Sprint 3

Freyschmidt, Henry Lewis (HLF)
Hama, Zana Salih (ZSH)
Krasnovska, Paula (PK)
Krüger, Lucas (LK)
Prüger, Marvin Oliver (MOP)
Seep, Tom-Malte (TMS)
Zabel, Steven (SZ)
Henry J. v. Rooyen (HvR)

9. Juni 2024

	Verantwortung	Inhalt	Grafik	Korrektur
Backlog				
Änderungen am Backlog	SZ	SZ	-	LK
User-Storys & Tasks	SZ	SZ	-	Alle
Aufwandsschätzung der Tasks	Alle	Alle	-	LK
Ablaufplan zu den Tasks	SZ	SZ	SZ	LK
Feinentwurf				
Komponentendiagramm	SZ	LK	SZ	LK, HLF
Klassendiagramme	SZ	LK, HLF	SZ	LK, HLF
Implementation und Tests				
HTTP-Request-Tests	LK	LK	-	-
Logic-Tests	HLF	HLF	-	LK
Datenbank-Tests	MOP	MOP	-	HLF
Abnahmetest.T2	ZSH	ZSH	-	PK
Tracing - Komponenten im Code	LK, SZ	LK, SZ	-	-
Tracing - Beendeter Aufgaben	LK	LK	-	-
Laufender Prototyp	PK	PK, ZSH	PK, ZSH	-
Abweichung von Sprintplanung	PK	PK	-	-

Task	Kurzbeschreibung	Name	Anteil%
R1.F1	Erweitere P4.F1.	PK	100%
R1.F2	Erweitere P4.F2.	PK	100%
R1.F3	Erstelle Frontend-Funktionalität zum Anpassen der rechtebezogenen Rolle eines Nutzers.	PK	100%
R1.B1	Erweitere P4.B1.	HvR LK	90% 10%
R1.B2	Erstelle Backend-Funktionalität zum Anpassen der rechtebezogenen Rolle eines Nutzers.	LK	100%
R1.D1	Erstelle Speicher für rechtebezogene Rollen.	TMS MOP	50% 50%
R1.D2	Erweitere Speicher für Nutzer.	TMS MOP	50% 50%
R1.D3	Erstelle Daten-Funktionalität zum Anpassen der rechtebezogenen Rolle eines Nutzers.	TMS MOP	50% 50%
P4.F1	Erstelle Anzeige für Nutzer.	PK	100%
P4.F2	Erstelle Frontend-Funktionalität zum Anzeigen von P4.F1.	PK	100%
P4.F3	Erweitere Frontend-Funktionalität zum Anzeigen eines Profils.	LK	100%
P4.B1	Erstelle Backend-Funktionalität zum Ausgeben aller Nutzer.	HLF LK	90% 10%
P4.D1	Erstelle Daten-Funktionalität zur Ausgabe aller Nutzer.	TMS MOP	50% 50%
U2.F1	Erstelle Anzeige für eine User-Story.	ZSH	100%
U2.F2	Erstelle Frontend-Funktionalität zum Anzeigen einer User-Story.	ZSH	100%
U2.F3	Verändere Frontend-Funktionalität zum Anpassen einer User-Story.	ZSH	100%
U2.F4	Verändere Frontend-Funktionalität zum Erstellen einer User-Story.	ZSH	100%
U2.B1	Erstelle Backend-Funktionalität zum Ausgeben einer User-Story.	LK HLF	50% 50%
U2.D1	Erstelle Daten-Funktionalität zum Ausgeben einer User-Story.	TMS MOP	50% 50%
T2.F1	Erstelle Anzeige für eine Task.	ZSH	100%
T2.F2	Erstelle Frontend-Funktionalität zum Anzeigen einer Task.	ZSH	100%
T2.F3	Verändere Frontend-Funktionalität zum Anpassen einer Task.	ZSH PK	60% 40%
T2.F4	Verändere Frontend-Funktionalität zum Erstellen einer Task.	ZSH PK	60% 40%
T2.B1	Erstelle Backend-Funktionalität zum Ausgeben einer Task.	LK	100%
T2.D1	Erstelle Daten-Funktionalität zum Ausgeben einer Task.	TMS MOP	50% 50%
T6.F1	Erweitere T2.F1.	PK	100%
T6.F2	Erweitere Frontend-Funktionalität zum Anpassen einer Task.	PK	100%
T6.F3	Erweitere Frontend-Funktionalität zum Anpassen einer Task.	PK	100%
T6.F4	Erweitere Frontend-Funktionalität zum Anzeigen aller Tasks.	ZSH PK	50% 50%

Task	Kurzbeschreibung	Name	Anteil%
T7.F1	Erweitere T2.F1.	ZSH	100%
T7.F2	Erweitere Frontend-Funktionalität zum Anpassen einer Task.	ZSH	100%
T7.F3	Erweitere Frontend-Funktionalität zum Anzeigen aller Tasks.	ZSH	100%
T7.F4	Erweitere Frontend-Funktionalität zum Erstellen einer Task.	ZSH	100%
T7.B1	Erweitere T2.B1.	HLF LK	90% 10%
T7.B2	Erweitere Backend-Funktionalität zum Anpassen einer Task.	HLF LK	90% 10%
T7.B3	Erweitere Backend-Funktionalität zum Ausgeben aller Task.	HLF LK	90% 10%
T7.B4	Erweitere Backend-Funktionalität zum Erstellen einer Task.	HLF LK	90% 10%
T7.D1	Erweitere Speicher für Tasks.	TMS MOP	50% 50%
T7.D2	Erstellen Daten-Funktionalität zum Anpassen einer Task.	TMS MOP	50% 50%
T7.D3	Erweitere Daten-Funktionalität zum Erstellen einer Task.	TMS MOP	50% 50%
T8.F1	Erweitere T2.F1.	PK	100%
T8.F2	Erweitere Frontend-Funktionalität zum Anpassen einer Task.	PK	100%
T8.F3	Erweitere Frontend-Funktionalität zum Erstellen einer Task.	PK	100%
T8.B1	Erweitere T2.B1.	LK	100%
T8.B2	Erweitere Backend-Funktionalität zum Anpassen einer Task.	LK	100%
T8.B3	Erweitere Backend-Funktionalität zum Erstellen einer Task.	LK	100%
T8.D1	Erweitere Speicher für Tasks.	TMS MOP	50% 50%
T8.D2	Erstelle Daten-Funktionalität zum Anpassen der Liste zugeordneter Nutzer einer Task.	TMS MOP	50% 50%
T8.D3	Erweitere Daten-Funktionalität zum Erstellen einer Task.	TMS MOP	50% 50%
T9.F1	Erweitere T2.F1.	ZSH	100%
T9.F2	Erweitere Frontend-Funktionalität zum Anpassen einer Task.	ZSH	100%
T9.F3	Erweitere Frontend-Funktionalität zum Erstellen einer Task.	ZSH	100%
T9.B1	Erweitere T2.B1.	HLF LK	90% 10%
T9.B2	Erweitere Backend-Funktionalität zum Anpassen einer Task.	HLF LK	90% 10%
T9.B3	Erweitere Backend-Funktionalität zum Erstellen einer Task.	HLF LK	90% 10%
T9.D1	Erweitere Speicher für Tasks.	TMS MOP	50% 50%
T9.D2	Erstelle Daten-Funktionalität zum Anpassen der Abgabefrist einer Task.	TMS MOP	50% 50%
T9.D3	Erweitere Daten-Funktionalität zum Erstellen einer Task.	TMS MOP	50% 50%

Task	Kurzbeschreibung	Name	Anteil%
T10.F1	Erweitere T2.F1.	PK	100%
T10.F2	Erweitere Frontend-Funktionalität zum Anpassen einer Task.	PK	100%
T10.F3	Erweitere Frontend-Funktionalität zum Erstellen einer Task.	PK	100%
T10.B1	Erweitere T2.B1.	HLF LK	90% 10%
T10.B2	Erweitere Backend-Funktionalität zum Anpassen einer Task.	HLF LK	90% 10%
T10.B3	Erweitere Backend-Funktionalität zum Erstellen einer Task.	HLF LK	90% 10%
T10.D1	Erweitere Speicher für Tasks.	TMS MOP	50% 50%
T10.D2	Erstelle Daten-Funktionalität zum Anpassen der Aufwandsschätzung einer Task.	TMS MOP	50% 50%
T10.D3	Erweitere Daten-Funktionalität zum Erstellen einer Task.	TMS MOP	50% 50%
T11.F1	Erweitere T2.F1.	PK	100%
T11.F2	Erweitere Frontend-Funktionalität zum Anpassen einer Task.	PK	100%
T11.B1	Erweitere T2.B1	HvR LK	90% 10%
T11.B2	Erweitere Backend-Funktionalität zum Anpassen einer Task.	HvR LK	90% 10%
T11.D1	Erweitere Speicher für Tasks.	TMS MOP	50% 50%
T11.D2	Erstelle Daten-Funktionalität zum Anpassen der Bearbeitungszeit einer Task.	TMS MOP	50% 50%
T16.F1	Erweitere T2.F1.	PK	100%
T16.F2	Erweitere Frontend-Funktionalität zum Anpassen einer Task.	PK	100%
T16.F3	Erweitere Frontend-Funktionalität zum Erstellen einer Task.	PK	100%
T16.B1	Erweitere T2.B1.	LK	100%
T16.B2	Erweitere Backend-Funktionalität zum Anpassen einer Task.	LK	100%
T16.B3	Erweitere Backend-Funktionalität zum Erstellen einer Task.	LK	100%
T16.D1	Erstelle Funktionalität zum Anpassen der Zuordnung zu einer Task-Board einer Task.	TMS MOP	50% 50%
T16.D2	Erweitere Daten-Funktionalität zum Erstellen einer Task.	TMS MOP	50% 50%
T17.F1	Visualisierung der Aufwandsschätzung und Bearbeitungszeit einer Task	ZSH	100%
TB2.F1	Auswahl eines Task-Boards	ZSH	100%
TB2.F2	Anzeigen eines Task-Boards	ZSH	100%
TB2.B1	Aufrufen von TB2.D1	HLF LK	50% 50%

Task	Kurzbeschreibung	Name	Anteil%
TB2.D1	Ausgabe eines Task-Boards	TMS MOP	50% 50%
TB3.F1	Funktionalität zum Verschieben einer Task in eine andere Task-Liste	ZSH	100%
TB3.B1	Funktionalität anpassen der Task-Liste einer Task	LK	100%
TB3.D1	Zuordnen einer Task zu einer Task-Liste	TMS MOP	50% 50%
TB5.F1	Anzeige zugeordneter Nutzer	PK	100%
TB5.B1	Erweitere Task um Attribut Liste zugeordneter Nutzer	LK	100%
TB8.F1	Visualisierung von Aufwandsschätzung und Bearbeitungszeit	ZSH	100%
TB8.B1	Erweitere Task mit Attribut Aufwandsschätzung und Bearbeitungszeit	LK HLF	50% 50%
TB9.F1	Product-Owner kann Task-Boards erstellen	ZSH PK	50% 50%
TB9.B1	Backend-Funktionalität zum Erstellen eines Task-Boards	LK HLF	50% 50%
TB9.D1	Erstelle Speicher für Task-Boards	TMS MOP	50% 50%
TB9.D2	Erstelle Speicher für Task-Listen zu Task-Board	TMS MOP	50% 50%
TB9.D3	Erstellen eines Task-Board mit Task-Listen	TMS MOP	50% 50%
TB10.F1	Product-Owner kann Task-Board umbenennen	PK	100%
TB10.B1	Aufrufen von TB10.D1	LK	100%
TB10.D1	Umbenennen eines Task-Board	TMS MOP	50% 50%
TB11.F1	Produkt-Owner kann Task-Boards löschen	ZSH PK	90% 10%
TB11.B1	Aufrufen TB11.D1, TB11.D2 und belegen von betroffenen Task mit Default Task-List IDs	LK	100%
TB11.D1	Löschen eines Task-Boards	TMS MOP	50% 50%
TB11.D2	Löschen aller Task-Listen eines Task-Boards	TMS MOP	50% 50%
	Logic-Tests	HLF	100%
	HTTP-Tests	LK	100%
	DB-Tests	MOP TMS	50% 50%
	Product Owner	SZ	100%
	Anlegen und Verwalten von Tasks	SZ	100%

Inhaltsverzeichnis

1 Backlog

1.1 Änderungen am Backlog

Element	von	zu
R1	Als Manager kann ich die rechtebezogene Rolle eines Nutzers festlegen, um die Rechte des Nutzers im Projekt zu definieren.	Als Manager kann ich auf einem Profil die rechtebezogene Rolle eines Nutzers festlegen, um die Rechte des Nutzers im Projekt zu definieren.
R3		Als Manager kann ich eine visuelle Rolle umbenennen, um diese anzupassen.
R4		Als Manager kann ich eine visuelle Rolle löschen, damit Nutzer sich dieser nicht mehr zuordnen können.
R5		Als Nutzer kann ich mich auf meinem Profil einer visuellen Rolle zuordnen, damit diese auf meinem Profil sichtbar ist.

Tabelle 1: Rechtebezogene Rollen sollen auf dem Profil eines Nutzers festgelegt werden. Je rechtebezogener Rolle sollen Nutzer sich selbstständig ihre visuelle Rolle aussuchen können. Visuelle Rollen müssen umbenannt und gelöscht werden können.

Element	von	zu
T12	Als Nutzer kann ich einen Schätzungstracker aufrufen, welcher für beliebige, fertige Tasks eine Visualisierung ausgibt, wie sich die Aufwandsschätzungen von den Bearbeitungszeiten unterscheiden, um die Aufwandsschätzungen bewerten zu können.	
T17		Als Nutzer kann ich zu einem fertigen Task eine Visualisierung einsehen, welche darstellt, wie sich die Aufwandsschätzung und die Bearbeitungszeit unterscheiden, um die Aufwandsschätzung bewerten zu können.
TB8		Als Nutzer kann ich auf einem Task-Board zu der Menge an fertigen Tasks des Task-Boards eine Visualisierung einsehen, welche darstellt, wie sich die Aufwandsschätzungen und die Bearbeitungszeiten der Tasks unterscheiden, um die Aufwandsschätzungen bewerten zu können.

Tabelle 2: Das Konzept eines vom Nutzer gesteuerten Schätzungstrackers soll durch vom System sinnvoll positionierte Schätzungstrackings ersetzt werden.

Element	von	zu
TL8	Als Nutzer stehen mir zu jedem Task-Board die Task-Listen „freie Tasks“, „Tasks in Bearbeitung“, „Tasks unter Review“, „Tasks unter Test“ und „fertiggestellte Tasks“ zur Verfügung, um Tasks während der Bearbeitung einordnen zu können.	
TB9		Als Nutzer kann ich ein Task-Board mit den Task-Listen „freie Tasks“, „Tasks in Bearbeitung“, „Tasks unter Review“, „Tasks unter Test“ und „fertiggestellte Tasks“ erstellen, um Tasks strukturiert zu bearbeiten.
TB10		Als Nutzer kann ich ein Task-Board umbenennen, um es anzupassen.
TB11		Als Nutzer kann ich ein Task-Board löschen, um es aus dem System zu entfernen.

Tabelle 3: Nutzer sollten eigenständig Task-Boards erstellen, umbenennen und löschen können. TL8 wurde gelöscht, da TB9 diese umfasst.

Element	von	zu
T2	Als Nutzer kann ich eine Task auswählen, um mir alle zu ihr gespeicherten Informationen anzeigen lassen.	Als Nutzer kann ich eine Task auswählen, um mir alle zu ihr gespeicherten Informationen anzeigen zu lassen.
T10	Als Nutzer kann ich eine Task mit einer Aufwandseinschätzung versehen, um auszudrücken, wie lange die Bearbeitung der Task schätzungsweise dauert.	Als Nutzer kann ich eine Task mit einer Aufwandsschätzung versehen, um auszudrücken, wie lange die Bearbeitung der Task schätzungsweise dauert.

Tabelle 4: Die User-Storys wurden korrigiert.

1.2 Planung des Sprints

Die Tasks sind eingeteilt in die Teams [F] Frontend, [B] Backend und [D] Daten-Ebene. Das Frontend umfasst die Funktionalität zur Präsentation, das Backend umfasst die Funktionalität zur Logik und die Daten-Ebene umfasst die Funktionalität zur Datenspeicherung, -ausgabe und -manipulation.

- Product Owner: SZ
- Frontend-Entwicklung: PK, ZSH
- Backend-Entwicklung: HLF, LK, HvR
- Daten-Entwicklung: MOP, TMS

In diesem Sprint werden folgende inhaltliche Komplexe angegangen:

1. die Möglichkeit, Profile anderer Nutzer einzusehen,
2. die Einführung der Manager-Rolle, sodass diese rechtebezogene und visuelle Rollen verwaltet,
3. die Erweiterung von User-Storys und Tasks, sodass diese im System vollständig beschrieben und genutzt werden können und
4. die Einführung von Task-Listen und Task-Boards, sodass mittels der Task-Listen auf den Task-Boards der Status der Bearbeitung von Tasks dargestellt werden kann.

1.2.1 [R] Rolle

⊗ **R1** Als Manager kann ich auf einem Profil die rechtebezogene Rolle eines Nutzers festlegen, um die Rechte des Nutzers im Projekt zu definieren.

- ⊗ **R1.F1** Erweitere P4.F1, sodass die rechtebezogene Rolle eines Nutzers und alle rechtebezogenen Rollen als Auswahl angezeigt werden und zu einem Nutzer die Auswahl einer rechtebezogenen Rolle möglich ist.
  - ∗ Aufwandsschätzung: 1h durch PK
  - ∗ Bearbeitung: 1h durch PK
  - ∗ Review: 0,25h durch ZSH
  - ∗ Test: 0,25h durch ZSH
- ⊗ **R1.F2** Erweitere P4.F2 um alle rechtebezogenen Rollen.
  - ∗ Aufwandsschätzung: 1h durch PK
  - ∗ Bearbeitung: 1h durch PK
  - ∗ Review: 0,25h durch ZSH
  - ∗ Test: 0,25h durch ZSH
- ⊗ **R1.F3** Erstelle Frontend-Funktionalität zum Anpassen der rechtebezogenen Rolle eines Nutzers, welche, wenn ein Manager mit gültiger Session sie nutzt, R1.F1 ausliest und R1.B2 aufruft.
  - ∗ Aufwandsschätzung: 1h durch PK
  - ∗ Bearbeitung: 0,25h durch PK
  - ∗ Review: 0,25h durch ZSH
  - ∗ Test: 0,25h durch ZSH
- ⊗ **R1.B1** Erweitere P4.B1 um die rechtebezogenen Rollen der Nutzer.
  - ∗ Aufwandsschätzung: 1h
  - ∗ Bearbeitung: 1h durch HvR
- ⊗ **R1.B2** Erstelle Backend-Funktionalität zum Anpassen der rechtebezogenen Rolle, welche R1.D3 aufruft.
  - ∗ Aufwandsschätzung: 1,5h
  - ∗ Bearbeitung: 1,5h durch LK
  - ∗ Test: 0,5h durch LK
- ⊗ **R1.D1** Erstelle Speicher für rechtebezogene Rollen mit den Rollen „Developer“, „Product Owner“, „Manager“ und „Admin“.
  - ∗ Aufwandsschätzung: 3,5h
  - ∗ Bearbeitung: 3h durch TMS, MOP
  - ∗ Test: 0,5h durch TMS, MOP
- ⊗ **R1.D2** Erweitere Speicher für Nutzer um eine rechtebezogene Rolle mit Default-Wert „Developer“.
  - ∗ Aufwandsschätzung: 0,5h
  - ∗ Bearbeitung: 0,5h durch TMS, MOP
  - ∗ Test: 0,5h durch TMS, MOP
- ⊗ **R1.D3** Erstelle Daten-Funktionalität zum Anpassen der rechtebezogenen Rolle eines Nutzers.
  - ∗ Aufwandsschätzung: 1h
  - ∗ Bearbeitung: 0,75h durch TMS, MOP

    - Test: 0,5h durch TMS, MOP

◯ **R2** Als Manager kann ich zu einer rechtebezogenen Rolle eine visuelle Rolle erstellen, damit Nutzer sich anhand dieser einordnen können.

  ◯ **R2.F1** Erstelle Anzeige für Rollen, welche die Auswahl einer rechtebezogenen Rolle ermöglicht, zu dieser alle verlinkten visuellen Rollen anzeigen kann und die Eingabe einer visuellen Rolle ermöglicht.

  ◯ **R2.F2** Erstelle Frontend-Funktionalität zum Anzeigen von R2.F1, welche, wenn ein Nutzer mit gültiger Session sie nutzt, R2.B1 aufruft und R2.F1 anzeigt.

  ◯ **R2.F3** Erstelle Frontend-Funktionalität zum Erstellen visueller Rollen, welche, wenn ein Manager mit gültiger Session sie nutzt, R2.F1 ausliest und R2.B2 aufruft.

  ◯ **R2.B1** Erstelle Backend-Funktionalität zum Ausgeben aller zu einer rechtebezogenen Rolle verlinkten visuellen Rollen, welche R2.D2 aufruft und die visuellen Rollen ausgibt.

  ◯ **R2.B2** Erstelle Backend-Funktionalität zum Erstellen visueller Rollen, welche R2.D3 aufruft.

  ◯ **R2.D1** Erstelle Speicher für visuelle Rollen.

    - Aufwandsschätzung: 0,5h
    - Bearbeitung: 0,5h durch TMS, MOP
    - Test: 0,5h durch TMS, MOP

  ◯ **R2.D2** Erstelle Daten-Funktionalität zum Ausgeben aller zu einer rechtebezogenen Rolle verlinkten visuellen Rollen.

    - Aufwandsschätzung: 0,75h
    - Bearbeitung: 0,5h durch TMS, MOP
    - Test: 0,5h durch TMS, MOP

  ◯ **R2.D3** Erstelle Daten-Funktionalität zum Erstellen visueller Rollen, welche zu einer rechtebezogenen Rolle eine visuelle Rolle erstellt.

    - Aufwandsschätzung: 0,5h
    - Bearbeitung: 0,5h durch TMS, MOP
    - Test: 0,5h durch TMS, MOP

◯ **R3** Als Manager kann ich eine visuelle Rolle umbenennen, um sie anzupassen.

  ◯ **R3.F1** Erweitere R2.F1, sodass angezeigte visuelle Rollen umbenannt werden können.

  ◯ **R3.F2** Erstelle Frontend-Funktionalität zum Anpassen visueller Rollen, welche, wenn ein Manager mit gültiger Session sie nutzt, R3.F1 ausliest und R3.B1 aufruft.

  ◯ **R3.B1** Erstelle Backend-Funktionalität zum Anpassen visueller Rollen, welche R3.D1 aufruft.

  ◯ **R3.D1** Erstelle Daten-Funktionalität zum Umbenennen visueller Rollen, welche den Namen einer bestehenden visuellen Rolle anpasst.

    - Aufwandsschätzung: 0,25h
    - Bearbeitung: 0,25h durch TMS, MOP
    - Test: 0,5h durch TMS, MOP

◯ **R4** Als Manager kann ich eine visuelle Rolle löschen, damit Nutzer sich dieser nicht mehr zuordnen können.

  ◯ **R4.F1** Erweitere R2.F1, sodass angezeigte visuelle Rollen gelöscht werden können.

- ◯ **R4.F2** Erstelle Frontend-Funktionalität zum Löschen visueller Rollen, welche, wenn ein Manager mit gültiger Session sie nutzt, R4.F1 ausliest und R4.B1 aufruft.
- ◯ **R4.B1** Erstelle Backend-Funktionalität zum Löschen visueller Rollen, welche R4.D1 aufruft.
- ◯ **R4.D1** Erstelle Daten-Funktionalität zum Löschen visueller Rollen, welche eine visuelle Rolle löscht.
  - ∗ Aufwandsschätzung: 0,25h
  - ∗ Bearbeitung: 0,25h durch TMS, MOP
  - ∗ Test: 0,5h durch TMS, MOP

◯ **R5** Als Nutzer kann ich mich auf meinem Profil einer visuellen Rolle zuordnen, damit diese auf meinem Profil sichtbar ist.

- ◯ **R5.F1** Erweitere Anzeige für Profile um eine visuelle Rolle und eine Auswahl visueller Rollen.
  - ◯ **R5.F2** Erweitere Frontend-Funktionalität zum Anzeigen eines Profils um eine visuelle Rolle und die visuellen Rollen zur rechtebezogenen Rolle.
  - ◯ **R5.F3** Erweitere Frontend-Funktionalität zum Anpassen des Profils, sodass diese die ausgewählte visuelle Rolle von R5.F1 ausliest und R5.B2 aufruft.
  - ◯ **R5.B1** Erweitere Backend-Funktionalität zur Ausgabe eines Profils, sodass diese die visuelle Rolle des Nutzers ausgibt, R2.D2 mit der rechtebezogenen Rolle des Nutzers aufruft und auch die davon erhaltenen visuellen Rollen ausgibt.
  - ◯ **R5.B2** Erweitere Backend-Funktionalität zum Anpassen des Profils, sodass diese R5.D2 aufrufen kann.
  - ◯ **R5.D1** Erweitere Speicher für Nutzer um eine visuelle Rolle.
    - · Aufwandsschätzung: 0,25h
    - · Bearbeitung: 0,25h durch TMS, MOP
    - · Test: 0,5h durch TMS, MOP
  - ◯ **R5.D2** Erstelle Daten-Funktionalität zum Anpassen der visuellen Rolle eines Nutzers.
    - · Aufwandsschätzung: 5h
    - · Bearbeitung: 4h durch TMS, MOP
    - · Test: 0,5h durch TMS, MOP

1.2.2 [P] Profil

⊗ **P4** Als Nutzer kann ich mir das Profil eines Nutzers anzeigen lassen, um die Angaben des Nutzers auf seinem Profil zu sehen.

- ⊗ **P4.F1** Erstelle Anzeige für Nutzer, welche alle Nutzer anzeigt und die Auswahl eines Nutzers erlaubt.
  - ∗ Aufwandsschätzung: 1h durch PK
  - ∗ Bearbeitung: 1h durch PK
  - ∗ Review: 0,25h durch ZSh
  - ∗ Test: 0,25h durch ZSH
- ⊗ **P4.F2** Erstelle Frontend-Funktionalität zum Anzeigen von P4.F1, welche, wenn ein Nutzer mit gültiger Session sie nutzt, P4.B1 aufruft und P4.F1 anzeigt.
  - ∗ Aufwandsschätzung: 0,25h durch PK
  - ∗ Bearbeitung: 0,25h durch PK, LK
  - ∗ Review: 0,25h durch ZSH

- Test: 0,25h durch ZSH

⊗ **P4.F3** Erweitere Frontend-Funktionalität zum Anzeigen eines Profils, sodass diese einen Nutzer annimmt und dessen Profil anzeigt.

- Aufwandsschätzung: 1h durch PK
- Bearbeitung: 0,25h durch PK, LK
- Review: 0,25h durch ZSH
- Test: 0,25h durch ZSH

⊗ **P4.B1** Erstelle Backend-Funktionalität zum Ausgeben aller Nutzer, welche P4.D1 aufruft und die Nutzer übergibt.

- Aufwandsschätzung: 0,5h durch LK
- Bearbeitung: 0,5h durch LK
- Test: 1,5h durch LK

⊗ **P4.D1** Erstelle Daten-Funktionalität zur Ausgabe aller Nutzer.

- Aufwandsschätzung: 0,25h
- Bearbeitung: 0,25h durch TMS, MOP
- Test: 0,25h durch TMS, MOP

1.2.3 [U] User-Story

U2.U**U2** Als Nutzer kann ich eine User-Story auswählen, um mir alle zu ihr gespeicherten Informationen anzeigen lassen.

⊗ ⊗ **U2.F1** Erstelle Anzeige für eine User-Story, welche die Auswahl einer User-Story erlaubt und alle zu einer User-Story gespeicherten Informationen anzeigen kann.

- Aufwandsschätzung: 1h
- Bearbeitung: 1h
- Review: 0,25h durch PK
- Test: 0,25h durch PK

⊗ **U2.F2** Erstelle Frontend-Funktionalität zum Anzeigen einer User-Story, welche, wenn ein Nutzer mit gültiger Session die ausgewählte User-Story ändert, U2.F1 ausliest, U2.B1 aufruft und U2.F1 anzeigt.

- Aufwandsschätzung: 0,5h durch ZSH
- Bearbeitung: 0,5h durch ZSH
- Review: 0,25h durch PK
- Test: 0,25h durch PK

⊗ **U2.F3** Verändere Frontend-Funktionalität zum Anpassen einer User-Story, sodass die Anpassung mittels U2.F1 stattfindet.

- Aufwandsschätzung: 0,5h durch PK
- Bearbeitung: 0,5h durch PK
- Review: 0,25h durch LK
- Test: 0,25h durch PK

⊗ **U2.F4** Verändere Frontend-Funktionalität zum Erstellen einer User-Story, sodass die Erstellung mittels U2.F1 stattfindet.

- Aufwandsschätzung: 0,5h durch LK
- Bearbeitung: 0,5h durch LK
- Review: 0,25h durch PK

    - Test: 0,25h durch PK
- ⊗ **U2.B1** Erstelle Backend-Funktionalität zum Ausgeben einer User-Story, welche U2.D1 aufruft und eine User-Story ausgibt.
    - Aufwandsschätzung: 0,25h
    - Bearbeitung: 0,25h durch LK
    - Review: [time in 0,25h increments] durch [Names]
    - Test: 0,25h durch LK
- ⊗ **U2.D1** Erstelle Daten-Funktionalität zum Ausgeben einer User-Story.
    - Aufwandsschätzung: 0,25h
    - Bearbeitung: 0,25h durch TMS, MOP
    - Test: 0,25h durch TMS, MOP

1.2.4 [T] Task

- ⊗ **T2** Als Nutzer kann ich eine Task auswählen, um mir alle zu ihr gespeicherten Informationen anzeigen zu lassen.
    - ⊗ **T2.F1** Erstelle Anzeige für eine Task, welche die Auswahl einer Task erlaubt und alle zu einer Task gespeicherten Informationen anzeigen kann.
        - Aufwandsschätzung: 1h durch ZSH
        - Bearbeitung: 1h durch ZSH
        - Review: 0,25h durch PK
        - Test: 0,25h durch PK
    - ⊗ **T2.F2** Erstelle Frontend-Funktionalität zum Anzeigen einer Task, welche, wenn ein Nutzer mit gültiger Session die ausgewählte Task ändert, T2.F1 ausliest, T2.B1 aufruft und T2.F1 anzeigt.
        - Aufwandsschätzung: 0,5h durch ZSH
        - Bearbeitung: 2h durch ZSH
        - Review: 0,25h durch PK
        - Test: 0,25h durch PK
    - ⊗ **T2.F3** Verändere Frontend-Funktionalität zum Anpassen einer Task, sodass die Anpassung mittels T2.F1 stattfindet.
        - Aufwandsschätzung: 0,5h durch PK, ZSH
        - Bearbeitung: 0,5 durch PK, ZSH
        - Review: 0,25h durch PK, 0,25h durch ZSH
        - Test: 0,25h durch PK, 0,25h durch ZSH
    - ⊗ **T2.F4** Verändere Frontend-Funktionalität zum Erstellen einer Task, sodass die Erstellung mittels T2.F1 stattfindet.
        - Aufwandsschätzung: 0,5h durch PK, ZSH
        - Bearbeitung: 0,5 durch PK, ZSH
        - Review: 0,25h durch PK, 0,25h durch ZSH
        - Test: 0,25h durch PK, 0,25h durch ZSH
    - ⊗ **T2.B1** Erstelle Backend-Funktionalität zum Ausgeben einer Task, welche T2.D1 aufruft und eine Task ausgibt.
        - Aufwandsschätzung: 4h
        - Bearbeitung: 5h durch LK

    - Test: 0,5h durch LK
  - ⊗ **T2.D1** Erstelle Daten-Funktionalität zum Ausgeben einer Task.
    - Aufwandsschätzung: 0,25h
    - Bearbeitung: 0,25h durch TMS, MOP
    - Test: 0,5h durch TMS, MOP

- ⊗ **T6** Als Nutzer kann ich eine Task als fertig markieren, um anzuzeigen, dass diese Task vollständig erfüllt ist.
  - ⊗ **T6.F1** Erweitere T2.F1 um eine Markierung als fertig.
    - Aufwandsschätzung: 1h durch PK
    - Bearbeitung: 1h durch PK
    - Review: 0,25h durch ZSH
    - Test: 0,25h durch ZSH
  - ⊗ **T6.F2** Erweitere Frontend-Funktionalität zum Anpassen einer Task, sodass, wenn die Task auf ihrem Task-Board in die Task-Liste „fertiggestellte Tasks“ geschoben wird, sie als fertig markiert wird.
    - Aufwandsschätzung: 1h durch PK
    - Bearbeitung: 1h durch PK
    - Review: 0,25h durch ZSH
    - Test: 0,25h durch ZSH
  - ⊗ **T6.F3** Erweitere Frontend-Funktionalität zum Anpassen einer Task, sodass, wenn die Task auf ihrem Task-Board aus der Task-Liste „fertiggestellte Tasks“ geschoben wird, sie als unfertig markiert wird.
    - Aufwandsschätzung: 1h durch PK
    - Bearbeitung: 1h durch PK
    - Review: 0,25h durch ZSH
    - Test: 0,25h durch ZSH
  - ⊗ **T6.F4** Erweitere Frontend-Funktionalität zum Anzeigen aller Tasks, sodass als fertig markierte Tasks auch eine visuelle Markierung haben.
    - Aufwandsschätzung: 0,5h durch PK, ZSH
    - Bearbeitung: 0,25h durch PK, ZSH
    - Review: 0,25h durch PK, 0,25h durch ZSH
    - Test: 0,25h durch PK, 0,25h durch ZSH

- ⊗ **T7** Als Nutzer kann ich einer Task eine Priorität (Urgent > High > Normal > Low) zuordnen, um die Dringlichkeit der Task auszudrücken.
  - ⊗ **T7.F1** Erweitere T2.F1 um eine Priorität.
    - Aufwandsschätzung: 0,5h durch ZSH
    - Bearbeitung: 0,5h durch ZSH
    - Review: 0,25h durch PK
    - Test: 0,25h durch PK
  - ⊗ **T7.F2** Erweitere Frontend-Funktionalität zum Anpassen einer Task um die Priorität.
    - Aufwandsschätzung: 0,5h durch ZSH
    - Bearbeitung: 0,5h durch ZSH
    - Review: 0,25h durch PK

    - Test: 0,25h durch PK
- ⊗ **T7.F3** Erweitere Frontend-Funktionalität zum Anzeigen aller Tasks, sodass die Priorität einer Task auch angezeigt wird.
    - Aufwandsschätzung: 0,5h durch ZSH
    - Bearbeitung: 0,5h durch ZSH
    - Review: 0,25h durch PK
    - Test: 0,25h durch PK
- ⊗ **T7.F4** Erweitere Frontend-Funktionalität zum Erstellen einer Task um die Priorität.
    - Aufwandsschätzung: 0,5h durch ZSH
    - Bearbeitung: 0,5h durch ZSH
    - Review: 0,25h durch PK
    - Test: 0,25h durch PK
- ⊗ **T7.B1** Erweitere T2.B1 um die Priorität.
    - Aufwandsschätzung: 0,5h durch HLF
    - Bearbeitung: 0,5h durch HLF
    - Test: 0,25h durch HLF
- ⊗ **T7.B2** Erweitere Backend-Funktionalität zum Anpassen einer Task, sodass diese T7.D2 aufrufen kann.
    - Aufwandsschätzung: 0,5h durch HLF
    - Bearbeitung: 0,75h durch HLF
    - Test: 0,5h durch HLF
- ⊗ **T7.B3** Erweitere Backend-Funktionalität zum Ausgeben aller Tasks, sodass auch die Prioritäten von Tasks ausgegeben werden.
    - Aufwandsschätzung: 0,5h durch HLF
    - Bearbeitung: 0,5h durch HLF
    - Test: 0,25h durch HLF
- ⊗ **T7.B4** Erweitere Backend-Funktionalität zum Erstellen einer Task um die Priorität.
    - Aufwandsschätzung: 0,5h durch HLF
    - Bearbeitung: 0,75h durch HLF
    - Test: 0,5h durch HLF
- ⊗ **T7.D1** Erweitere Speicher für Tasks um eine Priorität.
    - Aufwandsschätzung: 0,25h
    - Bearbeitung: 0,25h durch TMS, MOP
    - Test: 0,5h durch TMS, MOP
- ⊗ **T7.D2** Erstelle Daten-Funktionalität zum Anpassen einer Task um die Priorität.
    - Aufwandsschätzung: 0,25h
    - Bearbeitung: 0,25h durch TMS, MOP
    - Test: 0,5h durch TMS, MOP
- ⊗ **T7.D3** Erweitere Daten-Funktionalität zum Erstellen einer Task um die Priorität.
    - Aufwandsschätzung: 0,5h
    - Bearbeitung: 0,5h durch TMS, MOP
    - Test: 0,5h durch TMS, MOP

⊗ **T8** Als Nutzer kann ich einer Task Nutzer zuordnen, um auszudrücken, dass die Task von diesen Nutzern erledigt wird.

⊗ **T8.F1** Erweitere T2.F1 um eine Auswahl von Nutzern.
  * Aufwandsschätzung: 0,5h durch PK
  * Bearbeitung: 0,5h durch PK
  * Review: 0,25h durch ZSH
  * Test: 0,25h durch ZSH

⊗ **T8.F2** Erweitere Frontend-Funktionalität zum Anpassen einer Task um die Liste zugeordneter Nutzer.
  * Aufwandsschätzung: 0,5h durch PK
  * Bearbeitung: 0,5h durch PK
  * Review: 0,25h durch ZSH
  * Test: 0,25h durch ZSH

⊗ **T8.F3** Erweitere Frontend-Funktionalität zum Erstellen einer Task um die Liste zugeordneter Nutzer.
  * Aufwandsschätzung: 0,5h durch PK
  * Bearbeitung: 0,5h durch PK
  * Review: 0,25h durch ZSH
  * Test: 0,25h durch ZSH

⊗ **T8.B1** Erweitere T2.B1 um eine Liste aller Nutzer und um die Liste zugeordneter Nutzer.
  * Aufwandsschätzung: 1h
  * Bearbeitung: 1h durch LK
  * Test: 0.75h durch HLF

⊗ **T8.B2** Erweitere Backend-Funktionalität zum Anpassen einer Task, sodass diese T8.D2 aufrufen kann.
  * Aufwandsschätzung: 2h
  * Bearbeitung: 2h durch LK

⊗ **T8.B3** Erweitere Backend-Funktionalität zum Erstellen einer Task um die Liste zugeordneter Nutzer.
  * Aufwandsschätzung: 2h
  * Bearbeitung: 3h durch LK
  * Test: 0,5h durch LK

⊗ **T8.D1** Erweitere Speicher für Tasks um eine Liste zugeordneter Nutzer.
  * Aufwandsschätzung: 0,5h
  * Bearbeitung: 0,5h durch TMS, MOP
  * Test: 0,5h durch TMS, MOP

⊗ **T8.D2** Erstelle Daten-Funktionalität zum Anpassen der Liste zugeordneter Nutzer einer Task.
  * Aufwandsschätzung: 1h
  * Bearbeitung: 0,75h durch TMS, MOP
  * Test: 0,5h durch TMS, MOP

⊗ **T8.D3** Erweitere Daten-Funktionalität zum Erstellen einer Task um die Liste zugeordneter Nutzer.
  * Aufwandsschätzung: 0,5h
  * Bearbeitung: 0,5h durch TMS, MOP
  * Test: 0,5h durch TMS, MOP

⊗ **T9** Als Nutzer kann ich eine Task mit einer Abgabefrist versehen, um zu verdeutlichen, wann die Ergebnisse der Task benötigt werden.

- ⊗ **T9.F1** Erweitere T2.F1 um eine Abgabefrist.
  - Aufwandsschätzung: 0,5h durch ZSH
  - Bearbeitung: 0,5h durch ZSH
  - Review: 0,25h durch PK
  - Test: 0,25h durch PK
- ⊗ **T9.F2** Erweitere Frontend-Funktionalität zum Anpassen einer Task um die Abgabefrist.
  - Aufwandsschätzung: 0,5h durch ZSH
  - Bearbeitung: 0,5h durch ZSH
  - Review: 0,25h durch PK
  - Test: 0,25h durch PK
- ⊗ **T9.F3** Erweitere Frontend-Funktionalität zum Erstellen einer Task um die Abgabefrist.
  - Aufwandsschätzung: 0,5h durch ZSH
  - Bearbeitung: 0,5h durch ZSH
  - Review: 0,25h durch PK
  - Test: 0,25h durch PK
- ⊗ **T9.B1** Erweitere T2.B1 um die Abgabefrist.
  - Aufwandsschätzung: 0,25h durch HLF
  - Bearbeitung: 0,5h durch HLF
  - Test: 0,25h durch HLF
- ⊗ **T9.B2** Erweitere Backend-Funktionalität zum Anpassen einer Task, sodass diese T9.D2 aufrufen kann.
  - Aufwandsschätzung: 0,5h durch HLF
  - Bearbeitung: 0,5h durch HLF
  - Test: 0,5h durch HLF
- ⊗ **T9.B3** Erweitere Backend-Funktionalität zum Erstellen einer Task um die Abgabefrist.
  - Aufwandsschätzung: 0,25h durch HLF
  - Bearbeitung: 0,5h durch HLF
  - Test: 0,5h durch HLF
- ⊗ **T9.D1** Erweitere Speicher für Tasks um eine Abgabefrist.
  - Aufwandsschätzung: 0,25h
  - Bearbeitung: 0,5h durch TMS, MOP
  - Test: 0,5h durch TMS, MOP
- ⊗ **T9.D2** Erstelle Daten-Funktionalität zum Anpassen der Abgabefrist einer Task.
  - Aufwandsschätzung: 0,5h
  - Bearbeitung: 0,5h durch TMS, MOP
  - Test: 0,5h durch TMS, MOP
- ⊗ **T9.D3** Erweitere Daten-Funktionalität zum Erstellen einer Task um die Abgabefrist.
  - Aufwandsschätzung: 0,5h
  - Bearbeitung: 0,5h durch TMS, MOP
  - Test: 0,5h durch TMS, MOP

⊗ **T10** Als Nutzer kann ich eine Task mit einer Aufwandsschätzung versehen, um auszudrücken, wie lange die Bearbeitung der Task schätzungsweise dauert.

- ⊗ **T10.F1** Erweitere T2.F1 um eine Aufwandsschätzung.
  - Aufwandsschätzung: 0,5h durch PK
  - Bearbeitung: 0,5h durch PK
  - Review: 0,25h durch ZSH
  - Test: 0,25h durch ZSH
- ⊗ **T10.F2** Erweitere Frontend-Funktionalität zum Anpassen einer Task um die Aufwandsschätzung.
  - Aufwandsschätzung: 0,5h durch PK
  - Bearbeitung: 0,25h durch PK
  - Review: 0,25h durch ZSH
  - Test: 0,25h durch ZSH
- ⊗ **T10.F3** Erweitere Frontend-Funktionalität zum Erstellen einer Task um die Aufwandsschätzung.
  - Aufwandsschätzung: 0,5h durch PK
  - Bearbeitung: 0,25h durch PK
  - Review: 0,25h durch ZSH
  - Test: 0,25h durch ZSH
- ⊗ **T10.B1** Erweitere T2.B1 um die Aufwandsschätzung.
  - Aufwandsschätzung: 0,5h durch HLF
  - Bearbeitung: 0,75h durch HLF
  - Test: 0,25h durch HLF
- ⊗ **T10.B2** Erweitere Backend-Funktionalität zum Anpassen einer Task, sodass diese T10.D2 aufrufen kann.
  - Aufwandsschätzung: 0,5h durch HLF
  - Bearbeitung: 0,5h durch HLF
  - Test: 0,5h durch HLF
- ⊗ **T10.B3** Erweitere Backend-Funktionalität zum Erstellen einer Task um die Aufwandsschätzung.
  - Aufwandsschätzung: 0,5h durch HLF
  - Bearbeitung: 0,5h durch HLF
  - Test: 0,5h durch HLF
- ⊗ **T10.D1** Erweitere Speicher für Tasks um eine Aufwandsschätzung.
  - Aufwandsschätzung: 0,25h
  - Bearbeitung: 0,5h durch TMS, MOP
  - Test: 0,5h durch TMS, MOP
- ⊗ **T10.D2** Erstelle Daten-Funktionalität zum Anpassen der Aufwandsschätzung einer Task.
  - Aufwandsschätzung: 0,5h
  - Bearbeitung: 0,5h durch TMS, MOP
  - Test: 0,5h durch TMS, MOP
- ⊗ **T10.D3** Erweitere Daten-Funktionalität zum Erstellen einer Task um die Aufwandsschätzung.
  - Aufwandsschätzung: 0,5h
  - Bearbeitung: 0,5h durch TMS, MOP
  - Test: 0,5h durch TMS, MOP

⊗ **T11** Als Nutzer kann ich Tasks nach deren Fertigstellung mit einer Bearbeitungszeit versehen, um auszudrücken, wie lange die Bearbeitung der Task dauerte.

- ⊗ **T11.F1** Erweitere T2.F1 um eine Bearbeitungszeit, die, falls die Task als fertig markiert ist, angezeigt wird.
  - Aufwandsschätzung: 0,25h durch PK
  - Bearbeitung: 0,25h durch PK
  - Review: 0,25h durch ZSH
  - Test: 0,25h durch ZSH
- ⊗ **T11.F2** Erweitere Frontend-Funktionalität zum Anpassen einer Task um die Bearbeitungszeit.
  - Aufwandsschätzung: 0,25h durch PK
  - Bearbeitung: 0,25h durch PK
  - Review: 0,25h durch ZSH
  - Test: 0,25h durch ZSH
- ⊗ **T11.B1** Erweitere T2.B1 um die Bearbeitungszeit.
  - Aufwandsschätzung: 0,5h
  - Bearbeitung: 0,5h durch HvR
- ⊗ **T11.B2** Erweitere Backend-Funktionalität zum Anpassen einer Task, sodass diese, falls die anzupassende Task als fertig markiert ist, T11.D2 aufrufen kann.
  - Aufwandsschätzung: 0,5h
  - Bearbeitung: 0,5h durch HvR
- ⊗ **T11.D1** Erweitere Speicher für Tasks um eine Bearbeitungszeit.
  - Aufwandsschätzung: 0,25h
  - Bearbeitung: 0,5h durch TMS, MOP
  - Test: 0,5h durch TMS, MOP
- ⊗ **T11.D2** Erstelle Daten-Funktionalität zum Anpassen der Bearbeitungszeit einer Task.
  - Aufwandsschätzung: 0,5h
  - Bearbeitung: 0,5h durch TMS, MOP
  - Test: 0,5h durch TMS, MOP

⊗ **T16** Als Nutzer kann ich einem Task-Board Tasks zuordnen, um anzugeben, dass diese auf diesem Task-Board zu bearbeiten sind.

- ⊗ **T16.F1** Erweitere T2.F1 um eine Zuordnung zu einem Task-Board.
  - Aufwandsschätzung: 0,25h durch PK
  - Bearbeitung: 0,25h durch PK
  - Review: 0,25h durch ZSH
  - Test: 0,25h durch ZSH
- ⊗ **T16.F2** Erweitere Frontend-Funktionalität zum Anpassen einer Task um die Zuordnung zu einem Task-Board.
  - Aufwandsschätzung: 0,25h durch PK
  - Bearbeitung: 0,25h durch PK
  - Review: 0,25h durch ZSH
  - Test: 0,25h durch ZSH

- ⊗ **T16.F3** Erweitere Frontend-Funktionalität zum Erstellen einer Task um die Zuordnung zu einer Task-Liste.
  - Aufwandsschätzung: 0,25h durch PK
  - Bearbeitung: 0,25h durch PK
  - Review: 0,25h durch ZSH
  - Test: 0,25h durch ZSH
- ⊗ **T16.B1** Erweitere T2.B1 um die Zuordnung zu einem Task-Board.
  - Aufwandsschätzung: 1h
  - Bearbeitung: 1h durch LK
  - Test: 1h durch LK
- ⊗ **T16.B2** Erweitere Backend-Funktionalität zum Anpassen einer Task, sodass diese T16.D1 aufrufen kann.
  - Aufwandsschätzung: 3h
  - Bearbeitung: 3h durch LK
  - Test: 1.5h durch LK
- ⊗ **T16.B3** Erweitere Backend-Funktionalität zum Erstellen einer Task, um die Zuordnung zu einer Task-Liste.
  - Aufwandsschätzung: 2h
  - Bearbeitung: 2h durch LK
  - Test: 0,5h durch LK
- ⊗ **T16.D1** Erstelle Funktionalität zum Anpassen der Zuordnung zu einem Task-Board einer Task, welche zu einer Task die Task-Liste „freie Tasks“ des Task-Boards als Zuordnung zu einer Task-Liste speichert.
  - Aufwandsschätzung: 8h
  - Bearbeitung: 5h durch TMS, MOP
  - Test: 0,5h durch TMS, MOP
- ⊗ **T16.D2** Erweitere Daten-Funktionalität zum Erstellen einer Task um die Zuordnung zu einer Task-Liste.
  - Aufwandsschätzung: 0,5h
  - Bearbeitung: 0,5h durch TMS, MOP
  - Test: 0,5h durch TMS, MOP

⊗ **T17** Als Nutzer kann ich zu einer fertigen Task eine Visualisierung einsehen, welche darstellt, wie sich die Aufwandsschätzung und die Bearbeitungszeit unterscheiden, um die Aufwandsschätzung bewerten zu können.

- ⊗ **T17.F1** Erweitere T2.F1 um eine Visualisierung, die für eine fertige Task mit Aufwandsschätzung und Bearbeitungszeit angezeigt wird und darstellt, wie sich die Aufwandsschätzung und die Bearbeitungszeit unterscheiden.
  - Aufwandsschätzung: 1h durch ZSH
  - Bearbeitung: 2h durch ZSH

1.2.5 [TB] Task-Board

⊗ **TB2** Als Nutzer kann ich ein Task-Board mit allen seinen Task-Listen aufrufen, um es zu verwalten.

- ⊗ **TB2.F1** Erstelle Anzeige für Task-Boards, welche zur Auswahl eines Task-Boards und dem Anzeigen der Task-Listen des Task-Boards mitsamt der zugeordneten Tasks dient.
  - Aufwandsschätzung: 0,5h durch ZSH
  - Bearbeitung: 0,5h durch ZSH
  - Review: 0,25h durch PK, ZSH
  - Test: 0,25h durch PK, ZSH
- ⊗ **TB2.F2** Erstelle Frontend-Funktionalität zum Anzeigen eines Task-Boards, welche, wenn ein Nutzer mit gültiger Session sie nutzt, TB2.F1 ausliest, TB2.B1 aufruft und TB2.F1 anzeigt.
  - Aufwandsschätzung: 0,5h durch ZSH
  - Bearbeitung: 0,5h durch ZSH
  - Review: 0,25h durch PK, ZSH
  - Test: 0,25h durch PK, ZSH
- ⊗ **TB2.B1** Erstelle Backend-Funktionalität zum Ausgeben eines Task-Boards, welche TB2.D1 aufruft und die Task-Listen und Tasks ausgibt.
  - Aufwandsschätzung: 1,5h
  - Bearbeitung: 1h durch LK
  - Test: 0,5 durch LK
- ⊗ **TB2.D1** Erstelle Daten-Funktionalität zum Ausgeben eines Task-Boards, welche zu einem Task-Board die Task-Listen mit ihren Tasks ausgibt.
  - Aufwandsschätzung: 3h
  - Bearbeitung: 4,5h durch TMS, MOP
  - Test: 0,5h durch TMS, MOP

⊗ **TB3** Als Nutzer kann ich auf einem Task-Board Tasks zwischen den angezeigten Task-Listen verschieben, um die Prozesse rund um die Bearbeitung der Tasks visuell anschaulich durchführen zu können.

- ⊗ **TB3.F1** Erstelle Frontend-Funktionalität zum Verschieben einer Task, welche, wenn ein Nutzer mit gültiger Session sie nutzt, es erlaubt, Tasks zwischen den Task-Listen eines Task-Boards zu verschieben, TB2.F1 ausliest und TB3.B1 aufruft.
  - Aufwandsschätzung: 0,5h durch ZSH
  - Bearbeitung: 0,5h durch ZSH
  - Review: 0,25h durch PK, ZSH
  - Test: 0,25h durch PK, ZSH
- ⊗ **TB3.B1** Erweitere Backend-Funktionalität zum Anpassen einer Task, sodass diese TB3.D1 aufrufen kann.
  - Aufwandsschätzung: 2,5h
  - Bearbeitung: 4h durch LK
  - Test: 1h durch LK
- ⊗ **TB3.D1** Erstelle Daten-Funktionalität zum Anpassen der Zuordnung zu einer Task-Liste einer Task.
  - Aufwandsschätzung: 0,5h
  - Bearbeitung: 0,5h durch TMS, MOP
  - Test: 0,5h durch TMS, MOP

◯ **TB5** Als Nutzer kann ich auf einem Task-Board einsehen, welcher Task welche Nutzer zugeordnet sind, um die Verteilung von Tasks zu erleichtern.

- ◯ **TB5.F1** Erweitere TB2.F1, sodass zu jeder Task die Liste zugeordneter Nutzer eingesehen werden kann.
  - Aufwandsschätzung: 2h durch PK
  - Bearbeitung: 1h durch PK
  - Review: 0,25h durch PK, ZSH
  - Test: 0,25h durch PK, ZSH
- ⊗ **TB5.B1** Erweitere Tasks von TB2.B1, um die Liste zugeordneter Nutzer.
  - Aufwandsschätzung: 0,25h
  - Bearbeitung: 0,25h durch LK
  - Test: 0,25h durch LK

⊗ **TB8** Als Nutzer kann ich auf einem Task-Board zu der Menge an fertigen Tasks des Task-Boards eine Visualisierung einsehen, welche darstellt, wie sich die Aufwandsschätzungen und die Bearbeitungszeiten der Tasks unterscheiden, um die Aufwandsschätzungen bewerten zu können.

- ⊗ **TB8.F1** Erweitere TB2.F1 um eine Visualisierung, die alle fertigen Tasks mit Aufwandsschätzung und Bearbeitungszeit einbezieht und darstellt, wie sich insgesamt die Aufwandsschätzungen und die Bearbeitungszeiten unterscheiden.
  - Aufwandsschätzung: 0,5h durch ZSH
  - Bearbeitung: 0,5h durch ZSH
  - Review: 0,5h durch ZSH, PK
  - Test: 0,5h durch ZSH, PK
- ⊗ **TB8.B1** Erweitere Tasks von TB2.B1 um die Aufwandsschätzung, die Markierung als fertig und die Bearbeitungszeit.
  - Aufwandsschätzung: 1,5h
  - Bearbeitung: 1,5h durch LK
  - Review: 0,25h durch PK, ZSH
  - Test: 0,25h durch LK

⊗ **TB9** Als Product Owner kann ich ein Task-Board mit den Task-Listen „freie Tasks“, „Tasks in Bearbeitung“, „Tasks unter Review“, „Tasks unter Test“und „fertiggestellte Tasks“ erstellen, um Tasks strukturiert zu bearbeiten.

- ⊗ **TB9.F1** Erstelle Frontend-Funktionalität zum Erstellen eines Task-Boards, welche, wenn ein Product Owner mit gültiger Session sie nutzt, TB9.B1 aufruft.
  - Aufwandsschätzung: 0,25h durch PK, ZSH
  - Bearbeitung: 0,5 durch PK, ZSh
  - Review: 0,25h durch PK, ZSH
  - Test: 0,25h durch PK, ZSH
- ⊗ **TB9.B1** Erstelle Backend-Funktionalität zum Erstellen eines Task-Boards, welche TB9.D3 aufruft.
  - Aufwandsschätzung: 1,5h
  - Bearbeitung: 1h durch LK
  - Test: 0,25h durch LK
- ⊗ **TB9.D1** Erstelle Speicher für Task-Boards.
  - Aufwandsschätzung: 0,25h
  - Bearbeitung: 0,25h durch TMS, MOP

    - Test: 0,25h durch TMS, MOP
  - ⊗ **TB9.D2** Erstelle Speicher für Task-Listen, welcher zu einer Task-Liste eine Zuordnung zu einem Task-Board speichert.
    - Aufwandsschätzung: 0,25h
    - Bearbeitung: 0,25h durch TMS, MOP
    - Test: 0,5h durch TMS, MOP
  - ⊗ **TB9.D3** Erstelle Daten-Funktionalität zum Erstellen eines Task-Boards, welche ein Task-Board in TB9.D1 und die zum Task-Board zugeordneten Task-Listen „freie Tasks“, „Tasks in Bearbeitung“, „Tasks unter Review“, „Tasks unter Test“ und „fertiggestellte Tasks“ in TB9.D2 speichert.
    - Aufwandsschätzung: 8h
    - Bearbeitung: 6,5h durch TMS, MOP
    - Test: 0,5h durch TMS, MOP
  - ⊗ **TB9.D4** Erweitere Speicher für Tasks um eine Zuordnung zu einer Task-Liste.
    - Aufwandsschätzung: 0,25h
    - Bearbeitung: 0,25h durch TMS, MOP
    - Test: 0,5h durch TMS, MOP

- ⊗ **TB10** Als Product Owner kann ich ein Task-Board umbenennen, um es anzupassen.
  - ⊗ **TB10.F1** Erstelle Frontend-Funktionalität zum Umbenennen eines Task-Boards, welche, wenn ein Product Owner mit gültiger Session sie nutzt, TB2.F1 ausliest und TB10.B1 aufruft.
    - Aufwandsschätzung: 0,5h durch PK
    - Bearbeitung: 1h durch PK
    - Review: 0,25h durch ZSH
    - Test: 0,25h durch ZSH
  - ⊗ **TB10.B1** Erstelle Backend-Funktionalität zum Umbenennen eines Task-Boards, welche TB10.D1 aufruft.
    - Aufwandsschätzung: 1,25h
    - Bearbeitung: 1,25h durch LK
    - Test: 0,25h durch LK
  - ⊗ **TB10.D1** Erstelle Daten-Funktionalität zum Umbenennen eines Task-Boards.
    - Aufwandsschätzung: 0,25h
    - Bearbeitung: 0,25h durch TMS, MOP
    - Test: 0,5h durch TMS, MOP

- ⊗ **TB11** Als Product Owner kann ich ein Task-Board löschen, um es aus dem System zu entfernen.
  - ⊗ **TB11.F1** Erstelle Frontend-Funktionalität zum Löschen eines Task-Boards, welche, wenn ein Product Owner mit gültiger Session sie nutzt, TB2.F1 ausliest und TB11.B1 aufruft.
    - Aufwandsschätzung: 0,5h durch ZSH, PK
    - Bearbeitung: 0,5h durch ZSH, PK
    - Review: 0,25h durch ZSH, PK
    - Test: 0,25h durch ZSH, PK
  - ⊗ **TB11.B1** Erstelle Backend-Funktionalität zum Löschen eines Task-Boards, welche zu jeweils jeder Task jeweils jeder Task-Liste des Task-Boards TB3.D1 (mit Default-Wert) aufruft und dann TB11.D2 und TB11.D1 aufruft.

- Aufwandsschätzung: 0,75h
- Bearbeitung: 0,5h durch LK
- Test: 0,25h durch LK

⊗ **TB11.D1** Erstelle Daten-Funktionalität zum Löschen eines Task-Boards.

- Aufwandsschätzung: 0,75h
- Bearbeitung: 0,75h durch TMS, MOP
- Test: 0,5h durch TMS, MOP

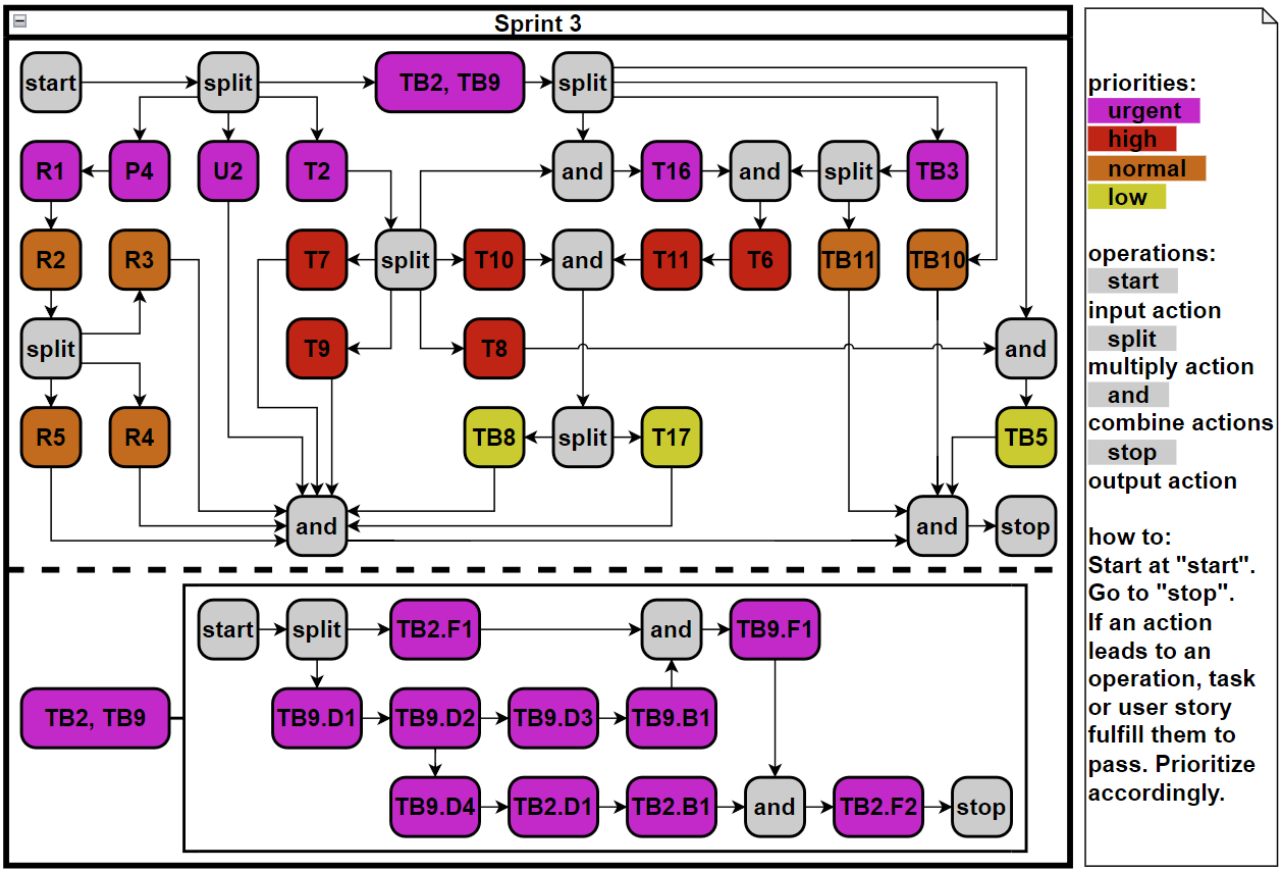
⊗ **TB11.D2** Erstelle Daten-Funktionalität zum Löschen der zu einem Task-Board verlinkten Task-Listen.

- Aufwandsschätzung: 0,25h
- Bearbeitung: 0,25h durch TMS, MOP
- Test: 0,25h durch TMS, MOP

Zusatzlicher Aufwand:

- Refactoring Frontend + Designänderungen: 28h durch ZSH

1.2.6 Ablaufplan zu User-Storys und Tasks



2 Feinentwurf

2.1 Komponentendiagramm

Die User-Storys R2, R3, R4 und R5 wurden in diesem Sprint nicht implementiert und sind deshalb nicht im Komponentendiagramm enthalten.

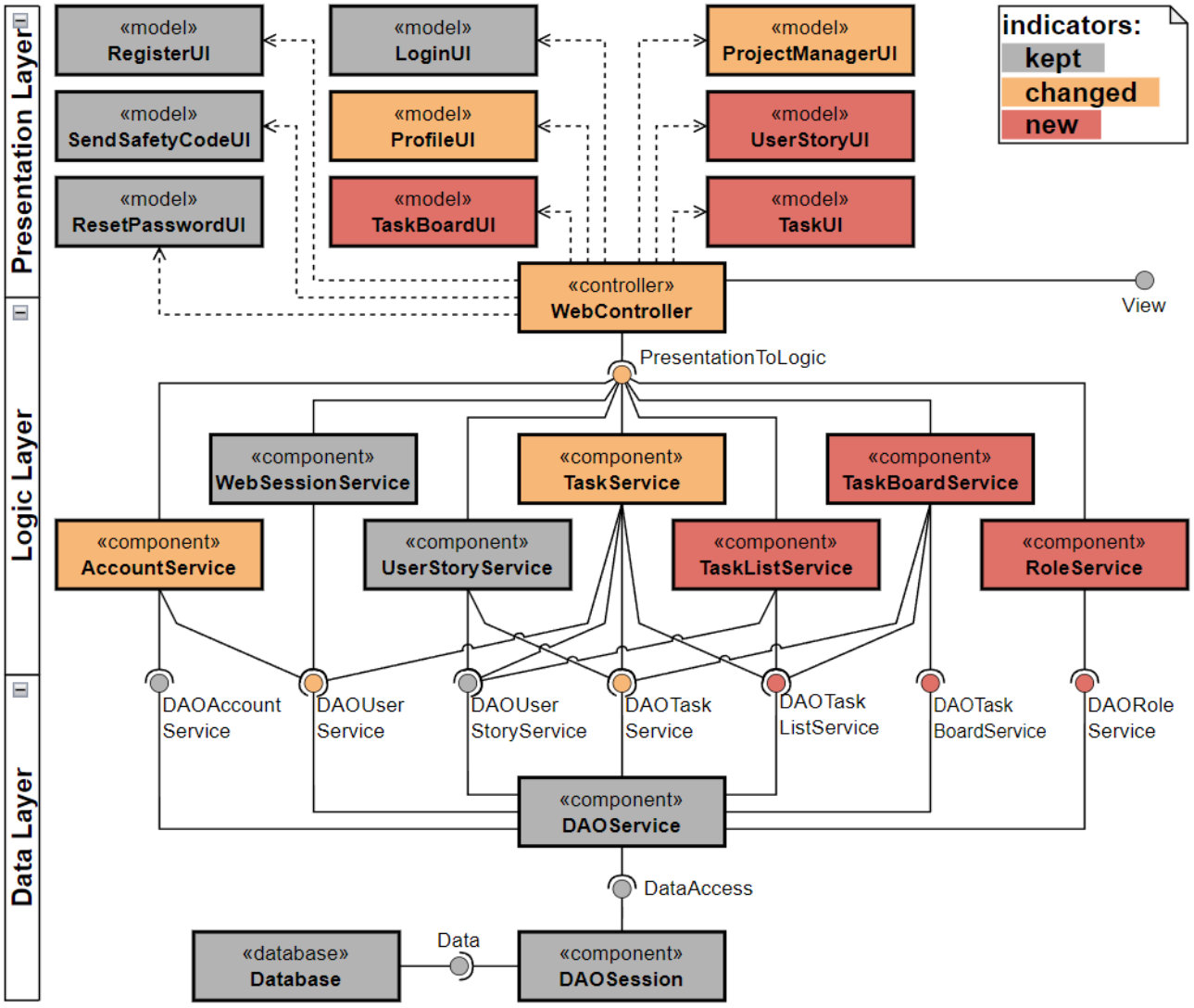


Abbildung 1: Komponentendiagramm Sprint 3

Wie in Abbildung 1 zu sehen, wurde diesen Sprint primär das Task-Board implementiert. So ist zu sehen (in rot), dass die Komponenten Task-BoardUI, UserStoryUI, TaskUI im PresentationLayer, TaskListService, TaskBoardService, sowie RoleService in Logic Layer und die Namensäquivalenten DAOServices im Data-Layer neu in das Komponentendiagramm aufgenommen wurden.
Diese Komponenten dienen primär der Darstellung und der Funktionalität des Task-Boards. Die weiteren Changes an den Komponenten (in gelb) beziehen sich im Allgemeinen auf die Profile von Nutzern und dessen Darstellung im Presentation Layer.

2.2 Klassendiagramme und Design Pattern

2.2.1 Designpattern

Diesen Sprint wurden keine weiteren Design Pattern in das Projekt eingebracht.

2.2.2 Klassendiagramm zum Backlog-Abschnitt Role

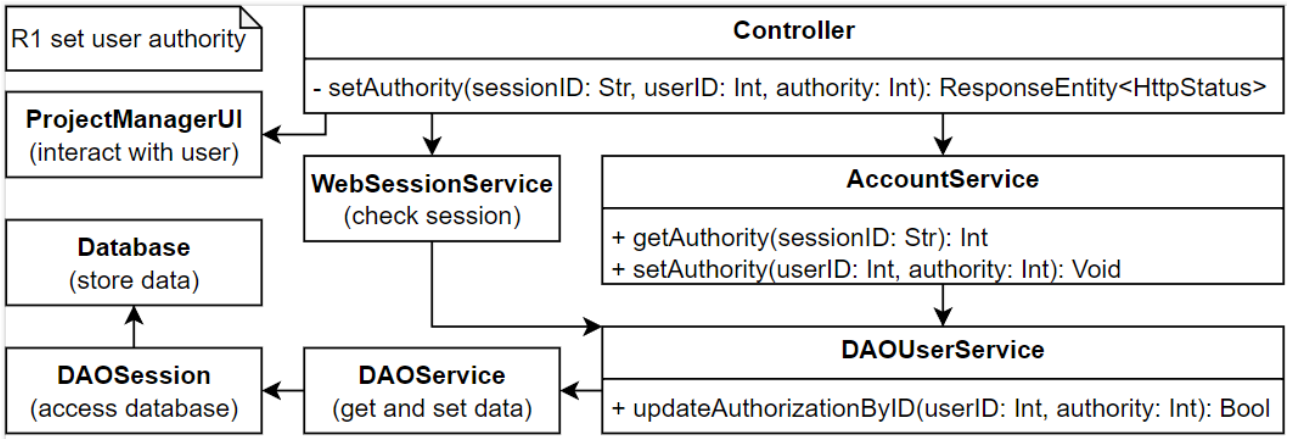


Abbildung 2: Klassendiagramm zum Abschnitt Role

Die User-Storys R2, R3, R4 und R5 wurden in diesem Sprint nicht implementiert und sind darum noch nicht im Klassendiagramm enthalten. Somit beinhaltet der Backlog-Abschnitt Role des 3. Sprints die User-Story R1 beziehungsweise das Verändern der rechte-bezogenen Rolle (Autorität), solang der Nutzer die benötigte Rolle (Manager) besitzt. Wenn ein Nutzer die Autorität eines beliebigen Nutzers verändern möchte, wird ein Request vom Client geschickt, der im “Controller“ die Methode “setAuthority“ aufruft, diese prüft zuerst, ob die übergebene SessionID noch gültig ist. Sobald dies gesichert ist, wird die Autorität des Nutzers mittels “getAuthority“ vom “AccountService“ angefragt. Diese verwendet “getbySession“, um aus der Datenbank den Nutzer auszulesen. Falls der Nutzer die benötigte Autorität hat, wird “setAuthority“ vom “AccountService“ ausgeführt, die wiederum “updateAuthorizationByID“ vom “DAOUserService“ verwendet, um mittels “DAOService“ die Autorität in der Datenbank zu ändern.

2.2.3 Klassendiagramm zum Backlog-Abschnitt Profile

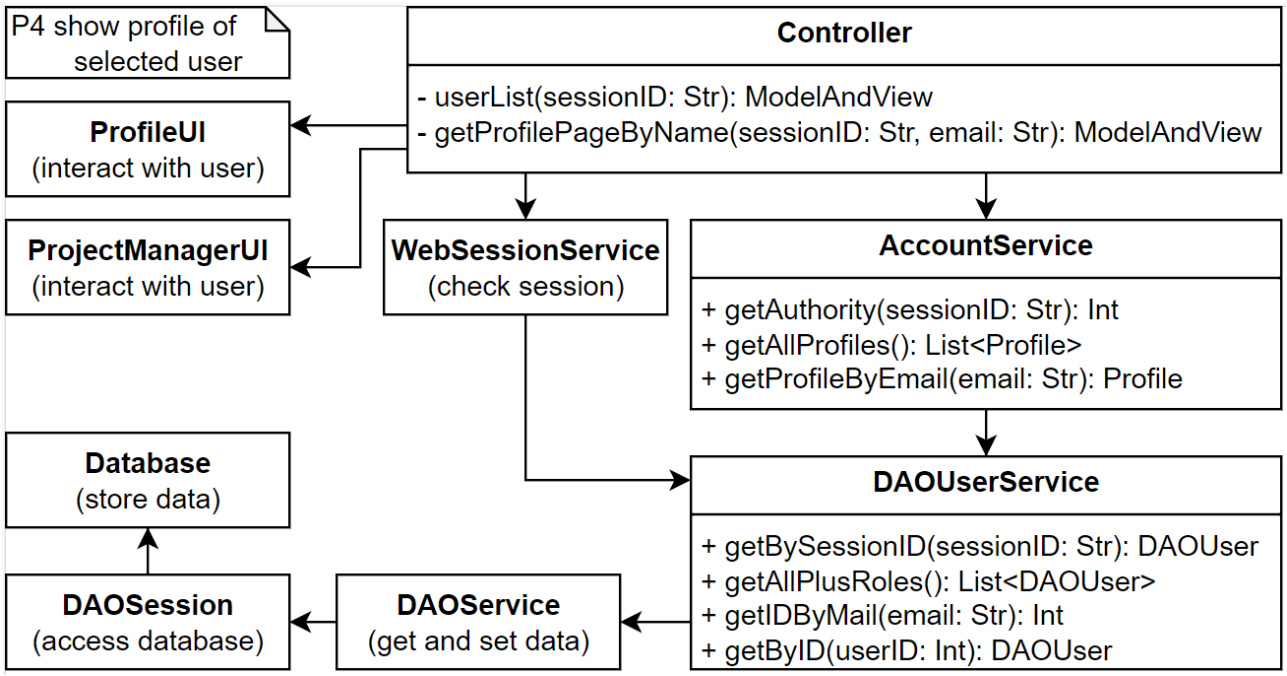


Abbildung 3: Klassendiagramm zum Abschnitt Profile

Der Backlog-Abschnitt Profil besteht aus der User-Story P4 beziehungsweise das Anzeigen des Profils eines beliebigen Nutzers. Um das Profil eines anderen Nutzers auswählen zu können, muss der Nutzer zum “Nuter“-Reiter vom “Project-Manager“ navigieren. Dort werden alle Nutzer angezeigt, von denen dieser sich eins auswählen kann. Damit der Nutzer alle Profile sieht, wird zu Beginn im „‘Controller“‘ die Methode „‘userList“‘ ausgeführt, welche, nachdem die Gültigkeit der SessionID sichergestellt wurde, im „‘AccountService“‘ die Methoden „‘getAllProfiles“‘ und „‘getAuthority“‘ aufruft. “getAuthority“ wird für eine andere Funktionalität von “userList“ benötigt, weshalb sie für diese User-Story vernachlässigt wird. „‘getAllProfiles“‘ ruft im „‘DAOUserService“‘ „‘getAllPlusRoles“‘ auf, die mittels „‘DAOService“‘ alle Nutzer aus der Datenbank erlangt. „‘getAllProfiles“‘ übergibt für jeden Nutzer die profilrelevanten Informationen an „‘userList“‘, welche die Information in die View integriert, und schickt diese an den Nutzer (Client). Sobald der Nutzer eines der Profile ausgewählt hat, wird ein Request an den „‘Controller“‘ geschickt, der die E-Mail des gewünschten Profils beinhaltet. Dies führt zum Aufruf der Methode „‘getProfilePageByName“‘, die, nach Sicherstellung der Gültigkeit der SessionID, wiederum im „‘AccountService“‘ die Methode „‘getProfileByEmail“‘ aufruft. Diese bekommt mittels „‘getIdByMail“‘ vom „‘DAOUserService“‘ die Nutzer-ID aus der Datenbank, die zur gegebenen E-Mail gehört. Die Nutzer-ID wird benötigt, um den Nutzer durch den Aufruf „‘getById“‘ vom „‘DAOUserService“‘ aus der Datenbank zu bekommen. „‘getProfileByEmail“‘ gibt die profilrelevanten Informationen an „‘getProfilPageByName“‘ zurück, der diese in die View des Nutzers integriert und das Profil des Nutzers angezeigt werden kann.

2.2.4 Klassendiagramm zum Backlog-Abschnitt User-Story

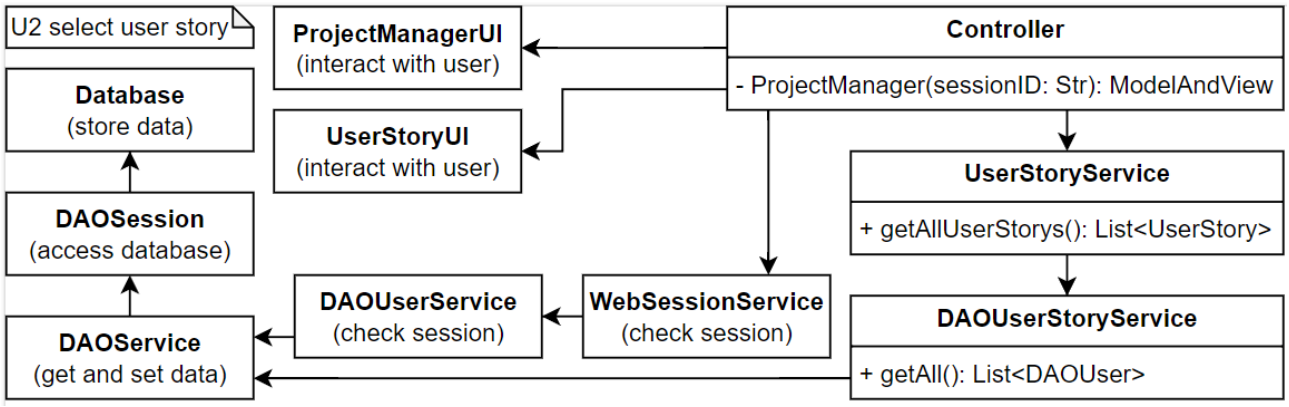


Abbildung 4: Klassendiagramm zum Abschnitt User-Story

Als zentrales Element zum Bearbeiten aller Attribute einer User-Story wurde die “UserStoryUI“ in der “ProjectManagerUI“ integriert. Diese wird beim Aufruf der “ProjectManager“-Funktion des Controllers zusammen mit der gleichnamigen “ProjectManagerUI“ generiert und über die Logic-Funktion “getAllUserStorys“ mit Daten belegt. Parallel wird über die Funktionen des “WebSessionService“ die Gültigkeit der SessionID des Nutzers gegenüber der Datenbank geprüft.

2.2.5 Klassendiagramm zum Backlog-Abschnitt Task

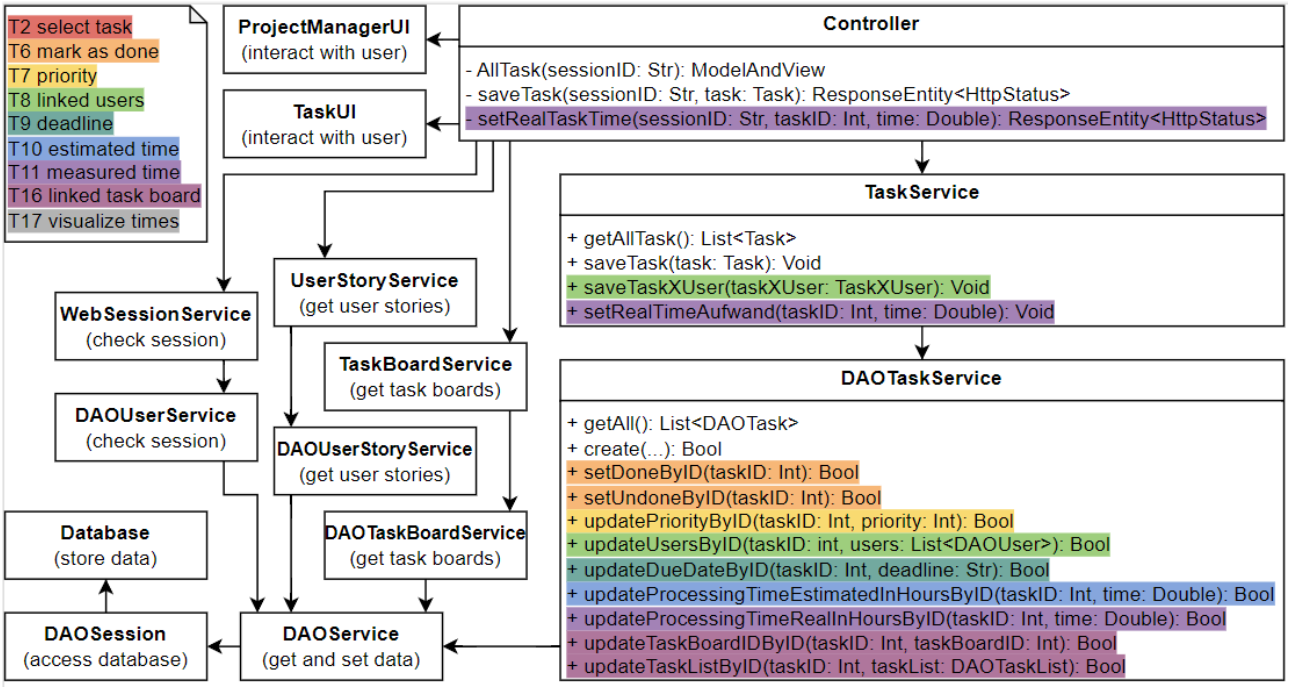


Abbildung 5: Klassendiagramm zum Abschnitt Task

In diesem Sprint beinhaltet der Backlog-Abschnitt Task das Auswählen einer Task (T2, rot in Abbildung 5) und das Erweitern der Task um verschiedenste Attribute: die Fertig-Markierung (T6, orange in Abbildung 5), die Priorität (T7, gelb in Abbildung 5), zugeteilte Nutzer (T8, hellgrün in Abbildung 5), die Abgabefrist (T9, grün in Abbildung 5), geschätzter Zeitaufwand (T10, blau in Abbildung 5), reeller Zeitaufwand (T11, lila in Abbildung 5), die Verknüpfung an ein Task-Board (T16, Magenta in Abbildung 5) und die Visualisierung des geschätzten und reellen Zeitaufwands (T17, grau in Abbildung 5).
Um eine Task auswählen zu können, muss der Nutzer zum Task-Board navigieren. Beim Aufruf dessen werden alle Tasks geladen. Somit wird beim Selektieren einer Task keine Abfrage der Datenbank benötigt.
Eine Task wird als fertig markiert genau dann, wenn die Task im Task-Board in die Spalte “fertiggestellte Tasks“ verschoben werden. Das Verschieben der Task führt zu einem Request, der im “Controller“ “saveTask“ aufruft. Nachdem die SessionID auf Gültigkeit geprüft wurde, wird im “TaskService“ die gleichnamige Methode aufgerufen. Diese ruft entweder “setDoneByID“, falls es als fertig markiert werden soll, oder “setUndoneByID“, falls es nicht mehr fertig markiert sein soll, im “DAOTaskService“ auf, welche mithilfe des “DAOService“ den Datenbankeintrag verändert.
Die Priorität einer Task kann sowohl bei der Erstellung einer Task angegeben werden als auch nachträglich beim Verändern der Task bearbeitet werden. Wenn der Nutzer eine Task erstellt, wird durch den dazugehörigen Request im “Controller“ “saveTask“ aufgerufen. Diese ruft nach Überprüfung der SessionID im “TaskService“ die Methode “saveTask“, die im “DAOTaskService“ die Methode “create“, die mittels “DAOService“ einen Eintrag in die Tasktabelle der Datenbank hinzufügt. Falls der Nutzer die Priorität nachträglich ändern möchte, wird durch den Request im “Controller“ “saveTask“ aufgerufen, welche die gleichnamige Methode im “TaskService“ verwendet. Diese ruft “updatePriorityByID“ vom “DAOTaskService“ auf, die mittels “DAOService“ die Priorität der Task in der Datenbank verändert.
Wenn ein Nutzer jemanden beim Erstellen oder Bearbeiten einer Task zuordnet, löst der Request

im “Controller“ den Aufruf von “saveTask“, dessen Funktionsweise ist im Sprint 2 näher beschrieben, und “setUserToTask“ aus. “setUserToTask“ ruft nach Validierung der SessionID “setUsers“ vom “TaskService“ auf. Mithilfe der Methode “updateUserById“ vom “DAOTaskService“ werden alle übergebenen Nutzer in der Datenbank an die gewünschte Task geknüpft.
Die Abgabefrist kann bei der Erstellung oder später bei der Bearbeitung einer Task festgelegt werden. Beide Request führen zum Aufruf der Methode “saveTask“ vom “Controller“, die nach Überprüfung der SessionID “saveTask“ vom “TaskService“ aufruft. Beim Erstellen der Task wird nun “create“ vom “DAOTaskService“ verwendet, um die Task inklusiv deren Abgabefrist mittels “DAOService“ in der Datenbank zu speichern. Beim Bearbeiten der Task wird “updateDueDateByID“ im “DAOTaskServiceäufgerufen, damit in der Datenbank die neue Abgabefrist der Task gespeichert wird.
Die geschätzte Zeit kann wie die zu vorigen Attribute beim Erstellen und Editieren einer Task festgelegt werden. Die Requests verursachen jeweils einen Aufruf der Methode “saveTask“ vom “Controller“, die wiederum die gleichnamige Methode im “TaskService“ aufruft. Im Falle der Bearbeitung einer Task, wird die Methode “updateProcessingTimeEstimatedInHoursByID“ des “DAOTaskService”verwendet, um mittels “DAOService“ den neuen geschätzten Zeitaufwand in der Datenbank abzuspeichern. Sonst wird “create“ vom “DAOTaskService“ aufgerufen, um die neue Task mit geschätzter Zeit in der Datenbank abzuspeichern.
Die reelle Bearbeitungszeit wird vom Nutzer verlangt, sobald er die Task in die “fertiggestellte Tasks“ Spalte vom Task-Board verschiebt. Der Request, der durch die Eingabe der Bearbeitungszeit entsteht, führt zum Methodenaufruf “setRealTaskTime“. Im “Controller“ verwendet “setRealTimeAufwand“ vom “TaskService“die “updateProcessingTimeRealInHoursByID“ des “DAOTaskService“, um mithilfe des “DAOService“ die Bearbeitungszeit in der Datenbank abzuspeichern.
Das Visualisieren des geschätzten und reellen Zeitaufwands ist nur vom Task-Board aus erreichbar. Somit sind die Daten der Tasks schon im Client vorhanden und es wird keine Abfrage der Datenbank benötigt.

2.2.6 Klassendiagramm zum Backlog-Abschnitt Task-Board

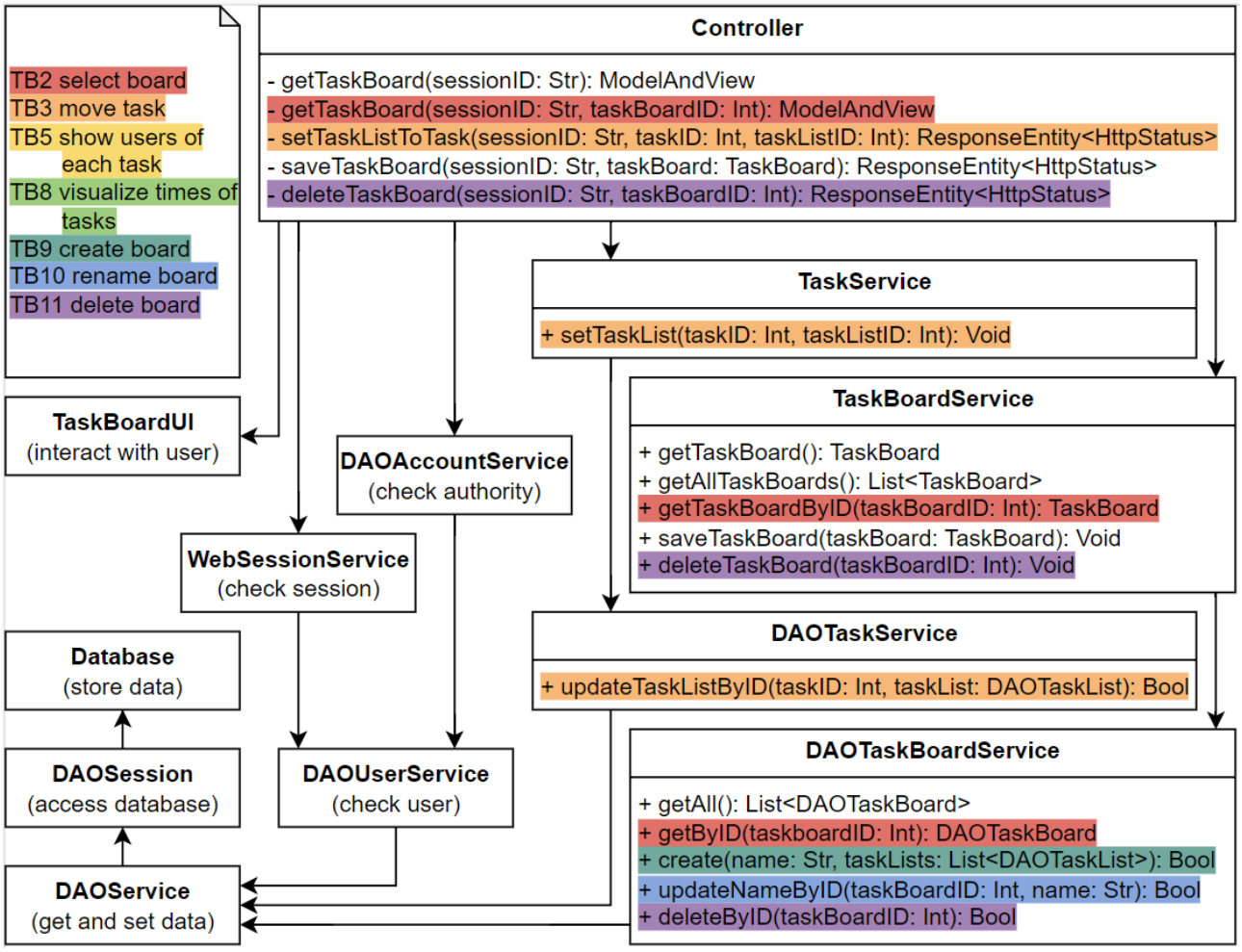


Abbildung 6: Klassendiagramm zum Abschnitt Task-Board

Abbildung 6 beschreibt alle Tasks zum Abschnitt Task-Board. Um diese zu unterscheiden, wurden die relevanten Funktionen mittels Farben differenziert, so wird Task TB2 (Abbildung 6 rot) durch das Aufrufen der Controller-Funktion “getTaskBoard“ erfüllt. Diese ermöglicht es über die TaskBoardService-Funktion “getTaskBoardByID“ und ferner der Daten-Funktion “getByID“ ein Task-Board aus der Datenbank auszuwählen und auszugeben.
Die Task TB3 (Abbildung 6 orange) ermöglicht es in der “TaskBoardUI“ Tasks mittels Drag-And-Drop in andere Task-Listen des geöffneten Task-Boards zu verschieben. Dies ruft dann die “setTaskListToTask“ Funktion des “Controllers“ aus, welche mittels der “setTaskList“ Funktion im “TaskService“ die “updateTaskListByID“ Funktion der Datenbank ausführt und somit die Zuweisung der Task zu einer neuen Task-Liste übernimmt.
TB5 (Abbildung 6 gelb) hat keine expliziten Funktionen zur Umsetzung in diesem Klassendiagramm benötigt, da es durch die Erweiterung existenter Funktionen und Objekte in den Task zum Abschnitt “Task“ schon erfüllt wurde. Dasselbe gilt für TB8 (Abbildung 6 hellgrün), da auch diese durch die Tasks des Abschnittes “Task“ erfüllt wurde.
Die Task TB9 (Abbildung 6 dunkelgrün) wird im Presentation Layer über die Controller-Funktion “saveTaskBoard“ (bewusst nicht farbig, weil es für mehrere Tasks benötigt wird) durch den Aufruf der “create“-Funktion realisiert. Sie ermöglicht das Erstellen eines neuen Task-Board und das Zuwei-

sen von Task-Listen zu diesem.
Ähnlich zu TB9 nutzt der Presentation Layer die “saveTaskBoard“ Funktion des “Controllers“ um TB10 (Abbildung 6 blau) zu erfüllen, welche in diesem Fall “updateNameByID“-Funktion des Database-Layers verwendet, um den Namen eines Task-Boards anzupassen.
Letztlich ist Task TB11 (Abbildung 6 violett) durch die Controller-Funktion “deleteTaskBoard“, welche die “deleteTaskBoard“-Funktion des “TaskBoardService“ und ferner die Funktion “deleteByID“ des “DAOTaskBoardService“, erfüllt. Wobei die Interaktion in der “TaskBoardUI“ somit das Löschen eines Task-Boards und der zugehörigen Task-Listen ermöglicht.
Parallel wird bei allen Tasks über den “WebSessionService“ die Gültigkeit der SessionID eines Nutzers gegenüber der Datenbank geprüft und bei allen Task, abgesehen von TB3, TB5 und TB8, wird die Autorität des Nutzers geprüft, um seine Rolle im Project zu verifizieren.

2.3 Datenbank

2.4 Neues/verändertes Verhalten

Aktuell basieren alle Funktionen auf dem bisher modellierten Verhalten, weshalb wir in diesem Sprint kein weiteres Verhalten hinzufügen oder bestehendes updaten werden. Die Sequenzdiagramme der letzten Sprint-Dokumente und des Grobentwurfes gelten weiterhin unter Berücksichtigung vergangener Updates.

2.5 Prototypen Frontend

Abbildung 7: Prototyp des Projekt-Managers

Es wurden in diesem Sprint Prototypen für den Projekt-Manager sowie für das Task-Board erstellt. Dabei gab es beim Projekt-Manager einige neue Änderungen im Vergleich zu dem letzten Projekt-Manager. Aus diesem Grund wurden die neuen Funktionen mit blau markiert, sodass im folgenden nur auf diese Neuerungen eingegangen wird.
Zum einen wurde geplant, eine komplett neue Ansicht für alle Nutzer zu erstellen. Hierbei besitzt jeder Nutzer eine Rechte-Rolle, welche ihnen den Zugriff auf bestimmte Funktionen erlauben, aber auch verweigern soll. Die Rechte-Rolle des Nutzers wurde dabei im Prototypen als ein einfacher Text dargestellt, wobei dies nun für Manager als Dropdown erkennbar gemacht wurde. Weiterhin wurden ursprünglich zwei neue Schaltflächen geplant, wie es durch das "E" und "+" zu sehen ist. Diese sollten nur für den Manager oder einer anderen Rolle mit genügend Rechten zu sehen sein. Das eine Fenster war für die Rechtevergabe geplant, das andere für die Erstellung und Bearbeitung von visuellen Rollen durch den Manager. Das ursprüngliche Rechte-Fenster wurde allerdings wie bereits erwähnt als ein Dropdown-Menü neben dem Nutzernamen implementiert, und die visuellen Rollen wurden in den nächsten Sprint verschoben. Näheres dazu wurde in der Sektion "Abweichungen zur Sprintplanung" erläutert. Weiterhin ist im Prototypen nicht die aktuell implementierte Funktion "Schätzungstracker" zu sehen, da dies ursprünglich im Task-Board geplant durch den Prototypen geplant wurde. In der aktuellen Version kann man jedoch auch im Projekt-Manager auf den Schätzungstracker zugreifen. Eine weitere Abweichung vom Prototypen sind die Eigenschaften der Tasks, wie "Aufwandsschätzung". Diese ist im Prototypen nicht enthalten, wurde jedoch im laufenden Prototypen hinzugefügt. Ebenfalls wurde die Fertig-Markierung einer Task nicht im Prototypen beschrieben.

Beim Prototypen für das Task-Board werden zum einen die Tasks zunächst mit ihren vier wichtigsten Eigenschaften dargestellt, welche als Erstes für den Nutzer sichtbar sein sollten: "Rolle, Priorität (farblich gekennzeichnet), Beschreibung, Abgabedatum". Diese Tasks werden in den Spalten "To-do, in Bearbeitung, in Review, Fertig" eingeteilt, wobei sich die Spaltennamen im laufenden Prototypen geändert haben. Weiterhin hat jedes Task-Board im Prototypen einen eigenen Namen, welche aufgelistet über den Spalten zu sehen sind. Die "+" Schaltfläche neben dem Namen soll es dem Manager möglich machen, ein neues Task-Board hinzuzufügen. Im Prototypen wurden jedoch viele Funktionalitäten, welche im laufenden Prototypen vorhanden sind, nicht entworfen. Dies betrifft das Löschen eines Task-Boards, das Ändern eines Task-Board Namen, den Schätzungstracker, die Angabe der Bearbeitungszeit bei einer fertigen Task, sowie die detaillierte Ansicht einer Task. Diese wurden erst im Laufe des Sprints entschieden und mit in die Tasks aufgenommen, sodass hier ein unfertiger Prototyp zu sehen ist.

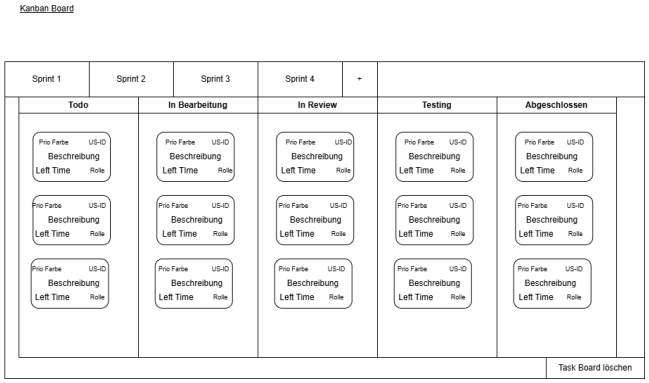


Abbildung 8: Prototyp des Task-Boards

3 Implementation und Tests

3.1 Tests

3.1.1 HTTP-Request-Test

Die Test-Logs sind zu finden unter:
Project/src/test/java/com/team3/project/logs/log_HttpRequestTest.txt
Der Coverage-Report bezüglich der WebControllers ist zu finden unter:
Project/src/test/java/com/team3/project/logs/HTTPCoverage
Es wurde endgültig eine Line-Coverage von 100% und eine Branch-Coverage von 90% erreicht.

TestArt.TestID	Erstellungszeit	Herkunft	Author
HTTP.T18	20.05.2024	Grobentwurf	Lucas Krüger
HTTP.T19	20.05.2024	Grobentwurf	Lucas Krüger
HTTP.T20	22.05.2024	Grobentwurf	Lucas Krüger
HTTP.T21	24.05.2024	Grobentwurf	Lucas Krüger
HTTP.T22	26.05.2024	Grobentwurf	Lucas Krüger
HTTP.T23	01.06.2024	Grobentwurf	Lucas Krüger
HTTP.T24	05.07.2024	Grobentwurf	Lucas Krüger
HTTP.T25	05.07.2024	Grobentwurf	Lucas Krüger
HTTP.T26	05.07.2024	Grobentwurf	Lucas Krüger
HTTP.T27	05.07.2024	Grobentwurf	Lucas Krüger

Darüber hinaus unterliefen alle Tests im Laufe des Sprints zwei Refactoring-Schritte, um das Hinzufügen von neuen Tests und letztlich die Coverage der Tests insgesamt zu verbessern.
Das letzte Refactoring der Tests geschah am 4.06.2024

3.1.2 Datenbank-Test

Die Datenbank-Tests prüfen die Funktionalitäten fürs Laden und schreiben von Daten aus der Datenbank.
Neu implementierte Tests:

TestArt.TestID	Erstellungszeit	Herkunft	Author
DB.R.T1	25.05.2024	Grobentwurf	Marvin Prüger
DB.R.T2	25.05.2024	Grobentwurf	Marvin Prüger
DB.T.T3	25.05.2024	Grobentwurf	Marvin Prüger
DB.T.T4	25.05.2024	Grobentwurf	Marvin Prüger
DB.T.T5	25.05.2024	Grobentwurf	Marvin Prüger
DB.TB.T1	26.05.2024	Grobentwurf	Marvin Prüger
DB.TB.T2	26.05.2024	Grobentwurf	Marvin Prüger
DB.TB.T3	26.05.2024	Grobentwurf	Marvin Prüger

Testlog für Datenbank-Tests unter:
Project/src/test/java/com/team3/project/logs/log_DatabaseTest.txt

3.1.3 Logic-Test

Die Logic-Tests überprüfen die Funktionalität der Methoden, die vom “Controller“ verwendet werden. Neu dazugekommene Test sind in der unten angrenzenden Tabelle dokumentiert. Die Test-Logs sind zu finden unter:
Project/src/test/java/com/team3/project/logs/log_LogicTest.txt

TestArt.TestID	Erstellungszeit	Herkunft	Erfolgreich abgeschlossen	Author
Logic.T9	17.05.2024	A2.B1	Ja	Henry Lewis Freyschmidt
Logic.T10	17.05.2024	A3.B1	Ja	Henry Lewis Freyschmidt
Logic.T11	19.05.2024	R1.B1	Ja	Henry Lewis Freyschmidt
Logic.T12	20.05.2024	R1.B2	Nein	Henry Lewis Freyschmidt
Logic.T13	19.05.2024	R2.B1	Nein	Henry Lewis Freyschmidt
Logic.T14	03.06.2024	R2.B2	Nein	Henry Lewis Freyschmidt
Logic.T15	25.05.2024	R3.B1	Nein	Henry Lewis Freyschmidt
Logic.T16	25.05.2024	R4.B1	Nein	Henry Lewis Freyschmidt
Logic.T17	23.05.2024	R5.B2	Nein	Henry Lewis Freyschmidt
Logic.T18	20.05.2024	P4.B1	Ja	Henry Lewis Freyschmidt
Logic.T19	08.06.2024	T8	Nein, jedoch funktioniert die Funktionalität	Henry Lewis Freyschmidt
Logic.T20	24.05.2024	T16.B2	Ja	Henry Lewis Freyschmidt
Logic.T21	03.06.2024	T16.B3	Ja	Henry Lewis Freyschmidt
Logic.T22	03.06.2024	T7.B4	Ja	Henry Lewis Freyschmidt
Logic.T23	03.06.2024	T4.B3	Ja	Henry Lewis Freyschmidt
Logic.T24	03.06.2024	T10.B3	Ja	Henry Lewis Freyschmidt
Logic.T25	03.06.2024	T10.B2	Ja	Henry Lewis Freyschmidt
Logic.T26	03.06.2024	T9.B3	Ja	Henry Lewis Freyschmidt
Logic.T27	03.06.2024	T9.B2	Ja	Henry Lewis Freyschmidt

3.2 Tracing - Komponenten im Code

Ausgehend von: .../projekt-team3-uebung3/Project/src/main/

Komponente	Implementation
User Interfaces	
RegisterUI	resources/templates/register.html
LoginUI	resources/templates/index.html
SendSafetyCodeUI	resources/templates/passwortForgot.html
ResetPasswortUI	resources/templates/neuesPasswort.html
ProfileUI	resources/templates/profil.html
ProjectManagerUI	resources/templates/projectManager.html
UserStoryUI	resources/templates/projectManager.html resources/templates/projectManager-TasksZuUserStory.html
TaskUI	resources/templates/projectManager-Tasks.html
TaskBoardUI	resources/templates/taskBoard.html
Controller	
WebController	java/com/team3/project/Controller/WebController.java
Facade	
PresentationToLogic	java/com/team3/project/Facede/PresentationToLogic.java
Services	
AccountService	java/com/team3/project/service/AccountService.java
RoleService	java/com/team3/project/service/RoleService.java
TaskBoardService	java/com/team3/project/service/TaskboardService.java
TaskListService	java/com/team3/project/service/TaskListService.java
TaskService	java/com/team3/project/service/TaskService.java
UserStoryService	java/com/team3/project/service/UserStoryService.java
WebSessionService	java/com/team3/project/service/WebSessionService.java
DAOServices	
DAOAccountService	java/com/team3/project/DAOService/DAOAccountService.java
DAORoleService	java/com/team3/project/DAOService/DAORoleService.java
DAOService	java/com/team3/project/DAOService/DAOService.java
DAOTaskBoardService	java/com/team3/project/DAOService/DAOTaskBoardService.java
DAOTaskListService	java/com/team3/project/DAOService/DAOTaskListService.java
DAOTaskService	java/com/team3/project/DAOService/DAOTaskService.java
DAOUserService	java/com/team3/project/DAOService/DAOUserService.java
DAOUserStoryService	java/com/team3/project/DAOService/DAOUserStoryService.java
Database	
Database	resources/DB/database.db

Tabelle 5: Tracing - Implementation

3.3 Tracing - Beendete Aufgaben

Die folgenden Aufgaben-Bezeichnungen beziehen sich auf die Bezeichner der Projektaufgabe.pdf und der Status ist eine subjektive Einschätzung unsererseits ohne Abspreche mit einem externen Produkt Owner oder Shareholder.

Aufgabe	Status	zugehörige Komponenten
01. Nutzerregistrierung	Erledigt	LoginUI RegisterUI SendSaftCodeUI ResetPasswortUI AccountService DAOUser-Table
02. User Story- & Backlog-Management	Erledigt	ProjectManagerUI UserStoryUI UserStoryService RoleService DAOUserStory-Table
03. Task Management	Erledigt	ProjectManagerUI TaskUI TaskService DAOTask-Table
04. Task/User Story Listen	laufend	ProjectManagerUI TaskBoardUI TaskService TaskBoardService UserStoryService DAOUserStory-Table DAOTask-Table DAOTaskBoard-Table
05. Nutzer Profile	Erledigt	ProfileUI ProjectManagerUI AccountService DAOUser-Table
06. Nutzer-Task Zuordnung	laufend	ProjectManagerUI RoleUI AccountService TaskService DAOTaskXUser-Table
07. Pair-Programming Planung	laufend	ProjectManagerUI RoleUI AccountService TaskService DAOTaskXUser-Table

Tabelle 6: Tracing - Aufgaben

08. Schätzungstracker	laufend	TaskBoardUI TaskService DAOTask-Table
09. Suche und Filter	anhaltend	ProjectManagerUI TaskService UserStoryService DAOTask-Table DAOUserStory-Table
10. Benachrichtigungen	anhaltend	TaskService DAOTask-Table
11. Agile Practices	in Planung	
12. Synchronisation	in Planung	TaskService UserStoryService UserService
13. Parallele Bearbeitung	in Planung	TaskBoardUI TaskService UserStoryService UserService
14. Multi-Nutzer	laufend	WebSessionService DAOUser-Table
15. Verfügbarkeit	-	
16. Updates	-	
17. Änderungen	-	

Tabelle 7: Tracing - Aufgaben

3.4 Laufender Prototyp

Im Projekt-Manager wurden viele neue Tasks umgesetzt, zum einen R1 für die Rechte eines Nutzers und die Ansicht aller Nutzer durch P4, zum anderen die Zuweisung von Nutzern zu einer Task durch T8 sowie der Schätzungstracker durch TB8, wobei dieser auch im Task-Board enthalten ist. Die erste neue Funktionalität im Projekt-Manager ist die Ansicht aller Nutzer in einem Menü sowie die Rechteverteilung für die Nutzer durch den Manager. Dies wurde mittels einer neuen Auswahlmöglichkeit im Projektmanager umgesetzt. Neben den Begriffen 'User Story' und 'Task' befindet sich nun 'Nutzer', wobei beim Draufklicken sich die Ansicht aller Nutzer öffnet. Jeder Nutzer wird mit seinem Namen und seiner Rechte angezeigt. Hier hat beispielsweise der Manager die Möglichkeit, durch eine Dropdown-Funktion die Rechte des jeweiligen Nutzers zu ändern (R1). Weiterhin wird man auf das Profil des Nutzers weitergeleitet, wenn man Doppelklick auf ein Nutzerkärtchen ausführt (P4).
Eine weitere Funktion, welche nicht im ursprünglichen Prototypen selbst aufgezeichnet wurde, ist die Zuweisung von Nutzern für die Task. Diese kann beim Erstellen und Ändern einer Task gefunden werden, indem beim Klicken der Schaltfläche "Nutzer zuweisen" ein Dropdown-Menü geöffnet wird, bei welchem Nutzer durch das Setzen eines Häkchens ausgewählt werden können. Wurden Nutzer zugewiesen und die Erstellung / Bearbeitung der Task gespeichert, so kann man dies auf dem Task-Board sehen. Durch Doppelklick auf eine Task erscheint wie bereits erwähnt eine vergrößerte Task, bei welcher einer der vielen Eigenschaften "Zugewiesene Nutzer" ist. Beim Hovern des Mauszeigers über diese Option erscheint anschließend ein unteres Fenster, welches die Liste aller zugewiesenen Nutzer enthält.
Bei der Bearbeitung sowie Erstellung einer Task gibt es weitere neue Eigenschaften, welche einer Task gegeben wurden, wie das Zuweisen zu einem Task-Board (T16) sowie einer Aufwandseinschätzung (T10). Die Zuweisung zu einem Task-Board wurde mittels eines Dropdown-Menüs umgesetzt, wobei die Aufwandsschätzung als Zahl eingegeben werden muss. Die Zahl ist dabei immer in 0,5 Schritten

und darf nicht 0 sein.
Weiterhin können Tasks eine Fertig-Markierung haben. Wird im Task-Board eine Task auf die Spalte “Fertig“ geschoben, so erscheint auf dieser Task im Projekt-Manager eine grüne Markierung “Erledigt“ mit einem Häkchen, dies erfüllt die Task T6. Die letzte neue Funktion des Projekt-Managers ist der Schätzungstracker. Diesen findet man neben dem großen dunkelblauen Titel “User-Stories“. Klickt man darauf, so öffnet sich der Schätzungstracker für alle bisher erstellten Tasks, welcher die Abweichung zwischen der Bearbeitungszeit und der geschätzten Zeit als Balkendiagramm darstellt (TB8).

Neben all den bereits angeführten Punkten wurden weitere Tasks erfüllt innerhalb des Projekt-Managers. Durch Klicken auf Userstories im Reiter des Projekt-Managers gelangt man zu der Tabelle für alle Userstories, wobei man in dieser Übersicht bereits alle zur Userstory gespeicherten Informationen sieht (U2). In dem man auf den “Bearbeiten“-Button drückt, welcher sich neben dem “x“-Button befindet innerhalb einer Userstory, so kann man alle sich in der Userstory befindlichen Labels verändern (U2). Im Reiter zum Projekt-Manager befindet sich ein “Userstory erstellen“-Button, mit dem sich eine Userstory erstellen lässt (U2.F4) mit den Parametern Titel, Beschreibung und Priorität.
Klickt man im Reiter auf Tasks, so kann man alle Tasks sehen innerhalb der Tabelle mitsamt aller zu diesen gespeicherten Informationen (T2). Will man eine Task bearbeiten, so klickt man auf den dazugehörigen Button neben dem “x“-Button und kann alle Labels der Task verändern (T2). Will man eine Task erstellen, so klickt man im Reiter auf den “Task erstellen“-Button und kann eine Task erstellen (T2). Will man eine Task als “Fertig“ markieren, so muss man diese im Task-Board auf die Spalte “fertiggestellte Tasks“ schieben. Danach wird eine Markierung im Projekt-Manager in Tasks zur jeweiligen Task hinzugefügt. Wird sie aus dieser Liste geschoben im Task-Board, so verschwindet auch die Markierung (T6). Im Bearbeitungs- oder Erstellungsmenü kann man die Priorität einer Task anpassen/festlegen. Diese wird dann auch im Project Manager angezeigt, innerhalb von Tasks (T7). Außerdem geht dasselbe mit der Abgabefrist (T9), der Aufwandsschätzung (T10) und der Zuordnung zu einer Task-Liste (T16). Möchte man eine Bearbeitungszeit zu einer Task hinzufügen, so geschieht dies im Task-Board. Wird eine Task auf “fertiggestellte Tasks“ geschoben, so öffnet sich ein Fenster, dass nach der Bearbeitungszeit fragt. Bestätigt man diese Eingabe, so wird sie in der gezoomten Task angezeigt, welche man durch Doppelklicken auf die Task erreicht (T11).

Zum Task-Board gelangt man durch Klicken der jeweiligen Option im Reiter. Die tabellarische Darstellung entspricht der Anzeige des Task-Boards (TB2.F1). Oberhalb der Tabelle findet man alle vorhandenen Task-Boards. Klickt man doppelt auf einen dieser Buttons, so gelangt man zum korrespondierenden Task-Board (TB2.F2). Innerhalb des Projekt-Managers lassen sich beispielsweise die zu einer Userstory erstellten Tasks einem Task-Board zuordnen. Diese werden dann im Task-Board angezeigt. Durch Drag-and-Drop lassen sich diese Tasks innerhalb der 5 Spalten der Tabelle verschieben (TB3).
Drückt man doppelt auf eine Task, so öffnet sich die Task mit allen zu ihr gehörenden Informationen. Hierbei findet man unter anderem durch Hovern über “Zugewiesene Nutzer“ die zugewiesenen Nutzer (TB5.F1). Allerdings noch nicht vollständig implementiert, da die Auflistung der Nutzer noch fehlt. Um die Visualisierung für alle fertigen Tasks einzusehen, muss man im Project Manager innerhalb der Userstories-Anzeige auf das Balkendiagramm-Symbol klicken. Dann öffnet sich eine Visualisierung in Form eines Balkendiagramms, das die Summe der Bearbeitungszeiten und Schätzungen in Relation setzt (TB8). Klickt man auf den “+“-Button innerhalb des Fensterreiters, so erstellt man ein neues Task-Board mit den Spalten ”’freie Tasks“’, “’Tasks in Bearbeitung“’, “’Tasks unter Review“’, “’Tasks unter Test“’ und “’fertiggestellte Tasks“’ (TB9).
Will man ein Task-Board umbenennen, so klickt man auf den “’Namen ändern“’-Button. Dabei wird ein Fenster geöffnet für die Namensänderung. Klickt man auf Speichern, so wird der Name des ausgewählten Task-Boards verändert (TB10). Ist die Löschung eines Task-Boards gewollt, so klickt man auf den “’x“’-Button innerhalb des Task-Board Buttons und das Task-Baord wird gelöscht (TB11).

3.5 Abweichungen von Sprintplanung

Die in diesem Sprint vorgenommenen Tasks sind größtenteils beendet worden, wobei lediglich zwei große Änderungen vorgenommen wurden. Zum einen wurde die Nutzer-Ansicht nicht wie geplant in der Profilansicht, sondern im Projekt Manager umgesetzt. Dies haben wir aufgrund der Nutzerfreundlichkeit für den Manager entschieden. Aufgrund dessen, dass der Manager bei Änderung von mehreren Rechten nicht auf jedes Profil einzeln klicken soll, wurde die Bearbeitungsmöglichkeit zentral auf den Projekt-Manager verlegt. Somit werden für jeden Nutzer seine Rechte angezeigt, welche mit einem einfachen Dropdown-Menü geändert werden können, statt eines Menüs bei der Profilseite.
Zum anderen wurde das Erstellen und Bearbeiten der visuellen Rollen durch den Manager in den nächsten Sprint verlagert, da dies zeitlich nicht geschafft wurde. Dies betrifft die Tasks R2, R3, R4 und R5. Die Implementierung des Task-Boards hat mit all seinen Funktionalitäten zu viel Zeit eingenommen, sodass das Erstellen, Bearbeiten der visuellen Rollen durch den Manager sowie der Auswahl dieser visuellen Rollen durch den Nutzer zeitlich nicht geschafft wurde. Weiterhin ist für die Implementierung der visuellen Rollen eine Anpassung des Projekt-Managers Profils, sowie des Task-Boards nötig gewesen. Diese Seiten waren allerdings in diesem Sprint alle noch unter Bearbeitung für andere Features, sodass dies die Entscheidung der Verschiebung ebenfalls bestärkt hatte.
Weitere, kleinere Abweichungen sind die Positionierungen der Schätzungstracker. Für den Schätzungstracker der einzelnen Tasks sollte dies im Projekt-Manager geschehen, dafür der Schätzungstracker für alle fertigen Tasks auf dem Task-Board. Dies wurde allerdings umgedreht umgesetzt, sodass der Schätzungstracker für die einzelnen Tasks auf dem Task-Board, und der Schätzungstracker für alle fertigen Tasks sich auf dem Projekt-Manager befindet. Ebenfalls wurde bereits die Anzeige der zugewiesenen Nutzer in der vergrößerten Task-Anzeige implementiert, obwohl dies ursprünglich für den nächsten Sprint gedacht war.Da diese Funktionalität jedoch noch nicht ganz fertig ist, muss dies im nächsten Sprint noch überarbeitet werden.

3.5.1 Technologien

In diesem Sprint gab es keine Änderungen an den genutzten Technologien.
Es wird weiterhin folgendes genutzt:

- Im Backend
  - Java
  - Spring
  - Spring Boot
- Im Frontend
  - HTML
  - JavaScript
- In der Datenbank-Schicht
  - SQLite
  - Hibernate