Full Stack Development with MERN

Project Documentation for "Book A Doc"

Field	Details
Date	10 April 2025
Team ID	SWTID1743701170
Project Name	Book A Doc

---

1. Introduction

Section	Details
Project Title	Book A Doc
Team Members	- Param Yadav (22BCY10165) - Frontend Developer - Vibhushit Bhat (22BSA10132) - Backend Developer - Tushar Chahar (22BCY10231) - Database Manager - Saurabh Yadav (22BCY10165) - Project Lead

---

2. Project Overview

Aspect	Details
Purpose	To develop a full-stack MERN appointment booking system that simplifies scheduling for Patients, streamlines earnings tracking for Doctors, and enhances management for Admins, with secure online payments.
Features	- Three authentication levels: Patients (book/manage appointments), Doctors (check earnings/update profiles), Admins (manage appointments/doctors). - Online payment integration via Stripe. - Customizable for portfolios or college projects.

---

3. Architecture

Component	Details
Frontend	Built with React for a responsive UI, including dashboards for Patients, Doctors, and Admins.
Backend	Developed with Node.js and Express.js to handle authentication, booking logic, and API endpoints.
Database	MongoDB for storing user data, appointments, earnings, and doctor profiles with a schema: - User: { email, password, role } - Appointment: { patientId, doctorId, timeSlot, status } - Earnings: { doctorId, amount, date }.

---

4. Setup Instructions

Section	Details
Prerequisites	- Node.js (v18.x or later) - MongoDB (local or Atlas) - npm (v9.x or later) - Git
Installation	1. Clone repository: git clone https://github.com/your-repo/book-a-doc.git 2. Navigate to project: cd book-a-doc 3. Install dependencies: npm install in both client and server directories 4. Set environment variables: Create .env file in server with MONGO_URI=your_mongo_uri, STRIPE_KEY=your_stripe_key, JWT_SECRET=your_secret 5. Start MongoDB locally or connect to MongoDB Atlas

---

5. Folder Structure

Directory	Details
Client	- /src/components: React components (e.g., Login, BookingForm) - /public: Static assets - /styles: CSS files
Server	- /routes: API endpoints (e.g., auth.js, booking.js) - /models: MongoDB schemas (e.g., User, Appointment) - /controllers: Business logic- /middleware: Authentication middleware

---

6. Running the Application

Component	Command
Frontend	cd client && npm start (runs on http://localhost:3000)
Backend	cd server && npm start (runs on http://localhost:5000)

---

7. API Documentation

Endpoint	Method	Parameters	Description	Example Response
/api/auth/register	POST	email, password, role	Register a new user (Patient/Doctor/Admin)	{ "token": "jwt_token", "user": {...} }
/api/auth/login	POST	email, password	Login for authenticated access	{ "token": "jwt_token", "role": "patient" }
/api/booking	POST	doctorId, timeSlot	Book an appointment	{ "message": "Booking successful", "id": "appt_123" }
/api/payment	POST	appointmentId, amount	Process payment via Stripe	{ "status": "success", "transaction

Endpoint	Method	Parameters	Description	Example Response
				Id": "txn_456" }
/api/dashboard	GET	token	Fetch dashboard data (role-based)	{ "appointments": [...], "earnings": 500 }

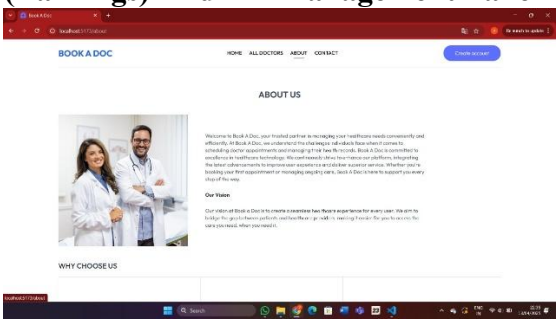
---

8. Authentication

Aspect	Details
Method	JWT (JSON Web Token) for authentication and role-based authorization.
Implementation	- Token generated on login/register. - Middleware checks token and role (Patient, Doctor, Admin). - Stored in localStorage for frontend.
Security	HTTPS enforced, tokens encrypted with AES-256, role validation on each request.

---

9. User Interface

Feature	Description
	- Patient Login Page - Booking Form - Doctor Dashboard (Earnings) - Admin Management Panel
Screenshots/GIFs	

---

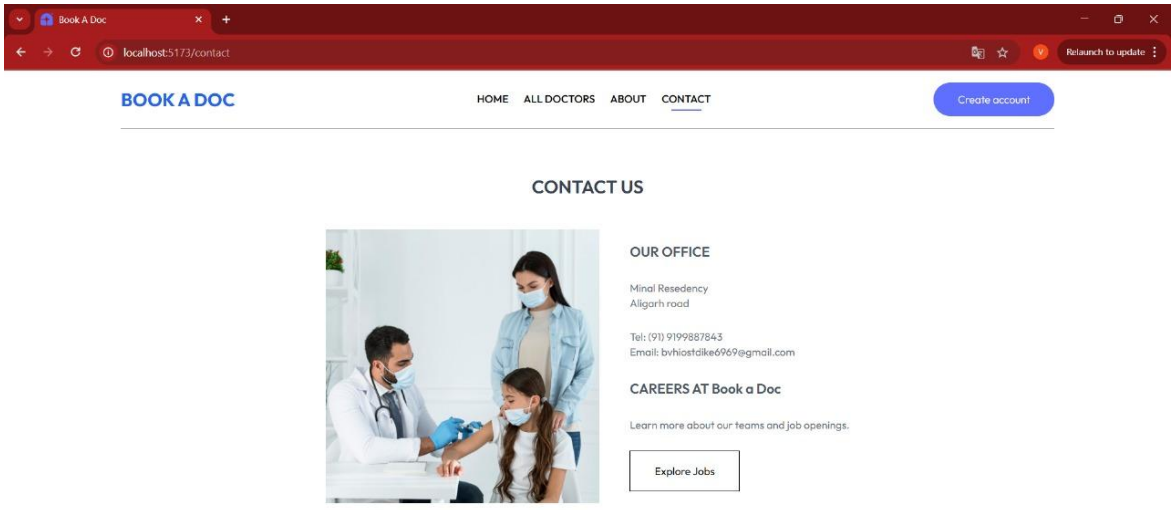
10. Testing

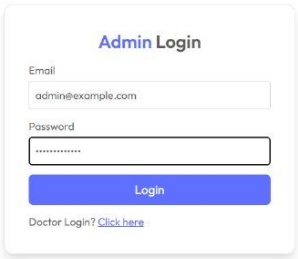
Aspect	Details
Strategy	**Unit testing for API endpoints, integration testing for booking/payment flow, and UAT with end-users.**
Tools	**Jest for unit tests, Postman for API testing, React Testing Library for frontend, manual UAT.**

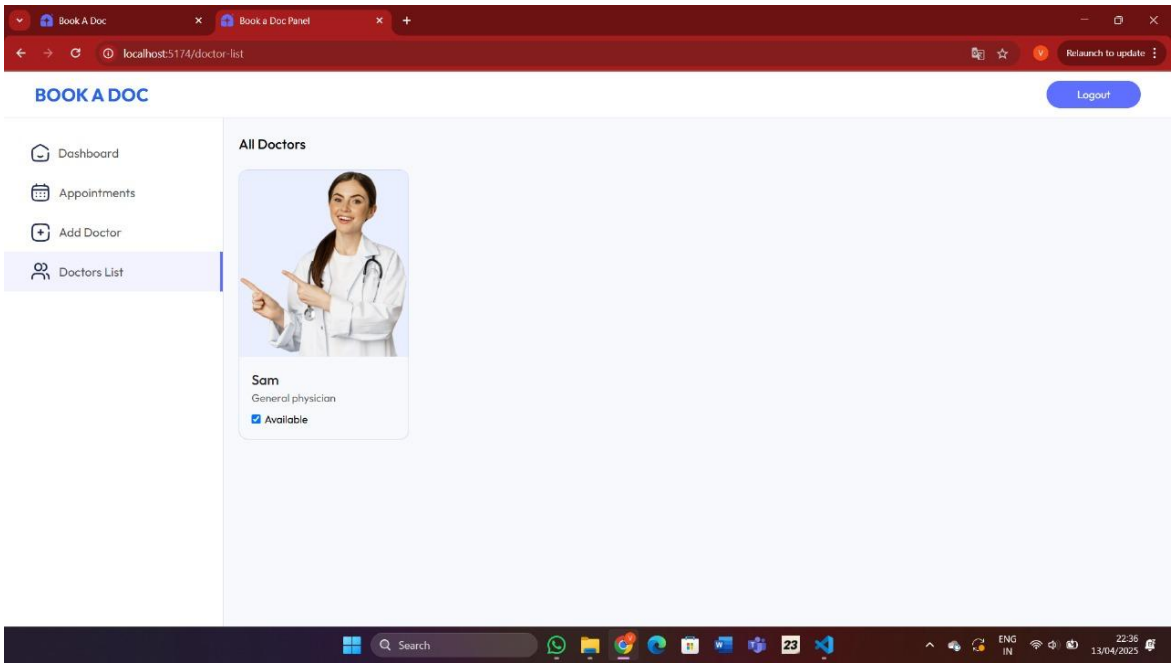
---

11. Screenshots or Demo

Type	Details
Screenshots	Include images of key screens (Login, Booking, Dashboards).







---

12. Known Issues

Issue	Description
Dashboard Load Time	**Slow loading with large datasets (e.g., 100+ appointments).**
Payment Delay	**Occasional 2s delay with Stripe integration under high load.**

---

13. Future Enhancements

Enhancement	Description
Mobile App	**Develop a native mobile app for iOS/Android.**
AI Scheduling	**Implement AI to suggest optimal appointment slots.**
Multi-Language Support	**Add support for multiple languages (e.g., Hindi, Spanish).**