

Project Report: Book a Doctor Using MERN

Members:

Farha Ansari
Aunkur Jha
Priyanshu Sharma
Ryan Santosh Joseph

1. INTRODUCTION

1.1 Project Overview

"**Book a Doctor**" is an innovative full-stack web application developed to streamline and modernize the process of scheduling medical appointments. Leveraging the robust capabilities of the **MERN stack**—MongoDB for the database, Express.js and Node.js for the backend, and React.js for the frontend—the platform delivers a responsive, efficient, and secure solution for healthcare scheduling.

This application serves as a bridge between patients and healthcare professionals, offering a centralized platform where users can effortlessly search for doctors based on specialization, location, or availability. Patients can view real-time schedules and book appointments directly without the need for phone calls or physical visits. The system features **role-based access control**, ensuring tailored functionalities and secure authentication for different user types—**patients**, **doctors**, and **administrators**. Each role is provided with a dedicated dashboard and specific permissions to interact with the system, such as managing appointments, updating availability, and monitoring system analytics.

The user interface is designed with a strong emphasis on **intuitiveness and responsiveness**, making it accessible across various devices, from desktops to mobile phones. Additionally, the integration of dynamic scheduling and calendar functionalities ensures that both patients and doctors can manage their time efficiently.

1.2 Purpose

The primary objective of this project is to **digitize and enhance the traditional healthcare appointment process**, eliminating common inefficiencies such as prolonged waiting times, scheduling conflicts, and communication gaps between patients and healthcare providers.

By adopting a digital-first approach, "*Book a Doctor*" seeks to improve healthcare accessibility, particularly in scenarios where time, mobility, or geographic location poses a challenge. The platform not only simplifies the user journey from appointment search to confirmation but also introduces transparency and automation into a system that has historically relied on manual coordination.

This project also serves as a demonstration of how **modern web development technologies** can be applied to solve real-world problems in the healthcare sector. By implementing scalable, maintainable, and secure architectural patterns, it highlights the practical utility of full-stack development in creating

solutions that can have meaningful social impact.

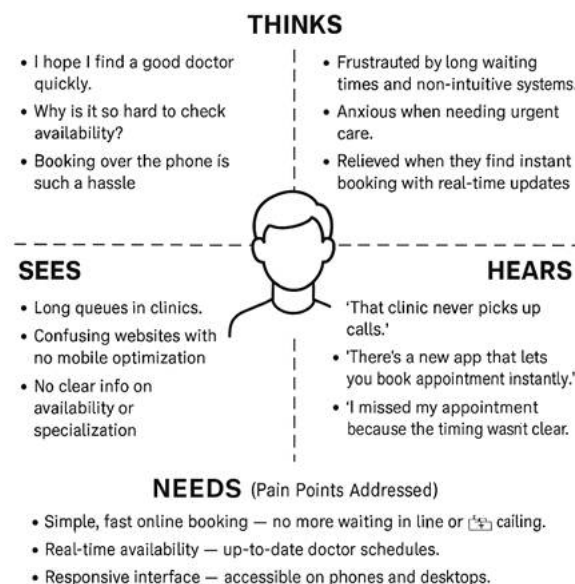
2. IDEATION PHASE

2.1 Problem Statement

The traditional process of booking medical appointments is often fraught with inefficiencies and barriers, both for patients and healthcare providers.

- **For Patients:** Many patients face significant hurdles when attempting to schedule appointments. These include restricted clinic operating hours, reliance on outdated and time-consuming phone-based booking systems, and the absence of real-time updates regarding doctor availability. Such limitations can lead to frustration, missed consultations, and delayed access to medical care—especially in urgent cases or for those in remote areas.
- **For Doctors:** Healthcare professionals often manage appointments manually or through fragmented systems, which increases the risk of double bookings, no-shows, and miscommunication. This inefficiency not only affects the doctor's workflow and revenue but also compromises the overall quality of patient care. The absence of centralized digital tools can result in additional administrative burdens, leaving less time for actual clinical work.

2.2 Empathy Map Canvas



Understanding the needs, pain points, and expectations of different stakeholders was critical during the ideation phase. An empathy-driven approach was employed to ensure the solution would address real user

problems effectively. The following summarizes key insights for each user role:

- **Patients**

- **Think & Feel:** Patients desire faster, more convenient access to healthcare services. They feel frustrated by delays and complex booking procedures.
- **Hear:** Word-of-mouth about long queues, unresponsive clinics, and missed appointments.
- **See:** Limited information on doctor availability, unclear scheduling processes.
- **Say & Do:** Often express dissatisfaction with the current system; resort to visiting clinics physically or making repeated phone calls.
- **Needs:**
 - Instant access to a list of doctors based on specialization/location
 - Real-time availability and instant confirmation
 - A mobile-friendly and user-friendly interface

- **Doctors**

- **Think & Feel:** Overwhelmed by time spent on non-clinical tasks like appointment management and administrative follow-ups.
- **Hear:** Complaints from patients about scheduling issues or communication delays.
- **See:** Inefficiencies in managing a growing patient list with outdated tools.
- **Say & Do:** Frequently look for ways to automate scheduling and record-keeping.
- **Needs:**
 - A centralized platform to view, update, and manage appointments
 - Patient record access during booking
 - Tools to reduce time spent on admin work

- **Admins**

- **Think & Feel:** Responsible for ensuring smooth operations, data integrity, and system functionality.
- **Hear:** Feedback from users needing support or reporting bugs/issues.
- **See:** The need for efficient user management and data monitoring.
- **Say & Do:** Monitor system usage, manage permissions, and troubleshoot issues.

- **Needs:**
 - Role-based user management
 - System-wide configuration control
 - Analytics and performance tracking

2.3 Brainstorming

During collaborative sessions, multiple ideas were generated and refined based on feasibility, impact, and user-centered design principles. The most promising features prioritized for implementation included:


- **Real-Time Availability Tracking:**
 - Dynamic schedules allow patients to view open slots and book immediately, reducing appointment conflicts and enabling better time management for doctors.
- **Role-Specific Dashboards:**
 - Custom dashboards tailored to each user type—patients, doctors, and admins—offering relevant tools and data views.
 - Patients can view their booking history and upcoming appointments.
 - Doctors can manage availability, appointments, and patient notes.
 - Admins can oversee user activity, control access, and configure system-wide settings.
- **Secure Authentication using JWT (JSON Web Tokens):**
 - Ensures that all user sessions are securely managed, reducing risks of unauthorized access or data breaches.
 - Token-based authentication supports scalability and secure interaction between the client and server.
- **Responsive UI with Modern Frameworks (e.g., Bootstrap):**
 - Ensures consistent and visually appealing user experiences across all screen sizes.
 - Enhances accessibility, especially for users on mobile devices, by using flexible layouts and intuitive navigation.

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

The customer journey outlines the sequence of steps a typical user follows when interacting with the

platform. Mapping this journey helps identify key touchpoints and ensure a seamless experience from discovery to continued engagement.

 CUSTOMER JOURNEY MAP — Patient Persona

Stage	Actions	Thoughts	Emotions	Pain Points	Opportunities/Solutions
1. Awareness	<ul style="list-style-type: none">- Searches online- Sees social media ad- Hears from friend	<ul style="list-style-type: none">"I need to see a doctor soon.""Is there any fast way to book?"	Curious, hopeful	<ul style="list-style-type: none">- Unfamiliarity with booking options- Trust issues	<ul style="list-style-type: none">- Social proof (testimonials)- SEO & ads targeting health queries
2. Consideration	<ul style="list-style-type: none">- Visits homepage- Browses doctors by specialty/location	<ul style="list-style-type: none">"Which doctor is nearby?""What are the ratings?"	Slightly confused, uncertain	<ul style="list-style-type: none">- Too many options- Difficult UI on other sites	<ul style="list-style-type: none">- Filters by location/rating- Clean UI & responsive design
3. Registration	<ul style="list-style-type: none">- Signs up and selects role as "Patient"- Enters basic details	<ul style="list-style-type: none">"Will my data be safe?""Hope this doesn't take too long."	Cautious, slightly impatient	<ul style="list-style-type: none">- Long or buggy registration- Poor UI feedback	<ul style="list-style-type: none">- Secure JWT auth- Smooth form validation- Mobile-first experience
4. Booking	<ul style="list-style-type: none">- Views doctor availability- Selects date & time- Confirms booking	<ul style="list-style-type: none">"Hope I picked the right time.""That was easy!"	Relieved, satisfied	<ul style="list-style-type: none">- Overlapping appointments- No confirmation message	<ul style="list-style-type: none">- Real-time calendar sync- Instant email/SMS confirmation
5. Pre-Visit	<ul style="list-style-type: none">- Gets appointment reminder- Views location map/directions	<ul style="list-style-type: none">"I'm ready for my visit.""Good thing I got reminded!"	Confident, prepared	<ul style="list-style-type: none">- No reminders- Confusion about clinic location	<ul style="list-style-type: none">- Automated reminders- Maps integration (Google Maps)
6. Post-Visit	<ul style="list-style-type: none">- Leaves a review- Views history- Books follow-up if needed	<ul style="list-style-type: none">"I'll give feedback.""Let me save this doctor for next time."	Grateful, engaged	<ul style="list-style-type: none">- No way to track past appointments- Forgetting to follow up	<ul style="list-style-type: none">- Doctor review & rating system- "My Appointments" dashboard
7. Retention	<ul style="list-style-type: none">- Logs in for follow-up- Recommends app to friends	<ul style="list-style-type: none">"This platform really helped me.""I'll tell others about this."	Loyal, happy	<ul style="list-style-type: none">- Lack of loyalty features	<ul style="list-style-type: none">- Referral system- Appointment history- Saved favorite doctors

1. Awareness

The user becomes aware of the *Book a Doctor* platform through online channels such as search engines, social media, advertisements, or word-of-mouth.

2. Registration

The user registers by creating an account using secure credentials. The platform verifies user identity and assigns roles such as patient, doctor, or administrator.

3. Search

The user explores available doctors using filters such as specialty, availability, location, or user ratings. The platform displays relevant results in real-time, enhancing decision-making.

4. Booking

The user selects a suitable time slot from a doctor's available schedule and confirms the appointment. An appointment summary is provided to both parties.

5. Post-Appointment

The user receives automated reminders prior to the appointment. Afterward, they can review the visit, cancel future appointments, or view their appointment history.

3.2 Solution Requirements

Functional Requirements

These define the essential features and operations of the application:

- **User Authentication**

Implementation of secure login and registration processes using token-based authentication. Role validation is performed during each session.

- **Appointment Booking and Cancellation**

Patients can schedule or cancel appointments. Doctors can manage their availability and update appointment statuses. Administrators can monitor all interactions.

- **Role-Based Dashboards**

Customized dashboards are provided for patients, doctors, and administrators, displaying relevant information and available actions based on user role.

Non-Functional Requirements

These address the system's performance, reliability, and usability:

- **Responsive User Interface**

The platform should be fully responsive across various devices and screen sizes, ensuring accessibility and ease of use.

- **Secure Data Handling**

All sensitive data, including login credentials and medical appointment details, should be encrypted and handled following industry-standard security practices.

- **Scalable Architecture**

The system should be designed to accommodate increasing numbers of users and features without compromising performance or stability.

3.3 Data Flow Diagram

Data Flow Description

The system follows a structured flow from user interaction to data persistence and feedback:

1. The user initiates actions via the React-based frontend (e.g., booking an appointment or logging in).
2. These actions generate HTTP requests sent to the Express.js backend via defined RESTful APIs.
3. The backend processes the requests, validates data, and interacts with the MongoDB database.
4. MongoDB returns the appropriate data to the backend.
5. The backend sends the response back to the frontend, where it is rendered for the user.

This data flow ensures a clear separation of concerns, facilitating maintainability and scalability.

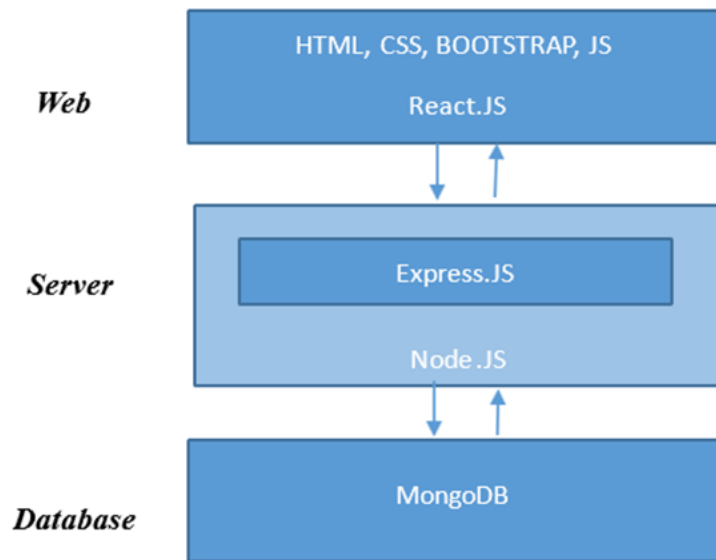


Fig. Data Flow Diagram

3.4 Technology Stack

The platform utilizes the following technologies, chosen for their efficiency, scalability, and compatibility within the full-stack development model:

Component	Technology	Purpose
Frontend	React.js, Bootstrap, HTML/CSS	To create an interactive and responsive user interface
Backend	Node.js, Express.js	To handle server-side logic and API management
Database	MongoDB	To store and manage user data, appointments, and schedules
Development Tools	Postman, GitHub, Render/Vercel	For API testing, version control, and deployment

- **Postman** is used to validate API endpoints during development.
- **GitHub** supports version control, collaboration, and issue tracking.
- **Render** or **Vercel** is employed for deploying the frontend and backend services to ensure continuous and reliable availability.

4. PROJECT DESIGN

4.1 Problem-Solution Fit

Traditional healthcare appointment systems rely heavily on manual processes such as phone calls or in-person scheduling, which are time-consuming, prone to error, and often inaccessible outside regular working hours. These systems result in long waiting times, scheduling conflicts, and inefficiencies in communication between patients and healthcare providers.

The proposed platform, *Book a Doctor*, addresses these issues by digitizing the scheduling process. It provides a centralized, real-time solution for appointment management, accessible 24/7. By automating critical operations such as booking, reminders, and role-based data access, the system reduces administrative burdens and minimizes human errors. This leads to improved patient satisfaction, better time management for doctors, and streamlined operational oversight for administrators.

4.2 Proposed Solution

The solution is designed around three core user roles, each with dedicated functionality tailored to their specific needs:

- **Patient Portal**
 - Allows patients to browse available doctors based on specialty, location, and availability.
 - Enables appointment booking, viewing of appointment history, and cancellation when necessary.
 - Provides timely reminders and access to relevant visit records or instructions.
- **Doctor Portal**
 - Offers doctors the ability to manage their availability by updating schedules dynamically.
 - Allows doctors to view and organize upcoming appointments, along with basic patient details.
 - Reduces time spent on administrative scheduling tasks, allowing a focus on clinical work.
- **Admin Panel**
 - Provides administrators with oversight capabilities, including user management, access control, and system configuration.
 - Enables resolution of disputes, monitoring of system activity, and enforcement of operational policies.
 - Facilitates auditing and platform analytics to ensure efficient performance.

4.3 Solution Architecture

The system is built on a modular and scalable full-stack architecture using the MERN stack, which

supports rapid development and flexibility. Each component is responsible for a specific layer in the application's workflow.

- **Frontend (React.js)**

- Implements a dynamic and responsive user interface using React.js.
- Facilitates seamless user interactions, routing, and state management.
- Integrates with REST APIs to fetch and display data in real-time.

- **Backend (Node.js with Express.js)**

- Acts as the server-side application that handles API requests, executes business logic, and manages data flow between the frontend and the database.
- Follows RESTful architecture principles for modular and scalable API design.

- **Database (MongoDB)**

- Stores all persistent data including user credentials, doctor profiles, patient information, and appointment records.
- Utilizes a document-oriented schema to allow flexibility in storing user-specific data.

- **Authentication (JSON Web Tokens - JWT)**

- Ensures secure and stateless authentication for all users.
- Upon successful login, users receive a token which must be included in all subsequent requests to access protected resources.
- Roles (patient, doctor, admin) are embedded in the token to support authorization throughout the application.

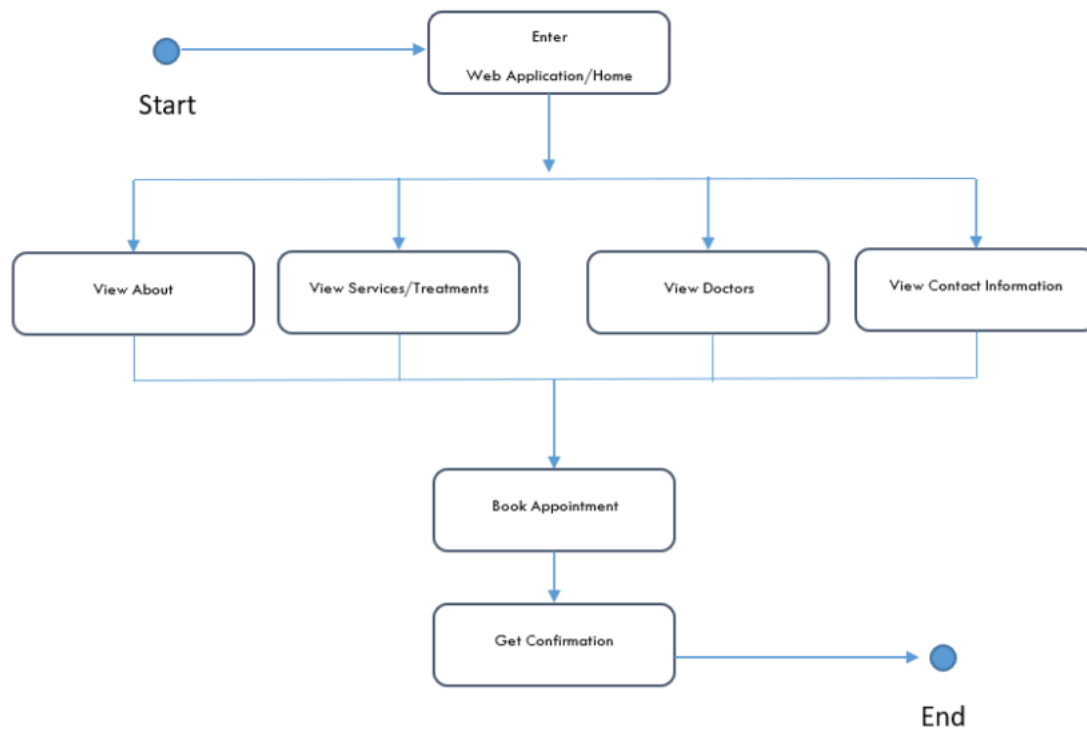


Fig. Activity Diagram (Patient)

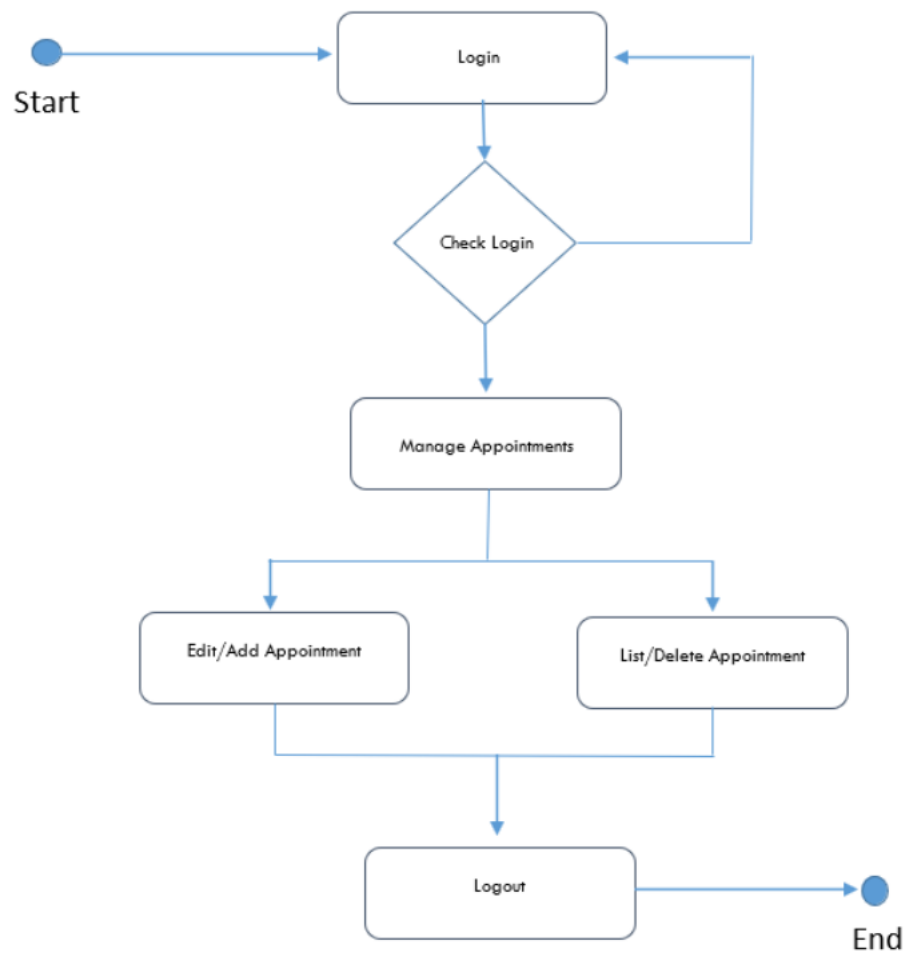


Fig. Admin Panel



Fig. Use Case Diagram

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

To ensure timely and structured development, the project was divided into four focused sprints, each targeting specific milestones. Agile methodology was adopted to facilitate iterative development and continuous improvement.

- **Sprint 1: UI/UX Design and Authentication Setup**
 - Designed responsive and accessible user interfaces for patient, doctor, and admin roles.
 - Implemented role-based navigation and layout components using React.js and Bootstrap.
 - Developed secure user authentication and authorization mechanisms using JWT and bcrypt.

- **Sprint 2: Backend API Development**

- Set up Express.js server with modular routing and middleware integration.
- Created RESTful APIs for core functionalities including user management, appointment booking, and role validation.
- Integrated MongoDB schemas for users, doctors, and appointments.

- **Sprint 3: Integration and Testing**

- Connected frontend components with backend APIs to enable end-to-end functionality.
- Conducted integration testing to validate seamless data flow between the client, server, and database.
- Addressed API response handling, form validation, and error feedback in the user interface.

- **Sprint 4: Deployment and Documentation**

- Deployed the application using Render/Vercel for both frontend and backend services.
- Verified production environment configurations and tested system reliability.
- Prepared detailed project documentation, including API references, usage guides, and system architecture.

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

Thorough testing was conducted to ensure the application met both functional and non-functional requirements. The testing covered system responsiveness, data security, and user interface consistency across platforms.

- **Load Testing**

- Simulated up to 500 concurrent users performing booking and browsing operations.
- The system maintained an average response time of under 2 seconds under load, confirming stability and scalability.

- **Security Testing**

- Verified authentication security by validating JWT tokens across all endpoints.
- Attempted unauthorized access scenarios to ensure that protected resources could not be accessed without valid credentials.

- Tested token expiry and refresh logic to prevent session hijacking.
- **UI Testing**
 - Conducted cross-browser compatibility tests on major browsers including Google Chrome, Mozilla Firefox, and Safari.
 - Verified consistent rendering, responsiveness, and component functionality across platforms and screen sizes.
 - Ensured accessibility standards were upheld, including proper tab navigation and readable font contrast.

```

22BH10111_Challenging_Task_1[1].py:nb X
C:\Users\wwwpr> AppData\Local\Microsoft\Windows\iNetCache> E> APW10LQD> 22BH10111_Challenging_Task_1[1].py:nb # Split data for traditional ML
+ Code + Markdown | Run All Clear All Outputs | Outline ...
[5]
# Evaluation function
def evaluate_model(y_true, y_pred, model_name):
    print(f"\n(model_name) Performance:")
    print("Accuracy:", accuracy_score(y_true, y_pred))
    print("F1-Score (macro):", f1_score(y_true, y_pred, average='macro'))
    print("\nClassification Report:")
    print(classification_report(y_true, y_pred))
    print("\nConfusion Matrix:")
    print(confusion_matrix(y_true, y_pred))
    print("\n" + "-"*30 + "\n")

# Evaluate models
evaluate_model(y_test, rf_pred, "Random Forest")
evaluate_model(y_test, svm_pred, "SVM")

...
Training Random Forest...
Training SVM...

VITE v5.3.5 ready in 235 ms
→ Local: http://localhost:5174/
→ Network: use --host to expose
→ press h + enter to show help

```

Code - Admin , Frontend and Backend Components

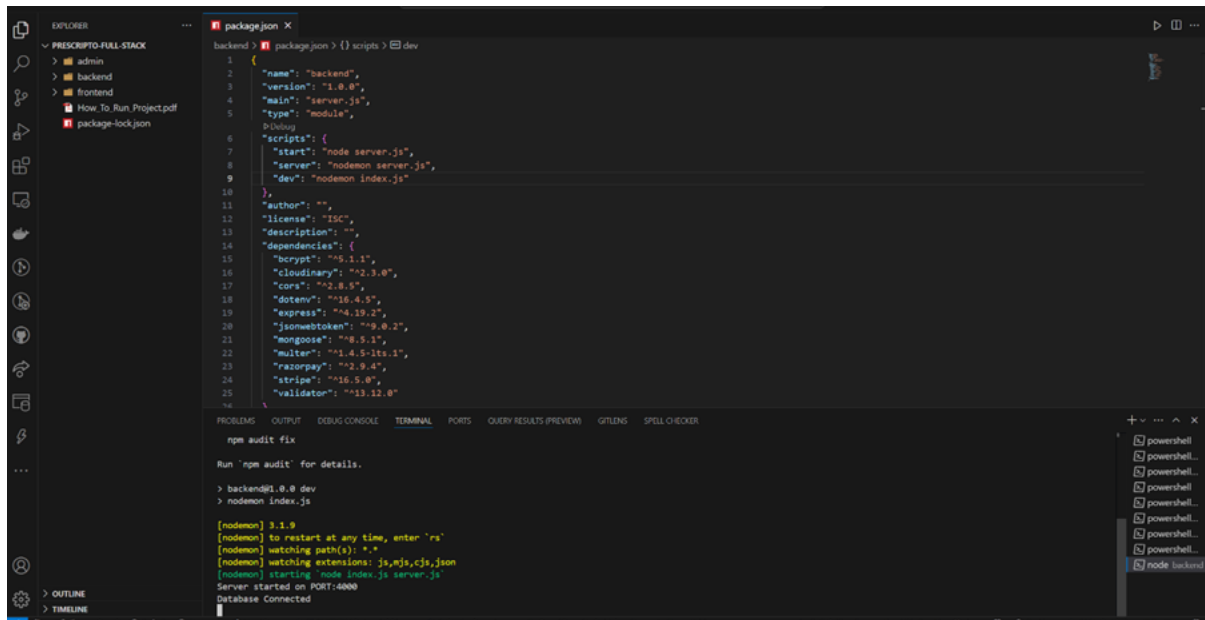
```

22BH10111_Challenging_Task_1[1].py:nb X
backend > .env
1  CURRENCY = "INR"
2  JWT_SECRET = "greatstack"
3
4  # Admin Panel Credentials
5  ADMIN_EMAIL = "admin@example.com"
6  ADMIN_PASSWORD = "greatstack123"
7
8  # MongoDB Setup ( required )
9  MONGODB_URI = "mongodb+srv://priyanshu18:priyanshu18@cluster0.ary7p.mongodb.net/"
10
11 # Cloudinary Setup ( required )
12 CLOUDINARY_NAME = "dmariblog"
13 CLOUDINARY_API_KEY = "813525257757454"
14 CLOUDINARY_SECRET_KEY = "Vnf_Z0AwV3PAe3KVf9KA5Nor2KA"
15
16 # Razorpay Payment Integration
17 RAZORPAY_KEY_ID = "----- Razorpay Key Id here -----"
18 RAZORPAY_KEY_SECRET = "----- Razorpay Key Secret here -----"
19
20 # Stripe Payment Integration
21 STRIPE_SECRET_KEY = "----- Stripe Secret Key here -----"

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS QUERY RESULTS (PREVIEW) GIT LENS SPELL CHECKER
VITE v5.3.5 ready in 235 ms
→ Local: http://localhost:5174/
→ Network: use --host to expose
→ press h + enter to show help

```

Credentials - Admin , panel Mangodb, clouinary, payment (Razorpay,stripe)



```
package.json
{
  "name": "backend",
  "version": "1.0.0",
  "main": "server.js",
  "type": "module",
  "scripts": {
    "start": "node server.js",
    "server": "nodemon server.js",
    "dev": "nodemon index.js"
  },
  "author": "",
  "license": "ISC",
  "description": "",
  "dependencies": {
    "bcrypt": "^5.1.1",
    "clouinary": "^2.3.0",
    "cors": "^2.8.5",
    "dotenv": "^16.4.3",
    "express": "^4.19.2",
    "jsonwebtoken": "^9.0.2",
    "mongoose": "^8.5.1",
    "multer": "^1.4.5-lts.1",
    "razorpay": "^2.9.4",
    "stripe": "^16.5.0",
    "validator": "^13.12.0"
  }
}
```

```
npm audit fix

Run 'npm audit' for details.

> backend@1.0.0 dev
> nodemon index.js

[nodemon] 3.1.9
[nodemon] to restart at any time, enter 'rs'
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting 'node index.js server.js'
Server started on PORT:4000
Database Connected
```

Database Connected

7. RESULTS

7.1 Output Screenshots

The following screenshots represent the core functionality and user interfaces of the *Book a Doctor* application, demonstrating how the system meets its design goals in terms of usability, responsiveness, and role-based access.

1. Homepage

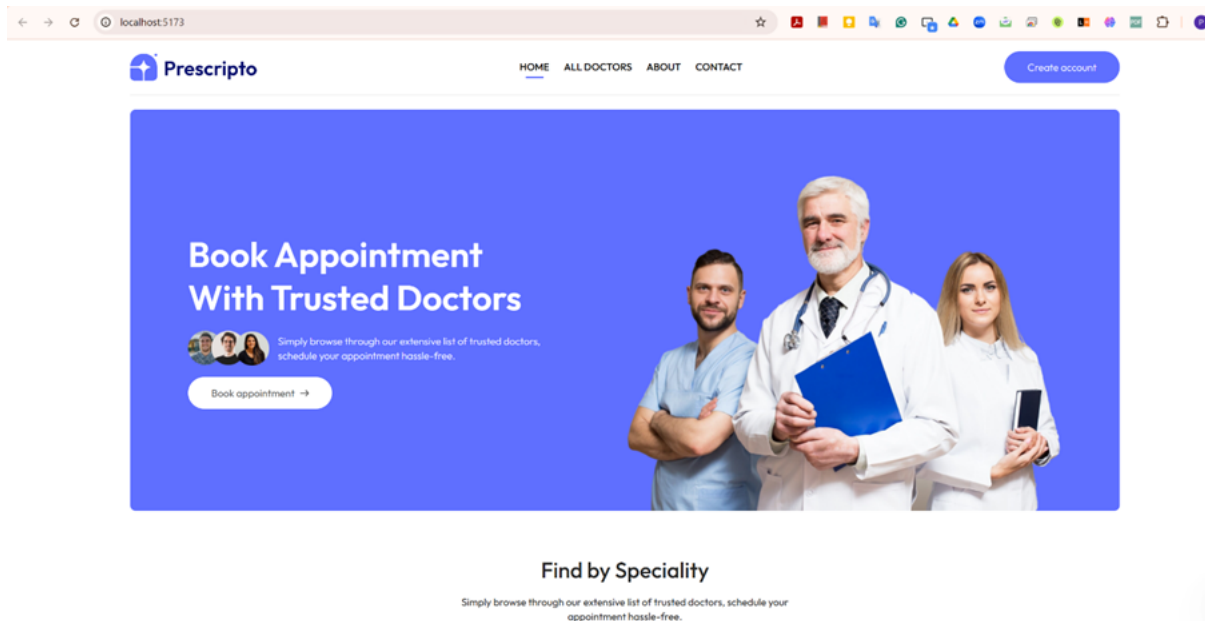
- Displays a user-friendly interface with a listing of available doctors.
- Includes search and filter options such as specialty, availability, and ratings to help users refine their choices.
- Provides quick access to login and registration options for new and returning users.

2. Appointment Booking Page

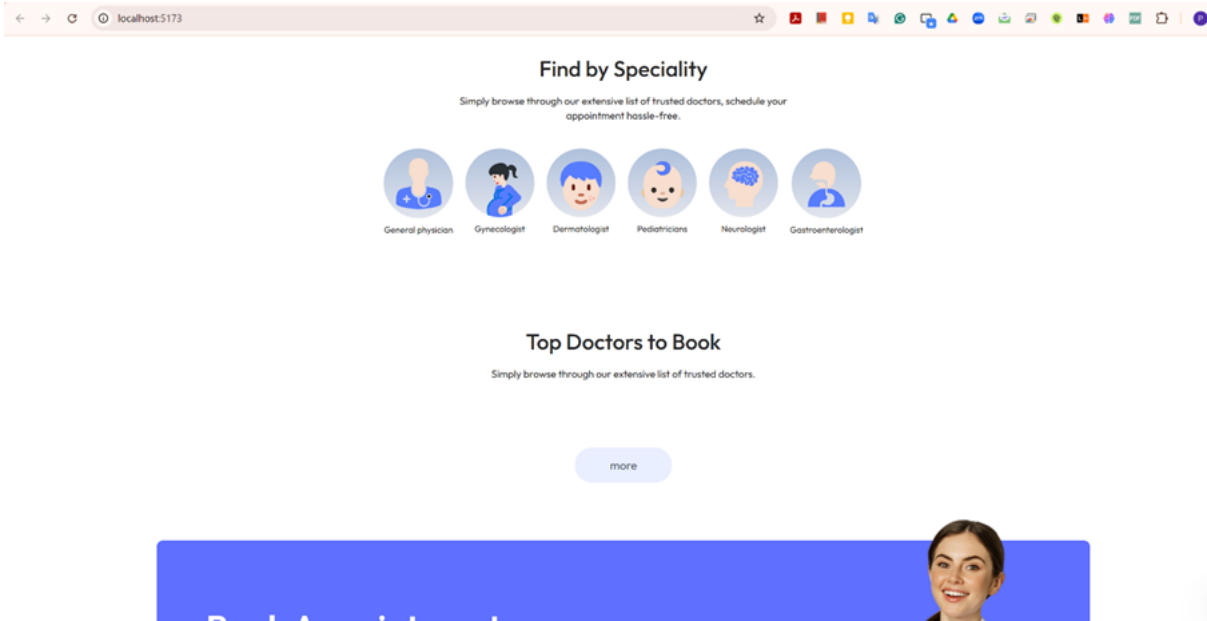
- Features an interactive calendar interface that allows patients to view real-time availability and select preferred time slots.
- Validates selected dates and times to prevent double bookings or conflicts.
- Displays doctor details alongside the booking form for clarity and confirmation.

3. Doctor Dashboard

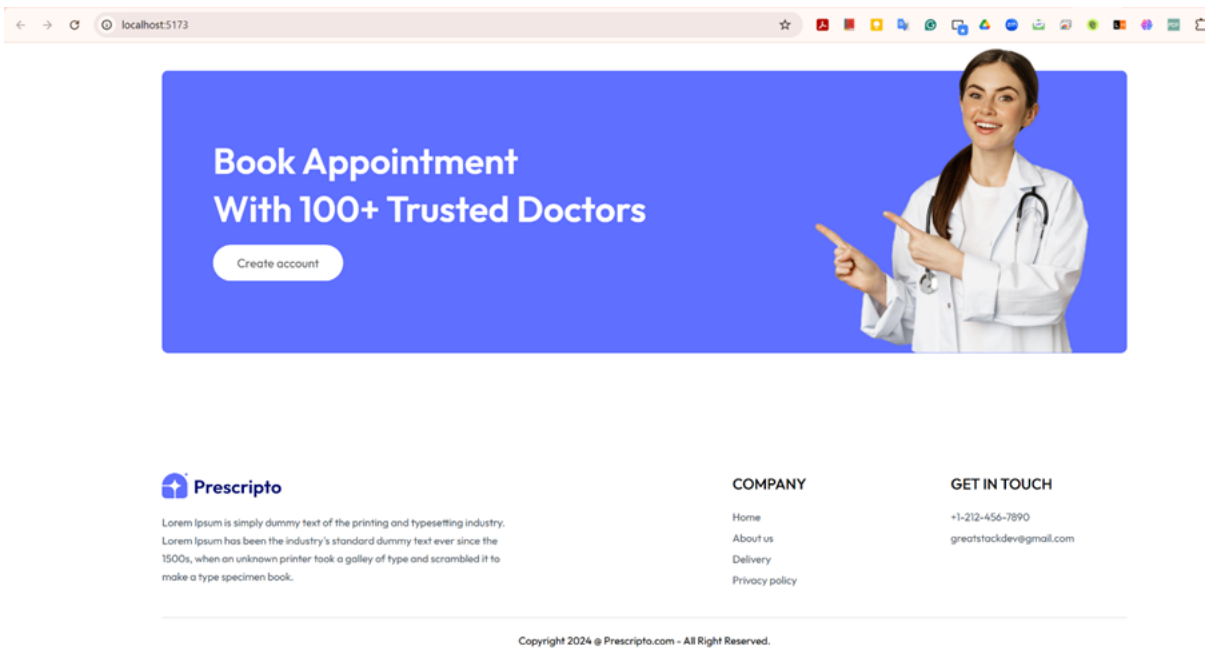
- Offers doctors a consolidated view of upcoming appointments, including patient names, appointment times, and statuses.
- Enables doctors to update appointment statuses (e.g., confirmed, completed, or cancelled).
- Provides basic patient information to support preparation ahead of each consultation.



Home Page



Home Page



Home Page

Create Account

Please sign up to book appointment

Full Name

Email

Password


[Create account](#)

Already have an account? [Login here](#)

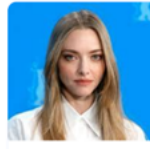
Create Account Page

Top Doctors to Book

Simply browse through our extensive list of trusted doctors.



• Available
Dr. Richard James
General physician

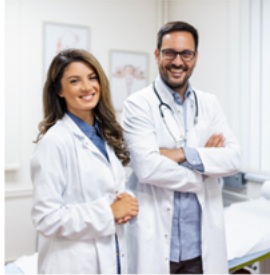


• Available
Amanda
Gynecologist

[more](#)

All Doctors Page

ABOUT US



Welcome to Prescripto, your trusted partner in managing your healthcare needs conveniently and efficiently. At Prescripto, we understand the challenges individuals face when it comes to scheduling doctor appointments and managing their health records.

Prescripto is committed to excellence in healthcare technology. We continuously strive to enhance our platform, integrating the latest advancements to improve user experience and deliver superior service. Whether you're booking your first appointment or managing ongoing care, Prescripto is here to support you every step of the way.

Our Vision

Our vision at Prescripto is to create a seamless healthcare experience for every user. We aim to bridge the gap between patients and healthcare providers, making it easier for you to access the care you need, when you need it.

WHY CHOOSE US

EFFICIENCY:

Streamlined appointment scheduling that fits into your busy lifestyle.

CONVENIENCE:

Access to a network of trusted healthcare professionals in your area.

PERSONALIZATION:

Tailored recommendations and reminders to help you stay on top of your health.

About Us Page

CONTACT US



OUR OFFICE

54709 Wilms Station
Suite 350, Washington, USA

Tel: (415) 555-0132
Email: greatstockdev@gmail.com

CAREERS AT PRESCRIPTO

Learn more about our teams and job openings.

[Explore Jobs](#)

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the

COMPANY

[Home](#)
[About us](#)

GET IN TOUCH

+1-212-456-7890
greatstockdev@gmail.com

Contact Us Page

Prescripto

HOMEALL DOCTORSABOUTCONTACT

92%

▼

Tony Stark

CONTACT INFORMATION

Email Id:tonystark@gmail.com

Phone:987654321

Address:

BASIC INFORMATION

Gender:Male

Birthday:2002-09-18

Edit

Patient Details Page

Prescripto

HOMEALL DOCTORSABOUTCONTACT

92%

▼

Amanda

MS - Gynecologist5 Year

About

Dr.Amanda MS in Gynecology

Appointment fee: ₹499

Booking slots

SAT12

SUN13

MON14

TUE15

WED16

THU17

FRI18

10:30 am

11:00 am

11:30 am

12:00 pm

12:30 pm

01:00 pm

01:30 pm

02:00 pm

02:30 pm

03:00 pm

Book an appointment

Related Doctors

Appointment Page

Prescripto

HOMEALL DOCTORSABOUTCONTACT

92%

▼

My appointments

Amanda

Gynecologist

Address:SerbiaNormway

Date & Time: 12 May 2025 | 12:30 PM

stripe

Razorpay

Cancel appointment

Appointment Fees Payment Interface

My appointments



Amanda
Gynecologist
Address:
Serbia
Norway
Date & Time: 12 May 2025 | 12:30 PM

Pay Online

Cancel appointment

Appointment Confirmation

My appointments



Dr. Richard James
General physician
Address:
47th Cross, Richmond
Seven Race Course, Delhi
Date & Time: 25 Apr 2025 | 11:00 AM

Completed

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

COMPANY

Home
About us
Delivery
Privacy policy

GET IN TOUCH

+1-212-456-7890
greatstackdev@gmail.com

My Appointments Page

- Dashboard
- Appointments
- Profile

₹ 40
Earnings2
Appointments2
Patients

Latest Bookings

Kapoor
Booking on 28 Apr 2025**Tony Stark**
Booking on 25 Apr 2025

Completed

Doctor Dashboard Login

Prescripto

Dashboard Panel



Doctor

Dashboard

Appointments

Profile

All Appointments

#	Patient	Payment	Age	Date & Time	Fees	Action
0	 Kapoor	CASH	23	28 Apr 2025, 02:00 PM	₹40	Completed
1	 Tony Stark	CASH	23	25 Apr 2025, 11:00 AM	₹40	Completed

Appointment Page (Doctor Interface)

Prescripto

Admin

Logout

Dashboard

Appointments

Add Doctor


Doctors List


2 Doctors


4 Appointments


3 Patients

Latest Bookings

 Amanda
Booking on 12 May 2025

 Amanda
Booking on 27 Apr 2025

 Dr. Richard James
Booking on 28 Apr 2025

 Dr. Richard James
Booking on 25 Apr 2025

Dashboard Interface (Admin)

Prescripto

Admin

Logout









Dashboard

Appointments

Add Doctor

Doctors List

All Appointments

#	Patient	Age	Date & Time	Doctor	Fees	Action
1	 Tony Stark	23	12 May 2025, 12:30 PM	 Amanda	₹499	
2	 Bruce	N/A	27 Apr 2025, 10:30 AM	 Amanda	₹499	Completed
3	 Kapoor	23	28 Apr 2025, 02:00 PM	 Dr. Richard James	₹40	Completed
4	 Tony Stark	23	25 Apr 2025, 11:00 AM	 Dr. Richard James	₹40	Completed

Appointment Page (Admin)

The screenshot shows the 'Add Doctor' form within the Prescripto Admin dashboard. The dashboard has a sidebar with links to Dashboard, Appointments, Add Doctor (highlighted), and Doctors List. The top right corner has a 'Logout' button. The 'Add Doctor' form includes the following fields:

- Upload doctor picture:** A circular icon with a plus sign and the text 'Upload doctor picture'.
- Your name:** A text input field labeled 'Name'.
- Speciality:** A dropdown menu with 'General physician' selected.
- Doctor Email:** A text input field labeled 'Email'.
- Degree:** A text input field.
- Set Password:** A text input field labeled 'Password'.
- Address:** Two text input fields labeled 'Address 1' and 'Address 2'.
- Experience:** A dropdown menu with '1 Year' selected.
- Fees:** A text input field labeled 'Doctor fees'.
- About Doctor:** A large text area labeled 'write about doctor'.

An 'Add doctor' button is located at the bottom of the form.

Add Doctor (Admin)

8. ADVANTAGES & DISADVANTAGES

Advantages

- **Reduces Administrative Workload**
The platform automates appointment scheduling, reminders, and user management, thereby minimizing the need for manual intervention by clinic staff.
- **Real-Time Updates Prevent Double Bookings**
Integrated availability tracking ensures that appointment slots are updated instantly, eliminating the risk of scheduling conflicts.
- **Accessible from Any Device**
Built with responsive design principles, the platform functions seamlessly on desktops, tablets, and mobile devices, ensuring accessibility for all users.

Disadvantages

- **Dependency on Internet Connectivity**

As a web-based platform, uninterrupted internet access is required for functionality. Users in low-connectivity areas may face accessibility issues.

- **Limited Offline Functionality**

The application currently does not support offline appointment management, which may restrict use in emergency scenarios or remote environments.

9. CONCLUSION

The *Book a Doctor* project successfully demonstrates the potential of full-stack web development—particularly the MERN stack—in solving real-world healthcare challenges. By digitizing the appointment booking process, the platform addresses inefficiencies found in traditional systems, enhances user experience through role-based interfaces, and promotes accessibility with responsive design.

The system showcases how modern technologies can be leveraged to reduce operational burdens, improve communication between patients and providers, and ensure more efficient use of healthcare resources. Overall, the project is a functional, scalable, and extensible solution aligned with contemporary healthcare needs.

10. FUTURE SCOPE

To further enhance functionality and user engagement, the following improvements are proposed for future iterations:

- **Integration of Payment Gateways**

Enable secure online payments for appointment fees using platforms like Razorpay or Stripe.

- **Video Consultation Support**

Incorporate WebRTC-based video calling features to facilitate remote consultations, particularly useful in telehealth scenarios.

- **AI-Driven Doctor Recommendations**

Leverage machine learning to suggest doctors based on user behavior, preferences, and appointment history.

- **Mobile Application Development**

Develop native Android and iOS applications to increase platform reach and improve accessibility on mobile devices.

11. APPENDIX

- **Source Code:** [GitHub Repository - <https://github.com/PriyanshuSharma18/book-a-doctor-frontend>]

Contains the complete codebase including frontend, backend, and configuration files.

- **Dataset:** [MongoDB Collections - *mongodb+srv://priyanshu18:priyanshu18@cluster0.ary7p.mongodb.net*]
- **Demo:** [Live Application Link - *https://book-a-doctor-frontend.vercel.app/*]
Deployed instance of the application showcasing all features and functionalities.