Flight Booking Application Documentation

Flight Booking Application Documentation

1. ER DIAGRAM OVERVIEW

The flight booking ER-diagram outlines the entities and their relationships within a flight booking system. Below is a summary:

Entities:

- User: Books flights and makes payments. One-to-many relation with bookings and payments.
- Booking: A record of a flight booked by a user, with specific details.
- Flight: Represents available flights with associated details and seat availability.
- Admin: Manages backend operations like bookings and flights.

Relationships:

- A User can make multiple Bookings and Payments.
- A Booking is associated with one Flight and one or more Passengers.
- The Admin oversees management and data operations.

2. PREREQUISITES FOR DEVELOPMENT

To develop the app using React.js, Node.js, and MongoDB, ensure the following:

Essential Tools:

- Node.js & npm: https://nodejs.org/en/download/
- MongoDB: https://www.mongodb.com/try/download/community
- Express.js: npm install express
- React.js: https://reactjs.org/docs/create-a-new-react-app.html
- Version Control (Git): https://git-scm.com/downloads

Development Environments:

- VS Code: https://code.visualstudio.com/download

- Sublime Text: https://www.sublimetext.com/download

- WebStorm: https://www.jetbrains.com/webstorm/download

3. PROJECT SETUP & STRUCTURE

Step 1: Clone the GitHub Repository:

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

Step 2: Install Dependencies:

```
npm install
```

Step 3: Start Development Server:

```
npm run dev
# or
npm run start
```

Visit: http://localhost:3000

Folder Structure:

- client/ - Frontend (React.js, Bootstrap, Axios)

- server/ - Backend (Node.js, Express.js, Mongoose)

4. APPLICATION FLOW

User:

- Registers account
- Searches for flights
- Books a flight
- Makes payments
- Cancels bookings

Admin:

- Adds and updates flights
- Manages bookings
- Monitors users

5. BACKEND DEVELOPMENT

Database Configuration:

- Set up with MongoDB Atlas or locally via MongoDB Compass
- Collections: users, bookings, flights, payments

Express.js Server:

- Middleware: body-parser, cors
- Routes: auth, bookings, flights, users
- Models: Mongoose schemas for each entity

User Authentication:

- Login/Signup routes

- Middleware to secure private routes

Admin Functions:

- Add/Update/Delete flights

- View bookings and users

Error Handling:

- Middleware for error capturing and custom messages

Reference Video: https://drive.google.com/file/d/11iNDCz0lJIv9zAS8nYbjrc1JXTPGtwsi/view?usp=drive_link

6. FRONTEND DEVELOPMENT

Login/Register:

- Form input for credentials

- Navigation to role-based dashboards (User/Admin/Flight Operator)

Flight Booking:

- Modal form for flight search (departure, destination)

- Display results with booking button

Booking History:

- Show past bookings

- Cancel booking option

Admin Dashboard:

- Add new flight form

- Update flight details

- View all flights/bookings/users

7. FINAL IMPLEMENTATION & DEMO

Once the development is complete:

- Run the server and client

- Test functionalities thoroughly

Demo Video: https://drive.google.com/file/d/1Q0XwKtAz7EkaKNJv3_gbo6mZE9nfuBTK/view?usp=sharing

UI Snapshots:

- Landing Page

- User Authentication

- Bookings Page

- Admin Dashboard

- All Users View

- Flight Operator View

- New Flight Entry Page

End of Document