**Full Stack Development with MERN**

**Flight Finder: Navigating Your Air Travel Options**

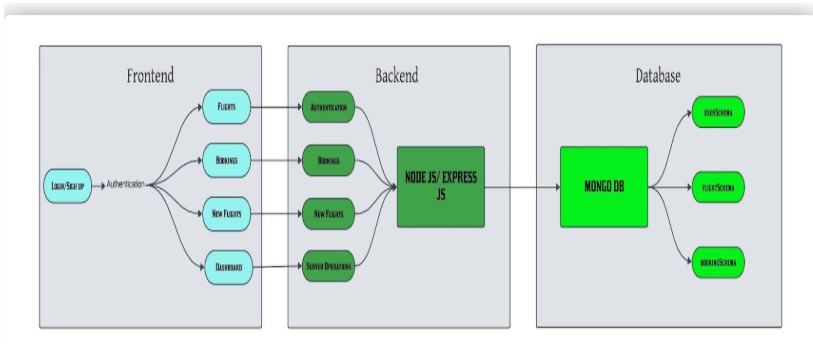
1. **Introduction:**

* **Project Title:** Flight Finder: Navigating Your Air Travel Options
* **Team Members:**
* **Team Leader :** Arja Geetha Priya
* **Team member :** Alajingi Indueeswari
* **Team member :** Annavarapu Triveni
* **Team member :** Amreen

# Project Overview

* **Purpose:**  
   FlightFinder is a full-stack flight booking web application designed to simplify and optimize the process of searching, filtering, and booking airline tickets. It ensures a smooth and efficient experience for both frequent travelers and occasional flyers.
* **Features:**
* User Authentication and Role-Based Access
* Flight Search with Filters (e.g., direct flights, preferred airlines)
* Real-Time Flight Listings
* Interactive Seat Selection
* Secure Payment Gateway Integration
* Admin Dashboard for Flight and Booking Management
* Responsive UI for all devices

### ****Architecture****



* **Frontend:**  
  Built using React.js, the frontend handles user interaction, search inputs, seat selection UI, and dynamic rendering of available flights using Axios to communicate with backend APIs.
* **Backend:**  
  Node.js with Express.js forms the backend, which includes RESTful API endpoints for user registration/login, flight search, booking handling, and admin tasks.
* **Database:**  
  MongoDB serves as the primary database to store:
* Users
* Flights
* Bookings

Mongoose is used for schema modeling and efficient database operations.

### ****Setup Instructions****

* **Prerequisites:­­­­­­**
* Node.js and npm: [Install Node.js](https://nodejs.org/en/download/)
* MongoDB: [Download MongoDB Community Edition](https://www.mongodb.com/try/download/community)
* Git: [Download Git](https://git-scm.com/downloads)
* Code Editor: Visual Studio Code recommended ([Download](https://code.visualstudio.com/download))
* **Installation Steps:**

1. Clone the repository

* git clone https://github.com/harsha-vardhan-reddy-07/Flight-Booking-App-MERN

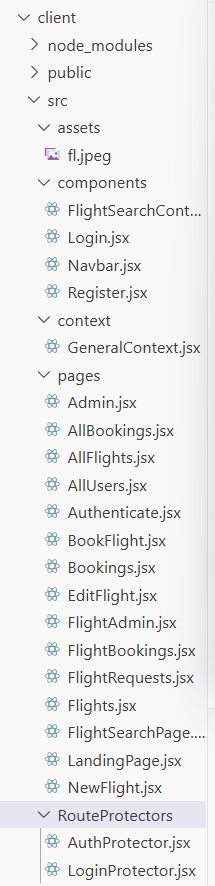
1. Navigate into the directory:

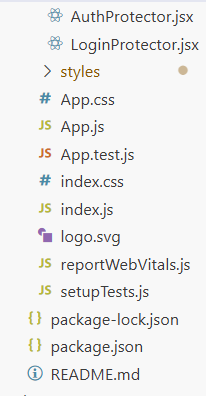
* cd Flight-Booking-App-MERN

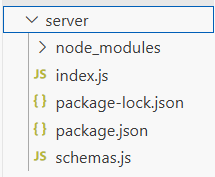
1. Install dependencies:

* npm install

### ****Folder Structure****







### ****Running the Application****

* **Frontend:**

cd client

npm start

* **Backend:**

cd server

npm run start

App runs by default at: <http://localhost:3000>

### ****API Documentation****

* **Example Endpoints:**
* POST /api/users/register – Register a new user
* POST /api/users/login – Login a user
* GET /api/flights – Retrieve all flights
* POST /api/bookings – Create a new booking
* GET /api/bookings/:userId – Get user-specific bookings

Each endpoint includes authentication (where required) and returns JSON data.

### ****Authentication****

* **JWT (JSON Web Token)** is used for authentication and authorization.
* Tokens are generated during login and stored in localStorage.
* Protected routes verify token validity before allowing access.

### ****User Interface****

* Clean and responsive UI using React, Bootstrap, and Material UI
* Components include:
  + Flight search form
  + Seat selection modal
  + Booking confirmation page

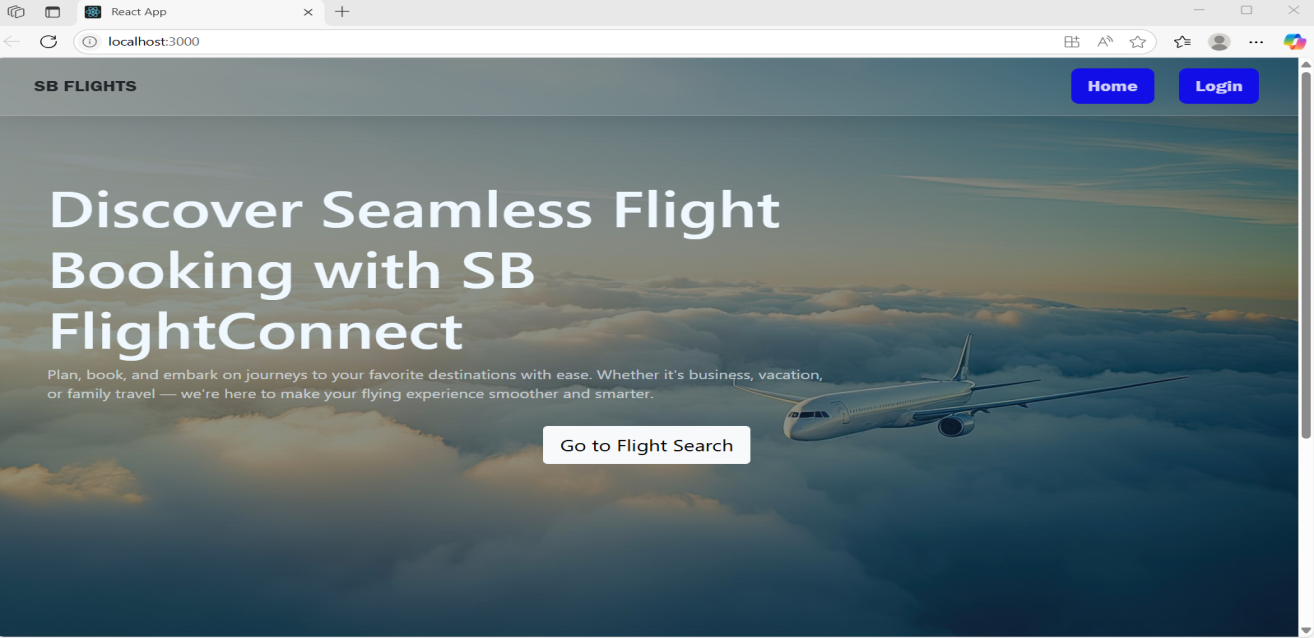
Screenshots and screen walkthroughs can be added here.

### ****Testing****

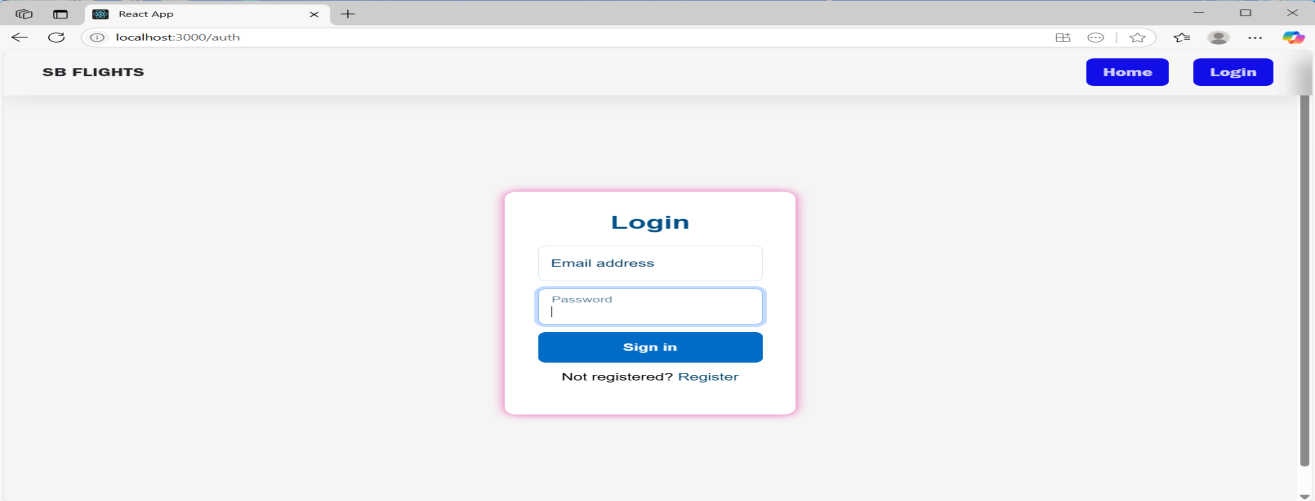
* **Strategy:**
* Manual testing across major browsers and devices
* Functional testing of major user flows (search, book, login)
* **Tools:**
* Postman for backend API testing
* Browser dev tools for UI responsiveness and console logs

### ****Screenshots or Demo****

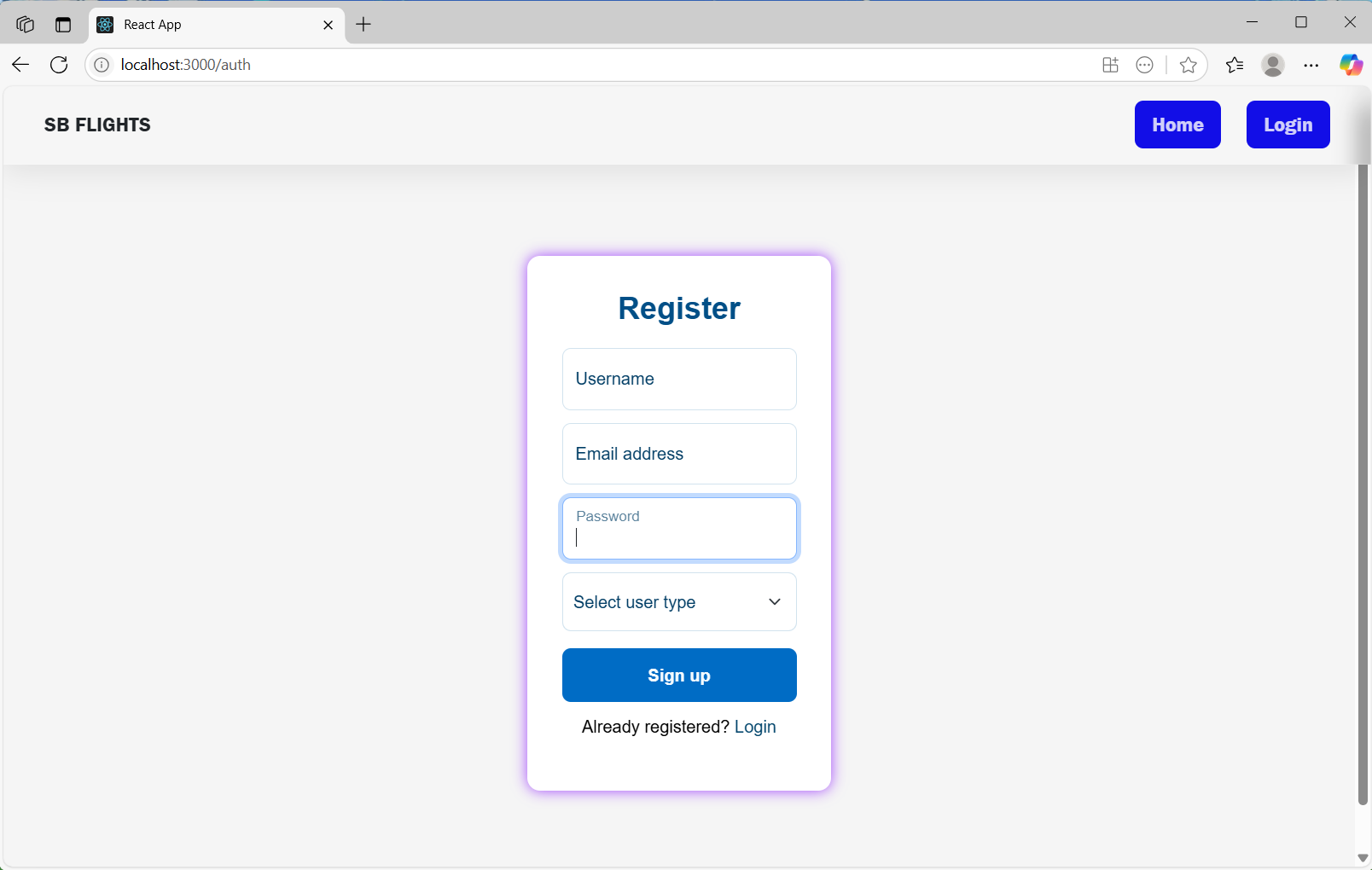
**Home Page:** In the home page, you will know about this website

****

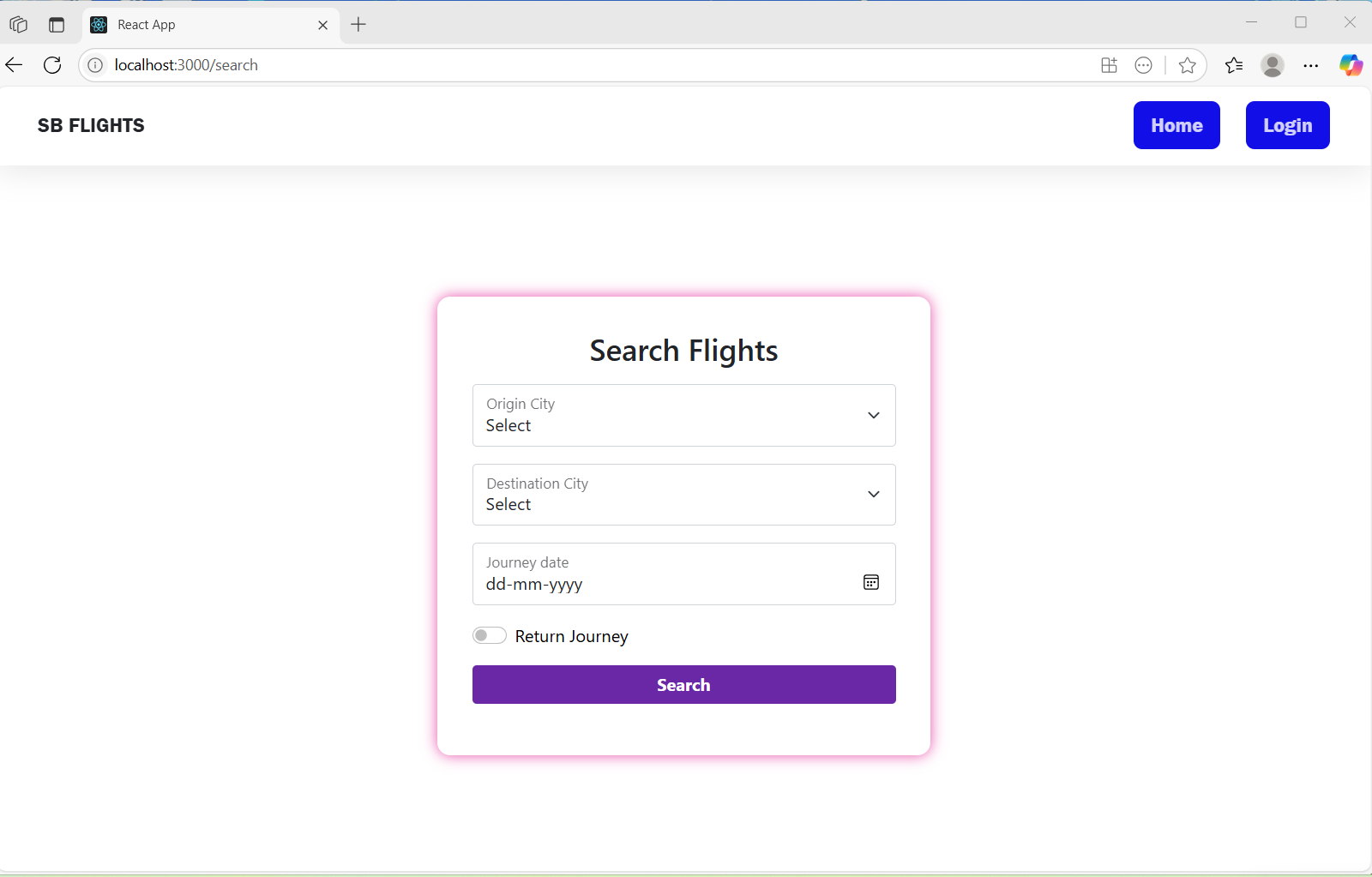
**Login Page:** Login into the website who are already registered in the registration page**.**

****

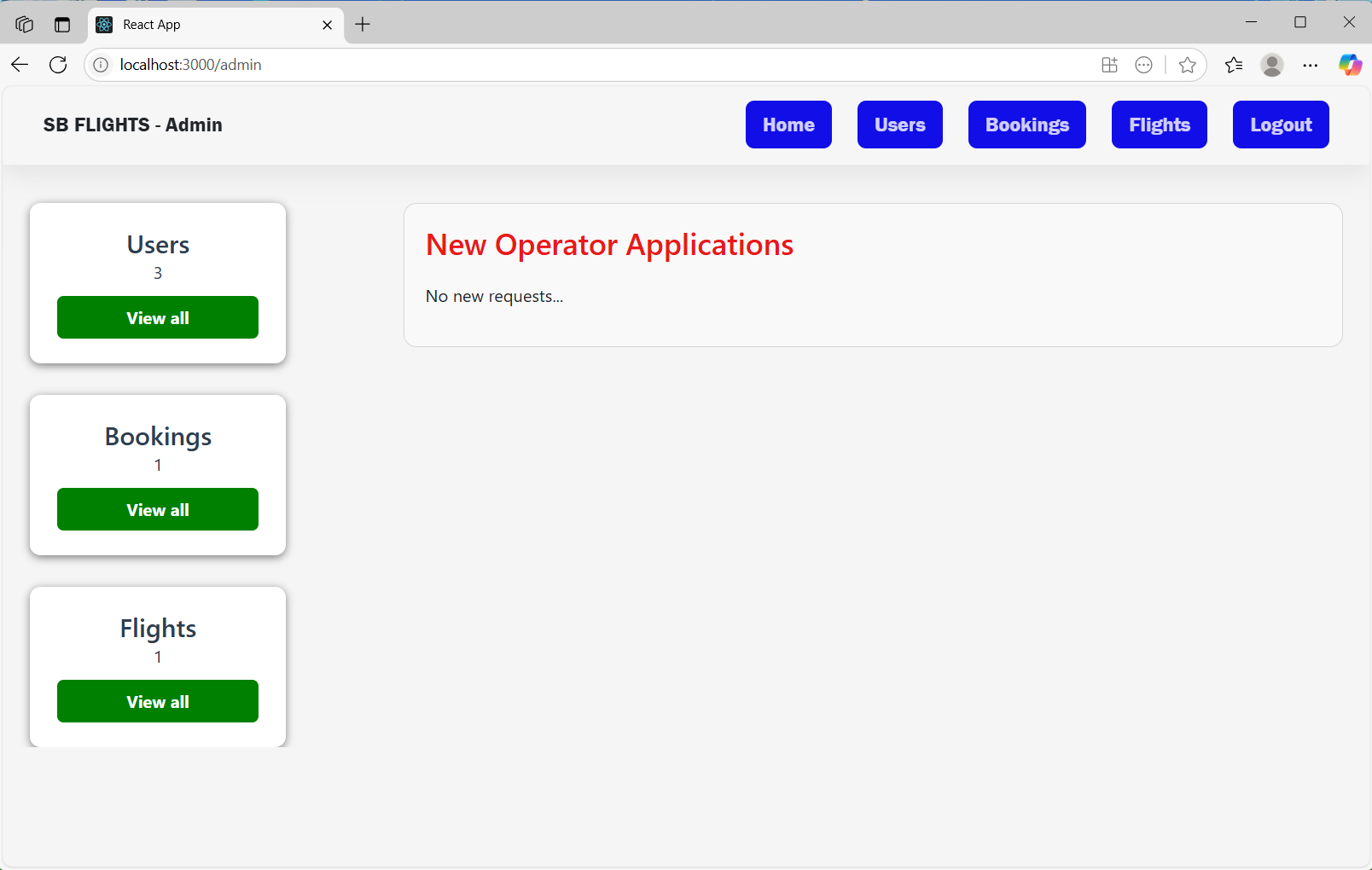
**Register Page:** In the registration page, New peoples like admin, flight operator and customers are registered. After that they will got to the login page

****

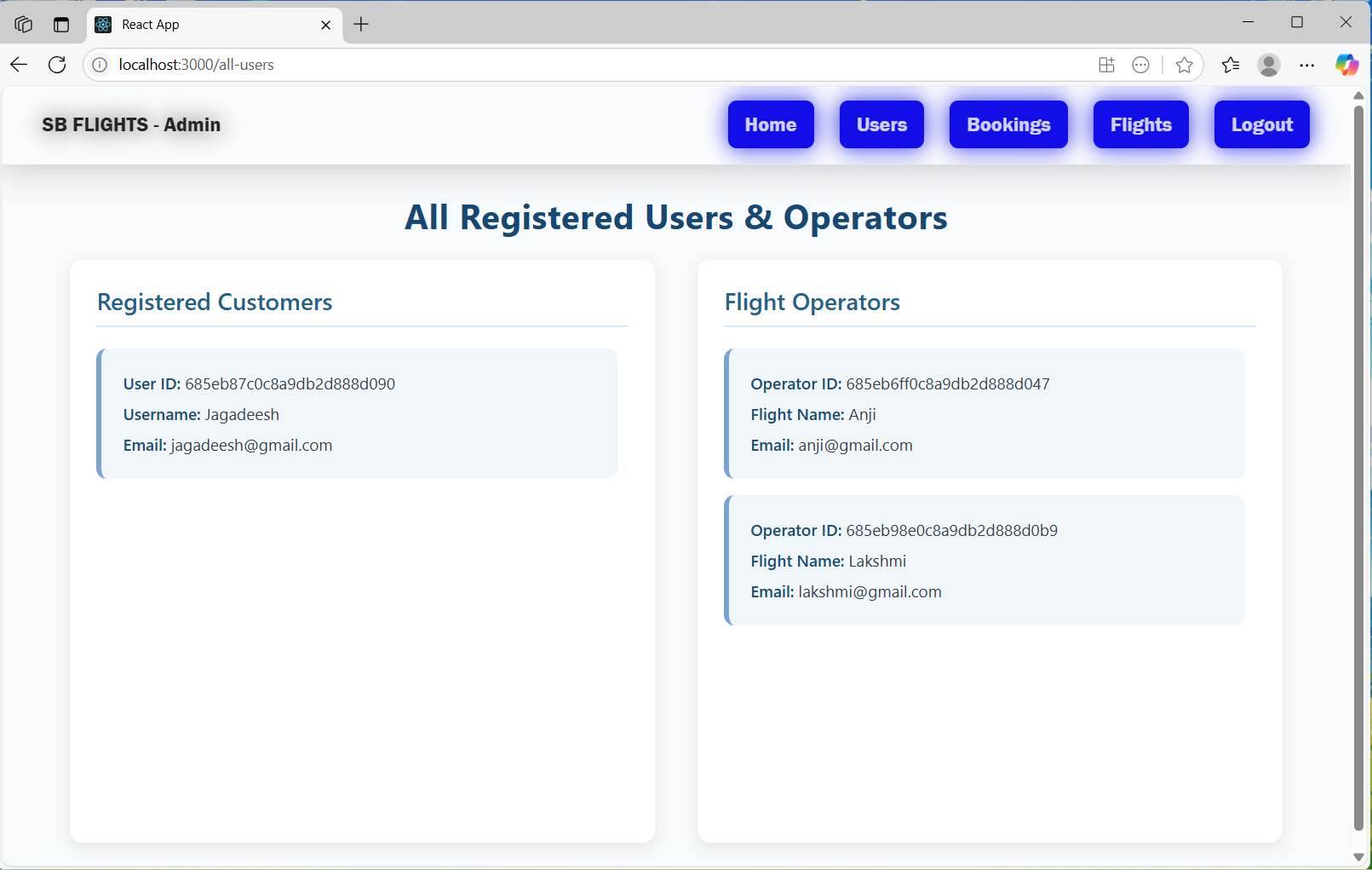
**Search Flight Page:** In this page, customers will search the Flights . If the flight is found then they will book otherwise,they don’t see their flights

****

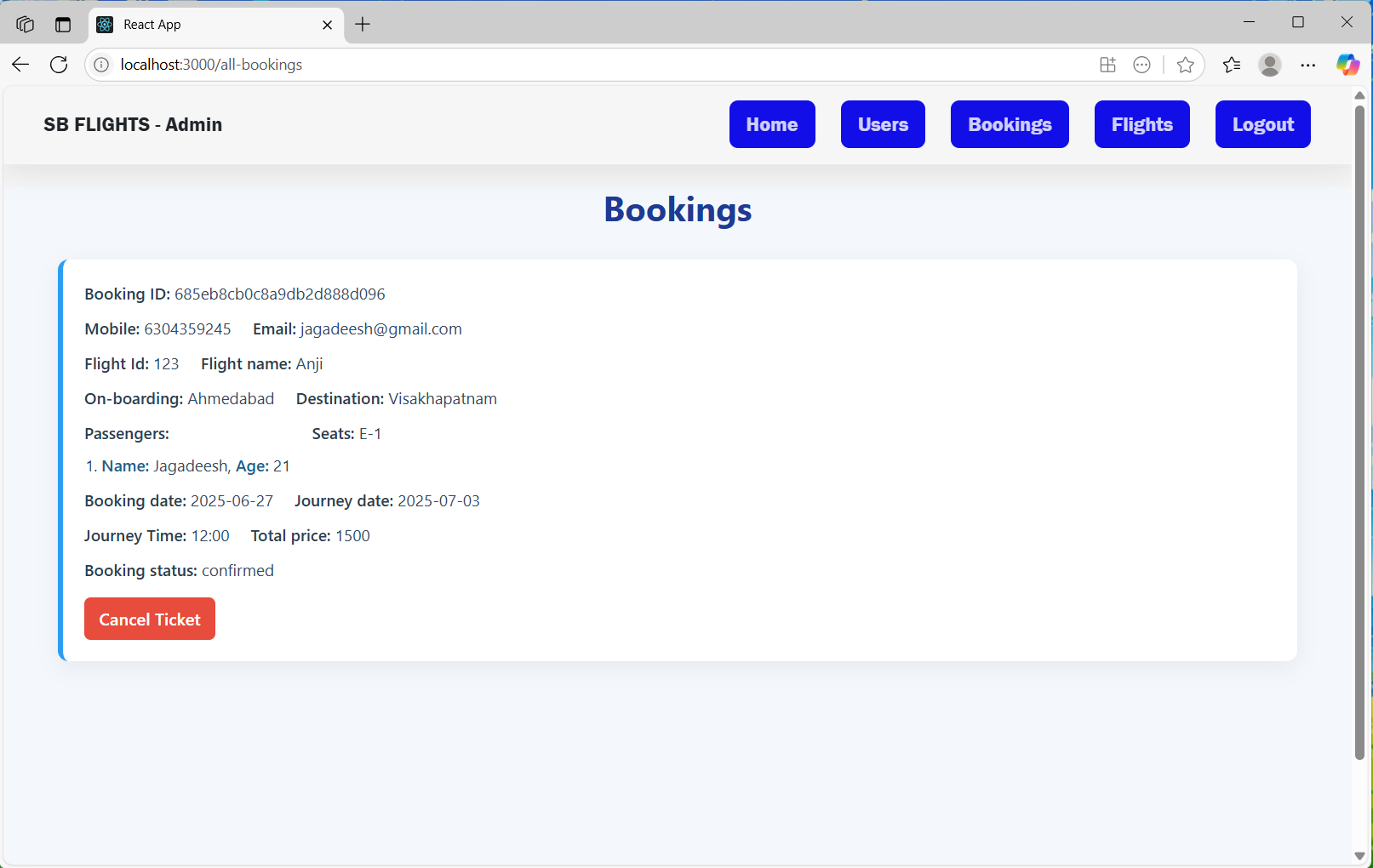
**Admin Page:**In this page the admin manages the number of users ,bookings,flights and all the flight operators

****

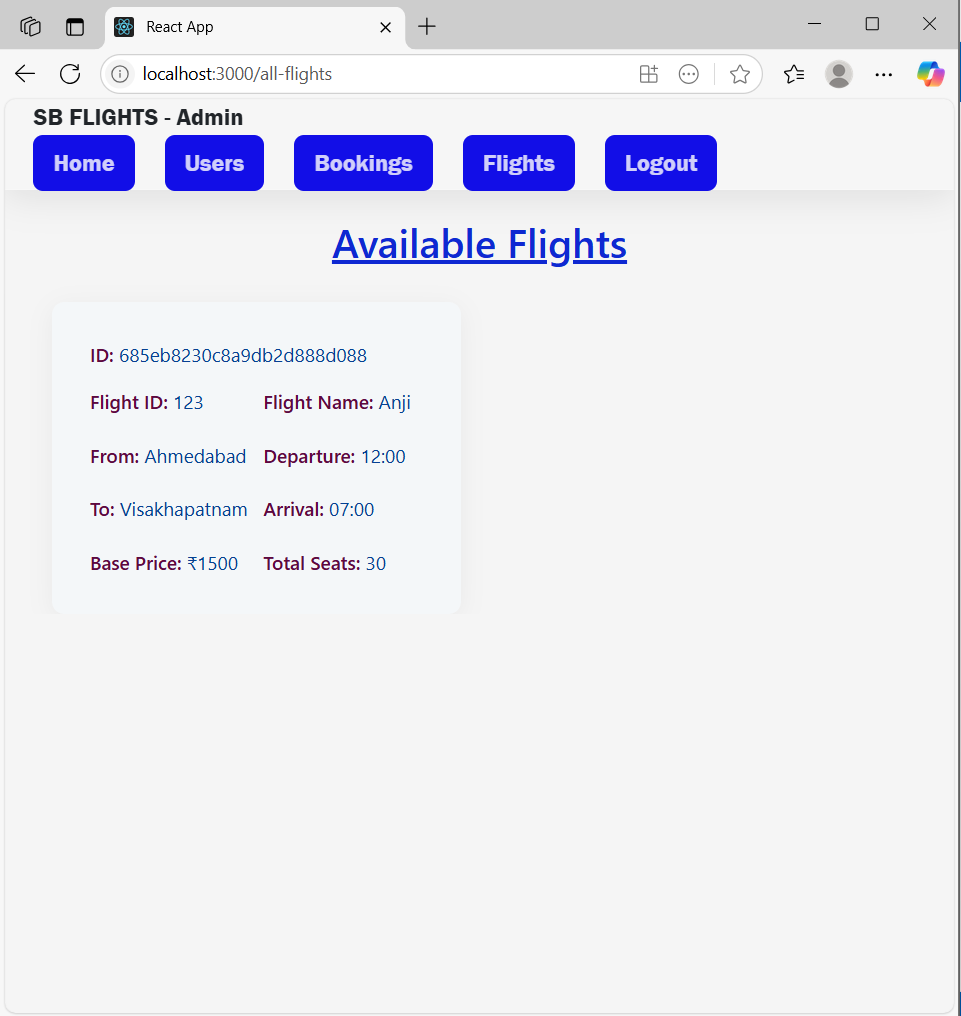
**All users and Operators:** Admin will see the all user and operators details



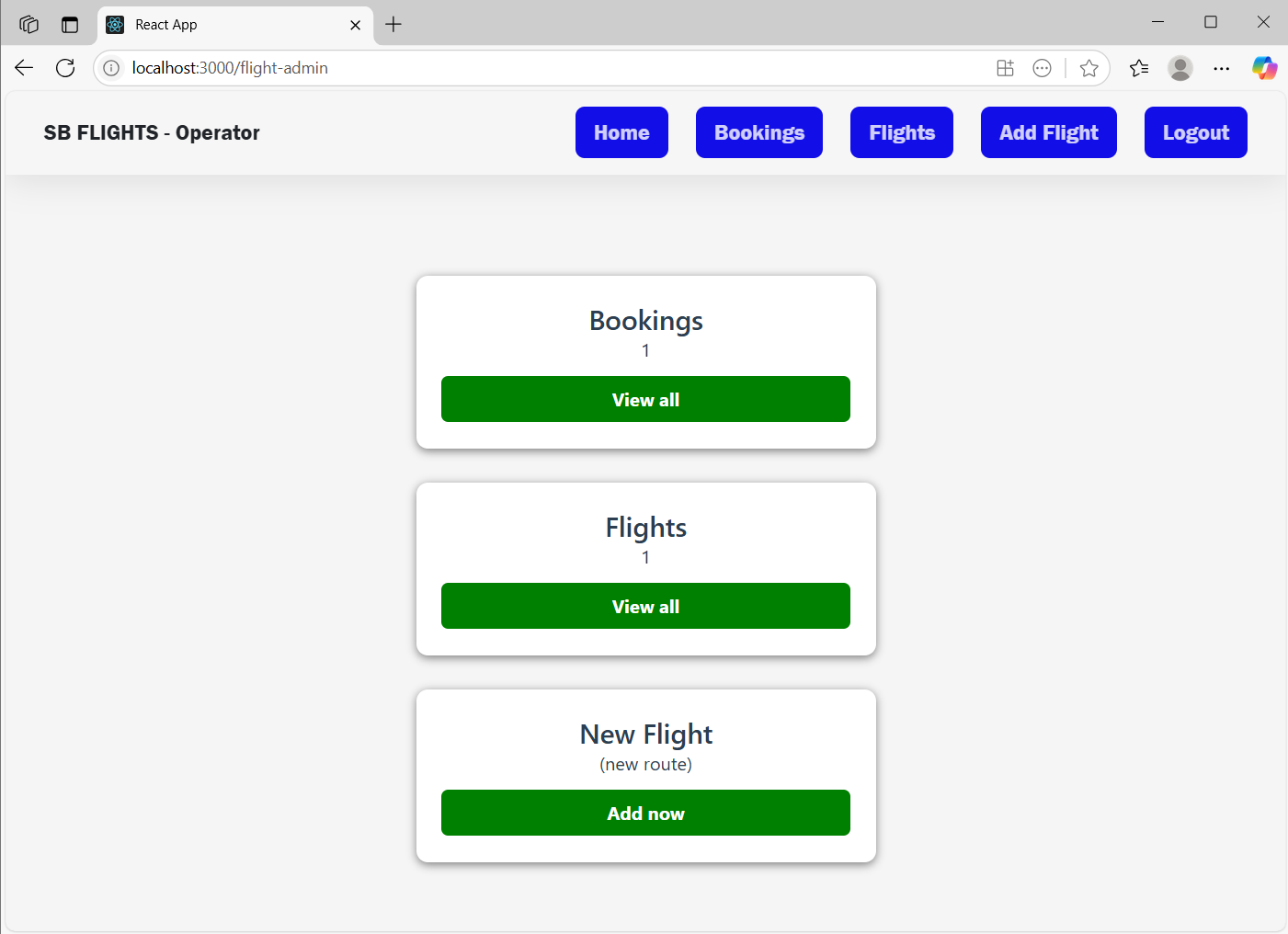
**Bookings:** Admin will see the all the bookings



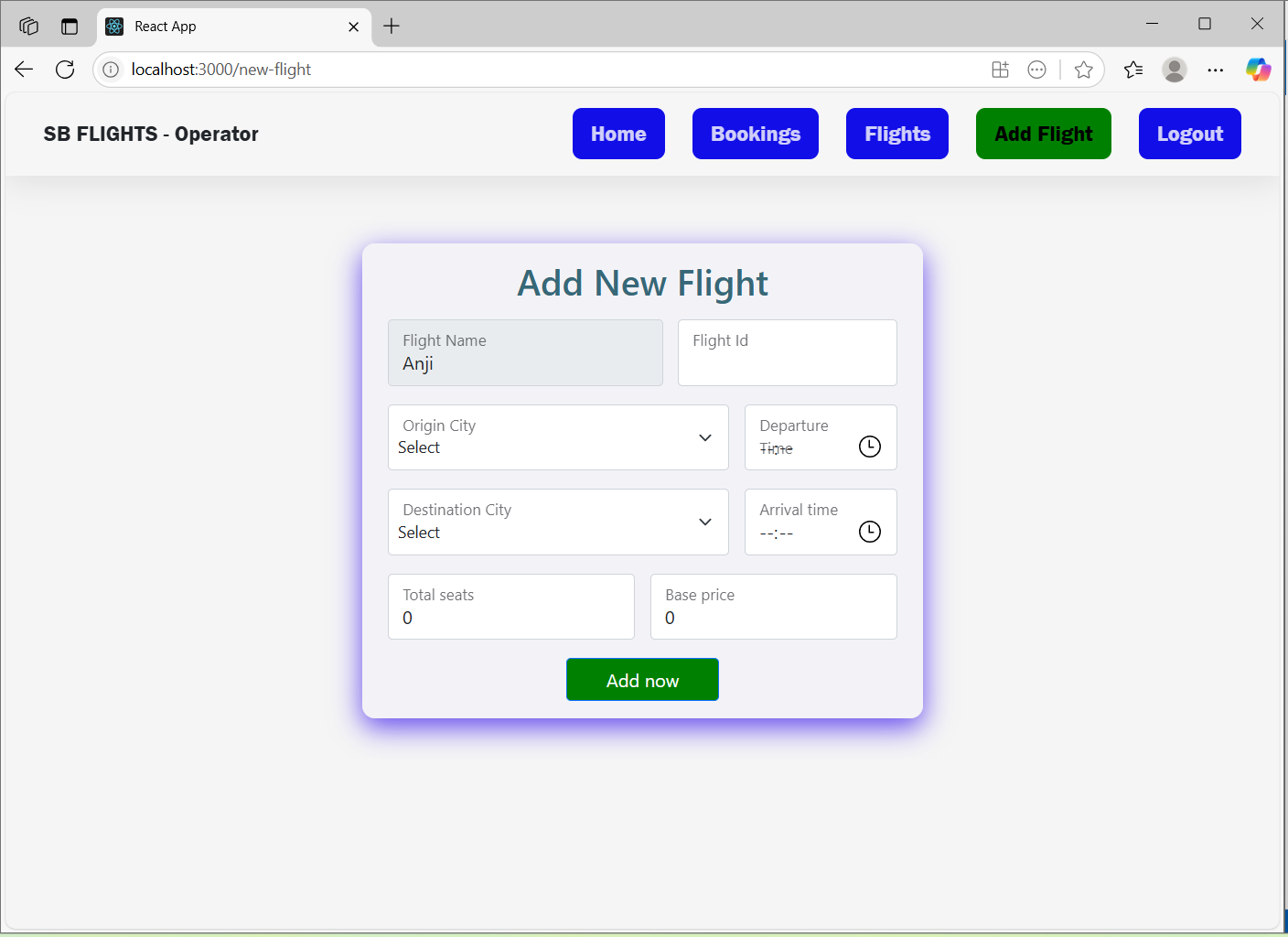
**All flights:** Admin will see the all available flights



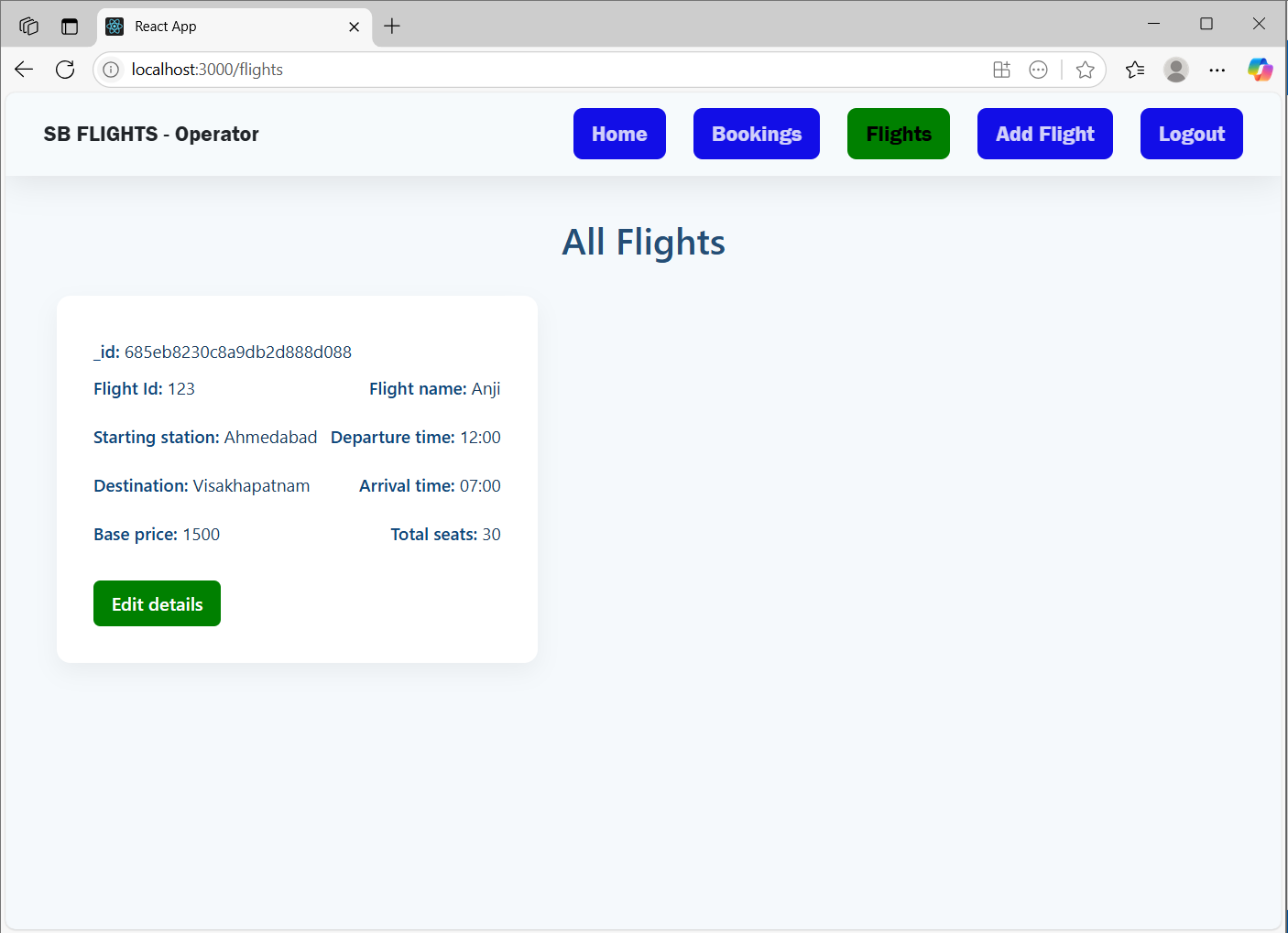
**Flight Operator page:** In this page th flight operator manages the flights ,bookings on their own flights



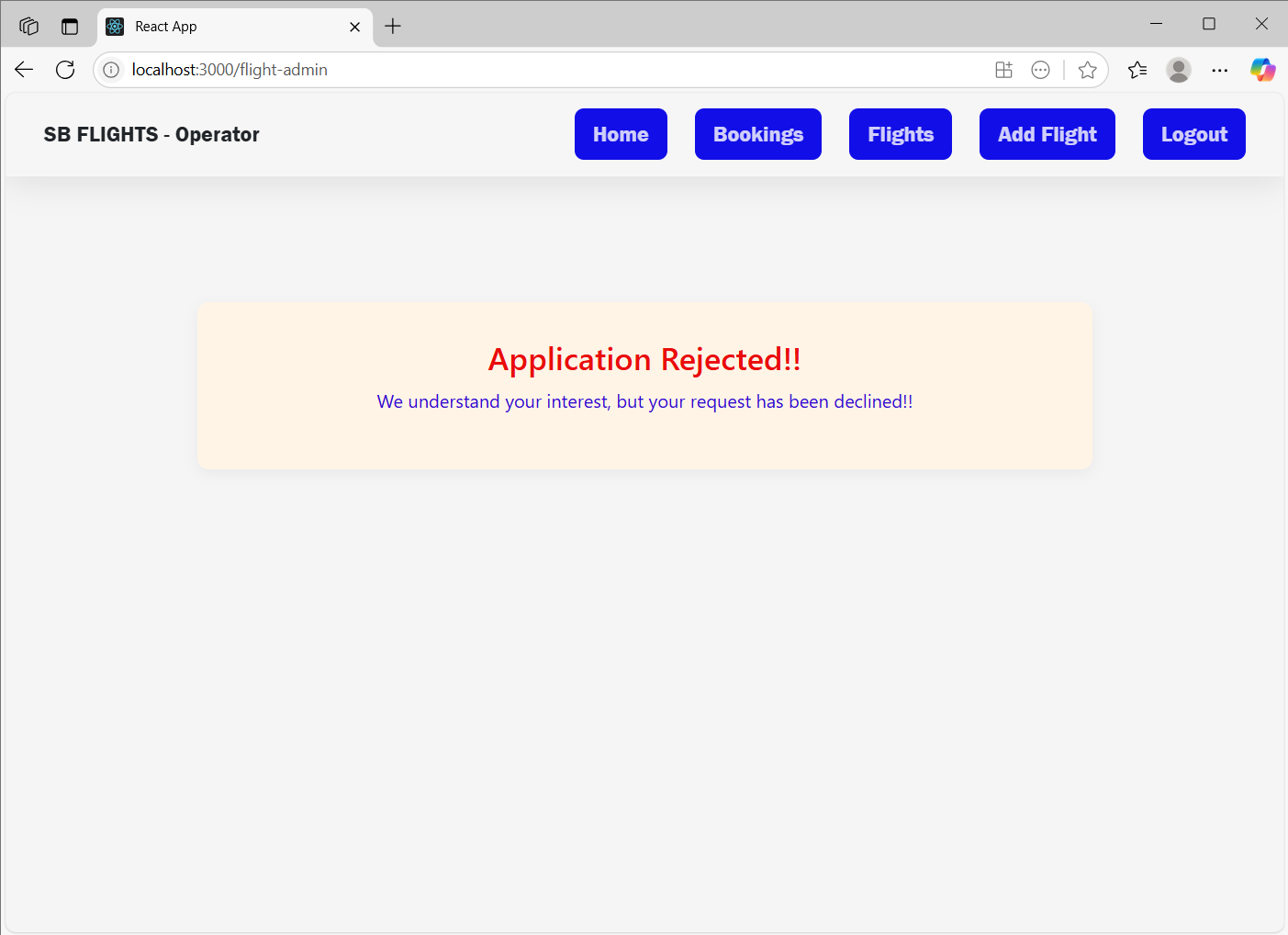
**Add flight Page:** In this page operator will add the new flight



**All Flights:** In this page ,They will see the all available flights that are added by the operator



**Operator Rejection page:** In this page,The operators see their application was rejected by the admin



### ****Known Issues****

* Some edge cases with real-time seat availability not fully handled
* Limited payment gateway options currently

### ****Future Enhancements****

* Integration with real airline APIs for live data
* Mobile App version using React Native
* Multilingual support
* Integration of notification (email/SMS) services
* Enhanced admin analytics dashboard