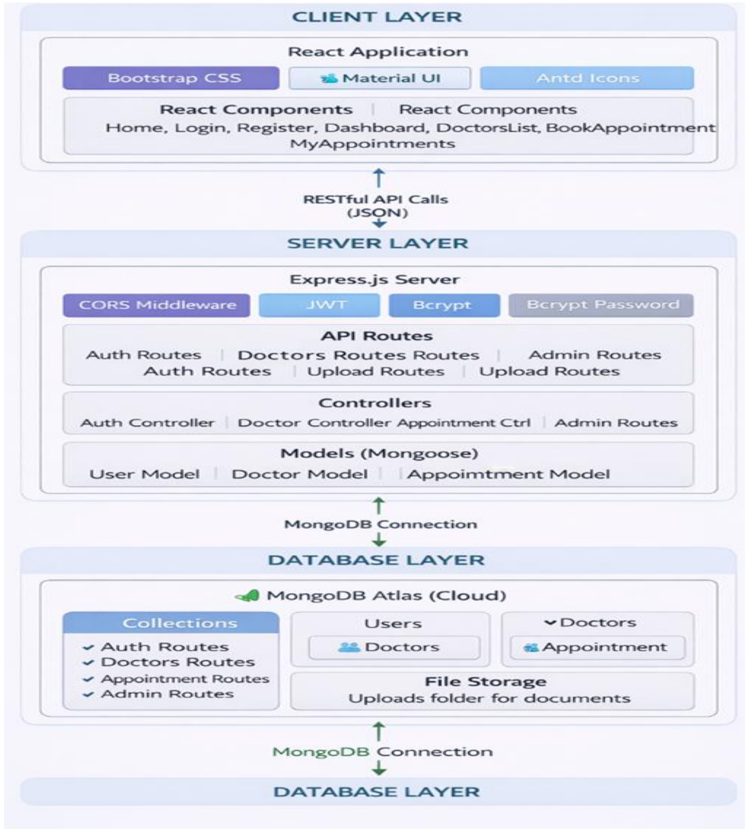
Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	16 February 2026
Team ID	LTVIP2026TMIDS35442
Project Name	DocSpot: Seamless Appointment Booking for Health
Maximum Marks	4 Marks

Technical Architecture:



Data Flow:

1. User interacts with React frontend
2. Axios sends HTTP requests to Express backend
3. Express routes handle requests via controllers
4. Controllers interact with MongoDB via Mongoose models
5. JWT middleware protects authenticated routes
6. Multer handles file uploads to uploads folder
7. Responses sent back to frontend

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Web application interface for patients, doctors, and admin	React.js 18.2.0
2.	UI Component Libraries	Styling and UI components for better user experience	Bootstrap 5.3, React-Bootstrap 2.10, MDB React UI Kit 8.0
3.	HTTP Client	Communication between frontend and backend	Axios 1.6
4.	Application Logic - Backend	Server-side logic, API endpoints, business logic	Node.js 22.18, Express.js 4.18
5.	Authentication	User authentication and authorization	JWT (jsonwebtoken 9.0), Bcryptjs 2.4
6.	Database	Primary data storage	MongoDB 7.5 (Mongoose ODM)
7.	Cloud Database	Hosted database service	MongoDB Atlas
8.	File Storage	Storage for uploaded medical documents	Local filesystem (uploads folder) with Multer 1.4
9.	Password Encryption	Secure password hashing	Bcryptjs 2.4
10.	Environment Configuration	Manage environment variables	Dotenv 16.3
11.	CORS	Cross-origin resource sharing	CORS 2.8
12.	Development Tools	Hot reloading for development	Nodemon 3.0

S.No	Component	Description	Technology
13.	Date/Time Handling	Parse and format dates	Moment.js
14.	Icons	Icon library for UI	React Icons
15.	Infrastructure (Server)	Application deployment	Localhost (Development)

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	All frameworks used are open-source	React.js, Node.js, Express.js, MongoDB, Bootstrap, JWT, Bcrypt
2.	Security Implementations	• Password hashing with bcrypt • JWT tokens for authentication • Protected routes with middleware • Role-based access control (Patient/Doctor/Admin) • Environment variables for secrets • Input validation and sanitization • CORS configuration • File upload type restrictions	Bcryptjs 2.4, JWT, Express Middleware, CORS, Multer
3.	Scalable Architecture	• Client-server architecture with separation of concerns • Stateless JWT authentication allows horizontal scaling • Modular code structure (controllers, models, routes) • MongoDB Atlas can scale with data growth • Indexed database collections for performance • React components reusable and modular	MERN Stack (MongoDB, Express, React, Node.js)
4.	Availability	• MongoDB Atlas provides 99.9% uptime • Application available 24/7 on localhost • Graceful error handling prevents crashes • Database connection retry on failure	MongoDB Atlas, Express error handling

S.No	Characteristics	Description	Technology
5.	Performance	• Database indexing for faster queries • JWT tokens reduce database lookups • Efficient MongoDB queries with Mongoose • React virtual DOM for fast rendering • API response time < 500ms	MongoDB indexes, JWT, React virtual DOM

References:

- https://reactjs.org/
- https://nodejs.org/
- https://expressjs.com/
- https://www.mongodb.com/atlas
- https://mongoosejs.com/
- https://jwt.io/
- https://github.com/axios/axios
- https://react-bootstrap.github.io/
- https://mui.com/
- https://ant.design/