsive Design vom Menü beschäftigt, Medieneintrag für Cebit bearbeitet
Holger	Mitarbeitermotivation Konzept erstellt + Recherche aus Praxis, Seminararbeiten gelesen, umgeguckt
Vanessa	Stackpanel neu organisiert, mit Checkliste für Project begonnen, Benutzerhandbuch für Login begonnen, Sliderpanel
Jessica	Archivierungskonzept bearbeitet, Buttons angelegt + Logik zum Archivieren für Challenge, Projekt, Business Case
Daniel	Berechtigungssystem bearbeitet, Apache Chero? integriert + Testen
Sebastian	Sliderpanel

Top 2: Allgemein

- Dokukonventionen:
    - TODO Package in LaTeX verwenden -> binden die Jans ein (damit nicht zwischen Confluence und LaTeX Dokument gewechselt werden muss)
    - Peer Review dennoch durchführen
    - Jan F. legt Doku Git an
- Motivationskonzept:
    - grüne Punkte potentiell umsetzbar
    - Zukünftig Erfolgsstorys in der Plattform anzeigen
    - Erinnerungen in Activity Stream okay, Auszulesen wann das letzte mal angeguckt wird zu viel für unser System
    - Aufpassen das Plattform nicht zu "social media" wird
    - Inzentives für Mitarbeiter über Plattform steuern?
    - Unterschiedliche Handlungsempfehlungen für Motivation geben
- Dokumentation:
    - Screenshots des aktuellen Projektstandes an Dirk und Tobi senden zur Dokumentation des Projektfortschrittes

A.7.37. Protokoll vom 16.02.2016

Protokoll_160211

Datum: 11.2.2016

Moderation: Tobias Kromke

Protokollant: H. Eichholz

Abwesend: Jan Wiesemann, Joachim Kurzhöfer, Jens Siewert

Top 1: Weekly Scrum

Person	Tätigkeit
Dirk Tesche	▪ Komponente für Lösungsvorschläge ▪ MVP-Triade für Kommentare
Artjom Baranow	▪ Anpassungen des Userprofil ▪ MVP-Triade für Wunschliste ▪ Minimalbeispiel für den UI-Wechsel
Jan Fischer	▪ Dokumentenmangement implementiert/angepasst
Patrick Smit	▪ Umbau des Menü auf responsive Design
Daniel Martin Ahlers	▪ Apache Giro ▪ Workaround implimentiert
H. Eichholz	• Umstellung der Checkliste • Aufbau auf Einfahren der Checkliste
Jessica Schulte	• Implementierung des Archivierungskonzepts
Vanessa Dering	• Komponente GRID beschäftigt und Implimentierung • Anpassung der Anzeige Projekts im Header • Zeichnungen für Hintergrundbilder
Sebastian Beckmann	• Business Case-Übersicht erstellt • Auslesen aus der DB umgesetzt • Dokugit angelegt
Tobias Kromke	• Komponente für Lösungsvorschläge • MVP-Triade für Kommentare

Top 02: Abstimmungen und organisatorische Themen

- Kapitel für die technologische Evolotion
- Bearbeitung der Dokumentation erfolgt in der Zukunft ausschließlich im Dokugit!
- Hinweis von Stefan, dass der Stil des Dokuments einheitlich sein soll (z.B. Bilder in schwarz/weiß...)

Kategorisierung Challenge/Lösungsvorschläge:

- Thema vorschoben, weil Joachim nicht da

Cebit:

- Fahrzeug mit Privatwagen und Abrechnung der Kosten über die Uni (Antrag kommt von Frau Franke)
- Monitor: Jeder guckt mal nach einem Monitor (bis Sonntag)
- Dienstreiseanträge müssen gestellt werden, dazu bekommt jeder einen Antrag von Patrick

Sonstiges:

- Erinnerung an Screenshots für Änderungen an Tobias und Dirk
- Vorstellung der Graphiken durch Vanessa und weitere Umsetzung durch Vanessa

A.7.38. Protokoll vom 18.02.2016

Protokoll_160218

Datum: 18.02.2016

Moderation: H. Eichholz

Protokollant: Jessica Schulte

Top 2: Abstimmungen und organisatorische Themen

- Text/Interweg für Pressestelle
    - Vanessa und Jan lassen sich interviewen
- Challenge umgebaut
- Kommentare erstellt
- Profilübersicht: Profile von anderen können angesehen werden
- Download und Upload für Challenge eingebaut (Viele DB Methoden geändert und muss noch in andere Sachen eingebaut werden)
- Checkliste im Projekt eingebaut
- Business Case fertig (Checkliste, Upload-Komponente)
- Berechtigungssystem: Fehler mit Sessions behoben, Berechtigungsmatrix eingebaut
- Projekt: Übergang zum Business Case
- Challenge: Übergang zum Projekt wird gebastelt
- Archivierung: ist fertig
- Activity Stream muss angepasst werden
- Alle Views müssen aufgrund des Berechtigungssystems umgebaut werden

A.7.39. Protokoll vom 25.02.2016

Protokoll_160225

Datum: 25.02.2016

Moderation: Jessica Schulte

Protokollant: Vanessa Dering

Top 1: Weekly Scrum

Name	Aufgabe
Holger	Berechtigungskonzept erstellt,dokumentiert und implementiert, fertiggestellt der View für Gewinne und Verluste
Tobias	Challenge und Lösungsvorschlag abgeschlossen, Kommentare werden in Projektansicht eingebaut werden, "Aufräumen" des Quellcodes, Profil fertig gestellt
Dirk	Challenge und Lösungsvorschlag abgeschlossen, Kommentare werden in Projektansicht eingebaut werden, "Aufräumen" des Quellcodes, Profil fertig gestellt
Daniel	Berechtigungskonzept fertig gestellt, dokumentiert und muss noch implementiert werden, Dokumentation bearbeitet
Jessica	Übergang Challenge -> Project weitestgehenst fertig gestellt
Vanessa	Kommentierung des Quellcodes, Bearbeitung von Anschaffungen
Sebastian	BC Dateiupload fertig gestellt
Patrick	Profilübersicht integriert,
Jan	kollab. Raum getestet und erweitert
Artjom	Challenge und Lösungsvorschlag abgeschlossen, Kommentare werden in Projektansicht eingebaut werden, "Aufräumen" des Quellcodes, Profil fertig gestellt, Themewechsel fertiggestellt

A.7.40. Protokoll vom 03.03.2016

Protokoll_160303

Datum: 03.03.2016

Moderation: H. Eichholz

Protokollant: Patrick Smit

Abwesend: Vanessa Dering, Jessica Schulte, Dirk Tesche, Jan Wiesemann

Top 2:

- Feedback fällt am 31.3 aus

Top 3: Cebit

- Es werden 2 Notebooks + 1 Monitor über die CeBIT versucht zu versichern (Klasse 1)
- Tobi schaut nach Schrauben für VESA-Halterung
- Fotos am ersten Tag machen für Projekttagebuch
- Fotos mit Ministern und Prof. Dr. Gomez

Top 4: Sonstiges

- Mittwoch großer Merge-Tag
- Produktvideos Donnerstag+Freitag
- Kapitel 1+2 grundlegend zur CeBIT abgeschlossen

A.8. Projektfortschritt

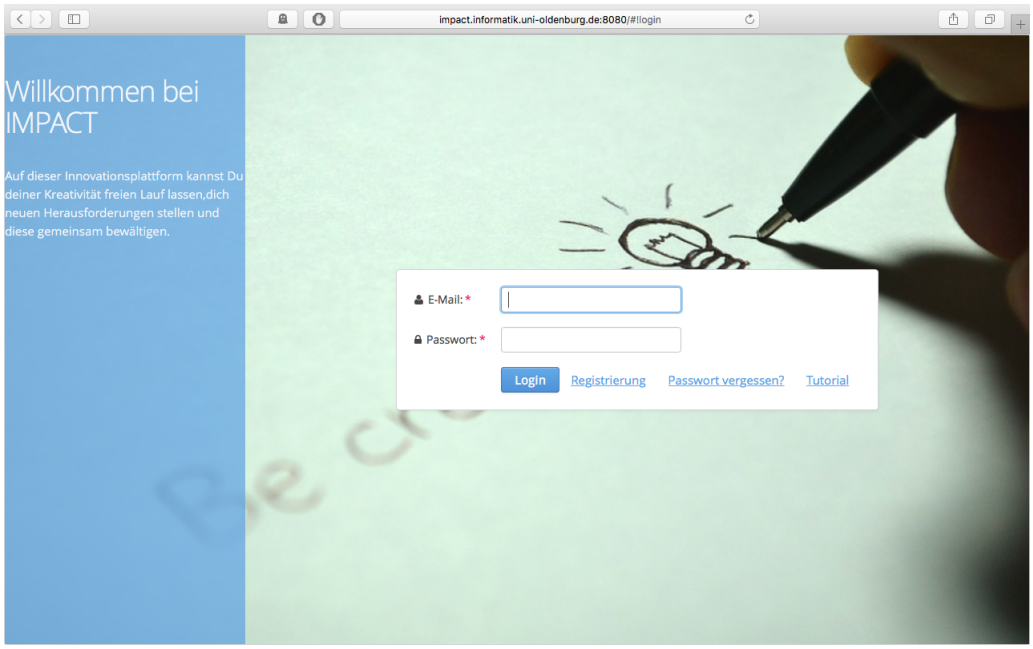


Abbildung 140: Sprint 9 - Stand Login

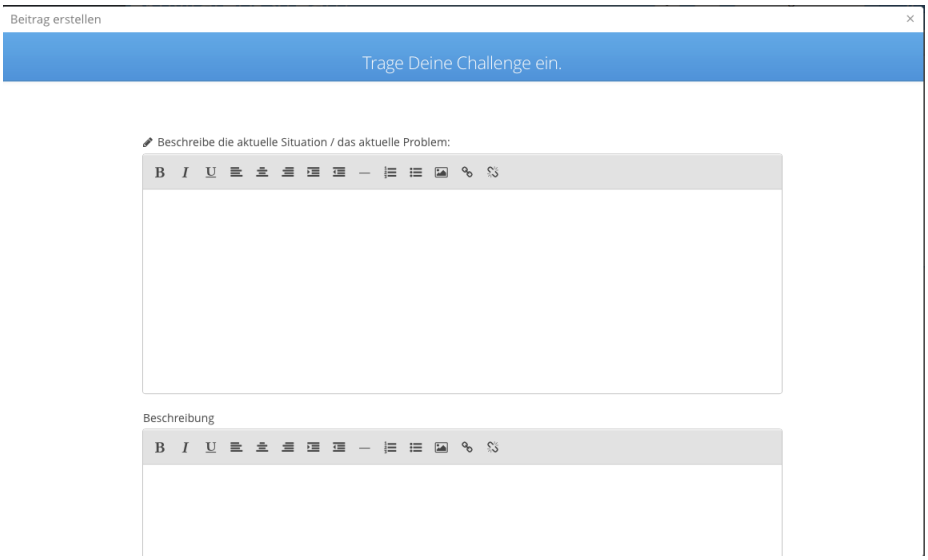


Abbildung 141: Sprint 9 - Stand Challenge Teil 1

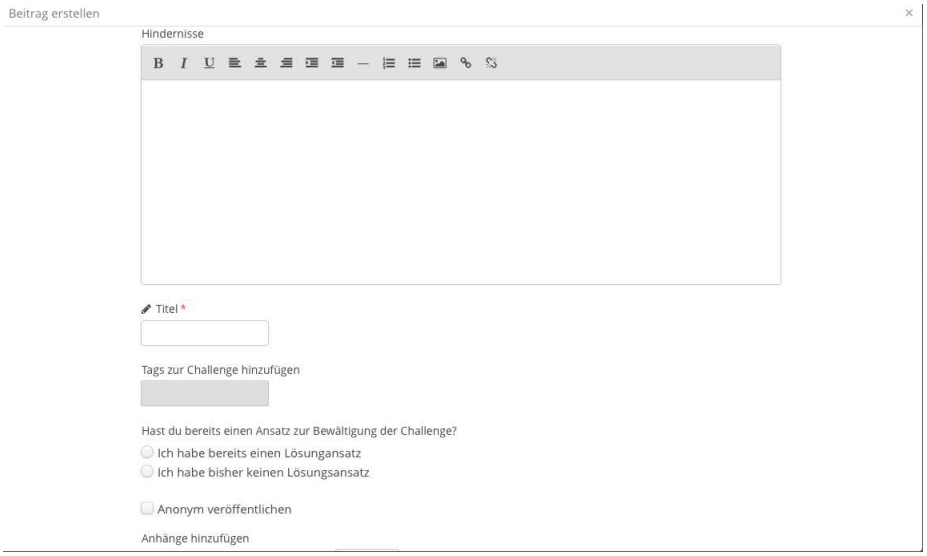


Abbildung 142: Sprint 9 - Stand Challenge Teil 2

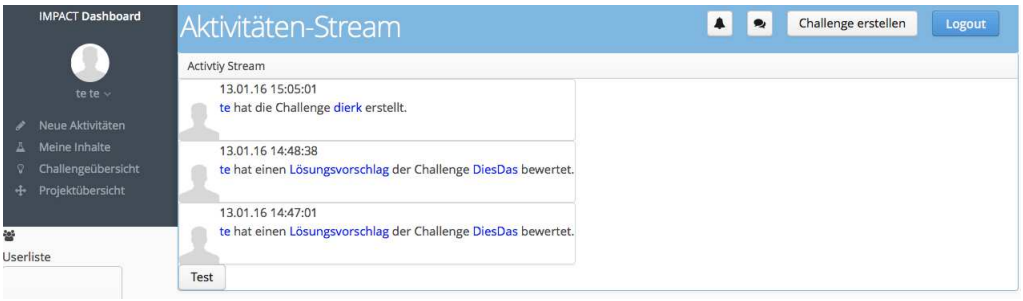


Abbildung 143: Sprint 9 - Stand Aktivitätenstream

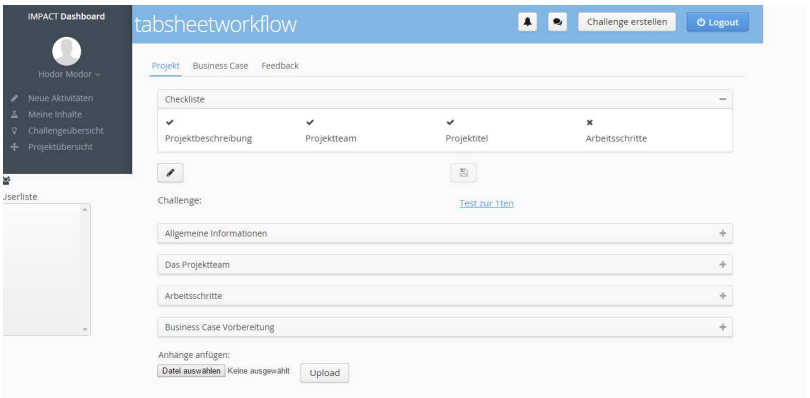


Abbildung 144: Sprint 9 - Stand Projekt Teil 1

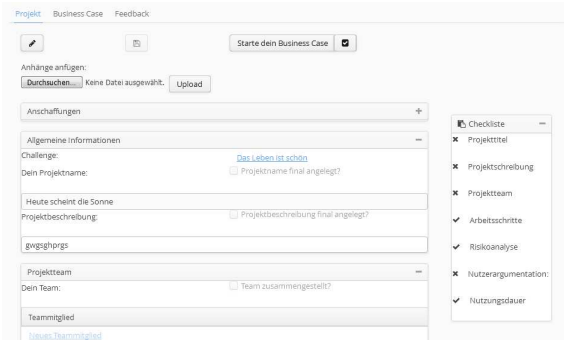


Abbildung 145: Sprint 9 - Stand Projekt Teil 2

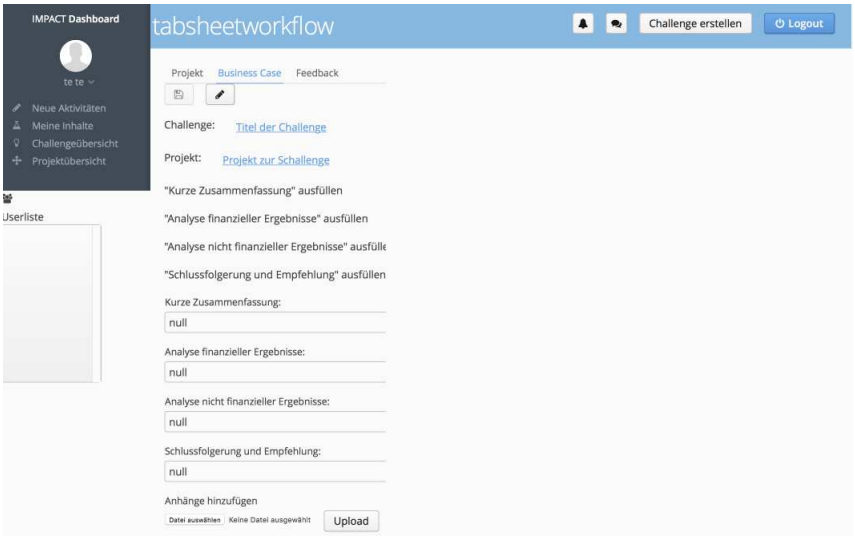


Abbildung 146: Sprint 9 - Stand Business Case Teil 1

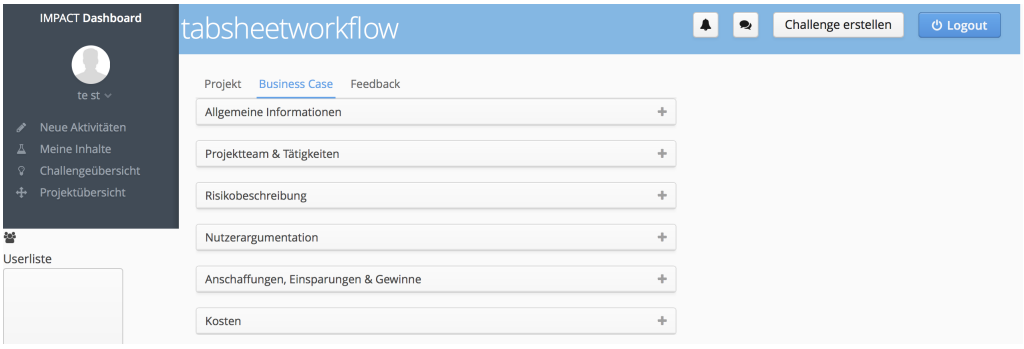


Abbildung 147: Sprint 9 - Stand Business Case Teil 2

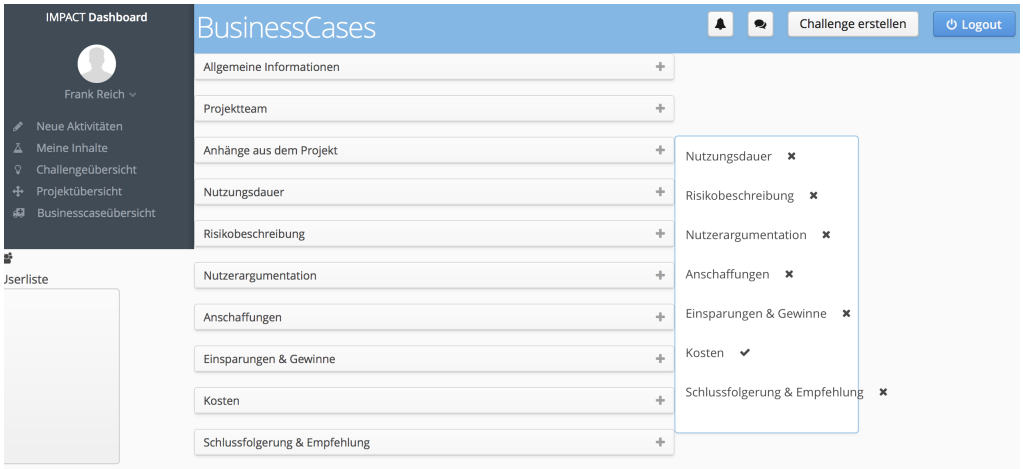


Abbildung 148: Sprint 9 - Stand Business Case Teil 3

Abschließende Erklärung

Wir versichern hiermit, dass wie unsere Dokumentation zur Projektgruppe *IMPACT* selbständig und ohne fremde Hilfe angefertigt haben, und dass wir alle von anderen Autoren wörtlich übernommenen Stellen wie auch die sich an die Gedankengänge anderer Autoren eng anlegenden Ausführungen meiner Arbeit besonders gekennzeichnet und die Quellen zitiert habe.

Oldenburg, den 08. April 2016

Daniel Martin Ahlers

Artjom Baranow

Sebastian Beckmann

Vanessa Dering

Holger Eichholz

Jan Fischer

Tobias Kromke

Jessica Schulte

Patrick Smit

Dirk Tesche

Jan Wiesemann

B. Anlagen

B.1. Seminararbeiten

B.1.1. Aufbau eines Business Case

Aufbau eines Business Case

Seminararbeit
im Rahmen der Projektgruppe IMPACT

Themensteller:	Prof. Dr.-Ing. Jorge Marx Gómez
Betreuer:	Dr. Joachim Kurzhöfer Dipl. Math. Jens Siewert M. Sc. Stefan Wunderlich
Vorgelegt von:	Sebastian Beckmann Franz-Poppe-Straße 19 26121 Oldenburg 0176 31262522 sebastian.beckmann@uni-oldenburg.de
Abgabetermin:	22. Juni 2015

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Im Rahmen der Projektgruppe IMPACT wird eine innerbetriebliche Innovationsplattform geschaffen, welche es erleichtern soll Ideen von Mitarbeiter innerhalb eines Unternehmens umzusetzen. In dieser Ausarbeitung wird nicht der eigentliche Prozess von der Idee hin zu einem konkreten Vorhaben betrachtet, sondern der Wirtschaftlichkeitsnachweis des durch die Plattform entstandenen Projektes. Dies ist ein wichtiger Bestandteil für die notwendige Bewertung, da Projekte einmalige Vorhaben sind, zu denen es meist keine Abschätzungen bezüglich ihrer Vorteilhaftigkeit gibt [Bru09]. Projekte werden stets nur vom Management durchgeführt, wenn sie wirtschaftlich von Vorteil sind. An diesem Punkt setzt der in dieser Ausarbeitung thematisierte Business Case an und versucht durch detaillierte Voraussagen und Analysen Aussagen über die Wirtschaftlichkeit eines Projektes zu treffen.

Ziel dieser Arbeit ist es zunächst die Grundlagen und den Aufbau eines Business Cases zu erläutert. Auf Basis dieser Erkenntnisse wird Schlussendlich eine Empfehlung ausgesprochen, wie ein Business Case in die Innovationsplattform integriert werden kann.

2 Grundlagen

In diesem Abschnitt der Seminararbeit werden die Grundlagen des Business Case dargestellt. So liefert Absatz 2.1 zuerst eine genauere Definition des Begriffs „Business Case“, in welcher Vorgehensweisen angeschnitten und Ziele aufgezeigt werden. Im nachfolgenden Abschnitt wird der Frage „Wozu wird ein Business Case benötigt?“ auf den Grund gegangen und Voraussetzung aufgezeigt, welche für die Erstellung notwendig sind. Danach werden die Vor- und Nachteile thematisiert und schlussendlich in Abschnitt 2.5 der thematisch verwandte Begriff „Business Plan“ erläutert und abgegrenzt.

2.1 Business Case

Ein Business Case ist ein Entscheidungsunterstützungsinstrument, welches zur betriebswirtschaftlichen Beurteilung einer Investition bzw. eines Projektes dient. Als inhaltlich passende Übersetzungen lassen sich Investitionsfolgenabschätzung oder Wirtschaftlichkeitsnachweis nennen. Folgende zentrale Frage versucht ein Business Case fundiert zu beantworten:

> „Welche finanziellen Konsequenzen entstehen, wenn eine (unternehmerische) Entscheidung so (und nicht anders) getroffen wird?“([Tas09], S. 15)

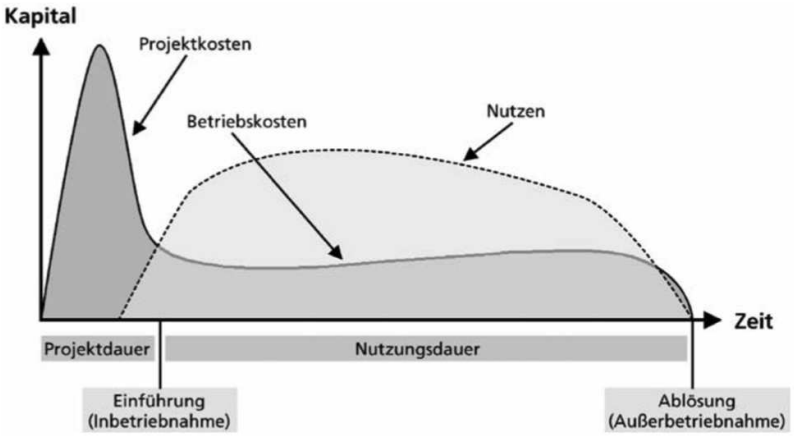


Abbildung 1: Vereinfachte Darstellung der Kosten und Nutzen eines Projektes [Bru09]

Ziel ist es schlussendlich finanziell vorteilhafte Vorhaben zu identifizieren und dem Management aufzuzeigen, welcher positive Gegenwert erwartet werden kann. Um ein Vorhaben als vorteilhaft einzustufen werden viele Faktoren zur Bestimmung mit einbezogen, welche in Abschnitt 3.1 genauer erläutert werden. Auch werden somit unvorteilhafte Projekte erkannt. Auf diese Weise ist das Management in der Lage, diese entweder zu verwerfen oder anzupassen und neu zu prüfen. Grundsätzlich gibt es keinen Standard zur Darstellung und Erstellung eines Business Case. Der Ersteller hat die freie Wahl den Business Case beispielsweise als Textdokument, als Tabellenkalkulation oder als multimediale Präsentation zu verfassen [Tas09]. Bei der Erstellung ist zu beachten, dass alle für das jeweilige Projekt relevanten Kosten und Nutzenaspekte vollständig erhoben bzw. bestimmt und dokumentiert werden. Abbildung 1 zeigt vereinfacht den Verlauf der Kosten und des Nutzen eines Vorhabens auf, welche in einem Business Case erfasst und gegenübergestellt werden. Durch Faktensammlungen und -analysen wird zu Beginn eine Basis erarbeitet, wodurch Annahmen hinsichtlich Kosten und Nutzen über den Projektzeitraums vorausgesagt werden können. Obwohl größtenteils wirtschaftliche Belange im Mittelpunkt stehen, werden auch nicht-monetäre Aspekte wie beispielsweise die Abwägung hinsichtlich Strategieorientierung berücksichtigt. Hierdurch wird eine Sicht auf das Gesamtprojekt dargestellt und somit dem Management eine Entscheidungsgrundlage geliefert [Bru09].

2.2 Wozu wird eine Business Case benötigt?

Im Mittelpunkt des unternehmerischen Denken und Handelns steht das wirtschaftliche Denken. Dies bedeutet für Vorhaben, dass sie vom Management nur angenommen und durchgeführt werden, falls diese für ein Unternehmen wirtschaftlich von vorteilhaft sind.

Da Projekte jedoch meist sehr vielschichtig und komplex sind, ist deren Gegenwert nicht direkt für die Entscheider erkennbar. Ein Business Case setzt an diesem Punkt an und versucht durch stichhaltige Annahmen und deren Gegenüberstellung den Gegenwert zu erheben. Hierbei gibt es zwei mögliche Investitionssituationen. Einerseits gibt es Durchführungsentscheidungen, bei welchen nur eine alternative Handlungsentscheidung der aktuellen Lage gegenübersteht. In diesem Fall liefert der Business Case eine klare Aussage darüber, ob das Vorhaben durchgeführt werden sollte oder nicht. Andererseits gibt es die Auswahlentscheidung. In dieser Situation stehen sich mehrere miteinander konkurrierende Handlungsalternativen gegenüber. Durch einen Business Case werden diese Alternativen analysiert und deren Wirtschaftlichkeit gegenübergestellt, sodass das Management in der Lage ist, die vorteilhaftere Lösung zu wählen. Falls der Business Case jedoch zeigt, das ein Vorhaben nicht rentabel ist, gibt es folgende drei Möglichkeiten:

- **Aussichtsreich:** Das Projekt könnte möglicherweise einen positiven Gegenwert liefern. In diesem Kontext sollte das Vorhaben unter Berücksichtigung der gewonnen Erkenntnisse überarbeitet und der Business Case wiederholt werden.
- **Aussichtslos (aber andere Gründe):** Das Vorhaben hat keinen wirtschaftlichen Vorteil, jedoch sprechen andere Gründe für die Realisierung dieses Projektes. So könnte beispielsweise das Projekt Voraussetzungen auf technologischer Basis für weitere Projekte schaffen. Diese Gründe müssen genau evaluiert werden, um zu erkennen, ob eine Realisierung des Projektes trotzdem lohnenswert ist.
- **Aussichtslos:** Das Vorhaben hat keinen wirtschaftlichen Vorteil und es sollte auch nicht aus einem anderen bestimmten Grund durchgeführt werden. In diesem Fall war die für den Business Case investierte Arbeit nicht umsonst, da so ca. 15- bis 20-mal Höhere Ausgaben, welche bei einer Durchführung einstanden wären, eingespart werden konnten [Bru09].

2.3 Voraussetzungen

Ein Business Case kann grundsätzlich für fast jedes Projekt erstellt werden, jedoch ist dies unter gewissen Gesichtspunkten nicht immer lohnenswert. Je genauer die folgenden Voraussetzungen gegeben sind, umso ratsamer bzw. effektiver ist die Erstellung eines Business Cases [Tas09]:

- Es muss mindestens zwei Handlungsalternativen geben zwischen denen einen Entscheidung getroffen werden muss, wobei eine Alternative auch die Beibehaltung des aktuellen Standes sein kann.

- Es steht nicht fest, welche Alternative gewählt werden soll. Diese bedeutet, dass beispielsweise vom Management nicht vorgeben werden darf, welches Ergebnis erzielt werden soll.
- Keine der beiden Handlungsalternativen lässt sich aufgrund ihrer Komplexität direkt als die Bessere identifizieren.
- Um Alternativen vergleichen zu können, lassen sich die Folgen der Entscheidungen vorwiegend in monetären Größen ausdrücken.

2.4 Vor- und Nachteile

Die Erstellung eines Business Case für eine Projekt bringt eine Reihe von Vorteilen und nur sehr wenige Nachteile mit sich. Nachteilig kann gesehen werden, dass ein Business Case viel Arbeit und Zeit zusätzlich zum eigentlichen Projekt erfordert. Dieses Argument wird jedoch hinfällig wenn man bedenkt, dass das Durchführen eines nicht rentablen Vorhabens das 15- bis 20-fache an Kosten verursachen würde. Es zeigt sich, dass je kostenintensiver die Auswirkungen eines Vorhaben sind, desto vorteilhafter ist es ein Business Case zu erstellen. Hinsichtlich der Entscheidungsfindung sind die ausschlaggebendsten Vorteile unter anderem folgende:

- Die Entscheidungssicherheit wird erhöht, da durch den Business Case die Folgen des Vorhabens auf den Unternehmenserfolg genau analysiert und aufgezeigt werden.
- Übersicht und Transparenz werden geschaffen, da alle relevanten Faktoren für das Projekt betrachtet und zusammengefasst werden.
- Nachvollziehbarkeit wird geschaffen, da alle getroffenen Annahmen sowie Kennzahlen genau dokumentiert und einsehbar sind.
- Durch eine genaue Betrachtung innerhalb des Business Cases ist es möglich nur rentable Vorschläge weiterzureichen. Diese Vorauswahl erspart dem Management somit Zeit und Arbeit diese auszusortieren.
- Vergleichbarkeit zwischen Projektalternativen oder -vorschlägen ergibt sich durch eine detaillierte Gegenüberstellung dieser.

Zusätzlich kann die Erstellung eines Business Cases die Projektplanung positiv unterstützen. Einerseits wird die technische Grundlage durch eine wirtschaftliche Planungsgrundlage ergänzt, wodurch die Planungsarbeit im unternehmerischen Sinne verbessert

wird. Andererseits werden durch die intensivere Befassung mit dem Projekt mögliche Risiken erkannt und vorbeugende Maßnahmen können dementsprechend frühzeitig eingeleitet werden [Bru09].

2.5 Abgrenzung gegenüber dem Business Plan

Ein Business Plan oder auch Geschäftsplan genannt, unterscheidet sich deutlich von einem Business Case. Wohingegen bei einem Business Case die wirtschaftlichen Auswirkungen eines Projektes im Fokus stehen, liefert ein Business Plan eine schriftliche Zusammenfassung eines zukünftigen unternehmerischen Vorhabens [Bru09]. Dies kann beispielsweise eine Fusion, eine Neugründung oder eine Eintritt in einen neuen Markt sein. Wesentlich ist hierbei, dass neben der Beschreibung auch der konkrete Weg zur Erreichung des Ziels aufgezeigt wird. So werden Strategien und Maßnahmen detailliert erläutert und begründet warum diese gewählt wurden, wobei für die Begründung der gewählten Vorgehensweisen vorher durchgeführte Business Case dienen können. Somit dient ein Business Plan meist vorrangig dem Zweck Geldgeber wie Banken oder Risikokapitalgeber zu überzeugen um notwendiges Startkapital zu erhalten [Tas09].

3 Aufbau eines Business Case

Wie schon in den Grundlagen erwähnt, gibt es keinen Standard zur Erstellung von Business Cases, welcher beispielsweise eine Vorlage oder Inhaltsangabe zur Verfügung stellt. Jedoch gibt es grundlegende erforderliche Bestandteile, die in Abschnitt 3.1 weiter erläutert werden. Der auf diesen folgende Absatz beschäftigt sich damit, inwiefern ein Business Case mit Hilfe der Innovationsplattform erstellt werden kann.

3.1 Unverzichtbare Bestandteile

Bei der Erstellung eines Business Cases gibt es einige unverzichtbare Elemente, welche sich inhaltlich in die folgenden fünf Blöcke gliedern lassen. Hierbei liegt es in der Verantwortung des Erstellers diese in eine logische Struktur und angemessene Form der Darstellung zu bringen. Zusätzlich zu den unverzichtbaren Bestandteilen können weitere neue Punkte mit aufgenommen werden, falls diese für den jeweiligen Business Case von großer Bedeutung sind [SRb].

3.1.1 Formale Elemente, Themenbeschreibung, Zweck, Zusammenfassung und Einleitung

Der Erste Block fasst Formale Elemente und einen ersten Einstieg ins Thema zusammen. Im Folgenden werden diese Aspekte genauer erläutert.

Formale Elemente Unter Formale Elemente sind die Punkte Titel und Untertitel, Adressat und Autor, Datumsangabe und Haftungsausschuss zusammengefasst. Eine sorgfältige Wahl des Titels sollte beachtet werden, da diesen der Leser zu erst sehen wird. So beschreibt er das Thema kurz und knapp und nennt die Art der Analyse wie beispielsweise „Kosten-Nutzen-Analyse der Einführung der Software XYZ“. Zusätzlich kann noch ein Untertitel hinzugefügt werden, welcher zur näheren Erklärung bestimmter Schwerpunkte (Analysezeitraum, Charakteristiken der Methoden, etc.) dient wie z.B. „Planung für die Geschäftsjahre 2015-2020“. Eine Angabe des Autors sollte stets stattfinden, da so Rückfragen direkt an diese Person herangetragen werden können. Die Angaben des Adressaten ist hingegen nur wichtig, falls der Business Case von Personen außerhalb der Zielgruppe gelesen wird. Da diese ihn besser beurteilen können, wenn bekannt ist, an wen er gerichtet ist. Des Weiteren ist bei der Bearbeitung stets eine Datumsangabe auf dem Titelblatt Pflicht, da so immer schnell die aktuellste Version des Business Cases gefunden werden kann. Auch die Angabe von einem Datum im Haupttext bei der Erhebung von Daten ist wesentlich, da sich diese im Laufe der Zeit ändern können. Die zusätzliche Angabe eines Haftausschusses ist optional und bietet sich nur an, falls der Business Case für Leser außerhalb der eigenen Firma erstellt worden ist. So kann sich der Ersteller beispielsweise rechtlich gegenüber möglichen falschen Vorhersagen schützen [SRb].

Themenbeschreibung Die Themenbeschreibung lässt sich als allein stehenden Absatz darstellen oder in einem anderen Bereichen wie der Einleitung unterbringen. In ihr werden die Hauptmaßnahmen und die daraus resultierenden Aktivitäten komprimiert und detailliert beschrieben. Zudem sollten vor allem die Ziele und dessen Reichweite thematisiert werden, die das Projekt hat. Dies ist besonders wichtig, da das Erreichen von Zielen einen bestimmbaren unternehmerischen Nutzen hat. Wohingegen der Wert von Maßnahmen, welche kein bestimmtes Ziel verfolgen, nur schwer abzuschätzen ist [SRb].

Zweck Auch dieses Element lässt sich als eigenen Absatz darstellen oder in einem anderen einleitenden Bereich unterbringen. Dieses Element verdeutlicht den Beteiligten, wieso sie sich mit dem Business Case beschäftigen müssen. Dies kann zwar schon in der Überschrift angedeutet sein, wird jedoch in diesem Absatz noch einmal detaillierter erarbei-

tet. So wird der eigentliche Zweck wie beispielsweise einen bestimmten Budgetantrag zu unterstützen durch zusätzlichen Informationen erweitert. Zudem ist es wichtig wer den Business Case benutzt und wann er Anwendung findet. Des Weiteren sollte aufgezeigt werden, welche Entscheidungs- oder Planungsprozesse benötigt und welche Informationen erforderlich sind. Denn es für den Ersteller notwendig zu wissen, dass beispielsweise nur Projekte, dessen Amortisationsdauer geringer als drei Jahre sind, vom Management finanziert werden [SRb].

Zusammenfassung Die Zusammenfassung auch bekannt als Executive Summary ist eine der wichtigsten Bestandteile eines Business Cases. Einerseits enthält sie eine kurz Beschreibung des Themas inklusive Analysemethoden und andererseits die wichtigsten Ergebnisse belegt durch Kennzahlen aus der Analyse. Folglich wird die Zusammenfassung erst als letztes geschrieben, da die Resultate erst vorliegen müssen. Im Business Case steht die Zusammenfassung zu Beginn, da einige Leser nur diese lesen und danach entscheiden ob das Vorhaben weiter verfolgt werden soll oder nicht [SRb]. Gerade deshalb ist es wichtig durch diesen Abschnitt die Aufmerksamkeit der Leser bzw. Entscheider zu gewinnen [Bru09].

Einleitung In der Einleitung wird das Umfeld des betrachteten Vorhabens beschrieben. Somit kann in diesem Abschnitt die Geschichte, der Kontext oder weitere Hintergrundinformationen mit einfließen, wodurch die Erwartungen für den Leser geformt werden. Die vorher erläuterten Elemente wie die Themenbeschreibung oder der Zweck können in diesem Absatz untergebracht werden [SRb].

3.1.2 Annahmen und Methoden

Für diesen Block, welcher sich mit den Annahmen und Methoden beschäftigt, gibt es zwei Möglichkeiten wie er im Dokument aufgeführt werden kann. Zum Einen kann er zwischen der Einleitung und dem Block der betriebswirtschaftlichen Auswirkungen stehen. Andererseits, falls die Unterlagen sehr umfangreich sind, reicht es einen kompakten Abschnitt über die Hauptannahmen und die Methodiken vor den finanziellen Ergebnissen aufzuführen. Genauere Definitionen beispielsweise über das Kostenmodell können in den Anhang ausgegliedert werden. Im Folgenden werden die Bestandteile, welche zur Beschreibung der Annahmen und Methoden dienen, genauer erläutert. Ziel ist es, dem Leser nachvollziehbar zu verdeutlichen woher die Ergebnisse und Daten stammen [SRc].

Verwendete Finanzkennzahlen Bevor der Leser die Ergebnisse aus der Finanzrechnung liest, sollte er wissen auf welchen Finanzkennzahlen diese beruhen. Somit sollten diese aufgeführt und zudem begründet werden, aus welchem Grund sie gewählt wurden. So kann beispielsweise darauf hingewiesen werden, dass bestimmte Vorschläge vorrangig auf der Grundlage des Return of Investments bewertet werden [SRc].

Annahmen Ein Business Case basiert auf Annahmen, da er Voraussagen über die Zukunft macht. Sofern es Annahmen gibt, bei denen es eine Unsicherheit gibt, ob der Leser dieselbe Treffen würde, ist es wichtig diese explizit im Business Case aufzuführen. Die wichtigsten Annahmen sollten unter dem Punkt „Hauptannahmen“ zu Beginn zusammengefasst werden und die restlichen können der Vollständigkeit halber im Anhang aufgeführt werden [SRc].

Umfang und Abgrenzungen Durch die Beschreibung des Umfanges wird der Gültigkeitsbereich des Business Cases festgelegt. Hierbei müssen die Zeit, Organisation und Funktion, der geografische Standort und die betroffene Technologie festgelegt werden. Angaben zur Abgrenzung sind vor allem wichtig, wenn die Kosten und der Nutzen bei unterschiedlichen Organisationseinheiten entstehen [SRc].

Szenarien Ein Szenario ist die konkrete Abfolge von Ereignissen und stellt einen zu bewerteten Vorschlag dar. So enthält es eine Fülle von Annahmen, wie Marktgröße oder Kraftstoffpreise, aus einer Vielzahl von Quellen. Auch Bestandteile sind Regeln, Modelle oder Logiken, um aufzuzeigen, was in der Analyse behandelt werden muss. Fasst man alle Elemente eines Szenarios zusammen, so erhält man eine detaillierte Schilderung einer möglichen Zukunft. Falls mehrere Handlungsalternativen bestehen, muss für jede zu beurteilende Alternative ein genaues Szenario geschaffen werden. Nicht zu vergessen ist die Erstellung des Ausgangsszenarios, welches im Deutschen auch bekannt ist als das „Aktueller Kurs und Geschwindigkeit“- oder „Ist“-Szenario. Es stellt die Zukunft dar, wenn sich nichts verändern würde. Mit Hilfe dieses Ausgangsszenario ist es möglich durch einen Vergleich den Nutzen der Alternativen zu identifizieren. Ob eine vollständige Präsentation dieses „Ist“-Szenarios gegenüber dem Leser notwendig ist, hängt von dem Zweck, den die Analyse verfolgt, ab [SRc].

Kostenmodell Durch das Kostenmodell wird sichergestellt, dass keine überflüssigen und nur die relevanten Kostenpositionen im Business Case berücksichtigt werden. Es ist eine, wie Abbildung 2 zeigt, strukturierte Liste, die aus Kostenpositionen besteht. In die jeweili-

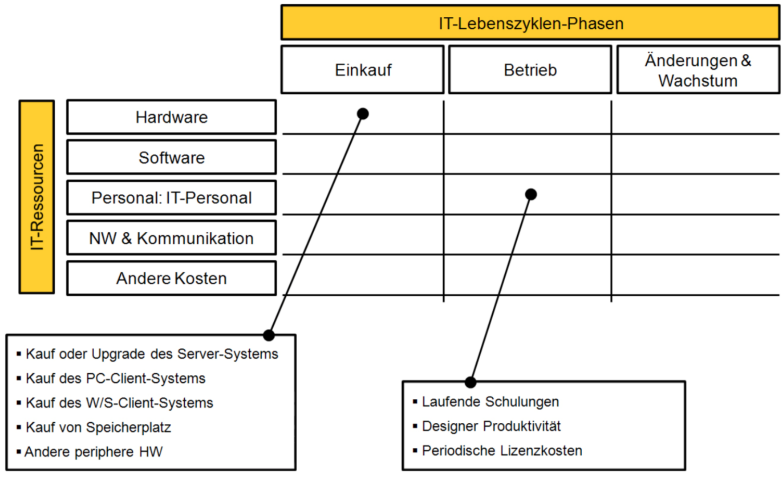


Abbildung 2: Kostenmodell eines Business Cases für das Upgrade eines IT-Systems [SRc]

gen Zellen können Gruppen von Kostenpositionen wie beispielsweise laufende Schulungen, Designer Produktivität und periodische Lizenzkosten eingetragen werden. Horizontal ist das Modell gewöhnlich in die drei Kategorien Einkauf, Betrieb und Änderungen & Wachstum gegliedert. Dies hat den Hintergrund da Betriebskosten anders gemanagt und geplant werden als Anschaffungskosten oder Wachstum- und Veränderungskosten. Die vertikale Einteilung in Abbildung 2 ist in diesem Fall auf die Kategorisierung von IT-Ressourcen spezialisiert und lässt sich beliebig für andere Business Cases anpassen. Falls eine Kostenposition sich keiner Zelle zuordnen lässt, so gehört dieser Posten nicht in die Analyse. Durch die Erstellung eines solchen Modells ergibt sich ein effektives Hilfsmittel zur Identifizierung und Analyse der Kostenseite. Zudem wird eine Vergleichbarkeit von verschiedenen Alternativen durch Anwendung des gleichen Kostenmodells geschaffen [SRc].

Nutzerargumentation Neben Kosteneinsparungen haben Firmen und Organisationen auch noch andere Ziele, die sie verfolgen. Jedoch lässt sich der betriebswirtschaftliche Nutzen nicht immer direkt in den Business Case integrieren. Die Nutzerargumentation bietet eine Grundlage, um diese Beiträge in angepasster Form in die Berechnung mit einzubeziehen. Das Ziel der gestiegenen Kundenzufriedenheit beispielsweise bedarf einer Nutzerargumentation, wie sie in Tabelle 1 abgebildet ist, um den Nutzen zu messen und einen Geldwert zuweisen zu können. So kann die Kundenzufriedenheit mit guten Ergebnissen in Kundenumfragen bestimmt und durch höhere Verkaufszahlen im Business Case dargestellt werden [SRc].

A.	Ist für uns eine höhere Kundenzufriedenheit ein wichtiges Unternehmensziel? Wenn „ja“, dann...
B.	Hat eine höhere Kundenzufriedenheit einen Wert? Wenn „ja“, dann...
C.	Anhand welcher greifbaren Größe (Kennzahl) können wir messen, dass sich die Kundenzufriedenheit erhöht hat?
D.	Was ist das Zielmaß der Steigerung (gerechnet in dieser Kennzahl)?
E.	Wie hoch bemisst sich der Wert für unser Unternehmen, wenn wir das Ziel erreichen?
F.	Wird sich das Vorhaben auf die Kundenzufriedenheit auswirken? Wenn „ja“, um wie viel?
G.	Wie hoch ist der Wert dieses Beitrags (dieser kann 100% des Wertes aus Ëßein oder, wenn es noch andere Faktoren gibt, die zur Erreichung des Ziels beitragen, um den entsprechenden Satz geringer als 100%).

Tabelle 1: Schema zur Bestimmung von Nutzen am Beispiel der Kundenzufriedenheit [SRc]

Datenquellen und Methoden Da in der Nutzerargumentation möglicherweise sogenannte „weiche Nutzenfaktoren“ eine Rolle spielen können, ist es wichtig so früh wie möglich diese mit der Zielgruppe abzustimmen und zu vereinbaren. Einerseits steht zur Frage ob bestimmte Faktoren berücksichtigt werden sollen und andererseits wie sie bewertet werden sollen. Des Weiteren ist bei der Nutzerargumention und dem Kostenmodell wichtig, dass alle Aussagen die getroffen werden für den Leser nachvollziehbar sind. Dies bedeutet, dass Datenquellen aufgeführt und Methoden, welche zur Berechnung von Werten verwendet wurden, genau beschrieben werden müssen [SRc].

3.1.3 Betriebswirtschaftliche Auswirkungen

In diesem Block werden die finanziellen Konsequenzen dargestellt, welche sich durch die Umsetzung des vorher beschriebenen Vorhabens ergeben. Die Analysen und Daten werden vorerst sachlich dargestellt, da Raum für Empfehlungen und Interpretationen erst in den letzten beiden Blöcken geboten wird [SRd].

Finanzmodell Nachdem mit dem Kostenmodell und der Nutzerargumention die Grundlage zur Bewertung geschaffen wurde, werden innerhalb dieses Elements erstmals die Kosten dem Nutzen gegenübergestellt. So kann beispielsweise ein Finanzmodell nur aus einer einzigen Cashflow-Rechnung bestehen, welche das Herzstücke eines Business Cases ist. Eine Zusammenfassung einer Cashflow-Rechnung ist in Abbildung 3 zu sehen. Der Nutzen und die Betriebskosten sind in dem Schaubild über die Projektlaufzeit nach Jahren aufgelistet. Hierbei ist zu beachten, dass diese Werte aus vorausgegangen Berechnungen stammen. Für

Cashflow-Zusammenfassung Geldzuflüsse (Geldabflüsse) (in TEuro)	Jahr 0 Aug 2010	Jahr 1 Aug 2011	Jahr 2 Aug 2012	Jahr 3 Aug 2013	Jahr 4 Aug 2014	Jahr 5 Aug 2015	**TOTAL**
Nutzen	320,0	1.491,5	1.753,1	2.007,1	2.246,7	2.493,1	**10.311,5**
Betriebsausgaben	-44,8	-1.076,1	-1.457,3	-1.586,5	-1.687,1	-1.873,6	**-7.725,5**
Netto Cashflow	**275,2**	**415,4**	**295,8**	**420,6**	**559,6**	**619,4**	**2.586,0**
Kumulativer Netto Cashflow	275,2	690,6	986,4	1.407,0	1.966,5	2.586,0	**2.586,0**
Diskontierter Cashflow							
bei 9.0 %	275,2	381,1	249,0	324,8	396,4	402,6	**2.029,0**
bei 15.0 %	275,2	361,2	223,7	276,5	319,9	308,0	**1.764,5**

Abbildung 3: Übersicht keiner Cashflow-Rechnung [SRd]

die Errechnung der Gesamtkosten bzw. des Gesamtnutzen pro Jahr müssen viele Kostenpositionen berücksichtigt werden, welche durch eine genaue Auflistung zuvor ausgerechnet wurden. Somit lässt sich nun der Netto Cashflow pro Jahr und für die gesamte Projektzeit ausrechnen, indem die Betriebsausgaben vom Nutzen abgezogen werden. Somit wird das erwirtschaftete Kapital ohne Steuerzahlungen oder ähnliches angegeben [Ame98]. Dies ist die wichtigste Größe bei der Cashflow-Rechnung, da sich aus ihr alle anderen Größen ableiten lassen wie beispielsweise der abgezinste Cashflow oder die Amortisationsdauer [SRd].

Analyse der Ergebnisse Nach der Erstellung des Finanzmodells folgt eine Analyse der resultierten Ergebnisse. Üblicherweise beginnt diese mit einer Übersicht über die Kennzahlen, welche auf den Netto-Cashflow-Entwicklungen basieren. Es sollten detaillierte Ausführungen vermieden werden, da die Zahlen zuerst im Vordergrund stehen und für sich selbst sprechen sollen. Beispielsweise bietet es sich diesbezüglich an, aus den errechneten Finanzkennzahlen grafische Darstellungen zu bilden. Konkretere Ausführungen folgen in den Abschnitten Sensitivität, Risiken, Schlussfolgerungen und Empfehlungen [SRd].

Nicht finanzielle Ergebnisse Einigen Einflussgrößen, wie beispielsweise das Image des Unternehmens, können nicht auf akzeptable Weise in Geldwerten ausgedrückt werden, da eine Schätzung zu schwer ist. Durch diese Tatsache tauchen sie nicht im Finanzmodell auf, obwohl sie möglicherweise wichtige Unternehmensziele sind. Somit ist es wesentlich diese trotzdem aufzuführen und sie bei einem Vergleich mit einzubeziehen [SRd]

3.1.4 Sensitivität und Risiko

Grundlegend ist bei Business Cases zu beachten, dass ihre Ergebnisse Vorhersagen über die Zukunft sind. Somit weisen sie eine gewisse Unsicherheit auf. In den folgenden Abschnitten werden zunächst Risikofaktoren ermittelt und mögliche Wahrscheinlichkeiten des gesamten

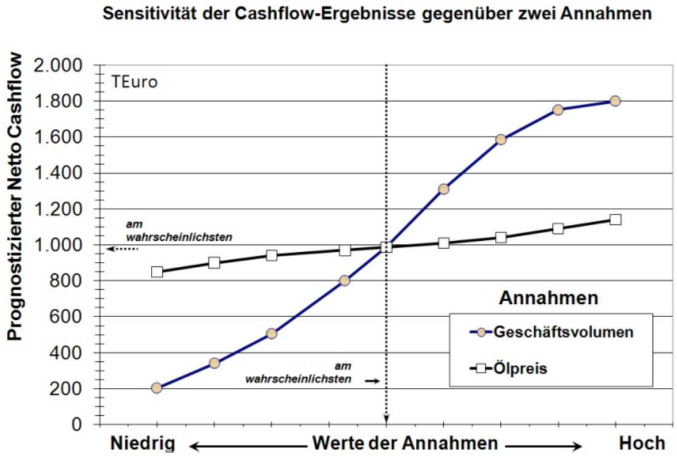


Abbildung 4: Darstellung einer einfachen Empfindlichkeitsanalyse [SRa]

Cashflows aufgeführt. Im Business Case selbst, können diese Punkte in einem Abschnitt zusammenfasst werden.

Sensitivitätsanalyse Die Sensitivitätsanalyse untersucht die Auswirkungen auf den Netto-Cashflow, wenn sich bestimmte Annahmen wie beispielsweise das Geschäftsvolumen ändern und alle anderen Faktoren jeweils eingefroren werden. Es empfiehlt sich mit entsprechender Software eine Empfindlichkeitsanalyse für diese Annahmen durchzuführen und somit Faktoren zu identifizieren, welche bei nur geringen Änderungen große Auswirkungen auf den Kapitalfluss haben. Abbildung 4 zeigt solch eine Analyse, bei welcher das Geschäftsvolumen und der Ölpreis analysiert werden. Es zeigt sich hierbei, dass der Ölpreis nur geringe Auswirkungen auf den prognostizierten Netto Cashflow hat. Das Geschäftsvolumen hingegen verändert je nach angekommener Größe maßgeblich den Netto Cashflow und kann somit als Risikofaktor eingestuft werden. Einige dieser Faktoren, wie zum Beispiel das Ausbildungsniveau der Mitarbeiter, lassen sich vom Management zu einem gewissen Grad kontrollieren oder beeinflussen. So wird neben der Identifizierung zudem aufgezeigt, welche Verantwortung bei den Entscheidern liegt, damit die Annahmen tatsächlich eintreffen [SRa].

Risikoanalyse Die Risikoanalyse prüft entgegen der Sensitivitätsanalyse keine einzelnen Faktoren, sondern analysiert das Risiko des Gesamtprojektes. Dies lässt sich beispielsweise mit einer wie in Abbildung 5 gezeigten Monte-Carlo-Simulation aufzeigen. In diesem Beispiel ist mit einem Ergebnis von 5 Mio. Euro mit eine Wahrscheinlichkeit von 50% zu rechnen. Wohingegen nur mit einer 20% Wahrscheinlichkeit ein Cashflow von 6 Mio. Euro

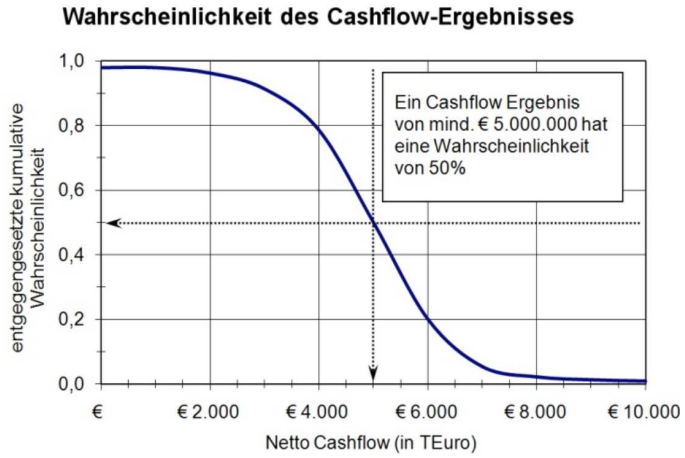


Abbildung 5: Monte-Carlo-Simulation im Zuge der Risikoanalyse [SRa]

erwartet werden kann. So werden dem Management die finanziellen Risiken des Projektes verdeutlicht [SRa].

3.1.5 Schlussfolgerungen und Empfehlungen

Im letzten Abschnitt des Business Cases werden die zuvor errechneten Zahlen interpretiert und es werden Verbindungen zu den Zielen, Maßnahmen und Entscheidungen hergestellt. Es ist nicht davon auszugehen, das alle Leser direkt die gleichen Schlussfolgerungen aus den finanziellen Analysen und Ergebnissen ableiten. Daher ist es wichtig die Argumentation diesbezüglich knapp und vollständig zu gestalten und mit den Beweisen aus den vorangegangenen Abschnitten zu belegen. Des Weiteren gehen aus den Analysen zum Teil auch unerwartete Ergebnisse hervor. Auf diese Chancen oder Risiken sollten besonders eingegangen werden. Auch betriebswirtschaftlichen Ziele, welche vorher formuliert wurden, sollten aufgegriffen und sich an ihnen orientiert werden. Des Weiteren sollte auf wichtige Entscheidungskriterien wie beispielsweise die Amortisationsdauer hingewiesen werden. Schlussendlich werden konkrete Empfehlungen gegenüber dem Management, wie zum Beispiel „Finanzieren Sie Vorschlag B“, ausgesprochen.

3.2 Unterbringung eines Business Cases in einer Innovationsplattform

Der in Abschnitt 3.1 beschriebene Aufbau zeigt, das die Erstellung eines Business Cases eine sehr komplexe Aufgabe ist, die zum Teil viel Fachwissen benötigt. Hieraus lässt sich schlussfolgern, dass die Umsetzung eines kompletten Business Cases innerhalb der Innovationsplattform nicht umsetzbar ist. Vielmehr bietet es sich an einen „Vorab Business

Case“ während des Innovationsprozesses zu erstellen, welcher zwar nicht die Ausmaße eines richtigen Business Case hat, jedoch fundierte Aussagen über die Wirtschaftlichkeit des Projektes treffen sollte. Viele Bestandteile wie beispielsweise der Titel und Untertitel oder die verfolgten Ziele ergeben sich schon automatisch während der Ideenfindung. Diese müssten lediglich abgegriffen und für den „Vorab Business Case“ zur Verfügung gestellt werden. Zudem sollte bei der Erstellung des Vorhabens darauf geachtet werden, dass der Ersteller durch vorgefertigte Eingabefelder geleitet wird, für den Business Case wichtige Informationen bereitzustellen. Ein wesentlicher zu betrachtender Faktor wäre hier beispielsweise das Kostenmodell. Werden die resultierenden Kosten und Nutzen schon während der Erstellung detailliert aufgelistet, so erleichtert dies die Cashflow-Rechnung deutlich. Dies könnte durch einfache Kosten- und Nutzen-Tabellen realisiert werden. Letztlich die Schlussbetrachtung mit der persönlichen Interpretation und die anfängliche Zusammenfassung sollten explizit vom Ersteller selbst geschrieben werden, da komplexe Zusammenhänge erklärt und belegt werden müssen. Punkte wie die Sensitivitäts- und Risikoanalyse könnten hingegen entfallen, da sie sehr umfangreich sind und zu viel Fachwissen benötigen.

Falls nun durch das Erstellten des kleinen Business Case ersichtlich wird, dass es sich um ein wirtschaftlich vorteilhaftes Vorhaben handelt, kann ein richtiger Business Case in Auftrag gegeben werden. Dessen Erstellung jedoch müsste von Fachpersonal und außerhalb der Innovationsplattform stattfinden. In diesem wird das Projekt hinsichtlich aller unverzichtbaren Bestandteile und gegebenenfalls zusätzlicher analysiert. Nach der Erstellung sollte der Business Case in die Plattform mittels Datei-Upload oder ähnlichem eingebunden werden, damit er für die Benutzer zugänglich ist und kein Medienbruch entsteht.

4 Fazit

Es hat sich gezeigt, dass der Business Case ein gutes Entscheidungsunterstützungsinstrument ist, um die Wirtschaftlichkeit eines Projektes zu bestimmen. Aufgrund seiner Komplexität bietet es sich jedoch an, einen „Vorab Business Case“ in die Innovationsplattform zu integrieren, welcher weniger umfangreich ist. In diesem werden zum Großteil die Elemente berücksichtigt, welche sich bei der Erstellung des Vorhabens ergeben. Nur einige Rechnungen und Interpretationen, welche auch ohne viel Zusatzwissen vom Nutzer erledigt werden können, werden ergänzt. So ergibt sich eine Grundlage, auf der beschlossen werden kann, ob ein richtiger Business Case von Fachkräften gerechnet werden sollte. Nach Erstellung kann dieser schlussendlich beispielsweise via Datei-Upload den Nutzern auf der Plattform zur Verfügung gestellt werden.

Literatur

[Ame98] Matthias Amen. *Erstellung von Kapitalflußrechnungen*, volume 2. 1998.

[Bru09] Ralph Brugger. *Der IT Business Case: Kosten erfassen und analysieren-Nutzen erkennen und quantifizieren-Wirtschaftlichkeit nachweisen und realisieren.* Springer-Verlag, 2009.

[SRa] Marty Schmidt and Johannes Ritter. So erstellen sie einen business case teil 4: Sensitivität, risiko, empfehlungen.

[SRb] Marty Schmidt and Johannes Ritter. So schreiben sie einen business case teil 1: Formalien und einstieg.

[SRc] Marty Schmidt and Johannes Ritter. So schreiben sie einen business case teil 2: Annahmen und methoden.

[SRd] Marty Schmidt and Johannes Ritter. So schreiben sie einen business case teil 3: Betriebswirtschaftliche auswirkungen.

[Tas09] Andreas Taschner. *Business Cases.* Springer, 2009.

B.1.2. Design Thinking

Design Thinking

Seminararbeit
im Rahmen der Projektgruppe

Themensteller:	Prof. Dr.-Ing. Jorge Marx Gómez
Betreuer:	Dipl. Math. Jens Siewert Stefan Wunderlich, M. Sc. Dr. Joachim Kurzhöfer
Vorgelegt von:	Jan Fischer Thorner Str. 7 26122 Oldenburg +49 174 749 55 69 jan.fischer@uni-oldenburg.de
Abgabetermin:	20. August 2015

Inhaltsverzeichnis

Abbildungsverzeichnis

1. Einleitung

Innovationen stellen für Unternehmen einen zentralen Wettbewerbsfaktor dar: Als Abgrenzung zur Konkurrenz durch neue Produkte oder Dienstleistungen machen sie ein Unternehmen einzigartig, als neues Verfahren in der Produktion erlauben sie Kosten zu sparen und ein Produkt günstiger als die Konkurrenz anzubieten. Jedoch stellen kreative Ideen oder neue Forschungsergebnisse noch keine Innovation dar. Erst wenn die Umsetzung dieser Ideen in ein konkretes Produkt oder eine Technologie (Invention) einen Markt durchdringt (Diffusion) ist als Ergebnis eine Innovation entstanden (vgl. [MPD14], S. 7.) Diese Ergebnisse sind in der Regel das Resultat von Designprozessen, damit ist jedoch nicht Design im Sinne von „Dinge schön(er) machen gemeint“ sondern, die durchdachte, kreative Lösung eines Problems (vgl. [MG13], S. 6.). Diese kreativen Problemlösungen sind nur selten ein Zufall oder das Ergebnis eines einzelnen Genies, viel mehr sind Innovation das Resultat einer strukturierten Herangehensweise. Design Thinking stellt eine solch strukturierte Herangehensweise dar, mit der es möglich sein soll schwierige Probleme zu lösen und innovative Ideen und Produkte hervorzubringen. Der Einsatz von Design Thinking in der Softwareentwicklung stellt den nächsten logischen Schritt dar. Ähnlich wie im *human-centered design* steht damit bereits bei der Entwicklung das spätere Nutzungserlebnis im Vordergrund (vgl. [LM10], S. 10).

1.1. Zielsetzung und Vorgehensweise

Ziel dieser Arbeit ist es Design Thinking als Problemlösungsstrategie zu untersuchen, die zu Grunde liegenden Elemente zu identifizieren und ein konkretes Vorgehensmodell vorzustellen. Des Weiteren wird auf der Basis der wesentlichen Elemente der Einsatz von Design Thinking in Softwareentwicklungsprojekten geprüft. Abschließend wird im Rahmen des Fazits eine Handlungsempfehlung gegeben, ob und gegebenenfalls wie Design Thinking innerhalb der Projektgruppe IMPACT umgesetzt werden kann.

In Abschnitt 2 wird dazu zunächst Design Thinking vorgestellt, abgegrenzt und die wesentlichen Elemente erläutert. Im weiteren Verlauf wird das Vorgehensmodell der HPI School of Design in Potsdam genauer betrachtet. Der darauffolgende Abschnitt 3 untersucht wie Design Thinking sinnvoll in Softwareentwicklungsprojekten, im speziellen im Einsatz mit agilen Softwareentwicklungsmethoden, genutzt werden kann. Im Zuge dessen wird ein Praxisbeispiel vorgestellt und daraus letztendlich Erkenntnisse für den Einsatz von Design Thinking in Softwareprojekten abgleitet. Die Arbeit schließt mit Abschnitt 4, welcher das Fazit sowie die Handlungsempfehlung enthält.

2. Design Thinking

In diesem Abschnitt wird Design Thinking genauer betrachtet, dazu wird zunächst der Ursprung des Begriffs beleuchtet und dessen verschiedenen Bedeutungen diskutiert. Anschließend wird betrachtet wie und warum sich Design Thinking von anderen Problemlösungsstrategien unterscheidet. In diesem Zusammenhang wird die Zielsetzung des Design Thinking beschrieben. Es folgt eine Erläuterung der zentralen Elemente und Übersicht der verschieden Vorgehensmodelle und deren Gemeinsamkeiten. Abschließend wird der iterative Prozess im Design Thinking und die dazugehörigen Phasen nach dem Modell der HPI School of Design Thinking in Potsdam betrachtet.

2.1. Ursprung

Das Konzept des Design Thinking entstand bereits in den 60er und 70er Jahren, während jedoch die Verknüpfung mit dem Begriff Design Thinking erst Jahrzehnte später statt fand (vgl. [Win11], S.9). So schreibt Simon ([Sim69]) 1969 von der Wissenschaft des Designs und Rowe ([Row91]) veröffentlicht 1991 unter dem Titel „Design Thinking“ eine Sammlung von Methoden und Übungen, die im Rahmen des Architekturstudiums an der University of Pennsylvania verwendet wurden. Ein genauer Zeitpunkt, wann der Begriff Design Thinking zum ersten Mal in seiner jetzigen Form verwendet wurde, ist nicht eindeutig geklärt. Ein möglicher Zeitpunkt wird vom damaligen Dekan der Rotman School of Management Martin und Brown, dem CEO von IDEO, einer amerikanischen Design- und Innovationsberatung, die später das Konzept Design Thinking vermarkten, genannt. Demnach sei 2003 als Geburtsstunde des Design Thinking anzusehen, als die ersten konkreten Gespräche über die Methode Design Thinking geführt wurden (vgl. [Tis09]). Diese Annahme wird durch Engchuan gestärkt, welche die Literatur zu Design Thinking in zwei Kategorien aufteilt: Literatur, die nach der Begriffsschöpfung in 2003 entstand und Literatur, die bereits vorher die Vorgehensweise von Designern bei der Problemlösung beschreibt (vgl. [Eng12], S. 7). Nach Weinberg gilt David Kelley, Mitbegründer von IDEO, als Erfinder des Innovationsansatzes Design Thinking, der von den Stanford Professoren Terry Winograd und Larry Leifer maßgeblich geprägt wurde (vgl. [pfe12], S. 247) . In diesem Zusammenhang steht auch die Gründung im Jahr 2003 der *d.school* bzw. des „Hasso-Plattner-Institute of Design“ in Stanford, die seit 2005 Design Thinking als formale Methode lehrt (vgl. [PML10], S. V).

2.1.1. Begriffsdiskussion

Ähnlich wie der Erschaffungszeitpunkt ist auch die Bedeutung des Begriffs Design Thinking nicht eindeutig geklärt. Nach David Kelley, der sich 2009 in einem Interview dazu äußerte, ist Design Thinking eine Methodologie (bzw. Methode), die es ermöglicht eine Lösung zu finden, auf die keiner zuvor gekommen ist (vgl. [Tis09]). Weitere Bedeutungen sind: Design Thinking als Prozess mit fest definierter Ein- und Ausgabe, als Geisteshaltung oder Denkweise („mindset“) oder als eigene wissenschaftliche Disziplin (vgl. [Win11], S. 11ff). Welche Definition zutreffend ist, kann in der Literatur nicht eindeutig geklärt werden. Ebenso ist eine mehrfache Bedeutung möglich, so beschreibt die HPI d.school in Potsdam Design Thinking als „ein Prozess UND ein Mindset. Und noch viel mehr.“ ([hpi]) und an anderer Stelle als „neuartige Methode zur Entwicklung innovativer Ideen“ ([hpi07]).

2.1.2. Design Thinking als Problemlösungsstrategie

Nach Engchuan definiert Hasso Plattner, Mitgründer des Softwareunternehmens SAP und Förderer des „Hasso-Plattner-Institute of Design“, Design Thinking wie folgt:

> „Design Thinking ist eine systematische Innovationsmethode, die in allen Lebensbereichen angewendet werden kann. Design Thinking ist kein Algorythmus [sic!], also eine genau definierte Handlungsvorschrift zur Lösung eines Problems, (...), sondern eine Heuristik, die ganz bestimmte Verfahrensschritte vorgibt, die sich in der Praxis in einer bestimmten Abfolge als zweckmäßig erwiesen haben und die unter ganz bestimmten Bedingungen, (...), ihr vollständiges Erfolgsspektrum entfalten können.“
>
> ([PMW09], S. 103, zitiert nach [Eng12], S. 9)

Diese Definition vereint die Bedeutung des Design Thinking als Methode mit dem Prozesscharakter, der bereits in Abschnitt 2.1.1 erwähnt wurde. Design Thinking als Vorgehensmodell zur Problemlösung scheint somit als geeignetste Definition für die Zielsetzung dieser Arbeit.

2.2. Ziele des Design Thinking

Praktisch wird Design Thinking zur Erreichung von zwei Zielen eingesetzt, wobei diese sich üblicherweise überschneiden: Zum einen sollen damit neue, kreative Ideen und Innovationen generiert werden. Zum anderen dient es dazu komplexe Probleme zu lösen, die mit der üblichen Herangehensweise nicht lösbar sind (vgl. [Win11], S. 13). In den beiden folgenden Abschnitten werden diese Ziele genauer erläutert.

2.2.1. Innovation

Betrachtet man das erste Ziel, die Erschaffung von Innovationen, stellt sich die Frage, wie eine Innovation definiert ist. Porter versteht darunter „die Generierung von neuem Wissen und dessen Umsetzung in neue Produkte, Dienstleistungen, Geschäftsmodelle“ ([RA12]). Die Erzeugung von neuen Ideen stellt nach Porter hingegen nur eine Invention, eine Erfindung dar, die erst durch die „ökonomische Nutzbarmachung“ zu einer Innovation wird ([RA12]). Im Design Thinking selbst existiert keine eigenständige Definition für den Begriff der Innovation. Allerdings beschreibt Brown Rahmenbedingungen (*constraints*) für die Entwicklung erfolgreicher Ideen, die er als drei überlappende Bereiche beschreibt: *desirability*, *viability* und *feasibility* ([Bro09], vgl. 19). In der deutschen Praxis-Literatur zu Design Thinking werden diese Begriffe in *Wünschbarkeit*, *Wirtschaftlichkeit* und *Machbarkeit* übersetzt und als Perspektiven oder Merkmale beschrieben (vgl. [MG13], S. 10ff). Nach Engchuan wird dabei der Wünschbarkeit (dem Faktor Mensch) der höchste Stellenwert zugeschrieben, da Innovationen im Design Thinking nutzerzentriert (*human-centered*) sind (vgl. [Eng12], S. 13). Eine Innovation stellt die erfolgreiche Balance zwischen den drei Bereichen dar: Die neuartige Ideen muss die Bedürfnisse der potentiellen Nutzer befriedigen, dabei gleichzeitig mit den gegebenen technischen Möglichkeiten realisierbar sein und letztendlich auch wirtschaftlich erfolgreich sein.

2.2.2. Probleme

Um zu verstehen, was das zweite Ziel „die Lösung von komplexen Problemen“ aussagt, ist es wichtig die verschiedenen Arten von Problemen zu definieren und festzulegen was ein komplexes Problem ausmacht. Nach Engchuan und Winkler können Probleme in drei Kategorien eingeteilt werden (vgl. [Eng12], S. 10; vgl. [Win11], S. 13f):

- **Well-defined problems:** Einfach Probleme, die klar strukturiert sind. Sowohl das Ziel als auch die Vorgehensweise zu Lösungsgenerierung sind bekannt. Die Lösung eines solchen Problems ist als Routineaufgabe zu verstehen und die Lösungen sind entweder falsch oder richtig.
- **Ill-structured problems:** Die Struktur von diesen Problemen und die Rahmenbedingungen sind nicht eindeutig definiert. Auf Grund dessen wären viele Lösungen möglich, die jedoch von dem jeweiligen Kontext und den dazu gehörigen Rahmenbedingungen abhängig sind.
- **Wicked problems:** Diese Art von Problemen sind noch unklarer formuliert, es ist kein Ziel definiert und jede Lösung kann noch weiterentwickelt werden. Zudem sind

die Rahmenbedingungen (zumindest teilweise) und die Interessen der beteiligten Parteien widersprüchlich.

Da bei *ill-structured* und *wicked problems* rein analytische Verfahren, bei denen Probleme in Teilprobleme zerlegt werden, zu keinem Ergebnis führen können, eignet sich Design Thinking vor allem für diese Art von Problemen (vgl. [Win11], S. 14).

2.3. Abgrenzung zu anderen Problemlösungsstrategien

Nachfolgend werden die Unterschiede zu klassischen Strategien und wesentlichen Merkmale des Design Thinking erläutert.

2.3.1. Lineares vs. iteratives Vorgehen

Wie in Abschnitt 2.2.2 bereits erwähnt, eignet sich Design Thinking dazu Probleme zu lösen, die schlecht strukturiert sind und mit anderen Strategien nicht gelöst werden können. Die Vorgehensweise ist bei herkömmlichen Problemlösungsstrategien linear. Die Problemlösung erfolgt nach einem fest vorgegebenem Muster in einer strikten Reihenfolge: Als Anfangspunkt wird das Problem betrachtet, analysiert und definiert, daraus werden Anforderungen an die Problemlösung abgeleitet. Die Lösung des eigentlichen Problems erfolgt im nächsten Schritt, indem diese Anforderungen sinnvoll kombiniert werden (vgl. [Eng12], S. 14). Der Erfolg dieser Strategie hängt davon ab, wie genau das Problem definiert, analysiert und in Teilprobleme zerlegt werden kann, denn nur mit ausreichend Erkenntnissen über das Problem kann eine Lösung abgleitet werden. Die *ill-structured* oder *wicked problems*, die im Design Thinking betrachtet werden, sind auf diese Weise jedoch nicht lösbar, da diese Probleme zu komplex und die Rahmenbedingung nicht ausreichend bestimmbar sind (vgl. [Eng12], S. 14). Die Problemlösung im Design Thinking erfolgt hingegen iterativ. Mit jeder Iteration wird das Problem bzw. der Problembereich neu und genauer definiert. Die zunächst unbekannten Rahmenbedingung werden in diesem Prozess nach und nach erarbeitet. Anzumerken ist hierbei, dass damit nicht die sequentielle Wiederholung von Prozessschritten, sondern das ständige Wechseln zwischen verschiedenen Phasen gemeint ist (vgl. [Win11], S. 26). Die einzelnen Phasen im Design Thinking und deren Bedeutung werden im Abschnitt 2.5.3 erläutert.

2.3.2. Divergentes und Konvergentes Denken

Ein wesentliches Merkmal, das ebenso eine Abgrenzung zu anderen klassischen Problemlösungsstrategien darstellt, ist die Kombination von divergenten und konvergentem Denken in

abwechselnden Phasen (vgl. [Bro09], S. 41f). Beim konvergenten (zusammenlaufenden) Denken werden dabei mehrere einzelne Faktoren bis hin zu einer Lösung gebündelt, dieser Vorgang läuft logisch, planmäßig und streng rational ab. Dieses Vorgehen stellt einen praktischen Weg dar existierende Alternativen abzuwägen (vgl. [Bro09], S.41), hemmt jedoch die Schaffung von gänzlichem Neuen. Da hingegen zeichnet sich divergentes Denken durch eine offene und unsystematische Herangehensweise aus, die es ermöglicht viele Lösungs-Alternativen zu generieren (vgl. [Eng12], S. 17). Die Kombination aus divergentem und konvergentem Denken ermöglicht es zunächst viele Alternativen zu generieren durch die sich jedoch auch die Komplexität von möglichen Lösungen erhöht. Durch das anschließende konvergentes Denken werden diese gebündelt und sinnvoll kombiniert. Dieser Vorgang wird im Abschnitt 2.5 noch genauer betrachtet.

2.3.3. Analyse und Synthese

Nach Brown stellen Analayse und Synthese die natürliche Ergänzung zu divergentem und konvergentem Denken dar (vgl. [Bro09], S. 42) . Ähnlich wie bei klassischen linearen Problemlösungsstrategien dient die Analyse dazu komplex Sachverhalte in Teilkomponenten zu zerlegen, um diese besser zu verstehen. In der analytischen Phase des Design Thinking werden möglichst viele Daten und Informationen gesammelt. In der darauf folgenden synthetischen Phase werden diese Information verdichtet (vgl. [Eng12], S. 18). Nach Brown ist das der eigentliche kreative Prozess, bei dem aus den Massen von Daten Muster erkannt werden (vgl. [Bro09], S. 43). Die Kombination von divergentem und konvergentem Denken im Rahmen von Analyse und Synthese stellt den Grundgedanken des Design Thinking dar (vgl. [Bro09], S. 43).

2.3.4. Weitere Merkmale

Ein weiteres Abgrenzungsmerkmal stellt der menschzentrierter Ansatz (*human-centered*) dar. Wie schon in Abschnitt 2.2.1 bereits erwähnt, stellt der Faktor Mensch, in Form der Wünschbarkeit den wichtigsten Part der Innovation dar. Des Weiteren nennt Winkler noch das ganzheitliche Denken (*holistic thinking*) oder das Systemdenken (*system thinking*), bei dem das gesamte Umfeld oder die Interaktion von Produkten und Services betrachtet wird (vgl. [Win11], S. 28).

2.4. Voraussetzungen und Rahmenbedingungen

Nachdem in den vorherigen Abschnitten die wesentlichen Merkmale und Abgrenzungen zu anderen Problemlösungsstrategien genannt wurden, werden in diesem Abschnitt die

Voraussetzungen für ein erfolgreiches Design Thinking und die dazugehörigen Rahmenbindungen erläutert. Nach der HPI School of Design Thinking in Potsdam stellen die „drei großen P“ *People*, *Place* und *Process* dabei die Grundvoraussetzungen für das Design Thinking dar, die im Folgenden genauer erläutert werden [hpi].

2.4.1. Design Thinking-Teams

Das richtige Design-Thinking-Team stellt eine Grundvoraussetzung dar, dabei sollte das Projekt-Team aus Personen mit unterschiedlichen Hintergründen aus diversen Disziplinen gebildet werden (vgl. [Win11], S. 30). Die HPI School of Desing Thinking setzt dabei auf multidisziplinäre Teams (vgl. [hpi07]), während Brown von interdisziplinären Teams ausgeht (vgl. [Bro09], S. 23). Nach Brown unterscheiden sich die Teams wie folgt: In multidisziplinären Teams bilden sich aus den Experten Befürworter der eigenen Spezialisierung, das Ergebnis stellt lediglich einen Kompromiss aller Disziplinen dar. In interdisziplinären Teams werden kollektive Ideen entwickelt, für die sich jeder verantwortlich fühlt (vgl. [Eng12], S. 25; vgl. [Bro09], S. 23).

2.4.2. Design Thinker

Damit multi- bzw. interdisziplinäre Teams zusammenarbeiten können, stellt Plattner bestimmte Anforderungen an diese Personen und spricht von *t-shaped*-Persönlichkeiten (vgl. [PML10], S. 36). Auf horizontaler Ebene müssen Design Thinker demnach bestimmte *Soft Skills* aufweisen: Empathie, integratives Denken, Experimentierfreude, Teamfähigkeit, Optimismus und Kreativität (vgl. [Eng12], S. 25). Vor allem die Fähigkeit zur Empathie stellt eine wichtige Eigenschaft dar, ohne die der menschzentrierte Ansatz des Design Thinking nicht möglich wäre. Die vertikale Basis einer *t-shaped*-Persönlichkeit stellen die Fähigkeiten und das Fachwissen einer speziellen Disziplin dar.

2.4.3. Raumkonzepte

Neben der HPI School of Design Thinking wird sowohl in der Arbeit von Engchuan, als auch in der Praxis-Literatur davon ausgegangen das spezielle Raumkonzepte dem Design Thinking bzw. benötigten Kreativität förderlich sind. Diese Räume zeichnen sich vor allem dadurch aus, das diese flexibel gestaltbar sind. Mittels mobilem Mobiliar, z.B. verschiebbaren Trennwände und Tischen, lässt sich der Raum umgestalten und an die jeweiligen Bedürfnisse anpassen. Die Trennwände und andere vertikalen Flächen sollen dazu dienen, Information und Ergebnisse des Design Thinking festzuhalten und darzustellen. Dazu sollten ausreichend Materialien, zum Basteln von Prototypen oder Visualisierung von Ideen

verfügbar sein (vgl. [hpi]; vgl. [Eng12], S. 26; vgl. [MG13], S. 20f).

2.4.4. Regeln

Für ein erfolgreiches Design-Thinking existieren verschiedene Regeln, die dazu dienen den optimalen Ablauf zu gewährleisten und kreative Prozessen und Zusammenarbeit zu fördern. Die nachfolgenden Regeln stellen einen Auszug aus 12 Regeln dar. Diese beziehen sich vor allem auf Phasen der Ideengenerierung, sind jedoch auch allgemein gültig:

- *Fail early and often*: Dieses Paradigma stellt eine der wichtigsten Grundregeln dar. Im Design Thinking wird scheitern nicht als Versagen sondern als Lernprozess angesehen, durch die iterative Vorgehensweise wird durch jede Wiederholung aus den vorherigen Fehlern gelernt. Dabei gilt: Je früher sich etwas als nicht zielführend herausstellt, desto besser ist es (vgl. [Eng12], S. 28).
- *Don't talk. Do!*: Ein begrenzter Zeitrahmen sorgt dafür, dass Ideen nicht einer innerlichen Zensur unterlaufen.
- *Be visual*: Ideen sind oftmals besser zu verstehn, wenn diese visaulisiert werden. Ein weiterer Vorteil ist, dass andere nicht durch das Lesen eines Wortes nur diesen Ansatz verfolgen, sondern jeder die Visualisierung anders wahrnimmt und darunter andere Dinge versteht (vgl. [Eng12], S. 2).
- *Build on ideas of other*: Um mit Design Thinking andersartige Ideen durch die Kombination von vielen Experten zu erreichen, sollten die Ideen der anderen nicht nur hingenommen sondern, mit den eigenen Fähigkeiten weiterentwickelt werden.
- *Think human centered*: Wie in den Abschnitten 2.2.1 und 2.3.4 bereits erwähnt, steht bei allem im Design Thinking der Mensch im Vordergrund.

In vollständige Auflistung der Regeln findet sich in [ER13], S. 23.

2.5. Vorgehensweise

In der Literatur werden zur Vorgehensweise im Design Thinking zahlreiche Prozess- oder Phasenmodelle vorgeschlagen. Obwohl Design Thinking selbst kein Prozess ist, stellt die Vorgehensweise im Design Thinking eine Reihenfolge von Design-Prozessen dar, die durch ein Modell veranschaulicht werden kann (vgl. [Win11], S. 14). Winkler, Lindberg und Raimer geben jeweils in ihren Arbeiten eine Übersicht, über die in der Literatur vertretenen Modelle. Diese unterscheiden sich in Aktualität, Abstraktionsebene und Reifegrad und

Autor/ Institution	Prozess-Schritte/-Phasen							
Bauer & Eagen (2008, S.66ff)	Understanding			Dreaming		Building		
	Immersing		Redefining	Imagining	Opting	Prototyping		Presenting
Brown (2008, S. 5), Brown & Wyatt (2010, S.33ff)	Inspiration			Ideation				Implementation
Argentur Dark Horse (2011)	Verstehen	Beobachten	Synthese	Ideen		Prototypen	Testen	
Dunne & Martin (2006, S.518)				Generate Ideas (Abduction)		Predict Consequences (Deduction)	Test	Generalize (Induction)
D-School am Hasso Plattner Institut, Potsdam (2010)	Understand	Observe	Point-of-View	Ideate		Prototype	Test	
Argentur Ingosu (2011)	Verstehen	Beobachten	Definieren	Ideen bilden		Visualisieren	Testen	
Kelly (2004, S.6f)	Understand	Observe		Visualize		Evaluate and Refine		Implementation
Lindberg et al. (2010, S.247ff)	Grasping External Knowledge		(Re)Framing the Design Problem	Ideating	Concept Specifying	Making it tangible		
			Synthesizing		Synthesizing			
			Path Selecting		Path Selecting			
	Knowledge Pooling							
	eine sinnvolle Zuordnung von den "Six Working Rules" war nicht möglich;							
Stickdorn & Schneider (2010, S. 122ff)	Exploration			Creation		Reflection		Implementation
Universität St. Gallen (2011)	(Re)Define the problem | Needfinding & instant expertise			Brainstorm & ideation		Prototype	Test	

Abbildung 1: Die Abbildung zeigt eine Überblick über die in der Literatur vorhandenen Vorgehensmodelle zum Design Thinking, adaptiert nach [Win11], S. 24.

der daraus resultierende Anzahl von Phasen oder Prozessschritten (vgl. [Win11], S. 14; [Lin13], S. 178ff; [Rai14], S. 180f). Die Abbildung 1 zeigt eine Auswahl und verdeutlicht die Unterschiede und Gemeinsamkeiten. Neben den in Abschnitt 2.3 genannten Merkmalen haben alle Modelle eine weitere Gemeinsamkeit: Obwohl sie eine unterschiedlich Anzahl an Phasen haben, lässt sich die Sequenz auf ein Prozessmodell reduzieren, das sich in zwei Bereiche aufteilen lässt, die jeweils in konvergente und divergente Phasen geteilt werden können. Die Abbildung 2 zeigt dieses Phasenmodell, welches im Folgenden erläutert wird.

2.5.1. Problemraum explorieren

Da Design Thinking vor allem für das Lösung von unklaren Problemen genutzt wird, stellt das Explorieren des Problemraums einen wesentlichen Teil des Design Thinking dar. In der divergenten Phase wird dabei versucht so viele Daten wie möglich über das Problem zu erheben. In der darauffolgenden konvergenten Phase werden die erhobenen Daten auf tiefer gehende Probleme analysiert und selektiert. Nach Winkler folgt als nächstes ein *reframing* bei dem das Problem anhand der gewonnenen Informationen neu definiert wird,

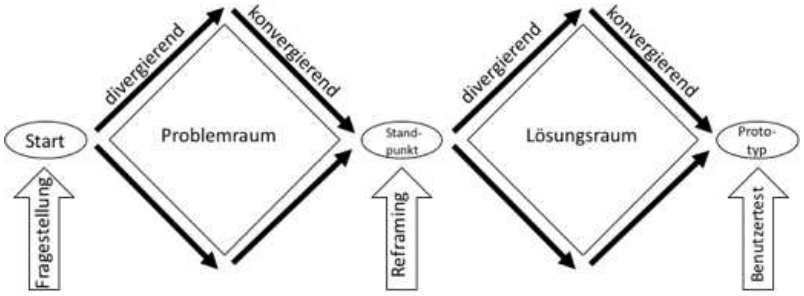


Abbildung 2: Das Phasenmodell zeigt das Zusammenspiel von divergierenden und konvergierenden Denkprozessen, adaptiert nach [Rai14], S.179.

sowie die Aufstellung von Theorien und Hypothesen (vgl. [Win11], S. 25). Ziel ist neben der Gewinnung von neuen Erkenntnissen die Festlegung der Rahmenbedingungen für die möglichen Lösungen.

2.5.2. Lösungsraum explorieren

Nachdem das Problem neu und genauer definiert wurde, werden in einer weiteren divergenten Phase möglichste viele Ideen und Lösungen dazu generiert. Ziel ist es das Problem aus möglichst vielen Perspektiven zu betrachten um somit eine optimale Lösung zu finden (vgl. [Win11], S. 25). In der anschließenden konvergenten Phase werden die zuvor entwickelten Ideen auf ihre Umsetzbarkeit geprüft. Verschiedene Lösungen werden dabei kombiniert um möglichst viele Aspekte abzudecken (vgl. [Win11], S. 25). Es erfolgt eine Reduktion auf nur wenige Ideen, die daraufhin weiterverfolgt und getestet werden.

2.5.3. Design Thinking-Phasen nach HPI

In den vorherigen Abschnitten wurden die wesentlichen Merkmale, sowie die Gemeinsamkeiten aller Vorgehensmodelle erläutert. In diesem Abschnitt wird das von der HPI School of Design Thinking in Potsdam entwickelte Vorgehensmodell genauer betrachtet und Praxisbeispiele zur Anwendung der einzelnen Phasen geliefert. Die Wahl fiel auf dieses Modell, da es zum einen in der Praxisliteratur weit verbreitet ist (vgl. [Eng12], S. 29) und zum anderen diese Variante ebenfalls bei SAP eingesetzt wird (vgl. [SAP12]). Das Modell ist in sechs Schritte oder Phasen aufgeteilt, die in der Abbildung 3 gezeigt werden. Wie schon in Abschnitt 2.3.1 angemerkt sind diese Schritte nicht als sequenzielle Abfolge zu verstehen, viel mehr sind es einzelnen Phasen, die untereinander iterativ verbunden sind und sich überlappen (vgl. [Eng12], S. 30). Zwischen den Phasen finden immer wieder Rückkopplungen statt, ein linearer Ablauf ist weder beabsichtigt noch gewünscht. Ob eine

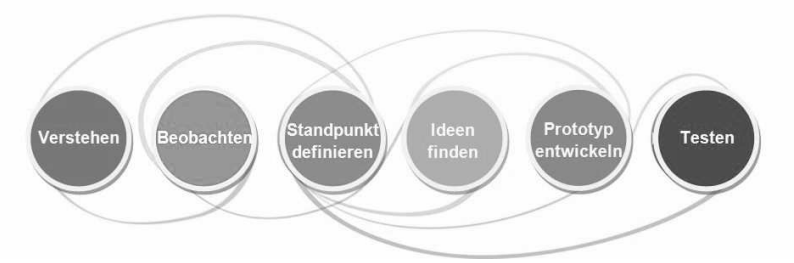


Abbildung 3: Das Vorgehensmodell zeigt die einzelnen Phasen nach der HPI School of Design Thinking, und wie diese miteinander iterativ verbunden sind [TS10]

Phase abgeschlossen ist oder diese wiederholt werden muss, wird durch das Team oder das Versagen einer Hypothese oder eines Prototypen entschieden.

Verstehen: Nach Guertler und Meyer entsteht eine echte Innovation meist erst aus dem tiefen Verständnis des Problems, daher widmen sich praktisch alle Phasen, abgesehen von der Ideenfindung, dem Verständnis des Nutzers und seines Problems (vgl. [MG13], S. 35). In der ersten Phase geht es darum, das Problem und sein Umfeld zu erfassen. Dazu gehört die richtige Aufgaben- bzw. Problemstellung, wie in Abschnitt 2.2.2 bereits erwähnt, werden im Design Thinking vor allem unklare Probleme (*wicked problems*) gelöst. Von daher ist die richtige Fragestellung entscheidend für den Erfolg: Bei einer zu vagen Fragestellung wird es schwierig ein konkretes Problem zu definieren, bei einem zu kleinen Problemraum werden möglicherweise nur bestehende Lösungen erweitert, anstatt neue Ideen hervorzubringen. Guertler und Meyer geben folgendes Beispiel: Die Frage „Wie können wir Menschen ehrlicher machen“ ist zu ungenau um tatsächlich eine Lösung zu finden, hingegen enthält die Fragestellung „Wie können wir Fahrradschlösser sicherer machen “ bereits einen Großteil der Lösung und wird keine Innovationen hervorbringen. Die Formulierung „Wie kann der Diebstahl von Fahrrädern in Großstädten reduziert werden?“ ist eine präzise Vision und bietet dennoch genügend Platz für neue Ideen (vgl. [MG13], S. 26.). Außerdem werden wie in der klassischen Projektplanung zeitliche und inhaltliche Prioritäten, sowie Maßstäbe zu späteren Bewertung des Erfolgs festgelegt. Aus der abschließenden Definition der Zielgruppe ergibt sich das Betrachtungsfeld für die nächste Phase (vgl. [Eng12], S. 30).

Beobachten: Diese Phase spielt eine wichtige Rolle für das weitere Vorgehen. Beim Beobachten sollen Eindrücke über die Nutzer gewonnen werden. Das Vorgehensmodell der HPI School of Design Thinking in Stanford fasst das Verstehen und Beobachten als *emphasize*

zusammen (vgl. [Lin13], S. 197f). Durch die Empathie, die für den Nutzer entwickelt wird, sollen die später entwickelten Lösungen genau den Bedürfnissen der Zielgruppe entsprechen. Nach Engchuan sollen die versteckten Bedürfnisse der Menschen durch Empathie antizipiert werden (vgl. [Eng12], S. 31.). Dabei werden qualitative Forschungsmethoden bevorzugt. So steht das persönliche Gespräch und die Beobachtung im Vordergrund. Konventionelle Marktforschungsmethoden, wie die Zielgruppe direkt nach ihren Bedürfnissen fragen, sind nicht zielführend, da sich die Nutzer oft ihrer Bedürfnisse nicht bewusst sind (vgl. [Eng12], S. 31.). Neben der Beobachtung der Zielgruppe, ist es sinnvoll selbst die Rolle des Anwenders einzunehmen um ein besseres Verständnis für die Situation und das Problem zu erlangen (vgl. [MG13], S. 43). In Bezug auf Abschnitt 2.3.2 stellt das Beobachten einen divergenten Denkprozess dar, bei dem möglichst viele Informationen gewonnen werden sollen. Diese werden analysiert und dokumentiert und dienen als Basis für die nächste Phase.

Standpunkt definieren: Diese Phase wird auch als *Storytelling* bezeichnet: Nachdem die Recherche in der Regel in Kleingruppen durchgeführt wird, trifft sich das gesamte Projektteam und trägt abwechselnd die Ergebnisse der vorherigen Phase vor (vgl. [MG13], S. 44). Die gewonnenen Erkenntnisse werden gemeinsam interpretiert und gewichtet, dabei wird am meisten Wert auf die Aspekte gelegt, die überraschend, erstaunend oder erschreckend waren (vgl. [MG13], S. 44.). Auf der so entstehenden Wissensbasis wird gemeinsam im Team entschieden, ob das Problem richtig verstanden wurde und die Beobachtungsphase abgeschlossenen wird oder ob eine weitere Iteration der vorgehenden Phase nötig ist (vgl. [MG13], S. 45). Sind genügend Informationen und Erkenntnisse vorhanden, wird von der Betrachtung des Problemraums nun zur Betrachtung des Lösungsraums übergegangen. Dazu wird ein gemeinsamer Standpunkt definiert, indem die konkreten Beobachtungen abstrahiert werden. In dieser Phase der Synthese werden nun aus den Beobachtungen Mustern abgeleitet. Eine Methode dazu ist der Entwurf einer Persona, diese Persona stellt einen idealtypischen, fiktiven Charakter dar, dessen Problem gelöst werden soll (vgl. [Eng12], S. 32). Neben der Persona existieren weitere Methoden um den Standpunkt zu definieren bzw. die Informationen sinnvoll zugliedern. Die folgenden Beispiele stellen einen Ausschnitt dar (vgl. [Eng12], S. 33; vgl. [MG13], S. 46ff):

- **Venn-Diagramme** dienen dazu überlappende Aspekte zu gruppieren
- **Storyboards** eigenen sich am besten um Informationen darzustellen, die auf „Geschichten“ z.B. einem Tagesablauf basieren

- **2x2 Matrizen** sind hilfreich um Zusammenhänge zwischen zwei Aspekten herzustellen, dazu werden zwei frei Aspekte gewählt und die gesammelten Information in ein 2-Achsen-Diagramm übertragen

Basierend auf dem Standpunkt werden neue Fragestellungen gebildet, die in der nächsten Phase gelöst werden sollen (vgl. [MG13], S. 49).

Ideen finden: In dieser divergierenden Phase geht es darum, in kurzer Zeit möglichst viele Ideen zu generieren, dabei steht die Quantität vor der Qualität (vgl. [Eng12], S. 33). Innerhalb dieser Phase können verschiedene Kreativitätstechniken angewendet werden, als bewährt gilt das Brainstorming. Die folgenden „Verhaltensregeln“ stellen dabei eine Hilfe dar, damit das Brainstorming ein Erfolg wird (vgl. [MG13], S. 53; vgl. Abschnitt 2.4.4):

- Ideen werden nicht bewertet oder diskutiert.
- Keine Idee ist zu verrückt.
- Ideen sollen visuell, nicht schriftlich vorgestellt werden.
- Die Ideen anderer sollen weiterentwickelt werden.

Wie bereits erwähnt findet keine Wertung statt, erst in der nächsten Phase werden die Ideen selektiert und optimiert (vgl. [Eng12], S. 33).

Prototypen entwickeln: In dieser Phase werden die generierten Ideen durch den Einsatz von Prototypen verfeinert. Dazu werden zunächst einige Ideen ausgewählt und diese im Team weiter erarbeitet. Mittels einfacher Prototypen werden die Ideen im Lösungsraum erprobt (vgl. [Eng12], S. 33). Diese Prototypen stellen keine voll funktionsfähigen aufwendigen Konstrukte dar, denn wie im Abschnitt 2.2.2 erläutert geht es darum, möglichst früh eine Idee auszuprobieren, um gegebenenfalls aus dem Misserfolg neue Erkenntnisse über die Lösung zu gewinnen. Prototypen können in Form von Diagrammen, Modellen, Storyboards, Rollenspielen oder klickbaren Anwendungen vorliegen (vgl. [Eng12], S. 34).

Testen/Verfeinern: In der nächsten Phase werden die entwickelten Prototypen gemeinsam mit der Zielgruppe getestet. Durch die Beobachtung werden neue Erkenntnisse über die Schwächen und Stärken einer Idee gewonnen. Diese werden in einer neuen Iteration weiter verfeinert (vgl. [Eng12], S. 34). In der Praxis hat sich gezeigt: Je unfertiger ein Prototyp zu seien scheint, desto ehrlicher ist das Feedback der Nutzer. Ebenso fällt es einem Team leichter einen Prototyp zu verwerfen, wenn in diesen noch nicht soviel Energie und Zeit investiert wurde (vgl. [MG13], S. 57).

Implementierung: Das sechs-stufige Vorgehensmodell der HPI School of Design Thinking sieht keine Phase der Implementierung vor, da „die Implementation von Design-Konzepten bei Unternehmen und Auftraggebern im Gegensatz zu kommerziellen Design-Agenturen nicht die primäre Zielsetzung ist“ ([Lin13], S. 200). Dennoch gibt es andere Vorgehensmodelle welche die Implementierung als Teil des Design Thinking betrachten (vgl. Abbildung 1), auf die in dieser Arbeit nicht weiter eingegangen wird. Gürtler und Meyer betonen, dass die erarbeiteten Lösungen „selbst nach mehrmaligen Iterationen noch Prototypen sind“ ([MG13], S. 58), deren wirtschaftliche und technische Umsetzung durch zusätzliche Prototypen, z. B. Businessmodells oder Pilotprojekte, zu prüfen seien (vgl. [MG13], S.59). Wie durch Design Thinking entwickelte Ideen in der Softwareentwicklung umgesetzt werden können, wird im folgenden Abschnitt 3 erläutert.

3. Design Thinking in der Software-Entwicklung

Nachdem in den vorherigen Schritten Design Thinking mit seinen wichtigsten Bestandteilen und das Vorgehensmodell nach der HPI School of Design Thinking vorgestellt wurde, widmet sich dieser Teil der Arbeit der Fragestellung ob und gegebenenfalls wie Design Thinking eine sinnvolle Ergänzung in der Software-Entwicklung darstellen kann. Dazu wird zunächst betrachtet wie Softwareentwicklung, insbesondere agile Entwicklungsmethoden, und Design Thinking miteinander vereinbar sind. Darauf folgend wird ein Praxisbeispiels von SAP SE betrachtet und daraus Erkenntnisse für die konkrete Integration von Design Thinking abgleitet.

3.1. Gemeinsamkeiten und Widersprüche von AGILE und Design Thinking

Für Raimer beantwortet Design Thinking die Frage nach dem „Was?“ mit bewerteten, prototypischen Lösungen. Die Umsetzung, die Frage nach dem „Wie“, ist durch andere Methoden zu beantworten (vgl. [Rai14], S. 184). Wie in Abschnitt 2.5.3 bereits erläutert, sieht das Vorgehensmodell keine Phase der Implementierung vor. Es stellt sich die Frage, wie Design Thinking und Softwareentwicklung konkret in Verbindung genutzt werden können. Betrachtet man hierzu verschiedenen Modelle der Softwareentwicklung wird schnell klar, dass Design Thinking nicht mit Standardansätzen wie dem Wasserfall-Modell vereinbar ist. Design Thinking als iterative Vorgehensweise, mit divergenten Denkprozessen, lässt sich nur schwer mit dem linearen, analytischen, durchgeplanten und auf dem Meilensteinen basierenden Ablauf des Wasserfall-Modells kombinieren (vgl. [LM10], S. 8). Hingegen wird

in agilen Entwicklungsmethoden (besonders in Scrum[1]) die Problembeschreibung nicht als vorheriger Schritt der Entwicklung angesehen, sondern die Entwicklung erfolgt parallel zur Problemlösung. Anstelle von strikten Projektplänen mit festen Meilensteinen setzen agile Entwicklungsmethoden auf klare Regeln und Rollen, die dem Projekt-Team ermöglichen flexibel genug auf neue Herausforderungen zu reagieren. Ebenso wird verstärkt auf die Zusammenarbeit z. B. in Form von Pairprogramming gesetzt (vgl. [LM10], S. 9). Es zeigen sich weitere Gemeinsamkeiten mit dem Design Thinking: Agile Entwicklung liefert eine vielfältige Auswahl von Techniken um während der Entwicklung in Kontakt mit dem Nutzer in Verbindung zu stehen. Bespiele hierfür sind der *Prodct Owner* und das *Product Backlog* in Scrum. Durch häufiges Feedback des Nutzers wird das Softwareprodukt in iterativen Entwicklungsdurchgängen (sog. *Sprints*) stetig verbessert. Agile Entwicklungsmethoden sind damit deutlich besser für den Einsatz von Design Thinking geeignet, jedoch existieren auch hier Widersprüche, die den Einsatz von Design Thinking erschweren: Zwar werden ebenso Prototypen eingesetzt, die schon zu Anfang nutzbar sind, wenn auch mit einem geringen Umfang an Funktionen. Diese werden jedoch inkrementell weiter entwickelt und hemmen somit radikale Innovationen sowie das divergente Denken (vgl. [LM10], S. 9). Somit beschränkt bereits der erste Prototyp das Endprodukt auf eine Reihe von Möglichkeiten. Agile Entwicklung ist somit flexibel was die Erweiterung von bestehen Prototypen angeht, da jedoch das Verwerfen der Prototypen nicht vorgesehen ist, ist es nicht möglich die gesamte Entwicklung in eine andere Richtung zu lenken (vgl. [LM10], S. 9). Es zeigt sich, dass agile Softwareentwicklung und Design Thinking einige Gemeinsamkeiten, jedoch auch Widersprüche aufweisen. Dass die Integration dennoch möglich ist und ins besondere wie, zeigt der folgende Abschnitt.

3.2. Praxisbeispiel der SAP SE

Der vorherige Abschnitt zeigte auf, welche Gemeinsamkeit und Unterschiede das Design Thinking und agile Softwareentwicklung haben. Dieser Abschnitt stellt nun ein Praxisbeispiel vor, bei dem Design Thinking und agile Entwicklungsmethoden zusammen genutzt wurden, um ein Informationssystem für das „Sailing Team Germany“ (STG) zu entwickeln [HM12]. Das Projekt gliederte sich in drei Projekt-Phasen, in denen letztendlich alle der in Abschnitt 2.5.3 beschrieben Phasen des Vorgehensmodells durchlaufen wurden:

- **Research:** Auf Grund der fehlenden Erfahrung des Teams, inklusive des *Product Owner*, war es wichtig eine ausgiebige Nutzerforschung zu betreiben, um sich mit

[1] Für ein besseres Verständnis von agilen Softwareentwicklungsmethoden, speziell Scrum, verweise ich an dieser Stelle auf die Seminararbeit von Dering [Der15]

der Domäne Segeln vertraut zu machen. Dazu beobachtete das Team jeweils in kleineren Gruppen innerhalb von mehreren *Sprints* Segler und ihre Trainer bei der Vorbereitung, bei Wettkämpfen und im Umgang mit den Trainern. Sie analysierten und verglichen die bisherigen Hilfsmittel (sowohl analoge als auch digitale) und führten Interviews mit den Seglern ihren Trainern (vgl. [HM12], S. 224f).

- **Synthesis:** In der zweiten Phase erarbeitete das Team mit Methoden aus dem Design Thinking eine erste Version des *Product Backlog*. Dazu wurden zunächst die gesammelten Informationen in der Gruppe geteilt. Innerhalb eines *Storytelling* wurden die Beobachtung, im Bezug auf die Bedürfnisse, erläutert. Anschließend wurden die Nutzerbedürfnisse gruppiert. Diese Gruppierung ergab sehr grobe Ansprüche, wie das Endprodukt auszusehen habe. Zum besseren Verständnis wurden Personas entwickelt, die die Software nutzen würden. Anhand der Personas und den gruppierten Bedürfnisses wurde eine *User-Story-Map* erarbeitet die letztendlich zu einem groben *Product Backlog* wurde (vgl. [HM12], S. 225ff).

- **Development:** In der dritten Phase begann die eigentliche Entwicklung der Software, dabei setze das Team auf Scrum, wie schon in der ersten Phase. Zunächst wurden simple Prototypen entwickelt und die *User Stories* mit dem Seglerteam abgesprochen. Während der Entwicklung verfeinerte sich durch das Feedback das *Product Backlog* immer weiter. Nach und nach wurden aus simplen Papier-Prototypen weiterentwickelte *High-Fidility-Prototyps* auf Basis von HTML5. Wenn eine *User Story* implementiert und ausgeliefert wurde, konnte das Segler-Team diese direkt testen und weiteres Feedback geben. In der Phase setzte das Team auch weiterhin auf Methoden des Design Thinking um schwierige oder unklare *User Stories* neu zu definieren (vgl. [HM12], S. 230).

Mit Hilfe von Design Thinking und agiler Softwareentwicklung entstand als Ergebnis „SAP Sail Better“, eine Cloud-basierte Wissensdatenbank, die es erlaubt, segelspezifisches Wissen für bestimmte Segelreviere und Wetterbedingungen zu speichern und zu recherchieren (vgl. [MG13], S.66).

3.2.1. Erkenntnisse für den praktischen Einsatz von Design Thinking

Aus dem Praxisbeispiel lassen sich mehrere Erkenntnisse für den Einsatz Design Thinking bei Softwareprojekten ableiten, welche im Folgenden erläutert werden:

- **Scrum als Prozess-Framwork:** Design Thinking liefert kein grundlegendes Prozessframework mit klaren Rollen und Artefakten, von daher ist es sinnvoll bereits

im Design Thinking auf Scrum zurückzugreifen. Bei unerfahrenen Teams kann der *Scrum-Master* als Coach agieren und das Team durch die Phasen leiten. Kurze *Sprints* eigenen sich als zeitlicher Rahmen um die Länge des Prozesses überschaubar zu halten. Das *Product Backlog* kann bereits während des Design Thinking mit konkreten *User Stories* und abstrakten Bedürfnissen gefüllt werden (vgl. [HGF13], S. 206f).

- **Fokus auf Endanwender:** Nicht nur bei der Erstellung des *Product Backlog* sondern bereits bei der Definition des Problems und der Erarbeitung der Lösung, sowie der letztendlichen Produkt-Vision sollte der Endanwender im Vordergrund stehen. Dabei stellen Interviews und Beobachtungen die beste Wahl dar. Es hat sich gezeigt, dass der Übergang zur Implementierung fließend ist, daher bietet es sich an auch während der Implementierung auf das Feedback der Anwender zurückzugreifen (vgl. [HGF13], S. 207).

- **Wegwerf-Protoypen:** Im Gegensatz zum Prinzip der Reduzierung von Verschwendung aus dem *Lean Software Development*, steht das Entwickeln von Wegwerf-Protoypen. Diese helfen die zahlreichen Ideen zu veranschaulichen. Frühe Prototypen mit der Zielgruppe zu testen verhindert außerdem Fehlentwicklungen, die nachträglich nur noch schwer zu bereinigen sind (vgl. [HGF13], S. 207f).

- ***User-Story-Maps* als Bindeglied:** *User-Story-Maps* dienen dazu um die Produkt-Vision in das *Product Backlog* umzuwandeln. Dabei werden mit Hilfe von Personas einzelne *User Stories* anhand des zugrunde liegenden Geschäftsprozessen strukturiert. Diese Methode kann ebenfalls dazu genutzt werden, um die Ergebnisse der ersten Phasen (Beobachten, Verstehen, Standpunkt definieren) in strukturierte Anforderung für die Implementierung zu überführen (vgl. [HGF13], S. 208).

Es zeigt sich somit, dass Design Thinking eine sinnvolle Ergänzung zur agilen Softwareentwicklungsmethode Scrum dargestellt.

4. Fazit

Design Thinking stellt einen alternativen Ansatz zur Problemlösung dar. Als nutzerzentrierte Problemlösestrategie wird der Problem(re)definition besonders viel Aufmerksamkeit gewidmet. Durch die iterative Vorgehensweise lassen sich unklare Problembeschreibungen auf wesentliche Bedürfnisse der Menschen reduzieren. Dazu setzt das Design Thinking als wesentliches Elemente auf abwechselnde Phasen des divergenten und konvergenten

Denkens in Kombination mit der Analyse und anschließender Synthese. Durch diese Merkmale ist es möglich, dass im Rahmen des Design Thinking Ideen hervorgebracht werden, die durch eine lineare rein analytische Vorgehensweise nie entstanden wären. Durch die frühzeitige Konzentration auf Wirtschaftlichkeit, Machbarkeit und Wünschbarkeit können aus diesen Ideen Innovationen hervorgehen. Die meisten Vorgehensmodelle, wie auch das Modell der HPI School of Design Thinking, betrachten dabei nur den Prozess zur Idee, nicht die Implementierung selbst. Für die Softwareentwicklung muss somit ein Übergang vom Design Thinking zur Implementierung geschaffen werden. Das Praxisbeispiel von SAP zeigt, dass mittels SCRUM als Prozessframework und der Methode der *User-Story-Map* als Schnittstelle, der Übergang erfolgreich sein kann.

4.1. Design Thinking in der Projektgruppe IMPACT

Dieser Abschnitt widmet sich letztendlich der Frage wie Design Thinking in der Projektgruppe genutzt werden kann, dazu wird zunächst geprüft, ob die Aufgabenstellung der Projektgruppe sowie das Team für das Design Thinking geeignet sind. In Abschnitt 2.2.2 wurde erläutert, das Design Thinking vor allem für *ill-structured* und *wicked problems* geeignet sei. Die Aufgabenstellung der Projektgruppe:

> „Schaffung einer web-basierten Plattform zur Unterstützung innerbetrieblicher Innovationsprozesse“ (siehe [imp15], S. 4)

stellt auf Grund der offenen Nebenbedingungen (Anreizsystem, Einbeziehung der Mitarbeiter, Entwicklung eines Wokflows) (vgl. [imp15], S. 5f) ein *ill-structered problem* dar. Nach der HPI School of Design Thinking ist für das Design Thinking ein interdisziplinäres Team nötig oder zumindest von Vorteil (vgl. Abschnitt 2.4.2). Interdisziplinarität kann das Team als Gruppe von Informatik- bzw. Wirtschaftsinformatik-Studenten nicht vorweisen, allerdings hat sich in dem Praxisbeispiel gezeigt, dass dies nicht von Nöten sein muss. Viel mehr ist eine Betrachtung des Problems aus verschiedenen Perspektiven von Vorteil, diese kann durch die Bildung von Kleingruppen mit unterschiedlichen Schwerpunkten, z.B. Fronted-Design, Gamification, Workflow, erreicht werden. Ebenso lässt sich durch Einsatz des Moderationkoffers der Abteilung VLBA, sowie deren Räume mit Whiteboards und Tafeln die letzte Voraussetzung der offen Raumkonzepte umsetzen.

Nachdem gezeigt wurde, dass das Problem sowie die Projektgruppe prinzipiell für Design Thinking geeignet sei, werden nun beispielhaft einige Vorschläge geliefert, wie die Elemente des Design Thinking umgesetzt werden können. Es empfiehlt sich dabei das Praxisbeispiel von SAP zur Orientierung zu nutzen. Demnach sollte Design Thinking in

Kombination mit SCRUM dazu genutzt werden um eine klare Produktvision zu entwickeln, bei dem die Endanwender im Fokus stehen. Dazu bietet es sich an Interviews mit den späteren Nutzern zu führen und diese eine Zeit lang in ihrer Arbeitsumgebung und Zusammenarbeit miteinander zu beobachten um zu verstehen, wie das Endprodukt optimal eingesetzt werden kann. In einem nächsten Schritt sollten dazu Personas abgleitet werden, für diese die Software entwickelt wird. Es bietet sich dabei an verschiedene Hierarchieebenen zu betrachten z.B. Mitarbeiter, Projektleiter und Vorstand um dabei ein möglichst breites Feld von Anwendern abzudecken. Mit Hilfe der Personas und den bereits erwähnten *User-Story-Maps* können die im Rahmen des Design Thinking entwickelten Ideen in konkret zu implementierende Funktionen umgewandelt werden. Der Einsatz von Prototypen vor und während der Entwicklung in Form von *Low-Fidility-Protoyps* aus Papier als Verdeutlichung des Konzepts, sowie klickbaren Softwareprototypen innerhalb von Tests mit der Zielgruppe wird ausdrücklich empfohlen. Diese Tests stellen zwar einen Mehraufwand dar, jedoch wird dadurch wertvolles Feedback geliefert und es kann somit frühzeitig eine Fehlentwicklung des Projekts verhindert werden. Bei unklaren oder nicht umsetzbaren *User-Stories* empfiehlt es sich, während der Implementierungsphase auf die Methoden des Design Thinking zurückzugreifen um diese neu zu definieren. Insgesamt ist anzumerken, dass der Einsatz des Design Thinking zu empfehlen ist, da dessen Mehraufwand durch die genannten potentiellen Mehrwerte aufgewogen wird.

A. Anhang

Literatur

[Bro09] T. Brown. *Change by Design.* Harper Business, 2009.

[Der15] J.-V. Dering. Softwareentwicklungs-Vorgehensmodell, 2015.

[Eng12] R. Engchuan. Design Thinking in der internationalen Entwicklungszusammenarbeit, Juni 2012.

[ER13] J. Erbeldinger and T. Ramge. *Durch die Decke denken.* Redline Wirtschaft, 2013.

[HGF13] T. Hildenbrand, J. Gürtler, and M. Fassunge. *Scrum als Framework für die Produktinnovation*, pages 191–212. Symposion Publishing GmbH, 2013.

[HM12] T. Hildenbrand and J. Meyer. Intertwining lean and design thinking: software product development from empathy to shipment. In *Software for People.* Springer, 2012.

[hpi] Design Thinking – Mindset. HPI School of Design Thinking: `http://hpi.de/school-of-design-thinking/design-thinking/mindset.html`. Zuletzt besucht am 05.08.2015.

[hpi07] Design Thinking. Online-Flyer zur HPI School of Design Potsdam: `http://www.hpi.uni-potsdam.de/fileadmin/hpi/d-school/presse/material/HPI-D-School_Flyer.pdf`, 2007. Zuletzt besucht am 05.08.2015.

[imp15] iEIMS - Intraenterprise innovation management system. Foliensatz zum Kickoff-Meeting, 2015.

[Lin13] T. Lindberg. Design-Thinking-Diskurse, 2013.

[LM10] T. Lindberg and C. Meinel. Design Thinking in IT Development? In *Electronic Collequium in Design Thinking Research, Report*, number 1, 2010.

[MG13] J. Meyer and J. Gürtler. *30 Minuten Design Thinking.* GABAL Verlag GmbH, 2013.

[MPD14] T. Müller-Prothmann and N. Dörr. *Innovationsmanagement.* Carl Hanser Verlag GmbH Co KG, 2014.

[pfe12] *Querdenken im Team*, pages 247–252. Springer, 2012.

[PML10] H. Plattner, C. Meinel, and L. Leifer. *Design Thinking: Understand – Improve – Apply*. Understanding Innovation. Springer Berlin Heidelberg, 2010.

[PMW09] H. Plattner, C. Meinel, and U. Weinberg. *Design Thinking*. Hasso Plattner Institut, 2009.

[RA12] G. Reger and S. Adelheim. Innovationsmangement – Bedeutung von Innovation. Enzyklopädie der Wirtschaftsinformatik Online-Lexikon: `http://www.enzyklopaedie-der-wirtschaftsinformatik.de/wi-enzyklopaedie/lexikon/daten-wissen/Wissensmanagement/Wissensmanagement--Konzepte-des/Innovationsmanagement`, 2012. Zuletzt besucht am 05.08.2015.

[Rai14] S. Raimer. *Design Thinking, Lean, Agile: Schlüssel zur Innovation*. Symposion Publishing GmbH, 2014.

[Row91] P. Rowe. *Design thinking*. MIT press, Cambridge, 1991.

[SAP12] Design Thinking at SAP. SAP `https://experience.sap.com/basics/post-48/`, 2012. Zuletzt besucht am 08.08.2015.

[Sim69] H. Simon. *The sciences of the artificial*. MIT Press,, Cambridge, 1969.

[Tis09] L. Tischler. IDEO'S David Kelly on "Design Thinking". Fast Company: `http://www.fastcompany.com/1139331/ideos-david-kelley-design-thinking`, 2009. Zuletzt besucht am 05.08.2015.

[TS10] Maria Tagwerker-Sturm. Design Thinking als innovative Entwicklungsmethode. www.inknowaction.com: urlhttp://www.inknowaction.com/blog/2010/06/12/design-thinking-als-innovative-entwicklungsmethode/, 2010. Zuletzt besucht am 12.08.15.

[Win11] M. Winkler. Innovative Teams im Design Thinking, 2011.

B.1.3. Entwicklungsstufen des Innovationsmanagements

Thema:

Entwicklungsstufen des Innovationsmanagements:
von der Idee bis zum Produkt

Schriftliche Ausarbeitung im Rahmen des Projekts IMPACT

Abteilung Wirtschaftsinformatik 1:
Very Large Business Applications

Betreuer: Jens Siewert
Stefan Wunderlich
Dr. Joachim Kurzhöfer

vorgelegt von: Jessica Schulte
Unnerloogsweg 22
26817 Rhauderfehn
04952/9524644
E-Mail: jessica.schulte@uni-oldenburg.de

Abgabetermin: 22. Juni 2015

Inhaltsverzeichnis

1 Einleitung

Innovationsmanagement ist die Planung, Steuerung und Kontrolle von der Generierung einer Idee bis hin zum fertigen Produkt, dass aus dieser Innovation entstanden ist. Wichtigkeit gewinnt das Innovationsmanagement wegen einer immer schneller veränderten Marktsituation. Möchte ein Unternehmen in einem solchen Markt bestehen, so muss es innovativ sein. Innovationsmanagement unterstützt dabei bei der systematischen Erfassung von Ideen, deren Weiterentwicklung und der letztendlichen Umsetzung. Hierfür gibt es eine Vielzahl von Innovationsprozessmodellen, die Prozesse und Handlungsempfehlungen innerhalb des Innovationsmanagement abbilden. Gemeinsamkeit aller Modelle ist, dass sie den Prozess von der Idee bis hin zum Produkt in mehrere Entwicklungsstufen abbilden. Die unstrukturierte erste Idee wird dabei innerhalb der Stufen immer mehr verfeinert. Aufgrund der Vielzahl an Modellen, müssen die Schwerpunkte und die Vor- und Nachteile in Bezug auf das Anwendungsgebiet herausgearbeitet werden und anschließend an das jeweilige Einsatzgebiet angepasst werden.

Bisher gibt es kein strukturiertes Vorgehen innerhalb des zu entwickelnden Innovationsmanagementportals. Um dieses zu ermöglichen, werden verschiedene Innovationsprozessmodelle untersucht. Insgesamt werden drei Modelle näher betrachtet. Das Stage Gate Modell von Cooper, das sechs Phasen definiert, wird vorgestellt. Des Weiteren wird der Value Proposition Cycle von Hughes gezeigt, mit dem eine Idee in mehreren Zyklen abgearbeitet werden kann. Außerdem wird das Modell von Vahs und Brem dargestellt, das Ideen mit Hilfe einer umfangreichen Ideenfindung generiert. Dabei werden deren Besonderheiten, sowie Vorteile und Nachteile bezüglich eines webbasierten Innovationsmanagementportals aufgezeigt.

Die Innovationsprozessmodelle werden jeweils im Ganzen dargestellt. Da aber für das Portal die Prozesse von der Idee bis zum Business Case am meisten Bedeutung haben, wird der Fokus in dieser Arbeit auf diese Phasen gelegt. Die darauf folgenden werden nur aufgrund der Vollständigkeit kurz erwähnt. Es wird deutlich welche Funktionen die einzelnen Prozesse in den verschiedenen Modellen haben.

Aus den Ergebnissen der Untersuchung kann anschließend ein Prozess für ein webbasiertes Innovationsmanagementportal entwickelt werden.

2 Innovationsprozessmodelle

In diesem Kapitel werden die Innovationsprozessmodelle vorgestellt. Zunächst gibt es zu diesen eine allgemeine Beschreibung. Anschließend werden drei verschiedene Innovationsprozessmodelle vorgestellt wobei deren Vor- und Nachteile gegenübergestellt werden.

Zuerst wird das Stage Gate Modell von Cooper, das sechs Phasen definiert vorgestellt. Des Weiteren wird der Value Proposition Cycle von Hughes, mit dem eine Idee in mehreren Zyklen abgearbeitet wird, gezeigt. Außerdem wird das Modell von Vahs und Brem dargestellt, das Ideen mit Hilfe einer umfangreichen Phase der Ideenfindung generiert.

Die Schwerpunkte werden hier auf die Phasen bis zur Erstellung eines Business Case gesetzt, da diese für die spätere Entwicklung eines Innovationsmanagementportals von höherer Bedeutung sind. Die nachgelagerten Prozessschritte finden größtenteils außerhalb des Innovationsmanagementportals statt und werden daher nur kurz zur Vollständigkeit erwähnt.

2.1 Allgemeines

Die Entwicklungsstufen des Innovationsmanagements können in einem Innovationsprozessmodell abgebildet werden. Dieses beinhaltet die Abfolge aller Aktivitäten, die zu einer Innovation führen. Es gibt verschiedene Modelle, in denen der Innovationsprozess jeweils unterschiedlich definiert wird. Sie dienen allgemein alle der Standardisierung von real ablaufenden Innovationsprozessen (Vgl. Verworn/Herstatt(2000), S. 1). Gemeinsamkeit aller Modelle ist, dass die den Prozess von der Entstehung einer Idee bis hin zum Produkt abbilden. Sie unterscheiden sich nur im Detailierungsgrad der einzelnen Stufen und in deren Schwerpunktsetzung. (Vgl. Vahs/Brem(2015), S. 230)

Die Aufgabe, die jedes Modell erfüllen muss, ist die Sicherstellung der einzelnen Schritte von der Initiierung einer Idee bis hin zur Markteinführung (Vgl. Vahs/Brem(2015), S.229). Diese Aufgabe hat zum Ziel, dass Ideen systematisch abgearbeitet werden und das größtmögliche Potential aus diesen geschöpft werden kann.

Als Hintergrund für die Entstehung von Innovationsprozessmodellen können empirische Untersuchungen herangezogen werden, die ergaben, dass Unternehmen mit einem standardisierten Innovationsprozess erfolgreicher sind als andere Unternehmen ohne einen standardisierten Innovationsprozess. (Vgl. Vahs/Brem(2015), S. 235)

2.2 Stage Gate Modell

Das Stage Gate Modell beinhaltet einen Prozess zur praktischen Gestaltung der Produktentstehung von der Idee bis zum marktfähigen Produkt. Hiermit kann kontinuierlich der Projektfortschritt überprüft werden. (Vgl. Institut für Technologie und Arbeit (o. J.), S. 5) Bei den Prozessbestandteilen wird zwischen Phasen und Kontrollpunkten, auch Gates genannt, unterschieden. Es gibt drei Generationen des Stage Gate Prozesses. Er wird jeweils mit einer Idee angestoßen. Dieses Modell wird vorgestellt, weil es das weitverbreitetste ist und von vielen größeren Unternehmen (wie IBM, 3M, General Motors) jeweils in einer angepassten, auf das Unternehmen zugeschnittene Form, angewendet wird. (Vgl. Verworn/Herstatt(2000), S. 4)

Besonderheit dieses Modells ist, dass nach jeder Phase im Kontrollpunkt entschieden wird, ob die Innovation den zuvor festgelegten Anforderungen bezüglich Kosten, Zeit und Markt entspricht. Hier wird dann über Fortführung oder Abbruch des Innovationsprojekts entschieden. Dazu wird in der Regel ein Gremium aus verschiedenen Fachkräften und Managern bestimmt, das die Bewertung des Phasenergebnisses vornimmt. (Vgl. Heesen(2009), S. 72) Die ersten beiden Generationen werden im Folgenden vorgestellt.

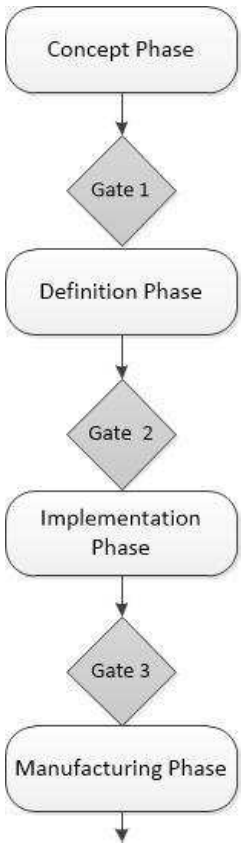


Abbildung 2.1 Stage Gate Modell der ersten Generation (Vgl. Verworn/Herstatt(2000), S.3)

Das erste Stage Gate Modell (siehe Abbildung 2.1. Stage Gate Modell der ersten Generation), auch Phase Review Modell genannt, wurde 1960 von der NASA entwickelt. Es wird in vier

Phasen geteilt. Die jeweils nächste Phase beginnt erst nach vollständigen Abschluss der vorgelagerten Phase.

In der ersten Phase, der „Concept Phase“, wird eine Innovation generiert. In der darauffolgenden „Definition Phase“ wird das Projekt definiert. Die „Implemantation Phase“ beinhaltet die Umsetzung der Innovation als Prototypen. In der „Manufacturing Phase“ wird die Innovation in die Produktion überführt.

Ein Nachteil bei diesem Modell ist die geringe Flexibilität. Diese entsteht dadurch, dass die nachgelagerten Phasen so lange ausgesetzt werden, bis eine Entscheidung über die vorgelagerte Phase getroffen wurde. Der Prozess wird dadurch verlangsamt. Ein weiterer Nachteil ist die ausschließlich technische Ausrichtung des Prozesses. Die Aufgaben des Marketings fehlen ebenfalls. (Vgl. Verworn/Herstatt(2000), S. 4)

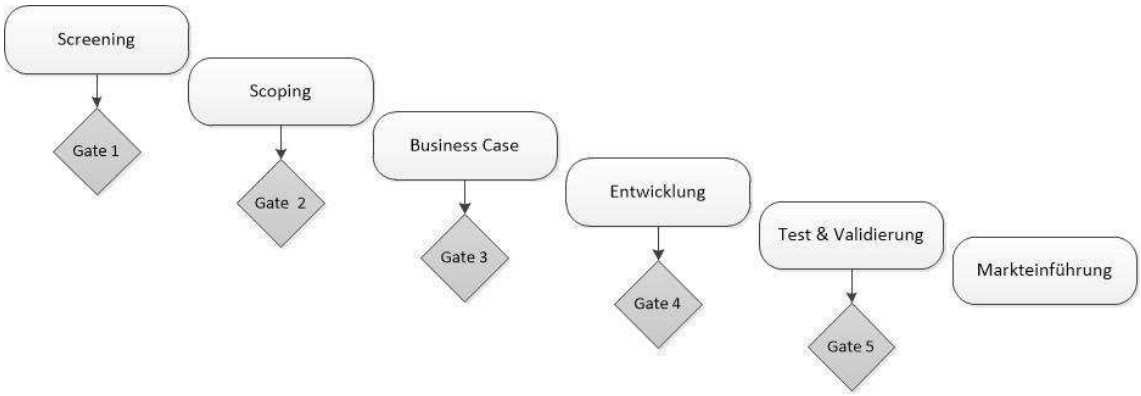


Abbildung 2.2 Stage Gate Modell der zweiten Generation
(Vgl. Müller-Prothmann /Dörr(2009), S. 25)

Im Stage Gate Modell der zweiten Generation, das von Cooper u.a. entwickelt wurde (siehe Abbildung 2.2 Stage Gate Modell der zweiten Generation), beginnt die nächste Phase schon vor Abschluss der vorgelagerten Phase. Die Kontrollpunkte sind hier flexibler gestaltet. Der Prozess wird dadurch beschleunigt. Es beinhaltet außerdem auch die Betrachtung des Marktes durch die Business Case Erstellung und die technische Realisierbarkeit unter anderem durch „Gate 1“.

Die einzelnen Phasen werden folgend näher beschrieben (Vgl. Cooper(2002), S. 149ff):

In der ersten Phase „Screening“ werden zunächst Ideen gefunden und erfasst. Daraufhin werden im „Gate 1“ die guten Ideen nach bestimmten Kriterien ausgesiebt. Diese Kriterien könnten technische Machbarkeit, Redundanz und Marktattraktivität sein. Es sind aber auch andere Kriterien möglich. Diese müssen bei der Implementierung des Prozesses definiert werden. Die Entscheidung über Ideen fällt in dieser Phase weniger streng aus, um einen Kreativitätsprozess in Gang zu halten.

In der zweiten Phase „Scoping“ werden Aufgaben und Untersuchungszusammenhänge definiert. Es wird untersucht, welche konkreten Vorteile die Idee bezüglich des Marktes bringt. Hierauf folgt dann im „Gate 2“ eine Überprüfung dieser Vorteile. Außerdem werden die Kriterien aus der ersten Phase erneut überprüft und eventuell erweitert.

In der dritten Phase „Business Case“ wird der Rahmen des aus Idee entstehenden Projektes abgesteckt. Es findet eine umfangreiche Marktforschung und eine Finanzanalyse statt.

Bevor die Phase der Entwicklung beginnt wird das Innovationsprojekt anhand des erstellten Business Case im „Gate 3“ untersucht. Dabei werden sowohl die Qualität als auch die Ergebnisse des Business Case herangezogen.

Ein Vorteil des Stage Gate Modells ist, dass es feste Stufen, also fest definierte Phasen besitzt, in denen sich der Prozess einteilen lässt. Es ist transparent für alle Mitarbeiter und bezieht verschiedene Unternehmensbereiche mit ein. Also es arbeiten interdisziplinäre Teams miteinander (Vgl. Müller/Graubner(2008), S. 13). Außerdem ermöglichen die Kontrollpunkte die frühe Erkennung von Misserfolgen. Somit werden auch Qualitätskontrollen durchgeführt (Vgl. Verworn/Herstatt(2000), S. 3).

Ein Nachteil ist, der sequentielle Ablauf der Phasen. Paralleles Arbeiten an einzelnen Phasen ist nur eingeschränkt möglich. Die Durchführung ist sehr zeitintensiv, da die Überprüfung in den Kontrollpunkten viel Zeit in Anspruch nimmt. Als Voraussetzung muss die Motivation der Mitarbeiter zum Einreichen von Ideen und das richtige Verständnis vom Prozess schon gegeben sein, um Erfolge zu erzielen. (Vgl. Müller/Graubner(2008), S. 13)

2.3 Value Proposition Cycle

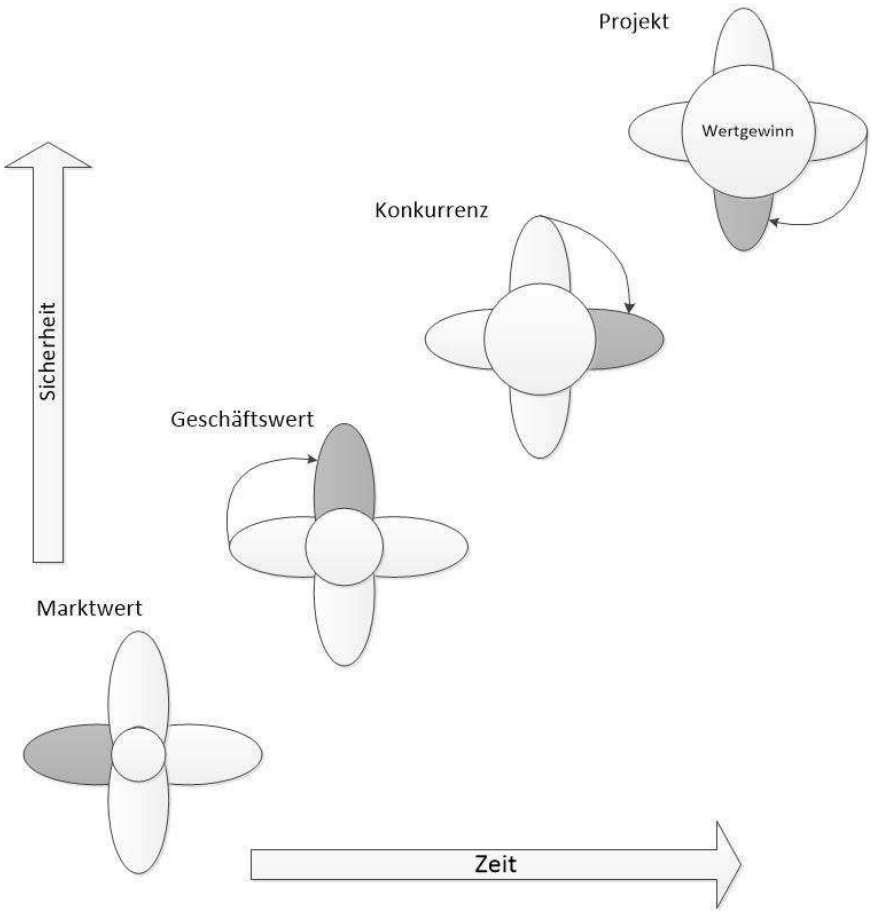


Abbildung 2.3 Value Proposition Cycle nach der Ideenfindung
(Vgl. Hughes/Chafin(1996), S. 93)

Der Value Proposition Cycle (siehe Abbildung 2.3 Value Proposition Cycle nach der Ideenfindung) baut auf das Stage Modell auf und beschäftigt sich damit, die Geschwindigkeit des Prozesses zu optimieren. Es bezieht sich vor allem auf die Bewertung der Ideen in den Kontrollpunkten nach jeder absolvierten Stufe. Besonderheiten sind die zyklischen Elemente. In den vier Zyklen ist jeweils der Wert einer Innovation für das Unternehmen, der Markt, die Wettbewerbsfähigkeit, sowie die Machbarkeit zu untersuchen. (Vgl. SpinnRaum (2014))

Durch kontinuierliches Lernen innerhalb der einzelnen Zyklen, Konsensbildung und Fokussierung auf den zusätzlichen Wert für Kunden soll die Effizienz und Effektivität des Innovationsprozesses erhöht werden (Vgl. Verworn/Herstatt(2000), S. 6).

Dieses Modell wird beschrieben, weil es eine Idee an vier Fragen abarbeitet und es sich insbesondere in den Phasen der Ideenfindung als auch der Ideenbewertung, also dem eigentlichen Nutzenversprechen der Idee eignet.

Nach der Ideenfindung können die folgenden Fragen in den einzelnen Zyklen gestellt werden:

1. Wie ist der Marktwert der Idee?
2. Wie ist der Geschäftswert der Idee?
3. Was macht die Konkurrenz?
4. Können wir die Idee umsetzen?

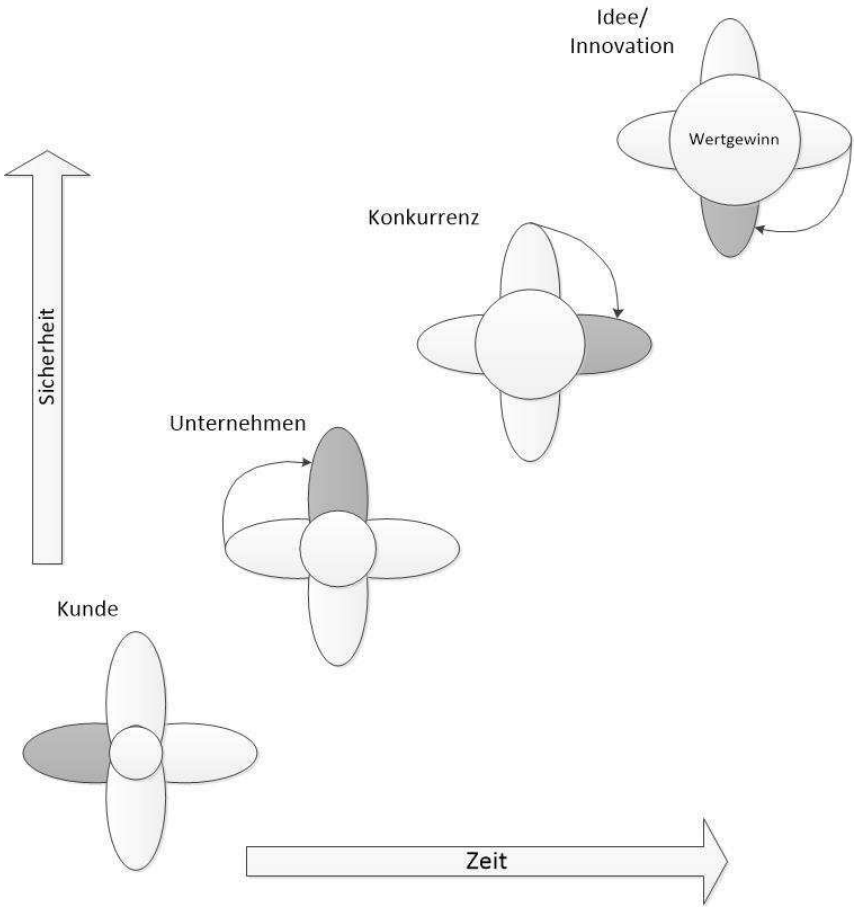


Abbildung 2.4 Value Proposition Cycle zur Ideenfindung
(Vgl. Hughes/Chafin(1996), S. 93)

Zur Ideenfindung können folgenden Fragen gestellt werden (siehe Abbildung 2.4 Value Proposition Cycle zur Ideenfindung):

1. Was wollen die Kunden?
2. Was bietet das Unternehmen, um Kundenwünsche zu erfüllen?
3. Was macht die Konkurrenz, um Kundenwünsche zu erfüllen?
4. Was können wir machen, um Kundenwünsche zu erfüllen?

Der Vorteil des Value Proposition Cycle ist zum Einen, dass es sich im Unterschied zum Stage Gate Modell um einen iterativen Prozess handelt. Der Zyklus wird kontinuierlich durchlaufen.

Dadurch wird ein gleichzeitiges kontinuierliches Lernen ermöglicht. (Vgl. Müller/Graubner(2008), S. 21) Es lässt dadurch auch die erneute Untersuchung einer bereits archivierten Idee zu.

Dem entgegengestellt ist der Nachteil, dass es sich um ein abstraktes Modell handelt. Dadurch werden keine genauen Handlungsempfehlungen aufgezeigt, sondern es dient nur als eine Art Regelwerk, anhand dessen die Ideen abgearbeitet werden können, um sie zu generieren oder zu bewerten. (Vgl. Müller/Graubner(2008), S. 21)

2.4 Innovationsprozesse nach Vahs/Brem

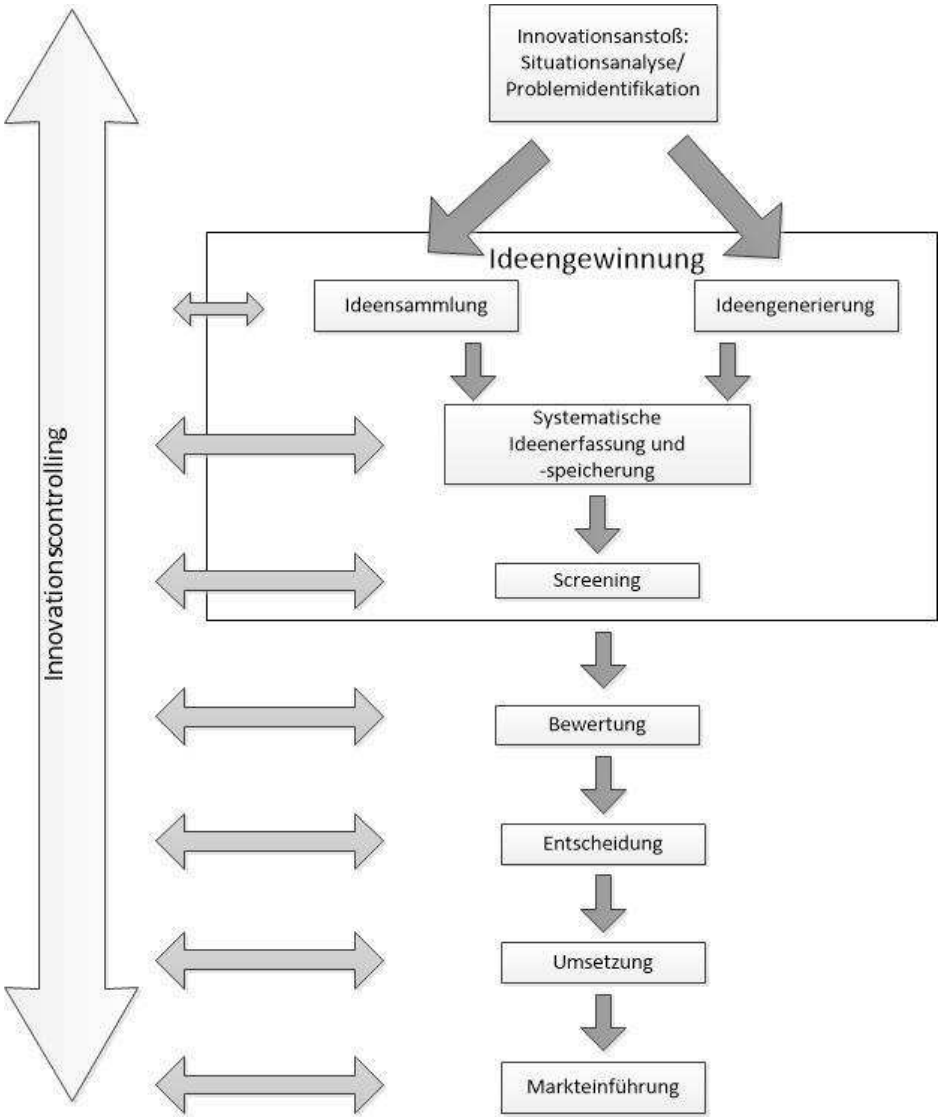


Abbildung 2.5: Innovationsprozessmodell nach Vahs und Brem (2015), S.230

Das dritte Modell, das vorgestellt wird, ist das Innovationsprozessmodell nach Vahs und Brem (siehe Abbildung 2.5: Innovationsprozessmodell nach Vahs und Brem). Dieses wird

beschrieben, weil es von Vahs und Brem als idealtypisches Modell vorgestellt wird. Es versucht die Konzepte mehrerer anderer Modelle zu vereinen (Vgl. Vahs/Brem(2015), S. 230). Unter anderem besitzt es einen ähnlichen Ablauf wie das Stage Gate Modell, nur der Detaillierungsgrad der einzelnen Stufen ist höher.

Das Modell besteht aus sechs verschiedenen Phasen.

1. **Innovationsanstoß oder Situationsanalyse/Problemidentifikation**: In der Situationsanalyse werden das Umfeld und die Entwicklungstendenzen des Unternehmens herausgestellt. Auf Basis dessen werden Probleme identifiziert. Diese können auftreten, wenn sich Kundenbedürfnisse ändern oder Konkurrenz-, Markt- und Technologieentwicklungen sowie Entwicklungen im eigenen Unternehmen stattgefunden haben. Der Ist-Zustand und der Soll-Zustand stimmen dann nicht mehr überein. (Vgl. Vahs/Brem(2015), S. 231)

2. **Ideengewinnung**: Auf Basis der Problemidentifikationen findet eine Ideengewinnung statt. Die Ideen sind damit Problemlösungsvorschläge.

2.1. **Ideensammlung**: Ideen können auch ohne vorherige Problembeschreibung eingereicht werden. Diese können nicht nur von den Mitarbeitern, sondern beispielsweise auch von Kunden oder anderen Stakeholdern stammen. In der Phase der Ideensammlung werden diese Ideen untersucht und dem Problem zugeordnet.

2.2. **Ideengenerierung**: Parallel zu der Phase der Ideensammlung kann die Ideengenerierung stattfinden. Hier werden mit verschiedenen Kreativitätsmethoden oder Kreativitätsworkshops neue Ideen mit Bezug auf das Problem generiert.

2.3. **Systematische Ideenerfassung und -speicherung**: Eine systematische Ideenerfassung und -speicherung ist notwendig, um Ideen weiter bearbeiten und vergleichen zu können. In dieser Phase wird auch untersucht, ob die Ideen zur Problemlösung beitragen oder eventuell einem anderen Problem zugeordnet werden können. Auch wenn keines zu dieser Idee zugeordnet werden kann, wird diese zunächst gespeichert.

2.4. **Screening**: In der Phase „Screening“ wird eine suchfeldorientierte Selektion durchgeführt. Diese dient dazu, dass das eigentliche Problem, dass am Anfang des Prozesses definiert wurde, nicht aus der Betrachtung fällt. Das Suchfeld entspricht also den im Problem erfassten Gegenstand, beispielsweise einem Produkt. (Vgl. Vahs/Brem(2015), S. 232f)

3. **Bewertung**: Der Bewertungsprozess sollte sehr sorgfältig erfolgen. Die Bewertung erfolgt auf Managementebene. Ein Gremium aus verschiedenen Fachkräften setzt sich zur Bewertung zusammen. Die Ideen können mit Hilfe verschiedener Methoden

bewertet werden. Eine Methode ist zum Beispiel das Erstellen eines Business Case. Zusätzlich kann auch eine Bewertung durch Mitarbeiter mit Hilfe von Online Portalen durchgeführt werden, in denen Mitarbeiter Punkte für Ideen vergeben können. (Vgl. Vahs/Brem(2015), S. 233)

4. **Auswahl**: Nach der Bewertung der Ideen findet die Auswahl auf Basis der Ergebnisse der vorherigen Phase statt. Die Auswahl kann vom selben Gremium übernommen werden, das die Bewertung der Ideen vorgenommen hat. Diese Auswahl wird dann dem Top-Management vorgelegt, das schließlich über die Umsetzung entscheidet.
   Ideen, die nicht ausgewählt werden, werden für eine eventuell spätere erneute Bewertung archiviert. (Vgl. Vahs/Brem(2015), S. 233)

Anschließend folgt die fünfte Phase der **Umsetzung**, in der eine ausgewählte Idee umgesetzt wird. Die sechste Phase ist die **Markteinführung**, in der das erstellte Produkt eingeführt wird.

Ein Vorteil dieses Modells ist die detaillierte Aufteilung der Phase der Ideengewinnung. Eine weitere Besonderheit ist das Innovationscontrolling, das bei der Planung, Steuerung und Kontrolle der Aktivitäten unterstützt. Außerdem basiert die Ideengewinnung auf eine bestimmte Problemstellung.

Dem gegenübergestellt ist der Nachteil, dass es keine Rückkopplungen zu vorgelagerten Prozessschritten gibt. Der Prozess ist also streng sequentiell. Somit ist kein paralleles Durchlaufen der Phasen möglich. Die Phasen werden außerdem nicht durch Meilensteine abgegrenzt, wie es beim Stage Gate Modell der Fall ist. (Vgl. Heesen(2009), S. 70f)

3 Fazit

In dieser Ausarbeitung sind drei Innovationsmanagementmodelle vorgestellt worden. Es gibt darüber hinaus noch eine Vielzahl weiterer Modelle, die diesen aber ähnlich sind. Sie unterscheiden sich meist nur durch die unterschiedliche Setzung von Schwerpunkten bezüglich der einzelnen Phasen. So wird der Schwerpunkt beim Stage-Gate Modell von Cooper auf die Kontrollpunkte nach jeder Phase gesetzt. Beim Value Proposition Cycle wird eine iterative Vorgehensweise vorgestellt. Das Modell von Vahs und Brem ist besonders auf die Ideengewinnung und der Bewertung von Ideen ausgerichtet.

Innovationsprozessmodelle ermöglichen es, einen strukturierten Prozess für das Innovationsmanagementportal zu entwickeln. Ein erstes Beispiel, wie ein solcher Prozess aussehen kann, befindet sich im Anhang dieser Seminararbeit.

Neben der Auswahl der idealen Phasen bei der Erstellung des Prozesses werden die Ausgestaltung der Übergänge zu den einzelnen Phasen, die Verteilung von Zuständigkeiten für einzelne Phasen und der Prozess der Ideengewinnung sein. Bei der Bewertung und Auswahl von Ideen müssen eventuelle Rückkopplungen zu vorherigen Prozessschritten beachtet werden.

Anhang

A Innovationsprozess für den Anwendungsfall eines IT Unternehmens

Aus den Erkenntnissen dieser Ausarbeitung kann ein erster Entwurf eines Innovationsprozesses für ein webbasiertes Innovationsmanagementportal erstellt werden. Der folgende Prozess beinhaltet die Phasen Problemerfassung, Ideengenerierung und Ideenerfassung, Ideenprüfung und die Erstellung des Business Case.

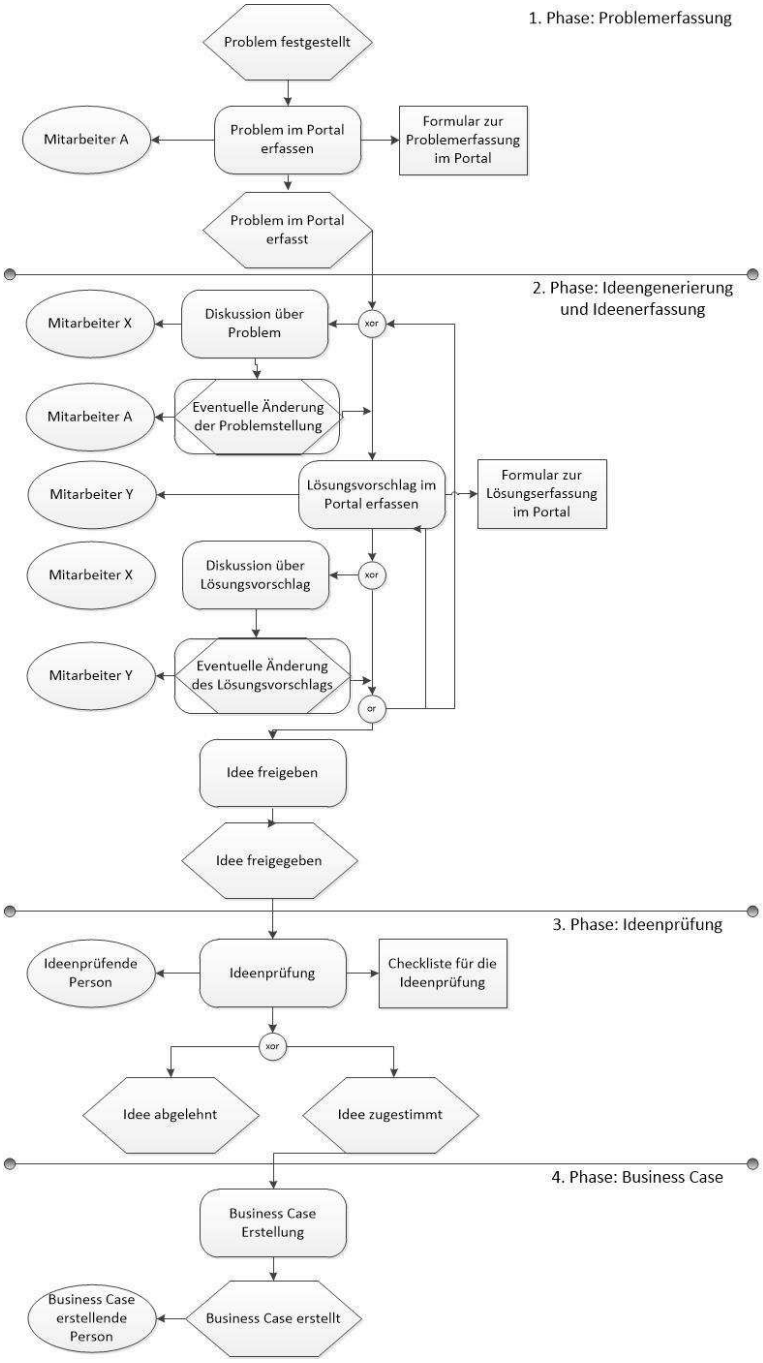


Abbildung A.1 Beispiel eines Prozesses für ein Innovationsmanagementportal

1. Phase: Problemerfassung
   Der Prozess beginnt mit einem Problem, dass im Portal erfasst wird.
   Das Formular zur Problemerfassung könnte beispielsweise den Titel, die erfassende Person, das Datum, eine Kategorie des Problems (nach Dringlichkeit oder Unternehmensbereichen oder nach Produkten), den Ist-Zustand bezüglich des Problems und den Soll-Zustand bezüglich des Problems enthalten. Das Problem wird dabei von einem Mitarbeiter erfasst. Dieser muss nicht gleichzeitig eine Lösung, also eine Idee zum Lösen des Problems parat haben.
2. Phase: Ideengenerierung und Ideenerfassung
   Die zweite Phase ist die Ideengenerierung und die Ideenerfassung. Eine Möglichkeit ist, dass Mitarbeiter X Diskussionsbedarf über das Problem sieht und daher eine Diskussion startet. Eventuell ändert Mitarbeiter A daraufhin nochmals die Problembeschreibung.

   Gleichzeitig kann ein Mitarbeiter Y schon den ersten Lösungsvorschlag für das Problem einreichen. Dieser Mitarbeiter Y kann auch Mitarbeiter A oder X sein. Über diesen Lösungsvorschlag kann dann wieder diskutiert werden und eventuelle Anpassungen an den Lösungsvorschlag gemacht werden. Es können dann anschließend noch weitere Diskussionen stattfinden und neue Lösungsvorschläge eingereicht werden.

   Nach einem bestimmten Ereignis, dass noch nicht weiter festgelegt ist, ist die Idee dann freigabebereit. Die Idee wird entweder automatisiert oder durch eine bestimmte Person für die Ideenprüfung freigegeben.

3. Phase: Ideenprüfung
   Die Ideenprüfung wird von einer bestimmten Person durchgeführt, beispielsweise könnten Zuständigkeiten pro Kategorie festgelegt werden. Diese Person prüft die Idee anhand von Checklisten. Dadurch, dass schon vorher über das Problem und die Idee diskutiert wurde, ist der Aufwand in dieser Phase für die Person eher gering. Sie prüft die Idee beispielsweise auf Redundanzen, auf das tatsächliche Vorhandensein des Problems und auf Umsetzbarkeit.

4. Phase: Business Case
   Der Business Case für das Projekt zur Umsetzung der Idee wird in dieser Phase erstellt.

Literaturverzeichnis

Monographien

Vahs, D.; Brem, A. (2015): Innovationsmanagement: Von der Idee zur erfolgreichen Vermarktung. Stuttgart

Heesen, M. (2009): Innovationsportfoliomanagement: Bewertung von Innovationsprojekten in kleinen und mittelgroßen Unternehmen der Automobilzuliefererindustrie, Dissertation, Universität Duisburg-Essen, Essen

Cooper, Robert G. (2002): Top oder Flop in der Produktentwicklung. Erfolgsstrategien: Von der Idee zum Launch, Weinheim

Müller-Prothmann, T; Dörr, N (2009): Innovationsmanagement, München

Zeitschriften

Hughes, G. D.; Chafin, D. C. (1996): „Turning New Product Development into a Continuous Learning Process", in: Journal of Product Innovation Management, Jg. 13, S. 89-104.

Arbeitspapiere

Verworn, B.; Herstatt, C. (2000): Modelle des Innovationsprozesses: Eine Einführung, Arbeitspapier Nr. 6, Technische Universität Hamburg-Harburg, Hamburg

Müller, M.; Graubner, C. (2008): Wissensmanagement & Innovationsmanagement: Innvoationsprozzess I, Universität Erlangen-Nürnberg, Technische Fakultät (http://wwwdh.cs.fau.de/IMMD8/Lectures/WMIM/scinovis-folien_vorlesung-wmim_2008-06-30_innovationsprozess_Ix.pdf 22.06.2015)

Internet-Adressen

Institut für Technologie und Arbeit (o. J.): Innovation, Innovationsprozesse und Innovationsmanagement.
http://www.optimus-spitzencluster.de/innovationinnovationsprozesseundinnovationsmanagement.pdf 22.06.2015

SpinnRaum (2014): Alternativen zum Stage Gate-Prozess der Innovation. http://www.spinnraum.at/alternativen-zum-stage-gate-prozess-der-innovation/ 22.06.2015

Erfolgsbeteiligung für Mitarbeiter im Ideenmanagement

Seminararbeit
*i*m Rahmen der Projektgruppe IMPACT

Themensteller:	Prof. Dr.-Ing. Jorge Marx Gómez
Betreuer:	Dr. Joachim Kurzhöfer Dipl.-Math. Jens Siewert Stefan Wunderlich, M. Sc.
Vorgelegt von:	Daniel Martin Ahlers Flötenstr. 70b 26125 Oldenburg daniel.martin.ahlers@uni-oldenburg.de
Abgabetermin:	22. Juni 2015

Inhaltsverzeichnis

1 Einleitung

Diese Seminararbeit behandelt einen Teil der Mitarbeitermotivation im Unternehmen, die Erfolgsbeteiligung für Mitarbeiter im Ideenmanagement. Als Einführung der Arbeit wird zunächst auf die Historie des Ideenmanagements eingegangen. Es wird beschrieben, in welchem betrieblichen Umfeld das in der Zeit genannte betriebliche Vorschlagswesen eingeführt wurde, um im Verlauf der Arbeit, bis zum Ende hin, eine Grundlage für Verbesserungen zu stellen.

Da die Erfolgsbeteiligung für Mitarbeiter eine von vielen Motivationsformen darstellt, werden im Anschluss Grundbegriffe der Motivation eines Menschen beschrieben. Es werden mehrere unterschiedliche Anreize aufgezeigt, auf Basis dessen ein jeder Mensch handelt.

Darauf folgend wird das Thema der Kreativität behandelt. Diese stellt die Grundlage eines Ideenmanagements dar, da Kreativität der Schlüssel zu neuen Ideen ist. Es wird grundlegend beschrieben, welche Anforderungen kreatives Denken und Handeln begünstigen, um darauf aufbauend beleuchten zu können, auf welche Weise eine Förderung der Mitarbeiter durchgeführt werden kann.

Zum Abschluss wird es zur Frage kommen, in welchem Umfeld eine Mitarbeiterbeteiligung Sinn machen kann, ob sie in der heutigen Zeit noch eine Daseinsberechtigung inne hat und Alternativen aufgezeigt, die Anstelle der Mitarbeiterbeiteilung treten oder diese ergänzen können.

2 Die Historie des Ideenmanagements

In diesem Kapitel geht es um die Entstehung des heutigen Ideenmanagements, welche zum ersten Mal in der Geschichte als betriebliches Vorschlagswesen beschrieben wurde. Um sinnvoll auf die Erfolgsbeteiligung für Mitarbeiter eingehen zu können, ist das Verständnis über die Betriebsbedingungen zur Zeit des Entstehens der Vorschlagswesen in Unternehmen wichtig. Auf Basis dessen können Unterschiede zu Bedingungen in heutiger Zeit geschlossen werden.

2.1 Das Vorschlagswesen zu Beginn des 20. Jahrhunderts

Das Vorschlagswesen in einem Unternehmen wurde zuerst im Jahre 1750 in der schwedisch königlichen Kommission entwickelt. Viele Jahre später erst folgten die USA und kurz darauf auch Deutschland mit einem solchen System, ein besonderer Boom setzte zu Beginn des 20. Jahrhunderts ein, in dem viele große Unternehmen von diesem System profitieren wollten. Ziel war die Förderung und Anerkennung von Mitarbeitern, um mit

den Ideen Arbeitsabläufe effizienter zu gestalten, den Arbeitsplatz sicherer zu machen, Unfälle zu minimieren, die Qualität zu steigern und global Ansätze umzusetzen, um einen Wettbewerbsvorteil zu erhalten. [Tho09, S. 121ff.]

Die planerischen und auszuführenden Tätigkeiten wurden zu dieser Zeit strikt getrennt. Im Rahmen der Industrialisierung im 19. Jahrhundert entstanden riesige Fabrikhallen, getrimmt auf Effizienz. Fließbandarbeit wurde propagiert, auszuführende Mitarbeiter hatten mit einfachen Handgriffen Maschinen zu bedienen, schnell, effektiv und effizient. Sie konnten sich selten mit ihrer Arbeit identifizieren. Das Nachdenken, insbesondere über Betriebsprozesse, war nicht vorgesehen. [Sch96, S. 10ff.]

Dass dies nicht der menschlichen Natur entspricht, erkannte man in Deutschland zum Ende des 19. Jahrhunderts. Alfred Krupp führte als erster deutscher Unternehmer das betriebliche Vorschlagswesen in seinem Unternehmen ein. Auch in anderen Firmen, wie AEG im Jahr 1901 oder Bayer 1909, erkannten man bald, dass sich Mitarbeiter automatisch über ihre Arbeitsumgebung Gedanken machten. Gedanken, die die Effizienz steigern könnten oder Kosten senken. Dementsprechend wurde das betriebliche Vorschlagswesen ursprünglich als Rationalisierungsinstrument entworfen, mit dem möglichst effizient ein Wettbewerbsvorteil errungen werden sollte. [Tho09, S. 121ff.]

Den Mitarbeitern wurde als Gegenleistung für eine eingereichte und später umgesetzte Idee ein Obolus zugesprochen. Durch den geringen Verdienst bei hoher Stundenleistung konnte dies für die Arbeiter ein durchaus attraktives Modell sein. [Sch96, S. 10ff.]

2.2 Arbeiten im 21. Jahrhundert

Das betriebliche Vorschlagswesen existiert heute, in anderer Form und unter dem Begriff Ideenmanagement, in mehr Firmen denn je. Trotz dessen muss das heutige Ideenmanagement klar vom früheren betrieblichen Vorschlagswesen differenziert werden. Insbesondere haben die Mitarbeiter einer Firma in der Gegenwart einen anderen Stellenwert, als dies zu Ende des 19. Jahrhunderts war, das System musste angepasst werden. [Tho09, S. 121ff.]

Zunächst wird in modernen Unternehmen eine flache Hierarchie angestrebt, in der Mitarbeiter ein gewisses Maß an Mitspracherecht bekommen. Es wird Wert darauf gelegt, den Mitarbeiter auch non-monetär zum Arbeiten zu motivieren. In Arbeitsverträgen sind teilweise Klauseln enthalten, in denen Mitarbeitern explizit erlaubt wird, persönlichen Angelegenheiten nachzugehen - in entsprechendem Rahmen. Auch wenn in heutiger Zeit weiterhin Gehälter als Aufwandsentschädigung betrachtet werden können, können Arbeiter einer Firma inzwischen durchaus ein respektables Leben führen. [Tho09, S. 121ff.]

2.3 Fazit

Die Arbeitsumgebung hat sich seit der Industrialisierung und der Zeit von ersten betrieblichen Vorschlagswesen signifikant verändert. Menschen handeln aus einer anderen Intention heraus, werden stärker in Unternehmensentscheidungen mit eingebunden und erhalten für ihre Arbeit mehr Geld. Es muss daher die Frage behandelt werden, ob andere Anreizsysteme für das Vorschlagswesen bzw. dem heutigen Ideenmanagement etabliert werden müssen, als eine reine Geldzahlung.

3 Die Motivation eines Menschen

Dieser Teil der Arbeit beschäftigt sich mit Frage, was unter dem Begriff Motivation zu verstehen ist und aus welcher Intention heraus Menschen grundlegend handeln.

Das Leben eines jeden Menschen verläuft unterschiedlich, trotz dessen können Bereiche des Handelns in unterschiedlichen Kategorien miteinander verglichen werden. Beobachtet man Menschen in ihrem Handeln, stellt man fest, dass viele von ihnen einen bestimmten Antrieb haben, aus dem ihr Handeln folgt.

In einem Beispiel geht Person A gerne seinen Hobbys nach, spielt gar Instrumente, betreibt Sport. Das Arbeiten ist für ihn ein Mittel zum Zweck, er ist nicht bedacht darauf, sich in seinem Erfolg von anderen Menschen abzuheben. Person B verfolgt andere Interessen: Für ihn ist sein Beruf zum Hobby geworden, für andere Aktivitäten hat er ohnehin nur wenig Zeit. Er steigt in seinem Job schnell auf, ist stolz auf das, was er bereits geschafft hat. Aus den Aktivitäten heraus steigt sein Selbstwertgefühl.

Dieses sehr simple Beispiel polarisierender Personentypen verdeutlicht bereits, dass Menschen sehr unterschiedliche Intentionen zum Ausfüllen ihres Lebens haben. Während Person A aus eigenem Antrieb heraus handelt und sein Handeln danach bestimmt, was ihm Freude bereitet, handelt Person B aus dem Antrieb heraus, das Maximum aus seinem Handeln herauszuholen, in Form von Geld oder Respekt anderer Menschen ihm gegenüber.

Grundlegend kann das Handeln in zwei verschiedene Motivationsarten unterschieden werden, die intrinsische Motivation sowie die Extrinsische. [McC87] Abraham Maslow beschrieb 1954 in seiner Motivationstheorie die fünf Quellen der Motivation, bestehend aus zwei intrinsischen und drei extrinsische Motivationsformen.

3.1 Intrinsische Motivation

Die intrinsische Motivation bezeichnet eine Motivatonsform, bei der Handlungen vollständig aus eigenem Interesse heraus durchgeführt werden. Es existieren keinerlei externe Inter-

essen, welche sich in Versprechen oder gar Drohungen äußern können. Gründe für eine intrinsische Motivation entstehen z.B. aus Neugier an der Sache, Exploration oder dem Interesse an der Umwelt. Kennzeichnend für diese Form der Motivation ist darüber hinaus, dass Menschen über das Handeln nicht lange nachdenken, es wird direkt gehandelt. [McC87]

3.1.1 Interne Prozessmotivation

Als Teil der intrinsischen Motivation gilt die interne Prozessmotivation. Menschen, die aus diesem Antrieb heraus handeln, tun dies aus reinem Selbstwillen, dem Spaß und Interesse an der Sache. Oftmals werden Hobbys auf Basis der internen Prozessmotivation ausgeführt. [McC87]

3.1.2 Internes Selbstverständnis

Im Gegensatz zur internen Prozessmotivation ist das interne Selbstverständnis ebenfalls ein Antrieb, der von der betreffenden Person aus kommt, lediglich indirekt durch externes Einwirken entstehen kann. Das interne Selbstverständnis hat ihren Ursprung in einer Idealvorstellung, die in der Vergangenheit durch Externe suggeriert wurde oder durch eigene Handlungen entstanden ist. Der genaue Ursprung ist im Regelfall für die Person nicht mehr erkennbar. Das Handeln entsteht somit, um eigenen Vorstellungen gerecht zu werden. Beispielhaft können dies Arbeiter einer Firma sein, die aus reinem Selbstverständnis eine gute Arbeit leisten möchten, auch wenn sie für eine geringere Arbeit nicht weniger Lohn erhalten würden. [McC87]

3.2 Extrinsische Motivation

Die extrinsische Motivation betrachtet den externen Antrieb als Motivation für das Handeln. Insbesondere geht es bei Formen dieser Motivationsquelle darum, entweder eine direkte Gegenleistung in Form einer Belohnung zu erhalten oder entstehende Nachteile im Falle des Nichthandelns abzuwenden. [McC87]

3.2.1 Instrumentelle Motivation

Die Instrumentelle Motivation ist angelehnt an das so genannte Machtmotiv. Es geht bei dieser Form der Motivation um die Belohnung, die für bestimmtes Handeln erfolgt. Dies ist nicht zwangsweise an Geld gekoppelt, sondern kann sich auch in anderer Form von Macht äußern, beispielsweise in dem Bestreben, die Karriereleiter im Job aufzusteigen,

Leiter einer Abteilung/ Filiale etc. zu werden. Auch die reine Hoffnung auf diese Macht kann Menschen zum Handeln motivieren. [McC87]

3.2.2 Externes Selbstverständnis

Das externe Selbstverständnis weißt eine gewisse Ähnlichkeit zum internen Selbstverständnis auf. Auch diese Form der Motivation basiert darauf, dass eine Idealvorstellung existiert, die zu erfüllen versucht wird. Im Unterschied jedoch geht diese Vorstellung von der Umgebung aus. In dieser Form der Motivation wird durch das Handeln häufig ein Schaden abgewendet, der angedroht wird. In einem Beispiel würde ein Mitarbeiter einer Firma nicht deswegen besonders hohe Arbeitsleistung vollbringen, um seinen eigenen Vorstellungen gerecht zu werden, sondern um die Anforderung vom Arbeitgeber zu erfüllen, weil dieser ihn evtl. bei Nichterbringung kündigen würde. [McC87]

3.2.3 Internalisierung von Zielen

Bei der Internalisierung von Zielen machen sich Menschen höher gelegene Ziele zu eigen. Sie identifizieren sich damit und versuchen zum Erfolg zu gelangen. Ein typisches Beispiel ist ein Manager, welcher den Verlauf seine Firma optimieren, höheren Gewinne generieren und das Fortbestehen absichern möchte.

3.3 Überlagerung verschiedener Motivationsformen

Bei genauer Betrachtung des Beispiels in Abschnitt 3 wird schnell deutlich, dass dies sehr trivial gewählt ist. Insbesondere wurden Personen bezeichnet, welche lediglich aus einer einzelnen Intention heraus handeln. In der Realität spielen jedoch viele weitere Faktoren eine Rolle, insbesondere handeln Menschen oftmals aufgrund vielfältiger Gründe. Ein Mitarbeiter einer Firma handelt eventuell in erster Linie aus der Intention heraus, Geld für sich und seine Familie verdienen zu wollen, demnach spielt das Motiv der Instrumentellen Motivation eine große Rolle. [AZ09, S. 130ff.]

Trotz dessen kann er sich in seinem Job und dem dortigen Umfeld wohlfühlen, hat sich diese Arbeit ausgesucht, da sie ihm mitunter auch Spaß bereitet. In diesem Fall kann durchaus eine interne Prozessmotivation eintreten, in der der Mitarbeiter aus eigenem Interesse heraus handelt. Es kann sich zudem über einen entsprechenden Zeitraum ein Selbstverständnis aufbauen, dem der Mitarbeiter gerecht werden möchte. [AZ09, S. 130ff.]

Es stellt sich die Frage, ob und inwiefern diese Motivationsformen miteinander verknüpft werden oder gar konträr sind. Insbesondere beschäftigten sich Wissenschaftler mit dem Punkt, ob die intrinsische Motivation durch äußere Einflüsse und extrinsische Faktoren

vermindert wird, bezeichnet als Korrumpierungseffekt. Tendenziell scheint dieser Effekt durchaus folgerichtig zu sein. Ein Mitarbeiter einer Firma kann durchaus Spaß an seiner Arbeit haben, primär jedoch wird er häufig des Geldes wegen an jedem Werktag zur Arbeit erscheinen. [AZ09, S. 130ff.]

Eine eindeutige, wissenschaftliche Antwort über die Existenz dieses Effektes besteht zum aktuellen Zeitpunkt allerdings nicht. In empirischen Studien wurden Menschen beobachtet, die zunächst eine Tätigkeit aus gänzlich eigenem Antrieb durchführen und im Anschluss für die gleiche Tätigkeit eine feste Belohnung erhielten. Besonders nach Beenden dieser extrinsischen Motivation konnte festgestellt werden, dass Versuchspersonen einen geringeren eigenen Antrieb vorweisen konnten, als dies vor der Gabe der Belohnung der Fall war. Nicht geklärt ist die Frage danach, ob dies nur unter bestimmten Bedingungen zu beobachten ist und ob sich dieses Verhalten in die Realität übertragen lässt. [AZ09, S. 130ff.]

3.4 Fazit

In diesem Kapitel wurden die fünf Quellen der Motivation nach Maslov beschrieben, welche sich in intrinsischer und extrinsischer Motivation unterteilen. Die intrinsische Motivation beschreibt den eigenen Antrieb, bestimmte Tätigkeiten zu erledigen, z.B. aus Interesse oder dem Spaß an der Sache. Die extrinsische Motivation beschreibt den externen Antrieb, d.h. Tätigkeiten, die durch eine Belohnung durch Ausführung entstehen oder durch Bestrafung bei Nichterledigung. Abschließend wurden Theorien beleuchtet, nach denen Menschen durch einen externen Antrieb die bestehende interne Motivation verlieren können und somit durch eine feste Belohnung der eigene Antrieb an einer Sache vermindert werden kann.

4 Kreativität

Mit dem Begriff Kreativität wird im weitesten Sinne die Eigenschaft eines Menschen charakterisiert, etwas Neuartiges zu schaffen. Dies kann eine Lösung für ein bekannte Problem darstellen oder eine Entdeckung sein. [HH07, S. 9 ff.] Die Kreativität spielt daher in einem Ideenmanagement eine sehr große Rolle, so ist sie der Ursprung, auf Basis dessen neue Ideen entstehen. Daher wird in diesem Kapitel beleuchtet, welche Eigenschaften eines Menschen zu derartigem Denken und Handeln führen, sodass auf Basis dessen Optimierungen erarbeitet werden können.

Kreativität wird häufig als besondere Charakteristik bestimmten Personen zugeschrie-

ben, Künstlern beispielsweise, Visionären, die in ihrem Leben bereits etwas Neuartiges geschaffen haben. In der Wissenschaft jedoch betrachtet man die Kreativität als eine Eigenschaft aller Lebewesen. Trotz dessen wird deutlich, dass sehr unterschiedliche Personentypen zu unterschiedlichen Neuerungen in der Gesellschaft führen. Musiker haben beispielsweise häufig ein anderes Persönlichkeitsprofil als Erfinder im technischen Bereich.

Man hat feststellen können, dass vier wesentliche Elemente prägend sind: Die Begabung, Motivation, Persönlichkeit und Umgebung. [HH07, S. 11]

4.1 Begabung

Die Begabung für eine bestimmte Tätigkeit stellt eine wichtige Fähigkeit eines Menschen dar, um kreativ zu handeln. Unterschiedliche Begabungen können aus unterschiedlichen Intelligenzformen entspringen. Im Gehirn eines jeden Menschen sind manche Gehirnareale stärker und andere schwächer ausgeprägt. Je nachdem, welche Areale davon betroffen sind, fällt nicht nur die entsprechende Tätigkeit leichter, das Gehirn beschäftigt sich zudem unterbewusst stärker mit größer ausgeprägten Arealen. Kreative Ideen entspringen demnach häufig der Intelligenzform des Individuums, dessen Ursprung sehr vielschichtig sein kann. Gewisse Teile können angeboren sein oder bedingt durch das Handeln. Forscher haben festgestellt, dass gewisse Begabungen auch entstehen können, wenn sich Menschen in sehr frühem Kindes- und Jugendalter mit gewissen Tätigkeiten beschäftigen. [HH07, S. 11 ff.]

4.2 Motivation

Der zweite wichtige Punkt zum kreativen Handeln ist die Motivation. Diese ist oft angelehnt an den Punkt der Begabung. Es besteht Einigkeit darüber, dass Menschen zu kreativem Nachdenken angeregt werden, wenn sie ein persönliches Interesse an der Tätigkeit entwickeln, daher intrinsisch motiviert sind. Oftmals besteht die intrinsische Motivation genau denn, wenn ein Mensch Tätigkeiten ausführt, die der Begabung entsprechen. Trotz dessen können andere Faktoren eine Rolle spielen, zum Beispiel wenn der bestehende, intrinsische Antrieb durch externe Faktoren überlagert wird, siehe Abschnitt 3.3. [HH07, S. 15 ff.]

4.3 Persönlichkeit

Kreatives Denken allein reicht nicht aus, um auch kreativ Handeln zu können. Oft sind Bedingungen der Persönlichkeit ein großes Erfordernis, damit kreative Gedanken auch

in die Realität umgesetzt werden. Wichtig Bedingungen sind unter Anderem das Selbstwertgefühl, die Phantasie, Hingabefähigkeit und Frustrationstoleranz. Eine Person mit sehr geringem Selbstwertgefühl wird beispielsweise seine kreativen, neuartigen Gedanken selten mit anderen Menschen teilen wollen. Persönlichkeitseigenschaften entwickeln sich häufig in jungen Jahren, es kann allerdings durchaus noch zu Veränderungen kommen, sowohl zum Ausprägen von positiven, als auch negativen Eigenschaften. [HH07, S. 16]

4.4 Umgebung

Neben Bedingungen an die Persönlichkeit werden auch Bedingungen an die Umgebung gestellt, um kreatives Handeln zu begünstigen. Betrachtet wird, ob die Möglichkeit existiert, die Begabung in die Tat umsetzen zu können. Beispielsweise wird ein Mensch mit musikalischer Begabung diese niemals entdecken, wenn er in seinem Leben nicht die Möglichkeit erhält, ein Instrument zu spielen. [HH07, S. 17]

4.5 Fazit

Es wurde beschrieben, was genau unter dem Begriff der Kreativität zu verstehen ist und welche Bedingungen erfüllt sein müssen, um kreatives Denken und Handeln zu ermöglichen. Es wurden die vier wesentlichen Elemente beschrieben, die dies bedingen: Die Begabung, die Motivation, die Persönlichkeit und die Umgebung. Auf Basis dessen können im weiteren Verlauf Überlegungen durchgeführt werden, auf welche Weise vorhandenes Potential gefördert werden kann.

5 Schlussteil

In den vorherigen Kapiteln wurde beschrieben, in welchem geschichtlichem Umfeld betriebliche Vorschlagswesen entstanden sind und wie sich Arbeitsbedingungen zu jener Zeit zu denen in der Gegenwart differenzieren. Im Anschluss wurde auf die Motivation von Menschen eingegangen, insbesondere auf die verschiedenen Formen, die sich in intrinsischer und extrinsischer Motivation unterteilen und eine Teilbedingung der Kreativität sind. Diese wurde ebenfalls genauer beleuchtet und beschrieben, welche weiteren Bedingungen erfüllt sein müssen, um kreatives Denken und Handeln zu begünstigen.

In diesem Kapitel wird auf Basis dessen die Frage geklärt, inwiefern das kreative Handeln von Mitarbeitern optimiert werden kann, welche Motivationsformen dies begünstigen und die zentrale Frage gestellt, ob die Mitarbeiterbeteiligung unter diesen Gesichtspunkten eine reelle Daseinsberechtigung hat.

5.1 Anreizsysteme zur Erhöhung der kreativen Leistung

Zunächst sollten die Gegebenheiten einer Firma betrachtet und Ziele definiert werden, nach denen gehandelt werden soll. Sollen lediglich von Mitarbeitern einer Produktionsstätte Ideen eingereicht werden können, so kann eine Sammelstelle für Ideen mit anschließender Mitarbeiterbeteiligung ausreichend sein, wie dies zu Zeiten der Entstehung von betrieblichen Vorschlagswesen gehandhabt wurde.

Trotz dessen können andere Wege beschritten werden. Insbesondere stellt sich in dieser Arbeit die Frage, auf welche Weise Mitarbeiter motiviert werden können, am Prozess des Ideenmanagements teilzuhaben. Wie in Abschnitt 4 beschrieben, begünstigt das eigene Interesse an der Aktivität die kreative Leistung. Das Optimieren von intrinsischer Motivation ist in dem Sinne durchaus sinnvoll. Diese Form der Motivation von Grund auf aufzubauen, kann sich als schwierig gestalten. Zunächst sollten Überlegungen durchgeführt werden, mit dem Ziel, einen bestehenden intrinsischen Antrieb der Mitarbeiter zu erhalten. Dies geschieht im Regelfall durch das Abbauen vorhandener Hürden und das Aufbauen von Interesse und Neugier, beispielsweise, indem den Mitarbeitern ein gewisser Freiraum gelassen wird, eine bestehende Idee zu entwickeln und sich dahingehend frei entfalten zu können.

Auch wenn, wie in Abschnitt 3.3 beschrieben wurde, eine extrinsische Motivation die Intrinsische überlagern kann, ist es nicht immer von Nachteil, den Mitarbeitern externe Anreize zu geben. Insbesondere, wenn es darum geht, dass durch eine Idee Kosteneinsparungen für das Unternehmen entstanden sind, möchte der Mitarbeiter unter Umständen daran beteiligt werden. Dies hat in dem Kontext jedoch den Charakter, die eigentliche Kosteneinsparung für das Unternehmen zu honorieren, anstelle den Mitarbeiter zum Einreichen einer Idee zu motivieren.

5.2 Fazit

Diese Arbeit ist explizit keine direkte Anleitung, um Mitarbeiter zu motivieren, am Ideenmanagement eines Unternehmens teilzuhaben. Vielmehr wurde beschrieben, welche Bereiche beachtet werden müssen, um Motivation und Kreativität zu fördern. Als Fazit kann gewertet werden, dass die Mitarbeiterbeteiligung in Unternehmen prinzipiell keine schlechte Möglichkeit ist, um dieses Ziel zu erreichen. Je nach Unternehmenskultur und -politik sollten jedoch auch andere Wege betrachtet werden. Insbesondere kann es sinnvoll sein, dass die Beteiligung für Mitarbeiter nur einen kleinen Teil der Motivation darstellt bzw. nur eine ergänzende Funktion inne hat. Man sollte beachten, dass diese Form der Motivation das Problem beinhalten kann, dass das eigene Interesse der Mitarbeiter verloren geht

und damit eine Verminderung der kreativen Leistung einhergeht. Die Mitarbeiterbeteiligung im Ideenmanagement sollte demnach mit Bedacht angewendet werden.

Literatur

[AZ09] Kathrin M. Möslein Ansgar Zerfaß. *Kommunikation als Erfolgsfaktor im Innovationsmanagement: Strategien im Zeitalter der Open Innovation*. Gabler Verlag, 2009.

[HH07] Rainer M. Holm-Hadulla. *Kreativität - Konzept und Lebensstil*. Vandenhoeck and Ruprecht, 2007.

[McC87] David C. McClelland. *Human Motivation*. Cambridge University Press, 1987.

[Sch96] Gerhard Schildt. *Die Arbeiterschaft im 19. und 20. Jahrhundert*. Oldenbourg Verlag, 1996.

[Tho09] Norbert Thom. *Vom Vorschlagswesen zum Ideen- und Verbesserungsmanagement: Kontinuierliche Weiterentwicklung eines Managementkonzepts*. Lang Verlag, 2009.

B.1.5. Gamification

Thema:

Gamification

Seminararbeit
im Rahmen der Projektgruppe „IMPACT“

Abteilung Wirtschaftsinformatik 1:
Very Large Business Applications

Themensteller:	Prof. Dr.-Ing. Jorge Marx Gómez
Betreuer:	Dr. Joachim Kurzhöfer Stefan Wunderlich Jens Siewert
vorgelegt von:	Patrick Smit Artillerieweg 44 26129 Oldenburg E-Mail: patrick.smit@uni-oldenburg.de
Abgabetermin:	22. Juni 2015

Inhaltsverzeichnis

Verzeichnis der Abkürzungen und Akronyme

IMPACT	Innovation Management Plattform To Activate Creative Thoughts
IMP	Innovations-Management-Plattform
PvP	Player versus Player

Abbildungsverzeichnis

1 Einleitung

Die Projektgruppe „Innovation Management Plattform To Activate Creative Thoughts“ (kurz: IMPACT) hat die Aufgabe eine innerbetriebliche Innovations-Management-Plattform (kurz: IMP) zu konzipieren, zu entwickeln und einzuführen. Um die vielen Themen aufzubereiten, die eine solche Plattform beeinflussen, wurden an die Mitglieder unterschiedlichste Themen verteilt. Mit diesen Themen haben wir uns in der Seminarphase beschäftigt und als Ergebnis entsteht pro Thema eine Präsentation als auch eine Ausarbeitung.

Ziel dieser Seminarphase ist es in dem jeweiligen Thema Wissen anzueignen und dieses mit Hilfe der Präsentation den anderen Gruppenmitgliedern zu vermitteln. Dieses Zusatzwissen soll uns helfen fundierte Entscheidungen bezüglicher unserer IMP zu treffen.

In dieser Ausarbeitung werde ich das Thema Gamification genauer vorstellen. Zuerst gebe ich eine kurze Definition und schaue auf die historische Entwicklung. Im Anschluss werde ich das Gamification Framework „Octalysis“ von Yu-kai Chou vorstellen. In Kapitel vier werden die Potenziale und Anwendungsgebiete von Gamification aufgezeigt. Im darauf folgenden Kapitel geht es um das Gamification Desgin Problem. Im vorletzten Kapitel werde ich meine Empfehlungen an die Projektgruppe geben und begründen. Der Abschluss bildet ein Fazit als auch ein Ausblick in die Zukunft.

2 Definition und Geschichte

2.1 Definition Gamification

Gartner definiert Gamification als „ […] the use of game mechanics and experience design to digitally engage and motivate people to achieve their goals“ (Burke (2014)).

Die Schlüsselelemente sind dabei folgende (frei übersetzt) (vgl. Burke (2014)):

- Unter den „game mechanics“ werden Elemente wie z.B. Punkte, Abzeichen und Ranglisten zusammengefasst. Diese sind in sehr vielen Spielen verbreitet.
- „Experience design“ beschreibt die Reise des Spielers, die durch Elemente beeinflusst werden wie z.B. das Gameplay und die Story-Line.
- „Digitally engage“ bedeutet, dass die Spieler über Computer, Handys und sonstige digitale Geräte zugreifen und weniger mit anderen Personen direkt interagieren.
- Das Ziel der Gamification ist es die Benutzer zu motivieren ihr Verhalten zu ändern, Fähigkeiten zu entwickeln oder Innovationen voranzutreiben.
- Und schlussendlich sollen die Spieler ihre Ziele erreichen. Wenn Ziele der Organisationen mit denen der Spieler zusammenhängen, dann erreicht die Organisation ihre Ziele, wenn die Spieler ihre Ziele erreichen.

2.2 Die Geschichte von Gamification

Auch wenn das Wort Gamification noch sehr unbekannt ist, so ist das Konzept an sich ein sehr altes. Schon vor über hundert Jahren wurden Spiele und Spaß genutzt um die Angestellten zu motivieren und die Arbeit angenehmer zu gestalten. So konnten die „Boy-Scouts“ bereits 1911 verschiedene Ränge und Abzeichen erreichen (vgl. Fitz-Walter (2013)).

Das Wort Gamification wurde erstmals in den Jahren 2002/2003 verwendet um die Arbeit von Nick Pellings zu beschreiben. Im Jahr 2008 folgte dann die erste Nutzung in einem Dokument. Aber erst 2011 gelangte das Wort zu mehr Bekanntheit, als Gartner es zu dem „Hype Cycle“[1] hinzugefügt hatte (vgl. Fitz-Walter (2013)).

[1] Der Hype Cycle stellt die Phasen der öffentlichen Aufmerksamkeit dar, welche eine neue Technologie bei der Einführung durchläuft.

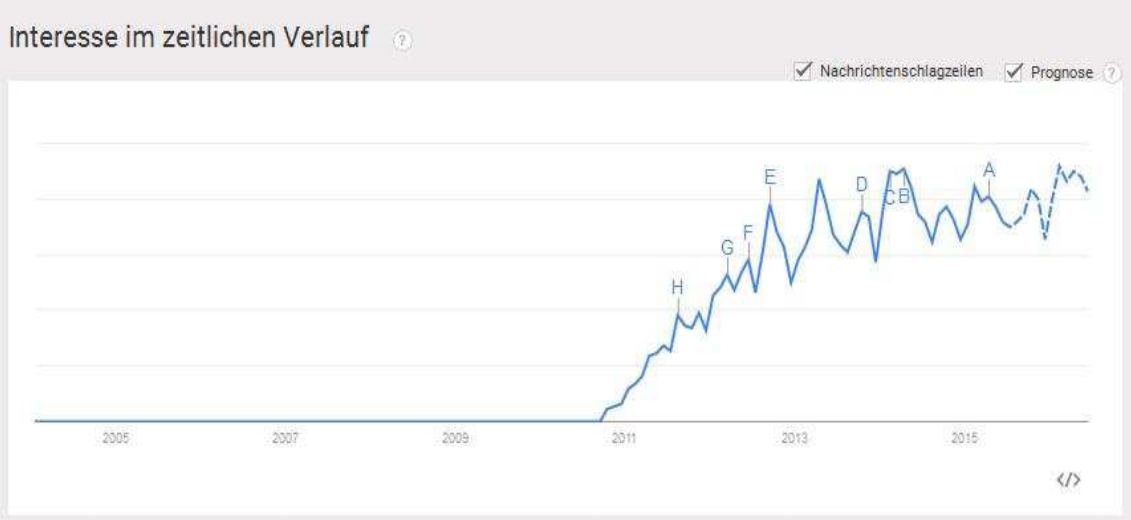


Abb. 2.1 Gamification bei Google Trends (Quelle: Google Trends (2015))

In Abbildung 2.1 erkennt man, dass ab 2011, als Gartner „Gamification“ zum Hype-Cycle hinzugefügt hat, ein signifikanter Anstieg existiert. In den letzten Jahren ist das Interesse trotz kleiner Schwankungen relativ konstant und auch die Prognose verzeichnet keine Ausreißer.

Suchanfragen	Beliebteste	Zunehmende
gamification examples	100	
gamification education	100	
what is gamification	70	
gamification learning	70	
gamification design	65	
gamification marketing	60	
social gamification	60	

Abb. 2.2 Die beliebtesten Suchanfragen (Quelle: Google Trends (2015))

In Abbildung 2.2 werden die beliebtesten Suchanfragen aufgezeigt. Das Gamification vor allem im Zusammenhang mit education/learning und Marketing gesucht wird, ist dabei nicht überraschend. So sind es vor allem die beiden Branchen, in denen sich Gamification bereits in den letzten Jahren durchgesetzt hat (vgl. Diercks (2014)).

3 Octalysis Gamification Framework

Das folgende Kapitel beinhaltet das Octalysis Gamification Framework von Yu-Kai Chou und bezieht sich auf (Chou (2015)).

Yu-Kai Chou, der Erfinder des Gamification Framworks, ist ein Gamification Pionier seit 2003 und lehrt unter anderem an der Stanford Universität und gilt als einer der Top 3 Gamification-Gurus.

In seinem Gamification Framework beschreibt er die 8 treibenden Bereiche, die Gamification ausmachen. Auf Grund der 8 Bereiche erfolgte auch die Namensgebung zu Octalysis (lat. Octo = acht).

3.1 Das Framework und seine Bereiche

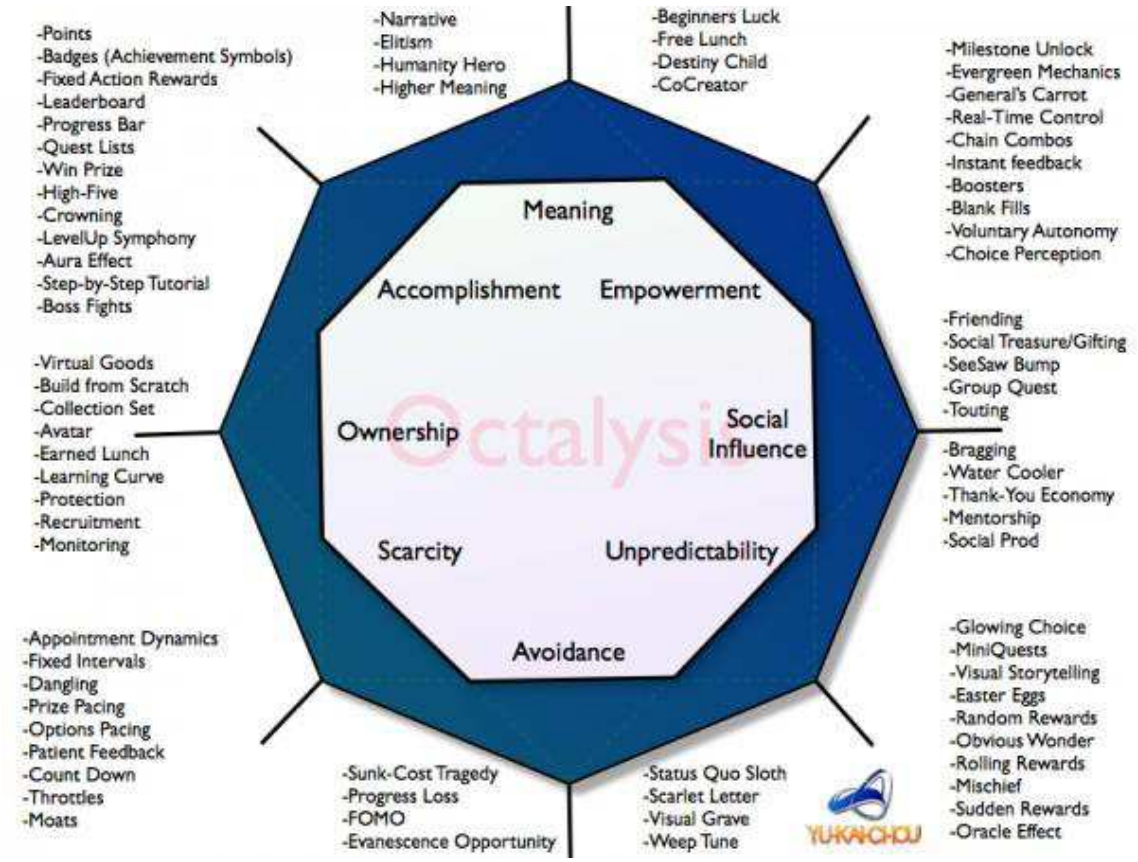


Abb. 3.1 Das Octalysis Framework (Quelle: Chou (2015))

Wie in Abbildung 3.1 zu erkennen, gibt es folgende 8 treibende Kräfte im Gamification-Framework:

- **Meaning:** Der Spieler glaubt, er sei für etwas Größeres bestimmt. Dieses kann z.B. durch Glück bei den Startitems hervorgerufen werden.

- **Empowerment of Creativity:** Unterschiedliche Wege und Möglichkeiten steigert und nutzt zugleich die Kreativität der Spieler. Wichtig hierbei ist, dass die Ergebnisse der Kreativität sichtbar sein.

  Beispiel: Wenn es in einem Spiel zwei Wege gibt, sollten diese auch zu unterschiedlichen Story-Lines führen.

- **Social Influence:** Beinhaltet alle sozialen Elemente, die einen Menschen antreiben wie z.B. Akzeptanz, Verantwortung und Wettbewerb.

- **Unpredictability:** Jeder Mensch verfolgt das Ziel herauszufinden was als nächstes passiert. Wenn der Spieler jedoch nicht gleich die Lösung vorgesetzt bekommt, denkt er in der Zwischenzeit häufiger darüber nach, was passieren könnte.

- **Avoidance:** Beschreibt das Verlangen etwas Negatives abzuwenden.

- **Scarity & Impatience:** Hier werden die Elemente untergebracht, die auf der Angst und Ungeduld der Spieler basieren wie z.B. Count-Downs oder variable Preise.

- **Ownership:** Das Gefühl etwas zu besitzen spielt eine wichtige Rolle. Wenn das Erstellen eines Profils oder Avatars mehr Zeit braucht, so steigt auch automatisch die persönliche Wertigkeit.

- **Accomplishment:** Beschreibt den inneren Antrieb einen Fortschritt zu erzielen. Das Wort „Herausforderung“ ist ein elementarer Bestandteil.

Innerhalb dieser Bereiche werden die typischen Werkzeuge, Elemente und Methoden aufgezählt, die den jeweiligen Bereich in Spielen und Anwendungen repräsentieren.

3.2 Left Brain/ Right Brain Einteilung

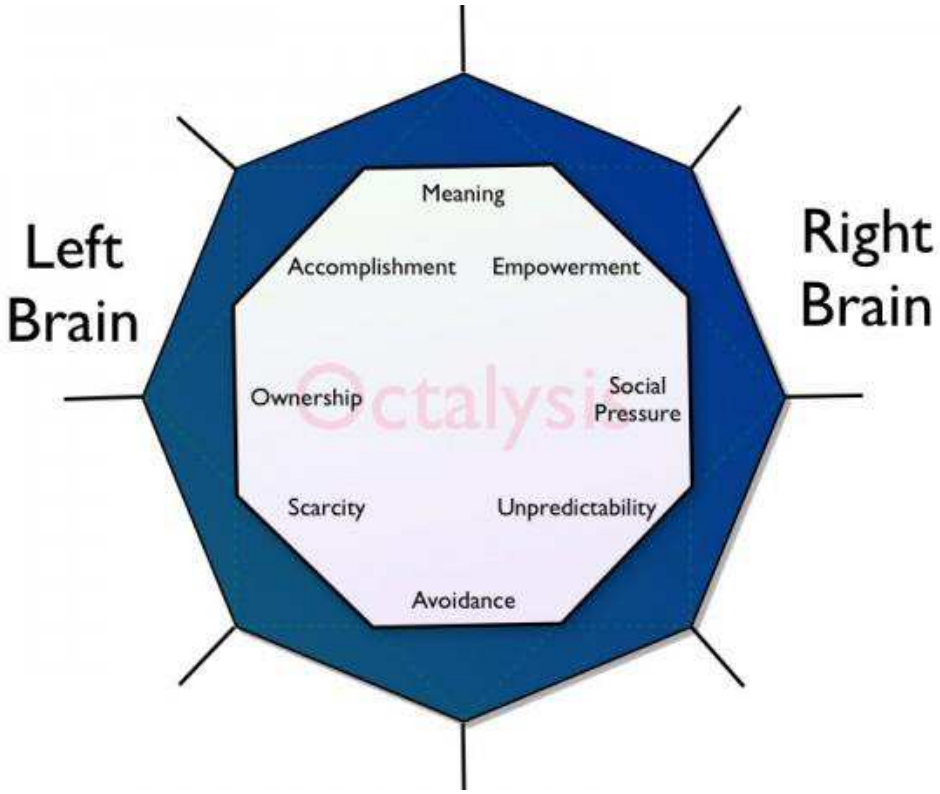


Abb. 3.2 Einteilung in Left-/Right Brain (Quelle: Chou (2015))

Auch die Anordnung der jeweiligen Bereiche hat eine tiefergehende Bedeutung. So finden sich auf der linken Seite des Octagons vor allem die Bereiche, die auch die linke Gehirnhälfte ansprechen. Diese ist hauptsächlich für die Logik, Kalkulationen und Besitz zuständig.

Dementsprechend befinden sich rechts die Bereiche, die die rechte Gehirnhälfte vorwiegend ansprechen. Diese ist für Kreativität, Selbstdarstellung und soziale Aspekte zuständig.

Dabei ist zu erwähnen, dass aus Forschungssicht eine solche links/rechts Einteilung der Bereiche nicht haltbar ist. Diese Einteilung wurde primär vorgenommen um das Framwork einfacher und effektiver zu gestalten. So wurden durch die Einteilung die logischen von den emotionalen Bereichen getrennt.

3.3 White Hat/ Black Hat Einteilung

Die White Hat-/ Black Hat Gamification Einteilung (siehe Abb. 3.3) beruht auf die verschiedenen Motivationen, die einen Menschen antreiben.

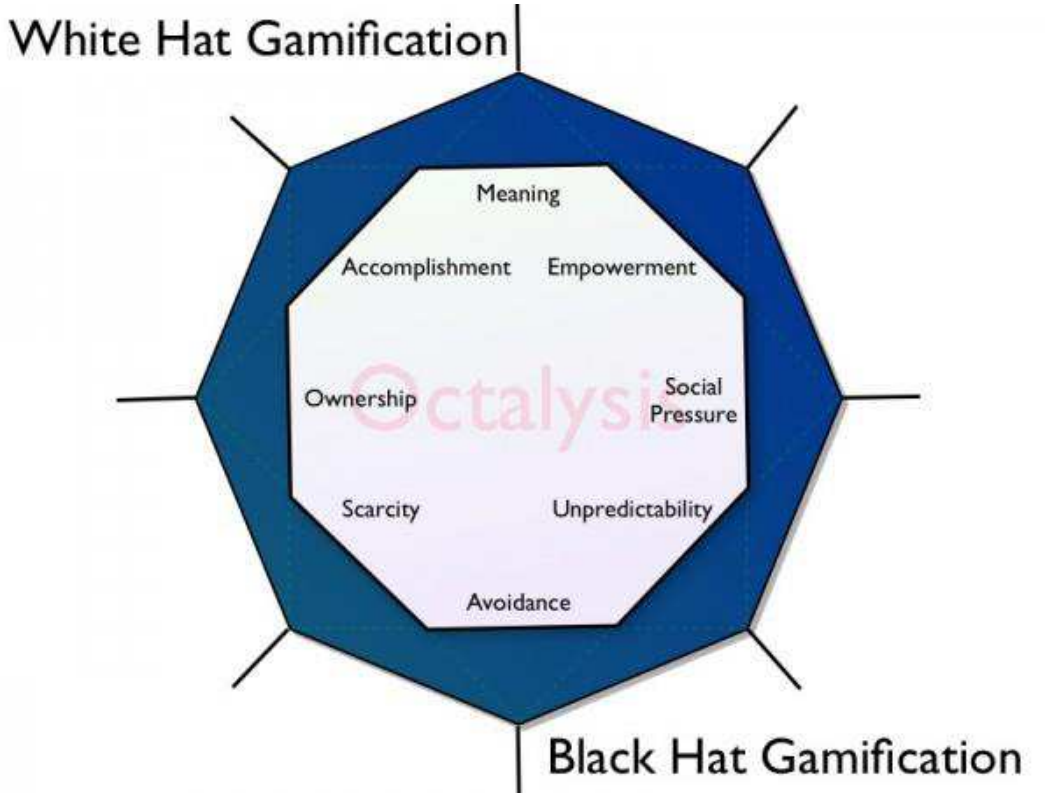


Abb. 3.3 White Hat-/Black Hat Gamification (Quelle: Chou (2015))

Der obere Bereich stellt die sogenannte White Hat Gamification dar und beruht auf die positven Motivationen, während der unter Black Hat-Abschnitt auf negative Motivation abzielt.

Dabei ist zu beachten, dass die negativen Motivationen keineswegs schlecht sein müssen. So sind die oft der Antrieb für Menschen ins Fitness-Studio zu gehen, gesünder zu essen oder pünktlich aufzustehen.

3.4 Octalysis Score

Ein gutes Gamification System muss nicht zwangsweise alle 8 Bereiche integrieren. Aber die Bereiche, die verwendet werden, müssen von hoher Qualität sein um Gamification erfolgreich umzusetzen. Einige sehr erfolgreiche Produkte setzen auf den sozialen Einfluss, während andere durch Angst sich am Markt behaupten.

Um dennoch unterschiedliche Systeme miteinander vergleichen zu können, wurde der Octalysis Score eingeführt.

Mit Hilfe von persönlichen Einschätzungen, Daten und Erfahrungen werden die 8 Bereiche mit einer Zahl zwischen 1-10 bewertet und dann jeweils quadriert. Die Summe aller 8 Bereiche ergibt dann den Octalysis Score.

3.5 Gamification Beispiel mit Octalysis

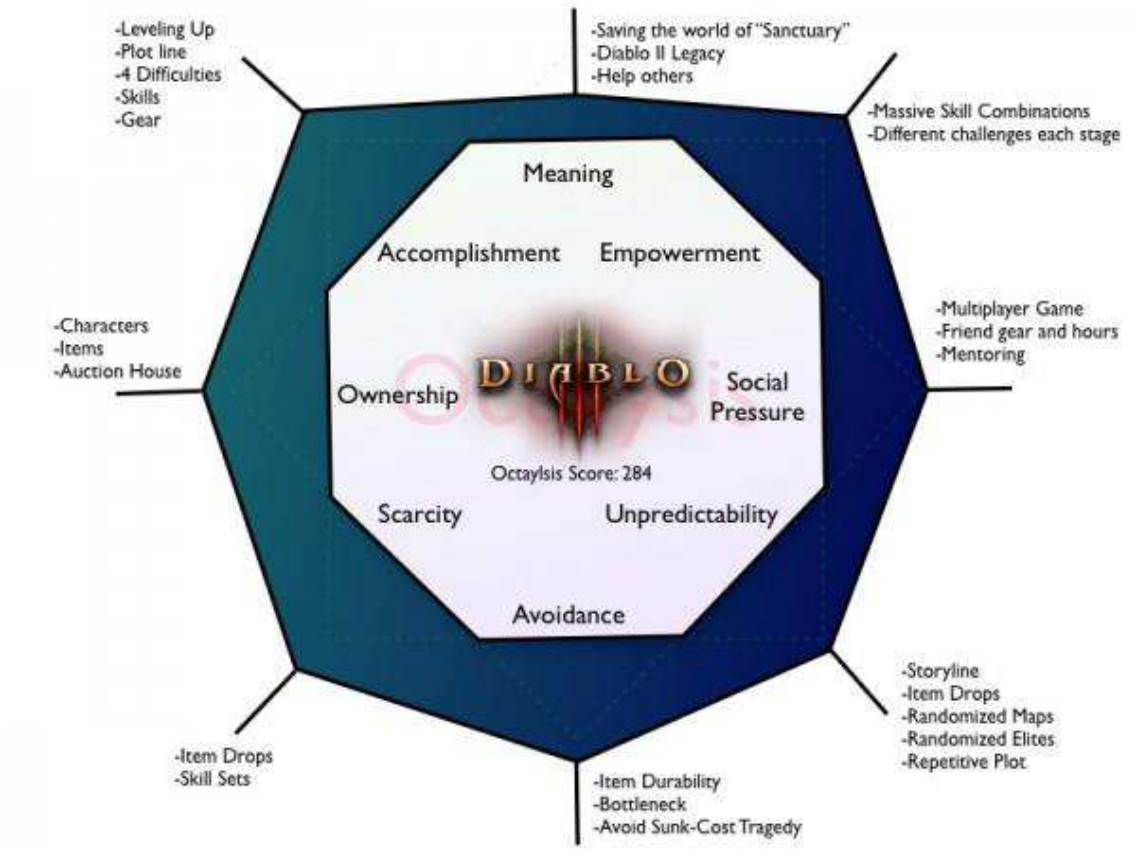


Abb. 3.4 Diablo 3 – Octalysis Score (Quelle: Chou (2015))

In Abbildung 3.4 sieht man die Octalysis Bewertung für das Spiel Diablo 3.

Doch wie gelangt man zu diesem Wert und der Grafik?

1. Die Werkzeuge/Methoden jedes Bereiches werden im Spiel identifiziert.

   (Bsp. Ownership: Characters, Items, Auction House)

2. Bewertung dieser Werkzeuge/Methoden an Hand von Erfahrungen, Daten und persönlichen Einschätzungen.

3. Um den Bereichswert zu erhalten wird die Bewertung quadriert. Je größer der Wert ist, desto weiter ist der Punkt von der Mitte entfernt.

4. Die Bereichspunkte aller 8 Bereiche werden miteinander verbunden.

Hier in dem Beispiel von Diablo 3 sind die Punkte alle relativ gleich weit vom Zentrum entfernt, so dass alle Bereiche sehr ausgeglichen vorkommen.

Wird nun das Beispiel von FarmVille betrachtet (Abbildung 3.5), so fällt auf, dass vor allem die linke Seite (left brain) und der Bereich der negativen Motivationen (black hat) angesprochen wird.

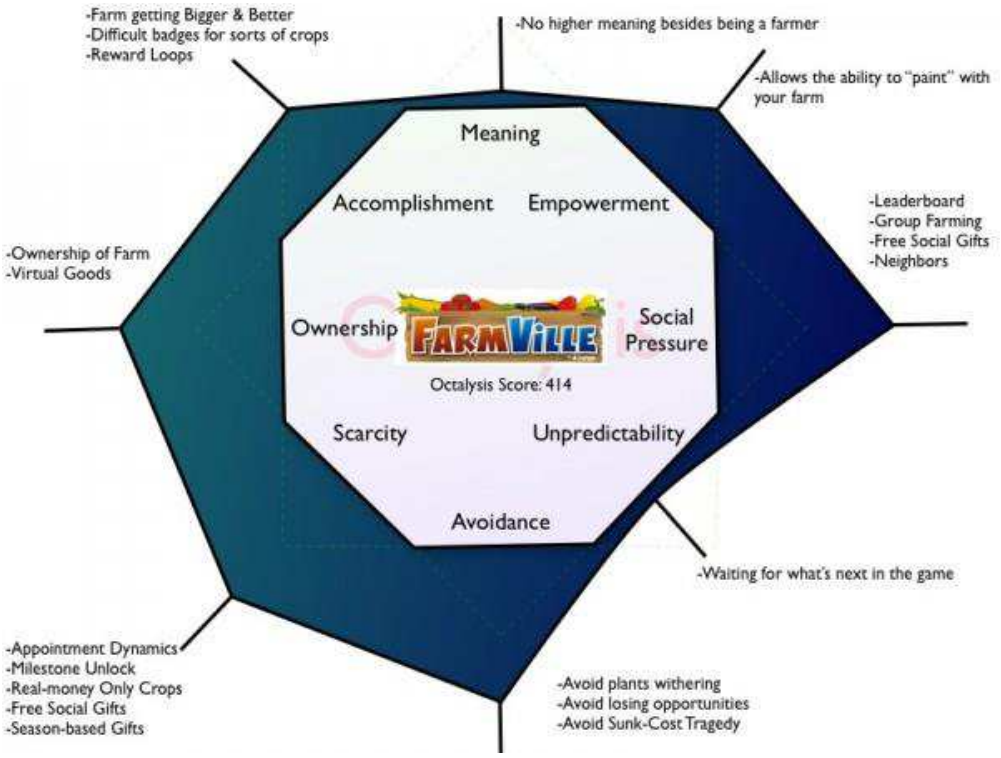


Abb. 3.5 FarmVille – Octalysis Score (Quelle: Chou (2015))

3.6 Weitere Level des Octalysis Frameworks

Die ganzen vorherigen Abschnitte bilden lediglich das Level 1 des Gamification Frameworks Octalysis.

3.6.1 Level 2 Octalysis Framework

Im 2. Level werden die 8 Kernbereiche auf die Zeitreise eines Spielers von der Entdeckung bis zum End-Game angewendet. Diese Zeitreise lässt sich in 4 Phasen gliedern (Abbildung 3.6).

1. Discovery: Die Gründe, warum Spieler sich überhaupt auf ihre Reise durch das Spiel begeben.

2. Onboarding: Die Phase in der den Spielern die Regeln und Möglichkeiten des Spieles näher gebracht werden.

3. Scaffolding: Die regelmäßigen Aktionen um ein bestimmtes Ziel zu erreichen (Quest, Stufenaufstiege, Events, etc.)

4. Endgame: Die Phase in der die Veteranen des Spieles bei Laune gehalten werden.

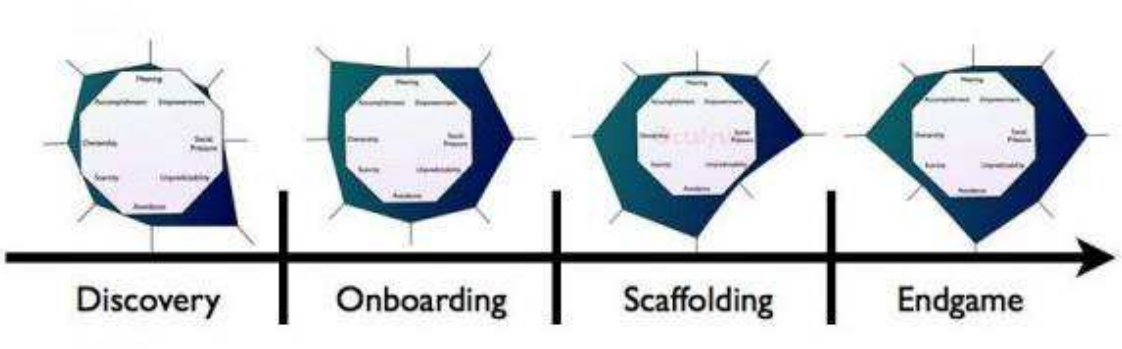


Abb. 3.6 Level 2 Octalysis Framework (Quelle: Chou (2015))

3.6.2 Level 3 Octalysis Framework

Im 3. Level des Octalysis Frameworks werden neben den 8 Bereichen, den 4 zeitlichen Phasen auch die unterschiedlichen Spielertypen in Betracht gezogen (Abbildung 3.7).

Die unterschiedlichen Spielertypen entstammen der Taxonomie von Richard Bartle (vgl. Kyatric (2013)).

1. Killers: Die Killer lieben das Drama und die Aufmerksamkeit und fallen auch häufiger durch Hacks, Cheats und Bots auf. Sie werden alles dafür tun, um einen der mächtigsten „Player versus Player“ (PvP) Charaktere zu haben.

2. Achievers: Sie lieben die Herausforderung und wachsen an den Aufgaben. Dabei können diese Aufgaben vom Spiel kommen als auch persönliche sein. Je schwieriger ein Ziel zu erreichen ist, desto größer ist die Freude.

3. Explorers: Die Explorer erkunden primär die Welt und kennen jede Ecke, Trick und Fehler im Spiel.

4. Socializers: Diese Gruppe von Spielern ist fokussiert auf Beziehungen mit anderen Spielern und der Umwelt. Das Spiel an sich rückt dabei oft in den Hintergrund. Engagieren sich häufig in der Community und managen Gilden.

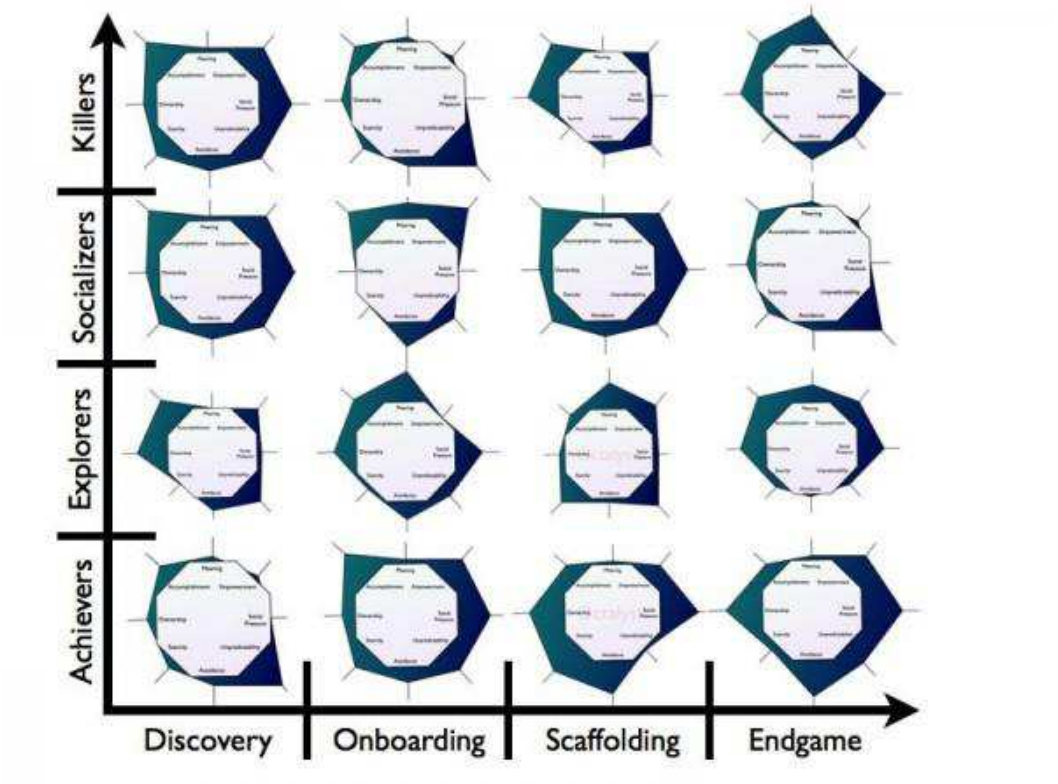


Abb. 3.7 Level 3 Octalysis Framework (Quelle: Chou (2015))

Laut dem Erfinder Yu-Kai Chou gibt es insgesamt 5 Levels, wobei Level 4 und 5 noch nicht öffentlich bekannt sind. Er schreibt, dass es erst eine Hand voll Menschen auf dieser Welt gibt, die Level 4 und 5 kennen. Für diese Levels benötige es ein tiefergehendes Verständnis von Design-Prinzipien und Analysen.

4 Potenziale und Anwendungsbereiche von Gamification

Motivationsunterstützung und das Vermitteln von Flow-Erlebnissen (Flow-Erleben ist ein besonders positiver Bewusstseinszustand – (Wurzler/Stenger) sind die primären Potenziale von Gamification (vgl. Blohm/Leimeister (2013)).

Dabei muss zunächst einmal zwischen intrinsischer und extrinsischer Motivation unterschieden werden:

Intrinsische Motivation „bezieht sich auf einen Zustand, bei dem wegen eines inneren Anreizes, der in der Tätigkeit selbst liegt, z.B. im Empfinden des Flow-Erlebens gehandelt wird (Maier).“

Die extrinsische Motivation „bezieht sich auf einen Zustand, bei dem wegen äußerer Gründe, d.h. wegen der Konsequenzen der Handlungsergebnisse (z.B. positive Personalbeurteilung, Gehaltssteigerung etc.), gehandelt wird (Maier).“

Traditionelle Anreizsysteme basieren sehr häufig auf der Erhöhung der extrinsischen Motivation wie z.B. Gehaltserhöhungen oder Bonuszahlungen. Diese Anreize haben oft nur einen kurzfristigen Einfluss, welcher zudem schnell bei Gewöhnung abnimmt. Durch IT-basierte Systeme ist es möglich die intrinsische Motivation anzusprechen (vgl. Blohm/Leimeister (2013)):

- **Steigern von Zufriedenheit:** Durch eine visualisierte, kontinuierliche Dokumentation der eigenen Fortschritte lassen sich realistische Ziele ableiten und dass lässt ein Gefühl von hoher Leistungsfähigkeit entstehen.

- **Vermitteln von Optimismus:** Elemente des Gamifications führen zu Erfolgserlebnissen (Stufenaufstiege, erfolgreiche Quest, etc.).

- **Ermöglichen sozialer Interaktionen:** Ein Austausch oder Wettbewerb zwischen den Mitgliedern führt zu sozialen Interaktionen.

- **Vermitteln von Bedeutung:** Elemente des Gamifications geben dem Nutzer das Gefühl, er sei Teilhaber einer Lösung, die über die eigenen Möglichkeiten hinausgeht.

Weiterhin besitzt Gamification die Möglichkeit sowohl Verhaltensweisen zu ändern als auch Lernprozesse effektiv zu fördern. Dabei werden Verhaltensänderungen mit positiven emotionalen Rückmeldungen verknüpft, die in das Unterbewusstsein vordringen und nach einer Trainingsphase internalisiert werden (vgl. Blohm/Leimeister (2013)).

Komplexe Aufgaben werden durch Gamification zerteilt in eine Vielzahl von Teilaufgaben und Meilensteine. Diese Teilaufgaben werden nach dem „Versuch- und Irrtum-Prinzip“ solange

wiederholt, bis die Problemstellung erfolgreich gelöst werden konnte oder der Nutzer eine neue Fähigkeitsstufe erreicht hat (vgl. Blohm/Leimeister (2013)).

„Gamification ermöglicht die Unterstützung und Transformation betrieblicher Wertschöpfungsprozesse (Blohm/Leimeister (2013))“. Haupteinsatzbereich ist derzeit vor allem im Marketing (Kundenbindung und Image) und bei der Unterstützung von Lernprozessen (vgl. Blohm/Leimeister (2013)).

5 Gamification Desgin Problem

Das Gamification-Elemente nicht von heute auf morgen ohne weitere Überlegungen in bestehende oder neue Software integriert werden kann, dürfte an Hand der Komplexität, die bereits durch das Framework und die Potenziale aufgezeigt wurden, nicht verwunderlich sein.

Zunächst einmal soll Gamification aus Betriebssicht den Wert eines Gesamtzieles maximieren. Wie in Kapitel 3.6.2 bereits aufgeführt, ist bei der Implementierung von Gamification-Elemente zu beachten, dass es unterschiedliche Spielertypen gibt. Jeder dieser Spielertypen reagiert auch unterschiedlich auf die Gamification-Elemente (vgl. Meder/Jain (2014)).

„We consider the gamification design problem as the problem of assigning each user a game desgin element that maximizes their expected contribution to achieve some goal“ (Meder/Jain (2014)).

Somit lässt sich ein generelles Modell erstellen, welches aus den folgenden Komponenten besteht (vgl. Meder/Jain (2014)):

1. Eine Aufgabe *T* die ausgeführt werden muss um ein Ziel zu erreichen
2. Eine Auswahl an Spieldesign-Elementen $g \sum G$
3. Eine Auswahl an Spielern $u \sum U$ welche Aufgabe T ausführen und dabei von *G* unterstützt werden
4. Eine aufgabenabhängige Grundwahrheit $f^*: U \rightarrow G$
5. Eine Funktion Klasse F die aus Funktionen mit der Form $f: U \rightarrow G$ besteht

Das Gamification Problem ist das Problem eine Funktion $f \sum F$ zu finden die so nah wie möglich an der Grundwahrheit f^* liegt. Die Grundwahrheit f^* ist eine Funktion, die jedem Nutzer *u* mit einem Spieldesign-Element *g* verknüpft, bei dem der Motivationszuwachs ein vordefiniertes Ziel zu erreichen maximal ist (vgl. Meder/Jain (2014)).

Um den Rahmen und die Komplexität der Ausarbeitung nicht zu sprengen, sind bei weiterem Interesse Formeln, Rechenwege und Matrizen der Quelle (Meder/Jain (2014)) zu entnehmen.

Weiterhin ist bei diesem Modell anzumerken, dass die Zeitphasen einer Anwendung, wie sie in Kapitel 3.6.1 zu finden sind, nicht berücksichtigt wurden.

Eine Möglichkeit sich diesem Problem anzunähern besteht darin, die Nutzer in Segmenten zu kategorisieren. In dem Fall von Gamification wäre dieses die vier unterschiedlichen Spielertypen. Sobald ein Nutzer in eine der Kategorien eingeteilt

wurde, können ihm die dazu passenden Gamification Elemente zugewiesen werden. Die Schwierigkeit liegt eindeutig in der Zuordnung der Nutzer. So können sich auch innerhalb der Spielertypen-Segmente die Charaktere und Ausprägungen, wie z.B. die Motivation, unterscheiden. Dieses könnte im schlimmsten Fall dazu führen, dass die Nutzer demotiviert werden (vgl. Meder/Jain (2014)).

Aus diesem Grund hat sich Gamification bisher vor allem im Marketing durchgesetzt. Dort wird die Zielgruppe oft durch das Produkt selber vorgegeben und diese tritt größtenteils als homogene Gruppe auf. Als Beispiel ist hier das NikeFuel zu erwähnen. Die Sensoren in den Nike-Schuhen senden die Laufleistung an Apple-Endgeräte und werden dort in NikeFuel umgerechnet und visualisiert. Für das Erreichen bestimmter Meilensteine werden Abzeichen verliehen. Diese können zum Beispiel in sozialen Netzwerken geteilt werden (vgl. Blohm/Leimeister (2013)).

Zunächst einmal ist die Gruppe vorbestimmt durch die Verwendung der Nike-Schuhe mit Sensoren als auch Apple-Endgeräte. Weiterhin verbindet die Gruppenmitglieder die Eigenschaft, dass sie (gerne) Laufen gehen. Damit sind die grundlegenden Eigenschaften aller Mitglieder identisch. Und selbst wenn kein Interesse besteht, kann man immer noch Laufen ohne die Synchronisation mit den mobilen Endgeräten zu nutzen.

6 Empfehlungen an die Projektgruppe

Nachdem die wichtigsten theoretischen Inhalte in den vorgegangen Kapiteln aufgearbeitet wurden, folgt nun die Empfehlung für unsere Projektgruppe.

Zunächst einmal muss festgehalten werden, dass es sich bei dem Thema Gamification um ein sehr umfassendes als auch komplexes Thema handelt. Durch die bloße Implementierung von Levels und Auszeichnungen ohne weiterführende Intentionen wird keine zusätzliche Motivation geschaffen (vgl. Chou (2015)).

Um Gamification-Elemente wissenschaftlich und begründet einzusetzen, müssten laut dem Gamification Design Problem (Kapitel 5) zunächst einmal die Nutzer genauer analysiert und kategorisiert werden. Dieses ist auf Grund von zeitlichen als auch personellen Ressourcen im Rahmen der Projektarbeit nicht möglich.

Dementsprechend sollte die Projektgruppe nicht den Anspruch erheben, eine wissenschaftlich fundierte Gamification-IMP zu entwickeln.

Nichts desto trotz besteht die Möglichkeit einfache Gamification-Elemente in die Plattform zu integrieren.

Aus den acht Bereichen des Frameworks würde ich folgende Elemente für eine gruppeninterne Diskussion empfehlen:

- Ein ausführliches Profil (Avatar, Lebenslauf, Profil/Reward-Übersicht)
- **Accomplishment:** Punkte, Abzeichen, Levelaufstiege, Progress Bar, Step-by-Step-Tutorial
- **Ownership:** Avatare, virtuelle Güter (ggf. Investitionen in Ideen)
- **Scarity:** Feedback, Count Down
- **Unpredictability:** Easter Eggs, Mini Quest, Random/Sudden Rewards
- **Social Influence:** Group Quest, „Thank-You“-Economy, Mentorship
- **Empowerment:** Meilensteine, Instant-Feedback

7 Fazit und Ausblick

Wie in der Empfehlung an die Projektgruppe schon festgehalten, handelt es sich bei dem Thema Gamification um ein sehr ausführliches und komplexes Thema. Das Framework von Yu-Kai Chou deckt dabei schon sehr viele Bereiche ab, ist aber bisher erst leider bis Level 3 veröffentlicht. Die Schwierigkeit bei der Implementierung liegt in dem Gamification Design Problem zu Grunde. Die Nutzer haben unterschiedliche Charaktere und reagieren somit auch auf Gamification Elemente unterschiedlich. Während Person A durch Element X motiviert wird, kann dieses bei Person B und Element X schon komplett anders sein. Aus diesem Grund ist Gamification vor allem im Marketing-Bereich präsent, da dort oftmals durch das Produkt selber die Zielgruppe, die oft homogen auftritt, vorgegeben ist.

Für die Zukunft ist Gamification auf jeden Fall als ein spannendes und auch erfolgsversprechendes Themenfeld anzusehen. Der eco-Trend Report 2020 bekräftigt diese Vermutung. Die dort befragten IT-Experten gehen davon aus, dass sich Gamification auch auf anderen Bereiche ausbreitet.

Das Gamification Design Problem dürfte dabei wohl die größte Hürde spielen. Aber durch weitere Forschungen, detaillierte Analysen und Entwicklungen bin ich optimistisch, dass dieses Problem auf ein Minimum reduziert werden kann.

Literaturverzeichnis

Blohm, L.; Leimeister, J. (2013): Gamification- Gestaltung IT-basierter Zusatzdienstleistungen zur Motivationsunterstützung und Verhaltensänderung. Wirtschaftsinformatik, Heft 4/2013, S. 275-278 DOI 10.1007/s11576-013-0368-0

Burke, B. (2014): Gartner Redefines Gamification http://blogs.gartner.com/brian_burke/2014/04/04/gartner-redefines- gamification/

Chou, Y. (2015): Octalysis: Complete Gamification Framework http://www.yukaichou.com/gamification-examples/octalysis-complete-gamification-framework/

Diercks, J. (2014): eco-Verband: Gamification macht sich im Alltag breit http://www.heise.de/ix/meldung/eco-Verband-Gamification-macht-sich-im-Alltag-breit-2195620.html

Fitz-Walter, Z. (2013): A brief history of gamification http://zefcan.com/2013/01/a-brief-history-of-gamification/

Google Trends (2015): Gamification bei Google Trends http://www.google.com/trends/explore#q=Gamification&cmpt=q&tz=

Kyatric (2013): Bartle's Taxonomy of Player Types (And Why It Doesn't Apply to Everything) http://gamedevelopment.tutsplus.com/articles/bartles-taxonomy-of-player-types-and-why-it-doesnt-apply-to-everything--gamedev-4173

Maier, G.: Gabler Wirtschaftslexikon – intrinsische Motivation http://wirtschaftslexikon.gabler.de/Definition/intrinsische-motivation.html

Maier, G.: Gabler Wirtschaftslexikon – extrinsische Motivation http://wirtschaftslexikon.gabler.de/Definition/extrinsische-motivation.html

Meder, M.; Jain, B. (2014): The Gamification Design Problem arXiv:1407.0843v1 [cs.HC]

Wurzler, M; Stenger, P: Flow & Usability – Website zur Magisterarbeit von Marco Wurzler und Philipp Stenger http://www.flow-usability.de/flowerleben.htm

B.1.6. Kommunikation und Konfliktmanagement

Kommunikation und Konfliktmanagement

Schriftliche Ausarbeitung
im Rahmen der Projektgruppe IMPACT

Themensteller:	Prof. Dr.-Ing. Jorge Marx Gómez
Betreuer:	Dr. Joachim Kurzhöfer Dipl.-Math. Jens Siewert Stefan Wunderlich, M. Sc.
Vorgelegt von:	Tobias Kromke Hinterm Grambker Dorfe 5 28719 Bremen Tobias.Kromke@uni-oldenburg.de
Abgabetermin:	22. Juni 2015

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Kommunikation stellt ein Bindungsglied zwischen Individuen dar mit dem Ziel Informationen zu vermitteln. Dies gelingt unter Umständen nicht immer eindeutig, was zu Konflikten führen kann. Diese können in unterschiedlichen Formen und Ausprägungen auftreten. Da ein Konflikt zu einem Einbruch der Produktivität und dem Wohlbefinden innerhalb einer Gruppe führen kann, sollten diese frühzeitig erkannt und behandelt werden. Außerdem besteht die Möglichkeit, ihnen durch bewusste Kommunikation vorzubeugen. Jedoch kann nicht jeder Konflikt verhindert werden. In solchen Fällen sind diese gezielt zu behandeln. Hierzu kann das Konfliktmanagement herangezogen werden. Die leitende Frage dieser Arbeit richtet sich daher darauf, wie innerhalb der Projektgruppe IMPACT Konflikten vorgebeugt werden kann und wie diese im Falle eines Auftretens schnell und effizient behandelt werden können.
Mit dieser Intention wird zunächst ein Bewusstsein für die Kommunikation geschaffen. Hierfür werden zwei Modelle sowie die Axiome der Kommunikation nach Watzlawick [WBJ74] eingeführt. Im weiteren Verlauf wird zunächst die Bedeutung eines sozialen Konfliktes erläutert, um darauf aufbauend zum Konfliktmanagement überzuleiten. In diesem Rahmen werden unterschiedliche Vorgehensweisen der Intervention dargestellt. Weiterhin werden aufgrund der in Abschnitt 3.3 vorgelegten Eskalationsstufen Alternativen zur Behandlung angerissen.
Im abschließenden Resümee wird auf Grundlage des dargestellten Sachverhaltes ein Leitfaden für das Verhalten im Konfliktfall für die Projektgruppe abgebildet.

2 Kommunikation

Die soziale Handlung der Kommunikation ist durch eine Interaktion zwischen Menschen geprägt. Somit kann geschlussfolgert werden, dass sie als Prozess mindestens so alt wie die Menschheit selbst ist [Rom14]. Die Interaktion, welche innerhalb der Kommunikation stattfindet, ist interpersonell und wird mittels eines sprachlichen Kodes realisiert [KKP90]. Dieser nutzt die verbale, paraverbale sowie non-verbale Dimensionen. Kommunikation findet somit immer zwischen mindestens zwei Beteiligten statt und bewegt sich auf mehreren Ebenen.
Es kann ein Modell genutzt werden, um diesen Sachverhalt zu verdeutlichen. Die bestehenden Kommunikationsmodelle sind dabei vom Sender-Empfänger-Modell aus der Informationstheorie geprägt [Her01]. Die Abbildung 1 bestehend aus Information Source, Transmitter, Receiver, Destination und Noise Source zeigt das besagte Modell [Sha49].

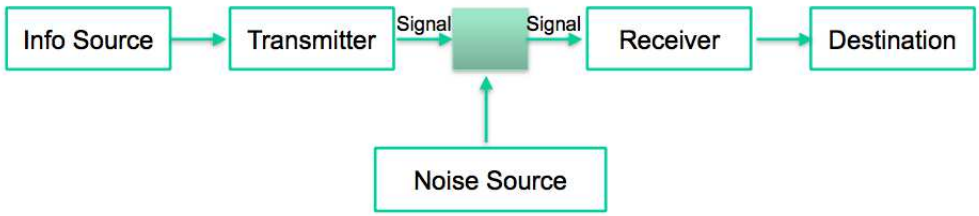


Abbildung 1: Shannon-Weaver-Modell aus der Informationstheorie in Anlehnung an[Sha49]

Das Verbindungsstück wird durch den Kanal realisiert, über welchen kommuniziert wird. Kernidee ist es, dass in der Information Source eine Nachricht ausgewählt wird, die über den Kanal versendet wird. Sie kann verschiedenartig aufgebaut sein. Beispielhaft kann sie aus Zahlen und Buchstaben bestehen. Diese Nachricht wird anschließend im Transmitter entsprechend des Receivers und des verwendeten Kanals in ein für sie passendes Signal umgewandelt. Während der Übermittlung des Signals durch den Kanal kann dieses durch Einflüsse verfälscht werden, was durch die Noise Source dargestellt wird. Die Aufgabe des Receivers besteht darin, das eingehende Signal zu empfangen und aus ihm die Ausgangsnachricht zu generieren. Diese reicht er an die Destination weiter, welche die in der Nachricht adressierte Person darstellt.
Für die menschliche Kommunikation lässt sich dieses Modell weiter abstrahieren. Somit kann sie auf den direkten Austausch einer Nachricht zwischen einem Sender und einem Empfänger herunter gebrochen werden. Für den weiteren Verlauf dieser Arbeit kann daher davon ausgegangen werden, dass das simpelste Kommunikationsmodell durch den Austausch von Nachrichten zwischen einer sendenden und einer empfangenden Partei beschrieben wird. Dabei sind die Rollen des Senders und Empfängers nicht fix und können somit getauscht werden.

2.1 Axiome der Kommunikation

Die fünf Axiome der Kommunikation von Watzlawick et al. [WBJ74] stellen einen Definitionsversuch dar, mit dem Hintergrund die Gesetzmäßigkeiten menschlicher Kommunikation möglichst formal untersuchen zu können. Ein Anspruch auf Vollständigkeit wurde dabei nicht erhoben, jedoch zeigt sich, dass sie bisher unverändert in Lehr- und Fachbüchern verwendet werden. Die dabei entstandenen Punkte können Abbildung 2 entnommen werden. Der erste Punkt beschreibt dabei, dass es einer Person nicht möglich ist nicht zu kommunizieren. Denn selbst wenn diese auf eine Frage nicht antwortet oder wegschaut,

1. Man kann nicht nicht kommunizieren
2. Jede Kommunikation hat einen Inhalts- und einen Beziehungsaspekt, derart, dass letzterer den ersteren bestimmt und daher eine Metakommunikation ist
3. Die (unterschiedliche) Interpunktion von Ereignisfolgen definiert die Natur der Beziehung
4. Es gibt digitale und analoge Kommunikation
5. Interaktionen sind entweder symmetrisch oder komplementär, je nachdem, ob die Beziehung zwischen den Partnern auf Gleichwertigkeit oder Unterschiedlichkeit beruht

Abbildung 2: Axiome der Kommunikation nach Watzlawick [WBJ74]

ist dieses bewusste Ignorieren der Kommunikation die Nachricht, die er seinem Gegenüber vermittelt. Sogar durch ein bewusstes Abblocken von Kommunikationen wird somit kommuniziert.
Das zweite Axiom deutet auf die unterschiedlichen Aspekte einer kommunizierten Botschaft hin. Somit steckt hinter jeder Aussage eine inhaltliche Nachricht sowie eine Beschreibung der Beziehung zwischen den Kommunizierenden. Unterhalten sich beispielsweise zwei Personen, wobei Person A der Person B mitteilt, dass die Aufgabe heute noch zu erledigen ist, so ist der Sachinhalt dieser Botschaft offensichtlich die schnelle Bearbeitung der Aufgabe. Der Beziehungsaspekt vermag dem Außenstehenden dabei nicht so klar zu sein. Angenommen Person A ist der Vorgesetzte von B, kann dies als eine direkte Zuweisung der Aufgabe von A an B verstanden werden. Stehen diese Personen jedoch anders zueinander in Beziehung, beispielsweise das Person B der Vorgesetzter von A ist, wäre eine Mitteilung von A an B als eine Art Information anzusehen, mit dem Wunsch, dass die Aufgabe bitte einem Arbeiter zuzuordnen ist. Der Beziehungsaspekt erweitert somit die inhaltliche Dimension der Botschaft, wobei die genaue Beziehung nicht immer eindeutig sein muss und somit zum korrekten Verständnis erst gebildet werden muss. In der Regel findet dies jedoch in kurzer Zeit statt.
Der dritte Aspekt behandelt die Tatsache, dass jeder in die Kommunikation Involvierte ihr durch seine Aktionen eine Struktur gibt. Es werden vergleichbaren Situationen im Rahmen unterschiedlicher Kulturen andere Strukturen genutzt. Das kann dazu führen, dass ein Partner die Handlungen des Anderen zum Anlass nimmt weiterführende Aktionen durchzuführen. Der Andere wiederum nimmt diese als Anlass für seinerseits neue Handlungen. Dies spitzt sich in soweit zu, dass die ausschlaggebende Aktion nicht mehr klar

ist und sich die Beteiligten gegenseitig vorwerfen angefangen zu haben und selber nur zu reagieren. Dies kann in einer iterativen Abfolge von Reaktionen münden, wodurch sich die Situation immer weiter zuspitzt. Umgangssprachlich ließe sich dies als einen Teufelskreislauf bezeichnen.
Beim vierten Axiom wird auf die Unterscheidung zwischen digitaler und analoger Kommunikation hingewiesen. Bestandteile ohne eine Abstufung sind dabei die digitalen. Sie sind logisch und exakt, wie beispielsweise ein Wort oder Symbol. Sie ist demnach emotionslos. Der emotionale Bestandteil einer Kommunikation findet somit auf analoger Ebene statt. Sie erlaubt Abstufungen, wobei ihre Ausdrucksmittel Mimik, Gestik oder Lautstärke sein können.
Als letztes ist es möglich bei Interaktionen zwischen symmetrisch und komplementär zu unterscheiden. Dies ist davon abhängig, ob die Partner eine gleichwertige oder unterschiedliche Beziehung zueinander besitzen. Symmetrisch wäre somit eine Interaktion zwischen Freunden, Partnern oder Kollegen. Komplementäre Beziehungen sind unter Anderem zwischen Eltern und Kindern, Lehrern und Schülern, Vorgesetzten und Untergeordneten gegeben.

2.2 Kommunikationsquadrat

Das Kommunikationsquadrat ist eine Weiterentwicklung des Organonmodells von Karl Bühler[Büh65]. Dabei werden die bestehenden Aspekte Ausdruck, Appel und Darstellung einer Nachricht durch Schulz von Thun um den Beziehungsaspekt erweitert, welcher sich aus dem zweiten Axiom des Abschnitts 2.1 herleiten lässt. Somit wird das aus Kapitel 2 bekannte Sender-Empfänger-Modell in soweit erweitert, dass die überbrachte Mitteilung vier Bereiche abdeckt. Diese sind nach Schulz von Thun der Sachinhalt, die Beziehung, der Appel und die Selbstoffenbarung. In Abbildung 3 wird verdeutlicht, wie die Nachricht vier unterschiedliche Facetten besitzt. Die Vermittlung auf den unterschiedlichen Ebenen kann entweder offen oder verdeckt stattfinden, weiterhin können diese Informationen bewusst oder unbewusst versendet und empfangen werden.
Der Sachinhalt beschreibt den reinen Inhaltsaspekt der Botschaft ohne weitere Intentionen. Er ist somit frei von jeglichen Gefühlen oder ähnlichem.
Auf Appel-Ebene vermittelt der Sender dem Empfänger was er von diesem erwartet. Dabei kann diese Botschaft gezielt versteckt gesendet werden um zu vermeiden, dass der Empfänger eine Gegenleistung fordert. Entgegen des Sachinhaltes beschreibt der Beziehungsaspekt die emotionalen Bestandteile der Nachricht. Zusätzlich kann aus ihm abgeleitet werden in welchem Verhältnis die Kommunikationspartner zueinander stehen.

Abbildung 3: Kommunikationsquadrat nach Friedmann Schulz von Thun[SvTRS00]

Die Selbstoffenbarung kann vom Sender bewusst in Form einer Selbstdarstellung oder unbewusst durch seine Körpersprache erfolgen. Zuweilen ist es möglich, dass dieser Bestandteil der Nachricht dem Empfänger deutlicher bewusst wird als dem Sender.

3 Konflikt

Um eine effiziente Behandlung von Konflikten durchführen zu können, muss zunächst festgehalten werden, ab welchem Zeitpunkt von einem solchen die Rede sein kann. Ein sozialer Konflikt lässt sich nach Glasl [Gla04] als: „eine Interaktion zwischen Aktoren (Individuen, Gruppen, Organisationen, usw.), wobei wenigstens ein Aktor eine Differenz bzw. Unvereinbarkeiten im Wahrnehmen und Denken bzw. Vorstellen und im Fühlen und im Wollen mit dem anderen Aktor (den anderen Aktoren) in der Art erlebt, dass beim Verwirklichen dessen, was der Aktor denkt, fühlt oder will eine Beeinträchtigung durch einen anderen Aktor (die anderen Aktoren) erfolge“ beschrieben.
Die wichtigen Aspekte dieser Definition sind nach den Autoren, dass eine Kommunikation zwischen den beteiligten Parteien stattfinden, wobei ihre Aktionen aufeinander bezogen sind. Weiterhin ist herauszustellen, dass die Unvereinbarkeit nicht von beiden Akteuren erlebt werden muss. Ausreichend ist demnach, wenn sie bei mindestens einem feststellbar ist. Außerdem ist zu berücksichtigen, dass auch wenn diese Differenz auf der kognitiven Ebene (Denken/Vorstellen/Wahrnehmen) vorzufinden ist, eine verbale Kommunikation benötigt wird, die als Anstoß dieser erkennbar ist. Somit kann kein sozialer Konflikt rein auf Tatsache des Denkens oder Vorstellens ohne reale Handlung entstehen. Zusätzlich muss sich die auftretende Unvereinbarkeit im Verlauf des Konfliktes auch auf die Gefühls- bzw. Willensebene ausbreiten. Dabei erlebt mindestens eine der beteiligten Parteien die Interaktion so, dass die Gegenpartei für die entsprechende Einschränkung verantwortlich ist.

Die Wahrnehmung einer Beeinträchtigung ist für diese Definition existentiell, da sich ohne ihr Erleben jegliche Interaktion als Konflikt beschreiben ließe. Darüber hinaus ist für einen sozialen Konflikt relevant, dass die (subjektiv) erlebte Unvereinbarkeit direkten Einfluss auf das Verhalten gegenüber der (vermeidlichen) Konfliktpartei besitzt.
Es kann geschlussfolgert werden, dass im Falle einer Unvereinbarkeit, die lediglich einer Partei bewusst ist, die Haltung und das Auftreten der anderen Partei ihr gegenüber zu einem direkten Konflikt zwischen beiden Parteien führen kann. Neben den sozialen Konflikten kann nach [Sch90] zwischen mehreren Arten unterschieden werden. Im Umfang dieser Unterscheidung wird der persönliche Konflikt vorgestellt, welcher zwar nicht als ein sozialer einzustufen ist. Dennoch ist er für das Verständnis von Reibungen wichtig, weshalb er für die Arbeit innerhalb der Projektgruppe erwähnt werden muss.

3.1 Konfliktarten

Konflikte können auf verschiedene Weisen bestehen. Diese sind davon abhängig, wie viele Akteure in ihn verwickelt sind und welche Seiten sie vertreten. Nach Schwarz kann zwischen persönlichen, Paar-, Dreiecks-, Gruppen-, Organisations, Institutions- und Systemkonflikten untergliedert werden [Sch90]. Die beiden Letzten werden dabei in dieser Arbeit nicht weiter betrachtet, da sie für die Interaktion innerhalb der Projektgruppe eine untergeordnete Rolle darstellen.
Persönliche Konflikte sind eben solche, denen sich jeder durch das Konkurrieren eigener Interessen stellen muss. Die Behandlung dieser inneren Spannungen ist somit ausschlaggebend für die Entwicklung der eigenen Persönlichkeit. Weiterhin können sie auf Grund der Reizungen Anlass zu Konflikten mit anderen Parteien geben, da man im Umgang mit diesen überempfindlich reagieren kann.
Unter Paarkonflikten sind eben solche einzuordnen, die zwischen Zwei Personen auftreten. Die Reibungen treten in diesem Fall auf, wenn die Interessen der Akteure als Individuen mit denen des Paares konkurrieren. Problematisch ist es hier, die Balance zwischen eigenen und gemeinschaftlichen Vorstellungen zu finden. Konflikte entstehen hierbei, wenn sich (im Extremfall) zumindest eine Partei dazu entschließt vollkommen die eigenen Interessen zu verfolgen, obwohl die Ziele des Paares dadurch nicht mehr erreicht werden können. In diesem Fall würde die Selbstverwirklichung eines Individuums der Beziehung des Paares stark schaden. Andererseits kann eine extreme Fokussierung auf die Interessen der Zweierkonstellation Einschränkungen für die Individuen bedeuten.
Bei einem Dreieckskonflikt wird das bestehende Paar um eine weitere Person erweitert. Der große Unterschied besteht nun darin, dass nun nicht weiter eine einzige Beziehung zwi-

schen den zwei Beteiligten besteht, sondern das nun drei Beziehungen zwischen den drei Akteuren gegeben sind. Damit verbunden ist ein Wandel, weg von den Personen als Kernaspekt der Konflikte, hin zu den Beziehungen als ausschlaggebenden Punkt. Denn selbst wenn die Einzelinteressen aller Parteien übereinstimmen, so können die unterschiedlichen Beziehungen zwischen jeweils zwei Akteuren miteinander in Konflikt geraten. Denkbar sind hier Situationen in denen Zwei Beteiligte um die Aufmerksamkeit der dritten Person kämpfen um so die Beziehung zu dieser vertiefen zu können.
Gruppenkonflikte stellen eine weitere Art dar. Bei ihnen wird die Dreierkonstellation um mindestens eine Partei erweitert. Somit erweitert sich die Anzahl an Beziehungen, die ein jedes Mitglied besitzt. Innerhalb einer Gruppe sind unterschiedliche Konfliktarten vorstellbar. So können in ihr wiederum Ausprägungen der zuvor vorgestellten Konflikte auftreten. Eine neue Art in der Gruppe ist der Führungskonflikt. In diesem richtet sich die zentrale Problemstellung auf die Festlegung eines klaren Anführers. Dieser kann unter Umständen nicht immer klar erkennbar sein, gerade vor dem Hintergrund, dass bei den Gruppenfunktionen eine Unterteilung zwischen zielorientiert und gruppenorientiert durchführbar ist. Zu den letzteren zählt beispielsweise die Pflege emotionaler Befindlichkeiten der Gruppe. Zu den zielorientierten Funktionen zählen unter Anderem die Weitergabe von Informationen und Setzung der Ziele. Für diese beiden Funktionen können innerhalb einer Gruppe zwei unterschiedliche Personen als Anführer herausstellen. Konflikte entstehen jedoch erst, wenn die Interessen von ihnen im Bezug auf die Gruppe konkurrieren.
Die nächsthöhere Instanz stellen die Organisationskonflikte. In einer größeren Organisation können sich ebenso Konflikte der zuvor beschriebenen Arten ausbilden. Erweitert werden diese durch den Umstand, dass Gruppen innerhalb einer Organisation untereinander einen Konflikt besitzen. Ein weiteres Problem, das auftreten kann, stellt der Abteilungsegoismus dar. In Erscheinung tritt dies, wenn Mitarbeiter eine Abteilung rein im Interesse dieser handelt, auch wenn es nicht mit den Ansichten der Firma übereinstimmt und so einen Konflikt erzeugt.

3.2 Konfliktstile

Neben den Konfliktarten kann das Verhalten der beteiligten Parteien nach Fischer und Wiswede in unterschiedliche Stile unterteilt werden [FW09]. Diese greifen den in Abschnitt 2 angeschnittenen Sachverhalt auf, dass nicht jeder Mensch gleich kommuniziert. Daraus kann abgeleitet werden, dass im Falle eines Konfliktes die Akteure unterschiedlich reagieren können. Dabei kann grundsätzlich zwischen der Orientierung an den eigenen Interessen sowie der Orientierung an der Beziehung zur Konfliktpartei unterschieden werden. Abhängig

vom Einsatz für die eigenen Vorstellungen und Berücksichtigung derer der Konfliktpartei lassen sich vier Felder bestimmen.
Sind die Realisierung der eigenen Ziele sowie die Beziehung zum Konfliktpartner generell von geringem Interesse, kann von einer Vermeidungsstrategie gesprochen werden. Hierbei wird primär versucht einen Konflikt zu vermeiden, wofür die eigenen Ziele und der Beziehungsaspekt vernachlässigt werden. Verlagert sich der Fokus auf die Beziehung, bei gleichbleibend geringer Orientierung an den eigenen Wünschen, kann dieses als Nachgeben angesehen werden. Wohingegen eine Realisierung der eigenen Ziele bei sich Vernachlässigung des Beziehungsaspektes als ein Durchsetzen einer Partei anzusehen ist. Um allgemein problemlösend zu agieren sollte daher die Orientierung an selbst gesetzten Zielen sowie dem Verhältnis zur anderen Konfliktpartei eine hohe Ausprägung mit sich führen.

3.3 Phasenmodell der Eskalation

Neben der Unterscheidung nach Konfliktarten und -stilen lässt sich die Tiefe des Konfliktes anhand von Eskalationsstufen bestimmen. Glasl definiert dazu neun Stufen, welche in Abbildung 4 absteigend dargestellt sind [Gla04]. Sie können als immer tieferes hinein driften

1. Verhärtung
2. Debatte, Polemik
3. Taten statt Worte
4. Sorgen um Images und Koalition
5. Gesichtsverlust
6. Drohstrategien
7. begrenzte Vernichtungsschläge
8. Zersplitterung
9. gemeinsam in den Abgrund

Abbildung 4: Phasenmodell der Eskalation nach Glasl [Gla04]

innerhalb des Konfliktes, bei Betreten einer neuen Stufe interpretiert werden. So lässt sich ein Konflikt metaphorisch als Grube darstellen, wobei ein immer tieferes abrutschen in sie einen härteren Weg hinaus zur Folge hat. Die möglichen Eskalationsstufen sind dabei die Verhärtung, Debatte und Polemik, Taten statt Worte, Sorgen um Image und Koalition, Gesichtsverlust, Drohstrategien, begrenzte Vernichtungsschläge, Zersplitterung sowie die neunte Stufe gemeinsam in den Abgrund.
Die oberste Ebene stellt die Verhärtung dar, wobei diese sich von Außen betrachtet nur geringfügig von den gewöhnlichen Umgangsformen differenziert. Hier verhärten sich aufeinanderprallende Standpunkte, wobei sich weder feste Parteien noch Lager bilden. Weiterhin kann das Bewusstsein einer Spannung bei den Beteiligten eine hemmende Wirkung haben,

jedoch besteht die Überzeugung diese durch Gespräche lösen zu können. Die zweite Stufe wird als Debatte oder Polemik bezeichnet. In ihr radikalisieren sich die Konfliktparteien durch Aufkommen eines Schwarz-Weiß-Denkens. Hierbei kann es auch zu verbaler Gewalt kommen. Dabei besteht eine Diskrepanz zwischen dem Oberton und dem verwendeten Unterton in der Kommunikation. Zusätzlich versuchen sich die Parteien gegenseitig ab zu werten. Dabei wird auch nicht davor zurückgeschreckt, Dritte für seinen Standpunkt zu gewinnen, um so Zuspruch zu erlangen.
Führt das Reden scheinbar nicht weiter, folgen in der nächsten Stufe Taten anstelle von Worten. Die Konfliktparteien stellen sich dabei vor vollendete Tatsachen, wobei die Empathie scheinbar verloren geht. Die hier auftretende Diskrepanz besteht zwischen verbalem und nonverbalem Verhalten, wobei das nonverbale dominiert. Dies bürgt die Gefahr einer Fehlinterpretation, gerade weil ein Misstrauen herrscht und pessimistisch antizipiert wird.
In der Phase der Images und Koalitionen liegt der Fokus auf dem Werben von Anhängern, mit welchen eine Koalition gegen die Konfliktpartei geschlossen wird. Dabei versuchen sich die Seiten gegenseitig in negativ behaftete Rollen zu drängen, um deren Chance neue Unterstützer zu gewinnen zu behindern. Dabei werden auch Stereotypen und Klischees bedient, um Gerüchte bezüglich des Wissens und Könnens der anderen Partei zu streuen.
Die anschließende Phase wird als Gesichtsverlust bezeichnet, bei welchem direkte öffentliche Angriffe vorgenommen werden. Hier findet unter Vorwurf des Verrats eine inszenierte Demaskierung der Konfliktparteien statt, was zu einer Isolation aufgrund eines Ausstoßes aus der Gruppe herbeigeführt wird.
Darauf folgen Drohstrategien, bei denen sich die Beteiligten gegenseitig bedrohen und gegebenenfalls auch erpressen. Dadurch kann bei ihnen vermehrt Stress entstehen. Weiterhin kann es eine Gefahr für ihre Glaubwürdigkeit mit sich bringen, da diese an ihre Aktivitäten gebunden ist.
Die siebte Phase wird begrenzte Vernichtungsschläge genannt. Hier verliert die Kommunikation zwischen den Konfliktparteien ihre menschliche Qualität, was ein Denken in „Dingkategorien“zur Folge hat. Dabei werden Werte ins Gegenteil umgekehrt, sodass ein als relativ gering anzusehender eigener Schaden als Erfolg gewertet wird. Vorangetrieben wird dies noch durch die Tatsache, dass die Beteiligten Angriffe auf den Gegner als passende Antwort einordnen.
Kommt es zur Zersplitterung liegt der Fokus auf dem Stilllegen des gegnerischen Systems. Dabei wird versucht dieses zu isolieren um so wichtige Bestandteile auszuschalten und so zum Verfall zu führen.
Die absolute Konfrontation findet in der letzten Phase statt. Sie wird auch als gemeinsam in den Abgrund beschrieben, wobei ein Punkt erreicht ist, an dem die Konfliktparteien

nicht mehr zurückkönnen. In diesem Zusammenhang wird die eigene Vernichtung in Kauf genommen, sofern sichergestellt ist, dass der Konkurrent ebenfalls zerstört wird.

4 Konfliktmanagement

Im Rahmen des Konfliktmanagements können verschiedene Ansätze genutzt werden um Konflikte richtig zu behandeln. In diesem Kapitel werden dazu ausgewählte Ansätze vorgestellt. Zusätzlich wird eine Handlungsempfehlung gegeben, wie das Verhalten im Falle eines Konfliktes realisiert werden sollte.

4.1 Interventionen

Die Interventionsmöglichkeit bei der Konfliktbehandlung lassen sich nach Glasl auf zwei Ebenen untergliedern. Zum einen kann zwischen präventiven und kurativen Methoden differenziert werden. Weiterhin kann eine Unterscheidung von eskalierenden und de-eskalierenden Verfahren getroffen werden. Die Abbildung 5 veranschaulicht dies, so ergeben sich aus der Kombination der beiden Aspekte vier Felder, welche im Rahmen unterschiedlicher Situationen angewendet werden können.
Das erste Feld entspricht hier den präventiven de-eskalierenden Möglichkeiten. Kann ein Training in Kommunikationsmethoden oder die Vereinbarung von Informationsregeln dazu beitragen, Kommunikationsproblemen, die zu Konflikten führen können, vorzubeugen. In den Bereich der präventiven eskalierenden Verfahren kann das gezielte Ansprechen von Sorgen und Ängsten beispielsweise in Konfrontationssitzungen eingeordnet werden. Das verfolgte Ziel ist es hierbei zu verhindern, dass sich vorhandene Konflikte zwischen den Beteiligen festsetzen.
Dem entgegen stehen die kurativen Ansätze. Bei der de-eskalierenden Variante kann eine Rekonstruktion des Konfliktverlaufes umgesetzt werden. Hierbei sollten auch die unterschiedlichen Standpunkte der Beteiligten genau beleuchtet werden um sicherzustellen, dass ein Verständnis für die Wünsche der Gegenpartei erreicht wird.
Eskalierend kann hingegen versucht werden bei festsitzenden Konflikten aufzulockern. Eine mögliche Technik an dieser Stelle sind Rollenspiele, bei denen bewusst eine Dramatisierung und Übertreibung stattfindet. Werden für die Klärung Interessensvertreter für beide Seiten bestimmt, so können diese den durch ihnen vertretenen Akteur dazu ermutigen sich für seine Ansichten stark zu machen.

	de-eskalierend	eskalierend
präventiv	Vereinbarung von Informationsregeln um Kommunikationsproblemen vorzubeugen Training in Kommunikationsmethoden	gezieltes Ansprechen von Sorgen und Ängsten Konfrontationssitzung um zu verhindern, dass sich ein Konflikt festsetzt
kurativ	Rekonstruktion und Klärung des Konfliktverlaufes Klärung der unterschiedlichen Standpunkte	Bestehende festgesetzte Konflikte durch Rollenspiele dramatisieren und übertreiben Akteure ermutigen für Interessen einzusetzen durch Interessenvertreter

Abbildung 5: Interventionen der Konfliktbehandlung nach Glasl [Gla04]

4.2 Konfliktbehandlung

Bei der Bearbeitung von Konflikten sollte immer berücksichtigt werden, wie stark dieser ausgeprägt ist. So kann anhand der in Abschnitt 3.3 vorgestellten Eskalationsstufen eine Einteilung erfolgen. Für diese werden in [Gla04] unterschiedliche Behandlungsansätze vorgestellt. kann im Rahmen der ersten drei Stufen eine Moderation verwendet werden. Die moderierende Person vermittelt dabei lediglich. Das bedeutet sie gibt keine eigenen Vorschläge, sondern stellt einen formalen Rahmen für die Konfliktparteien. Weiterhin sorgt sie durch die Visualisierung der Sachverhalte und Klärung von Begriffen dafür, die Missverständnisse aufzuklären.
Im Umfang der dritten bis fünften Eskalationsstufe ist es möglich den Konflikt mittels einer Prozessbegleitung zu lösen. Hier greift die vermittelnde Person - anders als bei der Moderation - aktiv ein und hilft dabei Lösungen zu finden. Dazu werden die Parteien einzeln auf die Gegenüberstellung vorbereitet. Ziel dabei ist es, dass sie ihr eigenes Verhalten kritisch hinterfragen um so Einsicht und Motivation zu schaffen den Konflikt zielführend zu lösen.

Befinden sich die Parteien zwischen der Phase Sorgen um Images und Koalitionen bis hin zu begrenzten Vernichtungsschlägen, was durch die vierte bis siebte Stufe dargestellt wird, so reicht eine reine Prozessbegleitung unter Umständen nicht aus, wenn das Selbstbild der Beteiligten deutlich erschüttert ist. An dieser Stelle tritt die sozio-therapeutische Prozessbegleitung in Kraft. Der Unterschied zur einfachen Variante liegt darin, dass der beratende Anteil einem therapeutischen Ansatz weichen muss, um so Ängste abzubauen und neue Hoffnungen zu wecken die aussichtslos scheinende Situation dennoch lösen zu können.
Zwischen der fünften und siebten Stufe ist die Vermittlung eine mögliche Alternative zur sozio-therapeutischen Prozessbegleitung. Sie kann verwendet werden, wenn bei den Konfliktparteien der Wunsch geschaffen werden kann weitere Schäden zu vermeiden, eine gemeinschaftliche Einigung jedoch nicht in Aussicht steht. Für diesen Fall kann ein Vermittler herangezogen werden, welcher durch eine Person realisiert wird, der beide Parteien Vertrauen schenken. Dieser kann Vorschläge zur Bewältigung des Konfliktes machen oder unter Umständen den Vorschlag einer Partei als seinen eigenen ausgeben, um die Akzeptanz beim Konkurrenten zu erhöhen.
Zwischen der sechsten und achten Stufe kann ein Schiedsverfahren verwendet werden. Die Beendigung des Konflikts erfolgt an dieser Stelle durch eine Regulierung und Überprüfung des Verhaltens der Parteien. Die eingreifende Person ist dabei eine unbefangene, welche die Fakten dargestellt bekommt und auf Grundlage dieser eine Entscheidung trifft.
Macht es den Anschein, dass der Konflikt nicht gelöst werden kann, so ist der Machteingriff eine potentielle Variante innerhalb der letzten drei Stufen. Dabei zwingt eine unabhängige Instanz die Konfliktparteien dazu, sich anzupassen. Wenn überdies keine Bereitschaft durch die Parteien besteht den Konflikt zu lösen, so muss die Machtinstanz in der Lage sein seine Position zu behaupten und den Beteiligten die vorgeschriebenen Verhaltensweisen auf zu zwängen.

4.3 Verhalten im Konfliktfall

Das Ansprechen eines Konfliktes stellt den ersten Schritt zur Behandlung. Dazu haben Schwäbisch und Siems [SS74] Schritte definiert, welche die ersten der Abbildung 6 darstellen.
Der erste Schritt umfasst dabei das Anmelden einer Störung. Dabei sollte darauf geachtet werden, dass dieses nicht beschuldigend stattfindet. Zusätzlich sollte es weder provokativ, noch zu allgemein sein. Um dies zu vermeiden und seine Offenheit für neue Lösungen zu zeigen sollte daher konkret und zeitnah in der Ich-Form eine Störung angemeldet werden. Ist den Beteiligten bewusst, dass ein Problem vorliegt so sind im zweiten Schritt die Hin-

1. Anmelden einer Störung
2. Klärung der Bedürfnisräume
3. Formulieren von Wünschen
4. Entwicklung von Ideen
5. Be- und Auswertung der Ideen
6. Erprobung der ausgewählten Idee
7. Evaluation

Abbildung 6: Sieben Schritte zur Konfliktbehandlung

tergrundbedürfnisse beider zu klären. Dazu werden von beiden Seiten seine Bedürfnisse dargestellt, wobei die Andere jeweils diese akzeptiert und versucht nachzuvollziehen. Im Anschluss stellen die Parteien die Bedürfnisse des jeweils Anderen dar, um so sicherzustellen, dass diese richtig verstanden wurden und die Fähigkeit zu stärken sich in den Gegenüber hinein zu versetzen.
Der dritte Schritt beinhaltet die Formulierung von Wünschen. Hier definieren die Konfliktparteien jeweils ihre Wünsche an den Anderen. Dies verhindert wiederum falsche Erwartungen und schafft es den Fokus auf die tatsächlichen Probleme zu lenken. Ersatzprobleme können somit entlarvt werden, wodurch mehr Klarheit für den wesentlichen Streitpunkt erreicht wird.
Sind die Wünsche klar, so können Lösungen entwickelt werden, welche diesen entsprechen. Wichtig dabei ist es kreativ zu sein und keine Wertung vorzunehmen. Dazu könnte beispielsweise das Brainstorming verwendet werden. Dabei sind auch Ansätze erwünscht, die zunächst unsinnig erscheinen. Freude und Heiterkeit sind in diesem Prozess erwünscht, so können freie Fantasie-Lösungen Anstoß für eine wirkliche Umsetzung sein oder auch nur durch den Spaß dafür sorgen, dass sich die Teilnehmer intensiv mit der Aufgabe beschäftigen.
Ist die Erstellung von Lösungsansätzen abgeschlossen folgt im fünften Schritt die Bewertung dieser. An dieser Stelle ist es den Beteiligten erlaubt Einwände zu äußern, da das Ziel darin besteht einen Weg zu finden, der für beide Parteien akzeptabel ist. Das gegenseitige Respektieren ist hierbei ein wichtiger Bestandteil um eine vorläufige Lösung auszuwählen.
Nach der Auswahl folgt die Erprobung. Dazu wird ein Zeitraum definiert in welchem der Lösungsansatz erprobt wird. Die Festlegung einer Spanne gibt den Beteiligten dabei Sicherheit. Sie werden nicht vor einen endgültigen und abschließenden Weg gestellt, sondern sie erproben ihn nur in einem definierten Rahmen. Dies stärkt die Bereitschaft tatsächlich die vereinbarten Punkte umzusetzen.
Ist der Erprobungszeitraum abgeschlossen findet eine Evaluation statt. Hier werden die Erfahrungen beider Seiten dargestellt und ausgewertet. Dabei ergibt sich, ob eine Lösung

weiter verwendet werden kann oder angepasst werden muss. Das Verwerfen des bestehenden Weges und die Auswahl eines neuen ist in diesem Fall auch möglich.

5 Schlussteil

Die zentrale Fragestellung dieser Arbeit diente der Vorbeugung und Behandlung von Konflikten innerhalb der Projektgruppe IMPACT. Zu diesem Aspekt wurde zunächst Grundlagen der Kommunikation erläutert, um anhand dieser das Bewusstsein für Kommunikationsprobleme zu schaffen, da diese Auslöser für Konfliktsituationen sein können.
Hier wurde zunächst das Sender-Empfänger-Modell von [Sha49] vorgestellt, um es für den menschlichen Austausch von Informationen auf die einfachste Ebene herunter zu brechen. Dieses bestand aus einem Sender, welcher eine Nachricht an einem Empfänger übermittelt. Weiterführend verfeinert wurde dies durch die vier Seiten des Kommunikationsquadrats von [SvTRS00]. Es ergänzt das bekannte um vier Aspekte, die mit jeder Nachricht bewusst oder unbewusst überbracht werden. Neben den Modellen wurden die Axiome der Kommunikation von [WBJ74] vorgestellt, mit deren Hilfe diese Interaktion genauer definiert wurde.
Eine weitere Definition wurde im Zusammenhang mit dem sozialen Konflikt gegeben. Dieser wurde erläutert, da ein Bewusstsein für diese positiv für ihre Behandlung sein kann. Es wurden zunächst verschiedene Arten von Konflikten und deren Ausprägung dargestellt. Bei den Konfliktstilen wurde ein Bewusstsein dafür geschaffen, dass unterschiedliche Personen eben so verschieden auf Situationen reagieren können. Dies verdeutlichte die Tatsache, dass jeder Mensch individuell ist und somit nicht eine einheitliche Lösung für jeden Fall angeboten werden kann. Da sich Störungen jedoch anhand bestimmter Kriterien kategorisieren lassen können, wurde das Phasenmodell der Eskalation nach [Gla04] eingeführt. Hier wurden neun Eskalationsstufen vorgestellt, welche als immer tieferes Abdriften in den Konflikt verstanden werden können.
Im Umfang des Konfliktmanagements wurden diese neun Stufen aufgegriffen. Dabei wurden zu jeder Behandlungsmethoden nach [Gla04] vorgestellt und erläutert. Neben diesen wurden Interventionen zur Behandlung von Konflikten dargestellt. Hier ist zwischen eskalierenden und de-eskalierenden Methoden zu unterscheiden, welche entweder präventiv oder kurativ einzusetzen sind. In Abschnitt 4.3 wurden abschließend sieben Schritte nach (siems) (masc....) vorgestellt, wie sich die Mitglieder der Projektgruppe im Falle eines Konflikts verhalten sollten. Diese sind als Handlungsempfehlung anzusehen und werden durch einen Ansprechpartner innerhalb der Gruppe realisiert.

Literatur

[Büh65] Karl Bühler. *Die Darstellungsfunktion der Sprache*. G. Fischer, 1965.

[FW09] Lorenz Fischer and Günter Wiswede. *Grundlagen der Sozialpsychologie*. Walter de Gruyter GmbH & Co KG, 2009.

[Gla04] Friedrich Glasl. Konfliktmanagement. ein handbuch für führungskräfte, beraterinnen und berater. 8. aktualisierte auflage, 2004.

[Her01] Thomas Herrmann. Kommunikation und kooperation. In *CSCW-Kompendium*, pages 15–25. Springer, 2001.

[KKP90] Karlfried Knapp and Annelie Knapp-Potthoff. *Interkulturelle Kommunikation*. na, 1990.

[Rom14] Jan Rommerskirchen. Soziologie und kommunikation. In *Soziologie & Kommunikation*. Springer, 2014.

[Sch90] Gerhard Schwarz. *Konfliktmanagement: Sechs Grundmodelle der Konfliktlösung*. Gabler Verlag, 1990.

[Sha49] Claude Elwood Shannon. Communication in the presence of noise. *Proceedings of the IRE*, 37(1):10–21, 1949.

[SS74] Lutz Schwäbisch and Martin Siems. *Anleitung zum sozialen Lernen für Paare, Gruppen und Erzieher: Kommunikations-und Verhaltenstraining*. Rowohlt, 1974.

[SvTRS00] Friedemann Schulz von Thun, Johannes Ruppel, and Roswitha Stratmann. Miteinander reden: Kommunikationspsychologie für führungskräfte. *Reinbek: Rowohlt*, 2000.

[WBJ74] Paul Watzlawick, Janet H Beavin, and Don D Jackson. *Menschliche Kommunikation: Formen, Störungen, Paradoxien*. H. Huber, 1974.

Thema:

Motivation und Anreizsysteme

Seminararbeit
im Rahmen der Projektgruppe IMPACT

Abteilung Wirtschaftsinformatik 1:
Very Large Business Applications

Betreuer: Dr. Joachim Kurzhöfer, Dipl.-Math. Jens Siewert, Stefan Wunderlich MSc.
vorgelegt von: Holger Eichholz
E-Mail: holger.eichholz@uni-oldenburg.de

Abgabetermin: 22. Juni 2015

Inhaltsverzeichnis

Verzeichnis der Abkürzungen und Akronyme

AS	Anreizsystem(e)
Abb.	Abbildung
d.h.	Das heißt

Abbildungsverzeichnis

Tabellenverzeichnis

1 Ziel und Überblick

Diese Ausarbeitung im Rahmen des Seminars zur Projektgruppe „Impact“ beschäftigt sich mit den beiden Thematiken „Motivation“ und „Anreizsysteme“ und soll dabei einen Einblick in die beiden Thematiken geben. Der Fokus liegt dabei auf Motivation und Anreizsysteme im betrieblichen Umfeld, da dies bei der späteren Umsetzung des Innovationsmanagementsystems in der Projektgruppe relevant sein kann. Die Bezüge und Möglichkeiten bei außerbetrieblichen Bereichen der Thematiken, werden kurz erwähnt, aber es wird nicht intensiver drauf eingegangen, da dies für die Thematik der Projektgruppe „Impact“ weniger relevant ist.

Dazu beschäftigen sich zunächst zwei Kapitel getrennt voneinander mit den Thematiken. Dabei werden zunächst die Begriffe definiert, genauer betrachtet und Beispiele gegeben. Im Anschluss werden beide Thematiken in Verbindung gesetzt und Möglichkeiten des Einsatzes im Rahmen der Projektgruppe gegeben. Das abschließende Fazit gibt eine Empfehlung ab, wie insbesondere die Anreizsysteme eingesetzt werden können.

Das Ziel der Ausarbeitung besteht darin, sich grundlegend mit den beiden Thematiken zu beschäftigen, alle Beteiligten an der Projektgruppe auf einen gemeinsamen Stand zu bringen und die Möglichkeiten des Einsatzes in der späteren Umsetzung des Projektgruppenthemas aufzuzeigen.

2 Motivation

Dieses Kapitel beschäftigt sich mit der Thematik *Motivation*, dabei wird zunächst eine Definition gegeben, wichtige Begriffe genauer betrachtet, zwei bekannte Motivationstheorien vorgestellt und Beispiele gegeben.

2.1 Definitionen im Umfeld von Motivation

Der Begriff der Motivation wird von Gabler Wirtschafslexikon wie folgt definiert:

„Zustand einer Person, der sie dazu veranlasst, eine bestimmte Handlungsalternative auszuwählen, um ein bestimmtes **Ergebnis** zu erreichen und der dafür sorgt, dass diese Person ihr Verhalten hinsichtlich Richtung und Intensität beibehält. Im Gegensatz zu den beim Menschen begrenzten biologischen **Antrieben** sind Motivation und einzelne **Motive** gelernt bzw. in Sozialisationsprozessen vermittelt. Der Begriff der Motivation wird oft auch im Sinn von Handlungsantrieben oder **Bedürfnissen** verwendet“. (Maier, et al.)

Diese Definition zeigt auf, dass die Basis für die Erreichung von Zielen (Ergebnissen) auf den Motiven und Bedürfnissen einer betroffenen Person beruht. Diese können dabei auf angeborenen Antrieben oder von erlernten Handlungsantrieben beruhen. Die beiden wichtigsten Begriffe sind dabei die Motive und Bedürfnisse.

Der Begriff des Bedürfnisses kann auf zwei unterschiedliche Arten definiert werden. Im Bereich des Marketings ist ein Bedürfnis ein Wunsch, der auf dem Empfinden eines Mangels herrührt. Dabei werden zwischen natürlichen Bedürfnissen, gesellschaftlichen Bedürfnissen (Kollektivbedürfnisse) und Grundbedürfnisse unterschieden. (Piekenbrock, et al.) Auf die genaue Beschreibung der Bedürfnisse wird in Kap. 2.2.1 in Form der Maslowschen Bedürfnispyramide des amerikanischen Psychologen Abraham Maslow genauer eingegangen. Neben der Definition im Bereich des Marketings gibt es eine weitere im Bereich der Marktpsychologie/Arbeits- und Organisationspsychologie, dabei wird das Bedürfnis als Motiv bezeichnet.

Dabei bezeichnen Motive „zeitlich relativ überdauernde psychische Eigenschaften von Personen. Sie werden im Zuge der Sozialisation erworben und bilden ein verhältnismäßig stabiles System“. (Esch, et. al) Ein Motiv setzt sich dabei aus zwei Komponenten zusammen, zum Einen die aktivierende Komponente, dies können z.B. Emotionen sein und zum Anderen kognitive Komponente, wie z.B. das bewusste Anstreben von Zielen.

Eine genauere Unterteilung der Motive erfolgt nach *niederen Motiven* und *höheren Motiven*. Bei den niederen Motiven handelt es sich z.B. um Triebe und Emotionen, wie Hunger, Durst,

Schlaf oder Sexualität, die jeder Mensch in sich trägt. Unter den höheren Motiven sind z.B. soziale Motive oder das Streben nach Selbstverwirklichung zu verstehen.

Die Erfassung von Motiven erfolgt in der Regel über eine Befragung, da diese je nach aktueller Lebenssituation des Befragten unterschiedlich sein kann und Motive individuell sind. Mit den individuellen Motiven beschäftigt sich insbesondere das Marketing, indem auf die aktivierende Komponenten eingegangen werden und es somit einer Verstärkung der sozialen Motive kommt.

2.2 Bedeutende Theorien

Dieser Abschnitt zeigt die beiden wichtigsten Theorien im Zusammenhang mit der Motivation. Dies sind zum Einen die Maslowsche Bedürfnispyramide und zum Anderen die Zwei-Faktoren-Theorie von Herzberg.

2.2.1 Maslowsche Bedürfnispyramide

Die Bedürfnistheorie des amerikanischen Psychologen Abraham Maslow zählt zu den bekanntesten und weitverbreittesten Inhaltstheorien und wird insbesondere im Bereich des Marketings eingesetzt. Die Grundlage für diese Theorie bieten die umfangreichen Motivlisten des amerikanischen Psychologen Henry Murray. (Vgl. Wiedmann, 2006, S. 23ff.) Die mehrstufige Bedürfnistheorie wird meistens in einer Pyramide dargestellt (Vgl. Abb. 1). Neben dieser klassischen Darstellung von Maslow aus dem Jahr 1954 gibt es heutzutage Darstellungen, die um die Stufe *Internet* erweitert wurde.

Abbildung 1:Die Bedürfnispyramide nach Maslow (Entnommen aus: Universität Duisburg)[1]

[1] https://www.uni-due.de/edit/lp/motivation/maslow (aufgerufen am 27.5.2015)

Die Basis der maslowschen Bedürfnispyramide bilden die fundamentalen und physiologischen Bedürfnisse, dabei handelt es sich um alle elementaren Bedürfnisse, wie das Verlangen nach z.B. Hunger, Durst, Schlaf. Diese Bedürfnisse dienen zur Aufrechterhaltung des normalen Organismuskreislaufs, also jene Bedürfnissen, die zum Leben und Überleben des Menschen erforderlich ist. Sind diese Bedürfnisse erfüllt, dann folgen die Sicherheitsbedürfnisse, also das Verlangen nach langfristiger Herstellung und Aufrechterhaltung von „Struktur, Ordnung, Recht und Grenzziehung." (Wiedmann 2006, S. 24) In dieser Stufe wird nach Sicherheit, Schutz, Furcht, Ordnung und Verhaltensregelungen gestrebt. Dazu Zählen z.B. das Bedürfnis nach einem sicheren Arbeitsplatz oder einem Sparkonto. Die dritte Stufe sind die sozialen Bedürfnisse, die ein konkretes Verlangen nach Geselligkeit, Freundschaft und Sympathie umfassen, dazu gehört z.B. die Anerkennung in einem Verein oder im direkten Umfeld. Sind die Bedürfnisse in dieser Stufe ebenfalls erfüllt, folgen in der vierten Stufe das Bedürfnis nach Wertschätzung. Diese umfassen das Streben nach Anerkennung, Prestige, Einfluss, Beachtung, Wunsch nach Erfolg, Unabhängigkeit und Kompetenz. Diese Stufe lässt sich dabei in zwei Bereiche unterteilen, zum Einen zur Person selbst, zum Anderen zur Leistung.

Bei den Bedürfnissen dieser vier Stufen handelt es sich um Bedürfnisse, die auf Defizitmotiven basieren. Dem gegenüber stehen die Wachstumsmotive der fünfen Stufe, die die Selbstverwirklichungsbedürfnisse beinhaltet.

In seiner Bedürfnistheorie geht Abraham Maslow davon aus, dass ein Mensch diese hierarchische Darstellung von unten durchläuft. Maslow nimmt dabei an, welches Bedürfnis der individuelle Mensch verspürt, wenn dieser über längere Zeit in einer leeren Umgebung ist. Seine Theorie besagt, dass ein Mensch die Stufen durchläuft, aber die Bedürfnisse einer Stufe nicht zu 100% erfüllt sein müssen, damit er die nächste Stufe erreichen kann. Es hängt dabei immer vom individuellen Empfinden ab, welcher Prozentsatz ausreichend ist.

Obwohl diese Theorie sehr verbreitet ist, gibt es bisher keine empirischen Belege für Maslows Ansatz. Zudem werden in der Pyramide legendlich die Bedürfnisse beschrieben und keine Motivierungspotential genannt. Auch ist die grundsätzliche Hypothese der Motivabfolge nicht haltbar, da es Menschen in extremen Lebensbedingungen oft an den elementaren, physiologischen Bedürfnissen fehlst, aber gleichzeitig soziale Bedürfnisse befriedigt werden. Trotz all dieser Punkte gilt die Maslowsche Bedürfnispyramide unter Wirtschaftswissenschaftlern als die bekannteste Motivationstheorie.

2.2.2 Zwei-Faktoren-Theorie nach Herzberg (Vgl. Wiedmann 2006, S. 26ff.)

Eine weitere Theorie, die im Bereich der Motivation recht verbreitet ist, ist die Zweifaktorentheorie nach Herzberg. Dabei wird festgelegt, dass bestimmte Faktoren zur Unzufriedenheit (Hygienefaktoren) und andere zur Zufriedenheit führen können (Motivatoren).

Die Zwei-Faktoren-Theorie bezieht sich zwar auf die Arbeitszufriedenheit, aber hat gleichzeitig eine große Relevanz für die Arbeitsmotivation. Die Theorie basiert auf dem Bedürfnis der Selbstverwirklichung. Es findet eine Einteilung in zwei Kategorieklassen statt. Die erste Kategorie befasst sich dabei mit den Gegebenheiten im Arbeitsumfeld, diese werden als Hygienefaktoren bezeichnet. Dazu zählen Beziehungen, Status, Ansehen, Arbeitsplatzsicherheit oder Bezahlung. Fehlen diese Faktoren führt dies zur Unzufriedenheit. Dem entgegenstehen die Motivatoren, die zur Förderung der Entstehung von Arbeitszufriedenheit führen, dazu gehören z.B. Leistungserfolg, Anerkennung, Verantwortung, Aufstiegsmöglichkeiten. Besonders die Motivatoren führen zur Befriedigung nach Selbstverwirklichung. Die Motivatoren haben da einen großen Einfluss auf die Mitarbeiter. Die Theorie von Herzberg führe mit der Einteilung zur Überdenkung von motivierenden Anreizen gegenüber dem Mitarbeiter.

2.3 Intrinsischer und extrinsischer Motivation

Im Bereich der Motivation kann zwischen der intrinsischen und extrinsischen Motivation unterschieden werden. Die intrinsische Motivation „bezieht sich auf einen Zustand, bei dem wegen eines inneren Anreizes, der in der Tätigkeit selbst liegt, z.B. im Empfinden des Flow-Erlebens gehandelt wird“. (Maier) Es handelt sich also um eine Motivation, die vom inneren des Individuums kommt und zur Erreichung eines Ziels dient. Ein Beispiel dafür ist die Motivation, die erforderlich ist, wenn ein Marathonläufer bereits über 30km von 42,195km absolviert hat, ihm alles weh tut und am Liebsten den Lauf abbrechen würde. Er bringt jedoch die innerliche Motivation auf, um seinen inneren *Schweinehund* zu überwinden und sein Ziel, den Zieleinlauf zu erreichen.

Der intrinsischen Motivation steht die extrinsische Motivation gegenüber, diese „bezieht sich auf einen Zustand, bei dem wegen äußerer Gründe, d.h. wegen der Konsequenzen der Handlungsergebnisse […] gehandelt wird“. (Maier) Es handelt sich somit um eine Motivation, die von außen an das Individuum herangetragen wird. Auch hier gibt es ein Beispiel des Marathonläufers, der von seinem Trainer/Betreuer im Gespräch und über Anweisungen von außen motiviert wird, das gesetzte Ziel des Zieleinlaufs zu erreichen.

Die intrinsische Motivation wird oft als die Basis für kreative Lösungen angesehen und war früher eher negativ bewertet, da diese durch Anreize bzw. Belohnung vermindert wurde. Dieser Effekt tritt jedoch nur ein, wenn die handelnde „Person allein für die Ausführung einer Tätigkeit ohne Bezug zu einem Leistungskriterium belohnt wird.“ (Maier) Die intrinsische Motivation lässt sich durch Motivatoren steigern und somit das Motivationspotenzial der Tätigkeit erhöhen. Bei den Motivatoren handelt es sich z.B. „um Leitung, Anerkennung, Aufgabeninhalte, Verantwortung, Aufstiegsperspektive, sowie Wachstumsmöglichkeiten“. (Maier)

2.4 Motivationsmodell nach Staehle

Das Motivationsmodell nach Staehle ist ein einfaches Motivationsmodell, was jedoch die Zusammenhänge von Bedürfnis, Motiv, Anreize, Verhalten bis zur Zielerreichung übersichtlich darstellt. Generell handelt sich um eine Motivationstheorie, die versucht zu klären, wie und warum ein Mitarbeiter in einem Unternehmen eine bestimmte Verhaltensweise zeigt.

Das Motivationsmodel zeigt, dass die Basis ein Bedürfnis ist, dass durch ein Mangelempfinden hervorgerufen wird. Daraus bildet sich ein Motiv, was sich bereits auf die Zielerreichung auswirkt. In der Übereinstimmung von Anreiz und Motiv führt dies zu einer Aktivierung des Motives. Die Aktivierung führt zu einem bestimmen, zielgerichteten Verhalten, was im idealen Fall zu einer Zielerreichung führt. Ist ein Ziel erreicht, so wird ein neues Bedürfnis bzw. Motiv angesprochen.

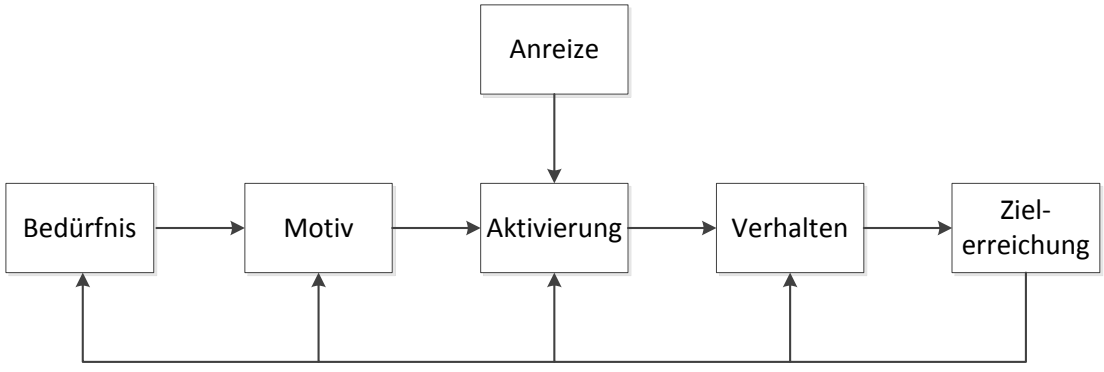


Abbildung 2:Motivationsmodell nach Staehle

(Entnommen aus: Huber 2014, S.15)

Dieses gesamte Motivationsmodell hat zum Ziel, dass durch die Ausrichtung auf Bedürfnisbefriedigung ein bestimmtes Verhalten hervorgerufen wird, was zur Zielerreichung führt. Die „Motivation resultiert demnach immer dann, wenn ein Mitarbeiter in einer bestimmten Situation Anreize wahrnimmt, die ein oder mehrere seiner Motive aktivieren bzw. zu deren Befriedigung beitragen.“ (Huber 2014, S.15) Somit ist es für ein Unternehmen besonders wichtig, dass es die

Bedürfnisse und Motive ihrer Mitarbeiter kennt, um somit gezielte Anreize geben zu können, damit sowohl die Ziele des Unternehmens, als auch des einzelnen Mitarbeiters erreicht werden.

Im Zusammenhang mit der Motivationstheorie findet eine Unterscheidung zwischen Inhalts- und Prozesstheorie statt. Dabei zieht die Inhaltstheorie auf die Frage ab, was einen Menschen zu einem bestimmten Verhalten antreibt bzw. erzeugt und aufrechterhält. Als Basis dafür wird z.B. die Maslowsche Bedürfnispyramide (vgl. Kap. 2.2) betrachtet. Aus dieser können gezielte Anreize abgeleitet werden. Der Inhaltstheorie steht die Prozesstheorie gegenüber, die der Frage nachgeht, wie ein Verhalten erzeugt, gelenkt, erhalten und abgebrochen werden kann. Dabei wird insbesondere die Wahrscheinlichkeit, bestimmte Leistungsziele und Handlungsergebnisse zu erreichen, betrachtet. Beide Theorien „verdeutlichen jeweils die besondere Relevanz, die der Auswahl verhaltenswirksamer Anreize im Rahmen der Gestaltung betrieblicher Anreizsysteme bzw. für die zielorientierte Beeinflussung des Mitarbeiterverhaltens zukommt“. (Huber 2014, S.16)

Auf die Thematik der Anreizsysteme wird im Kapitel 3 dieser Ausarbeitung genauer eingegangen.

2.5 Motivation im unternehmerischen Umfeld

Gerade im unternehmerischen Umfeld kommt der Motivation eine große Bedeutung zu. Ohne eine angemessene, passende Motivation kann das Unternehmen nicht die gesetzten Ziele erreichen. Aus diesem Grund sind verschiedene Faktoren erforderlich, damit die Mitarbeiter motiviert werden, die gewünschte Leistung zu erreichen und das gewünschte Leistungspotenzial auszuschöpfen. Die kann zum Einen über persönliche Mitarbeitergespräche erreicht werden und zum Anderen über Anreize, die die Motive der Mitarbeiter berühren. Diese reichen von mehr Freiheiten z.B. bei der Arbeitszeit, über Aufstiegsmöglichkeiten und Anerkennung bis zu monetäre Prämien. Dieser Abschnitt befasst sich mit der Frage, welche Möglichkeiten es zur Motivation der Mitarbeiter gibt.

2.5.1 Erhaltungs- und Entfaltungsfaktoren

Die Abb. 3 zeigt einen Überblick über die vorhandenen Erhaltungs- und Entfaltungsfaktoren in der Arbeitswelt. Bei den Entfaltungsfaktoren handelt es sich um Randbedingungen, die eine Motivation in der Arbeitswelt hervorrufen. Dabei handelt es sich um die Personalpolitik, Organisations- und Führungsformen z.B. in Arbeits- oder Gruppenarbeit oder Austreten von Vorgesetzen, Arbeitsbedingungen z.B. in Bereitstellung von Pausenräumen, Getränken und die Aus-

wirkung auf das Privatleben z.B. über ein Gleitzeitkonto oder Schichteinteilung. Zudem sind das Gehalt und die zwischenmenschlichen Beziehungen Erhaltungsfaktoren, die als extrinsische Motivation durch das jeweilige Unternehmen beeinflusst werden können und sich auf das gesamte Unternehmen auswirkt. Zu den Erhaltungsfaktoren kommen die Entfaltungsfaktoren, die sich auf einzelne Mitarbeiter beziehen, dazu zählen die Leistungsbestätigung z.B. regelmäßige Mitarbeitergespräche, berufliches Vorwärtskommen z.B. Schulungsmöglichkeiten und die zugeteile Aufgaben, Kompetenz und zugeordnete Verantwortung.

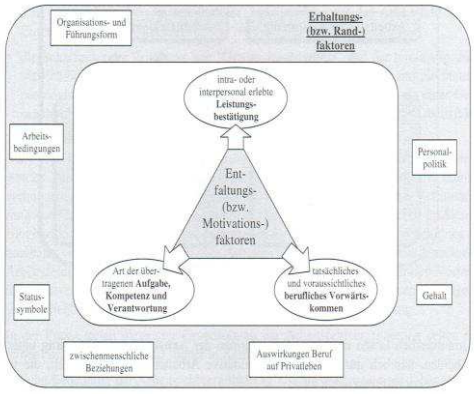


Abbildung 3:Überblick der Erhaltungs- und Erfolgsfaktoren in der Arbeitswelt (Entnommen aus: Link 2011, S. 95)

Die Erhaltungs- und Entfaltungsfaktoren sind für die Motivation im unternehmerischen Umfeld wichtig, aber werden im Rahmen dieser Ausarbeitung nicht genauer betrachtet.

2.5.2 Umfrage zu Motivationsfaktoren

Die Abbildung 4 zeigt das Ergebnis einer Umfrage der Universität Trier. Dabei wurden Mitarbeiter aus einem Stahlunternehmen nach ihrer intrinsischen und extrinsischen Motivation befragt und diese konnten die Motivationsfaktoren gewichten. Bei der Auswertung fällt auf, dass die Kategorien, bis auf die flexible Arbeitszeit eher als positiv bewertet wurden.

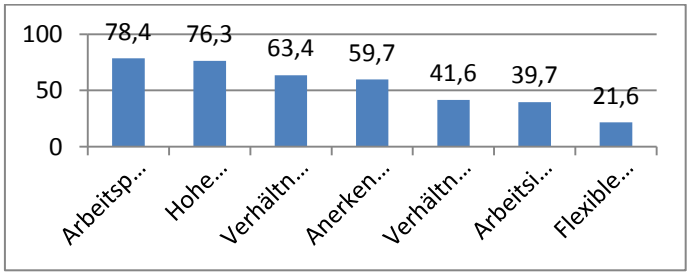


Abbildung 4:Häufigkeiten von „sehr wichtigen“ Motivationsfaktoren (in %)

(Entnommen aus: Wickert 2004, S.60)

Mit der Arbeitsplatzsicherheit und dem hohen Gehalt haben für die befragten Mitarbeiter zwei extrinsische Motivationsfaktoren die höchsten Wichtigkeiten. Dies kann als Beleg für die Wichtigkeit der extrinsischen Motivation gesehen werden. Wenn man die Ergebnisse der Umfrage genauer betrachtet zeigt sich, dass diese Motivationsfaktoren besonders in den unteren und mittleren Einkommensgruppen als sehr wichtig angesehen werden. Aus diesem Ergebnis kann die Folgerung geschlossen werden, dass das hohe Gehalt zwar ein wichtiger Motivationsfaktor ist, aber eben nicht ausschließlich betrachtet werden muss.

3 Anreizsysteme

Dieses Kapitel beschäftigt sich mit der Thematik *Anreizsysteme*. Dabei wird zunächst eine Definition gegeben und wichtige Begriffe erläutert. Anschließend werden inner- und außerbetriebliche Anreizsysteme betrachtet, jedoch nur die innerbetrieblichen AS intensiver, da diese für die Projektgruppe „Impact“ von Relevanz sind. Im Anschluss werden Anforderungen aufgezeigt, die bei der Entwicklung eines Anreizsystems erforderlich sind. Der Abschluss dieses Kapitels gibt einige Beispiele, wie Anreizsysteme in der Praxis eingesetzt werden.

3.1 Definition und wichtige Begriffe

Der Begriff des Anreizsystems wird von Dr. Robert Huber, wie folgt definiert. „Unter Anreizsystemen werden Systeme verstanden, welche *Bemessungsgrundlagen* mit *Anreizen* durch *Belohnungsfunktionen* verknüpfen, um das Verhalten der Mitarbeiter (*Adressaten*) unternehmenszielkonform zu steuern“. (Huber 2014, S. 13)

Die wichtigsten Begriffe bei der Definition sind Anreiz, Bemessungsgrundlage, Belohnungsfunktion, Ausschüttungspolitik und Adressatenkreis. Diese werden in den folgenden Abschnitten erklärt und definiert.

Der Begriff des Anreizes lässt sich dabei auf unterschiedliche Weise definieren. Nach der Definition von Dr. Robert Huber werden als Anreiz „die positiven (Belohnung) oder negativen (Bestrafung) Konsequenzen bezeichnet, die Folge einer erwünschten bzw. unerwünschten Leistung eintreten und mindestens ein Motiv des Betroffenen tangieren.“ (Huber 2014, S. 14) Somit soll durch den richtigen Anreiz eine erwünschte Leistung, also Ziel, erreicht werden. Der Anreiz muss zusätzlich auf die Individualität der betroffenen Person abgestimmt sein, damit der Anreiz seine Wirkung nicht verfehlt. Das Wirtschaftslexikon von Gabler definiert den Begriff des Anreizes mit einem höheren Bezug zur in Kap. 2 genannten Thematik der Motivation und im Unternehmensumfeld. „Anreize, die im Tätigkeitsvollzug selbst liegen, verbinden sich mit *intrinsischer Motivation*. Anreize, die schwerpunktmäßig im Arbeitsumfeld (Kollegen) oder in den Folgen des Tätigkeitsvollzugs liegen (monetäre Anreize), verbinden sich mit *extrinsischer Motivation.* (Peukert, et al.) Auf die monetären Anreize wird im weiteren Verlauf der Ausarbeitung detaillierter eingegangen.

„Als Bemessungsgrundlage werden Beurteilungsgrößen bezeichnet, die Auskunft über Leistung von Mitarbeitern geben“. (Huber 2014, S. 19) Diese leitet sich aus den Unternehmenszielen ab und ist ein zentraler Parameter bei der Anreizgewährung. Gleichzeitig bildet die Bemessungsgrundlage die Basis für ein erfolgreiches Anreizsystem. Sie verknüpft dabei die Mitarbeiter- und

Unternehmensperspektive. Dabei setzt sich in den letzten Jahren der Ansatz durch, dass die Bemessungsgrundlage für eine Belohnung längerfristig betrachtet wird.

„Die Belohnungsfunktion beschreibt den funktionalen Zusammenhang zwischen einer (oder mehreren) Bemessungsgrundlage(n) und der Anreizmenge“. (Huber 2014, S.21) Für das Unternehmen biete die Belohnungsfunktion die Möglichkeit zur Gestaltung und Verknüpfung von Bemessungsgrundlagen. Für das Unternehmen ist es dabei eher schwierig die richtige Unter- und Obergrenzen zu finden.

Einer weiterer Begriff bei im Umfeld von Anreizssystemen ist die Ausschüttungspolitik, dabei werden die Zeitpunkte festgelegt, zu welchen variable, leistungsbezogene Belohnung an die Mitarbeiter ausgeschüttet werden“. (Huber 2014, S.23) Der Zeitpunkt der Ausschüttung kann dabei einen großen Einfluss haben, wie dieser Anreiz auf die Leistung des Mitarbeiters hat. Bei der Gestaltung der Ausschüttungspolitik ist es wichtig, dass die Ausschüttung vom Mitarbeiter in Bezug zu seiner geleiteten Arbeit steht, sonst verfehlt dies das Ziel der Motivation.

Der letzte, wichtige Begriff, der für die Gestaltung eines Anreizsystems erforderlich ist der Adressatenkreis, denn „durch die Festlegung eines Adressatenkreises wird bestimmt, welche Personen durch ein Anreizsystem incentiviert werden“. (Huber 2014, S.24) Der Adressatenkreis spielt bei der Ausgestaltung eines Anreizsystems eine zentrale Rolle, da durch ihn bestimmt wird, welche Mitarbeiter einbezogen werden sollen. Die Auswahl erfolgt dabei aufgrund von Kostenabwägungen und Komplexitätgründen. Bei der Gestaltung kann es vorkommen, dass unterschiedliche Anreizsysteme vorhanden sind, die auf die jeweiligen Bedürfnisse einzelner Mitarbeitergruppen abgestimmt sind.

3.2 Anreizsysteme im inner- und außerbetrieblichen Umfeld

Im Zusammenhang mit Anreizsystemen kann zwischen inner- und außerbetrieblichen AS unterschieden werden. Diese werden in diesem Abschnitt erklärt und beispielhaft aufgezeigt. Die Außerbetrieblichen Anreizsysteme werden dabei nur kurz betrachtet, da diese für die weitere Umsetzung der Projektgruppe weniger relevant sind.

3.2.1 Außerbetriebliche Anreizsysteme

Bei den außerbetrieblichen Anreizsystemen handelt es sich um Programme und AS zur Kundenbindung. Dies kann in Form von Rabattprogrammen z.B. Preisnachlässe, Bonusprogramme z.B. umsatzabhängige Boni in Form von Prämien oder Mehrwertprogramm z.B. Zusatzleistungen für loyale Kunden gestaltet sein. Alle AS im außerbetrieblichen Umfeld haben die Ziele, dass der Kunde an das Unternehmen gebunden wird. (Vgl. Heinemann 2012, S. 51ff.)

Für die Umsetzung der AS werden häufig Kundenkarten eingesetzt, die die folgenden Ziele haben:

- Schärfung der Position gegenüber der Konkurrenz
- Akquisition von Neukunden
- Verbesserte Kundenkenntnisse durch den Aufbau einer Kundendatenbank
- Produktinteresse fördern
- Direkte Kundenansprache mit individuellen Bedürfnissen

Die Kundenkarten helfen somit dem Unternehmen, dass sie den Kunden langfristig an sich binden können und die Unternehmensstrategie an die Bedürfnisse der Kunden anpassen können. Neben diesen Unternehmensspezifischen Kundenkarten gibt es diese auf regionaler Ebene, dabei schließen sich verschiedene Unternehmen zusammen und schaffen somit den Anreiz die Kunden lokal zu Binden.

Dies wird meistens in Form von Bonusprogrammen umgesetzt. Einige bekannte Beispiele dafür sind die DeutschlandCard, die z.B. von Edeka, Esso, Hammer, Vergölst, Sonnenklar TV oder der Deutschen Bank akzeptiert wird und der Kunde u.a. bei seinem Einkauf Punkte sammelt. Ein weiteres Beispiel ist das Payback-Programm, bei dem die Kunden, z.B. bei ATU, DM, Lufthansa, Aral, WMF, Thalia, Real, Galeria Kaufhof, ebenfalls Punkte erhalten und diese dann eintauschen können.

Die Kundenkarten einzelner Unternehmen sind dabei sehr unterschiedlich gehalten. Dies beginnt bei der kostenlosen Mitgliedschaft (z.B. bei der Ikea-FamilyCard) oder einem jährlichen Beitrag (z.B. bei der Douglas Card). Zudem ist der Leistungsumfang der jeweiligen Kundenkarte sehr unterschiedlich. Bei der FamilyCard von Ikea beinhaltet diese beispielsweise die Teilnahme an Verlosungen, Transportversicherung für IKEA-Ware, Einladung zu Events oder exklusive Angebote. Bei der DouglasCard der Drogerie Douglas beinhaltet die Kundenkarte beispielsweise Einkaufsgutscheine, Rabattcoupons oder den Probenversand. Zugleich bietet die DouglasCard eine Kreditkartenfunktion zur bargeldlosen Zahlung mit monatlicher Abrechnung.

Neben den recht umfangreichen Kundenkarten gibt es einfachere Bonusprogramme, bei denen ein Sofortrabatt für jeden Einkauf gewehrt wird z.B. bei der PartnerCard von Hagebau in Höhe von 3% oder Prämienprogramme, wie bei Shell ClubSmart oder UCI-Kinowelt, wo mit jedem Einkauf Punkte gesammelt werden, die in speziellen Prämienshops eingetauscht werden können.

Die Vielfalt der Kundenkarten und Angebote zeigt, dass die Anreizsysteme zur Kundenbindung im außerbetrieblichen Umfeld einen sehr wichtigen Stellenwert haben.

3.2.2 Innerbetriebliche Anreizsysteme

Im innerbetrieblichen Umfeld kann bei den Anreizsystemen zwischen AS zur Gewinnung von neuen Mitarbeitern oder der AS zur Zielerreichung unterschieden werden.

Die AS zur Mitarbeitergewinnung werden hier nicht weiter betrachtet, da dies für die Projektgruppe „Impact“ weniger wichtig ist. Bei den AS zur Zielerreichung wird nach Hentze zwischen den monetären und nichtmonetären AS unterschiedenen. „Unter die monetären Anreize fallen die direkte Entlohnung, die Erfolgsbeteiligung und die betrieblichen Sozialleistungen“. (Link 2011, S. 254) Dem entgegen stehen die nichtmonetären AS, diese „umfassen die soziale Kommunikation, die Gruppenmitgliedschaft, die Führung, die Arbeitszeit- und Pausenregelung, den Arbeitsinhalt, die Arbeitsplatzgestaltung, die Personalentwicklung und die Aufstiegsmöglichkeiten sowie das betriebliche Vorschlagswesen“. (Link 2011, S. 254)

In seiner Veröffentlichung stellt Hungenberg die materiellen und immateriellen Anreize den unternehmensinternen und unternehmensexternen Anreizen gegenüber. Diese Gegenüberstellung zeigt beispielhaft, dass es bei der Gestaltung von AS viele verschiedene Möglichkeiten gibt. Diese reichen von der Kombination von materiellen Anreizen mit den unternehmensinterne Anreizen z.B. ein Firmenwagen oder eine Vergütung bis zu immateriellen Anreize, wie die flexible Arbeitszeitregelung. Hungenberg zeigt somit, dass es eine Vielzahl von Möglichkeiten zur Gestaltung der AS gibt.

	Materielle Anreize	Immaterielle Anreize
Unternehmensinterene Anreize	• Vergütung • Firmenwagen • Versicherungsleistungen	• Arbeitszeitregelungen • Aufgabenfeld- und entwicklung • Sozialer Status
Unternehmensexterne Anreize	• Vermögensberatung • Steuersysteme • Systeme der sozialen Sicherung	• Weiterbildung • Kulturangebot • Soziales Umfeld

Tabelle 1: Anreizarten nach Hungenberg

3.3 Anforderungen an ein Anreizsystem

Bei der Gestaltung eines AS im Unternehmen müssen die Punkte Individualität, Langfristigkeit, Qualifikationsorientierung, Flexibilität, Leistungsorientierung, Transparenz und Wirtschaftlich-

keit (Vgl. Boenigk 2011, S. 140) beachtet werden. Die zu beachtenden Anforderungen werden in diesem Abschnitt genauer betrachtet.

Die wichtigste Anforderung für die erfolgreiche Umsetzung eines AS besteht in der Individualität, dabei werden die Anreize so gestaltet, dass die spezifischen Leistungsmotive der Mitarbeiter angesprochen werden. Ohne diese Individualität des AS werden die Ziele zur Leistungserreichung durch das AS verfehlt, da diese sich nicht angesprochen fühlen. Die Umsetzung für jeden einzelnen Mitarbeiter ist nahezu unmöglich, daher sollte das AS zumindest für eine Mitarbeitergruppe z.B. in der Produktion oder Logistik bzw. Altersstufe abgestimmt werden, auch wenn dadurch mehrere AS in einem Unternehmen entstehen.

Die zweite Anforderung besteht in der Langfristigkeit. Bei der Einführung ist das AS nicht perfekt, sodass eine schrittweise Anpassung an die Motivstrukturen erfolgen muss. Auch wird das AS nicht von Beginn an funktionieren, auch wenn alles dafür getan wird, damit das AS akzeptiert wird. Ein AS sorgt eher für einen langfristigen Erfolg zur Erreichung der Unternehmensziele.

Eine weitere Anforderung an das AS stellt die Qualifikationsorientierung dar. Dabei muss sichergestellt werden, dass auch jeder Mitarbeiter die Möglichkeit bekommt, dass dieser sich am AS beteiligen kann. Dazu muss das Integrationsbewusstsein der Mitarbeiter schrittweise ausgebaut werden.

Die vierte, wichtige Anforderung ist die Flexibilität. Das AS ist darf kein starres System sein, was sich nicht an die veränderten Bedingungen anpassen kann. Dies beginnt schon dabei, dass die Anreize sich in Krisensituationen an die wirtschaftliche Lage des Unternehmens anpassen können. Auch verändern sich die Motive der Mitarbeiter, an die sich das AS flexibel anpassen muss.

Die nächste Anforderung an das AS ist die Leistungsorientierung. Die Anreize müssen dabei so gestaltet sein, dass diese auch von den Mitarbeitern erreicht werden können. Zudem muss ein Zusammenhang zwischen Leistungsverhalten und Anreiznutzen geschaffen werden.

Für die Akzeptanz und zur Verbesserung des Betriebsklimas besteht die Anforderung der Transparenz an das AS. Die Transparenz sorgt dafür, dass die Mitarbeiter einen Zusammenhang zwischen der Leistung und dem jeweiligen Anreiz gezogen werden kann. Dazu ist ein regelmäßiges Feedback erforderlich. Auch für die Kollegen wird dadurch sichtbar, für welche Leistung der Kollege den Anreiz bekommen hat.

Aus Sicht des Unternehmens steht die Anforderung, dass das AS wirtschaftlich sein muss. Die Anreize müssen so gestaltet werden, dass die Kosten dafür getragen werden können und das Unternehmen nicht Gefahr läuft Insolvent zu werden. Zudem müssen die Kosten für die administrativen Aufgaben des AS überschaubar sein und z.B. nicht Mitarbeiter nur mit der Betreuung des AS beschäftigt sind.

3.4 Anreizsysteme in der Praxis

Auch in der Praxis sind die AS verbreitet. Dabei treten verschiedene Konzepte auf. Meistens sind diese im Zusammenhang mit Bonuszahlungen. Diese haben jedoch meisten jedoch nur eine Auswirkung von einem Monat auf die geleistete Arbeit. (Vgl. Friesike, S., et. al)

Meistens habe andere Anreize eine langfristigere Auswirkung auf die Leistung und Motivation der Mitarbeiter. Dies sind ein paar Beispiele:

Bei IBM gibt es ein Programm namens „Research Fellows“, dabei wird den Mitarbeitern die Möglichkeit gegeben, dass diese ihre Passion in der Forschung für sechs Monate ausleben kann und sich somit selbstverwirklichen. Das Chemieunternehmen Ciba ha eine Ideenbörse, dabei werden die Ideen durch das Team beurteilt und die Mitarbeiter bekommen freie Zeit für die Verwirklichung ihrer Idee. Bei Google gibt es den Spielplatz, dabei gibt es ein großes Angebot, sodass die Grenze zwischen Arbeit und Feierabend fließend werden. Die Volkswagen AG hat Mitarbeiter aus der Entwicklung für ein halbes Jahr in die USA geschickt, sodass ein direkter Bezug zwischen der Belohnung und Arbeit gezogen werden. Diese Reise war ein Gesprächsthema über Jahre, sodass die Belohnung eine sehr lange Wirkung hatte.

Dies zeigt, dass es in der Praxis viele Ansätze gibt, die ihre Wirkung erfüllen. Oft kommen auch Punktesysteme zum Einsatz, die in Prämienshops eingetauscht werden können.

3.5 Zusammenhang und Bedeutung für die Projektgruppe

Für die Umsetzung in der Projektgruppe besitzen beide Thematiken eine hohe Relevanz. Die Mitarbeiter müssen motiviert werden, sodass sie bei der Umsetzung des Innovationsmanagement mitwirken und so z.B. Fragebögen beantworten, am Test mitwirken und am Entwicklungsprozess beteiligt werden. Dies ist gleichzeitig eine Möglichkeit, dass die Mitarbeiter motiviert werden, dass diese das fertige System akzeptieren und damit gerne mit dem System arbeiten. Die Folge sind gute Ideen und Vorschläge, die zu innovativen Lösungen führen. Für die aktive Beteiligung an Ideen und der Umsetzung helfen die AS zur Motivation.

Für den Erfolg der Umsetzung wird es somit erforderlich sein, dass ein transparentes, wirtschaftliches, flexibles und individuelles AS gestaltet wird. Zwar kann die endgültige Form des AS zu Beginn der Umsetzung mit geringerer Priorität betrachtet werden. Je weiter das Projekt fortgeschritten ist, desto wichtiger wird der Einsatz eines AS. Die Anreizsysteme helfen somit bei der Erreichung der Ziele innerhalb der Projektgruppe.

4 Fazit und Vorschläge

Der Zusammenhang zwischen beiden Thematiken stellte sich im Laufe der Bearbeitung des Seminarthemas als enger heraus, als zunächst erwartet. Die Motivation mit deren individuellen Motiven und Bedürfnissen aus den Motivationstheorien bilden die Basis für die Gestaltung eines Anreizsystems. Bei der Gestaltung des AS müssen die individuellen Bedürfnisse der Mitarbeiter berücksichtigt werden.

Für die Projektgruppe sollte ein Anreizsystem gestaltet werden, um die Mitarbeiter zu Motivieren, das diese Vorschläge und Ideen einbringen und sich bei der späteren Umsetzung dieser beteiligen. Der Vorschlag wäre hierbei ein Bonussystem, bei dem die Mitarbeiter für bestimmte Aktionen (z.B. Abgabe von Vorschlägen, Verteilung von Bewertungspunkten etc.) Punkte bekommen, die in einem Prämienshop gegen Produkte z.B. Gutscheine, Artikel, Freizeitaktivitäten oder Geldprämien oder Freizeit eingetauscht werden können. Dies kann auch in der Kombination mit Elementen der Gamification verbunden werden. Die genaue Gestaltung der Prämien muss in Abstimmung an die individuellen Bedürfnisse der Mitarbeiter erfolgen. Zusätzlich muss es auch möglich sein, dass die Vorgesetzten individuelle Punkte verteilen können, die z.B. von der eingebrachten Zeit bei der Umsetzung einer Idee berücksichtigt. Die Prämien sollten dabei in verschiedene Bereiche eingeteilt werden, sodass der Mitarbeiter eine Wahlmöglichkeit hat.

Um die Akzeptanz, Gestaltung und Annahme des AS zu motivieren, sollten die Mitarbeiter frühzeitig in den Entwicklungsprozess eingebunden werden. Dies unterstützt dabei, dass die Mitarbeiter im Einsatz das Innovationsmanagementsystem aktiv nutzen und sich beteiligen. Von der Geschäftsleitung sollten zur Motivation weiterhin festgelegt werden, dass das System jederzeit genutzt werden kann. Somit wird dem Mitarbeiter das Gefühl gegeben, dass sich die die Nutzung nicht auf die Arbeitszeit auswirkt.

Ein funktionierendes Anreizsystem und die passende Motivation ist somit ein wichtiger Faktor für die erfolgreiche Umsetzung der Projektgruppe „Impact“. Die genaue Ausgestaltung kann jedoch erst im Laufe der Entwicklung und bei einem feststehenden Prozess gemacht werden.

Literaturverzeichnis

Boenigk, M. (2001): Umsetzung der Integrierten Kommunikation (1. Auflage). Wiesbaden: Gabler Verlag

Esch, F., Maier, G., Kirchgeorg, M.: Springer Gabler Verlag (Herausgeber), Gabler Wirtschaftslexikon, Stichwort: Motiv, online im Internet: http://wirtschaftslexikon.gabler.de/Archiv/57497/motiv-v7.html (aufgerufen am 23.4.2015)

Friesike, S., Gassmann, O.: http://die-erfinder.3mdeutschland.de/innovationskultur/der-gummibarchen-effekt-monetare-anreize-sind-fuer-mitarbeiter-nicht-alles (aufgerufen am 8.5.2015)

Heinemann, G. (2012): Der neue Online-Handel (4. Auflage).Wiesbaden: Springer Gabler

Huber, H. (2014): Nachhaltigkeitsorientierte Anreizsysteme. Lohmar: Josef EUL Verlag GmbH

Link. J. (2011): Führungssysteme (6. Auflage). München: Verlag Franz Vahlen GmbH

Maier, G., Kirchgeorg, M.: Springer Gabler Verlag (Herausgeber), Gabler Wirtschaftslexikon, Stichwort: Motivation, online im Internet: http://wirtschaftslexikon.gabler.de/Archiv/55007/motivation-v6.html (aufgerufen am 22.4.2015)

Maier, G.: Springer Gabler Verlag (Herausgeber), Gabler Wirtschaftslexikon, Stichwort: intrinsische Motivation, online im Internet: http://wirtschaftslexikon.gabler.de/Archiv/57320/intrinsische-motivation-v6.html (aufgerufen am 10.5.2015)

Maier, G.: Springer Gabler Verlag (Herausgeber), Gabler Wirtschaftslexikon, Stichwort: extrinsische Motivation, online im Internet: http://wirtschaftslexikon.gabler.de/Archiv/57321/extrinsische-motivation-v5.html (aufgerufen am 10.5.2015)

Maier, G.: Springer Gabler Verlag (Herausgeber), Gabler Wirtschaftslexikon, Stichwort: intrinsische Motivation, online im Internet: http://wirtschaftslexikon.gabler.de/Archiv/57320/intrinsische-motivation-v6.html (ausgerufen am 10.5.2015)

Maier, G.: Springer Gabler Verlag (Herausgeber), Gabler Wirtschaftslexikon, Stichwort: Motivatoren, online im Internet: http://wirtschaftslexikon.gabler.de/Archiv/77706/motivatoren-v4.html (aufgerufen am 10.5.2015)

Peukert, H., Maier, G., Eggert, W., Minter, S.: Springer Gabler Verlag (Herausgeber), Gabler Wirtschaftslexikon, Stichwort: Anreiz, online im Internet: http://wirtschaftslexikon.gabler.de/Archiv/3627/anreiz-v13.html (aufgerufen am 12.5.2015)

Piekenbrock, D., Maier, G., Kirchgeorg, M.: Springer Gabler Verlag (Herausgeber), Gabler Wirtschaftslexikon, Stichwort: Bedürfnis, online im Internet: http://wirtschaftslexikon.gabler.de/Archiv/57500/beduerfnis-v7.html (aufgerufen am 22.4.2015)

Wickert, H., Hardes, H.-D. (2004): Praxisbeispiele zur Erfolgs- und Kapitalbeteiligung der Mitarbeiter. München: Rainer Hampp Verlag

Wiedmann, S. (2006): Erfolgsfaktoren der Mitarbeiterführung (1. Auflage).Wiesbaden: Deutscher Universitätsverlag

B.1.8. Oberflächendesign UI und Usability

Thema:

User Interface und Usability

Seminararbeit
im Rahmen der Projektgruppe IMPACT

Abteilung Wirtschaftsinformatik 1:
Very Large Business Applications

Themensteller:	Prof. Dr.-Ing. Jorge Marx Gómez
Betreuer:	Dr. Joachim Kurzhöfer Dipl.-MA. Jens Siewert Stefan Wunderlich, M.Sc

vorgelegt von:	Artjom Baranow Elly-Heuss-Knapp Str. 12a 27793 Wildeshausen Tel: 0172 5253255 E-Mail: artjom.baranow@uni-oldenburg.de

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Zu Beginn dieser Seminararbeit erfolgt eine kurze Einführung in das Thema, indem der Hintergrund der Thematik motiviert und näher auf die Problemstellung eingegangen wird. Zum Abschluss der Aufbau der Arbeit, um einen besseren Überblick der Inhalte zu verschaffen.

1.1 Motivation und Problemstellung

Heutzutage ist das Angebot an Software und Webseiten vermeintlich grenzenlos, nichtsdestotrotz wird es täglich erweitert. Im Hinblick darauf stellt sich die zentrale Frage, wer überhaupt Aussagen darüber trifft, in welchem Maße die Qualität und Benutzbarkeit einer Software sowie einer Webseite ausfallen. Die Antwort darauf ist *Usability*.

Das Überprüfung der Usability eines Systems ist auf der einen Seite zwar zunächst mit hohen Kosten verbunden, auf der anderen Seite fallen diese jedoch äußerst gering aus, wenn diese mit den Kosten verglichen werden, die beispielsweise für nachträgliche Systemanpassungen hervorgerufen werden. (Vgl. Nielsen, 1993)

Eine positive Usability ruft beim Nutzer eine höhere Akzeptanz des Systems hervor. Entstehen Unannehmlichkeiten in Form von Problemen hinsichtlich der Bedienung oder ähnlichem, kann es dazu führen, dass bei dem Anwender Frustrationsmomente auftreten. Treten diese Probleme in häufiger Form auf, kann dies dazu beitragen, den Grad der Akzeptanz bei dem Benutzer so voranzutreiben, dass dieser sich dazu entschließt, das bestehende System durch ein neues zu ersetzen. Demnach stellt die Usability einen essentiellen Schritt dar, der sich in direkter Form auf die Anzahl der Benutzer und somit auf die zahlende Kundschaft auswirkt. (Vgl. Nielsen, 1993)

Aus diesem Grund reicht es nicht aus, die gewünschten Informationen auf einer Webseite oder zu repräsentieren. Es wird fortlaufend bedeutsamer, diese Informationen so vorzulegen, dass sie auf eine schnelle, einfache und zufriedenstellende Art und Weise erreicht werden können. Dieser Aspekt nimmt demnach eine wichtige Rolle ein, um die Erfolgschancen zu optimieren und gewinnbringende Maßnahmen voranzutreiben. (Vgl. Nielsen, et al., 2006)

1.2 Aufbau der Arbeit

Die Arbeit lässt sich in fünf Teile gliedern. Kapitel eins umfasst die Einleitung, dazu gehören die Motivation und die Problemstellung der Arbeit. Nach dieser Darstellung folgen in Kapitel zwei und drei zum besseren Verständnis der Arbeit die Grundlagen. Hier werden wesentliche Begriffsdefinitionen erläutert, wie die Grundbegriffe User Interface und Usability. Im nächsten Schritt wird näher auf die Merkmale beider Begriffe eingegangen. Zudem werden zu jedem Begriff Kriterien aufgezeigt, die herangezogen werden können, um ergonomische Benutzerschnittstellen sowie gute Usability zu erreichen

In Kapitel vier wird eine Methode und Vorgehensweise erläutert, wie Usability in den Softwareentwicklungsprozess integriert werden kann. Hierbei wird der Begriff Usability Engineering näher erklärt, der dazu beiträgt, Komponenten des Software Engineerings mit den Methoden der Usability zu verbinden und dadurch ein benutzerorientierter Entwicklungsprozess entsteht. Im Anschluss daran werden diesbezüglich wesentliche Punkte beleuchtet und Handlungsempfehlungen aufgezeigt, die für die Projektgruppe „Innovation Management Platform to Activate Creative Thoughts“ als relevant erachtet werden können.

Den Abschluss der Arbeit bildet Kapitel fünf. Dort wird eine Schlussbetrachtung aufgezeigt, welches die Ergebnisse zusammenfasst sowie ein Fazit zieht.

2 User Interface

In diesem Kapitel der Seminararbeit erfolgen zu Anfang Begriffsdefinitionen, die mit der Thematik des User Interfaces und der Problemstellung einhergehen. Zunächst wird auf den Begriff des User Interface näher eingegangen, um dadurch ein besseres Verständnis zu gewährleisten.

2.1 Begriffsbestimmung

Der Begriff User Interface wird im Deutschen als eine Benutzerschnittstelle aufgefasst. Diese kann aus zwei unterschiedlichen Blickwinkeln betrachtet werden. Auf der einen Seite ist die Menge der Interaktionsmöglichkeiten gemeint sowie die dazugehörigen Hilfskomponenten. Dazu gehören zum Beispiel Softwarekomponenten im Bereich der Benutzeroberfläche. Auf der anderen Seite ist die komplette Hard- und Software gemeint, die zur Computerbedienung herangezogen wird. Dazu kann unter anderem die Peripherie gehören, wodurch dem Anwender zum Beispiel die Möglichkeit gegeben wird, einen Computer bedienen zu können. (Vgl. Herczeg, 2005)

Hix und *Hartson* definieren zudem, dass eine Verständlichkeit zwischen dem Computer auf der einen Seite und dem Menschen auf der anderen Seite geschafft werden muss. Der Nachrichtenaustausch zwischen diesen Parteien erfolgt dabei auf unterschiedlichen Zeichensystemen. Diese müssen auf einen gemeinsamen Code abgebildet werden, um eine erfolgreiche Informationsübertragung zu gewährleisten. Benutzerschnittstellen repräsentieren hierbei aus Anwendersicht sämtliche Aktionen und Zustände (zum Beispiel Eingaben), um diese im nächsten Schritt in eine computerverständliche Repräsentation zu übersetzen, damit die Maschine die Möglichkeit erlangt, diese Kommunikation zu verarbeiten. Im umgekehrten Schritt übersetzen Benutzerschnittstellen aus Computersicht ebenfalls die einzelnen Aktionen und Zustände (Ausgaben), damit der Mensch den Kommunikationsaustausch in verständlicher Art und Weise repräsentiert bekommt, sodass Reaktionen stattfinden können. (Vgl. Hix, et al., 1993)

2.2 Arten von User Interfaces

In der Vergangenheit der Computerentwicklung waren die eingesetzten Rechner nicht effizient genug, um beispielsweise graphische Benutzeroberflächen einzusetzen. Zu Beginn existierten lediglich die Command Line Interfaces, die sich weiter zu Text User Interfaces und anderen Benutzerschnittstellen entwickelt haben. In den nachfolgen Abschnitten werden typische Arten

aufgezeigt, um einen besseren Überblick der Thematik zu verschaffen. Unterschieden wird unter anderem zwischen folgenden Benutzerschnittstellen: (Vgl. OnPage, 2015)

- **Command Line Interface:** Hierbei gibt der Benutzer die Befehle per Tastatur in eine Kommandozeile ein. Ein Beispiel hierfür sind Kommandozeileninterpreter.

- **Text User Interface:** Diese Benutzerschnittstelle ist zeichenorientiert. Die Ausführung erfolgt dabei im Textmodus und die Navigation in der Regel über die Tastatur. Einsatzgebiete sind zum Beispiel Bootloader[1] oder BIOS-Setup-Programme.

- **Graphical User Interface:** Das Graphical User Interface, auch GUI genannt, ist eine Oberfläche, die bei gängiger Software genutzt und in der Regel mit der Maus bedient wird.

- **Voice User Interface:** Dabei handelt es sich um sprachbasierte Benutzerschnittstellen. Der Nutzer kommuniziert dabei durch Sprachein- und Sprachausgaben mit der Maschine.

- **Natural User Interface:** Zu den natürlichen Benutzerschnittstellen gehören beispielsweise die klassischen Touchscreens, die auf den heutigen Smartphones oder Tablets vertreten sind. Die Interaktion erfolgt dabei in natürlicher und intuitiver Art, in der Regel über Finger- und Handbewegungen.

2.3 Kriterien für das Design ergonomischer Benutzerschnittstellen

Damit Benutzerschnittstellen interaktiver Systeme, so wie das in Kapitel 2.1 beschrieben worden ist (Mensch-Maschine-Interaktion), sowohl effizient als auch effektiv bedient werden können, ist es von essentieller Bedeutung, diesbezüglich feste Kriterien zu definieren. Dieses Kapitel dient hierbei lediglich der Heranführung und Erläuterung dieser Kriterien. Entsprechende Verknüpfung in Bezug Nutzerzufriedenheit, Effektivität und Effizienz werden in Kapitel 3.1 näher erläutert.

Ein Grundsatz, der allgemein anerkannt wird, ist der Abschnitt 110 der DIN EN ISO 9241 – 10 Norm. Um Vergleichbarkeit zu gewährleisten, werden in diesem Kapitel diese Grundsätze mit den Grundsätzen des Autors *Baggen* verglichen, die er in seinem Buch *Gestaltungskriterien für eine benutzerorientierte Software* definiert hat. *Baggen* beschreibt hierbei folgende Grundsätze

[1] Bootloader wird im Volksmund auch unter einem Startprogramm aufgefasst. Dabei handelt es sich um eine Software, die gewöhnlich durch die Firmware (zum Beispiel das BIOS) eines Computers geladen und durchgeführt wird. (Vgl. archlinux, 2015)

zur Gestaltung und Bewertung von Schnittstellen zwischen Benutzer und System (Dialoggestaltung).

Aufgabenangemessenheit

„Ein Dialog ist aufgabenangemessen, wenn er den Benutzer unterstützt, seine Arbeitsaufgabe effektiv und effizient zu erledigen“. (ISO, 1998)

Nach *Baggen* soll ein Dialogsystem alle Funktionalitäten beinhalten und dafür sorgen, dass der Benutzer entlastet wird. Der Benutzer soll in seinem Arbeitsablauf unterstützt und lediglich die Informationen erhalten, die im Nutzungskontext als relevant erachtet werden. (Vgl. Baggen, 2003)

Selbstbeschreibungsfähigkeit

„Ein Dialog ist selbstbeschreibungsfähig, wenn jeder einzelne Dialogschritt durch Rückmeldung des Dialogsystems unmittelbar verständlich ist oder dem Benutzer auf Anfrage erklärt wird“. (ISO, 1998)

Das System soll dafür sorgen, dass der Benutzer so exakt wie möglich durch die einzelnen Arbeitsschritte geleitet wird, um Kommunikationsprobleme zu verhindern. (Vgl. Baggen, 2003)

Erwartungskonformität

„Ein Dialog ist erwartungskonform, wenn er konsistent ist und den Merkmalen des Benutzers entspricht, zum Beispiel seinen Kenntnissen aus dem Arbeitsgebiet, seiner Ausbildung und seiner Erfahrung sowie den allgemein anerkannten Konventionen“. (ISO, 1998)

Baggen trifft hierbei die Aussage darüber, dass die Informationsdarstellung innerhalb eines Systems in einheitlicher Form visualisiert werden muss, sodass sich der Cursor beispielsweise an der Stelle befindet, wo die Eingabe erwartet wird. (Vgl. Baggen, 2003)

Fehlertoleranz

„Ein Dialog ist fehlertolerant, wenn das beabsichtigte Arbeitsergebnis trotz erkennbar fehlerhafter Eingaben, entweder mit keinem oder mit minimalem Korrekturaufwand seitens des Benutzers erreicht werden kann“. (ISO, 1998)

Hierbei veranschaulicht der Autor, dass das System dazu beitragen soll, dem Benutzer auf seine Fehler hinzuweisen und bei der Fehlervermeidung Unterstützung zu leisten. Vgl. Baggen, 2003)

Individualisierbarkeit

„Ein Dialog ist individualisierbar, wenn das Dialogsystem Anpassungen an die Erfordernisse der Arbeitsaufgabe sowie an die individuellen Fähigkeiten und Vorlieben des Benutzers zulässt“. (ISO, 1998)

Dieses Kriterium soll dazu beitragen, Anpassungen anhand der Fähigkeiten des Benutzers vornehmen zu können, zum Beispiel durch die Auswahl der Sprache. Vgl. Baggen, 2003)

Lernförderlichkeit

„Ein Dialog ist lernförderlich, wenn er den Benutzer beim Erlernen des Dialogsystems unterstützt und anleitet“. (ISO, 1998)

Benutzer sollen bei dem Erlernen des Dialogsystems unterstützt werden, zum Beispiel in Form von Tutorials, Hilfe bei der Navigation oder einen Überblick der benötigen Funktionen erhalten. (Vgl. Baggen, 2003)

Steuerbarkeit

„*Ein Dialog ist steuerbar, wenn der Benutzer in der Lage ist, den Dialogablauf zu starten sowie seine Richtung und Geschwindigkeit zu beeinflussen, bis das Ziel erreicht ist“.* (ISO, 1998)

In diesem Zusammenhang betont der Autor explizit, dass eine Unterbrechung der Arbeit ohne Datenverlust möglich sein muss, sodass der Nutzer die für ihn beste Konfiguration bestimmten kann.

Die Vergleichbarkeit beider Grundsätze zeigt auf, dass sich die Aussagen von *Baggen* mit denen der anerkannten internationalen Norm widerspiegeln. Vereinzelt kann es zusätzlich dazu kommen, dass die sieben genannten Grundsätze nicht in jeder Situation streng voneinander getrennt werden und sich je nach Anwendungsfall, überschneiden können. Der Vergleich dieser beiden Grundsätze soll in erster Linie verdeutlichen, dass Normen und Richtlinien nicht als Vorschrift gesehen werden, sondern als Anregung dienen sollen, um seine Usability-Ziele effizienter und effektiver zu erreichen.

3 Usability

Im folgenden Kapitel werden spezielle Begriffe, die für das Thema Usability von zentraler Bedeutung sind, erläutert. Im Zuge dessen werden des Weiteren wesentliche Bereiche im Kontext des Usability-Begriffs näher aufgezeigt.

3.1 Begriffsbestimmung

Bei dem Begriff Usability handelt es sich um einen sehr vielfältigen Begriff. Bevor dieser hergeleitet wird, folgen zu Beginn typische Definitionen.

„*Usability applies to all aspects of a system with human might interact, including installation and mightness procedures.“* (Nielsen, 1993)

„Usability ist the measure of the quality oft he User experience when interacting with something – wether a Web Site, a traditional software application, or any other device the user can operate in some way or antoher.“ (Nielsen, 1993)

„Der Begriff Usability stammt aus dem Englischen. Er setzt sich aus zwei Worten zusammen, to use (benutzen) und the ability (die Fähigkeit). Übersetzt wird der Begriff mit Gebrauchstauglichkeit oder aber auch Brauchbarkeit.“ (Böhringer, et al., 2011)

Diese drei Definitionen sollen näher verdeutlichen, wie vielschichtig dieser Begriff aufgefasst wird. Ungeachtet dessen, aus welchen Bestandteilen sich der Usability-Begriff zusammensetzt, existiert keine allgemeingültige Definition, die den Begriff eindeutig beschreibt. Einen Ansatz liefert jedoch die DIN EN ISO 9241 – 11 Norm.

Diese Norm drückt sagt aus, dass die *„Usability eines Produktes das Ausmaß ist, in dem es von einem bestimmten Benutzer verwendet werden kann, um bestimmte Ziele in einem bestimmten Kontext effektiv, effizient und zufriedenstellend zu erreichen.“* (ISO2, 1998)

Folgende Grafik soll diesen Kontext detaillierter aufzeigen:

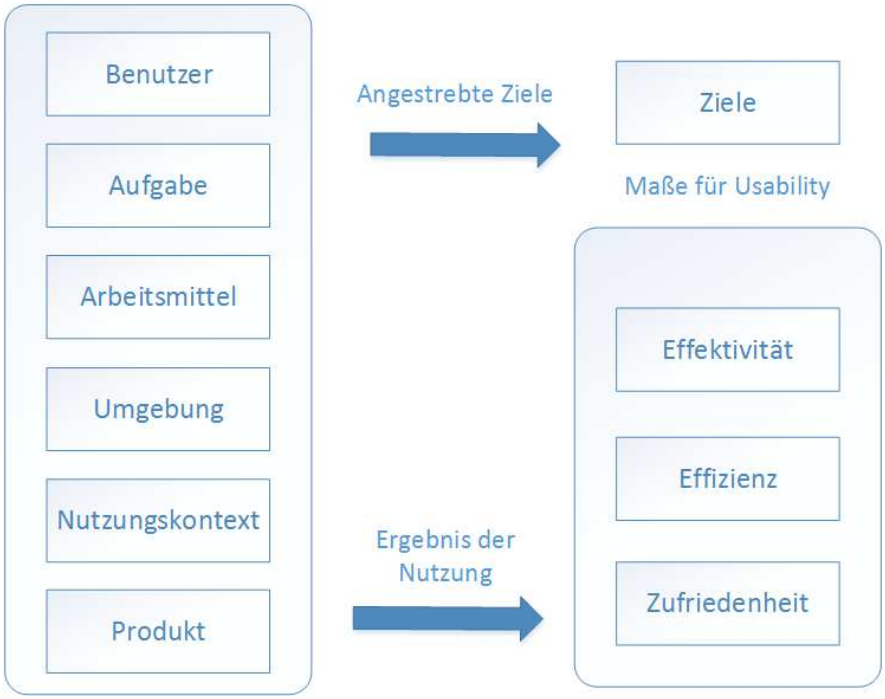


Abbildung 3.1: Usability nach DIN EN ISO 9241 - 11

Quelle: Eigene Darstellung in Anlehnung an: Sarodnick und Brau, 2006

Die Grafik beschreibt demnach, dass die Usability eines Produktes den Maßstab dafür definiert, wie der Nutzer in qualitativer und quantitativer Form mit einem Produkt zurechtkommt. Anhand dieses Maßstabes wird definiert, wie hoch der Grad ausfällt, in dem die vom Benutzer festgelegten Ziele in einem bestimmten Kontext effektiv, effizient und zufriedenstellend zu Stande gebracht werden können. (Vgl. Hartmann, 2008)

Die *Effektivität* beschreibt die Genauigkeit und Vollständigkeit, um ein bestimmtes Ziel zu erreichen. Unter der *Effizienz* wird das Verhältnis zwischen der Effektivität im Vergleich zum Aufwand verstanden, um ein Ziel zu erreichen. Die *Zufriedenheit* bei der Benutzung eines Produktes kann beispielsweise auf einer Skala aufgenommen werden. Im *Nutzungskontext* wird der Anwender mit seinen den jeweiligen persönlichen Merkmalen beschrieben, dazu gehören zum Beispiel Alter, Beruf und Bildung. Des Weiteren wird dieser Kontext dazu verwendet, um auf die verfügbaren Arbeitsmittel als auch die Umgebung des Nutzers, sowohl in physischer als auch sozialer Sicht, einzugehen. (Vgl. ISO, 1998; Nielsen, 93)

3.2 Usability Modell nach Nielsen

Anhand der Abbildung 3.1 sowie der dazugehörigen Kapitelerläuterungen wurde aufgezeigt, dass der Begriff *Usability* mehrdimensional aufgefasst werden kann und somit eine Eigenschaft von Benutzerschnittstellen abbildet. Folgende fünf Komponenten nach Nielsen sollen diese Eigenschaft zusätzlich hervorheben und Komponenten aufzeigen, die den Usability-Begriff charakterisieren. (Vgl. Nielsen, 1993)

Erlernbarkeit

Die Erlernbarkeit gibt an, wie leicht es einem Benutzer fällt, eine einfache Aufgabe durchzuführen, wenn zum ersten Mal mit dem System gearbeitet wird.

Effizienz

Diese bestimmt die Schnelligkeit der Benutzer bei der Durchführung von Aufgaben, wenn bereits Erfahrungen mit dem System gesammelt worden sind.

Erinnerbarkeit

Die Erinnerbarkeit zeigt auf, wie gut ein Benutzer bereits erlangte Kenntnisse ausüben kann, nachdem das System über einen längeren Zeitraum nicht genutzt wurde.

Fehler

Treffen Aussagen über die Anzahl sowie den Schweregrad der Fehler. Des Weiteren wird angegeben, wie gut sich der Benutzer von dem entstandenen Fehler erholen kann.

Zufriedenstellung

Diese soll dazu beitragen, eine angenehme Benutzung des Systems zu gewährleisten.

Usability stellt nach *Nielsen* somit ein mehrdimensionales Qualitätsmerkmal dar, welches besagt, inwieweit ein System Fehleranfälligkeit aufweist, ob ein Benutzer mit einem Produkt arbeiten kann, wie schnell die dazugehörige Benutzung erlernbar ist, wie leicht sich der Benutzer bestimmte Vorgänge merken kann und wie viel Aufwand letztendlich erbracht werden muss, um das Ziel zufriedenstellend zu erreichen. Nielsen prägte diesbezüglich folgendes Zitat:

„Wenn die Nutzer einen Gegenstand weder nutzen möchten noch können, bräuchte er gar nicht existieren“. (Nielsen, et al., 2006)

3.3 Usability im World Wide Web

Nach den allgemeinen Erklärungen in Kapitel 2.1 und Kapitel 3.1 wird nun der Begriff Usability im Kontext der Nutzung im World Wide Web näher erläutert. Web-Usability beschäftigt sich im Vergleich zu der in Kapitel 3.1 aufgezeigten Definition darüber hinaus mit dem Produkt als Webseite mit den dazugehörigen Dimensionen Inhalt (Content-Design), Gestaltung (Page-Design) sowie Struktur (Site-Design). Der Einsatz von Usability im Kontext des Internets verlangt demnach eine erweiterte Betrachtung, da zusätzliche Faktoren berücksichtigt werden müssen. (Schweibenz, et al., 2003)

In dem darauffolgenden Kapitel wird zunächst die hohe Bedeutsamkeit der Usability im World Wide Web aufgezeigt, indem die Gründe für dessen Einsatz näher beleuchtet werden.

3.3.1 Gründe für Web Usability

Laut *Schweibenz* und *Thissen* wird folgenden drei Punkten besondere Wichtigkeit zugesprochen, weshalb Usability im Internet gebraucht wird. (Vgl. Schweibenz, et al., 2003)

1. **Wachstum der Internetnutzung**: Das Wachstum der Internetnutzer steigt in einem stetigen Verlauf. Dies hat zur Folge, dass im Zuge dessen die Anzahl von Internet-Angeboten dementsprechend steigen wird. Internetseiten, die im Verhältnis zu anderen Seiten eine positivere Bedienbarkeit aufweisen, werden in diesem Fall mit höherer Wahrscheinlichkeit höheren Erfolg verzeichnen können.

2. **Netzgeschwindigkeit**: Der zweite Grund kann der Netzgeschwindigkeit beigemessen werden. Eine standardisierte Anwendung nutzt die kompletten Ressourcen eines Computers aus und die Kommunikationen zwischen den einzelnen Komponenten erfolgt in der Regel sehr effizient. Der Nutzer hat diesbezüglich konforme Erwartungen im Bereich der Internetnutzung. Dieser möchte dabei sein Ziel und Vorhaben in einer angemessen Zeit erreichen. Somit ist es im Sinne einer effizienten Web-Usability nicht akzeptabel, wenn im Internet die Kommunikation zwischen den Komponenten eine wesentlich langsamere Geschwindigkeit aufweist.

3. **Nutzergruppen**: Einen weiteren Grund bilden die unterschiedlichen Nutzergruppen im Bereich des Internets ab. Im Bereich des World Wide Webs können im Gegensatz zu einer standardisierten Software die Nutzergruppen beispielsweise variieren und unterschiedlich ausfallen. Hierbei muss darauf geachtet werden, dass im Internet mit verschiedenen Nutzergruppen gerechnet werden muss. Für einen erfahrenen Benutzer bietet es sich zum Beispiel an, eine Suchfunktion einzubauen,

während es für einen Anfänger essentieller ist, eine gut und verständlich strukturierte Navigation zu bedienen.

Laut Nielsen ist es besonders wesentlich, folgende Aspekte genauer zu betrachten: (Vgl. Nielsen, et al., 2006)

1. **Vielfalt des Webangebots**: In der Mitte der neunziger Jahre existierten weniger als 10 Millionen Webseiten weltweit. Heutzutage hat sich die Situation um das zehnfache gesteigert. Mit der stetigen Anzahl der Internetseiten entwickelte sich parallel das Nutzerverhalten gegenüber diesen. Der Benutzer kann in diesem Fall aufgrund des umfassenden Angebots die Webseite wechseln, da es immer eine Alternative gibt, mit der das gewünschte Ziel erreicht werden kann.

2. **Internet als Allzweckwerkzeug**: Aufgrund der im ersten Punkt aufgezeigten Vielfalt des Webangebots kann das Internet in der heutigen Zeit für die verschiedensten Bedürfnisse eingesetzt werden, da heutzutage so gut wie alles online möglich zu sein scheint. Somit machen sich die Benutzer das Internet zu einem Werkzeug, um eine Vielzahl an Wünschen auf einfache Art und Weise zu befriedigen.

3. **Webseiten sind kostenlos**: Ein Webauftritt kann in diesem Fall mit dem kostenlosen Eintreten in ein lokales Geschäft verglichen werden. Wenn ein Unternehmen sich beispielsweise dazu entschließt, dem Kunden etwas zum Verkauf anzubieten, dann muss dafür Sorge getragen werden, diese Webseite und den gesamten Abwicklungsprozess leicht bedienbar zu gestalten.

4. **Vertrauen und Wohlwollen**: Nutzer betreten eine Webseite nicht mit einem unendlichen Maß an positiver Einstellung. Diese Einstellung kann sowohl benutzerabhängig als auch stimmungsabhängig beeinflusst werden. Treten zusätzliche Probleme im Bereich der Usability auf, so kann das Wohlwollen erheblich gesenkt werden, bis der schlimmste Fall eintritt und ein Nutzer in einem verärgerten Zustand die Webseite verlässt. Im Gegensatz dazu existieren eine Vielzahl vertrauensbildender Maßnahmen, wodurch der Nutzer dazu animiert wird, die Webseite in wiederholter Form zu besuchen.

3.3.2 Richtlinien für Web Usability

Zu Beginn des Kapitels 3 wurde bereits deutlich, dass es sich bei Usability um einen abstrakten Begriff handelt. Um Usability bewerten zu können, ist es von Nöten, messbare Kriterien zu definieren. Auf dem Nutzerverhalten im Bereich des Internets (siehe Kapitel 3.3.1) basieren hierbei eine Vielzahl von allgemeinen Richtlinien. Hierbei ist zu erwähnen, dass nur eine Auswahl an allgemeinen Richtlinien präsentiert wird, da der Umfang den Rahmen dieser Arbeit überschreiten würde. Die Auswahl beschränkt sich nach den häufigsten Ursachen, die für das Scheitern von Nutzern im Web verantwortlich sind. Folgende Abbildung soll die erwähnten Ursachen visualisieren:

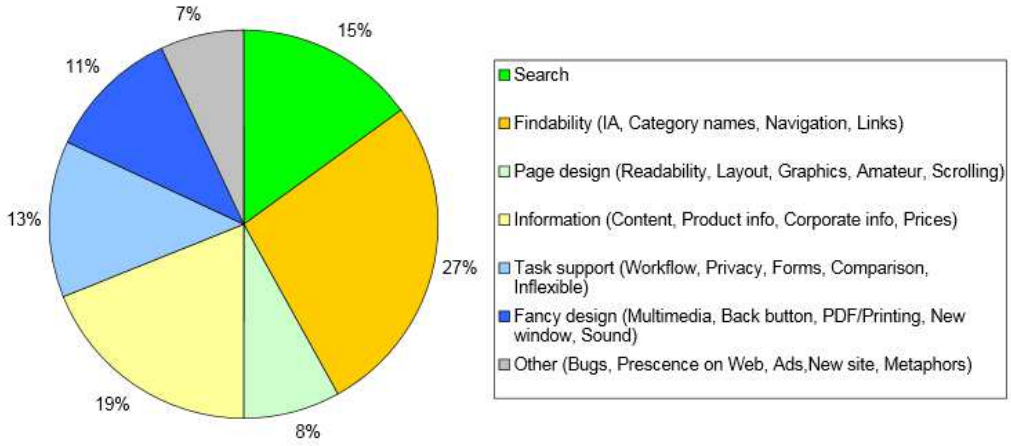


Abbildung 3.2: Ursachen, an denen Nutzer im Web scheitern nach Kategorien

Quelle: Nielsen & Loranger, 2006

Anhand der Grafik wird sofort ersichtlich, dass explizit den Punkten *Information* und *Findability* besondere Betrachtung entgegengebracht werden muss.

Struktur einer Webseite

Hierbei ist es besonders wichtig darauf zu achten, dass Benutzer mit dem Besuch einer Webseite aufgrund bisheriger Erfahrungen im Internet mit bestimmten Erwartungen an die Struktur einer Webseite herantreten, zum Beispiel der Punkt, wo entsprechende Informationen zu finden sind. Im Zuge dessen sollten Produkte beispielsweise nicht nach den Namen der einzelnen Hersteller organisiert werden, sondern vielmehr nach Attributen, die im Nutzungskontext angesiedelt werden, wie zum Beispiel Größe, Preis oder ähnliches. (Vgl. Nielsen, et al., 2006)

Konsistente, redundanzfreie und den Konventionen entsprechende Navigation

Eine konsistente Webseite trägt dazu bei, den Benutzern deren aktuellen Standort innerhalb der Internetseite deutlich zu machen und zu erkennen, welche Maßnahmen im nächsten Schritt getätigt werden müssen. Des Weiteren ist es sinnvoll, die Navigation nach den Konventionen im Web entweder links oder oben zu platzieren, um eine schnelle Auffindung zu gewährleisten. Die Hauptnavigation muss dabei in jedem Fall sichtbar sein. Im Bereich der Redundanz kommt es in Häufiger Form vor, dass mehrere ähnliche Navigationsbereiche dazu beitragen, die intuitive Bedienung zu erschweren, da die Struktur der Webseite dabei in einem schlechteren Maße zu erkennen sei. Weniger Elemente auf einer Internetseite können schneller erfasst werden und sind zudem leichter einzuprägen. (Vgl. Nielsen, et al., 2006)

Aussagekräftige Links und Kategorienamen

Kategorienamen und Links sollten exakt beschrieben werden, damit der Nutzer vorhersehen kann, was ihn im nächsten Schritt erwartet. Daher ist es ratsam, allgemeine Begriffe wie „weiter“ oder „hier klicken“ nicht zu verwenden. Effektiver sind in diesem Fall kurze Links, die mit einem Schlüsselwort verknüpft sind. Stehen diese Links zum Beispiel in einer Liste, sind diese für den Nutzer schneller wahrnehmbar, wenn die Links nicht allesamt mit dem gleichartigen Wort anfangen. (Vgl. Nielsen, et al., 2006)

Klare Unterscheidbarkeit von Links und Inhalten

Hierbei ist darauf zu achten, dass ein Link nicht nur Auskunft darüber geben soll, an welchen Ort er führt, sondern auch darauf hinweisen soll, wenn die betreffende Seite zum Beispiel gerade besucht wird. In anderen Fällen ist lediglich substanziell, diesen auch klar als solchen zu kennzeichnen. Das heißt in diesem Fall nicht, dass dieser heutzutage zwangsweise blau und unterstrichen sein muss, umgekehrt heißt das aber auch, dass ein normaler Text nicht blau gestaltet werden sollte, da viele Benutzer diesen dann als Verlinkung in Zusammenhang bringen. (Vgl. Nielsen, et al., 2006)

4 Usability Engineering

Anhand der in Kapitel 2 und 3 aufgezeigten Punkte wurde herauskristallisiert, dass der Nutzer bei der Produktentwicklung immer im Mittelpunkt stehen sollte. Eine optimale Lösung kann für den Nutzer jedoch dann gewährleistet werden, wenn der Aspekt der Usability von Anfang an in den Entwicklungsprozess eingeplant und integriert wird. Um diese Ziele zu erreichen, wurde das *Usability Engineering* eingeführt. Dabei handelt es sich um einen Prozess, der die allgemeinen Komponenten des Software Engineering, wie zum Beispiel die Anforderungserhebung mit den Methoden der Usability verbindet. Das Ziel soll dabei sein, Usability als Produkteigenschaft messbar zu gestalten. (Vgl. Nielsen, 1993)

Der Autor Nielsen definiert in seinem Buch den Begriff mit folgenden Sätzen:

„Usability engineering is not a one-shot affair where user interface is fixed up before the release of a product. Rather, usability engineering is a set of activities that ideally take place throughout the lifecycle oft the product, with significant activities happening at the early stages before the user interface has even been designed.“ (Nielsen, 1993)

In der Regel ist es so, dass Usability-Experten am Ende eines Projekts involviert werden, um einen Usability-Test durchzuführen. Die Praxis hat gezeigt, dass dabei die Resultate dieser Tests häufig nicht mehr in das Produkt umgesetzt werden können. Gründe sind meistens die fehlende Zeit bis zur Veröffentlichung oder das Produkt sich bereits auf dem Markt befindet und gravierende Änderungen dadurch nicht mehr getätigt werden können. Zudem kommt hinzu, dass letzte Änderungen wesentlich kostenintensiver ausfallen. Das beweist, dass es unabkömmlich ist, eine Usability-Evaluation bereits in den frühen Phasen der Produktentwicklung einzubeziehen, um somit das Usability Engineering mit dem Software Engineering zu verknüpfen. (Vgl. Nielsen, 1993)

4.1 Usability Engineering Lifecycle

Der *Usability Engineering Lifecycle* bringt die einzelnen Aufgaben in die korrekte Reihenfolge, die in den Entwicklungszyklus eines Produktes integriert werden, um Usability zu erreichen. *Mayhew* definiert diesbezüglich folgende Aufgaben: (Vgl. Mayhew, 1999)

- Strukturierte Aufgaben, um die Usability Anforderungen zu analysieren
- Explizite Aufgaben, um Usability Ziele festzulegen, die sich in der Regel aus der Anforderungsanalyse ergeben
- Aufgaben zur Unterstützung des Designs der Benutzerschnittstelle

Mayhew hat diesbezüglich ein Modell entwickelt, welches diesen Prozess verdeutlicht und diesen in einzelne Phasen unterteilt. Die nachfolgende Abbildung 4.1 ist an Mayhew angelehnt und wurde der *Hochschule für Technik und Wirtschaft Chur* entnommen, da diese den Prozess in vereinfachter Form darstellt, welcher sich effektiver auf das Projekt IMPACT projizieren lässt.

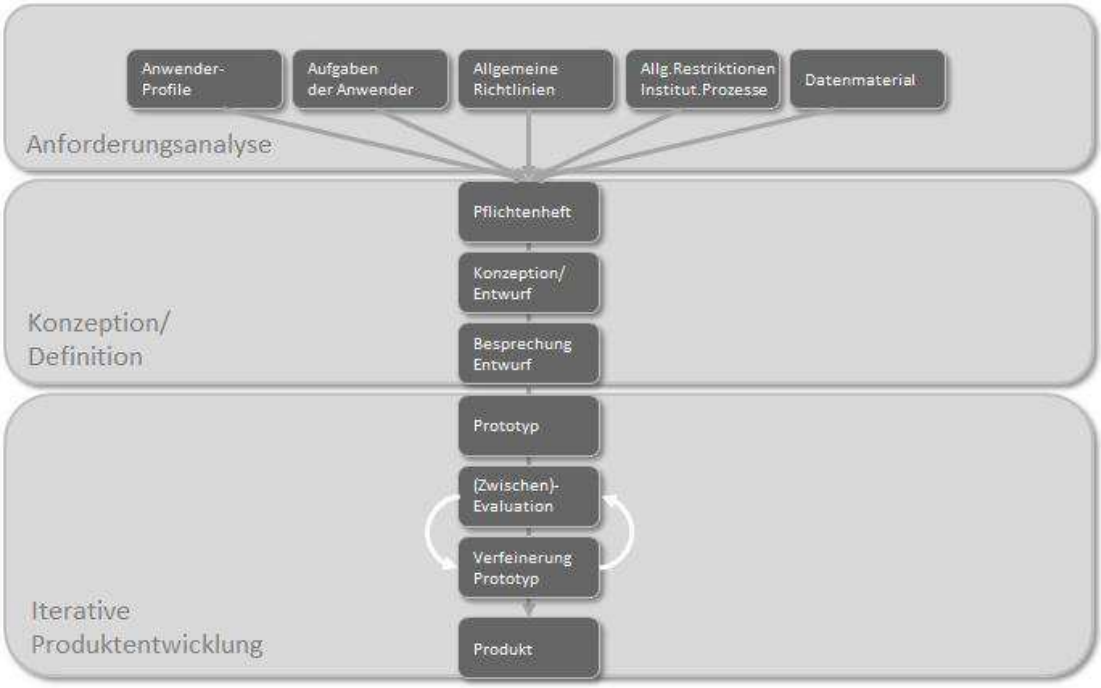


Abbildung 4.1: Phasen des Usability-Engineering-Lifecycles

Quelle: HTW Chur in Anlehnung an: Mayhew, 1999

Anforderungsanalyse

Zu Beginn dieser Phase werden Anforderungen, die der Endanwender an das Produkt stellt, gesammelt und definiert. Diesbezüglich ist es ratsam, vor der Anforderungsdefinition die eigentliche Zielgruppe zu definieren. Hierbei werden sogenannte Nutzerprofile erstellt. Dabei werden zum Beispiel Umfragen eingesetzt, aus denen Rückschlüsse in Bezug auf die Anforderungen gezogen werden können. Nach der Bestimmung des Nutzerprofils muss im nächsten Schritt geklärt werden, in welchem Aufgabengebiet das Produkt Unterstützung leisten soll und wie die einzelnen Szenarien und Anwendungsfälle aufgebaut sind. Des Weiteren sollten Usability-Ziele festgelegt werden. Diese sollen dabei in erster Linie dazu dienen, den Fokus auf das Design der Anwendung zu legen und um zukünftige Bewertungen effizienter durchführen zu können. Darüber hinaus ist es ratsam, allgemeine Normen, Gesetze und Usability-Richtlinien zu berücksichtigen. Entsprechende Normen und Kriterien wurden in den Kapiteln 2.3 und 3.3.2 abgehandelt.

Diese Phase nimmt in der Projektgruppe IMPACT ebenfalls eine wichtige Rolle ein, da diese in einem frühen Zeitraum des Projekts einbezogen werden kann und in dieser Phase eine Vielzahl essentieller Punkte aufgegriffen werden, die sich im zukünftigen Endprodukt widerspiegeln. Zu Beginn können auch hier Anforderungen definiert werden, die für das innerbetriebliche Innovationsmanagementsystem umgesetzt werden müssen. Vor der Anforderungsdefinition wurde in Absprache mit Dr. Joachim Kurzhöfer die genaue Zielgruppe besprochen, sodass die Anforderungen effizient gesammelt werden können. Im nächsten Schritt erfolgt eine Online-Umfrage bezüglich des zukünftigen Innovationsmanagementsystems, welches die Projektgruppe IMPACT entwickeln soll. Um diese Anforderungen zu festigen, sollen zusätzliche Interviews mit potentiellen Nutzern geführt werden, um die Anforderungsdefinition zu festigen. Im Bereich der Usability-Richtlinien und Normen können die aufgezählten Punkte in den Kapiteln 2.3 und 3.3.2 genutzt werden. (Vgl. HTW Chur; Mayhew, 1999)

Da innerhalb der Projektgruppe das Vorgehensmodell „Scrum“ genutzt wird, werden im Rahmen der Anforderungsanalyse sogenannte „User-Storys“ eingesetzt, da es die übliche Vorgehensweise der agilen Softwareentwicklung darstellt. Zudem können neben den in dieser Arbeit aufgezeigten Richtlinien für Usability eigene auf die Anwendung zugeschnittene Grundsätze definiert werden, da es in agilen Softwareprojekten dazu kommen kann, dass sich bestimmte Anforderungen ändern oder zusätzliche hinzukommen.

Konzeption und Definition

Die Ergebnisse, die aus der Anforderungsanalyse entstanden sind, werden in das Pflichtenheft eingetragen. Dieses gibt Auskunft über die einzelnen Arbeitsschritte, die für die Entwicklung des Produkts als relevant erachtet werden. Im nächsten Schritt erfolgt anhand der vorliegenden Ergebnisse der Entwurf konzeptioneller Modelle. Dazu gehören zum Beispiel Entwürfe der Benutzeroberfläche in Form von Wireframes[2] und Mockups[3]. Ein vorheriger Entwurf der Benutzeroberfläche hat beispielsweise den Vorteil, frühzeitig Schwachstellen identifizieren oder vermeiden zu können. (Vgl. HTW Chur; Mayhew, 1993)

Die Ergebnisse der Anforderungsanalyse können innerhalb der Projektgruppe beispielsweise in die Projektdokumentation eingepflegt werden und mithilfe des genutzten Projektmanagementtools „JIRA“ in Form eines Boards festgehalten werden. Zur Visualisierung

---

[2] Bei einem Wireframe handelt es sich um eine optische Darstellung der Anforderungen an eine Software. Die Funktionen werden dabei nur grafisch angedeutet. Der Wireframe wird meist aus einer Skizze gefertigt und stellt in der Konzeptionsphase den Schritt der reinen Skizzierung einer Anwendung dar. (Vgl. Arnowitz, et al., 2010)

[3] Im Vergleich zum Wireframe digitalisiert das Mockup, wie die Anwendung zukünftig gestaltet beziehungsweise aufgebaut sein könnte. Aus Entwicklungs- und Kundensicht ist ein Mockup somit konkreter und anschaulicher als ein Wireframe. (Vgl. Lauesen, 2005).

der unterschiedlichen Masken des Innovationsmanagementsystems können beispielsweise Wireframes verwendet werden. Die Funktionen werden dabei nur grafisch angedeutet. Ein Wireframe wird meist aus einer Skizze gefertigt und stellt in der Konzeption den Schritt nach der reinen Skizzierung einer Anwendung dar. Es dient dabei als saubere Darstellung von Skizzen. Gegenüber einem Mockup verfügt das Wireframe hingegen über keinerlei Designinformationen, wie die Ausrichtung nach einem Corporate Identity. Das Wireframe gibt darüber Aufschluss, wie die Anordnung der User-Interface-Elemente angedacht ist. (Vgl. Arnowitz, et al. 2010)

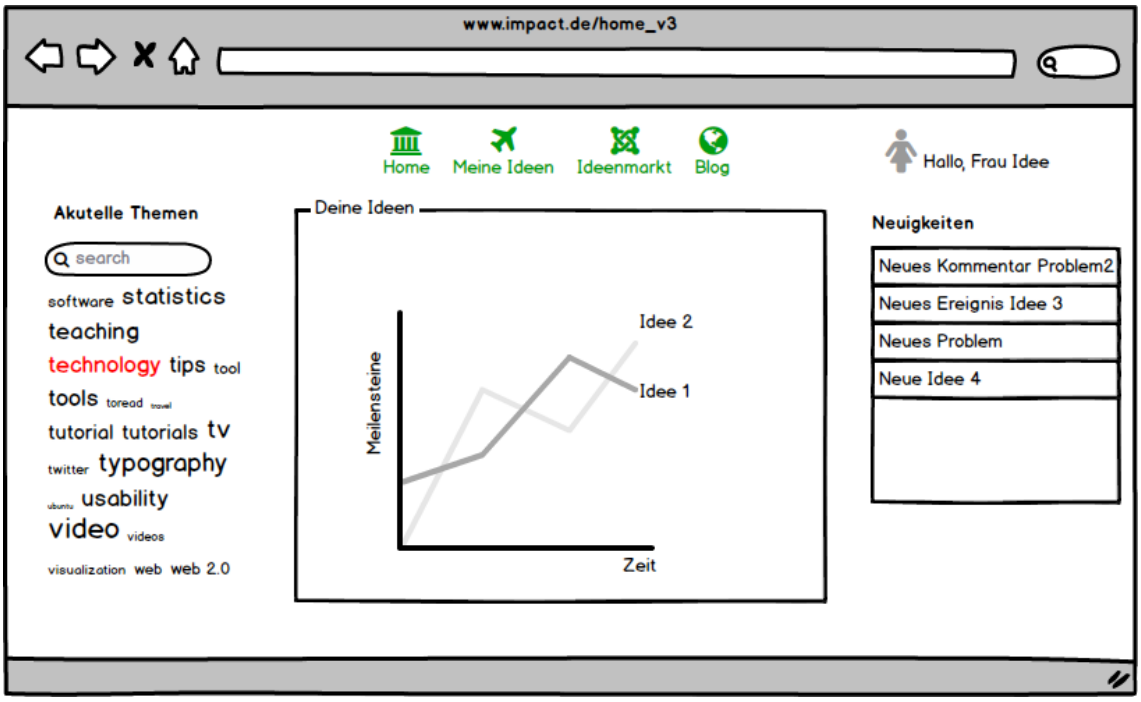


Abbildung 4.2: Startseite Wireframe von IMPACT

Quelle: Eigene Darstellung

Wie in Abbildung 4.2 veranschaulicht, werden im Wireframe lediglich Anforderungsdefinitionen getroffen. Neben der Ausrichtung und Platzierung von Elementen lässt sich auch erkennen, welche funktionalen Anforderungen an die Anwendung gestellt werden, beispielsweise eine Suchfunktion. Wireframes können bereits mit vielen kostenlosen Programmen, wie zum Beispiel Balsamiq Mockup, gestaltet werden. Diese Programme dienen dabei oft nicht nur zur Umsetzung von Wireframes, sondern auch direkt zur Erstellung von Mockups. Für die Projektgruppe IMPACT ist dies von daher auch zu empfehlen und kann als Handlungsempfehlung angesehen werden.

In Verbindung mit Scrum ist diese Variante des Entwurfs ebenfalls von Vorteil. Agile Entwicklungsmethoden zeichnen sich durch kürzere Entwicklungszyklen aus, was oft dazu führt, dass die Konzentration auf funktionale Anforderungen fokussiert wird und nicht-

funktionale Aspekte wie Usability, des Öfteren in den Hintergrund geraten. Dieses Problem wurde im Rahmen eines Forschungsprojekts der University of Calgary behandelt. Dabei wurde ein Softwaretool namens „Active Story Enchanced“ (http://activestoryenhanced.codeplex.com/) entwickelt, welches die Erstellung von Wireframes und Mockups durch Freihand-Skizzen sowohl vereinfachen als auch beschleunigen soll. Dadurch soll insbesondere die im Usability-Engineering genutzte Methode auch in Verbindung agiler Entwicklungsprojekte genutzt werden können. Daher bietet es sich auch in diesem Fall an, diese Empfehlung auf die Projektgruppe IMPACT zu projizieren und während der Entwicklung des Innovationsmanagementsystems einzubeziehen. (Mittelstandsforschung, 2011)

Iterative Produktentwicklung

Wurden grundlegende Konzepte und Entwürfe festgelegt, können funktionale Prototypen realisiert werden. Diese können aus Nutzersicht bewertet und bei Bedarf optimiert werden, sodass eine iterative Produktentwicklung stattfindet. Dabei ist es ratsam, bestimmte Funktionalitäten und deren Features frühestmöglich hinsichtlich der Benutzerfreundlichkeit zu überprüfen. Je nach Produktumfang und Komplexität des Produkts kann entschieden werden, wie weitreichend die Evaluationsdurchläufe ausfallen und Nachbesserungen getätigt werden müssen, bis das Produkt letztendlich fertiggestellt ist. (Vgl. HTW Chur; Mayhew, 1999)

Da es sich bei dem Projekt um ein studentisches Projekt handelt und zudem Scrum als Vorgehensmodell genutzt wird, müssen hierbei einige Anpassungen getroffen werden. Laut der *Ergonomen Usability AG* wird gute Usability des Endergebnisses erreicht, wenn der Scrum-Prozess mit Elementen angereichert wird, die eine Benutzerorientierte Entwicklung mit einbeziehen. Folgende Abbildung 4.3 verdeutlicht, inwieweit Usability im Scrum-Entwicklungsprozess integriert wird:

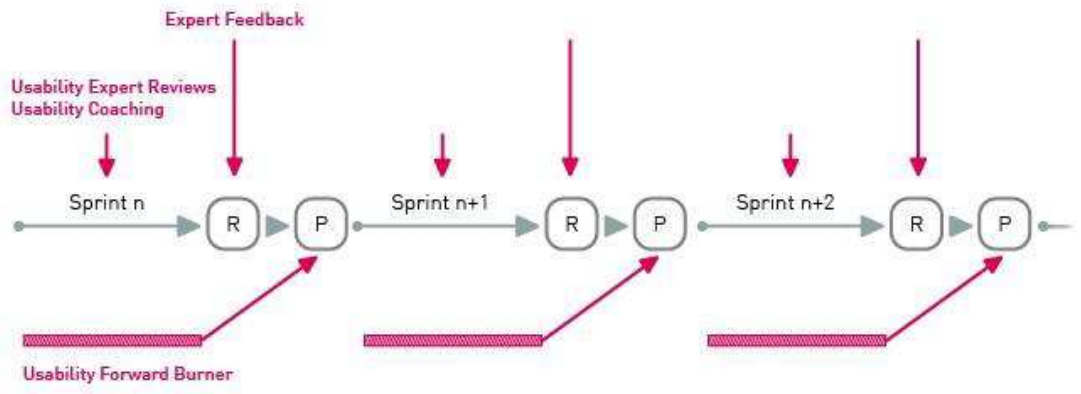


Abbildung 4.3: Usability im Scrum-Prozess

Quelle: Die Ergonomen Usabillity AG, 2012

Die User-Stories beschreiben in vollständiger Weise, wie die Anwendung umgesetzt werden soll, nämlich in Form von Anforderungen. Der Usability-Forward-Burner (ein vorangestellter Usability-Sprint vor dem eigentlichen Entwicklungssprint) vergleicht die definierten User-Stories mit den Benutzern. Basis bilden hierfür die zuvor aufgezeigten Wireframes und Mockups. Im nächsten Schritt wird das Usability-Coaching in den Sprint integriert. Die dort produzierten Tabellen, Eingabemasken, Workflows und weitere Elemente können beispielsweise auf Usability überprüft werden. Dies kann im Rahmen der Projektgruppe zum Beispiel in den wöchentlichen Meetings mit den jeweiligen Betreuern der Projektgruppe erfolgen. Zudem können interne Vorschläge und Anmerkungen an den Product-Owner weitergereicht werden, der diese dann weiterleitet und Feedback einholt. Diese Verbesserungsvorschläge können direkt dem Entwicklungsteam übergeben werden. Die hat den Vorteil, dass die Benutzungsqualität schon vor dem eigentlichen Sprint-Review optimiert werden kann. Hinzu kommt, dass hinsichtlich des Product-Backlogs Entlastungsmöglichkeiten realisiert werden können. (Ergonomen, 2012)

Im Bereich der Projektgruppe IMPACT wird das komplette Team als Entwicklerteam angesehen, jedoch gibt es vereinzelte Rollen, die vergeben wurden, auch die des Product Owners. In diesem Fall bietet es sich an, wenn die Projektgruppe als Ganzes mit den Betreuern des Projekts interagiert und die Sichtweisen teilt.

5 Schlussbetrachtung

Zu Beginn werden die Ergebnisse zusammengefasst. Im Anschluss wird ein Fazit gezogen, indem insbesondere der Hauptteil des Projekts reflektiert wird und die Handlungsempfehlungen für die Projektgruppe näher beleuchtet werden.

5.1 Zusammenfassung der Ergebnisse

Die vorliegende Arbeit hatte das Ziel, grundlegende Informationen im Bereich des User Interfaces und Usability aufzuzeigen. Aus diesen Informationen wurden Handlungsempfehlungen abgeleitet, wie die Projektgruppe IMPACT diese innerhalb des Projekts umsetzen kann und welche Informationen als relevant verzeichnet werden können.

Nach der genauen Beschreibung von Motivation und Problemstellung sowie dem Aufbau der Arbeit in Kapitel eins, wurden in Kapitel zwei generelle Betrachtungen zu User Interfaces aufgeführt. Dabei wurde der Begrifft schrittweise bestimmt, indem unter anderem der Bezug zur Mensch-Maschine-Interaktion dargelegt wurde und verschiedene Arten von User Interfaces aufgezeigt wurden. Zudem wurden wesentliche Kriterien für das Design ergonomischer Benutzerschnittstellen dargelegt. Kapitel drei widmete sich des zweiten Schwerpunkts dieser Arbeit, nämlich der Usability. Hierbei erfolgten zu Beginn Begriffsbestimmungen im Bereich der Usability. Dabei wurde deutlich, dass dieser Begriff sehr vielschichtig ist und mehrdimensional aufgefasst werden kann. Im Zuge dessen wurde Usability im Kontext des World Wide Webs näher betrachtet. Zu diesem Punkt wurden Gründe und Richtlinien aufgezählt, denen besondere Wichtigkeit zugesprochen wird, um eine gute Usability zu erreichen. Im vierten Kapitel dieser Arbeit wurden Möglichkeiten aufgezeigt, wie Usability in einen Softwareentwicklungsprozess integriert werden kann. Hierzu wurde zu Beginn der Begriff des Usability-Engineerings aufgeführt. Des Weiteren wurde der Usability-Engineering-Lifecycle in diesen Kontext näher gebracht, da dieser Auskunft darüber erteilt, wie die wesentlichen Aufgaben in den Entwicklungszyklus eines Produktes aufgenommen werden. Darauffolgend wurden diesbezüglich wesentliche Punkte beleuchtet und Handlungsempfehlungen abgeleitet, die für die Projektgruppe IMPACT als relevant erachtet werden können.

5.2 Fazit

Generell lässt sich festhalten, dass Usability und speziell das Usability-Engineering eine essentielle Rolle im Bereich des Entwicklungsprozesses eines Produktes einnehmen. Aufgrund der Themenkomplexität bietet es sich an, die in Kapitel 4.1 aufgezeigten Handlungsempfehlungen in die Innovationsplattform zu integrieren. Dabei handelt es sich lediglich um eine vereinfachte Variante des Usability-Engineerings und kann somit effizienter innerhalb eines studentischen Projekts realisiert werden. Der Schwerpunkt sollte hierbei insbesondere den Phasen A*nforderungsanalyse* sowie *Konzeption und Entwurf* zugesprochen werden, da die dort zu treffenden Entscheidungen übergeordnete Auswirkungen auf das Endprodukt haben können und sich auf das gesamte Projekt auswirken.

Usability-Engineering kann somit als Werkzeug angesehen werden, welches innerhalb des Scrum-Prozesses eine positive Unterstützung offeriert. Es darf jedoch nicht außer Acht gelassen werden, dass die endgültige Entscheidung für ein Produkt letztendlich auf den individuellen Anforderungen eines Projektes basiert.

Literaturverzeichnis

AG, Die Ergonomen Usability. 2012. User Centered Scrum. [Online] 2012.
archlinux. 2015. archlinux: Boot laders. [Online] 2015. [Zitat vom: 20. Juni 2015.]
Arnowitz, Johnathan, Arent, Michael und Berger, Nevin. 2010. *Effective prototyping for software makers.* s.l. : Elsevier, 2010.
Baggen, Robert. 2003. *Normen und Zertifizierung. Usability praktisch umsetzen. .* München : Carl Hanser, 2003.
Böhringer, Joachim, Bühler, Peter und Schlaich, Patrick. 2011. *Kompendium der Mediengestaltung.* Heidelberg : Springer Verlag, 2011.
Chur, Hochschule für Technik und Wirtschaft. 2013. Das Usability Engineering Lifecycle. [Online] 2013.
DIN EN ISO 9241 - 11. 1998. *Ergonomie der Mensch-Maschine-Interaktion.* Berlin : s.n., 1998.
—. 1998. *Ergonomische Anforderungen für Bürotätigkeiten und Bildschirmgeräten.* Berlin : s.n., 1998.
Hartmann, Markus. 2008. *Usability Untersuchung eines Internetauftrittes nach DIN EN ISO 9241.* s.l. : Diplomica Verlag, 2008.
Herczeg, Michael. 2005. *Software-Ergonomie: Grundlagen der Mensch-Maschine-Kommunikation.* München : s.n., 2005.
Hix, Deborah und Hartson, Rex H. 1993. *Developing User Interfaces: Ensuring Usability Through Product & Process .* s.l. : John Wiley & Sons, 1993.
Lauesen, Soren. 2005. *User interface design: a software engineering perspective.* [Hrsg.] Pearson Educatoin. 2005.
Mayhew, Deborah J. 1999. *The Usability Engineering Lifecycle.* San Diego : Academic Press, 1999.
Mittelstandsforschung, Institut für. 2011. Usability in Germany. *Agile Entwciklung und Usability-Engineering - ein Widerspruch?* [Online] 2011.
Nielsen, Jakob und Loranger, Hoa. 2006. *Web-Usability.* München : Addison-Wesley, 2006.
Nielsen, Jakob. 1993. *Usability Engineering.* s.l. : Elsevier, 1993.
OnPage. 2015. OnPage Wiki: Digitales Marketing Lexikon. *User Interface.* [Online] 2015. [Zitat vom: 20. Juni 2015.]
Sarodnick, Florian und Brau, Henning. 2006. *Methoden der Usability Evaluation.* s.l. : Hans Huber, 2006.
Schweibenz, Werner und Thissen, Frank. 2003. *Qualität im Web: Benutzerfreundliche Webseiten durch Usability-Evaluation.* s.l. : Springer, 2003.

B.1.9. Programmierkonventionen

Programmierkonventionen

Seminararbeit
im Rahmen der Projektgruppe IMPACT

Themensteller:	Prof. Dr.-Ing. Jorge Marx Gmez
Betreuer:	Dr. Joachim Kurzhfer
	Jens Siewert
	Stefan Wunderlich
Vorgelegt von:	Jan Wiesemann
	Kaspersweg 13a
	26131 Oldenburg
	0151/18740884
	jan.wiesemann@uni-oldenburg.de
Abgabetermin:	22. Juni 2015

Inhaltsverzeichnis

Abkürzungsverzeichnis

ABAP	Advanced Business Application Programming
MISRA	Motor Industry Software Reliability Association
OOP	Objektorientierte Programmierung
SQL	Structured Query Language

Quelltextverzeichnis

1 Einleitung

1.1 Problemstellung

Die Verbreitung von Computern in viele alltägliche Anwendungsbereiche nimmt zu. Damit gewinnt auch die Sicherstellung ihrer zuverlässigen und korrekten Funktionsweise an Bedeutung. (Vgl. Liggesmeyer (2001), S. 2)
So kann sich qualitativ schlechte Software negativ auf den Erfolg eines Unternehmens auswirken. Ein fehlerhaftes Produkt führt nach Vertrauens- und Kundenverlust zu finanziellem Schaden. Je nachdem, wo die Software zum Einsatz kommt, kann sogar gesundheitlicher Schaden entstehen. Bedienbare, zuverlässige und korrekte Software hat somit direkte Auswirkungen auf die Wirtschaft, die Gesundheit und das allgemeine Wohlergehen. (Vgl. Schneider (2012), S. 6f.)
Um dabei einen Kompromiss aus Qualität und Kosten zu erzielen und so die Wirtschaftlichkeit sicherzustellen, müssen Fehler im frühestmöglichen Stadium erkannt und behoben werden. Nur so lassen sich Fehlervermeidungs- und Fehlerbeseitungungskosten reduzieren. (Vgl. Schneider (2012), S. 16)
Dazu kann man schon bei der Entwicklung des Quelltextes einer Software ansetzen. Mit der Festlegung und Einhaltung bestimmter Regeln lassen sich Fehler vermeiden und so die Softwarequalität erhöhen.
Empfehlungen zum Programmieren gibt es bereits seit den 1970er Jahren. Viele gehen dabei allerdings nicht über die einfache Nennung von Regeln hinaus. So spricht beispielsweise Timm Grams, ehemaliger Informatikprofessor an der Hochschule Fulda, in seinem Werk „Denkfallen und Programmierfehler“ von einer „größtmöglichen Einfachheit der Ablauf- und Datenstrukturen“ (Grams (1990), S. 82) oder von „exakten Namen“ (Grams (1990), S. 86) für Variablen. Was darunter im Detail zu verstehen ist, bleibt jedoch offen.

1.2 Zielsetzung

Ziel dieser Arbeit ist es, konkrete und praxisnahe Empfehlungen für das Schreiben von Quelltext zu geben, um so Lesern die Entwicklung von qualitativ hochwertigen Programmen zu ermöglichen. Außerdem sollen Hinweise gegeben werden, wie sich die Einführung der daraus abgeleiteten Regeln in der beruflichen Praxis durchführen und deren Einhaltung sicherstellen lässt.
Dabei richtet sich die Arbeit an Leser, die bereits Erfahrungen im Umgang mit Programmiersprachen besitzen und mit den Terminologien vertraut sind.

1.3 Vorgehensweise

Die Arbeit gliedert sich in sechs Kapitel. Nach der Einleitung geht der Grundlagenteil in Kapitel 2 zunächst auf die Gründe und Ziele sowie Arten und Inhalte von Programmierkonventionen ein. Darauf folgt Kapitel 3 mit einer Auswahl von Programmierkonventionen, welche die gebräuchlichsten Dinge behandeln. Kapitel 4 beschreibt Möglichkeiten, die Einhaltung von Regeln sicherzustellen während Kapitel 5 darauf eingeht, was es bei dem Einsatz von Programmierkonventionen in Teams zu beachten gibt. In Kapitel 6 folgt schließlich das Fazit.
Die Werke „Clean Code – A Handbook of Agile Software Craftsmanship“ des US-amerikanischer Softwareentwicklers Robert C. Martin und „Weniger schlecht Programmieren“ des deutschen Softwareentwicklers Johannes Jander und der deutschen Journalistin und Schriftstellerin Kathrin Passig sind besonders hervorzuheben. Sie gehen auf einzigartig detaillierte Weise auf Programmierpraktiken ein und finden deshalb in dieser Arbeit die meiste Verwendung.
Zum besseren Verständnis werden in dieser Arbeit Quelltextbeispiele verwendet. Diese erheben trotz Ähnlichkeit zur Programmiersprache Java keinen Anspruch auf Korrektheit und stellen vielmehr Pseudocode dar, der zur Veranschaulichung der beschriebenen Konzepte dient.

2 Grundlagen von Programmierkonventionen

2.1 Begriffserklärung

Programmierer müssen beim Schreiben von Quelltext häufig bestimmte Regeln zum Programmierstil beachten. Diese Programmierregeln bezeichnet man als Programmierkonventionen. (Vgl. Liggesmeyer (2001), S. 251)

2.2 Gründe

Programmiersprachen wurden entwickelt, um die Maschinensprache von Computern in eine für Menschen lesbare Form zu bringen. Menschen denken allerdings weniger logisch und so bietet sich Raum für Missverständnisse. Darüber hinaus enthält der reine Quelltext keine weiteren Informationen über Fehler, Zeit oder Budget. (Vgl. Passig/Jander (2014), S. 17f.)
Laut dem englischen Softwarearchitekten und Buchautor Martin Fowler sei leicht verständliche Software sogar wichtiger als ein effizientes Programm. So sei eine kürzere Einarbeitungszeit in fremden Quelltext gegenüber mehr Rechenzeit durch in ineffizientes Programm als wichtiger zu bewerten. Außerdem halte ein verständlicher Quelltext den Kopf frei für die wesentlichen Dinge und vereinfache die Fehlersuche. Ein gutes Design sei somit ausschlaggebend für schnelle Softwareentwicklung. (Vgl. Fowler (2005), S. 44f.)
Betrachtet man den Lebenszyklus eines Programms, fallen laut dem Unternehmen Sun Microsystems 80 % der Kosten für die Wartung an. Da diese Wartung nie nur von einer Person durchgeführt wird, braucht es Methoden, um die Lesbarkeit des Quelltextes zu erhöhen. (Vgl. Sun Microsystems, Inc. (1997), S. 1)

2.3 Ziele

Programmierkonventionen tragen dazu bei bestimmte Qualitätseigenschaften einzuhalten, um den Erfolg eines Softwareprojektes sicherzustellen. Fehlerträchtige Konstrukte, welche die Verständlichkeit des Quelltextes reduzieren oder die Portabilität behindern sollen vermieden werden. (Vgl. Liggesmeyer (2001), S. 251)
Sie dienen dazu, Programmfehler zu vermeiden und liefern Vorgehensweisen, um den Quelltext bei zunehmender Komplexität weiterhin überblicken und verstehen zu können. (Vgl. Grams (1990), S. 82)
Dazu soll unter anderem die Verständlichkeit, Änderbarkeit und Aussagefähigkeit von Bezeichnern und Kommentaren sichergestellt werden. (Vgl. Liggesmeyer (2001), S. 286)

2.4 Arten

Programmierkonventionen gibt es in unterschiedlichen Bereichen. Sie gelten für ganze Unternehmen oder einzelne Projekte und Programmiersprachen. (Vgl. Liggesmeyer (2001), S. 251)
Dabei unterscheidet man zwei Bereiche. Sollen die Regeln durch Vereinheitlichung des Programmierstils die Effizienz erhöhen, spricht man von Notationskonventionen. Zielen sie dagegen durch die Verwendung einheitlicher Lösungsmuster auf die Reduktion von Fehlern ab, handelt es sich um Sprachkonventionen. (Vgl. Hoffmann (2013), S. 65f.)
Die jeweiligen Regeln können einschränkend oder erweiternd sein. Erstere verbieten bestimmten Anweisungen oder Konstrukte während letztere Vorgaben zu speziellen Anwendungsfällen enthalten. Dabei kann es sich zum Beispiel um die Fehlerbehandlung oder die Prüfung von gültigen Werten bei der Ein- und Ausgabe handeln. (Vgl. Liggesmeyer (2001), S. 255)
Außerdem gibt es Softwarerichtlinien, die über syntaktische und semantische Regelungen hinausgehen und zum Beispiel Vorgaben zum Gebrauch einer bestimmten Programmiersprache geben. (Vgl. Hoffmann (2013), S. 65)

2.5 Inhalte

Programmierkonventionen behandeln je nach Anwendungsgebiet verschiedene Themen. Die offiziellen Java Code Conventions regeln die Dateiorganisation, Formatierung des Quellcodes, Benennung von Objekten und geben mit Quelltextbeispielen Empfehlungen für spezielle Programmierpraktiken. (Vgl. Sun Microsystems, Inc. (1997))
Der offizielle Google Java Style schreibt ebenfalls die Verwendung von Quelldateien, Formatierungen, Benennung von Objekten und Programmierpraktiken vor. (Vgl. Google, Inc. (2014))
Die Entwicklungsrichtlinie eines großen deutschen IT-Dienstleisters für die Programmiersprache Advanced Business Application Programming (ABAP) des Softwareherstellers SAP geht zusätzlich auf Regelungen zum Aufbau von Programmen, Mehrsprachigkeit, Berechtigungsprüfungen oder Vorgaben zur Dokumentation ein.
Die 1998 von der Motor Industry Software Reliability Association (MISRA) entwickelten Richtlinien für die Programmiersprache C enthalten heute 141 Regeln zur Verwendung von Bezeichnern, Datentypen, Konstanten, Kontrollstrukturen oder Funktionen. Mittlerweile kommen die ursprünglich für die Automobilindustrie erstellten Richtlinien auch in anderen Branchen wie der Avionik oder der Medizintechnik zum Einsatz. (Vgl. Hoffmann (2013), S. 74f.)

3 Ausgewählte Programmierkonventionen

3.1 Sprache

Für nicht englischsprachige Programmierer stellt sich die Frage, ob bei der Benennung von Objekten und dem Schreiben von Kommentaren die Muttersprache oder Englisch verwendet werden sollte. Argumente gibt es für beide Varianten.
Für die Muttersprache spricht die bessere Beherrschbarkeit und somit die geringere Fehleranfälligkeit. Es werden weniger Schreibfehler gemacht und Namen und Kommentare sind einfacher festzulegen.
Für Englisch spricht die Tatsache, dass Programmiersprachen grundsätzlich auf englischen Begriffen beruhen. Eine Zweisprachigkeit im Quelltext (siehe Quelltext 1) würde die Lesbarkeit einschränken.

```
1 //Negativbeispiel
2 getKundenNummerFromDatenbank();
3
4 //Positivbeispiel
5 getCustomerNumberFromDatabase();
```

Quelltext 1: Verwendung von englischer und deutscher Sprache

Außerdem sorgt die Verwendung der Muttersprache für eine Vorverurteilung der Qualität des Gesamtproduktes, da die Vermutung nahe liegt, das gerade Programmieranfänger ihre Muttersprache verwenden.
Wenn die Muttersprache verwendet wird, steht bei Problemen zunächst nur eine kleine Online-Community zur Verfügung, die beim Lösen von Problemen unterstützen kann. Beim Adressieren der internationalen Online-Community oder generell bei der Veröffentlichung muss der Quelltext zunächst übersetzt werden.
Aus den genannten Gründen empfehlen Johannes Jander und Kathrin Passig durchgehend Englisch zu verwenden. Dabei sei amerikanisches Englisch dem britischen Englisch vorzuziehen, da ein Großteil der heute verfügbaren Software aus den USA stamme. (Vgl. Passig/Jander (2014), S. 20ff.)

3.2 Formatierung

Während sich die Funktionalität eines Programms mit der Zeit ändern kann, bleibt die einmal festgelegt Formatierung und damit die Lesbarkeit meist erhalten. Ein sinnvoll formatierter Quelltextes bildet außerdem die Grundlage für eine optimale Kommunikation zwischen mehreren Programmierern. (Vgl. Martin (2009), S. 76)

Untersuchungen zur Lesbarkeit verschiedenen Quelltextgestaltungen haben große Unterschiede zwischen halbwegs brauchbarem und gut lesbarem Quelltext ergeben. Ab diesem Punkt sind weitere Feinheiten weniger entscheidend. Sie sollten jedoch einheitlich sein. Es ist immer sinnvoll, sich einen bereits existierenden Stil anzueignen, als einen eigenen zu erfinden. (Vgl. Passig/Jander (2014), S. 25)

3.2.1 Vertikale Formatierung

Kurze Dateien mit wenig Zeilen sind grundsätzlich einfacher zu verstehen als lange. Dabei sollte der Detaillierungsgrad von oben nach unten zunehmen.
Stark zusammenhängende Elemente wie Deklarationen sollten direkt hintereinander stehen während gedanklich voneinander abzugrenzende Blöcke und Konzepte durch Leerzeilen zu trennen sind.
Um zeitaufwändiges Suchen zu verhindern, sollten Objekte möglichst nah an ihrer Verwendung deklariert werden. Nach dem gleichen Prinzip sind Instanzvariablen am Anfang einer Klasse zu deklarieren. Ebenso macht es Sinn, voneinander abhängige und sich gegenseitig aufrufende Funktionen dicht beieinander zu positionieren. (Vgl. Martin (2009), S. 76ff.)
Das Setzen von geschweiften Klammern bei Verzweigungen, Schleifen, Methoden oder anderen Blöcken kann auf unterschiedliche Art und Weise erfolgen. Dabei spielt es keine Rolle, welche Variante gewählt wird, solange sie einheitlich zum Einsatz kommt. Bei der etablierten und empfohlenen Variante kommt die öffnende Klammer in die Zeile des Blockanfangs und die schließende Klammer unter die letzte Zeile des Blockes (siehe Quelltext 2). (Vgl. Passig/Jander (2014), S. 23ff.)

```
1 //Vriante 1 (empfohlen)
2 public int getVariable1() {
3   return variable1;
4 }
5
6 //Variante 2
7 public int getVariable1()
8 {
9   return variable1;
10 }
11
12 //Variante 3
13 public int getVariable1(){
14 return variable1;}
```

Quelltext 2: Varianten zur Klammernsetzung bei Blöcken

Kommen bei einer Verzweigung multiple Bedingungen zum Einsatz, so sollten diese nicht in einer langen Zeile sondern untereinander stehen (siehe Quelltext 3). So ist auf einen Blick erkennbar, um welche unterschiedlichen Bedingungen es sich handelt. (Vgl. Passig/Jander (2014), S. 23ff.)

```
1 //Negativbeispiel
2 if ((a > b) || (a > c)){
3   return;
4 }
5
6 //Positivbeispiel
7 if ((a > b) ||
8     (a > c)){
9   return;
10 }
```

Quelltext 3: Multiple Bedingungen

3.2.2 Horizontale Formatierung

Untersuchungen haben ergeben, dass die durchschnittliche Zeichenanzahl einer Quelltextzeile in der Praxis zwischen 20 und 60 Zeichen liegt. Um die Lesbarkeit nicht zu gefährden, empfiehlt Martin Fowler maximal 120 Zeichen pro Zeile zu verwenden. (Vgl. Martin (2009), S. 85ff.)
Benötigt eine Zeile dennoch einen Zeilenumbruch, sollte dieser nicht hart hinter dem letzten Zeichen eingefügt werden, sondern an einer sinnvollen Stelle erfolgen (siehe Quelltext 4). (Vgl. Passig/Jander (2014), S. 23ff.)

```
1 //Negativbeispiel
2 Scroll.setLayoutParams(new LayoutParams(LayoutParams.FILL_PATENT, Layo
3   utParams.FILL_PARENT));
4
5 //Positivbeispiel
6 Scroll.setLayoutParams(new LayoutParams(LayoutParams.FILL_PATENT,
7   LayoutParams.FILL_PARENT));
```

Quelltext 4: Zeilenumbrüche bei langen Zeilen

Ein Leerzeichen sollte zum Einsatz kommen, um Argumente voneinander abzugrenzen, die nicht zusammen gehören. Dies sind in der Regel die beiden von Operatoren verarbeiteten Argumente (siehe Quelltext 5).

```
1 //Negativbeispiel
2 sum=var1+var2;
3
4 //Positivbeispiel
5 sum = var1 + var2;
```

Quelltext 5: Leerzeichen bei Operatoren

Bei einer starken Zusammengehörigkeit, wie zum Beispiel den in Klammern stehenden Parametern eines Methodenaufrufes, macht es Sinn, keinen Zwischenraum zu verwenden (siehe Quelltext 6).

```
1 //Negativbeispiel
2 print (dokument1);
3
4 //Positivbeispiel
5 print(dokument1);
```

Quelltext 6: Leerzeichen beim Methodenaufruf

Bei der Auflistung mehrerer Variablen wird oft leerer Raum eingesetzt, um die Deklarationen, Definitionen und Initialisierung optisch untereinander anzuordnen. Dies führt jedoch dazu, dass der Blick eher vertikal durch die so erzeugten Spalten wandert als sich auf eine logisch zusammenhängende Zeile zu konzentrieren und ist deshalb laut Robert C. Martin nicht empfohlen (siehe Quelltext 7).

```
1 //Negativbeispiel
2 int     Kundennummer  = 12345;
3 boolean getsDiscount  = true;
4 string  Nachname      = "Mustermann";
5
6 //Positivbeispiel
7 int Kundennummer = 12345;
8 boolean getsDiscount = true;
9 string Nachname = "Mustermann";
```

Quelltext 7: Deklaration, Definition und Initialisierung mehrerer Variablen

Einrückungen von Blöcken und Methodeninhalten helfen, den Quelltext schneller mit dem Auge zu Scannen. Eine Ausnahme stellen häufig einzeilige Verzweigungen oder Schleifen dar, die jedoch nicht zu empfehlen sind, da sie leicht übersehen werden (siehe Quelltext 8). (Vgl. Martin (2009), S. 85ff.)

```
1 //Negativbeispiel
2 if(a < b){print("a ist kleiner als b.");}
3
4 //Positivbeispiel
5 if(a < b){
6   print("a ist kleiner als b.");
7 }
```

Quelltext 8: Zeilenumbrüche bei einer Verzweigung

Einrückungen sollten grundsätzlich mit Leerzeichen durchgeführt werden, da in einigen Sprachen der Tabulator Teil der Semantik sein kann. Ob zwei, vier oder acht Leerzeichen wie im Linux-Kernel verwendet werden, spielt keine Rolle, solange die gewählte Methode einheitlich umgesetzt wird. (Vgl. Passig/Jander (2014), S. 23ff.)

3.3 Namensgebung

3.3.1 Anforderungen an Namen

Verständlich Ein guter Bezeichner sollte ohne Kommentar verständlich sein und die Nachvollziehbarkeit des Quelltextes ohne zusätzliches implizites Wissen ermöglichen. Deshalb macht es Sinn, Aufwand in die Findung eines passenden Namens zu investieren und so Zeit bei eventuell nötigen Umbenennungen zu sparen. (Vgl. Martin (2009), S. 18)

Um einen passenden Namen zu finden, sollte man sich die Frage nach der Funktion oder dem Zweck eines Objektes beantworten. Dies hilft gleichzeitig, sich klarzumachen, was man tun möchte. Wenn es schnell gehen muss, um einen wichtigen Gedanken nicht zu verlieren, sind auch unzulängliche Namen erlaubt, solange diese später angepasst werden. (Vgl. Passig/Jander (2014), S. 33f.)

Namen mit nur einem Buchstaben sollten nur bei Schleifenzählern (`i`, `j`, `k`) oder Koordinaten (`x`, `y`, `z`) zum Einsatz kommen. Ein Vorteil langer Namen gegenüber kurzen ist, dass Tippfehler schnell erkannt und behoben werden können, Fehler durch falsches Verständnis jedoch nicht. Erstere können bereits durch den Einsatz von Werkzeugen zur Code-Completion vermieden werden. (Vgl. Passig/Jander (2014), S. 35f.)

Unterscheidbar Um eine einfache Differenzierung von ähnlichen Variablen zu gewährleisten, sollten sich Namen an mehr als einer Stelle unterscheiden. Dabei sollte auf ähnlich aussehende Zeichen verzichtet werden (siehe Quelltext 9). (Vgl. Martin (2009), S. 19f.)

```
1 //Negativbeispiele
2 string anima1;
3 string animal;
4
5 getUsername();
6 getUsernames();
7
8 string carBig;
9 string carSmall;
10
11 //Positivbeispiele
12 string animation_eins;
13 string animal;
14
15 getUsername();
16 getAllUsernames();
17
18 string bigCar;
19 string smallCar;
```

Quelltext 9: Unterscheidung von ähnlich aussehenden Namen

Außerdem sollte laut Kathrin Passig und Johannes Jander der entscheidende Unterschied am Anfang der Variablen stehen (siehe Quelltext 9). (Vgl. Passig/Jander (2014), S. 36) Für die Code-Completion ist es allerdings sinnvoller, wenn der Überbegriff am Anfang steht und anschließend das Detail ausgewählt werden kann.

Des Weiteren sollten keine Füllwörter und nichtssagende Ausdrücke wie `Info` oder `Data` verwendet werden. Eine Nummerierung von Variablen trägt ebenfalls nicht zur Unterscheidung bei. (Vgl. Martin (2009), S. 20f.)

Einheitlich Es gibt verschiedene Möglichkeiten, nach denen Objekte benannt werden können. Welche Methode letztendlich verwendet wird ist weniger entscheidend als diese auch konsequent einzusetzen. Grundsätzlich sollte allerdings alles getrennt werden, was als eigenes Wort verstanden werden kann (siehe Quelltext 10). (Vgl. Passig/Jander (2014), S. 30)

```
1 //Variante 1 (empfohlen)
2 int carSpeed;
3
4 //Variante 2
5 int car_speed;
```

Quelltext 10: Varianten zur Benennung von Objekten

Sprechbar Namen sollte man aussprechen können. Auch, wenn dies meist nur im Kopf geschieht. Beim Programmieren handelt es sich um eine soziale Tätigkeit, da der Quelltext mit anderen diskutiert wird. Unaussprechliche Namen sind dafür hinderlich. (Vgl. Martin (2009), S. 21f.)

Suchbar Für das Benutzen der Suchfunktion ist es hilfreich, wenn eindeutige und nicht zu kurze Bezeichner gewählt werden. Besonders der Einsatz von Konstanten kann dabei helfen. So ist es einfacher den Namen `WORK_DAYS_PER_WEEK` als die Zahl `5` im Quelltext zu finden. (Vgl. Martin (2009), S. 22f.)

Direkt Namen sollten genau das Ausdrücken, was sie meinen. Humor, Niedlichkeit, Originalität, Metaphern oder Wortspiele erfordern implizites Wissen über den Kontext und eignen sich daher nicht für den Austausch unter mehreren Programmierern. (Vgl. Martin (2009), S. 26ff.)

3.3.2 Variablen

Sollte es für eine Variable wichtig sein, dass dem Programmierer bei ihrer Verwendung Zusatzinformationen bekannt sind, macht es Sinn, diese in den Namen aufzunehmen. Die ist zum Beispiel bei physikalischen Einheiten der Fall. (Vgl. Passig/Jander (2014), S. 38)
Bei der Bezeichnung von Arrays stellt sich die Frage, ob der Name im Singular oder Plural geschrieben werden soll. Für beide Varianten gibt es Argumente. Bei der Deklaration ist der Plural lesbarer, während die Verwendung des Singular beim Aufruf eines Elementes sinnvoller erscheint (siehe Quelltext 11). (Vgl. Passig/Jander (2014), S. 31)

```
1 //Variante 1 (empfohlen)
2 string[ ] colour = {red, green, blue};
3 paintWall(colour[1]);
4
5 //Variante 2
6 string[ ] colours = {red, green, blue};
7 paintWall(colours[1]);
```

Quelltext 11: Varianten zur Benennung von Arrays

Boolesche Variablen lassen sich am besten mit dem englischen Wort `is` kombinieren. Damit wird eine natürliche Ausdrucksweise wie zum Beispiel bei `isEnabled` oder `isValid` erreicht. Es ist darauf zu achten, keine doppelte Verneinung wie bei `isDisabled = false` zu verwenden.

Darüber hinaus hat sich der Begriff `toogle` etabliert, um unabhängig vom Ausgangszustand von einem Zustand zum anderen zu wechseln. (Vgl. Passig/Jander (2014), S. 55ff.)
Namen von Konstanten werden in nahezu allen Programmiersprachen in Großbuchstaben geschrieben. (Vgl. Passig/Jander (2014), S. 38)

3.3.3 Funktionen

Funktionsnamen sollten beschreiben was eine Funktion tut und nicht wie sie es tut, da sich dieser Teil ändern kann. Wie das Beispiel `setCounter` zeigt sollte ihr Name mit einem Verb beginnen an das sich das behandelte Objekt anschließt. (Vgl. Passig/Jander (2014), S. 37ff.)
Zu lange Funktionsnamen deuten darauf hin, dass die Funktion besser in weitere Unterfunktionen aufgeteilt werden sollte. Es sollte nicht versucht werden einen schöneren, abstrakteren oder einfacheren Namen zu finden, ohne den Inhalt der Funktion entsprechend anzupassen.
Zu vage Funktionsnamen wie `eintragBearbeiten` sind zu vermeiden. Spätestens, wenn eine Funktion mit dem Namen `eintragVerarbeiten` aufgerufen wird, ist der Inhalt einer dieser Funktionen nicht mehr zu erkennen. (Vgl. Passig/Jander (2014), S. 33f.)

3.3.4 Objektorientierte Programmierung

Die Objektorientierte Programmierung (OOP) verwendet besondere Elemente wie Klassen und Methoden, deren Verwendung ebenfalls spezielle Regeln erfordert.
Klassen sollten immer mit Substantiven bezeichnet werden. Dabei hilft es, an Berufe wie zum Beispiel `Worker` oder `Writer` und Geräte wie zum Beispiel `Converter` oder `Adapter` zu denken. Zu vage Namen wie `Manager` oder `Handler` geben keinen Hinweis darauf, welche konkrete Funktion die Klasse erfüllt. In den Namen sollte nicht mit aufgenommen werden, dass es sich um eine Klasse handelt.
Bei der Benennung von Interfaces macht es Sinn, von Verben abgeleitete Adjektive einzusetzen, da alle Objekte, die ein Interface implementieren, bestimmte Handlungen durchführen können. Beispiele dafür sind `Comparable` oder `Runnable`. (Vgl. Passig/Jander (2014), S. 52f.)
Methodennamen sollten ein Verb enthalten. Dabei ist das Objekt, wie es bei Funktionen der Fall ist, nicht mit in den Namen aufzunehmen, da der Zusammenhang automatisch beim Aufruf hergestellt wird (siehe Quelltext 12). (Vgl. Passig/Jander (2014), S. 41)

```
1 //Negativbeispiel
2 Abbreviation.expandAbbreviation();
3
4 //Positivbeispiel
5 Abbreviation.expand();
```

Quelltext 12: Methodenaufruf in der OOP

3.3.5 Datenbanken

Da die Structured Query Language (SQL) vor der Zeit des Internets entstanden ist, gibt es viele unterschiedliche Konventionen aber auch Diskussionen, um einen Standard zu erreichen. Es gibt jedoch einige Empfehlungen, die Sinn machen.
Tabellen- und Spaltennamen sollten ohne Leerzeichen geschrieben werden. Dabei sollten Dinge, die mehrfach vorkommen auch gleich bezeichnet werden. So ist zum Beispiel die einheitliche Verwendung von `PLZ` gegenüber der Nennung von `PLZ`, `Postleitzahl` und `ZipCode` vorzuziehen.
Tabellen sollten immer nach ihrem Inhalt bezeichnet werden. Dabei macht es grundsätzlich Sinn den Plural zu verwenden. Englische Begriffe wie `Information` oder `Furniture`, die keinen Plural besitzen, sollten dagegen gemieden werden. (Vgl. Passig/Jander (2014), S. 53ff.)
Spaltennamen sollten ebenfalls beschreiben, was sie beinhalten und nicht etwa `leftColumn` oder `rightColumn` heißen, da sich die Position der Spalte innerhalb der Tabelle ändern kann. (Vgl. Passig/Jander (2014), S. 37)
Die Primär- und Sekundärschlüssel sollten immer `ID` am Ende des Namens tragen. Der Primärschlüssel wird dabei aus dem eigenen Tabellennamen gebildet während ein Sekundärschlüssel den Namen der ursprünglichen Tabelle trägt.
Spalten, die ein Datum oder eine Zeit enthalten, sollte statt einer generischen Bezeichnung eine konkrete Bedeutung gegeben werden (siehe Quelltext 13). (Vgl. Passig/Jander (2014), S. 53ff.)

```
1 //Negativbeispiel
2 date DATE
3 time TIME
4
5 //Positivbeispiel
6 createdDate DATE
7 insertTime TIME
```

Quelltext 13: Verwendung von `DATE` und `TIME` in SQL

3.3.6 Namensänderungen

Wenn sich die Anforderungen an eine Variable ändern oder eine Funktion angepasst wird, kann es sein, dass der gewählte Name den Inhalt nicht mehr ausreichend repräsentiert. Der nötige Änderungsaufwand lohnt sich, da sich dadurch die Codequalität erhöht.
Dabei sollte auf keinen Fall die Funktion „Suchen und Ersetzen“ verwendet werden, da sonst auch Stellen, die nur die gleiche Zeichenfolge enthalten ohne die Variable zu meinen, ersetzt werden. Deshalb gibt es spezielle Refactoring-Werkzeuge, die Namensänderungen auch über ganze Projekte hinweg durchführen können. Grundsätzlich sollte das Programm danach auf seine Lauffähigkeit überprüft werden. (Vgl. Passig/Jander (2014), S. 31f.)

3.4 Kommentare

Quelltext wird nicht nur für Maschinen sondern auch für Menschen geschrieben. Da sich aus ihm nicht immer sofort alle Informationen erschließen, benötigt man erläuternde Kommentare. Diese sollten den Quelltext nicht eins zu eins wiedergeben, sondern den Gesamtkontext erklären und über Ideen, Gründe und Absichten aufklären. (Vgl. Passig/Jander (2014), S. 62ff.)
Es lohnt sich, Aufwand in knappe und treffsichere Kommentare zu investieren, die das Gedächtnis und den Denkapparat des Lesers entlasten. So kann man beispielsweise Module mit Kurzbeschreibungen versehen oder die Bedeutung von Variablen erläutern. (Vgl. Grams (1990), S. 82)
Kommentare sollten direkt zusammen mit dem Quelltextes und nicht erst zeitlich am Ende eingefügt werden. Noch hilfreicher ist es, sie vorher zu schreiben, um sich klarzumachen, was man tun möchte. Dabei geht es darum, warum der Quelltext etwas tut und nicht was der Quelltext tut. (Vgl. Passig/Jander (2014), S. 67f.)
Grundsätzlich stellen Kommentare allerdings eine Redundanz zum Quelltext dar, da sie nur das ausdrücken, was er ohnehin sagt. Deshalb sollte man nicht jede einzelne Zeile kommentieren, sondern mit wenig Kommentaren die Verbindung zwischen der Aufgabenstellung und dem Programm herstellen (siehe Quelltext 14). (Vgl. Grams (1990), S. 85)
Aufgrund der Redundanz zwischen Quelltext und Kommentaren können diese falsche Informationen enthalten, wenn nur der Quelltext geändert wurde. Deshalb sollte die Wahrheit immer im Code gesucht werden. (Vgl. Martin (2009), S. 54)

```
1 //Negativbeispiel
2   tmp = zahl1; //Wert von zahl1 in tmp speichern
3   zahl1 = zahl2; //Wert von zahl2 in zahl1 speichern
4   zahl2 = tmp; //Wert von tmp in zahl2 speichern
5
6 //Positivbeispiel
7   //Zahlen tauschen
8   tmp = zahl1;
9   zahl1 = zahl2;
10  zahl2 = tmp;
```

Quelltext 14: Verwendung von Kommentaren

Gute Kommentare liefern zusätzliche Informationen über Dokumente, Copyrights oder Autorennamen. Sie beschreiben Absichten und bestehende Probleme oder erklären komplizierte und unklare Ausdrücke. Außerdem können sie Warnungen geben, auf Einschränkungen hinweisen oder zukünftige Pläne beschreiben. (Vgl. Martin (2009), S. 55ff.)
Schlechte Kommentare dagegen entstehen, wenn sie nur deshalb eingefügt werden, weil der Prozess sie vorschreibt. Negativ sind außerdem irreführende Kommentare oder Kommentare, die Abkürzungen enthalten und selbst Erläuterungen benötigen. Eine früher in Kommentaren gepflegte Änderungshistorie ist heute in Zeiten von Versionskontrollsystemen nicht mehr sinnvoll. Auskommentierter Quelltext im Endprodukt sollte ebenfalls vermieden werden. (Vgl. Martin (2009), S. 59ff.)
Viele Kommentare sind ein Zeichen für schlechten Quelltext. Ein gut geschriebenes Programm macht Kommentare überflüssig. Fügt man beispielsweise einen Kommentar ein, der erklärt, was eine Methode tut, sollte besser die Methode umbenannt werden. (Vgl. Fowler (2005), S. 82)

4 Einhaltung von Programmierkonventionen

Sich auf bestimmte Programmierkonventionen zu einigen ist in der Praxis nicht ausreichend. Vielmehr sollten Methoden und Werkzeuge zum Einsatz kommen, um die Einhaltung der Regeln sicherzustellen. (Vgl. Martin (2009), S. 76)
Zur Überprüfung der in den Notationskonventionen festgelegten Syntax können Werkzeuge eingesetzt werden, die häufig bereits in die Entwicklungsumgebung integriert sind. Nach einmaliger Anpassung an die festgelegten Regeln kann die Überprüfung des Quelltextes automatisch erfolgen. Dabei können mit geringem Aufwand beispielsweise Formatierungen angepasst oder verbotene Anweisungen identifiziert werden. (Vgl. Liggesmeyer (2001), S. 252)
Ein weiterer Grund für den geringen Aufwand bei der automatischen Überprüfung des Quelltextes ist, dass das Programm nicht extra compiliert und ausgeführt werden muss. (Vgl. Liggesmeyer (2001), S. 249)
Vorsicht ist bei Werkzeugen für automatische Quelltextkorrekturen geboten. Zum einen erkennen Versionskontrollsysteme nicht immer den Unterschied und zum anderen können sich Formatierungsregeln im Laufe des Projektes ändern. (Vgl. Passig/Jander (2014), S. 26f.)
Die Überprüfung von Sprachkonventionen zur Semantik kann nur von Menschen durchgeführt werden, da sie eine Interpretation erfordern. Ein automatisches Werkzeug kann zum Beispiel nicht entscheiden, ob der Name einer Variable aussagekräftig ist oder nicht. Deshalb müssen zeit- und kostenintensive Prüfungen der Semantik manuell durchgeführt werden. Eine Variante sind regelmäßige Code-Reviews mit definierten Regeln oder Checklisten. (Vgl. Liggesmeyer (2001), S. 252)
Code Reviews haben außerdem den Vorteil, dass durch sie das Wissen im Entwicklerteam weitergegeben und verteilt wird. Die gemeinsame Überprüfung sorgt für Klarheit über den Quelltext und gibt anderen die Gelegenheit, Vorschläge und Lösungsmöglichkeiten zu äußern. Schon bei einem Team von zwei Entwicklern werden diese Vorteile spürbar. Den Extremfall dieser Variante stellt die agile Softwareentwicklungsmethode Extreme Programming dar. (Vgl. Fowler (2005), S. 47)

5 Programmierkonventionen im Team

Während Programmieren ursprünglich eine Einzeltätigkeit war, kommt heute kaum ein Softwareprojekt ohne Arbeitsteilung aus. Ein Grund dafür ist die zunehmende Größe und Komplexität der zu entwickelnden Software, die alleine nicht zu bewältigen ist. Der Einsatz von Teams erfordert deshalb Absprachen innerhalb der Gruppe.
Außerdem sorgt die zunehmende Mitarbeiterfluktuation für häufig wechselnde Teammitglieder. Die Abwanderung von Wissen muss verhindert und das einfache und schnelle Einarbeiten neuer Mitarbeiter ermöglicht werden. (Vgl. Hoffmann (2013), S. 14f.)
Das grundsätzliche Problem dabei ist, dass sich jeder Entwickler seinen eigenen Programmierstil angeeignet hat. Es spielt keine Rolle, wie programmiert wird, beim Aufeinandertreffen mehrerer Philosophien gibt es Probleme, da Software immer für mehrere Leser geschrieben wird. (Vgl. Passig/Jander (2014), S. 23)
Bei einer Teamneugründung ist es deshalb unabdingbar, vor der Zusammenarbeit einen Programmierstil festzulegen. Anschließend sollten alle Teammitglieder diesen Regeln zustimmen und sich auch daran halten. (Vgl. Martin (2009), S. 76)
Sich auf einen Stil zu einigen sorgt für konsistente Quelltexte, was wiederum die Verständlichkeit erhöht. Der Leser erwartet, dass gleiche Dinge auch das gleiche bedeuten. Dabei sollte es keine Rolle spielen, von wem der jeweiligen Quelltext geschrieben wurde. Ansonsten wird die Komplexität unnötig erhöht. (Vgl. Martin (2009), S. 90)
Tritt man einem bereits bestehenden Team bei, sollte man zunächst beobachten und sich dem geltenden oder etablierten Programmierstil anpassen. Die Vorteile der Vorgaben sind in jedem Fall größer als der Umgewöhnungsaufwand. Diskussionen über richtige und falsche Programmierkonventionen sind unproduktiv und zeitraubend, da sich Argumente für jeden Ansatz finden lassen. Bei Problemen sollte entweder nachgegeben oder auf die Zusammenarbeit verzichtet werden. (Vgl. Passig/Jander (2014), S. 26f.)
Die Erfahrung zeigt, dass besonders Anfänger Konventionen ignorieren. Dies geschieht nicht aus Überzeugung sondern aus Unkenntnis über deren Relevanz. Um etwas optimieren zu können sollte das Alte jedoch zunächst verstanden werden. Wird die eigene Idee weiterhin favorisiert, wird dies zu Problemen in der Zusammenarbeit führen. (Vgl. Passig/Jander (2014), S. 19f.)
Auf keinen Fall sollte fremder Quelltext zeitaufwändig umformatiert werden. Sollte man sich dennoch für die Umbenennung eines Objektes entscheiden, muss dies immer mit dem Team abgesprochen werden. Man könnte den Sinn und Zweck der Variable falsch interpretiert haben oder sie könnte an anderer Stelle wie zum Beispiel in der Dokumentation erwähnt werden. (Vgl. Passig/Jander (2014), S. 31f.)

6 Fazit

Programmierkonventionen sind ein einfaches aber wirksames Mittel, um die Softwarequalität zu erhöhen. Besonders in der Zusammenarbeit unter mehreren Programmierern sind sie unabdingbar. Hat man sich auf einen Stil geeinigt, sollten automatische Werkzeuge und Code-Reviews zum Einsatz kommen, um die Einhaltung sicherzustellen.
Dabei sollten bewährte und allgemein bekannte Methoden verwendet werden, damit sich neue Teammitglieder schnell zurechtfinden. Wird in einem Projekt zum Beispiel in der Programmiersprache Java entwickelt, macht es Sinn, die eigenen Programmierkonventionen an die offiziellen Java Code Conventions anzulehnen. Die darin genannten Ansätze sind bereits in unterschiedlichen Umgebungen erprobt. Außerdem ist der Quelltext für Leser einfacher zugänglich und man kann bei Problemen mit schnellerer Hilfe rechnen.
Allerdings sollten laut dem deutschen Informatiker Peter Liggesmeyer die Inhalte von im Internet verfügbaren Programmierkonventionen grundsätzlich kritisch betrachtet werden, da sie einer näheren Analyse meist nicht standhalten. (Vgl. Liggesmeyer (2001), S. 252)
Werden die einmal festgelegten Konventionen konsequent umgesetzt, kann viel Zeit bei der späteren Fehlersuche und Fehlerbehebung eingespart werden. Dies steigert die Softwarequalität und verringert die Entwicklungszeit. Bei richtiger Anwendung sind Programmierkonventionen damit ein entscheidender Faktor für den Erfolg eines Softwareprojektes.

Literaturverzeichnis

Fowler, M. (2005): *Refactoring – Wie Sie das Design vorhandener Software verbessern*, Addison-Wesley Verlag, München.

Google, Inc. (2014): *Google Java Style*, URL: `http://google-styleguide.googlecode.com/svn/trunk/javaguide.html` (besucht am 13. 05. 2015).

Grams, T. (1990): *Denkfallen und Programmierfehler*, Springer-Verlag, Berlin.

Hoffmann, D. W. (2013): *Software-Qualität*, 2. aktualisierte und korrigierte Auflage, Springer Vieweg, Berlin u.a.

Liggesmeyer, P. (2001): *Software-Qualität: Testen, Analysieren und Verifizieren von Software*, Spektrum Akademischer Verlag, Heidelberg u.a.

Martin, R. C. (2009): *Clean Code – A Handbook of Agile Software Craftsmanship*, Pearson Education, Inc., Boston.

Passig, K.; Jander, J. (2014): *Weniger schlecht programmieren*, 1. korrigierter Nachdruck, O'Reilly Verlag, Köln.

Schneider, K. (2012): *Abenteuer Softwarequalität: Grundlagen und Verfahren für Qualitätssicherung und Qualitätsmanagement*, 2. überarbeitete und erweiterte Auflage, dpunkt.verlag, Heidelberg.

Sun Microsystems, Inc. (1997): *Java Code Conventions*, URL: `http://www.oracle.com/technetwork/java/codeconventions-150003.pdf` (besucht am 13. 05. 2015).

B.1.10. Softwareentwicklungs-Vorgehensmodell

Thema:

Softwareentwicklungs-Vorgehensmodell

Seminararbeit
im Rahmen der Projektgruppe „Innovation Management Platform to Activate Creative Thoughts"

Abteilung Wirtschaftsinformatik 1:
Very Large Business Applications

Themensteller: Prof. Dr.-Ing. Jorge Marx Gómez
Betreuer: Stefan Wunderlich M.Sc.,
Dipl.-MA. Jens Siewert,
Dr. Joachim Kurzhöfer

vorgelegt von: Jana-Vanessa Dering
Ingelheimer Straße 1
28199 Bremen

E-Mail: vanessa.dering@uni-oldenburg.de

Abgabetermin: 22. Juni 2015

Inhaltsverzeichnis

Verzeichnis der Abkürzungen und Akronyme

IMPACT	Innovation Management Platform to Activate Creative Thoughts
SiP	Story in Progress
WiP	Work in Progress

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Im Folgenden werden die Motivation, Vorgehensweise und das Ziel der Seminararbeit erläutert.

1.1 Motivation

Um eine Software zu entwickeln, bedarf es ein großes Maß an fachlicher Kompetenz und methodischer Kompetenz. Bei der Auswahl, Einführung und Nutzung der Methode kommt es auf die Zusammenarbeit der beiden Kompetenzen an. Es müssen verschiedene Aufgaben zur Entwicklung erstellt, koordiniert, geleitet und bearbeitet werden. Um diese Entwicklung effizient zu gestalten ist es notwendig, verschiedene Prozessschritte zu erstellen um die Komplexität einer Software zu verringern. Durch die Einteilung ist es möglich, komplexe Aufgaben aufzuteilen und koordiniert zu bearbeiten. Die Softwareentwicklungsmethode bietet einen vorgegeben Rahmen um die Unterteilung in Prozessschritt zu erleichtern, eine klare Verantwortungsstruktur und Koordination über die Entwicklung zu gewährleisten. Mit Hilfe dieser Methode sollen grundlegende Regeln zum Ablauf der Entwicklung erstellt und eingehalten werden. Um die Kreativität innerhalb der Entwicklung und Flexibilität zu fördern, werden agile Vorgehensmodelle gewählt. Diese geben allen Beteiligten einen gewissen Freiraum zum kreativen Arbeiten. Hier ist jedoch zu beachten, dass der Verwaltungsaspekt in den Hintergrund gerät. Innerhalb dieser Seminararbeit werden nur lediglich Vorgehensmodelle vorgestellt.

Innerhalb des Rahmens der Projektgruppe „IMPACT“ in der VLBA Abteilung der Carl-von-Ossietzky Universität Oldenburg soll ein Projekt, welches sich über zwei Semester erstreckt, zum Ziel haben, eine Innovationsmanagementplattform für Lufthansa Industry Solutions Oldenburg zu entwickeln und bereitzustellen

1.2 Ziel und Vorgehensweise

Ziel dieser Seminararbeit ist es, verschiedene agile Vorgehensmodelle vorzustellen und die Anpassungen der Methode Scrum an die Projektgruppe „IMPACT“ zu erläutern.

Im Kapitel 2 werden zunächst grundlegende Informationen zur agilen Softwareentwicklung gegeben. Anschließend werden die Methoden Scrum, Kanban, Scrumban vorgestellt und deren Unterschiede herausgearbeitet. Die Entscheidung der Projektgruppe „IMPACT“ wird kurz erläutert um darauf folgen, die ausgewählte Methode Scrum im Kapitel 3 genauer zu erklären. Ergänzend wird die Anpassung an die Projektgruppe aufgeführt. Im Kapitel Fazit/ Ausblick werden zusätzliche Vorschläge an die Projektgruppe unterbreitet.

2 Agile Methoden

In diesem Kapitel werden zunächst Grundlagen vermittelt und anschließend drei verschiedene agile Methoden vorgestellt.

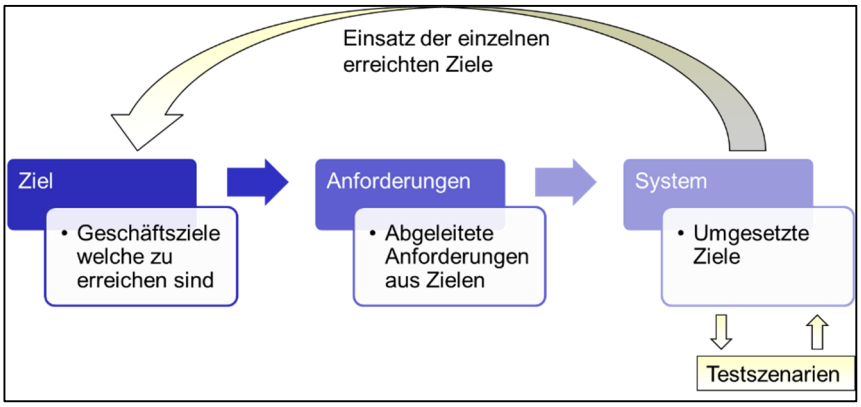
2.1 Grundlagen

In der agilen Softwareentwicklung ist das Ziel, schnell vorzeigbare Ergebnisse zu erzielen, welche an den Kunden angepasst sind (Vgl. Bleek, W.-G., Wolf, H. (2010), S.7). Dabei wird besonders Wert auf die Kommunikation mit dem Auftraggeber und –nehmer und dem Team, die schnelle Reaktion auf den Kunden, die Einfachheit der Technik und das Einhalten der Methodik gelegt. Diese Grundlagen wurden im agilen Manifest festgehalten.

„… ***Individuen und Interaktionen*** *mehr als Prozesse und Werkzeuge.* ***Funktionierende Software*** *mehr als umfassende Dokumentation.* ***Zusammenarbeit mit dem Kunden*** *mehr als Vertragsverhandlung.* ***Reagieren auf Veränderung*** *mehr als das Befolgen eines Plans…“*(Agilemanifesto (2001)). In diesem Auszug des Manifests wird verdeutlicht, dass es sich um die Kommunikation, den Kunden und um ein schnelles Ergebnis dreht. Jedoch sind die anderen Merkmale wie „Prozesse und Werkzeuge“, „Umfassende Dokumentation“ und „Befolgen eines Planes“ ebenfalls relevant und sollten nicht vergessen werden. Hinter dem Manifest stehen 12 Prinzipien, auch „agiles Prinzip“ genannt, die als Leitfaden für die Arbeit mit agilen Methoden aufgefasst werden können. Einige dieser Prinzipien lauten wie folgt:

- Veränderungen nutzen dem Wettbewerbsvorteil des Kunden,
- Zusammenarbeit der Fachexperten und Entwickler,
- Liefere funktionierende Software und
- Technische Exzellenz und gutes Design (Vgl. zu diesem Abschnitt AgilePrinzipien (2001)).

In der agilen Softwareentwicklung geht es um das iterative Vorgehen in der Programmierung, im Team und im Management (Vgl. IT-Agile (o.J.)). Zu Beginn der agilen Methode steht zunächst die Definition von Geschäftszielen, die meistens aus dem Unternehmen heraus oder vom Kunden vorgegeben werden. Daraus werden die Anforderungen an das System abgeleitet und in dem System umgesetzt. Um zu kontrollieren ob diese Anforderungen erfolgreich umgesetzt wurden, werden Testszenarien erarbeitet, die das System durchlaufen muss. Dieses gesamte Vorgehen ist iterativ, d.h. er wird für jedes Ziel erneut durchlaufen (siehe Abbildung 2-1). Dadurch ist es möglich, dass bereits eine Teilfunktionlität des Systems genutzt werden und auf Veränderungswünsche schnell eingegangen werden kann.



Quelle: in Anlehung an [IT-Agile (o.J.)]

Abbildung 2-1 Iterativer Vorgang agiler Methoden

Den erfolgreichen Einsatz dieser Methoden zu gewährleisten, müssen die Geschäftsziele eindeutig formuliert, kommuniziert und priorisiert werden. Diese bilden die Grundlage für die Entwicklung und sind somit für einen Erfolg der Software mitverantwortlich.

Diese allgemeine Vorgehensweise ist zum Teil in abgewandelter Form in den Methoden Extreme Programming (XP), Scrum, Kanban und Scrumban wiederzufinden.

2.2 Scrum

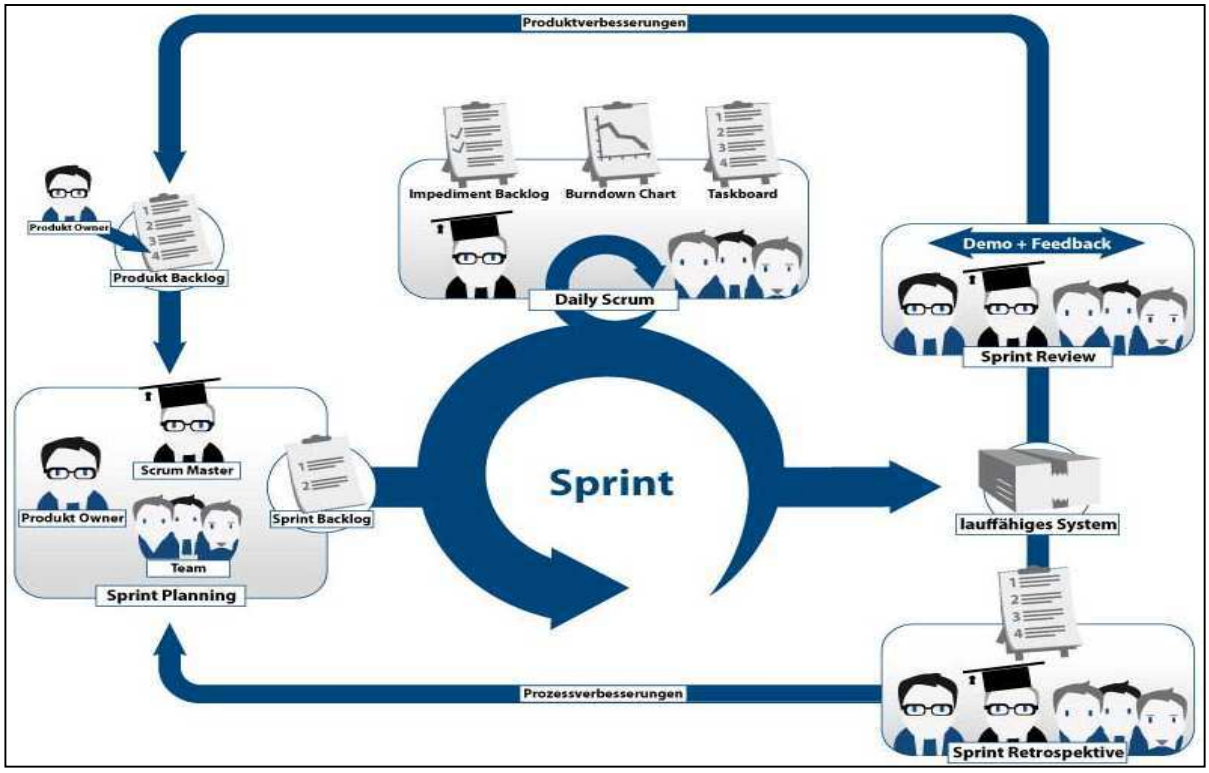
Scrum basiert auf der allgemeinen agilen Vorgehensweise und legt dabei seinen Schwerpunkt auf die Kommunikation und Selbstorganisation des Scrum-Teams. Es ist ein Framework, welches aus Rollen, Meetings und Artefakten besteht und alle Aktivitäten der Produktentwicklung abläuft (Vgl. B. Gloger (2013), S.9).

Scrum legt Wert auf eine klare Rollenverteilung innerhalb des Prozesses. Rollen, die in jedem Projekt, welches mit Scrum als Softwareentwicklungsmethode arbeitet, belegt sein müssen, sind der Product Owner, das Entwicklungsteam und der Scrum Master. Darüber hinaus existieren weitere Rollen wie Manager, Anwender und Kunde. Diese Rollen können sich aber in Projekten überschneiden oder fehlen. Der *Product Owner* wird vom Kunden oder aus dem eigenen Unternehmen gestellt und vertritt die Wünsche des Kunden gegenüber dem Entwicklerteam. Er ist verantwortlich dafür, dass das Team die Funktionalitäten in richtiger Reihenfolge entwickelt und arbeitet kontinuierlich an dem Product Backlog (Vgl. B. Gloger (2013), S.12). Das *Entwicklungsteam* entwickelt das Produkt und ist selbstständig organisiert. Es besteht meistens aus 5-10 Mitgliedern. Der *Scrum Master* ist verantwortlich für die Einhaltung des Scrum-Prozesses und greift nicht direkt in das Geschehen ein. Er ist ebenfalls dafür zuständig, Probleme zu beseitigen, die das Team aufhalten könnte(Vgl. B. Gloger (2013), S.12).

Artefakte, die innerhalb von Scrum erstellt und bearbeitet werden, sind zum einen das *Product Backlog*, das eine priorisierte Liste der zu liefernden Funktionalitäten darstellt. Diese werden auch *Product Backlog Items* genannt und in *User Stories*[1] formuliert. Oftmals werden Product Backlog Items und Anforderungen synonym verwendet. Es muss aber darauf hingewiesen werden, dass Anforderungen mit vielen Informationen, z.B. der Entwicklung angereichert sind. Diese Informationen werden erst in dem Sprint Planning und weiteren Meetings hinzugefügt (Vgl. zu diesem Abschnitt B. Gloger (2013), S.81).

Der *Sprint Backlog* enthält alle Items aus dem Product Backlog, welche innerhalb eines Sprints bearbeitet werden sollen. Diese werden vom Entwicklerteam mit konkreten Entwicklungstätigkeiten ergänzt. Das *Impediment Backlog* ist ebenfalls eine Liste, welche alle Blocker (Impediment), die das Team an einer effektiven Arbeitsweise hindern, beinhaltet. Es muss dann versucht werden, diese Blocker zu beseitigen. Wenn diese prozessual bedingt sind, ist der Scrum Master für die Beseitigung dieser Hindernisse verantwortlich. Ein weiteres Artefakt ist das *Produkt-Inkrement*, welches als eine Art Teilsystem aufgefasst werden kann. Es beinhaltet funktionsfähige Funktionen, die dem Kunden präsentiert und ausgeliefert werden könnten. Innerhalb des Prozesses finden vor, nach und während eines Sprints Meetings statt, die dem gesamten Team zum Fortschritt der Projektes verhelfen sollen.

In der Abbildung 2-2 ist zu erkennen, dass zum Beginn eines jenen Sprints das *Sprint Planning* steht. In diesem Meeting wird das Ziel eines Sprints definiert und die dazu passenden Product Backlog Items ausgewählt und besprochen, wie diese zu erreichen sind. Dazu zählen Auswahl der Architektur der



Quelle:[SCRUM Kompakt (2015)]

Abbildung 2-2 Ablauf eines Scrum-Prozesses

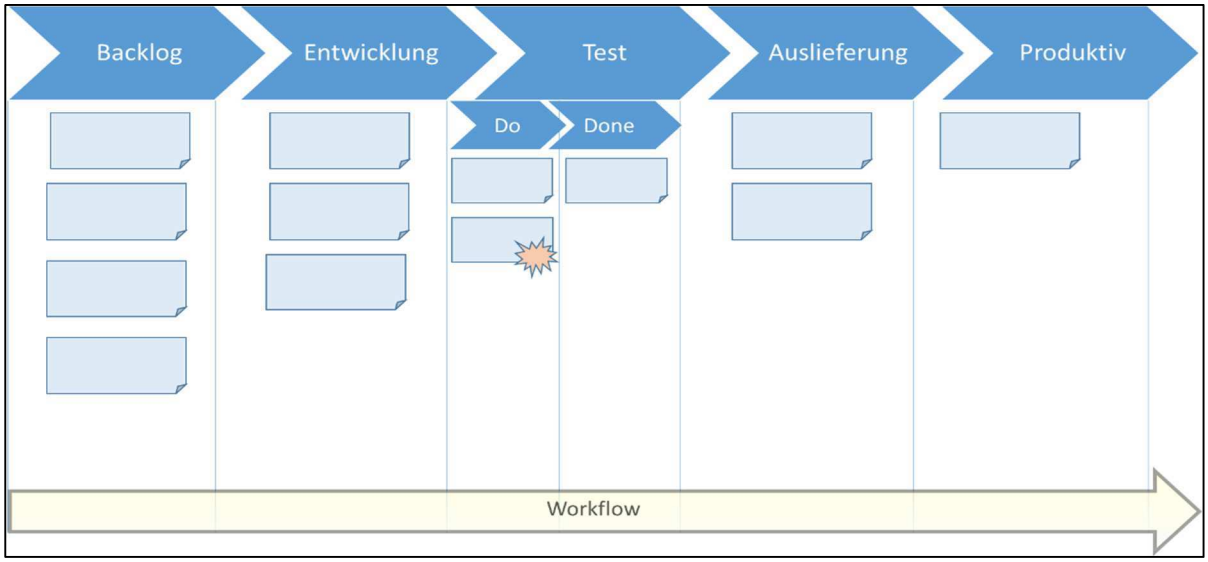
Applikation, Test Cases und Interfaces (Vgl. zu diesem Abschnitt B. Gloger (2013), S.13).
Wenn alle notwendigen Aufgaben notiert sind, werden diese in das Sprint Backlog überführt. Anwesend ist das gesamte Scrum-Team. Danach startet der zeitlich begrenzte Sprint, innerhalb dessen täglich das *Daily Scrum* stattfindet, bei dem jedes Mitglied des Entwicklerteams kurz erklärt, welche Aufgabe erledigt ist, aufgenommen wird oder Probleme bereitet. Geleitet wird das Meeting vom Scrum Master. Ist der Sprint nach der vorgegeben Zeit erledigt und ein lauffähiges System bereitgestellt (siehe Abbildung 2-2) findet zum einen das *Sprint Review* und zum anderen die *Sprint Retrospektive* statt. In dem Sprint Review werden die erstellten Funktionalitäten präsentiert und besprochen. Es ist möglich ein Feedback zu den lauffähigen Funktionalitäten zu geben und damit die Produktqualität zu erhöhen. Eventuelle daraus entstandene Anforderungen können in das Product Backlog als Product Backlog Items eingefügt werden. Die Sprint Retrospektive setzt auf die Prozessverbesserung. Hierbei werden Arbeitsabläufe analysiert und festgestellt, welche sich ändern müssen damit das Entwicklerteam produktiver arbeiten kann. Die Ergebnisse werden in dem Impediment Backlog festgehalten und in dem nächsten Sprint Planning als Verbesserungsvorschläge eingebracht (Vgl. zu diesem Abschnitt B. Gloger (2013), S.13).

2.3 Kanban

Kanban ist ebenfalls eine agile Softwareentwicklungsmethode, welche sich aber klar von dem Scrum-Prozess abgrenzt. Es ist aus dem Toyota-Produktionssystem heraus entstanden und ist stark von der Just-in-time Produktion beeinflusst. David J. Anderson entwickelte Grundprinzipien für Kanban, die wie folgt lauten: *„Dort beginnen, wo man sich gerade befindet. Inkrementelle, evolutionäre Veränderungen anstreben. Auf bestehenden Rollen, Abläufen und Prozessen aufsetzen und diese respektieren. Leadership auf allen Ebenen in der Organisation fördern.“* (Kieninger, *T.* (o.J.)). Daraus folgt, dass Kanban keine eigenen definierten Rollen, Abläufe und Prozesse einführt, sondern sich an das Betriebsumfeld und dessen Strukturen anpasst. Ebenso soll das Führen unter visionären und kreativen Bedingungen gefördert werden. Es ist ein stetiger Prozess und setzt nicht auf iterative Vorgänge wie Scrum.

Virtualisierung des Workflow, Limitierung des Work in Progress (WiP), Steuerung und Messen, Prozessregeln und *kontinuierliche Verbesserung (Kaizen)* sind die umzusetzenden Schritte innerhalb Kanbans. Zu Beginn der Methode wird der Workflow visualisiert. Zunächst müssen Prozessschritte ermittelt und voneinander abgegrenzt werden. Anschließend an dem *Kanban-Board* visualisiert werden (siehe Abbildung 2-3). Aufgaben (oder auch Tickets genannt) werden mittels Karten dargestellt und an dem Board befestigt. Diese wandern dann im Laufe der Zeit in Richtung des Workflows, über das Board. Dabei wird das *Pull-Prinzip* angewendet, das bedeutet, es können keine Tickets von einem Prozessschritt in den folgenden geschoben, sondern nur vom folgenden gezogen werden. Zusätzlich ist es möglich blockierte Tickets zu markieren (siehe Abbildung 2-3 mit Stern). Die einzelnen Prozessschritte sollten zusätzlich unterteilt werden, z.B. in die Phasen „Do“ und „Done“

(siehe Abbildung 2-3). Hierbei ist zu beachten, dass genau definiert wird, wann ein Item bearbeitet und wann es fertiggestellt ist. Durch diese Unterteilung ist es einfacher den Status der Tickets zu verfolgen.



Quelle: in Anlehnung an [Leopold, K., Kaltenecker, S. (2012), S.25]

Abbildung 2-3 Kanban-Board

Durch die Visualisierung ist es möglich, Engpässe innerhalb des Prozesses zu erkennen. Bspw. könnte es zu einem Ticketstau in der Entwicklung kommen, sodass die nachgelagerten Prozessschritte nicht mit Aufgaben versorgt werden. Eine Methode, die diese Problematik vorbeugen soll, ist die Limitierung der *Work in Progress*. Es wird zunächst festgelegt wie viele Tickets innerhalb eines Schrittes bearbeitet werden können. Diese Festlegung ist keineswegs starr, sondern kann an das Team fortwährend angepasst und verändert werden. Bei der Limitierung ist es notwendig zu wissen, wie viele Mitarbeiter innerhalb eines Prozessschrittes arbeiten und wie viele Aufgaben jede Person bearbeiten kann. Generell ist zu erkennen, dass z.B. ein WiP-Limit > 1 pro Person meistens hilfreich ist, wenn es innerhalb des Prozesses zu Blockaden kommt (Vgl. Leopold, K., Kaltenecker, S. (2012), S.45). Wenn es z.B. drei Mitarbeiter gibt und jeder zwei Aufgaben parallel bearbeiten kann, ist es möglich das WiP-Limit für diesen Prozessschritt auf sechs zusetzen. WiP-Limits können ebenfalls als Regler angesehen werden, mit denen man den Durchlauf kontinuierlich verbessern und anpassen kann.

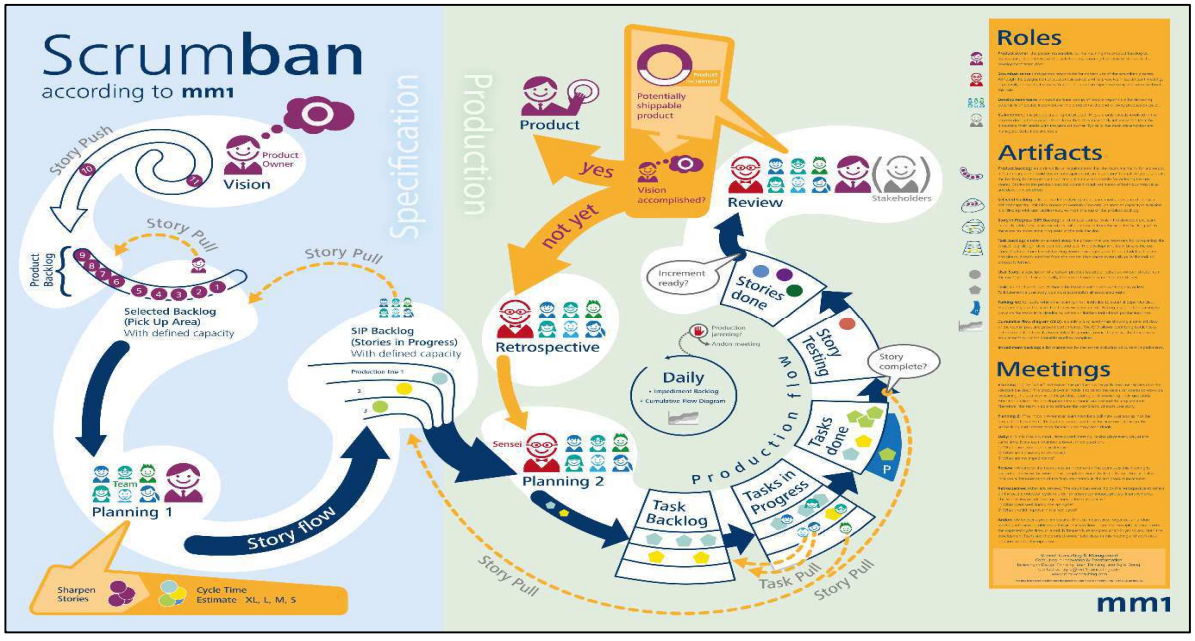
Um Tickets bewerten zu können, werden *Service Klassen* eingeführt. Dadurch können Aufgaben zielgerichtet bearbeitet und fertiggestellt werden. Die Klassen werden anhand des „Cost of Delay“ ermittelt. Es findet eine Staffelung nach Verzögerung und die daraus entstehenden Kosten statt. Weitverbreitete Klassen sind „Beschleunigt“, „Fester Liefertermin“, „Standard“ und „Unbestimmbare Kosten“. Die Auflistung beginnt mit der höchsten bewerteten Klasse („Beschleunigt“) und endet mit der niedrigsten bewerteten Klasse („Unbestimmbare Kosten“) (Vgl. zu diesem Abschnitt Leopold, K., Kaltenecker, S. (2012), S.53).

Es wird anhand der Klassen und WiP-Limits eine Steuerung und Messung des Prozesses gewährleistet. Durch Setzen der Limits und Definieren der Klassen ist es möglich, zum einen den Prozess zu steuern und zum anderen anhand der Durchlaufzeiten und Kosten zu messen, wie produktiv der Prozess durchlaufen wird.

Die kontinuierliche Verbesserung wird zum einen, ähnlich wie in Scrum, mit *Standup Meetings*, *Operation Reviews* und *Retrospektiven* und zum anderen mit den bereits erwähnten Methoden gewährleistet. Ziel ist es den Prozess zu optimieren, sodass es zu keinen Verzögerungen kommt und ein optimales an den Kunden angepasstes Produkt erstellt wird.

2.4 Scrumban

Scrumban ist ein Hybridverfahren aus Scrum und Kanban. Es verbindet die Teamarbeit von Scrum mit der fortlaufenden Verbesserung der Menge an Aufgaben innerhalb eines Prozessschritts von Kanban (Vgl. Swisher, W. (2014), S.1). In Scrumban gibt es ähnlich wie in Scrum Rollen, die zu besetzen sind. *Product Owner* und *Entwicklerteam* sind mit den Rollen aus Scrum gleichzusetzen. Unterschiede gibt es jedoch beim Scrum Master bzw. in Scrumban *Scrumban Sensei*, dieser ist nicht zwangsläufig notwendig. Ebenso ist das *Product Backlog* und *Impediment Backlog* wiederzufinden. Weitere Artefakte sind *Selected Backlog*, *Story in Progress (SiP)*, *Task Backlog*, *User Story*, *Task*, *Parking Lot* und *Cumulative flow diagram (CFD)*. In das Selected Backlog werden Anforderungen aus dem Product Backlog überführt. Es gibt aber eine Begrenzung der möglichen Menge an Anforderungen (ähnlich wie WiP-Limit in Kanban). Erst wenn neue Kapazitäten frei werden, ist es möglich neue Aufgaben in das Selected Backlog zu ziehen. In dem SiP werden dann anschließend alle User Stories aufgeführt, die derzeitig zu bearbeiten sind (siehe Abbildung 2-4).



Quelle: [mm1 (2013)]

Abbildung 2-4 Vorgehensweise Scrumban

Hier können ebenfalls nicht beliebig viele User Stories bereitgestellt werden. Es erfolgt hier ebenfalls eine Limitierung.
Das Task Backlog stellt konkrete Aufgaben bereit, die zu erledigen sind. Diese Aufgaben wurden aus den User Stories abgeleitet. Der *Production Flow* (siehe Abbildung 2-4) wird auf einem Kanban Board abgebildet, auf dem die gleichen Bedingungen gelten wie in Kanban (siehe Kapitel 2.3 Kanban).

Scrumban setzt im Gegensatz zu Kanban wieder verstärkt auf Meetings. Anhand der Meetings ist es möglich den Ablauf der Methode nachvollziehen. Wie in Scrum wird zunächst das Product Backlog vom Product Owner alleine oder im Team erstellt. Der folgende Ablauf der Methode wird einmalig zu Beginn komplett durchlaufen. Anschließend nur dann, wenn eine neue User Story in das Selected Backlog geladen wird. Angenommen einer dieser beiden Fälle tritt ein, wird das Meeting *Sprint Planning#1* danach stattfinden. In diesem Meeting trifft das Entwickler Team und der Product Owner aufeinander und die neue User Story wird besprochen und die Anforderungen erkannt. Nachfolgend werden die User Stories in das SiP geschrieben und ein weiteres Meeting, *Sprint Planning#2*, findet statt. In diesem Meeting werden dann die Arbeitspakete aus den User Stories extrahiert und in das Task Backlog geschrieben. Das Meeting wird vom Scrumban Sensei geleitet. Folgend durchlaufen die einzelnen Tickets das Board. Am Ende wird nun geprüft ob das gesamte Produkt oder Vision (siehe Abbildung 2-4) fertiggestellt wurde. Es werden wie in den beiden vorherigen Methoden ebenfalls Wert auf eine Retrospektive und Review wertgelegt. Sollte es zu Problemen in dem Prozess oder mit blockierten Tickets kommen, kann der Scrumban Sensei ein *Andon Meeting* veranlassen. Dort trifft das gesamte Scrumban-Team zusammen und versucht eine Lösung zu finden.

2.5 Auswahl der Methode

Durch eine Nutzwertanalyse über die einzelnen Methoden ist IMPACT zu dem Ergebnis gekommen, Scrum als Softwareentwicklungs-Vorgehensmodell zu nutzen (siehe Tabelle 2-1). Die Bewertungen erstrecken sich von eins bis drei, wobei der Wert eins aussagt, dass die Methode die Anforderung gering bis gar nicht erfüllt und drei die optimale Erfüllung darstellt.

Kriterien	**Gewicht**	**Scrum**		**Kanban**		**Scrumban**	
		Bewertung	Gesamt	Bewertung	Gesamt	Bewertung	Gesamt
Zeitliche Limitierung	50%	3	1,5	1	0,5	1	0,5
Rollenverteilung	30%	3	0,9	1	0,3	3	0,9
Erfahrungswerte der Mitglieder	10%	2	0,2	1	0,1	1	0,1
Visualisierung des Prozesses	10%	2	0,2	3	0,3	3	0,3
Ergebnis	100%		**2,8**		**1,2**		**1,8**

Quelle. Eigene Darstellung

Tabelle 2-1 Nutzwertanalyse der Softwareentwicklungs-Vorgehensmodelle

Die Methode Scrum erzielte das höchste Ergebnis mit 2,8. Bei den wichtigsten Kriterien zur Auswahl der Methode „Zeitliche Limitierung“ und „Rollenverteilung“ wurde Höchstpunktzahl erreicht. Das Kriterium „Zeitliche Limitierung“ ist besonders wichtig in dem Kontext von IMPACT. Es ist notwendig, dass die Aufgaben fristgerecht bearbeitet und fertiggestellt werden, da das gesamte Projekt zeitlich limitiert ist. Kanban und Scrumban sind fortlaufende Prozesse, die eine zeitliche Limitierung nicht in dem Umfang von Scrum liefern. Ebenso soll durch die zeitlich begrenzten Sprints die Motivation der Mitglieder gesteigert werden, da am Ende ein Produkt erschaffen wurde. Zusätzlich erfüllt Scrum und Scrumban das Kriterium „Rollenverteilung“ in vollem Umfang. Beide Methoden stellen eine Rollenstruktur zur Verfügung. Dieses Kriterium kann als wichtig erachtet werden, da das Team, welches zum Teil dezentral arbeiten wird, Ansprechpartner zu prozeduralen, organisatorischen und anforderungsspezifischen Fragen benötigt. Kanban hat keine vorgesehene Rollenverteilung und erfüllt dieses Kriterium nicht. Ein weiteres Kriterium - „Erfahrungswerte der Mitglieder“ - wurde von Scrum zum Teil erfüllt. Wie bereits erwähnt, besitzen einige Teammitglieder Erfahrungen mit dieser Methode und können diese einbringen.

3 Scrum

Um Scrum in IMPACT nutzen zu können, müssen Anpassungen vorgenommen werden. Diese Anpassungen sind in den Rollen und im generellen Ablauf des Prozesses zu finden.

3.1 Rollen

Eine klare und strikte Trennung der einzelnen Rollen kann es in der Projektgruppe nicht geben. Scrum Master und Product Owner belegen jeweils Doppelrollen, da sie ebenso in das Entwicklungsteam integriert sind. Zusätzlich ist ein Projektmanager im Team vorhanden. Es können die Rollen User, Manager und Customer belegt werden. Um aber die notwendigen Rollen für IMPACT genauer zu erläutern, werden die Rollen Manager und User nicht genauer beleuchtet.

3.1.1 Product Owner

Der „Product Owner“ ist als Auftraggeber richtungsweisend für das Produkt, indem er dem Entwicklerteam eine „Vision“ vermittelt (Vgl. B. Gloger (2013), S.78 ff.). Ziel ist es, dass die Entwickler eine klare Vorstellung des Endproduktes haben, damit sie motiviert darauf hinarbeiten. Ebenso liegt in seinem Aufgabenbereich das Erstellen und Priorisieren des Product Backlogs. Die Vision wird im Product Backlog konkretisiert und Product Backlog Items daraus abgeleitet. Das Backlog und die Items können auch von dem Entwicklerteam und müssen nicht zwangsläufig von dem Product Owner geschrieben werden. Ist das Product Backlog fertig gestellt, priorisiert der Product Owner die einzelnen Items. Dadurch kann er eine Reihenfolge der zu entwickelnden Items festlegen und steuert den Ablauf der Programmierung. Gleichermaßen zählt es zu seinen Aufgaben nach einem Sprint das lauffähige System zu betrachten und zu bewerten. Er gibt dem Entwicklerteam ein Feedback über das System und kann hier neue Ideen einbringen. Diese Rolle ist ein entscheidender Faktor für das Gelingen des Projektes, da er Entscheidungen bezüglich des Produktes treffen darf.

Diese Rolle wird in IMPACT durch Patrick Smit belegt. Er wird sehr eng mit Herrn Kurzhöfer zusammenarbeiten. Herr Kurzhöfer repräsentiert den Kunden Lufthansa Industry Solutions Oldenburg. Er wird in Absprache mit Herrn Kurzhöfer, ebenso Entscheidungsbefugnis bezüglich des Systems bekommen. Ebenso wird er in dem Entwicklungsteam seine Aufgabe erfüllen. Wie vorgehsehen werden alle Mitglieder von IMPACT das Product Backlog und die Items erstellen.

3.1.2 Scrum Master

Der Scrum Master ist für den Ablauf in Scrum zuständig. Er achtet darauf, dass der Prozess korrekt eingeführt und gelebt wird. Er versucht Impediments (Blocker) zu beseitigen, die das Team am

produktiven Arbeiten hindern. Die Impediments werden vonm Scrum Master in die Impediments Liste eingetragen und priorisiert. Impediments können z.B. folgenden Bereichen entspringen: Software-Entwicklungsprozess, Kommunikation und Abstimmung und persönliche Konflikte innerhalb des Teams (Vgl. B. Gloger (2013) S.89). Die Zusammenarbeit zwischen Entwicklerteam und Product Owner ist von Bedeutung. Er hilft den Beteiligten bei internen Problemen und hält es dazu an, die Regeln von Scrum einzuhalten. Weiterhin hilft er dem Team die Ziele zu erreichen. Daraus kann eine Situation entstehen, in der der Scrum Master Entscheidungen treffen muss. Dies kann der Fall sein, wenn z.B. der Product Owner nicht zu erreichen ist und eine schnelle Entscheidung getroffen werden muss (Vgl. zu diesem Abschnitt B. Gloger (2013) S.98 ff.).
Der Scrum Master trägt dabei die volle Verantwortung. Zusätzlich zählt es zu seinen Aufgaben, z.B. das Daily Scrum und das Sprint Review zu moderieren.

Die Rolle wird in der Projektgruppe von Dirk Tesche belegt. Er wird mit dem Projektleiter zusammenarbeiten und darauf achten, dass der Scrum Prozess eingehalten wird. Zu seinen Aufgaben zählen, das *Planning Poker* zu leiten und in Jira & Confluence Task anzulegen und zu pflegen. Er wird dem Team bei prozessualen Unklarheiten zur Seite stehen und mögliche Blocker beseitigen.

3.1.3 Entwicklerteam

Das Entwicklerteam erstellt das Produkt. Ebenso sammelt und bearbeitet es die Anforderungen, designt, testet und stellt ein fertiges Produkt bereit (Vgl. B. Gloger (2013), S. 66). Dabei ist es zuständig für die Qualität, das Liefern des Produktes[2], der Aufwandsschätzung und der Selbstorganisation. Es muss alles in seiner Macht Stehende unternehmen um das Sprintziel zu erreichen und dabei die bereits genannten Faktoren berücksichtigen (Vgl. B. Gloger (2013), S.77). Jedes Teammitglied ist dabei wertvoll und ist dazu angehalten, die definierten Ziele umzusetzen.

Die gesamte Projektgruppe erfüllt diese Definition eines Entwicklerteams.

3.1.4 Customer

Der Kunde fordert das Produkt und stellt gewisse Anforderungen an dieses. Er arbeitet eng mit dem Product Owner zusammen und kann diesen selbst stellen oder jemanden aus dem Projektteam wählen. Es sollte die Möglichkeit bestehen, entwickelte Funktionalitäten zu bewerten und ein Feedback zu diesen zu geben. Zusätzlich sollte er in der Lage sein, neue Anforderungen und Wünsche zu äußern, so dass diese übernommen werden.

---

[2] nach einem Sprint und das gesamte Produkt

Die Projektgruppe besetzt diese Rolle mit dem Betreuer Herr Kurzhöfer. Er wird in den Sprint Plannings anwesend sein und im engen Kontakt mit dem Product Owner Patrick Smit stehen.

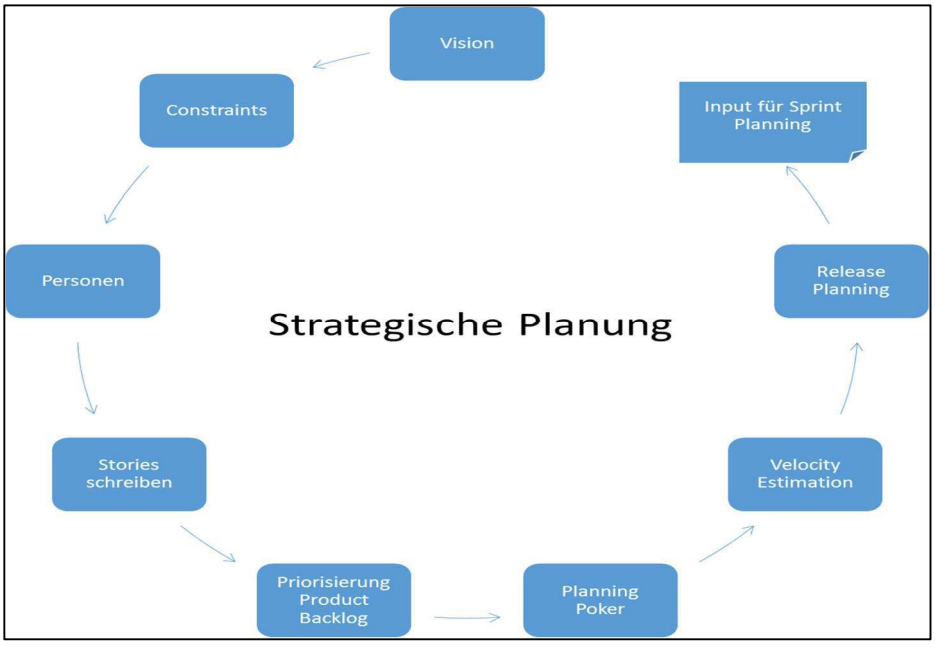
3.2 Artefakte

Die bereits oben genannten Artefakte (siehe Kapitel 2.2 Scrum), das Sprint Goal und der Releaseplan werden als Artefakte in IMPACT implementiert. Der Releaseplan zeigt dem Team, in welchen Sprints es welche Backlog Items umsetzen kann. Dieser Plan soll dabei nur Informationen darüber liefern, wann welches Backlog Item zu erwarten ist und keine verbindlichen Pläne anzeigen.

Der Releaseplan kann mit dem Meilensteinplan verbunden werden, dadurch ist es möglich gezielter Sprints zu planen und einen schnellen Fortschritt zu erzielen.

3.3 Ablauf des Prozesses

Grundsätzlich könen in zwei Planungsbereiche unterschieden werden: Zum einen die taktische Planung, bei der einzelne Sprints geplant werden, und zum anderen die strategische Planung, bei der es um die Langzeit-Planung des Projektes geht.



Quelle: in Anlehnung an: B.Gloger (2013), S. 151

Abbildung 3-1 Strategische Planung in Scrum

In Abbildung 3-1 ist das Vorgehen der strategischen Planung abgebildet, angefangen bei der Vision des Product Owners bis hin zum Input für das Sprint Planning. Wie in Kapitel 3.1.1, ist dieser für die Vision zuständig. Er erschafft eine Vision, um die Teammitglieder zu motivieren und ihnen ein *Big*

Picture mitzugeben. Um diese Version zu finden, gibt es die Methoden *Freewriting* und *Elevator Pitch*. Anschließend werden Rahmenbedingungen (*Constraints*) geschaffen. Die Constraints erstrecken sich über die Bereiche Design, Feature, Technologie, Standards und rechtliche Bedingungen. (Vgl. zu diesem Abschnitt B-.Gloger (2013), S.124)
Zusätzlich soll eine Person erstellt werden, die das Produkt benutzt. Sie soll dazu dienen, entwickelte Funktionalitäten aus der Anwenderperspektive zu betrachten und zu hinterfragen. Diese Person wird durch Interviews mit Usern geprägt und immer weiter verfeinert. Ist die Vision fertiggestellt, werden die User Stories geschrieben und priorisiert. Die Eigenschaften und Funktionen werden in dem Product Backlog festgehalten. Wichtig ist, dass die Items nicht technisch spezifiziert werden. Ein mögliche Formulierung für ein Item könnte wie folgt lauten: „Als User benötige ich die Möglichkeit, mich anzumelden, damit ich meine Ideen einsehen kann“. Der Product Owner ist allein für die Priorisierung zuständig. Liegt ein bewertetes Product Backlog vor, kann das Planning Poker vorgenommen werden. Es ist eine Methode, die innerhalb des Teams dazu dient, den Arbeitsaufwand abzuschätzen (Planning Poker (o.J.)).
Im folgenden Schritt wird die *Velocity*, d.h. die Kapazität des Teams bestimmt. Damit ist gemeint, wie schnell ein Sprint absolviert werden kann. Beginnend mit dem ersten Product Backlog Item wird geschaut, ob das Team dieses in dem Sprint erreichen kann. Wenn das Team bereits Sprints absolviert hat, kann es die bisherigen entwickelten Funktionalitäten als Richtwert nehmen. (Vgl. zu diesem Abschnitt B. Gloger, S. 148 ff.)
Ergänzend wird die Strategie Planung durch den Releaseplan. Durch die Velocity ist es nun möglich die Länge des Projektes und die Anzahl der Sprints zu planen. Dabei gilt die Regel $\frac{Anzahl\ Stories}{Kapazität\ des\ Teams} = Anzahl\ Sprints$. Jedoch ist dies nur ein grober Richtwert und ist von vielen weiteren Faktoren abhängig, so kann z.B. der Velocity erst nach drei bis fünf Sprints korrekt ermittelt werden. (Vgl. zu diesem Abschnitt B-Gloger, S. 149)
Die aus diesem Vorgang ermittelten Informationen werden für das Sprint Planning benötigt.

Der Ablauf des Scrum-Prozesses wird sich stark an die Gegebenheiten anpassen. Das Team arbeitet zum Teil dezentral, ist nicht jeden Tag verfügbar und wird in den meisten Fällen nicht gleichzeitig arbeiten.

3.3.1 Estimation Meeting

Das Estimation Meeting wird mit dem gesamten Scrum-Team abgehalten. Ziel ist es, die Stories des Product Backlogs kennenzulernen, deren Funktionsaufwand abzuschätzen und bisherige Stories zu teilen, zusammenzufügen oder neue hinzuzufügen. Der Product Owner wird die einzelnen Stories vorstellen und das Team wird mithilfe von Planning Poker den Aufwand abschätzen. Daraus entsteht ein geschätztes Product Backlog, welches in dem folgenden Sprint Planning mit weiteren Informationen angereichert und in das Selected Backlog überführt wird. Das geschätzte Product

Backlog sollte Items für ca. drei Sprints ausweisen können (Vgl. zu diesem Abschnitt B. Gloger (2013), Beilage S. 4 ff.).
Das Estimation Meeting ist in der Strategie-Planung enthalten und nimmt viele Elemente daraus auf.
Dauer: Wöchentlich, circa 30 Minuten.
Teilnehmer: Entwicklerteam, Scrum Master, Product Owner

Vorschlag zu Anpassungen:
IMPACT sollte ebenfalls das Estimation Meeting einführen. Jedoch muss darauf geachtete werden, dass der zeitliche Rahmen eingehalten wird. Ob es notwendig ist, dieses Meeting wöchentlich abzuhalten, sollte nach den ersten Sprints entschieden werden. Das Scrum-Team sollte darüber entscheiden ob es ausreicht, vor jedem neuen Sprint dieses Meeting abzuhalten oder ob wöchentlich neue Abschätzungen notwendig sind. Gerade am Anfang wäre es eventuell hilfreich dieses Meeting wöchentlich zu halten, da gerade die Aufwandschätzung schwierig erscheint und zusätzlich zu diskutieren ist, ob es neue Stories entstanden sind oder erstehen sollen. Zudem wird kein Planning Poker in diesem Meeting stattfinden. Das Planning Poker soll im Sprint Planning#2 über die Aufgaben stattfinden.

3.3.2 Sprint Planning

Das Sprint Planning dient zur taktischen Planung. Nach B. Gloger (2013) wird das Sprint Planning in zwei Meetings aufgeteilt. Beide Meetings werden an einem Tag abgehalten. Am Ende des Sprint Planning#1 steht das Sprintziel fest und alle Scrum-Teammitglieder verpflichten sich dieses zu erfüllen.

Sprint Planning#1

Ziel ist im Sprint Planning#1, die Menge der zu erreichenden Items zu begrenzen. Daraus entsteht das Selected Product Backlog. Die User Stories werden einzeln durchlaufen und für jedes die Anforderungen geklärt, Constraints bestimmt und Abnahmekriterien ermittelt. Anschließend geht der Scrum Master sicher, ob die Stories im jetzigen Sprint bearbeitet werden können. Dieser Vorgang wiederholt sich für jedes Item und endet ca. 20 Minuten vor Ende der Zeit. In den letzten 20 Minuten durchläuft der Scrum Master mit dem Entwicklerteam nochmals alle Items und stellt sicher, dass diese umgesetzt werden sollen. Dabei muss das Team einstimmig entscheiden. Ist dies nicht der Fall, wird das vorherige Item als letztes Item gesichert. Anschließend verlassen alle bis auf den Scrum Master und das Entwicklerteam den Raum und der Scrum Master vergewissert sich über die Richtigkeit der Items (Vgl. zu diesem Abschnitt B. Gloger, Beilage S.7 ff.).
Nach Zustimmung ist die Entscheidung getroffen, das Sprint Goal steht fest und alle Scrum-Teammitglieder verpflichten sich dieses zu erfüllen.
Dauer: 60 Minuten pro Sprint Woche (max. 4 Stunden)

Teilnehmer: Entwicklerteam, Scrum Master, Product Owner, Customer

Vorschlag zu Anpassungen:

Das Sprint Planning#1 sollte komplett übernommen werden. Jedoch ist zu beachten, dass der Scrum Master und der Product Owner ebenso Teammitglieder sind. Der Scrum Master sollte in diesem Fall eher in der Scrum Master Rolle sein, da dieses Meeting den Grundbaustein für den kommenden Sprint bildet. Dasselbe gilt für den Product Owner.

Sprint Planning#2

Ziel des Sprint Planning#2 ist es, das Design der zuvor festgelegten Items zu definieren. Dazu werden zunächst offene Fragen zu den User Stories geklärt und anschließend technische Fragen diskutiert. Hierzu gehören z.B. die Schnittstelle, Datenbanken, Architektur und das Zusammenspiel mit bisherigen umgesetzten Funktionalitäten. Anschließend wird das nächste Item bearbeitet. In den letzten 10 Minuten werden auf Post-its oder Kärtchen erste Aufgaben geschrieben und an das *Task Board* gehängt. Das Task Board dient zur Übersichtlichkeit der Aufgaben und deren Status. Es kann z.B. die Spalten “Selected Backlog Items” bzw. “Stories”, “Tasks to Do”, “Work in Progress” und “Done”. Die einzelnen Aufgaben wandern mit der Zeit über das Board. Unter der Spalte „Stories“ werden die Stories in priorisierter Reihenfolge gehängt. In der Spalte „Tasks to Do“ werden die dazugehörigen Aufgaben gesammelt (Vgl. zu diesem Abschnitt B. Gloger, S.167 ff.).
Das Task Board ist somit eine Technik für das Sprint Backlog. Desweiteren sollen die Aufgaben kleineren Gruppen zugeteilt werden. Diese Gruppen sind nach Themengebiet unterteilt, wie z.B. Programmierung und Oberflächendesign.
Dauer: 60 Minuten pro Sprint-Woche (max. 4 Stunden)
Teilnehmer: Entwicklerteam, Scrum Master, eventuell Product Owner und Customer

Vorschlag zu Anpassungen:

Auch dieses Meeting sollte mit seinen Regeln komplett übernommen werden. Da IMPACT Jira & Confluence benutzt, kann das Task Board oder auch Scrum Board dort virtuell umgesetzt werden. Um jedoch besser in den Scrum-Prozess zu finden, könnte gerade in der Anfangszeit ein Whiteboard oder ähnliches als Taskboard eingesetzt werden. Dort ist es einfacher Notizen hinzuzufügen oder Änderungen vorzunehmen. Wenn alle Mitglieder mit dem Prozess vertraut sind, kann auf dieses Board verzichtet und nur noch mit Jira & Confluence gearbeitet werden. Ebenso sollte sich das Team einig werden, wann eine Aufgabe als „Done“ zu definieren ist. Zusätzlich könnte das Sprint Backlog auch in einer Excel-Tabelle geführt werden, wenn der Bedarf besteht.

3.3.3 Sprint

Wenn das Sprint Planning beendet ist, startet der Sprint. Dieser kann bis zu 4-5 Wochen dauern und sollte möglichst immer zu Ende geführt werden. Um den Fortschritt innerhalb des Sprints nachvollziehen zu können, können Techniken wie Task Board und *Burn-Down-Chart* genutzt werden. Innerhalb des Sprints findet täglich das Daily Scrum statt.

Daily Scrum

Ziel ist es, dass das Team den Tag plant und koordiniert. Es wird der Fortschritt jedes einzelnen auf dem Task Board festgehalten und Impediments von dem Scrum Master auf seine Impediments List aufgenommen. Neue Aufgaben werden vergeben. In diesem Meeting sollten jedoch keine technischen Diskussionen auftreten. Das Task Board und das Burn-Down-Chart werden in diesem Meeting aktualisiert aber nicht bearbeitet.
Dauer: 15 Minuten täglich
Teilnehmer: Entwicklerteam und Scrum Master im Stehen.

Vorschlag zu Anpassungen:
Das Daily Scrum ist in der Projektgruppe so nicht auszuführen. Es ist dem Team nicht möglich, jeden Tag an einer Aufgabe zu arbeiten und anwesend zu sein. Die Zeitabstände müssten hier auf eine Woche gesetzt werden, es handelt sich also um ein *Weekly Scrum*. Jedoch sollte, wenn das Projekt in einer sehr arbeitsaufwendigen Phase ist, die zeitliche Frequenz erhöht werden. Ein Vorschlag wäre zweimal wöchentlich. Ebenso sind 15 Minuten relativ wenig wenn es nur einmal in der Woche zu einem Meeting kommt. Zunächst könnte die Dauer auf 20 Minuten und ggf. höher oder niedriger gesetzt werden.

3.3.4 Sprint Review

Das Sprint Review findet zum Abschluss eines Sprints statt. Ziel ist ein Feedback von allen Beteiligten zu bekommen, um das Produkt besser zu gestalten. Das Entwicklerteam führt die neuen Funktionalitäten vor und lässt den User diese ausprobieren. Aus dem Feedback können z.B. neue Stories entstehen, die in das Product Backlog aufgenommen werden. Wichtig ist, dass nur Funktionalitäten vorgestellt werden die auch den „Done“-Status erreicht haben (Vgl. zu diesem Abschnitt B. Gloger, Beilage S. 20 ff.).
Dauer: 90 Minuten
Teilnehmer: Entwicklerteam, Scrum Master, Product Owner, Customer

Vorschlag zu Anpassungen:

Das Sprint Review sollte ebenfalls in vollem Umfang übernommen werden. Es ist mit eines der wichtigsten Bausteine, um mit dem Kunden zu agieren und das Produkt den Kundenwünschen entsprechend zu gestalten.

3.3.5 Sprint Retrospektive

Die Sprint Retrospektive sollte im Anschluss an das Sprint Review stattfinden. Es dient dazu den Arbeitsablauf innerhalb und außerhalb des Teams zu reflektieren. Ziel ist es den Arbeitsablauf weiter zu optimieren, damit das Team produktiver arbeiten kann. Es bildet damit das wichtigste Element innerhalb des Scrum-Prozesses. Zunächst soll zusammengefasst werden, was alles gut gelaufen ist und anschließend mögliche Verbesserungen zusammen getragen werden. Das Entwicklerteam und der Scrum Master tragen die Vorschläge auf dafür vorgesehene Kärtchen auf, die anschließend unter den jeweiligen Fragen angeheftet werden. Dabei stellt jeder kurz seine Karte vor. Die Vorschläge werden dann gruppiert und versucht in dem nächsten Sprint umzusetzen (Vgl. zu diesem Abschnitt B. Gloger, Beilage S.22 ff).
Dauer: 90 Minuten
Teilnehmer: Entwicklerteam, Scrum Master, eventuell Product Owner

Vorschlag zu Anpassungen:
Die Sprint Retrospektive sollte ebenfalls in den Scrum-Prozess von IMPACT übernommen werden. Dieses Meeting wird besonders wichtig sein um den Prozess zu verbessern und ihn besser kennenzulernen. Es muss aber darauf geachtet werden, dass es zu keinen Vorwürfen innerhalb des Teams kommt, sondern auf einer fachlichen Ebene Vorschläge entgegen genommen werden.

4 Fazit/ Ausblick

Die Wahl der Softwareentwicklungsmethode ist mit Scrum gefallen. Scrum ermöglicht es, den Prozess an die Projektgruppe anzupassen und zu erweitern. Mit Scrum kann die gesamte Projektgruppe strukturiert, koordiniert und produktiv arbeiten. Dabei ist es wichtig die Selbstständigkeit der Projektgruppe zu fördern. In diesem Rahmen wird es ihr ermöglicht, kreativ auf neue Vorschläge und Veränderungen einzugehen. Jedes Mitglied kann neue Bereiche kennenlernen und seine fachlichen Kompetenzen ausbauen.

Es sollte parallel zum Task Board das Burn Down Chart gepflegt und genutzt werden. Dieses Chart sollte direkt von Beginn an in Jira & Confluence gepflegt werden. Es ist zusätzlich eine Unterstützung um den Arbeitsverlauf innerhalb des Sprintes zu beobachten und diesen später zu reflektieren. Es ist möglich daraus Blocker frühzeitig zu erkennen.

Ergänzend müssen Regeln aufgestellt werden, die klären wann ein Task als „Done“ bezeichnet werden darf. Grundsätzlich sollte es nicht möglich sein, Tasks mit dem Status „Done“ wieder in einen anderen Status zu schieben. Aus einem „Done“-Taskstatus können aber neue Tasks abgeleitet werden. Eine mögliche Definition des Status könnte sein, dass eine Task auf „Done“ gesetzt werden darf, wenn die Aufgabe komplett erfüllt ist. Dafür ist es notwendig, dass die Tasks granular vorliegen und eine genaue Aufgabenbeschreibung besitzen. Sollte es nicht möglich sein dieses Ticket in dem Sprint zu bearbeiten, wird dieses wieder in das Product Backlog als User Story übernommen und muss dann im nächsten Sprint bearbeitet werden. Es wird aber nicht zum Sprintende zu „Done“ erklärt. Dadurch ist es auch möglich in der Sprintretrospektive zu klären ob der Prozess optimiert werden muss. Im nächsten Sprint Planning könnte nochmal geschaut werden ob eventuell die Task zu groß ist und nochmal aufgeteilt werden muss. Wichtg ist, dass für jedes Teammitglied klar wird wann der Status erreicht ist. Nicht zu vergessen ist dabei aber, dass im Team gearbeitet wird und wenn ein Mitglied nicht weiterkommt, es sich die Hilfe der anderen sicher sein kann. Erst wenn sich die gesamte Projektgruppe als selbstorganisiertes Team versteht, kann der Scrumprozess in vollen Umfang umgesetzt werden.

Literaturverzeichnis

Agilemanifesto (2001): Manifest für Agile Softwareentwicklung URL:http://agilemanifesto.org/iso/de/ [01.06.2015]

AgilePrinzipien (2001): Prinzipien hinter dem agilen Manifest URL: http://agilemanifesto.org/iso/de/principles.html [01.06.2015]

B. Gloger (2013): Scrum: Produkte zuverlässig und schnell Entwickeln, 4.Aufl. München, ISBN: 978-3-446-43338-0

Bleek, W.-G., Wolf, H. (2010): Agile Softwareentwicklung: Werte, Konzepte und Methoden. 2.Aufl. Heidelberg, ISBN: 978-3-89864-701-4

Kieninger, T. (o.J.): Kanban Kompakt, URL: http://www.braintime.de/methoden/ueberblick-kanban-beratung/kanban-grundlagen-kompakt/ [05.06.2015]

IT-Agile (o.J.): Was ist agile Softwareentwicklung? URL: http://www.it-agile.de/wissen/methoden/agilitaet/ [01.06.2015]

Leopold, K., Kaltenecker, S. (2012): Kanban in der IT: Eine Kultur der kontinuierlichen Verbesserung schaffen, München, ISBN: 978-3-446-43059-4

mm1 (2013): Scrumban Plakat, URL: http://mm1.de/fileadmin/content/Media_und_Poster/mm1_Scrumban_Poster.png [21.05.2015]

Planning Poker (o.J.): Planning Poker – Das Spiel zur Aufwandsschätzung, URL: http://www.planningpoker.de/ [04.06.2015]

Scrum Kompakt (2015): Scrum-Prozess, URL: http://www.scrum-kompakt.de/einfuehrung-in-scrum/scrum-prozess/ [03.06.2015]

Swisher, W. (2014): Implementing Scrumban: A short guide to implementing Scrumban at your organization, URL: https://switchingtoscrum.files.wordpress.com/2013/12/implementing-scrumban_v1-32.pdf [05.06.2015]

B.1.11. Testmanagement

Testmanagement

Schriftliche Ausarbeitung
im Rahmen der Projektgruppe
IMPACT

Themensteller: Prof. Dr.-Ing. Jorge Marx Gómez
Betreuer: Dr. Joachim Kurzhöfer
Dipl.-Math. Jens Siewert
Stefan Wunderlich M. Sc.

Vorgelegt von: Dirk Tesche
Wilhelm-Raabe-Straße 9
26131, Oldenburg
0179 69 91 939
E-Mail: dirk.tesche@uni-oldenburg.de

Abgabetermin: 22 Juni 2015

Inhaltsverzeichnis

Verzeichnis der Abkürzungen und Akronyme

ISTQB	International Software Testing Qualifications Board
u.a.	Unter anderem
z.B.	Zum Beispiel

Abbildungsverzeichnis

1 Einleitung

Das erste Kapitel wird dem Leser die Ziele und die Motivation des Autors verdeutlichen. Des Weiteren werden die Notwendigkeit eines Testmanagements und die entstehenden Probleme bei einem fehlerhaften Testmanagement erläutert. Um den Fokus auf einige wesentliche Punkte richten zu können, werden die Rahmenbedingungen und das methodische Vorgehen beschrieben.

1.1 Problemstellung

Als entscheidende Faktoren eines Softwareentwicklungsprojektes werden Kosten, Zeit und Qualität angesehen. Um diese entsprechend den verfügbaren Ressourcen effizient einzusetzen, wird ein Projektmanagement benötigt, welches die Verteilung der Ressourcen koordiniert. Insbesondere der Faktor Zeit wird in vielen Projekten als die kritischste Komponente bewertet. In den klassischen sequentiellen Softwareprojekten wurden die Software-Tests zum Ende des gesamten Projektes geplant. Durch diese Positionierung wurden die Testphase durch das Projektmanagement häufig als Pufferzeit genutzt, um so das Projekt im vorgegeben Zeitrahmen abzuschließen zu können. Eine Vernachlässigung der Software-Tests kann zu Softwarefehlern führen, welche einen Einfluss auf die Qualität und Kosten haben. Die Problematik der fehlerhaften Software wurde zum Jahrtausendwechsel deutlich. Aufgrund des besonderen Datumswechsels waren zahlreiche Anpassungen notwendig, um die Funktionsfähigkeit der Software zu gewährleisten. Durch diese bis dato einmalige Situation wurde das Verständnis für die Notwendigkeit von Software-Tests geschärft (Vgl. Baumgartner, et al. 2013, S. 18). Um diesen neuen Stellenwert zu entsprechen, wird in den angesprochenen Softwareentwicklungsprojekten dem Testmanagement eine elementare Bedeutung zu teil. Zu den Aufgaben des Testmanagement zählen u.a. die Organisation, Entwicklung und Koordination der Tests, sowie deren Dokumentation und Kontrolle. Diese Arbeit skizziert in den nachfolgenden Kapiteln die grundlegenden Elemente und Aufgaben des Testmanagements.

1.2 Eingrenzung des Untersuchungsspektrums

Das Aufgabenspektrum des Testmanagements wird im Rahmen dieser Arbeit nicht vollständig abgebildet. Diese Seminararbeit wird den Bedürfnissen der Projektgruppe IMPACT entsprechend nur einige Punkte behandeln. Im Rahmen dieser Arbeit wird dem Leser in den Grundlagen eine Übersicht über den fundamentalen Testprozess und deren Ausprägungen vermittelt. Des Weiteren werden die für die Projektgruppe entscheidenden Arten von Tests und agilen Methoden beschrieben. Ein wesentlicher Bestandteil dieser Arbeit wird es sein, im dritten Kapitel die konzeptionelle

Entwicklung eines Tests im Rahmen einer Testautomatisierung zu untersuchen. Hierbei wird das Framework JUnit zur möglichen Unterstützung betrachtet.

1.3 Motivation und Ziele

Die Projektgruppe IMPACT wird im Rahmen eines Softwareentwicklungsprojektes eine Innovationsplattform entwickeln und diese programmieren. Im Rahmen dieser Tätigkeiten werden entsprechend den Entwicklungsstufen verschiedene Arten von Test und Methoden verwendet werden. Hierzu ist es elementar, dass alle Mitglieder der Projektgruppe über die Bedeutung von Software-Tests, deren Entwicklung und Durchführung umfassend informiert werden. Die Ziele eines umfassenden Testmanagements werden im Allgemeinen innerhalb des Projektmanagements definiert. Hierbei wird ein wesentlicher Faktor sein, die Ressourcen Zeit und Geld effizient zu verwalten. Die Bedeutung eines Testmanagements wird bei der Betrachtung der vollendeten Software deutlich. Werden keine oder nur ungenügend Software-Tests durchgeführt, kann es bei Auslieferung der Software zu Fehlern, welche durch die Endanwender festgestellt werden, kommen. Diese Fehler verursachen neben dem Imageschaden erneute Kosten, die durch ausreichende Software-Tests vermieden werden können. Aus diesem Grund sollte ein umfassendes Testmanagement im jedem Softwareentwicklungsprojekt ein fester Bestandteil sein.

2 Grundlagen zum Testmanagement

Im folgenden Abschnitt werden die anfallenden Managementtätigkeiten des Testprozesses in Softwareprojekten erläutert. Des Weiteren werden die wesentlichen Arten von Tests, welche während des Softwareentwicklungsprozesses anfallen, aufgegriffen. Dieses Kapitel schließt mit einer Einführung in agile Methoden und Techniken ab.

2.1 Testprozess

Der Testprozess nach ISTQB wird, wie in Abbildung 1 beschrieben, in die Phasen Testplanung, Analyse / Entwurf, Realisierung / Durchführung, Bewertung der Endkriterien und Abschluss der Testaktivitäten gegliedert (Vgl. ISTQB 2011, S. 5 ff.). Die Tätigkeiten der Testüberwachung und -steuerung werden als Querschnittsaufgaben wahrgenommen, welche in jeder Phase des Prozesses vom Testmanagement durchgeführt werden.

Abbildung 1 - Fundamentaler Testprozess (In Anlehnung an: Spillner, et al. 2014, S. 14)

Im weiteren Verlauf dieses Kapitels werden die Bestandteile und Tätigkeiten der einzelnen Phasen beschrieben und die entscheidenden Artefakte der jeweiligen Phase aus Sicht des Testmanagements vorgestellt.

Testplanung

In der ersten Phase der Testplanung werden die Testziele und die Strategie definiert und mit dem gesamten Team abgestimmt. Insbesondere werden in dieser Phase die Testmethoden, Testumgebung und benötigte Ressourcen definiert (Vgl. Spillner, et al. 2014, S. 14 ff.). Als ein weiterer Bestandteil dieser Phase werden Planung und Organisation der einzelnen Testaktivitäten, vor allem die Definition der einzelnen Aufgaben innerhalb des Testumfangs sowie die Zielsetzung der Tests definiert und dokumentiert. Die Priorisierung der Aufgaben und die Zeitplanung werden in agilen Projekten mit dem Team ermittelt (vgl. Spillner, et al. 2014, S. 22 ff.). Parallel dazu werden die Aufgaben der Steuerung und Überwachung durch das Team wahrgenommen. Hierbei werden insbesondere nach weiteren Iterationen Korrekturmaßnahmen an der Testplanung durchgeführt und der Testfortschritt überwacht (Vgl. Baumgartner, et al. 2013, S. 39). Die Aufgaben von Testanalyse und -entwurf kennzeichnen die nächste Phase des Testprozesses.

Testanalyse und -entwurf

Die in der Testplanung entstandenen Testaktivitäten werden in der Analyse weiter bearbeitet. In dieser Phase werden anhand dieser Informationen die Testbedingungen identifiziert, das bedeutet, wie intensiv und mit welchem Detailierungsgrad getestet wird. Einfluss auf die Identifikation der Testbedingungen haben u.a. bereits erkannte Risiken, das Beziehungsgeflecht einzelner Funktionen und Methoden, Managementanforderungen und die Wahl der Testwerkzeuge (Vgl. Spillner, et al. 2014, S. 40 ff.). Anschließend werden die Testfälle entworfen und vorbereitet. Das methodische Vorgehen beim Entwerfen eines Tests wird in Kapitel 3 beschrieben. Die erstellten Testbedingungen und die entworfenen Tests werden in einer Testdokumentation fixiert. Diese Dokumentation dient in weiteren Iterationen als Grundlage für das Review, welches im Rahmen der Testüberwachung durchgeführt wird (Vgl. Baumgartner, et al. 2013, S. 41).

Testrealisierung und -durchführung

Während der Testrealisierung werden durch das gesamte Team die Realisierung und die Priorisierung endgültig festgelegt. Dazu dient im SCRUM-Prozess das Product Backlog. (Vgl. Seminararbeit SE-Vorgehensmodell) Damit die Tests durchgeführt werden können, werden die Testdaten und -szenarien erstellt und die endgültige Testumgebung aufgesetzt (Vgl. Spillner, et al. 2014, S. 45 f.). Des Weiteren werden die entwickelten Software-Tests zu Testsuiten bzw. Testszenarien zusammengefasst. Nachdem alle vorbereiteten Maßnahmen abgeschlossen wurden, werden die Tests gemäß Testablaufplan durchgeführt. Anschließend werden die Ergebnisse mit den erwarteten Ergebnissen verglichen und besprochen (Vgl. Baumgartner, et al. 2013, S. 42 f.). Anhand dieser Auswertung wird entschieden, ob es weitere Nachtest in der aktuellen oder nachfolgenden Iteration geben wird. Alle Entscheidungen werden im Rahmen der Testdokumentation protokolliert.

Bewertung von Endkriterien

In dieser Phase werden die Testprotokolle ausgewertet. Auf Grundlage dieser Auswertung wird entschieden, ob die zuvor im Testkonzept festgelegte Anzahl an Tests ausreicht oder ob zusätzliche Tests notwendig werden. In agilen Projekten ist diese Auswertung ein Bestandteil der täglichen Gruppen-Meetings (Vgl. Baumgartner, et al. 2013, S. 43). Des Weiteren wird der Ist-Aufwand mit dem zuvor geplanten Aufwand verglichen, um so die Aufwandschätzungen für neue Iterationen zu optimieren. Abschließend wird an allen Iterationen ein Testabschlussbericht erstellt und dem Kunden übergeben (Vgl. Baumgartner, et al. 2013, S. 44).

Abschluss der Testaktivitäten

Der Abschluss der Testaktivitäten wird zur Kontrolle der Ergebnisse, Fertigstellung der Testdokumentation und Archivierung der Testumgebung, -mittel und der Infrastruktur verwendet. Diese Phase schließt mit einer Retrospektive ab, um so die gemachten Erfahrungen rückblickend zu analysieren (Vgl. Spillner, et al. 2014, S. 54).

Alle Aktivitäten innerhalb des Testprozesses werden durch das Testmanagement überwacht und gesteuert. In agilen Projekten werden diese Managementaufgaben von dem gesamten Team wahrgenommen und miteinander abgestimmt.

2.2 Arten von Tests

Im folgenden Abschnitt werden die im Softwareentwicklungsprozess gängigen Tests beschrieben und erläutert. Insbesondere werden neben einer Begriffserklärung die Testumgebung und die Testziele beschrieben.

Modultest

Mit dem Modultest wird die erste Teststufe beschrieben. Er wird auch je Programmiersprache auch als Komponenten-, Unit- oder Klassentest bezeichnet (Vgl. Linz und Spillner 2012, S. 44). Innerhalb eines Modultest werden Funktionen, Methoden und einzelne Klassen getestet. Das Ziel des jeweiligen Modultests wird anhand der zuvor definierten Spezifikationen für die Funktionalität abgeleitet (Vgl. Baumgartner, et al. 2013, S. 145). Sie werden im Allgemeinen in der Entwicklungsumgebung durch den Programmierer durchgeführt. Hierbei ist es notwendig, dass zuvor exakte formale Vorgaben zur Durchführung und Protokollierung gemacht werden. Dadurch soll gewährleistet werden, dass Software-Tests einen systematischen Ansatz in Form eines analytischen Vorgehens verfolgt (Vgl. Linz und Spillner 2012, S 45 ff.). Hierzu kann insbesondere die

Testautomation mit Hilfe eines Test-Frameworks verwendet werden. Dies wird in Kapitel 2.3.1 beschrieben.

Integrationstest

Der Integrationstest wird verwendet, um die Zusammenarbeit mehrerer Systemteile von einzelnen Modulen über Teilsysteme bis zum Gesamtsystem zu überprüfen. Hierbei werden die korrekte Verwendung von Schnittstellen, Parameter und die Nutzung gemeinsamer Ressourcen getestet. Der Integrationstest wird wie der Modultest in der Entwicklungsumgebung durchgeführt und eignet sich ebenfalls für eine Testautomation. Die Überprüfung der korrekten Interaktion, wie der Austausch von Daten oder der Zugriff auf eine gemeinsame Datenbasis, wird als Testziel definiert (Vgl. Linz und Spillner 2012, S. 52 ff.).

Systemtest

Während des Systemtests wird das Gesamtsystem unter möglichst realistischen Bedingungen überprüft. Es dient ebenso zur internen Vorbereitung auf die Abnahme des Herstellers (Vgl. Baumgartner, et al. 2013, S. 146). Der Systemtest wird in einer Testumgebung durchgeführt, welche der späteren Produktivumgebung möglichst ähnlich sein wird. Auch die benötigten Testdaten werden den späteren Daten ähneln. Die Überprüfung von funktionalen und nicht funktionalen Anforderungen wird als Ziel des Systemtest definiert (Vgl. Linz und Spillner 2012, S. 60 ff.).

Akzeptanztest

Der Akzeptanztest wird auch als Abnahmetest bezeichnet, welcher das Gesamtsystem in der Kundenumgebung testet. Er bildet die Grundlage zur Abnahme des Kunden. Hierbei wird der Fokus auf die Verwendbarkeit des Systems durch den Endnutzer gerichtet. Die nicht funktionalen Anforderungen werden innerhalb dieses Tests überwiegend getestet. Häufig werden hierzu die Endanwender hinzugezogen, um so die vertraglichen Anforderungen testen zu können und ggf. Nachbesserungen vorzubereiten. Die Abnahme des Kunden wird als Ziel des Akzeptanztests beschrieben (Vgl. Linz und Spillner 2012, S. 64 ff.).

2.3 Agile Methoden und Techniken

Wie in den vorhergehenden Kapiteln beschrieben werden Software-Tests für eine qualitativ hochwertige Software benötigt. Insbesondere bei unterschiedlichen Entwicklern ist das frühzeitige Testen einer Software von großem Vorteil. Im Rahmen der Continuous Integration, welches das ständige Bauen, Testen und Analysieren von Software beschreibt, spielen Testmethoden eine entscheidende Rolle (Vgl. Behrendt 2012, S. 13 f.). Im Folgenden werden drei agile Testmethoden beschrieben.

2.3.1 Testautomation mit JUnit

Unter Testautomation wird das Automatisieren von Aktivitäten im Testprozess verstanden. Sie ist eine Begleiterscheinung des agilen Testens (Vgl. Baumgartner, et al. 2013, S. 21). In agilen Projekten, insbesondere in SCRUM-Projekten, machen die kurzen Entwicklungsphasen innerhalb eines Sprints, die Automatisierung von Software-Tests unabdingbar. Durch die Testautomation werden die Aktivitäten der Testadministration, Testfallerstellung, Testdurchführung, Testauswertung und Testdokumentation automatisiert und können dadurch in kurzen Zeitintervallen bewältigt werden. Änderungen am Quellcode können unmittelbar auf seine Auswirkungen getestet werden (Vgl. Baumgartner, et al. 2013, S. 143 ff.). Hierfür werden Automatisierungswerkzeuge, wie das Framework JUnit benötigt.

JUnit ist ein Framework zum Testen von Java-Programmen, welches zur Automatisierung in fast allen Teststufen geeignet ist. Das bedeutet, dass JUnit vom Modul- bis zum Systemtest verwendet werden kann. JUnit wird am häufigsten beim Modul- und Integrationstest verwendet (Vgl. Baumgartner, et al. 2013, S. 147). Ein JUnit-Test wird in der gleichen Programmiersprache wie der zu testende Code geschrieben. Diese Tests können von verschiedenen Autoren geschrieben und abschließend in sogenannten Testsuiten zusammengeführt werden. Mit JUnit werden Testklassen erstellt, die Tests automatisiert durchführen, indem die zu testende Komponente aufgerufen wird und Ergebnisse auswerten und anschließend darstellen (vgl. Baumgartner, et al. 2013, S. 147). JUnit arbeitet mit zwei Ergebnissen. Entweder wird der Test bestanden und wird somit grün oder er wird nicht bestanden und wird daher rot. Zum Schreiben einer Testklasse werden Testfixtures verwendet, welche einen zuvor definierten Vorbereitungscode enthalten. Dadurch wird von JUnit eine Möglichkeit geboten, die Programmierarbeit am jeweiligen Test zu verringern (Vgl. Tamm 2013, S. 60 ff.). Das Framework JUnit ist ein Bestandteil der Entwicklungsumgebung Eclipse, daher können die Tests von den Entwicklern selbst geschrieben und durchgeführt werden. Es ist ein Automatisierungswerkzeug, welches aufgrund seiner übersichtlichen Struktur, besonders für die testgetriebene und die abnahmetestgetriebene Entwicklung geeignet ist und daher insbesondere in agilen Projekten verwendet wird.

2.3.2 Testgetriebene Entwicklung

Testgetriebene Entwicklung ist ein Beispiel für ein grundliegendes Prinzip des Testens. Hierbei wird der Test geschrieben, bevor der Programmcode entwickelt wurde. Bei der testgetriebenen Entwicklung wird Test unter Verwendung von Automatisierungswerkzeugen erzeugt (Vgl. ISTQB 2014, S. 31). Dieser Test beschreibt die Funktion eines Teils des Programmcodes. Er wird nach der Erstellung automatisiert gestartet und schlägt aufgrund des noch fehlenden Programm-

codes fehl. Anschließend wird der Programmcode erstellt und der Test wird automatisiert wiederholt, solange der Programmcode noch nicht einwandfrei funktioniert. Sobald der Test erfolgreich abgeschlossen wurde, wird der Programmcode einen Review unterzogen. Die Veränderung des Quellcodes wird durch erneutes Testen überprüft und angepasst. Dieser Prozess wird in der Abbildung 2 skizziert (Vgl. Tamm 2013, S. 114 ff.).

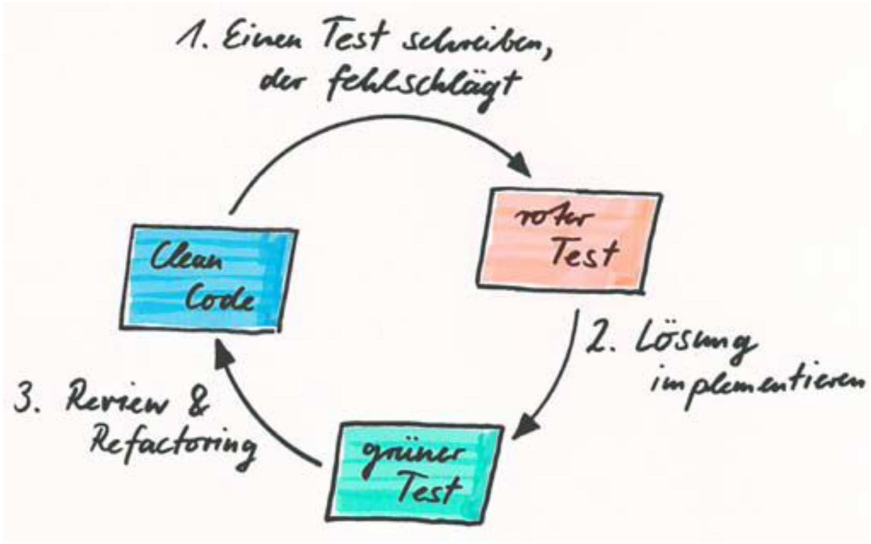


Abbildung 2 - Schematische Darstellung der testgetriebenen Entwicklung
(Entnommen aus: Tamm 2013, S. 91)

Die testgetriebene Entwicklung wird überwiegend auf der Modultestebene durchgeführt, da die geschriebenen Tests auf Modulebene und Code bezogen geschrieben werden.

2.3.3 Abnahmetestgetriebene Entwicklung

Die abnahmetestgetriebene Entwicklung stellt eine Variante der testgetriebenen Entwicklung dar. Die Absicht dieser agilen Methode ist es, verbindliche Akzeptanzkriterien zu entwickeln. Dazu werden vor jeder Iteration, Sprint oder Sprint Planning Meeting durch das gesamte Team, die vom Auftraggeber definierten Akzeptanzkriterien analysiert. Die Analyse wird anhand von konkreten Interpretationsbeispielen besprochen und die Kriterien abgeleitet (vgl. Baumgartner, et al. 2013, S. 121). Die hierbei auftretenden unterschiedlichen Sichtweisen dienen als Diskussionsgrundlage, die in der User-Story festgehalten wird. Auf dieser Grundlage wird ein Akzeptanztest erstellt. Dieser Test soll wie bei der testgetriebenen Entwicklung fehlschlagen. Auf Grundlage dieses Tests werden einzelne Tests abgeleitet, auf deren Basis ein neuer Programmcode entsteht. Ab diesem Zeitpunkt wird das Verfahren wie bei der testgetriebenen Entwicklung durchgeführt. Erst nachdem alle Kriterien des Akzeptanztests erfüllt sind, wird der Prozess beendet (vgl. Tamm

2013, S. 111 ff.). Die abnahmegetriebene Entwicklung wird in Abbildung 3 schematisch beschrieben.

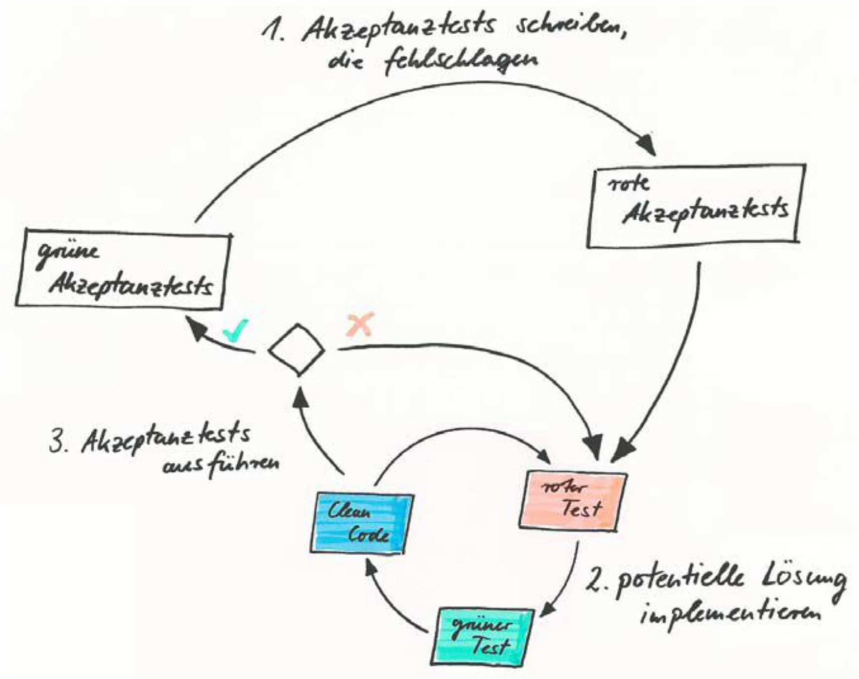


Abbildung 3 - Schematische Darstellung der abnahmetestgetriebenen Entwicklung
(Entnommen aus: Tamm 2013, S. 111)

Die abnahmetestgetriebene Entwicklung ist ein kollaborativer Ansatz, welcher allen Beteiligten ermöglicht, das Verhalten der einzelnen Komponenten und dazu notwendigen Maßnahmen zu verstehen (vgl. ISTQB 2014, S. 31).

Im nachfolgenden Kapitel wird der Fokus auf das konzeptionelle Erstellen von Software-Tests gelegt und beschreibt eine mögliche Vorgehensweise.

3 Konzeptionelle Entwicklung von Testfällen

Die im Kapitel 2.3 beschriebenen Methoden benötigen entsprechend der Entwicklungsstufe eigene Software-Tests. Die Suche nach Testfällen erfolgt explorativ, das bedeutet, dass der Tester vom Bekannten zum Unbekannten, von der Unit über ihre Methoden und ihren möglichen Eingabewerten zu den zu erwartenden Ausgabenwerten gehen wird (Vgl. Westphal 2006, S. 134). Dazu wird der Tester die funktionalen Anforderungen an eine Software in fachlichen Tests wiedergeben. Diese Software-Tests werden auch als Black-Box-Test[1] bezeichnet, das bedeutet, dass die Tests ohne Kenntnisse der inneren Funktionsweise durchgeführt werden. Diese Tests werden durch die Analyse der Funktionsweise abgeleitet (Vgl. Link 2005, S. 67). Diese Testfälle werden für System-, Integrations- und Modultests benötigt. Um einen Testfall zu entwickeln werden die zu programmierenden Abläufe in Form eines Ablaufdiagramms erfasst. Dieses wird zunächst ohne Ausnahmen und Fehlerbehandlungen erfasst, um so den einfachen Kontrollfluss abzubilden (Vgl. Vigenschow 2005, S. 48 f.). Vergleiche Abbildung 4, in der die Aktivitäten an einem Geldautomaten abgebildet werden.

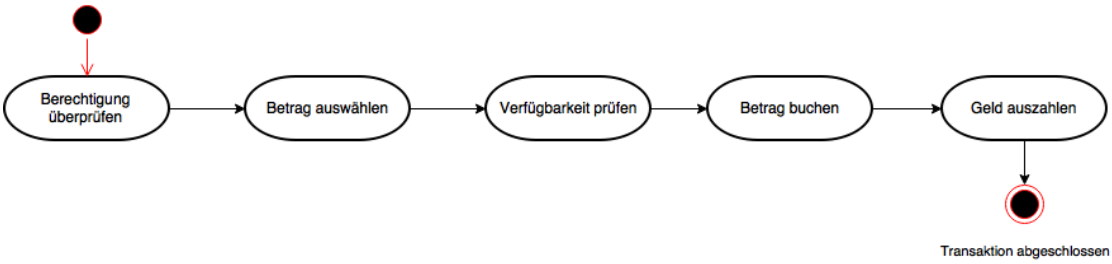


Abbildung 4 - Beispiel eines Anwendungsfalls (In Anlehnung an: Vigenschow 2005, S. 49)

Durch die Abbildung des Kontrollflusses ohne Ausnahmen oder Fehlerbehandlungen werden die ersten Aktivitäten skizziert. Dies stellt den ersten Schritt zur Erstellung eines Software-Tests dar. Um den Software-Test zu erstellen, werden im zweiten Schritt die bereits erwähnten Ausnahmen und Fehlerbehandlung benötigt. Dazu wird das in Abbildung 4 skizzierte Diagramm um die Ausnahmen und Fehlerbehandlungen erweitert, wie in Abbildung 5 dargestellt. Diese Art der Darstellung wird verwendet, damit die fachlichen und technischen Ausnahmen übersichtlich dokumentiert werden (Vgl. Vigenschow 2005, S. 49 ff.). Dieser Vorteil wird bei einer Zusammenfassung in Form einer Tabelle oder eines Fließtextes ermöglicht.

[1] Ist eine Methode, die sich auf das funktionsorientierte Testen beschränkt. Das Bedeutet, dass nur die Anforderungen und Spezifikationen getestet werden. Kenntnisse über die innere Funktionsweise werden nicht benötigt. (Vgl. Vigenschow 2005, S. 34)

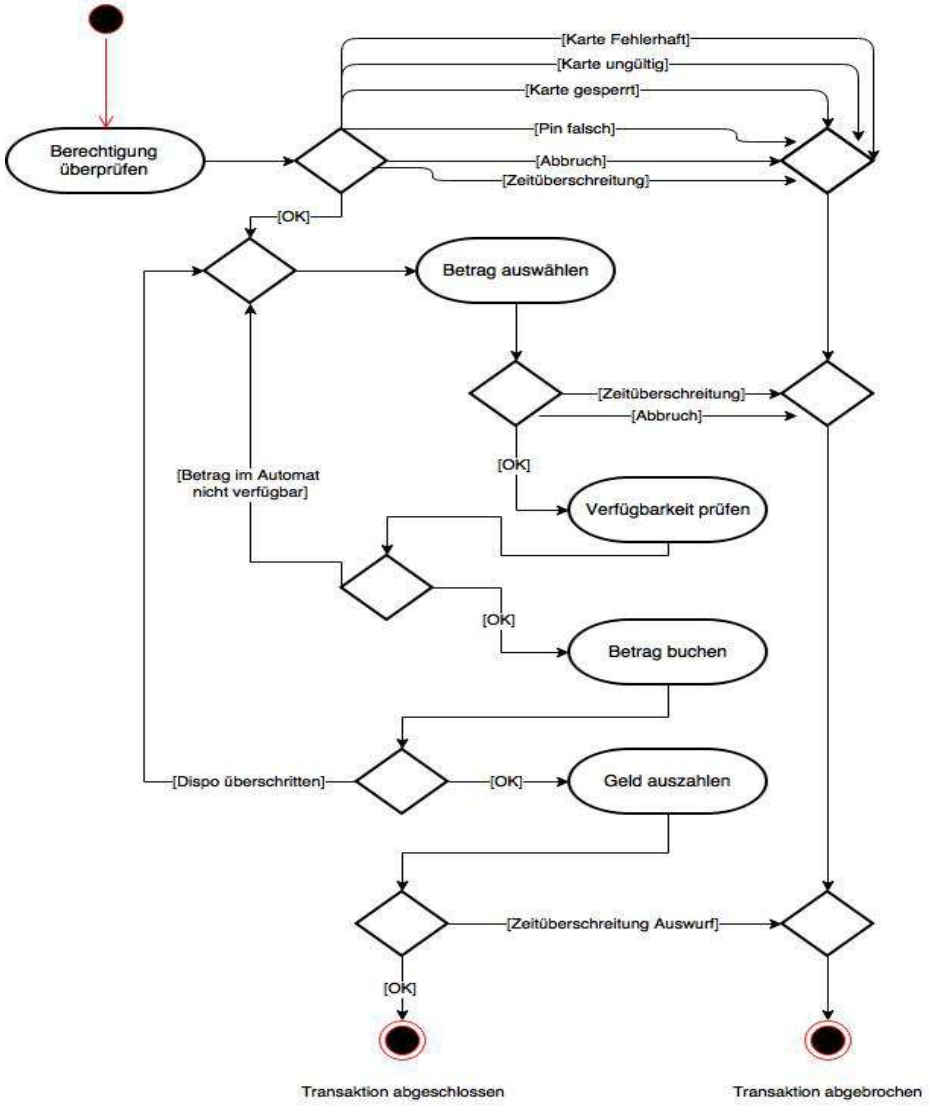


Abbildung 5 - Anwendungsfall erweitert um Varianten und Ausnahmen
(In Anlehnung an: Vigenschow 2005, S. 50)

Anhand dieses Diagramms werden Testfälle für Modultests erstellt, welche als Basis für spätere Integrationstest und Systemtest dienen. Dies wird ermöglicht, indem der Fokus auf die verschiedenen Ausnahmen und Fehlerbehandlung, unter der Prämisse, welche Fehler auftreten könnten, gelenkt wird (Vgl. Vigenschow 2005, S. 50). Um einzelne Testfälle für Modultests abzuleiten, werden die einzelnen Aktivitätsschritte als Methode einer Klasse betrachtet. Es werden für die einzelnen Aktivitäten Testfälle für jede Ausnahme und Fehlerbehandlung geschrieben. Jeder Testfall setzt sich aus der Festlegung der Ausganssituation, Vorgaben der Testdaten, erwartete Ergebnisse und Nachbedingungen zur Fehlerbehandlung zusammen (Vgl. Spillner, et al. 2014, S. 43). Um den Testfall final zu erstellen, werden Testdaten benötigt. Diese können zum einen durch die Bildung von Grenz- und Extremwerten und zum anderen durch Äquivalenzklassen ermittelt werden. Grenz- und Extremwerte werden zur effizienten Testgestaltung verwendet, die Ermittlung der einzelnen Werte werden in agilen Projekten durch das Team in Form von Mindmaps

oder Tabellen durchgeführt und somit visualisiert. Hierbei werden die Informationen für die Testdaten gesammelt. Äquivalenzklassen werden verwendet, um eine hohe Fehlerentdeckungsrate bei einer geringen Anzahl von Testfällen zu gewährleisten (Vgl. Vigenschow 2005, S. 65 ff.). Dazu werden nach der Analyse der Eingabedaten diese Informationen durch Klassifizierung der Wertebereiche, zu Äquivalenzklassen zusammengefasst.

Der vorangestellte Text beschreibt eine Methode, zur Erstellung von Software-Test aus dem Bereich des funktionsorientierten Testens. Im Folgenden Abschnitt werden die Ergebnisse der Arbeit bewertet.

4 Bewertung der Ergebnisse

Die Aufgaben, welche innerhalb des Testmanagements anfallen, werden in agilen Projekten vom gesamten Team koordiniert, durchgeführt und überwacht. Dies ermöglicht eine effiziente Aufgabenteilung, die von allen Mitgliedern getragen wird. Das bedeutet, dass alle Mitglieder des Teams in die wesentlichen Entscheidungen und deren Vorbereitung involviert werden. Bereits in der Testplanung werden grundlegende Maßnahmen für den weiteren Testbetrieb getroffen. Die Auswahl einer agilen Testmethode wirkt sich auf das gesamte Softwareentwicklungsprojekt aus. Die in der Arbeit vorgestellten Testmethoden wurden im Rahmen der Projektgruppe zuvor als wesentliche Methoden identifiziert. Alle drei Methoden verwenden die Automation, um die eigenen Tests effizient zu gestalten. Hierfür wurde das Framework JUnit ausgewählt und beschrieben, welches bereits einen integralen Bestanteil der Entwicklungsumgebung Eclipse, welche durch die Projektgruppe als Standard festgelegt wurde, darstellt. Dies ermöglicht ein schnelles Arbeiten und erleichtert die Erstellung von Software-Tests. Insbesondere die testgetriebene Entwicklung stellt eine Möglichkeit dar, das Testen und Programmieren der Software in ein optimales Verhältnis zu bringen. Dadurch, dass zuerst der Test geschrieben wird und anschließend der Programmcode, ermöglicht dem Projektgruppenteam bereits in einer frühen Entwicklungsphase fehlerfrei zu arbeiten und reduziert die Möglichkeit in einer späteren Entwicklungsstufe wieder zum Anfang zurückkehren zu müssen. Das eigentliche Erstellen eines Testfalls, das heißt die konzeptionelle Entwicklung wird die wesentliche Herausforderung darstellen. Die hierzu im Kapitel 3 vorgestellte Methode bietet zumindest für die sogenannten Black-Box-Tests eine mögliche Herangehensweise. Auf diesem Weg können die Spezifikationen getestet werden. Die praktische Anwendung im Rahmen der Projektgruppe wird zeigen, inwieweit das analytische Vorgehen anwendbar sein wird.

5 Abschließendes Resümee

Das abschließende Kapitel fasst die geleistete Arbeit in kurzer Form zusammen und bietet dem Autor die Gelegenheit ein Resümee zu ziehen, welches die Arbeit abschließt.

5.1 Zusammenfassung der Arbeit

Einleitend mit der Problemstellung und den Zielen dieser Arbeit wurden dem Leser die Bedeutung von Software-Tests und die daraus resultierende Notwendigkeit eines umfassenden Testmanagements aufgezeigt. In den Grundlagen wurde zunächst eine Übersicht des fundamentalen Testprozesses und seiner Inhalte aus Sicht des Testmanagement skizziert. Hierbei wurden die einzelnen Phasen des Prozesses beschrieben und die wesentlichen Punkte der entsprechenden Phase skizziert und deren Besonderheiten in agilen Projekten hervorgehoben. Im weiteren Verlauf dieses Kapitels wurden die grundlegenden Arten von Test beschrieben und deren Testobjekte sowie Testziele aufgezeigt. Im Anschluss wurden drei agile Methoden beschrieben. Hierzu gehörte die Testautomation unter Verwendung des Framework JUnit, welches im Rahmen der Projektgruppe eingesetzt wird. Die Einführung in die Testautomation wurde als Grundlage für die zwei nachfolgenden Methoden, der testgetriebenen- und abnahmegetriebenen Entwicklung, verwendet. Die Auswahl der Methoden wurde zuvor mit der Projektgruppe abgestimmt. Im dritten Kapitel wurde der Fokus auf das Finden und konzeptionelle Erstellen von Tests gerichtet. Hierbei wurde verdeutlicht, wie fachliche Testfälle mittels analytischer Methoden entwickelt werden können. Im weiteren Verlauf dieses Kapitels wurde aufgezeigt, wie auf der Grundlage der analytischen Methoden Testdaten, unter Verwendung von Grenz- und Extremwerte sowie Äquivalenzklassen, abgeleitet werden. Im vierten Kapitel wurden die Untersuchungsergebnisse bewertet und anschließend im letzten Kapitel zusammengefasst. Ein Fazit des Autors zu den einzelnen Punkten, insbesondere zu der Bewertung der Untersuchungsergebnisse schließt diese Arbeit ab.

5.2 Fazit

Mit dem Erstellen dieser Arbeit wurden wichtige Erkenntnisse im Bereich des Testmanagements gewonnen. Zum einen wurde der Testprozess mit den wesentlichen Punkten aufgegriffen und die entscheidenden Phasen dargestellt. Dies wird bei der Planung der Software-Tests innerhalb der Projektgruppe eine entscheidende Hilfestellung sein. Insbesondere die Beschreibung der einzelnen agilen Methoden, welche vor allem für Modul- und Integrationstest verwendet werden, werden in den ersten Sprints eine Möglichkeit bieten das Projekt voranzubringen. Die eigentliche Schwierigkeit für das Projektteam wird nicht das Erstellen der Tests unter Verwendung von JUnit sein, sondern die konzeptionelle Entwicklung eines Software-Tests. Hierbei wird der analytische

Ansatz, welcher im Kapitel 3 beschrieben wird, eine Möglichkeit seins funktionsorientierte Tests zu erstellen. Im Laufe des Projektes werden im Rahmen des Testmanagements weitere Probleme und Unklarheiten entstehen, da diese in Rahmen dieser Arbeit nicht abgebildet werden konnten. Diese Probleme werden aber durch das Projektteam gelöst werden, da diese Arbeit ein solides Fundament darstellt, auf dem aufgebaut werden kann.

Literaturverzeichnis

Baumgartner, Manfred, Martin Klonk, Helmut Pichler, Richard Seidel, und Siegfried Tanczos. *Agile Testing - Der Agile Weg zur Qualität.* München: Carl Hanser Verlag, 2013.

Behrendt, Mario. *Jenkins.* Köln: O`Reilly Verlag, 2012.

ISTQB. *ISTQB Certified Tester - Certified Tester Foundation Level Syllabus.* Bünde: German Testing Board e.V., 2011.

ISTQB. *ISTQB Certified Tester - Foundation Level Extension Syllabus Agile Tester.* Bünde: German Testing Board e.V., 2014.

Link, Johannes. *Softwaretests mit JUnit.* 2. Heidelberg: dpunkt.verlag, 2005.

Linz, Tilo, und Andreas Spillner. *Basiswissen Softwaretest - Aus- und Weiterbildung zum Certified Tester.* 5. Auflage. Heidelberg, 2012.

Spillner, Andreas, Thomas Roßner, Mario Winter, und Tilo Linz. *Praxiswissen Softwaretest - Testmanagement.* 4. . Heidelberg: dpunkt.Verlag, 2014.

Tamm, Michael. *JUnit - Profiwissen.* Heidelberg: dpunkt.verlag, 2013.

Vigenschow, Uwe. *Objektorientiertes Test und Testautomatisierung in der Praxis.* Heidelberg: dpunkt.verlag, 2005.

Westphal, Frank. *Testgetriebene Entwicklung mit JUnit & FIT.* Heidelberg: dpunkt.verlag, 2006.

B.2. Benutzerhandbuch der Projektgruppe IMPACT

Benutzerhandbuch

der Projektgruppe

Vorgelegt von: Daniel Martin Ahlers, Artjom Baranow, Sebastian Beckmann, Vanessa Dering, Holger Eichholz, Jan Fischer, Tobias Kromke, Jessica Schulte, Patrick Smit, Dirk Tesche, Jan Wiesemann

Abgabetermin: 8. April 2016

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Dieses Benutzerhandbuch dient den Anwendern der Innovationsmanagementplattform IMPACT und stellt Informationen zum Umgang mit dem System bereit. Im Handbuch wird hauptsächlich die Bedienung der Plattform beschrieben.
Für tiefer gehende fachliche Fragen steht die Projektdokumentation zur Verfügung, in der eine ausführliche Ausführung der technischen Umsetzung des Systems zu finden ist. Bei Fragen bezüglich Installation steht das Installationshandbuch zur Verfügung.

1.1 Prozessdurchlauf

Die Innovationsmanagement-Plattform IMPACT bietet einem Unternehmen einen ganz einheitlichen Ansatz eines Innovationsprozesses. In vielen Unternehmen spielt das Innovationsmanagement eine untergeordnete Rolle. Die Prozesse sind oft intransparent, langwierig und bieten wenig Raum für gemeinsame Ideenfindung. Hier setzt die Innovationsmanagement-Plattform *IMPACT* an. Die einzigartige Lösung bildet den vollständigen Prozess von der Problemerfassung bis hin zum Feedback ab. Die Anwender haben über kollaborative Methoden die Möglichkeit, gemeinsam an Vorschlägen und Projekten zu arbeiten. Dadurch wird nicht nur das Wissen unterschiedlicher Unternehmensbereiche gebündelt, sondern auch die Qualität der Ergebnisse gesteigert. Die interdisziplinäre Zusammenarbeit wird zusätzlich durch Ansätze aus dem Design-Thinking unterstützt. Durch diese humanorientierte Vorgehensweise sollen benutzerorientierte und zielgruppengerechte Lösungen entstehen. Um die Motivation und die Anreize zur Ideenentwicklung auch langfristig aufrechterhalten zu können, werden Elemente aus dem Gamification-Umfeld eingesetzt. Bewertungs- und Rangsysteme sorgen für einen spielerischen Umgang mit der Software, unterstützen kreative Denkprozesse und erhöhen den Spaß an der Ideengenerierung. Arbeitsfortschritte und Entscheidungen sind für die Mitarbeiter jederzeit transparent gestaltet. Zusammengefasst werden in einem Unternehmen die vorhandenen Innovationspotenziale besser ausgeschöpft und ermöglichen dadurch eine Erhöhung der strategischen Wettbewerbsfähigkeit.
Der Prozess startet mit dem Eintragen einer Challenge. Eine Challenge kann ein Problem, Idee oder Vorschlag darstellen. Die Challenge kann bearbeitet, bewertet und kommentiert werden. Zu dieser kann ein Lösungsvorschlag abgegeben werden. Ein Lösungsvorschlag ist eine konkrete Anregung, um ein Problem zu beheben oder eine Idee umzusetzen.
Aus einem Lösungsvorschlag kann ein Projekt erstellt werden. Ein Projekt ist die Konkretisierung eines Lösungsvorschlags. Es werden innerhalb dieses Prozessschrittes Informationen gesammelt, um dem Lösungsvorschlag bzw. das Projekt umzusetzen, dazu zählen Arbeitsschritte, Projektteam, Gewinne und Anschaffungen. Das Projektteam arbeitet das Projekt weitestgehend aus, um anschließend eine finanzielle Betrachtung, in Form eines

Business Cases, durchzuführen.
Der Business Case wird außerhalb des Systems erstellt und bewertet. Innerhalb des Systems sind grundlegende Informationen vorhanden, um dieses auszuführen. Dazu zählen Nutzerargumentation, Risikobetrachtung und Nutzungsdauer. Die Bewertung des Business Cases kann positiv als auch negativ ausfallen. Fällt die Bewertung positiv aus, so kann ein Feedback zu dem Business Case abgegeben werden. Kommt eine negative Rückmeldung zustande, so ist zum einen möglich, dass der Business Case wieder in die Projektphase zurückgeführt werden kann und zum anderen wird dieser negativ beendet. Auch hier ist es möglich ein Feedback abzugeben.
In einem Feedback können die Themen konkretisiert werden, die der Benutzer als besonders wichtig erachtet. Dadurch kann gewährleistet werden, dass das gesamte Unternehmen aus den bisherigen Ideen und Problemen neues Wissen ableiten kann.
In der Abbildung 1 ist der Prozessdurchlauf grafisch aufbereitet und einsehbar.

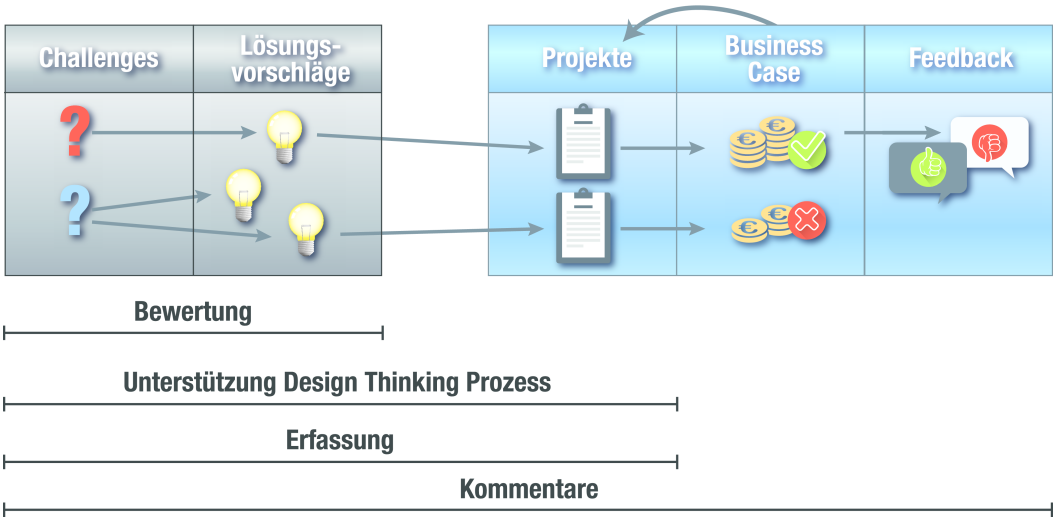


Abbildung 1: Prozessmodell der Innovationsmanagement-Plattform IMPACT

1.2 Einführung IMPACT Weboberfläche

Die Weboberfläche ist generell in drei Bereiche aufgeteilt. Zum einem die Menüleiste (Orange umrandet, vgl. Abbildung 2), die Kopfzeile (Gelb umrandet, vgl. Abbildung 2) und der eigentliche Content-Bereich (Rot umrandet, vgl. Abbildung 2), im dem die verschiedenen Ansichten geladen werden. Während der Content-Bereich sich verändert, bleiben die beiden Bereiche der Menüleiste und Kopfzeile permanent sichtbar.

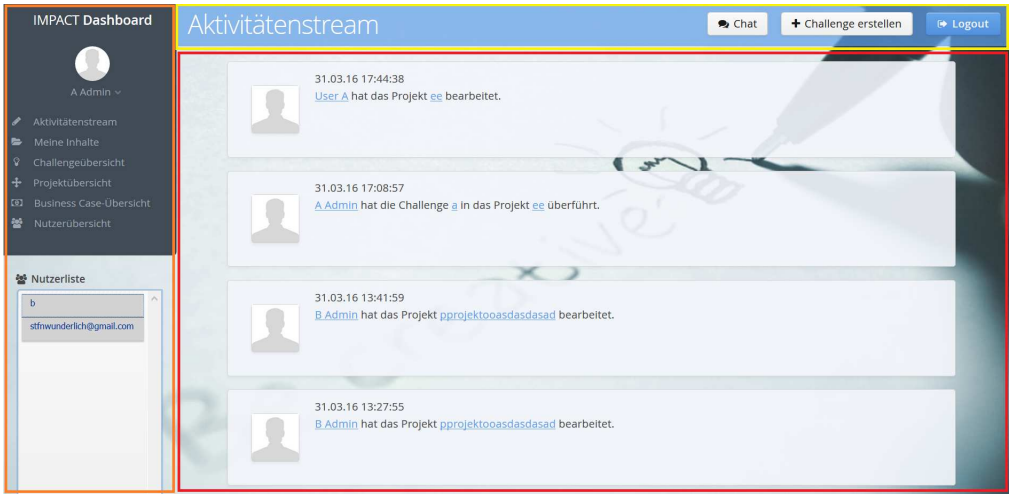


Abbildung 2: Weboberfläche der Innovationsmanagement-Plattform

1.2.1 Menüleiste

Über die Menüleiste können der Aktivitätenstream, das Nutzerprofil, Meine Inhalte, Challengeübersicht, Projektübersicht, Business Case-Übersicht, Nutzerübersicht und eine Nutzerauswahl erreicht werden. Die jeweiligen Ansichten werden nicht weiter erläutert, da diese selbsterklärend sind.

1.2.2 Kopfzeile

In der Kopfzeile sind die Funktionalitäten zum Erstellen einer Challenge, zum Öffnen einer Chatnachricht und zum Logout des Systems, zu finden. Zusätzlich wird in diesem Bereich angezeigt, in welcher Ansicht man sich gerade befindet.

1.2.3 Content-Bereich

In dem Content-Bereich werden die jeweiligen Ansichten geladen, die aufgerufen werden.Lediglich in diesem Bereich findet eine ständige Veränderung statt. Somit können alle Ansichten über die Menüleiste oder über die Kopfzeile aufgerufen werden. Der Aktivitätenstream ist der Start-Content, in dem bei jedem Aufruf neue Aktivitäten innerhalb des Systems bekannt gemacht werden.

2 Benutzerrollen

Die Benutzerrollen in der Innovationsmanagementplattform IMPACT sind weitestgehend an den Prozess angepasst. Jeder Anwender kann mehrere Rollen bekleiden und diverse Funktionen ausführen. In den folgenden Kapiteln werden die möglichen Rollen kurz erläutert und vorgestellt.

2.1 Standardanwender

Generell besitzt jeder Benutzer das Recht eines Standardanwenders. Dieser ist in der Lage andere Profile, Challenges, Feedbacks und Projekte einzusehen. Zusätzlich besitzt er dir Möglichkeit, eine Challenge, Kommentar und Bewertung zu erstellen.

2.2 Projektleiter

Ein Projektleiter kann in dem Erstellungsprozess eines Projektes als Teil eines Projektteams ausgewählt werden. Dieser besitzt die Rechte, die Informationen innerhalb eines Projektes als final zu markieren. Darüber hinaus besitzt der Leiter das Recht, das Projekt in die Business Case-Phase weiter zu leiten.

2.3 Projektteammitglied

Ein Projektteammitglied kann innerhalb eines bestehenden Projektes hinzugefügt werden oder während des Erstellungsprozesses. Ein Mitglied ist in der Lage ein Projekt zu bearbeiten und zu verändern.

2.4 Controller

Die Rolle eines Controllers sollte von einem Anwender übernommen werden, der fundierte Kenntnisse besitzt, um einen Business Case auszuführen. Diese Rolle bearbeitet einen Business Case, um ihn anschließend dem Entscheider zur Bewertung zu übergeben.

2.5 Entscheider

Der Entscheider bewertet den fertiggestellten Business Case und gibt dem System entsprechend Rückmeldung. Dazu steht ihm in der Ansicht *Business Case* ein entsprechendes Feld zur Verfügung. Ebenso besitzt er die Möglichkeit, das Projekt vom Business Case-Schritt wieder in die Projektphase zu leiten.

3 Registrierung, Anmeldung, Abmeldung

Beim Aufrufen der Anwendung IMPACT wird zunächst die Möglichkeit gegeben, sich in dem System anzumelden. Ist noch kein Login vorhanden, muss der Anwender sich zunächst über den Button *Registrierung* registrieren. In der Abbildung 3 ist die Loginansicht abgebildet.

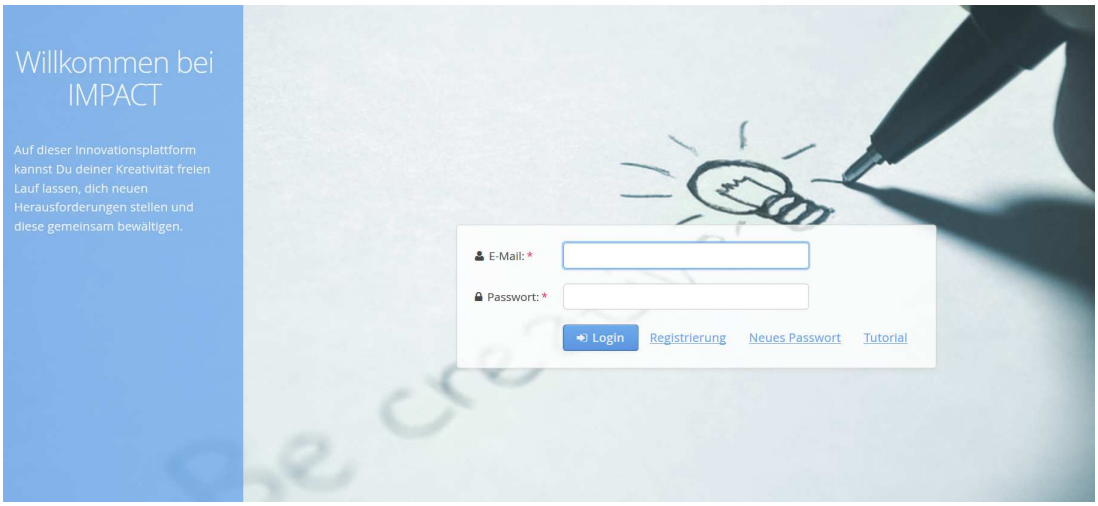


Abbildung 3: Startansicht

3.1 Registrierung

Durch das Betätigen des Buttons *Registrieren* wird man zu der Registrierungsansicht weitergeleitet. Dort müssen die Felder Vorname, Name, E-Mail, E-Mail wiederholen, Passwort und Passwort wiederholen ausgefüllt sein, um sich registrieren zu können. Die E-Mail Adresse wird als Benutzername gewählt. Bedingungen, die erfüllt werden müssen und was sich genau hinter den Eingabefelder, verbirgt, ist in der Tabelle 1 zu finden.

Bezeichnung	Pflichtfeld	Beschreibung
Vorname	Ja	Vorname des Anwenders
Nachname	Ja	Nachname des Anwenders
E-Mail	Ja	E-Mail Adresse des Anwenders. Es ist notwendig ein @-Zeichen in der Adresse zu benutzen
E-Mail wiederholen	Ja	Wiederholen der zuvor eingegebenen E-Mail Adresse
Passwort	Ja	Das Passwort muss 8 oder mehr Zeichen und mindestens eine Ziffer beinhalten
Passwort wiederholen	Ja	Wiederholen des zuvor eingegebenen Passwortes
Abteilung	Nein	Angabe, in welcher Abteilung der Anwender arbeitet
Tätigkeitsbezeichnung	Nein	Bezeichnung des Berufs

Tabelle 1: Felder der Registrierung

Nach dem Registrieren, welches in Abbildung 4 dargestellt ist, wird man wieder zu der Login Ansicht geleitet und kann sich mit seinen zuvor erstellen Login-Daten anmelden.

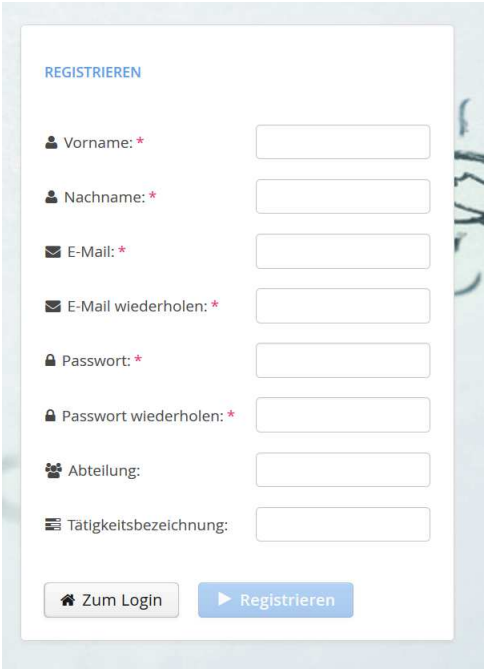


Abbildung 4: Registrierung

3.2 Anmeldung

Die Anmeldung findet über die Ansicht 3 statt. Dafür muss der Anwender seine E-Mail Adresse und das Passwort eingeben. In der Tabelle 2 sind die Felder mit entsprechenden Beschreibungen aufgeführt. Zuvor muss der Anwender jedoch registriert sein. Bei dem erfolgreichen Anmelden wird der Anwender zur Lobby weitergeleitet.

Bezeichnung	**Pflichtfeld**	**Beschreibung**
E-Mail	Ja	E-Mail des Anwenders
Passwort	Ja	Passwort des Anwenders

Tabelle 2: Anmeldefelder

3.3 Abmeldung

Die Abmeldung des Anwenders erfolgt entweder über die Kopfzeile, in dem sich der Button *Logout* befindet oder über das Nutzerprofil, in welchem die Möglichkeit besteht, dass der Anwender sich auszuloggen kann.

4 Persönliche Einstellung

Persönliche Einstellungen werden über das Nutzerprofil vorgenommen. Dort besteht die Möglichkeit, persönliche Informationen zu ändern, zu entfernen und zu erweitern. Über *Menüleiste ->Profil bearbeiten* gelangt der Anwender zu der Profilansicht seines eigenen Profils (vgl. Abbildung 5).

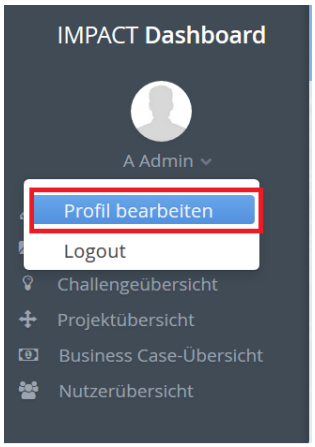


Abbildung 5: Weg zum Nutzerprofil

Zu der Nutzerprofilansicht gelangt der Benutzer ebenfalls über den Weg *Nutzerübersicht ->Benutzer*.
Generell ist die Nutzerprofilübersicht in die Reiter *Allgemeine Informationen*, *Präferenzen* und *Administration* unterteilt. Um die Änderungen an dem Profil zu speichern, muss der *Speicher*-Button getätigt werden.

4.1 Allgemeine Informationen

In dem Reiter *Allgemeine Informationen* sind alle Informationen zu einem Anwender zu finden. Es besteht zudem die Möglichkeit, ein neues Theme zu wählen, so dass die Anwendung ein neues Aussehen bekommt. Darüber hinaus gelangt man von dieser Ansicht zu der Ansicht *Meine Inhalte*. Zusätzlich hat der Anwender die Möglichkeit, das Profilbild zu ändern. Alle Eingabefelder sind in der Tabelle 3 erläutert und aufgelistet.

Bezeichnung	Pflichtfeld	Beschreibung
Vorname	Nein	Vorname des Anwenders
Nachname	Nein	Nachname des Anwenders
Akademischer Titel	Nein	Auswahl des akademischen Titels.
Geburtsdatum	Nein	Eingabe des Datums erfolgt über die Kalenderfunktion oder per direkter Eingabe
E-Mail	Ja	Angabe einer gültigen E-Mail Adresse
Telefonnummer	Nein	mögliche Angabe der Mobiltelefonnummer
Büro-/Raumnummer	Nein	Angabe der Telefonnummer des Arbeitsplatzes
Abteilung	Nein	Bezeichnung der Abteilung
Tätigkeitsbezeichnung	Nein	Bezeichnung des Berufs
Besondere Fähigkeiten	Nein	Angabe von besonderen fachlichen und sozialen Eigenschaften
Projekterfahrung	Nein	Möglichkeit, um bisherige Projekterfahrungen einzutragen
Theme	Nein	Auswahl von weiteren Themes. Ermöglichen es, das Erscheinungsbild der Plattform zu ändern
Meine Inhalte	Nein	Link zu der Übersicht *Meine Inhalte*

Tabelle 3: Felder des Nutzerprofils

Um ein Passwort zu ändern, muss der Button *Passwort ändern* betätigt werden. In der Abbildung 6 ist zu erkennen, dass zunächst das alte Passwort eingegeben werden muss. Anschließend muss das neue Passwort zweimal eingegeben werden. Genauere Beschreibungen der einzelnen Eingabefelder sind in der Tabelle 4 zu finden.

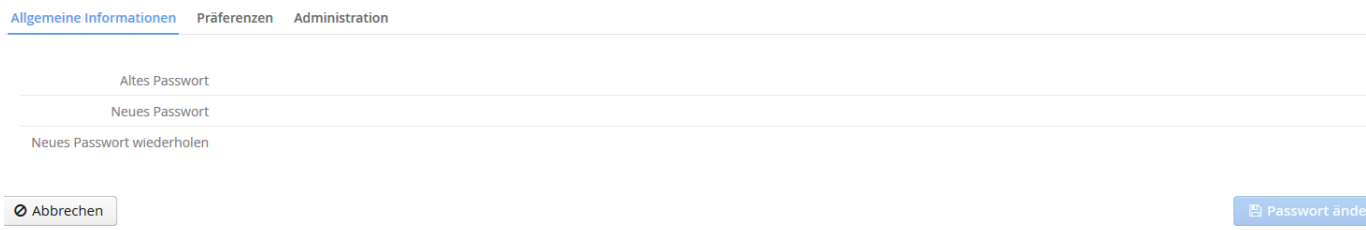


Abbildung 6: Passwort ändern

Bezeichnung	Pflichtfeld	Beschreibung
Altes Passwort	Ja	Eingabe des zu ändernden Passwortes
Neues Passwort	Ja	Das Passwort muss 8 oder mehr Zeichen, Mindestens eine Ziffer beinhalten
Neues Passwort wiederholen	Ja	Das Passwort, welches bereits bei *Neues Passwort* eingefügt wurde, erneut eingeben

Tabelle 4: Felder zum Ändern des Passwortes

4.2 Präferenzen

In dem Reiter *Präferenzen* können mögliche Wünsche eingetragen werden. Dazu zählen z. B. Fortbildungen, Eintrittskarten oder ähnliches. Die Wünsche eines Anwenders können von der Rolle Projektleiter und Entscheider eingesehen werden. Diese Wünsche sollen die Mitarbeitermotivation steigern, in dem diese evtl. erfüllt werden. Durch das Betätigen des *Speicher*-Buttons wird die Wunschliste gespeichert. In der Abbildung 7 ist die Wunschliste zu erkennen.

Abbildung 7: Wunschliste erstellen

4.3 Administration

In diesem Reiter können die Rollen eines Anwenders eingesehen und verändert werden. Das Verändern der Rollen ist jedoch nur einem Anwender mit Administrationsrechten möglich. In der Abbildung 8 ist zu erkennen, dass Rollen hinzugefügt oder entfernt werden können.

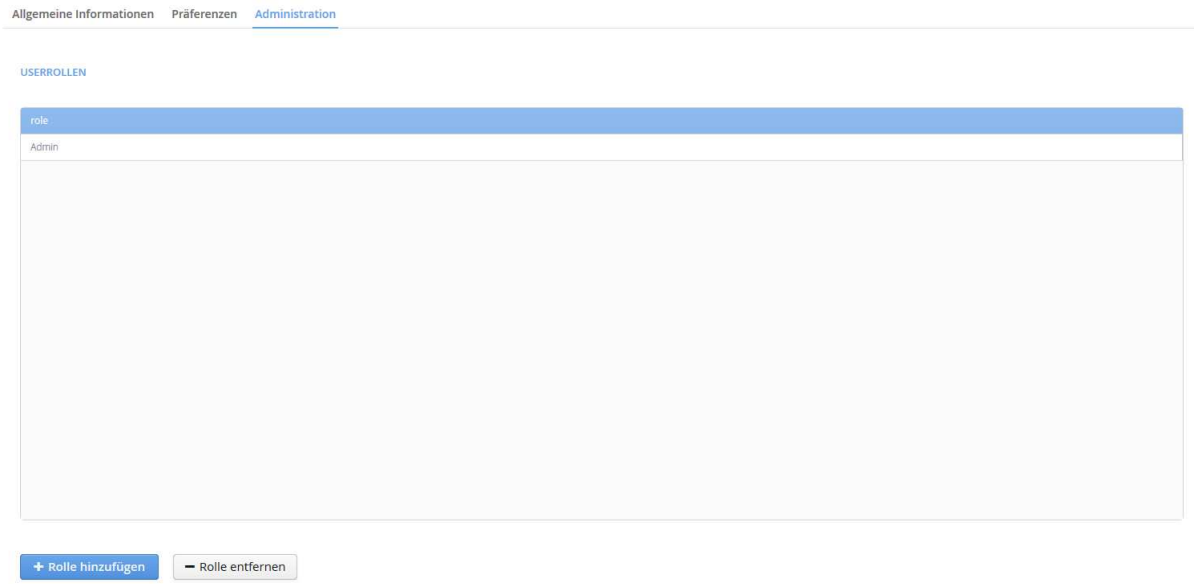


Abbildung 8: Anwenderrollen verwalten

5 Navigationselemente

Die Navigation in der Innovationsmanagementplattform wird über die Menüleiste oder der Kopfzeile geregelt (siehe Kapitel 1.2).

5.1 Aktivitätenstream

Von der Menüleiste wird der Aktivitätenstream über den entsprechenden Button in der Menüleiste erreicht. Im Aktivitätenstream werden alle Aktivitäten innerhalb der Plattform aufgelistet. Um zu der Aktivität an dem Element zu gelangen, muss noch das entsprechende unterstrichende Element ausgewählt werden. In der Abbildung 9 sind die Elemente rot umrandet. Ebenso ist es möglich, sich das Anwenderprofil anzusehen, von dem Anwender welcher die Aktivität ausgeführt hat. In der Abbildung 9 ist das Element orange umrandet.

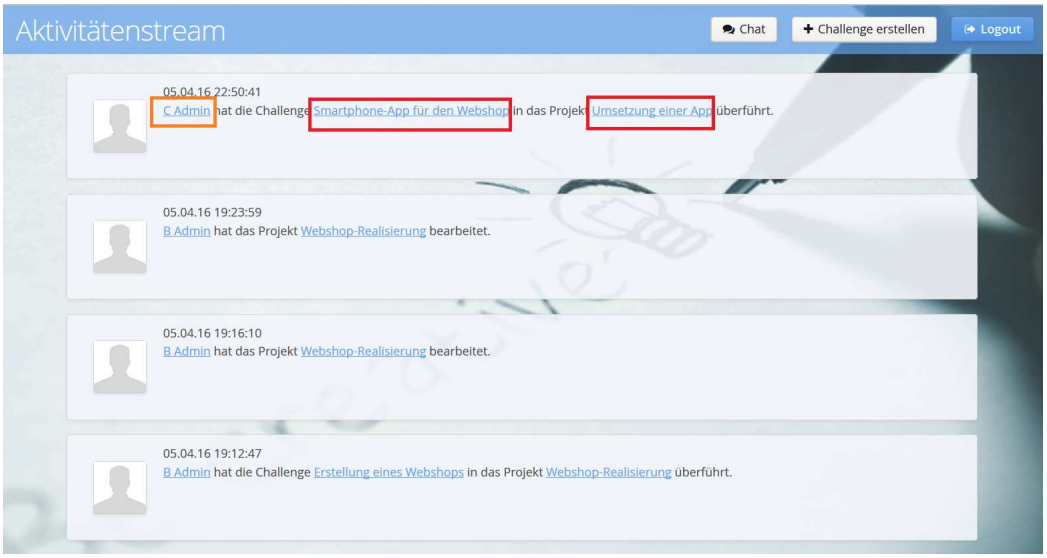


Abbildung 9: Aktivitätenstream

5.2 Meine Inhalte

Um alle Elemente anzuzeigen, die ein Nutzer bearbeiten darf oder erstellt hat, gibt es die Übersicht *Meine Inhalte*. Dort können Challenges, Lösungsvorschläge, Kommentare, Bewertungen und Projekte eingesehen werden. Die Abbildung 10 zeigt die verschiedenen Reiter der Übersicht *Meine Inhalte*. Die Elemente in den entsprechenden Tabellen können anschließend ausgewählt werden, um zu einer detaillierten Einzelansicht zu gelangen.

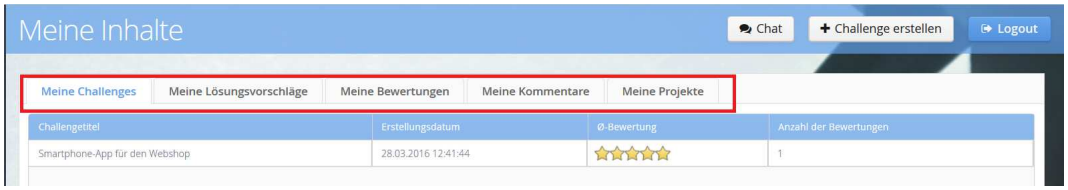


Abbildung 10: Meine Inhalte- Übersicht

5.3 Challengeübersicht

Um einen Überblick aller Challenges innerhalb der Plattform zu erlangen, ist die *Challengeübersicht* notwendig. Alle Challenges sind dort aufgelistet und sind sortierbar. Durch das Klicken auf die Kopfzeile des jeweiligen zu sortieren Attributes wird die Sortierung vorgenommen. In der Abbildung 18 ist die *Challengeübersicht* zu erkennen. Ebenfalls sind archivierte Challenges einsehbar. Diese sind Challenges, welche nicht weiter bearbeitet werden. Sie können aber jederzeit wieder aktiviert werden. Es können alle vorhandenn Challenges eingesehen werden, doch nur die bearbeitet werden, bei denen der Anwender als Ersteller eingetragen ist.

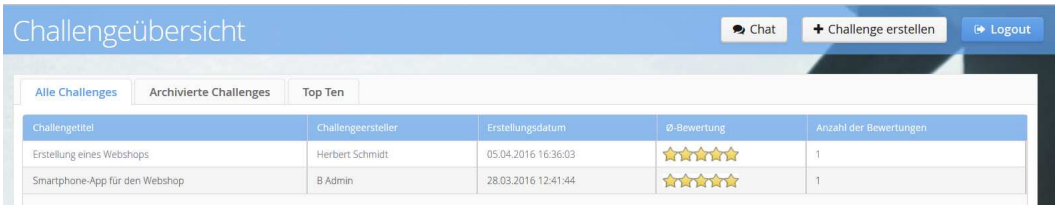


Abbildung 11: Challengeübersicht

5.4 Projektübersicht

Um einen Überblick der vorhandenen Projekte des Systems zu erlangen, ist die *Projektübersicht* vorhanden. Es können ebenfalls alle Projekte eingesehen werden, jedoch ist die Bearbeitung dieser nur möglich, wenn der Anwender die Rolle eines Teammitglieds oder des Projektleiters besitzt. Ebenso sind archivierte Projekte einsehbar. Diese werden zurzeit nicht bearbeitet, können aber jederzeit wieder aktiviert werden. In der Abbildung 12 ist die Übersicht aufgeführt. In dieser Ansicht ist es ebenfalls möglich zu sortieren.