Realisierungsbericht

|  |  |
| --- | --- |
| **Status** | In Arbeit / In Prüfung / Abgeschlossen |
| **Projektname** | **S**chlaues**R**aum**B**uchungs**S**ystem (SRBS) |
| **Projektleiter** | Winkler Olivier |
| **Auftraggeber** | Blaser Sabine |
| **Autoren** | Winkelmann Domenico Winkler Olivier |
| **Verteiler** | SBB |

**Änderungskontrolle, Prüfung, Genehmigung**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Datum | Beschreibung, Bemerkung | Name oder Rolle |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Definitionen und Abkürzungen**

|  |  |
| --- | --- |
| Begriff / Abkürzung | Bedeutung |
|  |  |

**Referenzen**

|  |  |
| --- | --- |
| Referenz | Titel, Quelle |
| [1] |  |
| [2] |  |
| [3] |  |

**Inhaltsverzeichnis**

[1 Zusammenfassung 3](#_Toc25940358)

[2 Technische Detailspezifikation 3](#_Toc25940359)

[2.1 Systemdesign 3](#_Toc25940360)

[2.1.1 Struktur 3](#_Toc25940361)

[2.1.2 Beschreibung der Elemente 3](#_Toc25940362)

[2.2 Schnittstellendefinitionen 4](#_Toc25940363)

[2.3 Sicherheit (ISDS) 4](#_Toc25940364)

[2.4 Anforderungszuordnung 4](#_Toc25940365)

[3 Systemdokumentation 5](#_Toc25940366)

[3.1 Konfigurations-Dokumentation 5](#_Toc25940367)

[3.2 Benutzerhandbuch 9](#_Toc25940368)

[3.2.1 Systemübersicht 9](#_Toc25940369)

[3.2.2 Anwenderfunktionalität 9](#_Toc25940370)

[3.3 Supporthandbuch 11](#_Toc25940371)

[3.3.1 Massnahmen bei Benutzerproblemen 11](#_Toc25940372)

[3.3.2 Massnahmen bei technischen Problemen 11](#_Toc25940373)

[4 Systemtest 12](#_Toc25940374)

[4.1 Testspezifikation 12](#_Toc25940375)

[4.1.1 Kritikalität der Funktionseinheit 12](#_Toc25940376)

[4.1.2 Testanforderungen 12](#_Toc25940377)

[4.1.3 Testverfahren 12](#_Toc25940378)

[4.1.4 Testkriterien 12](#_Toc25940379)

[4.1.5 Testfälle 13](#_Toc25940380)

[4.2 Testprozedur 15](#_Toc25940381)

[4.2.1 Vorbereitung 15](#_Toc25940382)

[4.2.2 Voraussetzungen: 15](#_Toc25940383)

[4.2.3 Konfiguration: 15](#_Toc25940384)

[4.2.4 Durchführung: 15](#_Toc25940385)

[4.2.5 Nachbearbeitung: 15](#_Toc25940386)

[4.3 Testprotokoll 16](#_Toc25940387)

[4.3.1 Testobjekt 16](#_Toc25940388)

[4.3.2 Testresultate 17](#_Toc25940389)

[4.3.3 Testauswertung 18](#_Toc25940390)

[5 Weiterführung der Projektplanung 19](#_Toc25940391)

[5.1 Abgleich von Planung und tatsächlichem Verlauf der Phase Konzept 19](#_Toc25940392)

[5.2 Aktualisierung der Risikosituation 20](#_Toc25940393)

[5.3 Planung der nächsten Phase 21](#_Toc25940394)

[6 Phasenfreigabe 22](#_Toc25940395)

**Abbildungsverzeichnis**

[Abbildung 1 3](file:///C:\Users\E502439\AppData\Local\Temp\scp32980\home\3.%20Lehrjahr%202019%20-%202020\M306\Realisierung\Realisierungsbericht_Olivier_Domenico.docx#_Toc25940566)

[Abbildung 2 5](file:///C:\Users\E502439\AppData\Local\Temp\scp32980\home\3.%20Lehrjahr%202019%20-%202020\M306\Realisierung\Realisierungsbericht_Olivier_Domenico.docx#_Toc25940567)

[Abbildung 3 9](file:///C:\Users\E502439\AppData\Local\Temp\scp32980\home\3.%20Lehrjahr%202019%20-%202020\M306\Realisierung\Realisierungsbericht_Olivier_Domenico.docx#_Toc25940568)

[Abbildung 4 9](file:///C:\Users\E502439\AppData\Local\Temp\scp32980\home\3.%20Lehrjahr%202019%20-%202020\M306\Realisierung\Realisierungsbericht_Olivier_Domenico.docx#_Toc25940569)

[Abbildung 5 10](file:///C:\Users\E502439\AppData\Local\Temp\scp32980\home\3.%20Lehrjahr%202019%20-%202020\M306\Realisierung\Realisierungsbericht_Olivier_Domenico.docx#_Toc25940570)

[Abbildung 6 10](#_Toc25940571)

[Abbildung 7 11](file:///C:\Users\E502439\AppData\Local\Temp\scp32980\home\3.%20Lehrjahr%202019%20-%202020\M306\Realisierung\Realisierungsbericht_Olivier_Domenico.docx#_Toc25940572)

[Abbildung 8 11](file:///C:\Users\E502439\AppData\Local\Temp\scp32980\home\3.%20Lehrjahr%202019%20-%202020\M306\Realisierung\Realisierungsbericht_Olivier_Domenico.docx#_Toc25940573)

[Abbildung 9 11](file:///C:\Users\E502439\AppData\Local\Temp\scp32980\home\3.%20Lehrjahr%202019%20-%202020\M306\Realisierung\Realisierungsbericht_Olivier_Domenico.docx#_Toc25940574)

[Abbildung 10 19](file:///C:\Users\E502439\AppData\Local\Temp\scp32980\home\3.%20Lehrjahr%202019%20-%202020\M306\Realisierung\Realisierungsbericht_Olivier_Domenico.docx#_Toc25940575)

[Abbildung 11 21](file:///C:\Users\E502439\AppData\Local\Temp\scp32980\home\3.%20Lehrjahr%202019%20-%202020\M306\Realisierung\Realisierungsbericht_Olivier_Domenico.docx#_Toc25940576)

# Zusammenfassung

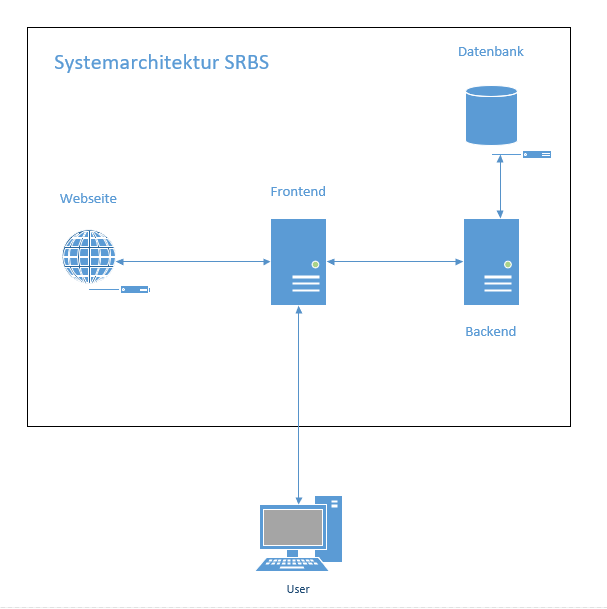
In diesem Dokument geht es um die Realisationsphase. Es werden die erreichten Punkte und Milestones aufgezählt. Das Dokument dient dazu unserem Kunden den Stand der Dinge zu präsentieren. Der Kunde kann die einzelnen Schritte von Anfang an sehen, wie sein Produkt umgesetzt wurde.

# Technische Detailspezifikation

## Systemdesign

Während der Umsetzung von unserem Konzept hat sich unsere Systemarchitektur nicht verändert. Zurzeit besteht immer noch ein Front- sowie Backend mitsamt einer Datenbank für die Behausung von Daten.

### Struktur



Abbildung

### Beschreibung der Elemente

**Frontend:**

**Module**: Frontend in Angular (Libraries von SBB werden verwendet)

**Backend & Datenbank:**

**Module**: Backend in Spring Boot, Datenbank in SQL angebunden

## Schnittstellendefinitionen

**Externe Schnittstelle**

* **ES1**: Zugriff auf Webseite (Benutzer ruft via URL unsere Webapplikation auf)

**Interne Schnittstelle**

* **IS1**: Frontend enthält benötigte Daten von Backend
* **IS2**: Backend wird von Frontend angesprochen.
* **IS3**: Backend bekommt gespeicherte Buchungen von der Datenbank
* **IS4**: Datenbank bekommt Anfrage für Datentransfer zum Backend
* **IS5**: Webseite wird dem Benutzer angezeigt, dahinter Frontend -> Backend -> Datenbank

## Sicherheit (ISDS)

Unsere Applikation verfügt über keine schützenswerten Daten. Da dieser Fall eintrifft hat unsere Applikation keine speziellen Anforderungen an Informationssicherheit & Datenschutz. Unsere Applikation ist im internen Firmennetz verfügbar und enthält Informationen über die Buchung von XY, Zeit der Buchung und die Person, die den Raum gebucht hat.

## Anforderungszuordnung

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **AFo.-Nr.** | **Anforderung (Stichwort)** | **ES1** | **IS1** | **IS2** | **IS3** | **IS4** | **IS5** | **Frontend** | **Backend & Datenbak** |
| A1 | Verfügbarkeit | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** |
| A1.1 | Performance |  | **X** | **X** | **X** | **X** | **X** | **X** | **X** |
| A2 | Datenmanagement |  | **X** | **X** | **X** | **X** | **X** | **X** | **X** |
| A3 | Buchungen erstellen | **X** |  | **X** | **X** | **X** | **X** | **X** | **X** |
| A4 | Buchungen bearbeiten | **X** |  | **X** | **X** | **X** | **X** | **X** | **X** |
| A5 | Validation |  | **X** |  |  |  | **X** | **X** | **X** |
| A6 | Code-Guidelines |  | **X** | **X** | **X** | **X** |  | **X** | **X** |
| A7 | Einhaltung Standard |  | **X** | **X** | **X** | **X** |  | **X** | **X** |
| A8 | Benutzerfreundlichkeit | **X** | **X** |  |  |  | **X** | **X** | **X** |
| A9 | Fehlerausgabe | **X** |  |  |  |  | **X** | **X** | **X** |
| A10 | Erweiterbarkeit |  |  | **X** | **X** | **X** |  | **X** | **X** |

# Systemdokumentation

## Konfigurations-Dokumentation

In diesem Abschnitt wird beschrieben, wie unsere Applikation auf- und umgesetzt wurde. Zuerst werden die verschiedenen benötigten Komponenten beschrieben und anschliessend die Umsetzung. Unsere Applikation wurde in Angular 8 und Spring Boot 2 geschrieben.

**Installation Angular 8**

Um ein Angularprojekt zu starten, wird dies zuerst installiert. Zudem installieren wird Komponente aus der Public-SBB-Library. Diese verwenden wird um einerseits den Standard einzuhalten und können so vorgefertigte Komponente in unsere Applikation einbauen. Dazu kommen die Icons, die standardmässig auf SBB-Webseiten verwendet werden.



Installation:

Installation von ersten verwendeten Package:

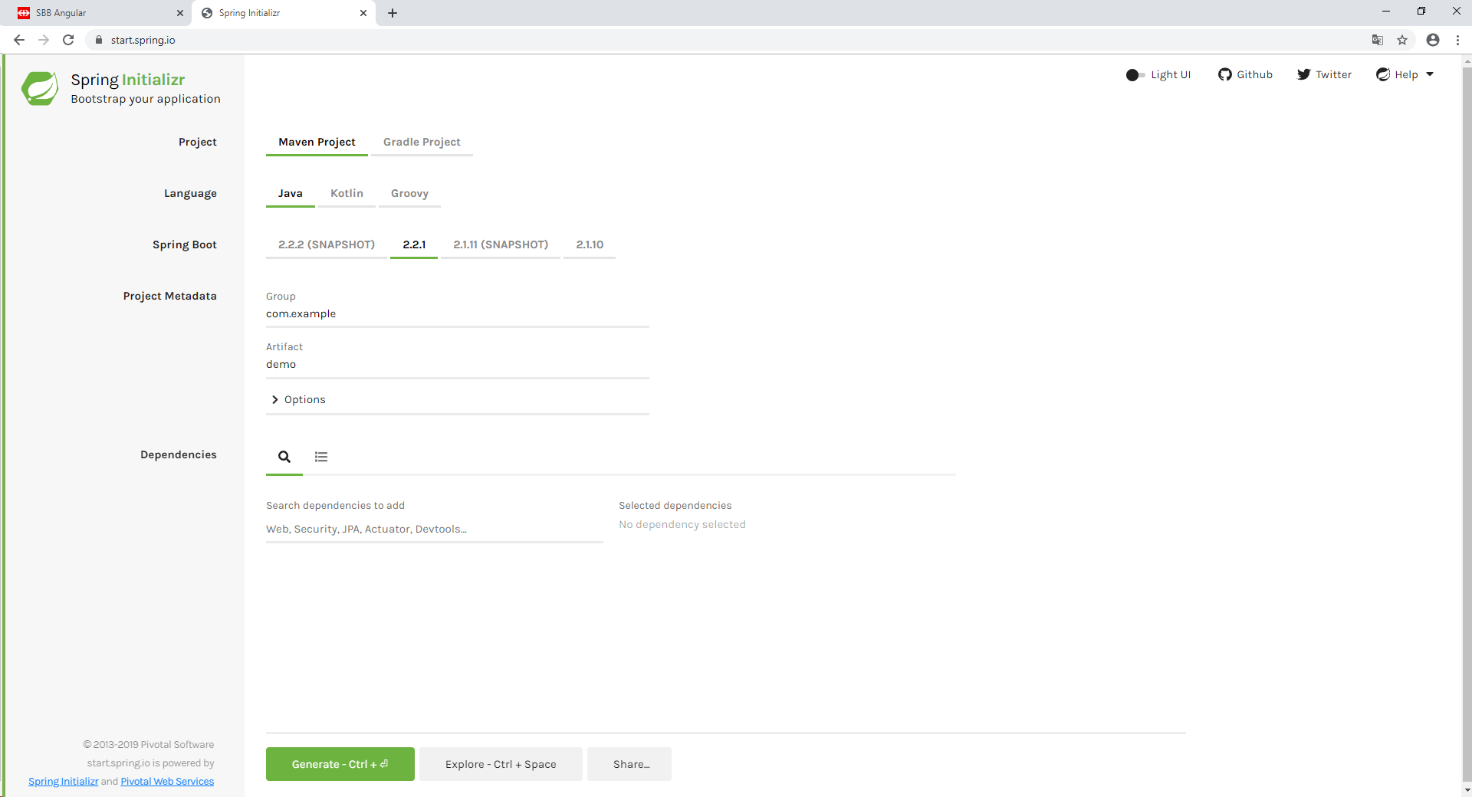


Installation von zweiten verwendeten Package:



Installation Spring Boot 2

Ein Spring Boot Projekt kann mit einem Initializer ganz einfach erstellt werden. Die verschiedenen Properties werden dort angegeben und anhand von diesen wird ein Projekt erstellt. Mit diesem Tool kann sehr schnell ein solches Projekt initialisiert werden.



Abbildung

**Umsetzung Frontend**

Im Frontend wurden die Komponenten aus den oben installierten Libraries verwendet. Hier werden nun einige aufgelistet.

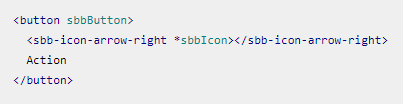
Kalender um ein Datum und Zeit auszuwählen. Wird für die Start- & Endzeit einer Buchung verwendet.



Tabelle um Daten anzuzeigen. Diese wird auf der Startseite verwendet, auf der der Benutzer alle Buchung einsehen kann.



Button. Dieser wird mehrfach verwendet. Der Vorteil von diesem Button ist, dass er vorgestylt ist und man Icons in seinen Body anheften kann.



Dies ist der Header. Dieser ist ganz oben anzutreffen. Beinhaltet z.B. das SBB-Logo

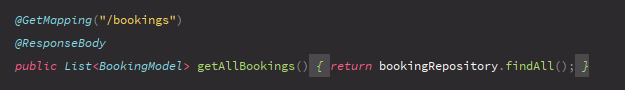


**Umsetzung Backend**

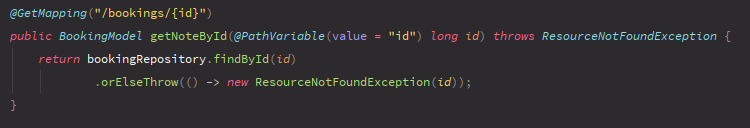
Das Backend ist in Spring Boot 2 geschrieben. Als Datenbank wird Hibernate verwendet. Hibernate ist ein ORM-Framework, dass von Java verwendet wird. Das Backend ist einem MVC aufgeteilt.

Der Controller verarbeitet die kommenden Anfragen vom Frontend. Diese werden direkt an eine Klasse weitergeleitet, die von JPARepository erbt. Diese Klasse gehört in die Spring DATA Abteilung. Mit dieser Klasse kann Hibernate eine Datenbank erstellen und diese verwalten.

GET-Request um alle Buchungen aus der Datenbank zu holen.

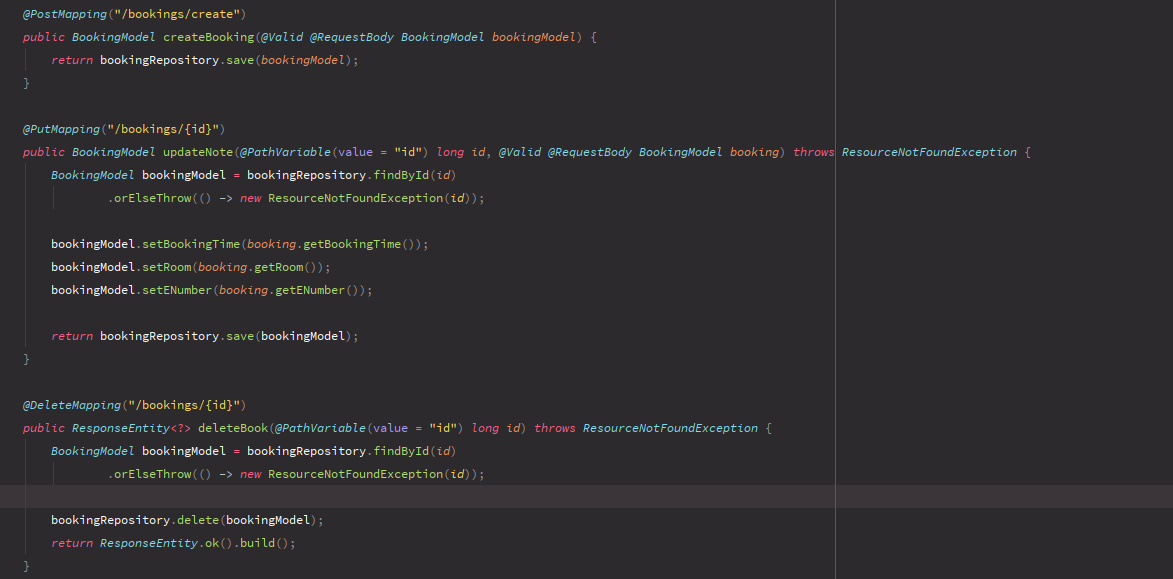


GET-Request um eine bestimmte Buchung mittels ID zu finden.

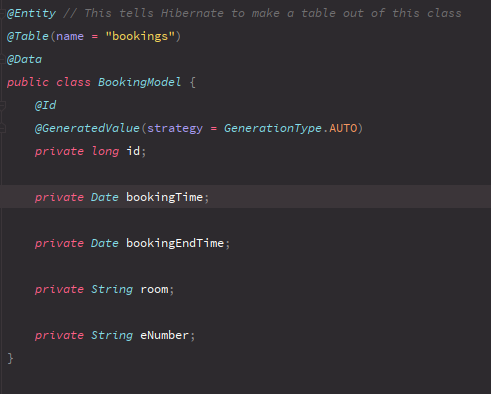


1. Methode POST-Request um eine Buchung zu erstellen.
2. Methode PUT-Request um eine Buchung zu aktualisieren.
3. Methode DELETE-Request um eine Buchung zu löschen.

Bei der POST-Methoden werden die Values vom Frontend ans Backend geschickt, welches diese dann richtig verarbeitet und in der Datenbank speichert. Die PUT-Methode funktioniert ähnlich und die DELETE-Methode löscht das Element mit der angegeben ID.



Untenstehend ist das Model aufgelistet. Hier sieht man alle Attribute, die eine Buchung hat.



## Benutzerhandbuch

### Systemübersicht

Mit unserer Applikation können Benutzer einen Sitzungsraum buchen. Der Benutzer kann eine Buchung für einen Raum erstellen, bearbeiten und löschen. Die Webseite ist nach SBB-Standard umgesetzt und kann so intern & extern ohne Probleme eingesetzt werden.

### Anwenderfunktionalität

**Buchungen ansehen**

Ein Bild, das Screenshot enthält.

Automatisch generierte BeschreibungWenn der Benutzer die Webseite besucht, sieht er alle Buchungen. Bei jeder Buchung ist die jeweilige Anfangszeit der Reservierung sowie das Ende hinterlegt. Zudem wird angezeigt welchen Raum um diese Zeit besetzt ist. Auch sieht man die Benutzernummer, mit der man entnehmen kann, wer die Buchung durchgeführt hat. Neben all diesen Angaben kann man mit dem Stiftsymbol die jeweilige Buchung bearbeiten und mit dem Papierkorb sogar löschen.

Abbildung

**Buchung erstellen**

Ein Bild, das Screenshot enthält.

Automatisch generierte BeschreibungWenn der Benutzer eine neue Buchung erfassen möchte, kann er dies indem er auf den roten Button oberhalb der Tabelle klickt, tun. Danach gelingt er in den Erstellungsmodus einer Buchung. Hier kann er die Zeiten, Raumnummer und seine Benutzernummer eingeben. Diese Änderungen werden dann auf der Übersicht angezeigt. Falls der Benutzer die Buchung nicht erstellen möchte, kann er diese verwerfen. Bevor dies aber geschieht wird eine Flashmessage angezeigt, in der der Benutzer gefragt wird, ob er wirklich seine Angaben verlieren möchte.

Abbildung

**Buchung bearbeiten**

Wenn der Benutzer auf den Stift klickt, kommt er in den Bearbeitungsmodus. Alle Angaben werden angezeigt und können individuell geändert werden. Die Buchung kann mit dem roten Button gespeichert werden. Falls der Benutzer Änderungen vorgenommen hat, welcher er nicht gespeichert hat, wird eine Flashmessage angezeigt in der gefragt wird ob die Änderungen wirklich ohne zu speichern verloren gehen sollen.

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

Abbildung

**Buchung löschen**

Falls der Benutzer eine Buchung löschen möchte, kann er dies tun indem er auf das Papierkorbsymbol klickt. Danach erscheint folgende Flashmessage, die ihn nochmals fragt ob wirklich die Buchung entfernt werden soll.

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

Abbildung

## Supporthandbuch

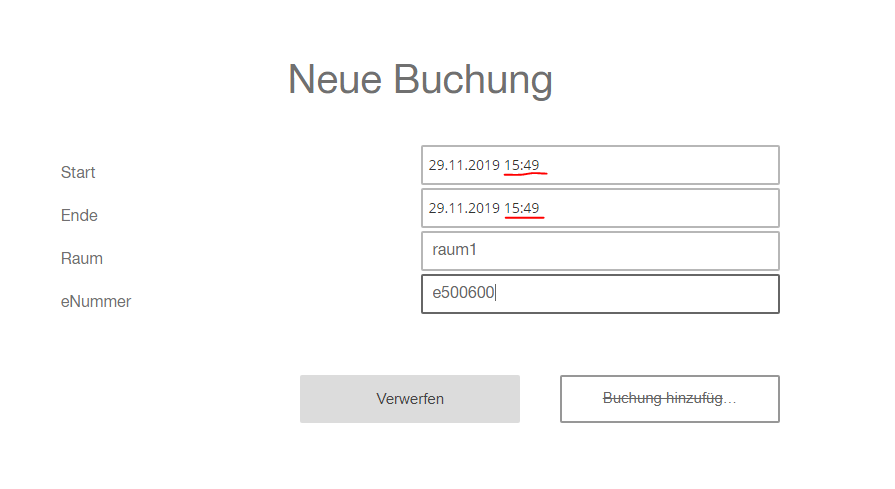
### Massnahmen bei Benutzerproblemen

Buchung kann nicht erstellt werden, da das Endatum kleiner als das Startdatum ist.  
Um dies zu lösen, muss das Endatum erhöht werden.



Abbildung

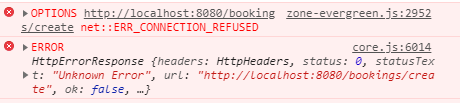
Buchung kann nicht erstellt werden, da das beide Daten und Zeiten identisch sind.   
Um dies zu lösen, muss die Endzeit oder das Enddatum erhöht werden.



Abbildung

### Massnahmen bei technischen Problemen

Backend -Verbindung schlägt fehl. Der Benutzer kann nun den Support kontaktieren.



Abbildung

# Systemtest

## Testspezifikation

### Kritikalität der Funktionseinheit

Für jeden Testfall ist eine Kritikalität definiert. Falls einer der Testfälle nicht erfolgreich durchgeführt werden kann, wird anhand von der Kritikalität entschieden ob dies zulässig ist oder das Produkt vor der Abgabe noch verbessert werden muss.

### Testanforderungen

Die Tests werden während der normalen Bedingungen getestet. Zudem wird geschaut was passiert, wenn verschiedene Komponente nicht richtig funktionieren z.B. das Backend nicht responsfähig ist.

### Testverfahren

Die Tests werden in drei Abschnitten aufgebaut. Zuerst wird ein Test die «Vorbereitung» durchlaufen. Gefolgt von der «Durchführung» und der «Auswertung».

### Testkriterien

**Abdeckungsgrad:**

Die Testfälle sind darauf abgestimmt alle Anforderungen auf ihre Funktionalität zu testen.

**Checklisten**:  
Die dazugehörigen Checklisten befinden sich direkt in den Testfällen falls benötigt.

**Ende-Kriterien**:  
Der Testfall gilt als abgenommen, wenn alle Kriterien erfolgreich bestanden worden sind.

### Testfälle

|  |  |
| --- | --- |
| **Nr.** | **1** |
| Kritikalität | Hoch |
| Anwendungsfall | Backend bekommt benötigte Daten von Datenbank |
| Ausgangssituation | Datenanforderung (Eine Buchung anzeigen) von Backend an Datenbank mittels GET-Request |
| Erwartetes Ergebnis | Die Datenbank gibt alle gespeicherten Daten zurück, welche durch das Backend angefordert wurden. |
| Benötigte Tools/Apps | * Frontend * Backend (+ Datenbank) |

|  |  |
| --- | --- |
| **Nr.** | **2** |
| Kritikalität | Hoch |
| Anwendungsfall | Frontend bekommt benötigte Daten von Backend |
| Ausgangssituation | Datenanforderung (Eine Buchung anzeigen) von Frontend an Backend mittels GET-Request via URL |
| Erwartetes Ergebnis | Das Backend gibt die benötigten Daten ans Frontend, welche zuvor vom Backend aus der Datenbank gelesen werden. Das Frontend zeigt anschliessend die gespeicherten Daten dem Benutzer an. |
| Benötigte Tools/Apps | * Frontend * Backend (+ Datenbank) |

|  |  |
| --- | --- |
| **Nr.** | **3** |
| Kritikalität | Hoch |
| Anwendungsfall | Frontend (Webseite) wird dem Benutzer korrekt angezeigt via Localhost (127.0.0.1) |
| Ausgangssituation | Der Benutzer ruft das Frontend resp. die Webseite mit einer URL auf und diese wird ihm dann vorgeführt. |
| Erwartetes Ergebnis | Der Benutzer kann die Webseite einsehen und mit dieser interagieren. |
| Benötigte Tools/Apps | * Frontend * Backend (+ Datenbank) |

|  |  |
| --- | --- |
| **Nr.** | **4** |
| Kritikalität | Hoch |
| Anwendungsfall | Benutzer erstellt eine neue Buchung |
| Ausgangssituation | Der Benutzer erstellt im GUI eine neue Buchung und diese soll korrekt hinterlegt werden. |
| Erwartetes Ergebnis | Der Benutzer erstellt seine Buchung mit allen benötigten Daten. Diese werden vom Frontend korrekt verarbeitet und ans Backend weitergeleitet. Dieses sendet die Daten an die Datenbank und werden dort hinterlegt. |
| Benötigte Tools/Apps | * Frontend * Backend (+ Datenbank) |

|  |  |
| --- | --- |
| **Nr.** | **5** |
| Kritikalität | Hoch |
| Anwendungsfall | Benutzer bearbeitet eine Buchung |
| Ausgangssituation | Der Benutzer editiert eine Buchung im GUI und die geänderten Daten werden gespeichert. |
| Erwartetes Ergebnis | Der Benutzer kann eine Buchung im GUI bearbeiten. Er kann alle Daten bearbeiten und diese werden anschliessend korrekt in der Datenbank gespeichert und somit ist die Buchung mit den Änderungen aktualisiert worden. |
| Benötigte Tools/Apps | * Frontend * Backend (+ Datenbank) |

|  |  |
| --- | --- |
| **Nr.** | **6** |
| Kritikalität | Hoch |
| Anwendungsfall | Benutzer löscht eine Buchung |
| Ausgangssituation | Der Benutzer löscht eine Buchung im GUI. |
| Erwartetes Ergebnis | Der Benutzer kann eine Buchung löschen. Diese wird dann auch in der Datenbank entfernt und somit auch nicht mehr angezeigt. |
| Benötigte Tools/Apps | * Frontend * Backend (+ Datenbank) |

## Testprozedur

Die Tests sind der Reihe nach von Nummer eins bis sechs abzuarbeiten. Funktioniert der erste Testfall nicht, funktionieren auch die nachfolgenden Testfälle nicht korrekt.

|  |  |
| --- | --- |
| **Nr.** | **Ausgangszustand der Testfälle** |
| 1 | Front- & Backend sind gestartet. Chrome ist installiert für optimale Anzeige |
| 2 | Front- & Backend sind gestartet. Chrome ist installiert für optimale Anzeige |
| 3 | Front- & Backend sind gestartet. Chrome ist installiert für optimale Anzeige |
| 4 | Front- & Backend sind gestartet. Chrome ist installiert für optimale Anzeige |
| 5 | Front- & Backend sind gestartet. Chrome ist installiert für optimale Anzeige |
| 6 | Front- & Backend sind gestartet. Chrome ist installiert für optimale Anzeige |

### Vorbereitung

### Voraussetzungen:

Für die Tests muss das Frontend sowie das Backend gestartet werden. Ohne Backend kann das Frontend keine Daten anzeigen und das Backend ist ohne Frontend benutzerunfreundlich. Dies ist der einzig benötigte Schritt in der Vorbereitung um die Tests durchzuführen.

### Konfiguration:

Die Konfiguration sollte so sein, wie sie in diesem Dokument konfiguriert wurde. Bei Abweichungen kann es zu Fehlern und Anzeigebugs kommen, da wichtige Ressourcen fehlen.

### Durchführung:

|  |  |
| --- | --- |
| **Nr.** | **Interaktion der Testfälle** |
| 1 – 6 | Auf dem Testgerät werden beide Applikationen (Front- & Backend) gestartet. Der Benutzer besucht die Webseite, welche Buchungen zeigt.  Benutzer sieht alle Buchungen. Front- & Backend zeigen alles korrekt an. |

### Nachbearbeitung:

|  |  |
| --- | --- |
| **Nr.** | **Nummer des Testes** |
| Erwartet Resultat | Beschreibung des Resultates, dass vor dem Test erwartet wurde. |
| Tatsächliches Resultat | Beschreibung des Resultates, dass beim Test herausgekommen ist. |
| Folgen | Beschreibung der Folgen, die entstanden sind bei Abweichungen. |

Zum Schluss dieses Kapitels wird eine Testauswertung verfasst. Diese Auswertig fällt je nach Testresultate besser oder schlechter aus. Zudem muss in dieser Verfassung geklärt werden ob bei anfälligen Fehlern das Projekt trotzdem dem Kunden übergeben werden kann und die Fehler schlussendlich nach dem Release mit einem Patch nachzubessern.

## Testprotokoll

### Testobjekt

|  |  |
| --- | --- |
| **Nr.** | **1** |
| Testobjekte | Front- & Backend inklusive Datenbank |
| Tester | Winkler Olivier & Winkelmann Domenico |
| Ort | Zimmer IET203 GIBB Bern |
| Datum | 26.11.2019 um 14:00 Uhr |

|  |  |
| --- | --- |
| **Nr.** | **2** |
| Testobjekte | Front- & Backend inklusive Datenbank |
| Tester | Winkler Olivier & Winkelmann Domenico |
| Ort | Zimmer IET203 GIBB Bern |
| Datum | 26.11.2019 um 14:10 Uhr |

|  |  |
| --- | --- |
| **Nr.** | **3** |
| Testobjekte | Front- & Backend inklusive Datenbank |
| Tester | Winkler Olivier & Winkelmann Domenico |
| Ort | Zimmer IET203 GIBB Bern |
| Datum | 26.11.2019 um 14:20 Uhr |

|  |  |
| --- | --- |
| **Nr.** | **4** |
| Testobjekte | Front- & Backend inklusive Datenbank |
| Tester | Winkler Olivier & Winkelmann Domenico |
| Ort | Zimmer IET203 GIBB Bern |
| Datum | 26.11.2019 um 14:30 Uhr |

|  |  |
| --- | --- |
| **Nr.** | **5** |
| Testobjekte | Front- & Backend inklusive Datenbank |
| Tester | Winkler Olivier & Winkelmann Domenico |
| Ort | Zimmer IET203 GIBB Bern |
| Datum | 26.11.2019 um 14:40 Uhr |

|  |  |
| --- | --- |
| **Nr.** | **6** |
| Testobjekte | Front- & Backend inklusive Datenbank |
| Tester | Winkler Olivier & Winkelmann Domenico |
| Ort | Zimmer IET203 GIBB Bern |
| Datum | 26.11.2019 um 14:50 Uhr |

**Versionen der Testkomponenten**

|  |  |
| --- | --- |
| **Komponente** | **Version** |
| Frontend | Angular 8 |
| Backend | Spring Boot 2.1 |
| Datenbank | Hibernate 5.3.10.Final |

### Testresultate

|  |  |
| --- | --- |
| **Nr.** | **1** |
| Erwartet Resultat | Die Datenbank gibt alle gespeicherten Daten zurück, welche durch das Backend angefordert wurden. |
| Tatsächliches Resultat | Das Frontend hat die Daten aus der Datenbank angezeigt. Das heisst, dass die Daten vom Backend gelesen werden konnten und anschliessend erfolgreich ans Frontend weitergeleitet wurden. |
| Folgen | Keine |

|  |  |
| --- | --- |
| **Nr.** | **2** |
| Erwartet Resultat | Das Backend gibt die benötigten Daten ans Frontend, welche zuvor vom Backend aus der Datenbank gelesen werden. Das Frontend zeigt anschliessend die gespeicherten Daten dem Benutzer an. |
| Tatsächliches Resultat | Der Tester konnte alle Daten auf der Startseite einsehen. Mit diesem Resultat lässt sich daraus ziehen, dass alle Daten korrekt verarbeitet wurden. |
| Folgen | Keine |

|  |  |
| --- | --- |
| **Nr.** | **3** |
| Erwartet Resultat | Der Benutzer kann die Webseite einsehen und mit dieser interagieren. |
| Tatsächliches Resultat | Die Webseite wurde korrekt angezeigt. Zu Testzwecken wurde Google Chrome verwendet, da in diesem Browser die Anzeige perfekt garantiert ist. Die Webseite war in allen Punkten voll funktionsfähig. |
| Folgen | Keine |

|  |  |
| --- | --- |
| **Nr.** | **4** |
| Erwartet Resultat | Der Benutzer erstellt seine Buchung mit allen benötigten Daten. Diese werden vom Frontend korrekt verarbeitet und ans Backend weitergeleitet. Dieses sendet die Daten an die Datenbank und werden dort hinterlegt. |
| Tatsächliches Resultat | Auf der Webseite liess sich eine neue Buchung hinzufügen. Diese wurde dann korrekt bei der Übersicht angezeigt, sowie in der Datenbank. |
| Folgen | Keine |

|  |  |
| --- | --- |
| **Nr.** | **5** |
| Erwartet Resultat | Der Benutzer kann eine Buchung im GUI bearbeiten. Er kann alle Daten bearbeiten und diese werden anschliessend korrekt in der Datenbank gespeichert und somit ist die Buchung mit den Änderungen aktualisiert worden. |
| Tatsächliches Resultat | Auf der Übersicht wurde eine spezifische Buchung angeklickt. Somit kommt man in die Bearbeitungsansicht. In dieser View wurden alle Daten korrekt angezeigt und konnten individuell angepasst werden und schlussendlich gespeichert werden. |
| Folgen | Keine |

|  |  |
| --- | --- |
| **Nr.** | **6** |
| Erwartet Resultat | Der Benutzer kann eine Buchung löschen. Diese wird dann auch in der Datenbank entfernt und somit auch nicht mehr angezeigt. |
| Tatsächliches Resultat | Auf der Übersicht wurde bei einer Buchung auf das Papierkorbsymbol geklickt. Nach der Nachfrage ob der Benutzer wirklich die Buchung löschen möchte, wurde die Buchung ordnungsgemäss gelöscht. |
| Folgen | Keine |

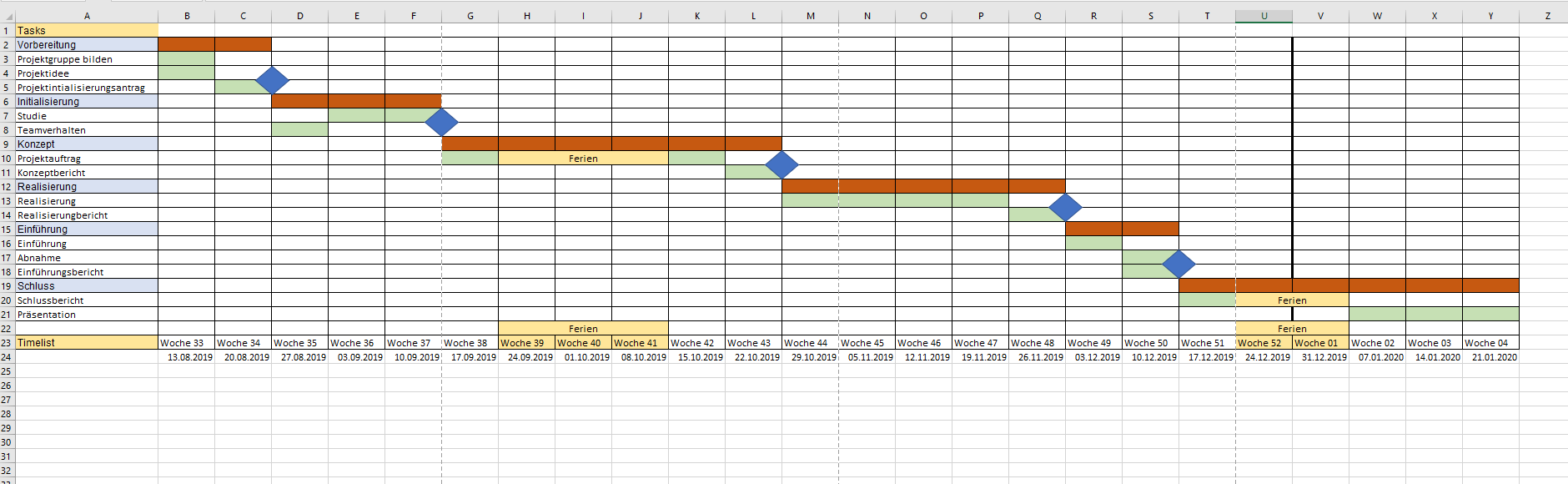
### Testauswertung

Die Testfälle sind mit der vollen Punktzahl durchgekommen. Es gab während des Testing’s keinerlei Abweichungen von unseren Definitionen. Die Perfektion während der Entwicklungsphase, die durch unser Projektteam vorgenommen wurde, zahlte sich aus, da die Applikation dank diesem Aufwand sorgenfrei läuft.

# Weiterführung der Projektplanung

## Abgleich von Planung und tatsächlichem Verlauf der Phase Konzept

Die Realisationsphase ist ganz genau nach unserer Planung durchlaufen. Da in einigen Kompetenzbereichen schon Wissen vorhanden war, konnten wir in diesen bereits mit wenig Zeitaufwand vieles erreichen und die übrige Zeit in Bereiche stecken, die uns als Projektteam noch nicht ganz bekannt waren.



Abbildung

## Aktualisierung der Risikosituation

Bereits im Konzeptbericht wurden die Risiken überarbeitet. Die Konzeptphase war die vorherige Phase. In der Realisation haben die Risiken sich nicht gross verändert. Dazugekommen sind während der Phase keine Risiken. Jedoch hat ein Risiko abgenommen.

**Risiken Konzeptphase**

|  |  |  |  |
| --- | --- | --- | --- |
| **Beschreibung** | **Wahrscheinlichkeit** | **Auswirkung** | **Massnahme** |
| **Zu hoher Aufwand** | 2 | Gross | Aufwand wurde geschätzt und verteilt auf Teammitglieder. Stände Statusmeldung im Projekt |
| **Zeitplan ungenau, Termin können nicht eingehalten werden** | 2 | Gross | Genügend Spielraum einplanen. Zeitplan kann durcheinander geraten bei Verzögerungen durch Ausfälle. An anderen Tagen verlorene Zeit aufholen |
| **Falsche Zielsetzung** | 2 | Mittel | Zieldefinition nach **SMART** |
| **Zu wenig Wissen** | 2 | Mittel | Knowhowtransfer, Knowhow aneignen, welches für das Projekt benötigt wird |
| **Ausfälle durch Krankheiten** | 2 | Klein | Aufgaben des Ausgefallenen werden auf andere Teammitglieder verteilt um grössere Verzögerungen zu verhindern. |
| **Ausfälle durch Entwicklungsumgebung** | 3 | Gross | Wöchentliches Backup von Entwicklungsumgebung erstellen. Diese ist sehr anfällig gegen Störungen. |
| **Unklarheiten** | 2 | Mittel | Bei Unklarheiten werden diese so schnell wie möglich geklärt |
| **Missverständnisse im Entwicklungsteam** | 3 | Mittel | Möglichst hoher Austausch zwischen Teammitglieder um dies zu verhindern. |
| **Zu hohe Kosten** | 1 | Gross | Aufwand wird ständig mit den Kosten verglichen |

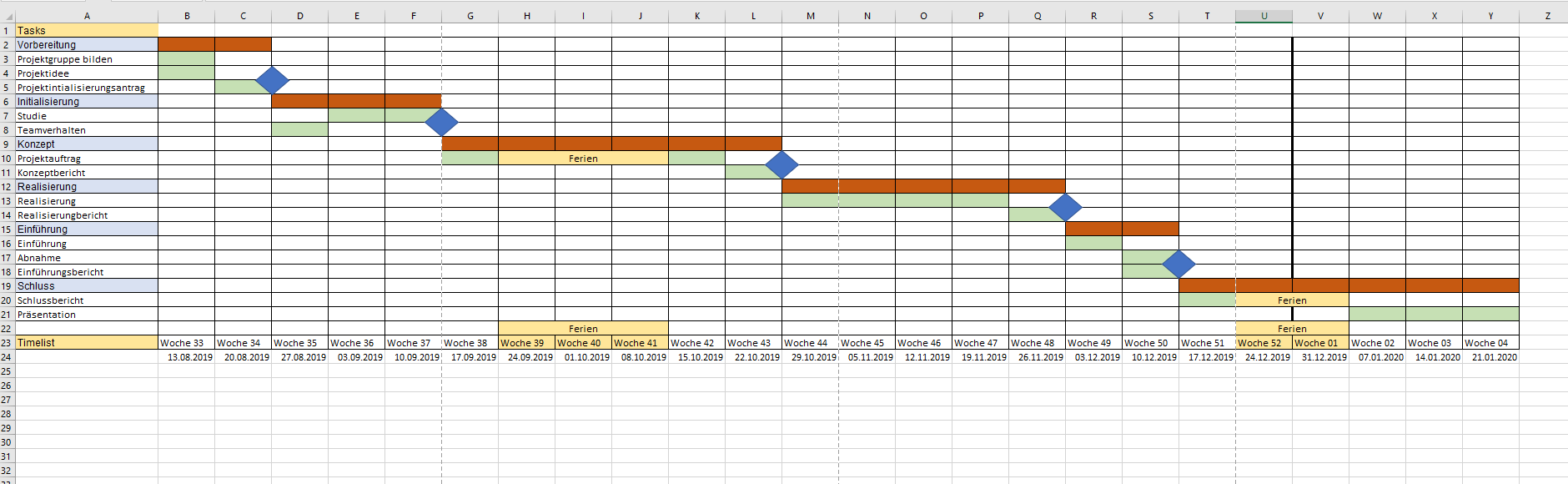
**Risiken Realisationsphase**

|  |  |  |  |
| --- | --- | --- | --- |
| **Beschreibung** | **Wahrscheinlichkeit** | **Auswirkung** | **Massnahme** |
| **Zu hoher Aufwand** | 2 | Gross | Aufwand wurde geschätzt und verteilt auf Teammitglieder. Stände Statusmeldung im Projekt |
| **Zeitplan ungenau, Termin können nicht eingehalten werden** | 2 | Gross | Genügend Spielraum einplanen. Zeitplan kann durcheinander geraten bei Verzögerungen durch Ausfälle. An anderen Tagen verlorene Zeit aufholen |
| **Falsche Zielsetzung** | 2 | Mittel | Zieldefinition nach **SMART** |
| **Zu wenig Wissen** | 2 | Mittel | Knowhowtransfer, Knowhow aneignen, welches für das Projekt benötigt wird |
| **Ausfälle durch Krankheiten** | 2 | Klein | Aufgaben des Ausgefallenen werden auf andere Teammitglieder verteilt um grössere Verzögerungen zu verhindern. |
| **Ausfälle durch Entwicklungsumgebung** | 3 | Gross | Wöchentliches Backup von Entwicklungsumgebung erstellen. Diese ist sehr anfällig gegen Störungen. |
| **Unklarheiten** | 2 | Mittel | Bei Unklarheiten werden diese so schnell wie möglich geklärt |
| **Missverständnisse im Entwicklungsteam** | 3 | Mittel | Möglichst hoher Austausch zwischen Teammitglieder um dies zu verhindern. |

## Planung der nächsten Phase

Der Projektplan für die kommenden Phasen ist folgendermassen aus:

Nun folgen nur noch 2 Phasen bis zum Ende dieses Projektes. Die nächste Phase wird die Einführung sein. In der Einführung wird unser Produkt bei den Kunden eingeführt und das Feedback von Ihnen genauer untersucht & korrigiert. Die dann darauffolgende Phase wird der Schluss sein. In dieser Phase wird ein Bericht über das ganze Projekt verfasst. Zudem werden in dieser Phase die Präsentationen der verschiedenen Projekte stattfinden.

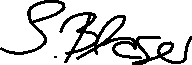


Abbildung

# Phasenfreigabe

Hiermit bestätigt der Auftraggeber die Freigabe des Projekts in die nächste Phase:





­­­­­­­­­­­­­­­­­­­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Der Auftraggeber Der Projektleiter

(Ort, Datum, Unterschrift) (Ort, Datum, Unterschrift)