CCAPDEV Term 1, AY 2025 – 2026

Machine Project – Project Specifications Document

Groupings: At most 4 members in a group

Online Airline Ticketing System

Project Overview

You are tasked to design and implement an **Online Airline Ticketing System**. The system should allow passengers to search for flights, book tickets, select optional packages (meals, seat selection, baggage), and manage reservations (modify/cancel).

The project is divided into **3 milestones**. Each milestone builds upon the previous one and introduces additional tools and requirements.

Milestone 1 — Client-Side UI Design (HTML, CSS, JavaScript, jQuery, Bootstrap)

Goal

Design the **front-end user interface** for the airline ticketing system. Focus only on **client-side development**.

Requirements

1. **Flight Search Page**
   - Form inputs for origin, destination, departure date, return date, passenger count.
   - Display list of available flights (dummy/static data).
   - Flight details include: airline, flight number, departure/arrival time, price.
2. **Reservation Form**
   - Passenger details: name, email, passport number.
   - Optional package selection:
     - Meal options (dropdown: standard, vegetarian, kosher, etc.)
     - Seat selection (interactive seat map UI using Bootstrap grid & jQuery click events).
     - Extra baggage (checkbox or number input).

   - Show a **summary panel** with total price calculation (JavaScript updates dynamically).
3. **Reservation List (My Bookings)**
   - Display booked reservations (dummy/static data).
   - Ability to select a booking and view details.
4. **Other Pages**
   - Relevant admin pages as mentioned in Milestone 2 (User Management and Flights, UI only)
5. **UI / UX Guidelines**
   - Use **Bootstrap** for layout and styling.
   - Use **jQuery** for interactivity (seat selection, dynamic price updates, form validation).
   - Ensure **responsive design** for desktop and mobile.
   - Provide a clear **navigation bar** with links: *Search Flights, Book Flight, My Reservations, Profile*.

Milestone 2 — Backend CRUD Implementation (Node.js, Express, Handlebars, MongoDB)

Goal

Implement **full-stack CRUD functionality** for all features. Connect the front-end to a backend system with persistent data storage.

Requirements

1. **User Management**
   - Register, login, and manage user profiles.
   - Users can edit their profile details.
   - Admin accounts can manage flights and reservations.
2. **Flights**
   - Admin can create, update, delete, and view flights.
   - Flights should include: flight number, origin, destination, schedule, aircraft type, seat capacity.
3. **Reservations**
   - Passengers can create bookings (store in MongoDB).
   - Modify booking details (change seat, meal, baggage).
   - Cancel reservations with status update.
4. **Optional Packages**
   - Store selected meal, seat, and baggage per passenger in MongoDB.
   - Allow editing/removing these packages after booking.
5. **Views**

  - Use **Handlebars** for rendering pages (search results, reservation details, booking form).
  - Show dynamic data from MongoDB (not static anymore).

6. **CRUD Coverage**
   - **Create**: Book a flight.
   - **Read**: View flights and reservations.
   - **Update**: Modify reservations, change profile, update seat/meal/baggage.
   - **Delete**: Cancel reservations, delete user accounts, remove flights (admin only).

Milestone 3 — Security, Testing, and RESTful Features

Goal

Enhance the system with **security, automated testing, and API-based online check-in functionality**.

Requirements

1. **Security**
   - Hash passwords before storing (using bcrypt).
   - Implement **session-based authentication** (only logged-in users can access booking features).
   - Protect admin routes from normal users.
   - Incorporate server logging of all user actions and system errors.
2. **Unit Testing**
   - Write **Jest tests** for:
     - User authentication (register/login).
     - Reservation creation and cancellation.
     - Flight creation (admin).
   - Ensure tests cover both valid and invalid scenarios.
3. **Online Check-in (RESTful API)**
   - Implement an API endpoint for online check-in:
     - `POST /api/checkin` with booking reference and passenger details.
     - Assign boarding pass number and mark passenger as *checked-in*.
   - The application front-end should allow passengers to check-in via a simple form (PNR + last name).
   - On successful check-in, show confirmation and seat assignment.
4. **System Hardening**
   - Input validation (server-side and client-side).
   - Prevent double seat booking (atomic updates in DB).
   - Use HTTPS in deployment (conceptual requirement for project).
5. **Proper Design and Coding**
   - Ensure to follow MVC and relevant Design Patterns where applicable.

Deliverables per Milestone

Milestone 1 (UI Only)

- Flight search form (dummy data).
- Seat map with selection.
- Booking form with packages.
- Summary page with dynamic price updates.
- My Reservations page (dummy data)
- Relevant Admin Pages.

Milestone 2 (Full CRUD)

- Connected backend with MongoDB.
- Flights, reservations, and users fully managed with CRUD.
- Dynamic Handlebars templates for all pages.
- Reservation modification and cancellation working with persistent data.

Milestone 3 (Security & API)

- Passwords securely hashed.
- Authentication and session management implemented.
- Unit tests with Jest covering main workflows.
- Online check-in feature available via REST API.
- Booking system hardened against invalid inputs.
- Proper coding practices
- System logging.

IMPORTANT: Note that for every instance of a create feature that adds data in the database of the web application, there should be corresponding features that will allow the corresponding user to read, update, and delete the data from the database. For example, if you have a feature that allows the user to create a post, the user should also be allowed to read/view the post, update/edit the post, and delete/remove the post.

WORKING WITH GROUPMATES

For this project, you are encouraged to work in groups of at most 4 members. Make sure that each member of the group has approximately the same amount of contribution for the project. Problems with groupmates must be discussed internally within the group, and if needed, with the lecturer.

USE OF AI

As this is common core course and as stated in the course syllabus, AI should not be used in any portion of the submission of this project.