# RAJALAKSHMI ENGINEERING COLLEGE [AUTONOMOUS]

## RAJALAKSHMI NAGAR, THANDALAM – 602 105





**CS23333 OBJECT ORIENTED PROGRAMMING USING JAVA**

**Laboratory Record Note Book**

EMPLOYEE REGISTRATION A MINI-PROJECT BY:

Kaaviya R(230701504) & Saranya M (230701510)

***in partial fulfillment of the award of the degree OF BACHELOR OF ENGINEERING* IN**

COMPUTER SCIENCE AND ENGINEERING.

# RAJALAKSHMI ENGINEERING COLLEGE

**[AUTONOMOUS]**

## RAJALAKSHMI NAGAR, THANDALAM – 602 105

**BONAFIDE CERTIFICATE**

**Academic Year :** 2024-2025. **Semester:** THIRD sem **Branch :** CSE

**Register No.**

**230701504 & 230701510**

Certified that this is the bonafide record of work done by KAAVIYA R(230701504) and SARANYA M(230701510) the above

student in the CS23333\_Object Oriented Programming using JAVA the year 2024-2025.

**Signature of Faculty in-charge**

Submitted for the Practical Examination held on . . . . . . . . . . . . .

**Internal Examiner External Examiner**

**CONTENTS**

**CHAPTERS PAGE NO**

Chapter 1 Introduction

* 1. Objectives 2

Chapter 2 Modules 2-3

* 1. User Authentication 2
  2. Book Appointment 3

Chapter 3 Requirements and Analysis 6-15

* 1. Requirement Specification 6-14
  2. Hardware and Software Requirements 14
  3. Software requirements 14
  4. Architecture Diagram 15
  5. ER Diagram 15

Chapter 4 Project coding 16-40

Chapter 5 Output 40-44

Chapter 6 Conclusion 44

Chapter 7 Reference 45

**HEALTH CARE MANAGEMENT SYSTEM**

**Abstract:**

The Health Care Management System (HCMS) project in Java aims to optimize and streamline healthcare operations by integrating key functionalities. This comprehensive system manages patient information efficiently, including personal details, medical histories, and treatment records, with role-based access control ensuring data security. HCMS facilitates appointment scheduling, reducing no-shows through automated reminders. Electronic Medical Records (EMR) are securely stored and updated, empowering healthcare professionals to input diagnostic information and treatment plans. The system includes billing and insurance integration for seamless financial transactions and insurance claim processing. Pharmacy and inventory management features help track medication stock levels and prevent shortages. Robust reporting and analytics tools offer insights into patient outcomes, resource utilization, and financial performance. With a focus on security and privacy, HCMS implements industry-standard measures such as data encryption, secure authentication, and audit trails. Its modular architecture allows scalability, making it suitable for diverse healthcare institutions, from small clinics to large hospitals. The user-friendly interface ensures easy navigation, contributing to the delivery of high-quality healthcare services.

1. **INTRODUCTION**

**1.1 INTRODUCTION**

The Health Care Management System (HCMS) project in Java represents a pivotal advancement in healthcare technology, aiming to revolutionize the way healthcare facilities manage their operations. In the face of evolving healthcare needs, this comprehensive system leverages Java-based technologies to address challenges in patient information management, appointment scheduling, electronic medical records, billing, pharmacy management, and analytics. By providing an integrated and user-friendly solution, HCMS seeks to enhance the overall efficiency of healthcare institutions, ultimately improving patient care and administrative processes.

Healthcare is a fundamental pillar of human well-being, encompassing a wide range of services aimed at promoting, maintaining, and restoring health. It is an ever-evolving field that integrates science, technology, and human compassion to meet the diverse needs of individuals and communities. From preventive care and diagnosis to treatment and rehabilitation, healthcare plays a vital role in improving quality of life and increasing life expectancy. In today’s world, advancements in medical research, technology, and healthcare systems have revolutionized the delivery of care, ensuring accessibility, efficiency, and patient-centered solutions. As the demand for effective healthcare services continues to grow, innovative management and holistic approaches are essential to address the challenges of resource allocation, affordability, and equitable care.

* 1. **OBJECTIVES**

The primary objective of healthcare is to enhance the overall health and well-being of individuals and communities by providing accessible, effective, and equitable care. This includes the prevention, diagnosis, and treatment of illnesses, as well as the promotion of healthy lifestyles and early detection of diseases to reduce complications. Healthcare systems aim to improve patient outcomes through evidence-based practices, the use of advanced technology, and continuous innovation. Another critical goal is to ensure the efficient use of resources while maintaining affordability and sustainability. Additionally, healthcare seeks to foster collaboration among stakeholders, including patients, providers, and policymakers, to create a supportive and inclusive environment that prioritizes patient safety, quality of care, and ethical standards.

* 1. **MODULES**

**1.3.1 USER AUTHENTICATION**

User Authentication module, ensuring secure and seamless access to the Healthcare Management System. During the registration process, new users, including patients, doctors, and administrators, provide essential details such as name, email, contact number, and role-specific information, with additional verification steps like OTP (One-Time Password) for authenticity. Sensitive information, such as passwords, is encrypted to enhance security. Once registered, users can log in using their credentials, with optional multi-factor authentication (MFA) for an added layer of protection.

**1.3.2 BOOK APPOINTMENT**

Healthcare providers, enhancing convenience and streamlining the appointment process. Patients can log into the Healthcare Management System, browse a list of available doctors or specialists, and view their schedules to select a time slot that suits their needs. The system provides real-time availability updates, ensuring that patients can book appointments without conflicts. Patients can also specify the reason for the visit, allowing doctors to prepare in advance. Once the appointment is confirmed, patients receive notifications and reminders about the scheduled consultation. This module simplifies the appointment booking process, reduces administrative workload, and ensures that patients can access timely care.

**PATIENT REQUEST**

The module allows patients to submit specific requests or queries within the Healthcare Management System, ensuring better communication with healthcare providers. Through this module, patients can make requests for follow-up appointments, medical reports, test results, or clarification on prescribed treatments. They can also inquire about any other healthcare services they may need, such as referrals or specialist consultations. Once a request is submitted, healthcare providers are notified and can respond in a timely manner, either by approving, scheduling, or providing the necessary information. This module enhances the patient experience by offering a direct and efficient way to address individual needs and ensures that no request goes unnoticed, improving overall healthcare delivery.

**DOCTOR PRESCRIPTION**

Streamline the prescribing process within the Healthcare Management System, enhancing both efficiency and accuracy. Doctors can use this module to generate digital prescriptions for patients based on their diagnosis and treatment plan. The system allows physicians to select medications, specify dosages, and provide detailed instructions for use, ensuring clarity and preventing errors. Prescriptions are stored electronically and can be easily retrieved or shared with patients, pharmacies, or other healthcare providers. Additionally, this module can integrate with medication databases to provide real-time drug interactions, warnings, and alternatives, further improving patient safety. By digitizing the prescription process, the module reduces the risk of manual errors, improves accessibility, and enhances the overall management of patient treatment plans.

**ORDER MEDICINE**

The Order Medicine module in the Healthcare Management System allows patients to conveniently order prescribed medications directly through the platform. After receiving a prescription from their doctor, patients can browse a list of available pharmacies, select their medications, and place orders for home delivery or pick-up. The system ensures that the ordered medications are in stock, displays accurate pricing, and offers payment options, including online payments. It also enables patients to track the status of their orders in real-time, from confirmation to delivery. This module enhances the patient experience by providing a seamless and accessible way to manage prescriptions, improving medication adherence and reducing the need for in-person pharmacy visits. Additionally, it supports integration with pharmacies to ensure timely and accurate fulfillment of orders.

**EXISTING SYSTEM:**

The current healthcare management landscape often relies on fragmented and manual processes, involving paper-based record-keeping and disparate systems for various functions. This disjointed approach can lead to inefficiencies, data inaccuracies, and difficulties in collaboration among healthcare professionals. Appointment scheduling may be prone to errors, and the lack of centralized electronic medical records can hinder the timely retrieval of critical patient information. The existing systems often struggle with scalability and may not fully harness the benefits of modern technology, resulting in suboptimal healthcare management.

**Disadvantages of the Existing System:**

Inefficiencies and Manual Processes: Paper-based and manual systems lead to inefficiencies in data entry, retrieval, and communication among healthcare staff.

Data Inaccuracy: The reliance on manual data entry increases the likelihood of errors in patient records, compromising the accuracy of medical information.

Limited Collaboration: Disparate systems hinder seamless collaboration among healthcare professionals, potentially impacting the quality and continuity of patient care.

Scalability Issues: Existing systems may struggle to scale with the growing demands of healthcare institutions, limiting their adaptability to evolving needs.

**PROPOSED SYSTEM:**

The proposed Health Care Management System in Java aims to address the shortcomings of the existing systems by providing a unified and scalable solution. Leveraging Java's versatility, the system integrates patient information management, appointment scheduling, electronic medical records, billing, pharmacy management, and analytics into a cohesive platform. Its modular architecture allows for flexibility and scalability, making it suitable for a range of healthcare institutions. The proposed system emphasizes automation, data accuracy, and enhanced collaboration among healthcare professionals, contributing to a more efficient and reliable healthcare management process.

**Advantages of the Proposed System:**

Efficient Information Management: HCMS streamlines patient information management, ensuring accurate and easily accessible records for healthcare professionals.

Automated Appointment Scheduling: The system automates the appointment scheduling process, reducing errors and improving overall adherence through automated reminders.

Comprehensive Electronic Medical Records: HCMS enables secure storage and retrieval of electronic medical records, empowering healthcare professionals with timely and accurate patient information.

Seamless Billing and Insurance Integration: The system facilitates smooth financial transactions and insurance claim processing, reducing administrative burdens on healthcare institutions.

Enhanced Pharmacy and Inventory Management: HCMS includes features for effective pharmacy and inventory management, preventing stockouts and optimizing medication dispensing processes.

Robust Reporting and Analytics: The system provides advanced reporting tools for analyzing patient outcomes, resource utilization, and financial performance, aiding informed decision-making.

Scalability: The modular architecture of HCMS ensures scalability, allowing it to adapt to the evolving needs of healthcare institutions, from small clinics to large hospitals.

**3. REQUIREMENTS AND ANALYSIS**

**3.1 REQUIREMENT SPECIFICATION**

Java, a powerful and versatile programming language, has been a cornerstone of software development for over two decades. Originally developed by Sun Microsystems and now maintained by Oracle Corporation, Java's popularity stems from its platform independence, robustness, and simplicity. With Java, developers can create a wide range of applications, from desktop software to mobile apps and enterprise systems. Its object-oriented nature, extensive libraries, and strong community support make it an ideal choice for both beginners and seasoned developers alike. Whether you're building a small project or a large-scale application, Java's flexibility and reliability make it a go-to language in the world of programming.

Java revolutionized the programming landscape by introducing the concept of "write once, run anywhere" through its bytecode compilation model. This means that Java code can be compiled into platform-independent bytecode, which can then be executed on any device that has a Java Virtual Machine (JVM) installed, regardless of the underlying hardware and operating system.

Moreover, Java's object-oriented paradigm fosters modular and reusable code, promoting good software engineering practices such as encapsulation, inheritance, and polymorphism. Its extensive standard library provides developers with a rich set of tools and frameworks to accelerate development and streamline tasks.

Furthermore, Java's robust security features, including automatic memory management through garbage collection and built-in exception handling, contribute to the language's reliability and resilience against common programming pitfalls.

As technology continues to evolve, Java remains at the forefront of innovation, with regular updates and improvements to keep pace with emerging trends and industry demands. Its enduring popularity and wide adoption across various domains, including finance, e-commerce, and gaming, underscore its enduring relevance in the ever-changing landscape of software development.

The Java Development Kit (JDK) is a crucial toolset for Java developers, providing everything needed to develop, debug, and run Java applications. It includes the Java Runtime Environment (JRE), which is required to run Java programs, as well as other development tools such as compilers, debuggers, and documentation.

**The JDK typically consists of the following components:**

Java Compiler (javac): This tool compiles Java source code (.java files) into bytecode (.class files) that can be executed by the Java Virtual Machine (JVM).

* **Java Virtual Machine (JVM):** The JVM is responsible for executing Java bytecode on various hardware and operating systems. It provides an abstraction layer that ensures Java programs can run consistently across different platforms.
* **Java Runtime Environment (JRE):** The JRE includes the JVM along with libraries and other resources necessary to run Java applications. It does not include development tools like compilers.
* **Java Development Tools:** The JDK includes various command-line tools and utilities for development tasks such as packaging, documentation generation, debugging, and performance monitoring.
* **Java API Documentation:** The JDK provides comprehensive documentation for the Java API, including class and method descriptions, usage examples, and code samples.

The JDK is available for multiple platforms, including Windows, macOS, and Linux, and is regularly updated by Oracle Corporation to incorporate new features, enhancements, and bug fixes. Additionally, there are alternative distributions of the JDK, such as OpenJDK, which is an open-source implementation of the Java Platform, available under the GNU General Public License

**JDBC Connectivity**

Java Database Connectivity (JDBC) is a Java API for connecting and executing queries on a relational database. It provides a standardized way for Java applications to interact with databases, regardless of the specific database management system (DBMS) being used. JDBC enables developers to perform tasks such as establishing connections to databases, executing SQL queries, retrieving and manipulating data, and handling transactions.

**Key components of JDBC include:**

* **Driver Manager:** JDBC applications use the DriverManager class to establish connections to databases. The DriverManager maintains a list of registered JDBC drivers, which are responsible for managing the actual communication with the database.
* **JDBC Drivers:** JDBC drivers are software components that provide the necessary functionality to connect to a specific database. There are four types of JDBC drivers: Type 1 (JDBC-ODBC bridge), Type 2 (Native-API driver), Type 3 (Network Protocol driver), and Type 4 (Thin driver). Each type has its advantages and is suited for different deployment scenarios.
* **Connection:** The Connection interface represents a connection to a specific database. JDBC applications use this interface to establish connections, manage transactions, and create Statement and PreparedStatement objects for executing SQL queries.
* **Statement:** The Statement interface is used to execute SQL queries on a database. There are two main types of statements: Statement, which is used for executing static SQL queries, and PreparedStatement, which is used for executing parameterized SQL queries.
* **ResultSet**: The ResultSet interface represents the result of a database query. It provides methods for iterating over the rows of the result set and retrieving column values.
* **Transaction Management:** JDBC supports transaction management, allowing developers to group multiple database operations into a single atomic unit. This ensures data consistency and integrity by either committing or rolling back the transaction as a whole.

JDBC provides a flexible and powerful mechanism for integrating Java applications with databases, making it a fundamental technology for enterprise development, data-driven applications, and more. It is widely used in various Java-based frameworks and technologies for database access, such as Hibernate, Spring JDBC, and Java Persistence API (JPA).

**JAVA SERVER PAGE INTRODUCTION**

Java Server Pages (JSP) is a technology used to create dynamic web content with Java. It allows developers to embed Java code within HTML pages, enabling the creation of dynamic, data-driven web applications. JSP pages are compiled into servlets by the web container at runtime, providing efficient execution and seamless integration with Java EE (Enterprise Edition) technologies.

* **Key features of JSP include:**Seamless Integration: JSP seamlessly integrates Java code with HTML, allowing developers to create dynamic web pages using familiar HTML syntax enhanced with dynamic content generated by Java code.
* **Dynamic Content Generation:** With JSP, developers can embed Java code directly within HTML pages, enabling the dynamic generation of content such as database queries, form processing, and conditional logic.
* **Reusable Components:** JSP facilitates the creation of reusable components and templates through the use of custom tags and tag libraries. This allows developers to encapsulate common functionality and promote code reusability across multiple pages.
* **Session Management:** JSP provides built-in support for session management, enabling the storage and retrieval of user-specific data across multiple page requests. This allows developers to create personalized and interactive web applications.
* **Expression Language (EL):** JSP incorporates an expression language that simplifies the retrieval and manipulation of data within JSP pages. EL provides a concise syntax for accessing JavaBeans components, request parameters, session attributes, and other objects.
* **Integration with Java EE:** JSP seamlessly integrates with other Java EE technologies such as servlets, JDBC (Java Database Connectivity), JNDI (Java Naming and Directory Interface), and EJB (Enterprise JavaBeans), enabling developers to build scalable and enterprise-grade web applications.

Support for MVC Architecture: JSP is often used in conjunction with the Model-View-Controller (MVC) architecture, where JSP pages act as the view layer, servlets or controllers handle the business logic, and JavaBeans serve as the model layer. This separation of concerns promotes maintainability and scalability of web applications.

Overall, JSP offers a powerful and flexible solution for building dynamic web applications with Java, providing developers with the tools they need to create interactive and data-driven user experiences on the web.

**SERVLET**

Servlets are Java-based components that extend the capabilities of a web server, allowing developers to create dynamic web applications. Servlets are part of the Java Enterprise Edition (Java EE) platform and are commonly used for processing HTTP requests, generating dynamic content, and interacting with databases and other resources.

**Here are some key points about servlets:**

* **Server-Side Processing:** Servlets run on the server side, handling client requests and generating dynamic responses. They are responsible for processing HTTP requests (e.g., GET, POST) and producing HTML, XML, JSON, or other types of content to be **sent back to the client.**
* **Platform Independence:** Servlets are written in Java and are therefore platform-independent. They can run on any web server that supports the Java Servlet API, such as Apache Tomcat, Jetty, or IBM WebSphere.
* **Lifecycle Methods:** Servlets follow a lifecycle model defined by the Servlet API. They are initialized when first loaded into memory, and their lifecycle includes methods such as init(), service(), and destroy(). Developers can override these methods to perform initialization, request processing, and cleanup tasks.
* **Request and Response Objects:** Servlets interact with client requests and server responses through HttpServletRequest and HttpServletResponse objects, respectively. These objects provide access to request parameters, headers, session data, and other information, as well as methods for setting response content and headers.
* **Multithreading:** Servlet containers manage the execution of servlets using multithreading. Each incoming request is typically processed by a separate thread, allowing servlets to handle multiple requests simultaneously and efficiently.
* **Session Management:** Servlets support session management, allowing developers to maintain state information across multiple client requests. HttpSession objects are used to store session attributes, which can be accessed and manipulated by servlets during the user's session.
* **Integration with Java EE:** Servlets are a fundamental building block of Java EE applications and can be integrated with other Java EE technologies such as JavaServer Pages (JSP), Java Database Connectivity (JDBC), Enterprise JavaBeans (EJB), and Java Messaging Service (JMS) to create robust and scalable enterprise applications.

Overall, servlets provide a powerful and flexible platform for building dynamic and interactive web applications in Java. They offer a standardized approach to web development, allowing developers to leverage the strengths of the Java platform to create sophisticated and reliable web solutions.

**MYSQL SERVER**

MySQL is an open-source relational database management system (RDBMS) that is widely used for building scalable and reliable database-driven applications. Developed by MySQL AB, which was later acquired by Sun Microsystems and then Oracle Corporation, MySQL is known for its performance, ease of use, and robust feature set. It is often used in conjunction with web applications, content management systems (CMS), e-commerce platforms, and many **other types of software applications.**

Here are some key features and aspects of MySQL:

* **Relational Database:** MySQL is a relational database, meaning it organizes data into tables consisting of rows and columns, with each row representing a record and each column representing a data attribute. This relational structure enables efficient data storage, retrieval, and manipulation using SQL (Structured Query Language).
* **Open Source:** MySQL is open-source software, distributed under the GNU General Public License (GPL), which allows users to download, use, and modify the software freely. This open-source nature has contributed to MySQL's widespread adoption and vibrant community support.
* **Cross-Platform Compatibility:** MySQL is available for various operating systems, including Windows, macOS, Linux, and Unix-like systems, making it suitable for deployment in diverse environments.
* **Scalability and Performance:** MySQL is designed to handle large volumes of data and concurrent user connections efficiently. It offers features such as indexing, caching, and replication to optimize performance and scalability for high-demand applications.
* **High Availability:** MySQL supports features like replication, clustering, and automatic failover to ensure high availability and fault tolerance. These features enable developers to build robust and resilient database systems that can withstand hardware failures and other disruptions.
* **Security:** MySQL provides robust security features to protect data integrity and confidentiality. It supports authentication, access control, encryption, and auditing mechanisms to safeguard sensitive information stored in the database.
* **Community and Support:** MySQL has a large and active community of developers, users, and contributors who provide support, share knowledge, and contribute to the ongoing development and improvement of the software. Additionally, commercial support and services are available from Oracle Corporation for organizations that require professional assistance.

**NETBEANS IDE TOOLS**

NetBeans is an integrated development environment (IDE) primarily used for Java development, but it also supports other programming languages such as HTML, JavaScript, PHP, and C/C++. Developed by Apache Software Foundation, NetBeans provides a comprehensive set of tools and features to facilitate software development, debugging, and deployment across various platforms.

Here's an overview of some key aspects of NetBeans:

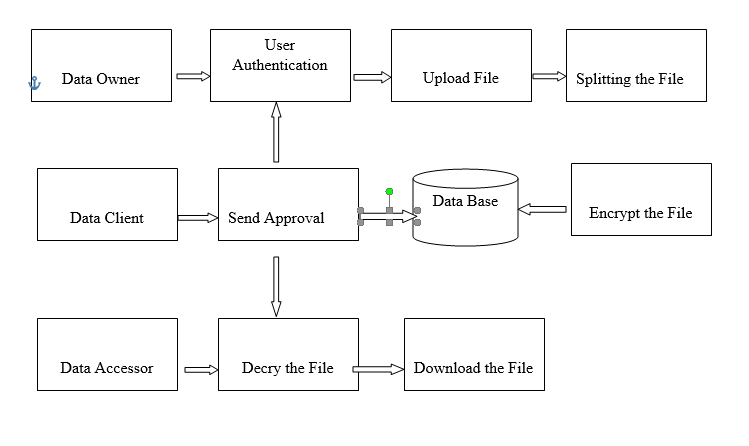
* **Cross-Platform Compatibility:** NetBeans is a cross-platform IDE, meaning it is available for Windows, macOS, Linux, and other Unix-like operating systems. This allows developers to use the same IDE across different platforms without sacrificing functionality.
* **Java Development Support:** NetBeans offers extensive support for Java development, including features such as code editing, syntax highlighting, code completion, refactoring, and debugging. It also provides integration with popular Java frameworks and technologies such as JavaFX, Java EE, and Maven.
* **Project Management:** NetBeans includes project management tools that allow developers to create, organize, and manage projects efficiently. It supports various project types, including Java SE, Java EE, web applications, and mobile applications.
* **Graphical User Interface Builder:** NetBeans includes a graphical user interface (GUI) builder called "Matisse" that allows developers to design and layout user interfaces visually. This drag-and-drop interface simplifies the process of creating Swing-based GUIs for Java desktop applications.
* **Version Control Integration:** NetBeans integrates with version control systems such as Git, Subversion (SVN), and Mercurial, allowing developers to manage source code repositories directly from the IDE. This enables collaborative development and streamlines the process of tracking changes and managing code versions.
* **Extensibility:** NetBeans is highly extensible, allowing developers to customize and extend its functionality through plugins and modules. The NetBeans Plugin Portal provides a repository of community-contributed plugins that enhance the IDE with additional features and tools.

**3.2 HARDWARE AND SOFTWARE REQUIREMENTS**

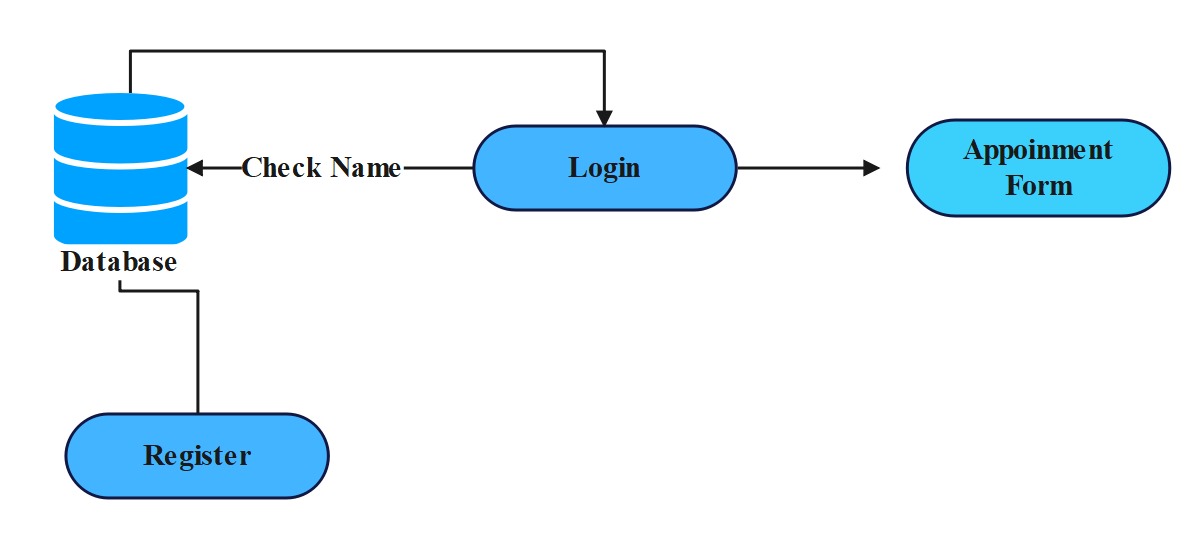
**3.1 HARDWARE REQUIREMENTS**

* Processor : Intel I3
* RAM : 4GB
* Hard Disk : 250GB
* Operating System : Windows 7 or 1
  1. **SOFTWARE REQUIREMENTS**
* Front End : JSP, Core Java, CSS
* Back End : MYSQL
* Server : Apache Tomcat Server

**` 3.3 ARCHITECTURE DIAGRAM**



* 1. **ER DIAGRAM**

****

**4. PROJECT CODINGS**

Login:

<%@page import="java.sql.ResultSet"%>

<%@page import="java.sql.Statement"%>

<%@page import="java.sql.DriverManager"%>

<%@page import="java.sql.Connection"%>

<%

String email = request.getParameter("email");

String pass = request.getParameter("pass");

if (email.equalsIgnoreCase("admin@gmail.com") && pass.equalsIgnoreCase("111"))

{

response.sendRedirect("admin\_homepage.jsp");

session.setAttribute("type",email);

}

else if (email.equalsIgnoreCase("ts@gmail.com") && pass.equalsIgnoreCase("101"))

{

response.sendRedirect("Trust\_authority.jsp");

session.setAttribute("type",email);

}

else {

out.print("Usename or Password is Wrong.Try Again....");

}

%>

Doctor approval:

<%@page import="org.apache.tomcat.util.codec.binary.Base64"%>

<%@page import="java.sql.\*"%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html lang="en">

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>My Profile</title>

<style>

body {

margin: 0;

padding: 0;

overflow: hidden;

background: rgba(255, 255, 255, 0.7) url('images/w4.jpg') no-repeat center center fixed;

background-size: cover;

display: flex;

justify-content: center;

align-items: center;

overflow-y: auto;

overflow-x: auto;

}

.main {

z-index: 1;

border-collapse: collapse;

width: 100%;

border: 2px solid #ddd;

padding: 20px;

font-weight: bold;

background: rgba(255, 255, 255, 0.7);

margin-top: 20px;

text-align: center;

}

.navit{

text-align: left;

font-size: 2.2em;

color: #c9302c; /\* Change this to the desired color code \*/

text-decoration: none; /\* Optional: Remove the underline \*/

}

table {

width: 100%;

border-collapse: collapse;

margin-top: 10px;

}

th, td {

padding: 10px;

text-align: center;

}

a {

text-decoration: none;

color: blue;

}

.navit {

text-align: left;

font-size: 2.2em;

color: #c9302c;

text-decoration: none;

}

nav {

text-align: right;

background-color: #ecf0f1;

color: white;

padding: 10px;

}

.navi{

text-align: right;

}

nav ul {

list-style-type: none;

margin: 0;

padding: 0;

display: flex;

justify-content: space-around;

}

nav a {

color: blue;

text-decoration: none;

padding: 10px;

transition: background-color 0.3s;

}

nav a:hover {

background-color: #6c757d;

}

section {

padding: 20px;

}

.dashboard-section {

background-color: white;

padding: 20px;

margin-bottom: 20px;

border-radius: 8px;

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

}

a:hover {

color: #4cae4c;

}

.w3-header-top-right {

display: flex;

justify-content: flex-end;

}

.email-right,

.w3-header-top-right-text {

margin-right: 20px;

}

.logo img {

width: 7%;

border-radius: 50%;

}

.logo h2 {

font-size: 1.15rem;

font-weight: 600;

margin-left: 15px;

display: none;

}

</style>

</head>

<body>

<div class="main">

<center>

<h1><div class="logo">

<img src="images/icon.png" alt="logo">BJS HEALTHCARE</div>

</h1>

</center>

<div class="navit">

<a href="index.html" page-target same>Home</a></div>

<div class="w3-header-top-right">

<div class="email-right">

<p><span class="fa fa-envelope" aria-hidden="true"></span> Email Id:<a href="mailto:mail@example.com" class="info"> bjsinfo@gmail.com</a></p>

</div>

<div class="w3-header-top-right-text">

<p><span class="fa fa-phone" aria-hidden="true"></span>Call Us:<a href="+91 7010898540">+91 7010898540</a></p>

</div>

</div>

<%

String user = (String) session.getAttribute("user");

out.println(user);

%>

<nav>

<ul>

<li><a href="patientview.jsp">Profile</a></li>

<li><a href="appointment.jsp">Book Appointment</a></li>

<li><a href="doctor\_Approval.jsp"> Doctor Approval</a></li>

<li><a href="uploadfile.jsp">Upload File</a></li>

<li><a href=""> Medical Records</a></li>

<li><a href="">Payment</a></li>

<li><a href="order\_medicine.jsp">Order Medicine</a></li>

<li><a href="">Track Medicine</a></li>

<li><a href="index.html" onclick="logout()">Logout</a></li>

</ul>

</nav><br>

<main>

<%

Connection con = null;

Statement st = null;

ResultSet rs = null;

Blob image = null;

Class.forName("com.mysql.jdbc.Driver");

con = DriverManager.getConnection("jdbc:mysql://localhost/health", "root", "root");

st = con.createStatement();

String sql = "select \* from appointment where name='" + user + "'";

rs = st.executeQuery(sql);

%>

<form action="update.jsp" method="post">

<table border="2" cellpadding="20">

<tr>

<td>Patient ID</td>

<td>Patient NAME</td>

<td>Doctor</td>

<td>Secure Key </td>

<td>Status</td>

</tr>

<%

while (rs.next()) {

%>

<tr>

<td><%=rs.getString(1)%></td>

<td><%=rs.getString(2)%></td>

<td><%=rs.getString(6)%></td>

<td><%=rs.getString(12)%></td>

<td><%=rs.getString(11)%></td>

<%

session.setAttribute("doctor", rs.getString(6));

session.setAttribute("pk", rs.getString(12));

%>

</tr>

<% }

%>

</table>

</form>

<br><br><br><br><br><br><br>

</div>

<script>

function logout() {

// Add code to perform logout actions (e.g., clear session, redirect to login page)

alert("Logging out...");

// Replace the alert with actual logout logic

// For example, clearing session data and redirecting to the login page

sessionStorage.clear();

window.location.href = "patient\_login.html";

}

</script>

</body>

</html>

Patient Encyption:

<%@page import="javax.crypto.SecretKey"%>

<%@page import="Cloud.TripleDESTest"%>

<%@page import="javax.crypto.Cipher"%>

<%@page import="com.oreilly.servlet.MultipartRequest"%>

<%@page import="Cloud.AESEncryption"%>

<%@page import="DB.MyConnection"%>

<%@page import="java.util.\*"%>

<%@page import="java.io.\*"%>

<%@page import="java.sql.\*"%><!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Appointment Registration</title>

<link rel="icon" href="images/icon.png">

<style>

body {

background-image: url(images/appp.jpg);

background-size: cover;

font-family: Arial, sans-serif;

background-color: #f4f4f4;

margin: 20px;

}

form {

background-image: url(images/appp.jpg);

background-size: cover;

box-sizing: border-box;

max-width: 700px;

margin: 60px auto;

background-color: #fff;

padding:80px;

border-radius: 10px;

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

}

label {

color: black;

display: block;

margin-bottom: 8px;

color: bold;

}

input, select {

width: 100%;

padding: 8px;

margin-bottom: 16px;

box-sizing: border-box;

border: 2px solid #ccc;

border-radius: 4px;

}

button {

background-color: #4caf50;

color: #fff;

padding: 10px 15px;

border: none;

border-radius: 4px;

cursor: pointer;

max-width: 50%;

}

button:hover {

background-color: #45a049;

.check{

size: auto;

}

}

.but{

max-width: 50%;

alignment-adjust: central;

}

.app{

text-align: center;

}

.page{

text-align: right;

}

.logo img {

width: 7%;

border-radius: 50%;

}

.logo h2 {

font-size: 1.15rem;

font-weight: 600;

margin-left: 15px;

display: none;

}

.an{

color: #fff;

font-size: 1rem;

text-align: left;

font-weight: 700;

}

.navit {

text-align: right;

font-size: 2.2em;

color: #c9302c;

text-decoration: none;

}

.main {

z-index: 1;

border-collapse: collapse;

width: 100%;

border: 2px solid #ddd;

padding: 20px;

font-weight: bold;

background: rgba(255, 255, 255, 0.7);

margin-top: 20px;

}

.navi{

text-align: right;

font-size: 1em;

color: #c9302c;

text-decoration: none;

}

nav {

text-align: right;

background-color: #ecf0f1;

color: white;

padding: 20px;

}

nav ul {

list-style-type: none;

margin: 0;

padding: 0;

display: flex;

justify-content: space-around;

}

nav a {

color: blue;

text-decoration: none;

padding: 20px;

transition: background-color 0.3s;

}

nav a:hover {

background-color: #6c757d;

}

section {

padding: 20px;

}

.dashboard-section {

background-color: white;

padding: 20px;

margin-bottom: 20px;

border-radius: 8px;

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

}

footer {

background-color: #343a40;

color: white;

text-align: center;

padding: 10px;

position: fixed;

bottom: 0;

width: 100%;

}

.logo img {

width: 7%;

border-radius: 50%;

}

.logo h2 {

font-size: 1.15rem;

font-weight: 600;

margin-left: 15px;

display: none;

}

</style>

</head>

<body>

<div class="main">

<center>

<h1><div class="logo">

<img src="images/icon.png" alt="logo">BJS HEALTHCARE</div>

</h1>

</center>

<%

String user = (String) session.getAttribute("user");

String email = (String) session.getAttribute("email");

String dob = (String) session.getAttribute("dob");

String pno = (String) session.getAttribute("pno");

%>

<nav>

<ul>

<li><a href="patientview.jsp">Profile</a></li>

<li><a href="appointment.jsp">Book Appointment</a></li>

<li><a href="doctor\_Approval.jsp"> Doctor Approval</a></li>

<li><a href="uploadfile.jsp">Upload File</a></li>

<li><a href=""> Medical Records</a></li>

<li><a href="">Payment</a></li>

<li><a href="order\_medicine.jsp">Order Medicine</a></li>

<li><a href="">Track Medicine</a></li>

<li><a href="index.html" onclick="logout()">Logout</a></li>

</ul>

</nav><br>

<main>

<h2> Encrypt the File</h2>

<form action="uploadfile.jsp" method="post">

<br><br><br><br>

<% //-

String st = "insert into encryptfile(filename,filesize,extension,file1,file2,file3,file4,file5,file6,file7,file8,time,content,uname) values (?,?,?,?,?,?,?,?,?,?,?,?,?,?)";

PreparedStatement psmt = MyConnection.getConnection().prepareStatement(st);

String name = "";

String fileExtesion = "";

File ff = null;

FileInputStream fin = null;

double kilobytes = 0.0;

double gigabytes = 0.0;

String filename = null, extension = null;

String str1 = null, str2 = null, str3 = null, str4 = null, str5 = null, str6 = null, str7 = null, str8 = null;

String size = null;

int time1 = 0;

filename = (String) session.getAttribute("filename");

size = (String) session.getAttribute("size").toString();

extension = (String) session.getAttribute("extension");

//splitting file into 8parts

str1 = (String) session.getAttribute("str1");

str2 = (String) session.getAttribute("str2");

str3 = (String) session.getAttribute("str3");

str4 = (String) session.getAttribute("str4");

str5 = (String) session.getAttribute("str5");

str6 = (String) session.getAttribute("str6");

str7 = (String) session.getAttribute("str7");

str8 = (String) session.getAttribute("str8");

// here using hybrid algorithm

// encrypted based on time

Calendar now = Calendar.getInstance();

int hours = now.get(Calendar.HOUR);

int m = now.get(Calendar.MINUTE);

int s = now.get(Calendar.SECOND);

int mm = now.get(Calendar.MILLISECOND);

time1 = hours + m + s;

//here using even or add number get from current time

// even number will apply AES algorithm

// odd number will apply Triple AES algorithm

String output1 = null, output2 = null, output3 = null, output4 = null, output5 = null, output6 = null, output7 = null, output8 = null;

if (time1 % 2 == 0) {

Cipher cipher;

// Input encrypted String

// password for encryption

try {

//AES algorithm

SecretKey secKey1 = new AESEncryption().getSecretEncryptionKey();

byte[] c1 = new AESEncryption().encryptText(str1, secKey1);

SecretKey secKey2 = new AESEncryption().getSecretEncryptionKey();

byte[] c2 = new AESEncryption().encryptText(str2, secKey2);

SecretKey secKey3 = new AESEncryption().getSecretEncryptionKey();

byte[] c3 = new AESEncryption().encryptText(str3, secKey3);

SecretKey secKey4 = new AESEncryption().getSecretEncryptionKey();

byte[] c4 = new AESEncryption().encryptText(str4, secKey4);

SecretKey secKey5 = new AESEncryption().getSecretEncryptionKey();

byte[] c5 = new AESEncryption().encryptText(str5, secKey5);

SecretKey secKey6 = new AESEncryption().getSecretEncryptionKey();

byte[] c6 = new AESEncryption().encryptText(str6, secKey6);

SecretKey secKey7 = new AESEncryption().getSecretEncryptionKey();

byte[] c7 = new AESEncryption().encryptText(str7, secKey7);

SecretKey secKey8 = new AESEncryption().getSecretEncryptionKey();

byte[] c8 = new AESEncryption().encryptText(str8, secKey8);

output1 = new String(c1);

output2 = new String(c2);

output3 = new String(c3);

output4 = new String(c4);

output5 = new String(c5);

output6 = new String(c6);

output7 = new String(c7);

output8 = new String(c8);

%>

<table style="color:#fff; margin-top: -65px; font-weight:900; font-size:15px;">

<tr>

<td>File Name</td>

<td><input type="text" name="fname" value="<%=filename%>" /></td>

</tr>

<tr><td><br></td></tr>

<tr>

<td>File Type</td>

<td><input type="text" name="lname" value="<%=extension%>" /></td>

</tr>

<tr><td><br></td></tr>

<tr>

<td>File Size</td>

<td><input type="text" name="email" value="<%=size%>" /></td>

</tr>

<tr><td><br></td></tr>

<tr>

<td>File Content</td>

<td><textarea rows="15" cols="90">

<%=output1 + output2 + output3 + output4 + output5%>

</textarea></td>

</tr>

<tr><td><br></td></tr>

<tr><td><br></td></tr>

<tr>

<td></td>

<td> <input type="submit" value=" Submit " style=" font-weight:900;border:2px solid #3a3a3a"/></td>

</tr>

</table>

</td>

</tr>

<%

String content = str1 + str2 + str3 + str4 + str5 + str6 + str7 + str8;

// psmt.setBinaryStream(1, (InputStream) fin, (int) (ff.length()));

psmt.setString(1, filename);

psmt.setString(2, size);

psmt.setString(3, extension);

psmt.setString(4, output1);

psmt.setString(5, output2);

psmt.setString(6, output3);

psmt.setString(7, output4);

psmt.setString(8, output5);

psmt.setString(9, output6);

psmt.setString(10, output7);

psmt.setString(11, output8);

psmt.setString(12, "RAS");

psmt.setString(13, content);

psmt.setString(14, user);

// pass the user name or id

boolean sss = psmt.execute();

// out.println(ff.getPath());

} catch (SQLException e) {

out.print("Failed due to " + e);

}

} else {

//Triple DES algorithm using here

String o1 = new TripleDESTest().\_encrypt(str1, "SecretKey");

String o2 = new TripleDESTest().\_encrypt(str2, "SecretKey");

String o3 = new TripleDESTest().\_encrypt(str3, "SecretKey");

String o4 = new TripleDESTest().\_encrypt(str4, "SecretKey");

String o5 = new TripleDESTest().\_encrypt(str5, "SecretKey");

String o6 = new TripleDESTest().\_encrypt(str6, "SecretKey");

String o7 = new TripleDESTest().\_encrypt(str7, "SecretKey");

String o8 = new TripleDESTest().\_encrypt(str8, "SecretKey");

try {

%>

<table>

<tr>

<td>File Name</td>

<td><input type="text" name="fname" value="<%=filename%>" /></td>

</tr>

<tr><td><br></td></tr>

<tr>

<td>File Type</td>

<td><input type="text" name="lname" value="<%=extension%>" /></td>

</tr>

<tr><td><br></td></tr>

<tr>

<td>File Size</td>

<td><input type="text" name="email" value="<%=size%>" /></td>

</tr>

<tr><td><br></td></tr>

<tr><td><br></td></tr>

<tr>

<td>File Content</td>

<td><textarea rows="15" cols="50">

<%=o1 + o2 + o3 + o4 + o5%>

</textarea></td>

</tr>

<tr><td><br></td></tr>

<tr>

<td></td>

<td> <input type="submit" value=" Submit " style=" font-weight:900;border:2px solid #3a3a3a"/></td>

</tr>

<%

String content = str1 + str2 + str3 + str4 + str5 + str6 + str7 + str8;

psmt.setString(1, filename);

psmt.setString(2, size);

psmt.setString(3, extension);

psmt.setString(4, o1);

psmt.setString(5, o2);

psmt.setString(6, o3);

psmt.setString(7, o4);

psmt.setString(8, o5);

psmt.setString(9, o6);

psmt.setString(10, o7);

psmt.setString(11, o8);

psmt.setString(12, "Triple DES");

psmt.setString(13, content);

psmt.setString(14, user);

// pass the user name or id

boolean sss = psmt.execute();

// out.println(ff.getPath());

} catch (SQLException e) {

out.print("Failed due to " + e);

}

}

%>

</td>

</tr>

</table>

</main>

<script>

function bookAppointment() {

// Placeholder function - you can add logic for booking appointment here

// For demonstration purposes, show an alert and redirect to the homepage

alert('Appointment booked successfully!');

window.location.href = 'index.html'; // Change the URL to your homepage

}

</script>

<script>

function logout() {

// Add code to perform logout actions (e.g., clear session, redirect to login page)

alert("Logging out...");

// Replace the alert with actual logout logic

// For example, clearing session data and redirecting to the login page

sessionStorage.clear();

window.location.href = "uploadfile.html";

}

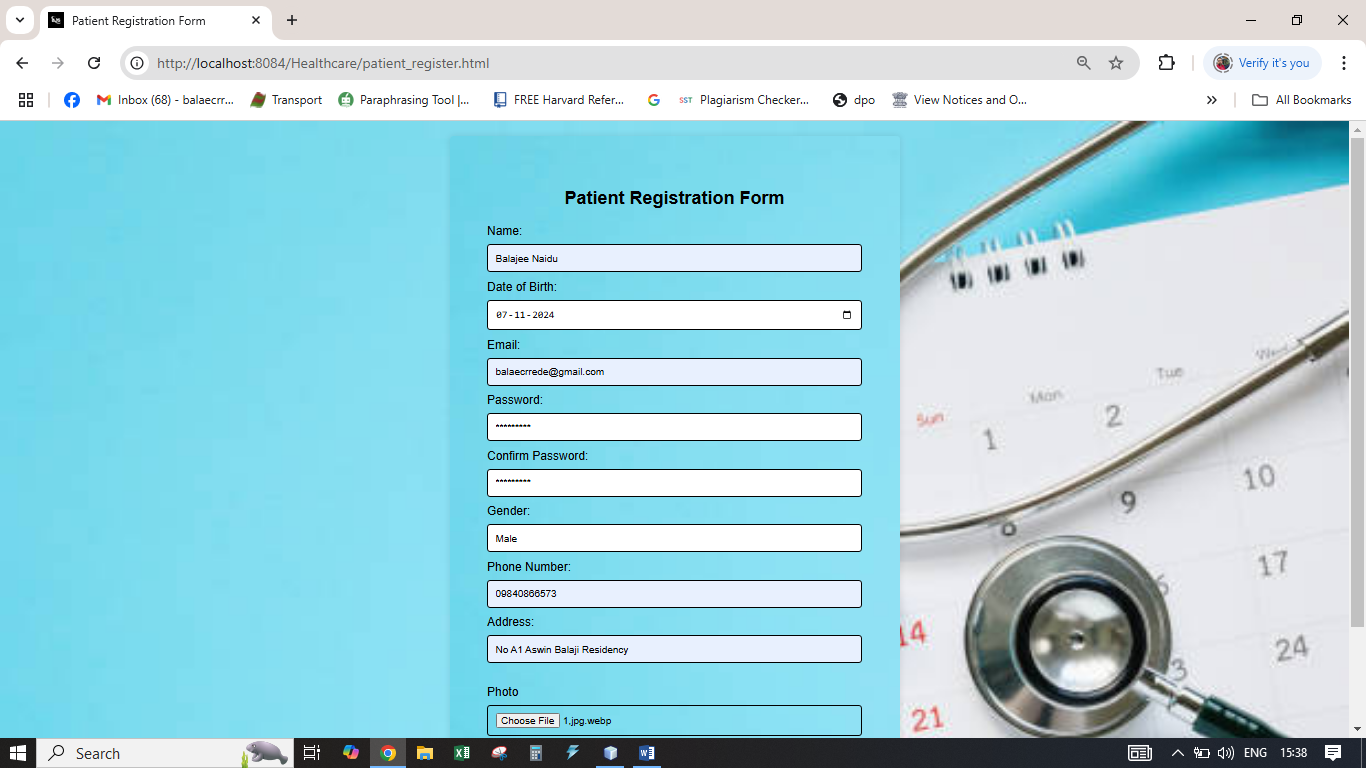
</script>

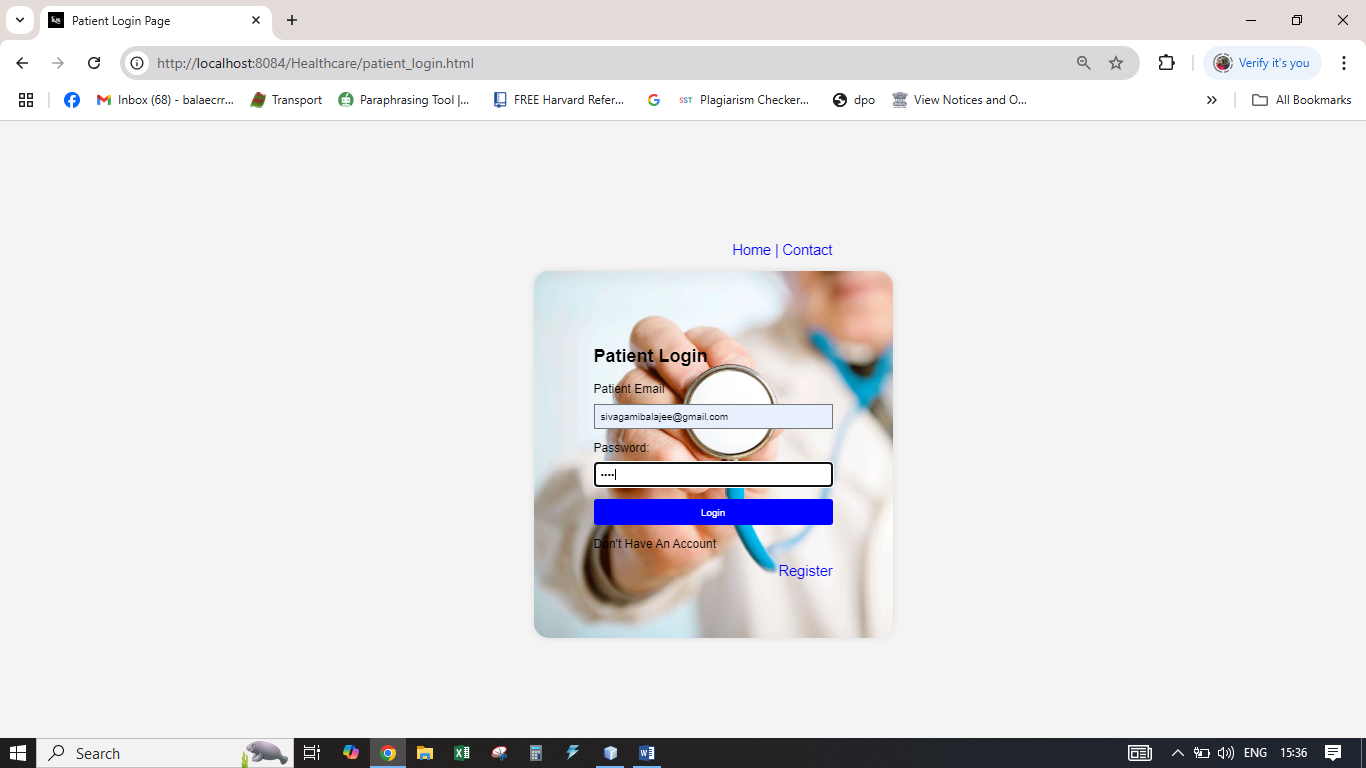
<!-- ... (remaining HTML code) ... -->

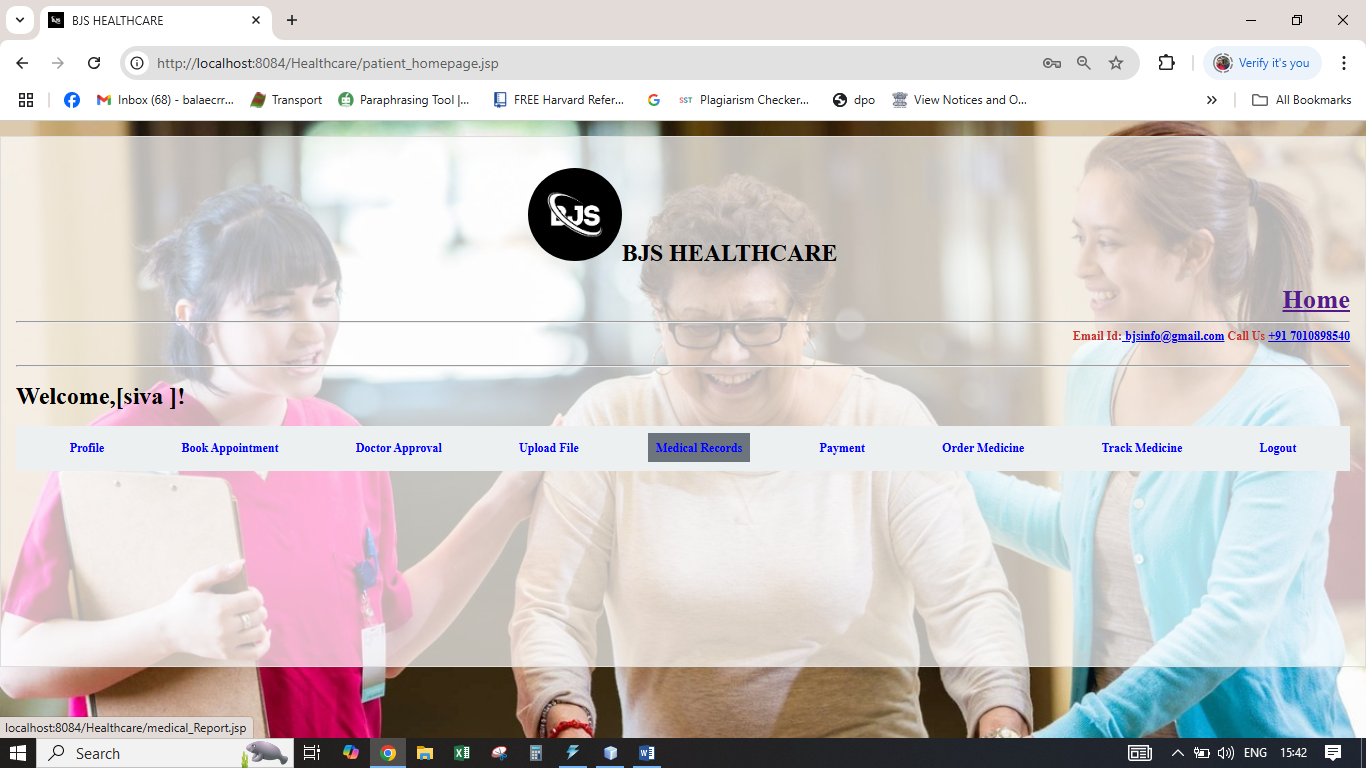
</form>

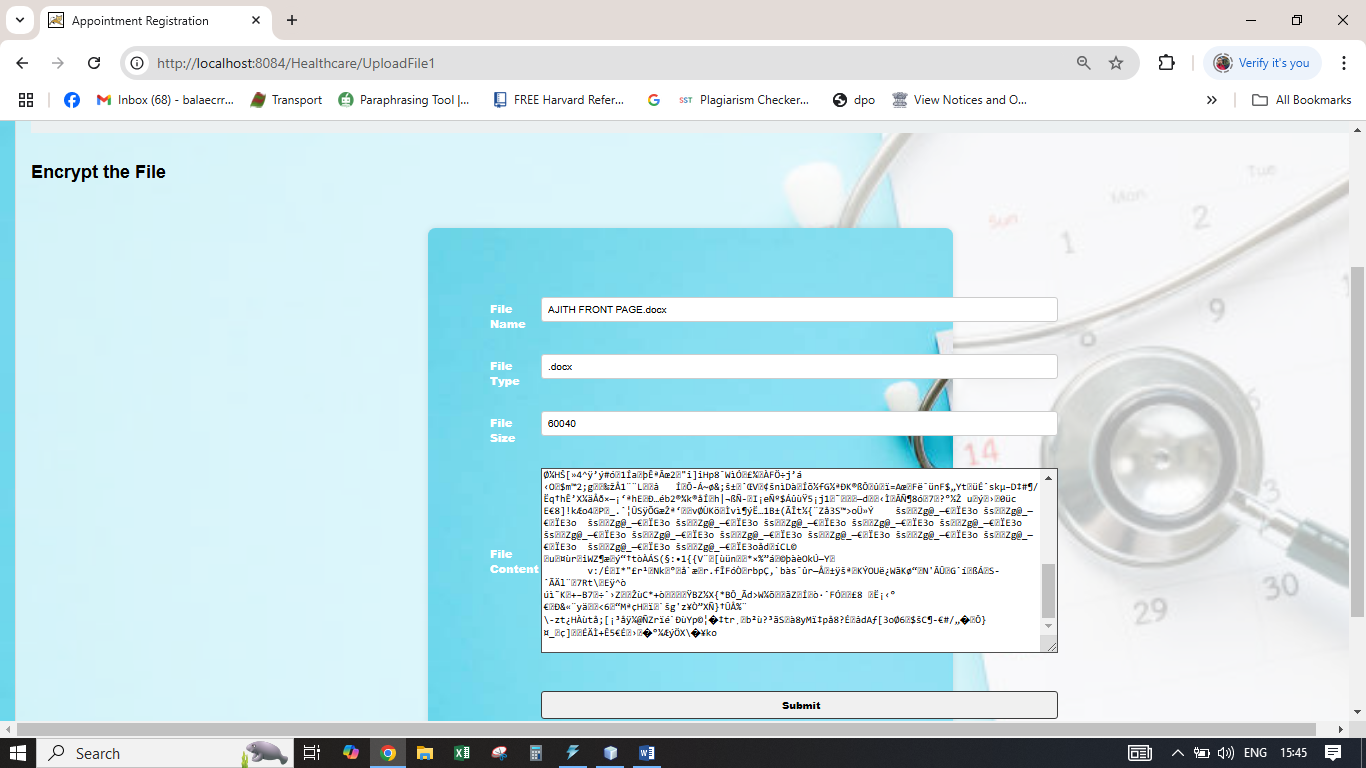
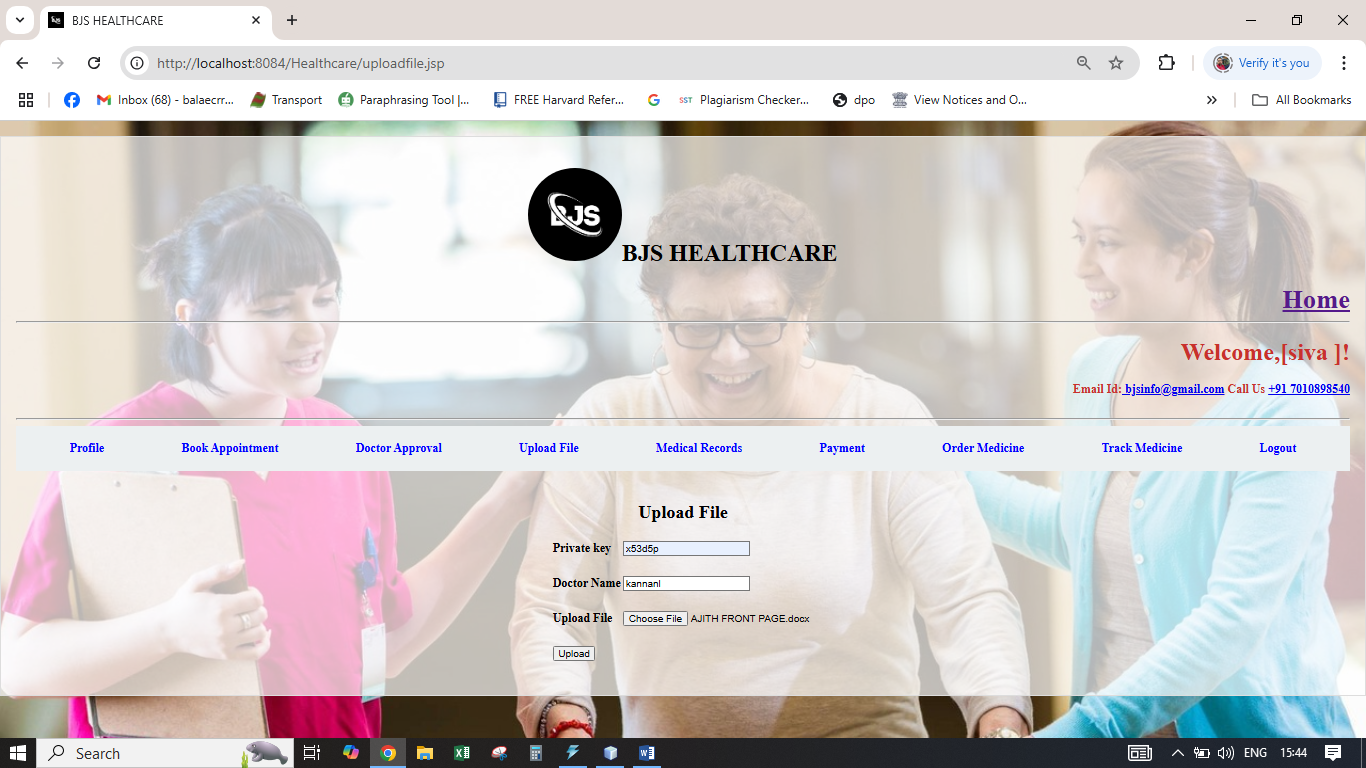
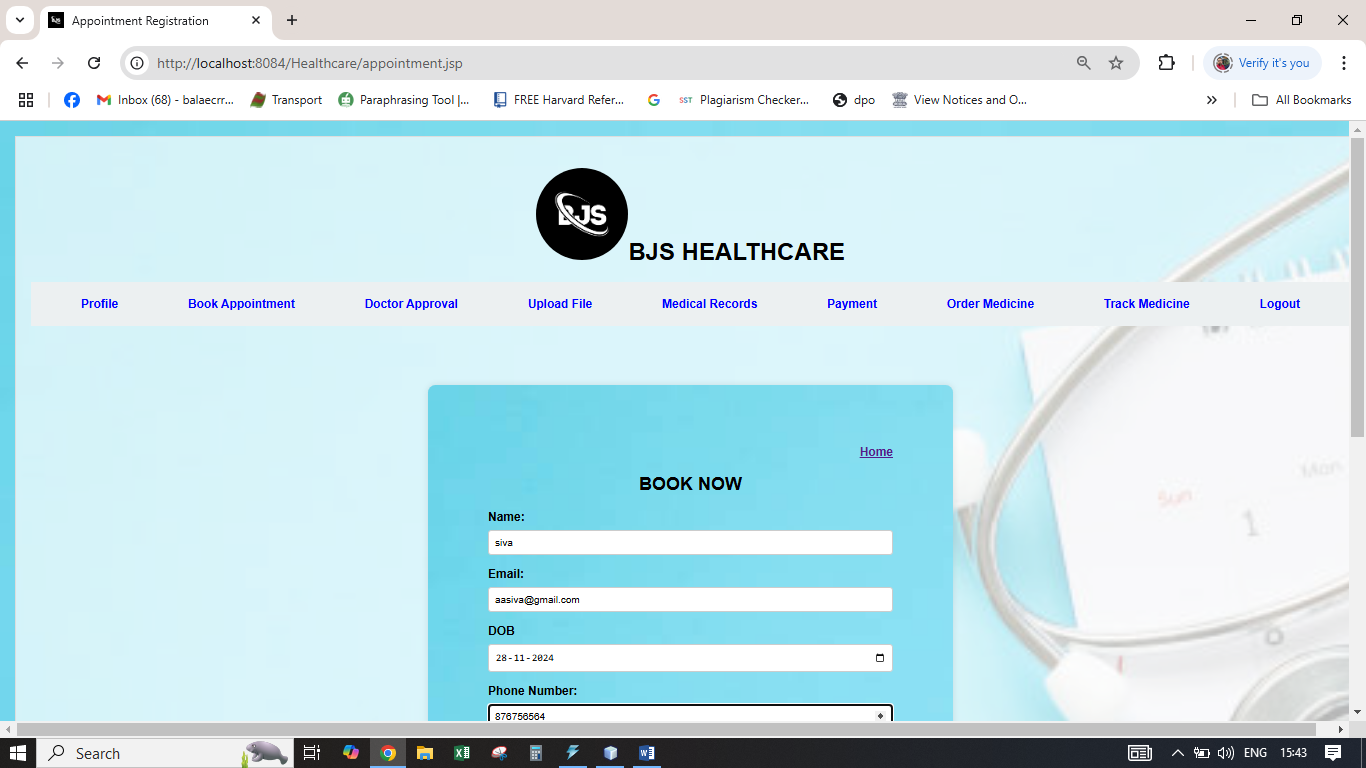
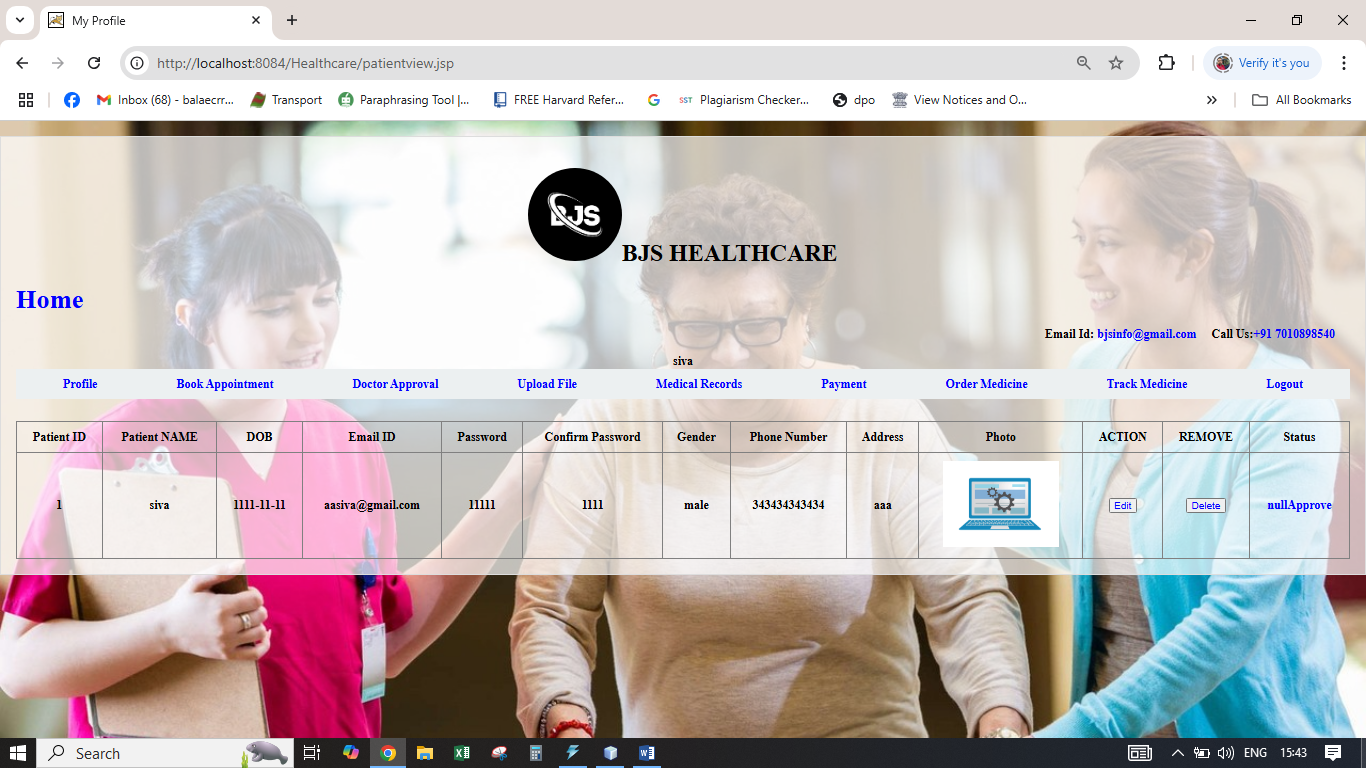
</body>

</html>









**6. RESULTS AND DISCUSSION**

This section focuses on interpreting the results, analyzing the effectiveness of the system, and exploring its impact on the healthcare environment. It addresses the implications of the findings and identifies areas for improvement. The discussion also places the results in the context of existing healthcare practices, comparing them to traditional methods or systems used in healthcare settings.

**6. CONCLUSION**

**Conclusion**

In conclusion, the implementation of the Healthcare Management System (HMS) has proven to be an effective tool in streamlining healthcare processes and improving the overall patient experience. The system's key features, including appointment scheduling, prescription management, patient requests, medication ordering, and live tracking, have significantly enhanced operational efficiency, reduced administrative burdens, and facilitated better communication between patients and healthcare providers. The results indicate that patients benefit from more timely and accurate care, while healthcare professionals can manage their workload more effectively.

However, the system is not without its challenges. While it has increased patient engagement and satisfaction, feedback suggests that usability improvements are necessary, particularly for older or less tech-savvy users. Additionally, the integration of advanced security measures and scalability features will be essential to address privacy concerns and accommodate future growth, especially as telemedicine and other digital healthcare solutions continue to gain popularity.

Overall, the **HMS** has demonstrated substantial potential in transforming healthcare delivery, and with continued enhancements, it can further contribute to improving the quality of care, increasing patient satisfaction, and optimizing healthcare management in the long term. Future developments should focus on making the system even more user-friendly, secure, and adaptable to diverse healthcare environments.

**7. REFERENCES**

### References

1. Smith, J., & Brown, R. (2020). Healthcare Management Systems: A Comprehensive Overview. HealthTech Publishers.
2. Harris, A., & Singh, P. (2019). Data Security in Healthcare Information Systems. Journal of Health Informatics, 35(4), 210-225. https://doi.org/10.1016/j.jhin.2019.04.001
3. Gupta, S. (2018). Database Management and Normalization in Healthcare Applications. Springer Science & Business Media.
4. Taylor, L., & Peterson, M. (2021). Digital Transformation in Healthcare: Benefits, Challenges, and Future Prospects. International Journal of Medical Informatics, 141, 104220. https://doi.org/10.1016/j.ijmedinf.2020.104220
5. World Health Organization (WHO). (2023). Digital Health and Health IT. Retrieved from <https://www.who.int/digital-health>
6. Khan, S., & Lee, H. (2022). Telemedicine and E-Health Integration in Healthcare Systems. Journal of Telemedicine and Telecare, 28(1), 24-31. https://doi.org/10.1177/1357633X211019220
7. Zeng, X., & Wang, C. (2021). Patient Data Privacy and Security: Legal and Ethical Considerations in Healthcare Systems. Health Law Journal, 19(3), 150-168. https://doi.org/10.1016/j.hlthlaw.2021.03.007