Teil 1: IPA Dokumentation

|  |  |  |  |
| --- | --- | --- | --- |
| IPA-Daten |  | | |
| Firmenname | Technische Fachschule Bern (Abteilung Informatik) | | |
| Berufsfachschule | Technische Fachschule Bern | | |
| Experten | VEX: Fabian Hutzli  HEX: Andy Bula  NEX: Xaver Imboden-Wilfling | | |
| Verantwortliche Fachkraft | Samuel Hess | | |
| Berufsbildner | Giulio Iannattone | | |
| Fachrichtung | BET | | |
| Projektvorgehensmodell | HERMES 5 | | |
| Jahrgang und Kanton | IPA 2020, Kanton Bern | | |
| Autor | Kabilan Sivanamam | | |
| Ausgabedatum | 28.02.2020 | | |
| Status | In Arbeit | In Prüfung | Zur Nutzung genehmigt |

Tabelle 1: IPA-Daten

|  |  |
| --- | --- |
| Beteiligter Personenkreis |  |
| Projektleiter | Kabilan Sivanamam |
| Fachspezialist | Kabilan Sivanamam |
| Tester | James Chibuzor |

Tabelle 2: Beteiligter Personenkreis

# Hinweise zur Formatierung

## Allgemein

Allgemeine Formatierung des Dokumentes.

|  |  |
| --- | --- |
| Formatierung | Wert |
| Schriftart | Open Sans |
| Schriftfarbe | RGB 50, 50, 50 |
| Schriftgrösse | 11 Pt. |
| Zeilenabstand | 1,5 Zeilen |

Tabelle 3: Allgemeine Formatierung

## Tabellen

Formatierung von Schrift und Grösse der Standard-Tabellen.

|  |  |
| --- | --- |
| Formatierung | Wert |
| Farbe der Titelzeile | RGB 0, 130, 180 |
| Zeilenhöhe | Auto |
| Zeilenabstand | Auto |
| Textausrichtung | Links |
| Rahmenlinien | Alle |

Tabelle 4: Tabellen Formatierung

## Konsoleneingaben

Die folgende Darstellung soll eine Konsoleneingabe darstellen:

kabi@Kabilans-MacBook-Pro-2 ~ % npm i --save knex

Es kann sein, dass ich auf einem Windows PC / Laptop weiter arbeiten muss und daher nicht alle Konsoleneingaben gleich abbilden kann wie oben.

## Code

In der Dokumentation werden Code-Ausschnitte folgendermassen dargestellt:

var knex = require('../knex')

# Kurzbeschreibung der IPA

## Info

Dies ist ein Kurzbeschrieb meiner IPA, die sich an den Leser mit Fachwissen im Bereich der Informatik richtet. Dieser Kurzbeschrieb soll dem Leser einen besseren Überblick verschaffen und helfen den Inhalt zu verstehen.

## Ausgangssituation

Die Technische Fachschule Bern bietet mit Schnuppertagen einen Einblick in das Berufsleben von Informatikern. Um an einem Schnuppertag teilzunehmen, muss man sich zuvor angemeldet haben. Das Anmeldeverfahren ist aber sehr altmodisch und nicht attraktiv. Um sich anzumelden, muss man entweder ein Anmeldeformular ausdrucken, ausfüllen und per Post absenden oder ein Anmeldeformular im Browser ausfüllen, ausdrucken und per Post absenden. Wie man sieht, muss man in beiden Fällen ein Dokument mühsam ausdrucken und per Post absenden. Der einzige Unterschied ist, dass man bei einer Version das Formular elektronisch ausfüllen kann und bei der anderen nicht. Somit entstand das Bedürfnis eine Web-App zu erstellen, mit der man sich unkompliziert und direkt anmelden kann.

## Realisierung

Im Rahmen dieser IPA wurde eine Schnupper-App erstellt, die in Form einer Webapplikation zur Verfügung steht.

Das ganzer wurde mit der Projektmethode «HERMES 5» geplant und durchgeführt.

Das genauere Vorgehen mit HERMES 5 ist in der Dokumentation beschrieben. Zudem ist in der Dokumentation ersichtlich, mit welchen Technologien die Web-App gebaut wurde.

## Ergebnis

Inhaltsverzeichnis

[1 Hinweise zur Formatierung 2](#_Toc33103911)

[1.1 Allgemein 2](#_Toc33103912)

[1.2 Tabellen 2](#_Toc33103913)

[1.3 Konsoleneingaben 3](#_Toc33103914)

[1.4 Code 3](#_Toc33103915)

[Kurzbeschreibung der IPA 4](#_Toc33103916)

[1.5 Info 4](#_Toc33103917)

[1.6 Ausgangssituation 4](#_Toc33103918)

[1.7 Realisierung 4](#_Toc33103919)

[1.8 Ergebnis 4](#_Toc33103920)

[2 Aufgabenstellung 11](#_Toc33103921)

[2.1 Titel der Arbeit 11](#_Toc33103922)

[2.2 Thematik 11](#_Toc33103923)

[2.3 Ausgangslage 11](#_Toc33103924)

[2.4 Detaillierte Aufgabenstellung 12](#_Toc33103925)

[2.5 Mittel und Methoden 12](#_Toc33103926)

[2.6 Vorkenntnisse 13](#_Toc33103927)

[2.7 Vorarbeiten 13](#_Toc33103928)

[2.8 Neue Lerninhalte 13](#_Toc33103929)

[2.9 Arbeiten in den letzten 6 Monaten 13](#_Toc33103930)

[3 Standards 14](#_Toc33103931)

[4 ISDS 15](#_Toc33103932)

[4.1 Datensicherung der IPA 15](#_Toc33103933)

[4.2 Namenskonzept der Daten 15](#_Toc33103934)

[4.3 Wiederherstellung 15](#_Toc33103935)

[4.3.1 Wiederherstellen per Github 16](#_Toc33103936)

[4.3.2 Wiederherstellung per Github Desktop 16](#_Toc33103937)

[4.4 Projektdaten 17](#_Toc33103938)

[5 Organisation der IPA-Ergebnisse 18](#_Toc33103939)

[5.1 Dokumentenablage 18](#_Toc33103940)

[5.1.1 Ordnerstruktur Tabelle 18](#_Toc33103941)

[5.2 Arbeitsplatz 19](#_Toc33103942)

[5.3 Lokaler Laptop 19](#_Toc33103943)

[6 Projektvorgehen 20](#_Toc33103944)

[6.1 Projektmethode 20](#_Toc33103945)

[6.2 Szenarien 20](#_Toc33103946)

[6.2.1 Ausgewähltes Szenario 21](#_Toc33103947)

[6.3 Einzelne Phasen 21](#_Toc33103948)

[6.4 Meilensteine 22](#_Toc33103949)

[6.5 Module 23](#_Toc33103950)

[7 IPA Projektorganisation inkl. Projektrollen 25](#_Toc33103951)

[7.1 Kontaktdaten 26](#_Toc33103952)

[7.2 Projektrollen 27](#_Toc33103953)

[8 Risikioanalyse 28](#_Toc33103954)

[8.1 Legenden 29](#_Toc33103955)

[8.1.1 Schadensausmass 29](#_Toc33103956)

[8.1.2 Eintrittswahrscheinlichkeit 29](#_Toc33103957)

[8.2 Risikograph 30](#_Toc33103958)

[8.2.1 Vor den Massnahmen 30](#_Toc33103959)

[8.2.2 Nach den Massnahmen 31](#_Toc33103960)

[9 Zeitplan 32](#_Toc33103961)

[9.1 Legenden 33](#_Toc33103962)

[9.2 Meilensteine 33](#_Toc33103963)

[10 Arbeitsjournal 34](#_Toc33103964)

[10.1 Tag 1: Mittwoch 12.02.2020 34](#_Toc33103965)

[10.2 Tag 2: Donnerstag 13.02.2020 36](#_Toc33103966)

[10.3 Tag 3: Freitag 14.02.2020 37](#_Toc33103967)

[10.4 Tag 4: Montag 17.02.2020 39](#_Toc33103968)

[10.5 Tag 5: Mittwoch 19.02.2020 41](#_Toc33103969)

[10.6 Tag 6: Donnerstag 20.02.2020 42](#_Toc33103970)

[10.7 Tag 7: Freitag 21.02.2020 43](#_Toc33103971)

[10.8 Tag 8: Montag 24.02.2020 44](#_Toc33103972)

[10.9 Tag 9: Mittwoch 26.02.2020 45](#_Toc33103973)

[10.10 Tag 10: Donnerstag 27.02.2020 46](#_Toc33103974)

[10.11 Tag 11: Freitag 28.02.2020 47](#_Toc33103975)

[11 Abschlussbericht 48](#_Toc33103976)

[11.1 Vergleich IST – SOLL 48](#_Toc33103977)

[11.2 Fazit zur IPA 48](#_Toc33103978)

[11.3 Persönliches Fazit 48](#_Toc33103979)

[11.4 Schlussreflexion 48](#_Toc33103980)

[12 Selbständigkeitserklärung und Rechtliches für Teil 1 und 2 49](#_Toc33103981)

[13 Teil 2: Projektdokumentation 50](#_Toc33103982)

[14 Einführung 51](#_Toc33103983)

[14.1 Mehrwert 51](#_Toc33103984)

[15 Initialisierung 52](#_Toc33103985)

[15.1 IST-Situation 52](#_Toc33103986)

[15.2 Abgrenzung IST 52](#_Toc33103987)

[15.3 SOLL-Situation 53](#_Toc33103988)

[15.4 Abgrenzung SOLL 53](#_Toc33103989)

[15.5 Vorgehensziele 54](#_Toc33103990)

[15.6 Anforderungen 54](#_Toc33103991)

[15.6.1 Funktionale Anforderungen 54](#_Toc33103992)

[15.6.2 Nicht funktionale Anforderungen 56](#_Toc33103993)

[15.7 Variantenvergleich 56](#_Toc33103994)

[16 Konzept 57](#_Toc33103995)

[16.1 Namenskonzept 57](#_Toc33103996)

[16.2 Datenbankmodell 58](#_Toc33103997)

[16.3 Softwarearchitektur 59](#_Toc33103998)

[16.4 Back-End-Konzept 60](#_Toc33103999)

[16.4.1 Basis des Back-End 60](#_Toc33104000)

[16.4.2 Schnittstellen für nicht angemeldete Benutzer 60](#_Toc33104001)

[16.4.3 Schnittstelle für angemeldete Benutzer 61](#_Toc33104002)

[16.5 Front-End-Konzept 62](#_Toc33104003)

[16.5.1 Startseite 62](#_Toc33104004)

[16.5.2 Datumsauswahl 63](#_Toc33104005)

[16.5.3 Personendaten 63](#_Toc33104006)

[16.5.4 Bestätigungsnachricht 64](#_Toc33104007)

[16.5.5 Login 64](#_Toc33104008)

[16.5.6 Datumsauswahl 65](#_Toc33104009)

[16.5.7 Schülerdaten 65](#_Toc33104010)

[16.6 Testkonzept 66](#_Toc33104011)

[16.6.1 Testszenario 66](#_Toc33104012)

[16.6.2 Testrahmen 66](#_Toc33104013)

[16.6.3 Testvorgehen 66](#_Toc33104014)

[16.6.4 Testdaten 67](#_Toc33104015)

[16.6.5 Testmethode 67](#_Toc33104016)

[16.6.6 Testziele 67](#_Toc33104017)

[16.6.7 Testwiederholung 67](#_Toc33104018)

[16.6.8 Fehlerklassen 68](#_Toc33104019)

[16.6.9 Testkategorien 68](#_Toc33104020)

[16.6.10 Testtabelle 69](#_Toc33104021)

[16.6.11 Testfälle 70](#_Toc33104022)

[16.7 Sicherheitskonzept 80](#_Toc33104023)

[16.7.1 Datenbanksicherheit 80](#_Toc33104024)

[16.7.2 Verschlüsselung 80](#_Toc33104025)

[17 Realisierung 81](#_Toc33104026)

[17.1 Einrichtung der Projektumgebung 81](#_Toc33104027)

[17.1.1 Front-End 81](#_Toc33104028)

[17.1.2 Back-End 83](#_Toc33104029)

[17.2 Aufsetzung der Datenbank 83](#_Toc33104030)

[17.3 Back-End-Struktur einrichten 83](#_Toc33104031)

[17.4 Back-End-Pfad / -Schnittstellen einrichten 83](#_Toc33104032)

[17.5 Front-End UI Umsetzung 83](#_Toc33104033)

[17.6 Front-End Funktionen einrichten 83](#_Toc33104034)

[17.7 Validierung der Benutzereingaben 83](#_Toc33104035)

[17.8 Back-End-Pfade / -Schnittstellen einrichten 83](#_Toc33104036)

[17.9 Code Clean-Up 83](#_Toc33104037)

[17.10 Back-End testen 83](#_Toc33104038)

[17.11 Front-End testen 83](#_Toc33104039)

[17.12 Code dokumentieren 83](#_Toc33104040)

[17.13 Dokumentation überarbeiten 84](#_Toc33104041)

[18 Einführungsteil 85](#_Toc33104042)

[18.1 Einführungskonzept 85](#_Toc33104043)

[19 Abbildungsverzeichnis 86](#_Toc33104044)

[20 Tabellenverzeichnis 86](#_Toc33104045)

[21 Literatur und Quellenverzeichnis 88](#_Toc33104046)

[22 Anhänge 88](#_Toc33104047)

[22.1 Sitzungsprotokolle 88](#_Toc33104048)

[22.2 Nachweis der Versionsverwaltung und Datensicherung 88](#_Toc33104049)

[22.3 Code / Scripts 88](#_Toc33104050)

[23 Freigaben 89](#_Toc33104051)

[24 Protokoll Expertenbesuch 90](#_Toc33104052)

[24.1 Erster Expertenbesuch 90](#_Toc33104053)

[24.2 Zweiter Expertenbesuch 91](#_Toc33104054)

# Aufgabenstellung

## Titel der Arbeit

Software für die Administration von Schnupperlehren an der TF Bern

## Thematik

Eine Schnupper-App, die das Anmeldeverfahren der Schnuppertage vereinfachen soll und den Lehrern helfen soll, die Schnuppertage besser zu organisieren und zu navigieren. Schüler füllen Anmeldeformulare online aus und Lehrer mit Accounts haben Zugriff auf die Daten.

## Ausgangslage

Für die Anmeldung zur Schnupperlehre stehen heute zwei PDF-Formulare zur Verfügung (siehe https://www.tfbern.ch/berufserkundung/berufserkundung/schnupperlehren-(1)). Letztlich muss man aber bei beiden Varianten einen Ausdruck machen. Einerseits besteht mit den zwei Formularen eine unnötige und verwirrende Redundanz, andererseits fehlt die direkte elektronische Übermittlung ohne den Umweg via Ausdruck und Post oder Scan. Die Administration der Schnupperlehren wird heute vom Sekretariat und den Lehrpersonen mit herkömmlicher Bürosoftware erledigt.

Das Bewerbungsverfahren für Schnupperlehren an der Technischen Fachschule Bern soll attraktiver und einfacher gestaltet werden. Neu sollen sich Interessierte via Web Formular für eine Schnupperlehre anmelden können. Zudem sollen Mitarbeitende bei der Durchführung der Schnuppertage durch eine Software unterstützt werden.

Die aktuelle Webseite der TF Bern wird extern auf einer ASP.NET Plattform gehostet. Es ist jedoch eine Neugestaltung des Webauftritts geplant wobei vermutlich auch die Plattform ändern wird. Daher wird die Integration dieser neuen App in die bestehende Website nicht gefordert. Vielmehr wird für diese Arbeit vorgegeben, dass die Entwicklung auf einer node.js Plattform stattfinden soll. Ein Betriebssystem wird nicht vorgegeben, weder für die Entwicklung noch für den späteren Betrieb. Das Hosting erfolgt später intern im eigenen Serverraum oder in der Cloud.

## Detaillierte Aufgabenstellung

Es ist eine datenbankgestützte Webapplikation zu erstellen, welche folgende Anforderungen erfüllt:

1. Bewerber können sich für eine Schnupperlehre anmelden (ohne vorgängige Registrierung)
2. Mitarbeiter erhalten ein persönliches Benutzerkonto.
3. Es gibt eine ansprechende und intuitive Benutzeroberfläche.
4. Die angemeldeten Kandidaten können Schnupperlehrgängen zugeordnet werden.
5. Es wird eine Teilnehmerliste angezeigt, welche bearbeitet und exportiert werden kann.
6. Es können Terminbestätigungen für die Teilnehmer per E-Mail versendet werden.
7. Am Schnuppertag kann mithilfe der App die Anwesenheitskontrolle durchgeführt werden.
8. Weitere Unterstützung bei der Durchführung (z.B. Berücksichtigung von Nachteilsausgleichen oder Essenswünschen).
9. Responsive Design: Die Applikation ist lauffähig auf verschiedenen Gerätetypen wie PC und Smartphone. Getestet wird mindestens mit Chrome unter Windows sowie Chrome unter Android und Safari unter iOS.

Die Applikation ist zu spezifizieren, zu entwerfen, zu entwickeln und zu testen. Abzugeben ist nebst dem Quellcode die gebaute Software, so dass sie für anschliessende Benutzertests auf einem Webserver intern oder extern gehostet werden kann. Der ganze Entwicklungsprozess ist zu dokumentieren.

## Mittel und Methoden

Als Projektmanagementmethode wird Hermes 5 IPA angewendet.

Die technischen Mittel und Methoden sind:

* Visual Studio Code
* JavaScript Framework vue.js mit Plugins (z.B. Vue-Router, Vuex, Vuetify etc.) inkl. Entwicklungsumgebung Webpack
* MySQL Server und Workbench
* Node.js mit Paketmanager (npm) und diversen Bibliotheken (z.B. Express, Knex, MySQL, etc.)

## Vorkenntnisse

* Website-Prototyp für Kunden der TF Bern erstellt (TF Bern)
* Klassen-Hub von inf17 erstellt (TF Bern)
* Inventar Web-App für Abteilung Informatik (TF Bern)
* Absenzen Web-App (in Progress) (TF Bern)
* Event-Webseite für ein Festival erstellt (Privat)

## Vorarbeiten

* Erfahrungen mit MySQL Server und Workbench gesammelt
* Erfahrungen mit REST APIs
* Erste Erfahrungen mit Benutzerlogins mit Web-Tokens unter node.js gemacht
* Design Framework Vuetify kennengelernt

## Neue Lerninhalte

Die erwähnten Mittel und Methoden werden vertieft.

Neu für den Kandidaten ist der serverseitige Versand von E-Mails unter node.js. Dies wurde im Unterricht bisher noch nicht behandelt und muss selbständig erarbeitet werden.

## Arbeiten in den letzten 6 Monaten

Es wurde hauptsächlich an einem Web-Hub und an einer Inventar-App gearbeitet. Dabei wurden mit folgenden Technologien gearbeitet:

- Node.js

- Vue.js und Vuetify

- Firebase und MySQL

# Standards

Die folgende Tabelle zeigt die Firmenstandards der Technischen Fachschule Bern. Die Standards werden bei der Technischen Fachschule Bern sowohl auch während der IPA verwendet.

|  |  |
| --- | --- |
| Standard | Beschreibung |
| Namenskonzept | Die Technische Fachschule Bern hat kein vorgegebenes Namenskonzept. Daher wurde ein eigenes Namenskonzept entwickelt. Das Namenskonzept ist im Abschnitt «Namenskonzept» zu finden. |
| Dokumentvorlage | Die Dokumentation wurde anhand der Dokumentationsvorgabe von PkOrg erstellt und wurde an die Standards der Technischen Fachschule Bern angepasst (Kopf-, Fusszeile und Farbschema). |
| Projektabwicklung | Die Technische Fachschule Bern hat keine vorgegebene Projektabwicklungsmethode. Jeder Projektleiter bestimmt dies selbst. |
| Sicherheitskonzept | Die Technische Fachschule Bern hat kein festes Sicherheitskonzept festgelegt. Das Sicherheitskonzept wird daher von dem Kandidaten erstellt. Das Sicherheitskonzept ist im Abschnitt «Sicherheitskonzept» ersichtlich. |
| Programm- / Script-Code | Die Technische Fachschule Bern hat keine festen Vorgaben für Programm- oder Script-Code. Die Technische Fachschule Bern verwendet deshalb die Coding Convention der ICT Berufsbildung Bern. |

Tabelle 5: Standards

# ISDS

## Datensicherung der IPA

Die Daten der IPA werden täglich am Ende des Tages auf Github im Repo «IPA2020» hochgeladen. Das Hochladen erfolgt entweder durch «Git», «Github Desktop» oder auf der Webseite von Github selbst. «Github Desktop» wird jedoch bevorzugt.

Nach dem Hochladen der Dateien, werden diese wieder heruntergeladen, um sie auch lokal auf dem Rechner zu haben. Die Version, die hochgeladen wurde, ist die «Master-Version» und weiterhin genutzt.

Nur der IPA-Kandidat hat auf das Repo Zugriff, da es durch ein Passwort geschützt ist.

## Namenskonzept der Daten

Alle Dateien, die nicht zur Software gehören, haben ein Namenskonzept.

Dokumentbezeichnung Kandidat

Beispiel: IPA\_Dokumentation\_Kabilan\_Sivanamam.docx

## Wiederherstellung

Mit Github ist es möglich, jede Version herunterzuladen, die man hochgeladen hat. Da alle Daten täglich am Abend hochgeladen, versioniert und gesichert werden, kann man sie jederzeit herunterladen. Es besteht ein maximaler Datenverlust von 8 Stunden.

### Wiederherstellen per Github

A screenshot of a computer screen

Description automatically generated

### Wiederherstellung per Github Desktop

A screenshot of a social media post

Description automatically generated

## Projektdaten

Die Projektdaten werden ebenfalls wie alle IPA-Daten auf Github hochgeladen und werden somit täglich versioniert und gesichert. Die Wiederherstellung ist identisch wie mit den IPA-Dokumenten.

# Organisation der IPA-Ergebnisse

## Dokumentenablage

Alle IPA Daten sind im «Repo-Clone» «IPA2020» abgelegt. Nur der IPA-Kandidat hat Zugriff auf die lokalen Daten.

A close up of a screen

Description automatically generated

Abbildung 1: Dokumentenablage

Die Daten für die App sind ebenfalls in diesem Ordner abgelegt.

### Ordnerstruktur Tabelle

|  |  |
| --- | --- |
| Ordner | Beschreibung |
| 01\_okumentation | Beinhaltet die «Master-Version» der Dokumentation. |
| 02\_zeitplan | Beinhaltet die aktuellste Version des Zeitplans. |
| 03\_anhang | Alle Dokumente und Daten die in den Anhang gehören (bzw. Quellcode) |
| 04\_frontend | Das gesamte Front-End wird hier entwickelt und abgelegt. |
| 05\_backend | Das gesamte Back-End wird hier entwickelt und abgelegt. |
| 06\_tests | Alle Daten, die für die Tests benötigt werden, werden hier abgelegt. |

Tabelle 6: Ordnerstruktur

## Arbeitsplatz

Während der gesamten IPA wird der Kandidat an seinem Arbeitsplatz im «Annex» arbeiten.

A desk with a computer sitting on a table

Description automatically generated

Abbildung 2: Arbeitsplatz

## Lokaler Laptop

|  |  |  |
| --- | --- | --- |
| Bild | Komponenten | Beschreibung |
| Bildergebnis für macbook pro 13 | Hersteller | Apple |
| OS | macOS Catalina |
| CPU | 2,3 GHz Dual-Core Intel Core i5 |
| RAM | 8 GB DDR3 |

Tabelle 7: Lokaler Laptop

# Projektvorgehen

## Projektmethode

Für die IPA-Arbeit wird «Hermes 5» verwendet.

Dieses Akronym steht für:

**H**andbuch der **E**lektronischen **R**echenzentren des Bundes, eine **M**ethode zur **E**ntwicklung von **S**ystemen.

HERMES ist die Projektmanagementmethode für Projekte im Bereich der Informatik, der Entwicklung von Dienstleistungen und Produkten sowie der Anpassung der Geschäftsorganisation. HERMES unterstützt die Steuerung, Führung und Ausführung von Projekten verschiedener Charakteristiken und Komplexität. HERMES hat eine klare, einfach verständliche Methodenstruktur, ist modular aufgebaut und erweiterbar. Nachfolgend sind die wesentlichen Methodenelemente und ihr Zusammenspiel beschrieben.

## Szenarien

In einer Organisation werden verschiedenartige Projekte durchgeführt. Die Projekte können sich bezüglich ihres Inhalts und der Komplexität stark unterscheiden. Um der Vielfalt der Projekte gerecht zu werden, bietet HERMES Szenarien an.

Ein Szenario ist auf die Durchführung von Projekten mit einer spezifischen Charakteristik ausgerichtet. Das Szenario beinhaltet genau diejenigen Methodenelemente von HERMES, die für das Projekt von Bedeutung sind. Dadurch ist HERMES rasch und einfach anwendbar. Die Abbildung 1 zeigt ein beispielhaftes Portfolio mit den zu den Projekten passenden Szenarien.

### Ausgewähltes Szenario

Für das Projekt ist das Szenario IT-Individualanwendung passend und wird somit ausgewählt.

## Einzelne Phasen

Abbildung 3: Phasen SmartArt

|  |  |
| --- | --- |
| Phase | Beschreibung |
| Initialisierung | Die Initialisierung verschafft eine klare Ausgangslage. |
| Konzept | Nach dem Variantenentscheid werden genaue Konzepte für das ganze Projekt erstellt. |
| Realisierung | Das erstellte Konzept wird umgesetzt und getestet. |
| Einführung | Das Projekt wird nicht in naher Zukunft eingeführt. |

Tabelle 8: Phasen

## Meilensteine

Die Meilensteine für das Projekt sehen folgendermassen aus:

|  |
| --- |
| Meilensteine |
| Freigabe für Teil 2: Initialisierung |
| Freigabe für Teil 2: Konzept |
| Freigabe für Teil 2: Realisierung |
| Abgabe auf PkOrg |

Tabelle 9: Meilensteine

## Module

|  |  |
| --- | --- |
| Modul | Beschreibung |
| Projektsteuerung | * Das Projekt initialisieren, kontinuierlich steuern und mit den übergeordneten Zielen und Vorgaben der Stammorganisation in Übereinstimmung halten. * Anliegen der Stakeholder berücksichtigen und integrieren, Risiken managen und Entscheide treffen. * Das Projekt abschliessen. |
| Projektführung | * Das Projekt planen, führen und in den definierten Rahmenbedingungen von Zeit und Kosten mit dem geforderten Ergebnis zum Ziel bringen. * Die Interessen der Stakeholder kennen, die Kommunikation führen und Entscheide sicherstellen. * Risiken managen, Probleme bewältigen und Erfahrungen berücksichtigen. * Leistungen vereinbaren und steuern, das Änderungsmanagment und die Qualitätssicherung führen. |
| Projektgrundlagen | * Die Studie erarbeiten, damit der Variantenentscheid gefällt werden kann. * Die Rechtsgrundlagen klären und den Schutzbedarf analysieren. * Die Voraussetzungen schaffen, um den Projektmanagementplan und den Projektauftrag zu erarbeiten. |
| Produkt | * Das Konzept erarbeiten und das Produkt erstellen oder beschaffen. * Realisierung und Integration von IT-Systemen werden über das Modul IT-System abgewickelt. |
| Geschäftsorganisation | * Eine Geschäftsorganisation mit Aufbau- und Ablauforganisation neu konzipieren oder verändern, realisieren und einführen. |
| Einführungsorganisation | * Organisatorische Aufgaben und Massnahmen durchführen, um den Übergang vom alten zum neuen Zustand zu unterstützen. * Umfasst das Organisations-Change-Management. * Enthält die Vorabnahme und Abnahme. |

Tabelle 10: Module

# IPA Projektorganisation inkl. Projektrollen

A screenshot of a cell phone

Description automatically generated

Abbildung 4: Projektorganisation

## Kontaktdaten

|  |  |
| --- | --- |
| Name | Daten |
| Auftraggeber | Tel.: 031 337 37 37  Email: info@tfbern.ch |
| HEX | Tel.: 079 593 83 44  Email: chefexperte@ict-berufsbildung-bern.ch |
| NEX | Tel.: 078 640 19 76  Email: xaver.imboden@greenmail.ch |
| VEX | Fabian Hutzli |
| Berufsbildner | Tel.: 031 337 38 28  Email: giulio.iannattone@tfbern.ch |
| Verantwortliche Fachkraft | Tel.: 078 949 88 55  Email: samuel.hess@tfbern.ch |
| Projektleiter | Tel.: 079 154 00 79  Email: kabilan.sivanamam@gmail.com |
| Fachspezialist |
| Tester | Tel.: 078 949 88 55  Email: samuel.hess@tfbern.ch |

Tabelle 11: Kontaktdaten

## Projektrollen

|  |  |
| --- | --- |
| Rolle | Beschreibung |
| Auftraggeber | Der Auftraggeber erteilt den Auftrag. Bei der IPA-Durchführung ist der  Auftraggeber gleichzeitig der Fachvorgesetzte und arbeitet an der  Bewertung mit. |
| HEX / NEX | Die IPA wird von zwei Experten begleitet und beurteilt. Die Experten  bewerten die Präsentation, sowie das Fachgespräch. Sie  verantworten gemeinsam die Gesamtnote. |
| VEX | Die Gesamtüberprüfung des Projektes wird durch den  Projektausschuss gemacht. Gleichzeitig stellt dieser die fachliche  Korrektheit sicher. |
| Verantwortliche Fachkraft | Die Verantwortliche Fachkraft ist für die Betreuung und Unterstützung  des IPA Kandidaten verantwortlich. Er ist ebenfalls die erste  Ansprechperson bei anfälligen Fragen und begleitet den Kandidaten  während der Durchführung der IPA. |
| Projektleiter | Die Rolle als Projektleiter übernimmt der IPA Kandidat Kabilan Sivanamam selbst. Das Projekt wird vom Projektleiter durchgeführt. Er ist das Bindeglied aller beteiligten Personen. Er ist verantwortlich dafür, dass der Zeitplan eingehalten wird und die Qualität den Anforderungen  entspricht. |
| Fachspezialist | Für die Projektumsetzung ist der Fachspezialist verantwortlich. Er  realisiert das Projekt gemäss den Anforderungen. Diese Rolle  übernimmt ebenfalls der IPA Kandidat Kabilan Sivanamam. |
| Tester | Der Tester führt die Testfälle gemäss Testkonzept durch und prüft, ob  das Produkt richtig funktioniert. |

Tabelle 12: Projektrollen

# Risikioanalyse

Mit Hilfe der folgenden Risikoanalyse werden Risiken und ihre Auswirkungen auf das Projekt gezeigt. Es wurde auch eine 2. Tabelle eingefügt, die das Schadensmass und die Eintrittswahrscheinlichkeit der Risiken zeigt, nach dem man die Präventionsmassnahmen ergriffen hat.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Nr. | Risikobeschreibung | Auswirkung | Vor Massnahme | | Massnahmen | Nach Massnahme | |
| Schadensausmass | Eintrittswahrscheinlichkeit | Schadensausmass | Eintrittswahrscheinlichkeit |
| R1 | Zeitmangel | Das Projekt kann nicht pünktlich fertiggestellt werden. | S4 | W3 | Erstellen eines Zeitplans mit genügend Reserven, welcher dann streng befolgt wird. | S3 | W2 |
| R2 | Krankheit / Unfall | Das Projekt kann nicht pünktlich fertiggestellt werden. | S4 | W2 | Der Hauptexperte wird unverzüglich informiert und bei einem Arzttermin wird ein obligatorisches Arztzeugnis entnommen. Das weitere Vorgehen wird anschliessend besprochen. | S1 | W2 |
| R3 | Datenverlust | Die aktuelle Version der Dokumentation kann nicht weiterbearbeitet werden. | S4 | W2 | Die Daten werden kontinuierlich durch ein Backup gesichert. Die Daten befinden sich auf dem Laptop des IPA Kandidaten und auf einem externen physischen Speichermedium. | S1 | W2 |
| R4 | Verbindungsabbruch des Internets | Es kann nicht mehr auf benötigte Ressourcen, welche sich im Internet befinden, zugegriffen werden. | S3 | W2 | Notfalls wird vom IPA Kandidaten eine Verbindung mit dem Mobiltelefon hergestellt, falls dies nicht möglich ist wird der HEX direkt informiert, um das weitere Vorgehen zu besprechen. | S1 | W2 |
| R5 | Systemausfall | Aufgrund eines Systemausfalls kann die IPA nicht fortgeführt bzw. nicht pünktlich abgeschlossen werden. | S4 | W2 | Bei einem auftretenden Systemausfall wird unverzüglich der IPA HEX informiert. | S1 | W2 |

Tabelle 13: Risikoanalyse

## Legenden

### Schadensausmass

|  |  |
| --- | --- |
| Abkürzung | Beschreibung |
| S1 | Keine Abwertung |
| S2 | Geringe Abwertung |
| S3 | Hohe Abwertung |
| S4 | Nichtbestehen der IPA |

Tabelle 14: Schadensausmass

### Eintrittswahrscheinlichkeit

|  |  |
| --- | --- |
| Abkürzung | Beschreibung |
| W1 | Sehr unwahrscheinlich |
| W2 | Eher unwahrscheinlich |
| W3 | Neutral |
| W4 | Eher wahrscheinlich |
| W5 | Sehr wahrscheinlich |

Tabelle 15: Eintrittswahrscheinlichkeit

## Risikograph

### Vor den Massnahmen

Der folgende Risikograph zeigt die Risiken vor dem Eingreifen der Massnahmen.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sehr wahrscheinlich |  |  |  |  |
| Eher wahrscheinlich |  |  |  |  |
| Neutral |  |  |  | R1 |
| Eher unwahrscheinlich |  |  | R4 | R2, R3, R5 |
| Sehr unwahrscheinlich |  |  |  |  |
|  | **Keine Abwertung** | **Geringe Abwertung** | **Hohe Abwertung** | **Nichtbestehen der IPA** |

Tabelle 16: Risikograph vor den Massnahmen

### Nach den Massnahmen

Der folgende Risikograph zeigt die Risiken nach dem Eingreifen der Massnahmen.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sehr wahrscheinlich |  |  |  |  |
| Eher wahrscheinlich |  |  |  |  |
| Neutral |  |  |  |  |
| Eher unwahrscheinlich | R2, R3, R4, R5 |  | R1 |  |
| Sehr unwahrscheinlich |  |  |  |  |
|  | **Keine Abwertung** | **Geringe Abwertung** | **Hohe Abwertung** | **Nichtbestehen der IPA** |

Tabelle 17: Risikograph nach den Massnahemen

# Zeitplan

A close up of a building

Description automatically generated

Abbildung 5: Zeitplan

## Legenden

|  |  |
| --- | --- |
| Bedeutung | Zeichen, Farbe, etc. |
| Soll-Zeit |  |
| Ist-Zeit |  |
| Länger als geplant |  |
| Kürzer als geplant |  |
| Berufsschule |  |
| Vormittag | VM |
| Nachmittag | NM |
| Meilenstein |  |

Tabelle 18: Zeitplanlegenden

## Meilensteine

|  |  |
| --- | --- |
| Nr. | Meilenstein |
| 1 | Phase «Initialisierung freigegeben» |
| 2 | Phase «Konzept» freigegeben |
| 3 | Phase «Realisierung» freigegeben |
| 4 | IPA hochgeladen |

Tabelle 19: Zeitplanmeilensteine

# Arbeitsjournal

## Tag 1: Mittwoch 12.02.2020

|  |  |  |  |
| --- | --- | --- | --- |
| Aufgabe | Beteiligte Personen | Geplanter Aufwand (h) | Effektiver Aufwand (h) |
| Zeitplan erstellen + Dokumentationsvorlage erstellen | Kabilan Sivanamam | 2 | 2 |
| An Teil: 1 arbeiten (Kurzfassung, Aufgabenstellung, etc.) | Kabilan Sivanamam | 3 | 3 |
| Sitzung mit verantwortlicher Fachkraft | Kabilan Sivanamam, Samuel Hess | 1 | 1 |
| Sitzung mit Experten und verantwortlicher Fachkraft | Kabilan Sivanamam, Samuel Hess, Andy Bula | 1 | 1 |
| Arbeitsjournal und Backup | Kabilan Sivanamam | 1 | 1 |
| **Tagesablauf** | | | |
| Ganz am Anfang des Tages gab es ein Problem mit der Anmeldung mit meinem Benutzerkonto (TF Bern). Dieses Problem entstand durch das ablaufen der Passwortperiode. Im weiteren Verlauf des Tages, wurde ich nur mit 2 kleinen Problemen in Word und Excel konfrontiert. Es handelte sich um Formatierungsfehler. Beim Expertengespräch hatte ich die Chance alle meine Fragen beantworten zu lassen vom Experten und diese Chance nutze ich auch. Beim Gespräch wurden Dinge wie: Abgabe, Zeitplan etc. besprochen. Es gab nichts technisches zu dokumentieren. | | | |
| **Probleme** | | **Lösungen** | |
| Problem bei der Anmeldung | | Sich im weissen Netz anmelden und Passwort ändern. | |
| Formatierungsfehler in Word | | Vorlage neu erstellen. | |
| Formatierungsfehler in Excel | | Regeln anpassen für die Berechnung der Zeit. | |
| **Reflexion** | | | |
| Was habe ich gut gemacht:   * Ich habe immer wieder kleine Anpassungen gemacht im Zeitplan für Zukünftige Aufgaben, nachdem ich ein Gespür für die Einplanung erhalten habe. * Ich habe nach Zeitplan gearbeitet und konnte die Zeiten einhalten.   Was könnte ich besser machen:   * Ich hätte das Passwort vor den Ferien ändern sollen, um solche Probleme zu vermeiden. Hätte ich es geändert, hätte ich keine Zeit verschwendet mit dem Passwort. | | | |
| **Weiteres Vorgehen** | | | |
| Nach erfolgreichem Abschluss von Teil 1 gehe ich über zu Teil 2: Initialisierung. Wenn alles nach Plan läuft, sollte Teil 2: Initialisierung auch nach einem Tag fertig sein. | | | |

Tabelle 20: Arbeitsjournal 12.02.2020

## Tag 2: Donnerstag 13.02.2020

|  |  |  |  |
| --- | --- | --- | --- |
| Aufgabe | Beteiligte Personen | Geplanter Aufwand (h) | Effektiver Aufwand (h) |
| Teil 2: Initialisierung | Kabilan Sivanamam | 6 | 6 |
| Arbeitsjournal + Backup | Kabilan Sivanamam | 1 | 1 |
| Sitzung mit verantwortlicher Fachkraft | Kabilan Sivanamam, Samuel Hess | 1 | 1 |
| **Tagesablauf** | | | |
| Am Anfang des Tages habe ich kleine Änderungen machen müssen in «Teil 1». Danach musste ich Änderungen im Zeitplan vornehmen, da meine Einträge für die Sitzungen mit meiner verantwortlichen Fachkraft nicht mit dem Kalender von meiner verantwortlichen Fachkraft übereinstimmten. Danach lief alles wie geplant. Technisches gab es bis jetzt noch nicht darum kann ich im Tagesablauf auch nichts nennen, das technisch bezogen ist. | | | |
| **Probleme** | | **Lösungen** | |
| Zukünftige Termine konnten nicht eingehalten werden. | | Termine mit verantwortlicher Fachkraft überarbeiten. | |
| **Reflexion** | | | |
| Was habe ich gut gemacht:   * Ich habe gut nach Zeitplan gearbeitet und die Zeiten konnten durch die einfachen Arbeiten eingehalten werden.   Was könnte ich besser machen:   * Mich besser mit der verantwortlichen Fachkraft absprechen. * Mehr Zeit frei lassen im Zeitplan | | | |
| **Weiteres Vorgehen** | | | |
| Die Initialisierung abschliessen und sofort an den Konzepten arbeiten! | | | |

Tabelle 21: Arbeitsjournal 13.02.2020

## Tag 3: Freitag 14.02.2020

|  |  |  |  |
| --- | --- | --- | --- |
| Aufgabe | Beteiligte Personen | Geplanter Aufwand (h) | Effektiver Aufwand (h) |
| Teil 2: Initialisierung prüfen | Kabilan Sivanamam | 1 | 1 |
| Namenskonzept + DB-Modell | Kabilan Sivanamam | 1 | 4 |
| Back-End + Front-End-Konzept | Kabilan Sivanamam | 2 | 2 |
| Backupkonzept | Kabilan Sivanamam | 2 | 0 |
| Testkonzept | Kabilan Sivanamam | 1 | 0 |
| Arbeitsjournal und Backup | Kabilan Sivanamam | 1 | 1 |
| **Tagesablauf** | | | |
| Am Anfang des Tages habe ich gemerkt, dass ich mich komplett überschätzt habe und zu wenig Zeit eingeplant habe für das DB Konzept. Die anderen Konzepte waren kein Problem. Für das Front-End musste ich noch ein Mock-Up erstellen. Das Mock-Up habe ich mit Adobe XD erstellt. Das Mock-Up lehnt an Material- und Flat-Design an. Bis auf das gibt es nichts zu berichten (auch nichts technisches.) | | | |
| **Probleme** | | **Lösungen** | |
| Keine | | Keine | |
| **Reflexion** | | | |
| Was habe ich gut gemacht:   * Ich habe trotz Überschätzung viel schaffen können und ich rechne damit, dass ich die Zeit wiedergewinnen kann in der Realisierung.   Was könnte ich besser machen:   * Nächstes Mal sollte ich die Zeit kritischer einplanen und um Rat fragen. | | | |
| **Weiteres Vorgehen** | | | |
| Am nächsten Tag die Phase Konzept abschliessen und weiterfahren mit dem Konzept Realisierung. | | | |

Tabelle 22: Arbeitsjournal 14.02.2020

## Tag 4: Montag 17.02.2020

|  |  |  |  |
| --- | --- | --- | --- |
| Aufgabe | Beteiligte Personen | Geplanter Aufwand (h) | Effektiver Aufwand (h) |
| Back-End + Front-End Konzept erstellen | Kabilan Sivanamam | 0 | 2 |
| Sitzung mit verantwortlicher Fachkraft | Kabilan Sivanamam, Samuel Hess | 1 | 1 |
| Einrichtung der Projektumgebung | Kabilan Sivanamam | 1 | 0 |
| Aufsetzung der Datenbank | Kabilan Sivanamam | 1 | 0 |
| Arbeitsjournal und Backup | Kabilan Sivanamam | 1 | 1 |
| **Tagesablauf** | | | |
| Da ich schon wusste, dass ich hinterher hänge im Zeitplan konnte ich meine Arbeit dementsprechend anpassen, um etwas aufzuholen. Laut Zeitplan sollte ich heute schon angefangen haben mit der Realisierung, aber ich habe noch kein Testkonzept erstellt. Dies ging länger als erwartet. Im Moment habe ich mir nur Gedanken darüber gemacht wie so ein Testkonzept aussieht und im Internet recherchiert, wie man eins erstellt. | | | |
| **Probleme** | | **Lösungen** | |
| Keine | | Keine | |
| **Reflexion** | | | |
| Was habe ich gut gemacht:   * Heute habe ich meine Arbeitsweise so anpassen können, um etwas aufzuholen. Wenn dies weiterhin so gut läuft, sollte ich irgendeinmal in der Realisierung wieder aufgeholt haben.   Was könnte ich besser machen:   * Normalerweise recherchiere ich zuerst bevor ich jemanden um Hilfe bete. Doch bei der IPA sollte ich in Zukunft direkt zu jemanden gehen, der sich auskennt. | | | |
| **Weiteres Vorgehen** | | | |
| * Testkonzepterstellen / fertigstellen * Weiterfahren mit der Realisierung | | | |

Tabelle 23: Arbeitsjournal 17.02.2020

## Tag 5: Mittwoch 19.02.2020

|  |  |  |  |
| --- | --- | --- | --- |
| Aufgabe | Beteiligte Personen | Geplanter Aufwand (h) | Effektiver Aufwand (h) |
| Testkonzept entwickelt | Kabilan Sivanamam | 0 | 7 |
| Back-End-Struktur einrichten | Kabilan Sivanamam | 7 | 0 |
| Arbeitsjournal und Backup | Kabilan Sivanamam | 1 | 1 |
| **Tagesablauf** | | | |
| Heute habe ich den ganzen Tag mit dem Testkonzept gearbeitet. Ich kam kurz zur Realisierung doch musste wieder abbrechen, da ich etwas ungenau war im Testkonzept. | | | |
| **Probleme** | | **Lösungen** | |
| Keine | | Keine | |
| **Reflexion** | | | |
| Was habe ich gut gemacht:   * Ich bin nicht weiter zurückgefallen und kann weiter nach Zeitplan arbeiten (Reihenfolge der Arbeit).   Was kann ich besser machen:   * Schneller arbeiten. Ich bin mir sicher, dass ich die Arbeit schneller erledigen kann, wenn ich mich nicht so oft ablenken lies. * Nicht anderen Schülern helfen, die nicht an der IPA arbeiten. Habe «kurz» einem anderen Schüler geholfen und dabei die Zeit nicht berücksichtigt. | | | |
| **Weiteres Vorgehen** | | | |
| Das Testkonzept benötigt noch ca. 1 h. Dementsprechend arbeite ich morgen weiter am Testkonzept und fahren dann wieder weiter mit der Realisierung. | | | |

Tabelle 24: Arbeitsjournal 19.02.2020

## Tag 6: Donnerstag 20.02.2020

|  |  |  |  |
| --- | --- | --- | --- |
| Aufgabe | Beteiligte Personen | Geplanter Aufwand (h) | Effektiver Aufwand (h) |
| Testkonzept erstellen | Kabilan Sivanamam | 1 | 1 |
| Aufsetzen der Projektumgebung | Kabilan Sivanamam | 1 | 2 |
| Aufsetzung der Datenbank | Kabilan Sivanamam | 1 | 4 |
| Arbeitsjournal und Backup | Kabilan Sivanamam | 1 | 1 |
| **Tagesablauf** | | | |
| Heute habe ich das Testkonzept fertiggestellt und konnte direkt mit der Realisierung Anfangen. Es gab kleine Abweichungen im Zeitplan, aber ich hole weiterhin auf. Ich wurde noch mit der Datenbankaufsetzung fertig, doch muss noch den Dokumentationsteil schreiben. Bei der Projektumgebung musste ich nur die Ordnerstruktur anlegen und die Basis schreiben. Nachher habe ich weiter gemacht mit der Aufsetzung der DB und konnte ein Schema mit den Tabellen und dessen Attributen erstellen. | | | |
| **Probleme** | | **Lösungen** | |
| Word-Formatierung (Nummerierungs- und Titel-Abstand) | | Absatz abstand manuell einstellen. | |
| Node hat nicht mehr funktioniert | | Neuste Version von Node herunterladen | |
| **Reflexion** | | | |
| Was habe ich gut gemacht:   * Ich habe schnell und präzise gearbeitet, so dass ich bald wieder nach Zeitplan arbeiten kann.   Was kann ich besser machen:   * Konzept immer im Auge behalten. Hätte ich das gemacht, wäre ich noch schneller fertig geworden und hätte mit der Back-End-/Server-Struktur anfangen können. * Mehr Aufmerksamkeit dem Zeitplan widmen. Hätte so bessere Chancen aufzuholen und das Projekt zeitgerecht abzugeben. | | | |
| **Weiteres Vorgehen** | | | |
| Datenbankaufsetzung fertigstellen (Dokumentation) und weiterfahren mit der Server-/Back-End-Struktur. | | | |

Tabelle 25: Arbeitsjournal 20.02.2020

## Tag 7: Freitag 21.02.2020

|  |  |  |  |
| --- | --- | --- | --- |
| Aufgabe | Beteiligte Personen | Geplanter Aufwand (h) | Effektiver Aufwand (h) |
| Aufsetzung der Datenbank | Kabilan Sivanamam | 1 | 1 |
| Server-Struktur einrichten | Kabilan Sivanamam | 7 | 2 |
| Server-Endpunkte / -Schnittstellen erstellen | Kabilan Sivanamam | 7 | 4 |
| Arbeitsjournal und Backup | Kabilan Sivanamam | 1 | 1 |
| **Tagesablauf** | | | |
| Am Anfang des Tages habe ich mit der Arbeit von gestern abgeschlossen, da ich am vorherigen Tag die Dokumentation nicht weiter geführt habe beim Kapitel «Aufsetzung der Datenbank». Nachdem ich mit der gestrigen Arbeit abgeschlossen habe, habe ich weiter nach Zeitplan gearbeitet. Es war vorgesehen die Server-Struktur einzurichten und nicht mehr, jedoch hat die Einrichtung der Server-Struktur nicht so lange gedauert wie erwartet. Daher konnte ich gleich mit der Erstellung der Endpunkte anfangen und diese am Ende des Tages abschliessen. Zu den Arbeiten, die ich erledigt habe, gehören folgende Dinge dazu:   * Datenbankmodell erstellen * Knex initialisieren * Knex konfigurieren * Datenbank fertigstellen (fast) * Code dokumentieren und beschreiben * Einzelne Pakete / Abhängigkeiten beschreiben * Grundgerüst für Front-End und Back-End erstellen * Server-Gerüst fertig stellen (fast) * Nodemailer fertiggestellt * Email template erstellt | | | |
| **Probleme** | | **Lösungen** | |
| Ausführen von gewissen Knex code snippets | | Keine Lösung gefunden aber ein Workaround, dass es mir erlaubt puren SQL Code auszuführen (knex raw.()). | |
| HTTP-Requests haben nicht funktioniert | | Bei einigen Request muss man das Format zu JSON ändern. | |
| Middleware konnte nicht exportiert werden | | Common JS nutzen und nicht nach ES6 Standard arbeiten. (module.export anstatt export default) | |
| **Reflexion** | | | |
| Was habe ich gut gemacht / Erfolge:   * Ich habe sehr schnell und effizient gearbeitet und hänge nur noch ein paar Stunden hinter dem Zeitplan her. * Zeitplan strikt eingehalten.   Was kann ich besser machen / Misserfolge:   * Nicht geschafft die Tabellen nur mit knex zu erstellen und zu ändern. * Habe leider ein wenig Zeit verschwendet bei den Endpunkten für die User. Ich muss nämlich nicht alle Endpunkte implementieren und habe sie trotzdem erstellt. Hätte es vermeiden können indem ich das Konzept beachtet hätte, denn ich habe dokumentiert welche Endpunkte nicht erstellt werden müssen. | | | |
| **Weiteres Vorgehen** | | | |
| Datenbank und Server-Gerüst fertigstellen und danach gleich mit dem Front-End UI beginnen. | | | |

Tabelle 26: Arbeitsjournal 21.02.2020

## Tag 8: Montag 24.02.2020

|  |  |  |  |
| --- | --- | --- | --- |
| Aufgabe | Beteiligte Personen | Geplanter Aufwand (h) | Effektiver Aufwand (h) |
|  | Kabilan Sivanamam |  |  |
| Arbeitsjournal und Backup | Kabilan Sivanamam | 1 | 1 |
| **Tagesablauf** | | | |
|  | | | |
| **Probleme** | | **Lösungen** | |
|  | |  | |
|  | |  | |
| **Reflexion** | | | |
| Was habe ich gut gemacht / Erfolge:   * X   Was kann ich besser machen / Misserfolge:   * x | | | |
| **Weiteres Vorgehen** | | | |
|  | | | |

Tabelle 27: Arbeitsjournal 24.02.2020

## Tag 9: Mittwoch 26.02.2020

|  |  |  |  |
| --- | --- | --- | --- |
| Aufgabe | Beteiligte Personen | Geplanter Aufwand (h) | Effektiver Aufwand (h) |
|  | Kabilan Sivanamam |  |  |
| Arbeitsjournal und Backup | Kabilan Sivanamam | 1 | 1 |
| **Tagesablauf** | | | |
|  | | | |
| **Probleme** | | **Lösungen** | |
|  | |  | |
|  | |  | |
| **Reflexion** | | | |
| Was habe ich gut gemacht / Erfolge:   * X   Was kann ich besser machen / Misserfolge:   * x | | | |
| **Weiteres Vorgehen** | | | |
|  | | | |

Tabelle 28: Arbeitsjournal 26.02.2020

## Tag 10: Donnerstag 27.02.2020

|  |  |  |  |
| --- | --- | --- | --- |
| Aufgabe | Beteiligte Personen | Geplanter Aufwand (h) | Effektiver Aufwand (h) |
|  | Kabilan Sivanamam |  |  |
| Arbeitsjournal und Backup | Kabilan Sivanamam | 1 | 1 |
| **Tagesablauf** | | | |
|  | | | |
| **Probleme** | | **Lösungen** | |
|  | |  | |
|  | |  | |
| **Reflexion** | | | |
| Was habe ich gut gemacht / Erfolge:   * X   Was kann ich besser machen / Misserfolge:   * x | | | |
| **Weiteres Vorgehen** | | | |
|  | | | |

Tabelle 29: Arbeitsjournal 27.02.2020

## Tag 11: Freitag 28.02.2020

|  |  |  |  |
| --- | --- | --- | --- |
| Aufgabe | Beteiligte Personen | Geplanter Aufwand (h) | Effektiver Aufwand (h) |
|  | Kabilan Sivanamam |  |  |
| Arbeitsjournal und Backup | Kabilan Sivanamam | 1 | 1 |
| **Tagesablauf** | | | |
|  | | | |
| **Probleme** | | **Lösungen** | |
|  | |  | |
|  | |  | |
| **Reflexion** | | | |
| Was habe ich gut gemacht / Erfolge:   * X   Was kann ich besser machen / Misserfolge:   * x | | | |
| **Weiteres Vorgehen** | | | |
|  | | | |

Tabelle 30: Arbeitsjournal 28.02.2020

# Abschlussbericht

Text schreiben

## Vergleich IST – SOLL

Text schreiben

## Fazit zur IPA

Text schreiben

## Persönliches Fazit

Text schreiben

## Schlussreflexion

Text schreiben

# Selbständigkeitserklärung und Rechtliches für Teil 1 und 2

Text schreiben

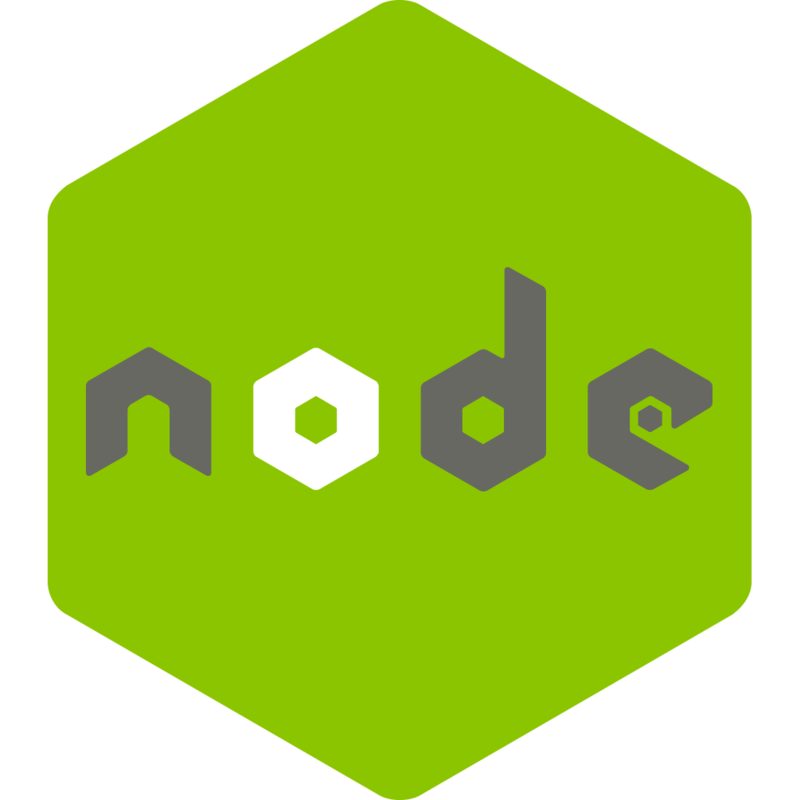
Unterschrift Lernender \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Unterschrift VF \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Teil 2: Projektdokumentation

IPA Projektname: Schnupper-App

Autor: Kabilan Sivanamam



# Einführung

Schon seit längerem besteht ein Bedarf nach einer App, die das Anmeldeverfahren für eine Schnupperlehre vereinfacht und auch bei der Umsetzung der Schnuppertage hilft. Im Moment können Schüler sich auf 2 Arten für eine Schnupperlehre anmelden.

1. Formular ausdrucken, ausfüllen, per Post absenden.
2. Formular online ausfüllen, ausdrucken, per Post senden.

Bei beiden Arten müssen die Schüler das Formular auf mühsame Art ausfüllen, ausdrucken und per Post senden.

## Mehrwert

1. Ein einfacheres Anmeldeverfahren bedeutet, dass sich mehr Schüler anmelden und somit mehr Kandidaten rekrutiert werden können für eine Lehre in der Technischen Fachschule Bern.
2. Es wird weniger Fehler geben bei der Anmeldung, was den Aufwand für die Korrektur verringert.
3. Lehrer können sich besser Vorbereiten auf Schüler. Somit kann man die Anzahl Reklamationen verringern.
4. Lehrer und Betreuer arbeiten effizienter und die Fehlerrisiko wird gesenkt (z. B. Falsches Essen bestellt)

# Initialisierung

## IST-Situation

Die IST-Situation sieht folgendermassen aus:

|  |  |
| --- | --- |
| Nr. | Situation |
| 1 | Formular muss in jedem Fall ausgedruckt werden. |
| 2 | Formular muss in jedem Fall zur Post gebracht werden. |
| 3 | Bei der elektronischen Anmeldung, die das Verfahren vereinfachen sollte, kann man sich nicht direkt, fehlerfrei und ohne Umwege anmelden. (E-Mail-Client öffnen, formales Mail schreiben, Fehler bei der Adresse) |
| 4 | Bietet keine Möglichkeit für Nachteilsausgleich (Behinderungen, Allergien, etc. können nicht genannt werden). |
| 5 | Keine Möglichkeit unter Betreuern und Lehrern Anwesenheit zu prüfen. |
| 6 | Datum für Schnupperlehren müssen umständlich geändert werden. |
| 7 | Grosser administrativer Aufwand. |

Tabelle 31: IST-Situation

## Abgrenzung IST

|  |
| --- |
| Abgrenzung |
| Das neue System muss nicht in das alte integriert werden. Daher ist keine Kompatibilität nötig. |

Tabelle 32: Abgrenzung IST

## SOLL-Situation

Die SOLL-Situation sieht folgendermassen aus:

|  |  |
| --- | --- |
| Nr. | Situation |
| 1 | Schüler sollen sich auf einem einzigen Kommunikationsweg anmelden. (Kein Durcheinander mehr) |
| 2 | Schüler können sich direkt auf der Webseite anmelden. |
| 3 | Eingaben sollen validiert werden, um Fehler zu vermeiden. |
| 4 | Schüler sollen per E-Mail benachrichtigt werden, falls ihre Anmeldung angenommen wurde. |
| 5 | Lehrer und Betreuer sollen mehr involviert sein durch die App. |
| 6 | Lehrer und Betreuer sollen schnell und einfach die Schnuppertermine steuern können. |
| 7 | Lehrer und Betreuer sollten die Anwesenheit der Schüler prüfen können. |
| 8 | Lehrer und Betreuer sollen besser vorbereitet sein, falls es bei einem Nachteilsausgleich nötig ist. |

Tabelle 33: SOLL-Situation

## Abgrenzung SOLL

|  |
| --- |
| Abgrenzung |
| Einführung der der App ist nicht Bestandteil des Projekts. Das Projekt endet mit dem Release der Software. |

Tabelle 34: Abgrenzung SOLL

## Vorgehensziele

Unten sind die Vorgehensziele aufgelistet und beschrieben:

|  |  |
| --- | --- |
| Ziel | Beschreibung |
| HERMES 5 | Das Projekt wird mit der Projektmethode HERMES 5 abgewickelt. |
| Einhalten des Zeitplans und Transparenz | Zeitplan wird streng eingehalten und transparent geführt. |
| Arbeitsjournal | Jeden Abend wird das Arbeitsjournal geführt. |
| Backup | Jeden Abend wird ein Commit gemacht von der «Master-Version» zur Versionsverwaltung und Sicherung. |
| IPA Abgabe | Die fertiggestellte Version des Produkts wird auf PkOrg hochgeladen. |

Tabelle 35: Vorgehensziele

## Anforderungen

Unten sind die Anforderungen aufgeteilt in funktionale Anforderungen und nicht funktionale Anforderungen.

### Funktionale Anforderungen

|  |  |
| --- | --- |
| Anforderung | Beschreibung |
| Senden von Formulardaten | Schülern ist es möglich ein Anmeldeformular auszufüllen und abzusenden. |
| Schülereingaben validieren | Schülerdaten sollten erst validiert werden, bevor man sie in die Datenbank speichert. |
| Login | Mitarbeiter sollten mit ihrem Benutzerkonto sich anmelden können. |
| Einsicht auf Daten der Schüler | Mitarbeiter haben Einsicht auf die Schülerdaten. |
| Bestätigungsmail senden | Mitarbeiter können den Schülern ein Bestätigungsmail senden. |
| Schülerdaten ändern / korrigieren | Die Schülerdaten, auf die man Einsicht hat, sollte man bearbeiten können. |
| Schüler löschen | Man kann Schüler aus dem System löschen. |
| Einsicht auf Schnupperdaten | Man hat Einsicht auf die Schnupperdaten wie z. B. Starttermin, Endtermin, Uhrzeit. |
| Relevanz der Schnupperdaten ändern | Mitarbeiter können kontrollieren welche Schnupperdaten auf der Webseite angezeigt werden. |
| Einsicht auf Profil | Mitarbeiter haben volle Einsicht auf ihr eigenes Profil. |
| Profil bearbeiten | Mitarbeiter können ihr Passwort ändern |
| Profil löschen | Mitarbeiter können ihr Profil löschen. |
| Logout | Eingeloggte Benutzer können sich wieder ausloggen. |

Tabelle 36: Funktionale Anforderungen

### Nicht funktionale Anforderungen

|  |  |
| --- | --- |
| Anforderungen | Beschreibung |
| Ansprechendes Design | Die Webseite soll schön gestaltet werden. |
| Lesbarer Code gemäss Coding-Convention | Der Code, der geschrieben wurde, soll verständlich sein. |

Tabelle 37: Nicht funktionale Anforderungen

## Variantenvergleich

Ein Variantenvergleich ist nicht nötig, da die zu verwendenden Technologien in der Aufgabenstellung vorgegeben sind.

# Konzept

## Namenskonzept

|  |
| --- |
| Das Namenskonzept für die Datenbanktabellen |
| Folgendes ist zu beachten:   1. camelCase 2. Mehrzahl   Beispiel 1: users  Beispiel 2: wantedLists |

Tabelle 38: Namenskonzept für Datenbanktabellen

|  |
| --- |
| Das Namenskonzept für die Datenbankattribute |
| Folgendes ist zu beachten:   1. Name der Tabelle + Attributname 2. Name der Tabelle und Attributname mit einem Under-Score trennen 3. Attributname ist kleingeschrieben   Beispiel 1: users\_id  Beispiel 2: wantedLists\_id |

Tabelle 39: Namenskonzept für Datenbankattribute

|  |
| --- |
| Das Namenskonzept für Benutzernamen |
| Folgendes ist zu beachten:   1. Abteilungskürzel + die ersten 2 Buchstaben des Nachnamens + die ersten 2 Buchstaben des Vornamens 2. Abteilungskürzel wird mit einem Bindestrich getrennt. 3. Alles klein geschrieben   Beispiel 1: Samuel Hess = inf-hesa  Beispiel 2: Max Mustermann = ele-muma |

Tabelle 40: Namenskonzept für Benutzernamen

## Datenbankmodell

Bei diesem Modell handelt es sich um eine relationale Datenbank.

A screenshot of text

Description automatically generated

Abbildung 6: Datenbankmodell

## Softwarearchitektur

A screenshot of a cell phone

Description automatically generated

Abbildung 7: Softwarearchitektur

## Back-End-Konzept

### Basis des Back-End

Folgendes wird erstellt als Basis für das Back-End:

1. Es wird eine Server- / App-Datei erstellt, welche auf einen gewissen Port hört.
2. Middleware für die Authentifikation erstellen, die später für die Routen gebraucht wird.

### Endpunkte für nicht angemeldete Benutzer

Es braucht keine Middleware und es wird eine Route erstellt mit folgenden Endpunkten:

|  |  |  |
| --- | --- | --- |
| Methode | Endpunkt | Beschreibung |
| GET | /visitors/dates | Holt sich die verfügbaren Daten. |
| POST | /visitors/applicants | Soll einen Eintrag machen in der Datenbank. |
| POST | /visitors/login | Zum Einloggen in das jeweilige Benutzerkonto. |

Tabelle 41: Spezifische Back-End-Schnittstellen für nicht angemeldete Benutzer

### Endpunkte für angemeldete Benutzer

Hier wird eine Middleware benötigt, für die Authentifikation der Benutzer und es wird eine Route erstellt mit folgenden Endpunkten:

|  |  |  |
| --- | --- | --- |
| Methode | Endpunkt | Beschreibung |
| GET | /employees/profil | Holt sich alle verfügbaren Daten. |
| PUT | /employees/profil | Soll damit Passwort ändern können. |
| DELETE | /employees/profil | Löscht das Profil / User. |
| GET | /employees/users | Holt sich alle User-Daten. |
| POST | /employees/users | Fügt einen Benutzer hinzu. |
| GET | /employees/dates | Holt sich alle verfügbaren Daten. |
| POST | /employees/dates | Kann so ein Datum hinzufügen. |
| PUT | /employees/dates | Ändert die Eigenschaften eines Datums. |
| DELETE | /employees/dates | Löscht ein Datum. |
| GET | /employees/applicants | Holt sich alle verfügbaren Schülerdaten. |
| POST | /employees/applicants | Kann so einen Schüler hinzufügen. |
| PUT | /employees/applicants | Ändert die Eigenschaften eines Users. |
| DELETE | /employees/applicants | Löscht einen Schüler. |

Tabelle 42: Spezifische Back-End-Schnittstellen für angemeldete Benutzer

## Front-End-Konzept

Unten findet man die Mock-Ups von dem Front-End zusammen mit einer kurzen Beschreibung.

### Startseite

A screenshot of a cell phone

Description automatically generated

Abbildung 8: Front-End Mock-Up Startseite

### Datumsauswahl

A screenshot of a cell phone

Description automatically generated

Abbildung 9: Front-End Mock-Up Datumsauswahl

### Personendaten

A screenshot of a cell phone

Description automatically generated

Abbildung 10: Front-End Mock-Up Personendaten

### Bestätigungsnachricht

A screenshot of a cell phone

Description automatically generated

Abbildung 11: Front-End Mock-Up Bestätigungsnachricht

### Login

A screenshot of a cell phone

Description automatically generated

Abbildung 12: Front-End Mock-Up Login

### Datumsauswahl

A screenshot of a cell phone

Description automatically generated

Abbildung 13: Front-End Mock-Up Schnuppertagsinformationen für angemeldete Benutzer

### Schülerdaten

A screenshot of a cell phone

Description automatically generated

Abbildung 14: Front-End Mock-Up Schülerdaten

## Testkonzept

In diesem Abschnitt wird das Testkonzept definiert und das Vorgehen etwas genauer geschildert. Das Testen der App soll helfen vorhandene Fehler zu finden und zu verstehen (um sie später zu beheben).

### Testszenario

Getestet wird die fertiggestellte App. Das Verfahren wird von James Chibuzor durchgeführt.

### Testrahmen

Dem Tester steht der Rechner, auf dem die App entwickelt worden ist und die Testfälle zur Verfügung. Falls die Testperson kein eigenes Smartphone besitzt, kann diese das Smartphone des Kandidaten nutzen.

Die Tests werden auf folgenden Geräten durchgeführt:

* Entwicklungsrechner
* Smartphone (iPhone 8 oder neuer)

Die Tests werden auf folgenden Browsern durchgeführt:

* Chrome
* Safari

### Testvorgehen

Die Testszenarien werden von dem Tester durchgeführt und dabei müssen Daten angegeben werden:

* Tatsächliches Resultat
* Fehlerklasse
* Kommentar

### Testdaten

Der Tester benötigt nur ein Benutzerkonto, das vom Kandidaten zur Verfügung gestellt wird. Es werden 2 Konten zur Verfügung gestellt, falls das eine Konto nicht mehr funktionieren sollte.

|  |  |
| --- | --- |
| Benutzername | Passwort |
| admin | admin |
| root | root |

Tabelle 43

### Testmethode

Die Tests werden zum grössten Teil Blackbox-Tests sein. Dabei gibt es einen Input des Benutzers und ein Output erwartetes Output des Systems. Kenntnisse zur Struktur der Web-App oder Programmierung sind nicht notwendig.

Es werden auch einige Whitebox-Tests geben, die sollen zeigen ob das System geschützt ist vor Angreifer mit Wissen über die Struktur und Programmierung der Web-App.

Wie schon oben genannt wird James Chibuzor die Tests durchführen, doch die Whitebox-Tests müssen von dem Kandidaten durchgeführt werden.

### Testziele

Wenn alle Tests durchgeführt wurden, ist die Realisierung beendet.

### Testwiederholung

Die Tests werden eventuell mehrmals durchgeführt. Die mehrmalige Durchführung der Tests findet aber erst statt, wenn bei der ersten Durchführung schwere Mängel festgestellt wurden.

### Fehlerklassen

|  |  |  |
| --- | --- | --- |
| Klassen-ID | Fehlerklasse | Beschreibung |
| M0 | Keine Mängel | Das tatsächliche Resultat stimmt mit dem erwarteten Resultat überein. |
| M1 | Kleine Mängel | Das tatsächliche Resultat stimmt bis auf kleinere Abweichungen mit dem erwarteten Resultat überein. |
| M2 | Mittlere Mängel | Das tatsächliche Resultat stimmt nur halbwegs mit dem erwarteten Resultat überein. |
| M3 | Schwere Mängel | Das tatsächliche Resultat stimmt fast bis gar nicht mit dem erwarteten Resultat überein. |

Tabelle 44

### Testkategorien

|  |  |  |
| --- | --- | --- |
| Kategorien-ID | Kategorie | Beschreibung |
| K1 | Formular | Alle Testfälle die etwas mit dem Ausfüllen des Formulars zu tun haben. |
| K2 | Login | Alle Testfälle die etwas mit dem Login zu tun haben. |
| K3 | Dashboard | Alle Testfälle die etwas mit dem Dashboard zu tun haben. |
| K4 | Profil | Alle Testfälle die mit dem Benutzerkonto und dessen Profil zu tun haben. |
| K5 | Design | Alle Testfälle die etwas mit dem Design zu tun haben (z. B. Responsive) |

Tabelle 45

### Testtabelle

|  |  |
| --- | --- |
| Testfall-ID | Für die Identifizierung der Testfälle |
| Kategorie | Für die Kategorisierung der Testfälle |
| Testmethode | Für die angewendete Testmethode |
| Testfall | Für die Beschreibung des Testfalls |
| Voraussetzung | Welche Voraussetzungen müssen getroffen werden? |
| Beschreibung | Beschreibung der Testschritte |
| Testschritte | Welche Schritte wurden durchgeführt? |
| Testinformationen | Welche Informationen wurden genutzt (z. B. Passwort, Benutzername)? |
| Erwartetes Resultat | Für das zu erwartende Resultat |
| Tatsächliches Resultat | Für das eigentliche Resultat |
| Kommentar des Testers | Kommentar des Testers |
| Fehlerklasse | Um welche Fehlerklasse handelt es sich hierbei? |

Tabelle 46

### Testfälle

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Testfall-ID | T1 | | | | |
| Kategorie | K1 | | | | |
| Testmethode | Blackbox | | | | |
| Testfall | Anmeldeformular senden | | | | |
| Voraussetzung | 1. Front-End muss laufen 2. Back-End muss laufen (API) 3. Anmeldeseite muss offen sein | | | | |
| Beschreibung | Kein Input | Ungültige E-Mail | Checkbox nicht angekreuzt | | Alle Daten korrekt eingegeben |
| Testschritte | Nichts eingeben | Alle Daten eingeben mit ungültiger E-Mail | Alle Daten korrekt eingeben ohne Checkbox | | Alle Daten korrekt eingeben mit Checkbox |
| Testinformationen | Keine | Vorname  Nachname  Alter  E-Mail  Info | Vorname  Nachname  Alter  E-Mail  Info | | Vorname  Nachname  Alter  E-Mail  Info |
| Erwartetes Resultat | Der «Senden» Knopf wird nicht angezeigt. | Validierung zeigt, dass die E-Mail nicht korrekt ist. | «Senden» Knopf wird nicht angezeigt. | | «Senden» Knopf wird angezeigt |
| Tatsächliches Resultat |  |  |  | |  |
| Kommentar des Testers |  | | | | |
| Fehlerklasse |  |  | |  | |

Tabelle 47

|  |  |
| --- | --- |
| Testfall-ID | T2 |
| Kategorie | K5 |
| Testmethode | Blackbox |
| Testfall | Responsive Design testen |
| Voraussetzung | 1. Front-End muss laufen 2. Back-End muss laufen (API) 3. Jede Seite der Web-App muss verfügbar sein |
| Beschreibung | Kein Input |
| Testschritte | Nichts eingeben bei den Personendaten |
| Testinformationen | Keine |
| Erwartetes Resultat | Alle Seiten behalten ihre Struktur bei und passen sich nur an, wenn die Mobilansicht aufgerufen wird. |
| Tatsächliches Resultat |  |
| Kommentar des Testers |  |
| Fehlerklasse |  |

Tabelle 48

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Testfall-ID | T3 | | | | | | | | |
| Kategorie | K2 | | | | | | | | |
| Testmethode | Blackbox | | | | | | | | |
| Testfall | Login testen | | | | | | | | |
| Voraussetzung | 1. Front-End muss laufen 2. Back-End muss laufen (API) 3. Login-Seite muss offen sein | | | | | | | | |
| Beschreibung | Anmelden mit falschem Benutzername | Anmelden mit falschem Passwort | | | Anmelden mit falschem Benutzername und Passwort | | Anmelden mit den richtigen Benutzerdaten | | |
| Testschritte | Falschen Benutzernamen eingeben und das richtige Passwort eingeben | Richtigen Benutzernamen angeben und falsches Passwort | | Falschen Benutzernamen und falsches Passwort eingeben | | | | Richtigen Benutzernamen und richtiges Passwort eingeben. | |
| Testinformationen | Benutzername: admin, Passwort: admin  Benutzername: root, Passwort: root  Für die falschen Anmeldedaten kann man sich etwas beliebiges ausdenken. | | | | | | | | |
| Erwartetes Resultat | Fehler bei der Anmeldung | | Fehler bei der Anmeldung | | | Fehler bei der Anmeldung | | | Weiterleitung zum Dashboard |
| Tatsächliches Resultat |  | | | | | | | | |
| Kommentar des Testers |  | | | | | | | | |
| Fehlerklasse |  | | | | | | | | |

Tabelle 49

|  |  |
| --- | --- |
| Testfall-ID | T4 |
| Kategorie | K3 |
| Testmethode | Blackbox |
| Testfall | Einsehen von Schülerdaten |
| Voraussetzung | 1. Front-End muss laufen 2. Back-End muss laufen (API) 3. Dashboard-Seite muss offen sein |
| Beschreibung | Kein Input |
| Testschritte | Zum Dashboard navigieren |
| Testinformationen | Keine |
| Erwartetes Resultat | Alle Schülerdaten sollten jetzt einsehbar sein für den angemeldeten Benutzer. |
| Tatsächliches Resultat |  |
| Kommentar des Testers |  |
| Fehlerklasse |  |

Tabelle 50

|  |  |
| --- | --- |
| Testfall-ID | T5 |
| Kategorie | K3 |
| Testmethode | Blackbox |
| Testfall | Daten einsehen und Möglichkeit deren Attribute zu ändern |
| Voraussetzung | 1. Front-End muss laufen 2. Back-End muss laufen (API) 3. User muss sich auf der Dashboard-Seite befinden |
| Beschreibung | Kein Input |
| Testschritte | Nichts eingeben bei den Personendaten |
| Testinformationen | Keine |
| Erwartetes Resultat | Alle relevanten und nicht relevanten Daten sind einsehbar und man kann deren Attribute ändern. |
| Tatsächliches Resultat |  |
| Kommentar des Testers |  |
| Fehlerklasse |  |

Tabelle 51

|  |  |
| --- | --- |
| Testfall-ID | T6 |
| Kategorie | K5 |
| Testmethode | Blackbox |
| Testfall | Profildaten einsehen |
| Voraussetzung | 1. Front-End muss laufen 2. Back-End muss laufen (API) 3. Profil-Seite muss offen sein |
| Beschreibung | Kein Input |
| Testschritte | Zum Profil navigieren. |
| Testinformationen | Keine |
| Erwartetes Resultat | Benutzer ID, Benutzername, Vorname und Nachname sind einsehbar. |
| Tatsächliches Resultat |  |
| Kommentar des Testers |  |
| Fehlerklasse |  |

Tabelle 52

|  |  |
| --- | --- |
| Testfall-ID | T7 |
| Kategorie | K5 |
| Testmethode | Blackbox |
| Testfall | Passwort ändern |
| Voraussetzung | 1. Front-End muss laufen 2. Back-End muss laufen (API) 3. Profil-Seite muss offen sein |
| Beschreibung | Kein Input |
| Testschritte | Profilseite öffnen und Knopf zum Löschen drücken. |
| Testinformationen | Keine |
| Erwartetes Resultat | Profil wird gelöscht, User wird abgemeldet und zur Login-Seite weitergeleitet. |
| Tatsächliches Resultat |  |
| Kommentar des Testers |  |
| Fehlerklasse |  |

Tabelle 53

|  |  |
| --- | --- |
| Testfall-ID | T8 |
| Kategorie | K5 |
| Testmethode | Blackbox |
| Testfall | Darf keine Seite öffnen, die eine Berechtigung benötigt. |
| Voraussetzung | 1. Front-End muss laufen 2. Back-End muss laufen (API) 3. Seite im Browser öffnen |
| Beschreibung | Probieren auf geschützte Seiten zuzugreifen |
| Testschritte | Profilseite öffnen und Knopf zum Löschen drücken. |
| Testinformationen | Keine |
| Erwartetes Resultat | User wird zur Login-Seite weitergeleitet. |
| Tatsächliches Resultat |  |
| Kommentar des Testers |  |
| Fehlerklasse |  |

Tabelle 54

|  |  |
| --- | --- |
| Testfall-ID | T9 |
| Kategorie | K5 |
| Testmethode | Blackbox |
| Testfall | Falsche Token funktionieren nicht |
| Voraussetzung | 1. Front-End muss laufen 2. Back-End muss laufen (API) |
| Beschreibung | Falschen Token in dem lokalen Speicher hinzufügen und probieren auf geschützte Seiten zuzugreifen. |
| Testschritte | Mit F12 die Entwicklungstools öffnen und in das Register «Applications». Im Register Applications einen Token erstellen mit einem beliebigen Wert. Nach dem der falsche Token eingefügt wurde, probieren auf eine geschützte Seite zuzugreifen. |
| Testinformationen | Keine |
| Erwartetes Resultat | User gelangt auf die Seite aber erhält keine Daten vom Server. |
| Tatsächliches Resultat |  |
| Kommentar des Testers |  |
| Fehlerklasse |  |

Tabelle 55

|  |  |
| --- | --- |
| Testfall-ID | T10 |
| Kategorie | K5 |
| Testmethode | Blackbox |
| Testfall | Passwort Validierung |
| Voraussetzung | 1. Front-End muss laufen 2. Back-End muss laufen (API) 3. Dashboard muss offen und verfügbar sein |
| Beschreibung | Prüfen ob das eingegebene Passwort stark genug ist bevor man einen User erstellt. |
| Testschritte | Im Dashboard zur User-Section navigieren und dort den Knopf neuer Benutzer erstellen drücken, alle Angaben eingeben mit einem Passwort, dass kürzer ist als 8 Zeichen. |
| Testinformationen | Benutzername: test, Passwort: test, Vorname: test, Nachname: test |
| Erwartetes Resultat | Button «Erstellen» ist ausgeblendet und es wird aufgefordert ein anderes Passwort einzugeben. |
| Tatsächliches Resultat |  |
| Kommentar des Testers |  |
| Fehlerklasse |  |

Tabelle 56

## Sicherheitskonzept

### Datenbanksicherheit

Es wird ein technisches Benutzerkonto erstellt, welcher alle SQL-Abfragen übernimmt. Dies ist notwendig, da sonst der «root», der alle Rechte hat, jede Abfrage machen muss.

### Verschlüsselung

Da die Schüler auf der Seite schützenswerte Daten angeben, muss die Seite verschlüsselt werden. Die Einführung der App ist jedoch nicht Bestandteil der IPA und kann somit ignoriert werden.

# Realisierung

In diesem Teil der Dokumentation wird die Realisierung dokumentiert.

## Einrichtung der Projektumgebung

### Serverplattform

Bei diesem Projekt dient Node als Serverplattform und ist zwingend herunterzuladen von der offiziellen Node Webseite (<https://nodejs.org/en/>).

Wichtig ist, dass man LTS Version herunterladet, da bei der anderen Version (Current), die Stabilität nicht garantiert werden kann.

Version: 12.16.1

### NPM

Um Entwicklungspakete zu installieren, brauchen es einen «Package manager». NPM eignet sich sehr gut als Package manager und wird mit der Installation von Node mitinstalliert.

Version: 6.13.7

### Vue CLI

Um mit Vue zu arbeiten reicht ein Vue CDN-Link. Da aber ein ganzes Projekt erstellt werden soll, wird empfohlen die Vue CLI zu installieren.

Version: 3.7.0

### MySQL Community-Server

Im Detailbeschrieb ist vorgegeben mit MySQL zu arbeiten. Daher wird der MySQL Community-Server installiert.

Version: 8.0.18

### MySQL Workbench

Um mit dem MySQL-Community-Server arbeiten zu können, wird die MySQL Workbench installiert, um auf die DB zuzugreifen und Änderungen vorzunehmen. Da vorgehsehen wird mit Knex zu arbeiten ist es auch zwingend die MySQL Workbench herunterzuladen.

Version: 8.0.18

### REST Client

Um die Endpunkte eines Servers so effizient wie möglich zu testen, braucht es einen REST Client, mit dem man Abfragen erstellen kann. Für dieses Projekt wurde «REST Client», ein Plug-In für VS Code installiert.

### Browser

Für die Entwicklung und für die Tests werden 2 verschiedene Browser genutzt. Der Hauptbrowser wird Chrome sein und Safari wird der sekundäre Browser sein.

Auf Chrome wird entwickelt sowohl auch getestet. Bei Safari hingegen, wird nur getestet, da die nötigen Entwicklungstools fehlen.

### Nodemon

Mit Nodemon kann man den Node-Server starten und diesen automatisch neu laden. Falls man am Server-Code etwas ändert und diese Änderungen speichert, wird Nodemon den Server neu laden.

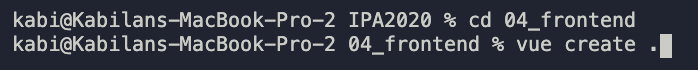
### Front-End

Um ein Vue Projekt zu initialisieren, muss man folgenden Code im Terminal ausführen:

«vue create .»

Wichtig ist, den Punkt nicht zu vergessen, wenn man schon einen Projektordner erstellt hat. Wenn man keinen Projektordner erstellt hat, kann man anstatt des Punktes den gewollten Namen eingeben für das Projekt.

Wie schon im Kapitel «Dokumentenablage» erklärt, habe ich schon einen Projektordner erstellt mit den benötigten Unterordnern. Daher kann ich in meinem Fall einfach in den gewollten Unterordner navigieren und dort das Vue-Projekt initialisieren.



Bei den Konfigurationen habe ich nicht die Standardkonfiguration ausgewählt, sondern diese selbst erstellt. Die Konfiguration sieht folgender massen aus:

A close up of a screen

Description automatically generated

Nach der Initialisierung kann man das Ganze mit dem Befehl «npm run serve» starten. Da ich aber noch Plug-Ins installieren muss, ist es sinnlos jetzt am Code weiter zu arbeiten, da durch das Installieren von einigen Plug-Ins der Code wieder gelöscht wird. Daher sollte man zuerst alle Plug-Ins installieren bevor man anfängt mit dem Code.

Nach der Vue-Initialisierung installiere ich auch gleich Vuetify. Vuetify wird folgendem Befehl installiert:



Für die Vuetify-Konfiguration habe ich «Default» ausgewählt.

Wenn man jetzt «npm run serve» wird die Seite mit Node lokal gehostet. Diese kann man mit den Daten, die die Konsole ausgibt, erreichen.

A screenshot of a cell phone

Description automatically generated

Wenn man jetzt der Adresse folgt, sollte man zur pre-build Seite gelangen, die folgendermassen aussieht:

A screenshot of a cell phone

Description automatically generated

Nun steht das Front-End und man kann jetzt damit arbeiten. Das Node-Hosting kann man mit «ctrl + c» beenden.

### Back-End

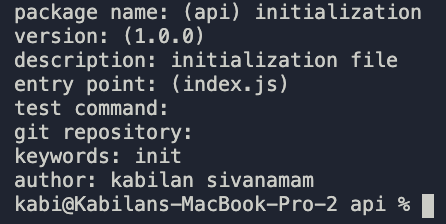
Das ganze Back-End besteht aus einer API und einer Datenbank. Dementsprechend wird das Back-End mit 2 Ordnern aufgeteilt (API und DB). Im Ordner API befinden sich die Server-Dateien und die Routen. Im Ordner DB befinden sich alle Datenbank relevanten Dateien.



Das API und die DB wird mit folgendem Befehl initialisiert:



Initialisierungskonfiguration der API:



Folgende Packages müssen noch per NPM hinzugefügt werden:

* Express
* JWT
* Knex
* Bcrypt
* Nodemailer
* MySQL

Das API und die DB wird mit folgendem Befehl initialisiert:



Initialisierungskonfiguration von der DB:

A black and silver text on a white background

Description automatically generated

Folgende Packages müssen noch per NPM hinzugefügt werden:

* Knex
* MySQL

### Versionen

In den folgenden Abbildungen werden die Versionen vorgewiesen.

Node, NPM, Vue CLI:

A close up of a logo

Description automatically generated

MySQL Server:

A screenshot of a cell phone

Description automatically generated

MySQL Workbench:

A close up of a logo

Description automatically generated

## Ordnerstruktur

### Frontend

Die Grundstruktur wurde schon im Voraus durch Vue erstellt.

**frontend/public**Darin befindet sich das index file und das Icon.

**frontend/assets**Darin befinden sich die Anhänge wie Bilder etc.

**frontend/components**Hier befinden sich alle Vue-Komponenten

**frontend/plugins**Hier befinden sich Plugins wie Vuetify

**frontend/Router**Hier befindet sich die Rout-File, welches verantwortlich ist für die Navigation

**frontend/store**Hier befindet ich der store oder auck bekannt als vuex

**frontend/views**Hier befinden sich die einzelnen Seiten der Web-App

**backend/api**Server relevante Dateien

**backend/db**Datenbank relevante Dateien

A screen shot of a smart phone

Description automatically generated

### Backend

A screenshot of a cell phone

Description automatically generated

## Aufsetzung der Datenbank

Für dieses Projekt wird eine MySQL Datenbank eingerichtet. Sobald der Server (MySQL Community Server) und der Client (MySQL Workbench) installiert wurden, kann man gleich den Client starten und eine Verbinddung herstellen zum Server.

Nachdem der Server verbunden wurde, kann man ein DB Schema erstellen für das Projekt. Sobald das erledigt wurde, kann man Knex initialisieren und die Migrationen und die Seeds erstellen.

Knex Initialisierung:



Umgebungsanpassung in Knexfile.js:

development: {

client: 'mysql',

connection: {

host: 'localhost',

user: 'root',

database: 'tasterDay',

password: 'gibbiX123456'

},

pool: {

min: 2,

max: 10

},

migrations: {

tableName: 'knex\_migrations'

}

},

Konfigurations- und Umgebungs-Einstellung:

var config = require('./knexfile')['development'];

var knex = require('./node\_modules/knex')(config);

module.exports = knex;

### Knex Migration

Die Migration kann man erstellen, indem man folgenden Code eingibt:



Durch das Initialisieren wird ein Ordner namens «migrations» erstellt. In diesem Ordner befindet sich eine Migrationsdatei. Der Name der Datei entspricht dem Zeitpunkt, an dem die Datei erstellt wurde. Der Zeitpunkt wird in Form eines «Timestamp» dargestellt.

In dieser Datei befinden sich 2 «async» Funktionen, in denen man den Code für die DB hinzufügen kann. Mit dem Code in «Export Up» erstelle ich die Tabellen im dementsprechenden Schema und mit dem Code in «Export Down», kann ich die DB abbauen.

Export Up kann man aufrufen, indem man folgenden Befehl ausführt:



Export Down kann man aufrufen, indem man folgenden Befehl ausführt:



Der Code innerhalb der Funktionen sieht folgendermassen aus:

await knex.schema.createTable('dates', table => {

table.integer('dates\_id').primary()

table.string('dates\_start').notNullable()

table.string('dates\_end').notNullable()

table.string('dates\_time').defaultTo(20)

table.boolean('dates\_relevant').notNullable().defaultTo(1)

})

Mit «knex.schema.createTable(‘dates’)» erstellt man eine Tabelle mit dem Namen «dates». Alles innerhalb dieser Pfeilfunktion definiert die Tabellenattribute.

«table.integer(‘dates\_id’).primary()» erstellt ein Attribut namens „dates\_id“. „table.integer“ deklariert den Datentyp (INTEGER) und „primary()“ deklariert dieses Attribut als „Primary Key“.

RAW SQL Code:

await knex.raw("ALTER TABLE `tasterDay`.`dates` CHANGE COLUMN `dates\_id` `dates\_id` INT(11) NOT NULL AUTO\_INCREMENT;")

RAW SQL Code wird genutzt, um puren SQL Code auszuführen. Knex hat Probleme Attribute zu erstellen, welche automatisch zunehmen sollen (AUTO INCREMENT). Darum gibt es die Möglichkeit mit raw() puren SQL Code auszuführen.

Die Knex Migration für die Erstellung der DB Tabellen und deren Attribute sieht folgendermassen aus:

*// ANCHOR this will be executed, if latest migration is initialized.*

exports.up = async function (knex) {

*// ANCHOR Creates table 'dates'*

await knex.schema.createTable('dates', table => {

table.integer('dates\_id').primary()

table.string('dates\_start').notNullable()

table.string('dates\_end').notNullable()

table.string('dates\_time').defaultTo(20)

table.boolean('dates\_relevant').notNullable().defaultTo(1)

})

*// ANCHOR Raw SQL Code because knex can't execute all of the settings like "increment"*

await knex.raw("ALTER TABLE `tasterDay`.`dates` CHANGE COLUMN `dates\_id` `dates\_id` INT(11) NOT NULL AUTO\_INCREMENT;")

*// ANCHOR Creates table 'applicants'*

await knex.schema.createTable('applicants', table => {

table.integer('applicants\_id').primary()

table.string('applicants\_firstname').notNullable()

table.string('applicants\_lastname').notNullable()

table.integer('applicants\_age').notNullable()

table.string('applicants\_email').notNullable()

table.text('applicants\_info').notNullable()

table.integer('applicants\_dates\_id').notNullable()

table.foreign('applicants\_dates\_id').references('dates\_id').inTable('dates')

table.boolean('applicants\_present').notNullable().defaultTo(0)

table.boolean('applicants\_notified').notNullable().defaultTo(0)

table.timestamps(false, true)

})

*// ANCHOR Raw SQL Code because knex can't execute all of the settings like "increment"*

await knex.raw("ALTER TABLE `tasterDay`.`applicants` CHANGE COLUMN `applicants\_id` `applicants\_id` INT(11) NOT NULL AUTO\_INCREMENT;")

*// Creats table 'users'*

await knex.schema.createTable('users', table => {

table.integer('users\_id').primary()

table.string('users\_username').notNullable().unique()

table.string('users\_password').notNullable()

table.string('users\_firstname').notNullable()

table.string('users\_lastname').notNullable()

})

*// ANCHOR Raw SQL Code because knex can't execute all of the settings like "increment"*

await knex.raw("ALTER TABLE `tasterDay`.`users` CHANGE COLUMN `users\_id` `users\_id` INT(11) NOT NULL AUTO\_INCREMENT;")

};

*// ANCHOR this will be executed, if migrations are rolled back.*

exports.down = async function (knex) {

await knex.schema.dropTable('users')

await knex.schema.dropTable('applicants')

await knex.schema.dropTable('dates')

};

Das Skript für die Migration findet man auch im Anhang.

### Knex Seed

Die Seeds kann man erstellen, indem man folgenden Code eingibt:



Durch das Initialisieren wird ein Ordner namens «seeds» erstellt. In diesem Ordner befindet sich eine Seed-Datei mit dem Initialisierungsnamen. In diesem Fall heisst die Datei «init».

In der Seed-Datei befinden sich Beispieldaten, die in die DB eingefügt werden können.

Bcrypt import:

var bcrypt = require('../../api/node\_modules/bcrypt');

Hier wird Bcrypt importiert um das Passwort von einem Beispiel-User zu hashen.

User erstellen:

var users = async () => {

return {

users\_id: 1,

users\_username: 'admin',

users\_password: await bcrypt.hash('admin', 10),

users\_firstname: 'admin',

users\_lastname: 'admin'

}

}

Hier wird der User «admin» erstellt. Das Ganze wird in ein Objekt verpackt mit den obenstehenden Informationen wie id, username und password.

Tabellen leeren:

await knex('users').del()

await knex('applicants').del()

await knex('dates').del()

Falls die Tabellen neu befüllt werden müssen für Entwicklungszwecke, werden die bestehenden Daten gelöscht.

User-Tabelle befüllen:

await knex('users').insert(await users())

Hier wird die User-Tabelle mit dem vorher erstellten User befüllt. Es können natürlich mehrere User erstellt und hinzugefügt werden.

Daten-Tabelle befüllen:

await knex('dates').insert([{

dates\_start: '01.01.2020',

dates\_end: '02.01.2020',

dates\_time: '13.00 Uhr',

dates\_relevant: 1

},

{

dates\_start: '08.01.2020',

dates\_end: '02.01.2020',

dates\_time: '13.00 Uhr',

dates\_relevant: 1

},

{

dates\_start: '15.01.2020',

dates\_end: '16.01.2020',

dates\_time: '13.00 Uhr',

dates\_relevant: 1

},

{

dates\_start: '22.01.2020',

dates\_end: '23.01.2020',

dates\_time: '13.00 Uhr',

dates\_relevant: 1

},

{

dates\_start: '29.01.2020',

dates\_end: '30.01.2020',

dates\_time: '13.00 Uhr',

dates\_relevant: 1

},

{

dates\_start: '01.02.2020',

dates\_end: '02.02.2020',

dates\_time: '13.00 Uhr',

dates\_relevant: 1

}

])

Mit «await knex(‘dates’)» referenziert man eine Tabelle und mit insert() kann man das Objekt oder das Array hinzufügen. In diesem Fall wird ein Array mit Daten in die Tabelle «dates» gespeichert.

Die Beispieldaten können mit folgendem Befehl eingefügt werden:



Die komplette Datei sieht folgendermassen aus:

*// ANCHOR Creats default user 'admin'.*

var bcrypt = require('../../api/node\_modules/bcrypt');

var users = async () => {

return {

users\_id: 1,

users\_username: 'admin',

users\_password: await bcrypt.hash('admin', 10),

users\_firstname: 'admin',

users\_lastname: 'admin'

}

}

exports.seed = async function (knex) {

*// ANCHOR In case tables aren't empty, tables will be deleted.*

await knex('users').del()

await knex('applicants').del()

await knex('dates').del()

*// ANCHOR Inserts created user 'admin'.*

await knex('users').insert(await users())

*// ANCHOR Inserts default dates in table 'dates'*

await knex('dates').insert([{

dates\_start: '01.01.2020',

dates\_end: '02.01.2020',

dates\_time: '13.00 Uhr',

dates\_relevant: 1

},

{

dates\_start: '08.01.2020',

dates\_end: '02.01.2020',

dates\_time: '13.00 Uhr',

dates\_relevant: 1

},

{

dates\_start: '15.01.2020',

dates\_end: '16.01.2020',

dates\_time: '13.00 Uhr',

dates\_relevant: 1

},

{

dates\_start: '22.01.2020',

dates\_end: '23.01.2020',

dates\_time: '13.00 Uhr',

dates\_relevant: 1

},

{

dates\_start: '29.01.2020',

dates\_end: '30.01.2020',

dates\_time: '13.00 Uhr',

dates\_relevant: 1

},

{

dates\_start: '01.02.2020',

dates\_end: '02.02.2020',

dates\_time: '13.00 Uhr',

dates\_relevant: 1

}

])

*// Inserts default applicants in table 'applicants'*

await knex('applicants').insert([{

applicants\_id: 1,

applicants\_firstname: 'Josiah',

applicants\_lastname: 'Schiess',

applicants\_age: 20,

applicants\_email: 'josiah.schiess@gmail.com',

applicants\_info: 'Allergisch auf Milch.',

applicants\_dates\_id: 1,

applicants\_present: 1

},

{

applicants\_id: 2,

applicants\_firstname: 'Kabilan',

applicants\_lastname: 'Sivanamam',

applicants\_age: 21,

applicants\_email: 'kabilan.sivanamam@gmail.com',

applicants\_info: 'Allergisch auf Nüsse.',

applicants\_dates\_id: 2,

applicants\_present: 0

},

{

applicants\_id: 3,

applicants\_firstname: 'James',

applicants\_lastname: 'Chibuzor',

applicants\_age: 20,

applicants\_email: 'james.chibuzor@gmail.com',

applicants\_info: 'Allergisch auf Geld.',

applicants\_dates\_id: 3,

applicants\_present: 0

},

])

};

Das Skript für den Seed findet man ebenfalls im Anhang.

Um das ganze einfacher zu machen, kann man ein Skript erstellen, dass alle 3 Skripts ausführt. Dieses Skript befindet sich im package.json und sieht folgendermassen aus:

"scripts": {

"reset": "knex migrate:rollback && knex migrate:latest && knex seed:run"

},

Man kann das Skript folgendermassen ausführen:



Man muss aber darauf achten, dass man sich im richtigen Verzeichnis befindet (/05\_backend/db).

Sobald der Code ausgeführt wird, werden die Tabellen in der DB erstellt.

### ERM

Das ERM wurde mit Hilfe der «Reverse Engineering» Funktion in MySQL Workbench erstellt. Dargestellt wird das EM mit der «Martin-Notation».

A screenshot of a cell phone

Description automatically generated

## Server-Struktur einrichten

### Basis

In diesem Kapitel werden die Struktur und die Basis des Servers eingerichtet.

Um anfangen zu können, muss man in den API-Ordner. Dort erstellt man eine Server-Datei. Diese kann einen beliebigen Namen haben. Für dieses Projekt nenne ich die Datei «server.js».

Server.js sieht folgendermassen aus:

var express = require('./node\_modules/express')

var app = express()

var checkAuth = require('./middleware/checkAuth')

*// SECTION Router use*

app.use(express.json())

*// ANCHOR app (express) listens to port 3000 and logs the statement below.*

var port = 3000

app.listen(port, () => console.log('Listening to port ' + port))

*// ANCHOR Requires route for visitors and uses all the endpoints in required file.*

var visitorsRoute = require('./routes/visitors')

app.use('/visitors', visitorsRoute)

*// ANCHOR Requires route for employees, checks auth and then uses all endpoints in required file.*

var employeeRoute = require('./routes/employees')

app.use('/employees', checkAuth, employeeRoute)

*// ANCHOR Logs all reqs.*

app.use(function (req, res, next) {

console.log(`${new Date().toLocaleTimeString()} ${new Date().toLocaleDateString()} | ${req.method} request on ${req.originalUrl}`);

next()

});

*// !SECTION*

var express = require('./node\_modules/express')

var app = express()

var checkAuth = require('./middleware/checkAuth')

Diese Pakete werden gebraucht, um die App zu bauen. Express ist ein Back-End-Framework zur Erstellung von Webanwendungen und „checkAuth“ ist eine Authentifizierungs-Middleware, die im Ordner Middleware ist.

Der folgende Code zergliedert einkommende Abfragen mit JSON Fracht.

app.use(express.json())

So wird festgelegt auf welchen Port die API hören soll:

var port = 3000

app.listen(port, () => console.log('Listening to port ' + port))

Fordert die Routen für Besucher und Angestellte. Nutzt die Routen mit den entsprechenden Middlewares:

var visitorsRoute = require('./routes/visitors')

app.use('/visitors', visitorsRoute)

var employeeRoute = require('./routes/employees')

app.use('/employees', checkAuth, employeeRoute)

Macht ein log bei eingehenden Abfragen:

app.use(function (req, res, next) {

console.log(`${new Date().toLocaleTimeString()} ${new Date().toLocaleDateString()} | ${req.method} request on ${req.originalUrl}`);

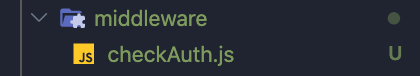
next()

});

### Middleware

Es wird ein Ordner erstellt mit Middlewares. Im Moment gibt es nur eine Middleware und die ist zuständig für die Authentifikation der User.

Die Middleware für die Authentifikation sieht folgendermassen aus:



var jwt = require('../node\_modules/jsonwebtoken')

function checkAuth(req, res, next) {

var header = req.headers.authorization;

if (typeof header !== 'undefined') {

var token = req.headers.authorization.split(' ')[1];

try {

req.decodedToken = jwt.verify(token, process.env.JWT\_TOKEN)

} catch (error) {

res.status(403)

}

next();

} else {

res.sendStatus(403).send('no Bearer token')

}

}

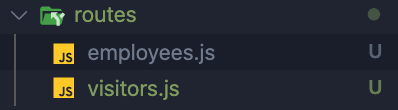
module.exports = checkAuth

### Routen

Man erstellt im API-Ordner einen Ordner mit dem Namen „routes“. In dem Routenordner kann 2 Dateien erstellen für je eine Route. Die Routen heissen „employees“ und „visitors“. In diesen 2 Ordner erstellt man später die Endpunkte / Schnittstellen für den User.

Die Routen sehen so aus (Endpunkteinträge kommen später hinzu):

Ordnerstruktur:



Route für angemeldete Benutzer:

*// SECTION Packages*

*// ANCHOR Requires all the neccesary packages.*

var express = require('../node\_modules/express')

var router = express.Router()

var knex = require('../knex')

var bcrypt = require('../node\_modules/bcrypt')

var nodemailer = require('../node\_modules/nodemailer');

*// !SECTION*

*// Exports the express router*

module.exports = router

Route für nicht angemeldete Benutzer:

*// SECTION Packages*

*// ANCHOR Requires all the neccesary packages.*

var express = require('../node\_modules/express')

var router = express.Router()

var knex = require('../knex');

var bcrypt = require('../node\_modules/bcrypt')

var jwt = require('../node\_modules/jsonwebtoken')

*// !SECTION*

*// Exports the express router*

module.exports = router

### Datenbankverbindung

Um nicht immer wieder die Verbindungsdaten einzugeben wurde eine Datei erstellt, welche die alle Informationen hat. Diese Datei wird dann importiert und immer referenziert, wenn die Verbindungsdaten angefordert werden. In diesem Fall heist die Datei «knex.js». Man muss besonders aufpassen, dass man nicht redundante Dateien hat oder Dateien die Extrem ähnlich heissen. In diesem Fall ist es aber kein Problem, da in diesem Verzeichnis keine ähnlichen Dateien sind.

In der Datei sind folgende Informationen enthalten:

*/\*\**

*\* export configured knex instance*

*\*/*

module.exports = require('knex')({

client: 'mysql',

connection: {

host: '127.0.0.1',

user: 'root',

password: 'gibbiX123456',

database: 'schnupper'

}

})

In der Datei sind Daten angegeben wird DB Client, Host-IP, Benutzername, Passwort, Name der Datenbank.

Diese Datei ist nicht notwendig, aber sehr hilfreich.

## Server-Endpunkte / -Schnittstellen einrichten

Die Endpunkte werden nach Konzept erstellt. Und sehen folgendermassen aus:

*// SECTION Packages*

*// ANCHOR Requires all the neccesary packages.*

var express = require('../node\_modules/express')

var router = express.Router()

var knex = require('../knex');

var bcrypt = require('../node\_modules/bcrypt')

var jwt = require('../node\_modules/jsonwebtoken')

*// !SECTION*

*// SECTION Dates*

*// ANCHOR Handles incoming GET-Requests for '/dates'.*

router.get('/dates', async (req, res) => {

try {

var data = await knex('dates').where({dates\_relevant: 1})

res.status(200).send(data)

console.log('Dates were requested and delivered.');

} catch (err) {

res.status(500).send(err)

console.log(err);

}

})

*// !SECTION*

*// SECTION Applicants*

*// ANCHOR Handles incoming POST-Requests for '/applicants'.*

router.post('/applicants', async (req, res) => {

var {

applicants\_firstname,

applicants\_lastname,

applicants\_age,

applicants\_email,

applicants\_info,

applicants\_dates\_id

} = req.body

try {

await knex('applicants').insert({

applicants\_firstname: applicants\_firstname,

applicants\_lastname: applicants\_lastname,

applicants\_age: applicants\_age,

applicants\_email: applicants\_email,

applicants\_info: applicants\_info,

applicants\_dates\_id: applicants\_dates\_id

})

var successMessage = {

on: true,

title: 'Danke für deine Anmeldung!',

paragraphOne: 'Wir haben deine Anmeldung zum Schnuppern erhalten und senden dir in den nächsten Tagen eine Bestätigungsmail mit allen notwendigen Informationen.',

paragraphTwo: 'Falls du trotzdem nicht kommen kannst oder willst, kannst du dich jederzeit telefonisch oder per E-Mail abmelden.',

paragraphThree: 'Wir bedanken uns nochmals für dein Interesse und freuen uns dich bald kennenlernen!',

color: 'info'

}

res.status(200).send(successMessage)

console.log()

} catch (err) {

var errorMsg = {

on: true,

title: 'Ups...',

paragraphOne: 'Die Anmeldung hat nicht funktioniert.',

paragraphTwo: 'Versuche es später nochmal.',

paragraphThree: '',

color: 'error'

}

res.status(500).send(errorMsg)

}

})

*// !SECTION*

*// SECTION Login*

*// ANCHOR Handles incoming POST-Requests for '/login'.*

router.post('/login', async (req, res) => {

var {

users\_username,

users\_password

} = req.body

var db\_user = await knex('users').where({

users\_username: users\_username

})

db\_user = db\_user[0]

if (db\_user == null) {

return res.sendStatus(400)

} else {

try {

if (await bcrypt.compare(users\_password, db\_user.users\_password)) {

var tokendata = {

id: db\_user.users\_id,

username: users\_username

}

var token = jwt.sign(tokendata, process.env.JWT\_TOKEN)

var auth = 'Bearer ' + token

res.status(200).send(auth)

console.log('User ' + db\_user.users\_id + ` '${users\_username}' has logged in.`);

} else {

res.sendStatus(403)

}

} catch (err) {

res.sendStatus(500)

console.log(err);

}

}

})

*// !SECTION*

*// Exports the express router.*

module.exports = router

Hier werden die notwendigen Pakete angefordert und sind aufrufbar mit einer Variabel:

var express = require('../node\_modules/express')

var router = express.Router()

var knex = require('../knex');

var bcrypt = require('../node\_modules/bcrypt')

var jwt = require('../node\_modules/jsonwebtoken')

Dieser Teil ist zuständig für die Handhabung aller eingehenden GET-Anfragen für «dates»:

router.get('/dates', async (req, res) => {

try {

var data = await knex('dates').where({dates\_relevant: 1})

res.status(200).send(data)

console.log('Dates were requested and delivered.');

} catch (err) {

res.status(500)

console.log(err);

}

})

Dieser Teil ist zuständig für die Handhabung aller eingehenden POST-Anfragen für «applicants»:

router.post('/applicants', async (req, res) => {

var {

applicants\_firstname,

applicants\_lastname,

applicants\_age,

applicants\_email,

applicants\_info,

applicants\_dates\_id

} = req.body

try {

await knex('applicants').insert({

applicants\_firstname: applicants\_firstname,

applicants\_lastname: applicants\_lastname,

applicants\_age: applicants\_age,

applicants\_email: applicants\_email,

applicants\_info: applicants\_info,

applicants\_dates\_id: applicants\_dates\_id

})

var successMessage = {

on: true,

title: 'Danke für deine Anmeldung!',

paragraphOne: 'Wir haben deine Anmeldung zum Schnuppern erhalten und senden dir in den nächsten Tagen eine Bestätigungsmail mit allen notwendigen Informationen.',

paragraphTwo: 'Falls du trotzdem nicht kommen kannst oder willst, kannst du dich jederzeit telefonisch oder per E-Mail abmelden.',

paragraphThree: 'Wir bedanken uns nochmals für dein Interesse und freuen uns dich bald kennenlernen!',

color: 'info'

}

res.status(200).send(successMessage)

console.log()

} catch (err) {

var errorMsg = {

on: true,

title: 'Ups...',

paragraphOne: 'Die Anmeldung hat nicht funktioniert.',

paragraphTwo: 'Versuche es später nochmal.',

paragraphThree: '',

color: 'error'

}

res.status(500).send(errorMsg)

}

})

Dieser Teil ist zuständig für die Handhabung aller eingehenden POST-Anfragen für das Login:

router.post('/login', async (req, res) => {

var {

users\_username,

users\_password

} = req.body

var db\_user = await knex('users').where({

users\_username: users\_username

})

db\_user = db\_user[0]

if (db\_user == null) {

return res.sendStatus(400)

} else {

try {

if (await bcrypt.compare(users\_password, db\_user.users\_password)) {

var tokendata = {

id: db\_user.users\_id,

username: users\_username

}

var token = jwt.sign(tokendata, process.env.JWT\_TOKEN)

var auth = 'Bearer ' + token

res.status(200).send(auth)

console.log('User ' + db\_user.users\_id + ` '${users\_username}' has logged in.`);

} else {

res.sendStatus(403)

}

} catch (err) {

res.sendStatus(500)

console.log(err);

}

}

})

Der Aufbau sieht folgendermassen aus:

router.get('/dates', async (req, res) => {

try {

var data = await knex('dates').where({dates\_relevant: 1})

res.status(200).send(data)

console.log('Dates were requested and delivered.');

} catch (err) {

res.status(500)

console.log(err);

}

})

«router» ist ein Express Router und ist eine eingebaute Middleware von Express.

Nach «router» kommt die HTTP Request Methde (GET, POST, PUT, DELETE).

In den Klammern der HTTP-Methode kommt der Name des Endpunkts hinein und eine async Pfeilfunktion mit den Parametern «req» und «res». Innerhalb dieser Pfeilfunktion kommt ein «try catch» vor. Das try catch versucht den Code im try auszuführen und falls dieser Code nicht ausgeführt werden kann, fängt es den Fehler auf und unterdrückt ihn. Danach kann man das weitere Vorgehen definieren. Dabei muss man aber beachten den Error nicht zu senden, da man so riskiert einen Einblick in die Struktur des Backend zu erhalten.

Das gleiche gilt für die Route «employees».

## Front-End UI Umsetzung

Text schreiben

## Front-End Funktionen einrichten

Text schreiben

## Validierung der Benutzereingaben

Text schreiben

## Back-End-Pfade / -Schnittstellen einrichten

Text schreiben

## Code Clean-Up

Text schreiben

## Back-End testen

Text schreiben

## Front-End testen

Text schreiben

## Code dokumentieren

Text schreiben

## Dokumentation überarbeiten

Text schreiben

# Einführungsteil

Text schreiben

## Einführungskonzept

Text schreiben

# Abbildungsverzeichnis

[Abbildung 1: Dokumentenablage 18](#_Toc33079444)

[Abbildung 2: Arbeitsplatz 19](#_Toc33079445)

[Abbildung 3: Phasen SmartArt 21](#_Toc33079446)

[Abbildung 4: Projektorganisation 25](#_Toc33079447)

[Abbildung 5: Zeitplan 32](#_Toc33079448)

[Abbildung 6: Datenbankmodell 58](#_Toc33079449)

[Abbildung 7: Softwarearchitektur 59](#_Toc33079450)

[Abbildung 8: Front-End Mock-Up Startseite 62](#_Toc33079451)

[Abbildung 9: Front-End Mock-Up Datumsauswahl 63](#_Toc33079452)

[Abbildung 10: Front-End Mock-Up Personendaten 63](#_Toc33079453)

[Abbildung 11: Front-End Mock-Up Bestätigungsnachricht 64](#_Toc33079454)

[Abbildung 12: Front-End Mock-Up Login 64](#_Toc33079455)

[Abbildung 13: Front-End Mock-Up Schnuppertagsinformationen für angemeldete Benutzer 65](#_Toc33079456)

[Abbildung 14: Front-End Mock-Up Schülerdaten 65](#_Toc33079457)

# Tabellenverzeichnis

[Tabelle 1: IPA-Daten 1](#_Toc33079389)

[Tabelle 2: Beteiligter Personenkreis 1](#_Toc33079390)

[Tabelle 3: Allgemeine Formatierung 2](#_Toc33079391)

[Tabelle 4: Tabellen Formatierung 2](#_Toc33079392)

[Tabelle 5: Standards 14](#_Toc33079393)

[Tabelle 6: Ordnerstruktur 18](#_Toc33079394)

[Tabelle 7: Lokaler Laptop 19](#_Toc33079395)

[Tabelle 8: Phasen 21](#_Toc33079396)

[Tabelle 9: Meilensteine 22](#_Toc33079397)

[Tabelle 10: Module 24](#_Toc33079398)

[Tabelle 11: Kontaktdaten 26](#_Toc33079399)

[Tabelle 12: Projektrollen 27](#_Toc33079400)

[Tabelle 13: Risikoanalyse 28](#_Toc33079401)

[Tabelle 14: Schadensausmass 29](#_Toc33079402)

[Tabelle 15: Eintrittswahrscheinlichkeit 29](#_Toc33079403)

[Tabelle 16: Risikograph vor den Massnahmen 30](#_Toc33079404)

[Tabelle 17: Risikograph nach den Massnahemen 31](#_Toc33079405)

[Tabelle 18: Zeitplanlegenden 33](#_Toc33079406)

[Tabelle 19: Zeitplanmeilensteine 33](#_Toc33079407)

[Tabelle 20: Arbeitsjournal 12.02.2020 35](#_Toc33079408)

[Tabelle 21: Arbeitsjournal 13.02.2020 36](#_Toc33079409)

[Tabelle 22: Arbeitsjournal 14.02.2020 38](#_Toc33079410)

[Tabelle 23: Arbeitsjournal 17.02.2020 40](#_Toc33079411)

[Tabelle 24: Arbeitsjournal 19.02.2020 41](#_Toc33079412)

[Tabelle 25: Arbeitsjournal 20.02.2020 42](#_Toc33079413)

[Tabelle 26: Arbeitsjournal 21.02.2020 43](#_Toc33079414)

[Tabelle 27: Arbeitsjournal 24.02.2020 44](#_Toc33079415)

[Tabelle 28: Arbeitsjournal 26.02.2020 45](#_Toc33079416)

[Tabelle 29: Arbeitsjournal 27.02.2020 46](#_Toc33079417)

[Tabelle 30: Arbeitsjournal 28.02.2020 47](#_Toc33079418)

[Tabelle 31: IST-Situation 52](#_Toc33079419)

[Tabelle 32: Abgrenzung IST 52](#_Toc33079420)

[Tabelle 33: SOLL-Situation 53](#_Toc33079421)

[Tabelle 34: Abgrenzung SOLL 53](#_Toc33079422)

[Tabelle 35: Vorgehensziele 54](#_Toc33079423)

[Tabelle 36: Funktionale Anforderungen 55](#_Toc33079424)

[Tabelle 37: Nicht funktionale Anforderungen 56](#_Toc33079425)

[Tabelle 38: Namenskonzept für Datenbanktabellen 57](#_Toc33079426)

[Tabelle 39: Namenskonzept für Datenbankattribute 57](#_Toc33079427)

[Tabelle 40: Namenskonzept für Benutzernamen 57](#_Toc33079428)

[Tabelle 41: Spezifische Back-End-Schnittstellen für nicht angemeldete Benutzer 60](#_Toc33079429)

[Tabelle 42: Spezifische Back-End-Schnittstellen für angemeldete Benutzer 61](#_Toc33079430)

[Tabelle 43 67](#_Toc33079431)

[Tabelle 44 68](#_Toc33079432)

[Tabelle 45 68](#_Toc33079433)

[Tabelle 46 69](#_Toc33079434)

[Tabelle 47 70](#_Toc33079435)

[Tabelle 48 71](#_Toc33079436)

[Tabelle 49 72](#_Toc33079437)

[Tabelle 50 73](#_Toc33079438)

[Tabelle 51 74](#_Toc33079439)

[Tabelle 52 75](#_Toc33079440)

[Tabelle 53: Freigabe 83](#_Toc33079441)

[Tabelle 54: Erster Expertenbesuch 84](#_Toc33079442)

[Tabelle 55: Zweiter Expertenbesuch 85](#_Toc33079443)

# Literatur und Quellenverzeichnis

Text schreiben

# Anhänge

Text schreiben

## Sitzungsprotokolle

Text schreiben

## Nachweis der Versionsverwaltung und Datensicherung

Kurz und genau beschreiben wie die Versionsverwaltung funktioniert.

## Code / Scripts

Keinen Quellcode copy pasten!!!!

Referenz zum Anhang machen und kurz beschreiben was der Anhang macht.

# Freigaben

Mit der Unterschrift ist die verantwortliche Fachkraft einverstanden mit der Qualität der Phase.

|  |  |  |
| --- | --- | --- |
| Phase | Projektleiter | Verantwortliche Fachkraft |
| Initialisierung | Kabilan Sivanamam | Samuel Hess |
| Konzept | Kabilan Sivanamam | Samuel Hess |
| Realisierung | Kabilan Sivanamam | Samuel Hess |

Tabelle 57: Freigabe

# Protokoll Expertenbesuch

## Erster Expertenbesuch

Datum: Mittwoch 12.02.2020

Beteiligter Personenkreis: Kabilan Sivanamam, Samuel Hess, Andy Bula

|  |  |
| --- | --- |
| Traktanden | Entscheidung |
| Firma hat keine Codeing-Standards. Welche Standards soll man benutzen? | Coding-Conventions von ICT-Berufsbildung |
| Ist Teil 2: Einführung nötig? | Phase “Einführung” weglassen und kurze Beschreibung einfügen |
| Source-Code hochladen mit oder ohne Packages? | Ohne Packages |
| Zeitplan anpassen? | Ja. Auf 1, 2 oder 4 Stundenblöcke |

Tabelle 58: Erster Expertenbesuch

## Zweiter Expertenbesuch

Datum: Montag 24.02.2020

Beteiligter Personenkreis: Kabilan Sivanamam, Samuel Hess, Andy Bula

|  |  |
| --- | --- |
| Traktanden | Entscheidung |
|  |  |
|  |  |
|  |  |
|  |  |

Tabelle 59: Zweiter Expertenbesuch