Exam project work

Tecnologie e applicazioni web, a.y. 2024/2025

This year’s project involves developing a web application, including a server with REST-style APIs and a single-page application (SPA) front-end, that allows users to search for flights from major airlines and purchase tickets.
The system must handle two types of users: airlines and passengers. Airlines can add routes operated by their aircraft. Each route connects a specific departure airport (city) to a destination and is served by one or more aircraft during specific periods. Each flight has associated costs depending on the ticket type (economy, business, first class). Additionally, airlines can view statistics on the number of passengers on their flights, total revenue, and the most in-demand routes.
Passengers can search for flights to travel from one location to another. A trip may include up to 1 intermediate stop, provided there is enough time for the transfer (at least 2 hours). The application should allow sorting the available alternatives by various parameters, such as cost, duration, and number of stops.
Once a preferred option is selected, passengers can proceed with ticket purchase by choosing their seat for each flight and selecting any extras, such as additional baggage or extra legroom.
Finally, the system must provide users with real-time seat availability for each flight. The number of available seats must be updated automatically as soon as a ticket is purchased.
Flight searches can be performed by any user, including anonymous ones. However, all other operations require authentication.

Architecture

The system must comprise:

- A REST-style **backend** web service, implemented in TypeScript or JavaScript, and running on Node.js. The backend must use Express.js for routing. For the database, you can choose among MongoDB (as seen during the course) **or any classic Relational database**.

- A web **front-end** implemented as a Single Page Application (SPA) using the Angular framework.

Note: Each component (Backend, Database, Front-end) must run in a separate Docker container.

Requirements

Your application must implement at least the following features:

1. User management
    1. Registration of new users;
    2. Registration of new arlines "by invitation". Airlines cannot register themselves but must be added by an admin user who sets a temporary password. Then, at the first login, the new airline must set its password and any relevant information;
    3. Deletion of existing users (admin only)

Notes:

- **Login is required for all the system functions available to airlines**.
- **The admin account is created programmatically the first time the backend is executed**

2. Routes management (Airlines only)
    1. Creation of new routes;
    2. Creation of new aircrafts;
    3. Creation of flights (aircraft flying a specific route at a specific time);
    4. Setting costs for each type of tickets;
    5. Vew statistics about the number of passengers on their flights, total revenue, and the most in-demand routes;
3. Flight search
    1. Any (unregistered) user can search for flight trips from two locations. Once a trip is selected, tickets can be purchased after login;
4. Tickets and seats
    1. Passengers can buy tickets for different flights. During the purchase, some extras can be selected such as additional baggage or extra legroom
    2. Passengers can select their seat during the purchase

You are free to implement additional features beyond those listed above. Implementing additional features does not guarantee the achievement of honors, but it contributes to a positive judgment.

Important:
When the backend starts, it must preload some test data (users, airlines, flights, etc.) into the database to simplify testing the application during the exam.

Submission

Projects must be submitted via Moodle on the following page:

https://moodle.unive.it/mod/assign/view.php?id=959985

Each member must submit the same file, named `<group_name>.zip`, if working in a group.

The package must contain:

- **The report of each student**, in PDF format, named `<student_firstname>_<student_surname>_<matriculation_number>.pdf`
- A `README.txt` file with instructions on how to run the entire application. For example, you can briefly summarize the shell commands to compile the sources, start the backend, frontend, etc.
- All the application source code, clearly divided into backend and frontend.

NOTES:

- Do not submit any external libraries. Please delete all the `node_modules` directories before submission.
- Students can form groups of up to 3 people freely. Moodle will not help or enforce group submission. Individual students can simply use their surname as their group name.

Report

A report describing the system architecture, the software components used, and how they contribute to achieving the required functionalities must be submitted with the application. The report is **strictly individual** and will serve as a basis for the oral discussion of the project.

The report must contain:

- A description of the system **architecture**, listing all the different components and their relationships (i.e. How they work together to implement the application requirements).
- A description of the **data model**, either the ER diagram if you've chosen a relational database or the list of collections and the documents inside each collection if using MongoDB.

- A precise description of the **REST APIs**. Such a description must include a list of endpoints, along with their parameters and the data exchanged. Provide examples in JSON format whenever possible.
- A description of how user **authentication** is managed, along with the associated workflow.
- A description of the Angular front end, with a concise list of the implemented **components, services,** and **routes**.
- Some examples, possibly with **screenshots**, showing the typical application workflow for each different user role.

Q&A

For any questions about the project work or the course in general, don't hesitate to contact the teacher at:
filippo.bergamasco@unive.it

I'll be happy to arrange a meeting if needed!

Additional information about the exam is available at:
https://moodle.unive.it/mod/page/view.php?id=913371

Prof. Filippo Bergamasco,
Tecnologie e Applicazioni Web, a.y. 2024/2025