Documentation Flights Management App

Gheorghe Cosmina

**10 Business Requirements:**

1. User Registration and Authentication. Users must register with their personal details (name, email, password, etc.) and log in getting a token. Admins can only be added directly from the database.
2. Flight Search Functionality.Users should be able to search for flights based on source city, destination city, and date. No parameter is required for the search.
3. Flight Booking System. Users must be able to book flights and select the number of baggages.
4. Flight final Price calculation. For each booking a final flight price is calculated by using an available coupon and adding the price for the baggages.
5. Coupon Generation. After every successful booking, a unique coupon should be generated and assigned to the user.
6. Admin Management of Flights. Admins should be able to add, edit, or delete flight details.
7. Aircraft Management. Admins should be able to add or update details about the aircraft (e.g., capacity, type, model).
8. Airport and City Management. Admins must be able to manage airports and their associated cities to support flight operations.
9. Booking History. Users should be able to view their booking histor.
10. Role-Based Access Control. Ensure separate functionality and permissions for users and admins. Regular users shouldn’t have access to admin-only operations.

**5 Main Features for MVP**

* User Registration, Authentication, and Role Management. Users can register and log in. Admins have separate access with elevated permissions. The provided token will be used for each request.
* Flight Search. A search feature to filter flights by source city, destination city, and date. Results should include details such as departure time, arrival time, departure airport, arrival airport, and available seats.
* Flight Booking with Coupon Issuance. Users can book flights. A coupon is automatically generated upon successful booking and linked to the user.
* Admin Management Panel. Admins can add, update, and delete flights and aircraft. Admins can also manage airports associated with flights.
* User Dashboard for Booking History. Users can view a list of past and current bookings.

**Mapping Objects to Business Requirements**

* User/Person: Manage authentication, registration, and booking actions. A Person can contain detailed data about a user or an employee (ex. Pilot).
* Flight: Core object for user searches and admin management.
* Aircraft: Associated with flights, managed by admins.
* Booking: Links users, flights, and coupons. Tracks user transactions.
* Coupon: Reward generated after a booking.
* City/Airport: Associated with flight operations, allowing search by location.

You can access the full API documentation by downloading the folder swagger\_html from GitHub and opening the index.html. I was unable to publish the Swagger documentation because the project kept getting deleted every time I tried to share it or re-enter the website.

Additionally, you can use the raw source code swagger\_code.yaml that generates the documentation by pasting it into https://editor.swagger.io/.

Methods

**Aircraft**

**createAircraft** post /api/v1/aircraft/

**deleteAircraft** delete /api/v1/aircraft/{id}

**getAircraftById** get /api/v1/aircraft/{id}

**getAircraftsByName** get /api/v1/aircraft/name/{name}

**getAllAircrafts** get /api/v1/aircraft/

**updateAircraft** put /api/v1/aircraft/{id}

**Airport**

**createAirport** post /api/v1/airport/

**deleteAirport** delete /api/v1/airport/{id}

**getAirportById** get /api/v1/airport/{id}

**getAirportsByCity** get /api/v1/airport/city/{name}

**getAllAirports** get /api/v1/airport/

**updateAirport** put /api/v1/airport/{id}

**Auth**

**loginUser** post /api/v1/auth/login

**registerUser** post /api/v1/auth/register

**Booking**

**createBooking** post /api/v1/bookings

**getAllBookings** get /api/v1/bookings

**Coupon**

**deleteCoupon** delete /api/v1/coupons/{id}

**getCouponsForUser** get /api/v1/coupons/user/{userId}

**Flight**

**addFlight** post /api/v1/flights

**deleteFlight** delete /api/v1/flights/{id}

**getAllFlights** get /api/v1/flights

**searchFlights** get /api/v1/flights/search

**updateFlight** put /api/v1/flights/{id}

**Person**

**createPerson** post /api/v1/person/

**deletePerson** delete /api/v1/person/{id}

**getAllPersons** get /api/v1/person/

**getPersonById** get /api/v1/person/{id}

**getPersonsByNationality** get /api/v1/person/nationality/{nationality}

**updatePerson** put /api/v1/person/{id}