Concordia Institute for Information System Engineering (CIISE)
Concordia University

INSE 6170 Network Security Architecture and Management

Project 1: Scan the IoT devices

Submitted to: **Professor Carol Fung**
Submitted by:

Aniket Agarwal	40266485
Kalyani Batle	40243967

Objectives

- Target operating system – Windows 10
- Scan our Wi-Fi network for active IoT devices
- Provide IP and MAC address about each discovered device
- Enable users to input device information and capture packets
- Save captured packets in a pcap file format
- Save pcap filename with the device info in database for 10 IoT devices.

IoT Devices

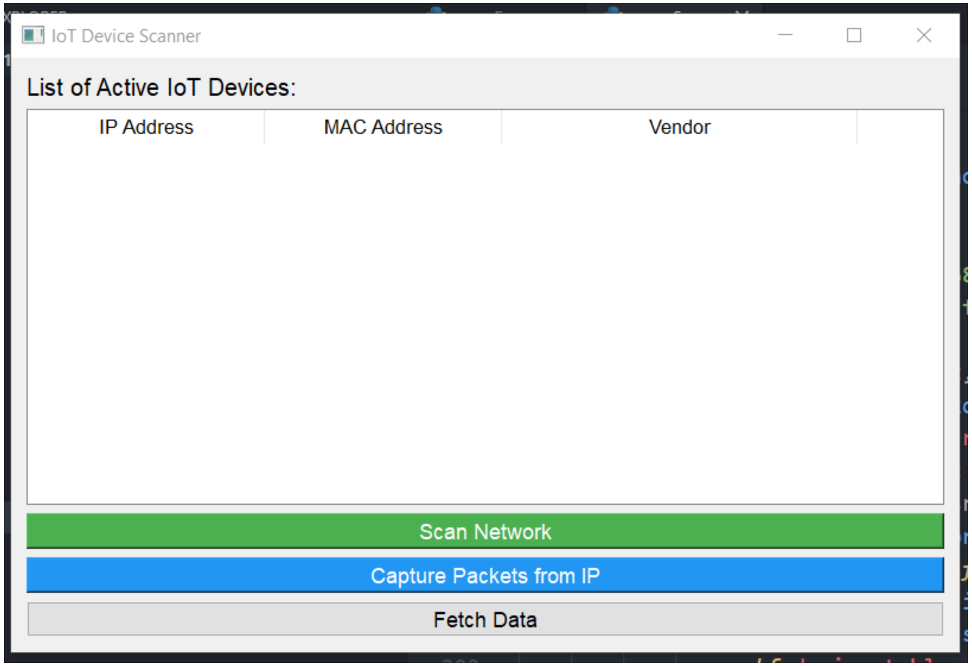
- IoT Devices are everyday objects embedded with sensors, software and connectivity to exchange data over the internet.
- It has wide range.
- Each IoT device has a public IP over the internet and a respective MAC address.

Tools and Libraries Used to Scan Wi-Fi network for IoT Devices

- *Python3*
- *PyQt5* (for GUI interface)
- *Scapy* to send ARP requests to discover IoT devices and capture packets to and from a specified device in the network.
- *Requests* to send API request to mavendors.com to get the vendor details of respective MAC addresses of the devices discovered.
- *Sqlite3* to store filename and respective IP of device in the database.

PyQt5

- Python library to create cross-platform desktop applications with GUI.
- Offers wide range of widgets, layouts and tools for building sophisticated desktop apps.



IoT Device Scanner

List of Active IoT Devices:

IP Address	MAC Address	Vendor

Scapy

- Powerful Python Library for packet manipulation and network analysis.
- Allow packet capturing, send and receive packets on a network.
- Makes network discovery easier with Python.

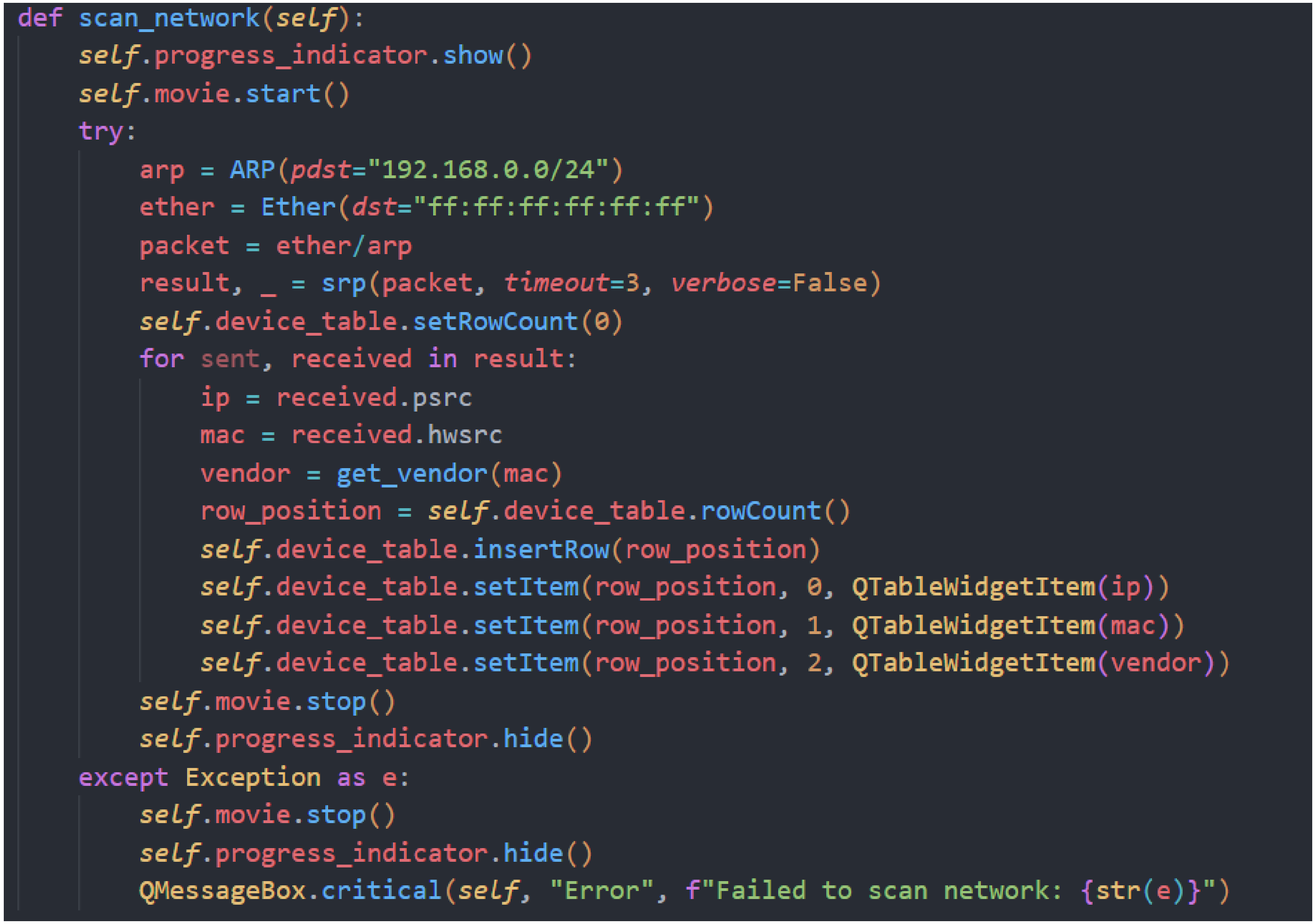
Sqlite3

- In-built library in python to create database instances.
- No need to maintain a separate database server.
- Lightweight solution for small to medium sized applications.

Algorithm to Scan Wi-Fi Network

- Scapy generates an ARP object having destination address as IP range of local Wi-Fi network.
- Generate Ethernet object having destination MAC address as broadcast address to send ARP request to all IP in network.
- Packet is created and send using srp function of scapy.
- Packets received are noted and device info is stored and displayed using PyQt5 GUI interface.

Source code for Scanning network

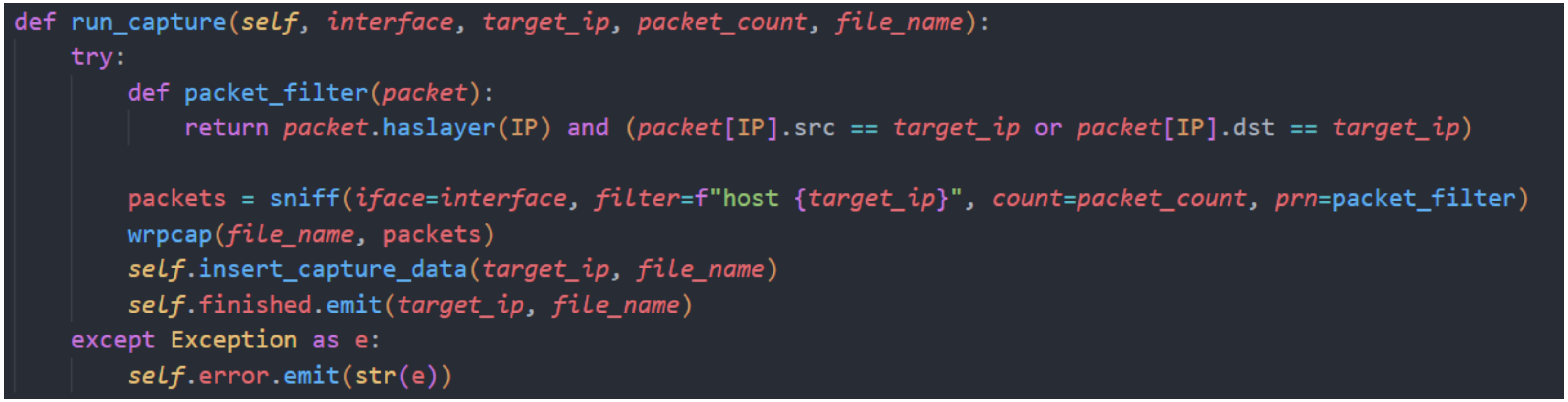


```python
def scan_network(self):
    self.progress_indicator.show()
    self.movie.start()
    try:
        arp = ARP(pdst="192.168.0.0/24")
        ether = Ether(dst="ff:ff:ff:ff:ff:ff")
        packet = ether/arp
        result, _ = srp(packet, timeout=3, verbose=False)
        self.device_table.setRowCount(0)
        for sent, received in result:
            ip = received.psrc
            mac = received.hwsrc
            vendor = get_vendor(mac)
            row_position = self.device_table.rowCount()
            self.device_table.insertRow(row_position)
            self.device_table.setItem(row_position, 0, QTableWidgetItem(ip))
            self.device_table.setItem(row_position, 1, QTableWidgetItem(mac))
            self.device_table.setItem(row_position, 2, QTableWidgetItem(vendor))
        self.movie.stop()
        self.progress_indicator.hide()
    except Exception as e:
        self.movie.stop()
        self.progress_indicator.hide()
        QMessageBox.critical(self, "Error", f"Failed to scan network: {str(e)}")
```

Algorithm to Capture Packets through a Device

- Scapy uses sniff() to capture the packets through a specific device.
- First, packets are filtered based on the device IP as src or dst fields and presence of IP layer.
- Next, packets are captured based on the packet count given by the user.
- Lastly, the packets are stored in a pcap file using wrpcap() function of scapy and stored in the database using sqlite3 INSERT statement.

Source code for Packet Capture

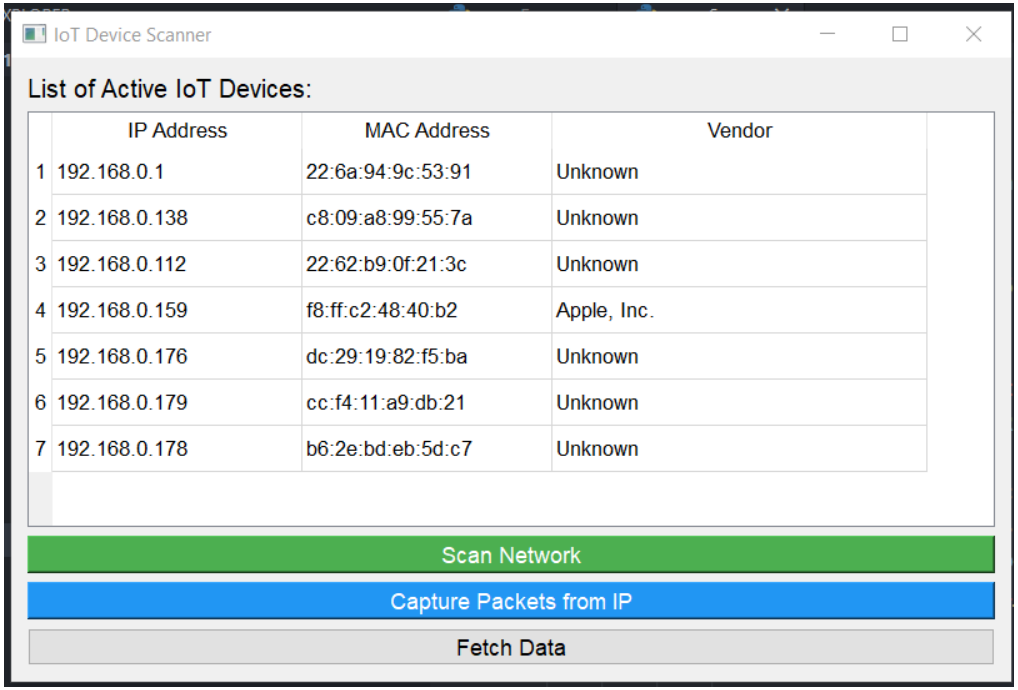


```
def run_capture(self, interface, target_ip, packet_count, file_name):
    try:
        def packet_filter(packet):
            return packet.haslayer(IP) and (packet[IP].src == target_ip or packet[IP].dst == target_ip)

        packets = sniff(iface=interface, filter=f"host {target_ip}", count=packet_count, prn=packet_filter)
        wrpcap(file_name, packets)
        self.insert_capture_data(target_ip, file_name)
        self.finished.emit(target_ip, file_name)
    except Exception as e:
        self.error.emit(str(e))
```

Displaying Device Information

- IP Address
- MAC Address
- Vendor Details

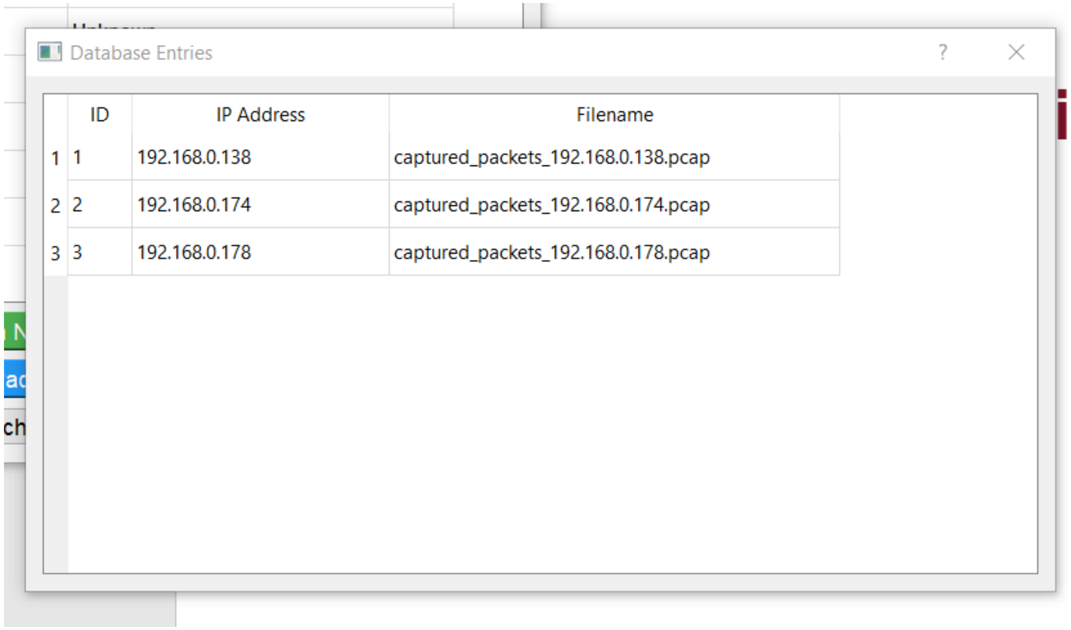


List of Active IoT Devices:

	IP Address	MAC Address	Vendor
1	192.168.0.1	22:6a:94:9c:53:91	Unknown
2	192.168.0.138	c8:09:a8:99:55:7a	Unknown
3	192.168.0.112	22:62:b9:0f:21:3c	Unknown
4	192.168.0.159	f8:ff:c2:48:40:b2	Apple, Inc.
5	192.168.0.176	dc:29:19:82:f5:ba	Unknown
6	192.168.0.179	cc:f4:11:a9:db:21	Unknown
7	192.168.0.178	b6:2e:bd:eb:5d:c7	Unknown

Displaying Stored Device Info in Database

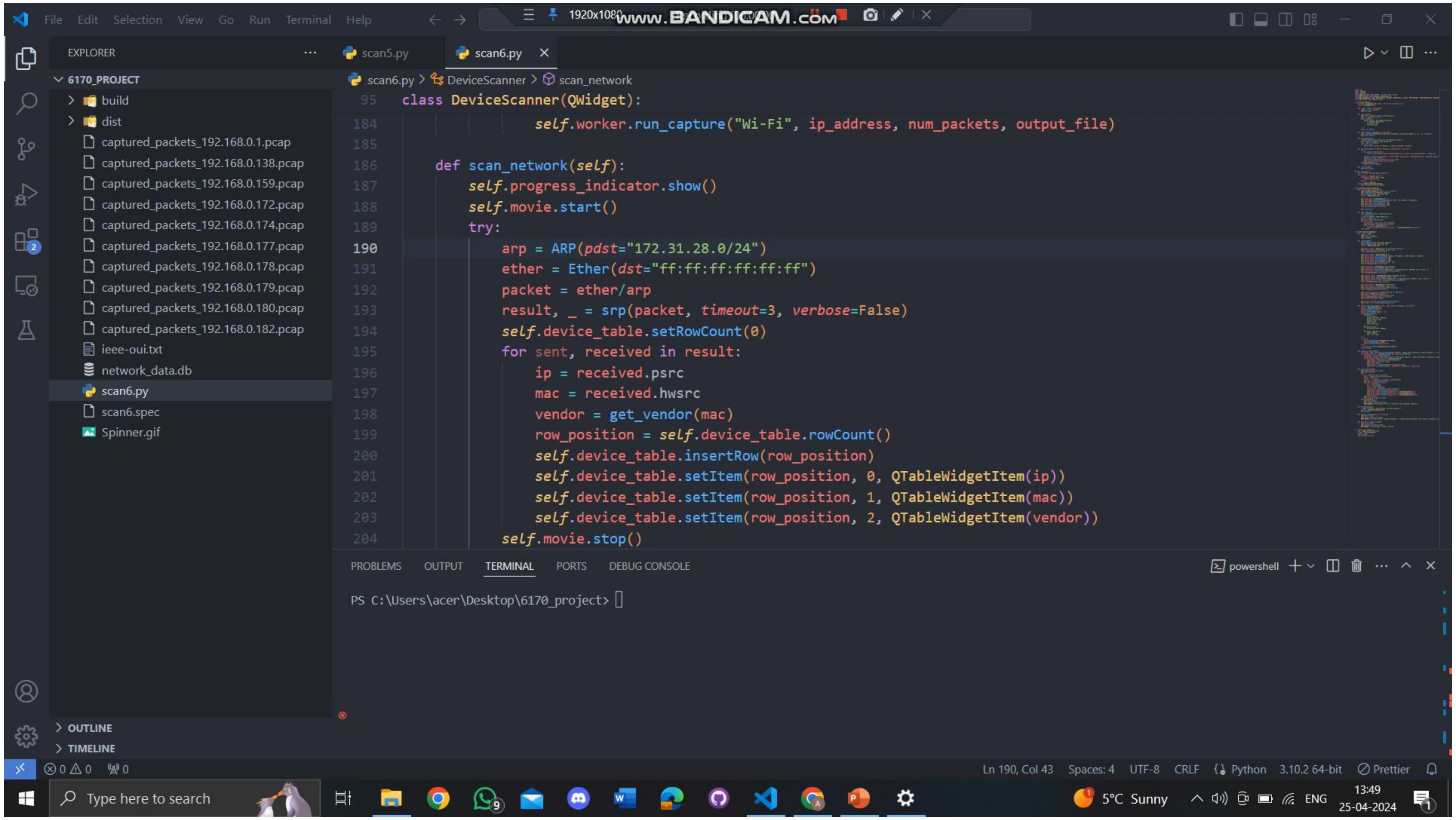
- IP Address
- Packet filename



Database Entries

	ID	IP Address	Filename
1	1	192.168.0.138	captured_packets_192.168.0.138.pcap
2	2	192.168.0.174	captured_packets_192.168.0.174.pcap
3	3	192.168.0.178	captured_packets_192.168.0.178.pcap

Implementation Demo


File
Edit
Selection
View
Go
Run
Terminal
Help
1920x1080
www.BANDICAM.com
EXPLORER
6170_PROJECT
build
dist
captured_packets_192.168.0.1.pcap
captured_packets_192.168.0.138.pcap
captured_packets_192.168.0.159.pcap
captured_packets_192.168.0.172.pcap
captured_packets_192.168.0.174.pcap
captured_packets_192.168.0.177.pcap
captured_packets_192.168.0.178.pcap
captured_packets_192.168.0.179.pcap
captured_packets_192.168.0.180.pcap
captured_packets_192.168.0.182.pcap
ieee-oui.txt
network_data.db
scan6.py
scan6.spec
Spinner.gif
OUTLINE
TIMELINE
scan5.py
scan6.py
scan6.py > DeviceScanner > scan_network
PROBLEMS
OUTPUT
TERMINAL
PORTS
DEBUG CONSOLE
powershell
Ln 190, Col 43
Spaces: 4
UTF-8
CRLF
Python
3.10.2 64-bit
Prettier
Type here to search
5°C Sunny
ENG
13:49
25-04-2024

```python
class DeviceScanner(QWidget):
                self.worker.run_capture("Wi-Fi", ip_address, num_packets, output_file)

    def scan_network(self):
        self.progress_indicator.show()
        self.movie.start()
        try:
            arp = ARP(pdst="172.31.28.0/24")
            ether = Ether(dst="ff:ff:ff:ff:ff:ff")
            packet = ether/arp
            result, _ = srp(packet, timeout=3, verbose=False)
            self.device_table.setRowCount(0)
            for sent, received in result:
                ip = received.psrc
                mac = received.hwsrc
                vendor = get_vendor(mac)
                row_position = self.device_table.rowCount()
                self.device_table.insertRow(row_position)
                self.device_table.setItem(row_position, 0, QTableWidgetItem(ip))
                self.device_table.setItem(row_position, 1, QTableWidgetItem(mac))
                self.device_table.setItem(row_position, 2, QTableWidgetItem(vendor))
            self.movie.stop()
```

```
PS C:\Users\acer\Desktop\6170_project>
```

Conclusion

- In conclusion, the provided application effectively fulfills the specified requirements for scanning a Wi-Fi network for active IoT devices, capturing packets, and saving device information along with the captured packets.
- the application effectively combines the functionalities of network scanning, packet capturing, and database management into a coordinated solution. It provides users with a convenient way to discover and monitor IoT devices on their Wi-Fi network while also facilitating packet capture for further analysis.

Individual Contribution of Each Team Member

Name	Contribution
Aniket Agarwal (40266485)	• Researched about tools to use • Implemented PyQT5 code to make GUI interface • Prepared presentation slides
Kalyani Batle (40243967)	• Researched about tools to use • Implemented code to scan network and display discovered IoT devices • Prepared presentation slides

THANK YOU