Personalplanung für den Helpdesk

Individuelle Praktische Arbeit 2019

Projektdaten

Kunde Basler Versicherungen Corporate IT (Helpdesk)

Auftraggeber Matthias Cullmann

Autor Elia Reutlinger

Ort Basel-Stadt

Zeitraum 24.04. – 10.05.2019

Version [**TODO**]

Status In Arbeit [**TODO**]

Beteiligte Personen

Anwender Corporate IT Helpdesk

Prüfung Matthias Cullmann *Verantwortliche Fachkraft*

Beat Sommer *Hauptexperte*

Felix Tobler *Nebenexperte*

Genehmigung Tschan Günter *Validierungsexperte*

Dokumentenmanagement

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Datum | Autor | Beschreibung |
| 0.1 | 24.04.2019 | Elia R. | Dokument erstellt, Hauptkapitel notiert, Arbeitsjournal erstellt. |
| 0.1.1 | 25.04.2019 | Elia R. | Grundlegende Kapitel erweitert. |
| 0.2 | 02.05.2019 | Elia R. | Detaillierte Aufgabenstellung eingefügt und Analyse geschrieben. Kapitel Organisation und Firmenstandards begonnen. |
| 0.3 | 03.05.2019 | Elia R. | Kapitel «Projektmanagement & Planung» beendet, «Auswertung der Aufgabenstellung» beendet, Kapitel «Initialisierung» begonnen, zahlreiche Verbesserungen (Details & Grammatik). |
| 0.4 | 07.05.2019 | Elia R. | Kapitel «Initialisierung» abgeschlossen, Grafiken hinzugefügt. |
| 0.5 | 07.05.2019 | Elia R. | Testkonzept verbessert, Kapitel «Realisierung» begonnen. |
| 0.6 | 08.05.2019 | Elia R. |  |

Glossar

[**TODO** (ganz am Ende)]

Inhaltsverzeichnis

[Personalplanung für den Helpdesk 1](#_Toc8132056)

[1 Kurzfassung 5](#_Toc8132057)

[1.1 Ausgangssituation 5](#_Toc8132058)

[1.2 Ziel 5](#_Toc8132059)

[1.3 Umsetzung 5](#_Toc8132060)

[1.4 Ergebnis 5](#_Toc8132061)

[Teil 1: Ablauf und Umfeld 6](#_Toc8132062)

[2 Detaillierte Aufgabenstellung 6](#_Toc8132063)

[2.1 Ausgangslage 6](#_Toc8132064)

[2.2 Detaillierte Aufgabenstellung 6](#_Toc8132065)

[2.3 Tests 8](#_Toc8132066)

[2.4 Mittel und Methoden 9](#_Toc8132067)

[2.5 Vorkenntnisse 9](#_Toc8132068)

[2.6 Vorarbeiten 9](#_Toc8132069)

[2.7 Neue Lerninhalte 10](#_Toc8132070)

[2.8 Arbeiten in den letzten 6 Monaten 10](#_Toc8132071)

[3 Analyse der Aufgabenstellung 11](#_Toc8132072)

[3.1 Kurzfassung Ist/Soll 11](#_Toc8132073)

[3.2 Messbare Ziele 11](#_Toc8132074)

[3.3 Dokumentation 11](#_Toc8132075)

[4 Projektmanagement und Planung 12](#_Toc8132076)

[4.1 Organisation 12](#_Toc8132077)

[4.1.1 Zeitplan & Termine 12](#_Toc8132078)

[4.1.2 Projektaufbauorganisation 13](#_Toc8132079)

[4.1.3 Arbeitsplatz 14](#_Toc8132080)

[4.1.4 Datensicherung & Versionierung 14](#_Toc8132081)

[4.2 Auswertung der Aufgabenstellung 16](#_Toc8132082)

[4.2.1 Wandlung der vorgegebenen Tests 16](#_Toc8132083)

[4.2.2 Mittel und Methoden 16](#_Toc8132084)

[4.2.3 Abhängigkeiten zu Vorarbeiten 17](#_Toc8132085)

[5 Firmenstandards 18](#_Toc8132086)

[5.1 Schriftarten 18](#_Toc8132087)

[5.2 Farben 18](#_Toc8132088)

[5.3 Logos 19](#_Toc8132089)

[5.4 Formularelemente 19](#_Toc8132090)

[6 Arbeitsjournal 20](#_Toc8132091)

[6.1 Tag 1 – 24.04.2019 20](#_Toc8132092)

[6.2 Tag 2 – 25.04.2019 21](#_Toc8132093)

[6.3 Tag 3 – 26.04.2019 22](#_Toc8132094)

[6.4 Tag 4 – 30.04.2019 23](#_Toc8132095)

[6.5 Tag 5 – 02.05.2019 24](#_Toc8132096)

[6.6 Tag 6 – 03.05.2019 25](#_Toc8132097)

[6.7 Tag 7 – 07.05.2019 26](#_Toc8132098)

[6.8 Tag 8 – 08.05.2019 27](#_Toc8132099)

[6.9 Tag 9 – 09.05.2019 28](#_Toc8132100)

[6.10 Tag 10 – 10.05.2019 29](#_Toc8132101)

[7 Abschlussbericht 30](#_Toc8132102)

[7.1 Ist-Soll 30](#_Toc8132103)

[7.2 Schwierigkeiten 30](#_Toc8132104)

[7.3 Fazit zum Projekt 30](#_Toc8132105)

[7.3.1 Entstandene Abweichungen zur Aufgabenstellung 30](#_Toc8132106)

[7.4 Persönliches Fazit 30](#_Toc8132107)

[7.5 Schlussreflexion 30](#_Toc8132108)

[7.5.1 Arbeitsmethodik 30](#_Toc8132109)

[7.5.2 Werkzeuge 30](#_Toc8132110)

[Teil 2: Dokumentation und Umsetzung 31](#_Toc8132111)

[8 Initialisierung 31](#_Toc8132112)

[8.1 Analyse Vue.js 31](#_Toc8132113)

[8.1.1 Konzept 32](#_Toc8132114)

[8.1.2 Vergleich 32](#_Toc8132115)

[8.2 Vertiefte Studie Ist-Zustand & Umgebung 33](#_Toc8132116)

[8.3 Umsetzung Projektmethode & Vorgehensmodell 35](#_Toc8132117)

[8.4 Anforderungen 37](#_Toc8132118)

[8.4.1 Funktional 38](#_Toc8132119)

[8.4.2 Nicht-Funktional 38](#_Toc8132120)

[8.5 Ziele 39](#_Toc8132121)

[8.5.1 Projekt 39](#_Toc8132122)

[8.5.2 Persönlich 39](#_Toc8132123)

[8.6 Informationssicherheit & Datenschutz 39](#_Toc8132124)

[8.7 Konzepte 40](#_Toc8132125)

[8.7.1 Testkonzept (Use-Cases) 40](#_Toc8132126)

[8.7.2 Sicherheitskonzept 43](#_Toc8132127)

[9 Realisierung 47](#_Toc8132128)

[9.1 Grundlegende Konfiguration 47](#_Toc8132129)

[9.1.1 Ansichten / Routes 47](#_Toc8132130)

[9.1.2 Mehrsprachigkeit 48](#_Toc8132131)

[9.1.3 Authentifizierung 48](#_Toc8132132)

[9.2 Umsetzung Design 50](#_Toc8132133)

[9.3 Umsetzung der Ansichten 50](#_Toc8132134)

[9.3.1 Support 50](#_Toc8132135)

[9.3.2 Dashboard 50](#_Toc8132136)

[9.3.3 Einstellungen 50](#_Toc8132137)

[9.3.4 Plan-Einstellungen 51](#_Toc8132138)

[9.3.5 Dienstplan 51](#_Toc8132139)

[9.4 Testprotokoll 51](#_Toc8132140)

[9.5 Sicherheitsprüfung 51](#_Toc8132141)

[10 Abschluss 51](#_Toc8132142)

[10.1 Auswertung des Kanban-Boards 51](#_Toc8132143)

[10.2 Funktionen & Verbesserungen für die Zukunft 51](#_Toc8132144)

[10.3 Danksagung 51](#_Toc8132145)

[11 Anhang 52](#_Toc8132146)

[11.1 Quellenverzeichnis 52](#_Toc8132147)

[11.2 Abbildungsverzeichnis 52](#_Toc8132148)

# Kurzfassung

## Ausgangssituation

## Ziel

## Umsetzung

## Ergebnis

Teil 1: Ablauf und Umfeld

# Detaillierte Aufgabenstellung

Folgend die Definition der Aufgabenstellung, welche als «detaillierte Aufgabenstellung» auf pkorg.ch erfasst wurde.

## Ausgangslage

Der IT-Helpdesk der Baloise Group besteht aus einem Team von ca. 30 Mitarbeiter und erbringt Dienstleistungen in 5 Sprachen für die Ländergesellschaften der Baloise Group in Belgien, Luxemburg, Deutschland und der Schweiz.

Aktuell werden die Einsätze der Mitarbeiter in einer Excel-Tabelle geplant. Diese Tabelle soll in eine Webanwendung umgewandelt werden, damit die Inhalte mit wenig Aufwand anpassbar sowie jederzeit von überall abrufbar sind. Ebenso sollen Bedienungsfehler und Datenverluste vermieden werden.

Der Helpdesk der Baloise befindet sich, wie auch die Softwareentwicklung Schweiz, in deren Rahmen die IPA durchgeführt wird, am Hauptsitz der Baloise, Aeschengraben 21, 4051 Basel.

## Detaillierte Aufgabenstellung

Grundlegende Informationen

Resultat der Aufgabe soll eine Webanwendung zur Einsatzplanung am Helpdesk sein. Damit ist die Entwicklung eines Front-Ends zu einem bereits bestehenden Back-End (einer Rest-API) gemeint. Dieses Back-End sowie die dazugehörige Datenbank werden bereits vor der IPA geplant und umgesetzt. Die Anwendung soll von mehreren Benutzern genutzt werden können, wobei sich diese in Teams unterteilen und durch Rollen unterschiedliche Berechtigungen haben.

Struktur der Benutzerverwaltung

Im Anschluss an die IPA soll in der Anwendung die Verwaltung von Teams implementiert werden. Für die IPA und zu Demo-Zwecken soll jedoch nur ein vordefiniertes Team «Helpdesk» bestehen. Die Teamverwaltung ist also während der IPA nicht umzusetzen.

Ein Team beinhaltet mindestens einen Teamleiter und ein Mitglied. Teamleiter können weitere Mitglieder hinzufügen, deren Rolle (Teamleiter oder Mitglied) bearbeiten, und Mitglieder entfernen. Teamleiter und Mitglieder haben jeweils nur Zugriff auf Daten, welche das eigene Team betreffen.

Das hinzufügen neuer Mitglieder soll anhand deren interner Benutzerkennung geschehen, worauf während der IPA aber noch nicht zugegriffen werden kann. Deshalb sollen für die IPA und zu Demo-Zwecken einige vordefinierte Mitglieder definiert werden, wobei man diese noch nicht durch das Front-End erweitern kann. Die Funktion für das Hinzufügen neuer Mitglieder ist also während der IPA nicht umzusetzen.

Struktur der Anwendung

In der Anwendung können die Einsätze von Mitgliedern eines Teams von deren Teamleitern geplant werden. Dazu gehören Schichten (Was zu tun ist, z.B. Vor-Ort Support, Telefon-Support, ...) und Tageszeiten (Wann es zu tun ist, z.B. Morgens, Mittags, Abends, …). Ein Einsatz ist die Zuordnung eines Mitglieds zu einer Schicht und einer Tageszeit an einem bestimmten Datum (z.B. Herr Maier, Telefon-Support, Mittags, 12.12.2019).

Diese Anforderungen sowie die der Benutzerverwaltung fordern also mindestens folgende Ansichten (mit geschätzter Implementierungsdauer):

* Eine Tageszeiten-Ansicht, in welcher Teamleiter die verfügbaren Tageszeiten bearbeiten können. (4h)
* Eine Schichten-Ansicht, durch welche Teamleiter die verfügbaren Schichten bearbeiten können (Möglicherweise zusammen mit der Tageszeiten-Ansicht zu implementieren). (4h)
* Eine Einsatz-Ansicht, auf welcher alle Einsätze von Mitgliedern eingesehen und von Teamleitern bearbeitet werden können. (8h)
* Eine Einstellungen-Ansicht, in welcher ein Teamleiter die Rollen der Mitglieder bearbeiten kann und jeder Benutzer die Sprache seiner Oberfläche anpassen kann. (4h)
* Eine einfache Login-Ansicht (Details folgen) für Demo-Zwecke. (2h)

Summe der Aufwände ist 22H, mit Projektkonfiguration und Recherchen ca. 26H.

Authentifizierung

Im Anschluss an die IPA wird die Anwendung in der Baloise intern aufgesetzt. Dabei kommt für die Authentifizierung eine bereits vorhandene (Single Sign-On) Infrastruktur zum Einsatz, weshalb im Front-End bzw. während der IPA keine Login- und Registrierungs-Maske implementiert werden soll. Wie bereits beschrieben soll es für Demo-Zwecke bereits vordefinierte Teams und Benutzer geben, wobei eine möglichst einfache Ansicht für den Login erstellt werden soll. Damit soll man sich durch auswählen eines vordefinierten Benutzers mit dessem Profil anmelden können.

Die im Voraus implementierte Rest-API arbeitet mit JWT (Json Web Token), welche vom Front-End nur noch verwaltet werden müssen.

Weitere Kriterien

Die Anwendung soll Mobile-Fähig sein. Das bedeutet, dass sie auch im Browser eines Smartphones problemlos dargestellt werden kann, und alle Funktionen hier gleich funktionieren. Die Gestaltung der Oberfläche kann abhängig vom Inhalt beliebig sein, wobei die Farben und Logos von den Corporate Design Richtlinien der Baloise (https://webstyleguide.baloise.com/) abgeleitet werden sollen. Da die Mitarbeiter des Helpdesks unterschiedliche Sprachen sprechen soll die Anwendung in 2 Sprachen zur Verfügung stehen: Englisch und Deutsch. Die Sprache soll vom Benutzer selbst angepasst werden können.

Abgrenzungen

Es ist zu beachten, dass für die IPA nur das Front-End umzusetzen ist. Dies bedeutet, dass jegliche Arbeiten bezüglich Datenbank und Back-End bereits im Voraus erledigt sind und keinen fachlichen Bezug zur IPA haben. Dies betrifft also Details zu Attributen in der Datenbank und deren Struktur, sowie weiterer Parameter. Ebenso gehören dazu die Berücksichtigung von Sicherheit- und möglichen Programmier-Standards im Back-End.

Projektmethode

Da die Implementierung als Teil der IPA zeitlich begrenzt ist und nicht viel Raum für Besprechungen und Abklärungen bleibt, entschieden wir uns für die Projektmethode «Kanban». Die Umsetzung dieser wird in Verbindung mit GitHub Issues online abgehalten. Dadurch kann der Projektfortschritt und der anstehende Arbeitsaufwand jederzeit abgerufen werden.

## Tests

Zur Prüfung der korrekten Funktion der Anwendung sollen manuelle Tests definiert und mit einem Protokoll ausgeführt und festgehalten werden. Notwendige Test, welche die vorausgesetzten Funktionen der Anwendung bestätigen, sind folgende:

Einsatz planen

Es ist einem Teamleiter möglich, den Einsatz eines Mitglieds zu planen. Dazu gehört die Auswahl einer Schicht in einer Tageszeit. Die Daten werden anschliessend korrekt verarbeitet und im Back-End gespeichert.

Einsatz bearbeiten

Es ist einem Teamleiter möglich, vorhandene Einsätze zu verändern (die Schicht anzupassen) und zu löschen. Die Daten werden anschliessend korrekt verarbeitet und im Back-End gespeichert.

Einsätze darstellen

Es soll allen Mitgliedern eines Teams möglich sein, die Einsätze von sich selbst und der anderen einzusehen. Dies in einer dafür sinnvoll formatierten Tabelle.

Schichten & Tageszeiten bearbeiten

Es soll einem Teamleiter möglich sein, die Schichten und Tageszeiten zu erweitern, bearbeiten und löschen. Falls es Abhängigkeiten zu geplanten Einsätzen gibt, sollen die betroffenen Einsätze gelöscht werden. Die Daten werden anschliessend korrekt verarbeitet und im Back-End gespeichert.

Rollen der Mitglieder bearbeiten

Es soll einem Teamleiter möglich sein, die Rollen (Teamleiter oder Mitglied) der anderen Mitglieder anzupassen. Die Daten werden anschliessend korrekt verarbeitet und im Back-End gespeichert.

Sprache anpassen

Alle Nutzer der Anwendung sollen die dargestellte Sprache anpassen können. Die Auswahl soll dabei aus Deutsch und Englisch bestehen. Die Daten werden anschliessend korrekt verarbeitet und im Back-End gespeichert.

## Mittel und Methoden

Hardware

* Notebook

Software

* Atom (Code-Editor)
* Git
* GitHub Desktop
* Vue CLI
* XAMPP (Lokale Back-End Umgebung)
* Google Chrome
* Word, Excel, Paint.Net (Grafiken)

Dienste

* Github, Waffle.io (Kanban)
* Netlify (für Demo-Zwecke)

Sprachen & Abhängigkeiten

* HTML, CSS, JavaScript, XML, JSON
* VueJS, Vuetify (Design), i18n (Übersetzungen), Axios (XMLHttpRequests),…

Firmenvorgaben

* Farbpalette der Corporate-Design Richtlinien
* Firmenlogos

Methoden

* Kanban (Waffle.io)

## Vorkenntnisse

Der Lernende hat bereits alle erwähnten Mittel und Methoden mindestens 1 Mal angewendet. VueJS ist jedoch bisher am wenigsten vertieft worden, während die Kenntnisse in allen anderen Punkten sehr gefestigt sind. Der Lernende arbeitet bereits seit mehreren Jahren mit den meisten Mitteln und mit VueJS seit einigen Monaten.

## Vorarbeiten

Rest-API

Während der IPA wird eine im Voraus umgesetzte Rest-API (Representational State Transfer Application Programming Interface) zur Verfügung stehen, welche hauptsächlich vom Lernenden mit PHP und MySQL implementiert wurde. Sessions sind somit Stateless wobei für die Validierung JWT (JSON Web Token) verwendet wird. Damit steht bereits eine Struktur der Daten und möglichen Abfragen fest, an welcher sich die Umsetzung der IPA orientieren muss.

Git-Repository

Um in der Durchführungs-Phase der IPA direkt mit der Entwicklung starten zu können, wird bereits im Voraus ein Git-Repository auf GitHub eingerichtet. Das Repository wird unter der firmeneigenen Organisation «Baloise» erstellt und «pelan-ui» genannt. Das Urheberecht gehört somit der Baloise unter der Verwendung einer MIT-Lizenz.

Kanban-Board

Mit dem Git-Repository wird auch ein Kaban-Board eingerichtet, welches voraussichtlich mit der Integration von Waffle.io zustande kommt. Es werden auch bereits die Anforderungen in Form von GitHub-Issues eingetragen.

## Neue Lerninhalte

Möglicherweise werden während der Entwicklung weitere Abhängigkeiten benötigt, zu welchen sich der Lernende in den entsprechenden Dokumentationen informieren muss. Zu den bekannten Mitteln kann sich der Lernende im Internet auf den zugehörigen Webseiten oder in Forums informieren. Die Umsetzung der Arbeit sollte jedoch nicht an einer unerwartet grossen, neuen Kompetenz scheitern, da es sich hierbei nur um kleinere Abhängigkeiten handeln wird.

## Arbeiten in den letzten 6 Monaten

In den letzten Monaten bestanden die Arbeiten des Lernenden aus hauptsächlich Front-End lastigen Arbeiten im Bereich Web-Development. Dabei wurden folgende Techniken genutzt: HTML, CSS, JavaScript, PHP, SQL, REST, Java, JavaScript-Libraries (JQuery, VueJS, Bootstrap,…).

Ein grosses Projekt war dabei der «Cash-Calculator», welcher den Leistungslohn aller Lernenden der Baloise berechnet. Dies geschieht anhand unterschiedlicher Parameter wie Noten und Verhalten.

Der «Cash-Calculator» steht allen Lernenden und der Personalabteilung zur Verfügung, und wird regelmässig genutzt. Für die Umsetzung wurden hier PHP und JavaScript (JQuery) genutzt, wobei der lernende auch in Kontakt mit der internen Authentifizierungs-Infrastruktur «Medusa» kam.

Darüber hinaus hat der Lernende eine Landing-Page für Start-Ups auf der Baloise.ch Webseite implementiert, über welche interessierte Start-Ups mit Mitarbeitern der Baloise in Kontakt treten können. Diese Aufgabe wurde mit Java im Back-End und JavaScript im Front-End realisiert.

# Analyse der Aufgabenstellung

Folgende Kapitel sollen die Aufgabenstellung aus pkorg.ch analysieren und wichtige Details für die Durchführungsphase erläutern.

## Kurzfassung Ist/Soll

Ist-Zustand

Die Einsätze der Mitarbeiter werden in einer Excel-Tabelle geplant, welche sich auf einem Netzlaufwerk befindet. Die Tabelle bildet die Einsätze aller Mitarbeiter für ein Semester ab, und kann nur mit einem Passwort bearbeitet werden.

Soll-Zustand

Für die Einsatzplanung steht eine Webanwendung bereit, welche von allen Mitarbeitern genutzt werden könnte. Die Berechtigung zur Bearbeitung der Einsätze ist abhängig von der Benutzergruppe des angemeldeten Benutzers und Tageszeiten sowie Schichten können von Teamleitern mit wenig Aufwand bearbeitet werden.

## Messbare Ziele

* Die Anwendung kann ohne weitere Anpassungen produktiv eingesetzt werden.
* Der Arbeitsfortschritt wird mit der Projektmethode «Kanban» dokumentiert.
* Änderungen in der Planung, die während der Realisierung auftreten, sind dokumentiert.
* Die Anwendung bietet die Auswahl zwischen 2 Sprachen an.
* Tageszeiten und Schichten lassen sich dynamisch anpassen.
* Teamleiter können die Gruppe eines Benutzers anpassen.
* Teamleiter können Einsätze planen, welche von allen Teammitgliedern eingesehen werden können.

## Dokumentation

Die Dokumentation der Entwicklung findet auf GitHub statt. Die anfallenden Arbeitsschritte werden als Issues notiert und mit Labels geordnet. Dadurch können sowohl Änderungen der Arbeit als auch Fortschritte festgehalten werden, wobei Kanban genutzt wird. Änderungen in der Planung und Probleme werden ebenfalls mit einem bestimmten Label notiert.

# Projektmanagement und Planung

## Organisation

Die Durchführung einer IPA fordert eine gute Planung und Organisation. Dazu gehört neben einem Zeitplan auch eine klare Aufteilung der Rollen, damit bei möglichen Rückfragen die richtige Ansprechperson kontaktiert werden kann.

### Zeitplan & Termine

Der Zeitplan soll in der Planung eine Einschätzung des anfallenden Arbeitsaufwandes abbilden. Dazu wird die Arbeit in die 2 Abschnitte «Entwicklung» und «Dokumentation» aufgeteilt, wobei das tägliche Ergänzen des Arbeitsjournals berücksichtigt werden muss. Da dies nur zwischen 10-20 Minuten dauern soll, wurde es im Zeitplan nicht explizit erwähnt. Nach Abschluss des Projekts soll ein Ist-Soll-Vergleich die Einhaltung des Zeitplans erkennbar machen.

Durchführungsblock

Startblock 3: 24.04.2019 – 10.05.2019

Durchführungs-Ort: Basler Versicherungen AG, Aeschengraben 21, 4002 Basel

Abgabe: 10.05.2019 um 18:00 Uhr

Termine

Vorbesprechung 05.04.2019 um 16:00 Uhr, @ Basler Versicherungen AG

Erster Besuch 25.04.2019 um 16:30 Uhr, @ Basler Versicherungen AG

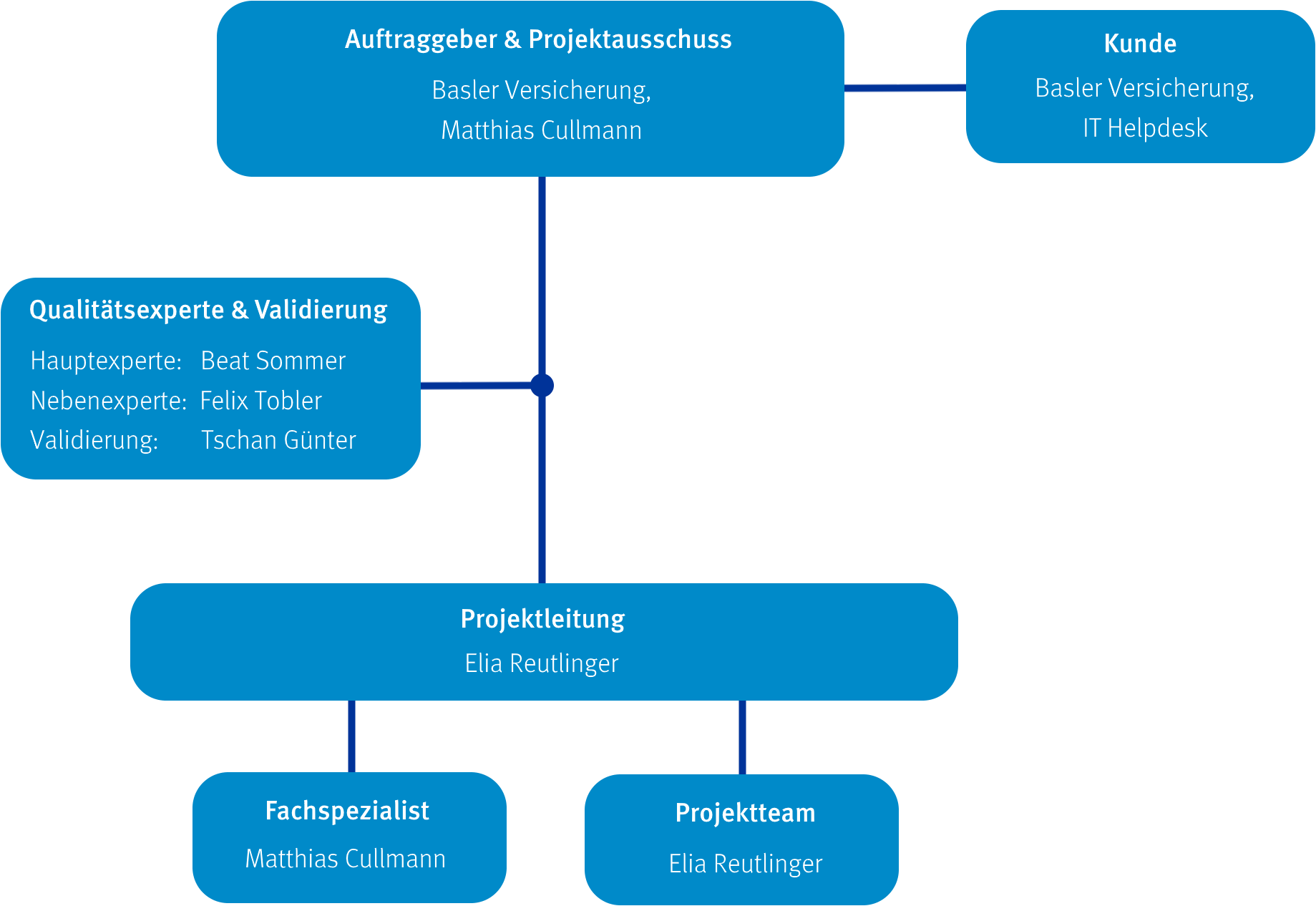
Zweiter Besuch 03.05.2019 um 11:00 Uhr, @ Basler Versicherungen AG

Präsentation & Fachgespräch 21.05.2019 um 15:00 Uhr, @ Basler Versicherungen AG

[**TODO** weitere Termine ergänzen, wenn nötig]

[**TODO**: Zeitplan einfügen (ganz am Ende)]

### Projektaufbauorganisation



Rollen

Auftraggeber Matthias Cullmann

Projektausschuss Matthias Cullmann

Kunde Corporate IT Helpdesk

Qualitätsexperten Beat Sommer, Felix Tobler

Validierungsexperte Tschan Günter

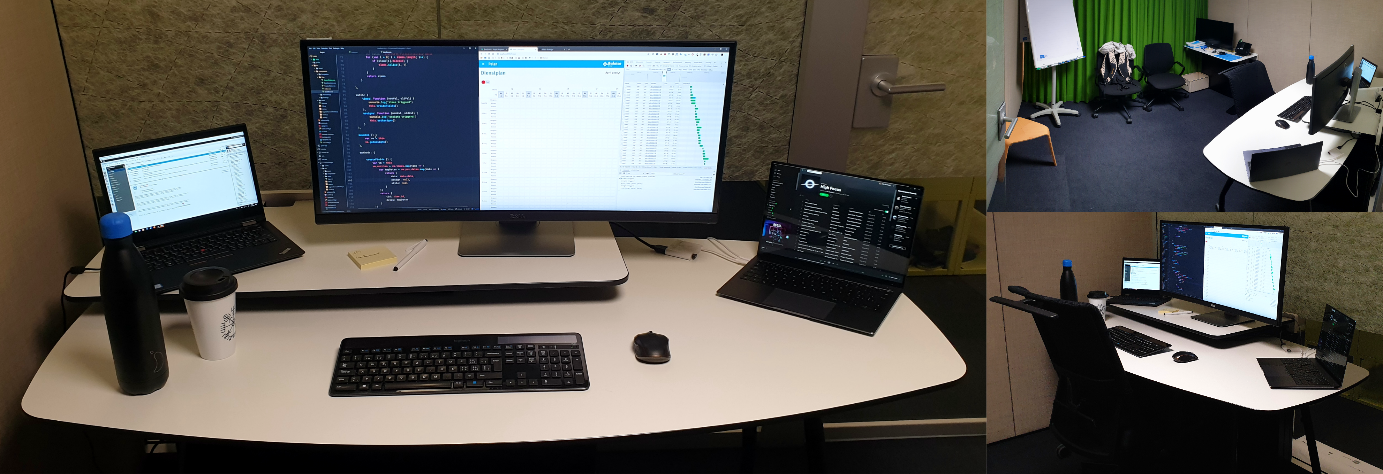
Projektleitung Elia Reutlinger

Fachspezialist Matthias Cullmann

Projektteam Elia Reutlinger

### Arbeitsplatz

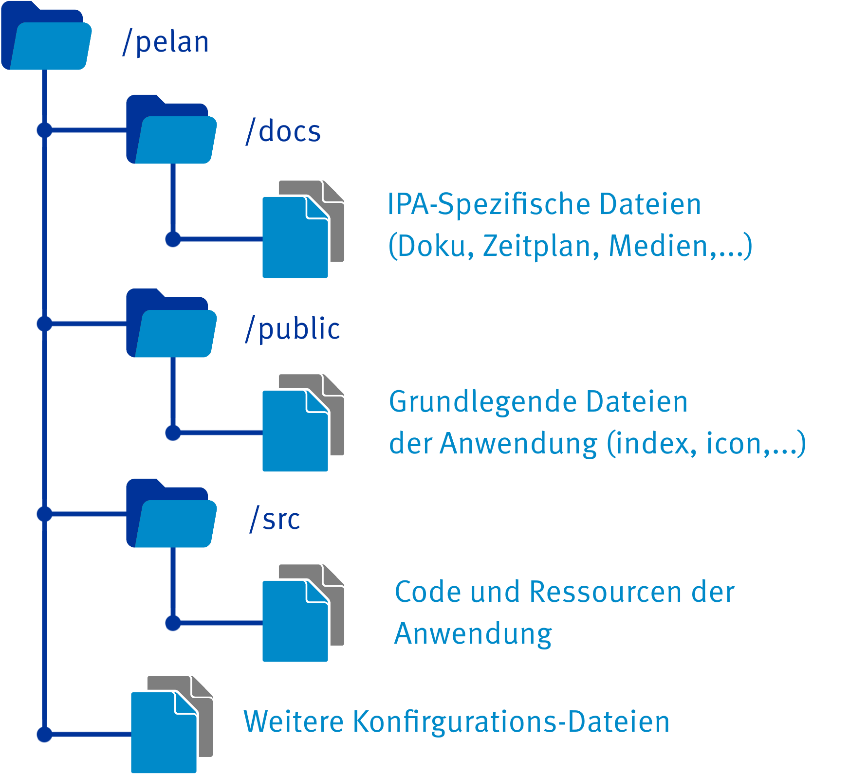
Die Ausführung der IPA findet im Hauptsitz der Basler Versicherungen statt. Im 4. Stock befinden sich neben dem Flex-Office einige Rückzugsräume, wobei einer von diesen für die IPA reserviert ist.



### Datensicherung & Versionierung

Die Datensicherung während der IPA findet vorrangig auf GitHub statt. Dabei wird die Anwendung mit dem IPA-Bericht in einem separaten Verzeichnis hochgeladen, was nach jeder grösseren Änderung mindestens 1 Mal pro Tag geschehen sollte. Mit jedem Hochladen können ein Titel sowie eine Beschreibung des Fortschritts notiert werden, wodurch erkennbar gemacht werden soll, was genau in der Version geändert wurde. Gleichzeitig kann man auf GitHub-Issues verweisen, wodurch im Kanban ersichtlich wird, in welcher Version welcher Arbeitsschritt erledigt wurde.

Als zusätzliche Absicherung bei Ausfällen erstelle ich auf einer externen Festplatte ein tägliches Backup der gesamten Partition meines Notebooks. Dies ist zwar laut Erkenntnissen aus der IPA-Vorbesprechung keine direkte Anforderung der IPA, ermöglicht es mir aber, im Notfall schneller zu meiner Arbeit zurückzukehren.



## Auswertung der Aufgabenstellung

In diesem Kapitel wird die Nutzung der detaillierten Aufgabenstellung als Quelle erläutert. Die Reflektion und Umwandlung der dort bestimmten Kriterien sollen hier erkennbar gemacht werden, da sich die Implementierung der Anwendung ausschliesslich an diesen orientierte.

### Wandlung der vorgegebenen Tests

Zur Prüfung der korrekten Funktion der Anwendung werden manuelle Tests durch ein Konzept definiert und mit einem Protokoll ausgeführt und festgehalten. Notwendige Test, welche die vorausgesetzten Funktionen der Anwendung bestätigen, wurden in der Aufgabenstellung auf pkorg.ch bereits grob beschrieben und müssen im Testkonzept enthalten sein. Dabei werden diese mit einzelnen Testschritten, erwarteten Resultaten und wenn nötig weiteren Details ergänzt. Die vordefinierten Tests bilden gleichzeitig die Use-Cases der Anwendung, nach welchen sich die Implementierung richtete.

### Mittel und Methoden

Zur Entwicklung der Anwendung und Erfassung des IPA-Berichts werden einige Tools benötigt, mit denen diese Aufwände optimal erledigt werden können. Da die Entwicklung losgelöst von internen Abhängigkeiten sein soll wird ein privates Notebook verwendet. Dies führt auch zu einer Absicherung vor möglichen Komplikationen mit der Infrastruktur der Baloise (beispielsweise Proxys, Firewalls und fehlende Berechtigungen).

Der Schwerpunkt während der Entwicklung lag bei unterschiedlichen Programmiersprachen und Frameworks, die ich in der Vergangenheit schon genutzt habe und dabei bereits einige Kenntnisse sammeln konnte.

* HTML & CSS

Zur Strukturierung und Gestaltung der Anwendung. Seit ungefähr 8 Jahren bei den meisten meiner Projekte in Verwendung.

* JavaScript

Für Funktionalitäten und die grundlegende Entwicklung der Anwendung. Bekannt seit Beginn meiner Ausbildung und bereits in zahlreichen Projekten verwendet.

* Vue.js

Webframework (JavaScript) zur Entwicklung clientseitiger Anwendungen. Erstmals im Dezember 2018 angewendet und bis jetzt 2 Projekte damit realisiert.

Bei der Verwendung von Vue.js entstehen in der Regel weitere Abhängigkeiten. Beispielsweise wird für das Design die Komponenten-Bibliothek Vuetify genutzt, durch die ein gängiges «Material Design» zeitnah realisiert werden kann.

Software & Dienste

Damit die Entwicklung lokal und mit Testdaten stattfinden kann, wird eine Instanz der Pelan-API benötigt. Da diese mit PHP und einer Datenbank arbeitet kann sie durch XAMPP betrieben werden. XAMPP ermöglicht unter anderem die Nutzung eines Apache Webservers auf einem lokalen Gerät, wobei zahlreiche Optionen zur Instanziierung einer API mitgeliefert werden.[[1]](#footnote-1)

Atom ist ein von GitHub entwickelter Texteditor, mit dem der Code von Pelan geschrieben wurde. Er bietet die Installation von zahlreichen Plugins an, durch die das Entwickeln vereinfacht werden soll. Weil die Anwendung auf GitHub dokumentiert und publiziert werden soll, wird GitHub Desktop verwendet. GitHub Desktop ist eine grafische Oberfläche zur Nutzung von Git, wodurch Änderungen im Code einfacher ersichtlich sind und schnell hochgeladen werden können. [[2]](#footnote-2) Durch die Integration eines Project-Boards auf GitHub werden die Issues des Pelan-Repositorys mit einem Kanban-Board visualisiert.

Zur Demonstration wird die Anwendung mit dem Online-Dienst Netlify unter einer Subdomain publiziert. Netlify bietet kostenloses Hosting von statischen Webseiten an, wobei ein Deployment-Script genutzt werden kann. Dadurch ist es möglich, einen Produktions-Build der Anwendung zu erstellen, was die endgültige Funktion garantieren soll. Alternativ könnte die Anwendung auch lokal im produktiven Modus demonstriert werden. [[3]](#footnote-3)

### Abhängigkeiten zu Vorarbeiten

Das Back-End zur Anwendung wurde bereits vor Beginn der IPA implementiert. Es besteht aus einer Rest-API, welche unter der Verwendung von JSON-Web-Tokens (JWT[[4]](#footnote-4)) Anfragen verarbeiten kann. Die Authentifizierung findet ausschliesslich im Back-End statt, da dies dem produktiven Einsatz innerhalb der Baloise entspricht. Die Implementierung des Front-Ends muss sich nach den bestehenden Endpunkten der API richten. Während der IPA dürfen keine gravierenden Anpassungen an der API vorgenommen werden, da dies nicht im Rahmen der Aufgabenstellung liegt.

Es wurde vorweg ein Repository unter der Organisation «Baloise» auf GitHub erstellt und mit grundlegenden Anforderungen/Aufgaben (Issues) in Berücksichtigung der detaillierten Aufgabenstellung ergänzt. Ebenfalls in der Aufgabenstellung festgelegt ist die Verwendung von Kanban als Projektmethode. Die Umsetzung findet in Form eines «Project-Boards» auf GitHub statt, wobei die Anforderungen als Issues visualisiert werden. Die Durchführung der IPA und Dokumentierung des Codes orientieren sich an diesem Kanban-Board. Des Weiteren muss mindestens ein tägliches Backup des Codes in dieses Repository geladen werden.

# Firmenstandards

Für den Wiedererkennungswert der Baloise in der Anwendung sollen möglichst viele Firmenstandards übernommen werden. Da die Funktion der Anwendung jedoch im Vordergrund stehen sollte, wird keine vollständige Umsetzung erwartet, was auch den Rahmen einer IPA sprengen würde. Da die Komponenten-Bibliothek «Vuetify» zur Umsetzung verwendet werden soll, sind hier bereits einige Abweichungen von den Firmenstandards eingeplant. Grundlegende Standards, die verwendet werden müssen, sind anschliessend in diesem Kapitel genauer dokumentiert. Sie stammen aus dem Webstyleguide[[5]](#footnote-5) und den Corporate Design Richtlinien[[6]](#footnote-6) der Baloise. Bei Unklarheiten zu Einzelfällen soll das weitere Vorgehen mit dem Fachvorgesetzten besprochen und entschieden werden.

## Schriftarten

Die Baloise nutzt grundsätzlich die Schriftarten der MetaPro-Reihe. Die textuellen Inhalte der Anwendung sollen deshalb ausschliesslich mit diesen angezeigt werden. Dabei stehen 2 Versionen zur Verfügung, welche je nach Zweck (Überschrift, Titel, Text, …) eingesetzt werden können.

## Farben

Der Webstyleguide der Baloise unterscheidet 3 Farbsorten:

Primärfarben

Diese bilden die beiden Blautöne der Baloise, #003399 und #008AC9, sowie ein dunkler Grauton #444444 als Textfarbe. Dieser hat jedoch lediglich den Zweck eine angenehme Lesbarkeit an Bildschirmen zu gewährleisten und könnte entsprechend mit einem gleichwertigen Farbton ersetzt/kombiniert werden.

Sekundärfarben

Für Benachrichtigungen sowie Erfolgs- und Fehlermeldungen wurden Sekundärfarben definiert. Sie bestehen aus einem Grün #2DB200, Rot #FF3366 und Orange #FF9900 (oder #FF8304). Der Einsatz dieser ist kontext-bezogen was bedeutet, dass sie nur für bestimmte Akzente verwendet werden sollten.

Flächenfarben

Hintergründe dienen der inhaltlichen Abgrenzung auf einer Webseite, weshalb die Baloise auch hier einige Farbabstufungen definiert hat. Diese Abstufungen gehen jeweils vom Blauton #99D0E9 und Grauton #7E7E7E aus. Eine genaue Vorgabe zum Einsatz dieser Farben besteht nicht, wobei die Grautöne möglichst vermieden werden sollen.

## Logos

Damit Benutzer der Anwendung die Zuordnung zur Baloise direkt erkennen können, soll das Baloise-Group-Logo über alle Seiten der Anwendung präsent sein. Da voraussichtlich keine Funktionen für Soziale-Medien oder andere Kanäle bestehen sollten, ist dies auch das einzige geplante Logo.



## Formularelemente [[7]](#footnote-7)

Für eine tiefgreifende Umsetzung des Baloise-Webstyleguide müssten sämtliche Formularelemente eigenständig angepasst werden. Da dies jedoch über den Zeitrahmen der IPA hinausgehen könnte und keinen weiteren Einfluss auf die Funktion der Anwendung hat, sollte zur Veranschaulichung mindestens 1 Element übernommen werden. Folgende Elemente könnten dazu verwendet werden, wobei der Webstyleguide der Baloise auch weitere dokumentiert hat.

Textfelder

Ein Textfeld sollte durch eine blaue Umrandung (#008AC9) dargestellt werden. Als Platzhalter kann eine Info zur Zeichenbegrenzung eingefügt werden. Sowohl wenn der Benutzer das Feld anwählt, als auch während einer Eingabe, soll der Hintergrund des Textfeldes in einen abgewandelten Blauton übergehen.



Buttons

Navigations-Knöpfe wie ein «Absenden» Knopf sollten in ihrer Grundform als rechteckige Fläche mit orangem Hintergrund und weissem Text dargestellt werden. Bei Berührung mit dem Cursor des Benutzers sowie beim Anklicken sollte die Hintergrundfarbe in einen Blauton übergehen.



# Arbeitsjournal

## Tag 1 – 24.04.2019

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zeit | Tätigkeit | Beteiligte | Aufwand «Soll» | Aufwand «Ist» |
| 07:00 | Dokument erstellt/formatiert, Vue-Projekt auf GitHub initialisiert. | Elia Reutlinger | 02:00:00 | 01:30:00 |
| 08:30 | Konfigurationen vorgenommen (Design, i18n, PWA, Vue, Routing). | Elia Reutlinger | 04:00:00 | 02:30:00 |
| 12:00 | Authentifizierung implementiert + kleinere Optimierungen. | Elia Reutlinger | 02:00:00 | 03:30:00 |
|  | | | 08:00:00 | 07:30:00 |

Tagesthemen

Mit Beginn der IPA stand heute die grundlegende Konfiguration des Vue-Projekts sowie die Implementierung der Authentifizierung mit der API auf dem Plan. Nachdem ich das neue Projekt initialisiert und die Doku erstellt habe, nahm ich einige Einstellungen an Abhängigkeiten vor, damit ich weiterhin problemlos arbeiten kann. Die meisten Einstellungen konnte ich aus meinen alten Projekten übernehmen, wobei ich schneller fertig war als erwartet. Die Authentifizierung dauerte dafür länger, da ich mich etwas länger mit dem Routing nach Berechtigungen auseinandersetzen musste.

Hilfestellungen

Stackoverflow.com, Github.com, vuejs.org, alte Projekte (github.com/erleiuat/minska)

Reflexion +

Ich konnte alle Ziele des Tages erreichen und war sogar etwas schneller fertig als geplant. Deshalb konnte ich noch einige Verbesserungen am Design und der Performance der App vornehmen. Insgesamt bin ich mit der heutigen Arbeit sehr zufrieden, da ich mir durch diese Grundlagen nun ein besseres Bild vom Endprodukt machen kann.

Reflexion –

Ich habe mich im Zeitplan mit den heutigen Aufwänden ein wenig verschätzt, da ich dabei zu sehr von meinen bisherigen Projekten ausgegangen bin. Ich denke aber, dass sich das auf heute beschränken wird.

Erkenntnisse

Ich habe eine bessere Umsetzung fürs Routing nach Berechtigungen gefunden als in meinen bisherigen Projekten. Wahrscheinlich werde ich das irgendwann auch da verbessern.



## Tag 2 – 25.04.2019

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zeit | Tätigkeit | Beteiligte | Aufwand «Soll» | Aufwand «Ist» |
| 07:30 | PWA Eingerichtet | Elia Reutlinger | 02:00:00 | 00:30:00 |
| 08:00 | «Plan-Einstellungen» implementiert | Elia Reutlinger | 04:00:00 | 03:30:00 |
| 12:00 | «Einstellungen» implementiert | Elia Reutlinger | 02:00:00 | 02:30:00 |
| 14:30 | Mehrsprachigkeit angepasst | Elia Reutlinger | 00:00:00 | 01:30:00 |
|  | | | 08:00:00 | 08:00:00 |

Tagesthemen

Heute mussten noch die letzten Einstellungen zum PWA (Progressive-Web-App) gemacht werden, wobei ich mit diesen bereits gestern in der «Konfigurations-Phase» angefangen habe. Deshalb benötigte ich hier weniger Zeit, wobei ich meine Schätzung von 2 Stunden auch etwas übertrieben finde. Anschliessend habe ich mit den «Plan-Einstellungen» die erste Seite implementiert, wobei keine erwähnenswerten Probleme entstanden sind. Bei der «Einstellungen»-Seite gab es jedoch am Ende einen Fehler, da die Sprachauswahl zwar gespeichert, aber nicht von der Oberfläche übernommen wurde. Deshalb musste ich dort nochmals nachhaken, was etwas Zeit beansprucht hat.

Hilfestellungen

Stackoverflow.com, kazupon.github.io/vue-i18n

Reflexion +

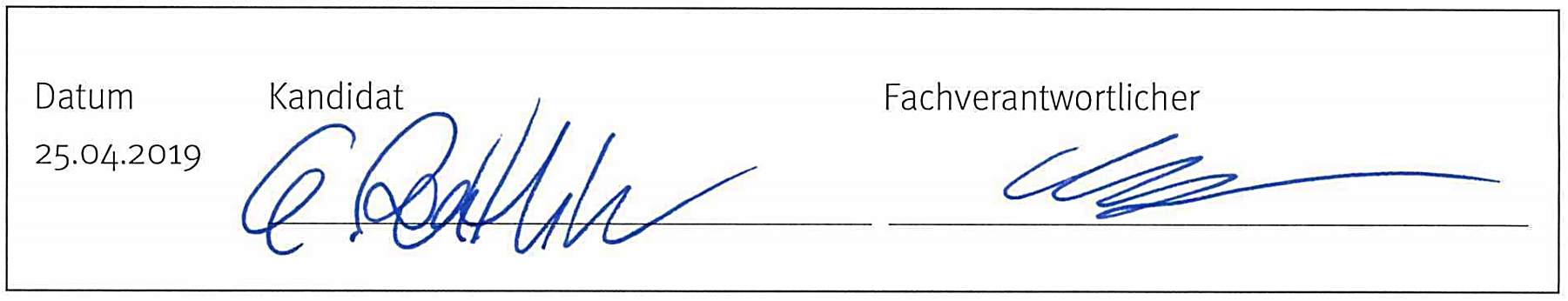
Ich hatte sozusagen Glück, dass ich trotz dem einen Fehler im Zeitplan blieb. Ich konnte ansonsten alles wie geplant implementieren und folgte den Schätzungen zum Zeitaufwand fast genau. Die beiden Seiten finde ich vom Design her ziemlich gelungen, da sie doch noch sehr übersichtlich und «modern/material» wirken, auch wenn viele Daten angezeigt werden. Ausserdem konnte ich das Formular zum Bearbeiten/Erstellen von Schichten nach Baloise-Standards kreieren, wodurch ich dieses Kriterium auch schon erfüllen konnte.

Reflexion –

Wegen dem Fehler habe ich im Internet länger nach einer Lösung gesucht aber keine gefunden. Ich habe die Abhängigkeit entfernt und neu installiert, was auch nicht erfolgreich war. Deshalb habe ich selbst über die Ursache nachgedacht und entdeckt, dass das Problem an der Verteilung der Übersetzungen lag (die Sprache lässt sich nicht in einer Komponente ändern, welche Übersetzungen definiert). Davon steht aber absolut nichts in der Doku des Plugins, was ich sehr schlecht finde.

Erkenntnisse

Obwohl ich das Problem mit der Anpassung der Sprache so noch nie hatte, werde ich es in zukünftigen Projekten direkt berücksichtigen können und damit einiges an Zeit sparen.



## Tag 3 – 26.04.2019

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zeit | Tätigkeit | Beteiligte | Aufwand «Soll» | Aufwand «Ist» |
| 07:00 | Doku-Kapitel erweitert/vertieft. | Elia Reutlinger | 00:00 | 01:00 |
| 08:00 | Implementierung «Plan» begonnen | Elia Reutlinger | 04:00 | 03:00 |
| 12:00 | Implementierung «Plan» | Elia Reutlinger | 04:00 | 04:00 |
|  | | | 08:00 | 08:00 |

Tagesthemen

Nach dem 1. Expertenbesuch gestern Nachmittag bin ich dem Rat von Herr Sommer gefolgt und habe deshalb heute als erstes die Kapitel der Doku erweitert. Dabei versuchte ich die Kriterien und alle Punkte, welche die Entwicklung betreffen, direkt zu beachten. Anschliessend konnte ich mit dem Plan beginnen, welcher sozusagen den wichtigsten Teil der ganzen Anwendung darstellt. Ich benötigte einige Versuche, bis ich eine funktionsfähige Struktur der Komponenten fand, bei welcher die Daten «reaktiv» bleiben. Dabei musste ich auch auf das Design achten, da nun unterschiedliche Tabellen zum Einsatz kommen, welche sich aber Design-Eigenschaften teilen müssen. Obwohl ich mit dem Design schon soweit fertig bin, konnte ich noch keine optimale Lösung für die Reaktivität finden, vor allem auf der Ebene der einzelnen Einsätze (unterste Ebene/Komponente).

Hilfestellungen

Stackoverflow.org, vuejs.org / vuex.vuejs.org, alligator.io

Reflexion +

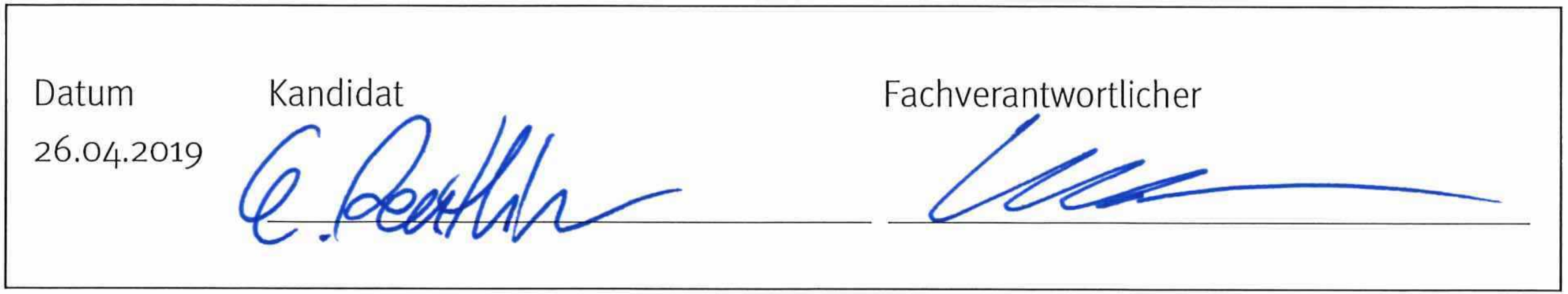
Durch die Vertiefung der Kapitel weiss ich besser, was in der Dokumentations-Phase auf mich zu kommt und kann bereits jetzt einiges notieren, was ich später noch brauche bzw. berücksichtigen sollte.

Reflexion –

Die Reaktivität von Daten zwischen Komponenten ist eine etwas komplexere Thematik als ich es mir vorgestellt habe. Ich musste heute schon ziemlich grübeln und viel recherchieren, um einen Fortschritt zu erzielen, wobei ich aber noch keine endgültige Lösung finden konnte.

Erkenntnisse

Vuex (ein globaler Speicher für Vue.js) löst nicht in jedem Fall eine Aktualisierung in einer Komponente aus. Da der Plan eine sehr hohe Dynamik der Daten fordert (variable Anzahl Benutzer und geplante Einsätze) musste ich (und muss ich noch) mich dazu genauer informieren.



## Tag 4 – 30.04.2019

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zeit | Tätigkeit | Beteiligte | Aufwand «Soll» | Aufwand «Ist» |
| 07:00 | Implementierung «Plan» (Vormittag) | Elia Reutlinger | 04:00 | 04:00 |
| 12:00 | Implementierung «Plan» (Nachmittag) | Elia Reutlinger | 00:00 | 03:00 |
| 15:00 | Implementierung «Dashboard» | Elia Reutlinger | 02:00 | 01:00 |
|  | | | *08:00* | 08:00 |

Tagesthemen

Heute sollte laut Zeitplan die Plan-Seite fertig implementiert werden. Dazu musste ich mich zuerst genauer über die Reaktivität von Komponenten informieren, weil ich letzte Woche dort stehen geblieben bin. Ich testete unterschiedliche Möglichkeiten um die Einsätze dynamisch im globalen Speicher abzulegen, was mich einiges an Zeit kostete. Als ich die Lösung fand, hatte ich bereits länger benötigt als im Zeitplan angegeben. Ich musste dann aber noch einige andere Komponenten an die neue Lösung anpassen, was noch mehr Zeit benötigte. Schlussendlich habe ich zwar den Plan fertiggestellt, aber die anderen Tagesziele (Dashboard- & Support-Seite) nicht ganz erreicht. Ich denke aber, dass ich trotzdem pünktlich mit der Entwicklungs-Phase fertig sein werde, da ich für die letzten Schritte wahrscheinlich nicht so viel Zeit benötige wie geplant.

Hilfestellungen

Stackoverflow.org, vuejs.org/v2/guide/reactivity, vuejs.org/v2/guide/computed, vuex.vuejs.org

Reflexion +

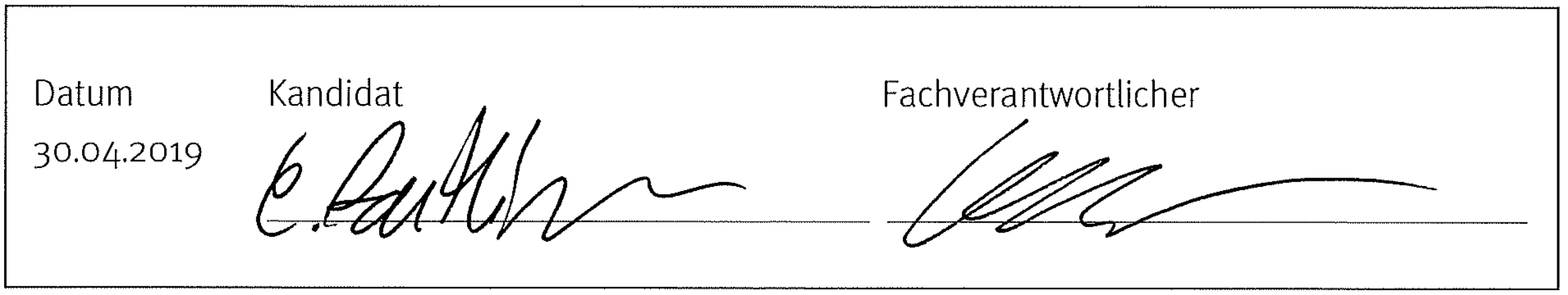
Die schlussendliche Umsetzung des Plans finde ich äusserst gut. Nun kann man die Daten, die der Plan nutzt, überall dynamisch verändern und der Plan aktualisiert sich zuverlässig. Auch wenn es länger gedauert hat denke ich, dass diese Zeit so richtig investiert wurde.

Reflexion –

Mit der neuen Lösung musste ich einige andere Komponenten anpassen, die ich bereits zuvor beendet hatte. Ohne diese Anpassungen hätten die Komponenten aber gewisse Daten nicht richtig verarbeitet, sodass keine Reaktivität entstanden wäre. Die Anpassungen waren also notwendig.

Erkenntnisse

Die Lösung war die Funktion «Object.assign()» mit einer Reinitialisierung des betroffenen Objektes, wodurch sich Daten bzw. Objekte auch nach der Initialisierung der Vue-Instanz als dynamische Objekte zum globalen Speicher hinzufügen lassen. In Verbindung mit einigen «Watchers» sind die einzelnen Benutzer-Zeilen bzw. Einsätze nun auch reaktiv/dynamisch.



## Tag 5 – 02.05.2019

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zeit | Tätigkeit | Beteiligt | H-«Soll» | H-«Ist» |
| 07:30 | Dashboard fertig implementiert | Elia R. | 00:00 | 01:00 |
| 08:30 | Support-Seite implementiert | Elia R. | 00:00 | 01:00 |
| 09:30 | Fehlermeldungen implementiert / verbessert | Elia R. | 02:00 | 01:30 |
| 11:00 | Zahlreiche Verbesserungen | Elia R. | 02:00 | 01:00 |
| 12:30 | Weitere, kleinere Verbesserungen (App fertig) | Elia R. | 00:00 | 00:30 |
| 13:00 | Doku: Aufgabenstellung & Analyse fertig | Elia R. | 02:00 | 01:30 |
| 14:30 | Doku: Planung & Standards begonnen | Elia R. | 02:00 | 01:30 |

Tagesthemen

Heute habe ich das Dashboard und die Support-Seite fertiggestellt. Die Fehlermeldungen sind ebenfalls fertig und wurden noch weiter verbessert. Anschliessend habe ich noch sehr viele Punkte verbessert, wobei die Responsiveness, Performance und Firmenstandards im Fokus lagen. Bei einem kurzen Test erschien noch ein Bug bei der Verwaltung der Tageszeiten, welcher aber ohne viel Aufwand behoben werden konnte. Zum Schluss habe ich den Code nochmals verschönert/formatiert und weitgehend kommentiert. Damit war die Implementierungsphase auch abgeschlossen und ich begann mit den ersten Kapiteln der Doku. Dabei bemerkte ich, dass die Einordnung im Zeitplan nicht viel Sinn ergibt, weshalb ich den Zeitplan leicht angepasst habe. Ich konnte 2 ganze Kapitel beenden und auch schon die Firmenstandards und das Projektmanagement beginnen.

Hilfestellungen

webstyleguide.baloise.com, vuetifyjs.com

Reflexion +

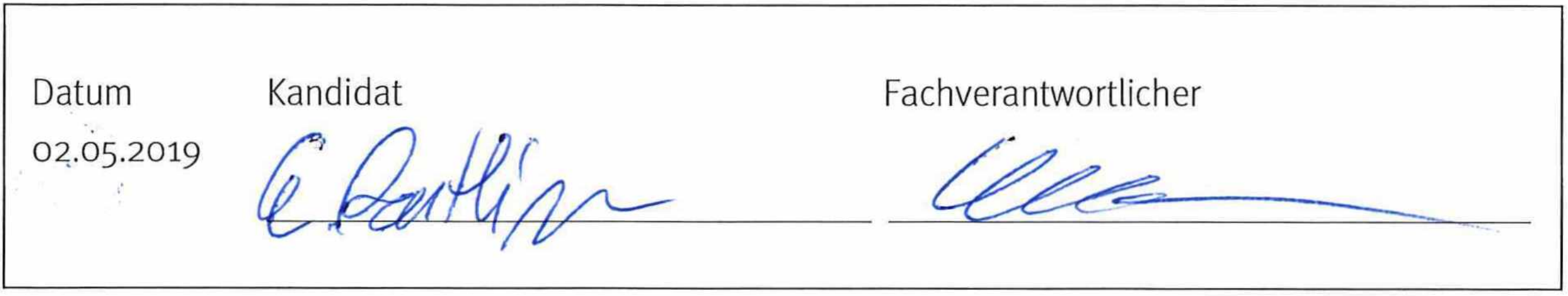
Ich finde es sehr gut, dass ich fast genau nach Plan mit der Implementierung fertig wurde. Obwohl es einiges nachzuholen gab, schaffte ich es, alles zu erledigen und sogar noch viel zu verbessern. Ich konnte auch noch meinen Code «verschönern», indem ich alles formatiert und kommentiert habe. Dabei würde ich sogar sagen, dass ich noch nie einen saubereren Code geschrieben habe.

Reflexion –

Die Bearbeitung der Tageszeiten funktionierte nicht mehr richtig und musste verbessert werden. Des Weiteren schienen die Schriftarten nicht ganz korrekt zu sein, worauf ich die Offiziellen von Intern herunterladen musste. Dabei habe ich auch gleich weitere Standards verbessern müssen.

Erkenntnisse

Ich habe herausgefunden, wie man Schriftarten dynamisch nachladen kann (durch eine Einstellung vom Plugin «Webpack»). Dadurch wurde die Performance der App sehr viel verbessert.



## Tag 6 – 03.05.2019

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zeit | Tätigkeit | Beteiligt | H-«Soll» | H-«Ist» |
| 07:30 | Doku: Projektmanagement & Planung fertiggestellt | Elia R. | 02:00 | 01:30 |
| 09:00 | Doku: Auswertung der Aufgabenstellung | Elia R. | 02:00 | 01:00 |
| 10:00 | Doku: Verbesserungen & Notizen | Elia R. | 00:00 | 01:00 |
| 12:00 | Doku: Kapitel «Initialisierung» begonnen | Elia R. | 02:00 | 4:30 |

Tagesthemen

Heute konnte ich die Kapitel «Projektmanagement und Planung» und die Auswertung fertigstellen. Da dabei keine uneingeplanten Arbeiten entstanden investierte ich noch etwas Zeit in zahlreiche Verbesserungen (bzgl. Detailgrad & Grammatik). Dabei hatten auch die Erkenntnisse Einfluss, welche ich durch die Fragen des heutigen Expertenbesuchs erhalten habe. Am Kapitel «Initialisierung» konnte ich sogar länger arbeiten, da die vorherigen Kapitel nicht so lange dauerten.

Hilfestellungen

IPA-Experten, IPA-Kriterienkatalog, webstyleguide.baloise.com, Wikipedia.org, Baloise.ch

Reflexion +

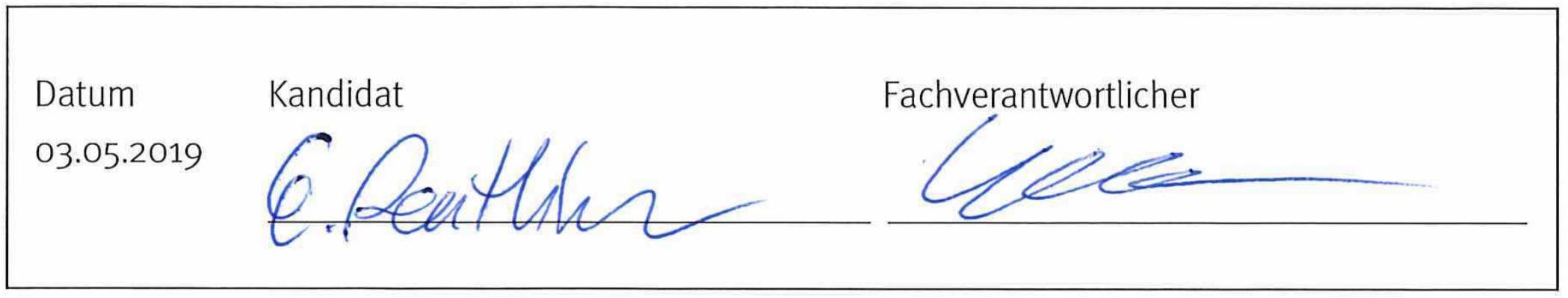
Ich bin nach wie vor äusserst froh darüber, dass die Implementierung abgeschlossen und ohne grössere Probleme verlaufen ist. Der Expertenbesuch hat ebenfalls einige Fragen geklärt, wodurch ich nun ein ziemlich klares Ziel vor Augen habe. Des Weiteren finde ich es gut, dass ich heute bereits mehr erledigen konnte, als im Zeitplan geplant.

Reflexion –

Das Standardkriterium Nr. 1 aus Teil A sagt, dass die korrekte Anwendung der Projektmanagement-Methode (in unserem Fall das Kanban-Board) im IPA-Bericht ersichtlich sein muss. Ich kann mir jedoch noch nicht konkret vorstellen, wie ich das realisieren sollte, da unser Kanban ausschliesslich auf GitHub.com geführt wird. Möglicherweise werde ich eine Art von Diagramm einsetzen, wobei ich mir hier noch nicht ganz sicher bin.

Erkenntnisse

Nach dem Expertenbesuch weiss ich nun um einiges besser, was ich wie in der Dokumentation beschreiben sollte.



## Tag 7 – 07.05.2019

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zeit | Tätigkeit | Beteiligt | H-«Soll» | H-«Ist» |
| 07:00 | Kleinigkeiten an der App verbessert. | Elia R. | 00:00 | 00:30 |
| 07:30 | Doku: Kapitel «Initialisierung» fertiggestellt. | Elia R. | 06:00 | 03:30 |
| 12:00 | Doku: Testkonzept verfeinert | Elia R. | 00:00 | 00:30 |
| 12:30 | Doku: Kapitel «Realisierung» begonnen | Elia R. | 02:00 | 03:30 |

Tagesthemen

Laut Zeitplan soll heute das Kapitel «Initialisierung» abgeschlossen sein, worauf mit dem Kapitel «Realisierung» begonnen werden kann. So ist es auch eingetroffen, wobei ich am Morgen noch kleinere Anpassungen an der App vorgenommen habe, die mir beim Betrachten des Codes aufgefallen sind (Design-Definitionen ausgelagert für dynamische Einbindung). Das Kapitel «Initialisierung» konnte ich bereits vor dem Mittagessen abschliessen. Nach einer Abgleichung mit den Kriterien bestand aber noch Verbesserungspotential beim Testkonzept, was ich auch gleich angepasst habe. Anschliessend konnte ich (früher als geplant) mit dem Kapitel «Realisierung» beginnen.

Hilfestellungen

Github.com/webpack-contrib (Style-Loader), IPA-Kriterien,

Informatik-aktuell.de (Testkonzept-Normen nach «IEEE 829»), Owasp.org (Top 10)

Reflexion +

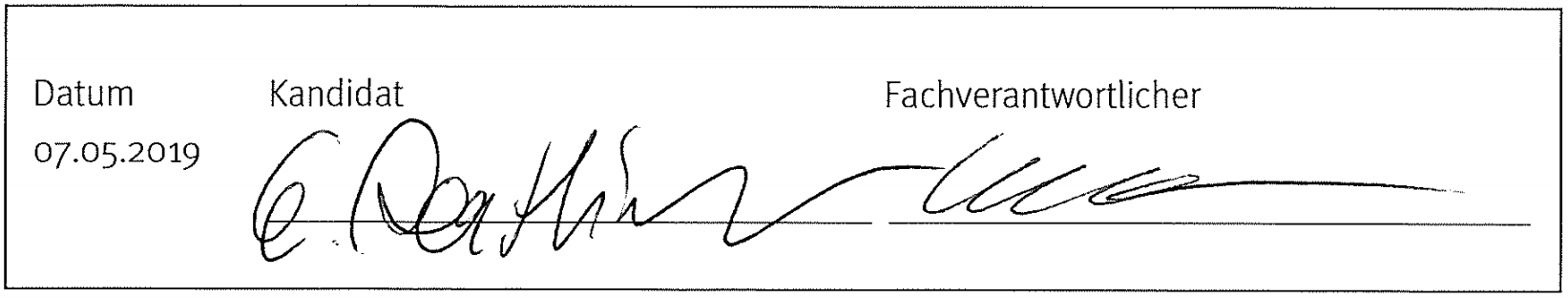
Ich konnte früher mit dem Kapitel «Initialisierung» fertig werden und finde es an sich sehr gelungen. Ich denke es hat einen klaren «Roten-Faden» und kann Lesern einen guten Überblick zum Rahmen der IPA verschaffen. Das Testkonzept sollte jetzt auch alle Kriterien nach Katalog erfüllen.

Reflexion –

Das Kapitel «Realisierung» wird schätzungsweise das Grösste, wobei die Doku schon jetzt sehr viele Wörter enthält. Ich muss zum Schluss nochmals prüfen, ob ich auch wirklich keine Wiederholungen im Text habe und möglichen Ballast vermeide, während genügend Informationen für ein gutes Verständnis beim Leser vorhanden sind.

Erkenntnisse

Um das Sicherheitskonzept zu erfassen habe ich mich erneut mit den «OWASP Top 10» Sicherheitsrisiken auseinandergesetzt. Dabei sind mir weitere Risiken im Front-End aufgefallen, die ich in der Planung bzw. im GitHub-Issue #18 nicht notiert habe. Bei dieser Vertiefung konnte ich noch einige neue Details zu den ganzen Risiken auffassen.



## Tag 8 – 08.05.2019

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zeit | Tätigkeit | Beteiligt | H-«Soll» | H-«Ist» |
| 07:20 | «Farb-Wähler» zur Schicht-Bearbeitung gewechselt. | Elia R. | 00:00 | 00:10 |
| 07:30 | An Kapitel «Realisierung» gearbeitet. | Elia R. | 04:00 | 04:00 |
| 12:00 |  | Elia R. | 00:00 | 00:00 |
| 00:00 |  | Elia R. | 00:00 | 00:00 |

Tagesthemen

…

Hilfestellungen

…

Reflexion +

…

Reflexion –

…

Erkenntnisse

…

|  |
| --- |
| Datum Kandidat Fachverantwortlicher  08.05.2019  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

## Tag 9 – 09.05.2019

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zeit | Tätigkeit | Beteiligt | H-«Soll» | H-«Ist» |
| 00:00 |  | Elia R. | 00:00 | 00:00 |
| 00:00 |  | Elia R. | 00:00 | 00:00 |
| 00:00 |  | Elia R. | 00:00 | 00:00 |

Tagesthemen

…

Hilfestellungen

…

Reflexion +

…

Reflexion –

…

Erkenntnisse

…

|  |
| --- |
| Datum Kandidat Fachverantwortlicher  09.05.2019  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

## Tag 10 – 10.05.2019

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zeit | Tätigkeit | Beteiligt | H-«Soll» | H-«Ist» |
| 00:00 |  | Elia R. | 00:00 | 00:00 |
| 00:00 |  | Elia R. | 00:00 | 00:00 |
| 00:00 |  | Elia R. | 00:00 | 00:00 |

Tagesthemen

…

Hilfestellungen

…

Reflexion +

…

Reflexion –

…

Erkenntnisse

…

|  |
| --- |
| Datum Kandidat Fachverantwortlicher  10.05.2019  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

# Abschlussbericht

## Ist-Soll

## Schwierigkeiten

## Fazit zum Projekt

### Entstandene Abweichungen zur Aufgabenstellung

## Persönliches Fazit

## Schlussreflexion

[**TODO**: Unterschiedlichste Sachen reflektieren **UND** vergleichen. Z.b: …]

### Arbeitsmethodik

### Werkzeuge

Teil 2: Dokumentation und Umsetzung

# Initialisierung

## Analyse Vue.js

Vue.js ist ein JavaScript-Framework zur Realisierung von Single-Page-Webanwendungen. Durch die Nutzung eines Mapping-Routers kann eine Webanwendung jedoch in mehrere Seiten unterteilt werden. Hauptmerkmale von Vue.js sind unter anderem:[[8]](#footnote-8)

* Komponentenbasierte Umsetzung (Wiederverwendbarkeit)
* Viel Flexibilität
* Hohe Performance
* Grosse Community (= gute Dokumentation und reichlich Unterstützung)

Bei korrekter Umsetzung des Konzepts kann mit Vue.js eine professionelle Webanwendung implementiert werden, die mit eigenen Scripts und Komponenten beliebig erweitert werden kann. Deshalb sollte sich Vue.js zur Realisierung einer Einsatzplanung eignen, während für bestimmte Anforderungen noch zusätzliche Abhängigkeiten eingebunden wurden. Zu den wichtigsten gehören dabei folgende:

* Durch «Vuetify» lässt sich ein grundlegendes Design schnell umsetzen, wobei zahlreiche Variablen individuell angepasst werden können. Die mitgelieferten Komponenten entsprechen den Material-Design Standards und lassen sich dynamisch einbinden.
* Für eine mehrsprachige Anwendung kann «Vue-i18n» genutzt werden. Dies ist das gängigste Modul, um Übersetzungen innerhalb von Komponenten anwendbar zu machen.
* Mit «Axios» können Anfragen an einen Server bzw. eine API asynchron durchgeführt werden. Dabei lassen sich die Parameter einer Anfrage beliebig anpassen.

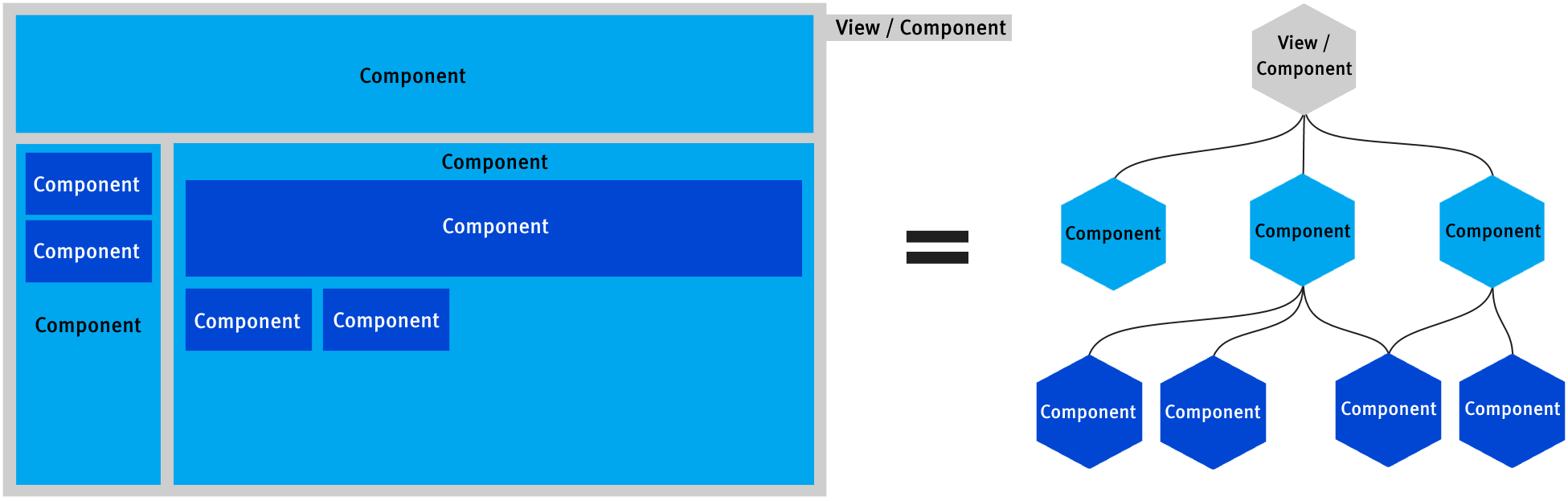
Durch weitere Entwicklungs-Abhängigkeiten wurde die Implementierung einfacher gemacht, wodurch ich mich stärker auf die wichtigen Punkte konzentrieren konnte. Schlussendlich war es also möglich, durch Vue.js eine Anwendung performant, stabil und schnell umzusetzen.

### Konzept

Vue.js baut auf einem gängigen Konzept der Realisierung von Webanwendungen auf. Dabei kommen Komponenten zum Einsatz, welche aus einer Darstellungs-Struktur (HTML), mehreren Funktionen (JS) und Design-Definitionen (CSS) bestehen. Dadurch sollen die Elemente einer Anwendung modularisiert werden. Komponenten können dann beliebig als View oder in anderen Komponenten verwendet und dynamisch eingebunden werden.

Views stellen die unterschiedlichen Ansichten dar, wobei hier in der Regel nicht die Hauptfunktionen eingebaut werden. Es wird definiert, welche Komponenten in der View dargestellt werden, und wie diese miteinander in Verbindung stehen.

Der Sinn hinter den Komponenten ist die Wiederverwendbarkeit von Elementen, wodurch diese an unterschiedlichen Stellen genutzt werden können und bei Anpassungen nur eine Stelle im Code bearbeitet werden muss.



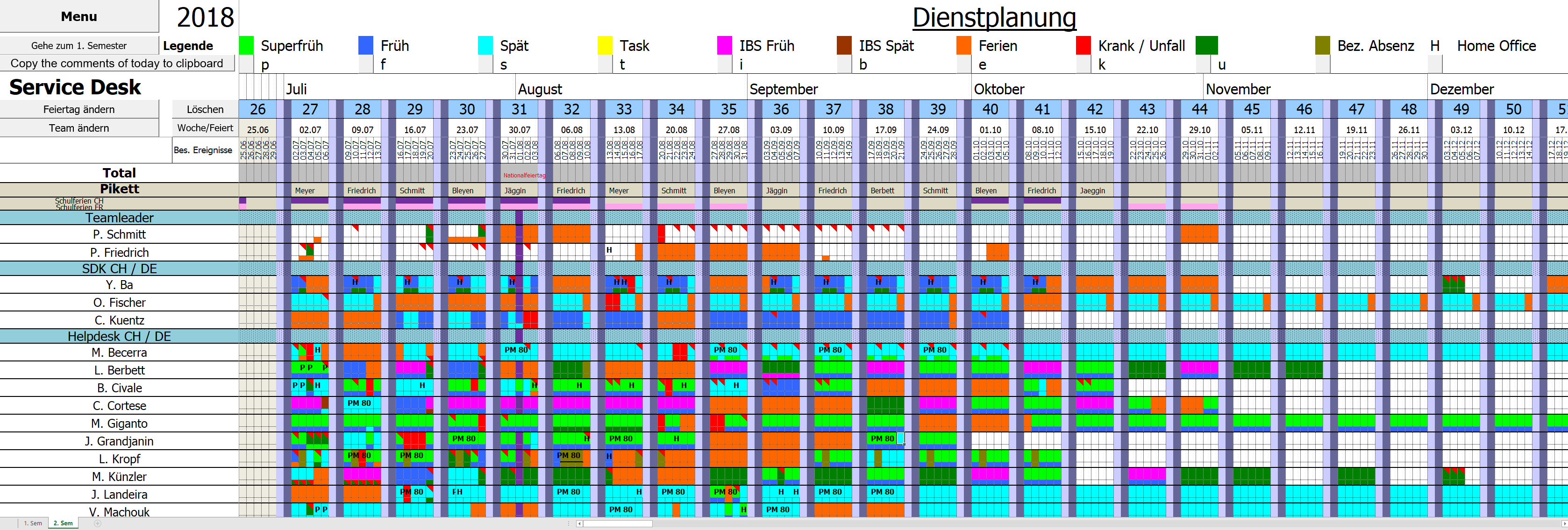
Mit Vuex entsteht zudem eine Art globaler Speicher (der sogenannte «Store»), auf welchen alle Komponenten zugreifen können. Dabei ist es möglich den Inhalt einer Komponente mit diesem Speicher zu verknüpfen, sodass bei Veränderungen an einer anderen Stelle der Inhalt in der verknüpften Komponente automatisch aktualisiert wird.

### Vergleich

Zur Auswahl standen auch weitere Frameworks wie React oder Angular, mit welchen ich bisher keine oder nur wenig Erfahrung habe. Nach einer kurzen Evaluation der Frameworks stellte sich für mich heraus, dass React und Angular sehr viele zusätzliche Abhängigkeiten liefern, die in dieser Anwendung nicht notwendig sind. Da diese die Performance der Anwendung beeinflussen können, und einen Zeitaufwand für die Konfiguration benötigen, habe ich mich gegen die Verwendung dieser Frameworks entschieden.

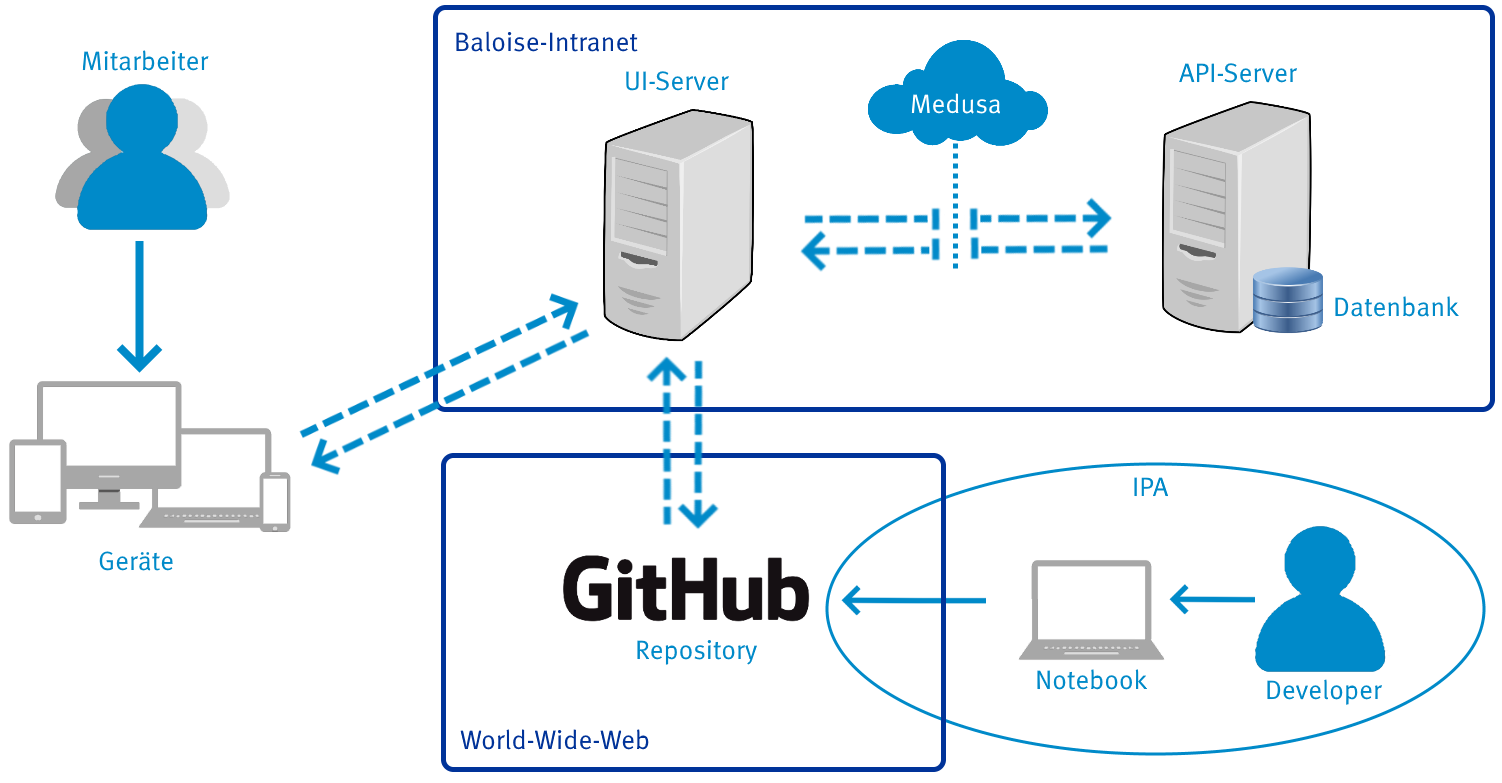
## Vertiefte Studie Ist-Zustand & Umgebung

Die Einsatzplanung am Helpdesk wird mit einer komplexen Excel-Tabelle geführt. Darin wird ein gesamtes Semester abgebildet sowie alle Mitarbeiter des Teams, was die Übersicht sehr einschränkt. Abhilfe würde hier beispielsweise die Darstellung von nur einem Monat bieten.



Die Berechtigungen werden durch einen Passwort-Schutz umgesetzt, wodurch nur bestimmte Personen Änderungen an der Tabelle vornehmen können. Andere können das Dokument lediglich einsehen aber nicht bearbeiten. Damit alle darauf zugreifen können, wurde es auf einem Netzlaufwerk gespeichert. Dabei wird es bei einem Zugriff für andere Mitarbeiter gesperrt, was die Dynamik einschränkt. Gleichzeitig besteht hier ein Mehraufwand durch das Suchen der Datei und die Eingabe des Passwortes, was durch eine Webanwendung nicht mehr notwendig wäre.

Um Änderungen am Plan vorzunehmen, wurden Makros eingebaut. Die Nutzung von Makros birgt jedoch ein Sicherheitsrisiko und kann zu Verständnisproblemen führen. Klar definierte Funktionen zur Bearbeitung mit einer benutzerfreundlichen Darstellung würden dies optimieren. Eine Webanwendung kann bei korrekter Implementierung auch die Sicherheitsrisiken eliminieren.



Die Umgebung, in welche die Anwendung nach der IPA eingebettet werden soll, besteht aus der internen Infrastruktur der Baloise. Dabei ist für die Anwendung selbst ein Server eingeplant, welcher den Code kompiliert und intern veröffentlicht. Der Code wird dazu aus dem GitHub-Repository geladen, auf welchem die IPA stattfindet. Als wichtigste und aktuell einzige Schnittstelle gilt die Pelan-API, welche sich ebenfalls auf einem internen Server befindet und Zugriffe über Medusa erlaubt.

Die Pelan-API ist als Back-End zuständig für alles, was mit der Verarbeitung und Verwaltung von Daten zu tun hat. Zusätzlich muss die API bei einer Anfrage prüfen, ob die notwendigen Berechtigungen zum Abrufen gewisser Daten vorhanden sind.

Jede Anfrage an das Back-End geht durch die interne Authentisierungsplattform «Medusa», was das Anmelden eines Benutzers ohne Angabe eines Passworts ermöglichen soll. Medusa passt dafür einige Parameter der Anfrage an und fügt Benutzerinformationen hinzu, welche anschliessend vom Back-End zur Authentifizierung verwendet werden können.

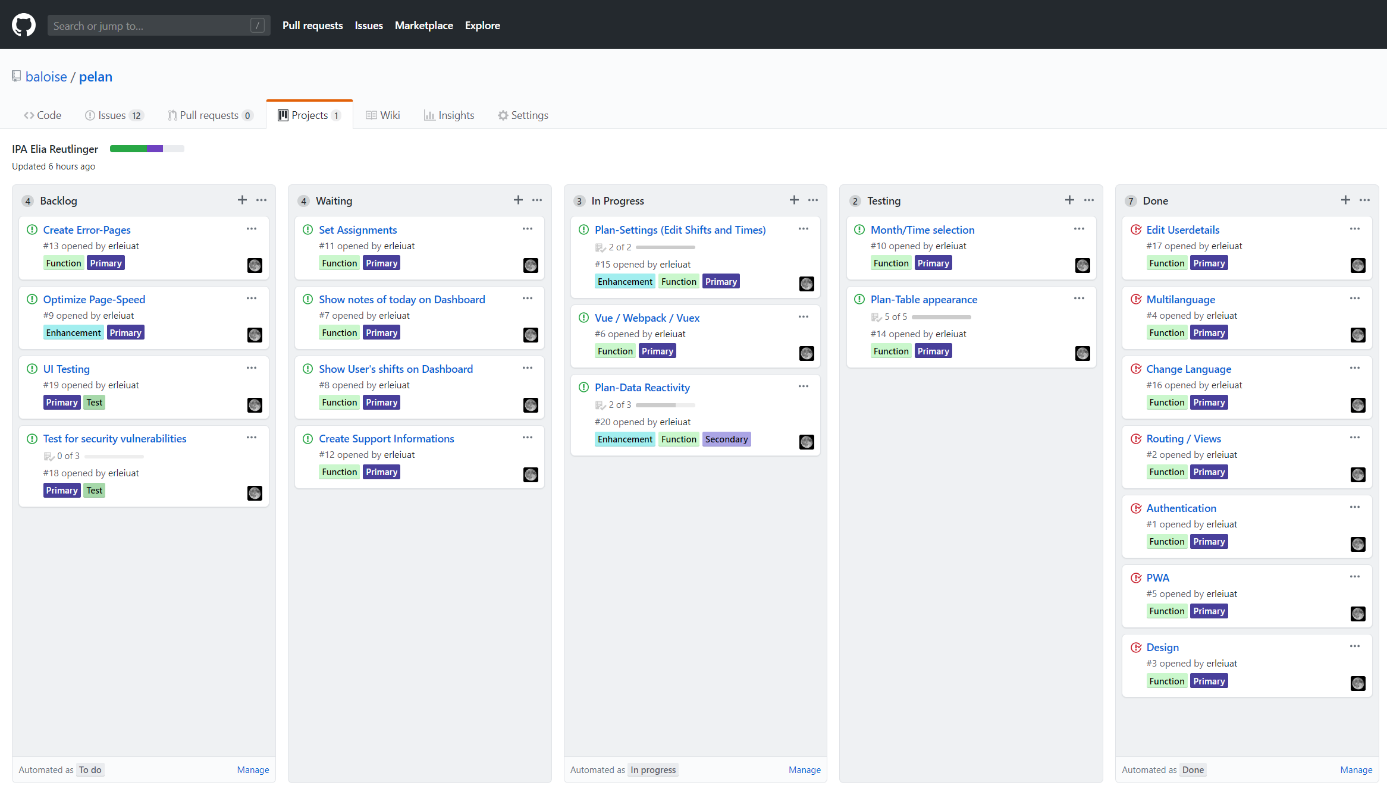
## Umsetzung Projektmethode & Vorgehensmodell

Die Projektmethode während der IPA wurde mit der detaillierten Aufgabenstellung auf Kanban festgelegt. Dies wurde so entschieden, weil die Baloise in der Informatik bereits vermehrt Kanban einsetzt und man damit auch Projekte in einem kleinen Zeitraum erfassen kann.

Kanban bildet die Arbeitsschritte der Implementierung als einzelne «Tasks/Aufgaben» ab, und ordnet diese unterschiedlichen Stationen zu. Diese Stationen beschreiben den aktuellen Stand des Tasks, wobei ich mich für übliche Stationen der Anwendungsentwicklung entschieden habe:

1. Backlog: Hier werden alle Tasks eingeteilt, die im Projekt benötigt werden und noch nicht begonnen wurden. Sie können durch die Anforderungen der Aufgabenstellung oder während der Umsetzung dieser Anforderungen entstehen.
2. Waiting: Diese «Zwischenstation» enthält alle Tasks, welche gerade nicht bearbeitet werden können, da dazu andere Tasks und/oder Abklärungen gemacht werden müssen.
3. In Progress: Wenn ein Task aktiv ist, wird er in dieser Station eingeteilt. Hier findet die Umsetzung bzw. Implementierung des Tasks statt, wobei dies zusammen mit dem Entwurf geschieht.
4. Testing: Bevor ein Task abgeschlossen werden kann, sollte mit dieser Station die korrekte Umsetzung getestet werden. Bei gewissen Tasks wie Abklärungen kann diese Station übersprungen werden.
5. Done: Alle erledigten Tasks werden hier abgelegt. Sie befinden sich dabei in einem Zustand, in welchem sie problemlos im Betrieb eingesetzt werden können. Dazu kommt eine Gelöst-Markierung. \*

\* Das Kanban wird auf GitHub durch ein «Project-Board» dargestellt. Die einzelnen Tasks sind dabei GitHub-Issues, welche nach Abschluss als «Closed/Gelöst» markiert werden können. Issues können bei Bedarf auch Labels tragen, welche auf die Art des Issues bzw. der Aufgabe schliessen lassen.



Kanban bildet an sich bereits eine Art von Vorgehensmodell, bei welchem die Anzahl paralleler Tasks und somit die Durchlaufzeit möglichst klein gehalten wird. Dadurch können Probleme und Engpässe schnell sichtbar gemacht werden. Die Stationen bzw. Arbeitsschritte eines Kanban-Boards entsprechen häufig denen des Wasserfallmodells, wobei für die IPA die Schritte «Anforderung», «Entwurf» und «Wartung» weggelassen wurden. Die Anforderungen wurden bereits vor Beginn der IPA definiert, und der Entwurf wird aus zeitlichen und technischen Gründen mit der Implementierung erarbeitet.[[9]](#footnote-9)

## Anforderungen

Das Ziel der IPA ist eine funktionsfähige Anwendung. Dazu bestehen funktionale Anforderungen an die Anwendung sowie nicht-funktionale während der Entwicklung, wobei die folgend definierten das Minimum an zu erfüllenden Anforderungen darstellen. Sie wurden durch die detaillierte Aufgabenstellung der IPA erstellt und zusätzlich vertieft, wobei die gesamten Anforderungen ersichtlich werden.



### Funktional

* Grundlegende Voraussetzungen
  + Einhaltung definierter Firmenstandards
  + Benutzerfreundlichkeit über die gesamte Anwendung
  + Unveränderte Verwendbarkeit für den produktiven Betrieb (1:1)
  + Einhaltung von Coding-Standards bzw. Best-Practice
  + Mehrsprachigkeit (mindestens 2 Sprachen)
  + Absicherung gegen gängige Angriffe
* Funktionen
  + Einsätze darstellen
  + Einsätze erstellen, bearbeiten und entfernen
  + Schichten und Tageszeiten erstellen, bearbeiten und löschen
  + Berechtigungen von Teammitgliedern bearbeiten
  + Sprache anpassen

### Nicht-Funktional

* Dokumentation des Fortschritts auf GitHub
* Dokumentierung von möglichen Änderungen und Problemen
* Dokumentation des Projekts als IPA-Bericht
  + Führung eines Arbeitsjournals
  + Testkonzept und Testprotokoll
  + Sicherheitskonzept und Sicherheitsprüfung
* Einsatzfähige Demo-Version mit Testdaten
* Präsentation des Projekts

## Ziele

### Projekt

Das Endprodukt des Projekts soll dem Helpdesk eine Alternative zur Excel-Tabelle liefern. Dabei sollen die Übersichtlichkeit und Verwaltung sichtlich verbessert worden sein. Weitere Ziele in Berücksichtigung der IPA Kriterien sind:

* Implementierung mit Führung eines Kanban-Boards.
* Dokumentierung und Veröffentlichung des Fortschritts auf GitHub.
* Für die Produktion einsetzbares Endprodukt.
* Wiedererkennbarkeit der Baloise im Endprodukt.
* Mehrsprachige Anwendung mit Erweiterungspotential.
* Berücksichtigung gängiger Sicherheitskriterien.

### Persönlich

Persönliche Ziele, welche ich mir selbst für die IPA gesetzt habe, sind:

* Vertiefung des Fachwissens zu Vue.js.
* Problemlose Abwicklung durch gute Planung (Zeitplan).
* Termingerechter Abschluss des Projekts.
* Umsetzung einer optimalen, produktiv anwendbaren Anwendung.

## Informationssicherheit & Datenschutz

Datenschutz ist bei der Baloise eine Voraussetzung für alle Anwendungen. Man muss strenglichst darauf achten, dass keine Daten nach aussen gelangen, die auf betriebsinterne Abläufe oder Personendaten schliessen lassen könnten.[[10]](#footnote-10)

Da die Anwendung selbst jedoch keine Daten speichert, sondern lediglich mit dem Back-End kommuniziert, müssen hier keine weiteren Vorsichtsmassnahmen getroffen werden. Das Back-End entscheidet auf welche Daten von wem zugegriffen werden kann, und übernimmt den gesamten Authentifizierungs-Prozess. Die Verbindung zwischen Front-End und Back-End benötigt in Berücksichtigung der aktuellen Anforderungen auch keine zusätzliche Massnahme, da beide nur betriebsintern zugänglich sein sollen.

Sicherheitsrisiken, welche im Front-End zu Problemen führen könnten wie DOM-Based und Reflected Cross-Site-Scripting müssen unabhängig vom Back-End geprüft werden. Deshalb werden dazu ein Sicherheitskonzept sowie entsprechende Tests/Überprüfungen erstellt.

## Konzepte

### Testkonzept (Use-Cases)

Einführung

Die folgenden Testfälle sollen überprüfen, ob die funktionalen Anforderungen zu den Funktionen der Anwendung korrekt implementiert wurden. Sie bilden die Use-Cases der Anwendung ab, welche aus den vorbestimmten Tests der detaillierten Aufgabenstellung abgeleitet wurden. Alle Testfälle folgen der «Black-Box» Methode und setzen die Verwendung eines Notebooks voraus. Dabei soll auf einem lokalen Produktions-Build der Anwendung manuell (von Hand) und aus der Sicht eines Benutzers getestet werden.

Testanlage / Voraussetzungen

Dieses Testkonzept richtet sich nach einer manuellen Durchführung der Tests in der Benutzeroberfläche. Dazu ist ein Computer mit einer lokalen, aktuellen Instanz der Pelan-API und einem Produktions-Build der Vue.js Anwendung notwendig. Neben den vollständig implementierten Ansichten des Pelan-UIs müssen in der API 2 Testbenutzer zur Verfügung stehen, welche unterschiedliche Rollen haben. Weitere Voraussetzungen, die abhängig vom Testfall bestehen, werden in der Beschreibung des betroffenen Testfalls notiert.

Methode und Mittel

Mit der «Black-Box» Methode wird lediglich die oberflächliche Funktionalität der Anwendung getestet. Geplant ist die Durchführung der Tests von Hand ohne weitere Hilfsmittel. Bei der Verwendung von Hilfsmitteln dürfen diese keinen Einfluss auf die Testresultate haben und müssen im Protokoll erwähnt werden.

#### FT-001 – Einsätze darstellen

Name FT-001

Beschreibung Auf der Plan-Ansicht sollen alle Teammitglieder die geplanten Einsätze auf einer geeigneten Tabelle einsehen können.

Voraussetzungen Mindestens ein Einsatz vorhanden (FT-002).

Szenario

|  |  |  |
| --- | --- | --- |
| Schritt | Aktion | Erwartetes Resultat |
| 1 | Plan-Ansicht aufrufen. | Einsatz erscheint im Plan |

#### FT-002 – Einsätze erstellen, bearbeiten und löschen

Name FT-002

Beschreibung Teamleiter können Einsätze erstellen, bearbeiten und löschen.

Szenario (als Teamleiter)

|  |  |  |
| --- | --- | --- |
| Schritt | Aktion | Erwartetes Resultat |
| 1 | Plan-Ansicht aufrufen. | Plan wird angezeigt. |
| 2 | Leeres Feld anklicken. | Bearbeitungs-Dialog öffnet sich. |
| 3 | Schicht auswählen & «Speichern» drücken. | Einsatz erscheint im Plan. |
| 4 | Feld mit Einsatz anklicken. | Bearbeitungs-Dialog öffnet sich. |
| 5 | Schicht ändern & «Speichern» drücken. | Änderung erscheint im Plan. |
| 6 | Feld mit Einsatz anklicken. | Bearbeitungs-Dialog öffnet sich. |
| 7 | «Löschen» drücken. | Einsatz erscheint nicht mehr im Plan. |

#### FT-003 – Sprache anpassen

Name FT-003

Beschreibung Jeder Benutzer kann die angezeigte Sprache seiner Oberfläche anpassen.

Szenario

|  |  |  |
| --- | --- | --- |
| Schritt | Aktion | Erwartetes Resultat |
| 1 | Einstellungen-Ansicht aufrufen. | Einstellungen werden angezeigt. |
| 2 | Sprache ändern und «Speichern» drücken. | Die Sprache der Oberfläche ändert sich. |

#### FT-004 – Schichten erstellen, bearbeiten und löschen

Name FT-004

Beschreibung Teamleiter können Schichten erstellen, bearbeiten und löschen.

Szenario (als Teamleiter)

|  |  |  |
| --- | --- | --- |
| Schritt | Aktion | Erwartetes Resultat |
| 1 | Bearbeitungs-Ansicht aufrufen. | Schichten werden angezeigt. |
| 2 | Neue Schicht erstellen & «Speichern» drücken. | Schicht erscheint in der Liste. |
| 3 | Schicht bearbeiten & «Speichern» drücken. | Liste wird aktualisiert. |
| 4 | Schicht bearbeiten & «Löschen» drücken. | Warnung wird angezeigt. |
| 5 | Warnung bestätigen. | Schicht erscheint nicht mehr in der Liste. |

#### FT-005 – Zeiten erstellen, bearbeiten und löschen

Name FT-005

Beschreibung Teamleiter können Tageszeiten erstellen, bearbeiten und löschen.

Voraussetzungen Mindestens eine Zeit vorhanden (FT-005).

Szenario (als Teamleiter)

|  |  |  |
| --- | --- | --- |
| Schritt | Aktion | Erwartetes Resultat |
| 1 | Bearbeitungs-Ansicht aufrufen. | Zeiten werden angezeigt. |
| 2 | Neue Zeit erstellen & «Speichern (+)» drücken. | Zeit erscheint in der Liste. |
| 3 | Name & Position der Zeit bearbeiten & «Speichern» drücken. | Erfolgsmeldung wird angezeigt (Änderungen gespeichert). |
| 4 | «Löschen» und anschliessend «Speichern» drücken. | Erfolgsmeldung wird angezeigt (Zeit wurde entfernt). |

#### FT-006 – Berechtigungen anpassen

Name FT-006

Beschreibung Teamleiter können die Berechtigungen bzw. Gruppe von Teammitgliedern anpassen.

Szenario (als Teamleiter)

|  |  |  |
| --- | --- | --- |
| Schritt | Aktion | Erwartetes Resultat |
| 1 | Einstellungen-Ansicht aufrufen. | Benutzerliste wird angezeigt. |
| 2 | Rolle eines Benutzers ändern (Normal🡪Admin). | Erfolgsmeldung erscheint (Änderungen gespeichert). |
| 3 | Test-API zur Anmeldung mit bearbeitetem Benutzer anpassen (Login-Key ändern), Browser-Cookies löschen und Anwendung neu laden. | Angemeldet mit bearbeitetem Benutzer, Admin-Funktionen werden angezeigt. |

### Sicherheitskonzept

Das Sicherheitskonzept orientiert sich an den aktuellsten «OWASP Top 10 Risiken für Anwendungs-sicherheit 2013\*»[[11]](#footnote-11). Damit können die gängigsten Angriffsmöglichkeiten analysiert und vermieden werden, wobei die Entwicklung des Front-Ends nur von einem Bruchteil der Top 10 Risiken betroffen sein kann. Die meisten Risiken müssen bei der Entwicklung des Back-Ends berücksichtigt werden, da hier die gesamten Daten verarbeitet und gespeichert werden. Folgend werden die Top 10 kurz erläutert und markiert, falls sie im Front-End berücksichtigt werden sollten.

* A1 – Injection

*Muss im Back-End verhindert werden, da hier SQL-Abfragen vorgenommen werden.*

* A2 – Fehler in Authentifizierung und Session-Management

*Muss im Back-End verhindert werden, da die gesamte Authentifizierung dort stattfindet.*

* A3 – Cross-Site-Scripting (wird in 3 Arten unterteilt)
  + Stored: Betrifft gespeicherte Daten und muss im Back-End verhindert werden. Rückgabewerte könnten jedoch im Front-End zu Problemen führen.
  + Reflected: Betrifft Formulareingaben welche vom Back-End ungespeichert zurückgegeben, und anschliessend im Front-End verwendet werden. Sollte im Back-End verhindert, kann aber im Front-End geprüft werden.
  + Dom-Based: Betrifft nur das Front-End und muss dort verhindert werden.
* A4 – Unsichere direkte Objektreferenzen:

*Muss im Back-End verhindert werden, da hier betroffene Abfragen mit dynamischen Parametern stattfinden würden.*

* A5 – Sicherheitsrelevante Fehlkonfiguration

*Betrifft die Server-Konfiguration im Back-End.*

* A6 – Verlust der Vertraulichkeit sensibler Daten

*Betrifft die Verschlüsselung von sensiblen Daten (Passwörter, Kreditkarten) im Back-End.*

* A7 – Fehlerhafte Autorisierung auf Anwendungsebene:

*Betrifft das Front-End, da hier Formulare und Seiten in Abhängigkeit von den Berechtigungen des angemeldeten Benutzers angezeigt werden.*

* A8 – Cross-Site-Request-Forgery (CSRF)

*Kann Front- und Back-End betreffen, je nachdem wie Tokens geprüft und eingesetzt werden.*

* A9 – Nutzung von Komponenten mit bekannten Schwachstellen

*Betrifft Front- und Back-End.*

* A10 – Ungeprüfte Um- und Weiterleitungen

*Betrifft Front- und Back-End.*

\* Die OWASP Top 10 von 2013 sind in Bezug auf Pelan inhaltlich gleich relevant wie die neuere Version von 2017, einzig die Reihenfolge der Risiken hat sich geändert.

#### A3 – Cross-Site-Scripting: Reflected/Stored

Grundlegende Erläuterung

Reflektiertes XSS ist möglich, wenn Benutzereingaben vom Back-End ungeprüft zurückgegeben, und anschliessend im Front-End eingesetzt werden. Wenn das Front-End Rückgabewerte des Back-Ends allgemein nicht überprüft, besteht zusätzlich die Gefahr von Stored-XSS.

Allgemeines Beispiel

Die Anwendung nutzt im Front-End eine einfache Variable, um die Berechtigung des Benutzers zu speichern. Ein Angreifer ändert eine Eingabe so ab, dass darin Code zum Verfälschen dieser Variable enthalten ist. Die Eingabe wird an den Server gesendet, und dieser gibt den Wert ohne Prüfung wieder zurück. Das Front-End nutzt den Wert zur Ausführung eines Scripts, wobei die Berechtigungen des Benutzers durch eine fehlende Überprüfung angepasst werden.

Vermeidungsmöglichkeiten

1. Rückgabewerte des Back-Ends vorgängig Überprüfen oder nicht unspezifisch evaluieren.
2. Durch korrekte Anwendung von Vue.js und den genutzten Abhängigkeiten deren vordefinierte Verteidigungs-mechanismen anwenden.

Überprüfung

Wird bei betroffenen Stellen im Code korrekt mit den Abhängigkeiten gearbeitet? Werden ansonsten eine Überprüfung und spezifische Verarbeitung durchgeführt?

#### A3 – Cross-Site-Scripting: Dom-Based

Grundlegende Erläuterung

Wenn unabhängig vom Back-End Variablen eingesetzt werden, welche vom Benutzer bearbeitet werden könnten, besteht die Gefahr von Dom-Based XSS.

Allgemeines Beispiel

Die Anwendung arbeitet mit Variablen, welche aus der URL gelesen werden. Ein Angreifer bearbeitet die URL in seinem Browser und ersetzt den Wert der Variable mit einem Script. Die Anwendung übernimmt das Script ungeprüft aus den Variablen und es kommt zu dessen Ausführung.

Vermeidungsmöglichkeiten

1. Auf die Nutzung der URL für Variablen verzichten.
2. Betroffene Werte vorgängig Überprüfen oder nur spezifisch verarbeiten.
3. Durch korrekte Anwendung von Vue.js und den genutzten Abhängigkeiten deren vordefinierte Verteidigungs-mechanismen anwenden.

Überprüfung

Wird bei betroffenen Stellen im Code korrekt mit den Abhängigkeiten gearbeitet? Werden ansonsten eine Überprüfung und spezifische Verarbeitung durchgeführt?

#### A7 – Fehlerhafte Autorisierung auf Anwendungsebene

Grundlegende Erläuterung

Wenn die Anzeige von bestimmten Ansichten oder Funktionen abhängig von einem Autorisierungs-mechanismus ist, besteht die Gefahr, dass Fehler zu unberechtigten Zugriffen führen können.

Allgemeines Beispiel

Der Mapping-Router einer Anwendung nutzt Pfadangaben aus der URL, um die zugehörige Seite zu laden. Wenn vor dem Laden einer Seite nicht überprüft wird, ob der Benutzer ausreichend Berechtigungen für diese hat, kann er theoretisch auf alle Seiten der Anwendung zugreifen. Wenn er dort auch Eingaben machen kann, welche im Back-End ungeprüft verarbeitet werden, kann er unberechtigt Funktionen ausnutzen.

Vermeidungsmöglichkeiten

1. Berechtigungen bei Anfragen im Back-End prüfen (generell notwendige Absicherung).
2. Vor dem Laden einer Seite und Funktionen die Berechtigungen überprüfen.

Überprüfung

Kann ich durch bearbeiten der URL eine Seite ohne entsprechende Berechtigungen aufrufen? Werden meine Anfragen im Back-End überprüft?

#### A8 – Cross-Site-Request-Forgery (CSRF)

Grundlegende Erläuterung

Wenn die Authentifizierung einer Anfrage an das Back-End mit Daten ausgeführt wird, welche der Browser mit jeder Anfrage automatisch übermittelt, besteht die Gefahr von CSRF.

Allgemeines Beispiel

Bei einer Anfrage fügt der Browser automatisch ein Cookie mit einem Token hinzu. Das Back-End verwendet den Token zur Authentifizierung. Da der Browser dies automatisch tut, könnte ein Angreifer sein Opfer zu einer versteckten Anfrage verleiten, ohne dass das Opfer etwas davon mitbekommt. Dies kann ein verstecktes Script auf einer anderen Webseite sein, welche vom Opfer aufgerufen wird.

Vermeidungsmöglichkeiten

1. Tokens nicht im Browser sondern in der Anwendung an Anfragen anfügen (Achtung: XSS!).
2. Um die XSS Gefahr von Punkt 1 zu vermeiden, eine Kombination von vom Browser hinzugefügten und in der Anwendung angehängten Token verwenden, und diese im Back-End überprüfen.

Überprüfung

Kann ich von einer anderen Webseite eine Anfrage erfolgreich an das Back-End senden?

#### A9 – Nutzung von Komponenten mit bekannten Schwachstellen

Grundlegende Erläuterung

Wenn eine Anwendung mit Komponenten und/oder Abhängigkeiten arbeitet, welche Sicherheitsschwachstellen aufweisen, können diese von Angreifern ausgenutzt werden. Diese Schwachstellen können dabei von jeglicher Art sein, weshalb die Berücksichtigung von diesem Sicherheitsrisiko eine Voraussetzung für den Schutz vor allen anderen Risiken ist.

Überprüfung

Sind zu verwendeten Abhängigkeiten Schwachstellen bekannt? Wenn ja, wurden diese in der Anwendung eigenständig eliminiert?

#### A10 – Ungeprüfte Um- und Weiterleitungen

Grundlegende Erläuterung

Einerseits betrifft dies dieselbe Problematik wie A7 (Mapping-Router überprüft Berechtigungen vor dem Anzeigen nicht) andererseits sind hier auch Weiterleitungen auf externe Webseiten gemeint.

Allgemeines Beispiel

Benutzer können in ihren Profilen eine Homepage angeben. Wenn ein anderer Benutzer nun auf den dazu generierten Link im Profil dieses Benutzers klickt, wird er auf dessen vermeintliche Homepage weitergeleitet. Diese Seite könnte jedoch auch schädlichen Code enthalten.

Vermeidungsmöglichkeiten

1. Grundsätzlich solche Weiterleitungen vermeiden.
2. Wenn Punkt 1 nicht umsetzbar ist, alle Weiterleitungen verifizieren oder den Benutzer zumindest warnen.

Überprüfung

Falls solche Weiterleitungen existieren, ist die Ziel-Webseite verifiziert/frei von schädlichem Code?

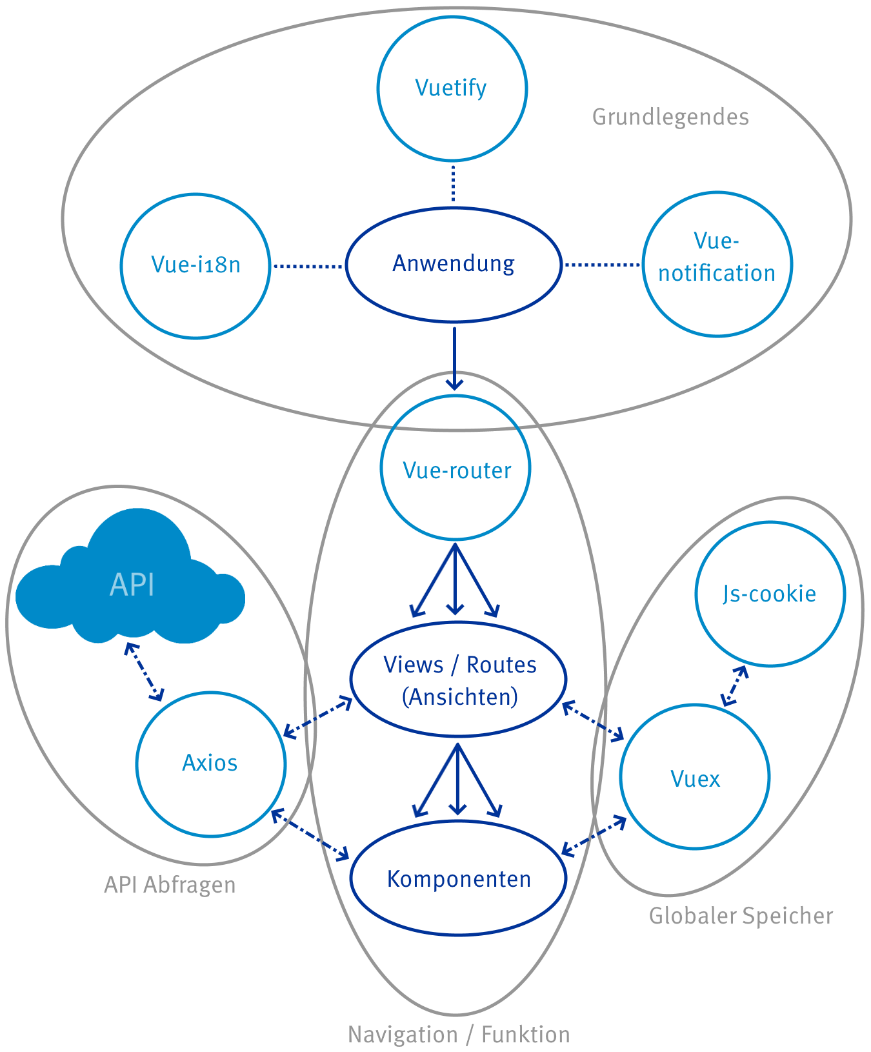
# Realisierung

## Grundlegende Konfiguration

Zur grundlegenden Konfiguration einer Vue.js Anwendung gehören unterschiedliche Aufgaben, welche vor Beginn des Implementierens der eigentlichen Funktionen zu erledigen sind. Dazu gehört unter anderem das Initialisieren einer Vue-Instanz und das Importieren und Zuweisen von Abhängigkeiten zu dieser Instanz. Gewisse Abhängigkeiten benötigen auch noch eine zusätzliche Konfiguration, welche in diesem Schritt vorgenommen wurde. Die wichtigsten Abhängigkeiten sind:

* Vuetify (Bibliothek mit zahlreichen Design-Komponenten).
* Vue-notification (Verarbeitung und Anzeige von Benachrichtigungen).
* Vue-router (Route-Mapping, sowie Laden und Wechseln von Seiten).
* Vuex (State-Management-Pattern => globaler, dynamischer Speicher für Daten der App).
* Vue-i18n (Mehrsprachigkeit / Übersetzungen definieren und dynamisch umschalten).
* Axios (Promise-Based HTTP-Client, Verarbeitung und Durchführung von Anfragen an die API).
* Js-cookie (Verwaltung von Browser-Cookies).

Mit Vuex können Daten global gespeichert und durch jede Komponente abgerufen werden. Hier kann man beispielsweise die Anzahl von Abfragen an die API vermindern, indem man Daten, welche auch von anderen Komponenten benötigt werden, gleich in diesem Speicher ablegt. Folgend eine Übersicht vom Nutzen der Abhängigkeiten in Relation zu den Ebenen der Anwendung:



### Ansichten / Routes

Vue.js ist ohne weitere Abhängigkeiten nur für Single-Page-Applications konzipiert. Da die Anwendung aber mehrere Ansichten benötigt, wird das Modul «vue-router» eingesetzt. Dadurch lassen sich unterschiedliche Ansichten (auch «Routen» genannt) dynamisch einbinden, was durch eine Navigation ausgelöst werden kann.

Für die grundlegende Konfigurierung wurden auch gleich alle Routen definiert, welche später benötigt und implementiert worden sind.

* Dashboard (/): Startseite mit grundlegenden Informationen zum aktuellen Tag.
* Plan (/plan): Ansicht für den Einsatzplan.
* Plan bearbeiten (/plansettings): Einstellungen zu Schichten und Tageszeiten.
* Support (/help): Kurze Sammlung wichtiger Kontaktinformationen.
* Einsitellungen (/settings): Sprachauswahl und Benutzerverwaltung (Teamleiter).

Zusätzlich wurden 2 Routen für die Fehlermeldungen 404 und 401 erstellt.

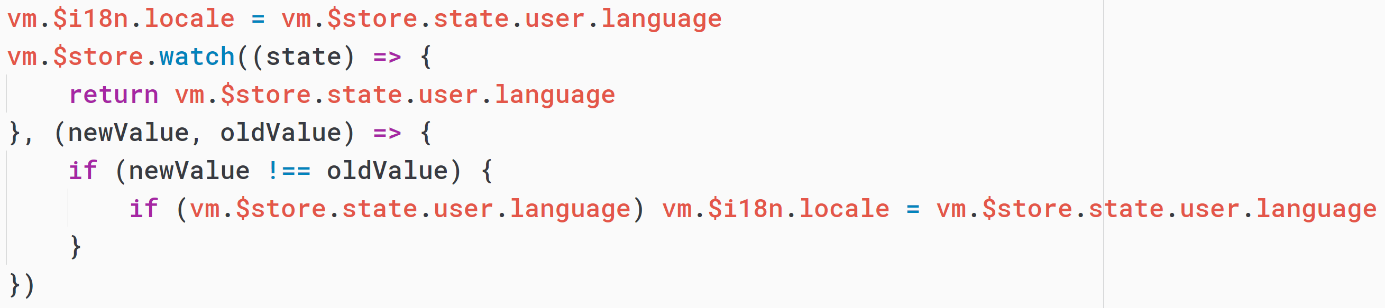
Mit jeder Route wurde sogleich eine Berechtigungsstufe definiert, welche vor der Anzeige einer Seite überprüft wird. Damit wird beispielsweise verhindert, dass ein normaler Benutzer zu den Planeinstellungen gelangen kann.



### Mehrsprachigkeit

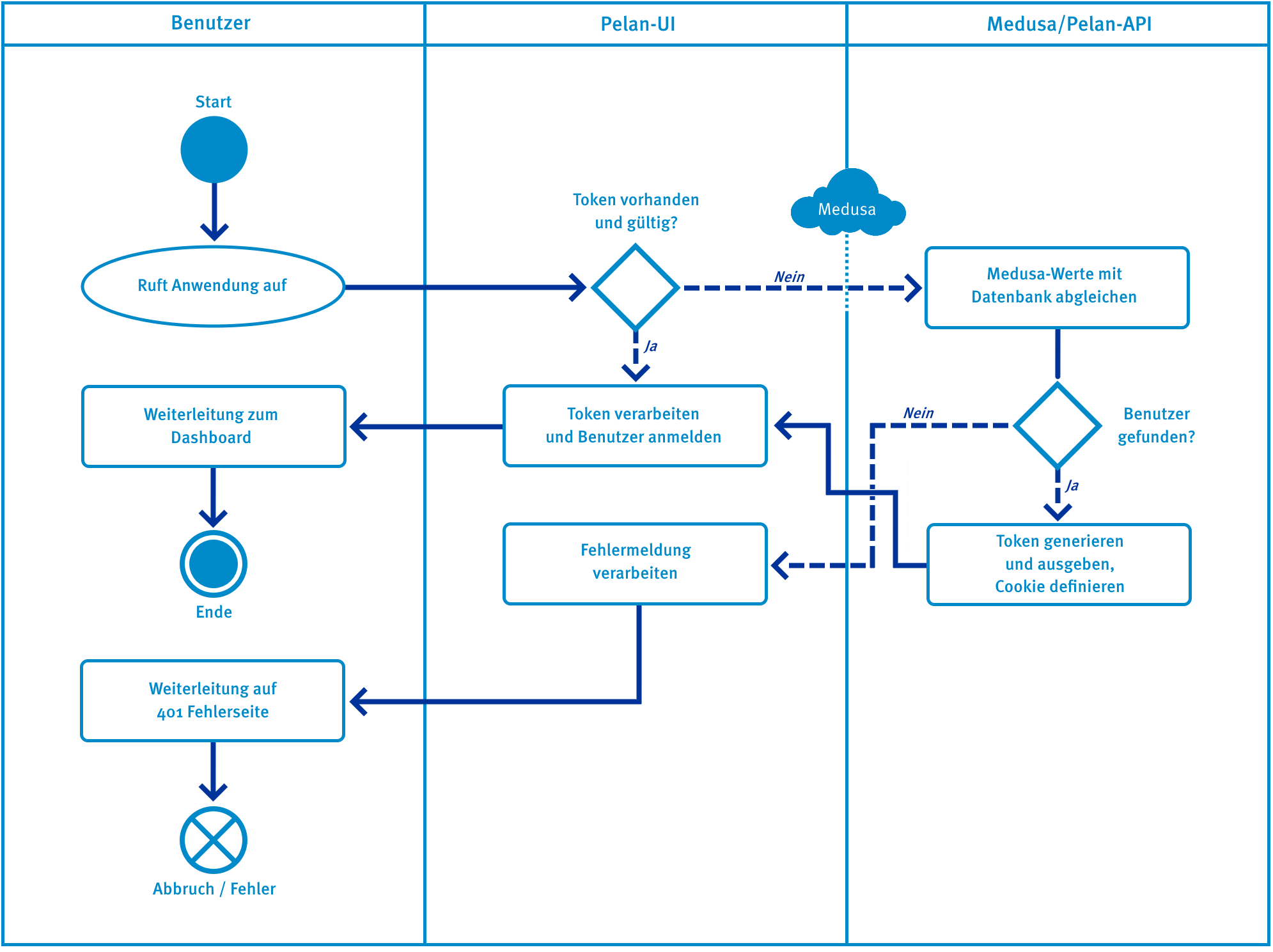
Um den Benutzern eine Sprachauswahl anbieten zu können, muss die Anwendung mehrsprachig konzipiert sein. Dazu wird das Modul «vue-i18n» eingesetzt, was Übersetzungen global und auf Komponentenebene realisierbar macht. Das Modul ist eines der gängigsten für Übersetzungen und lässt sich sehr einfach implementieren. Es kann jederzeit mit weiteren Sprachen erweitert werden, wobei für die IPA vorerst nur die Sprachen Deutsch und Englisch eingebaut wurden.

Ein Benutzer kann die Sprache seiner Oberfläche in der Einstellungen-Ansicht anpassen, was in den Benutzer-Details gespeichert wird. Durch eine «Beobachter»-Funktion von Änderungen an diesen Details wird schlussendlich die Sprache der Oberfläche aktualisiert.



### Authentifizierung

Durch die Verwendung einer Rest-API sollen Anfragen einen JSON-Web-Token enthalten. Dieser enthält Informationen zum Benutzer und wird von der API mit einer Signatur erstellt.[[12]](#footnote-12) Um ihn zu erhalten sendet die Anwendung eine Login-Anfrage an die API. Die API authentifiziert den Benutzer (was produktiv per Airlock-Medusa[[13]](#footnote-13) Single-Sign-On geschieht), erstellt einen Token und sendet diesen zurück.



Die Anwendung passt anschliessend den standardisierten «Authorization»-Header von Anfragen an, wobei der empfangene Token angefügt wird. Damit wird ab sofort jede Anfrage mit dem Token gesendet und kann von der API authentifiziert werden.

Ob eine Authentifizierung notwendig ist wird vor dem Darstellen jeder Ansicht geprüft. Beispielsweise kann die Support-Ansicht auch ohne Authentifizierung aufgerufen werden, während die Ansicht zu den Plan-Einstellungen neben normalen Berechtigungen auch die Teamleiter-Rolle benötigt. Im Code sieht dieser Ablauf folgendermassen aus:

1. Prüfen ob Token notwendig und vorhanden ist ( *beforeResolve()* )

Wenn nötig und nicht vorhanden weiter zu Schritt 2, ansonsten direkt zu Schritt 3.



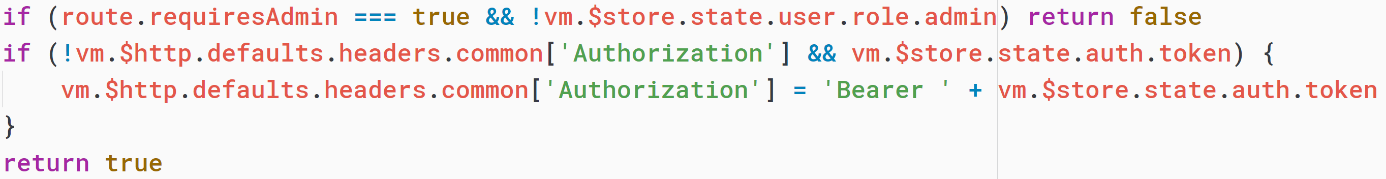
1. Login-Anfrage an API senden ( *getLogin()* )

Wenn erfolgreich Login-Prozess durchführen und «True» antworten, ansonsten Fehlermeldung und «False» zurückgeben. True/False-Werte anschliessend in Schritt 1 verarbeiten.

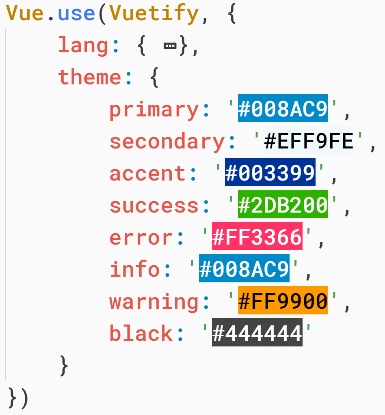


1. Abschliessende Prüfung/Verarbeitung ( *checkPerms()* )

Rolle prüfen und je nach Ansicht «True» oder «False» zurückgeben, was wiederum in Schritt 1 verarbeitet wird. Anschliessend Token an API-Anfragen binden.



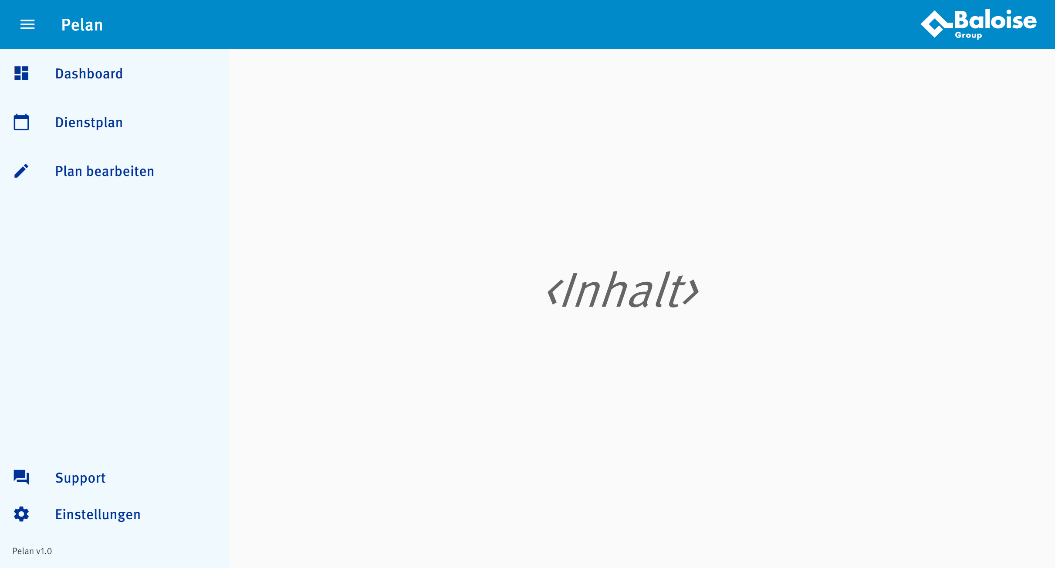
## Umsetzung Design

Die Umsetzung des Designs geschah unter Berücksichtigung der Firmenstandards mit Vuetify[[14]](#footnote-14). Dabei wurde ein standardisierter Aufbau der Oberfläche genutzt, welcher «Out of the Box» mit Vuetify dazu kommt. Dazu gehört eine Toolbar am oberen Bildschirmrand und eine Navigation auf der linken Seite, während der verbleibende Bereich vollständig für die unterschiedlichen Ansichten gedacht ist. Damit ist dieser also dynamisch, wobei die anderen 2 Elemente über die gesamte Anwendung am gleichen Ort bleiben.

Bei der Einbindung von Vuetify in die Vue-Instanz können zahlreiche Parameter angepasst werden. Hier wurde deshalb die zu verwendende Farbpalette an die Firmenstandards angepasst, wodurch sie später über die gesamte Anwendung eingesetzt werden konnte. Zusätzlich wurden globale CSS-Anpassungen vorgenommen, welche die Schriftarten importieren und die Schriftgrössen für unterschiedliche Elemente wie beispielsweise Titel, Überschrift und Text festlegten.

Die Toolbar enthält den Namen der Anwendung «Pelan» sowie das Baloise-Group Logo. Die Hintergrundfarbe wurde auf «Primary» festgelegt, wodurch die Farbe aus der Variablen «Primary» in der zuvor festgelegten Farbpalette eingesetzt wird.

Die Navigation enthält neben einem Verweis auf die Version der Anwendung alle Navigationspunkte, die dem Benutzer zur Verfügung stehen. Welche dies sind wird anhand der Benutzergruppe und der im Router definierten Berechtigungsstufe entschieden. Für eine bessere Benutzerfreundlichkeit wird jeder Navigationspunkt mit einem Icon dargestellt, welches auf die Funktion der Ansicht verweisen soll.



## Umsetzung der Ansichten

Dieses Kapitel soll die Umsetzung der einzelnen Ansichten sowie deren Funktionen erläutern. Dazu werden auch Probleme und Schwierigkeiten genannt, die teilweise vorgefallen sind. Die Gestaltung sowie der Detailgrad der folgenden Kapitel ist abhängig von der Komplexität der jeweils umgesetzten Ansicht, wobei das Verständnis des Lesers zum Gesamtsystem in Fokus liegt.

### Support

Die unkompliziertesten Ansichten sind das Dashboard und die Support-Seite. Hier befinden sich keine Funktionen, mit welchen der Benutzer Einfluss auf die Daten der Anwendung haben könnte, sondern nur statische Elemente. Für den Support werden nur grundlegende Kontaktinformationen angezeigt, welche aus der Adresse, Telefonnummer und E-Mail des Helpdesks, sowie (vorläufig) meinen persönlichen Kontaktdaten bestehen.

[**TODO**: Bild von Support-Infos]

### Dashboard

Das Dashboard besteht aus einer kleinen Begrüssungs-Nachricht sowie einer Auflistung der Notizen zu allen Einsätzen des Tages. Zusätzlich kann ein Benutzer hier die vorgesehenen Einsätze des aktuellen Tages sehen. Die Idee dahinter ist eine Startseite, auf der ein Benutzer die wichtigsten Informationen schnell und einfach abrufen kann, ohne dafür den Einsatzplan öffnen zu müssen. Des Weiteren soll die erste Seite, auf die ein Benutzer beim Öffnen der Anwendung geleitet wird, sehr performant sein, da dies die Ladezeit der gesamten Anwendung sehr beschleunigen kann.

[**TODO**: Bild von Dashboard]

Um die Informationen für das Dashboard zu erhalten sind mehrere Anfragen an die API nötig, da noch kein Endpunkt für das Abrufen durch eine einzige Anfrage existiert. Es entstanden 2 Anfragen für die Notizen und Einsätze, wobei zu den Einsätzen keine genauen Informationen zu den Schichten und Tageszeiten geliefert werden. Stattdessen werden Identifikations-Nummern angegeben, durch welche mit 2 weiteren Abfragen die Schichten und Tageszeiten zugeordnet werden können. Die Schichten und Tageszeiten werden dann in den globalen Speicher geladen, da sie auch im Einsatzplan und den Plan-Einstellungen benötigt werden könnten.

[**TODO**: Codeausschnitt einer API-Abfrage]

Es wurde abschliessend überprüft, ob die Geschwindigkeit der Seite akzeptabel ist und als Startseite in Frage kommt. Dies wurde mit dem Lighthouse-Audit Tool[[15]](#footnote-15) von Google erledigt, was ein gängiges Tool zur Auswertung der Akzeptanz von Webanwendungen ist.

[**TODO**: Screenshot Lighthouse]

Damit wird ersichtlich, dass das Ziel durchaus erreicht wurde, während die anderen Kriterien auch eine sehr gute Bewertung erhalten haben. Lighthouse spricht von einer optimalen Anwendung, wenn alle Kriterien einen Wert von mindestens 90 Punkten erreichen.

### Einstellungen

In den Einstellungen soll jeder Benutzer seine Sprache anpassen können. Ein Teamleiter soll hier zusätzlich die Rollen der Teammitglieder bearbeiten können, was in der Darstellung eine Abhängigkeit zu den Berechtigungen des aktuellen Benutzers bildet.

Beim Speichern der Einstellungen wird eine Anfrage an die API ausgelöst, welche zu Änderungen in der Datenbank führt. Dabei muss berücksichtigt werden, dass bei einer Änderung am aktuell angemeldeten Benutzer die Inhalte des JSON-Web-Tokens verändert werden. Wenn die API erkennt, dass die Änderungen zum aktuell angemeldeten Benutzer gehören, wird ein neuer Token mit den aktualisierten Informationen generiert und zurückgegeben. Der alte Token wird dadurch ungültig und muss mit dem empfangenen ausgetauscht werden, weshalb hier der Login-Prozess erneut ausgeführt wird.

[**TODO**: ACT-Diagramm]

[**TODO**: Bild der Einstellungen-Ansicht]

Anfragen an die API zu den Einstellungen des Benutzers folgen einem klaren Schema, welches auch Änderungen an anderen Benutzerdaten zulässt. Da nicht viel Aufwand nötig war, um die anderen Parameter in den Benutzereinstellungen bearbeitbar zu machen, wurden diese gleich mit implementiert. Deshalb können Teamleiter zusätzlich zur Rolle des Benutzers auch andere Informationen anpassen. Das Ganze wurde in einer Tabelle dargestellt, wobei sich die Werte durch einfaches Anklicken ändern lassen.

### Plan-Einstellungen

Damit ein Teamleiter alle möglichen Parameter am Einsatzplan bearbeiten kann, muss diese Ansicht Funktionen für das Verwalten der Schichten und Tageszeiten beinhalten. Dazu wurden 2 Komponenten (Schichten- & Zeitenbearbeitung) erstellt, welche in dieser Ansicht montiert werden.

Die Schichten werden in mehreren Karten dargestellt, wobei eine zusätzliche Karte das Erstellen einer neuen Schicht ermöglicht. Bestehende Schichten können bearbeitet oder gelöscht werden, wobei das Löschen durch Bestätigen einer kleinen Warnung geschieht. Dies soll verhindern, dass ein Benutzer die Schicht und die dazugehörigen Einsätze ungewollt aus dem Plan entfernt.

[**TODO**: Bild der Schichten]

Zum Erstellen und Bearbeiten einer Schicht gehört neben einigen textuellen Eingaben auch das Auswählen einer Farbe, mit welcher die Schicht später im Einsatzplan dargestellt wird. Um dies für den Benutzer bestmöglich darzustellen, entschied ich mich nach kurzer Suche das Modul «vue-color» einzubinden. Es bietet eine kleine Auswahl an zusätzlichen Formular-Elementen, durch die eine Farbauswahl getroffen werden kann.

Tageszeiten werden im Plan in einer bestimmten Reihenfolge dargestellt, welche sich auch anpassen lassen sollte. Deshalb wurde für die Verwaltung der Tageszeiten eine Liste implementiert, die das Verschieben von Listeneinträgen möglich macht. Auch hier wurde ein zusätzlicher Listeneintrag zum Hinzufügen einer neuen Tageszeit erstellt. Die Titel der existierenden Tageszeiten lassen sich in einem kleinen Formular bearbeiten und das Löschen geschieht per Knopfdruck. Wenn eine Änderung vorgenommen wurde wird der «Speichern» Button aktiv, welcher vom Benutzer zur Bestätigung der Änderung betätigt werden muss. Erst dann werden die Änderungen verarbeitet.

[**TODO**: Bild der Tageszeiten]

Der Ablauf einer Anfrage zum Bearbeiten einer Schicht und/oder einer Tageszeit folgt dem gleichen Schema/Use-Case wie das Speichern von Einstellungen. Einzig der Login-Prozess wird nicht benötigt und die veränderten Daten werden im globalen Speicher abgelegt. Entsprechend werden alle Komponenten, welche auf eine Schicht oder Tageszeit zugreifen, automatisch aktualisiert.

[**TODO**: ACT-Diagramm]

#### Anwendung der Firmenstandards

Mit dem Formular zum Erstellen und Bearbeiten einer Schicht wurden sogleich die Firmenstandards exemplarisch umgesetzt. Dafür wurden die Textfelder und deren Label, sowie der Button zum Absenden des Formulars, an die Darstellung nach Firmenstandard angepasst.

Die Umsetzung benötigte viele, individuelle Design-Definitionen, weshalb die Komponenten aus der Vuetify-Bibliothek nicht mehr genutzt werden konnten. Entsprechend musste auch eine angepasste Validierungs-Funktion zur Prüfung der Formularinhalte implementiert werden.

[**TODO**: Bild der Elemente]

### Dienstplan

Der komplexeste Teil der Anwendung war der Dienstplan. Während bisher vordefinierte Komponenten von Vuetify genutzt werden konnten, musste hier ein individuelles, an den Einsatzplan vom Helpdesk angepasstes Design erstellt werden. Dies bedeutete eine eigenständige Umsetzung aller grundlegenden Funktionen (dynamische Aktualisierung, Zwischenspeicherung) und der Tabelle, welche für den Einsatzplan angezeigt werden soll. Wie bei den Einstellungen bestehen auch hier Funktionen, die abhängig von den Berechtigungen des angemeldeten Benutzers angezeigt werden sollen. Der Einsatzplan selbst sowie Details zu den Einsätzen sollen jedoch für alle Benutzer einsehbar sein.

[**TODO**: Screenshot Dienstplan]

Die Komplexität des Dienstplans kostete einen höheren Zeitaufwand bei der Realisierung, weil dazu eine Tabelle benötigt wird, welche sich auf mehrere Komponenten aufteilt. Dabei konnte die Struktur der Kopfzeile im Plan-Container nicht auf die Benutzer-Zeilen übernommen werden und musste dort eigenständig definiert werden. Nach einigen Anläufen hat sich folgende Komponenten-Struktur als optimalste Umsetzung herausgestellt:

[**TODO**: Struktur irgendwie klar machen]

Es entstanden insgesamt 3 Komponenten, welche zusammen den Dienstplan bilden. Die Implementierung stellte sich dabei als äusserst komplex heraus, da zahlreiche Daten an Unterkomponenten vererbt werden müssen, damit sie richtig dargestellt werden können.

Diese Vererbungen stellen aber ein weiteres Risiko dar, denn je nach Anzahl von Datensätzen kann der Aufbau der Komponenten viele Rechenressourcen fordern, was die Zeit bis zur Anzeige verlängert. Ein kurzes Rechenbeispiel anhand der Teamgrösse des Helpdesks kann die Problematik veranschaulichen:

|  |  |  |
| --- | --- | --- |
| X = | Anzahl Teammitglieder | = 30 |
| Y = | Anzahl verschiedener Tageszeiten | = 3 |
| Z = | Anzahl darzustellender Tage | = 25 |
| T = | Millisekunden zum Aufbauen eines einzelnen Feldes in einer Benutzer-Zeile | = 1.8 |

T \* Z \* Y \* X = Millisekunden bis zur Anzeige des ganzen Plans

1.8 \* 25 \* 3 \* 30 = *4050 Millisekunden* = 4.05 Sekunden

Es wird klar, dass hier eine optimale Umsetzung notwendig ist, um möglichst wenig Zeit bis zur Anzeige zu verbrauchen. Das schlechteste Resultat, welches bei der Entwicklung je erzielt wurde, war eine Ladezeit von 6 Sekunden. Dabei wurden die Einsätze von jedem Mitglied vor der Anzeige der Benutzer-Zeile abgerufen. Erst anschliessend wurde die Benutzer-Zeile mit den Einsätzen abgebildet, wobei die Einsätze nicht global, sondern direkt in der Komponente registriert wurden.

Dies wurde mit der dynamischen, globalen Speicherung von Einsätzen verbessert (Vuex). Der globale Speicher an sich ist aber nicht mehr dynamisch, sobald die Vue-Instanz registriert wurde. Zum Hinzufügen dynamischer Objekte muss deshalb eine spezielle Funktion verwendet werden.

[**TODO**: Funktion im Code zeigen]

Nun wird dem globalen Speicher für jeden Benutzer ein Objekt zugewiesen, das die dynamische Speicherung der Einsätze zulässt. Nach dieser und weiteren kleineren Verbesserungen liegt die durchschnittliche Ladezeit bei ungefähr 2 Sekunden, während bereits zuvor eine Ladeanimation angezeigt wird. Damit sollte die Benutzerfreundlichkeit auch bei diesem sehr komplexen Element ausreichend gewährleistet sein.

[**TODO**: Interaktionsdiagramm]

## Testprotokoll

* + 1. FT-001 – Einsätze darstellen

### FT-002 – Einsätze erstellen, bearbeiten und löschen

### FT-003 – Sprache anpassen

### FT-004 – Schichten erstellen, bearbeiten und löschen

### FT-005 – Zeiten erstellen, bearbeiten und löschen

* + 1. FT-006 – Berechtigungen anpassen

## Sicherheitsprüfung

[**TODO: Anmerkung wegen SQL Injections**]

* + 1. A3 – Stored

Aus DB

### A3 – Reflected XSS

Nach bearbeitungen in UI

### A3 – Dom-Based XSS

Von url aus

### A7 – Fehlerhafte Autorisierung auf Anwendungsebene

### A8 – Cross-Site-Request-Forgery (CSRF)

### A9 – Nutzung von Komponenten mit bekannten Schwachstellen

### A10 – Ungeprüfte Um- und Weiterleitungen

# Abschluss

## Auswertung des Kanban-Boards

[**TODO** Issue-Verlauf während der IPA irgendwie grafisch darstellen]

## Funktionen & Verbesserungen für die Zukunft

## Danksagung

[**TODO**: Auch die erwähnen, die korrigiert haben.]

# Anhang

## Quellenverzeichnis

[**TODO**: Erwähnen wie alles korrigiert wurde.]

## Abbildungsverzeichnis

[**TODO**: Bilder/Diagramme jeweils kurz erklären! (Standardkriterium Teil2: #9)]

[**TODO**: Alle Bilder beschriften]

1. Apachefriends.org [↑](#footnote-ref-1)
2. Wikipedia.org – Atom (Texteditor) [↑](#footnote-ref-2)
3. Netlify.com [↑](#footnote-ref-3)
4. Jwt.io [↑](#footnote-ref-4)
5. Webstyleguide.baloise.com [↑](#footnote-ref-5)
6. Brandportal.baloise.com – Corporate Design Manual [↑](#footnote-ref-6)
7. Webstyleguide.baloise.com [↑](#footnote-ref-7)
8. Wikipedia.org – Vue.js [↑](#footnote-ref-8)
9. Wikipedia.org – Kanban (Softwareentwicklung) [↑](#footnote-ref-9)
10. Baloise.ch - Datenschutzbestimmungen [↑](#footnote-ref-10)
11. Owasp.org – Top 10 für Entwickler 2013 [↑](#footnote-ref-11)
12. Wikipedia.org – JSON Web Token [↑](#footnote-ref-12)
13. Airlock.com [↑](#footnote-ref-13)
14. Vuetifyjs.com [↑](#footnote-ref-14)
15. Developers.google.com – Tools for Web Developers – Lighthouse [↑](#footnote-ref-15)