# Project Design Phase-II
# Technology Stack (Architecture & Stack)

| | |
|---|---|
| Date | 13 April 2025 |
| Team ID | SWTID1743512004 |
| Project Name | FlightFinder |
| Maximum Marks | 4 Marks |

## Technical Architecture:

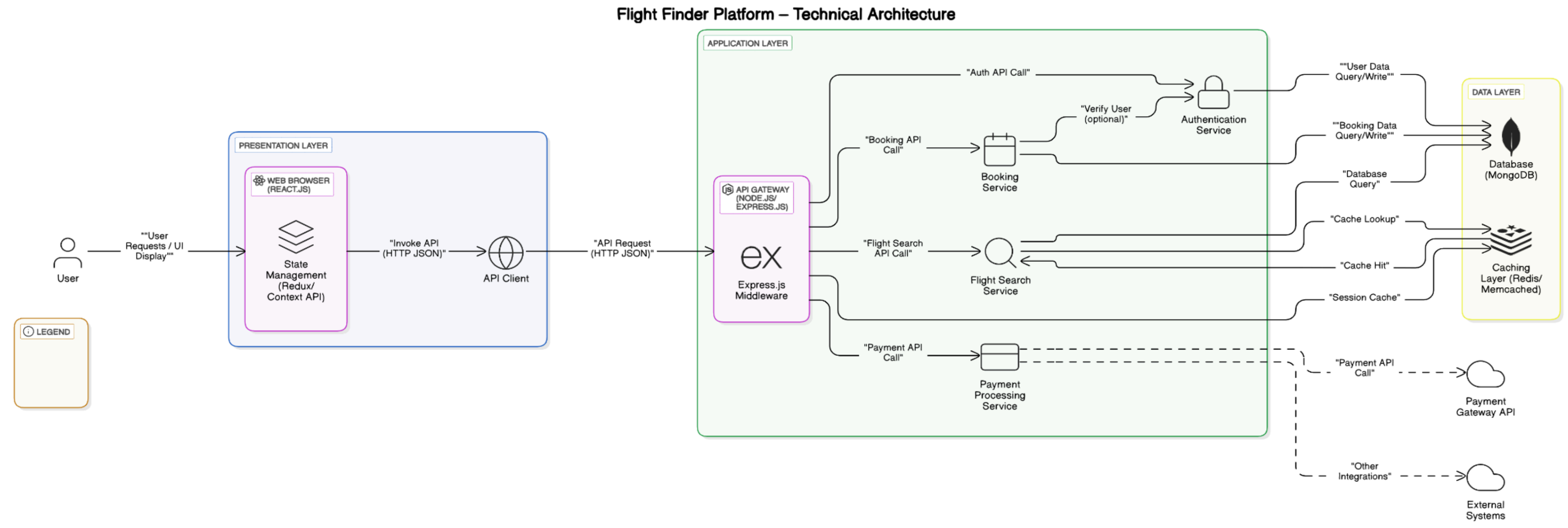

**Table-1 : Components & Technologies:**

| S.No | Component | Description | Technology |
|---|---|---|---|
| 1. | User Interface | How users interact with the application (Web UI). | HTML, CSS, JavaScript, React.js |
| 2. | Application Logic - Backend | Logic for handling user requests, business rules, and data processing. | Node.js with Express.js |
| 3. | Authentication Service | Handles user registration, login, and session management. | Node.js (using libraries like Passport, JWT) |
| 4. | Flight Search Engine | Logic for processing flight search queries and retrieving relevant data. | Node.js (interacting with the database) |
| 5. | Database | Stores application data such as user accounts, flight information, bookings. | MongoDB |
| 6. | Payment Gateway Integration | Payment Gateway Integration | Integration with Stripe, PayPal, or similar |
| 7. | Email Service Integration | For sending registration confirmations, booking notifications, etc. | SendGrid, Mailgun, or similar |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table-2: Application Characteristics:**

| S.No | Characteristics | Description | Technology |
|---|---|---|---|
| 1. | Flight Search & Filtering | Allows users to search for flights based on departure city, arrival city, departure date, and potentially other criteria like airline and price range. Filtering options enable users to refine their search results. | React (Frontend), Node.js (Backend), potentially database query languages. |
| 2. | Real-time Availability | Displays up-to-date information on the number of seats available for each flight. This ensures users see accurate booking possibilities. | Node.js (Backend), Database with real-time update capabilities (details not specified in the provided snippets). |
| 3. | User Authentication & Authorization | Securely authenticates users for login and manages different user roles (e.g., customer, admin) with varying levels of access and permissions. | Node.js (Backend) with authentication libraries (details not specified in the provided snippets). |
| 4. | Booking Management | Enables users to select flights, enter passenger details, and create bookings. The system records booking information, including user, flight, number of passengers, and total price. | React (Frontend), Node.js (Backend), Database (MongoDB or similar based on typical Node.js setups, though not explicitly stated). |
| 5. | Data Persistence | Stores information about flights, users, and bookings in a database for retrieval and management. This includes flight details, user profiles, and booking records. | Database (MongoDB or similar based on typical Node.js setups, though not explicitly stated). |
| 6 | API Integration (Internal) | The frontend and backend communicate via a defined set of API endpoints to exchange data related to flight searches, user authentication, and booking operations. | RESTful APIs (likely implemented using Express.js with Node.js). |

**References:**

https://c4model.com/

https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/

https://www.ibm.com/cloud/architecture

https://aws.amazon.com/architecture

https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d