

Personalplanung für den Helpdesk

Individuelle Praktische Arbeit 2019

Projektdaten

Kunde	Basler Versicherungen, Corporate IT (Helpdesk)
Auftraggeber	Matthias Cullmann
Autor	Elia Reutlinger
Ort	Basler Versicherungen, Aeschengraben 21, 4051 Basel
Zeitraum	24.04. – 10.05.2019
Version	1.0
Status	Fertiggestellt

Beteiligte Personen

Anwender	Corporate IT Helpdesk	
Prüfung	Matthias Cullmann	<i>Verantwortliche Fachkraft</i>
	Beat Sommer	<i>Hauptexperte</i>
	Felix Tobler	<i>Nebenexperte</i>
Genehmigung	Tschan Günter	<i>Validierungsexperte</i>

Glossar

A

<u>Abhängigkeiten</u>	<i>Plugin, Library, Erweiterung, (Zusatz-) Modul</i> <i>Erweiterungen zu einer bereits bestehenden Auswahl von vorgefertigten Methoden, Funktionen und/oder Komponenten.</i>
<u>API</u>	<i>Application Programming Interface</i> <i>Schnittstelle, welche von einem Softwaresystem anderen Programmen zur Verfügung gestellt wird, damit sich diese an das System anbinden können. Im Fall dieser Doku wird die API vom Front-End zum Abrufen und Bearbeiten von Daten genutzt.</i>

B

<u>Back-End</u>	<i>Backend, (API)</i> <i>Zur Schichtentrennung einer Anwendung, wobei mit Back-End meist vom Bereich der Datenverarbeitung/Logik und Speicherung auf der Datenbank gesprochen wird. Im Fall dieser Doku besteht das Back-End aus einer API und Datenbank.</i>
-----------------	--

C

<u>CSRF</u>	<i>Cross-Site-Request-Forgery</i> <i>Angriffsmethode, bei welcher ein Angreifer ein an einer Anwendung angemeldetes Opfer zur Ausführung einer Aktion führt, indem dem Opfer unbewusst Schadcode untergeschoben wird.</i>
<u>CSS</u>	<i>Cascading Style Sheets</i> <i>Stylesheet-Sprache für Webseiten zur Gestaltung von (HTML/DOM) Elementen.</i>

F

<u>Framework</u>	<i>Rahmenstruktur</i> <i>Programmiergerüst (meist auch eine Art "Library"), das einen Rahmen zur Verfügung stellt, in welchem ein Programmierer eine Anwendung erstellt. Oftmals wird auch die Struktur der Anwendung beeinflusst.</i>
------------------	---

Front-End

<u>Front-End</u>	<i>Zur Schichtentrennung einer Anwendung, wobei mit Front-End meist das UI gemeint ist. Im Fall dieser Doku besteht das Front-End aus der Vue.js Anwendung.</i>
------------------	---

G

Git

Software zur verteilten Versionsverwaltung von Dateien.

H

HTML

Hypertext Markup Language

Auszeichnungssprache zur Strukturierung von Webseiten und deren Inhalte/Elemente.

I

Issue

GitHub-Issue

Eine Aufgabe (Task) oder ein Problem. In Rahmen der IPA ein Aufwand, welcher auf einem Kanban-Board dargestellt werden kann.

J

JSON-Web-Token

JWT

Auf JSON basierender, verifizierbarer Access-Token, welcher sich zum Austausch der Identität eines Benutzers über verschiedene Schnittstellen eignet.

K

Kanban

Projektmethode in der Softwareentwicklung mit Wurzeln in der japanischen Autoindustrie.

Kanban-Board

Kanban-Tafel, Kanban, Project-Board

Ein Kanban-Board ist das übliche Werkzeug zur Umsetzung der Kanban-Projektmethode. Das Board besteht aus unterschiedlichen Abschnitten, welche Zustände von Aufwänden beschreiben sollen. Die Aufwände wandern bei der Umsetzung durch die Abschnitte, bis sie abgeschlossen sind.

L

Library

Bibliothek

Siehe "Abhängigkeiten"

M

Makro

Excel-Makros

Zusammengefasste Folge von Anweisungen oder Deklarationen, welche durch einen einfachen Aufruf ausgeführt werden können. Kann in Excel-Tabellen für zusätzliche Funktionen genutzt werden. Zusammengefasste Folge von Anweisungen oder Deklarationen, welche durch einen einfachen Aufruf ausgeführt werden können. Kann in Excel-Tabellen für zusätzliche Funktionen genutzt werden.

Mapping-Router

Route-Mapping, Router, Vuex

Modul/Tool zum Registrieren unterschiedlicher Ansichten einer Webanwendung, wobei oftmals auch Aufrufe von Seiten bzw. Pfaden einer Webseite über dieses Modul verarbeitet werden.

Material Design

Von Google Inc. entwickelte Designrichtlinien bzw. ein Design-Standard, welcher Angaben zur Umsetzung bestimmter Elemente eines UI definiert. Basierend auf dem Flat-Design Gestaltungsstil.

P

PHP

Hypertext-Preprocessor

Skriptsprache mit einer an C und Perl angelehnten Syntax. Im Rahmen dieser IPA im Back-End verwendet.

Plugin

Erweiterung, (Zusatz-) Modul

Siehe "Abhängigkeiten"

Project-Board

Kanban-Board, Kanban

Kanban-Board zu einem Repository bzw. dessen Issues auf GitHub.

Details siehe "Kanban-Board".

R

Repository

Depot/Lager

Verwaltetes Verzeichnis zur Speicherung von unterschiedlichen Daten. Im Rahmen der IPA ist damit die Dokumentenablage auf GitHub unter der Verwendung von Git gemeint.

Route-Mapping

Routing

Das Registrieren/Definieren von Routen/Ansichten einer Webanwendung.

Details siehe "Mapping-Router".

S

SQL

Structured Query Language

Datenbanksprache zur Definition von Datenstrukturen in Datenbanken sowie zum Abfragen und Bearbeiten von Datenbeständen.

U

UI

User-Interface, Benutzerschnittstelle

Eine Oberfläche bzw. Schnittstelle, mit welcher ein Benutzer mit einer Anwendung in Kontakt tritt und über welche der Benutzer Aktionen ausführen kann.

Use-Case

Anwendungsfall

Beschreibung eines Szenarios, das durch einen Benutzer beim Verfolgen eines bestimmten Ziels eintreten könnte.

V

Vue.js

VueJS, Vuejs, Vue

JavaScript-Framework (Web) zum Erstellen von Single-Page-Webanwendungen.

X

XAMPP

Programmpaket mit zahlreichen Werkzeugen zur Installation und Konfiguration eines Apache-Webservers.

XSS

Cross-Site-Scripting

Beschreibt das unberechtigte Einfügen von Kontexten/Skripten in eine Webanwendung. Wenn die Webanwendung die eingefügten Daten fälschlicherweise als vertrauenswürdig identifiziert, können daraus Angriffe ausgeführt werden bzw. Sicherheitsrisiken für die Anwendung entstehen.

Inhaltsverzeichnis

Personalplanung für den Helpdesk	1
1 Kurzfassung.....	9
1.1 Ausgangssituation	9
1.2 Ziel der Arbeit	9
1.3 Umsetzung.....	9
1.4 Ergebnis.....	9
Teil 1: Ablauf und Umfeld	10
2 Detaillierte Aufgabenstellung.....	10
2.1 Ausgangslage	10
2.2 Detaillierte Aufgabenstellung.....	10
2.3 Tests.....	12
2.4 Mittel und Methoden.....	13
2.5 Vorkenntnisse.....	13
2.6 Vorarbeiten.....	13
2.7 Neue Lerninhalte.....	14
2.8 Arbeiten in den letzten 6 Monaten.....	14
3 Analyse der Aufgabenstellung	15
3.1 Kurzfassung Ist/Soll.....	15
3.2 Messbare Ziele.....	15
3.3 Dokumentation	15
4 Projektmanagement und Planung.....	16
4.1 Organisation	16
4.1.1 Zeitplan & Termine	16
4.1.2 Projektaufbauorganisation	18
4.1.3 Arbeitsplatz.....	19
4.1.4 Datensicherung & Versionierung	19
4.2 Auswertung der Aufgabenstellung.....	20
4.2.1 Wandlung der vorgegebenen Tests	20
4.2.2 Mittel und Methoden	20
4.2.3 Abhängigkeiten zu Vorarbeiten.....	21
5 Firmenstandards	22
5.1 Schriftarten.....	22
5.2 Farben	22
5.3 Logos.....	23
5.4 Formularelemente	23
6 Arbeitsjournal	24
6.1 Tag 1 – 24.04.2019	24
6.2 Tag 2 – 25.04.2019.....	25
6.3 Tag 3 – 26.04.2019	26
6.4 Tag 4 – 30.04.2019	27
6.5 Tag 5 – 02.05.2019.....	28
6.6 Tag 6 – 03.05.2019	29
6.7 Tag 7 – 07.05.2019.....	30
6.8 Tag 8 – 08.05.2019	31
6.9 Tag 9 – 09.05.2019	32
6.10 Tag 10 – 10.05.2019	33

7	Abschlussbericht.....	34
7.1	Ist-Soll	34
7.2	Schwierigkeiten	34
7.3	Fazit zum Projekt.....	35
7.3.1	Entstandene Abweichungen zur Aufgabenstellung	35
7.4	Persönliches Fazit	36
7.5	Schlussreflexion	37
7.5.1	Arbeitsmethodik.....	37
7.5.2	Werkzeuge	37
7.5.3	Umsetzung	37
Teil 2: Dokumentation und Umsetzung.....		38
8	Initialisierung	38
8.1	Analyse Vue.js	38
8.1.1	Konzept.....	39
8.1.2	Vergleich	39
8.2	Vertiefte Studie Ist-Zustand & Umgebung.....	40
8.3	Umsetzung Projektmethode & Vorgehensmodell	42
8.4	Anforderungen	44
8.4.1	Funktional.....	45
8.4.2	Nicht-Funktional.....	45
8.5	Ziele	46
8.5.1	Projekt	46
8.5.2	Persönlich	46
8.6	Informationssicherheit & Datenschutz	46
8.7	Konzepte	47
8.7.1	Testkonzept (Use-Cases)	47
8.7.2	Sicherheitskonzept	50
9	Realisierung.....	54
9.1	Grundlegende Konfiguration	54
9.1.1	Ansichten / Routes.....	55
9.1.2	Mehrsprachigkeit	56
9.1.3	Authentifizierung.....	57
9.2	Umsetzung Design	59
9.3	Umsetzung der Ansichten	60
9.3.1	Support	60
9.3.2	Dashboard	61
9.3.3	Einstellungen	62
9.3.4	Plan-Einstellungen	64
9.3.5	Dienstplan	66
9.4	Testprotokoll.....	70
9.4.1	FT-001 – Einsätze darstellen	70
9.4.2	FT-002 – Einsätze erstellen, bearbeiten und löschen	70
9.4.3	FT-003 – Sprache anpassen.....	71
9.4.4	FT-004 – Schichten erstellen, bearbeiten und löschen	71
9.4.5	FT-005 – Zeiten erstellen, bearbeiten und löschen.....	72
9.5	Sicherheitsprüfung	73
9.5.1	A3 – Stored XSS	73
9.5.2	A3 – Reflected XSS	73
9.5.3	A3 – Dom-Based XSS.....	74
9.5.4	A7 – Fehlerhafte Autorisierung auf Anwendungsebene	74
9.5.5	A8 – Cross-Site-Request-Forgery (CSRF)	75
9.5.6	A9 – Nutzung von Komponenten mit bekannten Schwachstellen	76

9.5.7	A10 – Ungeprüfte Um- und Weiterleitungen	76
10	Abschluss	77
10.1	Auswertung des Kanban-Boards.....	77
10.2	Funktionen & Verbesserungen für die Zukunft	78
10.3	Danksagung.....	79
11	Anhang	80
11.1	Quellenverzeichnis	80
11.2	Abbildungsverzeichnis.....	81

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» und «Auswertung der Aufgabenstellung» beendet, «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.	Kapitel «Realisierung» abgeschlossen (bis auf einige Grafiken).
0.7	09.05.2019	Elia R.	Kapitel «Abschluss» und «Kurzfassung» geschrieben, letzte Grafiken eingefügt, alle Grafiken beschriftet, Abbildungsverzeichnis generiert.
0.8	09.05.2019	Elia R.	Kapitel «Abschlussbericht» erfasst, kleinere Korrekturen (Grammatik, Layout/Formatierung) vorgenommen.
0.9	10.05.2019	Elia R.	Zahlreiche Korrekturen und Verbesserungen + Quellenverzeichnis.
1.0	10.05.2019	Elia R.	Glossar erfasst, Verzeichnisse aktualisiert, Zeitplan und letztes Arbeitsjournal eingefügt.

1 Kurzfassung

1.1 Ausgangssituation

Der Helpdesk der Baloise Group nutzt zur Einsatzplanung seiner 30 Teammitglieder eine Excel-Tabelle, was einige Nachteile mit sich bringt. Das Team hat bereits vor längerem den Wunsch geäussert, diese Planung in einer Webanwendung vornehmen zu können, wodurch Aufwand und damit Zeit gespart werden soll. Das Ganze soll transparenter, dynamischer und unkomplizierter funktionieren als mit der Excel-Tabelle, welche durch unzählige und aufwändige Makros so gut wie möglich an die Bedürfnisse des Helpdesks angepasst wurde.

1.2 Ziel der Arbeit

Dem Helpdesk soll eine Webanwendung zur Verfügung stehen, welche die Verbesserungen mit sich bringt und nach der produktiven Einrichtung auf einem Server direkt genutzt werden kann. Sie soll dem Workflow des Excel-Plans folgen oder ihn weiter verbessern, aber auf jeden Fall eine Erleichterung darstellen. Dabei soll die Anwendung auch gut auf mobilen Endgeräten dargestellt werden können, damit die Benutzer auch unterwegs ihre Einsätze abrufen können.

1.3 Umsetzung

Die Umsetzung erfolgte mit dem JavaScript-Framework «Vue.js» und verlief ohne beachtliche Probleme. Durch die sehr gute Planung und Vorarbeit konnte die Webanwendung ohne weiteres umgesetzt werden, und es wurden keine erwähnenswerten Absprachen, etwa wegen Unklarheiten oder Problemen benötigt. Einzig eine falsche Einschätzung bezüglich der Komplexität des Dienstplans führte zu einem etwas höheren Zeitaufwand für die Implementierung von diesem, was aber ohne Weiteres wieder ausgeglichen werden konnte. Mit der Zielsetzung, dem Endbenutzer das zügige und einfache Verwalten eines Dienstplans zu ermöglichen, wurden zahlreiche Ansichten und Funktionen implementiert. Gleichzeitig wurden die Firmenstandards der Baloise berücksichtigt und auch viel Wert auf ein modernes und anspruchsvolles «Material Design» gelegt.

1.4 Ergebnis

Das Endprodukt dieser Arbeit heisst Pelan, wobei Pelan aus einer API als Back-End Lösung und einem UI im Front-End besteht. Der Auftrag während der IPA bestand aus der Entwicklung des UIs zur bereits bestehenden API. Es entstand eine funktionsfähige, vollständige und produktiv einsetzbare Webanwendung. Sie kann alle Anforderungen der Excel-Tabelle erfüllen, wozu zahlreiche Verbesserungen und Optimierungen gemacht wurden. Zugriffe werden sehr performant verarbeitet und können auch durch ein Smartphone getätigter werden, da die Anwendung auch lückenlos für mobile Endgeräte konzipiert ist.

Zahlreiche abschliessende Tests und eine tiefgreifende Sicherheitsanalyse bestätigen die Einhaltung der Anforderungen an die Anwendung, weswegen sie ohne weiteres auf der Baloise-internen Umgebung aufgesetzt werden kann.

Teil 1: Ablauf und Umfeld

2 Detaillierte Aufgabenstellung

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

2.1 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.

2.2 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 diesem 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.

2.3 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.

2.4 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)

2.5 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.

2.6 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 Kanban-Board eingerichtet, welches voraussichtlich mit der Integration von Waffle.io zustande kommt. Es werden auch bereits die Anforderungen in Form von GitHub-Issues eingetragen.

2.7 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.

2.8 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.

3 Analyse der Aufgabenstellung

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

3.1 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 Mitarbeitenden 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 kann. 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.

3.2 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.

3.3 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.

4 Projektmanagement und Planung

4.1 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.

4.1.1 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

Zeitplan

Elia Reutlinger IPA 2019		Tage	Tag 1 Mittwoch, 24.4.		Tag 2 Donnerstag, 25.4.		Tag 3 Freitag, 26.4.		Tag 4 Dienstag, 30.4.		Tag 5 Donnerstag, 2.5.		Tag 6 Freitag, 3.5.		Tag 7 Dienstag, 7.5.		Tag 8 Mittwoch, 8.5.		Tag 9 Donnerstag, 9.5.		Tag 10 Freitag, 10.5.	
Phase & Tätigkeit		Stunden	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
Initialisierung	Arbeitsjournal & Projekt aufsetzen	Soll Ist	2 1.5																			
Entwicklung		Soll Ist	34 34																			
Grundlegende Konfiguration	Design, Mehrsprachigkeit, Konfigurationen	Soll Ist	4 2.5																			
	Authentifizierung und Mapping/Routing	Soll Ist	2 3.5																			
	Progressive WebApp PWA	Soll Ist	2 1																			
Seiten und Funktionen	Planeinstellungen	Soll Ist	4 3.5																			
	Einstellungen	Soll Ist	2 2.5																			
	Einsatzplan	Soll Ist	12 15																			
	Dashboard	Soll Ist	2 2																			
	Support / Hilfe	Soll Ist	2 1																			
	Fehlermeldungen	Soll Ist	2 1																			
	Abschliessende Optimierungen	Soll Ist	1 1																			
Abschliessende Optimierungen	Ungeplante Arbeiten	Soll Ist	1 1																			
	Dokumentation	Soll Ist	44 43																			
Erster Teil 1.0	Aufgabenstellung & Analyse	Soll Ist	2 1.5																			
	Projektmanagement, Planung & Firmenstandards	Soll Ist	4 3																			
	Auswertung der Aufgabenstellung	Soll Ist	2 1																			
	Uneingeplante Arbeiten	Soll Ist	2 0.5																			
Zweiter Teil	Initialisierung	Soll Ist	8 8																			
	Realisierung	Soll Ist	10 14																			
	Abschluss & Anhänge	Soll Ist	2 2																			
Erster Teil 2.0	Kurzfassung	Soll Ist	2 2																			
	Abschlussbericht	Soll Ist	4 3																			
Abschliessende Optimierungen	Grammatik, Satzbau, Formatierung	Soll Ist	4 4																			
	Uneingeplante Arbeiten	Soll Ist	4 4																			

4.1.2 Projektaufbauorganisation



Abbildung 1 - Organigramm der Projektrollen

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

4.1.3 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.

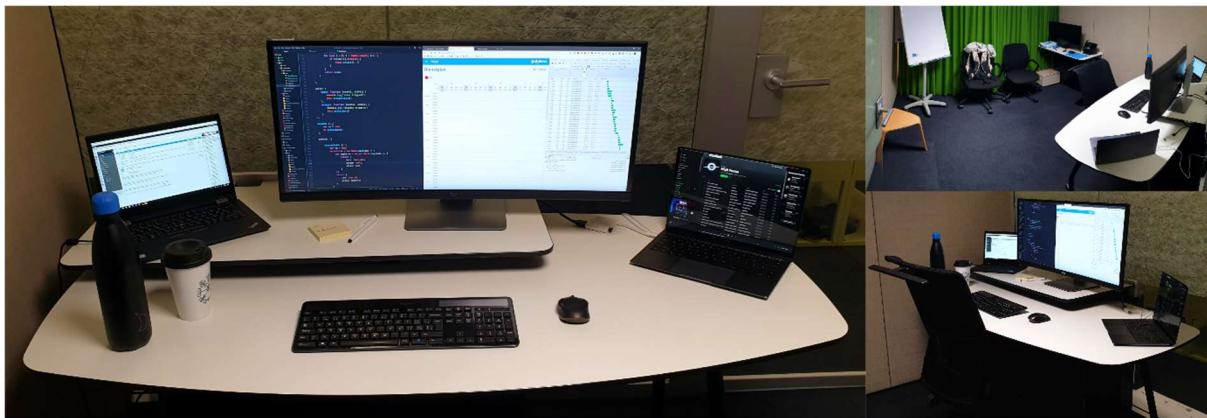


Abbildung 2 - Arbeitsplatz während der IPA

4.1.4 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 wird auf einer externen Festplatte ein tägliches Backup der gesamten Systempartition erstellt. Dies ist zwar laut Erkenntnissen aus der IPA-Vorbesprechung keine direkte Anforderung der IPA, ermöglicht es jedoch, im Notfall schneller zur Arbeit zurückzukehren.

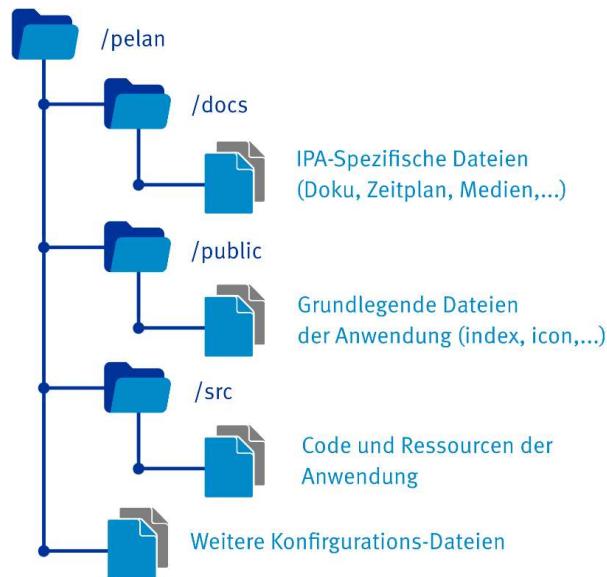


Abbildung 3 - Struktur der IPA-Daten

4.2 Auswertung der Aufgabenstellung

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

4.2.1 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 durch einzelne Testschritte, erwartete Resultate und weitere Details vertieft. Die vordefinierten Tests bilden gleichzeitig die Use-Cases der Anwendung, nach welchen sich die Implementierung richtete.

4.2.2 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

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.² 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. 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 im Internet zu erstellen, was die endgültige Funktion garantieren soll. Alternativ könnte die Anwendung auch lokal im produktiven Modus demonstriert werden.³

4.2.3 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 (JWT⁴) 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.

¹ Apachefriends.org

² Wikipedia.org – Atom (Texteditor)

³ Netlify.com

⁴ Jwt.io

5 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 Webstyleguide⁵ und den Corporate Design Richtlinien⁶ der Baloise. Bei Unklarheiten zu Einzelfällen soll das weitere Vorgehen mit dem Fachvorgesetzten besprochen und entschieden werden.

5.1 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.

5.2 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 #99DoE9 und Grauton #7E7E7E aus. Eine genaue Vorgabe zum Einsatz dieser Farben besteht nicht, wobei die Grautöne möglichst vermieden werden sollen.

⁵ Webstyleguide.baloise.com

⁶ Brandportal.baloise.com – Corporate Design Manual

5.3 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.



Abbildung 4 - Baloise-Group Logo

5.4 Formularelemente 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.



Abbildung 5 - Textfelder nach Firmenstandard

Buttons

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



Abbildung 6 - Buttons nach Firmenstandard

⁷ Webstyleguide.baloise.com

6 Arbeitsjournal

6.1 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.

Ort, Datum	Kandidat	Fachverantwortlicher
25.04.2019		

6.2 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

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.

Datum	Kandidat	Fachverantwortlicher
25.04.2019		

6.3 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 «reakтив» 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 +

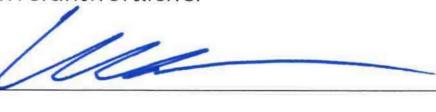
Durch die Vertiefung der Kapitel weiss ich besser, was in der Dokumentations-Phase auf mich zu kommt und kann bereits jetzt einiges notieren, was ich später noch brauche bzw. berücksichtigen sollte.

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.

Datum	Kandidat	Fachverantwortlicher
26.04.2019		

6.4 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 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.

Datum	Kandidat	Fachverantwortlicher
30.04.2019		

6.5 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, vuetyjs.com

Reflexion +

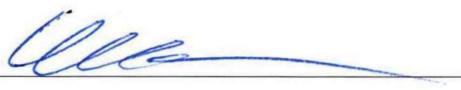
Ich finde es sehr gut, dass ich fast genau nach Plan mit der Implementierung fertig wurde. Obwohl es einiges nachzuholen gab, schaffte ich es, alles zu erledigen und sogar noch viel zu verbessern. Ich konnte auch noch meinen Code «verschönern», indem ich alles formatiert und kommentiert habe. Dabei würde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

Datum	Kandidat	Fachverantwortlicher
02.05.2019		

6.6 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.

Datum
03.05.2019

Kandidat



Fachverantwortlicher



6.7 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.

Datum	Kandidat	Fachverantwortlicher
07.05.2019		

6.8 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	Kapitel «Realisierung» abgeschlossen.	Elia R.	04:00	04:00

Tagesthemen

Heute habe ich wie geplant nur am Kapitel «Realisierung» gearbeitet. Das Tagesziel war die Vervollständigung dieses Kapitels, was ich auch erreicht habe. Das Einzige, was abseits vom Plan verlaufen ist, war eine kleine Anpassung an der App. Dabei habe ich lediglich die Komponente zur Farbauswahl bei den Schichten ersetzt, da mir das Design der Bisherigen nicht sehr zugesagt hat.

Hilfestellungen

Owasp.org (Top 10), Test- und Sicherheitskonzept aus Kapitel «Initialisierung»

Reflexion +

Ich musste heute viel erledigen, da das Kapitel «Realisierung» einige IPA-Kriterien beeinflusst, welche ich so gut wie möglich erfüllen will. Dabei lag die Erläuterung des Gesamtsystems im Vordergrund, was wie erwartet eine gute Übersicht und einen «Roten-Faden» forderte. Ich finde aber, dass ich die Aufgabe sehr gut lösen konnte, da der Text mit den Grafiken eigentlich alle wichtigen Punkte so ausführlich wie nötig erklärt. Die Anpassung der Komponente zur Farbauswahl bei den Schichten hat meiner Meinung nach das Formular um einiges ansehnlicher gemacht, wobei so gut wie kein Zeitaufwand dafür nötig war. Ich finde es auch immer wieder gut, neben der Doku auch ein wenig an der Anwendung arbeiten zu können.

Reflexion –

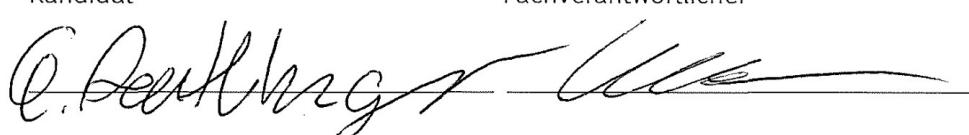
Da ich das Kapitel so gut wie möglich erarbeiten wollte, fehlte mir am Ende die Zeit, um den Text grob auf Grammatik/Rechtschreibung zu prüfen. Mir ist jedoch klar, dass der Text noch verbessert werden muss, weshalb ich bei der abschliessenden Überprüfung dieses Kapitel genauer betrachten sollte.

Erkenntnisse

Da heute hauptsächlich «Macherarbeiten» erledigt werden mussten, entstanden keine grösseren Erkenntnisse. Ich habe jedoch ein Sicherheitsrisiko zu Axios entdeckt, als ich die Sicherheitsprüfung durchgeführt habe. Auch wenn Pelan nicht betroffen ist, kann ich nun in anderen Anwendungen auf dieses Risiko achten.

Datum
08.05.2019

Kandidat



Fachverantwortlicher

6.9 Tag 9 – 09.05.2019

Zeit	Tätigkeit	Beteiligt	H-«Soll»	H-«Ist»
07:30	Kapitel «Abschluss» erfasst.	Elia R.	02:00	02:30
10:00	Kurzfassung geschrieben, letzte Grafiken eingefügt und alle beschriftet, Abbildungsverzeichnis generiert.	Elia R.	02:00	01:30
12:00	Kapitel «Abschlussbericht» erfasst.	Elia R.	04:00	03:00
15:00	Mit abschliessenden Arbeiten/Optimierungen begonnen (Glossar, Korrektur, Formatierung, Kriterien-Check, ...).	Elia R.	00:00	01:00

Tagesthemen

Heute war geplant die Dokumentation so weit abzuschliessen, dass morgen nur noch abschliessende Arbeiten wie Korrigieren, Formatieren und ähnliche gemacht werden müssen. Dazu habe ich das Kapitel «Abschluss» im 2. Teil der Arbeit geschrieben und alle noch fehlenden Abbildungen eingefügt und Beschriftungen erstellt, worauf ich das Abbildungsverzeichnis erstellt habe. Dann konnte ich die Kurzfassung schreiben und das letzte Kapitel «Abschlussbericht» fertigstellen. Da ich etwas früher fertig war als geplant, begann ich schon mit den Aufgaben von morgen.

Hilfestellungen

Mein altes Referat zu Kanban, GitHub-API (zum Abrufen der Issues vom Projekt)

Reflexion +

Da ich heute etwas früher fertig war, konnte ich schon etwas vorarbeiten. Dabei habe ich unter anderem das Layout der gesamten Doku überprüft und leicht verbessert, wodurch ich nun sehr zufrieden mit der Darstellung bin. Auch die Texte, die ich heute erfasst habe, sind aus meiner Sicht sehr gut, da sie einen klaren, abrundenden Abschluss der Doku bilden. Speziell der Vergleich von Pelan zur Excel-Tabelle gefällt mir und lässt sehen, was in dieser IPA geschaffen wurde.

Reflexion –

Beim Erstellen der Statistiken zum Kanban-Board musste ich mich länger damit auseinandersetzen, wie ich an die Daten von GitHub komme und diese anschliessend in einer Excel-Tabelle geeignet darstelle. Ich finde es schade, dass GitHub zum Erstellen von solchen Statistiken noch nichts anbietet.

Erkenntnisse

Wie vor einiger Zeit erwähnt war ich mir nicht sicher, wie ich die Anwendung von Kanban in der Doku nachweisen/beschreiben sollte. Ich erinnerte mich aber an mein altes Referat zu Kanban, worauf ich in der Doku von diesem Referat nach Ideen suchte. Dabei stiess ich auf die Durchsatzstatistiken, welche das ITIL-Services-Team der Baloise zur Analyse ihres Kanban-Boards erstellt. Darin sah ich die bisher beste Art zur Visualisierung von gelösten und neu hinzugekommenen Tasks eines Kanban-Boards, weshalb ich kurzerhand eine solche Statistik zum Kanban der IPA erstellte.

Datum
09.05.2019

Kandidat



Fachverantwortlicher



6.10 Tag 10 – 10.05.2019

Zeit	Tätigkeit	Beteiligt	H-«Soll»	H-«Ist»
07:30	Zahlreiche Verbesserungen zu Grammatik, Satzstellung, Rechtschreibung, Leserlichkeit u.Ä. vorgenommen.	Elia R.	04:00	04:00
12:00	Glossar erfasst, Zeitplan eingefügt, weitere Details verbessert bis zur Fertigstellung der Dokumentation.	Elia R.	04:00	04:00

Tagesthemen

Zu Beginn des letzten Tages der IPA habe ich die soweit fertiggestellte Doku noch im Detail auf mögliche Optimierungen geprüft und diese auch gleich umgesetzt. Dabei haben mir auch die Ratschläge geholfen, welche ich durch meine Familie erhalten habe. Sie nahmen sich gestern Abend noch kurzfristig Zeit um das Dokument zu lesen, wobei die Verständlichkeit im Fokus lag. Das Feedback von ihnen hat mir sehr geholfen, da es aus der Perspektive von Personen ist, die überhaupt nichts mit Informatik zu tun haben. Am Nachmittag habe ich dann noch das Glossar sowie den Zeitplan eingefügt, wodurch die Arbeit eigentlich abgeschlossen war. Zur endgültigen Fertigstellung habe ich noch geringfügige Verbesserungen am Layout und weiteren Details vorgenommen.

Hilfestellungen

Meine Familie (Tipps zum Verständnis), einige Wikipedia.org Beiträge für das Glossar

Reflexion +

Der heutige Tag ist soweit ohne Zwischenfälle oder ähnliches verlaufen, wodurch ich mich problemlos auf Details konzentrieren konnte. Beim abschliessenden Durchlesen der gesamten Doku muss ich sagen, dass ich grundsätzlich sehr zufrieden mit der Arbeit bin, wobei mir auch das Layout gefällt.

Reflexion –

Mir ist heute eigentlich nichts Negatives aufgefallen was eine Reflexion benötigten würde. Ich bin froh, dass die Arbeit soweit abgeschlossen ist und hoffe, dass die Abgabe auch so problemlos verläuft.

Erkenntnisse

Es entstanden heute auch keine erwähnenswerten Erkenntnisse.

Datum
10.05.2019

Kandidat



Fachverantwortlicher



7 Abschlussbericht

7.1 Ist-Soll

Da der Helpdesk einen vermeidbar hohen Aufwand bei der Einsatzplanung der Mitarbeitenden betreibt, war das Ziel der Arbeit, dem Helpdesk eine bessere Alternative zur Einsatzplanung per Excel zu bieten. Das Resultat der Arbeit ist eine dynamische, produktionsfähige Webanwendung für die Einsatzplanung eines Teams.



Abbildung 7 - Vergleich der Dienstpläne als Tabelle und in Pelan

Abschliessend kann man also sagen, dass das Ziel durchaus erreicht wurde. Der Helpdesk kann sich nun für die neue Webanwendung entscheiden und diese nutzen, sobald sie intern aufgesetzt wurde. Dabei kann die Webanwendung die gleichen Anforderungen erfüllen wie das Excel, wobei sie dynamischer und einfacher zu handhaben ist.

7.2 Schwierigkeiten

Ungeplante Schwierigkeiten können während der Entwicklung einer Anwendung durchaus auftreten. Dabei sind die meisten nicht von grosser Bedeutung und stören den Arbeitsablauf kaum. Während der Implementierung der Anwendung kam es ebenfalls an gewissen Stellen zu Abweichungen von der geplanten Umsetzungsmethode, wobei die meisten zeitlich nur einen geringen Einfluss hatten.

Die grösste Schwierigkeit ergab sich bei der Umsetzung des eigentlichen Dienstplans. Mir war zwar bewusst, dass dieser eigenständig umgesetzt werden muss, jedoch war meine Einschätzung des Aufgabenumfangs nicht richtig. Ich musste hier mehr Zeit investieren als geplant, da mir die Komplexität der Anforderung erst nach Beginn der Entwicklung klar wurde. Die dynamische Speicherung der Einsätze im globalen Speicher wurde komplexer als erwartet und es schien mir, als wenn mit jedem Fortschritt an einer Stelle, an einer anderen neuen Aufgaben entstanden, die zur Funktion der zuvor implementierten benötigt wurden.

Insgesamt dauerte die Implementierung des Dienstplans länger als im Zeitplan geschätzt, weswegen ich mich bei den anschliessenden Aufgaben beeilen sollte. Die Umsetzung dieser dauerte jedoch kürzer als geplant, wobei ich Sie optimal umsetzen konnte und es schliesslich schaffte, die Entwicklungsphase nach Zeitplan zu beenden.

Gravierende Schwierigkeiten gab es glücklicherweise nicht, was nicht zuletzt an der guten Planung und meiner unabhängigen Arbeitsmethode lag. Ich kann mir vorstellen, dass mehr Probleme entstanden wären, wenn die Entwicklung mit internen Abhängigkeiten durchgeführt worden wäre.

7.3 Fazit zum Projekt

Die Umsetzung einer solchen Anwendung in einer solch kleinen Zeitspanne ist an sich bereits eine Herausforderung, die ohne konkrete Planung und gute Vorkenntnisse nicht bewältigt werden könnte. Eine Abweichung von einem eingeschätzten Zeitaufwand könnte hier gravierende Folgen haben, welche die Fertigstellung des Projekts gefährden könnten. Während der Entwicklung führte ein ungeplanter Mehraufwand zu einem Zeitverlust, der glücklicherweise ausgeglichen werden konnte. Schlussendlich wurde erfolgreich ein Endprodukt erstellt, welches aktuellen Standards entspricht und bereit für die produktive Nutzung ist. Dies konnte durch abschliessende Tests überprüft und bestätigt werden.

7.3.1 Entstandene Abweichungen zur Aufgabenstellung

Bereits vor Beginn des Durchführungsblocks kam es zu unvorhersehbaren Abweichungen, welche die im Voraus definierte, detaillierte Aufgabenstellung veränderten. Da aber keine Abweichungen entstanden, die die eigentlichen Anforderungen blockierten, waren keine zusätzlichen Fachgespräche notwendig.

Struktur der Benutzerverwaltung

Hier wurden alle vordefinierten Anforderungen übernommen, wobei sogleich eine optionale Erweiterung hinzugezogen wurde. Die Teamleiter können nicht nur die Rollen der Benutzer, sondern gleichzeitig auch einige weitere Details wie den Namen und Spitznamen anpassen. Dies wurde so entschieden, da der Aufwand schlussendlich genauso gross war, und die Funktion irgendwann wahrscheinlich ohnehin angefragt worden wäre.

Struktur der Anwendung

Hier wurden grundsätzlich auch alle angeforderten Ansichten implementiert, wobei die Bearbeitung der Tageszeiten und Schichten in einer gemeinsamen Ansicht umgesetzt wurde. Die Login-Ansicht, welche zu Demozwecken implementiert werden sollte, wurde komplett gestrichen, da sie einerseits in Berücksichtigung der Medusa-SSO Infrastruktur der Baloise überhaupt keinen Sinn macht, und andererseits zur Demonstration eher im Back-end implementiert werden soll.

Dazu entschieden wir uns, weil die Einhaltung der Gütestufe 3 des Bewertungskriteriums «Brauchbarkeit (Applikation)» eigentlich gar keine Login-Ansicht zulassen würde.

Darüber hinaus wurde eine weitere optionale Ansicht implementiert. Mit der «Support»-Ansicht soll dem Benutzer ein schneller Zugang zu wichtigen Kontaktdaten ermöglicht werden. Dazu entschieden wir uns, um die Benutzerfreundlichkeit der Anwendung weiter zu steigern, wobei kein erwähnenswerter Mehraufwand entstanden ist.

Mittel und Methoden

Durch eine unvorhersehbare Mitteilung des Kanban-Dienstes «Waffle.io» über die Einstellung ihrer Dienste mussten wir für das Kanban-Board eine Alternative finden. Schlussendlich entschieden wir uns für ein einfaches GitHub-Project, welches fast die gleichen Kompetenzen erfüllt.

7.4 Persönliches Fazit

Nach der Umsetzung der Anwendung bleiben mir einige wichtige Eindrücke und Erkenntnisse. Der kurze Zeitraum, der für die Programmierung geplant war, hat meine gesamte Konzentration und höchste Produktionsfähigkeit benötigt, um die Anwendung termingerecht zu realisieren. Es hätten zusätzliche Details behandelt werden können, wenn ich hier mehr Zeit gehabt hätte. Ich gelangte auch kurz in eine Stresssituation bei der Implementierung des Dienstplans. Durch diese Erfahrung weiss ich nun, dass eine genaue Planung/Einschätzung der Anforderung entscheidend für den Erfolg eines Projekts sein kann.

Letztendlich bin ich sehr zufrieden mit der Anwendung. Sie deckt die Anforderungen, wurde professionell umgesetzt und kann dem Helpdesk durchaus helfen, indem Sie die Planung einfacher gestaltet als eine Excel-Tabelle. Die Umsetzung mit Vue.js hat mir sehr zugesagt, da ich dieses Framework zur Realisierung von Webanwendungen bevorzuge. Ich habe darauf geachtet, dass möglichst viel modularisiert wird und der Code übersichtlich und verständlich bleibt, wodurch auch nachfolgende Entwickler problemlos mit dem Code arbeiten können.

7.5 Schlussreflexion

7.5.1 Arbeitsmethodik

Während der Entwicklung nutzte ich Kanban zur Visualisierung meines Fortschritts. Ich habe Kanban bereits zuvor bei anderen Projekten genutzt und kenne auch detaillierte Funktionen, da ich vor einiger Zeit ein Referat zu dieser Methodik gehalten habe. Meiner Meinung nach ist Kanban eine gute Wahl in der Softwareentwicklung, da das System unkompliziert und übersichtlich ist. Das hat sich für mich auch während der IPA herausgestellt, weshalb ich in Zukunft bei Kanban bleiben würde. Ich finde, dass Kanban im Vergleich zu anderen Methoden wie z.B. Scrum eine dynamischere und selbstständigere Arbeitsweise zulässt, wobei auch zeitliche Abweichungen und andere Probleme schnell sichtbar werden.

7.5.2 Werkzeuge

Mit den eingesetzten Sprachen, Programmen und Diensten bin ich grundsätzlich zufrieden. Mir war wichtig, dass ich mir hier keine Grenzen, sondern klare Strukturen und wenn möglich sogar Abhilfe schaffe. Dieses Ziel habe ich auch erreicht, wobei ich aber Verbesserungspotential sehe: Der Texteditor Atom bietet zwar viele Möglichkeiten, lässt sich aber nur schwer für Debugging u.Ä. verwenden. Hier könnte ich auf ein fortgeschritteneres Programm setzen. Ich bemerkte auch, dass Netlify zur Demonstration der Anwendung nicht notwendig wäre, da ein lokaler Produktions-Build dieselben Kriterien erfüllt. Bei meinem nächsten Projekt würde ich womöglich auf diesen Dienst verzichten.

7.5.3 Umsetzung

Ich denke, dass die eingesetzten Abhängigkeiten und die allgemeine Umsetzung der Ansichten sehr gut gewählt ist und den «Best-Practices» entspricht. Hier würde ich eigentlich nichts anders machen und auch beim nächsten Projekt auf diese Methoden setzen. Einzig die Abfragen der Einsätze im Dienstplan konnte nicht ganz optimal umgesetzt werden, da dafür der Endpunkt auf der API fehlt. Ich würde die Einsätze lieber gesammelt mit einer Abfrage anstatt mit mehreren Abfragen für jeden einzelnen Benutzer durchführen. Da mir dies bei der Vorarbeit jedoch nicht in den Sinn kam, fehlt aktuell der entsprechende Endpunkt und die Abfragen mussten einzeln implementiert werden.

Mit der Performance der Anwendung bin ich sehr zufrieden. Mir ist die Ladezeit einer Anwendung sowie die Dauer zur Verarbeitung von Daten sehr wichtig, da das für mich einen ausschlaggebenden Punkt zur Benutzerfreundlichkeit darstellt. Meiner Meinung nach sollten solche Geschwindigkeiten heutzutage bei allen Anwendungen Standard sein, was leider nicht immer der Fall ist. Ich denke, dass ich die Performance von Pelan sogar noch etwas erhöhen könnte, wenn ich serverseitiges Rendern einsetzen würde. Dies kann für eine Vue.js Anwendung beispielsweise durch das erweiterte «Nuxt.js» Framework realisiert werden, was aber zusätzlichen Aufwand und Komplexität bedeutet. Deshalb passte es leider nicht in den Rahmen dieser IPA und wurde für Pelan nicht genutzt.

Teil 2: Dokumentation und Umsetzung

8 Initialisierung

8.1 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

- 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 wird die Implementierung einfacher gemacht, wodurch man sich stärker auf die wichtigen Punkte konzentrieren kann. Schlussendlich soll also ermöglicht werden, durch Vue.js eine Anwendung performant, stabil und schnell umzusetzen.

⁸ Wikipedia.org – Vue.js

8.1.1 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.

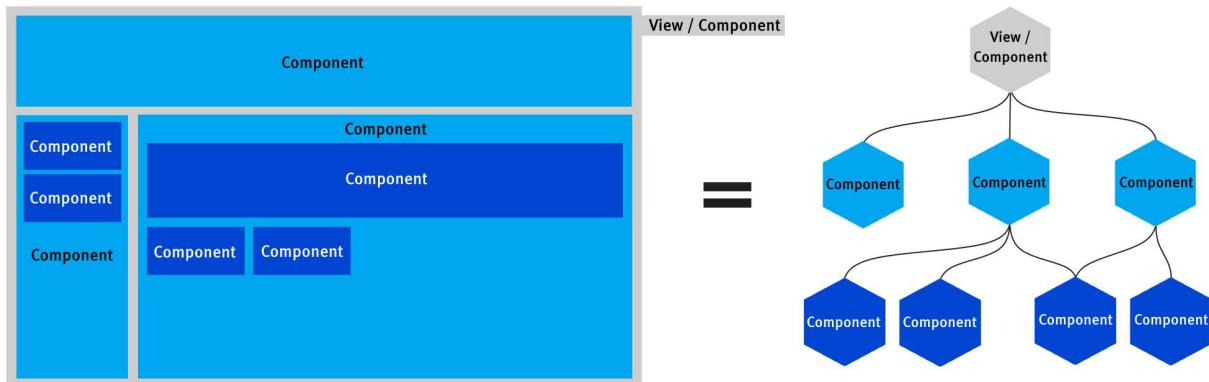


Abbildung 8 - Vue.js Komponenten Konzept

Mit der Erweiterung «Vuex» entsteht zudem eine Art globaler Speicher (der sogenannte «Store»), auf welchen alle Komponenten zugreifen können. Dadurch wird 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.

8.1.2 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.

8.2 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.

Abbildung 9 - Dienstplan als Excel-Tabelle

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.

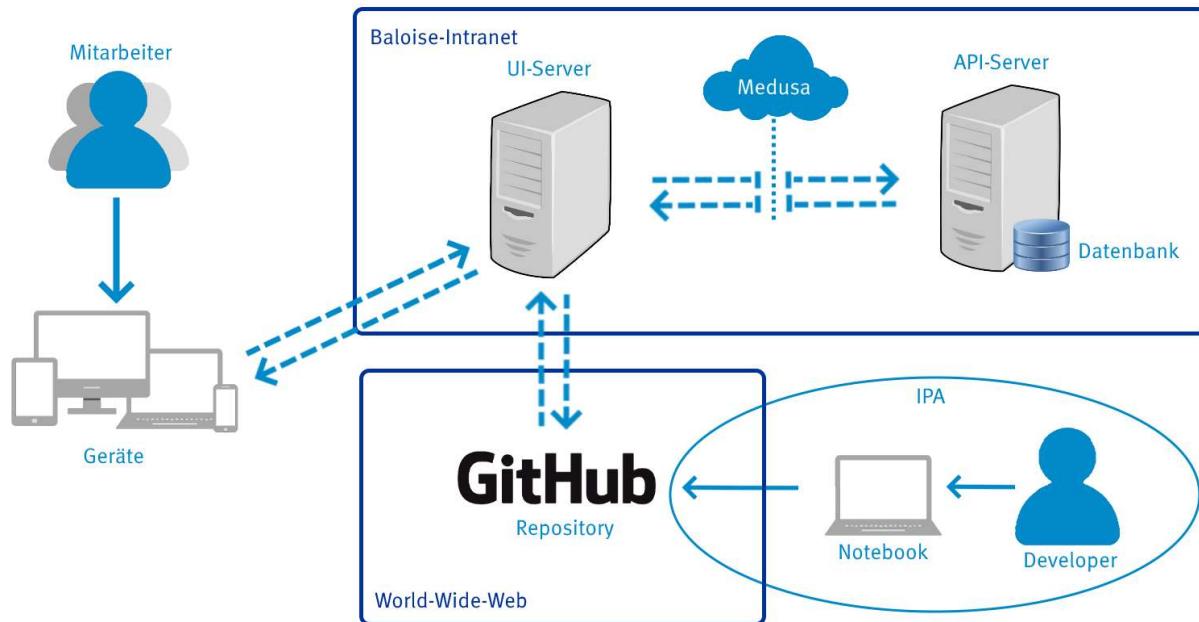


Abbildung 10 - Umgebung und Schnittstellen zum Projekt

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 oder Bearbeiten gewisser Daten beim Benutzer 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. Das Back-End generiert bei erfolgreicher Authentisierung zur Validierung weiterer Anfragen einen Token, welcher im Front-End nur noch verarbeitet/eingebunden werden muss.

8.3 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.

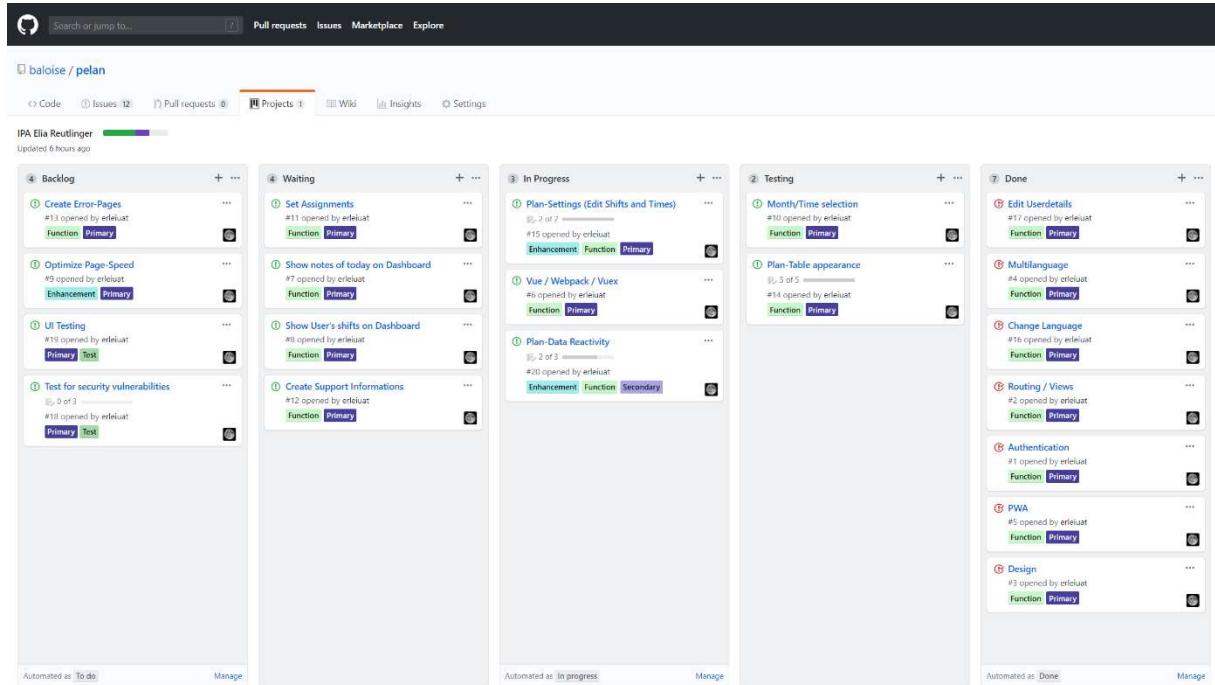


Abbildung 11 - Kanban-/Project-Board auf GitHub

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

⁹ Wikipedia.org – Kanban (Softwareentwicklung)

8.4 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, wodurch die gesamten Anforderungen ersichtlich werden.

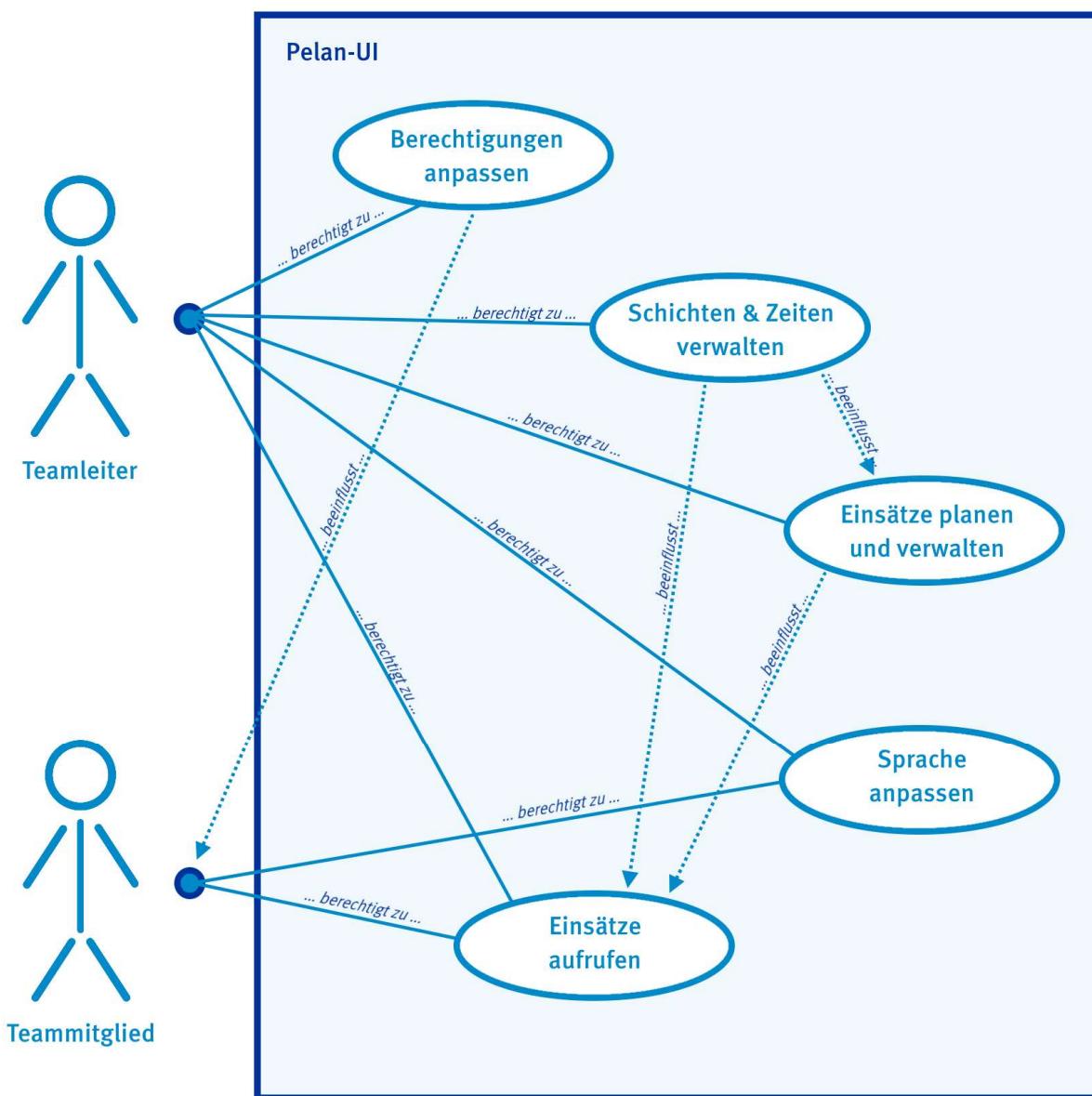


Abbildung 12 - Use-Case-Diagramm über alle Anforderungen

8.4.1 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

8.4.2 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

8.5 Ziele

8.5.1 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.

8.5.2 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.

8.6 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

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.

¹⁰ Baloise.ch - Datenschutzbestimmungen

8.7 Konzepte

8.7.1 Testkonzept (Use-Cases)

Einführung

Die folgenden Testfälle sollen überprüfen, ob die Anforderungen an die 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-Uls 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.

8.7.1.1 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

8.7.1.2 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.

8.7.1.3 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.

8.7.1.4 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.

8.7.1.5 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).

8.7.1.6 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.

8.7.2 Sicherheitskonzept

Dieses Konzept orientiert sich an den «OWASP Top 10 Risiken für Anwendungssicherheit 2013».^{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, bei der sich einzig die Reihenfolge der Risiken geändert hat.

¹¹ Owasp.org – Top 10 für Entwickler 2013

8.7.2.1 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 Verteidigungsmechanismen anwenden.

Überprüfung

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

8.7.2.2 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 der 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 Verteidigungsmechanismen anwenden.

Überprüfung

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

8.7.2.3 A7 – Fehlerhafte Autorisierung auf Anwendungsebene

Grundlegende Erläuterung

Wenn die Anzeige von bestimmten Ansichten oder Funktionen abhängig von einem Autorisierungsmechanismus 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?

8.7.2.4 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, die 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?

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

8.7.2.6 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?

9 Realisierung

9.1 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:

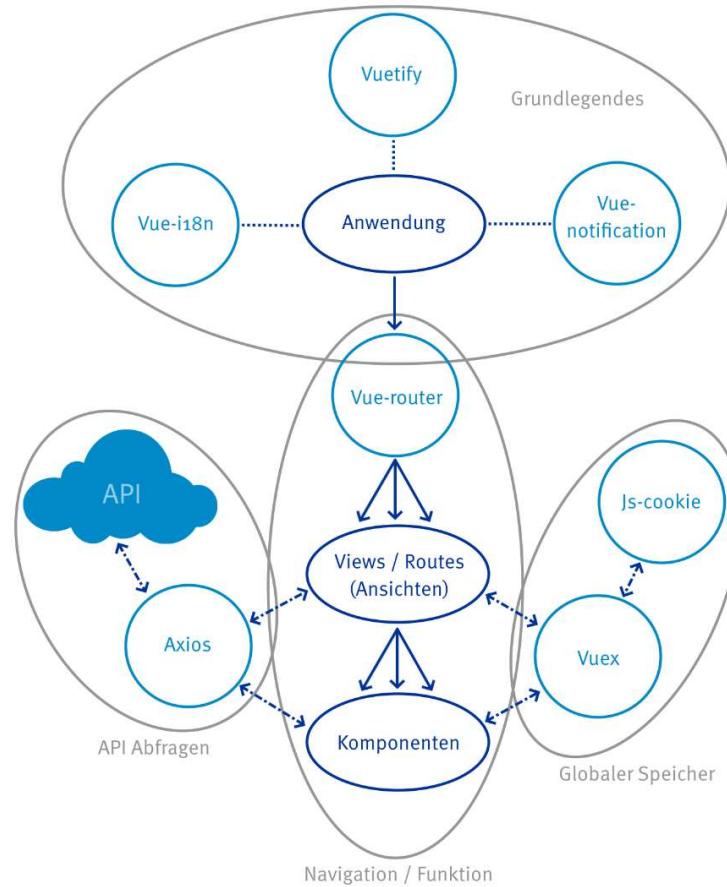


Abbildung 13 - Abhängigkeiten in Relation

9.1.1 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 Dienstplan.
- **Plan bearbeiten** (/plansettings): Einstellungen zu Schichten und Tageszeiten (Teamleiter).
- **Support** (/help): Kurze Sammlung wichtiger Kontaktinformationen.
- **Einstellungen** (/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.

```
path: '/plan',
name: 'plan',
component: () => import('./views/Secure/Plan.vue'),
meta: {
  requiresAuth: true,
  requiresAdmin: false
}
```

Abbildung 14 - Codebeispiel: Registrieren einer Route/Ansicht

9.1.2 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.

```
i18n: {
  messages: {
    en: {
      good: {= "You don't have any assignments today."},
      notes: 'Notes of the day',
      shifts: "Today's assignments"
    },
    de: {
      good: {= "Du hast heute keine Notizen."},
      notes: 'Notizen des Tages',
      shifts: 'Heutige Einsätze'
    }
  }
}
```

Abbildung 15 - Codebeispiel: i18n Übersetzungen

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.

```
vm.$i18n.locale = vm.$store.state.user.language
vm.$store.watch((state) => {
  return vm.$store.state.user.language
}, (newValue, oldValue) => {
  if (newValue !== oldValue) {
    if (vm.$store.state.user.language) vm.$i18n.locale = vm.$store.state.user.language
  }
})
```

Abbildung 16 - Codebeispiel: "Watcher"-Funktion zum Anpassen der Sprache

9.1.3 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 Um ihn zu erhalten sendet die Anwendung eine Login-Anfrage an die API. Die API authentifiziert den Benutzer (was produktiv per Airlock-Medusa¹³ Single-Sign-On geschieht), erstellt einen Token und sendet diesen zurück.

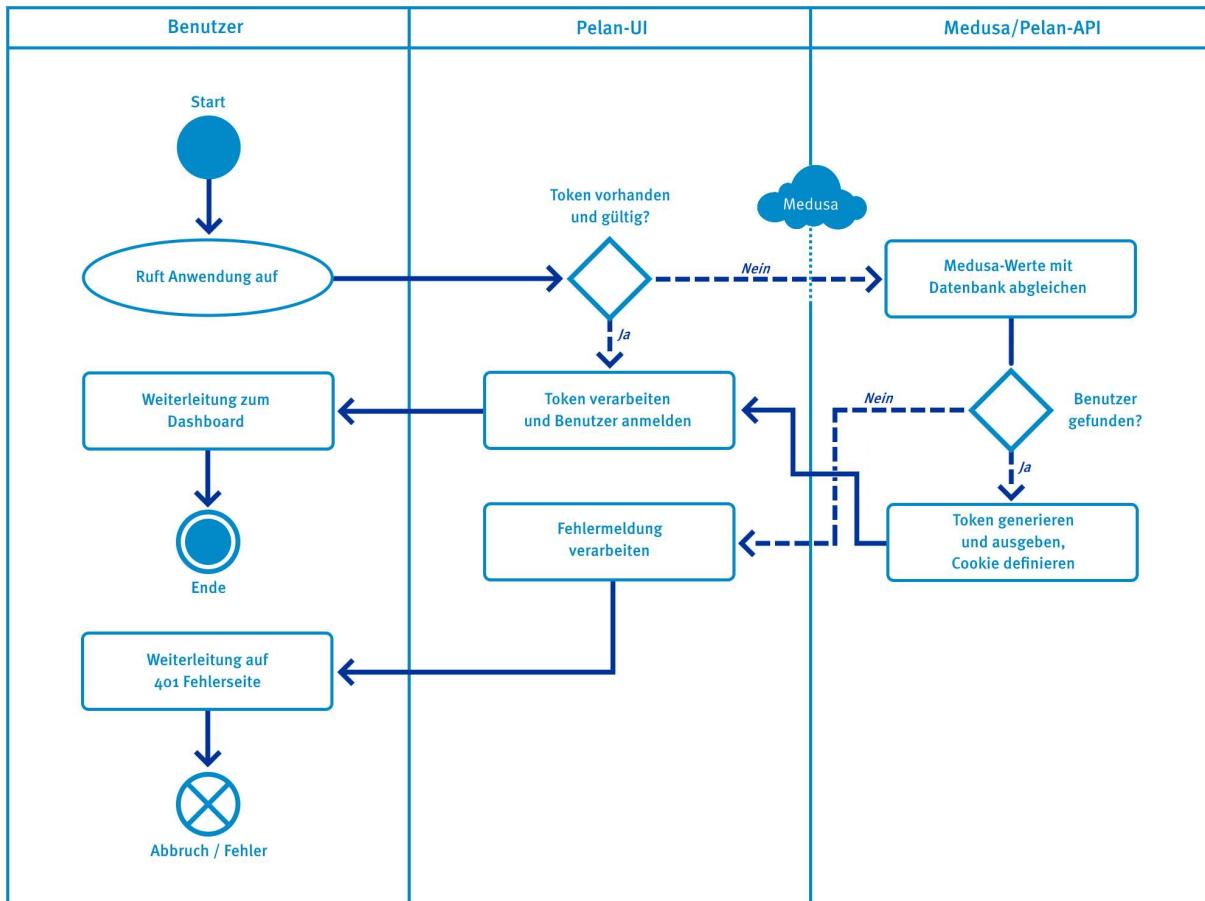


Abbildung 17 - Sequenzdiagramm zur Authentifizierung

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:

¹² Wikipedia.org – JSON Web Token

¹³ Airlock.com

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

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

```
if (to.meta.requiresAuth === true && !vm.$store.state.auth.token) {  
    vm.getLogin(function (done) {  
        if (done && vm.checkPerms(to.meta)) next()  
        else vm.$router.push({ name: 'nopermission' })  
    })  
} else if (vm.checkPerms(to.meta)) next()  
else vm.$router.push({ name: 'nopermission' })
```

Abbildung 18 - Codebeispiel: Authentifizierung Schritt 1

2. 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.

```
vm.$http.get('user/login/').then(function (response) {  
    vm.$store.commit('login')  
    callback(true)  
}).catch(function () {  
    vm.$notify({ type: 'error', text: vm.$t('alert.authFail') })  
    callback(false)  
})
```

Abbildung 19 - Codebeispiel: Authentifizierung Schritt 2

3. 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.

```
if (route.requiresAdmin === true && !vm.$store.state.user.role.admin) return false  
if (!vm.$http.defaults.headers.common['Authorization'] && vm.$store.state.auth.token) {  
    vm.$http.defaults.headers.common['Authorization'] = 'Bearer ' + vm.$store.state.auth.token  
}  
return true
```

Abbildung 20 - Codebeispiel: Authentifizierung Schritt 3

9.2 Umsetzung Design

Die Umsetzung des Designs geschah unter Berücksichtigung der Firmenstandards mit Vuetify¹⁴. 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 kann. Zusätzlich wurden globale CSS-Anpassungen vorgenommen, welche die Schriftarten importieren und die Schriftgrößen für unterschiedliche Elemente wie beispielsweise Titel, Überschrift und Text festlegen.

```
Vue.use(Vuetify, {  
  lang: { ... },  
  theme: {  
    primary: '#008AC9',  
    secondary: '#EFF9FE',  
    accent: '#003399',  
    success: '#2DB200',  
    error: '#FF3366',  
    info: '#008AC9',  
    warning: '#FF9900',  
    black: '#444444'  
  }  
})
```

Abbildung 21 - Codebeispiel: Farben in Vuetify-Konfiguration

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.



Abbildung 22 - Design-Struktur der Anwendung

¹⁴ Vuetifyjs.com

9.3 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 über das Gesamtsystem im Fokus liegt.

9.3.1 Support

Die Ansichten, die am wenigsten Aufwand in der Umsetzung forderten, sind das Dashboard und die Support-Seite. Hier befinden sich schliesslich keine komplexen 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.

Support

Wenn du Fragen oder Probleme hast, kontaktiere bitte eine der folgenden Adressen.

Baloise Helpdesk

Basler Versicherungen
Aeschengraben 21
4002 Basel

helpdesk@baloise.ch
[+41 58 285 77 77](tel:+41582857777)

Elia Reutlinger

Elia Reutlinger
Schwarzwaldallee 33
4058 Basel

mail@eliareutlinger.ch

Abbildung 23 - Support-Ansicht mit Kontaktinformationen

9.3.2 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 Dienstplan 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.

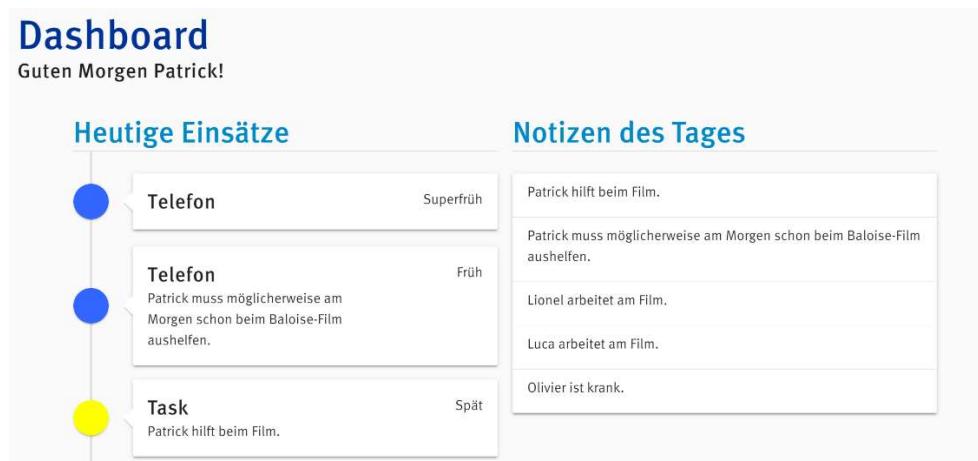


Abbildung 24 - Dashboard-Ansicht

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 mit 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 Dienstplan und den Plan-Einstellungen benötigt werden könnten.

```
if (!vm.$store.state.app.shifts.length) {
    vm.$http.get('shift/read/').then(function (response) {
        if (response.data.content) vm.$store.state.app.shifts = response.data.content
    })
}

if (!vm.$store.state.app.times.length) {
    vm.$http.get('daytime/read/').then(function (response) {
        if (response.data.content) vm.$store.state.app.times = response.data.content
    })
}
```

Abbildung 25 - Codebeispiel: Abrufen von Schichten und Tageszeiten

Es wurde abschliessend überprüft, ob die Geschwindigkeit der Dashboard-Seite akzeptabel ist und als Startseite in Frage kommt. Dies wurde mit dem Lighthouse-Audit Tool¹⁵ von Google erledigt, was ein gängiges Tool zur Auswertung der Akzeptanz von Webanwendungen ist.



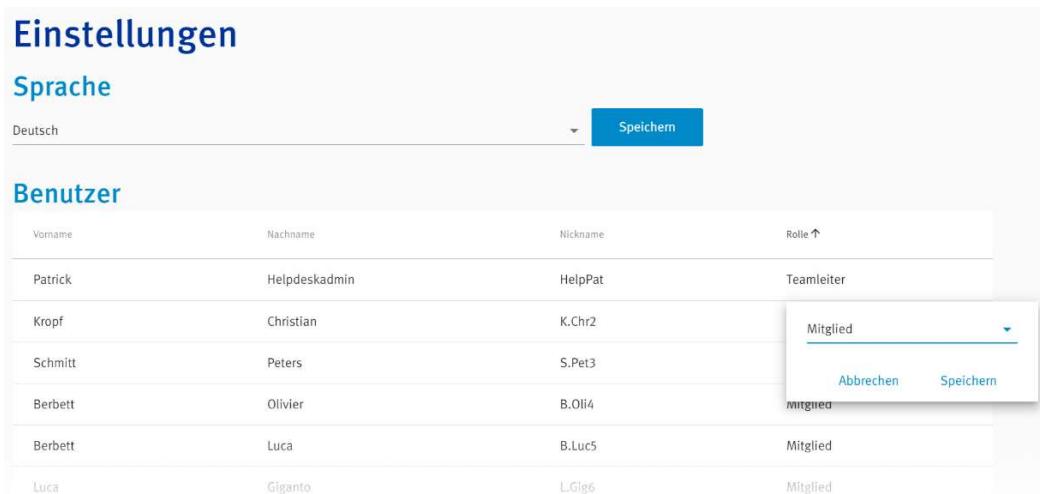
Abbildung 26 - Performance-Test durch Lighthouse-Audit

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.

9.3.3 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.



The screenshot shows the 'Einstellungen' (Settings) page. It includes sections for 'Sprache' (Language) set to 'Deutsch' with a 'Speichern' (Save) button, and 'Benutzer' (Users). The 'Benutzer' table lists users with their roles:

Vorname	Nachname	Nickname	Rolle ↑
Patrick	Helpdeskadmin	HelpPat	Teamleiter
Kropf	Christian	K.Chr2	Mitglied
Schmitt	Peters	S.Pet3	Mitglied
Berbett	Olivier	B.Oli4	Mitglied
Berbett	Luca	B.Luc5	Mitglied
Luca	Giganto	L.Gig6	Mitglied

A dropdown menu for 'Rolle' is open over the 'Mitglied' row, showing options 'Mitglied' (selected), 'Abbrechen', and 'Speichern'.

Abbildung 27 - Einstellungen-Ansicht (als Teamleiter)

¹⁵ Developers.google.com – Tools for Web Developers – Lighthouse

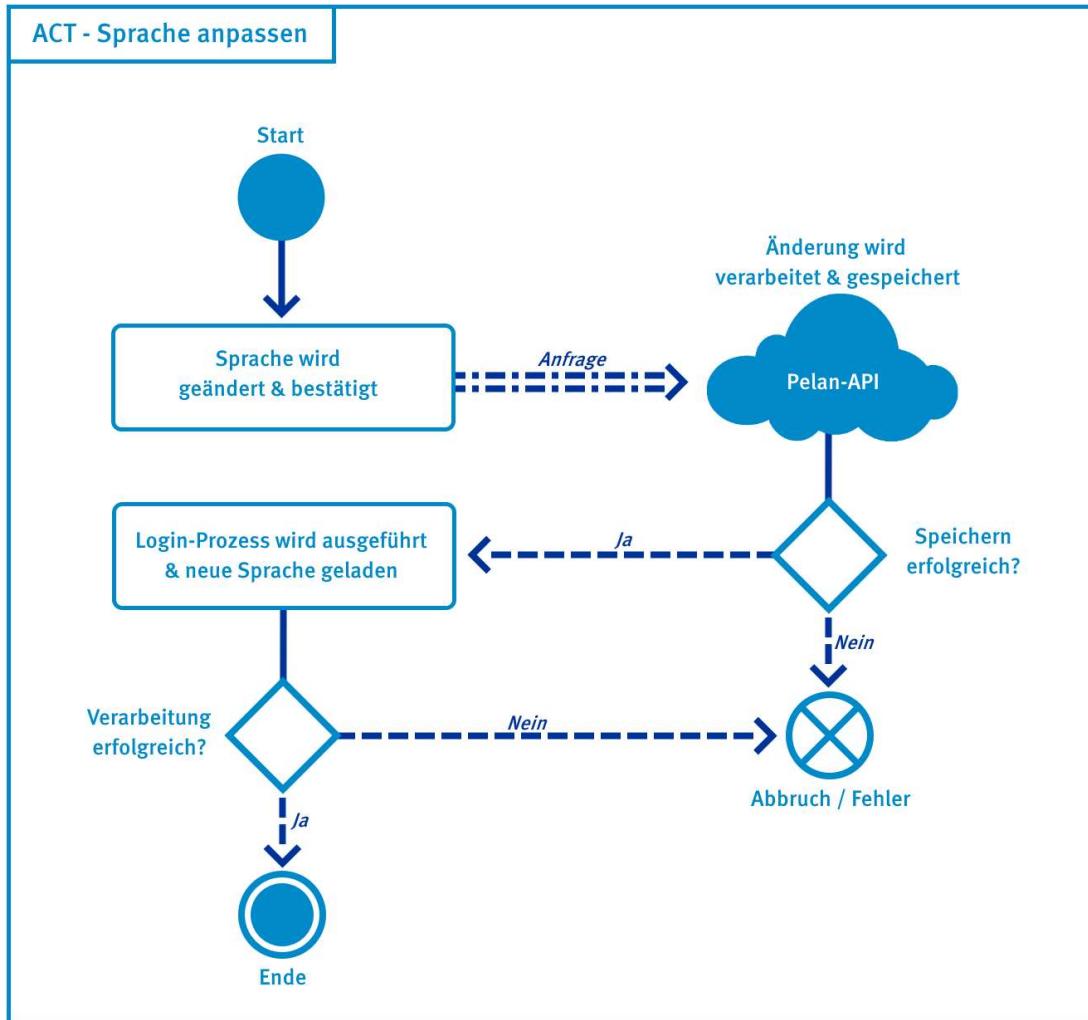


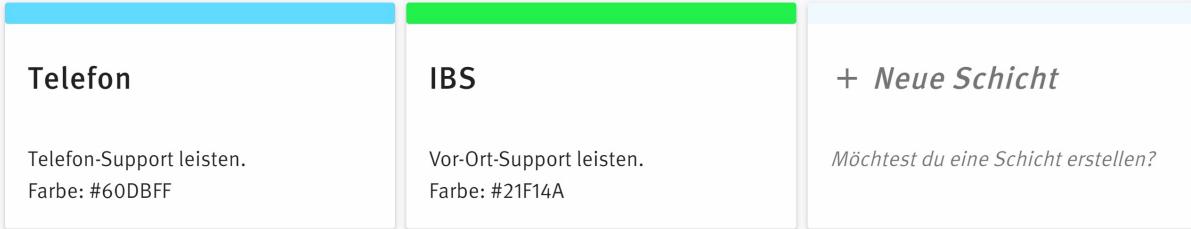
Abbildung 28 - Aktivitätsdiagramm "Sprache anpassen"

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.

9.3.4 Plan-Einstellungen

Damit ein Teamleiter alle möglichen Parameter zum Dienstplan 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.



The screenshot shows three cards side-by-side:

- Telefon**: Telefon-Support leisten. Farbe: #60DBFF
- IBS**: Vor-Ort-Support leisten. Farbe: #21F14A
- + Neue Schicht**: Möchtest du eine Schicht erstellen?

Abbildung 29 - Darstellung der Schichten in den Plan-Einstellungen

Zum Erstellen und Bearbeiten einer Schicht gehört neben einigen textuellen Eingaben auch das Auswählen einer Farbe, mit welcher die Schicht später im Dienstplan 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.



The screenshot shows a list of times with edit and delete icons:

- 1 Superfrüh
- 2 Spät
- 3 Früh

A new entry input field at the bottom is labeled "Titel" and has a "+" icon to its right.

Achtung: Das Löschen einer Zeit entfernt sie und alle ihr zugeordneten Einsätze aus dem Einsatzplan. Dies kann nicht rückgängig gemacht werden.

Abbildung 30 - Darstellung der Tageszeiten in den Plan-Einstellungen

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.

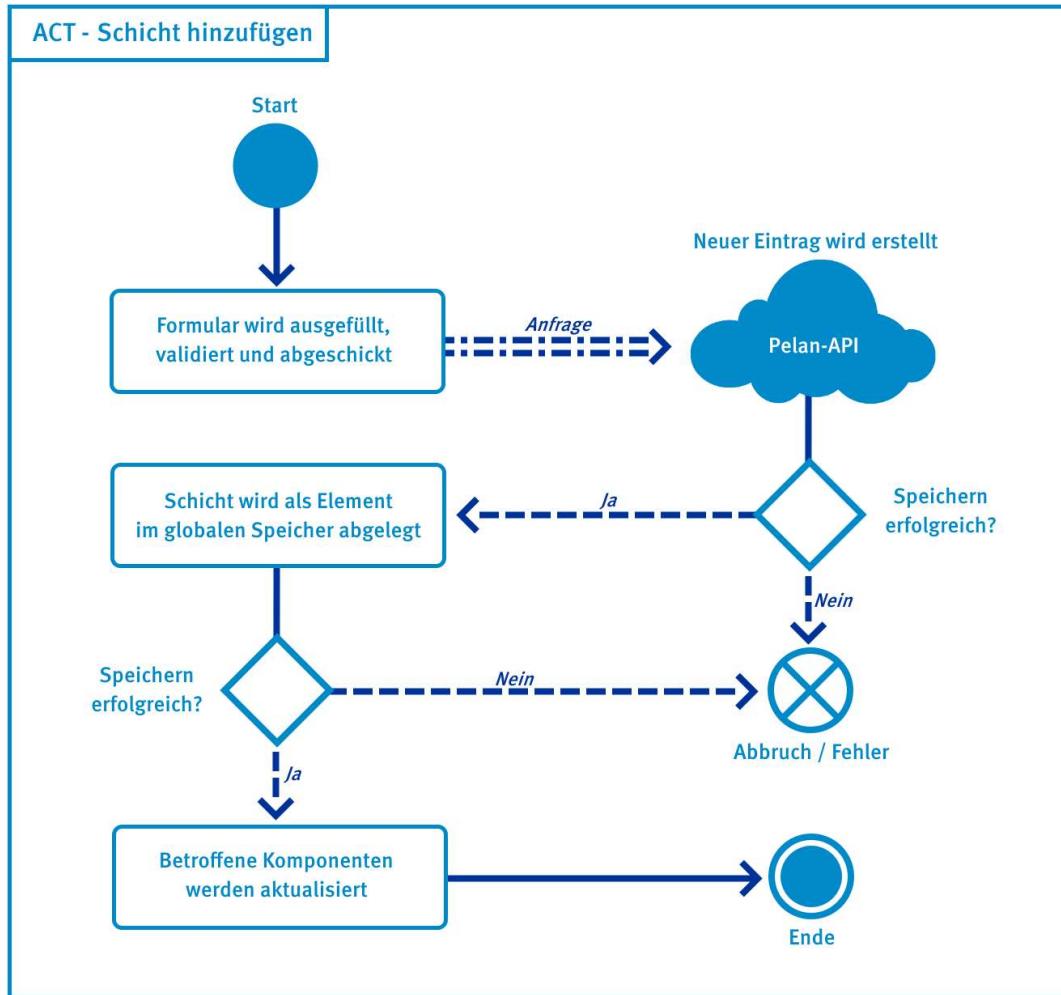


Abbildung 31 - Aktivitätsdiagramm "Schicht hinzufügen"

9.3.4.1 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.

Komponenten nach Firmenstandard		Die Umsetzung benötigte viele, individuelle Design-Definitionen, weshalb die Komponenten aus der Vuetify-Bibliothek nicht mehr genutzt werden konnten. Entsprechend musste auch eine angepasste Validierungsfunktion zur Prüfung der Formularinhalte implementiert werden.
Textfelder		
Titel	<input type="text" value="Demo"/>	
Beschreibung	<input type="text"/>	
Button	<input type="button" value="Hinzufügen"/>	

Abbildung 32 - Komponenten/Design nach Firmenstandard

9.3.5 Dienstplan

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

Dienstplan												Mai 2019					
Woche	19			20			21			22			Mo	Di	Mi	Do	Fr
	Tag	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do					
HelpPat	Super																
	Frlh																
	Spät																
K.Chr2	Super																
	Frlh																
	Spät																
S.Per3	Super																
	Frlh																
	Spät																
B.Olia	Super																
	Frlh																
	Spät																
B.LucG	Super																
	Frlh																
	Spät																

Abbildung 33 - Dienstplan-Ansicht

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



Abbildung 34 - Aufbau/Struktur der Komponenten im Dienstplan

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

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

$$\begin{aligned} T * Z * Y * X &= \text{Millisekunden bis zur Anzeige des ganzen Plans} \\ 1.8 * 25 * 3 * 30 &= \underline{\underline{4050 \text{ Millisekunden} = 4.05 \text{ Sekunden}}} \end{aligned}$$

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 (durch Vuex) verbessert. 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.

```
users.forEach(function (user) {
  vm.$store.state.app.assigns = Object.assign({}, 
    vm.$store.state.app.assigns, { [user.id]: [] })
})
})
```

Abbildung 35 - Codebeispiel: `Object.Assign()` ermöglicht dynamisches Zuweisen neuer Objekte

Damit 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.

Durch die hohe Komplexität der Beziehungen zwischen den unterschiedlichen Komponenten kann die Übersicht leicht verloren gehen. Eine detaillierte, textuelle Aufklärung zu allen Elementen würde jedoch über die Anforderungen hinaus gehen und auch Punkte beschreiben, welche nicht zum allgemeinen Verständnis notwendig wären. Deshalb folgt hier eine Grafik, welche die Einordnung der Komponenten, deren wichtigste Daten und Funktionen, die Beziehungen untereinander sowie die Nutzung von grundlegenden Teilen der Anwendung beschreiben soll. Die wesentlichen, zur Funktion des Plans elementaren Komponenten werden anschliessend noch detaillierter erläutert.

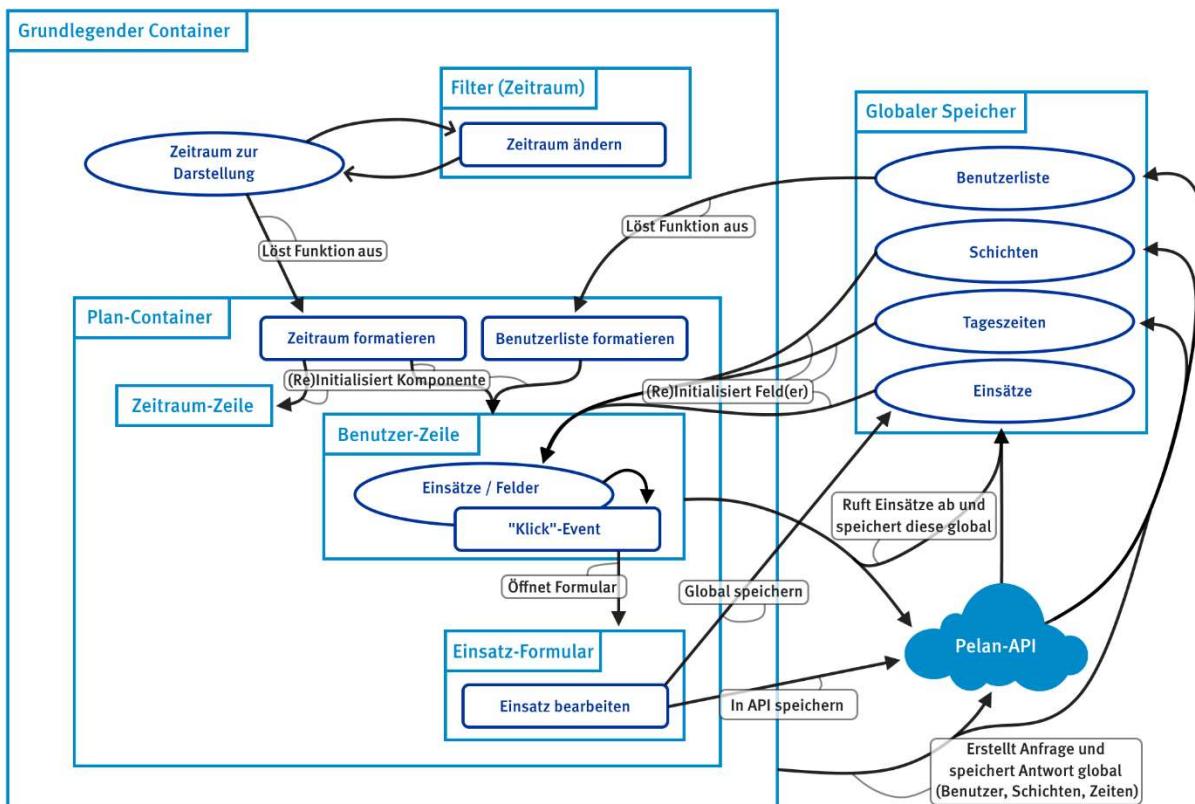


Abbildung 36 - Interaktionsübersichtsdiagramm zu den Elementen im Dienstplan



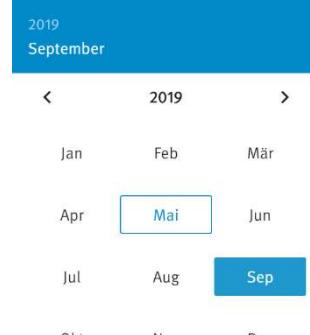
Abbildung 37 - Legende zum Diagramm

9.3.5.1 Grundlegender Container

Hier ist der Umfassende Rahmen (die View) gemeint, in welchem der Dienstplan dargestellt wird. Dazu gehören eine Überschrift und ein Filter, der zur Navigation des Plans benötigt wird. Über diesen Filter kann ein Benutzer den darzustellenden Monat auswählen. Zusätzlich kann die Anzeige von Wochenenden dynamisch ein- und ausgeschaltet werden.

```
<DateSelection v-model="filter"/>
<PlanBase :startDate="filter.start" :endDate="filter.end" />
```

Abbildung 38 - Codebeispiel: Vererbung der Auswahl zum Zeitraum



Wochenenden anzeigen

Abbildung 39 - Formular zur Auswahl des Zeitraums

9.3.5.2 Plan-Container

Der Plan-Container ist ein weiterer Rahmen, der die beiden Teile der Plan-Tabelle strukturiert. Die verfügbaren Schichten werden ebenfalls hier als Legende über der Tabelle dargestellt. Der erste Teil der Tabelle ist eine Kopfzeile, welche die dargestellten Wochen und Tage beschreibt. Diese werden, abhängig von der Auswahl im grundlegenden Container, gleich hier generiert und angezeigt.

Dabei werden die Schichten, Tageszeiten und Benutzer von der API abgefragt, worauf für jeden Benutzer die «Benutzer-Zeile» initialisiert wird. Gleichzeitig wird dem globalen Speicher ein neues Objekt zugewiesen, in welchem die Einsätze des Benutzers abgelegt werden können.

```
<UserRow v-for="u in userList" :key="u.id" :usr="u" :prc="uow" @assign="doEdit" />
```

Abbildung 40 - Codebeispiel: Benutzer-Zeile als Komponente

Des Weiteren befindet sich auf dieser Ebene auch das Formular zur Bearbeitung von Einsätzen. Es ist jedoch zunächst nicht sichtbar, da es erst durch eine «Benutzer-Zeile» ausgelöst werden muss. Die «Benutzer-Zeile» gibt dabei Informationen zum betroffenen Einsatz mit, welche dann im Formular dargestellt und bearbeitet werden können.

9.3.5.3 Benutzer-Zeile(n)

Diese Komponente wird für jedes Teammitglied generiert und abgebildet. Durch die vererbten Informationen zum Benutzer und dem ausgewählten Zeitraum werden die zugehörigen Einsätze von der API abgefragt. Die erhaltenen Daten werden dann im globalen Speicher abgelegt, worauf eine «Beobachter»-Funktion ausgelöst wird und die Anzeige der Einsätze durchführt.

```
watch: {
    assigns: function (newVal, oldVal) {
        this.createEntries()
    }
}
```

Abbildung 41 - Codebeispiel: "Watcher"-Funktion auf Änderungen in den Einsätzen

Durch Anklicken eines Einsatzes wird ein Event ausgelöst, welches das Bearbeitungs-Formular öffnet, wobei Informationen zum Einsatz mitgegeben werden. Da die Einsätze dynamisch im globalen Speicher abgelegt wurden, aktualisiert sich die Komponente bei einer Änderung automatisch.

9.4 Testprotokoll

Zum Schluss der Implementierungsphase wurden die vorbestimmten Funktionen/Use-Cases nach dem Testkonzept auf ihre Funktionalität überprüft. Zur Erfüllung der Anforderungen müssen alle Tests positive Ergebnisse liefern. Die Tests wurden mit den im Konzept definierten Voraussetzungen durchgeführt, wobei keine zusätzlichen Hilfsmittel eingesetzt wurden.

Name	Tester	Datum	Status
FT-001	Elia Reutlinger	08.05.2019	Erfolgreich
FT-002	Elia Reutlinger	08.05.2019	Erfolgreich
FT-003	Elia Reutlinger	08.05.2019	Erfolgreich
FT-004	Elia Reutlinger	08.05.2019	Erfolgreich
FT-005	Elia Reutlinger	08.05.2019	Erfolgreich
FT-006	Elia Reutlinger	08.05.2019	Erfolgreich

9.4.1 FT-001 – Einsätze darstellen

Name	FT-001
Tester / Umgebung	Elia Reutlinger / Nach Konzept
Datum	08.05.2019
Erwartetes Resultat	Einsatz erscheint im Plan.
Effektives Resultat	Einsatz ist im Plan erschienen.
Teststatus	Erfolgreich, ohne Bemerkungen.

Ablauf

- + Neuen Einsatz nach FT-002 definiert und Entwicklungsumgebung neu gestartet.
1. Anwendung geöffnet und auf Plan-Ansicht navigiert.

9.4.2 FT-002 – Einsätze erstellen, bearbeiten und löschen

Name	FT-002
Tester / Umgebung	Elia Reutlinger / Nach Konzept
Datum	08.05.2019
Erwartete Resultate	1. Einsatz erscheint, 2. Änderung erscheint, 3. Einsatz erscheint nicht mehr
Effektives Resultat	Alle erwarteten Resultate sind eingetroffen.
Teststatus	Erfolgreich, ohne Bemerkungen.

Ablauf

1. Dienstplan geöffnet.
2. Leeren/ungeplanten Einsatz angeklickt.
3. Schicht in Formular ausgewählt und gespeichert (1. Erwartetes Resultat eingetroffen).
4. Erstellten Einsatz angeklickt.
5. Andere Schicht in Formular ausgewählt und gespeichert (2. Erwartetes Resultat eingetroffen).
6. Erstellten Einsatz angeklickt.
7. «Löschen» Taste angeklickt (3. Erwartetes Resultat eingetroffen).

9.4.3 FT-003 – Sprache anpassen

Name	FT-003
Tester / Umgebung	Elia Reutlinger / Nach Konzept
Datum	08.05.2019
Erwartetes Resultat	Die Sprache der Oberfläche ändert sich.
Effektives Resultat	Die Sprache der Oberfläche hat sich geändert.
Teststatus	Erfolgreich

Ablauf

1. Einstellungen geöffnet.
2. Die Sprache von Deutsch auf Englisch geändert und gespeichert.
3. Übersetzungen zusätzlich auf anderen Seiten überprüft.

Bemerkungen

Es entstanden keine Probleme. Zu beachten ist jedoch, dass von Benutzern erstellte Texte wie die Namen der Schichten und Tageszeiten nur in einer Sprache verfügbar sind. Hier ist der Benutzer für eine allgemein klare Namensgebung verantwortlich, weshalb im Test diese Texte nicht berücksichtigt werden.

9.4.4 FT-004 – Schichten erstellen, bearbeiten und löschen

Name	FT-004
Tester / Umgebung	Elia Reutlinger / Nach Konzept
Datum	08.05.2019
Erwartetes Resultat	1. Schicht erscheint, 2. Änderung erscheint, 3. Schicht erscheint nicht mehr.
Effektives Resultat	Alle erwarteten Resultate sind eingetroffen.
Teststatus	Erfolgreich, ohne Bemerkungen.

Ablauf

1. Plan-Bearbeitung geöffnet.
2. Schicht mit Formular definiert und gespeichert (1. Erwartetes Resultat eingetroffen).
+ Dienstplan geöffnet und Einsatz nach FT-002 mit neuer Schicht erstellt.
3. Erstellte Schicht bearbeitet und gespeichert (2. Erwartetes Resultat eingetroffen).
+ Dienstplan geöffnet und Änderungen überprüft.
4. «Löschen» Knopf im Formular zur erstellten Schicht angeklickt.
5. Warnungs-Dialog bestätigt (3. Erwartetes Resultat eingetroffen).
+ Dienstplan geöffnet und Änderungen überprüft.

9.4.5 FT-005 – Zeiten erstellen, bearbeiten und löschen

Name	FT-005
Tester / Umgebung	Elia Reutlinger / Nach Konzept
Datum	08.05.2019
Erwartetes Resultat	1. Zeit erscheint, 2. Änderung erfolgreich, 3. Zeit erscheint nicht mehr.
Effektives Resultat	Alle erwarteten Resultate sind eingetroffen.
Teststatus	Erfolgreich, ohne Bemerkungen.

Ablauf

1. Plan-Bearbeitung geöffnet.
2. Zeit erstellt und gespeichert (1. Erwartetes Resultat eingetroffen).
+ Dienstplan geöffnet und Einsatz nach FT-002 in neuer Zeit erstellt.
3. Erstellte Zeit verändert und gespeichert (2. Erwartetes Resultat eingetroffen).
+ Dienstplan geöffnet und Änderungen überprüft.
4. «Löschen» bei Zeit geklickt und gespeichert (3. Erwartetes Resultat eingetroffen).
+ Dienstplan geöffnet und Änderungen überprüft.

9.4.6 FT-006 – Berechtigungen anpassen

Name	FT-006
Tester / Umgebung	Elia Reutlinger / Nach Konzept
Datum	08.05.2019
Erwartetes Resultat	Die Berechtigungen werden übernommen.
Effektives Resultat	Der Benutzer hat die neuen Berechtigungen erhalten.
Teststatus	Erfolgreich

Ablauf

1. Einstellungen als Teamleiter geöffnet.
2. Die Rolle eines bestimmten Mitglieds von «Mitglied» auf «Teamleiter» geändert.
3. In Anwendung abgemeldet (Cookies entfernt), API Login-Parameter auf das bearbeitete Mitglied gesetzt und Anwendung aufgerufen (angemeldet).
+ Eigene Rolle von «Teamleiter» auf «Mitglied» geändert.

Bemerkungen

Es entstanden keine Probleme. Stattdessen war es sogar möglich, die eigene Benutzergruppe wieder auf Mitglied zu ändern, wobei sich die Oberfläche direkt entsprechend anpasste.

9.5 Sicherheitsprüfung

Nach der Implementierung wurde eine Sicherheitsüberprüfung gemäss den im Sicherheitskonzept definierten Gefahren durchgeführt.

Allgemeines zu Cross-Site-Scripting

Da eine Überprüfung der Eingabewerte auf SQL- und Script-Injection im Back-End ausgeführt wird, müsste zur Vermeidung von Stored XSS nicht auf die Validierung von Eingabewerten geachtet werden. Wenn sich unter Rückgabewerten der API trotzdem XSS-Code befinden sollte (Reflected und Stored), wird dieser nicht ausgeführt, da es nie zur Ausführung eines Rückgabewertes kommt. Dies wird durch Vue.js garantiert, indem dynamische Werte grundsätzlich nicht als Code evaluiert werden.

Die einzige Ausnahme ist der «v-html» Parameter, welcher in einer Komponente zur Darstellung von reinem HTML-Code eingesetzt werden kann. Zusammen mit dem HTML-Code könnte hier auch Script-Code dargestellt und evaluiert werden. Die Anwendung nutzt den «v-html» Parameter jedoch in keiner Komponente, weshalb dieses Risiko auch nicht besteht. Zur Sicherheit wurden die 3 XSS-Varianten trotzdem in der Anwendung überprüft.

9.5.1 A3 – Stored XSS

Hierfür wurde schädlicher Code direkt in der Datenbank gespeichert, um die Validierung im Back-End zu umgehen. Anschliessend wurde der betroffene Datensatz abgerufen und konnte ohne Probleme dargestellt werden.

Schichten



Abbildung 42 - Stored-XSS kann problemlos dargestellt werden.

9.5.2 A3 – Reflected XSS

Dieses Risiko könnte durch die falsche Verwendung von Daten bestehen, welche im Front-End bearbeitet und sogleich wieder dargestellt werden. Als Beispiel dienten hier wieder die Schichten, da diese in den Plan-Einstellungen bearbeitet und anschliessend 1:1 auf dem Dashboard abgebildet werden können. Das Einfügen von schädlichem Code als Titel einer Schicht hat jedoch nirgends Probleme verursacht.

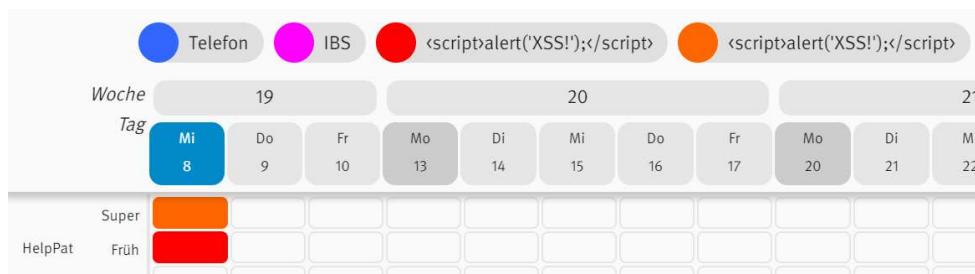


Abbildung 43 - Reflected-XSS führt zu keinen Problemen

9.5.3 A3 – Dom-Based XSS

Pelan arbeitet praktisch mit keinen Parametern aus der URL um Funktionen auszuführen. Das Routing durch den Vue-Router richtet sich jedoch ausschliesslich nach der URL, weshalb hier ein Risiko bestehen könnte. Die Pfadangaben werden vom Router aus der URL gelesen und evaluiert, um den Benutzer auf die richtige Seite umzuleiten. Schädlicher Code in der URL könnte also ebenfalls evaluiert werden, was aber in Berücksichtigung von Kriterium A9 vom Vue-Router selbst vermieden werden muss. Überprüft wurde trotzdem, jedoch wird wie erwartet die Fehlerseite 404 angezeigt.

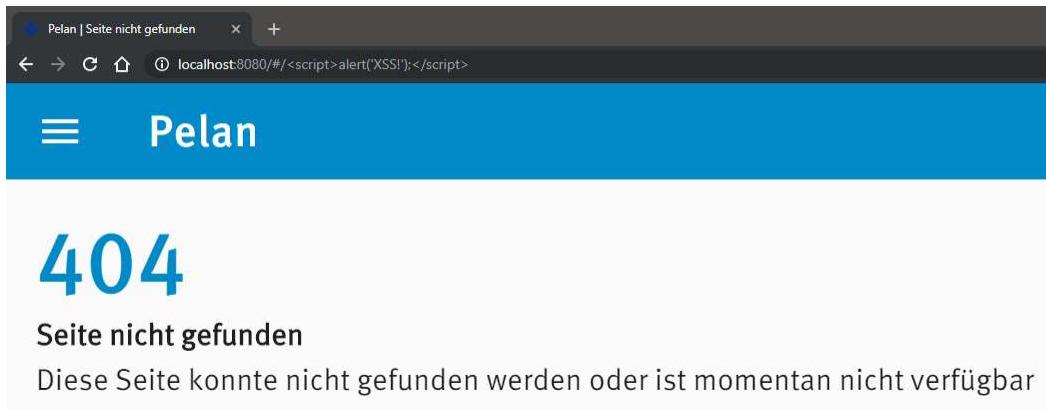


Abbildung 44 - Dom-Based-XSS führt zu keinem Ergebnis

9.5.4 A7 – Fehlerhafte Autorisierung auf Anwendungsebene

Die Anwendung ist an den folgenden 2 Stellen betroffen:

- **Laden einer Seite durch den «vue-router»**

Der vue-router orientiert sich an Pfadangaben aus der URL, um eine bestimmte Seite der Anwendung darzustellen. Dabei kann die URL vom Benutzer verändert werden, weshalb vor dem Laden einer Seite die Berechtigungen überprüft werden müssen. Diese Überprüfung wird beim Öffnen der Anwendung als allererstes eingebunden und orientiert sich an der Benutzergruppe und Berechtigungsstufe der angefragten Route. Überprüft wurde dies, indem als normaler Nutzer die Plan-Einstellungen aufgerufen wurden, wobei wie erwartet die 401 (Keine Rechte) Fehlerseite erschien.

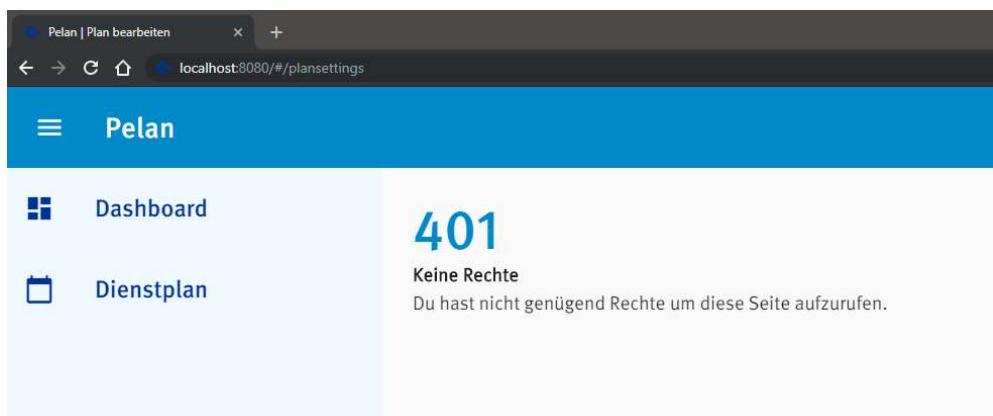


Abbildung 45 - Unberechtigte Zugriffe sind nicht möglich

- **Unberechtigtes Darstellen einer Komponente oder eines Formulars**

Die Einstellungen sowie der Dienstplan enthalten Elemente, welche nur für Teamleiter verfügbar sein sollen. Da sich diese Elemente auf allgemein zugänglichen Seiten befinden, müssen sie eigenständig versteckt werden, wenn der Benutzer zu wenig Rechte besitzt. Dies geschieht durch eine Überprüfung der Benutzerrolle vor dem Initialisieren der betroffenen Komponente. Getestet wurde dieses Risiko durch Aufrufen der betroffenen Seiten, wobei wie erwartet die Elemente nicht dargestellt wurden.

```
<Users v-if="$store.state.user.role.admin" />
```

Abbildung 46 - Codebeispiel: Prüfung von Berechtigungen vor der Initialisierung einer Komponente

Allgemeine Anmerkung

Falls ein Angreifer einen alternativen Weg finden sollte, um sich unberechtigten Zugang zu Formularen zu verschaffen, sollte trotzdem kein Sicherheitsproblem entstehen. Wenn es zu einer unberechtigten Anfrage an die API kommen würde, wird diese dort sogleich abgebrochen, da die API die Berechtigungen jeder Abfrage mit dem JSON-Web-Token und der Datenbank abgleicht.

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

Durch die Konfiguration der API ist die Anwendung gezwungen, einen zusätzlichen «Header» an jede Anfrage zu binden. Dieser enthält einen Sicherheits-Token und wird nicht automatisch vom Browser mit einer Anfrage versendet, wodurch eine Anfrage ausgelöst von einer anderen Webseite nicht gelingen kann.

Eine Gefahr würde bestehen, wenn dies der einzige Token wäre, der zur Authentifizierung im Back-End genutzt werden würde. Dann könnte ein Angreifer den Token abgreifen und damit selbst Anfragen versenden. Da die API jedoch noch weitere Token registriert, auf welche weder durch den Benutzer noch einen Angreifer zugegriffen werden kann, besteht dieses Risiko auch nicht. Da dies aber Sache der API ist wird es in dieser Dokumentation nicht detaillierter erläutert.

9.5.6 A9 – Nutzung von Komponenten mit bekannten Schwachstellen

Die verwendeten Abhängigkeiten wurden vor ihrem Einsatz auf bekannte Schwachstellen geprüft.

Dazu wurde die «Snyk Vulnerability DB»¹⁶ genutzt, woraus folgendes geschlossen werden konnte:

Abhängigkeit (Version)	Bekannte Risiken	Massnahmen
Vue (2.6.10)	Keine	Nicht notwendig
Vue-color (2.7.0)	Keine	Nicht notwendig
Vue-i18n (8.11.2)	Keine	Nicht notwendig
Vue-notification (1.3.16)	Keine	Nicht notwendig
Vue-router (3.0.6)	Keine	Nicht notwendig
Vuex (3.1.0)	Keine	Nicht notwendig
Vuetify (1.5.14)	Keine	Nicht notwendig
Js-cookie (2.2.0)	Keine	Nicht notwendig
Register-service-worker (1.6.2)	Keine	Nicht notwendig
Axios (0.18.0)	Denial of Service (DoS)*	Nicht notwendig

* Zu Axios ist aktuell ein DoS-Problem bekannt, welches bei der Verwendung des «*maxContentLength*» Parameters auftritt. Dieser Parameter sollte nach einer Anfrage die Verarbeitung der Rückgabewerte verhindern, falls diese insgesamt länger sind als im Parameter definiert. In der aktuellen Version wird der Parameter angeblich nicht berücksichtigt, wodurch ein «Denial of Service» Risiko besteht. Betroffen ist Pelan aber nicht, da dieser Parameter nie verwendet wird.

Bei der Überprüfung der Abhängigkeiten ist schlussendlich kein problematisches Risiko gefunden worden, weshalb es nicht nötig war, zusätzliche Massnahmen zu ergreifen.

9.5.7 A10 – Ungeprüfte Um- und Weiterleitungen

Die Anwendung enthält zum aktuellen Zeitpunkt keine Weiterleitungen auf externe Webseiten. Jedoch werden unterschiedliche Inhalte von externen Ressourcen geladen, welche allerdings überprüft und frei von schädlichem Code sind. Des Weiteren bestehen auf der Support-Seite einige Verlinkungen, welche jedoch keine Webseiten öffnen (E-Mail- und Telefon-Links) und damit auch kein Risiko darstellen.

Externe Ressourcen werden von folgenden, verifizierten Endpunkten geladen:

- «Material Icons» Schriftart, Quelle: fonts.googleapis.com (Google).
- CSS-Animationen, Quelle: cdn.jsdelivr.net/npm/animate.css@3.5.1 (CDN).
- Allgemeine Anwendungsinhalte, Quelle: Pelan-API (Lokal bzw. Baloise-Intern).

¹⁶ Snyk.io – Vulnerability DB

10 Abschluss

10.1 Auswertung des Kanban-Boards

Diese Diagramme sollen die Umsetzung des Kanban-Boards auf GitHub visualisieren. Dabei ist zu erwähnen, dass als Vorrarbeit bereits 19 Issues mit grundlegenden Aufgaben erfasst wurden, während die folgenden Grafiken nur die Durchführungsphase der IPA abbilden.

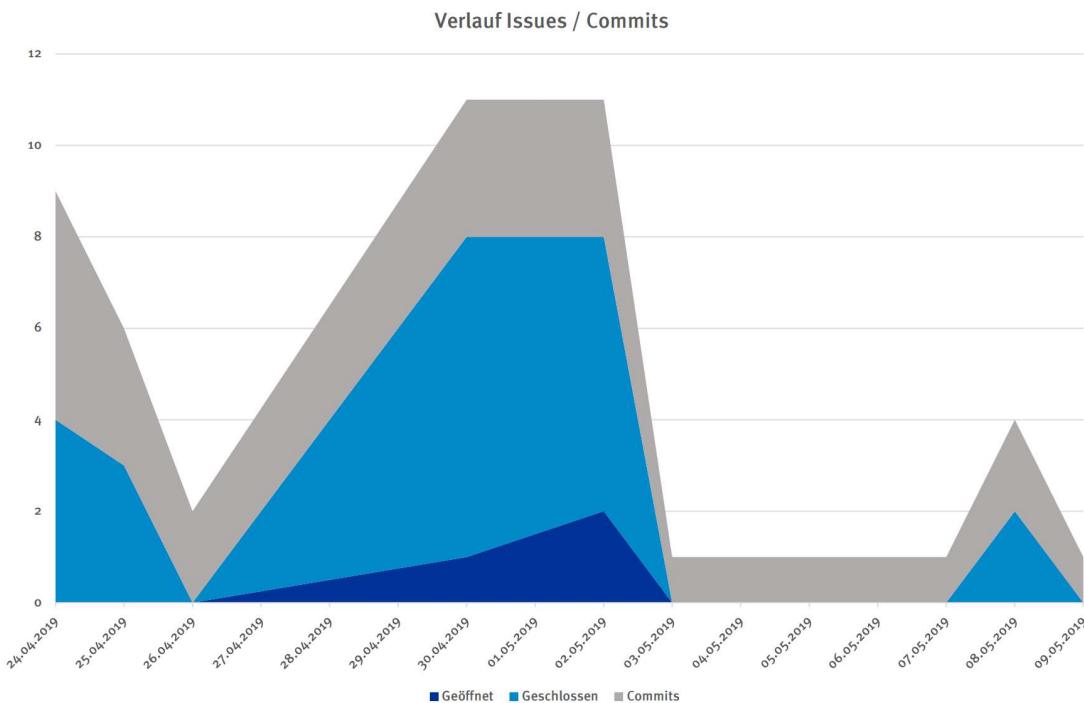


Abbildung 47 - Kanban-Auswertung Diagramm 1

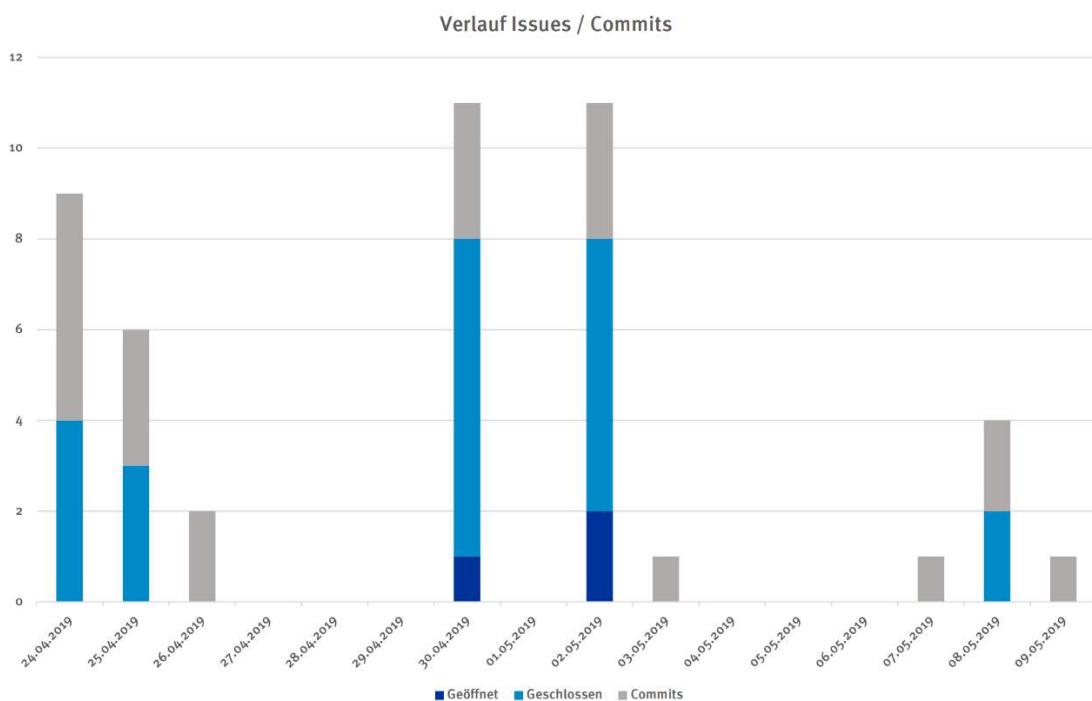


Abbildung 48 - Kanban-Auswertung Diagramm 2

10.2 Funktionen & Verbesserungen für die Zukunft

Um die Anwendung termingerecht zu implementieren musste während der IPA leider auf einige Details verzichtet werden. Der vorgegebene Zeitrahmen war ohne Zweifel sehr knapp, weswegen die Entwicklung schnellstmöglich erledigt werden musste. Daraus folgt, dass an manchen Stellen Optimierungen oder tiefgreifende Funktionen bewusst weggelassen wurden. Es lag in diesem Sinne eher die problemlose Funktion der Anforderungen im Fokus und weniger die Umsetzung von optimierenden Details. Da ein sehr dynamisches Framework verwendet wird, sind solche Optimierungen in Zukunft problemlos und schnell möglich, wobei der aktuelle Stand lediglich die Anforderungen der detaillierten Aufgabenstellung bzw. des Helpdesks erfüllt. Für den Helpdesk könnte die Anwendung bereits jetzt und ohne weitere Massnahmen produktiv eingerichtet werden, worauf aber weitere Schritte folgen werden, da die Anwendung eines Tages von allen Teams der Baloise genutzt werden können soll.

Mehrere Teams

Eine definitiv umzusetzende Funktion ist eine Art Team-Verwaltung auf UI-Ebene. Bei der im Voraus implementierten Pelan-API wurde dieser Aspekt in der Logik und Datenbank bereits berücksichtigt. Es wäre also grundsätzlich möglich, mehrere Teams mit eigenen Plänen zu betreiben. Da dies jedoch keine Anforderung während der IPA war, ist es über die jetzige Anwendung noch nicht möglich ein eigenes Team zu erstellen. Die Umsetzung sollte jedoch nicht viel Aufwand mit sich bringen, da es sich nur um eine weitere Ansicht mit einem einfachen Formular handeln sollte.

Kompakte API-Anfragen

Beim Laden der Einsätze auf dem Dienstplan wird aktuell für jeden einzelnen Benutzer eine Anfrage an die API erstellt. Dadurch entstehen (im Fall des Helpdesk-Teams) bei jedem Aufruf des Plans ungefähr 30 parallele Anfragen. Wenn man davon ausgeht, dass nur der Helpdesk die Anwendung nutzt, würde dies kein allzu grosses Problem darstellen. Aber gerade in Berücksichtigung des vorherigen Punktes könnten mehrere Teams durch eine steigende Anzahl von Anfragen die API irgendwann überlasten, was zu einer verlangsamten Auslieferung der Daten führt. Während der IPA konnte dieser Punkt leider noch nicht berücksichtigt werden, da die Pelan-API aktuell nur Abfragen zu einzelnen Benutzern anbietet. Man müsste die API zuerst erweitern, um anschliessend eine einzelne Anfrage für mehrere Benutzer zu ermöglichen.

Pinsel-Tool

Eine weitere Idee zur Vereinfachung des Dienstplans bzw. Beschleunigung der Planung ist eine Art Pinsel, mit welcher ein Teamleiter mehrere Einsätze gleichzeitig Planen kann. Während aktuell ein Feld des Plans angeklickt und per Formular definiert werden muss, könnte in Zukunft eine Schicht aus der Legende gewählt, und durch einfaches überstreichen im Plan zugeteilt werden. Nach grober Einschätzung würde die Implementierung dieser Funktion aber viel Aufwand mit sich bringen, da dazu weitere Abhängigkeiten und zusätzliche API-Endpunkte benötigt werden würden.

Tests

Für die IPA wurden manuelle UI-Test (auch «Blackbox»-Tests genannt) definiert und durchgeführt. Diese testen nur die sichtbare Oberfläche und nicht die korrekte Verarbeitung im Hintergrund, sind aber sehr unkompliziert durchführbar und benötigen wenig Zeit. Da während der IPA nicht viel Zeit zur Verfügung stand reichten uns diese Tests. Um die Anwendung aber auch auf ihre Funktionalitäten und das Zusammenspiel der Komponenten zu testen, könnte man hier noch automatisierte Tests hinzufügen. Dazu bestehen bereits zahlreiche Testing-Frameworks, welche auch speziell auf Vue.js Anwendungen ausgelegt sind.

10.3 Danksagung

Während der IPA sowie in der Vorbereitungsphase standen mir zahlreiche Personen zur Seite, die mir mit Ratschlägen und Unterstützung weitergeholfen haben. Ohne diese Unterstützung wäre ich womöglich nicht im Stande gewesen, diese Abschlussarbeit in solch einem Rahmen durchzuführen.

Speziell danke ich meinem Fachvorgesetzten Matthias Cullmann, der mich in den letzten 1.5 Jahren in der Baloise begleitet hat. Durch die großartige Zusammenarbeit und die äusserst freie Wahl von Aufgaben und Umsetzungsmöglichkeiten konnte ich mir viele neue Herausforderungen setzen. Die vergangenen 1.5 Jahre waren somit die lehrreichsten und eindrücklichsten meiner gesamten Lehrzeit. Ich konnte viel selbstständig arbeiten und meine Kenntnisse umfänglich erweitern, wobei mir Matthias bei unterschiedlichsten Fragen immer geholfen hat.

Auch möchte ich mich bei der Nachwuchsentwicklung und namentlich bei Adrian Fritsch bedanken, da ich mich hier bedenkenlos mit allen Anliegen melden konnte und man mir immer tatkräftig behilflich war. Mit dem Cash-Calculator Projekt hatte ich ausserdem bereits im 3. Lehrjahr die Chance, einen mir vielbedeutenden Beitrag für die Baloise zu leisten, womit ich auch zum ersten Mal richtig viele Eindrücke und Erkenntnisse zur Projektabhandlung in einem grossen Betrieb sammeln konnte. Das Projekt stellte wegen meinen damals noch dürftigen Kenntnissen ein gewisses Risiko dar, war aber eine grosse Bereicherung für mich und meine Ausbildung.

Des Weiteren möchte ich noch allen danken, die mich während meiner Lehre im Betrieb und der Berufsschule begleitet haben. Ich war immer froh über jede Art von Unterstützung, die ich erhalten konnte und finde es grossartig, so viele neue Bekanntschaften gemacht zu haben. Dazu gehört auch meine Familie, welche immer hinter mir stand und mich bei Bedarf in allen Bereichen stützte. Ausserdem haben sie sich noch kurzfristig zum Gegenlesen dieser Doku Zeit genommen und mir abschliessend einige Tipps gegeben.

11 Anhang

11.1 Quellenverzeichnis

Allgemeine Anmerkung

Diese Arbeit wurde mit der Rechtschreibprüfung von MS Word sowie manuell überprüft. Das gesamte Projekt wurde auf GitHub durchgeführt, wo sich auch das Kanban-Board und Rohdaten zur Doku und der Anwendung finden lassen. Über folgenden Link kann das Repository erreicht werden:

<https://github.com/baloise/pelan>

Quellen

- Apachefriends.org	Informationen über die Anwendung «XAMPP»
- Wikipedia.org	Für unterschiedliche Bereiche
- Netlify.com	Für Details zum Dienst
- Jwt.io	Für Details über «JSON-Web-Token»
- Webstyleguide.baloise.com	Firmenstandards
- Brandportal.baloise.com	Firmenstandards
- Baloise.ch	Datenschutzbestimmungen
- Owasp.org	Sicherheitsrelevante Informationen/Richtlinien
- Airlock.com	Informationen zu «Airlock-Medusa»
- Vuetifyjs.com	Theoretisches zur Library «Vuetify»
- Developers.google.com	Details zu Lighthouse-Audit Tests
- Snyk.io	Daten über bekannte Sicherheitsschwachstellen

Weitere Quellen, welche beispielsweise einen Einfluss auf Erarbeitung der Anwendung hatten, wurden im entsprechenden Arbeitsjournal erwähnt.

Grafiken/Diagramme/Bilder

Alle Grafiken* stammen entweder aus Eigenarbeit oder sind Bildschirmaufnahmen aus unterschiedlichen Anwendungen. Keine dieser Grafiken existiert in dieser Form in anderen Plattformen/Dokumenten. Als Inspiration könnten die oben genannten Quellen gedient haben.

- * Das Baloise-Group Logo sowie die abgebildeten Elemente zu den Firmenstandards stammen von den in den Quellen genannten Webseiten.

11.2 Abbildungsverzeichnis

Abbildung 1 - Organigramm der Projektrollen	18
Abbildung 2 - Arbeitsplatz während der IPA	19
Abbildung 3 - Struktur der IPA-Daten	19
Abbildung 4 - Baloise-Group Logo	23
Abbildung 5 - Textfelder nach Firmenstandard	23
Abbildung 6 - Buttons nach Firmenstandard	23
Abbildung 7 - Vergleich der Dienstpläne als Tabelle und in Pelan	34
Abbildung 8 - Vue.js Komponenten Konzept	39
Abbildung 9 - Dienstplan als Excel-Tabelle	40
Abbildung 10 - Umgebung und Schnittstellen zum Projekt	41
Abbildung 11 - Kanban-/Project-Board auf GitHub	43
Abbildung 12 - Use-Case-Diagramm über alle Anforderungen	44
Abbildung 13 - Abhängigkeiten in Relation	54
Abbildung 14 - Codebeispiel: Registrieren einer Route/Ansicht	55
Abbildung 15 - Codebeispiel: i18n Übersetzungen	56
Abbildung 16 - Codebeispiel: "Watcher"-Funktion zum Anpassen der Sprache	56
Abbildung 17 - Sequenzdiagramm zur Authentifizierung	57
Abbildung 18 - Codebeispiel: Authentifizierung Schritt 1	58
Abbildung 19 - Codebeispiel: Authentifizierung Schritt 2	58
Abbildung 20 - Codebeispiel: Authentifizierung Schritt 3	58
Abbildung 21 - Codebeispiel: Farben in Vuetify-Konfiguration	59
Abbildung 22 - Design-Struktur der Anwendung	59
Abbildung 23 - Support-Ansicht mit Kontaktinformationen	60
Abbildung 24 - Dashboard-Ansicht	61
Abbildung 25 - Codebeispiel: Abrufen von Schichten und Tageszeiten	61
Abbildung 26 - Performance-Test durch Lighthouse-Audit	62
Abbildung 27 - Einstellungen-Ansicht (als Teamleiter)	62
Abbildung 28 - Aktivitätsdiagramm "Sprache anpassen"	63
Abbildung 29 - Darstellung der Schichten in den Plan-Einstellungen	64
Abbildung 30 - Darstellung der Tageszeiten in den Plan-Einstellungen	64
Abbildung 31 - Aktivitätsdiagramm "Schicht hinzufügen"	65
Abbildung 32 - Komponenten/Design nach Firmenstandard	65
Abbildung 33 - Dienstplan-Ansicht	66
Abbildung 34 - Aufbau/Struktur der Komponenten im Dienstplan	66
Abbildung 35 - Codebeispiel: Objekt.Assign() ermöglicht dynamisches Zuweisen neuer Objekte	67
Abbildung 36 - Interaktionsübersichtsdiagramm zu den Elementen im Dienstplan	68
Abbildung 37 - Legende zum Diagramm	68

Abbildung 38 - Codebeispiel: Vererbung der Auswahl zum Zeitraum	69
Abbildung 39 - Formular zur Auswahl des Zeitraums	69
Abbildung 40 - Codebeispiel: Benutzer-Zeile als Komponente.....	69
Abbildung 41 - Codebeispiel: "Watcher"-Funktion auf Änderungen in den Einsätzen.....	69
Abbildung 42 - Stored-XSS kann problemlos dargestellt werden.....	73
Abbildung 43 - Reflected-XSS führt zu keinen Problemen.....	73
Abbildung 44 - Dom-Based-XSS führt zu keinem Ergebnis	74
Abbildung 45 - Unberechtigte Zugriffe sind nicht möglich	74
Abbildung 46 - Codebeispiel: Prüfung von Berechtigungen vor der Initialisierung einer Komponente ..	75
Abbildung 47 - Kanban-Auswertung Diagramm 1.....	77
Abbildung 48 - Kanban-Auswertung Diagramm 2	77