SEAMLESS APPOINTMENT BOOKING FOR HEALTH

Introduction:

Welcome to our Health Booking Web App, your trusted platform for easy and hassle-free doctor appointments! With our user-friendly interface and advanced booking system, we aim to simplify the way you connect with healthcare professionals. Whether you need a routine check-up, specialist consultation, or follow-up care, our platform helps you find the right doctor based on your preferences such as specialty, location, and availability.

Browse through a wide range of verified doctors, select a convenient time slot with real-time availability, and book your appointment in just a few clicks. No more waiting on calls or visiting clinics just to schedule an appointment. Manage your bookings, reschedule when needed, and receive instant confirmations and updates — all from the comfort of your home.

Experience smarter, faster, and more reliable healthcare appointment booking with our Health Booking Web App today!

Description:

Our Health Booking Web App is more than just an appointment scheduling platform—it's a complete healthcare management solution designed to simplify and enhance your medical experience. Imagine having access to trusted doctors and healthcare professionals right at your fingertips, anytime and anywhere.

From routine check-ups to specialist consultations, our platform connects patients with a wide network of verified doctors across various specialties. With our intuitive and user-friendly interface, you can easily browse doctors based on specialty, location, experience, and availability. Finding the right healthcare provider has never been easier.

Our advanced real-time booking system allows you to select appointment slots that perfectly match your schedule. Whether you prefer morning, evening, or weekend appointments, our app offers flexible options to accommodate your needs. You can upload medical records, insurance documents, and receive instant confirmation notifications—eliminating the need for long phone calls or waiting in queues.

Beyond booking, our platform empowers users to manage their appointments efficiently. Patients can reschedule or cancel appointments, track booking history, and receive timely reminders. Doctors can manage their schedules, confirm appointments, and update consultation details seamlessly. Meanwhile, the admin ensures smooth platform operations, verifies doctor registrations, and maintains compliance with privacy and service policies.

After consultations, patients can receive visit summaries, prescriptions, and follow-up instructions directly through the app, making healthcare management organized and stress-free.Our commitment goes beyond convenience—we focus on reliability, security, and quality healthcare access for everyone.Experience smarter, faster, and more connected healthcare with our Health Booking Web App—bringing medical care closer to you, whenever you need it.

Scenario Based Case Study:

Meet Ananya, a dedicated school teacher with a busy academic schedule. Between managing classes, preparing lesson plans, and supporting her students, Ananya often struggles to find time for her regular medical check-ups. Although she values her health, booking doctor appointments through traditional methods feels time-consuming and inconvenient.

➢ Ananya's Solution: The Health Booking Web App

Ananya discovers the Health Booking Web App, a reliable and efficient platform designed to simplify the process of scheduling doctor appointments. The app connects patients with verified healthcare professionals across various specialties.

User Registration and Authentication: Ananya registers on the app by entering her email address and creating a secure password. She logs in safely, ensuring her personal and medical information is protected.

Browsing Doctors: After logging in, Ananya accesses her dashboard, where she can view a list of available doctors. She filters them based on specialty (such as General Physician), location, availability, and patient ratings to find the most suitable doctor.

Booking an Appointment: Once she selects a doctor, Ananya clicks on "Book Now." She chooses a convenient date and time using the real-time availability feature. She also uploads her previous medical reports for reference. After submitting her request, she receives a notification confirming that her appointment request has been sent.

Appointment Confirmation: The doctor reviews Ananya's request and confirms the appointment. She receives a confirmation message with full details including date, time, and clinic location.

Appointment Management: Ananya can track her upcoming appointments through the dashboard. If necessary, she can easily reschedule or cancel the appointment without needing to call the clinic.

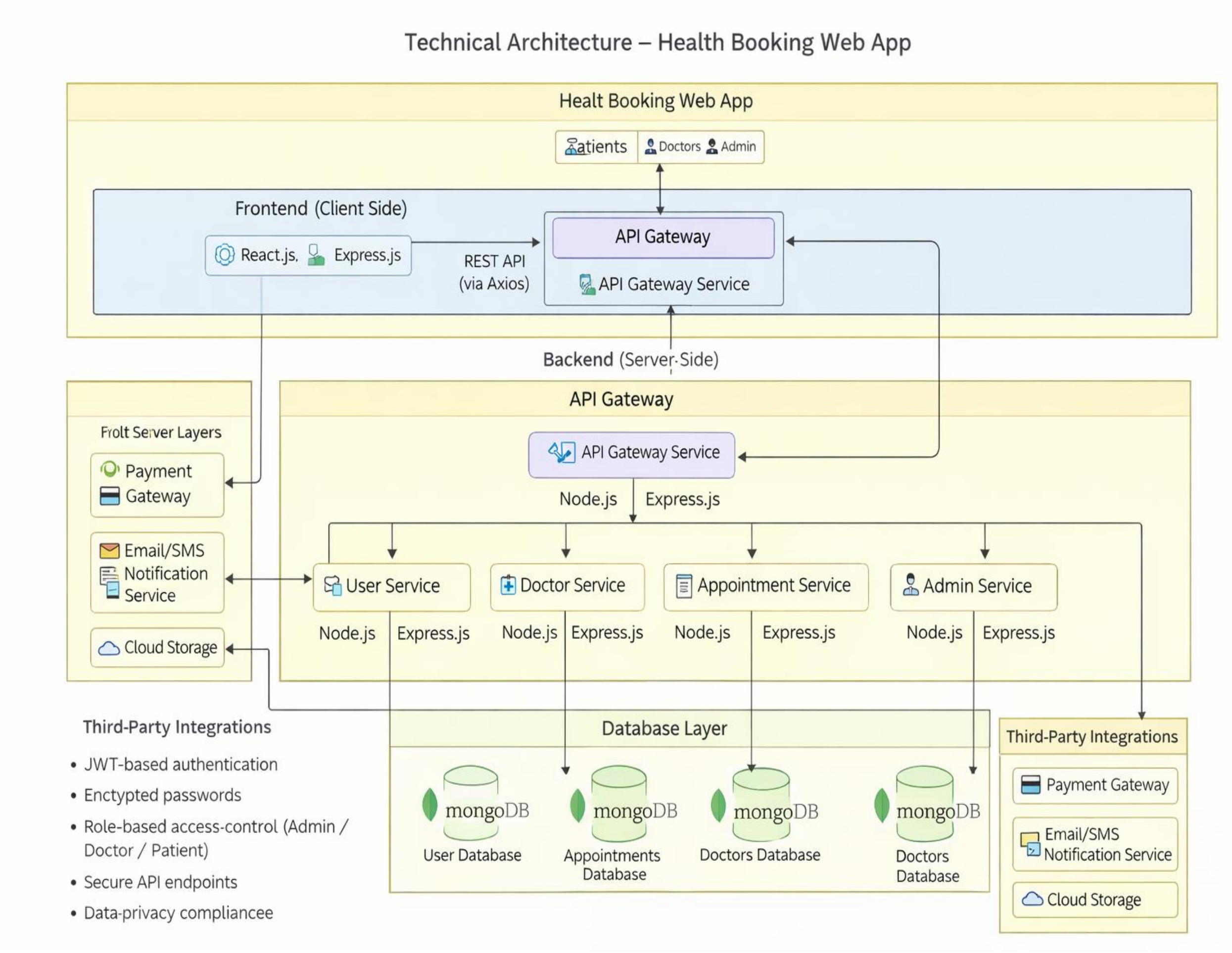
Doctor's Role: The approved doctor logs into the system, views scheduled appointments, confirms bookings, and manages patient records efficiently.

Admin Management: Meanwhile, the admin oversees platform operations, verifies new doctor registrations, ensures compliance with healthcare standards, and maintains system integrity.

Consultation and Follow-Up: On the day of the appointment, Ananya visits the doctor for her consultation. After the visit, the doctor updates her medical records and provides prescriptions or follow-up instructions through the app.

Ananya's Experience: With the Health Booking Web App, Ananya can manage her healthcare conveniently despite her busy teaching schedule. The platform saves her valuable time, reduces stress, and ensures she receives timely medical attention whenever needed.

Technical Architecture:-



The technical architecture of the Health Booking Web App follows a client-server model, where the frontend represents the client and the backend serves as the server. The frontend is responsible for the user interface, user interaction, and presentation layer, allowing patients, doctors, and administrators to access the system easily. It enables users to register, log in, browse doctors, book appointments, upload medical documents, and manage their schedules through an intuitive and responsive interface.

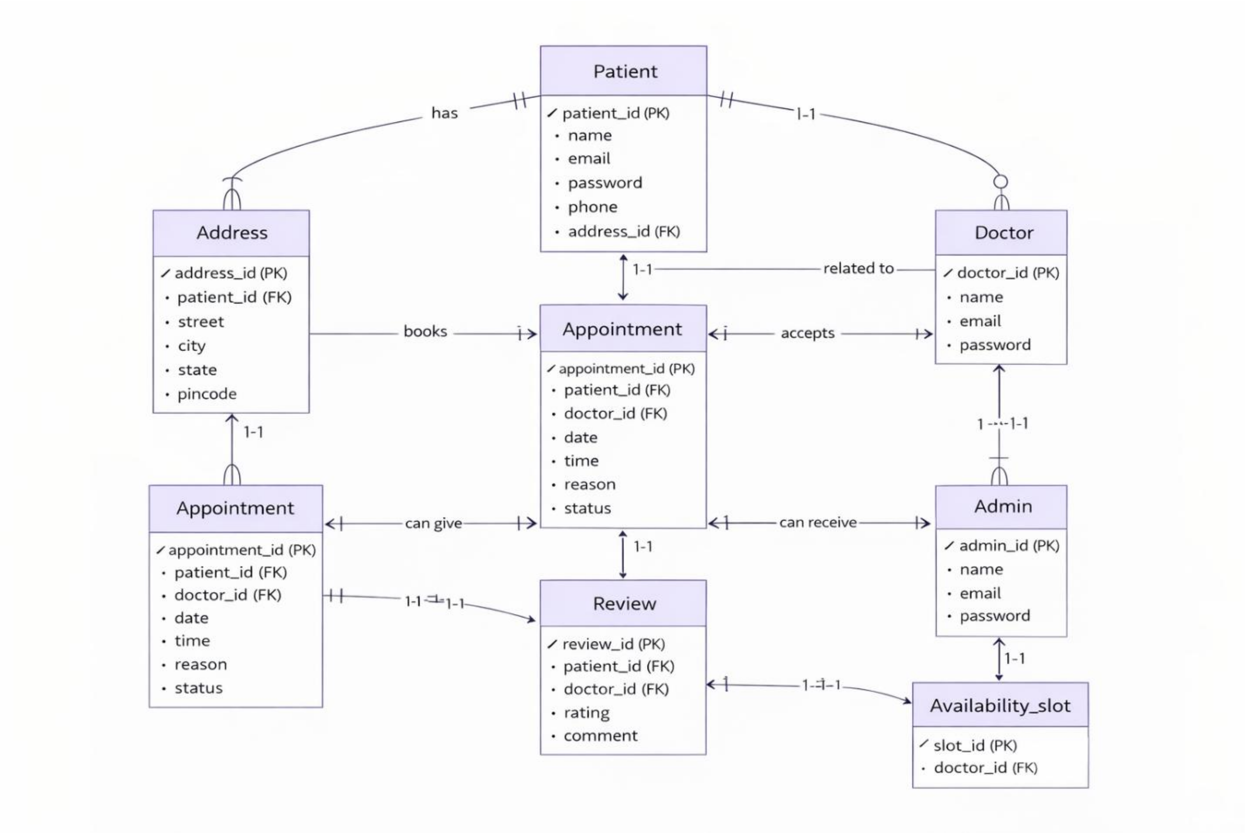
The backend handles data storage, business logic, authentication, appointment scheduling, and system management. It processes user requests, verifies doctor registrations, manages appointment confirmations, and ensures secure access control for different roles such as Admin, Doctor, and Patient.

Communication between the frontend and backend is facilitated through RESTful APIs, typically using Axios for HTTP requests. These APIs enable secure and seamless data exchange between the client and server, ensuring real-time appointment updates and notifications.

For data storage and retrieval, the system uses MongoDB, which efficiently stores user profiles, doctor information, appointment records, medical documents, and booking history. The backend may also integrate with external services such as payment gateways for consultation fees, email/SMS notification services for appointment reminders, and cloud storage for securely storing uploaded medical records.

This architecture ensures scalability, security, and smooth functionality, providing a reliable and efficient healthcare appointment booking experience for all users.

ER-Diagram:



The technical architecture of the Doctor Appointment Web Application follows a client-server model, where the frontend acts as the client and the backend functions as the server.

The frontend is responsible for providing the user interface, handling user interactions, and presenting information to patients, doctors, and administrators. It allows users to register, log in, search for doctors, view doctor profiles, book appointments, and manage their appointment history. The frontend is developed using technologies such as HTML, CSS, JavaScript, and React (if implemented).

The backend manages the core functionality of the system, including business logic, appointment scheduling, user authentication, and database operations. It ensures secure login using JWT authentication, prevents double booking of appointment slots, and handles role-based access control for patients, doctors, and administrators. The backend is built using Node.js and Express.js.

The system uses MongoDB as the database to store user details, doctor information, appointment records, availability schedules, and admin data. The backend communicates with the database using Mongoose for efficient data modeling and querying.Communication between the frontend and backend is facilitated through RESTful APIs, enabling secure and seamless data exchange over HTTP/HTTPS. The application may also integrate with external services such as email or SMS notification systems to send appointment confirmations and updates.This layered architecture ensures scalability, security, and efficient management of the doctor appointment booking process.

Key Features:

Doctor Directory: Our DocSpot app provides a comprehensive doctor directory with various specialties and categories. Users can easily search, browse, and filter doctors based on their preferences, such as specialty, location, experience, and availability, making it easy to find the right healthcare provider.

Appointment Booking and Confirmation: The app includes an appointment booking feature that enables users to select doctors, choose available time slots, and book appointments. The system provides real-time availability and sends appointment confirmation notifications, ensuring a smooth and convenient booking experience.

Doctor Reviews and Ratings: Patients can provide feedback and rate doctors based on their consultation experience. This helps other users make informed decisions and builds trust and transparency within the platform.

Appointment Tracking: Once an appointment is booked, users can track its status in real time. They receive updates such as pending, confirmed, completed, or cancelled, providing clarity and peace of mind.

Patient Dashboard: The application offers a personalized dashboard for patients to manage their activities. Users can view appointment history, upcoming appointments, cancel or reschedule bookings, and access prescriptions and medical records.

Doctor Dashboard: Doctors have access to a dedicated dashboard where they can manage appointment requests, view schedules, confirm or reject bookings, update availability, and add consultation notes and prescriptions.

Admin Dashboard: For administrators, DocSpot provides a powerful dashboard to manage doctors, patients, and appointments. Admins can approve doctor registrations, monitor platform activities, and ensure smooth system operation.

Appointment Management: The system manages the complete appointment lifecycle, including booking, confirmation, consultation, and completion. Users and doctors can view appointment history and update appointment status when necessary.

Search and Filtering: Users can search doctors using keywords and apply filters such as specialization, location, availability, and ratings to quickly find suitable healthcare providers.

Medical Records Management: The platform allows doctors to upload prescriptions and consultation summaries, and patients can securely access and download their medical records anytime.

PRE REQUISITES:

Here are the key prerequisites for developing a full-stack application using Node.js, Express.js, MongoDB, and React.js:

- **Node.js and npm:**

Node.js is a powerful JavaScript runtime environment that allows you to run JavaScript code on the server side. It provides a scalable and efficient platform for building network applications.

Install Node.js and npm on your development machine, as they are required to run JavaScript on the server side.

Download: https://nodejs.org/en/download/

Installation instructions: https://nodejs.org/en/download/package-manager/

npm init

- **Express.js:**

Express.js is a fast and minimalist web application framework for Node.js. It simplifies the process of creating robust APIs and web applications, offering features like routing, middleware support, and modular architecture.

Install Express.js, a web application framework for Node.js, which handles server-side routing, middleware, and API development.

Installation: Open your command prompt or terminal and run the following command:

npm install express

- **MongoDB:**

MongoDB is a flexible and scalable NoSQL database that stores data in a JSON-like format. It provides high performance, horizontal scalability, and seamless integration with Node.js, making it ideal for handling large amounts of structured and unstructured data.

Set up a MongoDB database to store your application's data.

Download: https://www.mongodb.com/try/download/community

Installation instructions: https://docs.mongodb.com/manual/installation/

- **Moment.js:**

Moment.js is a JavaScript package that makes it simple to parse, validate, manipulate, and display date/time in JavaScript. Moment. js allows you to display dates in a human-readable format based on your location. Install React.js, a JavaScript library for building user interfaces.

Follow the installation guide: https://momentjs.com/

- **React.js:**

React.js is a popular JavaScript library for building user interfaces. It enables developers to create interactive and reusable UI components, making it easier to build dynamic and responsive web applications.

Install React.js, a JavaScript library for building user interfaces.

Follow the installation guide: https://reactjs.org/docs/create-a-new-react-app.html

- **Antd:**

Ant Design is a React. js UI library that contains easy-to-use components that are useful for building interactive user interfaces. It is very easy to use as well as integrate. It is one of the smart options to design web applications using react.

Follow the installation guide: https://ant.design/docs/react/introduce

- **HTML, CSS, and JavaScript**: Basic knowledge of HTML for creating the structure of your app, CSS for styling, and JavaScript for client-side interactivity is essential.

- **Database Connectivity**: Use a MongoDB driver or an Object-Document Mapping (ODM) library like Mongoose to connect your Node.js server with the MongoDB database and perform CRUD (Create, Read, Update, Delete) operations. To Connect the Database with Node JS go through the below provided link:

https://www.section.io/engineering-education/nodejs- mongoosejs-mongodb/

- **Front-end Framework**: Utilize Reactjs to build the user-facing part of the application, including entering booking room, status of the booking, and user interfaces for the admin dashboard.
- For making better UI we have also used some libraries like material UI and boostrap.
- Install Dependencies
- Navigate into the cloned repository directory:
- cd book-a-doctor
- Install the required dependencies by running the following commands:
- cd frontend
- npm install
- cd ../backend
- npm install
- Start the Development Server
- To start the development server, execute the following command:
- npm start
- The book a doctor app will be accessible at http://localhost:3000
- You have successfully installed and set up the online complaint registration and management app on your local machine. You can now proceed with further customization, development, and testing.

Roles and Responsibility

User:-

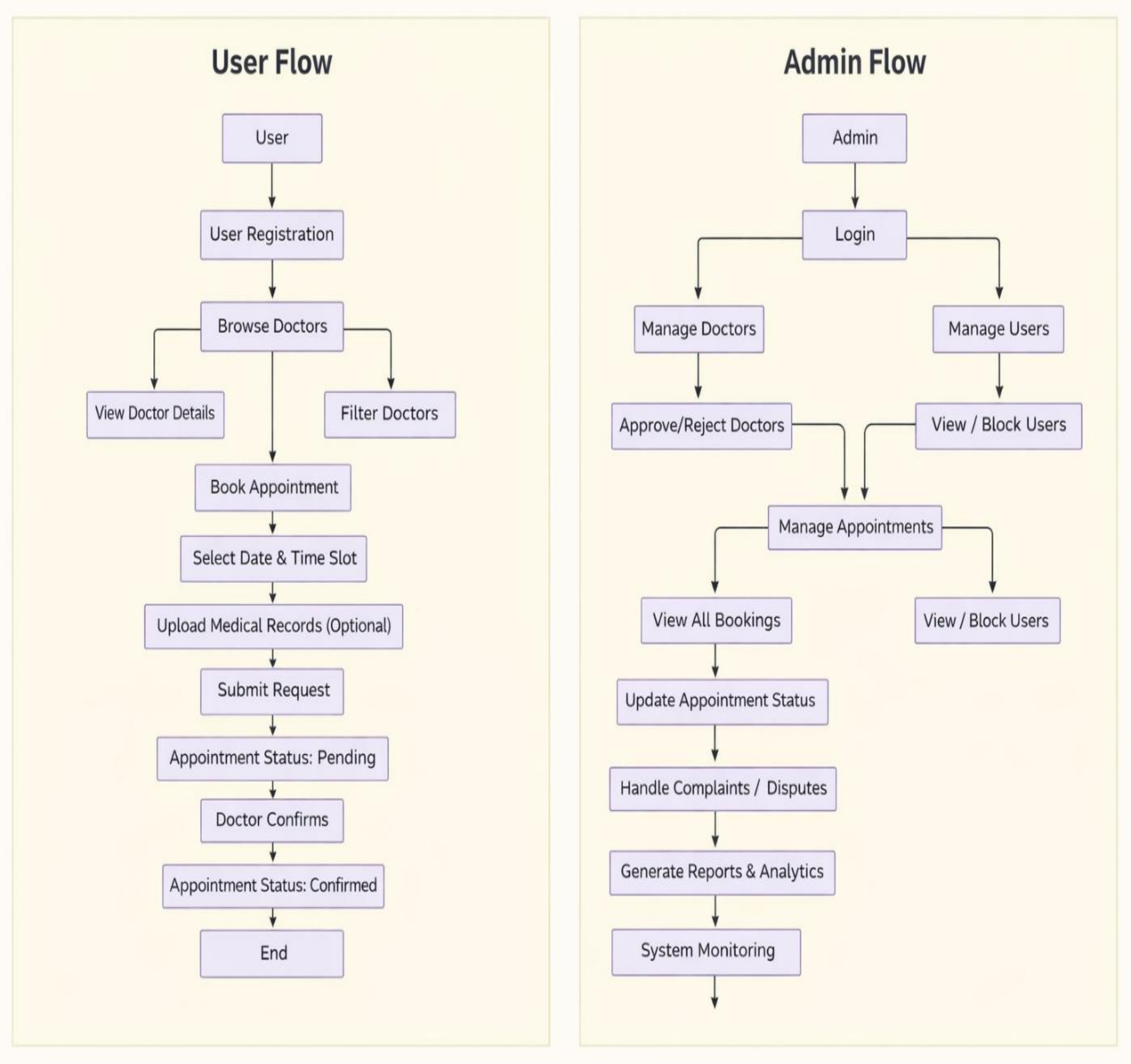
- Registration and Authentication: Users are responsible for creating an account on the DocSpot platform and securely logging in using their registered email and password to access booking and medical features.
- Profile Management: Users must provide accurate personal information such as name, contact details, age, and medical history (if required) and keep their profile updated.
- Browsing Doctors: Users can browse the list of available doctors, filter by specialty, location, availability, and ratings, and select a suitable healthcare provider.
- Appointment Booking: Users are responsible for selecting an appropriate date and time slot, uploading required medical documents (if any), and submitting the appointment request.
- Appointment Management: Users can view upcoming and past appointments, reschedule or cancel appointments within the allowed time frame, and monitor appointment status updates.
- Payment Responsibility (If Applicable): If online consultation fees are applicable, users are responsible for completing payments using the available payment methods provided by the platform.
- Medical Record Access: Users can view and download consultation summaries, prescriptions, and follow-up instructions shared by the doctor.

- Feedback and Reviews: After completing an appointment, users can provide ratings and reviews based on their consultation experience to help other patients make informed decisions.

Admin:-

- User Management: Admins are responsible for managing all user accounts on the platform, including patients and doctors. They can create, update, suspend, or delete accounts when necessary to ensure proper platform usage.
- Doctor Approval and Verification: Admins review and verify doctor registration requests by checking credentials, licenses, and qualifications before approving them to join the platform.
- Appointment Monitoring and Management: Admins can monitor all appointments booked on the platform, track their status (Pending, Confirmed, Completed, Cancelled), and intervene in case of scheduling conflicts or disputes.
- Platform Content Management: Admins manage informational content such as FAQs, health articles, privacy policies, terms and conditions, and other system-related pages to keep the platform updated and informative.
- Medical Data Oversight: While maintaining privacy, admins ensure that patient and doctor data is stored securely and handled according to healthcare data protection regulations.
- Analytics and Reporting: Admins generate reports and analyze system data, including
    1. Number of registered users
    2. Total appointments booked
    3. Doctor performance statistics
    4. Platform usage trends
- These insights help improve system efficiency and decision-making.
- Compliance and Security: Admins ensure that the platform complies with healthcare regulations, privacy laws, and data protection standards. They are responsible for maintaining system security and preventing unauthorized access.
- Dispute Resolution and Support: Admins handle user complaints, resolve appointment-related issues, and provide support to both patients and doctors to maintain a smooth experience.
- System Maintenance and Updates: Admins oversee system updates, feature improvements, bug fixes, and overall technical maintenance of the platform.
- Marketing and Growth Management: Admins can manage promotional campaigns, notifications, and awareness programs to attract new users and improve engagement on the platform.

Admin & User Flow:



The project flow for the DocSpot – Doctor Appointment Booking Web App involves user (patient) actions such as registering and logging in, browsing doctors, filtering by specialty or availability, viewing doctor details, booking an appointment, selecting a date and time slot, uploading medical records (if required), and submitting the appointment request. After confirmation by the doctor, the patient attends the consultation, receives a prescription or visit summary, and may provide feedback and ratings.

Admin actions include verifying and approving doctor registrations, managing patient and doctor accounts, monitoring all appointments, handling cancellations or disputes, updating appointment statuses, maintaining platform content, generating reports and analytics, and ensuring data security and compliance with healthcare regulations.

PROJECT FLOW:-

Before starting to work on this project, let’s see the demo.

Demo link:-
https://drive.google.com/file/d/1jqYHwLICgfFmfNVqGqNw2aI7IQSBOBuK/view?usp=sharing

Use the code in:-

https://drive.google.com/drive/folders/1DUkGI9UmXP-98C5DYMgH1k5Z1Dy9mC50?usp=sharing
or follow the videos below for better understanding.

Milestone 1: Project Setup and Configuration
1. Install required tools and software:
- Node.js.
- MongoDB.
- Create-react-app.

2. Create project folders and files:
- Client folders.
- Server folders.

3. Install Packages:
Frontend npm Packages
- Axios.
- React-Router-Dom.
- Bootstrap.
- React-Bootstrap.
- React-Icons.

Backend npm Packages
- Express.
- Mongoose.
- Cors.
- Dotenv.
- Jsonwebtoken.
- Bcryptjs.
- Nodemon.

Reference Link:-
https://drive.google.com/drive/folders/1DUkGI9UmXP-98C5DYMgH1k5Z1Dy9mC50?usp=sharing

Milestone 2: Backend Development:
- **Setup Express Server**
  1. Create **index.js** file in the backend folder.
  2. Create **.env file** and define:
     - PORT number
     - MongoDB URL
     - JWT Secret Key
  3. Configure the server by adding:
     - express
     - cors

    - dotenv
    - database connection
- **Implement Data Models**
  1. Define Mongoose schemas:
     - User → userModel.js
     - Doctor → docModel.js
     - Appointment → appointmentModel.js
  2. Store all schemas in:
     - schemas folder
  3. Implement CRUD operations:
     - Create user
     - Read doctor data
     - Book appointment
     - Update doctor profile
     - Delete appointment
- **Define API Routes**
  1. Create route files:
     - routes/userRoutes.js
     - routes/doctorRoutes.js
     - routes/adminRoutes.js
  2. Define routes for:
     - User registration
     - User login
     - Get doctors list
     - Book appointment
     - Get appointments
  3. Implement route handling using controllers:
     - controllers/userC.js
     - controllers/doctorC.js
     - controllers/adminC.js
- **User Authentication**
  1. Create routes and controllers for:
     - User registration
     - User login
  2. Use:
     - jsonwebtoken for token verification
- **Authentication Middleware**
  1. Create middleware file:
     - middlewares/authMiddleware.js
  2. Protect private routes such as:
     - Booking appointment
     - Viewing appointments
- **Error Handling**
  1. Implement error handling in:
     - controllers folder
     - index.js
  2. Return error messages such as:
     - Invalid login
     - User not found
     - Server error

3. Return HTTP status codes:
   - 200 – Success
   - 400 – Bad Request
   - 401 – Unauthorized
   - 500 – Server Error

Reference Link:-
https://drive.google.com/drive/folders/1DUkGI9UmXP-98C5DYMgH1k5Z1Dy9mC50?usp=sharing

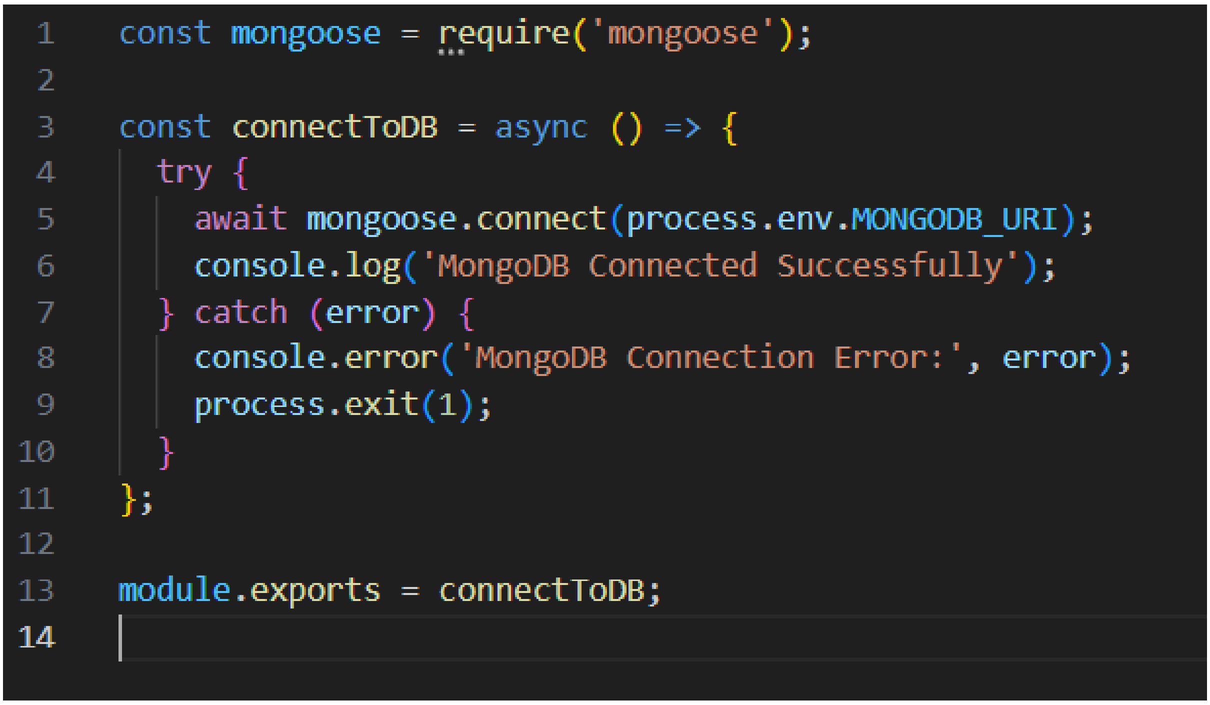
Milestone 3: Database:

1. Configure MongoDB:

- Install Mongoose.
- Create database connection.
- Create Schemas & Models.

2. Connect database to backend:

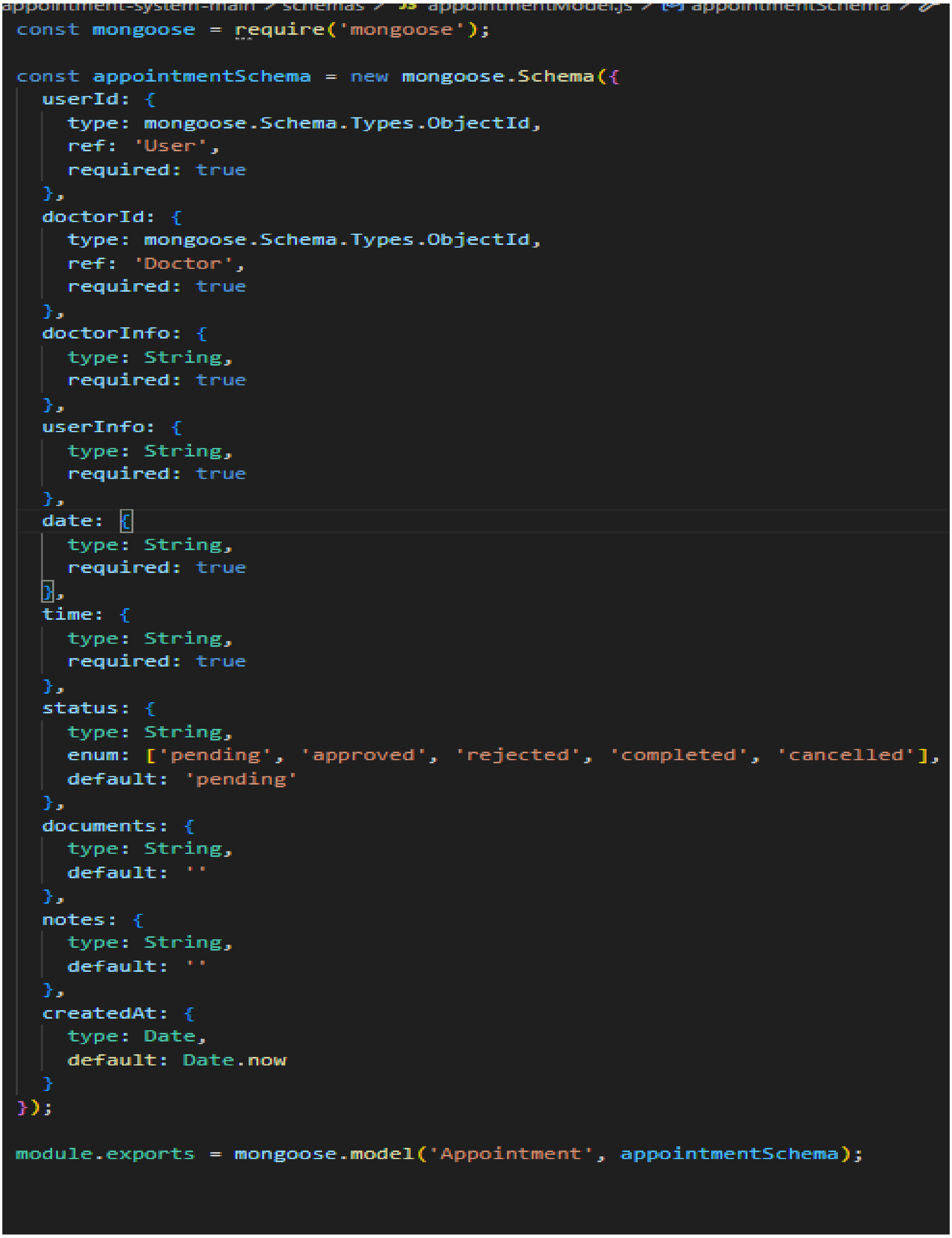
Now, make sure the database is connected before performing any of the actions through the backend. The connection code looks similar to the one provided below.



```
const mongoose = require('mongoose');

const connectToDB = async () => {
  try {
    await mongoose.connect(process.env.MONGODB_URI);
    console.log('MongoDB Connected Successfully');
  } catch (error) {
    console.error('MongoDB Connection Error:', error);
    process.exit(1);
  }
};

module.exports = connectToDB;

```

3. Configure Schema:

Firstly, configure the Schemas for MongoDB database, to store the data in such pattern. Use the data from the ER diagrams to create the schemas.
The schemas are looks like for the Application.



appointment-system-main > schemas > JS appointmentModel.js > [@] appointmentSchema >

```
const mongoose = require('mongoose');

const appointmentSchema = new mongoose.Schema({
  userId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'User',
    required: true
  },
  doctorId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'Doctor',
    required: true
  },
  doctorInfo: {
    type: String,
    required: true
  },
  userInfo: {
    type: String,
    required: true
  },
  date: {
    type: String,
    required: true
  },
  time: {
    type: String,
    required: true
  },
  status: {
    type: String,
    enum: ['pending', 'approved', 'rejected', 'completed', 'cancelled'],
    default: 'pending'
  },
  documents: {
    type: String,
    default: ''
  },
  notes: {
    type: String,
    default: ''
  },
  createdAt: {
    type: Date,
    default: Date.now
  }
});

module.exports = mongoose.model('Appointment', appointmentSchema);
```

Milestone 4: Frontend Development:

Installation of required tools:

- For frontend, we use:

1. React
2. Bootstrap
3. Material UI
4. Axios
5. Antd
6. mdb-react-ui-kit
7. react-bootstrap

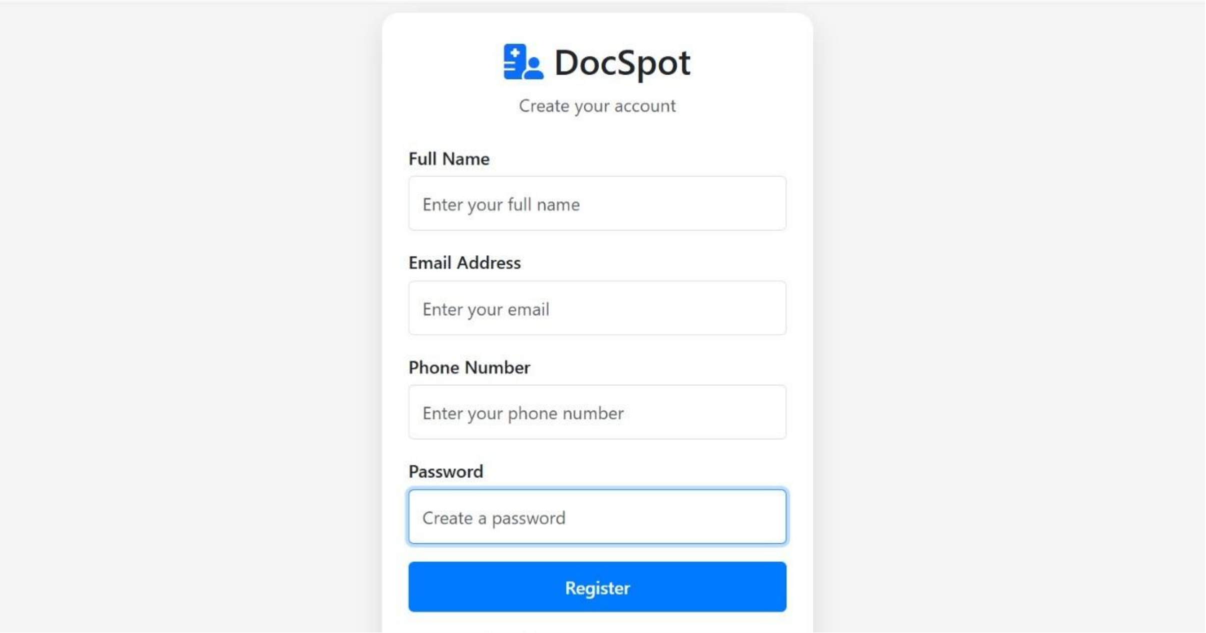
Reference:-

https://drive.google.com/drive/folders/10gN9iCfSRd3uUu3MEgFSMe6keG2n-kqF?usp=sharing

Milestone 5: Project Implementation:

Finally, after finishing coding the projects we run the whole project to test it's working process and look for bugs. Now, let's have a final look at the working of our Darshan Ease.

Login Page:-



DocSpot

Create your account

Full Name

Enter your full name

Email Address

Enter your email

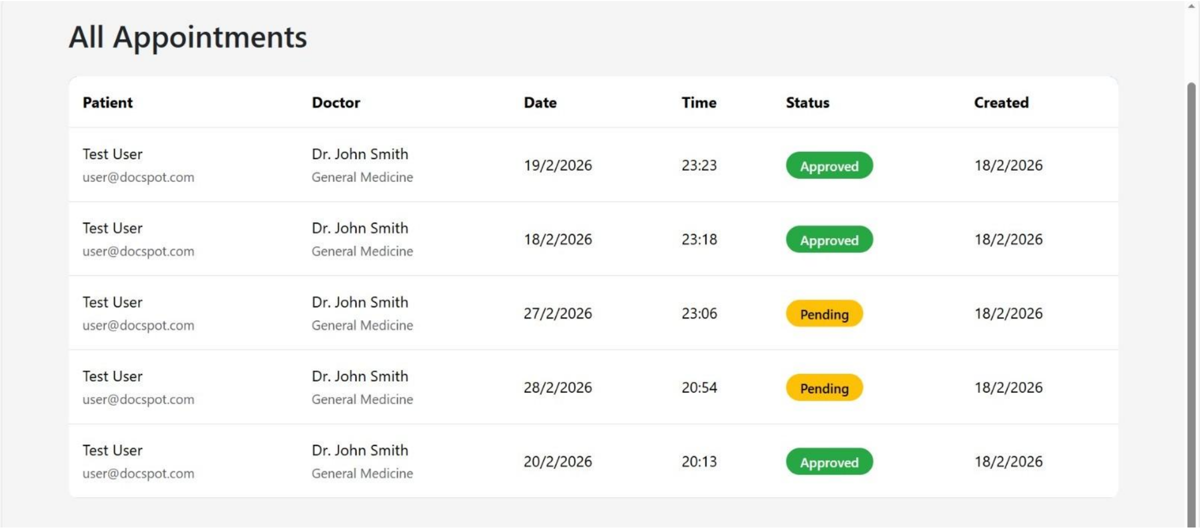
Phone Number

Enter your phone number

Password

Create a password

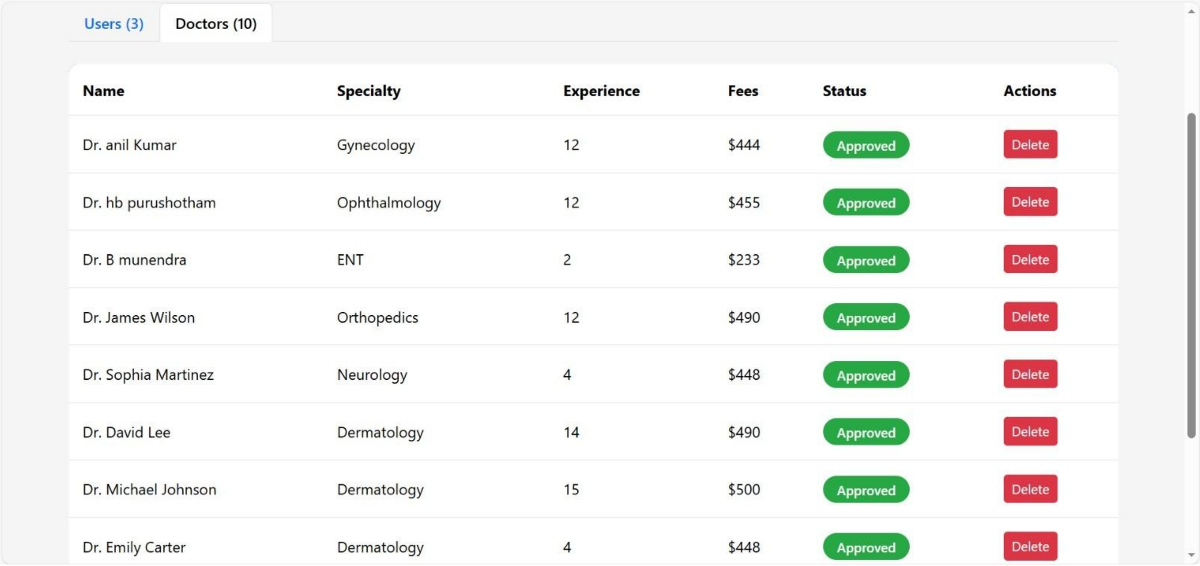
All Appointments view:-



All Appointments

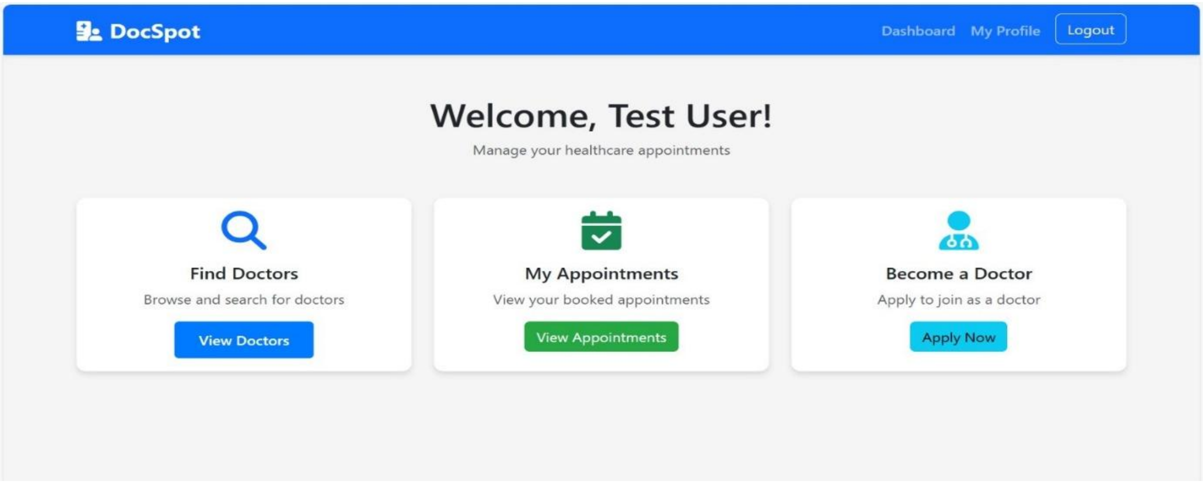
Patient	Doctor	Date	Time	Status	Created
Test User user@docspot.com	Dr. John Smith General Medicine	19/2/2026	23:23	Approved	18/2/2026
Test User user@docspot.com	Dr. John Smith General Medicine	18/2/2026	23:18	Approved	18/2/2026
Test User user@docspot.com	Dr. John Smith General Medicine	27/2/2026	23:06	Pending	18/2/2026
Test User user@docspot.com	Dr. John Smith General Medicine	28/2/2026	20:54	Pending	18/2/2026
Test User user@docspot.com	Dr. John Smith General Medicine	20/2/2026	20:13	Approved	18/2/2026

Doctor List:-



Name	Specialty	Experience	Fees	Status	Actions
Dr. anil Kumar	Gynecology	12	$444	Approved	Delete
Dr. hb purushotham	Ophthalmology	12	$455	Approved	Delete
Dr. B munendra	ENT	2	$233	Approved	Delete
Dr. James Wilson	Orthopedics	12	$490	Approved	Delete
Dr. Sophia Martinez	Neurology	4	$448	Approved	Delete
Dr. David Lee	Dermatology	14	$490	Approved	Delete
Dr. Michael Johnson	Dermatology	15	$500	Approved	Delete
Dr. Emily Carter	Dermatology	4	$448	Approved	Delete

Test User:-



Welcome, Test User!

Manage your healthcare appointments

Find Doctors

Browse and search for doctors

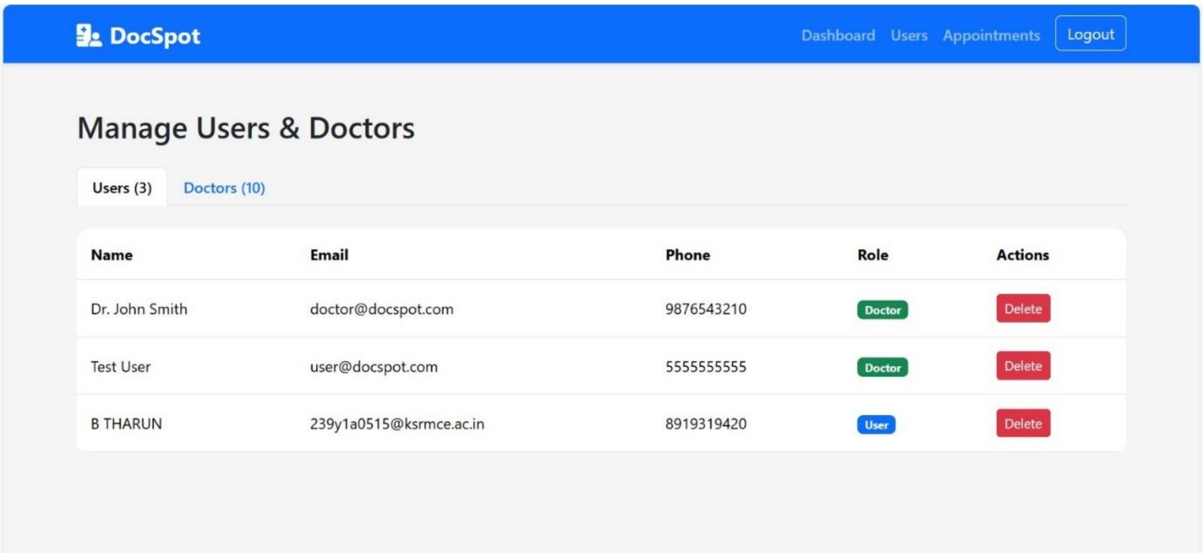
My Appointments

View your booked appointments

Become a Doctor

Apply to join as a doctor

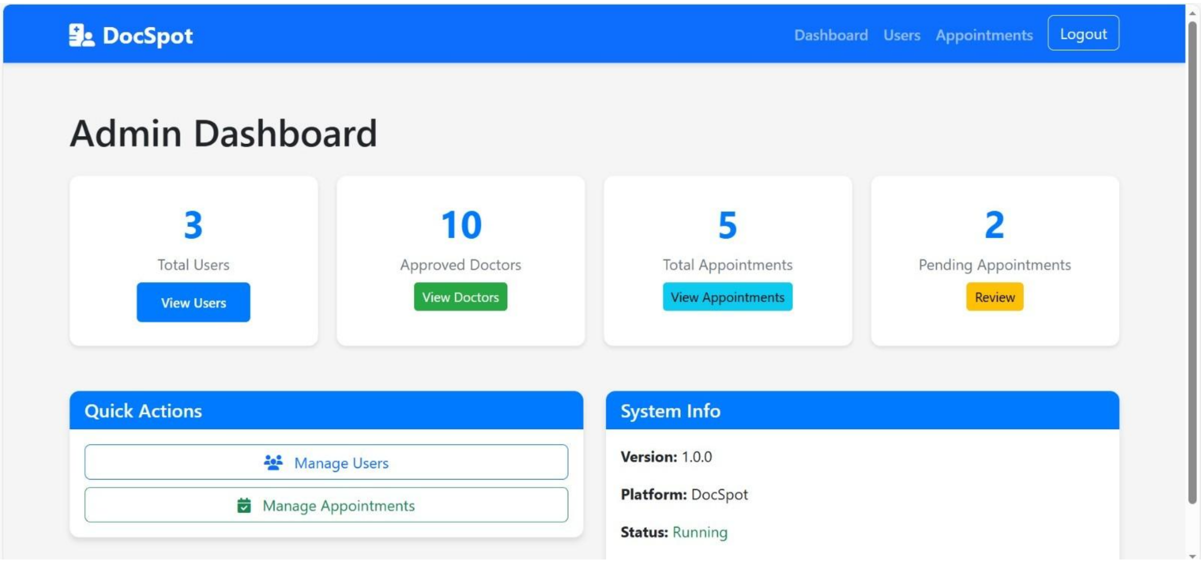
Manage Users &Doctors:-



Manage Users & Doctors

Name	Email	Phone	Role	Actions
Dr. John Smith	doctor@docspot.com	9876543210	Doctor	Delete
Test User	user@docspot.com	5555555555	Doctor	Delete
B THARUN	239y1a0515@ksrmce.ac.in	8919319420	User	Delete

Admin Dashboard:-



Admin Dashboard

3 Total Users

10 Approved Doctors

5 Total Appointments

2 Pending Appointments

Quick Actions

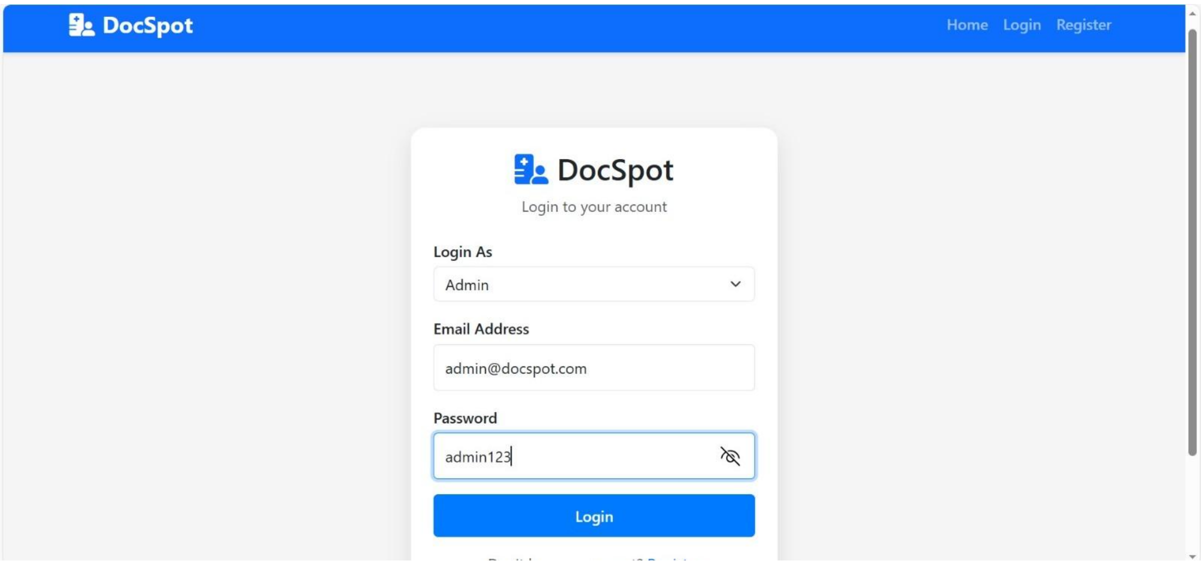
System Info

Version: 1.0.0

Platform: DocSpot

Status: Running

Admin Login:-



DocSpot

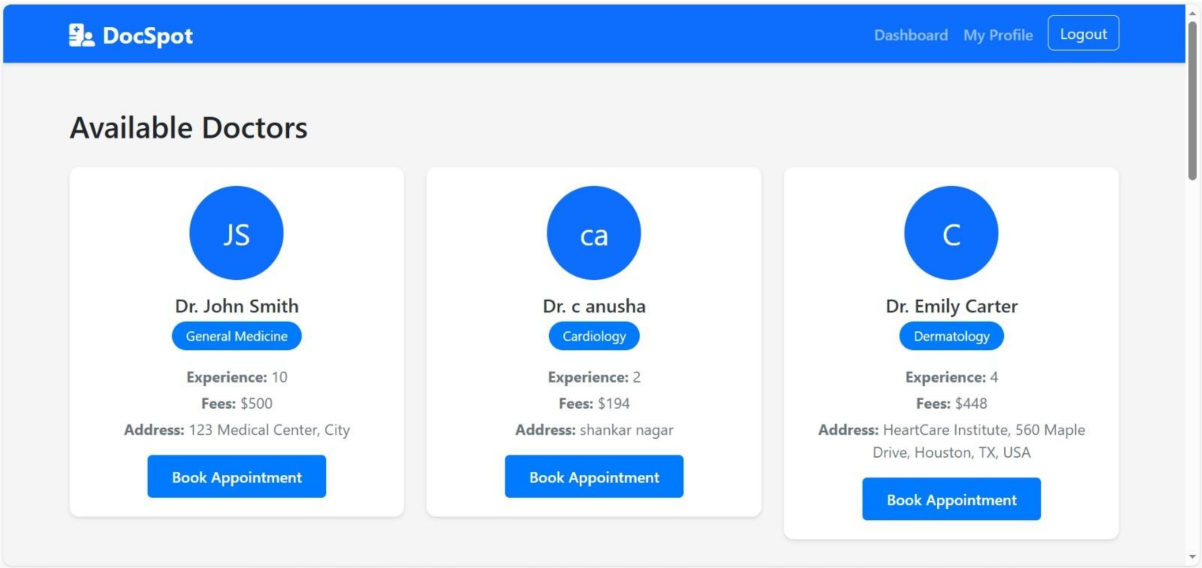
Login to your account

Login As: Admin

Email Address: admin@docspot.com

Password: admin123

Available Doctors:-



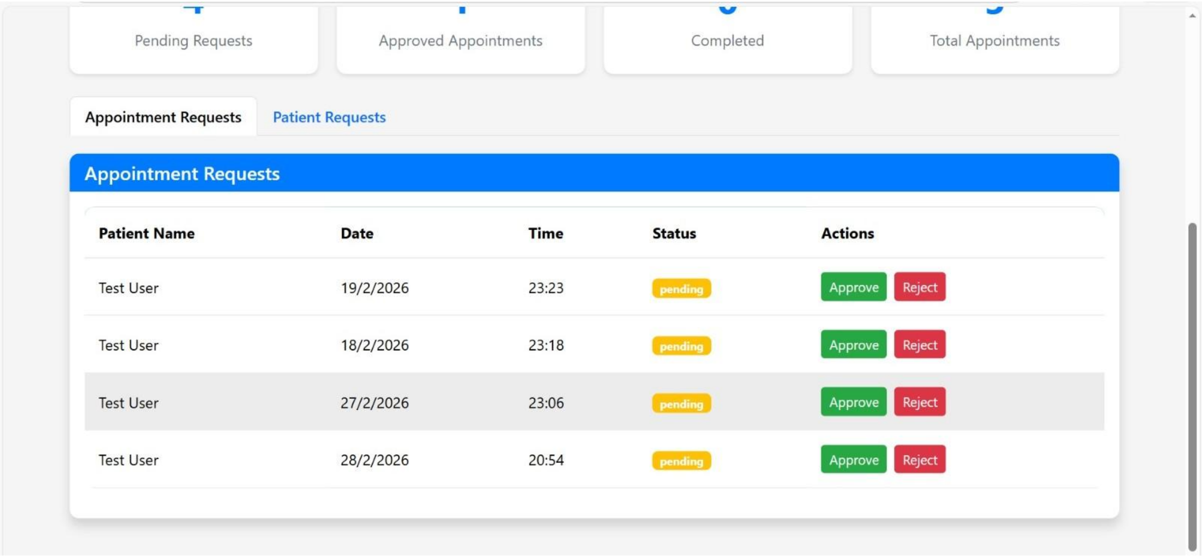
Available Doctors

JS
Dr. John Smith
General Medicine
Experience: 10
Fees: $500
Address: 123 Medical Center, City

ca
Dr. c anusha
Cardiology
Experience: 2
Fees: $194
Address: shankar nagar

C
Dr. Emily Carter
Dermatology
Experience: 4
Fees: $448
Address: HeartCare Institute, 560 Maple Drive, Houston, TX, USA

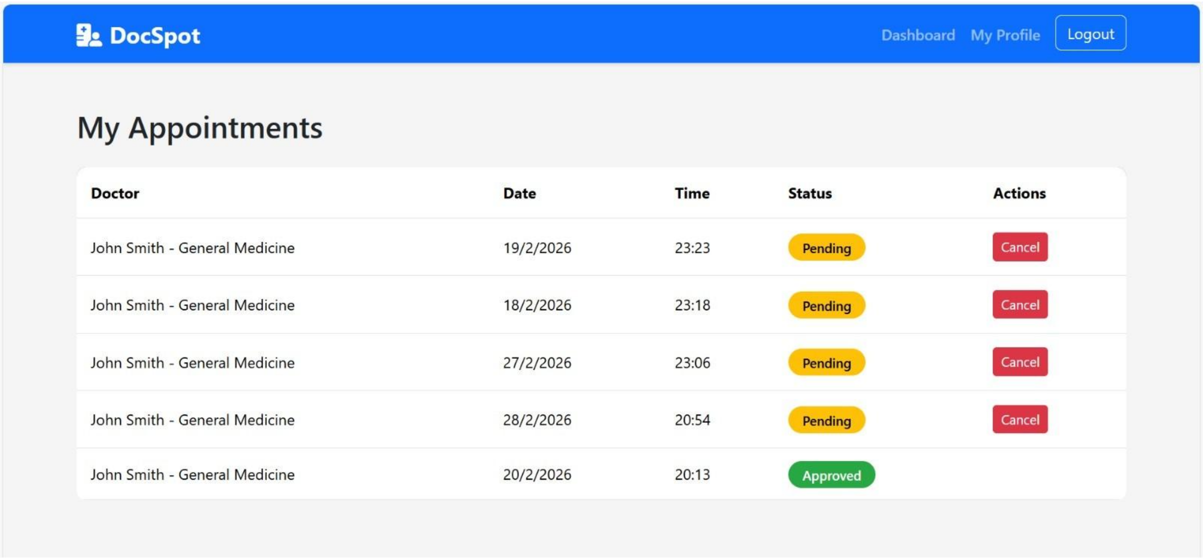
Appointment Requirements:-



Appointment Requests

Patient Name	Date	Time	Status	Actions
Test User	19/2/2026	23:23	pending	Approve Reject
Test User	18/2/2026	23:18	pending	Approve Reject
Test User	27/2/2026	23:06	pending	Approve Reject
Test User	28/2/2026	20:54	pending	Approve Reject

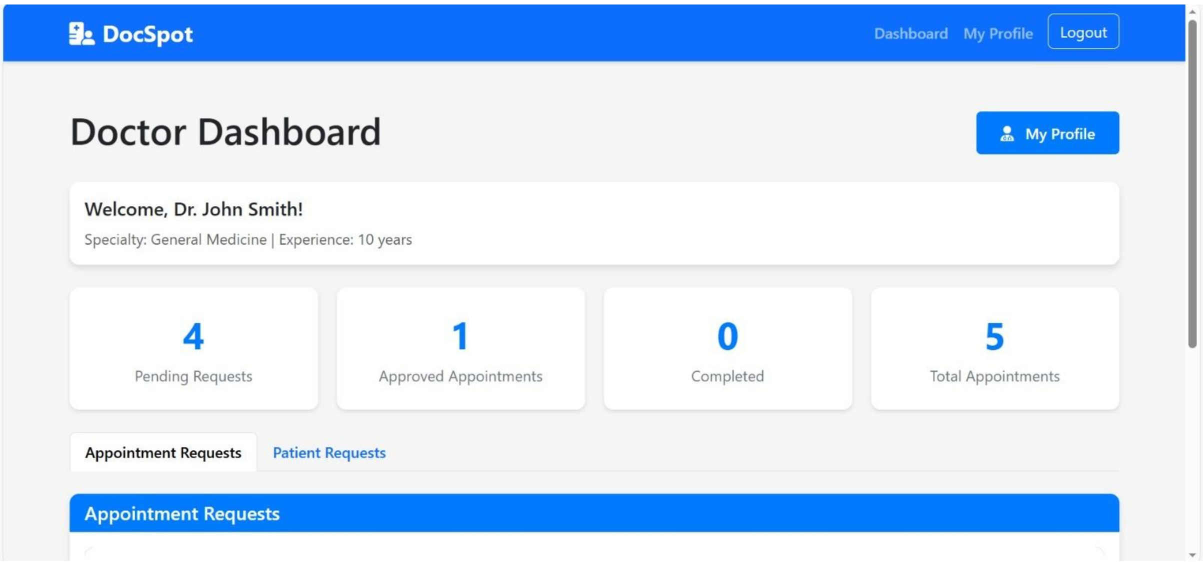
My Appointments:-



My Appointments

Doctor	Date	Time	Status	Actions
John Smith - General Medicine	19/2/2026	23:23	Pending	Cancel
John Smith - General Medicine	18/2/2026	23:18	Pending	Cancel
John Smith - General Medicine	27/2/2026	23:06	Pending	Cancel
John Smith - General Medicine	28/2/2026	20:54	Pending	Cancel
John Smith - General Medicine	20/2/2026	20:13	Approved	

Doctor Dashboard:-



Doctor Dashboard

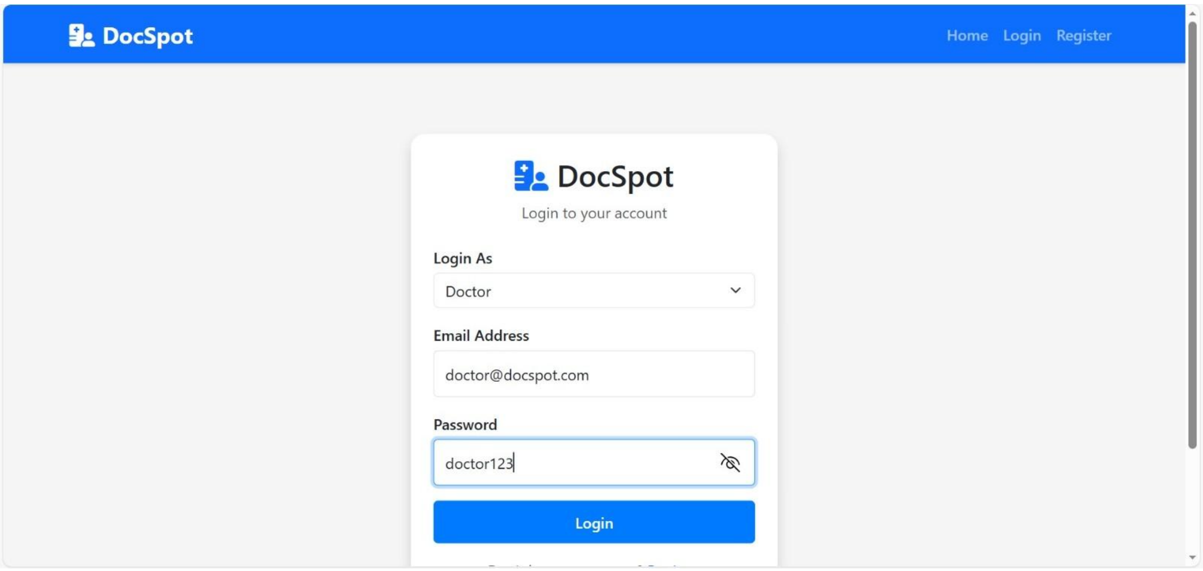
Welcome, Dr. John Smith!

Specialty: General Medicine | Experience: 10 years

4	1	0	5
Pending Requests	Approved Appointments	Completed	Total Appointments

Appointment Requests

Doctor Dashboard:-



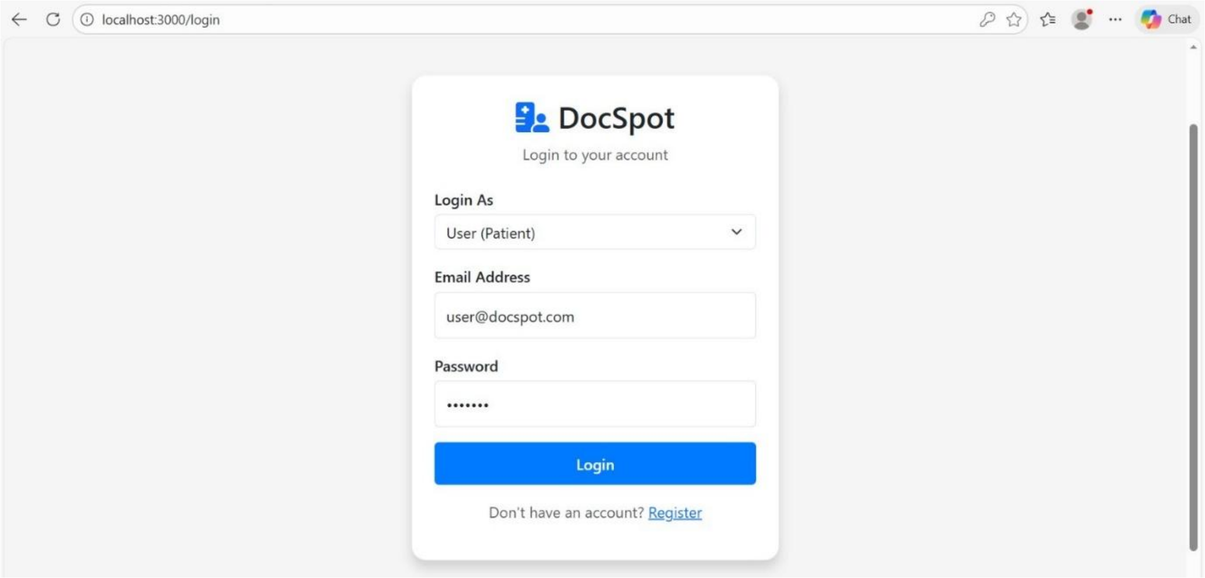
Login to your account

Login As: Doctor

Email Address: doctor@docspot.com

Password: doctor123

User Login :-



Login to your account

Login As: User (Patient)

Email Address: user@docspot.com

Password: •••••••

Don't have an account? Register