NEW HORIZON
COLLEGE OF ENGINEERING
Autonomous College Permanently Affiliated to VTU, Approved by AICTE & UGC
Accredited by NAAC with 'A' Grade, Accredited by NBA

A MINI PROJECT REPORT

on

HOSPITAL MANAGEMENT SYSTEM

Submitted by,

RAMNATH KINI

1NH24CS164

Under the guidance of-

Ms. DIVYANSHI CHHABRA

ASSOCIATE PROFESSOR

In partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

Academic Year:2024-25 (ODD SEM)

NEW HORIZON
COLLEGE OF ENGINEERING
Autonomous College Permanently Affiliated to VTU, Approved by AICTE & UGC
Accredited by NAAC with 'A' Grade, Accredited by NBA

This is to certify that the mini project work titled "HOSPITAL MANAGEMENT SYSTEM" is a Bonafide work carried out by RAMNATH KINI (1NH24CS164) in partial fulfilment of the degree of Bachelor of Engineering in Computer Science and Engineering of the New Horizon College of Engineering during the year 2025-2026.

Signature of Guide

Signature of HOD

SEMESTER END EXAMINATION

Name of the Examiner *Signature with date*

1. _____ _____

2. _____ _____

ABSTRACT

The Responsive Hospital Appointment Form is a web-based system developed using HTML, CSS, and JavaScript. Traditional hospital appointment forms often overwhelm users due to their length and poor structure. This project introduces a modern, interactive, and user-friendly appointment form that divides the appointment process into three clear stages: Patient registration, Book Appointment and Doctors List.

Form validation ensures correctness of user input, while JavaScript manages scroll transitions and error handling. Once each detail in respective fields is successfully submitted, a success text sequence gets displayed, confirming completion.

This project aims to improve form completion rates, enhance user experience, and demonstrate modern web design using responsive layouts and interactive front-end technologies.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be impossible without the mention of the people who made it possible, who's constant guidance and encouragement crowned our efforts with success.

I have great pleasure in expressing gratitude to **Dr. Mohan Manghnani**, Chairman, New Horizon Educational Institutions, for providing necessary infrastructure and creating good environment.

I take this opportunity to express my profound gratitude to **Dr. Manjunatha,** Principal, New Horizon College of Engineering, for the constant support and encouragement.

I would like to thank **Dr. Anandhi R J**, Professor and Dean-Academics, NHCE, for her valuable guidance.

I would also like to thank **Dr. B. Rajalakshmi,** Professor and HOD, Department of Computer Science and Engineering, for the constant support.

I also express my gratitude to **Ms. Divyanshi Chhabra**, Associate Professor, Department of Computer Science and Engineering, my mini project reviewer, for constantly monitoring the development of the project and setting up precise deadlines. Her valuable suggestions were the motivating factors in completing the work.

RAMNATH KINI

(1NH24CS164)

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

As healthcare services become more digitized, hospitals are increasingly adopting simple, user-friendly application forms to streamline patient intake and administrative tasks. Traditional paper-based forms are often lengthy, repetitive, and confusing, causing delays and frustration for patients seeking timely care. A well-designed digital hospital application form minimizes clutter, guides users step-by-step, and ensures essential information is captured accurately. This not only reduces administrative workload but also enhances the patient's overall experience, making the registration process smoother and more efficient.

This project introduces a Responsive Hospital Application Form, designed to simplify the application process through step-wise data collection, professional design and interactive feedback.

The system uses HTML for structural layout, CSS for styling and responsiveness, and JavaScript for dynamic behaviour such as progress tracking, field validation, and animated success acknowledgement.

1.1 PROBLEM DEFINITION

Traditional hospital application forms are often long, confusing, and slow to complete, creating frustration for patients and staff. A simplified, modern multi-step form is needed to make the process clearer and more efficient.

Common issues with current hospital forms include:

1. **Big amount of information packed into one long page.**

2. **Poor user experience, leading to incomplete submissions.**

3. **Little to no input validation, causing errors in patient data.**

4. **Non-responsive layouts that are hard to use on mobile devices.**

5. **No progress indicators, leaving users unsure of how much is left.**

6. **Lack of helpful feedback, making the interface feel outdated.**

Therefore, a clean, responsive, and easy-to-navigate hospital application system is required.

1.2 OBJECTIVES

The major objectives of the project are**:**

1. To design a simple and user-friendly patient application interface.

2. To divide the registration process into clear, manageable steps.

3. To ensure correct user input through validation for every step.

4. To create a responsive layout that works smoothly across all devices.

5. To display a success text confirming submission.

6. To improve the overall user experience compared to traditional forms.

1.3 METHODOLOGIES TO BE FOLLOWED

The development of the job application portal follows the following methodology:

1. Requirement Analysis

- Identifying user needs.
- Understanding the limitations of traditional forms.
- Collecting necessary functional and non-functional requirements.

2. System Design

- Designing UI layout for three-step form.
- Planning field arrangement and validation points.
- Creating flow diagrams for user navigation.

3. Front-End Development

- Implementing structure using HTML.
- Designing a responsive interface using CSS.
- Adding dynamic behaviour using JavaScript.

4. Testing

- Testing validation rules.
- Checking navigation between steps.

- Verifying responsiveness across screen sizes.
- Ensuring file upload works correctly.

5. Deployment

- Hosting on a local or cloud server.
- Optimizing for performance.

CHAPTER 2

FUNDAMENTALS OF THE LANGUAGES USED

2.1 HTML (Hyper Text Markup Language)

HTML is the standard markup language for creating web pages. It structures the content of a webpage using elements enclosed in tags. HTML (Hyper Text Markup Language) was created by Tim Berners-Lee in 1989 as part of the World Wide Web project at CERN, aimed at enabling scientists to share documents through hyperlinks. The first version of HTML was introduced in 1991, consisting of 18 basic tags like <h1> and <p>. In 1995, HTML 2.0 became the first standardized version, introducing new features like forms and tables. Subsequent updates, such as HTML 3.2 in 1997, added support for CSS, while HTML 4.0 in 1999 integrated multimedia elements and scripting capabilities. During the late 1990s and early 2000s, XHTML emerged as a stricter version of HTML but faced adoption challenges. HTML5, introduced in 2014, revolutionized the web by adding semantic elements, multimedia capabilities, and cross-platform support, cementing its role as the cornerstone of modern web development.

Key Features:

- **Structure:** Defines elements like headings, paragraphs, lists, and images.
- **Links:** Connects to other documents using hyperlinks.
- **Forms:** Captures user input.

Common HTML Tags and Their Descriptions:

Tag	Description

<html>	Root element of an HTML document.
<head>	Contains metadata, title, and linked resources.
<title>	Defines the title displayed in the browser tab.
<body>	Contains the main content of the document.
<h1> to <h6>	Defines headings, <h1> is the largest.
<p>	Defines a paragraph.
<a>	Creates hyperlinks using the href attribute.
<img>	Embeds images using the src and alt attributes.
	Defines an unordered (bulleted) list.
	Defines an ordered (numbered) list.
	Defines a list item.
<table>	Creates a table structure.
<tr>	Table row.
<td>	Table cell.
<th>	Table header.
	Defines a block container.
	Defines an inline container.
<form>	Defines a form for user input.
<input>	Defines an input field (e.g., text, button).

<button>	Defines a clickable button.

Table No. 2.1.1 HTML Tags

2.2 CSS (Cascading Style Sheets)

CSS (Cascading Style Sheets) was introduced in 1996 by the World Wide Web Consortium(W3C) as a solution to separate the content of web pages from their presentation. The concept was first proposed by Hakon Wium Lie in 1994, aiming to provide web developers with a way to control the appearance of web pages without altering their HTML structure. The first specification, CSS1, included basic styling features such as font customization, text alignment, and margins. In 1998, CSS2 expanded its capabilities with features like positioning, media types, and table styling. However, inconsistent browser support slowed adoption during the early 2000s. CSS3, introduced as a modular update, brought transformative features like animations, transitions, and responsive design through media queries. Today, CSS remains a fundamental technology in web development, empowering developers to create visually engaging and responsive websites. It defines the presentation of HTML documents, including layout, colours, fonts, and spacing, ensuring a seamless user experience across devices.

Key Features:

- **Selectors:** Targets HTML elements (e.g., p, #id, .class).
- **Box Model:** Defines padding, border, and margin for layout.
- **Flexibility:** Enables responsive design and animations.

Common CSS Properties and Their Descriptions:

Property	Description
Colour	Sets the text colour.

font-family	Specifies the font.
font-size	Sets the text size.
Property	**Description**
Background-colour	Sets the background colour of an element.
Margin	Defines space outside the element's border.
Padding	Defines space inside the element's border.
Border	Defines the border around an element.
text-align	Aligns text (e.g., left, right, center).
Display	Specifies layout behaviour (e.g., block, inline).

Table No. 2.2.1 CSS Tags

2.3 JAVA SCRIPT

JavaScript is a high-level, lightweight, and versatile programming language primarily used to create interactive and dynamic web applications. It enables developers to add features such as animations, form validations, and real-time content updates. Initially developed by Brendan Eich at Netscape in 1995, it was originally called Mocha, later renamed LiveScript, and eventually JavaScript to align with the popularity of Java at the time. JavaScript gained widespread adoption with the introduction of ECMAScript standards in 1997, which ensured consistency across implementations. Over the years, JavaScript has evolved significantly, adding modern features like asynchronous programming, modularization, and extensive libraries, making it a cornerstone of modern web development alongside HTML and CSS.

Key Features and examples:

- **Basics:** Understanding variables (var, let, const), data types, operators, and expressions.
- **Functions:** Writing reusable blocks of code to perform specific tasks.
- **DOM Manipulation:** Selecting and modifying elements dynamically.
- **Event Handling:** Responding to user actions like clicks and keypresses.
- **Form Validation:** Ensuring proper user input.

CHAPTER 3

REQUIREMENT SPECIFICATION

3.1 HARDWARE REQUIREMENTS:

1. **Processor:**
   - Minimum: Dual-core processor (e.g., Intel Core i3 or equivalent).
   - Recommended: Quad-core processor or higher for multitasking.

2. **RAM**:
   - Minimum: 2 GB (sufficient for lightweight operations).

3. **Storage:**
   - A minimum of 100 MB of free space (to store the HTML file, assets, and browser cache).

4. **Display:**
   - Screen resolution of at least 1024x768 to accommodate the form UI.
   - A modern display supporting standard web colours.

5. **Input Devices:**
   - Keyboard and mouse (or touch input for tablets and mobile devices).

6. **Internet Connection:**
   - Required since some animations in the document are procured from sources in the internet.

3.2 SOFTWARE REQUIREMENTS:

1. **Operating System:**
    - Windows (7 or higher), macOS (10.12 or higher), or a modern Linux distribution.
    - For mobile: Android 5.0+ or iOS 10+.

2. **Web Browser:**
    - Modern web browsers like:
    - Google Chrome (latest version preferred).
    - Mozilla Firefox (latest version preferred).
    - Microsoft Edge (latest version).
    - Safari (for macOS/iOS).
    - Ensure the browser supports ES6 JavaScript and CSS3 features.

3. **Text Editor/IDE (for development or modification):**
    - Notepad, Visual Studio Code, Sublime Text, Atom, or any basic text editor.

4. **Web Server (Optional):**
    - For local hosting (if not opening directly in a browser):
    - Use a lightweight server like Python's built-in HTTP server (python -m http.server) or tools like XAMPP.

5. **Dependencies:**
    - No external libraries are used, so no additional installations are required.
    - Ensure JavaScript is enabled in the browser.

CHAPTER 4

DESIGN

4.1 DESIGN GOALS

1. Simplicity and Clarity – Clean layout with clear instructions.

2. Multi-Step Navigation – Avoid overwhelming the user.

3. Smooth Transitions – Using CSS animations.

4. Progress Indication – Dynamic progress bar for step tracking.

5. Responsiveness – Works on desktops, tablets, and phones.

6. Readable Typography – Simple fonts and spacing.

7. Consistent Colour Theme – Professional and modern appearance.

CHAPTER 5

IMPLEMENTATION

5.1 HTML

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Hospital Management System</title>
</head>
<body>
  <header>
    <h1>CityCare Hospital</h1>
    <p>Modern. Professional. Trusted Healthcare.</p>
  </header>

  <nav>
    <a href="#patients">Patient Registration</a>
    <a href="#appointments">Appointments</a>
    <a href="#doctors">Doctors</a>
  </nav>

  <div class="hero">
    <div class="hero-text">
      <h2>Welcome to CityCare Hospital</h2>
      <p>Your health is our top priority. Book appointments, register patients and explore our
      expert doctors with ease.</p>
    </div>
    <img src="https://cdn-icons-png.flaticon.com/512/2966/2966327.png" width="200" />
  </div>

  <div class="container">
    <section id="patients" class="card">
      <h2>Patient Registration</h2>
      <input type="text" id="pname" placeholder="Patient Name" />
      <input type="number" id="page" placeholder="Age" />
      <select id="pgender">
        <option value="">Select Gender</option>
        <option value="Male">Male</option>
        <option value="Female">Female</option>
        <option value="Other">Other</option>
      </select>
      <button class="btn" onclick="registerPatient()">Register</button>
      <p id="pmsg"></p>
    </section>

    <section id="appointments" class="card">
      <h2>Book Appointment</h2>
```

Figure 5.1.1 – HTML(i)

```
  <input type="text" id="aname" placeholder="Patient Name" />
  <input type="date" id="adate" />
  <select id="adoctor">
    <option value="">Select Doctor</option>
    <option>Dr. Rao – Cardiologist</option>
    <option>Dr. Sharma – Neurologist</option>
    <option>Dr. Mehta – Orthopedic</option>
    <option>Dr. Pao – Gynecologist</option>
    <option value="">Dr. Varma – ENT specalist</option>
    <option value="">Dr. Jheta – Opthamologist</option>
  </select>
  <button class="btn" onclick="bookAppointment()">Book</button>
  <p id="amsg"></p>
</section>

<section id="doctors" class="card">
  <h2>Doctors List</h2>
  <div id="docList" class="doctor-cards">
    <!-- Example Doctor Card -->
    <div class="doctor-card">
      <img src="https://www.sheknows.com/wp-content/uploads/2024/04/kanye-west.jpg?w=1440"
      alt="Dr. Rao" class="doctor-img" />
      <div class="doctor-details">
        <h3>Dr. Rao</h3>
        <p>Cardiologist</p>
      </div>
    </div>
    <div class="doctor-card">
      <img src="https://img.buzzfeed.com/buzzfeed-static/static/2015-01/30/11/enhanced/
      webdr09/original-26702-1422635651-12.jpg?downsize=700%3A%2A&output-quality=auto&
      output-format=auto" alt="Dr. Sharma" class="doctor-img" />
      <div class="doctor-details">
        <h3>Dr. Sharma</h3>
        <p>Neurologist</p>
      </div>
    </div>
    <div class="doctor-card">
      <img src="https://static01.nyt.com/images/2023/10/26/arts/24morgan-freeman1/
      24morgan-freeman1-mediumSquareAt3X.jpg" alt="Dr. Mehta" class="doctor-img" />
      <div class="doctor-details">
        <h3>Dr. Mehta</h3>
```

Figure 5.1.2 – HTML(ii)

```
            <p>Orthopedic</p>
          </div>
        </div>
        <div class="doctor-card">
          <img src="https://s43365.pcdn.co/wp-content/uploads/2022/12/Jackie-Chan.jpeg"
          alt="Dr. Pao" class="doctor-img" />
          <div class="doctor-details">
            <h3>Dr. Pao</h3>
            <p>Gynecologist</p>
          </div>
        </div>
        <div class="doctor-card">
          <img src="https://res.cloudinary.com/dr6f47ka1/image/fetch/w_300,h_300,c_fill,g_face,
          f_auto/https://works-files.s3.amazonaws.com/5e9499fb-4913-4233-bd2f-e07ef51fd245"
          alt="Dr. Varma" class="doctor-img" />
          <div class="doctor-details">
            <h3>Dr. Varma</h3>
            <p>ENT specalist</p>
          </div>
        </div>
        <div class="doctor-card">
          <img src="https://encrypted-tbn0.gstatic.com/images?
          q=tbn:ANd9GcRmCi7Aj_raDf7MqKi4339cxavMWrj-uBzlEg&s" alt="Dr. Jheta"
          class="doctor-img" />
          <div class="doctor-details">
            <h3>Dr. Jheta</h3>
            <p>Opthamologist</p>
          </div></div></div>
    </section>
  </div>
  <footer>
    <p>© 2025 CityCare Hospital - All Rights Reserved</p>
  </footer>
</body>
</html>
```

Figure 5.1.3 – HTML(iii)

5.2 CSS

```
<style>
  :root {
    --primary: #0a74da;
    --primary-dark: #084c95;
    --bg: #f5f7fa;
    --card-bg: #ffffff;
    --shadow: 0 4px 18px rgba(0,0,0,0.1);
  }
  body {
    font-family: "Segoe UI", sans-serif;
    margin: 0;
    background: var(--bg);
  }
  header {
    background: var(--primary);
    color: #fff;
    padding: 50px 20px;
    text-align: center;
    box-shadow: var(--shadow);
  }
  header h1 {
    font-size: 2.5rem;
    margin: 0;
    letter-spacing: 1px;
  }
  nav {
    background: #fff;
    padding: 12px 30px;
    display: flex;
    justify-content: center;
    gap: 40px;
    box-shadow: var(--shadow);
    position: sticky;
    top: 0;
    z-index: 1000;
  }
```

Figure 5.2.1 – CSS(i)

```css
nav a {
  color: var(--primary-dark);
  text-decoration: none;
  font-weight: bold;
  font-size: 1rem;
  padding-bottom: 6px;
  border-bottom: 3px solid transparent;
  transition: 0.3s;
}
nav a:hover {
  border-bottom: 3px solid var(--primary);
}
.hero {
  margin: 40px auto;
  max-width: 1100px;
  background: var(--card-bg);
  padding: 30px;
  border-radius: 16px;
  box-shadow: var(--shadow);
  display: flex;
  justify-content: space-between;
  align-items: center;
  gap: 30px;
}
.hero-text h2 {
  font-size: 2rem;
  margin-bottom: 10px;
  color: var(--primary-dark);
}
.container {
  padding: 20px;
  max-width: 1100px;
  margin: auto;
}
```

Figure 5.2.2 – CSS(ii)

```css
.card {
  background: var(--card-bg);
  padding: 25px;
  margin: 25px 0;
  border-radius: 14px;
  box-shadow: var(--shadow);
  transition: 0.3s;
}
.card:hover {
  transform: translateY(-3px);
}
h2 {
  color: var(--primary-dark);
  margin-bottom: 20px;
}
.btn {
  background: var(--primary);
  color: #fff;
  border: none;
  padding: 12px 18px;
  border-radius: 8px;
  cursor: pointer;
  font-weight: bold;
  transition: 0.3s;
}
.btn:hover {
  background: var(--primary-dark);
}
input, select {
  width: 95%;
  padding: 12px;
  margin: 10px 0;
  border-radius: 8px;
  border: 1px solid #ccc;
  font-size: 1rem;
}
```

Figure 5.2.3 – CSS(iii)

```css
.doctor-cards {
  display: flex;
  gap: 20px;
  flex-wrap: wrap;
  justify-content: space-between;
}

.doctor-card {
  background: var(--card-bg);
  padding: 20px;
  border-radius: 14px;
  box-shadow: var(--shadow);
  display: flex;
  align-items: center;
  gap: 15px;
  width: 280px;
  transition: 0.3s;
}

.doctor-card:hover {
  transform: translateY(-3px);
}

.doctor-img {
  width: 80px;
  height: 80px;
  border-radius: 50%;
  object-fit: cover;
}
```

Figure 5.2.4 – CSS(iv)

```
  .doctor-details h3 {
    font-size: 1.2rem;
    margin: 0;
    color: var(--primary-dark);
  }

  .doctor-details p {
    margin: 5px 0 0;
    color: #555;
    font-size: 0.9rem;
  }

  footer {
    text-align: center;
    padding: 20px;
    background: var(--primary);
    color: #fff;
    margin-top: 40px;
  }
</style>
```

Figure 5.2.5 – CSS(v)

5.3 JAVASCRIPT

```
<script>
  function registerPatient() {
    let name = document.getElementById('pname').value;
    let age = document.getElementById('page').value;
    let gender = document.getElementById('pgender').value;
    if (!name || !age || !gender) {
      document.getElementById('pmsg').textContent = "Please fill all fields";
      return;
    }
    document.getElementById('pmsg').textContent = "Patient Registered Successfully!";
  }

  function bookAppointment() {
    let name = document.getElementById('aname').value;
    let date = document.getElementById('adate').value;
    let doctor = document.getElementById('adoctor').value;
    if (!name || !date || !doctor) {
      document.getElementById('amsg').textContent = "Please fill all fields";
      return;
    }
    document.getElementById('amsg').textContent = "Appointment Booked Successfully!";
  }

  // Smooth scrolling for nav links
  document.querySelectorAll('nav a').forEach(anchor => {
    anchor.addEventListener('click', function(e) {
      e.preventDefault(); // Prevent default jump
      const targetID = this.getAttribute('href').substring(1); // remove #
      const targetSection = document.getElementById(targetID);
      if (targetSection) {
        window.scrollTo({
          top: targetSection.offsetTop - 80, // adjust offset for sticky nav
          behavior: 'smooth'
        });
      }
    });
  });
</script>
```

Figure 5.3 - JavaScript

CHAPTER 6

RESULTS

6.1 HEADER SECTION & NAVIGATION BAR

Figure 6.1 – Header section & Navigation Bar

6.2 HERO SECTION

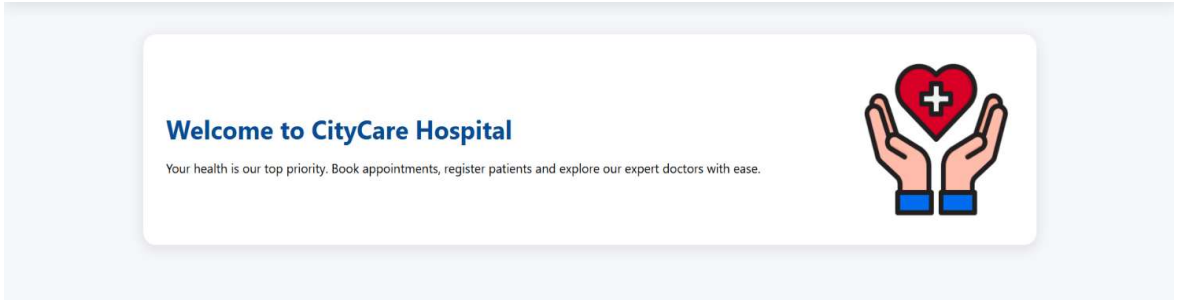

Figure 6.2 – Hero Section

6.3 PATIENT REGISTRATION SECTION

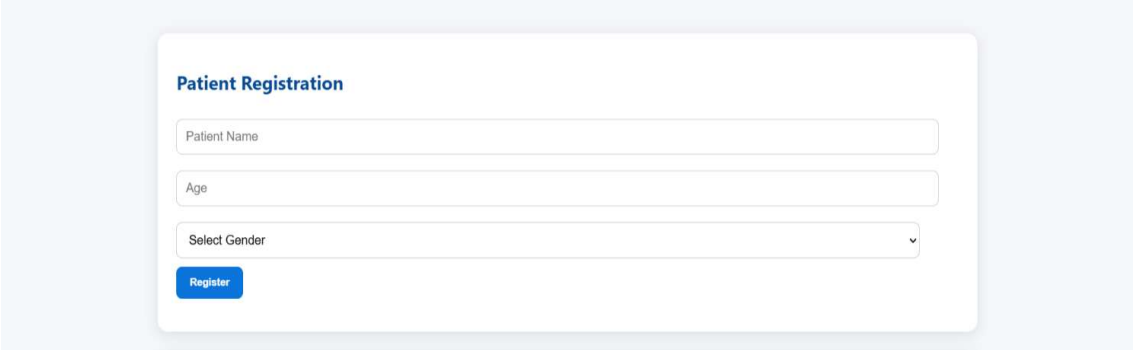

Figure 6.3 – Patient Registration Section

6.4 APPOINTMENT BOOKING SECTION

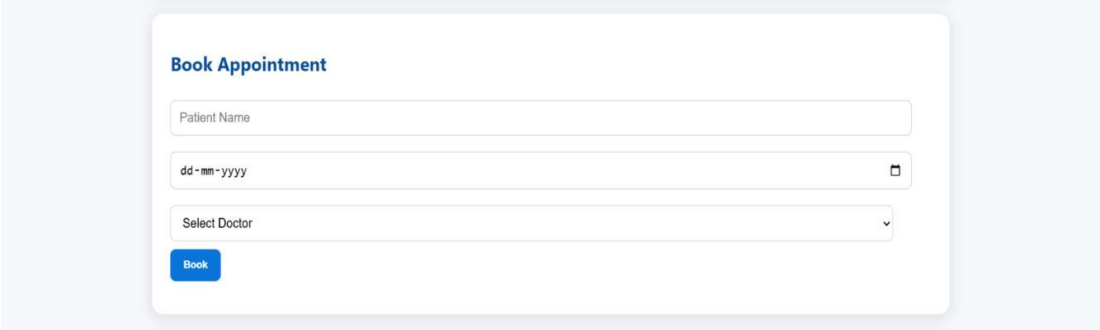

Figure 6.4 – Appointment Booking Section

6.5 DOCTORS LIST SECTION

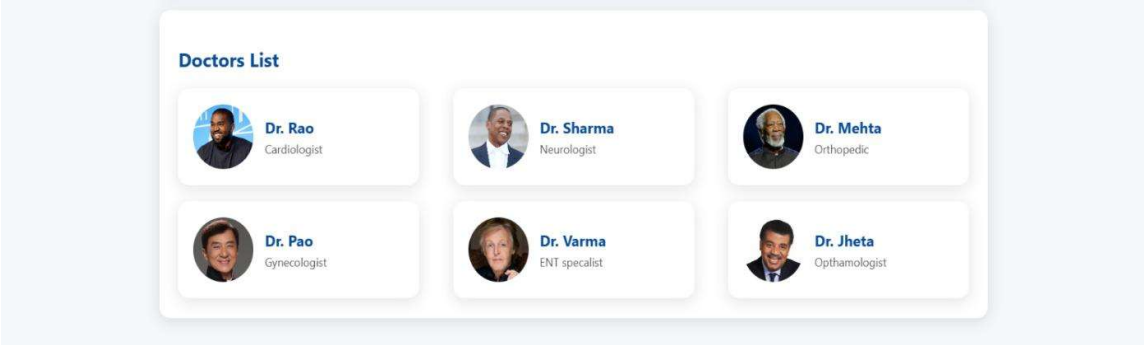

Figure 6.5 – Doctors List Section

6.6 FOOTER SECTION

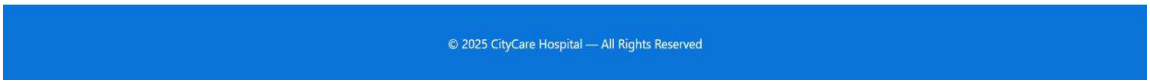

Figure 6.6 – Footer Section

CHAPTER 7

CONCLUSION

The hospital management system designed to make everyday tasks like registering patients and booking appointments simpler and more efficient. The page is neatly divided into sections that focus on different aspects, such as patient registration, making appointments, and browsing doctor profiles. The layout is clean and easy to navigate, with interactive elements like buttons and dropdown menus that guide users through each step. The design ensures that no matter what device you're using, the interface remains clear and accessible, creating a seamless experience for both hospital staff and patients.

Behind the scenes, the system is set up to provide immediate feedback and make sure all necessary information is filled out before proceeding. For example, when registering a patient or booking an appointment, the system checks if every field is complete and notifies the user if anything is missing. This helps prevent errors and ensures that everything runs smoothly. The design also uses consistent colors and styles across the site, creating a professional and cohesive look that matches the modern and reliable image that a hospital needs to project to its visitors.

REFERENCES

- https://devdevout.com/css/css-animated-backgrounds
- https://www.w3schools.com/tags/tag_html.asp
- https://github.com/topics/web-development-project
- https://www.w3schools.com/js/js_object_property.asp
- https://developer.mozilla.org/en-US/docs/Learn/CSS