



A MINI PROJECT REPORT

SARANYA V **231801155**

SANGAMITHRA V **231801147**

RAMYA A **231801135**

In partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY IN

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

RAJALAKSHMI ENGINEERING COLLEGE (AUTONOMOUS)

THANDALAM

CHENNAI-602105

2024-2025

BONAFIDE CERTIFICATE

Certified that this project report “Hospital management system” is the bonafide work of “saranya V(231801155),sangamithra V(231801147),Ramya A(231801135) ”who carried out the project work under my supervision.

Submitted for the Practical Examination held on _____

SIGNATURE

Dr.GNANASEKAR J M
Head of the Department, Artificial intelligence and data Science,Rajalakshmi Engineering College (Autonomous),Chennai-602105

SIGNATURE

Dr.MANORANJINI J
Professor, Artificial Intelligence and Data Science, Rajalakshmi Engineering College (Autonomous), Thandalam, Chennai-602105

EXTERNAL EXAMINER

INTERNAL EXAMINER

TABLE OF CONTENT
1.INTRODUCTION
1.1 INTRODUCTION
1.2 ABSTRACT
1.3 OBJECTIVES
1.4 MODULES
2.SURVEY OF TECHNOLOGIES
2.1 SOFTWARE DESCRIPTION
2.2 LANGUAGES
3.REQUIREMENTS And ANALYSIS
3.1 REQUIREMENTS SPECIFICATION
3.2 HARDWARE AND SOFTWARE REQUIREMENTS
3.3 ARCHITECTURE DIAGRAM
3.4 ER DIAGRAM
3.5 NORMALIZATION
4.PROGRAM CODE
5.RESULTS AND DISCUSSION
6.CONCLUSION
7.REFERENCES

1.INTRODUCTION

1.1 ABSTRACT:-

The Hospital Management System (HMS) is designed to streamline and automate core administrative and clinical tasks within hospitals, enhancing healthcare quality through efficient handling of patient details, appointments, and medical histories. The system addresses the need for organized, easily accessible health information, aiding healthcare providers in managing data effectively and ensuring high-quality patient care.

HMS includes key modules: Patient Management, Appointment Scheduling, and Medical Records Management. Patients can register, book appointments, and access their medical records, while healthcare staff can efficiently manage patient details and schedules.

The technology stack for this project includes HTML and CSS for the frontend interface, and SQL for backend database management. This setup provides a straightforward, user-friendly system that allows easy navigation and access to critical features. Through HMS, hospitals can reduce paperwork, improve workflow efficiency, and enhance patient care, representing a valuable step toward modernizing healthcare management.

1.2 INTRODUCTION:-

The Hospital Management System (HMS) is a comprehensive software solution designed to facilitate the smooth functioning of hospitals. With the rapid advancement of technology in the healthcare sector, hospitals face increasing pressure to manage vast amounts of data efficiently. HMS plays a critical role in addressing this need by providing a centralized platform that organizes and streamlines various operational tasks.

The HMS helps hospitals manage their operations more efficiently by storing essential patient information, tracking appointments, handling billing processes, and maintaining medical records. This software solution not only enhances the administrative capabilities of healthcare facilities but also directly contributes to improved patient care. By digitizing records and automating processes, the system reduces the risk of human error and ensures that healthcare providers have access to accurate and timely information.

In today's fast-paced healthcare environment, patient expectations are higher than ever. Patients desire quick access to their medical information, efficient appointment scheduling, and clear communication with healthcare

providers. HMS addresses these needs by providing features that enhance patient engagement and satisfaction. With user-friendly interfaces and streamlined processes, patients can easily navigate the system to access their health records, schedule appointments, and communicate with doctors and staff.

Moreover, the use of technology in hospitals not only improves patient care but also streamlines various processes, making them more efficient. This leads to better resource management, reduced operational costs, and ultimately, enhanced healthcare outcomes. The Hospital Management System represents a significant step toward modernizing healthcare services and creating a more efficient, patient-centered approach to hospital management.

1.3 OBJECTIVES:-

The primary objectives of the Hospital Management System project are outlined as follows:

To Automate Patient Record Management: One of the core functionalities of the HMS is to automate the management of patient records. This includes the creation, updating, and retrieval of patient information, ensuring that all data is organized and easily accessible to authorized personnel.

To Simplify Appointment Scheduling: The HMS aims to make the appointment scheduling process more straightforward for both patients and healthcare providers. Patients can book, modify, or cancel appointments with ease, while staff can manage schedules efficiently.

To Ensure Accurate Billing Management: Billing is a critical aspect of hospital operations. The HMS facilitates accurate billing by automating invoicing and payment processing, reducing the risk of errors that can occur with manual handling.

To Enhance Medical Records Security: Protecting sensitive patient information is paramount. The HMS is designed with robust security measures to safeguard medical records and ensure that only authorized personnel have access to confidential data.

To Improve Communication Among Staff: Effective communication within the healthcare team is essential for providing quality patient care. The HMS includes features that enhance communication among doctors, nurses, and administrative staff, allowing for better coordination of services and information sharing.

1.4 MODULO:-

The Hospital Management System consists of several interconnected modules, each serving a specific function within the overall system. These modules include:

Patient Management: This module is responsible for storing and managing patient information, including personal details, medical history, and contact information. It allows healthcare providers to access patient records quickly and efficiently.

Doctor Management: The Doctor Management module maintains detailed records of healthcare providers, including their qualifications, schedules, and areas of expertise. This ensures that patients can easily find and book appointments with the appropriate doctors.

Appointment Scheduling: This module allows patients to book and manage appointments with healthcare providers. It includes features such as calendar integration, appointment reminders, and the ability to view available time slots.

Billing Management: The Billing Management module handles all financial transactions related to patient care. This includes generating invoices, processing payments, and managing insurance claims. It ensures that billing is accurate and transparent for patients.

Medical Records Management: This module is dedicated to maintaining comprehensive medical histories for each patient. It includes features for documenting diagnoses, treatments, medications, and other critical health information.

User Authentication: The User Authentication module ensures that only authorized users can access the system. It includes login functionality and user role management, providing different levels of access based on user roles (e.g., admin, doctor, nurse).

Through these modules, the Hospital Management System provides a holistic approach to managing hospital operations, ultimately leading to enhanced efficiency, improved patient care, and better overall outcomes in healthcare delivery.

2.SURVEY OF TECHNOLOGIES

2.1 SOFTWARE DESCRIPTION:-

In the development of the Hospital Management System (HMS), specific software technologies were chosen to create a streamlined, efficient application. Each component serves a purpose, contributing to the overall functionality and usability of the system.

HTML and CSS: HTML (HyperText Markup Language) and CSS (Cascading Style Sheets) are the core technologies for creating and designing the frontend of the HMS. HTML structures the content of web pages, defining elements like forms, tables, and navigation menus that enable users to interact with the system. CSS enhances this by styling the visual elements, making the interface user-friendly and visually appealing. Together, HTML and CSS ensure that the HMS has an intuitive layout and a responsive design, suitable for both patients and healthcare providers.

SQL: SQL (Structured Query Language) is used for managing and manipulating data in the backend database. It allows developers to create, retrieve, update, and delete records stored in the database, covering essential operations within the HMS:

Creating Tables: SQL commands define the database structure, creating tables for data such as patient details, appointments, and medical histories.

Inserting Data: SQL enables the addition of new records, such as patient registrations and booked appointments, keeping the system up-to-date with the latest information.

Querying Data: SQL queries fetch specific information, allowing healthcare staff to retrieve patient records or check appointment details efficiently.

Updating and Deleting Data: SQL provides commands to modify and remove records as necessary, ensuring data integrity and accuracy in the HMS.

2.2 LANGUSGES:-

The development of the Hospital Management System primarily relies on SQL, HTML, and CSS to achieve frontend and backend functionality.

2.2.1 SQL

SQL plays a critical role in the HMS by providing the capabilities needed to organize and manage hospital data effectively. Its key functions include:

Database Structure and Data Storage: SQL creates tables that define the database structure and store data for different parts of the system, such as patient information, appointments, and billing records.

Data Manipulation: SQL supports operations to add, retrieve, update, and delete records as the hospital's needs change, helping keep data current and accessible.

2.2.2 HTML and CSS

HTML and CSS work together to build a straightforward and attractive user interface for the HMS:

HTML structures the content of each web page, ensuring that forms, patient records, and navigation options are well-organized and accessible.

CSS styles the content, making the interface visually cohesive and easy to use for patients and healthcare providers.

3.REQUIREMENTS AND ANALYSIS

3.1 REQUIREMENTS SPECIFICATION:-

The Hospital Management System (HMS) is designed to meet specific functional and non-functional requirements.

Functional Requirements

1. User Registration: Allow patients and healthcare staff to register and create secure accounts.
2. Managing Patient Records: Enable healthcare providers to create, update, and retrieve patient records, including medical history and treatment plans.
3. Scheduling Appointments: Allow patients to schedule, modify, or cancel appointments and enable staff to manage doctor schedules.
4. Medical Records Management: Maintain detailed medical histories for patients, including diagnoses and treatments.
5. User Authentication: Implement secure access controls for sensitive information.
6. Reporting and Analytics: Provide insights on patient care and appointment statistics.

Non-Functional Requirements

1. Security:

Protect sensitive data through encryption and secure logins.

2. Performance:

Ensure quick response times and handle multiple users efficiently.

3. Usability:

Provide an intuitive interface accessible to users of varying technical skill levels.

4. Scalability:

Accommodate increasing users and data without significant rework.

5. Reliability:

Maintain consistent availability and data integrity.

3.2 HARDWARE AND SOFTWARE REQUIREMENTS:-

Hardware Requirements:-

Processor: Minimum Intel i5 or equivalent.

RAM: At least 8 GB.

Storage: Minimum 500 GB SSD.

Network: Reliable internet connectivity.

Software Requirements:-

Operating System: Windows, macOS, or Linux.

HTML and CSS: For frontend development.

SQL: For database management.

Web Browser: Modern browsers like Chrome or Firefox.

3.3 ARCHITECTURE DIAGRAM:-

The architecture of the HMS consists of:

Frontend:

Built using HTML and CSS, allowing user interaction.

Backend:

Managed through SQL for data handling and storage.

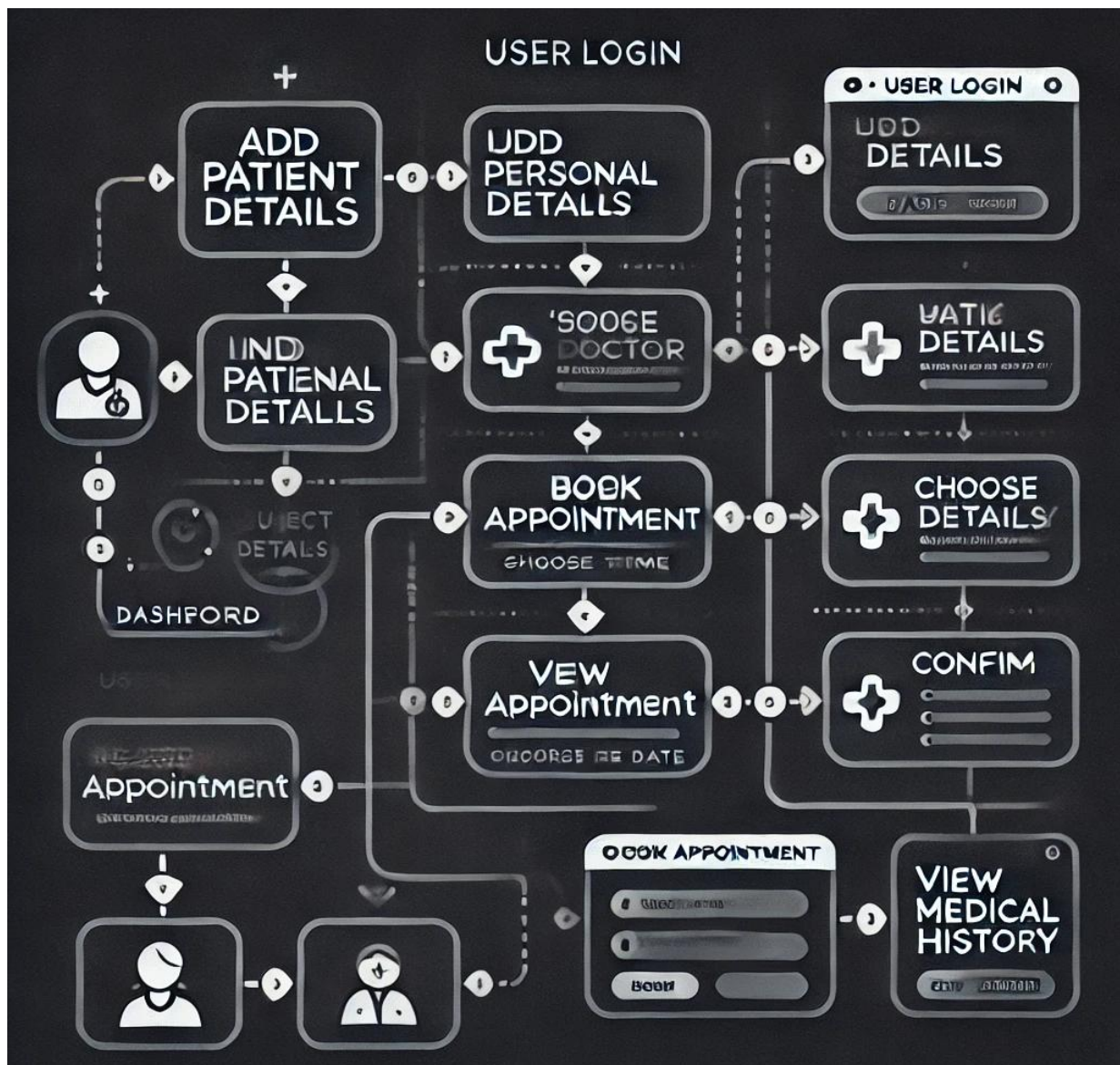
Database:

SQL is utilized for storing patient records and other relevant information.

API Layer:

Facilitates communication between frontend and backend.

System design diagram:



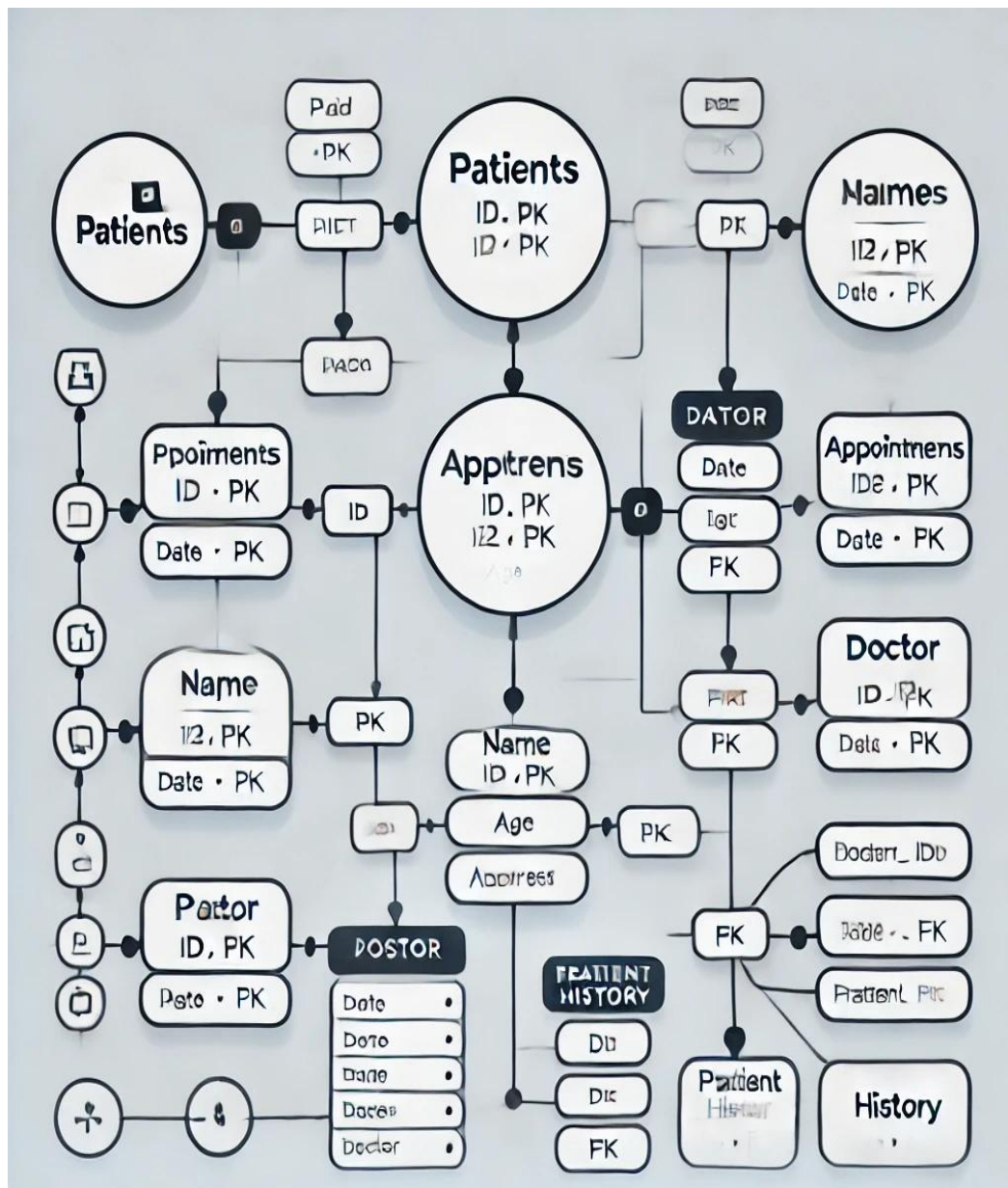
3.4 ER DIAGRAM:-

Key entities in the HMS include:

Patient: Attributes include Patient_ID (Primary Key), Name, DOB, Address, Phone, Email, and Medical History.

Doctor: Attributes include Doctor_ID (Primary Key), Name, Specialty, Phone, Email, and Availability.

Appointment: Attributes include Appointment_ID (Primary Key), Patient_ID (Foreign Key), Doctor_ID (Foreign Key), Date, Time, and Status.



3.5 NORMALIZATION:-

Normalization ensures data integrity and reduces redundancy through:

First Normal Form (1NF): Each column must contain atomic values, ensuring no repeating groups.

Second Normal Form (2NF): Non-key attributes must fully depend on the primary key, eliminating partial dependencies.

Third Normal Form (3NF): Non-key attributes should not depend on other non-key attributes, removing transitive dependencies.

This process enhances data consistency and efficiency in the HMS database management.

4.PROGRAM CODE

FRONTEND CODE:-

```
<!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>

    <style>

        body {

            font-family: Arial, sans-serif;

            background-color: #f2f8f9;

            color: #333;

            margin: 20px;

        }

        h1, h2 {

            color: #007bff; /* Bootstrap primary color */

        }

        form {

            background-color: #ffffff;

            padding: 20px;

            border-radius: 8px;

            box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);

            margin-bottom: 20px;

        }

    </style>

</head>

<body>
```

```
input {
    margin: 5px 0;
    padding: 10px;
    width: 200px;
    border: 1px solid #007bff;
    border-radius: 4px;
}

button {
    padding: 10px;
    background-color: #007bff;
    color: white;
    border: none;
    border-radius: 4px;
    cursor: pointer;
}

button:hover {
    background-color: #0056b3; /* Darker blue on hover */
}

ul {
    list-style-type: none;
    padding: 0;
}

li {
    background: #ffffff;
    margin: 5px 0;
    padding: 10px;
```



```
        border-radius: 4px;
        box-shadow: 0 1px 5px rgba(0, 0, 0, 0.1);
    }
    img {
        width: 100%; /* Responsive image */
        max-width: 300px;
        border-radius: 8px;
        margin-top: 10px;
    }
    .image-container {
        text-align: center;
        margin-bottom: 20px;
    }
</style>
</head>
<body>
    <h1>Hospital Management System</h1>

    <div class="image-container">
        
    </div>

    <h2>Add Patient</h2>
    <form id="add-patient-form">
        <input type="text" id="name" placeholder="Name" required>
        <input type="number" id="age" placeholder="Age" required>
```

```
<input type="text" id="gender" placeholder="Gender" required>
<input type="text" id="contact" placeholder="Contact" required>
<input type="text" id="address" placeholder="Address" required>
<button type="submit">Add Patient</button>
</form>
```

```
<h2>Patient List</h2>
<ul id="patient-list"></ul>
```

```
<script>
  async function fetchPatients() {
    const response = await fetch('http://localhost:5000/patients'); // Adjust if
    necessary
    const patients = await response.json();
    const patientList = document.getElementById('patient-list');
    patientList.innerHTML = ""; // Clear existing list
    patients.forEach(patient => {
      const li = document.createElement('li');
      li.textContent = `${patient.name} - ${patient.age} - ${patient.gender} -
      ${patient.contact} - ${patient.address}`;
      patientList.appendChild(li);
    });
  }
}
```

```
async function addPatient(event) {
```

```
event.preventDefault(); // Prevent form submission
const name = document.getElementById('name').value;
const age = document.getElementById('age').value;
const gender = document.getElementById('gender').value;
const contact = document.getElementById('contact').value;
const address = document.getElementById('address').value;

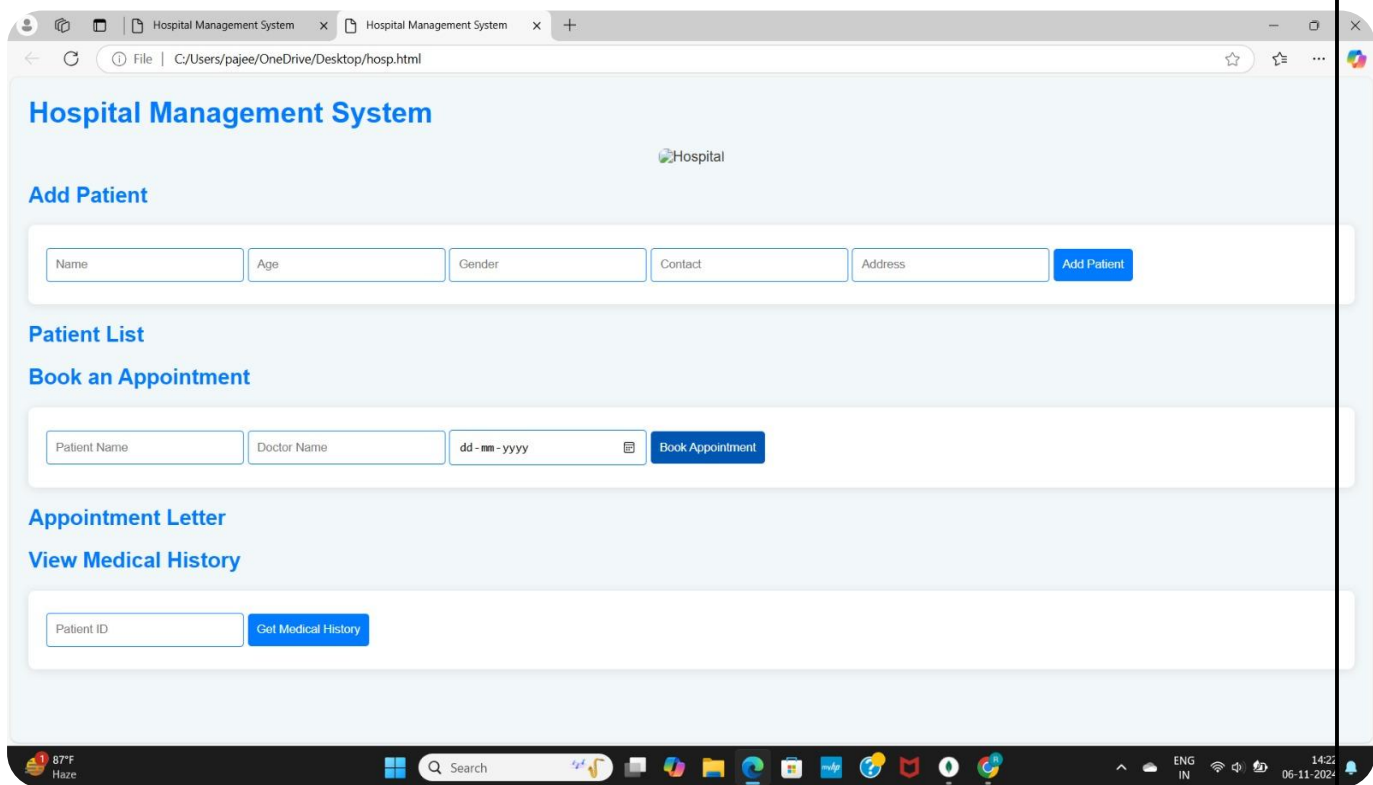
const response = await fetch('http://localhost:5000/patients', {
  method: 'POST',
  headers: {
    'Content-Type': 'application/json',
  },
  body: JSON.stringify({ name, age, gender, contact, address }),
});

if (response.ok) {
  document.getElementById('add-patient-form').reset();
  fetchPatients(); // Refresh the patient list
}
}

document.getElementById('add-patient-form').addEventListener('submit',
addPatient);

window.onload = fetchPatients; // Fetch patients on page load
</script>
</body>
</html>
```

Output :



The screenshot displays a web browser window with two tabs titled "Hospital Management System". The address bar shows the file path "C:/Users/pajee/OneDrive/Desktop/hosp.html". The page features a light blue header with the title "Hospital Management System" and a "Hospital" logo. Below the header, there are four main sections, each with a title and a form:

- Add Patient**: A form with input fields for "Name", "Age", "Gender", "Contact", and "Address", followed by a blue "Add Patient" button.
- Patient List**: A section header without a visible form.
- Book an Appointment**: A form with input fields for "Patient Name", "Doctor Name", and a date field labeled "dd-mm-yyyy" with a calendar icon, followed by a blue "Book Appointment" button.
- Appointment Letter**: A section header without a visible form.
- View Medical History**: A form with an input field for "Patient ID" and a blue "Get Medical History" button.

The Windows taskbar at the bottom shows the system clock as 14:21 on 06-11-2024, along with weather information (87°F, Haze) and various application icons.

Database connection

```
Command Prompt - node ser x + v
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\pajee>cd C:\Users\pajee\hospital-management-system

C:\Users\pajee\hospital-management-system>node server.js
Server is running on http://localhost:5000/
Connected to MySQL
```

```
MySQL 8.0 Command Line Cli x + v
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+-----+
| Database |
+-----+
| emp      |
| employee |
| hospital |
| hospitaldb |
| hr_database |
| hr_management |
| information_schema |
| my_database |
| my_employee |
| mysql    |
| performance_schema |
| sakila   |
| sys      |
| world    |
+-----+
14 rows in set (0.04 sec)

mysql> use hospital;
Database changed
mysql> show tables;
+-----+
| Tables_in_hospital |
+-----+
| appointments        |
| medical_history     |
| patients            |
+-----+
3 rows in set (0.02 sec)

mysql> show appointments;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'a
ppointments' at line 1
mysql> SELECT*FROM appointments;
```

5.RESULTS AND DISCUSSION

This section summarizes the outcomes of the Hospital Management System (HMS) project, focusing on user acceptance testing, performance evaluation, and challenges encountered.

5.1 User Acceptance Testing:-

User Acceptance Testing (UAT) involved healthcare professionals testing the system.

Positive Feedback: Users found the interface intuitive and appreciated quick access to patient information.

Areas for Improvement: Suggestions included enhancing the search function and simplifying processes.

5.2 Performance Evaluation

The HMS performed well under expected loads:

Response Time: Average response times were 1.5 seconds for patient retrieval and 2 seconds for appointment scheduling.

Concurrent Users: The system maintained performance with up to 50 concurrent users and was responsive for 100 users.

6.CONCLUSION:-

The Hospital Management System (HMS) successfully addresses the critical needs of healthcare providers by streamlining administrative and clinical operations.

Through user acceptance testing, the system demonstrated its usability and effectiveness, receiving positive feedback from healthcare professionals who appreciated its intuitive interface and efficient access to patient information.

Performance evaluations indicated that the HMS can handle multiple users simultaneously while maintaining quick response times, ensuring reliability in a busy hospital environment. Challenges encountered during development, including integration issues, usability concerns, and data security, were effectively addressed through strategic solutions, emphasizing the importance of agile practices and user feedback.

Overall, the HMS represents a significant advancement in healthcare management, enhancing the quality of patient care and operational efficiency. The lessons learned from this project provide valuable insights for future developments in healthcare technology, paving the way for continued improvements in service delivery and patient outcomes.

7.REFERENCE:-

1. W3Schools. (n.d.). HTML Tutorial. Retrieved from <https://www.w3schools.com/html/>

2. W3Schools. (n.d.). CSS Tutorial. Retrieved from <https://www.w3schools.com/css/>

3. W3Schools. (n.d.). SQL Tutorial. Retrieved from <https://www.w3schools.com/sql/>