

**S.I.E.S College of Arts, Science and Commerce(Autonomous)**  
**Sion(W), Mumbai – 400 022.**

**CERTIFICATE**

This is to certify that MRs. **Sheikh Mohammad Shariq Aslam** Roll No. **TCS2324075** has successfully completed the necessary course of experiments in the subject of **Wireless Sensor Networks & Mobile Communication** during the academic year **2023 – 2024** complying with the requirements of **University of Mumbai**, for the course of **TYBSc Computer Science [Semester-VI]**.

Prof. In-Charge  
**JESICA D'CRUZ**

Examination date:

Examiner's Signature & Date:

Head of the Department

College Seal

Name of Instructor: Ms. Jessica D'cruz

Sr. No	Description	Date	Faculty Signature
1	Understanding the Sensor Node Hardware	08th January 2024	
2	A) Exploring and understanding TinyOS computational concepts B) Understanding Tossim	08th January 2024	
3	A) Design Smoke detection and fire prevention system using cisco packet tracer B) Design Smoke detection and fire prevention system using cisco packet tracer	08th January 2024 15th January 2024	
4	Understanding, Reading and Analyzing Routing Table of network	05th January 2024	
5	Design Smoke detection and fire prevention system using cisco packet tracer	29th February 2024	
6		05th February 2024	
7	Create Mac Protocol simulation implementation for wireless sensor networks	06 <sup>th</sup> February 2024	
8	Simulate Mobile Adhoc Network with Directional Antenna	06 <sup>th</sup> February 2024	
9	Create a mobile network using Cell Tower, Central Office Server, Web browser and Web Server. Simulate connection between them	06 <sup>th</sup> February 2024	

**Prof. Manoj Singh**



## Wireless Sensor Networks & Mobile Communication

### Practical No.1

### DEPARTMENT OF COMPUTER SCIENCE

<b>Name:</b>	Shariq Sheikh	<b>Roll Number</b>	TCS2324075
<b>Paper Code:</b>	SIUSCS61	<b>Class</b>	TYBSc(Computer Science)
<b>Topic:</b>	Sensor Node Hardware	<b>Batch</b>	II
<b>Date:</b>	08th January 2024	<b>Practical No</b>	1

#### A) AIM:

### Understanding the Sensor Node Hardware

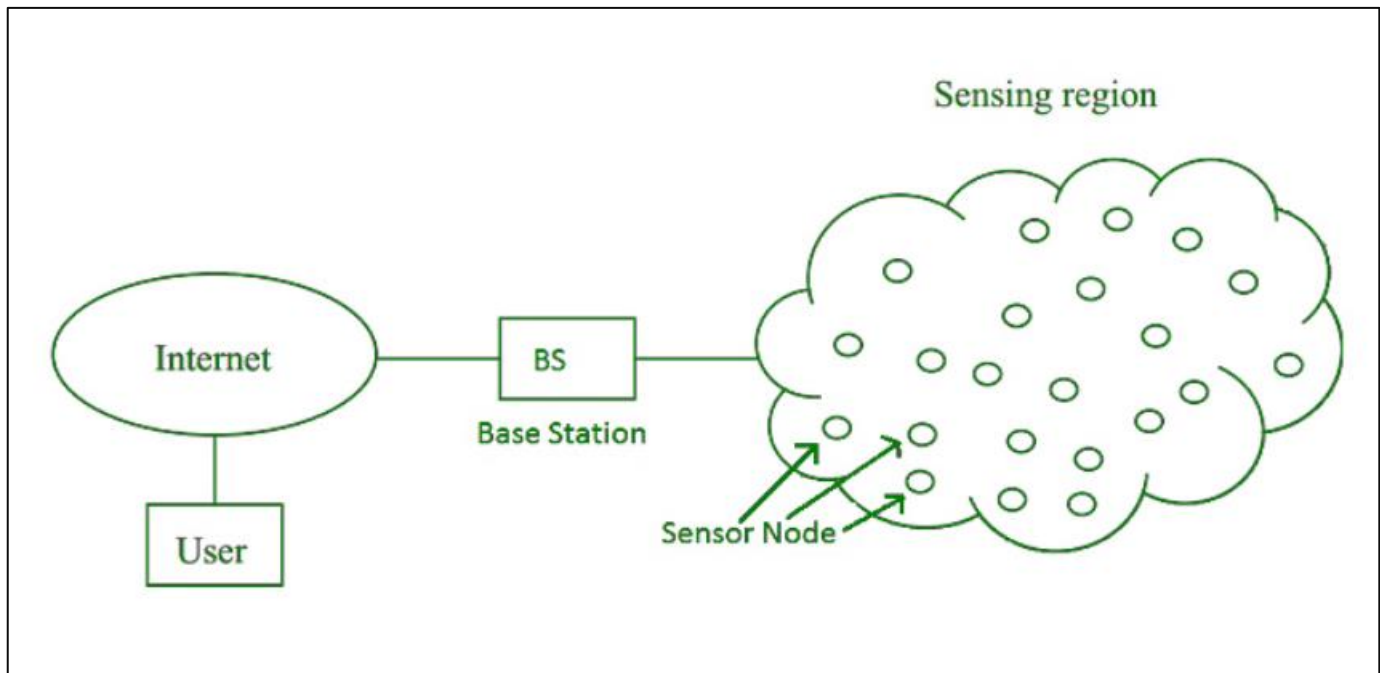
#### B) DESCRIPTION:

**Wireless Sensor Network (WSN)** is an infrastructure-less wireless network that is deployed in a large number of wireless sensors in an ad-hoc manner that is used to monitor the system, physical or environmental conditions.

Sensor nodes are used in WSN with the onboard processor that manages and monitors the environment in a particular area. They are connected to the Base Station which acts as a processing unit in the WSN System.

Base Station in a WSN System is connected through the Internet to share data.

### C) NETWORK TOPOLOGY, CONFIGURATIONS AND OUTPUT:



#### Hardware of WSN:

##### 1. Sensors:

Sensors in WSN are used to capture the environmental variables and which is used for data acquisition. Sensor signals are converted into electrical signals.

These components are responsible for data acquisition, i.e., they collect environmental data (variables) and convert it into electrical signals through a process known as transduction. In lay terms, sensors are tiny electric noses, ears, and fingers that 'feel' the environment and tell a computer what it 'senses' in a language it understands.

##### 2. Sensor Nodes:

A sensor node is a combination of different subunits and all they help to perform the functionality of the sensor node. Different units help to sense, record, monitor and analyze the data which is collected from physical conditions. Even though the name, a Sensor Node comprises not only the sensing component but also other important characteristics like processing of recorded data, communication with servers and storage units to store recorded data. With all these characteristics, components and enrichments, a Sensor Node takes responsibility for data collection, data correlation, and fusion of data from other sensors with its own data and network analysis.

### **3. Radio Nodes:**

These components are equipped with a microcontroller for data processing, a transceiver for wireless communication, external memory for data storage, and a power source to remain operational. They receive the sensor's electrical signals and send this data to the WLAN access point.

### **4. Wireless Access Point:**

A Wireless Access Point (WAP) is a networking device that allows connecting the devices with the wired network. A Wireless Access Point (WAP) is used to create the WLAN (Wireless Local Area Network), it is commonly used in large offices and buildings which have expanded businesses.

It is easier and simpler to understand and implant the device. It can be fixed, mobile or hybrid proliferated in the 21st century. The availability, confidentiality, and integrity of the communication and network are a responsibility and to be ensured about that.

A wireless AP connects the wired networks to the wireless client. It eases access to the network for mobile users which increases productivity and reduces the infrastructure cost.

### **5. WLAN Access Point:**

It receives the data which is sent by the Radio nodes wirelessly, generally through the internet.

### **6. Evaluation Software:**

The data received by the WLAN Access Point is processed by a software called as Evaluation Software for presenting the report to the users for further processing of the data which can be used for processing, analysis, storage, and mining of the data.

### **7. Base Station:**

A base station serves as a central connection point for a wireless device to communicate. It further connects the device to other networks or devices, usually through dedicated high bandwidth wire or fiber optic connections.

The base station sends commands to the sensor nodes and the sensor node perform the task by collaborating with each other. After collecting the necessary data, the sensor nodes send the data back to the base station.

A base station also acts as a gateway to other networks through the internet. After receiving the data from the sensor nodes, a base station performs simple data processing and sends the updated information to the user using internet.

If each sensor node is connected to the base station, it is known as Single-hop network architecture. Although long distance transmission is possible, the energy consumption for communication will be significantly higher than data collection and computation.

8. **Graphical User Interface:**

A graphical user interface (GUI) is a digital interface in which a user interacts with graphical components such as icons, buttons, and menus.

9. **Actuators:**

Actuators allow a WSN node to influence its environment, providing a feedback channel through which its decisions can be enacted. Throughout the history of computing, there has been a trend for the ratio of processing elements to people to increase, resulting in the creation and popularization of new usage paradigms.



# Wireless Sensor Networks & Mobile Communication

## Practical No.2

### DEPARTMENT OF COMPUTER SCIENCE

<b>Name:</b>	Shariq Sheikh	<b>Roll Number</b>	TCS2324075
<b>Paper Code:</b>	SIUSCS61	<b>Class</b>	TYBSc(Computer Science)
<b>Topic:</b>	TinyOs and Tossim	<b>Batch</b>	II
<b>Date:</b>	08th January 2024	<b>Practical No</b>	2

#### A) AIM:

##### 1) Exploring and understanding TinyOS computational concepts: Events, Commands and Task

- nesC model
- nesC Components

##### 2) Understanding TOSSIM for

- Mote-mote radio communication
- Mote-PC serial communication

#### B) DESCRIPTION:

**TinyOS** is an embedded, component-based operating system and platform for low-power wireless devices, such as those used in wireless sensor networks (WSNs), smartdust, ubiquitous computing, personal area networks, building automation, and smart meters. It is written in the programming language nesC, as a set of cooperating tasks and processes.

TOSSIM is a discrete event emulator for the execution of nesC model on TinyOS-Mica hardware [1]. In TOSSIM, an event is generated for each transmitted or received bit or packet.

## C) NETWORK TOPOLOGY, CONFIGURATIONS AND OUTPUT:

### A) Understanding TinyOS computational concepts: Events, Commands, Task, nesC components and model

A TinyOS program is a graph of components, each of which is an independent computational entity that exposes one or more interfaces. Components have three computational abstractions: commands, events, and tasks. Commands and events are mechanisms for inter-component communication, while tasks are used to express intra-component concurrency

#### Events:

An event in TinyOS is essentially a named placeholder for a function. It represents an occurrence or trigger that can be handled by the system. In TinyOS, events are a fundamental concept used for asynchronous programming. They provide a mechanism for handling and responding to external stimuli, such as sensor readings or incoming messages, in a non-blocking manner. Events are central to the event-driven programming paradigm in TinyOS, and they play a crucial role in managing the flow of execution in the system.

#### Commands:

A command is a named function or operation that a component provides to other components. It defines a specific action that can be requested by a client component. In TinyOS, commands are a mechanism for providing a standardized interface for components to interact with each other. Commands allow one component to request specific actions or services from another component in a well-defined manner. They are part of the component-based programming model used in TinyOS.

#### Tasks:

A task in TinyOS is a named function or operation that represents a unit of work to be performed. Tasks are used to structure the execution flow of the program and handle concurrent activities. A task in TinyOS is a named function or operation that represents a unit of work to be performed. Tasks are used to structure the execution flow of the program and handle concurrent activities.

#### NesC Components:

In TinyOS, components are a fundamental building block of the programming model. Components are modular units of code that encapsulate functionality, providing a way to structure and organize the software. Components are written in the nesC (pronounced "nes-C") programming language, which is specifically designed for developing embedded systems and wireless sensor networks. Here are key points about nesC components in TinyOS:

1. **Definition:** A nesC component is a self-contained unit of code that encapsulates a specific functionality or service. Components in TinyOS are designed to be modular and composable, promoting code reuse and maintainability.



2. **Interface-Implementation Separation:** Components in TinyOS follow the interface-implementation separation principle. An interface declares a set of commands, events, and functions that a component provides or expects from other components. The implementation contains the actual code that realizes the behavior defined in the interface.
3. **Interfaces:** Interfaces define the communication contract between components. They specify a set of commands, events, and functions that components can use to interact with each other. Interfaces allow for the decoupling of components and enable the development of interchangeable and interoperable software modules.
4. **Component Wiring:** Components can be connected or "wired" together to form an application. Wiring is the process of specifying how components are connected, determining how they communicate and collaborate. This wiring is often done in the configuration file of a TinyOS application.
5. **Configuration:** Components can be parameterized and configured, allowing for flexibility in adapting their behavior to different requirements. Configuration settings can be adjusted at compile time, providing a way to customize the behavior of the software.

## B) Understanding TOSSIM for

### - Mote-mote radio communication:

Mote to mote communication is the radio communication in Tiny os. This introduces us the interfaces and components in Tiny os which supports the radio communication. And also we learn the basics how to use the message\_ t that is a message buffer which is used to send the message buffer to the radio and receives the message buffer from the radio. Tiny os provides us with the interfaces and the components.

Interfaces are used to consider the existing communication services and the components are used to implement the interfaces. These components and interfaces use a message buffer called message\_ t that is implemented as nesC structure. This message buffer message\_ t was used as TOS\_ Msg in the first version of Tiny os and in the latest version of it has been replaced as message\_ t. In the first version of the tiny os the message buffers were accessed directly but in the latest version they cannot be accessed directly instead this function can be read and written in the form of mutator and accessor functions.

## **- Mote-PC serial communication**

Mote-to-PC radio communication is a crucial aspect of wireless sensor networks (WSNs), facilitating the transfer of data between sensor nodes (motes) deployed in the field and a central personal computer (PC). This communication enables the collection, monitoring, and analysis of data generated by the sensor nodes. Here's a brief overview:

### **1. Hardware Setup:**

- Motes are equipped with radio transceivers for wireless communication, often following standards like IEEE 802.15.4 or Zigbee.
- The PC may have a compatible radio interface or a dedicated gateway device to communicate with the sensor nodes.

### **2. Communication Protocols:**

- Motes and the PC use compatible communication protocols, defining how data is formatted, transmitted, and received.
- Protocols may include addressing, synchronization, and error-handling mechanisms.

### **3. Data Transmission:**

- Motes periodically sense the environment, collect data, and transmit it wirelessly to the PC or through an intermediary gateway.
- Data may include sensor readings, status information, or event notifications.

### **4. Gateway Device:**

- A gateway device may be employed to interface between the wireless sensor network and the PC. It receives data from sensor nodes, processes or aggregates it, and forwards it to the PC.

### **5. PC Software:**

- The PC runs software responsible for receiving, parsing, and processing data from the sensor nodes.
- Communication drivers, middleware, and visualization components are commonly part of the software stack.

### **6. Security Measures:**

- Security measures, such as encryption and authentication, are often implemented to protect the communication between sensor nodes and the PC, ensuring data integrity and preventing unauthorized access.

### **7. Data Analysis and Visualization:**

- Once received by the PC, the data can be analyzed, visualized, and used for decision-making.
- Visualization tools, databases, and analysis software help make sense of the collected data.

### **8. Application Examples:**

- Mote-to-PC communication is used in various applications, including environmental monitoring, industrial automation, healthcare, and smart agriculture.
- It enables real-time tracking, control, and management of distributed sensor nodes.

### **9. Energy Considerations:**

- Energy-efficient communication strategies, such as duty cycling and data aggregation, are often employed in mote-to-PC communication to maximize the lifespan of sensor nodes.

In summary, mote-to-PC radio communication is a foundational element in WSNs, enabling the seamless exchange of data between sensor nodes and central processing units for analysis, monitoring, and decision-making in diverse applications.



# Wireless Sensor Networks & Mobile Communication

## Practical No. 3A

### DEPARTMENT OF COMPUTER SCIENCE

<b>Name:</b>	Shariq Sheikh	<b>Roll Number</b>	TCS2324075
<b>Paper Code:</b>	SIUSCS61	<b>Class</b>	TYBSc(Computer Science)
<b>Topic:</b>	Smoke detection and fire prevention system	<b>Batch</b>	II
<b>Date:</b>	08th January 2024	<b>Practical No</b>	3A

**A) AIM: Design Smoke detection and fire prevention system using cisco packet tracer**

#### **B) DESCRIPTION:**

Sensor is a device that is used to gather information about a physical process or a physical phenomenon and translate it into electrical signals that can be processed, measured and analysed. The term physical process Sensor can be any real-world information like temperature, pressure, light, sound, motion, position, flow, humidity, radiation etc. **Here, Smoke Detector is a sensor.**

Actuator is a device that converts the electrical signals into the physical events or characteristics. It takes the input from the system and gives output to the environment. **Here, FireSprinkler and LawnSprinkler is the Actuator.**

## C) CODE AND OUTPUT:

192.168.10.1

Physical Config Services **Desktop** Programming Attributes

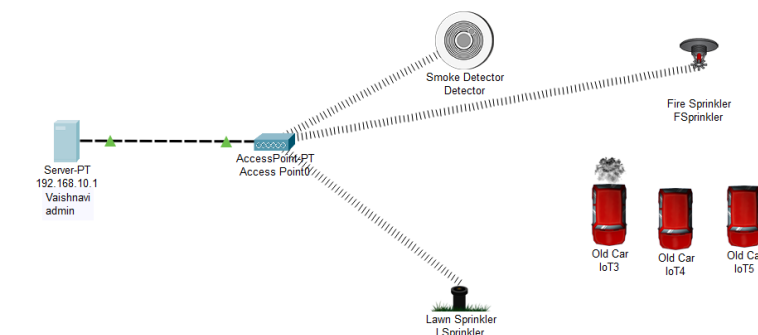
Web Browser X

< > URL  Go Stop

IoT Server - Device Conditions [Home](#) | [Conditions](#) | [Editor](#) | [Log Out](#)

Actions		Enabled	Name	Condition	Actions
<a href="#">Edit</a>	<a href="#">Remove</a>	Yes	SprinklerOn	Detector Level > 0.5	Set LSprinkler Status to true
<a href="#">Edit</a>	<a href="#">Remove</a>	Yes	SpinklerOf	Detector Level < 0.5	Set LSprinkler Status to false
<a href="#">Edit</a>	<a href="#">Remove</a>	Yes	FireSprinkler	Detector Level > 0.5	Set FSprinkler Status to true
<a href="#">Edit</a>	<a href="#">Remove</a>	Yes	frsprinkler	Detector Level < 0.5	Set FSprinkler Status to false

[Add](#)



192.168.10.1

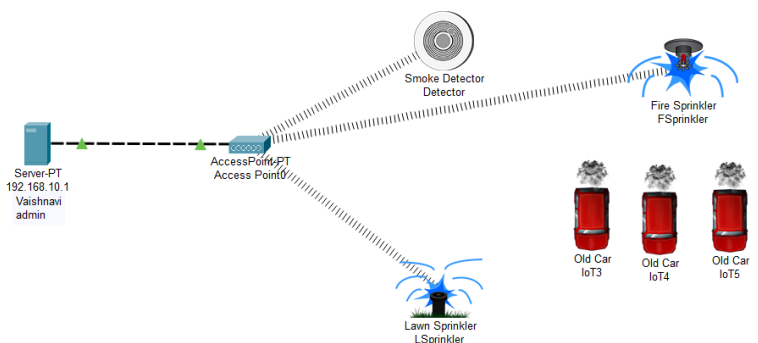
Physical Config Services **Desktop** Programming Attributes

Web Browser X

< > URL  Go Stop

IoT Server - Devices [Home](#) | [Conditions](#) | [Editor](#) | [Log Out](#)

- Detector (PTT0810F80N-) Smoke Detector
  - Alarm Level: 0.18189
- LSprinkler (PTT0810X7IB-) Lawn Sprinkler
  - Status: ■
- FSprinkler (PTT0810SEU9-) Fire Sprinkler
  - Status: ■



192.168.10.1

Physical Config Services **Desktop** Programming Attributes

Web Browser X

< > URL  Go Stop

IoT Server - Devices [Home](#) | [Conditions](#) | [Editor](#) | [Log Out](#)

- Detector (PTT0810F80N-) Smoke Detector
  - Alarm Level: 0.52187
- LSprinkler (PTT0810X7IB-) Lawn Sprinkler
  - Status: ■
- FSprinkler (PTT0810SEU9-) Fire Sprinkler
  - Status: ■

## Global settings same for all

Global Settings

Display NameLSprinkler

Serial NumberPTT0810X7IB-

InterfacesWireless0

Gateway/DNS IPv4

☐ DHCP

☒ Static

Default Gateway

DNS Server

Gateway/DNS IPv6

☒ Automatic

☐ Static

Default Gateway

DNS Server

IoT Server

☐ None

☐ Home Gateway

☒ Remote Server

Server Address192.168.10.1

User NameVaishnavi

Passwordadmin

Refresh

## LSprinkler

LSprinkler

SpecificationsPhysicalConfigAttributes

GLOBAL

Settings

Algorithm Settings

Files

INTERFACE

Wireless0

Bluetooth

Wireless0

Port Status

On

Bandwidth24 Mbps

MAC Address0009.7CE6.1575

SSIDCisco

Authentication

☒ Disabled

☐ WEP

WEP Key

☐ WPA-PSK

☐ WPA2-PSK

PSK Pass Phrase

☐ WPA

☐ WPA2

Password

☐ 802.1X

Method:MD5

User Name

Password

Encryption TypeDisabled

IP Configuration

☐ DHCP

☒ Static

IPv4 Address192.168.10.4

Subnet Mask255.255.255.0

IPv6 Configuration

☒ Automatic

☐ Static

IPv6 Address

Link Local AddressFE80::209:7CFF:FEE6:1575



# Wireless Sensor Networks & Mobile Communication

## Practical No. 3B

### DEPARTMENT OF COMPUTER SCIENCE

Name:	Shariq Sheikh	Roll Number	TCS2324075
Paper Code:	SIUSCS61	Class	TYBSc(Computer Science)
Topic:	Smart Garden System	Batch	II
Date:	15th January 2024	Practical No	3A

A) AIM: Design Smart Garden system using cisco packet tracer

#### B) DESCRIPTION:

Sensor is a device that is used to gather information about a physical process or a physical phenomenon and translate it into electrical signals that can be processed, measured and analysed. The term physical process Sensor can be any real-world information like temperature, pressure, light, sound, motion, position, flow, humidity, radiation etc. **Here, Water Level Monitor is a sensor.**

Actuator is a device that converts the electrical signals into the physical events or characteristics. It takes the input from the system and gives output to the environment. **Here, LawnSprinkler is the Actuator.**

#### C) CODE AND OUTPUT:

Smartphone0

Physical Config **Desktop** Programming Attributes

IoT Monitor X

IoT Server - Device Conditions Home | Conditions | Editor | Log Out

Actions		Enabled	Name	Condition	Actions
Edit	Remove	Yes	top_on	Monitor1 Water Level < 5.0 cm	Set LS1 Status to true
Edit	Remove	Yes	top_off	Monitor1 Water Level > 5.0 cm	Set LS1 Status to false
Edit	Remove	Yes	bottom_on	Monitor2 Water Level < 5.0 cm	Set LS2 Status to true
Edit	Remove	Yes	bottom_off	Monitor2 Water Level > 5.0 cm	Set LS2 Status to false

Add

Name of Instructor: Ms. Jessica D'cruz

## i) When water level goes below 5 cm : ( LawnSprinkler is On)

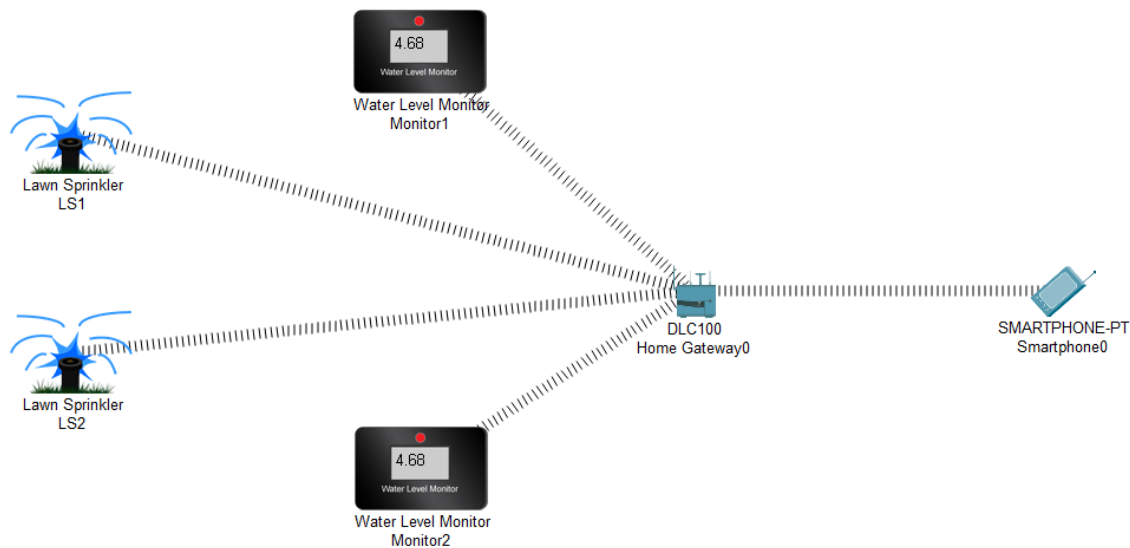
Smartphone0

Physical Config **Desktop** Programming Attributes

IoT Monitor X

IoT Server - Devices Home | Conditions | Editor | Log Out

LS1 (PTT0810DKMF-)	Lawn Sprinkler
Status	
Monitor1 (PTT0810Y6S0-)	Water Level Monitor
Water Level	3.6 cm
Monitor2 (PTT081066BZ-)	Water Level Monitor
Water Level	3.6 cm
LS2 (PTT08102T38-)	Lawn Sprinkler
Status	





## ii) when water level goes above 5cm : (LawnSprinkler is Off)

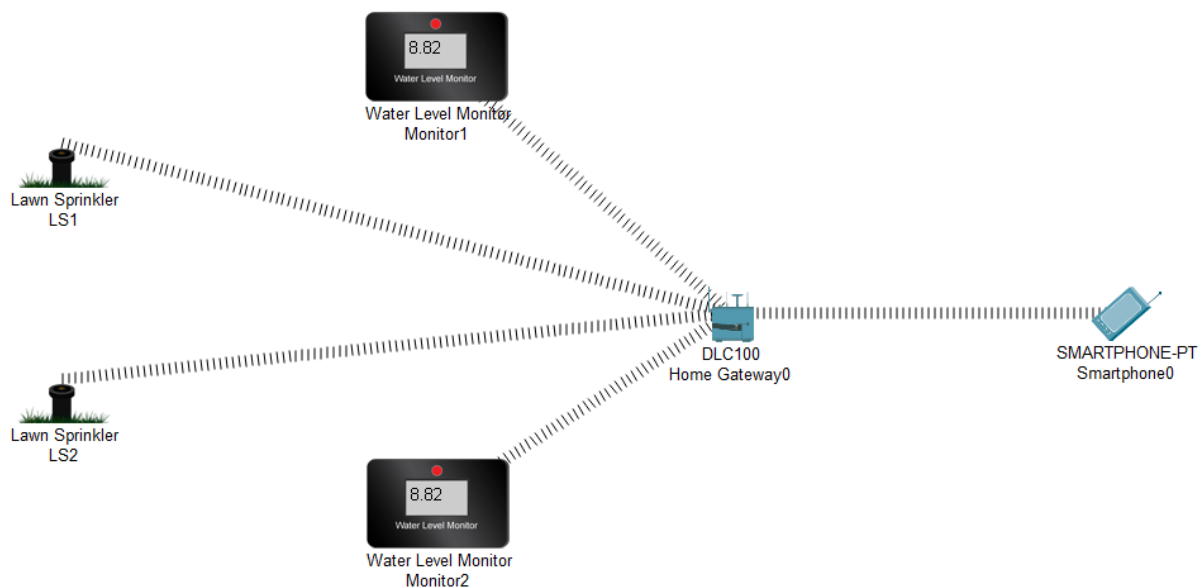
Smartphone0

Physical Config **Desktop** Programming Attributes

IoT Monitor X

IoT Server - Devices Home | Conditions | Editor | Log Out

LS1 (PTT0810DKMF-)	Lawn Sprinkler
Status	
Monitor1 (PTT0810Y6S0-)	Water Level Monitor
Water Level	8.1 cm
Monitor2 (PTT081066BZ-)	Water Level Monitor
Water Level	8.1 cm
LS2 (PTT08102T38-)	Lawn Sprinkler
Status	



Home Gateway0

Physical **Config** GUI Attributes

**GLOBAL**

Settings

Algorithm Settings

**INTERFACE**

Internet

LAN

Wireless

**Wireless Settings**

SSID: HomeGateway

2.4 GHz Channel: 6 - 2.437GHz

Coverage Range (meters): 250.00

Authentication:

☐ Disabled ☐ WEP ☒ WPA2-PSK ☐ WPA ☐ WPA2

WEP Key:

PSK Pass Phrase: Vaishnavi

**RADIUS Server Settings**

IP Address:

Shared Secret:

Encryption Type: AES

LS1

Specifications Physical **Config** Attributes

**GLOBAL**

Settings

Algorithm Settings

Files

**INTERFACE**

Wireless0

Bluetooth

**Wireless0**

Port Status: ☒ On

Bandwidth: 300 Mbps

MAC Address: 000B.BEDE.BEB2

SSID: HomeGateway

Authentication:

☐ Disabled ☐ WEP ☒ WPA2-PSK ☐ WPA ☐ WPA2 ☐ 802.1X

WEP Key:

PSK Pass Phrase: Vaishnavi

User ID:

Password:

Method: MD5

User Name:

Password:

Encryption Type: AES

LS2

Specifications
Physical
**Config**
Attributes

**GLOBAL**
Settings
Algorithm Settings
Files
**INTERFACE**
Wireless0
Bluetooth

Wireless0

Port Status ☒ On
Bandwidth 300 Mbps
MAC Address 0050.0FEE.B96A
SSID HomeGateway

Authentication
☐ Disabled
☐ WEP
WEP Key
☐ WPA-PSK
☒ WPA2-PSK
PSK Pass Phrase Vaishnavi
☐ WPA
☐ WPA2
User ID
Password
Method: MD5
User Name
Password
Encryption Type AES

Monitor1

Specifications
Physical
**Config**
Attributes

**GLOBAL**
Settings
Algorithm Settings
Files
**INTERFACE**
Wireless0
Bluetooth

Wireless0

Port Status ☒ On
Bandwidth 300 Mbps
MAC Address 000A.F3C4.0601
SSID HomeGateway

Authentication
☐ Disabled
☐ WEP
WEP Key
☐ WPA-PSK
☒ WPA2-PSK
PSK Pass Phrase Vaishnavi
☐ WPA
☐ WPA2
User ID
Password
Method: MD5
User Name
Password
Encryption Type AES

Name of Instructor: Ms. Jessica D'cruz

Monitor2

Specifications
Physical
**Config**
Attributes

**GLOBAL**
Settings
Algorithm Settings
Files
**INTERFACE**
Wireless0
Bluetooth

Wireless0

Port Status
☒ On

Bandwidth
300 Mbps

MAC Address
0001.423D.0DC4

SSID
HomeGateway

Authentication

☐ Disabled
☐ WEP
☐ WPA-PSK
☒ WPA2-PSK
☐ WPA
☐ 802.1X

Method:

MD5

WEP Key

PSK Pass Phrase

User ID

Password

User Name

Password

Vaishnavi

Encryption Type
AES

Smartphone0

Physical
**Config**
Desktop
Programming
Attributes

**GLOBAL**
Settings
Algorithm Settings
**INTERFACE**
Wireless0
3G/4G Cell1
Bluetooth

Wireless0

Port Status
☒ On

Bandwidth
300 Mbps

MAC Address
00E0.A38C.3880

SSID
HomeGateway

Authentication

☐ Disabled
☐ WEP
☐ WPA-PSK
☒ WPA2-PSK
☐ WPA
☐ 802.1X

Method:

MD5

WEP Key

PSK Pass Phrase

User ID

Password

User Name

Password

Vaishnavi

Encryption Type
AES



# Wireless Sensor Networks & Mobile Communication

## Practical No. 4

### DEPARTMENT OF COMPUTER SCIENCE

Name:	Shariq Sheikh	Roll Number	TCS2324075
Paper Code:	SIUSCS61	Class	TYBSc(Computer Science)
Topic:	Simple ADHOC Network	Batch	II
Date:	05th February 2024	Practical No	4

**A) AIM: Create and simulate A SIMPLE ADHOC Network (OMNET)**

#### B) DESCRIPTION:

- An ad hoc network is a temporary network created between two devices without utilizing any other networking infrastructure. These networks exist for a single session, often for a specific purpose like transferring files or sharing an Internet connection. Ad hoc networks do not require a router or wireless access point.

#### Code And Output :

```
package inet.examples.adhoc.Simple_adhoc;

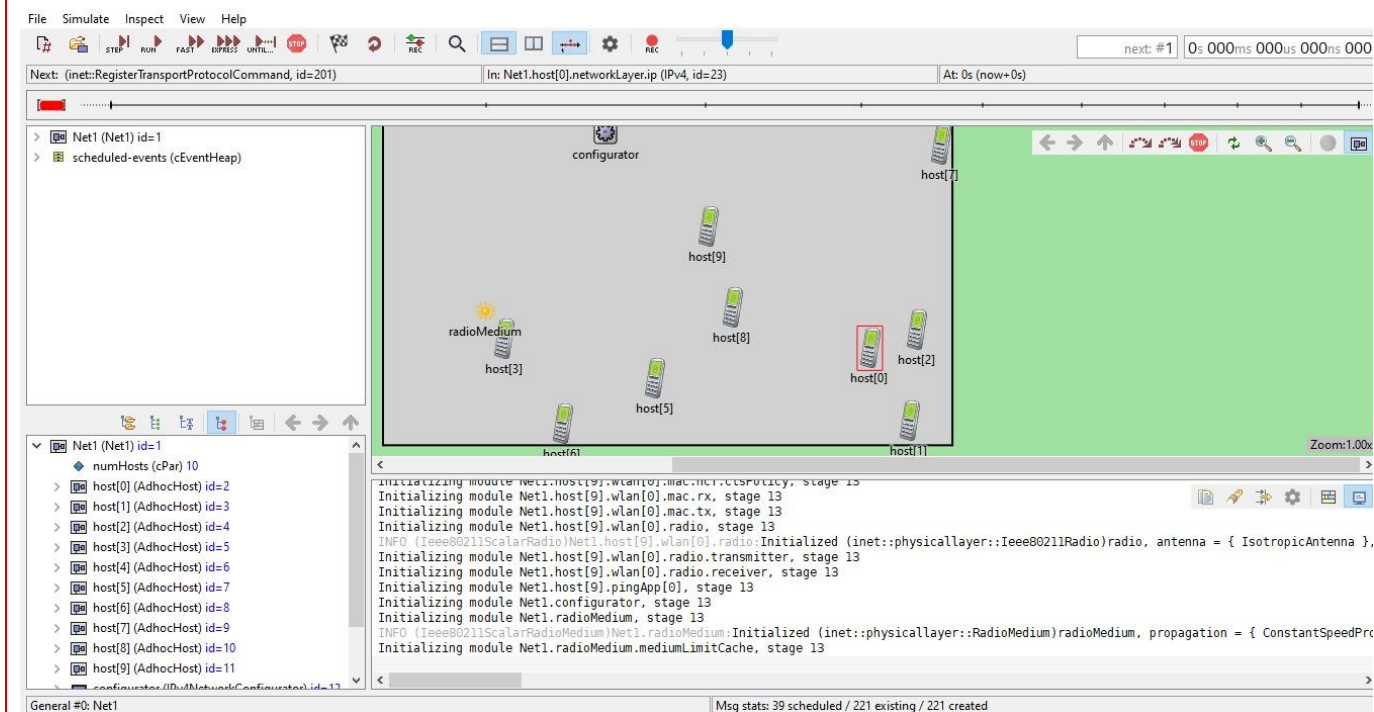
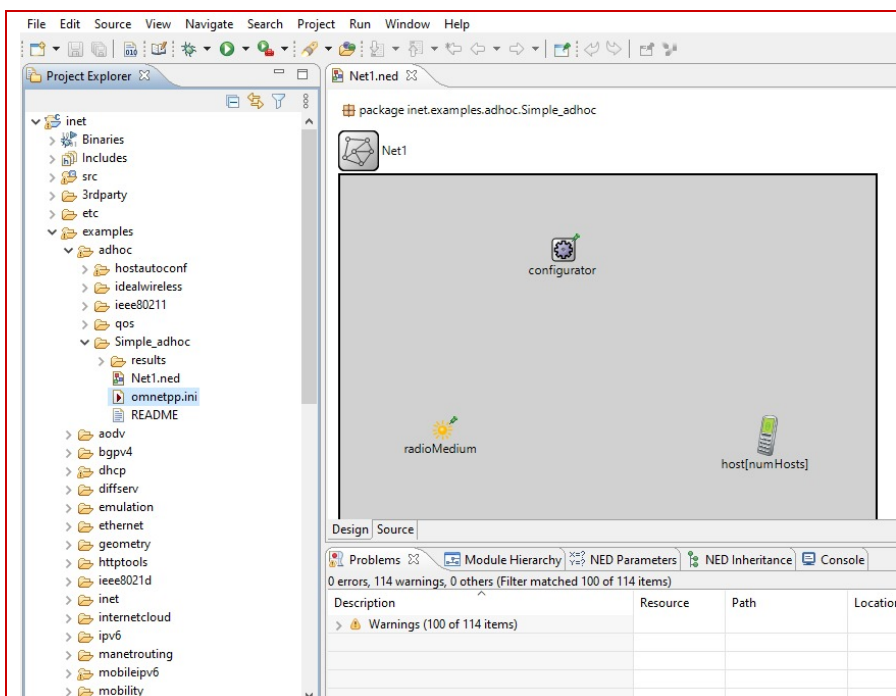
// numOfHosts: 10
// parametric: true
// static: false

import inet.networklayer.configurator.ipv4.IPv4NetworkConfigurator;
import inet.node.inet.AdhocHost;
import inet.physicallayer.ieee80211.packetlevel.Ieee80211ScalarRadioMedium;

network Net1
{
    parameters:
        int numHosts;
    submodules:
        host[numHosts]: AdhocHost {
            parameters:
                @display("r=, ,#707070");
        }

        configurator: IPv4NetworkConfigurator {
            @display("p=219,73");
        }

        radioMedium: Ieee80211ScalarRadioMedium {
            parameters:
                @display("p=100,250");
        }
}
```



OMNeT++/QtEnv (release) - General #0 - omnetpp.ini - C:\Users\cs18\Downloads\omnetpp-5.6.2\WSN\_18\inet-3.7.1-src\inet\examples\adhoc\Simple\_adhoc

File Simulate Inspect View Help

last: #2'007 3s 694ms 620us 376ns 603ps

Next: ping1 (inet:physicallayer::RadioFrame, id=2126) In: Net1.host[4].wlan[0].radio (lee80211ScalarRadio, id=359) At: 3.694620539791s (now+0.000000163188s)

[ ]

Net1 (Net1) id=1  
 > scheduled-events (cEventHeap)

numHosts (cPar) 10  
 > host[0] (AdhocHost) id=2  
 > host[1] (AdhocHost) id=3  
 > host[2] (AdhocHost) id=4  
 > host[3] (AdhocHost) id=5  
 > host[4] (AdhocHost) id=6  
 > host[5] (AdhocHost) id=7  
 > host[6] (AdhocHost) id=8  
 > host[7] (AdhocHost) id=9  
 > host[8] (AdhocHost) id=10  
 > host[9] (AdhocHost) id=11

Initializing module Net1.host[9].wlan[0].mac.hcf.edcaDataRecoveryProcedures[0], stage 13  
 Initializing module Net1.host[9].wlan[0].mac.hcf.edcaDataRecoveryProcedures[1], stage 13  
 Initializing module Net1.host[9].wlan[0].mac.hcf.edcaDataRecoveryProcedures[2], stage 13  
 Initializing module Net1.host[9].wlan[0].mac.hcf.edcaDataRecoveryProcedures[3], stage 13  
 Initializing module Net1.host[9].wlan[0].mac.hcf.edcaTxopProcedures[0], stage 13  
 Initializing module Net1.host[9].wlan[0].mac.hcf.edcaTxopProcedures[1], stage 13  
 Initializing module Net1.host[9].wlan[0].mac.hcf.edcaTxopProcedures[2], stage 13  
 Initializing module Net1.host[9].wlan[0].mac.hcf.edcaTxopProcedures[3], stage 13  
 Initializing module Net1.host[9].wlan[0].mac.hcf.originatorAckPolicy, stage 13  
 Initializing module Net1.host[9].wlan[0].mac.hcf.recipientAckPolicy, stage 13  
 Initializing module Net1.host[9].wlan[0].mac.hcf.rtsPolicy, stage 13  
 Initializing module Net1.host[9].wlan[0].mac.hcf.ctsPolicy, stage 13  
 Initializing module Net1.host[9].wlan[0].mac.rx, stage 13

General #0: Net1

Msg stats: 30 scheduled / 583 existing / 2144 created

Problems Module Hierarchy NED Parameters NED Inheritance Console

Simple\_adhoc [OMNeT++ Simulation] C:\Users\cs18\Downloads\omnetpp-5.6.2\bin\opp\_run.exe (2/16/24 4:46 PM - )

```

Net1.host[8].pingApp[0]: reply of 60 bytes from 10.0.0.1 icmp_seq=0 ttl=31 time=3.168548105 msec (ping0)
Net1.host[6].pingApp[0]: reply of 60 bytes from 10.0.0.1 icmp_seq=0 ttl=31 time=2.99103576 msec (ping0)
Net1.host[1].pingApp[0]: reply of 60 bytes from 10.0.0.1 icmp_seq=0 ttl=31 time=3.188090916 msec (ping0)
Net1.host[4].pingApp[0]: reply of 60 bytes from 10.0.0.1 icmp_seq=0 ttl=31 time=3.052688858 msec (ping0)
  
```



# Wireless Sensor Networks & Mobile Communication

## Practical No. 5

### DEPARTMENT OF COMPUTER SCIENCE

<b>Name:</b>	Shariq Sheikh	<b>Roll Number</b>	TCS2324075
<b>Paper Code:</b>	SIUSCS61	<b>Class</b>	TYBSc(Computer Science)
<b>Topic:</b>	OSPF and RIP Protocol	<b>Batch</b>	II
<b>Date:</b>	29th February 2024	<b>Practical No</b>	5

#### A) AIM: Understanding, Reading and Analyzing Routing Table of Network

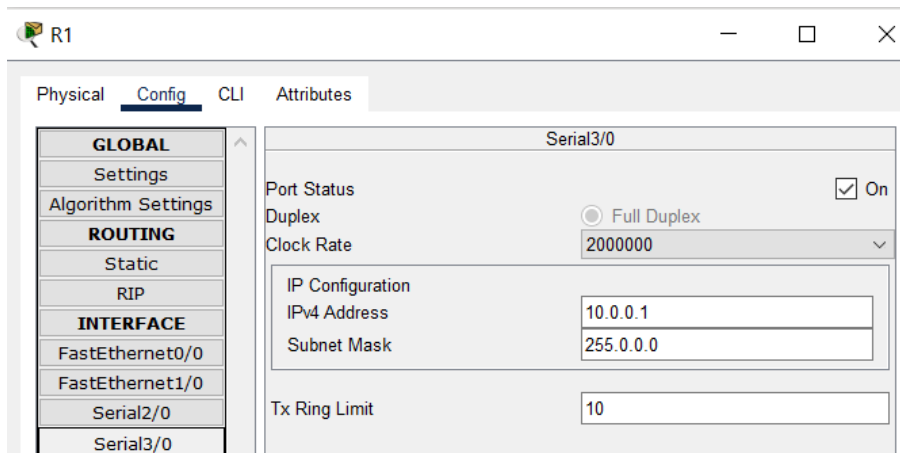
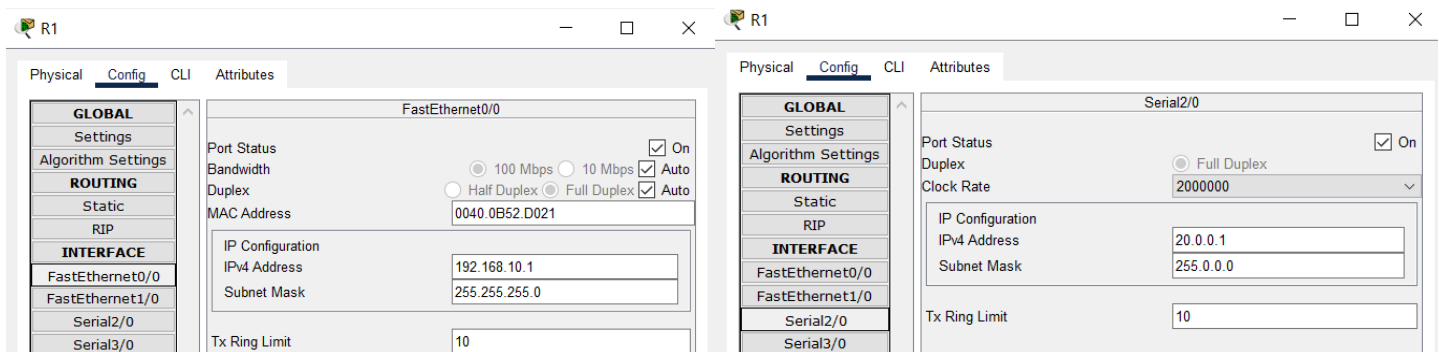
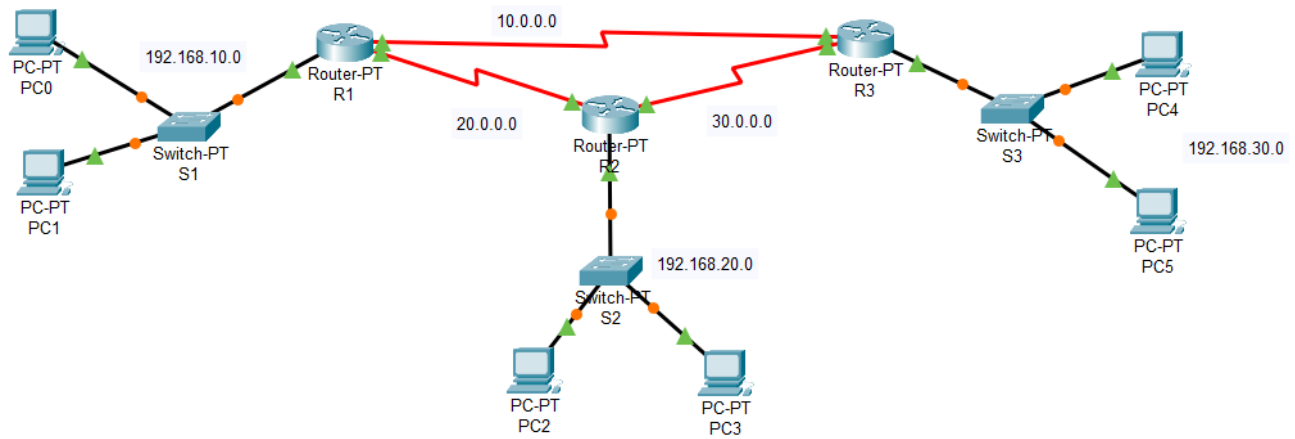
#### B) DESCRIPTION:

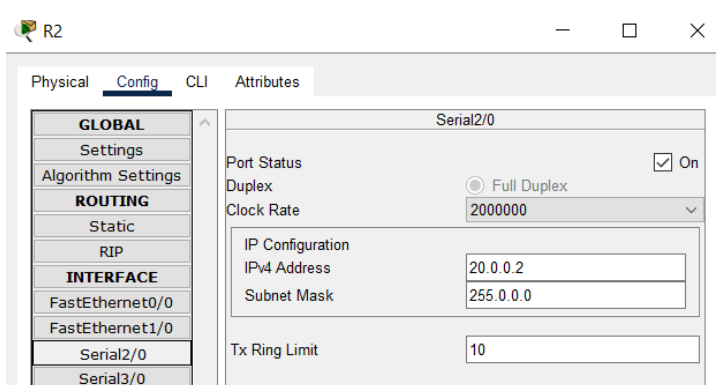
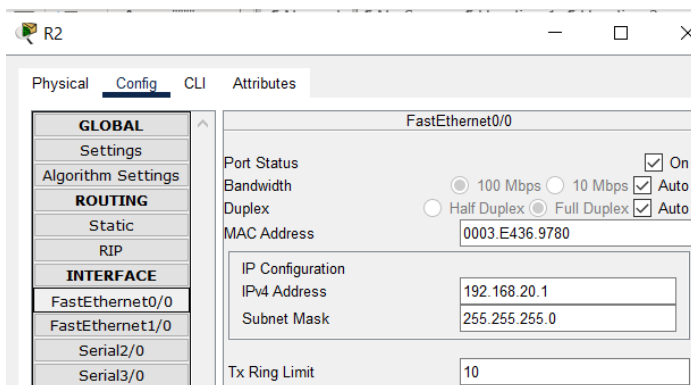
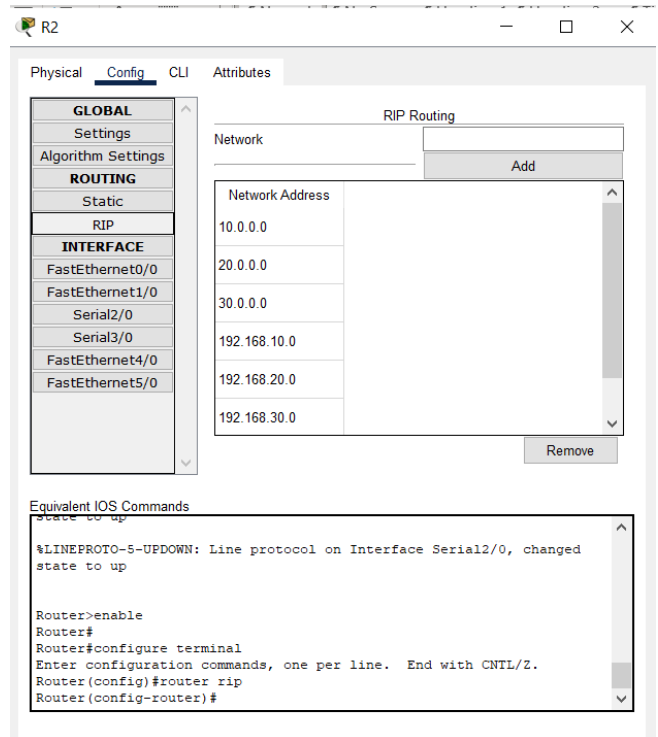
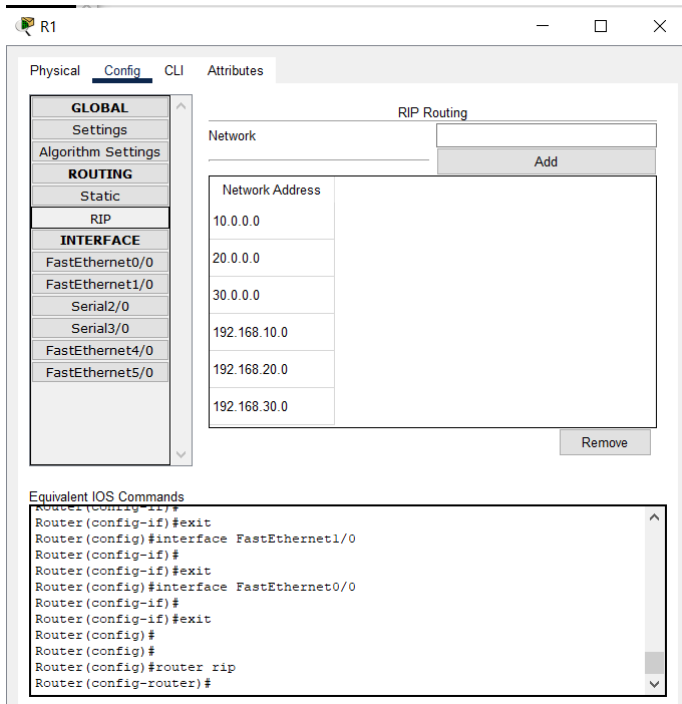
- RIP stands for Routing Information Protocol in which distance vector routing protocol is used for data/packet transmission. In the Routing Information Protocol (RIP), the maximum number of Hop is 15, because it prevents routing loops from source to destination. Compared to other routing protocols, RIP (Routing Information Protocol) is poor and limited in size i.e. small network. The main advantage of using RIP is it uses the UDP (User Datagram Protocol).
- OSPF stands for Open Shortest Path First which uses a link-state routing algorithm. Using the link state information which is available in routers, it constructs the topology in which topology determines the routing table for routing decisions. It is a router protocol which is used to find the best path for packets when they are passing through the set of connected networks simultaneously. The main disadvantage of OSPF is that it is difficult than other protocols.



## Code And Output :

### A) RIP Protocol :





R2

Physical **Config** CLI Attributes

**GLOBAL**

Settings

Algorithm Settings

**ROUTING**

Static

RIP

**INTERFACE**

FastEthernet0/0

FastEthernet1/0

Serial2/0

Serial3/0

Serial3/0

Port Status ☒ On

Duplex ☐ Full Duplex

Clock Rate 2000000

IP Configuration

IPv4 Address 30.0.0.1

Subnet Mask 255.0.0.0

Tx Ring Limit 10

R3

Physical **Config** CLI Attributes

**GLOBAL**

Settings

Algorithm Settings

**ROUTING**

Static

RIP

**INTERFACE**

FastEthernet0/0

FastEthernet1/0

Serial2/0

Serial3/0

FastEthernet4/0

FastEthernet5/0

RIP Routing

Network

Add

Network Address
10.0.0.0
20.0.0.0
30.0.0.0
192.168.10.0
192.168.20.0
192.168.30.0

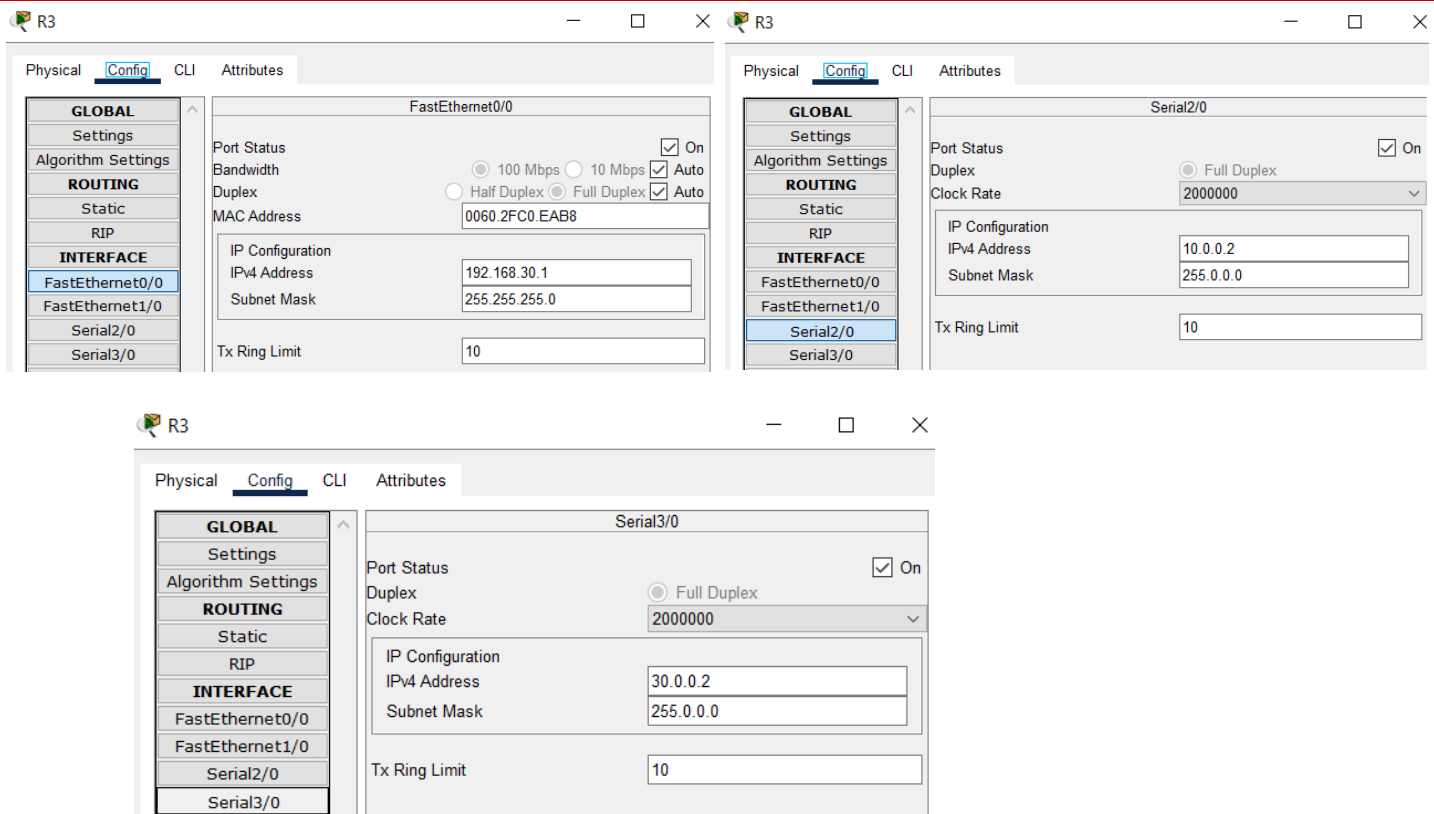
Remove

Equivalent IOS Commands

```

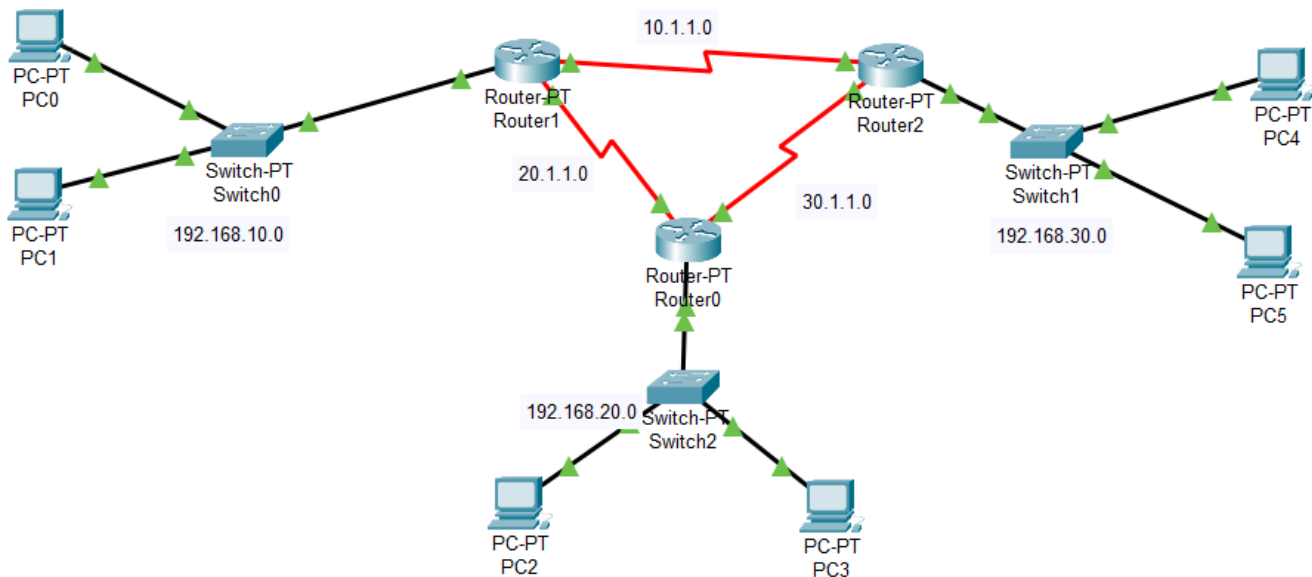
to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial12/0, changed state
to up

Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#router rip
Router(config-router)#
  
```



**Code And Output :**

## B) OSPF Protocol :



Router1

```
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial3/0, changed state to down

00:22:53: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.30.1 on Serial3/0 from FULL to DOWN, Neighbor
Down: Interface down or detached

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to down

00:22:53: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.30.1 on Serial2/0 from FULL to DOWN, Neighbor
Down: Interface down or detached

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial3/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to up

00:23:03: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.30.1 on Serial3/0 from LOADING to FULL, Loading
Done

00:23:03: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.30.1 on Serial2/0 from LOADING to FULL, Loading
Done
```

Router1

Physical Config CLI Attributes

GLOBAL	FastEthernet0/0
Settings	Port Status <input checked="" type="checkbox"/> On
Algorithm Settings	Bandwidth <input checked="" type="radio"/> 100 Mbps <input type="radio"/> 10 Mbps <input checked="" type="checkbox"/> Auto
ROUTING	Duplex <input type="radio"/> Half Duplex <input checked="" type="radio"/> Full Duplex <input checked="" type="checkbox"/> Auto
Static	MAC Address 0010.1176.76E0
RIP	IP Configuration
INTERFACE	IPv4 Address 192.168.10.1
FastEthernet0/0	Subnet Mask 255.255.255.0
FastEthernet1/0	
Serial2/0	
Serial3/0	Tx Ring Limit 10

Name of Instructor: Ms. Jessica D'cruz

Router1

Physical **Config** CLI Attributes

**GLOBAL**

- Settings
- Algorithm Settings

**ROUTING**

- Static
- RIP

**INTERFACE**

- FastEthernet0/0
- FastEthernet1/0
- Serial2/0**
- Serial3/0
- FastEthernet4/0
- FastEthernet5/0

**Serial2/0**

Port Status ☒ On

Duplex ☐ Full Duplex

Clock Rate 2000000

IP Configuration

IPv4 Address 10.1.1.1

Subnet Mask 255.0.0.0

Tx Ring Limit 10

Router1

Physical **Config** CLI Attributes

**GLOBAL**

- Settings
- Algorithm Settings

**ROUTING**

- Static
- RIP

**INTERFACE**

- FastEthernet0/0
- FastEthernet1/0
- Serial2/0
- Serial3/0**

**Serial3/0**

Port Status ☒ On

Duplex ☐ Full Duplex

Clock Rate 2000000

IP Configuration

IPv4 Address 20.1.1.1

Subnet Mask 255.0.0.0

Tx Ring Limit 10

Router0

```

00:22:53: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.10.1 on Serial2/0 from FULL to DOWN, Neighbor
Down: Interface down or detached

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial3/0, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial3/0, changed state to up

00:23:03: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.10.1 on Serial2/0 from LOADING to FULL, Loading
Done

```

Router0

Physical **Config** CLI Attributes

**GLOBAL**

Settings

Algorithm Settings

**ROUTING**

Static

RIP

**INTERFACE**

FastEthernet0/0

FastEthernet1/0

Serial2/0

Serial3/0

FastEthernet0/0

Port Status ☒ On

Bandwidth ☐ 100 Mbps ☐ 10 Mbps ☒ Auto

Duplex ☐ Half Duplex ☒ Full Duplex ☒ Auto

MAC Address 00D0.D311.82E0

IP Configuration

IPv4 Address 192.168.20.1

Subnet Mask 255.255.255.0

Tx Ring Limit 10

Router0

Physical **Config** CLI Attributes

**GLOBAL**

Settings

Algorithm Settings

**ROUTING**

Static

RIP

**INTERFACE**

FastEthernet0/0

FastEthernet1/0

Serial2/0

Serial3/0

Serial2/0

Port Status ☒ On

Duplex ☒ Full Duplex

Clock Rate 2000000

IP Configuration

IPv4 Address 20.1.1.2

Subnet Mask 255.0.0.0

Tx Ring Limit 10

Router0

Physical **Config** CLI Attributes

**GLOBAL**

Settings

Algorithm Settings

**ROUTING**

Static

RIP

**INTERFACE**

FastEthernet0/0

FastEthernet1/0

Serial2/0

Serial3/0

Serial3/0

Port Status ☒ On

Duplex ☒ Full Duplex

Clock Rate 2000000

IP Configuration

IPv4 Address 192.168.30.1

Subnet Mask 255.255.255.0

Tx Ring Limit 10

Router2

```
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to down

00:22:53: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.10.1 on Serial2/0 from FULL to DOWN, Neighbor
Down: Interface down or detached

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial3/0, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial3/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to up

00:23:03: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.10.1 on Serial2/0 from LOADING to FULL, Loading
Done
```

Router2

Physical Config CLI Attributes

**GLOBAL**  
 Settings  
 Algorithm Settings  
**ROUTING**  
 Static  
 RIP  
**INTERFACE**  
 FastEthernet0/0  
 FastEthernet1/0  
 Serial2/0  
 Serial3/0

FastEthernet0/0

Port Status ☒ On  
 Bandwidth ☒ 100 Mbps ☐ 10 Mbps ☒ Auto  
 Duplex ☐ Half Duplex ☒ Full Duplex ☒ Auto  
 MAC Address 0002.1670.664E  
 IP Configuration  
 IPv4 Address 192.168.30.1  
 Subnet Mask 255.255.255.0  
 Tx Ring Limit 10

**GLOBAL**  
 Settings  
 Algorithm Settings  
**ROUTING**  
 Static  
 RIP  
**INTERFACE**  
 FastEthernet0/0  
 FastEthernet1/0  
 Serial2/0  
 Serial3/0

Serial2/0

Port Status ☒ On  
 Duplex ☒ Full Duplex  
 Clock Rate 2000000  
 IP Configuration  
 IPv4 Address 10.1.1.2  
 Subnet Mask 255.0.0.0  
 Tx Ring Limit 10

Router2

Physical Config CLI Attributes

**GLOBAL**  
 Settings  
 Algorithm Settings  
**ROUTING**  
 Static  
 RIP  
**INTERFACE**  
 FastEthernet0/0  
 FastEthernet1/0  
 Serial2/0  
 Serial3/0

Serial3/0

Port Status ☒ On  
 Duplex ☒ Full Duplex  
 Clock Rate 2000000  
 IP Configuration  
 IPv4 Address 30.1.1.2  
 Subnet Mask 255.0.0.0  
 Tx Ring Limit 10





# Wireless Sensor Networks & Mobile Communication

## Practical No.6

### DEPARTMENT OF COMPUTER SCIENCE

Name:	Shariq Sheikh	Roll Number	TCS2324075
Paper Code:	SIUSCS61	Class	T.Y. B.Sc. (Computer Science)
Topic:	Wireless network	Batch	11
Date:	05th February 2024	Practical No	6

**A) AIM:** Simulate a network for simple wireless network.

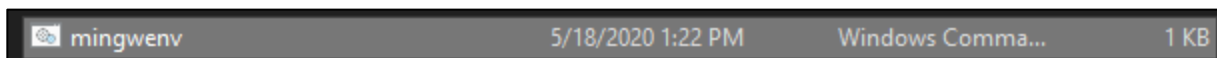
#### **B) DESCRIPTION:**

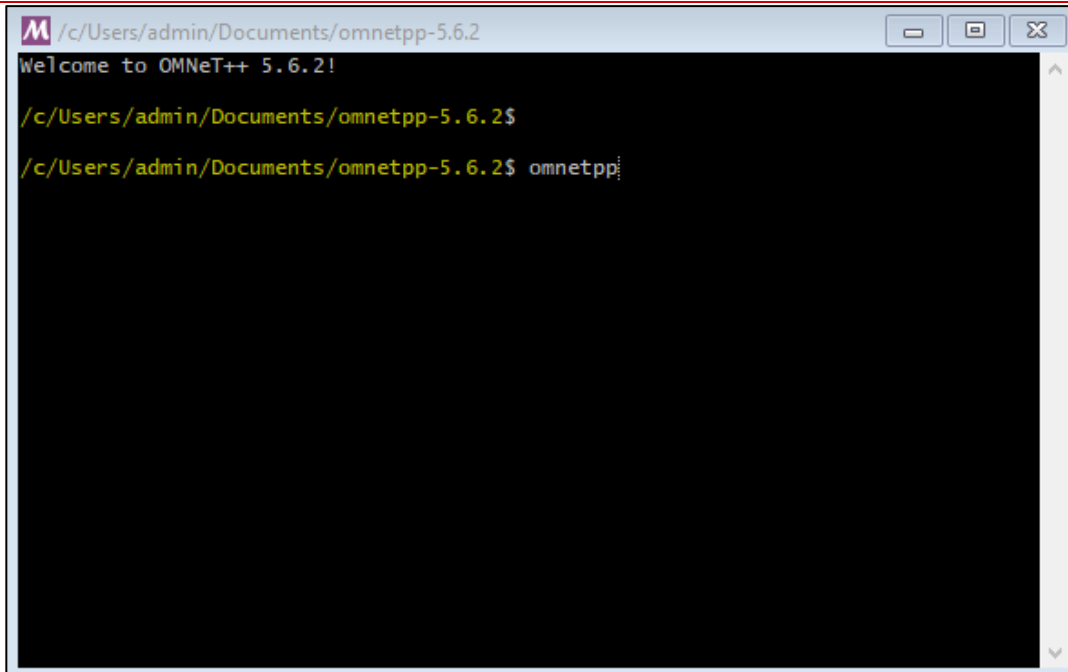
Computer networks that are not connected by cables are called wireless networks. They generally use radio waves for communication between the network nodes. They allow devices to be connected to the network while roaming around within the network coverage. It increases the mobility of network devices connected to the system since the devices need not be connected to each other. Types of Wireless Networks: Wireless LANs, Wireless MANs and Wireless WANs. Wireless networks require very limited or no wires. Thus, it reduces the equipment and setup costs.

#### **C) NETWORK TOPOLOGY, CONFIGURATIONS AND OUTPUT:**

##### **Configurations:**

Step 1: Open omnet folder → Open mingwenv → Type omnetpp in console

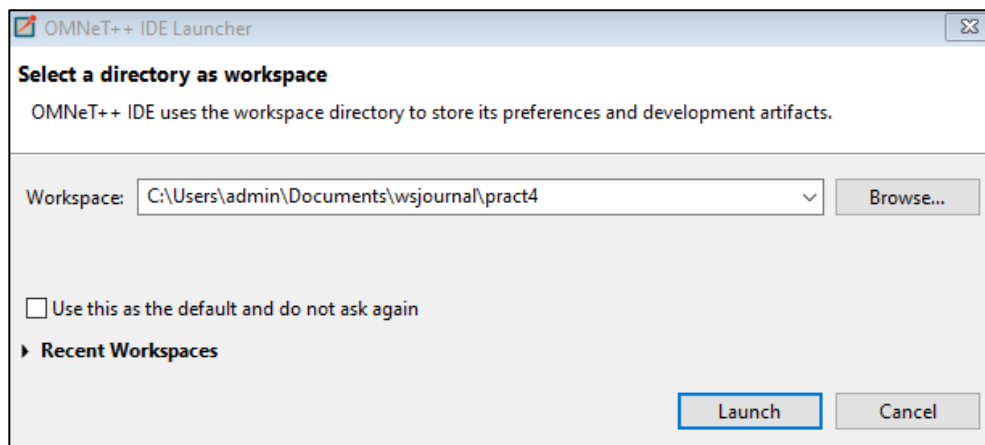




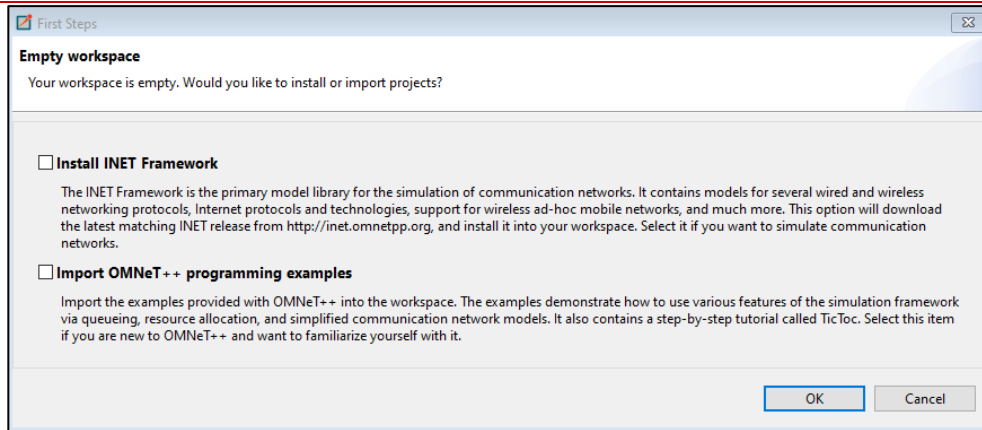
```
M /c/Users/admin/Documents/omnetpp-5.6.2
Welcome to OMNeT++ 5.6.2!

/c/Users/admin/Documents/omnetpp-5.6.2$
/c/Users/admin/Documents/omnetpp-5.6.2$ omnetpp:
```

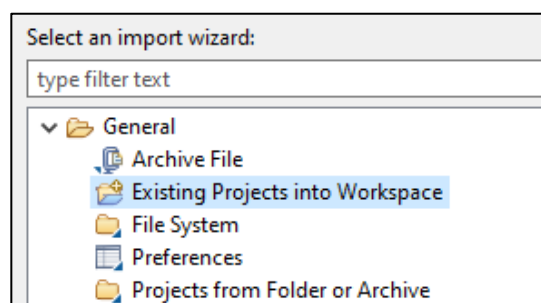
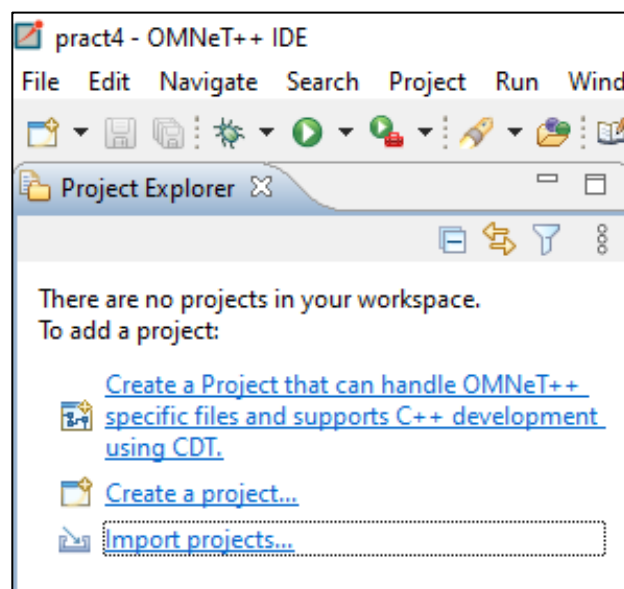
Step 2: Create workspace outside omnet and inet folder → Launch

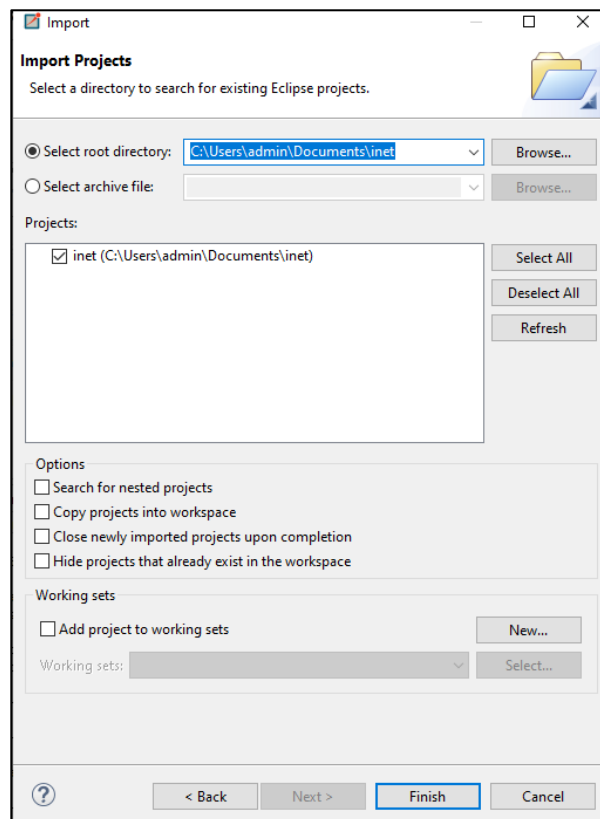
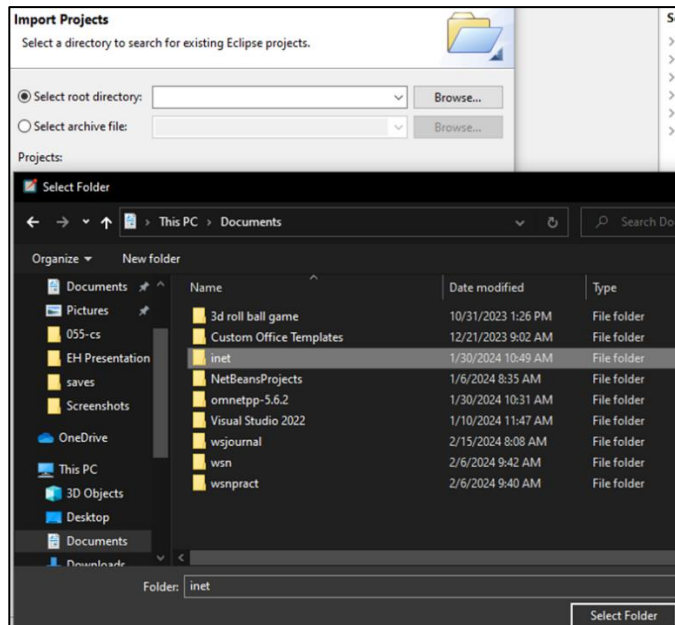


Step 3: Uncheck the boxes and click ok

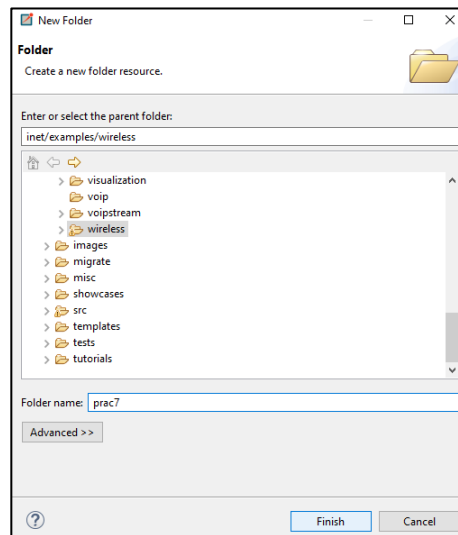
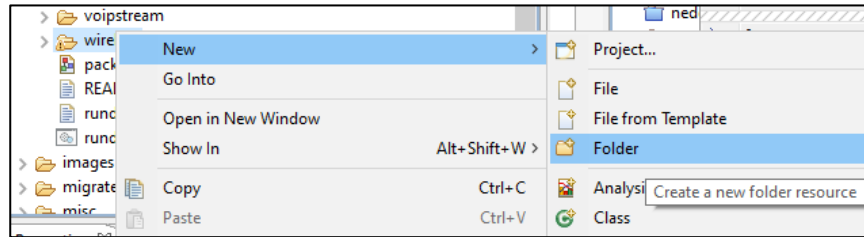


Step 4: Click import projects → General → Existing Projects into Workspace → Next → Browse inet folder location (under “select root directory”) → Check inet → Finish

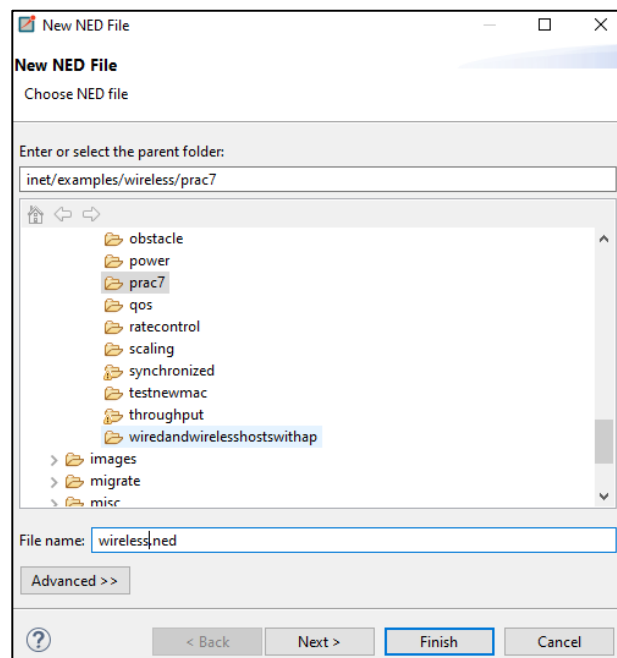
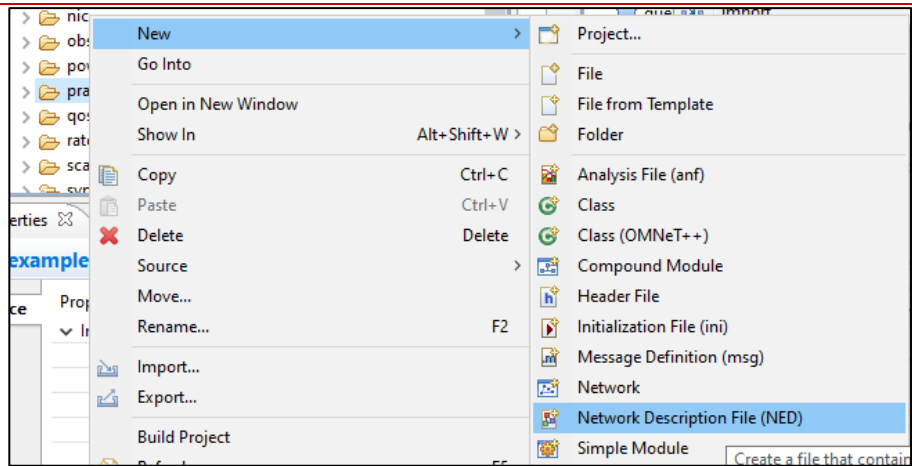


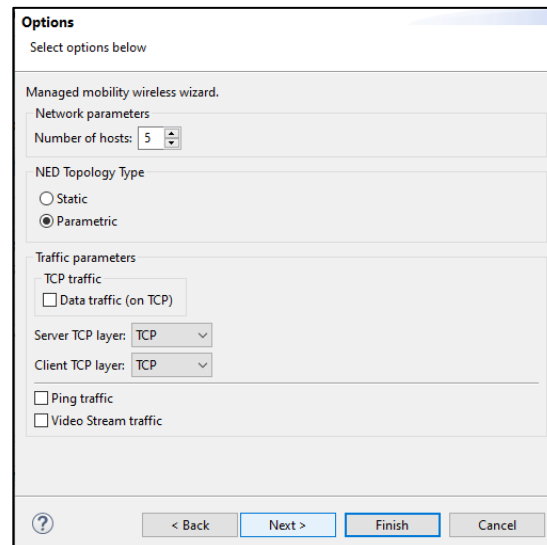
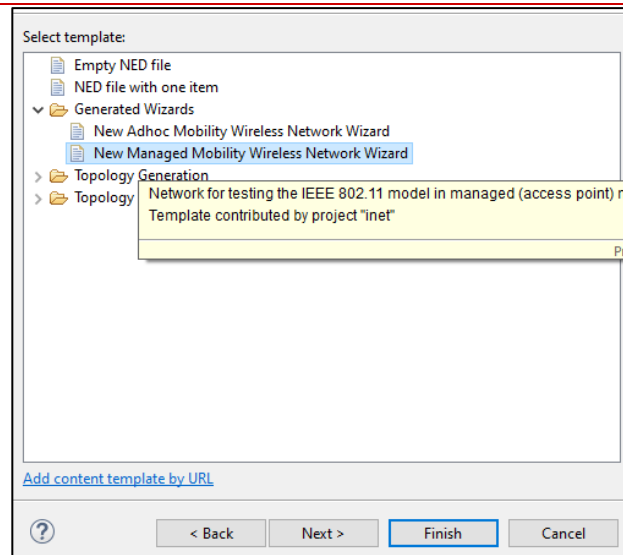


Step 5: Click inet folder → Examples → wireless → Right click → New → Folder → Provide folder name → Finish



Step 6: Right click on the created folder → New → Select NED → Provide a name → Next → Click “generated wizards” → Select “New managed mobility wireless network wizard” → Next → Finish





Step 7: Go to wireless.ned (of your created practical folder; under wireless folder) → Source → Do the following changes as mentioned below → Save the changes

```

Getting Started | At a Glance | wireless.ned X
package inet.examples.wireless.prac7;

// numOfHosts: 5

import inet.networklayer.configurator.ipv4.IPv4NetworkConfigurator;
import inet.node.inet.WirelessHost;
import inet.node.wireless.AccessPoint;
import inet.physicallayer.ieee80211.packetlevel.Ieee80211ScalarRadioMedium;

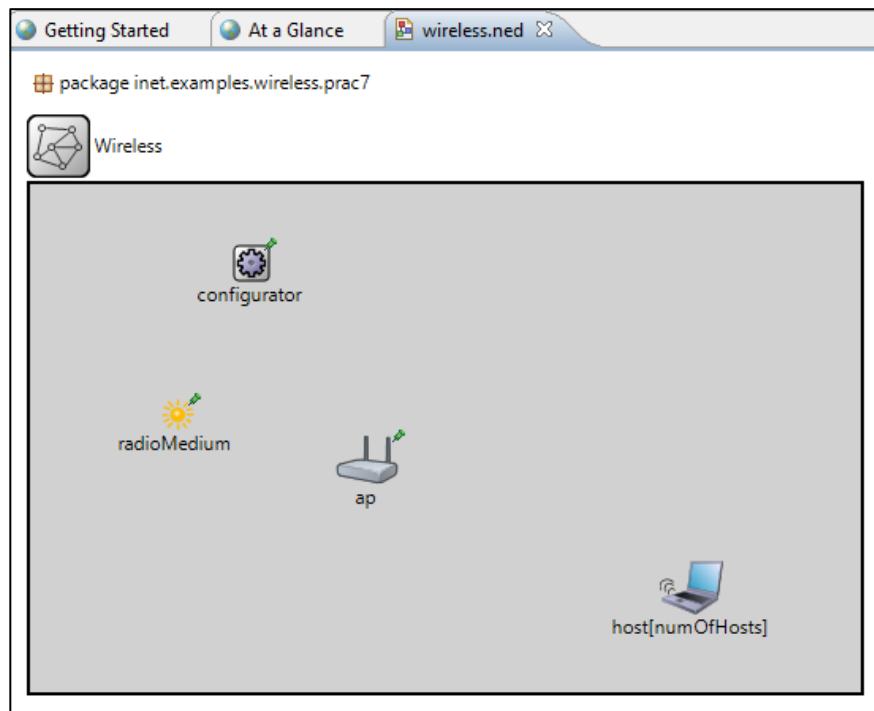
network Wireless
{
    parameters:
        int numOfHosts;

    submodules:
        host[numOfHosts]: WirelessHost
        {
            @display("r=, ,#707070");
        }

        ap: AccessPoint
        {
            @display("p=213,174;r=, ,#707070");
        }

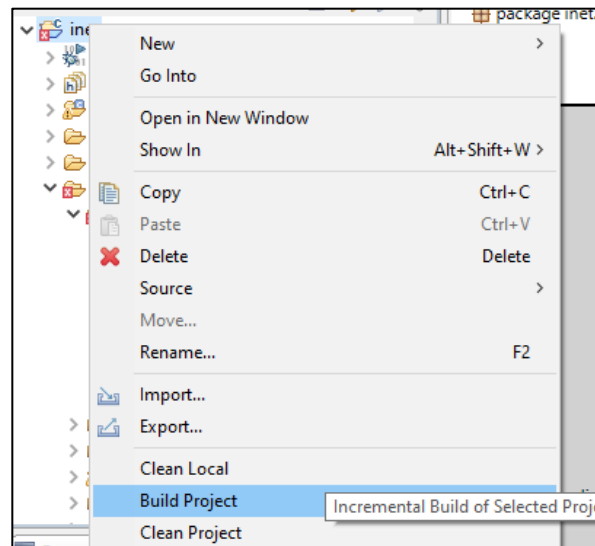
        configurator: IPv4NetworkConfigurator
        {
            @display("p=140,50");
        }
        radioMedium:Ieee80211ScalarRadioMedium
        {
            parameters:
                @display("p=93,146");
        }
}

```



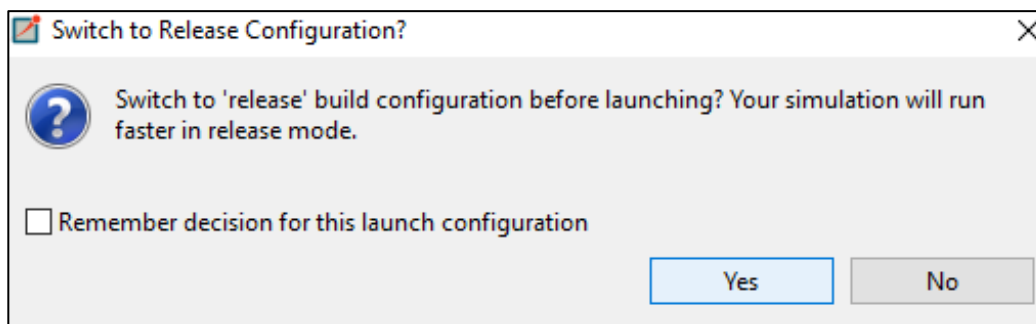
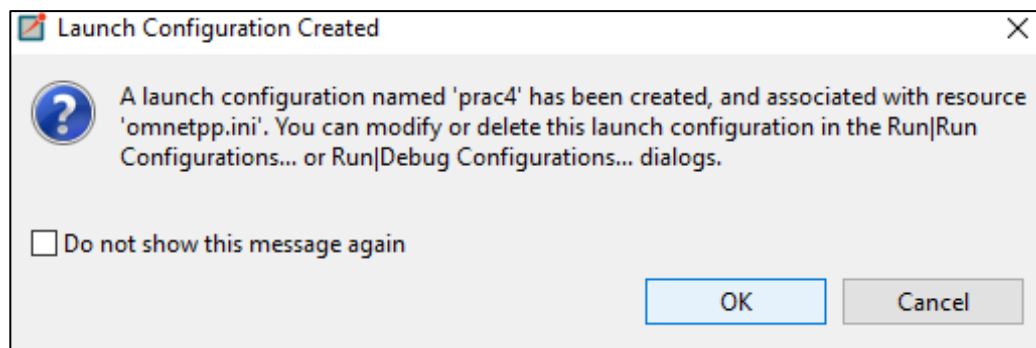
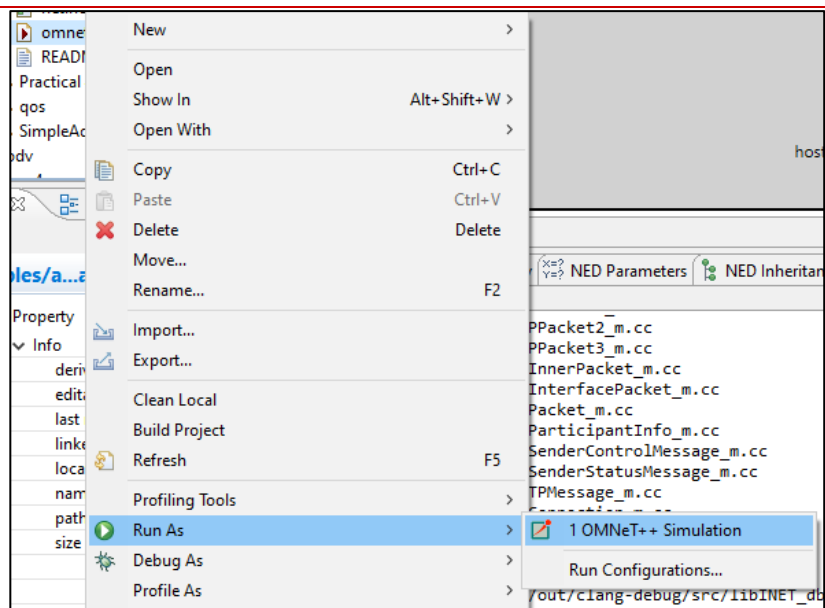


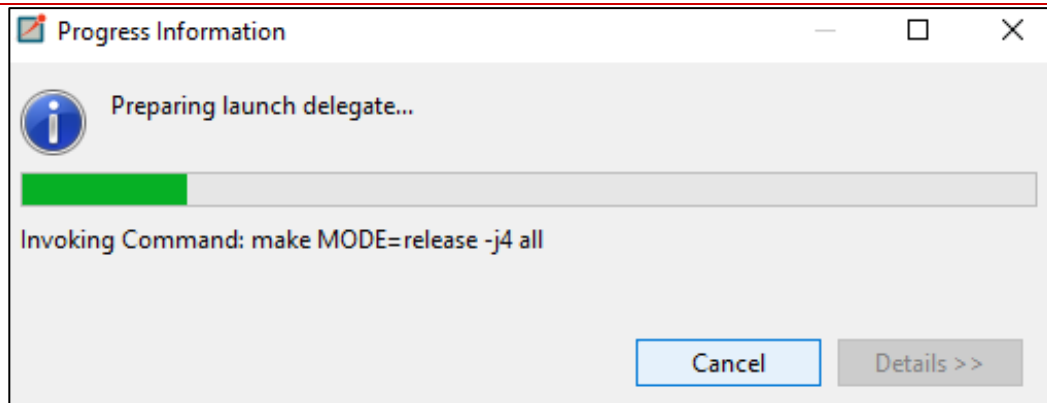
Step 8: Now, for testing our simulation: Right click on inet → Select build project → Right click on omnetpp.ini (present in your created practical folder) → Select “run as” → Select “omnet++ simulation → Click OK → Click Yes → Done



```
CDT Build Console [inet]
inet/transportlayer/rtp/RTCPacket2_m.cc
inet/transportlayer/rtp/RTCPacket3_m.cc
inet/transportlayer/rtp/RTPInnerPacket_m.cc
inet/transportlayer/rtp/RTPInterfacePacket_m.cc
inet/transportlayer/rtp/RTPPacket_m.cc
inet/transportlayer/rtp/RTPParticipantInfo_m.cc
inet/transportlayer/rtp/RTPSenderControlMessage_m.cc
inet/transportlayer/rtp/RTPSenderStatusMessage_m.cc
inet/transportlayer/sctp/SCTPMessage_m.cc
inet/transportlayer/tcp/TCPConnection_m.cc
inet/transportlayer/tcp_common/TCPSegment_m.cc
inet/transportlayer/udp/UDPPacket_m.cc
Creating .DEF file for libINET_dbg.dll - needed to fix an issue where clang is not exporting type info for classes on Windows
Creating shared library: ../out/clang-debug/src/libINET_dbg.dll
make[1]: Leaving directory '/c/Users/admin/Documents/inet/src'

10:19:19 Build Finished. 0 errors, 21 warnings. (took 5m:25s.972ms)
```





**Output:**

OMNeT++/QtEnv (release) - General #0 - omnetpp.ini - C:\Users\admin\Documents\inet\examples\wireless\prac7

File Simulate Inspect View Help

Next: (inet::RegisterTransportProtocolCommand, id=119) In: Wireless.host[0].networkLayer.ip (IPv4, id=18) At: 0s (now=0s)

Wireless (Wireless) id=1

- numOfHosts (cPar) 5
- host[0] (WirelessHost) id=2
- host[1] (WirelessHost) id=3
- host[2] (WirelessHost) id=4
- host[3] (WirelessHost) id=5
- host[4] (WirelessHost) id=6
- ap (AccessPoint) id=7
- configurator (IPv4NetworkConfigurator) id=8
- radioMedium (Ieee80211ScalarRadioMedium) id=
- canvas (cCanvas) 1 toplevel figure(s)

Wireless

Initializing module Wireless.ap.wlan[0].mac.tx, stage 13  
 Initializing module Wireless.ap.wlan[0].radio, stage 13  
 INFO (Ieee80211ScalarRadio) Wireless.ap.wlan[0].radio: Initialized (inet::physicalLayer::Ieee80211Radio) radio, antenna = { IsotropicAntenna }, transmitter = { Ieee80211ScalarTransmitter }, stage 13  
 Initializing module Wireless.ap.wlan[0].radio.transmitter, stage 13  
 Initializing module Wireless.ap.wlan[0].radio.receiver, stage 13  
 Initializing module Wireless.configurator, stage 13  
 INFO (Ieee80211ScalarRadioMedium) Wireless.radioMedium: Initialized (inet::physicalLayer::RadioMedium) radioMedium, propagation = { ConstantSpeedPropagation, ignoreMovementDuringTransmissions }, stage 13  
 Initializing module Wireless.radioMedium.mediumLimitCache, stage 13

General #0: Wireless

Msg stats: 21 scheduled / 129 existing / 129 created

OMNeT++/QtEnv (release) - General #0 - omnetpp.ini - C:\Users\admin\Documents\inet\examples\wireless\prac7

File Simulate Inspect View Help

Next: transmissionTimer (omnetpp::Message, id=111) In: Wireless.ap.wlan[0].radio (Ieee80211ScalarRadio, id=406) At: 0.022278138552s (now=0.00039541359s)

Next simulation event

Wireless (Wireless) id=1

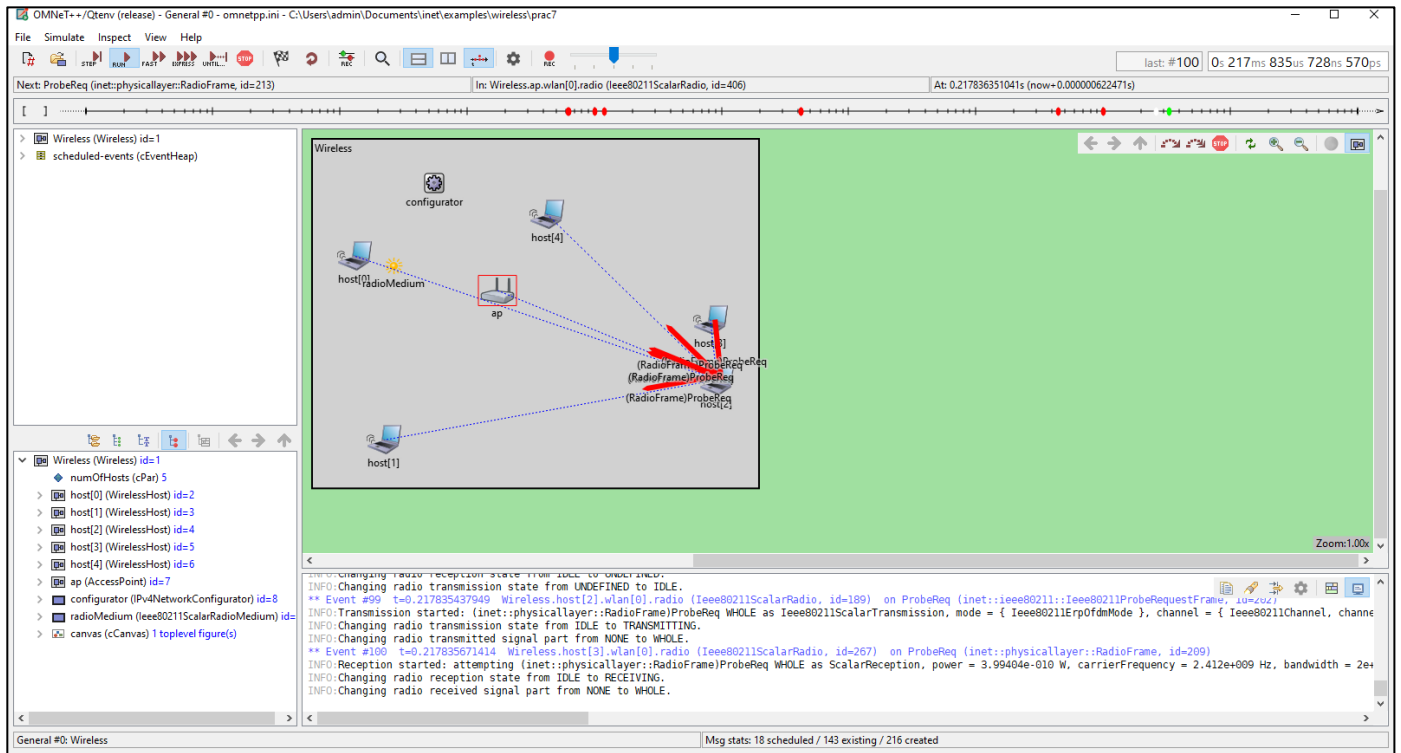
- numOfHosts (cPar) 5
- host[0] (WirelessHost) id=2
- host[1] (WirelessHost) id=3
- host[2] (WirelessHost) id=4
- host[3] (WirelessHost) id=5
- host[4] (WirelessHost) id=6
- ap (AccessPoint) id=7
- configurator (IPv4NetworkConfigurator) id=8
- radioMedium (Ieee80211ScalarRadioMedium) id=
- canvas (cCanvas) 1 toplevel figure(s)

Wireless

INFO: Changing radio reception state from NONE to RECEIVING.  
 \*\* Event #21 t=0.022232955939 Wireless.host[3].wlan[0].radio (Ieee80211ScalarRadio, id=267) on Beacon (inet::physicalLayer::RadioFrame, id=139)  
 INFO: Reception started: attempting (inet::physicalLayer::RadioFrame)Beacon WHOLE as ScalarReception, power = 3.25837e-011 W, carrierFrequency = 2.412e+009 Hz, bandwidth = 2e+006 Hz  
 INFO: Changing radio reception state from IDLE to RECEIVING.  
 \*\* Event #22 t=0.022233050967 Wireless.host[2].wlan[0].radio (Ieee80211ScalarRadio, id=189) on Beacon (inet::physicalLayer::RadioFrame, id=137)  
 INFO: Reception started: attempting (inet::physicalLayer::RadioFrame)Beacon WHOLE as ScalarReception, power = 2.61844e-011 W, carrierFrequency = 2.412e+009 Hz, bandwidth = 2e+006 Hz  
 INFO: Changing radio reception state from IDLE to RECEIVING.  
 INFO: Changing radio received signal part from NONE to WHOLE.

General #0: Wireless

Msg stats: 18 scheduled / 137 existing / 148 created





# Wireless Sensor Networks & Mobile Communication

## Practical No.7

### DEPARTMENT OF COMPUTER SCIENCE

Name:	Shariq Sheikh	Roll Number	TCS2324075
Paper Code:	SIUSCS61	Class	TYBSc(Computer Science)
Topic:	Mac Protocol	Batch	II
Date:	06th February 2024	Practical No	7

#### A) AIM:

**Create a Mac protocol simulation implementation for wireless Sensor Networks**

#### B) DESCRIPTION:

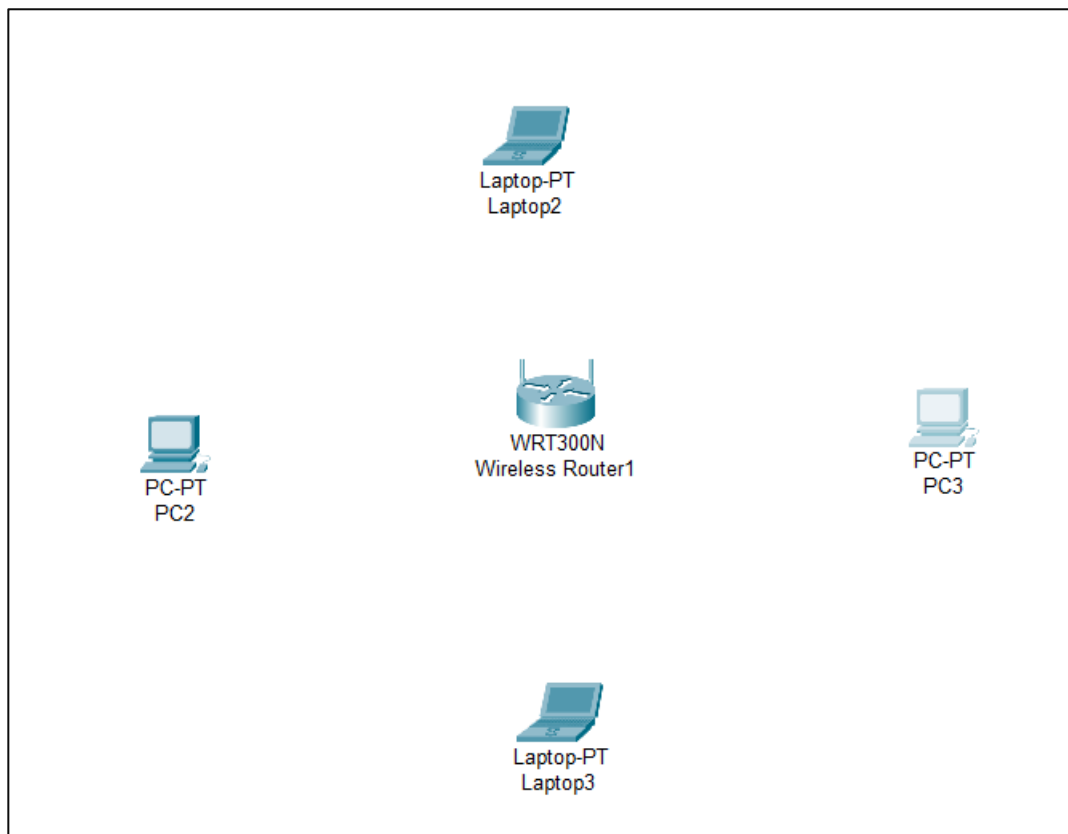
The Medium Access Control (MAC) layer is a sublayer of the data link layer that deals with the protocol access to the physical network medium. MAC protocols control how devices in a network access and use the shared communication medium to avoid collisions and ensure efficient communication.

- CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance):** Commonly used in wireless networks, CSMA/CA involves nodes checking for the channel's availability before transmitting to avoid collisions.
- CSMA/CD (Carrier Sense Multiple Access with Collision Detection):** Historically used in wired networks like Ethernet, CSMA/CD detects collisions and takes steps to resolve them.
- TDMA (Time Division Multiple Access):** In TDMA, time is divided into slots, and each node is assigned a specific time slot for transmission. This approach is often used in satellite communication and cellular networks.
- FDMA (Frequency Division Multiple Access):** FDMA allocates different frequency bands to different nodes for simultaneous transmission, commonly used in analog communication systems.

5. **CDMA (Code Division Multiple Access):** CDMA assigns a unique code to each node, allowing multiple nodes to transmit simultaneously on the same frequency band. It's widely used in digital cellular networks.

### C) NETWORK TOPOLOGY, CONFIGURATIONS AND OUTPUT:

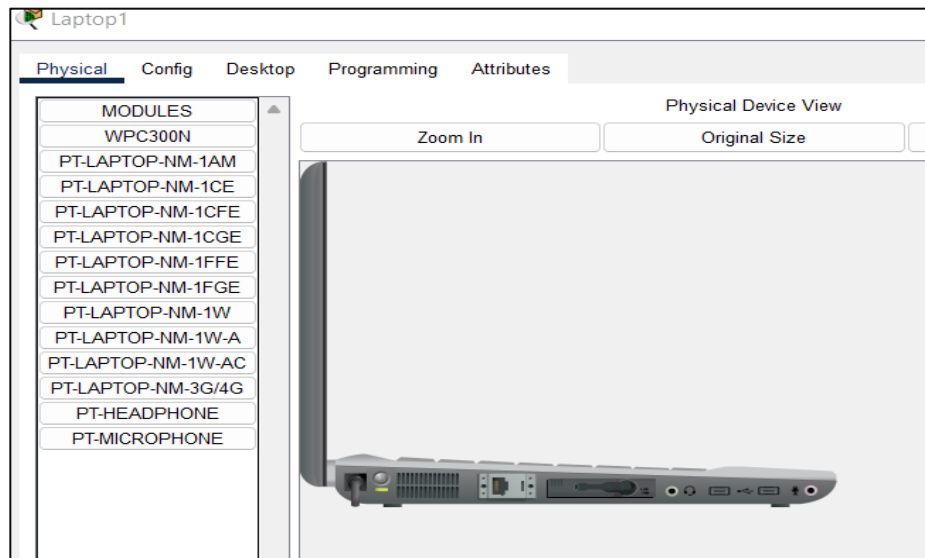
**Network topology:**



## Configurations:

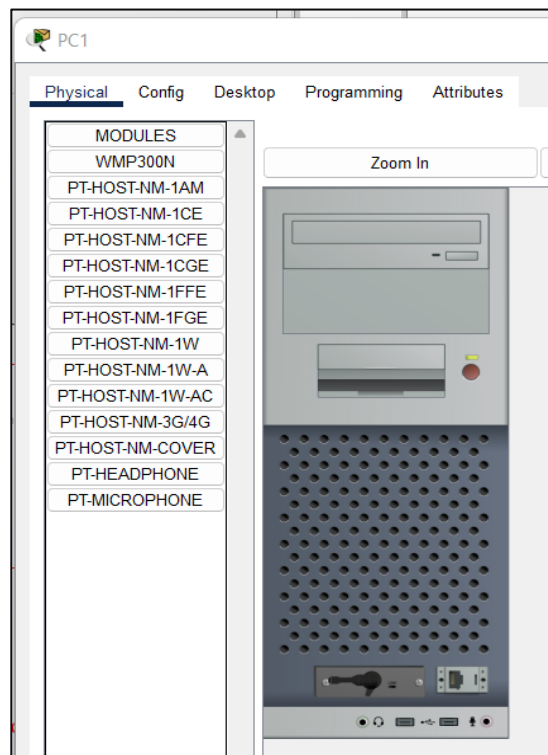
### Configurations done on Laptops:

Physical configuration: Adding the wireless interface



### Configuration done on PCs:

Physical configuration: Adding the wireless interface



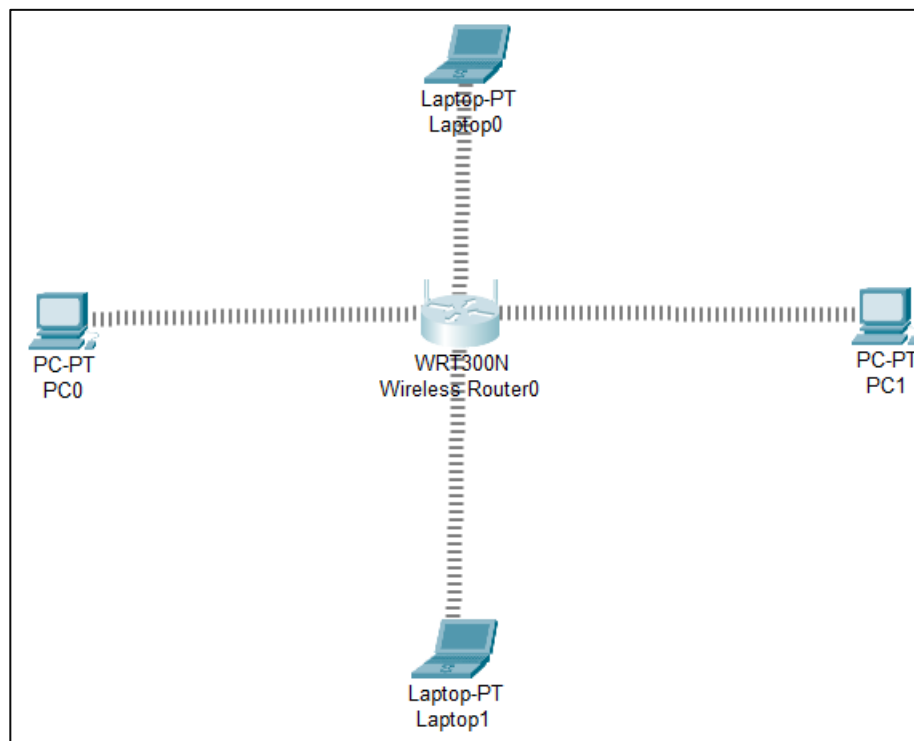


## Configuration done on Router:

Adding the MAC Address of Pcs in the MAC Address filter list(present in the GUI)

Wireless		Setup	Wireless	Security	Access Restrictions	Applications & Gaming	Admin
		Basic Wireless Settings		Wireless Security	Guest Network	Wireless MAC Filter	
<b>Wireless MAC Filter</b>							
Wireless Port: 2.4G							
<input checked="" type="radio"/> Enabled <input type="radio"/> Disabled							
<input checked="" type="radio"/> Prevent PCs listed below from accessing the wireless network <input type="radio"/> Permit PCs listed below to access wireless network							
Wireless Client List							
MAC Address filter list	MAC 01:	00:01:43:19:AD:C0			MAC 26:	00:00:00:00:00:00	
	MAC 02:	00:D0:FF:B7:9B:30			MAC 27:	00:00:00:00:00:00	
	MAC 03:	00:00:00:00:00:00			MAC 28:	00:00:00:00:00:00	
	MAC 04:	00:00:00:00:00:00			MAC 29:	00:00:00:00:00:00	

## Output





# Wireless Sensor Networks & Mobile Communication

## Practical No.8

### DEPARTMENT OF COMPUTER SCIENCE

<b>Name:</b>	Shariq Sheikh	<b>Roll Number</b>	TCS2324075
<b>Paper Code:</b>	SIUSCS61	<b>Class</b>	TYBSc(Computer Science)
<b>Topic:</b>	Mobile Adhoc Network	<b>Batch</b>	II
<b>Date:</b>	06th February 2024	<b>Practical No</b>	8

#### A) AIM:

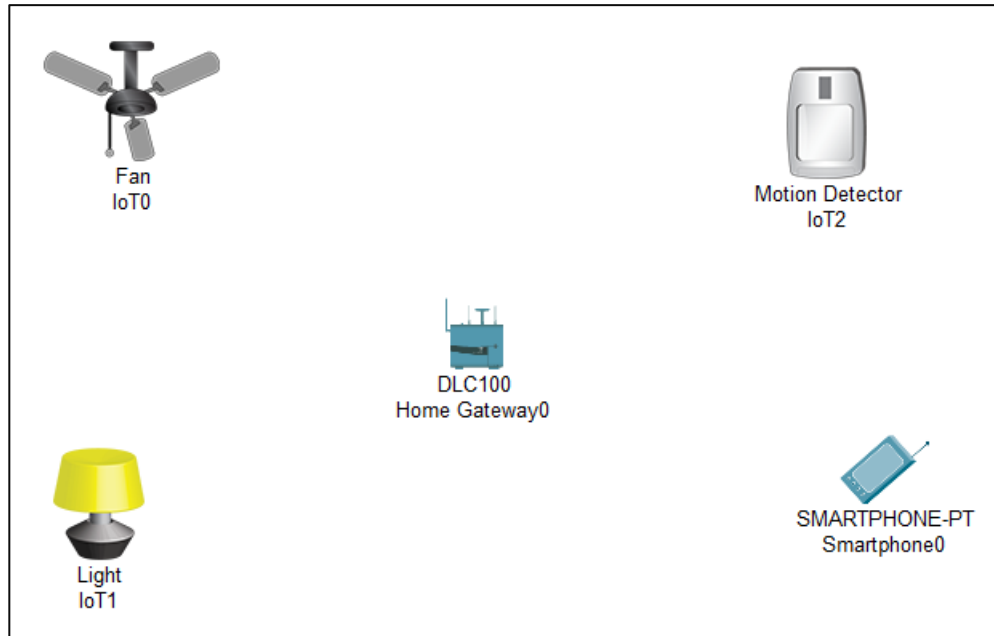
**Simulate Mobile Adhoc Network with Directional Antenna**

#### B) DESCRIPTION:

MANET stands for Mobile Adhoc Network also called a wireless Adhoc network or Adhoc wireless network that usually has a routable networking environment on top of a Link Layer ad hoc network.. They consist of a set of mobile nodes connected wirelessly in a self-configured, self-healing network without having a fixed infrastructure. MANET nodes are free to move randomly as the network topology changes frequently. Each node behaves as a router as they forward traffic to other specified nodes in the network.

## C) NETWORK TOPOLOGY, CONFIGURATIONS AND OUTPUT:

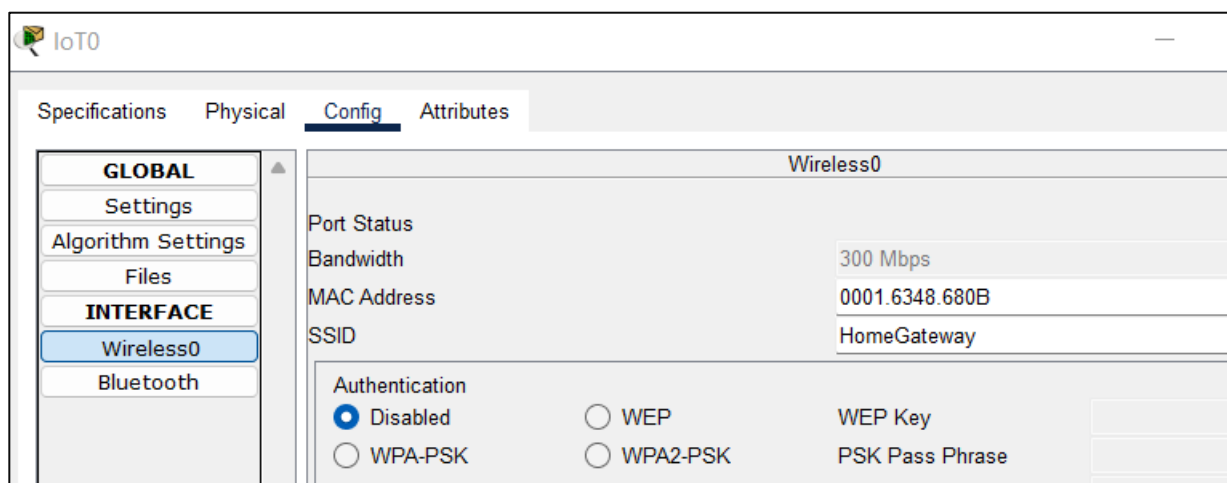
### Network topology:



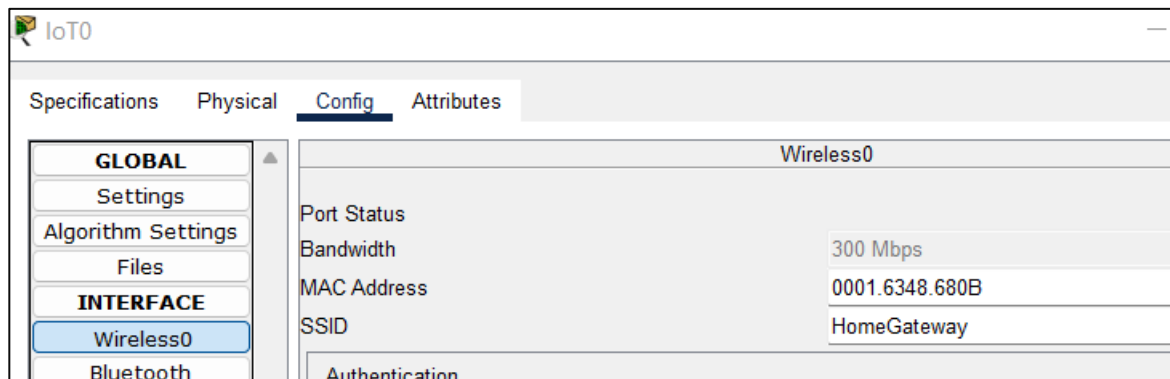
### Configurations:

Copy the SSID of Homegateway and paste it to the ssid on fan, light and motion detector.

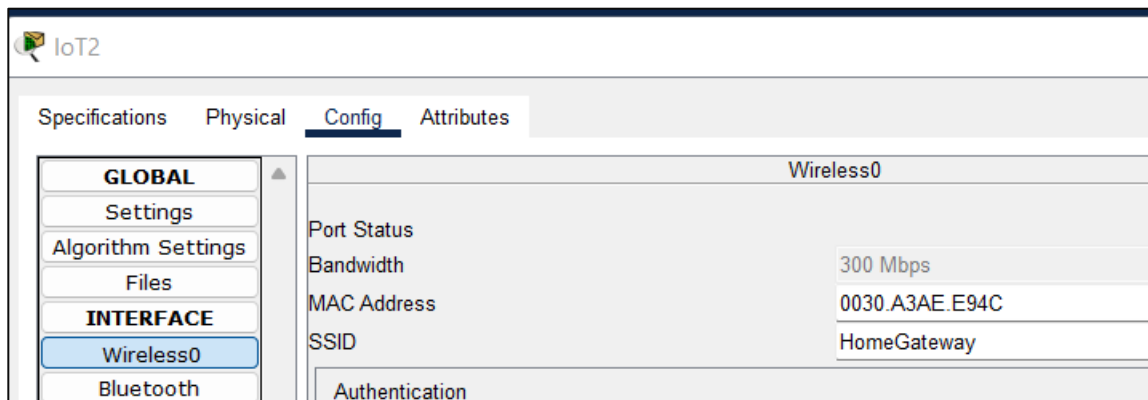
For Light:



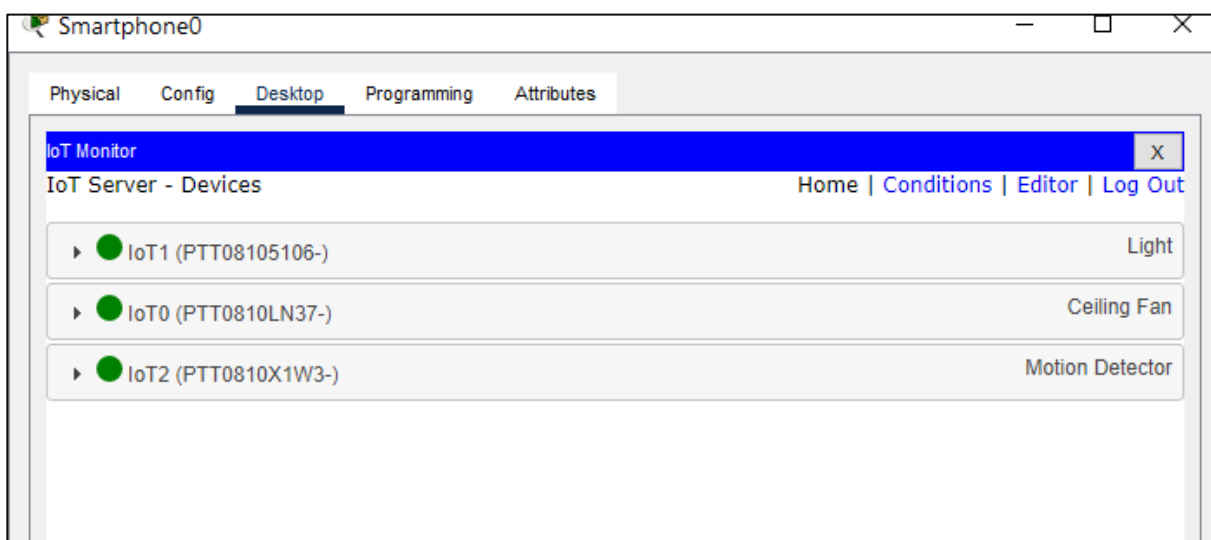
For fan:



For Motion detector:



For smartphone(Go to Desktop>>IOT monitor)



Physical Config **Desktop** Programming Attributes

IoT Monitor X

IoT Server - Device Conditions [Home](#) | [Conditions](#) | [Editor](#) | [Log Out](#)

Actions	Enabled	Name	Condition	Actions
<a href="#">Edit</a> <a href="#">Remove</a>	Yes	decon	IoT2 On is true	Set IoT1 Status to On Set IoT0 Status to High
<a href="#">Edit</a> <a href="#">Remove</a>	Yes	dec_off	IoT2 On is false	Set IoT1 Status to Off Set IoT0 Status to Off

[Add](#)

**EDIT RULE**

Name

Enabled ☒

If:

Match **All**

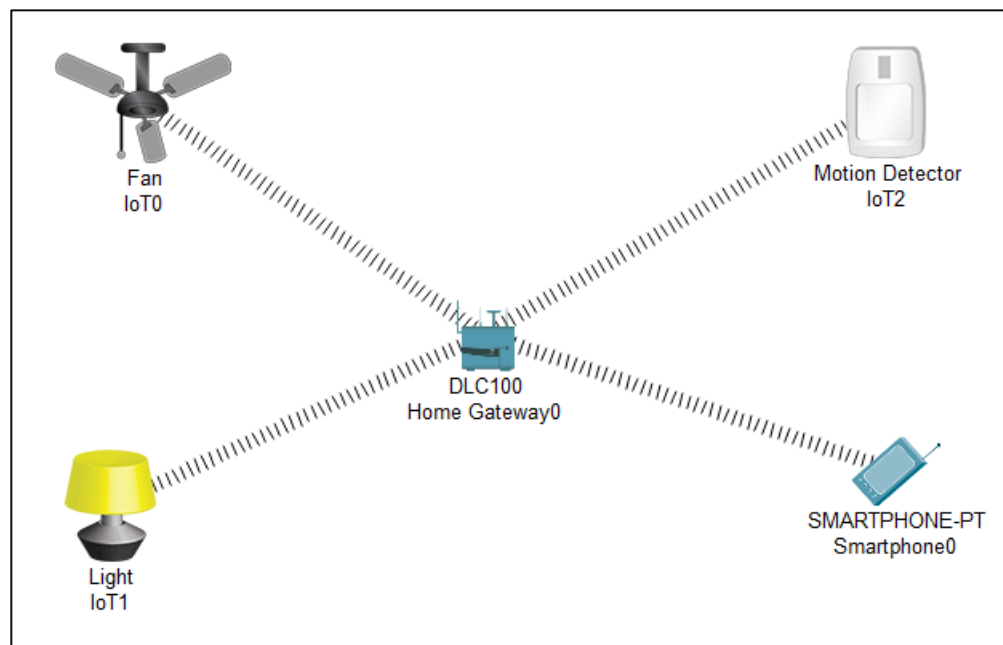
is

Then set:

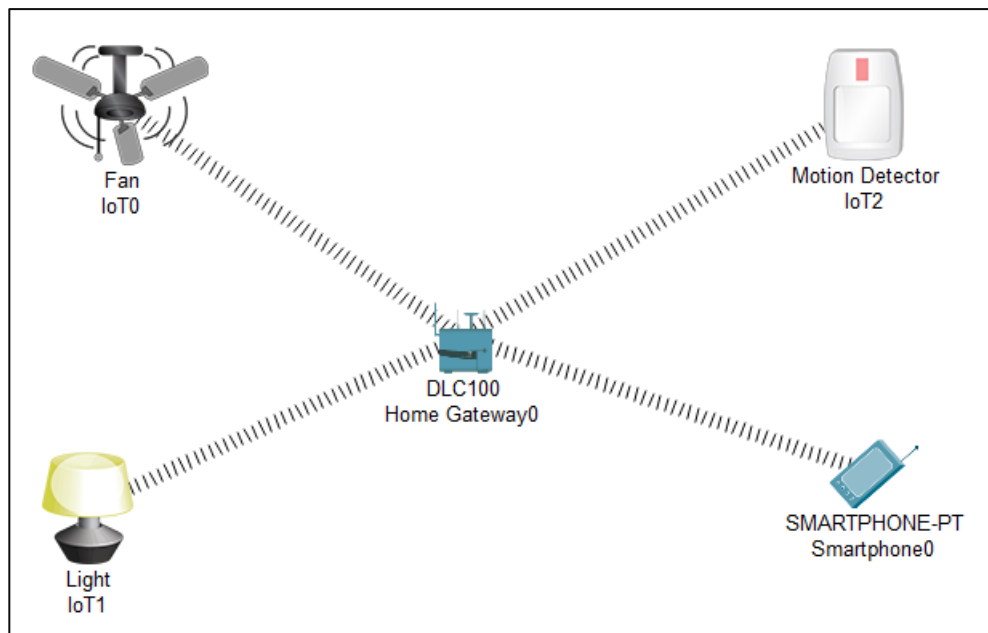
to

to

## Output



When motion detector is on (press alt with motion detector):





# Wireless Sensor Networks & Mobile Communication

## Practical No.9

### DEPARTMENT OF COMPUTER SCIENCE

Name:	Shariq Sheikh	Roll Number	TCS2324075
Paper Code:	SIUSCS61	Class	TYBSc(Computer Science)
Topic:	Mobile Network	Batch	II
Date:	06th February 2024	Practical No	9

#### A) AIM:

**Create a mobile network using Cell Tower, Central Office Server, Web browser and Web Server.**

**Simulate connection between them.**

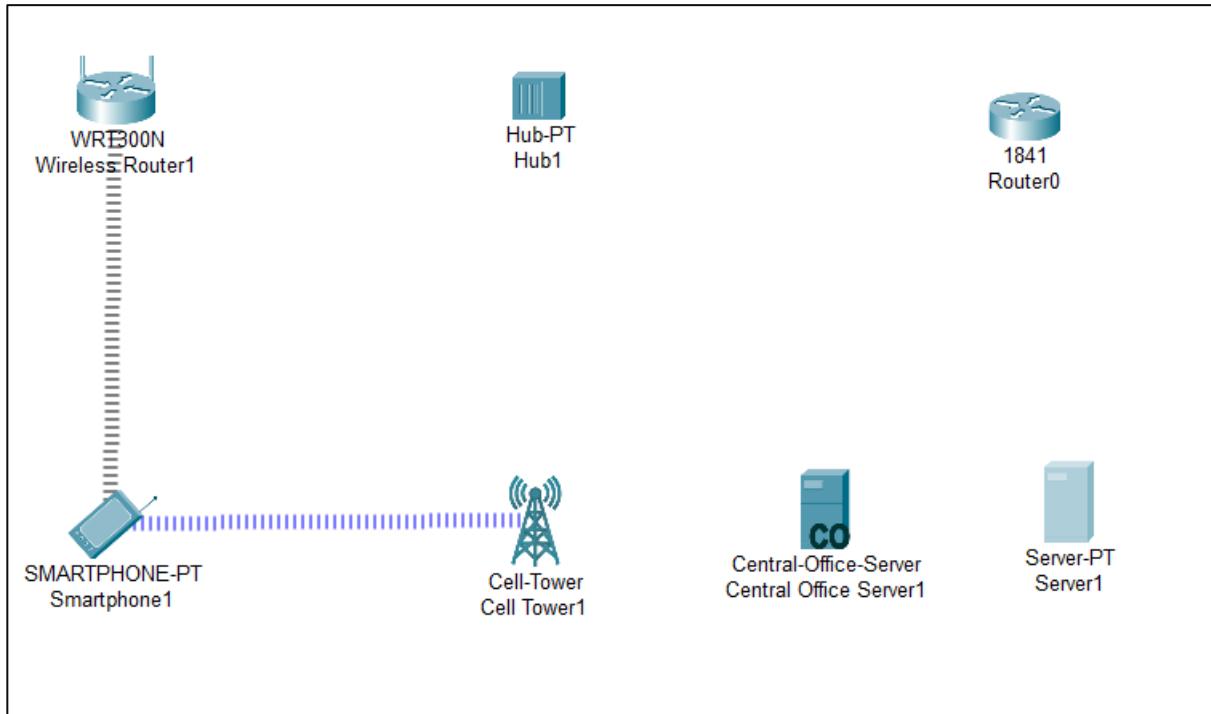
#### B) DESCRIPTION:

An ad hoc network is one that is spontaneously formed when devices connect and communicate with each other. The term ad hoc is a Latin word that literally means "for this," implying improvised or impromptu.

Ad hoc networks are mostly wireless local area networks (WLANs). The devices communicate with each other directly instead of relying on a base station or access points as in wireless LANs for data transfer co-ordination. Each device participates in routing activity, by determining the route using the routing algorithm and forwarding data to other devices via this route.

## C) NETWORK TOPOLOGY, CONFIGURATIONS AND OUTPUT:

### Network topology



### Configurations:

#### For 1841 Router:

Router1

Physical **Config** CLI Attributes

**GLOBAL**

- Settings
- Algorithm Settings

**ROUTING**

- Static
- RIP

**SWITCHING**

- VLAN Database

**INTERFACE**

- FastEthernet0/0**
- FastEthernet0/1

**FastEthernet0/0**

Port Status ☒ On

Bandwidth ☐ 100 Mbps ☐ 10 Mbps ☒ Auto

Duplex ☐ Half Duplex ☐ Full Duplex ☒ Auto

MAC Address 0010.11AE.2901

IP Configuration

IPv4 Address 20.0.0.1

Subnet Mask 255.0.0.0

Tx Ring Limit 10



Router1

Physical
Config
CLI
Attributes

GLOBAL

Settings

Algorithm Settings

ROUTING

Static

RIP

SWITCHING

VLAN Database

INTERFACE

FastEthernet0/0

FastEthernet0/1

FastEthernet0/1

Port Status
Bandwidth
Duplex
MAC Address

100 Mbps

10 Mbps

Half Duplex

Full Duplex

On

Auto

Auto

0010.11AE.2902

IP Configuration

IPv4 Address

Subnet Mask

21.0.0.1

255.0.0.0

Tx Ring Limit

10

For Server-Pt:

Server0

Physical
Config
Services
Desktop
Programming
Attributes

GLOBAL

Settings

Algorithm Settings

INTERFACE

FastEthernet0

FastEthernet0

Port Status
Bandwidth
Duplex
MAC Address

100 Mbps

10 Mbps

Half Duplex

Full Duplex

On

Auto

Auto

00E0.B066.D86E

IP Configuration

DHCP

Static

IPv4 Address

Subnet Mask

21.0.0.2

255.0.0.0

IPv6 Configuration

Automatic

Static

IPv6 Address

Link Local Address: FE80::2E0:B0FF:FE66:D86E

Name of Instructor: Ms. Jessica D'cruz

## Output

