**Full Stack Development with MERN**

**Project Documentation format**

**1. Introduction**

**Project** **Title**: FlightHub - Flight Booking Application

**Developed By**: Kavin N (Full Stack Developer)

The FlightHub – Flight Booking Application is a full-stack web platform that enables travelers to search, compare, and book domestic/international flights online. The system provides real-time seat availability, secure booking and payment, and a dedicated admin panel for managing flights and bookings.  
The FSD outlines the functional and technical behavior of the application and serves as a reference for developers, testers, and stakeholders.

**2. Project Overview**

1. **Purpose:** To build a convenient online flight booking system where users can browse, evaluate, and purchase flight tickets digitally without visiting physical counters. The platform focuses on usability, real-time availability, and reliable decision-making.
2. **Key Features:**

* Secure user registration and login
* Search and filter flights by origin, destination, date, airline & class
* Booking with passenger details and dummy payment
* View booking history and cancel reservation
* Admin dashboard for managing flights and bookings

**3. Architecture**

* **Frontend**: Developed using React with reusable components. Context API handles authentication and booking state. React Router ensures smooth navigation across pages.
* **Backend**: Developed using Node.js & Express.js with MVC structure. Routes handle incoming requests, controllers process logic, and models interact with the database.
* **Database**: MongoDB stores users, flights, and booking data using Mongoose schemas to define structure and relationships

**4. Setup Instructions**

• **Prerequisites:** Node.js, MongoDB, Git, VS Code.

• **Installation Overview:**

Clone the project repository

Install dependencies for both frontend and backend

Configure environment variables (e.g., MongoDB URI)

Start the development servers

**5. Folder Structure**

• **Client:** Consists of all React‑based UI features — pages, components, CSS files, assets, and context providers.

• **Server:** Includes routes, controllers, models, and middleware that handle backend logic and database interaction.

**6. Running the Application**

To run the project locally:

* **Frontend:** npm start in the client directory.
* **Backend:** npm start in the server directory.

**7. API Documentation**

• The backend exposes several REST APIs for user, book, wishlist, cart, and admin operations.

**Endpoint** **Method** **Purpose**

/api/auth/register POST Register a new user

/api/auth/login POST Authenticate user and return JWT token

/api/auth/me GET Get logged-in user profile

/api/auth/profile PUT Update user profile

**Flight** **Endpoints:**

**Endpoint** **Method** **Purpose**

/api/flights GET Fetch all available flights

/api/flights/:id GET Get selected flight details

/api/flights POST Add new flight (Admin only)

/api/flights/:id PUT Edit/update flight details (Admin only)

/api/flights/:id DELETE Delete a flight (Admin only)

**Booking Endpoints**

**Endpoint** **Method** **Purpose**

/api/bookings POST Create a new booking

/api/bookings/my-bookings GET View user’s booking history

/api/bookings/:id GET Get booking details

/api/bookings/:id PUT Update booking / Change status

/api/bookings/:id/cancel DELETE Cancel a booking

/api/bookings/:id/reschedule PUT Reschedule flight booking

**8. Authentication**

Authentication in FlightHub – Flight Booking Application is handled securely on the backend using **JWT** (**JSON** **Web** **Tokens**) to ensure safe access to user accounts and admin privileges.

* Login credentials are verified against the database before any access is granted.
* Upon successful login, a JWT token is generated and stored in the browser (localStorage) to authenticate future requests.
* Passwords are secured using bcryptjs hashing, ensuring they are never stored in plain text.
* Every protected API request is validated using authentication middleware, which checks for a valid JWT token before allowing access.

**Role-Based Access Control** (RBAC) is implemented:

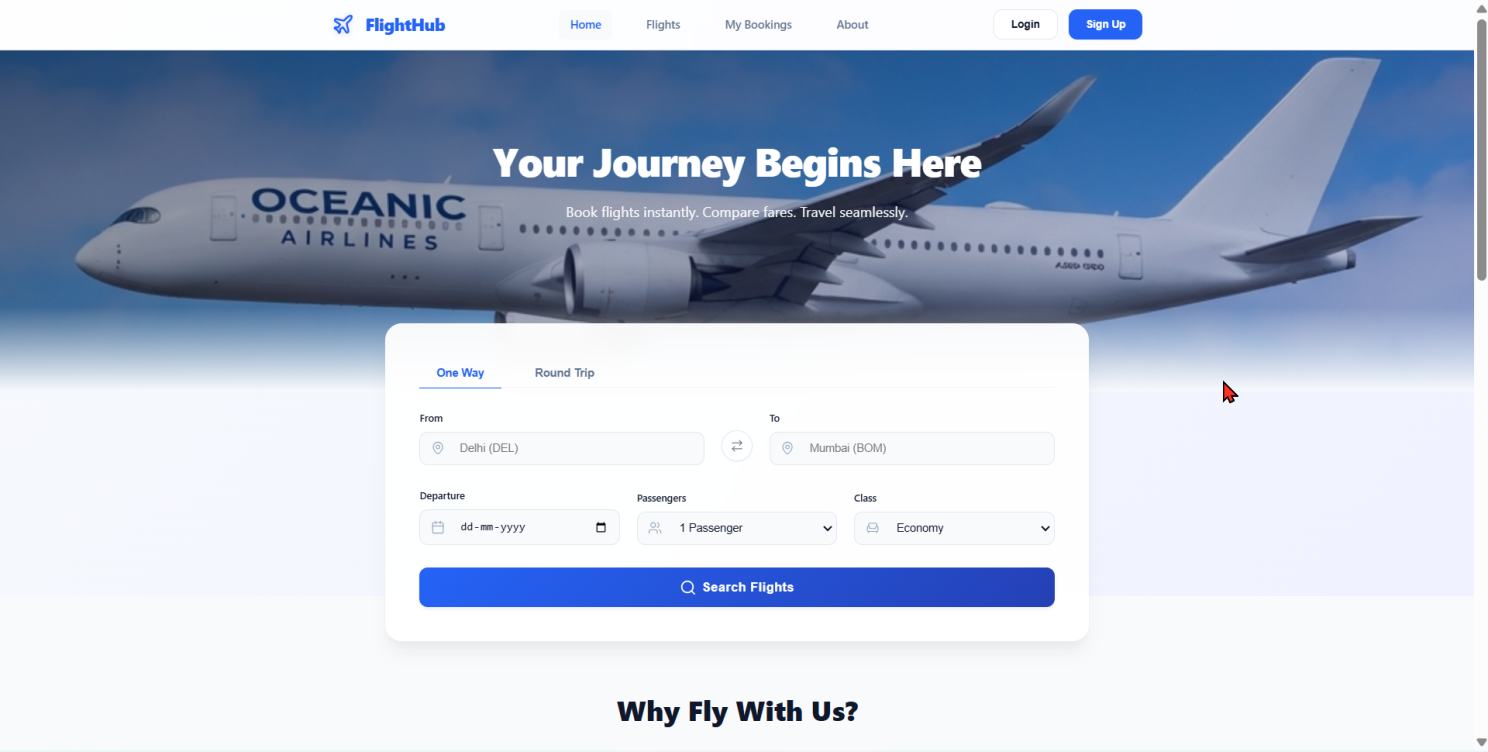
* Users (Travelers) can search flights, manage bookings, update profile details, and perform customer operations.
* Admin’s have elevated permissions such as adding flights, editing flight schedules, updating booking statuses, and managing users.

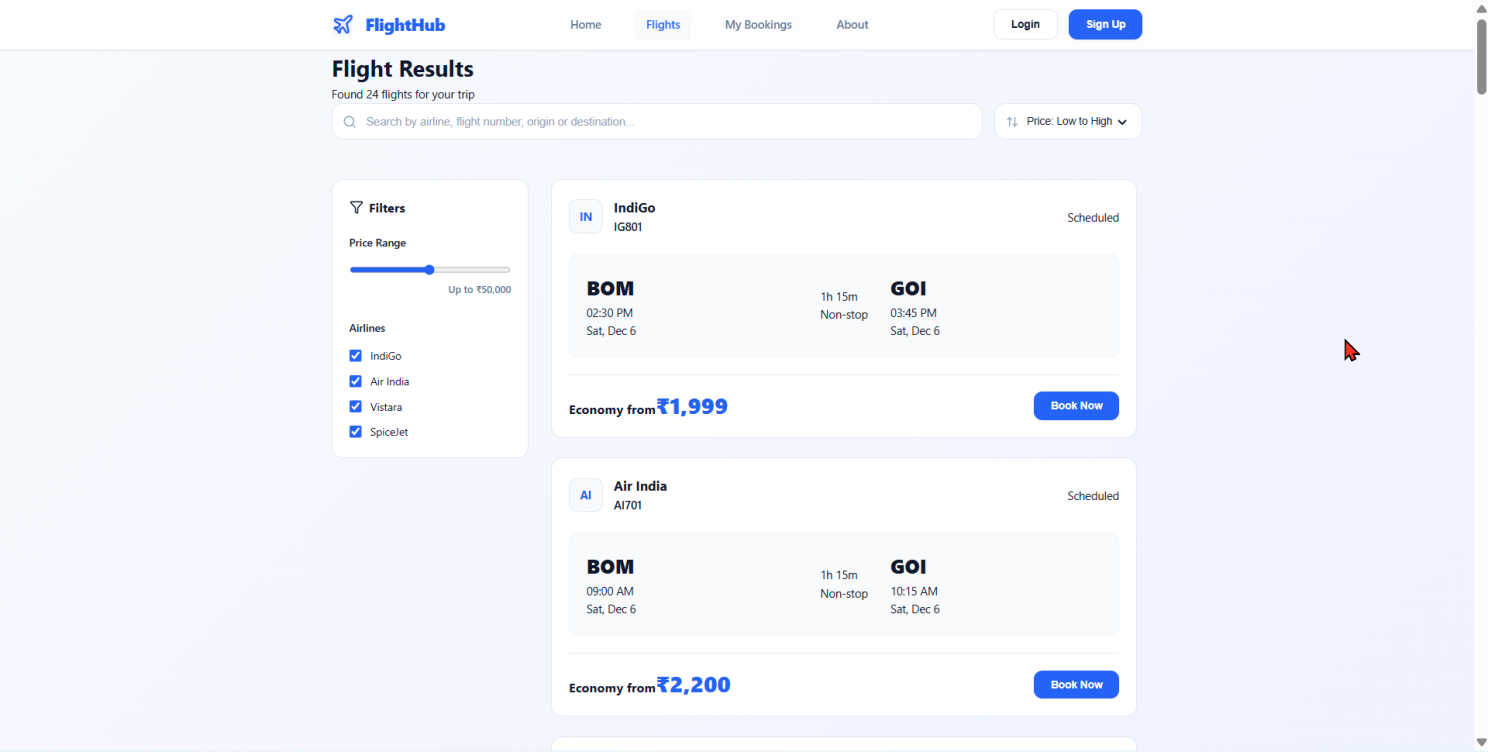
This ensures that each user only has access to features authorized for their role, maintaining security, privacy, and system integrity.

**9. User Interface**

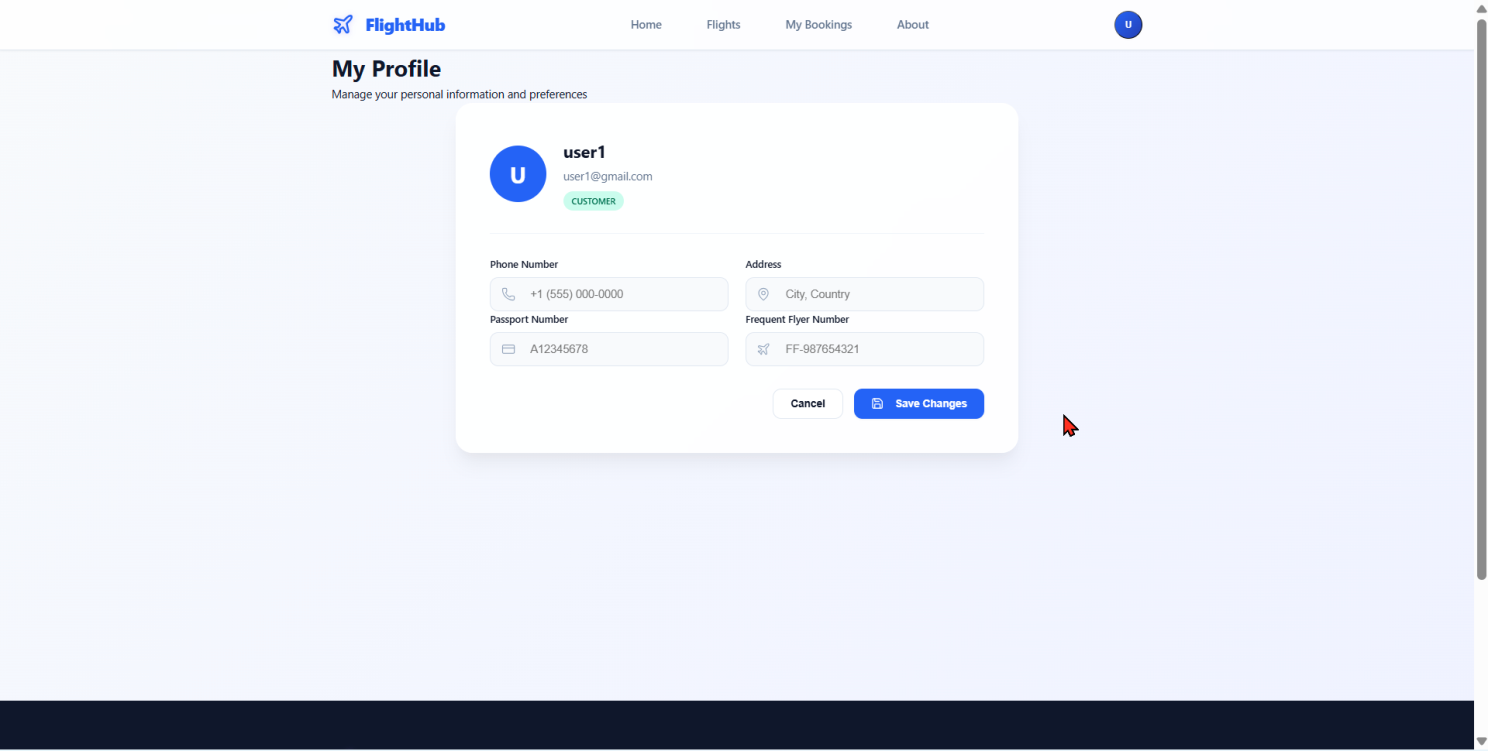
• This section contains visual proof of functionality. Screenshots should be included for:

**Homepage**:

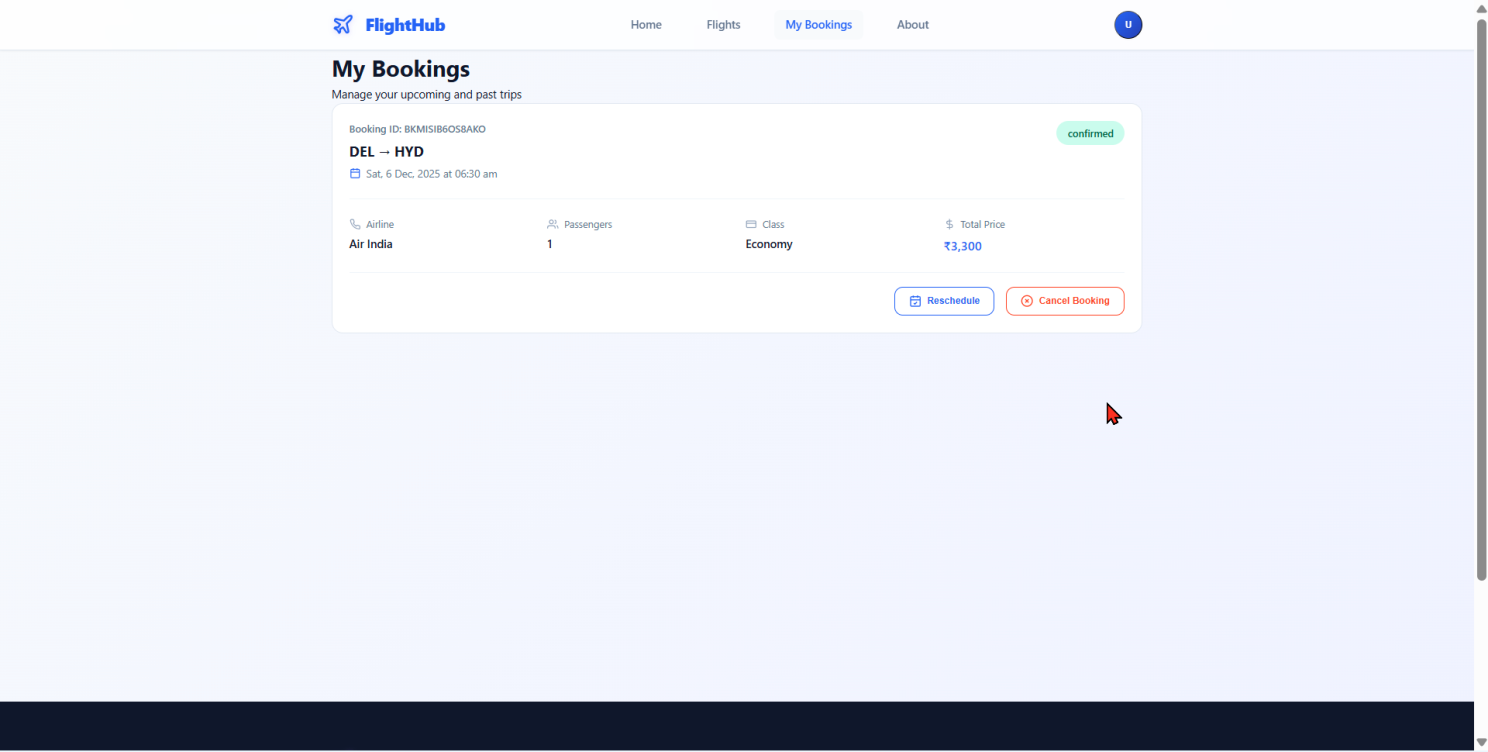


**Flight** **Search Section**

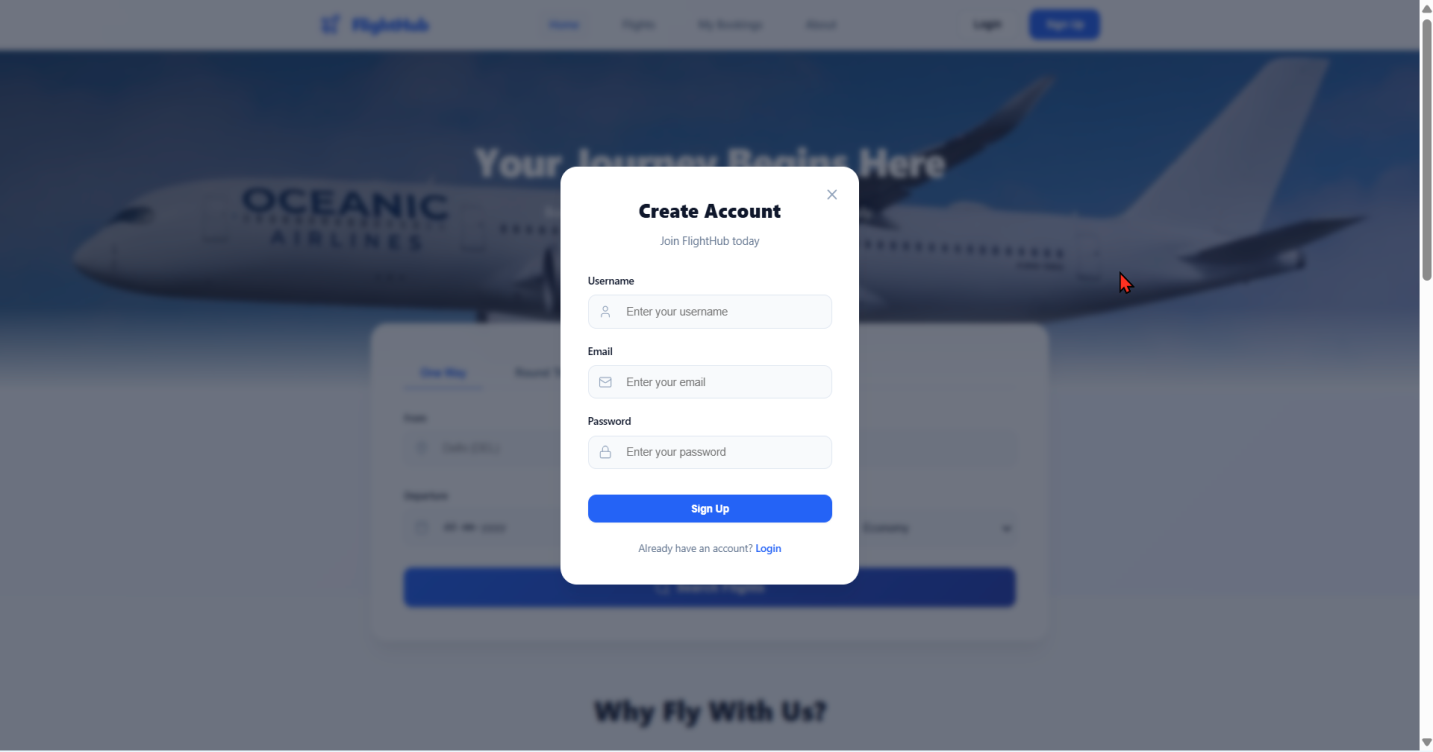
**User** **Profile**:



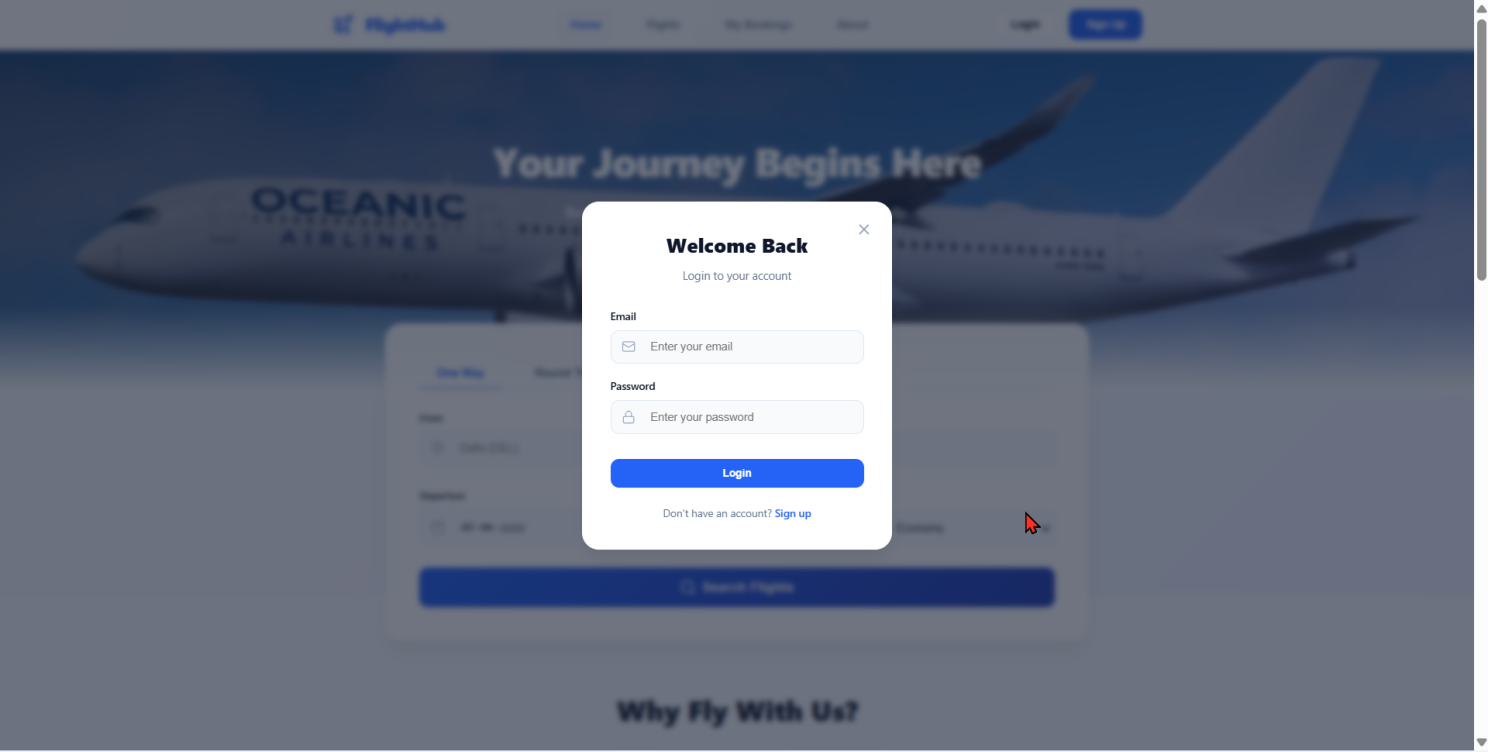
**My Bookings:**



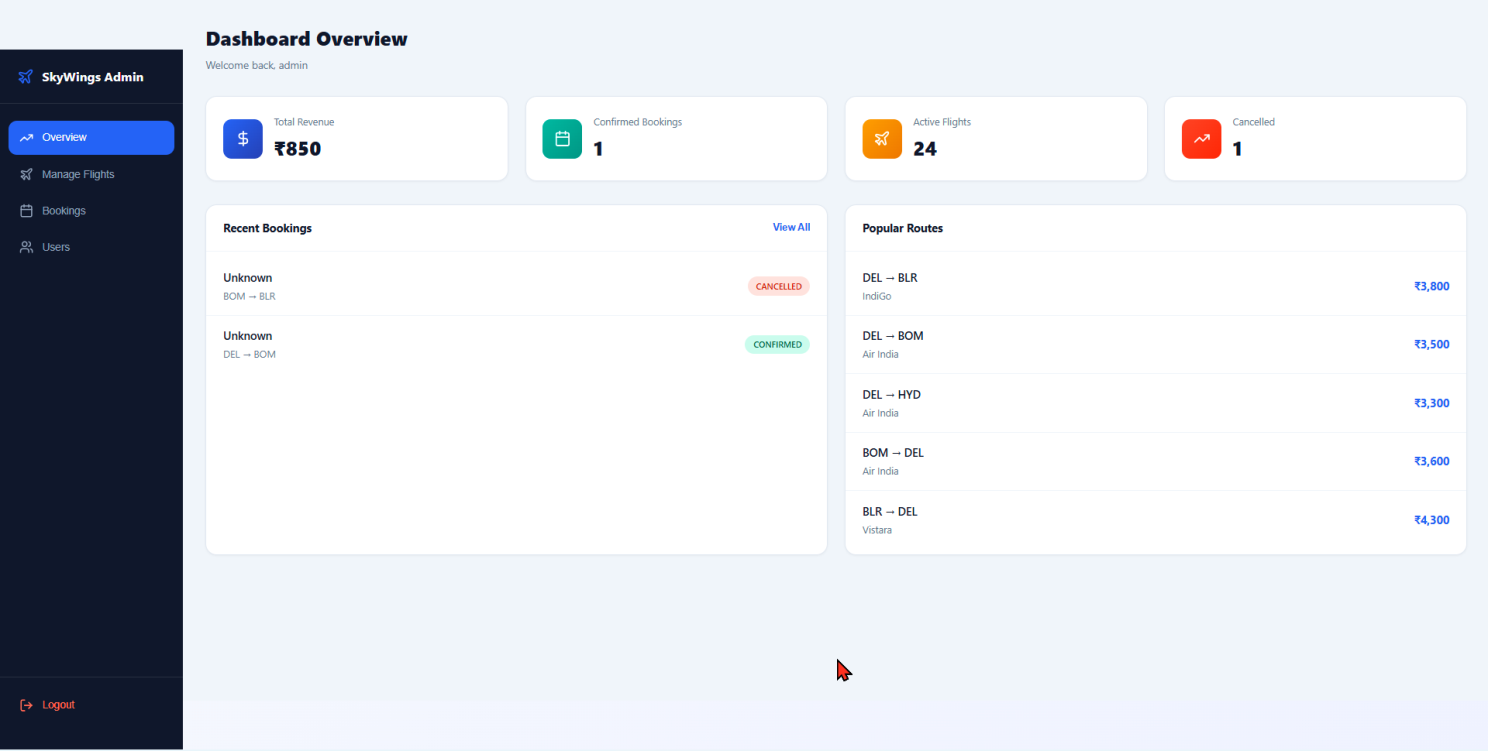
**Register**:



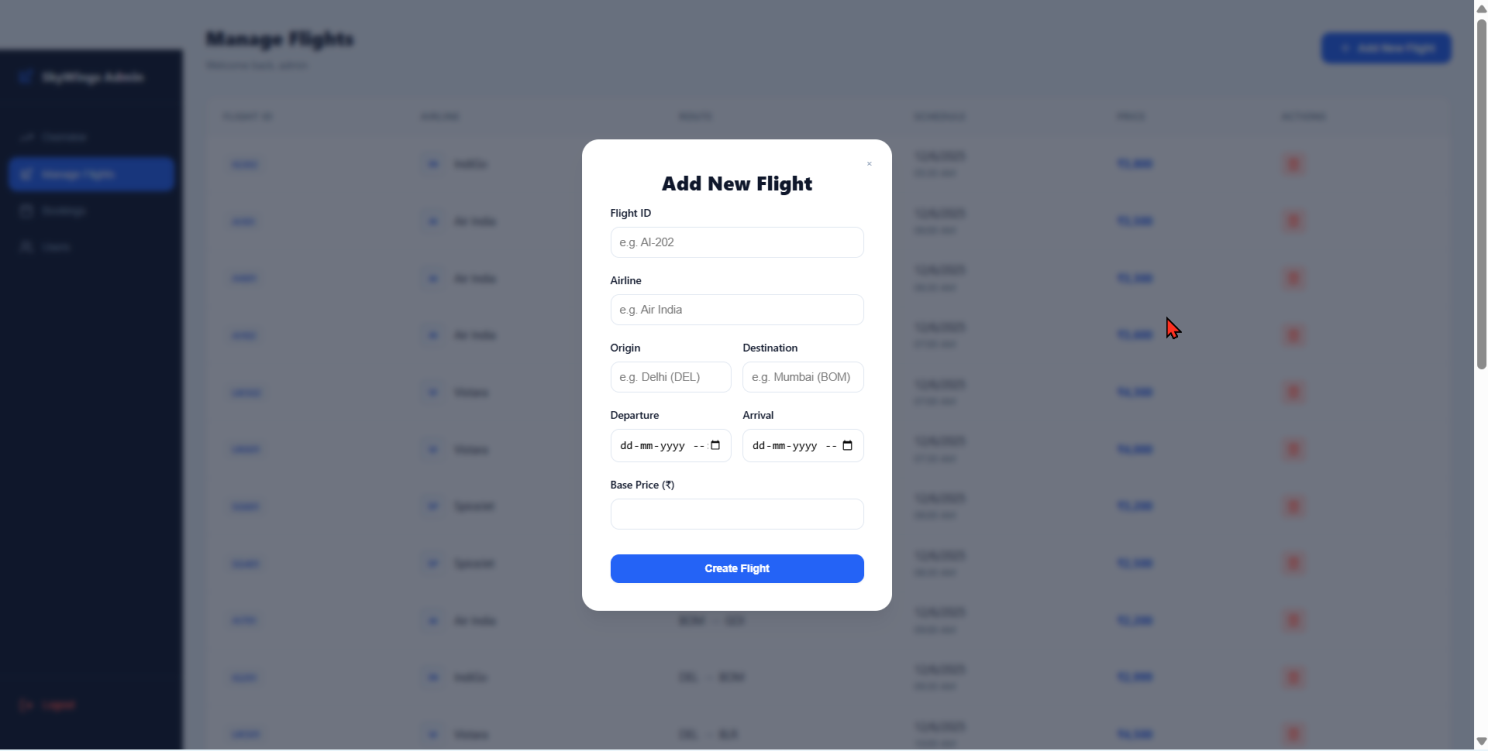
**Login**:



**Admin** **Panel**:



**Add** / **Edit** **Flights** (**Admin**)



**10. Testing**

Testing was performed using:

* Manual UI testing for all user actions and admin actions
* Postman for API endpoints
* Debugging of flight search, profile update, booking functionality

**11. Screenshots or Demo**

• The application is fully functional and has been tested with real‑time usage. Screenshots of the working system have been captured and attached in the documentation, showing:

* Homepage
* Flight Search Section
* User Profile
* My Booking
* Login & Register process
* Admin dashboard
* Add / Edit Flight (Admin)

Demo Link:

<https://drive.google.com/file/d/12sRnQYDlbprY2rtrBvyW_bIlkvf0NB2V/view?usp=sharing>

**12. Known Issues**

| **Issue** | **Description** |
| --- | --- |
| Dummy payment gateway | A dummy payment method is implemented; real payment gateway integration is not available yet |
| Seat allocation sync | Real-time multi-user seat lock not implemented |
|  |  |

**13. Future Enhancements**

Future improvements that can make the platform more scalable and user-friendly:

**Feature** **Purpose**

Real Payment Gateway Enable online transactions

AI-based price prediction Help customers book at best price

Flight delay notification system Real-time updates

Mobile App (iOS / Android) Wider accessibility

Loyalty / Reward Program Customer retention

| **14**. **Conclusion**  The FlightHub application successfully provides a user-friendly and modern approach for real-time flight search and booking. The modular architecture, secure authentication, and role-based dashboard effectively support smooth operations for both customers and administrators. The system is scalable and ready for integration of additional features in future phases. | |  |
| --- | --- | --- |
|  |
|  | |  |