**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

**Date:** 31 January 2025  
**Team ID:** LTVIP2025TMID54470  
**Project Name:** DocSpot: Seamless Appointment Booking for Health  
**Maximum Marks:** 4 Marks

**✅ Technical Architecture Overview**

DocSpot follows a **client-server architecture** utilizing the **MERN stack** for seamless full-stack development. The application includes distinct modules for **patients, doctors, and admins** with real-time appointment scheduling, verification, and notifications. It is hosted on a cloud-based environment for better scalability and uptime.

*(Architecture diagram placeholder)*  
*→ A visual flow showing Patient UI ↔ Backend API ↔ MongoDB with role-based access (Patient, Doctor, Admin)*

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

| **S.No** | **Component** | **Description** | **Technology** |
| --- | --- | --- | --- |
| 1 | User Interface | Web-based UI for Patients, Doctors, Admins | HTML, CSS, JavaScript, React.js |
| 2 | Application Logic-1 | Handles business logic and routing | Node.js, Express.js |
| 3 | Application Logic-2 | Real-time communication for booking updates | WebSockets / Socket.IO |
| 4 | Application Logic-3 | Email confirmation and notification logic | Nodemailer, Cron Jobs |
| 5 | Database | Stores user info, doctor profiles, appointments, records | MongoDB (NoSQL) |
| 6 | Cloud Database | Cloud-hosted NoSQL DB instance | MongoDB Atlas |
| 7 | File Storage | Storage for prescriptions, reports, and uploads | Cloudinary / Local FileSystem |
| 8 | External API-1 | Email communication service | Gmail API / SendGrid |
| 9 | External API-2 | Optional verification (if used for Aadhaar/telemedicine) | Aadhaar API / Digilocker (optional) |
| 10 | Machine Learning Model | [Optional future extension] For symptom-based triage | TensorFlow.js or external ML API |
| 11 | Infrastructure | Deployment environment | Vercel (Frontend), Render/Heroku (Backend), MongoDB Atlas (DB) |

**✅ Table-2: Application Characteristics**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
| 1 | Open-Source Frameworks | Frontend and Backend frameworks used | React.js, Node.js, Express.js, MongoDB |
| 2 | Security Implementations | Data encryption, authentication, and secure routing | HTTPS, JWT (JSON Web Tokens), bcrypt, CORS, Helmet.js, OWASP Guidelines |

Let me know if you'd like:

* A **system architecture diagram** image generated.
* This section inserted directly into your document.
* An export in .docx or .pdf format.

Ready for the next part whenever you are!