**Certificate**

This is to clarify that **veera venkata Pratap inolu** the Roll No “**22HU1A0412**” has carried out a research work entitled “ Docspot Seamless appointment Booking for Health” for a project work ,2025. The study was conducted under my supervision. I further certify that project work has been submitted to the smart Bridge

Veera Venkata Pratap

Department of ECE

Chebrolu engineering college

**Declaratipon**

I do here by declare that the project work entitled “**Docspot seamless Appointment Booking for Healt**h **”** presented for the fulfilment of +3 final year ECE of 6th semester, 2025 has been carried out by me and has not been previously submitted to any other university , college or organization for academic qualification on certificate or any degree.

I do here by warrant that the work, I have presented does not match to any existing copy right acts.

Veera Venkata Pratap inolu

+3 final year , ECE

ROLL NO:- 22HU1a0412

Chebrolu Engineering college

**Acknowledgement**

I would liketo gratefully acknowledge the following peopke and organization or their assistance and advice that have made the project successful. I am grateful to honrable.

**SmartBridge** lect for giving me permission to pursue this study. I am especially thankful to my friend rajesh for this guidance and cooperation during my project work.. they have encouraged me during my project work . a special word . of thanks to him .

he first image is of frontend part which is showing all the files and folders that have been used in UI development

The second image is of Backend part which is showing all the files and folders that have been used in backend development

**INTRODUCTION**

**Application Flow:** The project has 2 type of user – Customer and Doctor and other will be Admin which takes care to all the user. The roles and responsibilities of these two types of users can be inferred from the API endpoints defined in the code. Here is a summary:

Customer/Ordinary:

Create an account and log in to the system using their email and password.

They will be shown automatically all the doctors in their dashboard.

After clicking on the Book Now, a form will generate in which date of appointment and documents need to send.

They can sees the status of their appointment and can get a notification if the appointment is schedule or not.

The user can also cancel it`s booking in booking history page and can change the status of booking.

Admin:

Manage and monitor the overall operation of the appointment and the type of users and doctors to the application.

He monitors the applicant of all doctors and approve them and then doctors are registered in the app.

Implement and enforce platform policies, terms of service, and privacy regulations.

Doctor:

Gets the approval from the admin for his doctor account.

Manages all the appointments that are getting from the users

\

**STRUCTURE OF PROJECT**

1. **Front end**
2. docspot/
3. ├── index.html
4. ├── style.css
5. └── images/
6. ├── logo.png
7. ├── doctor1.jpg
8. └── doctor2.jpg
9. You’ll need to place your actual images inside the images/ folder or use placeholder image URLs.
10. 📄 index.html
11. html
12. Copy
13. Edit
14. <!DOCTYPE html>
15. <html lang="en">
16. <head>
17. <meta charset="UTF-8" />
18. <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
19. <title>DocSpot - Seamless Appointment Booking</title>
20. <link rel="stylesheet" href="style.css" />
21. </head>
22. <body>
23. <header>
24. <img src="images/logo.png" alt="DocSpot Logo" class="logo" />
25. <nav>
26. <ul>
27. <li><a href="#home">Home</a></li>
28. <li><a href="#doctors">Doctors</a></li>
29. <li><a href="#book">Book Appointment</a></li>
30. <li><a href="#contact">Contact</a></li>
31. </ul>
32. </nav>
33. </header>
34. <section class="hero" id="home">
35. <h1>Welcome to DocSpot</h1>
36. <p>Book appointments with trusted doctors in just a few clicks</p>
37. <a href="#book" class="btn">Book Now</a>
38. </section>
39. <section class="doctors" id="doctors">
40. <h2>Our Top Doctors</h2>
41. <div class="doctor-list">
42. <div class="doctor-card">
43. <img src="images/doctor1.jpg" alt="Dr. Smith" />
44. <h3>Dr. Anna Smith</h3>
45. <p>Cardiologist</p>
46. </div>
47. <div class="doctor-card">
48. <img src="images/doctor2.jpg" alt="Dr. John" />
49. <h3>Dr. John Lee</h3>
50. <p>Dermatologist</p>
51. </div>
52. </div>
53. </section>
54. <section class="appointment" id="book">
55. <h2>Book Your Appointment</h2>
56. <form>
57. <input type="text" placeholder="Your Name" required />
58. <input type="email" placeholder="Email Address" required />
59. <select required>
60. <option value="">Select Doctor</option>
61. <option>Dr. Anna Smith</option>
62. <option>Dr. John Lee</option>
63. </select>
64. <input type="date" required />
65. <button type="submit" class="btn">Confirm Booking</button>
66. </form>
67. </section>
68. <footer id="contact">
69. <p>Contact us: support@docspot.com | +123-456-7890</p>
70. <p>&copy; 2025 DocSpot. All rights reserved.</p>
71. </footer>
72. </body>
73. </html>
74. 🎨 style.css
75. css
76. Copy
77. Edit
78. /\* Basic Reset \*/
79. \* {
80. margin: 0;
81. padding: 0;
82. box-sizing: border-box;
83. font-family: 'Arial', sans-serif;
84. }
85. body {
86. background-color: #f8f9fa;
87. color: #333;
88. }
89. /\* Header \*/
90. header {
91. background-color: #0d6efd;
92. display: flex;
93. align-items: center;
94. justify-content: space-between;
95. padding: 1rem 2rem;
96. color: white;
97. }
98. .logo {
99. height: 40px;
100. }
101. nav ul {
102. list-style: none;
103. display: flex;
104. gap: 1.5rem;
105. }
106. nav a {
107. color: white;
108. text-decoration: none;
109. font-weight: bold;
110. }
111. /\* Hero Section \*/
112. .hero {
113. text-align: center;
114. padding: 4rem 2rem;
115. background: linear-gradient(to right, #0d6efd, #66b2ff);
116. color: white;
117. }
118. .hero .btn {
119. margin-top: 1rem;
120. padding: 0.8rem 2rem;
121. background: white;
122. color: #0d6efd;
123. border: none;
124. font-size: 1rem;
125. cursor: pointer;
126. border-radius: 4px;
127. transition: 0.3s;
128. }
129. .hero .btn:hover {
130. background-color: #dbe9ff;
131. }
132. /\* Section Styles \*/
133. section {
134. padding: 3rem 2rem;
135. }
136. h2 {
137. text-align: center;
138. margin-bottom: 2rem;
139. }
140. /\* Doctors Section \*/
141. .doctors {
142. background-color: #fff;
143. }
144. .doctor-list {
145. display: flex;
146. justify-content: center;
147. gap: 2rem;
148. flex-wrap: wrap;
149. }
150. .doctor-card {
151. background: #f0f0f0;
152. padding: 1rem;
153. border-radius: 10px;
154. width: 200px;
155. text-align: center;
156. box-shadow: 0 2px 5px rgba(0,0,0,0.1);
157. }
158. .doctor-card img {
159. width: 100%;
160. border-radius: 50%;
161. margin-bottom: 1rem;
162. }
163. /\* Appointment Section \*/
164. .appointment {
165. background-color: #e9f2ff;
166. text-align: center;
167. }
168. .appointment form {
169. display: flex;
170. flex-direction: column;
171. gap: 1rem;
172. max-width: 400px;
173. margin: auto;
174. }
175. form input, form select, form button {
176. padding: 0.75rem;
177. border: 1px solid #ccc;
178. border-radius: 5px;
179. }
180. form button.btn {
181. background-color: #0d6efd;
182. color: white;
183. border: none;
184. cursor: pointer;
185. transition: 0.3s;
186. }
187. form button.btn:hover {
188. background-color: #0056d2;
189. }
190. /\* Footer \*/
191. footer {
192. background-color: #0d6efd;
193. color: white;
194. text-align: center;
195. padding: 1rem;
196. }
197. 🖼️ Images
198. Place your images inside the /images/ folder:
199. logo.png – your logo (e.g., 150x40)
200. doctor1.jpg – doctor image (200x200)
201. doctor2.jpg – another doctor
202. Alternative: Use placeholder URLs like:
203. html
204. Copy
205. Edit
206. <img src="https://via.placeholder.com/200" alt="Doctor Image" />

he first image is of frontend part which is showing all the files and folders that have been used in UI development

The second image is of Backend part which is showing all the files and folders that have been used in backend developmentApplication Flow: The project has 2 type of user – Customer and Doctor and other will be Admin which takes care to all the user. The roles and responsibilities of these two types of users can be inferred from the API endpoints defined in the code. Here is a summary: Customer/Ordinary: Create an account and log in to the system using their email and password.

They will be shown automatically all the doctors in their dashboard.

After clicking on the Book Now, a form will generate in which date of appointment and documents need to send.

They can sees the status of their appointment and can get a notification if the appointment is schedule or not.

The user can also cancel it`s booking in booking history page and can change the status of booking.

Admin:

Manage and monitor the overall operation of the appointment and the type of users and doctors to the application. He monitors the applicant of all doctors and approve them and then doctors are registered in the app.Implement and enforce platform policies, terms of service, and privacy regulation

Doctor:

Gets the approval from the admin for his doctor account.

Manages all the appointments that are getting from the users

**Back end**

1. **Folder Structure**
2. docspot/
3. ├── backend/
4. │ ├── server.js
5. │ ├── appointments.json
6. ├── frontend/
7. │ ├── index.html
8. │ ├── style.css

9│ └── images/

**2.Initialize Node.js**

npm init -y

npm install express body-parser cors fs

1. **server.js**

const express = require('express');

const bodyParser = require('body-parser');

const fs = require('fs');

const cors = require('cors');

const app = express();

const PORT = 3000;

// Middleware

app.use(cors());

app.use(bodyParser.json());

app.use(bodyParser.urlencoded({ extended: true }));

// Load or initialize appointments file

const DATA\_FILE = './appointments.json';

if (!fs.existsSync(DATA\_FILE)) {

fs.writeFileSync(DATA\_FILE, JSON.stringify([]));

}

// Route: Home (API status)

app.get('/', (req, res) => {

res.send('DocSpot API is running');

});

// Route: Handle appointment booking

app.post('/api/book', (req, res) => {

const { name, email, doctor, date } = req.body;

if (!name || !email || !doctor || !date) {

return res.status(400).json({ error: 'All fields are required' });

}

const appointment = { name, email, doctor, date, bookedAt: new Date() };

const data = JSON.parse(fs.readFileSync(DATA\_FILE));

data.push(appointment);

fs.writeFileSync(DATA\_FILE, JSON.stringify(data, null, 2));

res.status(201).json({ message: 'Appointment booked successfully!', appointment });

});

// Start server

app.listen(PORT, () => {

console.log(`Server running on http://localhost:${PORT}`);

});

**4.Connect HTML Form to Backend**

<form id="bookingForm">

<input type="text" id="name" placeholder="Your Name" required />

<input type="email" id="email" placeholder="Email Address" required />

<select id="doctor" required>

<option value="">Select Doctor</option>

<option>Dr. Anna Smith</option>

<option>Dr. John Lee</option>

</select>

<input type="date" id="date" required />

<button type="submit" class="btn">Confirm Booking</button>

</form>

<p id="responseMessage"></p>

<script>

const form = document.getElementById('bookingForm');

const responseMsg = document.getElementById('responseMessage');

form.addEventListener('submit', async (e) => {

e.preventDefault();

const appointment = {

name: document.getElementById('name').value,

email: document.getElementById('email').value,

doctor: document.getElementById('doctor').value,

date: document.getElementById('date').value

};

try {

const res = await fetch('http://localhost:3000/api/book', {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify(appointment)

});

const data = await res.json();

responseMsg.textContent = data.message || data.error;

responseMsg.style.color = res.ok ? 'green' : 'red';

} catch (error) {

responseMsg.textContent = 'Server error. Try again later.';

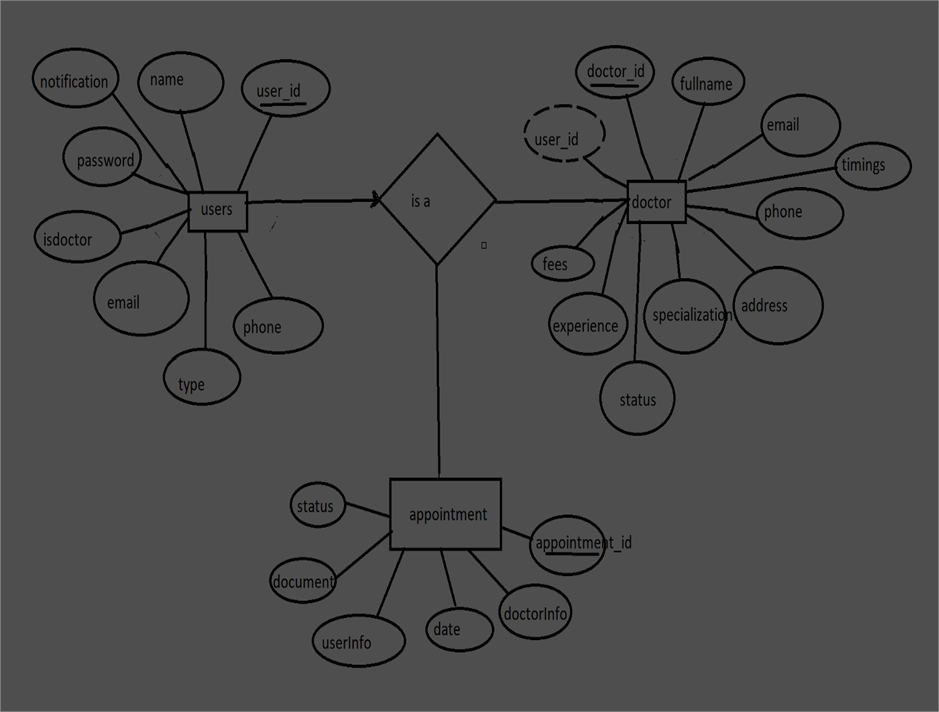
responseMsg.style.color = 'red';

}

});

</script>

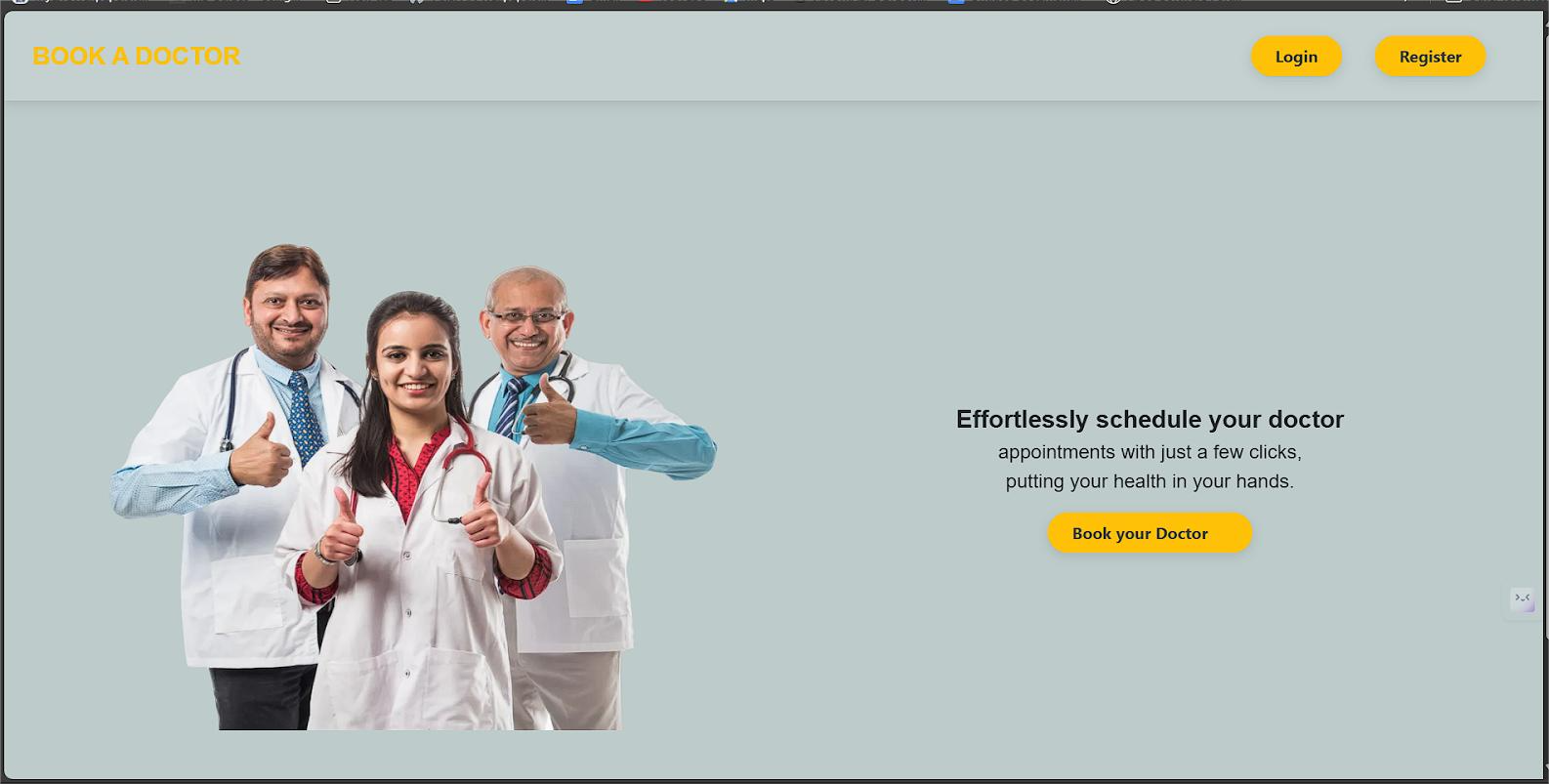
**ER Diagram**



An **Entity-Relationship (ER) diagram** is a foundational tool in database design that visually represents the relationships between various data elements within a system. It is particularly useful in the early stages of software development to map out the data model before implementing the actual database. In the case of the **DocSpot** appointment booking platform, an ER diagram helps define the structure of essential components such as **Users**, **Doctors**, **Appointments**, and potentially **Admins** or **Feedback**. Each **entity** (like User or Doctor) is represented as a rectangle, and its **attributes** (like name, email, or specialization) are shown as connected ovals. **Relationships** (like "books" or "manages") between entities are illustrated with diamonds, and cardinality (e.g., one-to-many, many-to-one) indicates how many records can be linked between entities.

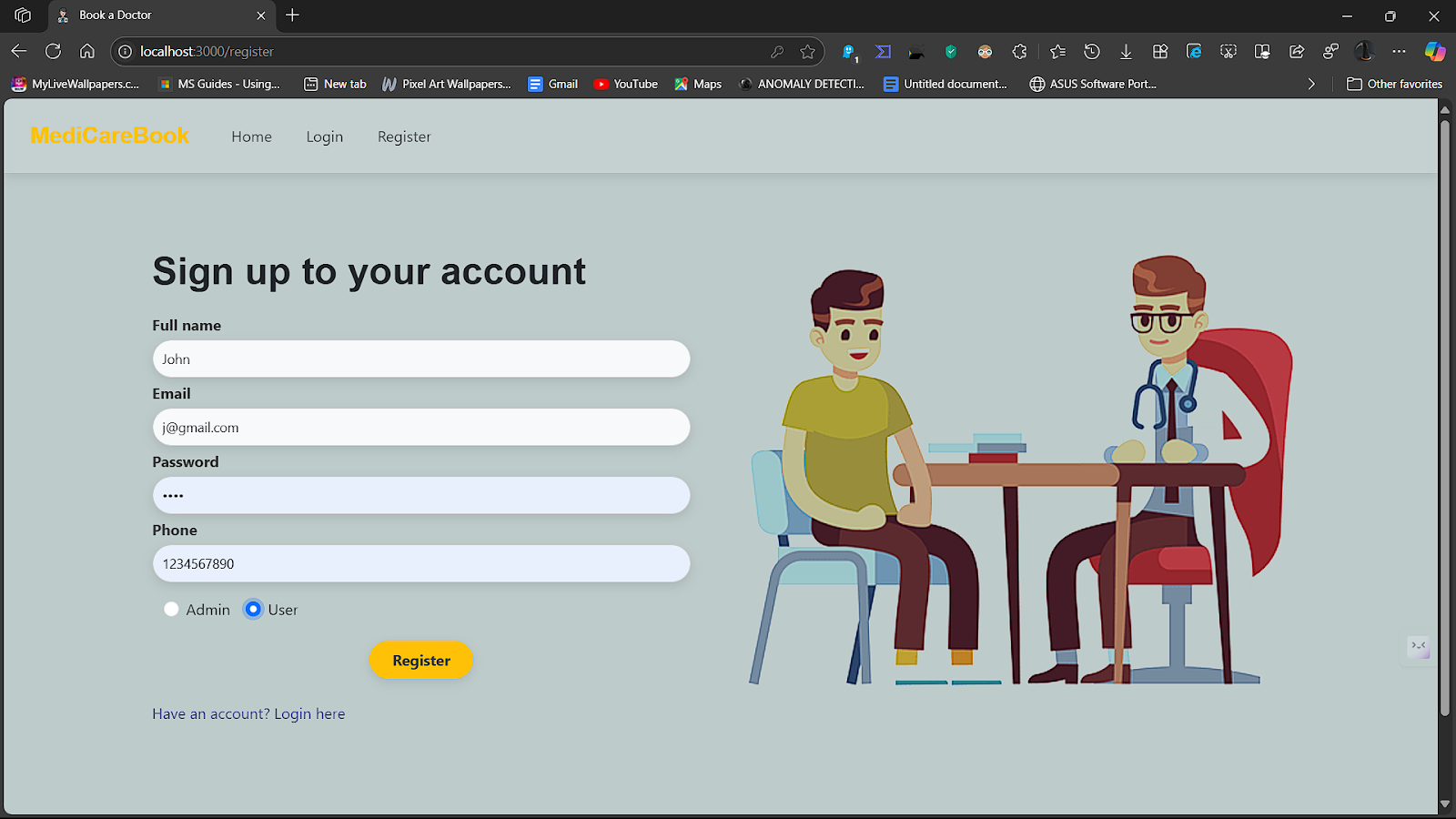
For example, in DocSpot, a **User** can book multiple **Appointments**, and each **Appointment** is linked to one **Doctor**. This many-to-one relationship helps define how the tables in the database will be structured, with foreign keys like user\_id and doctor\_id stored in the Appointment table. An ER diagram can also help identify **data constraints**, **optional relationships**, and **data normalization opportunities** to avoid redundancy. Furthermore, it serves as a communication bridge between technical and non-technical stakeholders by providing a clear and visual overview of how data flows and connects within the system. Overall, the ER diagram is critical for ensuring the DocSpot system is efficient, scalable, and logically organized before moving into backend development and database implementation.

**Landing page**



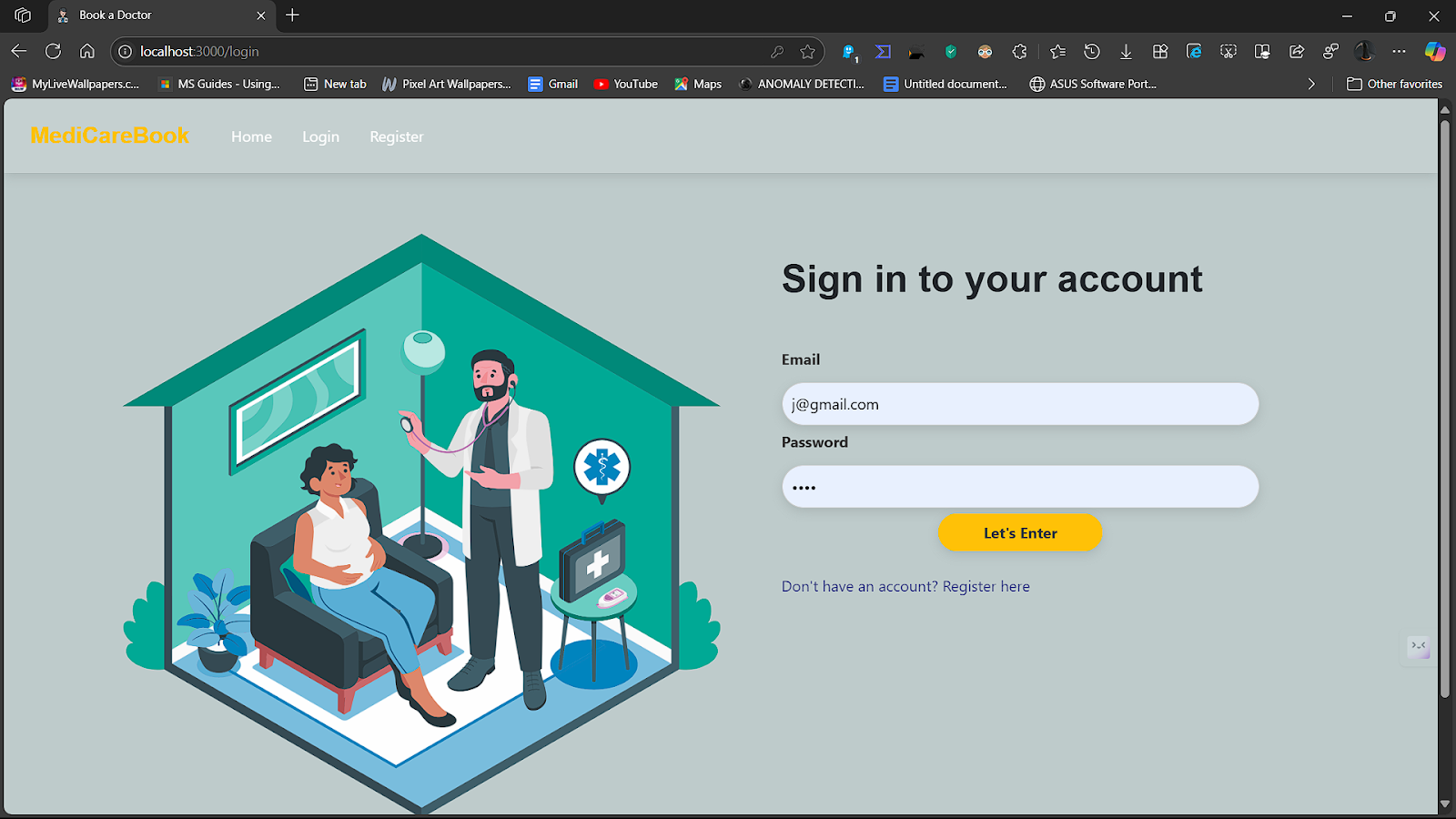
A landing page is a standalone web page created specifically to capture the attention of visitors and guide them toward a specific action, such as booking an appointment or signing up. For the **DocSpot** healthcare appointment platform, the landing page acts as the first impression for users, introducing them to the service and encouraging them to schedule a consultation with a doctor. It typically includes a clean layout with key elements like a prominent headline, a brief description of the platform's benefits, featured doctors, a booking form, and a strong call-to-action (CTA) button such as "Book Now." The purpose of this page is to simplify the user journey, build trust through visuals and testimonials, and convert visitors into patients by making the booking process as seamless as possible.

**Registered page**

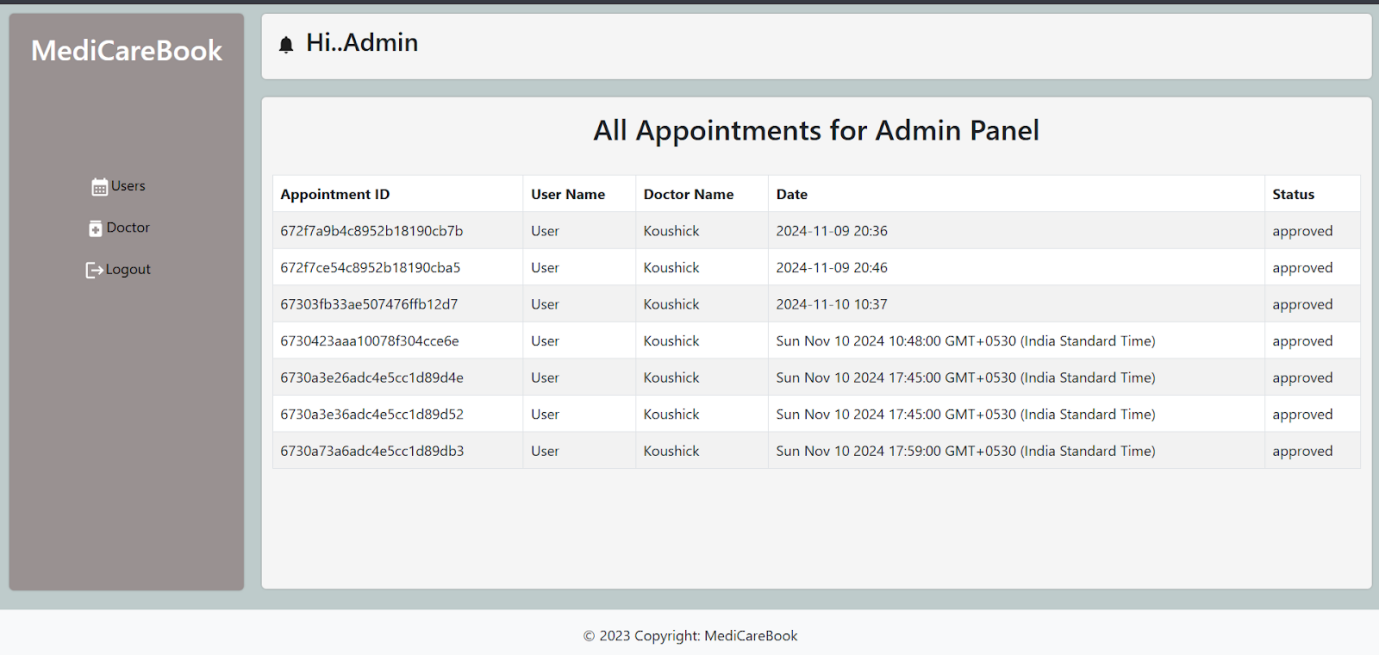


A landing page is a standalone web page created specifically to capture the attention of visitors and guide them toward a specific action, such as booking an appointment or signing up. For the **DocSpot** healthcare appointment platform, the landing page acts as the first impression for users, introducing them to the service and encouraging them to schedule a consultation with a doctor. It typically includes a clean layout with key elements like a prominent headline, a brief description of the platform's benefits, featured doctors, a booking form, and a strong call-to-action (CTA) button such as "Book Now." The purpose of this page is to simplify the user journey, build trust through visuals and testimonials, and convert visitors into patients by making the booking process as seamless as possible.

**Login page**

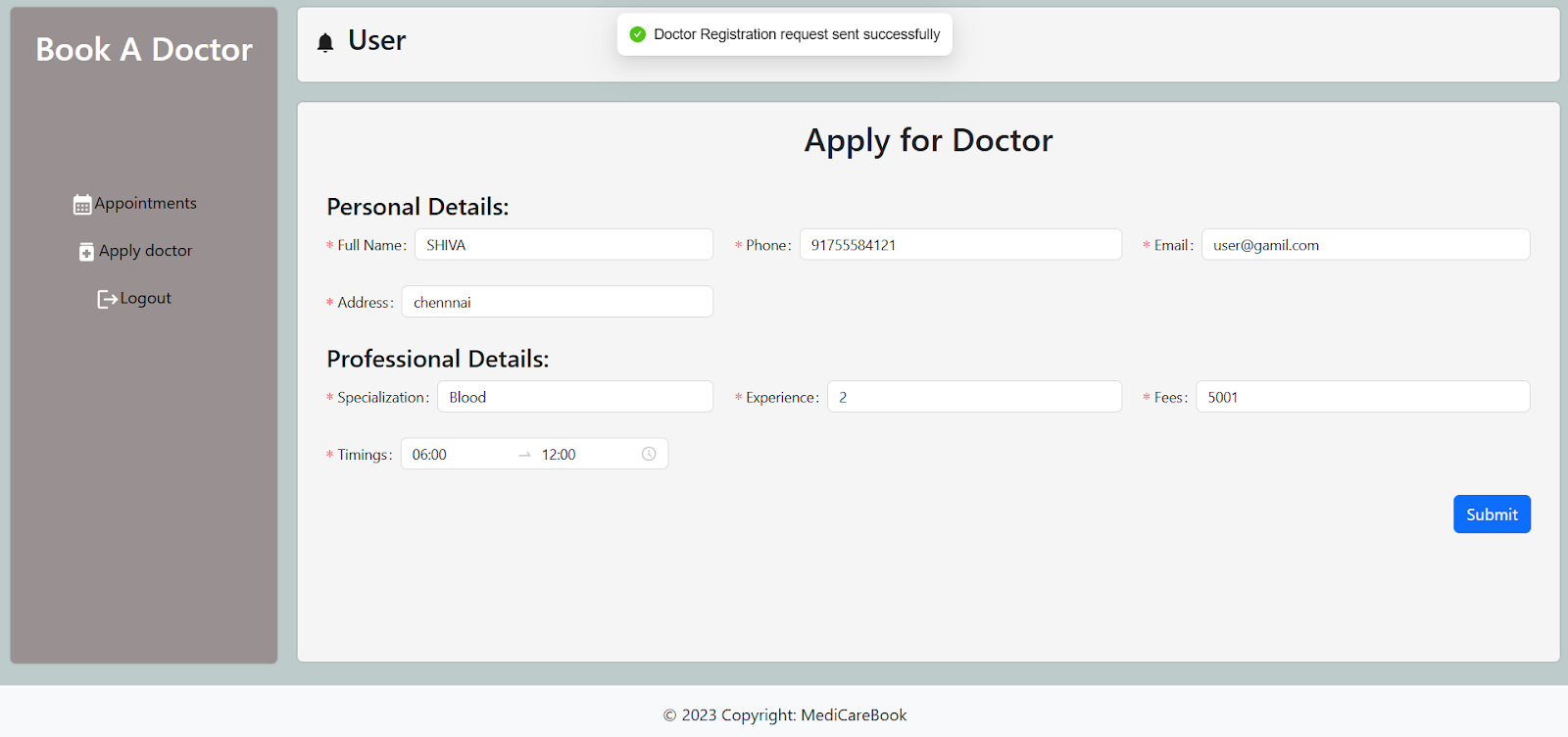


A **login page** is a secure entry point where registered users enter their credentials—typically an email or username and a password—to access their personalized dashboard or services. In the **DocSpot** healthcare platform, the login page enables both patients and doctors to sign in to their accounts, view or manage appointments, and access medical information relevant to them. This page plays a vital role in maintaining data privacy and ensuring that only authorized users can access sensitive health-related content. A well-designed login page is clean, intuitive, and includes features like "Forgot Password" and password visibility toggles to improve usability. For DocSpot, it ensures that users can quickly and securely return to the platform to continue booking, managing, or attending appointments.



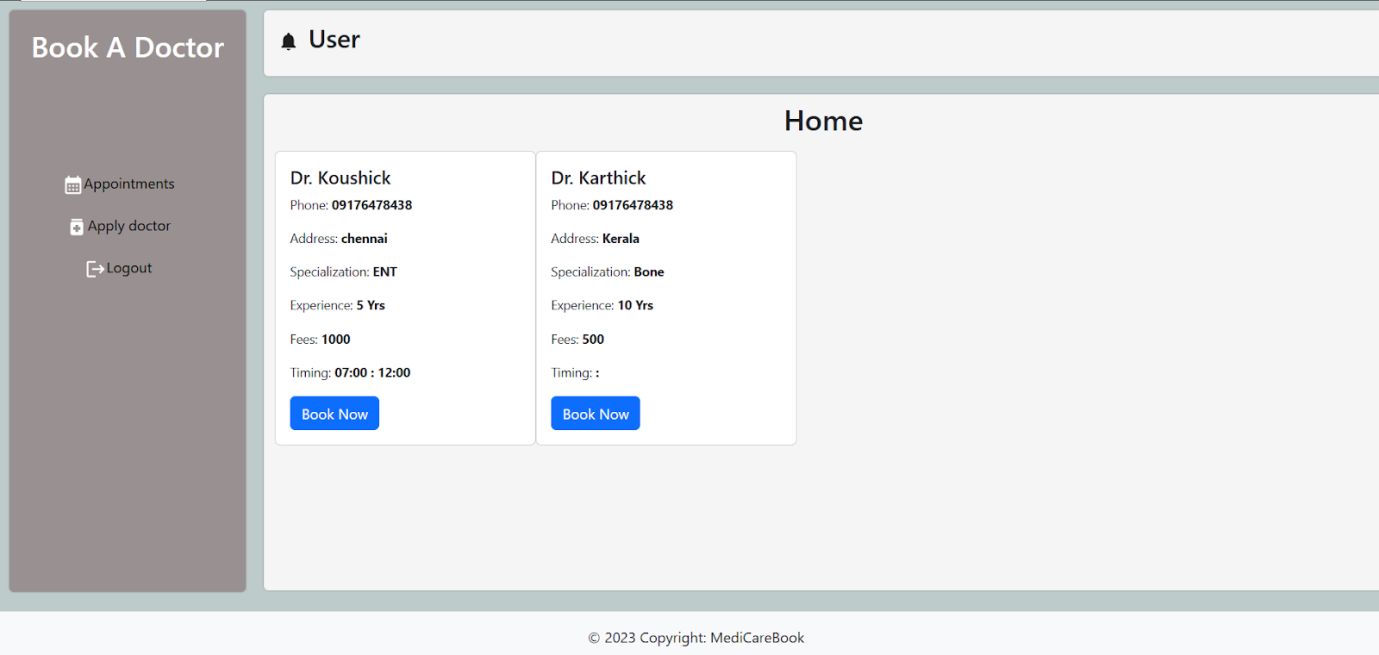
**Admin DashBoard**

An **admin dashboard** is a secure, centralized interface that allows administrators to manage and oversee the entire system. In the **DocSpot** healthcare appointment platform, the admin dashboard provides powerful tools for managing users (patients and doctors), monitoring appointments, updating doctor availability, handling reports, and maintaining platform security. It typically includes charts, tables, and key metrics—such as the number of bookings, active users, or appointment statuses—to give admins a real-time overview of system performance. The dashboard may also offer controls to approve or deactivate doctors, send notifications, and manage site content. A well-designed admin dashboard ensures that platform operations run smoothly, data remains organized, and users have a seamless experience across the app.



**Doctor Dash Board**

A **doctor dashboard** is a personalized control panel designed specifically for doctors to manage their appointments, interact with patients, and monitor their schedule. In the **DocSpot** healthcare platform, the doctor dashboard provides key features such as viewing upcoming and past appointments, updating availability, managing patient details, and responding to appointment requests. It may also include tools for writing notes, sending follow-ups, or reviewing patient histories if integrated. The interface is typically clean and user-friendly, giving doctors a clear view of their daily workload and patient interactions. By using the dashboard, doctors can streamline their practice, reduce administrative tasks, and focus more on patient care within the DocSpot system.



**User Dash Board**

A **user dashboard** is a personalized area within the **DocSpot** platform where patients (users) can manage their healthcare interactions efficiently. After logging in, users are directed to this dashboard to view upcoming and past appointments, select or change doctors, update personal information, and cancel or reschedule bookings if needed. The dashboard may also include appointment reminders, medical history (if integrated), and messages or feedback from doctors. Designed for ease of use, the user dashboard provides a clear, organized view of all healthcare activities, ensuring that patients can conveniently manage their health appointments and stay connected with their doctors—all in one place.

### **Project FlowDemo Video and reference code link**

reference video link:

<https://drive.google.com/drive/folders/1pteT8STdObONWwELNDHRK9biItLuiJ-1?usp=sharing>

 reference code link:

[Doctor Appointment Booking Using MERN Source Code](https://drive.google.com/drive/folders/1k8dAHWUpJ1poLIjZK_V4U3Pl5u8DkWo1?usp=sharing)