**Project Design Phase**

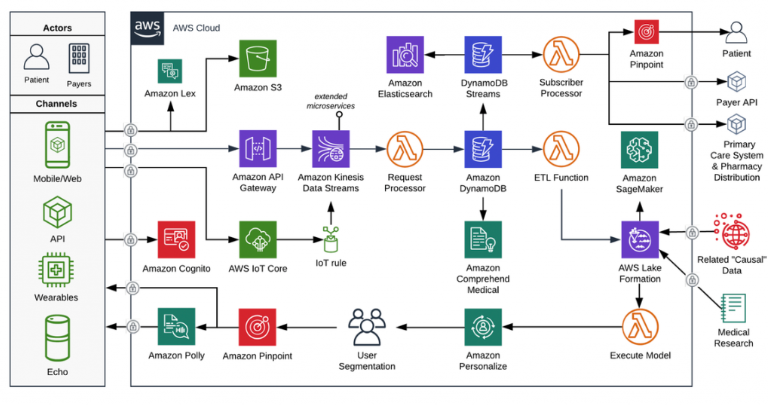
**Solution Architecture**

|  |  |
| --- | --- |
| Date | 27-06-2025 |
| Team ID | LTVIP2025TMID46688 |
| Project Name | DocSpot |
| Maximum Marks | 4 Marks |

**Solution Architecture – DocSpot**

**Overview**

The solution architecture outlines the structure, components, and interactions of the DocSpot system. It helps visualize how different parts of the platform communicate and work together to deliver a seamless appointment booking experience. This architecture is designed to be modular, scalable, and secure, using the MERN stack (MongoDB, Express.js, React.js, Node.js) along with JWT-based authentication.



**System Components**

1. **Frontend (React.js):**
   * User-friendly interface for users, doctors, and admin
   * Handles registration, login, dashboard views, and appointment interactions
   * Bootstrap used for responsive UI
2. **Backend (Node.js + Express.js):**
   * Handles API requests, routing, and server-side logic
   * Connects to MongoDB for data storage
   * Implements authentication, authorization, and role-based access
3. **Database (MongoDB):**
   * Stores user data, doctor applications, appointments, and admin records
   * NoSQL structure provides flexibility and scalability
4. **Authentication (JWT):**
   * Ensures secure login and route protection
   * Provides role-based access for users, doctors, and admins

**Data Flow Diagram (Simplified)**

**User / Doctor / Admin Interface** (React)  
⬇️  
**API Calls** → **Express Server (Node.js)**  
⬇️  
**MongoDB** (Data stored and fetched)  
⬆️  
**Responses sent back to frontend**

**Security Layer**

* JWT-based token authentication
* Role-based route protection (user, doctor, admin)
* Data validation and error handling

**Benefits of This Architecture**

* **Modular & Reusable**: Easy to scale or add new features
* **Secure**: JWT and role-based access ensures safety
* **Responsive**: Frontend handles mobile and desktop views
* **Seamless Experience**: Clear separation of concerns between frontend and backend