Design and Architecture

The Project is devided in Java Backend with a PostgreSQL Database and a React Frontend.

Backend

The Database is used as a Docker Container. This command was used to get the DB up and running:

```
docker run -d --rm --name postgresdb -e POSTGRES_USER=postgres -e POSTGRES_PASSWORD=postgres -p 5432:5432 -v pgdata:/var/lib/postgresql/data postgres
```

PGAdmin was used as a UI-Tool to access the Database and fool around with Queries and App-Testing.

To be specific about the layered structure: I used the structure of **persistence-layer, service-layer and presentation-layer** as mentioned in the course. The persistence is implemented by Jakarta and PostgreSQL, the service-layer uses Spring and the presentation-layer is built in React.

The communication between persistence and service-layer is handled
by DTOs, the communication between Front- and Backend is handled by a REST API that uses http-requests. In the project bones there is also a Postman Collection of some Requests.

The implementation of the MapApi class handles communication with openrouteservice to retreive addresses as well as directions.

For logging Slf4j was used.

Lessons Learned

Separating the layers allows changes to be made without affecting
the rest of the project a lot. I can see many advantages for developing in teams because it allows splitting development between people. As I had to do it alone I couldn't experience those advantages though.

Also the use of Hibernate makes it incredible easy to implement relational Database queries fast, which is cool.

Unit Tests

The unit tests check the functionalities of the service-layer because this is the most critical part of the program. There are no Unit tests for the frontend.

Time Spent

I spent about 25-30 hours on the project. It was done within a weekend. Obviously not everything was impleneted in this time.

Link to Git

https://github.com/JhnnsEbck/SWEN2_TourPlanner.git