1. Introduction

Project Title:

Docspot: Seamless Appointment Booking For Health

Team Members:

.Team ID : LTVIP2025TMID58569

.Team Leader : Nagireddy Venkateswarlu(Role no:22X51A0597)

.Team member : Pontala Venkata Vardhan(Role no:22X51A05B0)

.Team member : Challa Tharaka Rajesh(Role no:22X51A0519)

.Team member : Ummadi Hemalatha(Roll no:22X51A05E9)

2. Project Overview

.Purpose:

To provide users with a seamless platform to book healthcare appointments with doctors in real-time, improving accessibility and patient management.

.Features:

*User and Doctor Registration/Login

*Real-time appointment booking

*Doctor availability management

*Email confirmation/notifications

*Secure user authentication

*Admin panel for monitoring

3. Architecture

.Frontend (React):

*Built with React and Tailwind CSS

*Uses Axios to communicate with backend APIs

*React Router for navigation

*JWT-based auth integration for route protection

.Backend (Node.js + Express):

*RESTful API using Express.js

*Middleware for authentication and error handling

*MVC architecture

.Database (MongoDB):

*Collections: Users, Doctors, Appointments

*Mongoose schemas for validation and interaction

*Secure data access with role-based permissions

4. Setup Instructions

.Prerequisites:

* Node.js (v18+)

* MongoDB (local or cloud via MongoDB Atlas)

* Git

.Installation Steps:

bash

git clone https://github.com/your-username/docspot.git
cd docspot

Client Setup:

bash

cd client

npm install

Server Setup:

Bash

cd ../server npm install

Environment Variables:

Create .env file in the server folder:

MONGO_URI=your_mongodb_uri JWT_SECRET=your_secret_key

PORT=5000

5. Folder Structure*

Client (React Frontend**):**

client/

├— public/

├— src/

├— components/

├— pages/

├— services/

├— App.js

Server (Node.js Backend):

server/

├── controllers/

├── models/

├── routes/

├── middleware/

└── server.js

6. Running the Application:

Frontend:

cd client npm start **Backend:**
bash

cd server

npm start

7. API Documentation:

POST /api/auth/register Register a user name, email, password

POST /api/auth/login Login and get token email, password

GET /api/doctors Get list of doctors -

POST /api/appointments Book an appointment userId, doctorId, date

8. Authentication:

* JWT-based authentication
*Tokens stored in localStorage

* Protected routes using middleware

* Role-based access (User, Doctor, Admin)

9. User Interface:

* Homepage

* Login/Register

* Doctor Dashboard

* Appointment Form

* Confirmation Page

10. Testing:

Tools Used:

* Jest for unit tests

* Postman for API testing

* React Testing Library for UI components

Strategy:

* Component testing

* API endpoint validation

*Manual UI walkthroughs

11. Screenshots or Demo:

* **Demo Link:** [https://docspot-demo.vercel.app]

Demo Screens:

1. Landing Page

Clean homepage with tagline: “Book Health Appointments in Seconds”

2. Buttons: "Book Appointment", "Login", "Register"

3. User Login/Signup Page

4. Simple form with input fields

5. Toggle between patient and doctor login

6. Appointment Booking Page

7. Search doctor by specialty/location

8. Calendar to pick date/time

9. "Book Now" button

10. Doctor Dashboard

11. View upcoming appointments
12. Accept/Decline requests

13. Patient Dashboard
Cancel/reschedule option
Admin Panel (Optional)
Manage doctors/patients

* **Image Screenshots link:**

* ![Login Page]

https://i.imgur.com/MDWnTKS.png

* ![Doctor List]

https://i.imgur.com/YU0hXbQ.png

* ![Booking Confirmation]

https://i.imgur.com/GnGeHvb.png

12. Known Issues:

* Real-time slot update may occasionally lag without WebSocket

* No payment integration yet

* Limited filtering for doctors (e.g., by specialization)

13. Future Enhancements:

* Add payment gateway integration (Razorpay/Stripe)

* SMS reminders for appointments

* Video consultation via WebRTC
* Doctor rating & review system

* Admin analytics dashboard