

Doctor Appointment Booking Website

A Project Report

submitted in partial fulfillment of the requirements

of

Electronics and Telecommunication Engineering

by

Suryansh Ambekar, suryansh.ambekar@mitaoe.ac.in

Kaustubh Mahajan, kaustubh.mahajan@mitaoe.ac.in

Tuhinansh Sharma, tuhinansh.sharma@mitaoe.ac.in

Mayank Soni, mayank.soni@mitaoe.ac.in

Under the Guidance of

Kaushal Joshi

ACKNOWLEDGEMENT

We would like to take this opportunity to express our heartfelt gratitude to everyone who has supported us, directly or indirectly, throughout this thesis work.

First and foremost, we extend our sincere thanks to our supervisor, Kaushal Joshi, for being an exceptional mentor and guide. His unwavering support, valuable advice, and constructive feedback have been instrumental in shaping the direction and success of this project. His encouragement and belief in my abilities provided a constant source of motivation and inspiration.

Working under his guidance over the past year has been an enriching experience. He has been a reliable source of assistance, not only in the technical aspects of this project but also in other areas of the program. His insightful discussions and guidance have greatly contributed to my personal and professional growth.

Beyond project-related matters, his teachings and encouragement have helped me become a more responsible and confident professional. I am deeply grateful for the opportunity to work with him and for all the knowledge and skills I have gained under his mentorship.

Suryansh Ambekar

Kaustubh Mahajan

Tuhinansh Sharma

Mayank Soni

ABSTRACT of the Project

The Doctor Appointment Booking System is a web-based platform designed to address inefficiencies in traditional appointment management. Manual systems often lack centralization, accuracy, and convenience for patients, doctors, and administrators. This project aims to create a scalable, user-friendly solution for scheduling appointments, managing doctor profiles, and streamlining administrative tasks.

The platform is built using the MERN stack (MongoDB, Express.js, React, and Node.js). The frontend enables patients to browse doctors by specialty, book appointments, and access a responsive interface, while doctors can manage their profiles and schedules. The backend uses Express.js and MongoDB to handle authentication, appointment management, and data storage, with bcrypt and JWT ensuring secure user authentication. An admin panel facilitates the addition or removal of doctors and appointment cancellations.

The methodology involved designing a modular architecture, creating a secure database schema, and implementing robust APIs for seamless communication between the frontend and backend. The project was tested to ensure usability, functionality, and responsiveness across devices.

Key results include a fully functional platform with features like filtered doctor searches, secure login systems, and efficient appointment management. The system reduces dependency on manual processes, enhancing convenience for all stakeholders.

In conclusion, this project demonstrates how modern web technologies can solve real-world challenges in healthcare management. By offering a centralized and efficient solution, the platform improves accessibility and paves the way for future enhancements, such as online payment integration and teleconsultation services.

TABLE OF CONTENTS

Abstract

List of Figures

Chapter 1. Introduction

- 1.1 Problem Statement
- 1.2 Motivation
- 1.3 Objectives
- 1.4. Scope of the Project

Chapter 2. Literature Survey

- 2.1 Review relevant literature or previous work in this domain.
- 2.2 Mention any existing models, techniques, or methodologies related to the problem
- 2.3 Highlight the gaps or limitations in existing solutions and how your project will address them.

Chapter 3. Proposed Methodology

- 3.1 System Design
- 3.2 Modules used
- 3.3 Advantages
- 3.4. Software Requirements

Chapter 4. Implementation and Results

Chapter 5. Discussion and Conclusion

- 5.1 Key Findings
- 5.2 GitHub Link
- 5.3 Video Demonstration Link
- 5.4. Deployed Render Link
- 5.5. Limitations

5.6.. Future Work

5.7. Conclusion

References

Appendices

LIST OF FIGURES

		Page No.
Figure 1	System Architecture Flowchart	10
Figure 2	Home Page 1	13
Figure 3	Home Page 2	13
Figure 4	All Doctors Page	14
Figure 5	About Us Page	14
Figure 6	Dashboard	15
Figure 7	Appointment Page	15
Figure 8	Add Doctors Page	16
Figure 9	Doctor Lists Page	16

CHAPTER 1

Introduction

1.1 Problem Statement

The current process of booking doctor appointments is often done manually, which is not only time-consuming but also inefficient and prone to errors. Patients face difficulties in finding the right doctor based on their specific needs, such as specialization or availability. On the other hand, doctors and administrators often lack a centralized system to manage schedules, patient records, and other critical data. This leads to confusion, delays, and mismanagement, negatively affecting the overall experience of healthcare services. Addressing this problem is essential to improve the efficiency, reliability, and accessibility of appointment scheduling for both patients and healthcare providers.

1.2 Motivation:

This project was chosen to address the challenges faced by patients, doctors, and administrators in managing doctor appointments. With the growing reliance on technology, an online system can significantly simplify the process of scheduling and managing appointments. The motivation comes from the need to create a platform that makes healthcare more accessible and organized. This system not only benefits individual clinics and hospitals but can also be scaled for larger healthcare networks. Additionally, the project serves as an opportunity to apply the MERN stack for developing a complete web application, combining practical learning with real-world impact. The potential applications of the project include improving patient-doctor interaction, reducing administrative workload, and enhancing the overall quality of healthcare services.

1.3 Objective:

The main objective of this project is to design and implement an online doctor appointment booking system that is user-friendly, reliable, and efficient. The system aims to:

- Allow patients to easily search for doctors based on specialization or availability and book appointments.
- Provide doctors with tools to manage their profiles, schedules, and appointments efficiently.
- Enable administrators to oversee the platform, manage doctor profiles, and cancel or modify appointments when necessary.

The focus is to build a system that simplifies the process for all users while ensuring data security and smooth functionality.

1.4 Scope of the Project:

The project is designed to create a functional and scalable system for managing doctor appointments. The platform includes:

- **Patient Features:** Searching for doctors by specialty, booking appointments, and accessing appointment details.
- **Doctor Features:** Managing profiles, updating availability, and viewing appointments.
- **Admin Features:** Adding, updating, and removing doctor information, as well as managing appointment cancellations.

The current scope does not include advanced features like online payments or teleconsultations, as the focus is on building a robust foundation. These features, along with others like multilingual support and AI-based doctor recommendations, can be added in future iterations.

CHAPTER 2

Literature Survey

2.1 Review relevant literature or previous work in this domain.

Several systems and methodologies have been developed to address the problem of managing doctor appointments, each with unique features and limitations. These solutions range from simple appointment scheduling software to comprehensive healthcare management systems. Below is a summary of existing work, techniques, and methodologies related to the problem.

Existing Models and Techniques

- **Manual Appointment Systems:**

Traditional methods involve in-person or telephone-based appointment bookings. While simple, they lack efficiency, are prone to errors, and often lead to scheduling conflicts.

- **Standalone Appointment Software:**

Applications such as Calendly and TimeTrade focus on general appointment scheduling. These tools lack healthcare-specific features like patient history integration, doctor specialization filters, or profile management.

- **Healthcare Management Platforms:**

Comprehensive platforms like Practo and Zocdoc include features for searching doctors, booking appointments, and managing patient records. These systems are effective but are often subscription-based, making them less accessible for smaller clinics or individual practitioners.

- **Online Portals with Payment Integration:**

Some platforms incorporate online payment gateways for convenience, but these are limited by integration challenges and security concerns.

- **AI and ML-Based Solutions:**

Advanced systems use AI for doctor recommendations and chatbots for patient support.

2.2 Mention any existing models, techniques, or methodologies related to the problem.

Cost Barrier: Many comprehensive platforms are subscription-based, making them unaffordable for small-scale clinics or individual doctors.

Complexity: Some systems are overly complex, requiring extensive training to use effectively.

Customization: Limited customization options for tailoring the system to specific needs.

Scalability: Existing solutions may not be easily scalable to accommodate growing user bases or feature expansions.

Security: Many platforms do not prioritize data security, putting sensitive patient information at risk.

2.3 Highlight the gaps or limitations in existing solutions and how your project will address them.

This project provides a simpler, cost-effective solution specifically tailored to healthcare needs:

1. Customizability:

- The system is built using the MERN stack, allowing for future scalability and customization.

2. Healthcare-Specific Features:

- Focus on features like filtering doctors by specialization, managing doctor profiles, and simplifying the booking process.

3. User-Friendliness:

- Intuitive interfaces for patients, doctors, and administrators, reducing the learning curve.

4. Data Security:

- Incorporates bcrypt for password hashing and JWT for secure authentication, ensuring data protection.

5. Future-Ready Design:

- While basic functionalities like booking and profile management are the focus, the system is designed to integrate advanced features like teleconsultation, online payment, and AI-based recommendations in the future.

CHAPTER 3

Proposed Methodology

The methodology involves building a web-based doctor appointment booking system using the MERN stack. The development process includes:

- **Requirement Analysis:** Identifying user needs for patients, doctors, and administrators.
- **System Design:** Developing the architecture for the platform, including frontend, backend, and database layers.
- **Module Development:** Implementing key modules such as appointment booking, profile management, authentication, and admin panel functionalities.
- **Testing:** Conducting unit, integration, and usability testing to ensure a bug-free, user-friendly platform.
- **Deployment:** Hosting the application on a suitable platform for public access.

3.1 System Design

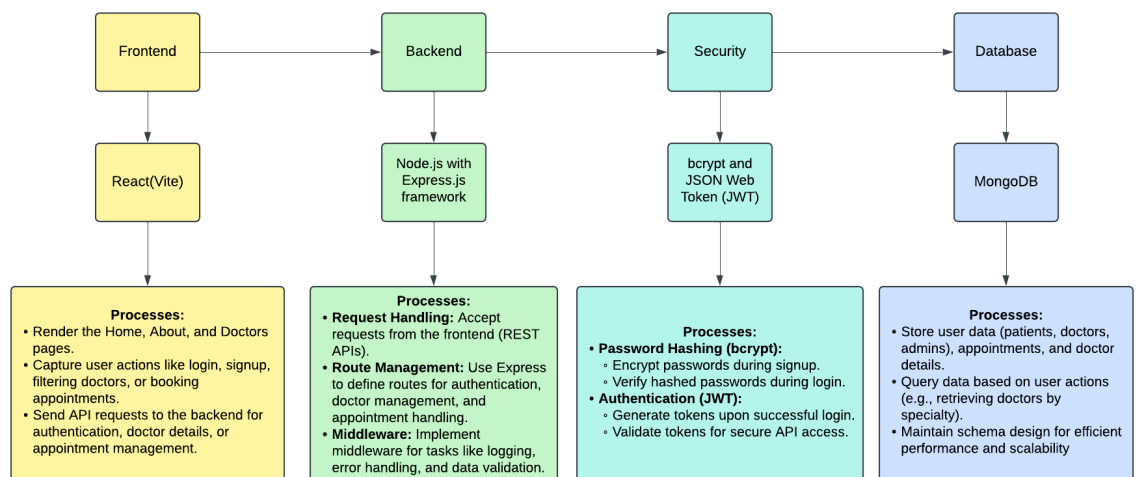


Figure 1: System Architecture Flowchart

Frontend (Client-Side): Built using React.js, providing a dynamic, responsive, and interactive user interface.

Features implemented include:

Home Page: Displays general information about the application and its functionality.

About Page: Provides details about the platform, its purpose, and how to use it.

Doctors Page: Lists available doctors with filters for specialties, making it easy for patients to find the right doctor.

Backend (Server-Side): Developed using Node.js with Express.js as the framework for API routing and request handling.

Features implemented include:

Authentication: Used bcrypt for securely hashing passwords and JWT (JSON Web Tokens) for user authentication and session management.

Doctor Management: APIs to add or remove doctors.

Appointment Handling: APIs for creating, updating, and deleting patient appointments.

Database: Used MongoDB for storing user data (patients, doctors, and admin), appointment details, and doctor information. Ensured schema design optimizes queries for better performance.

Appointment Booking: Allows patients to schedule appointments with their chosen doctors.

User Authentication: Enables login and signup functionality for patients and doctors.

Admin Panel: A dedicated section of the frontend built to provide admin-level control and management.

Features implemented include:

Manage Appointments: Allows the admin to cancel or modify appointments as required.

Doctor Management: Enables the admin to add, update, or remove doctor profiles and their availability.

System Monitoring: Admin can view overall platform statistics (optional for advanced tracking).

3.2 Modules Used

Authentication Module:

- Handles secure login and signup for patients, doctors, and admins.

Doctor Management Module:

- Allows adding, editing, and removing doctor profiles and schedules.

Appointment Booking Module:

- Enables patients to search for doctors by specialty and book appointments.

Admin Module:

- Provides tools to manage doctors and appointments effectively.

3.3 Advantages

- Simplifies appointment scheduling for patients.
- Centralized system for managing doctor profiles and appointments.
- Secure data handling using encryption and token-based authentication.
- Scalable and customizable for future enhancements.
- Accessible across devices with a responsive design.

3.4 Software Requirements

1. Frontend Technologies:

- React.js
- React-Router-DOM
- Axios
- React-Toastify

2. Backend Technologies:

- Node.js
- Express.js
- Bcryptjs
- Cloudinary
- Jsonwebtoken
- CORS

3. Database:

- MongoDB

4. Development Tools:

- Visual Studio Code
- Postman for API testing

5. Hosting Platform:

- Render and Veluru

6. Version Control:

- Git and GitHub for managing source code.

CHAPTER 4

Implementation and Result

User Panel:-

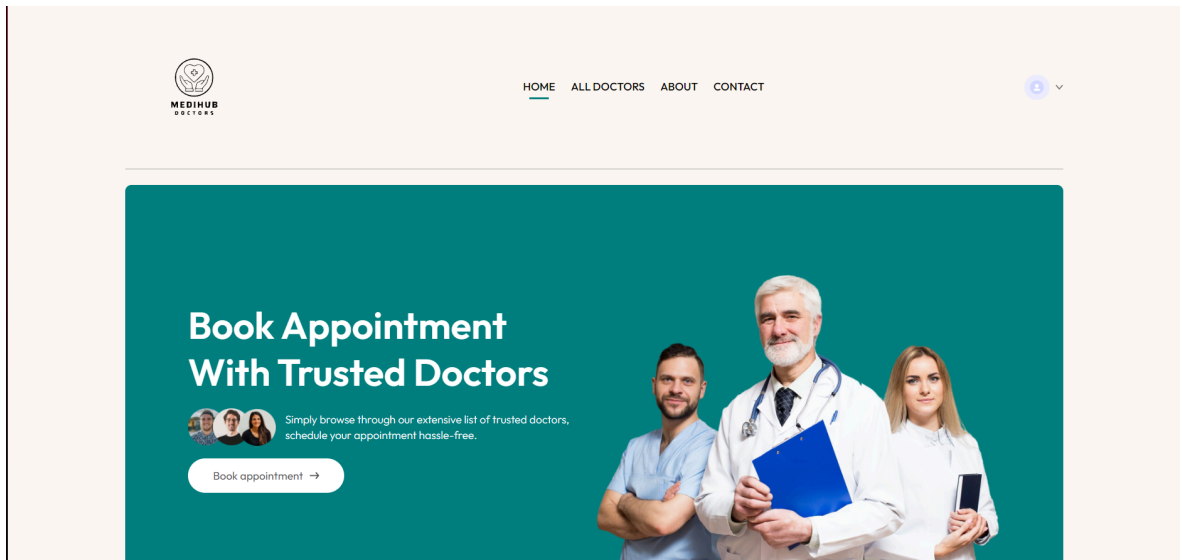


Figure 2: Home Page 1

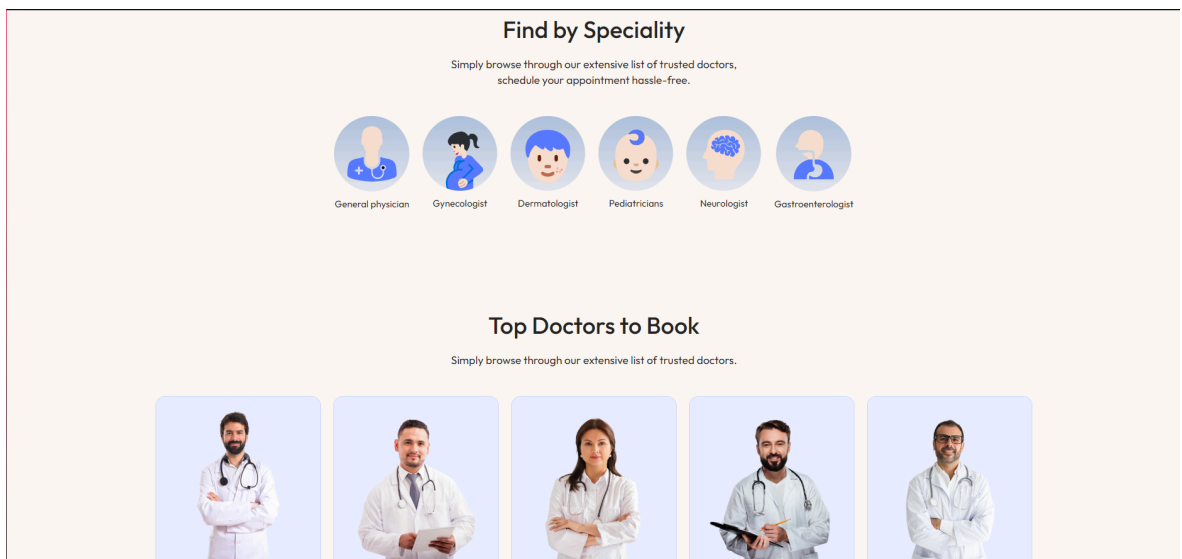


Figure 3: Home Page 2

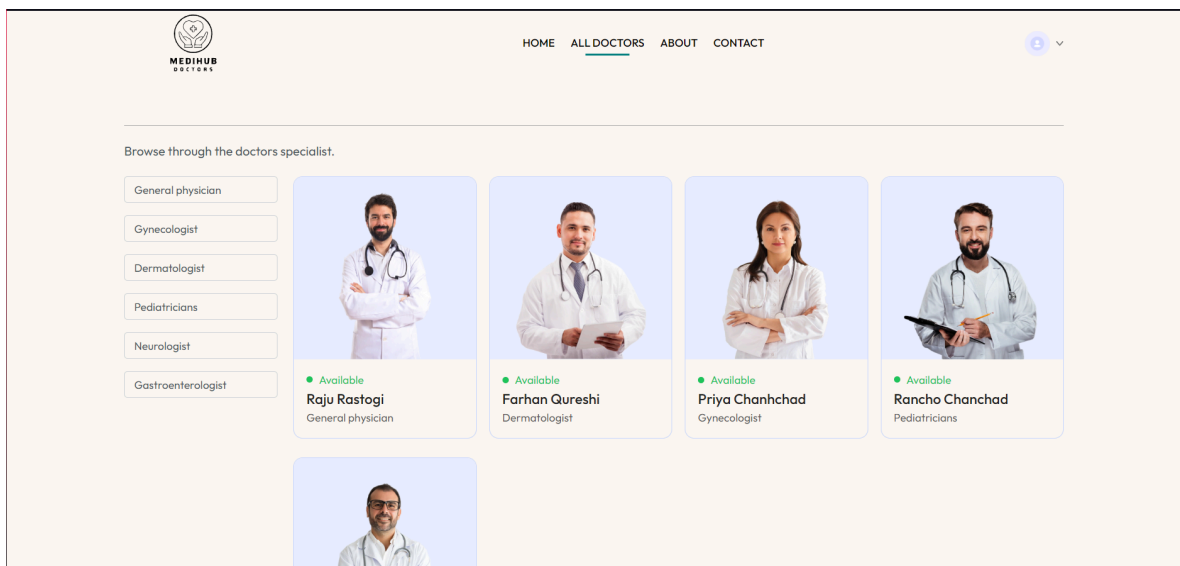


Figure 4: All Doctors Page

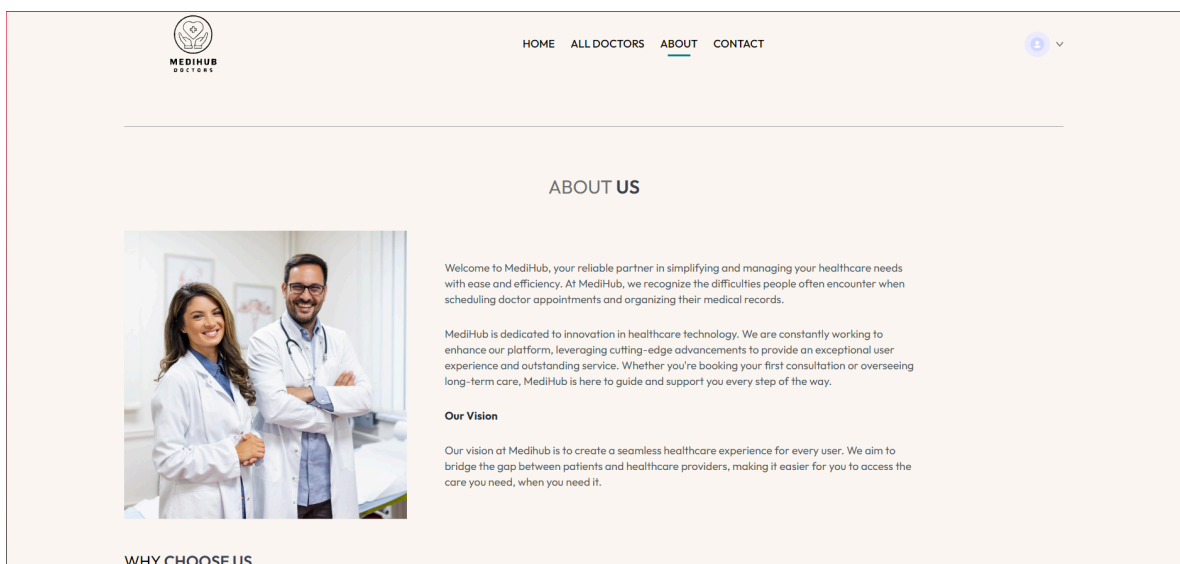


Figure 5: About Us Page

Admin Panel:-

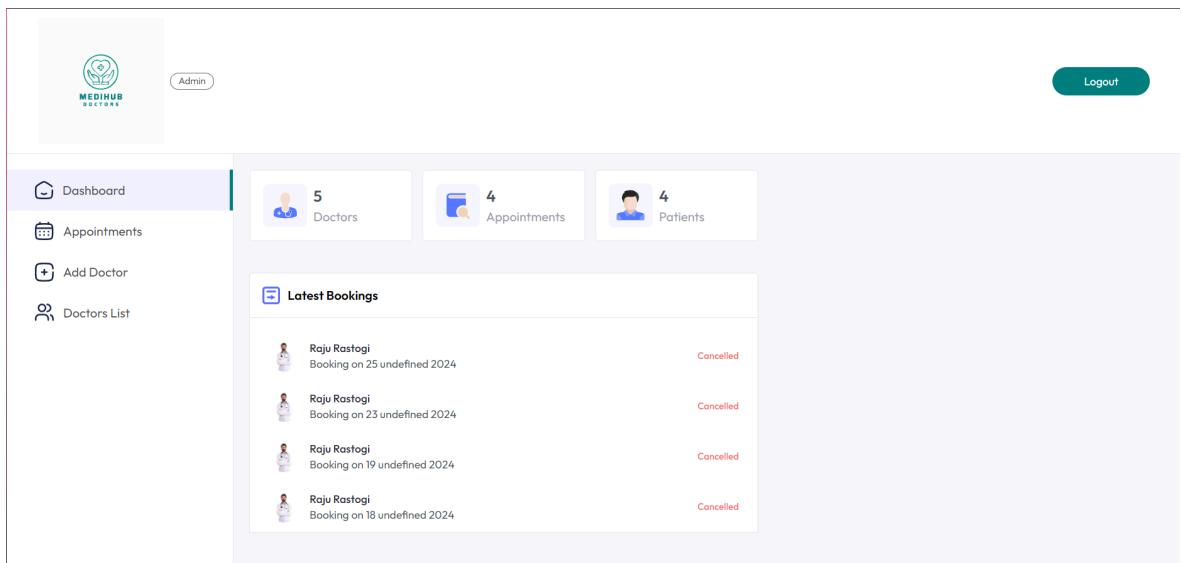


Figure 6: Dashboard

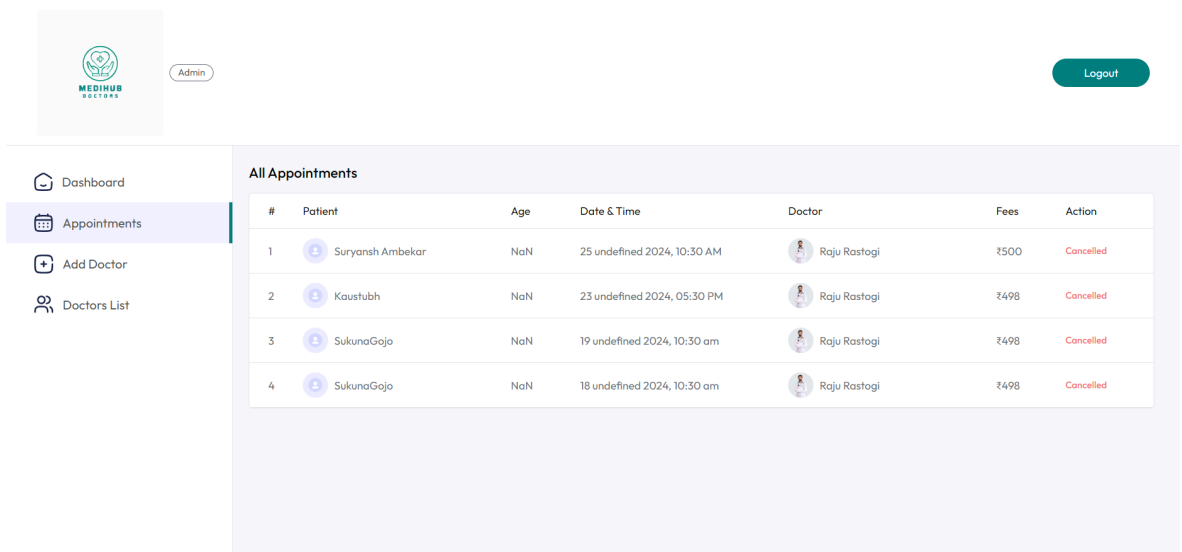
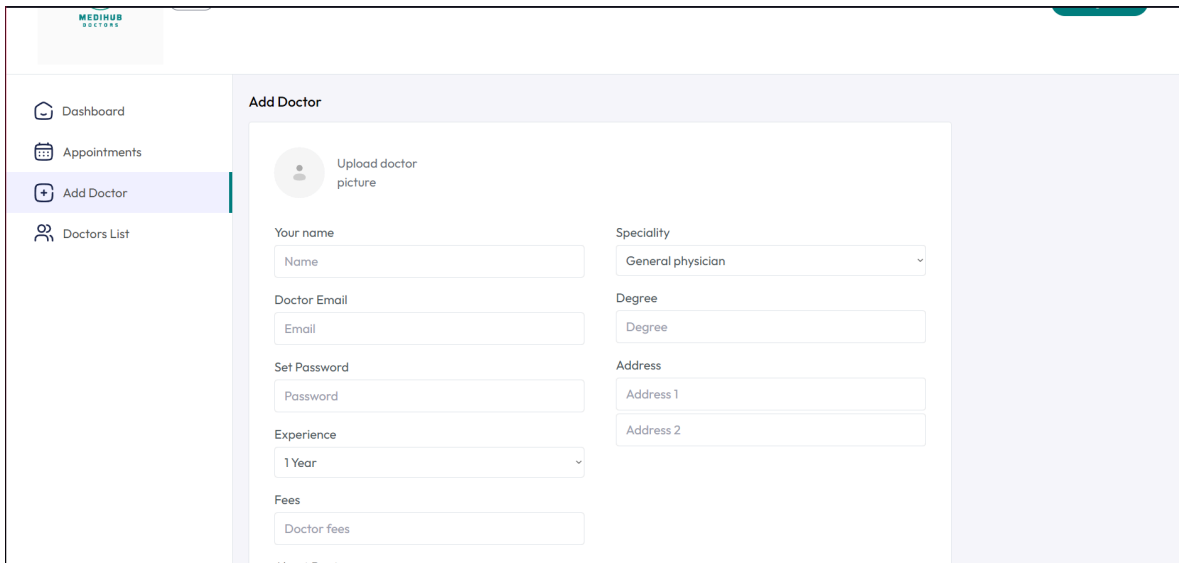


Figure 7: Appointment page



MEDIHUB DOCTORS

- Dashboard
- Appointments
- Add Doctor**
- Doctors List

Add Doctor

Upload doctor picture

Your name
Name

Speciality
General physician

Doctor Email
Email

Degree
Degree

Set Password
Password

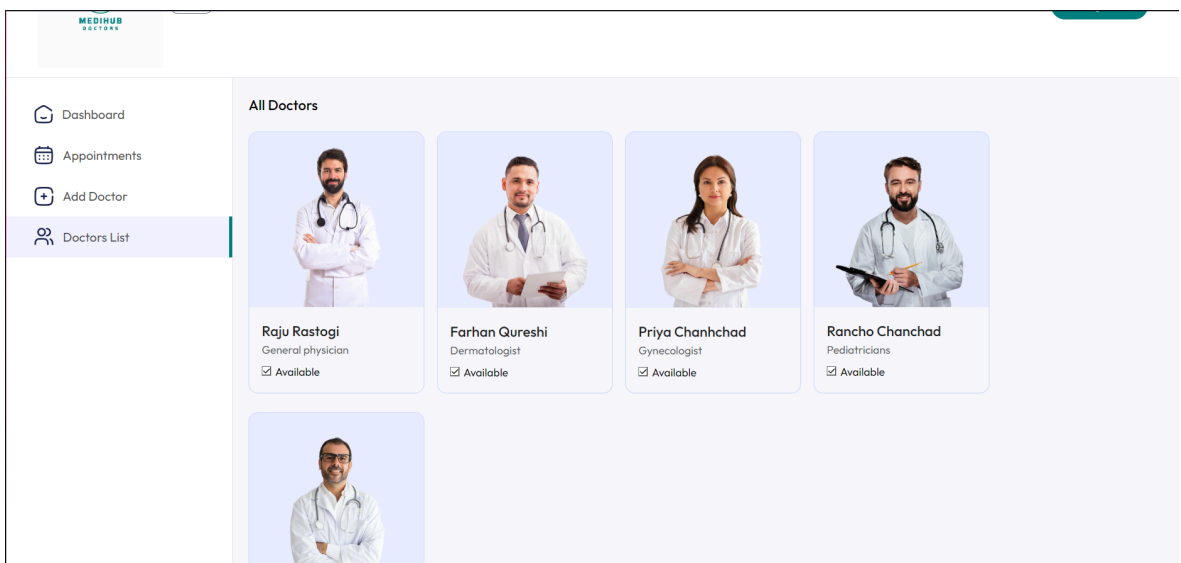
Address
Address 1
Address 2

Experience
1 Year

Fees
Doctor fees

About Doctor

Figure 8: Add Doctor Page



MEDIHUB DOCTORS

- Dashboard
- Appointments
- Add Doctor
- Doctors List**

All Doctors






 Raju Rastogi General physician <input checked="" type="checkbox"/> Available	 Farhan Qureshi Dermatologist <input checked="" type="checkbox"/> Available	 Priya Chanhchad Gynecologist <input checked="" type="checkbox"/> Available	 Rancho Chanhchad Pediatricians <input checked="" type="checkbox"/> Available
			

Figure 9: Doctor List Page

CHAPTER 5

Discussion and Conclusion

5.1 Key Findings:

The doctor appointment booking system effectively simplifies appointment scheduling and management for patients, doctors, and administrators. It provides an intuitive interface that allows patients to search for doctors using filters like specialty, enhancing usability and efficiency. Administrators benefit from the platform's admin panel, which enables seamless management of doctor profiles and appointments. The system also ensures secure user authentication and data handling, protecting sensitive information. By leveraging technologies such as React, Node.js, and MongoDB, the project achieves scalability and demonstrates the potential for further enhancement.

5.2 Git Hub Link of the Project:

<https://github.com/kaudrubh/b9gWNX4YPifGj5efwQ2cvx.git>

5.3 Video Recording of Project:

https://drive.google.com/drive/folders/1sYFPTPyEjy0BD-zOA5DkWic3xyZEGV_-?usp=sharing

5.4 Deployed Render Link:

1. **User Page:-**<https://b9gwnx4ypifgj5efwq2cvx-frontend.onrender.com>
2. **Admin Page:-**<https://b9gwnx4ypifgj5efwq2cvx-frontend.onrender.com>

5.5 Limitations:

The current system has certain limitations that need to be addressed. It lacks online payment integration, which reduces its usability for scenarios requiring pre-paid bookings. Appointment management features are basic and do not include functionalities like automated reminders or real-time availability updates. Additionally, the admin panel does not provide detailed analytics or reporting capabilities, which limits its usefulness for monitoring system performance. The system may also encounter challenges with scalability as user volume increases, necessitating further optimization.

5.6 Future Work:

Future developments can significantly enhance the functionality of the system. Integrating an online payment gateway will allow patients to make secure payments while booking appointments. Adding advanced features such as SMS

or email reminders for appointments and real-time updates for doctor availability will improve user experience. A virtual consultation feature can be incorporated to enable remote interactions between doctors and patients. Expanding the admin panel to include detailed analytics and performance reports will aid in better decision-making. Additionally, creating a mobile application will improve accessibility, making the system more versatile and user-friendly.

5.7 Conclusion:

This project successfully addresses the challenges of manual appointment scheduling by providing a user-friendly, secure, and efficient digital platform. It has demonstrated how modern web technologies can be used to develop scalable solutions for healthcare management. While there are areas for improvement, the system serves as a strong foundation for future enhancements. Its overall impact lies in streamlining healthcare processes, improving user convenience, and setting a benchmark for similar applications in other domains.

REFERENCES

- [1]. •React Documentation: <https://reactjs.org/docs/getting-started.html>
- [2]. •Node.js Documentation: <https://nodejs.org/en/docs/>
- [3]. •Express Documentation: <https://expressjs.com/>
- [4]. •MongoDB Documentation: <https://www.mongodb.com/docs/>
- [5]. •JWT Authentication: <https://jwt.io/introduction>
- [6]. •MERN Stack Tutorials: <https://www.freecodecamp.org/news/learn-the-mern-stack/>
- [7]. •Web App Security Best Practices: <https://owasp.org/>
- [8]. •Agile Development for Projects: <https://www.atlassian.com/agile>

Appendices

- **GitHub Link of the Project:**

<https://github.com/kaudrubh/b9gWNX4YPifGj5efwQ2cvx.git>

- **Video Recording of Project:**

<https://drive.google.com/drive/folders/1sYFPTPyEjy0BD-zOA5DkWic3xyZEGV-?usp=sharing>

- **Deployed Render Link:**

3. **User Page:-**<https://b9gwnx4ypifgj5efwq2cvx-frontend.onrender.com>

4. **Admin Page:-**<https://b9gwnx4ypifgj5efwq2cvx-frontend.onrender.com>