SEP Projektmappe

**Projektmappe des Projektes**

<Projektname>

**Dokumentation des Projektes**

Gruppe <Gruppenbezeichner>:

<Teilnehmer 1>

<Teilnehmer 2>

…

<Teilnehmer n>

**Hinweis**

An vielen Stellen findet Ihr im Dokument folgendes Kästchen:

*Dies ist eine Hilfestellung.*

Diese Kästen dienen dazu, Euch kurze Informationen über Ziele und Inhalte der jeweiligen Abschnitte zu geben. Auch die Beispiele und Templates dienen dazu, euch bei der Dokumentation eures Projektes zu unterstützen. **Sowohl die Kästchen als auch die Beispiele und Templates sind spätestens zur finalen Abgabe der Projektmappe vollständig zu entfernen.** Betrachtet dieses Dokument bitte nicht als Aufgabe, die man von oben nach unten abarbeiten soll; es soll vielmehr als durchgängige Dokumentation eurer Projektarbeit dienen und fortlaufend erweitert bzw. angepasst werden, sodass am Ende des SEPs der Entwicklungsprozess Eurer Software vollständig dokumentiert ist.

Das SEP-Team wünscht Euch  
**Viel Erfolg**

Inhalt

[Projektbeschreibung 4](#_Toc115257124)

[Zyklus I 5](#_Toc115257125)

[Spezifikationsplanung 5](#_Toc115257126)

[User-Stories 6](#_Toc115257127)

[Papierprototypen 7](#_Toc115257128)

[Strukturdiagramme (Komponenten- und Klassendiagramme) 7](#_Toc115257129)

[Verhaltensdiagramme (Kommunikationsdiagramme) 7](#_Toc115257130)

[Funktionalitätsplanung 8](#_Toc115257131)

[Systemtests 9](#_Toc115257132)

[Zyklus II 11](#_Toc115257133)

[Spezifikationsplanung 11](#_Toc115257134)

[User-Stories 11](#_Toc115257135)

[Papierprototypen 12](#_Toc115257136)

[Strukturdiagramme (Komponenten- und Klassendiagramme) 12](#_Toc115257137)

[Verhaltensdiagramme (Kommunikationsdiagramme) 12](#_Toc115257138)

[Funktionalitätsplanung 13](#_Toc115257139)

[Modultests 14](#_Toc115257140)

[Systemtests 14](#_Toc115257141)

[Zyklus III 15](#_Toc115257142)

[Spezifikationsplanung 15](#_Toc115257143)

[User-Stories 15](#_Toc115257144)

[Papierprototypen 16](#_Toc115257145)

[Strukturdiagramme (Komponenten- und Klassendiagramme) 16](#_Toc115257146)

[Verhaltensdiagramme (Kommunikationsdiagramme) 16](#_Toc115257147)

[Funktionalitätsplanung 17](#_Toc115257148)

[Modultests 18](#_Toc115257149)

[Systemtests 18](#_Toc115257150)

[Nutzerhandbuch 19](#_Toc115257151)

[Technische Anforderungen 19](#_Toc115257152)

[Installationsanleitung 19](#_Toc115257153)

[Bedienungsanleitung 19](#_Toc115257154)

# Projektbeschreibung

**Zyklus 1**

***(Allgemeines):***

Im ersten Zyklus soll zunächst die Grundstruktur des Systems implementiert werden. Dem System soll eine Service-Architektur (bestehend aus mindestens einem Frontend-Service und einem BackendService) zu Grunde liegen. Sämtliche Daten des Systems sollen persistent gespeichert werden.

***(Registrierung eines Nutzers):***

Nutzer sollen die Möglichkeit haben sich zu registrieren und somit ein Profil zu erstellen. Ein Nutzerprofil soll aus dem Vor- und Nachnamen des Nutzers, einer gültigen E-Mailadresse, dem Geburtsdatum des Nutzers, einem Passwort und einem optionalen Profilbild bestehen. Ein Nutzer soll die Möglichkeit haben sich im System mittels seines Profils einzuloggen. Der Login-Vorgang soll durch eine Zwei-Faktor-Authentifizierung realisiert werden, bei der, neben dem Passwort, ein zufällig generierter Sicherheitscode eingegeben werden muss, der per E-Mail an den Nutzer versendet wird. Für die Abnahmen: Es soll einen „super“ Sicherheitscode existieren, der ebenfalls akzeptiert wird.

***(Registrierung der Systemadministratoren):***

Nutzer sollen die Möglichkeit haben, sich als Systemadministrator zu registrieren und somit ihr Profil zu erstellen. Diese Systemadministrator-Profile sollen aus dem Vor- und Nachnamen, einer gültigen EMailadresse und einem Passwort bestehen. Ein Nutzer soll die Möglichkeit haben sich im System mittels seines Systemadministrator-Profils einzuloggen. Der Login-Vorgang soll durch eine ZweiFaktor-Authentifizierung realisiert werden, bei der, neben dem Passwort, ein zufällig generierter Sicherheitscode eingegeben werden muss, der per E-Mail an den Nutzer versendet wird. Für die Abnahmen: Es soll ein „super“ Sicherheitscode existieren, der ebenfalls akzeptiert wird.

***(Ligen erstellen):***

Systemadministratoren sollen verschiedene Ligen anlegen können. Eine Liga besteht aus einem Namen, einem Spielplan (bestehend aus mehreren Spieltagen) und einem optionalen Liga-Bild. Ein Spieltag besteht aus mehreren einzelnen Spielen. In jedem Spiel treten zwei unterschiedliche Mannschaften (eine Heimmannschaft und eine Auswärtsmannschaft) gegeneinander an. Ein Systemadministrator soll die Möglichkeit haben den Spielplan mit Hilfe einer CSV-Datei1 einzulesen. Alle diese Liga-Daten (Liga, Spieltage, Spiele) sollen im Nachhinein für jeden Nutzer einsehbar sein. Ein Systemadministrator soll Liga-Daten manuell ändern können. 1 Als

***(Systemdatum ändern):***

Der Systemadministrator soll die Möglichkeit haben das Datum, nach dem sich das System richtet zu ändern. Das Systemdatum definiert welche Spiele aus dem Spielplan noch anstehen und welche schon beendet wurden.

Grundlage für die Spielpläne und die Ergebnisse sollen historische Daten genutzt werden: <https://github.com/footballcsv/deutschland>

# Zyklus I

## Spezifikationsplanung

Jedes Artefakt, das im Rahmen des SEP erstellt wird, muss mit dem Namen genau einer Verantwortlichen/eines Verantwortlichen versehen werden. Das bedeutet, dass jede User Story, jeder Papierprototyp, jedes Komponentendiagramm, jedes Klassendiagramm, jedes Kommunikationsdiagramm und sämtliche Tests mit dem Namen der/des Verantwortlichen versehen und hier in der untenstehenden Tabelle entsprechend eingetragen werden muss. Natürlich kann die Gruppe gemeinsam an einem Artefakt arbeiten, als Verantwortlicher sollte aber genau eine Person eingetragen werden.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Artefakt** | **Art des Artefakts** | **Verantwortlicher** | **Status** |
| **1.** | Registrierung der Nutzer |  |  |  |
| 1.1 | User Story 1 | User Story | Felix | In Bearbeitung |
| 1.2 | Login-Fenster | Papierprototyp | Felix | In Bearbeitung |
| 1.2 | Registrierungs-Fenster | Papierprototyp | Felix | In Bearbeitung |
| **2.** | Backend Architektur |  |  |  |
| 2.1 | Komponentendiagramm Backend | Komponentendiagramm | Felix | In Bearbeitung |
| 2.2 | Klassendiagramm Backend | Klassendiagramm | Felix | In Bearbeitung |
| 2.3 | Kommunikationsdiagramm Backend | Kommunikationsdiagramm | Felix | In Bearbeitung |
| 3. | Frontend-Architektur |  |  |  |
| 3.1 | Komponentendiagramm Frontend | Komponentendiagramm | Alex | In Bearbeitung |
| 3.2 | Klassendiagramm Frontend | Klassendiagramm | Alex | In Bearbeitung |
| 3.3 | Kommunikationsdiagramm  Frontend | Kommunikationsdiagramm | Alex | In Bearbeitung |
| 4 | Allgemeine Verhaltensstruktur |  |  |  |
| 4.1 | Verhaltensdiagramme | Komponentendiagramm | Alex | In Bearbeitung |
| 4.2 | Klassendiagramme | Klassendiagramm | Alex | In Bearbeitung |
| 4.3 | Komponentendiagramme | Kommunikationsdiagramm | Alex | In Bearbeitung |
| 5 | Registrierung Admins |  |  |  |
| 5.1 | User stories | User Story | Felix | In Bearbeitung |
| 5.2 | Login-Fenster | Papierprototyp | Felix | In Bearbeitung |
| 5.3 | Registrierungs-Fenster | Papierprototyp | Felix | In Bearbeitung |
| 6 | Ligen erstellen |  |  |  |
| 6.1 | User Stories 1 | User Story | Tom | In Bearbeitung |
| 6.2 | Login-Fenster | Papierprototyp | Tom | In Bearbeitung |
| 6.3 | Registrierungs-Fenster | Papierprototyp | Tom | In Bearbeitung |
| 7 | Systemdatum ändern |  |  |  |
| 7.1 | User Stories | User Stories | Tom | In Bearbeitung |
| 7.2 | Login-Fenster | Papierprototyp | Tom | In Bearbeitung |
| 7.3 | Registrierungs-Fenster | Papierprototyp | Tom | In Bearbeitung |

## User-Stories

**Template:**

|  |  |
| --- | --- |
| **User Story-ID** | <Eindeutiger Identifizierer> |
| **User Story-Beschreibung** | <Text der User Story mittels Satzschablone:  Als <Rolle> möchte ich <Ziel> [, um/sodass <Nutzen>]  (s. Foliensatz „Anforderungen“)> |
| **Geschätzter Realisierungsaufwand** | <Einschätzung der Zeit, die benötigt wird, um die Userstory zu implementieren> |
| **Priorität** | <Wichtigkeit der User Story hinsichtlich der Aufgabenstellung z.B. hoch, mittel niedrig> |
| **Autor** | <Hier bitte nur einen Zuständigen eintragen z.B. Max Mustermann> |
| **Abhängigkeiten zu anderen User Stories** | <Auflistung verwandter User Stories> |

**Schlechtes Beispiel:**

|  |  |
| --- | --- |
| **User Story-ID** |  |
| **User Story-Beschreibung** | Ich möchte ich rechtzeitig informiert werden, wenn ein Patient einen Termin nicht wahrnimmt. |
| **Geschätzter Realisierungsaufwand** | 1337 |
| **Priorität** | - |
| **Autor** | Emmett Brown, Rick Sanchez, Amelia Pond |
| **Abhängigkeiten zu anderen User Stories** |  |

**Gutes Beispiel:**

|  |  |
| --- | --- |
| **User Story-ID** | 1.6 |
| **User Story-Beschreibung** | Als Arzt möchte ich mindesten fünf Minuten vor dem Termin informiert werden, wenn ein Patient einen Termin nicht wahrnimmt, sodass ich andere Patienten vorziehen kann. |
| **Geschätzter Realisierungsaufwand** | 2 Tage |
| **Priorität** | Hoch |
| **Autor** | Emmett Brown |
| **Abhängigkeiten zu anderen User Stories** | 1.3, 1.5 |

Papierprototypen

Das Erstellen eines Papierprototypen dient als Methode des Brainstormings, Designs, Herstellens, Testens und des Kommunizierens von Benutzer Interfaces.

## Strukturdiagramme (Komponenten- und Klassendiagramme)

Im SEP soll die statische Struktur des Systems mittels Komponenten- und Klassendiagramme modelliert werden. Ein Komponenten- und Klassendiagramme dienen der grafischen Darstellung von Komponenten/Klassen, Schnittstellen und deren Beziehungen. Die Diagrammtypen helfen dabei, Quellcode und Implementierungsarbeiten zu strukturieren, bevor diese starten und ermöglicht eine Aufteilung der Programmieraufgaben.

## Verhaltensdiagramme (Kommunikationsdiagramme)

Im SEP soll das dynamische Verhalten des Systems mittels Kommunikationsidagramme modelliert werden. Ein Kommunikationsdiagramme ermöglicht die grafische Darstellung des Nachrichtenaustausches zwischen Systemobjekten. Systemobjekte können Komponenten im Komponentendiagramm und Klassen im Klassendiagramm sein. Kommunikationsdiagramme zielen darauf ab, die Zusammenarbeit der Systemobjekte darzustellen

## Funktionalitätsplanung

Anhand der Funktionalitätsplanung werdet Ihr geprüft. Diese sollte dementsprechend **immer** aktuell gehalten werden und pro Funktionalität **nur einen** Verantwortlichen enthalten. Als Quellcodereferenz solltet Ihr immer euer Package, eure Klasse und die dazugehörige Methode angeben. ggf. könnt Ihr auch Zeilenangaben machen.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Funktionalität** | **Verantwortlicher** | **Abhängige**  **Funktionalitäten** | **Verknüpfte**  **User-Stories** | **Quellcode-**  **referenz** | **Status** |
| **1.** | **Registrierung von Admins** |  |  |  |  |  |
| 1.1 | Registrierungsview im Frontend | Max Muster | 1.2 |  |  | Fertig |
| 1.2 | Datenbank-modell des Admins | Max Muster | 1.3, 1.4 |  |  | WIP |
| 1.2.1 | …. |  |  |  |  |  |
| **2.** | **Anliegen von Liegen** |  |  |  |  |  |
| 2.1 | Ligaübersicht  im Frontend | Anja Muster | 2.4, 2,7 |  |  | Fertig |
| … |  |  |  |  |  |  |

## Systemtests

Systemtests sind Tests des Gesamtsystems gegen die Anforderungen nach erfolgreicher Integration. Eingaben und Sollverhalten werden dabei aus der Anforderungs-spezifikation abgeleitet.

Die Systemtests werden von Eurer Parallelgruppe spezifiziert und durchgeführt, daher ist dieser Bereich von den Mitgliedern der Parallelgruppe auszufüllen.

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | 03.03.2019 | | |
| **Tester** | Martina Musterfrau | | |
| **SW-Version** | V 0.1.2 | | |
| **Vorbedin-gung(en)** | Nutzer „Max Mustermann“ ist am System mit Passwort „geheim“ registriert | | |
| **Schritt** | **Aktion (User)** | **Erwartete Reaktion (System)** | **√ / X** |
| 1 | Der Benutzer gibt den Benutzername „Max Mustermann“ auf der Tastatur ein. | Das System zeigt „Max Mustermann“ auf dem Display an. | **√** |
| 2 | Der Benutzer gibt das Passwort „geheim“ auf der Tastatur ein. | Das System zeigt das Passwort durch „\*“-Symbole zensiert an. | **√** |
| 3 | Der Benutzer klickt auf „Anmelden“. | Das System zeigt die Meldung „Anmeldung erfolgreich“ auf dem Display an. | **X** |
| **Nachbe-dingung(en)** | Nutzer ist am System angemeldet, Anmeldezeitpunkt ist im System gespeichert. | | **X** |
| **Testurteil** | **Test nicht bestanden.** | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | 03.03.2019 | | |
| **Tester** | Martina Musterfrau | | |
| **SW-Version** | V 0.1.2 | | |
| **Vorbedin-gung(en)** | Nutzer „Max Mustermann“ ist am System mit Passwort „geheim“ registriert | | |
| **Schritt** | **Aktion (User)** | **Erwartete Reaktion (System)** | **√ / X** |
| 1 | Der Benutzer gibt den Benutzername „Max Mustermann“ auf der Tastatur ein. | Das System zeigt „Max Mustermann“ auf dem Display an. | **√** |
| 2 | Der Benutzer gibt das Passwort „geheim“ auf der Tastatur ein. | Das System zeigt das Passwort durch „\*“-Symbole zensiert an. | **√** |
| 3 | Der Benutzer klickt auf „Anmelden“. | Das System zeigt die Meldung „Anmeldung erfolgreich“ auf dem Display an. | **√** |
| **Nachbe-dingung(en)** | Nutzer ist am System angemeldet, Anmeldezeitpunkt ist im System gespeichert. | | **√** |
| **Testurteil** | **Test bestanden.** | | |

# Zyklus II

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Artefakt** | **Art des Artefakts** | **Verantwortlicher** | **Status** |
|  |  |  |  |  |
| … |  |  |  |  |
|  |  |  |  |  |
| … |  |  |  |  |
| … |  |  |  |  |

## Spezifikationsplanung

## User-Stories

Template:

|  |  |
| --- | --- |
| **User Story-ID** |  |
| **User Story-Beschreibung** |  |
| **Geschätzter Realisierungsaufwand** |  |
| **Priorität** |  |
| **Autor** |  |
| **Abhängigkeiten zu anderen User Stories** |  |

Papierprototypen

## Strukturdiagramme (Komponenten- und Klassendiagramme)

## Verhaltensdiagramme (Kommunikationsdiagramme)

## Funktionalitätsplanung

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Funktionalität** | **Verantwortlicher** | **Abhängige**  **Funktionalitäten** | **Verknüpfte**  **User-Stories** | **Quellcode-**  **referenz** | **Status** |
| **1.** |  |  |  |  |  |  |
| 1.1 |  |  |  |  |  |  |
| 1.2 |  |  |  |  |  |  |
| 1.2.1 | …. |  |  |  |  |  |
| **2.** |  |  |  |  |  |  |
| 2.1 |  |  |  |  |  |  |
| … |  |  |  |  |  |  |

## Modultests

Modultests sind Komponententests. Diese werden in der Softwareentwicklung angewendet, um die funktionalen Einzelteile (Units) von Computerprogrammen zu testen.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Getestete Funktionalität** | **Quellcode Referenz** | **Status** |
|  |  |  |  |

## Systemtests

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** |  | | |
| **Tester** |  | | |
| **SW-Version** |  | | |
| **Vorbedin-gung(en)** |  | | |
| **Schritt** | **Aktion (User)** | **Erwartete Reaktion (System)** | **√ / X** |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |
| **Nachbe-dingung(en)** |  | |  |
| **Testurteil** |  | | |

# Zyklus III

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Artefakt** | **Art des Artefakts** | **Verantwortlicher** | **Status** |
|  |  |  |  |  |
| … |  |  |  |  |
|  |  |  |  |  |
| … |  |  |  |  |
| … |  |  |  |  |

## Spezifikationsplanung

## User-Stories

Template:

|  |  |
| --- | --- |
| **User Story-ID** |  |
| **User Story-Beschreibung** |  |
| **Geschätzter Realisierungsaufwand** |  |
| **Priorität** |  |
| **Autor** |  |
| **Abhängigkeiten zu anderen User Stories** |  |
| **Zugehörige Szenarien** |  |

Papierprototypen

## Strukturdiagramme (Komponenten- und Klassendiagramme)

## Verhaltensdiagramme (Kommunikationsdiagramme)

## Funktionalitätsplanung

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Funktionalität** | **Verantwortlicher** | **Abhängige**  **Funktionalitäten** | **Verknüpfte**  **User-Stories** | **Quellcode-**  **referenz** | **Status** |
| **1.** |  |  |  |  |  |  |
| 1.1 |  |  |  |  |  |  |
| 1.2 |  |  |  |  |  |  |
| 1.2.1 | …. |  |  |  |  |  |
| **2.** |  |  |  |  |  |  |
| 2.1 |  |  |  |  |  |  |
| … |  |  |  |  |  |  |

## Modultests

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Getestete Funktionalität** | **Quellcode Referenz** | **Status** |
|  |  |  |  |

## Systemtests

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** |  | | |
| **Tester** |  | | |
| **SW-Version** |  | | |
| **Vorbedin-gung(en)** |  | | |
| **Schritt** | **Aktion (User)** | **Erwartete Reaktion (System)** | **√ / X** |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |
| **Nachbe-dingung(en)** |  | |  |
| **Testurteil** |  | | |

# Nutzerhandbuch

## Technische Anforderungen

Technische Mindestanforderungen, welche das System benötigt, um wie gewünscht bedienbar zu sein.

## Installationsanleitung

Genaue Erläuterung, wie das entwickelte System vollkommen funktionsfähig auf einem Rechner in Betrieb genommen werden kann.

## Bedienungsanleitung

Genaue Erläuterung, wie das entwickelte System zu bedienen ist.