Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	19-07-2025
Team ID	LTVIP2025TMID50887
Project Name	DocSpot
Maximum Marks	4 Marks

Technical Architecture:

DocSpot is designed with a robust and scalable 4-tier architecture consisting of:

• **Presentation Layer (Frontend):** Responsive web and mobile applications providing intuitive interfaces for patients, healthcare providers, and administrators to manage appointments, profiles, and healthcare services.

• **Application Layer (Backend):** RESTful API services handling core business logic including appointment scheduling, user authentication, payment processing, notification management, and healthcare provider verification.

• **Data Storage Layer:** Secure and encrypted database systems storing user profiles, appointment records, medical documents, payment information, and healthcare provider credentials with HIPAA compliance.

Table-1: Components & Technologies:

Component	Description	Technology
User Interface (Web)	Responsive web application for patients and healthcare providers	React.js, TypeScript, Tailwind CSS, HTML5, CSS3
API Gateway	Centralized entry point for all client requests and API management	Express.js, Node.js, API Gateway (AWS/Azure)
Authentication Service	User authentication, authorization, and session management	JWT, OAuth 2.0, bcrypt
Appointment Logic	Core appointment booking, scheduling, and management system	Node.js, Express.js, Mongoose ODM
Notification Service	Real-time notifications, SMS, email, and push notifications	Socket.io, Twilio API, SendGrid, Firebase FCM
Admin Panel	Administrative dashboard for system management and analytics	React.js, Redux, Chart.js, Material-UI
Database Management	Primary database for storing all application data	MongoDB, MongoDB Atlas, Mongoose

Table-2: Application Characteristics:

Characteristics	Description	Technology
Frontend Frameworks	Modern JavaScript frameworks for responsive UI development	React.js, Next.js, Vue.js, Angular
Backend Frameworks	Server-side frameworks for API development and business logic	Node.js, Express.js, Fastify, NestJS
Database Technologies	NoSQL and SQL databases for different data requirements	MongoDB, PostgreSQL, MySQL, Redis
Testing Framework	Automated testing for quality assurance	Jest, Mocha, Cypress, React Testing Library
API Architecture	RESTful APIs with microservices architecture	REST API, GraphQL, Microservices, API Gateway
Development Tools	Development environment and productivity tools	VS Code, Postman, Git, npm/yarn, Webpack
Responsive Design	Cross-platform compatibility and responsive layouts	Bootstrap, Tailwind CSS, Material-UI, Ant Design
Real-time Features	Live updates, notifications, and real-time communication	WebSockets, Socket.io, Server-Sent Events
Performance Optimization	Caching, CDN, and performance monitoring	Redis, CDN (CloudFlare), Lazy Loading, Code Splitting
Scalability Solutions	Horizontal and vertical scaling capabilities	Load Balancers, Auto Scaling Groups, Microservices

Technology Justification:

Frontend Technologies:

- **React.js**: Component-based architecture for maintainable and scalable UI
- **TypeScript**: Type safety and better development experience
- **CSS**: Utility-first CSS framework for rapid UI development

Backend Technologies:

- **Node.js**: JavaScript runtime for fast, scalable server-side applications
- **Express.js**: Minimal and flexible web framework for building APIs
- **MongoDB**: Flexible NoSQL database for handling diverse healthcare data

Integration Technologies:

- **REST APIs**: Standard communication protocol for web services
- **WebSockets**: Real-time bidirectional communication for notifications
- **JWT**: Secure token-based authentication for stateless sessions

References:

1. **React.js Documentation**: https://reactjs.org/docs/getting-started.html
2. **Node.js Best Practices**: https://github.com/goldbergyoni/nodebestpractices
3. **Express.js Guide**: https://expressjs.com/en/guide/routing.html
4. **MongoDB Documentation**: https://docs.mongodb.com/
5. **JSON Web Token Reference**: https://jwt.io/introduction/
6. **RESTful API Design**: https://restfulapi.net/