**CLOUD AND BLOCKCHAIN POWERED HOSPITAL MANAGEMENT: A NEW ERA OF HEALTHCARE SOLUTIONS**

**A MINI PROJECT REPORT**

***Submitted by***

**PRARTHANAH.G (211422104345)**

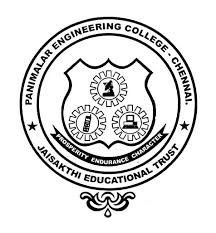
***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

***in***

**COMPUTER SCIENCE AND ENGINEERING**



**PANIMALAR ENGINEERING COLLEGE**

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

**NOVEMBER 2024**

**ACKNOWLEDGEMENT**

We express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We would like to extend our heartfelt and sincere thanks to our Directors

**Tmt. C. VIJAYARAJESWARI , Dr. C . SAKTHIKUMAR , M.E. , Ph.D.,** and **Tmt. SARANYASREE SAKTHIKUMAR B.E.,M.B.A.,** for providing us with the necessary facilities for completion of this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.,** for his timely concern and encouragement provided to us throughout the course.

We thank the HOD of CSE Department , **Dr. L. JABA SHEELA, M.E.,Ph.D.** for the support extended throughout the project.

We would like to thank my Project Guide **Dr.** **M.S. VINMATHI, M.E.,Ph.D.,** and all the faculty members of the Department of CSE for their advice and suggestions for the successful completion of the project.

**PRARTHANAH.G (211422104345)**

**Organizational History and Structure**

Panimalar Engineering College, a Christian Minority Institution of Higher Education governed by JAISAKTHI Educational Trust aims at imparting quality engineering education for the aspiring youth. The College is accredited by National Board of Accreditation (NBA), New Delhi, approved by All India Council for Technical Education (AICTE) and recognized by UGC with 12(B) & 2(f) status. The college is located near Poonamallee, Chennai, and is well connected by road covering, Chennai, Kancheepuram and Thiruvallur districts. The Trust started Panimalar Engineering College in the year 2000 in accordance with the general policy of the Government of Tamil Nadu. The policy emphasizes to give high priority to meet the demand for trained engineers for various industrial and development projects in the state of Tamil Nadu and the rest of India.

This sprawling campus provides a holistic education in an ambience that makes no compromise on discipline, dedication and commitment. It strives to inculcate the spirit of learning in the campus offering state of art facilities to the students. The college ensures that the students, who pass out of the college, turn out to be academically brilliant, morally upright and empowered individuals.

**Vision**

To transform the budding engineers into academically excellent, highly intellectual and self-disciplined engineering graduates to mould them as good citizens with the spirit of integrity and morality that would cater to the needs of our nation.

**Mission**

To impart quality education with high standards of excellence in engineering and technology, to provide an excellent infrastructure in a serene and conductive atmosphere that would motivate the students in their pursuit of knowledge in the field of engineer and technology.

TABLE OF CONTENTS

| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
| --- | --- | --- |
| **1.** | **ABSTRACT** | 1 |
| **2.** | **PROBLEM STATEMENT** | 2 |
| **3.** | **INTRODUCTION** | 3 |
| **4.** | **EXISTING METHODS** | 4 |
|  | 4.1 Disadvantages |  |
| **5.** | **PROPOSED WORK** | 5 |
|  | 5.1 Advantages |  |
| **6.** | **SYSTEM REQUIREMENTS** | 6 |
|  | 6.1 Hardware requirements |  |
|  | 6.2 Software requirements |  |
| **7.** | **SYSTEM ARCHITECTURE** | 7 |
|  | 7.1 Architecture overview |  |
|  | 7.2 Description of the modules |  |
| **8.** | **SYSTEM DESIGN** | 11 |
|  | 8.1 Use case diagram |  |
|  | 8.2 Class diagram |  |
|  | 8.3 Sequence diagram |  |
|  | 8.4 Collaboration diagram |  |
| **9.** | 8.5 Deployment diagram  8.6 Data flow diagram  8.7 Entity relationship diagram  **SYSTEM IMPLEMENTATION** | 20 |
|  | 9.1 Client-side coding |  |
|  | 9.2 Server-side coding |  |
| **10.** | **SYSTEM TESTING** | 34 |
|  | 10.1 Test Cases  10.2 Performance Analysis |  |
| **11.** | **CONCLUSION** | 40 |
|  | 11.1 Future Enhancements |  |
| **12.** | **APPENDICES** | 44 |
|  | 12.1 Sample Screenshots |  |
| **13.** | **REFERENCES** | 47 |
|  |  |  |

**1. ABSTRACT**

In today’s fast-evolving healthcare landscape, the integration of advanced technologies is crucial for enhancing operational efficiency, data security and patient care. This project aims to develop a hospital management system leveraging the power of cloud computing and blockchain technology.

By utilizing cloud computing, the system ensures that healthcare providers have better access to data, facilitating better decision-making and improved patient outcomes. The integration of blockchain technology ensures data integrity, enhances transparency and protects against unauthorized access and data breaches. To safeguard sensitive patient information, the platform utilizes the Advanced Encryption Standard (AES) algorithm for robust data encryption.

Through this innovative approach, the project aims to revolutionize hospital management by combining the benefits of cloud computing and blockchain technology, ultimately leading to a more secure, efficient and patient-centric healthcare system.

The new hospital website enhances digital healthcare by providing an intuitive interface for easy access to appointments, medical information, and patient resources. Designed to streamline interactions and improve user experience, this

platform ensures seamless access to essential services. The initiative marks a significant step in modernizing the hospital’s digital presence and improving patient engagement and satisfaction.

**2.PROBLEM STATEMENT**

Hospitals today face significant challenges in managing patient data, ensuring operational efficiency, and maintaining security and compliance. These existing problems hinder effective care delivery and operational effectiveness, underscoring the need for innovative solutions.

This can lead to a number of problems, including:

1. Traditional hospital systems struggle with protecting patient data from unauthorized access and breaches, creating a need for enhanced security through cloud and blockchain technology.
2. Patients often lack control over their health data and consent management, creating a need for systems that offer better control and visibility using cloud and blockchain technologies.
3. High costs associated with traditional hospital management systems require a more cost-effective approach, which can be achieved through cloud and blockchain technology.
4. Ensuring the accuracy and immutability of patient data is crucial, and blockchain technology offers a reliable solution for maintaining data integrity.
5. During emergencies, the need for rapid access to patient information and effective coordination is critical, which can be improved through cloud and blockchain-powered systems.

**3.INTRODUCTION**

The "Cloud and Blockchain Powered Hospital Management" project is poised to transform the landscape of healthcare by integrating two of the most revolutionary technologies of our time: cloud computing and blockchain. In the face of escalating challenges such as fragmented data systems, inefficiencies in hospital operations, and pressing concerns over data security and privacy, this project offers a comprehensive and future-ready solution.

Cloud computing introduces unparalleled scalability and flexibility, enabling hospitals to manage vast amounts of data with real-time accessibility and operational agility. By shifting data storage and management to the cloud, healthcare institutions can streamline their operations, reduce overhead costs, and enhance their ability to respond swiftly to emerging needs.

Simultaneously, blockchain technology provides a secure and transparent framework for handling sensitive health information. Its immutable ledger ensures the integrity of patient records and combats issues such as fraud and unauthorized data manipulation. Blockchain’s decentralized nature also facilitates seamless data sharing and interoperability across various healthcare providers, enhancing collaborative care and patient outcomes.

Together, these technologies offer a synergistic approach to overcoming the inefficiencies and vulnerabilities of traditional hospital management systems. The project not only aims to improve administrative efficiency and compliance with regulatory standards but also seeks to empower patients with greater control over their health data. By setting new benchmarks for data security, operational efficiency, and patient engagement, the "Cloud and Blockchain Powered Hospital Management" project represents a significant leap toward a more resilient, transparent, and patient-centric healthcare system.

**4.EXISTING METHODS**

Existing hospital management systems face challenges with outdated technologies and inefficient methods. Centralized databases, paper records, legacy EHRs, and manual processes often lead to data silos, security issues, and operational inefficiencies, highlighting the need for more advanced and integrated solutions. Some of the existing systems are:

* **Traditional Database Management Systems (DBMS):** Centralized systems prone to data silos and security vulnerabilities.
* **Paper-Based Records:** Physical paperwork that risks loss, damage, and inefficient data retrieval.
* **Legacy Electronic Health Records (EHR) Systems:** Outdated systems with integration and security challenges.
* **On-Premises Software Solutions:** Local software that is costly to maintain and lacks scalability.
* **Manual Data Entry and Paper Trails:** Error-prone and time-consuming processes leading to inefficiencies.

**4.1 DISADVANTAGES**

1. Fragmented patient information due to isolated systems.

2. Increased risk of data breaches and unauthorized access.

3. Errors and inefficiencies from manual processes and outdated software.

4. High costs and resource demands for maintaining legacy systems.

5. Poor integration limiting effective data sharing and collaboration.

**5. PROPOSED WORK**

The proposed hospital website system incorporates cloud and blockchain technologies to enhance both functionality and security. Cloud computing will provide scalable, reliable access to services and patient data, while blockchain ensures secure, transparent management of medical records and transactions. Java will be used as the primary programming language, facilitated by NetBeans for efficient development and coding. SQLyog Community will handle database management, offering advanced tools for SQL querying and optimization. This integrated approach aims to streamline patient interactions, bolster data security, and modernize the overall digital experience.

**5.1 ADVANTAGES**

1. Scalability and reliability through cloud computing.

2. Enhanced security and transparency with blockchain.

3. Efficient development using Java and NetBeans.

4. Optimized database management with SQLyog Community.

5. Improved patient interactions and modernized digital experience.

1. **SYSTEM REQUIREMENTS**

Software needs include the Java Development Kit (JDK) and NetBeans IDE for development, SQLyog Community for database management. Hardware should include reliable network infrastructure and servers with adequate CPU, memory, and storage. Users need compatible web browsers and accounts with appropriate permissions for various roles, while development and testing environments should be separated from production.

* 1. **HARDWARE REQUIREMENTS**
* System : Dual core.
* Hard Disk : 40 GB.
* Floppy Drive : 1.44 Mb.
* Monitor : 15 VGA Colour.
* Mouse : Logitech.
* Ram : 4 GB.

**6.2 SOFTWARE REQUIREMENTS**

* Software : Net beans IDE, SQLyog Community
* Operating system : Windows XP/7/10/11.
* Coding Language : JAVA
* Data Base : MYSQL

1. **SYSTEM ARCHITECTURE**

**7.1 ARCHITECTURE OVERVIEW**

Fig 1. Architecture diagram

This architecture diagram illustrates a secure healthcare system where doctors, patients and cloud technology interact to manage and access sensitive patient information. The system begins with the doctor, who registers their details into the system and can view patient information after decrypting it. Patient data is uploaded through a patient sensor, which securely transmits the data in an encrypted format to a central database. The cloud server plays a crucial role in managing access control, ensuring the security and integrity of the system. It is responsible for providing the necessary keys to doctors, allowing them to decrypt and view patient information. Additionally, the cloud server has oversight of both doctor and patient details, authorizing access to ensure that only verified personnel can access sensitive medical data. This design ensures secure communication and proper authorization while handling encrypted medical records in a cloud-based environment.

**7.2 DESCRIPTION OF THE MODULES**

1. **DOCTOR MODULE:**

* **Login:**

Secure access for doctors with credentials.

* **View Profile:**

Update and view personal and professional details.

* **Read Sensor Patient Data & Add:**

Access and input real-time sensor data from patients.

* **View Sensor Patient Details:**

Review detailed patient sensor data and trends.

* **View Key Request:**

Manage and review requests for data access or medical keys.

* **View Clinical Reports:**

Access and review patient clinical reports and test results.

* **Logout:**

Securely exit the system to protect data.

1. **PATIENT SENSOR MODULE:**

* **Register:**

Sign up and configure sensor devices for patient monitoring.

* **View Profile:**

Access and update personal details related to sensor use.

* **Read Sensor Patient Details:**

Review data collected from patient sensors, such as vital signs.

* **View Key Request:**

Manage and view requests for access to sensor data or medical keys.

* **View Clinical Reports:**

Access clinical reports related to sensor data and patient health.

* **Logout:**

Securely exit the system to protect personal and medical information.

1. **CLOUD SERVER MODULE:**

* **Login:**

Secure access to the cloud server with credentials.

* **View and Authorize Doctor:**

Review and approve doctor access requests.

* **View Clinical Report:**

Access and review clinical reports stored on the cloud.

* **View Patient Details:**

Access detailed patient information.

* **Trace Patients by Blockchain:**

Track and verify patient data transactions using blockchain technology.

* **View Access Control Request:**

Manage and review requests for data access permissions.

* **View Encryption Key Requests:**

Handle requests for encryption keys used in data protection.

* **View Key Transaction:**

Monitor and review transactions involving security keys.

* **View Result in Chart:**

Visualize data and results using charts for better analysis.

* **Logout:**

Securely exit the system to protect sensitive information.

1. **SYSTEM DESIGN**

The hospital management system is designed to incorporate cutting-edge technologies to deliver a secure, efficient, and user-friendly experience. At the heart of the system is a web-based interface that provides patients, medical staff, and administrators with intuitive access to various functionalities from any internet-enabled device. This interface is developed using Java, facilitated by the NetBeans Integrated Development Environment (IDE), which streamlines the coding process and enhances development efficiency. For managing and optimizing database operations, the system employs SQLyog Community, a powerful tool that supports advanced SQL querying, indexing, and overall database management. Security and data integrity are paramount, which is why the system integrates blockchain technology to handle medical records and transactions. Blockchain provides an immutable ledger that ensures all records are secure, transparent, and tamper-proof. This technology enhances trust and accountability in data handling, critical for maintaining the integrity of medical records. The system is hosted on a cloud infrastructure, which offers scalability and reliability, accommodating varying workloads and ensuring high availability. Cloud computing allows for flexible resource management, providing the necessary computational power and storage capacity as needed.

To protect sensitive information, the system uses AES (Advanced Encryption Standard) for encryption. AES is a highly secure encryption algorithm that protects data both in transit and at rest, ensuring that patient information and medical records remain confidential and secure against unauthorized access. This encryption, coupled with secure cloud hosting and blockchain technology, forms a robust security framework designed to meet the demanding needs of modern healthcare environments. This comprehensive design aims to enhance operational efficiency, improve data security, and provide a modern, reliable digital experience for all users.

**8.1 USE CASE DIAGRAM**

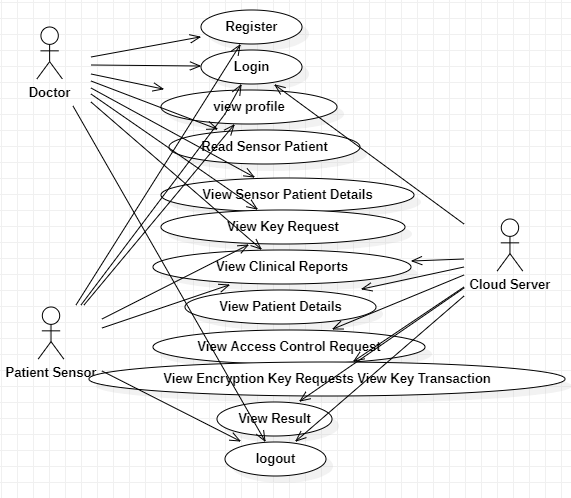


Fig 2. Use case diagram

The use case diagram for the hospital management system illustrates the interactions between different user roles—patients, doctors, and cloud server. Patients can register, view medical records, and schedule appointments, while doctors can update patient information and manage treatment plans. Administrators oversee user accounts and system operations, ensuring data security and compliance. The system also handles secure transactions and data encryption using AES and blockchain technology, ensuring the protection and integrity of sensitive information. This diagram provides a clear overview of how users interact with the system and the key processes involved.

**8.2 CLASS DIAGRAM**

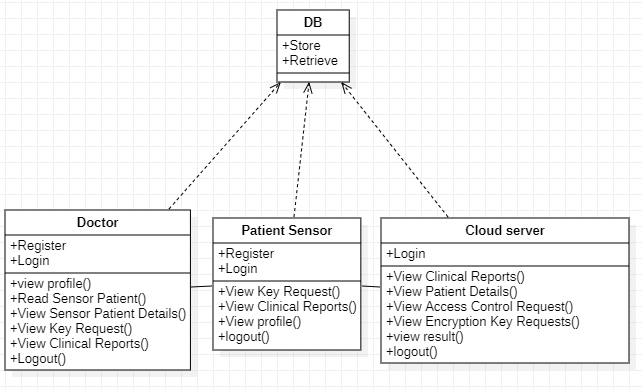


Fig 3. Class diagram

The class diagram for the hospital management system outlines the main classes and their relationships. It includes classes such as Patient, Doctor, Data base and cloud server. Each class has specific attributes and methods, such as managing patient details, scheduling appointments, and handling medical records. The diagram illustrates relationships like one-to-many associations between patients and appointments, and doctors and appointments, as well as administrators managing user accounts. This diagram provides a structured view of the system's data and its interactions.

**8.3 SEQUENCE DIAGRAM**

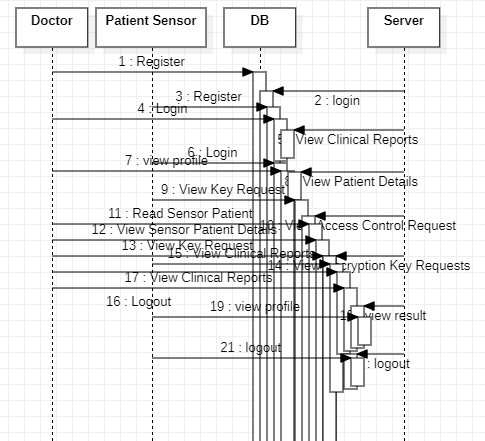


Fig 4. Sequence diagram

The sequence diagram for the hospital management system details the interactions between users and the system over time. It illustrates how a patient schedules an appointment, the system processes the request, and updates the doctor's and patient's schedules. The diagram also shows how a doctor retrieves and updates medical records, and how administrators handle user account management. It captures the flow of messages and operations, providing a clear view of the system's dynamic behaviour and the sequence of events during typical interactions.

**8.4 COLLABORATION DIAGRAM**

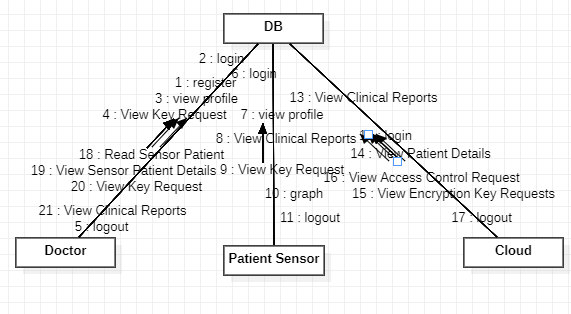


Fig 5. Collaboration diagram

The collaboration diagram for the hospital management system shows the interactions between various objects and their relationships during a process. It details how objects such as patients, doctors, and administrators interact with the system to perform tasks like scheduling appointments, updating medical records, and managing user accounts. The diagram emphasizes the structure of object relationships and the flow of messages between them, providing a visual representation of how different components work together to accomplish system functions.

**8.5 DEPLOYMENT DIAGRAM**

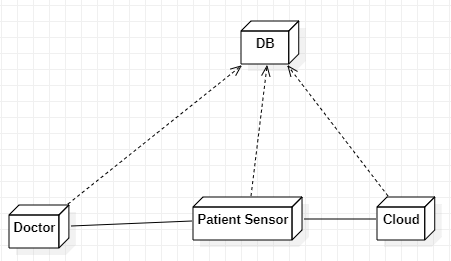


Fig 6. Deployment diagram

The deployment diagram for the hospital management system illustrates the physical setup of the system's components across different servers. It shows how the Database Server stores patient and medical records securely and how it interacts with both the Cloud Server and Patient Server. The Cloud Server handles scalable application hosting and processes requests from the Patient Server and Doctor Server, which facilitate user interactions for patients and healthcare providers, respectively. All components are interconnected, ensuring seamless data flow and functionality between the servers, enabling efficient and secure management of hospital operations.

**8.6 DATA FLOW DIAGRAM**

**LEVEL 0:**

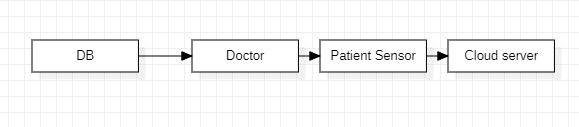


Fig 7. DFD Level 0

**LEVEL 1:**

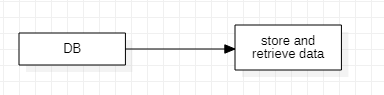
****

Fig 8. DFD Level 1

**LEVEL 2:**

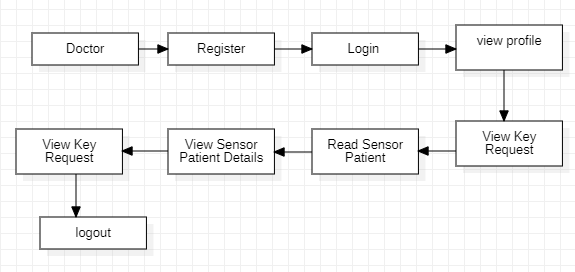


Fig 9. DFD Level 2

**LEVEL 3:**

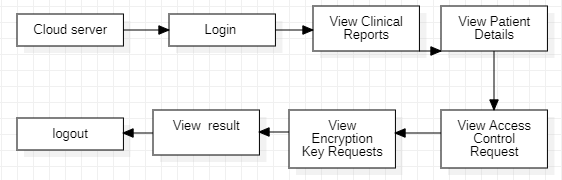


Fig 10. DFD Level 3

**LEVEL 4:**

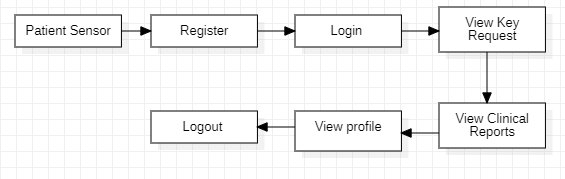


Fig 10. DFD Level 4

The data flow diagram for the hospital management system depicts the movement and transformation of data through the system. It illustrates how data is input by users, processed by various system components, and then output or stored. Key elements include data sources like patient inputs and medical records, processing stages such as appointment scheduling and record updates, and data storage locations like the database. This diagram provides a clear view of how information flows through the system, highlighting interactions between users and the system's data management processes.

**8.7 ENTITY RELATIONSHIP DIAGRAM**

Doctor

Cloud server

Patient sensor

Fig 11. ER diagram

The ER (Entity-Relationship) diagram for the hospital management system illustrates the data structure by showing entities like Patients, Doctors, and Appointments, along with their attributes and relationships. It provides a clear view of how these data elements are interconnected, aiding in the design and management of the database schema.

1. **SYSTEM IMPLEMENTATION**

**9.1 CLIENT-SIDE CODING (FRONT END)**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

<meta name="description" content="">

<meta name="author" content="">

<link href="https://fonts.googleapis.com/css?family=Open+Sans:100,200,300,400,500,600,700,800,900" rel="stylesheet">

<title>S.S. Nursing Home</title>

<!-- Bootstrap core CSS -->

<link href="vendor/bootstrap/css/bootstrap.min.css" rel="stylesheet">

<!-- Additional CSS Files -->

<link rel="stylesheet" href="assets/css/fontawesome.css">

<link rel="stylesheet" href="assets/css/templatemo-eduwell-style.css">

<link rel="stylesheet" href="assets/css/owl.css">

<link rel="stylesheet" href="assets/css/lightbox.css">

<link href="https://cdnjs.cloudflare.com/ajax/libs/simple-line-icons/2.4.1/css/simple-line-icons.min.css" rel="stylesheet">

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css">

<link rel="stylesheet" href="assets/css/style.css">

<!--TemplateMo 573 EduWell

https://templatemo.com/tm-573-eduwell

-->

</head>

<style>

.registration-form{

padding: 50px 0;

}

.registration-form form{

background-color: #fff;

max-width: 600px;

margin: auto;

padding: 50px 70px;

border-top-left-radius: 30px;

border-top-right-radius: 30px;

box-shadow: 0px 2px 10px rgba(0, 0, 0, 0.075);

}

.registration-form .form-icon{

text-align: center;

background-color: #5891ff;

border-radius: 50%;

font-size: 40px;

color: white;

width: 100px;

height: 100px;

margin: auto;

margin-bottom: 50px;

line-height: 100px;

}

.registration-form .item{

border-radius: 20px;

margin-bottom: 25px;

padding: 10px 20px;

}

.registration-form .create-account{

border-radius: 30px;

padding: 10px 20px;

font-size: 18px;

font-weight: bold;

background-color: #5791ff;

border: none;

color: white;

margin-top: 20px;

}

.registration-form .social-media{

max-width: 600px;

background-color: #fff;

margin: auto;

padding: 35px 0;

text-align: center;

border-bottom-left-radius: 30px;

border-bottom-right-radius: 30px;

color: #9fadca;

border-top: 1px solid #dee9ff;

box-shadow: 0px 2px 10px rgba(0, 0, 0, 0.075);

}

.registration-form .social-icons{

margin-top: 30px;

margin-bottom: 16px;

}

.registration-form .social-icons a{

font-size: 23px;

margin: 0 3px;

color: #5691ff;

border: 1px solid;

border-radius: 50%;

width: 45px;

display: inline-block;

height: 45px;

text-align: center;

background-color: #fff;

line-height: 45px;

}

.registration-form .social-icons a:hover{

text-decoration: none;

opacity: 0.6;

}

@media (max-width: 576px) {

.registration-form form

padding: 50px 20px;

}

.registration-form .form-icon{

width: 70px;

height: 70px;

font-size: 30px;

line-height: 70px;

}

}

</style>

<body>

<!-- \*\*\*\*\* Header Area Start \*\*\*\*\* -->

<header class="header-area header-sticky"

<div class="container">

<div class="row">

<div class="col-12">

<nav class="main-nav">

<!-- \*\*\*\*\* Logo Start \*\*\*\*\* -->

<a href="index.html" class="logo">

<h2 style=" margin-left:-50px;font-size:60px;"><em >S.S. Nursing Home </em></h2>

</a>

<!-- \*\*\*\*\* Logo End \*\*\*\*\* -->

<!-- \*\*\*\*\* Menu Start \*\*\*\*\* -->

<ul class="nav">

<li><a href="index.html" class="active">Home</a></li>

<li><a href="patient.jsp">Patient </a></li>

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

<li><a href="central.jsp">Central Server</a></li>

</ul>

<a class='menu-trigger'>

<span>Menu</span

</a>

<!-- \*\*\*\*\* Menu End \*\*\*\*\* -->

</nav>

</div>

</div>

</div>

</header>

<section class="main-banner" id="top">

<div class="container">

<div class="row">

<div class="col-lg-6 align-self-center">

<div class="header-text">

<h2><em>Welcome Central server</em></h2>

<em> <form action="cloginc.jsp" method="post">

<div class="form-group">

<input type="text" class="form-control item" name="user" id="user" placeholder="Username">

</div>

<div class="form-group">

<input type="password" class="form-control item" name="pass" id="pass" placeholder="Password">

</div>

<div class="form-group">

<div class="main-button-gradient">

</div>

</div>

<div class="row">

<div class="col-lg-6">

<div class="main-button-gradient">

<input type="submit" class="form-control item" style=" background-color: violet;" textcolor="white" value="Login">

</div>

</div>

<div class="col-lg-6">

<div class="main-button-gradient">

<div class="scroll-to-section"><a href="dreg.jsp">New Register</a></div>

</div>

</div></div>

</form>

</em>

</div>

</div>

<div class="col-lg-6">

<div class="right-image">

<img src="assets/images/sign.jpg" alt="">

</div>

</div>

</div>

</div>

</section>

</body>

</html>

**9.2 SERVER-SIDE CODING (BACK END)**

/\*

SQLyog Community v12.02 (32 bit)

MySQL - 5.5.29 : Database - blockhealthcare

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

/\*!40101 SET NAMES utf8 \*/;

/\*!40101 SET SQL\_MODE=''\*/;

/\*!40014 SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0 \*/;

/\*!40014 SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0 \*/;

/\*!40101 SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='NO\_AUTO\_VALUE\_ON\_ZERO' \*/;

/\*!40111 SET @OLD\_SQL\_NOTES=@@SQL\_NOTES, SQL\_NOTES=0 \*/;

CREATE DATABASE /\*!32312 IF NOT EXISTS\*/`blockhealthcare` /\*!40100 DEFAULT CHARACTER SET latin1 \*/;

USE `blockhealthcare`;

/\*Table structure for table `blockchain` \*/

DROP TABLE IF EXISTS `blockchain`;

CREATE TABLE `blockchain` (

  `id` int(11) NOT NULL AUTO\_INCREMENT,

  `digsignature` varchar(300) DEFAULT NULL,

  PRIMARY KEY (`id`) ENGINE=InnoDB AUTO\_INCREMENT=93 DEFAULT CHARSET=latin1;

/\*Data for the table `blockchain` \*/

insert into `blockchain`(`id`,`digsignature`) values

/\*Table structure for table `dreg` \*/

DROP TABLE IF EXISTS `dreg`;

CREATE TABLE `dreg` (

  `id` int(200) NOT NULL AUTO\_INCREMENT,

  `dname` varchar(200) DEFAULT NULL,

  `pass` varchar(200) DEFAULT NULL,

  `hname` varchar(200) DEFAULT NULL,

  `des` varchar(200) DEFAULT NULL,

  `spec` varchar(200) DEFAULT NULL,

  `dob` varchar(200) DEFAULT NULL,

  `email` varchar(200) DEFAULT NULL,

  `contact` varchar(200) DEFAULT NULL,

  `location` varchar(200) DEFAULT NULL,

  `status` varchar(200) DEFAULT 'Waiting',

  PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=21 DEFAULT CHARSET=latin1;

/\*Data for the table `dreg` \*/

insert  into `dreg`(`id`,`dname`,`pass`,`hname`,`des`,`spec`,`dob`,`email`,`contact`,`location`,`status`) values

(16,'Lakshmi','lakshmi','Apollo','Doctor','Cardiologists','1998-06-05','lakshmi@gmail.com','898879898','Chennai','Activated'),

(17,'Camy','camy','Apollo','Doctor','Gynecologists','2023-03-16','camy@gmail.com','8739579345','Chennai','Activated'),

(18,'Jhansi','jhansi','Apollo','Doctor','General','2023-03-09','jhansi@gmail.com','8797878798','Chennai','Waiting'),

(19,'Vel','vel','Vijaya','Doctor','Cardiologists','3332-03-31','vel@gmail.com','8737846778','Chennai','Activated'),

(20,'Ram','ram','SMK','Doctor','General','1985-05-14','ram@gmail.com','7895642565','Chennai','Activated');

/\*Table structure for table `preg` \*/

DROP TABLE IF EXISTS `preg`;

CREATE TABLE `preg` (

  `id` int(22) NOT NULL AUTO\_INCREMENT,

  `pname` varchar(222) DEFAULT NULL,

  `pass` varchar(222) DEFAULT NULL,

  `dob` varchar(222) DEFAULT NULL,

  `email` varchar(222) DEFAULT NULL,

  `contact` varchar(222) DEFAULT NULL,

  `location` varchar(222) DEFAULT NULL,

  `status` varchar(222) DEFAULT 'Waiting',

  PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=5 DEFAULT CHARSET=latin1;

/\*Data for the table `preg` \*/

insert  into `preg`(`id`,`pname`,`pass`,`dob`,`email`,`contact`,`location`,`status`) values (3,'Deepa','deepa','4344-0606','deepa@gmail.com','8355454588','Chennai','Activated'),

(4,'Arun','arun','2000-11-12','arun@gmail.com','9875486521','chennai','Activated');

/\*Table structure for table `request` \*/

DROP TABLE IF EXISTS `request`;

CREATE TABLE `request` (

  `sno` int(22) NOT NULL AUTO\_INCREMENT,

  `pname` varchar(222) DEFAULT NULL,

  `keyy` varchar(222) DEFAULT 'Generate Symetric Key',

  `status` varchar(222) DEFAULT 'Waiting',

  PRIMARY KEY (`sno`)

) ENGINE=InnoDB AUTO\_INCREMENT=12 DEFAULT CHARSET=latin1;

/\*Data for the table `request` \*/

insert  into `request`(`sno`,`pname`,`keyy`,`status`) values (1,'Deepa','512509629D305FF9','Permitted'),

(5,'Deepa','Generate Symetric Key','Waiting'),

(6,'Deepa','Generate Symetric Key','Waiting'),

(7,'Deepa','Generate Symetric Key','Waiting'),

(8,'Arun','Generate Symetric Key','Waiting'),

(9,'Arun','Generate Symetric Key','Waiting'),

(10,'Arun','5CB1B2F3446BA207','Permitted'),

(11,'arun','F0FA7CD84167C829','Permitted');

/\*Table structure for table `request1` \*/

DROP TABLE IF EXISTS `request1`;

CREATE TABLE `request1` (

  `sno` int(22) NOT NULL AUTO\_INCREMENT,

  `pname` varchar(222) DEFAULT NULL,

  `patient` varchar(222) DEFAULT NULL,

  `dname` varchar(222) DEFAULT NULL,

  `status` varchar(222) DEFAULT 'Waiting',

  `pstatus` varchar(222) DEFAULT 'Waiting',

  `file` varchar(222) DEFAULT NULL,

  `keyy` varchar(222) DEFAULT 'Waiting',

  `estatus` varchar(222) DEFAULT 'Waiting',

  `sol` varchar(222) DEFAULT NULL,

  PRIMARY KEY (`sno`)

) ENGINE=InnoDB AUTO\_INCREMENT=13 DEFAULT CHARSET=latin1;

/\*Data for the table `request1` \*/

insert into `request1`(`sno`,`pname`,`patient`,`dname`,`status`,`pstatus`,`file`,`keyy`,`estatus`,`sol`) values

(5,'Deepa','Rani','Camy','Permitted','Accepted','Resource Allocation.txt','512509629D305FF9','Key Sent','Do yoga Daily. No need any operation. Take proper food'),

(6,'Deepa','Rani','Camy','Waiting','Waiting','Resource Allocation.txt','512509629D305FF9','Waiting',NULL),

(7,'Arun','Arun','Camy','Permitted','Accepted','Blockhealthcare.sql','5CB1B2F3446BA207','Key Sent','ghhg'),

(12,'Arun','Arun','Camy','Permitted','Waiting','Blockhealthcare.sql','5CB1B2F3446BA207','Waiting','ghhg');

/\*Table structure for table `upload` \*/

DROP TABLE IF EXISTS `upload`;

CREATE TABLE `upload` (

  `sno` int(22) NOT NULL AUTO\_INCREMENT,

  `keyy` varchar(222) DEFAULT NULL,

  `pname` varchar(222) DEFAULT NULL,

  `patient` varchar(222) DEFAULT NULL,

  `dob` varchar(222) DEFAULT NULL,

  `gen` varchar(222) DEFAULT NULL,

  `diseas` varchar(222) DEFAULT NULL,

  `age` varchar(222) DEFAULT NULL,

  `blood` varchar(222) DEFAULT NULL,

  `email` varchar(222) DEFAULT NULL,

  `addr` varchar(222) DEFAULT NULL,

  `con` varchar(222) DEFAULT NULL,

  `dname` varchar(222) DEFAULT NULL,

  `file` varchar(222) DEFAULT NULL,

  PRIMARY KEY (`sno`)

) ENGINE=InnoDB AUTO\_INCREMENT=8 DEFAULT CHARSET=latin1;

/\*Data for the table `upload` \*/

insert  into `upload`(`sno`,`keyy`,`pname`,`patient`,`dob`,`gen`,`diseas`,`age`,`blood`,`email`,`addr`,`con`,`dname`,`file`) values (4,'512509629D305FF9','Deepa','Rani','14/09/1996','Female','Fever','30','o+ve','rani@gmail.com','Chennai','9879789676','Camy','Resource Allocation.txt'),

(5,'5CB1B2F3446BA207','Arun','Arun','12/09/1999','Male','Cancer','24','B+','arun12@gmail.com','Chennai','87665756454','','Blockhealthcare.sql'),

(6,'5CB1B2F3446BA207','Arun','Arun','12/09/1999','Male','Cancer','30','O+','arun12@gmail.com','Chennai','78967867546','Camy','Blockhealthcare.sql'),

(7,'5CB1B2F3446BA207','Arun','Arun','12/09/1999','Male','Cancer','30','O+','arun12@gmail.com','Chennai','78967867546','Camy','Blockhealthcare.sql');

/\*!40101 SET SQL\_MODE=@OLD\_SQL\_MODE \*/;

/\*!40014 SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS \*/;

/\*!40014 SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS \*/;

/\*!40111 SET SQL\_NOTES=@OLD\_SQL\_NOTES \*/;

1. **. SYSTEM TESTING**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**10.1 TEST CASES**

**UNIT TESTING:**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration.

Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results. Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**Test strategy and approach:**

**Test objectives:**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

**Features to be tested:**

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

**INTEGRATION TESTING:**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**FUNCTION TESTING:**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**SYSTEM TESTING:**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**WHITE BOX TESTING:**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**BLACK BOX TESTING:**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**ACCEPTANCE TESTING:**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:**

All the test cases mentioned above passed successfully. No defects encountered.

**10.2 PERFORMANCE ANALYSIS**

A thorough performance analysis of a hospital management system encompasses several critical dimensions to ensure its effectiveness in optimizing hospital operations and enhancing patient care. First, the core functionalities of the system must be rigorously evaluated to confirm that it covers essential modules such as patient registration, appointment scheduling, electronic medical records , billing, pharmacy management, and inventory control. It is crucial that these modules integrate seamlessly with other hospital systems, such as laboratory and radiology information systems, to facilitate smooth data flow and reduce operational silos. The system’s user experience is another vital aspect, requiring an intuitive and user-friendly interface that supports various roles—doctors, nurses, administrative staff—and is accessible across different devices like desktops, tablets, and smartphones.

Performance metrics are critical for assessing the system’s efficiency and include response time, which measures how quickly the system processes user inputs and queries. System throughput evaluates the ability to handle multiple simultaneous users and transactions without degradation in performance. Scalability is also essential, as it determines the system’s capacity to grow in terms of user load and data volume without compromising performance. Reliability and availability are assessed by examining system uptime and error rates, with the goal of achieving minimal downtime and high system stability.

Security is a paramount concern, involving robust data protection mechanisms such as encryption, access controls, and adherence to regulations like HIPAA or GDPR. An effective system must also include comprehensive audit trails to track user activities and modifications to patient records, ensuring data integrity and accountability. Data management capabilities are scrutinized to ensure accurate data entry, retrieval, and consistency, along with effective backup and recovery processes to safeguard against data loss.

Customization and flexibility are key for adapting the system to specific hospital workflows and needs. This includes evaluating how easily the system can be tailored to fit unique operational requirements and how efficiently updates and maintenance can be managed without disrupting daily activities. Cost efficiency is another important aspect, involving an analysis of the total cost of ownership (TCO), which includes licensing, implementation, training, and ongoing maintenance, alongside assessing the return on investment (ROI) based on improved operational efficiency and patient care outcomes.

Training and support services are essential for ensuring that staff can effectively utilize the system. This involves evaluating the quality of training programs provided and the responsiveness and effectiveness of technical support. Finally, compliance with regulatory standards and industry interoperability are critical for ensuring that the system meets legal requirements and can effectively communicate with other healthcare systems.

In summary, a comprehensive performance analysis of an hospital management system should address these multifaceted aspects to ensure that the system delivers optimal functionality, reliability, and security, while also being adaptable, cost-effective, and supportive of superior patient care.

**11. CONCLUSION**

The performance of the hospital management system is significantly bolstered by the incorporation of cutting-edge technologies, namely cloud computing, blockchain, NetBeans, and SQLyog Community. These technologies collectively enhance the system's efficiency, security, and adaptability, providing a comprehensive solution for managing hospital operations and improving patient care.

Cloud Computing plays a pivotal role in the system architecture by offering scalable and flexible infrastructure. This technology enables the system to handle vast amounts of data and a high volume of simultaneous users with ease. The cloud environment ensures that the HMS remains accessible from various devices and locations, promoting efficient management and real-time access to critical information. Additionally, cloud services offer robust backup and disaster recovery solutions, which are essential for maintaining high availability and minimizing downtime.

Blockchain Technology contributes a significant layer of security to the management system. By leveraging blockchain’s immutable and transparent ledger capabilities, the system enhances the integrity and security of patient data. Every transaction and record update is securely logged and cannot be altered retroactively, which minimizes the risk of data tampering and fraud. This not only ensures compliance with stringent data protection regulations but also builds trust with patients and stakeholders by safeguarding the accuracy and authenticity of medical records.

NetBeans is utilized as the development environment for the system, providing a versatile platform for creating and managing the software. Its integrated development environment (IDE) offers comprehensive tools for coding, debugging, and testing, which streamline the development process. NetBeans supports multiple programming languages and frameworks, making it an ideal choice for building a robust and maintainable hospital management system. Its user-friendly interface and powerful features facilitate efficient development workflows and help ensure that the system meets high-quality standards.

SQLyog Community serves as the database management tool for the system, offering efficient and effective management of the system’s data. SQLyog Community provides intuitive database administration capabilities, enabling smooth handling of data storage, retrieval, and manipulation. Its features support optimized database performance and ensure that data integrity is maintained across various modules. By facilitating efficient data management practices, SQLyog Community helps ensure that the system remains responsive and reliable.

In summary, the integration of cloud computing, blockchain technology, NetBeans, and SQLyog Community into the hospital management system creates a powerful, secure, and adaptable platform. These technologies collectively enhance the system's capability to manage complex hospital operations, ensure data security and integrity, and support superior patient care. As the healthcare landscape continues to evolve, the system’s adoption of these technologies positions it well to address future challenges and opportunities, making it a vital tool for modern healthcare management.

**11.1 FUTURE ENHANCEMENTS**

As healthcare continues to advance, hospital management systems must evolve to keep pace with technological innovations and changing needs. Future enhancements in hospital management system focus on integrating cutting-edge technologies and improving system capabilities to optimize patient care, streamline operations, and ensure data security. Key areas for development include the adoption of artificial intelligence, advanced data analytics, improved interoperability, and enhanced mobile access, all aimed at delivering a more efficient, secure, and responsive healthcare environment.

Looking ahead, several key enhancements could significantly improve the performance and functionality of a hospital management system, ensuring it continues to meet the evolving needs of healthcare providers and patients:

1. **Artificial Intelligence (AI)**: Integrate AI for predictive analytics, personalized treatment recommendations, and automated patient interactions.
2. **Advanced Data Analytics**: Use sophisticated analytics for deeper insights into operations, patient outcomes, and cost management.
3. **Interoperability**: Enhance integration with other healthcare systems using standards like FHIR and HL7.
4. **Blockchain**: Implement blockchain for improved data security, patient consent management, and supply chain transparency.
5. **Telemedicine**: Incorporate video consultations and telemedicine workflows to support remote care and virtual follow-ups.
6. **Mobile Access**: Expand mobile app features for secure access to patient records, remote monitoring, and communication.
7. **Personalized Patient Portals**: Develop portals with customized health information, resources, and integration with wearables.
8. **Natural Language Processing (NLP)**: Use NLP for efficient data entry and accurate medical record documentation.
9. **Improved User Interfaces**: Enhance UI design for better usability, accessibility, and user satisfaction.
10. **Cloud and Edge Computing**: Combine cloud computing with edge computing for faster, real-time data processing.
11. **Regulatory Compliance**: Implement automatic updates to ensure compliance with evolving healthcare regulations.
12. **Enhanced Training and Support**: Expand training programs and introduce AI-driven support for effective technical assistance.

**12. APPENDICES**

**12.1 SAMPLE SCREENSOTS**

**HOME PAGE**

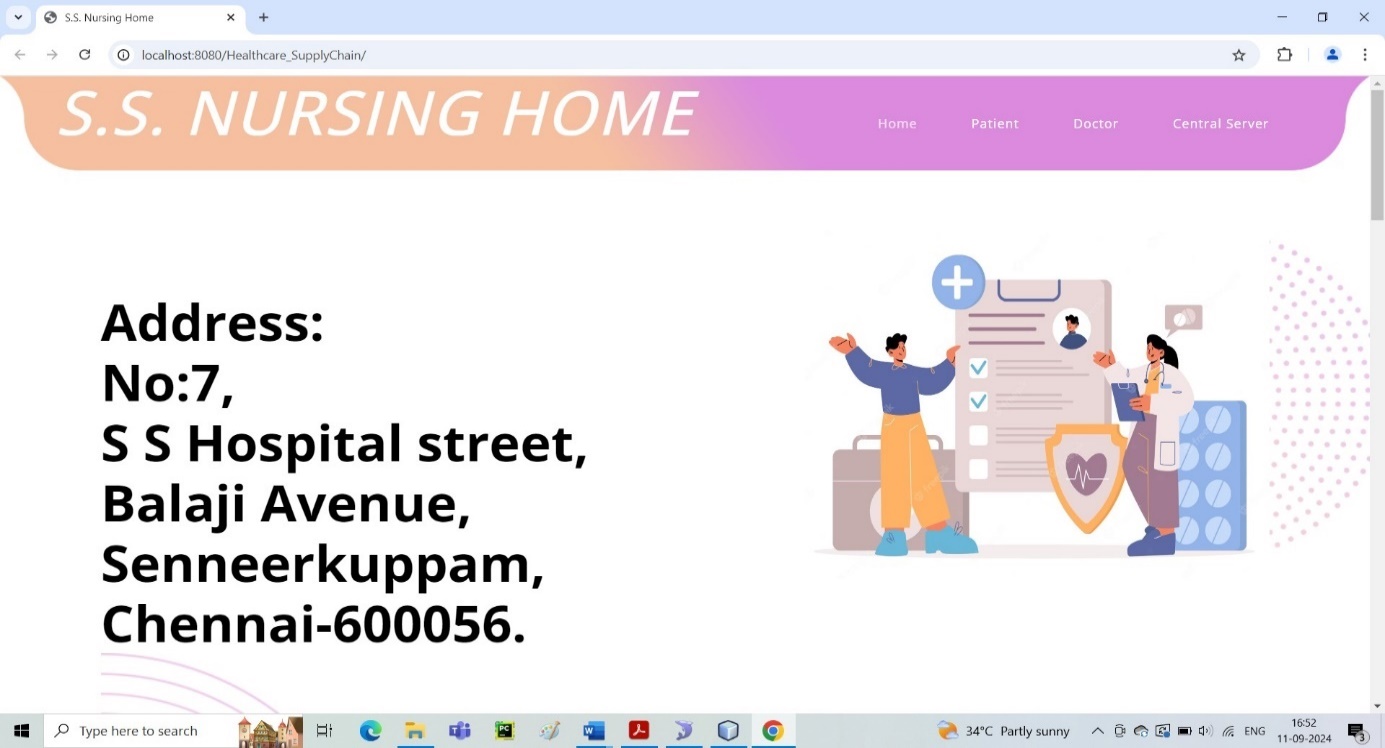
****

Fig 12. Home Page Screenshot

**CENTRAL SERVER LOGIN PAGE**

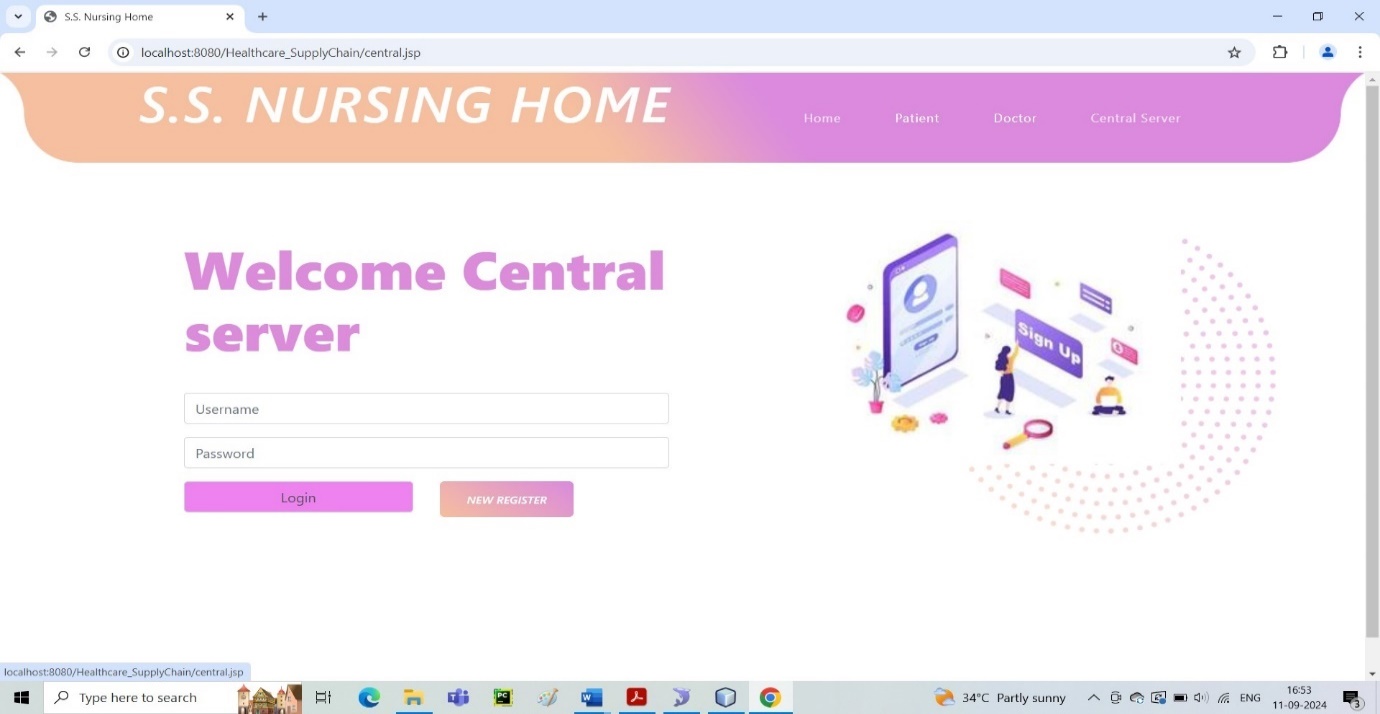
****

Fig 13. Central server login page screenshot

**PATIENT REGISTRATION PAGE**

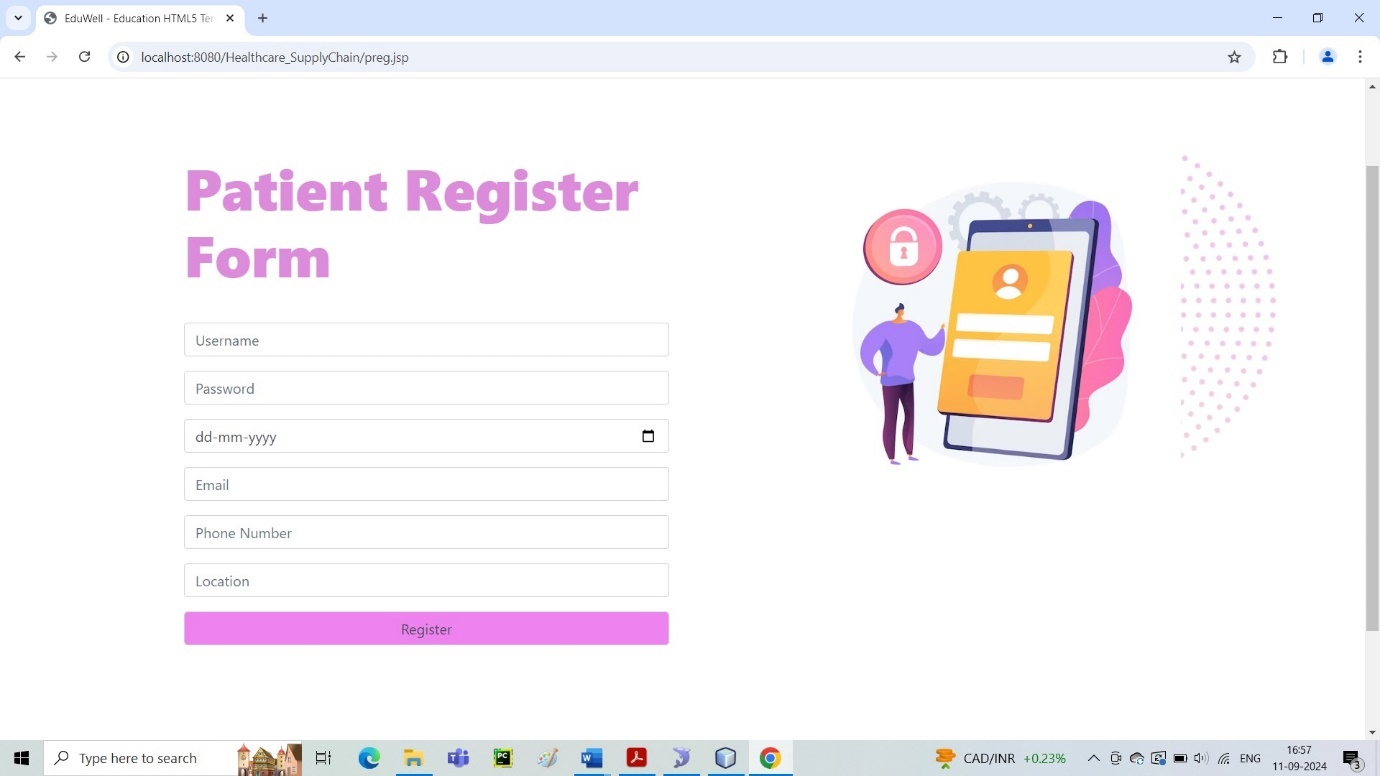
****

Fig 14. Patient registration page screenshot

**PATIENT LOGIN PAGE**

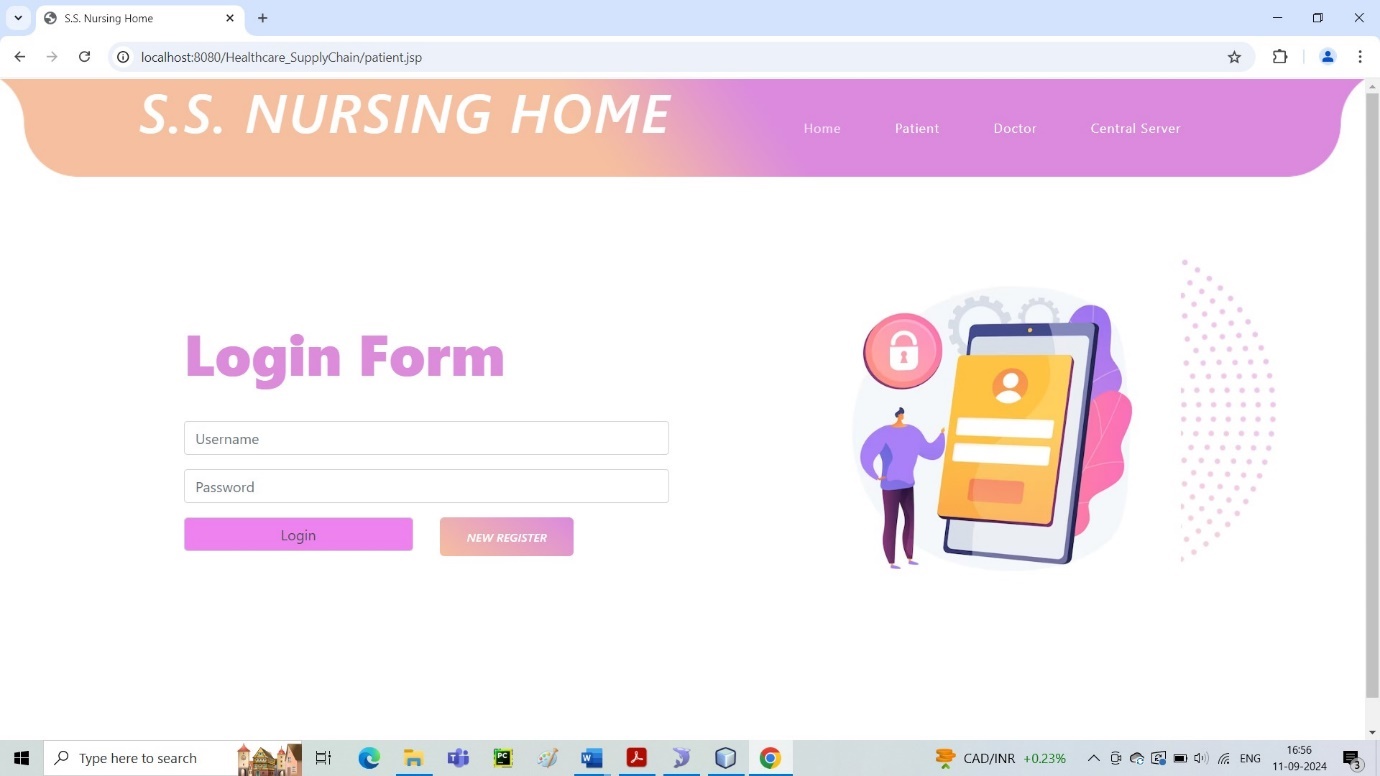
****

Fig 15. Patient login page screenshot

**DOCTOR REGISTRATION PAGE**

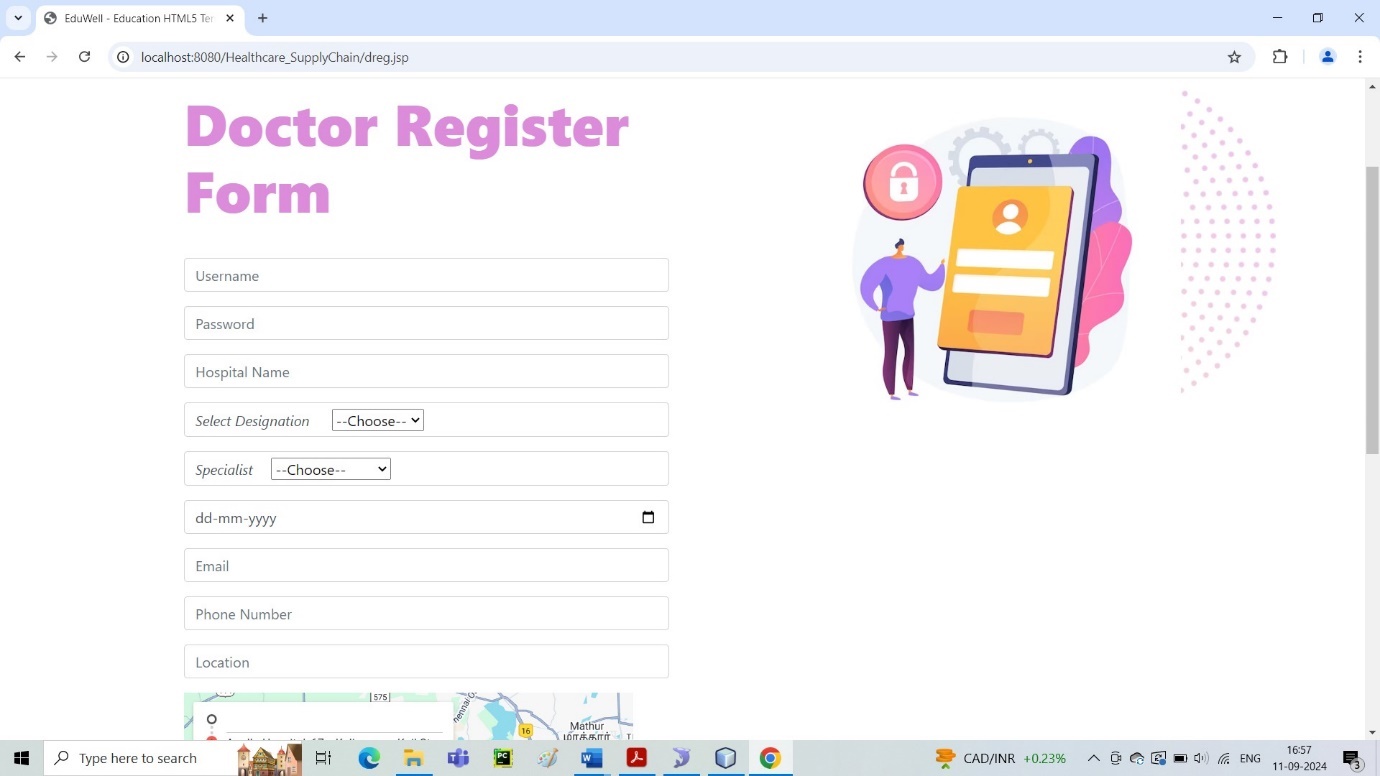
****

Fig 16. Doctor registration page screenshot

**DOCTOR LOGIN PAGE**

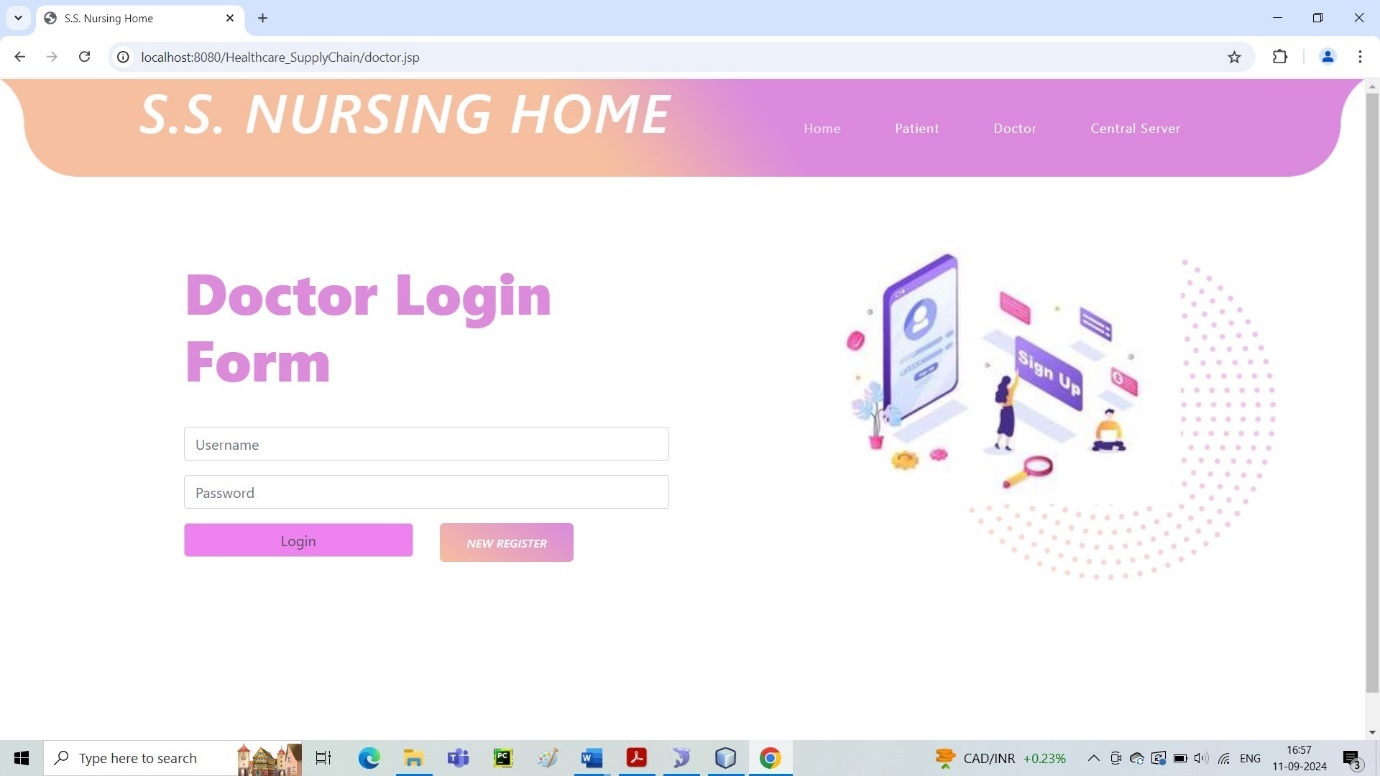
****

Fig 17. Doctor Login page screenshot

**13. REFERENCES**

[1] Y. Sun, F. P. Lo, and B. Lo, “Security and Privacy for the Internet of Medical Things Enabled Healthcare Systems: A Survey,” IEEE Access, vol. 7, pp. 183 339–183 355, 2019.

[2] V. P. Yanambaka, S. P. Mohanty, E. Kougianos, and D. Puthal, “PMsec: Physical Unclonable Function-Based Robust and Lightweight Authentication in the Internet of Medical Things,” IEEE Transactions on Consumer Electronics, vol. 65, no. 3, pp. 388–397, 2019.

[3] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum, and N. Ghani, “Demystifying IoT Security: An Exhaustive Survey on IoT Vulnerabilities and a First Empirical Look on Internet-Scale IoT Exploitations,” IEEE Communications Surveys Tutorials, vol. 21, no. 3, pp. 2702– 2733, 2019.

[4] T. Tekeste, H. Saleh, B. Mohammad, and M. Ismail, IoT for Healthcare: Ultra Low Power ECG Processing System for IoT Devices. Cham: Springer International Publishing, 2019, pp. 7–12.

[5] Y. Zhang, M. Qiu, C. Tsai, M. M. Hassan, and A. Alamri, “Health-CPS: Healthcare Cyber-Physical System Assisted by Cloud and Big Data,” IEEE Systems Journal, vol. 11, no. 1, pp. 88–95, 2017.

[6] X. Xu, X. Zhang, H. Gao, Y. Xue, L. Qi, and W. Dou, “BeCome: Blockchain-Enabled Computation Offloading for IoT in Mobile Edge Computing,” IEEE Transactions on Industrial Informatics, vol. 16, no. 6, pp. 4187–4195, 2020.

[7] S. Biswas, K. Sharif, F. Li, and S. P. Mohanty, “Blockchain for E-Healthcare Systems: Easier Said Than Done,” IEEE Computer, vol. 53, no. 7, pp. 57–67, 2020.

[8] S. Biswas, K. Sharif, F. Li, S. Maharjan, S. P. Mohanty, and Y. Wang, “PoBT: A Lightweight Consensus Algorithm for Scalable IoT Business Blockchain,” IEEE Internet of Things Journal, vol. 7, no. 3, pp. 2343–2355, 2020.

[9] E. Bhaskara Santhosh, S. Priyanka, and A. K. Pradhan, “SHPI: Smart Healthcare System for Patients in ICU using IoT,” in Advanced Networks and Telecommunications Systems, 2019, pp. 1–6.

[10] T. M. Fernandez-Caram ´ es and P. Fraga-Lamas, “A Re- ´ view on the Use of Blockchain for the Internet of Things,” IEEE Access, vol. 6, pp. 32 979–33 001, 2018