# Sprint Report Back End

10015, 10016, 10054

Spring 2025

## Table of Contents


**1 Sprint 1 - Week 6-7** **2**
- 1.1 Sprint goal . . . 2
- 1.2 Work and assignments . . . 2
- 1.3 Accomplishments . . . 3

**2 Sprint 2 - Week 8-9** **3**
- 2.1 Sprint goal . . . 3
- 2.2 Work and assignments . . . 3
- 2.3 Accomplishments . . . 3

**3 Sprint 3 - Week 10-11** **4**
- 3.1 Sprint goal . . . 5
- 3.2 Work and assignments . . . 5
- 3.3 Accomplishments . . . 5

**4 Sprint 4 - Week 12-13** **5**
- 4.1 Sprint goal . . . 6
- 4.2 Work and assignments . . . 6
- 4.3 Accomplishments . . . 6

**5 Sprint 5 - Week 15-17** **7**
- 5.1 Sprint goal . . . 8
- 5.2 Work and assignments . . . 8
- 5.3 Accomplishments . . . 8

**6 Sprint 6 - Week 18-19** **9**
- 6.1 Sprint goal . . . 9
- 6.2 Work and assignments . . . 9
- 6.3 Accomplishments . . . 9

**7 Sprint 7 - Week 20** **10**
- 7.1 Sprint goal . . . 11
- 7.2 Work and assignments . . . 11
- 7.3 Accomplishments . . . 11

**8 Sprint 8 - Week 21** **12**
- 8.1 Sprint goal . . . 12
- 8.2 Work and assignments . . . 12
- 8.3 Accomplishments . . . 13

# 1 Sprint 1 - Week 6-7

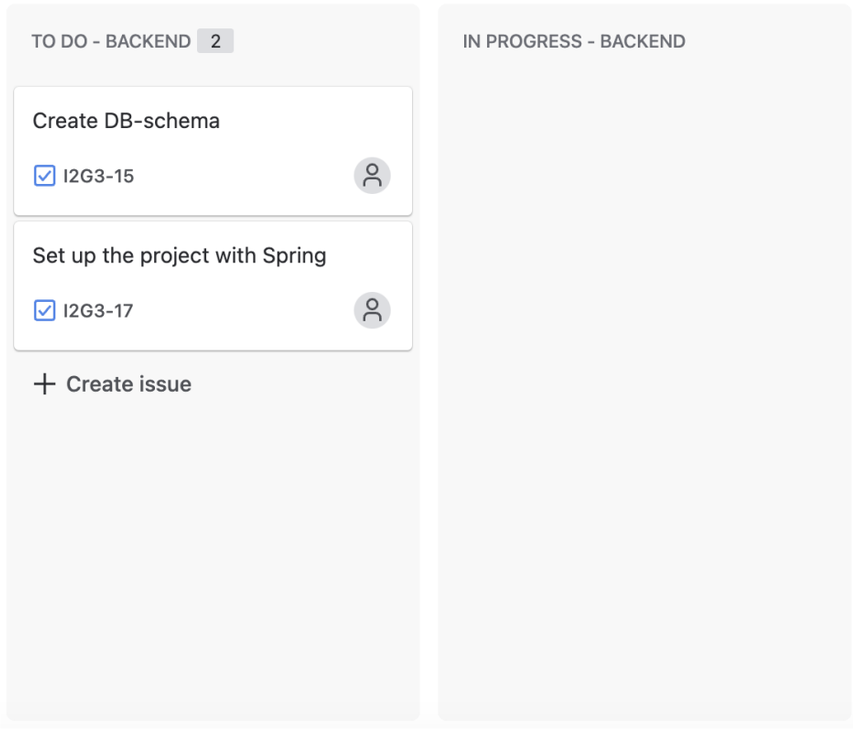


Figure 1: Sprint 1

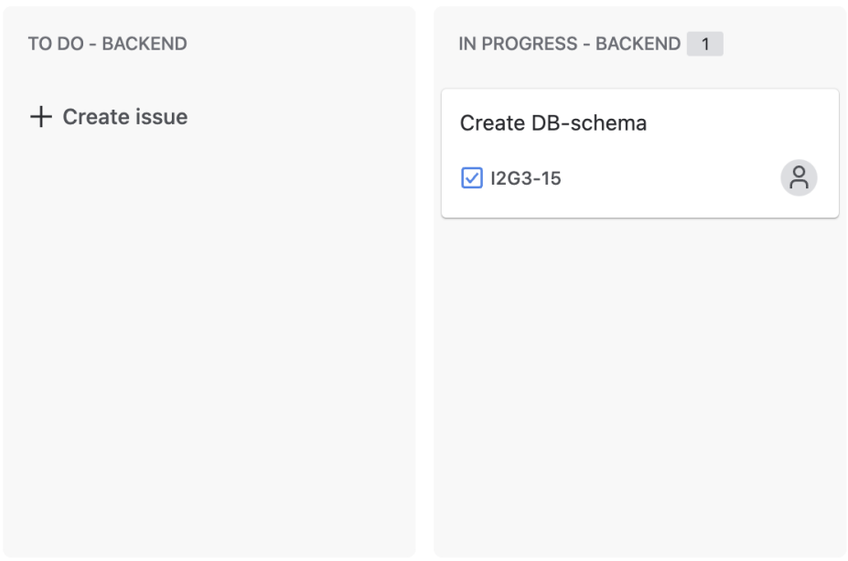


Figure 2: Sprint 1 - week 2

## 1.1 Sprint goal

The goal for Sprint 1 is to create a DB-schema for how the database should be. This will help us and be a guideline for how to implement this later on. Sprint 1's goal is to make the foundation for the rest of the project. Another goal for the sprint is to set up the project with Spring Boot.

## 1.2 Work and assignments

10015: Helped with DB-schema.
10016: Backend: Contributed to creating DB-schema.
10054: Created a new Spring project and started in DB-schema.

### 1.3 Accomplishments

In Sprint 1, we focused on laying the technical foundation for the backend. The Spring Boot project was successfully initialized, providing the base structure for our backend development moving forward.

We also started work on designing the database schema (ER diagram), identifying the core entities, attributes, and relationships required for the application. This schema will guide future implementation of entities, repositories, and data logic.

After reviewing the schema with the teacher, we received constructive feedback regarding areas for improvement - such as clarifying relationships and refining certain field types. These adjustments have been noted and will be addressed in Sprint 2.

Overall, the sprint achieved its main objective of setting up the backend environment and beginning the design of a well-structured and scalable database.

## 2 Sprint 2 - Week 8-9

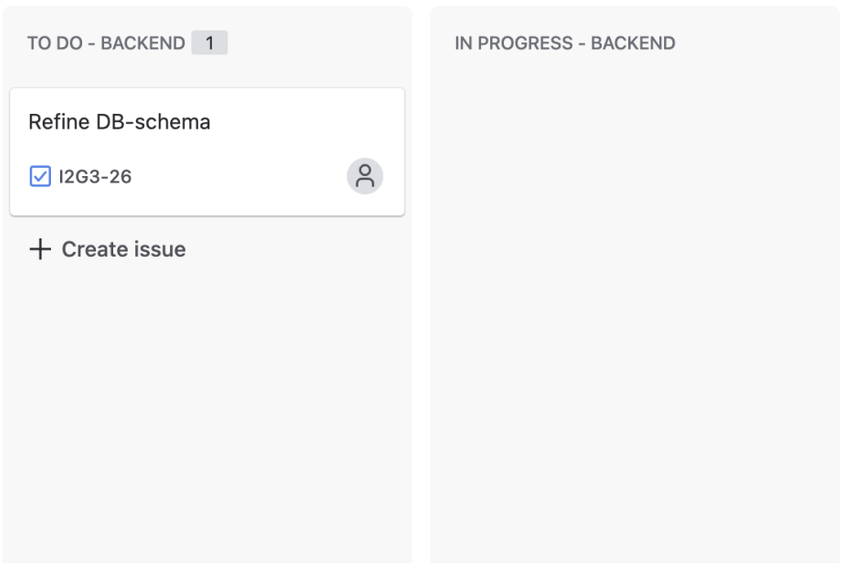


Figure 3: Sprint 2

### 2.1 Sprint goal

The goal for Sprint 2 is to refine the DB-schema in order to be ready for the next phase of the project.

### 2.2 Work and assignments

Everyone helped with the refinement of the DB-schema, where 10016 had the lead.

### 2.3 Accomplishments

Completed the DB-schema.

# 3 Sprint 3 - Week 10-11

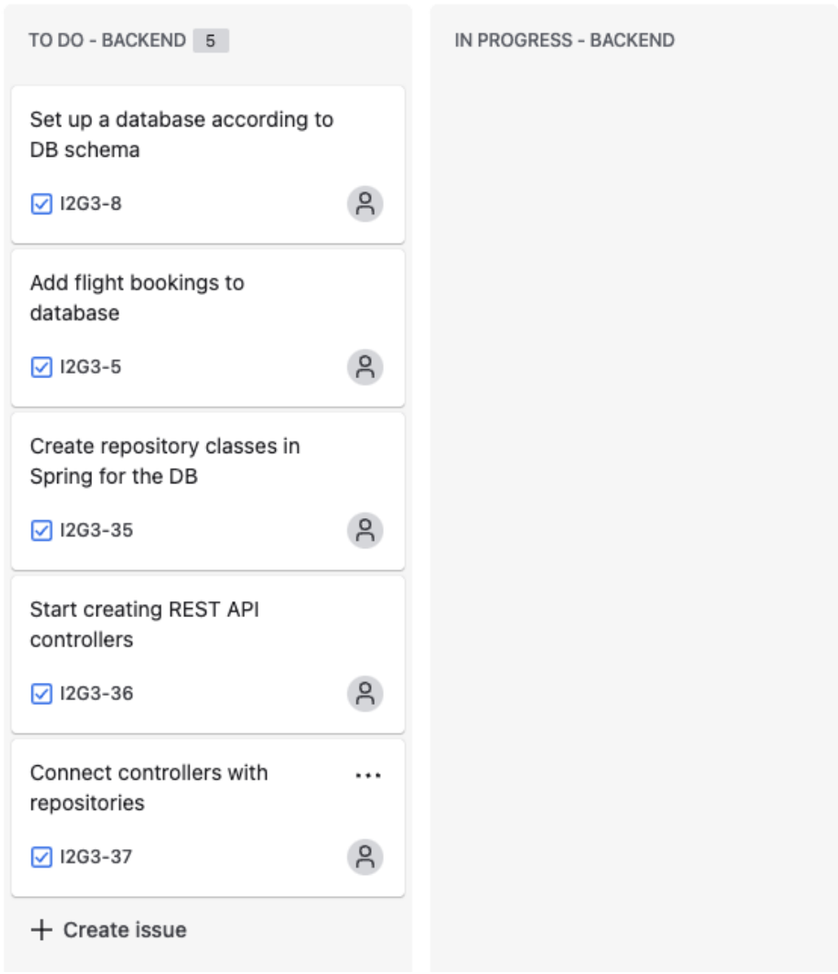


Figure 4: Sprint 3

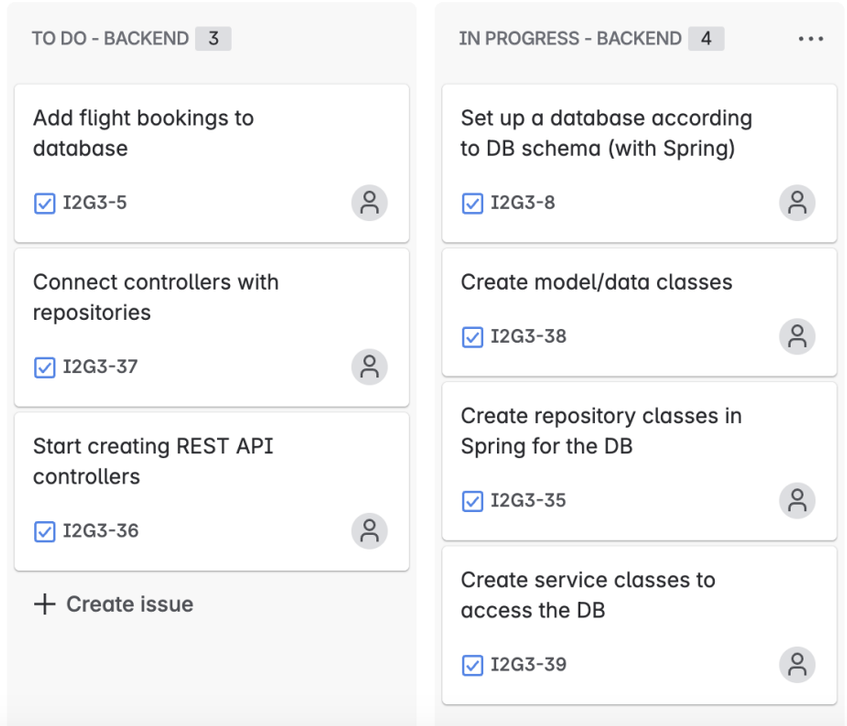


Figure 5: Sprint 3 - week 2

### 3.1 Sprint goal

The goal for Sprint 3 is to start setting up the database according to the DB-schema and start filling it with data. We will also start writing the necessary code to create REST API controllers, repositories, as well as connecting the controllers and repositories together.

### 3.2 Work and assignments

10015: Created Feedback entity, repository, and service according to the DB Schema.
10016: Created Flight, Airline, and Airport entity, repository, and service according to the DB Schema.
10054: Added dependencies for JPA and updated application.properties to connect to our database. Created User, Booking, Passenger, and Price entity, repository, and service according to the DB Schema.

### 3.3 Accomplishments

Although we realized early in the sprint that our initial goal was too ambitious, we still made substantial progress toward building the backend foundation. To manage the workload, we broke down larger tasks into smaller, more manageable issues, allowing for better tracking and progress throughout the sprint.

We successfully created all the entity, repository, and service classes defined in our database schema. These include key domain models such as User, Booking, Flight, Airline, Airport, Price and Feedback. This structured setup is essential for enabling REST API development in the next phase.

In addition, we configured the backend environment by adding JPA dependencies and updating the application.properties file to establish a working connection with our database.

While controller implementation will continue in the next sprint, the core data access layer is now in place, laying a strong foundation for building and testing the application logic.

## 4 Sprint 4 - Week 12-13

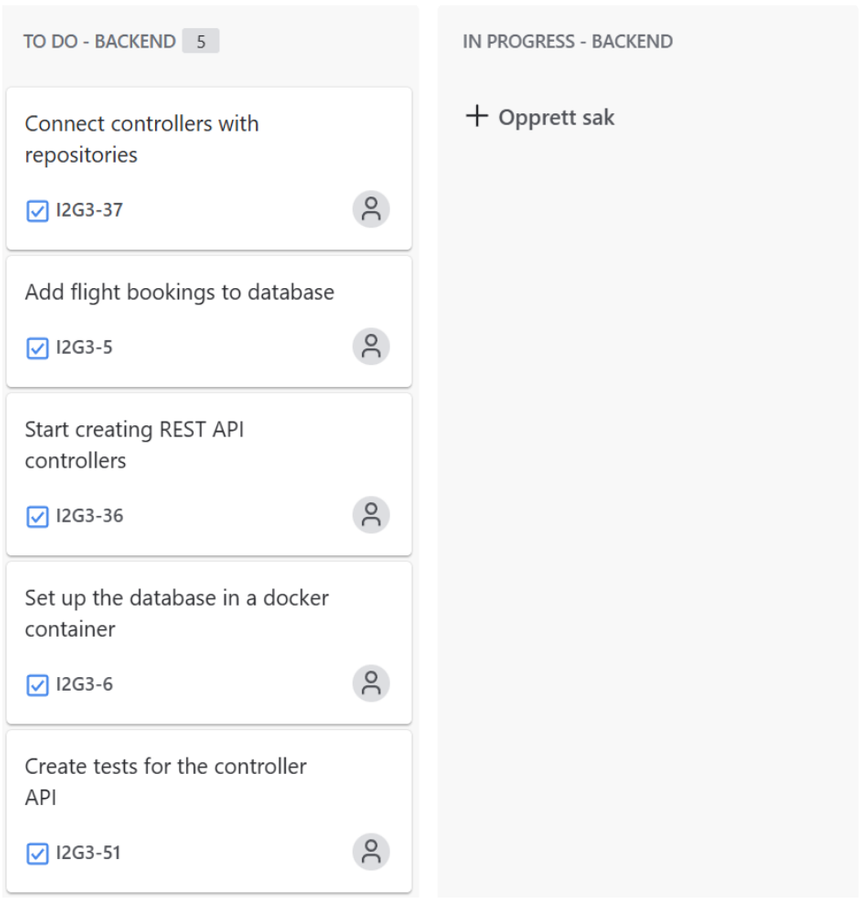


Figure 6: Sprint 4

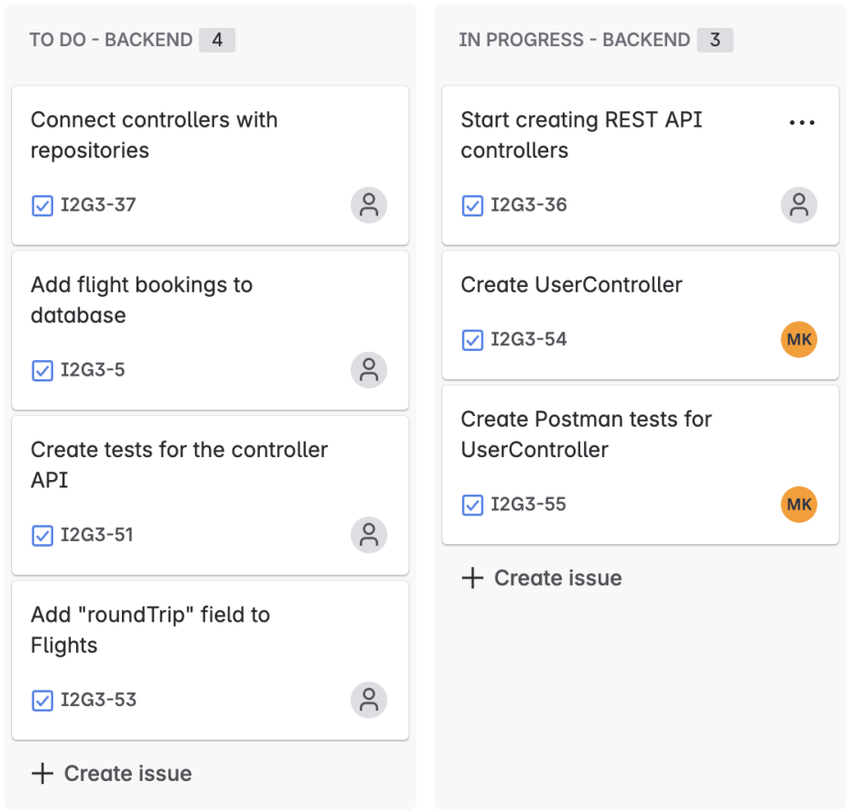


Figure 7: Sprint 4 - week 2

## 4.1 Sprint goal

The goal for this sprint is to set up the database in a Docker container and start creating the controllers necessary for doing REST API calls to the back end, as well as creating proper tests in Postman. If we have enough time at the end of the sprint, we will start populating the database with data.

## 4.2 Work and assignments

10015: Worked mainly on the front-end this sprint. Created the POSTMAN tests for 'Feedback'.
10016: Created postman tests for Flight, Booking, and Price. Adjusted the structure of Flight, Price and Booking entities to successfully pass all Postman tests. Created all necessary relations between all entities.
10054: Set up the database in a Docker container. Created controller classes and Postman tests for User, Airport, and Airline.

## 4.3 Accomplishments

During this print, we focused on backend development, specifically setting up the REST API layer, testing endpoints, and configuring the database for deployment. One of the first major tasks completed was setting up the MySQL database inside a docker container, providing a portable and consistent environment for development and testing.

We made a significant progress in creating controller classes for all major entities, including User, Airport, Airline, Flight, Booking, Price and Feedback. These controllers enable the core CRUD operations needed for frontend-backend communication. To ensure everything worked correctly, we also developed extensive POSTMAN tests for each controller, validating both the endpoints and their expected behaviors.

As development progressed, we recognized that our original sprint issues were too broad. To improve clarity and progress tracking, we refined our sprint plan by breaking down tasks into smaller, more specific issues. This gave us better overview of ongoing work and helped distribute tasks more efficiently.

Adjustments were also made to the entity relationships and structure - particularly for Flight, Booking and Price - to ensure the database schema aligned with our testing needs and business logic.

Given the workload and the need for thorough testing, we agreed to extend the spring by one week. Overall, the sprint laid a solid foundation for backend functionality and ensured that API endpoints were stable and ready for frontend integration.

# 5 Sprint 5 - Week 15-17

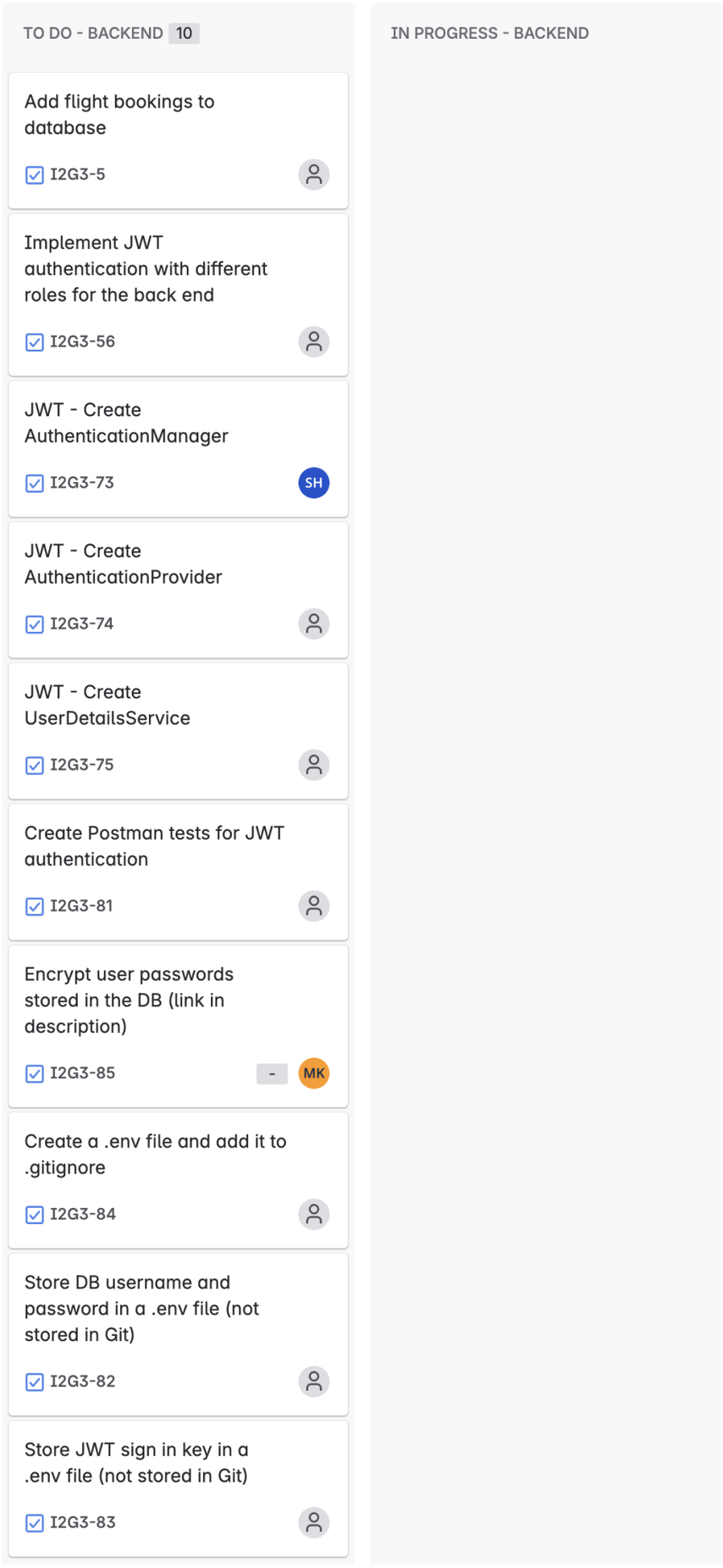


Figure 8: Sprint 5

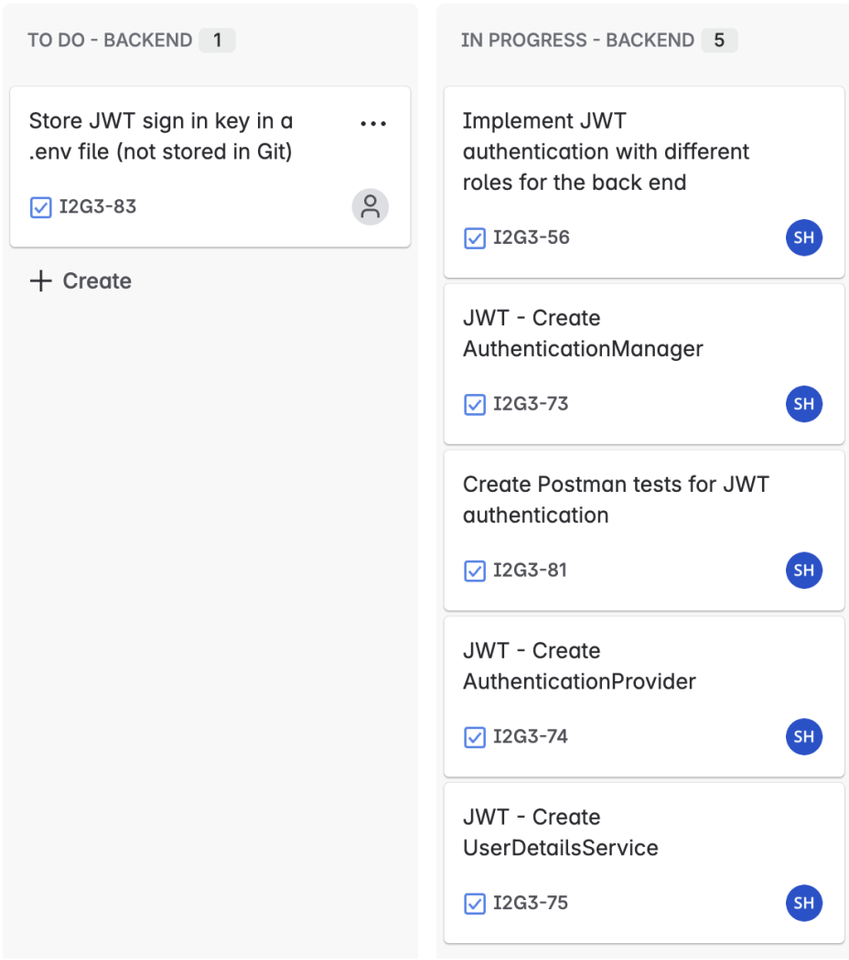


Figure 9: Sprint 5 - week 3

## 5.1 Sprint goal

The main goal for this sprint is to finally populate the database with data. Other goals is to add authentication to the back end using Jason Web Tokens (JWT) with tests in Postman, add .env file to store variables in a more secure environment and add encryption to user passwords stored in the database. Due to the easter break in the middle of this sprint, we added an extra week to the sprint.

## 5.2 Work and assignments

10015: Created the neccesary classes for JWT in backend. Fully implementing JWT for backend with POSTMAN tests. Created the user Chuck and Dave and gave them their appropiate roles. Also verifying that Chuck that is an ADMIN gets an ADMIN token, and Dave gets a regular user token. Also secured endpoints with roles, so certain endpoints are locked to admin or authorized users.
10016: Added RoundTripFlight data transfer object and updated FlightController. Implemented the search method to ensure that user can search flights based on origin, destination, date and trip type.
10054: Added encryption to user passwords stored in the database. Created tests in Postman that adds data in the database using HTTP POST. This includes ”dummy-data” for testing with front end. Added .env file to store DB password.

## 5.3 Accomplishments

This sprint marked a significant step toward production readiness, as we focused on security, data population, and improving backend infrastructure.

We successfully implemented JWT-based authentication in the backend. This included creating the necessary configuration classes, securing endpoints based on user roles and generating role-specific tokens - where Chuck (ADMIN) and Dave (USER) recieve the correct token types. The solution was verified with POSTMAN tests to ensure proper access control and token handling.

We also introduced password encryption, ensuring that user credentials are securely hashed before being stored in the database. In addition, we improved our backend setup by introducing a .env file to store

sensitive environment variables, such as database credentials, keeping them out of version control for better security.

To prepare the system for full-stack integration and testing, we created POSTMAN tests for HTTP POST operations that populate the database with structured dummy data - this included flights, airlines, users and related entitites. These test ensure consistent and reliable data for frontend development and validation.

On the feature side, we introduced a RoundTripFlightDTO and enhanced the FlightController to support search fincuionality based on origin, destination, travel, date, and round-trip filtering.

Despite the Easter break, we completed all critical goals by extending the sprint with one additional week. The progress made during this sprint laid a solid groundwork for secure, feature-rich interactions between the frontend and backend.

# 6 Sprint 6 - Week 18-19

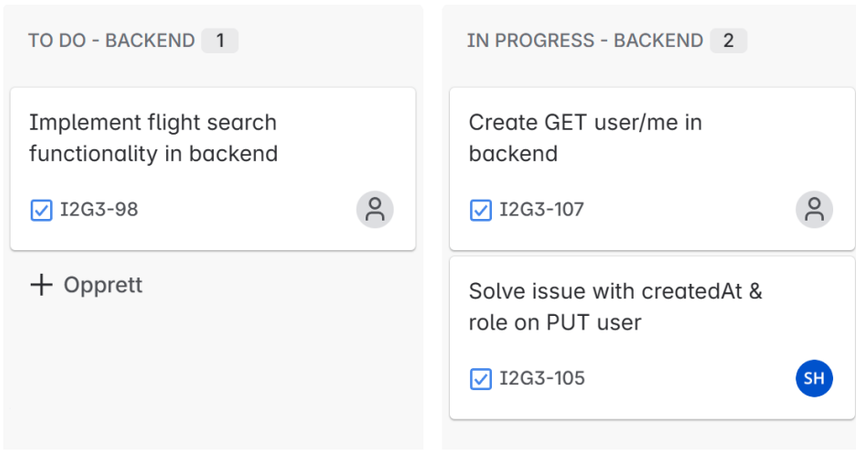


Figure 10: Sprint 6 - Week 18-19

No issues in the second week.

## 6.1 Sprint goal

The goal for this sprint is to complete the tasks that were not finalized in the last sprint. To create GET user/me in backend so a user can see their own profile and also edit own information. Another goal is to refactor some code to solve a problem with when a user is created to get the correct time for when the user is created.

## 6.2 Work and assignments

10015: Finished the JWT authentication. Refactored the JWT token to use userId instead of email. This was to solve the problem that arised when a user were to change their email. As it was, when a user changed their email they had to sign out and then sign in again with their new email address. Now, when JWT is using userId, the user can change their email, and their token is still valid. Created POSTMAN tests for this as well. Created methods and the possibility for users to see own profile and edit own information for certain fields.
10016: Adjusted the search result method to correctly filter flights based on given dates. Fixed issue with price fields appear as null in database after adding a price to flight. Created searchFlights method, added CORS config to security configuration to ensure that frontend can send requests to backend.
10054: Refactored some classes to return an object in the http response to be used in the front end application.

## 6.3 Accomplishments

This sprint was focused on finalizing key user-related features and improving backend robustness through thoughtful refactoring. We successfully completed all planned goals and even introduced additional improvements that enhanced backend flexibility and usability.

A major accomplishment was the update to our JWT authentication strategy. Previously, tokens were tied to a user's email, which caused issues when users updated their email adresses. This sprint, we refactored

the token to use the user ID instead, solving this problem and ensuring a seamless experience for users who update their credentials. This change was tested and verified using POSTMAN.

We also implemented the GET /users/me endpoint, allowing authenticated users to view and edit their own profile information - such as name, phone number, and country - while protecting more sensitive fields. This functionality improves user control and supports a more dynamic frontend experience.

In terms of features, we refined the flight search logic to correctly filter results based on departure and return dates, improving search accuracy. An issue related to price fields being stored as null after associating a price with a flight was also identified and resolved.

Additionally, CORS configuration was added to the Sprint Security setup, ensuring that requests from the frontend application are allowed and handled correctly across origins.

Lastly, we refactored several backend classes to return clean and structured DTO's (data objects) in HTTP responses, making it easier for the frontend to consume the API and improving maintainability overall.

All sprint goals were successfully met, and the system is now more secure, user-friendly, and better integrated with the frontend.

# 7 Sprint 7 - Week 20

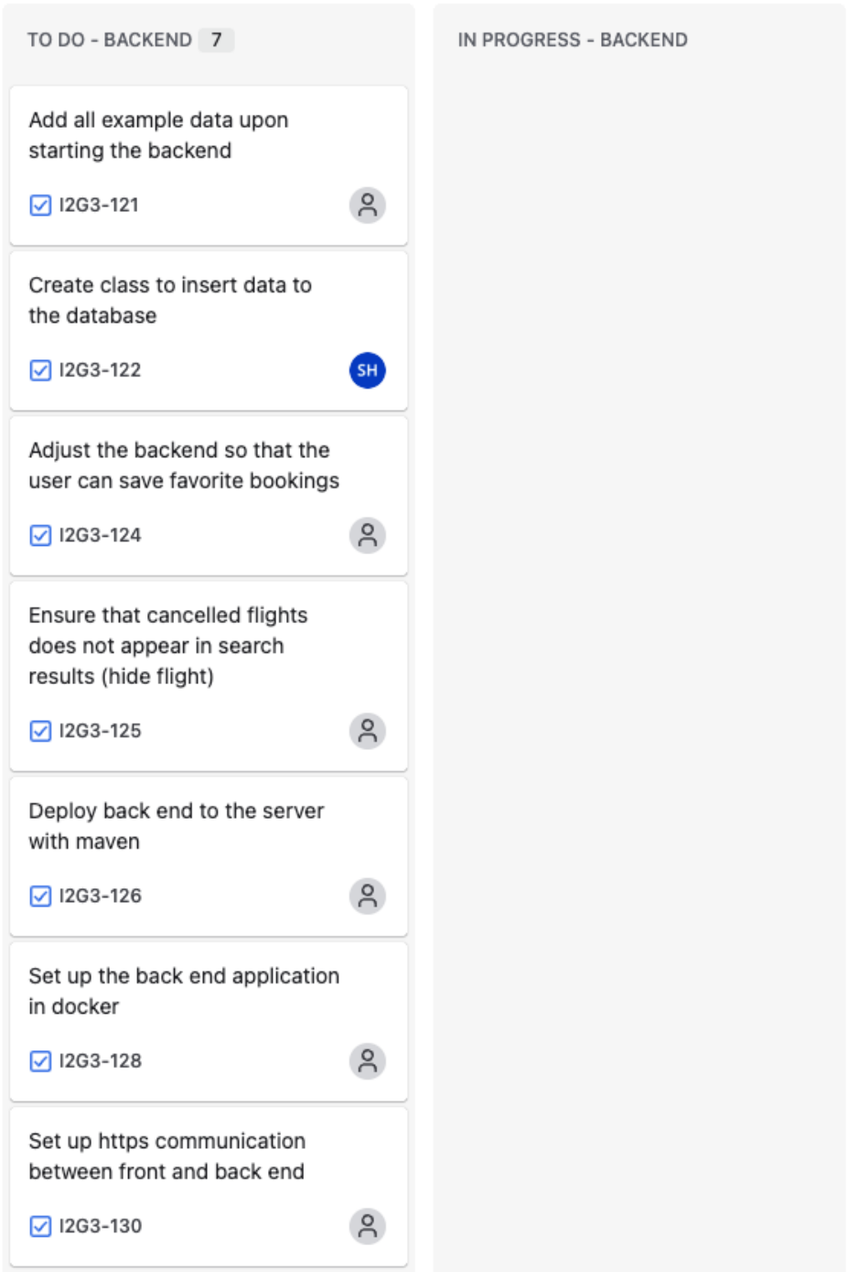


Figure 11: Sprint 7

## 7.1 Sprint goal

The goal for this sprint is to populate the database with data when the application starts, enable the user to save bookings, show/hide flights by setting the flight status to ”canceled”, ”delayed” or ”scheduled”, set up the application in docker, launch it to the server and use https to communicate with the front end.

## 7.2 Work and assignments

10015: Worked on populating the database with data when the application starts. Now when the application starts the database is populated with airlines, airports, flights with prices and four regular users Dave, John, Bob and Sarah and one admin user Chuck. Also Checkstyled the entire project and fixed some errors. Added feedback from each regular user.
10016: Fixed issue with canceled flights appearing in the search results. Extended the booking service with getBookingByUserId to ensure that a user can get a list of the saved booking/trips.
10054: Downloaded docker on the server and got Spring Boot and the database to run in Docker on the server. Tried to set up https communication betwen front and back end, but met some challenges regarding the records on the DNS server.

## 7.3 Accomplishments

In this sprint, we successfully completed all our core goals related to data population, feature development, deployment, and security setup.

We implemented automatic database population on application startup, ensuring that each time the backend launches, it preloads a realistic dataset. This includes predefined airlines, airports, flights (with associated prices), four regular users and one admin user. Each user was also associated with a sample feedback entry, enabling a better testing experience for the frontend.

The flight status feature was also completed. Flights can now be shown or hidden in search results depending on their status - Scheduled, Canceled, or Delayed. This was enforced in the backend logic, and an issue where canceled flights were still appearing in search results was identified and resolved.

On the booking side, we extended the backend with a getBookingByUserId method, enabling frontend to retrieve all saved trips for the currently authenticated user.

For deployment, we installed docker on the server and successfully containerized both the Spring Boot application and the MySQL database, running them together on the server.

Although we attempted to configure HTTPS using Certbot and Nginx as per the provided guide, we encountered a blocking issue: the provided domain name pointed to a private IPv4 address, causing Certbot’s public DNS validation to fail. As a result, we could not complete HTTPS setup through the intended method, and will explore alternative options if time allows.

Despite this issue, the sprint was highly productive, delivering essential features and pushing the application toward a deploayble and testable state.

# 8 Sprint 8 - Week 21

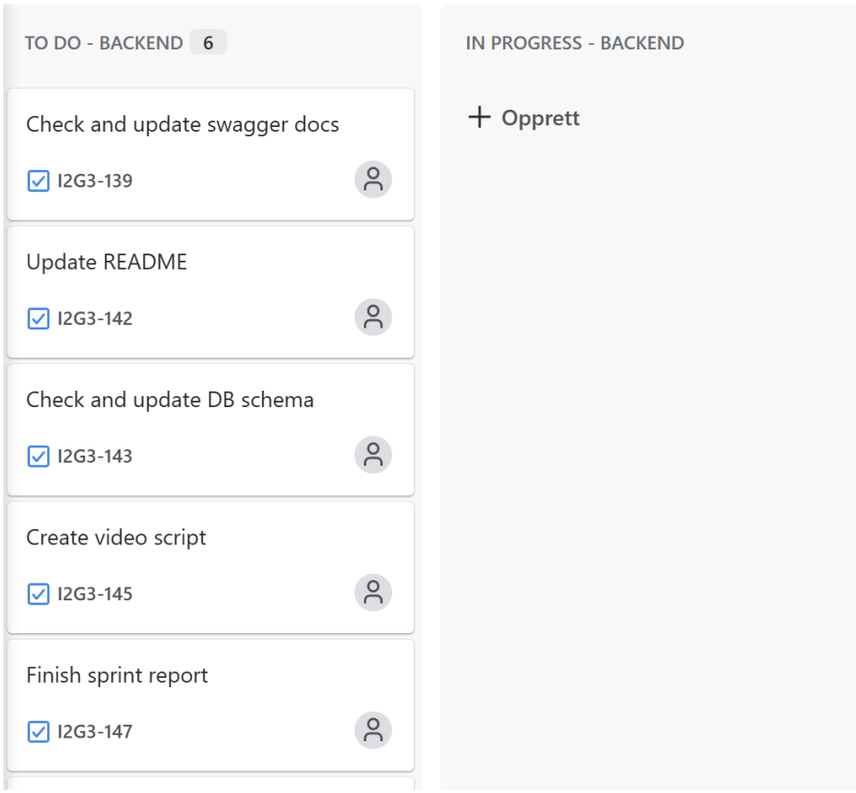


Figure 12: Sprint 8

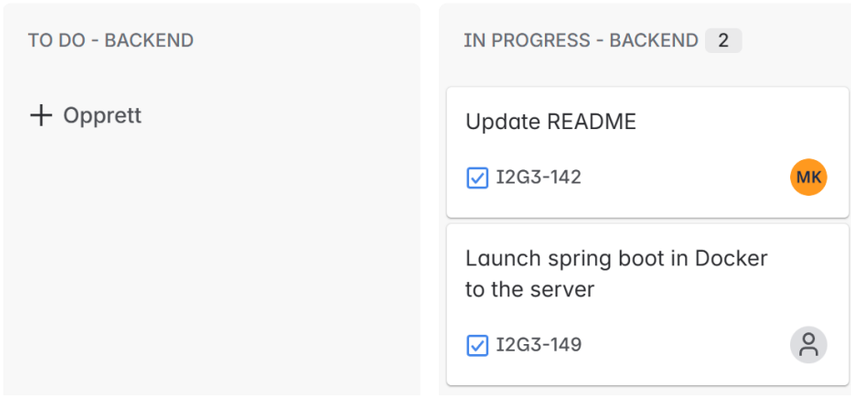


Figure 13: Sprint 8 - Midweek

## 8.1 Sprint goal

The goal for this sprint is to finalize the project, clean up the code, finish and update documentation for the final delivery, as well as produce a video-script for the video presentation of the project.

## 8.2 Work and assignments

10015: Added and updated some swagger doc. Check styled the project and fixed some minor issues with the check style. Worked mostly on documentation and updating / cleaning it up.
10016: Adjusted the Booking entity with an additional numberOfTravellers field. Added some additional data for better user experience. Updated DB-schema and postman tests.
10054: Added and updated some swagger docs, cleaned up some code, updated readme and launched the application on the server with docker.

Everyone worked on finishing the sprint report and writing the video script.

## 8.3 Accomplishments

In this final sprint, we successfully achieved all remaining goals necessary to complete and deliver the project. The focus was on cleaning up the codebase, finalizing documentation, and preparing presentation materials for submission.

We thoroghly updated and refined the Swagger documentation, ensuring that all API endpoints are clearly described with accurate request and response formats. The README file was revised to include final setup instructions, usage notes, and deployment details, making it easier for future developers or evaluators to understand and run the project.

From a technical perspective, we completed the deployment of the backend in a docker container and verified that the application was running correctly on the server. Additional clean-up tasks were completed, including check-style corrections and removal of redundant or outdated code to ensure high code quality for the final version.

Feature-wise, we made a final adjustment to the Booking entity adding a numberOfTravellers field to enhance the booking process. This required updates to the database schema and corresponding POSTMAN tests, which were completed as part of the spring.

As a team, we collaboratively worked on writing the video script and recorded the final presentation video to summarize our work across both frontend and backend.

By the end of the sprint, all planned goals were completed, and the project was ready for final submission, both techincally and in terms of ducomentation and presentation.