Department of Electrical and

Computer Engineering

Computer Networks ENCS3320

Project#1 Due: 08/12/2023

Student #1:Majed Alghoul 1202940

Student #2:Mariam Abukhdear 1222273

Student #3:Yara Obaid 1212482

Section NO. 4

Instructor : Dr. Imad Tartir

Table Of Contents

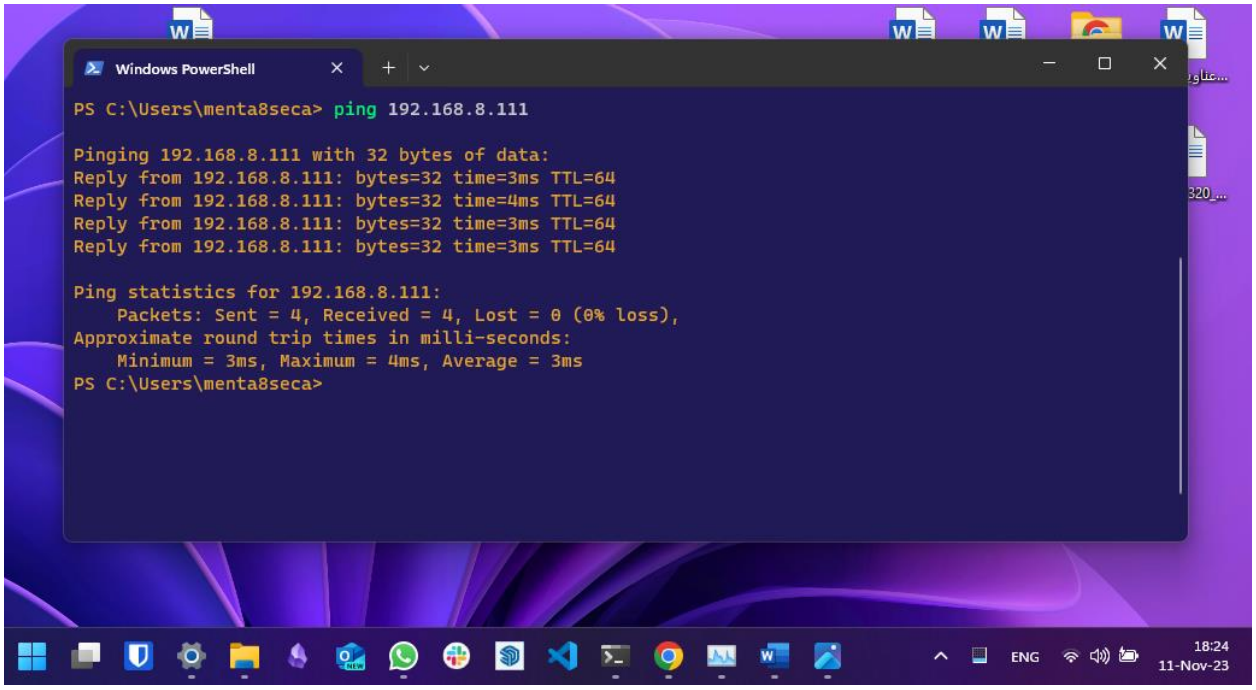
Task #1:

1- ping: a network utility command used to access the round -trip time(RTT) it takes for data to travel from one device to another within a network

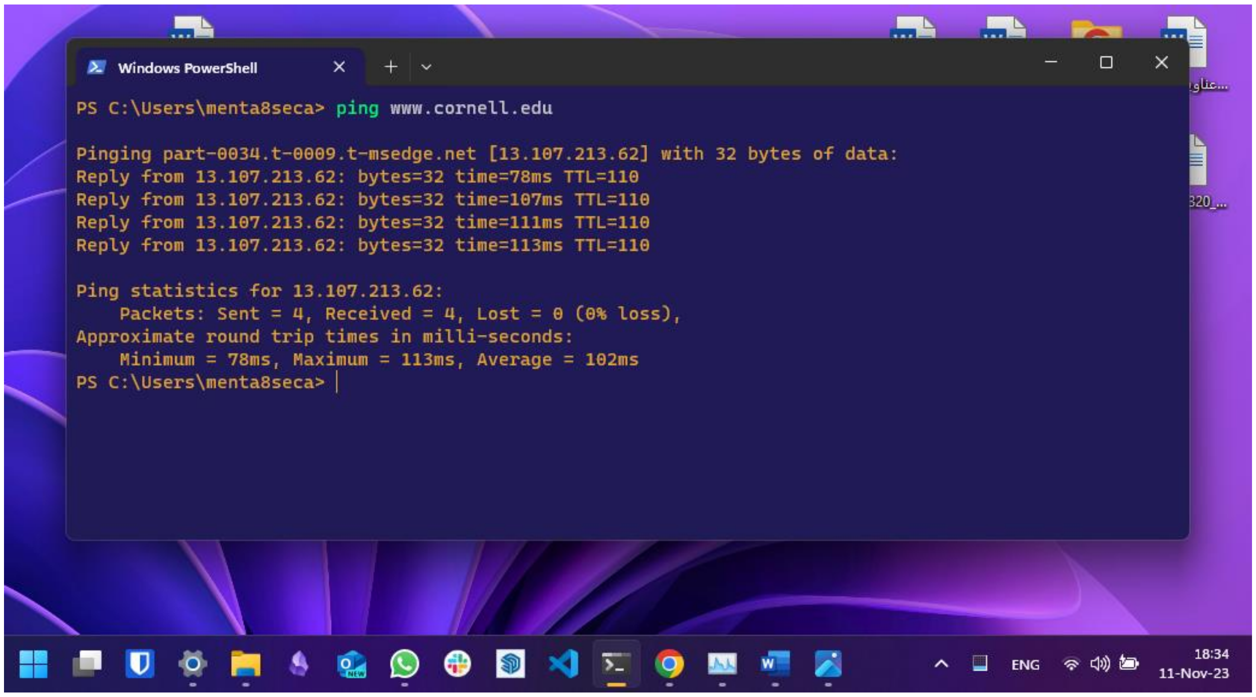
2- Tracert (trace route) : a network diagnostic tool that can be used to see the exact path of the data package and is usually a server or web host

3- Nslookup (name server lookup): a network administration tool used for querying Domain Name System (DNS) servers to retrieve information about domain names, IP addresses, and other DNS records.

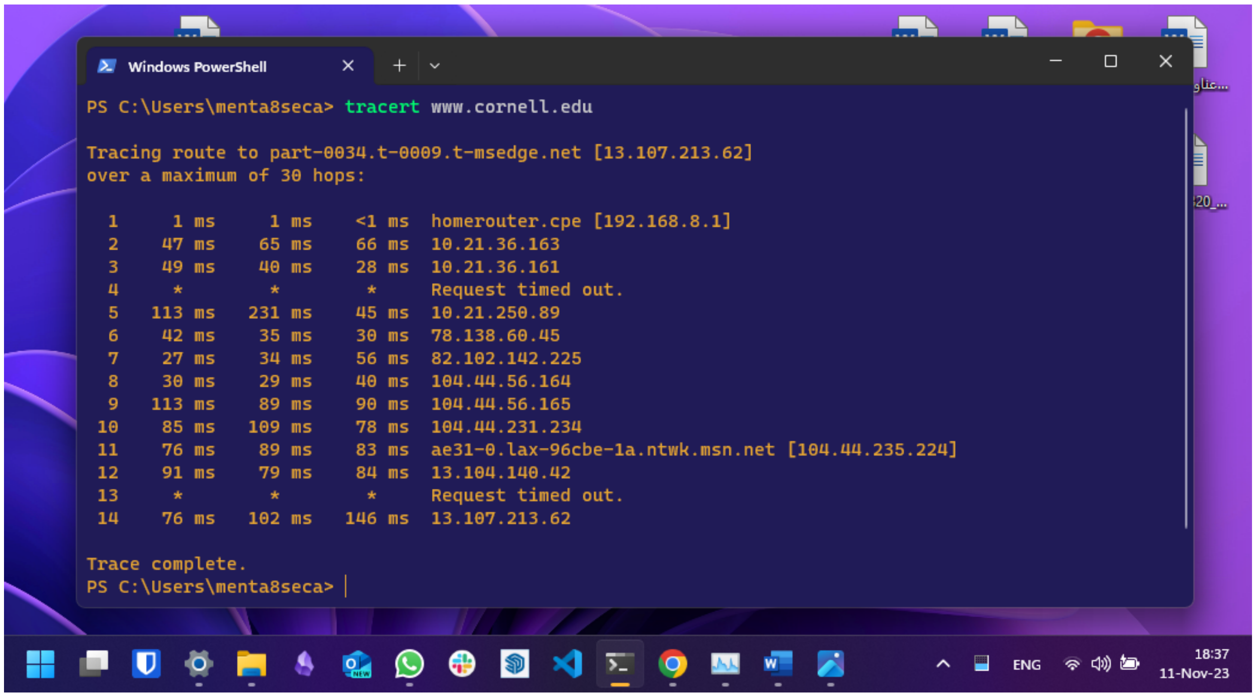
4- telnet: is a network protocol and that allows a user to communicate with a remote device or server over a TCP/IP network.



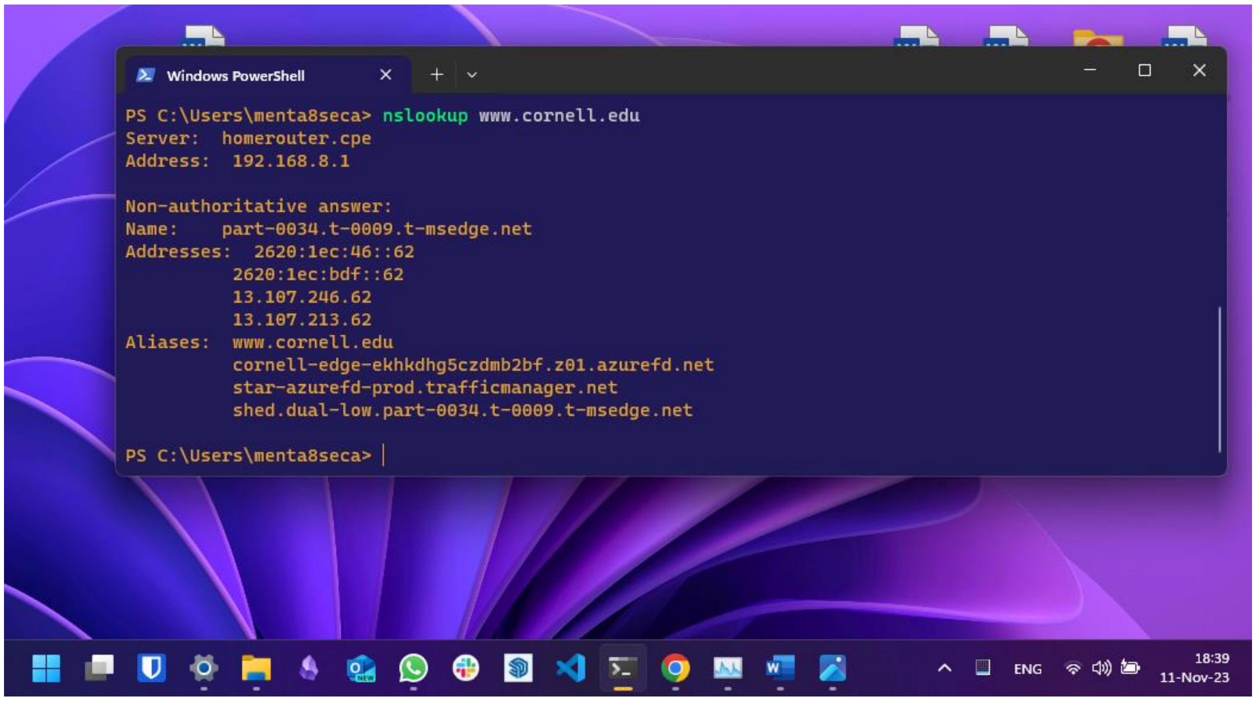
```
PS C:\Users\menta8seca> ping 192.168.8.111

Pinging 192.168.8.111 with 32 bytes of data:
Reply from 192.168.8.111: bytes=32 time=3ms TTL=64
Reply from 192.168.8.111: bytes=32 time=4ms TTL=64
Reply from 192.168.8.111: bytes=32 time=3ms TTL=64
Reply from 192.168.8.111: bytes=32 time=3ms TTL=64

Ping statistics for 192.168.8.111:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 3ms, Maximum = 4ms, Average = 3ms
PS C:\Users\menta8seca>
```



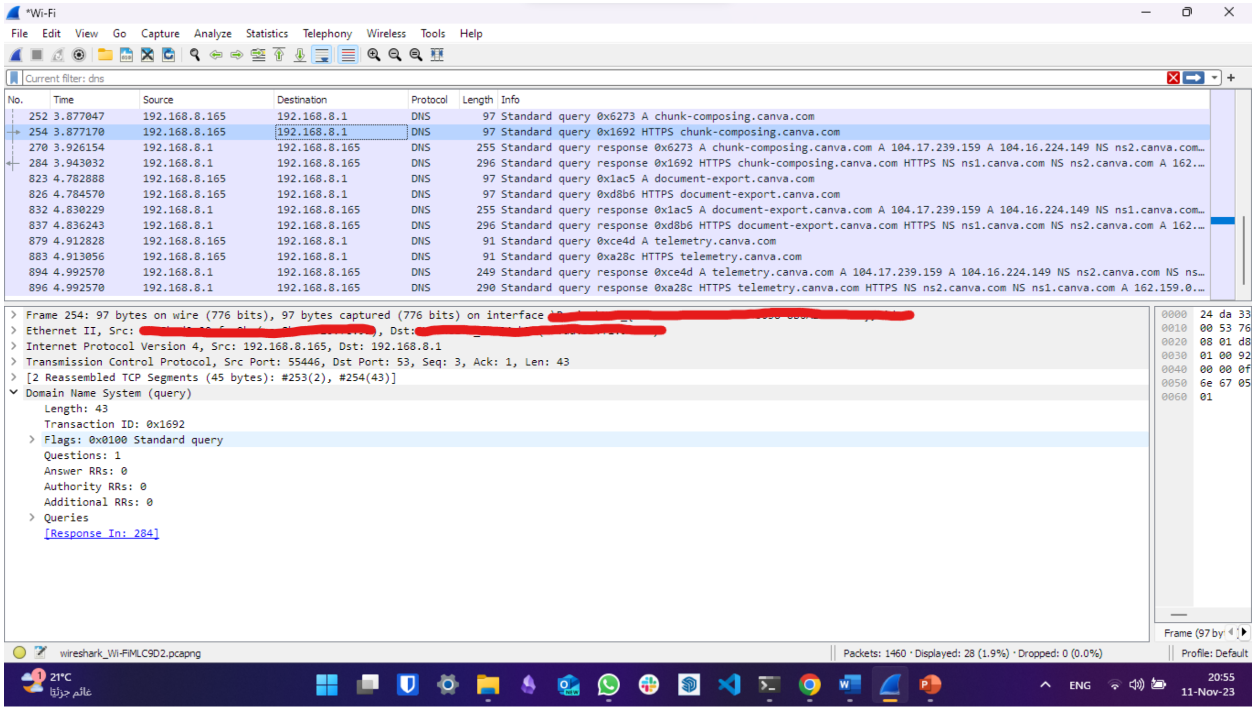
```
PS C:\Users\menta8seca> ping www.cornell.edu

Pinging part-0034.t-0009.t-msedge.net [13.107.213.62] with 32 bytes of data:
Reply from 13.107.213.62: bytes=32 time=78ms TTL=110
Reply from 13.107.213.62: bytes=32 time=107ms TTL=110
Reply from 13.107.213.62: bytes=32 time=111ms TTL=110
Reply from 13.107.213.62: bytes=32 time=113ms TTL=110

Ping statistics for 13.107.213.62:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 78ms, Maximum = 113ms, Average = 102ms
PS C:\Users\menta8seca>
```



```
PS C:\Users\menta8seca> tracert www.cornell.edu

Tracing route to part-0034.t-0009.t-msedge.net [13.107.213.62]
over a maximum of 30 hops:

  1     1 ms     1 ms    <1 ms  homerouter.cpe [192.168.8.1]
  2    47 ms    65 ms    66 ms  10.21.36.163
  3    49 ms    40 ms    28 ms  10.21.36.161
  4     *        *        *     Request timed out.
  5   113 ms   231 ms    45 ms  10.21.250.89
  6    42 ms    35 ms    30 ms  78.138.60.45
  7    27 ms    34 ms    56 ms  82.102.142.225
  8    30 ms    29 ms    40 ms  104.44.56.164
  9   113 ms    89 ms    90 ms  104.44.56.165
 10    85 ms   109 ms    78 ms  104.44.231.234
 11    76 ms    89 ms    83 ms  ae31-0.lax-96cbe-1a.ntwk.msn.net [104.44.235.224]
 12    91 ms    79 ms    84 ms  13.104.140.42
 13     *        *        *     Request timed out.
 14    76 ms   102 ms   146 ms  13.107.213.62

Trace complete.
PS C:\Users\menta8seca>
```

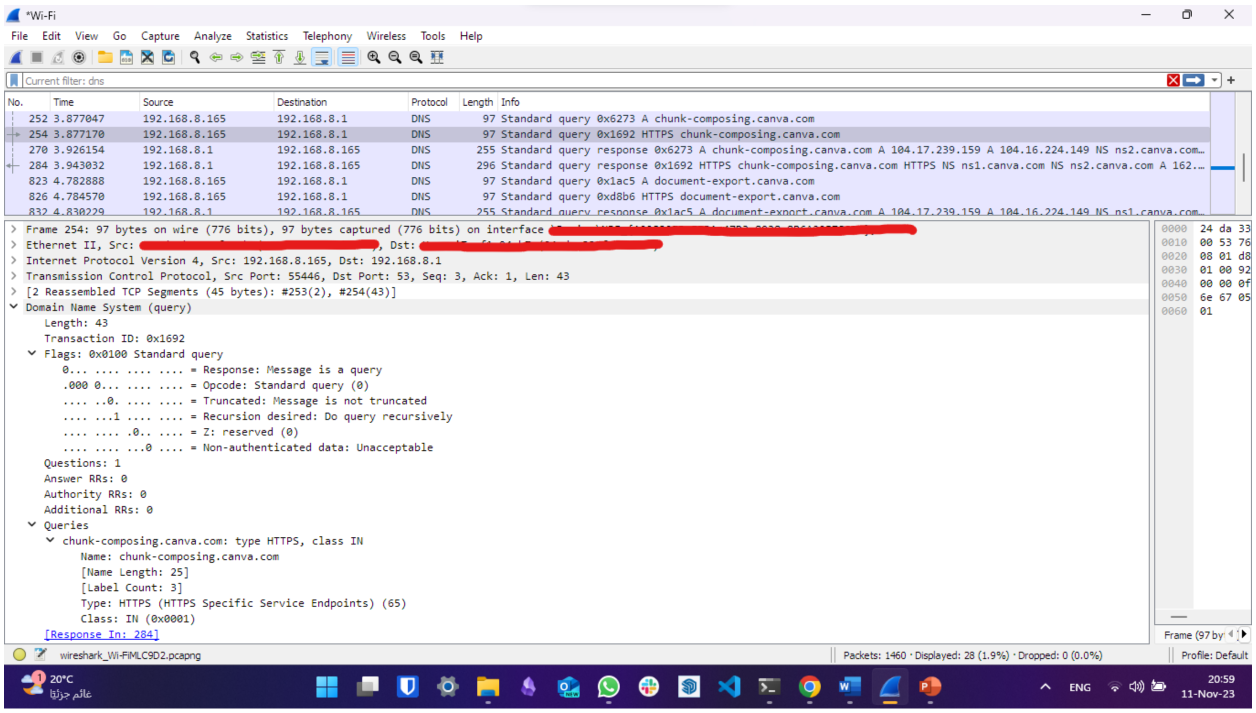


```
PS C:\Users\menta8seca> nslookup www.cornell.edu
Server:  homerouter.cpe
Address:  192.168.8.1

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

PS C:\Users\menta8seca>
```



```
Current filter: dns

No.    Time       Source          Destination     Protocol Length Info
  252 3.877047   192.168.8.165   192.168.8.1     DNS        97 Standard query 0x6273 A chunk-composing.canva.com
  254 3.877170   192.168.8.165   192.168.8.1     DNS        97 Standard query 0x1692 HTTPS chunk-composing.canva.com
  270 3.926154   192.168.8.1     192.168.8.165   DNS       255 Standard query response 0x6273 A chunk-composing.canva.com A 104.17.239.159 A 104.16.224.149 NS ns2.canva.com…
  284 3.943032   192.168.8.1     192.168.8.165   DNS       296 Standard query response 0x1692 HTTPS chunk-composing.canva.com HTTPS NS ns1.canva.com NS ns2.canva.com A 162.…
  823 4.782888   192.168.8.165   192.168.8.1     DNS        97 Standard query 0x1ac5 A document-export.canva.com
  826 4.784570   192.168.8.165   192.168.8.1     DNS        97 Standard query 0xd8b6 HTTPS document-export.canva.com
  832 4.830229   192.168.8.1     192.168.8.165   DNS       255 Standard query response 0x1ac5 A document-export.canva.com A 104.17.239.159 A 104.16.224.149 NS ns1.canva.com…
  837 4.836243   192.168.8.1     192.168.8.165   DNS       296 Standard query response 0xd8b6 HTTPS document-export.canva.com HTTPS NS ns1.canva.com NS ns2.canva.com A 162.…
  879 4.912828   192.168.8.165   192.168.8.1     DNS        91 Standard query 0xce4d A telemetry.canva.com
  883 4.913056   192.168.8.165   192.168.8.1     DNS        91 Standard query 0xa28c HTTPS telemetry.canva.com
  894 4.992570   192.168.8.1     192.168.8.165   DNS       249 Standard query response 0xce4d A telemetry.canva.com A 104.17.239.159 A 104.16.224.149 NS ns2.canva.com NS ns…
  896 4.992570   192.168.8.1     192.168.8.165   DNS       290 Standard query response 0xa28c HTTPS telemetry.canva.com HTTPS NS ns2.canva.com NS ns1.canva.com A 162.159.0.…

> Frame 254: 97 bytes on wire (776 bits), 97 bytes captured (776 bits) on interface [redacted]
> Ethernet II, Src: [redacted], Dst: [redacted]
> Internet Protocol Version 4, Src: 192.168.8.165, Dst: 192.168.8.1
> Transmission Control Protocol, Src Port: 55446, Dst Port: 53, Seq: 3, Ack: 1, Len: 43
> [2 Reassembled TCP Segments (45 bytes): #253(2), #254(43)]
v Domain Name System (query)
    Length: 43
    Transaction ID: 0x1692
  > Flags: 0x0100 Standard query
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  > Queries
    [Response In: 284]

wireshark_Wi-FiMLC9D2.pcapng    Packets: 1460 · Displayed: 28 (1.9%) · Dropped: 0 (0.0%)    Profile: Default
```

What appears in the picture are the DNS packets that are sent and received specifically to and from canva.com, it can be seen that there's 3 pairs of DNS requests and 3 pairs of DNS responses, a packet per pair requests the IP address of canva.com, and after that is received an HTTP request happens to bring the actual data



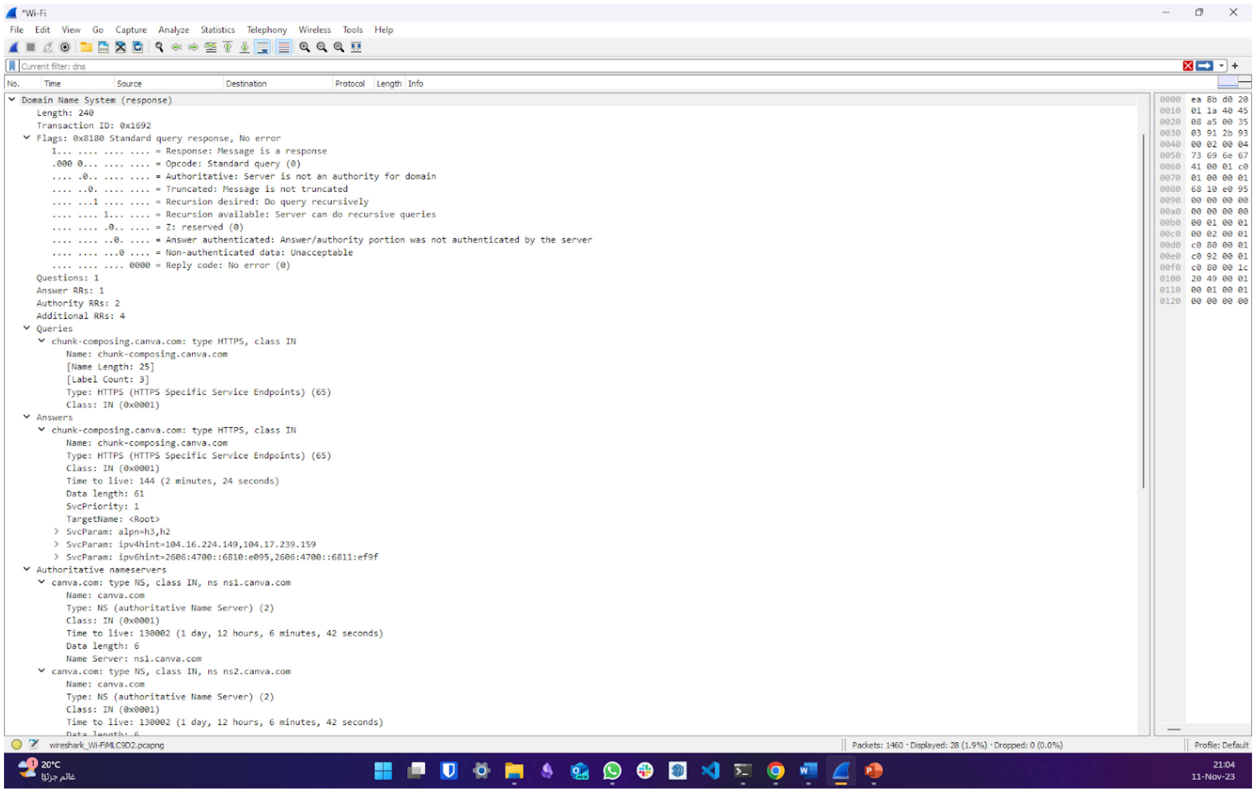
```
No.  Time      Source         Destination    Protocol Length Info
 252 3.877047  192.168.8.165  192.168.8.1    DNS       97 Standard query 0x6273 A chunk-composing.canva.com
 254 3.877170  192.168.8.165  192.168.8.1    DNS       97 Standard query 0x1692 HTTPS chunk-composing.canva.com
 270 3.926154  192.168.8.1    192.168.8.165  DNS      255 Standard query response 0x6273 A chunk-composing.canva.com A 104.17.239.159 A 104.16.224.149 NS ns2.canva.com…
 284 3.943032  192.168.8.1    192.168.8.165  DNS      296 Standard query response 0x1692 HTTPS chunk-composing.canva.com HTTPS NS ns1.canva.com NS ns2.canva.com A 162.…
 823 4.782888  192.168.8.165  192.168.8.1    DNS       97 Standard query 0x1ac5 A document-export.canva.com
 826 4.784570  192.168.8.165  192.168.8.1    DNS       97 Standard query 0xd8b6 HTTPS document-export.canva.com
 832 4.830229  192.168.8.1    192.168.8.165  DNS      255 Standard query response 0x1ac5 A document-export.canva.com A 104.17.239.159 A 104.16.224.149 NS ns1.canva.com…
```

```
> Frame 254: 97 bytes on wire (776 bits), 97 bytes captured (776 bits) on interface [redacted]
> Ethernet II, Src: [redacted], Dst: [redacted]
> Internet Protocol Version 4, Src: 192.168.8.165, Dst: 192.168.8.1
> Transmission Control Protocol, Src Port: 55446, Dst Port: 53, Seq: 3, Ack: 1, Len: 43
> [2 Reassembled TCP Segments (45 bytes): #253(2), #254(43)]
v Domain Name System (query)
    Length: 43
    Transaction ID: 0x1692
  v Flags: 0x0100 Standard query
      0... .... .... .... = Response: Message is a query
      .000 0... .... .... = Opcode: Standard query (0)
      .... ..0. .... .... = Truncated: Message is not truncated
      .... ...1 .... .... = Recursion desired: Do query recursively
      .... .... .0.. .... = Z: reserved (0)
      .... .... ...0 .... = Non-authenticated data: Unacceptable
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  v Queries
    v chunk-composing.canva.com: type HTTPS, class IN
        Name: chunk-composing.canva.com
        [Name Length: 25]
        [Label Count: 3]
        Type: HTTPS (HTTPS Specific Service Endpoints) (65)
        Class: IN (0x0001)
    [Response In: 284]
```

```
0000  24 da 33
0010  00 53 76
0020  08 01 d8
0030  01 00 92
0040  00 00 0f
0050  6e 67 05
0060  01
```

```
wireshark_Wi-FiMLC9D2.pcapng    Packets: 1460 · Displayed: 28 (1.9%) · Dropped: 0 (0.0%)    Profile: Default
```

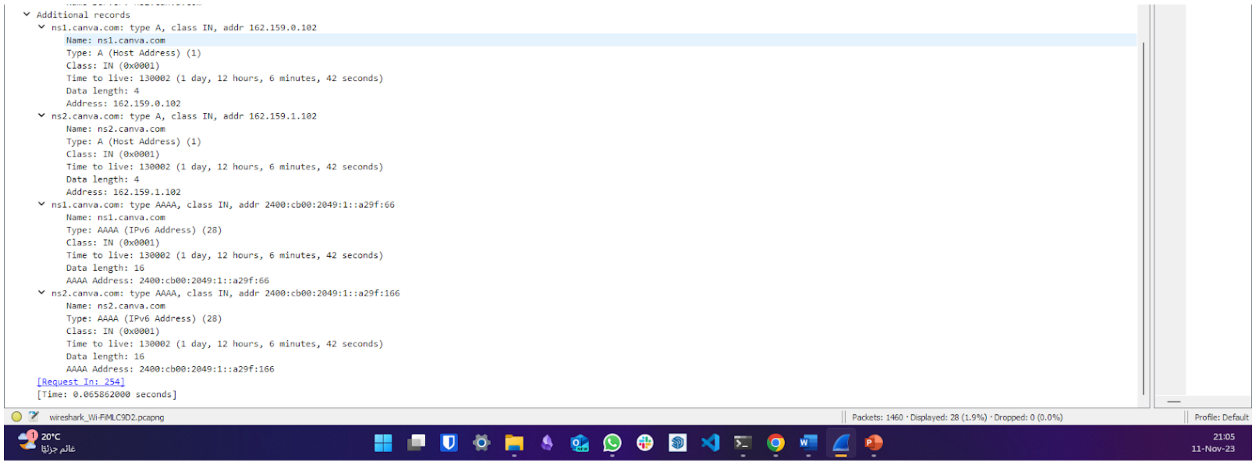
This picture shows the DNS message of the http request, it can be seen that the request has an id, a bunch of flags, 1 question, and a query.



```
v Domain Name System (response)
    Length: 240
    Transaction ID: 0x1692
  v Flags: 0x8180 Standard query response, No error
      1... .... .... .... = Response: Message is a response
      .000 0... .... .... = Opcode: Standard query (0)
      .... .0.. .... .... = Authoritative: Server is not an authority for domain
      .... ..0. .... .... = Truncated: Message is not truncated
      .... ...1 .... .... = Recursion desired: Do query recursively
      .... .... 1... .... = Recursion available: Server can do recursive queries
      .... .... .0.. .... = Z: reserved (0)
      .... .... ..0. .... = Answer authenticated: Answer/authority portion was not authenticated by the server
      .... .... ...0 .... = Non-authenticated data: Unacceptable
      .... .... .... 0000 = Reply code: No error (0)
    Questions: 1
    Answer RRs: 1
    Authority RRs: 2
    Additional RRs: 4
  v Queries
    v chunk-composing.canva.com: type HTTPS, class IN
        Name: chunk-composing.canva.com
        [Name Length: 25]
        [Label Count: 3]
        Type: HTTPS (HTTPS Specific Service Endpoints) (65)
        Class: IN (0x0001)
  v Answers
    v chunk-composing.canva.com: type HTTPS, class IN
        Name: chunk-composing.canva.com
        Type: HTTPS (HTTPS Specific Service Endpoints) (65)
        Class: IN (0x0001)
        Time to live: 144 (2 minutes, 24 seconds)
        Data length: 61
        SvcPriority: 1
        TargetName: <Root>
      > SvcParam: alpn=h3,h2
      > SvcParam: ipv4hint=104.16.224.149,104.17.239.159
      > SvcParam: ipv6hint=2606:4700::6810:e095,2606:4700::6811:ef9f
  v Authoritative nameservers
    v canva.com: type NS, class IN, ns ns1.canva.com
        Name: canva.com
        Type: NS (authoritative Name Server) (2)
        Class: IN (0x0001)
        Time to live: 130002 (1 day, 12 hours, 6 minutes, 42 seconds)
        Data length: 6
        Name Server: ns1.canva.com
    v canva.com: type NS, class IN, ns ns2.canva.com
        Name: canva.com
        Type: NS (authoritative Name Server) (2)
        Class: IN (0x0001)
        Time to live: 130002 (1 day, 12 hours, 6 minutes, 42 seconds)
```

```
0000  ea 8b d0 20
0010  01 1a 40 45
0020  08 a5 00 35
0030  03 91 2b 93
0040  00 02 00 04
0050  73 69 6e 67
0060  41 00 01 c0
0070  01 00 00 01
0080  68 10 e0 95
0090  00 00 00 00
00a0  00 00 00 00
00b0  00 01 00 01
00c0  00 02 00 01
00d0  c0 80 00 01
00e0  c0 92 00 01
00f0  c0 80 00 1c
0100  20 49 00 01
0110  00 01 00 01
0120  00 00 00 00
```

```
wireshark_Wi-FiMLC9D2.pcapng    Packets: 1460 · Displayed: 28 (1.9%) · Dropped: 0 (0.0%)    Profile: Default
```

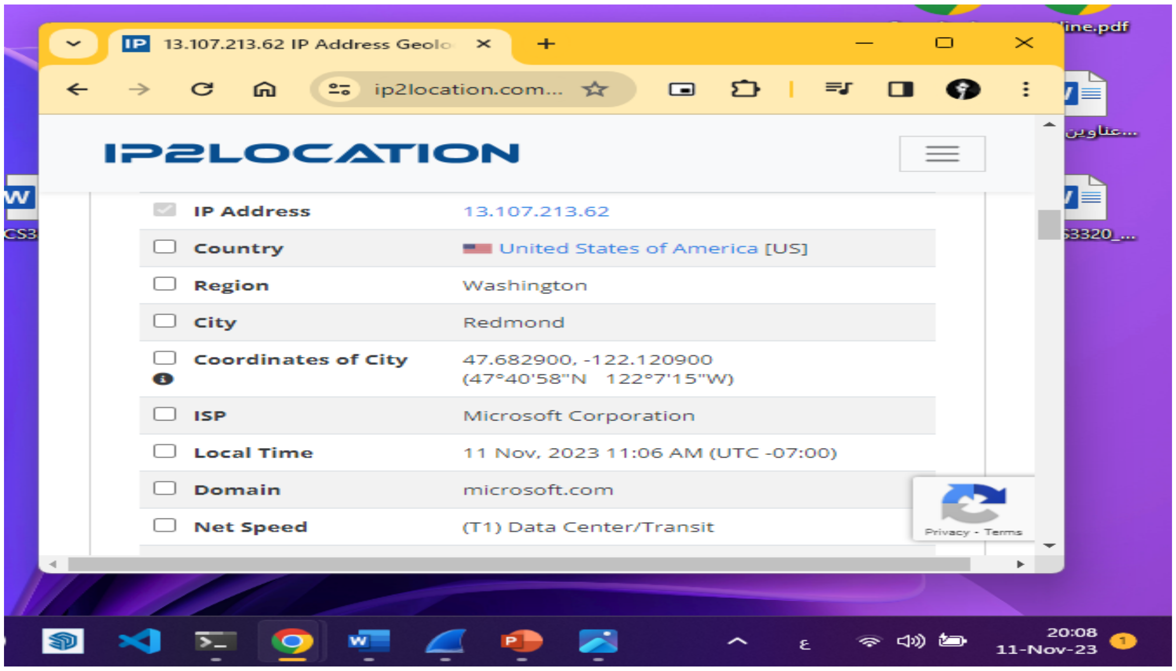


```
Additional records
  ns1.canva.com: type A, class IN, addr 162.159.0.102
      Name: ns1.canva.com
      Type: A (Host Address) (1)
      Class: IN (0x0001)
      Time to live: 130002 (1 day, 12 hours, 6 minutes, 42 seconds)
      Data length: 4
      Address: 162.159.0.102
  ns2.canva.com: type A, class IN, addr 162.159.1.102
      Name: ns2.canva.com
      Type: A (Host Address) (1)
      Class: IN (0x0001)
      Time to live: 130002 (1 day, 12 hours, 6 minutes, 42 seconds)
      Data length: 4
      Address: 162.159.1.102
  ns1.canva.com: type AAAA, class IN, addr 2400:cb00:2049:1::a29f:66
      Name: ns1.canva.com
      Type: AAAA (IPv6 Address) (28)
      Class: IN (0x0001)
      Time to live: 130002 (1 day, 12 hours, 6 minutes, 42 seconds)
      Data length: 16
      AAAA Address: 2400:cb00:2049:1::a29f:66
  ns2.canva.com: type AAAA, class IN, addr 2400:cb00:2049:1::a29f:166
      Name: ns2.canva.com
      Type: AAAA (IPv6 Address) (28)
      Class: IN (0x0001)
      Time to live: 130002 (1 day, 12 hours, 6 minutes, 42 seconds)
      Data length: 16
      AAAA Address: 2400:cb00:2049:1::a29f:166
[Request In: 254]
[Time: 0.065862000 seconds]
```

wireshark_Wi-FiMLC9D2.pcapng
Packets: 1460 · Displayed: 28 (1.9%) · Dropped: 0 (0.0%)
Profile: Default
20°C
غائم جزئيا
21:05
11-Nov-23

In the 2 pictures above, the HTTP response packet message appears to include the same transaction ID as the request, some flags, one question, and an answer, with 2 authoritative nameservers, 4 additional records, and finally the DNS time of 65ms

From the ping results, do you think the response you have got is from the USA? Explain your answer briefly.
Answer: yes, because after ping resolved www.cornell.edu IP address from a DNS server, we tracked the location of the servers hosting it using that IP, as shown in the the picture down below



IP Address	13.107.213.62
Country	United States of America [US]
Region	Washington
City	Redmond
Coordinates of City	47.682900, -122.120900 (47°40'58"N 122°7'15"W)
ISP	Microsoft Corporation
Local Time	11 Nov, 2023 11:06 AM (UTC -07:00)
Domain	microsoft.com
Net Speed	(T1) Data Center/Transit

Task #2

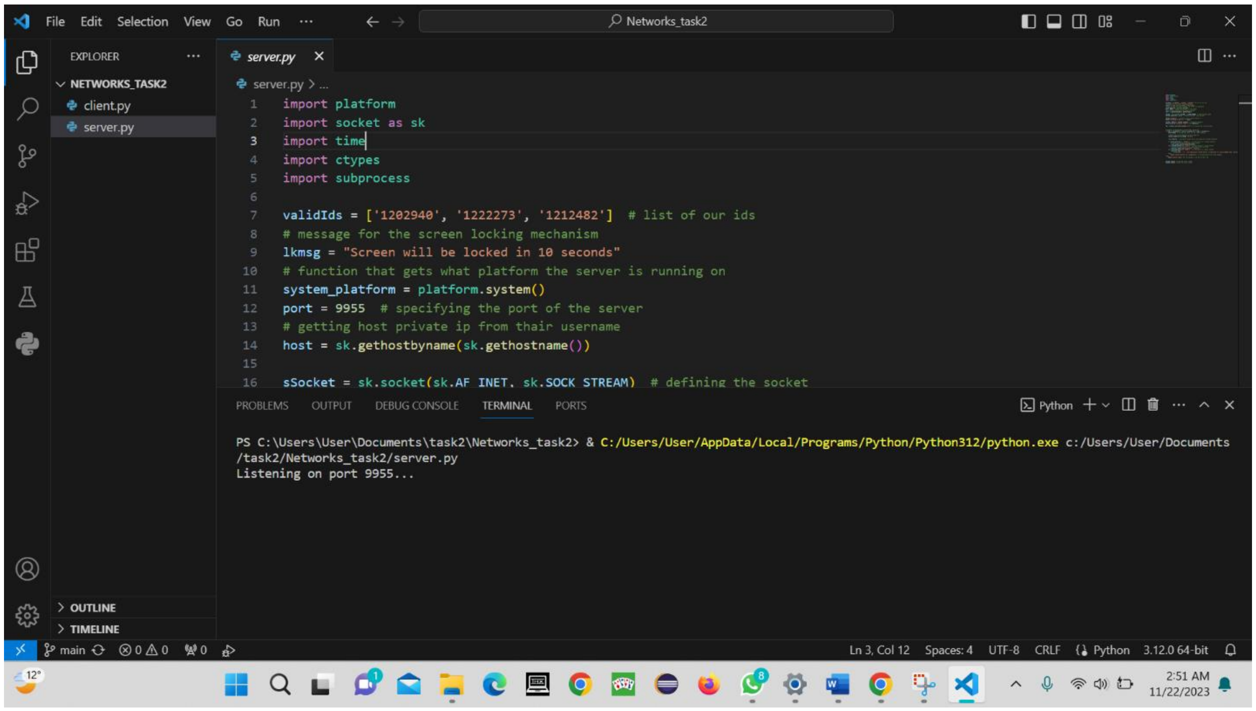
Server code:

```python
import platform
import socket as sk
import time
import ctypes
import subprocess

validIds = ['1202940', '1222273', '1212482']  # list of our ids
# message for the screen locking mechanism
lkmsg = "Screen will be locked in 10 seconds"
# function that gets what platform the server is running on
system_platform = platform.system()
port = 9955  # specifying the port of the server
# getting host private ip from thair username
host = sk.gethostbyname(sk.gethostname())

sSocket = sk.socket(sk.AF_INET, sk.SOCK_STREAM)  # defining the socket
sSocket.bind((host, port))  # binding the host with the port

sSocket.listen(1)  # listening with a request queue of 1
print("Listening on port %s..." % port)

csocket, address = sSocket.accept()  # accepting requests
print("Received connection from %s" % str(address))

msg = csocket.recv(1024).decode("utf-8")  # recieving the client message


# checking if the message is one of the 3 valid ids
if (msg == validIds[0] or msg == validIds[1] or msg == validIds[2]):
    print(lkmsg)  # warning the server for the screen locking

    # sending the warning message to the client too
    csocket.sendall(bytes(lkmsg, "utf-8"))

    time.sleep(10)  # wating 10 seconds before performing the locking mechanism

    if (system_platform == 'Windows'):  # if the server on a windows machine
        # screen locking function from ctypes library
        ctypes.windll.user32.LockWorkStation()
    elif (system_platform == 'Linux'):  # if the server on a linux machine
        # screen locking function from subprocess library on linux
        subprocess.run(["xdg-screensaver", "lock"])
    elif (system_platform == 'macOS'):  # if the server on a macos machine
        subprocess.run(
            ["osascript", "-e", 'tell application "System Events" to keystroke
"q" using {command down, control down}'])  # screen locking function from
subprocess library on linux
    else:
        print('system platform not recognized') # if system platform not found
somehow
else:
    print("Invalid input") #if the message is not one of the 3 ids


csocket.close() # close the client socket
sSocket.close() # close the server socket
```

client code:

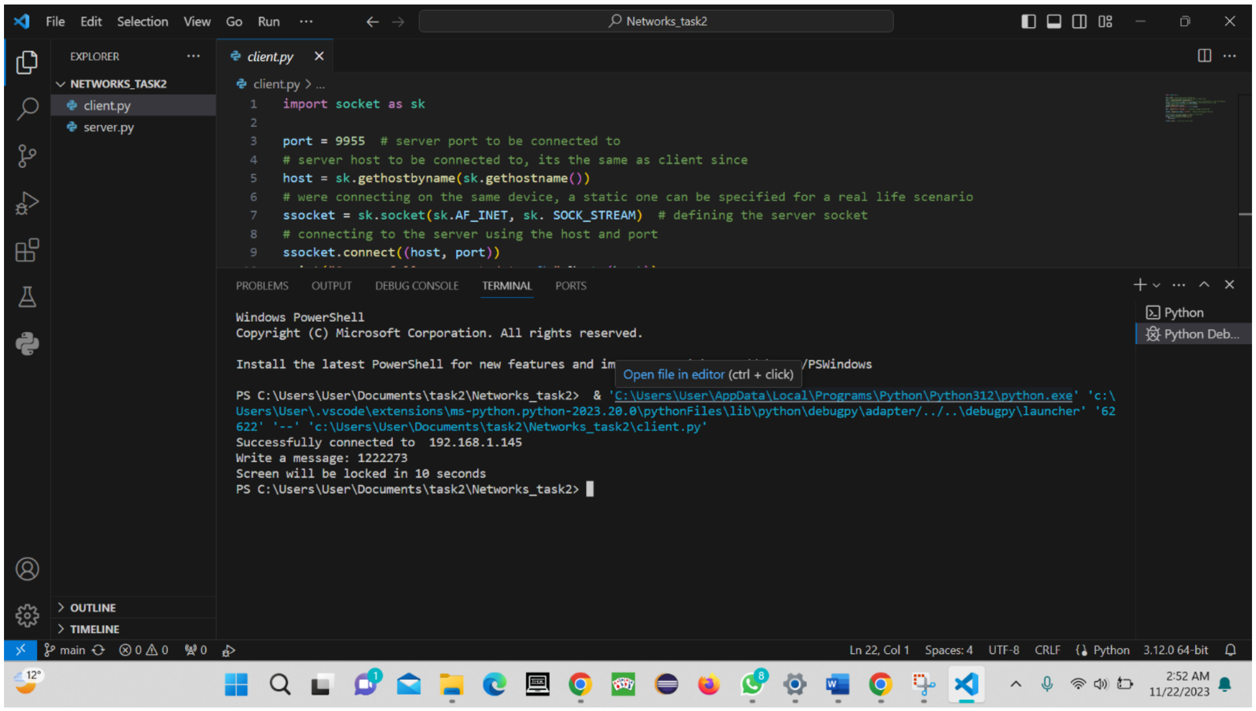
```
import socket as sk

port = 9955  # server port to be connected to
# server host to be connected to, its the same as client since
host = sk.gethostbyname(sk.gethostname())
# were connecting on the same device, a static one can be specified for a real
life scenario
ssocket = sk.socket(sk.AF_INET, sk. SOCK_STREAM)  # defining the server socket
# connecting to the server using the host and port
ssocket.connect((host, port))
print("Successfully connected to  %s" % str(host))

mess = input("Write a message: ")  # getting a message from the user

ssocket. sendall(bytes(mess, "utf-8"))  # sending the message to the user

# recieving the warning message of locking if it was sent
warn = ssocket.recv(1024).decode("utf-8")
if warn:  # if warning was sent print it
    print(warn)

ssocket.close()  # close the client socket
```

When we entered a valid ID number, a message appears on the server side indicating that the screen will lock in 10 seconds. A similar message is also sent to the client's end. The screen successfully locked after 10 seconds



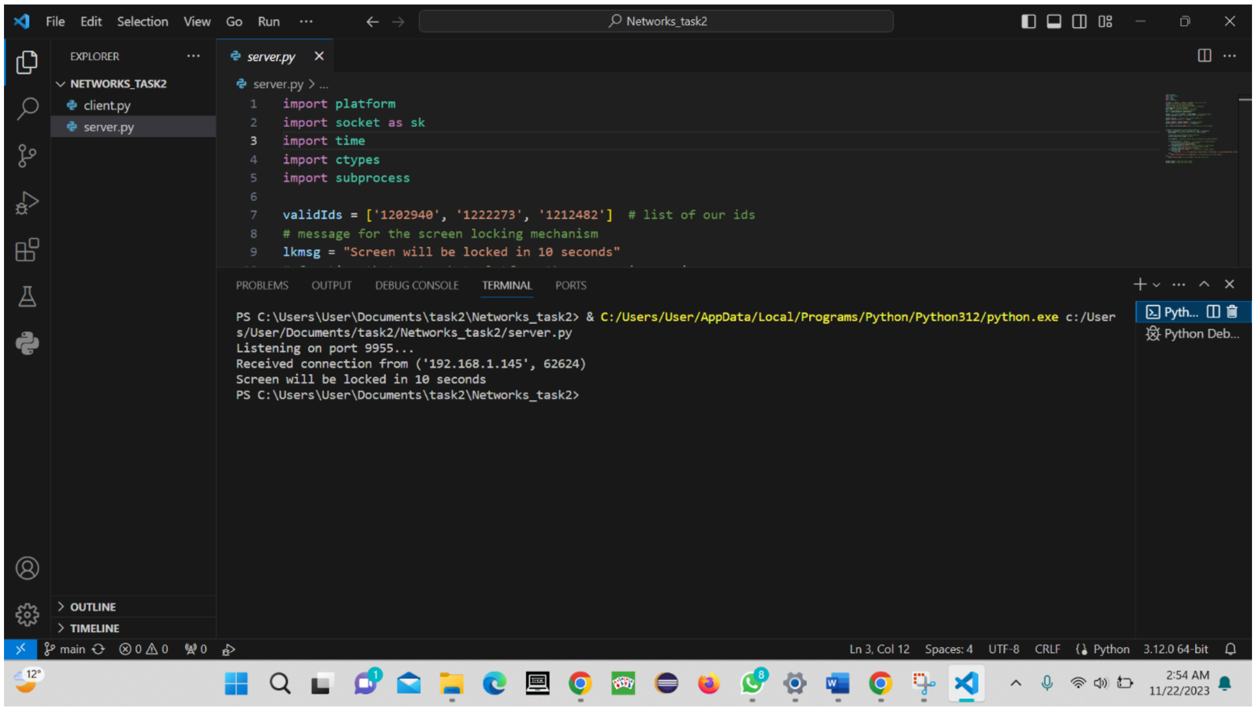
```python
import platform
import socket as sk
import time
import ctypes
import subprocess

validIds = ['1202940', '1222273', '1212482']  # list of our ids
# message for the screen locking mechanism
lkmsg = "Screen will be locked in 10 seconds"
# function that gets what platform the server is running on
system_platform = platform.system()
port = 9955  # specifying the port of the server
# getting host private ip from thair username
host = sk.gethostbyname(sk.gethostname())

sSocket = sk.socket(sk.AF_INET, sk.SOCK_STREAM)  # defining the socket
```

```
PS C:\Users\User\Documents\task2\Networks_task2> & C:/Users/User/AppData/Local/Programs/Python/Python312/python.exe c:/Users/User/Documents
/task2/Networks_task2/server.py
Listening on port 9955...
```



```python
import socket as sk

port = 9955  # server port to be connected to
# server host to be connected to, its the same as client since
host = sk.gethostbyname(sk.gethostname())
# were connecting on the same device, a static one can be specified for a real life scenario
ssocket = sk.socket(sk.AF_INET, sk. SOCK_STREAM)  # defining the server socket
# connecting to the server using the host and port
ssocket.connect((host, port))
```

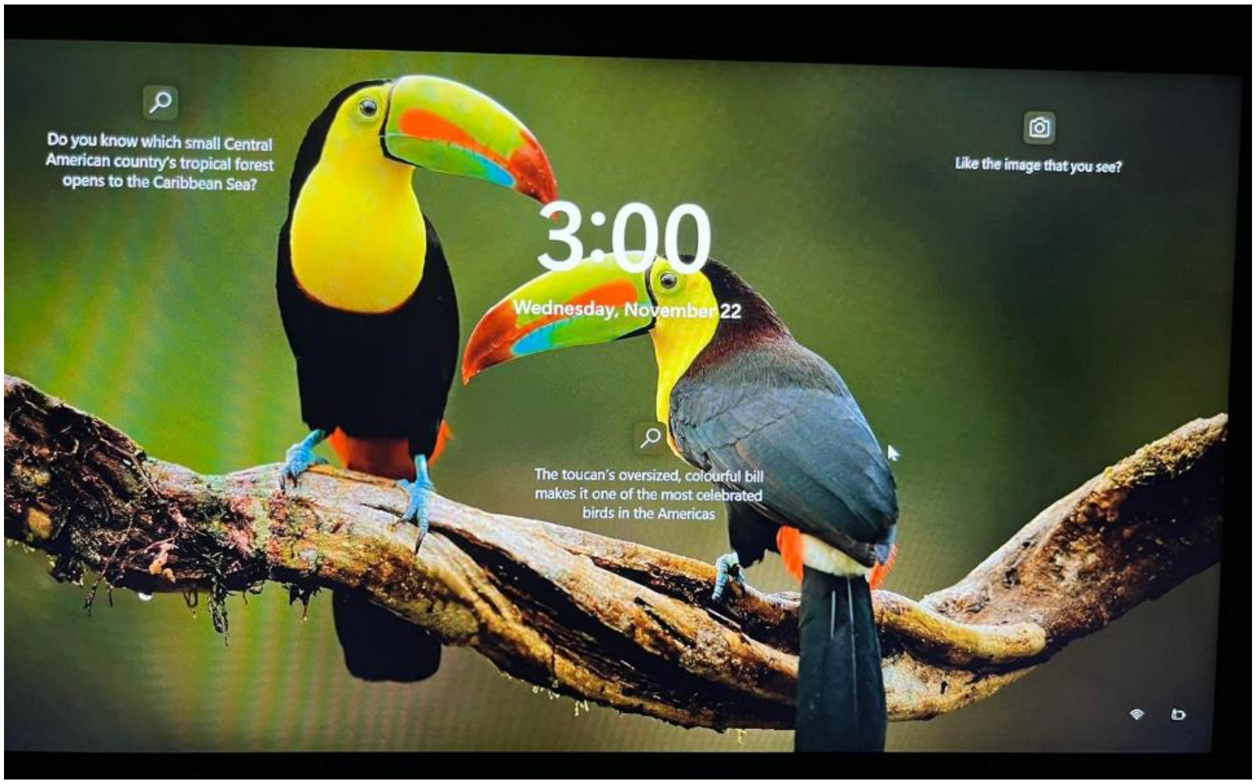
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and im                         /PSWindows

PS C:\Users\User\Documents\task2\Networks_task2>  & 'C:\Users\User\AppData\Local\Programs\Python\Python312\python.exe' 'c:\
Users\User\.vscode\extensions\ms-python.python-2023.20.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher' '62
622' '--' 'c:\Users\User\Documents\task2\Networks_task2\client.py'
Successfully connected to  192.168.1.145
Write a message: 1222273
Screen will be locked in 10 seconds
PS C:\Users\User\Documents\task2\Networks_task2> 
```



```python
import platform
import socket as sk
import time
import ctypes
import subprocess

validIds = ['1202940', '1222273', '1212482']  # list of our ids
# message for the screen locking mechanism
lkmsg = "Screen will be locked in 10 seconds"
```

```
PS C:\Users\User\Documents\task2\Networks_task2> & C:/Users/User/AppData/Local/Programs/Python/Python312/python.exe c:/User
s/User/Documents/task2/Networks_task2/server.py
Listening on port 9955...
Received connection from ('192.168.1.145', 62624)
Screen will be locked in 10 seconds
PS C:\Users\User\Documents\task2\Networks_task2>
```

Screen after 10 second:


Do you know which small Central American country's tropical forest opens to the Caribbean Sea?
Like the image that you see?
3:00
Wednesday, November 22
The toucan's oversized, colourful bill makes it one of the most celebrated birds in the Americas

Task #3

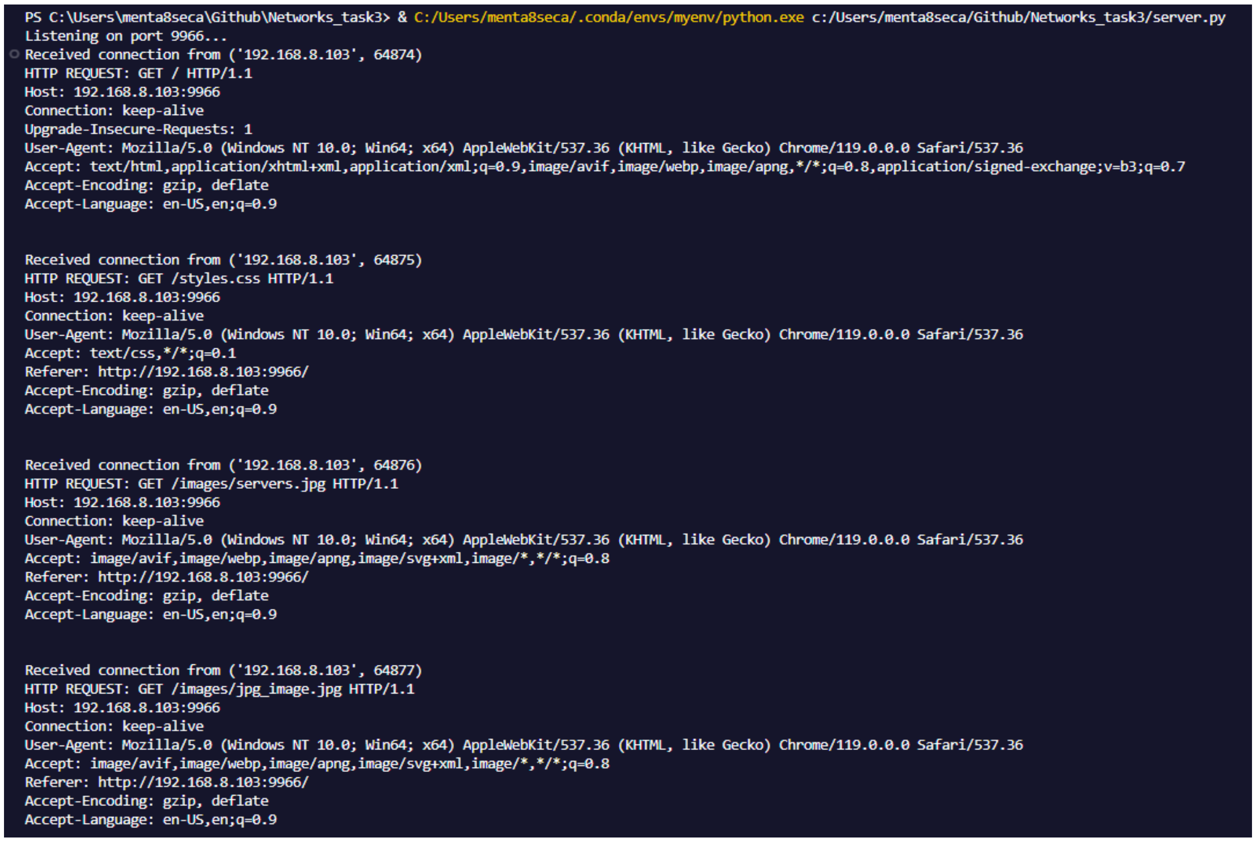
from rfce2616, what is Content-Type in the HTTP request and why do we need it?

```
 Content-Type is an entity-header field that indicates the media type of
the entity-body sent to the recipient or, in the case of the HEAD method,
the media type that would have been sent had the request been a GET.

Because it tells the server what kind of data is being sent in the request
so that the server can properly interpret and process it,it is crucial for
both the client and server to understand how to handle the data being
exchanged
```

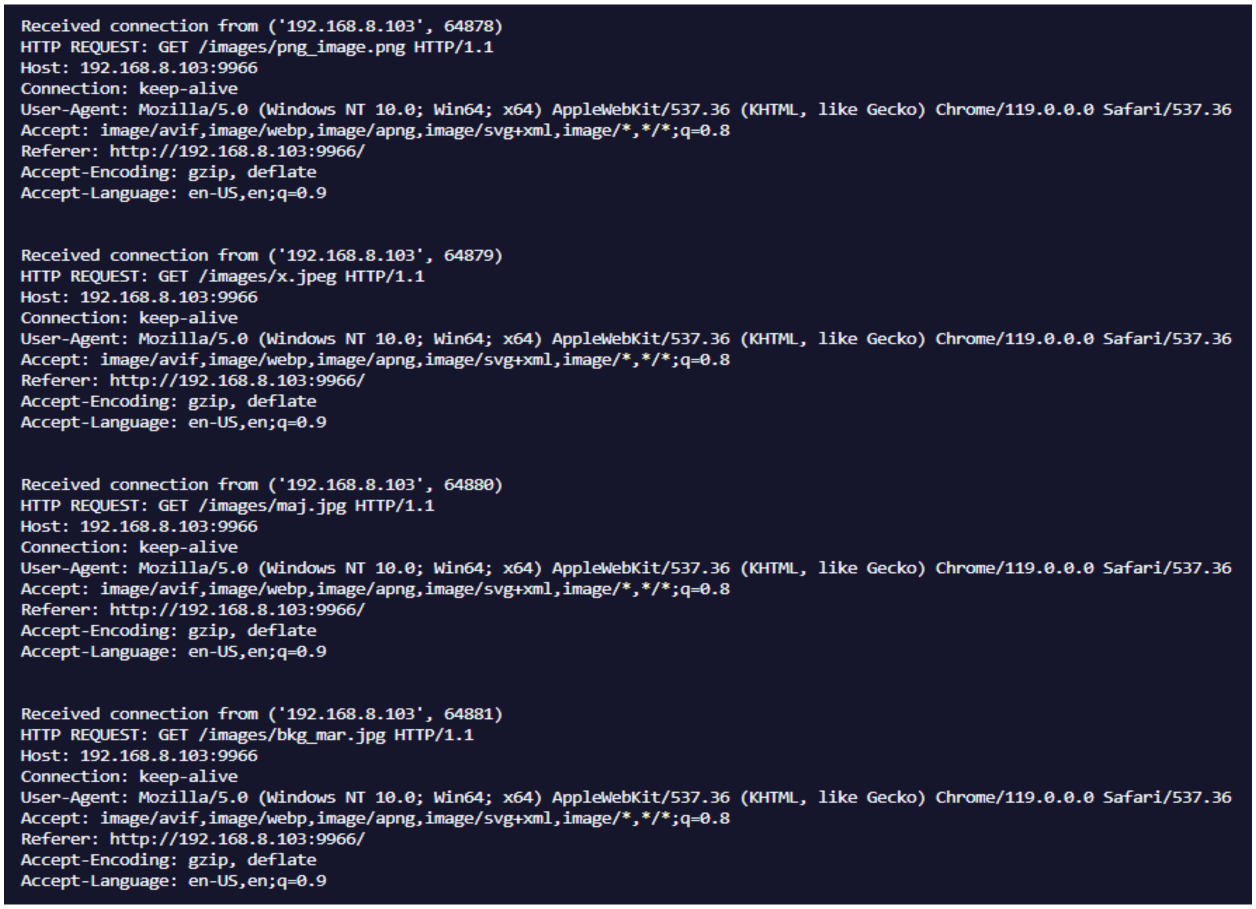
Server python code:

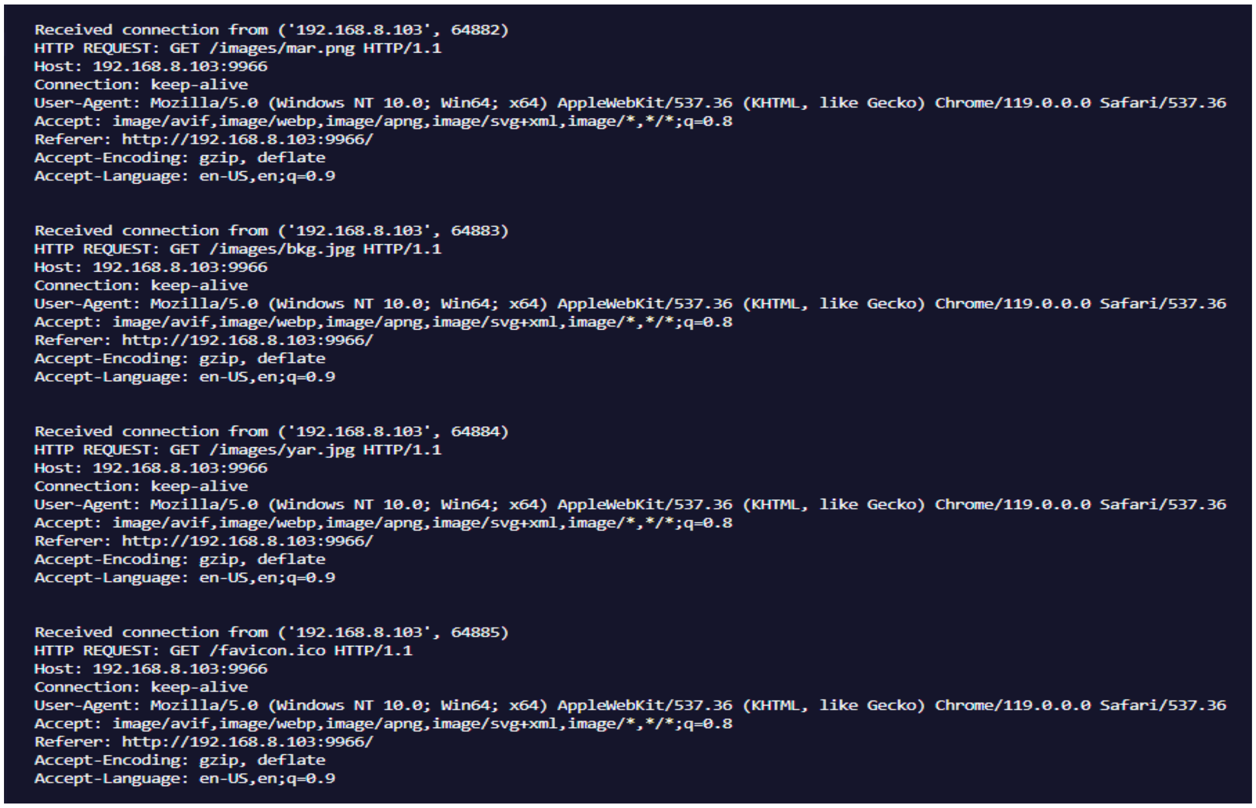
```
    import socket as sk  # importing the socket library as sk



# A function responsible for handling response sending
def send_response(csocket, status, type, filee):
    # sending the response status
    csocket.send(bytes('HTTP/1.1 '+status+'\r\n', "utf-8"))
    # sending the response content-type
    csocket.send(bytes('Content-Type: '+type+'\r\n', "utf-8"))
    csocket.send(bytes('\r\n', "utf-8"))
    if isinstance(filee, str):  # if the file is text being html or css it is
sent with encoding
        csocket.send(filee.encode())
    else:  # else the file is binary and its sent without encoding
        csocket.send(filee)


def read_file(loc, rtype):  # A function responsible for handling file opening
and reading
    try:  # try statement to catch reading file exceptions to handle them
properly
        if rtype == 'rb':  # if reading type is binary no encoding is specified
            f1 = open(loc, rtype)
        else:  # encoding is specified if file reading is text
            f1 = open(loc, rtype, encoding="utf-8")
        file_read = f1.read()  # reading the file
        f1.close()
        return file_read  # returning the read file
    except OSError:
        return None  # returning None if the file doesnt exist to identify that
it doesnt


def call_error(csocket, address):  # a function responsible for handling error
page calling
    # calling the read function giving it the error.html file with reading type
of r
    temp = read_file("error.html", "r")
    # formatting the read error.html file to insert the address and port in a
location specified by {info}
    order = temp.format(info=address)
    # sending a 404 response with the formatted error.html file with the send
function
    send_response(csocket, '404 Not Found', 'text/html', order)


def main():  # main function where code starts
    port = 9966  # specifying the port
    # giving the host variable the device local ip to start hosting using its
host name
    host = sk.gethostbyname(sk.gethostname())

    filename = None  # A variable for holding the value of the file name being
requested
    filetype = None  # A variable for holding the content-type for the variable
being requested
    statustype = None  # A variable for holding the appropriate response status
of the response
    rtype = None  # A variable for holding what type of reading should be
performed when opeing a file

    # defining the server socket
    sSocket = sk.socket(sk.AF_INET, sk.SOCK_STREAM)
    sSocket.bind((host, port))  # binding the host with the port

    # listening to clients with a request queue with the size of 5
    sSocket.listen(5)
    print("Listening on port %s..." % port)

    while True:  # server listening infinite loop
        rtype = 'r'  # defaulting the reading type to r
        csocket, address = sSocket.accept()  # accepting requests
        print("Received connection from %s" % str(address))
        msg = csocket.recv(1024).decode("utf-8")  # recieving the full request
        print('HTTP REQUEST: '+msg)
        request = msg.split()[1]  # stripping the message out of the request

        # specifying the request info for the main_en.html file
        if (request == '/') or (request == ('/main_en.html')) or (request ==
'/en') or (request == '/index.html'):
            filename = 'main_en.html'  # specifying the file name
            filetype = 'text/html'  # specifying the file content-type
        elif (request == '/ar'):  # specifying the request info for the
main_ar.html file
            filename = 'main_ar.html'
            filetype = 'text/html'
        # specifying the request info for other .html files and .css files
        elif (request.endswith(".html")) or (request.endswith(".css")):
            filename = request[1:]  # stripping the file path from the first /
            # taking out the file extension from the file path
            filetype = 'text/'+request.split('.')[-1]
        # specifying the request info for .jpg .jpeg and .png files
        elif request.endswith('.png') or request.endswith('.jpg') or
request.endswith('.jpeg'):
            filename = request[1:]
            filetype = 'image/'+request.split('.')[-1]
            rtype = 'rb'  # overriding the default readinng type to binary
reading
        # for handling the redirection requests
        elif (request == '/cr') or (request == '/so') or (request == '/rt'):
            # specifying the response status of the respond
            csocket.send('HTTP/1.1 307 Temporary Redirect\r\n'.encode())
            if (request == '/cr'):  # redirecting to cornell website
                csocket.send('Location://cornell.edu\r\n'.encode())
            elif (request == '/so'):  # redirecting to stackoverflow website
                csocket.send('Location://stackoverflow.com\r\n'.encode())
            elif (request == '/rt'):  # redirecting to ritaj website
                csocket.send('Location://ritaj.birzeit.edu\r\n'.encode())
            csocket.send('\r\n'.encode())
            csocket.close()  # closing the client socket since the request is
done
            continue  # jumping to the next loop
        else:
            # calling for the error page since the request is wrong
            call_error(csocket, address)
            csocket.close()
            continue
        statustype = '200 ok'  # specifying the response status
        # calling the read function to fetch the file read given the file path
and the reading type
        order = read_file(filename, rtype)
        if order:  # if file exists a 200 ok response is sent
            send_response(csocket, statustype, filetype, order)
        else:  # if the file doesnt exist a call for the error function is sent
with the client socket and both the address and port
            call_error(csocket, address)

        csocket.close()

    sSocket.close()


if __name__ == "__main__":  # calling the main function when the server script
starts running
    main()
```

The program should print the HTTP requests on the terminal window (command line window).



```
PS C:\Users\menta8seca\Github\Networks_task3> & C:/Users/menta8seca/.conda/envs/myenv/python.exe c:/Users/menta8seca/Github/Networks_task3/server.py
Listening on port 9966...
Received connection from ('192.168.8.103', 64874)
HTTP REQUEST: GET / HTTP/1.1
Host: 192.168.8.103:9966
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9


Received connection from ('192.168.8.103', 64875)
HTTP REQUEST: GET /styles.css HTTP/1.1
Host: 192.168.8.103:9966
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36
Accept: text/css,*/*;q=0.1
Referer: http://192.168.8.103:9966/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9


Received connection from ('192.168.8.103', 64876)
HTTP REQUEST: GET /images/servers.jpg HTTP/1.1
Host: 192.168.8.103:9966
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://192.168.8.103:9966/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9


Received connection from ('192.168.8.103', 64877)
HTTP REQUEST: GET /images/jpg_image.jpg HTTP/1.1
Host: 192.168.8.103:9966
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://192.168.8.103:9966/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```


```
Received connection from ('192.168.8.103', 64878)
HTTP REQUEST: GET /images/png_image.png HTTP/1.1
Host: 192.168.8.103:9966
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://192.168.8.103:9966/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9


Received connection from ('192.168.8.103', 64879)
HTTP REQUEST: GET /images/x.jpeg HTTP/1.1
Host: 192.168.8.103:9966
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://192.168.8.103:9966/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9


Received connection from ('192.168.8.103', 64880)
HTTP REQUEST: GET /images/maj.jpg HTTP/1.1
Host: 192.168.8.103:9966
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://192.168.8.103:9966/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9


Received connection from ('192.168.8.103', 64881)
HTTP REQUEST: GET /images/bkg_mar.jpg HTTP/1.1
Host: 192.168.8.103:9966
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://192.168.8.103:9966/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```


```
Received connection from ('192.168.8.103', 64882)
HTTP REQUEST: GET /images/mar.png HTTP/1.1
Host: 192.168.8.103:9966
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://192.168.8.103:9966/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9


Received connection from ('192.168.8.103', 64883)
HTTP REQUEST: GET /images/bkg.jpg HTTP/1.1
Host: 192.168.8.103:9966
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://192.168.8.103:9966/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9


Received connection from ('192.168.8.103', 64884)
HTTP REQUEST: GET /images/yar.jpg HTTP/1.1
Host: 192.168.8.103:9966
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://192.168.8.103:9966/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9


Received connection from ('192.168.8.103', 64885)
HTTP REQUEST: GET /favicon.ico HTTP/1.1
Host: 192.168.8.103:9966
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://192.168.8.103:9966/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

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 (request == '/') or (request == ('/main_en.html')) or (request ==
'/en') or (request == '/index.html'):

            filename = 'main_en.html'  # specifying the file name
            filetype = 'text/html'  # specifying the file content-type

order = read_file(filename, rtype)
        if order:  # if file exists a 200 ok response is sent
            send_response(csocket, statustype, filetype, order)
        else:  # if the file doesnt exist a call for the error function is sent
with the client socket and both the address and port
            call_error(csocket, address)
```

"ENCS3320-My Tiny Webserver 23/24" in the title

```
<head>
   <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>ENCS3320-My Tiny Webserver 23/24</title>
    <link rel="stylesheet" href="styles.css">
</head>
```

"Welcome to our course Computer Networks, This is a tiny webserver" (part of the phrase is in Blue)

```
<div class="text-overlay1">
            <h2>Welcome to our course: <span style="color: rgb(255, 217,
0);">Computer Networks</span><br> This is a
                tiny <span style="color: rgb(0, 238,
255);">webserver</span></h2>
        </div>
```

“Content-Type” box:

```
 <div class="ans1"><span style="font-size: 113%;"><strong>Why do we need
it?</strong></span>
           <hr><br><strong>Because it tells the server what kind of data is
being sent in the
                request so that the server can properly interpret and process
it,it is crucial for both the client and
                server to understand how to handle the data being
exchanged</strong>
        </div>
        <div class="q1">

           <span style="font-size: 113%;"><strong>What is Content-Type in the
HTTP request?</strong></span>
           <hr><br><strong>Content-Type is an entity-header field that
                indicates the media type of the
                entity-body sent to the recipient or, in the case of the HEAD
method,
                the media type that would have been sent had the request been a
GET</strong>

        </div>
```

Divide the page in different boxes and put student’s information in the different boxes

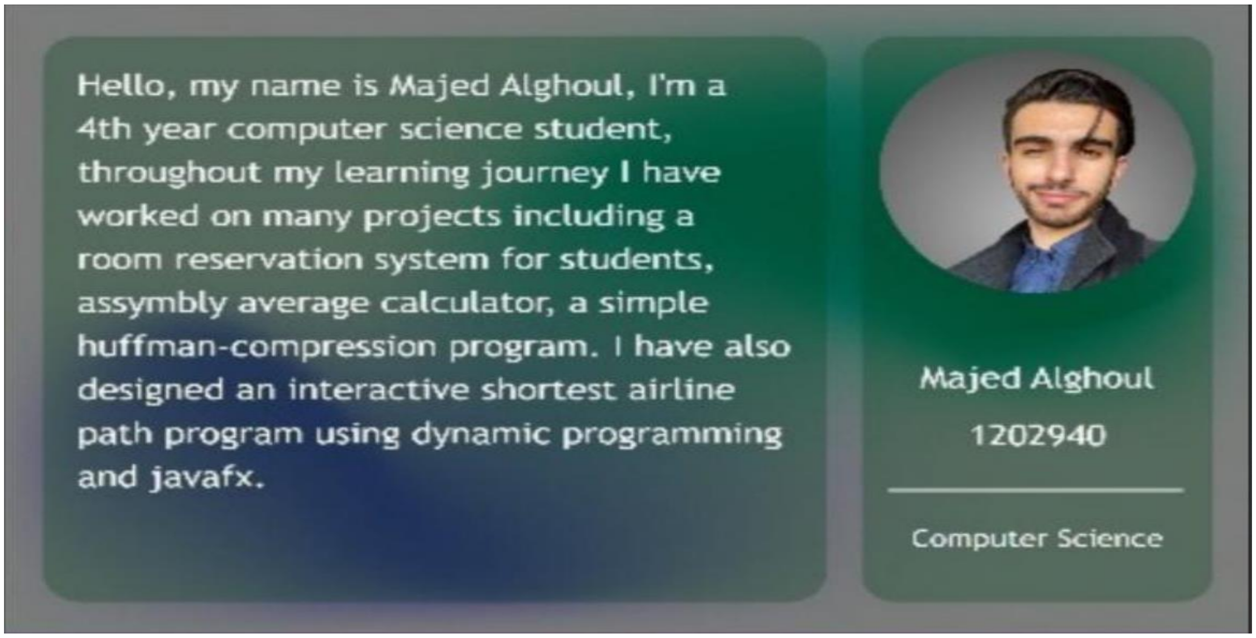
```
<div class="yar">
        <div class="glasss"></div>
       <img id="tests" src="images/x.jpeg" alt="">
        <div class="minicontainer">
            <div class="yar-desc" id="maj-card">Hello, my name is Majed
Alghoul, I'm a 4th year computer
                science student, throughout my learning journey I have
worked on many projects
                including a room reservation system for students, assymbly
average calculator, a simple huffman-compression
                program. I have also designed an interactive shortest
airline path program using dynamic programming and
               javafx.</div>
                <div class="yar-card" id="maj-card">
                   <img id="yarimg" src="images/maj.jpg" alt="">
                    <p class="pp1">Majed Alghoul<br>1202940</p>
                    <hr id="sep3">
                    <p class="pp2">Computer Science</p>
               </div>
        </div>


   </div>
```

```
<div class="yar">
        <div class="glasss"></div>
        <img id="tests" src="images/bkg_mar.jpg" alt="">
        <div class="minicontainer">
            <div class="yar-desc" id="mar-card">Hello, my name is mariam
Abukhdear and I'm a second year cyber
                security student, throughout my learning journey I have worked on
many projects including project of
                estaplishing a special train for passenger,
                and aproject to estaplish a banck that contains all the
information using java language , in
                addition to some security projects.</div>
                <div class="yar-card" id="mar-card">
                    <img id="yarimg" src="images/mar.png" alt="">
                    <p class="pp1">Mariam Abukhdear<br>1222273</p>
                    <hr id="sep3">
                    <p class="pp2">Cyber Security</p>
                </div>
        </div>
    </div>
```

```
<div class="yar">
        <div class="glasss"></div>
        <img id="tests" src="images/bkg.jpg" alt="">
        <div class="minicontainer">
            <div class="yar-desc" id="yar-cardd"> it is Hello, my name is Yara
Obaid and I'm a 3rd year computer
                engineering student, throughout my learning journey I have worked
on many projects
                including an interactive university database to store
                students information, a mini calculator, a complicated
                calculator for large numbers using the linked list data
structure. I have also designed an
                interactive digital menu for a pizza restaurant using OOP and
javafx.</div>

            <div class="yar-card" id="yar-cardd">
                <img id="yarimg" src="images/yar.jpg" alt="">
                <p class="pp1">Yara Obaid<br>1212482</p>
                <hr id="sep3">
                <p class="pp2">Computer Engineering</p>
            </div>
        </div>
    </div>
```

Group members names and IDs (each one in a box)

```
<div class="yar-card" id="maj-card">
                    <img id="yarimg" src="images/maj.jpg" alt="">
                    <p class="pp1">Majed Alghoul<br>1202940</p>
                    <hr id="sep3">
                    <p class="pp2">Computer Science</p>
                </div>
<div class="yar-card" id="mar-card">
                    <img id="yarimg" src="images/mar.png" alt="">
                    <p class="pp1">Mariam Abukhdear<br>1222273</p>
                    <hr id="sep3">
                    <p class="pp2">Cyber Security</p>
                </div>
<div class="yar-card" id="yar-cardd">
                <img id="yarimg" src="images/yar.jpg" alt="">
                <p class="pp1">Yara Obaid<br>1212482</p>
                <hr id="sep3">
                <p class="pp2">Computer Engineering</p>
            </div>
```

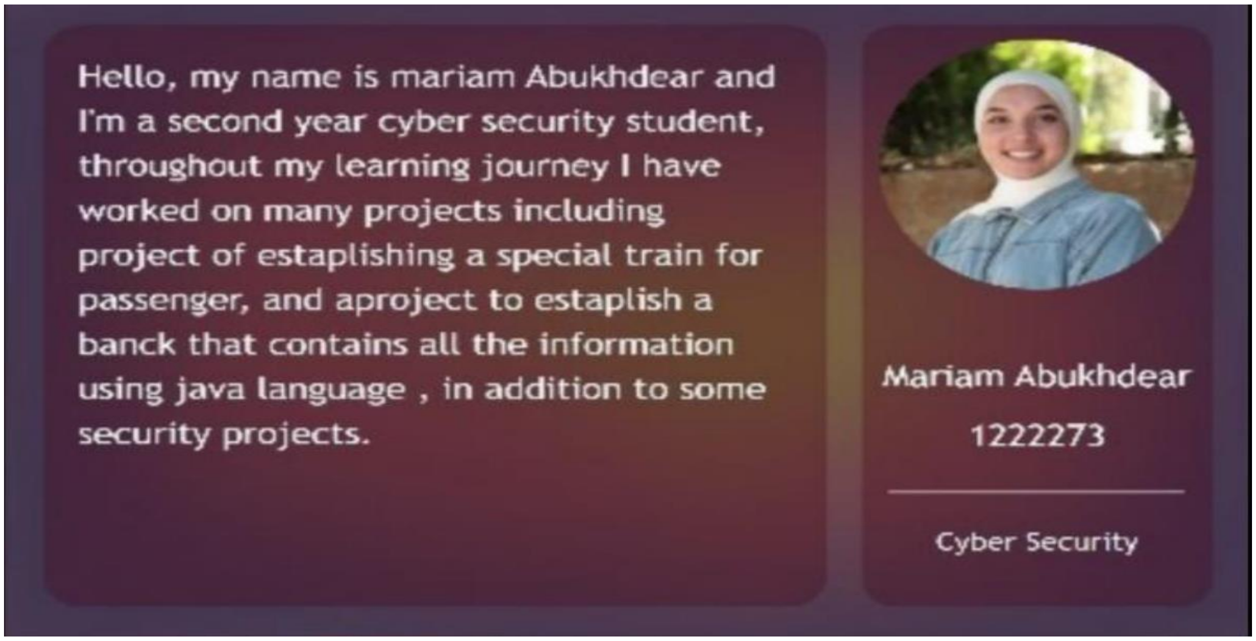


Hello, my name is Majed Alghoul, I'm a 4th year computer science student, throughout my learning journey I have worked on many projects including a room reservation system for students, assymbly average calculator, a simple huffman-compression program. I have also designed an interactive shortest airline path program using dynamic programming and javafx.

Majed Alghoul

1202940

Computer Science

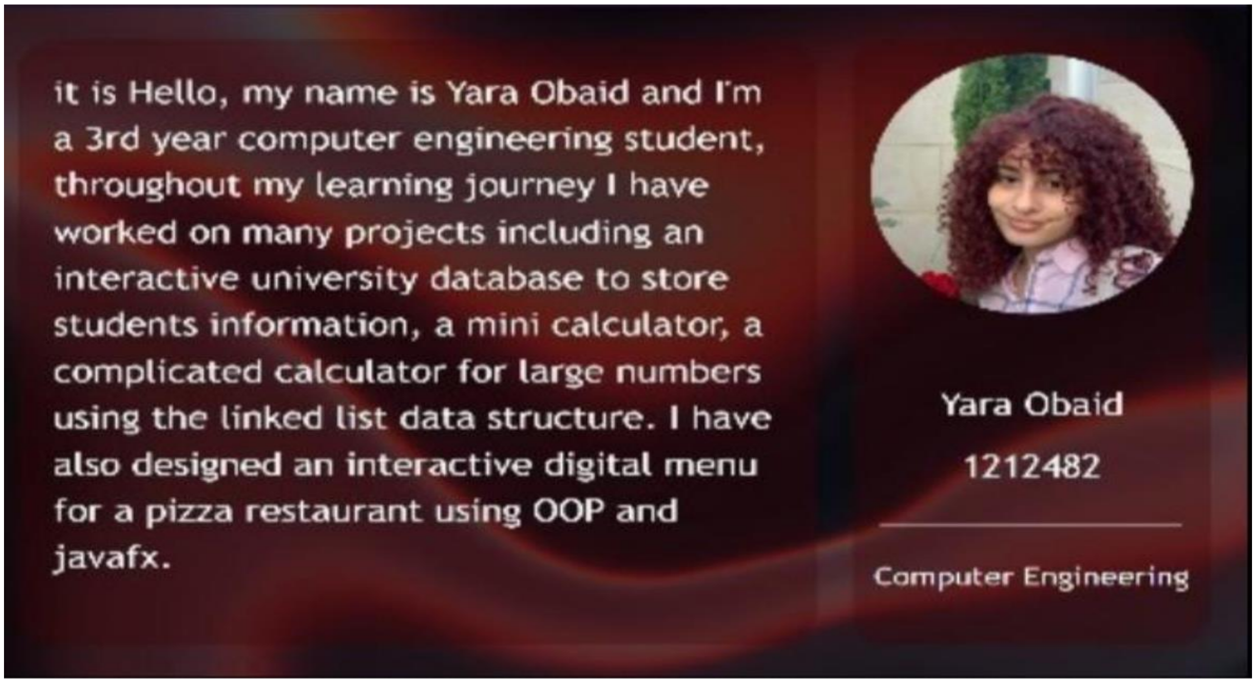


Hello, my name is mariam Abukhdear and I'm a second year cyber security student, throughout my learning journey I have worked on many projects including project of estaplishing a special train for passenger, and aproject to estaplish a banck that contains all the information using java language , in addition to some security projects.

Mariam Abukhdear

1222273

Cyber Security



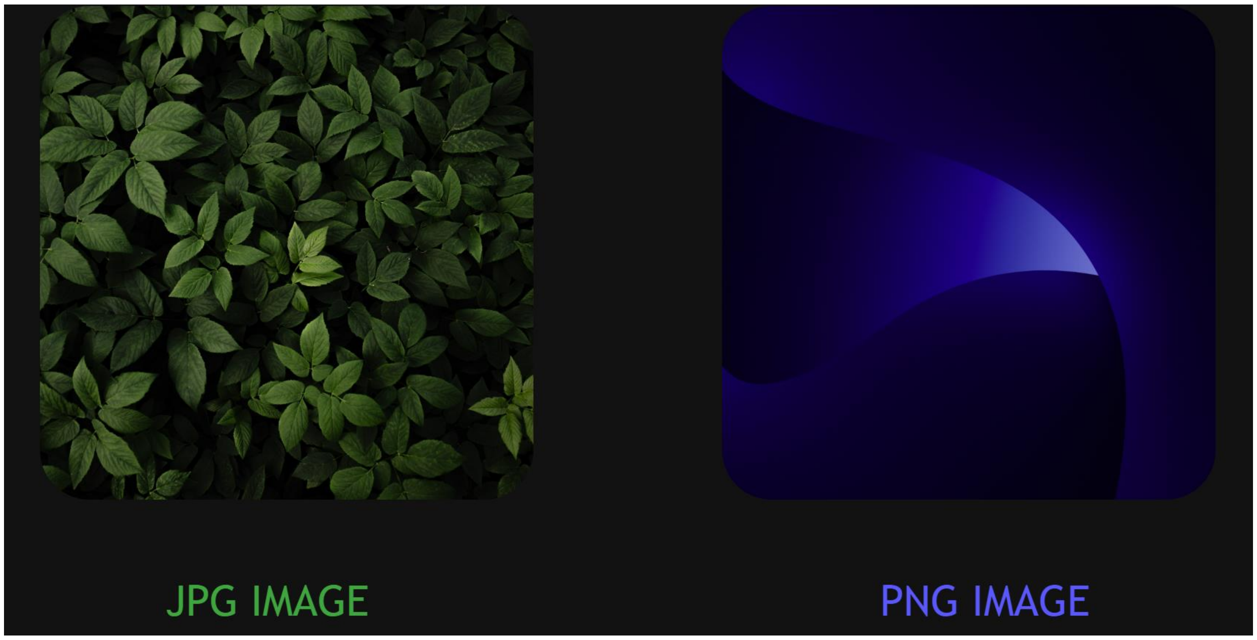
it is Hello, my name is Yara Obaid and I'm a 3rd year computer engineering student, throughout my learning journey I have worked on many projects including an interactive university database to store students information, a mini calculator, a complicated calculator for large numbers using the linked list data structure. I have also designed an interactive digital menu for a pizza restaurant using OOP and javafx.

Yara Obaid

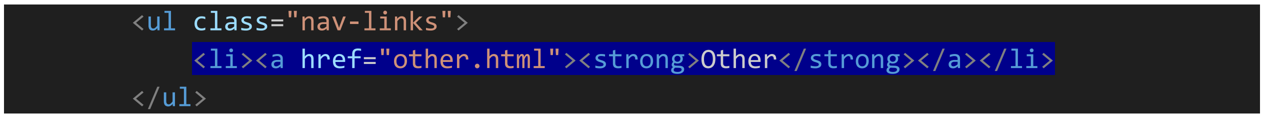
1212482

Computer Engineering

JPG and PNG images

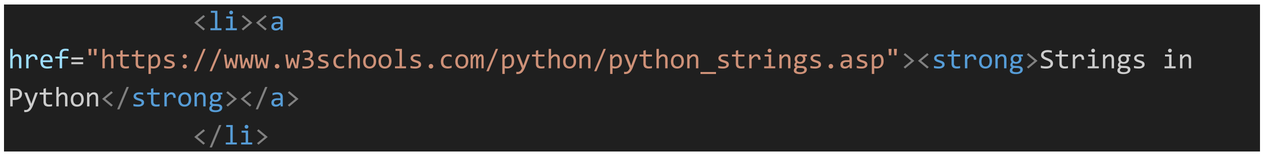


Html local file

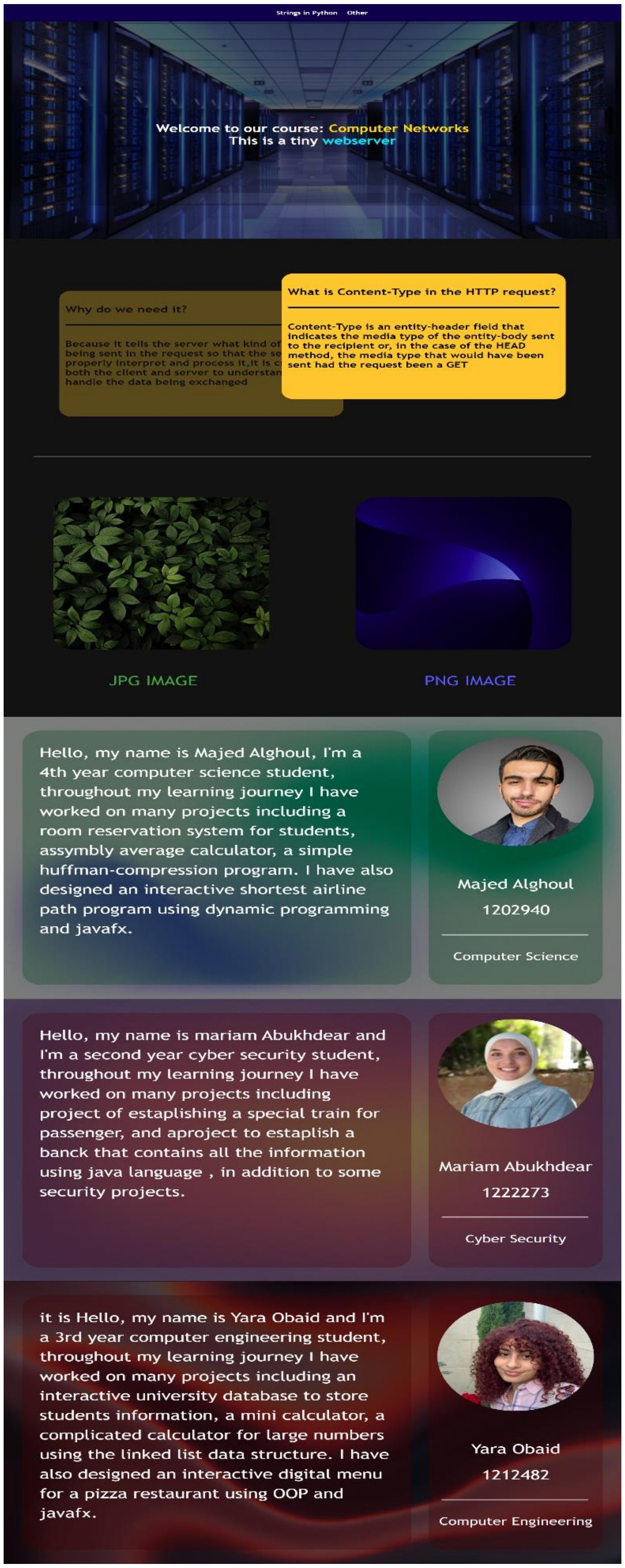


```
        <ul class="nav-links">
            <li><a href="other.html"><strong>Other</strong></a></li>
        </ul>
```

String in python link



```
            <li><a
href="https://www.w3schools.com/python/python_strings.asp"><strong>Strings in
Python</strong></a>
            </li>
```



Welcome to our course: Computer Networks
This is a tiny webserver

Why do we need it?

Because it tells the server what kind of
being sent in the request so that the se
properly interpret and process it,it is c
both the client and server to understan
handle the data being exchanged

What is Content-Type in the HTTP request?

Content-Type is an entity-header field that indicates the media type of the entity-body sent to the recipient or, in the case of the HEAD method, the media type that would have been sent had the request been a GET

JPG IMAGE

PNG IMAGE

Hello, my name is Majed Alghoul, I'm a 4th year computer science student, throughout my learning journey I have worked on many projects including a room reservation system for students, assymbly average calculator, a simple huffman-compression program. I have also designed an interactive shortest airline path program using dynamic programming and javafx.

Majed Alghoul
1202940
Computer Science

Hello, my name is mariam Abukhdear and I'm a second year cyber security student, throughout my learning journey I have worked on many projects including project of estaplishing a special train for passenger, and aproject to estaplish a banck that contains all the information using java language , in addition to some security projects.

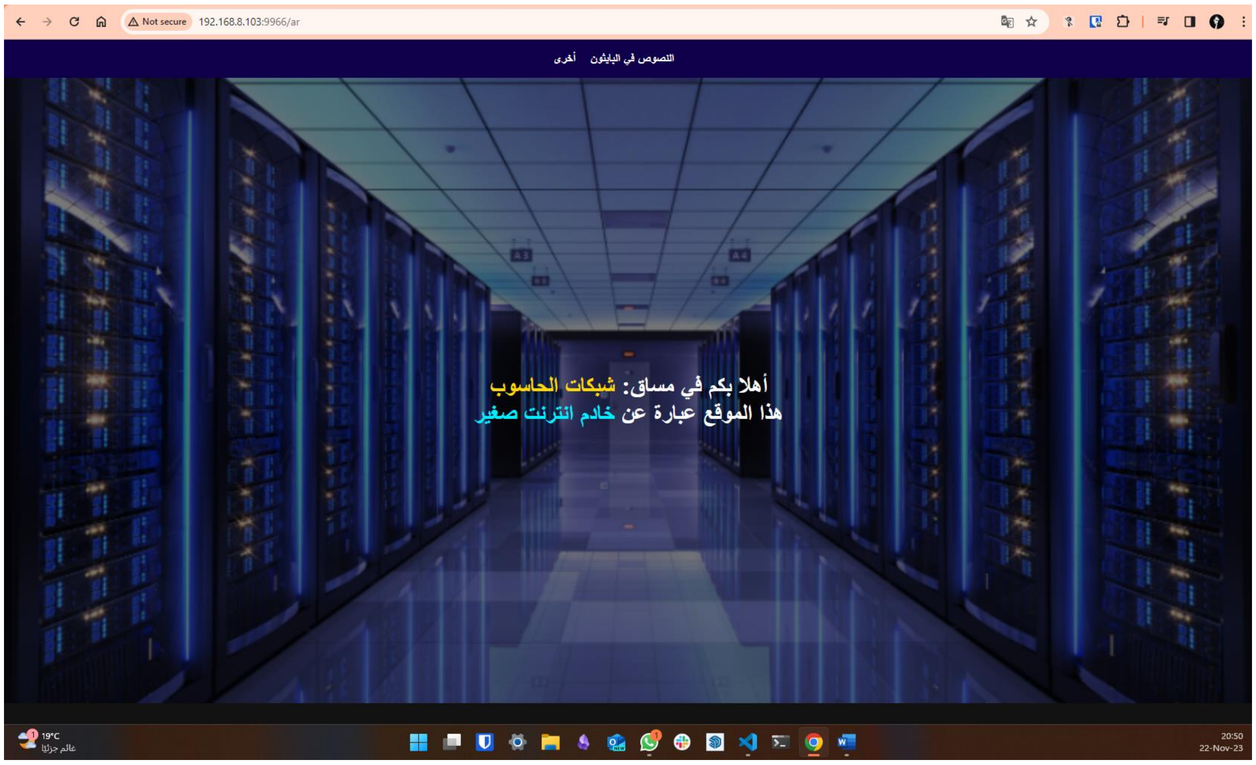
Mariam Abukhdear
1222273
Cyber Security

it is Hello, my name is Yara Obaid and I'm a 3rd year computer engineering student, throughout my learning journey I have worked on many projects including an interactive university database to store students information, a mini calculator, a complicated calculator for large numbers using the linked list data structure. I have also designed an interactive digital menu for a pizza restaurant using OOP and javafx.

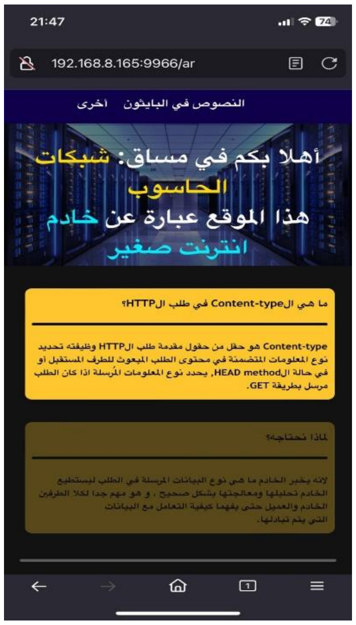
Yara Obaid
1212482
Computer Engineering

If the request is /ar then the server response with main_ar.html which is an Arabic version of main_en.html

```
 elif (request == '/ar'):  # specifying the request info for the main_ar.html
file

            filename = 'main_ar.html'
            filetype = 'text/html'




order = read_file(filename, rtype)
        if order:  # if file exists a 200 ok response is sent
            send_response(csocket, statustype, filetype, order)
        else:  # if the file doesnt exist a call for the error function is sent
with the client socket and both the address and port
            call_error(csocket, address)
```



أهلا بكم في مساق: شبكات الحاسوب
هذا الموقع عبارة عن خادم انترنت صغير



21:47

192.168.8.165:9966/ar

النصوص في البايثون أخرى

أهلا بكم في مساق: شبكات الحاسوب
هذا الموقع عبارة عن خادم انترنت صغير

ما هي الContent-type في طلب الHTTP؟

Content-type هو حقل من حقول مقدمة طلب الHTTP وظيفته تحديد نوع المعلومات المتضمنة في محتوى الطلب المبعوث للطرف المستقبل أو في حالة الHEAD method, يحدد نوع المعلومات المُرسلة اذا كان الطلب مرسل بطريقة GET.

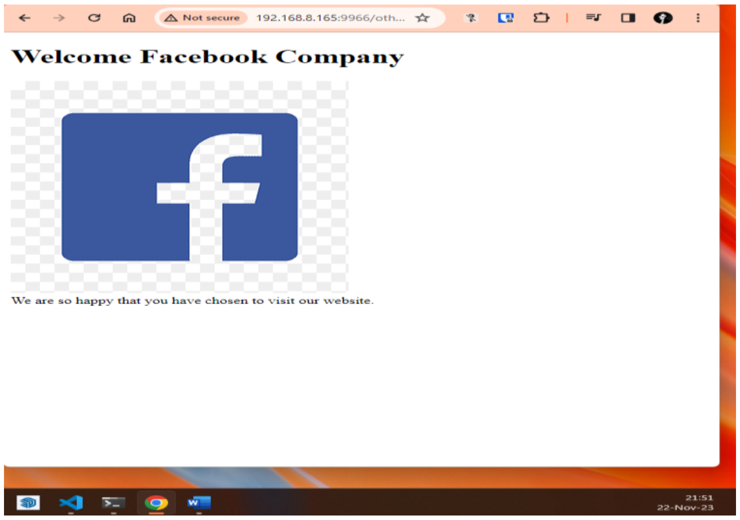
لماذا نحتاجه؟

لانه يخبر الخادم ما هي نوع البيانات المرسلة في الطلب ليستطيع الخادم تحليلها ومعالجتها بشكل صحيح ، و هو مهم جدا لكلا الطرفين الخادم والعميل حتى يفهما كيفية التعامل مع البيانات التي يتم تبادلها.

Same as the main_en.html file, when /ar request the Arabic version of the main html file is sent with all of its links

if the request is an .**html** file then the server should send the requested html file with Content-Type: text/html. You can use any html file.

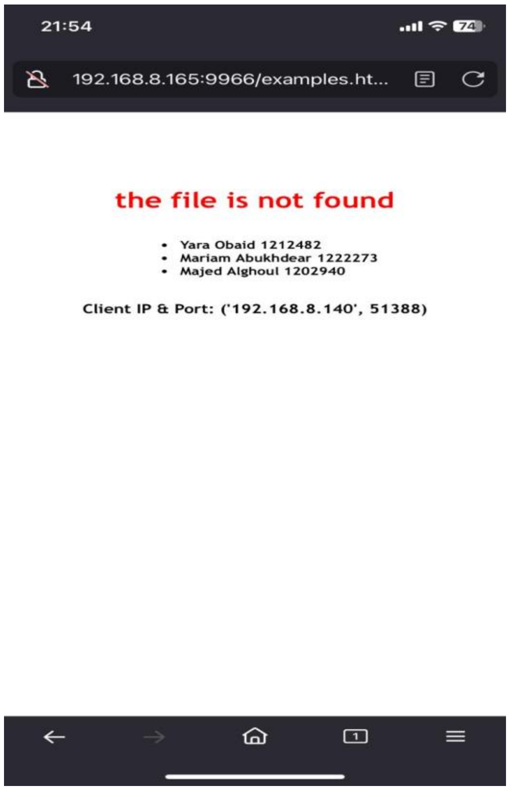
```
elif (request.endswith(".html")) or (request.endswith(".css")):

          filename = request[1:]  # stripping the file path from the first /
          # taking out the file extension from the file path
          filetype = 'text/'+request.split('.')[-1]
```



192.168.8.165:9966/oth...

Welcome Facebook Company

We are so happy that you have chosen to visit our website.


21:54

192.168.8.165:9966/examples.ht...

the file is not found

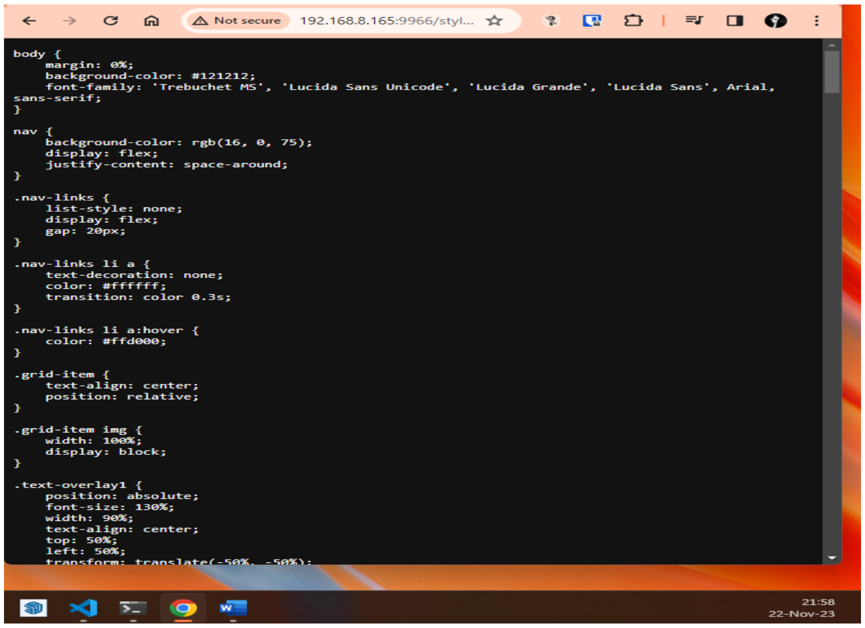
- Yara Obaid 1212482
- Mariam Abukhdear 1222273
- Majed Alghoul 1202940

Client IP & Port: ('192.168.8.140', 51388)

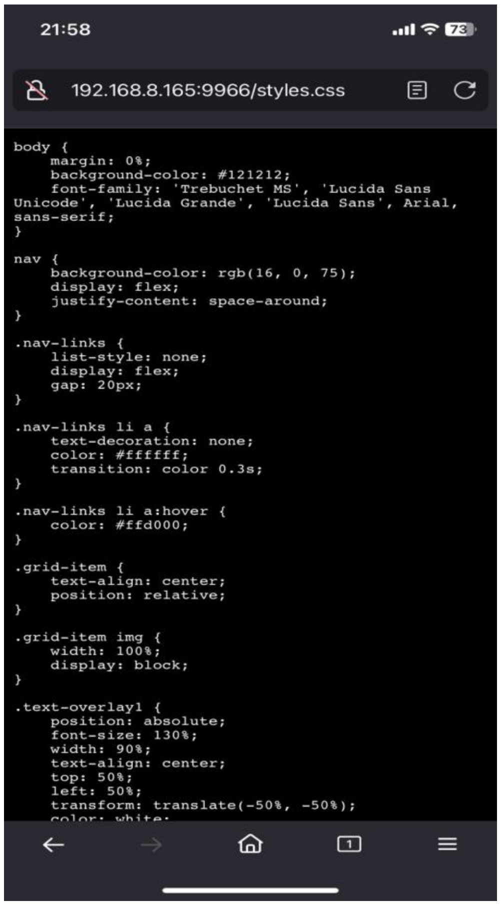
When a file being requested that doesn't exist, like / examples.html the server responds with the 404 file not found error page, and When the file is other .html, the "welcom facebook company" page will appear.

if the request is a **.css** file then the server should send the requested css file with Content-Type: text/css. You can use any CSS file

```
  elif (request.endswith(".html")) or (request.endswith(".css")):

            filename = request[1:]  # stripping the file path from the first /
            # taking out the file extension from the file path
            filetype = 'text/'+request.split('.')[-1]
```



```
body {
    margin: 0%;
    background-color: #121212;
    font-family: 'Trebuchet MS', 'Lucida Sans Unicode', 'Lucida Grande', 'Lucida Sans', Arial,
sans-serif;
}

nav {
    background-color: rgb(16, 0, 75);
    display: flex;
    justify-content: space-around;
}

.nav-links {
    list-style: none;
    display: flex;
    gap: 20px;
}

.nav-links li a {
    text-decoration: none;
    color: #ffffff;
    transition: color 0.3s;
}

.nav-links li a:hover {
    color: #ffd000;
}

.grid-item {
    text-align: center;
    position: relative;
}

.grid-item img {
    width: 100%;
    display: block;
}

.text-overlay1 {
    position: absolute;
    font-size: 130%;
    width: 90%;
    text-align: center;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
```



```
body {
    margin: 0%;
    background-color: #121212;
    font-family: 'Trebuchet MS', 'Lucida Sans
Unicode', 'Lucida Grande', 'Lucida Sans', Arial,
sans-serif;
}

nav {
    background-color: rgb(16, 0, 75);
    display: flex;
    justify-content: space-around;
}

.nav-links {
    list-style: none;
    display: flex;
    gap: 20px;
}

.nav-links li a {
    text-decoration: none;
    color: #ffffff;
    transition: color 0.3s;
}

.nav-links li a:hover {
    color: #ffd000;
}

.grid-item {
    text-align: center;
    position: relative;
}

.grid-item img {
    width: 100%;
    display: block;
}

.text-overlay1 {
    position: absolute;
    font-size: 130%;
    width: 90%;
    text-align: center;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
    color: white;
```

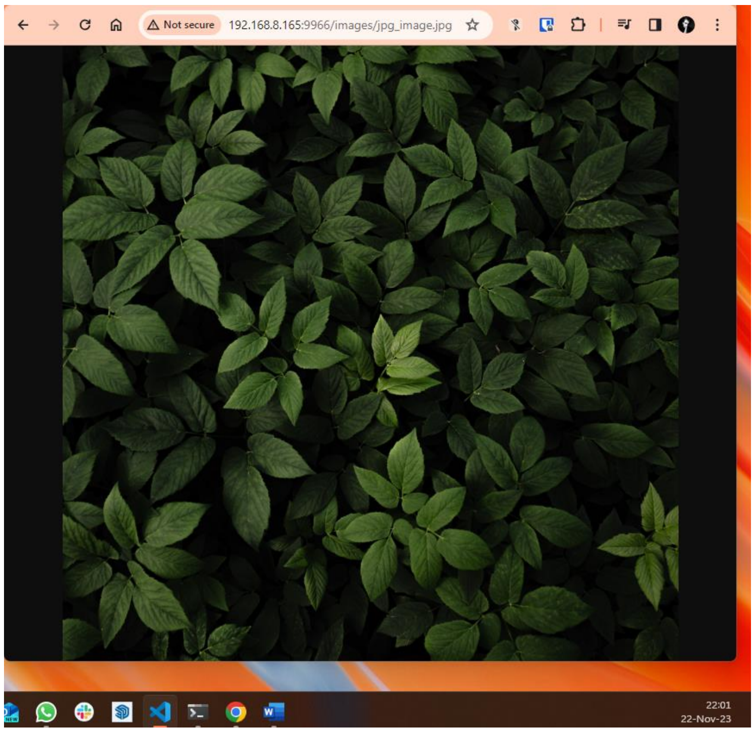
When a css file is being requested, as long as it exists like the styles.css file, the server replies with the appropriate css file

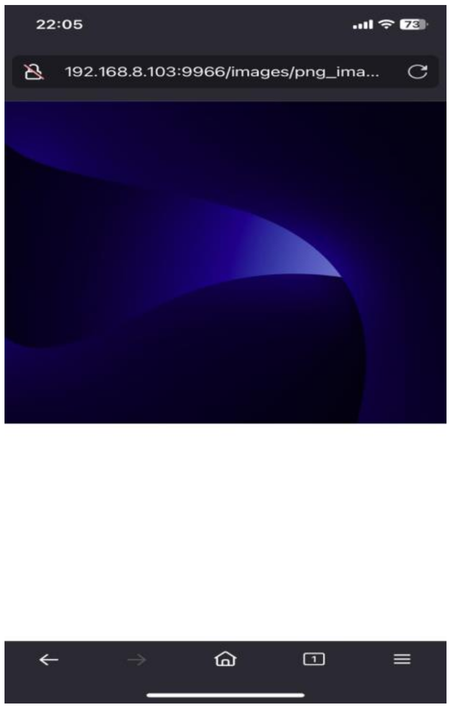
if the request is a **.jpg** then the server should send the jpg image with Content-Type: image/jpeg. You can use any image.

```
 elif request.endswith('.png') or request.endswith('.jpg') or
request.endswith('.jpeg'):

       filename = request[1:]
       filetype = 'image/'+request.split('.')[-1]
       rtype = 'rb'
```

if the request is a .**png** then the server should send the png image with Content-Type: image/png. You can use any image.

```
 elif request.endswith('.png') or request.endswith('.jpg') or
request.endswith('.jpeg'):

       filename = request[1:]
       filetype = 'image/'+request.split('.')[-1]
       rtype = 'rb'
```





When an image file, it being a jpg jpeg or png is requested, as long as they exist like the images/png_image.png or images/jpg_image.png the server responds with the appropriate file.

Use the status code 307 Temporary Redirect to redirect the following

```
        csocket.send('HTTP/1.1 307 Temporary Redirect\r\n'.encode())
```

If the request is /cr then redirect to cornell.edu website

```
if (request == '/cr'):  # redirecting to cornell website

                csocket.send('Location://cornell.edu\r\n'.encode())
```

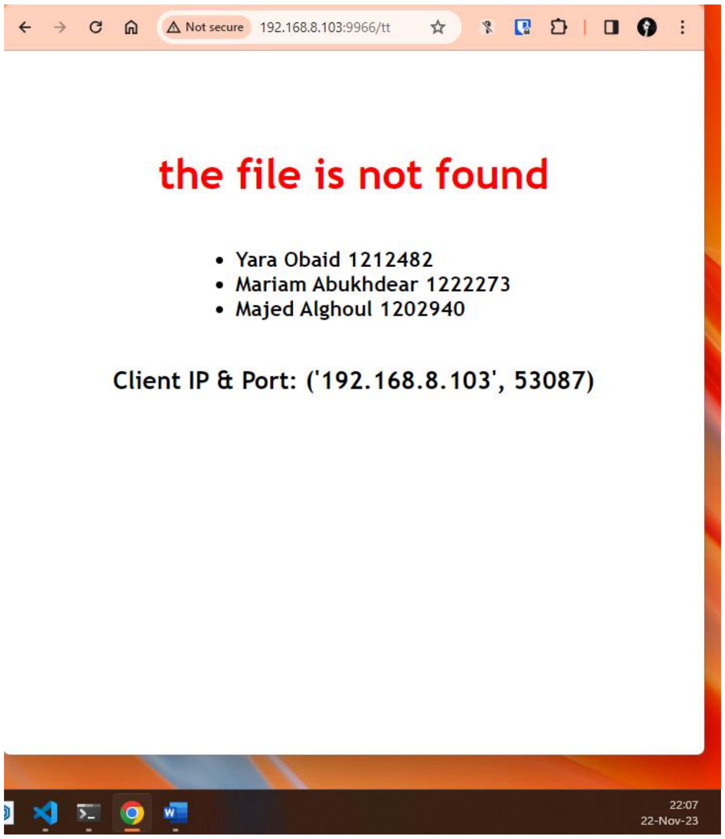
If the request is /so then redirect to stackoverflow.com website

```
 elif (request == '/so'):  # redirecting to stackoverflow website

                csocket.send('Location://stackoverflow.com\r\n'.encode())
```

If the request is /rt then redirect to ritaj website

```
elif (request == '/rt'):  # redirecting to ritaj website

                csocket.send('Location://ritaj.birzeit.edu\r\n'.encode())
```

"HTTP/1.1 404 Not Found" in the response status

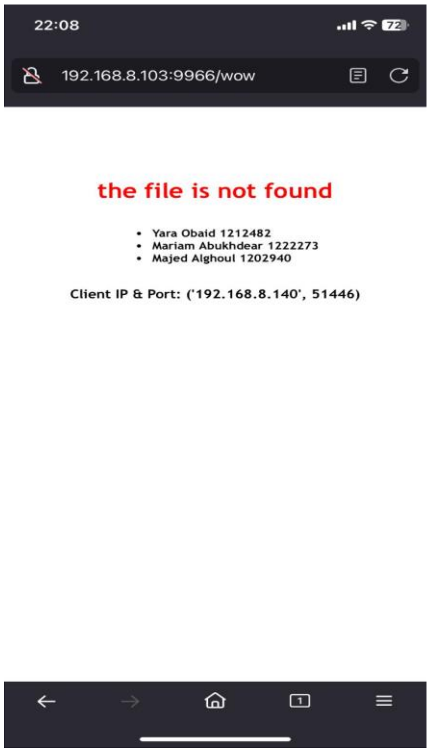


192.168.8.103:9966/tt

the file is not found

- Yara Obaid 1212482
- Mariam Abukhdear 1222273
- Majed Alghoul 1202940

Client IP & Port: ('192.168.8.103', 53087)



192.168.8.103:9966/wow

the file is not found

- Yara Obaid 1212482
- Mariam Abukhdear 1222273
- Majed Alghoul 1202940

Client IP & Port: ('192.168.8.140', 51446)

When a query being requested like a random location, like /home or /wow the server

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="styles.css">
    <title>Error 404</title>
</head>
<body style="background-color: white;">
    <div id="containerrr">
        <h1 style="color: red;">the file is not found</h1>
        <ul>
            <li>
                <strong>Yara Obaid 1212482</strong>
            </li>
            <li>
                <strong>Mariam Abukhdear 1222273</strong>
            </li>
            <li>
                <strong>Majed Alghoul 1202940</strong>
            </li>
        </ul>
        <h3>Client IP & Port: {info}</h3>
    </div>

</body>
</html>
```

responds with a 404 file not found error page

```python
def call_error(csocket, address):  # a function responsible for handling error
page calling
    # calling the read function giving it the error.html file with reading type
of r
    temp = read_file("error.html", "r")
    # formatting the read error.html file to insert the address and port in a
location specified by {info}
    order = temp.format(info=address)
    # sending a 404 response with the formatted error.html file with the send
function
    send_response(csocket, '404 Not Found', 'text/html', order)
```