Einsatzplanungstool für Compass Security

Studienarbeit

Studiengang Informatik
OST – Ostschweizer Fachhochschule
Campus Rapperswil-Jona

Herbstsemester 2021

Autoren:	Abdullah Almaz & Ursin Zimmermann
Betreuer:	Prof. Dr. Markus Stolze
Projektpartner:	Cyrill Brunschwiler (Compass Security, Jona)

Abstract

Firmen welche Dienstleistungen für Kunden erbringen, benötigen so gut wie immer eine Software fürs Projektmanagement. Daher gibt es auch sehr viele Projektmanagementlösungen, die diese Firmen dabei unterstützen sollen. Im Falle der Compass Security wird aktuell eine solche Projektmanagement-Software verwendet, welche aber nun durch eine neue Webapplikation ergänzt werden soll.

Diese Arbeit befasst sich mit der Anforderungsanalyse, Konzeption und Umsetzung eines Prototyps dieser Webapplikation für die Einsatzplanung von Mitarbeitern. Im Rahmen dieser Arbeit wurden mit dem Auftraggeber Anforderungen iterativ erhoben.
Basierend auf den gesammelten Anforderungen entstand der hierbei entwickelte Prototyp. Dieser umfasst für die Compass Security zentrale Funktionalitäten, wie zum Beispiel Single Sign-on, Synchronisation von Updates über verschiedene Geräte, sowie die Planung von Mitarbeiter-Einsätzen mittels direkter Manipulation.

Vor Abschluss der Arbeit wurden mit Mitarbeitern der Compass Security ein Usability Test durchgeführt, um wichtige Inputs zur User Experience zu sammeln. Kritische Fehler oder schnell umsetzbare Features wurden noch vor Ende der Arbeit umgesetzt. Die weiteren Massnahmen, welche durch diese Tests entstanden sind, wurden für die spätere Weiterentwicklung dokumentiert und soweit sinnvoll mit einer Zeitschätzung ergänzt.

Management Summary

Einsatzplanungstool für Compass Security

Diplomanden	Abdullah Almaz & Ursin Zimmermann
Betreuer	Prof. Dr. Markus Stolze
Themengebiet	Internet-Technologien und -Anwendungen
Industriepartner	Compass Security

Ausgangslage:

Die Compass Security arbeitet mit einer Projektmanagement-Software. In dieser wird auch geplant welche Mitarbeiter wann an welchem Projekt arbeiten. Neu soll diese mit einer Webapplikation erweitert werden, in welcher die Planung übersichtlicher, einfacher und effizienter wird. Wichtig ist, dass bereits bestehende Daten zu Projekten, Mitarbeitern und deren geplanten Absenzen fortlaufend in die Webapplikation importiert werden.

Vorgehen/Technologien:

In Zusammenarbeit mit der Compass Security wurde eine Anforderungsanalyse erarbeitet, in welcher definiert wurde, dass ein Prototyp entstehen soll, welcher dann auch weiter ausgebaut werden kann. Der Prototyp soll aus einer Webapplikation, einem Backend und einer Datenbank bestehen. Die Webapplikation wurde mit Angular entwickelt und das Backend mit Java Play. Bei der Datenbank handelt es sich um eine MySQL Datenbank.

Ergebnis:

Die Webapplikation ist stark auf die Bedienung mit einer Maus ausgelegt, da auf Wunsch von Compass Security die Projektplanung mit Drag & Drop konzipiert und realisiert wurde. Diese kommuniziert über eine REST Schnittstelle mit dem Backend. Da mehrere Benutzer zeitgleich an der Planung arbeiten können, sendet das Backend Updates über WebSockets an alle verbundenen Clients.
Die Primärdaten aus der bisherigen Software werden in einem konfigurierten Intervall in die Datenbank importiert und ebenso über WebSockets an alle verbundenen Clients gesendet.
Die Software wird mit automatischen Unit- und Integrationstests abgesichert. Zusätzlich wurden Performance Tests an der Software durchgeführt und deren Resultate dokumentiert.
Gegen Ende der Arbeit wurden auch Usability Tests mit Mitarbeitenden der Compass Security durchgeführt. Dadurch konnten weitere Feature-Wünsche und Fehler in der Software identifiziert und dokumentiert werden.

Da die Studienarbeit als Grundlage zur Weiterentwicklung dient, wurden alle wichtigen Funktionalitäten, bekannte Fehler sowie potenzielle Optimierungen dokumentiert.

Inhaltsverzeichnis

Abbildungsverzeichnis

Aufgabenstellung

Aufgabenstellung Studienarbeit «Einsatzplanungtool Compass»

1. Betreuer

Prof. Dr. Markus Stolze, OST, IFS
markus.stolze@ost.ch

2. Praxispartner

Compass Security
Cyril Brunschwiler
cyrill.brunschwiler@compass-security.com

3. Studierende

- Abdullah Almaz <abdullah.almaz@ost.ch>
- Ursin Zimmermann <ursin.zimmermann@ost.ch>

4. Ziel der Arbeit

Erstellung eines Einsatzplanungstools für Compass Security

5. Problembeschrieb und Auftrag

Die folgenden Teilaufgaben sind zu bearbeiten

- Analyse: Erhebung und Dokumentation der Anforderungen
  - Beschreibung der Ausgangssituation. Beschreibung des Status Quo:
    Messung/Dokumentation von Zeiten für die Planung eines einfachen und komplexen Einsatzes (plus Liste möglicher Fehlerquellen und «Zeitfresser»)
  - Dokumentation der «Technology Constraints»
  - Funktionale Anforderungen (notwendig, optional)
  - Nicht-Funktionale Anforderungen (notwendig, optional)
- Entwurf: Entwicklung Architektur und Vorschlag UX
  - Analyse sinnvoller Libraries und Frameworks
  - Beschreibung der Nutzer und Rollen
  - Prototyping UX (Paper, Scenario Walkthrough, ...)
    Definition eines MVP + Hypothesen zu Ausbauschritten
    Dokumentation der Erwartungen zu Zeiten für die Planung des einfachen und komplexen Einsatzes (plus Liste von vermiedenen Fehlerquellen und Zeitfressern mit Begründung)
  - Backend/Frontend Separation & API Prototyping
  - Demonstration Architektur-Prototyp und validiertem UX Vorschlag ca. zur Mitte des Projektes.

- Implementation und Test (mit Referenz auf FA/NFA)
  - Dokumentation des Erreichungsgrads der FA/NFA
  - Beschreibung weiter Limitation
  - Developer und User-Dokumentation (Quick-Start) für die unterschiedlichen Rollen
  - Beschreibung weitere Ausbauschritte mit Aufwandschätzung

6. Termine

KW 38 Beginn der Arbeit

KW 51 Freitag, 17 Uhr, Abgabe der Studienarbeit

7. Beurteilung

Eine erfolgreiche Studienarbeit zählt 8 ECTS-Punkte pro Studierenden.
Für 1 ECTS Punkt ist eine Arbeitsleistung von 30 Stunden.

Für die Beurteilung ist der verantwortliche Dozent zuständig.

Gesichtspunkt	Gewicht
1. Organisation, Durchführung	20%
2. Formale Qualität des Berichts (Gliederung, Darstellung, Sprache der gesamten Dokumentation)	20%
3. Analyse, Entwurf, Auswertung	20%
4. Technische Umsetzung	40%

8. Zulässige Hilfsmittel und weitere Betreuung

Keine

9. Schlussbestimmungen

Im Übrigen gelten die Bestimmungen des Studiengangs Informatik für Studienarbeiten.

Rapperswil, 21 Sept 2021

Prof. Dr. Markus Stolze

Teil 1: Technischer Bericht

1. Einführung

1.1 Rahmenbedingungen, Umfeld, Definitionen, Abgrenzungen

Aktuell existiert bei Compass Security eine Applikation, welche eine integrierte Einsatzplanung hat. Diese wird für die Zeiterfassung sowie das Erfassen und Planen der Projekte verwendet. Die Einsatzplanung dieser Applikation macht jedoch das Planen der Projektzuweisungen sehr mühselig. Hier einige Praxisbeispiele der aktuellen Software:

- Wenn Mitarbeiter auf ein Projekt eingeplant werden, muss dies über Formulare gemacht werden
- Das Erstellen einer Grafischen Übersicht (siehe Abbildung 1) dauert mehrere Minuten und zeigt nur einen gewünschten Zeitbereich an
- Konflikte in der Planung können erst in der grafischen Übersicht erkannt werden

Daher will Compass Security dies zukünftig in einem neuen Einsatzplanungstool erledigen.

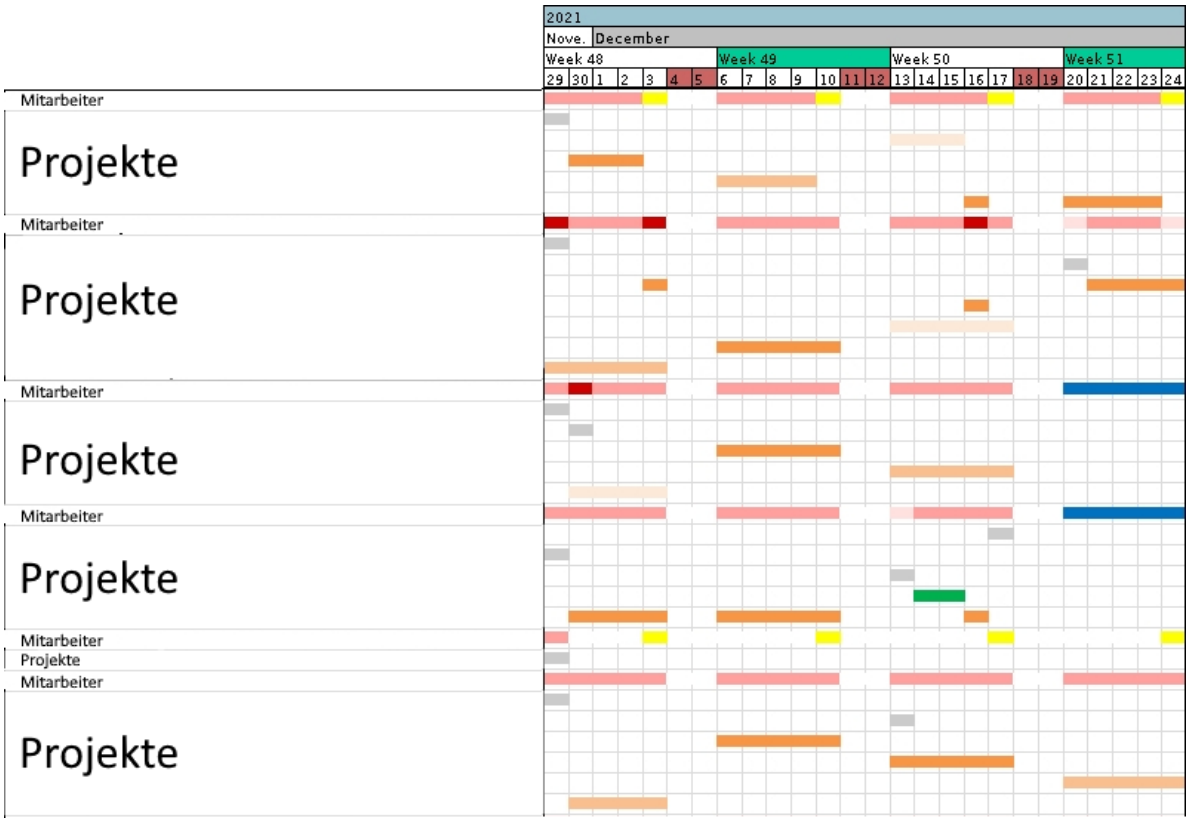


Abbildung 1: Screenshot Compass aktuelle Software (Quelle: Cyrill Brunschwiler)

Da die Zeit- und Projekterfassung jedoch weiterhin in der bestehenden Software gemacht wird, soll unsere Applikation Primärdaten wie Mitarbeiter, deren Absenzen und Ferien sowie Projektinformationen in einem definierten Rhythmus importieren.

1.1.1 Ähnliche Produkte / Konkurrenz Lösungen

Es gibt diverse Projektplanungstools auf dem Markt. Die meisten Lösungen fokussieren sich auf Kanban Boards oder Gantt-Diagramme ohne Mitarbeiter-Zuweisungen. Viele der grossen Planungstools sind eher

als Standard-Lösungen umgesetzt, damit diese ein breiteres Kundenfeld abdecken können, im Gegensatz zu dieser der Compass Security zugeschnittenen Einsatzplanung.

Hier einige Projektplanungstools:

- Factro [1]
  - sehr vielseitig und bietet viele Funktionen an
  - Eingebaute Zeiterfassung
- Trello [2]
  - spezialisiert auf Agile Teams
  - einfach erweiterbar
- Microsoft Planner [3]
  - spezialisiert für Teams die Kanban verwenden
  - mit Office vernetzt
- OpenProject [4]
  - Open-Source
  - umfangreiche Funktionen

Diese und weitere Projektplanungstools sind in folgenden Rezensionen aufgeführt:
https://www.fuer-gruender.de/blog/projektmanagement-tools/
https://www.factro.de/blog/projektmanagement-software-vergleich/

1.2 Vorgehen, Aufbau der Arbeit

Im Rahmen unserer Arbeit war unser erstes Ziel und gleichzeitiger Meilenstein die Erhebung der Anforderungsanalyse. Im Austausch mit Cyrill Brunschwiler haben wir seine fachlichen Anforderungen aufgenommen, technisch ergänzt und überarbeitet und schliesslich wiederum von ihm verifizieren lassen. Nach dem Abschluss der Anforderungsanalyse wurden nötige Vorbereitungen für die Entwicklung getroffen, darunter z.B. Repositories erstellen, CI Builds konfigurieren und Libraries analysieren.

Anschliessend wurden Meilensteine wie Entwurf (Architekturprototyp), Implementation & Test (MVP) und Dokumentation (finale Abgabe) definiert und geplant.

Nach einer erfolgreichen Demonstration unseres Architekturprototyps haben wir uns auf die Entwicklung der Features konzentriert, welche im Kapitel «4.6 MVP» verlangt werden. Wir mussten dann, aufgrund von Verzögerungen im Backend (siehe Kapitel «10.6.1.1 Neue Technologien»), einige Features aus dem MVP streichen, um sicherzustellen, dass alle entwickelten Features sauber getestet und dokumentiert sind.

Neben den Tests im Code wurden mit einigen Mitarbeiter der Compass Security ein Usability Test durchgeführt und die Ergebnisse davon im Kapitel «8 Usability Tests» dokumentiert.

Zum Abschluss haben wir alle erledigten und noch offenen Features dokumentiert, sodass die Person, welche die Weiterentwicklung übernimmt, eine möglichst einfache Arbeitsübernahme hat.

2. Umsetzungskonzept

2.1 Grundstruktur

Unsere Software besteht aus drei Komponenten: einem Frontend, einem Backend und einer Datenbank.

Frontend

Das Frontend stellt den Inhalt der Software dar. Es ist auf die Bedienung mit einer Maus ausgelegt, da viele Komponenten mittels Drag & Drop verändert werden können. Zusätzlich wird im Frontend eine Anmeldung der Compass Security vorausgesetzt.

Das Frontend wurde mit Angular 12 entwickelt.

Backend

Das Backend verwaltet alle Daten der Datenbank und bereitet diese so auf, dass das Frontend damit arbeiten kann. Zusätzlich bietet das Backend sogenannte WebSockets an, um Live-Updates an das Frontend zu senden und des Weiteren werden Daten der bereits bestehenden Software importiert.

Das Backend wurde mit dem Java Play Framework entwickelt.

Datenbank

Die Datenbank persistiert die Daten, welche unsere Software verwendet.

Es wurde eine MySQL Datenbank verwendet.
Weitere Details zur Datenbank können im Kapitel «6 Architektur & Design» gefunden werden.

2.2 Wesentliche Komponenten / Features

Die kommenden Kapitel gehen, oberflächlich, auf die wichtigsten Komponenten unserer Software ein. Weitere Komponenten und genauere Details sind im Kapitel «7 Implementation» beschrieben.

2.2.1 Eingabe mit der Maus

Die komplette Planung auf der Webseite kann mit der Maus erstellt werden. Entsprechend können viele Elemente mittels Drag & Drop verändert oder erstellt werden. So können zum Beispiel Projekte an Mitarbeitern zugewiesen werden oder die Dauer von einzelnen Planungen angepasst werden.

Um einen Mitarbeiter einem Projekt zuzuweisen, kann aus der Projektliste oder aus den angepinnten Projekten, ein Projekt gezogen («drag») werden und auf dem Planungsraster an der gewünschten Position losgelassen («drop») werden.
Die neuerstellte Planung kann dann wiederum mit Drag & Drop verlängert oder verkürzt werden.

2.2.2 Echtzeit-Updates

Eine der wichtigsten Anforderungen an die Software war, dass mehrere Benutzer zeitgleich an der Einsatzplanung arbeiten können. Somit sollen Benutzer die Änderungen anderer Benutzer in Echtzeit erhalten. Dies wurde in unserer Software mit WebSockets gelöst.

Wenn ein Benutzer die Webseite öffnet, verbindet sich das Frontend mit dem WebSocket des Backends (1). Wenn nun ein Benutzer etwas an seiner Planung verändert, wird diese Änderung über die Schnittstelle dem Backend mitgeteilt (2). Das Backend führt die Änderungen an den Daten durch und sendet dann die angepassten Daten über den WebSocket an alle verbundenen Benutzer (3).
Die «Abbildung 2: WebSocket Beispiel» dient als Veranschaulichung des obigen Beispiels.

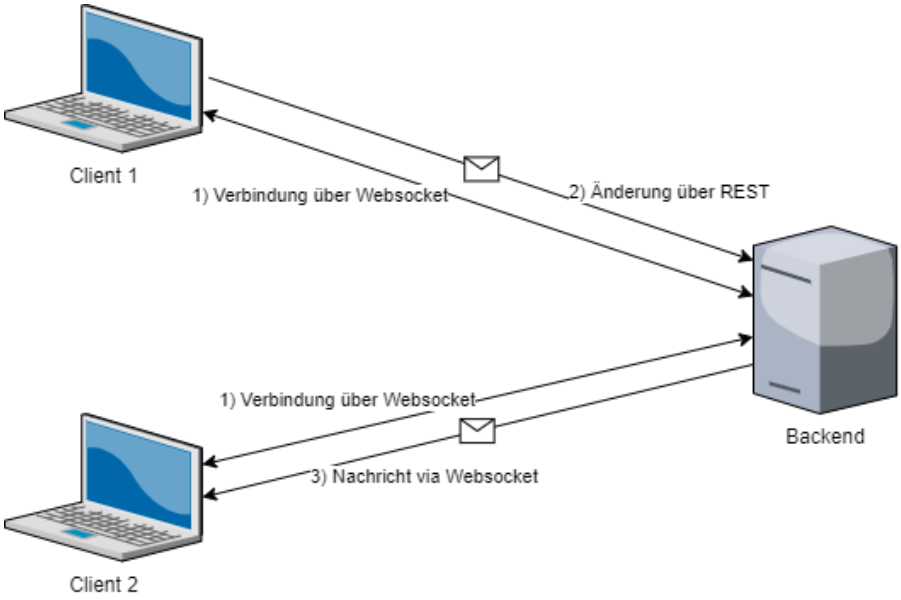


Abbildung 2: WebSocket Beispiel

2.2.3 Stundenberechnung

Da die Compass Security auf ihren Kundenprojekten eine Soll-Zeit haben, müssen die bisherig geplanten Stunden berechnet werden. Dies wurde mit einem Algorithmus gelöst.
Dieser berechnet die geplanten Stunden eines Projektes und beachtet hier auch, ob ein Mitarbeiter gleichzeitig an zwei Projekten arbeitet oder nicht. Ist dies der Fall, kann nur mit der Hälfte der Zeit gerechnet werden.
Kurz gesagt, ein Mitarbeiter, der heute nur am Projekt X arbeitet, wird vermutlich 8 Stunden bzw. einen Arbeitstag daran arbeiten. Wiederum ein anderer Mitarbeiter, der heute an den Projekt X und Y arbeiten soll, wird voraussichtlich 4 Stunden bzw. einen halben Arbeitstag pro Projekt arbeiten.
Da in diesem Algorithmus nur mit Planungen gearbeitet wird und nicht mit dem effektiven Aufwand, reicht die Annahme, dass ein Mitarbeiter entweder vollständig an einem Projekt oder zur Hälfte an zwei Projekten arbeitet.

3. Resultate und Ausblick

3.1 Zielerreichung

Im Rahmen dieser Arbeit konnten wir der Compass Security einen Prototyp der geforderten Webapplikation bereitstellen. Dieser Prototyp deckt so gut wie alle geforderten Anforderungen des MVPs ab. Die nicht erfüllten Anforderungen des MVPs (siehe Kapitel «9.1.1 Resultate funktionaler Anforderungen») wurden in Absprache mit Cyrill Brunschwiler tiefer priorisiert als Anforderungen, welche während der Umsetzung entstanden sind.
Einige nicht funktionale Anforderungen wurden ebenfalls tiefer priorisiert, damit wichtige Anforderungen im Frontend abgeschlossen werden konnten.

Genauer Resultate zum Umsetzungsgrad der funktional und der nicht-funktionalen Anforderungen werden im Kapitel «9.1 Resultate» behandelt.

3.2 Schlussfolgerungen und Ausblick

Als Resultat dieser 14-wöchigen Studienarbeit entstand eine sogenannte «customized»-Webapplikation, was so viel heisst wie, dass diese Software stark auf die Anforderungen und Prozesse der Compass Security zugeschnitten wurde. Daher kann diese Webapplikation schlechter mit den bekannten Softwarelösungen für Projektmanagement (siehe Kapitel «1.1.1 Ähnliche Produkte / Konkurrenz Lösungen») verglichen werden.

Dennoch kristallisieren sich einige Parallelen zu anderen Projektmanagement-Lösungen heraus.
Zum Beispiel setzen einige dieser Tools stark auf Drag & Drop, welches sich bei uns ebenfalls als die aktuelle Hauptinteraktion innerhalb der Benutzeroberfläche abzeichnet. Als weiteres Beispiel zeigt sich die verblüffende Ähnlichkeit von der Darstellung des Zeitstrahls.

Als abschliessendes Feedback von Cyrill Brunschwiler haben wir folgendes erhalten:

«Unsere wichtigsten Bedürfnisse wurden in dieser Studienarbeit sehr gezielt und effizient umgesetzt. In der Rolle des Auftraggebers fühlte ich mich zu jeder Zeit sehr gut abgeholt und beraten. Abdullah und Ursin verfügen über Fähigkeiten, die über das reine Engineering hinausgehen und die unter Ingenieuren ihresgleichen suchen.»

3.2.1 Weiterentwicklung

Zum Abschluss dieser Arbeit werden die Software und deren Code für die Weiterentwicklung an Compass Security übergeben.
Wir haben der Compass Security ebenfalls zugesagt, nach Abschluss der Arbeit dediziert Unterstützung zu leisten, falls bei der Weiterentwicklung Fragen entstehen sollten.

Detaillierte und technische Anhaltspunkte zur Weiterentwicklung werden im Kapitel «9.2 Weiterentwicklung» aufgeführt.

3.2.2 Gemeinsames Fazit

Im Grossen und Ganzen sind wir sehr zufrieden mit unserer erbrachten Leistung.

Wir haben im Frontend die beiden Libraries «Nx» und «NgRx» verwendet, um diese primär kennenzulernen. Wir sind begeistert von deren Mehrwert. «Nx» bringt als Erweiterung von Angular viele nützliche Funktionen, wie z.B. Codegenerierung und ermöglicht schnellere Build- und Testzeiten. NgRx ermöglicht uns sogenannte «Stores» zu erstellen, mit denen wir alle Daten an einer zentralen Stelle lesen und bearbeiten können. Dadurch entstand ein sauberer und verständlicher Code. Wir werden diese beiden Libraries in zukünftigen Angular-Projekten sicherlich wiederverwenden, sofern deren Verwendung Sinn macht.

Was wir bei einem nächsten Projekt jedoch anders machen würden, wäre auf weniger verbreitete Frameworks zu verzichten, bzw. in Absprache mit dem Kunden diese technische Beschränkung aufheben.

Durch «trial and error» ging viel unserer produktiven Zeit verloren, da nicht jeder Anwendungsfall des uns vorgesetzten «Play Frameworks» dokumentiert ist (siehe Kapitel «10.6.1 Eingetretene Risiken»). Unserer Meinung nach ist die Community des Frameworks eher beschränkt in ihrer Grösse und des Weiteren ist diese bereits kleinere Community noch in die Programmiersprachen Java und Scala gespalten, da das Framework in beiden Sprachen verwendet werden kann.

3.3 Dank

Wir möchten uns hier bei allen Personen bedanken, welche uns während der Studienarbeit unterstützt haben. Speziell möchten wir Markus Stolze danken für die Unterstützung während der gesamten Arbeit. Zudem bedanken wir uns bei der Compass Security, namentlich Cyrill Brunschwiler, Ivan Büttler, Ivano Somaini und Erwin Moro für die gute Zusammenarbeit sowie bei Michael Gfeller für die Code-Reviews und die Feedbacks zur Optimierung unserer Software.
Zu guter Letzt möchten wir uns bei Sara Varela und Daniel Häfliger für das Korrekturlesen unserer Arbeit bedanken.

Teil 2: SW-Projektdokumentation

In den nachfolgenden Kapiteln wird das Einsatzplanungstool auch mit dem technischen Namen «1337Planner» bezeichnet.

4. Anforderungsspezifikation

4.1 Rollen

Unsere User Stories enthalten die beiden Rollen «Planer» und «Analyst».

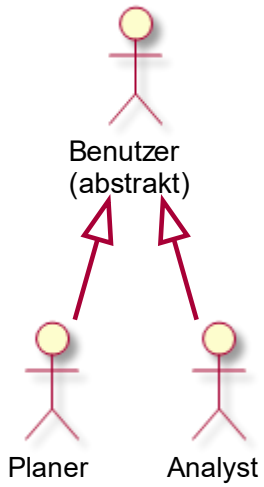


Abbildung 3: Rollen der User Stories

Der «Benutzer» ist eine abstrakte Rolle, welche niemand innehaben kann.
Diese Rolle soll garantieren, dass jede konkrete Rolle authentifiziert ist.

Die beiden Rollen «Planer» und «Analyst» repräsentieren die beiden Autorisierungen.

4.2 User Stories

Zum Umfang des MVP gehören, die mit den roten Sternchen (*) markierten Akzeptanzkriterien.

4.2.1 User Story-1

Als Benutzer
möchte ich mich mit meinem Compass Security-Account über SSO anmelden können,
um mich nicht mehrmals anmelden zu müssen.

Technische Akzeptanzkriterien:

1. *Die Applikation leitet nicht-authentifizierte Benutzer auf den IDP von Compass Security weiter.
2. *Die Applikation liest aus den Claims des Tokens, welches vom IDP der Compass Security kommt, die Rolle des Benutzers aus.

4.2.2 User Story-2

Als Planer
möchte ich die Planung, An- und Abwesenheiten von den Mitarbeiten einsehen können,
um dann eine Planung zu erstellen.

Akzeptanzkriterien:

1. *Planer hat Einsicht auf die Zuteilungen von den Mitarbeitern zu Projekten.
2. *Alle Ferien, Feiertage sowie sonstige Absenzen von allen Mitarbeitern können vom Planer eingesehen werden.
3. *Die Planung von mindestens 5 Wochen können auf einem Überblick eingesehen werden.
4. *Ein Planer kann Standorte ein- und ausblenden.
5. *Aktuellen Projekte und deren Status sind für nicht farbenblinde Personen farblich unterscheidbar.
6. Favoriten werden standardmässig auf der Ansicht aufgeklappt.

4.2.3 User Story-3

Als Planer
möchte ich eine Übersicht zur Erstellung von Planungen mit Zuweisungen von Projekten und Mitarbeitern,
um eine Planung erstellen zu können.

Akzeptanzkriterien:

1. *Es kann nach einem gewünschten Projekt gesucht werden.
2. *Projekte können gepinnt werden und bleiben ersichtlich.
3. *Planer können Mitarbeiter maximal zwei Projekten parallel zuweisen.
    a. Es werden immer zwei Zeilen pro Mitarbeiter angezeigt.
    b. Die Planungszeit von parallel geplanten Projekten beträgt die Hälfte der Zeit.
    c. Die Zeile, in welcher die Planung erstellt wurde, soll persistiert werden.
4. *Planer können ihre Mitarbeiter einem Projekt auf einen halben Tag genau zuweisen.
5. *Planer können Planungsblöcke erstellen und deren Länge anpassen.
6. *Planer können Planungsblöcke anderen Mitarbeiter zuweisen via Drag & Drop.
7. *Projekte, deren Ist-Plan-Zeit kleiner ist als deren Soll-Plan-Zeit, werden dem Planer angezeigt.
8. *Planungskonflikte können erkannt werden:
    a. Verplante Zeit eines Projekts überschreitet oder erfüllt nicht dessen Soll-Zeit.
    b. Überschneidung von Planung und Absenz eines Mitarbeiters.
9. *Die Datenanpassungen von mehreren Planern soll in Echtzeit synchronisiert werden.
   D.h. Wenn Planer A eine Änderung an den Daten vornimmt, sieht Planer B diese sofort, ohne einen die Seite neuzuladen.
    a. Bei Problemen bei der automatischen Speicherung oder Synchronisation, wird eine entsprechende Meldung angezeigt.
    b. Bei erfolgreicher Synchronisation soll dem Benutzer ein Hinweis angezeigt werden.

Optionale Features:

1. Nicht eingeplante Projekte können eingesehen werden.
2. Daten vom Planungstool können exportiert werden.
3. Projekte können erfasst werden.

4. Planer können erkennen, welche anderen Planer gerade welche Projekte und Mitarbeiter verplanen.
5. Planungen können direkt erstellt werden, ohne vorher ein Projekt auszuwählen.

4.2.4 User-Story-4

Als Planer
möchte ich, dass die Daten wie Absenzen, Feiertage, Mitarbeiter und Projekte aus dem Primärsystem in einem bestimmten Zeitintervall importiert werden,
um in der Anwendung mit den aktuellen Daten zu arbeiten.

Technische Akzeptanzkriterien:

1. *Die csv-Dateien werden im Backend über einen Fileshare ausgelesen und in die Datenbank importiert bzw. synchronisiert
    a. Das Intervall des Pollings und der Fileshare Pfad, Benutzer und Passwort sollen konfigurierbar sein.
    b. Ferien, Absenzen und Feiertage werden bei jedem Import gelöscht und überschrieben, da diese über keinen Primärschlüssel verfügen.

4.2.5 User Story-5

Als Analyst
möchte ich meine persönliche Planung und die dazu relevanten Informationen sehen,
um mir einen Überblick über meine Planung zu verschaffen.

Akzeptanzkriterien:

1. Einsicht auf persönliche Planung.
    a. Welche Projekte wurden mit zugeteilt.
    b. Wann sind welche Mitarbeiter in den gleichen Projekten eingeteilt.
2. Die Projektliste ist sortiert nach Startdatum.
3. Es ist klar visualisiert, in welchen Projekten man Projektmanager ist.
4. Man erhält eine Übersicht von den Absenzen ab heute. Diese Übersicht ähnelt einer Statistik. (Absence of today / future absences)
5. Man erhält eine Übersicht der Absenzen als Liste. (Termine)

4.3 Nicht-funktionalen Anforderungen (NFR) & Qualitätsattribute

Wir verwenden als Grundlage unserer NFRs den ISO 25010 Standard [5].
Wir gehen nur auf die, für unser Projekt relevanten Bereiche, des ISO Standards ein.

Die Ergebnisse der Tests sind im Kapitel «9.1.2 Resultate nicht-funktionaler Anforderungen» aufgeführt.

4.3.1 Usability

Der 1337Planner soll für neue Benutzer verständlich gemacht werden mit Hilfe einer Anleitung.
Anwender des alten Systems sollen sich ohne Anleitung zurechtfinden.

Nummer	Akzeptanzkriterien	Tests
NFR1	Ein komplett neuer Nutzer soll sich nach dem Lesen der Benutzeranleitung im 1337Planner zurechtfinden können.	Usability Test mit 2 neuen Benutzer, welcher zuerst eine Demonstration erhält. (siehe «8 Usability »)
NFR2	Ein Nutzer, der mit der alten Planung gearbeitet hat, findet sich im 1337Planner zurecht.	Usability Test mit 2 Benutzern, welche bereits mit der alten Software gearbeitet haben. (siehe «8 Usability »)

4.3.2 Security

Es sind Sicherheitsmassnahmen vorhanden, welche vor bekannten Attacken schützen.

Nummer	Akzeptanzkriterien	Tests
NFR3	Die Applikation und API sind geschützt vor «OWASP Top 10 - 2021»-Schwachstellen [6].	Compass Security führt nach Projektabschluss Penetration Tests auf dem 1337Planner durch. Typische Gegenmassnahmen für die «OWASP Top 10 - 2021» sind vor Projektabschluss vorhanden.

Falls nach den Penetration Tests dringend Anpassungsbedarf besteht, wird gemäss Cyrill Brunschwiler, die Compass Security die nötigen Anpassungen selbst vornehmen.

4.3.3 Portability

Die jeweiligen Softwarekomponenten des 1337Planner's sollen unabhängig voneinander laufen, so dass sie ausgetauscht oder wiederverwendet werden können.
Die Software soll plattformunabhängig funktionieren.

Nummer	Akzeptanzkriterien	Tests
NFR4	Das Planungs-Widget soll unabhängig von anderen Komponenten funktionieren, so dass es wieder verwendet werden kann.	Ein Entwickler, welcher sich mit Angular auskennt, ist in der Lage das Planung Widget anzuwenden ohne Erklärung.

4.3.4 Maintainability

Der MVP bietet eine hohe Testabdeckung.

Nummer	Akzeptanzkriterien	Tests
NFR5	Die Unittests bieten eine Code-Abdeckung von 90% auf Methoden und Codezeilen.	Die Unittests haben eine Code-Abdeckung von 90% auf Methoden und Codezeilen.
NFR6	Alle Unit Tests sind OK	Alle Unit Tests sind OK

NFR7	Integrationstests wurden erstellt und sind OK	Integrationstests wurden erstellt und sind OK

4.3.5 Performance

Die Webapplikation soll bei Änderungen von Zeitbereichen und beim Speichern nicht blockieren und schnell reagieren. Es soll ein Gefühl entstehen, dass die Applikation schnell reagiert und nicht stockt.

Nummer	Akzeptanzkriterien	Tests
NFR8	Nach Anpassungen des Zeitbereichs, werden die neuen Daten innerhalb von maximal 1 Sekunde angezeigt.	Nutzer beurteilen die Performance beim Anpassen des Zeitbereichs während eines Usability Tests als angenehm und passend.
NFR9	Nach einer Planungsanpassung durch einen anderen Benutzer, werden dessen Änderung bei mir nach maximal 3 Sekunden sichtbar, ohne die UX zu stören.	Während eines Usability Tests empfinden Tester das asynchrone Synchronisieren der Daten als nicht-störend.

4.4 Technische Beschränkungen

Es sollen folgende Technologien für die Entwicklung der jeweiligen Schichten verwendet werden:

Frontend	Angular
Backend	Java Play Framework
Datenbank	MySQL

Des Weiteren soll das ganze System in Docker-Containern laufen.

Die Authentifizierung und Autorisierung läuft über den IDP von Compass Security.
Die Berechtigungen werden von Compass Security verwaltet.

4.5 Entscheidungen zu UI & UX

1. Durch die Vorgabe, dass Projekte mit zufälligen Farben und deren Status, Absenzen und Ferien mit vordefinierten Farben gekennzeichnet werden, wurde entschieden, dass die Accessibility für Farbenblinde vorerst nicht gewährleistet werden soll.
    a. Um zu verhindern, dass die ganze Applikation durch viel grelle Farben unüberschaubar wird, wurde entschieden nur die gepinnten Projekte einzufärben. Dafür wurden 10 relativ helle und dezente Farben verwendet, welche sich nicht mit den vordefinierten Farben der Compass Security überschneiden.
2. Da die angeforderte Bedienung des Frontend primär über die Maus (Drag & Drop) stattfindet, wurde entschieden im Rahmen dieser Arbeit keine Accessibility für Benutzer, welche nur mit der Tastatur arbeiten, zu gewährleisten.

3. Die Applikation soll und wird nur auf Desktops verwendet, darum wird auf eine mobile Ansicht verzichtet. Ein flexibles Layout wird dennoch umgesetzt.
4. In der Benutzeranleitung wird im Verlauf des Projekts festgelegt, welche minimale Bildschirmbreite notwendig ist, um den Zeitbereich von 5 Wochen einzusehen. (User Story 2 / AK-3)
    a. Die meisten Benutzer in der Compass Security besitzen einen Screen mit der Auflösung: 2560x1440.
5. Die Standard-Tastenkombination «CTRL + F» soll übersteuert werden und das Inputfeld für die Projektsuche fokussieren.

4.6 MVP

Im Endprodukt sollen mindestens folgende Akzeptanzkriterien erfüllt sein:

User Stories	**User-Story 1**	**User-Story 2**	**User-Story 3**	**User-Story 4**
Akzeptanzkriterien	1 bis 2	1 bis 5	1 bis 9	1

Diese sind im Kapitel «4.2 User Stories» mit einem roten Sternchen (*) hervorgehoben.

Nachdem der MVP umgesetzt worden ist, sind folgende Features in der gegebenen Reihenfolge zu priorisieren:

1. Teil der Akzeptanzkriterien aber nicht des MVP
    a. Favoriten (User Story-2 / AK 6-7) und Analyst Rolle (User Story-5 / AK 1-5)
2. Optionale Features oder spätere Wunsch-Features
    a. Diese werden vor einer Umsetzung nochmals mit Cyrill Brunschwiler priorisiert.

4.7 Weitere Anforderungsentscheidungen

Nach Absprache mit Cyrill Brunschwiler wurden folgende Entscheidungen für die Anforderungen gefällt:

1. Im Rahmen des Projekts wird keine Benutzeranleitung erstellt.
2. Eine explizite Entwickleranleitung wird nicht benötigt, solange jedes Software-Repository ein ausführliches Readme enthält, welches jegliche Informationen beinhaltet, um die eigene Entwicklungsinfrastruktur aufzusetzen.

5. Analyse

5.1 Domain Model

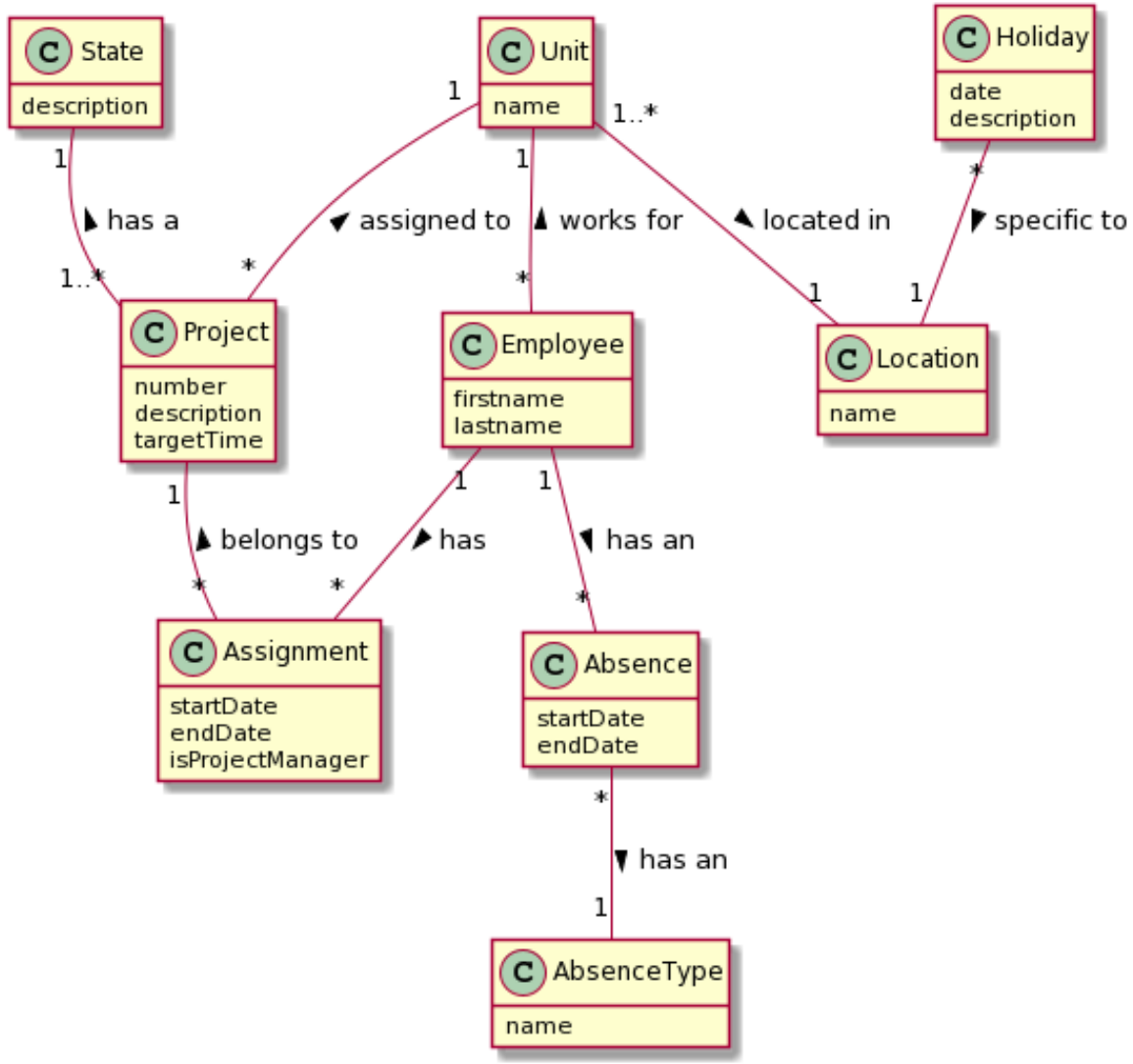


Abbildung 4: Domain Model

Spezifische Ausführungen zu einzelnen Feldern des Domain Models können im «6.2 Datenbank Model» eingesehen werden.

6. Architektur & Design

6.1 Grobübersicht der Infrastruktur

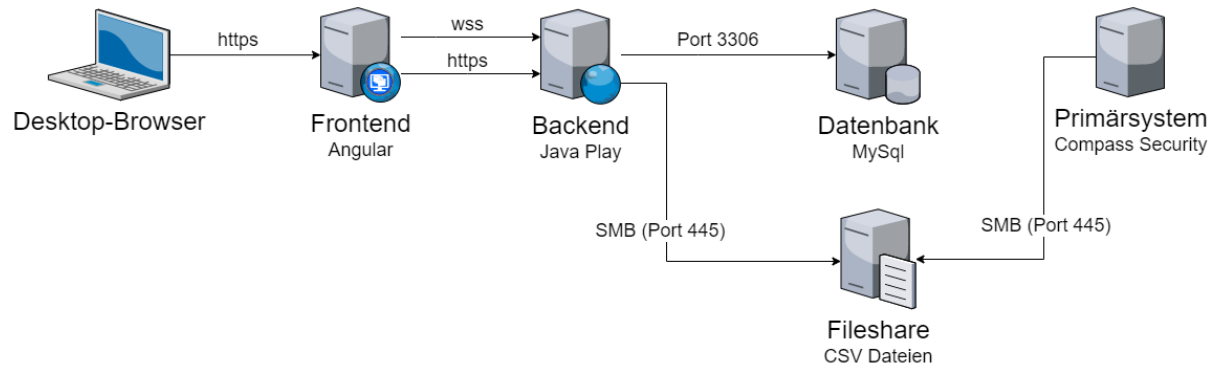


Abbildung 5: Architektur Übersicht

6.2 Datenbank Model

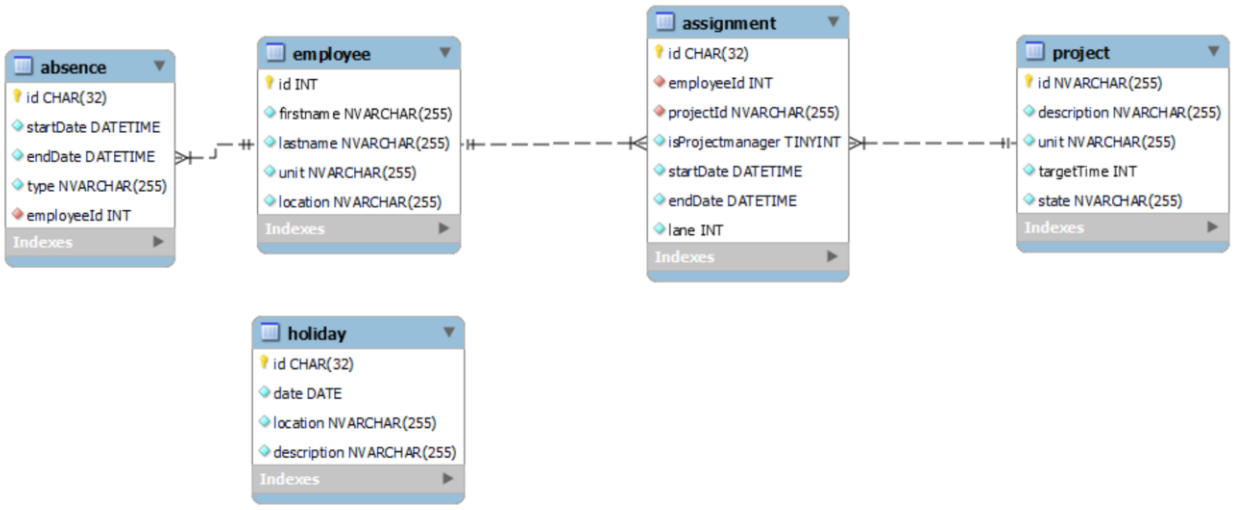


Abbildung 6: Datenbank Modell

Absence:
In der Absence Tabelle befinden sich alle Absenzen der jeweiligen Employees. Diese werden importiert. Eine Absenz hat einen Typ, sowie ein Start- und End Datum.

Employee:
In der Tabelle Employee werden alle Mitarbeiter gespeichert mit ihren jeweiligen Ids, Namen, Standorten und Abteilungen.

Assignment:
In der Tabelle Assignment werden alle Zuweisungen der Mitarbeiter auf Projekte eingetragen. Falls der zugewiesene Mitarbeiter Manager des zugewiesenen Projektes ist, wird das isProjectmanager Feld auf

true gesetzt. Und in Lane wird gespeichert in welcher Lane der beiden Lanes eines Mitarbeiters die Zuweisung gemacht wurde.

Project:
Die Tabelle Project beinhaltet alle vorhandenen Projekte. Projekte haben eine Nummer welche als ID gespeichert werden. Eine kurze Beschreibung um welches Projekt sich handelt. Einer Target Time welche die erwartete Anzahl an benötigte Arbeitstage sind. Eine Unit, in der zusätzlich die Währung, mit welcher das Projekt bezahlt wird, eingetragen ist und einen State welcher den aktuellen Stand des Projekts beschreibt.

Holiday:
In Holiday werden alle Feiertage des jeweiligen Standorts gespeichert. Da wir die Standorte nicht in einer eigenen Entität abbilden, hat sie keine Beziehung zu anderen Tabellen.

Weitere Anmerkungen:
Da das Favoriten-Feature nicht Teil des MVPs ist, wurde in der ersten Datenbankversion keine Tabelle für Favoriten erstellt.

Die Daten der Tabellen Absence, Holiday, Employee und Project werden importiert. Die Daten der Tabellen Absence und Holiday haben keine vordefinierte ID und da diese Datensätze immer wieder gelöscht werden beim Import werden für diese eine GUID generiert.

Da die Standorte nur in Form von Text im Employee existieren, können die Beziehung nicht äquivalent zum Domain Model erzeugt werden. Deswegen besitzt z.B. die Tabelle «Holiday» keine Beziehungen zu anderen Tabellen und wird erst im Frontend mit den Standorten verknüpft und angezeigt.
Es wurde entschieden die Verbindung zum Standort im Frontend zu lösen statt beim Import, weil nicht garantiert werden konnte, dass sich der Name der Standort nicht im Primärsystem ändern wird. Dadurch wird eine zu starke Abhängigkeit zum Primärsystem vermieden.

6.3 Frontend

6.3.1 Frontend Architektur

Im Repository des Frontends, steht die Compodoc Dokumentation zur Verfügung.

- Repository https://github.com/CompassSecurity/1337Planner.Frontend klonen
- Allenfalls den Befehl «npm run compodoc» ausführen, um die Dokumentation zu aktualisieren
- Datei «documentation/index.html» im Browser öffnen

6.3.2 Verzeichnisstruktur

Für die Verzeichnisstruktur wurde die standardmässige Nx Angular Struktur verwendet.

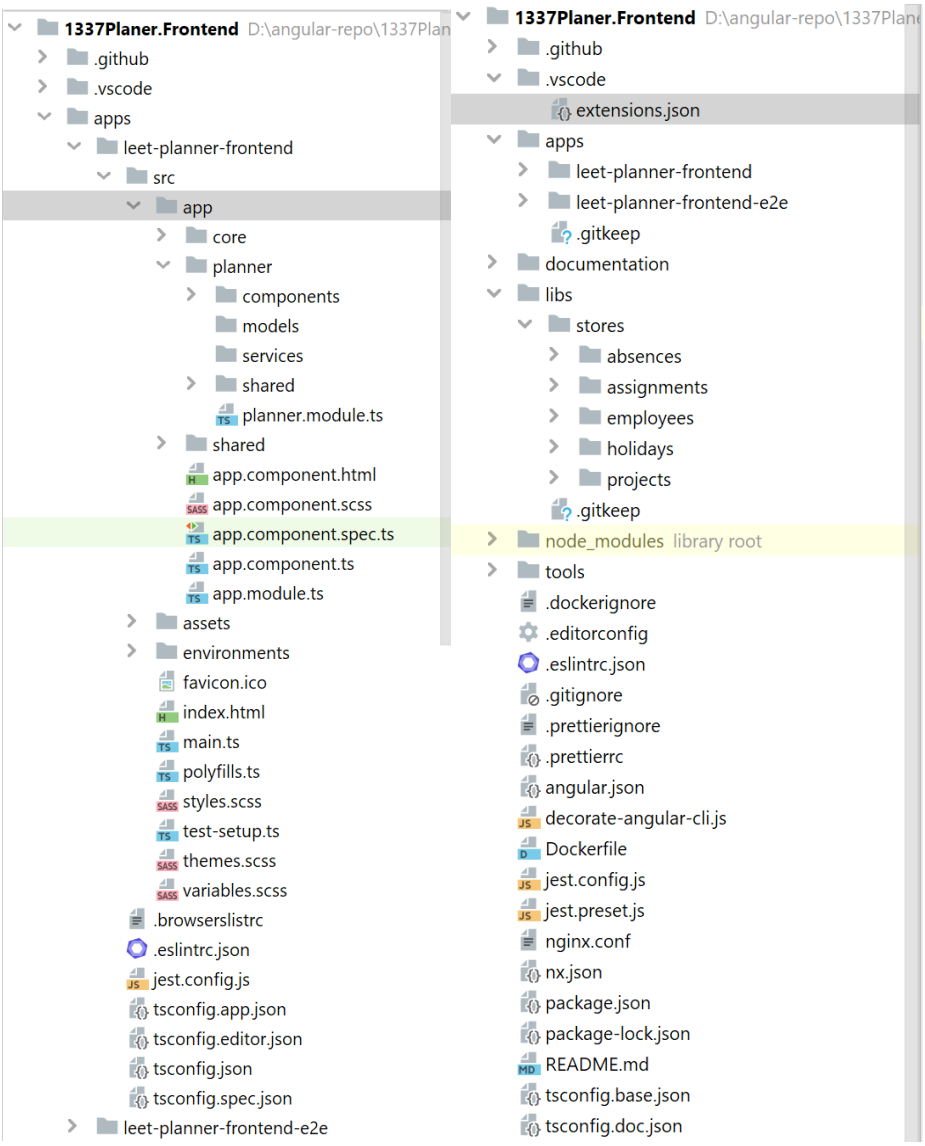


Abbildung 7: Verzeichnisstruktur Frontend

6.3.3 UI Entscheide & Beschränkungen

Viele Libraries und Beispiele verwenden für Ansichten von Gant-Diagrammen entweder «<table>» Attribute oder im CSS «position: absolute».
Wir haben uns dazu entschieden, für die Anzeige des Grids der Planungen keine Library zu verwenden, sondern diese selbst mit CSS Grid zu erstellen, da dieses einerseits sehr performant ist und andererseits durch reines CSS die Positionierung von Elementen im UI ermöglicht wird. (Reihenfolge vom Markup spielt dadurch keine Rolle)

Jedoch bringen CSS Grids in verschiedensten Browsern eine aktuelle Einschränkung mit sich. Aktuell sind wegen Arbeitsspeicherverbrauch auf Browsern nur maximal 1000 Zeilen bzw. 1000 Spalten zugelassen (siehe https://www.w3.org/TR/css-grid-1/#overlarge-grids). D.h. unser Grid kann nur ca. 20 Monate in der Zeitachse anzeigen.

Gemäss Cyrill Brunschwiler reichen bereits 15 Monate (3 Monate in Vergangenheit, 12 Monat in die Zukunft) in der Zeitachse. Daher wird hier kein Workaround benötigt

6.3.4 Libraries & Frameworks

Die Informationen zu den 3rd-Party-Lizenzen des gesamten Frontend befinden sich im Repository im «3rdpartylicenses.txt»

Angular Material	Material Komponenten & CDK wurden verwendet für Elemente wie z.B. Input, Buttons, Coloring, Drag&Drop, usw. Die Benutzung dieser quasi Standard-Library erspart es uns, eine eigene Komponenten Library zu erstellen. https://material.angular.io/ Alternativen: • Microsoft Fluent Design System (https://www.microsoft.com/design/fluent/#/) Die Entscheidung fiel auf «Angular Material», da dieses mehrere von uns benötigte Komponenten mitbringt.
Microsoft Authentication Library	Ist eine Zusammensetzung von mehrere offizielle Libraries von Microsoft, welche die Logik für die Authentifizierung im Azure AD übernimmt bzw. standardisiert ermöglicht. Wir verwenden diese Library, um die ganze Authentifizierung und Autorisierung für die Anforderungen «4.2.1 User Story-1» im Frontend zu ermöglichen. https://www.npmjs.com/package/@azure/msal-angular https://www.npmjs.com/package/@azure/msal-browser
Nx	Nx ist ein Entwicklungsframework. Diese bringt Support für moderne Tools wie Jest, Cypress, NgRx, usw. Zusätzlich verwenden wir das Computation Caching von Nx Cloud.

	Wir haben dieses Framework verwendet, um es kennenzulernen und die gewonnenen Erfahrungen in unserem betrieblichen Umfeld verwenden zu können. https://nx.dev/
NgRx	NgRx ist ein bekanntes Framework für State-Management und wird meist in Enterprise Lösungen verwendet. Wir haben uns entschieden dieses zu verwenden, um dieses einerseits kennenzulernen. Andererseits da wir mit Websocktes arbeiten, um den State bzw. die Daten an einem Ort zu lesen und verarbeiten, ohne eine eigene komplexe Logik fürs State-Management zu schreiben. https://ngrx.io/
Compodoc	Compodoc ist ein Open-Source Dokumentationstool. Mit einfachen Konfigurationen und einem Befehl, kann man sich ganze Dokumentationen von Angular-Applikationen generieren lassen. Wir haben diese Library verwendet, um unser Frontend zu dokumentieren. https://compodoc.app/ Alternativen und Rezensionen: https://www.alternativeten.com/software/alternative/compodoc/ Compodoc ist auf die Dokumentation von Angular-Applikationen ausgelegt, daher wurde entschieden, dieses zu verwenden.
angular-resizable-element	angular-resizable-element ist eine Library, auf die wir während unserer Recherche gestossen sind, welches die Verlängerung von unseren Planungen übernimmt. Damit ist gemeint, dass man z.B. die Länger eines div-Element mit Drag & Drop anpassen kann. (siehe Demo: https://mattlewis92.github.io/angular-resizable-element/) https://www.npmjs.com/package/angular-resizable-element Alternativen: • Eigene Logik schreiben • https://www.npmjs.com/package/angular-resize-element Da angular-resizable-element die Funktionalität mitbringt, welche eine selbstgeschriebene Komponente anbieten müsste, haben wir uns entschieden diese Library zu verwenden.

6.4 Backend

6.4.1 Architektur & Verzeichnisstruktur

Wir verwenden die standardisierte Play Framework Anatomie. Eine detaillierte Beschreibung dazu finden sie auf der Play Framework Dokumentation.

Zusätzlich haben wir weitere Packages hinzugefügt mit Task, Actros und Utils. In Tasks befinden sich alle Klassen, welche im Hintergrund vom Backend aufgerufen werden (Datenimport, löschen alter Assignments). Actors beinhaltet die Logik für den WebSocket. Und im Utils Package ist eine Hilfsklasse welche im Controllers Ordner verwendet wird, um die Antworten der REST Schnittstelle zu vereinheitlichen.

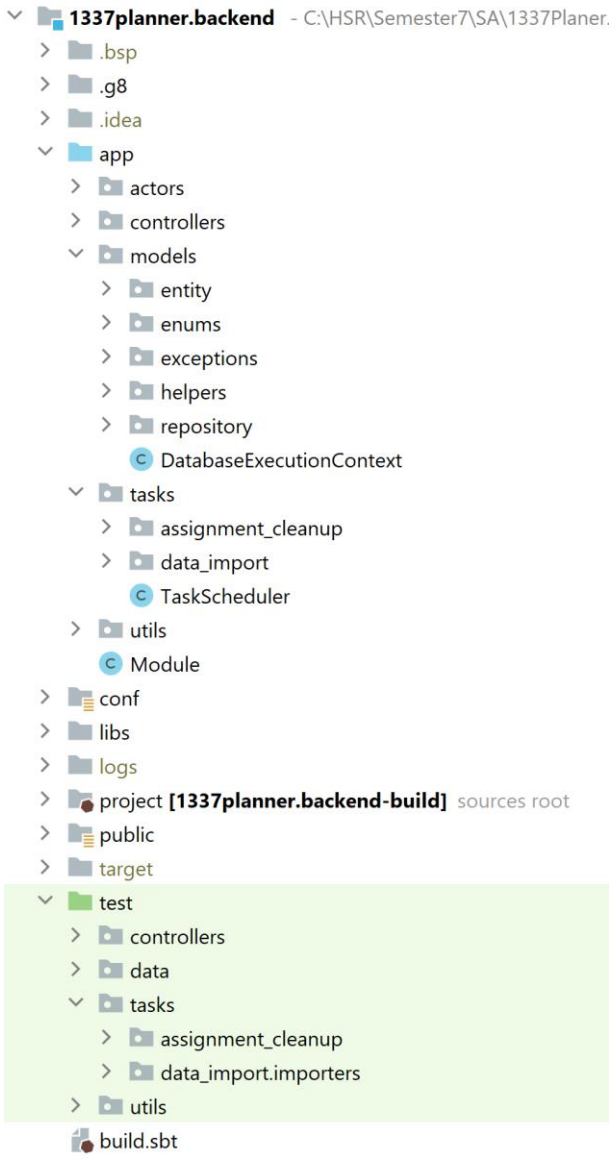


Abbildung 8: Verzeichnisstruktur Backend

6.4.2 Backend Entscheide & Beschränkungen

Für das Deployment der Applikation mittels Docker mussten wir einen umständlichen Workaround umsetzen. Nämlich schreibt ein Entwickler für ein Dockerdeployment ein sogenanntes Dockerfile. Dieses enthält Schritte und Befehle, um aus der Software ein Dockerimage zu erzeugen.
Im Source Code unseres Backends existiert kein Dockerfile. Das Dockerfile wird jeweils in der CI Pipeline erstellt. Dies wird durch den Befehl «docker:stage» über die sbt-shell gemacht. In unserer CI Pipeline wird zuerst dieser Befehl ausgeführt, um das Dockerfile zu erzeugen und danach wird in derselben Pipeline das erzeugte Dockerfile verwendet, um ein Image zu erstellen und dieses auf GitHub Packages hochzuladen.

Für lokale Docker-Images kann man, in der sbt-shell, entweder in der IntelliJ IDE oder der separaten Shell, einfach den Befehl «docker:publishLocal» ausführen.

6.4.3 Libraries & Frameworks

Da sich die Compass Security gewünscht hat, dass wir das Backend mit Hilfe des Play Framework erstellen, waren wir bei den Libraries & Frameworks sehr beschränkt im Backend. Zwar bietet Play viele Funktionalitäten an mit diversen integrierten Libraries. Dies erschwert es jedoch alternative Libraries zu verwenden. Da wir beide keinerlei Erfahrungen mit Play hatten einigten wir uns darauf, dass wir möglichst mit den Empfohlenen Libraries & Frameworks arbeiten.

Zusätzlich kann man Play auch mit Scala verwenden. Das heisst man kann auch mit den Libraries von Scala arbeiten. Da wir aber beide Scala nicht kannten entschieden wir gegen die Verwendung von Scala Libraries im Backend.

Play Framework	Wir verwenden das Play Framework für unser Backend da dies von der Compass Security so gewünscht wurde. Hier einige Alternative für unser Backend, welche wir in Betracht gezogen hätten, wäre uns nicht vorgegeben worden dies mit dem Play Framework umzusetzen: • .NET • Spring
Hibernate mit JPA	Wir verwenden Hibernate als OR-Mapper. Hibernate verwendet JPA, um mit der Datenbank zu kommunizieren. Wir verwenden JPA da es vom Play Framework empfohlen wird. Es gab zwar andere mögliche Datenbankzugriffs-Bibliotheken, mit ScalaQuery und Anorm, diese basieren aber auf Scala.
H2 Datenbank	Wir verwenden eine H2 in-Memory Datenbank für unsere Integrationstests. Diese stellt sicher, dass die Integrationstests nicht nur lokal funktionieren, sondern auch in den GitHub Actions. Als Alternative hätte man auch eine externe Datenbank nehmen können. Jedoch hätte man mit einer externen Datenbank das Risiko, dass sie nicht läuft oder erreichbar ist.

Mockito & JUnit	Wird verwendet, um das Backend zu testen. Beide Libraries werden vom Play Framework empfohlen. Auch gäbe es die Alternative ScalaTest, welche man mit Scala umsetzen müsste.

6.5 Package-Diagramme

Dies ist das Package-Diagramm des Backends. In den leeren Packages Repository und Entity wären noch je fünf Klassen, aber diese wurden leer gelassen da diese repetitiv sind und das Klassendiagram unnötig aufblasen. Es gibt für jeden Controller, ausser den WebSocketController, ein entsprechendes Entity und ein dazugehöriges Repository.
Das Importers-Package ist aus demselben Grund wie eben leer, da dies für alle Entity-Typen ausser Assignment einen Import besitzt. (Absence, Employee, Holiday und Project)
Die Abhängigkeiten der Klassen wurden leer gelassen, um das Klassendiagram übersichtlicher zu machen.

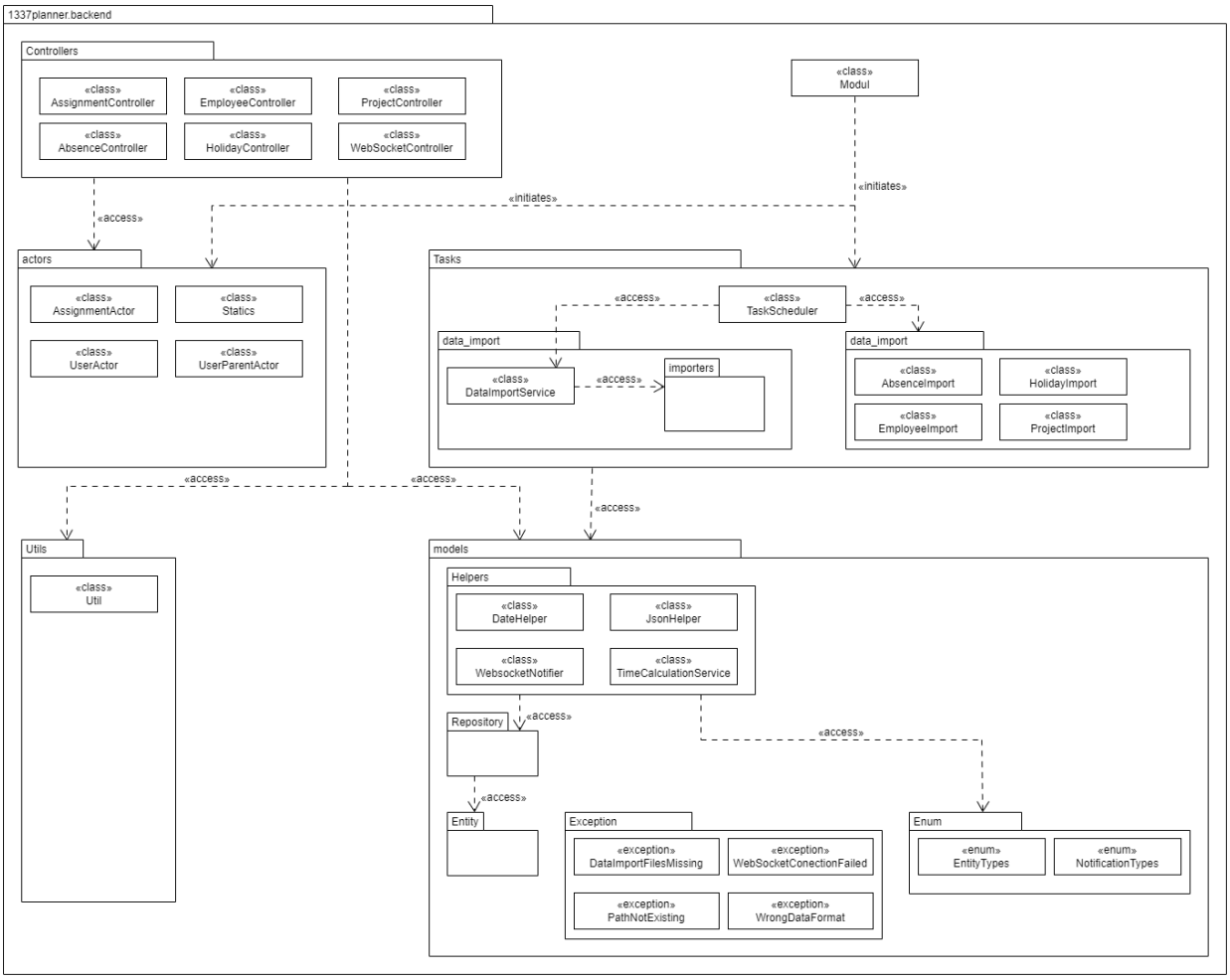


Abbildung 9: Klassendiagramm

7. Implementation

7.1 Implementation

Jegliche Konfigurationsmöglichkeiten des Frontends und des Backends sind im jeweiligen Readme beschrieben.

7.1.1 Frontend

So sieht eine dicht verplante Ansicht des Frontends aus Demodaten aus.

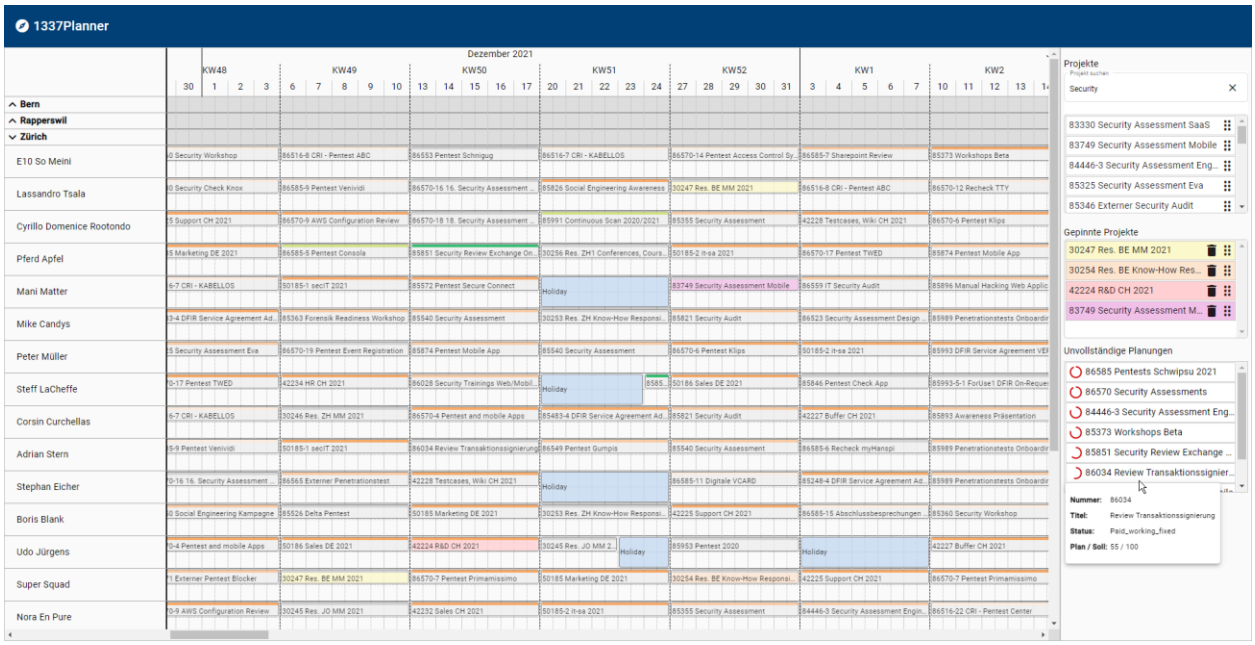


Abbildung 10: Screenshot voll verplante Ansicht

7.1.1.1 Aufbau & CSS-Grids

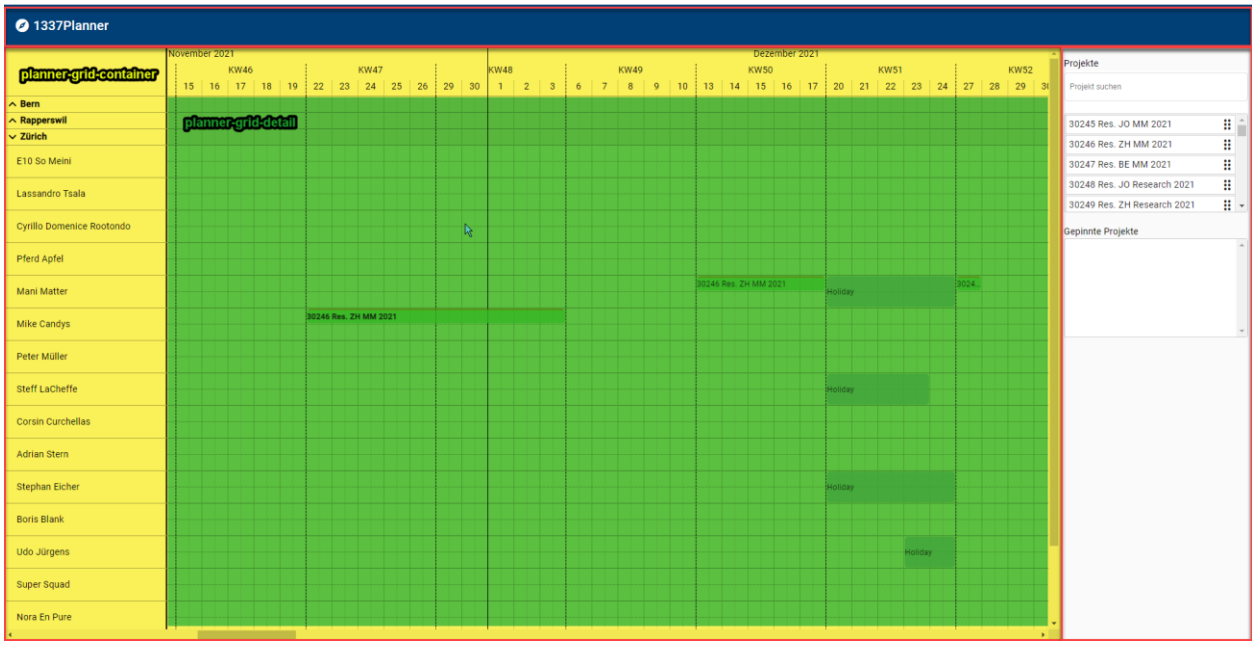


Abbildung 11: Frontend CSS Grids

Unser Frontend besitzt nur eine Ansicht. Diese Ansicht besteht grundsätzlich aus 3 verschachtelten CSS-Grids (keine subgrids).

Auf der «Abbildung 11: Frontend CSS Grids» sind diese 3 Grids farblich gekennzeichnet:

- Das erste Grid (rote Rahmen) dient rein zur Trennung von den 3 Hauptlayout-Komponenten. Diese wären «header», «grid-container» und «sidebar».
- Das zweite Grid (gelbe Fläche) baut die Spalten anhand der Daten und die Zeilen anhand der Mitarbeiter und deren Standort auf.
  Somit ist die einzige Aufgabe dieses Grids, die Berechnung und Darstellung von Spalten und Zeilen.
- Das dritte und letzte Grid (grüne Fläche) ist einerseits zur Entlastung des zweiten Grids, damit eine einzige Komponente nicht sämtliche Logik beinhaltet und andererseits zur Abgrenzung der Funktionalität.
  Im diesem Grid werden nämlich Drag & Drop sowie weitere User-Interaktionen abgehandelt.

7.1.1.2 Store

Für die Entwicklung der Data Stores mit ngrx haben wir uns an den Feature Workflow von Nx orientiert.
https://nx.dev/l/a/guides/misc-ngrx#feature-workflow

Das heisst jede Entität (Project, Assignment, etc.) befindet sich im gleichen Store, aber befinden sich in einzelnen Feature-Modulen.

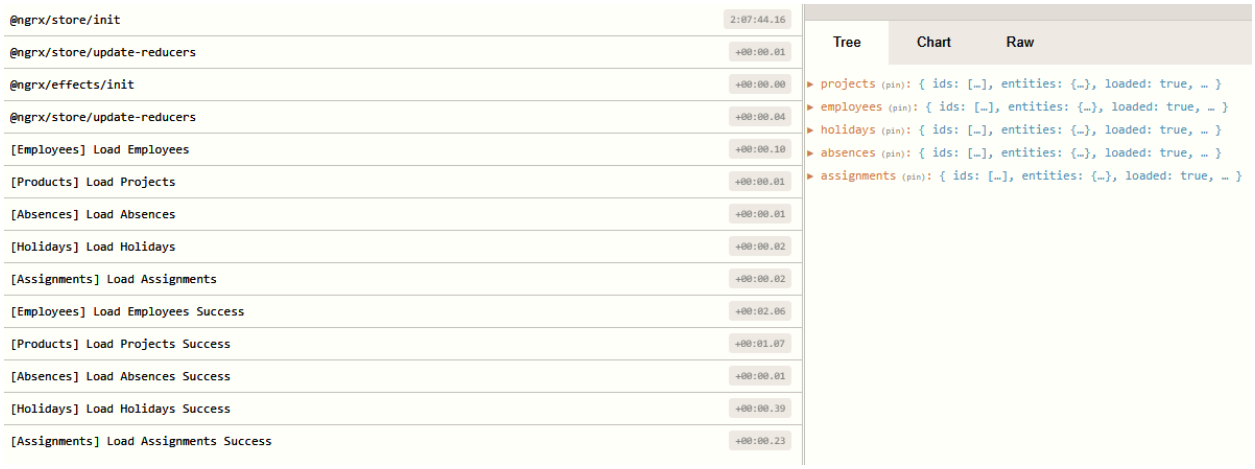


Abbildung 12: Store in den DevTools

Wir haben uns entschieden diese Library und deren Store zu verwenden, da wir Daten sowohl durch Benutzerinteraktionen editieren als auch durch Benachrichtigungen über WebSockets. Dadurch entfällt eine komplexe Logik, um das UI zu notifizieren und das State-Management über die ganze Applikation aufrechtzuerhalten.
Sobald nämlich Daten im Store angepasst werden, bekommen alle Komponenten, die diese Daten verwenden das mit, ohne dass diese speziell notifiziert werden müssen.

Ein weiterer grossen Vorteil, welcher dieser Store mitbringt, ist «optimistic update».
Was folgendes bewirkt:

Ich als Benutzer ändere etwas an den Daten durch z.B. durch Drag & Drop.
Nun werden meine Änderungen sofort in den Store geschrieben und danach anschliessend auf dem Backend angepasst. Dadurch bin ich nicht blockiert und kann weiterarbeiten.
Sollte jedoch der Server eine Änderung nicht vornehmen können oder einen Fehler zurückgeben, wird eine «undo-Action» auf dem lokalen Store ausgeführt, welche mir die Änderung verwirft und den Fehler anzeigt.
Ein Benutzer kann von sich aus, keine Änderungen rückgängig machen.

7.1.1.3 Drag & Drop

Für «Drag & Drop» wurde die sogenannten «DropLists» von Angular Material CDK benutzt, da diese einerseits alle Anforderung an der benötigten Funktionalität erfüllen und andererseits keine weiteren Abhängigkeiten mit sich bringen. Alternative Libraries, welche nicht kostenpflichtig sind, waren meist erweitere oder angepasste Varianten der «DropLists».

Auf der gesamten Ansicht existieren 3 Bereiche für Drag & Drop.

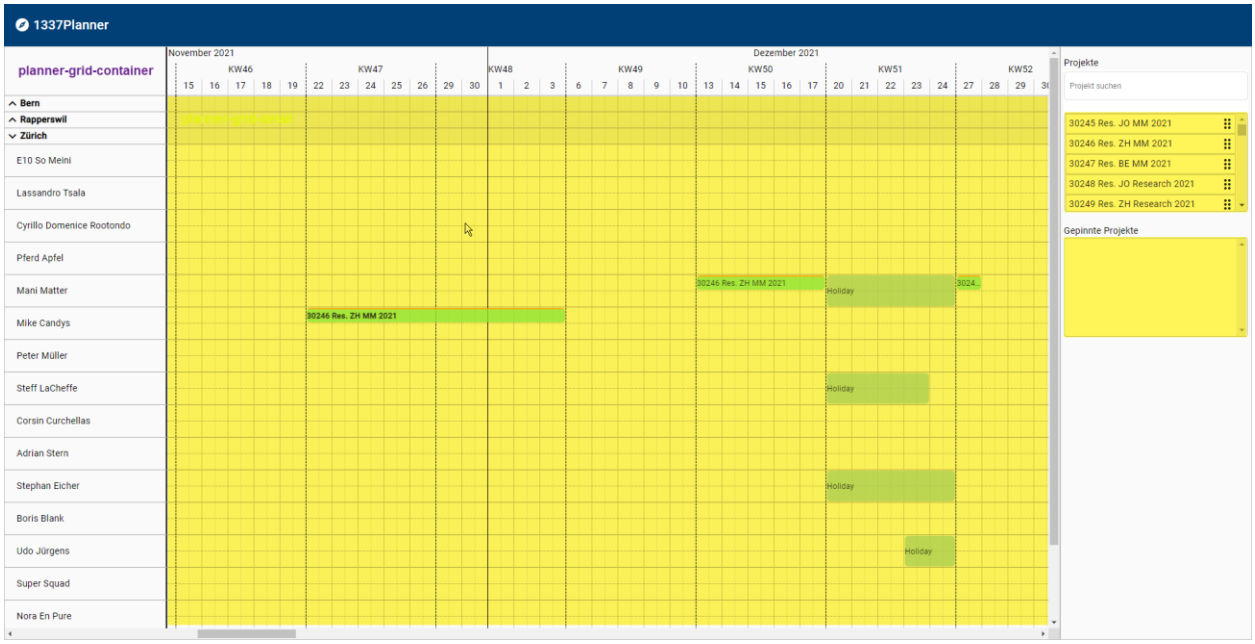


Abbildung 13: DropLists im UI

Die grosse DropList in «Abbildung 13: DropLists im UI» (grosse gelbe Fläche) berechnet, während dem Ziehen (nach 50ms Stillstand) und nach dem Loslassen anhand der Position der Cursors, welches Datum und welcher Mitarbeiter an der entsprechenden Stelle ist.
Während dem Ziehen wird ein Platzhalter angezeigt, damit sich der Benutzer vor dem Loslassen vergewissern kann, dass dies die Aktion ist, welcher er durchführen möchte.
Dieses Platzhalter-Feature entstand nach einem Austausch mit Prof. Dr. Stolze und ist daher nicht in der Anforderungsanalyse enthalten.

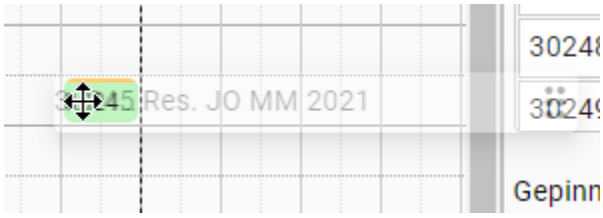


Abbildung 14: Drag & Drop Platzhalter

Nach dem Loslassen wird eine Planung mit den berechneten Daten erstellt oder angepasst, in den Store und ins Backend geschrieben.

7.1.1.4 Resizable

Mit Hilfe der Library «angular-resizable-element» kann man die Grösse von HTML-Elementen ändern. Durch diese Funktionalität kann man Assignments verlängern oder verkürzen.

In der Library wurde konfiguriert, dass die Vergrösserungsschritte genau die Breite der Grid-Columns betragen. Am Ende der Resize-Aktion wird anhand der verschobenen Pixelbreite berechnet, welchem Halbtag diese Änderung entspricht und dann das entsprechende Startdatum oder Enddatum gesetzt.

Durch einfaches Klicken am Rand, ziehen und loslassen, können die Planungen verlängert oder verkürzt werden.

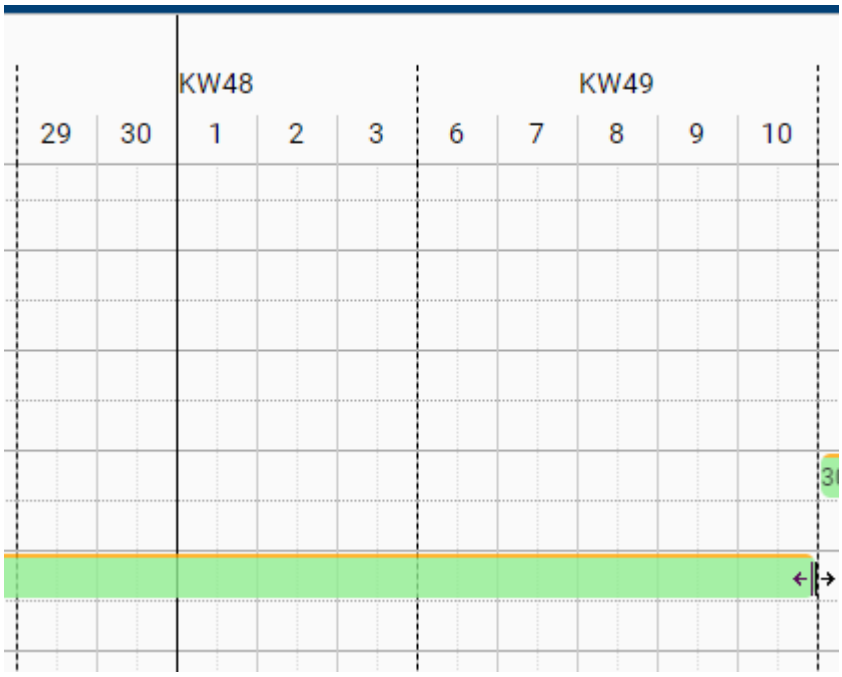


Abbildung 15: Planung mit anpassbarer Grösse

7.1.1.5 Authentication

Für die Authentifizierung und Autorisierung haben wir im Frontend die «Microsoft Authentication Library» verwendet. D.h. sobald die Applikation produktiv in Betrieb ist, wird eine Authentifizierung bei dem konfigurierten «Azure AD Tenant» verlangt. Wenn sich im Speicher der Applikation noch Informationen zur Authentisierung befinden, findet eine Weiterleitung zur entsprechenden Webseite von Microsoft statt.

Sollte man dort bereits eingeloggt sein, wird sofort eine erneute Weiterleitung zur 1337Planner Applikation stattfinden, mit dem entsprechenden Token und weiteren Authentication-Informationen. Dies wäre der typische Fall für SSO.
Falls man sich noch nicht auf dem Microsoft Tenant angemeldet war, findet die Weiterleitung erst nach einer erfolgreichen Anmeldung statt.

Sobald die Applikation die Authentication-Informationen bekommt, wird noch im Token überprüft, ob die konfigurierte Rolle «Planner» vorhanden ist. Erst danach stellt sich das Frontend dar.

Ein abstraktes Sequenzdiagramm dieses Authentifizierungsprozess sieht wie folgt aus:
Die Interaktion zwischen verschiedenen «Participants» (gelbe Kästchen) erfolgt über Redirects.

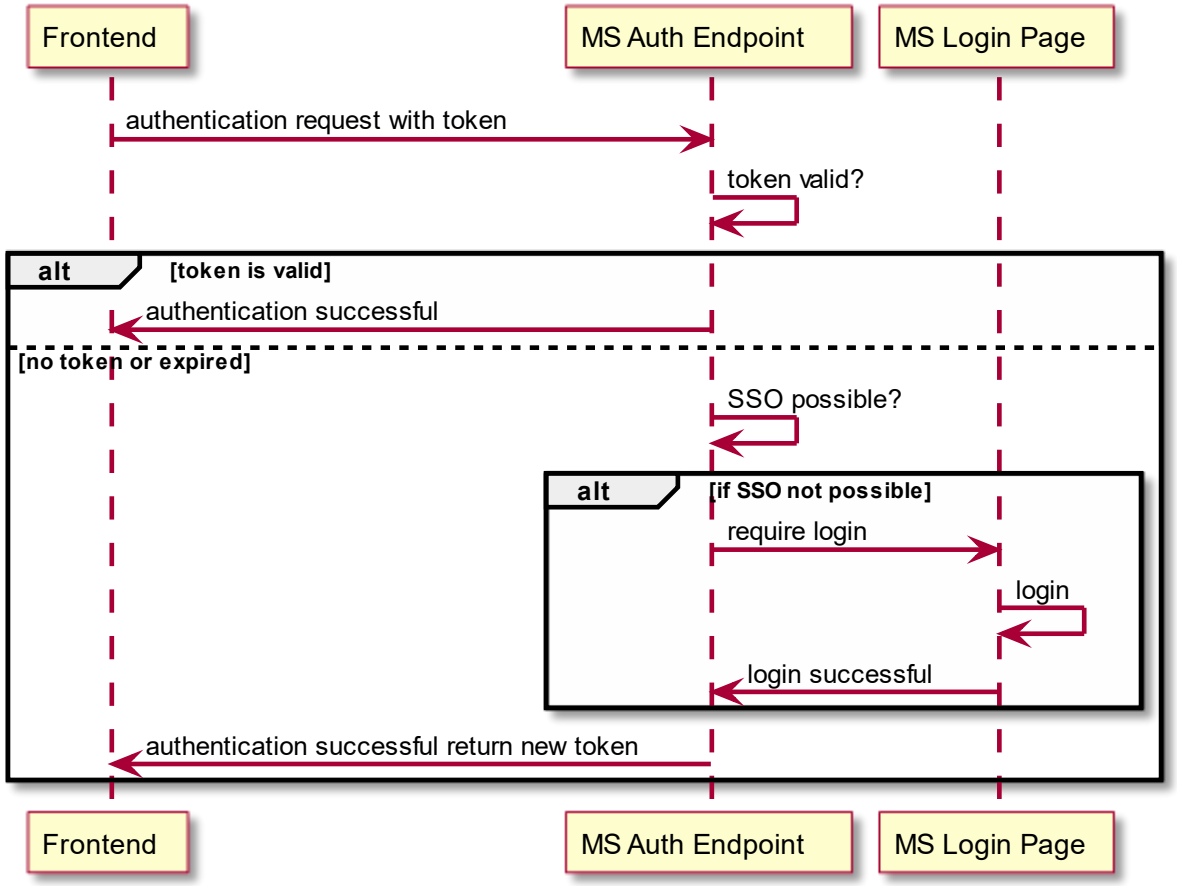


Abbildung 16: Sequenzdiagramm Frontend Authentifizierung

7.1.2 Backend

7.1.2.1 WebSocket

Wir verwenden WebSockets um jegliche Änderungen an Daten, dem Frontend dynamisch mitzuteilen. Dieser Ablauft funktioniert so:

Sobald ein Benutzer die Seite öffnet und sich authentifiziert hat, verbindet er sich via WebSockets zum Backends. Falls nun ein Benutzer etwas an den Daten ändert, z.B. verschieben oder vergrössern einer Planung, wird im Backend die Änderung via WebSockets allen abonnierten Clients mitgeteilt.
Die Daten werden im JSON-Format versendet.

Sobald ein User die Webseite schliesst, wird im Frontend die Websocket-Verbindung beendet.

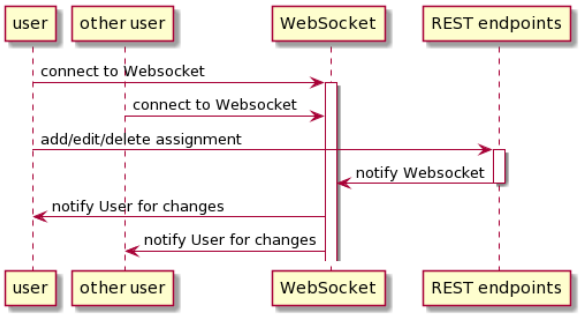


Abbildung 17: Sequenzdiagramm WebSocket Funktionalität

Es werden bei Änderungen an den Planungen und beim Import der Daten eine Nachricht an alle User gesendet.

Beim Verbinden auf den Websocket wird zusätzlich geprüft, von welcher Origin auf den Websocket zugegriffen wird, um CSRF Attacken zu verhindern. Alle zugelassenen Origins können in der Konfiguration definiert werden.

7.1.2.2 Authentication

Da im Frontend der Benutzer authentifiziert wird muss im Backend nur geprüft werden, ob das mitgelieferte Token gültig ist. Dazu wurde die Klasse «TokenAuthorization.cs» erstellt. Diese kann in den Controllern eingebaut werden, sodass vor den jeweiligen Methodenaufrufen zuerst das Token validiert wird.

```
@TokenAuthorized
public class AbsenceController extends Controller {
    private final AbsenceRepository absenceRepositor
    private final HttpExecutionContext ec;
    private final Logger log = LoggerFactory.getLogg

    @Inject
    public AbsenceController(AbsenceRepository absen

    @Transactional
    public CompletionStage<Result> getAbsences() {
```

Abbildung 18: Einbindung der Token Autorisation in einem Controller

Ablauf der Token-Validierung:

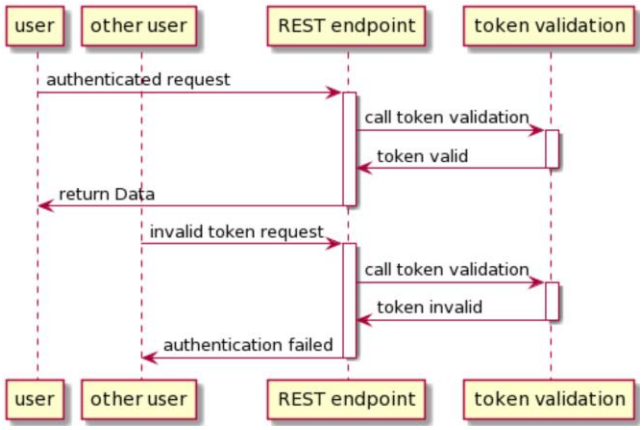


Abbildung 19: Sequenzdiagramm Backend Authentifizierung

Um bei der Entwicklung und Ausführung der Integrationstests nicht verhindert zu sein, wurde zusätzlich eine Konfiguration eingebaut, mit welcher die Validierung des Tokens übersprungen werden kann.

7.1.2.3 Daten Import

Da unser Programm mit den Daten des aktuell verwendenden Programms arbeiten soll, stellt uns Compass Security diese Daten in Form von CSV Dateien zur Verfügung. Wir erhalten 4 Dateien, in welchen jeweils folgende Daten enthalten sind:

Analysten Datei	Mitarbeiter der Compass Security
Analysten Projekte Datei	Projekte der Compass Security
Analysten Workingtime Datei	Absenzen der Mitarbeiter (Ferien, Kompensation, Militär, etc.)
Analysten Public Holiday Datei	Die jeweiligen Feiertage der Standorte. (Weinachten, Neujahr, etc.)

Abbildung 20: Importdaten

Diese Dateien werden vom Backend beim Start und dann in einem definierten Zeitintervall neu importiert. Alle vorhandenen «Workingtime»- und «Public Holiday»-Daten in der Datenbank werden jeweils gelöscht beim Importieren, um veraltete Daten zu entfernen. Bei den Projekt- und Mitarbeiterdaten werden bestehende Datensätze geupdatet oder falls sie noch nicht vorhanden sind, neu erstellt.

Falls während des Imports ein Fehler auftritt, wird das Frontend mittels WebSocket informiert, dass ein Fehler aufgetreten ist, damit das Frontend diese Benutzer anzeigen kann.

Der Ablageort der Dateien und der jeweilige Zeitintervall in welchem die Daten importiert werden, muss in den Konfigurationen definiert werden.

7.1.2.4 Zeitberechnung

Damit ein Planer immer sehen kann, wie viel Zeit auf einem Projekt geplant wurde, wird im Backend immer die aktuell geplante Zeit eines Projekts berechnet. Diese Zeit berechnet sich anhand der jeweiligen Planungen. Nachfolgend wird dazu noch ein Beispiel gemacht. Die Zeit wird berechnet, sobald das Backend gestartet wird und bei jeder Änderung einer Planung.

Erklärung der Kalkulation:

Jeder Mitarbeiter hat im UI zwei Zeilen und ein Tag ist aufgeteilt in Morgen und Nachmittag. Das heisst ein Mitarbeiter arbeitet entweder pro Halbtag 0.5 Tage oder 0.25 Tage an einem Projekt. 0.5 Tage können dann geleistet werden, wenn nur in einer Lane ein Projekt geplant ist, 0.25 Tage, falls beide Lanes eines Halbtages belegt sind.

Der Algorithmus kalkuliert die Stunden jedes einzelnen Mitarbeiters separat und addiert dann die erhaltenen Werte, um auf das Total zu kommen.
Die berechneten Stunden werden in einem Dictionary gespeichert, in welchem zu jedem Projekt die Anzahl geplanter Stunden gespeichert werden. Dieses Dictionary kann dann via WebSockets dem Client gesendet werden kann.

Auf der nächsten Seite befindet sich ein technisches Aktivitätsdiagramm, welches den Ablauf des Algorithmus erklärt. Das Diagramm entspricht nicht eins zu eins der umgesetzten Logik, damit das Diagramm einfacher zu verstehen ist.
Nach dem Diagramm ist der Algorithmus noch anhand eines Praxisbeispiel erklärt, um den Ablauf der Berechnung aufzuzeigen.

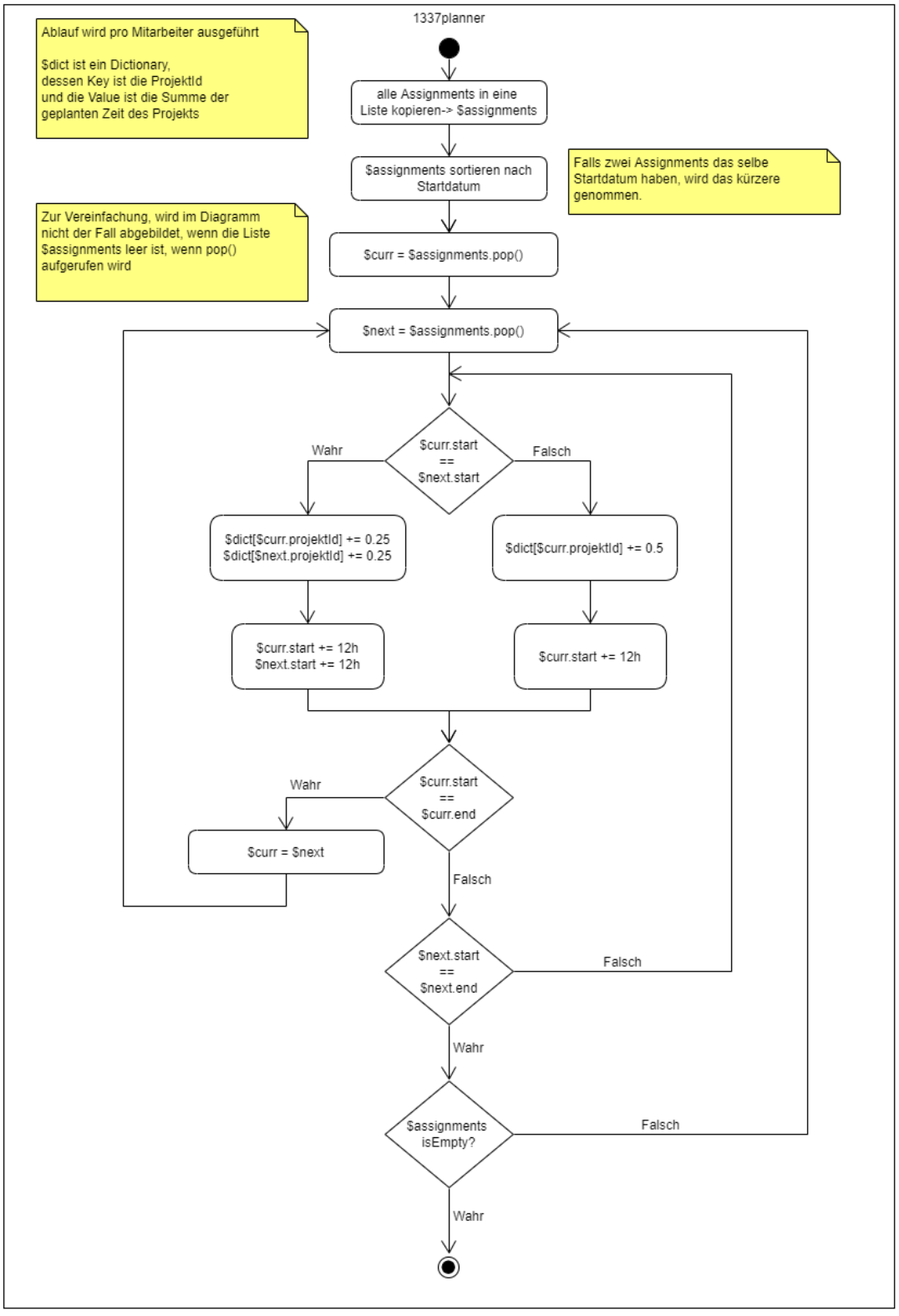


Abbildung 21: Zeitberechnungslogik Diagramm

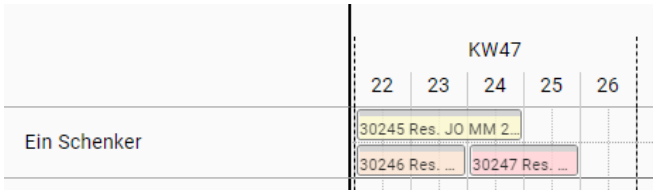


Abbildung 22: Zeitkalkulationsbeispiel

Wir haben im Beispiel 3 Assignments, diese haben alle unterschiedlichen Projekte. Im Nachfolgenden Beispiel wird, wenn von den Assignments gesprochen wird, jeweils die letzte Nummer der Projektnummer (z.B.: 30245 -> 5) verwendet, um diese voneinander zu unterscheiden.

Da 5 und 6 zuerst starten und 6 kürzer ist, wird 6 als erstes selektiert. Nun wird das Start Datum von 6 mit dem von 5 verglichen da 5 das frühste Start Datum der übrigen Assignments hat. Da die Start Daten gleich sind, werden auf beiden Projekten je ¼ Tag dazu gerechnet. Nun wird dies für die nächsten halben Tage auch gemacht bis zum Ende des 23. dort ist Assignment 6 zu Ende.
Nun wird wieder verglichen, welches Projekt als nächstes startet und falls sie wieder am gleichen Datum starten, wird das kürzere genommen. In unserem Fall wäre dies das Projekt 5.
Nun wird 5 also mit 7 verglichen und da sie gleichzeitig eingeplant sind bis 5 zu Ende ist, wird je einen halben Tag pro Projekt dazu gerechnet.
Da nun nur noch 7 übrig ist wird es selektiert und für die offene Planungszeit 1 Tag zur Planungszeit von 7 dazu gerechnet.

Am Ende haben die Projekte folgende Zeiten:
5 -> 1.5 Tage
6 -> 1 Tag
7 -> 1.5 Tage

7.1.2.5 Konfigurationen

Alle Konfigurationen und weitere Informationen, welche für den weiteren Betrieb der Software zentral sind, sind im Readme des Repositories beschrieben.

Hier werden zusätzlich noch einige wichtige Konfigurationen vom Backend erwähnt.

data-import.file-path	Beinhaltet den Pfad an welchem die Import Daten abgelegt werden.
data-import.refresh-interval	Definiert das Intervall in welchem die Import Daten neu importiert werden kann in Minuten.
websocket-valid-origins	Alle Origins von welchen man auf den WebSocket verbinden darf.
websocket.only-accept-same-origin	Legt fest, ob die Origin Validierung gemacht wird bei Verbindungen auf die Websockets.
hostname	Legt fest unter welchem Hostnamen das Backend läuft. Wird für die Verbindung auf den eigenen WebSocket verwendet.

play.server.*.timeout	Wird verwendet, um festzulegen wie lange WebSockets offenbleiben. Bei infinite oder null werden WebSockets nicht automatisch geschlossen.
authorization.use-authorization	Legt fest, ob die Token Validierung aktiviert ist.

7.2 Automatische Testverfahren

7.2.1 Frontend

Im Frontend wird bei jedem Pull-Request in den develop-Branch eine CI-Pipeline gestartet, welche überprüft ob alle Tests erfolgreich sind, bevor man «mergen» kann.

7.2.1.1 Unit-Tests

Die Pipeline überprüft, ob alle Unit-Tests erfolgreich durchlaufen und alle ESLint-Rules eingehalten sind.

Die Unit-Testabdeckung sieht am 20.12.2021 wie folgt aus:

All files

95.74% Statements 608/635 **79.09%** Branches 174/220 **94.36%** Functions 134/142 **96.05%** Lines 536/558

Press *n* or *j* to go to the next uncovered block, *b*, *p* or *k* for the previous block.

File	Statements		Branches		Functions		Lines	
app/core/components/spinner-overlay	100%	5/5	100%	0/0	100%	0/0	100%	3/3
app/core/services	100%	21/21	100%	2/2	100%	5/5	100%	17/17
app/planner/components/assignment	98.21%	55/56	60%	12/20	91.66%	11/12	100%	51/51
app/planner/components/employee	100%	41/41	90.9%	10/11	100%	9/9	100%	36/36
app/planner/components/grid-container	100%	80/80	88.23%	15/17	100%	17/17	100%	75/75
app/planner/components/grid-detail	95.18%	79/83	69.44%	25/36	93.75%	15/16	95%	76/80
app/planner/components/sidebar	91.35%	74/81	70%	21/30	92.85%	26/28	91.78%	67/73
app/planner/pipes/date-to-column	100%	23/23	100%	13/13	100%	5/5	100%	18/18
app/planner/pipes/filters	86.36%	38/44	100%	7/7	84.61%	11/13	86.2%	25/29
app/planner/services	96.77%	60/62	100%	15/15	85.71%	12/14	98.18%	54/55
app/planner/utils	94.92%	131/138	78.26%	54/69	100%	23/23	94.16%	113/120
environments	100%	1/1	100%	0/0	100%	0/0	100%	1/1

Abbildung 23: Abdeckung Frontend Unit-Tests

7.2.1.2 Integrationstests

Für die Integrationstests wurde Cypress verwendet.
Vor den Test wird durch die Methode «cy.intercept» garantiert, dass bei jedem Request dessen Reponse mit einen spezifischen JSON ersetzt wird.

Es wurden 3 Integrationstests erstellt, zwei davon repräsentieren einfache sunny-case-Tests unserer wichtigsten Aktionen.

1. Der erste Test kontrolliert, ob der Header den richtigen Text beinhaltet. Dieser Test bietet allein keinen Mehrwert, soll aber sicherstellen, dass die Seite korrekt dargestellt wird, für darauffolgenden Tests

2. Der zweite Test führt eine Drag & Drop Aktion durch bzw. dieser erstellt ein Assignment. Dafür wird ein Projekt gezogen und auf dem Grid losgelassen. Danach wird überprüft, ob das Grid eine Assignment darstellt.
3. Beim dritten Test wird getestet, ob ein Assignment verlängert werden kann. Hier wird am rechten Rand des HTML-Elements gezogen und danach weiter rechts losgelassen. Schliesslich wird überprüft, ob die Länge des HTML-Elements in einem vordefinierten Bereich ist.

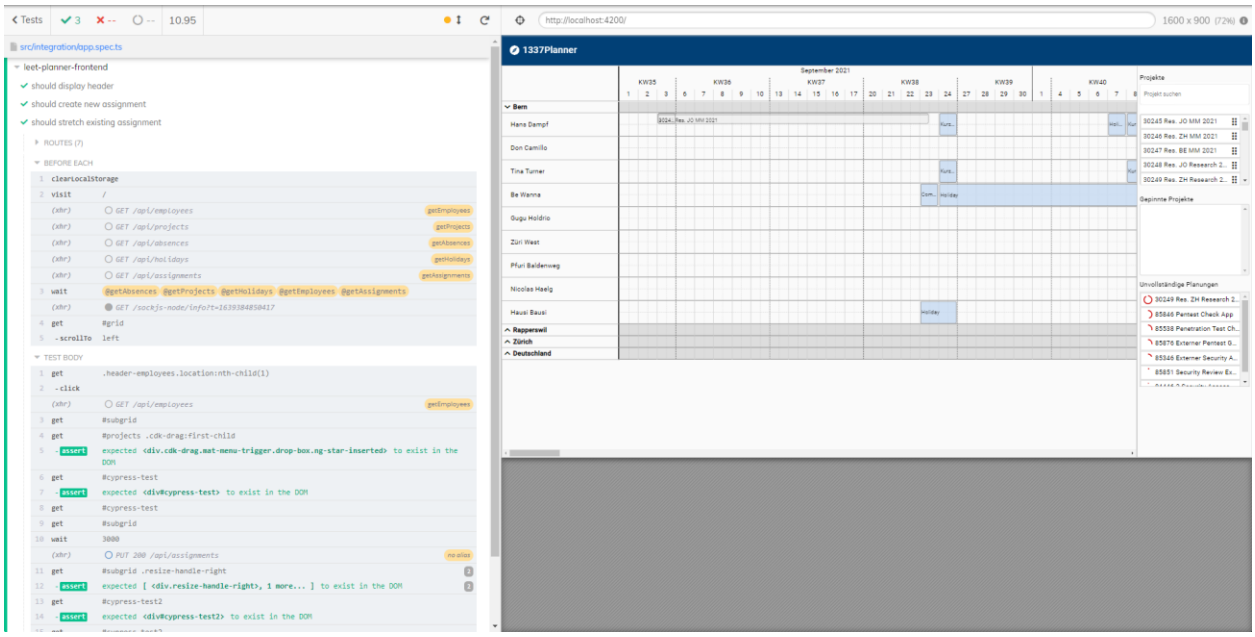


Abbildung 24: Screenshot Cypress Tests

7.2.2 Backend

Im Backend wird bei jedem Pull-Request in den develop-Branch eine CI-Pipeline gestartet, welche überprüft ob alle Tests erfolgreich sind, bevor man «mergen» kann.

7.2.2.1 Unit-Tests

Die Pipeline überprüft, ob alle Unit-Tests erfolgreich durchlaufen.

Die Unit-Testabdeckung sieht am 20.12.2021 wie folgt aus:

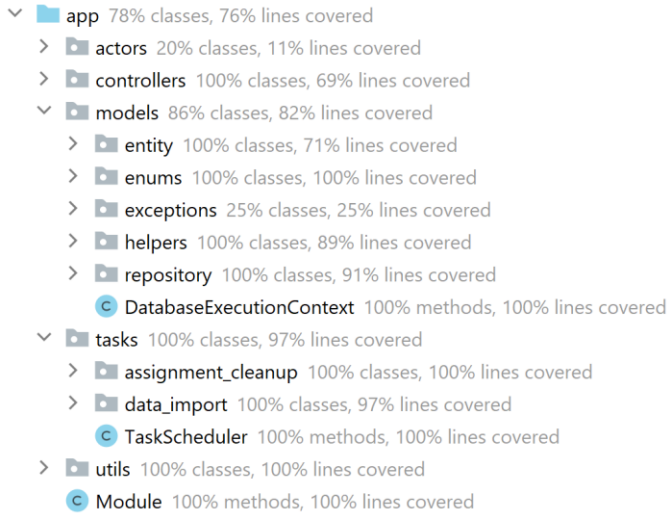


Abbildung 25: Testabdeckung Backend

7.2.2.2 Integrationstests

Im Backend wurden alle REST-Schnittstellen mit Integrationstests getestet. Wie bei den Unit-Tests verwenden wir hier JUnit und Mockito.

7.2.2.3 Testen der WebSockets

Da wir während der Umsetzung der WebSockets sehr viel Zeit verloren haben (siehe Kapitel «10.6.1 Eingetretene Risiken»), entschieden wir uns dazu die WebSockets nicht zu testen. Diese sollten aber in Zukunft mit Integrationstests getestet werden. Eine nützliche Hilfe hierzu wäre folgendes Beispiel auf welchem unsere WebSockets basieren:

https://github.com/playframework/play-samples/tree/2.8.x/play-java-websocket-example

8. Usability Tests

8.1 Testdurchführung vom 06.12.2021

Wir haben am 06.12.2021 mit zwei Mitarbeitern der Firma Compass Security Usability Tests durchgeführt. Die Tests wurden bei der Compass Security vor Ort durchgeführt. Beide Mitarbeiter arbeiten aktuell mit der bisherigen Einsatzplanungssoftware. Sie testeten nacheinander, damit die Meinung des einen die des anderen nicht beeinflusst und dass möglichst viel und allenfalls identisches Feedback entsteht.

Unser Projektpartner Cyrill Brunschwiler hat nicht mitgetestet, da er durch sein Vorwissen die Software bereits kannte und er auch der Hauptideengeber des Projekts ist.

Es wurde zwei möglichst alltagsnahe Szenarien als Aufgaben beschrieben, welche die Tester während den Tests durchführten. Die Szenarien wurden anhand der Gespräche, welche bis dahin mit Cyrill Brunschwiler geführt wurden, gestaltet.
Die Szenarien, welche wir den Testern gegeben haben, befinden sich im Anhang.

Zusätzlich haben wir den Testern ein kurzes Bild gegeben, welches die speziellen Eigenheiten der Software darstellt (Drag & Drop und Infoboxen).
Dieses befindet sich ebenfalls im Anhang.

Während den Tests wurden alle Anmerkungen und auffälliges Verhalten der Tester notiert.
Zur gleichen Zeit wurden die Tester falls nötig oder bei Fragen unterstützt.

(Die grün eingefärbten Massnahmen wurden bereits umgesetzt.)

Typ	Problem	Massnahme
Benutzer Inputs	Während des Öffnens eines Standorts weiss man nicht, ob die Webseite etwas macht oder nicht.	Einbauen eines Ladezeichens beim Öffnen oder Schliessen eines Standorts.
	Die dicken Striche im Raster, welche das Monatsende zeigen, verwirren mehr als das sie helfen.	Entfernen der Monatslinien aus dem untersten Grid.
	Sich mit Scrollen zurecht finden ist schwierig aufgrund des grossen Zeitbereichs	Einbauen eines Schnellnavigiermechanismus, welcher zum gewünschten Zeitpunkt scrollt.
	Projektleiter können nur für die Dauer eines Assignments gesetzt werden	Pro Projekt einen Projektleiter setzten und nicht pro Assignment
	Bei vielen Mitarbeitern pro Standort ist es schwer mehrere gewünschte Mitarbeiter einfach zu sehen	Mitarbeiter als Favoriten markierbar machen und im Standort zuoberst anzeigen.
	Bei vielen Assignments ist es schwer zu sehen, welcher Mitarbeiter noch freie Kapazität hat.	Eine Ansicht erstellen, in welcher die freien Ressourcen eines Mitarbeiters ersichtlich sind über eine gewisse Zeitperiode. Evtl. sogar farblich markieren.

	Man sieht nicht, welche Mitarbeiter alles auf einem Projekt eingeplant sind	Eine Ansicht erstellen in welcher ersichtlich ist, welche Mitarbeiter an einem Projekt beteiligt sind
	Wenn man alle Standorte offen hat, ist es schwer einen Mitarbeiter zu finden.	Suchfunktion für Mitarbeiter erstellen
	Auf kleineren Bildschirmen ist die Schrift zu klein. Nach dem Zoom der Zeitbereich zu klein.	Evtl. via Hover den Text vergrössern.
	Blöcke sind initial immer nur einen Tag lang, es wäre aber schön, wenn die Standardgrösse pro Benutzer einstellbar ist.	Möglichkeit zur Einstellung einer Standardblockgrösse anbieten.
Fehler	Wenn ein Gepinntes Projekt auf den Planungsraster gezogen wird, ändert sich die anzuzeigende Farbe	Die korrekte Farbe anzeigen während dem Ziehen eines Projekts
	Bei gepinnten Projekten ist die geplante Zeit immer 0.	Fehler beheben, welcher die Zeit nicht anzeigt.
	Darstellungsfehler, nach dem Verlängern von Assignments, wenn das Start- oder End-datum vor und nach den Änderungen in verschiedenen Zeitzonen ist.	Fehler beheben.
	Projekte können nicht auf ½ Tage geplant werden.	Fehler beheben.
	Projekte können bei Fehler länger als die maximale Planzeit eingeplant werden.	Spätestes End Datum ist das späteste angezeigte Datum.
	Beim Erstellen / Verschieben eines Assignments kann über Absenzen / Ferien geplant werden.	Das Programm soll automatisch um Absenzen / Ferien planen.
Unsere Inputs	Die vielen unterschiedlichen Linien im Raster verwirren den Benutzer	Zukünftig eine Einstellungen Seite einbauen, welche einem ermöglicht das Raster zu individualisieren.
	Je nach Standort werden Mitarbeiter teilweise nur auf ein Projekt eingeplant.	Auf einer Einstellungsseite ermöglichen das Mitarbeiter eine oder zwei Zeilen haben. Pro Standort oder Abteilung.
	Gepinnte Projekte verschwinden nach einem Neu laden der Seiten.	Gepinnte Projekte pro User speichern.
	Neue Benutzer hatten trotz Cheat Sheet und Unterstützung Probleme mit der Bedienung der Webseite	Interne Schulung für Mitarbeiter.

9. Resultate & Weiterentwicklung

9.1 Resultate

9.1.1 Resultate funktionaler Anforderungen

Folgende Tabelle zeigt, welche User Stories des MVP erfolgreich umgesetzt werden konnten und welche nicht:

User Stories	User-Story 1	User-Story 2	User-Story 3	User-Story 4	User-Story 5
Erfüllte Akzeptanzkriterien	1 bis 2	3 bis 5	1 bis 7, (9)	1	-
Nicht erfüllte Akzeptanzkriterien	-	1-2 (teilweise erfüllt)	8, (9)	-	1 - 5

Das Hauptproblem für die Nicht-Erfüllung der verbleibenden Anforderungen war die fehlende Zeit. Besonders nennenswert Faktor, welcher uns viel Zeit gekostet hat, ist die Umsetzung der WebSockets im Play Framework. Dieses Risiko wurde von Anfang an berücksichtigt und ist im Kapitel «10.6.1.1 Neue Technologien» ausführlich dokumentiert.

User-Story 2 / AK 1-2

Die grundsätzliche Anforderung wurde erfüllt, jedoch entstand während dem Usability Tests Feedback, welches darauf hindeutet, dass man diese Features noch weiter ausbauen muss, damit diese einen praktischen Mehrwert bringen. (Wir empfehlen zuerst eine neue Anforderungsanalyse.)

User-Story 3 / AK 9

Bei Problemen während Synchronisation wird über den WebSocket eine entsprechende Nachricht geschickt. Diese wird im Web aktuell noch nicht angezeigt. (ca. 8h verbleibender Aufwand mit Tests)

9.1.2 Resultate nicht-funktionaler Anforderungen

Usability (NFR1 und NFR2)

Die definierten Usability Tests wurden durchgeführt und sind im Kapitel «8 Usability Tests» dokumentiert.

Da nicht alles Massnahmen aus den Usability Tests umgesetzt wurden, ist diese Anforderung nicht komplett erfüllt.

Security (NFR3)

Massnahmen gegen die gängigen Sicherheitslücken wurden getroffen und Best-Pratices bzgl. Security von den beiden Frameworks (Play Framework, Angular) befolgt. Compass Security führt nach der Übergabe selbst Penetrationstest an der Applikation durch, um die korrekte Umsetzung dieses NFRs zu verifizieren.

Portability (NFR4)

Das Grid-Widget ist wiederverwendbar, jedoch noch nicht von aussen parametrisierbar, da dies die Performance der Lösung beinträchtigen würde. Dadurch, dass alle Daten im Store sind und daraus gelesen werden, läuft das Frontend performant. Durch ein kleiner Refactoring (ca. 4h) kann das Widget komplett parametrierbar gemacht werden.

Das Frontend weist eine weitaus komplexere Logik als das Backend auf. Michael Gfeller hat im Rahmen dieser Arbeit zwei Code-Reviews in beiden Repositories durchgeführt. Er konnte das Frontend nur mit Hilfe von dieser Dokumentation und dem jeweiligen Readme verstehen und Verbesserungen und Optimierungen einbringen. Aus diesem Grund wird dieser NFR als erfüllt betrachtet.

Maintainability (NFR5-7)

Die definierten Anforderungen dieses NFRs sind erfüllt und können im Kapitel «7.2 Automatische Testverfahren» eingesehen werden.

Performance (NFR8 und NFR9)

NFR8 entfällt durch die aktuelle Umsetzung, da immer der gesamte relevante Zeitbereich angezeigt wird. Falls die Performanceoptimierung «Scroll-Bereich einschränken» im Kapitel «9.2.4 Performance» umgesetzt wird, müsste dieser NFR berücksichtigt werden.

NFR9 wurde im Rahmen von Meetings mit Cyrill Brunschwiler und den Usability Tests den zukünftigen Benutzern demonstriert.
Da die Synchronisation bisher nicht als störend bezeichnet worden ist, wird dieser NFR ebenfalls als erfüllt betrachtet.

9.2 Weiterentwicklung

9.2.1 Nicht komplett eingebundene Features

Die Authentication wurde im Backend noch nicht komplett eingebaut da unser Wissen nicht ausreichte in der verbliebenen Zeit die Überprüfung des Tokens einzubauen. Es existiert daher im GitHub Repository des Backends ein Pull Request welcher mit dem Tag «work in progress» markiert wurde. Er beinhaltet den aktuellen Stand des «develop» Branches.

In diesem Branch wurde bereits eine Annotation eingebaut, welche in Klassen mittels «@TokenAuthorized» eingebaut werden kann. Nun kann mit der «@Transactional» Annotation die Methode «call» der Klasse TokenAuthorization aufgerufen werden.

Beispiel der Einbindung in eine Klasse:

```
@TokenAuthorized
public class HolidayController extends Controller {

    private final HolidayRepository holidayRepository;
    private final HttpExecutionContext ec;
    private final Logger log = LoggerFactory.getLogger(this.getClass());

    @Inject
    public HolidayController(HolidayRepository holidayRepository, HttpExecutionContext ec) {...}

    @Transactional
    public CompletionStage<Result> getHolidays() {
        return holidayRepository
```

Abbildung 26: Einbau der Annotation Token Authorized

Die Methode «call» prüft, ob eine Token Validierung nötig ist. Diese kann mittels der Konfiguration aktiviert werden (siehe Kapitel 7.1.2.5 Konfigurationen). Falls eine Token Validierung nötig ist, wird, wenn ein Token vorhanden ist, die Methode «isValidToken» der Klasse TokenHelper aufgerufen. Diese Methode wurde bisher noch nicht implementiert. Die Methode muss noch das Token validieren mit Hilfe der Azure Authentication der Compass Security.

9.2.2 Features

Diverse noch nicht umgesetzte Features sind bereits in anderen Kapiteln aufgeführt und werden hier nur nochmals referenziert.

- Alle Akzeptanzkriterien, welche nicht zum MVP gehören oder noch nicht umgesetzt worden sind und weitere aufgeführte optionale Features im Kapitel «4 Anforderungsspezifikation».
- Noch nicht umgesetzten Massnahmen im Kapitel «8 Usability Tests»
- Nicht komplett abgeschlossene Features des MVPs im Kapitel «9.1.1 Resultate funktionaler Anforderungen»
- Die WebSocket Funktionen des Backends sollten noch getestet werden mit Integrationstests. Weitere Infos dazu befinden sich im Kapitel «7.2.2.3 Testen der WebSockets»

9.2.3 Bekannte oder potenzielle Fehler

- Noch nicht behobene Fehler im Kapitel «8 Usability Tests»
- Bei vielen geplanten Assignments, funktioniert die Berechnung der Zeiten nur beim Start. Sobald was verändert wird, sind die Zeiten leer.
    - Wir empfehlen den Fehler zuerst genauer zu analysieren und dann den Fehler zu beheben.

9.2.4 Performance

Innerhalb der Arbeit wurden Performance-Tests vorgenommen. Vor diesem Test wurde jedem Mitarbeiter wochenweise ein zufälliges Projekt zugewiesen. D.h. es wurden knapp 3000 «Assignments» verteilt auf 38 Mitarbeitern auf dem Frontend angezeigt. Dabei war die initiale Ladezeit (bis die Applikation bedienbar ist) nicht sehr zufriedenstellend, da dies ca. 10 Sekunden dauert.
Während der Bedienung leidet die Performance ebenso von spürbaren Verzögerungen.
Beim Auf- und Zuklappen von Standorten dauert das «Rerendering» auch deutlich länger (ca. 5 Sekunden), was aber weniger schwerwiegend ist, da ein Benutzer nicht ständig Standorte auf und zuklappt.

Für zukünftige Performanceoptimierungen empfehlen wir folgende Möglichkeiten in Betracht zu ziehen:

- **Scroll-Bereich einschränken**
  Statt die kompletten 18 Monate auf dem Zeitstrahl zu rendern, wäre es möglich jeweils nur eine gewisse Anzahl z.B. 6 Monate anzuzeigen, z.B. Januar bis Juni. Sollte ein Benutzer dann etwas bis August planen wollen, könnte man über einen Button oder via EventListeners eine Logik triggern, welche die hinteren 3 Monate entlädt und die nächste 3 Monate lädt. D.h. das Frontend würde jetzt April bis September anzeigen.

  Um dies umzusetzen, müssten ebenfalls auf dem Backend Query-Parameter eingeführt werden, um die Assignments nach einem Zeitbereich zu filtern.
  Im Frontend müsste man einige Codestellen erweitern, aber es benötigt kein grosses Refactoring.

  Wir schätzen einen Arbeitsaufwand von ca. 4 Tagen

- **Selektion von Mitarbeitern**
  Aktuell werden immer alle Mitarbeiter eines Standorts angezeigt. Bei Standorten mit vielen Mitarbeitern, kann es dazu kommen, dass die Daten von mehr als einem Dutzend Mitarbeiten geladen und gerendert werden, obwohl man sich vielleicht gar nicht für alle diese interessiert.
  Daher wäre es eine potenzielle Optimierung, welche ebenfalls in den Usability-Test aufgekommen ist, Filter auf die Mitarbeiter anzuwenden.

  Diese Anforderung und Optimierung müsste sicherlich fachlich hinterfragt und genauer spezifiziert werden.

  Wir empfehlen weitere Anforderungsanalyse, bevor ein Arbeitsaufwand geschätzt werden kann.

- **Virtual Scrolling**
  Die grössten Performanceprobleme entstehen durch das Rendern von tausenden von Komponenten. Angular Material und andere Libraries bieten Funktionalität für Virtual Scrolling an.
  Dadurch wird der Applikation ermöglicht, nur die sichtbaren Bereiche zu rendern.
  Um dies zu erreichen ist allerdings ein grösserer Refactoring-Aufwand im Frontend nötig.

  Um diese Funktionalität zu nutzen, müssen Markup und Daten meist in eine vordefinierte Struktur gebracht werden, damit diese Libraries verwendet werden können.

  Wir empfehlen eine technische Machbarkeitsanalyse.
  Falls es machbar wäre, schätzen wir einen Arbeitsaufwand von ca. 10 Tagen

10. Projektmanagement

10.1 Prototypen, Releases, Meilensteine

Meilensteine	Anforderungsanalyse	Entwurf	Implementation & Test	Abgabe
Termin	04.10.2021	25.10.2021	13.12.2021	24.12.2021

10.2 Team, Rollen und Verantwortlichkeiten

Teammitglied	Rolle	Verantwortungen
Abdullah Almaz	Software-Entwickler	- Entwicklung & Testing Frontend - Unterstützung Backend - Projektmanagement - Dokumentation / Bericht
Ursin Zimmermann	Software-Entwickler	- Entwicklung & Testing Backend - Unterstützung Frontend - Projektmanagement - Dokumentation / Bericht

10.3 Entwicklungswerkzeuge & eingesetzte Software

Tools & Software	Verwendung
Balsamiq	Mockups, Wireframes
Cypress & Jest	Frontend Integration-Testing
ESLint	Frontend Code Quality
GitHub (Git)	Code-Versionierung, Code-Reviews, CI, CD, Release Management
Jest	Frontend Mocking & Unit-Testing
JetBrains IntelliJ Ultimate	Entwicklung Backend (Java Play Framework)
JetBrains WebStorm	Entwicklung Frontend (Angular)
JUnit & Mockito	Backend Mocking, Unit-Testing & Integration-Testing
MS Office Produkte	Dokumentation

MySQL Workbench	Datenbankdesign
Plantuml & Umlet & draw.io	Diagramme erstellen für Dokumentation
SonarLint	Backend Code Quality
YouTrack	Projektplanung, Zeiterfassung, Issue-Management, Sprint-Planung

10.4 Lizenzen

Library	Lizenz
Play	Apache 2 Lizenz
Mockito	MIT Lizenz
H2	Mozilla Public License V2.0
Hibernate	GNU Lesser General Public License V2.1
MySQL Connector Java	GNU General Public License V2.0

10.5 Aufwandschätzung, Zeitplan, Projektplan

Der Projektplan, die Zeitplanung und ein Export all unserer Tickets mit den dazugehörigen Aufwandschätzungen befinden sich im Anhang.

10.6 Risiken

Unsere Risikoanalyse soll uns eine Übersicht über mögliche Risiken geben. Wir ergänzen während des Projekts fortlaufend unsere Risiken.

Wir versuchen die meisten unserer Risiken mit Tests abzudecken und auf diese Art zu vermeiden. Falls jedoch ein Risiko trotzdem eintritt, haben wir eine Reservezeit eingeplant von 32 Stunden. Diese Zeit ergab sich aus der anfänglichen Zeit aller Risiken aufgerundet auf 8 Stunden.

10.6.1 Eingetretene Risiken

Folgende Risiken sind eingetreten während unserer Studienarbeit:

10.6.1.1 Neue Technologien

Im Backend verwenden wir einen WebSocket, um alle verbundenen Clients zu benachrichtigen, falls sich etwas an den Daten geändert hat. Die Entwicklung dieses Websocket dauerte länger als geplant, aufgrund der folgenden Gründe:

- Die Dokumentation zur Entwicklung eines Websocket im Play Framework ist dürftig dokumentiert.

- WebSockets verwenden im Play Framework das Actor Model, welches wir nur schlecht kannten.
- Wir hatten niemanden den wir um Unterstützung bitten konnten, da sich niemand aus unserem Umfeld mit dem Play Framework auskennt.

10.6.2 Risikotabelle und Risikomatrix

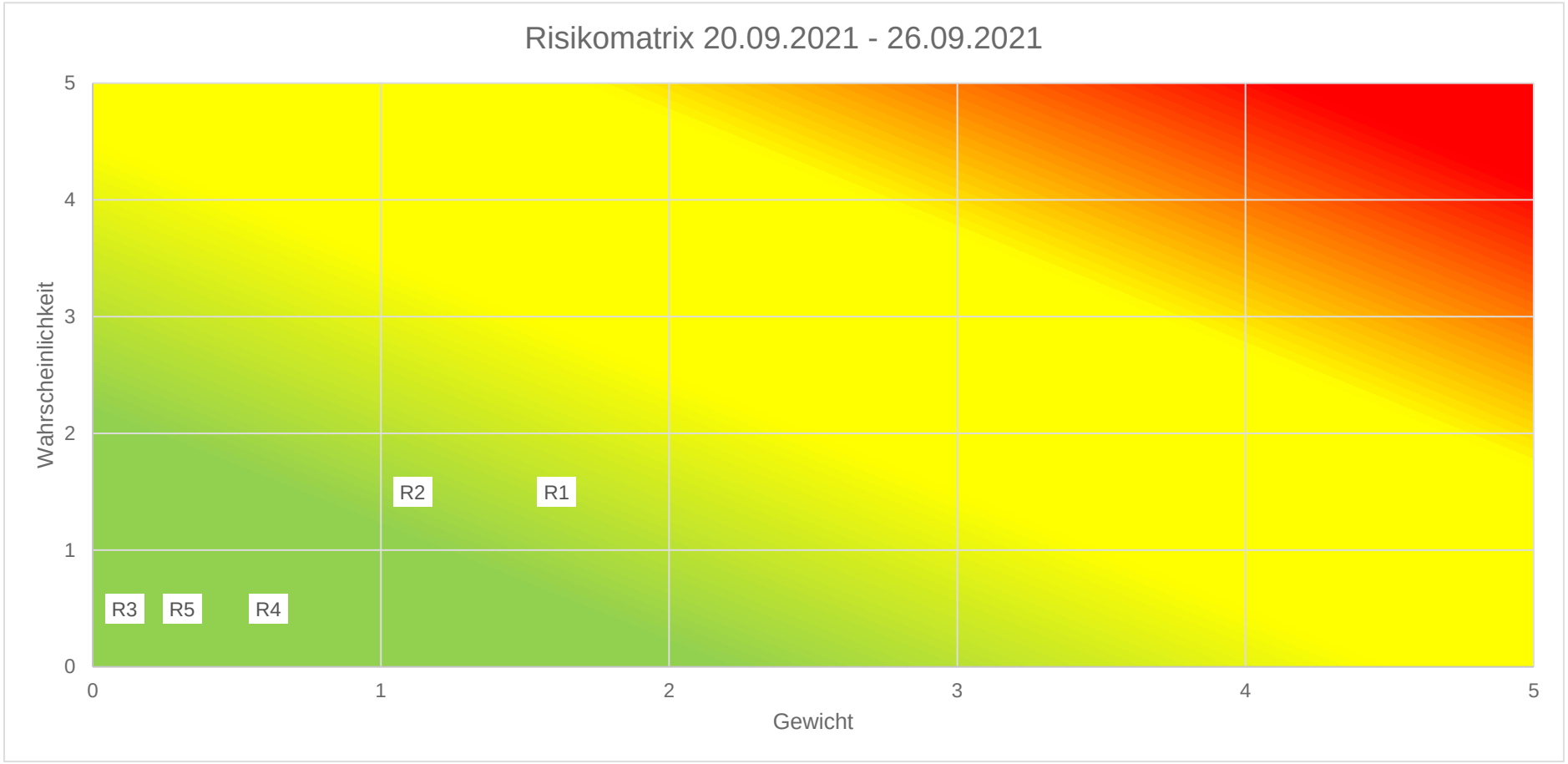
Unsere Risikotabelle und Risikomatrix basieren auf der Vorlage von thehosblog.com [7].

Alle 2 Wochen werden die Risiken nochmals aktualisiert.

Die Änderungen von Gewicht oder Wahrscheinlichkeit sind in fetter Schrift hervorgehoben.
Zusätzlich werden die positiven Änderungen mit **grüner** und die negativen mit **roter** Schrift gekennzeichnet.

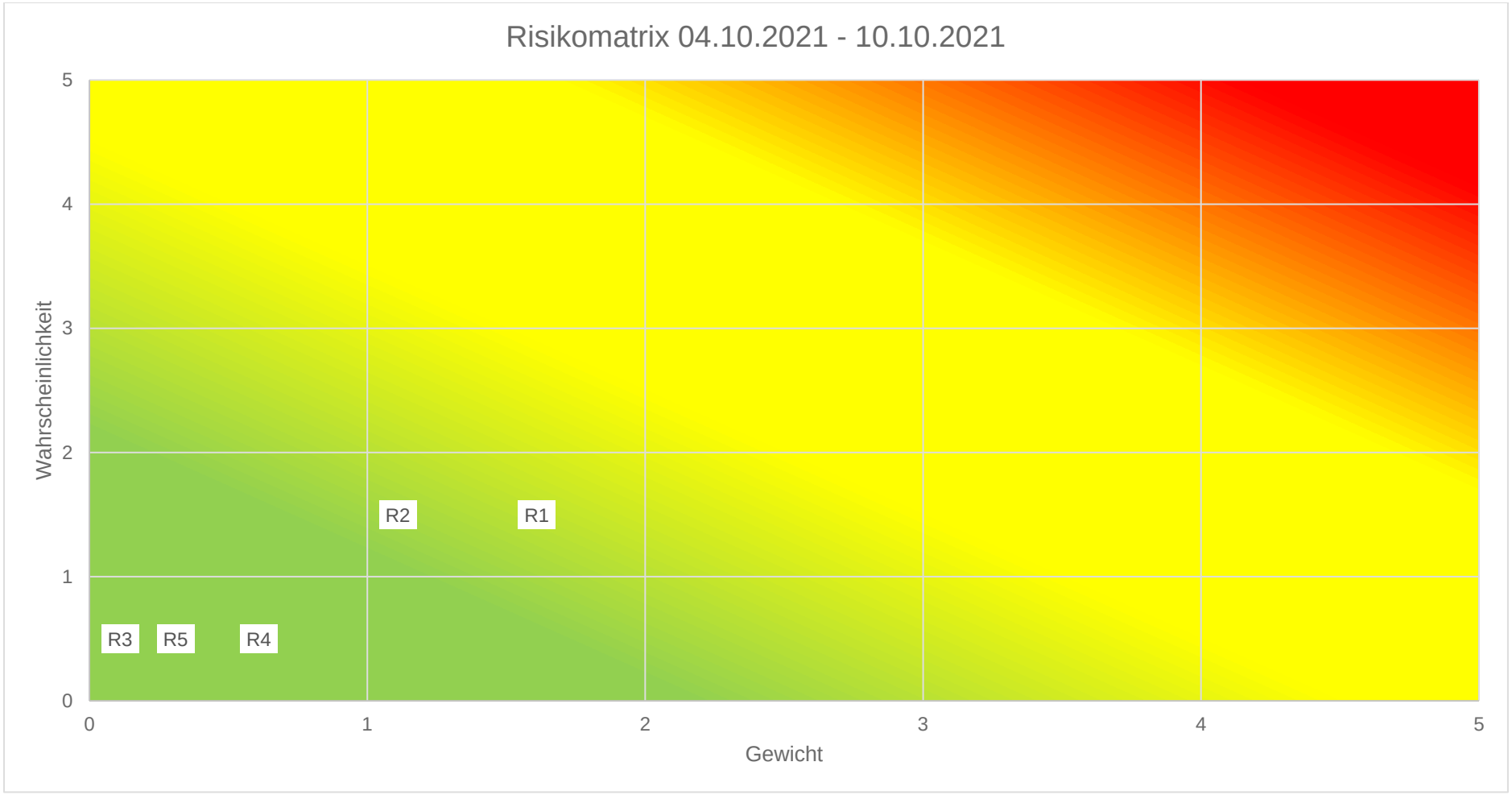
10.6.2.1 Risiko Auswertung vom 20.09.2021 – 26.09.2021

Nr	Titel	Beschreibung	Wahrscheinlichkeit	Gewicht	Vorbeugung	Verhalten beim Eintreten
R1	Neue Technologien	Backend (Java Play) und/oder Deployment (Docker) nimmt länger in Anspruch als geplant.	1.33	3	Vertiefung in Java Play und Docker	Pair-Programming, Unterstützung durch Mitstudenten
R2	Komplexität Frontend	Die Entwicklung der komplexen Komponenten im Frontend dauert länger als geplant.	3	4	Planung der komplexen Komponenten	Libraries verwenden oder mehr Zeit investieren
R3	IDP Compass Security	Die Einrichtung des IDP seitens Compass Security dauert länger als gedacht oder kann nicht wie besprochen eingebunden werden.	1.25	1	Entwicklung ohne Authentifizierung	Vorbereitung einer einfachen nachträglichen Anbindung
R4	Daten Import/Qualität	Eine schlechte Qualität der zu importierenden Daten erschwert den Import.	2.5	4	Datenbereinigung seitens Compass Security	Datenbereinigung beim Import
R5	Continuous Integration Java Play	Das Einrichten der Continuous Integration für das Java Play Projekt dauert länger als geplant	2.5	2	Informieren über Continuous Integration mit JavaPlay	Zusammenarbeit, Unterstützung durch Mitstudenten


Risikomatrix 20.09.2021 - 26.09.2021
Wahrscheinlichkeit
Gewicht
0
1
2
3
4
5
R1
R2
R3
R4
R5

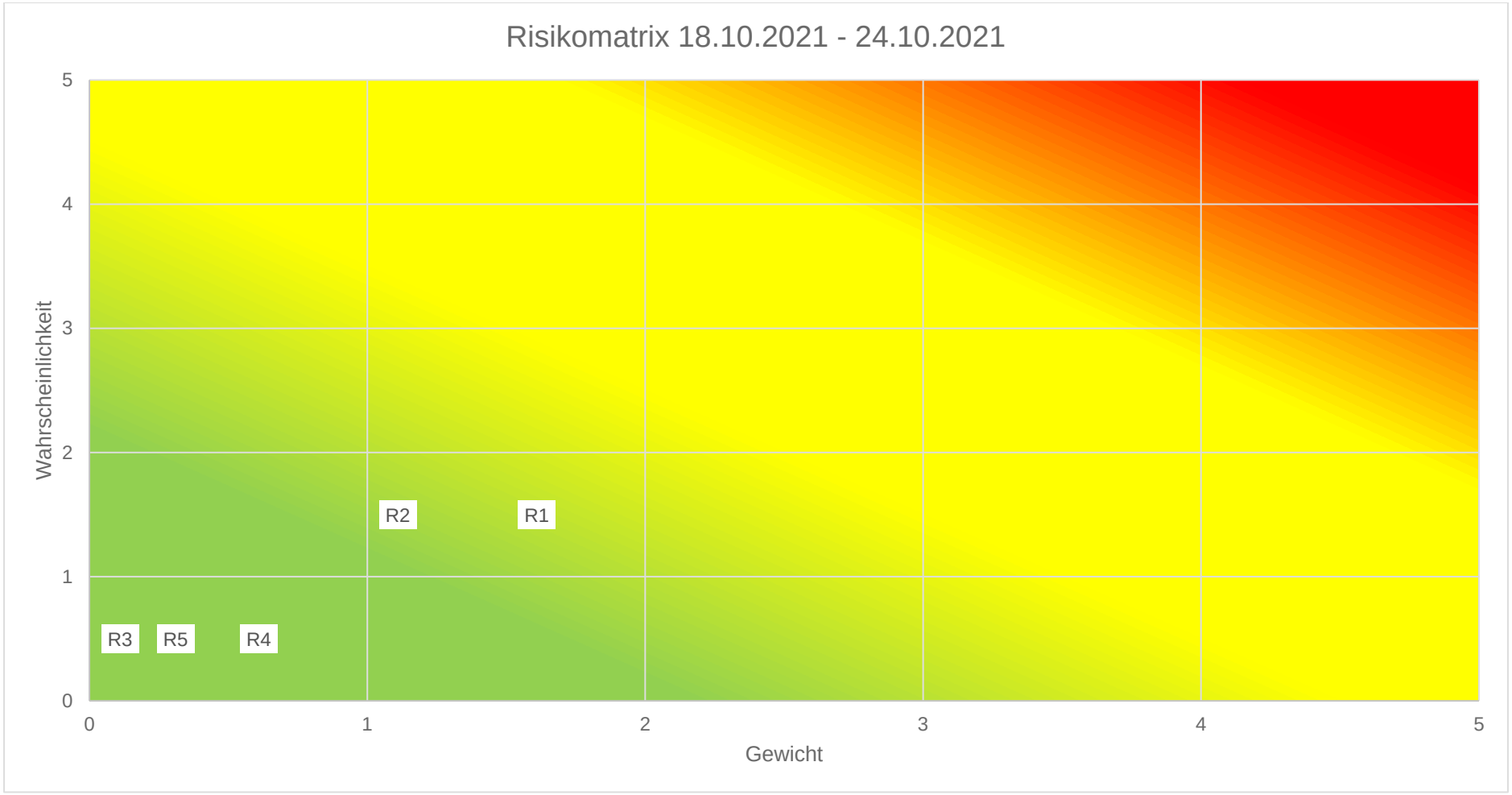
10.6.2.2 Risiko Auswertung vom 04.10.2021 – 10.10.2021

Nr	Titel	Beschreibung	Wahrscheinlichkeit	Gewicht	Vorbeugung	Verhalten beim Eintreten
R1	Neue Technologien	Backend (Java Play) und/oder Deployment (Docker) nimmt länger in Anspruch als geplant.	1.33	3	Vertiefung in Java Play und Docker	Pair-Programming, Unterstützung durch Mitstudenten
R2	Komplexität Frontend	Die Entwicklung der komplexen Komponenten im Frontend dauert länger als geplant.	3	**3**	Planung der komplexen Komponenten	Libraries verwenden oder mehr Zeit investieren
R3	IDP Compass Security	Die Einrichtung des IDP seitens Compass Security dauert länger als gedacht oder kann nicht wie besprochen eingebunden werden.	1.25	1	Entwicklung ohne Authentifizierung	Vorbereitung einer einfachen nachträglichen Anbindung
R4	Daten Import/Qualität	Eine schlechte Qualität der zu importierenden Daten erschwert den Import.	**2**	**2**	Datenbereinigung seitens Compass Security	Datenbereinigung beim Import
R5	Continuous Integration Java Play	Das Einrichten der Continuous Integration für das Java Play Projekt dauert länger als geplant	**1.5**	2	Informieren über Continuous Integration mit JavaPlay	Zusammenarbeit, Unterstützung durch Mitstudenten


Risikomatrix 04.10.2021 - 10.10.2021
Wahrscheinlichkeit
Gewicht
0
1
2
3
4
5
R1
R2
R3
R4
R5

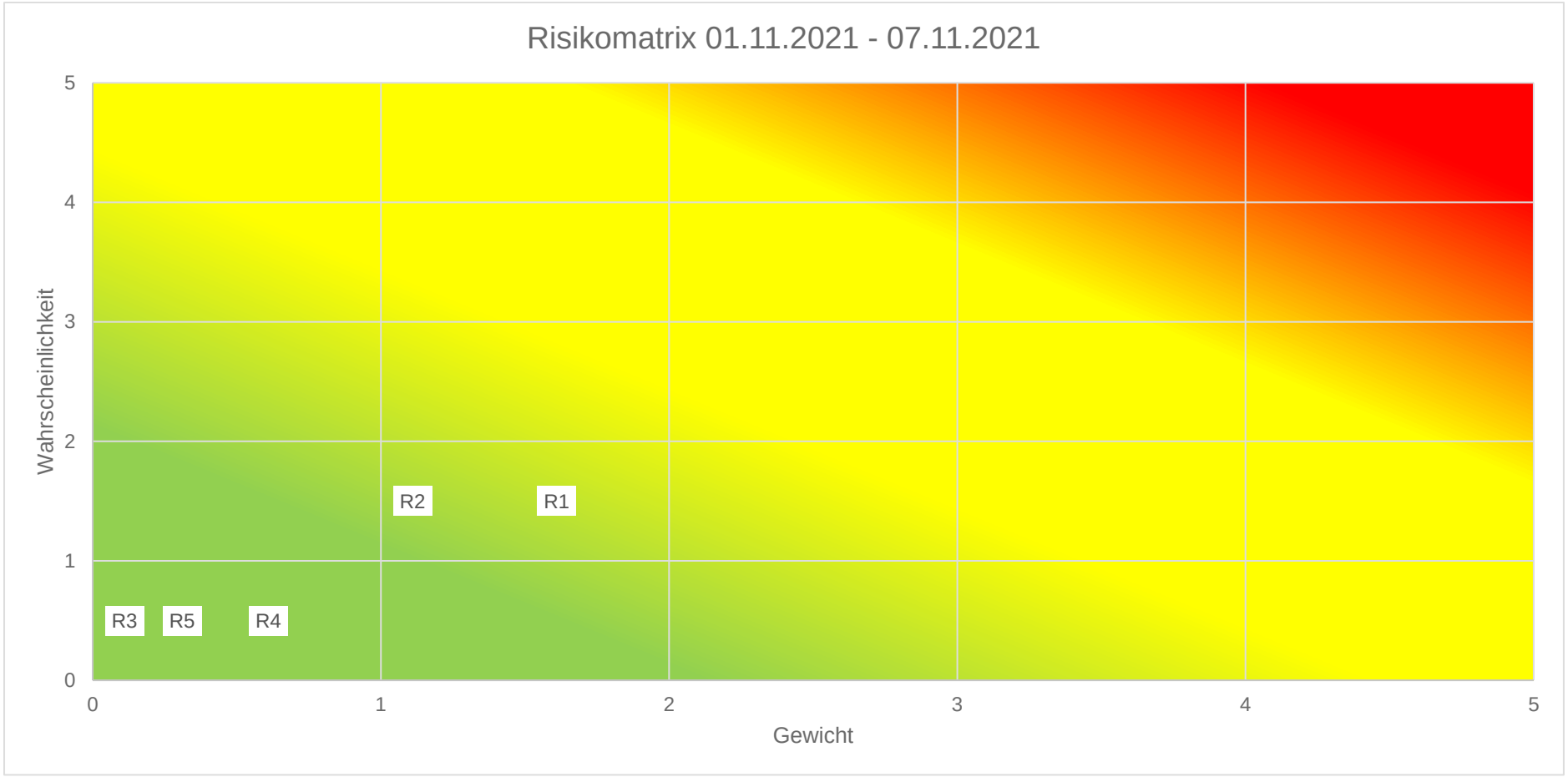
10.6.2.3 Risiko Auswertung vom 18.10.2021 – 24.10.2021

Nr	Titel	Beschreibung	Wahrscheinlichkeit	Gewicht	Vorbeugung	Verhalten beim Eintreten
R1	Neue Technologien	Backend (Java Play) und/oder Deployment (Docker) nimmt länger in Anspruch als geplant.	**1**	**2**	Vertiefung in Java Play und Docker	Pair-Programming, Unterstützung durch Mitstudenten
R2	Komplexität Frontend	Die Entwicklung der komplexen Komponenten im Frontend dauert länger als geplant.	**2**	**3**	Planung der komplexen Komponenten	Libraries verwenden oder mehr Zeit investieren
R3	IDP Compass Security	Die Einrichtung des IDP seitens Compass Security dauert länger als gedacht oder kann nicht wie besprochen eingebunden werden.	1.25	1	Entwicklung ohne Authentifizierung	Vorbereitung einer einfachen nachträglichen Anbindung
R4	Daten Import/Qualität	Eine schlechte Qualität der zu importierenden Daten erschwert den Import.	**0.5**	2	Datenbereinigung seitens Compass Security	Datenbereinigung beim Import
R5	Continuous Integration Java Play	Das Einrichten der Continuous Integration für das Java Play Projekt dauert länger als geplant	1.5	**1**	Informieren über Continuous Integration mit JavaPlay	Zusammenarbeit, Unterstützung durch Mitstudenten


Risikomatrix 18.10.2021 - 24.10.2021
Wahrscheinlichkeit
Gewicht
0
1
2
3
4
5
R1
R2
R3
R4
R5

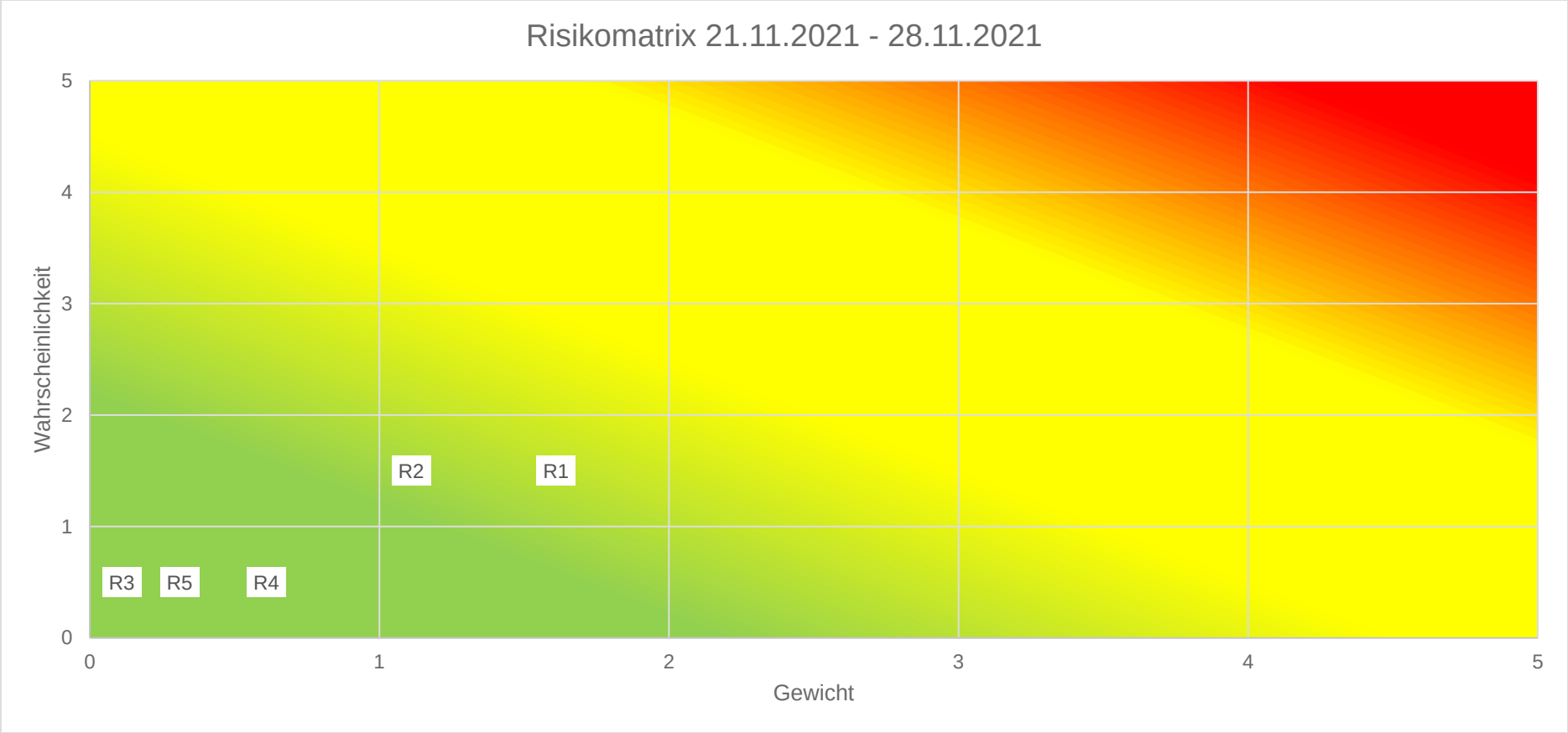
10.6.2.4 Risiko Auswertung vom 01.11.2021 – 07.11.2021

Nr	Titel	Beschreibung	Wahrscheinlichkeit	Gewicht	Vorbeugung	Verhalten beim Eintreten
R1	Neue Technologien	Backend (Java Play) und/oder Deployment (Docker) nimmt länger in Anspruch als geplant.	2	2.5	Vertiefung in Java Play und Docker	Pair-Programming, Unterstützung durch Mitstudenten
R2	Komplexität Frontend	Die Entwicklung der komplexen Komponenten im Frontend dauert länger als geplant.	2	2	Planung der komplexen Komponenten	Libraries verwenden oder mehr Zeit investieren
R3	IDP Compass Security	Die Einrichtung des IDP seitens Compass Security dauert länger als gedacht oder kann nicht wie besprochen eingebunden werden.	0	0	Entwicklung ohne Authentifizierung	Vorbereitung einer einfachen nachträglichen Anbindung
R4	Daten Import/Qualität	Eine schlechte Qualität der zu importierenden Daten erschwert den Import.	0.5	0.5	Datenbereinigung seitens Compass Security	Datenbereinigung beim Import
R5	Continuous Integration Java Play	Das Einrichten der Continuous Integration für das Java Play Projekt dauert länger als geplant	0	0.1	Informieren über Continuous Integration mit JavaPlay	Zusammenarbeit, Unterstützung durch Mitstudenten


Risikomatrix 01.11.2021 - 07.11.2021
Wahrscheinlichkeit
Gewicht
0
1
2
3
4
5
R1
R2
R3
R4
R5

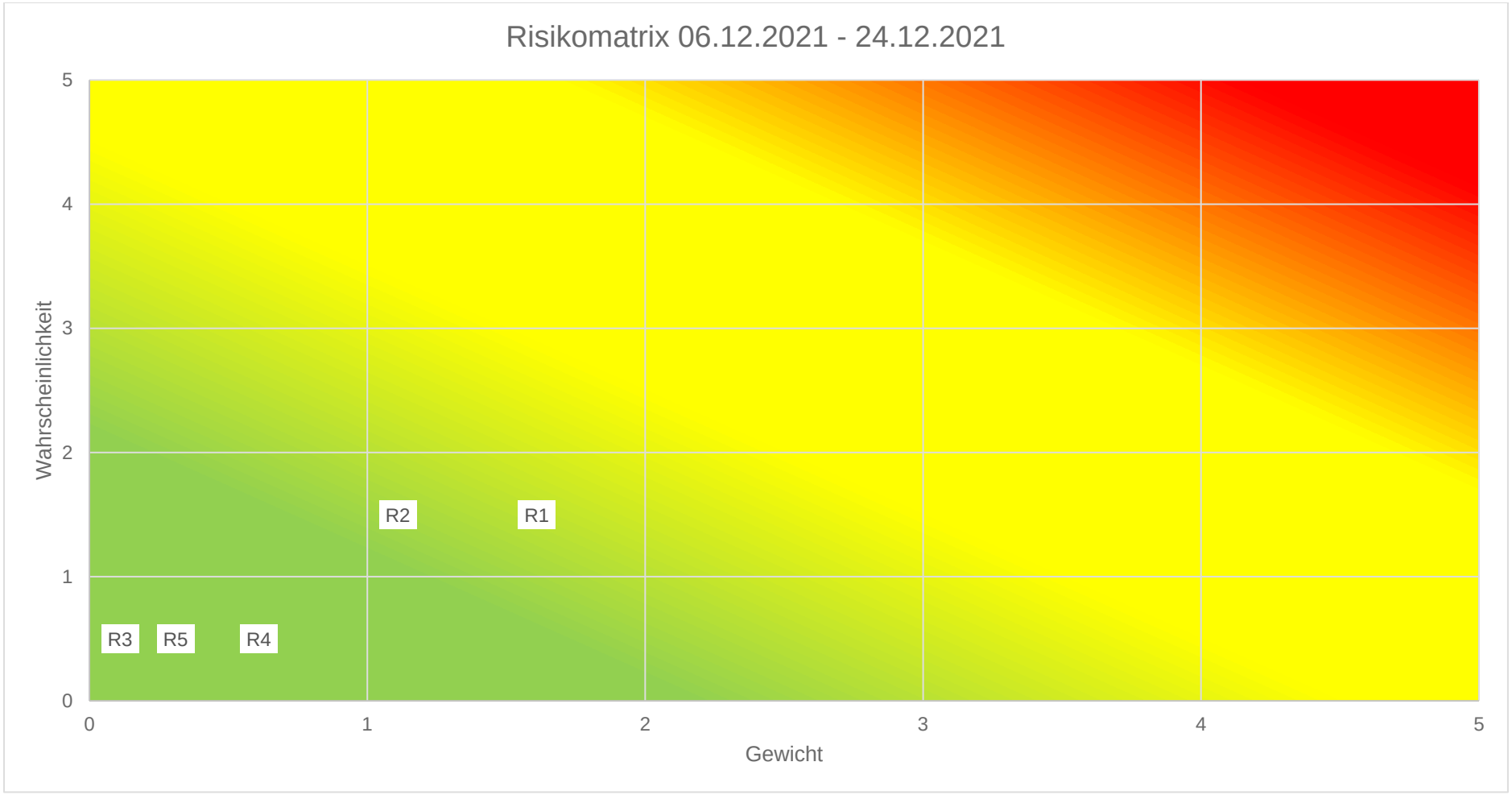
10.6.2.5 Risiko Auswertung vom 21.11.2021 – 28.11.2021

Nr	Titel	Beschreibung	Wahrschein-lichkeit	Gewicht	Vorbeugung	Verhalten beim Eintre-ten
R1	Neue Technologien	Backend (Java Play) und/oder Deployment (Docker) nimmt länger in Anspruch als ge-plant.	**1.5**	2.5	Vertiefung in Java Play und Docker	Pair-Programming, Unterstützung durch Mit-studenten
R2	Komplexität Frontend	Die Entwicklung der komple-xen Komponenten im Front-end dauert länger als geplant.	**1**	2	Planung der komplexen Komponenten	Libraries verwenden oder mehr Zeit investieren
R3	IDP Compass Security	Die Einrichtung des IDP sei-tens Compass Security dauert länger als gedacht oder kann nicht wie besprochen einge-bunden werden.	0	0	Entwicklung ohne Au-thentifizierung	Vorbereitung einer einfa-chen nachträglichen An-bindung
R4	Daten Import/Qualität	Eine schlechte Qualität der zu importierenden Daten er-schwert den Import.	**0**	0.5	Datenbereinigung sei-tens Compass Security	Datenbereinigung beim Import
R5	Continuous Integration Java Play	Das Einrichten der Continuous Integration für das Java Play Projekt dauert länger als ge-plant	0	**0.1**	Informieren über Conti-nuous Integration mit JavaPlay	Zusammenarbeit, Unter-stützung durch Mitstu-denten


Risikomatrix 21.11.2021 - 28.11.2021
Wahrscheinlichkeit
Gewicht
0
1
2
3
4
5
R1
R2
R3
R4
R5

10.6.2.6 Risiko Auswertung vom 06.12.2021 – 24.12.2021

Nr	Titel	Beschreibung	Wahrschein-lichkeit	Gewicht	Vorbeugung	Verhalten beim Eintre-ten
R1	Neue Technologien	Backend (Java Play) und/oder Deployment (Docker) nimmt länger in Anspruch als ge-plant.	1	1.5	Vertiefung in Java Play und Docker	Pair-Programming, Unterstützung durch Mit-studenten
R2	Komplexität Frontend	Die Entwicklung der komple-xen Komponenten im Front-end dauert länger als geplant.	1	1	Planung der komplexen Komponenten	Libraries verwenden oder mehr Zeit investieren
R3	IDP Compass Security	Die Einrichtung des IDP sei-tens Compass Security dauert länger als gedacht oder kann nicht wie besprochen einge-bunden werden.	0	0	Entwicklung ohne Au-thentifizierung	Vorbereitung einer einfa-chen nachträglichen An-bindung
R4	Daten Import/Qualität	Eine schlechte Qualität der zu importierenden Daten er-schwert den Import.	0	0.5	Datenbereinigung sei-tens Compass Security	Datenbereinigung beim Import
R5	Continuous Integration Java Play	Das Einrichten der Continuous Integration für das Java Play Projekt dauert länger als ge-plant	0	0.1	Informieren über Conti-nuous Integration mit JavaPlay	Zusammenarbeit, Unter-stützung durch Mitstu-denten


Risikomatrix 06.12.2021 - 24.12.2021
Wahrscheinlichkeit
Gewicht
0
1
2
3
4
5
R1
R2
R3
R4
R5

11. Projektmonitoring

11.1 Burndown-Diagramm

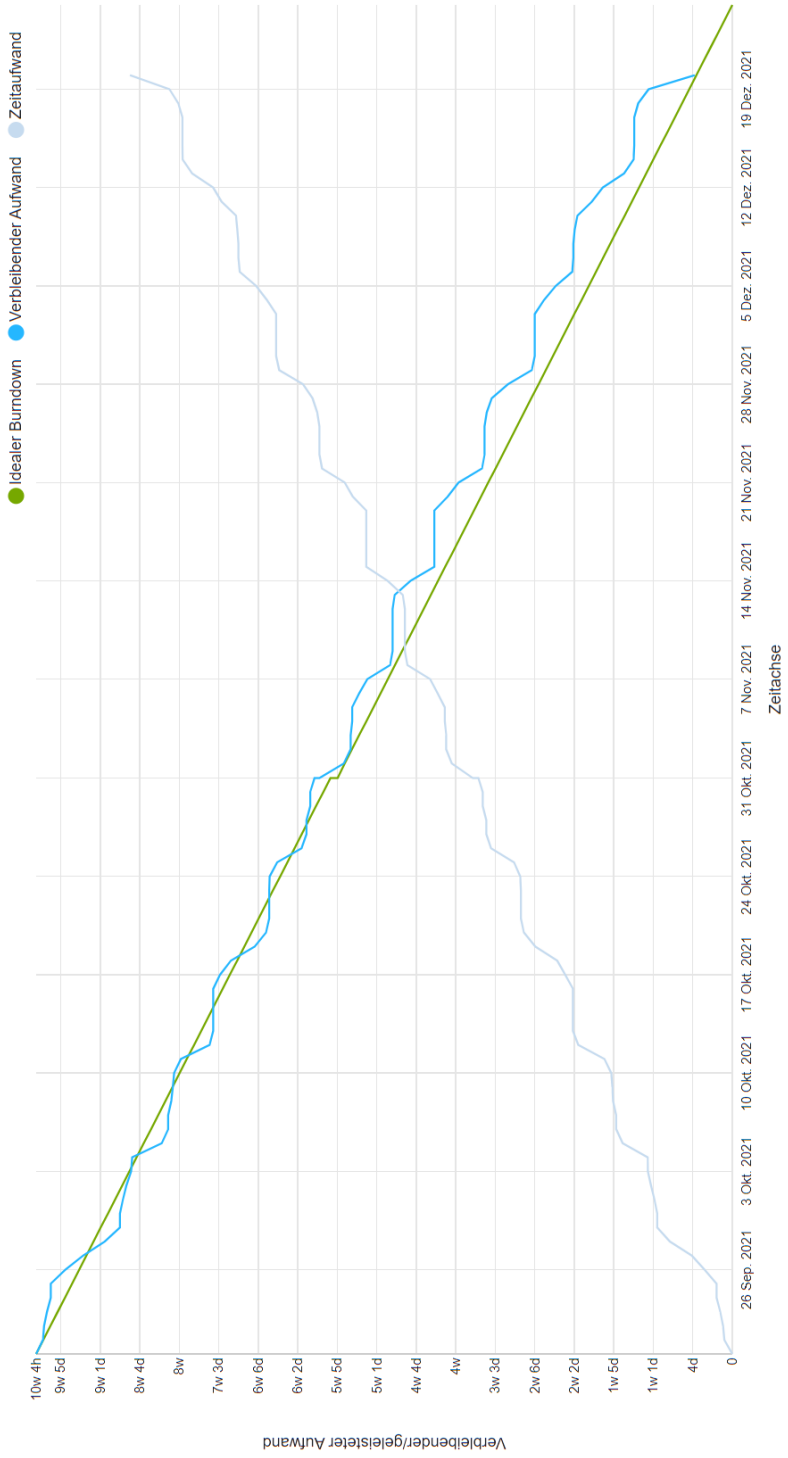


Abbildung 27: Burndown-Diagramm

11.2 Aufwandverteilung

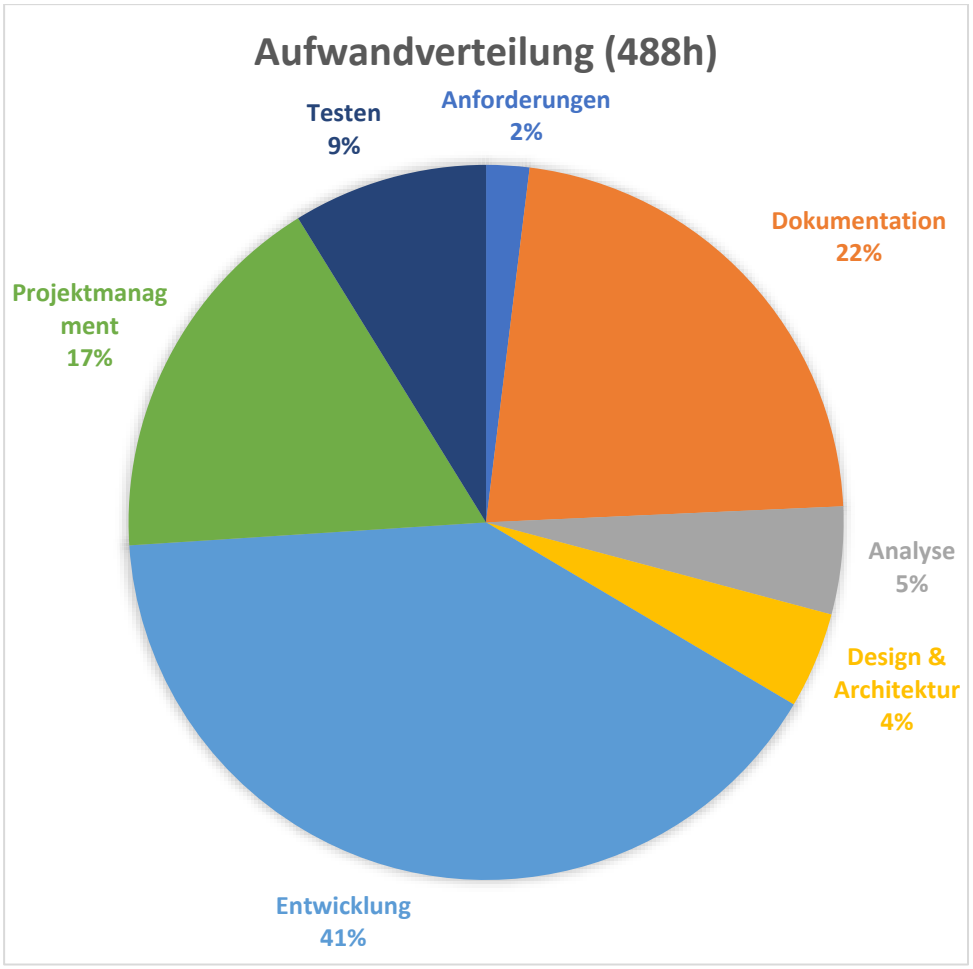


Abbildung 28: Aufwandverteilung der 488 Stunden

Ausführungen zu den Verhältnissen:

- Entwicklung > Testen
  - Während der Entwicklung wurden oftmals auch Tests geschrieben oder angepasst, diese Aufwände flossen, aber gemäss unseren Absprachen in die Entwicklung, solange das Hauptziel eines Tickets die Entwicklung betrifft.
  - Die dedizierten Aufwände der Kategorie «Testen» entstanden vor allem durch die dedizierte Fokussierung auf Tests mit Tickets wie zum Beispiel die Testabdeckung zu erhöhen oder Integrationstests zu schreiben.
- Projektmanagement > Anforderungen
  - Der Aufwand von der Kategorie «Anforderungen» wirkt besonders klein, da sie jedoch weitaus mehr Zeit benötigt hat.
    Grund dafür ist, dass die Anforderungen mehrheitlich während Meetings mit Cyrill Brunschwiler erarbeitet und dokumentiert wurden. Da jegliche Meeting-Dauer auf Projektmanagement gebucht wurde, erklärt dies den vermeintlich kleinen Aufwand für die Kategorie «Anforderungen».

11.3 Code-Metriken

In den folgenden Kapiteln sind die Metriken des Source-Codes aufgeführt.
Die Werte zeigen die Anzahl Zeilen ohne Kommentare und ohne Leerzeilen.

11.3.1 Frontend

	Eigenleistung	Tests	Fremdleistung oder adaptiert	Gesamt
TypeScript	1’515 (+1’189 Libraries)	1’122	146	3’972
Sass	320	0	331	651
HTML	498	0	7	505

Die 1189 Zeilen Code aus den Libraries (nicht Third-Party Libraries) wurden speziell ausgewiesen, da ein Teil dieses Codes generiert wurde und nach und nach ständig angepasst und erweitert wurde.

11.3.2 Backend

	Eigenleistung	Tests	Fremdleistung oder adaptiert	Gesamt
Java	1’709	690	307	2’706
XML	0	0	47	47
SQL	79 (generiert)	0	0	79

Das Backend besteht aus 59 Klassen. Davon sind 14 Test-Klassen und 7 Klassen beinhalten Code, welcher entweder als Fremdleistung gekennzeichnet ist oder der Code von den offiziellen Play Framework-Templates adaptierter wurde.

12. Glossar & Abkürzungsverzeichnis

A

Absence
Technische Bezeichnung einer Absenz. (z.B. Ferien, Militär, Kompensation) 24, 25, 26, 33
Angular
Ein TypeScript-basiertes Front-End-Webapplikationsframework..... 13, 16, 21, 22, 27, 28, 29, 37, 51, 55
Assignment
Zuweisung eines Mitarbeiters auf ein Projekt 24, 25, 26, 30, 33, 35, 38, 44, 47, 49, 50, 53, 54

C

CD
Continous Deployment 55
CI
Continous Integration 12, 31, 46, 47, 55

D

Docker
Open-Source Software zur Isolation von Anwendungen in virtuellen Containern..... 22, 31, 58, 60, 62, 64, 66, 68

E

Employee
Technische Bezeichnung eines Mitarbeiters der Compass Security 24, 25, 26, 33

H

Holiday
Technische Bezeichnung eines Feiertags 24, 25, 26, 33

I

IDP
Identity Provider..... 18, 22, 58, 60, 62, 64, 66, 68

J

Java
Eine objektorientierte Programmiersprache.....13, 16, 22, 55, 56, 58, 60, 62, 64, 66, 68, 72

L

Location
Technische Bezeichnung eines Standorts 24, 25

M

MVP
Minimum Viable Product 12, 15, 18, 21, 23, 26, 51, 53
MySQL
Ein relationales Datenbankverwaltungssystem. 13, 22, 56

P

Play Framework
Ein auf Scala basiertes Open-Source-Webanwendungs-Framework. ... 13, 16, 22, 30, 32, 51, 55, 56, 58, 60, 62, 64, 66, 68, 72
Project
Technische Bezeichnung eines Projekts der Compass Security 24, 25, 26, 33, 35

S

Scala
Eine funktionale und objektorientierte Programmiersprache 16, 31
SSO
Single Sign-on 18, 39

U

Unit
Technische Bezeichnung einer Abteilung 24, 25, 26

W

WebSocket
Ein Netzwerkprotokoll für bidirektionale Verbindungen 13, 30, 36, 40, 42, 44, 45, 48, 51, 53, 56

13. Literatur- & Quellenverzeichnis

[1] Factro, [Online]. Available: https://www.factro.de/. [Zugriff am 09 10 2021].

[2] Trello, [Online]. Available: https://trello.com/. [Zugriff am 09 10 2021].

[3] Microsoft, [Online]. Available: https://www.microsoft.com/de-ch/microsoft-365/business/task-management-software. [Zugriff am 09 10 2021].

[4] OpenProject, [Online]. Available: https://www.openproject.org/de/. [Zugriff am 09 10 2021].

[5] ISO25000, «ISO 25000,» 2021. [Online]. Available: https://iso25000.com/index.php/en/iso-25000-standards/iso-25010. [Zugriff am 27 September 2021].

[6] «OWASP TOP 10:2021,» [Online]. Available: https://owasp.org/Top10/. [Zugriff am 1 10 2021].

[7] A. Thehos, «at Excel-Blog – Andreas Thehos,» [Online]. Available: https://thehosblog.com/2019/02/22/risikomatrix-in-excel/. [Zugriff am 27 September 2021].