FlightFinder: Navigating Your Air Travel Options

1. INTRODUCTION

1.1 Project Overview

FlightFinder is a web-based or mobile application that allows users to search, compare, and book airline tickets across multiple providers. It aims to provide a seamless, user-friendly experience for budget-conscious and time-sensitive travelers.

1.2 Purpose

The purpose of this project is to simplify the process of air travel planning by integrating various airline APIs and presenting real-time data, allowing users to make informed decisions regarding flight prices, durations, layovers, and amenities.

---

2. IDEATION PHASE

2.1 Problem Statement

Travelers often struggle to find the best flights due to fragmented information across various booking sites. There's a need for a centralized platform that aggregates, compares, and filters flights effectively.

2.2 Empathy Map Canvas

- **Think & Feel**: Wants a reliable, affordable flight with minimal hassle.
- **See**: Sees confusing options, inconsistent pricing.
- **Hear**: "Prices are always changing," "Try using different apps."
- **Say & Do**: Compares 3–4 platforms, bookmarks options.
- **Pain**: Time-consuming search, unexpected fees, poor UX.
- **Gain**: Finds a good deal quickly, trust in the platform.

2.3 Brainstorming

- Flight price comparison
- API integration (Amadeus, Skyscanner, etc.)
- Filter options (baggage, layovers, time, price)
- Booking reminders or price alerts
- Simple, mobile-first UI

---

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

Stages: **Awareness → Search → Compare → Select → Book**
Touchpoints: Website/App, Notifications, Support
Pain Points: Price fluctuation, decision fatigue

3.2 Solution Requirement

- Functional Requirements: Flight search, filter, compare, sort, book
- Non-functional Requirements: Fast response, mobile-responsive, secure data handling

3.3 Data Flow Diagram (DFD)

Level 0: User → FlightFinder → Flight APIs → Results to User
Level 1: Search Request → Aggregator → Filter Module → Display Results

3.4 Technology Stack

- **Frontend**: React / Flutter / HTML-CSS-JS
- **Backend**: Node.js / Python Flask
- **API**: Amadeus / Skyscanner / Kiwi API
- **Database**: MongoDB / PostgreSQL
- **Deployment**: Heroku / Vercel / AWS

---

4. PROJECT DESIGN

4.1 Problem-Solution Fit

Problem: Users are overwhelmed by inconsistent flight info
Solution: One-stop aggregator platform offering clarity, filters, and real-time pricing

4.2 Proposed Solution

FlightFinder offers a modern interface with an integrated backend to fetch, display, and compare flight options. Includes user authentication and optional booking reminders.

4.3 Solution Architecture

- User Interface
- Middleware/API Aggregator
- External APIs (Flight Data)
- Database (User data, search history)

---

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

- Week 1–2: Ideation & Requirement Analysis
- Week 3–4: UI Design & API Research
- Week 5–6: Backend Development
- Week 7: Integration & Testing
- Week 8: Deployment & Documentation

---

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

- Tested API response time under load
- UI load time benchmarked under slow network
- Memory and CPU usage tested during batch search

---

7. RESULTS

7.1 Output Screenshots

- Home page
- Flight search results
- Filter panel
- Booking summary

---

8. ADVANTAGES & DISADVANTAGES

Advantages

- Real-time data aggregation
- Easy filtering and sorting
- Mobile-friendly interface

Disadvantages

- Reliance on third-party APIs
- No direct booking (redirects to partner sites)
- May require frequent API key renewal

---

9. CONCLUSION

FlightFinder successfully addresses a real-world problem by offering a centralized, efficient, and intuitive flight comparison tool. It enhances user experience by reducing the time and effort required in booking travel.

---

10. FUTURE SCOPE

- Add hotel and car rental options
- Implement price prediction with ML
- Personalized recommendations
- Chatbot integration for support
