Faculty of Engineering & Technology

Electrical & Computer Engineering Department

ENCS3320-Computer Networks
Project#1

Prepared by:

Name: Ahmad Abu Saleem **ID**:1201315

Name : Sami Saed **ID**: 1221643

Name : Reem says **ID**:1220827

Instructor: Dr. Imad Tartir

Section: 4
Date: 21/12/2023

Part 1:

1-In your own words, what are ping, tracert, nslookup, and telnet (write one sentence for each one)

1) Ping: A command used to determine the time it takes for the data to travel between two

2)Tracert: is a command-line utility that traces the path of network packets from the source to

the destination.

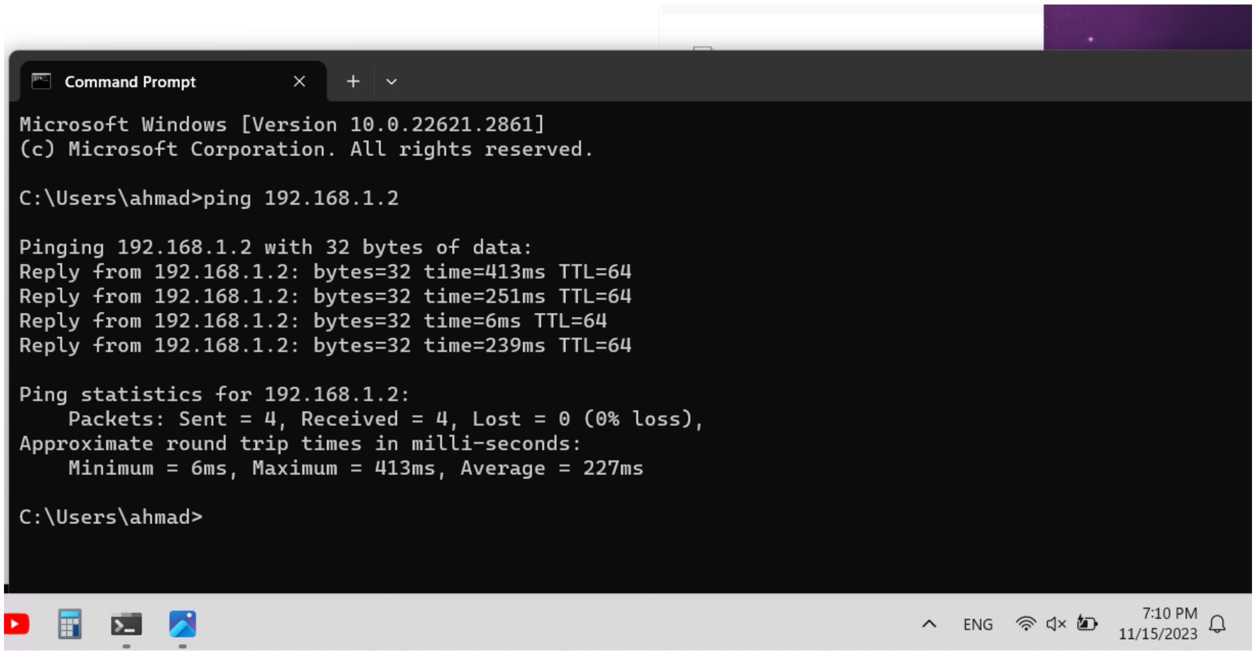
3)Nslookup: is a command-line tool used to query DNS to retrieve information about a domain,

including its IP address, IPV4 address, and alias name.

4) Telnet is a network protocol used for establishing remote connections to devices or servers, enabling users to interact with the target system through a command-line interface.

2-Make sure that your computer is connected to the internet and then run the following commands:

1-Ping a device in the same network, e.g. from a laptop to a smartphone.



```
Microsoft Windows [Version 10.0.22621.2861]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ahmad>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:
Reply from 192.168.1.2: bytes=32 time=413ms TTL=64
Reply from 192.168.1.2: bytes=32 time=251ms TTL=64
Reply from 192.168.1.2: bytes=32 time=6ms TTL=64
Reply from 192.168.1.2: bytes=32 time=239ms TTL=64

Ping statistics for 192.168.1.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 6ms, Maximum = 413ms, Average = 227ms

C:\Users\ahmad>
```

In the previous screenshot, we made Pinging 192.168.1.2 with 32 bytes of data: the ping utility is sending packets of size 32 bytes to the specified IP address.

Reply from 192.168.1.2: bytes = 32, time = 413ms TTL=64: These lines represent the replies received from the destination IP address (a smartphone). Each line indicates a successful reply, displaying the response time in milliseconds (time = 413 ms), the size of the packet in bytes (bytes = 32), and the Time to Live (TTL) value, which represents the number of hops the packet can traverse before being discarded (TTL = 64).

The "time=413ms" indicates that the round trip time for the ICMP (Internet Control Message Protocol) echo request and reply packets to travel from the sender (laptop) to the destination (192.168.1.2) and back to the sender is 413 milliseconds.

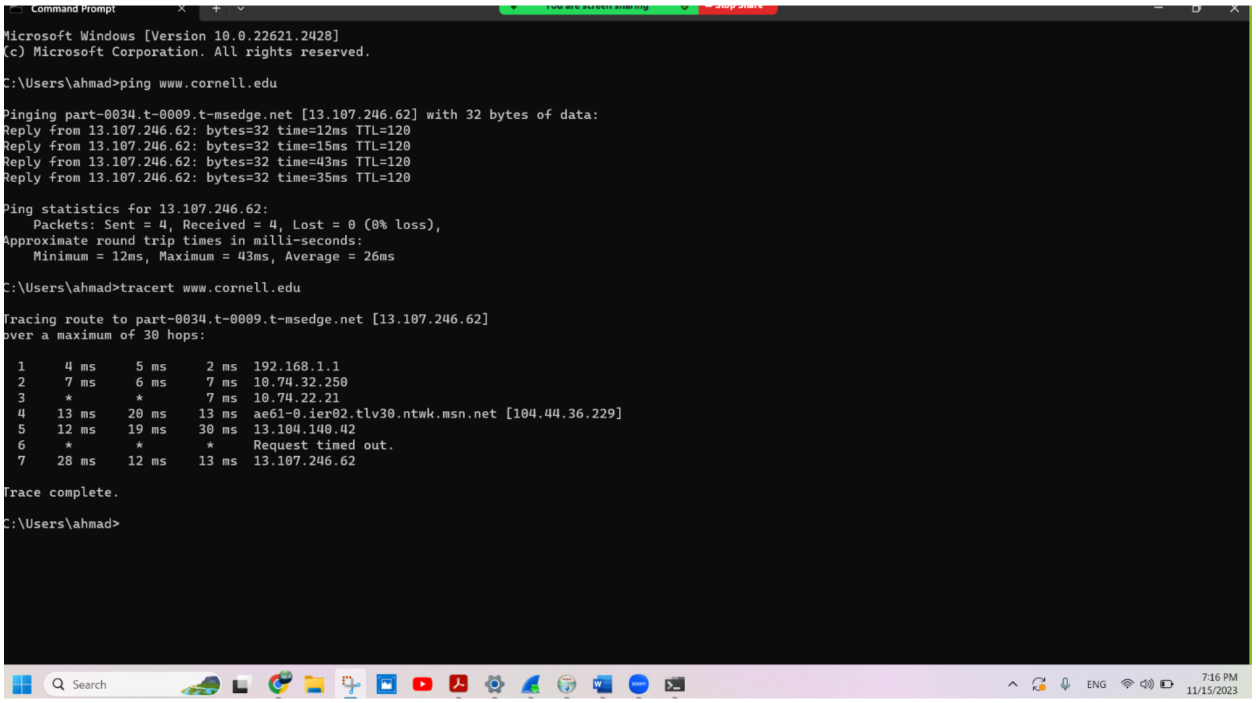
Ping statistics for 192.168.1.2, this line provides a summary of the ping statistics for the destination IP address (a smartphone).

Packets: Sent = 4, Received = 4, Lost = 0 (0% loss): This line indicates the number of packets sent, received, and lost during the ping operation. In our situation, all packets were received successfully, resulting in 0% packet loss.

Approximate round trip times in milliseconds: This section provides additional information about the

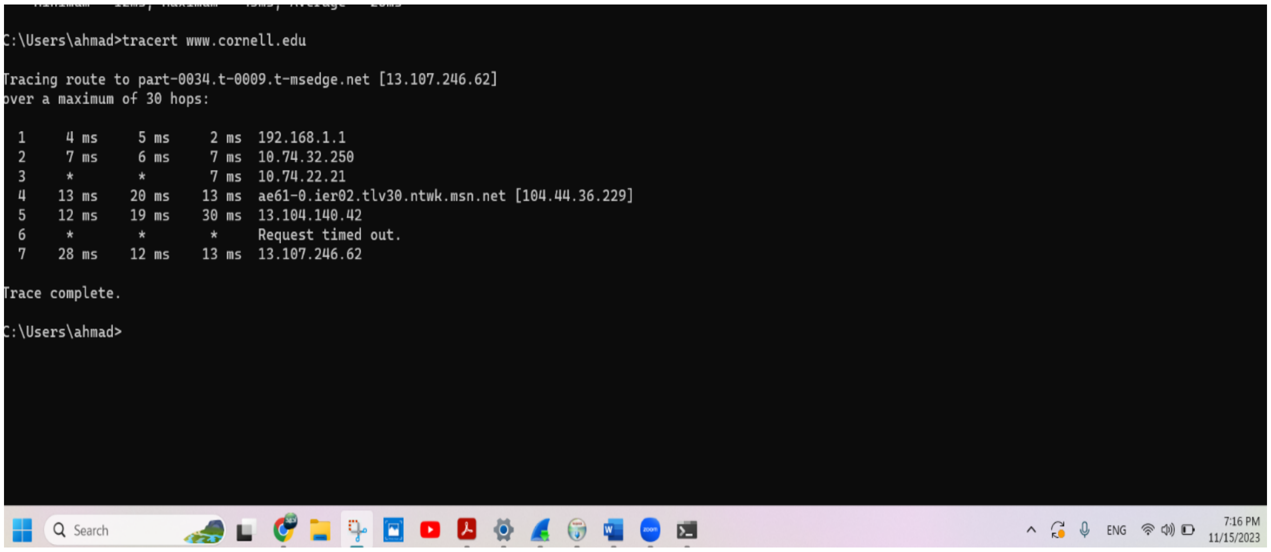
response times for the sent packets. Minimum = 6ms, Maximum = 413ms, Average = 227ms: These values represent the minimum, maximum, and average response times, respectively, measured in milliseconds

2-ping www.cornell.edu



```
Microsoft Windows [Version 10.0.22621.2428]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ahmad>ping www.cornell.edu

Pinging part-0034.t-0009.t-msedge.net [13.107.246.62] with 32 bytes of data:
Reply from 13.107.246.62: bytes=32 time=12ms TTL=120
Reply from 13.107.246.62: bytes=32 time=15ms TTL=120
Reply from 13.107.246.62: bytes=32 time=43ms TTL=120
Reply from 13.107.246.62: bytes=32 time=35ms TTL=120

Ping statistics for 13.107.246.62:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 12ms, Maximum = 43ms, Average = 26ms

C:\Users\ahmad>tracert www.cornell.edu

Tracing route to part-0034.t-0009.t-msedge.net [13.107.246.62]
over a maximum of 30 hops:

  1     4 ms     5 ms     2 ms  192.168.1.1
  2     7 ms     6 ms     7 ms  10.74.32.250
  3     *        *        7 ms  10.74.22.21
  4    13 ms    20 ms    13 ms  ae61-0.ier02.tlv30.ntwk.msn.net [104.44.36.229]
  5    12 ms    19 ms    30 ms  13.104.140.42
  6     *        *        *     Request timed out.
  7    28 ms    12 ms    13 ms  13.107.246.62

Trace complete.

C:\Users\ahmad>
```

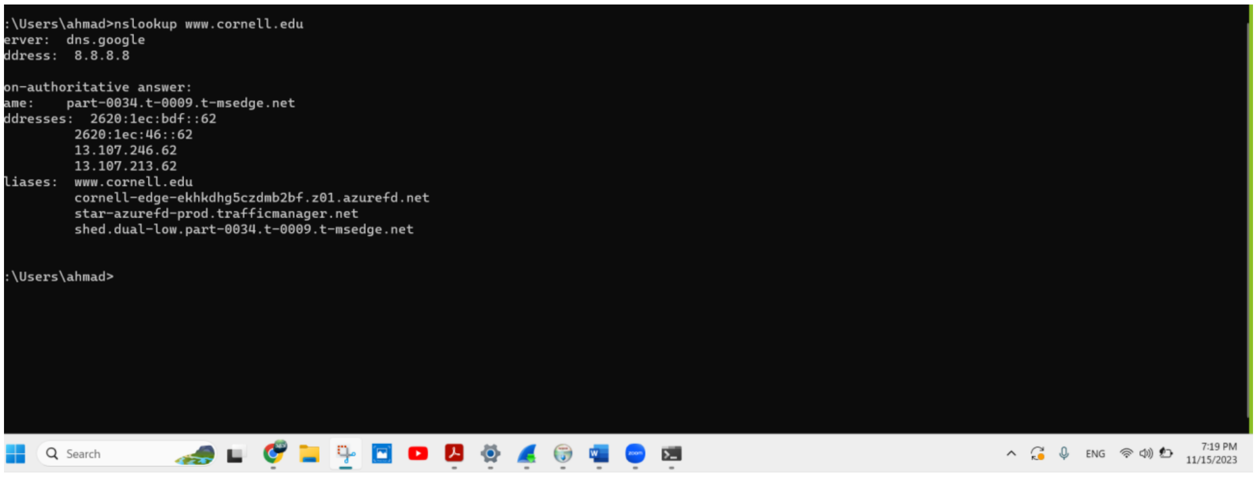
3-tracert www.cornell.edu



```
C:\Users\ahmad>tracert www.cornell.edu

Tracing route to part-0034.t-0009.t-msedge.net [13.107.246.62]
over a maximum of 30 hops:

  1     4 ms     5 ms     2 ms  192.168.1.1
  2     7 ms     6 ms     7 ms  10.74.32.250
  3     *        *        7 ms  10.74.22.21
  4    13 ms    20 ms    13 ms  ae61-0.ier02.tlv30.ntwk.msn.net [104.44.36.229]
  5    12 ms    19 ms    30 ms  13.104.140.42
  6     *        *        *     Request timed out.
  7    28 ms    12 ms    13 ms  13.107.246.62

Trace complete.

C:\Users\ahmad>
```

The output shows that the packets pass through seven network devices before reaching the destination, with round-trip times ranging from 2 to 30 milliseconds. The first hops are on the local network, followed by several hops on the Cogent Communications network, before finally reaching the destination at IP address 13.107.246.62 This server appears to be hosting the website for Cornell University

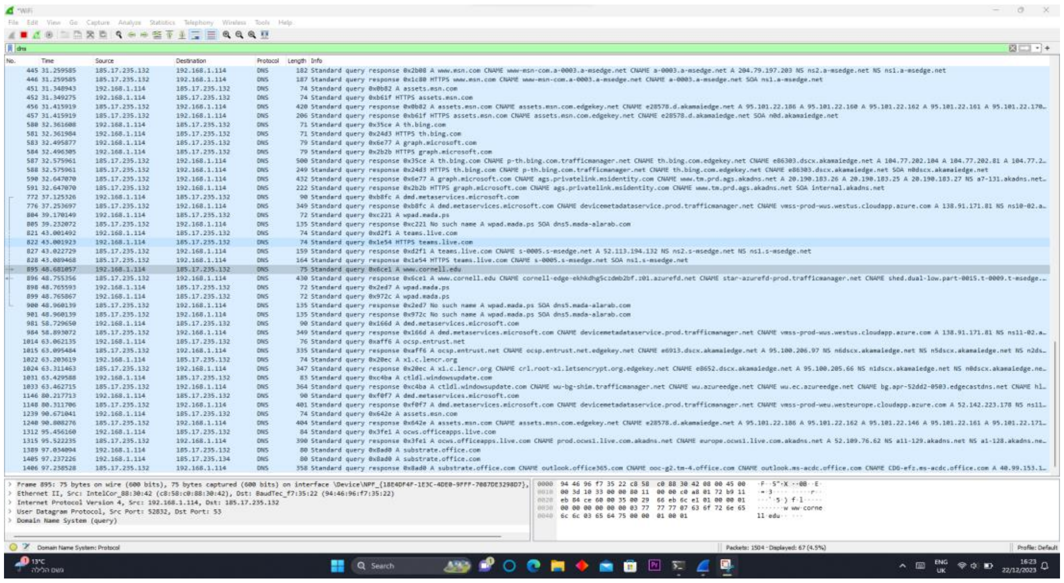
4-nslookup www.cornell.edu



```
:\Users\ahmad>nslookup www.cornell.edu
erver:  dns.google
ddress:  8.8.8.8

on-authoritative answer:
ame:    part-0034.t-0009.t-msedge.net
ddresses:  2620:1ec:bdf::62
          2620:1ec:46::62
          13.107.246.62
          13.107.213.62
liases:   www.cornell.edu
          cornell-edge-ekhkdhg5czdmb2bf.z01.azurefd.net
          star-azurefd-prod.trafficmanager.net
          shed.dual-low.part-0034.t-0009.t-msedge.net


:\Users\ahmad>
```

The output above shows the alias name that represents two names for the same IP address. In our case, it shows the actual name for the website, www.cornell.edu, and we discovered it is the alias name, and as we know in the DNS records, the real name takes the type CNAME and the alias name takes the type A.

5-use Wireshark to capture some DNS messages.



```
No.   Time        Source          Destination     Protocol Length Info
  445 31.259585  185.17.235.132  192.168.1.114   DNS   182 Standard query response 0x2b08 A www.msn.com CNAME www-msn-com.a-0003.a-msedge.net CNAME a-0003.a-msedge.net A 204.79.197.203 NS ns2.a-msedge.net NS ns1.a-msedge.net
  446 31.259585  185.17.235.132  192.168.1.114   DNS   187 Standard query response 0x1c80 HTTPS www.msn.com CNAME www-msn-com.a-0003.a-msedge.net CNAME a-0003.a-msedge.net SOA ns1.a-msedge.net
  451 31.348943  192.168.1.114   185.17.235.132  DNS    74 Standard query 0x0b82 A assets.msn.com
  452 31.349275  192.168.1.114   185.17.235.132  DNS    74 Standard query 0xb61f HTTPS assets.msn.com
  456 31.415919  185.17.235.132  192.168.1.114   DNS   420 Standard query response 0x0b82 A assets.msn.com CNAME assets.msn.com.edgekey.net CNAME e28578.d.akamaiedge.net A 95.101.22.186 A 95.101.22.160 A 95.101.22.162 A 95.101.22.161 A 95.101.22.170…
  457 31.415919  185.17.235.132  192.168.1.114   DNS   206 Standard query response 0xb61f HTTPS assets.msn.com CNAME assets.msn.com.edgekey.net CNAME e28578.d.akamaiedge.net SOA n0d.akamaiedge.net
  580 32.361608  192.168.1.114   185.17.235.132  DNS    71 Standard query 0x35ce A th.bing.com
  581 32.361984  192.168.1.114   185.17.235.132  DNS    71 Standard query 0x24d3 HTTPS th.bing.com
  583 32.495877  192.168.1.114   185.17.235.132  DNS    79 Standard query 0x6e77 A graph.microsoft.com
  584 32.496305  192.168.1.114   185.17.235.132  DNS    79 Standard query 0x2b2b HTTPS graph.microsoft.com
  587 32.575961  185.17.235.132  192.168.1.114   DNS   500 Standard query response 0x35ce A th.bing.com CNAME p-th.bing.com.trafficmanager.net CNAME th.bing.com.edgekey.net CNAME e86303.dscx.akamaiedge.net A 104.77.202.104 A 104.77.202.81 A 104.77.2…
  588 32.575961  185.17.235.132  192.168.1.114   DNS   249 Standard query response 0x24d3 HTTPS th.bing.com CNAME p-th.bing.com.trafficmanager.net CNAME th.bing.com.edgekey.net CNAME e86303.dscx.akamaiedge.net SOA n0dscx.akamaiedge.net
  590 32.647070  185.17.235.132  192.168.1.114   DNS   432 Standard query response 0x6e77 A graph.microsoft.com CNAME ags.privatelink.msidentity.com CNAME www.tm.prd.ags.akadns.net A 20.190.183.26 A 20.190.183.25 A 20.190.183.27 NS a7-131.akadns.net…
  591 32.647070  185.17.235.132  192.168.1.114   DNS   222 Standard query response 0x2b2b HTTPS graph.microsoft.com CNAME ags.privatelink.msidentity.com CNAME www.tm.prd.ags.akadns.net SOA internal.akadns.net
  772 37.125326  192.168.1.114   185.17.235.132  DNS    90 Standard query 0xb8fc A dmd.metaservices.microsoft.com
  776 37.253697  185.17.235.132  192.168.1.114   DNS   349 Standard query response 0xb8fc A dmd.metaservices.microsoft.com CNAME devicemetadataservice.prod.trafficmanager.net CNAME vmss-prod-wus.westus.cloudapp.azure.com A 138.91.171.81 NS ns10-02.a…
  804 39.170149  192.168.1.114   185.17.235.132  DNS    72 Standard query 0xc221 A wpad.mada.ps
  805 39.232072  185.17.235.132  192.168.1.114   DNS   135 Standard query response 0xc221 No such name A wpad.mada.ps SOA dns5.mada-alarab.com
  821 43.001492  192.168.1.114   185.17.235.132  DNS    74 Standard query 0xd2f1 A teams.live.com
  822 43.001923  192.168.1.114   185.17.235.132  DNS    74 Standard query 0x1e54 HTTPS teams.live.com
  827 43.022729  185.17.235.132  192.168.1.114   DNS   159 Standard query response 0xd2f1 A teams.live.com CNAME s-0005.s-msedge.net A 52.113.194.132 NS ns2.s-msedge.net NS ns1.s-msedge.net
  828 43.089468  185.17.235.132  192.168.1.114   DNS   164 Standard query response 0x1e54 HTTPS teams.live.com CNAME s-0005.s-msedge.net SOA ns1.s-msedge.net
  895 48.681057  192.168.1.114   185.17.235.132  DNS    75 Standard query 0x6ce1 A www.cornell.edu
  896 48.755356  185.17.235.132  192.168.1.114   DNS   430 Standard query response 0x6ce1 A www.cornell.edu CNAME cornell-edge-ekhkdhg5czdmb2bf.z01.azurefd.net CNAME star-azurefd-prod.trafficmanager.net CNAME shed.dual-low.part-0015.t-0009.t-msedge…
  898 48.765593  192.168.1.114   185.17.235.132  DNS    72 Standard query 0x2ed7 A wpad.mada.ps
  899 48.765867  192.168.1.114   185.17.235.132  DNS    72 Standard query 0x972c A wpad.mada.ps
  900 48.960139  185.17.235.132  192.168.1.114   DNS   135 Standard query response 0x2ed7 No such name A wpad.mada.ps SOA dns5.mada-alarab.com
  901 48.960139  185.17.235.132  192.168.1.114   DNS   135 Standard query response 0x972c No such name A wpad.mada.ps SOA dns5.mada-alarab.com
  981 58.729650  192.168.1.114   185.17.235.132  DNS    90 Standard query 0x166d A dmd.metaservices.microsoft.com
  984 58.893072  185.17.235.132  192.168.1.114   DNS   349 Standard query response 0x166d A dmd.metaservices.microsoft.com CNAME devicemetadataservice.prod.trafficmanager.net CNAME vmss-prod-wus.westus.cloudapp.azure.com A 138.91.171.81 NS ns11-02.a…
 1014 63.062135  192.168.1.114   185.17.235.132  DNS    76 Standard query 0xaff6 A ocsp.entrust.net
 1015 63.095484  185.17.235.132  192.168.1.114   DNS   335 Standard query response 0xaff6 A ocsp.entrust.net CNAME ocsp.entrust.net.edgekey.net CNAME e6913.dscx.akamaiedge.net A 95.100.206.97 NS n6dscx.akamaiedge.net NS n5dscx.akamaiedge.net NS n2ds…
 1022 63.203619  192.168.1.114   185.17.235.132  DNS    74 Standard query 0x20ec A x1.c.lencr.org
 1024 63.311463  185.17.235.132  192.168.1.114   DNS   347 Standard query response 0x20ec A x1.c.lencr.org CNAME crl.root-x1.letsencrypt.org.edgekey.net CNAME e8652.dscx.akamaiedge.net A 95.100.205.66 NS n1dscx.akamaiedge.net NS n0dscx.akamaiedge.ne…
 1031 63.429588  192.168.1.114   185.17.235.132  DNS    83 Standard query 0xc4ba A ctldl.windowsupdate.com
 1033 63.462715  185.17.235.132  192.168.1.114   DNS   364 Standard query response 0xc4ba A ctldl.windowsupdate.com CNAME wu-bg-shim.trafficmanager.net CNAME wu.azureedge.net CNAME wu.ec.azureedge.net CNAME bg.apr-52dd2-0503.edgecastdns.net CNAME hl…
 1146 80.217713  192.168.1.114   185.17.235.132  DNS    90 Standard query 0xf0f7 A dmd.metaservices.microsoft.com
 1148 80.311706  185.17.235.132  192.168.1.114   DNS   401 Standard query response 0xf0f7 A dmd.metaservices.microsoft.com CNAME devicemetadataservice.prod.trafficmanager.net CNAME vmss-prod-weu.westeurope.cloudapp.azure.com A 52.142.223.178 NS ns11…
 1239 90.671041  192.168.1.114   185.17.235.132  DNS    74 Standard query 0x642e A assets.msn.com
 1240 90.808276  185.17.235.132  192.168.1.114   DNS   404 Standard query response 0x642e A assets.msn.com CNAME assets.msn.com.edgekey.net CNAME e28578.d.akamaiedge.net A 95.101.22.186 A 95.101.22.162 A 95.101.22.146 A 95.101.22.161 A 95.101.22.171…
 1312 95.456160  192.168.1.114   185.17.235.132  DNS    84 Standard query 0x3fe1 A ocws.officeapps.live.com
 1315 95.522235  185.17.235.132  192.168.1.114   DNS   390 Standard query response 0x3fe1 A ocws.officeapps.live.com CNAME prod.ocws1.live.com.akadns.net CNAME europe.ocws1.live.com.akadns.net A 52.109.76.62 NS a11-129.akadns.net NS a1-128.akadns.ne…
 1389 97.034094  192.168.1.114   185.17.235.132  DNS    80 Standard query 0x8ad0 A substrate.office.com
 1405 97.237226  192.168.1.114   185.17.235.134  DNS    80 Standard query 0x8ad0 A substrate.office.com
 1406 97.238528  185.17.235.132  192.168.1.114   DNS   358 Standard query response 0x8ad0 A substrate.office.com CNAME outlook.office365.com CNAME ooc-g2.tm-4.office.com CNAME outlook.ms-acdc.office.com CNAME CDG-efz.ms-acdc.office.com A 40.99.153.1…
```

```
> Frame 895: 75 bytes on wire (600 bits), 75 bytes captured (600 bits) on interface \Device\NPF_{18E4DF4F-1E3C-4DE0-9FFF-7087DE3298D7},
> Ethernet II, Src: IntelCor_88:30:42 (c8:58:c0:88:30:42), Dst: BaudTec_f7:35:22 (94:46:96:f7:35:22)
> Internet Protocol Version 4, Src: 192.168.1.114, Dst: 185.17.235.132
> User Datagram Protocol, Src Port: 52832, Dst Port: 53
> Domain Name System (query)
```

```
0000  94 46 96 f7 35 22 c8 58  c0 88 30 42 08 00 45 00   ·F··5"·X ··0B··E·
0010  00 3d 10 33 00 00 80 11  00 00 c0 a8 01 72 b9 11   ·=·3···· ·····r··
0020  eb 84 ce 60 00 35 00 29  66 eb 6c e1 01 00 00 01   ···`·5·) f·l·····
0030  00 00 00 00 00 00 03 77  77 77 07 63 6f 72 6e 65   ·······w ww·corne
0040  6c 6c 03 65 64 75 00 00  01 00 01                  ll·edu·· ···
```

*WiFi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
dns
Domain Name System: Protocol
Packets: 1504 · Displayed: 67 (4.5%)
Profile: Default
13°C
גשם הלילה
Search
ENG UK
16:23
22/12/2023

From the ping results, do you think the response you have got is from the USA? Explain your answer briefly.

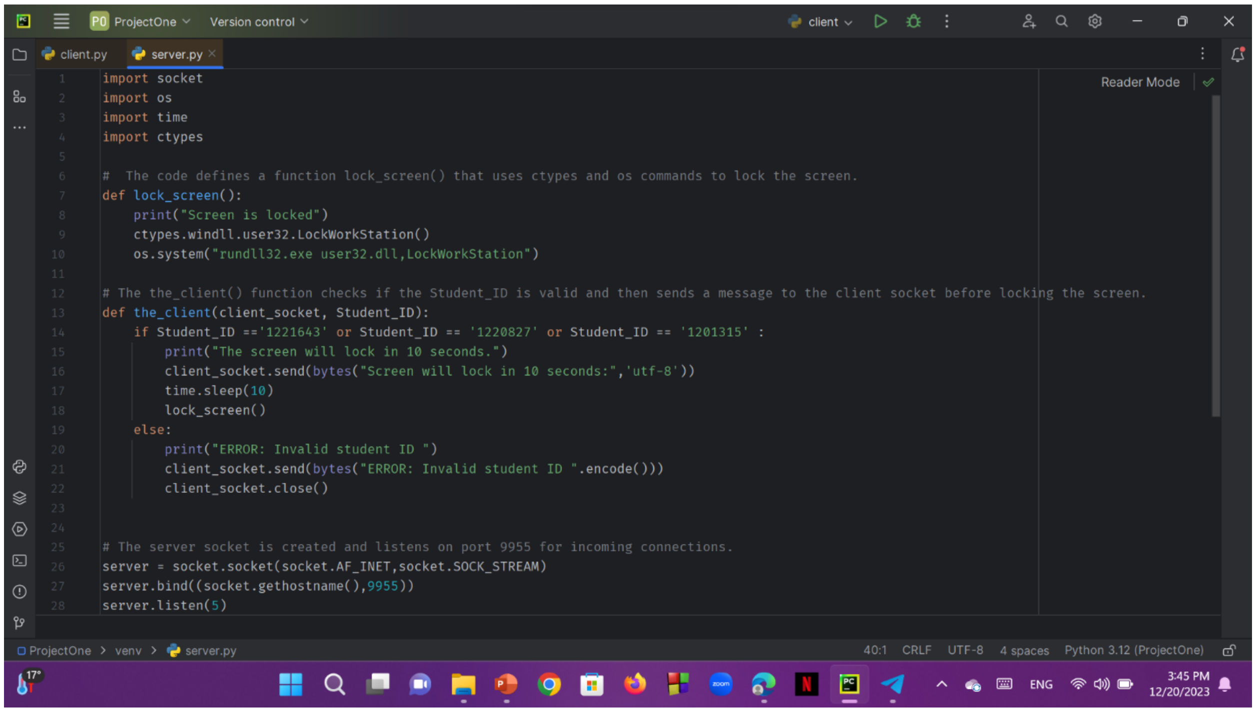
Yes from the information, the IP address is 13.107.246.62 Cornell University, then the response obtained from a ping is likely from the United States, as Cornell University is located in the United States. The IP address is a key identifier for networked devices.

Part 2

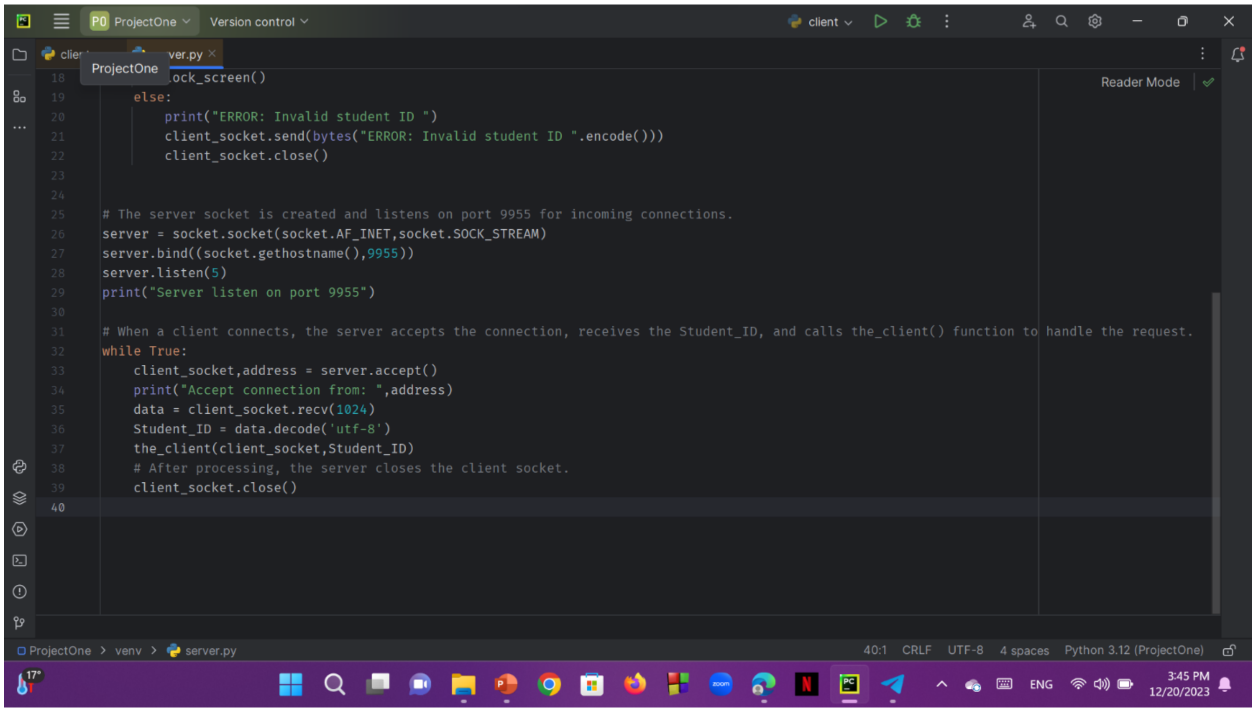
First, we need a server socket and a client socket to send and receive message between both.

We will start with server socket:

The server socket

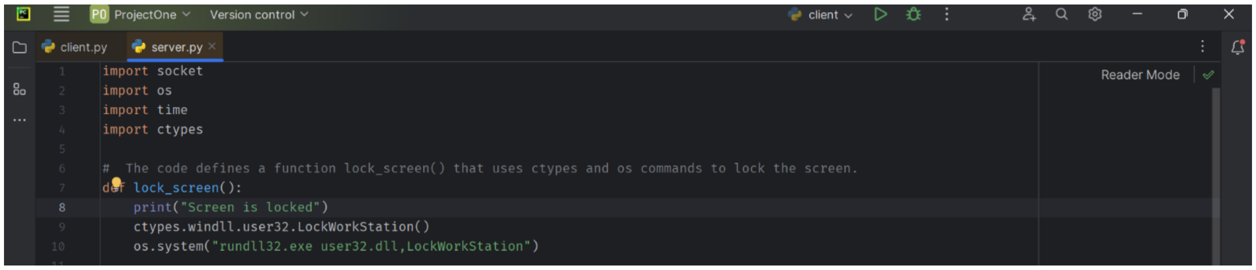


```python
import socket
import os
import time
import ctypes

#  The code defines a function lock_screen() that uses ctypes and os commands to lock the screen.
def lock_screen():
    print("Screen is locked")
    ctypes.windll.user32.LockWorkStation()
    os.system("rundll32.exe user32.dll,LockWorkStation")

# The the_client() function checks if the Student_ID is valid and then sends a message to the client socket before locking the screen.
def the_client(client_socket, Student_ID):
    if Student_ID =='1221643' or Student_ID == '1220827' or Student_ID == '1201315' :
        print("The screen will lock in 10 seconds.")
        client_socket.send(bytes("Screen will lock in 10 seconds:",'utf-8'))
        time.sleep(10)
        lock_screen()
    else:
        print("ERROR: Invalid student ID ")
        client_socket.send(bytes("ERROR: Invalid student ID ".encode()))
        client_socket.close()


# The server socket is created and listens on port 9955 for incoming connections.
server = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
server.bind((socket.gethostname(),9955))
server.listen(5)
```



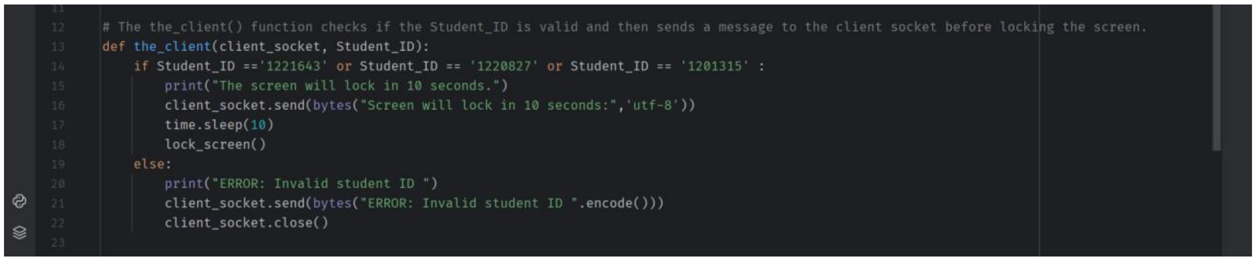
```python
        lock_screen()
    else:
        print("ERROR: Invalid student ID ")
        client_socket.send(bytes("ERROR: Invalid student ID ".encode()))
        client_socket.close()


# The server socket is created and listens on port 9955 for incoming connections.
server = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
server.bind((socket.gethostname(),9955))
server.listen(5)
print("Server listen on port 9955")

# When a client connects, the server accepts the connection, receives the Student_ID, and calls the_client() function to handle the request.
while True:
    client_socket,address = server.accept()
    print("Accept connection from: ",address)
    data = client_socket.recv(1024)
    Student_ID = data.decode('utf-8')
    the_client(client_socket,Student_ID)
    # After processing, the server closes the client socket.
    client_socket.close()
```

We create a lock_screen () function it is defined toto lock the screen using ctypes and os commands. It prints a message and then locks the screen using the LockWorkStation () function from the user32 library.



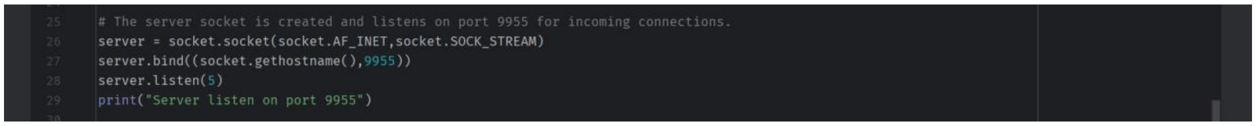
```python
import socket
import os
import time
import ctypes

#  The code defines a function lock_screen() that uses ctypes and os commands to lock the screen.
def lock_screen():
    print("Screen is locked")
    ctypes.windll.user32.LockWorkStation()
    os.system("rundll32.exe user32.dll,LockWorkStation")
```

the_client () function takes i a Student_ID as parameters. It checks if the Student_ID is valid and sends a message to the client socket before locking the screen if the ID is valid. If the ID is invalid, it sends an error message to the client socket and closes the connection.



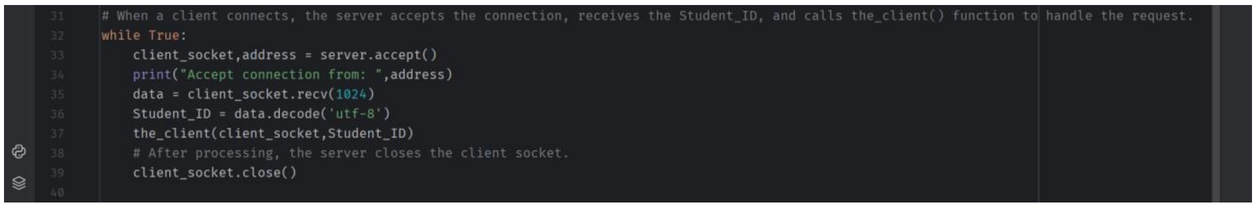
```python
# The the_client() function checks if the Student_ID is valid and then sends a message to the client socket before locking the screen.
def the_client(client_socket, Student_ID):
    if Student_ID =='1221643' or Student_ID == '1220827' or Student_ID == '1201315' :
        print("The screen will lock in 10 seconds.")
        client_socket.send(bytes("Screen will lock in 10 seconds:",'utf-8'))
        time.sleep(10)
        lock_screen()
    else:
        print("ERROR: Invalid student ID ")
        client_socket.send(bytes("ERROR: Invalid student ID ".encode()))
        client_socket.close()
```

Server socket is created using socket.socket(), specifying AF_INET as the address and SOCK_STREAM as the socket type. The server binds to the hostname on port 9955 and listens for incoming connections with a backlog of 5.



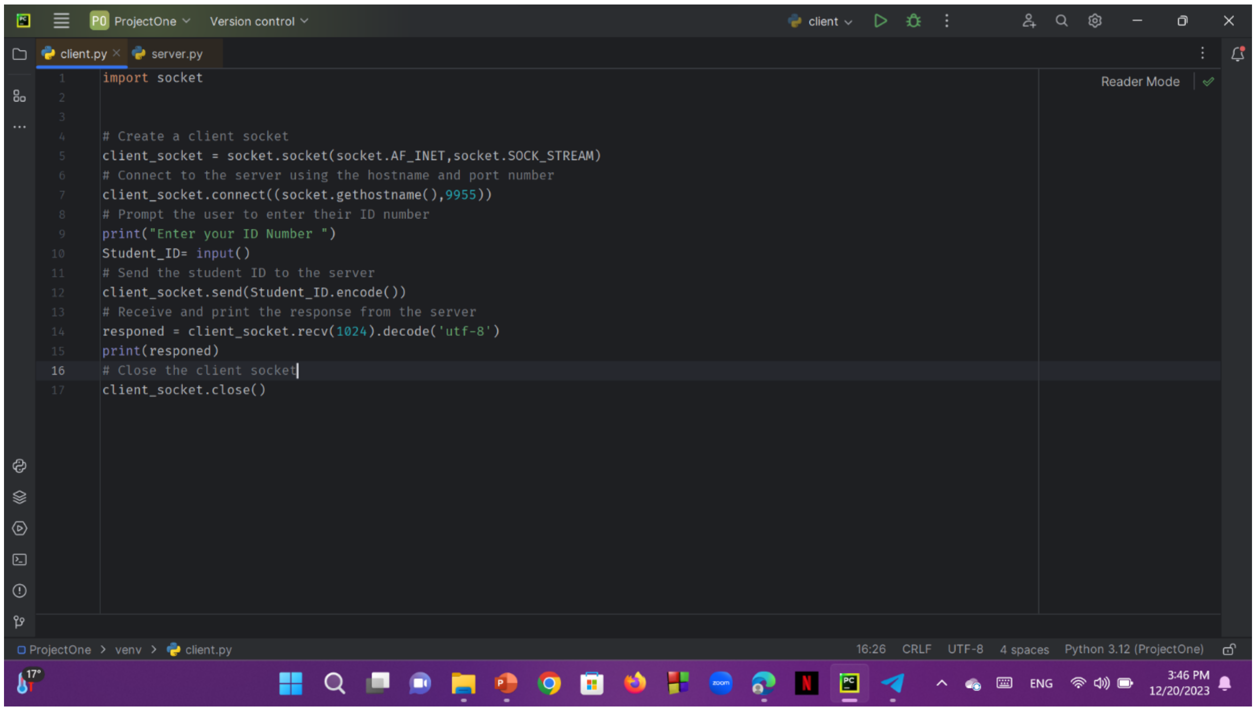
```python
# The server socket is created and listens on port 9955 for incoming connections.
server = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
server.bind((socket.gethostname(),9955))
server.listen(5)
print("Server listen on port 9955")
```

In infinite loop, the server accepts incoming connections from clients using accept(). When a client connects, it receives data (the Student_ID) from the client socket, decodes it, and calls the_client() function to handle the request.



```python
# When a client connects, the server accepts the connection, receives the Student_ID, and calls the_client() function to handle the request.
while True:
    client_socket,address = server.accept()
    print("Accept connection from: ",address)
    data = client_socket.recv(1024)
    Student_ID = data.decode('utf-8')
    the_client(client_socket,Student_ID)
    # After processing, the server closes the client socket.
    client_socket.close()
```

Client socket:


ProjectOne
Version control
client.py
server.py
client
Reader Mode
ProjectOne > venv > client.py
16:26 CRLF UTF-8 4 spaces Python 3.12 (ProjectOne)
3:46 PM
12/20/2023

```python
import socket


# Create a client socket
client_socket = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
# Connect to the server using the hostname and port number
client_socket.connect((socket.gethostname(),9955))
# Prompt the user to enter their ID number
print("Enter your ID Number ")
Student_ID= input()
# Send the student ID to the server
client_socket.send(Student_ID.encode())
# Receive and print the response from the server
responed = client_socket.recv(1024).decode('utf-8')
print(responed)
# Close the client socket
client_socket.close()
```

Create a client socket using the socket.socket() function, specifying the address (AF_INET for IPv4) and the socket type (SOCK_STREAM for TCP).

Connect to the server using the connect() method of the client_socket object, providing the hostname and port number as a tuple.

Prompt the user to enter their ID number .

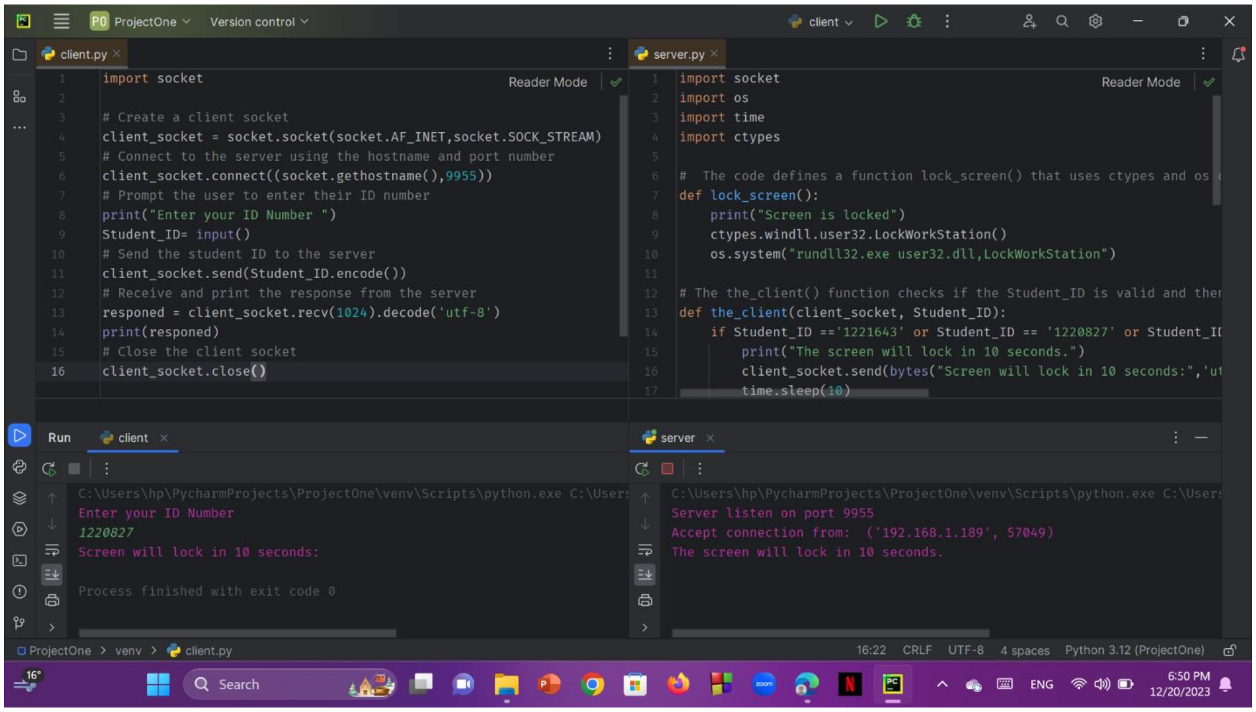
Read the user's input for their ID number and store it in a variable called Student_ID.

Send the student ID to the server using the send() method of the client_socket object, encoding it into bytes using encode().

Receive a response from the server using the recv() method of client_socket, specifying a buffer size of 1024 bytes, and decode it from bytes to a string using decode('utf-8').

Print out the response from the server .

The output in console :



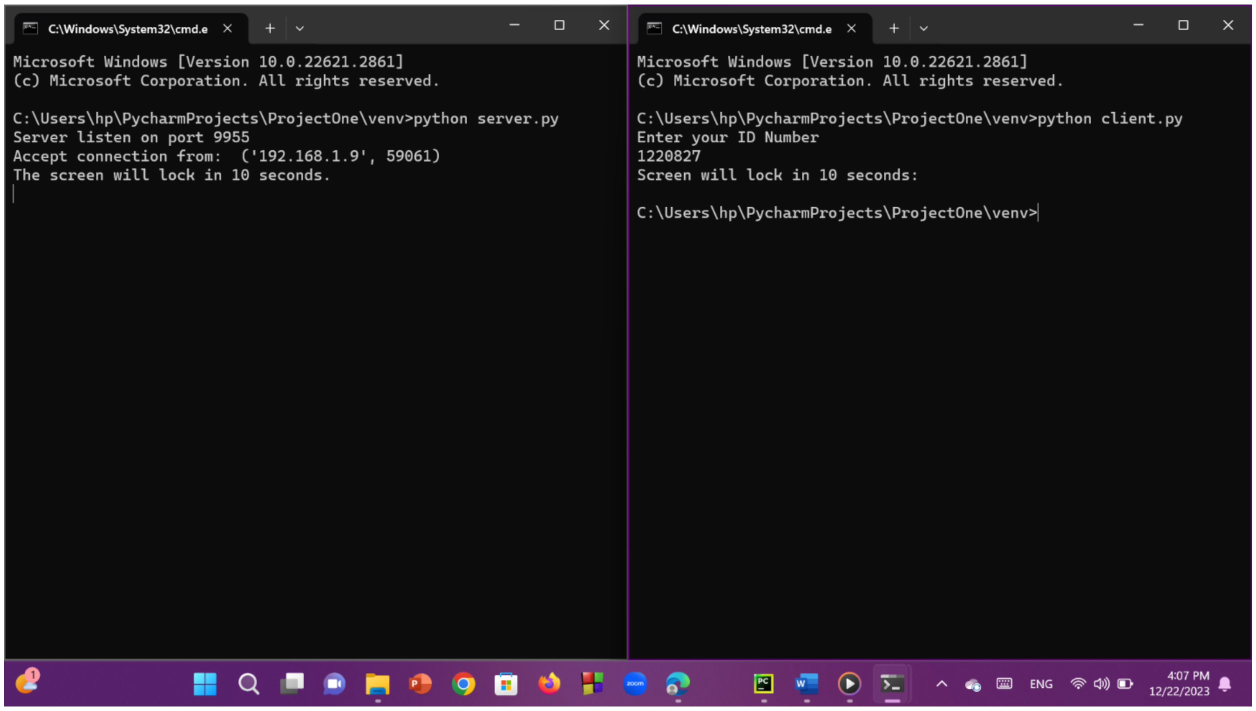
```python
import socket

# Create a client socket
client_socket = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
# Connect to the server using the hostname and port number
client_socket.connect((socket.gethostname(),9955))
# Prompt the user to enter their ID number
print("Enter your ID Number ")
Student_ID= input()
# Send the student ID to the server
client_socket.send(Student_ID.encode())
# Receive and print the response from the server
responed = client_socket.recv(1024).decode('utf-8')
print(responed)
# Close the client socket
client_socket.close()
```

```python
import socket
import os
import time
import ctypes

#  The code defines a function lock_screen() that uses ctypes and os
def lock_screen():
    print("Screen is locked")
    ctypes.windll.user32.LockWorkStation()
    os.system("rundll32.exe user32.dll,LockWorkStation")

# The the_client() function checks if the Student_ID is valid and ther
def the_client(client_socket, Student_ID):
    if Student_ID =='1221643' or Student_ID == '1220827' or Student_II
        print("The screen will lock in 10 seconds.")
        client_socket.send(bytes("Screen will lock in 10 seconds:",'ut
        time.sleep(10)
```

```
C:\Users\hp\PycharmProjects\ProjectOne\venv\Scripts\python.exe C:\Users
Enter your ID Number
1220827
Screen will lock in 10 seconds:

Process finished with exit code 0
```

```
C:\Users\hp\PycharmProjects\ProjectOne\venv\Scripts\python.exe C:\Users
Server listen on port 9955
Accept connection from:  ('192.168.1.189', 57049)
The screen will lock in 10 seconds.
```

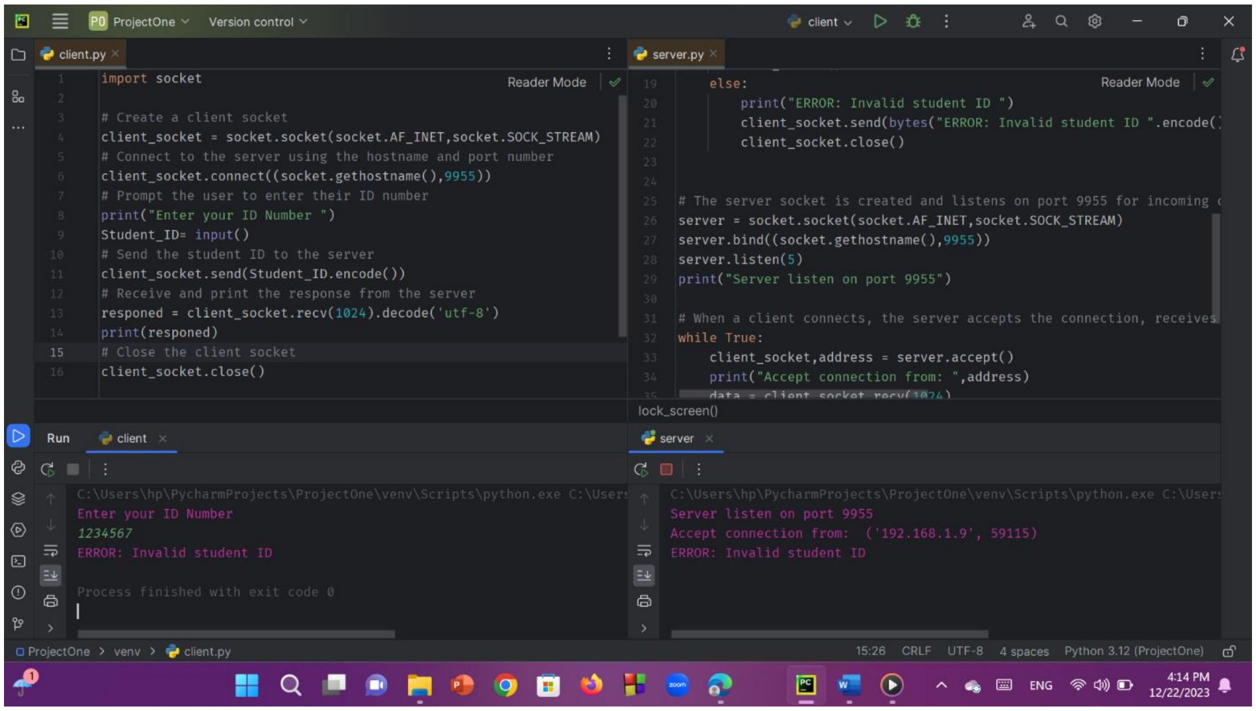
The output in command :



```
Microsoft Windows [Version 10.0.22621.2861]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp\PycharmProjects\ProjectOne\venv>python server.py
Server listen on port 9955
Accept connection from:  ('192.168.1.9', 59061)
The screen will lock in 10 seconds.
```

```
Microsoft Windows [Version 10.0.22621.2861]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp\PycharmProjects\ProjectOne\venv>python client.py
Enter your ID Number
1220827
Screen will lock in 10 seconds:

C:\Users\hp\PycharmProjects\ProjectOne\venv>
```

The output when the user enter Invalid student id in console:



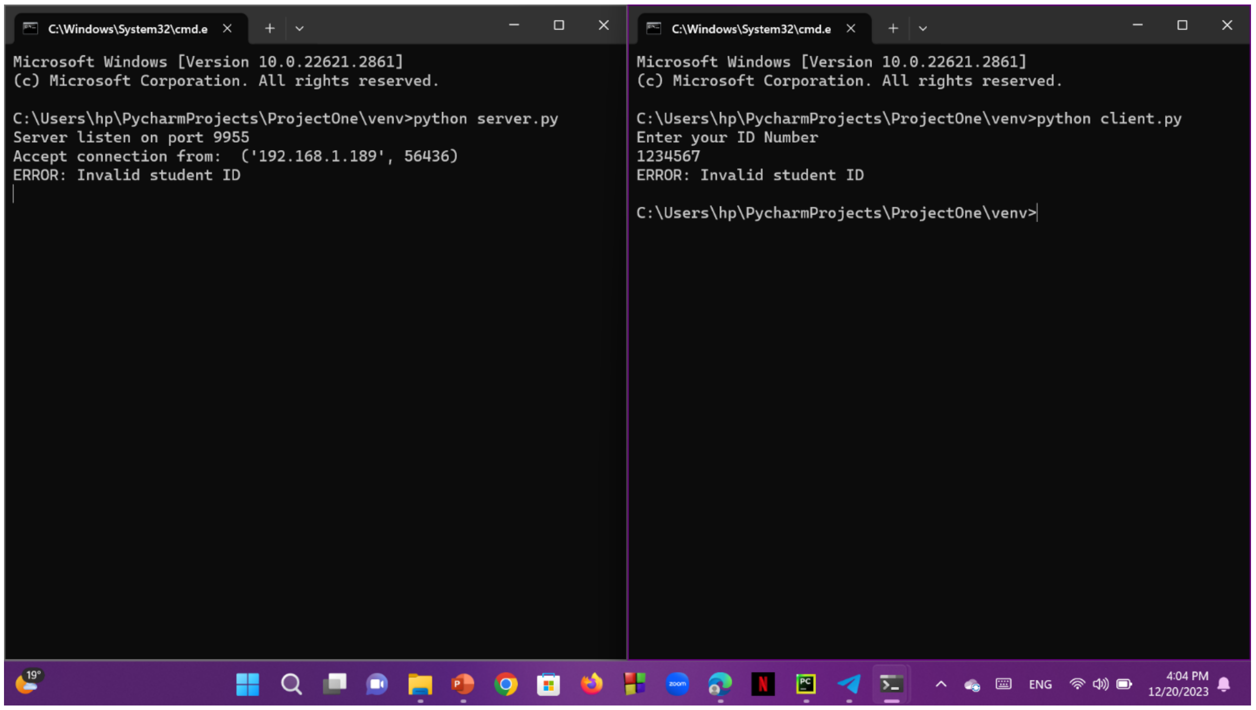
```
import socket

# Create a client socket
client_socket = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
# Connect to the server using the hostname and port number
client_socket.connect((socket.gethostname(),9955))
# Prompt the user to enter their ID number
print("Enter your ID Number ")
Student_ID= input()
# Send the student ID to the server
client_socket.send(Student_ID.encode())
# Receive and print the response from the server
responed = client_socket.recv(1024).decode('utf-8')
print(responed)
# Close the client socket
client_socket.close()
```

```
    else:
        print("ERROR: Invalid student ID ")
        client_socket.send(bytes("ERROR: Invalid student ID ".encode(
        client_socket.close()


# The server socket is created and listens on port 9955 for incoming
server = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
server.bind((socket.gethostname(),9955))
server.listen(5)
print("Server listen on port 9955")

# When a client connects, the server accepts the connection, receives
while True:
    client_socket,address = server.accept()
    print("Accept connection from: ",address)
```

```
C:\Users\hp\PycharmProjects\ProjectOne\venv\Scripts\python.exe C:\Users
Enter your ID Number
1234567
ERROR: Invalid student ID

Process finished with exit code 0
```

```
C:\Users\hp\PycharmProjects\ProjectOne\venv\Scripts\python.exe C:\Users
Server listen on port 9955
Accept connection from:  ('192.168.1.9', 59115)
ERROR: Invalid student ID
```

The output when the user enter Invalid student id in command:



```
Microsoft Windows [Version 10.0.22621.2861]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp\PycharmProjects\ProjectOne\venv>python server.py
Server listen on port 9955
Accept connection from:  ('192.168.1.189', 56436)
ERROR: Invalid student ID
```

```
Microsoft Windows [Version 10.0.22621.2861]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp\PycharmProjects\ProjectOne\venv>python client.py
Enter your ID Number
1234567
ERROR: Invalid student ID

C:\Users\hp\PycharmProjects\ProjectOne\venv>
```

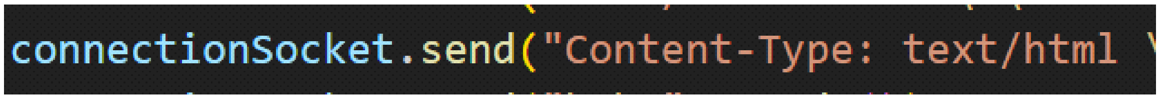
Part 3:

1.

Interpretation of Data: The Content-Type header is crucial for the server to interpret the incoming data correctly. Different types of data require different processing. For instance, HTML content is rendered differently from JSON data.

Error Handling: If the server receives data that it cannot interpret or process, it can return an appropriate error response (e.g., 415 Unsupported Media Type) based on the Content-Type header.

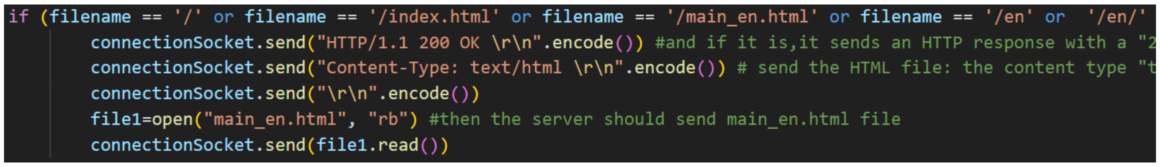
In summary, the Content-Type header is essential for proper communication between the client and server by specifying the type of data being sent or received, enabling correct interpretation and processing of the payload.



```
connectionSocket.send("Content-Type: text/html \
```

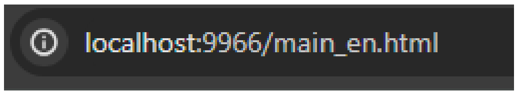
2.

if the request is / or /index.html or /main_en.html or /en (for example localhost:9966/ or localhost:9966/en) then the server should send main_en.html file with Content-Type: text/html.



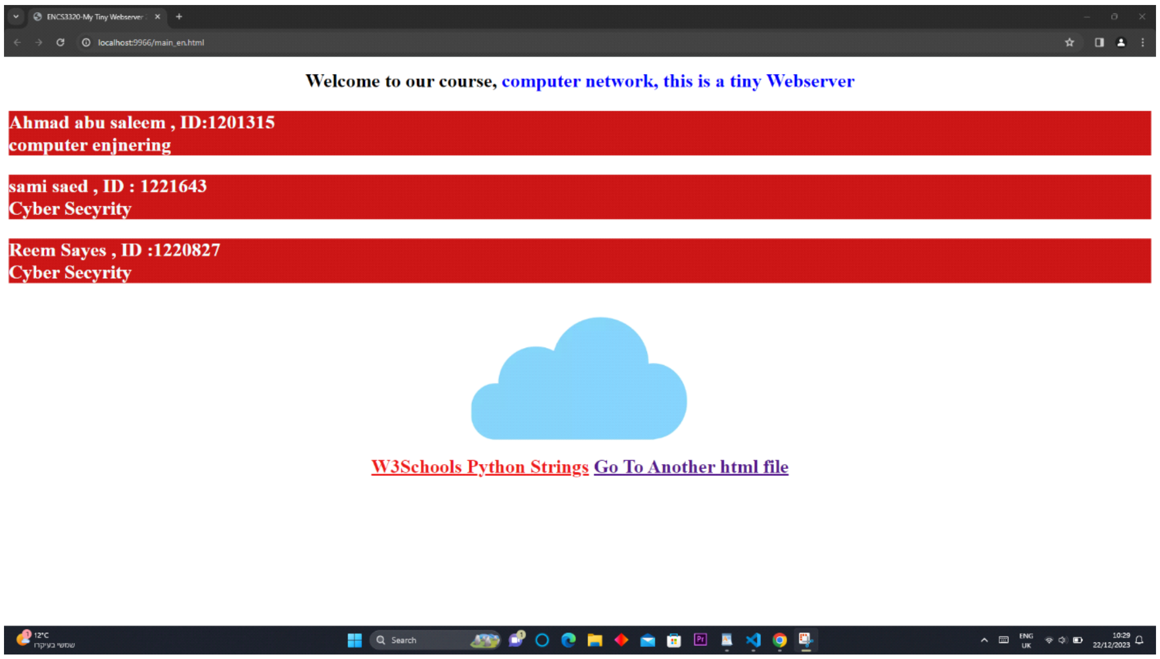
```
if (filename == '/' or filename == '/index.html' or filename == '/main_en.html' or filename == '/en' or  '/en/'
        connectionSocket.send("HTTP/1.1 200 OK \r\n".encode()) #and if it is,it sends an HTTP response with a "2
        connectionSocket.send("Content-Type: text/html \r\n".encode()) # send the HTML file: the content type "t
        connectionSocket.send("\r\n".encode())
        file1=open("main_en.html", "rb") #then the server should send main_en.html file
        connectionSocket.send(file1.read())
```

the result of the if statement is that when you search for("index.html", "main_en.html", "en") will open the main page in ("English).



localhost:9966/main_en.html

3.this is the main HTML file in ("English") and it has student info about the student who worked on the project the page has a link to another HTML file and a link to ("https://www.w3schools.com/python/python_strings.asp").



Welcome to our course, computer network, this is a tiny Webserver

Ahmad abu saleem , ID:1201315
computer enjnering

sami saed , ID : 1221643
Cyber Secyrity

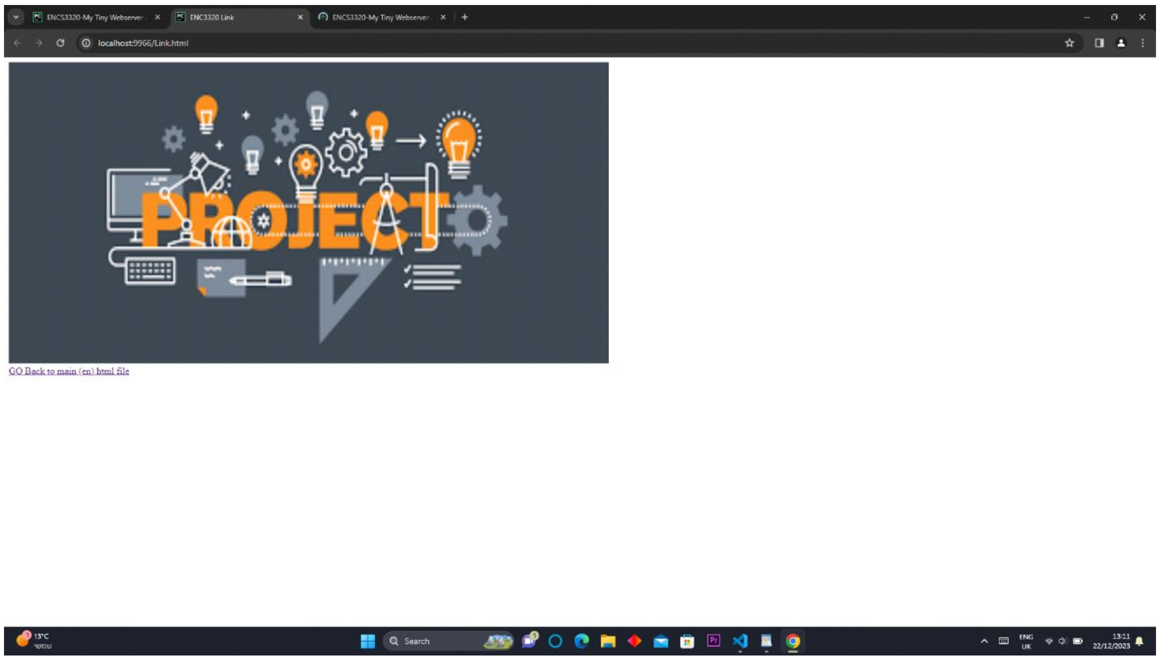
Reem Sayes , ID :1220827
Cyber Secyrity

W3Schools Python Strings Go To Another html file

the link to another HTML file :

Go To Another html file

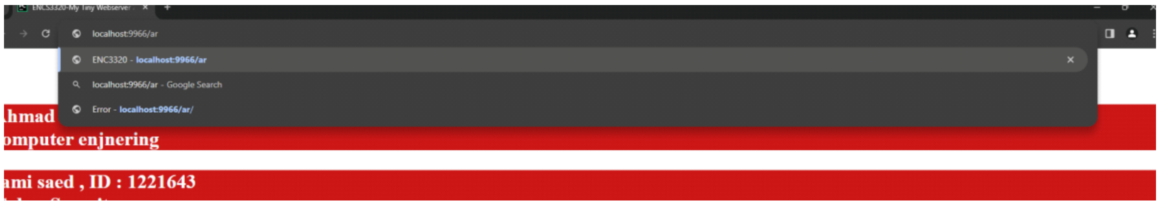
will open a new window of another HTML file :-


localhost:9966/Link.html

GO Back to main (en) html file

4.

when you search for ("/ar"):


localhost:9966/ar

ENC3320 - localhost:9966/ar

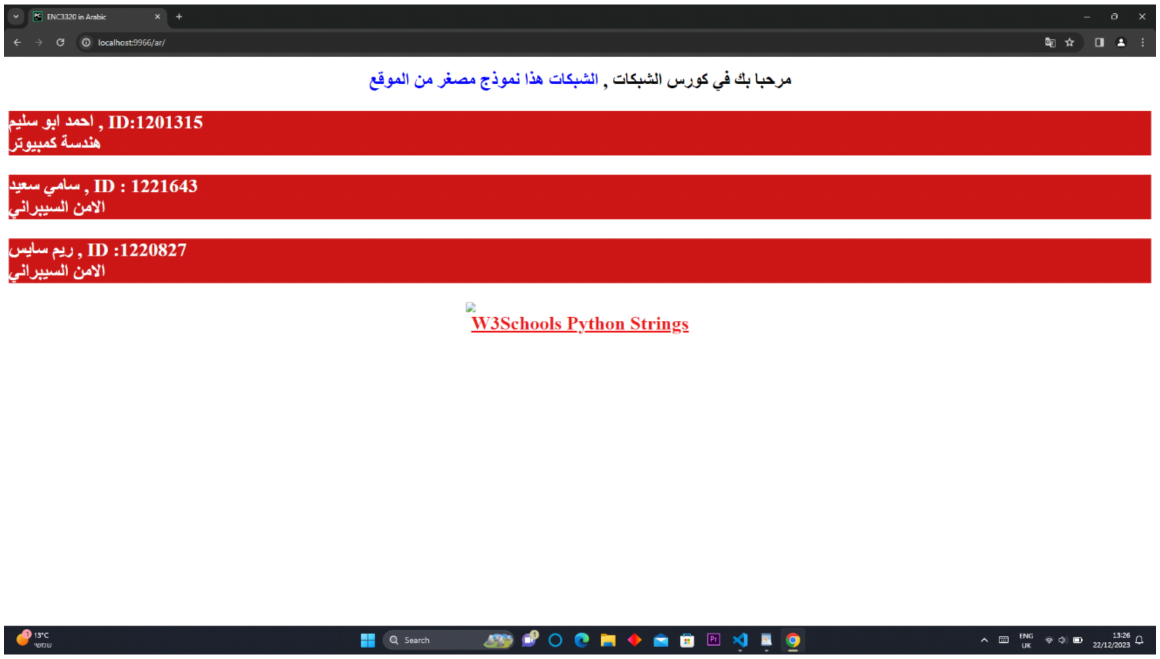
localhost:9966/ar - Google Search

Error - localhost:9966/ar/

hmad
omputer enjnering

ami saed , ID : 1221643

then this page will appear:



مرحبا بك في كورس الشبكات , الشبكات هذا نموذج مصغر من الموقع

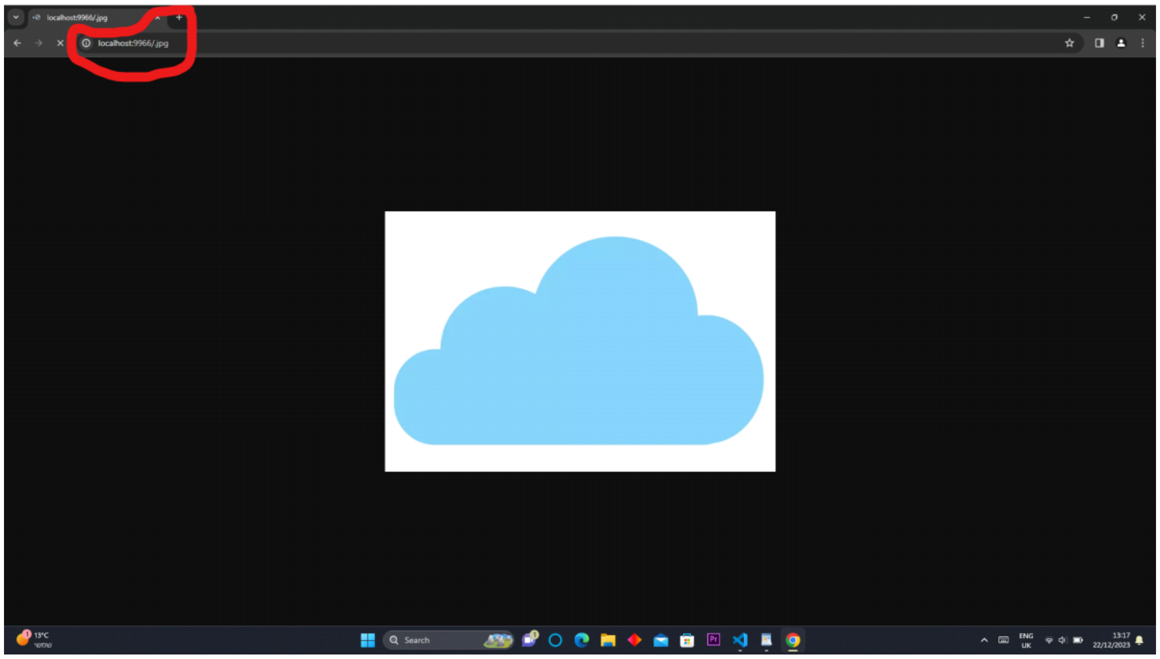
احمد ابو سليم , ID:1201315
هندسة كمبيوتر

سامي سعيد , ID : 1221643
الامن السيبراني

ريم سايس , ID :1220827
الامن السيبراني

W3Schools Python Strings

when you search for (".jpg"):



note: these are examples and the same thing when you search for. Png,css,html file.

5.

Use the status code 307 Temporary Redirect :)

Search for the URL links (/cr,/so,/rt):

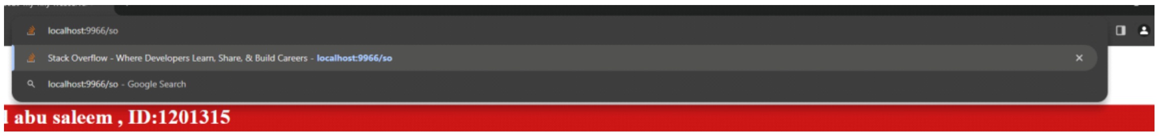
/cr for cornell.edu website.

/so for stackoverflow.com.

/rt for ritaj.com.

some examples:-

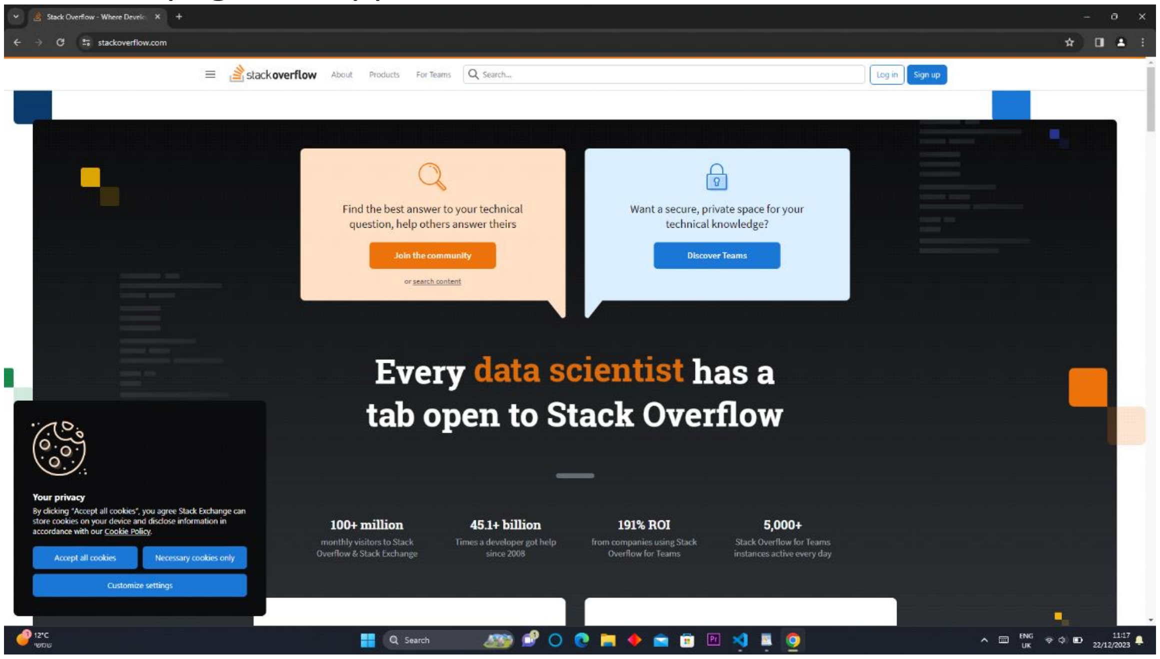
you search for /so and then you will be redirected to Stackoverflow.com



```
localhost:9966/so
Stack Overflow - Where Developers Learn, Share, & Build Careers - localhost:9966/so
localhost:9966/so - Google Search
```

l abu saleem , ID:1201315

then this page will appear:-



Find the best answer to your technical question, help others answer theirs

Want a secure, private space for your technical knowledge?

Every data scientist has a tab open to Stack Overflow

100+ million monthly visitors to Stack Overflow & Stack Exchange

45.1+ billion Times a developer got help since 2008

191% ROI from companies using Stack Overflow for Teams

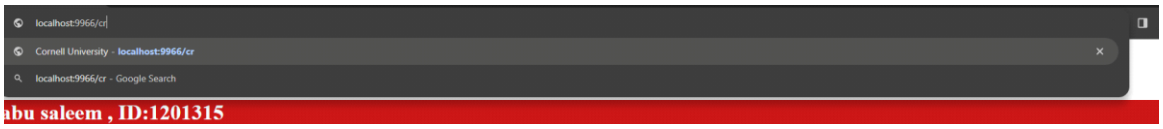
5,000+ Stack Overflow for Teams instances active every day

Your privacy

By clicking "Accept all cookies", you agree Stack Exchange can store cookies on your device and disclose information in accordance with our Cookie Policy.

another example:-

if you search for /cr then you will be redirected to cornell.edu :

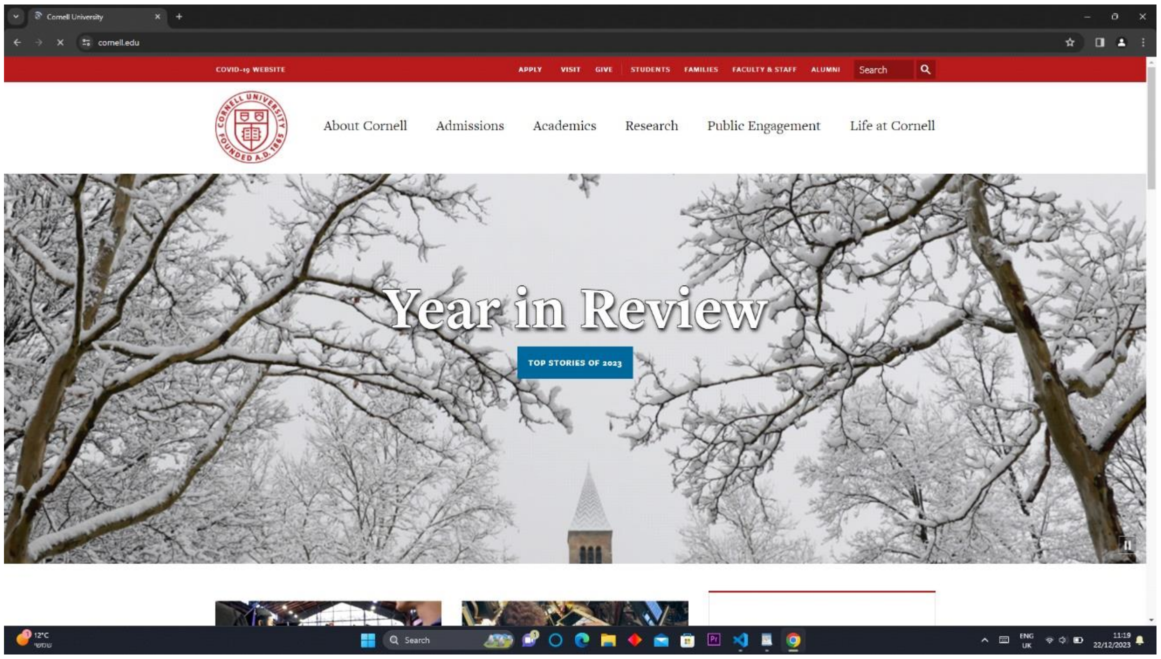

localhost:9966/cr

Cornell University - localhost:9966/cr

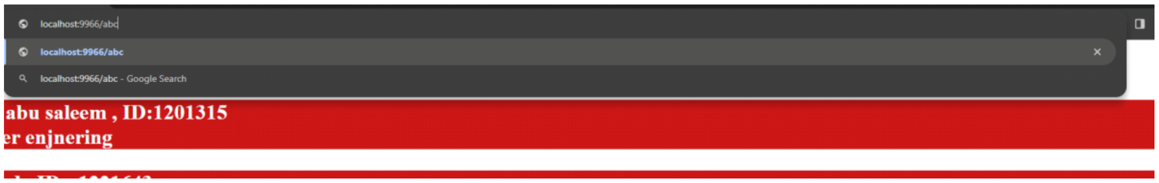
localhost:9966/cr - Google Search

abu saleem , ID:1201315

then this page will appear:



6. The Error 404 will appear when you enter the wrong URL address:

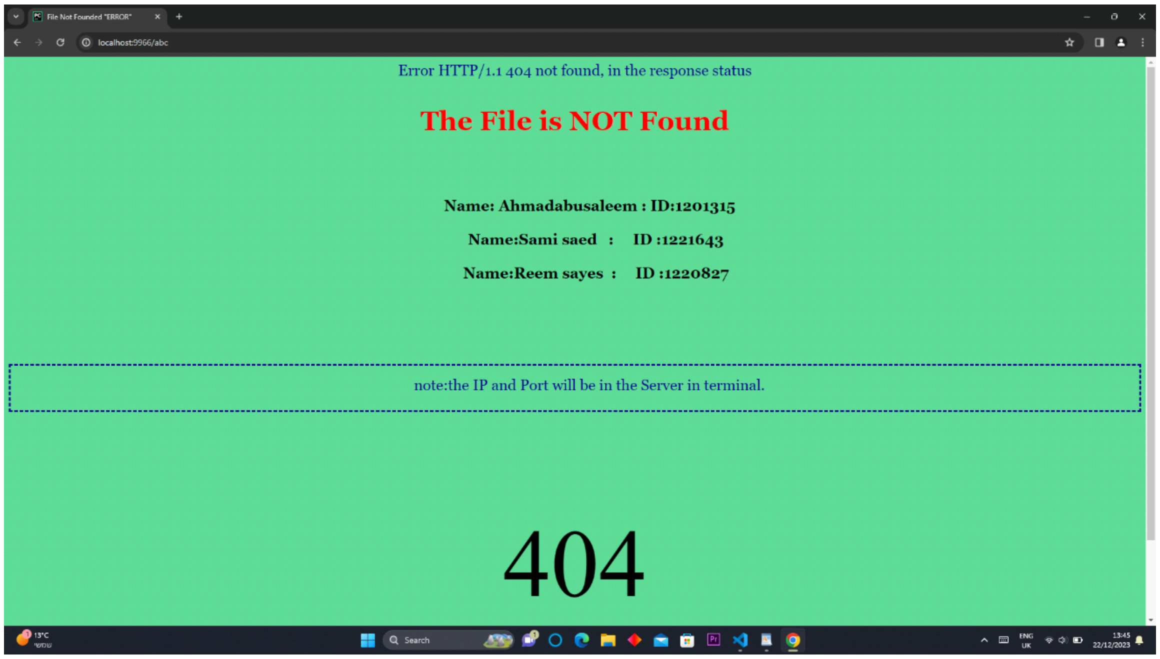

localhost:9966/abc

localhost:9966/abc

localhost:9966/abc - Google Search

abu saleem , ID:1201315
er enjnering

then this page will appear for you:



Error HTTP/1.1 404 not found, in the response status

The File is NOT Found

Name: Ahmadabusaleem : ID:1201315

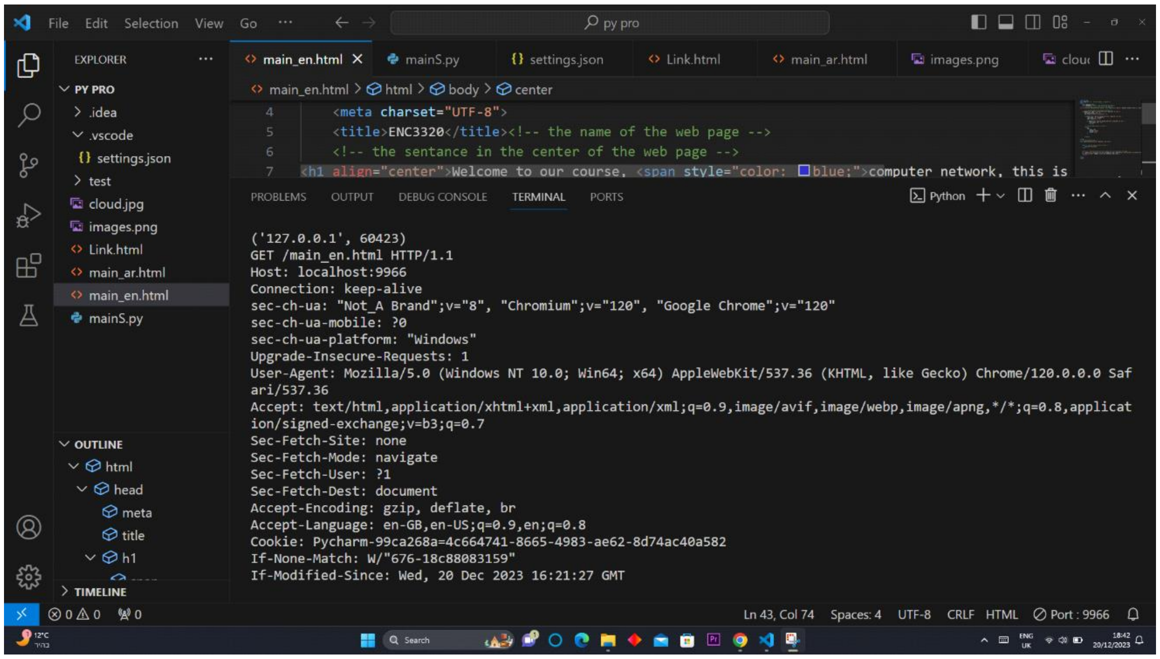
Name:Sami saed : ID :1221643

Name:Reem sayes : ID :1220827

note:the IP and Port will be in the Server in terminal.

404

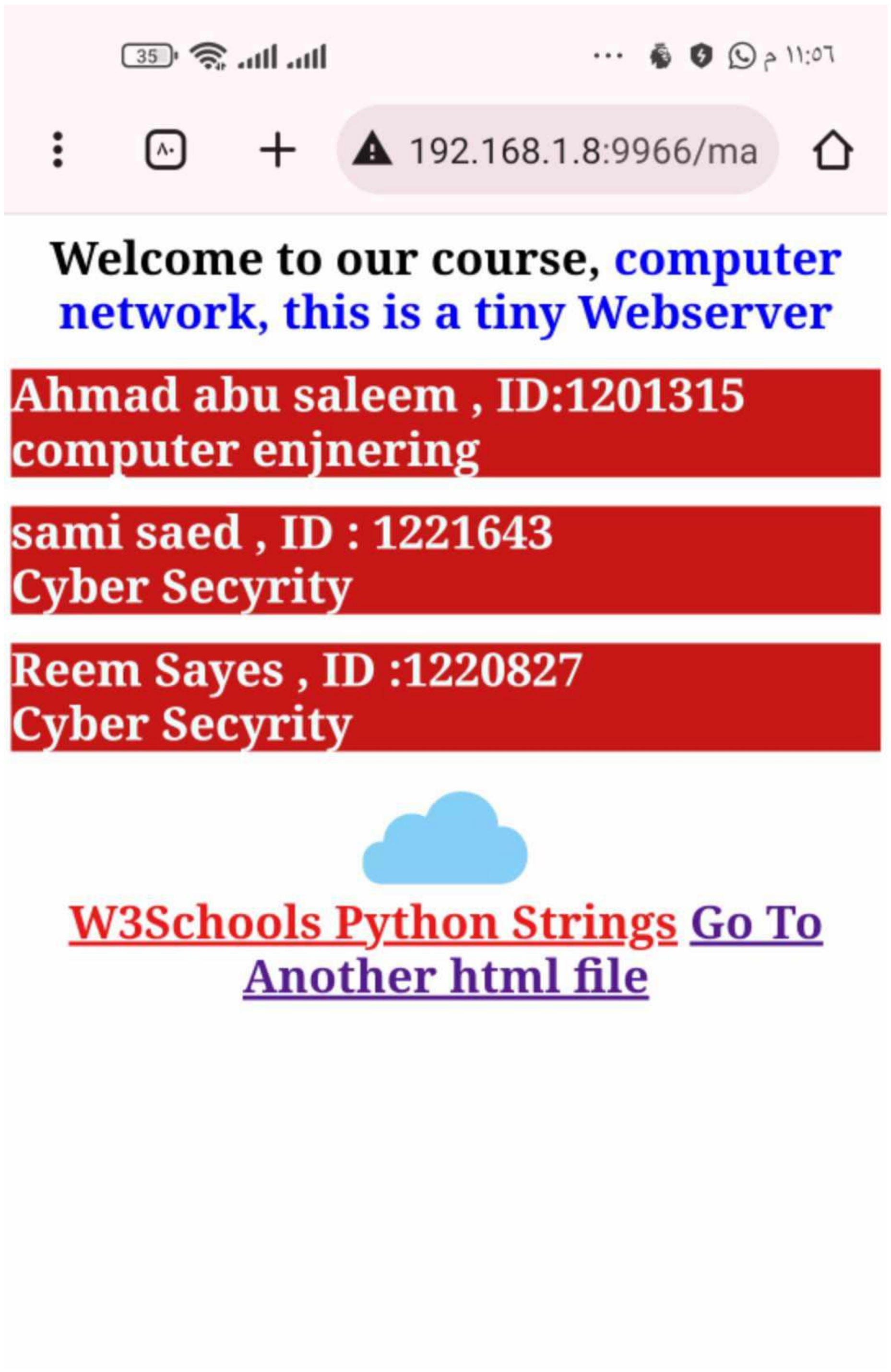
7. in the terminal window the HTTP request will appear :



```
<meta charset="UTF-8">
<title>ENC3320</title><!-- the name of the web page -->
<!-- the sentance in the center of the web page -->
<h1 align="center">Welcome to our course, <span style="color: blue;">computer network, this is
```

```
('127.0.0.1', 60423)
GET /main_en.html HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Google Chrome";v="120"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Saf
ari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,applicat
ion/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Cookie: Pycharm-99ca268a=4c664741-8665-4983-ae62-8d74ac40a582
If-None-Match: W/"676-18c88083159"
If-Modified-Since: Wed, 20 Dec 2023 16:21:27 GMT
```

Screenshot from another device: for link.html, file not found, main_en.html, and main_ar.html



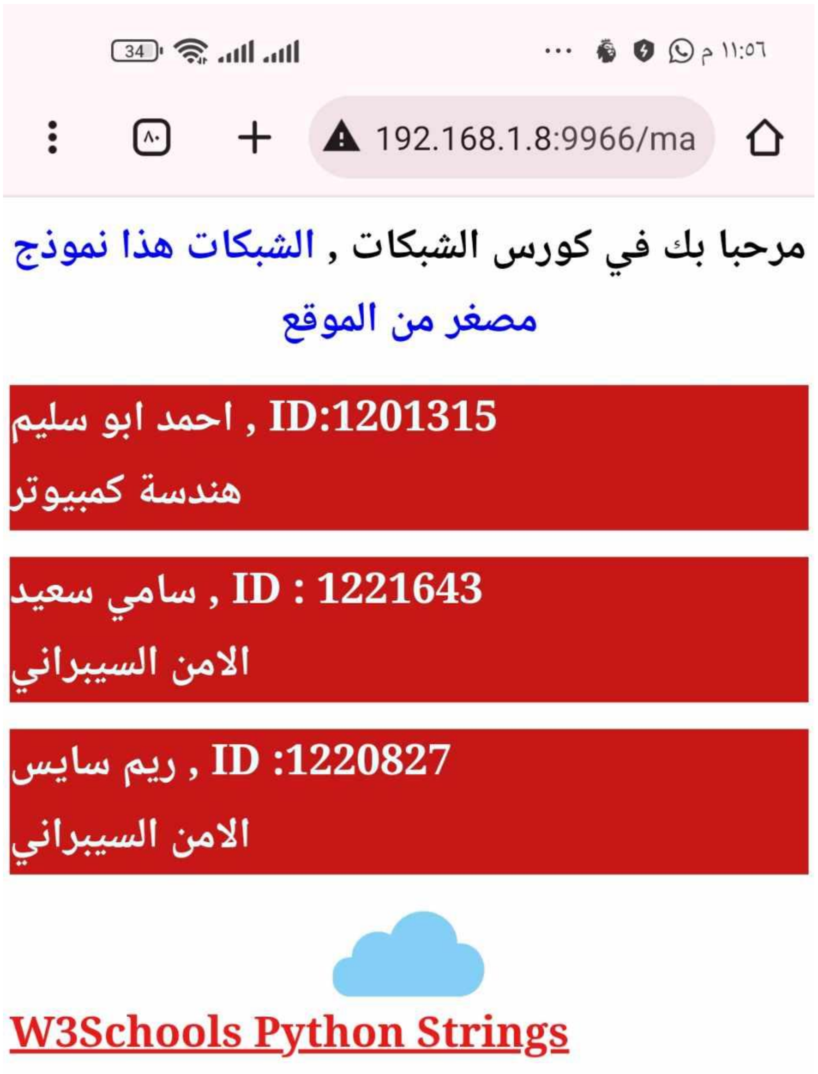
Welcome to our course, computer network, this is a tiny Webserver

Ahmad abu saleem , ID:1201315
computer enjnering

sami saed , ID : 1221643
Cyber Secyrity

Reem Sayes , ID :1220827
Cyber Secyrity

W3Schools Python Strings Go To Another html file



١١:٥٦ م

34

192.168.1.8:9966/ma

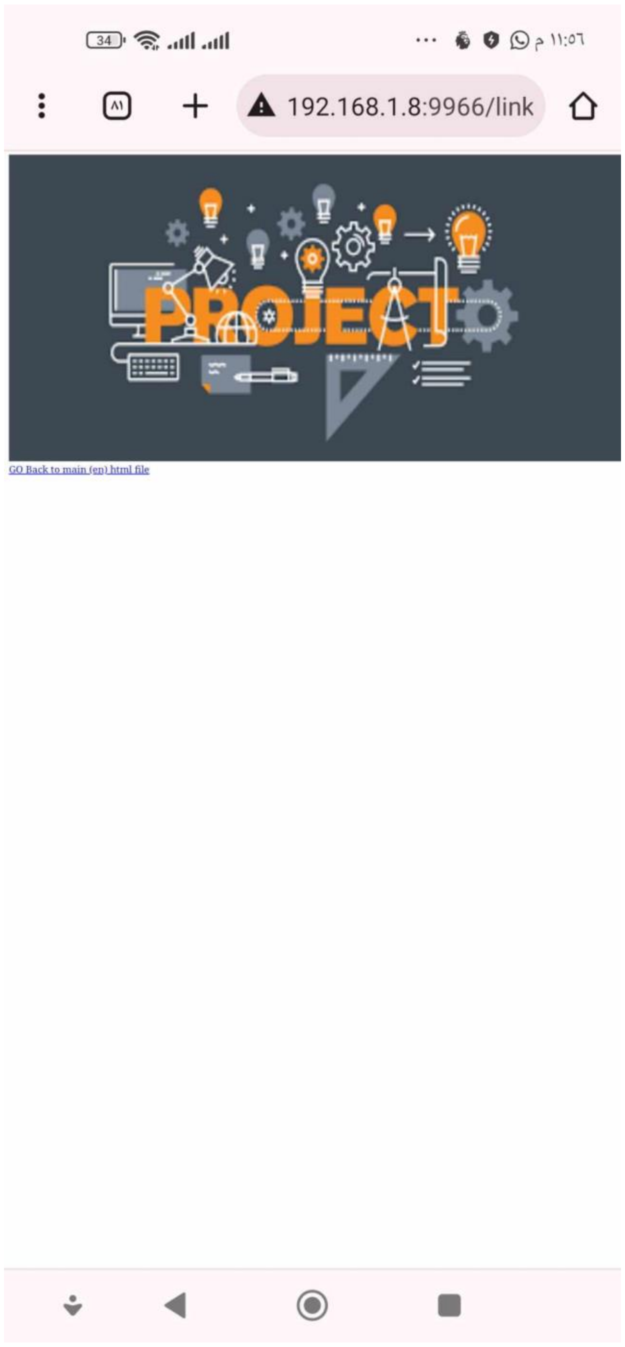
مرحبا بك في كورس الشبكات , الشبكات هذا نموذج مصغر من الموقع

احمد ابو سليم , ID:1201315
هندسة كمبيوتر

سامي سعيد , ID : 1221643
الامن السيبراني

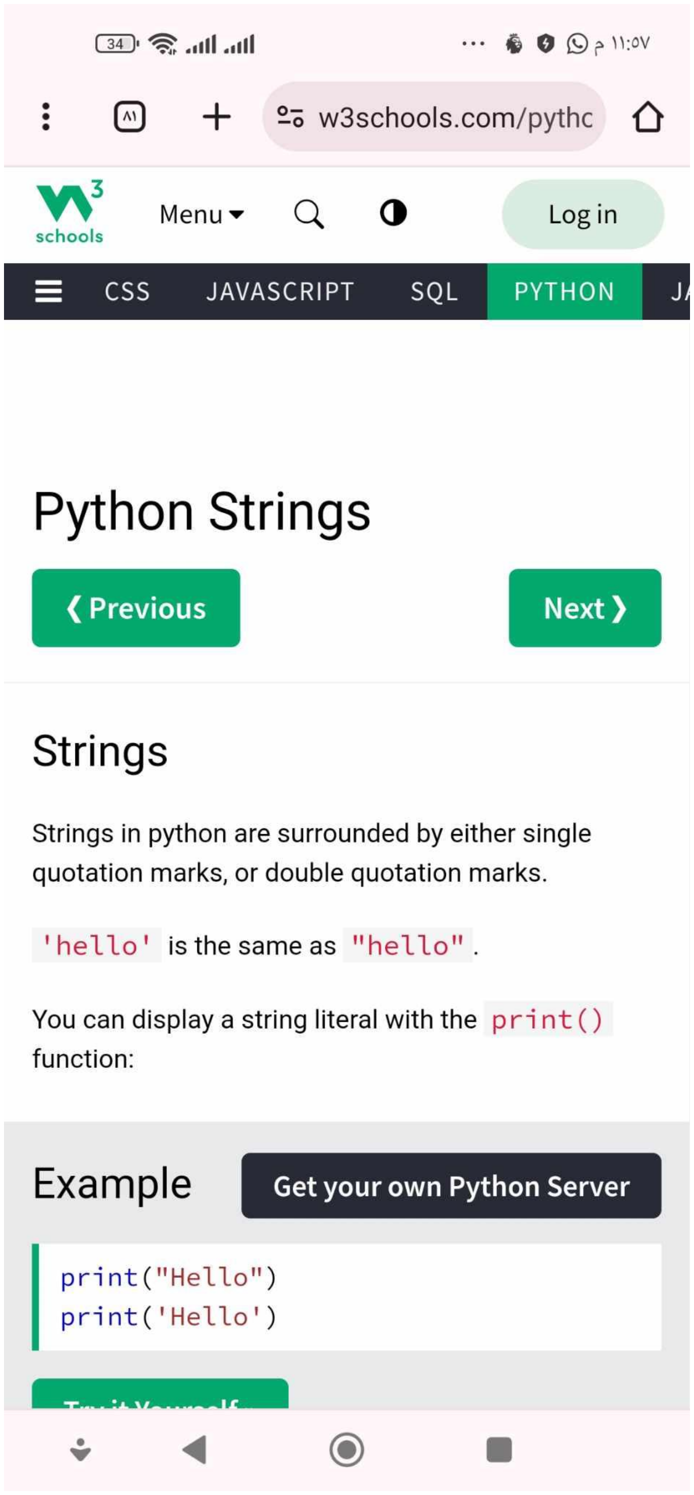
ريم سايس , ID :1220827
الامن السيبراني

W3Schools Python Strings



192.168.1.8:9966/link

GO Back to main (en) html file



Python Strings

Strings

Strings in python are surrounded by either single quotation marks, or double quotation marks.

`'hello'` is the same as `"hello"`.

You can display a string literal with the `print()` function:

Example

```
print("Hello")
print('Hello')
```