Meeting 1 (Week 2) - meeting notes 18/11/2021

Announcements

- Agenda should be uploaded on GitLab at least 24h before the meeting, we can still change it after uploading, but make sure that something is uploaded. Any format is fine.
- Notes should be taken during each meeting (we can have 2 people taking notes), it doesn't matter which format we use. Deadline for uploading notes: Sunday
- For all the other documents, they have to be in LaTex, other formats will not be accepted.
- If we have to do some graphs or sth then you don't have to use LaTex - use Lucidchart
- The meetings will start with a stand up meeting when each member stands up and says what they did last week and (if we are fast) also what they will do next week

Requirements

- Make sure that you are explicit as much as possible, so for example specify that a user can authenticate as a user/secretary/admin
- WE SHOULD REWRITE THE REQUIREMENTS AS USER STORIES CHANGE THE FORMAT. (a booking should have a limit <-not a user story)

Functional

- Security with password: just store encrypted password in the database, look into spring security, maybe there is a way to do it automatically.
- We can add as a could-have max time for a booking
- Don't care about the time zones - assume that everything is in the UTC
- SPLIT THE USER STORIES BETWEEN MUST/SHOULD/COULD MORE EVENLY, ALSO THINK OF MORE REQUIREMENTS, BECAUSE WE ARE MISSING SOME. Must - the minimum working product, could - nice to have things, won't - things that would limit the application
- Putting meetings in all participants' schedules - it's a nice could-have. As a must: put only in the person who creates the booking schedule
- Put editing in the backlog
- Should haves: scheduling in advance

Non-functional

- For the modular approach specify the microservices

Q&A

- Secretary verification: example: there are multiple profs that all have one secretary. Just put it in the db, by for example as a user role
- Admin vs secretary: admin can delete things and secretary cannot
- Admin can be one: one username and one password
- We can do this: when you run the app, it just adds some example entities to the database. Because our application is only a demo
- Splitting microservices: have multiple gradle projects <- different folder for each microservice. You can make separate git repositories or just separate branches (there won't be any conflicts, because they are completely independent)
- Look up Spring Security, we probably don't need a microservice for authentication

Database

- We should specify how we want to connect the microservices
- Change isEmployee to a user role
- Ana will ask about secretaries
- Admin table, Luuk's idea - ok, but make sure to specify it somewhere what is it for
- We can add the participants for a booking later
- Spring is better with having list of Users instead of list of userIDs
- If you didn't work on the backend in OOPP make sure to look into Spring and how it works
- Equipment should be a list of strings not a string
- Deadline for sending the backlog to Ana - Sunday 23:59

For next week

- Entire db designed, official deadline is next sunday
- Start coding, but we need issues for that and for the issues we need the final backlog, so hurry up with the final backlog
- You can put the user stories as a description of an issue
- Add roles to the Code of Conduct