Challenges and Risks

Our project had risks in many ways. Many problems that we could not solve had to be solved creatively. As an example of such, we now present the following problem:

As learned in the database lecture we wanted to model our database with inheritance. So companies and admins (and later also extensions, e.g. students) should all be users. Therefore we had to define foreign keys and foreign key constraints between the tables “company”, “admin” and “user”. We could have done this without problems in raw SQL, but not in sequelize and certainly not in the less documented sequelize-typescript. For a long time, we tried out what the framework does with which input values. We found out that we can read the exact SQL commands in the terminal which sequelize executes.

After further trying the structure of the database was correct, so we were faced with the next problem. If now a user was deleted, then also the connected entries from "Company" or "Admin" should be deleted. Finally we were able to implement this as well. So far we could not test anything, because we did not have a working frontend or requests yet.

As we spent more time on the project, we got to know many useful tools.

- Internet with answers to common problems (especially StackOverflow)
- “SQlite DB Browser" for viewing databases
- Postman for automated/organized sending of requests (at first we always sent requests with the browser, until we noticed that the browser only sends get-requests)

Learning outcomes

We have strictly separated the backend and frontend teams. Therefore, we have made very specific and individual learning progress. These will not be discussed here.

As a team, we were able to learn a lot from this project:

Strict separation frontend/backend team

This separation made us very efficient. Since there was little to clarify (mostly the requests, we created a document specifying those) only WhatsApp and a few meetings were needed to exchange information between the teams.

Through this specialization, each team became absolute specialists in its field. Towards the end of the project, however, some disadvantages became apparent:

- The problems became more and more specific and complicated. WhatsApp and meetings were no longer enough. First we created text documents (e.g. todo.txt). This was not enough either. Finally we used the Github function "Project".
- Unfortunately we could not learn anything from the other team. For example, the backend team now has no idea about Angular and Bootstrap.
- Both teams had enough work at first. With time, however, the backend team's work became increasingly easier. Therefore we had to make adjustments to the teams towards the end so that the workload could be balanced better.

Orientation in a unfamiliar environment

We had lectures on OOP and Java at the university. The whole web development was a completely new area for us. Also the work with many tools and rather new technology was demanding. As an example: It plays a major role whether you use Angular 1, 2 or 6. A solution on the Internet for an older version might not be useful anymore.

Finally it worked and we can deliver a finished product. It was interesting and fun. Furthermore the learning outcome takes away some of the insecurity about future projects in other contexts.