Sprint Retrospective: Week 4

User Story	**Task**	**Assig ned**	**Estimated Effort (in h)**	**Actual Effort (in h)**	**Done (yes/no)**	**Notes**
Authenticatio n for different users	Create User microservice	Mircea	4	5	Yes	Most of the User microservice was just the authentication from the template so there were not a lot of things to add. However, I spent a lot of time understanding how security works, how communicatio n and the structure of a microservice works.
There should be a field in User to designate the faculty/ies a user is assigned to	Add Faculties to User	Ilia	3.5	7	Yes	Harder than it seemed, I had some issues deciding on how to implement a user to be assigned to multiple faculties so I implemented only a single faculty; Filip has helped

						me with the multiple faculties functionality eventually.
Microservices should be able to run	Create Jobs & Schedule microservice	Axel	3	3	Yes	
User should be able to create/delete/get jobs	Create Job Class, Database and Endpoints to create/delete/get jobs.	Axel	3	8	Yes	Initial problems connecting and saving to the database.
The resource nodes need to be stored in the cluster database	Create Cluster Database & Cluster Class	Timur	3	5	Yes	Understanding how the jpa repository works was hard.
Job microservice must be able to request a job to be scheduled	Create Schedule Database & Schedule Class	Piotr	3	3	Yes	
After authenticating role should be accessible in the JWT token so other microservices will be able	Add roles to JWT token	Filip	3	8	Yes	Way more difficult than expected.

to use this information						
Visualize issue assignment	Configure Gitlab Issues Board	Axel	1	1	Yes	
Job should be able to be removed	Tests for removeJob method in Scheduler	Ilia	1	1	Yes	
Decouple the logic from the actual endpoints	Refactor Jobs microservice endpoints and added	Mircea	4	4	Yes	I took some inspiration from the example microservice that uses a @Service to implement the actual logic of the endpoints.

Main Problems Encountered

Problem 1

Description: Difficulties to connect to the H2 Database with IntelliJ, in order to visualize the Database.

Reaction: Research on the internet, then discussing in group, then contacting our TA Timur Oberhuber, then talking with the Head TA.

Problem 2

Description: Difficulty to add a Role to the JWT token.

Reaction: Research on the internet, trying to understand how Spring Security works in the Authentication Microservice. Discuss issues within the group.

Adjustments for the next Sprint Plan:

- improve issue assigning so that no task overlap with others
- be more strict with approval rules and review merge requests

Sprint Retrospective: Week 5

User Story	**Task**	**Assigned**	**Estimated Effort (in h)**	**Actual Effort (in h)**	**Done (yes/no)**	**Notes**
User can be assigned to multiple faculties	Add multiple faculties for one user	Filip	2	2	yes	Needed to come up with a way to store multiple faculties but once I got it, it was easy.
Classes like Jobs should be accessible in several microservices	Create commons module	Axel, Mircea, Piotr	2	5	yes	It was a bit difficult to figure out the right gradle config.
Jobs need to know if they are scheduled	Create Endpoint for the Scheduler to update the JobStatus	Axel	2	2	no	
Check if an user is authorized to make such request	Add authorization for microservices	Filip	2	2	Yes	
Employees should be able to see all free resources for the next day	get all free resources (employee) for the next day	Filip	2	6	Yes	Got stuck with project build(Problem 3) so I could not test it.

Methods should work	tests for job MS and overall jUnit testing	Filip	3	3	Yes	
Jobs should be able to be scheduled	Scheduler should have logic to schedule jobs	Piotr	2	5	Yes	Figuring out annotations for mock testing and trying different ways of making requests took much longer than expected.
Jobs should be able to be unscheduled	Scheduler should have logic to unschedule jobs	Piotr	1	1	Yes	
User is able to delete nodes	Cluster should have logic to delete the node. And when that happens, notify the scheduler of the change. When requesting for resources don't include the deleted nodes	Timur	5	5	Yes	
Improve code	Refactor and add tests for JobService	Mircea	4	4	Yes	

Add DTO	Added DTOs to the Jobs Microservice	Mircea	2	2	Yes	Used the @Data annotation
Methods should work	Test the removeJob method in Scheduler	Ilia	2	2	Yes	
Methods should work	Added tests in Commons and Authentication	Ilia	2	2	Yes	

Main Problems Encountered

Problem 1

Description: Create commons folder for classes that are used in multiple Microservices. The Job entity from the commons folder could not be stored in the JobRepository from the Job Microservice.

Reaction: Try to understand the error statements and look for similar solved problems on the internet. Consult the group for help.

Problem 2

Description: Trying to run the project once it stopped working with gradle bootRun in inteliJ. The error showed that the commons folder is not seen in the microservices, however, every other group mate was able to run it.

Reaction: Gradle name-of-ms:bootRun manually typed in the terminal. Somehow this runs it from the parent directory where is commons folder also located. This finally worked.

Adjustments for the next Sprint Plan:

- improve communication and collaboration so that less misunderstandings occur
- set clear goals and milestones for the next sprint

Sprint Retrospective: Week 6

User Story	Task	Assigned	Estimated Effort (in h)	Actual Effort (in h)	Done (yes/no)	Notes
Admin has access to all free resources	get all free resources for all the faculties and see out of how many it is(admin)	Filip	3	2	Yes	
Improve code	refactor and rework some users MS and added get all faculties in db endpoint	filip	2	2	Yes	
Admin can change the faculty of users	allow admin to change faculty of some user	filip	2	2	Yes	
Methods should work	Test commons	Axel	3	3	Yes	
User must be able to see if their Jobs are scheduled	Notify User if job is scheduled	Axel	3	3	Yes	
Approved jobs can be scheduled	Create a scheduleJob request to the scheduler	Axel	3	4	Yes	

Automatic approval of Jobs	Jobs will be automatically approved 6h before a day starts	Axel	5	8	yes	Difficulties to make the integration test but worked at the end.
Check resource requirement	Check if the CPU needed for a job is not smaller than its GPU or memory requirement.	Piotr	2	2	Yes	
Update scheduler in case resources change	The Scheduler must have logic for updating the schedule in case the amount of resources changes.	Piotr	3	3	Yes	
Cluster can receive and send DTO's	The cluster must be able to receive and send DTO to other microservice when needed.	Timur	5	5	Yes	Hard to fix a bug that gives no errors.
Authenticati on	add faculty to jwt token + integration tests in clusters	Filip	2	5	Yes	problem with nested requests. I had to add faculty to the jwt token
A faculty account	Implement and test	Mircea	10	20	Yes	Before starting to

can approve or reject jobs.	Chain of Responsibility design pattern					implement I had to research about how the design pattern should be coded optimally. Moreover, the testing was hard and time consuming as it was dependent on 2 other microservices which had some endpoints that had a bad design for what Chain of Responsibility needed.
Methods should work	Added tests for every microservice. Tested most of the classes.	Ilia	4	4	Yes	Quite a lot of repetitive work.
Job approval rules	Implement that 5 minutes before the start of the day, no requests are accepted	Ilia	2	2	Yes	
	Part 2 report chain of	Mircea	2	2	Yes	

	responsibility design pattern					
Scheduler makes use of different strategies	Implement Strategy design pattern	Piotr	4	5	Yes	It took quite some time to refactor the tests.
	Part 2 report: strategy design pattern	Piotr	1	1.5	Yes	

Main Problems Encountered

Problem 1

Description: Working on several issues that depend on each other in a group.

Reaction: Talk very openly, regularly and directly in what is done when, from whom and how.

Problem 2

Description: RequestBody doesn't parse JSON correctly if the getMethod is named differently to the attribute it gets.

Reaction: Make GetMethod and attribute consistently have the same name.

Adjustments for the next Sprint Plan:

- No sprints anymore