FFHS
Fernfachhochschule Schweiz
Zürich | Basel | Bern | Brig
Mitglied der SUPSI

Semesterarbeit des Studiengangs CAS Fullstack-Development

von

Kay Roman Hertenstein

Roggwil, Januar – Juni 2025

Inhaltsverzeichnis

1 Applikationsidee

Die Idee für die Applikation, kam aus meinem Freundeskreis. Einige meiner Bekannten kommen aus dem Handwerklichen Tätigkeitsgebiet – dabei ging es darum dass häufig für kleinere handwerkliche/technische Arbeiten ein Unternehmen oder ein Spezialist beauftragt werden muss – was Verhältnismässig schnell hohe Kosten verursachen kann. Daraus entstand die Idee eine Applikation zu entwickeln welche eine einfache Möglichkeit bietet Dienstleistungen / Tätigkeiten anzubieten. Das Zielpublikum sind Fachkräfte welche sich im Nebenerwerb ein zusätzliches Einkommen verdienen wollen.

1.1 Anwendungsumfang

Die Anwendung ist eine Single-Page-Applikation (SPA) mit einem Dienstleistungs-Board als zentrale Funktionalität. Das Dienstleistungs-Board ermöglicht eine effiziente Suche und Navigation innerhalb der verfügbaren Angebote durch eine leistungsfähige Volltextsuche und Filterfunktionen.
Benutzer können sich registrieren und nach der Anmeldung eigene Dienstleistungs-Angebote erstellen, bearbeiten und löschen. Zudem sollen registrierte Nutzer ein öffentliches Profil mit ihren Kontaktinformationen erstellen und verwalten, welches für Nutzer als Referenz der angebotenen Dienstleistungen sichtbar ist.

1.1.1 Nicht - Funktionale Anforderungen Frontend

Das Frontend ist eine Webbasierte Benutzeroberfläche, die eine effiziente Interaktion mit der Applikation ermöglicht. In den nicht funktionalen Anforderungen werden die Rahmenbedingungen der Applikation sowie die Merkmale der Benutzerfreundlichkeit definiert. Die jeweilige nicht funktionale Anforderung enthält einen Verweis [**NFAFE-ID**] auf die genaue Spezifizierung, diese sind im Dokument **Spezifizierung_technische_Anforderungen_kh** Abschnitt 1.1 Spezifizierung Nicht Funktionale Anforderungen Frontend NFAFE definiert.

Allgemeines

- Responsives Webdesign **[NFAFE ID-001]**:
    - Optimierung der Benutzeroberfläche für verschiedene Endgeräte (Desktop, Tablet und Smartphone).

Perfomance [NFAFE ID-002]:

- Die Applikation soll ein genügendes Resultat im Prüfen mit Perfomance-Analyse Tools (z.B. Google Lighthouse) erzielen.

Verfügbarkeit [NFAFE ID-003]:

- Die Anwendung gewährleistet eine konsistente Nutzerfreundlichkeit durch benutzerfreundliche Fehlerseiten und angemessene Fehlermeldungen.

Sicherheit

- Nutzung von JWT (JSON Web Token) mit sicheren Ablaufmechanismen. **[NFAFE ID-004]**
- Formulare **[NFAFE ID-005]**:
    - Validierung der Eingaben zur Sicherstellung der Datenintegrität und Verhinderung von Injection-Angriffen.

- Implementierung eines «Honeypot»-Feldes zur Erkennung und Abwehr von Bot-Eingaben.
- Begrenzung der maximalen Anzahl an Anmeldeversuchen zur Prävention von Brute-Force-Angriffen. (optional)

1.1.2 Funktionale Anforderungen Frontend

In den funktionalen Anforderungen werden die konkreten Anwendungsfälle die das Frontend mit der dazu passenden Logik umsetzen soll definiert. Die jeweilige funktionale Anforderung enthält einen Verweis [**FAFE ID**] auf die genaue Spezifizierung, diese sind im Dokument **Spezifizierung_technische_Anforderungen_kh** Abschnitt 1.2 Spezifizierung Funktionale Anforderungen Frontend FAFE definiert.

Allgemeine Funktionen

- Navigation innerhalb der Webanwendung **[FAFE ID-001]**:
  - Bereitstellen einer Navigation für eine optimale Benutzerführung.
- Kontaktformular **[FAFE ID-011]**:
  - Bereitstellen einer Kontakformulars für Support- oder allgemeine Anfragen.
- Marketinganalysen **[FAFE ID-012]**:
  - Bereitstellen einer Übersicht der definierten Analysen/Auswertungen.
- Usability **[FAFE ID-013]**:
  - Eine URL kann geteilt um eine spezifische Suche erneut darzustellen.

- Dienstleistungs-Übersicht und Suchfunktion:
  - Auflistung von Dienstleistungen mit relevanten Informationen (Titel, Beschreibung, Kategorie). **[FAFE ID-002]**
  - Volltextsuche & Filterung **[FAFE ID-003]**:
    - Implementierung einer leistungsfähigen Suchfunktion für gezielte Dienstleistungs-Suche.
    - Filteroptionen nach Kriterien: Kategorie, Ort
  - Anzeige Kontaktinformationen/öffentliches Profil **[FAFE ID-004]**:
    - Durch die Auswahl einer Dienstleistung erfolgt eine Weiterleitung zur Profilvorschau-Seite mit Kontaktinformationen des Dienstleisters.

- Registrierung und Authentifizierung **[FAFE ID-005]**:
  - Bereitstellung von nutzerfreundlichen Registrierungs- und Anmeldeformularen mit Validierung der Eingaben.
  - Ein Benutzer muss sich registrieren und anmelden können, um Zugriff auf sein persönliches Dashboard und die verfügbaren Funktionen der Plattform zu erhalten.
  - Ein Benutzer soll die Möglichkeit haben sein Passwort zurückzusetzen. (Passwort vergessen) **[FAFE ID-010]**

Funktionen für angemeldete Benutzer

- Dashboard & Verwaltung:
  - Bereitstellung einer zentralen Übersicht. **[FAFE ID-006]**
  - Verwaltung von Kontoeinstellungen und Profildaten. **[FAFE ID-007]**
- Dienstleistungs-Verwaltung **[FAFE ID-008]**:
  - Möglichkeit, eigene Dienstleistungen zu erstellen, bearbeiten und löschen (CRUD-Operationen).

    - Möglichkeit, eine Suche für Dienstleister zu erstellen, bearbeiten und löschen (CRUD-Operationen).
- Öffentliches Profil **[FAFE ID-009]**:
    - Erstellung und Verwaltung eines öffentlichen Profils, das für andere Nutzer oder Interessenten sichtbar ist.

1.1.3 Nicht Funktionale Anforderungen Backend

In den nicht-funktionalen Anforderungen werden die Rahmenbedingungen des Backends sowie spezifische Merkmale zur Benutzerfreundlichkeit und Leistung definiert. Jede Anforderung enthält einen Verweis **[NFABE-ID]** auf die genaue Spezifizierung, diese sind im Dokument **Spezifizierung_technische_Anforderungen_kh** Abschnitt 1.3 Spezifizierung Nicht Funktionale Anforderungen Backend NFABE definiert.

Leistung [NFABE ID-001]
- Das System soll eine akzeptable Performance bereitstellen (Geschwindigkeit API – Anfragen)
- Datenbankabfragen sollen optimiert sein und mit minimaler Latenz ausgeführt werden.

Sicherheit [NFABE ID-002]
- Datenverschlüsselung: alle sensiblen Daten müssen bei der Speicherung verschlüsselt werden.
- Schutz vor Angriffen: Das System implementiert Schutzmechanismen vor gängigen Angriffsmethoden.
- Authentifizierung und Autorisierung: Das Backend muss eine sichere Authentifizierung implementieren.

1.1.4 Funktionale Anforderungen Backend

In den Funktionalen Anforderungen werden die konkreten Anwendungsfälle spezifiziert und deren Funktionsablauf beschrieben. Jede Anforderung enthält einen Verweis **[FABE-ID]** auf die genaue Spezifizierung, diese sind im Dokument **Spezifizierung_technische_Anforderungen_kh** Abschnitt 1.4 Spezifizierung Funktionale Anforderungen Backend FABE definiert.

Request-Verarbeitung [FABE ID-001]
- Entgegennahme und Bearbeitung eingehender Anfragen gemäss definierten API-Spezifikationen.

Datenverarbeitung und Konsistenz [FABE ID-002]
- Datenentgegennahme:
    - Validierung und Parsing eingehender Datenstrukturen.
- Datenspeicherung:
    - Persistente Speicherung in einer relationalen Datenbank.

Benachrichtigung [FABE ID-010]
- Das System kann automatisierte Benachrichtigungen via Email machen.

Analyse für Marketingzwecke [FABE ID-011]
- Das System erstellt definierte Auswertungen bezüglich Anmeldungen, Dienstleistungen und Kategorien.

API – Endpunkte
Bereitstellung von RESTful – API-Endpunkten für folgenden Entitäten:

- User – Management:
    - Authentifizierung **[FABE ID-003]**:
        - Sicherer Login-Prozess mit Passwort-Hashing und Token-basierter Authentifizierung JWT.
        - Passwort zurücksetzten via Email/Authorisierungslink. **[FABE ID-009]**
    - Registrierung **[FABE ID-004]**:
        - Erstellung neuer Benutzerkonten.
    - Kontoverwaltung (CRUD) **[FABE ID-005]**:
        - Bearbeitung, Aktualisierung und Löschung des Benutzerkontos.

- Dienstleistungen:
    - Abrufen aller verfügbaren Dienstleistungen. **[FABE ID-006]**
    - Verwaltung von Dienstleistungs – Daten mittels CRUD-Operationen (Create, Read, Update, Delete). **[FABE ID-007]**

- Öffentliches Profil **[FABE ID-008]**:
    - Verwaltung des öffentlichen Profils mittels CRUD-Operationen (Create, Read, Update, Delete).

1.1.5 API-Spezifikationen Backend
Die API-Spezifikation wurde basierend auf den funktionalen Anforderungen im Dokument **"Spezifizierung technische Anforderungen"** definiert. Die erste API-Spezifikation dient als Vorlage für die initiale Implementierung und stellt eine prototypische Spezifikation dar. Sie orientiert sich an den vorgegebenen Designmustern und wird entsprechend des Modulunterrichts optimiert.

Allgemeine API-Prinzipien:

- **Format:** JSON
- **Authentifizierung:** Bearer Token mit JWT
- **Statuscodes:** Standard-HTTP-Statuscodes zur Fehler- und Erfolgsmeldung
- **Versionierung:** `keine Versionnierung`

1.1.5.1 API-Endpunkte

Die API- Endpunkte wurden detailliert im Code mit ‘drf-spectacular’ dokumentiert. Nach erfolgreicher Inbetriebnahme des Projekts gemäss README.md ist die Dokumentation auf der Root-URL des Backends verfügbar (http://127.0.0.1:8000/).
Im Folgenden eine Zusammenfassung der Endpunkte:

Usermanagement:
- **POST /api/auth/register/** – Registriert einen neuen Benutzer
- **GET /api/auth/registered-user/** – Benutzerdaten gemäss Token bereitstellen
- **PATCH /api/auth/registered-user/** – Benutzerdaten updaten
- **DEL /api/auth/registered-user/** – Benutzer (Account) löschen
- **GET /api/dashboard-data/** – Dashboard Daten für autorisierten User abfragen

Autorisierung:
- **POST /api/auth/login/** – Benutzerlogin
- **POST /api/auth/logout/** – Benutzerlogout / Token Invalidation
- **POST /api/auth/password-forgot/** – Passwort-reset Link anfordern
- **POST /api/auth/password-reset/** – Passwort Reset
- **POST /api/auth/password-change/** – Passwort Update
- **POST /api/auth/refresh/** – Access Token erneuern
- **POST /api/auth/verify/** – Gültigkeit Token prüfen

Dienstleistungen:
- **POST /api/public-service/** – Neues Inserat/Dienstleistung erstellen
- **PUT /api/public-service/** – Ein Inserat/Dienstleistung updaten
- **DEL /api/public-service/** – Ein Inserat/Dienstleistung löschen

Öffentliches Profil:
- **PATCH /api/public-profile/** – Öffentliches Profil updaten
- **GET /api/public-profile/get/<id>** – Öffentliches Profil abrufen

Analysen / Statistiken:
- **GET /api/analytics-data/** – Statistiken abrufen
- **POST /api/analytics-data/create-visit/** – Eintrag Seitenbesuch erstellen

Weitere:
- **POST /api/contact-message/** – Kontaktanfragen
- **GET /api/public-data/**– Öffentliche Daten Abrufen

1.1.6 Datenbank Spezifikationen Backend

Das folgende Diagramm beschreibt die Struktur und Spezifikation der relationalen Datenbank des Backend-Systems.

2 Übersicht Technologien

Der Technologie Stack wurde auf Basis des Modulunterrichts definiert.

2.1 Architektur

Für die Umsetzung des Projekts habe ich mich bewusst für eine Trennung von Frontend und Backend entschieden. Diese Entscheidung basiert auf der Zielsetzung, eine konsistente und modulare Architektur der Applikation zu gewährleisten. Zudem wurde die Annahme berücksichtigt, dass die Anwendung in Zukunft weiterentwickelt wird, etwa durch die Entwicklung einer mobilen App mit React Native (dieser Aspekt ist jedoch nicht Teil der aktuellen Semesterarbeit).

Die Anwendung wird mit **Client-Side Rendering (CSR)** entwickelt. Diese Wahl habe ich getroffen, mit der Aussicht, dass die Komplexität der Anwendung in späteren Entwicklungsstadien deutlich zunimmt.

2.2 Frontend - Stack

- **Framework**: React.js (v19.0.0)
- **Programmiersprache**: JavaScript (ES6+)
- **Styling**: CSS / SCSS
- **State Management**: Redux
- **UI– Bibliothek**: Material-UI (MUI)
- **Build-Tool**: Vite
- **Testing**: Vitest, React Testing Library

2.3 Backend – Stack

- **Programmiersprache**: Python (3.11)
- **Framework**: Django (v5.1.7)
- **Datenbank**: PostgreSQL
- **API-Schnittstelle**: Restful API
- **Authentifizierung**: JWT (JSON Web Token)
- **Testing**: Python Unittest

2.4 Infrastruktur & Deployment

- **Versionskontrolle**: Git / Gitlab
- **Containerisierung**: Docker mit Docker Compose (optional)

3 Zeitmanagement

Ich habe das Zeitmanagement unter Berücksichtigung des Modulunterrichts definiert. Besonders habe ich bei der Planung Reserve Zeit eingeplant. Jede Phase wird im Dokument: **Semesterarbeit_Logbuch_kh** festgehalten.

3.1 Zeitmanagement Frontend

Phase	Aufgabe	Startdatum	Dauer	Logbuch/ID
Planung & Vorbereitung	Erstellen der Vorgaben / Spezifizierung	08.02.25	4 Wochen	1.1
Setup & Grundstruktur	Initialisierung des Projekts (Git, Basisstruktur)	22.02.25	2 Wochen	1.2
	Einarbeiten UI Libary / Theme erarbeiten	22.02.25	2 Wochen	1.2
UI/UX-Entwicklung	Umsetzung der Kernansichten gemäß Mockups	14.03.25	2 Wochen	1.2
Interaktive Funktionen & Logik	Implementierung der UI-Logik mit Testdaten	28.03.25	4 Wochen	1.2, 1.3
Testing & Optimierung	UI-Tests (Vitest, React Testing Library) & Refactoring	02.05.25	TDD / 4 Wochen	1.4
Finalisierung & Backend-Anbindung	Integration mit Backend & API-Anbindung	02.05.25	4 Wochen	1.2, 1.3, 1.4
	Präsentation	07.06.25		
	Reserve – für finale Anpassungen & Abgabe	07.06.25	2 Wochen	1.4, 1.5

3.2 Zeitmanagement Backend

Phase	Aufgabe	Startdatum	Dauer	Logbuch/ID
Planung & Vorbereitung	Erstellen der Vorgaben / Spezifizierung	08.02.25	4 Wochen	1.1
Setup & Grundstruktur	Initialisierung des Projekts (Git, Basisstruktur)	08.03.25	1 Woche	1.2
Datenbank & API-Design	Datenbankmodellierung & Migrationen (ORM, Relationsstrukturen)	15.03.25	4 Wochen	1.2
API-Entwicklung	Implementierung der API-Endpunkte & Geschäftslogik	12.04.25	4 Wochen	1.2, 1.3
Sicherheit & Authentifizierung	Implementierung von JWT (Authentifizierung & Autorisierung)	10.05.25	2 Wochen	1.2, 1.3
Admin-Panel (optional)	Konfiguration des Admin-Panels			
Testing & Refactoring	Testen der API-Endpunkte & Refactoring	24.05.25	2 Wochen	1.4
Finalisierung & Frontend-Integration	Frontend-Integration & Feinabstimmung	24.05.25	2 Wochen	1.4, 1.5
	Präsentation	07.06.25		
	Reserve – für finale Anpassungen & Abgabe	07.06.25	2 Wochen	1.4, 1.5

4 Wireframes & Mockup

Die Wireframes wurden mit Figma umgesetzt, dabei ist zu erwähnen, dass, das Mockup eher für den visuellen Aspekt und den Vorgaben des Moduls entsprechend umgesetzt wurde. Ich beziehe mich hier auf den Prozess der agilen Entwicklungsmethode besonders bei der Gestaltung und Design wähle ich einen möglichst flexiblen Ansatz um mich optimal der verwendeten UI Libary anzupassen.

https://www.figma.com/design/TEmmnTdPA9ZjcmP3sihkc1/Sidelink

5 Logbuch

Für eine konsistente und übersichtliche Logbuchführung wurde ein separates Dokument erstellt. **Semsesterarbeit_Logbuch_kh**

6 Reflexion

In diesem Projekt werden die meisten Themen abgedeckt, die in diesem Modul behandelt werden. Die Idee einer Dienstleistungsplattform bietet zahlreiche Möglichkeiten zur Verfeinerung und Erweiterung – beispielsweise durch ein zusätzliches Board zur Fachkraftsuche. Dennoch habe ich mich bewusst auf die essenziellen MUSS-Anforderungen konzentriert, um den Fokus der Spezifikation klar zu definieren und den Zeitplan einhalten zu können. Ich freue mich auf die praktische Umsetzung des Projekts und bin gespannt auf das Feedback der Lehrpersonen.

Quellenverzeichnis

- Umsetzung Wireframes & Mockup (Design Elemente als Vorlage): https://modernize-react-dark.netlify.app/dashboards/ecommerce
- Umsetzung Wireframes & Mockup generell: https://mui.com/?srsltid=AfmBOorVyUHTYQ89sF24XxX7p28YZZNYl2YpSHVpUFg7Z1wvQjnoHJYf

Anhang

- Dokument: Spezifizierung_technische_Anforderungen_kh
- Dokument: Semesterarbeit_Logbuch_kh
- Dokument: Release_semesterarbeit
- Präsentation