

## **1.INTRODUCTION**

**PROJECT TITLE:** Flight Booking App

**TEAM MEMBERS:**

Karunya A – Backend Implementation  
Juliet Mary A – Frontend Implementation  
Sumithra V– Framework Development  
Sujitha P – Package and Command-Line Setup

## **2.PROJECT OVERVIEW**

**Purpose:** The flight booking app aims to make travel planning seamless, efficient, and personalized. It offers users a convenient way to search, compare, and book flights while providing travel information, deals, and customer support.

**Features:**

User-friendly navigation and search filters.

Real-time flight availability and pricing.

Multiple payment methods.

Flight tracking and notifications.

Personalized travel recommendations.

Integration with loyalty programs and travel insurance.

## **3.ARCHITECTURE**

**Frontend:**

/assets: Static assets (images, icons).

/components: Reusable UI components.

/pages: Pages for routes like search results, flight details, checkout, etc.

/services: API service functions for interacting with the backend.

/styles: Global styles and theming.

/utils: Helper functions (e.g., date formatting).

App.js: Main component housing routing and layout.

Index.js: Entry point for the app.

### **Backend:**

/config: Configuration for the database and environment variables.

/controllers: Handles API requests for flight bookings, user data, etc.

/models: Defines database schemas for flights, bookings, and users.

/routes: API endpoints (e.g., search, book flights).

/services: Core business logic (e.g., availability check, booking).

/middleware: Security (authentication, rate limiting).

/utils: Utility modules (email service, payment processing).

App.js: Express app setup.

Server.js: Server entry point.

### **Database:**

Schemas for users, flights, bookings, payments, and loyalty programs ensure smooth app functionality.

## **4.SETUP INSTRUCTION**

### **Prerequisites:**

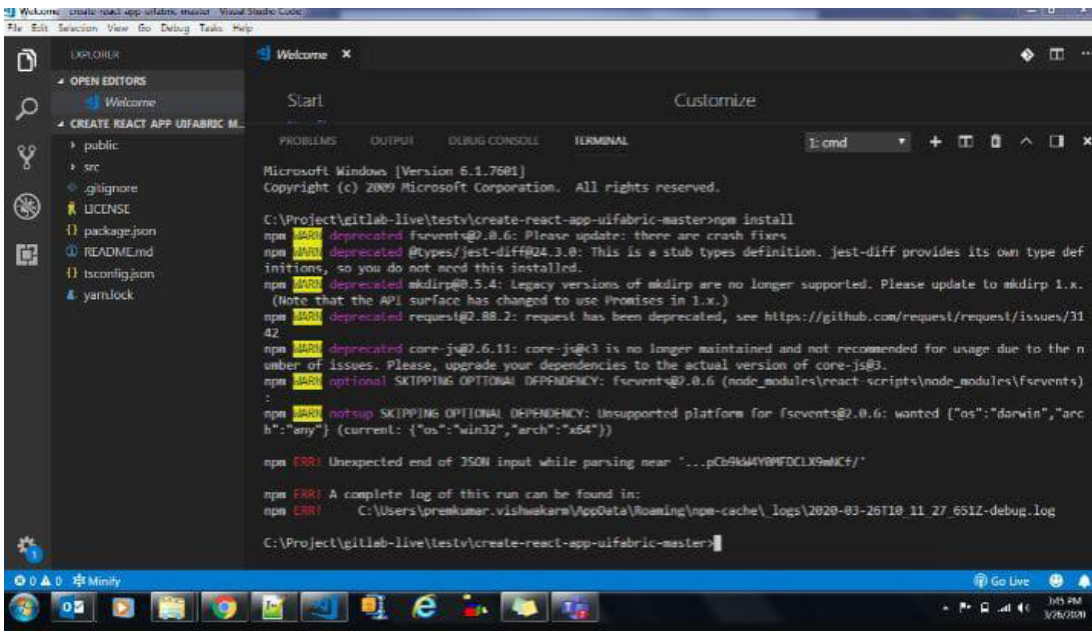
JavaScript, Node.js, React.js, MongoDB.

## Installation:

Install dependencies using npm.

Set up React for the frontend and Node.js for the backend.

Configure MongoDB for the database.



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Project\gitlab-live\test\create-react-app-uifabric-master>npm install
npm WARN deprecated fsevents@2.0.6: Please update: there are crash fixes
npm WARN deprecated @types/jest-diff@24.3.0: This is a stub types definition. jest-diff provides its own type definitions, so you do not need this installed.
npm WARN deprecated mkdirp@0.5.4: Legacy versions of mkdirp are no longer supported. Please update to mkdirp 1.x. (Note that the API surface has changed to use Promises in 1.x.)
npm WARN deprecated request@2.88.2: request has been deprecated, see https://github.com/request/request/issues/3142
npm WARN deprecated core-js@2.6.11: core-js@3 is no longer maintained and not recommended for usage due to the number of issues. Please, upgrade your dependencies to the actual version of core-js@3.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@2.0.6 (node_modules\react-scripts\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@2.0.6: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

npm ERR! Unexpected end of JSON input while parsing near "...pCt9k3MY0WFDCLX9mKt/"

npm ERR! A complete log of this run can be found in:
npm ERR!   C:\Users\premkumar.vishwakara\AppData\Roaming\npm-cache\_logs\2020-03-26T19_11_27_651Z-debug.log

C:\Project\gitlab-live\test\create-react-app-uifabric-master>
```

## 5.FOLDER STRUCTURE

### Client:

SearchFlightsPage.js: Search bar and flight filters.

FlightDetailsPage.js: Flight info and booking form.

CartPage.js: Review selected flights.

LoginPage.js: User authentication.

ProfilePage.js: View and manage bookings.

PaymentPage.js: Complete payments.

#### **Server:**

/config/db.js: MongoDB setup.

/controllers/flightController.js: Flight search and booking logic.

/routes/flightRoutes.js: API endpoints for flights.

/middleware/authMiddleware.js: JWT verification.

/utils/paymentGateway.js: Payment handling.

### **6.RUNNING THE APPLICATION**

**Frontend:** Run npm start in the client directory.

**Backend:** Run npm start in the server directory.

### **7.API DOCUMENTATION**

#### **User APIs:**

**Register User:** POST /api/auth/register – Register a new user.

**Login User:** POST /api/auth/login – Authenticate and return a JWT.

#### **Flight APIs:**

**Search Flights:** GET /api/flights – Retrieve flight options based on criteria.

**Get Flight by ID:** GET /api/flights/:id – Retrieve flight details.

**Book Flight:** POST /api/flights/:id/book – Book a selected flight.

#### **Payment APIs:**

**Initiate Payment:** POST /api/payments – Handle flight payment processing.

## **8.AUTHENTICATION**

Use JWT for user authentication to ensure secure sessions and access control.

## **9.USER INTERFACE**

**Flight Search:** Search and filter flights.

**Flight Details:** View complete flight info.

**Booking Summary:** Display selected options and total costs.

**Profile:** Manage account and booking history.

## **10.TESTING**

Unit tests for flight search, booking, and payments.

Integration tests for frontend-backend interaction.

API testing for proper responses and validations.

Load testing to ensure performance during peak usage.

## **11.KNOWN ISSUES**

Real-time price fluctuation handling.

UI bugs on smaller devices.

Payment gateway API errors.

## **12.FUTURE ENHANCEMENT**

**AI Travel Assistant:** Offer real-time travel recommendations.

**Multi-City Bookings:** Add support for complex itineraries.

**Integration:** Include hotel bookings and car rentals.

**Offline Access:** Enable saving itineraries offline.

