Hospital Network Design

A MINI PROJECT REPORT

18CSC302J- COMPUTER NETWORKS

Submitted by

T.ASHISH	**RA2111032010004**
D.R.DEEKSHITH	**RA2111032010005**
B.ABHINAY REDDY	**RA2111032010007**

SCHOOL OF COMPUTING

BACHELOR OF TECHNOLOGY

COMPUTER SCIENCE AND ENGINEERING

(WITH SPECIALIZATION IN INTERNET OF THINGS)

SRM INSTITUTE OF SCIENCE & TECHNOLOGY

(KATTANKULATHUR CAMPUS)

CHENNAI- 603203

SRM INSTITUTE OF SCIENCE & TECHNOLOGY

KATTANKULATHUR – 603 203

BONAFIDE CERTIFICATE

Certified that this mini project report "Hospital Network Design" is the bonafide work of "ASHISH (RA2111032010004),D.R.DEEKSHITH(RA2111032010005) and ABHINAY REDDY (RA2111032010007)" who carried out the project work for **18CSC302J – Computer Networks** under my supervision at **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY,** Kattankulathur during the academic year 2023 – 2024.

SIGNATURE	**SIGNATURE**
Faculty In-Charge	**HEAD OF THE DEPARTMENT**
Dr.R.PRABHU	**Dr. Annapurani Panaiyappan. K**
Assistant Professor	Professor and Head,
Department of Networking and Communications	Department of Networking and Communications
SRM Institute of Science and Technology	SRM Institute of Science and Technology

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 MOTIVATION

Developing a hospital network system using socket programming can offer numerous benefits and address specific needs and challenges within healthcare institutions. Here are some reasons for developing such a system:

Real-Time Communication: Socket programming enables real-time communication between various components of a hospital network, such as medical devices, patient monitoring systems, and healthcare professionals. This allows for immediate data transmission and decision-making, crucial in emergency situations.

Data Sharing: Hospitals generate and handle vast amounts of data daily, including patient records, medical images, lab results, and more. Socket programming can facilitate secure and efficient data sharing among different departments, ensuring that healthcare providers have access to the most up-to-date information.

Hospital network systems can use sockets to facilitate remote monitoring of patients. This is particularly important for critical care units where healthcare providers need to keep a close eye on patients' vital signs and condition.

Developing a hospital network system using socket programming offers a wide range of uses and benefits to healthcare facilities, professionals, and patients.

1.2SCOPE

A hospital network system using socket programming can be developed to facilitate communication between different departments of a hospital. For example, the pharmacy department can communicate with the billing department to ensure that patients are charged accurately for their medications. The laboratory department can communicate with the radiology department to ensure that patients receive the correct tests and treatments.

The system can be designed using a client-server architecture, where each department is a client and the server is responsible for managing all the communication between them. The server can be hosted on a central computer within the hospital, while each department can have its own client application installed on their respective computers.

1.3OBJECTIVES

PRIMARY OBJECTIVES:

- The primary goals of developing a hospital network system using socket programming are to improve communication and collaboration between different departments of a hospital, leading to better patient care and outcomes. The system can be designed using a client-server architecture, where each department is a client and the server is responsible for managing all the communication between them. The server can be hosted on a central computer within the hospital, while each department can have its own client application installed on their respective computers.
- The client applications can be designed to send requests to the server, which will then forward them to the appropriate department. The server can also be designed to send notifications to each department when new requests are received or when certain events occur.
- . Developing a hospital network system using socket programming can help improve communication and collaboration between different departments of a hospital, leading to better patient care and outcomes.

SECONDARY OBJECTIVES:

- Redundancy and Failover: Implementing redundancy and failover mechanisms to ensure system availability and data integrity, even in the event of network or hardware failures.
- Scalability: Designing the system to be easily scalable to accommodate future growth and additional healthcare services, devices, and users..
- Cost Reduction: Reducing operational costs by automating routine tasks, optimizing resource utilization, and minimizing the need for manual data entry and record-keeping.
- Efficient Resource Management: Efficiently managing healthcare resources, such as staff scheduling, room allocation, and equipment utilization.

CHAPTER 2

SOFTWARE REQUIREMENTS

2.1 TECHNOLOGIES USED

- Programming Languages:
  Python
- Web Frameworks:
  Figma,Canva.
- Web Sockets Library:
  Socket.

CHAPTER 3

METHODOLOGY/ DESIGN

3.1 SYSTEM COMPONENTS

Client-Side Components:

- User Interface (UI)
- WebSocket Client.

Server-Side Components:

- WebSocket Server
- Chat Application Logic
- Hospital working logic
- User Authentication and Management
- Database

Infrastructure Components:

- Web Server.
- Database Server

3.2 USER INTERFACE

Interface:

- In a **Hospital Management System** is used to build dynamic and interactive user interfaces for the system. It enables high performance, simplifies development and maintenance, and provides a better user experience.
- experienced designers create visually stunning and intuitive interfaces that enhance user experience. Trust us to transform your mobile app into an aesthetic masterpiece that captivates users and drives success

A dynamic, responsive, and user-friendly interface.

High performance due to its virtual DOM feature.

Easy to maintain and update due to its component-based structure.

3.3 SERVER-SIDE

Web Socket Server:

- Web Socket Protocol Handler.
- Connection Management.
- Message Routing.
- Real-Time Event Handling.

Logic:

- Input Handling
- Execution
- Error Handling

3.4 WEB SOCKETS

Integrating WebSockets into your "Hospital Management System using Socket Programming" project involves incorporating WebSocket communication to enable real-time interactions, such as chat, while maintaining the Hospital functionality. Here's a detailed overview of how WebSockets were integrated into the project's architecture:

1. Selection of WebSocket Technology:
2. WebSocket Server Setup:
3. WebSocket Client Integration:
4. Server-Client Handshake:
5. Establish WebSocket Connections:

When a administrator accesses the website, the WebSocket server establishes a WebSocket connection with the other wards. The server may assign a unique identifier to the connection for tracking.

3.5 FEATURES AND FUNCTIONALITIES

- ### Appointment Management:

Integrate appointment widgets in your online Hospital management system. Besides, enable easy scheduling for patients with the hospital website.

- ### Billing Management

It is difficult to maintain and track separate bills for treatments, testing labs, and diagnostics. Hence, integrate the billing system in your hospital. Besides, get customized alerts for payment dues too.

- ### Prescription Management

Make a note of commonly used medicines. Further, keep track of the availability of frequently used medicines in the pharmacy. On the other hand, switch to digital prescriptions to avoid wrong medication.

CHAPTER 4

RESULTS

4.1 IMPLEMENTATION

1. Importing the socket Module:

   - The code starts by importing the socket module, which provides the necessary functions and classes to create and manage network sockets.

2. Hospital Function:

   Hospital Management Systems (HMS), such as those built using the powerful React JS framework, comprise various modules to provide comprehensive healthcare services. Let’s delve into the various modules of a Hospital Management System in React JS and their functions.

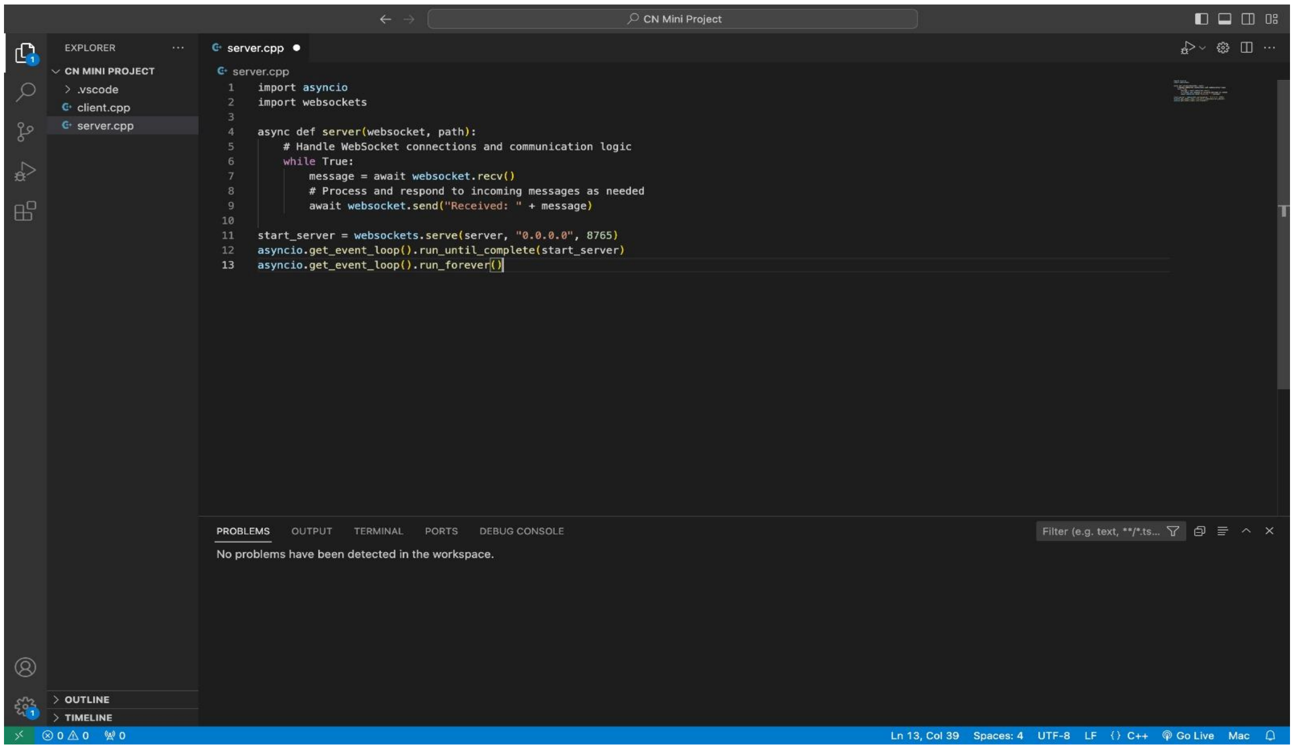
3. main Function:

A Hospital Management System Project in React JS should cater to the diverse needs of a healthcare setup. Here are some functional requirements:

- User-Friendly Interface: The system should have a user-friendly interface that is easy to navigate for users with varying tech proficiency.
- Data Management: It should efficiently manage vast amounts of data, including patient records, appointment details, medical reports, and billing information.
- Integration Capabilities: The system needs to seamlessly integrate with other software in the hospital, such as billing software or lab equipment software.
- Reliability and Scalability: A Hospital Management System React should be reliable, with minimal downtime, and scalable to accommodate the growth of the hospital.
- Regulatory Compliance: The system should comply with healthcare regulations such as HIPAA (Health Insurance Portability and Accountability Act) for data privacy and security.

4.2OUTPUTS

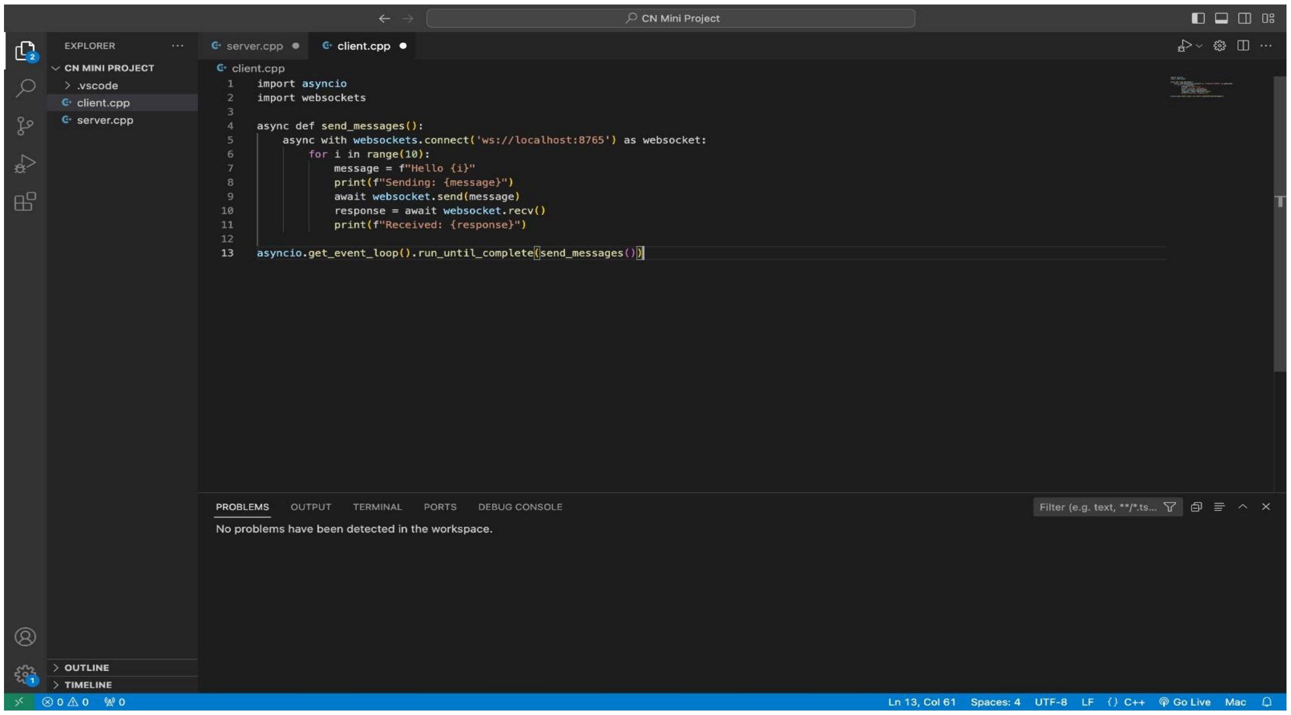
SERVER:



```
import asyncio
import websockets

async def server(websocket, path):
    # Handle WebSocket connections and communication logic
    while True:
        message = await websocket.recv()
        # Process and respond to incoming messages as needed
        await websocket.send("Received: " + message)

start_server = websockets.serve(server, "0.0.0.0", 8765)
asyncio.get_event_loop().run_until_complete(start_server)
asyncio.get_event_loop().run_forever()
```

No problems have been detected in the workspace.

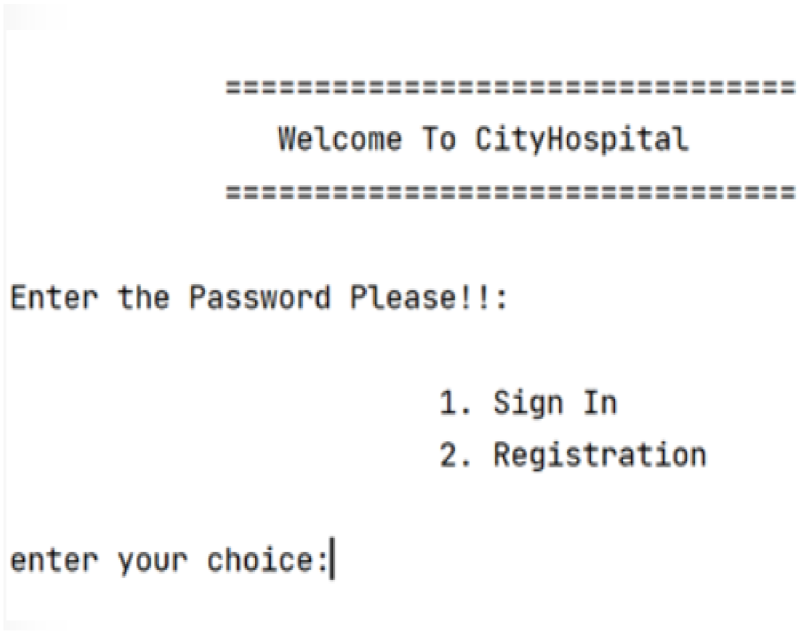
CLIENT:



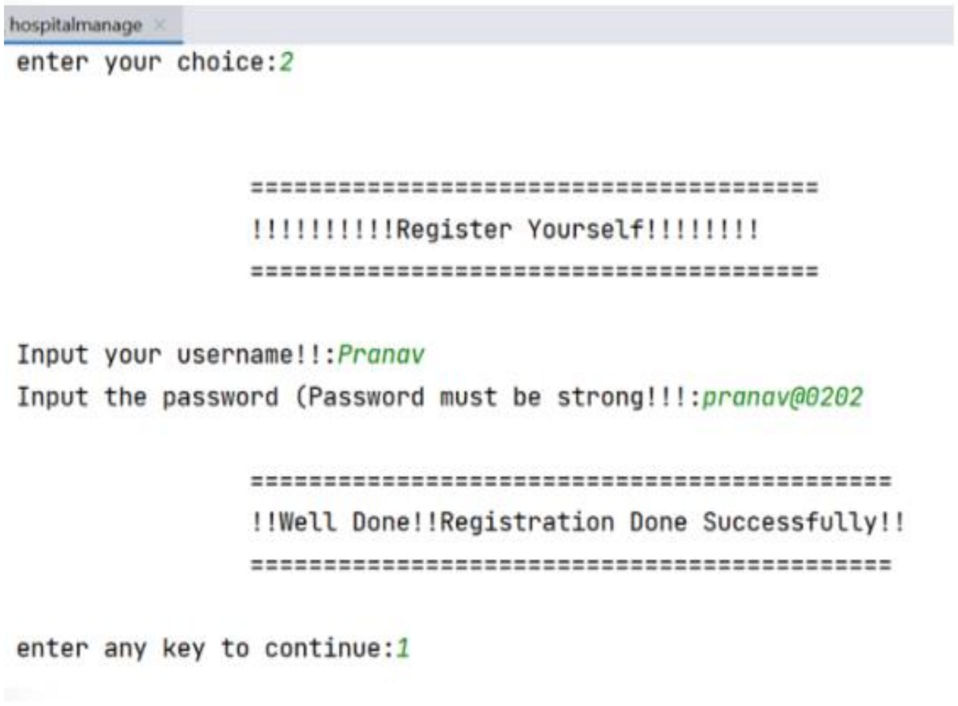
```
import asyncio
import websockets

async def send_messages():
    async with websockets.connect('ws://localhost:8765') as websocket:
        for i in range(10):
            message = f"Hello {i}"
            print(f"Sending: {message}")
            await websocket.send(message)
            response = await websocket.recv()
            print(f"Received: {response}")

asyncio.get_event_loop().run_until_complete(send_messages())
```

No problems have been detected in the workspace.

FINAL OUTPUT:

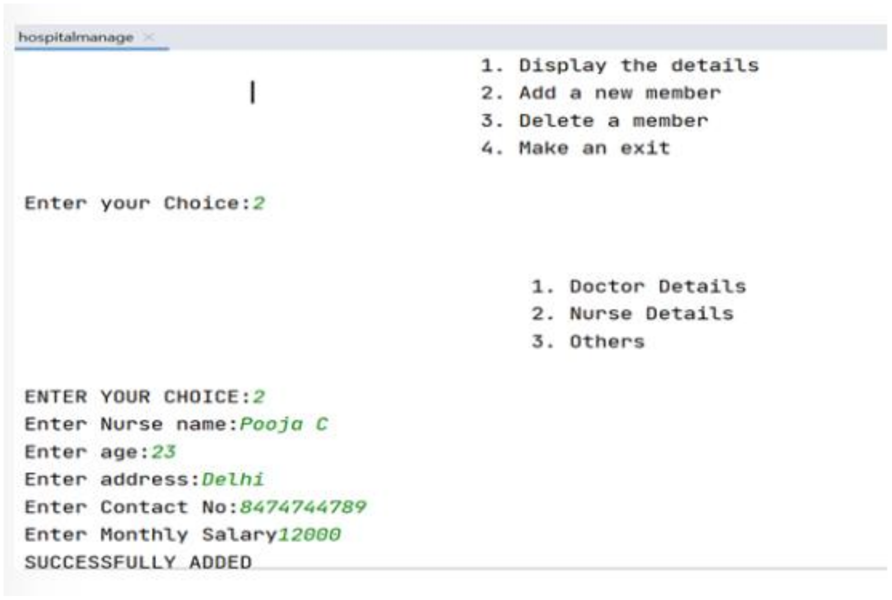


```
          ===================================
             Welcome To CityHospital
          ===================================


Enter the Password Please!!:

                        1. Sign In
                        2. Registration


enter your choice:
```



hospitalmanage

```
enter your choice:2


                ========================================
                !!!!!!!!!!Register Yourself!!!!!!!!
                ========================================


Input your username!!:Pranav
Input the password (Password must be strong!!!:pranav@0202

                =============================================
                !!Well Done!!Registration Done Successfully!!
                =============================================


enter any key to continue:1
```

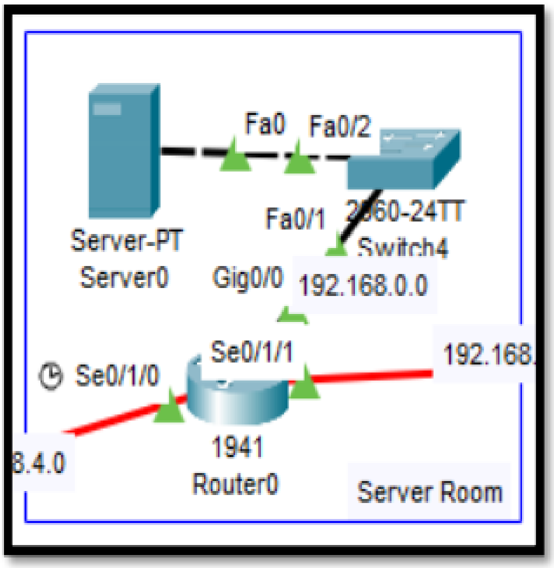


```
hospitalmanage
                                        1. Display the details
                                        2. Add a new member
                                        3. Delete a member
                                        4. Make an exit

Enter your Choice:2

                                           1. Doctor Details
                                           2. Nurse Details
                                           3. Others

ENTER YOUR CHOICE:2
Enter Nurse name:Pooja C
Enter age:23
Enter address:Delhi
Enter Contact No:8474744789
Enter Monthly Salary12000
SUCCESSFULLY ADDED
```

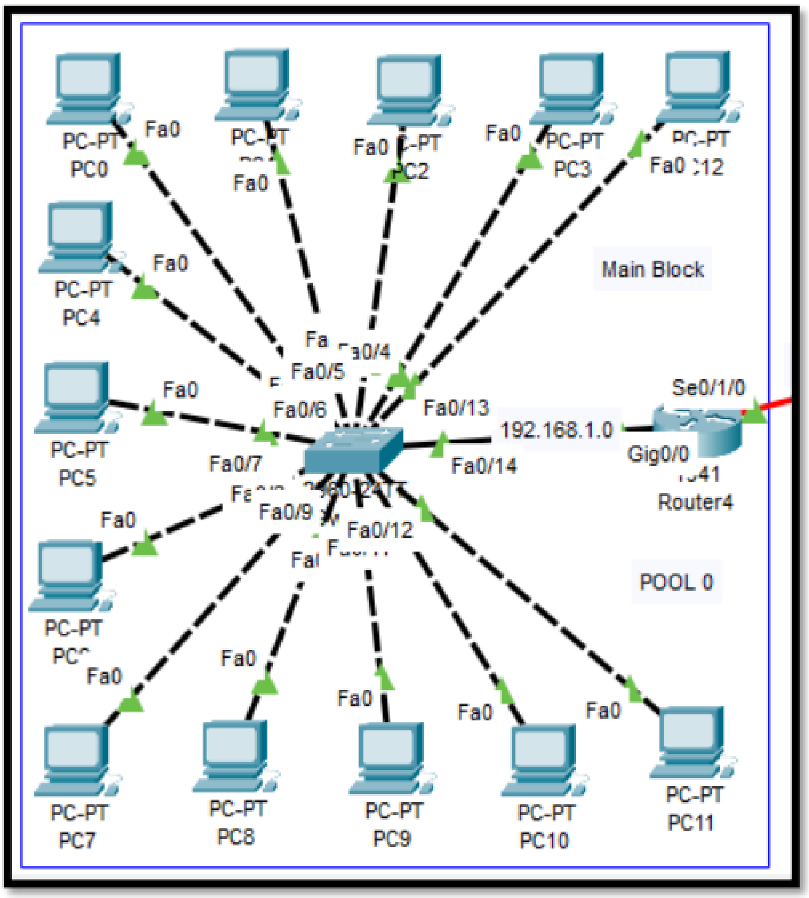
METHOD 2:

CISCO PACKET TRACER

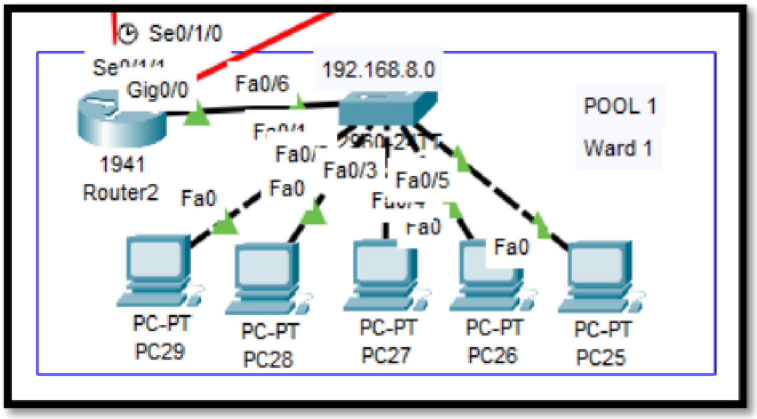
NETWORK TOPOLOGY DIAGRAM



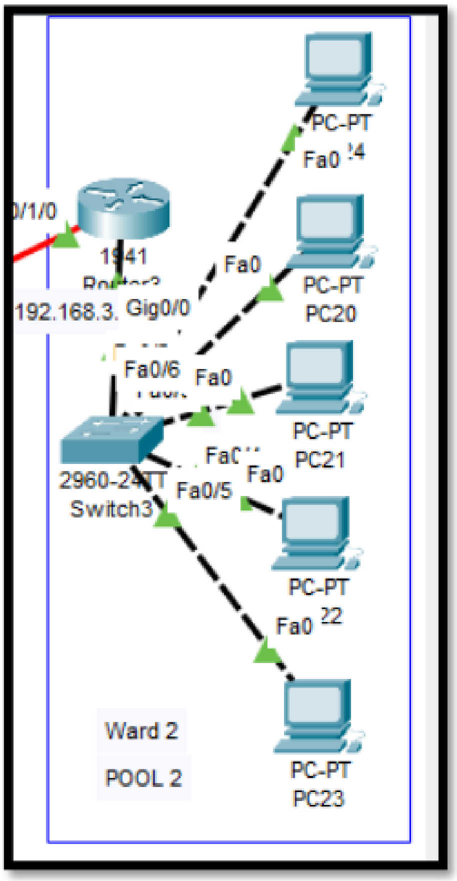
Server room



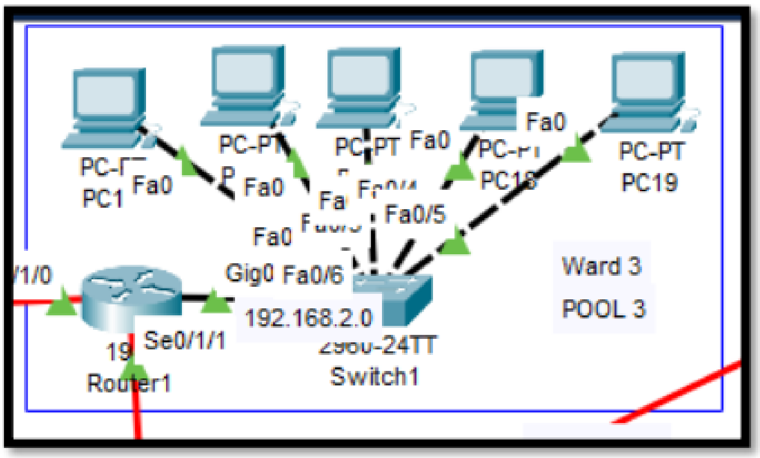
Main block



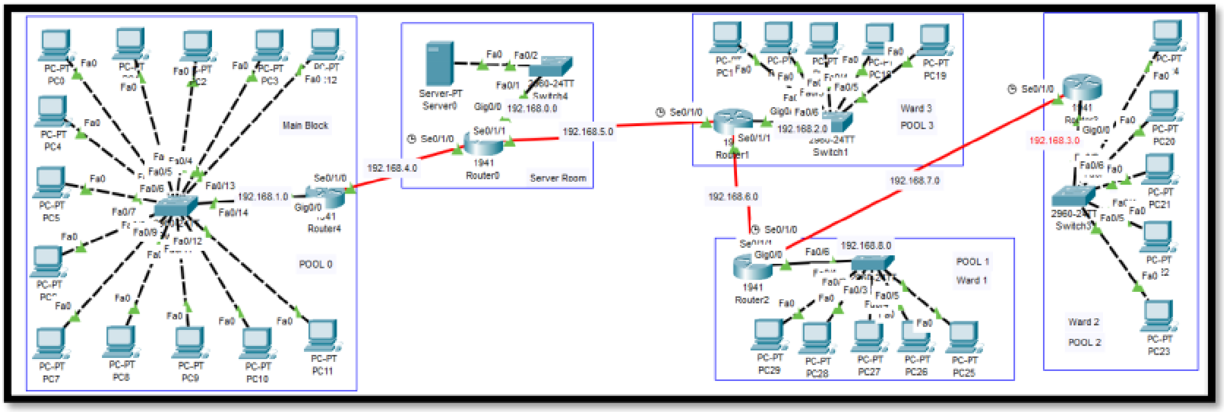
Ward 1


0/1/0
1941
Router3
192.168.3.
Gig0/0
Fa0
Fa0/6
Fa0
Fa0
2960-24TT
Switch3
Fa0/5
Fa0
PC-PT
PC24
PC-PT
PC20
PC-PT
PC21
PC-PT
PC22
PC-PT
PC23
Ward 2
POOL 2

Ward 2

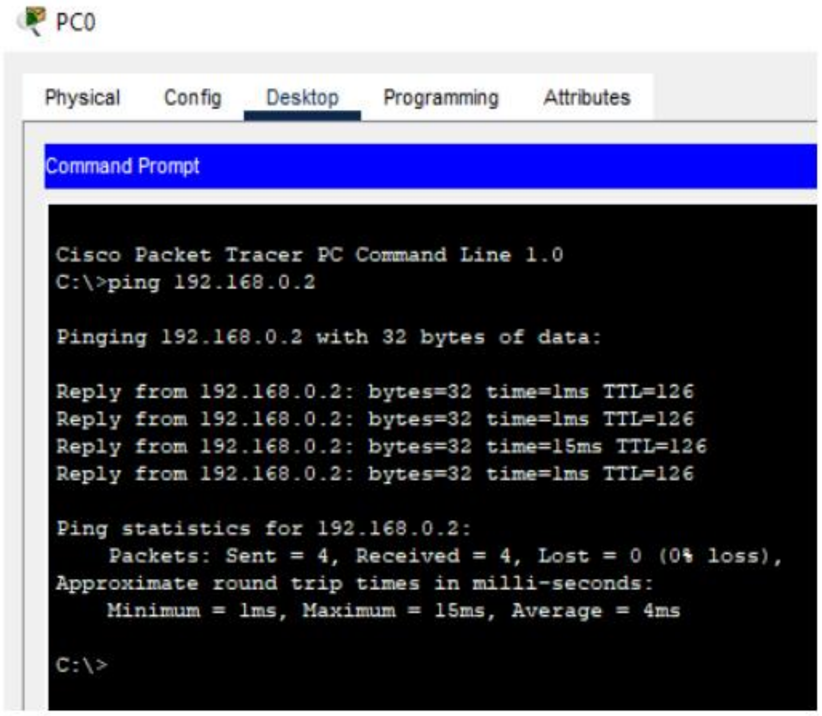

PC-PT
PC1
Fa0
PC-PT
Fa0
PC-PT
Fa0
PC-PT
PC18
Fa0
PC-PT
PC19
Fa0/5
Gig0
Fa0/6
192.168.2.0
2960-24TT
Switch1
Se0/1/1
1941
Router1
0/1/0
Ward 3
POOL 3

Ward 3



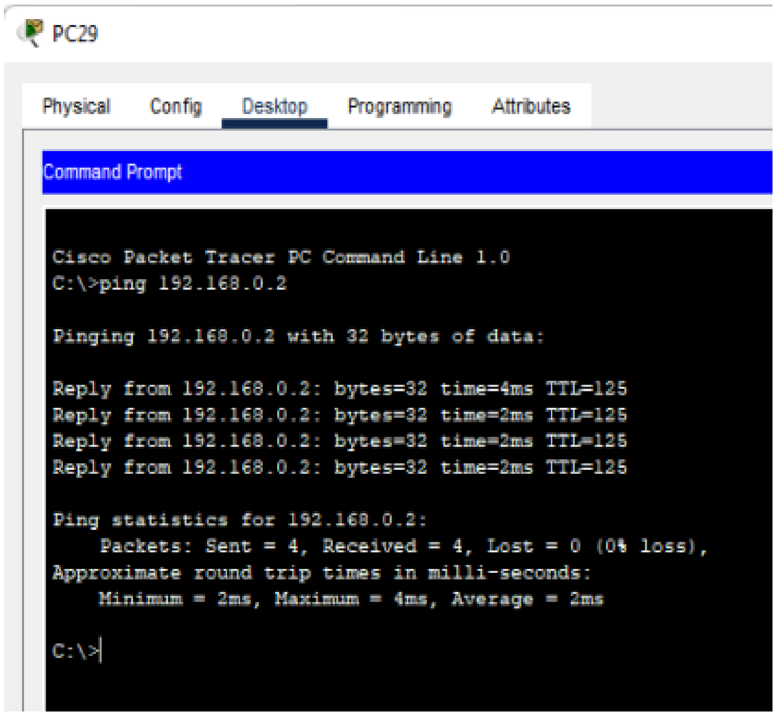
Hospital network

1. Pinging server from a pc of main block:



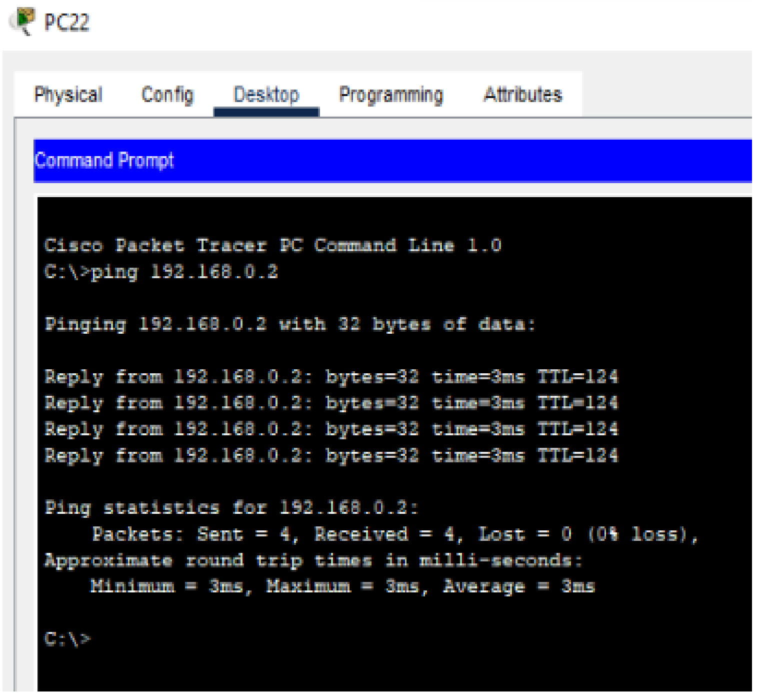
```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.0.2

Pinging 192.168.0.2 with 32 bytes of data:

Reply from 192.168.0.2: bytes=32 time=1ms TTL=126
Reply from 192.168.0.2: bytes=32 time=1ms TTL=126
Reply from 192.168.0.2: bytes=32 time=15ms TTL=126
Reply from 192.168.0.2: bytes=32 time=1ms TTL=126

Ping statistics for 192.168.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 15ms, Average = 4ms

C:\>
```

2. Pinging server from a pc of ward 1:



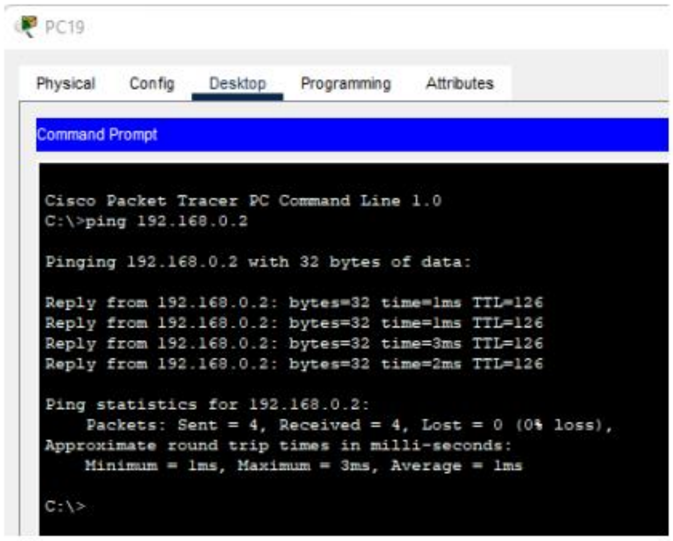
```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.0.2

Pinging 192.168.0.2 with 32 bytes of data:

Reply from 192.168.0.2: bytes=32 time=4ms TTL=125
Reply from 192.168.0.2: bytes=32 time=2ms TTL=125
Reply from 192.168.0.2: bytes=32 time=2ms TTL=125
Reply from 192.168.0.2: bytes=32 time=2ms TTL=125

Ping statistics for 192.168.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 4ms, Average = 2ms

C:\>
```

3. Pinging server from a pc of ward 2:



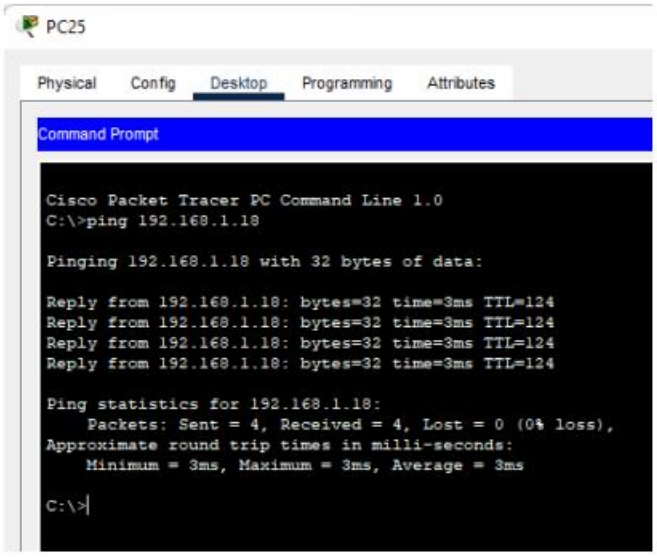
```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.0.2

Pinging 192.168.0.2 with 32 bytes of data:

Reply from 192.168.0.2: bytes=32 time=3ms TTL=124
Reply from 192.168.0.2: bytes=32 time=3ms TTL=124
Reply from 192.168.0.2: bytes=32 time=3ms TTL=124
Reply from 192.168.0.2: bytes=32 time=3ms TTL=124

Ping statistics for 192.168.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 3ms, Maximum = 3ms, Average = 3ms

C:\>
```

4.Pinging server from a pc of ward 3:



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.0.2

Pinging 192.168.0.2 with 32 bytes of data:

Reply from 192.168.0.2: bytes=32 time=1ms TTL=126
Reply from 192.168.0.2: bytes=32 time=1ms TTL=126
Reply from 192.168.0.2: bytes=32 time=3ms TTL=126
Reply from 192.168.0.2: bytes=32 time=2ms TTL=126

Ping statistics for 192.168.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 3ms, Average = 1ms

C:\>
```

5. Pinging PC0 (main block) from a PC25 (ward 1):



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.1.18

Pinging 192.168.1.18 with 32 bytes of data:

Reply from 192.168.1.18: bytes=32 time=3ms TTL=124
Reply from 192.168.1.18: bytes=32 time=3ms TTL=124
Reply from 192.168.1.18: bytes=32 time=3ms TTL=124
Reply from 192.168.1.18: bytes=32 time=3ms TTL=124

Ping statistics for 192.168.1.18:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 3ms, Maximum = 3ms, Average = 3ms

C:\>
```

Our objective is to check if every computer is able to access the hospital management software from each location. Therefore, we have checked that if one pc from the main block and each block is able to ping the server or not, i.e, data packets are sent and the reachability of a host on a network is tested. Also, the communication between wards and main block is tested by pinging them. Above are some of the screenshots.

4.2DISCUSSION OF RESULTS

Taking into account all the mentioned details, we can make the conclusion that the hospital management system is the inevitable part of the lifecycle of the modern medical institution. It automates numerous daily operations and enables smooth interactions of the users. Developing the hospital system software is a great opportunity to create the distinct, efficient and fast delivering healthcare model. Implementation of hospital management system project helps to store all the kinds of records, provide coordination and user communication, implement policies, improve day-to-day operations, arrange the supply chain, manage financial and human resources, and market hospital services. This beneficial decision covers the needs of the patients, staff and hospital authorities and simplifies their interactions. It has become the usual approach to manage the hospital. Many clinics have already experienced its advantages and continue developing new hospital management system project modules

CHAPTER 5

CONCLUSION

Working on the project was an excellent experience. It helped us to understand the importance of planning, designing and implementation so far we have learnt in our theory books. It helped us unleashing our creativity while working in a team. It also realized the importance of team working, communication as a part of this project. The project was successfully completed after a lot of efforts and work hours. This project underwent number of compiling, debugging, removing errors, making it bug free, adding more facilities in Hospital Management System and interactivity making it more reliable and useful. This project focused that scheduling a project and adhering to that schedule creates a hard sense of time- management. It has also let us known that co-operative teamwork always produce effective results. The entire project has been developed and deployed as per the requirements stated by the user. It is found to be bug free as per the testing.

There are also few features which can be integrated with this system to make it more flexible. Below list shows the future points to be consider : • Getting the current status of patient. • Including a different module for pharmacy, LAB, Bed Allotment and many more. • Including a Frequently Asked Questions Section. Finally, we like to conclude that we put all our efforts throughout the development of our project. .

REFERENCES

Python or Node.js: Common server-side programming languages for building the WebSocket server and handling business logic.

Socket.IO: A popular library for implementing WebSocket-based real-time applications, often used with Node.js.