FULL STACK DEVELOPMENT WITH

MERN PROJECT DOCUMENTATION

# Introduction

Project Title : Flight finder -Navigating your Air Travel Option

Team id : LTVIP2025TMID59024 Team

Members:

# Team Leader:

Yarlagadda Nehapriya(22481A05P9) – Full Stack Developer & Project Coordinator

Responsible for overall planning, coordination, GitHub management, and integration of frontend and backend.

# Team Member:

Yadala Pujitha (23481A12P4)– Frontend Developer

Works on the React-based UI, handles component design, page routing, and user interactions.

# Team Member:

Y Devisri(24485A1226) – Backend Developer

Builds RESTful APIs using Node.js and Express.js, manages authentication and server logic.

# Team Member:

Yaddanapudi Murari (22481A12I7)– Database Administrator Designs and manages MongoDB schemas, handles CRUD operations and ensures data consistency.

# Project Overview

* + **Purpose:**

To simplify flight booking and management by providing users with real-time availability, secure bookings, and user-friendly search features.

# Features:

* + - Search and filter flights o Book tickets with user login o Admin panel for managing flights o Email confirmation and booking history

# Architecture

* + **Frontend:**

Built with React.js using components, hooks, and React Router for navigation. UI powered by Material-UI.

# Backend:

Developed using Node.js and Express.js. RESTful API handles all CRUD operations related to users, flights, and bookings.

# Database:

MongoDB stores user data, flight details, and booking records. Mongoose is used for schema design and data validation.

* **Frontend (React.js) Components**: o SearchForm.js: Handles user inputs (source, destination, dates).
  + FlightList.js: Displays ranked results from ML model.

**State Management**: Redux for shared state (user sessions, bookings).

* **Backend (Node.js/Express.js) API Routes**: javascript app .get ('/api/flights', flightController .search);

app.post('/api/bookings', authMiddleware , bookingController.create); 

**Middleware**: JWT validation, rate limiting. **Database (MongoDB) Schemas**: javascript

const User = new Schema({ email: { type: String, unique: true },

bookings: [{ type: Schema.Types.ObjectId, ref: 'Booking' }]

});

# Setup Instructions

**Prerequisites:**

* + Node.js >= 18
  + MongoDB installed and running
  + npm or yarn **Installation:**
    - git clone https://github.com/your-username/FlightFinder.git o

cd FlightFinder o cd server o npm install o cd ../client o npm install

# Environment Variables:

Create a .env file in the /server folder with:

PORT=3000 MONGODB URI-

mongodb+srv://nehapriya:<db\_password>@cluster0.zghkpqk.mongodb.net/?retryWrit es=true&w=majority&appName=Cluster0

# Folder Structure

* + **Client:**

/client

├── /src

│ ├── /components

│ ├── /pages

│ ├── /api

│ ├── App.js

│ └── index.js

* + Server:

/server

├── /controllers

├── /routes

├── /models

├── server.js

└── .env **5.Running**

# the Application Frontend:

* + cd client
  + npm start **Backend:**
  + cd server
  + npm start

# API Documentation

* + **POST /api/auth/register**

Registers a new user.

Request Body: { name, email, password }

Response: { success, token }

* + **POST /api/auth/login** Logs in an existing user.

# GET /api/flights

Retrieves list of available flights.

# POST /api/bookings

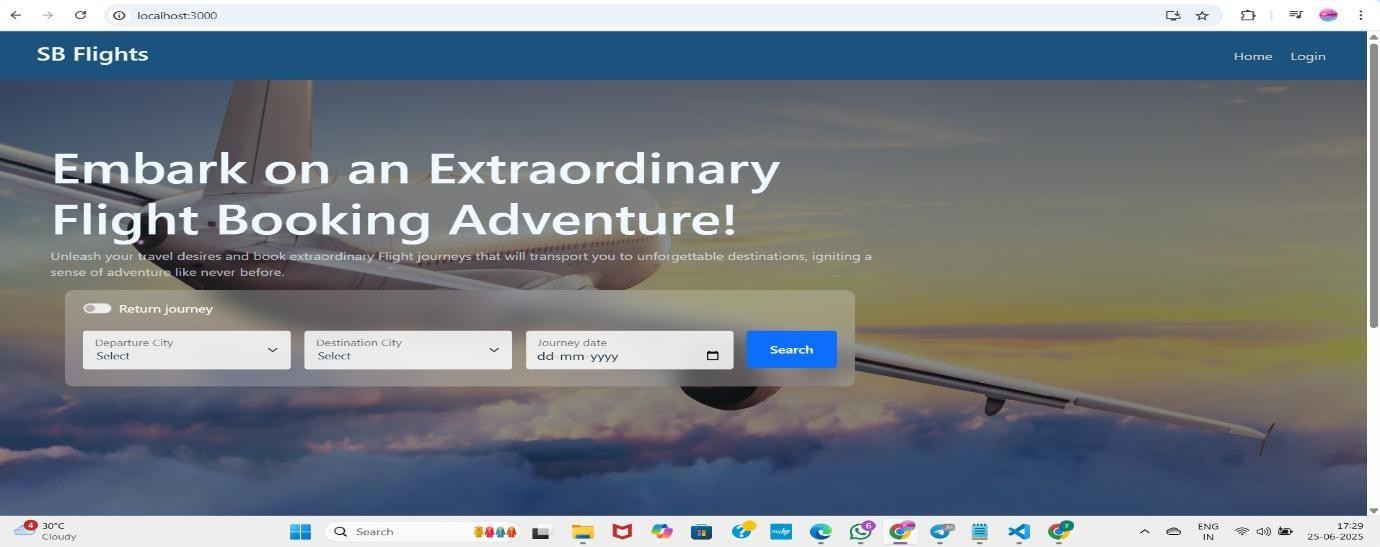
Books a flight for the logged-in user.

# Authentication

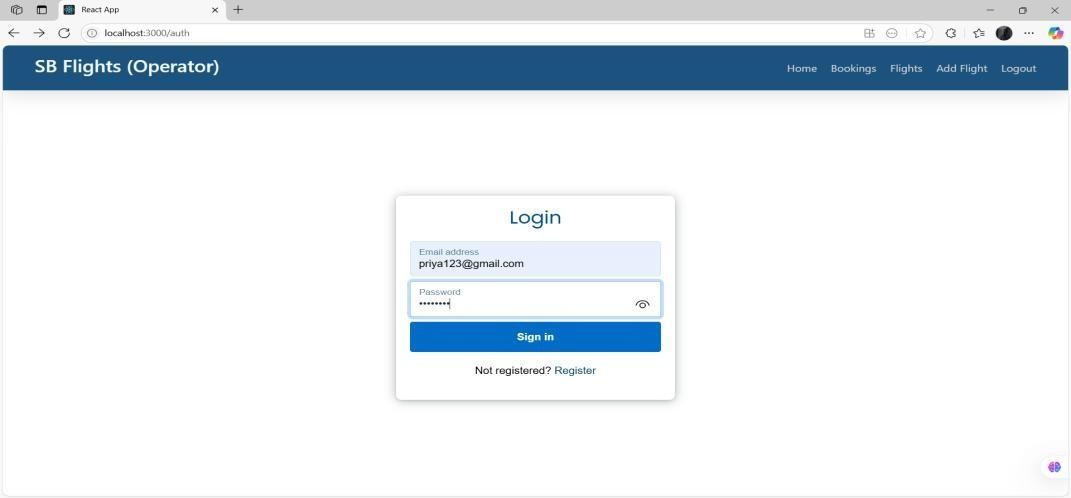
* + Implemented using JWT (JSON Web Token).
  + Tokens are stored in HTTP-only cookies.
  + Protected routes validate tokens before granting access.

# User Interface

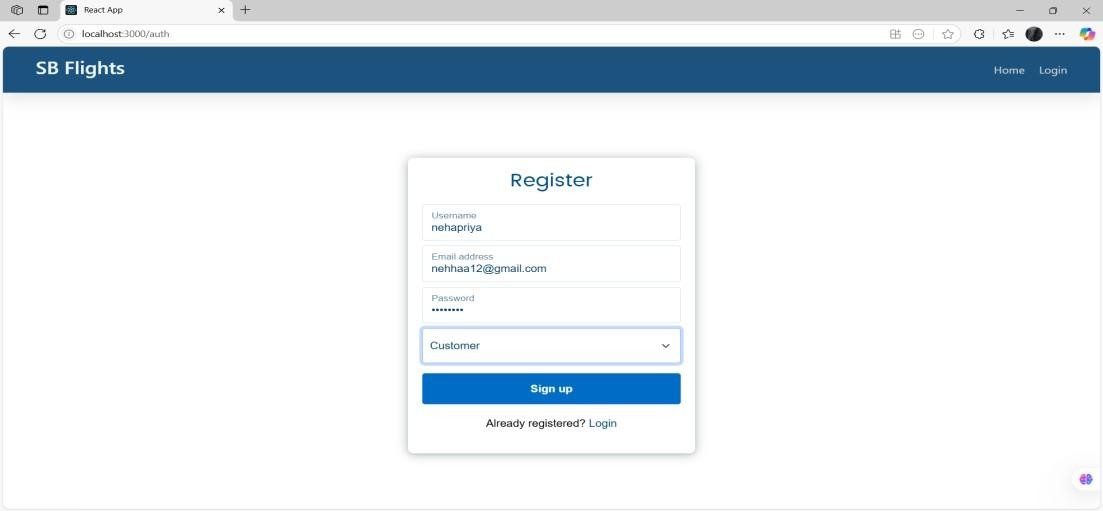
Home Page with flight search



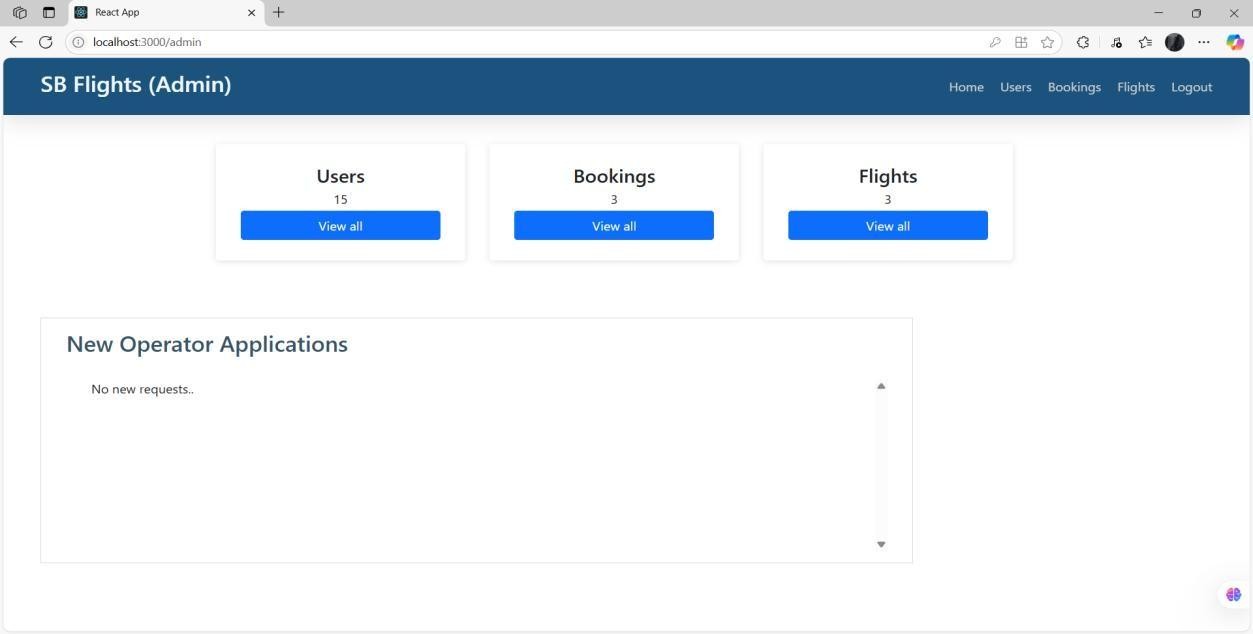
Login form



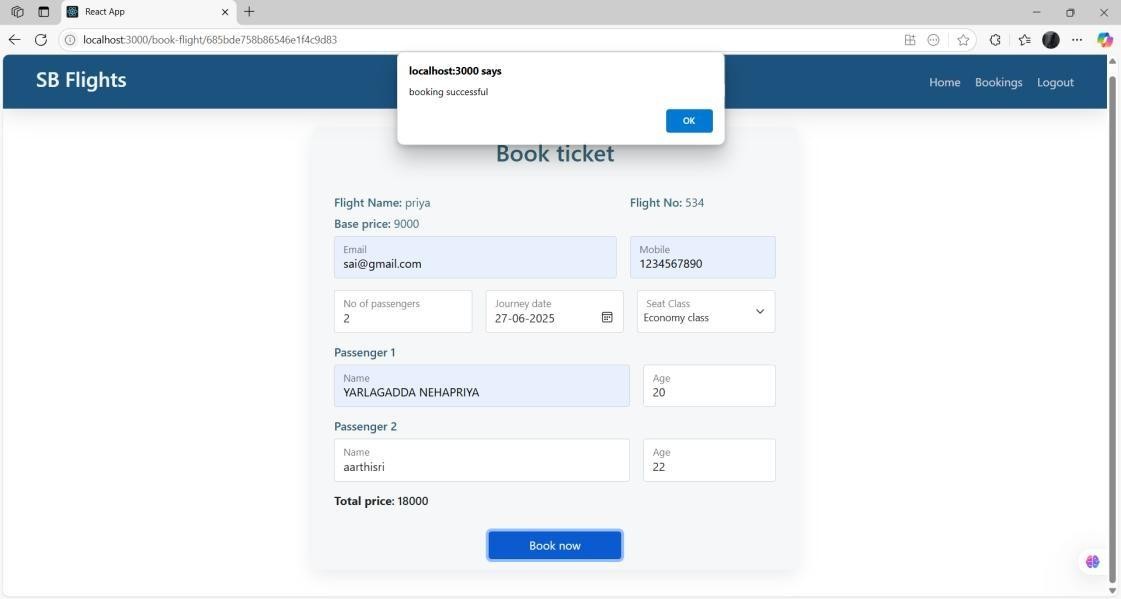
Register form

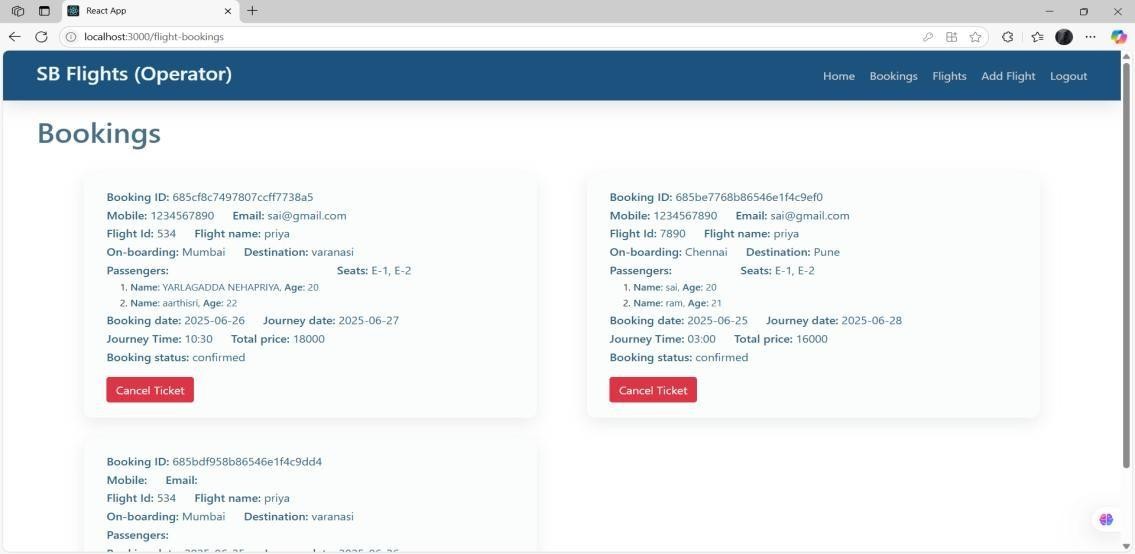


Admin dashboard



Booking confirmation page





# Testing

* + Manual testing on Postman for backend APIs
  + Basic component tests in React using Jest and React Testing Library

# Demo Link:

<https://drive.google.com/file/d/1pC1eQYezBeBYpmnp9M>[4ȠWzxmS1ZNfF/view?usp=sharing](https://drive.google.com/file/d/1pC1eQYezBeBYpmnp9M-4fjWzxmS1ZNfF/view?usp=sharing)

# drive link:

<https://drive.google.com/drive/folders/1lhr3vbwaqIPwEwIfraJw-G0kh9aA9WY0?usp=sharing>

# Known Issues

* + No password recovery implemented yet
  + UI responsiveness needs improvement on tablets

# Future Enhancements

* + Payment integration with Razorpay/Stripe
  + Role-based admin access
  + Real-time flight tracking integration