

# Plattform Freiwilligenarbeit – Konzept & Prototyp

<b>Themenbereiche:</b>	Anforderungsanalyse, Softwarearchitektur, Webentwicklung
<b>Studierender:</b>	Marcel von Wyl, Philipp Gröbelbauer
<b>Dozent:</b>	Prof. Jörg Hofstetter
<b>Experte:</b>	Dominique Portmann
<b>Wirtschaftspartner:</b>	Stadt Sempach
<b>Keywords:</b>	Web-Application, REST API, Java, Angular, Requirements Engineering

## 1. Aufgabenstellung

In Sempach bestehen bereits diverse Angebote, die von Einwohnern auf freiwilliger Basis durchgeführt werden. Es wird beispielsweise Unterstützung für Stellenlose und Asylsuchende angeboten und auch in anderen Bereichen werden Einsätze geleistet. Dennoch sind sich viele ihrer Möglichkeiten nicht bewusst – sie wissen nicht, wo sie sich freiwillig engagieren können. Anderen ist unklar, an welche Stelle sie sich wenden können, um Unterstützung durch Freiwillige in Anspruch zu nehmen. Um dem entgegenzuwirken soll die Freiwilligenarbeit und Nachbarschaftshilfe in der Stadt Sempach zukünftig besser koordiniert sein.

Ziel der vorliegenden Arbeit war es, ein Konzept und einen funktionalen Prototyp einer Webapplikation zu erstellen, die zukünftig zur Koordination der Freiwilligenarbeit in Sempach eingesetzt werden soll. Der Projektauftrag umfasste ausserdem eine ausführliche Anforderungsanalyse mit dem Kunden, sowie eine Konkurrenzanalyse.

## Ergebnisse

Im Rahmen dieser Bachelor-Diplomarbeit wurden folgende Ergebnisse erarbeitet:

- **Anforderungen:**  
In einem iterativen Prozess, der mehrere Wochen dauerte, wurden gemeinsam mit dem Kunden die Anforderungen an das System ermittelt. Einerseits wurden sie schriftlich in Form von User Stories festgehalten, andererseits wurde auch ein visueller Prototyp erstellt.
- **Konkurrenzanalyse:**  
Um sicherzustellen, dass sich die Webapplikation von bestehenden Lösungen abheben kann, wurde eine Konkurrenzanalyse durchgeführt.
- **Java Backend:**  
Es wurde ein Java Backend entwickelt, das Ressourcen über eine REST Schnittstelle anbietet. Es verwaltet Daten in einer PostgreSQL Datenbank und übernimmt weitere

Funktionen wie Sicherheit (Authentifizierung und Autorisierung) oder den automatischen Versand von E-Mails.

- **Angular Frontend:**  
Das Frontend der Webapplikation wurde mit Angular umgesetzt. Es umfasst sowohl Darstellung als auch Logik der einzelnen Seiten, die durch den Einsatz des Frameworks Bootstrap «responsive» sind.
- **REST API:**  
Die Kommunikation zwischen Frontend und Backend geschieht über eine REST Schnittstelle. Die Definition dieser Schnittstelle wurde separat entworfen und dokumentiert.

## Lösungskonzept

Um das Bedürfnis nach besserer Koordination der Freiwilligenarbeit zu befriedigen, wurde eine Webapplikation konzipiert, die folgende Funktionalitäten bietet:

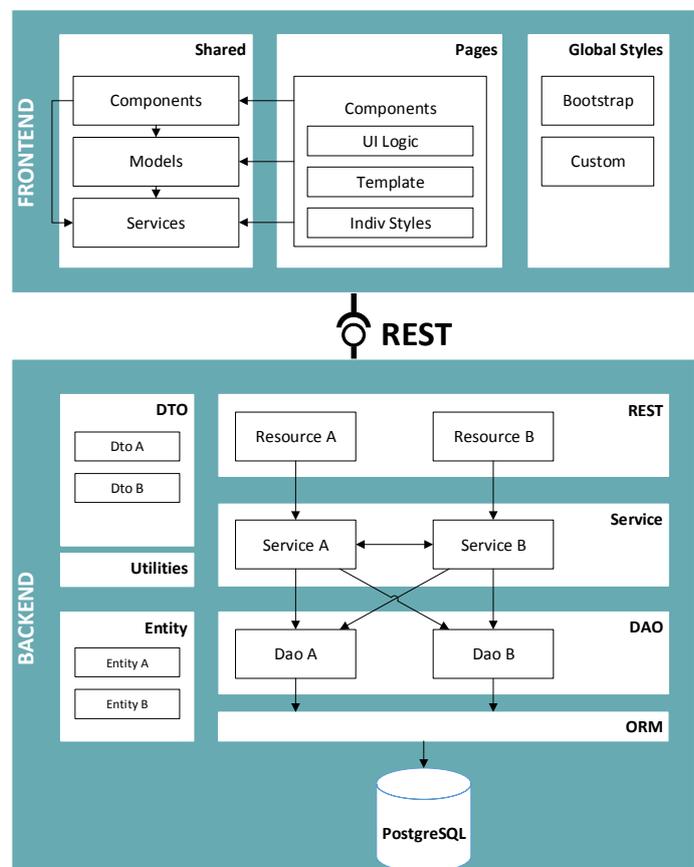
- Wer sich freiwillig engagieren möchte, sieht eine Liste von Inseraten, auf die er/sie sich direkt bewerben kann.
- Wer Hilfe durch Freiwillige in Anspruch nehmen möchte, der/die sieht eine Liste von bestehenden Angeboten, die jeweils auch eine Kontaktperson angeben.
- Diese Inserate und Angebote werden verwaltet durch Organisationen, die die Freiwilligenarbeit koordinieren.

## Architektur

Rechts ist die entworfene Architektur für die Webapplikation zu sehen. Das Frontend bezieht über eine REST Schnittstelle die Ressourcen beim Backend.

Das Backend ist in mehrere Schichten unterteilt. Die DAO-Schicht (Data Access Object) greift über den ORM auf die PostgreSQL Datenbank zu. Sie wird verwendet von der darüberliegenden Service-Schicht, die alle Geschäftslogik abhandelt. Die oberste Schicht ist für das Anbieten der Ressourcen über die REST-Schnittstelle zuständig. Ausserdem werden ab der Service-Schicht und für die Datenübertragung DTO's (Data Transfer Object) eingesetzt. Die unteren Schichten arbeiten mit Entities, die den Zugriff auf die Datenbank abstrahieren und vereinfachen.

Die Architektur des Frontends ist grösstenteils vom Angular Framework vorgegeben. Sie besteht aus Pages, Components, Models und Services. Die Darstellung wird durch Bootstrap und eigene CSS Regeln definiert.



## Sicherheitskonzept

Die Sicherheit ist ein wichtiger Aspekt, da es sich um eine öffentlich verfügbare Applikation handelt. Auf ein Login für alle Benutzer wurde aber bewusst verzichtet, da dies für einige Personen abschreckend wirken könnte. Die folgenden Punkte zeigen das Grundprinzip des Sicherheitskonzepts auf:

- Jegliche Kommunikation zwischen Server und Client ist mit SSL verschlüsselt.
- Grundsätzlich sind alle Schreibzugriffe abgesichert, sprich nur für authentifizierte Benutzer erlaubt (eine Ausnahme ist die Bewerbung).
- Ein User meldet sich mit Benutzername und Passwort an, sodass er einen Auth-Token erhält, der bei jedem Request mitgesendet wird.
- Die Authentifizierung und Autorisierung findet immer und ausschliesslich auf dem Server statt.
- Ob eine Ressource oder eine Operation geschützt ist, und welche Rollen dafür berechtigt sind wird jeweils direkt auf der Ressource respektive der Operation definiert.

## 2. Spezielle Herausforderungen

### Anforderungsanalyse

Beim Aufnehmen der Anforderungen zeigten sich bereits früh im Projekt Stolpersteine. Erfahrung im Bereich der Anforderungsanalyse war kaum vorhanden und die Kommunikation mit dem Auftraggeber führte mehrmals zu Missverständnissen. Einerseits war es herausfordernd die Freiwilligenarbeit-spezifischen Sachverhalte zu verstehen. Andererseits war es auch schwierig die technische, für den Auftraggeber schwer verständliche Seite nicht in die Diskussion einzubringen. Wiederholte Besprechungen führten jedoch zu deutlich besseren Ergebnissen.

### REST API

Neuland für beide Projektmitglieder und somit eine spezielle Herausforderung war das Entwerfen des REST APIs. Es mussten alle relevanten Ressourcen ermittelt und in eine sinnvolle Struktur gebracht werden. Die verschiedenen Zugriffe darauf – erstellen, lesen, verändern, löschen (CRUD) - galt es zudem konsistent auf http-Methoden abzubilden.

### Stateless Server

Auf dem Server wird kein Zustand gespeichert. Es gibt auch keine Usersession. Dies wurde möglich durch den Einsatz des self-contained JSON Web Token (JWT), der jegliche zum Token gehörenden Informationen in sich selbst speichert. Ein Authentication- und Authorization-Filter fängt jeden Request ab, überprüft ob die angeforderte Ressource gesichert ist, wenn ja, ob der Token gültig und berechtigt ist und erlaubt oder blockiert am Ende die Anfrage.

Etwas komplizierter wurde es mit dem Organisationszugriff. Denn Organisation A darf nicht auf die Daten von Organisation B zugreifen, aber beide Organisationen haben die Rolle um auf die entsprechende Ressource zuzugreifen. Deshalb muss an den entsprechenden Stellen eine Methode aufgerufen werden, die überprüft, ob die Organisation die im Token mitgeliefert wird, auch die Organisation ist, auf die zugegriffen werden soll.

## 3. Ausblick

Die Vision des Auftraggebers ist erst dann erfüllt, wenn die Webapplikation produktiv im Einsatz ist. Im Rahmen der Diplomarbeit wurde viel erreicht, aber einsatzbereit ist der Prototyp nicht. Es fehlen beispielsweise Registrierungsfunktion, einige Seiten im Frontend, sowie Statusmeldungen.

Auf jeden Fall soll die Software in einem mittelfristigen Zeithorizont für die Stadt Sempach im produktiven Einsatz zur Verfügung stehen. Um dies zu ermöglichen wird die Arbeit voraussichtlich vom Projektteam dieser BDA oder einem Teil davon weitergeführt. Unter welchen Bedingungen dies geschehen wird, ist zu diesem Zeitpunkt nicht klar.

Langfristig gesehen ist eine Ausbreitung auf weitere Gemeinden möglich. Bereits im Rahmen der BDA wurde auch mit dem Sozialvorsteher der Gemeinde Meggen über den visuellen Prototypen und das Konzept diskutiert, wobei er Interesse an der Plattform angemeldet hat. Auch weitere Gemeinden sehen Verbesserungspotenzial bei der Koordination der Freiwilligenarbeit und haben ein Budget für mögliche Projekte. Die Situation muss gründlich analysiert werden, aber möglicherweise besteht der Markt für eine Standardsoftware.