DocSpot

Seamless Appointment Booking for Health

Team ID: LTVIP2025TMID53889

Internship: APSCHE - Short Term - Full Stack Developer (MERN Stack) STB3

Submission Date: June 29, 2025

Team Members:

Bodapati Ramadevi

Angam Mahidhar

Bhavaraju Raj Koushal

Routhu Manoj

1 Introduction

Healthcare systems often struggle to deliver seamless access to consultations, particularly in high-demand regions. Manual booking processes can be time-consuming and unstructured. Our project, **DocSpot**, aims to digitize and simplify this process using modern web technologies.

DocSpot is a MERN (MongoDB, Express, React, Node) stack-based full-stack web application that enables users (patients) to register, browse doctors, book appointments, and upload medical reports. It also allows doctors to manage bookings, and administrators to monitor the system and approve new doctors.

This project is developed as part of the APSCHE Short Term Internship in Full Stack Development (MERN Stack), under the guidance and learning modules provided by APSCHE and industry mentors.

2 Technology Stack

The application leverages the modern and scalable MERN stack:

- **Frontend (React.js):** Responsible for dynamic and interactive user interfaces. React Router is used for navigation, and Material UI enhances the visual design.
- **Backend (Node.js + Express.js):** Handles API endpoints, routing, and business logic. Middleware ensures secure authentication and file uploads.
- **Database (MongoDB):** A NoSQL database that stores user data, doctor information, appointment details, and uploaded documents.
- **Authentication:** JSON Web Tokens (JWT) manage secure sessions, while bcrypt ensures passwords are hashed and securely stored.

3 Features Implemented

- **User Registration and Login:** Separate login flows for patients, doctors, and admins with proper role-based access.
- **Doctor Search and Filters:** Patients can search doctors by specialty, experience, and availability.
- **Appointment Booking:** Real-time booking with confirmation messages and slot selection.
- **File Upload:** Patients can attach medical reports with appointments for better diagnosis.
- **Doctor Dashboard:** Doctors can view their upcoming appointments and manage availability.
- **Admin Panel:** Admins can approve or reject doctor applications and monitor the system.

4 Folder Structure

The codebase is modular and well-structured to separate concerns and support scalability.

```
projectFiles/
 client/        # React frontend with components, routes, pages
 server/        # Express backend with routes, models, controllers
 Document/      # PDF, reports, and documentation files
 VideoDemo/     # Contains a link to the project demo video
```

5 System Architecture & Workflow

The architecture of the DocSpot application follows a traditional MERN stack structure with clear separation of concerns across layers:

- **Frontend (React.js):** The UI is built using reusable components and navigated using React Router. It interacts with the backend via Axios for API calls.
- **Backend (Node.js + Express.js):** Acts as the controller, processing all client requests, performing business logic, and interacting with the database.
- **Database (MongoDB):** Stores structured documents for users, appointments, doctors, and uploaded files.
- **Authentication:** Uses JWT for session-based login and role-based access (patient, doctor, admin).

User Flow Overview:

1. A user registers and logs into the application.
2. The frontend sends requests to backend APIs for fetching or modifying data.
3. Based on the user role, access is granted to appropriate dashboards (Patient, Doctor, Admin).
4. Doctors manage their appointment slots, while patients book and upload reports.
5. Admins review new doctor accounts and manage platform activity.

6 Demo Video

A complete walkthrough of the project, including patient flow, doctor dashboard, and admin functionalities, is available at:
`https://drive.google.com/file/d/1ipkLDleI25x5kPPHsPR101QGFUxZ56pH/view?usp=sharing`

7 Challenges and Learnings

- **Version Control and Collaboration:** We learned to use Git and GitHub effectively for collaborative development and version management.
- **Authentication and Security:** Implementing JWT for session management and bcrypt for password encryption gave us insight into real-world security standards.

- **API Design and State Management:** Structuring backend APIs and handling frontend state in React using hooks was a key learning area.
- **Deployment Readiness:** We ensured the code was structured and documented well for real-world deployment and scalability.

8 Conclusion

DocSpot demonstrates the power and flexibility of full-stack development in solving practical healthcare scheduling challenges. From real-time booking to admin control, this project reflects teamwork, MERN stack mastery, and problem-solving skills.

With potential for real-world impact, the app can be enhanced further by:

- Adding a payment gateway for paid consultations
- Implementing a chat interface between patients and doctors
- Enabling real-time notifications and reminders

Submitted on: Sunday 29th June, 2025