Sprint Retrospective, Iteration #2

User story #	Task #	Assigned to	Estimated effort	Real effort	Done	Notes
Luca Becheanu	Implement jwt in identity service (generate token, grant authorities for requests)	Luca Becheanu	Large	Large	Yes	Encountered database and passwordEncoder issues as described in main problems.
	Fix error messages in requests, log errors	Luca Becheanu	Small	Small	Yes	Errors given by exceptions were not specific enough, now the user has a more in-depth answer of what went wrong.
Matthijs de Goede	Create JUnit tests for the Calendar service	Alexandru Bobe and Matthijs de Goede	Large	Large	Yes, but not for the refactored edition	We encountered that we had to instantiate a lot of objects before we could even start testing. Hence, we came up with a lot of helper methods to instantiate certain objects and with array fields to store them efficiently.
	Implement real entities for the Calendar service	Matthijs de Goede	Medium	Medium	Yes	I decided to get rid of the RequestedLecture and ScheduledLecture entities and just use the Lecture entity from the database for both versions of the lecture.
	Connecting the Calendar service to the database	Alexandru Bobe and Matthijs de Goede	Medium	Large	Yes, but it needs to be tested properly	We ran into some issues with Spring annotations which took a lot of time to fix. But now we have repositories to do the work efficiently and we directly store

						Lectures in the database, rather than passing them back to the calling method.
	Solving PMD issues for the Calendar Service	Matthijs de Goede	Medium	Large	Yes	Our branch had a lot of PMD errors, and we had to check for checkstyle and write javadocs. This turned out to be quite time consuming.
Shared						
Timen Zandbergen	Implement test for identity service	Timen Zandbergen	Medium	Large	partially	I encountered problems with spring not finding the correct beans, not everything is tested yet
	Add example code for using JWT	Timen Zandbergen	Small	Small	partially	
	Add roles to the JWT token	Timen Zandbergen	Small	Small	Done	
Alexandru Bobe	Create JUnit tests for the Calendar service	Alexandru Bobe and Matthijs de Goede	Large	Large	Yes, but not for the refactored edition	We encountered that we had to instantiate a lot of objects before we could even start testing. Hence, we came up with a lot of helper methods to instantiate certain objects and with array fields to store them efficiently.
	Create real-life sized test	Alexandru Bobe	Medium	Medium	Yes	I randomly generated instances for all the classes needed.

	Connecting the Calendar service to the database	Alexandru Bobe and Matthijs de Goede	Medium	Large	Yes, but it needs to be tested properly	We ran into some issues with Spring annotations which took a lot of time to fix. But now we have repositories to do the work efficiently and we directly store Lectures in the database, rather than passing them back to the calling method.
Can Parlar	Refactoring and updating the subproject architecture and databases	Can Parlar	Small	Medium	Done	changed names of microservices, connected them to databases, and made it ready for starting on implementing a common architecture.
	Implementing rooms and restrictions microservices	Can Parlar	Large	Medium	Done	Started implementing the rooms and restrictions services. Finished entities, controllers, api definitions, repositories and tests. Still going to work in more tests and communication
Merdan Durmus	Implementing Course Management Service	Merdan Durmus	Large	Medium	Not Done	Implementation of last Spring was not correct since it was vulnerable to SQL attacks, also we now made use of Spring Repositories. Testing still has to be done.

Project: Corona-proof room scheduling
Group: OP29-SEM57

Main problems encountered

Problem 1 - Inability to schedule a meeting

Description

- At some point, we really needed to schedule a meeting to update each other about the progress. However everyone was also busy preparing for the upcoming midterm, which made it hard to schedule such a meeting. As different people worked on the project at different times of the day.

Reaction

- We made a quick call on discord and updated each other using the chat.

Problem 2 - Being unconnected

Description

- Because the services weren't connected yet, we couldn't really test the created API endpoints yet. We solved this issue by

Reaction

- creating mocks and we will now focus on the communication and testing in general.

Problem 3 - There is no PasswordEncoder mapped for the id "null"

Description

- Passwords must be encrypted before being added to the database, therefore specifying a password encoding algorithm is needed. Unfortunately, it was still giving the aforementioned error.

Reaction

- After multiple searches, it was also needed to apply the new password storage format to the method in User service. When providing the user to spring security to validate if the netid and password are valid or not, the password needs to have the encryption type attached in front of the string. (“{bcrypt}”+user.getPassword())

Adjustments for the next sprint plan

- Make sure that the workload is more evenly balanced
- More consistent branch naming, based on branches for issues
- Upload (test) reports in separate folders
- Shift from focus on the algorithm to a focus on the communication between microservices