

FlightFinder: Navigating Your Air Travel Options

1. Introduction

Project Title: FlightFinder: Navigating Your Air Travel Options

Team Members:

- Shaik Khizar Ahmad
- Ravi Boddapati
- Shaik Yasmin
- Ramani Yandra

2. Project Overview

Purpose:

FlightFinder is a MERN-based web app designed to simplify the flight booking process for users. It helps travelers easily search, compare, and book flights that suit their schedule, preferences, and budget.

Features:

- User registration & login
- Flight search with filters (price, class, stops, airlines)
- Seat selection feature
- Booking summary & confirmation
- Admin dashboard for managing flights
- Secure payment gateway integration

3. Architecture

Frontend (React):

- React components for flight search, results display, booking form, seat selection, and user authentication.
- State management using React Hooks or Context API.
- Axios used for making HTTP requests to the backend.

Backend (Node.js & Express.js):

- RESTful API for user management, flight data, booking operations, and admin controls.
- Middleware for authentication (JWT) and error handling. - Routes: /api/users, /api/flights, /api/bookings, /api/admin

Database (MongoDB):

- Collections: Users, Flights, Bookings
- Schema design includes user profiles, flight schedules, and booking details.
- Mongoose ODM used for schema validation and queries.

4. Setup Instructions

Prerequisites:

- Node.js
- MongoDB
- Git

Installation: `git clone https://github.com/your-username/flightfinder.git`
`cd flightfinder`

Install backend:
`cd server`
`npm install`

Install frontend:
`cd client`
`npm install`

Add environment variables in .env:
`MONGO_URI=your_mongodb_uri JWT_SECRET=your_jwt_secret`

5. Folder Structure

Client (React):

```
client/ └──
src/
├── components/
├── pages/
├── App.js
└── index.js
```

Server (Node.js):

```
server/ └──
controllers/
├── routes/
├── models/
├── middleware/
server.js
```

6. Running the Application

Frontend: `cd client`
`npm start`
Backend: `cd server`
`npm start`

7. API Documentation

Endpoints:

- /api/users/register [POST]: Register a new user
- /api/users/login [POST]: User login
- /api/flights [GET]: Get all available flights
- /api/flights/:id [GET]: Get flight details
- /api/bookings [POST]: Create a new booking
- /api/admin/flights [POST]: Add a new flight (Admin only)

8. Authentication

Uses JWT-based Authentication for secure login sessions.

Tokens are stored in HTTP-only cookies for security.

Middleware used to protect private routes (e.g., bookings, admin panel).

9. User Interface

Clean and modern UI built with React and CSS.

Features include:

- Search bar with auto-complete
- Flight results with filter/sort
- Interactive seat selection map
- Responsive design

10. Testing

Tools Used:

- Postman for API testing
- Jest & React Testing Library (optional)
- Manual testing across browsers

11. Known Issues

- Payment gateway integration in testing mode
- Admin role protection needs multi-level access
- UI performance can be optimized on slow network

12. Future Enhancements

- Integration with real-time airline APIs
- Invoice generation and email notifications
- Multi-language support
- Native mobile app (React Native)