Handout – Dinder

Tobias Raible, Lukas Schulz, Jannis Fuchs, Jonathan Schäfer, Manuel Franz

---

Wir haben uns für die Entwicklung einer Webanwendung entschieden, deren Hauptziel es ist, die Vermittlung verschiedener Dienstleistungen zu erleichtern. Unter dem Namen Dinder präsentieren wir euch nun unser aufregendes Projekt, das als eine Art "Tinder für Dienstleistungen" konzipiert ist.

Wir wollen mit Dinder eine neue Möglichkeit bieten, die Art und Weise zu revolutionieren, wie Menschen Dienstleistungen finden und anbieten.

Inhalt

Aufgabenverteilung

- Jonathan Schäfer – Backend -> Hibernate, REST-Endpoints
- Tobias Raible – Backend -> Redoc, Security (JWT), teils Hibernate
- Lukas Schulz – Frontend -> Main Page, Swipen
- Jannis Fuchs – Frontend -> Login, Registration
- Manuel Franz – Database -> Datenbankerstellung, Verbindung zum Backend

Zeitmanagement

Die bisherige Projektphase haben wir in vier Sprints mit einer Länge von je 2 Wochen eingeteilt. In der folgenden Tabelle sind die jeweiligen Zeiten der Teilnehmer dieses Projekts in die Sprints untergliedert aufgeschlüsselt.

		Sprint 1 17.10 - 31.10	Sprint 2 31.10 - 14.11	Sprint 3 14.11 - 28.11	Sprint 4 28.11 - 12.12
Gesamtzeit in h	95	11	32	29,75	22,25
Tobias Raible	16,5	1,5	5	7,5	2,5
Jonathan Schäfer	18,3	2,5	4,5	5,75	5,5
Lukas Schulz	18	2,5	9,5	3,5	2,5
Manuel Franz	22,3	2,5	7	5,5	7,25
Jannis Fuchs	20	2	6	7,5	4,5

Tabelle 1: Zeitüberblick

In der Abbildung 1 (Zeitmanagement) sind die summierten Zeiten der Teilnehmer über die Sprints verteilt dargestellt. Zur besseren Orientierung sind auch die Durchschnittswerte der ersten vier Sprints angegeben (blaue Linie).

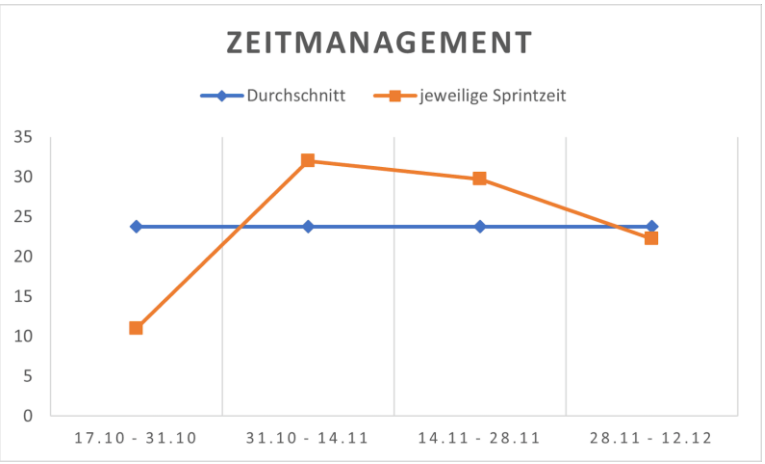


Abbildung 1: Zeitmanagement

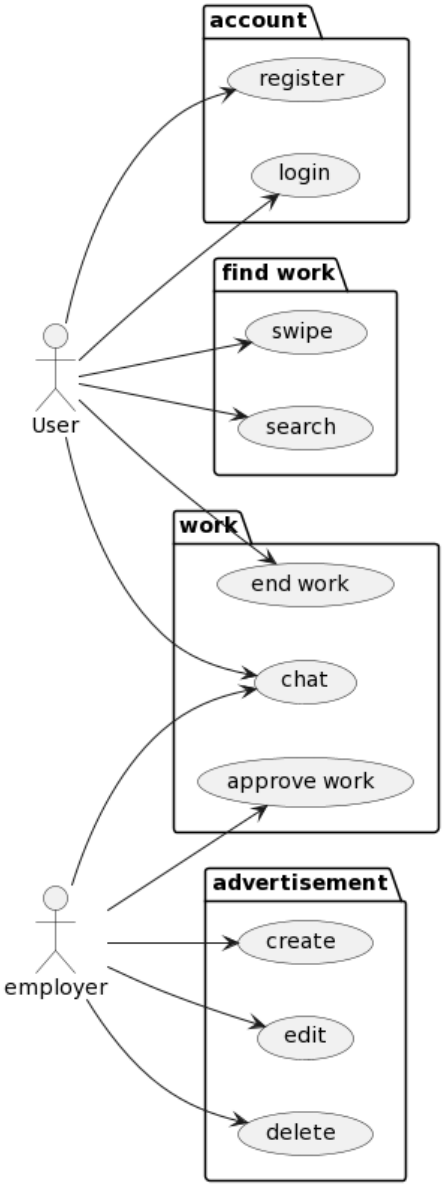
Tech-Stack

- **Frontend**
  - React
  - TypeScript
  - Redux
  - styled-components
  - vite

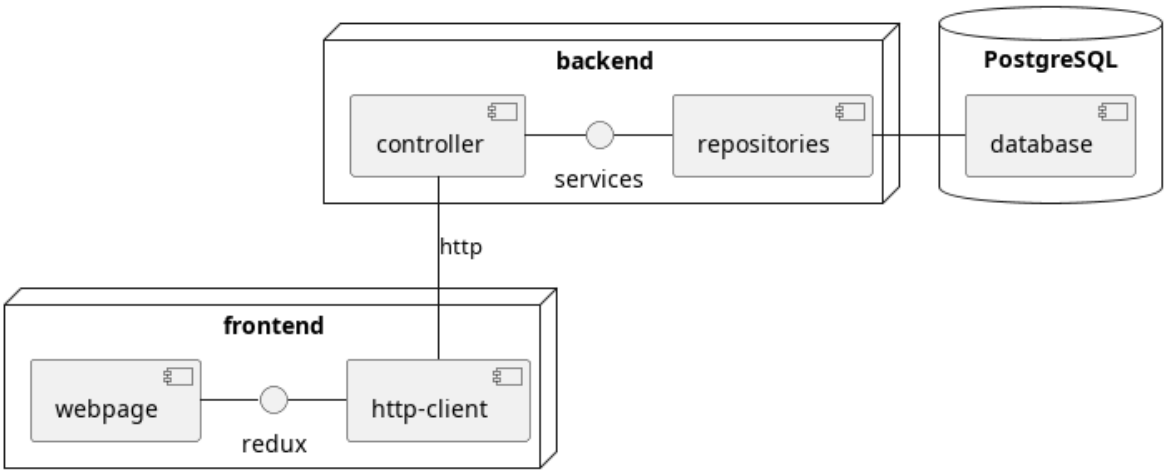
- **Backend**
  - Spring Boot
  - Hibernate
  - Redoc mit Springdoc
  - JSON Web Tokens
- **Datenbank**
  - PostgreSQL
  - Supabase
  - Flyway

Use-Case Diagramm

Im folgenden Use-Case Diagramm findet eine Aufteilung in User und Employer statt. Der Employer kann neue Dienstleistungsanzeigen erstellen. Der User kann mit der typischen Tinder Swipe Funktion eine Dienstleistungsanzeige auswählen.



Architekturstile/-entscheidungen



Generell haben wir das MVC-Pattern benutzt.

Dabei stellt die Webpage im Frontend die View dar. Der Controller im Backend entspricht dem Controller im MVC-Pattern. Die Entity-Klassen im Backend, sowie die Repository-Klassen repräsentieren das Model im MVC-Pattern.