**Project Design Phase**

**Solution Architecture**

|  |  |
| --- | --- |
| Date | 23 June 2025 |
| Team ID | LTVIP2025TMID20459 |
| Project Name | Docspot |
| Maximum Marks | 4 Marks |

To provide a user-friendly, secure, and scalable healthcare appointment booking system that bridges the gap between patients and healthcare providers.

* Effortless Appointment Booking.
* End-to-End Patient Management.
* Scalable user authentication and authorization.
* Secure and trackable payment transactions.
* Reliable real-time chat and notifications.

### **Architecture Style:**

### DocSpot adopts a **modular client-server architecture**, based on the **MERN stack** (MongoDB, Express.js, React.js, Node.js), ensuring a scalable, maintainable, and efficient system for handling real-time healthcare bookings.

### **2. Components Breakdown**

* **Frontend (React.js + UI Libraries):**
  + Role-based dashboards for patients, doctors, and admins
  + Axios for REST API calls
  + Responsive design with Material UI, Bootstrap, and Ant Design
  + Routing handled via React Router
* **Backend (Node.js + Express.js):**
  + RESTful APIs for all user actions (booking, login, updates)
  + Authentication using JWT/session tokens
  + Logic for slot locking, status updates, and doctor approvals
  + Multer middleware for file uploads
* **Database (MongoDB + Mongoose):**
  + Collections: Users, Appointments, Schedules, Reviews
  + Flexible schema for handling user roles and metadata
  + Indexed queries for faster doctor search and booking retrieval
* **Notification System:**
  + Integrated with email/SMS APIs (e.g., SendGrid, Twilio)
  + Triggers on booking, confirmation, cancellation, and reminders

### **3. Data Flow Overview**

**Patient Action (e.g., booking) → Frontend Form → Axios API Call → Express Backend → MongoDB** → Server Response → UI Update with Confirmation + Notification

### **4. Deployment Environment**

* **Local Development:** localhost:3000 for client, localhost:5000 for server
* **Production Deployment:** Hosted on cloud platforms like **Render** or **Vercel**
* **Codebase Structure:** /frontend and /backend with isolated dependencies and startup commands

### **5. Key Qualities Ensured:**

* **Scalability:** Easily handles growing users and doctors with modular backend routing and indexed data queries
* **Security:** Role-based authentication, file filters, protected routes
* **Maintainability:** Clearly separated components with REST APIs and reusable React components
* **User Experience:** Fast loading dashboards, toast notifications, and intuitive layout

