SEP Projektmappe

***Projektmappe des Projektes***

SchachEP

**Dokumentation des Projektes**

Gruppe I:

Ayoub Ouamar

Seher Tezer

Jonas Giesen

Kaan Hisiroglu

Yusuf Cobandir

Mario Mai

**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](#_heading=h.gjdgxs)

[Zyklus I 7](#_heading=h.1fob9te)

[Spezifikationsplanung 7](#_heading=h.3znysh7)

[User-Stories 8](#_heading=h.2et92p0)

[Papierprototypen 9](#_heading=h.1t3h5sf)

[Strukturdiagramme (Komponenten- und Klassendiagramme) 9](#_heading=h.4d34og8)

[Verhaltensdiagramme (Kommunikationsdiagramme) 9](#_heading=h.2s8eyo1)

[Funktionalitätsplanung 10](#_heading=h.17dp8vu)

[Systemtests 11](#_heading=h.3rdcrjn)

[Zyklus II 13](#_heading=h.26in1rg)

[Spezifikationsplanung 13](#_heading=h.lnxbz9)

[User-Stories 13](#_heading=h.35nkun2)

[Papierprototypen 14](#_heading=h.1ksv4uv)

[Strukturdiagramme (Komponenten- und Klassendiagramme) 14](#_heading=h.44sinio)

[Verhaltensdiagramme (Kommunikationsdiagramme) 14](#_heading=h.2jxsxqh)

[Funktionalitätsplanung 15](#_heading=h.z337ya)

[Modultests 16](#_heading=h.3j2qqm3)

[Systemtests 16](#_heading=h.1y810tw)

[Zyklus III 17](#_heading=h.2xcytpi)

[Spezifikationsplanung 17](#_heading=h.1ci93xb)

[User-Stories 17](#_heading=h.3whwml4)

[Papierprototypen 18](#_heading=h.2bn6wsx)

[Strukturdiagramme (Komponenten- und Klassendiagramme) 18](#_heading=h.qsh70q)

[Verhaltensdiagramme (Kommunikationsdiagramme) 18](#_heading=h.3as4poj)

[Funktionalitätsplanung 19](#_heading=h.1pxezwc)

[Modultests 20](#_heading=h.49x2ik5)

[Systemtests 20](#_heading=h.2p2csry)

[Nutzerhandbuch 21](#_heading=h.147n2zr)

[Technische Anforderungen 21](#_heading=h.3o7alnk)

[Installationsanleitung 21](#_heading=h.23ckvvd)

[Bedienungsanleitung 21](#_heading=h.ihv636)

# Projektbeschreibung

Einleitung

Das SchachEP soll ein System werden, dass den Anwendern das Spielen von Schachpartien ermöglicht. Dabei ermöglicht das Systeme es Schachpartien mit Freunden oder Computer-Gegnern zu spielen. Außerdem verfügt es über ein Punktesystem und einem Leaderboard. Ferner beinhaltet es SinglePlayer Schachpuzzle sowie die Möglichkeit spannende Schachspiele zu streamen.

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-Mail-Adresse, dem Geburtsdatum des Nutzers, einem Passwort, Punkte (vergleichbar mit ELO) und einem optionalen Profilbild bestehen. Jeder Spieler startet mit 500 Punkten. Ein Nutzer soll in der Lage sein, sich sein Profil anzuschauen. Ein Nutzer soll ebenfalls die Möglichkeit haben, sich im System mittels seiner Zugangsdaten einzuloggen. Der Login-Vorgang soll durch eine Zwei-Faktor-Authentifizierung gesichert 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.

(Freundesliste) Nutzer sollen die Möglichkeit haben, andere Nutzer als Freunde hinzuzufügen. Eine Freundesliste soll für jeden Nutzer angelegt werden, in der er seine Freunde sehen und verwalten kann. Der Nutzer soll in der Lage sein, Freundschaftsanfragen zu senden und anzunehmen oder abzulehnen. Wenn ein Nutzer eine Freundschaftsanfrage erhält, soll er darüber per E-Mail benachrichtigt werden. Die Freundesliste soll öffentlich oder privat einstellbar sein, so dass andere Nutzer sehen können, wer in der Liste enthalten ist, oder ob sie privat ist. Ein Nutzer kann sich das Profil von jedem seiner Freunde anschauen.

(Schachpartie erstellen) Ein Benutzer kann eine Schachpartie mit einem eindeutigen Namen und einen Timer erstellen und einen weiteren Benutzer (entweder ein Benutzer, welcher durch eine Suchfunktion gefunden wurde oder einen Freund) einladen. Der eingeladene Benutzer kann die Einladung annehmen oder ablehnen. Nimmt der eingeladene Benutzer die Einladung an, können beide Spieler einen Platzhalter sehen. Jeder Spieler kann die Schachpartie ausblenden (nicht diese verlassen!) und nachträglich wieder beitreten.

Zyklus 2

(Chat) Nutzer sollen die Möglichkeit haben, mit ihren Freunden über das System zu chatten. Hierfür soll eine Chat-Funktion implementiert werden, der es den Nutzern ermöglicht, private oder Gruppenchats zu erstellen und Nachrichten zu senden. Die Nutzer sollen in der Lage sein, Nachrichten in Echtzeit zu senden und zu empfangen. Außerdem sollen die Nutzer die Möglichkeit haben, Nachrichten zu löschen oder zu bearbeiten, solange der Empfänger sie noch nicht gelesen hat.

(Schachpartie spielen) Eine erstellte Schachpartie soll nun spielbar werden. Dazu sollen ein Sachbrett mit allen Schachfiguren sichtbar sein, sobald eine Schachpartie geöffnet wird. Weiterhin sollen die offiziellen Schachregeln implementiert werden (z.B. abwechselnde Züge zwischen den Spielern, erlaubten Züge der Figuren auf dem Feld, Schlagen von Figuren auf dem Feld, Rochade, etc.)1 . Außerdem soll ein Timer für jeden Spieler implementiert werden, nach dessen Ablauf der entsprechende Spieler das Spiel verloren hat. Während ein Spieler nicht am Zug ist, ist der Timer pausiert. Weiterhin soll ein Spieler zur jeder Zeit die Partie aufgeben können. Beim Gewinn einer Schachpartie erhält der Gewinner 10 Punkte und der Verlierer verliert 10 Punkte. Im Falle eines Remis sollen keine Punkte vergeben werden.

(Schachclub) Ein Benutzer kann einen Schachclub mit einem eindeutigen Namen erstellen. Jeder Nutzer kann die Liste aller erstellten Schachclubs einsehen. Weiterhin ist jeder Benutzer in der Lage einen Schachclub beizutreten. Nutzer innerhalb eines Schachclubs können miteinander über einen Chat in Echtzeit kommunizieren (siehe Chat). Außerdem soll im Profil eines jeden Nutzers ersichtlich sein, in welchen Schachclubs er beigetreten ist.

(Schachpuzzle) Insgesamt sollen 10 vordefinierte Schachpuzzles existieren, welche von einem Benutzer in das System mittels einer CSV-Datei2 importiert werden können. Ein Schachpuzzle zeigt dabei ein laufendes Schachspiel. Das Puzzle besteht darin, dass der Benutzer den besten nächsten Zug ermitteln muss. Sobald der Benutzer 3 vordefinierten Schachpuzzle erfolgreich löst, erhält er eine Anerkennung auf seinem Profil mit der Aufschrift „Schachexperte“.

(Leaderboard) Jeder Benutzer kann ein Leaderboard einsehen, welches die besten Spieler anhand ihrer Punkte in absteigender Folge anzeigt.

Zyklus 3

(Spielhistorie) Jeder Spieler soll in seinem Profil die drei zuletzt gespielten Partien einsehen können. Hierbei sollen Name und Punkte der Teilnehmer sowie das Ergebnis angezeigt werden.

(Schachspiel wiederholen) Ein Benutzer soll in der Lage sein sich den Spielverlauf eines jeden Spieles der Spielhistorie nachträglich anzuschauen. Dabei soll es möglich sein, die Züge manuell umzuschalten (d.h. sich den nächsten bzw. vorherigen Zug anzuschauen).

(Export/Import von Partien) Der Verlauf einzelner Spiele aus der Spielhistorie soll als PGN3 exportiert werden können. Weiterhin soll ein Spieler in der Lage sein, in einer PGN-Datei gespeicherte Züge einer Partie zu importieren. Anschließend soll er die einzelnen Züge dieses Spiels auf dem Schachbrett anzeigen lassen können. Die Funktionalität entspricht der des Features Schachspiel wiederholen.

(Schachpartie gegen Computer spielen) Ein Spieler kann eine Schachpartie mit einem Computer-Gegner starten4 . Der Computer-Gegner soll dabei in mindestens drei Schwierigkeitsstufen einstellbar sein. Das Spielverlauf gleicht dem der Funktionalität Schachpartie spielen.

(Streaming von Live-Spielen) Spieler sollen in der Lage sein, live gespielte Spiele öffentlich zu streamen. Weiterhin sollen Spieler, welche sich ein live-gestreamtes Spiel anschauen wollen, eine Liste sehen, in welcher alle aktuell gestreamten Spiele angezeigt werden. Aus dieser Liste können die Spieler ein gestreamtes Spiel auswählen und dieses anschließend live anschauen.

(Assistentenbot) Spieler sollen in der Lage sein einen Assistentenbot um Rat hinsichtlich des nächsten zu spielenden Zuges einzuholen. Dieser nächste zu spielende Zug soll dabei der beste nächste zu spielende Zug sein. Für die Hilfe des Assistentenbot müssen die Spieler allerdings einen Punkt bezahlen.

# 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 | Registrierung für ein Nutzerprofil | User-Story | Seher Tezer | Fertig |
| 1.2 | Login mit Zugangsdaten | User Story | Seher Tezer | Fertig |
| 1.2 | Login-Vorgang mit Zwei Faktor-Authentifizierung | User Story | Seher Tezer | Fertig |
| 1.3 | Zusendung eines Sicherheitscodes per E-Mail | User Story | Jonas Giesen | Fertig |
| 1.4 | Übersicht Profilseite | User Story | Jonas Giesen | Fertig |
| 1.5 | Höhe der Startpunktzahl | User Story | Jonas Giesen | Fertig |
| 1.6 | Registrierungs-Fenster | Papierprototyp | Seher Tezer | Fertig |
| 1.7 | Login-Fenster | Papierprototyp | Seher Tezer | Fertig |
| 1.8 | Profil des User | Papierprototyp | Jonas Giesen | Fertig |
| 1.9 | Fenster Authentifizierungscode | Papierprototyp | Jonas Giesen | Fertig |
| 1.10 | Optionales Profilbild | User Story | Seher Tezer | Fertig |
| **2.Freundesliste** | | | | |
| 2.1 | Hinzufügen von Kontakten in eine Freundesliste | User Story | Mario Mai | Fertig |
| 2.2 | Verwalten meiner Freundesliste | User Story | Kaan Hisiroglu | Fertig |
| 2.3 | Annehmen von Freundschaftsanfragen | User Story | Kaan Hisiroglu | Fertig |
| 2.4 | Benachrichtigung bei Freundschaftsanfrage per E-Mail | User Story | Mario Mai | Fertig |
| 2.5 | Freundschaftsanfragen versenden | User Story | Mario Mai | Fertig |
| 2.6 | Sichtbarkeit der Freundesliste | User Story | Mario Mai | Fertig |
| 2.7 | Freundesliste | Papierprototyp | Mario Mai | Fertig |
| 2.8 | Freund hinzufügen | Papierprototyp | Kaan Hisiroglu | Fertig |
| 2.9 | Freundschaftsanfragen und Spieleinladungen | Papierprototyp | Kaan Hisiroglu | Fertig |
| 2.10 | Profile von Freunden | Papierprototyp | Kaan Hisiroglu | Fertig |
| 2.11 | Ablehnen von Freundschaftsanfragen | User Story | Kaan Hisiroglu | Fertig |
| **3.Schachpartie erstellen** | | | | |
| 3.1 | Partie-Erstellung mit Namen und Timer | User Story | Ayoub Ouamar | Fertig |
| 3.2 | Suchen und Einladen eines Benutzers | User Story | Ayoub Ouamar | Fertig |
| 3.3 | Freunde einladen | User Story | Ayoub Ouamar | Fertig |
| 3.4 | Einladung annehmen/ablehnen | User Story | Yusuf Cobandir | Fertig |
| 3.5 | Partie ausblenden und erneut beitreten | User Story | Yusuf Cobandir | Fertig |
| 3.6 | Startseite | Papierprototyp | Ayoub Ouamar | Fertig |
| 3.7 | Startseite mit Login | Papierprototyp | Ayoub Ouamar | Fertig |
| 3.8 | Partie gegen Computer erstellen | Papierprototyp | Yusuf Cobandir | Fertig |
| 3.9 | Partie gegen Benutzer erstellen | Papierprototyp | Ayoub Ouamar | Fertig |
| 3.10 | Schachpartie + Partie ausblenden | Papierprototyp | Yusuf Cobandir | Fertig |
| 3.11 | Partie wieder beitreten | Papierprototyp | Yusuf Cobandir | Fertig |
| **4.Diagramme** | | | | |
| 4.1 | Komponentendiagramm | Diagramm | Ayoub | Fertig |
| 4.2 | Klassendiagramm | Diagramm | Jonas | Fertig |
| 4.3 | Kommunikationsdiagramm | Diagramm | Mario | Fertig |

## 

## User-Stories

|  |  |
| --- | --- |
| **User Story-ID** | 1.1 |
| **User Story-Beschreibung** | Als User möchte ich mich registrieren können, um ein Profil für das Schachspiel zu erstellen. |
| **Geschätzter Realisierungsaufwand** | 1 Tag |
| **Priorität** | Hoch |
| **Autor** | Seher Tezer |
| **Abhängigkeiten zu anderen User Stories** |  |

|  |  |
| --- | --- |
| **User Story-ID** | 1.2 |
| **User Story-Beschreibung** | Als User möchte ich mich mit meinen Zugangsdaten anmelden können, damit ich eingeloggt bin. |
| **Geschätzter Realisierungsaufwand** | 1 Tag |
| **Priorität** | Hoch |
| **Autor** | Seher Tezer |
| **Abhängigkeiten zu anderen User Stories** | 1.1 |

|  |  |
| --- | --- |
| **User Story-ID** | 1.3 |
| **User Story-Beschreibung** | Als User möchte ich eine Zwei-Faktor-Authentifizierung (2FA) haben, um die Sicherheit meines Kontos zu erhöhen. |
| **Geschätzter Realisierungsaufwand** | 2 Tage |
| **Priorität** | Hoch |
| **Autor** | Seher Tezer |
| **Abhängigkeiten zu anderen User Stories** | 1.1, 1.2 |

|  |  |
| --- | --- |
| **User Story-ID** | 1.4 |
| **User Story-Beschreibung** | Als User möchte ich den Sicherheitscode per E-Mail zugeschickt bekommen, um die Sicherheit meines Kontos zu erhöhen. |
| **Geschätzter Realisierungsaufwand** | 1 Tage |
| **Priorität** | Hoch |
| **Autor** | Jonas Giesen |
| **Abhängigkeiten zu anderen User Stories** | 1.1, 1.2, 1.3 |

|  |  |
| --- | --- |
| **User Story-ID** | 1.5 |
| **User Story-Beschreibung** | Als User möchte ich mir meine Profilseite anschauen können, um einen Überblick über meine Profildaten zu erhalten. |
| **Geschätzter Realisierungsaufwand** | 0,5 Tage |
| **Priorität** | Hoch |
| **Autor** | Jonas Giesen |
| **Abhängigkeiten zu anderen User Stories** | 1.1, 1.2 |

|  |  |
| --- | --- |
| **User Story-ID** | 1.6 |
| **User Story-Beschreibung** | Als User möchte ich mit einer Spielerbewertung von 500 anfangen, um einen Anreiz zu haben, meine Fähigkeiten zu verbessern. |
| **Geschätzter Realisierungsaufwand** | 0,5 Tage |
| **Priorität** | Hoch |
| **Autor** | Jonas Giesen |
| **Abhängigkeiten zu anderen User Stories** | 1.1, 1.2, 1.3 |

|  |  |
| --- | --- |
| **User Story-ID** | 1.10 |
| **User Story-Beschreibung** | Als User möchte ich die Möglichkeit haben, optional ein Profilbild hochzuladen, damit mich andere identifizieren können. |
| **Geschätzter Realisierungsaufwand** | 1 Tage |
| **Priorität** | Hoch |
| **Autor** | Seher Tezer |
| **Abhängigkeiten zu anderen User Stories** | 1.1, 1.2, 1.5 |

|  |  |
| --- | --- |
| **User Story-ID** | 2.1 |
| **User Story-Beschreibung** | Als User möchte ich andere User in Freundeslisten abspeichern können, um ihre Profile zu sehen und mit ihnen in Kontakt zu bleiben. |
| **Geschätzter Realisierungsaufwand** | 1 Tag |
| **Priorität** | Hoch |
| **Autor** | Mario Mai |
| **Abhängigkeiten zu anderen User Stories** |  |

|  |  |
| --- | --- |
| **User Story-ID** | 2.2 |
| **User Story-Beschreibung** | Als User möchte ich die Möglichkeit haben, meine selbst angelegten Kontakte in der Freundesliste löschen zu können, damit ich diese gepflegt halten kann. |
| **Geschätzter Realisierungsaufwand** | 0,5 Tage |
| **Priorität** | Hoch |
| **Autor** | Kaan Hisiroglu |
| **Abhängigkeiten zu anderen User Stories** | 2.1 |

|  |  |
| --- | --- |
| **User Story-ID** | 2.3 |
| **User Story-Beschreibung** | Als User möchte ich über gesendete Freundschaftsanfragen selbst entscheiden, ob ich diese annehme, damit ich ungewollte User nicht in meiner Freundesliste habe. |
| **Geschätzter Realisierungsaufwand** | 2 Tage |
| **Priorität** | Hoch |
| **Autor** | Kaan Hisiroglu |
| **Abhängigkeiten zu anderen User Stories** | 2.1 |

|  |  |
| --- | --- |
| **User Story-ID** | 2.4 |
| **User Story-Beschreibung** | Als User möchte ich per E-Mail benachrichtigt werden, wenn ich eine Freundschaftsanfrage erhalte, damit ich schnell auf die Anfrage reagieren kann. |
| **Geschätzter Realisierungsaufwand** | 0,5 Tage |
| **Priorität** | Hoch |
| **Autor** | Mario Mai |
| **Abhängigkeiten zu anderen User Stories** | 2.1, 2.5 |

|  |  |
| --- | --- |
| **User Story-ID** | 2.5 |
| **User Story-Beschreibung** | Als User möchte ich anderen User eine Freundschaftsanfrage schicken, damit ich sie in meiner Freundesliste speichern kann, um gegen sie eine Partie Schach zu spielen. |
| **Geschätzter Realisierungsaufwand** | 2 Tage |
| **Priorität** | Hoch |
| **Autor** | Mario Mai |
| **Abhängigkeiten zu anderen User Stories** | 2.1 |

|  |  |
| --- | --- |
| **User Story-ID** | 2.6 |
| **User Story-Beschreibung** | Als User möchte ich die Möglichkeit haben, die Sichtbarkeit meiner Freundesliste einzustellen, entweder öffentlich zugänglich oder nur für mich sichtbar, damit ich die Privatsphäre meiner Freundesliste anpassen kann. |
| **Geschätzter Realisierungsaufwand** | 2 Tage |
| **Priorität** | Hoch |
| **Autor** | Mario Mai |
| **Abhängigkeiten zu anderen User Stories** | 2.1 |

|  |  |
| --- | --- |
| **User Story-ID** | 2.11 |
| **User Story-Beschreibung** | Als User möchte ich über Freundschaftsanfragen selbst entscheiden, ob ich diese ablehne, damit ich ungewollte User nicht in meiner Freundesliste habe. |
| **Geschätzter Realisierungsaufwand** | 2 Tage |
| **Priorität** | Hoch |
| **Autor** | Kaan Hisiroglu |
| **Abhängigkeiten zu anderen User Stories** | 2.1 |

| **User Story-ID** | 3.1 |
| --- | --- |
| **User Story-Beschreibung** | Als Benutzer möchte ich eine Schachpartie mit einem eindeutigen Namen und Timer erstellen können, um individuelle Spiele organisieren zu können. |
| **Geschätzter Realisierungsaufwand** | 5 Tage |
| **Priorität** | Hoch |
| **Autor** | Ayoub Ouamar |
| **Abhängigkeiten zu anderen User Stories** | - |

|  |  |
| --- | --- |
| **User Story-ID** | 3.2 |
| **User Story-Beschreibung** | Als Benutzer möchte ich die Möglichkeit haben, einen anderen Benutzer durch eine Suchfunktion zu finden, um diese Person zu einer Partie Schach einladen zu können. |
| **Geschätzter Realisierungsaufwand** | 6 Tage |
| **Priorität** | Hoch |
| **Autor** | Ayoub Ouamar |
| **Abhängigkeiten zu anderen User Stories** | 3.1 |

|  |  |
| --- | --- |
| **User Story-ID** | 3.3 |
| **User Story-Beschreibung** | Als Benutzer möchte ich einen meiner Freunde direkt zu einer Schachpartie einladen können, um mit bekannten Gegner zu spielen |
| **Geschätzter Realisierungsaufwand** | 3 Tage |
| **Priorität** | Hoch |
| **Autor** | Ayoub Ouamar |
| **Abhängigkeiten zu anderen User Stories** | 3.1 3.2 |

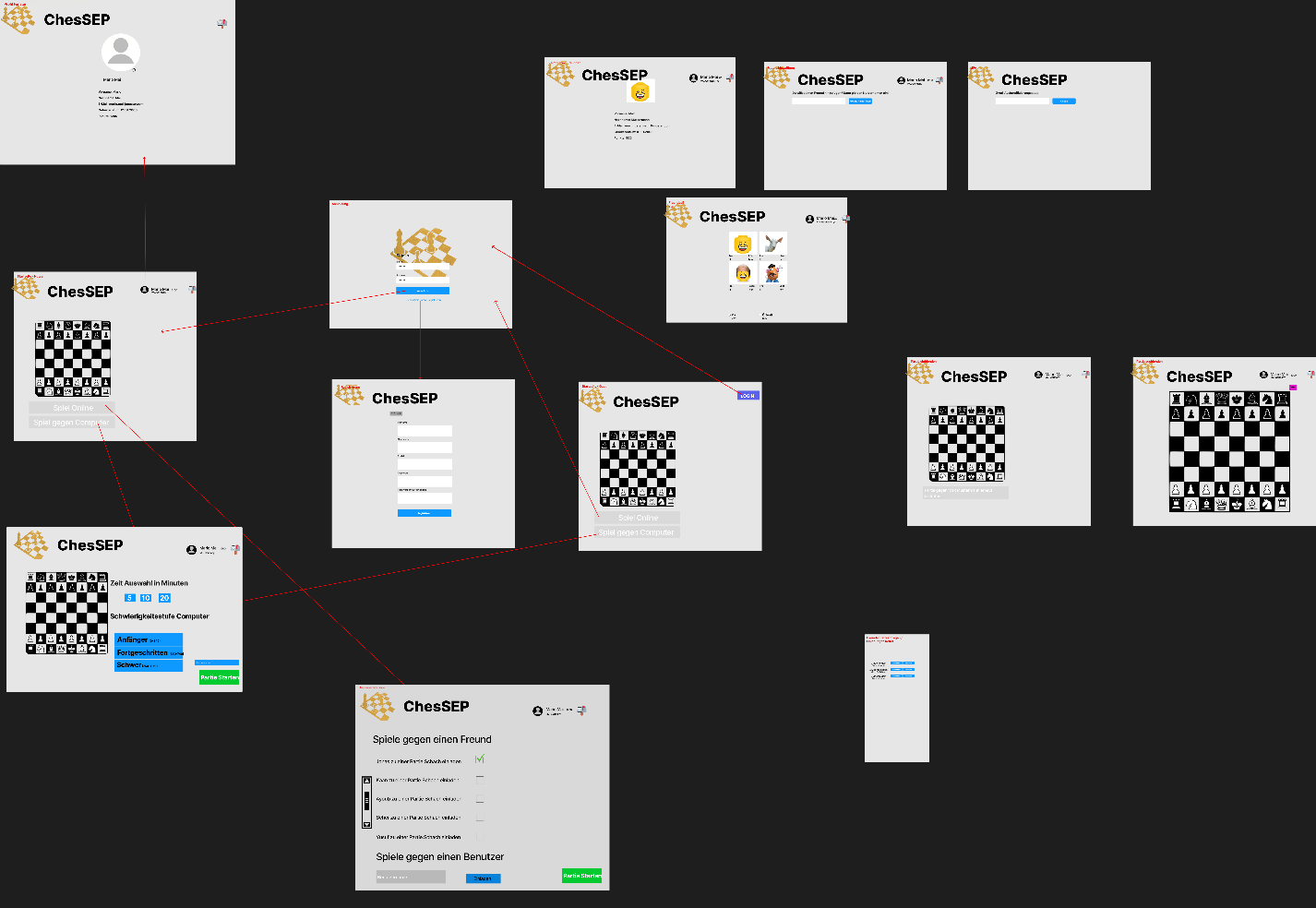
|  |  |
| --- | --- |
| **User Story-ID** | 3.4 |
| **User Story-Beschreibung** | Als eingeladener Benutzer möchte ich die Möglichkeit haben, eine Spieleinladung anzunehmen oder abzulehnen, um zu entscheiden, ob ich am Spiel teilnehmen möchte oder nicht. |
| **Geschätzter Realisierungsaufwand** | 4 Tage |
| **Priorität** | Hoch |
| **Autor** | Yusuf Cobandir |
| **Abhängigkeiten zu anderen User Stories** | 3.3 |

|  |  |
| --- | --- |
| **User Story-ID** | 3.5 |
| **User Story-Beschreibung** | Als Spieler möchte ich die Möglichkeit haben, die aktuelle Partie auszublenden (ohne das Spiel zu verlassen) und später wieder beizutreten, um bei Bedarf andere Aufgaben auf der Plattform zu erledigen |
| **Geschätzter Realisierungsaufwand** | 8 Tage |
| **Priorität** | Hoch |
| **Autor** | Yusuf Cobandir |
| **Abhängigkeiten zu anderen User Stories** | 3.1 3.2 |

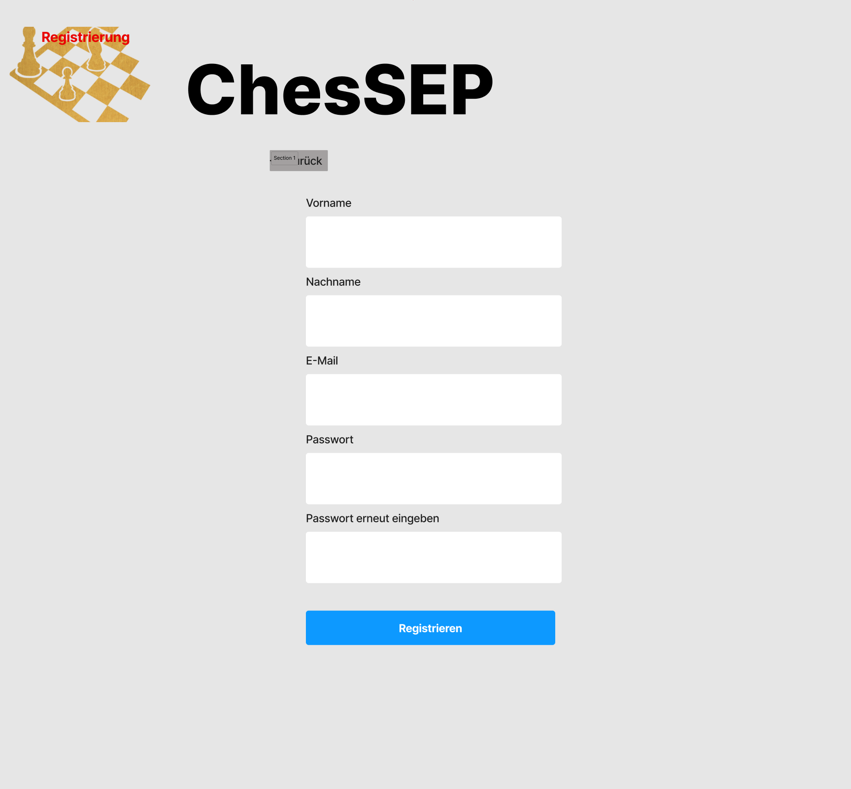
Papierprototypen

Link:

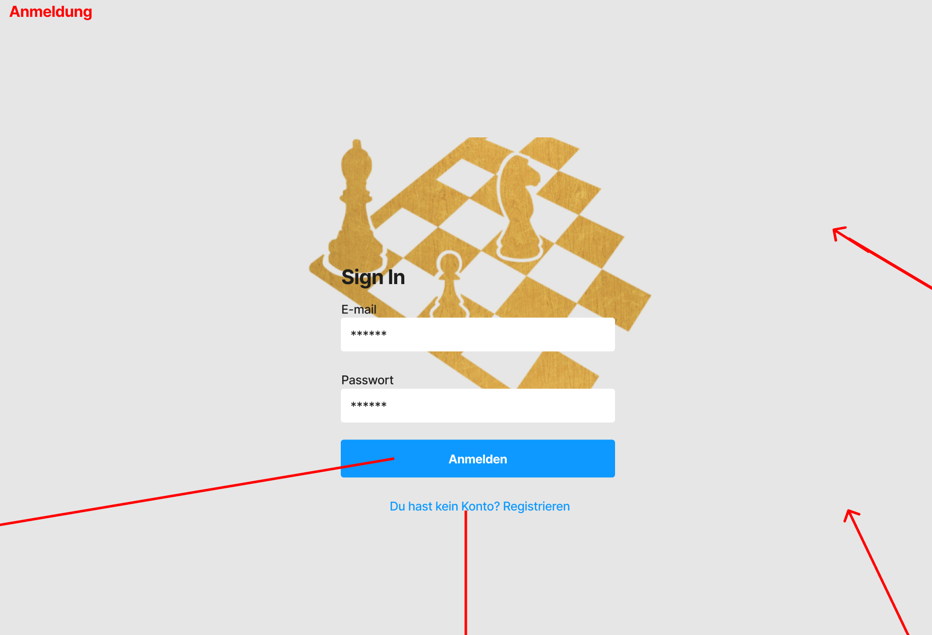
<https://www.figma.com/files/recents-and-sharing/recently-viewed?fuid=1296431075451929896>



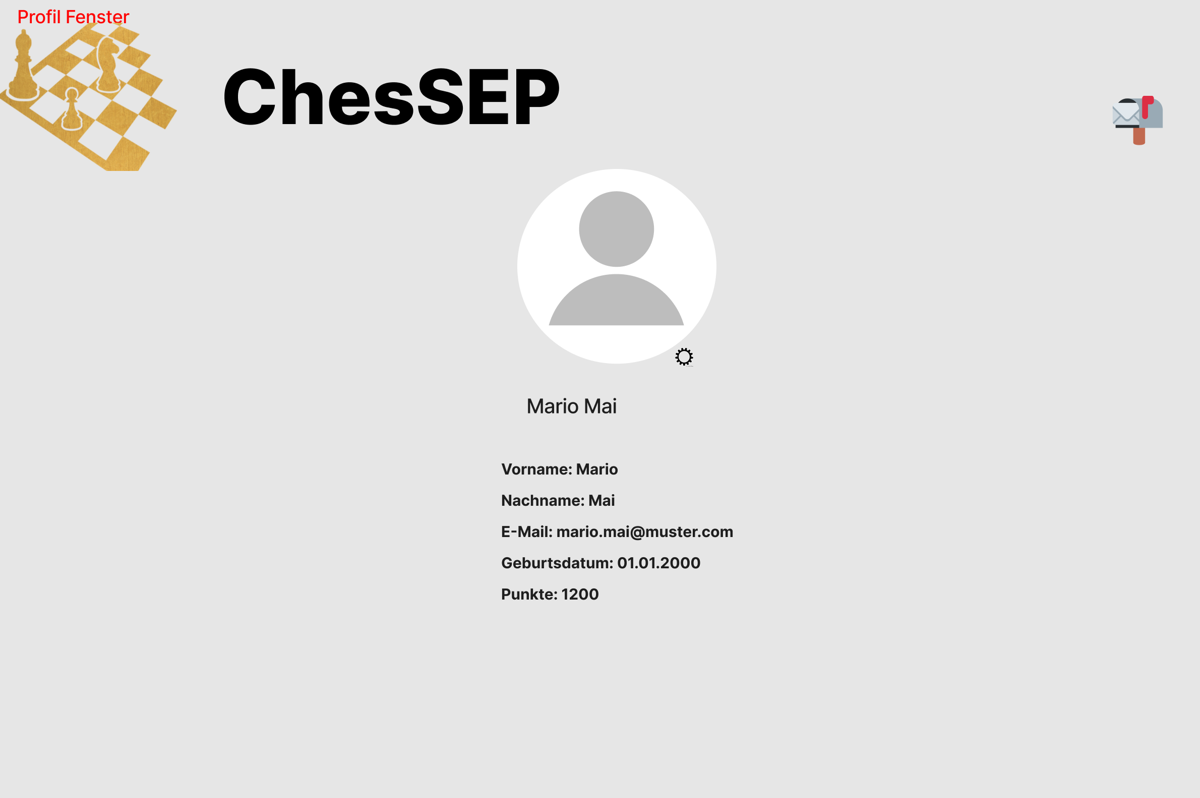
1.6 Registrierungs-Fenster



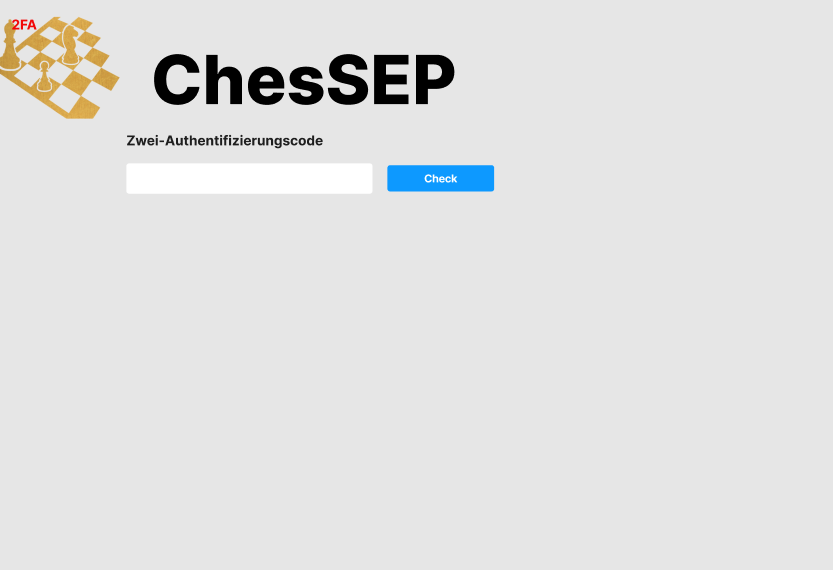
1.7 Login-Fenster

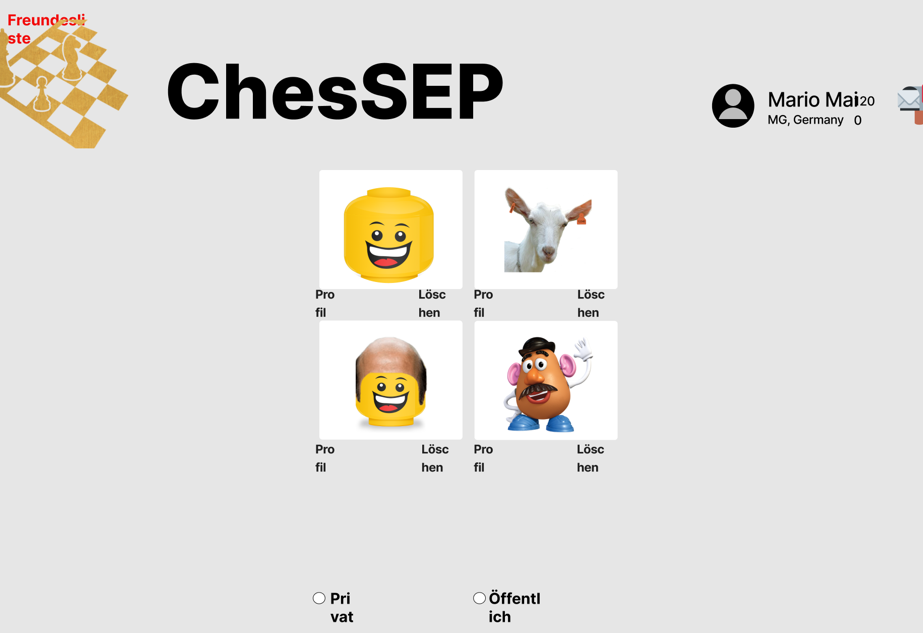


1.8 Profil des Users

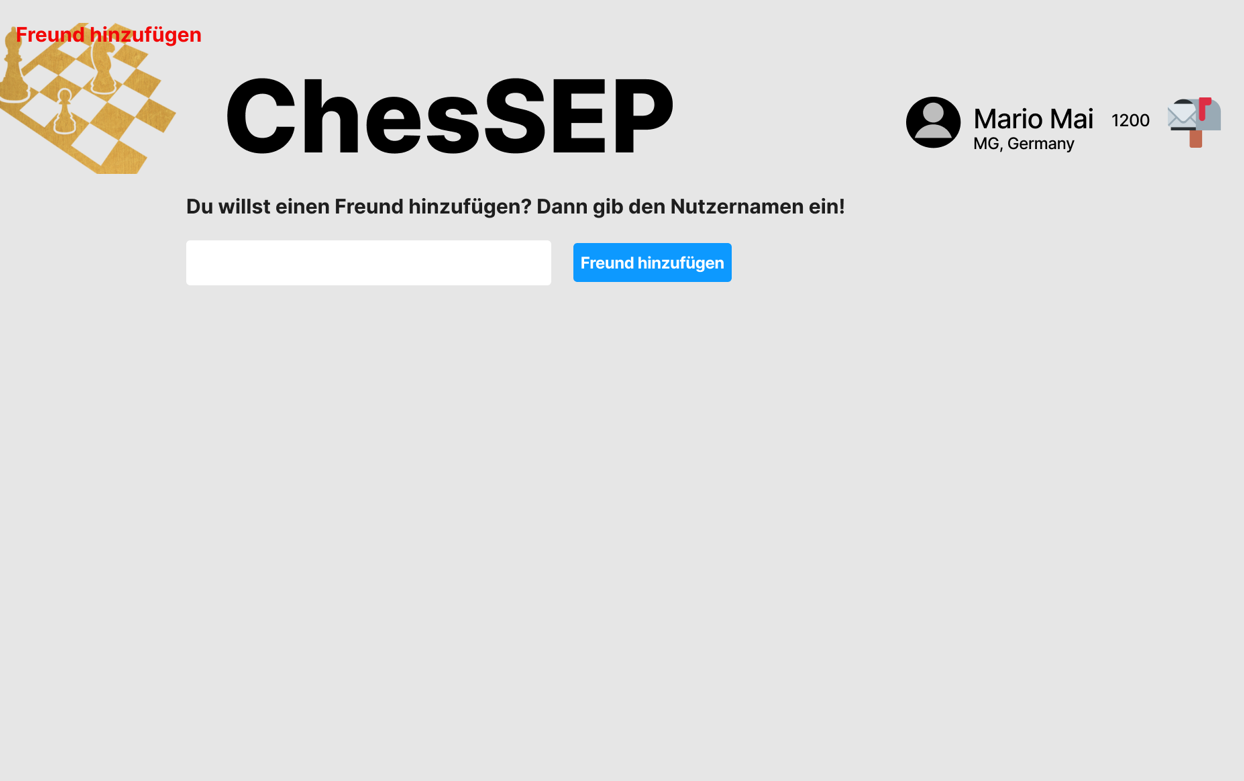


1.9 Fenster Authentifizierungscode

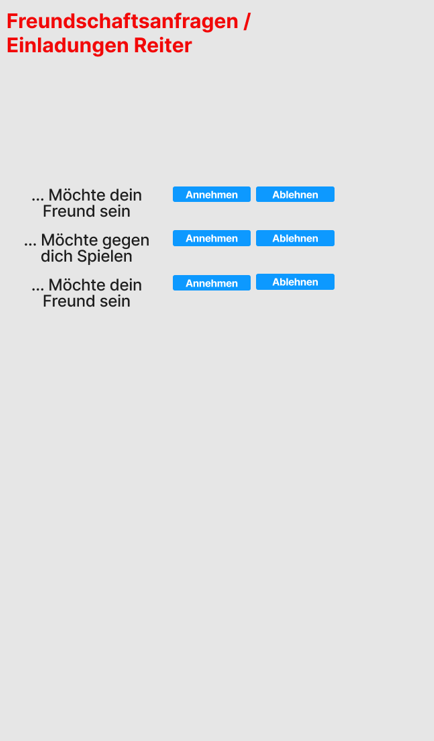


2.7 Freundesliste

2.8 Freund hinzufügen



2.9 Freundschaftsanfragen und Spieleinladungen



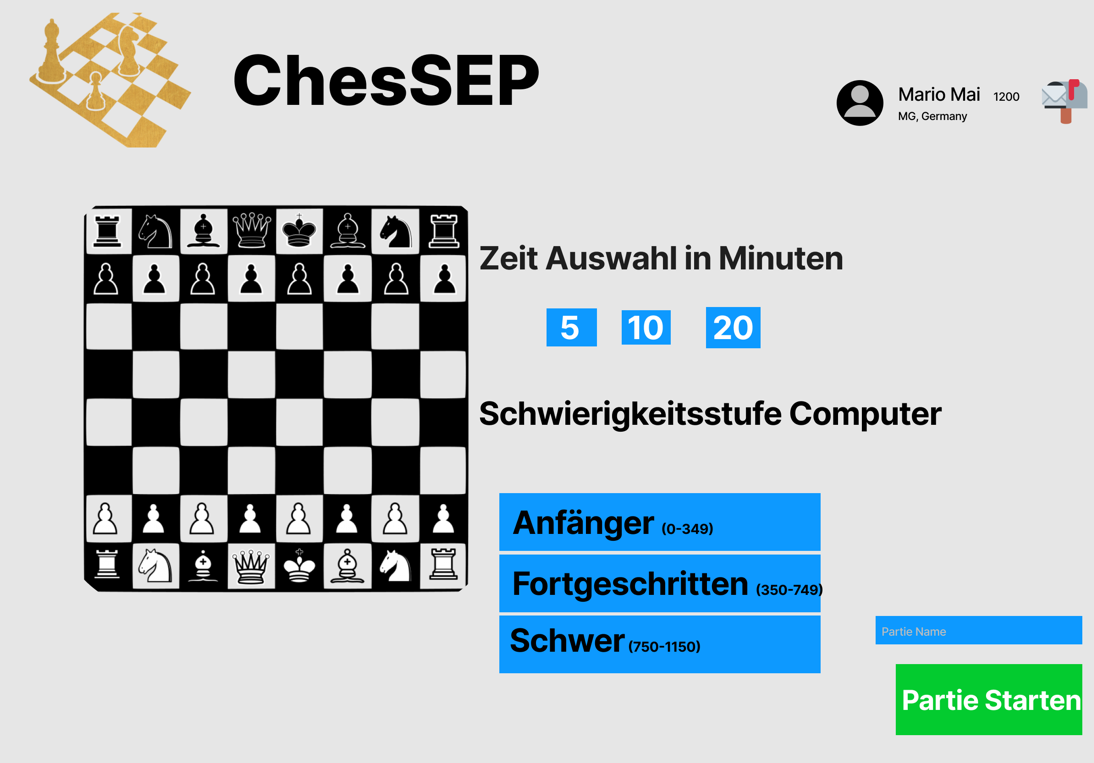
2.10 Profil von Freunden

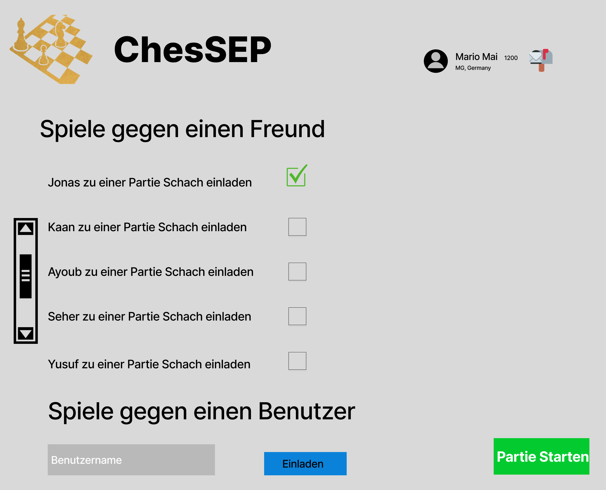
3.6 Startseite



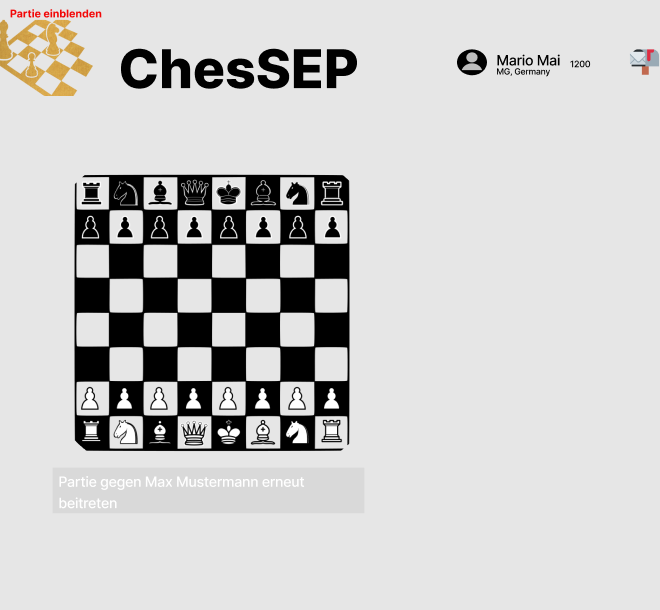
3.7 Startseite mit Login



3.8 Partie gegen Computer erstellen

3.9 Spiel gegen Benutzer erstellen

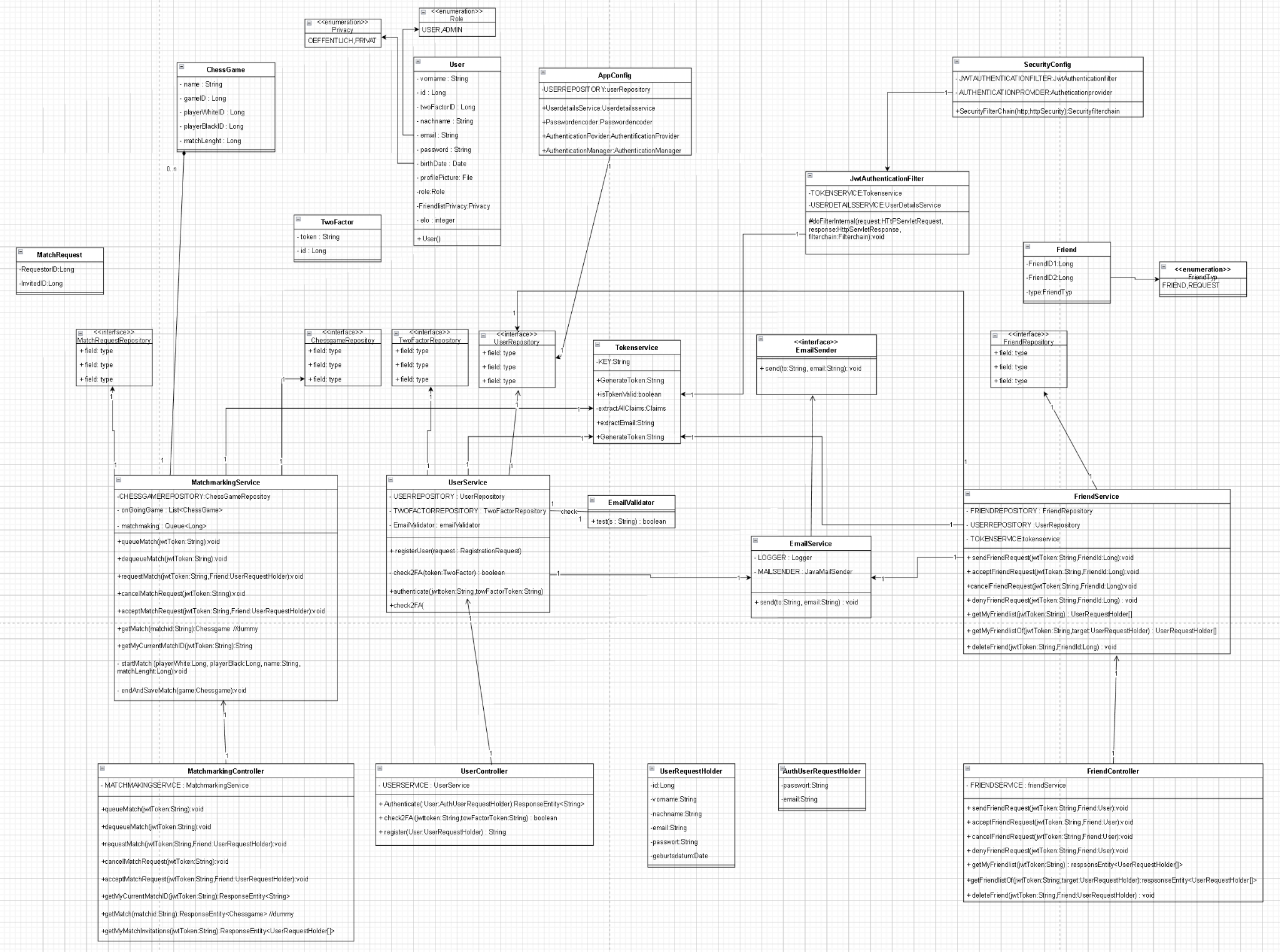
3.10 Partie wieder beitreten



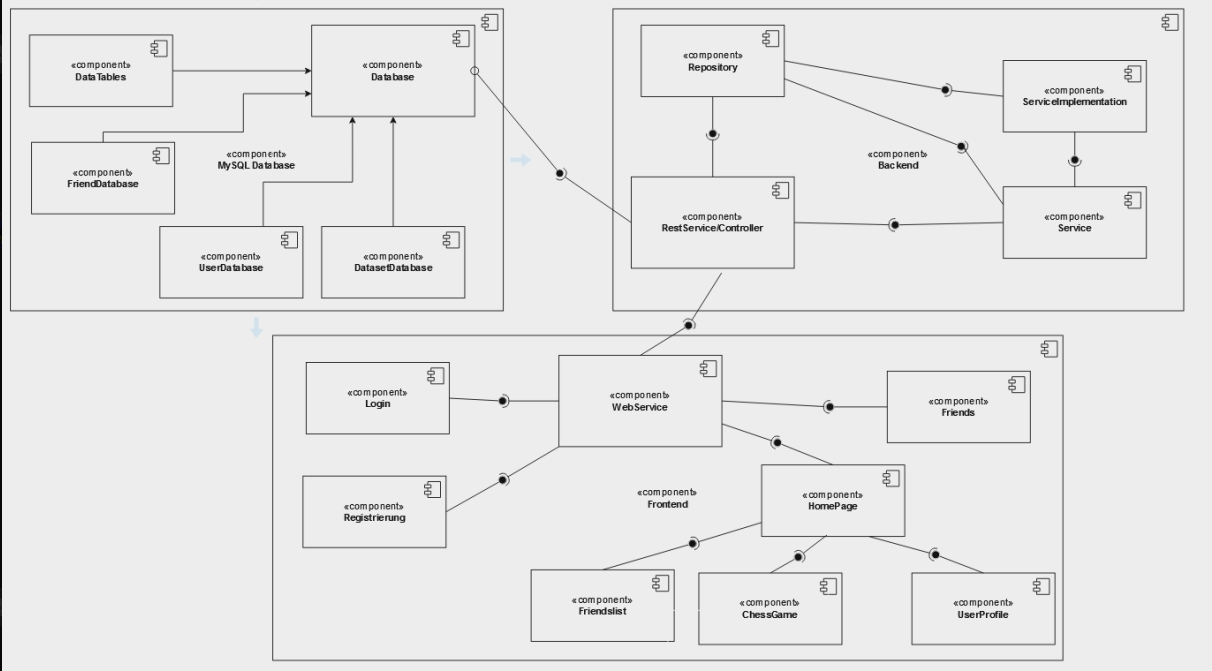
3.11 Schachpartie + Partie ausblenden



## Strukturdiagramme (Komponenten- und Klassendiagramme)

Klassendiagramm zu finden: https://git.uni-due.de/sep/wintersemester\_2023-24/gruppe-i-l/gruppe-i/-/blob/main/Projektmappe/Zyklus%201/Klassendiagramm\_Zyklus\_IV2.drawio

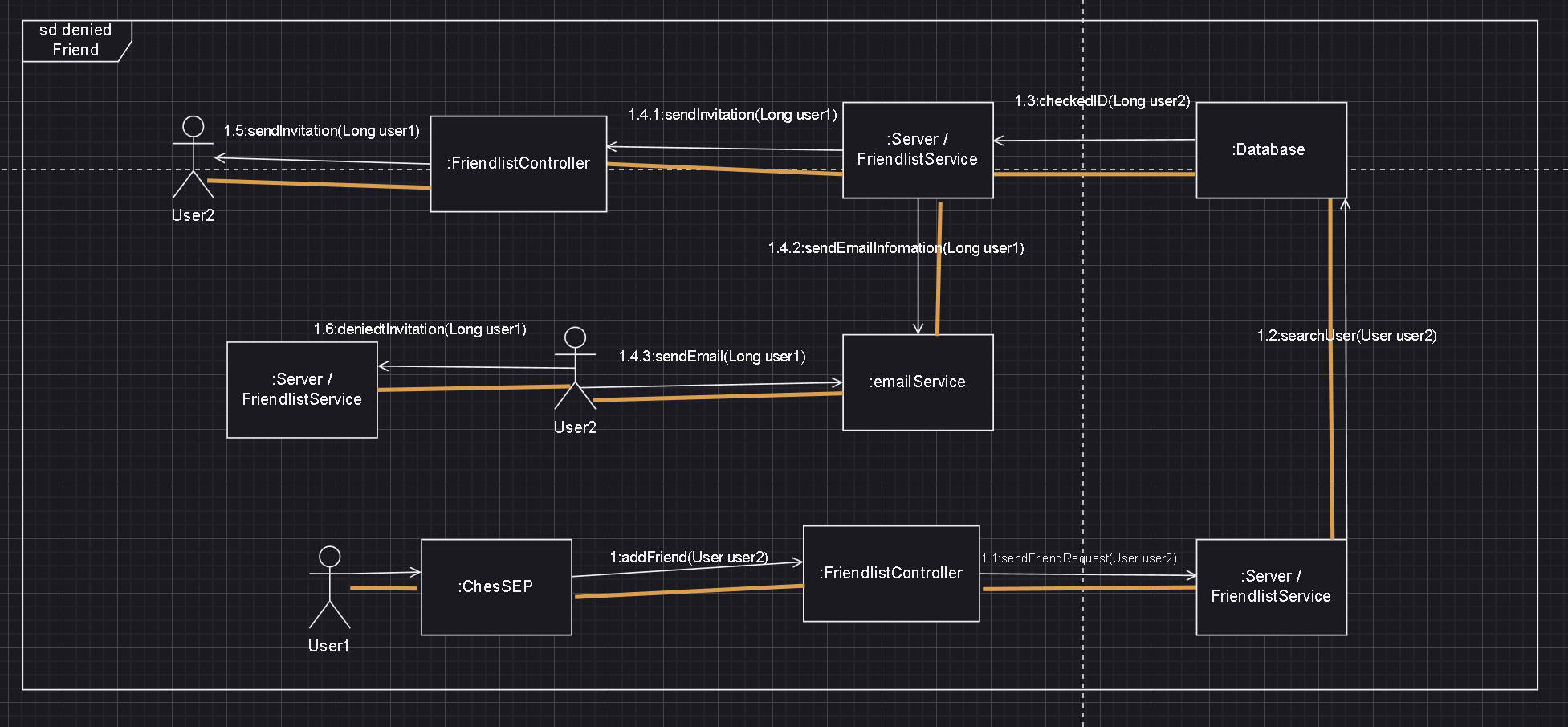
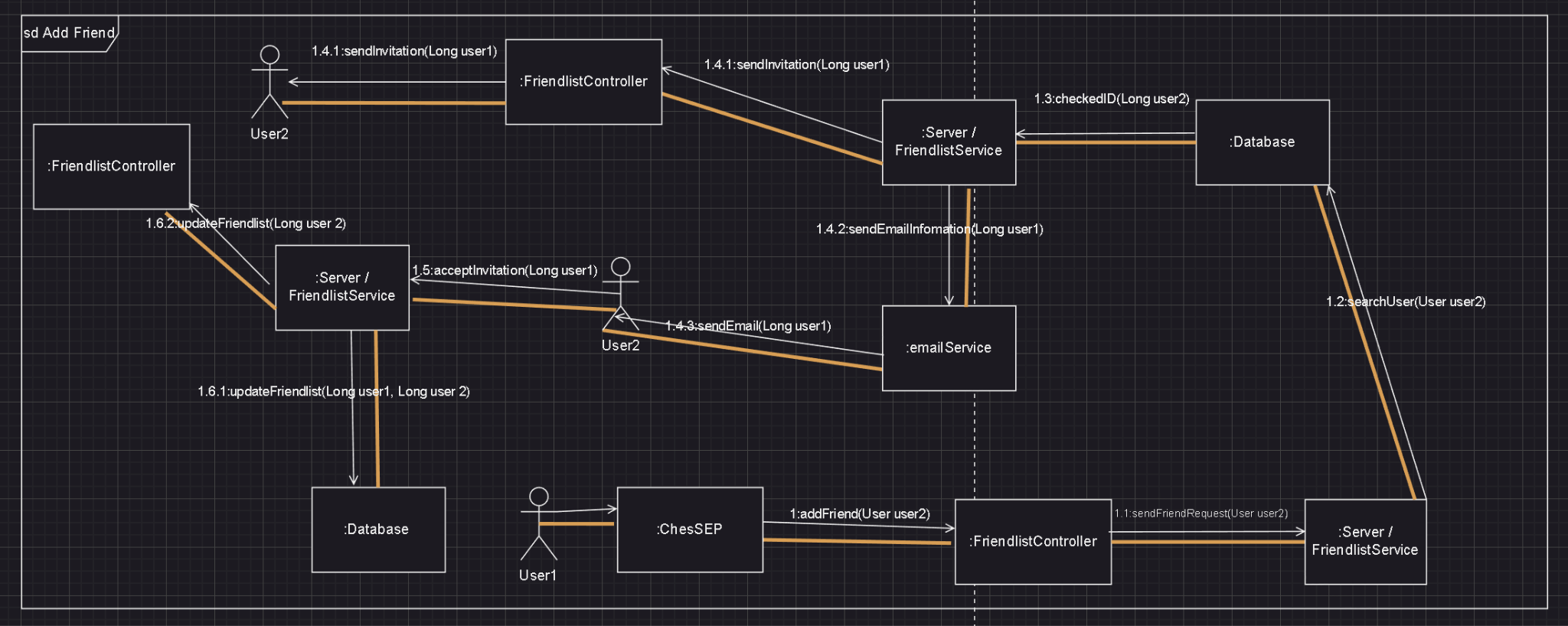
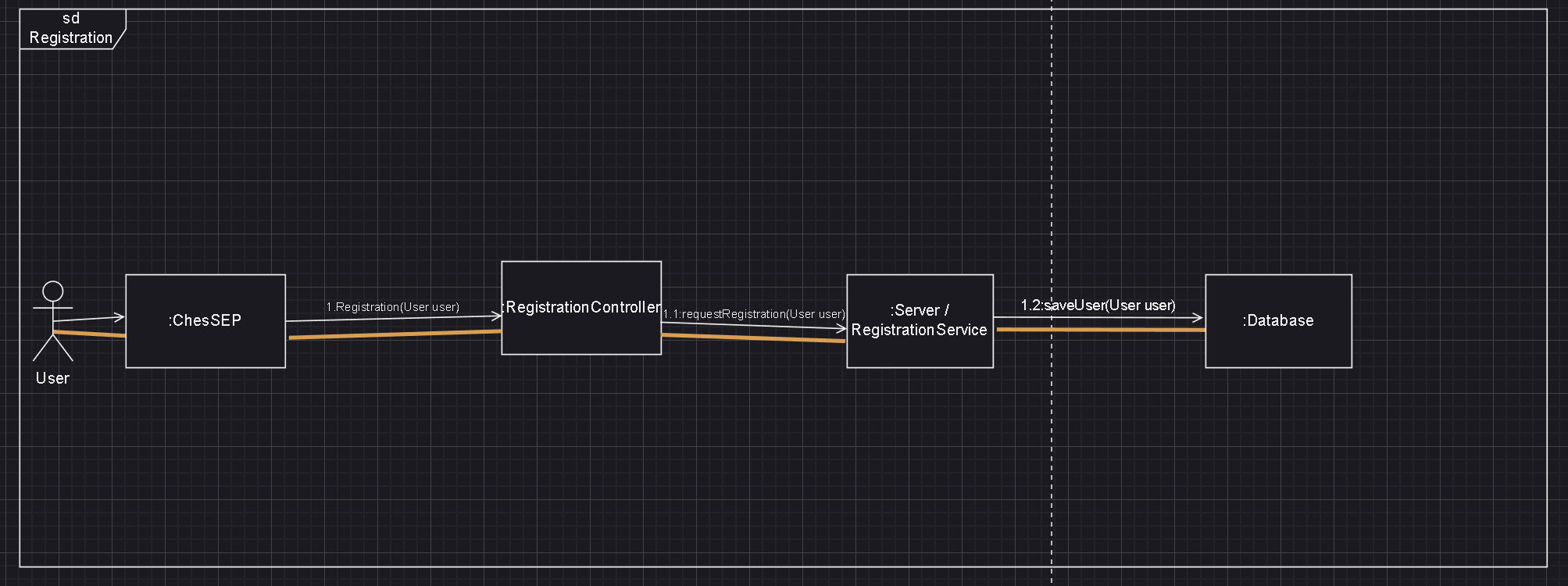
Komponentendiagramm zu finden: https://git.uni-due.de/sep/wintersemester\_2023-24/gruppe-i-l/gruppe-i/-/blob/main/Projektmappe/Zyklus%201/Komponentendiagramm.drawio



## Verhaltensdiagramme (Kommunikationsdiagramme)

Kommunikationsdiagramm zu finden:

https://git.uni-due.de/sep/wintersemester\_2023-24/gruppe-i-l/gruppe-i/-/blob/main/Projektmappe/Zyklus%201/Kommunikationsdiagramm\_Zyklus\_I.drawio



## Funktionalitätsplanung

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Funktionalität** | **Verantwortlicher** | **Abhängige**  **Funktionalitäten** | **Verknüpfte**  **User-Stories** | **Quellcode-**  **referenz** | **Status** |
| **1.** | **Backend** |  |  |  |  |  |
| 1.1 | Registrierung | Jonas | - | 1.1 | Package: com.ChesSEP.ChesSEP.User, com.ChesSEP.ChesSEP.Security | Fertig |
| 1.2 | Login | Jonas | 1.1 | 1.2 | Package: com.ChesSEP.ChesSEP.User, com.ChesSEP.ChesSEP.Security | Fertig |
| 1.3. | Zwei-Factor Authentifikation | Mario | 1.1, 1.2 | 1.2 | Package: com.ChesSEP.ChesSEP.TwoFactorAuthentication | Fertig |
| 1.3.1 | E-Mail Service | Mario | - | 1.3, 2.4 | Package: com.CheSEP.ChesSEP.Email | Fertig |
| 1.4 | Freundesliste | Yusuf | 1.1, 1.2 | 2.1, 2.2, 2.3, 2.5, 2.6 | Package: com.CheSEP.ChesSEP.Friendlist | Fertig |
| 1.5 | Schach Partie | Yusuf | 1.1, 1.2, 1.4 | 3.1, 3.2, 3.3, 3.4, 3.5 | Package: com.CheSEP.ChesSEP.ChessGame | Fertig |
| **2.** | **Frontend** |  |  |  |  |  |
| 2.1 | Header vor dem Login | Seher | - | - |  | Verworfen |
| 2.2 | Header nach dem Login | Kaan | - | 1.2 | Postlogin\header | Fertig |
| 2.3 | Registrierung | Seher | 2.11 | 1.1 | Prelogin\register | Fertig |
| 2.4 | Authentifizierung | Ayoub | 2.11, 2.3, 2.5 | 1.3, 1.4 | Prelogin\authenticate | Fertig |
| 2.5 | Login | Ayoub | 2.11, 2.3 | 1.2 | Prelogin\login | Fertig |
| 2.6 | Profilansicht vom User | Seher | 2.11, 2.3, 2.5 | 1.5, 1.10 | Postlogin\user-profile-view | Fertig |
| 2.7 | Freundesliste | Kaan | 2.11, 2.3, 2.5 | 2.1, 2.2, 2.6 | Postlogin\friendslist | Fertig |
| 2.8 | Freund hinzufügen | Kaan | 2.11, 2.3, 2.5 | 2.1, 2.5 | Postlogin\addfriend | Fertig |
| 2.9 | Freundschaftsanfragen + Spieleinladungen | Ayoub | 2.11, 2.3, 2.5 | 2.3, 2.4, 2.11, 3.4 | Postlogin\invitation | Fertig |
| 2.10 | Profil von Freunden | Seher | 2.11, 2.3, 2.5, 2.7 | 2.1 | Postlogin\friend-profile-view | Fertig |
| 2.11 | Startseite | Ayoub | - | 1.6, 3.1 | Postlogin\homepage | Fertig |
| 2.12 | Partie gegen Computer erstellen | Ayoub | 2.11 | 3.1 |  | Verworfen |
| 2.13 | Partie gegen Benutzer erstellen | Seher | 2.11 | 3.1, 3.2, 3.3 | Postlogin\create-play-against-user | Fertig |
| 2.14 | Partie ausblenden | Kaan | 2.11, 2.12, 2.13 | 3.5 |  | Verworfen |
| 2.15 | Partie wieder beitreten | Kaan | 2.11, 2.12, 2.13, 2.14 | 3.5 |  | Verworfen |
| 2.16 | Warteschlange für Partie Schach | Kaan | 2.11 |  | Postlogin\waiting | Fertig |
| 2.17 | Freundesliste vom Freund | Kaan | 2.7, 2.10 |  | Postlogin\friendlist-of-friends | Fertig |
| 2.18 | Schachpartie mit Platzhalter | Ayoub | 2.13 |  | Postlogin\play-game-against-user | Fertig |

## Systemtests

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | 17.11.2023 | | |
| **Tester** | Mario Mai | | |
| **SW-Version** | V 0.0.1 | | |
| **Vorbedin-gung(en)** | Nutzer „Mario Mai“, geboren “23.11.1995”, E-Mail “[testzweckeio@gmail.com](mailto:testzweckeio@gmail.com)” will sich mit dem Passwort “123456” registriert sich mit den genannten Daten. | | |
| **Schritt** | **Aktion (User)** | **Erwartete Reaktion (System)** | **√ / X** |
| 1 | Der Benutzer gibt den Vor- und Nachnamen„Mario Mai“ in das Registerfeld ein | Die Daten sollen angezeigt werden | **√** |
| 2 | Der Benutzer gibt die Email „testzweckeio@gmail.com“ in das Registerfeld | Die Email wird ordnungsgemäß angezeigt | **√** |
| 3 | Der Benutzer gibt sein Geburtsdatum “23.11.1995” in das Registerfeld ein | Das Feld soll das Datum in das richtige Format trennen | **√** |
| 4 | Der Benutzer gibt den sein Passwort„123456“ in das Register Feld ein | Das Passwortfeld wird in der Form \*\*\* angezeigt | **√** |
| 5 | Der Benutzer klickt auf den Registrierbutton | Man soll ein Assert bekommen, mit einer Meldung, ob die Daten gespeichert worden sind. |  |
| **Nachbe-dingung(en)** | Der Benutzer ist mit den Daten erfolgreich registriert und persistent in der Datenbank gespeichert | | **√** |
| **Testurteil** | **Test bestanden.** | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | 17.11.2023 | | |
| **Tester** | Mario Mai | | |
| **SW-Version** | V 0.0.1 | | |
| **Vorbedin-gung(en)** | Benutzer Mario Mai will sich mit seinen Logindaten anmelden | | |
| **Schritt** | **Aktion (User)** | **Erwartete Reaktion (System)** | **√ / X** |
| 1 | Der Benutzer trägt seine E-Mail in das Loginfeld ein | Daten werden angezeigt | **√** |
| 2 | Der Benutzer trägt sein Passwort in das Loginfeld ein | Das Passwort wird mit \*\*\* geschrieben | **√** |
| 3 | Der Benutzer klickt auf Anmelden | Man soll weitergeleitet werden, wenn der Login erfolgreich gewesen ist oder eine Meldung erhalten, wenn es fehlerhaft war | **√** |
| 4 | Der Benutzer kriegt einen 2-Faktor-Authentifizierung-Code auf seine angegebene E-Mail geschickt | Man soll eine E-Mail mit dem Code auf die angegebene E-Mail geschickt bekommen | **√** |
| 5 | Der Benutzer kann sich mit seinem 2-Faktor-Authentifizierungs-Code, erfolgreich anmelden | Bei erfolgreicher Eingabe soll man auf die Homepage weitergeleitet werden, oder man bekommt eine Meldung, dass es nicht geklappt hat | **√** |
| **Nachbe-dingung(en)** | Der Benutzer ist erfolgreich angemeldet und befindet sich auf der Startseite der ChesSEP Web-Anwendung | | **√** |
| **Testurteil** | **Test bestanden.** | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | 17.11.2023 | | |
| **Tester** | Mario Mai | | |
| **SW-Version** | V 0.0.1 | | |
| **Vorbedin-gung(en)** | Benutzer Mario Mai (“[testzweckeio@gmail.com](mailto:testzweckeio@gmail.com)”) will Benutzer X Y (“[mario-mai@gmx.net](mailto:mario-mai@gmx.net)”) als Freund hinzufügen (X Y ist schon persistent gespeichert) | | |
| **Schritt** | **Aktion (User)** | **Erwartete Reaktion (System)** | **√ / X** |
| 1 | Der Benutzer klickt auf das Feld “Freund hinzufügen” | Man soll in das Feld die E-Mail eintragen können | **√** |
| 2 | Der Benutzer wird benachrichtigt, ob die Anfrage erfolgreich verschickt wurde | Man soll ein Popup angezeigt bekommen, ob die Anfrage erfolgreich durchgegangen ist oder nicht (Der Freund soll eine E-Mail-Benachrichtigung bekommen) | **√** |
| 3 | Der Benutzer kann einen Freund nur einmal in seiner Freundesliste haben | Bei einer erneuten Anfrage, nachdem der Freund die Anfrage angenommen hat, soll das nichts passieren | **√** |
| **Nachbe-dingung(en)** | Der Benutzer hat seine Freundesliste mit X Y erweitert | | **√** |
| **Testurteil** | **Test bestanden.** | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | 17.11.2023 | | |
| **Tester** | Mario Mai | | |
| **SW-Version** | V 0.0.1 | | |
| **Vorbedin-gung(en)** | Benutzer Mario Mai hat eine Freundesliste | | |
| **Schritt** | **Aktion (User)** | **Erwartete Reaktion (System)** | **√ / X** |
| 1 | Der Benutzer klickt auf die Ansicht seiner Freundesliste | Wenn Freunde vorhanden sind, sollen alle Freunde mit Daten angezeigt werden | **√** |
| 2 | Der Benutzer klickt auf das Profil seines Freundes | Man wird weitergeleitet und kommt auf die Profilseite des Freundes und sieht Profilbild und Daten | **√** |
| 3 | Der Benutzer klickt auf die Freundesliste seines Freundes | Je nachdem, wie der User seine Privatsphäre angegeben hat, wird die Freundesliste angezeigt oder nicht. (Private = nicht angezeigt, Oeffentlich = wird angezeigt) | **√** |
| 4 | Der Benutzer stellt die Account Privatsphäre um | Durch die Änderung wird die Freundeslisten Ansicht verändert | **√** |
| 5 | Der Benutzer löscht seinen Freund | Der Freund wird aus der Freundesliste des Benutzers entfernt und visa verce | **√** |
| **Nachbe-dingung(en)** | Der Benutzer hat seine Freundesliste bearbeitet (Freund gelöscht und privacy geändert) | | **√** |
| **Testurteil** | **Test bestanden.** | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Tester** | Mario Mai | | |
| **SW-Version** | V 0.0.1 | | |
| **Vorbedin-gung(en)** | Benutzer Mario Mai hat eine Freundesliste | | |
| **Schritt** | **Aktion (User)** | **Erwartete Reaktion (System)** | **√ / X** |
| 1 | Der Benutzer klickt auf die Ansicht seiner Freundesliste | Wenn Freunde vorhanden sind, sollen alle Freunde mit Daten angezeigt werden | **√** |
| 2 | Der Benutzer klickt auf das Profil seines Freundes | Man wird weitergeleitet und kommt auf die Profilseite des Freundes und sieht Profilbild und Daten | **√** |
| 3 | Der Benutzer klickt auf die Freundesliste seines Freundes | Je nachdem, wie der User seine Privatsphäre angegeben hat, wird die Freundesliste angezeigt oder nicht. (Private = nicht angezeigt, Oeffentlich = wird angezeigt) | **√** |
| 4 | Der Benutzer stellt die Account Privatsphäre um | Durch die Änderung wird die Freundeslisten Ansicht verändert | **√** |
| 5 | Der Benutzer löscht seinen Freund | Der Freund wird aus der Freundesliste des Benutzers entfernt und visa verce | **√** |
| **Nachbe-dingung(en)** | Der Benutzer hat seine Freundesliste bearbeitet (Freund gelöscht und privacy geändert) | | **√** |
| **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

Freie Ports:

* 3307
* 4200
* 8080

Docker muss installiert sein

## Installationsanleitung

Klone das GIT-Repository und führe die Befehle aus der README aus

## Bedienungsanleitung