Network Traffic Analysis Tool[Major Project]

Introduction

The Network Traffic Analysis Tool is designed to monitor and analyze live network traffic. It captures packets passing through the system's network interface and extracts key details such as source IP, destination IP, ports, and protocols. This project provides insights into network usage and can help identify unusual or potentially malicious activity.

Tools Used

- Python
- Scapy
- Pandas
- Matplotlib
- VS Code
- Windows Command Prompt (Administrator)
- Npcap driver for packet sniffing support

Steps Involved in Building the Project

1. Set up Python environment and install required libraries using pip (scapy, pandas, matplotlib).
2. Used Scapy to sniff network packets in real-time.
3. Extracted important information from packets such as IPs, ports, and protocols.
4. Saved the extracted data into a CSV file for later use and analysis.
5. Created a Python script to read the CSV file and visualize data using bar charts (top IPs).
6. Ran all scripts with administrator rights to allow packet capture.
7. Verified outputs in terminal and saved results for review.

CODES and OUTPUT

```
from scapy.all import sniff

def packet_callback(packet):
    print(packet.summary())

print("Sniffing packets... Press Ctrl+C to stop.")
sniff(prn=packet_callback, count=20)
```

replacing

```
from scapy.all import sniff, IP, TCP, UDP, ICMP

def process_packet(packet):
    if IP in packet:
```

```
        src_ip = packet[IP].src
        dst_ip = packet[IP].dst
        if TCP in packet:
            protocol = "TCP"
            src_port = packet[TCP].sport
            dst_port = packet[TCP].dport
            print(f"[TCP] {src_ip}:{src_port} → {dst_ip}:{dst_port}")
        elif UDP in packet:
            protocol = "UDP"
            src_port = packet[UDP].sport
            dst_port = packet[UDP].dport
            print(f"[UDP] {src_ip}:{src_port} → {dst_ip}:{dst_port}")
        elif ICMP in packet:
            protocol = "ICMP"
            print(f"[ICMP] {src_ip} → {dst_ip}")
        else:
            print(f"[Other] {src_ip} → {dst_ip}")

print("Sniffing packets... Press Ctrl+C to stop.")
sniff(prn=process_packet, count=30)
```

again replacing

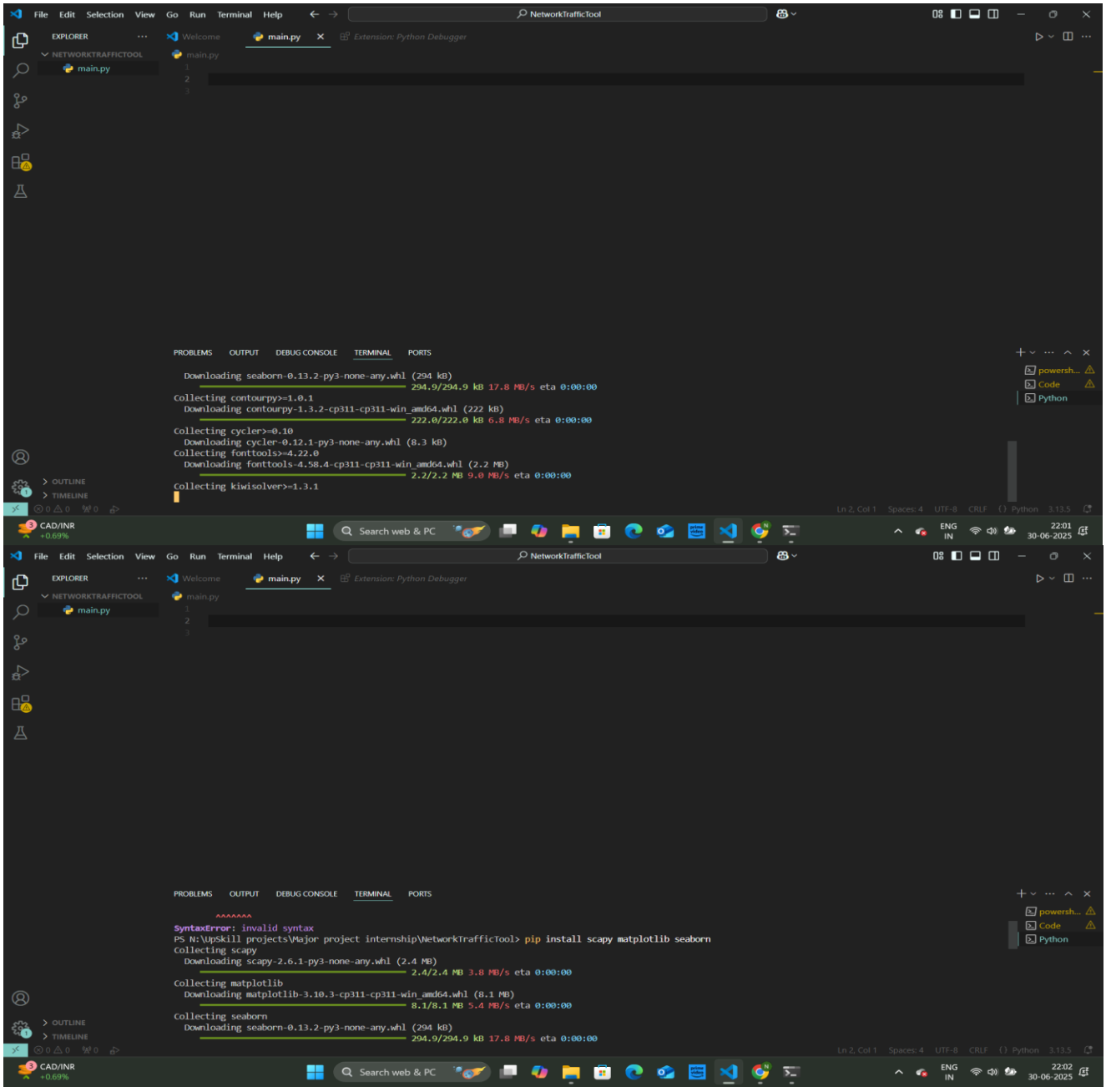
```
from scapy.all import sniff, IP, TCP, UDP, ICMP
import csv

csv_file = open("packet_log.csv", "w", newline="")
csv_writer = csv.writer(csv_file)
csv_writer.writerow(["Protocol", "Source IP", "Source Port", "Destination
IP", "Destination Port"])

def process_packet(packet):
    if IP in packet:
        src_ip = packet[IP].src
        dst_ip = packet[IP].dst

        if TCP in packet:
            protocol = "TCP"
            src_port = packet[TCP].sport
            dst_port = packet[TCP].dport
        elif UDP in packet:
            protocol = "UDP"
            src_port = packet[UDP].sport
            dst_port = packet[UDP].dport
        elif ICMP in packet:
            protocol = "ICMP"
            src_port = "-"
            dst_port = "-"
```

```
        else:
            protocol = "Other"
            src_port = "-"
            dst_port = "-"

        csv_writer.writerow([protocol, src_ip, src_port, dst_ip, dst_port])
        print(f"[{protocol}] {src_ip}:{src_port} → {dst_ip}:{dst_port}")

print("Sniffing packets and saving to packet_log.csv... Press Ctrl+C to
stop.")
sniff(prn=process_packet, count=50)
csv_file.close()
```

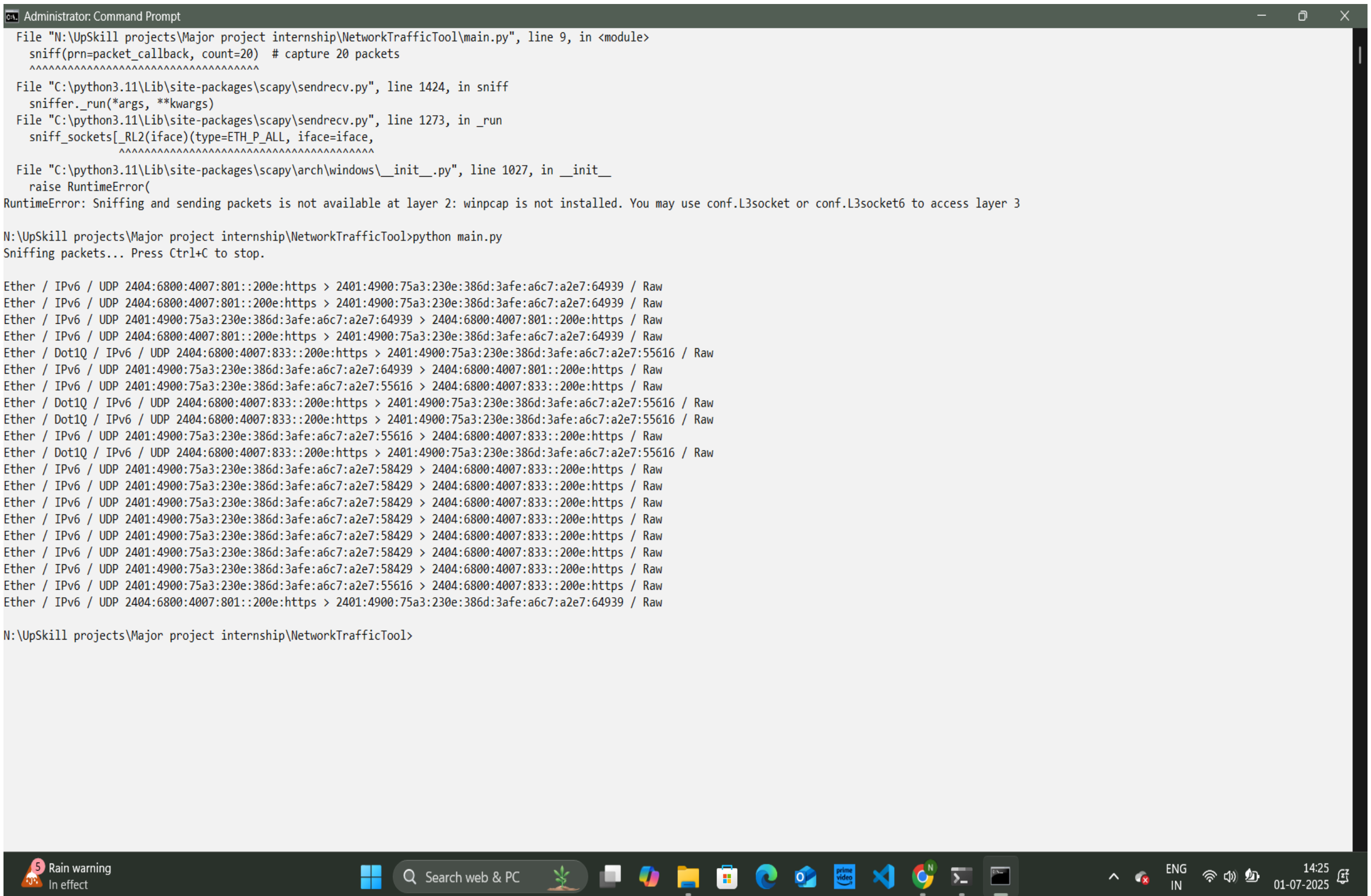
this create a csv file

To get the graph of the network traffic

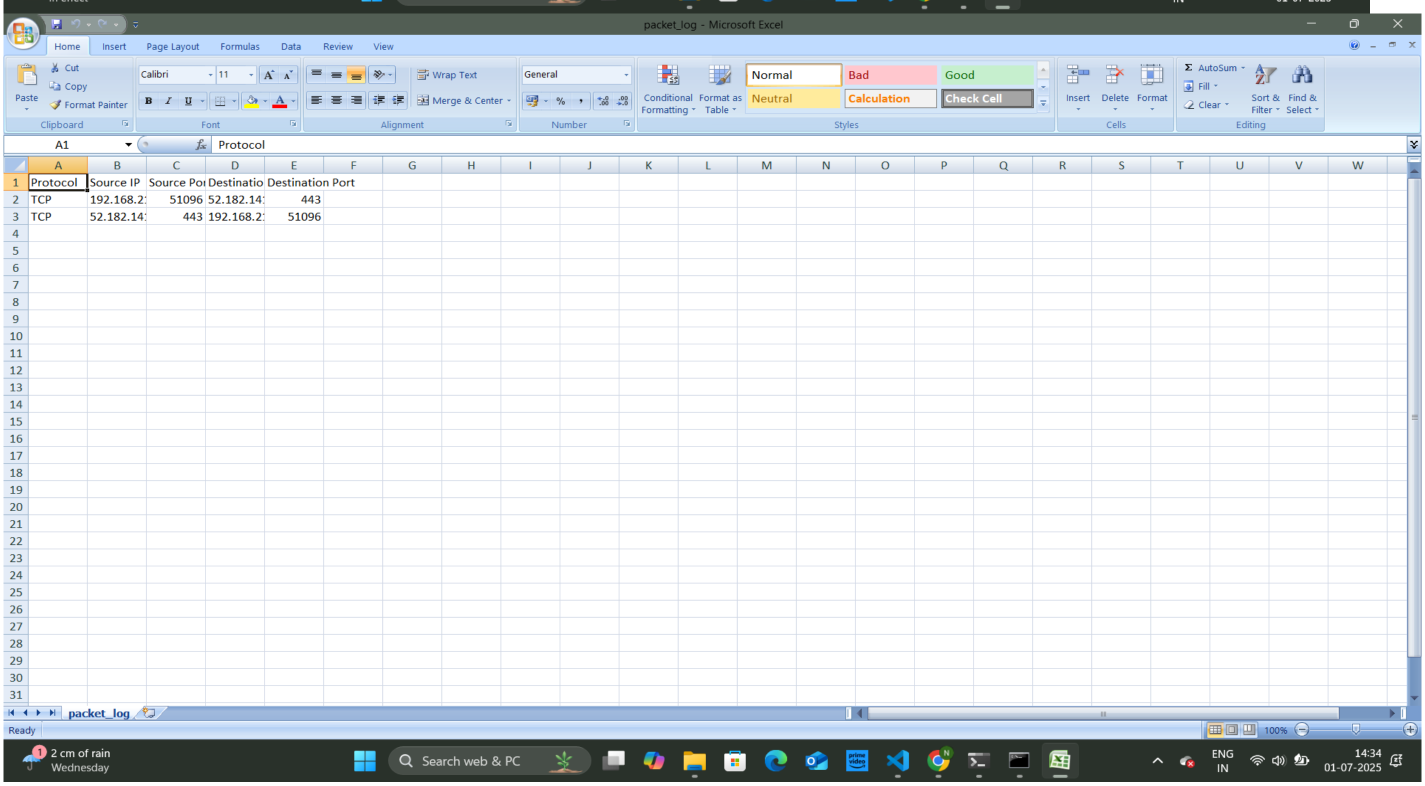
```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv("packet_log.csv")
top_sources = df['Source IP'].value_counts().head(5)

plt.figure(figsize=(8,5))
top_sources.plot(kind='bar', color='skyblue')
plt.title("Top 5 Source IPs")
plt.xlabel("IP Address")
plt.ylabel("Number of Packets")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```


File Edit Selection View Go Run Terminal Help
NetworkTrafficTool
EXPLORER
NETWORKTRAFFICTOOL
main.py
Welcome
main.py
Extension: Python Debugger
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Downloading seaborn-0.13.2-py3-none-any.whl (294 kB)
294.9/294.9 kB 17.8 MB/s eta 0:00:00
Collecting contourpy>=1.0.1
Downloading contourpy-1.3.2-cp311-cp311-win_amd64.whl (222 kB)
222.0/222.0 kB 6.8 MB/s eta 0:00:00
Collecting cycler>=0.10
Downloading cycler-0.12.1-py3-none-any.whl (8.3 kB)
Collecting fonttools>=4.22.0
Downloading fonttools-4.58.4-cp311-cp311-win_amd64.whl (2.2 MB)
2.2/2.2 MB 9.0 MB/s eta 0:00:00
Collecting kiwisolver>=1.3.1
powersh...
Code
Python
OUTLINE
TIMELINE
Ln 2, Col 1 Spaces: 4 UTF-8 CRLF Python 3.13.5
CAD/INR +0.69%
Search web & PC
ENG IN
22:01 30-06-2025

File Edit Selection View Go Run Terminal Help
NetworkTrafficTool
EXPLORER
NETWORKTRAFFICTOOL
main.py
Welcome
main.py
Extension: Python Debugger
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
^^^^^^^
SyntaxError: invalid syntax
PS N:\Upskill projects\Major project internship\NetworkTrafficTool> pip install scapy matplotlib seaborn
Collecting scapy
Downloading scapy-2.6.1-py3-none-any.whl (2.4 MB)
2.4/2.4 MB 3.8 MB/s eta 0:00:00
Collecting matplotlib
Downloading matplotlib-3.10.3-cp311-cp311-win_amd64.whl (8.1 MB)
8.1/8.1 MB 5.4 MB/s eta 0:00:00
Collecting seaborn
Downloading seaborn-0.13.2-py3-none-any.whl (294 kB)
294.9/294.9 kB 17.8 MB/s eta 0:00:00
powersh...
Code
Python
OUTLINE
TIMELINE
Ln 2, Col 1 Spaces: 4 UTF-8 CRLF Python 3.13.5
CAD/INR +0.69%
Search web & PC
ENG IN
22:02 30-06-2025

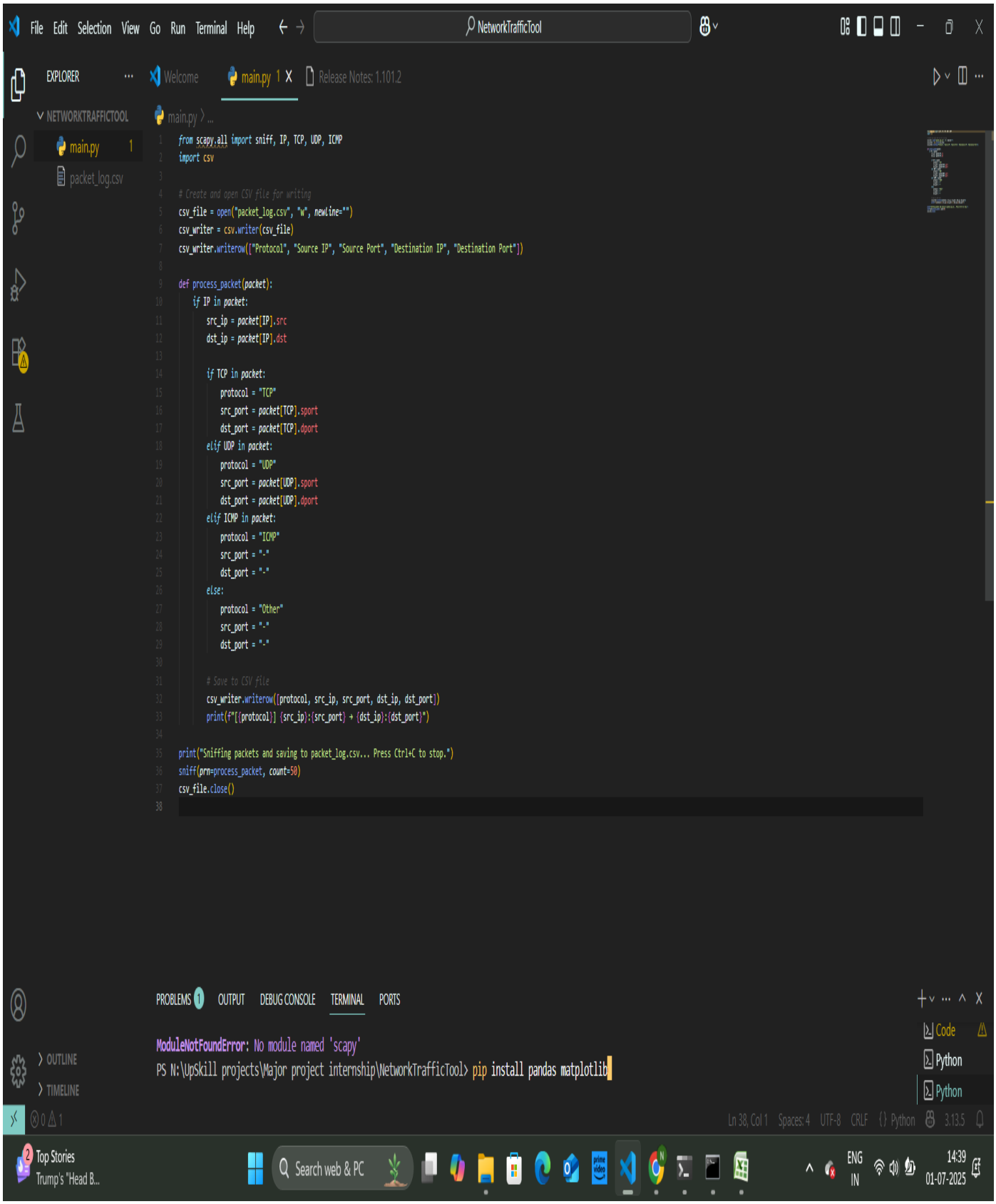


```
Administrator: Command Prompt
  File "N:\UpSkill projects\Major project internship\NetworkTrafficTool\main.py", line 9, in <module>
    sniff(prn=packet_callback, count=20)  # capture 20 packets
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\python3.11\Lib\site-packages\scapy\sendrecv.py", line 1424, in sniff
    sniffer._run(*args, **kwargs)
  File "C:\python3.11\Lib\site-packages\scapy\sendrecv.py", line 1273, in _run
    sniff_sockets[_RL2(iface)(type=ETH_P_ALL, iface=iface,
                  ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\python3.11\Lib\site-packages\scapy\arch\windows\__init__.py", line 1027, in __init__
    raise RuntimeError(
RuntimeError: Sniffing and sending packets is not available at layer 2: winpcap is not installed. You may use conf.L3socket or conf.L3socket6 to access layer 3

N:\UpSkill projects\Major project internship\NetworkTrafficTool>python main.py
Sniffing packets... Press Ctrl+C to stop.

Ether / IPv6 / UDP 2404:6800:4007:801::200e:https > 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:64939 / Raw
Ether / IPv6 / UDP 2404:6800:4007:801::200e:https > 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:64939 / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:64939 > 2404:6800:4007:801::200e:https / Raw
Ether / IPv6 / UDP 2404:6800:4007:801::200e:https > 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:64939 / Raw
Ether / Dot1Q / IPv6 / UDP 2404:6800:4007:833::200e:https > 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:55616 / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:64939 > 2404:6800:4007:801::200e:https / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:55616 > 2404:6800:4007:833::200e:https / Raw
Ether / Dot1Q / IPv6 / UDP 2404:6800:4007:833::200e:https > 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:55616 / Raw
Ether / Dot1Q / IPv6 / UDP 2404:6800:4007:833::200e:https > 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:55616 / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:55616 > 2404:6800:4007:833::200e:https / Raw
Ether / Dot1Q / IPv6 / UDP 2404:6800:4007:833::200e:https > 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:55616 / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:58429 > 2404:6800:4007:833::200e:https / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:58429 > 2404:6800:4007:833::200e:https / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:58429 > 2404:6800:4007:833::200e:https / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:58429 > 2404:6800:4007:833::200e:https / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:58429 > 2404:6800:4007:833::200e:https / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:58429 > 2404:6800:4007:833::200e:https / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:58429 > 2404:6800:4007:833::200e:https / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:55616 > 2404:6800:4007:833::200e:https / Raw
Ether / IPv6 / UDP 2404:6800:4007:801::200e:https > 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:64939 / Raw

N:\UpSkill projects\Major project internship\NetworkTrafficTool>
```

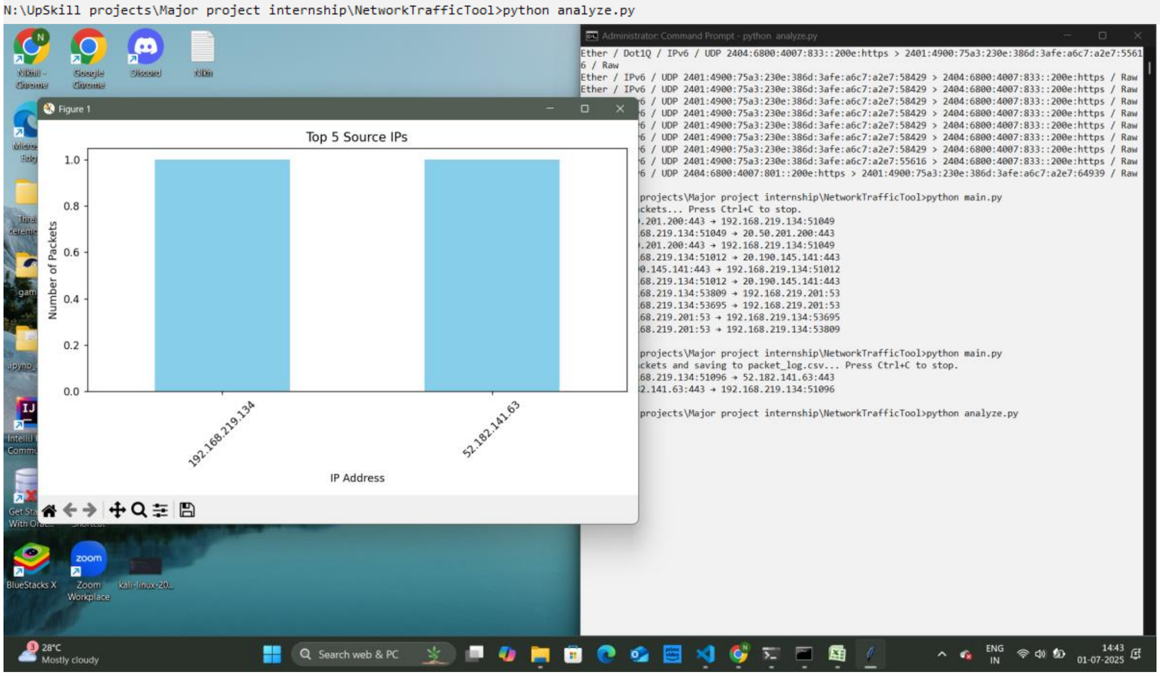
packet_log - Microsoft Excel

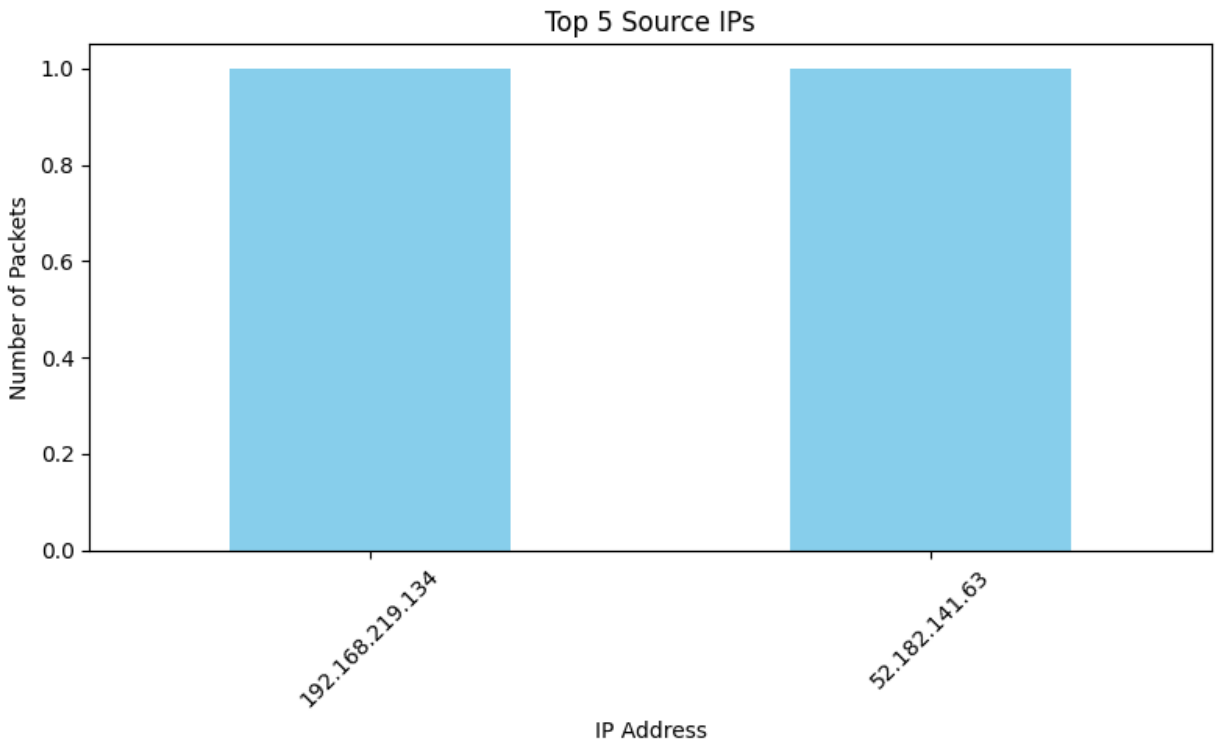
Protocol	Source IP	Source Poi	Destinatio	Destination Port
TCP	192.168.2:	51096	52.182.14:	443
TCP	52.182.14:	443	192.168.2:	51096


File Edit Selection View Go Run Terminal Help
NetworkTrafficTool
EXPLORER
Welcome
main.py 1
Release Notes: 1.101.2
NETWORKTRAFFICTOOL
main.py 1
packet_log.csv
main.py > ...
from scapy.all import sniff, IP, TCP, UDP, ICMP
import csv
# Create and open CSV file for writing
csv_file = open("packet_log.csv", "w", newline="")
csv_writer = csv.writer(csv_file)
csv_writer.writerow(["Protocol", "Source IP", "Source Port", "Destination IP", "Destination Port"])
def process_packet(packet):
if IP in packet:
src_ip = packet[IP].src
dst_ip = packet[IP].dst
if TCP in packet:
protocol = "TCP"
src_port = packet[TCP].sport
dst_port = packet[TCP].dport
elif UDP in packet:
protocol = "UDP"
src_port = packet[UDP].sport
dst_port = packet[UDP].dport
elif ICMP in packet:
protocol = "ICMP"
src_port = "-"
dst_port = "-"
else:
protocol = "Other"
src_port = "-"
dst_port = "-"
# Save to CSV file
csv_writer.writerow([protocol, src_ip, src_port, dst_ip, dst_port])
print(f"[{protocol}] {src_ip}:{src_port} → {dst_ip}:{dst_port}")
print("Sniffing packets and saving to packet_log.csv... Press Ctrl+C to stop.")
sniff(prn=process_packet, count=50)
csv_file.close()
PROBLEMS 1
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
ModuleNotFoundError: No module named 'scapy'
PS N:\UpSkill projects\Major project internship\NetworkTrafficTool> pip install pandas matplotlib
Code
Python
Python
OUTLINE
TIMELINE
Ln 38, Col 1
Spaces: 4
UTF-8
CRLF
Python
3.13.5
Top Stories
Trump's "Head B...
Search web & PC
ENG
IN
14:39
01-07-2025

Administrator: Command Prompt - python analyze.py

```
Microsoft Windows [Version 10.0.26100.4351]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32>N:

N:\>cd "UpSkill projects\Major project internship\NetworkTrafficTool"

N:\UpSkill projects\Major project internship\NetworkTrafficTool>python main.py
WARNING: No libpcap provider available ! pcap won't be used
Sniffing packets... Press Ctrl+C to stop.

Traceback (most recent call last):
  File "N:\UpSkill projects\Major project internship\NetworkTrafficTool\main.py", line 9, in <module>
    sniff(prn=packet_callback, count=20)  # capture 20 packets
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\python3.11\Lib\site-packages\scapy\sendrecv.py", line 1424, in sniff
    sniffer._run(*args, **kwargs)
  File "C:\python3.11\Lib\site-packages\scapy\sendrecv.py", line 1273, in _run
    sniff_sockets[_RL2(iface)(type=ETH_P_ALL, iface=iface,
                  ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\python3.11\Lib\site-packages\scapy\arch\windows\__init__.py", line 1027, in __init__
    raise RuntimeError(
RuntimeError: Sniffing and sending packets is not available at layer 2: winpcap is not installed. You may use conf.L3socket or conf.L3socket6 to access layer 3

N:\UpSkill projects\Major project internship\NetworkTrafficTool>python main.py
Sniffing packets... Press Ctrl+C to stop.

Ether / IPv6 / UDP 2404:6800:4007:801::200e:https > 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:64939 / Raw
Ether / IPv6 / UDP 2404:6800:4007:801::200e:https > 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:64939 / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:64939 > 2404:6800:4007:801::200e:https / Raw
Ether / IPv6 / UDP 2404:6800:4007:801::200e:https > 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:64939 / Raw
Ether / Dot1Q / IPv6 / UDP 2404:6800:4007:833::200e:https > 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:55616 / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:64939 > 2404:6800:4007:801::200e:https / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:55616 > 2404:6800:4007:833::200e:https / Raw
Ether / Dot1Q / IPv6 / UDP 2404:6800:4007:833::200e:https > 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:55616 / Raw
Ether / Dot1Q / IPv6 / UDP 2404:6800:4007:833::200e:https > 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:55616 / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:55616 > 2404:6800:4007:833::200e:https / Raw
Ether / Dot1Q / IPv6 / UDP 2404:6800:4007:833::200e:https > 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:55616 / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:58429 > 2404:6800:4007:833::200e:https / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:58429 > 2404:6800:4007:833::200e:https / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:58429 > 2404:6800:4007:833::200e:https / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:58429 > 2404:6800:4007:833::200e:https / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:58429 > 2404:6800:4007:833::200e:https / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:58429 > 2404:6800:4007:833::200e:https / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:58429 > 2404:6800:4007:833::200e:https / Raw
Ether / IPv6 / UDP 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:55616 > 2404:6800:4007:833::200e:https / Raw
Ether / IPv6 / UDP 2404:6800:4007:801::200e:https > 2401:4900:75a3:230e:386d:3afe:a6c7:a2e7:64939 / Raw

N:\UpSkill projects\Major project internship\NetworkTrafficTool>python main.py
Sniffing packets... Press Ctrl+C to stop.
[TCP] 20.50.201.200:443 → 192.168.219.134:51049
[TCP] 192.168.219.134:51049 → 20.50.201.200:443
[TCP] 20.50.201.200:443 → 192.168.219.134:51049
[TCP] 192.168.219.134:51012 → 20.190.145.141:443
[TCP] 20.190.145.141:443 → 192.168.219.134:51012
[TCP] 192.168.219.134:51012 → 20.190.145.141:443
[UDP] 192.168.219.134:53809 → 192.168.219.201:53
[UDP] 192.168.219.134:53695 → 192.168.219.201:53
[UDP] 192.168.219.201:53 → 192.168.219.134:53695
[UDP] 192.168.219.201:53 → 192.168.219.134:53809

N:\UpSkill projects\Major project internship\NetworkTrafficTool>python main.py
Sniffing packets and saving to packet_log.csv... Press Ctrl+C to stop.
[TCP] 192.168.219.134:51096 → 52.182.141.63:443
[TCP] 52.182.141.63:443 → 192.168.219.134:51096

N:\UpSkill projects\Major project internship\NetworkTrafficTool>python analyze.py
```





Conclusion

The Network Traffic Analysis Tool successfully demonstrates how real-time packet capture and analysis can be achieved using Python and Scapy.

SUBMITTED BY:-

Name : Nikhil Kumar Purohit

Gmail: nikhilkumarpurohit3@gmail.com

Mob no. : 8280057642