

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

**CERTIFICATE**

This is to certify that Miss./Mr **AJAY KUMAR UTHAYA KUMAR**  
Roll No. **TCS2324002** 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  
**Prof. Manoj Singh**

College Seal

**Name of Instructor: Jescia D'cruz**

## Index Page

Sr. No	Description	Page No	Date	Faculty Signature
1	Understanding the Sensor Node Hardware. (For Eg. Sensors, Node(Sensor mote), Base Station, Graphical User Interface	3	08/1/24	
2A	Exploring and Understanding TinyOS computational concepts – Events, Commands and Task. <ul style="list-style-type: none"> <li>• nesC model</li> <li>• nesC Components</li> </ul>	7	08/1/24	
2B	Exploring and Understanding TOSSIM computational concepts – Events, Commands and Task	11	08/1/24	
3A	Design smoke detection and fire prevention system using Cisco Packet Tracer	14	08/1/24	
3B	Design garden sprinkler system using Cisco Packet Tracer	24	15/1/24	
4	Create and simulate a simple adhoc network	33	05/2/24	
5	Understanding, Reading and Analyzing Routing Table of a network	39	29/1/24	
6	Implement a wireless sensor network simulation	47	05/2/24	
7	Create a MAC Protocol simulation implementation for wireless sensor network	55	06/2/24	
8	Stimulate a Mobile Adhoc network with directional antennas	58	06/2/24	
9	Create a mobile network using Cell Tower, Central office server, Web Browser and Web Server, Stimulate connection between them.	63	06/2/24	



# Wireless Sensor Networks & Mobile Communication

## Practical No.1

### DEPARTMENT OF COMPUTER SCIENCE

<b>Name:</b>	Ajay Kumar Uthaya Kumar	<b>Roll Number</b>	TCS2324002
<b>Paper Code:</b>	SIUSCS61	<b>Class</b>	TYBSc(Computer Science)
<b>Topic:</b>	Sensor Node Hardware	<b>Batch</b>	I
<b>Date:</b>	08/1/24	<b>Practical No</b>	1

**A) AIM:** Understanding the Sensor Node Hardware. (For Eg. Sensors, Node(Sensor mote), Base Station, Graphical User Interface)

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

A sensor is a device or module that detects and responds to some type of input from the physical environment. It translates physical phenomena like light, heat, sound, motion, etc., into measurable signals, often electrical, which can be further processed and analysed.

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

### **Theory**

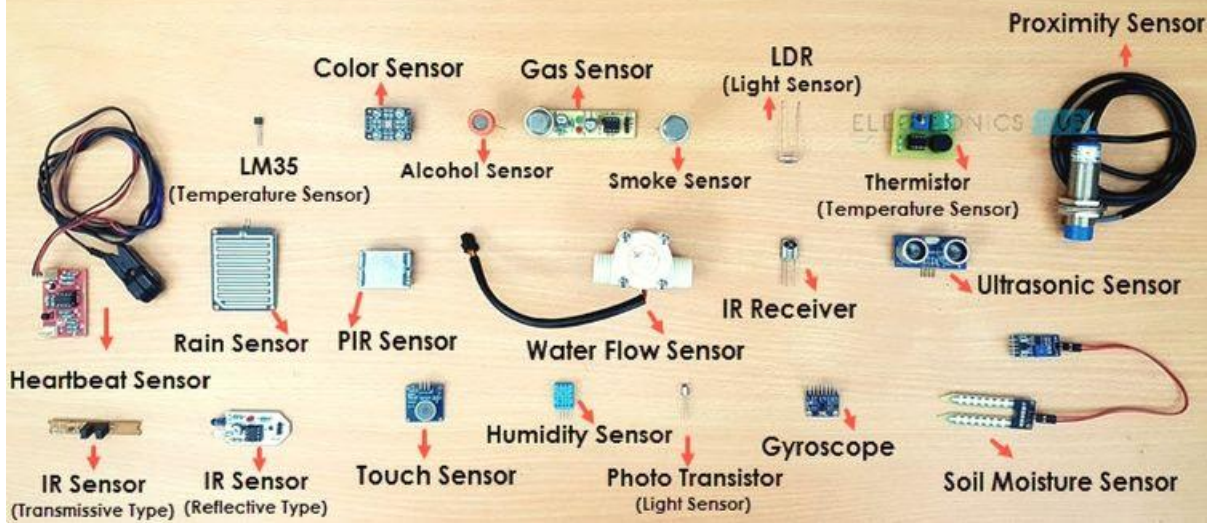
A sensor node is a self-contained unit equipped with sensors, processing capabilities, and communication interfaces. It's typically part of a larger sensor network where multiple sensor nodes collaborate to collect and transmit data. The hardware components of a sensor node include:

- a. **Sensors:** These are the primary components responsible for detecting environmental parameters such as temperature, humidity, pressure, light intensity, etc. Sensors come in various types and technologies, each suited for specific applications.
- b. **Node (Sensor Mote):** The node or sensor mote is the central component housing the sensors, processing unit, memory, and communication interfaces. It's often a compact and energy-efficient device designed for deployment in diverse environments, including remote and harsh locations.
- c. **Base Station:** The base station serves as the central hub or gateway for collecting data from multiple sensor nodes. It typically has more processing power and storage capacity compared to individual sensor nodes. The base station aggregates data, performs further analysis if required, and may also facilitate communication with external networks or systems.
- d. **Graphical User Interface (GUI):** The GUI provides a user-friendly interface for interacting with the sensor network. It allows users to visualize real-time data, configure settings, monitor sensor statuses, and analyze collected data through intuitive graphical representations. GUIs can be desktop applications, web-based dashboards, or mobile apps, depending on the requirements and accessibility preferences.

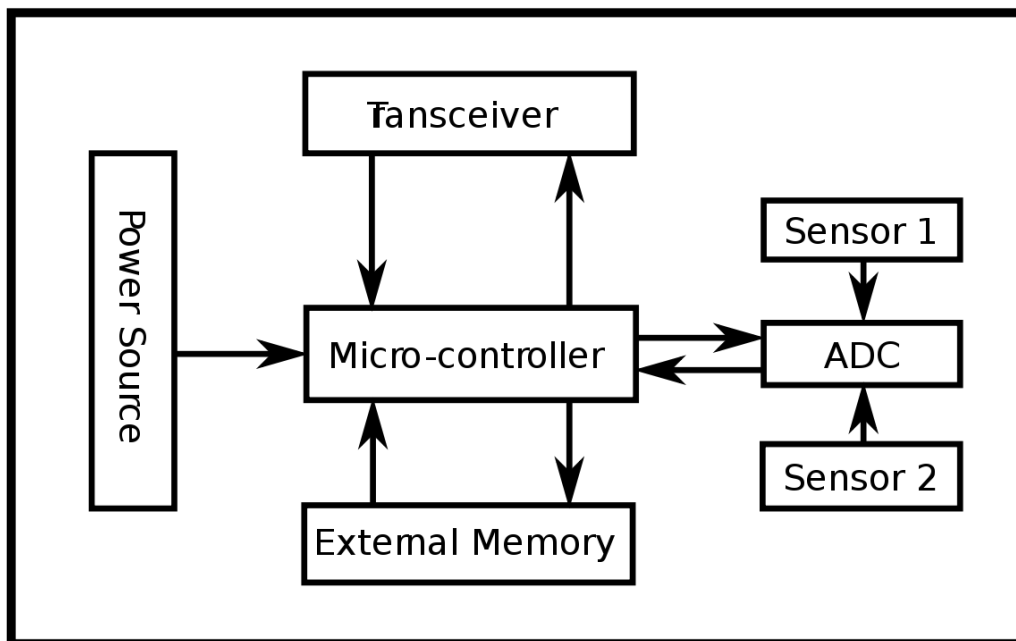
### **Output**

1. Sensors

# DIFFERENT TYPES OF SENSORS



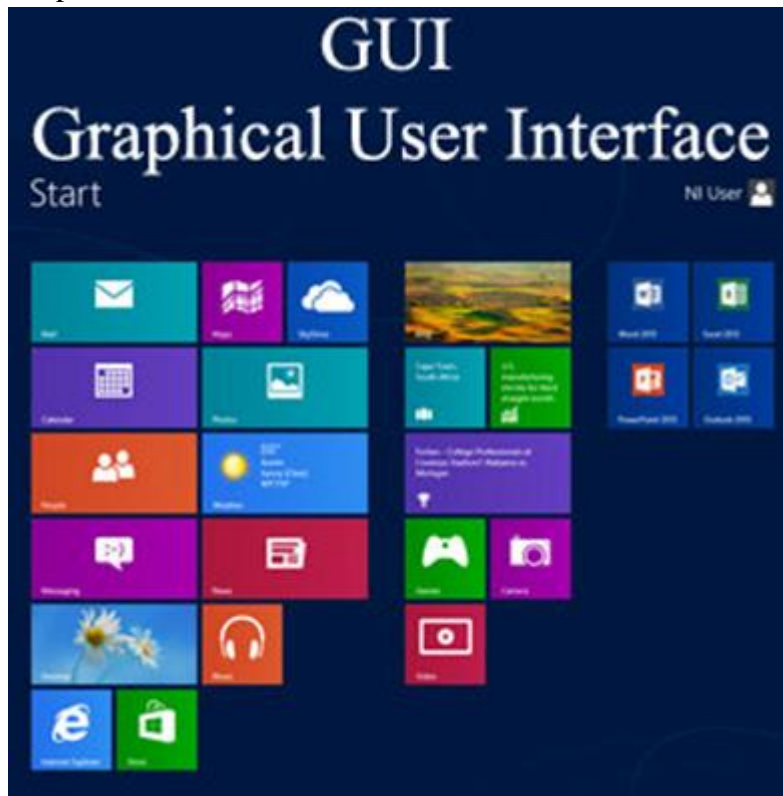
## 2. Node(Sensor Mote)



## 3. Base Station



#### 4. Graphical User Interface





# Wireless Sensor Networks & Mobile Communication

## Practical No.2A

### DEPARTMENT OF COMPUTER SCIENCE

<b>Name:</b>	Ajay Kumar Uthaya Kumar	<b>Roll Number</b>	TCS2324002
<b>Paper Code:</b>	SIUSCS61	<b>Class</b>	TYBSc(Computer Science)
<b>Topic:</b>	TinyOS computational concept	<b>Batch</b>	I
<b>Date:</b>	08/1/24	<b>Practical No</b>	2A

**A) AIM:** Exploring and Understanding TinyOS computational concepts – Events, Commands and Task.

- nesC model
- nesC Components

#### **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. It began as a collaboration between the University of California, Berkeley, Intel Research, and Crossbow Technology, was released as free and open-source software under a BSD license, and has since grown into an international consortium, the TinyOS Alliance. TinyOS has been used in space, being implemented in ESTCube-1.

## C) NETWORK TOPOLOGY, CONFIGURATIONS AND OUTPUT:

### Theory

TinyOS is an open-source operating system designed specifically for embedded systems and wireless sensor networks. It follows a component-based programming model and is primarily programmed using the nesC (network embedded systems C) language. Understanding the computational concepts in TinyOS, including Events, Commands, and Tasks, is crucial for effectively developing applications for resource-constrained devices.

#### 1. Events

- In TinyOS, Events are asynchronous signals or notifications that indicate the occurrence of specific conditions or events within the system.
- Events are typically generated by hardware interrupts, software components, or other external stimuli.
- They serve as triggers for initiating actions or executing code in response to certain events.
- Event handlers, also known as event-driven routines or functions, are registered to handle specific events.
- When an event occurs, the corresponding event handler is invoked to process the event.
- Events are essential for implementing reactive behavior and event-driven programming paradigms in TinyOS applications.
- Example: An event might be generated when a sensor reading is available, when a packet is received over the wireless network, or when a timer expires.

#### 2. Commands

- Commands in TinyOS represent synchronous operations or actions that can be invoked by software components to perform specific tasks or interact with hardware peripherals.
- Unlike events, commands are typically executed in a synchronous manner, meaning that the caller may block until the command execution completes.
- Commands encapsulate functionality and behavior that can be reused across different parts of the application.
- Components expose a set of commands that can be invoked by other components or application code to perform various operations.
- Example: Commands might include functions to configure sensor parameters, transmit data over the network, or control actuators in the environment.

#### 3. Tasks

- Tasks in TinyOS represent units of concurrent execution or lightweight threads of control within the system.
- Tasks are used to perform background processing, handle asynchronous events, and execute command sequences in a sequential manner.
- Each task typically corresponds to a specific functionality or operation within the application. Tasks are scheduled and managed by the TinyOS scheduler, which ensures that tasks are



executed in a timely and efficient manner, considering resource constraints and system priorities.

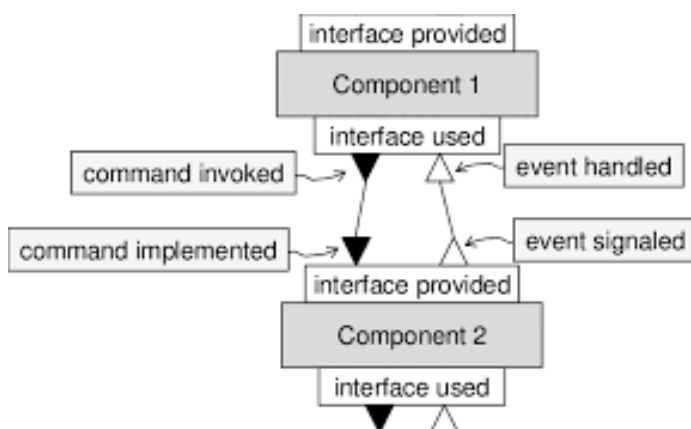
- Tasks can be initiated by events, commands, or other tasks, and they may communicate with each other through shared data structures or message passing.
- Example: A task might be responsible for periodically sampling sensor data, processing incoming packets from the network, or performing periodic maintenance tasks.

**nesC Model:** The nesC programming language is specifically designed for programming embedded systems and sensor networks. It follows a component-based model, where software functionality is organized into reusable and composable components. Components encapsulate related functionality, including event handlers, command interfaces, task implementations, and internal state. Components can interact with each other through well-defined interfaces, exchanging events, commands, and data. The nesC compiler translates nesC code into efficient C code, which is then compiled and executed on the target platform. nesC promotes modular design, code reuse, and maintainability, making it well-suited for developing complex applications for resource-constrained devices.

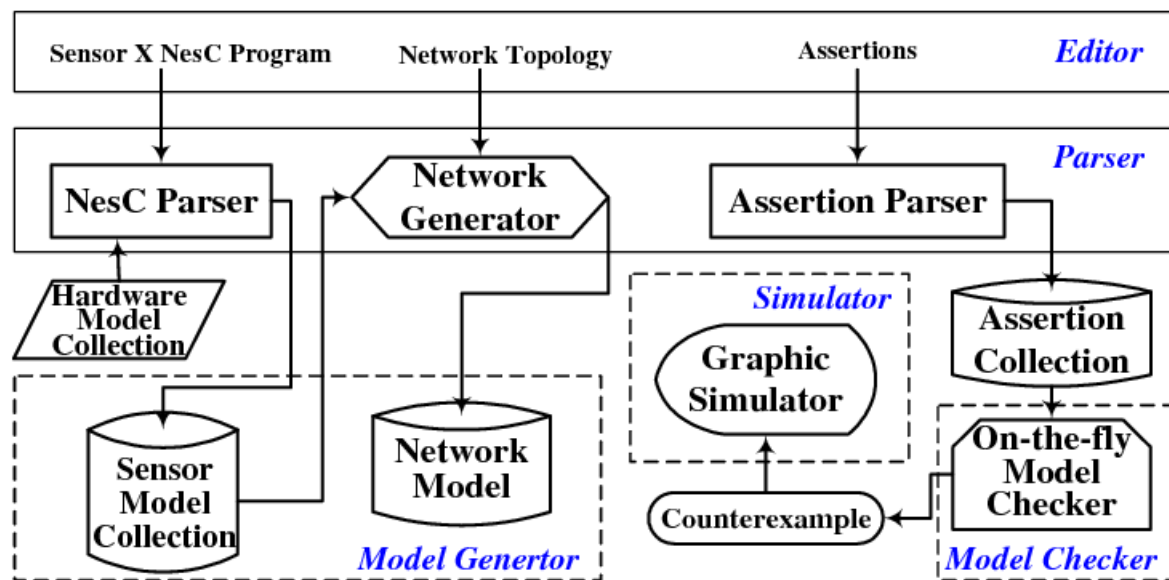
**nesC Components:** Components in nesC represent modular units of functionality that can be composed and interconnected to build larger applications. Each component encapsulates a coherent set of functionality, including event handlers, command interfaces, tasks, and internal state. Components define interfaces that specify the events they emit, the commands they provide, and the configuration parameters they accept. Components can be instantiated multiple times with different configurations, allowing for flexible and configurable application design. Components can communicate with each other through interface connections, enabling message passing, event propagation, and command invocation. Components can be developed independently, tested in isolation, and then integrated into larger systems, promoting code modularity and reusability. Example: A sensor component might expose events for new sensor readings, commands for configuring sensor parameters, and tasks for periodic data sampling.

## Output

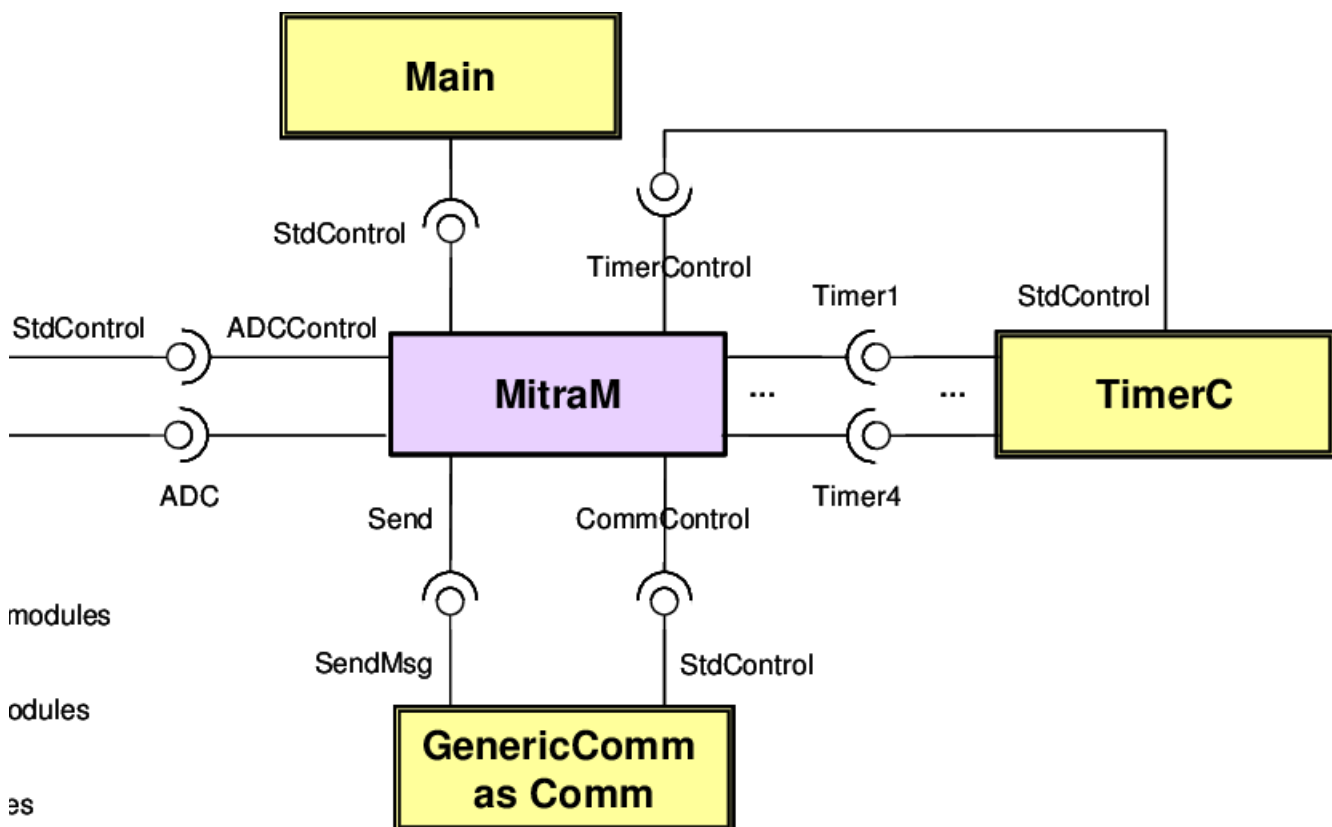
TinyOS



## nesC model



## nesC Component





# Wireless Sensor Networks & Mobile Communication

## Practical No.2B

### DEPARTMENT OF COMPUTER SCIENCE

<b>Name:</b>	Ajay Kumar Uthaya Kumar	<b>Roll Number</b>	TCS2324002
<b>Paper Code:</b>	SIUSCS61	<b>Class</b>	TYBSc(Computer Science)
<b>Topic:</b>	Tossim computational concept	<b>Batch</b>	I
<b>Date:</b>	08/1/24	<b>Practical No</b>	2B

**A) AIM:** Exploring and Understanding Tossim computational concepts – Events, Commands and Task.

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

Tossim Simulator is a discrete event simulator framework for TinyOS wireless sensor network. Instead of NS2, we implement Tossim, which captures the network behavior and attraction based on bit granularity, not on the packet level. It operate in a sensor network called motes, which deployed from IEEE based papers. Motes are referred to as tiny sensing and computation device for limited communication, computation, and also energy resources.

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

### **Theory**

TinyOS is an open-source operating system designed specifically for embedded systems and wireless sensor networks. It follows a component-based programming model and is primarily programmed using the nesC (network embedded systems C) language. Understanding the computational concepts in TinyOS, including Events, Commands, and Tasks, is crucial for effectively developing applications for resource-constrained devices.

#### **1. Events**

- In Tossim, events represent occurrences or incidents that trigger some action or response in the simulation.
- These events could be packet transmissions, receptions, node movements, changes in network topology, etc.
- Events are typically scheduled to occur at specific simulation times and can influence the behavior of the simulated network.

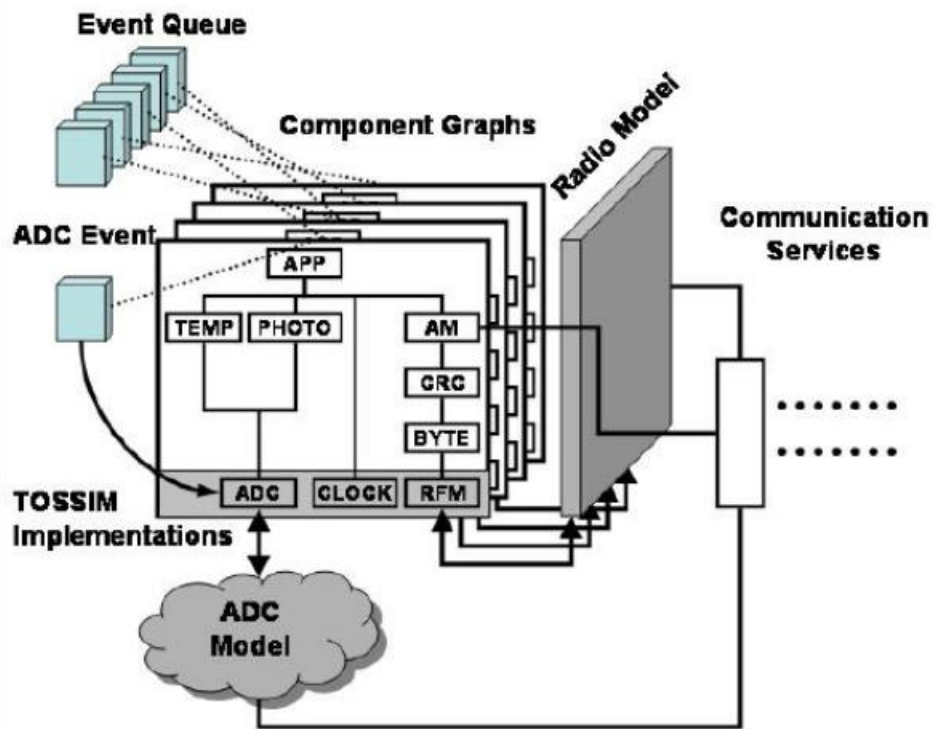
#### **2. Commands**

- Commands in Tossim are instructions or actions issued to control various aspects of the simulation.
- These commands can be used to start, pause, resume, or stop the simulation, modify simulation parameters, observe simulation state, etc.
- Users can issue commands through a command-line interface or scripting environment to interact with the simulation dynamically.

#### **3. Tasks**

- Tasks in Tossim are units of computation or activities that are executed by individual nodes in the simulated network.
- These tasks represent the functionality or operations performed by nodes, such as processing received packets, executing application logic, updating routing tables, etc.
- Tasks are typically scheduled and executed asynchronously based on events or timers within the simulation environment.

### **Output**





# Wireless Sensor Networks & Mobile Communication

## Practical No.3A

### DEPARTMENT OF COMPUTER SCIENCE

<b>Name:</b>	Ajay Kumar Uthaya Kumar	<b>Roll Number</b>	TCS2324002
<b>Paper Code:</b>	SIUSCS61	<b>Class</b>	TYBSc(Computer Science)
<b>Topic:</b>	Smoke detection and fire prevention system	<b>Batch</b>	I
<b>Date:</b>	08/1/24	<b>Practical No</b>	3A

**A) AIM:** Design smoke detection and fire prevention system using Cisco Packet Tracer

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

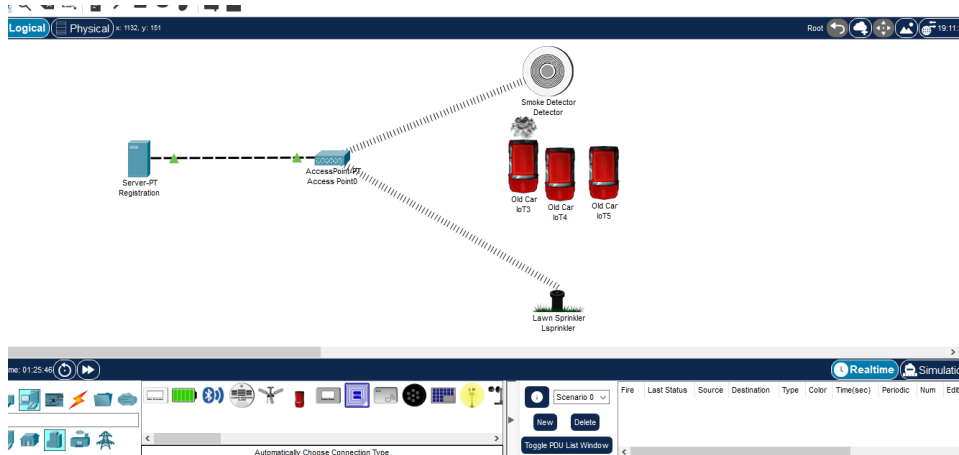
A smoke detection and fire prevention system is a crucial component of building safety infrastructure designed to detect the presence of smoke or fire and take appropriate action to mitigate the risk of fire-related incidents. Below is a detailed description outlining its components:

Components:

- **Smoke Detectors:** These devices are equipped with sensors that detect the presence of smoke particles in the air. There are different types of smoke detectors, including ionization, photoelectric, and dual-sensor detectors.
- **Fire Alarms:** Fire alarms are audible and visual alert systems triggered by smoke detectors or heat sensors. They notify occupants of the building about the presence of smoke or fire, allowing them to evacuate safely.
- **Heat Sensors:** Heat sensors detect changes in temperature, indicating the presence of fire. They complement smoke detectors by providing an additional layer of fire detection.
- **Sprinkler Systems:** Automatic sprinkler systems are activated in response to a fire alarm or heat detection, releasing water to suppress the fire and prevent its spread.

## C) NETWORK TOPOLOGY, CONFIGURATIONS AND OUTPUT:

Network topology(only for cisco packet tracer practical's):



Configurations:

For Server:

Registration

Physical Config **Services** Desktop Programming Attributes

**SERVICES**

- HTTP
- DHCP
- DHCPv6
- TFTP
- DNS
- SYSLOG
- AAA
- NTP
- EMAIL
- FTP
- IoT**
- VM Management
- Radius EAP

Registration Server

This service runs on top of the HTTP or HTTPS service.

Service ☒ On ☐ Off

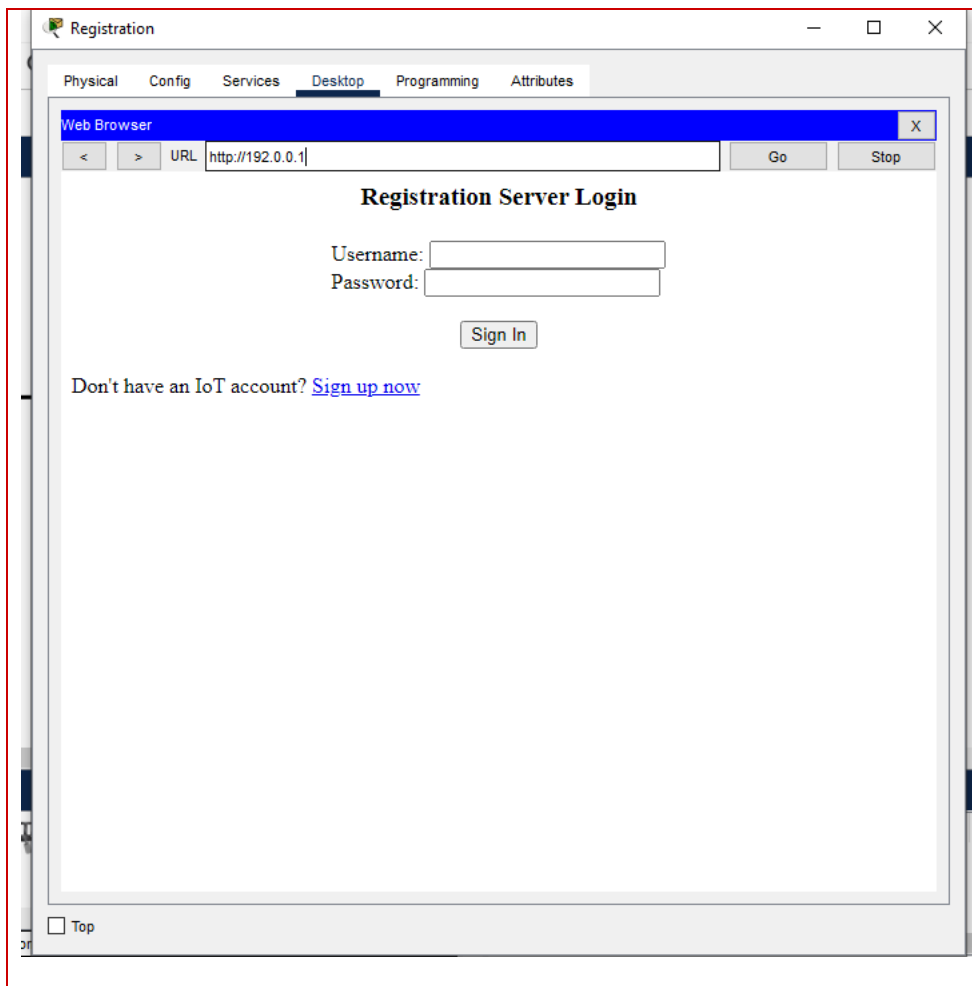
	Username	Password
1	Admin	123

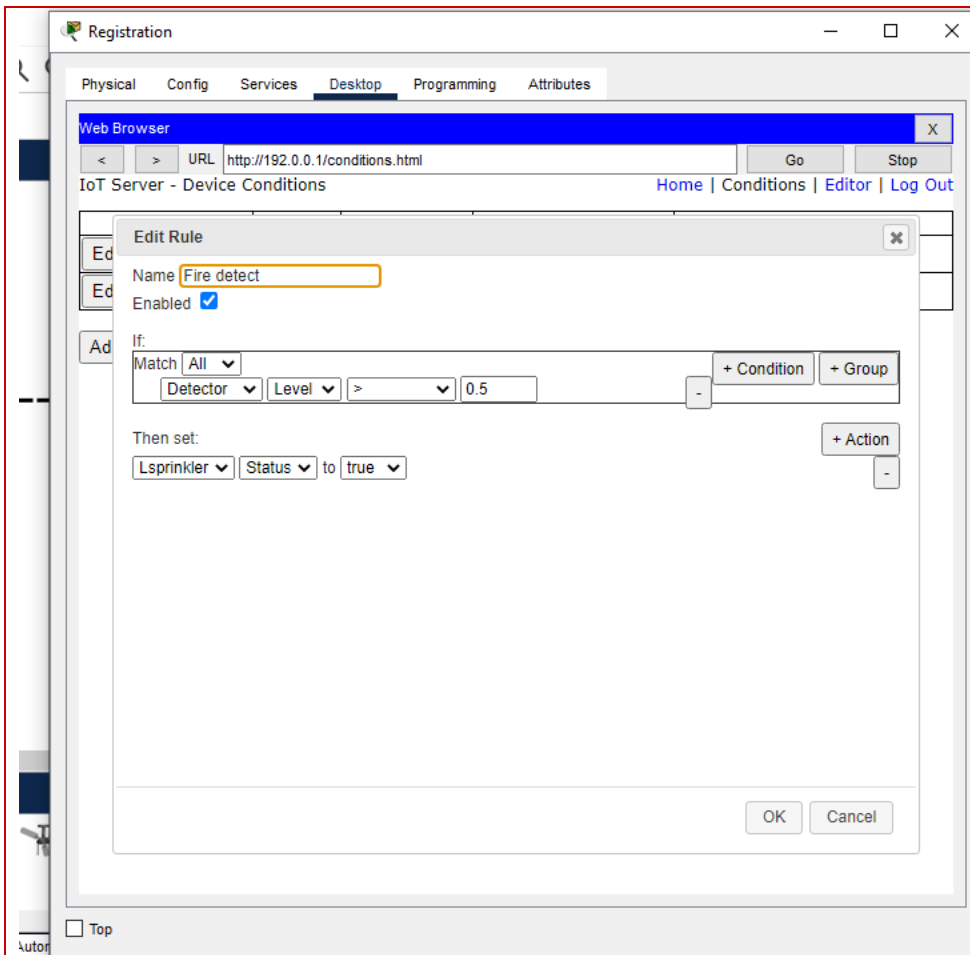
Delete

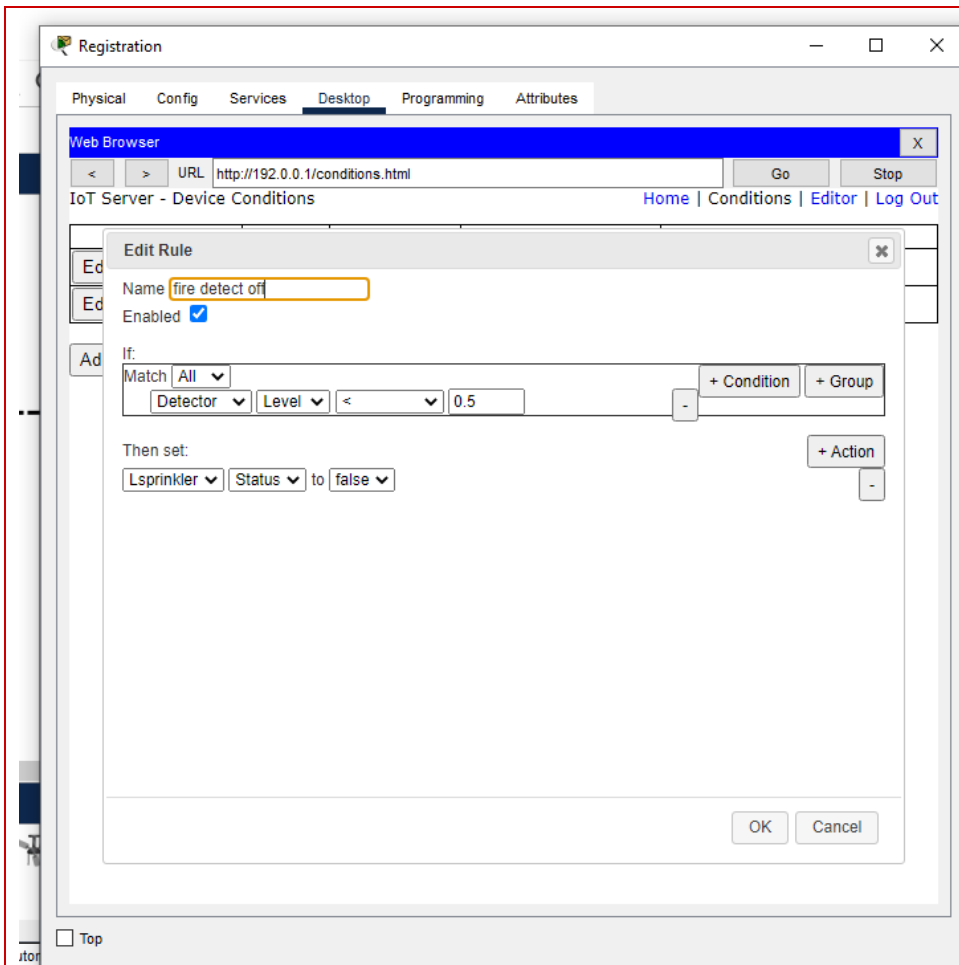
☐ Top



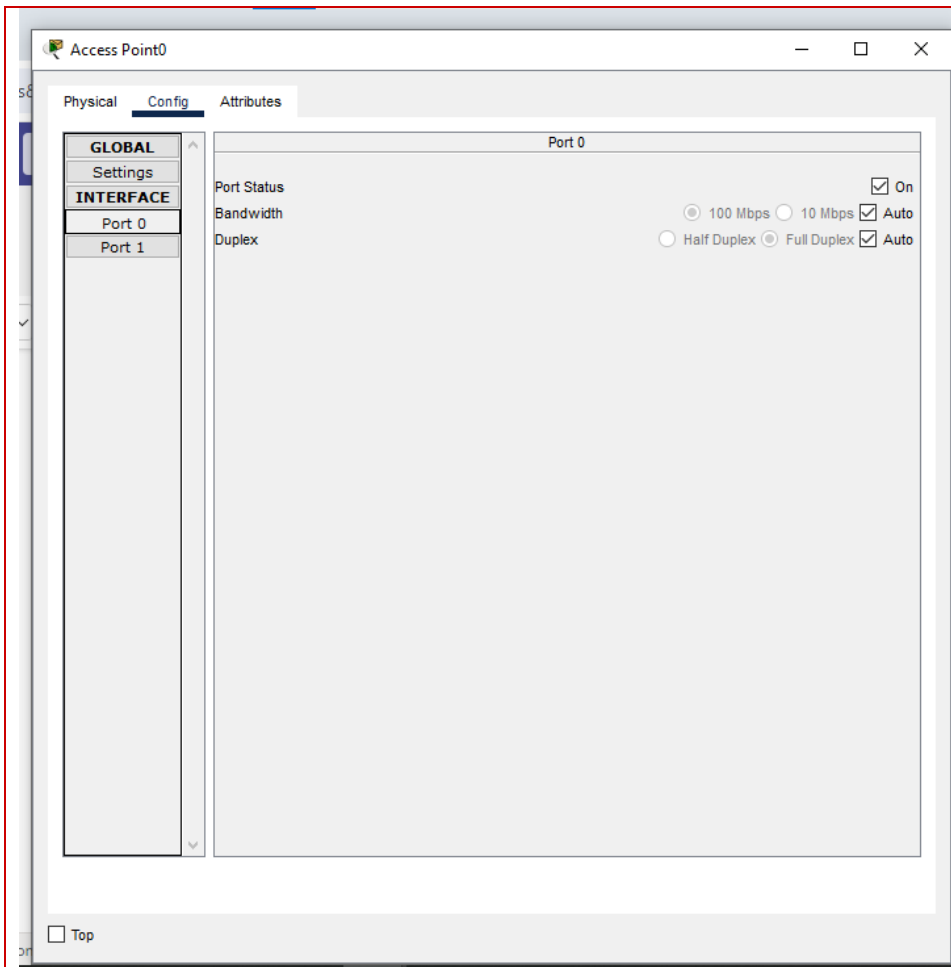








For Access-Point



Access Point0

Physical Config Attributes

GLOBAL

Settings

INTERFACE

Port 0

Port 1

Port 1

Port Status ☒ On

SSID Admin

2.4 GHz Channel 6

Coverage Range (meters) 140.00

Authentication

☒ Disabled ☐ WEP ☐ WPA-PSK ☐ WPA2-PSK

WEP Key

PSK Pass Phrase

User ID

Password

Encryption Type Disabled

☐ Top

For Detector

Detector

Specifications Physical **Config** Attributes

**GLOBAL**

- Settings
- Algorithm Settings
- Files

**INTERFACE**

- Wireless0**
- Bluetooth

**Wireless0**

Port Status ☒ On

Bandwidth 18 Mbps

MAC Address 0006.2A98.70A1

SSID Admin

Authentication

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

WEP Key

PSK Pass Phrase

User ID

Password

Method: MD5

User Name

Password

Encryption Type Disabled

IP Configuration

☐ DHCP ☒ Static

IPv4 Address 192.0.0.2

Subnet Mask 255.255.255.0

IPv6 Configuration

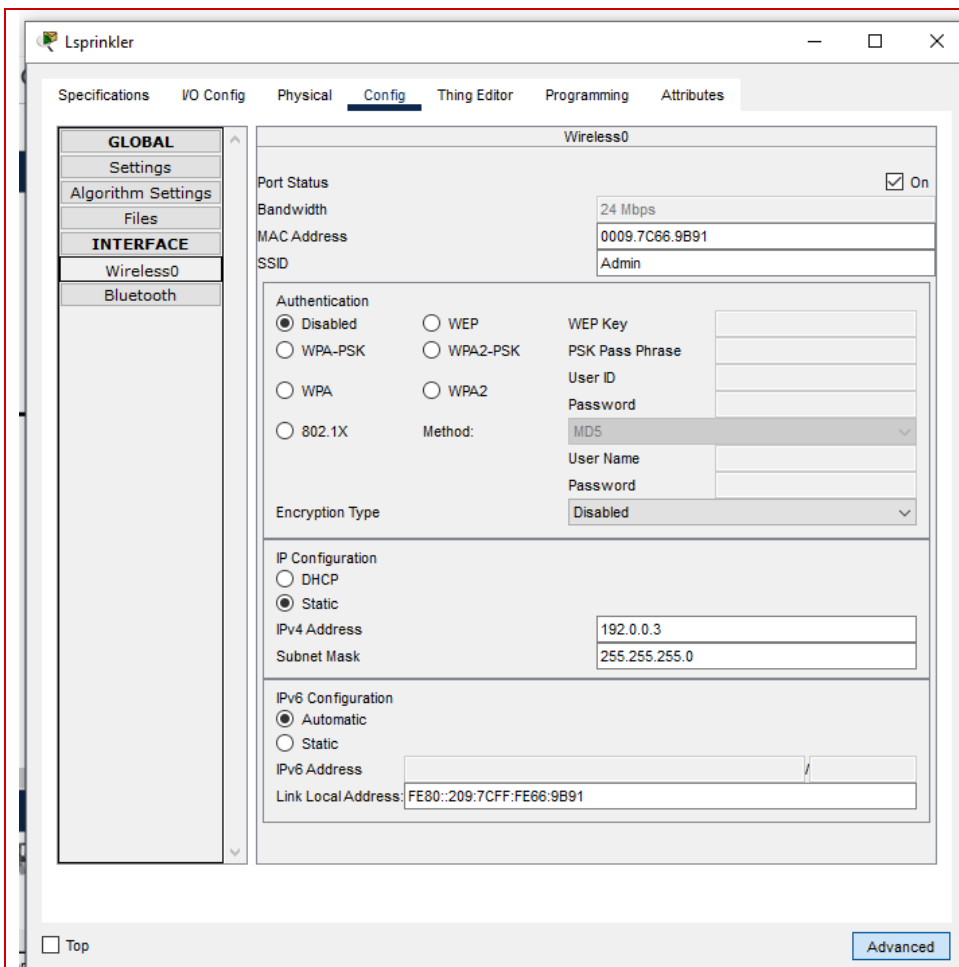
☒ Automatic ☐ Static

IPv6 Address

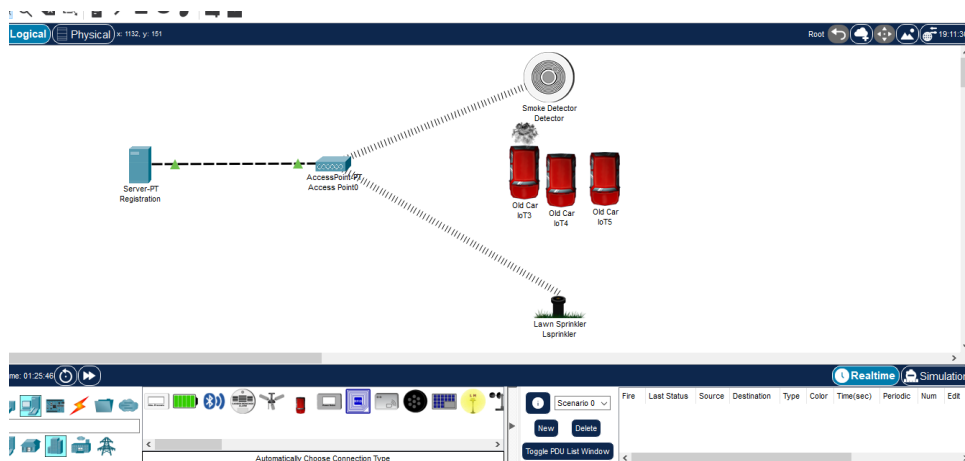
Link Local Address FE80::206:2AFF:FE98:70A1

☐ Top

For LSprinkler



## Output





## Wireless Sensor Networks & Mobile Communication

### Practical No.3B

#### DEPARTMENT OF COMPUTER SCIENCE

<b>Name:</b>	Ajay Kumar Uthaya Kumar	<b>Roll Number</b>	TCS2324002
<b>Paper Code:</b>	SIUSCS61	<b>Class</b>	TYBSc(Computer Science)
<b>Topic:</b>	Garden Sprinkler	<b>Batch</b>	I
<b>Date:</b>	15/1/24	<b>Practical No</b>	3B

**A) AIM:** Design garden sprinkler system using Cisco Packet Tracer

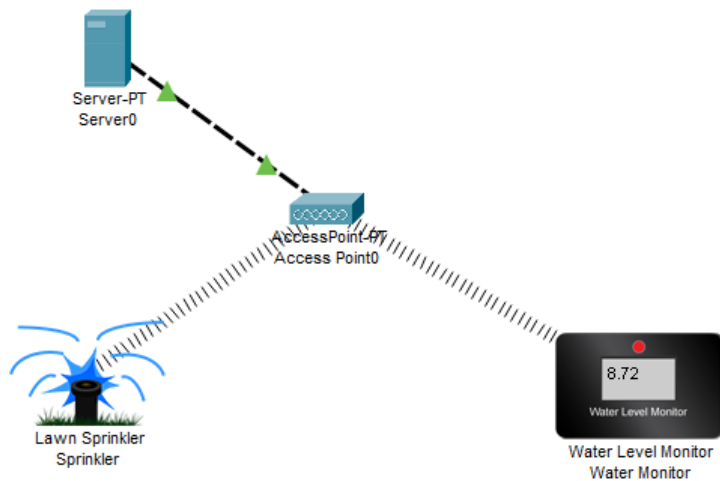
#### **B) DESCRIPTION:**

An irrigation sprinkler is a device used to irrigate (water) agricultural crops, lawns, landscapes, golf courses, and other areas. They are also used for cooling and for the control of airborne dust. Sprinkler irrigation is the method of applying water in a controlled manner in way similar to rainfall. The water is distributed through a network that may consist of pumps, valves, pipes, and sprinklers. Irrigation sprinklers can be used for residential, industrial, and agricultural usage. It is useful on uneven land where sufficient water is not available as well as on sandy soil. The perpendicular pipes, having rotating nozzles on top, are joined to the main pipeline at regular intervals. When water is pressurized through the main pipe it escapes from the rotating nozzles. It gets sprinkled on the crop. In sprinkler or overhead irrigation, water is piped to one more central locations within the field and distributed by overhead high pressure sprinklers or guns.



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

Network topology(only for cisco packet tracer practical's):



Configurations:

Server0

PhysicalConfigServicesDesktopProgrammingAttributes

GLOBAL

Settings

Algorithm Settings

INTERFACE

FastEthernet0

FastEthernet0

Port Status

☒ On

Bandwidth

☒ 100 Mbps

☐ 10 Mbps

☒ Auto

Duplex

☒ Half Duplex

☐ Full Duplex

☒ Auto

MAC Address

0001.C709.2BE8

IP Configuration

☐ DHCP

☒ Static

IPv4 Address

192.0.0.1

Subnet Mask

255.255.255.0

IPv6 Configuration

☐ Automatic

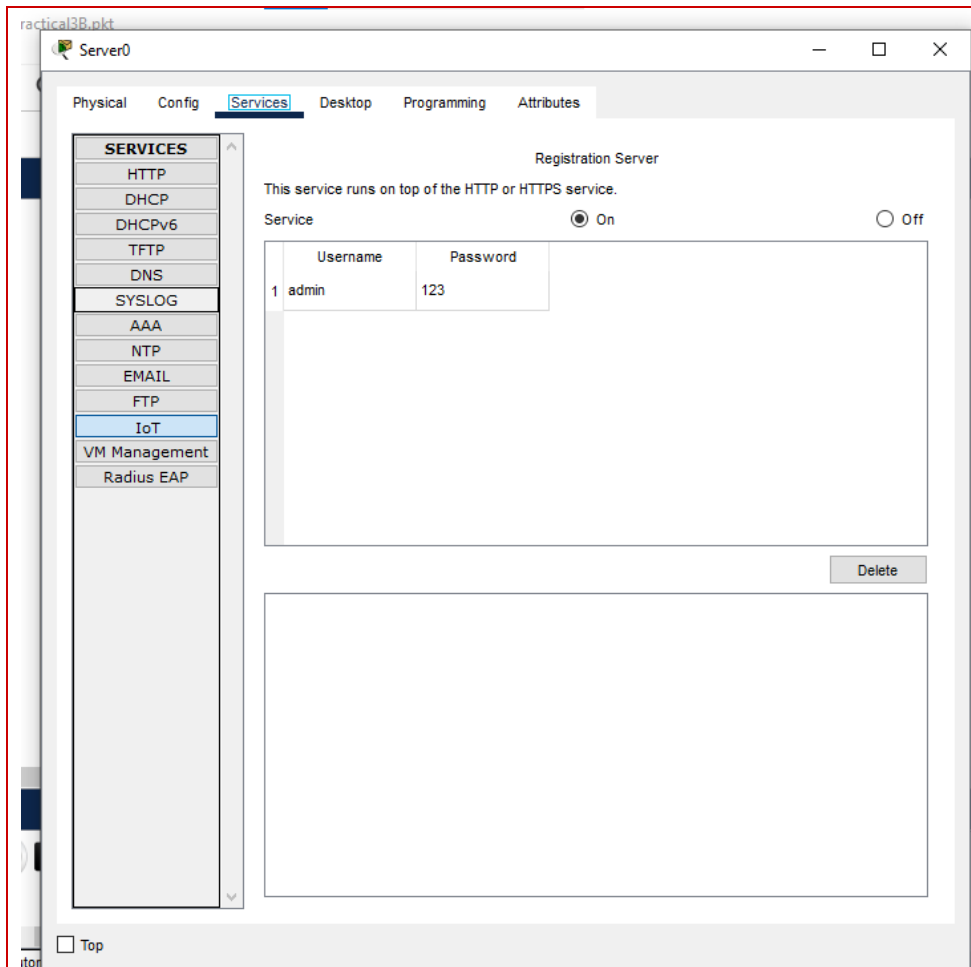
☒ Static

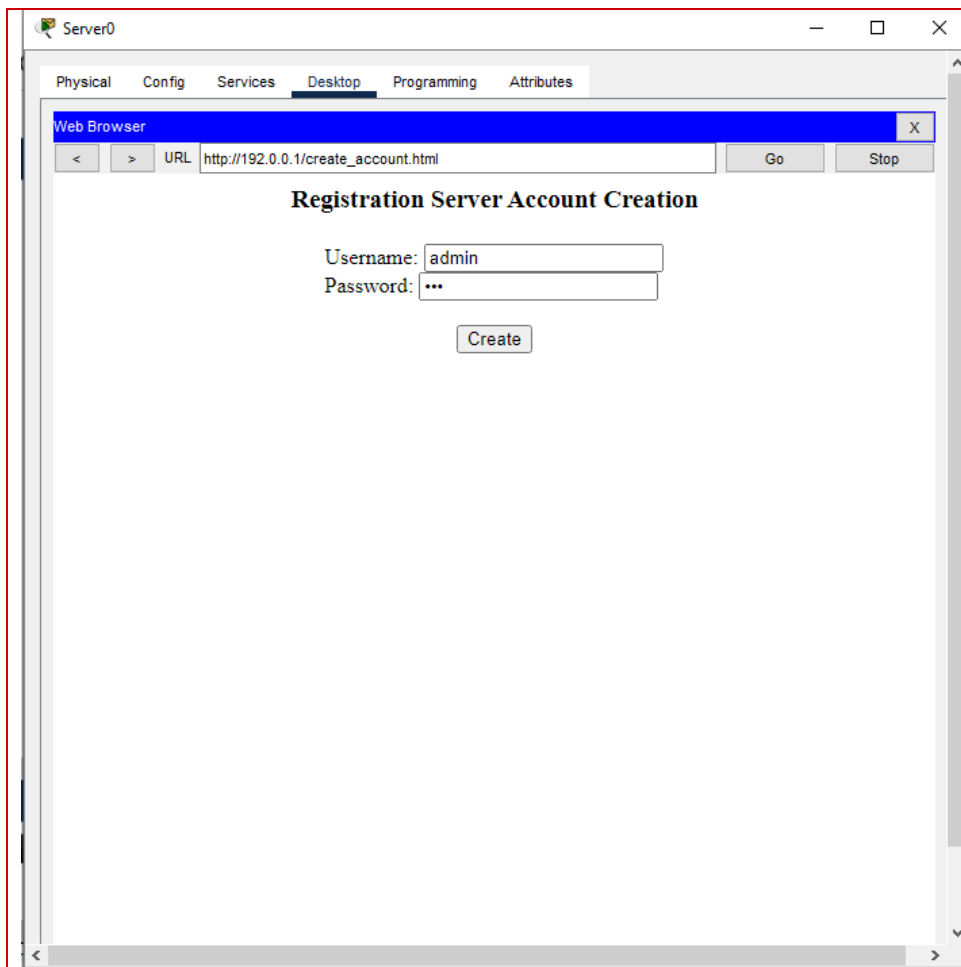
IPv6 Address

Link Local Address

FE80::201:C7FF:FE09:2BE8

☐ Top







Water Monitor

Specifications Physical **Config** Attributes

**GLOBAL**

Settings

Algorithm Settings

Files

**INTERFACE**

Wireless0

Bluetooth

**Wireless0**

Port Status ☒ On

Bandwidth 24 Mbps

MAC Address 000C:CF8E:6B54

SSID Admin

Authentication

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

WEP Key

PSK Pass Phrase

User ID

Password

Method: MD5

User Name

Password

Encryption Type Disabled

IP Configuration

☐ DHCP ☒ Static

IPv4 Address 192.0.0.3

Subnet Mask 255.255.255.0

IPv6 Configuration

☒ Automatic ☐ Static

IPv6 Address

Link Local Address: FE80::20C:CFFF:FE8E:6B54

Water Monitor

Specifications Physical **Config** Attributes

**GLOBAL**

Settings

Algorithm Settings

Files

**INTERFACE**

Wireless0

Bluetooth

Display Name Water Monitor

Serial Number PTT0810QKX

Interfaces Wireless0

Gateway/DNS IPv4

☐ DHCP ☒ Static

Default Gateway

DNS Server

Gateway/DNS IPv6

☒ Automatic ☐ Static

Default Gateway

DNS Server

MT Server

☐ None ☐ Home Gateway ☒ Remote Server

Server Address 192.0.0.1

User Name admin

Password 123

Top Advanced

Condition

Name of Instructor: Jescia D'cruz

Edit Rule

Name

Sprinkler On

Enabled

☒

If:

Match

All

+ Condition

+ Group

Water Monitor

Water Level

<=

10.0

cm

-

Then set:

+ Action

Sprinkler

Status

to

true

-

OK

Cancel

Edit Rule

Name

Sprinkler off

Enabled

☒

If:

Match

All

+ Condition

+ Group

Water Monitor

Water Level

>

10.0

cm

-

Then set:

+ Action

Sprinkler

Status

to

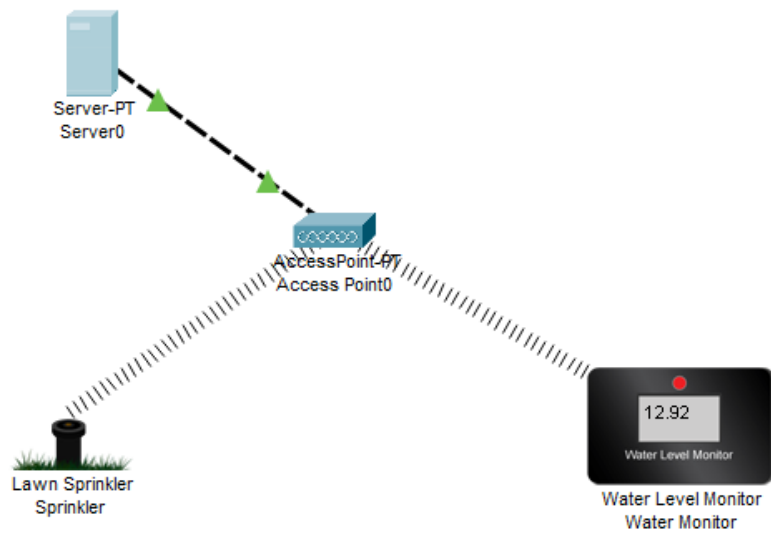
false

-

OK

Cancel

Output







# Wireless Sensor Networks & Mobile Communication

## Practical No.4

### DEPARTMENT OF COMPUTER SCIENCE

<b>Name:</b>	Ajay Kumar Uthaya Kumar	<b>Roll Number</b>	TCS2324002
<b>Paper Code:</b>	SIUSCS61	<b>Class</b>	TYBSc(Computer Science)
<b>Topic:</b>	Adhoc Network	<b>Batch</b>	I
<b>Date:</b>	05/2/24	<b>Practical No</b>	4

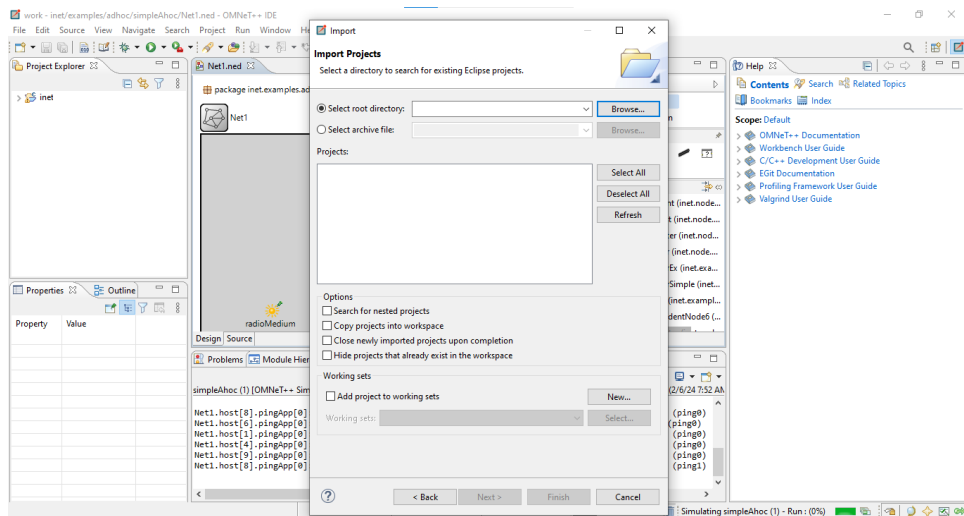
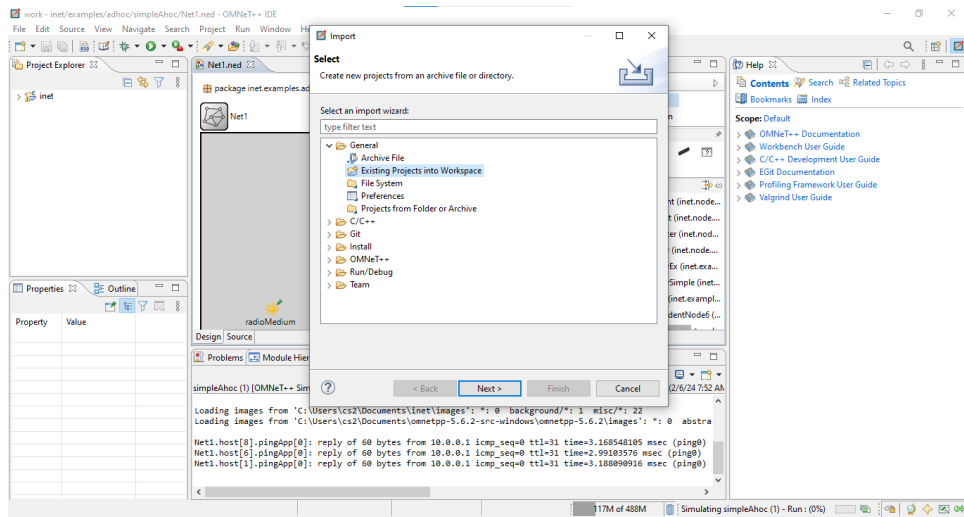
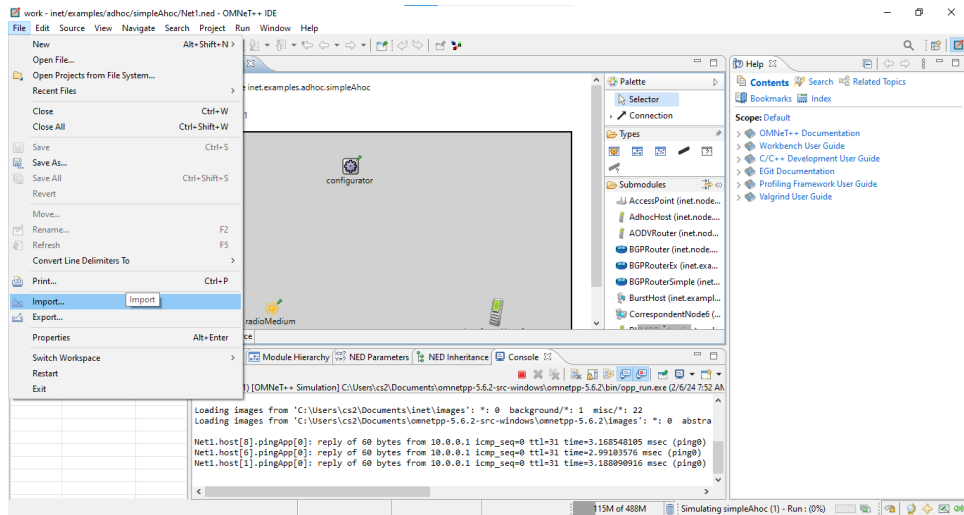
**A) AIM:** Create and simulate a simple adhoc network

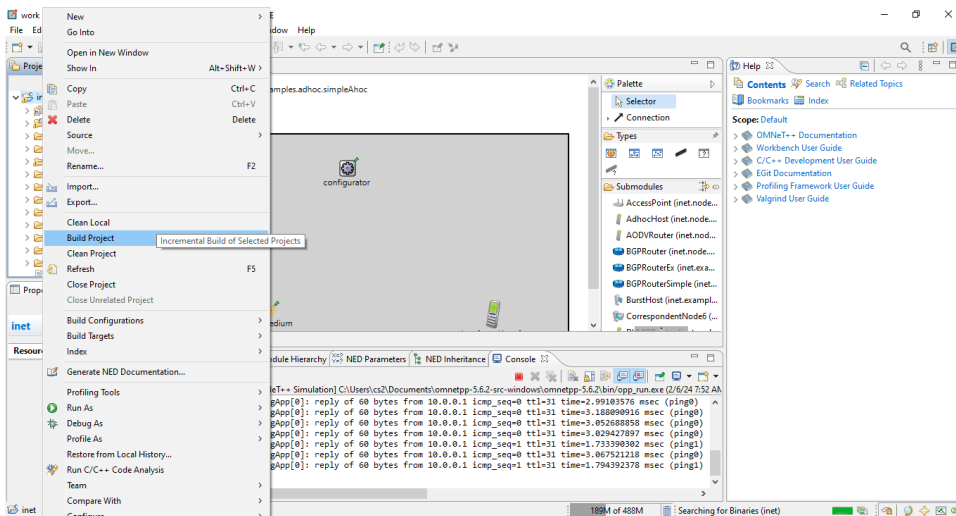
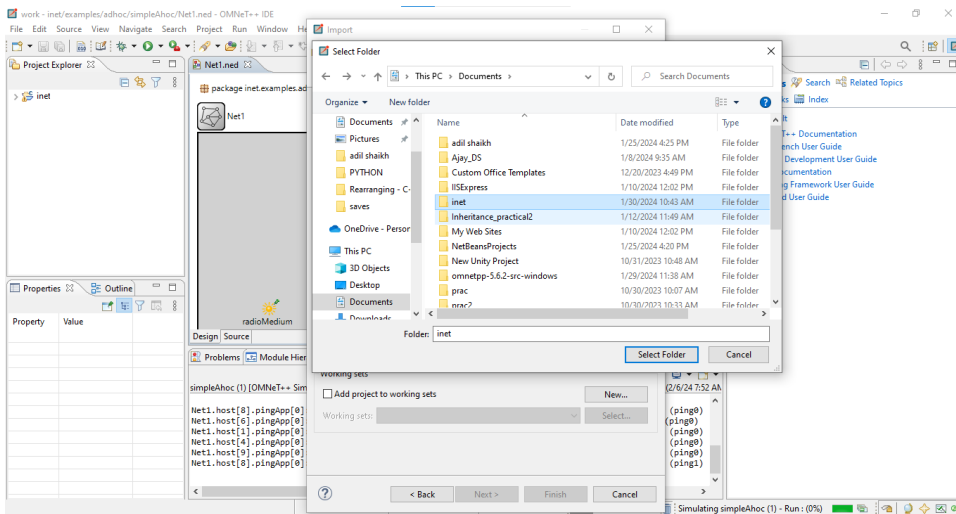
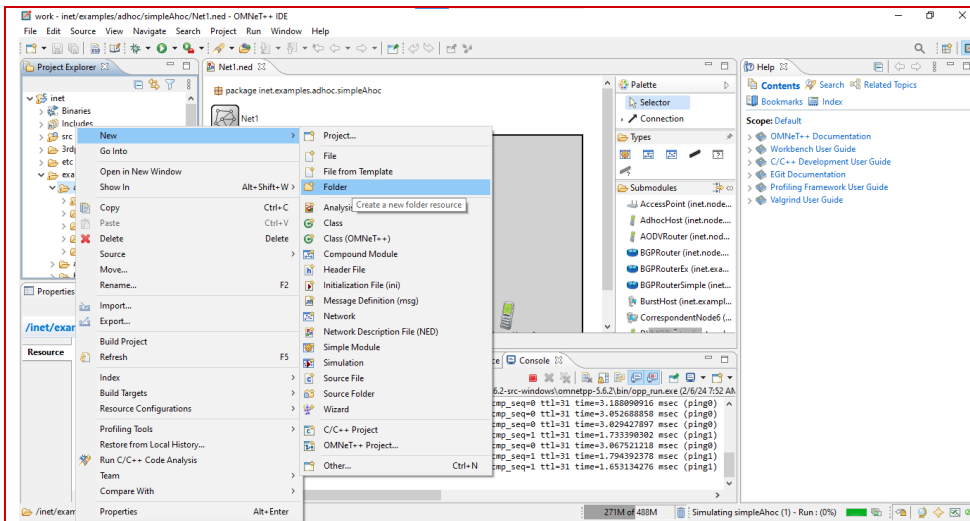
#### **B) DESCRIPTION:**

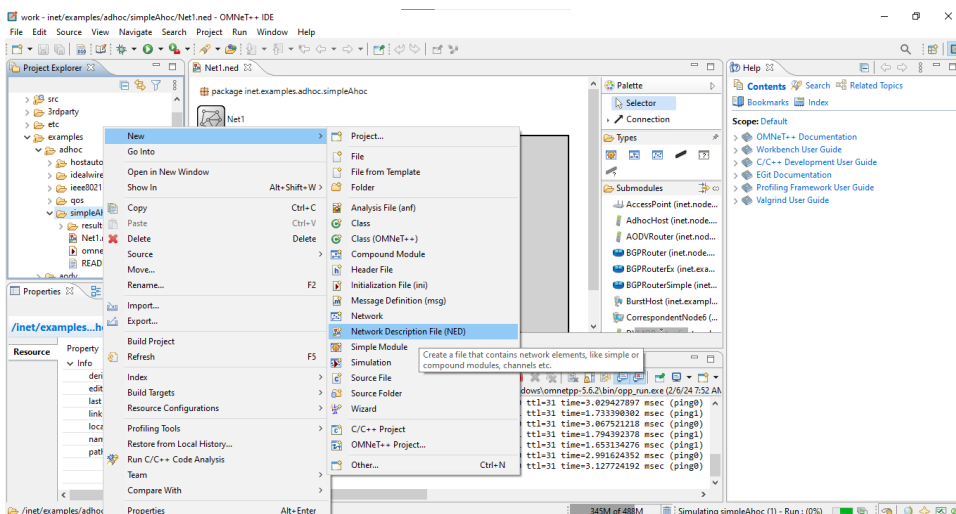
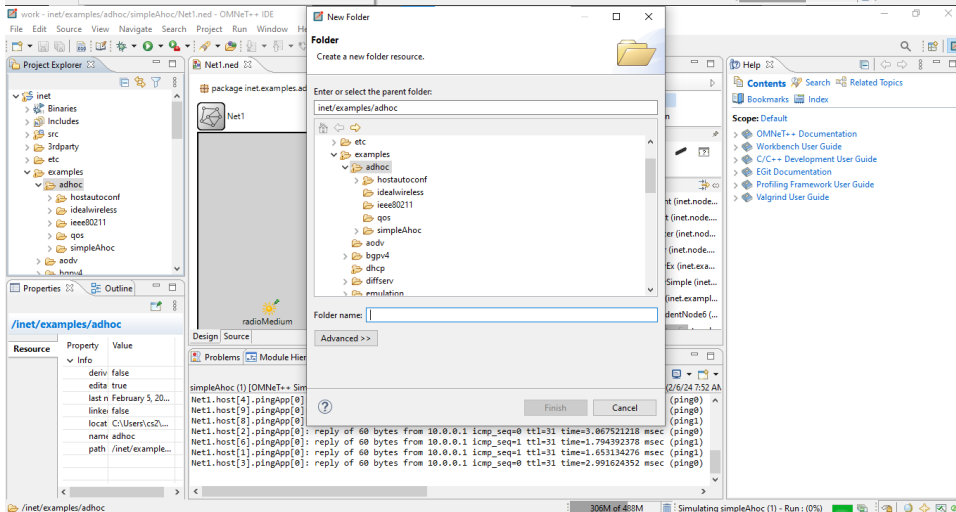
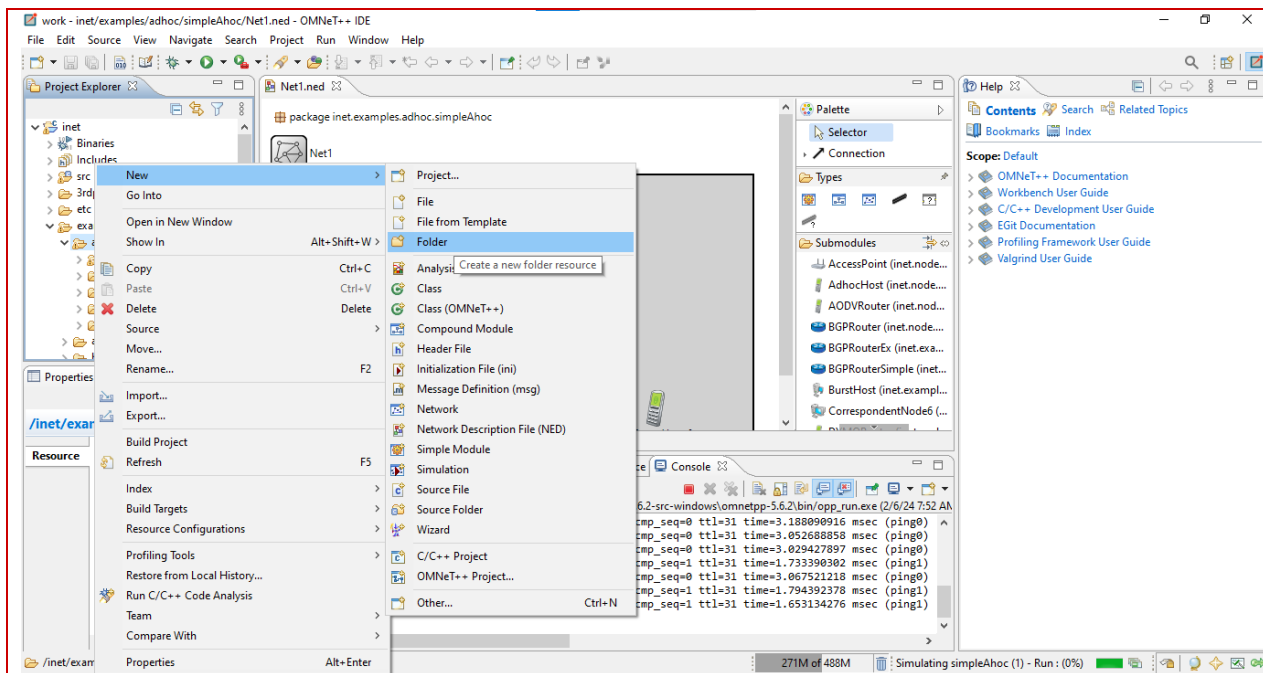
An ad hoc network is one that is spontaneously formed when devices connect and communicate with each other. The term ad hoc is a Latin word that literally means "for this," implying improvised or impromptu. Ad hoc networks are mostly wireless local area networks (WLANs). The devices communicate with 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.

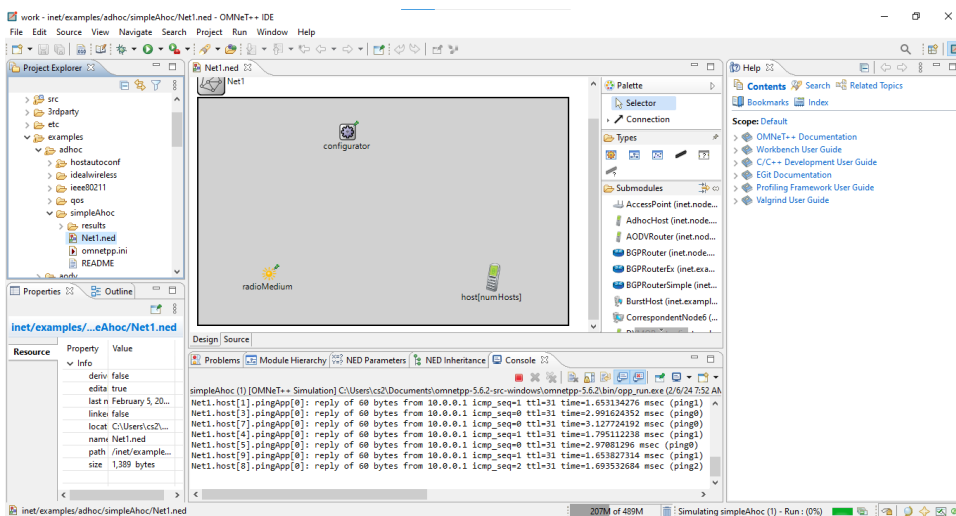
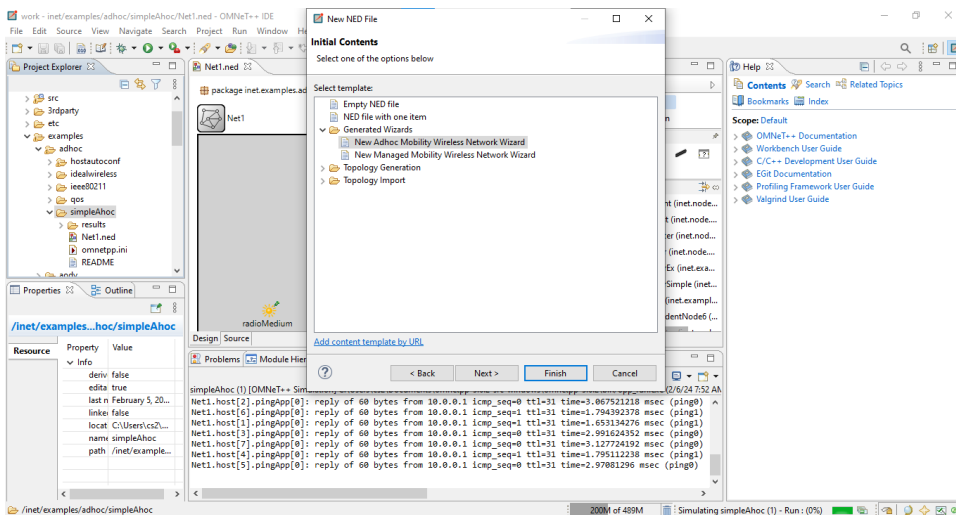
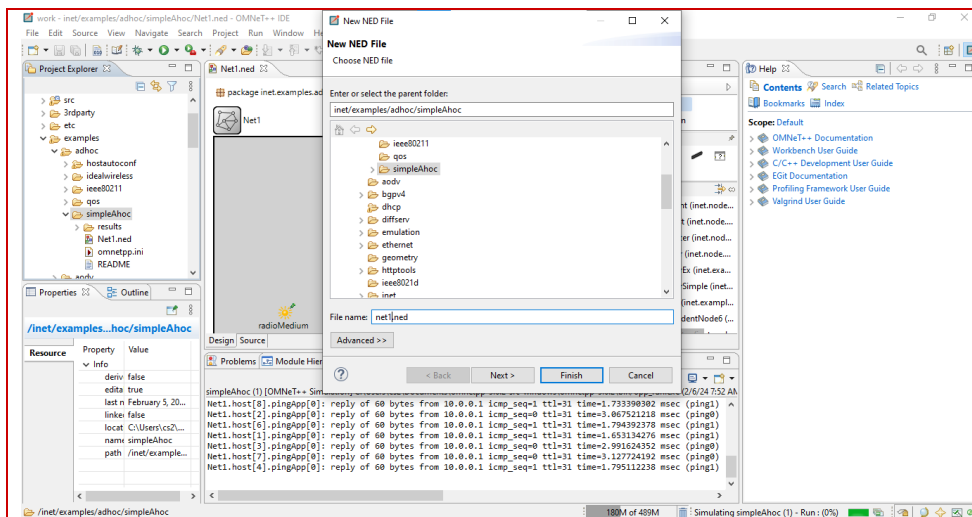
## C) NETWORK TOPOLOGY, CONFIGURATIONS AND OUTPUT:

### Configurations:



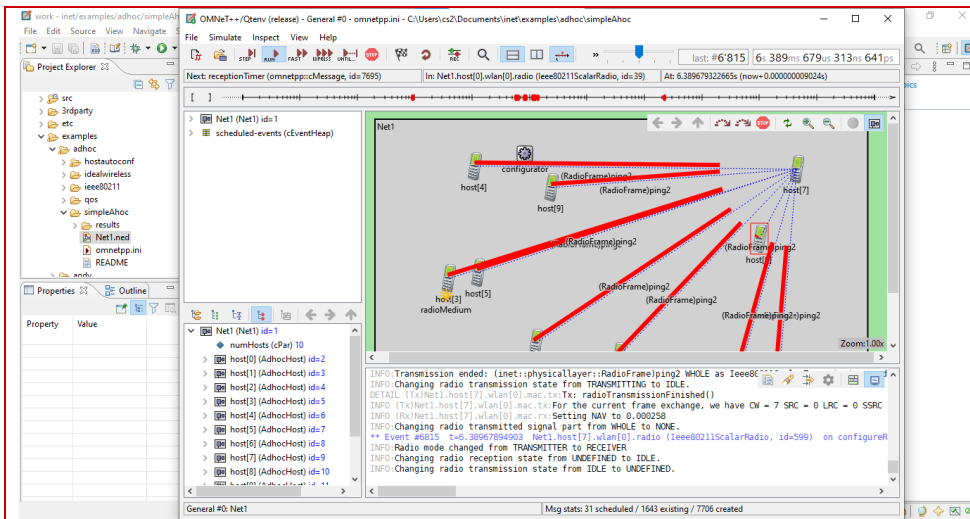






Output

Name of Instructor: Jescia D'cruz





# Wireless Sensor Networks & Mobile Communication

## Practical No.5

### DEPARTMENT OF COMPUTER SCIENCE

<b>Name:</b>	Ajay Kumar Uthaya Kumar	<b>Roll Number</b>	TCS2324002
<b>Paper Code:</b>	SIUSCS61	<b>Class</b>	TYBSc(Computer Science)
<b>Topic:</b>	Routing	<b>Batch</b>	I
<b>Date:</b>	29/1/24	<b>Practical No</b>	5

**A) AIM:** Understanding, Reading and Analyzing Routing Table of a network

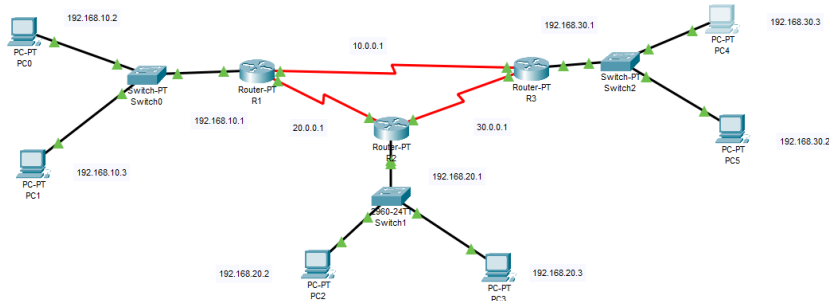
#### **B) DESCRIPTION:**

Network routing is the process of selecting a path across one or more networks. The principles of routing can apply to any type of network, from telephone networks to public transportation. In packet-switching networks, such as the Internet, routing selects the paths for Internet Protocol (IP) packets to travel from their origin to their destination. These Internet routing decisions are made by specialized pieces of network hardware called routers.

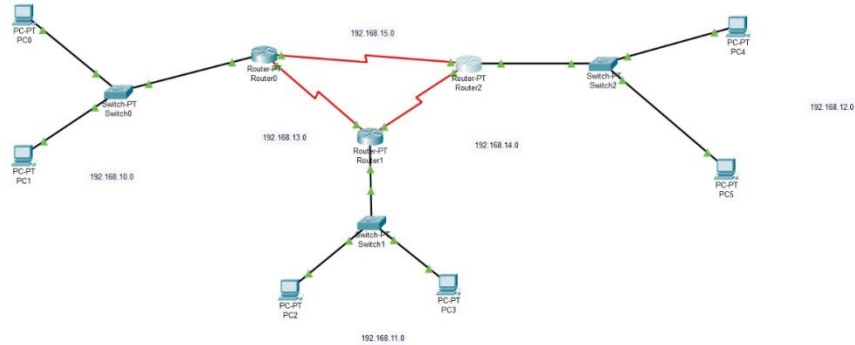
## C) NETWORK TOPOLOGY, CONFIGURATIONS AND OUTPUT:

Network topology(only for cisco packet tracer practical's):

### 1) Using RIP



### 2) Using OSPF



## Configurations:

### 1) Using RIP



PC0

Physical

Config

Desktop

Programming

Attributes

GLOBAL

Settings

Algorithm Settings

INTERFACE

FastEthernet0

Bluetooth

Global Settings

Display Name

PC0

Interfaces

FastEthernet0

Gateway/DNS IPv4

DHCP

Static

Default Gateway

192.168.10.0

DNS Server

Gateway/DNS IPv6

Automatic

Static

Default Gateway

DNS Server

PC0

Physical

Config

Desktop

Programming

Attributes

GLOBAL

Settings

Algorithm Settings

INTERFACE

FastEthernet0

Bluetooth

FastEthernet0

Port Status

☒ On

Bandwidth

☒ 100 Mbps

☐ 10 Mbps

☒ Auto

Duplex

☐ Half Duplex

☒ Full Duplex

☒ Auto

MAC Address

0060.5CBA.B643

IP Configuration

☐ DHCP

☒ Static

IPv4 Address

192.168.10.2

Subnet Mask

255.255.255.0

IPv6 Configuration

☐ Automatic

☒ Static

IPv6 Address

Link Local Address

FE80::260:5CFF:FEBA:B643

☐ Top

Name of Instructor: Jescia D’cruz

R1

Physical

Config

CLI

Attributes

GLOBAL

Settings

Algorithm Settings

ROUTING

Static

RIP

INTERFACE

FastEthernet0/0

FastEthernet1/0

Serial2/0

Serial3/0

FastEthernet4/0

FastEthernet5/0

FastEthernet0/0

Port Status

☒ On

Bandwidth

☒ 100 Mbps

☐ 10 Mbps

☒ Auto

Duplex

☐ Half Duplex

☒ Full Duplex

☒ Auto

MAC Address

0003.E4BA.C25B

IP Configuration

IPv4 Address

192.168.10.1

Subnet Mask

255.255.255.0

Tx Ring Limit

10

Equivalent IOS Commands

Router(config-router)#network 192.168.30.0

Router(config-router)#network 10.0.0.0

Router(config-router)#network 192.168.30.0

Router(config-router)#network 30.0.0.0

Router(config-router)#network 192.168.20.0

Router(config-router)#

Router(config-router)#

Router(config-router)#end

Router#configure terminal

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#interface FastEthernet0/0

Router(config-if)#

%SYS-5-CONFIG\_I: Configured from console by console

☐ Top

R1

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 20.0.0.1

Subnet Mask 255.0.0.0

Tx Ring Limit 10

Equivalent IOS Commands

```

Router(config-router)#network 30.0.0.0
Router(config-router)#network 192.168.20.0
Router(config-router)#
Router(config-router)#
Router(config-router)#end
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#interface FastEthernet0/0
Router(config-if)#
%SYS-5-CONFIG_I: Configured from console by console

Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#
  
```

R1

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

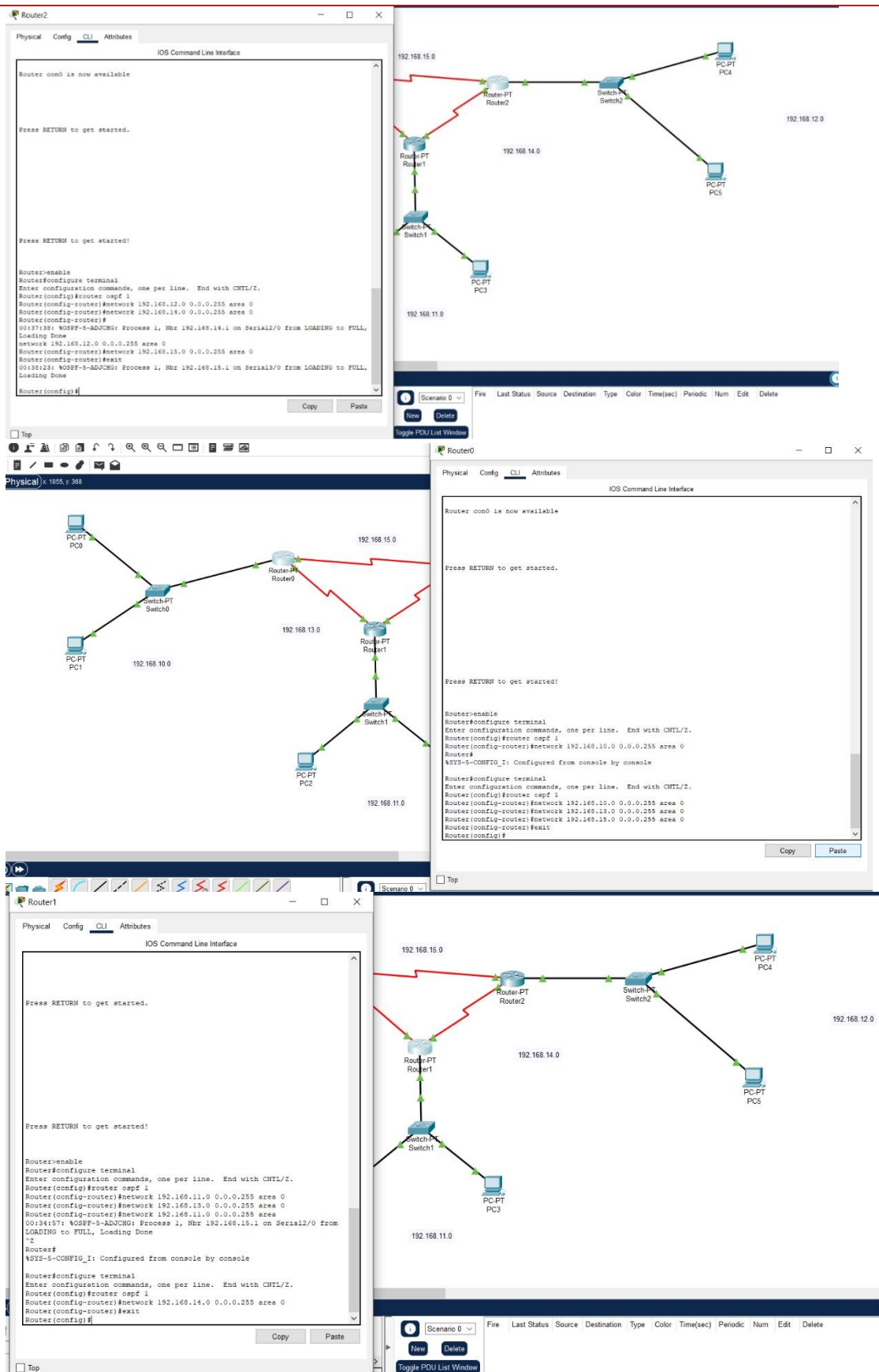
Remove

Equivalent IOS Commands

```

Router(config-router)#
Router(config-router)#end
  
```

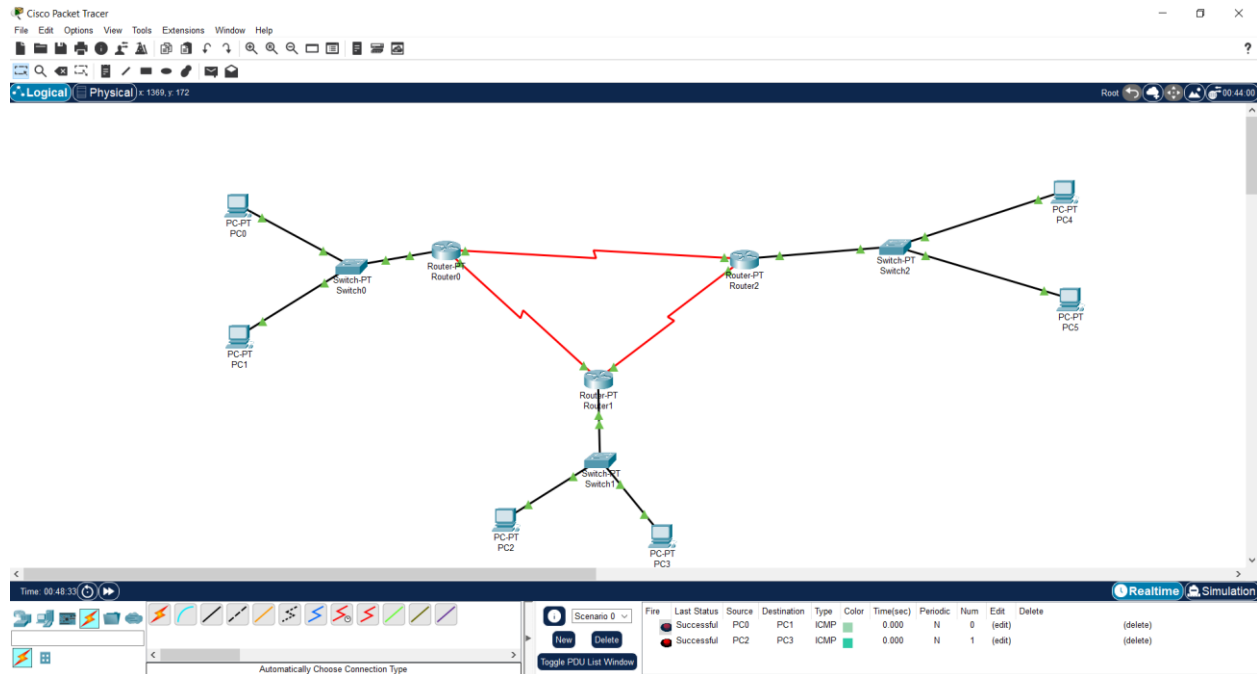
## 2) Using OSPF



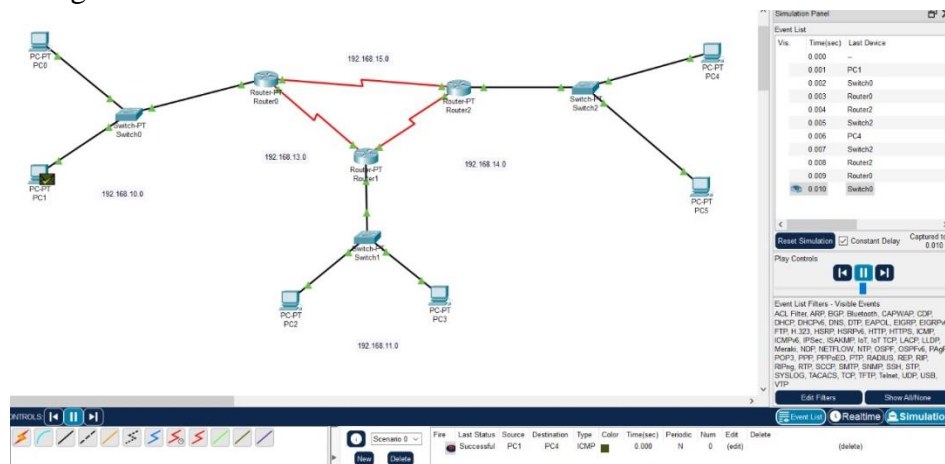
Output

Name of Instructor: Jescia D'cruz

## 1) Using RIP



## 2) Using OSPF





# Wireless Sensor Networks & Mobile Communication

## Practical No.6

### DEPARTMENT OF COMPUTER SCIENCE

<b>Name:</b>	Ajay Kumar Uthaya Kumar	<b>Roll Number</b>	TCS2324002
<b>Paper Code:</b>	SIUSCS61	<b>Class</b>	TYBSc(Computer Science)
<b>Topic:</b>	Wireless sensor network	<b>Batch</b>	I
<b>Date:</b>	05/1/24	<b>Practical No</b>	6

**A) AIM:** Implement a Wireless sensor network simulation

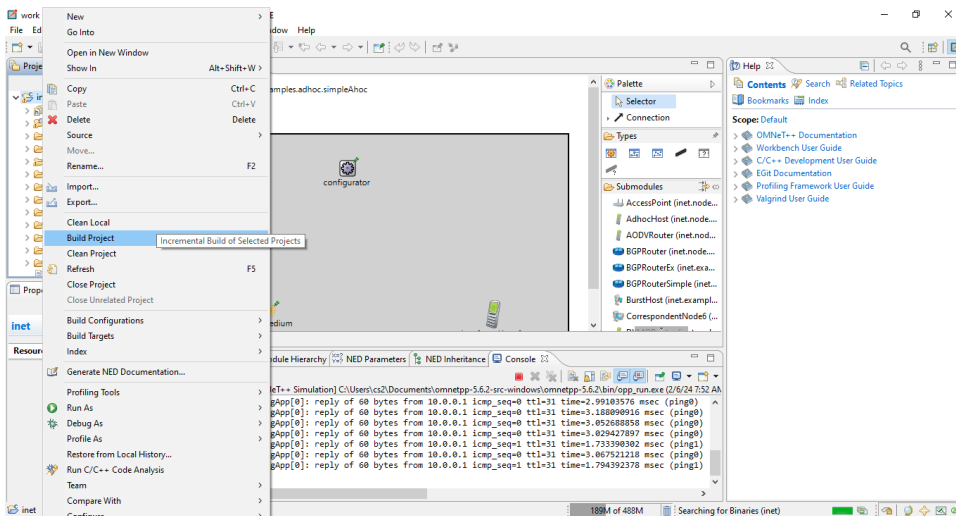
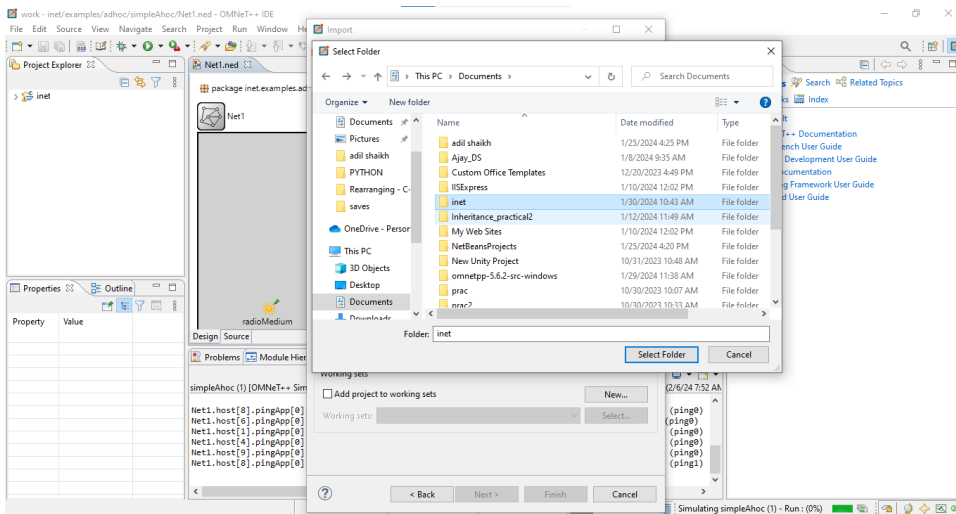
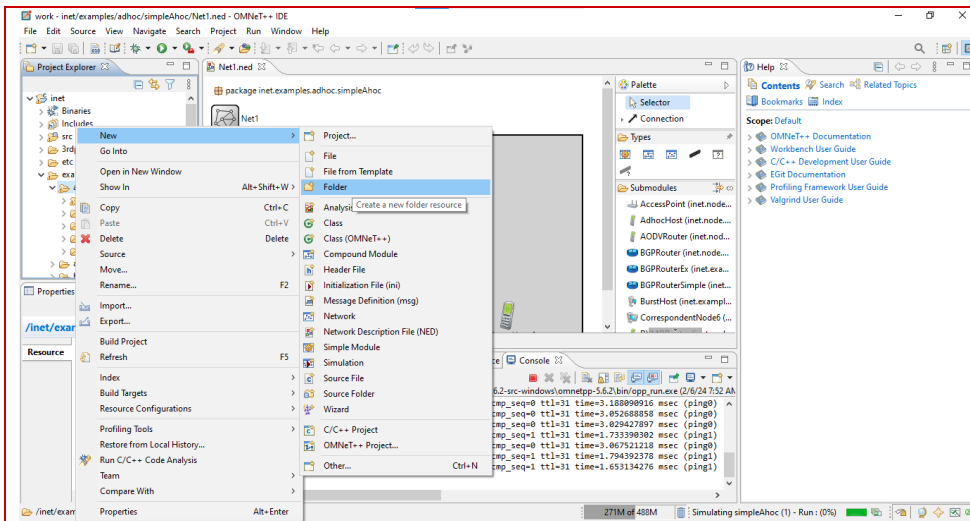
#### **B) DESCRIPTION:**

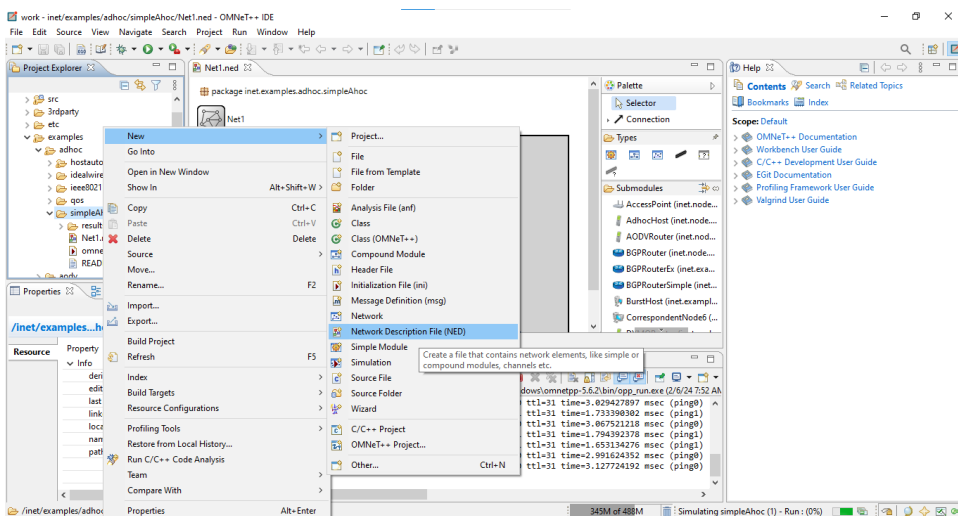
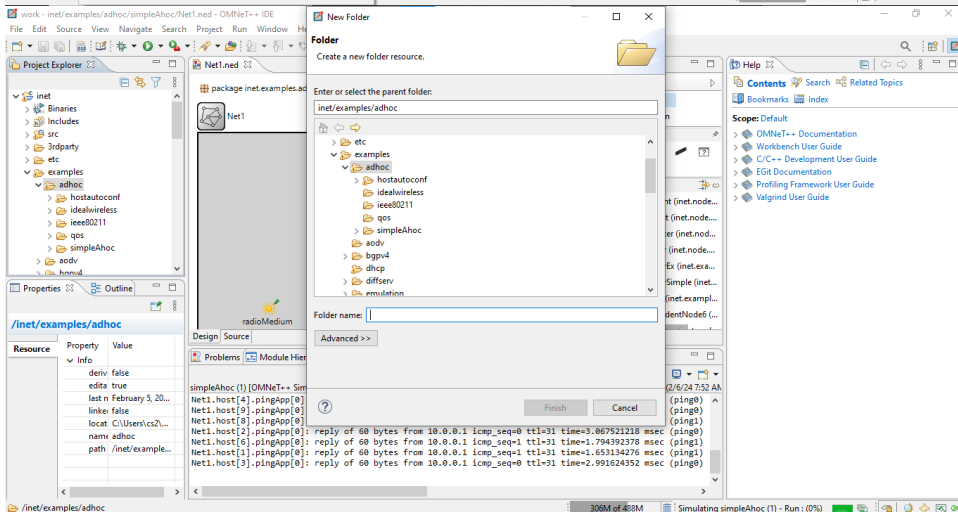
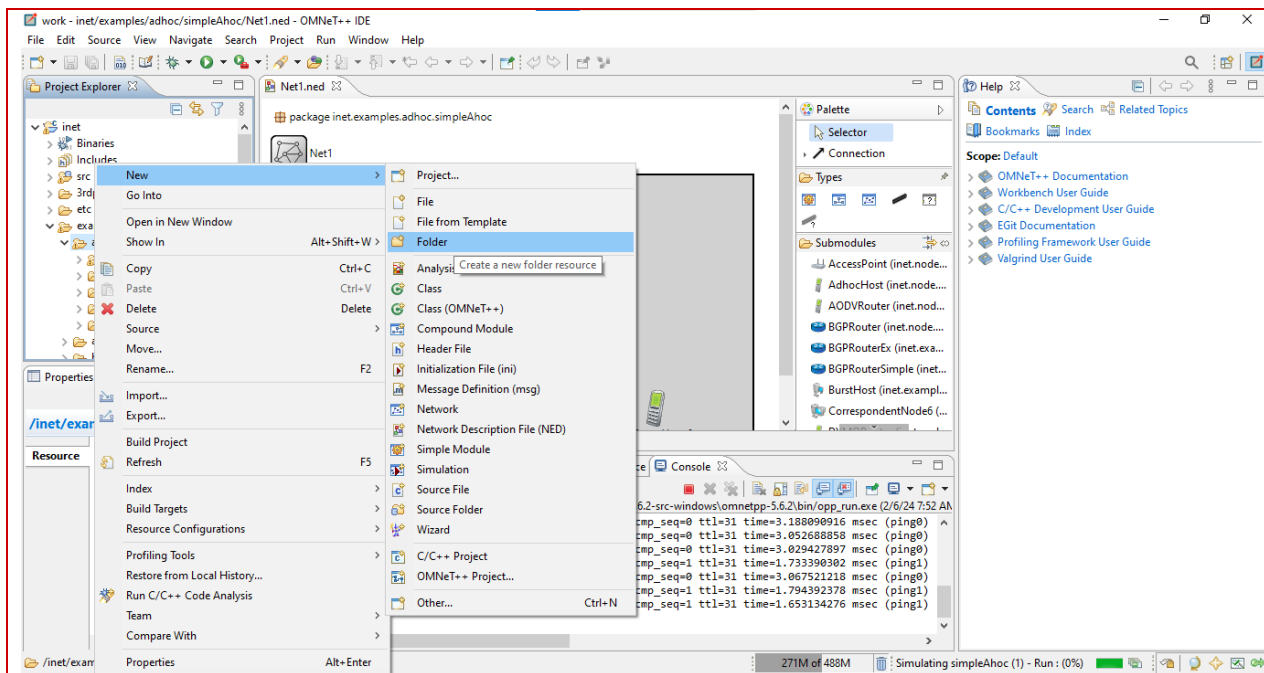
Wireless Sensor Network (WSN) is an infrastructure-less wireless network that is deployed in a large number of wireless sensors in an ad-hoc manner that is used to monitor the system, physical or environmental conditions. Sensor nodes are used in WSN with the onboard processor that manages and monitors 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.

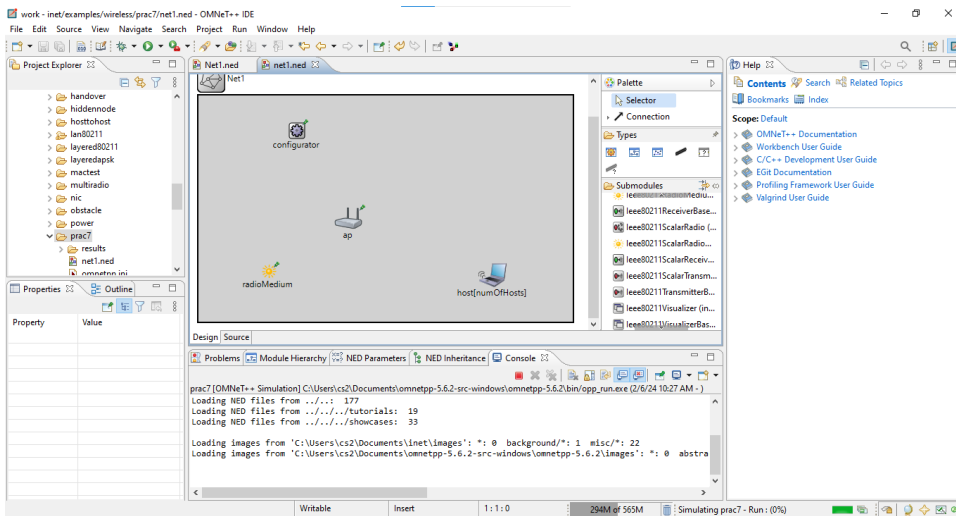
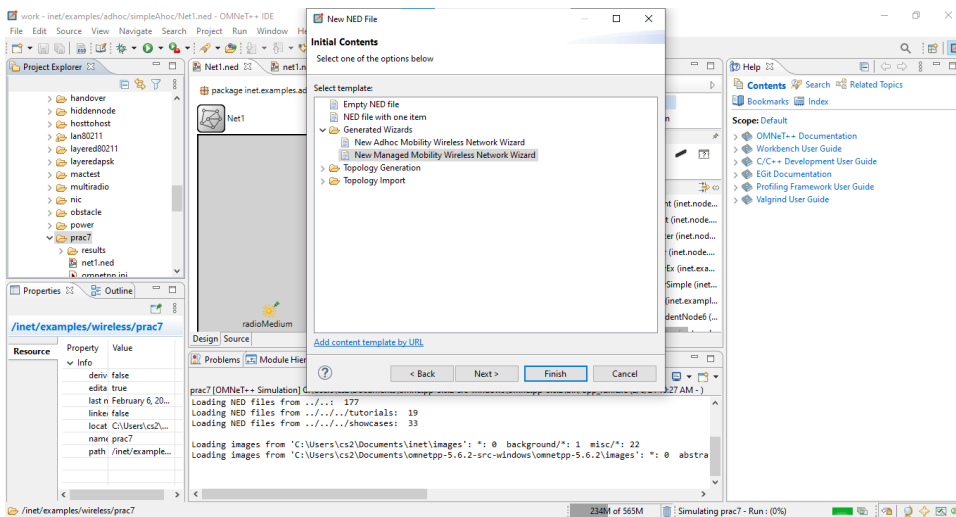
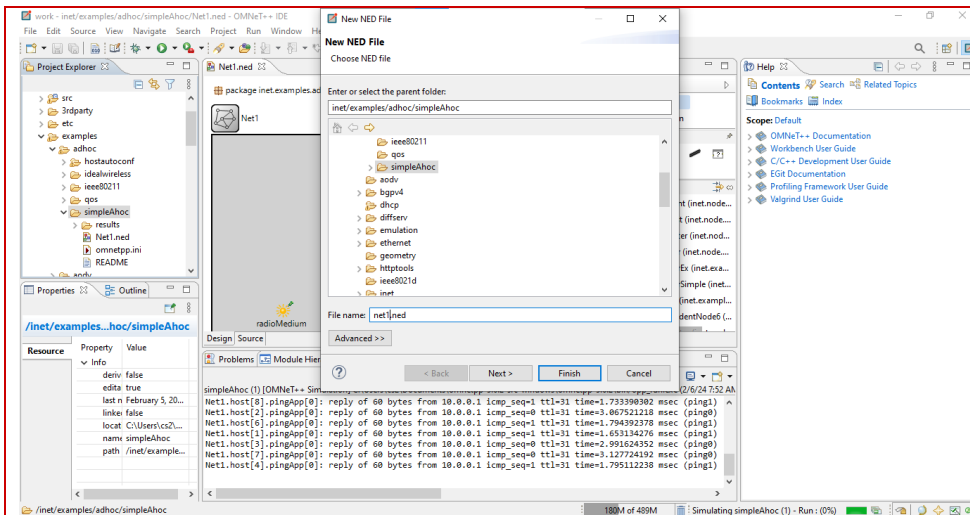
### Configurations:











Code:

//

// This program is free software: you can redistribute it and/or modify

```
// it under the terms of the GNU Lesser General Public License as published by
// the Free Software Foundation, either version 3 of the License, or
// (at your option) any later version.
//
// This program is distributed in the hope that it will be useful,
// but WITHOUT ANY WARRANTY; without even the implied warranty of
// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
// GNU Lesser General Public License for more details.
//
// You should have received a copy of the GNU Lesser General Public License
// along with this program. If not, see http://www.gnu.org/licenses/.
//

package inet.examples.wireless.prac7;

// numOfHosts: 5

import inet.examples.adhoc.hostautoconf.Host;
import inet.networklayer.configurator.ipv4.IPv4NetworkConfigurator;
import inet.node.inet.AdhocHost;
import inet.node.inet.WirelessHost;
import inet.node.wireless.AccessPoint;
import inet.physicallayer.ieee80211.packetlevel.ieee80211ScalarRadioMedium;

network Net1
{
    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=100,250");
```

```
}
```

```
}
```

**Output**

Word | Microsoft Teams

OMNet++ - QEnv (release) - General #0 - omnetpp.ini - C:\Users\c2\Documents\inet\examples\wireless\prac7

File Simulate Inspect View Help

Next: NAV (omnetpp::cMessage, id=72) [In: Net1.host[3].wlan[0].mac.rx (Rx, id=273)] [At: 0.544276545027s (now=0.00000010163s)]

Net1 (Net1) id=1

- scheduled-events (EventHeap)

Net1 (Net1) 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 (IP4NetworkConfigurator) id=8
- radioMedium (ieee80211ScalarRadio) id=9
- canvas (cCanvas) 1 [top level figure](#)

DETAIL (Tx)Net1.host[0].wlan[0].mac.tx.Tx: radioTransmissionFinished()

INFO: Changing radio transmitted signal part from WHOLE to NONE.

\*\* Event #1052 t=0.544276541359 Net1.host[0].wlan[0].radio (1ieee80211ScalarRadio, id=33) on configurator

INFO: Radio mode changed from TRANSMITTER to RECEIVER

INFO: Changing radio reception state from UNDEFINED to IDLE.

INFO: Changing radio transmission state from IDLE to UNDEFINED.

INFO: (Contention)Net1.host[0].wlan[0].mac.dcf.channelAccess.contention.backoffSlots = 5 slotTime = 0.0000

INFO: (Contention)Net1.host[0].wlan[0].mac.dcf.channelAccess.contention.waitInterval = 0.00015

\*\* Event #1053 t=0.544276516907 Net1.host[1].wlan[0].mac.rx (Rx, id=177) on selfmsg NAV (omnetpp::cMessage, id=72)

INFO: The radio channel has become free according to the NAV

General #0: Net1

Msg stats: 21 scheduled / 324 existing / 1306 created



# Wireless Sensor Networks & Mobile Communication

## Practical No.7

### DEPARTMENT OF COMPUTER SCIENCE

<b>Name:</b>	Ajay Kumar Uthaya Kumar	<b>Roll Number</b>	TCS2324002
<b>Paper Code:</b>	SIUSCS61	<b>Class</b>	TYBSc(Computer Science)
<b>Topic:</b>	MAC Protocol	<b>Batch</b>	I
<b>Date:</b>	06/2/24	<b>Practical No</b>	7

**A) AIM:** Create a MAC Protocol simulation implementation for wireless sensor network

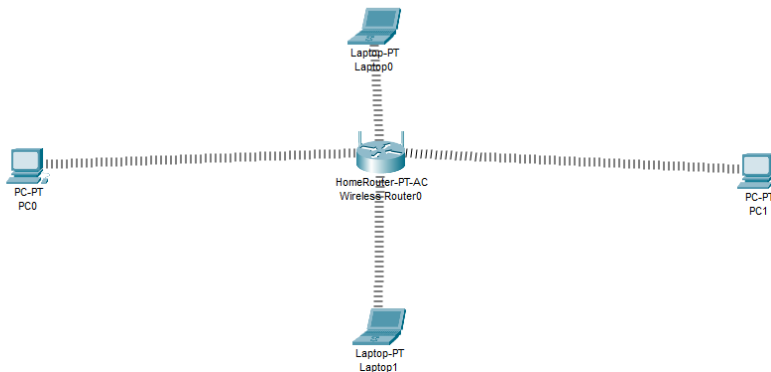
#### **B) DESCRIPTION:**

Media access control (MAC) protocols enforce a methodology to allow multiple devices access to a shared media network. Before LANs, communication between computing devices had been point-to-point. That is, two devices were connected by a dedicated channel. LANs are shared media networks, in which all devices attached to the network receive each transmission and must recognize which frames they should accept. Media sharing reduced the cost of the network, but also meant that MAG protocols were needed to coordinate use of the medium. There are two approaches to media access control in LANs: contention and token-passing.

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

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

Network topology(only for cisco packet tracer practical's):



### Configurations:

protocol.pkt

Laptop0

Physical Config Desktop Programming Attributes

**GLOBAL**

Settings

Algorithm Settings

**INTERFACE**

Wireless0

Bluetooth

**Wireless0**

Port Status ☒ On

Bandwidth 300 Mbps

MAC Address 0001.425D.3E05

SSID Default

Authentication

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

WEP Key

PSK Pass Phrase

User ID

Password

Method: MD5

User Name

Password

Encryption Type Disabled

IP Configuration

☒ DHCP ☐ Static

IPv4 Address 192.168.0.100

Subnet Mask 255.255.255.0

IPv6 Configuration

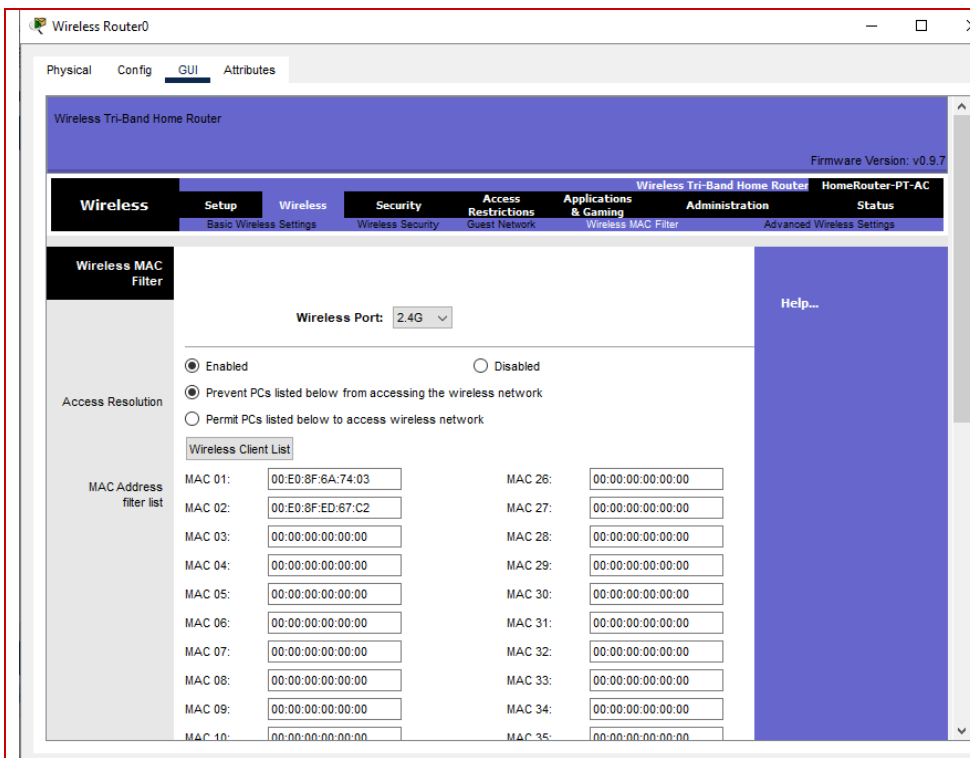
☒ Automatic ☐ Static

IPv6 Address

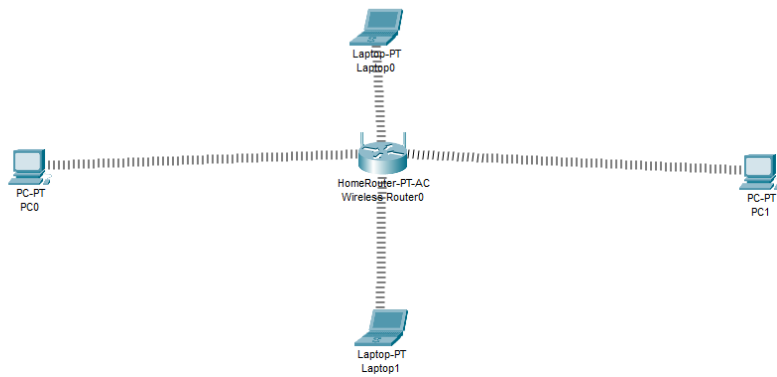
Link Local Address: FE80::201:42FF:FE5D:3E05

☐ Top





## Output





# Wireless Sensor Networks & Mobile Communication

## Practical No.8

### DEPARTMENT OF COMPUTER SCIENCE

<b>Name:</b>	Ajay Kumar Uthaya Kumar	<b>Roll Number</b>	TCS2324002
<b>Paper Code:</b>	SIUSCS61	<b>Class</b>	TYBSc(Computer Science)
<b>Topic:</b>	Mobile Adhoc	<b>Batch</b>	I
<b>Date:</b>	06/2/24	<b>Practical No</b>	8

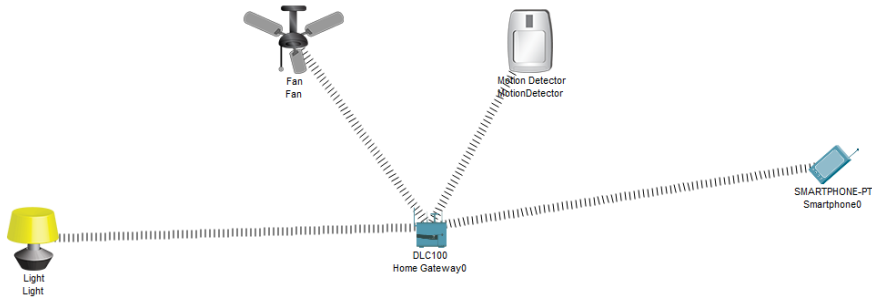
**A) AIM:** Stimulate a Mobile Adhoc network with directional antennas

#### **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(only for cisco packet tracer practical's):



### Configurations:

MANET.pkt

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 000B.BE17.C18A

SSID HomeGateway

**Authentication**

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

WEK Key

PSK Pass Phrase

User ID

Password

Method: MD5

User Name

Password

**Encryption Type** Disabled

**IP Configuration**

☒ DHCP ☐ Static

IPv4 Address 192.168.25.102

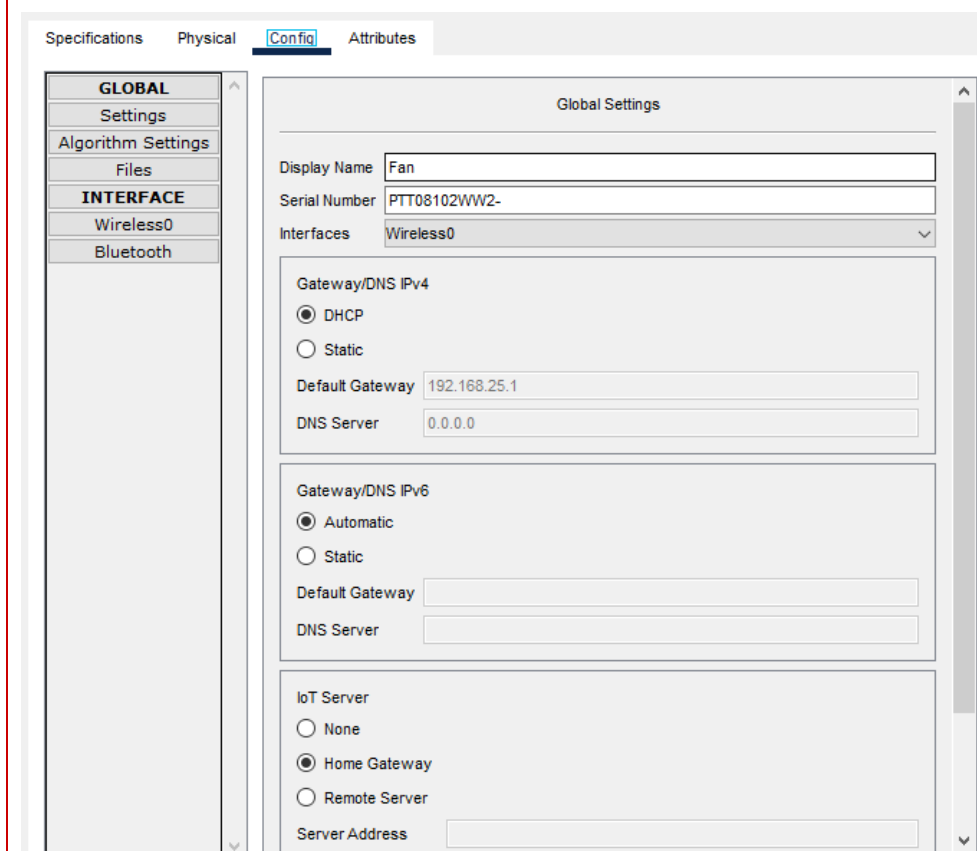
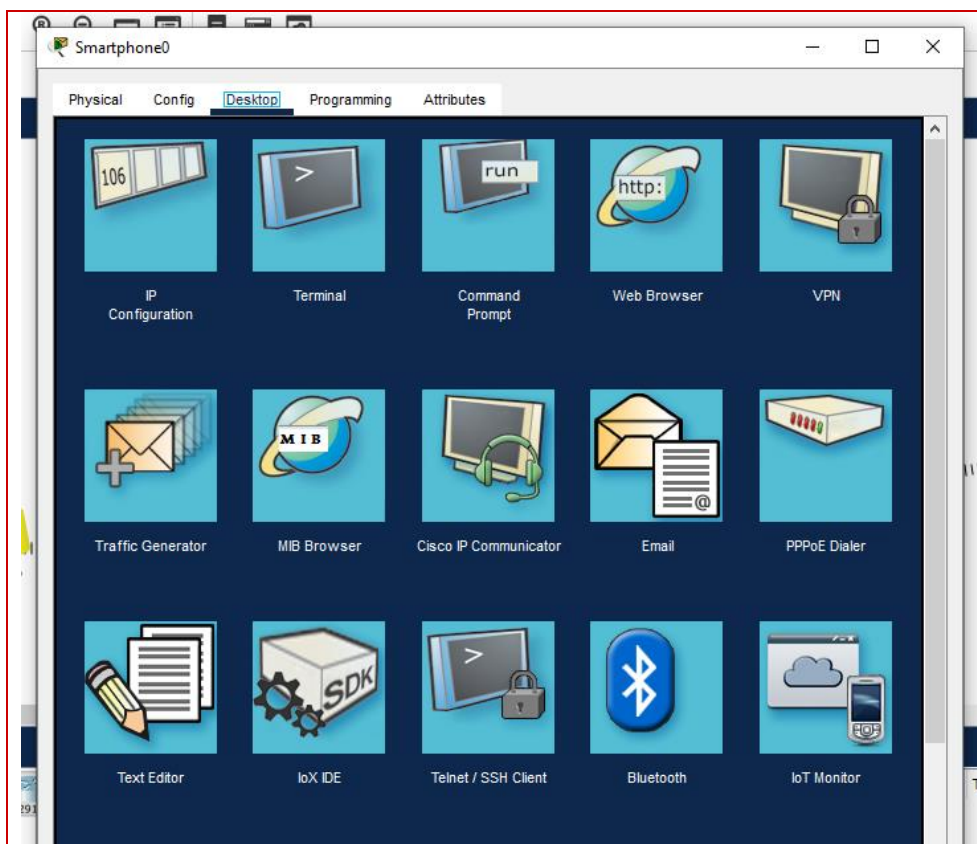
Subnet Mask 255.255.255.0

**IPv6 Configuration**

☒ Automatic ☐ Static

IPv6 Address

Link Local Address: FE80::20B:BEFF:FE17:C18A



IoT Monitor

X

IoT Server - Devices

Home | Conditions | Editor | Log Out

▶ ● Fan (PTT08102WW2-)

Ceiling Fan

▶ ● Light (PTT0810411Z-)

Light

▶ ● MotionDetector (PTT08106SN0-)

Motion Detector

## Conditions

Edit Rule

X

Name

Enabled ☒

If:

Match 

All

MotionDetector

On

is

true

-

+ Condition

+ Group

Then set:

Light

Status

to

On

Fan

Status

to

High

+ Action

-

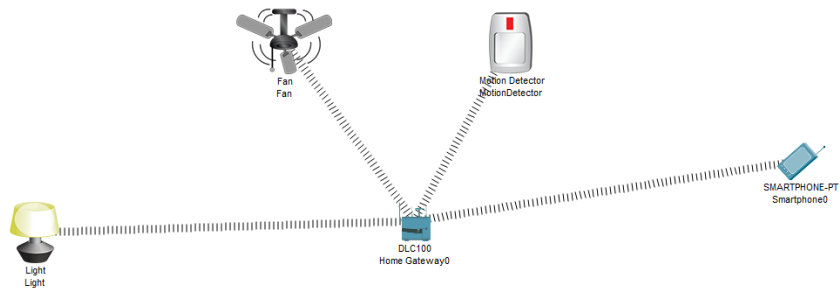
-

OK

Cancel

Actions	Enabled	Name	Condition	Actions
<div> <div>Edit Rule</div> <div> <div>Name</div> <div>app_off</div> </div> <div> <div>Enabled</div> <div><input checked="" type="checkbox"/></div> </div> <div> <div>If:</div> <div> <div>Match</div> <div>All</div> </div> <div> <div>MotionDetector</div> <div>On</div> <div>is</div> <div>false</div> </div> <div> <div>+ Condition</div> <div>+ Group</div> </div> </div> <div> <div>Then set:</div> <div> <div>Light</div> <div>Status</div> <div>to</div> <div>Off</div> </div> <div> <div>Fan</div> <div>Status</div> <div>to</div> <div>Off</div> </div> <div> <div>+ Action</div> <div>-</div> <div>-</div> </div> </div> <div> <div>OK</div> <div>Cancel</div> </div> </div>				

Output





# Wireless Sensor Networks & Mobile Communication

## Practical No.9

### DEPARTMENT OF COMPUTER SCIENCE

<b>Name:</b>	Ajay Kumar Uthaya Kumar	<b>Roll Number</b>	TCS2324002
<b>Paper Code:</b>	SIUSCS61	<b>Class</b>	TYBSc(Computer Science)
<b>Topic:</b>	Mobile Network	<b>Batch</b>	I
<b>Date:</b>	06/2/24	<b>Practical No</b>	9

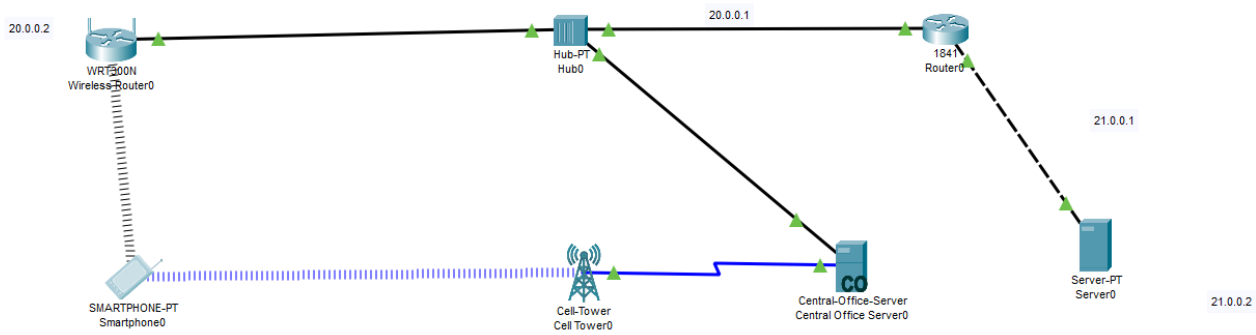
**A) AIM:** Create a mobile network using Cell Tower, Central office server, Web Browser and Web Server, Stimulate connection between them.

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

A mobile network (also wireless network) route's communications in the form of radio waves to and from users. It is composed of base stations that each cover a delimited area or "cell." When joined together these cells provide radio coverage over a wide geographic area. This enables a large number of portable transceivers (e.g., mobile phones, pagers, etc.) to communicate with each other and with fixed transceivers and telephones anywhere in the network, even if some of the transceivers are moving through more than one cell during transmission. Mobile networks are rapidly becoming the universal service delivery vehicle for all applications. The key question is whether they can manage to keep up with the underlying bandwidth demands. The rising demand for mobile broadband services has accelerated the move to LTE and LTE-Advanced. This latest mobile technology further increases not only bandwidth but also quality requirements of the backhaul network.

## C) NETWORK TOPOLOGY, CONFIGURATIONS AND OUTPUT:

Network topology(only for cisco packet tracer practical's):



### Configurations:

Wireless router configuration

The screenshot shows the configuration window for the WRT300N Wireless Router. The 'Config' tab is selected, and the 'Internet Settings' section is expanded. The configuration is as follows:

Category	Setting	Value
IP Configuration	<input type="radio"/> DHCP	
	<input checked="" type="radio"/> Static	
UserName	UserName	
	Password	
IPv4 Address	IPv4 Address	20.0.0.2
Subnet Mask	Subnet Mask	255.0.0.0
Default Gateway	Default Gateway	20.0.0.1
DNS Server	DNS Server	

1841 router configuration



Physical **Config** CLI Attributes

<b>GLOBAL</b>	<b>FastEthernet0/0</b>	
Settings		
Algorithm Settings		
<b>ROUTING</b>		
Static		
RIP		
<b>SWITCHING</b>		
VLAN Database		
<b>INTERFACE</b>		
FastEthernet0/0		
FastEthernet0/1		

Port Status	<input checked="" type="checkbox"/> On
Bandwidth	<input type="radio"/> 100 Mbps <input type="radio"/> 10 Mbps <input checked="" type="checkbox"/> Auto
Duplex	<input type="radio"/> Half Duplex <input type="radio"/> Full Duplex <input checked="" type="checkbox"/> Auto
MAC Address	000A.F329.7301
IP Configuration	
IPv4 Address	20.0.0.1
Subnet Mask	255.0.0.0
Tx Ring Limit	
	10

Equivalent IOS Commands

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1, changed state to up

```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#interface FastEthernet0/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet0/0
Router(config-if)#
```

Server configuration

Server0

Physical

Config

Services

Desktop

Programming

Attributes

GLOBAL

Settings

Algorithm Settings

INTERFACE

FastEthernet0

Global Settings

Display Name

Server0

Gateway/DNS IPv4

DHCP

Static

Default Gateway

21.0.0.1

DNS Server

Gateway/DNS IPv6

Automatic

Static

Default Gateway

DNS Server

Name of Instructor: Jescia D’cruz

Physical
Config
Services
Desktop
Programming
Attributes

GLOBAL

Settings

Algorithm Settings

INTERFACE

FastEthernet0

FastEthernet0

Port Status
☒ On

Bandwidth
☒ 100 Mbps
☐ 10 Mbps
☒ Auto

Duplex
☐ Half Duplex
☒ Full Duplex
☒ Auto

MAC Address
0004.9A4A.CAA9

IP Configuration
☐ DHCP
☒ Static

IPv4 Address
21.0.0.2

Subnet Mask
255.0.0.0

IPv6 Configuration
☐ Automatic
☒ Static

IPv6 Address

Link Local Address
FE80::204:9AFF:FE4A:CAA9

# Output

