Requirement Gathering and Analysis Phase

Date: 15/04/2025

Team ID: SWTID1743520385

Project Name: Book a Doctor Platform

Maximum Marks: —

# Technical Architecture

The Deliverable shall include the architectural diagram and the supporting technology stack as per the following tables. This document outlines how the Book a Doctor system is designed using open-source technologies, RESTful APIs, and a scalable 3-tier architecture deployed on modern cloud platforms.  
  
📌 Note: Architecture diagram to be added separately using tools like Lucidchart, draw.io, or MS Visio.

## Table 1: Components & Technologies

|  |  |  |  |
| --- | --- | --- | --- |
| S.No | Component | Description | Technology |
| 1 | User Interface | Web UI for doctor/patient/admin access | HTML, CSS, JavaScript, React.js |
| 2 | Application Logic-1 | Authentication & User Management | Node.js + Express.js |
| 3 | Application Logic-2 | Appointment Booking Flow | Node.js + Express Middleware |
| 4 | Application Logic-3 | Doctor Slot Management | Node.js + Mongoose ODM |
| 5 | Database | Stores user, doctor, appointment data | MongoDB |
| 6 | Cloud Database | Hosted database for production | MongoDB Atlas |
| 7 | File Storage | Profile pictures & documents | Cloudinary / Local |
| 8 | External API-1 | Email confirmations | Nodemailer (SMTP) |
| 9 | External API-2 | Future Integration (e.g., SMS) | Twilio / MSG91 |
| 10 | Machine Learning Model | Optional future feature for smart doctor matching | Scikit-learn / TensorFlow |
| 11 | Infrastructure (Server / Cloud) | Deployment of frontend/backend | Vercel (Frontend), Render (Backend) |

## Table 2: Application Characteristics

|  |  |  |
| --- | --- | --- |
| S.No | Characteristics | Description / Technology |
| 1 | Open-Source Frameworks | React.js, Node.js, Express.js, MongoDB |
| 2 | Security Implementations | JWT Auth, HTTPS, Helmet.js, Input Validation, bcrypt.js |
| 3 | Scalable Architecture | Three-tier architecture with microservices-ready backend |
| 4 | Availability | Deployed using cloud services with global edge delivery (Vercel + Render) |
| 5 | Performance | Optimized REST APIs, caching with in-memory data, responsive UI, CDN on Vercel |

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