A picture containing shape

Description automatically generatedProbe IPA

Dashboard für SBB go

Autor Winkler Olivier (IT-SWE-CCA)

Firmenname SBB AG

Lehrbetrieb login Berufsbildung AG

Abteilung IT-SWE-CCA

Berufsschule GIBB

Valid-Experte Winkelmann Domenico

Hauptexperte Aebi Joel

Nebenexperte Ritz Anthony

Verantwortliche Fachkraft Ghilardelli Marco

Berufsbildner Weber Fabian

Fachrichtung Applikationsentwickler (API)

Projektvorgehensmodell Hermes 5.1

Jahrgang 2021

Status Abgeschlossen

Version 2.0

A picture containing train, track, platform, station

Description automatically generatedAusgabedatum 29. Januar 2021

**Kurzfassung des IPA Berichts**

**Kurze Ausgangssituation**

**Umsetzung**

**Ergebnis**

Inhaltsverzeichnis

[Teil 1 – Administrativer Teil 6](#_Toc62485614)

[1 Aufgabenstellung 7](#_Toc62485615)

[1.1 Titel der Arbeit 7](#_Toc62485616)

[1.2 Ausgangslage 7](#_Toc62485617)

[1.3 Detaillierte Aufgabenstellung 7](#_Toc62485618)

[1.4 Mittel und Methoden 8](#_Toc62485619)

[1.5 Vorkenntnisse 8](#_Toc62485620)

[1.6 Vorarbeiten 8](#_Toc62485621)

[1.7 Neue Lerninhalte 8](#_Toc62485622)

[1.8 Arbeiten der letzten 6 Monate 8](#_Toc62485623)

[2 SBB Standards 9](#_Toc62485624)

[3 IPA-Schutzbedarfanalyse 9](#_Toc62485625)

[3.1 Informationssicherheit 9](#_Toc62485626)

[3.2 Datenverlust 9](#_Toc62485627)

[3.3 Datenschutz (ISDS) 9](#_Toc62485628)

[4 Organisation der IPA 10](#_Toc62485629)

[4.1 Arbeitsort 10](#_Toc62485630)

[4.2 Arbeitsplatz 10](#_Toc62485631)

[4.3 Datensicherung der IPA 10](#_Toc62485632)

[4.3.1 Dokumentenablage 10](#_Toc62485633)

[4.3.2 Versionierung 12](#_Toc62485634)

[4.3.3 Backup OneDrive 13](#_Toc62485635)

[4.3.4 Bitbucket (Ablage von Quellcode) 15](#_Toc62485636)

[4.3.5 Wiederherstellung von Daten 15](#_Toc62485637)

[5 Detailliertes Projektvorgehen 16](#_Toc62485638)

[5.1 Projektmethode 16](#_Toc62485639)

[5.2 Phasen 16](#_Toc62485640)

[5.2.1 Initialisierung 16](#_Toc62485641)

[5.2.2 Konzept 17](#_Toc62485642)

[5.2.3 Realisierung 17](#_Toc62485643)

[5.2.4 Einführung 17](#_Toc62485644)

[6 IPA Projektorganisation 18](#_Toc62485645)

[6.1 Projektaufbauorganisation 18](#_Toc62485646)

[6.2 Projektrollen 19](#_Toc62485647)

[7 Technische Risikoanalyse 20](#_Toc62485648)

[7.1 Risikograph 21](#_Toc62485649)

[7.1.1 Erkenntnisse aus der Risikoanalyse 21](#_Toc62485650)

[8 Zeitplanung / Meilenstein 22](#_Toc62485651)

[8.1 Phasenfreigabe 22](#_Toc62485652)

[8.2 Meilensteine 23](#_Toc62485653)

[8.3 Zeitplan 24](#_Toc62485654)

[9 Arbeitsjournal 25](#_Toc62485655)

[9.1 Tag 01 – Montag 11.01.2021 25](#_Toc62485656)

[9.2 Tag 02 – Dienstag 12.01.2021 27](#_Toc62485657)

[9.3 Tag 03 – Donnerstag 14.01.2021 29](#_Toc62485658)

[9.4 Tag 04 – Freitag 15.01.2021 30](#_Toc62485659)

[9.5 Tag 05 – Montag 18.01. 2021 32](#_Toc62485660)

[9.6 Tag 06 – Dienstag 19.01. 2021 34](#_Toc62485661)

[9.7 Tag 07 – Donnerstag 21.01. 2021 35](#_Toc62485662)

[9.8 Tag 08 – Freitag 22.01. 2021 36](#_Toc62485663)

[9.9 Tag 09 – Montag 25.01. 2021 38](#_Toc62485664)

[10 Abschlussbericht 39](#_Toc62485665)

[Teil 2 – Individueller praktischer Teil 40](#_Toc62485666)

[11 Initialisierung 41](#_Toc62485667)

[11.1 IST - Situation 41](#_Toc62485668)

[11.1.1 Analyse Webapplikation 41](#_Toc62485669)

[11.2 SOLL – Situation 43](#_Toc62485670)

[11.3 Anforderungen 44](#_Toc62485671)

[11.4 Nichtfunktionale Anforderungen 45](#_Toc62485672)

[11.5 Persönliche Vorgehensziele 46](#_Toc62485673)

[11.6 Projektziele 46](#_Toc62485674)

[11.7 Variantenvergleich 47](#_Toc62485675)

[11.7.1 Kriterien 47](#_Toc62485676)

[11.7.2 Variante 1: Dashboard ohne Library erstellen 48](#_Toc62485677)

[11.7.3 Variante 2: Dashboard mit Library erstellen 49](#_Toc62485678)

[11.7.4 Entscheidung & Begründung 50](#_Toc62485679)

[12 Konzept 51](#_Toc62485680)

[12.1 UseCases 51](#_Toc62485681)

[12.1.1 UseCase: Daten aller Studien anzeigen 51](#_Toc62485682)

[12.1.2 UseCase: Daten einer einzelnen Studie anzeigen 52](#_Toc62485683)

[12.1.3 UseCase: Berührungspunkte nach Kriterien filtern 53](#_Toc62485684)

[12.2 UseCase – Diagramm 54](#_Toc62485685)

[12.3 Systemmodellierung 55](#_Toc62485686)

[12.3.1 Datenbankentwurf 55](#_Toc62485687)

[12.3.2 ERD Diagramm 55](#_Toc62485688)

[12.3.3 System Diagramm 57](#_Toc62485689)

[12.3.4 Komponenten Diagramm 58](#_Toc62485690)

[12.3.5 Klassendiagramm 59](#_Toc62485691)

[12.4 Technische Spezifikationen 60](#_Toc62485692)

[12.4.1 Datenfluss 60](#_Toc62485693)

[12.4.2 REST-Schnittstellen 61](#_Toc62485694)

[12.4.3 Datenbank-Schnittstellen 65](#_Toc62485695)

[12.4.4 Stammdaten 66](#_Toc62485696)

[12.5 Fachliche Spezifikationen 66](#_Toc62485697)

[12.5.1 Rollen & Berechtigungen 66](#_Toc62485698)

[12.6 Mockups 67](#_Toc62485699)

[12.7 Build und Deployment 68](#_Toc62485700)

[12.8 Testkonzept 69](#_Toc62485701)

[12.8.1 Testumgebung 69](#_Toc62485702)

[12.8.2 Testmethode 69](#_Toc62485703)

[12.8.3 Testziele 70](#_Toc62485704)

[12.8.4 Testfälle 70](#_Toc62485705)

[13 Realisierung 73](#_Toc62485706)

[13.1 Systemdarstellung 73](#_Toc62485707)

[13.1.1 Implementation der UseCases 73](#_Toc62485708)

[13.1.2 Stammdaten 77](#_Toc62485709)

[13.1.3 Implementation REST Schnittstellen 77](#_Toc62485710)

[13.1.4 Implementation Frontend 77](#_Toc62485711)

[13.1.5 Implementation Testing 78](#_Toc62485712)

[13.2 Testing 84](#_Toc62485713)

[13.2.1 Testfallübersicht 84](#_Toc62485714)

[13.2.2 Testobjekte 85](#_Toc62485715)

[13.2.3 Testdurchführung / Testergebnisse 89](#_Toc62485716)

[13.2.4 Unittests 91](#_Toc62485717)

[13.2.5 Testfazit 92](#_Toc62485718)

[13.3 Vergleich Vorher / Nachher 92](#_Toc62485719)

[13.3.1 Ansicht Dashboard – Alle Studien 93](#_Toc62485720)

[13.3.2 Ansicht Dashboard – Eine Studie 94](#_Toc62485721)

[13.3.3 Filtern der Touchpointbewertungen 95](#_Toc62485722)

[13.4 Weiteres Vorgehen 96](#_Toc62485723)

[13.4.1 Einführung vorbereiten 96](#_Toc62485724)

[13.4.2 Schutzmassnahmen umsetzten 96](#_Toc62485725)

[14 Selbständigkeitserklärung 97](#_Toc62485726)

[15 Abbildungsverzeichnis 98](#_Toc62485727)

[16 Tabellenverzeichnis 99](#_Toc62485728)

[17 Quellenverzeichnis 100](#_Toc62485729)

[18 Abkürzungsverzeichnis und Glossar 101](#_Toc62485730)

[19 Anhang 102](#_Toc62485731)

# Teil 1 – Administrativer Teil

IPA Projektname: Dashboard für SBB go

Autor: Winkler Olivier (IT-SWE-CCA)

A picture containing train, track, platform, station

Description automatically generatedA picture containing red, accessory, dark, laser

Description automatically generated

# Aufgabenstellung

In diesem Kapitel wird die Aufgabenstellung für die Probe IPA aufgezeichnet.

## Titel der Arbeit

Dashboard für Studienverwaltung Kundenzufriedenheit SBB go

## Ausgangslage

Um die Wichtigkeit von Berührungspunkten der SBB aus Kundensicht und die Zufriedenheit der Kunden sowie die gesamte Customer Journey durch Kunden zu evaluieren und validieren, wurde eine mobile App sowie eine WebApp zur Verwaltung und Analyse entwickelt.

## Detaillierte Aufgabenstellung

Das bestehende Produkt soll um folgende Funktionen erweitert werden.

* Dashboard
* Auf einer neuen Seite muss eine Übersicht in Form eines Dashboards erstellt werden.
* Das Dashboard zeigt dem Benutzer eine geeignete Übersicht der Zufriedenheit der bewerteten und codierten Berührungspunkte einer Studie
* Dem Benutzer wird eine Übersicht der demografischen Daten der Studienteilnehmenden angezeigt. Es sollen die Altersverteilung, Aboverteilung und das Geschlecht über die Studienteilnehmenden angezeigt werden.
* Dem Benutzer wird eine Verteilung der bewerteten gesamten Journeys angezeigt.
* Filter
* Mittels eines Filters kann die Studie ausgewählt werden. Das Dashboard zeigt nur die Daten der ausgewählten Studie an.
* Bei der Übersicht der Zufriedenheit der bewerteten und codierten Berührungspunkte kann nach dem Typ des Berührungspunktes gefiltert werden.
* CI/CD
* Die umgesetzten Features werden mit der vorhandenen CI/CD Pipeline auf der Entwicklungsumgebung ausgerollt.
* Tests
* Test werden gemäss den Beurteilungskriterien erstellt und dokumentiert
* Neben dem IPA-Bericht werden keine weiteren Dokumente erstellt.

## Mittel und Methoden

**Hardware**

Macbook Pro 2013 (15 Zoll & durch SBB gemanaged), 1x Ultrawide Monitor, Arbeitsplatz inklusive Stehpult

**Software**

Intellij IDEA Ultimate Edition für Entwicklung Front- & Backend

Adobe XD als Designtool

**Programmiersprachen / Frameworks**

Datenbank H2 & PostgreSQL

Backend à Java, Spring Boot, Spring Framework, JPA, Mockito

Frontend à Angular, HTML & SCSS, Typescript

**Projektmanagement-Methode**

Für das Projektmanagement wird die Wasserfall Methode nach Hermes 5.1 verwendet.

## Vorkenntnisse

Die verwendeten Technologien (Angular & Spring Boot) hat der Lehrling schon in einigen Projekten verwendet. Das Projekt «SBB go» hat der Lehrling mithilfe seines Teams im Sommer 2020 entwickelt und betreibt seither dieses Projekt aktiv.

## Vorarbeiten

Bevor die Probe IPA gestartet wurde, wurde bereits die Datensicherung auf OneDrive erstellt. Zudem wurden Mockups für das Dashboard erstellt, um so eine visuelle Vorstellung des Endprodukts zu bekommen. Für die Organisation im Team wurde ein Ticket auf JIRA erstellt mitsamt den Anforderungen an das Feature und ein Termin für die tägliche Synchronisation zwischen dem Lehrling und der Fachperson versendet.

## Neue Lerninhalte

Der Lehrling wird sich während der Probe IPA mehrheitlich neu mit dem Testing von Angular beschäftigen. Auch die Verwendung von Diagrammen in Angular ist für den Kandidaten neu. Der Lehrling kann jederzeit auf seine zugeordnete Fachperson bei allfälligen Fragen zugehen.

## Arbeiten der letzten 6 Monate

Der Lehrling ist seit Juni 2020 im KAT Team. Bis und mit September 2020 verbrachte der Lehrling mit zwei Teammitgliedern im Projekt «SBB go». In diesem Zeitraum wurde das Produkt entwickelt. Somit konnte der Lehrling von Anfang an mitwirken und sich so gute Kenntnisse im Projekt aneignen. Die verwendeten Schnittstellen und Tools sind dem Kandidaten bekannt. Nach «SBB go» wurde der Lehrling in ein anderes Projekt eingeteilt namens «RAMSI». Dort konnte er Erfahrungen in Angular sammeln und sein Fachwissen so vertiefen.

# SBB Standards

Die SBB verwendet gewisse Code Conventions, welche zum Teil intern definiert wurden. Für Java hat die SBB eigene Guidelines geschrieben. Für Angular-Applikationen werden die Guidelines von Angular direkt verwendet. Zudem sind momentan für Angular spezifische Guidelines im Aufbau für die SBB. All diese Dokumente befinden sich im Anhang dieses Dokumentes.

# IPA-Schutzbedarfanalyse

Im folgenden Abschnitt wird aufgezeigt, wie die Datensicherung während der Probe IPA durchgeführt wird. Um dies aufzuzeigen zu können, müssen die Daten genauer analysiert werden, um so die Wichtigkeit finden zu können und somit auch der verbunden Schutzwert.

## Informationssicherheit

Alle Informationen (Aufgabenstellung, Dokumente aus PKorg, Kriterienkatalog), welche für die Probe IPA relevant sind, sind bereits im Voraus gespeichert worden. So sind die Dokumente lokal auf dem Arbeitscomputer, als auch auf OneDrive hinterlegt. Diese Ablage ist persönlich und somit auch nur für mich zugänglich ausser ich gebe Personen explizit Berechtigungen.

Informationen aus dem Internet oder anderen Informationsquellen, die während der Probe IPA erlangt werden, werden in ein Quellenverzeichnis festgehalten. In meinem Arbeitsjournal wird festgehalten, welche Fragen ich während der Arbeit an meinen Fachexperten hatte.

In der Realisierung wird der bereits bestehende Code erweitert, da es sich bei meinem Feature um eine Weiterentwicklung handelt. Der bestehende Code und der neu geschriebene Code wird markiert und sorgfältig dokumentiert. So kann garantiert werden, dass alle Informationen gekennzeichnet werden, welche von welche nicht oder nur teilweise von mir stammen.

## Datenverlust

Das Risiko eines Datenverlusts beschäftigt und alltäglich. Um bei einer wichtigen Arbeit wie der IPA dieses Risiko so gering wie möglich zu halten, gehe ich folgendermassen vor:

Alle relevanten Dokumente werden jeweils zweimal pro Tag auf den persönlichen OneDrive Ordner gespeichert. Zusätzlich wird dieses Verfahren auch auf einer externen Festplatte (SSD) durchgeführt. Dank der angewendeten Versionierung, kann jederzeit auf eine vorherige Version zurückgegriffen werden.

Der Quellcode mitsamt der IPA Dokumentation wird mindestens einmal pro Tag auf das GIT-Repository (Bitbucket) geladen. Detaillierte Angaben über die Datensicherung befinden sich im Kapitel *«4.3 Datensicherung der IPA»*

## Datenschutz (ISDS)

Die Applikation «SBB go» verfügt über insgesamt drei Datenbank, für jeden Abschnitt von der kontinuierlichen Integration eine Datenbank. Das Projekt verwendet die Daten von Benutzern und Daten, die durch die Benutzer gesammelt werden. Sensible Daten sind nur wenige Vorhanden. Der Name und die Telefonnummer einer Person werden initial angegeben. Diese Daten müssen geschützt bleiben.

Die Applikation ist nur für zugriffsberechtigte Personen durch ein Login zugänglich. Zudem werden diese sensiblen Daten nach Ablauf der sechsmonatigen Aufbewahrungsdauer nach Studienabschluss automatisch gelöscht. Für die Arbeit meiner IPA sind diese Daten nicht relevant und somit nicht beachtenswert.

# Organisation der IPA

Im untenstehenden Kapitel wird aufgezeigt, wie die Datensicherung im Detail gehandhabt wird.

## Arbeitsort

Aufgrund der momentanen Pandemie befinde ich mich im Home-Office und arbeite während der IPA von zu Hause aus.

Vorgesehener Arbeitsort während der IPA Zeit wäre folgende Adresse:

Hilfikerstrasse 1, 6. OG, Sektor D

3014 Bern

## Arbeitsplatz

Mein Arbeitsplatz befindet sich in meinem Zimmer zu Hause. Für das Arbeiten von zu Hause aus bin ich bestens ausgestattet. Für den ergonomischen Aspekt verfüge ich über ein elektrisch höhenverstellbares Stehpult, um während der Arbeit Abwechslung zu haben. Während der IPA wird mein persönlicher Arbeitscomputer verwendet. Auf diesem sind alle benötigten Tools und Programme bereits installiert, um so möglichst effizient arbeiten zu können. Als Peripherie verwende ich eine kabellose Maus mitsamt Mausmappe. Als Bildschirm verwende ich einen Ultrawide Monitor (34 Zoll) um so möglichst viel Überblick zu haben. Gegen akustische Ablenkungen trage ich während der IPA meistens Kopfhörer. Um möglichst ein konstantes Level der Konzentration zu behalten, lege ich mein persönliches als auch geschäftliches Smartphone zur Seite.

// TODO Bild von Arbeitsplatz

## Datensicherung der IPA

Um einen Datenverlust zu verhindern und somit auch die Gefahr einer verheerenden Kettenreaktion zu minimieren, werde ich alle wichtigen Daten versionieren und auf voneinander unabhängigen Medien speichern.

### Dokumentenablage

Die Dokumente, Diagramme, Bilder und weitere Materialien werden auf dem persönlichen Ordner auf OneDrive der SBB gespeichert. Dies geschieht jeweils mindestens zweimal täglich oder bei einer Änderung an einem Dokument in diesem Ordner. Dank der Cloud kann ich die Dokumente jederzeit und ortsunabhängig aufrufen, editieren und speichern. Die Daten werden zusätzlich lokal auf meinem Computer gespeichert. Um das Risiko eines möglichen Hardwareproblems entgegen zu wirken, werden die Daten auf eine externe Festplatte gespeichert. Als letzte Dokumentenablage wird das GIT-Repository des Projekts verwendet. Zwischen den drei unabhängigen Ablagen wird täglich eine Synchronisation durchgeführt. Einmal täglich wird die Dokumentation hochgeladen. Durch dieses Verfahren ist das Risiko eines möglichen Datenverlustes gering.

// TODO Bild Repo

Graphical user interface, text, application, email

Description automatically generated

Abbildung 1 OneDrive: Datensicherung

Graphical user interface, text

Description automatically generated

Abbildung 2 Macbook: Datensicherung

### Versionierung

Um möglichst wenig Arbeit bei einem Datenverlust zu verlieren, wird pro Halbtag eine Version des Dokuments wie auch des Zeitplans gespeichert. Diese werden mit der Namensgebung «*Vormittag*» und «*Nachmittag*» gekennzeichnet. So kann jederzeit auf jeden halben Tag zurückgegriffen werden.

Graphical user interface, text

Description automatically generated

Text

Description automatically generated

### Backup OneDrive

Graphical user interface, application

Description automatically generatedTable

Description automatically generated with medium confidenceTable

Description automatically generatedUntenstehend sind alle Backups aufgelistet, die während der zehntätigen Prüfungsdauer durchgeführt wurden.

Abbildung 4 OneDrive: Tägliches Backup - Erster Tag

Table

Description automatically generated

Table

Description automatically generated with medium confidence

### Bitbucket (Ablage von Quellcode)

Mindestens einmal pro Tag wird der Quellcode mitsamt Dokumentation auf das GIT-Repository der SBB (Bitbucket) hochgeladen. Dank der Versionierung von GIT kann der tägliche Stand nachverfolgt werden. Durch die jeweilige Commit-Message kann entnommen werden, was sich in diesem Commit befindet und geändert hat. Die Commit-Message wird nach Best Practices geschrieben.

//TODO Bild von Commits

### Wiederherstellung von Daten

Alle verwendeten Dokumente inklusive Quellcode können bei einem Verlust, pro Version von OneDrive oder von GIT wiederhergestellt werden. Daten können auch mit der externen Festplatte wiederhergestellt werden.

Text

Description automatically generatedGraphical user interface

Description automatically generated with medium confidence

# Detailliertes Projektvorgehen

Im folgenden Abschnitt werden das Projektvorgehensmodell und die dazugehörigen Phasen während der IPA erklärt.

## Projektmethode

Die IPA wird mit der Projektmethode Hermes 5.1 durchgeführt. Durch Hermes wird die Projektdauer in Phasen unterteilt. Bei meiner IPA ergibt dies eine Aufteilung in vier verschiedene Phasen. Durch diese Unterteilung ist die Planung der einzelnen Aufgaben genauer, einfacher und effizienter gestaltet, sowie das Zeitmanagement. Mit sogenannten Meilensteinen werden Phasenübergänge markiert und dienen als ein wichtiges Ziel. Die Meilensteine der IPA-Dauer sind im Kapitel «*8.2 Meilensteine*» genauer definiert.

## Phasen

Diagram, timeline

Description automatically generated with medium confidenceWie schon oben erwähnt gibt es in Hermes diverse Projektphasen. Mit jeder abgeschlossenen Phase ist ein weiter Schritt Richtung Ziel getätigt. Jede Phase hat ein spezifisches Produkt, wenn diese abgeschlossen wird. Untenstehend sind die Phasen von Hermes 5.1 aufgelistet.

### Initialisierung

Die erste Phase im Hermes Framework ist die «*Initialisierung*»*.* Diese Phase dient, wie der Name schon sagt zur Initialisierung des Auftrags und analysiert alle dazugehörigen Informationen. Die durch die Analyse bekommenen Projektgrundlagen und der Projektantrag ermöglichen den Entscheid über die Projektfreigabe. Wichtige Inhaltsstoffe der «*Initialisierungsphase*» sind folgende:

* **Situationsanalyse (IST / SOLL)**

Durch diese Analyse kann aufgezeigt werden, welchen momentan Stand das Produkt repräsentiert und die Arbeitsschritte, die zum Endprodukt führen.

* **Variantenvergleich**

Durch die Vielfalt der Lösungswege einen konkreten Ansatz zu finden ist der Variantenvergleich zuständig. Durch den Vergleich kristallisiert sich die beste Variante durch den direkten Vergleich gegenüber anderen Varianten heraus.

* **Risikoanalyse**

Risiken möglichst früh erkennen steht von Vorteil. Durch die Risikoanalyse werden bereits in der ersten Phase des Projekts die Risiken aufgelistet, geschätzt und durch entsprechend aufgezeichneter Massnahmen minimiert.

Durch diese Vorbereitungen ist die Durchführung des Projekts im Detail bekannt.

### Konzept

Die zweite Phase von Hermes ist die «*Konzeptphase*» oder «*Konzept*». In dieser Phase wird das Projekt konzipiert. Basierend auf den Informationen und Entscheidungen der Initialisierung können hier diverse Konzepte und Spezifikationen erarbeitet werden (Technische Spezifikationen, Testkonzept usw.). In diesen Konzepten kann entnommen werden, wie die verschiedenen Aufgaben aufgebaut sind und wie diese umgesetzt werden müssen. Ein Beispiel hierfür wären die UseCases. Diese werden in der Konzeptphase definiert und anhand diesen kann gemessen werden, ob die Funktionalitäten in der Realisierungsphase nach Planung funktionieren und eingeführt wurden. Die Systemmodellierung oder auch «*System-Architektur*» wird ebenfalls im Konzept erarbeitet und bietet einen Überblick auf das ganze System. Auch werden diverse Diagramme (ERM, Klassendiagramm usw.) erstellt, die die Software-Architektur verständlicher machen und visualisieren. Einer der wichtigsten Faktoren ist das Testkonzept. Es schreibt vor, wie die Applikation getestet werden soll und wie die Resultate ausfallen sollten.

### Realisierung

In der «*Realisierungsphase*» wird alles umgesetzt / realisiert. Durch das Wissen, welches in den beiden vorherigen Phasen erarbeitet wurde, sollte dies möglich sein. Mithilfe der UseCases können Funktionalitäten implementiert werden. Der dazugehörende Quellcode und Tests werden fachgerecht geschrieben. In der Realisierungsphase filtert sich eine Kernfunktion von Hermes hinaus. Dank der detaillierten Vorbereitungen in den vorherigen Phasen kann sich das Projektteam in der Realisierung komplett auf die Umsetzung fokussieren. Während der Umsetzung wird der geschriebene Code mit Hilfe des Testkonzepts überprüft und getestet. Zudem kann sichergestellt werden, dass durch das erarbeitet Testkonzept die Funktionalitäten der UseCases erreicht werden.

### Einführung

Die letzte vorgegeben Phase ist die «*Einführungsphase*». Unter der Einführung werden Präsentationen, Übergaben, Schulungen, Benutzerhandbuch schreiben usw. durchgeführt um möglichst ohne bemerkbare Unterbrüche das Produkt dem Kunden zu übergeben und die Inbetriebnahme sauber abzuschliessen.

Für meine IPA besteht keine Anforderung für ein solches Benutzerhandbuch und wird deshalb nicht mitgeliefert.

# IPA Projektorganisation

In diesem Abschnitt wird die Projektorganisation vorgestellt mitsamt den definierten Rollen und Verantwortlichkeiten.

## Projektaufbauorganisation

Aus dem obenstehenden Diagramm kann entnommen werden, dass der Auftraggeber der IPA Schneider Andrea ist. Andrea ist zudem «Product Owner» von SBB go und hat mir den Auftrag für die IPA erteilt. Die Idee eines Dashboards wurde bei einem Brainstorming durch Andrea vorgeschlagen und anschliessend als detaillierten Projektbeschrieb erfasst. Diesen Projektbeschrieb wurde durch Winkelmann Domenico validiert und angenommen. Aebi Joel übernimmt die Aufgaben des Hauptexperten, während Ritz Anthony als Nebenexpert dient. Persönlich werde ich durch die verantwortliche Fachkraft, welche Ghilardelli Marco ist, betreut. Ich selbst habe mehrere Aufgaben. Als Projektleiter plane ich alle meine Aufgaben und versuche meinen Zeitplan einzuhalten, um so Abweichungen auf einem Minimum zu halten. Zudem diene ich als Entwickler, Architekt und Tester der entwickelten Funktion.

Detaillierte Beschreibungen zu den verschiedenen Rollen können unter «*6.2 Projektrollen*» gefunden werden.

## Projektrollen

In folgender Tabelle sind alle Projektrollen und deren Interessen detailliert abgebildet.

|  |  |  |
| --- | --- | --- |
| **Rollen der IPA** | **Beschreibung** | **Interessen** |
| Auftraggeber | Der Auftraggeber definiert und formuliert den Auftrag mitsamt allen Anforderungen | * Endresultat |
| Projektausschuss | Der Projektausschuss dient als Unterstützung des Auftraggebers bei seinen Handlungen | * Endresultat |
| Qualitäts- & Sicherheitsmanager | Während der IPA übernehmen die Experten die Rolle des Qualitäts- und Sicherheitsmanager. Dieser überprüft ob während der geleisteten Arbeit kein Regelbruch gemacht wurde. Die Experten können während der IPA jederzeit eingreifen | * Code * Testing * Dokumentation * Endresultat |
| Projektleiter | Der Projektleiter betreut die Kommunikation zwischen den einzelnen Rollen und koordiniert so das Projektgeschehen | * Erreichen der Meilensteine * Zeitkoordinierung * Produktivität * Teamzusammenhalt * Endresultat |
| Architekt | Der Architekt ist für die Softwarearchitektur zuständig. Er definiert und modelliert alles von einzelner Komponente bis hin zum ganzen System | * Effiziente Architektur * Wartbarkeit * Erweiterbarkeit |
| Entwickler | Der Entwickler ist für die Umsetzung verantwortlich. Dabei müssen alle Anforderungen des Auftraggebers korrekt funktionstätig sein. | * Effiziente Umsetzung * Code Quality * Deployment |
| Tester | Der Tester testet anhand des Testkonzepts alle umgesetzten Anforderungen und koordiniert sich mit dem Entwickler bei Fehlverhalten | * Fehlerfreies Produkt * Qualitätssicherung |

# Technische Risikoanalyse

Diesen Abschnitt wird der Risikoanalyse gewidmet. Diese ist für eine solch wichtige Arbeit wie die individuelle praktische Arbeit notwendig, um Risiken erkennen und dessen Ausmass schätzen zu können. Damit kann für jedes individuelle Risiko eine Massnahme getroffen werden, die das Ausmass im Notfall mindern kann.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Nr.** | **Risikobeschreibung** | **Auswirkung** | **Vor Massnahme** | | | | **Massnahme / Erklärung** | **Nach Massnahme** | | | |
| **W** | **S** | **Risiko** | **Handlungsweise** | **W** | **S** | **Risiko** | **Handlungsweise** |
| 1 | Datenverlust von Dokumentation / Code | * Zeitverlust * Unpünktliche Abgabe | W3 | S3 | MITTEL | Risikominderung | Pro Tag mehrmals Dokumente speichern und Versionierung einhalten   * Dokumentation OneDrive * Dokumentation ext. Festplatte * Dokumentation Bitbucket | W2 | S2 | KLEIN | Risikoakzeptanz |
| 2 | Mangelnde Entwicklung / zu grosse Aufgabe zu entwickeln | * Zeitverlust * Neue Funktion kann nicht produktiv eingesetzt werden nach Ablauf der IPA Probezeit | W4 | S3 | HOCH | Risikominderung | Testkonzept stets einhalten  Fachvorgesetzten bei Blockaden um Hilfe bitten  Zeitplan im Auge behalten  UX vernachlässigen | W3 | S2 | MITTEL | Risikoakzeptanz |
| 3 | Softwareausfall | * Zeitverlust * Unpünktliche Abgabe | W3 | S2 | MITTEL | Risikominderung | Wichtige Unterlagen immer lokal, als auch in der Cloud speichern.  Software auf stabilen Versionen benutzen | W2 | S1 | KLEIN | Risikoakzeptanz |
| 4 | Hardwareausfall | * Zeitverlust * Unpünktliche Abgabe | W3 | S2 | MITTEL | Risikominderung | Ersatzhardware bereit haben | W2 | S1 | KLEIN | Risikoakzeptanz |
| 5 | Schlechte Code Qualität | * Qualität des Produkts nicht gewährleistet * Weiterentwicklung erschwert * Schlechte Qualität muss in Zukunft verbessert werden | W3 | S3 | MITTEL | Risikominderung | Standards der SBB einhalten  Testkonzept befolgen  Code Qualität Prüfungen durchführen | W2 | S1 | KLEIN | Risikoakzeptanz |
| 6 | Erweiterbarkeit eingeschränkt | * Betrieb wird beeinträchtigt * Produkt ist aufwendiger zu erweitern | W3 | S2 | MITTEL | Risikominderung | Neue Funktion gut konzipieren, um so möglichst alles definieren zu können | W2 | S1 | KLEIN | Risikoakzeptanz |
| 7 | Internetausfall | * Zeitverlust * Unpünktliche Abgabe | W3 | S2 | MITTEL | Risikominderung | Ersatzmöglichkeit (Privater Hotspot) bereit halten | W2 | S1 | KLEIN | Risikoakzeptanz |

## Risikograph

Schaden

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S1 | S2 | S3 | S4 |  |
|  | 3 | 2 |  | W4 |
|  | 6  7  4  5  2 | 1 |  | W3 |
| 7  6  3  4  5 | 1 |  |  | W2 |
|  |  |  |  | W1 |

Wahrscheinlichkeit

**Legende**

Vor Massnahme

Nach Massnahme

### Erkenntnisse aus der Risikoanalyse

Mit der Risikoanalyse konnte ich mir die bevorstehenden Risiken visualisieren. Dadurch bin ich mir diesen vertraut und kann gut vorbereitet dagegen vorgehen. Zudem zeigt die Analyse sehr gut auf, welche Faktoren gewisse Auswirkungen hervorbringen können. Durchaus fühle ich mich aber dank dieser Analyse sicher und weiss was im Notfall zu tun ist.

# Zeitplanung / Meilenstein

In diesem Kapitel ist der Zeitplan hinterlegt mitsamt der Erklärung der einzelnen Meilensteine.

## Phasenfreigabe

Eine neue Phase kann erst begonnen werden, wenn die alte Phase durch die Phasenfreigabe von Ghilardelli Marco freigegeben wird. Dies geschieht mit einer Unterschrift.

|  |  |  |
| --- | --- | --- |
| Phasenfreigabe | Datum | Unterschrift |
| Phasenfreigabe Initialisierung | 12.01.2021  09:30 Uhr | Shape  Description automatically generated with medium confidence |
| Phasenfreigabe Konzept | 12.01.2021  16:00 Uhr | Shape  Description automatically generated with medium confidence |
| Phasenfreigabe Realisierung | 15.01.2021  14:30 Uhr | Shape  Description automatically generated with medium confidence |
| Phasenfreigabe Einführung | 25.01.2021  16:00 Uhr | Shape  Description automatically generated with medium confidence |

## Meilensteine

Während der ganzen Projektzeit habe ich mir neun Meilensteine gesetzt. Mithilfe von dieser kann ich garantieren, das Ziel zu erreichen. Somit kann ich kleinere Ziele definieren und verbessern so die Effizienz und das Zeitmanagement.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nr. | Meilenstein | Geplant | Eingetreten | Beschreibung |
| 1 | Projektmanagement fertiggestellt  Freigabe Initialisierungsphase | 12.01.2021  12:00 Uhr | 12.01.2021  09:30 Uhr | Alle umfassenden Anforderungen im Projektmanagement sind fertiggestellt |
| 2 | Initialisierungsphase fertiggestellt  Freigabe Konzeptphase | 12.01.2021  14:30 Uhr | 12.01.2021  16:00 Uhr | Die Initialisierungsphase wurden mitsamt allen Bestandteilen abgeschlossen |
| 3 | Konzeptphase fertiggestellt  Freigabe Realisierungsphase | 15.01.2021  12:00 Uhr | 15.01.2021  14:30 Uhr | Alle benötigten Bestandteile für die Freigabe der Realisierung wurden in der Konzeptphase abgeschlossen. |
| 4 | Entwicklung Backend abgeschlossen | 19.01.2021  09:30 Uhr |  | Alle UseCases sind in der Backendschnittstelle implementiert und können vom Frontend verwendet werden. |
| 5 | Entwicklung Frontend abgeschlossen | 22.01.2021  09:30 Uhr |  | Das Frontend zeigt die gewünschten Daten aus dem Backend mithilfe von den definierten Diagrammen dar |
| 6 | Testkonzept angewendet | 22.01.2021  14:30 Uhr |  | Das Backend als auch Frontend wurde ausführlich mit Unittests getestet.  Beide Instanzen werden zusätzlich durch Integrationstest überprüft. |
| 7 | Technische Dokumentation fertiggestellt | 25.01.2021  12:00 Uhr |  | Die technische Dokumentation wurde sorgfältig vervollständigt. |
| 8 | Realisierungsphase fertiggestellt  Freigabe Einführungsphase | 26.01.2021  09:30 Uhr |  | Die Realisierungsphase ergibt das neue Dashboard für die Analyse. |
| 9 | Projektabschluss | 29.01.2021  12:00 Uhr |  | Das neue Feature funktioniert einwandfrei und die Dokumentation wurde hochgeladen. |

## Zeitplan

# Arbeitsjournal

In diesem Abschnitt wird für jeden einzelnen Arbeitstag ein separates Arbeitsjournal erfasst, welches einen Überblick auf die getätigten Aktivitäten des jeweiligen Tages bringt.

**Legende Markierung**

|  |  |
| --- | --- |
| Markierung | Beschreibung |
| Volle grüne Umrandung | Nicht geplante Vorarbeiten |
| Volle rote Umrandung | Nicht geplante Nacharbeiten |
| Gestrichelte grüne Umrandung | Frühzeitig beendete Arbeiten |
| Gestrichelte rote Umrandung | Verspätete beendete Arbeiten |

## Tag 01 – Montag 11.01.2021

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tätigkeit | Beteiligte Personen | | Geplanter Aufwand (Std) | Effektiver Aufwand (Std) |
| Kickoff mit Marco | Winkler Olivier  Ghilardelli  Marco | | 0.25 | 0.25 |
| Zeitplan erstellt | Winkler Olivier | | 2 | 3 |
| Administrativer Teil schreiben | Winkler Olivier | | 5.5 | 5.5 |
| Arbeitsjournal | Winkler  Olivier | | 0.25 | 0.25 |
| **Total** | | | 8.00 | 9.00 |
| Tagesablauf | | | | |
| Heute war der erste Tag meiner Probe IPA. Als erstes habe ich mich eingerichtet und alle benötigten Dokumente geöffnet. Bevor ich mit der IPA begonnen habe, habe ich mit Marco noch einen kurzes Kickoff-Meeting geführt. Als erstes erstellte ich einen Zeitplan für die Planung während der ganzen IPA. Nachdem ich den Zeitplan fürs erste erledigte, habe ich mit dem ganzen administrativen Teil der IPA begonnen. Ich konnte mich schrittweise an den Vorlagen der Vorgänger orientieren und wusste so was alles auf mich zu kommt. Jedoch bemerkte ich nach dem Mittag, dass die Dokumentationsvorlage der IPA geändert hatte und die technische Risikoanalyse jetzt vor der Initialisierungsphase erstellt werden soll. Durch dieses Malheur musste ich den Zeitplan nochmals überarbeiten. Dies hat mich Zeit gekostet. An jedem IPA Arbeitstag habe ich um 16:00 Uhr einen Abgleich mit meinem Fachexperten Ghilardelli Marco. Dort habe ich ihm noch zwei kleine organisatorische Fragen gestellt. Gegen Ende des Arbeitstages habe ich noch alles Übrige abgeschlossen und das Arbeitsjournal geschrieben. | | | | |
| Reflexion | | | | |
| Heute morgen war ich ein wenig nervös, denn ich wusste nicht genau was mich erwartet. Aber schon nach ein wenig Arbeitszeit schwand diese und ich konnte mich voll auf die IPA fokussieren. Eigentlich hat alles super funktioniert für den ersten Tag, ausser den Fehler mit dem Zeitplan, welcher mir Zeit gekostet hat. Ich bin aber trotzdem motiviert für den morgigen Tag. | | | | |
| Hilfestellung / Erkenntnisse | | | | |
| Frage:  Wer ist Haupt- & Validexperte in der Probe IPA? | | Antwort von Ghilardelli Marco:  Ich kann frei wählen welche Person ich möchte. | | |

## Tag 02 – Dienstag 12.01.2021

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tätigkeit | Beteiligte Personen | | Geplanter Aufwand (Std) | Effektiver Aufwand (Std) |
| Tagesplanung | Winkler Olivier | | 0.25 | 0.25 |
| Projektorganisation – Diagramm erstellt | Winkler Olivier | | 1 | 0.5 |
| Technische Risikoanalyse inklusiv Risikograph | Winkler Olivier | | 1 | 1 |
| Phasenübergabe Initialisierung | Winkler Olivier Ghilardelli Marco | | 0.25 | 0.25 |
| IST / SOLL Analyse erstellt | Winkler  Olivier | | 1 | 1 |
| Vorgehen & Ziele definiert | Winkler Olivier | | 0.5 | 0.5 |
| Funktionale- & Nichtfunktionale Anforderungen definiert | Winkler Olivier | | 0.5 | 0.5 |
| Variantenvergleich erstellt | Winkler Olivier | | 1.5 | 1.5 |
| Zeitplan überarbeitet | Winkler Olivier | | 0 | 1 |
| Phasenübergabe Konzept | Winkler Olivier Ghilardelli Marco | | 0.25 | 0.25 |
| UseCases erstellt | Winkler Olivier | | 1.5 | 1.5 |
| Arbeitsjournal | Winkler Olivier | | 0.5 | 0.5 |
| **Total** | | | 8.00 | 9.00 |
| Tagesablauf | | | | |
| Heute war der zweite Tag meiner Probe IPA. So startete ich heute mit der Tagesplanung. Ich wusste heute ist ein intensiver Tag und es wird viel zu erledigen geben. Das Ziel heute sind zwei Phasenübergänge also von dem Projektstart in die Initialisierung und dann schon in die Konzeptphase. Als ersten Tag habe ich mir die bereits gestern angefangene Projektorganisation der IPA. Diese konnte ich mit dem passenden Diagramm versehen und abschliessen. Das nächste TODO war die technische Risikoanalyse. Diese hat sich gegenüber dem letzten Jahr geändert. Die Anforderungen an diese Analyse sind nur noch auf technischen Risiken basierend. Zeitgerecht konnte ich die Analyse beenden und somit auch die Phase «*Projektstart*». Ghilardelli Marco gab mir dann die Phasenfreigabe für die Initialisierung. Nach einer kurzen Pause startete ich mit der IST / SOLL Analyse. In dem SOLL Zustand konnte ich vieles über das bisherige Projekt beschreiben und dokumentieren. Für den nächsten Schritt, die Definition des Vorgehens und der Ziele, habe ich mir überlegt welche Ziele ich gerne mit dieser Arbeit erreichen möchte und notierte diese dementsprechend. Der nächste Punkt im Zeitplan war die Definition oder funktionalen und nicht funktionalen Anforderungen an meine Implementation. Dort habe ich auch definiert welche Funktionen zwingen notwendig sind und welche einen wenig hohen Wichtigkeitsgrad besitzen. Leider habe ich dann nach diesem Task feststellen müssen, dass sich ein Fehler in meinen Zeitplan eingeschlichen hat. Diese passierte mir relativ weit oben im Plan und so musste ich sehr vieles überarbeiten, welches mich etwa eine Stunde Arbeitszeit gekostet hat. Nach dem ich dieses Malheur behoben habe, konnte ich mit Ghilardelli Marco die Phasenfreigabe für die Konzeptphase zeremonieren und unser alltäglicher Austausch während der Projektarbeit halten. In der Konzeptphase angelangt, konnte ich direkt mit den UseCases starten. Insgesamt habe ich drei UseCase, die ich definieren musste, modelliert. Am Ende des Tages war ich erschöpft und zugleich auch stolz was ich heute alles schaffen konnte. Als letztes habe ich noch das Arbeitsjournal geschrieben und meine Daten sorgfältig gesichert. | | | | |
| Reflexion | | | | |
| Heute wusste ich es würde ein strenger Tag werden. Dies motivierte mich jedoch sogar noch mehr. Grundsätzlich verlief der Tag sehr produktiv bis auf den kleinen Fehler mit dem Zeitplan. Ich konnte am Morgen sehr schnell die Projektstart-Phase abschliessen. Ich wollte unbedingt schon am Morgen damit fertig werden anstatt erst am Mittag. Dies konnte ich durch meinen Fleiss erreichen. Die Initialisierungsphase war für mich eine kurze Phase. Da ich die einzelnen Arbeiten im Zeitplan einzeln geplant habe, bin ich sehr gut vorangekommen und konnte diese Phase bereits in der Hälfte des Nachmittages abschliessen, welches mich sehr erfreute. Die zweite Phasenübergabe mit Ghilardelli Marco lief auch super. Ich konnte direkt in die Konzeptphase einsteigen und somit den Zeitplan optimal einhalten. Ich war stets den ganzen Tag hinweg motiviert. | | | | |
| Hilfestellung / Erkenntnisse | | | | |
| - | | - | | |

## Tag 03 – Donnerstag 14.01.2021

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tätigkeit | Beteiligte Personen | | Geplanter Aufwand (Std) | Effektiver Aufwand (Std) |
| Tagesplanung | Winkler Olivier | | 0.25 | 0.25 |
| Systemmodellierung | Winkler Olivier | | 2 | 3.5 |
| Fachliche & Technische Spezifikationen | Winkler Olivier | | 1.5 | 0 |
| Arbeitsjournal | Winkler Olivier | | 0.25 | 0.25 |
| **Total** | | | 4 | 4 |
| Tagesablauf | | | | |
| Heute war der dritte Tag meiner Probe IPA. Am Donnerstag arbeite ich jeweils nur am Vormittag, da ich am Nachmittag einen Vorbereitungskurs für die Berufsmaturität besuche. Begonnen habe ich wie jeden Tag mit der Tagesplanung. Ich merkte wie viel ich noch in der Konzeptphase machen muss. Mit dem Gedanken den Termin für die Phasenfreigabe nicht zu erreichen, hatte ich ein wenig. Als erstes habe ich dann mit der Systemmodellierung angefangen. Diese besteht nicht aus viel Text aber dafür umso mehr aus Diagrammen. In diesem Kapitel habe ich insgesamt fünf Diagramme erstellt, von ERD über Systemdiagramm, Komponentendiagramm bis hin zum Klassendiagramm. Laut Zeitplan hätte ich auch noch mit den fachlichen und technischen Spezifikationen beginnen sollen. Dies hat mir jedoch nicht gereicht und muss dies am morgigen Tag nachholen. | | | | |
| Reflexion | | | | |
| Heute war ein nicht so erfolgreicher Tag, wenn man den Aufwand gegenüber dem geplanten Aufwand sieht, vergleicht. Bei der Systemmodellierung habe ich mich einfach unterschätzt und mir war nicht bewusst, welchen Aufwand all diese Diagramme erwirtschaften. Trotzdem konnte ich vieles erarbeiten und meine Zeit effektiv nutzen. Ich versuche am Morgen die verlorene Zeit aufzuholen. | | | | |
| Hilfestellung / Erkenntnisse | | | | |
| - | | - | | |

## Tag 04 – Freitag 15.01.2021

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tätigkeit | Beteiligte Personen | | Geplanter Aufwand (Std) | Effektiver Aufwand (Std) |
| Tagesplanung | Winkler Olivier | | 0.25 | 0.25 |
| Systemmodellierung | Winkler Olivier | | 0 | 1 |
| Fachliche & Technische Spezifikationen | Winkler Olivier | | 1 | 1 |
| Mockups | Winkler Olivier | | 0.5 | 0.5 |
| Build & Deployment | Winkler Olivier | | 0.5 | 0.5 |
| Testkonzept | Winkler Olivier | | 2 | 2 |
| Phasenfreigabe & Daily | Winkler Olivier  Ghilardelli  Marco | | 0.25 | 0.25 |
| Frontend & Backend Entwicklung aufgesetzt | Winkler Olivier | | 3 | 2 |
| Arbeitsjournal | Winkler Olivier | | 0.25 | 0.25 |
| **Total** | | | 8 | 8 |
| Tagesablauf | | | | |
| Heute war der vierte Tag der IPA und der letzte dieser Woche. Wie auch die Tage zuvor habe ich mich zuerst um die Tagesplanung gekümmert. Mir war bewusst, dass ich zu den heutigen Tasks auch noch die Nacharbeiten von Gestern dazukommen, kurzgesagt ein hektischer Tag stand bevor. Zuerst habe ich mich natürlich an die Aufholjagt gewagt, die Systemmodellierung musste noch einen Feinschliff bekommen. Als ich nach ungefähr einer Stunde dies abschliessen konnte, habe ich mit dem ersten heute eingeplanten Task angefangen, den fachlichen und technischen Spezifikationen. Dort habe ich wieder einmal Diagramme zeichnen dürfen und habe vieles über die Schnittstellen beschrieben können. Danach habe ich mich noch an die kleinen Änderungen an den Mockups gemacht, da ich die bereits vor der IPA grösstenteils erstellt habe. Auch beim Build & Deployment Prozess habe ich ein Diagramm gezeichnet, welches den Entwicklungsablauf meines Projekts visualisiert. Als letztes in der Konzeptphase stand noch das Testkonzept. Dort habe ich diverse Ziele und meine Spezifikationen definiert. Zudem habe ich für jedes UseCase mindestens ein Testfall. Das Testkonzept ist ein wichtiger Bestandteil der späteren Realisierung und Sicherung der Qualität. Die Phasenfreigabe mit Ghilardelli Marco stand als nächstes auf dem Plan. Zusammen mit Marco habe ich meine bisherige Arbeit angeschaut und mich mit ihm über ein zwei Fragen ausgetauscht. Schlussendlich hat mir Marco die Erlaubnis für die Realisierungsphase gegeben. In der Realisierungsphase angelangt, habe ich direkt mit dem Aufsetzten des Projetks begonnen. Ich habe im Frontend als auch im Backend angefangen alle benötigten Klassen zu erstellen. Nach einiger Zeit bin ich dort fortgeschritten, wenn aber noch nicht ganz fertig. Zum Ende des Arbeitstages habe ich noch das Arbeitsjournal geschrieben und die tägliche Datensicherung durchgeführt. | | | | |
| Reflexion | | | | |
| Persönlich finde ich meine Leistung von heute wieder optimal. Ich konnte die verlorene Zeit von gestern rasch wieder einholen und die ungeplanten Arbeiten vervollständigen. Trotz ein wenig Zeitdruck seitens des Zeitplans habe ich mich gut geschlagen. Den Rest der Konzeptphase konnte ich ohne grössere Störungen und Unterbrechungen beenden. Dank meiner detaillierten und sorgfältigen Arbeit in diesem Abschnitt bin ich bestens auf die Realisierung gefasst. Somit kann ich mich auch gegen grobe Fehler rüsten und so den engen Zeitplan trotz eventuellen Unterbrüchen einhalten. Trotzdem musste ich in der Zeit etwas einschränken. Die Einführung in der Realisierung auf der Applikation habe ich anstatt den drei Stunden nur zwei Stunden lang vollbringen können, da sich die Phasenfreigabe ein wenig verzögert hat. Dank den Rückmeldungen von Ghilardelli Marco habe ich ein gutes Gefühl für das weitere Vorgehen. Nächste Woche wird vor allem viel Code geschrieben. Ich hoffe ich kann meinen Zeitplan sowie die Anforderungen einhalten. | | | | |
| Hilfestellung / Erkenntnisse | | | | |
| Frage:  Kannst du mir Zugriff auf die Azure API geben, damit ich die Rollen dokumentieren kann?  Soll eine Studie direkt über die URL aufgerufen werden können? | | Antwort von Ghilardelli Marco:  Ich glaube für dein UseCase wäre <https://rbt.sbb.ch/> die bessere Lösung  Ja das können wir so machen | | |

## Tag 05 – Montag 18.01. 2021

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tätigkeit | Beteiligte Personen | | Geplanter Aufwand (Std) | Effektiver Aufwand (Std) |
| Tagesplanung | Winkler Olivier | | 0.25 | 0.25 |
| Entwicklungsumgebung aufsetzten | Winkler Olivier | | 0 | 1 |
| Backend Studiendetails | Winkler Olivier | | 4 | 4 |
| Backend Touchpointdetails | Winkler Olivier | | 3 | 4 |
| Backend Filter nach Studie | Winkler Olivier | | 0.5 | 0.5 |
| Backend Filter der Attribute | Winkler Olivier | | 1 | 0.5 |
| Besprechung Marco | Winkler Olivier  Ghilardelli  Marco | | 0.25 | 0.25 |
| Arbeitsjournal | Winkler Olivier | | 0.25 | 0.25 |
| **Total** | | | 8 | 8 |
| Tagesablauf | | | | |
| Heute war der fünfte Tag meiner Probe IPA und somit bin ich genau in der Mitte der Probezeit angelangt. Schön nach IPA Routine habe ich mit der Tagesplanung die neue Woche gestartet. Der Zeitplan in der Entwicklungsphase ist sehr eng getaktet und mir wurde bewusst, welche Arbeiten heute fertig gestellt werden sollen. Heute morgen musste ich aber zuerst noch das Projekt korrekt aufsetzten. Testdaten für mein Feature waren noch nicht implementiert und vervollständigte dies. Nachdem ich die Nacharbeiten der letzten Woche beendet habe, habe ich mit der Weiterentwicklung im Backendbereich angefangen. Ich hatte bei diesem Task ein zwei Denkfehler und bin ständig stecken geblieben. Momentan habe ich einmal eine erste Version beendet, wenn auch dies sich mit dem folgenden Task überlappt hat. Die Zeit konnte ich beim nächsten Task wieder einholen, da die beiden Funktionen sehr ähnlich gestaltet sind und ich dies im ersten Task schon sozusagen erledigt habe. Bei der Fertigstellung der ersten beiden Tasks sind etwa bereits drei Viertel des Tages hinüber gewesen. Rückblickend bin ich nicht so effizient und schnell vorangekommen, wie ich mir das gewünscht hätte. Für die letzten beiden Task konnte ich die Zeit eigentlich einhalten, ausser der letzte, welchem ich mich etwa nur 30 Minuten widmet konnte. Dazwischen hatte ich noch den täglichen Abgleich mit Ghilardelli Marco. Dort habe ich von zwei kleinen Fragen gefragt und meinen bisherigen Stand erzählt. Über meine momentane Ausgangslage im Backend habe ich ihm erzählt und uns ein bisschen darüber unterhalten. Seitens Marco findet er den Anlauf jedoch in Ordnung und so werde ich diese Variante fortführen. Am Ende des Tages habe ich noch dieses Arbeitsjournal geschrieben und die Datensicherung vollbracht. | | | | |
| Reflexion | | | | |
| Heute war der erste richtige Programmiertag. Ich habe mich gefreut das Theoretische umzusetzen. Jedoch merkte ich schnell, dass ich doch noch einige Fragezeichen habe und mir den genauen Anfang noch ein wenig unklar war. Trotzdem habe ich mich an das Backend gewagt. Über den Tag verteilt konnte ich so mehr und mehr an das Ziel der Fertigstellung des Backend arbeiten. Zurückblickend kann ich aber sagen, dass heute nicht so mein Tag war, die Gründe weiss ich selbst nicht ganz genau. Ich hatte Mühe an Stellen, an denen ich mir das nicht gewohnt bin und hatte so Bammel den Zeitplan nicht einzuhalten zu können. Schlussendlich kann sich aber das Resultat am Ende des Tages sehen lassen. Ich muss morgen noch einige Änderungen vornehmen und dann ist das Backend auch schon bereit für das Frontend. | | | | |
| Hilfestellung / Erkenntnisse | | | | |
| Frage:  Kannst du mir helfen bei der Postmankonfiguration für die Authentifizierung? | | Antwort von Ghilardelli Marco:  Momentan funktioniert dies auch nicht bei mir mit PKCE mit den korrekten Daten. Ich führe die Request mit einem Token aus, welches durch das Frontend generiert wird. | | |

## Tag 06 – Dienstag 19.01. 2021

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tätigkeit | Beteiligte Personen | | Geplanter Aufwand (Std) | Effektiver Aufwand (Std) |
| Tagesplanung | Winkler Olivier | | 0.25 | 0.25 |
| Backend Berechnungen | Winkler Olivier | | 1 | 2 |
| Frontend Diagramme | Winkler Olivier | | 4 | 2 |
| Backend & Frontend Filter Touchpointname | Winkler Olivier | | 2 | 4 |
| Besprechung Marco | Winkler Olivier  Ghilardelli  Marco | | 0.15 | 0.15 |
| Arbeitsjournal | Winkler Olivier | | 0.25 | 0.25 |
| **Total** | | | 8 | 8 |
| Tagesablauf | | | | |
| Heute war der sechste Tag der IPA und wie immer habe ich mit der Tagesplanung begonnen. Wie schon gestern hatte ich für heute sehr viel eingeplant. Zudem kommen noch die nicht fertigen Arbeiten zu den heutigen Arbeiten hinzu, die ich erledigen muss. Heute konnte ich aber den Teil des Backends, bei dem ich gestern noch Mühe hatte, abschliessen. Den Filterteil hatte ich noch ausgeschlossen, da ich zuerst meine verfügbare Zeit für das Frontend eingesetzt habe. So konnte ich bereits ein wenig mit dem Frontendteil beginnen und somit auch gerade die Backendfunktionen überprüfen. Bei den Diagrammen habe ich aber momentan noch einen komischen Fehler und muss diesen unbedingt am Donnerstag lösen können. Die restliche Zeit habe ich für die allgemeine Entwicklung in den beiden Teilen gebraucht und für das Dailygespräch mit Ghilardelli Marco. Zuletzt habe ich noch das Arbeitsjournal und die Datensicherung gemacht. | | | | |
| Reflexion | | | | |
| Heute konnte ich viel programmieren. Jedoch war heute ein komischer Tag, da sich sehr komische Fehler in meinen Code eingeschlichen haben. So hat sich meine Motivation über die Zeit in Grenzen gehalten und bin auch nicht optimal vorwärtsgekommen. | | | | |
| Hilfestellung / Erkenntnisse | | | | |
| Frage:  Kannst du mir noch Daten für die Touchpoints geben? | | Antwort von Ghilardelli Marco:  Es ist im Backend bereits ein Skript mit all diesen Daten vorhanden, musst du nur ausführen. | | |

## Tag 07 – Donnerstag 21.01. 2021

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tätigkeit | Beteiligte Personen | | Geplanter Aufwand (Std) | Effektiver Aufwand (Std) |
| Tagesplanung | Winkler Olivier | | 0.25 | 0.25 |
| Frontend Diagramme & Tabelle | Winkler Olivier | | 2 | 3 |
| Backend Touchpointfilter | Winkler Olivier | | 1.5 | 0.5 |
| Arbeitsjournal | Winkler Olivier | | 0.25 | 0.25 |
| **Total** | | | 4 | 4 |
| Tagesablauf | | | | |
| Heute war der siebte Tag und wie täglich habe ich mit der Tagesplanung begonnen. Der Zeitplan ist immer noch eng getaktet und habe heute vor allem noch nicht fertige Arbeiten von Dienstag eingeplant. Im Frontend konnte ich den komischen Fehler endlich lösen und so die Diagramme fertig entwickeln. Danach habe ich mich an das Layout gewagt und prompt gemerkt, dass sich die Schriftgrössen bei einer grossen Skalierung verziehen und die Grösse ändert. Dies muss mich momentan, aber nicht gross interessieren, da ich nichts dagegen machen kann. Nach diesen Arbeiten habe ich noch die Tabelle der Touchpoints angefangen. Diese wird jeodch mit den Filteroptionen sehr komplex, welches mich gehindert hat, diesen Task zu vervollständigen. Im Backend habe ich ein wenig Code erfasst für die Filterung. Zum Schluss des Tages habe ich noch das Arbeitsjournal geschrieben und die Datensicherung vollzogen. | | | | |
| Reflexion | | | | |
| Heute war ein kurzer Tag. Ich konnte nicht allzu viel machen, da ich nur einen Halbtag zur Verfügung hatte. Jedoch konnte ich im Frontend einige grosse Schritte erzielen, sodass ich morgen darauf aufbauen kann. | | | | |
| Hilfestellung / Erkenntnisse | | | | |
| - | | - | | |

## Tag 08 – Freitag 22.01. 2021

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tätigkeit | Beteiligte Personen | | Geplanter Aufwand (Std) | Effektiver Aufwand (Std) |
| Tagesplanung | Winkler Olivier | | 0.25 | 0.25 |
| Frontend Diagramme & Tabelle | Winkler Olivier | | 1 | 3 |
| Frontend Update SBB Library | Winkler Olivier | | 0 | 1 |
| Backend Nachträgliche Berechnungen | Winkler Olivier | | 1.5 | 2.5 |
| Testkonzept anwenden | Winkler Olivier | | 4 | 0 |
| Besprechung Marco | Winkler Olivier  Ghilardelli  Marco | | 0.15 | 0.15 |
| Arbeitsjournal | Winkler Olivier | | 0.25 | 0.25 |
| **Total** | | | 8 | 8 |
| Tagesablauf | | | | |
| Heute war der achte Tag und wie täglich habe ich mit der Tagesplanung begonnen. Die geplanten Tätigkeiten sollten eigentlich schon laut Zeitplan teils abgeschlossen sein, jedoch bin ich durch die kleinen Verzögerungen in der Realisierung hinter den Zeitplan gefallen. Darum kann der heutige Tag eher als Aufholjagt angesehen werden. Am Morgen konnte ich im Frontend noch die kleinen Bugs fixen und die Diagramme verbessern. Leider hat sich ein weiterer Bug eingeschlichen, der mir den Kopf verdrehte. Bei dem Selectdropdown kann man eine Studie auswählen. Dies funktioniert auch so wie es sollte. Jedoch wird die Seite nicht neu geladen, wenn man bereits auf der Detailansicht einer Studie ist und eine andere auswählt. Mit einem Workaround konnte ich das Problem aber lösen, habe dementsprechend aber auch Zeit verloren. Als nächstes wagte ich mich an die Fertigstellung der Tabelle. Dank der Anleitung von der SBB Library konnte ich dies umsetzen, wenn auch nicht gerade pünktlich. Die Unpünktlich kam auf, als ein benötigtes Modul nicht installiert war und ich zuerst noch die Library updaten musste. Bereits vor der IPA hatte ich eigentlich das Projekt auf die neusten Versionen geupdatet um genau solche Situationen zu verhindern, leider hat dies diesmal nicht funktioniert ☹. Nachdem ich die Arbeiten im Frontend endlich abschliessen konnte, habe ich noch die letzten Berechnungen im Backend gemacht und ein bisschen aufgeräumt. Am Ende des Tages bin ich also mit der Entwicklung fertig und versuche nun die verlorene Zeit am Montag einzuholen. Für Montag ist daher vorgesehen, das Testkonzept anzuwenden und die technische Dokumentation fertigzustellen. Als letztes habe ich noch das Arbeitsjournal geschrieben und die Datensicherung gemacht. | | | | |
| Reflexion | | | | |
| Heute war trotz dem Nichteinhalten des Zeitplans ein effektiver Tag. Ich konnte wenigstens die Entwicklung abschliessen. Schlussendlich kann ich sagen, dass es mich genervt hat, immer Zeit zu verlieren bei Kleinigkeiten und so nur Stress zu generieren. Jedoch bin ich zufrieden, dass ich die Entwicklung wenigstens abschliessen konnte und alles einigermassen funktioniert, auch mit einem Workaround. Betreffend Zeitplan kann ich sagen, dass ich glücklicherweise einige Reservestunden eingeplant habe, um genau aus solchen Situationen nochmals glimpflich herauszukommen. Für Montag versuche ich jetzt alles zu testen und die Dokumentation auf Vordermann zu bringen. | | | | |
| Hilfestellung / Erkenntnisse | | | | |
| - | | - | | |

## Tag 09 – Montag 25.01. 2021

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tätigkeit | Beteiligte Personen | | Geplanter Aufwand (Std) | Effektiver Aufwand (Std) |
| Tagesplanung | Winkler Olivier | | 0.25 | 0.25 |
| Backend JUnit Tests | Winkler Olivier | | 0 | 2 |
| Testkonzept anwenden | Winkler Olivier | | 4 | 2 |
| Technische Dokumentation | Winkler Olivier | | 3 | 3 |
| Besprechung Marco & Phasenfreigabe | Winkler Olivier  Ghilardelli  Marco | | 0.15 | 0.15 |
| Arbeitsjournal | Winkler Olivier | | 0.25 | 0.25 |
| **Total** | | | 8 | 8 |
| Tagesablauf | | | | |
| Heute war der neunte Tag der IPA und wie immer habe ich mit der Tagesplanung begonnen. Laut Zeitplan hatte ich heute den Tag für die technische Dokumentation und noch ein bisschen Reserve für Nacharbeiten zur Verfügung. Jedoch habe ich am Morgen bemerkt, dass ich in der Entwicklung die Backendtests vergessen habe und so das Testkonzept nicht vollständig abschliessen kann. So habe ich die ersten zwei Stunden des Tages damit verbracht, Tests für meinen geschriebenen Code zu schreiben. Danach konnte ich mich auf das Testkonzept fokussieren. Im Testkonzept bin ich sehr gut vorangekommen und konnte so diesen Task bereits frühzeitig beenden. Als nächstes habe ich mich an die technische Dokumentation gewagt. Dort habe ich sozusagen die ganze Realisierungsphase in diesem Dokument geschrieben. Im Zeitplan bin ich auch mit diesem Task fertig geworden. Zum Schluss habe ich noch mit Marco das tägliche Meeting abgehalten inklusive der letzten Phasenfreigabe und das Arbeitsjournal geschrieben, inklusive Datensicherung. | | | | |
| Reflexion | | | | |
| Heute war ein produktiver Tag. Trotz den Schwierigkeiten am Anfang bei der Nacharbeitung der Tests im Backend bin ich im Zeitplan geblieben. Zudem konnte ich trotz diesen Verzögerungen im Testkonzept Zeit einsparen, indem ich schneller vorangekommen bin als geplant. Zudem habe ich die komplette technische Dokumentation fertigstellen können. Somit bin ich jetzt mit der Realisierungsphase am Ende. Für die restlichen Tage kann ich mich jetzt voll auf die letzten Baustellen in diesem Dokument fokussieren und alle Kriterien beachten. | | | | |
| Hilfestellung / Erkenntnisse | | | | |
| Frage:  Ist es in Ordnung, wenn ich bereits eine Pullrequest erstelle für das erste Review? | | Antwort von Ghilardelli Marco:  Klar, für ein richtiges Review müssen wir aber noch schauen. | | |

# Abschlussbericht

# Teil 2 – Individueller praktischer Teil

IPA Projektname: Dashboard für SBB go

Autor: Winkler Olivier (IT-SWE-CCA)

A picture containing red, accessory, dark, laser

Description automatically generated

# Initialisierung

In diesem Abschnitt wird die Initialisierung durchgeführt. Mit der Analyse wird ersichtlich in welchen Stand sich das Produkt befindet und wie es in Zukunft aussehen wird, nachdem die Änderungen eingeführt wurden. Unteranderem sind in der Initialisierung die IST SOLL Situation analysiert, Definition von Zielen und Anforderungen und der Variantenentscheid untergebracht.

## IST - Situation

SBB go ist seit September 2020 im produktiven Betrieb in der Abteilung der Kundenzufriedenheit. Das grosse Aushängeschild von diesem Produkt ist eine Mobile App. Die App wird von ausgewählten Personen oder auch Studienteilnehmenden verwendet, um ihre «*Touchpoints*» zu bewerten. Unter Touchpoints versteht man einen Punkt, wo sich der Kunde und die SBB treffen. Ein Beispiel hierfür wäre ein Ticketautomat. Die SBB stellt diese zur Verfügung und wartet diese dementsprechend auch während der Kunde diesen bedient und sich ein Ticket kauft. So sind alle Touchpoints Objekte, welche der Kunde direkt bei seiner Reise verwendet und benutzt. Ein solcher Touchpoint kann der Studienteilnehmer fotografieren und mit einer entsprechenden Bewertung der Studie beifügen. Am Ende der Studie sind so mehrere Reisen der Teilnehmenden vorhanden mitsamt Bewertungen der einzelnen Touchpoints. Im Hintergrund gibt es nebst der Mobile App auch noch eine Web App. Diese dient zur Verwaltung der Studie. Im der Web App beginnt als auch endet die Studie. Ausgewählte Personen der Abteilung Kundenzufriedenheit können im Tool eine neue Studie erstellen, editieren und löschen. Studien können individuell angepasst werden, zum Beispiel ist es möglich die Auswahlmöglichkeiten der Teilnehmenden in der Mobile App einzuschränken. In einem weiteren Abschnitt des Tools befinden sich alle Bewertungen aller Touchpoints. Diese sind in einem Rasterformat aufgelistet und weisen die jeweilige Bezeichnung als auch Bewertung auf. Diese Touchpoints können in ein CSV exportiert und heruntergeladen werden.

### Analyse Webapplikation

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Table

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

// Bilder von meiner Teststudie

## SOLL – Situation

Mit den bereits vorhandenen Kundendaten soll nun eine Art von Dashboard erstellt werden. Zurzeit ist die Web App nur zur Verwaltung der Studien fähig, soll mit dem Dashboard aber auch eine Art Analyse bekommen. So können die Verantwortlichen direkt erste Schlüsse aus der Auswertung der Daten ziehen, ohne diese zuerst über andere Programme hinweg verarbeiten lassen müssen. Die Daten sollen mit Hilfe von diversen Diagrammen benutzerfreundlich und interaktiv dargestellt werden. So soll es dem Benutzer möglich sein, die Daten von allen bisherigen Studien anzuzeigen, als auch von jeder einzelnen Studie individuell. Auch soll es eine Rangliste der einzelnen Touchpoints geben, die auflisten, welche Touchpoints am schlechtesten beziehungsweise am besten bewertet werden.

## Anforderungen

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr.** | **Was** | **MUSS / SOLL** | **Was** |
| 1 | Frontend (Präsentationsschicht) | MUSS | In einem eigenen Abschnitt (Im Header über «Dashboard» erreichbar) eine Übersicht in Form eines Dashboards darstellen, welches interaktiv bedienbar ist |
| 2 | Frontend (Präsentationsschicht) | MUSS | Dem Benutzer soll eine Übersicht der demografischen Daten (Altersverteilung, Aboverteilung und Geschlecht) der Studienteilnehmenden anzeigen |
| 3 | Frontend (Präsentationsschicht) | MUSS | Der Benutzer soll mit Hilfe eines Filters eine einzelne Studie auswählen können  Die Übersicht der Zufriedenheit bei den bewertenden Berührungspunkten soll ebenfalls nach Zufriedenheit sortiert werden können |
| 4 | Frontend (Präsentationsschicht) | MUSS | Das Dashboard ist in einem übersichtlichen und benutzerfreundlichen Design gestalten. Die Bedienung der einzelnen Grafiken sollte selbsterklärend sein |
| 5 | Frontend (Präsentationsschicht) | SOLL | Das Frontend soll eine Fehlermeldung anzeigen, wenn die Studie nicht existiert oder es einen serverseitigen Fehler gibt |
| 6 | Backend  (Verarbeitungsschicht) | MUSS | Das Backend muss die Daten aller Studien und einzelnen zusammensammeln können und an das Frontend schicken |
| 7 | Backend  (Verarbeitungsschicht) | MUSS | Eine Filterfunktion für die Präferenzen des Benutzers sollen bereit gestellt werden |
| 8 | Backend  (Verarbeitungsschicht) | SOLL | Backend soll Abfragen auf Datenbank mit einer geringen Latenz vervollständigen |

## Nichtfunktionale Anforderungen

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr.** | **Was** | **MUSS / SOLL** | **Was** |
| 1 | System | MUSS | Das ganze System ist verpflichtet die geltenden Standards der SBB einzuhalten |
| 2 | System | SOLL | Die Antwortzeiten des Systems sollten so gering wie möglich sein, um ein möglichst hoher Grad der Benutzerfreundlichkeit zu gewährleisten |
| 3 | System | MUSS | Das System muss mit Hilfe von Tests (Testkonzept) alle vordefinierten funktionalen Anforderungen auf ihre Funktionalitäten geprüft werden |
| 4 | System | SOLL | Das System soll keine unerwartenden Fehler an den Benutzer leiten, sondern diese verarbeiten |
| 5 | Frontend (Präsentationsschicht) | SOLL | Das Frontend soll eine Fehlermeldung anzeigen, wenn die Studie nicht existiert oder es einen serverseitigen Fehler gibt |
| 6 | Code | MUSS | Der geschriebene Quellcode muss den vorliegenden Guidelines und Richtlinien entsprechen |
| 7 | Versionskontrolle | MUSS | Alle gemachten Arbeiten müssen auf dem vorgesehen GIT-Branch abgelegt sein |

## Persönliche Vorgehensziele

Untenstehend habe ich persönliche Vorgehensziele definiert, die ich am Ende der IPA Probezeit erreicht haben will:

* **Zeitplan:** Das Projekt wurde fristgerecht fertiggestellt
* **Architektur:** Die Architektur aus der Konzeptphase wurde korrekt umgesetzt
* **Funktionalität (UseCases):** Alle Funktionen wurden implementiert und funktionieren wie erwartet
* **Testing:** Die implementierten Funktionen wurden getestet
* **Dokumentation:** Die Dokumentation ist vollständig und verständlich formuliert

## Projektziele

Nebst den persönlichen Zielen habe ich noch die Projektziele definiert. Nach Fertigstellung des Projekts sind folgende Punkte erreicht:

* **Produktivität:** Das neue Dashboard bietet den Benutzer einen Mehrwert. Die Benutzer können ihre Produktivität mit Hilfe des Dashboards steigern und das Verfahren der Analyse vereinfachen
* **Funktionalität:** Alle vordefinierten Anforderungen wurden eingehalten und wurden Zwecks Einführungsphase produktiv geschalten

## Variantenvergleich

Im folgenden Abschnitt wird mein Variantenvergleich und den schlussendlichen Variantenentscheid aufgelistet.

In der detaillierten Aufgabenstellung für meine IPA wird das Endprodukt referenziert. Es wird definiert, welche Anforderungen erfüllt werden müssen mit meiner Arbeit, um so den Mehrwert dieser Erweiterung erbringen zu können. Jedoch ist die Vorgehensweise der Architektur nicht enthalten. Somit kann ich wählen welchen Ansatz ich verwenden will für die Umsetzung. Um dies möglichst einfach und auf Fakten basierend entscheiden zu können, werde ich einen Vergleich zwischen unterschiedlichen Ansätzen durchführen. So kann sich die beste Variante herauskristallisieren.

### Kriterien

Um einen sinnvollen Vergleich ziehen zu können, brauche ich Kriterien, um die einzelnen Vor- und Nachtteile der Ansätze einstufen zu können. Ich werde die Kriterien jeweils mit einer Bewertung von 0 – 3 bewerten. Somit kann ich einteilen, wie stark sich das Kriterium auf den Ansatz ausprägt. Zusätzlich zu dieser Punktebewertung kommt noch eine Gewichtung hinzu. Diese Gewichtung ermöglicht es mir, einzelnen Kriterien eine höhere Wichtigskeitsstufe zuzuteilen.

|  |  |  |
| --- | --- | --- |
| **Kriterium** | **Gewichtung** | **Beschreibung** |
| Bestehende Komponente | 30% | Sind die benötigten Komponenten bereits vorhanden?  Sind diese mit dem bereits stehenden Projekt kompatibel? |
| SBB Design | 20% | Können mit den benötigten Komponenten die Designvorgaben eingehalten werden? |
| Komplexität | 40% | Ist die Implementation zu komplex?  Kann die Komplexität zu Zeitverlust führen? |
| Kenntnisse | 20% | Habe ich die Kenntnisse dies umzusetzen? |

### Variante 1: Dashboard ohne Library erstellen

Bei der ersten Variante müsste das Dashboard ohne Hilfe einer geeigneten Library umgesetzt werden. Verwenden dürfte ich jedoch die Angular Komponenten Library der SBB, welche intern für jegliche Webapplikation verwendet wird. Somit könnte ich die Designstandard einhalten. Für ein Dashboard könnte es jedoch schwierig werden. Die Library verfügt über einzelne Komponenten, welche hilfreich sein könnten. Die Möglichkeit das Dashboard zu strukturieren wäre vorhanden, sowie Icons und Piktogramme. Für irgendwelche Graphen ist die Library nicht geeignet und es müsste von Grund auf solche gebaut werden. Nebst den Einhaltungen des SBB Designs könnte ich auch selbst entscheiden wie diese Graphen aufgebaut wären und wie die Logik strukturiert wird. Für diese Arbeit, wo der Zeitplan sehr eng getaktet ist, ist diese Variante eher ungeeignet.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Kriterium** | **Gewichtung** | **Bewertung** | **Total** | **Begründung** |
| Bestehende Komponente | 30% | 0 | 0 | Komponente für die Graphen müssten selbst erstellt werden, SBB Library könnte einige Aufgaben übernehmen für die allgemeine Darstellung |
| SBB Design | 20% | 3 | 0.6 | Dank der SBB Library sind die Designvorgaben der SBB bereits im Projekt integriert. Bei Entwicklung der Graphen aus eigner Hand kann so auf diese zurückgegriffen werden |
| Komplexität | 40% | 1 | 0.4 | Das Erstellen der Graphen aus eigener Hand erhöht die Komplexität enorm und kann daher zu Zeitverlust führen |
| Kenntnisse | 20% | 2 | 0.4 | Kenntnisse der verwendenden Technologien sind vorhanden, jedoch müsste ich mich informieren, wie die Graphen am sinnvollsten konstruiert werden würden |
| **Gesamtbeurteilung** | **100%** | **6** | **1.40** |  |

### Variante 2: Dashboard mit Library erstellen

Bei der zweiten Variante könnte ich eine beliebige Library verwenden. Diese Library würde dann zusätzlich mit der bereits bestehenden Angular Library verwendet werden. Die Library, welche verwendet werden würde, wäre ApexCharts. Durch die Verwendung dieser könnten die aufwändige Darstellung und Logik der Graphen in dieser Library durchgeführt werden. Somit wird sich dies positiv auf den Projektverlauf auswirken, da die gewonnen Zeit der Qualität des Produkts gewidmet werden kann. Nebst der Zeit kann auch die Wartbarkeit erleichtert werden. Der Nachteil bei der Verwendung dieser Library wäre die Einschränkung. Es gibt nur die bestehenden Möglichkeiten und persönliche Präferenzen müssten wegfallen. Das Kriterium Design müsste bei der Library ein wenig umgebaut werden. Die verwendeten Farbdesigns der Library entsprechend nicht denen der SBB, dies kann jedoch mit Hilfe der Angular SBB Library und den verwendeten Technologien behoben werden. Diese jedoch im Verhältnis kleine Anpassung sollte dennoch gegenüber der ersten Variante vorgezogen werden, da der Zeitverlust nicht so gravierend wie bei der ersten Variante wäre.

// TODO <https://apexcharts.com/>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Kriterium** | **Gewichtung** | **Bewertung** | **Total** | **Begründung** |
| Bestehende Komponente | 30% | 2 | 0.3 | Das Zusammenspiel zwischen der SBB Library und ApexCharts funktioniert einwandfrei. |
| SBB Design | 20% | 1 | 0.2 | Die SBB Library liefert die Designstandards mit. Die Komponenten von ApexCharts müssten bei der Farbwahl der Graphen angepasst werden, da diese von den Standards abweichen. Dieses Vorgehen sollte jedoch den Projektverlauf nicht hindern |
| Komplexität | 40% | 2 | 0.8 | Die Komplexität ist deutlich geringer bei Verwendung von ApexCharts, da die ganze Logik der Graphen bereits dort implementiert ist. Dadurch kann Zeitverlust minimiert werden |
| Kenntnisse | 20% | 2 | 0.4 | Die SBB Library kenne ich bestens und müsste mich bei der ApexCharts – Library jedoch noch ein wenig einlesen |
| **Gesamtbeurteilung** | **100%** | **7** | **1.70** |  |

### Entscheidung & Begründung

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Kriterium** | **Gewichtung** | **Variante 1** | | **Variante 2** | |
| **Bewertung** | **Gesamt** | **Bewertung** | **Gesamt** |
| Bestehende Komponente | 30% | 0 | 0 | 2 | 0.3 |
| SBB Design | 20% | 3 | 0.6 | 1 | 0.2 |
| Komplexität | 40% | 1 | 0.4 | 2 | 0.8 |
| Kenntnisse | 20% | 2 | 0.4 | 2 | 0.4 |
| **Total** | **100%** | **6** | **1.40** |  | **1.70** |

Die obenstehende Tabelle zeigt das Endresultat der beiden Varianten. Wie man unschwer erkennen kann, hat sich die zweite Variante besser bewährt als die erste. Die zweite Variante hat gegenüber der Ersten einige Vorteile mehr. Bestehende Komponente der SBB Library, als auch der von ApexCharts kann ich verwenden. Somit kann ich die Realisierungsphase besser bestreiten und die Qualität des Produkts steigern.

# Konzept

Folgend wird nun die Konzeptphase meins Projekts vorgestellt. Dabei wird ersichtlich mit welchen Konzepten das Projekt in der Realisierung implementiert wird. Diese Konzepte strecken sich von UseCases bis hin zu Klassendiagrammen und Architekturmodellierungen.

## UseCases

In diesem Teilabschnitt werden alle für das Projekt benötigten UseCases im Detail beschrieben.

### UseCase: Daten aller Studien anzeigen

|  |  |
| --- | --- |
| **Anwendungsfall Nummer 01 – GET ALL STUDIES** | |
| Beschreibung | **Datenbank:**  Alle benötigten Daten sind für diese Abfrage bereits in der Datenbank hinterlegt. Bei diesem UseCase werden mit Hilfe von JOIN-Queries alle benötigten Daten in einem Paket an das Backend geschickt.  **Backend:**  Das Backend verarbeitet die Daten nach dem erfolgreichen Abfragen auf die DB. Die Daten werden in sogenannte «Data Transform Object» - Objekte umgewandelt und schliesslich an das Frontend geschickt.  **Frontend:**  Das Frontend zeigt die Daten in diversen visuellen Graphen dar. Es gibt drei interaktive Diagramme, welche bedient werden können. Zusätzlich gibt es kleinere Übersichten zu Statistiken der Studiendaten. Unterhalb befindet sich eine Tabelle mit allen bewertenden Berührungspunkten. |
| Benutzer | Admin von SBB go (Personen von Abteilung Kunde) |
| Vorbedingungen | Benutzer muss sich in Web Applikation eingeloggt haben  Backend sortiert Daten aus Datenbank und schickt diese ans Frontend |
| Ablauf | 1. Benutzer navigiert auf die Dashboardseite («Dashboard» in Header)  2. Daten werden bei Besuch der Seite geladen und angezeigt |
| Alternative Abläufe | 1. Benutzer navigiert mittels URL («/dashboard») auf die Dashbaordseite |

### UseCase: Daten einer einzelnen Studie anzeigen

|  |  |
| --- | --- |
| **Anwendungsfall Nummer 02 – GET STUDY BY ID** | |
| Beschreibung | **Datenbank:**  Alle benötigten Daten sind für diese Abfrage bereits in der Datenbank hinterlegt. Bei diesem UseCase werden die Daten von einer einzelnen Studie zusammengesucht und an das Backend geschickt.  **Backend:**  Das Backend bekommt vom Frontend eine spezifische Studie und schickt diese an die Datenbank. Mit den sortierten Daten, verarbeitet das Backend die Studie in ein «DTO» und schickt dies an das Frontend.  **Frontend:**  Der Benutzer kann im Frontend im Dropdown eine spezifische Studie auswählen, zu dieser er die genauen Statistiken will. Nur diese werden dann angezeigt. Die ausgewählte Studie wird an das Backend geschickt, wo die Datenabfrage weiterverarbeitet wird. |
| Benutzer | Admin von SBB go (Personen von Abteilung Kunde) |
| Vorbedingungen | Benutzer muss sich in Web Applikation eingeloggt haben  Benutzer muss sich in der Dashboardansicht befinden. |
| Ablauf | 1. Benutzer navigiert auf die Dashboardseite («Dashboard» in Header)  2. Daten werden bei Besuch der Seite geladen und angezeigt  3. Der Benutzer wählt im Dropdown eine spezifische Studie aus |
| Alternative Abläufe | Keine – Benutzer muss immer zuerst auf die Dashboardansicht |

### UseCase: Berührungspunkte nach Kriterien filtern

|  |  |
| --- | --- |
| **Anwendungsfall Nummer 03 – SORT TOUCHPOINTS** | |
| Beschreibung | **Backend:**  Das Backend bekommt die Anfrage mit den Suchparametern. Die Daten der Datenbank werden nochmals geholt und danach nach den Suchparametern sortiert. Schliesslich wird alles wieder in ein «DTO» verpackt und zurückgeschickt.  **Frontend:**  Der Benutzer kann im Frontend in der Tabelle der Touchpoints diese nach dessen Bewertungen sortieren. Zusätzlich kann er direkt nach einem Namen suchen / sortieren. Bei beiden Aktionen wird eine Abfrage an das Backend geschickt mit den jeweiligen Suchparameter. |
| Benutzer | Admin von SBB go (Personen von Abteilung Kunde) |
| Vorbedingungen | Benutzer muss sich in Web Applikation eingeloggt haben  Benutzer muss sich in der Dashboardansicht befinden. |
| Ablauf | 1. Benutzer navigiert auf die Dashboardseite («Dashboard» in Header)  2. Daten werden bei Besuch der Seite geladen und angezeigt  3. Der Benutzer sortiert in der Touchpointtabelle nach der schlechtesten / besten Bewertung oder sucht einen Berührungspunkt direkt bei Namen. |
| Alternative Abläufe | Keine – Benutzer muss immer zuerst auf die Dashboardansicht |

## UseCase – Diagramm

Folgende Abbildung zeigt das UseCase Diagramm. Aus diesem kann man die möglichen Funktionen der Applikation erkennen und die daraus resultierende weiterführende Funktion, die durch den Benutzer aktiviert werden können.

Diagram

Description automatically generated

Abbildung 5: UseCase - Diagramm

## Systemmodellierung

In diesem Kapitel sind alle wichtigen Informationen enthalten, wie das System modelliert wird.

### Datenbankentwurf

Die Anforderungen an das Dashboard basieren auf bereits vorhandenen Datensätzen. Jedoch müssen die Daten mittels Backend zusammengeholt werden, da diese nicht alle in derselben Tabelle enthalten sind. Da dies sich aber nicht auf die Datenbank ausdehnen wird, wurde dies nicht im Variantenentscheid beachtet. Die gesamte verwendete Tabellenstruktur wird in einem ERD Diagramm unter Kapitel «*12.5.2 ERD Diagramm*» verdeutlicht.

### ERD Diagramm

Wie schon im vorherigen Abschnitt erwähnt, wird für die Neuerung kein neuer Datenbankeintrag benötigt. Jedoch wird eine Tabelle verwendet mitsamt ihren drei Untertabellen. Folgend wird nun erklärt, was die Aufgabe der einzelnen Tabellen ist.

// <https://www.ibm.com/support/knowledgecenter/en/SS9UM9_9.1.0/com.ibm.datatools.transform.ldm.uml2.doc/topics/rldm2uml_type_map.html>

Touchpoint

Journey

Coding

JourneyReasons

Diagram

Description automatically generated

Abbildung 6: ERD der verwendeten Tabellen

### System Diagramm

Die Webapplikation von SBB go verwendet alle Daten aus der eigenen Datenbank. Die Applikation wird bereits auf Openshift mit Hilfe von «*AWS*» betreiben. Die dazukommenden Funktionen werden per Ende der Realisierungsphase und der Einführungsphase auf Openshift gebracht und ausgerollt. Detaillierte Informationen über den Build und Deployment Prozess sind unter «*12.9 Build und Deployment*» beschrieben.

Diagram

Description automatically generated

Abbildung 7: Diagramm Systemmodellierung

### Komponenten Diagramm

Das untenstehende Komponenten Diagramm zeigt die Kommunikationsabläufe zwischen den einzelnen Systemen. Auch ist die verwendende Schnittstelle der Komponenten aufgezeichnet.

Diagram

Description automatically generated with medium confidence

Abbildung 8: Komponenten Diagramm

### Klassendiagramm

Folgendes Diagramm zeigt alle für die IPA relevanten Klassen auf. Die Klassen sind mit einer Farbe unterteilt, um so die Art zu identifizieren.

Graphical user interface, diagram, application

Description automatically generatedController | Service | Repository | Entity & DTO

Abbildung 9: Klassendiagramm

## Technische Spezifikationen

Im folgenden Abschnitt werden die technischen Spezifikationen beschrieben und wie das System den Datenfluss verarbeitet. Dabei werden folgende Themen dokumentiert:

* Datenfluss «*12.6.1 Datenfluss*»
* REST-Schnittstellen «*12.6.2 REST-Schnittstellen*»
* Datenbank-Schnittstellen «*12.6.3 Datenbank-Schnittstellen*»
* Stammdaten «*12.6.4 Stammdaten*»

### Datenfluss

Die Daten für das Frontend werden über die REST-Schnittstelle von dem Backend zurückgegeben. Das Backend verfügt über einen Controller, Service und ein Repository für die Anbindung der Datenbank. Die Request wird im Backend von dem Controller entgegengenommen. Der Controller sendet die Anfrage schliesslich an den Service weiter, wo diese verarbeitet wird. Bei den Abfragen werden «DTO» Modelle verwendet, welche auch durch den Service gemapped werden. Der Service greift mit Hilfe des Repositorys auf die Daten der Datenbank zu. Die zurückgekommenen Daten werden im Service je nach UseCase verarbeitet oder sortiert und anschliessend zurück an den Controller geschickt, der diese als Response an das Frontend schickt.

Diagram

Description automatically generated

Abbildung 10: Datenfluss

### REST-Schnittstellen

Im folgenden Abschnitt wird beschrieben, wie die REST-Schnittstellen bei einer Abfrage agieren und welche Antwort das Frontend bekommt.

|  |  |
| --- | --- |
| **getStatisticsOfAllStudies** | |
| Schnittstelle | /dashboard |
| Methode | GET |
| URL | **Local:** http://localhost:8080/dashboard  **DEV**: https://sbbgo-dev.app.ose.sbb-aws.net/dashboard  **PROD:** https://sbbgo-prod.app.ose.sbb-aws.net/ |
| Beschreibung | Über diese Schnittstelle werden alle Statistiken aller Studien für das Dashboard geliefert. Die Daten sind bereits in der Datenbank enthalten. |
| Parameter | - |
| Request-Body | - |
| Response | [  {  "studyId": 1,  "startDate": "2020-05-01",  "endDate": "2020-05-05",  "trainClass": 50.5,  "gender": {  "male": 12,  "female": 15  },  "journeyReason": "Freizeitreise",  "averageHappiness": 4.5,  "ageGroup": {  "ageGroup1": 13,  "ageGroup2": 5,  "ageGroup3": 5,  "ageGroup4": 10  },  "journeyReview": {  "journeyCount": 25,  "happinessFactor1": 2,  "happinessFactor2": 4,  "happinessFactor3": 6,  "happinessFactor4": 1  },  "aboType": {  "aboGa": 4,  "aboHalf": 6,  "aboRegional": 2,  "aboP2P": 4,  "aboOther": 3  },  "touchpointReview": [  {  "type": "Billettautomat",  "review": 5  }  ]  }  ] |
| Exceptions | * Datenbank nicht erreichbar * Datenbank enthält keine Daten * Zugriff auf Datenbank verweigert |

|  |  |
| --- | --- |
| **getStatisticOfStudy** | |
| Schnittstelle | /dashboard |
| Methode | POST |
| URL | **Local:** http://localhost:8080/dashboard  **DEV**: https://sbbgo-dev.app.ose.sbb-aws.net/dashboard  **PROD:** https://sbbgo-prod.app.ose.sbb-aws.net/ |
| Beschreibung | Diese Schnittstelle gibt Daten zu einer spezifischen Studie zurück. Der Benutzer kann eine Studie anhand ihres Namens im Dashboard auswählen und mit der gesendeten ID wird diese in der Datenbank geholt. |
| Parameter | - |
| Request-Body | {  "begin": "",  "end": "",  "isCoded": **false**,  "studyId": 1,  "journeyTitleKeyWords": [],  "aboTypes": [],  "journeyReasonIds": []  } |
| Response | {  "studyId": 1,  "startDate": "2020-05-01",  "endDate": "2020-05-05",  "trainClass": 50.5,  "gender": {  "male": 12,  "female": 15  },  "journeyReason": "Freizeitreise",  "averageHappiness": 4.5,  "ageGroup": {  "ageGroup1": 13,  "ageGroup2": 5,  "ageGroup3": 5,  "ageGroup4": 10  },  "journeyReview": {  "journeyCount": 25,  "happinessFactor1": 2,  "happinessFactor2": 4,  "happinessFactor3": 6,  "happinessFactor4": 1  },  "aboType": {  "aboGa": 4,  "aboHalf": 6,  "aboRegional": 2,  "aboP2P": 4,  "aboOther": 3  },  "touchpointReview": [  {  "type": "Billettautomat",  "review": 5  }  ]  } |
| Exceptions | * Studie mit StudyID nicht vorhanden * Datenbank nicht erreichbar * Zugriff auf Datenbank verweigert |

|  |  |
| --- | --- |
| **filterTouchpointReviews** | |
| Schnittstelle | /dashboard/reviews?name={name} |
| Methode | POST |
| URL | **Local:** http://localhost:8080/dashboard/reviews?name={name}  **DEV**: https://sbbgo-dev.app.ose.sbb-aws.net/dashboard/reviews?name={name}  **PROD:** https://sbbgo-prod.app.ose.sbb-aws.net//reviews?name={name} |
| Beschreibung | Mit dieser Schnittstelle kann die Tabelle der Berührungspunkte nach einem Namen gefiltert werden |
| Parameter | Name eines Berührungspunktes (Wird beim Eintippen kontinuierlich ergänz – Beispiel «Billettautomat») |
| Request-Body | - |
| Response | [  {  "type": "Billettautomat",  "rating": 5  },  {  "type": "Auskunftsschalter",  "rating": 5  }  ] |
| Exceptions | * Berührungspunkt mit {name} nicht vorhanden * Datenbank nicht erreichbar * Zugriff auf Datenbank verweigert |

### Datenbank-Schnittstellen

In diesem Abschnitt werden alle Schnittstellen zu der Datenbank beschrieben. Die verwendete Technologie hierbei ist Spring Data JPA.

|  |  |
| --- | --- |
| **getAllStudies** | |
| Methode | GET |
| Beschreibung | Mit dieser Schnittstelle werden alle benötigten Daten für das Dashboard zurückgegeben. Dabei werden alle Studien zurückgegeben. |
| Request | getAllStudies() |
| Response | Objekt: List<DashboardDTO>() |
| Repository | TouchpointRepository |

|  |  |
| --- | --- |
| **findAllByStudyId** | |
| Methode | GET |
| Beschreibung | Mit dieser Schnittstelle werden alle benötigten Daten einer einzelnen Studie zurückgegeben. Dies wird mit dem Attribut «*StudyID*» in der Datenbank gefiltert. |
| Request | findAllByStudyId() |
| Response | Objekt: DashboardDTO() |
| Repository | TouchpointRepository |

### Stammdaten

Für meine Neuerung während der IPA Probezeit werden die bereits vorhandenen Daten in der Datenbank benötigt. Somit werden keine neuen Datensätze und Tabellen in der Datenbank benötigt. Eine detaillierte Auflistung kann unter «*12.5.2 ERD Diagramm*» eingesehen werden.

## Fachliche Spezifikationen

Innerhalb dieses Kapitels wird beschrieben, wie die Web Applikation mit Hilfe von Rollen geschützt wird und so nur ausgewählte Personen Zugriff haben.

### Rollen & Berechtigungen

Um die Web Applikation von SBB go zu verwendet, muss man als Administrator im Azure Active Directory hinzugeführt werden.

Graphical user interface, application, table

Description automatically generated

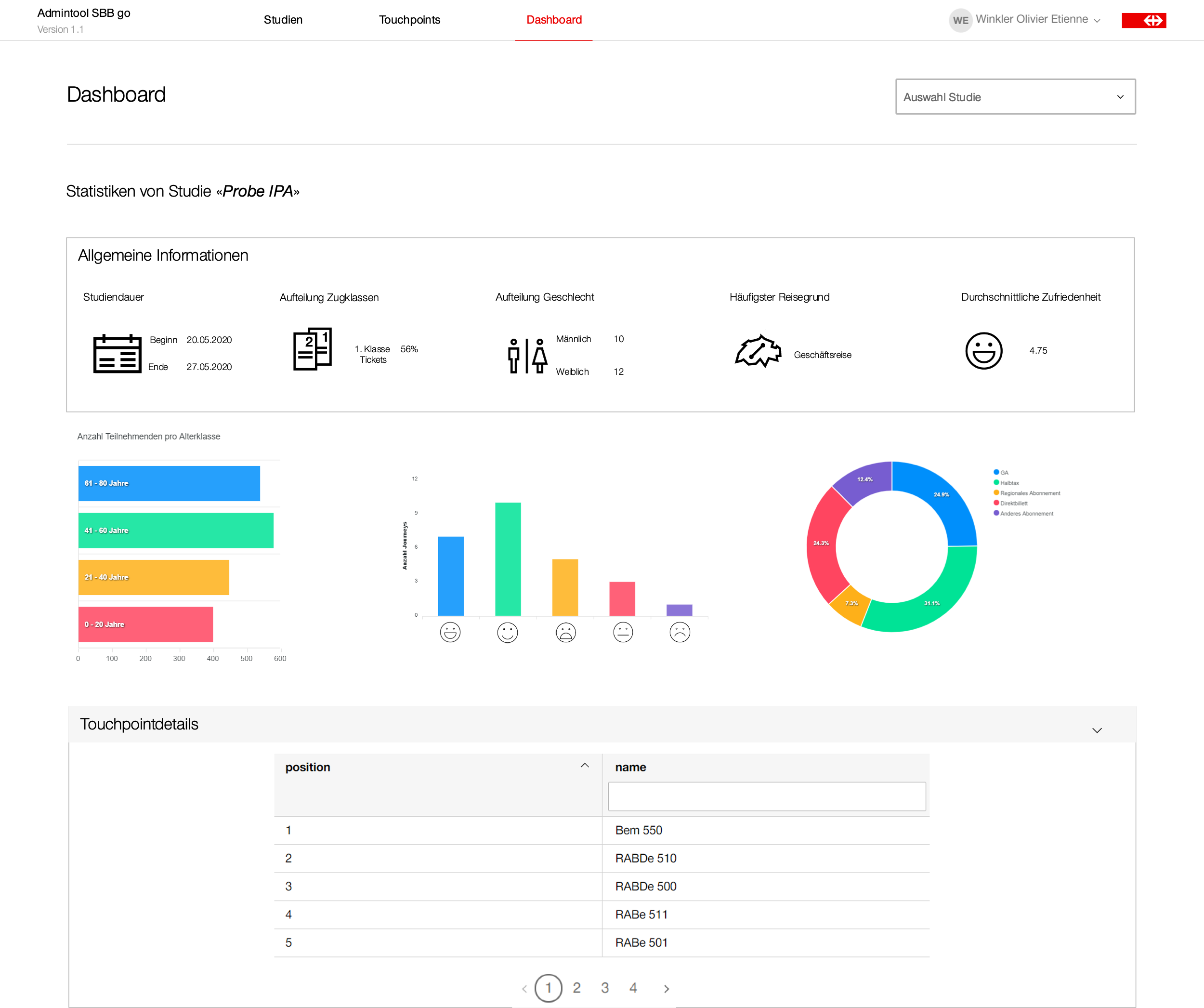
Graphical user interface, text, application, email

Description automatically generated// TODO Beschreibung und Bild

## Mockups

Die untenstehende Abbildung zeigt das Dashboard in seiner vollen Pracht. Das Mockup wurde mit der Software «*Adobe XD*» erstellt. Als Basis von diesem Mockup wurde die bereits auf «*Sketch*» hochgeladene erste Version genommen. Die Icons wurden direkt aus der eigenen Iconlibrary entnommen. Die Charts wurden mit Hilfe der Library «*ApexCharts*» erstellt.

Das Dashboard ist mit Hilfe eines Reiters im Header erreichbar. Es kann auch direkt über die URL «*/dashboard*» aufgerufen werden.



Mithilfe eines «*Dropdown*»-Menüs kann der Benutzer die Statistiken zu einer spezifischen Studie anzeigen lassen. Als Auswahlmöglichkeit dienen alle vorhandenen Studiennamen.

Im Bereich der Allgemeinen Informationen sind folgende Statistiken enthalten:

* Studiendauer

Zeigt das Start- sowie das Enddatum der Studie

* Aufteilung Zugklassen

Zeigt die Aufteilung der Zugklassen, in der die Studienteilnehmenden unterwegs waren

* Aufteilung Geschlecht

Zeigt die Aufteilung der Geschlechter

* Häufigster Reisegrund

Zeigt den häufigste angegebenen Reisegrund der Teilnehmenden

* Durchschnittliche Zufriedenheit

Zeigt die durchschnittliche Zufriedenheit während der Studie

Insgesamt verfügt das Dashboard über drei interaktive Diagramme.

* Altersklassen

Zeigt die Verteilung der Altersklassen in der Studie

* Zufriedenheit pro Reise

Zeigt alle Bewertungen pro Reise in der Studie an

* Aufteilung Abonnemente

Zeigt die Aufteilung der Abonnemente der Studienteilnehmer auf

Unter den Touchpointdetails wird eine Tabelle mit allen Berührungspunkten der Studie angezeigt. Die Tabelle wird standardmässig mit den schlechtesten Bewertungen angezeigt, kann jedoch nach bester / schlechtester Bewertung sortiert werden. Zudem verfügt die Tabelle über ein Textfeld, welches der Benutzer als Suchfunktion benötigen kann, um einen Berührungspunkt nach Namen zu filtern. Mithilfe des «*Paginators*» kann die Seite der Tabelle gewechselt werden.

## Build und Deployment

Der Build und Deployment Prozess ist bereits in Betrieb mit der aktuellen Version der Applikation. Die Änderungen, die während dieser IPA Probezeit vorgenommen werden, werden nach Abschluss der korrekten Phasen auf das bestehende Openshift-Projekt hochgeladen und ausgerollt. Die Applikation wird auf sogenannten zwei «Stages» laufen. In unserem Fall verwenden wir «DEV» und «PROD», was etwa so viel bedeutet wie Entwicklung und Produktion, also die erste Plattform ist für die Entwicklung zu gebrauchen, während die zweite nur für eine neue Version gebraucht wird. Durch diese Differenzierung können Änderungen bereits auf einer produktiven Umgebung getestet werden, bevor diese auf die Umgebung der Kunden hochgeladen wird.

Folgende Tools werden für den ganzen Prozess benötigt:

* **GIT / Bitbucket**

Durch das Verwaltungstool GIT kann ich meinen geschriebenen Code verwalten. Dank diesem Tool kann ich den Code immer in verschiedenen Versionen abrufen. Bitbucket ist unsere Versionierungsplattform, auf der der Code verwaltet wird. Für das Projekt und die Applikation ist ein eigenes Repository erstellt worden.

* **Jenkins**

Jenkins ist eine Plattform, die zur kontinuierlichen Integration von Software dient. Dank Jenkins wird mein Code überprüft und für die weiteren Schritte aufbereitet.

* **Artifactory**

Artifactory speichert alle Packages und Images von Maven und Docker. Wenn Jenkins das Projekt baut und untersucht wird auch eine Abbildung der Applikation mit Docker oder Maven in Artifactory gespeichert. Mit diesen kleinen Paketen ist es möglich die Applikation skalierbar und für die Virtualisierung bereit zu halten.

* **Openshift**

Openshfit ist für die Verwaltung von einzelnen Containern verantwortlich. In einem solchen Container kann dann die Applikation mit Hilfe des abgelegten Pakets von Artifactory operieren.

Graphical user interface, application

Description automatically generated

## Testkonzept

In diesem Abschnitt wird das Testkonzept erarbeitet. Dieses Konzept ist für die Realisierungsphase ausschlaggebend und sichert so die Qualität des Produktes. In dem Konzept wird genau definiert, was in der Realisierung getestet wird und wie dies von statten kommen soll. Bei den erarbeitenden Testfällen muss sichergestellt werden, dass diese die vorgegebenen UseCases abdecken. Nur so ist die Qualitätssicherung gewährleistet.

Bei einigen Projekten sind extra Projektmitarbeiter für das Testing vorgesehen. In meiner IPA bin ich selbst dafür vorgesehen, wie unter «*6.1 Projektaufbauorganisation*» beschrieben.

### Testumgebung

|  |  |
| --- | --- |
| **Tool / Software / Hardware** | **Beschreibung** |
| Logitech Maus | Persönliche Wireless Maus |
| MacBook Pro 2013 (15’) | Arbeitslaptop mit macOS Big Sur Version 11.1 |
| Dell U34W19 (34) | Persönlicher Ultrawide Monitor |
| Postman (Version 7.36.1) | Tool, um Schnittstellen zu testen. |
| IntellijIDEA (2020.03) | Entwicklungsprogramm mitsamt Applikation lokal |
| H2 Console | Webinterface für H2 Datenbank von Spring Boot |
| Brave Browser (Version 1.19.86) | Webbrowser, indem alles getestet wird |
| Testkonzept | Eigen erstelltes Testkonzept inklusive aller Testfälle |

### Testmethode

**Unittest**:

Mit Unittests kann die Qualität von Code sichergestellt werden. Mit einem solchen Test können einzelne Methoden getestet werden und komplexere Methoden mit Mockdaten versehen.

**Integrationstest**:

Mit Integrationstest kann die Qualität der ganzen Applikation sichergestellt werden. Durch den komplexeren Aufbau eines Integrationstest ist es möglich, viele UseCases abzudecken.

**UI-Test / Frontend-Test:**

Im Frontend kann mit einem eigenen Testframework getestet werden. So können Werte der einzelnen Felder simuliert werden und das Verhalten überprüft werden.

### Testziele

Folgende Ziele sollen durch die Tests erfolgreich erreicht werden:

* Geschriebener Code hat ein hohes Niveau an Qualität
* Schwerwiegende Fehler können durch Tests erkannt werden
* Alle Anwendungsfälle wurden implementiert und getestet

Das Testverfahren wird während der Realisierungsphase durchgeführt. Sollte im Fall der Fälle ein unerwarteter Fehler auftreten, so wird dieser sorgfältig dokumentiert und wenn möglich gelöst. Um gegen diesen Fall vorbereitet zu sein, wurde im Zeitplan eine Reservezeit von sechs Arbeitsstunden eingeplant.

### Testfälle

**Testfälle zu UseCase 1: Daten aller Studien anzeigen**

|  |  |
| --- | --- |
| ID | Testfall 01 |
| Beschreibung | Es wird getestet, ob der User die Statistiken aller Studien sieht, wenn er das Dashboard besucht (Über Header als auch mit URL). So sollten alle Illustrationen und Diagramme die richtigen Daten darstellen. |
| Vorbedingungen | * Testperson hat Zugriff auf Applikation * Die Applikation wurde lokal gestartet * Mockdaten sind geladen * <http://localhost:4200> wurde im Browser geöffnet |
| Testschritte | * Testperson klickt auf den Reiter «Dashboard» im Header * Die Testperson gelangt auf die Dashboardseite |
| Erwartetes Resultat | * Das Backend liefert die Datensätze aller Statistiken der Studien * Das Frontend zeigt die Daten korrekt an (Diagramme und Illustrationen verfügen über korrekte Daten) |

**Testfälle zu UseCase 2: Daten einer einzelnen Studie anzeigen**

|  |  |
| --- | --- |
| ID | Testfall 02 |
| Beschreibung | Es wird getestet, ob die Testperson alle Daten einer einzelnen Studie angezeigt bekommen kommt und diese wieder korrekt dargestellt werden |
| Vorbedingungen | * Testperson hat Zugriff auf Applikation * Die Applikation wurde lokal gestartet * Mockdaten sind geladen * <http://localhost:4200> wurde im Browser geöffnet |
| Testschritte | * Testperson klickt auf den Reiter «Dashboard» im Header * Auf der Dashboardseite wählt der Tester im Dropdownmenü eine Studie aus. |
| Erwartetes Resultat | * Das Backend liefert die Datensätze aller Statistiken der ausgewählten Studie * Das Frontend zeigt alle Daten der einzelnen Studie korrekt an. |

**Testfälle zu UseCase 3: Berührungspunkte nach Kriterien filtern**

|  |  |
| --- | --- |
| ID | Testfall 03 |
| Beschreibung | Es wird getestet ob sich die Tabelle der Touchpointdetails nach ihrer Bewertung filtern lässt |
| Vorbedingungen | * Testperson hat Zugriff auf Applikation * Die Applikation wurde lokal gestartet * Mockdaten sind geladen * <http://localhost:4200> wurde im Browser geöffnet |
| Testschritte | * Testperson klickt auf den Reiter «Dashboard» im Header * Der Tester sortiert mit Hilfe des Pfeilicons in dem Tabellenheader die Touchpoints nach ihrer Bewertung, zuerst nach den schlechtesten und anschliessend nach den besten |
| Erwartetes Resultat | * Das Backend liefert die Datensätze zu allen Studien / zu einer Studie * Das Frontend zeigt alle Daten korrekt an, bei Klick auf Pfeilicon der Tabelle werden die Touchpoints sortiert. |

|  |  |
| --- | --- |
| ID | Testfall 04 |
| Beschreibung | Es wird getestet ob sich die Tabelle der Touchpointbewertung nach einem Namen filtern lässt |
| Vorbedingungen | * Testperson hat Zugriff auf Applikation * Die Applikation wurde lokal gestartet * Mockdaten sind geladen * <http://localhost:4200> wurde im Browser geöffnet |
| Testschritte | * Testperson klickt auf den Reiter «Dashboard» im Header * Der Tester tippt nach und nach Buchstaben in das Textfeld hinein. |
| Erwartetes Resultat | * Das Backend sortiert die Touchpoints anhand der Eingabe des Users und gibt diese zurück. * Das Frontend zeigt die Touchpoints gefiltert an |

# Realisierung

Die Realisierungsphase dient dazu, die in der Konzeptphase erstellten konzeptionellen Anwendungsfälle in die Realität umzusetzen. Zum Schluss dieser Phase wird sichergestellt, dass alle Anwendungsfälle umgesetzt und abgedeckt sind und diese mithilfe von Tests überprüft wurden.

## Systemdarstellung

In diesem Kapitel werden die vollzogenen Änderungen dokumentiert. Aufgezeigt werden diverse Übersichten und die Implementation. Der konzeptionelle Ansatz kann unter *«12.3 Systemmodellierung*» entnommen werden.

### Implementation der UseCases

Folgend sind die Implementationen der UseCases in Form von Sequenzdiagrammen zu sehen. Diese wurden nach den Planungen aus der Konzeptphase umgesetzt.

**Backend: Statistiken aller Studien**

Für den ganzen Ablauf wurde ein Controller, ein Service mitsamt Logik, ein Service für den Erhalt der Daten und das Repository als Schnittstelle zur Datenbank verwendet. Dem Benutzer werden die Daten mithilfe des Frontend in der Dashboardseite angezeigt und er kann so eine Übersicht erhalten über alle Studiendaten.

Diagram

Description automatically generated with medium confidence

**Backend: Statistiken einer Studie:**

Der ganze Ablauf ist grundlegen identisch wie der Erste, jedoch werden hier die Daten nur von einer Studie verarbeitet. Somit wurde ein Controller, ein Service mitsamt Logik, ein Service für den Erhalt der Daten und das Repository als Schnittstelle zur Datenbank verwendet. Dem Benutzer werden die Daten mithilfe des Frontend in der Dashboardseite angezeigt. Der Benutzer kann mit diesem Ablauf einen Überblick auf eine spezifische Studie erhalten.

Diagram

Description automatically generated with medium confidence

**Backend: Daten für Auswahl von Studien**

Der letzte Ablauf dient zur ständigen Auswahl von Studien für ein Dropdown im Frontend. So kann der Benutzer in jeder Situation auf dem Dashboard eine einzelne Studie auswählen. Hier wurde ein Controller, ein Service mitsamt Logik, ein weiterer Service für den Erhalt der Daten und das dazugehörende Repository verwendet.

Diagram

Description automatically generated with medium confidence

### Stammdaten

Für die entwickelten Anwendungsfällen waren keine zusätzlichen Stammdaten nötig und somit ist dieses Kapitel hiermit leer.

### Implementation REST Schnittstellen

Alle erfassten REST Schnittstellen aus der Konzeptphase wurden planmässig erfasst. Somit gibt es keine Abweichungen zwischen Konzept und Realisierung. Die REST Schnittstellen wurden mithilfe von Tests getestet und abgedeckt.

### Implementation Frontend

Das Frontend wurde gemäss dem Mockup umgesetzt und wird im Kapitel «*13.3 Vergleich Vorher / Nachher*» weiter erklärt.

Graphical user interface, website

Description automatically generated

### Implementation Testing

Für die garantierte Sicherstellung der Datenverarbeitung wurden diverse Unittests geschrieben, um dies zu überprüfen. Diese Unittests sind über alle Anwendungsfällen verstreut und decken so den ganzen relevanten Code ab. Somit ist die Codequalität gesichert.

**Unittests für Dashboardcontroller**

**getStatisticsOfAllStudies**: Mit diesem Test wird getestet, ob die Statistiken von allen Studien zurückgegeben werden. Für die einzelne Studie wird das gleiche Verfahren vollzogen, somit ist für diesen Ablauf kein weiterer Test nötigt, da dies eine redundanter Testmethode wäre.

@Test  
void getStatisticsOfAllStudies\_returns200() throws Exception {  
 *doReturn*(DashboardDto.*from*(  
 Set.*of*(StudyDetailsDto.*from*(*mock*(TouchpointDto.class))),  
 Gender.*from*(10, 10),  
 AgeGroup.*from*(5, 3, 5, 1),  
 AboType.*from*(10, 5, 1, 6, 0),  
 50.0,  
 JourneyRating.*from*("Freizeit", 10, 4, 4, 3, 2, 5, 1),  
 List.*of*(new TouchpointRating())  
 )).when(dashboardServiceMock).getStatisticsOfStudies();  
  
 mvc.perform(*get*("/api/v1/dashboard")).andExpect(*status*().isOk())  
 .andExpect(*content*().string(*containsString*("Freizeit")));  
  
}

**getAllStudies:** Dieser Test deckt das Dropdown ab. Das Dropdown muss immer über alle Studien verfügen und deshalb gibt es diese Request. Im Test selbst wird getestet, ob die benötigten Daten vorhanden sind und zurückgeschickt.

@Test  
void getAllStudies\_returns200() throws Exception {  
 *doReturn*(Set.*of*(StudyDetailsDto.*from*(*mock*(TouchpointDto.class)))).when(dashboardServiceMock).getAllStudies();  
  
 String expectedResult = objectMapper.writeValueAsString(Set.*of*(StudyDetailsDto.*from*(*mock*(TouchpointDto.class))));  
  
 mvc.perform(*get*("/api/v1/dashboard/studies")).andExpect(*status*().isOk()).andExpect(*content*().string(expectedResult));  
}

**Unittests für DashboardService**

**getTouchpointRating**: Dieser Test überprüft, ob das benötigte Model korrekt verarbeitet wird und gibt dies entsprechend zurück.

@Test  
void getTouchpointRating\_returnsTouchpointRating() {  
 TouchpointRating touchpointRating = new TouchpointRating();  
 touchpointRating.setRating(5);  
 touchpointRating.setType("Perron");  
  
 *assertEquals*("Perron", touchpointRating.getType());  
}

**getJourney**: Dieser Test überprüft, ob das benötigte Model korrekt verarbeitet wird und gibt dies entsprechend zurück.

@Test  
void getJourney\_returnsJourney() {  
 Journey journey = new Journey();  
 journey.setJourneyId(1L);  
 journey.setJourneyReasons(List.*of*("Freizeitreise", "Geschäftsreise"));  
 journey.setOverallHappinessFactor(4);  
  
 *assertEquals*(List.*of*("Freizeitreise", "Geschäftsreise"), journey.getJourneyReasons());  
}

**getStudyDetails:** Dieser Test deckt die Studiendetails ab. Für jede Studie wird ein solches Model generiert. Im Testfall wird zehnmal ein solches Model in die Liste gespeichert. Da es sich aber um ein Set<> handelt und somit nur einzigartige Werte annimmt, wird nur ein Model schlussendlich zurückgegeben.

@Test  
void getStudyDetails\_returnsStudyDetailsDto() {  
 Set<StudyDetailsDto> studyDetailsDtos = new HashSet<>();  
  
 for (int i = 0; i < 10; i++) {  
 studyDetailsDtos.add(StudyDetailsDto.*from*(*mock*(TouchpointDto.class)));  
 }  
  
 *assertEquals*(1, studyDetailsDtos.size());  
}

**getGenders:** Dieser Test überprüft den Ablauf zur Zählung der einzelnen Geschlechter. Diese werden dann mit dem Model zurückgesendet.

@Test  
void getGenders\_returnsGender() {  
 List<String> genders = List.*of*("male", "male", "male", "male", "female", "female", "female", "female");  
  
 int male = (int) genders.stream().filter(gender -> gender.equals("male")).count();  
 int female = genders.size() - male;  
  
 Gender gender = Gender.*from*(male, female);  
  
 *assertEquals*(4, gender.getMale());  
 *assertEquals*(4, gender.getFemale());  
}

**getAgeGroups:** Dieser Test überprüft die Berechnung der einzelnen Altersgruppen. Für jede vorgegeben Altersklasse wird eine Zahl gezählt, die schliesslich mithilfe des Models verwendet werden kann.

@Test  
void getAgeGroups\_returnsAgeGroup() {  
 List<Integer> ageGroups = List.*of*(1920, 1930, 1940, 1950, 1960, 1970, 1980, 1990, 2000, 2010);  
  
 int boomer = 0, genX = 0, millenial = 0, genZ = 0;  
  
 for (Integer age : ageGroups) {  
 if (age <= 1946) boomer++;  
 else if (age <= 1976) genX++;  
 else if (age <= 1995) millenial++;  
 else if (age <= 2010) genZ++;  
 }  
  
 AgeGroup ageGroup = AgeGroup.*from*(boomer, genX, millenial, genZ);  
  
 *assertEquals*(3, ageGroup.getBabyBoomer());  
 *assertEquals*(3, ageGroup.getGenX());  
 *assertEquals*(2, ageGroup.getMillennial());  
 *assertEquals*(2, ageGroup.getGenZ());  
}

**getAboTypes:** Dieser Test dient der Überprüfung der Berechnung der diversen Abonnemente. Er ist ähnlich aufgebaut wie der vorherige Test. Auch hier werden die verschiedenen Typen gezählt und schliesslich für die Verwendung an ein Model übergeben.

@Test  
void getAboTypes\_returnsAboType() {  
 List<String> aboTypes = List.*of*("aboGa", "aboGa", "aboGa", "aboGa", "aboHalf", "aboHalf", "aboHalf");  
  
 int aboGa = 0, aboHalf = 0, aboRegional = 0, aboP2p = 0, aboOther = 0;  
  
 for (String s : aboTypes) {  
 switch (s) {  
 case "aboGa":  
 aboGa++;  
 break;  
 case "aboHalf":  
 aboHalf++;  
 break;  
 case "aboRegional":  
 aboRegional++;  
 break;  
 case "aboP2p":  
 aboP2p++;  
 break;  
 case "aboOther":  
 aboOther++;  
 break;  
 default:  
 break;  
 }  
 }  
  
  
 AboType aboType = AboType.*from*(aboGa, aboHalf, aboRegional, aboP2p, aboOther);  
  
 *assertEquals*(4, aboType.getAboGa());  
 *assertEquals*(3, aboType.getAboHalf());  
 *assertEquals*(0, aboType.getAboRegional());  
 *assertEquals*(0, aboType.getAboP2p());  
 *assertEquals*(0, aboType.getAboOther());  
}

**getTrainClass:** Dieser Test berechnet den Prozentanteil von Tickets in der ersten Klasse. Schlussendlich wird direkt der Prozentsatz zurückgegeben.

@Test  
void getTrainClasses\_returnsTrainClass() {  
 List<String> trainClasses = List.*of*("first", "first", "second", "second", "second", "second", "second", "second", "second", "second");  
  
 double total = trainClasses.size();  
 double firstClass = (double) trainClasses.stream().filter(trainClass -> trainClass.equals("first")).count();  
  
 *assertEquals*(20, (firstClass / total) \* 100);  
}

**getJourneys:** Dieser Test ist für die Berechnung der Journeys zuständig. Innerhalb werden gleich mehrere Kennzahlen berechnet. Zuerst werdne die benötigten Testdaten generiert. Danach findet die Berechnung statt und schlussendlich wieder die Zuweisung an das Model.

@Test  
void getJourneys\_returnsJourney() {  
 List<Journey> journeys = new ArrayList<>();  
  
 for (int i = 0; i < 5; i++) {  
 Journey journey = new Journey();  
 journey.setJourneyId((long) i);  
 journey.setJourneyReasons(List.*of*("Freizeit", "Freizeit", "Geschäftsreise"));  
 journey.setOverallHappinessFactor(i);  
 journeys.add(journey);  
 }  
  
 Set<Long> totalJourney = new HashSet<>();  
 double overallHappinessFactor;  
 int overallHappinessFactorJourney = 0, awesome = 0, good = 0, ok = 0, bad = 0, worst = 0;  
 List<String> journeyReasons = new ArrayList<>();  
  
 for (Journey journey : journeys) {  
 totalJourney.add(journey.getJourneyId());  
 overallHappinessFactorJourney += journey.getOverallHappinessFactor();  
 journeyReasons.addAll(journey.getJourneyReasons());  
  
 switch (journey.getOverallHappinessFactor()) {  
 case 1:  
 worst++;  
 break;  
 case 2:  
 bad++;  
 break;  
 case 3:  
 ok++;  
 break;  
 case 4:  
 good++;  
 break;  
 case 5:  
 awesome++;  
 break;  
 default:  
 break;  
 }  
 }  
  
 Map<String, Long> reasonsWithNumberOfJourneys = journeyReasons.stream().collect(Collectors.*groupingBy*(reason -> reason, Collectors.*counting*()));  
  
 Optional<Map.Entry<String, Long>> mostFrequentJourneyReason = reasonsWithNumberOfJourneys.entrySet()  
 .stream()  
 .max(Map.Entry.*comparingByValue*());  
  
 overallHappinessFactor = overallHappinessFactorJourney / totalJourney.size();  
  
 JourneyRating journeyRating = JourneyRating.*from*(mostFrequentJourneyReason.get().getKey(), totalJourney.size(), overallHappinessFactor, awesome, good, ok, bad, worst);  
  
 *assertEquals*(0, journeyRating.getTotalHappinessFactorAwesome());  
 *assertEquals*(1, journeyRating.getTotalHappinessFactorGood());  
 *assertEquals*(1, journeyRating.getTotalHappinessFactorOk());  
 *assertEquals*(1, journeyRating.getTotalHappinessFactorBad());  
 *assertEquals*(1, journeyRating.getTotalHappinessFactorWorst());  
 *assertEquals*(5, journeyRating.getTotalJourneys());  
 *assertEquals*(2.0, journeyRating.getOverallHappinessFactor());  
 *assertEquals*("Freizeit", journeyRating.getMostFrequentJourneyReason());  
}

## Testing

Im folgenden Kapitel wird das zuvor angelegte Testkonzept an den neu dazugekommenen Anwendungsfällen getestet.

### Testfallübersicht

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | UseCase | Beschreibung | Bemerkung | Ergebnis |
| Testfall 01 | 1 | Es wird getestet, ob der User die Statistiken aller Studien sieht, wenn er das Dashboard besucht (Über Header als auch mit URL). So sollten alle Illustrationen und Diagramme die richtigen Daten darstellen. | - | OK |
| Testfall 02 | 2 | Es wird getestet, ob die Testperson alle Daten einer einzelnen Studie angezeigt bekommen kommt und diese wieder korrekt dargestellt werden | - | OK |
| Testfall 03 | 3 | Es wird getestet ob sich die Tabelle der Touchpointdetails nach ihrer Bewertung filtern lässt | - | OK |
| Testfall 04 | 3 | Es wird getestet ob sich die Tabelle der Touchpointbewertung nach einem Namen filtern lässt | - | OK |

### Testobjekte

**Testfall 01**

Im ersten Testfall werden die Statistiken für das Dashboard von allen Studien geholt. Dies soll geschehen, wenn der Benutzer die Dashboardseite («/dashboard») besucht. Die dabei entstehende Abfrage gibt ein Model mit allen benötigten Datensätzen zurück, welche aus der Datenbank gelesen werden. Resultat von Backend:

{

"studyDetails": [

{

"studyId": 2,

"name": "name\_de",

"startDate": "2021-01-20",

"endDate": "2021-01-18"

},

{

"studyId": 1,

"name": "name\_de",

"startDate": "2020-05-17",

"endDate": "2020-05-18"

}

],

"gender": {

"male": 5,

"female": 3

},

"ageGroup": {

"babyBoomer": 0,

"genX": 0,

"millennial": 2,

"genZ": 6

},

"aboType": {

"aboGa": 3,

"aboHalf": 4,

"aboRegional": 1,

"aboP2p": 0,

"aboOther": 0

},

"trainClass": 37.5,

"journeyRating": {

"mostFrequentJourneyReason": "Geschäftsreise",

"totalJourneys": 7,

"overallHappinessFactor": 3.0,

"totalHappinessFactorAwesome": 1,

"totalHappinessFactorGood": 1,

"totalHappinessFactorOk": 4,

"totalHappinessFactorBad": 1,

"totalHappinessFactorWorst": 1

},

"touchpointRatings": [

{

"type": "Abonnemente und Billette",

"rating": 1.0

},

{

"type": "Abonnemente und Billette",

"rating": 5.0

},

{

"type": "Abonnemente und Billette",

"rating": 4.0

},

{

"type": "Anzeigetafel",

"rating": 3.0

},

{

"type": "Perron",

"rating": 2.0

},

{

"type": "Perron",

"rating": 1.0

}

]

}

**Testfall 02**

Im zweiten Testfall werden die Statistiken für das Dashboard nur von einer einzigen Studie angezeigt. Der Benutzer kann dafür die Studie aus dem dafür vorgesehenen Dropdown auswählen. Die Abfrage auf das Backend gibt das gleiche Model zurück, jedoch mit den Daten einer Studie. Resultat von Backend:

{

"studyDetails": [

{

"studyId": 1,

"name": "name\_de",

"startDate": "2020-05-17",

"endDate": "2020-05-18"

}

],

"gender": {

"male": 1,

"female": 2

},

"ageGroup": {

"babyBoomer": 0,

"genX": 0,

"millennial": 0,

"genZ": 3

},

"aboType": {

"aboGa": 1,

"aboHalf": 2,

"aboRegional": 0,

"aboP2p": 0,

"aboOther": 0

},

"trainClass": 0.0,

"journeyRating": {

"mostFrequentJourneyReason": "Geschäftsreise",

"totalJourneys": 2,

"overallHappinessFactor": 4.0,

"totalHappinessFactorAwesome": 0,

"totalHappinessFactorGood": 0,

"totalHappinessFactorOk": 3,

"totalHappinessFactorBad": 0,

"totalHappinessFactorWorst": 0

},

"touchpointRatings": [

{

"type": "Abonnemente und Billette",

"rating": 1.0

}

]

}

**Testfall 03**

Im dritten Testfall wird getestet, ob sich die Tabelle der Touchpointbewertungen nach der jeweiligen Bewertung eines Touchpoints filtern lässt. Dies kann mit einem kleinen Pfeilicon im Tabellenkopf passieren. Resultat in Frontend:

Table

Description automatically generated

**Testfall 04**

Im vierten und letzten Testfall wird getestet, ob sich die Tabelle der Touchpointbewertungen nach dem Namen von Touchpoints filtern lässt. Der Benutzer kann eine beliebige Buchstabenkombination eingeben und ihm werden die Resultate angezeigt. Resultat in Frontend:

**Table

Description automatically generated**

### Testdurchführung / Testergebnisse

|  |  |  |
| --- | --- | --- |
| ID / Bezeichnung | Testfall 01 | Statistiken aller Studien |
| Datum & Tester | 25. Januar 2021, Winkler Olivier | |
| Ergebnis | Statistiken von allen Studien werden korrekt und ohne Fehler angezeigt. | |
| Massnahme | - | |

|  |  |  |
| --- | --- | --- |
| ID / Bezeichnung | Testfall 02 | Statistiken einer Studie |
| Datum & Tester | 25. Januar 2021, Winkler Olivier | |
| Ergebnis | Die Auswahl der Studien funktioniert, die Statistiken werden nur von der einzelnen Studie angezeigt. | |
| Massnahme | - | |

|  |  |  |
| --- | --- | --- |
| ID / Bezeichnung | Testfall 03 | Touchpointbewertungen nach Bewertung filtern |
| Datum & Tester | 25. Januar 2021, Winkler Olivier | |
| Ergebnis | Zuerst werden alle Touchpoints mit der dazugehörigen Bewertung angezeigt. Bei Klick auf das Pfeilicon werden die Touchpoint nach der Bewertung sprich von der Grössten zur Kleinstens und umgekehrt sortiert. | |
| Massnahme | - | |

|  |  |  |
| --- | --- | --- |
| ID / Bezeichnung | Testfall 04 | Touchpointbewertungen nach Namen filtern |
| Datum & Tester | 25. Januar 2021, Winkler Olivier | |
| Ergebnis | Zuerst werden alle Touchpoints mit der dazugehörigen Bewertung angezeigt. Bei der kontinuierlichen Eingabe eines Buchstaben in das Inputfeld, werden die Touchpoints nach diesen gefiltert und nur die passenden angezeigt. | |
| Massnahme | - | |

### Unittests

Für alle funktionsrelevanten Methoden wurden Unittests geschrieben. So konnte die Testabdeckung nach SBB Standards von mindestens 80% eingehalten werden.

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text

Description automatically generated with medium confidence

### Testfazit

Alle durchgeführten Usertests im Front- als auch Backend sind erfolgreich durchgeführt worden und haben das erwartete Resultat zurückgegeben. Im Backend sind zudem die automatisierten Tests erfolgreich. Somit sind dank diesem Zustand keine weiteren Massnahmen mehr nötig.

## Vergleich Vorher / Nachher

In diesem Abschnitt wird die Webapplikation von «SBB go» zu der vorherigen Version ohne meine Änderungen verglichen.

Vor meiner IPA war das Dashboard eine reine Idee und nicht in einer Weise umgesetzt. Somit ist die vorherige Version ohne jegliche Art von Dashboard im Einsatz vorhanden. Das Dashboard kann über den Header oder direkt über die URL aufgerufen werden. Standardmässig werden die Statistiken aller Studien angezeigt. Auf der obersten Zeile kann das Dropdown erkannt werden. Der Benutzer kann dort eine Studie von den vorhandenen auswählen. Wenn eine Studie ausgewählt wird, wird die Seite automatisch neu geladen und mit den Daten dieser Studie befüllt. In der nächsten Zeile befinden sich allgemeine Informationen über die Studien / Studie. Es wird die Studiendauer, Aufteilung der Zugklassen, Aufteilung der Geschlechter, der häufigste Reisegrund und die durchschnittliche Zufriedenheit angezeigt. Wenn alle Studien angezeigt werden, bleibt die Studiendauer leer. Die Aufteilung der Zugklassen wird in Prozent angegeben und referenziert auf die Erstklassentickets der gesamten Reisen. Die Aufteilung der Geschlechter wird zwischen Mann und Frau getrennt. Der häufigste Reisegrund ist der meiste, welcher von den Studienteilnehmer ausgewählt wurde. Die durchschnittliche Zufriedenheit wird auf der Zufriedenheit der Reise berechnet.

Die Diagramme zeigen die Altersklassen, die Bewertungen der Reise und die Verteilung der Abonnemente auf. Alle Diagramme wurden mithilfe von einer NPM-Library erstellt. Sie sind interaktiv bedienbar und einfach zu erweitern. Zuunterst befindet sich eine Tabelle mit den Touchpointbewertungen. Diese ist standardmässig dank eines Accordion der SBB Angular Library versteckt. Die Tabelle kann nach der Bewertung als auch nach dem Namen des Touchpoints gefiltert werden.

Im direkten Vergleich zu dem Mockup sieht alles sehr ähnlich aus, dem ist auch so. Jedoch gibt es kleine Unterschiede, die sich bei der Entwicklung als «unnötig» herausgestellt haben. Als erstes wurden noch die Farben der SBB verwendet. Diese sind unter «» aufgelistet. Zudem sind beim zweiten Diagramm keine Smileys vorhanden, sondern nur Nummern. Das Problem bei der Verwendung von Smileys wäre die korrekte Position gewesen und das Fehlen eines fünften Bildes. Da sich die Icons der SBB manchmal in der Grösse unterscheiden können, konnte ich somit nicht die Konsistenz der Grösse sicherstellen und habe dies dem Zeitplan zuliebe auf Nummern beschränkt. Zuletzt habe ich noch eine unnötige Überschrift entfernt und die Icons bei den allgemeinen Icons auf die gleiche Höhe wie der Titel gebracht. Ansonsten ist das UI nach Mockup umgesetzt.

### Ansicht Dashboard – Alle Studien

Graphical user interface, website

Description automatically generatedUntenstehend ein Überblick auf die Applikation mitsamt dem Dashboard

### Ansicht Dashboard – Eine Studie

Folgden die Übersicht der Statistiken einer einzelnen Studie. Nun ist zu sehen, wie die Studiendauer nun angezeigt wird. Auch hat sich der Titel von «*Statistiken von Studien*» zu «*Statistiken von Studie Name der Studie*» geändert.

Graphical user interface, application

Description automatically generated

### Filtern der Touchpointbewertungen

Die Tabelle der Touchpointbewertungen kann mit dem Inputfeld nach dem Namen der Touchpoints gefiltert werden.

Table

Description automatically generated

Zusätzlich kann mithilfe des Pfeilicons die Bewertung von oben nach unten und umgekehrt gefiltert werden.

Table

Description automatically generated

## Weiteres Vorgehen

In diesem Kapitel wird das weitere Vorgehen nach der IPA Probezeit festgelegt.

### Einführung vorbereiten

Der erarbeitete Code wird nach meiner IPA nochmals überprüft und bei Gelegenheit überarbeitet. Eventuell kommen noch Anpassungen aus Seite des Auftraggebers, welche aber später umgesetzt werden. Wenn alles in Ordnung ist, kann mit Einverständnis eines Teamkollegen mein Code produktiv geschalten werden.

### Schutzmassnahmen umsetzten

Während der Probezeit wurde jeden Tag mehrere Backups von meiner Arbeit auf diverse Medien abgespeichert. Zudem wurde der Quellcode in der Realisierungsphase auf Bitbucket gepushed. Dank der ausschlaggebenden Commitmessages kann entnommen werden, welchen Stand die Arbeit hat.

# Selbständigkeitserklärung

Die lernende Person bestätigt mit ihrer Unterschrift diese IPA aus Eigenleistung erbracht und nach den Vorgaben der Prüfungskommission Informatik Kanton Bern erstellt zu haben. Die Angaben im Arbeitsjournal entsprechen dem geleisteten Arbeitsaufwand. Es ist der lernenden Person bewusst, dass Falschaussagen, nicht korrekt deklarierte Arbeitsleistungen, nicht korrekt deklarierte Fremdinhalte (Plagiate), mit der Note 1 sanktioniert werden. Die lernende Person bestätigt mit ihrer Unterschrift ebenso, alle erforderlichen Mittel, Benutzer, Systeme, Betreuung durch die verantwortliche Fachkraft, die obligatorische Informationsveranstaltung, sowie die zwei Expertenbesuche erhalten/besucht zu haben.

Winkler Olivier (Kandidat) Ghilardelli Marco (Fachverantwortlich)

Shape

Description automatically generated with medium confidenceShape

Description automatically generated with medium confidence

# Abbildungsverzeichnis

[Abbildung 1 OneDrive: Datensicherung 9](https://sbb-my.sharepoint.com/personal/olivier_winkler_sbb_ch/Documents/ProbeIPA_Winkler_Olivier/07_Versionierung/IPA_Tag4_15012021/04_Nachmittag_Probe_IPA_Winkler_Olivier.docx#_Toc61608018)

[Abbildung 2 Macbook: Datensicherung 9](https://sbb-my.sharepoint.com/personal/olivier_winkler_sbb_ch/Documents/ProbeIPA_Winkler_Olivier/07_Versionierung/IPA_Tag4_15012021/04_Nachmittag_Probe_IPA_Winkler_Olivier.docx#_Toc61608019)

[Abbildung 3 OneDrive: Tägliches Backup - Erster Tag 11](https://sbb-my.sharepoint.com/personal/olivier_winkler_sbb_ch/Documents/ProbeIPA_Winkler_Olivier/07_Versionierung/IPA_Tag4_15012021/04_Nachmittag_Probe_IPA_Winkler_Olivier.docx#_Toc61608020)

[Abbildung 4: UseCase - Diagramm 40](https://sbb-my.sharepoint.com/personal/olivier_winkler_sbb_ch/Documents/ProbeIPA_Winkler_Olivier/07_Versionierung/IPA_Tag4_15012021/04_Nachmittag_Probe_IPA_Winkler_Olivier.docx#_Toc61608021)

[Abbildung 5: ERD der verwendeten Tabellen 42](https://sbb-my.sharepoint.com/personal/olivier_winkler_sbb_ch/Documents/ProbeIPA_Winkler_Olivier/07_Versionierung/IPA_Tag4_15012021/04_Nachmittag_Probe_IPA_Winkler_Olivier.docx#_Toc61608022)

[Abbildung 6: Diagramm Systemmodellierung 43](https://sbb-my.sharepoint.com/personal/olivier_winkler_sbb_ch/Documents/ProbeIPA_Winkler_Olivier/07_Versionierung/IPA_Tag4_15012021/04_Nachmittag_Probe_IPA_Winkler_Olivier.docx#_Toc61608023)

[Abbildung 7: Komponenten Diagramm 44](https://sbb-my.sharepoint.com/personal/olivier_winkler_sbb_ch/Documents/ProbeIPA_Winkler_Olivier/07_Versionierung/IPA_Tag4_15012021/04_Nachmittag_Probe_IPA_Winkler_Olivier.docx#_Toc61608024)

[Abbildung 8: Klassendiagramm 45](https://sbb-my.sharepoint.com/personal/olivier_winkler_sbb_ch/Documents/ProbeIPA_Winkler_Olivier/07_Versionierung/IPA_Tag4_15012021/04_Nachmittag_Probe_IPA_Winkler_Olivier.docx#_Toc61608025)

[Abbildung 9: Datenfluss 46](https://sbb-my.sharepoint.com/personal/olivier_winkler_sbb_ch/Documents/ProbeIPA_Winkler_Olivier/07_Versionierung/IPA_Tag4_15012021/04_Nachmittag_Probe_IPA_Winkler_Olivier.docx#_Toc61608026)

# Tabellenverzeichnis

[Tabelle 1: Quellenverzeichnis 5](#_Toc56153241)

[Tabelle 2: Abkürzungsverzeichnis und Glossar 6](#_Toc56153242)

# Quellenverzeichnis

|  |  |  |
| --- | --- | --- |
| Quelle | Besucht | Einsatz |
|  |  |  |
|  |  |  |

Tabelle 1: Quellenverzeichnis

# Abkürzungsverzeichnis und Glossar

|  |  |
| --- | --- |
| Begriff | Beschreibung |
|  |  |
|  |  |

Tabelle 2: Abkürzungsverzeichnis und Glossar

# Anhang