📘 **DocSpot: Seamless Appointment Booking for Health**

**Full Stack MERN Project Documentation**

🔖 **1. Introduction**

**Project Title:**

DocSpot: Seamless Appointment Booking for Health

**Team Id: LTVIP2025TMID46651**

**Team Leader:**

Perupogu Rohith

**TeamMembers:**

Attim Siva Rani

P Joshna

**Overview:**

DocSpot is a full-stack healthcare web application that streamlines the process of booking medical appointments. It provides a user-friendly interface for patients to book appointments online, doctors to manage their schedules, and administrators to oversee system operations. The app includes secure role-based logins and integrated online payments for a complete appointment experience.

📌 **2. Project Overview**

**Purpose:**

To provide a centralized, convenient, and secure platform for healthcare appointment management accessible by patients, doctors, and administrators.

**Key Features:**

* User (Patient) registration, login, and appointment booking
* Doctor login and appointment management dashboard
* Admin panel to manage users and appointments
* Razorpay integration for online payments
* Secure authentication using JWT

🏗️ **3. System Architecture**

**Frontend (React.js)**

* Built with React and React Router
* UI components from Material UI, Bootstrap, Ant Design
* Axios for HTTP requests
* Responsive design for mobile and desktop

**Backend (Node.js + Express.js)**

* RESTful API architecture
* Role-based access control
* Razorpay payment API integration
* JWT authentication and route protection

**Database (MongoDB)**

* Collections:

o users (patients) o doctors o appointments o admins o payments

* Mongoose for schema definitions and data modeling

⚙️ **4. Setup Instructions Prerequisites:**

* Node.js (v18+)
* MongoDB (Local or Atlas)
* npm
* Git

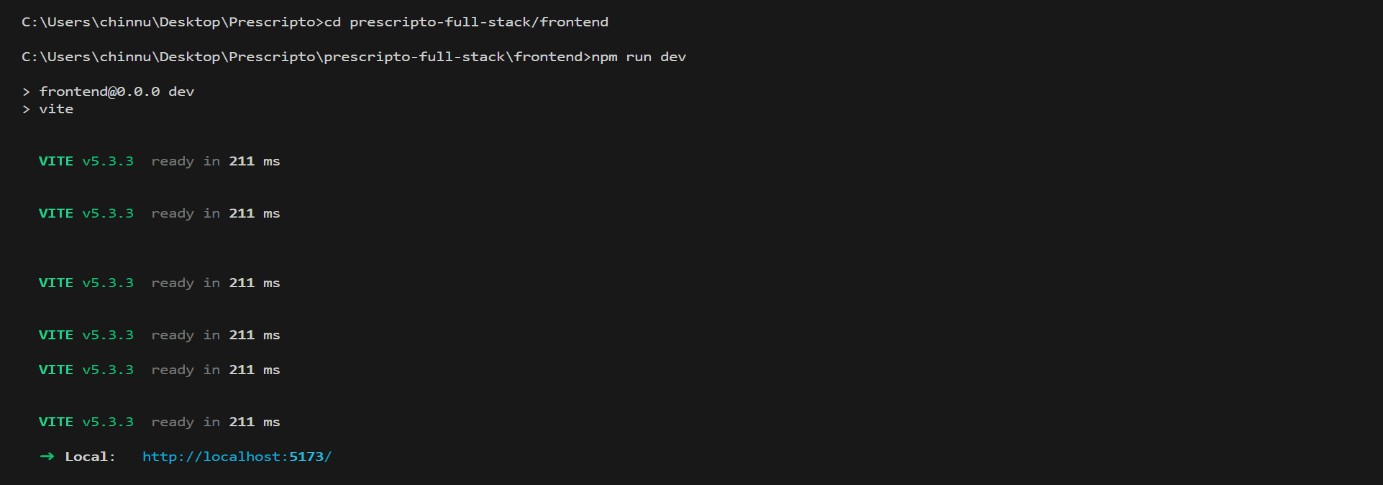
**Installation & Setup:**

bash

Copy code # Clone repository git clone https://github.com/your-username/docspot.git

# Frontend setup cd client npm install

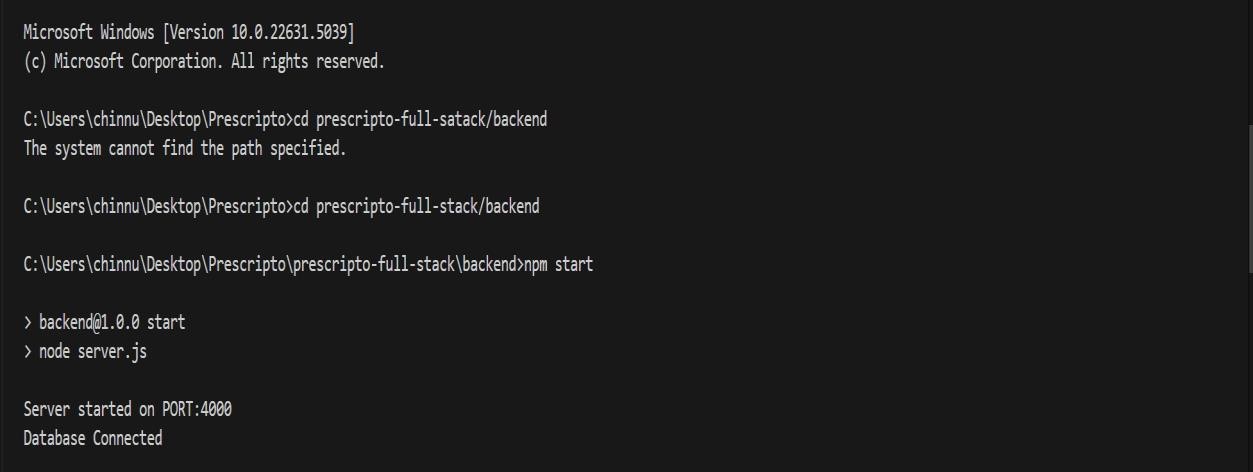
Frontend terminal:



# Backend setup

cd ../server npm install

Backend terminal:



# .env file in /server/

MONGO\_URI=your\_mongo\_uri

JWT\_SECRET=your\_jwt\_secret

RAZORPAY\_KEY=your\_razorpay\_key

RAZORPAY\_SECRET=your\_razorpay\_secret

PORT=5000

📁 **5. Folder Structure**

**Client:** pgsql Copy code

client/

├── public/

├── src/

│ ├── components/

│ ├── pages/

│ ├── context/ │ └── App.js, index.js

**Server:** pgsql Copy code server/

├── controllers/

├── routes/

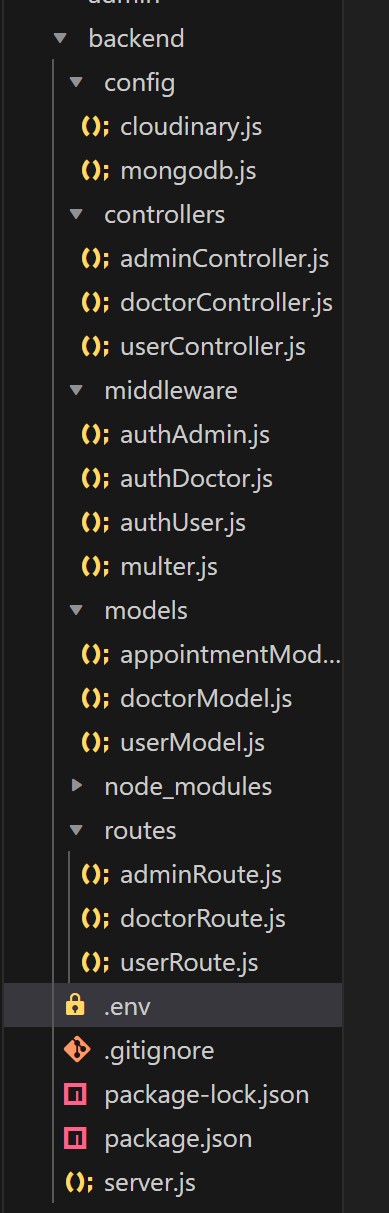
├── models/

├── config/

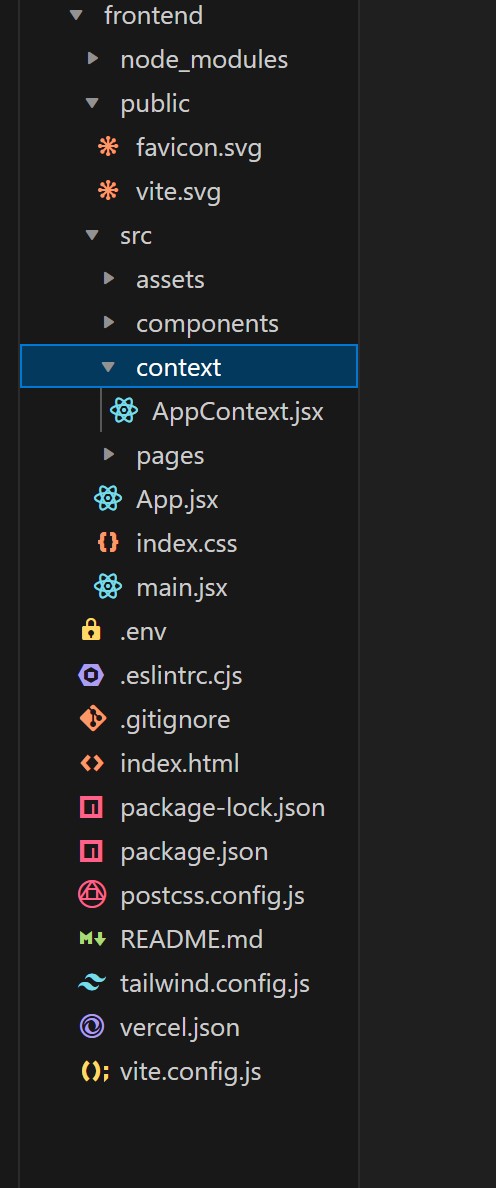
├── middlewares/

└── server.js

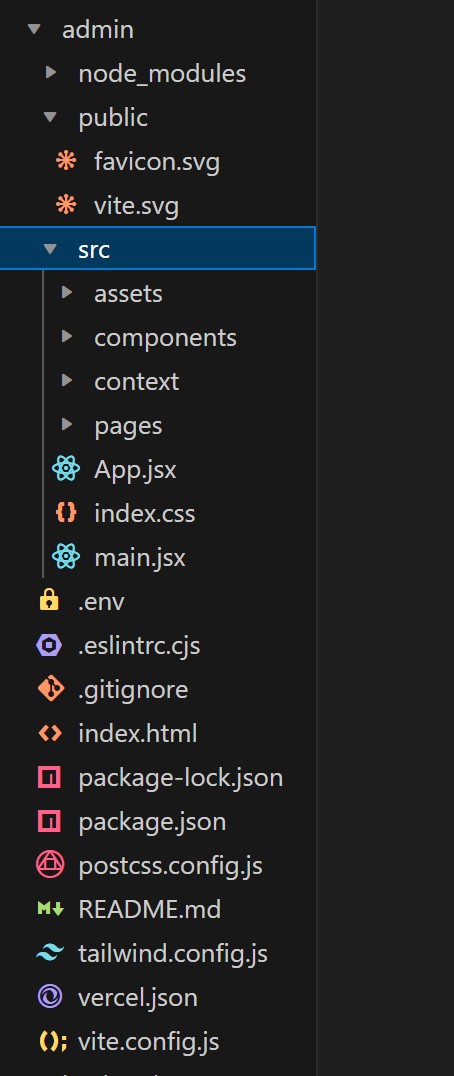
Backend Folder Structure:



Frontend Folder Structure:



Admin Folder Setup:



▶️ **6. Running the Application** bash

Copy code # Frontend cd client npm start

# Backend cd server

npm start

|  |  |
| --- | --- |
| 📡 **7. API Documentation**  **Method Endpoint Description** | |
| POST | /api/users/register Register a new user |
| POST | /api/users/login Login as user |
| POST | /api/doctors/login Doctor login |
| POST | /api/admin/login Admin login |
| POST | /api/appointments/book Book an appointment |
| GET | /api/doctors/:id/appointments Doctor views appointments |
| GET | /api/admin/appointments Admin views all appointments |
| POST | /api/payments/order Initiate Razorpay order |
| POST | /api/payments/verify Verify payment |

🔐 **8. Authentication & Authorization**

* **JWT Tokens** used for secure authentication
* **Protected Routes**: Based on roles (User, Doctor, Admin)
* **Token Flow**:
  + On login, token is generated and stored in local storage
  + Every protected request includes this token in the headers

🎨 **9. User Interface Screenshots**

Please upload the following screenshots with the specified names:

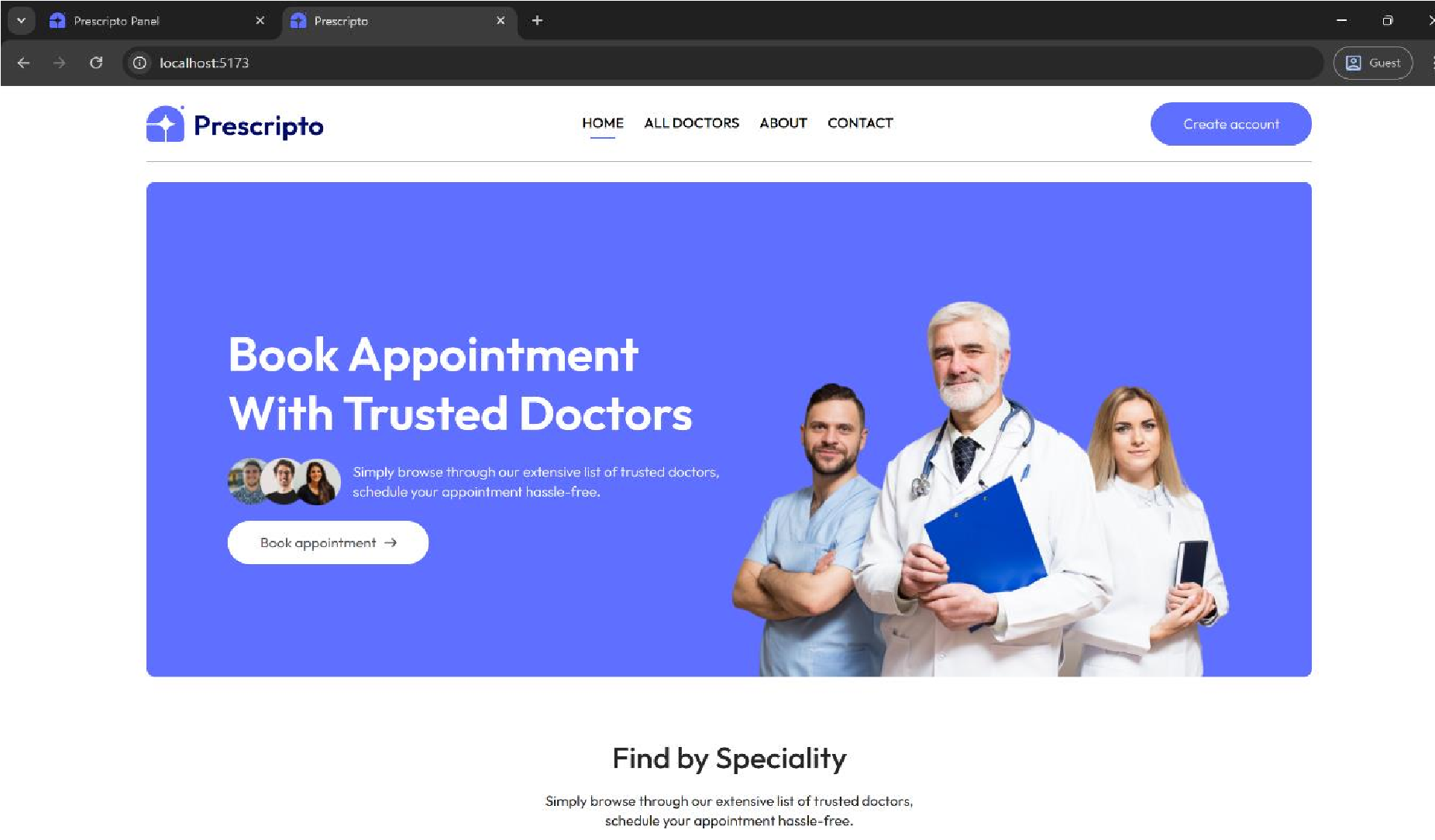
**Page Screenshot Name**

|  |  |
| --- | --- |
| User Login Page | user-login.png |
| Doctor Login Page | doctor-login.png |
| Admin Login Page | admin-login.png |
| User Dashboard (after login) | user-dashboard.png |
| Appointment Booking Page | book-appointment.png |
| Payment Checkout Page | payment-page.png |
| Doctor Dashboard | doctor-dashboard.png |
| Admin Dashboard | admin-dashboard.png |

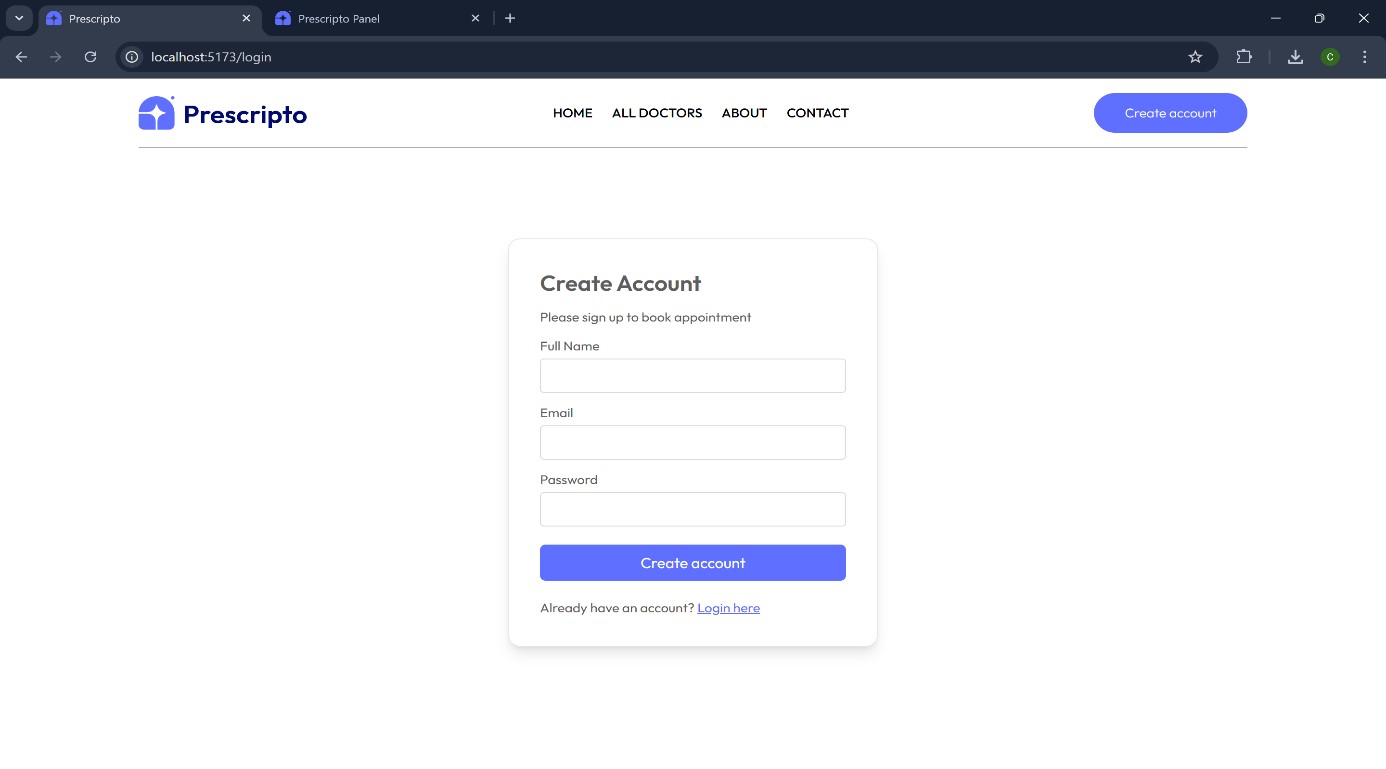
Success Message after Booking success-booking.png

Payment Success Page payment-success.png

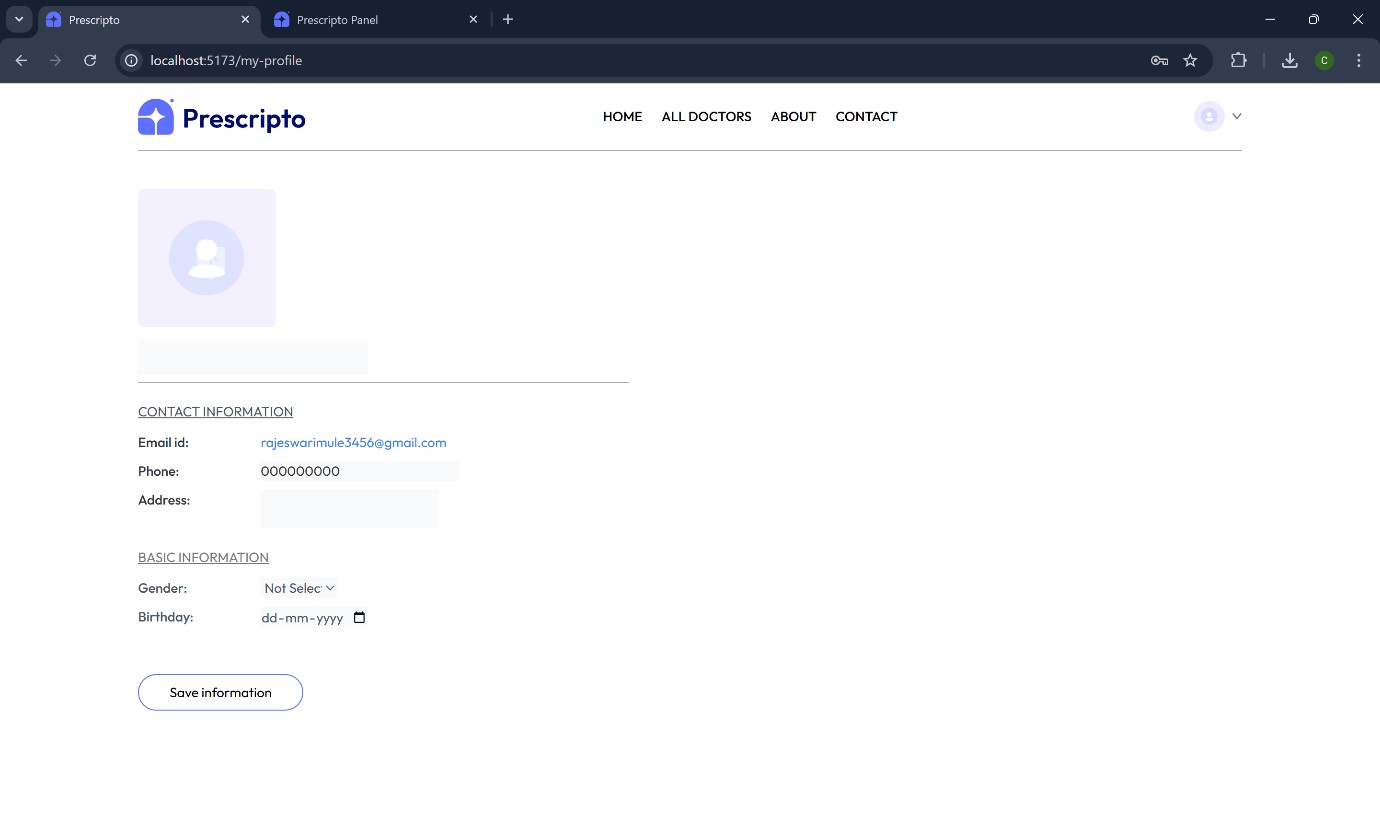
Home page(website):



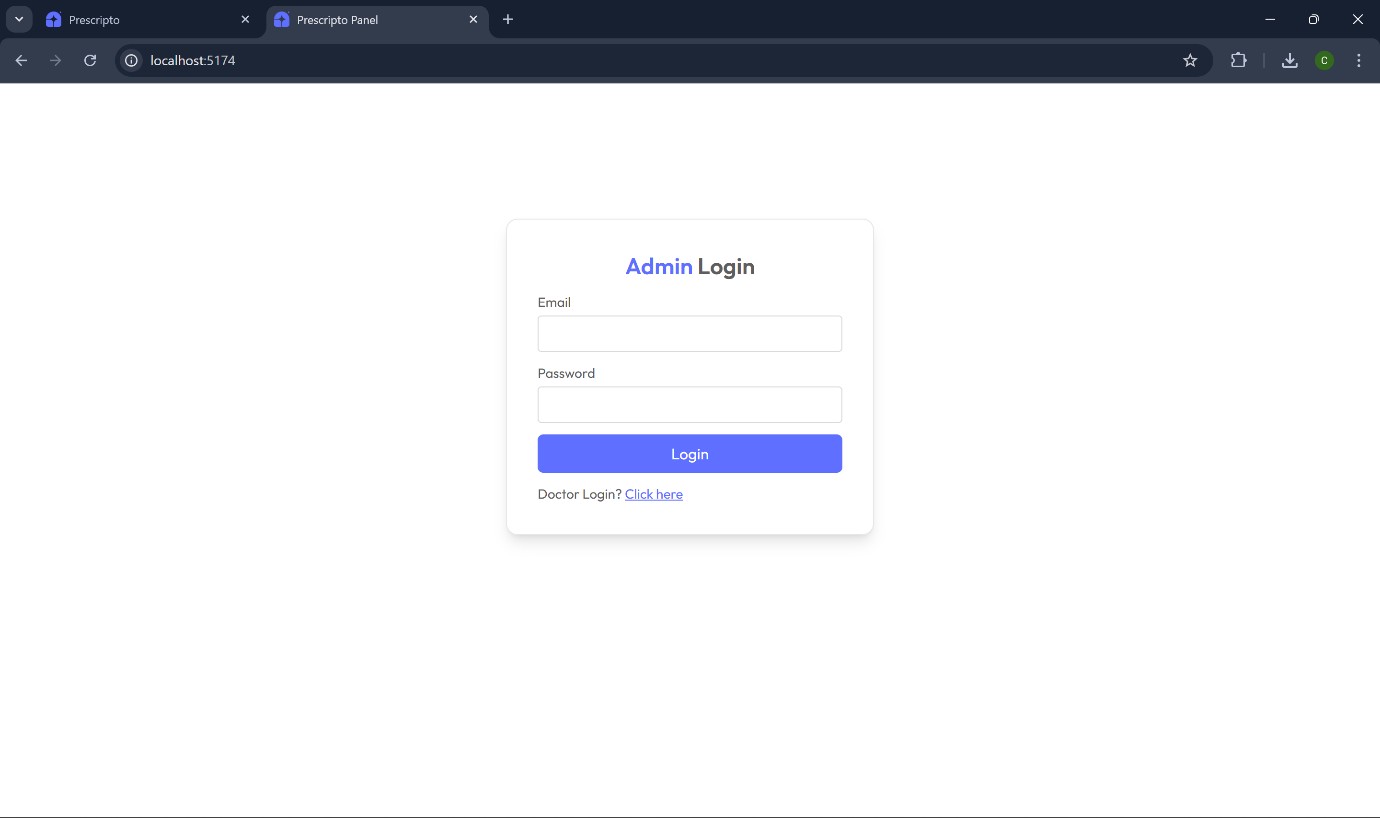
User Login:



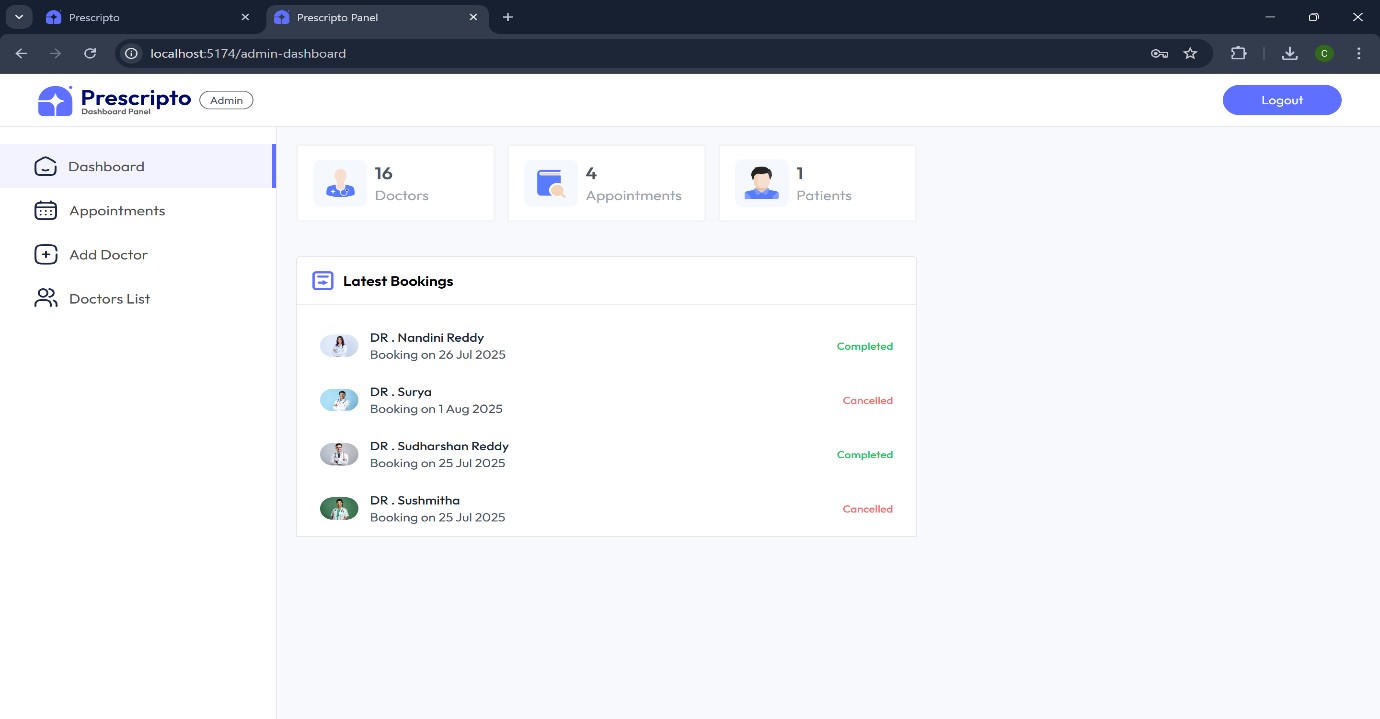
User Dashboard:



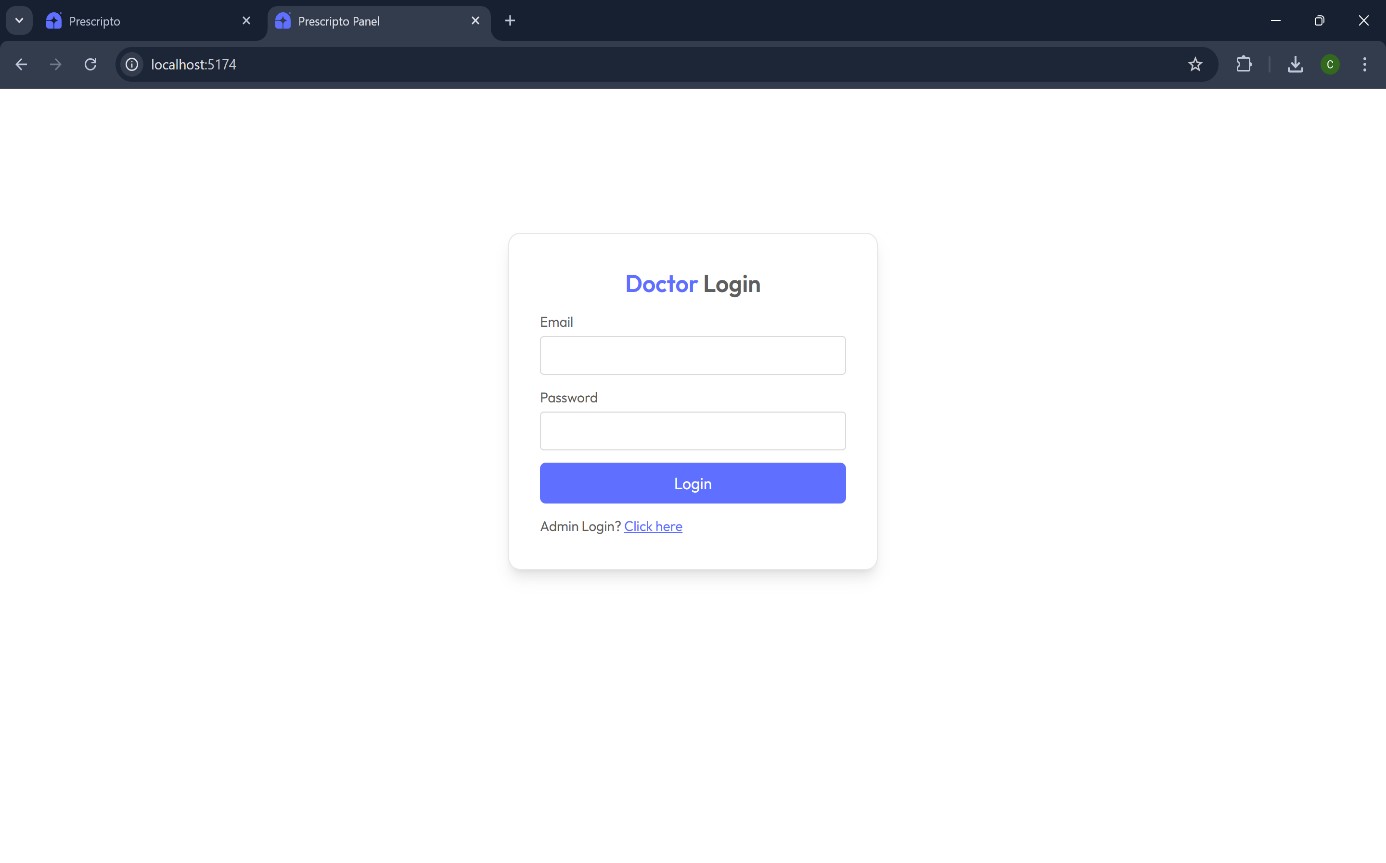
Admin Login Page:



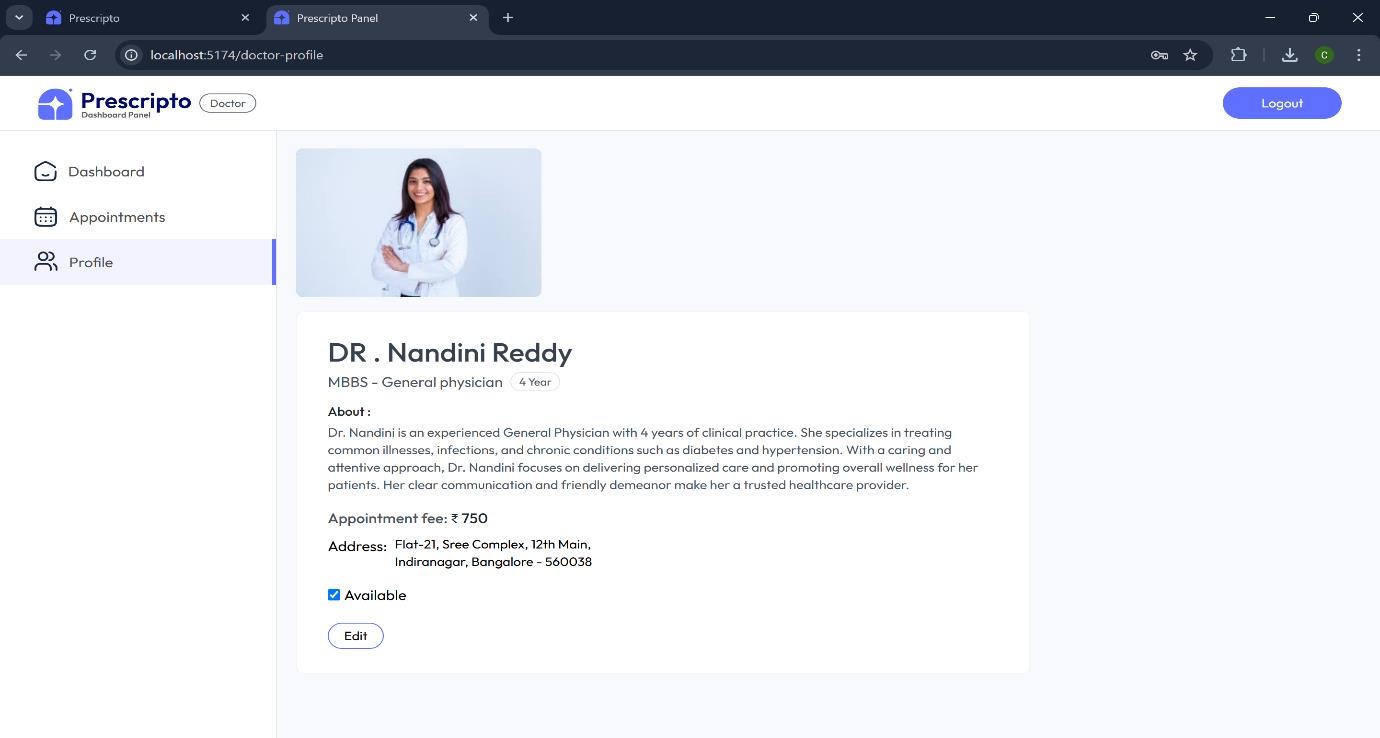
Admin Dashboard Page:



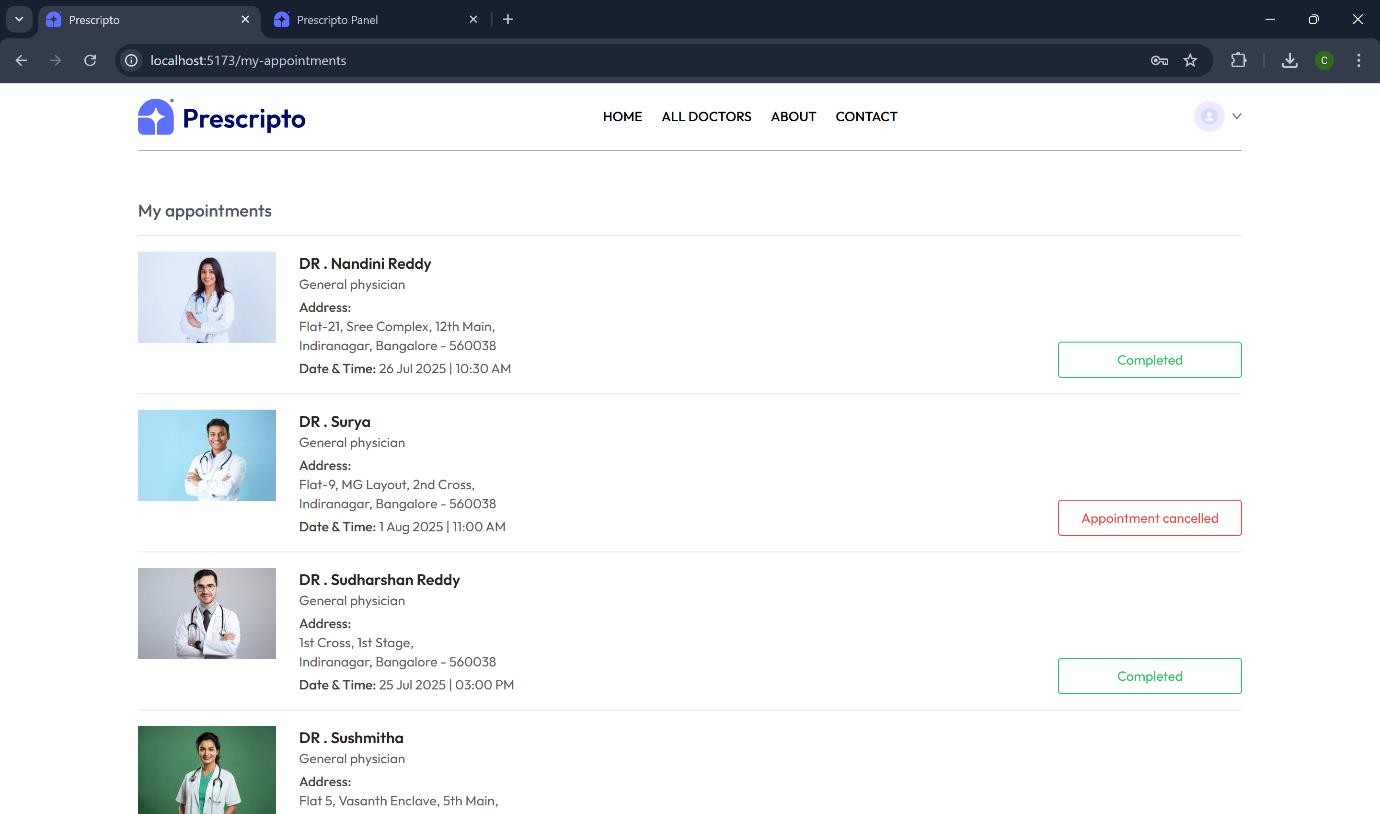
Doctor Login Page:



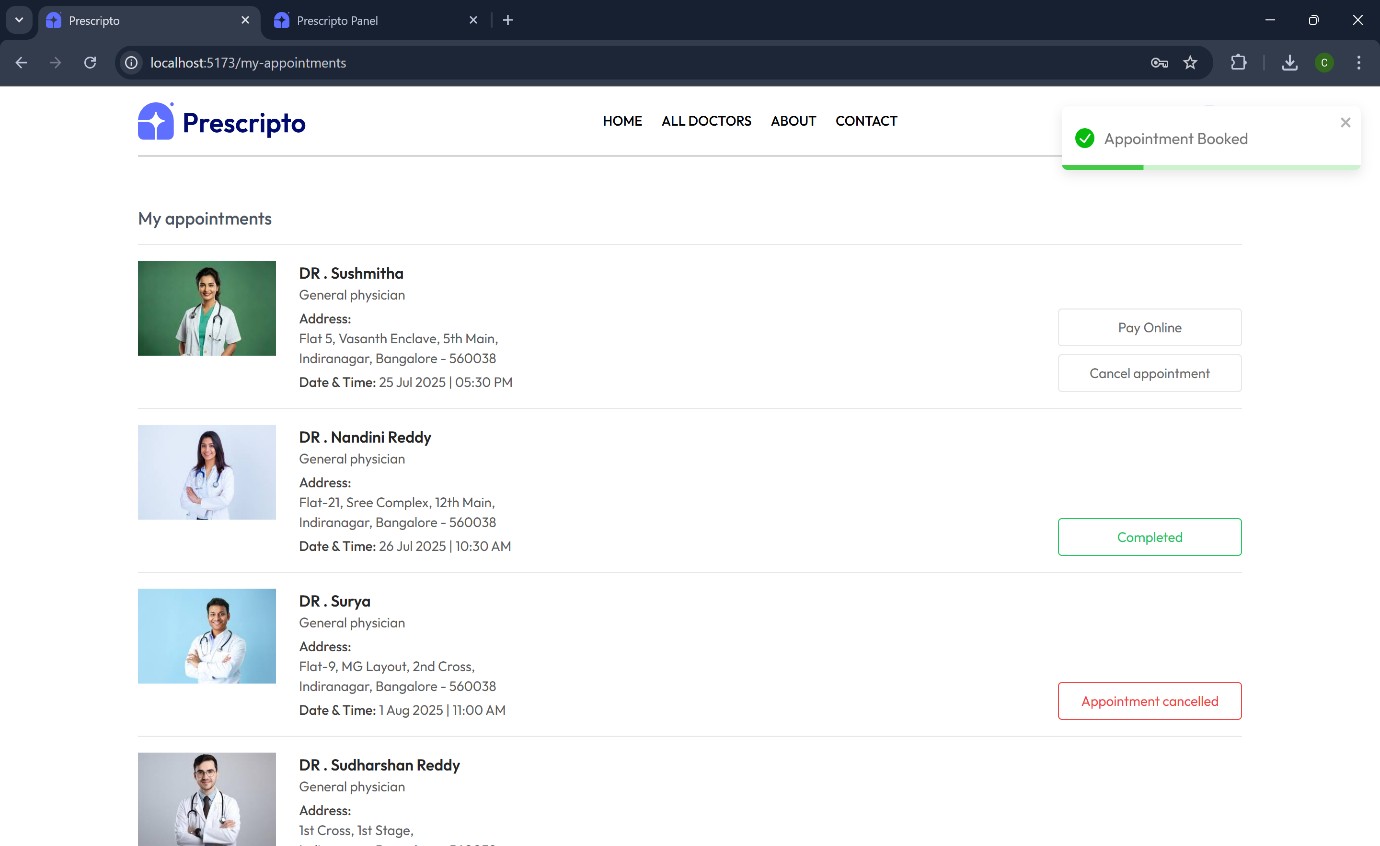
Doctor Dashboard:



Appointment Booking:



Appointment Booked Successfully:



🧪 **10. Testing**

**Tools Used:**

* Postman – Backend API testing
* Lighthouse – Frontend performance and accessibility
* Browser DevTools – UI and console validation

**Testing Coverage:**

* User login/logout and protected routes
* Doctor and admin access control
* Appointment conflict handling
* Payment success/failure scenarios

📷 **11. Demo or Showcase**

* **Screenshots**: As listed above
* **Optional Demo Video Link:** *[Insert YouTube/Google Drive demo link]*

⚠️ **12. Known Issues**

* No pagination for appointment list
* Doctor profile editing not implemented
* No email/SMS notifications yet

🚀 **13. Future Enhancements**

* Implement video consultations via WebRTC or Jitsi
* Add appointment filtering (by date/speciality)
* Notifications for appointment reminders
* Email confirmations for patients
* Real-time availability for doctors

📌 **14. Conclusion**

DocSpot successfully integrates frontend and backend technologies to deliver a secure and userfriendly platform for medical appointment management. With online payments and role-based control, it lays the groundwork for scalable future enhancements in digital healthcare.