**Project Design Phase-II**

**Data Flow Diagram & User Stories**

|  |  |
| --- | --- |
| Date | 30-06-2025 |
| Team ID | LTVIP2025TMID43096 |
| Project Name | Flight Finder: Navigating Your Air Travel Options |
| Maximum Marks | 4 Marks |

**Data Flow Diagrams:**

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

**The data flow of the Flight Booking System follows a clear and structured client-server model using the MERN stack (MongoDB, Express.js, React.js, Node.js). Here’s how data flows through the application:**

**1. User Interaction (Frontend - React)**

* **Users (Customers, Admins, or Flight Operators) interact with the application via a web-based UI built using React.**
* **The frontend captures input (like registration details, search queries, flight data, booking info) and sends it to the backend through HTTP requests using Axios.**

**2. API Requests (Backend - Node.js + Express.js)**

* **The Express.js server listens for incoming API calls from the frontend.**
* **It validates the request, applies business logic, and interacts with the MongoDB database through Mongoose models.**
* **Examples:**
  + **/register → Registers a new user.**
  + **/search-flights → Retrieves flights matching criteria.**
  + **/book-ticket → Books a flight and stores booking details.**

**3. Database Operations (MongoDB)**

* **MongoDB stores all persistent data including:**
  + **Users (customers, admins, operators)**
  + **Flights (added by operators)**
  + **Bookings (made by customers)**
* **Backend performs CRUD operations on these collections via Mongoose schemas.**
* **Each schema ensures the integrity and format of the data.**

**4. Response Handling (Backend to Frontend)**

* **After completing the database operation, the backend sends a JSON response back to the React frontend.**
* **The frontend uses this data to:**
  + **Show confirmation (e.g., "Booking Successful")**
  + **Update UI dynamically (e.g., available flights, booking status)**

**5. Role-Based Flow**

* **Customer:**
  + **Registers/logs in → Searches → Books flight → Views/cancels bookings.**
* **Flight Operator:**
  + **Waits for approval → Adds flights → Views bookings related to their flights.**
* **Admin:**
  + **Approves/rejects operators → Monitors users/bookings.**

**Summary**

**User (React)**

**⬇️ request (Axios)**

**Backend (Node + Express)**

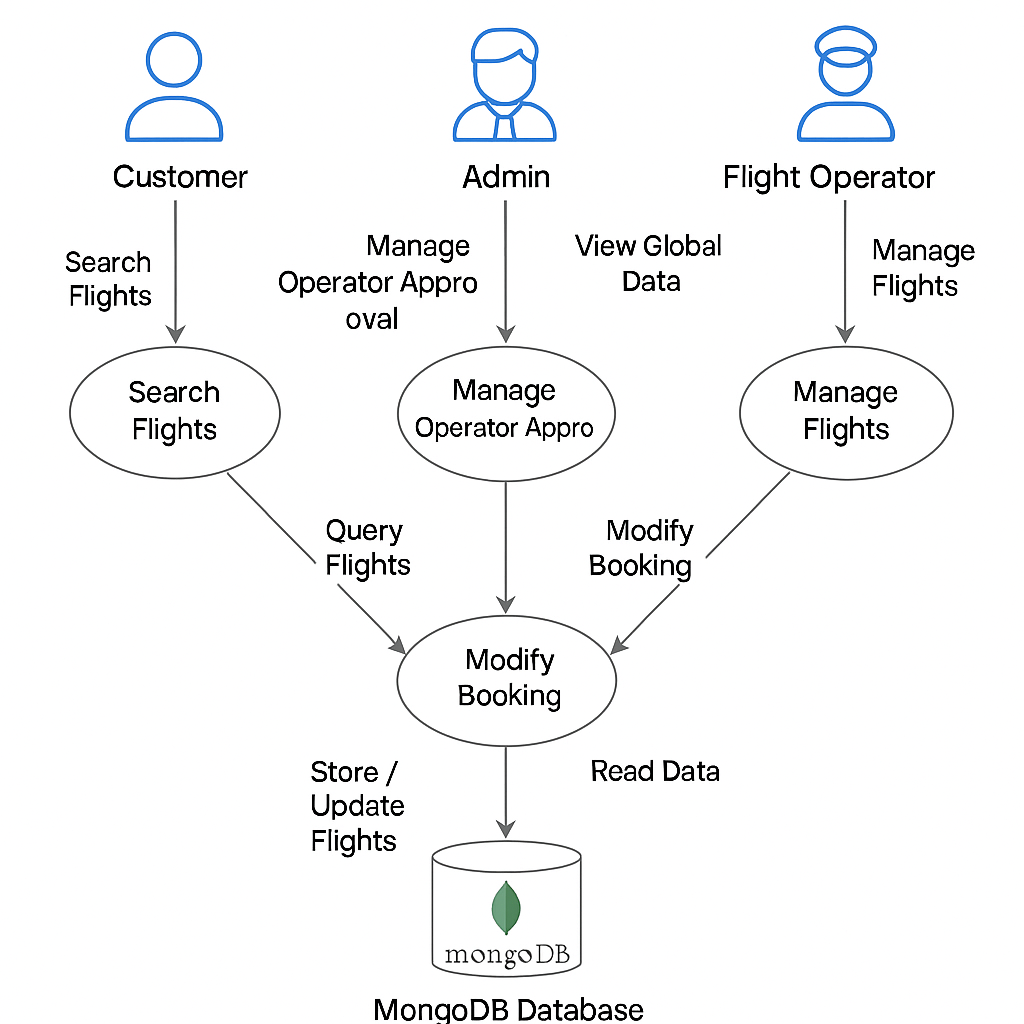
**⬇️/⬆️**

**Database (MongoDB via Mongoose)**

**⬆️ response (JSON)**

**React UI updates dynamically**

**This architecture ensures a seamless, scalable, and role-based flow of data between the frontend and backend while maintaining real-time interactivity and persistence.**

****

**User Stories**

Use the below template to list all the user stories for the product.

| **User Type** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Acceptance criteria** | **Priority** | **Release** |
| --- | --- | --- | --- | --- | --- | --- |
| Customer | Registration | USN-1 | As a customer, I can register by providing my name, email, password, and role. | I receive confirmation and can log in. | High | Sprint-1 |
| Customer | Login | USN-2 | As a customer, I can log in using my registered credentials. | Successful login redirects to customer dashboard. | High | Sprint-1 |
| Customer | Search Flights | USN-3 | As a customer, I can search available flights by date, origin, and destination. | I see a list of flights matching my search. | High | Sprint-2 |
| Customer | Book Tickets | USN-4 | As a customer, I can book tickets by selecting a flight and entering passenger details. | I receive a booking confirmation. | High | Sprint-2 |
| Customer | View/Cancel Bookings | USN-5 | As a customer, I can view and cancel my bookings. | I can see my bookings and cancel them if needed. | Medium | Sprint-1 |
| Flight Operator | Register | USN-6 | As a flight operator, I can register and wait for admin approval. | I see "Awaiting Approval" message. | High | Sprint-2 |
| Flight Operator | Add Flight | USN-7 | As a flight operator, I can add a new flight with route, timing, and seat info. | Flight is added and shown under "Your Flights". | High | Sprint-1 |
| Flight Operator | View Bookings | USN-8 | As a flight operator, I can view bookings related to my flights. | List of bookings is displayed correctly. | Medium | Sprint-2 |
| Admin | Login | USN-9 | As an admin, I can log in to the admin dashboard. | Admin dashboard is displayed. | High | Sprint-2 |
| Admin | Approve/Reject Operators | USN-10 | As an admin, I can approve or reject flight operator registrations. | Operator's status updates as approved/rejected. | High | Sprint-1 |
| Admin | Manage Flights | USN-11 | As an admin, I can view and manage all flight records. | Flights data is shown and editable. | Medium | Sprint-3 |
|  | | | | | | |