

IPA Dokumentation

IPA-Daten			
Projektname	Ressourcenplanungs-Webapp		
Firmenname	Technische Fachschule Bern, Abteilung Informatik		
Berufsschule	Technische Fachschule Bern		
Autor	Anderegg Yannis		
Experten	VEX: Meyer Katrin HEX: Müller Lorenz NEX: Gamez Daniel		
Verantwortliche Fachkraft	Altin Özcan		
Berufsbildner	Iannattone Giulio		
Fachrichtung	BET		
Projektvorgehensmodell	HERMES 5.1		
Jahrgang und Kanton	IPA 2021, Kanton Bern		
Ausgabedatum	19.03.2021		
Status	In Arbeit <input type="checkbox"/>	In Prüfung <input checked="" type="checkbox"/>	Zur Nutzung genehmigt <input type="checkbox"/>

Tabelle 1: IPA-Dokumentation

Versionsverlauf

Version:	Datum:	Beschreibung:
1.1	03.03.2021	Erstversion der Dokumentenvorlage
1.2	04.03.2021	Überarbeitete Dokumentenvorlage, Formatvorlagen etc.
1.3	04.03.2021	Phasenfreigabe Initialisierung
1.4	08.03.2021	Phasenfreigabe Konzept
1.5	11.03.2021	Phasenfreigabe Realisierung
1.6	18.03.2021	Realisierung abgeschlossen
1.7	19.03.2021	Überarbeitung der Dokumentation & Abgabe
1.8		

Tabelle 2: Versionsverlauf

1 Kurzfassung des IPA-Berichts

Um was es bei dieser IPA geht und wie die Ausgangssituation aussieht.

1.1 Info

Das folgende Dokument beschreibt die Ergebnisse der 10 Tage dauernden individuellen praktischen Arbeit (IPA).

Die dokumentierte Arbeit besteht aus einer lokal gehosteten Fullstack-Webapplikation. Die verwendeten Technologien repräsentieren den neusten Stand im Bereich Web-Development und wurden in diesem Dokument auf eine für den Leser möglichst verständliche Weise dokumentiert.

1.2 Kurze Ausgangssituation

In meiner IPA habe ich versucht ein Problem in der Schulumgebung zu lösen. Es war mir ein Bedürfnis, dass meine Applikation einen praktischen Nutzen erfüllt.

An der TF Bern werden einige repetitive Arbeiten durch die Schüler erledigt. Welcher Schüler wann zugeteilt ist, wurde bisher in einem unübersichtlichen Zeitplan festgehalten. Als Alternative dafür, und für weitere ähnliche repetitive Arbeiten, soll eine Java Script/Vue.js Webapplikation erstellt werden. Eine Lehrkraft soll einfach die Möglichkeit haben, die zu verwaltenden Schüler einem Termin zuzuteilen.

1.3 Umsetzung

Während dieser IPA wurde ein Webtool zu Administrierung von Terminen erstellt und dokumentiert. Es wurden hauptsächlich Java Script und darauf basierende Frameworks verwendet. Die Webapp nutzt das Basisframework Vue.js mit dem Design-Framework Vuetify.js im Frontend und Node.js mit dem Express.js Framework im Backend.

Umgesetzt wurde das Projekt mit der Projektmanagementmethode HERMES 5.1, welche verwendet wurde, um das ganze Projekt zu konzeptionieren und durchzuführen. Dabei wurden die Vorgaben von PKOrg, sowie der Kriterienkatalog für das Jahr 2021 beachtet.

1.4 Ergebnis

Die Ergebnisse sind in dem vorgegebenen Zeitrahmen entstanden. Der Kandidat hat sich bestmöglich an die zeitliche Planung gehalten, und jegliche Abweichung dokumentiert.

Das erarbeitete Produkt entspricht dem erwarteten Ergebnis und erfüllt die Kernanforderungen. Sämtliche Test-Cases wurden erfüllt.

INHALTSVERZEICHNIS

IPA Dokumentation	1
Versionsverlauf	2
1 Kurzfassung des IPA-Berichts	3
1.1 Info	3
1.2 Kurze Ausgangssituation	3
1.3 Umsetzung	3
1.4 Ergebnis	3
2 Aufgabenstellung	9
2.1 Titel der Arbeit	9
2.2 Ausgangslage	9
2.3 Detaillierte Aufgabenstellung	9
2.4 Mittel und Methoden	11
2.5 Vorkenntnisse	11
2.6 Vorarbeiten	11
2.7 Neue Lerninhalte	11
2.8 Arbeiten in den letzten 6 Monaten	11
3 Standards	12
4 IPA-Schutzbedarfsanalyse	13
4.1 Zugriff	13
4.2 Schützenswerte Daten	13
4.3 Passwörter	13
5 Organisation der IPA-Ergebnisse	14
5.1 Namenskonzept der Dateien	14
5.2 Lokales Datensicherungskonzept	14
5.3 Ergänzendes Datensicherungskonzept	14
5.4 Wiederherstellung	15
5.4.1 Dokumentation und Anhänge wiederherstellen	15
5.4.2 Entwicklungsdateien wiederherstellen	16
5.4.3 Wiederhergestellte Entwicklungsdatei	17
5.5 Dokumentenablage und Ordnerstruktur	18
5.6 Arbeitsumgebung	19
5.6.1 Arbeitsplatz	19
5.6.2 Lokales Arbeitsgerät	19

6 Projektvorgehen.....	20
6.1 Projektmethode.....	20
6.2 Phasen.....	20
7 Projektorganisation.....	21
7.1 Organigramm.....	21
7.2 Detaillierte Informationen zu Projektrollen	22
8 Risikoanalyse.....	23
8.1 Legende.....	23
8.1.1 Schadensausmass und Eintrittswahrscheinlichkeit.....	23
8.1.2 Gesamtrisikoindex.....	23
9 Risikograph	25
9.1 Informationen zur Risikoanalyse.....	26
10 Zeitplan.....	27
10.1 Meilensteine.....	28
11 Arbeitsjournal.....	29
11.1 Erster Tag, 3. März 2021	29
11.2 Zweiter Tag, 4. März 2021	31
11.3 Dritter Tag, 5. März 2021.....	33
11.4 Vierter Tag (Halbtag), 8. März 2021	35
11.5 Fünfter Tag, 10. März 2021	37
11.6 Sechster Tag, 11. März 2021	39
11.7 Siebter Tag, 12. März 2021	41
11.8 Achter Tag (Halbtag), 15. März 2021	43
11.9 Neunter Tag, 17. März 2021	45
11.10 Zehnter Tag, 18. März 2021	47
11.11 Elfter Tag, 19. März 2021.....	48
12 Abschlussbericht.....	49
12.1 Vergleich IST/SOLL	49
12.2 Persönliches Fazit und Schlussreflexion	49
13 Selbstständigkeitserklärung.....	50
14 Phasenfreigabe Initialisierung.....	51
Teil 2: Projektdokumentation	52
15 Einführung.....	53
16 Initialisierung	54

16.1 IST-Situation Beschrieb	55
16.2 SOLL-Situation Beschrieb	56
16.2.1 Übersichts-Seite	58
16.2.2 Kalenderansichts-Seite	58
16.3 Persönliche Vorgehensziele	58
16.4 Projektziele	59
16.5 Anforderungen	60
16.5.1 Funktionale Anforderungen	60
16.5.2 Nicht Funktionale Anforderungen	61
17 Phasenfreigabe Konzept	62
18 Konzept	63
18.1 Datenbank Design	63
18.2 Entity Relationship Diagram	64
18.2.1 Classes und Students	65
18.2.2 Events	65
18.2.3 Users	65
18.3 Passwortsicherheits-Konzept	66
18.3.1 Passwörter	66
18.3.2 Verschlüsselung	66
18.4 Skalierung	67
18.5 Systemarchitektur und Aufbau	68
18.5.1 Schnittstellen der Applikation	68
18.6 Anwendungsfälle (Use-Cases)	69
18.6.1 Anwendungsfälle im Detail	70
18.6.2 Umsetzung der Anwendungsfälle	71
18.7 Mockups	72
18.7.1 Login-Page	72
18.7.2 Termine	73
18.7.3 Kalender Seite	74
18.8 Testkonzept	75
18.8.1 Testdaten	75
18.8.2 Testrahmen	75
18.8.3 Testumgebung	75
18.8.4 Test-Methoden	75

18.8.5 Re-Testing	75
18.8.6 Mängelklassen und Testkategorien	76
18.8.7 Test-Tabelle	77
18.9 Testfälle	78
18.9.1 Test 1	78
18.9.2 Test 2	79
18.9.3 Test 3	80
18.9.4 Test 4	81
18.9.5 Test 5	82
18.9.6 Test 6	83
19 Realisierung	84
19.1 Projektumgebung	84
19.1.1 Packages	85
19.2 Ordnerstruktur	86
19.2.1 Frontend	86
19.2.2 Backend	87
19.3 Datenbank	88
19.3.1 Erstellung der Datenbank und Tabellen	89
19.4 Login	90
19.4.1 Ablauf eines Login Vorgangs	90
19.4.2 JSON-Webtoken	91
19.4.3 Bcrypt.js	93
19.5 Frontend	94
19.5.1 Aufbau der Komponenten	94
19.5.2 Router im Frontend	95
19.5.3 Frontend – Endpoints	96
19.5.4 Login.vue (Login Seite)	97
19.5.5 Events.vue (Terminansicht) - Design	98
19.5.6 Events.vue - Methoden	100
19.5.7 Calendar.vue (Kalenderansicht)	103
19.5.8 App.vue (Hauptfile Frontend)	105
19.5.9 Vergleich mit konzeptionellen Mockups	106
19.6 Backend	109
19.6.1 index.js – Hauptfile im Backend	109

19.6.2 Backend - Endpoints	110
19.6.3 API event.js	111
19.7 Weitere Schritte und Pläne in der Zukunft	112
19.8 Testfälle – Durchführung	113
19.8.1 Test 1	113
19.8.2 Test 2	115
19.8.3 Test 3	117
19.8.4 Test 4	119
19.8.5 Test 5	120
19.8.6 Test 6	121
20 Quellenverzeichnis	123
21 Glossar	124
22 Abbildungsverzeichnis	125
23 Tabellenverzeichnis	126
24 Backup Versionsverlauf	128
24.1 Gitlab	128
24.2 Lokal (PC und USB-Medium)	129
25 Sitzungsprotokolle	130
26 Anhang – Sourcecode	132
26.1 SQL – Script (create tables)	132
26.2 Backend (api/event.js)	135
26.3 Backend (db/migrations/)	136
26.4 Backend (db/seeds/users.js)	137
26.5 Backend (knexfile.js)	138
26.6 Backend (index.js)	139
26.7 Frontend (src/router/index.js)	142
26.8 Frontend (src/views/calendar.vue)	143
26.9 Frontend (src/views/events.vue)	149
26.10 Frontend (src/views/login.vue)	158
26.11 Frontend (src/app.vue)	160

2 Aufgabenstellung

Wird ohne Änderungen aus «PkOrg» übernommen.

2.1 Titel der Arbeit

Ressourcenplanungs-Webapp

2.2 Ausgangslage

An der Technischen Fachschule Bern gibt es unter den Klassen immer wieder Ämtli zu vergeben. Dabei müssen verschiedene Aufgaben an verschiedenen Wochentagen wahrgenommen werden. In der Regel wechseln sich die Lernenden von Woche zu Woche ab. Je nach «Ämtli» wird die Aufgabe in Gruppen oder als Einzelauftrag erledigt. Dieser «Einsatzplan» wird bisher als eine Excel-Liste geführt, wie man das erwarten würde. Allerdings haben sich einige Unschönheiten mit Excel herausgestellt: es gab zum Beispiel relativ grossen Aufwand, wenn ein Lernender krank war. Man musste nachfolgende Lernende im Einsatzplan nachrücken. Noch grösser war das Problem, wenn ein Lernender diese Aufgabe aus irgendeinem Grund gar nicht mehr wahrnehmen konnte (Kündigung, Dispens). Dann mussten alle Einsätze mühsam nachgerückt werden. Also wurde beschlossen unsere Einsatzpläne in eine Webapp zu verschieben.

2.3 Detaillierte Aufgabenstellung

Es muss also eine Webapplikation entstehen, womit die Einsatzplanung von wiederkehrenden «Ämtli» an der Technischen Fachschule Bern erstellt werden kann. Die Lösung des Problems mit den Abwesenheiten, ist NICHT Teil der IPA und wird nach der IPA analysiert und umgesetzt.

Ziele:

Z1: Die Einsatzpläne sollen für alle im Browser einsehbar sein.

Z2: Die Erstellung von Einsatzplänen soll für Lehrpersonen erleichtert werden.

Damit sich der Kandidat auf die wesentlichen Funktionen der Webapplikation konzentrieren kann, sollen die Lernenden einmalig mit einem SQL-Skript in der Datenbank eingefügt werden. Die Anforderungen an das SQL-Skript sind im Folgenden auch beschrieben.

Anforderungen:

Erstellen eines Einsatzplanes:

A1.1: Es soll ein Einsatzplan mit der entsprechenden Bezeichnung erstellt werden können. (z.B. «Bodenwischen am Freitag»)

A1.2: Für jeden Einsatzplan soll eine Klasse ausgewählt werden können, deren Schüler automatisch als Einsatzkräfte gesetzt werden.

A1.3: Für jeden Einsatzplan kann der Wochentag (MO-FR) angegeben werden, an dem der Einsatz ausgeführt wird (Es wird angenommen, dass ein Ämtli nur an einem Wochentag ausgeführt wird)

A1.4: Für jeden Einsatzplan kann die Gruppengrösse gesetzt werden (z.B. braucht es 3 Personen pro Einsatz fürs Bodenwischen am Freitag)

A1.5: Nachdem obige Einzelheiten angegeben worden sind, erstellt die Webapp auf Knopfdruck einen Einsatzplan für das jeweilige Ämtli mit der jeweiligen Gruppengrösse am gewünschten Wochentag. Dabei werden automatisch abwechselnd Lernende aus der gewählten Klasse gewählt.

Einsicht des Einsatzplanes:

A2.1: Der erstellte Einsatzplan wird übersichtlich dargestellt. Das heisst man kann alle Einsätze mit Personenzuteilung für den aktuellen und nächsten Monat sehen.

A2.2: Will man weiter in die Zukunft schauen, kann man die Ansicht «blättern».

Login:

A3.1: Wenn man nicht angemeldet ist, kann man die Einsatzpläne einsehen, aber nichts verändern (nur A2.1-2.2 verfügbar)

A3.2: User können sich auch einloggen. Dann ist "Erstellen eines Einsatzplanes" (A1.1-A1.5) auch verfügbar.

A3.3: User können sich nicht registrieren, sondern erhalten Credentials von einem Administrator.

Einmaliges Einlesen von Lernenden in die DB:

A4.1: Es besteht ein SQL Skript, welches automatisch alle Lernenden der Abteilung «Informatik» in die Datenbank einfügt, so dass sie bei A1.2 verwendet werden können.

Das Deployment der IPA auf einen Server ist nicht Teil der IPA. (Die App soll später nur im Intranet gehostet werden).

Es ist ein Testkonzept zu erstellen, welches die obigen Anforderungen abdeckt.

Es gelten die Coding Conventions der ICT-Berufsbildung.

2.4 Mittel und Methoden

Projektmethode: - HERMES 5

Technologie-Stack: - Vue.js & Vuetify (Frontend) - Express.js & MySQL (backend) - axios, knex.js

Entwicklungsumgebung: - Visual Studio Code auf Windows 10

Conventions: - Es gelten die von PkOrg zur Verfügung gestellten Coding-Conventions

2.5 Vorkenntnisse

Der Kandidat konnte bereits in vergangenen Modulen und Projekten viele der angewendeten Techniken kennenlernen.

Im Rahmen der Module 133 und 141, bzw. 151 wurden die Themen Vue.js, DB-Zugriff mittels Knex.js und SQL behandelt.

Des Weiteren konnte der Kandidat bei diversen Werkstattprojekten neue ergänzende Frameworks wie Express.js oder Axios anwenden.

2.6 Vorarbeiten

Da die Komponentenlibrary "Vuetify" zum Einsatz kommen soll, muss sich der Kandidat in "Vuetify" etwas einarbeiten und die Philosophie davon kennenlernen, sodass während der IPA mühelos Vuetify-Komponenten angewendet werden können.

Die DEV-Umgebung ist bereits installiert (MySQL Server Community, MySQL Workbench, Visual Studio Code, node package manager, etc) und muss nicht vorbereitet werden.

2.7 Neue Lerninhalte

Anwendung der Komponentenlibrary "Vuetify»

2.8 Arbeiten in den letzten 6 Monaten

Der Kandidat hat sich in den letzten 6 Monaten vor allem mit HTML, CSS, JAVASCRIPT, und dem Framework Vue.js beschäftigt. SQL & MySQL waren präsent.

3 Standards

Falls firmeninterne Standards existieren, sind diese hier beschrieben.

In der Tabelle im Anschluss sind sämtliche zu beachtenden Aspekte beschrieben. Es wurden firmeninterne Standards, wie auch Vorgaben von PkOrg beachtet.

Standard	Beschreibung
Dokumentenvorlage	Die Dokumentvorlage wurde anhand der Dokumentationsvorgaben von PkOrg erstellt.
Projektabwicklung	Die TF Bern verfügt über keine eigene Projektabwicklungsmethode. Jeder Projektleiter bestimmt diese selbst.
Sicherheitskonzept	Die Betriebsinformatiker an der TF Bern verfügen über kein Sicherheitskonzept. Das Sicherheitskonzept wird vom IPA Kandidaten erstellt und ist in der Dokumentation unter dem Punkt IPA-Schutzbedarfsanalyse aufgeführt.
Programmcode/Skripte	Die TF-Bern hat keine Standards bezüglich Coding Conventions. Daher werden die Coding-Convention von PkOrg verwendet, welche durch die ICT-Berufsbildung Bern bereitgestellt werden.

Tabelle 3: Standards

4 IPA-Schutzbedarfsanalyse

Welche Daten sind schützenswert und wie kann zuverlässiger Schutz gewährleistet werden?

4.1 Zugriff

Der Arbeitslaptop, sowie das TF Bern Firmeninterne Netzwerk sind vor externem Zugriff geschützt. Um sich im Netzwerk oder auf dem Arbeitsgerät anzumelden, ist ein Domänenpasswort nötig. Dieses ist komplex und nur der Kandidat kennt dieses.

4.2 Schützenswerte Daten

Im Zuge dieser IPA wurden keinerlei besonders schützenswerte oder datenschutzrechtlich relevante Daten verwendet. Die verwendete Entwicklungsdatenbank und die Skripts, um diese zu erstellen, enthalten jeweils nur Name und Vorname von Lernenden und dem Lehrkörper und sind somit nicht priorisiert zu behandeln, was Schutz angeht, da diese ebenfalls öffentlich einsehbar sind.

4.3 Passwörter

Alle Passwörter werden in dem digitalen Passwortvault von dem Anbieter «Dashlane» gesichert. So kann gewährleistet werden, dass kein Passwort verloren geht, oder vergessen wird.

Es werden sowohl die lokalen Logindaten, welche für die Entwicklung gebraucht werden, wie auch alle Passwörter von verwendeten Drittanbietern dort gesichert.

In der Entwicklung verwendete Passwörter werden in der Dokumentation ausgeblendet.

5 Organisation der IPA-Ergebnisse

Wie ist die Arbeitsweise organisiert und in welcher Umgebung wird gearbeitet?

5.1 Namenskonzept der Dateien

Die Dateinamen setzen sich aus dem entsprechenden Namen, Datum und der Versionsnummer zusammen.

5.2 Lokales Datensicherungskonzept

Der hauptsächlich verwendete Entwicklungslaptop enthält jeweils eine aktuelle Arbeitskopie der Dokumentation, sowie dem Projekt. Er ist durch ein Passwort geschützt. Im «. git» Ordner ist zudem ein lokaler Versionsverlauf der Arbeitsdateien gespeichert.

5.3 Ergänzendes Datensicherungskonzept

Als ergänzende Speicherlösung wurden eine Reihe von Cloudbasierten Speicherlösungen evaluiert. Es stehen mehrere Services zur Verfügung, um ein reibungsloses Arbeiten zu ermöglichen. Um die Arbeit plattform- und geräteübergreifend zu erleichtern, macht es Sinn, mehrere unterschiedliche Anbieter von Speicherlösungen zu verwenden. Der Service «Microsoft OneDrive» findet bei der Sicherung der Dokumentation Anwendung. Für das Sichern von Code und Dokumentation wurde der Anbieter «Gitlab» verwendet. Mittels der Software «Sourcetree» kann der Versionsverlauf verwaltet werden.

Als Ergänzungslösung dient der Dienst «Drive» von Google. Hier werden oft verwendete Dateien und Anhänge gesichert.

Der Zugriff auf Backup Lösungen von Drittanbietern ist immer durch ein Login mit Passwort gesichert (siehe Kapitel «IPA-Schutzbedarfsanalyse»).

Name der Lösung (Anbieter):	Einsatzzweck:
Google Drive	Backup (Vollsicherung)
Microsoft One Drive	Backup (Differenziell)
Gitlab über Software Sourcetree	Backup (Differenziell)

Tabelle 4: Cloud-Speicher

Als redundante Lösung steht ein USB-Flashdrive zur Verfügung, welches die tagesaktuelle Version der Dokumentation, sowie deren Anhänge enthält.

Speichermediumstyp und Bez,	Einsatzzweck:
SanDisk ULTRA 32GB Flashdrive	Backup (Vollsicherung) täglich

Tabelle 5: Portable Speichermedien

5.4 Wiederherstellung

5.4.1 Dokumentation und Anhänge wiederherstellen

Im Falle einer Wiederherstellung von Textdateien wird entweder die aktuelle Version von einem der Cloud-Speicher bezogen, oder es wird die portable Tages-Sicherung von dem USB-Flashdrive Medium geladen.


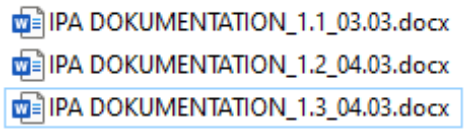
Beschreibung	Screenshot
Der «One Drive» Dienst von Microsoft bietet einen vollumfänglichen Versionsverlauf. Hier kann jederzeit die letzte Version wiederhergestellt werden.	
Auf einem USB-Flashdrive werden Tagesversionen gespeichert.	

Tabelle 6: Backup und Versionsverlauf

5.4.2 Entwicklungsdateien wiederherstellen

Bei der Wiederherstellung von Entwicklungsdateien kann mittels Sourcetree über GitLab zurückgesprungen werden. Hier steht ein Versionsverlauf zur Verfügung und alle Änderungen seit der letzten Speicherung sind aufgezeichnet.

Mit dem Button «Pull» lässt sich die letzte Version der Arbeit zurückholen. An dieser Stelle wird ein Restore Prozess mit Sourcetree dokumentiert. Als Alternative wäre es auch möglich, die neuste Version vom Projekt über die Gitlab Webapp herunterzuladen.

Im Sourcetree-GUI den Button «Pull» drücken. Danach erscheint eine Meldung zur Bestätigung. Danach kann das gewünschte File wiederhergestellt werden (siehe Screenshots).

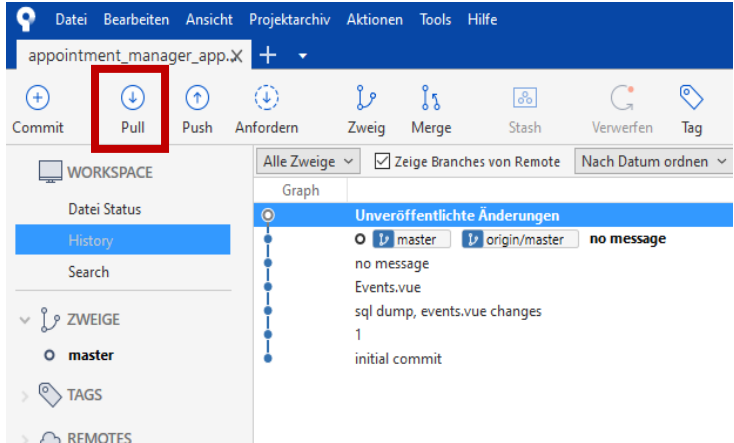
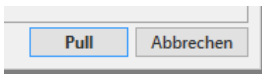
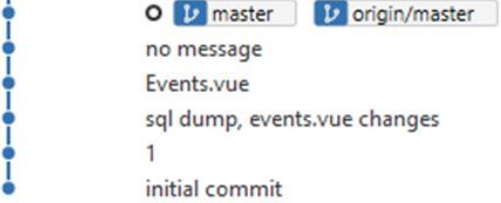
Schritt	Beschreibung	Screenshot
Sourcetree öffnen	Die Sourcetree-Desktop App öffnen. Oben links den Knopf "Pull" drücken.	
Meldung bestätigen	Die sich öffnende Meldung bestätigen mit "Pull".	
Sourcetree überprüfen	Im Sourcetree die Änderungen im Versionsverlauf ansehen. (Wiederhergestellte Datei im nächsten Kapitel)	

Tabelle 7: Wiederherstellung

5.4.3 Wiederhergestellte Entwicklungsdatei

Die heruntergeladene Version wird in der Sourcetree-App angezeigt. Dabei werden auch die letzten Änderungen farblich dargestellt (siehe Grafik unten).

```

1 1  ...<template>
2 2  ...<v-data-table
3 3  ...:headers="headers"
4 4  ...:items="projects"
5 5  ...sort-by="project"
6 6  ...class="elevation-1">
7 7  ...<v-data-table :headers="headers" :items="projects" sort-by="project" class="
8 8  ...<template v-slot:top>
9 9  ...<v-toolbar flat color="white">
10 10 ...<v-spacer></v-spacer>
11 11 ...<v-dialog v-model="dialog" max-width="900px">
12 12 ...<template v-slot:activator="{ on, attrs }">
13 13 ...<v-fab-transition>
14 14 ...<v-btn
15 15 ...color="#b47bff"
16 16 ...v-bind="attrs"
17 17 ...v-on="on"
18 18 ...fab
19 19 ...v-if="button"
20 20 ...dark>
21 21 ...<v-icon>mdi-plus</v-icon>
22 22 ...</v-btn>
    ...</v-fab-transition>
  
```

Abbildung 1: Wiederhergestellte Datei

5.5 Dokumentenablage und Ordnerstruktur

Alle relevanten Dateien, sowohl Code wie auch Dokumentation, werden im Stammordner «ipa_yannis» gesichert. In diesem Verzeichnis befinden sich jeweils Unterordner für Projekt und Dokumentations- Dateien. Im Stammordner befindet sich zusätzlich das Unterverzeichnis /BACKUP/. Es enthält eine Arbeitskopie Dieses Verzeichnis wird über ein Remote-Repository des Cloud-Speichers GitLab gesichert (siehe «Kapitel 5.3: Erg. Datensicherungskonzept»). Es enthält die Dokumentation, sowie den Zeitplan und sämtliche Anhänge.

Der Stammordner «ipa_yannis» enthält folgende Unterordner:

Ordner (Unterordner)	Beschreibung
Dokumentation	Aktuelle Tagesversion der IPA-Dokumentation.
Anhang	Alle Anhänge und Dateien wie Code oder Notizen und Protokolle.
Zeitplan	Versionsverlauf des Zeitplans.
Protokolle	Alle Dateien welche keinem der vorgängigen Ordner zugeordnet werden können.

Tabelle 8: Ordnerstruktur








Name	Änderungsdatum	Typ	Größe
 Projekt	04.03.2021 09:00	Dateiordner	
 Anhang	04.03.2021 09:00	Dateiordner	
 Zeitplan	04.03.2021 09:00	Dateiordner	
 Dokumentation	04.03.2021 09:00	Dateiordner	
 Protokolle	04.03.2021 09:00	Dateiordner	
 Individueller_IPA-Kriterienkatalog_Inform...	27.01.2021 16:40	Adobe Acrobat D...	1.308 KB
 todo.txt	24.02.2021 16:57	Textdokument	1 KB

Abbildung 2: Projektordner

5.6 Arbeitsumgebung

5.6.1 Arbeitsplatz

Der stationäre Arbeitsplatz ist bestmöglich ergonomisch eingerichtet. Er kann aber auch ohne grossen Aufwand an einen anderen Ort verschoben werden, falls dies durch eine besondere Situation nötig wäre. Der Arbeitsplatz befindet sich im «Annex» Gebäude der Technischen Fachschule Bern.

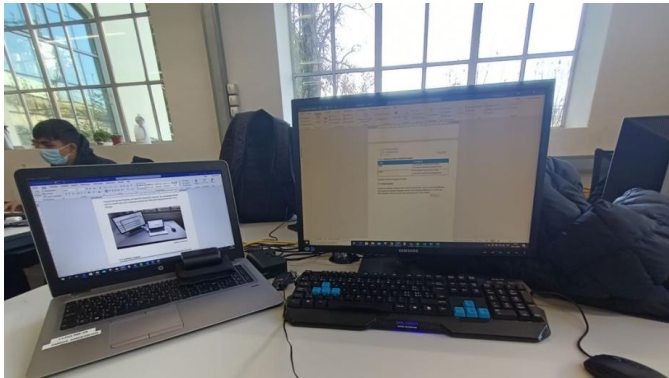


Abbildung 3: Arbeitsplatz

5.6.2 Lokales Arbeitsgerät


Bild	Spezifikationen	Detaillierte Informationen
	Betriebssystem	Windows 10
	Hersteller	Hewlett Packard
	Prozessor und Arbeit Speicher	2,4 GHz Intel Core i7, 16GB DDR3 RAM
	Installierte Entwicklungsumgebung	MS Visual Studio Code

Tabelle 9: Arbeitsgerät

6 Projektvorgehen

Mit welchen Methoden wird das Projekt durchgeführt? Wie werden Probleme systematisch mittels Zuhilfenahme einer Projektmanagementmethode gelöst?

6.1 Projektmethode

Das Projekt wird mit der Hermes 5 Projektmethode durchgeführt. Hierbei handelt es sich um eine Projektmethode nach dem Wasserfall Modell.

Bei HERMES 5 stehen die Ergebnisse im Zentrum. Sie werden mit anderen Methodenelementen wie Aufgaben oder Rollen in Beziehung gebracht und manche Ergebnisse werden bei der Anwendung durch Dokumentenvorlagen ergänzt. Eine eigentliche Visualisierung der Projektprozesse steht nicht im Vordergrund. Die Projektmethode durchläuft vier Phasen (siehe Grafik und Tabelle unten). Es wird das Szenario «IT-Individualanwendung» verwendet.



Abbildung 4: Phasenmodell

6.2 Phasen

Phase	Beschreibung
Initialisierung	Die Initialisierung schafft eine definierte Ausgangslage und stellt sicher, dass alle Projektziele mit denen auf PkOrg übereinstimmen. Die Projektgrundlagen und der Projektauftrag werden erarbeitet.
Konzept	Die Variante, welche in der Initialisierung gewählt wurde, wird präzisiert und weitere benötigte Konzepte werden erstellt. Die Ergebnisse werden so detailliert erarbeitet, dass eine dritte Person sämtliche Schritte nachvollziehen kann.
Realisierung	Das Programmieren der Chat-Applikation wird umgesetzt und die Datenbank erstellt. Testfälle werden durchgeführt, um die Sicherheit und die Zuverlässigkeit des Projektes zu gewährleisten.
Einführung	Das zu erarbeitende Projekt wird anhand der weiterführenden Arbeiten fertiggestellt und den Klassen (inklusive Lehrpersonen) vorgestellt. Das finale Projekt wird dann später einmal auf den Servern der TF Bern gehostet und verwaltet.

Abbildung 5: Phasenbeschrieb

7 Projektorganisation

Wer nimmt welche Rolle bei dem Projekt ein, und welche Personen sind inwiefern beteiligt?

7.1 Organigramm

Das Organigramm zeigt die Rollenverteilung bei dem Projekt auf.

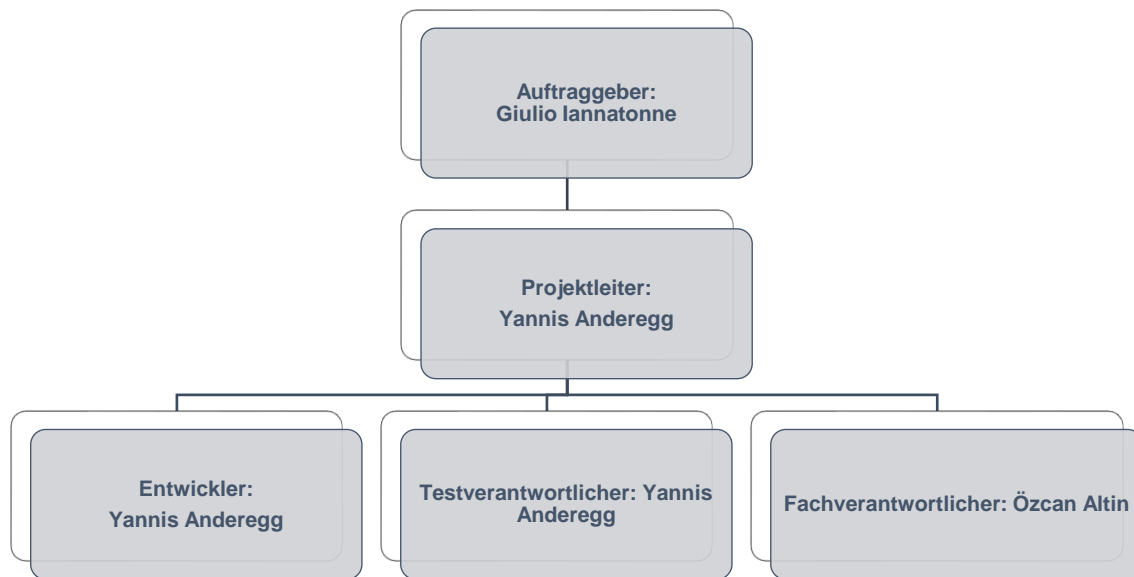


Abbildung 6: Organigramm

7.2 Detaillierte Informationen zu Projektrollen

Die Kontaktdaten aller beteiligten Personen, ergänzend zum Organigramm.

Projektrolle / Person	Tätigkeit	Kontaktdaten
Kandidat	Der Autor dieser Facharbeit.	Yannis Anderegg E-Mail: yannisanderegg@gmail.com Tel.: +41788839880 Firma: Technische Fachschule Bern Jungfraustrasse 83, 3800 Interlaken
Fachspezialist	Die für die technische Entwicklung verantwortliche Person.	
Projektleiter	Verantwortlicher für das Projekt.	
VEX (Valid Experte)	Der Valid Experte validiert die Aufgabenstellung.	Katrin Meyer
HEX (Hauptexperte)	Der Hauptexperte bewertet die Arbeit.	Lorenz Müller E-Mail: lm@nts.ch Tel.: 0793346266
NEX (Nebenexperte)	Ist als Unterstützung für den HEX da.	Daniel Gamez E-Mail: daniel.gamez@swisscom.com Tel.: +41796576385
Auftraggeber	Der Auftraggeber des Projekts.	Technische Fachschule Bern Lorrainestrasse 3 3013 Bern Telefon: 031 337 37 37 E-Mail: info@tfbern.ch
Berufsbildner	Zuständiger Vorgesetzter	Giulio Iannattonne E-Mail: giulio.iannattonne@tfbern.ch
Verantwortliche Fachkraft	Die Ansprechperson im Betrieb.	Özcan Altin E-Mail: oezcan.altin@tfbern.ch Tel.: 0764172626
Produkttester	Die Person welche das Testing durchführt.	Kavindu Jasin Pathiranage E-Mail: kavindu9@hotmail.com

Tabelle 10: Projektroll

8 Risikoanalyse

Die Risikoanalyse dient primär dazu das Projekt vor dem Scheitern zu bewahren. Sie soll die grössten Risiken übersichtlich aufzeigen mit Inbezugnahme von Schadensausmass und Eintrittswahrscheinlichkeit. Für jeden Risikofaktor werden mögliche Massnahmen erarbeitet, welche jenes Risiko minimieren, oder gänzlich eliminieren sollen. Nach den aufgezeigten ergriffenen Massnahmen wird das Schadensausmass erneut kalkuliert.

8.1 Legende

8.1.1 Schadensausmass und Eintrittswahrscheinlichkeit

Schadensausmass Kürzel:	Beschreibung:	Eintrittswahrscheinlichkeit Kürzel:	Beschreibung:
S1	Führt zu keiner Abwertung	W1	Unvorstellbar
S2	Geringe Abwertung	W2	Unwahrscheinlich
S3	Hohe Abwertung	W3	Eher vorstellbar
S4	Führt zu nicht bestehen	W4	Wahrscheinlich
		W5	Sehr wahrscheinlich

Tabelle 11: Schadensausmass & Eintrittswahrscheinlichkeit

8.1.2 Gesamtrisikoindex

Der Gesamtrisikoindex zeigt die Eintrittswahrscheinlichkeit und das Schadensausmass in einem Verhältnis auf. Dadurch kann ein Risiko als gesamtes eingeschätzt werden.

Gesamtrisikoindex:	Beschreibung:
G1	Kleines Risiko
G2	Mittleres Risiko
G3	Hohes Risiko
G4	Sehr Hohes Risiko

Tabelle 12: Gesamtrisikoindex













Nr.	Risikobeschreibung	Auswirkung	Vor Massnahmen		Massnahmen	Nach Massnahmen	
			Schadensausmass	Eintrittswahrscheinlichkeit		Schadensausmass	Eintrittswahrscheinlichkeit
R1	Frontend-Komponente "Kalender" kann nicht implementiert werden	Es entsteht eine Verzögerung oder es kann nicht rechtzeitig abgegeben werden.	S3 	W2	Da der Kandidat den Aufbau des Frameworks bereits kennt, ist es ihm möglich eine äusserst detaillierte Planung zu erstellen und mittels der verwendeten Projektmethode den Ablauf genau zu planen.	S2 	W1
R2	API-Endpunkte schlagen fehl	Verbindung Backend-Frontend kann nicht realisiert werden.	S4 	W2	Um das API bereits in der Entwicklung ausgiebig zu testen, setzt der Kandidat dafür vorgesehene Software ein, wie etwa Postman.	S2 	W1
R3	Login-Funktion kann nicht implementiert werden	Probleme bei der Umsetzung führen zu nichterfüllen der Anforderung «Login».	S3 	W2	Um das konzeptionelle Verständnis zu erlangen, plant er Kandidat sein Vorgehen ausgiebig. Des Weiteren wird er dieses Kapitel ausgiebig im Testkapitel auf alle Anforderungen prüfen.	S2 	W1
R4	Backend-Server kann nicht implementiert werden	Durch Probleme bei der Entwicklung kann der Backend-Server nicht fertiggestellt werden.	S4 	W1	Der Backend-Server wird in der Planung mit den Punkten «API» und «DB» abgeglichen. So wird sichergestellt, dass einzelne voneinander abhängige Komponenten sich nicht gegenseitig beeinträchtigen.	S3 	W1
R5	DB-Abfragen schlagen fehl	Fehler bei den Datenbankabfragen führen dazu, dass das Produkt unbrauchbar wird.	S4 	W2	Die Datenbankabfragen können sicher abgeschlossen werden, da der Kandidat vorher ein in sich schlüssiges Datenbank-Design umgesetzt hat.	S2 	W1

Tabelle 13: Risikoanalyse

9 Risikograph

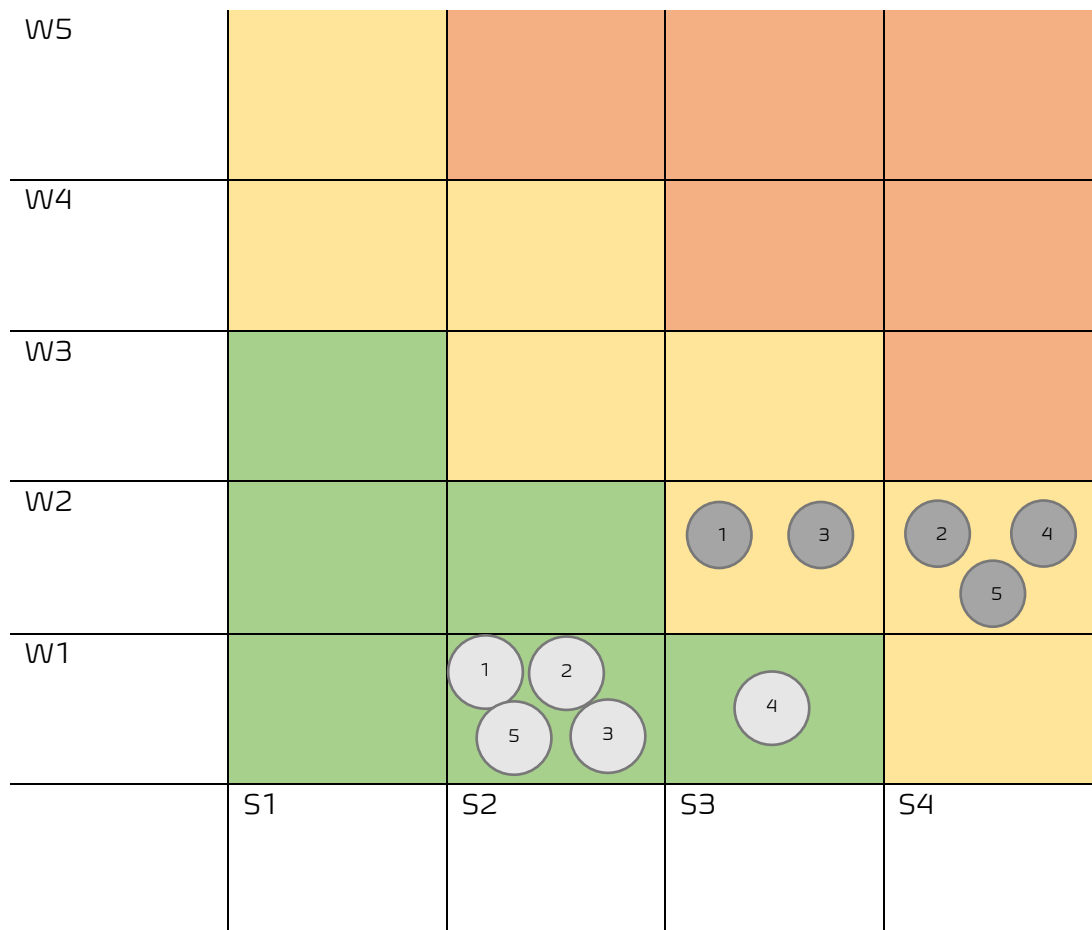


Tabelle 14: Risikomatrix




Risiko vor Massnahmen	Risiko nach Massnahmen
	

Tabelle 15: Legende Risikomatrix

9.1 Informationen zur Risikoanalyse

Als Zusammenfassung zur Risikoanalyse lässt sich sagen, dass sich alle aufgezeigten Risiken massgeblich reduzieren liessen. Anhand des «Gesamtrisikoindex» lässt sich sagen, dass jedes Risiko durch die ergriffenen Massnahmen um eine Stufe senken liess. Die gefährlichsten Risiken sind nach wie vor «R2» und «R5», da hier die Eintrittswahrscheinlichkeit und das Schadensausmass am grössten sind.

Yannis Anderegg

Legende		Nr.	Meilensteine
	Soll Zeit	1.00	Start der IPA
	Ist Zeit	2.00	Phase "Initialisierung" freigegeben
GE	Gesamtzeit	3.00	mit abgeschlossen / Konzept freigegeben
MA	Morgens	4.00	Phase "Relisierung" freigegeben
NM	Nachmittags	5.00	Phase "Relisierung" zum größten Teil abgesc
	Meilenstein	6.00	Phase "Realisierung" abgeschlossen
	Zusatzaufgaben	7.00	IPA auf PKOrg hochgeladen
	Überstunden		

10.1 Meilensteine

Alle im Projekt gesetzten Meilensteine zusammengefasst.

ID	Titel	Erreicht
1	Start der IPA	Ja
2	Phase "Initialisierung" freigegeben	Ja
3	Initialisierung abgeschlossen / Konzept freigegeben	Ja
4	Phase "Realisierung" freigegeben	Ja
5	Phase "Realisierung" zum grössten Teil abgeschlossen	Ja
6	Phase "Realisierung" abgeschlossen	Ja
7	IPA auf PkOrg hochgeladen	Ja

Tabelle 16: Meilensteine

11 Arbeitsjournal

Alle Arbeitstage sowie Halbtage werden kurz beschrieben. Es werden die Punkte Zeitplanung, Arbeitsschritte, Probleme und Lösungen thematisiert.

11.1 Erster Tag, 3. März 2021

Tätigkeiten	Involvierte Personen	Geplanter Aufwand (soll in Std)	Tatsächlicher Aufwand (ist in Std)
<i>Erstversion des Zeitplans</i>	AnY	2	2
<i>Erstversion der Dokumentation</i>	AnY	1	1.5
<i>Kurzfassung der IPA</i>	AnY	1	1
<i>Def. der Standards</i>	AnY	1	1
<i>IPA-Schutzbedarfsanalyse</i>	AnY	1.25	1.25
<i>Zwischengespräch mit VF</i>	AnY, AIÖ	0.25	0.25
<i>Arbeitsjournal & Sicherung</i>	AnY	0.5	0.5
<i>Expertenbesuch und anschl. Reflexion mit VF</i>	AnY, LoM, AIÖ, GaD	1	1
Total:		8	8.5

Tagesablauf:

Das Erste was ich gemacht habe, ist der Zeitplan. Diesen hatte ich bereits am Vortag als Vorlage erstellt. Nachdem ich zeitig fertig wurde, habe ich mich an die Erstversion der Doku gemacht. Die Formalitäten haben hier etwas mehr Zeit in Anspruch genommen als geplant. Bei der «Kurzfassung der IPA» und «IPA-Schutzbedarfsanalyse» hatte ich ein wenig Mühe einzuordnen, was ich schreiben sollte. Ich habe dann die Dokumentationsvorlage 2021 als Hilfestellung genommen.

Danach wurde spontan der erste Expertenbesuch durchgeführt. Ich hatte leider nicht viel Zeit, um mich auf diesen vorzubereiten. Dennoch war es sehr aufschlussreich. Ich konnte einige Korrekturvorschläge für meinen Zeitplan entgegennehmen. Die Zeit wurde schliesslich wegen dem Arbeitsjournal noch etwas knapp. Die Änderungen im Zeitplan beinhalten unter anderem mehr Zeit für das Arbeitsjournal. Dieses konnte ich heute zeitbedingt nicht ganz fertigstellen.

Hilfestellungen:

-Dokumentationsvorlage 2021 (Quelle 9)

Reflexion:

Gut:

Ich bin etwas unsicher in die IPA gestartet, doch habe dann schnell meinen Rhythmus gefunden. Ich habe das Gefühl, dass ich mich schnell an die neue Arbeitsmethodik gewöhnen werde.

Schlecht:

Bei einigen Teilen der heutigen Arbeit fiel es mir schwer die richtigen Worte zu finden und nur das Relevante aufzuschreiben und mich so an das Kriterium «Prägnanz» zu halten. Die Dokumentationsvorlage hat mir sehr geholfen. Für die Zukunft werde ich mich mehr auf diese beziehen, da ich vermutlich noch oft auf dieses Problem treffen werde. Nachdem Expertengespräch werde ich meine Zeitplanung teilweise überarbeiten.

Nächste Schritte:

- Risikoanalyse
- IST & SOLL Analyse
- Neuer verbesserter Zeitplan einreichen
- Ziele definieren
- Anforderungen definieren

Tabelle 17: Arb. Journal Tag 1



11.2 Zweiter Tag, 4. März 2021

Tätigkeiten	Involvierte Personen	Geplanter Aufwand (soll in Std)	Tatsächlicher Aufwand (ist in Std)
<i>Organisation der IPA-Ergebnisse</i>	AnY	0.5	0.5
<i>Projektvorgehen</i>	AnY	0.5	0.5
<i>Projektorganisation</i>	AnY	1	1
<i>Projekttrollen und Kontaktangaben</i>	AnY	2	2
<i>Feinschliff / Verbesserungen / Ergänzungen</i>	AnY	0.5	0.5
<i>Ist-Situation</i>	AnY	2	2
<i>Soll- Situation</i>	AnY	1	1.5
<i>Arbeitsjournal & Sicherung</i>	AnY	0.5	0.5
Total:		8	8.5

Tagesablauf:

Heute habe ich als erstes den überarbeiteten Zeitplan sowie mein erstes Arbeitsjournal eingereicht. Die Kapitel «Organisation der IPA-Ergebnisse» und Projektvorgehen, bzw. Projektorganisation nahmen nicht allzu viel Zeit in Anspruch. Bei der «Organisation der IPA-Ergebnisse» hatte ich deutlich mehr Aufwand als geplant. Ich habe dafür nur eine halbe Stunde eingeplant. Jedoch konnte ich es in der geplanten Zeit abschliessen, da ich mir bereits gestern Abend Gedanken darüber gemacht habe, wie ich das am besten umsetze. Ich musste dann nur noch die Fleissarbeit erledigen und ein paar Screenshots und Tabellen einfügen. Danach habe ich mit der Initialisierung angefangen. Ich habe die IST-Situation grosszügig geplant, von der Zeit her. Bei der Soll-Situation habe ich deutlich zu wenig Zeit eingeplant. Das Beschreiben eines nicht vorhandenen Zustandes hat sehr viel Zeit in Anspruch genommen. Ich hatte für den SOLL-Zustand fast doppelt so lange wie für den IST-Zustand. Am Ende hat sich das aber gegenseitig ein wenig kompensiert. Ich habe den SOLL-Zustand mit einer **halben Überstunde** kompensiert.

Hilfestellungen:

-

Reflexion:

Gut:

Die heutige Methodik bei der «Organisation der IPA-Ergebnisse» hat sich als sehr praktisch und effizient erwiesen. Am Vortag oder Vorabend ein Kapitel im Kopf bereits gänzlich durchzuplanen werde ich von nun an immer machen. Wenn man bereits weiß, was man schreiben möchte, geht die Arbeit deutlich schneller. Ich

werde von nun an eine Todo-Liste führen, und diese mit den «nächsten Schritten» aus den Arbeitsjournalen abgleichen. So habe ich immer den Überblick was als nächstes ansteht. Ich denke, dass ich im Verlauf der Arbeit noch öfter meine Methodenkompetenz anpassen werde. Diese Arbeitsweise ist etwas ganz Neues für mich.

Schlecht:

Ich hätte meine Ressourcen besser Einteilen sollen, für den Soll- und Ist-Zustand. Hätte ich dieses Kapitel in der Planung besser durchdacht, wäre ich wohl zum Schluss gekommen, dass der «Soll-Zustand» deutlich mehr Aufwand ist als der «Ist-Zustand».

Nächste Schritte:

- Initialisierung beenden
- Risikoanalyse
- > Grafiken machen
- Konzeptphase
- >Mockups

Tabelle 18: Arb. Journal Tag 2

11.3 Dritter Tag, 5. März 2021

Tätigkeiten	Involvierte Personen	Geplanter Aufwand (soll in Std)	Tatsächlicher Aufwand (ist in Std)
<i>Persönliche Vorgehensziele</i>	AnY	1	1
<i>Projektziele</i>	AnY	1	1
<i>Anforderungen</i>	AnY	2.5	2,5
<i>Risikoanalyse</i>	AnY	2.5	3
<i>Grafiken erstellen / Reserve</i>	AnY	0.5	0.5
<i>Arbeitsjournal & Sicherung</i>	AnY	0.5	0.5
Total:		8	8.5

Tagesablauf:

Heute habe ich meine Ziele für das Projekt definiert, was nicht allzu schwierig war. Viel schwieriger war es die Anforderungen, welche ich danach definiert habe, von den Zielen zu unterscheiden. Ich hatte die ganze Zeit das Gefühl, das ich Redundanzen habe. Als ich damit fertig war, habe ich mich mit der Risikoanalyse beschäftigt. Die eingeplante Zeit konnte ich unterbieten, da ich einen Weg gefunden habe für die Risikomatrix nur eine Tabelle zu machen. Die dadurch gesparte Zeit brauchte ich, um einige Formatierungen zu machen. Die übrige Zeit konnte ich dafür einsetzen, die anstehende Konzeptphase ein wenig vorzuplanen. Dazu habe ich mit Diagrams.net ein geeignetes Tool gefunden, um Grafiken etc. anzufertigen.

Hilfestellungen:

<https://app.diagrams.net/> (Quelle 2)

Reflexion:

Gut:

Heute war meine Zeitplanung sehr schlüssig. Ich hatte sogar noch genug Zeit die nächsten Schritte genau zu planen. Der Konzeptphase steht nun nichts mehr im Weg.

Schlecht:

Da beim ersten Expertenbesuch eine zu kurz geplante Konzeptphase bemängelt wurde, ist es entscheidend diese zeitig fertigzustellen. Ich hätte mir jedoch schon vorgängig Gedanken zur Umsetzung der Konzeptphase machen sollen. Ich denke, dass ich morgen nicht direkt starten kann, sondern mein genaues Vorgehen erst planen muss. Dies kostet mich wertvolle Zeit und hätte auf jeden Fall vermieden werden sollen.



Nächste Schritte:

- Konzeptphase beginnen
- >Vorgehen planen
- > Systemgrafiken, Mockups
- Zwischengespräch mit VF

Tabelle 19: Arb. Journal Tag 3

11.4 Vierter Tag (Halbtag), 8. März 2021

Tätigkeiten	Involvierte Personen	Geplanter Aufwand (soll in Std)	Tatsächlicher Aufwand (ist in Std)
<i>Grafiken erstellen & Reserve</i>	AnY	2	2
<i>Namenskonzept erstellen</i>	AnY	0.5	0.5
<i>Modell der Datenbank erstellen (ERM)</i>	AnY	1	1
<i>Arbeitsjournal & Sicherung</i>	AnY	0.5	0.5
Total:		4	4

Tagesablauf:

Am heutigen Halbtag habe ich unter anderem einige Systemgrafiken erstellt. Ich habe die Architektur der Applikation im Rahmen der Konzeptphase mit einer Grafik beschrieben. Ich hatte einige Probleme das Ganze grafisch übersichtlich darzustellen. Ich habe dann in älteren Schulunterlagen recherchiert. Danach stand das Namenskonzept auf dem Plan. Jedoch wurde mir im weiteren Verlauf klar, dass ich zu diesem Thema nicht wirklich viel sagen kann. Ich habe dann die übrig gebliebene Zeit genutzt, um an dem ersten ERM-Entwurf zu arbeiten. Beim ERM ist mir in den Sinn gekommen, dass ich viel Zeit sparen könnte, wenn ich das ERM in der Workbench Applikation erstellen würde, da ich dieses dann mittels Reverse-Engineerings zu einem SQL-Script übersetzen könnte. Ich habe mir auch schon einige Gedanken zu der Skalierung gemacht. Dabei habe ich kurz im Netz recherchiert wie die Limitierungen bei einer SQL-Datenbanken sind (siehe Hilfestellungen). Da der Tag nur aus einem Halbtag besteht, konnte ich keine riesigen Fortschritte erzielen, aber meine Zeitplanung konnte ich einhalten.

Hilfestellungen:

-SQL-Workbench Einstellungen ändern (Quelle 6)

Reflexion:

Gut:

Es ist heute das erste Mal vorgekommen, dass ich zu bestimmten Themenbereichen richtig intensive Recherche betreiben musste. Ich denke, dass dies jetzt bald noch öfters der Fall sein wird, da ich jetzt in Richtung Konzept/Initialisierung gehe. Für diese Arbeitsmethodik muss ich meine Quellen sauber dokumentieren. Dafür habe ich die Routine erarbeitet, dass ich zum Beispiel Quellen aus dem Internet in einem Session Saver speichere, so dass ich diese am Ende des Tages in mein Arbeitsjournal und in das Quellenverzeichnis aufnehmen kann. Dadurch wird eine lückenlose Dokumentation von Quellen sichergestellt. Des Weiteren werde ich aktuelle Themen, welche mich beschäftigen in einem Dokument zusammenfassen. Dort referenziere ich auf die

Quellen und trage entsprechende Informationen zusammen, welche ich später noch brauchen könnte.

Schlecht:

Ich hatte heute leider einige Probleme mit meiner Datenbank-Software (MySQL Workbench). Diese ist mehrmals abgestürzt und hat mich so einiges an Zeit gekostet. Ich werde die IDE neu installieren und gründlich auf ihre Funktionalität testen, bevor ich diese erneut einsetze.

Nächste Schritte:

- Konzeptionierung
- >Testkonzept
- >Mockups (diagrams.net)
- Zweiter Exptertenbesuch (Remote)

Tabelle 20: Arb. Journal Tag 4

11.5 Fünfter Tag, 10. März 2021

Tätigkeiten	Involvierte Personen	Geplanter Aufwand (soll in Std)	Tatsächlicher Aufwand (ist in Std)
<i>Modell der Datenbank erstellen (ERM)</i>	AnY	1.5	1.5
<i>Mock-ups</i>	AnY	2	2
<i>ISDS-Konzept</i>	AnY	1	1
<i>Testkonzept erstellen</i>	AnY	3	3.5
<i>Arbeitsjournal & Sicherung</i>	AnY	0.5	0.5
Total:		8	8.5

Tagesablauf:

Am heutigen Tag habe ich anfangs noch das ERM vollendet. Danach habe ich an den Mock-Ups gearbeitet. Diese waren nicht besonders schwer zu erstellen. Die einzige Herausforderung dabei war es sich in die Perspektive eines Users zu versetzen. Ich habe versucht die Mock-Ups so zu erstellen, dass ich diese in der Realisierung möglichst ähnlich umsetzen kann. Als nächstes habe ich ein ISDS-Konzept für das Projekt erarbeitet. Im Verlauf der Arbeit wurde mir jedoch klar, dass ich nicht viele schützenswerte Daten habe. Ich habe dann das Kapitel ISDS-Konzept in Passwortsicherheit umbenannt, da dies die einzigen wirklich schützenswerten Daten sind. Ich wurde trotzdem zeitig fertig mit dem ISDS-Konzept. Im Anschluss habe ich mich der Hauptaufgabe des Tages gewidmet, nämlich dem Testkonzept. Dieses zu konzeptionieren war die bis jetzt grösste Herausforderung, da es sehr schwierig ist die Tests genau aus der Sicht eines Users zu beschreiben. Man hat noch keine Applikation, also muss man hier sehr abstrakt denken. Ich habe schliesslich eine halbe Stunde überzogen und hab dies als **Überstunde** deklariert. Das Testkonzept werde ich unbedingt noch am zweiten Expertengespräch zur Sprache bringen. Ich denke, dass hier noch Verbesserungspotenzial besteht.

Hilfestellungen:

- VF kurz zum Testkonzept befragt
- ERM-, und allgemeine Infos zu Datenbanken
- > <https://phprog.ch/entity-relationship-modell-erm/>

Reflexion:

Gut:

Ich konnte heute sehr konzentriert arbeiten. Die neu eingeführte Methodik (genaues Planen der nächsten Arbeitsschritte am Vortag) hat sich wieder bezahlt gemacht. Ich musste fast keine Zeit mit Nachdenken verschwenden, sondern konnte direkt das umsetzen was ich am Vortag geplant habe. Ein weiteres Merkmal der heutigen Arbeitsmethodik war, dass ich bei etwas stockender oder langsamer Arbeitsweise zum nächsten (oder letzten) Kapitel vorgerückt bin.

Dadurch habe ich keine Zeit verloren, sondern konnte jeweils nach einer gewissen Zeit mit einem neuen Blickwinkel und Vorgehensweise weiterarbeiten. Gerade beim Testkonzept hat mir das heute sehr geholfen. Ich denke, dass sich diese Arbeitsweise vor allem in der Realisierung sehr bewähren wird.

Schlecht:

Ich war sehr unsicher was das Testkonzept angeht. Es war mir nicht sofort klar, wie ich die Informationen am besten sauber darstellen könnte. Ich hätte hier schon während der Definition der Ziele und Anforderungen über die Umsetzung des Testkonzepts nachdenken müssen. Ich konnte dann nach einem kurzen Input meiner VF weitermachen.

Nächste Schritte:

- Testkonzept überarbeiten (wenn nötig)
- >am Expertengespräch zur Sprache bringen!
- Anwendungsfälle definieren (Konzeptphase)

Tabelle 21: Arb. Journal Tag 5

11.6 Sechster Tag, 11. März 2021

Tätigkeiten	Involvierte Personen	Geplanter Aufwand (soll in Std)	Tatsächlicher Aufwand (ist in Std)
<i>Einrichten der Projektumgebung</i>	AnY	1	1
<i>Erstellen der Datenbank (SQL-Scripts)</i>	AnY	3	3
<i>Aufbau der Basisstruktur des Projektes</i>	AnY	1	1
<i>Aufbau des Backends mit REST-API</i>	AnY	2.5	3
<i>Arbeitsjournal & Sicherung</i>	AnY	0.5	0.5
Total:		8	8.5

Tagesablauf:

Am heutigen Tag habe ich mit der Realisierung begonnen. Ich habe als erstes die Projektumgebung beschrieben. Dies ging sehr einfach, da ich praktisch einfach nur die Kommandozeilenbefehle, die ich gemacht habe in meine Doku übertragen konnte und diese kurz beschrieben habe. Etwas schwieriger war die Arbeit an der Datenbank. Ich wusste durch die Schritte in der Konzeptphase zwar schon recht genau welche Tabellen mit welchen Relationen es geben wird, jedoch hatte ich einige Schwierigkeiten mit meiner Entwicklungssoftware (MySQL Workbench). Diese ist im Verlauf des Tages mehrmals abgestürzt und hat mich so zeitlich etwas aufgehalten. Dennoch konnte ich das Kapitel noch zeitig beenden. Danach habe ich kurz den Aufbau und Struktur der Komponenten beschrieben. Dies fiel mir nicht schwer, da ich hier schon ein gutes Vorwissen besitze. Die restliche Zeit habe ich damit verbracht mein REST-API zu entwickeln. Hier konnte ich einen ordentlichen Fortschritt erzielen, da die meisten Endpoints schon klar waren (diese wurden teilw. schon im «Use-Case Diagramm beschrieben). Morgen werde ich dieses Kapitel noch abschliessen. Um den Teil mit dem REST-API noch rechtzeitig zu beenden, wurde eine halbe **Überstunde** gemacht.

Hilfestellungen:

- Informationen zu Knex.js im Zusammenhang mit dem Aufbau der Datenbank (Quelle 4)
-> <https://devhints.io/knex>
- Axios Abfrage (Request ans Backend senden) (Quelle 7)
-> <https://stackoverflow.com/questions/51415439/how-can-i-add-raw-data-body-to-an-axios-request>

Reflexion:



Gut:

Heute war ich zwar etwas müde, aber konnte trotzdem viele Kapitel abschliessen. Ich bin nun sehr zufrieden das ich endlich in der Realisierung angelangt bin.

Schlecht:

Ich habe jedoch bemerkt, dass sich nun auch die Arbeitsmethodik etwas ändern muss. In der Realisierung geht es darum laufend zu dokumentieren was man gerade macht. Dies ist ein stetiger Prozess, welcher nicht vernachlässigt werden darf. Wenn man sich jedoch richtig in ein praktisches Problem verbissen hat, will man es lösen, anstatt es zu dokumentieren. Ich werde versuchen von nun an, laufend meine Ergebnisse zu dokumentieren.

Nächste Schritte:

- Realisierung
- >REST-API
- >Frontend
- Zweites Expertengespräch
- >Fragen stellen

Tabelle 22: Arb. Journal Tag 6

11.7 Siebter Tag, 12.März 2021

Tätigkeiten	Involvierte Personen	Geplanter Aufwand (soll in Std)	Tatsächlicher Aufwand (ist in Std)
<i>Aufbau des Backends mit REST-API</i>	AnY	1.5	1.5
<i>Aufbau des Frontends - Landinpage</i>	AnY, ÖzA	3	4
<i>Aufbau des Frontends - Kalenderfunktionen</i>	AnY	1.5	1.5
<i>Datenzugriff mittels HTTP-Methoden</i>	AnY	0.25	0.25
<i>Datenfluss zwischen Views</i>	AnY	0.25	0.25
<i>Zweiter Expertenbesuch</i>	AnY, LoM, AlÖ, GaD	1	1
<i>Arbeitsjournal & Sicherung</i>	AnY	0.5	0.5
Total:		8	9

Tagesablauf:

Am heutigen Tag war ich weiterhin mit der Realisierung beschäftigt. Ich habe vor allem am Frontend gearbeitet. Ich konnte die «Termine»-Seite in einem ersten Design-Entwurf beenden. Ich habe eine Problemstellung beschrieben, die ich hatte. Zu dieser musste ich meinen VF kurz befragen. Dazwischen habe ich noch ein paar Grafiken angefertigt, um das Verständnis zu erleichtern. Ich habe ausserdem an dem REST-API gearbeitet. Dieses ist in seiner Grundform vollendet. Am Nachmittag hatte ich dann noch das zweite Expertengespräch. In diesem konnten einige wichtige Fragen geklärt werden (siehe zweites Sitzungsprotokoll). Am nächsten Tag kann ich an die heutige Arbeit anknüpfen und die Views grösstenteils vollenden.

Hilfestellungen:

- VF zu Themenkomplex «Elemente in Array sortieren und löschen in einer Methode»
- Sequenzdiagramme mit integrierten if-Abfragen (Quelle 5)
<https://circle.visual-paradigm.com/selection-loops-combination/>

Reflexion:

Gut:

Ich habe den Aufbau der Views sehr schlüssig dokumentiert. Ich konnte mithilfe einiger Grafiken eine klare Übersicht schaffen.

Schlecht:

Ich hatte Heute ein Problem mit meiner Zeitplanung. Ich wollte mich, wie bisher, an meinen Zeitplan halten, und so die Kapitel in der vorgesehenen Zeit abarbeiten. Dies hat sich jedoch nun in der Realisierung als sehr schwierig herausgestellt. Weil einige Kapitel in meiner Zeitplanung in der Reihenfolge fehlplatziert waren, konnte ich heute nicht genau nach Zeitplanung arbeiten. Beispielsweise habe ich zuerst Zeit für den Aufbau des REST-Apis eingeplant, obwohl der Aufbau der Views mehr Sinn gemacht hätte.

Ich hatte eine sehr schwierige Problemstellung und musste das erste Mal Hilfe meiner VF in Anspruch nehmen. Ich habe die Problemstellung jedoch sehr gut dokumentiert.

Nächste Schritte:

- Daten der Views sortieren
- Datenfluss (Schnittstellen, Login etc.)

Tabelle 23: Arb. Journal Tag 7

11.8 Achter Tag (Halbtag), 15. März 2021

Tätigkeiten	Involvierte Personen	Geplanter Aufwand (soll in Std)	Tatsächlicher Aufwand (ist in Std)
<i>Datenfluss zwischen Views</i>	AnY	0.5	0.5
<i>Sortierfunktion der Daten</i>	AnY	3	3
<i>Arbeitsjournal & Sicherung</i>	AnY	0.5	0.5
Total:		4	4

Tagesablauf:

Der heutige Halbtag ging sehr schnell vorbei. Ich habe mich zuerst mit den Views beschäftigt. Leider hat das Kapitel mehr Aufwand erzeugt als geplant. Danach habe ich, die in der Realisierung wohl die anspruchsvollste, Aufgabe mit Sortierfunktionen angefangen. Ich bin nicht sehr weit gekommen, hier muss ich wohl noch weitere Zeit einplanen. Die Aufgabe ist sehr schwierig in der Umsetzung, da die Zusammenhänge recht komplex sind. Morgen werde ich dieses Kapitel hoffentlich abschliessen.

Hilfestellungen:

- Java Script Array bearbeiten (Quelle 8)
https://www.w3schools.com/jsref/jsref_copywithin.asp

Reflexion:

Gut:

Ich konnte sehr schnell über die angegebene Quelle zu den benötigten Informationen kommen. Die verwendete Quelle kann auch in Zukunft noch weiterverwendet werden. Ich bin sehr zufrieden das ich hier nicht viel Zeit mit suchen verschwendet habe.

Schlecht:

Der heutige Themenkomplex war sehr anspruchsvoll. Ich hatte das Problem, dass der kurze Halbtag nicht genug Zeit war, um mich richtig intensiv mit der Problemstellung auseinanderzusetzen. Ich hätte dies in der Zeitplanung berücksichtigen sollen. Es wäre optimaler gewesen, an den besagten Halbtagen eher administrative Aufgaben zu erledigen. Es fällt mir etwas schwer, meine Arbeitsschritte in der Realisierungsphase genau zu beschreiben. Ich versuche, Zusammenhänge mit Grafiken zu erklären und wie ein Aussenstehender zu denken.

Nächste Schritte:



- Abschluss Sortierfunktion in den Views
- Backend mit DB-Querys
- Dokumentation «Realisierung» vollenden

Tabelle 24: Arb. Journal Tag 8

11.9 Neunter Tag, 17. März 2021

Tätigkeiten	Involvierte Personen	Geplanter Aufwand (soll in Std)	Tatsächlicher Aufwand (ist in Std)
<i>Sortierfunktion der Daten</i>	AnY	0.5	0.5
<i>Login-Funktion</i>	AnY	2.5	2.5
<i>Zugriffs-Barriere</i>	AnY	1.5	1.5
<i>Testdaten bereitstellen</i>	AnY	0.5	0.5
<i>Testkonzept durchführen</i>	AnY	2.5	3
<i>Arbeitsjournal & Sicherung</i>	AnY	0.5	0.5
Total:		8	8.5

Tagesablauf:

In den heutigen Tag bin ich mit der Sortierfunktion der Daten gestartet. Hier habe ich an die letzte Arbeit angeknüpft und das Kapitel beendet. Danach habe ich noch an der Technischen Seite des Logins gearbeitet. Ich habe die Zugangsbarriere realisiert, welche die Schüler- von der Lehreransicht trennt. Den Rest des Tages habe ich damit verbracht das Durchführen der Tests vorzubereiten und diese gleich im Anschluss durchzuführen. Dies hat etwas länger als geplant gedauert (**30min Überstunde**). In der Realisierung ist das Ende greifbar und es fehlen nur noch einzelne Details. Ich habe kurz meinen VF zu einigen Kriterien befragt (siehe Quelle).

Hilfestellungen:

-VF zu Kriterien «Systemumfeld» und «Testkonzept»

Reflexion:

Gut:

Das gut verlaufene Testkonzept hat mir Aufschwung gegeben. Dieses durchzuführen war nicht allzu schwierig, da ja alles schon vorgegeben ist und nur ausgefüllt werden musste. Auch das Implementieren der Zugriffsbarriere hat sich als nicht allzu schwer herausgestellt. Der schwierigste Teil heute war es das Testprotokoll genau nach Vorgabe akribisch durchzuführen. Die Arbeitsmethodik genau nach Protokoll vorzugehen, hat mir hier sehr geholfen.

Schlecht:

Ich komme langsam in den Stress, so kurz vor dem Abgabedatum. Kleinere Arbeiten summieren sich und belasten meine Zeitplanung. Hier wäre eine bessere Verteilung über die gesamte Arbeit viel effizienter gewesen.

Nächste Schritte:



-
- Realisierung Feinschliff
 - >Kapitel «Backend» überarbeiten & beenden
 - Dokumentation durchlesen & korrigieren

Tabelle 25: Arb. Journal Tag 9

11.10 Zehnter Tag, 18. März 2021

Tätigkeiten	Involvierte Personen	Geplanter Aufwand (soll in Std)	Tatsächlicher Aufwand (ist in Std)
<i>Verbesserungen implementieren</i>	AnY	3.5	3.5
<i>Dok. Realisierung & Kurzfassung des IPA-Berichts</i>	AnY	4	4.5
<i>Arbeitsjournal & Sicherung</i>	AnY	0.5	0.5
Total:		8	8.5

Tagesablauf:

Heute habe ich meine Doku auf Fehler geprüft. Ich habe die Inhalts- Tabellen- und Glossarverzeichnisse aktualisiert. Danach habe ich noch andere Unstimmigkeiten wie Rechtschreibfehler ausgemerzt. Im Anschluss habe ich den letzten Teil der Realisierung abgeschlossen, bzw. dokumentiert. Ich habe eine erste Version der «Kurzfassung des IPA-Berichts» geschrieben. Ich habe vor allem für den letzten Teil der Realisierung sehr lange gebraucht und musste deswegen eine halbe **Überstunde** machen.

Hilfestellungen:

Reflexion:

Gut:

Ich bin relativ zufrieden mit dem Endresultat von Doku und Code. Ich bin heute sehr weit gekommen, besonders was kleine Verbesserungen angeht wie Rechtschreibung. Dabei bin ich methodisch vorgegangen und habe falsche Wörter im Dokument gesucht, um Folgefehler zu vermeiden.

Schlecht:

Die Zeitplanung wurde gegen Ende sehr knapp, da ich noch mehr als erwartet in der Realisierung dokumentieren musste. Der Grund dafür war ein gestriges Fehlverhalten bezüglich der «nächsten Schritte». Hätte ich hier mehr Zeit investiert, hätte ich meine Zeiteinteilung heute entsprechend anpassen können.

Nächste Schritte:

- Source Code in Anhang hochladen
- Finale Version der Dokumentation vorbereiten
- Abschlussbericht fertigschreiben

Tabelle 26: Arb. Journal Tag 10

11.11 Elfter Tag, 19. März 2021

Tätigkeiten	Involvierte Personen	Geplanter Aufwand (soll in Std)	Tatsächlicher Aufwand (ist in Std)
<i>Abschlussbericht</i>	AnY	3.5	3.5
<i>Finale Version der IPA hochladen</i>	AnY	4	4.5
<i>Arbeitsjournal & Sicherung</i>	AnY	0.5	0.5
Total:		8	8.5

Tagesablauf:

Am letzten Tag habe ich praktisch nur noch Verbesserungen gemacht. Ich habe viele kleinere Fehler korrigiert und Formatierungsoptionen erledigt. Ich hatte leider ein Problem mit Word, da dieses abgestürzt ist. Ich hatte glücklicherweise keinen Datenverlust. Danach habe ich noch meine Abbildungs- und Tabellenverzeichnisse aktualisiert. Ich habe die ganze Dokumentation noch einmal durchkontrolliert. Ich habe noch die letzte Überstunde gemacht um die finale Version der IPA kontrolliert.

Hilfestellungen:

Reflexion:

Gut:

Es wurde gegen Ende sehr stressig. Ich habe jedoch ein Gefühl der Erleichterung gespürt als ich endlich abgegeben konnte.

Schlecht:

Ich habe heute zwar methodisch gesehen sehr gut gearbeitet, doch ich wurde durch den Absturz von Word aufgehalten, und geriet dadurch in Stress.

Nächste Schritte:

Tabelle 27: Arb. Journal Tag 11

12 Abschlussbericht

Eine Reflexion und ein Rückblick zu den Projektzielen und einem IST/SOLL Vergleich.

12.1 Vergleich IST/SOLL

Alle konzeptionierten Anforderungen sowie Projektziele wurden erreicht. Die Testfälle wiesen keine höheren Mängel als Mängelklasse M1 auf und somit kann man das Testing durchaus als erfolgreich bezeichnen.

Die konzeptionierten Mockups konnten treffend in ein reales Design überführt werden.

Im Endergebnis ist eine Applikation entstanden, welche den gestellten Anforderungen gerecht wurde. Die Zeitplanung hätte sicher noch deutlich mehr Potenzial aufgewiesen, jedoch hat das Endergebnis nicht sonderlich darunter gelitten.

12.2 Persönliches Fazit und Schlussreflexion

Ich bin zufrieden mit dem Projekt. Auch wenn das Zeitmanagement nicht immer optimal war, finde ich das Endergebnis ist durchaus repräsentativ für meine Fachkompetenz. Die Dokumentation zu erstellen war eine der Kernschwierigkeiten. Dies lag vor allem daran, dass es schwierig war, die Doku aus einem externen Blickwinkel zu sehen. Ich war mir oft unsicher ob ich Zusammenhänge verständlich genug erklärt habe.

Die Realisierung hat mir aber Spass gemacht. Ich habe in dieser Zeit so viel gelernt wie nie zuvor. Die Erfahrung war äusserst lehrreich und ich werde die gelernten Prozesse von nun an in alle meine Projekte fest integrieren.

Obwohl die Arbeitsmethodik etwas vollkommen Neues für mich war, konnte ich von dieser erheblich profitieren. Ich hoffe das die Applikation noch weiterentwickelt wird und eines Tages im produktiven Umfeld den Arbeitsalltag vieler Personen erleichtern wird.

13 Selbstständigkeitserklärung

Die lernende Person bestätigt mit ihrer Unterschrift diese IPA aus Eigenleistung erbracht und nach den Vorgaben der Prüfungskommission Informatik Kanton Bern erstellt zu haben. Die Angaben im Arbeitsjournal entsprechen dem geleisteten Arbeitsaufwand.



Name	Unterschrift
Yannis Anderegg	
Özcan Altın	

Tabelle 28: Unterschriften

14 Phasenfreigabe Initialisierung


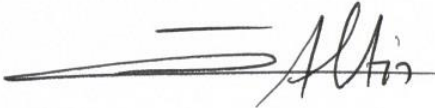
Name	Unterschrift
Yannis Anderegg	
Özcan Altın	

Tabelle 29: Unterschriften



Teil 2: Projektdokumentation



15 Einführung

Innerhalb des Lerninstitutes Technische Fachschule Bern Abteilung Informatik gibt es einige repetitive Aufgaben, die von den Schülern jede Woche erledigt werden. Dazu gehören interne Prozesse, wie der «Treffpunkt Mittwoch» oder das wöchentliche Putzen der Räumlichkeiten. Der «Treffpunkt Mittwoch» findet jede Woche statt. Hierbei handelt es sich um eine Art Schnuppertag, an dem meist Oberstufen-Schüler zusammen mit ihren Eltern teilnehmen. Der «Treffpunkt Mittwoch» wird von jeweils zwei Lernenden der TF Bern begleitet. Es gibt noch viele weitere Beispiele von solchen wöchentlichen Aufgaben. Etwa das Reinigen der Mensa, bei dem jeweils drei Lernende am Mittwoch oder Donnerstag zugeteilt sind. Jeder Schüler der Abteilung Informatik ist in der Regel mindestens einer solchen Aufgabe zugeteilt. Die Anzahl an zu erledigenden Aufgaben nimmt jedes Jahr zu. Damit steigt jedoch auch der Aufwand für Schüler und Lehrer beträchtlich.

Des Weiteren liegt es im Bereich des Möglichen, dass eine ähnliche Applikation für andere Zwecke, wie etwa Gruppenarbeiten mit Zuteilungen oder Präsentationsterminen benutzt werden könnte. Hierfür könnte eine abgeänderte Version der Applikation verwendet werden. Für eine Kalender-App gibt es Dutzende mögliche Einsatzbereiche in der Zukunft.

Für diese erwähnten repetitiven Aufgaben existieren im Moment durch die Lehrkräfte erstellte Einsatzpläne aus Papier oder im Excel Format. Die manuell erstellten, teilweise komplizierten und unübersichtlichen Pläne sollen durch eine modernere und intuitivere Lösung ersetzt werden, die sowohl dem Lehrkörper wie auch den Lernenden den ganzen Prozess erleichtern soll.

Eine solche neue Lösung könnte klassenübergreifend, oder zu einem späteren Zeitpunkt sogar abteilungsübergreifend eingesetzt werden. Die Applikation wird in der Zukunft auf dem TF Bern Intranet gehostet. So wird sie dem ganzen Nutzerkreis vollumfänglich zugänglich gemacht. Dieser Teil liegt jedoch noch in der Zukunft, und ist nicht Teil der Arbeit.

16 Initialisierung

Die Initialisierungsphase bildet als Vorläufer der Konzeptphase die Grundlage einer erfolgreichen Projektplanung. Sie beinhaltet einen momentanen Situationsbeschreibung (im Folgenden genannt IST-Situation).

Daraus wird die SOLL-Situation beschrieben. Des Weiteren werden die Ziele definiert, sowie ein oder mehrere Varianten Entscheide gefällt (falls nötig). Am Ende dieses Prozesses steht das Evaluieren von funktionalen- sowie nicht funktionalen Anforderungen an das Projekt.

16.1 IST-Situation Beschrieb

Die momentane Lösung für regelmässige wöchentliche Arbeiten, sind meist Excel-Sheets in ausgedruckter oder digitaler Form. Diese werden von Hand erstellt und sind oft nicht auf dem neusten Stand. Wenn etwa ein Lernender nicht länger an der TF Bern beschäftigt, oder jemand krank / abwesend ist, dann findet sich derjenige trotzdem noch in dem Excel-File. Solche und ähnliche Situationen sind schon oft vorgekommen, und sorgen meist für grosse Verwirrung. Im Folgenden wird genauso ein Einsatzplan etwas genauer beschrieben.

Reinigungsplan BINF18 (Freitag)		Kalenderwoche									
Nachname	Vorname	32	34	35	36	37	38	39	40	41	
Ferrari	Noah						x	Herbstferien			
Moser	Samis						x				
Henchiri	Rayan						x				
Duricova	Klara								k		
Hashmi	Usman Ahmad					x					
Jasin Pathiranage	Kavindu					x					
Keller	Dominik					x					
Makic	Marija				x						
Puspanathan	Adshay				x						
Mosegaard	Erik				x						
Schranz	Andrin			x							
Sivananthan	Niroj		x								
Vakuyev	Halid		x								
Vivilanathan	Vinujhan			x							
Bakare-Johnson	Hezekiah		x								
Mumenthaler	Masato	x									
Benedetti	Liam	x									
Anderegg	Yannis	x									

Die Grafik zeigt einen aktuellen Einsatzplan für die Klasse «INF18» für das Putzen jeden Freitag mit jeweils drei zugeteilten Schülern.

Er zeigt anschaulich die Defizite der IST-Situation und was an der neuen Lösung wichtig ist. In Zukunft werden auch genau dieser und ähnliche Einsatzpläne durch die Webapp ersetzt.

Abbildung 7: Beispiel Einsatzplan

Wie die Grafik zeigt, soll die neue Applikation die Defizite der alten Lösung eliminieren. Wenn beispielsweise ein Schüler wissen möchte, wer diese Woche für das Putzen eingeteilt ist, muss er zuerst das Dokument in der Serverablage oder in der «Teams» Dateiablage suchen gehen. Danach muss er die aktuelle Kalenderwoche finden und die zugeteilten Namen. Dies ist äusserst umständlich und zeitaufwendig.

Auch aus Sicht der Lehrer ist die momentane Lösung eher suboptimal. Um einen neuen Einsatzplan zu erstellen, braucht die Lehrperson die aktuelle Klassenliste, um alle Schüler einzutragen.

16.2 SOLL-Situation Beschrieb

In Zukunft soll es möglich sein, mit der neuen Applikation den bisherigen Prozess in nur einem Bruchteil der Zeit zu erledigen. Sowohl für Schüler als auch für Lehrpersonen soll die neue Lösung einen echten Mehrwert bieten.

Das erarbeitete Gedankengut sieht es vor ein Webtool zu erstellen, welches die komplizierten Pläne gänzlich ersetzen soll. Es soll eine Seite zur Administration geben, die es Lehrern ermöglicht einen oder mehrere Lernende einem Termin zuzuteilen. Des Weiteren sollen diese Termine dann auf einem modernen und übersichtlichen Kalender angezeigt werden. Diese Kalenderansicht soll auch Lernenden als Informationskanal zugänglich sein. Im Folgenden wird der grobe Aufbau der Applikation beschrieben.

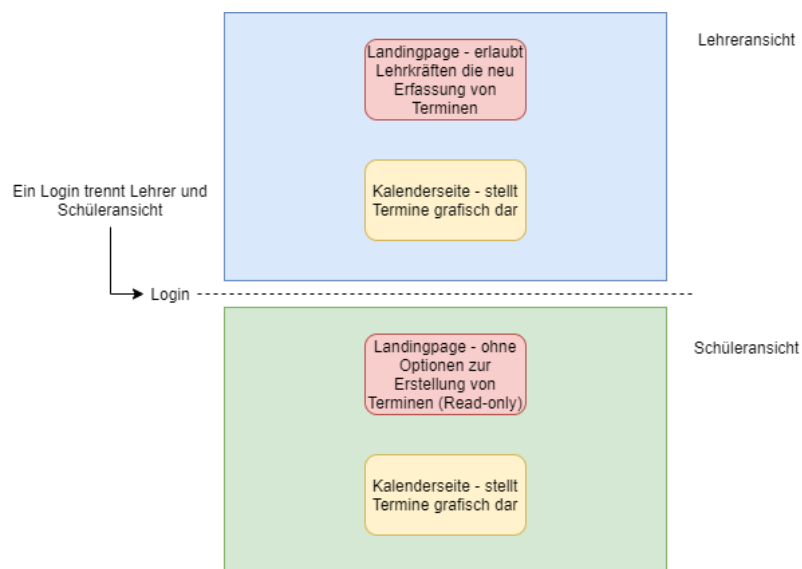


Abbildung 8: Aufbau der Applikation

Die Planung sieht es vor, die Lehrer- und die Schüleransicht durch ein Login zu trennen (siehe Abb. 4: Aufbau der Applikation). So kann ein Schüler zwar alle erstellten Terminsets einsehen, aber er kann diese nicht bearbeiten. Ein Schüler hat nur Lese-Berechtigung. Wenn sich ein User einloggt, erscheinen die Optionen einen neuen Termin zu erstellen oder einen vorhandenen zu bearbeiten.

In der Kalenderansicht werden die erstellten Termine nur angezeigt. Das heisst hier haben Schüler die gleiche Ansicht wie die Lehrpersonen.

Auf der Landingpage sind Termine tabellarisch dargestellt. Die hier dargestellten Termine werden dann genauso im Kalender angezeigt.

Wenn ein Lehrer einen neues Terminset erstellen möchte, muss er sich als erstes einmal einloggen. Hat er dies erledigt, gelangt er auf die Lehreransicht der Terminübersicht. Hier kann er über einen Button ein neues Terminset erstellen. Die Lehrperson trägt dann alle notwendigen Parameter ein, wie Name und Dauer des Events. Er trägt ausserdem die gewünschte Klasse, Gruppengrösse sowie die verantwortliche Lehrperson ein. Hat er diese Werte alle eingetragen, übergibt er der Applikation noch das gewünschte Startdatum, bzw. den Wochentag, an dem der Event stattfindet. Wenn er dann auf «Speichern» drückt, erstellt die Applikation automatisch ein Terminset, das sich wöchentlich am ausgewählten Wochentag wiederholt. Den erstellten Terminen werden dann zufällige Schüler zugeteilt, abhängig von der gewählten Klasse und Gruppengrösse.

Zu einem Termin gehören zwingend die notwendigen Attribute (siehe Grafik unten). Diese sind die gleichen wie bei der bisherigen Lösung. Die einzelnen Attribute stehen teilweise in einer Beziehung zueinander.

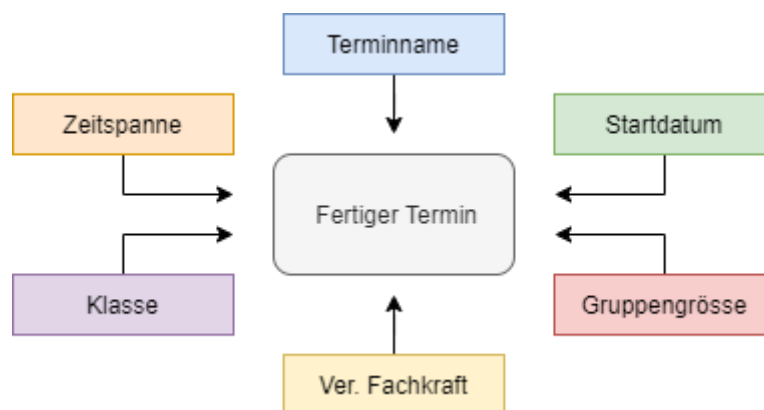


Abbildung 9: Termin und zug. Attribute

Um die Applikation möglichst benutzerfreundlich zu gestalten, wurden einige der wöchentlichen Aufgaben analysiert. Meistens findet die Zuteilung quartalweise oder semesterweise statt. Auf eine jährliche Zuteilung kann deswegen verzichtet werden. Am besten passen die Zeitspannen Ein-, Zwei- und Sechs Monate. Bei der Gruppengrösse machen 1, 2, 3 und 4 am meisten Sinn.

16.2.1 Übersichts-Seite

Die erste Seite, welche gleichzeitig auch die Landingpage ist, zeigt tabellarisch alle zuvor erfassten Termine mit sämtlichen zugehörigen Informationen auf. Hier ist es zwingend nötig sämtliche Use-Cases der Zielgruppe (Lehrkräfte) abzudecken.

Wenn eine Lehrperson einen neuen Termin erfassen möchte, soll sie dies einfach über einen Button tun können. Dann öffnet sich ein Dialogfenster, in welchem sie alle Informationen eingibt. Es braucht ein Auswahlfeld für eine sinnvolle Zeitspanne sowie ein weiteres Auswahlfeld für die zuständige Lehrperson. Des Weiteren braucht jeder Termin einen Namen und eine zugeteilte Klasse mit einem Feld für die Gruppengrösse. Nachdem alle nötigen Informationen durch den Lehrer erfasst wurden, bestätigt er diese und der Termin wird gespeichert.

16.2.2 Kalenderansichts-Seite

Der zentrale Kern der Applikation bildet eine Kalenderansicht, welche die zuvor erfassten Termine grafisch darstellt. Der Kalender bietet eine Ansicht für den aktuellen und den nächsten Monat. Er lässt sich aber bei Bedarf vor und zurück blättern.

Ein Termin ist von den anderen Terminen farblich klar zu unterscheiden. Wenn er durch den User angewählt wird, erscheint eine detaillierte Ansicht mit allen Informationen, die dem entsprechenden Termin zugehören.

16.3 Persönliche Vorgehensziele

Die persönlichen Vorgehensziele setzen sich aus Zielen zusammen, welche ich mir persönlich für dieses Projekt vorgenommen habe.

ID:	Ziel:	Beschreibung:
Z1	Dokumentieren nicht vernachlässigen	Es wird fortlaufend dokumentiert. Die Balance zwischen Coden und Dokumentieren ist ausgewogen.
Z2	Einhalten des Zeitmanagements	Der Zeitplan wird eingehalten. Allfällige Rückstände werden durch Überstunden kompensiert.
Z3	Perspektive ändern	Es wird darauf geachtet für den Leser verständlich zu dokumentieren. Dafür wird versucht die Perspektive des Lesers einzunehmen.

Tabelle 30: Ziele

16.4 Projektziele

Die Projektziele beziehen sich auf den fachlichen Teil der Arbeit. Sie sind aus dem Detailbeschreibung, sowie dem SOLL-Zustand abgeleitet.

ID:	Ziel:	Beschreibung:
Z1	Die Einsatzpläne müssen für alle im Browser einsehbar sein.	Sowohl Schüler wie auch Lehrer können die Einsatzpläne einsehen.
Z2	Die Erstellung von Einsatzplänen muss für Lehrpersonen erleichtert werden.	Der Prozess bei der Erstellung eines Einsatzplanes soll erleichtert werden.
Z3	Es kann ein Einsatzplan mit der entsprechenden Bezeichnung erstellt werden.	Für jeden Einsatzplan kann eine entsprechende Bezeichnung gewählt werden.
Z4	Für jeden Einsatzplan muss eine Klasse ausgewählt werden können, deren Schüler automatisch als Einsatzkräfte gesetzt werden.	Wenn eine Klasse einem Einsatzplan zugeteilt wird, werden zufällige Schüler dieser Klasse als Einsatzkräfte gesetzt.
Z5	Für jeden Einsatzplan kann der Wochentag (MO-FR) angegeben werden.	Einem Einsatzplan wird ein Startdatum mitgegeben. An diesem Wochentag wiederholt sich dieser Einsatz jeweils.
Z6	Für jeden Einsatzplan kann die Gruppengrösse gesetzt werden.	Dem Einsatzplan wird eine Gruppengrösse mitgegeben, welche bestimmt wie viele Schüler zugeteilt werden.
Z7	Die Webapp erstellt auf Knopfdruck einen Einsatzplan für das jeweilige Ämtli mit der jeweiligen Gruppengrösse am gewünschten Wochentag.	Wenn alle Parameter eines Einsatzplanes eingegeben wurden, speichert der User diesen und die Webapp bestimmt die Schüler automatisch per Zufall. Dabei werden automatisch abwechselnd Lernende aus der gewählten Klasse gewählt.
Z8	Der erstellte Einsatzplan wird übersichtlich dargestellt.	In der Kalenderansicht sieht man die Ansicht für den aktuellen und den nächsten Monat.
Z9	Wenn man nicht angemeldet ist, kann man die Einsatzpläne einsehen, aber nichts an ihnen verändern.	Für einen nicht angemeldeten Schüler ist es nicht möglich einen Einsatzplan zu erstellen.

Tabelle 31: Projektziele

16.5 Anforderungen

Welche Ansprüche werden an die Applikation gestellt? Was sind die technischen und funktionalen Anforderungen?

16.5.1 Funktionale Anforderungen

ID:	Anforderungsbeschreibung	Gewichtung:
A1	Login Funktion für Lehrperson -Vom Admin bereitgestellte Credentials -Gesperrte Funktionen wie erstellen und bearbeiten von Terminen sind eingeloggt möglich	Hoch
A2	Erstellen und bearbeiten von Terminen -Für Lehrpersonen ist es möglich Termine zu erstellen und diese, wenn gewünscht, zu bearbeiten -Für einen Schüler ist dies NICHT möglich	Hoch
A3	Einem einzelnen Termin sind immer unterschiedliche Schüler zugeordnet -Schüler sind nach einer festen Methode über eine zufällige Zahl zugeteilt -Es muss sichergestellt sein, dass ein Schüler nicht zweimal vorkommt in einem Termin -Bei einem ganzen Terminset sollen einzelne Schüler einer Klasse nicht öfter als andere zugeteilt werden	Hoch
A4	Ein Terminset lässt sich nur erstellen, wenn alle notwendigen Parameter übergeben werden. Dazu gehören (zwingend notwendig): -Name -Startdatum -Gruppengrösse -Klasse -Dauer Optionale Parameter (nicht zwingend notwendig): -Verantwortliche Lehrperson	Mittel
A5	Alle Parameter der Termine lassen sich in der Ansicht sortieren -Bei Buchstaben wird alphabetisch sortiert -Zahlen werden nach Grösse aufsteigend oder absteigend sortiert	Niedrig

Tabelle 32: Funktionale Anforderungen

16.5.2 Nicht Funktionale Anforderungen

ID:	Anforderungsbeschreibung	Gewichtung:
A1	Das GUI lässt sich ohne besondere Einweisung oder Vorkenntnisse problemlos bedienen -Eine externe Person (Schüler oder Lehrperson) kann die Applikation vollumfänglich bedienen, ohne dass irgendeine Art von Benutzeranleitung notwendig wäre -Dies wird durch das Testing im späteren Verlauf sichergestellt	Hoch
A2	Die Applikation stellt Informationen übersichtlich dar -Damit der eigentliche Mehrwert der Applikation gegeben ist, muss die Applikation alle Terminsets übersichtlich darstellen	Mittel
A3	Alle Termine sind in der Kalenderseite in einer anderen Farbe dargestellt -Die Farbe eines einzelnen Termins wird per Zufall gewählt	Niedrig
A4	Die Kalenderansicht zeigt eine zwei Monats Ansicht -Im Kalender wird jeweils der aktuelle und der nächste Monat abgebildet	Mittel

Tabelle 33: Nicht Funktionale Anforderungen



17 Phasenfreigabe Konzept



Name	Unterschrift
Yannis Anderegg	
Özcan Altın	

Tabelle 34: Unterschriften

18 Konzept

Das Projekt wird konkreter geplant und es werden erste Design Entwürfe erstellt.

Auf einen Variantenentscheid wurde verzichtet, da in der Aufgabenstellung der Technologie Stack bereits vorgegeben wurde.

Weitere Konzeptionelle Entscheidungen, bei denen ein Variantenvergleich Sinn ergeben hätte, wurden nicht getroffen.

18.1 Datenbank Design

In diesem Projekt trifft man viele verschiedene Daten an. Einerseits müssen die Schüler, wie in der Aufgabenstellung definiert, in einer SQL-Datenbank gespeichert sein, andererseits müssen auch die eigentlichen Terminsets ebenfalls in einer Datenbank gespeichert werden.

Beim Datenbank Design gibt es einige besondere Aspekte zu beachten. Wie schon in der Aufgabenstellung beschrieben, muss es ein SQL-Skript geben, welches alle Lernenden der Abteilung Informatik in eine Datenbank einfügt.

Ebenfalls wichtig zu beachten ist, dass sich viele der Daten im Laufe der Zeit noch ändern könnten, wie etwa wenn neue Schüler hinzukommen oder gehen.

18.2 Entity Relationship Diagram

Es wird ein ERD erstellt, welches sich im Laufe der Zeit noch ändern kann. Das ERD wird mithilfe der verwendeten DB-IDE erstellt. Dies ermöglicht auch zu einem späteren Zeitpunkt ein sog. Reverse-Engineering der Daten.

Das erstellte ERD sieht wie in der folgenden Grafik aus.

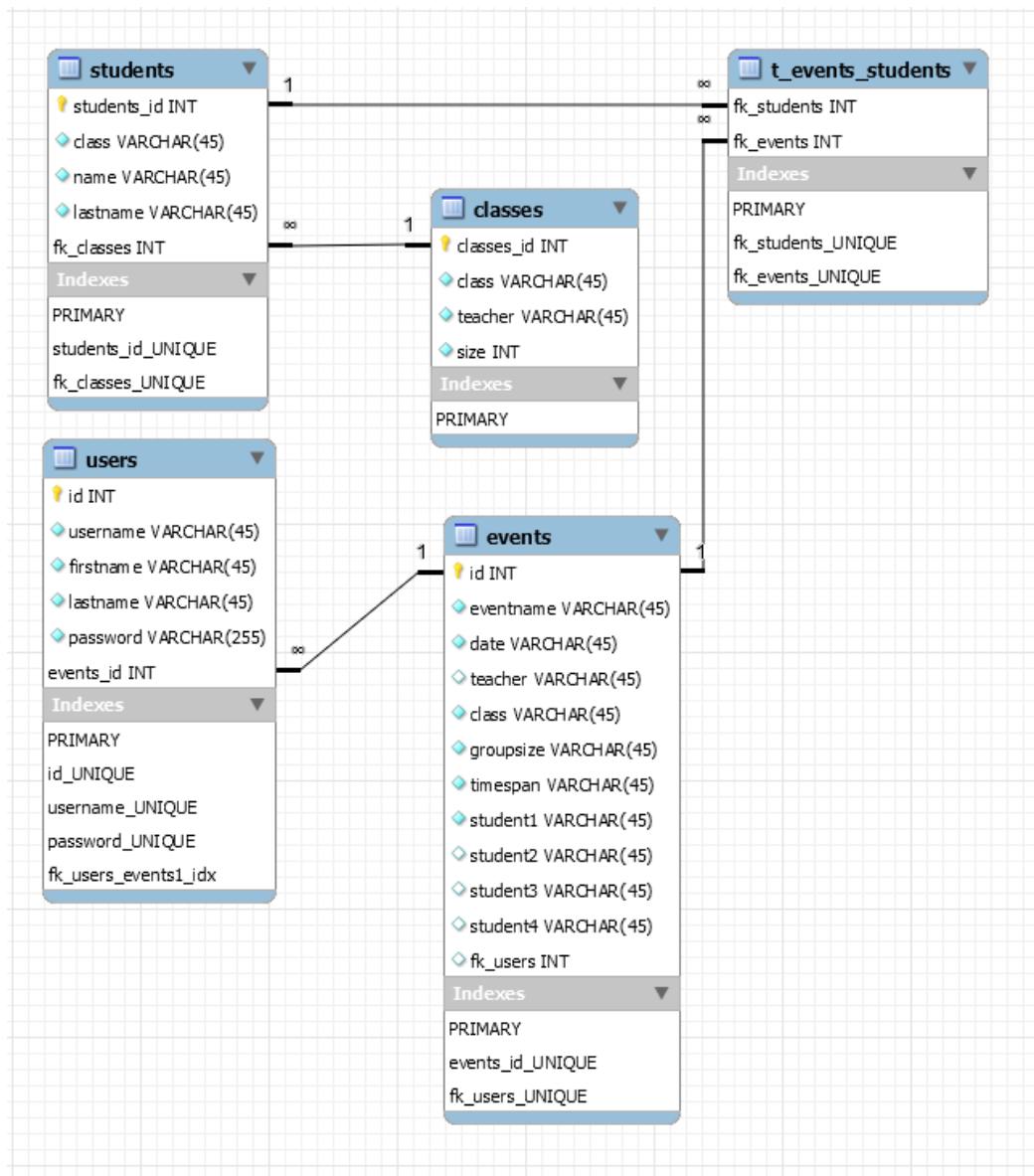


Abbildung 10: ERD

18.2.1 Classes und Students

In der Tabelle «Students» werden alle Schüler der Abteilung Informatik aufgeführt. Diese Tabelle ist so auch in der Aufgabenstellung erfasst. Es besteht eine «1:n» Beziehung zu der Tabelle «Classes». Da auch die Klasse als Parameter eines Termins übergeben werden muss, braucht es hier eine separate Tabelle für die Klassen. Die beiden Tabellen sind über eine Zwischentabelle verbunden.

Wenn es in Zukunft neue Schüler oder Klassen geben wird, können diese einfach in die Datenbank eingefügt werden.

18.2.2 Events

Wenn ein neuer Termin erfasst wird, dann wird dieser in der Tabelle «Events» gespeichert. Dabei werden Daten aus den Tabellen «Classes» und «Students» bezogen. Die Tabelle Events hat keine Relationen zu den anderen Tabellen.

18.2.3 Users

Für das Login braucht es noch eine weitere Tabelle. In dieser sind der oder die User gespeichert. Diese Tabelle hat keine Relation zu anderen Tabellen.

18.3 Passwortsicherheits-Konzept

Im Passwortsicherheits-Konzept wird beschrieben, wie Passwörter geschützt werden können.

18.3.1 Passwörter

Da die Planung es vorsieht, ein Login in der Applikation zu implementieren, müssen die zugehörigen Passwörter entsprechend geschützt werden.

Die Logindaten der Applikation sind statisch, das heisst sie werden vom Admin an die User ausgeteilt und können nicht geändert werden. Alle Passwörter müssen folgende Eigenschaften (Tabelle) aufweisen.

Eigenschaft	Umsetzung	Beitrag zur Sicherheit
Passwortlänge < 8	Urladung (Seeds) in die DB	Hoch
Sonderzeichen	Urladung (Seeds) in die DB	Hoch
Gross- Kleinbuchstaben	Urladung (Seeds) in die DB	Mittel

Tabelle 35: Passwörter

18.3.2 Verschlüsselung

Applikationsinterne Passwörter werden sowohl während der Kommunikation wie auch in der Datenbank in verschlüsselter Form gehandhabt. Hier wird ein Verschlüsselungsalgorithmus eingesetzt. Dies wird später in der Realisierung noch genauer beschrieben.

18.4 Skalierung

Die erstellte Webapp besitzt einen Nutzerkreis von ungefähr 70 Leuten. Die Abteilung Informatik hat ungefähr 60 Lernende verteilt auf drei Klassen und 8 Lehrkräfte. Es wird davon ausgegangen, dass eine Klasse etwa 3-12 wiederholende Aufgaben erfassen möchte. Ein Terminset hat eine maximale Zeitspanne von sechs Monaten. Für jeden Monat werden vier Termine erstellt (jede Woche einen Termin).

Daraus ergibt sich dann folgende Rechnung:

3 Klassen * 12 Terminsets * 6 Monate * 4 Wochen = **864 Termineinträge**

Es entstehen also 864 Einträge in die Tabelle «Events». Dies stellt kein Problem dar, da die Datenbank problemlos ein Mehrfaches davon speichern könnte. In der Entwicklung wird die IDE «MySQL Workbench» verwendet. Hier liegt das Limit der maximalen Anzahl an Records (Einträge) bei 1000 und wird somit ebenfalls nicht überschritten.

18.5 Systemarchitektur und Aufbau

Es wird der Aufbau des Programms dokumentiert, unter Zuhilfenahme von z.B. Diagrammen.

18.5.1 Schnittstellen der Applikation

Die Applikation als Ganzes gesehen, hat viele einzelne Komponenten. Folgende Grafik soll dabei für das nötige Verständnis sorgen.

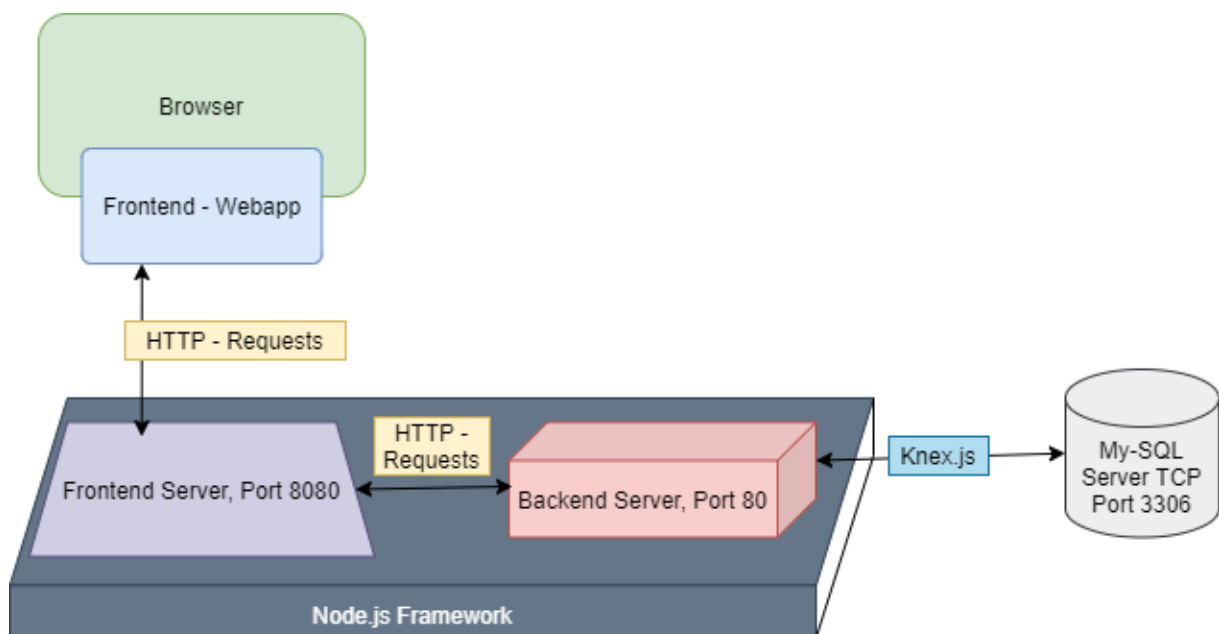


Abbildung 11: Aufbau und Struktur

Es gibt einen Frontend Entwicklungs-Server und den Backend-Server. Über HTTP-Abfragen kommunizieren beide miteinander. Das Backend bezieht mittels des Plugins «Knex.js» Daten aus der MySQL-Datenbank. Zur Kommunikation zwischen Frontend und Backend werden sog. APIs verwendet.

18.6 Anwendungsfälle (Use-Cases)

Um die Applikation zu konzeptionieren, werden alle möglichen Anwendungsfälle angeschaut und analysiert, bezüglich des Aufwands oder der Umsetzung. Das im Folgenden zu sehende «Use-Case Diagram» soll dies genauer verdeutlichen.

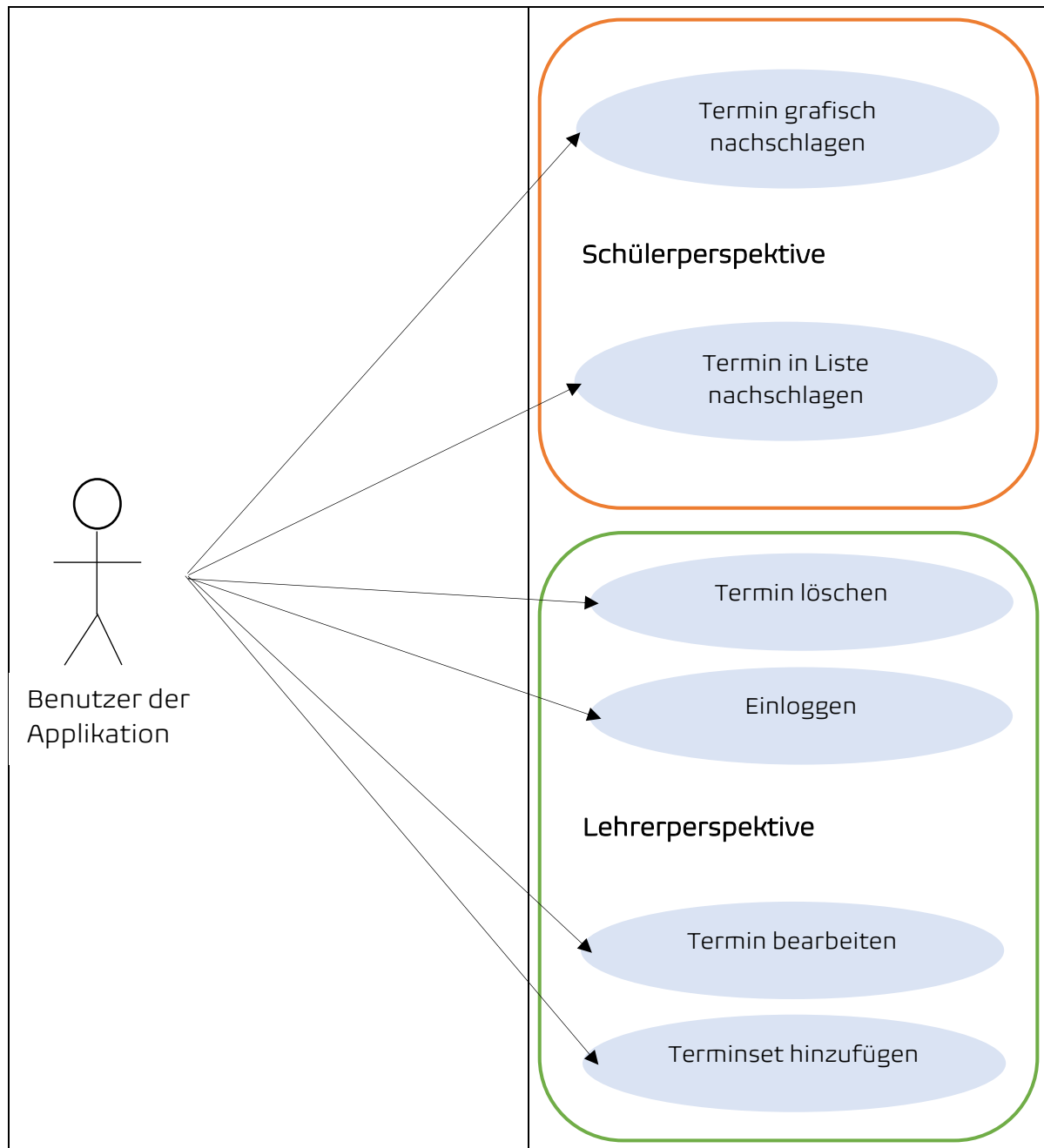


Tabelle 36: Anwendungsfälle

Im «Use-Case Diagram» sind die Anwendungsfälle in die Perspektive von Schüler und Lehrer unterteilt, da diese eine unterschiedliche Ansicht der Applikation besitzen.

Die insgesamt sechs Use-Cases zeigen gut auf was die Applikation können muss. Im Folgenden wird jeder Anwendungsfall noch genauer beschrieben.

18.6.1 Anwendungsfälle im Detail

Was die einzelnen Anwendungsfälle im genauen Detail bedeuten.

ID	Anwendungsfall	Perspektive	Beschreibung
A1	Termin grafisch nachschlagen	Schüler	Ein Schüler möchte sich einen Termin im Kalender ansehen
A2	Termin in Liste nachschlagen	Schüler	Ein Schüler will sich einen Termin in der tabellarischen Ansicht ansehen
A3	Termin löschen	Lehrer	Ein eingeloggter Lehrer will einen Termin löschen
A4	Einloggen	Lehrer	Ein Lehrer loggt sich in der Applikation ein
A5	Termin bearbeiten	Lehrer	Ein Lehrer ändert Werte eines vorhandenen Termins
A6	Terminset hinzufügen	Lehrer	Ein Lehrer fügt ein neues Terminset hinzu

Tabelle 37: Details Anwendungsfälle

18.6.2 Umsetzung der Anwendungsfälle

Die Anwendungsfälle sind sehr hilfreich beim Konzeptionieren der Applikation, aber sie können auch in Bezug auf die Projektmethodik und die Zeiteinteilung hilfreich sein. Die folgende Grafik zeigt die Anwendungsfälle mit Bezug auf die Methodenwahl, um diese umzusetzen. Eine Spalte zeigt ausserdem eine Schätzung, wie zeitintensiv jeder Anwendungsfall in der Realisierung sein wird.

ID	Anwendungsfall	Zeitintensität	Umsetzung	Technologie Stack
A1	Termin grafisch nachschlagen	Hoch	Mit dem Designframework Vuetify kann ein Kalender mit vorgefertigten Komponenten erstellt werden. Daten werden mit Knex.js aus der Datenbank geholt.	-Vuetify (V-Calendar) -Knex.js
A2	Termin in Liste nachschlagen	Mittel	Mit Middleware wie Axios oder Fetch können Daten aus dem Backend in das Frontend geladen werden.	-Axios -Fetch
A3	Termin löschen	Niedrig	Die http-Methode «delete» kann über Middleware wie Axios verwendet werden.	-HTTP (Methode «delete») -Axios, Fetch -Knex.js (DB-Zugriff)
A4	Einloggen	Mittel	Ein Token-basiertes Login lässt sich mit der Middleware «jsonwebtoken» realisieren.	-jsonwebtoken -Axios, Fetch -Knex.js (DB-Zugriff)
A5	Termin bearbeiten	Mittel	Mittels der http-Methode «update» lassen sich Einträge über das Backend in der DB verändern.	-HTTP (Methode «update») -Axios, Fetch -Knex.js (DB-Zugriff)
A6	Terminset hinzufügen	Sehr hoch	Mit der http-Methode «post» lassen sich neue Daten über das Backend in die DB eintragen.	-Axios, Fetch -Knex.js (DB-Zugriff)

Tabelle 38: Umsetzung Anwendungsfälle

18.7 Mockups

Die Mockups dienen dazu, das GUI in einem ersten Schritt zu entwerfen. In der späteren Realisierung werden designtechnische Entwicklungsentscheidungen unter Zuhilfenahme der Mockups entschieden.

18.7.1 Login-Page

Die erste Seite, die ein Nutzer sieht, ist immer die Login-Page. Diese nimmt einen Nutzernamen und ein Passwort entgegen.

Termine | Kalender | Logout

Login

Username

Password

Einloggen

Abbildung 12: Mockup Login-Page

Da, wie in der Aufgabenstellung definiert, das Designframework Vuetify verwendet wird, ist das GUI der Login-Page einem gewissen Layout unterworfen. Dies hat den Nachteil, dass man keine besonders grosse Freiheit hat beim Design der einzelnen Komponenten. Es lässt sich jedoch viel Zeit sparen, da vieles schon durch das Framework vordefiniert ist.

18.7.2 Termine

Die Termine / Events Seite zeigt alle einzelnen Termine tabellarisch auf. Diese Seite wird vor allem von den Lehrern benutzt, um die Termine zu administrieren. Es gibt einen «+» Button, um einen neuen Termin zu erstellen. Ein Termin kann durch die Optionen am rechten Rand bearbeitet oder gelöscht werden.

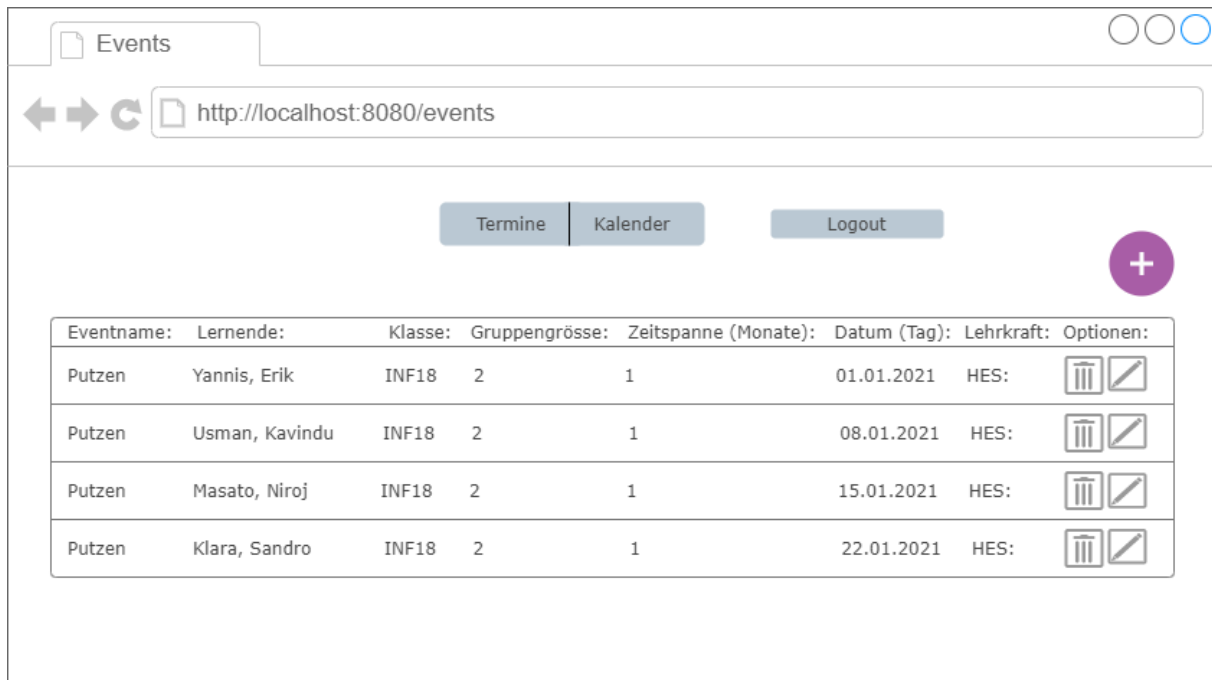


Abbildung 13: Mockup Termine

Im Mockup wurde als Beispiel ein vorhandenes Terminset eingefügt. Da das Attribut «Zeitspanne» auf einen Monat gesetzt ist, gibt es vier Einträge (jede Woche). Der Parameter «Gruppengröße» ist auf zwei gesetzt, also haben die Termine jeweils zwei zufällige Schüler zugeteilt.

18.7.3 Kalender Seite

Die Kalender Seite zeigt alle Termine in einem grafischen Kalender an.

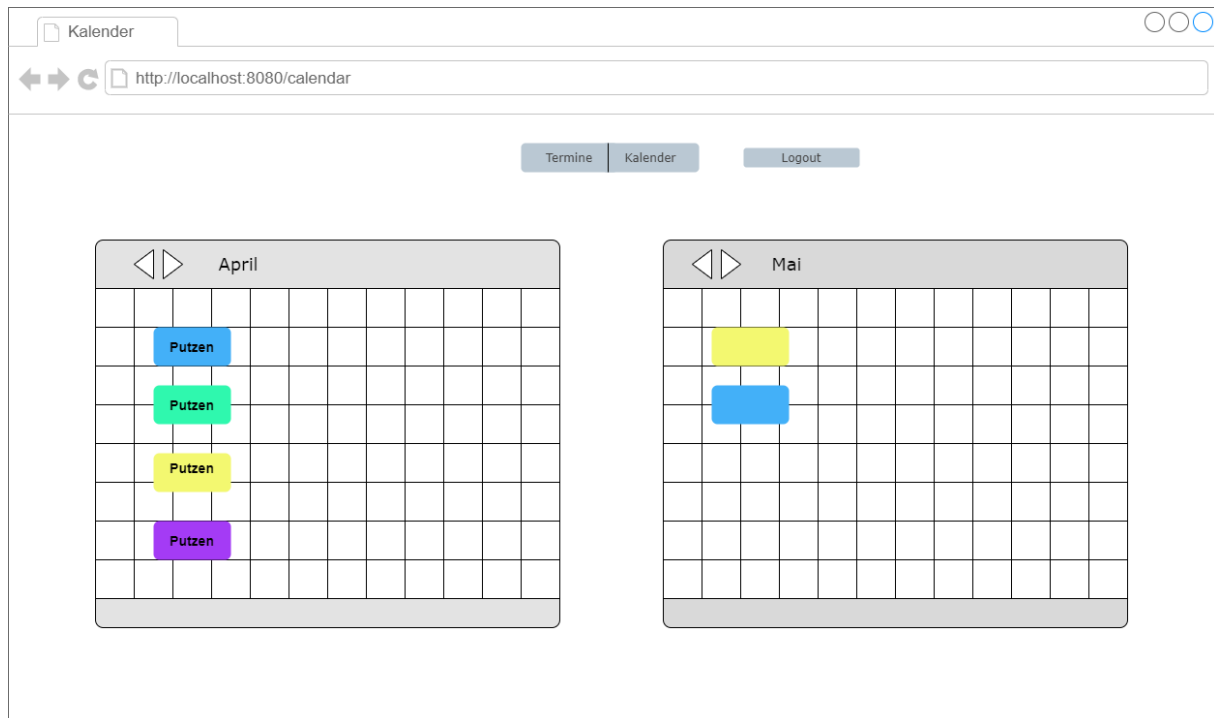


Abbildung 14: Mockup Kalender

Der Kalender zeigt, wie in den Anforderungen definiert, eine zwei-Monatsansicht. Über die Pfeile kann im Monat geblättert werden. Termine werden mit Zufallsfarben versehen.

18.8 Testkonzept

Das Testkonzept dient dazu, die Applikation auf alle nötigen Anforderungen zu testen, und Mängel aufzudecken. Die Testfälle werden am Ende der Realisierung durchgeführt.

18.8.1 Testdaten

Für die User-Tests wird dem Benutzer, bzw. dem Tester, ein Login zur Verfügung gestellt. Die Credentials sind die folgenden:

Username	Passwort
Admin	123

Tabelle 39: Testdaten

18.8.2 Testrahmen

Die Testfälle werden durch den Tester Kavindu Pathiranage durchgeführt. Dem Tester wird der zu erledigende Testfall gezeigt und das Login übergeben, danach folgen keine weiteren Instruktionen. Der Tester führt den Testfall nach der entsprechenden Beschreibung im Testkonzept durch.

18.8.3 Testumgebung

Alle Testfälle werden in folgenden Umgebungen durchgeführt:

Software	Version:
Google Chrome 64Bit (User-Tests)	Version 88.0.4324.190
Microsoft Edge 64Bit (User-Tests)	Version 44.18362.449.0

Tabelle 40: Testsoftware

18.8.4 Test-Methoden

Die meisten Tests werden mithilfe der Blackbox Testmethode durchgeführt. Bei Blackbox-Tests hat der Tester keine Vorkenntnisse oder ähnliches.

Bei White-Box Tests hat der User ein gewisses Vorwissen und Einblick in den Source Code.

18.8.5 Re-Testing

Im Falle eines Tests mit kritischer Mängelklasse, wird ein Re-Testing durchgeführt, nachdem die Fehler durch den Kandidaten behoben wurden. Bei erfolgreich abgeschlossenen Tests gilt die Realisierung als abgeschlossen.

18.8.6 Mängelklassen und Testkategorien

Es werden Mängelklassen definiert (siehe folgende Tabelle). Im Falle eines schweren Mangels, Klasse «M3» (folgende Tabelle), wird ein Re-Testing durchgeführt (siehe 13.6.5 «Re-Testing»).

ID	Fehlerklasse	Beschreibung
M0	Kein Mangel	Kein Mangel feststellbar in dem durchgeführten Test
M1	Leichter Mangel	Hauptteil des Tests bestanden mit kleinen Mängeln
M2	Mittlerer bis schwerer Mangel	Elementare Teile des Tests sind fehlgeschlagen
M3	Kritischer Mangel	Sämtliche Teile des Tests sind fehlgeschlagen, erfordert ein Re-Testing.

Tabelle 41: Mängelklassen

Zusätzlich werden alle durchgeführten Tests in eine Unterkategorie eingeteilt (siehe folgende Tabelle). Dies dient einem besseren Verständnis von Sinn und Zweck eines Tests.

ID	Kategorie	Beschreibung
K1	Login	Die Applikationsinterne Login-Funktion
K2	User-Input	Durch den User eingegebene Daten
K3	Output Applikation	Output durch die Applikation
K4	Grafischer Output	Grafischer Output durch die Applikation

Tabelle 42: Testkategorien

18.8.7 Test-Tabelle

Wie sieht der genaue Aufbau eines Testfalls aus? Welche Parameter gibt es und wie sieht die verwendete Testtabelle aus?

Überschrift	Beschreibung	
<i>Test-ID</i>	Zur eindeutigen Identifizierung eines Testfalls	
<i>Kategorie</i>	Kategorisierung eines Testfalls	
<i>Testmethode</i>	Verwendete Testmethode	
<i>Testfall</i>	Genaue Beschreibung eines Testfalls	
<i>Voraussetzungen</i>	Für einen Testfall nötige Voraussetzungen	
<i>Testbeschreibung</i>	Zu erfüllender Test 1	Zu erfüllender Test 2
<i>Testschritte</i>	Schritte zur Durchführung Test 1	Schritte zur Durchführung Test 2
<i>Testinformationen</i>	Vorwissen, um Test durchzuführen	
<i>Erwartetes Resultat</i>	Optimales Resultat eines Tests	Optimales Resultat eines Tests
<i>Tats. Resultat</i>	Tatsächliches Resultat eines Tests	Tatsächliches Resultat eines Tests
<i>Kommentar</i>	Kommentar bei Nichterfüllung eines Tests	Kommentar bei Nichterfüllung eines Tests
<i>Mangelklassifizierung</i>	Einteilung nach Mängelklasse (13.6.6 Mängelklassen)	Einteilung nach Mängelklasse (13.6.6 Mängelklassen)

Tabelle 43: Beispiel Testtabelle

18.9 Testfälle

Die Testfälle werden am Ende der Realisierung durchgeführt, sie werden jedoch schon hier konzeptioniert.

18.9.1 Test 1

Überschrift	Beschreibung				
Test-ID	T1				
Kategorie	K1 Login				
Testmethode	Blackbox				
Testfall	Der Benutzer loggt sich mit bereitgestellten Credentials ein.				
Voraussetzungen	-Der Kandidat stellt dem Tester die nötigen Login-Daten zur Verfügung -Im Moment des Tests ist kein Benutzer eingeloggt -Die Applikation ist lokal gehostet				
Testbeschreibung	Keine Eingabe	Falscher Username & Passwort	Richtiger Username & Falsches Passwort	Falscher Username & Richtiges Passwort	Richtiger Username & Passwort
Testschritte	1. Username eingeben 2. Passwort eingeben 3. Mittels «Einloggen» bestätigen				
Testinformationen		Username: «Test1» Passwort: «12345»	Username: «admin» Passwort: «12345»	Username: «Test1» Passwort: «123»	Username: «admin» Passwort: «123»
Erwartetes Resultat	Keine Reaktion in der Applikation feststellbar, Login Vorgang fehlgeschlagen				Login Vorgang erfolgreich
Tats. Resultat					
Kommentar					
Mangelklassifizierung					

Tabelle 44: Testfall 1

18.9.2 Test 2

Überschrift	Beschreibung	
Test-ID	T2	
Kategorie	K4 Grafischer Output	
Testmethode	Blackbox	
Testfall	In nicht-eingeloggtem Zustand ist es dem User nicht möglich ein Aufgabenset zu erstellen, zu bearbeiten oder zu löschen	
Voraussetzungen	-Im Moment des Tests ist kein Benutzer eingeloggt -Die Applikation ist lokal gehostet	
Testbeschreibung	Auf der Seite «Events» ist es dem User nur in eingeloggtem Zustand möglich über den «+» Button ein neues Terminset zu erstellen, oder über die Edit-Buttons vorhandene Termine zu bearbeiten oder zu löschen	
Testschritte	<ol style="list-style-type: none"> 1. Der User loggt sich aus 2. Der User befindet sich auf dem Pfad http://localhost:8080/Events 3. Der User navigiert in den Hauptbereich der Seite 	<ol style="list-style-type: none"> 1. Der User loggt sich ein (Testfall 1) 2. Der User befindet sich auf dem Pfad http://localhost:8080/Events 3. Der User navigiert in den Hauptbereich der Seite
Testinformationen		
Erwartetes Resultat	Der nicht-eingeloggte User ist nicht in der Lage den «+» Button und die Edit-Buttons zu sehen, und so ein Terminset zu bearbeiten, zu löschen oder hinzuzufügen	Der eingeloggte User ist in der Lage den «+» Button und die Edit-Buttons zu sehen, und so ein Terminset hinzuzufügen, zu bearbeiten oder zu löschen
Tats. Resultat		
Kommentar		
Mangelklassifizierung		

Tabelle 45: Testfall 2

18.9.3 Test 3

Überschrift	Beschreibung
Test-ID	T3
Kategorie	K2 User-Input
Testmethode	Blackbox
Testfall	Aufgabenset erstellen
Voraussetzungen	-Im Moment des Tests ist ein Benutzer eingeloggt -Die Applikation ist lokal gehostet
Testbeschreibung	Auf der Seite «Events» ist es dem User in eingeloggten Zustand möglich über den «+» Button ein neues Terminset zu erstellen
Testschritte	<ol style="list-style-type: none"> 1. Der User befindet sich auf dem Pfad http://localhost:8080/Events 2. Der User navigiert zu dem rechts-mittig lokalisierten «+» Button und drückt diesen 3. Im sich daraufhin öffnenden Dialog füllt der User folgende Zufallsdaten ein: <i>Name: «Test», Lehrer: «Altin Özcan», Zeitspanne: «1», Klasse: «inf18», Gruppengrösse: «2», Datum: «Montag»</i> 4. Er betätigt den «Speichern» Knopf
Testinformationen	
Erwartetes Resultat	Das erstellte Terminset erscheint in der Liste
Tats. Resultat	
Kommentar	
Mangelklassifizierung	

Tabelle 46: Testfall 3

18.9.4 Test 4

Überschrift	Beschreibung
Test-ID	T4
Kategorie	K2 User-Input
Testmethode	Blackbox
Testfall	Aufgabenset bearbeiten
Voraussetzungen	-Im Moment des Tests ist ein Benutzer eingeloggt -Die Applikation ist lokal gehostet
Testbeschreibung	Auf der Seite «Events» ist es dem User in eingeloggten Zustand möglich über die rechts angeordneten Symbole einen Termin zu bearbeiten
Testschritte	<ol style="list-style-type: none"> 1. Der User befindet sich auf dem Pfad http://localhost:8080/Events 2. Der User navigiert zu dem rechts-mittig lokalisierten Editierungs-Button (linker Button) und drückt diesen 3. Im sich daraufhin öffnenden Dialog ändert der User das Datum auf «Freitag» 4. Er betätigt den «Speichern» Knopf 5. Der Termin hat das neue Datum «Freitag» übernommen
Testinformationen	
Erwartetes Resultat	Der Termin erscheint mit den geänderten Parametern in der Liste
Tats. Resultat	
Kommentar	
Mangelklassifizierung	

Tabelle 47: Testfall 4

18.9.5 Test 5

Überschrift	Beschreibung
Test-ID	T5
Kategorie	K2 User-Input
Testmethode	Blackbox
Testfall	In eingeloggtem Zustand ist es dem User möglich ein Aufgabenset zu löschen
Voraussetzungen	-Im Moment des Tests ist ein Benutzer eingeloggt -Die Applikation ist lokal gehostet
Testbeschreibung	Auf der Seite «Events» ist es dem User in eingeloggten Zustand möglich, über das rechts angeordnete Symbol einen Termin zu löschen
Testschritte	<ol style="list-style-type: none"> 1. Der User befindet sich auf dem Pfad http://localhost:8080/Events 2. Der User navigiert zu dem rechts-mittig lokalisierten Editierungs-Button (rechter Button) und drückt diesen 3. Der entsprechend zugehörige Termin wurde aus der Liste gelöscht
Testinformationen	
Erwartetes Resultat	Der entsprechende Termin wurde aus der Liste gelöscht
Tats. Resultat	
Kommentar	
Mangelklassifizierung	

Tabelle 48: Testfall 5

18.9.6 Test 6

Überschrift	Beschreibung
Test-ID	T6
Kategorie	K4 Grafischer Output
Testmethode	Blackbox
Testfall	Termin im Kalender anzeigen
Voraussetzungen	-Die Applikation ist lokal gehostet -Es existiert ein vorgängig erstelltes Terminset mit Testdaten
Testbeschreibung	Der Benutzer überprüft, ob ein Termin korrekt im Kalender angezeigt wird
Testschritte	<ol style="list-style-type: none"> 1. Der User befindet sich auf dem Pfad http://localhost:8080/Events 2. Der User sucht den bei Testfall 3 erstellten Termin aus den vorhandenen Terminen aus, und merkt sich den Datumsparameter 3. Der User wechselt über die Navigation oder die URL auf den Pfad http://localhost:8080/Calendar 4. Der User sucht im grafischen Kalender den zuvor gemerkten Termin
Testinformationen	
Erwartetes Resultat	Der zuvor gemerkte Termin wird an der richtigen Stelle (korrektes Datum) im Kalender angezeigt
Tats. Resultat	
Kommentar	
Mangelklassifizierung	

Tabelle 49: Testfall 6

19 Realisierung

Die Durchführung des praktischen Teils und die Dokumentation des Source-Codes.

19.1 Projektumgebung

In der Konzeptphase unter «13.5.2 Umsetzung der Anwendungsfälle» wurden bereits einige Punkte zu möglichen Technologien gesagt. Daraus und aus der Aufgabenstellung kann nun der Technologie Stack abgeleitet werden

Da Node.js schon global installiert ist, braucht es hier keinen weiteren Befehl. In der folgenden Tabelle sind die genauen Kommandozeilenbefehle und die zugehörigen Parameter beschrieben.

Befehl	Parameter	Beschreibung
npm install -g @vue/cli	-g (global)	«npm» steht für «Node Package Manager». Die Node.js Umgebung ist auf dem Entwicklungslaptop bereits global vorinstalliert.
vue create appointment_manager_app		Installiert das Vue.js Framework und gibt dem Projekt den Namen. Bei der Installation kann das Framework modifiziert werden. Hier wird überall die Standard Einstellungen übergeben, bis auf den «Vue-Router», welcher für die Navigation benötigt wird.
vue add vuetify		Installiert das Designframework Vuetify.
npm install "PACKAGE" --save	--save (fügt die entsprechende Dependency im "package.json" File hinzu)	Installiert alle nötigen Packages (siehe nächstes Kapitel). Im Befehl steht "PACKAGE" nur als Platzhalter für das jeweilige Package.
npm init		Initialisiert das Backend.

Tabelle 50: Befehle zum Einrichten (Projektumgebung)

19.1.1 Packages

In dem Projekt werden viele zusätzliche Dienste gebraucht. Diese sogenannten Packages werden über «npm» (Node Package Manager) installiert. Alle nötigen Packages sind im Folgenden kurz beschrieben.

Package / Ort	Befehl	Beschreibung
Express / Backend	npm install express --save	Express ist ein auf Node.js basierendes Framework, welches für das REST-API benötigt wird.
Jsonwebtoken / Backend	npm install jsonwebtoken --save	Dieses Package wird für das Login gebraucht.
Knex / Backend	npm install knex --save	Für Datenbankabfragen wird Knex.js benötigt.
Mysql / Backend	npm install mysql --save	MySQL ist die verwendete Datenbank.
Axios / Backend	npm install axios --save	Axios wird für HTTP-Abfragen ins Backend benötigt.
V-Calendar / Frontend	npm install v-calendar --save	Für den Kalender in der Applikation braucht es diese Vuetify Extension.
Bcrypt / Backend	npm install bcryptjs --save	Die Logindaten müssen verschlüsselt werden.

Tabelle 51: Befehle Packages

19.2 Ordnerstruktur

Die Ordnerstruktur soll einen Überblick über alle Dateien aufzeigen.

19.2.1 Frontend

Die Grundstruktur des Frontends wird automatisch von Vue.js erstellt. Diese wird so beibehalten. Die meiste Arbeit am Projekt hat im dem /src/ Verzeichnis stattgefunden.

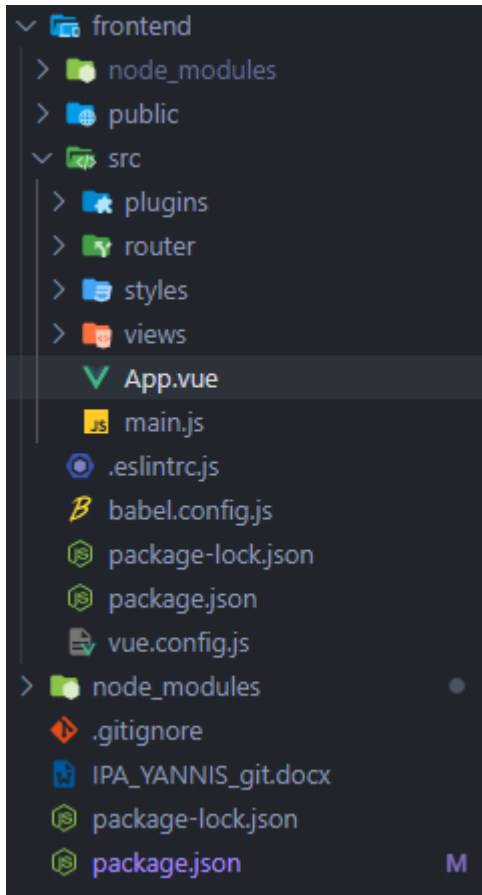
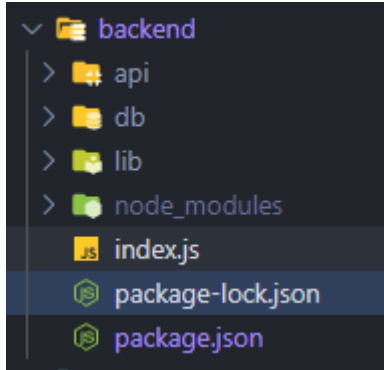
	<p>Plugins</p> <p>Enthält Middleware wie etwa Vuetify</p> <p>Router</p> <p>Das Plugin für das Routing mittels «Vue-Router»</p> <p>Styles</p> <p>CSS-Daten (falls vorhanden)</p> <p>Views</p> <p>Die einzelnen Seiten (Komponenten der Seite)</p> <p>App.vue</p> <p>Das Hauptfile von Vue verbindet alle Komponenten miteinander</p> <p>Main.js</p> <p>Enthält die Definition der Vue Instanz und den zugehörigen Plugins (z.B. Router)</p>
--	--

Abbildung 15: Ordnerstruktur

19.2.2 Backend

Im Backend ist der Aufbau, anders als im Frontend, nicht so sehr durch irgendein Framework bestimmt.



Api

Enthält die Endpunkte für das REST-API

Db

Alles in Bezug auf die Datenbank (Scripts, Zugriff etc.)

Index.js

Hauptdatei des Backends, importiert alle Dependencies und startet den Backend-Server

Abbildung 16:
Ordnerstruktur Backend

19.3 Datenbank

Die Datenbank wird über das File «knexfile.js» im Ordner /backend/db verbunden (siehe Ordnerstruktur). Das folgende Code-Snippet soll dies am konkreten Fall verdeutlichen.

```
module.exports = {  
  development: {  
    client: 'mysql',  
    connection: {  
      database: 'terminverwaltung_ipa',  
      user: [REDACTED],  
      password: [REDACTED]  
    },  
  },  
};
```

Es wird zuerst der Client angegeben, im Falle dieses Projekts «MySQL». Danach wird die eigentliche Verbindung hergestellt. Der Name der Datenbank, der User und das Passwort werden übergeben.

Über diese Angaben wird nun auf dem Entwicklungslaptop die Verbindung zwischen Webapplikation und lokalem Datenbankserver hergestellt. In einer produktiven Umgebung müssten also nur diese Zeilen Code geändert werden, um zu einem richtigen Datenbankserver zu wechseln.

19.3.1 Erstellung der Datenbank und Tabellen

Das Erstellen der einzelnen Tabellen ist über zwei Wege möglich. Einerseits können Tabellen über Knex.js mittels sogenannten Migrations erstellt werden. Das gleiche kann man jedoch auch mit einem gewöhnlichen SQL-Skript machen. Für dieses Projekt können theoretisch beide Wege verwendet werden.

Das Login weist keine Relationen zu irgendwelchen anderen Tabellen auf. Die Login Tabelle soll, die vom Admin an die Zielgruppe auszustellenden Login-Daten enthalten. Das heisst diese Tabelle braucht eine Umladung. Da Knex.js die Funktion von Umladungen (Knex Seeds) beinhaltet, ist es am einfachsten die Tabelle für die Login-Funktion komplett isoliert über den Knex.js Dienst zu erstellen und mit Daten zu füllen.

Die anderen Tabellen werden über ein klassisches SQL-Skript erstellt, wobei es sogar möglich wäre, diese über das MySQL-Workbench GUI zu erstellen.

Im Folgenden ist das Seed-File für die User des Logins abgebildet. Es lässt sich erkennen, dass auf die Tabelle «Users» zugegriffen wird. Es werden Daten mit «insert» in die Tabelle geschrieben. Die Syntax zeigt, dass immer zuerst der Spaltenname der Tabelle, und danach der einzusetzende Wert geschrieben wird.

```
exports.seed = function (knex) {  
  // Deletes existing data  
  return knex('users').del()  
    .then(function () {  
    // Inserts new Data  
    return knex('users').insert([  
      { username: 'admin', firstname: 'John', lastname: 'Doe', password: bcrypt.hashSync('password', 10) },  
    ]);  
  });  
};
```

Im Projekt gibt es nur das eben gezeigte Seed-File. Für das Login macht es Sinn ein Seed-File zu verwenden, da es so möglich ist, die Projektumgebung viel schneller einzurichten. Die anderen Tabellen werden über klassische SQL-Skripts mit Daten gefüllt.

19.4 Login

Wie wurde die Login-Funktion umgesetzt und welche Technologien wurden dabei verwendet?

19.4.1 Ablauf eines Login Vorgangs

Der Client empfängt im Frontend die vom User eingegebenen Login-Daten. Diese schickt er mit «Axios» an das Backend. Im Backend werden die Logindaten entgegengenommen und mit allen Logindaten in der MySQL-Datenbank verglichen. Da in dieser Applikation ein auf Token basierendes Login und keine Sessions verwendet werden, braucht es zwingend ein Token. Gibt es eine Übereinstimmung, schickt das Backend einen Token an das Frontend (jsonwebtoken). Die folgende Grafik beschreibt in einem Sequenzdiagramm diesen Vorgang.

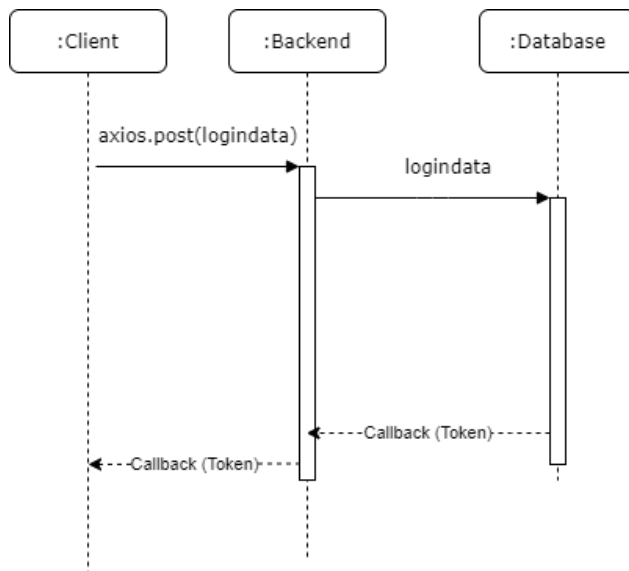


Abbildung 17: Sequenzdiagramm Login

Wie man in dem Diagramm gut sieht, verläuft ein Login-Vorgang sehr simpel. In dem Vorgang werden vor allem «Axios» und «jsonwebtoken» gebraucht. Der Token wird bei erfolgreichem Login im Localstorage des Browsers gespeichert. Er hat eine gewisse Gültigkeitsdauer, welche im Code bestimmt werden kann. Im Folgenden wird der genaue Aufbau von einem Token beschrieben.

19.4.2 JSON-Webtoken

Ein Token besteht aus einem Header und der Payload. Ausserdem beinhaltet er eine Signatur Key. Der Header besteht aus Informationen, wie dem verwendeten Algorithmus und dem Typ des Tokens. Die Payload besteht aus den verschlüsselten Daten sowie dem Ablaufdatum. Ausserdem wird die Payload mit dem Signatur Key signiert.

Im Falle dieser Webapp besteht die Payload nur aus den Token Daten. Der Signatur Key befindet sich als Umgebungsvariable auf dem Backend Server.

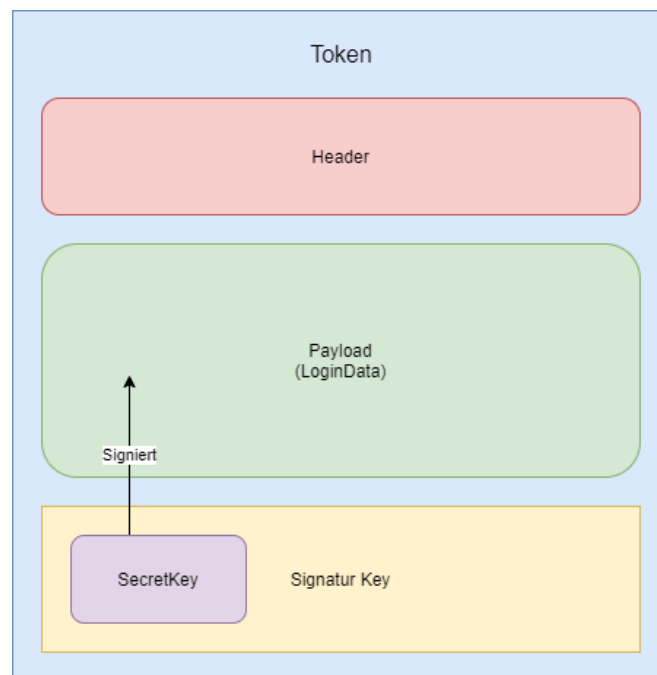


Abbildung 18: Token

In der Datei `jwt.js` im Verzeichnis `/lib/jwt.js` wird der Token definiert. Hier werden alle Parameter angegeben, wie die Lifespan (Lebensdauer/Gültigkeit) und die Verschlüsselung.

Es wird eine sogenannte «base64» Verschlüsselung verwendet, um den Token zu verschlüsseln und wieder zu entschlüsseln. Für die Lebensdauer wird das aktuelle Datum beim Erstellen des Tokens genommen, und plus 1000 Sekunden gerechnet. Das heisst ein Token ist ca. 15 Minuten gültig, bevor er erneuert werden muss. Wie das Ganze im Code umgesetzt wurde, wird nun im Folgenden veranschaulicht.

Alle, Files die sich auf den Token beziehen, werden automatisch generiert.

Die Funktion «CreateToken» generiert einen neuen Token mit den gewünschten Parametern, wie etwa der Lebensdauer und dem Algorithmus des Headers.

```
function createToken(payload, lifetime) {  
  let token = { header:{}, payload:{} }  
  token.header.alg = "HS256"  
  token.header.typ = "JWT"  
  token.payload = payload  
  token.payload.iat = Date.now()  
  token.payload.exp = token.payload.iat + 1000*lifetime  
  return signToken(token)  
}
```

Die Funktion «Encode» verschlüsselt den Token über die «base64Url». Dabei müssen noch einige Sonderzeichen ersetzt werden, da sonst eine Verschlüsselung nicht möglich wäre.

```
function encodeBase64Url(string) {  
  let base64Url = Buffer.from(string).toString('base64')  
    .replace(/\+/g, '-')  
    .replace(/\//g, '_')  
    .replace(/=+$/, '')  
  return base64Url  
}
```

19.4.3 Bcrypt.js

Damit die Logindaten während dem ganzen Prozedere verschlüsselt sind, braucht es die Package-Extension «Bcrypt». Mithilfe dieser können Logindaten sowohl in der Applikation, wie auch später in der Datenbank zuverlässig ver- und entschlüsselt werden.

Bcrypt basiert auf dem Blowfish-Algorithmus und ist somit sehr sicher, da es viel Rechenzeit benötigen würde, den Algorithmus zu entschlüsseln. Über eine Bruteforce-Attacke an Passwörter zu gelangen, wird so beinahe unmöglich.

Der Prozess, mit «bcrypt» ist sehr einfach. Da die Login-Daten (wie in 18.3.1 beschrieben), per Seed-File in die DB gespeichert werden, ist eine Modifizierung dieses Files notwendig. Nachdem bcrypt importiert wurde, muss nur eine Zeile Code geändert werden.

```
return knex('users').insert([
  { username: 'admin', firstname: 'John', lastname: 'Doe',
    password: bcrypt.hashSync('12345') },
]);
```

Wie man im Code sehen kann, wird mit «bcrypt.hashSync» das Passwort verschlüsselt in der DB gespeichert. In der folgenden Grafik ist dies ersichtlich.

	id	username	firstname	lastname	password
	3	admin	John	Doe	\$2a\$10\$8yUTDI0mIBCCfNoJeeN86eAqFQvMzP2hLMS290mSgrYh3.9oghQAa
»*	NULL	NULL	NULL	NULL	

Abbildung 19: DB-Eintrag verschl. Passwort

Wie in der Grafik ersichtlich wird, ist das Testpasswort, welches in die DB gespeichert wurde, nun verschlüsselt.

19.5 Frontend

Wie wurde das Frontend umgesetzt und wie ist der strukturelle Aufbau der Komponenten?

19.5.1 Aufbau der Komponenten

Das Frontend wird über folgende Commands gestartet.

Befehl	Ausführungspfad	Zweck (Beschrieb)
npm run serve	/frontend/	Startet den Frontend Development Server.

Tabelle 52: Commands - Frontend

Das Frontend ist im Ordner /Frontend/src/ zu finden. Das Vue.js Framework ist in das App.vue-File und die einzelnen Views unterteilt. Das App.vue-File ist eine Art Hauptfile für das Frontend. Alles was hier drin ist, erscheint auf jeder Seite. Einzelne Seiten sind Views. Für dieses Projekt brauche ich drei Views, eine Seite «Termine» und eine für den «Kalender». Die letzte braucht es schliesslich noch für das Login.

Im App.vue-File befindet sich die Navigationsleiste, da diese auf jeder Seite benötigt wird. Ebenfalls im App.vue-File ist der Logout-Button, da auch dieser auf jeder Seite sein soll.

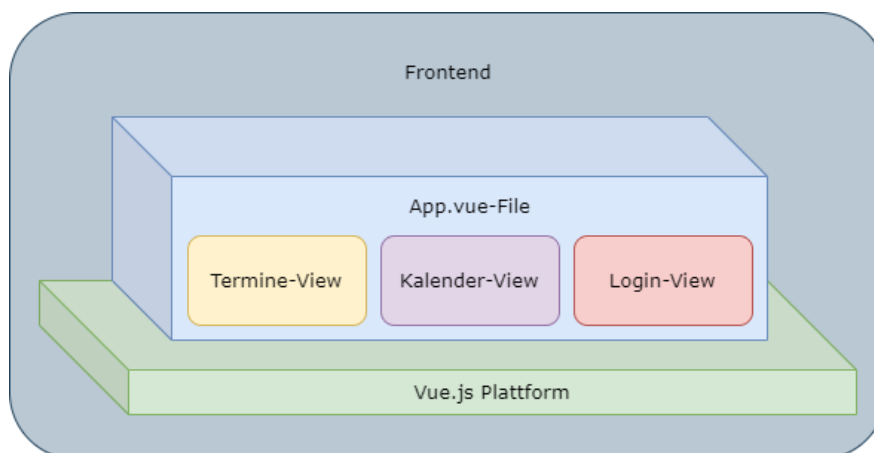


Abbildung 20: Views und App.vue

Wie in der Grafik gut zu sehen ist, besteht das ganze Frontend aus drei Views und dem App.vue-File. Folglich wird im Frontend faktisch nur an diesen vier Dateien gearbeitet.

19.5.2 Router im Frontend

Damit die Navigation in der Applikation funktioniert, wurde mit «Vue-Router» eine Navigations-Struktur erstellt.

Im Ordner «router» im Frontend liegt das File «index.js». In diesem wird der Router initialisiert.

Achtung, dieses File darf nicht verwechselt werden mit dem «index.js»-File im Backend!

Das File enthält alle Navigationspfade, welche in der Applikation benötigt werden. Im Folgenden wird der Code etwas genauer erläutert. Als erstes müssen die einzelnen Komponenten (Views) importiert werden.

```
import Vue from 'vue'
import VueRouter from 'vue-router'
import Events from '../views/Events.vue'
import Calendar from '../views/Calendar.vue'
import Login from '../views/Login.vue'
```

Nachdem die nötigen Imports abgeschlossen sind, werden die «routes» der URL definiert. In der folgenden Tabelle sind alle Routes beschrieben. Die Seite ist die View, welche bei der entsprechenden URL geladen wird.

Pfad (URL)	Seite (View)	Beschreibung
/	Login.vue	Wenn kein Pfad eingegeben wird, etwa beim ersten Aufrufen der App, wird die Login Seite geladen.
/calendar	Calendar.vue	Es wird zum Kalender geroutet.
/events	Events.vue	Es wird zur Seite Events geroutet.
/login	Login.vue	Es wird zur Login Seite geroutet.

Tabelle 53: Router

Konkret sieht eine definierte Route wie im Folgenden aus (Beispiel «Calendar.vue»).

```
{
  path: '/calendar',
  name: 'Calendar',
  component: Calendar
},
```

19.5.3 Frontend – Endpoints

Die Events-Ansicht soll erstellte Terminsets vom User entgegennehmen. Dafür muss ein Endpoint mit der POST-Methode erstellt werden. Da Termine auch bearbeitet oder gelöscht werden, müssen ebenfalls weitere Endpoints definiert werden. In der folgenden Tabelle sind alle gebrauchten Endpoints beschrieben.

Pfad	Methode	Payload	Beschreibung
/api/event	Fetch-GET	Table "Events"	Holt die ganze Tabelle "Events" aus der DB, um diese im Frontend anzuzeigen.
/api/event/ \${item.id}	Fetch-DELETE	Table "Events" - Row where "ID" matches	Bestimmten Termin über die ID löschen.
/api/event/ \${this.editedItem.id}	Fetch-PUT	Updated Row in Table "Events"	Bereits vorhandenen Termin ändern
/api/event	Fetch-POST	New Row in Table "Events"	Neues Terminset speichern
/api/get_class	Axios-GET	Classnames in "Classes"	Holt alle Klassen aus der Tabelle "Classes" um diese im Frontend anzuzeigen
/api/get_students	Axios-GET	Students from selected "Class"	Holt alle Schüler der ausgewählten Klasse

Tabelle 54: Endpoints

Die erstellten Endpoints funktionieren im Moment noch nicht. Diese müssen erst noch im Backend definiert werden.

Da die in einem Terminset benötigten Daten in einer separaten Tabelle ohne Relation zur «Events»-Tabelle gespeichert sind, wurde entschieden die entsprechenden Endpoints zu trennen. Das heisst konkret, dass Daten wie Schüler und Klassen über separate Endpoints in die «Events»-View geholt werden. Für diese Endpoints wurde nicht Fetch, sondern Axios verwendet (siehe obere Tabelle). Dies dient vor allem der Übersichtlichkeit im weiteren Verlaufe. Die Axios Endpunkte werden später beim Erstellen des Backends in einem getrennten File gespeichert.

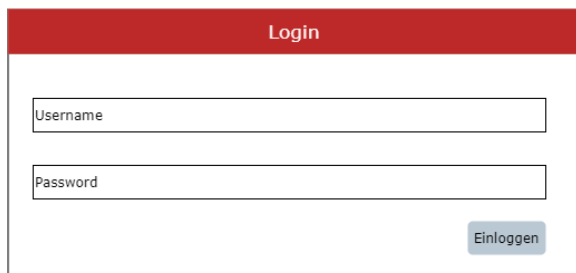
19.5.4 Login.vue (Login Seite)

Der Aufbau der Login Seite, bzw. Funktion ist sehr einfach. Es gibt einen Button «Einloggen». Wird dieser gedrückt, wird die Methode «Login» aufgerufen, welche die Credentials ans Backend schickt. Das View-File für das Login ist relativ unspektakulär und wird deshalb nicht allzu genau beschrieben.

In der Login-Methode wird der Webtoken gesetzt und es wird die Seite «Events.vue» geladen (bei erfolgreichem Login).

```
if (data.token) {  
  localStorage.setItem('token', data.token)  
  this.$router.push('/events')  
} else {  
  console.log("error");  
}
```

Der Aufbau der Login-Maske gestaltet sich relativ einfach. Hier können Standard-Komponenten von Vuetify verwendet werden. Die Mockups werden als Anhaltspunkt genommen und möglichst ähnlich so auch umgesetzt. Die folgende Abbildung vergleicht die reale Version mit dem vorher konzeptionierten Mockup.



Das Mockup zeigt eine Login-Maske mit einem roten Header 'Login'. Darunter befinden sich zwei Eingabefelder: 'Username' und 'Password'. Ein grauer Button mit der Aufschrift 'Einloggen' ist rechts unten positioniert.

Abbildung 23: Konzeptionelles Mockup



Das implementierte Login-Formular ist optisch fast identisch mit dem Mockup. Es hat einen roten Header 'Login', zwei Eingabefelder für 'Username' und 'Password' sowie einen grauen Button 'EINLOGGEN'.

Abbildung 24: Implementiertes Login

Wie man anhand der Grafiken erkennen kann, weicht die implementierte Version optisch kaum von dem entsprechenden Mockup ab.

Als Anmerkung kann man noch sagen, dass es in Zukunft ebenfalls problemlos möglich wäre eine Registrierungs-Funktion zu implementieren.

19.5.5 Events.vue (Terminansicht) - Design

Wie im Kapitel «Aufbau der Komponenten» beschrieben, gibt es drei Views. Die grösste und wichtigste davon ist Events.vue. Hier werden die Terminsets verwaltet und tabellarisch dargestellt.

Ein View-File ähnelt im Aufbau einer normalen HTML-Seite. Es gibt drei Haupttags, die im Folgenden kurz beschrieben werden.

```
<template>
```

Im «Template»-Tag befinden sich Design und Strukturkomponenten. Hier wird vor allem das Designframework Vuetify gebraucht.

```
</template>
```

```
<script>
```

Im «Script»-Tag findet sich der Vue spezifische JavaScript-Code.

```
</script>
```

```
<style>
```

Im «Style»-Tag lassen sich Designelemente mit CSS beschreiben.

```
</style>
```

In der Events-View braucht es eine Tabelle, welche im Mockup «13.7.2 Kalender Seite» konzeptioniert wurde. In dieser Tabelle werden alle Terminsets gelistet. Umgesetzt wird dies wie im Folgenden mit `<v-data-table>`.

Damit die Anordnung stimmt, werden weitere Komponenten geschachtelt aufgebaut. Als Parameter wird `<v-data-table>` der Headers-Array übergeben. So werden die entsprechenden Überschriften und die zugehörigen Daten angezeigt.

```
<template>
  <v-data-table :headers="headers" :items="projects" sort>

    <v-toolbar flat color="white">
      <v-dialog v-model="dialog">
        <template v-slot:activator="{ on, attrs }">
          <v-btn color="#b47bff">
            <v-icon>mdi-plus</v-icon>
          </v-btn>
        </template>
      </v-dialog>
    </v-toolbar>

    <template v-slot:[`item.actions`]="{ item }">
      <v-icon class="mr-2" @click="editItem(item)">mdi-pencil</v-icon>
      <v-icon @click="deleteItem(item)">mdi-delete</v-icon>
    </template>

  </v-data-table>
</template>
```

Innerhalb der Tabelle findet sich eine Überschriften-Zeile, welche mit `<v-toolbar>` gebaut wurde. Innerhalb dieser Toolbar befindet sich ein Button, über welchen ein Dialog aufgerufen wird. Dieser Button ist das im Mockup definierte «+». Über diesen Button kann ein neues Terminset gepusht werden.

Die in der Tabelle angezeigten Termine sollen, wie im Mockup definiert, bearbeitet und gelöscht werden können. Dafür werden unterhalb der Toolbar in einem Template-Tag zwei `<v-icon>`-Elemente definiert. Dies sind die kleinen Editierungs-Buttons, welche jeweils rechts von einem Termin erscheinen. Wenn man diese drückt, werden die Methoden «editItem», bzw. «deleteItem» aufgerufen.

Die Symbole können durch Vuetify sehr einfach implementiert werden. Der Code «mdi-pencil» innerhalb des `<v-icon>`-Tags, beschreibt beispielsweise einen kleinen Stift für die Editierung eines Termins.

19.5.6 Events.vue - Methoden

In der App gibt es einige Problemstellungen, welche über Methoden im Frontend gelöst werden müssen. In der Tabelle «Schüler» befinden sich etwa 60 Schüler verteilt auf drei Klassen. Die Schwierigkeit liegt darin, dass ein Termin mit beispielsweise vier zugeteilten Schülern nie dieselben Schüler enthalten darf. Zusätzlich muss bei einem Terminset mit mehreren Schülern sichergestellt werden, dass alle Schüler etwa gleich oft zugeteilt werden. Diese Problemstellung wird in der folgenden Grafik noch einmal verdeutlicht

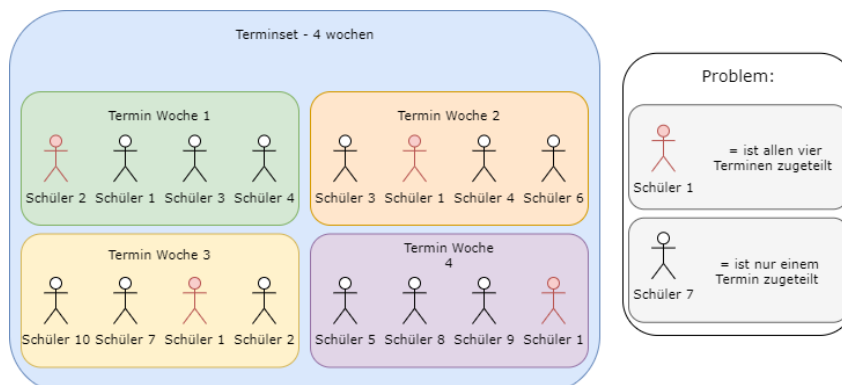


Abbildung 21: Problemstellung

Die Grafik zeigt das entscheidende Problem sehr deutlich. Angenommen eine Lehrkraft erstellt ein Terminset welches vier Wochen dauert. Die Lehrkraft setzt die Gruppengrösse auf vier Schüler. Das Problem ist, dass ein Schüler nicht öfters drankommen darf als ein anderer. In der oberen Grafik wird der Schüler 1 öfters eingeteilt als die anderen Schüler. Dieses Problem darf in der Applikation nicht auftauchen.

Der Lösungsansatz in der Praxis sieht so aus, dass es eine Methode gibt, welche bereits drangekommene Schüler aus dem Array aller Schüler löscht und diesen anschliessend neu sortiert. Mit dieser Lösung sollte das Problem nicht mehr vorkommen. Wie genau die Lösung im Detail aussieht, wird jetzt beschrieben.

Es wurden zwei Methoden erstellt, um das Problem zu lösen. Die Methode «SetStudents» teilt die richtige Anzahl an Schülern ein. Diese ist abhängig von der gewählten Gruppengrösse.

```
setStudents() {
  let groupsize = this.editedItem.groupsize;

  if (groupsize == 2) {
    this.editedItem.student1 = this.students[this.actualIndex];
    this.setNextIndex();
    this.editedItem.student2 = this.students[this.actualIndex];
    this.setNextIndex();
  }
}
```

Wie man in der Methode sieht, wurde eine if-Abfrage für die Gruppengrösse erstellt. Abgebildet ist hier nur der Fall «groupsize == 2», also eine Gruppengrösse von zwei Schülern. Für jeden Fall wurde ein if-Statement deklariert, also insgesamt vier Optionen (mögliche Gruppengrösse: 1,2,3,4).

In dem if-Statement sieht man, das «this.editedItem.student1» ein Wert aus dem Array «this.students» zugewiesen wird. Das gleiche passiert dann auch mit dem zweiten Schüler (editedItem.student2). Nach jeder Zuteilung wird die Methode «this.SetNextIndex» aufgerufen. Diese iteriert «this.actualIndex», so dass immer der nächste Schüler im Array genommen wird. Um die verwendeten Variablen und Arrays besser zu verstehen, werden diese in der folgenden Tabelle etwas genauer erklärt.

Name	Typ	Zweck
groupsize	Variable	Enthält die vom User eingegebene Gruppengrösse.
this.editedItem((Parameter))	Object-Array	Dieses Objekt enthält die vom User eingegebenen Parameter eines zu erstellenden Terminsets.
this.students	Array	Enthält alle Schüler der gewählten Klasse
this.actualIndex	Variable	Index welcher nach jedem gesetzten Schüler iteriert wird.

Tabelle 55: Variablen Events.vue

Die komplexeste Methode in der «Events»-View, ist die «save»-Methode. Sie ruft die eben erklärte Methode auf. Um den ganzen Prozess einer Speicherung eines Terminsets zu veranschaulichen, wurde das folgende Sequenzdiagramm erstellt.

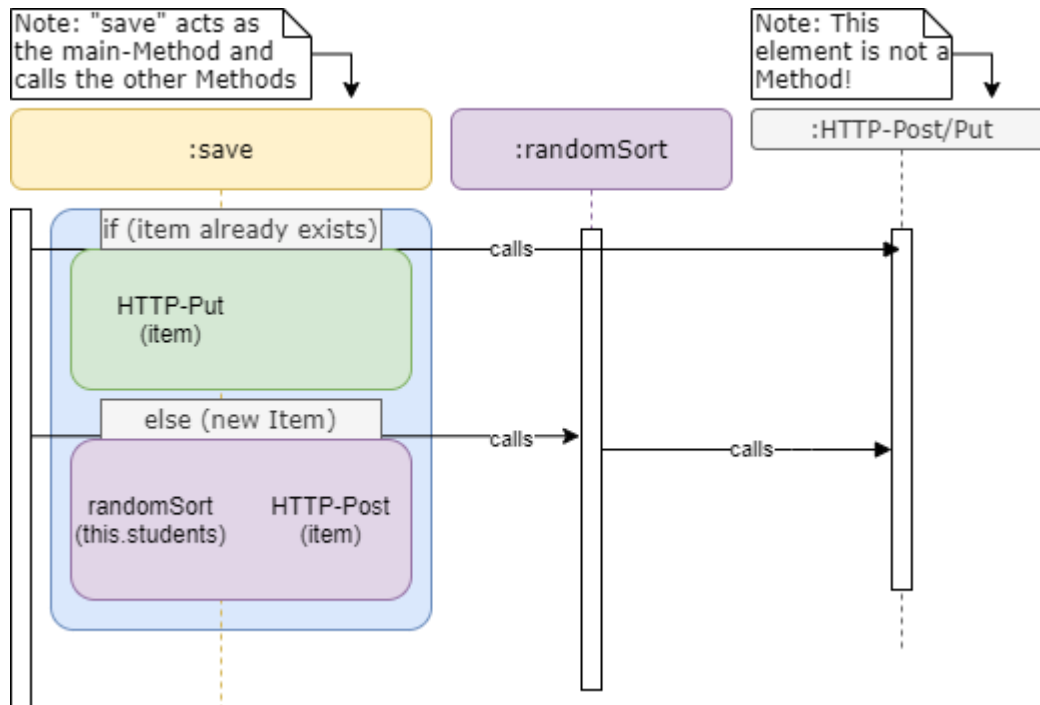


Abbildung 22: Save-Methode (Sequenzdiagramm))

Im Diagramm sind die drei Lebenslinien zu sehen. Die ersten beiden sind Methoden, die letzte ist nur zur Veranschaulichung abgebildet.

Wenn ein User alle Parameter eingibt, um ein Terminset zu erstellen, drückt er auf «Speichern». Es wird die Methode «save» aufgerufen, welche zuerst mit einem if-Statement prüft, ob hier ein komplett neues Terminset erstellt wird, oder ob ein vorhandenes bearbeitet wird. Im Falle eines neuen Terminsets wird der «else»-Teil ausgeführt (Grafik oben «else»). In diesem Teil des Statements wird zuerst die Methode «randomSort» aufgerufen (weiter unten beschrieben). Diese aufgerufene Methode sortiert die zugeteilten Schüler. Danach wird die gewählte Zeitspanne genommen und mit vier multipliziert (Monate zu Wochen).

```

let countWeeks = this.editedItem.timespan * 4;
this.actualIndex = 0;
  
```

Danach wird in einer for-Schleife so oft ein Termin gepusht (HTTP-Post), wie der User mit dem Parameter «Zeitspanne» gewählt hat. Dabei wird jedes Mal die Methode «setStudents» (oben beschrieben) erneut aufgerufen, so dass immer neue Schüler eingeteilt werden.

19.5.7 Calendar.vue (Kalenderansicht)

Für die Kalender-Ansicht wird eine eigene View gebaut. Wie im Mockup geplant, braucht es eine zwei-Monatsansicht über einen grafischen Kalender.

Der grundsätzliche Aufbau mit der «v-calendar» Komponente sieht wie im Folgenden aus.

```
<v-calendar
  ref="calendar"
  v-model="focus"
  color="primary"
  :events="events"
  :event-color="getEventColor"
  :type="type"
  @click:event="showEvent"
  @click:more="viewDay"
  @click:date="viewDay"
>
```

Das «v-calendar»-Objekt wird zwei Mal gebraucht, da jede Ansicht eines Monats eine eigene Instanz ist. Das ist das Grundgerüst des Kalenders, dieser enthält noch keine Daten.

Um den Kalender mit Daten zu füllen, wird eine Methode «Initialize» erstellt, mit welcher die zuvor erstellten Termine aus der DB geholt werden. Diese Methode ist «mounted», das heisst sie wird jedes Mal aufgerufen, wenn man die Seite neu lädt.

```
initialize () {
  fetch('api/event')
    .then(response => response.json())
    .then(data => {
      data.forEach(element => {
        let event = {
          name : element.eventname,
          start : element.date.substr(0, 10),
          color: this.colors[this.rnd(0, this.colors.length - 1)],
          timed: false,
          details1: element.class,
          details2: element.groupsize,
          details3: element.student1,
          details7: element.teacher }
        this.events.push(event)
      })
    })
}
```

Die «Initialize»-Methode holt alle Records aus der Tabelle «Events». Diese Daten werden zu einem Objekt konvertiert, welches die Columns als Attribute eines Termins entgegennimmt. In dem Array «Events» sind im Anschluss alle Termine gespeichert. Ein Termin hat eine Farbe, welche per Zufall generiert wird. Für die Zweimonatsansicht wird in einer Stringvariablen ein Datum gesetzt, welches einen Monat später ist als der erste Kalender. Dieser Code wird ebenfalls im «mounted»-Teil geschrieben.

```
let oneMonthLater = new Date(Date.now());
oneMonthLater.setMonth(oneMonthLater.getMonth() + 1);
this.secondFocus = oneMonthLater;
```

Das gesetzte Datum wird in die Variable «this.focus2» gepusht, welches das Startdatum für den zweiten Kalender ist.

Damit man die Monatsansicht auf die nächsten, bzw. auf die letzten, Monate umstellen kann, braucht es hier noch zwei Buttons und jeweils eine Methode, welche die Startdaten verschiebt.

```
<v-btn
  fab
  text
  small
  color="grey darken-2"
  @click="prev"
>
```

Dies ist der Button, welcher die Monatsansicht zurückschiebt (previous). Einen gleichen Button gibt es, um das Datum nach vorne zu schieben («next»). Die Buttons lösen jeweils eine Methode aus (@click). Die Methode addiert (bzw. subtrahiert) einen Monat zum aktuellen Datum.

```
prev () {
  //Two random Dates are generated
  let firstStart = new Date(this.firstFocus);
  let secondStart = new Date(this.secondFocus);
  //Subtracts one Month from each Date
  firstStart.setMonth(firstStart.getMonth() - 1);
  secondStart.setMonth(secondStart.getMonth() - 1);
  //The new dates are now in the "Focus" Variable for the Calendar
  this.firstFocus = firstStart;
  this.secondFocus = secondStart;
},
```


19.5.8 App.vue (Hauptfile Frontend)

Die bisher beschriebenen Teile des Backends waren die sog. «Views» (siehe Kap.18.5.1 Aufbau der Komponenten). Diese werden nun im «App.vue» zusammengefasst.

Alles was im «App.vue»-File geschrieben wird, erscheint jeweils auch auf jeder «View». Das heisst es macht Sinn Komponenten wie etwa die Navigationsleiste oder den «Logout»-Knopf in dieses File zu schreiben. Dies wurde auch genauso umgesetzt und wird im Folgenden beschrieben. Als erstes wird die Navigationsleiste gemacht (folgender Code).

```
<router-link to="/Events">
  <v-btn value="recent">
    Termine
  </v-btn>
</router-link>

<router-link to="/Calendar">
  <v-btn value="favorites">
    Kalender
  </v-btn>
</router-link>

<v-btn v-if="isLoggedIn" href="/" @click="logout" text>
  <span class="secondary--text mr-2" prepend-icon="mdi-
folder">Logout</span>
  <v-icon v-if="isLoggedIn">mdi-logout</v-icon>
</v-btn>
```

Wie man im Code gut erkennen kann, gibt es drei Buttons, welche jeweils als Router-Link definiert sind (siehe 18.5.2 Router). Das letzte Element im Tag «v-btn» ist nur zu sehen, wenn man eingeloggt ist. Dies hat den Grund, dass man den Button «Logout» nur sehen soll, wenn man tatsächlich eingeloggt ist.

Die ersten zwei Buttons sind die normale Menüführungsleiste, wie im Mockup definiert wurde.

19.5.9 Vergleich mit konzeptionellen Mockups

Die Seite «Events.vue» konnte dem konzeptionierten Mockup ähnlich gestaltet werden. Im Folgenden wird ein Vergleich des tatsächlichen Resultats mit dem konzeptionierten Mockup vollzogen.

Die erste Grafik zeigt jeweils das Mockup (übernommen aus «Konzept»), und die zweite Grafik zeigt einen Screenshot des produktiv umgesetzten Resultats.

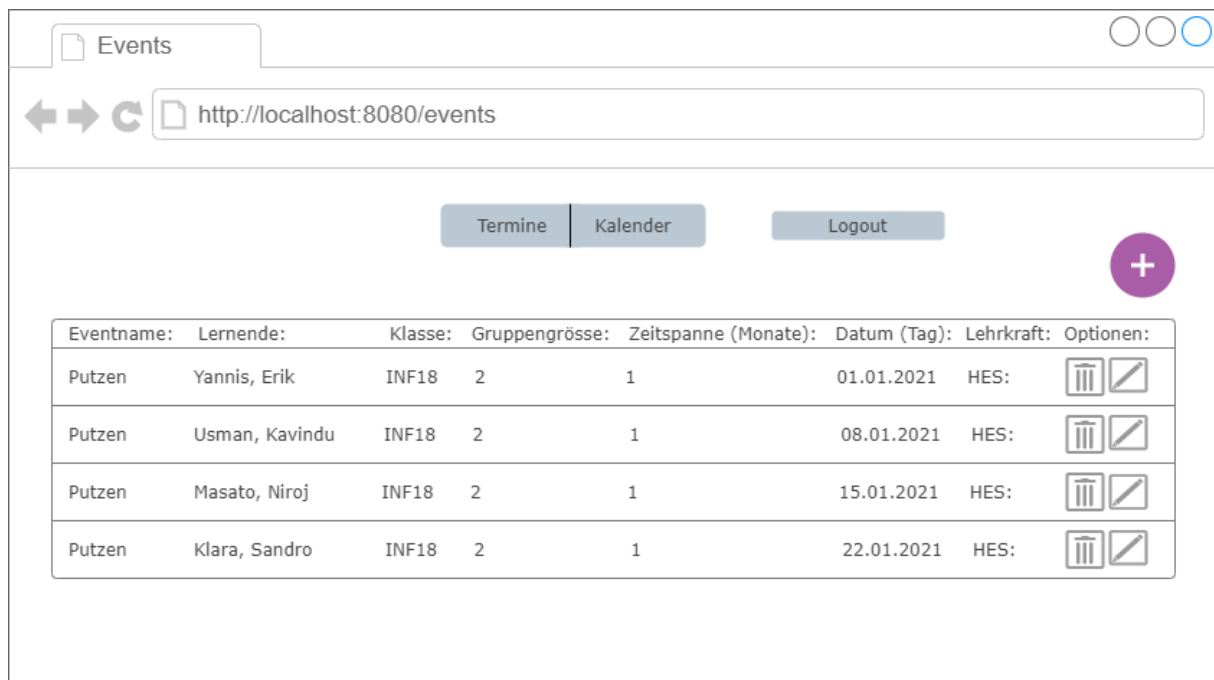


Abbildung 23: Mockup Events.vue

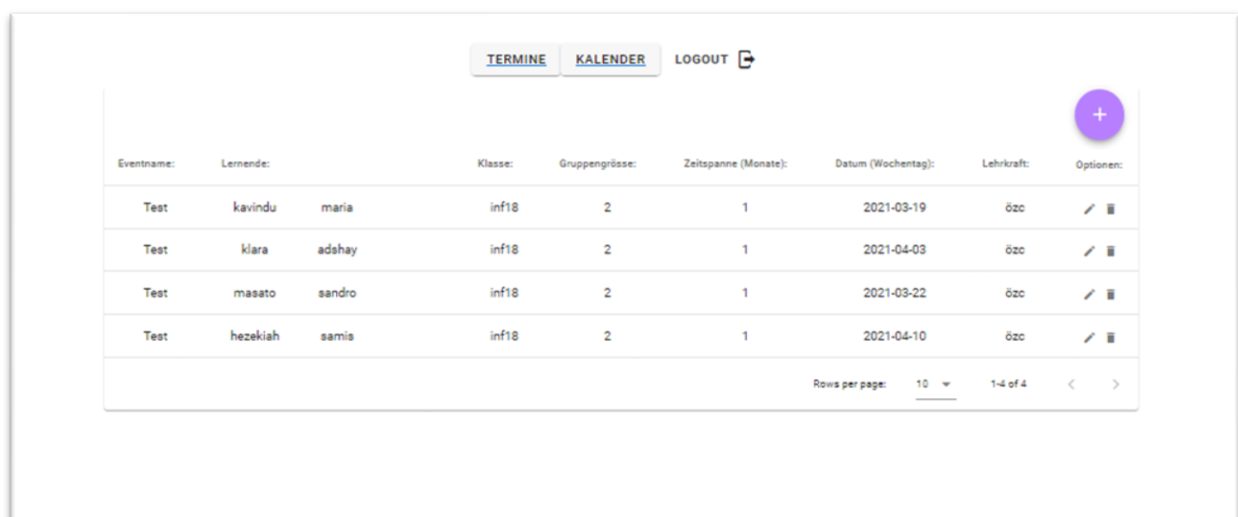


Abbildung 24: Umgesetztes Design Events.vue

Bei der Seite «Kalender» (calendar.vue) gab es einige Abweichungen zum Mockup. In dem konzeptionellen Mockup war eine Kalenderansicht mit zwei Monaten nebeneinander geplant. In der Umsetzung hat sich jedoch gezeigt, dass eine Ansicht mit übereinander angeordneten Elementen erheblich benutzerfreundlicher ist.

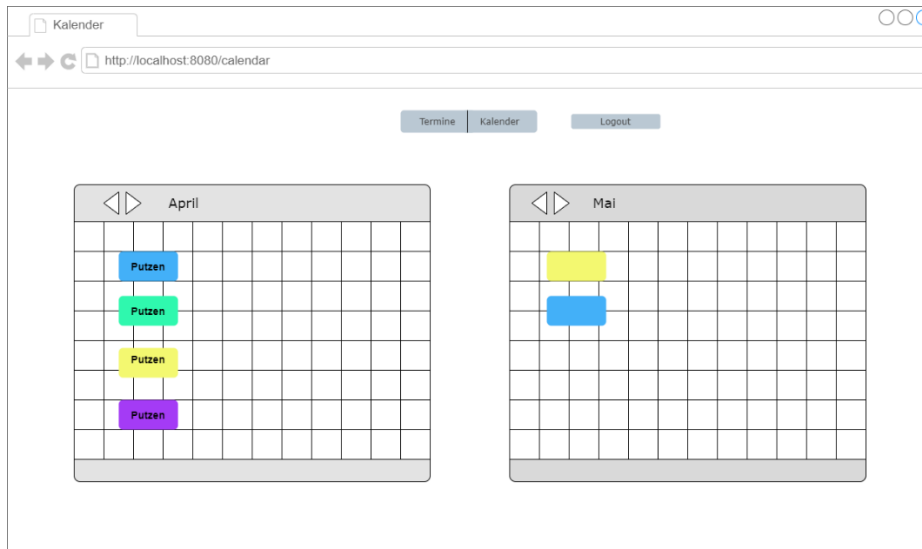


Abbildung 24: Mockup calendar.vue

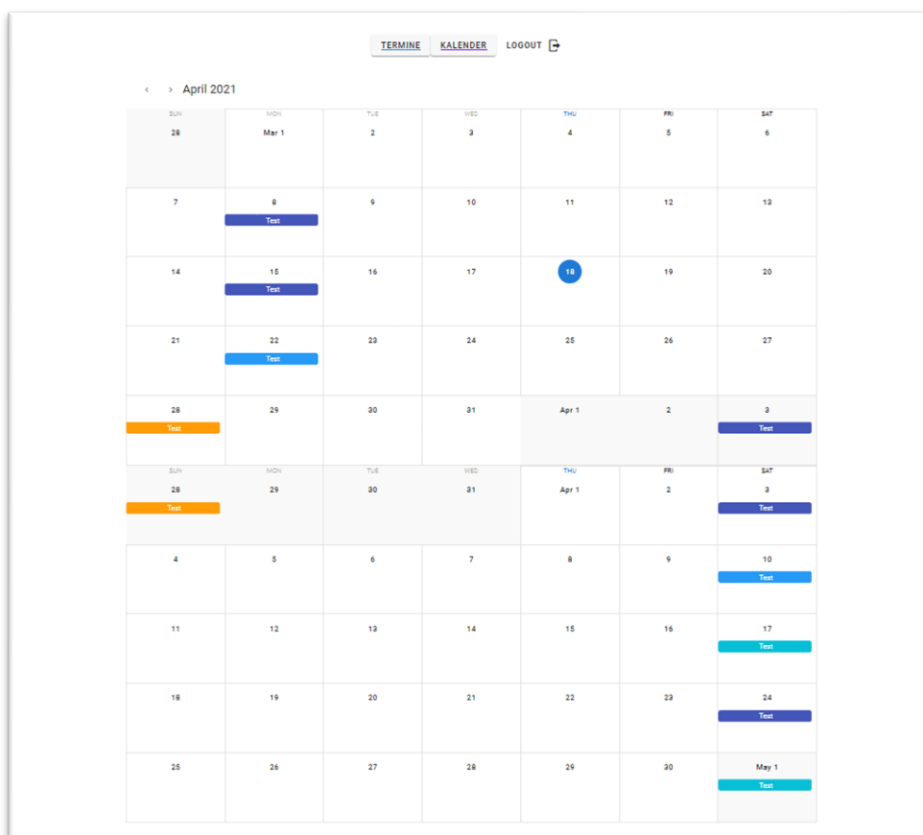
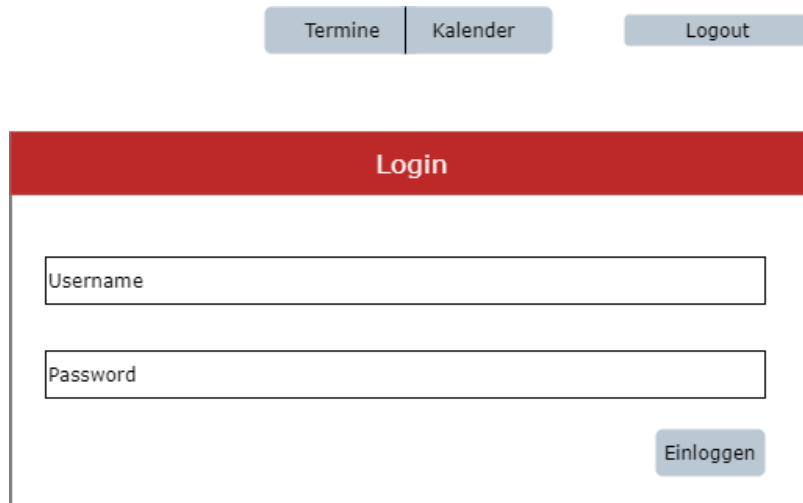


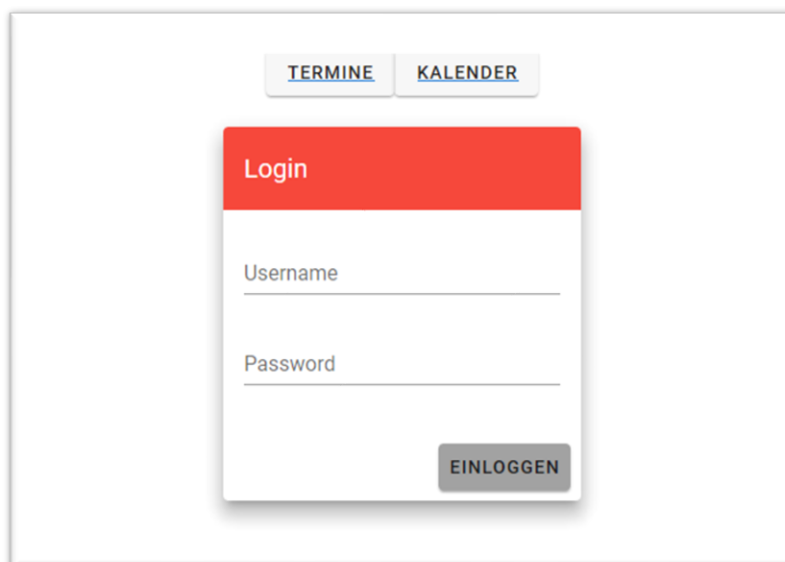
Abbildung 23: Umgesetztes Design calendar.vue

Die Login-Seite ist dem Mockup äusserst ähnlich gestaltet. Es gibt kaum nennenswerte Abweichungen und die Farbwahl stimmt ebenfalls mit dem Mockup überein.



The mockup shows a header with three buttons: 'Termine', 'Kalender', and 'Logout'. Below this is a red 'Login' header. The main form contains two input fields labeled 'Username' and 'Password', and a blue 'Einloggen' button.

Abbildung 25: Mockup Login.vue



The implemented version shows the same layout as the mockup. The navigation buttons 'TERMINE' and 'KALENDER' are in a light blue box. The 'Login' form has a red header, white input fields for 'Username' and 'Password', and a grey 'EINLOGGEN' button.

Abbildung 26: Umgesetzte Version Login.vue

19.6 Backend

Wie ist das Backend aufgebaut, und welche Komponenten gibt es?

19.6.1 index.js – Hauptfile im Backend

Das Hauptfile des Backends ist das File «index.js». Über dieses wird der Backendserver gestartet. Im Folgenden werden die wichtigsten Commands des Backends kurz beschrieben.

Befehl	Ausführungspfad	Zweck (Beschrieb)
nodemon	/backend/	Startet den Backendserver so, das dieser bei jeder Änderung automatisch neu gestartet wird.
node index.js	/backend/	Führt das File "index.js" aus und startet so das Backend. Nach Änderungen muss jedoch manuell neu gestartet werden.

Tabelle 56: Commands - Backend

Im index.js-File werden zuerst alle Dependencies verknüpft. Danach wird der Port für das Backend bestimmt und der Entwicklungsmodus wird festgelegt. Dieser ist auf «Development» gesetzt, da sich die App noch in der Entwicklungsphase befindet. Das «knexfile» wird verknüpft, um die Verbindung zur Datenbank herzustellen (siehe Kap. 17.3 «Datenbank»).

```
const port = 80
const enviroment = process.env.NODE_ENV || 'development';
const config = require('./db/knexfile.js').development;
app.listen(port, () => console.log(`App listening on port: ${port}!`));
```

Es gibt, wie in «17.5.4 Endpoints» beschrieben, noch ein weiteres File im Backend, welches die meisten Endpoints von «Events.vue» enthält. Dieses File muss jedoch noch im Hauptfile (index.js) verlinkt werden.

```
app.use('/api/*', express.json())
app.use('/api/event', require('./api/event'))
```

19.6.2 Backend - Endpoints

Die Endpoints sind, wie bereits erwähnt, auf zwei Files aufgeteilt. Im «index.js»-File ist der erste End Point für die Login Funktion. Wie bei «17.5.2 Login» bereits erwähnt, wird dieser Endpunkt ausgeführt, wenn ein User auf «Einloggen» drückt.

```
app.post('/api/user/login', async (req,res)=> {
  let credentials = req.body
  let answer = await knex('users')
    .where('username', credentials.username)
  let user = answer[0]
  let test = bcrypt.compareSync(credentials.password, user.password)
  if (test) {
    // if user is not in database it will be null
    let payload = {username:user.Username, role:user.role}
    let token = jwt.sign(payload, pepper)
    res.json({
      token: token
    })
  }
  else {
    res.json()
  }
})
```

In der ersten Zeile wird die HTTP-Methode definiert, in diesem Falle «POST», da Daten an die Datenbank geschickt werden. Die vom User eingegeben Daten sind in der Variablen «credentials» gespeichert. Danach wird in der DB nach dem eingegebenen Benutzernamen gesucht. Das Passwort kann nicht direkt verglichen werden, da dieses gehasht in der DB gespeichert ist. Mit der Funktion «bcrypt.compare» wird das eingegebene Passwort mit dem gehashten Passwort in der DB verglichen. Ist dieses identisch wird ein «True» Wert zurückgegeben.

In dem «if-Statement» wird der zuvor erwähnte Boolean-Wert genommen. Ist dieser «True», wird ein Token gesetzt und der User hat sich erfolgreich eingeloggt.

19.6.3 API event.js

Wie bereits erwähnt, wurden die Endpoints im Backend auf zwei Files verteilt. Im File «event.js» sind alle Endpoints gespeichert, welche Terminsets hinzufügen, löschen oder bearbeiten. Im Folgenden wird der «Löschen»-Endpoint beschrieben.

```
app.delete('/:id', (req, res) => {  
  let id = req.params.id  
  knex('events').delete().where('id',id)  
  .then(result => res.json(result))  
  .catch(err => console.log(err))  
})
```

Mit «app.delete» wird eine Delete-Abfrage an die Datenbank geschickt. Der Parameter «/:id», welcher weiter unten als Variable deklariert wird, übergibt der Abfrage die jeweilige ID des zu löschenden Termins.

Auf Zeile drei sieht man die konkrete Abfrage an die Datenbank. Übersetzt heisst diese, dass der Eintrag in der Tabelle «Events», bei welchem die übergebene «id» übereinstimmt, gelöscht werden soll.

19.7 Weitere Schritte und Pläne in der Zukunft

Die Applikation ist im momentanen Zustand so einsetzbar. Jedoch ist eine abteilungsinterne Weiterentwicklung vorgesehen. Es gibt noch einige Funktionen, welche geplant sind, wie etwa eine Wechselfunktion, wenn ein Schüler fehlt oder krank ist (betroffener Schüler wird ersetzt).

Auch weitere Funktionen sind denkbar. Etwa eine Notifikations-Funktion (von Termin betroffene Personen erhalten Nachricht), oder ein mehrstufiges Login mit einer Chatfunktion etc.

In Zukunft wird die finale Version der Applikation auf dem Intranet der Abt. Informatik gehostet. So ist sie dann allen Schülern zugänglich und erfüllt ihren vollen praktischen Nutzen. Folgende Grafik beschreibt das genaue Systemumfeld in welches die Applikation in Zukunft eingebettet werden könnte.

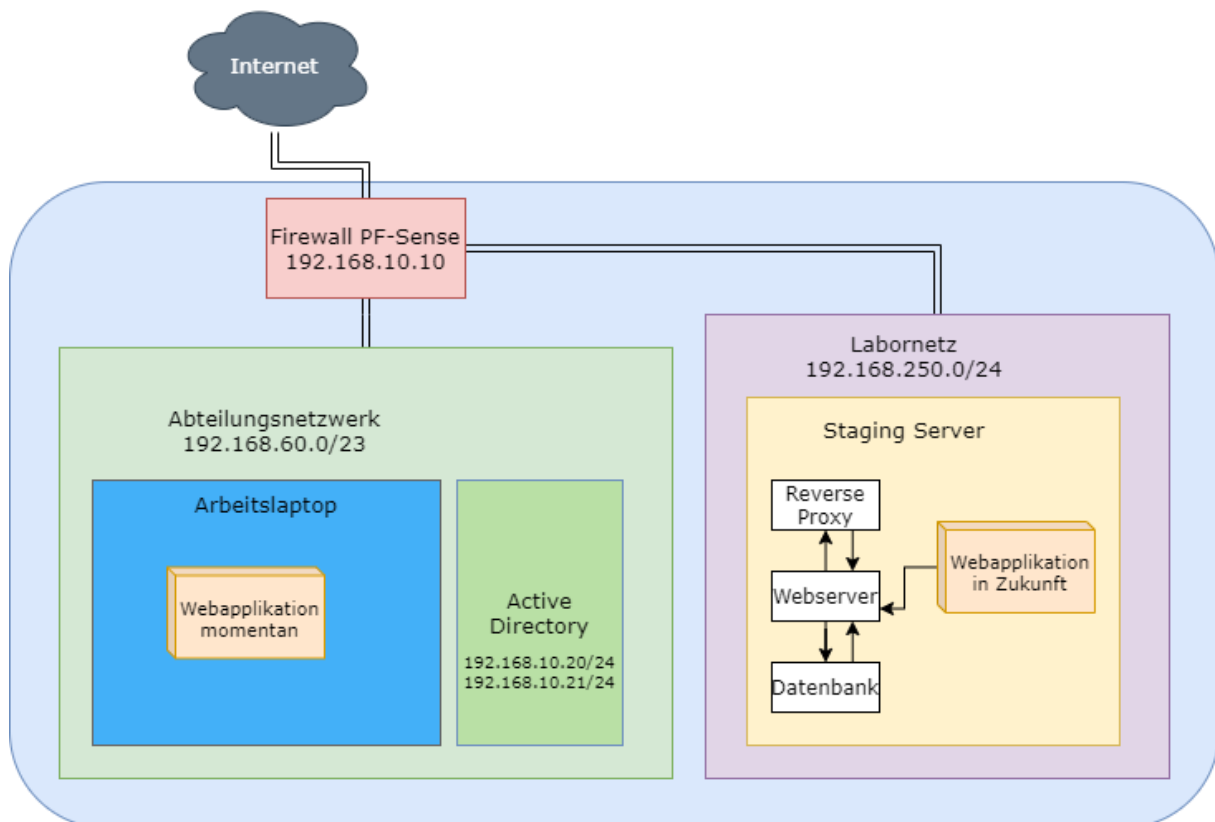


Abbildung 27: Systemumfeld Zukunft

19.8 Testfälle – Durchführung

Die im Konzept erstellten Testfälle werden durchgeführt.

Durchführungszeitpunkt: 17.03.2021, 17:00-17:30

Testperson	Unterschrift
Kavindu Jasin Pathiranage	

Tabelle 57: Unterschrift

19.8.1 Test 1

Überschrift	Beschreibung				
Test-ID	T1				
Kategorie	K1 Login				
Testmethode	Blackbox				
Testfall	Der Benutzer loggt sich mit bereitgestellten Credentials ein.				
Voraussetzungen	-Der Kandidat stellt dem Tester die nötigen Login-Daten zur Verfügung -Im Moment des Tests ist kein Benutzer eingeloggt -Die Applikation ist lokal gehostet				
Testbeschreibung	Keine Eingabe	Falscher Username & Passwort	Richtiger Username & Falsches Passwort	Falscher Username & Richtiges Passwort	Richtiger Username & Passwort
Testschritte	1. Username eingeben 2. Passwort eingeben 3. Mittels «Einloggen» bestätigen				
Testinformationen		Username: «Admin1» Passwort: «123»	Username: «Admin» Passwort: «123»	Username: «Admin1» Passwort: «12345»	Username: «Admin» Passwort: «12345»
Erwartetes Resultat	Keine Reaktion in der Applikation feststellbar, Login Vorgang fehlgeschlagen				Login Vorgang erfolgreich
Tats. Resultat	Keine Reaktion	Keine Reaktion	Keine Reaktion	Keine Reaktion	Erfolgreich weitergeleitet.
Kommentar					



<div><div>Login</div><div><div>Username</div><div>Password</div></div><div>EINLOGGEN</div></div>					
Mangelklassifizierung	MO	MO	MO	MO	MO

Tabelle 58: Testfall 1

19.8.2 Test 2

Überschrift	Beschreibung	
Test-ID	T2	
Kategorie	K4 Grafischer Output	
Testmethode	Blackbox	
Testfall	In nicht-eingeloggtem Zustand ist es dem User nicht möglich ein Aufgabenset zu erstellen, zu bearbeiten oder zu löschen	
Voraussetzungen	-Im Moment des Tests ist kein Benutzer eingeloggt -Die Applikation ist lokal gehostet	
Testbeschreibung	Auf der Seite «Events» ist es dem User nur in eingeloggtem Zustand möglich über den «+» Button ein neues Terminset zu erstellen, oder über die Edit-Buttons vorhandene Termine zu bearbeiten oder zu löschen	
Testschritte	<ol style="list-style-type: none"> 1. Der User loggt sich aus 2. Der User befindet sich auf dem Pfad http://localhost:8080/Events 3. Der User navigiert in den Hauptbereich der Seite 	<ol style="list-style-type: none"> 1. Der User loggt sich ein (Testfall 1) 2. Der User befindet sich auf dem Pfad http://localhost:8080/Events 3. Der User navigiert in den Hauptbereich der Seite
Testinformationen		
Erwartetes Resultat	Der nicht-eingeloggte User ist nicht in der Lage den «+» Button und die Edit-Buttons zu sehen, und so ein Terminset zu bearbeiten, zu löschen oder hinzuzufügen.	Der eingeloggte User ist in der Lage den «+» Button und die Edit-Buttons zu sehen, und so ein Terminset hinzuzufügen, zu bearbeiten oder zu löschen.
Tats. Resultat	Der Benutzer kann keine Termine eintragen / bearbeiten. Der «+» Knopf ist nicht sichtbar.	Die Editierungsoptionen und der «+» Knopf sind vorhanden.




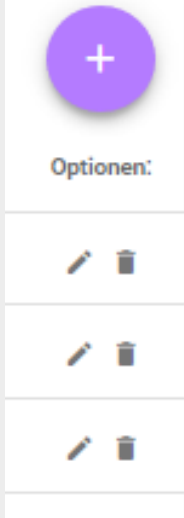
Kommentar				
Mangelklassifizierung	MO		MO	

Tabelle 59: Testfall 2

19.8.3 Test 3

Überschrift	Beschreibung																																			
Test-ID	T3																																			
Kategorie	K2 User-Input																																			
Testmethode	Blackbox																																			
Testfall	Aufgabenset erstellen																																			
Voraussetzungen	-Im Moment des Tests ist ein Benutzer eingeloggt -Die Applikation ist lokal gehostet																																			
Testbeschreibung	Auf der Seite «Events» ist es dem User in eingeloggten Zustand möglich über den «+» Button ein neues Terminset zu erstellen																																			
Testschritte	<ol style="list-style-type: none">1. Der User befindet sich auf dem Pfad http://localhost:8080/Events2. Der User navigiert zu dem rechts-mittig lokalisierten «+» Button und drückt diesen3. Im sich daraufhin öffnenden Dialog füllt der User folgende Zufallsdaten ein: <i>Name: «Test», Lehrer: «Altin Özcan», Zeitspanne: «1», Klasse: «inf18», Gruppengrösse: «2», Datum: «Montag»</i>4. Er betätigt den «Speichern» Knopf																																			
Testinformationen																																				
Erwartetes Resultat	Das erstellte Terminset erscheint in der Liste																																			
Tats. Resultat	Die Termine erscheinen wie gewünscht in der Liste. Jedoch muss die Seite aktualisiert werden (Leichter Mangel).																																			
Kommentar	<table><tr><th>Eventname:</th><th>Lernende:</th><th>Klasse:</th><th>Gruppengrösse:</th><th>Zeitspanne (Monate):</th><th>Datum (Wochentag):</th><th>Lehrkraft:</th></tr><tr><td>Test</td><td>kavindu maria</td><td>inf18</td><td>2</td><td>1</td><td>2021-03-28</td><td>özo</td></tr><tr><td>Test</td><td>klara adshay</td><td>inf18</td><td>2</td><td>1</td><td>2021-04-03</td><td>özo</td></tr><tr><td>Test</td><td>masato sandro</td><td>inf18</td><td>2</td><td>1</td><td>2021-03-22</td><td>özo</td></tr><tr><td>Test</td><td>hezekiah samis</td><td>inf18</td><td>2</td><td>1</td><td>2021-04-10</td><td>özo</td></tr></table>	Eventname:	Lernende:	Klasse:	Gruppengrösse:	Zeitspanne (Monate):	Datum (Wochentag):	Lehrkraft:	Test	kavindu maria	inf18	2	1	2021-03-28	özo	Test	klara adshay	inf18	2	1	2021-04-03	özo	Test	masato sandro	inf18	2	1	2021-03-22	özo	Test	hezekiah samis	inf18	2	1	2021-04-10	özo
Eventname:	Lernende:	Klasse:	Gruppengrösse:	Zeitspanne (Monate):	Datum (Wochentag):	Lehrkraft:																														
Test	kavindu maria	inf18	2	1	2021-03-28	özo																														
Test	klara adshay	inf18	2	1	2021-04-03	özo																														
Test	masato sandro	inf18	2	1	2021-03-22	özo																														
Test	hezekiah samis	inf18	2	1	2021-04-10	özo																														



Aufgabe editieren

Eventname:

Lehrer:

Zeitspanne:

Klassen:

Gruppengröße:

2021
Mon, Mar 22

< March 2021 >

S	M	T	W	T	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

Wählen sie bitte ein Datum aus!

SCHLIESSEN SPEICHERN

Mangelklassifizierung

M1

Tabelle 60: Testfall 3

19.8.4 Test 4

Überschrift	Beschreibung
Test-ID	T4
Kategorie	K2 User-Input
Testmethode	Blackbox
Testfall	Aufgabenset bearbeiten
Voraussetzungen	-Im Moment des Tests ist ein Benutzer eingeloggt -Die Applikation ist lokal gehostet
Testbeschreibung	Auf der Seite «Events» ist es dem User in eingeloggten Zustand möglich über die rechts angeordneten Symbole einen Termin zu bearbeiten
Testschritte	<ol style="list-style-type: none"> 1. Der User befindet sich auf dem Pfad http://localhost:8080/Events 2. Der User navigiert zu dem rechts-mittig lokalisierten Editierungs-Button (linker Button) und drückt diesen 3. Im sich daraufhin öffnenden Dialog ändert der User das Datum auf «Freitag» 4. Er betätigt den «Speichern» Knopf 5. Der Termin hat das neue Datum «Freitag» übernommen
Testinformationen	
Erwartetes Resultat	Der Termin erscheint mit den geänderten Parametern in der Liste
Tats. Resultat	Es wurde erfolgreich das Datum auf Freitag geändert.
Kommentar	<div> <div>Datum (Wochentag):</div> <div>2021-03-19</div> </div>
Mangelklassifizierung	MO

Tabelle 61: Testfall 4

19.8.5 Test 5

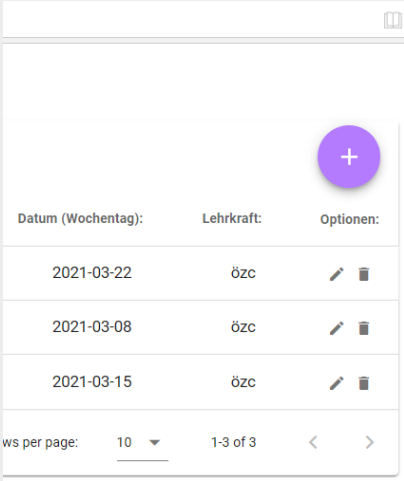
Überschrift	Beschreibung
Test-ID	T5
Kategorie	K2 User-Input
Testmethode	Blackbox
Testfall	In eingeloggtem Zustand ist es dem User möglich ein Aufgabenset zu löschen
Voraussetzungen	-Im Moment des Tests ist ein Benutzer eingeloggt -Die Applikation ist lokal gehostet
Testbeschreibung	Auf der Seite «Events» ist es dem User in eingeloggten Zustand möglich, über das rechts angeordnete Symbol einen Termin zu löschen
Testschritte	<ol style="list-style-type: none"> 1. Der User befindet sich auf dem Pfad http://localhost:8080/Events 2. Der User navigiert zu dem rechts-mittig lokalisierten Editierungs-Button (rechter Button) und drückt diesen 3. Der entsprechend zugehörige Termin wurde aus der Liste gelöscht
Testinformationen	
Erwartetes Resultat	Der entsprechende Termin wurde aus der Liste gelöscht
Tats. Resultat	Der Termin wurde erfolgreich gelöscht.
Kommentar	
Mangelklassifizierung	MO

Tabelle 62: Testfall 5

19.8.6 Test 6

Überschrift	Beschreibung
Test-ID	T6
Kategorie	K4 Grafischer Output
Testmethode	Blackbox
Testfall	Termin im Kalender anzeigen
Voraussetzungen	-Die Applikation ist lokal gehostet -Es existiert ein vorgängig erstelltes Terminset mit Testdaten
Testbeschreibung	Der Benutzer überprüft, ob ein Termin korrekt im Kalender angezeigt wird.
Testschritte	<ol style="list-style-type: none"> 1. Der User befindet sich auf dem Pfad http://localhost:8080/Events 2. Der User sucht den bei Testfall 3 erstellten Termin aus den vorhandenen Terminen aus, und merkt sich den Datumparameter 3. Der User wechselt über die Navigation oder die URL auf den Pfad http://localhost:8080/Calendar 4. Der User sucht im grafischen Kalender den zuvor gemerkten Termin
Testinformationen	
Erwartetes Resultat	Der zuvor gemerkte Termin wird an der richtigen Stelle (korrektes Datum) im Kalender angezeigt
Tats. Resultat	Die Termine sind ersichtlich

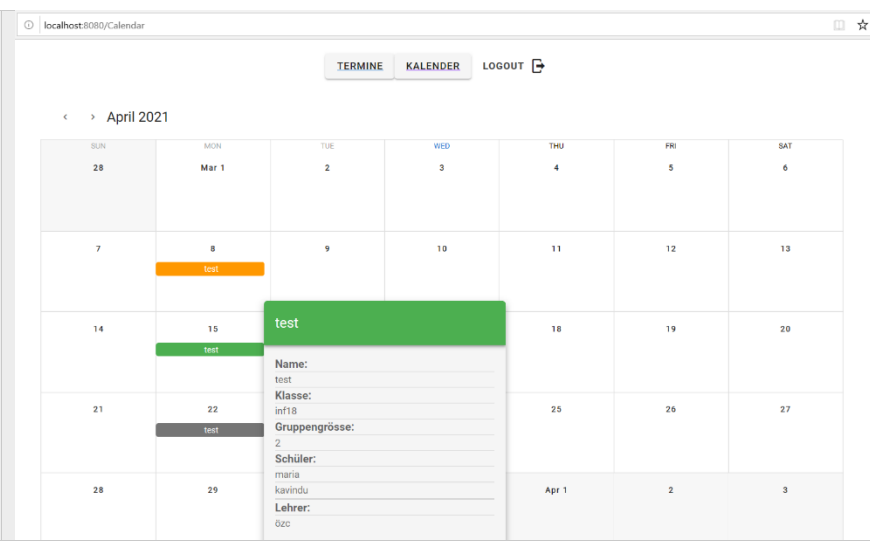
<p>Kommentar</p>	
<p>Mangelklassifizierung</p>	<p>MO</p>

Tabelle 63: Testfall 6

20 Quellenverzeichnis

Alle Quellen, welche während dieser Arbeit verwendet wurden, an einem Ort. Auf die ID wurde jeweils im Arbeitsjournal referenziert, zur Identifizierung einer Quelle.

ID	Beschreibung	Quelle
1	Alle Informationen zu der Hermes PM-Methode wurden der Weblösung des Referenzhandbuches entnommen.	https://www.hermes.admin.ch/de/projektmanagement/verstehen/ubersicht-hermes/methodenubersicht.html
2	Die Plattform, welche für sämtliche Grafiken verwendet wurde.	https://app.diagrams.net/
3	Informationen zu ERM und Datenbanken.	https://phpprog.ch/entity-relationship-modell-erm/
4	Informationsportal zu Knex.js -DB-Abfragen -Migrations & Seeds	https://devhints.io/knex
5	Sequenzdiagramme (Aufbau, if-Abfragen)	https://circle.visual-paradigm.com/selection-loops-combination/
6	SQL-Workbench Einstellungen -Row-Limitation (Default)	https://blog.sqlauthority.com/2014/01/12/mysql-change-the-limit-of-row-retrieved-in-mysql-workbench-2/#:~:text=MySQL%20Workbench%20by%20defaults%20limit,implemented%20for%20two%20major%20reasons.
7	Axios -Request Data (Daten mit ins Backend schicken)	https://stackoverflow.com/questions/51415439/how-can-i-add-raw-data-body-to-an-axios-request
8	Java Script -Array bearbeiten (Elemente kopieren)	https://www.w3schools.com/jsref/jsref_copywithin.asp
9	Dokumentationsvorlage 2021 und Coding-Conventions 2021	https://moodle.tfbern.ch/pluginfile.php/30522/mod_resource/content/1/IPA-Dokumentenvorgabe.pdf

Tabelle 64: Quellenverzeichnis

21 Glossar

Alle verwendeten Begriffe werden erklärt.

Begriff	Erklärung
Axios	Library wird für HTTP-Abfragen ins Backend benötigt
Bcrypt	Middleware zur Ver- und Entschlüsselung von Passwörtern
Blowfish-Algorithmus	Komplexer sehr sicherer Algorithmus zur Verschlüsselung von Passwörtern
Bruteforce	Bösartige Attacke mit endloser Anzahl an Requests
Dashlane	Digitaler Passwortsafe
DB-IDE	Datenbank Entwicklungsumgebung
Drive	Google-Dienst zur Cloudsicherung
Einsatzplan	Fest definiertes Set an Terminen abhängig von bestimmten Parametern
Endpoint	Datenschnittstelle zwischen Front- und Backend
Express.js	Backend Framework
Fetch	Zugriffsmethode, um Daten in Frontendkomponente zu holen
gitlab	Cloud Plattform für Applikationsentwicklung
GUI	Graphic User Interface, Synonym für Benutzeroberfläche
HERMES 5	Projektmanagementmethode
jsonwebtoken	Zur Login-Authentifizierung
knexfile	File benötigt zur Datenbankverbindung
Lifespan	Lebensdauer
Migrations	Files zur Erstellung von Tabellen und deren Attributen
Mysql	Datenbanksprache
MySQL Workbench	Datenbank Management Software mit grafischer Benutzeroberfläche
Node.js	Backend Plattform Framework
npm	Node Package Manager, Kommandozeilendienst
Reverse-Engineering	Rückwärtsentwickeln (Prozess)
Seeds	Files um Daten in eine Datenbank und deren Tabellen zu füllen
Sourcetree	Programm zu Git-Plattform
Terminsets	Mehrere Termine zu einem Set zusammengekommen
Treffpunkt Mittwoch	Interner Anlass an der Technischen Fachschule Bern
Vue.js	Frontend Framework
Vuetify.js	Frontend-Design Framework

Abbildung 28: Glossar

22 Abbildungsverzeichnis

Alle verwendeten oder erstellten Abbildungen in einem Verzeichnis.

Abbildung 1: Wiederhergestellte Datei.....	17
Abbildung 2: Projektordner	18
Abbildung 3: Arbeitsplatz.....	19
Abbildung 4: Phasenmodell.....	20
Abbildung 5: Phasenbeschreibung	20
Abbildung 6: Organigramm.....	21
Abbildung 8: Beispiel Einsatzplan.....	55
Abbildung 9: Aufbau der Applikation	56
Abbildung 10: Termin und zug. Attribute	57
Abbildung 12: ERD	64
Abbildung 13: Aufbau und Struktur.....	68
Abbildung 14: Mockup Login-Page.....	72
Abbildung 15: Mockup Termine	73
Abbildung 16: Mockup Kalender.....	74
Abbildung 17: Ordnerstruktur	86
Abbildung 18: Ordnerstruktur Backend	87
Abbildung 19: Sequenzdiagramm Login	90
Abbildung 20: Token	91
Abbildung 21: DB-Eintrag verschl. Passwort.....	93
Abbildung 22: Views und App.vue	94
Abbildung 23: Problemstellung.....	100
Abbildung 24: Save-Methode (Sequenzdiagramm))	102
Abbildung 25: Umgesetztes Design calendar.vue.....	107
Abbildung 26: Mockup calendar.vue	107
Abbildung 27: Mockup Login.vue	108
Abbildung 28: Umgesetzte Version Login.vue	108
Abbildung 29: Systemumfeld Zukunft.....	112
Abbildung 30: Glossar	124
Abbildung 31: Backupverlauf.....	128

23 Tabellenverzeichnis

Alle Tabellen dieser Dokumentation in einem Verzeichnis.

Tabelle 1: IPA-Dokumentation	1
Tabelle 2: Versionsverlauf.....	2
Tabelle 3: Standards	12
Tabelle 4: Cloud-Speicher	14
Tabelle 5: Portable Speichermedien	14
Tabelle 6: Backup und Versionsverlauf.....	15
Tabelle 7: Wiederherstellung.....	16
Tabelle 8: Ordnerstruktur	18
Tabelle 9: Arbeitsgerät.....	19
Tabelle 10: Projektroll.....	22
Tabelle 11: Schadensausmass & Eintrittswahrscheinlichkeit.....	23
Tabelle 12: Gesamtrisikoindex	23
Tabelle 13: Risikoanalyse	24
Tabelle 14: Risikomatrix	25
Tabelle 15: Legende Risikomatrix.....	25
Tabelle 16: Meilensteine.....	28
Tabelle 17: Arb. Journal Tag 1	30
Tabelle 18: Arb. Journal Tag 2.....	32
Tabelle 19: Arb. Journal Tag 3.....	34
Tabelle 20: Arb. Journal Tag 4.....	36
Tabelle 21: Arb. Journal Tag 5.....	38
Tabelle 22: Arb. Journal Tag 6.....	40
Tabelle 23: Arb. Journal Tag 7	42
Tabelle 24: Arb. Journal Tag 8.....	44
Tabelle 25: Arb. Journal Tag 9.....	46
Tabelle 26: Arb. Journal Tag 10	47
Tabelle 27: Arb. Journal Tag 11	48
Tabelle 28: Unterschriften	50
Tabelle 29: Unterschriften	51
Tabelle 30: Ziele.....	58
Tabelle 31: Projektziele	59
Tabelle 32: Funktionale Anforderungen.....	60
Tabelle 33: Nicht Funktionale Anforderungen	61
Tabelle 34: Unterschriften	62
Tabelle 35: Passwörter	66
Tabelle 36: Anwendungsfälle.....	69
Tabelle 37: Details Anwendungsfälle	70
Tabelle 38: Umsetzung Anwendungsfälle	71
Tabelle 39: Testdaten	75
Tabelle 40: Testsoftware.....	75
Tabelle 41: Mängelklassen.....	76
Tabelle 42: Testkategorien	76



Tabelle 43: Beispiel Testtabelle	77
Tabelle 44: Testfall 1	78
Tabelle 45: Testfall 2	79
Tabelle 46: Testfall 3	80
Tabelle 47: Testfall 4	81
Tabelle 48: Testfall 5	82
Tabelle 49: Testfall 6	83
Tabelle 50: Befehle zum Einrichten (Projektumgebung)	84
Tabelle 51: Befehle Packages	85
Tabelle 52: Commands - Frontend	94
Tabelle 53: Router	95
Tabelle 54: Endpoints	96
Tabelle 55: Variablen Events.vue	101
Tabelle 56: Commands - Backend	109
Tabelle 57: Unterschrift	113
Tabelle 58: Testfall 1	114
Tabelle 59: Testfall 2	116
Tabelle 60: Testfall 3	118
Tabelle 61: Testfall 4	119
Tabelle 62: Testfall 5	120
Tabelle 63: Testfall 6	122
Tabelle 64: Quellenverzeichnis	123
Tabelle 65: Sitzungsprotokoll 3.März	130
Tabelle 66: Sitzungsprotokoll 12.März	131

24 Backup Versionsverlauf

Der Versionsverlauf zeigt alle erstellten Backups und Versionen, die im Laufe dieses Projekts entstanden sind.

24.1 Gitlab












Yannis Anderegg > appointment_manager > Commits		
master	appointment_manager	Author Search by message
3 Mar, 2021 2 commits		
	Initial commit	Yannis Anderegg authored a week ago
4 Mar, 2021 1 commit		
	Initial commit	Yannis Anderegg authored a week ago
5 Mar, 2021 1 commit		
	Initial commit	Yannis Anderegg authored a weeks ago
8 Mar, 2021 1 commit		
	Initial commit	Yannis Anderegg authored a week ago
10 Mar, 2021 1 commit		
	Initial commit	Yannis Anderegg authored a week ago
11 Mar, 2021 1 commit		
	Initial commit	Yannis Anderegg authored a week ago
12 Mar, 2021 1 commit		
	Initial commit	Yannis Anderegg authored a week ago
15 Mar, 2021 1 commit		
	Initial commit	Yannis Anderegg authored a week ago
17 Mar, 2021 1 commit		
	Initial commit	Yannis Anderegg authored 3 days ago
18 Mar, 2021 1 commit		
	Initial commit	Yannis Anderegg authored 2 days ago
19 Mar, 2021 1 commit		
	Initial commit	Yannis Anderegg authored a day ago

Abbildung 29: Backupverlauf

24.2 Lokal (PC und USB-Medium)
























 IPA DOKUMENTATION_2.1_19.03.docx Autoren: Anderegg Yannis	 zeitplan_1.1_03.03.xlsx
 IPA DOKUMENTATION_2.1_19.03 - FinalVersion.docx Autoren: Anderegg Yannis	 zeitplan_1.2_04.03.xlsx
 IPA DOKUMENTATION_2.0_18.03.docx Autoren: Anderegg Yannis	 zeitplan_1.3_05.03.xlsx
 IPA DOKUMENTATION_1.9_17.03.docx Autoren: Anderegg Yannis	 zeitplan_1.4_08.03.xlsx
 IPA DOKUMENTATION_1.8_15.03.docx Autoren: Anderegg Yannis	 zeitplan_1.5_10.03.xlsx
 IPA DOKUMENTATION_1.7_12.03.docx Autoren: Anderegg Yannis	 zeitplan_1.6_11.03.xlsx
 IPA DOKUMENTATION_1.6_11.03.docx Autoren: Anderegg Yannis	 zeitplan_1.7_12.03.xlsx
 IPA DOKUMENTATION_1.5_10.03.docx Autoren: Anderegg Yannis	 zeitplan_1.8_15.03.xlsx
 IPA DOKUMENTATION_1.4_08.03.docx Autoren: Anderegg Yannis	 zeitplan_1.9_17.03.xlsx
 IPA DOKUMENTATION_1.3_05.03.docx Autoren: Anderegg Yannis	 zeitplan_2.0_18.03.xlsx
 IPA DOKUMENTATION_1.2_04.03.docx Autoren: Anderegg Yannis	 zeitplan_2.1_19.03.xlsx
 IPA DOKUMENTATION_1.1_03.03.docx Autoren: Anderegg Yannis	

Abbildung 33: Lokaler Versionsverlauf

Abbildung 32: Lokaler Versionsverlauf

25 Sitzungsprotokolle

Tag: 3. März 2021		Involvierte Personen: HEX, NEX, VF, Kandidat	
Besprochen: <ul style="list-style-type: none">-Zeitplan-Arbeitsjournal-Kriterien-Termine-Allgemeiner Ablauf-Individuelle Kriterien		Protokoll: <p>Der erste Expertenbesuch war so nicht eingeplant und deswegen relativ spontan. Trotzdem war dieses sehr aufschlussreich. Als erstes wurde die Aufgabenstellung besprochen. Danach wurden die individuellen Kriterien thematisiert. Hier wurde angemerkt das die Punkte der Indiv. Kriterien auf jeden Fall erreicht werden sollten. Danach wurde der Zeitplan besprochen. Dabei ging es vor allem um Formalitäten. Etwa welche HERMES Version welche Formatierung beinhaltet. Auch die Zeitblöcke wurden teilweise unter die Lupe genommen. Für das Arbeitsjournal und die Realisierung soll mehr Zeit eingeplant werden. Die Überschrift des Teil 1 – Formaler Teil soll in Initialisierung umbenannt werden. Danach wurden die Termine besprochen. Der zweite Expertenbesuch (virtuell, nicht vor Ort) wurden festgelegt. Der vorläufige Termin für die Präsentation und Demo wurde auf eine Woche nach Abgabe festgelegt.</p>	
Fazit: <p>Auch wenn dieser Besuch eher spontan war, konnte ich viele Informationen gewinnen, wie etwa zum Zeitplan oder zu einzelnen Themen wie der Risikoanalyse oder dem Variantenentscheid.</p>			

Tabelle 65: Sitzungsprotokoll 3.März



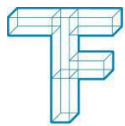
Tag: 12. März 2021		Involvierte Personen: HEX, NEX, VF, Kandidat	
Besprochen: <ul style="list-style-type: none">-Termine-Zeit in Fachgespräch, Demo usw.-ISDS-Konzept-IPA-Schutzbedarfsanalyse-Testkonzept-Hermes Website-Ist- Soll Beschreibung		Protokoll: <p>Am Anfang ging es um allgemeine Informationen und Termine. Im Anschluss haben wir den Zeitplan angeschaut. In der Doku haben wir alle Kapitel schnell überflogen. Ich hatte Fragen zu den Kapiteln ISDS, Schutzbedarfsanalyse und Testprotokoll. Durch technische Probleme kam es zu Stande das nur der Nebenexperte in dem Teams Call war. Trotzdem konnte mir der NEX zu vielen Fragen eine gute Antwort geben. Zum Beispiel hat er mir den Tipp gegeben, dass ich zur Schutzbedarfsanalyse noch Informationen auf der HERMES-Seite finden kann.</p> <p>Danach wurde kurz der Ablauf des Fachgesprächs besprochen und die Zeitparameter der Präsentation und Demo.</p> <p>Mein ERD war nur als Entwurf in der Doku, was für etwas Verwirrung bei den Experten gesorgt hat. Der IST- und SOLL Zustand wurde, was den Umfang angeht, als ausreichend erachtet.</p>	
Fazit: <p>Das zweite Gespräch war trotz der erwähnten Probleme sehr aufschlussreich. Ich bekam auf alle meine Fragen eine gute Antwort. Ich konnte ausserdem weitere Ideen für mein ISDS-Konzept sammeln. Bei den Testfällen war die Kernaussage der Fachpersonen das ich mich auf das wesentliche beschränken soll, jedoch aber alles akribisch genau beschreiben soll.</p>			

Tabelle 66: Sitzungsprotokoll 12.März

26 Anhang – Sourcecode

26.1 SQL – Script (create tables)

```
CREATE SCHEMA IF NOT EXISTS `terminverwaltung_ipa`;  
USE `terminverwaltung_ipa` ;  
  
CREATE TABLE IF NOT EXISTS `terminverwaltung_ipa`.`classes` (  
  `classes_id` INT NOT NULL AUTO_INCREMENT,  
  `class` VARCHAR(45) NOT NULL,  
  `teacher` VARCHAR(45) NOT NULL,  
  `size` INT NOT NULL,  
  PRIMARY KEY (`classes_id`))  
  
CREATE TABLE IF NOT EXISTS `terminverwaltung_ipa`.`events` (  
  `id` INT NOT NULL AUTO_INCREMENT,  
  `eventname` VARCHAR(45) NOT NULL,  
  `date` VARCHAR(45) NOT NULL,  
  `teacher` VARCHAR(45) NULL DEFAULT NULL,  
  `class` VARCHAR(45) NOT NULL,  
  `groupsize` VARCHAR(45) NOT NULL,  
  `timespan` VARCHAR(45) NOT NULL,  
  `student1` VARCHAR(45) NOT NULL,  
  `student2` VARCHAR(45) NULL DEFAULT NULL,  
  `student3` VARCHAR(45) NULL DEFAULT NULL,  
  `student4` VARCHAR(45) NULL DEFAULT NULL,  
  `fk_users` INT NULL DEFAULT NULL,  
  PRIMARY KEY (`id`),  
  UNIQUE INDEX `events_id_UNIQUE` (`id` ASC) VISIBLE,  
  UNIQUE INDEX `fk_users_UNIQUE` (`fk_users` ASC) VISIBLE)  
  
CREATE TABLE IF NOT EXISTS `terminverwaltung_ipa`.`students` (  
  `students_id` INT NOT NULL,  
  `class` VARCHAR(45) NOT NULL,  
  `name` VARCHAR(45) NOT NULL,  
  `lastname` VARCHAR(45) NOT NULL,  
  `fk_classes` INT NULL DEFAULT NULL,  
  PRIMARY KEY (`students_id`),  
  UNIQUE INDEX `students_id_UNIQUE` (`students_id` ASC) VISIBLE,  
  UNIQUE INDEX `fk_classes_UNIQUE` (`fk_classes` ASC) VISIBLE,  
  CONSTRAINT `fk_classes`  
    FOREIGN KEY (`fk_classes`)  
    REFERENCES `terminverwaltung_ipa`.`classes` (`classes_id`))  
  
CREATE TABLE IF NOT EXISTS `terminverwaltung_ipa`.`t_events_students` (  
  `fk_students` INT NOT NULL,
```



```
`fk_events` INT NOT NULL,  
PRIMARY KEY (`fk_students`, `fk_events`),  
UNIQUE INDEX `fk_students_UNIQUE` (`fk_students` ASC) VISIBLE,  
UNIQUE INDEX `fk_events_UNIQUE` (`fk_events` ASC) VISIBLE,  
//declare the foreign keys and constraints  
CONSTRAINT `fk_events`  
  FOREIGN KEY (`fk_events`)  
    REFERENCES `terminverwaltung_ipa`.`events` (`id`),  
CONSTRAINT `fk_students`  
  FOREIGN KEY (`fk_students`)  
    REFERENCES `terminverwaltung_ipa`.`students` (`students_id`))  
  
//create Table "users"  
CREATE TABLE IF NOT EXISTS `terminverwaltung_ipa`.`users` (  
  `id` INT NOT NULL AUTO_INCREMENT,  
  `username` VARCHAR(45) NOT NULL,  
  `firstname` VARCHAR(45) NOT NULL,  
  `lastname` VARCHAR(45) NOT NULL,  
  `password` VARCHAR(255) NOT NULL,  
  PRIMARY KEY (`id`),  
  UNIQUE INDEX `id_UNIQUE` (`id` ASC) VISIBLE,  
  UNIQUE INDEX `username_UNIQUE` (`username` ASC) VISIBLE,  
  UNIQUE INDEX `password_UNIQUE` (`password` ASC) VISIBLE)  
  
//all inserts of testdata in the DB  
  
INSERT INTO `terminverwaltung_ipa`.`classes` VALUES (1,'inf18','hes',16),(2,'inf19','özc',14),(3,'inf20','hem',19);  
INSERT INTO `terminverwaltung_ipa`.`students` (`students_id`, `class`, `name`, `lastname`)  
VALUES ('1', 'INF18','yannis','anderegg'),  
(2,'INF18','kavindu','pathiranage'),  
(3,'INF18','usman','hashmi'),(4,'INF18','erik','mosegaart'),  
(5,'INF18','noah','ferrari'),(6,'INF18','niroj','sivananthan'),  
(7,'INF18','adshay','puspanathan'),(8,'INF18','masato','mumenthaler'),  
(18,'INF18','hezekiah','johnson'),(19,'INF18','samis','moser'),  
(20,'INF18','liam','benedetti'),(21,'INF18','sandro','volery'),  
(23,'INF18','klara','duricova'),(24,'INF18','vinujhan','vivilanathan'),  
(25,'INF18','andrin','schrantz'),(26,'INF18','maria','makic'),(27,'INF19','haris','alilovsky'),  
(28,'INF19','ermias','bohn'),(29,'INF19','janic','deuble'),(30,'INF19','jonas','dietrich'),  
(31,'INF19','vinzent','fankhauser'),(32,'INF19','david','gloor'),(33,'INF19','keith','hager'),  
(34,'INF19','alec','jovicic'),(35,'INF19','michael','schmid'),(36,'INF19','levin','sigrist'),  
(37,'INF19','elio','thalmann'),(38,'INF19','emmanuel','walter'),(39,'INF19','yogarajah'),
```



```
(40,'INF19','agon','zeka'),(41,'INF20','ziyam','alphonsus'),(42,'INF20','efe',  
'avci'),(43,'INF20','nao','davesne'),  
(44,'INF20','cenk','eren'),(45,'INF20','sven','hayoz'),(46,'INF20','louis','ho  
lzer'),(47,'INF20','samuel','kabbani'),  
(48,'INF20','nicholas','krebs'),(49,'INF20','dhruv','kumar'),(50,'INF20','anht  
an','le'),(51,'INF20','diego','leuenberger'),  
(52,'INF20','hoang','meyer'),(53,'INF20','riki','mischon'),(54,'INF20','leon',  
'neiger'),(55,'INF20','danilo','pizurica'),  
(56,'INF20','detjon','ramadani'),(57,'INF20','linus','schaub'),(58,'INF20','ma  
el','tobler'),(59,'INF20','mathis','tomio');
```

26.2 Backend (api/event.js)

```
/* Author: Y.Anderegg
Date: 17.03.2021
Topic: Eventmanager IPA 2021 */

//All the necessary imports (KNEXfile for DB-CONNECTION)
const CONNECTION = require('../db/KNEXfile').development
const KNEX = require('KNEX')(CONNECTION)
const EXPRESS = require('EXPRESS')
const APP = EXPRESS.Router()

//Base-URL Endpoint, returns all Records from "Event" (Table)
APP.get('/', (req, res) => {
  KNEX('events').then(result => {res.json(result)})
  .catch(err => console.log(err))
})

//Returns a specific Record from "Events"
APP.get('/:id', (req, res) => {
  let id = req.params.id
  KNEX('events').where('id',id)
  .then(result => res.json(result))
  .catch(err => console.log(err))
})

//Inserts Record into "Events"
APP.post('/', (req, res) => {
  let record = req.body
  KNEX('events').insert(record)
  .then(result => res.json(result))
  .catch(err => console.log(err))
  console.log (record)
})

//Updates Record in "Events"
APP.put('/:id', (req, res) => {
  let record = req.body
  let id = req.params.id
  KNEX('events').update(record).where('id',id)
  .then(result => res.json(result))
  .catch(err => console.log(err))
})

//Deletes certain Record from "Events"
APP.delete('/:id', (req, res) => {
  let id = req.params.id
  KNEX('events').delete().where('id',id)
```

```
.then(result => res.json(result))  
.catch(err => console.log(err))  
})
```

```
module.exports = APP
```

26.3 Backend (db/migrations/)

```
/* Author: Y.Anderegg  
Date: 17.03.2021  
Topic: Eventmanager IPA 2021 */  
  
// Creates Table with the Attributes  
exports.up = function(knex) {  
  return knex.schema.createTable('users', (table) => {  
    table.increments('id')  
    table.string('username').nullable().unique()  
    table.string('firstname').nullable()  
    table.string('lastname').nullable()  
    table.string('password').nullable()  
  })  
};  
  
//If Table already exists  
exports.down = function(knex) {  
  return knex.schema.dropTableIfExists('users')  
};  
  
/*Important Commands:  
knex migrate:rollback  
knex migrate:latest  
*/
```


26.4 Backend (db/seeds/users.js)

```
/* Author: Y.Anderegg
Date: 17.03.2021
Topic: Eventmanager IPA 2021 */

const BCRYPT = require('bcryptjs')
let defaultPassword = '12345'
let hash = BCRYPT.hashSync(defaultPassword)

exports.seed = function (knex) {
  // Deletes existing data
  return knex('users').del()
    .then(function () {
      // Inserts new Data
      return knex('users').insert([
        { username: 'admin', firstname: 'John', lastname: 'Doe', password: BCR
YPT.hashSync('12345') },
      ]);
    });
};

/*Important Commands:
seed:make [options] <name>
knex seed:run [options]
*/
```

26.5 Backend (knexfile.js)

```
/* Author: Y.Anderegg
Date: 17.03.2021
Topic: Eventmanager IPA 2021 */

//Enables Connection to Database
module.exports = {
  development: {
    client: 'mysql',
    connection: {
      database: 'terminverwaltung_ipa',
      user: 'root',
      password: '12345'
    },
  },
};
```

26.6 Backend (index.js)

```
/* Author: Y.Anderegg
Date: 17.03.2021
Topic: Eventmanager IPA 2021 */

//ALL the necessary imPORTs
const EXPRESS = require('EXPRESS');
const BCRYPT = require('BCRYPTjs')
const APP = EXPRESS();
//Short-Term to use in the Endpoints ("APP")
APP.use(EXPRESS.json())

//Backend Environment PORT Number
const PORT = 80
//Development Mode (still in development)
const ENVIRONMENT = process.env.NODE_ENV || 'development';
const CONFIG = require('./db/KNEXfile.js').development;
const KNEX = require('KNEX')(CONFIG)
const JWT = require('jsonwebtoken')

//Serve the frontend
APP.use(EXPRESS.static('../frontend/dist'));

//Secret hash for Token
const PEPPER = "secret";

//Serve the api (/api/event.js)
APP.use('/api/*', EXPRESS.json())
APP.use('/api/event', require('./api/event'))

//Send Login Credentials to Database and compare them with Records from Table "
users"
APP.post('/api/user/login', async (req,res)=> {
  let credentials = req.body
  //console.log(credentials);
  let answer = await KNEX('users')
    .where('username', credentials.username)
  let user = answer[0]
  //Returns Boolean if the compared string matches
  let test = BCRYPT.compareSync(credentials.password, user.password)
  if (test) {
    //if mentioned Boolean is true then send Token to the Frontend
    let payload = {username:user.Username, role:user.role}
    let token = JWT.sign(payload, PEPPER)
    //Sends the Token back to the Frontend
    res.json({
      token: token
    })
  }
})
```



```
    })
  }
  //If the password does not match, send empty object
  else {
    res.json()
  }
})

//Get all the Records from "students" where name matches
APP.get('/api/inf18', async (req, res) => {
  //Asynchronous Function, selects data with "where" Parameter
  let person = await KNEX
    .from('students')
    .select('name')
    .where('class', 'inf18')
  //send response to frontend
  res.send(person)
})

//Get all the Records from "students" where name matches
APP.get('/api/inf19', async (req, res) => {
  let person = await KNEX
    .from('students')
    .select('name')
    .where('class', 'inf19')
  res.send(person)
})

//Get all the Records from "students" where name matches
APP.get('/api/inf20', async (req, res) => {
  let person = await KNEX
    .from('students')
    .select('name')
    .where('class', 'inf20')
  res.send(person)
})

/*Note: this commented out Endpoint did not work properly, workaround is Endpo
int "get_class"*/
/*  APP.get('/api/test', async (req, res) => {
  let info = req.body
  let person = await KNEX
    .from('students')
    .join('contacts', {'users.id': 'students.id'})
    .select('name')
    .whereIn('id', [1, 2, 3])
    .where('class', 'inf20')
  res.send(person)
}) */
```



```
//Get all the Classes with their Teachers from Table "Classes"  
APP.get('/api/get_class', async (req, res) => {  
  let classes = await KSEX  
  .from('classes')  
  .select('class')  
  .select('teacher')  
  res.send(classes)  
})  
  
//Backend listens to PORT which was declared in the Header of the File  
APP.listen(PORT, () => console.log(`APP listening on PORT: ${PORT}!`));
```



26.7 Frontend (src/router/index.js)

```
/* Author: Y.Anderegg
Date: 17.03.2021
Topic: Eventmanager IPA 2021 */

import Vue from 'vue'
import VueRouter from 'vue-router'
import Events from '../views/Events.vue'
import Calendar from '../views/Calendar.vue'
import Login from '../views/Login.vue'
//ALL imports of the Pages and Dependencies

//ROUTER-Links for Navigation trough the sites
Vue.use(VueRouter)
//Object contains Routes with Parameters
const routes = [
  {
    path: '/',
    name: 'Home',
    redirect: '/Login'
  },
  {
    path: '/events',
    name: 'Events',
    component: Events
  },
  {
    path: '/calendar',
    name: 'Calendar',
    component: Calendar
  },
  {
    path: '/login',
    name: 'Login',
    component: Login
  },
]

//ROUTER object
const ROUTER = new VueRouter({
  mode: 'history',
  base: process.env.BASE_URL,
  routes
})

export default ROUTER
```

26.8 Frontend (src/views/calendar.vue)

```
/* Author: Y.Anderegg
Date: 17.03.2021
Topic: Eventmanager IPA 2021 */

<template>
  <v-row class="fill-height">
    <v-col>
      <v-sheet height="64">
        <v-toolbar
          flat
        >
          <!-- Toolbar for the switches to next/previous Month -->
          <v-btn
            fab
            text
            small
            color="grey darken-2"
            @click="prev"
          >
            <!-- Icon to move one Month back -->
            <v-icon small>
              mdi-chevron-left
            </v-icon>
          </v-btn>
          <v-btn
            fab
            text
            small
            color="grey darken-2"
            @click="next"
          >
            <!-- Icon to move one Month forward -->
            <v-icon small>
              mdi-chevron-right
            </v-icon>
          </v-btn>
          <v-toolbar-title v-if="$refs.calendar">
            {{ $refs.calendar.title }}
          </v-toolbar-title>
          <v-spacer></v-spacer>
        </v-toolbar>
      </v-sheet>
      <v-sheet height="600">
        <!-- The calendar Obejct is made with "v-calendar" -->
        <!-- It gets the Events with their Attributes from array "events" -->
        <v-calendar
```



```
    ref="calendar"
    v-model="firstFocus"
    color="primary"
    :events="events"
    :event-color="getEventColor"
    :type="type"
    @click:event="showEvent"
    @click:more="viewDay"
    @click:date="viewDay"
  </v-calendar>
  <!-- The calendar Obejct is made with "v-calendar" -->
  <!-- It gets the Events with their Attributes from array "events" -->
  <!-- The second Calendar for the second Month -->
  <v-calendar
    ref="calendar"
    v-model="secondFocus"
    color="primary"
    :events="events"
    :event-color="getEventColor"
    :type="type"
    @click:event="showEvent"
    @click:more="viewDay"
    @click:date="viewDay"
  ></v-calendar>

  <!-- If a event is selected, a popup menu opens -->
  <v-menu
    v-model="selectedOpen"
    :close-on-content-click="false"
    :activator="selectedElement"
    offset-x
  >
    <v-card
      color="grey lighten-4"
      min-width="350px"
      flat
    >
      <!-- Every Attribute of the selected Event is getting displayed --
    >
      <v-toolbar
        :color="selectedEvent.color"
        dark
      >
        <v-toolbar-title v-html="selectedEvent.name"></v-toolbar-title>
        <v-spacer></v-spacer>
      </v-toolbar>
      <v-card-text>
```




```
<v-list-item-title class="font-weight-bold">Name:</v-list-item-  
title>  
    <v-divider></v-divider>  
    <span v-html="selectedEvent.name"></span>  
    <v-divider></v-divider>  
    <v-list-item-title class="font-weight-bold">Klasse:</v-list-  
item-title>  
    <v-divider></v-divider>  
    <span v-html="selectedEvent.class"></span>  
    <v-divider></v-divider>  
    <v-list-item-title class="font-weight-bold">Gruppengrösse:</v-  
list-item-title>  
    <v-divider></v-divider>  
    <span v-html="selectedEvent.groupsize"></span>  
    <v-divider></v-divider>  
    <v-list-item-title class="font-weight-bold">Schüler:</v-list-  
item-title>  
    <v-divider></v-divider>  
    <span v-html="selectedEvent.student1"></span>  
    <v-divider></v-divider>  
    <span v-html="selectedEvent.student2"></span>  
    <v-divider></v-divider>  
    <span v-html="selectedEvent.student3"></span>  
    <v-divider></v-divider>  
    <span v-html="selectedEvent.student4"></span>  
    <v-divider></v-divider>  
    <v-list-item-title class="font-weight-bold">Lehrer:</v-list-  
item-title>  
    <v-divider></v-divider>  
    <span v-html="selectedEvent.teacher"></span>  
</v-card-text>  
  
    <!-- To close the Menu of a Event -->  
    <v-card-actions>  
        <v-btn  
            text  
            color="secondary"  
            @click="selectedOpen = false">  
            Fertig  
        </v-btn>  
    </v-card-actions>  
</v-card>  
</v-menu>  
</v-sheet>  
</v-col>  
</v-row>  
</template>
```



```
<script>
  export default {
    /* Variables are getting declared, "focus" is for the startdate of a calendar */
    /* "Events" is a Array with all the Events displayed in the calendars */
    /* "Colors" are static and are later random mixed by a method */
    data: () => ({
      firstFocus: new Date(),
      secondFocus: new Date(),
      type: 'month',
      selectedEvent: {},
      selectedElement: null,
      selectedOpen: false,
      events: [],
      colors: ['blue', 'indigo', 'deep-purple', 'cyan', 'green', 'orange', 'grey darken-1'],
    }),

    //Is recalled at every reload of the site
    mounted () {
      this.$refs.calendar.checkChange()
      //Method "initialize" gets called to display all Events
      this.initialize()
      //For the second calendar a Date is declared (One month after the first calendar)
      let oneMonthLater = new Date(Date.now());
      oneMonthLater.setMonth(oneMonthLater.getMonth() + 1);
      this.secondFocus = oneMonthLater;
    },

    //All the Methods used in the View
    methods: {

      /* Over Fetch all Records from the Table "Events" are Loaded in
      (Note: "CODING CONVENTION" needs to be that long because of the Object Parameters) */
      initialize () {
        fetch('api/event')
          .then(response => response.json())
          .then(data => {
            data.forEach(element => {
              let event = { //for each record a new Object-Array Object is generated
                name : element.eventname,
                start : element.date.substr(0, 10),
                //As mentioned a random color is getting selected
                color: this.colors[this.rnd(0, this.colors.length - 1)],
                timed: false,

```



```
        class: element.class,  
        groupsize: element.groupsize,  
        student1: element.student1,  
        student2: element.student2,  
        student3: element.student3,  
        student4: element.student4,  
        teacher: element.teacher}  
        //The created Object-  
Array is pushed into "events" to display them in the calendars  
        this.events.push(event));  
    });  
},  
  
//Calendar Methods  
//Sets the Focus on the actual day  
viewDay ({ date }) {  
    this.focus = date  
    this.type = 'day'  
},  
//Returns the Color for an Event (random)  
getEventColor (event) {  
    return event.color  
},  
//The current view in the calendar should be empty (date gets generated  
elsewhere)  
setToday () {  
    this.focus = ''  
},  
//Method for the Button to switch to the previous Month  
prev () {  
    //Two random Dates are generated  
    let firstStart = new Date(this.firstFocus);  
    let secondStart = new Date(this.secondFocus);  
    //Subtracts one Month from each Date  
    firstStart.setMonth(firstStart.getMonth() - 1);  
    secondStart.setMonth(secondStart.getMonth() - 1);  
    //The new dates are now in the "Focus" Variable for the Calendar  
    this.firstFocus = firstStart;  
    this.secondFocus = secondStart;  
},  
//The same Method as "prev()" but this time one Month gets added to both  
dates  
next () {  
    let firstStart = new Date(this.firstFocus);  
    let secondStart = new Date(this.secondFocus);  
  
    firstStart.setMonth(firstStart.getMonth() + 1);
```



```
secondStart.setMonth(secondStart.getMonth() + 1);

this.firstFocus = firstStart;
this.secondFocus = secondStart;
},
//To show information if an event is clicked
showEvent ({ nativeEvent, event }) {
  //if an event is clicked
  const open = () => {
    this.selectedEvent = event
    this.selectedElement = nativeEvent.target
    setTimeout(() => {
      this.selectedOpen = true
    }, 10)
  }
  //if an event gets closed
  if (this.selectedOpen) {
    this.selectedOpen = false
    setTimeout(open, 10)
  } else {
    open()
  }
  nativeEvent.stopPropagation()
},
//Returns random Number for the Color mentioned before
rnd (a, b) {
  return Math.floor((b - a + 1) * Math.random()) + a
},
},
}
</script>
```

26.9 Frontend (src/views/events.vue)

```
/* Author: Y.Anderegg
Date: 17.03.2021
Topic: Eventmanager IPA 2021 */

<template>
  <v-data-table :headers="headers" :items="projects" sort-
by="project" class="elevation-1">
    <!-- data-table creates a table filled with values from Array :headers -->
    <template v-slot:top>
      <v-toolbar flat color="white">
        <v-spacer></v-spacer>
        <v-dialog v-model="dialog" max-width="900px">
          <template v-slot:activator="{ on, attrs }">
            <v-fab-transition>
              <v-btn color="#b47bff" v-bind="attrs" v-on="on" fab v-
if="button" dark>
                <!-- with "v-
if Parameter" the component is only visible if user is logged in -->
                <v-icon>mdi-plus</v-icon>
              </v-btn>
            </v-fab-transition>
          </template>
        <v-card>
          <v-card-title>
            <span class="headline">{{ formTitle }}</span>
          </v-card-title>
          <v-card-text>
            <v-container>
              <v-row>
                <!-- the following code-
blocks are input fields on the dialog -->
                <v-col cols="12" sm="6" md="6">
                  <v-text-field v-
model="editedItem.eventname" label="Eventname:"></v-text-field>
                <v-container fluid>
                  <v-row align="center">
                    <v-select :items="teachers" v-
model="editedItem.teacher" label="Lehrer:"></v-select>
                  </v-row>
                </v-container>

                <v-container fluid>
                  <v-row align="center">
                    <v-select
                      :items="timespans"
                      v-model="editedItem.timespan"

```



```
        label="Zeitspanne:"
    </v-select>
</v-row>
</v-container>

<!-- select class -->
<v-container fluid>
    <v-row align="center">
        <!--
- Every time the input changes "get_students" - Method is called -->
        <v-select
            @change="get_students"
            :items="classes"
            item-text="class"
            v-model="editedItem.class"
            label="Klassen"
        ></v-select>
    </v-row>
</v-container>

<v-container fluid>
    <v-row align="center">
        <v-select
            :items="groupsize"
            v-model="editedItem.groupsize"
            label="Gruppengrösse"
        ></v-select>
    </v-row>
</v-container>

<!-- NOTE: for development purposes the following v-
container can be helpful to read Data from DB -->
<!--
<v-container fluid>
    <v-select
        v-model="editedItem.student1"
        :items="students"
        label="Schüler"
        multiple
    >
        <template v-slot:prepend-item>
            <v-divider class="mt-2"></v-divider>
        </template>
        <template v-slot:append-item>
            <v-divider class="mb-2"></v-divider>
        </template>
    </v-select>
</v-container> -->
```

```

        </v-col>
        <v-col cols="12" sm="6" md="6">
          <v-row justify="center">
            <!--
- the datepicker returns a date from a small calendar -->
            <v-date-picker v-
model="editedItem.date">Wählen sie bitte ein Datum aus!</v-date-picker>
          </v-row>
        </v-col>
      </v-row>
    </v-container>
  </v-card-text>

  <v-card-actions>
    <v-spacer></v-spacer>
    <!-- quit-
buttons in the open dialog, if clicked they call the named Method -->
    <v-btn color="#0026ffc4" text @click="close">Schliessen</v-btn>
    <v-btn color="#0026ffc4" text @click="save">Speichern</v-btn>
  </v-card-actions>
</v-card>
</v-dialog>
</v-toolbar>
</template>
<template v-slot:[`item.actions`]="{ item }">
  <!-- edit-buttons on the right side of the data table -->
  <v-icon small class="mr-2" @click="editItem(item)" v-if="button">mdi-
pencil</v-icon>
  <v-icon small @click="deleteItem(item)" v-if="button">mdi-delete</v-
icon>
</template>
<template v-slot:no-data>
  <!--
- if no data is in Database a Button element appears that opens a help dialog
if clicked (Method "noContent")-->
  <v-
btn color="#b47bfff" @click="noContent">Noch keine Termine vorhanden!</v-btn>
</template>
</v-data-table>
</template>

<script>
//import dependencie "axios"
import axios from "axios";
export default {
  name: "Events",
  data: () => ({

```



```
//empty arrays are later filled with data from DB
students: [],
classes: [],
timespans: ["1", "3", "6"],
teachers: [],
groupsize: ["1", "2", "3", "4"],
dialog: false,
//array-object to display data in data-table
headers: [
  { text: "Eventname:", value: "eventname" },
  { text: "Lernende:", value: "student1" },
  { text: "", value: "student2" },
  { text: "", value: "student3" },
  { text: "", value: "student4" },
  { text: "Klasse:", value: "class" },
  { text: "Gruppengrösse:", value: "groupsize" },
  { text: "Zeitspanne (Monate):", value: "timespan" },
  { text: "Datum (Wochentag):", value: "date" },
  { text: "Lehrkraft:", value: "teacher" },
  { text: "Optionen:", value: "actions", sortable: false },
],
projects: [],
//default set on -1
editedIndex: -1,
//editedItem is the data-object generated by the user
editedItem: {
  eventname: "",
  groupsize: "",
},
//empty defaultItem
defaultItem: {
  eventname: "",
},
button: true,
//for the date-
picker in the frontend a date is returned (shortened to ten characters)
picker: new Date().toISOString().substr(0, 10),
//used for generating events
actualIndex: 0,
}),
computed: {
  //calculating the dialog state (dependent on clicked edit-button)
  formTitle() {
    return this.editedIndex === -1
      ? "Aufgabe hinzufügen"
      : "Aufgabe editieren";
  },
},
},
```




```
watch: {
  //closes the open dialog
  dialog(val) {
    val || this.close();
  },
},
//recalls "initialize" Method everytime
created() {
  this.initialize();
},
//mounted is computed on every reload of the site
mounted() {
  //calls Method to get Classes from DB to Frontend
  this.get_class();
  //for the appearance of the buttons the token is checked (if there is a token = button is visible)
  if (localStorage.token !== undefined) {
    console.log("Token vorhanden");
    this.button = true;
  } else {
    //else the Method is called to set the Button on false (not visible)
    this.setButton();
  }
},
//methods
methods: {
  //hide the Buttons in the frontend components
  setButton: function () {
    this.button = false;
  },

  //fetches all events saved in the DB and saves them in Local Array "this.projects"
  initialize() {
    fetch("api/event")
      .then((response) => response.json())
      .then((data) => {
        //console.log(data)
        this.projects = data;
      });
  },
  //if no content available, then a button calls a helping dialog
  noContent() {
    confirm('Erfassen sie eine Aufgabe mit "+"');
  },

  //method to change an event
  editItem(item) {
```



```
this.editedIndex = this.projects.indexOf(item);
//the new created user Object gets assigned to "item"
this.editedItem = Object.assign({}, item);
this.dialog = true;
},

//method to delete an event
deleteItem(item) {
  const index = this.projects.indexOf(item);
  //deletes it locally -> is still saved in DB!
  this.projects.splice(index, 1);
  //event with matching id gets deleted and then logged to console
  fetch(`/api/event/${item.id}`, {
    method: "DELETE",
  })
    .then((response) => response.json())
    .then((data) => {
      console.log(data);
    });
},

//method to close the dialog (user clicks "close")
close() {
  this.dialog = false;
  this.$nextTick(() => {
    this.editedItem = Object.assign({}, this.defaultItem);
    this.editedIndex = -1;
  });
},

// get all classes for dropdown list in frontend component
async get_class() {
  let response = await axios.get("/api/get_class");
  console.log(response.data);
  this.classes = response.data;
  //pushes the teacher for each class in the "this.teachers" Array (gets d
  isplayed in Frontend)
  response.data.forEach((classes) => {
    this.teachers.push(classes.teacher);
  });
},

//gets all the students from the user-selected class
async get_students() {
  //get the inserted classname and a clear array to save the data
  let classname = this.editedItem.class;
  this.students = [];
  //get students for selected class from backend
  let response = await axios.get("/api/" + classname);
```



```
//push student names to this.students
response.data.forEach((person) => {
  this.students.push(person.name);
});
//console.log("this.students:" + this.students);
},

//save-Method inserts the user-generated events in the database
/* note: "codingConventions" this Method needs to be as long as it is because of the
different user inputs who need to be set and saved*/
save() {
  //if-case for existing record (index -1)
  if (this.editedIndex > -1) {
    Object.assign(this.projects[this.editedIndex], this.editedItem);
    fetch(`/api/event/${this.editedItem.id}`, {
      method: "PUT",
      headers: { "Content-Type": "application/json" },
      body: JSON.stringify(this.editedItem),
    })
      .then((response) => response.json())
      .then((data) => {
        //logs the data object
        console.log("EventData:" + data);
      });
    //closes dialog if "save" is pressed
    this.close();
  } else {
    //saving a completely new record in DB, sort "students" array randomly
    this.students = this.randomSort(this.students);
    //the user-
    edited timespan for an event is in months, gets converted in weeks (*4)
    let countWeeks = this.editedItem.timespan * 4;
    this.actualIndex = 0;
    //loops as much times as the timespan is
    for (let week = 1; week <= countWeeks; week++) {
      //setStudents Method is called to get correct amount of students (groupsize)
      this.setStudents();
      //the date gets edited each time it goes through loop (one week later)
    }

    var newdate = this.setDate();
    this.editedItem.date = newdate;
    //new event is saved to database
    fetch(`/api/event`, {
      method: "POST",
      headers: { "Content-Type": "application/json" },
      body: JSON.stringify(this.editedItem),
    })
  }
}
```



```
    }).then((response) => response.json());
  }
  //closes "save" dialog
  this.close();
  //reloads all records from db so the created records are also displaye
d
  this.initialize();
}
},
setStudents() {
  //user-set groupsize saved in a variable
  let groupsize = this.editedItem.groupsize;
  //if the groupsize is 2 then two students are assigned to an event
  if (groupsize == 2) {
    this.editedItem.student1 = this.students[this.actualIndex];
    this.setNextIndex();
    this.editedItem.student2 = this.students[this.actualIndex];
    this.setNextIndex();
  }
  //case for a groupsize of 3
  } else if (groupsize == 3) {
    this.editedItem.student1 = this.students[this.actualIndex];
    this.setNextIndex();
    this.editedItem.student2 = this.students[this.actualIndex];
    this.setNextIndex();
    this.editedItem.student3 = this.students[this.actualIndex];
    this.setNextIndex();
  }
  //case for a groupsize of 4
  } else if (groupsize == 4) {
    this.editedItem.student1 = this.students[this.actualIndex];
    this.setNextIndex();
    this.editedItem.student2 = this.students[this.actualIndex];
    this.setNextIndex();
    this.editedItem.student3 = this.students[this.actualIndex];
    this.setNextIndex();
    this.editedItem.student4 = this.students[this.actualIndex];
    this.setNextIndex();
  }
  //case for a groupsize of 1 (last option)
  } else {
    this.editedItem.student1 = this.students[this.actualIndex];
    this.setNextIndex();
  }
},
//switches the index for assigning the students in the previous Method
setNextIndex() {
  //if Index is < -1 it add 1 and else its set to 0
  if (this.actualIndex < this.students.length - 1) {
    this.actualIndex++;
  } else {
```



```
        this.actualIndex = 0;
    }
},
//sort the "students" array
randomSort(array) {
    //the returned object
    let sourceArray = array;
    let newArray = [];
    //take one random elemt from sourceArray until empty
    while (sourceArray.length > 0) {
        //get random element and remove from source
        let randomIndex = this.getRandomInt(sourceArray.length);
        let randomElement = this.students[randomIndex];
        sourceArray.splice(randomIndex, 1);
        //push to new array
        newArray.push(randomElement);
    }
    return newArray;
},
//returns random int
getRandomInt(max) {
    return Math.floor(Math.random() * Math.floor(max));
},
//modifies the date of an event to one week later
setDate(dateObj) {
    dateObj = new Date(this.editedItem.date);
    dateObj.setDate(dateObj.getDate() + 7);
    //console.Log(dateObj);
    //shortened to substring
    dateObj = dateObj.toISOString().substr(0, 10);
    //console.Log("new Date:" + dateObj);
    //return generated object
    return dateObj;
},
},
};
</script>
```

26.10 Frontend (src/views/login.vue)

```

/* Author: Y.Anderegg
Date: 17.03.2021
Topic: Eventmanager IPA 2021 */

<template>
  <div class="login">
    <v-row align="center" justify="center">
      <v-col cols="12" sm="8" md="4">
        <!-- Login card -->
        <v-card class="elevation-12 mb-12">
          <v-toolbar color="red" dark flat>
            <!-- Login button -->
            <v-toolbar-title>Login</v-toolbar-title>
            <v-spacer />
          </v-toolbar>
          <v-card-text>
            <v-form>
              <!-- displays the login form input fields -->
              <v-text-field name="Username" label="Username" v-
model="loginData.username"></v-text-field>
              <v-text-
field name="Password" label="Password" type="password" v-
model="loginData.password"></v-text-field>
              <p style="color: red; text-align: center" v-
if="msg">{{ msg }}</p>
            </v-form>
          </v-card-text>
          <v-card-actions>
            <v-spacer />
            <!--
- Note: for the future a login with "register" Function would be possible! -->
            <!--<v-btn color="warning" @click="register">Register</v-btn>-->

            <!-- if user clicks login, the Login Method is called -->
            <v-btn color="grey" @click="login()">Einloggen</v-btn>
          </v-card-actions>
        </v-card>
      </v-col>
    </v-row>
  </div>
</template>

<script>
//necessary import
import axios from "axios";
export default {

```



```
data() {  
  return {  
    //default empty login data  
    loginData: {  
      username: '',  
      password: '',  
    },  
    //default empty login message (possible error message)  
    msg: '',  
  };  
},  
  
methods: {  
  //method is called if a user clicks on "login"  
  login() {  
    //console.log(this.loginData)  
    //login data is sent to backend  
    axios.post('/api/user/login', this.loginData)  
      .then(resp => {  
        let data = resp.data  
        //console.log(data)  
        //if theres a token the view "events" gets pushed, else a console erro  
r is sent  
        if (data.token) {  
          localStorage.setItem('token', data.token)  
          console.log("token set!");  
          this.$router.push('/events')  
        } else {  
          console.log("error! no token");  
        }  
      })  
  }  
},  
};  
</script>
```

26.11 Frontend (src/app.vue)

```
/* Author: Y.Anderegg
Date: 17.03.2021
Topic: Eventmanager IPA 2021 */

<template>
  <v-app>
    <div id="nav">
      <!-- navbar displayed on each site -->
      <router-link to="/Events">
        <v-btn value="recent">
          Termine
        </v-btn>
      </router-link>
      <router-link to="/Calendar">
        <v-btn value="favorites">
          Kalender
        </v-btn>
      </router-link>
      <!-- if a user is logged in the logout button is displayed -->
      <v-btn v-if="isLoggedIn" href="/" @click="logout" text>
        <span class="secondary--text mr-2" prepend-icon="mdi-
folder">Logout</span>
        <v-icon v-if="isLoggedIn">mdi-logout</v-icon>
      </v-btn>
    <v-main>
      <v-container>
        <router-view @loggedIn="loggedIn" />
      </v-container>
    </v-main>
  </div>
</v-app>
</template>

<script>
export default {
  data() {
    //on default no one is logged in
    return {
      isLoggedIn: false,
    };
  },
  mounted() {
    //Token getter
    let token = localStorage.getItem("token");
    //Checks if Token is there
    if (!token) {
```




```
    this.isLoggedIn = false;
    //Sets the Login on false if there is NO Token!
    if (this.$router.history.current.name !== "login") {
        this.$router.push("/login");
    }
    //if theres a token loggedIn is set on true
} else {
    this.isLoggedIn = true;
}
},
methods: {
    //default the Variable "isLoggedIn" is Set on "false" and if the User is
    Logged in it is then set to "True"
    loggedIn() {
        this.isLoggedIn = true;
    },
    //Logout deletes Token from Localstorage
    logout() {
        console.log("Logout sucessfull!");
        localStorage.removeItem("token");
    },
},
};
</script>

<style>
    /* style elements in CSS*/
    #nav {
        margin-bottom: 20px;
        margin-top: 20px;
        text-align: center;
    }
    #nav a {
        font-weight: bold;
        color: #0026ffc4;
    }
    #nav a.router-link-exact-active {
        color: #6f00ff;
    }
</style>
```