Flight Booking System

Project Documentation

GitHub Repository: <https://github.com/turagavishnusahitya/flight_booking_system>

YouTube link : [Flight Booking System](https://youtu.be/D-2oNFvOKTw)

# Team Details

**Team Leader**:

Tummala Keswanth (23PA1A12M0) - IT  
**Team Members:**

Turaga Vishnu Sahithya (22PA1A05G7) – CSE

Vijjana Sri Ramya (22PA1A45B9) - AI&DS

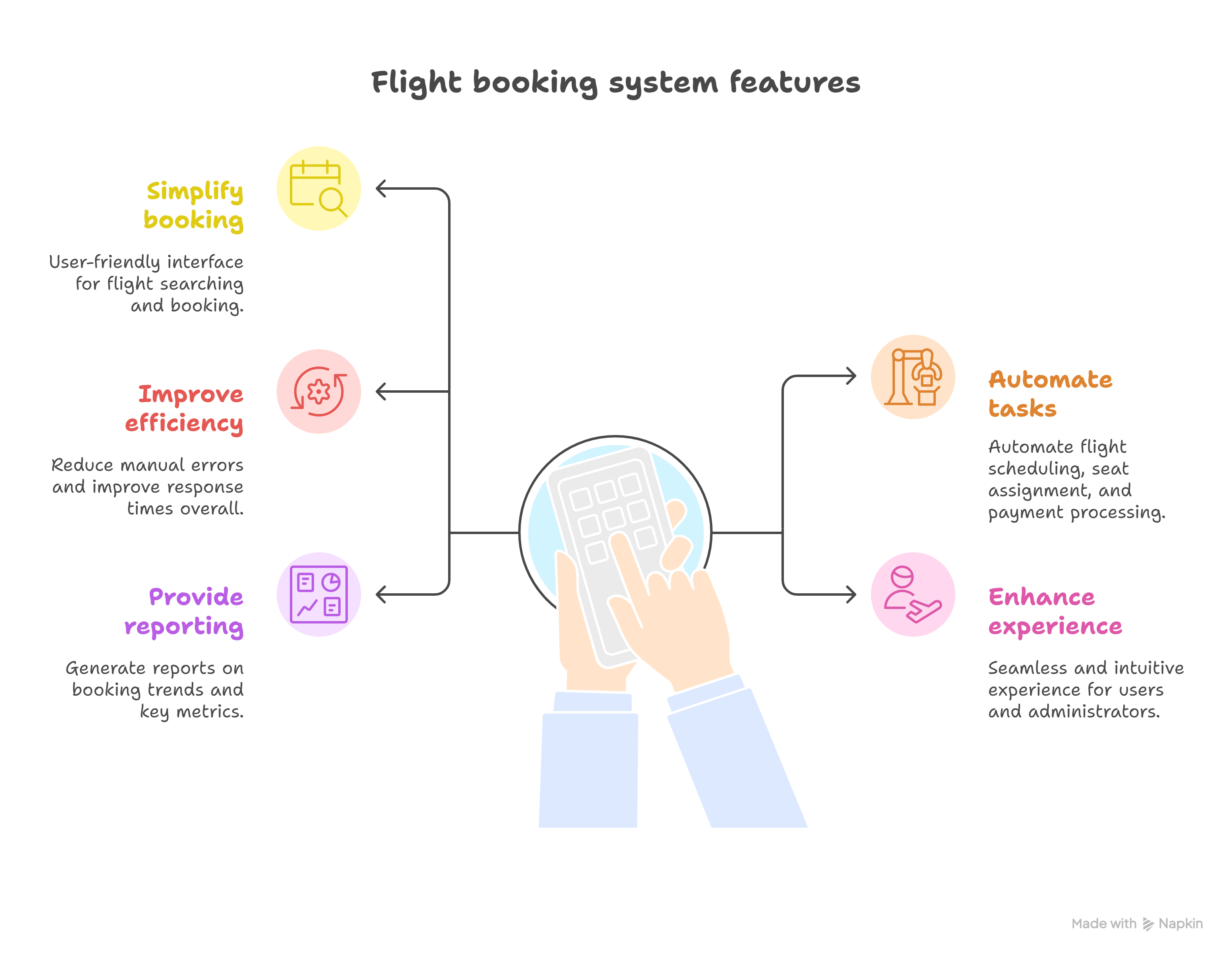
Vipparthi Harsha Vardhan (23PA1A45C8) -AI&DS

# Problem Statement

The traditional flight booking process involves inefficient and disconnected systems. This leads to issues like delayed responses, manual errors, and a poor user experience. The Flight Booking System aims to simplify this through an intuitive, automated platform for both users and administrators to manage bookings with ease.

# Technologies Used

- Frontend: React.js (UI)  
- Backend: Node.js with Express.js  
- Database: MongoDB (NoSQL)  
- Platform: MERN Stack



# Implementation Workflow

## 1. Setup Project Environment

**Purpose**: Initialize backend and frontend environments for MERN stack.

**Steps:**

1. Clone the repository from GitHub.

2. Run `npm install` in both the client and server directories.

3. Ensure MongoDB is running locally or provide MongoDB Atlas URI.

4. Run backend: `npm run server` and frontend: `npm start`.

## 2. User Registration & Authentication

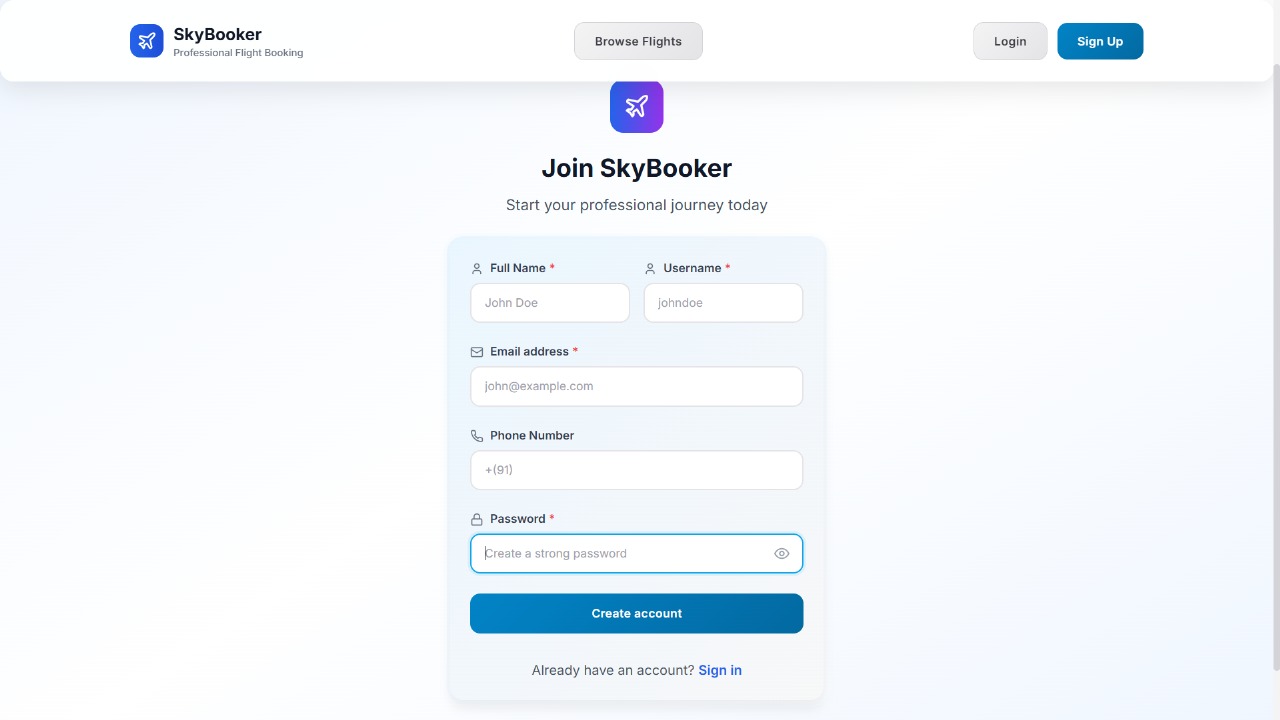
**Purpose:** Allow users to securely register and log in to the platform.

**Steps:**

1. Frontend form collects user details (name, email, password, etc.).

2. Backend validates and stores data in `users` collection.

3. JWT tokens manage session securely after login.



## 3. Flight Search and Booking

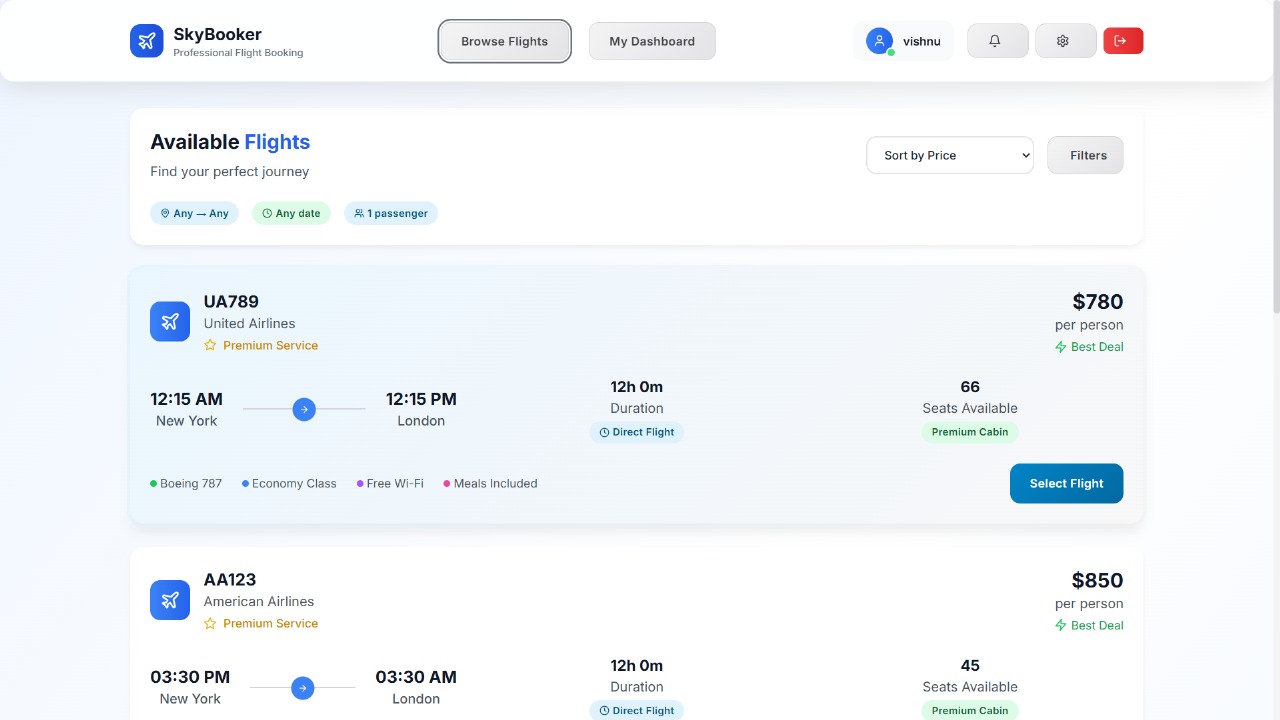
**Purpose**: Let users search for flights and book by entering passenger details.

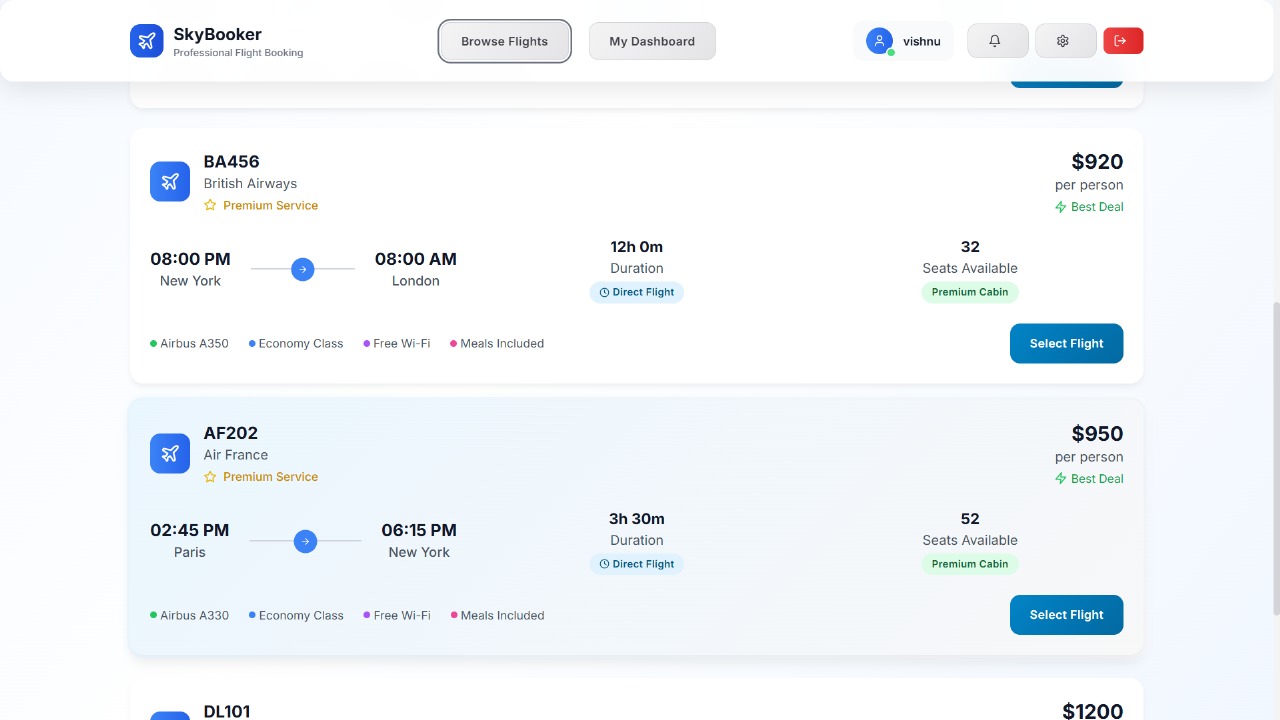
**Steps**:

1. User selects source, destination, date → matching flights are listed.

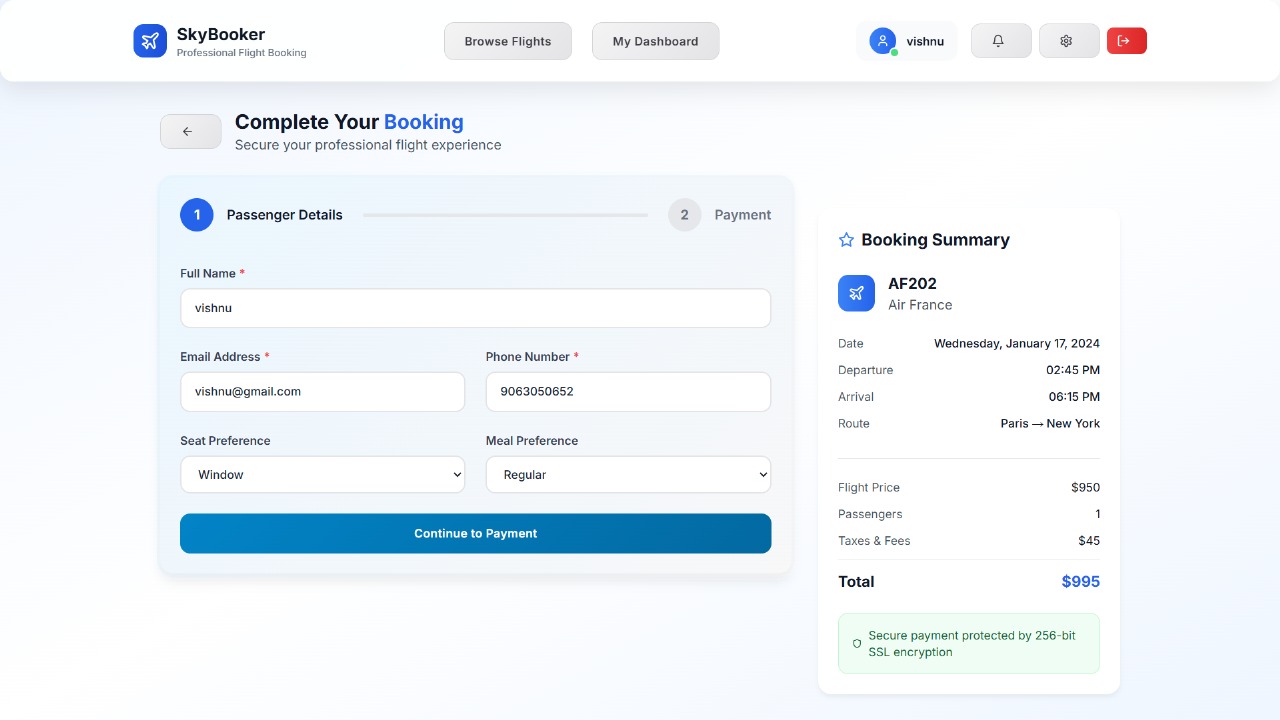
2. User chooses flight and enters passenger details (seat, meal, etc.).

3. Details saved in `bookings` collection with reference to user and flight.





## 4. Payment Integration (Mock)



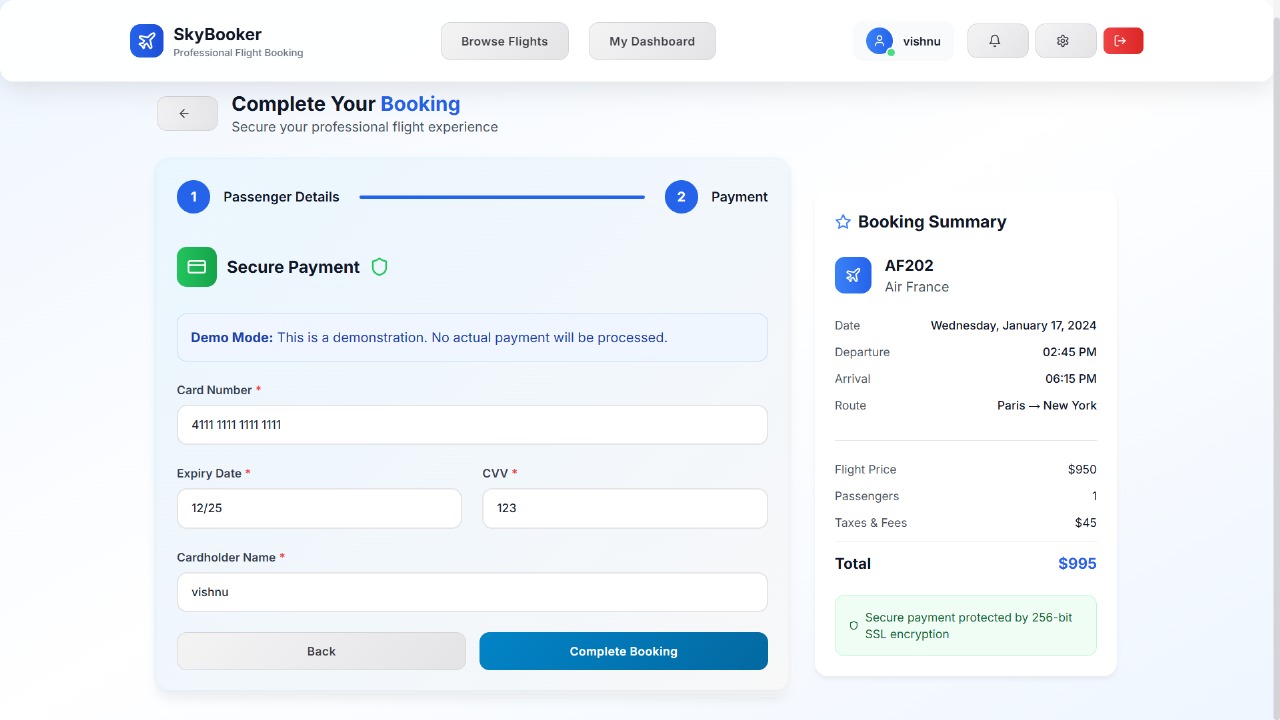
**Purpose**: Allow users to simulate payments securely before booking confirmation.

**Steps:**

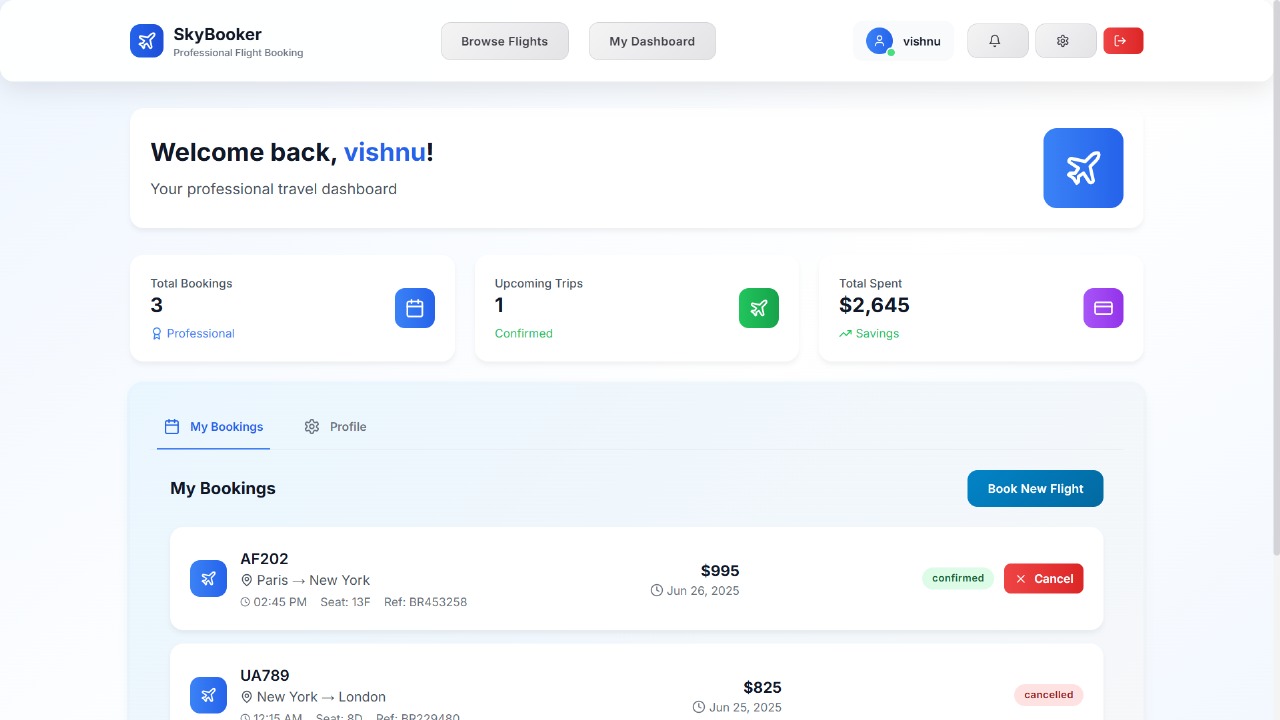
1. Users enter card details (Demo mode – no real transaction).

2. On confirmation, booking is finalized and saved.

3. Booking summary is shown to user with flight and payment breakdown.



## 5. Dashboard & Booking Management

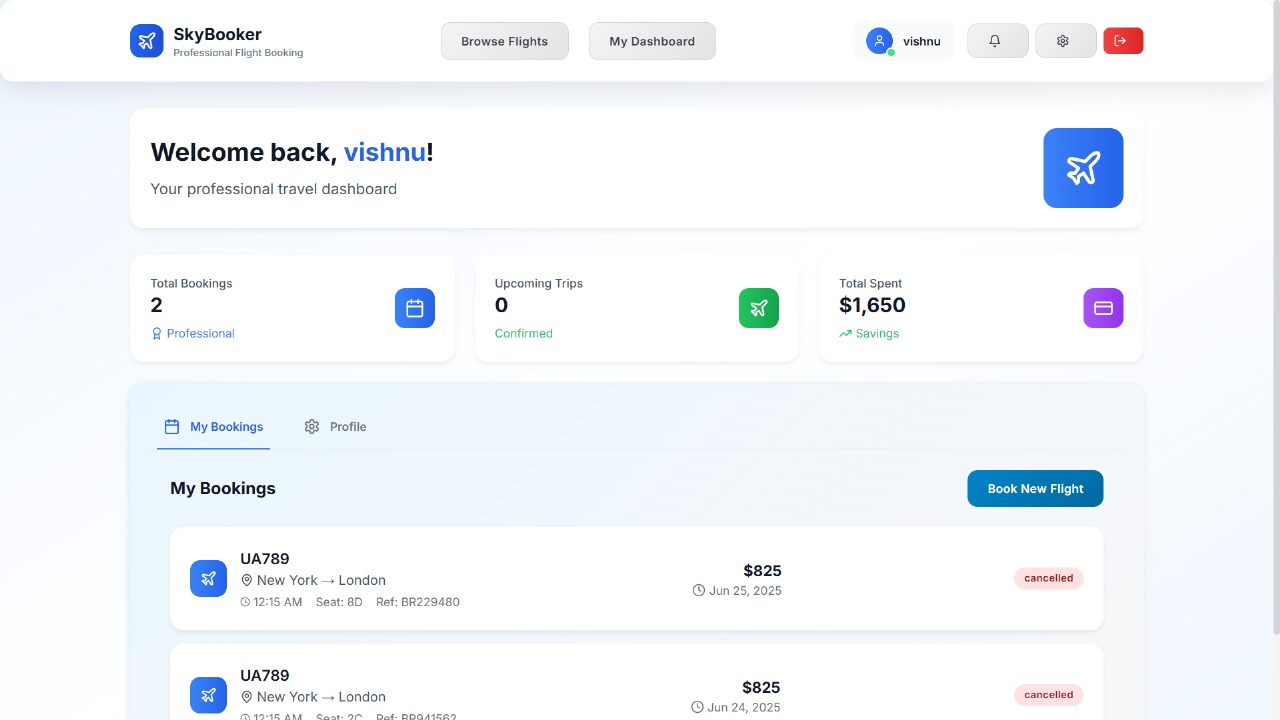


**Purpose**: Enable users to view, manage, or cancel their bookings from a personal dashboard.

**Steps:**

1. Dashboard shows total bookings, upcoming trips, and total amount spent.

2. Users can cancel any booking; status is updated in MongoDB.



## 6. Admin Module (Optional)

**Purpose**: Admins can manage flights and monitor bookings.

**Steps**:

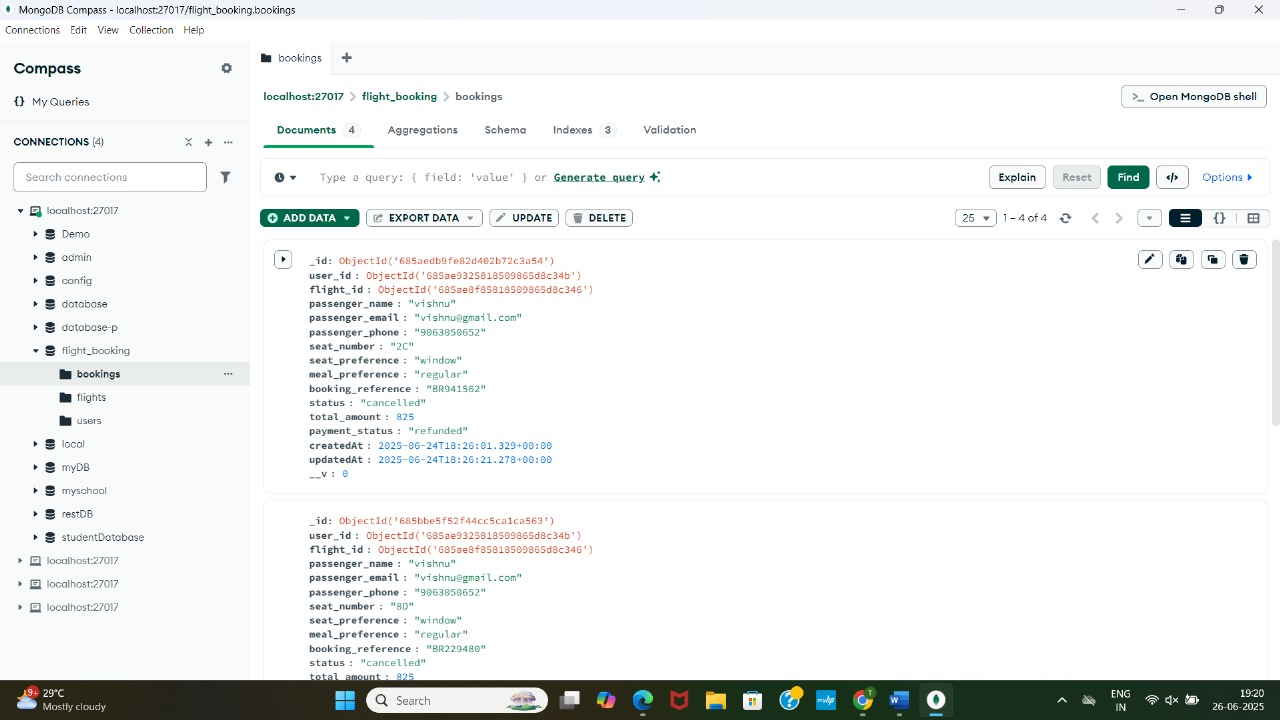
1. Admin login allows access to flight management dashboard.

2. Flights can be added/removed/edited from `flights` collection.

3. Admins can also view all user bookings.

# Database & MERN Stack Details

The system uses MongoDB to store all application data including users, flights, and bookings. Collections used are:  
- `users`: contains user credentials and profile information  
- `flights`: holds all flight information (timing, route, price, etc.)  
- `bookings`: links users and flights with passenger, seat, and payment details  
  
MongoDB Compass enables live tracking and visualization of data.



↑ Bookings collection in MongoDB Compass.

# Conclusion

The Flight Booking System project effectively digitizes the flight reservation workflow. Built on the robust MERN stack, it allows users to register, search flights, enter travel and payment details, and track or cancel bookings with ease. With a clean UI, secure backend, and flexible database, it sets the foundation for a scalable travel booking platform.