




# REQUIREMENTS SPECIFICATION

## MHC-PMS – TEAM ORANGE

14. April 2020

Adrian Berger, Oliver Kunz, Fabian K  ng, Lorenz Sieber, Jonas Herzog



## Inhaltsverzeichnis

1.	Preface.....	2
1.1	Zielpublikum .....	2
1.2	Änderungsprotokoll.....	2
2.	Einführung.....	2
3.	Glossary .....	3
4.	User Requirements.....	4
4.1	Use Case 3 – Details .....	5
4.2	Use Case 11 – Details.....	6
5.	System architecture .....	8
6.	System requirements specification .....	9
6.1	Funktionale Anforderungen .....	9
6.2	Nicht-funktionale Anforderungen .....	9
7.	System model .....	10
8.	System evolution .....	11
8.1	Soll-Zustand .....	11
8.2	Mögliche Erweiterungen .....	11
9.	Testing .....	12
9.1	Unit Tests .....	12
9.2	Integration Tests.....	12
9.3	Reviews.....	12
9.4	GUI und System Tests .....	12
10.	Appendices.....	12
11.	Anhang .....	13
11.1	Tabellenverzeichnis .....	13
11.2	Abbildungsverzeichnis .....	13

## 1. Preface

Dieses Dokument ist für Kunden, Manager und Ingenieure, welche in diesem Projekt mitwirken.

### 1.1 Zielpublikum

Benutzerrolle	Beschreibung
Systemkunden	Definiert die Anforderungen und liest diese um zu prüfen, ob diese den Anforderungen entsprechen. Kunden spezifizieren Änderungen an den Anforderungen.
Manager	Verwenden das Anforderungsdokument um ein Angebot für das System zu planen und um den Systementwicklungsprozess zu planen.
Systemingenieure	Verwenden die Anforderungen, um zu verstehen, welches System entwickelt werden soll.
Systemtestingenieure	Verwenden die Anforderungen, um Validierungstests für das System zu entwickeln.
Systemwartungsingenieure	Verwenden die Anforderungen, um das System und deren Beziehungen zwischen einander zu verstehen.

Tabelle 1: Zielpublikum

### 1.2 Änderungsprotokoll

Version	Autor	Änderungen	Datum
0.1	Lorenz Sieber	Entwurf: Kapitel System evolution	03.04.2020
0.2	Lorenz Sieber	Entwurf: Kapitel Testing	06.04.2020
0.3	Lorenz Sieber	Kleinere Anpassungen	07.04.2020
0.4	Jonas Herzog	Kapitel: System architecture	10.04.2020
0.5	Jonas Herzog	Kapitel: System models & System requirements specification, Glossar & Rechtschreibfehler	11.04.2020
0.6	Oliver Kunz	User requirements	12.04.2020
0.7	Adrian Berger	Entwurf: Kapitel Preface & Introduction	13.04.2020
0.8	Adrian Berger	Aktivitätendiagramm	13.04.2020
0.9	Fabian Küng	Kapitel: System requirements & System architecture	14.04.2020
1.0	Jonas Herzog	Kleine Korrekturen & Abschluss	14.04.2020

Tabelle 2: Änderungskontrolle

## 2. Einführung

Eine regionale Gesundheitsbehörde ist auf der Suche nach einem Patientenmanagementsystem für mentale Krankheiten. Dabei soll es die zwei folgenden Bedürfnisse abdecken:

- Das Vereinfachen der Behandlung von Patienten durch die medizinischen Angestellten (Ärzte und Gesundheitsbesucher) mithilfe von wichtigen Informationen
- Das Unterstützen von Patienten und deren Angehörigen im Umgang mit der Krankheit
- 

Das hier beschriebene System legt den primären Fokus auf den (*ambulant*) *Patienten*, welcher an einer *Depression* leidet.

Zusätzlich soll es aber auch für Ärzte nutzbar sein, welche über das System eine 1-n Beziehung zu den Patienten haben sollen. Für die Patienten untereinander gibt es keine spezifischen Funktionalitäten.

Im Umfang dieses Projekts sind keine Schnittstellen zu anderen Systemen vorgesehen. So läuft die Applikation grundsätzlich eigenständig. Bei Bedarf könnte man Verknüpfungen zu anderen Systemen in Form eines weiteren Projekts einbauen.

### 3. Glossary

Wort/Abkürzung	Beschreibung
Angular	Front-End-Webapplikationsframework, geschrieben in der Programmiersprache TypeScript.
Backend	Bereich der Applikation für einen bestimmten Benutzerbereich (Ärzte/Therapeuten).
Branch	Entwicklungszweig, worauf ein Entwickler unabhängig von anderen weiter arbeiten kann.
Cloud	Teile der Infrastruktur, die über das Internet zugänglich sind.
CSS	Cascading Style Sheets
Datenbank	System zur Datenspeicherung und deren Abfrage.
DSGVO	Datenschutz-Grundverordnung
Evergreen Browsers	Browser, welche automatisch auf zukünftige Versionen aktualisiert werden.
Framework	Programmiergerüst, welches einen Rahmen für eine Anwendung zur Verfügung stellt.
Frontend	Bereich der Applikation, welcher für normale Patienten sichtbar ist.
GUI	Graphical user interface, grafische Benutzeroberfläche.
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
Java	Die objektorientierte Programmiersprache, die für diese Applikation unter anderem verwendet wird.
JS	JavaScript
Load-Balancing	Lastverteilung auf verschiedene Server.
Merge	Einen Branch zurück in den Hauptbranch verschieben.
MHC	Mental Health Care
Open-Close Prinzip	Programm so entwerfen, dass Erweiterungen einfach möglich sind.
PMS	Patienten Management System
Spring	Java-Framework
SSL	Secure Sockets Layer
Responsive design	Design, welches so konzipiert ist, dass es im Layout für die jeweiligen Endgeräte passend ist.
Unit Test	Kleine, einzelne Funktionen der Applikation werden mit Unit Tests getestet.
Vaadin	Webframework für Java

Tabelle 3: Glossary

#### 4. User Requirements

Im Use-Case-Diagramm wurden allen Use-Cases aus Task 01 abgebildet und unten in Verbindung mit Ihren Aktoren gesetzt. Wichtig für uns ist die Umsetzung der im Diagramm abgebildeten Use-Cases. Es ist durchaus denkbar, dass im Verlauf der Entwicklung des PMS noch Abhängigkeiten von bestimmten Use-Cases angepasst werden müssen.

Nr.	Use-Case
01	Neuen Termin erfassen
02	Informationsseite
03	Stimmung erfassen
04	Neuen Patienten im System registrieren
05	Therapeut bestätigt Patiententermin
06	Therapeut fügt neues Medikament für den Patienten hinzu
07	Eintrag ins Tätigkeitsbuch
08	Erinnerung an die Medikamenteneinnahme
09	Im Notfall einen Therapeuten anrufen
10	Notruf wählen
11	Chatfunktion mit Fachperson

Tabelle 4: Nummerierung Use Cases

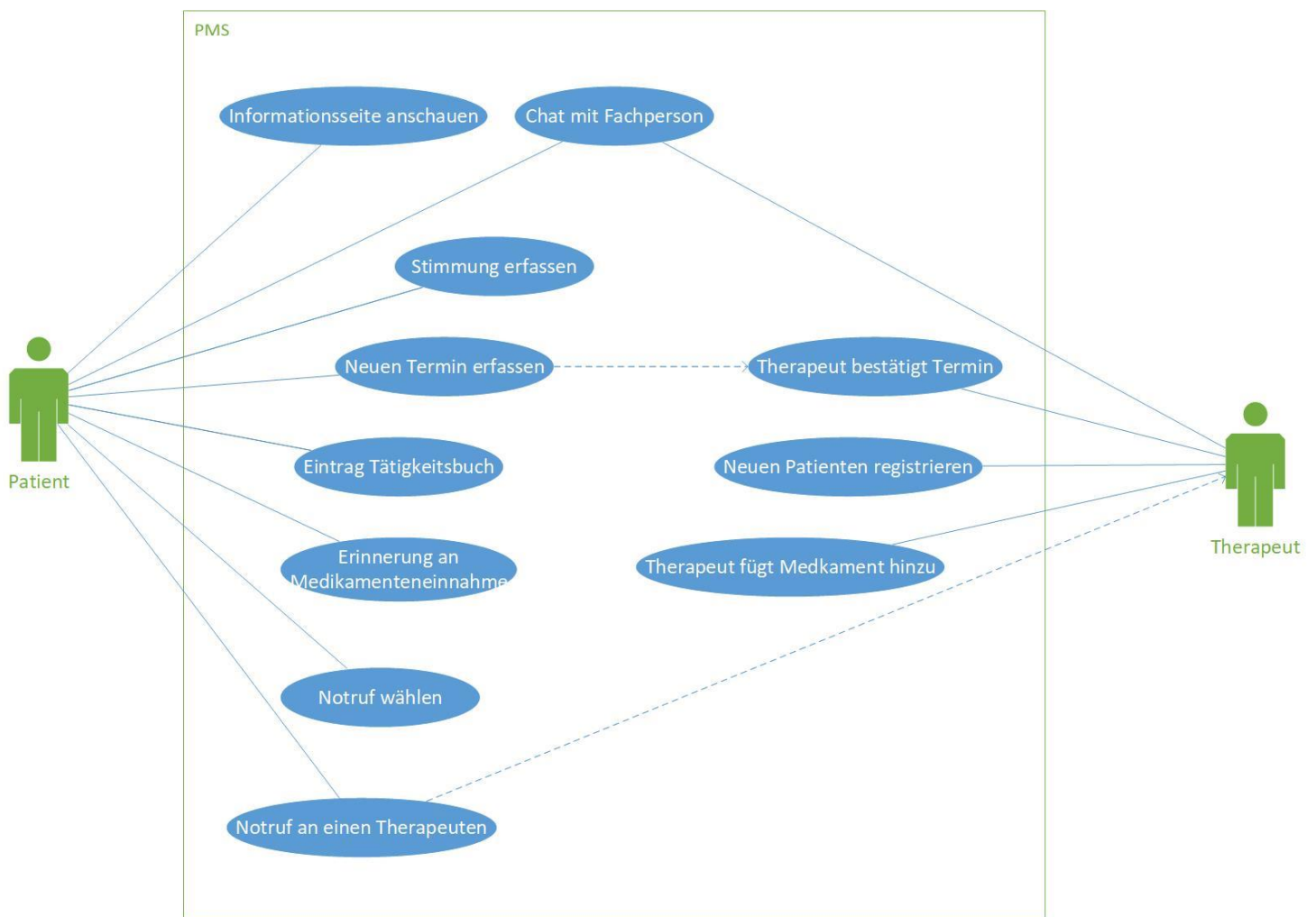


Abbildung 1: Use Case-Diagramm

#### 4.1 Use Case 3 – Details

Was	Text
Nr. und Name	03 Stimmung erfassen
Szenario	Der Patient muss jeden Tag seine Stimmung im PMS erfassen.
Kurzbeschreibung	Am Abend öffnet der Patient das PMS und trägt im heutigen Tag ein wie er seine Stimmung wahrgenommen hat und schreibt vielleicht noch einen Kommentar.
Beteiligte Akteure	Patient (Therapeut schaut sich das Tagebuch anschliessend an)
Auslöser/Vorbedingung	Patient ist vom Therapeut aufgefordert seine Stimmung täglich zu erfassen um den Fortschritt zu tracken.
Ergebnisse/Nachbedingung	Der Therapeut erhält eine Übersicht über den Verlauf der Stimmung des Patienten während der Behandlungsdauer.

Tabelle 5: Allgemeine Informationen zu Use Case 3

#### Ablauf

Nr.	Wer	Was
01	Patient	Öffnet das PMS um seinen täglichen Eintrag über seine Stimmung zu machen.
02	Patient	Beschreibt seine Stimmung anhand von verschiedenen Smileys und fügt optional noch einen Kommentar hinzu.
03	Patient & Therapeut	Sehen den Verlauf der Stimmung des Patienten über den Behandlungszeitraum.

Tabelle 6: Ablauf für Use Case 3

Folgendes Aktivitätsdiagramm zeigt diesen Ablauf:

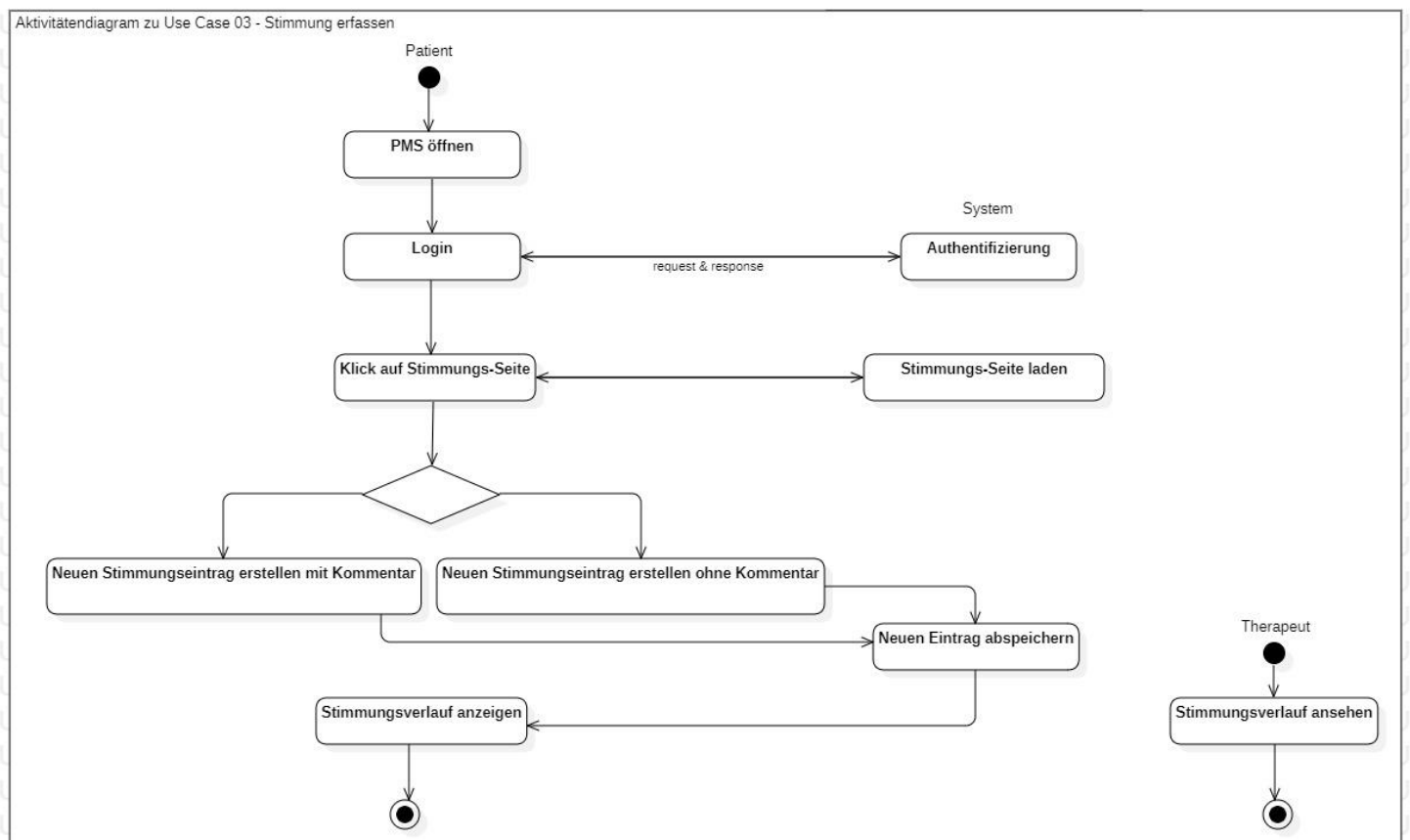


Abbildung 2: Aktivitätsdiagramm für Use Case 3

**Ausnahmen/Varianten**

Nr.	Wer	Was
01	Patient	Wenn die Stimmung nach 20:00Uhr noch nicht erfasst ist, könnte der Patient eine Push-Benachrichtigung erhalten.
02	Patient	Vergisst den Eintrag zu machen, so muss er diesen am nächsten Tag nachholen und die Verspätung wird im PMS vermerkt.
03	Therapeut	Kann den Patienten auffordern, detaillierter über seine Stimmung zu schreiben.

Tabelle 7: Ausnahmen für Use Case 3

**4.2 Use Case 11 – Details**

Was	Text
Nr. und Name	11 Chatfunktion mit Fachperson
Szenario	Der Patient ist sich unsicher bei seiner Behandlung oder will den Therapeuten über etwas informieren.
Kurzbeschreibung	Der Patient ist sich bezüglich der Wirkung/Nutzen eines Medikaments unsicher und öffnet das PMS, um mit seinem Therapeuten zu chatten. Der Therapeut kann dem Patienten gleich antworten und ihm die nötigen Informationen mitteilen.
Beteiligte Akteure	Patient & Therapeut
Auslöser/Vorbedingung	Eine Sache ist unklar oder der eine der Parteien will Informationen austauschen.
Ergebnisse/Nachbedingung	Der Patient ist beruhigt und weiss was er tun muss. Der Therapeut ist über die Lage des Patienten informiert.

Tabelle 8: Allgemeine Informationen zu Use Case 11

**Ablauf**

Nr.	Wer	Was
01	Patient	Ist Unsicher über die Wirkungen eines Medikaments.
02	Patient	Öffnet den Chat mit dem Therapeuten im PMS.
03	Patient	Schreibt dem Therapeuten sein Anliegen.
04	Therapeut	Erhält eine Benachrichtigung im PMS, dass eine neue Nachricht von Patient vorliegt.
05	Therapeut & Patient	Konversation im PMS.
06	Therapeut & Patient	Sind informiert über die gegenseitige Lage.

Tabelle 9: Ablauf für Use Case 11

Folgendes Aktivitätsdiagramm zeigt diesen Ablauf:

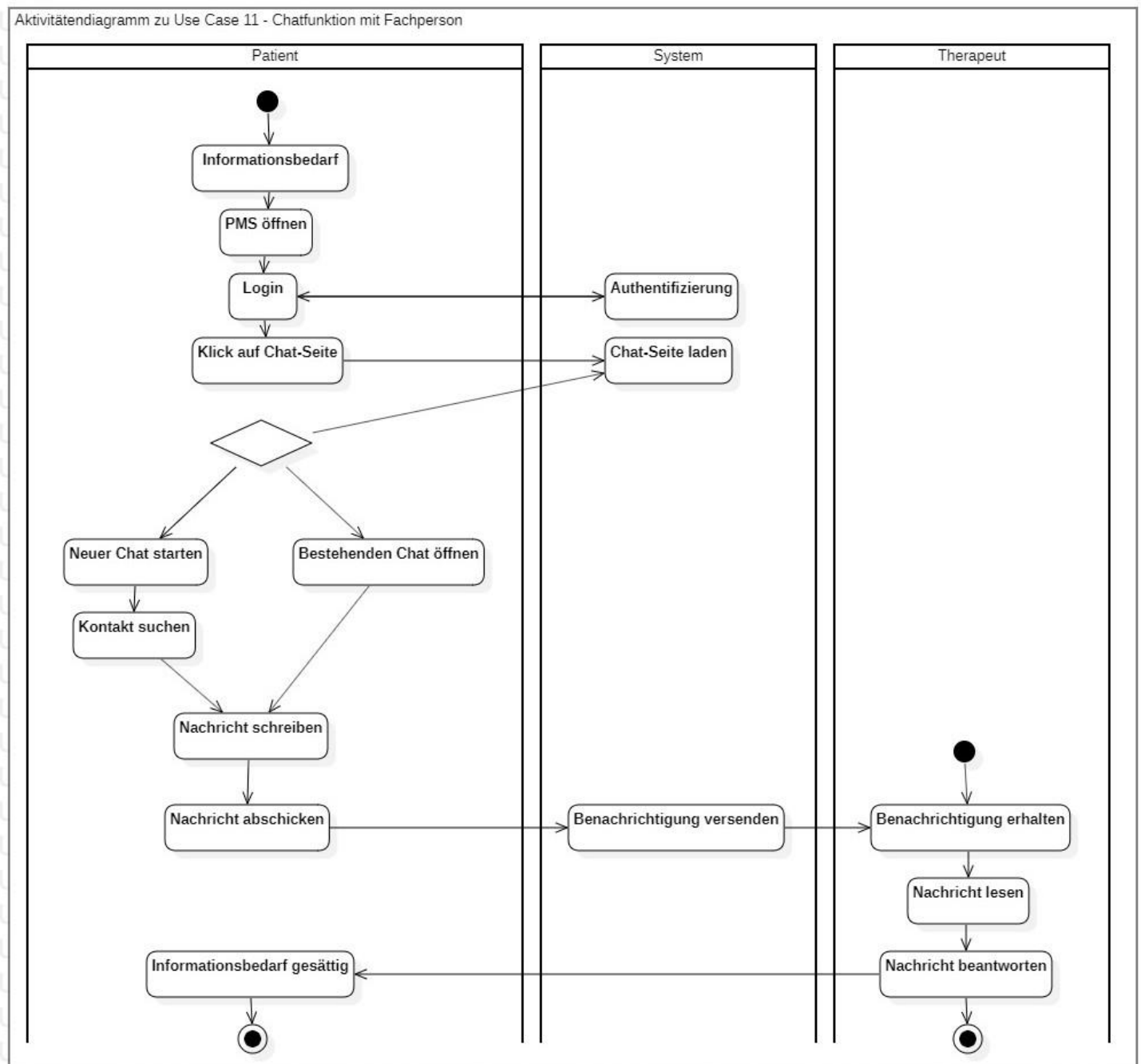


Abbildung 3: Aktivitätsdiagramm für Use Case 11

### Ausnahmen/Varianten

Nr.	Wer	Was
01	Therapeut	Die Anfrage des Patienten ist kompliziert und daher ruft ihn der Therapeut direkt an.
02	Patient	Der Therapeut antwortet nicht sofort und daher ruft er den Therapeuten direkt an.
03	Patient	Die Antworten im Chat reichen ihm nicht aus und er vereinbart einen Termin mit dem Therapeuten.

Tabelle 10: Ausnahmen für Use Case 11



## 5. System architecture

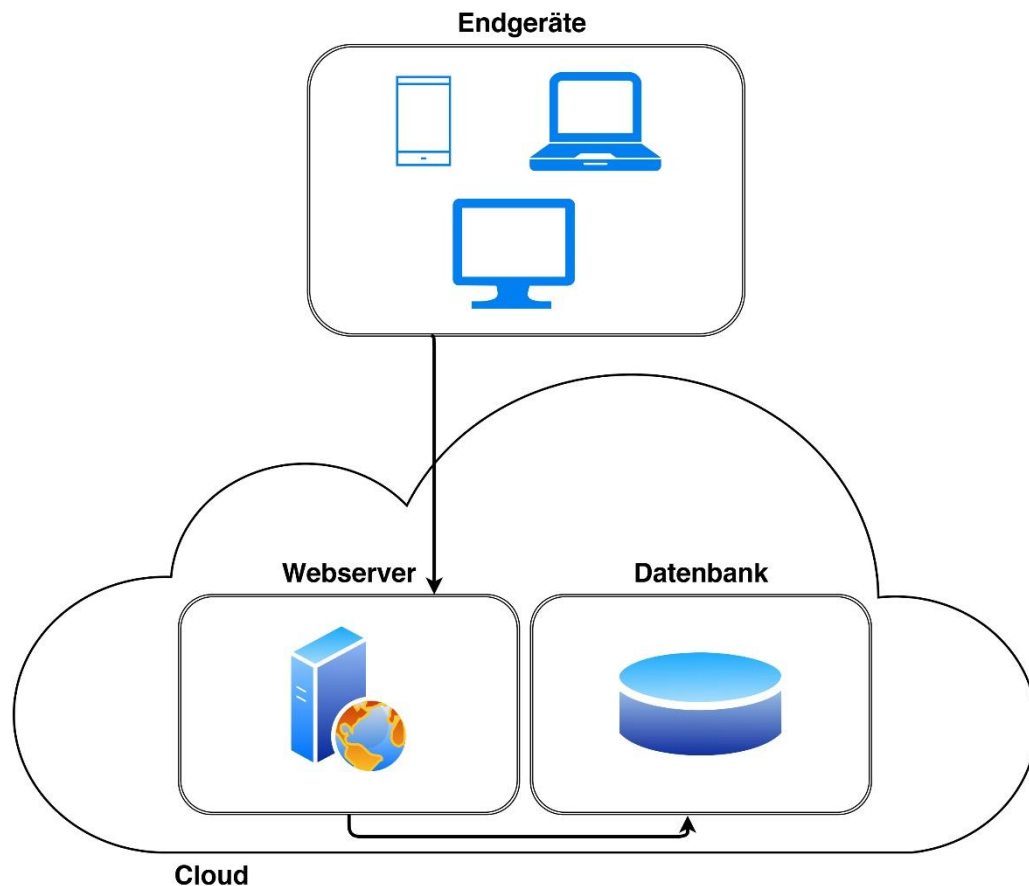


Abbildung 4: Systemarchitektur

Bei der System Architektur ist zu beachten, dass sich diese noch ändern kann. Während des technischen Refinement der Stories, kann es noch zu Änderungen kommen. Dies ist auch dem geschuldet, da vorgeschriebene Technologien/Einschränkungen noch nicht bekannt sind.

### Frontend

Das Frontend der Applikation wird eine Webapplikation (HTML, CSS, JS) sein, welche über Laptops, Computer und Smartphones über einen modernen Browser abrufbar ist. Der Aufruf hier geschieht über das HTTP-Protokoll mit einer SSL Verschlüsselung. Damit die Applikation überall identisch ist und die gleichen Funktionen bietet, wird ein responsive Design erstellt. Das Frontend kann mit Java (bspw. Vaadin) oder auch Angular umgesetzt werden. Das Frontend wird entsprechend auf einem Webserver zur Verfügung gestellt.

### Backend

Die Business Logik wird in Java geschrieben. Jegliche Daten werden nur im Backend auf der Datenbank abgelegt und nicht auf einem Endgerät des Kunden. Installationen sind keine notwendig. Zu prüfen ist hier, ob ein Framework wie bspw. Spring verwendet wird.

### Datenbank

Bei der Datenbank ist zu beachten, dass eine persistente Datenbank verwendet wird und nicht beispielsweise eine In-Memory-Datenbank. Die Daten müssen auch nach einem Neustart oder Absturz verfügbar sein. Die Datenbank ist bei dieser Applikation zentral und muss somit hochverfügbar sein. Auch zu beachten ist, dass Daten in der Datenbank datenschutztechnisch kritisch sind und somit hohen Sicherheitsanforderungen unterliegt.

## 6. System requirements specification

### 6.1 Funktionale Anforderungen

Nr.	Beschreibung	Use Case
01	Ein Patient kann über das App einen freien Termin beim Therapeuten suchen.	Use-Case-01
02	Ein zuvor gesuchter und freier Termin, kann beim Therapeuten angefragt werden.	Use-Case-01
03	Dem Patienten wird eine Informationsseite angezeigt, bei welcher er Informationen zu seiner Diagnose findet.	Use-Case-02
04	Der Benutzer kann seine Stimmung in regelmässigen Abständen erfassen (Bild, Text, ...).	Use-Case-03
05	Der Benutzer kann seine erfassten Stimmungen analysieren.	Use-Case-03
06	Ein Therapeut kann einen neuen Benutzer erstellen. Der Patient selbst soll keine Registrierung durchführen können. Ihm wird das Login schlussendlich überreicht.	Use-Case-04
07	(Out of scope) Ein bestehender Patient kann bearbeitet werden.	Use-Case-04
08	(Out of scope) Ein bestehender Patient kann gelöscht werden.	Use-Case-04
09	Ein Patient sowie Therapeut kann sich bei der App einloggen.	Voraussetzung für alle Use-Cases
10	Ein vom Patienten angefragter Termin kann durch den Therapeuten bestätigt werden.	Use-Case-05, Use-Case-01
11	Der Therapeut kann dem Patienten ein neues Medikament zur Einnahme hinzufügen.	Use-Case-06
12	(Out of scope) Der Therapeut kann die Einnahme des Medikaments bearbeiten.	Use-Case-06
13	Ähnlich wie bei der Stimmung erfassen, kann der Patient seine Tätigkeiten in regelmässigen Abständen erfassen.	Use-Case-07, Use-Case-03
14	Ähnlich wie bei der Stimmung erfassen, kann der Patient seine Tätigkeiten analysieren.	Use-Case-07, Use-Case-03
15	Dem Benutzer wird eine Benachrichtigung angezeigt, dass er ein Medikament, welches ihm der Arzt eingetragen hat, einnehmen muss.	Use-Case-08, Use-Case-06
16	Ein Patient kann die Einnahme des Medikamentes bestätigen.	Use-Case-08, Use-Case-06
17	Der Patient muss seinen Therapeuten im Notfall anrufen können.	Use-Case-09
18	Der Patient muss einen Notruf-Button haben, bei welchem er den Notfall (Zentrale) kontaktieren kann.	Use-Case-10
19	Dem Patienten sowie Therapeuten steht ein Chat zur Verfügung, bei welchen sie sich unterhalten/beraten können.	Use-Case-11
20	Dem Patienten steht eine Liste seiner/seines Therapeuten zur Verfügung.	Use-Case-11, Use-Case-09
21	Dem Therapeuten steht eine Liste seiner Patienten zur Verfügung.	Use-Case-11, Use-Case-04

Tabelle 11: Funktionale Anforderungen

### 6.2 Nicht-funktionale Anforderungen

Nr.	Art	Beschreibung
01	Sicherheit/Datenschutz	Benutzer können Daten nur sehen, wenn sie eingeloggt sind.
02	Sicherheit/Datenschutz	Benutzer können nur ihre eigenen Daten sehen.
03	Sicherheit/Datenschutz	(Out of scope) Benutzer sehen nur Sachen, zu welchen sie auch die Berechtigungen dafür haben.
04	Sicherheit/Datenschutz	Daten werden sicher und konsistent auf der Datenbank gespeichert. Bei einem Ausfall sind alle Daten mit dem aktuellsten Stand noch verfügbar.
05	Sicherheit/Datenschutz	(Out of scope) Die Applikation hält sich an die DSGVO.

06	Benutzerfreundlichkeit	Nebst IT-affinen Personen sollen auch unerfahrene Benutzer die Applikation einfach bedienen können.
07	Benutzerfreundlichkeit	Die App ist für das Handy sowie Laptop, Tablet, etc. implementiert (responsive Design)
08	Benutzerfreundlichkeit	Eingabefelder sollen validiert werden und dem Benutzer entsprechend gekennzeichnet werden
09	Benutzerfreundlichkeit	(Out of scope) Barrierefreiheit, für Menschen mit Behinderung, muss gegeben sein
10	Entwicklung	(Out of scope) Es gibt eine Möglichkeit, dem Benutzer Push-Benachrichtigungen zu schicken.
11	Entwicklung	Grössere Änderungen am Programmcode müssen von einer zweiten Person angesehen und freigegeben werden.
12	Entwicklung	Sourcecode sowie Dokumentation wird in die Versionsverwaltung eingeecheckt.
13	Verfügbarkeit	Die Applikation soll unterbruchsfrei durchgehend verfügbar sein.
14	Verfügbarkeit	Die Applikation soll mit den gängigsten und neusten Webbrowser (Evergreen Browsers) ausführbar sein.
15	Performance	Die Applikation soll in Seiten und Daten performant und ohne Wartezeit laden. Das bedeutet, dass keine Wartezeit von mehr als 3 Sekunden auftritt.
16	Rechtliches	(Out of scope) Das System hält sich an die Gesetztestexte <ul style="list-style-type: none"> <li>• Data Protection Act that governs the confidentiality of personal information.</li> <li>• Mental Health Act that governs the compulsory detention of patients deemed to be a danger to themselves or others.</li> <li>• Vorschriften swissmedic</li> </ul>

Tabelle 12: Nicht-funktionale Anforderungen

## 7. System model

Das Datenflow-Diagramm zeigt aus der Sicht eines Patienten, wie die Daten im System verarbeitet werden.

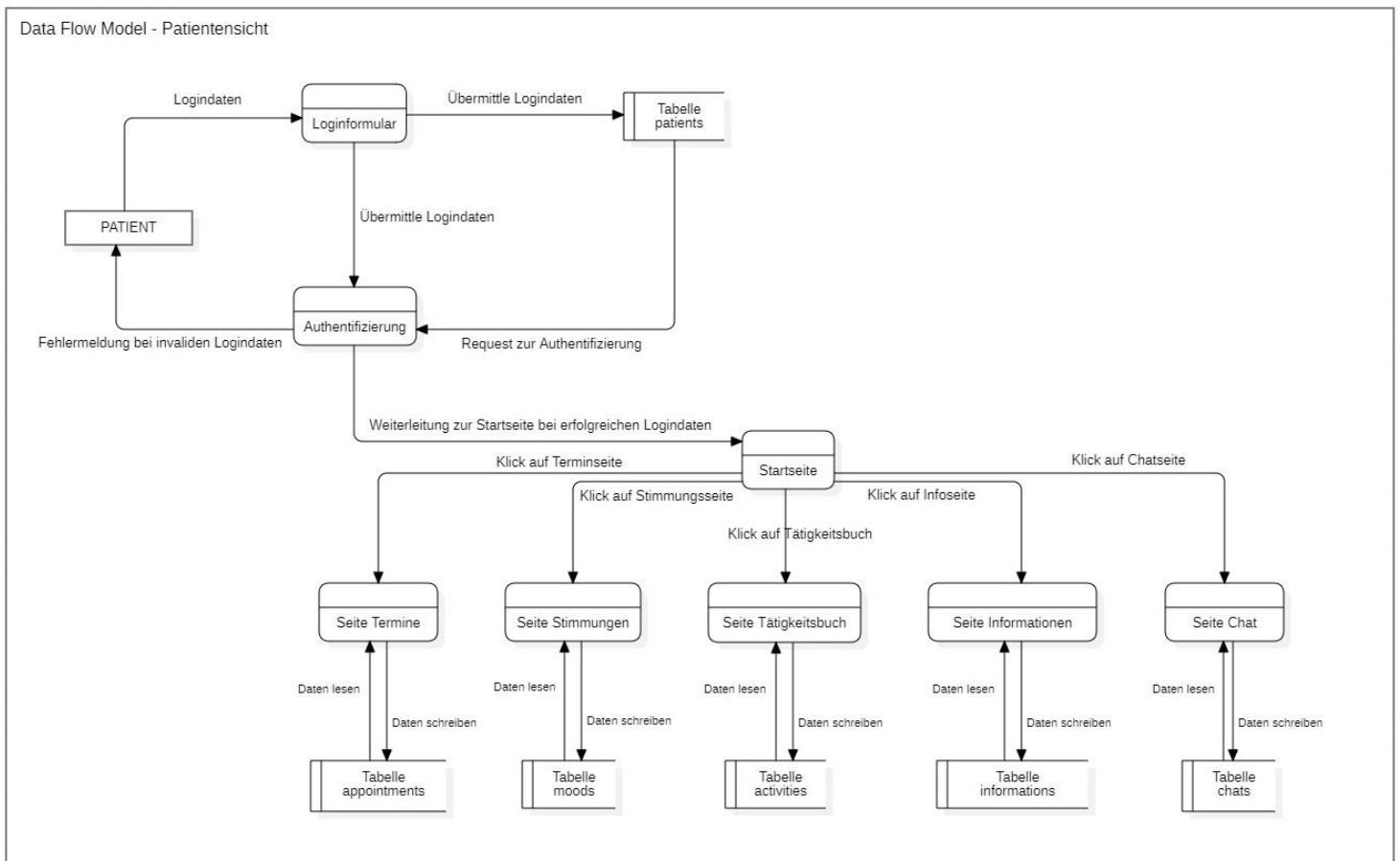


Abbildung 5: Daten-Flow-Diagramm

Die erste Seite ist ein Loginformular, bei welchem der Patient seinen Namen und ein Passwort eingeben muss. Danach werden diese Daten an die Datenbank sowie an den Authentifizierungsservice gesendet. Die Tabelle "patients" sendet das Resultat (Daten gefunden/nicht gefunden) an den Authentifizierungsservice. Der prüft, ob die eingegebenen Daten mit den Daten der Datenbank übereinstimmen. Falls ja, wird die Startseite der Webapplikation aufgerufen.

Auf der Startseite kann der Patient dann die verschiedenen Unterfunktionen auswählen. Bei einem Klick auf den Menüpunkt wird die entsprechende Seite geladen. Jede Seite schreibt und liest dann Daten von der Datenbank und ihrer entsprechenden Tabelle/n.

## **8. System evolution**

### **8.1 Soll-Zustand**

Zur Beendigung des Projektes soll eine Webapplikation zur Unterstützung von ambulanten Patienten, welche an einer Depression leiden, fertiggestellt sein.

Der behandelnde Therapeut erfasst jeweils den Patienten im System. Sobald der Patient erfasst wurde, kann sich dieser mit einem Browser von verschiedenen Geräten aus in das System einloggen.

Dem Patienten steht eine Chat-Funktion zur Verfügung um sich mit medizinischem Fachpersonal auszutauschen. In Notfällen kann der behandelte Therapeut direkt kontaktiert werden oder auch eine allgemeine Notfallnummer (z.B. 143 - Dargebotene Hand).

Um dem Patienten die Terminvereinbarung zu vereinfachen, kann eine Terminanfrage direkt über das System gestellt werden.

Der Behandelnde Therapeut kann für einen Patienten ein Medikament mit der benötigten Dosierung, Zeiten der Einnahme und eventuellen weiteren Informationen erfassen. Der Patient erhält Erinnerungen für die Einnahme der Medikamente.

Der Patient kann ein Tagebuch führen, in welchem er seine Stimmung und Tätigkeiten erfasst. Dazu sollen unterschiedliche Möglichkeiten zur Verfügung stehen um dies einfach zu erledigen.

Weiterhin sollen dem Patienten Informationsseiten zur Verfügung gestellt werden, auf welchen er sich über seine Krankheit und angrenzende Themen informieren kann.

### **8.2 Mögliche Erweiterungen**

In einem weiteren Schritt wäre es denkbar eine offline Version der Applikation anzubieten. Dies würde eine weitere Erleichterung für den Patienten darstellen, wenn er aus bestimmten Gründen momentan keinen Internetzugang hat. Dies bedeutet aber, dass Teile der Datenbank heruntergeladen werden müssten und später wieder synchronisiert werden. Dies ist ein zusätzlicher Aufwand, welcher momentan nicht angedacht ist.

Mit einer wachsenden Benutzerbasis könnte ein verteiltes System oder load balancing notwendig werden. Deshalb soll bereits beim Design darauf geachtet werden, dass dies ermöglicht wird.

Andere Änderungen oder Erweiterungen sind noch nicht angedacht. Trotzdem soll nach dem open-close Prinzip gearbeitet werden, dies ermöglicht es uns auf Kundenwünsche einzugehen und Anpassungen oder Erweiterungen unkompliziert umzusetzen.

## 9. Testing

### 9.1 Unit Tests

Alle Business Klassen sollen mit Unit Tests abgedeckt werden. Dabei soll darauf geachtet werden, dass clever getestet wird. Es soll nicht jede Zeile Code abgedeckt werden, da dies in der gegebenen Zeit nicht möglich sein wird.

Reine Datenklassen, wie sie z.B. im Modell vorkommen werden, werden explizit nicht getestet. Das Potential für Schaden bei solchen Klassen ist gering, da sie wenig bis keine Businesslogik enthalten. Somit ist der Aufwand für Unit Tests nicht gerechtfertigt.

### 9.2 Integration Tests

Von allen Entwicklern wird erwartet, dass neue Teile der Software integration tested werden, bevor sie in den Main-Branch gemerged werden. Diese Tests sind nicht formgebunden. Jedoch liegt es in der Verantwortung von jedem, das Team nicht durch unzureichend getestete merges zu blockieren.

### 9.3 Reviews

Reviews sollen immer dann angewendet werden, wenn der Entwickler dies für nötig hält. Reviews können dem gesamten Team oder einer Einzelperson zugewiesen werden.

### 9.4 GUI und System Tests

Tests auf der Präsentationsebene, und somit des Gesamtsystems, werden manuell durchgeführt. Für diese Tests besteht kein Test- oder Abnahmeprotokoll, wir arbeiten agil und nicht nach dem V-Modell und somit ist eine definitive Abnahme für ein Projekt dieser Grösse nicht nötig oder zielgerichtet.

Es existieren diverse Möglichkeiten und Tools, um GUI Tests zu automatisieren. Diese setzen aber alle voraus, dass das GUI nur noch minimal angepasst wird. Da wir in einer sehr frühen Phase des Projektes sind, sind grundlegende Änderungen der Benutzeroberfläche zu erwarten. Vorerst wird auf eine Automatisierung der GUI Tests verzichtet, jedoch soll diese für einen späteren Zeitpunkt ermöglicht werden. Deshalb werden allen GUI-Elementen bereits jetzt eindeutige IDs vergeben, was eine spätere Automatisierung deutlich vereinfachen wird.

## 10. Appendices

Folgende Tabelle zeigt, welche Technologien wir versuchen einzusetzen. Es gibt dazu keine speziellen Hardware-Anforderungen.

Technologie	Bemerkungen
Java	Version 11
Vaadin	Aktuellste Version
SQL	Datenbank zur persistenten Speicherung der Daten.
Browser	Evergreen Browsers
Webserver	(Out of scope) Hosting bei einem CH-Anbieter mit allen Standardfunktionen. Lokal kann beispielsweise XAMPP verwendet werden.

Tabelle 13: Appendices

## 11. Anhang

### 11.1 Tabellenverzeichnis

Tabelle 1: Zielpublikum .....	2
Tabelle 2: Änderungskontrolle .....	2
Tabelle 3: Glossary .....	3
Tabelle 4: Nummerierung Use Cases .....	4
Tabelle 5: Allgemeine Informationen zu Use Case 3 .....	5
Tabelle 6: Ablauf für Use Case 3 .....	5
Tabelle 7: Ausnahmen für Use Case 3 .....	6
Tabelle 8: Allgemeine Informationen zu Use Case 11 .....	6
Tabelle 9: Ablauf für Use Case 11 .....	6
Tabelle 10: Ausnahmen für Use Case 11 .....	7
Tabelle 11: Funktionale Anforderungen .....	9
Tabelle 12: Nicht-funktionale Anforderungen .....	10
Tabelle 13: Appendices .....	12

### 11.2 Abbildungsverzeichnis

Abbildung 1: Use Case-Diagramm .....	4
Abbildung 2: Aktivitätendiagramm für Use Case 3 .....	5
Abbildung 3: Aktivitätendiagramm für Use Case 11 .....	7
Abbildung 4: Systemarchitektur .....	8
Abbildung 5: Daten-Flow-Diagramm .....	10