# Final Report

1. INTRODUCTION

1.1 Project Overview

FlightFinder is a comprehensive full-stack web application developed to streamline and enhance the air travel booking experience. The application serves as a digital flight booking platform that allows users to effortlessly search for, filter, and book flights based on their preferences. It also incorporates robust features for authentication, user profile management, booking history, and administrative controls. By leveraging the MERN (MongoDB, Express.js, ReactJS, Node.js) technology stack, the application ensures responsive design, seamless data handling, and efficient backend operations.

1.2 Purpose

The primary aim of FlightFinder is to bridge the gap between complex flight booking systems and the need for a simple, efficient, and user-focused experience. The platform enables users to complete their entire booking journey — from searching for flights to confirming a booking — through a single intuitive interface. Additionally, administrative functionalities ensure streamlined management of flights and user data.

2. IDEATION PHASE

2.1 Problem Statement

Many existing flight booking platforms lack intuitive design, personalized filters, and real-time performance. Users often face difficulties in narrowing down options, especially when they have specific preferences like layovers, price limits, or class type. Furthermore, admins need an efficient system to manage bookings and flights dynamically.

2.2 Empathy Map Canvas

- Think & Feel: Users want a quick, secure, and reliable booking experience.
- Hear: Complaints about overly complex booking systems and limited flight filter options.
- See: Numerous listings without relevance to preferences.
- Say & Do: Ask for direct flights, look for price comparisons, try alternative platforms.
- Pain: Complicated UI, lack of control, poor filter mechanisms.
- Gain: Personalized results, easy interface, fast bookings.

2.3 Brainstorming

- Integration of secure user authentication and booking history.
- Filter-based flight search (e.g., date, location, class, price, layover).
- Admin panel for managing flights and bookings.

- Modular architecture using MERN stack for scalability.
- Use of Mongoose models for robust MongoDB schema handling.

## 3. REQUIREMENT ANALYSIS

### 3.1 Customer Journey Map

1. Visitor lands on homepage
2. Registers or logs in securely
3. Enters travel details (departure, arrival, date, class)
4. Browses available flight options with filters
5. Selects and confirms a flight booking
6. Admin logs in to monitor and manage flight entries and user data

### 3.2 Solution Requirement

- Frontend Requirements: Login/Signup forms, Search and booking forms, Responsive UI components using React
- Backend Requirements: RESTful APIs, Middleware for authentication (JWT), Mongoose schemas
- Admin Requirements: Add/edit/delete flight listings, Manage user accounts and bookings

### 3.4 Technology Stack

- Frontend: ReactJS, Bootstrap, Axios
- Backend: Node.js, Express.js
- Database: MongoDB (via Mongoose ODM)
- Version Control: Git, GitHub
- Development Tools: Visual Studio Code, Postman

## 4. PROJECT DESIGN

### 4.1 Problem-Solution Fit

FlightFinder focuses on streamlining the entire journey for both end users and admins by providing a minimalistic UI and applying targeted filters.

### 4.2 Proposed Solution

- User authentication (JWT-based)
- Dynamic flight filtering
- Booking confirmations and history
- Admin control panel for flight management

### 4.3 Solution Architecture

Frontend includes login, booking, admin dashboard. Backend uses ExpressJS for routing and MongoDB for storing users, bookings, flights.

## 5. PROJECT PLANNING & SCHEDULING

Setup structure
Dependencies
Schema creation
React development

Integration
Testing and debugging

## 6. FUNCTIONAL AND PERFORMANCE TESTING

APIs tested with Postman
Authentication tested
Frontend rendering within 300ms

## 7. RESULTS

- UI Screens: Login, Admin panel, Bookings
- ER Diagram and application flow
- Code structures and folder hierarchy

## 8. ADVANTAGES & DISADVANTAGES

Advantages: Scalable, real-time filtering, modular
Disadvantages: No real API, No payment gateway

## 9. CONCLUSION

FlightFinder is an efficient and practical application with a scalable architecture to handle real-world booking scenarios.

## 10. FUTURE SCOPE

- Integrate third-party APIs
- Payment gateway
- Mobile support
- Real-time notifications

## 11. APPENDIX

- Source code:

GitHub : https://github.com/LuqmanAftab/Flight-Booking-Application-using-MERN-stack