

Faculty of Engineering & Technology
Electrical & Computer Engineering Department
Computer Networks ENCS3320
Project 1
Socket Programming and Web Development

Group Members

Student name	Student ID	Section
Doaa Hatu	1211088	3
Raghad Owaisi	1200710	3
Hala Qurt	1220062	2

Instructor's: AbdelKarim Awwad, Mohammed Jubran

Date: 28/11/2023

Abstract

In this project we are supposed to learn many different things, we are going to run different commands on “Command Prompt” app to track packets, and also to read different DNS messages using Wireshark.

Then, we will use socket programming to implement a simple client-server system, since they contact with each other using TCP connections.

Finally, we are going to create a web server (using socket programming) to handle different HTTP requests from the client, and to handle several files. In addition to creating a web page using HTML and CSS, which contains the group member’s information.

Table of Contents

Abstract	2
Table of Contents	3
Table of figures	5
Part 1:	6
 1- In your own words, what are ping, traceroute, nslookup, and telnet (write one sentence for each one)	6
• Ping: Internet Package Grouper. it's the most widely used utility tool to troubleshooting. help to test the Host/server reachability, Internet connection, Network interface card and to test DNS issues.....	6
• Traceroute: It is used to see the exact path the data is taking on its way the destination by sending Internet Control Message Protocol (ICMP) echo packets to the destination.	6
• NSLOOKUP: it's a utility used to find the Ip address or DNS record of a given host name, and to find the attached domain for an IP address.	6
• Telnet: a Network protocol that provides a two-way collaborative abd text-based communication between two machines, and allows to connect to remote computers over a TCP/IP Network.	6
 2- Make sure that your computer is connected to the internet and then run the following commands:.....	7
1- Ping a device in the same network, e.g. from a laptop to a smartphone	7
2- ping www.cornell.edu.....	8
3- racert www.cornell.edu	9
4- nslookup www.cornell.edu	10
3- use wireshark to capture some DNS messages.	11
Part 2: Socket programming.....	12
• Server Code.....	12
• Client Code.....	14
Part 3: Web Server.....	18
• Web Server Code	18
0- From rfce2616, what is Content-Type in the HTTP request and why do we need it?	21
1- 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.	21
• 2- If the request is /ar then the server response with main_ar.html which is an Arabic version of main_en.html	24
3- 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.	25

4- 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	26
5- If the request is a .png then the server should send the png image with Content-Type: image/png. You can use any image.	27
6- If the request is a .jpg then the server should send the jpg image with Content-Type: image/jpeg. You can use any image.	28
7- Use the status code 307 Temporary Redirect to redirect the following	29
8- If the request is wrong or the file doesn't exist the server should return a simple HTML webpage that contains (Content-Type: text/html)	32
Codes.....	33
English Version.....	33
HTML (main_en.html)	33
style.css Code	40
Web server:.....	45
Arabic Version.....	49
HTML code (main_ar.html):.....	49
styleAr.css code:.....	56
Web Server	61
Testing the website from another device.....	64

Table of figures

<i>Figure 1: Part 1, part 2.1.</i>	7
<i>Figure 2: Part 1, part 2.2.</i>	8
<i>Figure 3: Part 1, part 2.3.</i>	9
<i>Figure 4:Part 1, part 2.4.</i>	10
<i>Figure 5: Part 1, part 3.</i>	11
<i>Figure 6:Part2, Server code.</i>	12
<i>Figure 7:Part2, Client code.</i>	14
<i>Figure 8: Part2, Running Server code.</i>	15
<i>Figure 9:Part2, Running Client code.</i>	16
<i>Figure 10: Part2, Entering Valid student ID.</i>	16
<i>Figure 11: Part2, Entering Invalid student ID.</i>	17
<i>Figure 12:part 3,webserver code.</i>	20
<i>Figure 13:main_en.</i>	21
<i>Figure 14:HTTP request for english</i>	23
<i>Figure 15:main_ar</i>	24
<i>Figure 16:HTTP request for arabic</i>	25
<i>Figure 17: English html file</i>	25
<i>Figure 18:HTTP request</i>	25
<i>Figure 19: Style code</i>	26
<i>Figure 20:Style HTTP request</i>	26
<i>Figure 21:png image</i>	27
<i>Figure 22:HTTP request for png</i>	27
<i>Figure 23:jpg image</i>	28
<i>Figure 24:HTTP request for jpg</i>	28
<i>Figure 25:HTTP request for cornell</i>	29
<i>Figure 26:cornell.edu</i>	29
<i>Figure 27:stachoverflow.com</i>	30
<i>Figure 28:HTTP request for stackoverflow</i>	30
<i>Figure 29:ritaj.edu</i>	31
<i>Figure 30:HTTP request for ritaj</i>	31
<i>Figure 31:request is wrong or the file doesn't exist</i>	32
<i>Figure 32:wrong HTTP request</i>	32
<i>Figure 33:English html code</i>	33
<i>Figure 34:error page HTML code</i>	39
<i>Figure 35:Style css code</i>	40
<i>Figure 36:English web server</i>	45
<i>Figure 37:html file</i>	47
<i>Figure 38:W3schools file</i>	48
<i>Figure 39:html Arabic code</i>	49
<i>Figure 40:Arabic style code</i>	56
<i>Figure 41:Arabic webserver</i>	61
<i>Figure 42:html Arabic file</i>	63
<i>Figure 43:Test from another device</i>	64

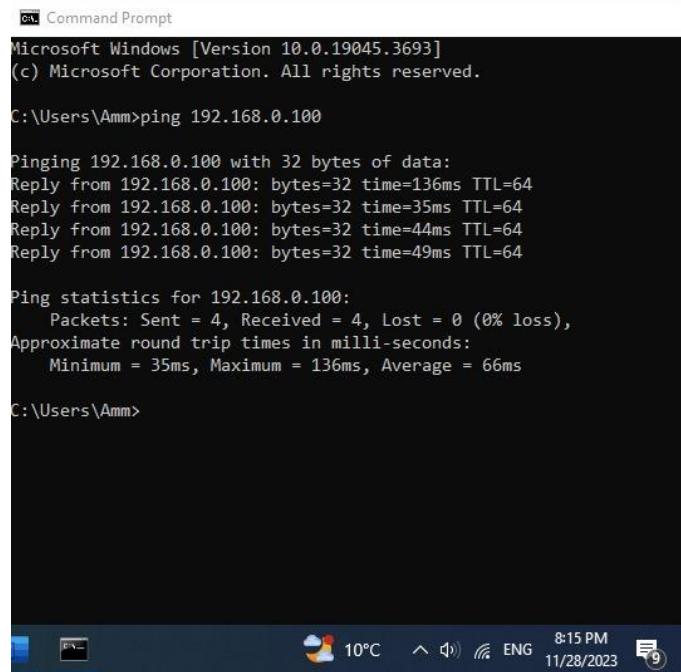
Part 1:

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

- **Ping:** Internet Package Grouper. it's the most widely used utility tool to troubleshooting. help to test the Host/server reachability, Internet connection, Network interface card and to test DNS issues.
- **Traceroute:** It is used to see the exact path the data is taking on its way the destination by sending Internet Control Message Protocol (ICMP) echo packets to the destination.
- **NSLOOKUP:** it's a utility used to find the Ip address or DNS record of a given host name, and to find the attached domain for an IP address.
- **Telnet:** a Network protocol that provides a two-way collaborative abd text-based communication between two machines, and allows to connect to remote computers over a TCP/IP Network.

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



```
Command Prompt
Microsoft Windows [Version 10.0.19045.3693]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amm>ping 192.168.0.100

Pinging 192.168.0.100 with 32 bytes of data:
Reply from 192.168.0.100: bytes=32 time=136ms TTL=64
Reply from 192.168.0.100: bytes=32 time=35ms TTL=64
Reply from 192.168.0.100: bytes=32 time=44ms TTL=64
Reply from 192.168.0.100: bytes=32 time=49ms TTL=64

Ping statistics for 192.168.0.100:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 35ms, Maximum = 136ms, Average = 66ms

C:\Users\Amm>
```

Figure 1: Part 1, part 2.1.

We used cmd to test the ping command by sending four data packets to the targeted Ip address (we used the Ip of a cell phone) the output of this command contained the bytes of the data packets we sent which it was (32 bytes).

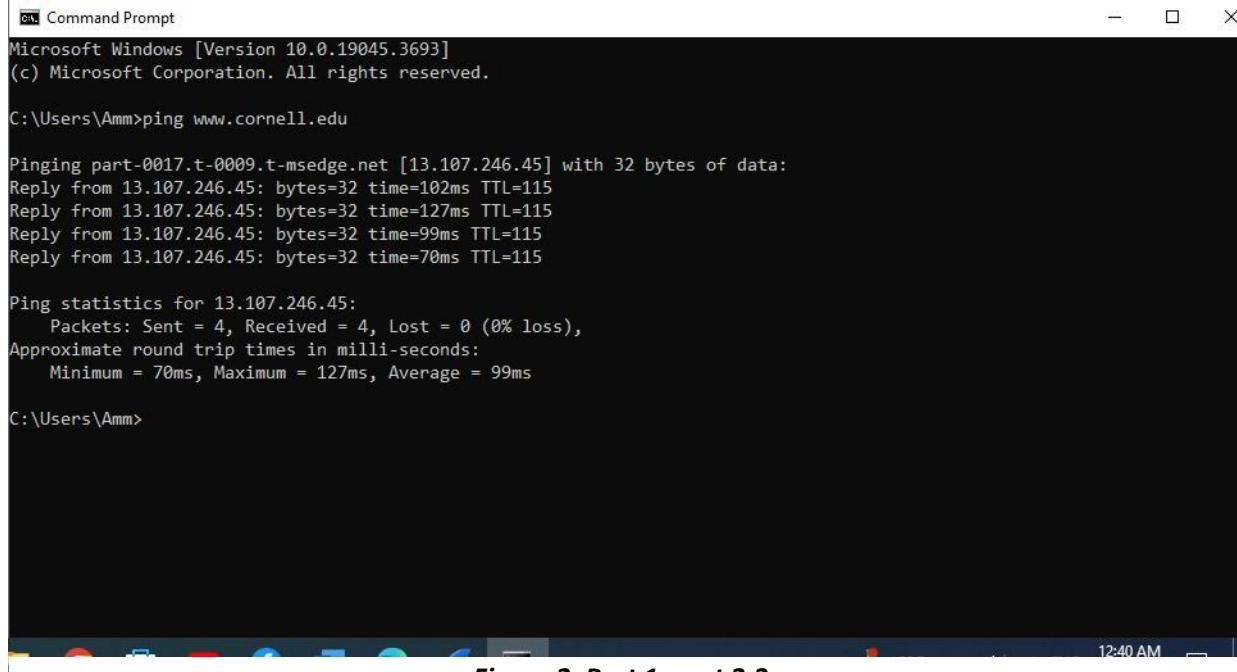
each one of the packets we sent had three parameters to describe it:

- 1- Size of each Packet: measured by bytes.
- 2- Time needed for each packet to send and received measured by milliseconds.
- 3- Time To Live (TTL): number of hops the packet is permitted to travel before being discarded.

In the above screenshot you can see that all the four packets we sent was successfully sent and received, the average round-trip time was 66 milliseconds.

2- ping www.cornell.edu

We used Cmd to preform ping www.cornell.edu , we saw the round-trip time (in



```
Command Prompt
Microsoft Windows [Version 10.0.19045.3693]
(c) Microsoft Corporation. All rights reserved.

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

Pinging part-0017.t-0009.t-msedge.net [13.107.246.45] with 32 bytes of data:
Reply from 13.107.246.45: bytes=32 time=102ms TTL=115
Reply from 13.107.246.45: bytes=32 time=127ms TTL=115
Reply from 13.107.246.45: bytes=32 time=99ms TTL=115
Reply from 13.107.246.45: bytes=32 time=70ms TTL=115

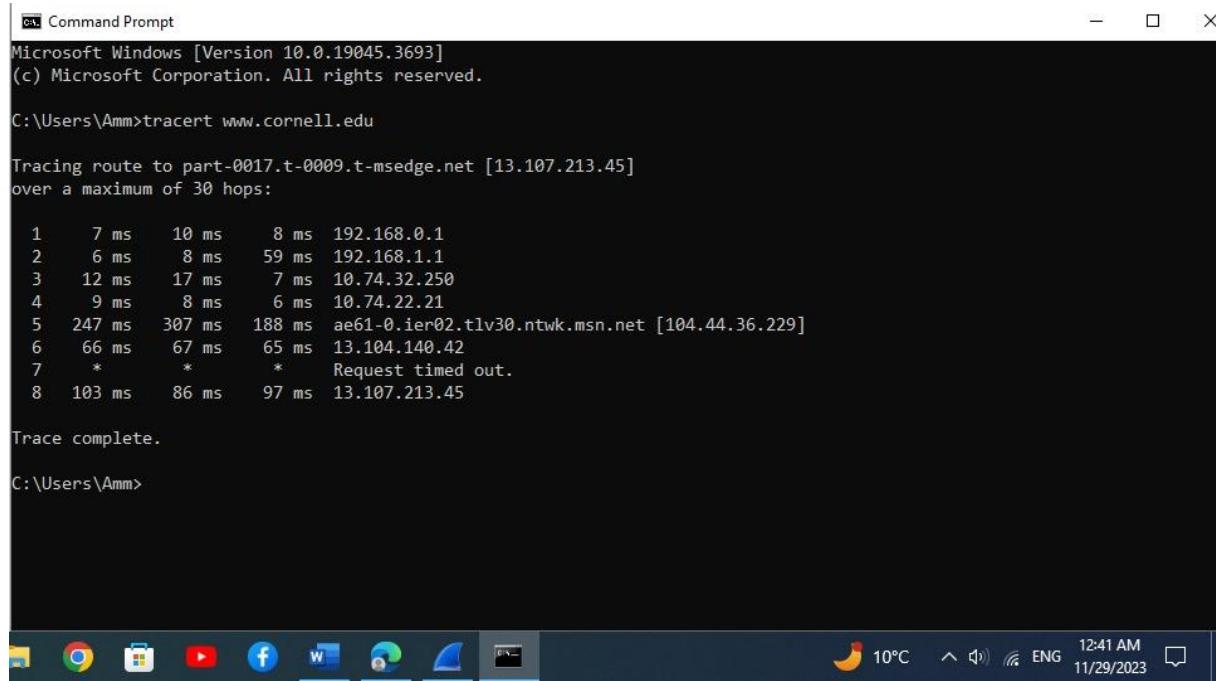
Ping statistics for 13.107.246.45:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 70ms, Maximum = 127ms, Average = 99ms

C:\Users\Amm>
```

Figure 2: Part 1, part 2.2.

milliseconds) for each packet , the Ip address of "part-0017.t-0009.t-msedge.net" is 13.107.246.45 . it also shows the replay of each message containing the size of each packet in bytes, the time from sending to receiving this packet and TTL which is the number of nods each packet has to pass through.

3- racert www.cornell.edu



```
Command Prompt
Microsoft Windows [Version 10.0.19045.3693]
(c) Microsoft Corporation. All rights reserved.

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

Tracing route to part-0017.t-0009.t-msedge.net [13.107.213.45]
over a maximum of 30 hops:

 1   7 ms    10 ms    8 ms  192.168.0.1
 2   6 ms    8 ms   59 ms  192.168.1.1
 3   12 ms   17 ms    7 ms  10.74.32.250
 4    9 ms    8 ms    6 ms  10.74.22.21
 5  247 ms   307 ms  188 ms  ae61-0.ier02.t1v30.ntwk.msn.net [104.44.36.229]
 6   66 ms   67 ms   65 ms  13.104.140.42
 7    *      *       * Request timed out.
 8  103 ms   86 ms   97 ms  13.107.213.45

Trace complete.

C:\Users\Amm>
```

Figure 3: Part 1, part 2.3.

We used Cmd to perform tracert, the first line shows the target IP address or domain being traced and its corresponding IP address. The Column Headers:

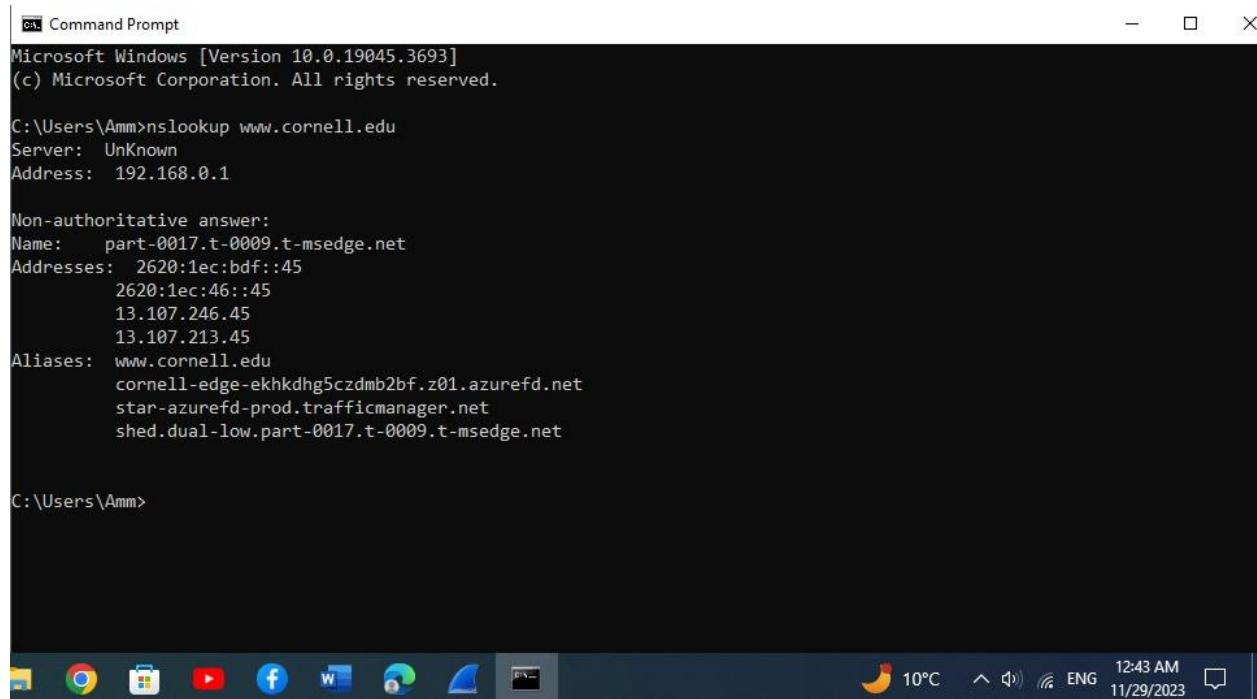
Hop: The number of the hop in the route.

Hostname (or IP address): The IP address or hostname of the router at that hop.

Time (ms): Round-trip time in milliseconds for the packet to reach that hop and back.

Rows represent a hop along the route. The first hop is typically our local router or gateway. Subsequent hops are intermediate routers or switches that the packet passes through. Times in milliseconds represent the round-trip time for a packet to reach that hop and return. The three times shown are the result of sending three separate packets to that hop.

4- nslookup www.cornell.edu



```
Command Prompt
Microsoft Windows [Version 10.0.19045.3693]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amm>nslookup www.cornell.edu
Server: UnKnown
Address: 192.168.0.1

Non-authoritative answer:
Name: part-0017.t-0009.t-msedge.net
Addresses: 2620:1ec:bdf::45
           2620:1ec:46::45
           13.107.246.45
           13.107.213.45
Aliases: www.cornell.edu
          cornell-edge-ekhkdhg5czdmb2bf.z01.azurefd.net
          star-azurefd-prod.trafficmanager.net
          shed.dual-low.part-0017.t-0009.t-msedge.net

C:\Users\Amm>
```

The screenshot shows a Microsoft Windows Command Prompt window titled "Command Prompt". The window displays the output of the "nslookup www.cornell.edu" command. The output includes the server information (UnKnown at 192.168.0.1), a non-authoritative answer for the name "www.cornell.edu" which resolves to "part-0017.t-0009.t-msedge.net", and additional aliases such as "cornell-edge-ekhkdhg5czdmb2bf.z01.azurefd.net" and "star-azurefd-prod.trafficmanager.net". The Command Prompt prompt is "C:\Users\Amm>". Below the window, the Windows taskbar is visible with various icons and system status indicators.

Figure 4:Part 1, part 2.4.

We used Cmd to perform nslookup command, the output was shown as follow:

Server: The DNS server that provided the resolution. In this example, it's "UnKnown" with an IP address of "192.168.0.1".

Non-authoritative answer: This indicates that the information was obtained from a DNS server that is not the ultimate authority for the domain.

Name: The resolved domain name. In this case, "www.cornell.edu" resolves to "part-0017.t-0009.t-msedge.net".

Addresses: The IP addresses associated with the resolved domain.

Aliases: Additional domain names associated with the resolved IP addresses.

3- use wireshark to capture some DNS messages.

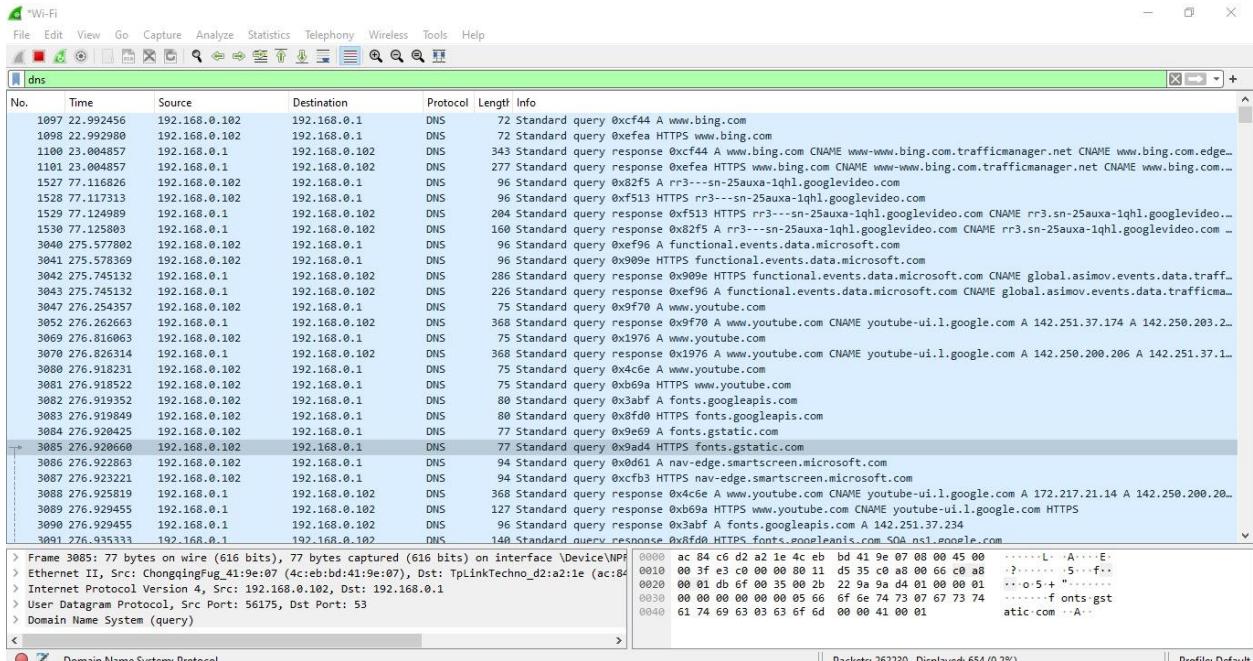


Figure 5: Part 1, part 3.

When capturing DNS (Domain Name System) messages in Wireshark, we can see DNS request and response packets, DNS record types, DNS Queries for local domains, DNS flags, etc.

DNS protocol translates human readable domain names into IP addresses.

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

The ping results alone, which measure Round Trip Time and network latency, do not provide information about the geographical location of the server associated with "www.cornell.edu." Determining the server's location requires additional methods such as geolocation services or querying the IP address using WHOIS services. By extracting the IP address (13.107.246.62) from the CMD prompt window and checking its approximate location on websites like iplocation.net or ipinfo.io, the results suggest a potential association with the USA, specifically Washington, or Canada, more precisely Quebec. However, it's crucial to emphasize that these findings offer a possible location, and the exact source cannot be definitively verified solely based on the ping results.

Part 2: Socket programming

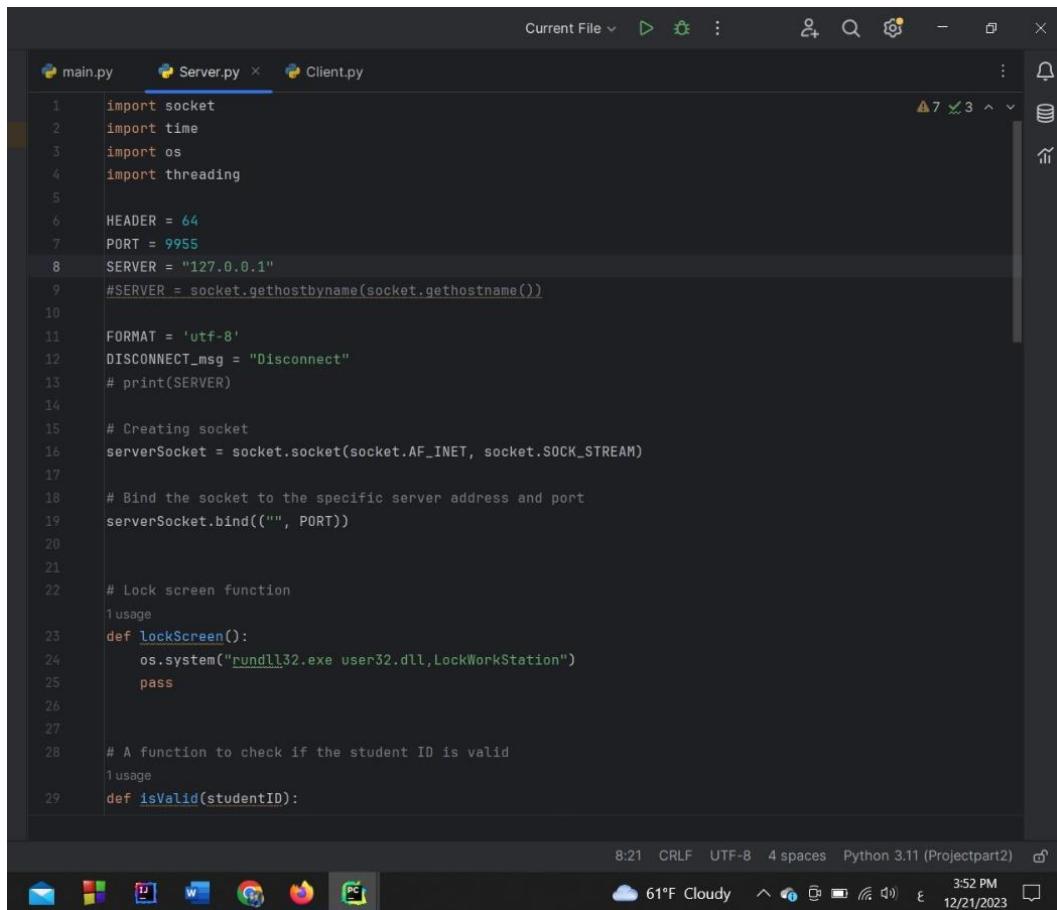
Using socket programming, implement TCP client and server applications in go, python, java or C. The server should listen on port 9955. The server waits for a message from a client.

If the message is with one of the students ID, the sever should do the following:

1. display a message on the server side that the OS will lock screen after 10 seconds
2. send a message to the client that the sever will lock screen after 10 seconds
3. then wait 10 seconds
4. then call a function lock the screen of the operating system (windows or Linux or MAC).

Any student ID of the group member should work. Any other student number or any text should display an error message on the server side and no lock screen should be done.

- **Server Code**



The screenshot shows a code editor window with three tabs: main.py, Server.py (which is the active tab), and Client.py. The code in Server.py is as follows:

```
1 import socket
2 import time
3 import os
4 import threading
5
6 HEADER = 64
7 PORT = 9955
8 SERVER = "127.0.0.1"
9 #SERVER = socket.gethostname()
10
11 FORMAT = 'utf-8'
12 DISCONNECT_msg = "Disconnect"
13 # print(SERVER)
14
15 # Creating socket
16 serverSocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
17
18 # Bind the socket to the specific server address and port
19 serverSocket.bind("", PORT)
20
21
22 # Lock screen function
23 #usage
24 def lockScreen():
25     os.system("rundll32.exe user32.dll,LockWorkStation")
26
27
28 # A function to check if the student ID is valid
29 #usage
30 def isValid(studentID):
```

The status bar at the bottom of the code editor shows the following information: 8:21 CRLF UTF-8 4 spaces Python 3.11 (Projectpart2) 3:52 PM 12/21/2023

Figure 6:Part2, Server code.

A screenshot of a code editor window titled "Server.py". The code implements a server that checks student IDs and communicates with clients. It includes a function to validate student IDs and another to handle client connections, sending a message back to the client if the ID is valid.

```
28     # A function to check if the student ID is valid
29     usage
30     def isValid(studentID):
31         validStudentIDs = {"1211088", "1200710", "1220062"}
32         return studentID in validStudentIDs
33
34     # Function to handle communication with the client
35     # Function to handle communication with the client
36     usage
37     def handleClient(conn, addr):
38         print(f"[New Connection]{addr} connected")
39         connected = True
40
41         while connected:
42             try:
43                 message_length = int(conn.recv(HEADER).decode(FORMAT))
44             except ValueError:
45                 print(f"[{addr}] Error: Invalid message length.")
46                 continue
47
48             message = conn.recv(message_length).decode(FORMAT)
49
50             # Check if the message is a valid ID
51             if isValid(message):
52                 # Display the message in the server's side
53                 print(f"[{addr}] Valid student ID received: {message} OS will lock screen after 10 seconds")
54
55             # Send the message to the client
56             conn.send(b"Server received a valid student ID. OS will lock screen after 10 seconds")
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84 D 84 if __name__ == "__main__":
85     print("Server is starting..")
86     start()
```

A screenshot of a code editor window titled "Server.py". The code has been modified to include a sleep operation and a lock screen feature. It also includes a start() function to begin the server. The code is annotated with comments explaining its purpose.

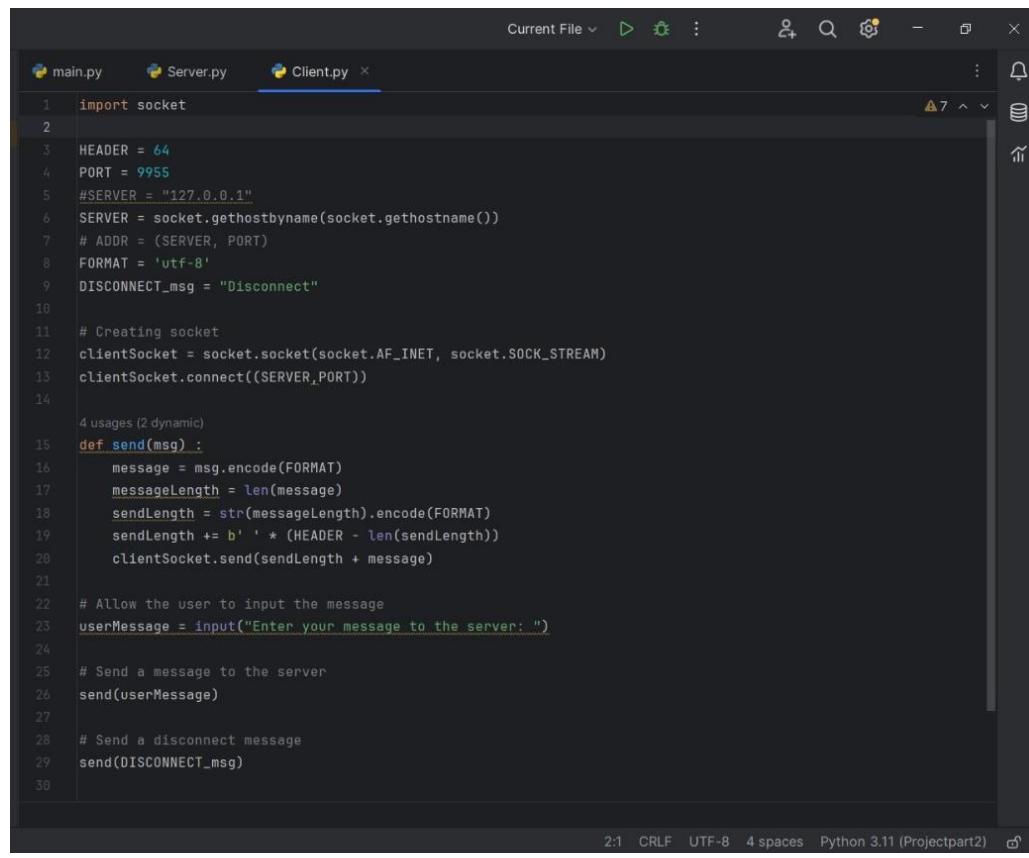
```
57     # Wait for 10 seconds
58     time.sleep(10)
59
60     # Lock the screen
61     lockScreen()
62
63     elif message == DISCONNECT_MSG:
64         connected = False
65
66     else:
67         print(f"[{addr}] Invalid message received: {message}")
68         # Tell the client that it is an invalid message
69         conn.send(b"Error: Invalid message.")
70
71     conn.close()
72
73     # Function to start the server
74     usage
75     def start():
76         # Listening for incoming connections
77         serverSocket.listen(1)
78         print(f"Server is listening on {SERVER}")
79         while True:
80             conn, addr = serverSocket.accept()
81             # Create a new thread for each connected client
82             thread = threading.Thread(target=handleClient, args=(conn, addr))
83             thread.start()
84 D 84 if __name__ == "__main__":
85     print("Server is starting..")
86     start()
```

Explanation:

In this code, we implemented a simple server using socket programming in python that listens on port 9955, and accepts connections from client (using handleClient() function, the server will tell that it is connected to the client, this is called Handshaking, since we are using TCP connection). The server will take the client message and check if it is a valid student id (one of the 3 ids of us) or not using isValid() function, if yes, it will send a message to the client that the operating system will lock the screen after 10 seconds, if not, it will tell the client that it is an invalid id, so no event will occur.

For each connected client, a thread will be created, so we imported the thread library.

- **Client Code**



The screenshot shows a code editor window with three tabs at the top: main.py, Server.py, and Client.py. The Client.py tab is active, displaying the following Python code:

```
1 import socket
2
3 HEADER = 64
4 PORT = 9955
5 #SERVER = "127.0.0.1"
6 SERVER = socket.gethostname()
7 # ADDR = (SERVER, PORT)
8 FORMAT = 'utf-8'
9 DISCONNECT_msg = "Disconnect"
10
11 # Creating socket
12 clientSocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
13 clientSocket.connect((SERVER,PORT))
14
15 def send(msg) :
16     message = msg.encode(FORMAT)
17     messageLength = len(message)
18     sendLength = str(messageLength).encode(FORMAT)
19     sendLength += b' ' * (HEADER - len(sendLength))
20     clientSocket.send(sendLength + message)
21
22 # Allow the user to input the message
23 userMessage = input("Enter your message to the server: ")
24
25 # Send a message to the server
26 send(userMessage)
27
28 # Send a disconnect message
29 send(DISCONNECT_msg)
30
```

The code implements a client socket that connects to a server on port 9955. It defines a send function to encode messages and includes logic to allow the user to input a message and send it to the server. A disconnect message is also sent.

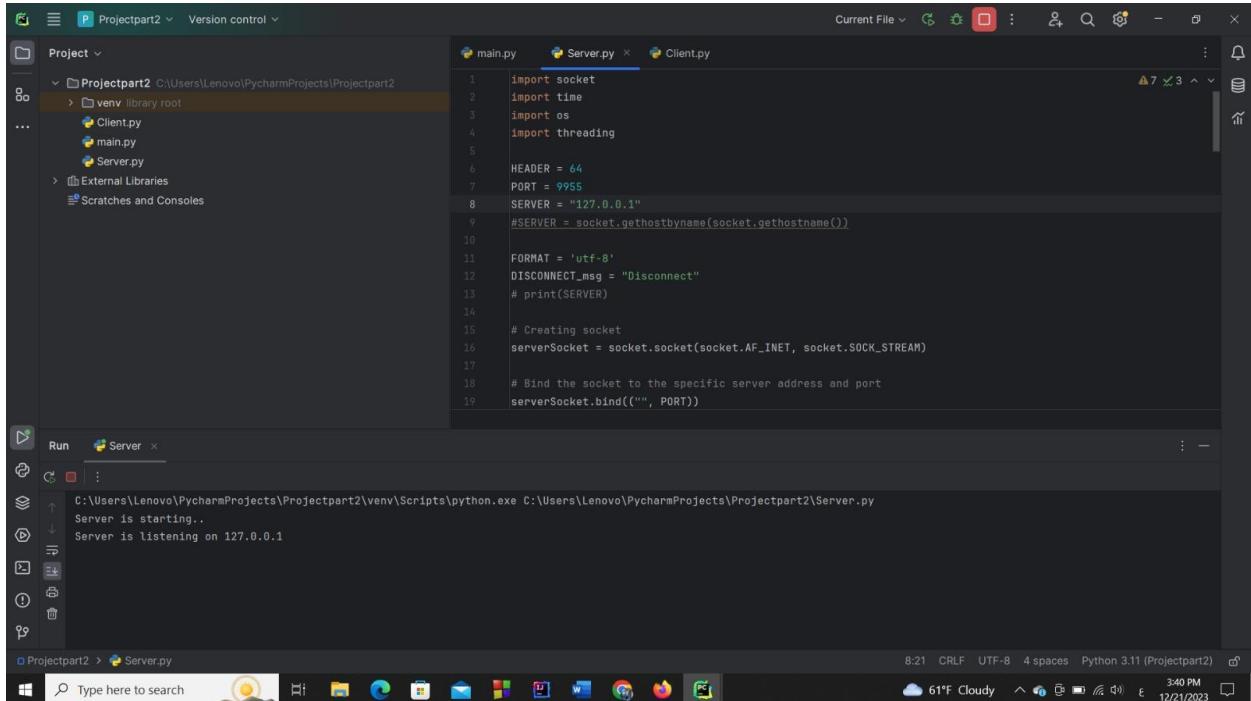
Figure 7:Part2, Client code.

Explanation:

In this code, we implemented a simple client that starts a TCP connection with the server using sockets. It contains a function that handles message sending, and determines the message length, it also allows the user to write the message to the server, then it closes the TCP connection.

Both client and server must be connected to the same port.

- **Running results**



```
Project part2 C:\Users\Lenovo\PycharmProjects\Projectpart2
  venv library root
  ...
  Client.py
  main.py
  Server.py
  External Libraries
  Scratches and Consoles

main.py  Server.py  Client.py
1 import socket
2 import time
3 import os
4 import threading
5
6 HEADER = 64
7 PORT = 9955
8 SERVER = "127.0.0.1"
9 #SERVER = socket.gethostname()
10
11 FORMAT = 'utf-8'
12 DISCONNECT_msg = "Disconnect"
13 # print(SERVER)
14
15 # Creating socket
16 serverSocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
17
18 # Bind the socket to the specific server address and port
19 serverSocket.bind("", PORT)

Run  Server
C:\Users\Lenovo\PycharmProjects\Projectpart2\venv\Scripts\python.exe C:\Users\Lenovo\PycharmProjects\Projectpart2\Server.py
Server is starting...
Server is listening on 127.0.0.1

8:21  CRLF  UTF-8  4 spaces  Python 3.11 (Projectpart2)  3:40 PM  61°F Cloudy  12/21/2023
```

Figure 8: Part2, Running Server code.

When we run the server the code, it prints that the server is starting, after it accepts the connection, it prints that the server is listening.

The screenshot shows the PyCharm IDE interface. The project structure on the left includes files main.py, Client.py, and Server.py. The main.py file is currently selected. The Client.py tab is open in the editor, displaying Python code for a socket client. The terminal window at the bottom shows the command `python C:\Users\Lenovo\PycharmProjects\Projectpart2\Client.py` being run, followed by the prompt "Enter your message to the server:". The status bar at the bottom right indicates the date and time as 12/21/2023, 3:40 PM.

```

1 import socket
2
3
4 HEADER = 64
5 PORT = 9955
6 SERVER = "127.0.0.1"
7 ADDR = (SERVER, PORT)
8 FORMAT = 'utf-8'
9 DISCONNECT_MSG = "Disconnect"
10
11 # Creating socket
12 clientSocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
13 clientSocket.connect((SERVER, PORT))
14
15 4 usages (2 dynamic)
16 def send(msg):
17     message = msg.encode(FORMAT)
18     messageLength = len(message)

```

Figure 9:Part2, Running Client code.

When running the client code, it asks the user to enter the message to the server

The screenshot shows the PyCharm IDE interface. The project structure on the left includes files main.py, Client.py, and Server.py. The main.py file is currently selected. The Server.py tab is open in the editor, displaying Python code for a socket server. The terminal window at the bottom shows the command `python C:\Users\Lenovo\PycharmProjects\Projectpart2\Server.py` being run, followed by the output: "Server is starting..", "Server is listening on 127.0.0.1", "[New Connection]('192.168.56.1', 62690) connected", and "[('192.168.56.1', 62690)] Valid student ID received: 1211088 OS will lock screen after 10 seconds". The status bar at the bottom right indicates the date and time as 12/21/2023, 3:41 PM.

```

1 import socket
2
3
4 HEADER = 64
5 PORT = 9955
6 SERVER = "127.0.0.1"
7 ADDR = (SERVER, PORT)
8 FORMAT = 'utf-8'
9 DISCONNECT_MSG = "Disconnect"
10
11 # Creating socket
12 clientSocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
13 clientSocket.connect((SERVER, PORT))
14
15 4 usages (2 dynamic)
16 def send(msg):
17     message = msg.encode(FORMAT)
18     messageLength = len(message)

```

Figure 10: Part2, Entering Valid student ID.

When I entered one of our student IDs, the server says that I enter a valid student ID so it will lock the screen after 10 seconds. This happens after the server accepts the client connection (connected).

The screenshot shows the PyCharm IDE interface. The project structure on the left includes files like Client.py, main.py, and Server.py. The main editor window displays the code for Client.py. The run tab is selected, showing the command: C:\Users\Lenovo\PycharmProjects\Projectpart2\venv\Scripts\python.exe C:\Users\Lenovo\PycharmProjects\Projectpart2\Server.py. The output tab shows the server's response: "Server is listening on 127.0.0.1", "[New Connection]('192.168.56.1', 62705) connected", and "[('192.168.56.1', 62705)] Invalid message received: 1211099". The status bar at the bottom indicates Python 3.11 (Projectpart2).

```
import socket
HEADER = 64
PORT = 9955
#SERVER = "127.0.0.1"
# SERVER = socket.gethostname()
# ADDR = (SERVER, PORT)
FORMAT = 'utf-8'
DISCONNECT_msg = "Disconnect"
# Creating socket
clientSocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
clientSocket.connect((SERVER, PORT))

4 usages (2 dynamic)
def send(msg):
    message = msg.encode(FORMAT)
    messageLength = len(message)
```

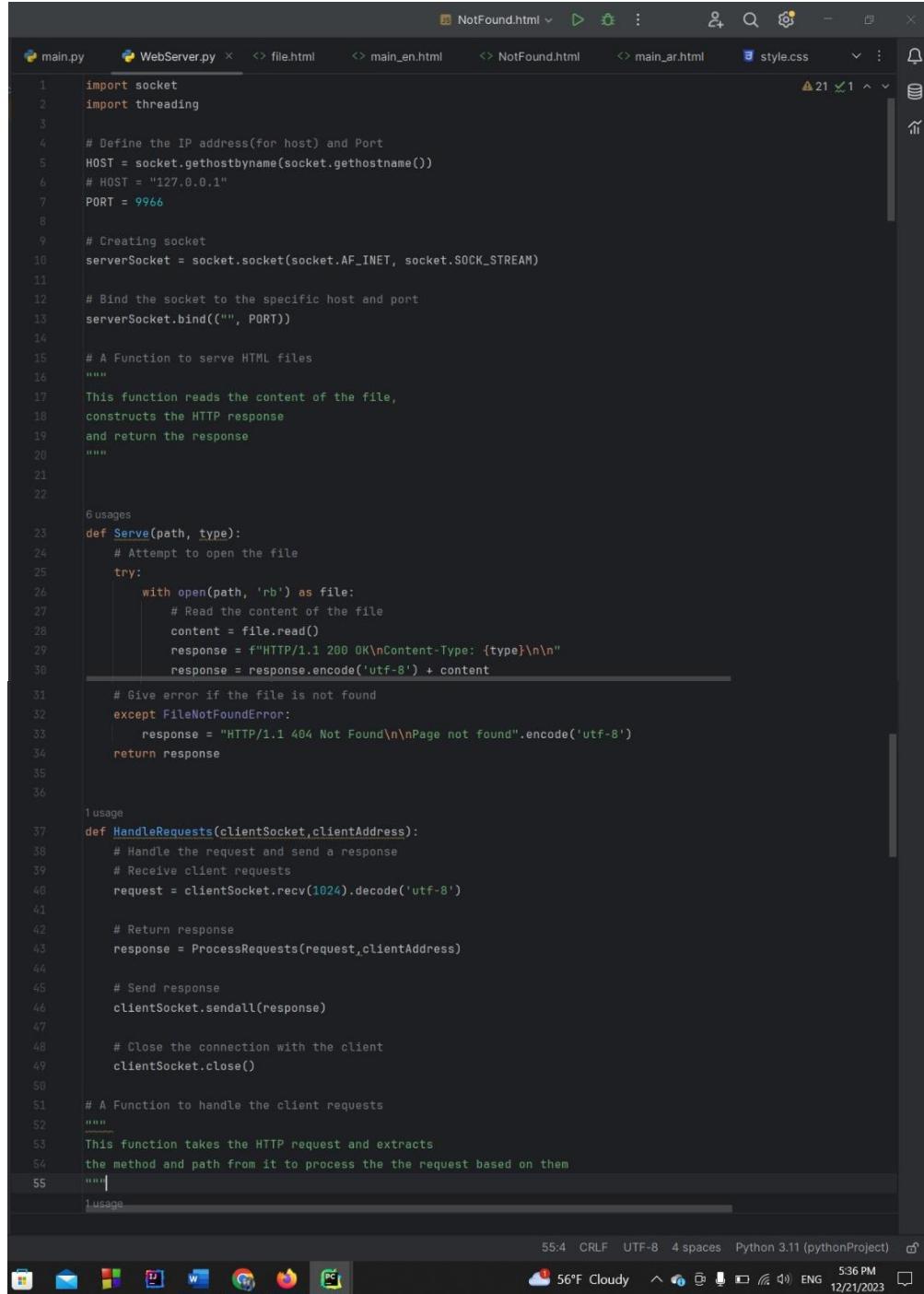
Figure 11: Part2, Entering Invalid student ID.

When I entered an invalid student ID from the client, the server will give a message that the message received is invalid.

Part 3: Web Server

Using socket programming, implement a simple but a complete web server in go, python, java or C that is listening on port 9966.

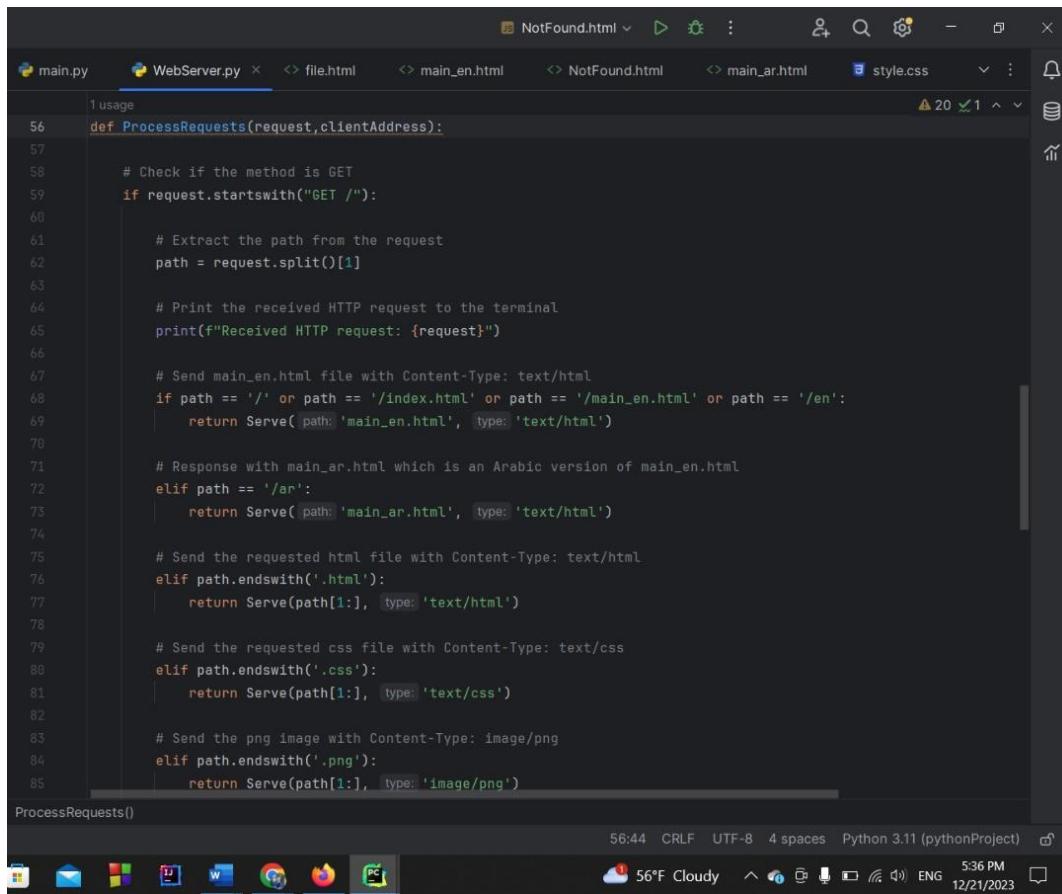
- **Web Server Code**



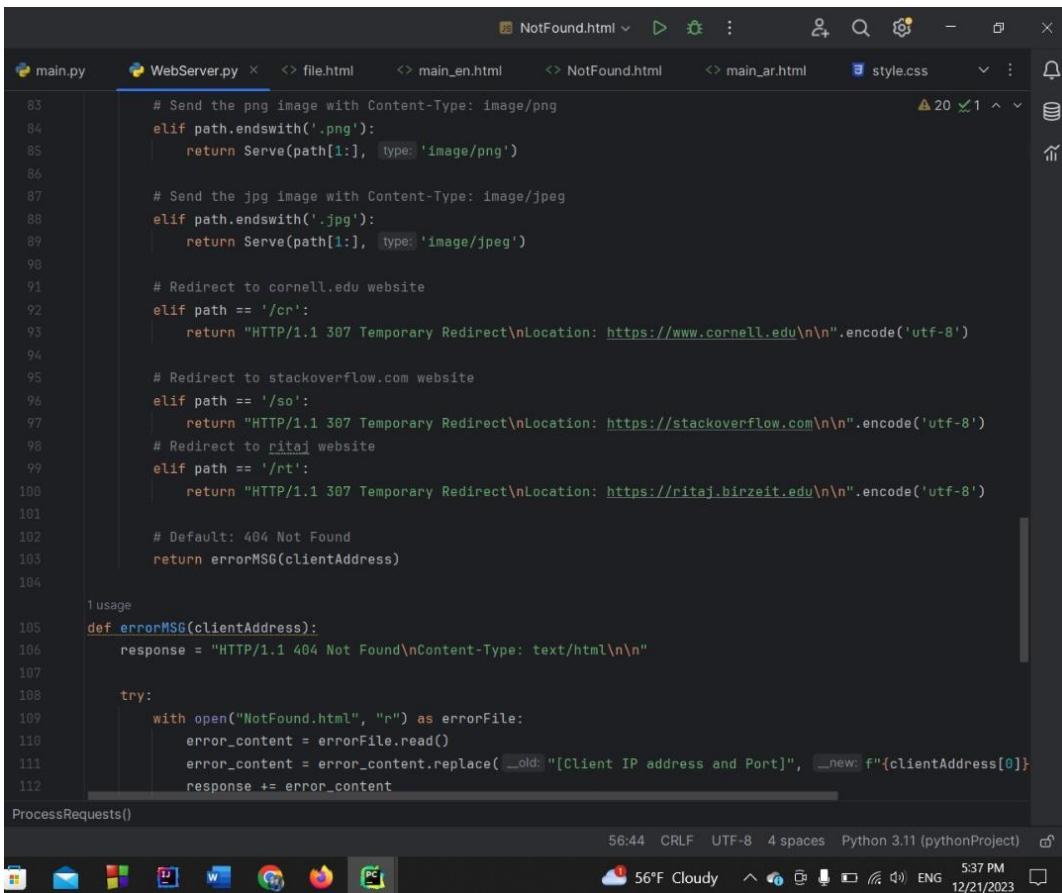
The screenshot shows a code editor window with several tabs at the top: main.py, WebServer.py (which is the active tab), file.html, main_en.html, NotFound.html, main_ar.html, and style.css. The code in the WebServer.py tab is as follows:

```
1 import socket
2 import threading
3
4 # Define the IP address(for host) and Port
5 HOST = socket.gethostname()
6 # HOST = "127.0.0.1"
7 PORT = 9966
8
9 # Creating socket
10 serverSocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
11
12 # Bind the socket to the specific host and port
13 serverSocket.bind("", PORT)
14
15 # A Function to serve HTML files
16 """
17 This function reads the content of the file,
18 constructs the HTTP response
19 and return the response
20 """
21
22
23 def Serve(path, type):
24     # Attempt to open the file
25     try:
26         with open(path, 'rb') as file:
27             # Read the content of the file
28             content = file.read()
29             response = f"HTTP/1.1 200 OK\nContent-Type: {type}\n\n"
30             response += response.encode('utf-8') + content
31
32         # Give error if the file is not found
33     except FileNotFoundError:
34         response = "HTTP/1.1 404 Not Found\n\nPage not found".encode('utf-8')
35
36
37 def HandleRequests(clientSocket, clientAddress):
38     # Handle the request and send a response
39     # Receive client requests
40     request = clientSocket.recv(1024).decode('utf-8')
41
42     # Return response
43     response = ProcessRequests(request, clientAddress)
44
45     # Send response
46     clientSocket.sendall(response)
47
48     # Close the connection with the client
49     clientSocket.close()
50
51 # A Function to handle the client requests
52 """
53 This function takes the HTTP request and extracts
54 the method and path from it to process the the request based on them
55 """
56
57
```

The status bar at the bottom shows: 55:4 CRLF UTF-8 4 spaces Python 3.11 (pythonProject) 5:36 PM 56°F Cloudy ENG 12/21/2023



```
1 usage
56 def ProcessRequests(request,clientAddress):
57
58     # Check if the method is GET
59     if request.startswith("GET /"):
60
61         # Extract the path from the request
62         path = request.split()[1]
63
64         # Print the received HTTP request to the terminal
65         print(f"Received HTTP request: {request}")
66
67         # Send main_en.html file with Content-Type: text/html
68         if path == '/' or path == '/index.html' or path == '/main_en.html' or path == '/en':
69             return Serve(path, 'main_en.html', type='text/html')
70
71         # Response with main_ar.html which is an Arabic version of main_en.html
72         elif path == '/ar':
73             return Serve(path, 'main_ar.html', type='text/html')
74
75         # Send the requested html file with Content-Type: text/html
76         elif path.endswith('.html'):
77             return Serve(path[1:], type='text/html')
78
79         # Send the requested css file with Content-Type: text/css
80         elif path.endswith('.css'):
81             return Serve(path[1:], type='text/css')
82
83         # Send the png image with Content-Type: image/png
84         elif path.endswith('.png'):
85             return Serve(path[1:], type='image/png')
86
87         # Send the jpg image with Content-Type: image/jpeg
88         elif path.endswith('.jpg'):
89             return Serve(path[1:], type='image/jpeg')
90
91         # Redirect to cornell.edu website
92         elif path == '/cr':
93             return "HTTP/1.1 307 Temporary Redirect\nLocation: https://www.cornell.edu\n\n".encode('utf-8')
94
95         # Redirect to stackoverflow.com website
96         elif path == '/so':
97             return "HTTP/1.1 307 Temporary Redirect\nLocation: https://stackoverflow.com\n\n".encode('utf-8')
98         # Redirect to ritaJ website
99         elif path == '/rt':
100            return "HTTP/1.1 307 Temporary Redirect\nLocation: https://ritaJ.birzeit.edu\n\n".encode('utf-8')
101
102     # Default: 404 Not Found
103     return errorMSG(clientAddress)
104
105 1 usage
106 def errorMSG(clientAddress):
107     response = "HTTP/1.1 404 Not Found\nContent-Type: text/html\n\n"
108
109     try:
110         with open("NotFound.html", "r") as errorFile:
111             error_content = errorFile.read()
112             error_content = error_content.replace(_old: "[Client IP address and Port]", _new: f"{clientAddress[0]}")
113             response += error_content
114
115     except:
116         response = "HTTP/1.1 404 Not Found\nContent-Type: text/html\n\n"
117
118     return response
```



```
1 usage
56 def ProcessRequests(request,clientAddress):
57
58     # Check if the method is GET
59     if request.startswith("GET /"):
60
61         # Extract the path from the request
62         path = request.split()[1]
63
64         # Print the received HTTP request to the terminal
65         print(f"Received HTTP request: {request}")
66
67         # Send main_en.html file with Content-Type: text/html
68         if path == '/' or path == '/index.html' or path == '/main_en.html' or path == '/en':
69             return Serve(path, 'main_en.html', type='text/html')
70
71         # Response with main_ar.html which is an Arabic version of main_en.html
72         elif path == '/ar':
73             return Serve(path, 'main_ar.html', type='text/html')
74
75         # Send the requested html file with Content-Type: text/html
76         elif path.endswith('.html'):
77             return Serve(path[1:], type='text/html')
78
79         # Send the requested css file with Content-Type: text/css
80         elif path.endswith('.css'):
81             return Serve(path[1:], type='text/css')
82
83         # Send the png image with Content-Type: image/png
84         elif path.endswith('.png'):
85             return Serve(path[1:], type='image/png')
86
87         # Send the jpg image with Content-Type: image/jpeg
88         elif path.endswith('.jpg'):
89             return Serve(path[1:], type='image/jpeg')
90
91         # Redirect to cornell.edu website
92         elif path == '/cr':
93             return "HTTP/1.1 307 Temporary Redirect\nLocation: https://www.cornell.edu\n\n".encode('utf-8')
94
95         # Redirect to stackoverflow.com website
96         elif path == '/so':
97             return "HTTP/1.1 307 Temporary Redirect\nLocation: https://stackoverflow.com\n\n".encode('utf-8')
98         # Redirect to ritaJ website
99         elif path == '/rt':
100            return "HTTP/1.1 307 Temporary Redirect\nLocation: https://ritaJ.birzeit.edu\n\n".encode('utf-8')
101
102     # Default: 404 Not Found
103     return errorMSG(clientAddress)
104
105 1 usage
106 def errorMSG(clientAddress):
107     response = "HTTP/1.1 404 Not Found\nContent-Type: text/html\n\n"
108
109     try:
110         with open("NotFound.html", "r") as errorFile:
111             error_content = errorFile.read()
112             error_content = error_content.replace(_old: "[Client IP address and Port]", _new: f"{clientAddress[0]}")
113             response += error_content
114
115     except:
116         response = "HTTP/1.1 404 Not Found\nContent-Type: text/html\n\n"
117
118     return response
```

The screenshot shows a code editor window with several tabs at the top: main.py, WebServer.py (which is currently selected), file.html, main_en.html, NotFound.html, main_ar.html, and style.css. The main.py tab contains the following Python code:

```
106
107
108     response += "HTTP/1.1 404 Not Found\nContent-Type: text/html\n\n"
109
110     try:
111         with open("NotFound.html", "r") as errorFile:
112             error_content = errorFile.read()
113             error_content = error_content.replace( _old: "[Client IP address and Port]", _new: f"{clientAddress[0]}")
114             response += error_content
115     except FileNotFoundError:
116         response += "<html><body><h1>404 Not Found</h1><p>The requested resource was not found.</p></body></html>"
117
118     return response.encode('utf-8')
119
120
121     # usage
122     def start():
123         # Listening for incoming connections
124         serverSocket.listen(1)
125         print(f"Server is listening on {PORT}..")
126         while True:
127             # Accept incoming client connections
128             clientSocket, clientAddress = serverSocket.accept()
129
130             # Create a new thread for each connected client
131             thread = threading.Thread(target=HandleRequests, args=(clientSocket, clientAddress))
132             thread.start()
133
134
135     if __name__ == "__main__":
136         print("Server is starting..")
137         start()
138
139
140     ProcessRequests()
```

The status bar at the bottom shows: 56:44 CRLF UTF-8 4 spaces Python 3.11 (pythonProject) 5:37 PM 56°F Cloudy 12/21/2023

Figure 12:part 3,webserver code.

Explanation:

In this part, we created a web server that accepts connections, and listen on port 9966, then it takes requests from client, and gives responses according to these requests.

The serve() function reads the content of the file, constructs the HTTP request, then returns the HTTP response.

The HandleRequests() function receives client requests, returns the response(depending on the client address), and closes the connection.

The ProcessRequests() function checks the method (HTTP request message) if it is a GET(for sending data to server), extracts the path from the request, and responses with the requested file.

The errorMSG() function handles the error state, so if the request is wrong, or the file does not exist, it shows the Error page we have designed.

The start() function starts listens for incoming connections from the client, and creates a thread for each client.

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

Content Type:

It is a header field that specifies the media type of the underlying data that specifies the entity format. We can obtain the referenced media type from decoding mechanisms done by Content-Encoding.

We need it in the HTTP request because:

the HTTP uses the Content-Type header fields to provide, open, and extensible data typing and type negotiation. Also, it is crucial because it helps the recipient (client or server) to understand how to parse and process the content.

1- 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.

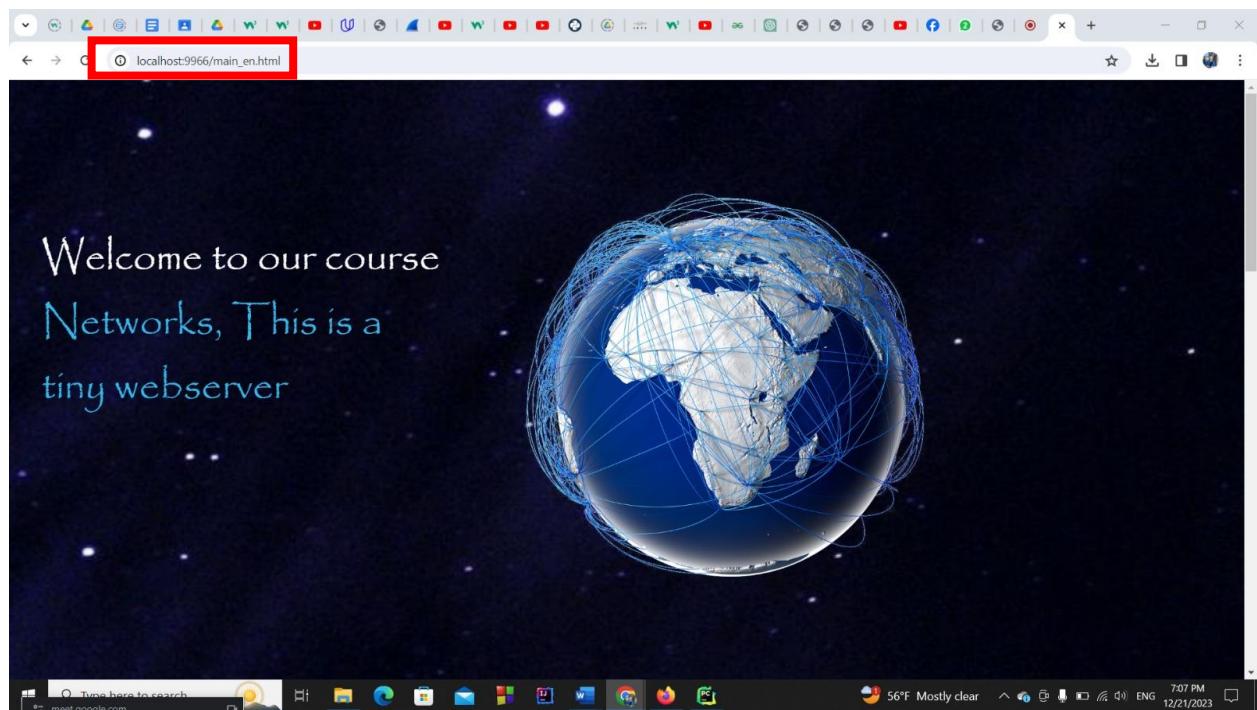
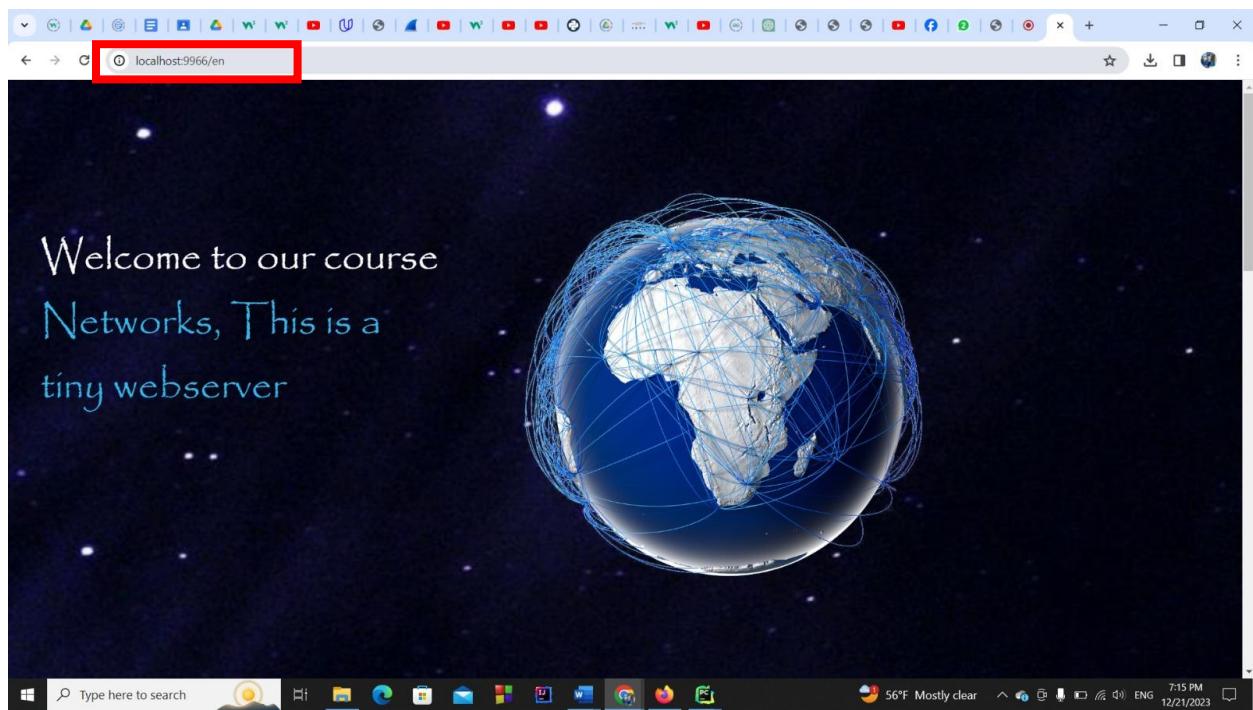
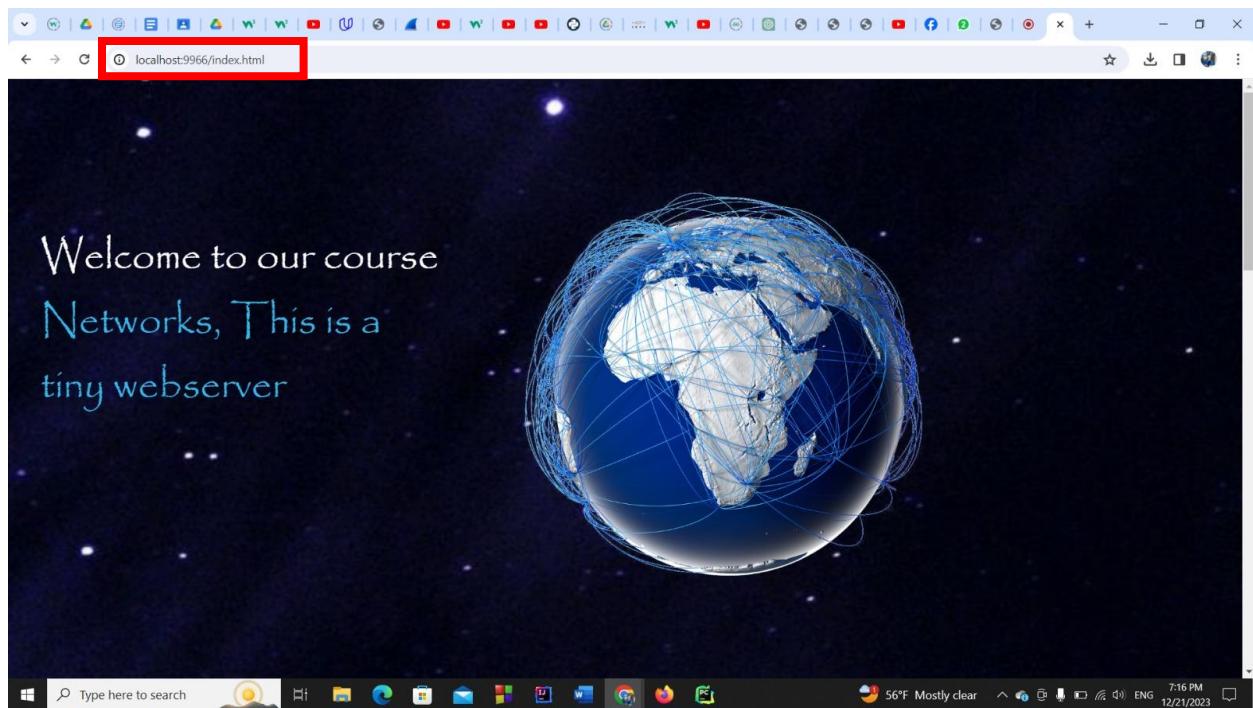


Figure 13:main_en.

The HTTP request '/main_en.html' redirects the client to our webpage.



The HTTP request '/en' redirects the client to our webpage.



The HTTP request '/index.html' redirects the client to our webpage.

```
↑ Received HTTP request: 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 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8,application/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-US,en;q=0.9
Cookie: Pycharm-24fe2247=1cd43fc4-83be-41c5-b33b-03405bf6eee7

pythonProject > WebServer.py
Type here to search 56°F Mostly clear 7:07 PM 12/21/2023
```

Figure 14:HTTP request for english

```
↑ Received HTTP request: GET / 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 Safari/537.36
Sec-Purpose: prefetch
Purpose: prefetch
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8,application/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-US,en;q=0.9
Cookie: Pycharm-24fe2247=1cd43fc4-83be-41c5-b33b-03405bf6eee7

pythonProject > WebServer.py
Type here to search 56°F Mostly clear 7:24 PM 12/21/2023
```

```
↑ Received HTTP request: GET /en 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 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8,application/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-US,en;q=0.9
Cookie: Pycharm-24fe2247=1cd43fc4-83be-41c5-b33b-03405bf6eee7

pythonProject > WebServer.py
Type here to search 56°F Mostly clear 7:25 PM 12/21/2023
```

```
↑ Received HTTP request: GET /index.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 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8,application/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-US,en;q=0.9
Cookie: Pycharm-24fe2247=1cd43fc4-83be-41c5-b33b-03405bf6eee7

pythonProject > WebServer.py
Type here to search 56°F Mostly clear 7:25 PM 12/21/2023
```

Explanation:

The previous HTTP request messages: (taking main_en.html as an example)

- The request line: “GET /main_en.html HTTP/1.1”.

GET: The method for sending data to server.

/main_en.html: The URL of the HTML file requested by the client.

HTTP/1.1: The version of the protocol used by the client.

- **The next line** in the terminal shows the IP address and the port number: ('127.0.0.1', 9966)
 - **The header lines:** the rest of the request message are header lines each with a header field name and a value.
 - **The body:** the body is empty.
-
- **2- If the request is /ar then the server response with main_ar.html which is an Arabic version of main_en.html**

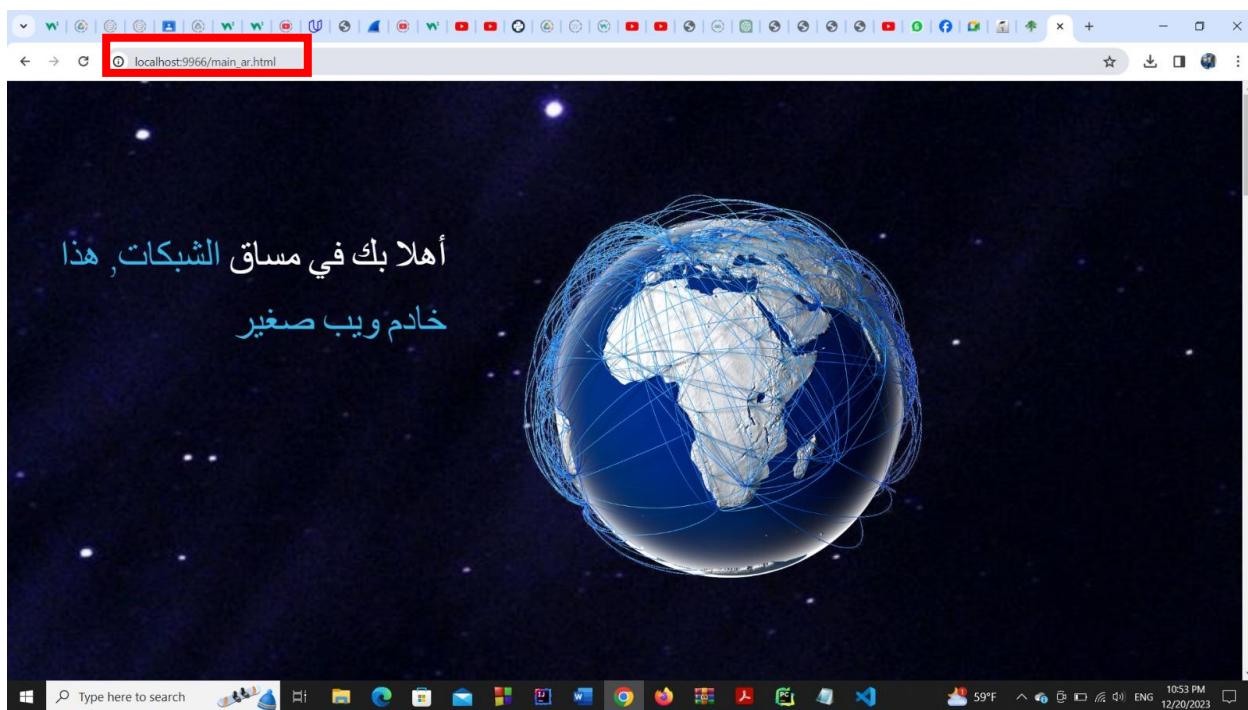


Figure 15:main_ar

The HTTP request ‘/main_ar.html’ redirects the client to our webpage.

```

Received HTTP request: GET /main_ar.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 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
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-24fe2247=1cd43fc4-83be-41c5-b33b-03405bf6eee7

```

Figure 16:HTTP request for arabic

- 3- 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.



Hello Dr.Abdalkareem & Dr.Mohammad

we Hope you enjoy our project



Figure 17: English html file

The HTTP request '/file.html' redirects the client to a webpage designed by html.

```

Received HTTP request: GET /file.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 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
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://localhost:9966/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-24fe2247=1cd43fc4-83be-41c5-b33b-03405bf6eee7

```

Figure 18:HTTP request

4- 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

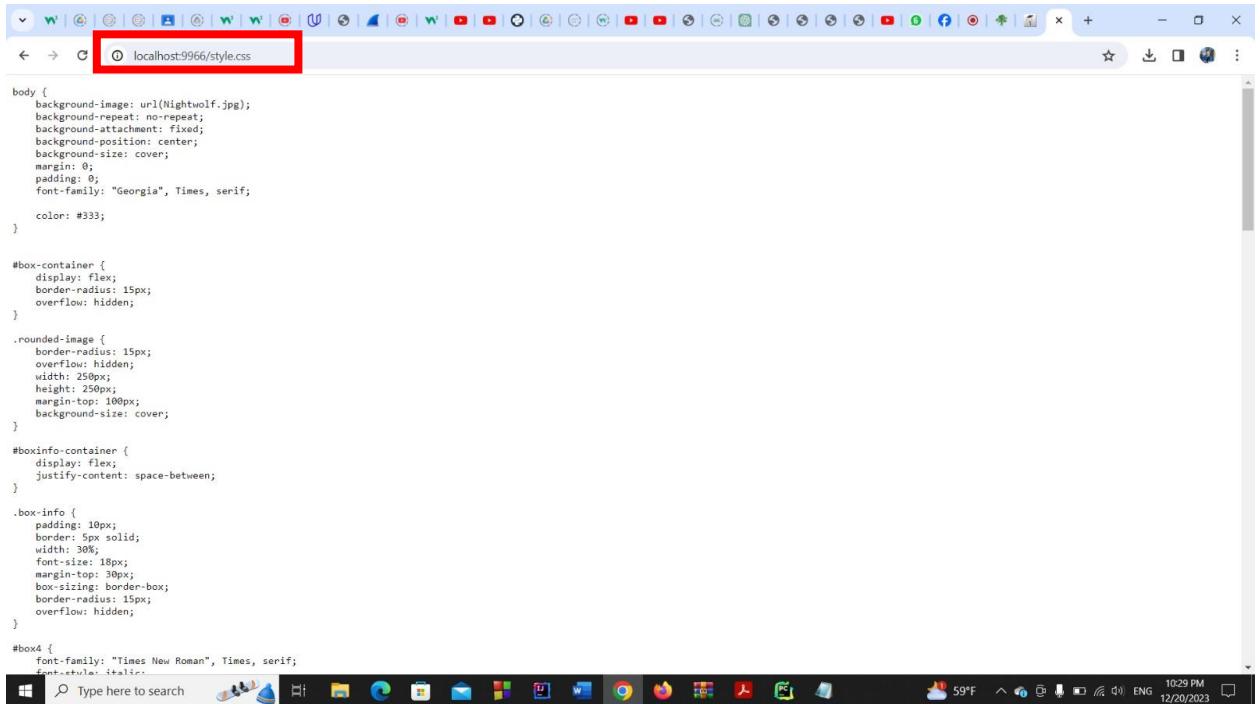


Figure 19: Style code

A screenshot of a terminal window showing an incoming HTTP request. The log entry is as follows:

```
↑ Received HTTP request: GET /style.css 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
→ User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36
→ sec-ch-ua-platform: "Windows"
Accept: text/css,*/*;q=0.1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: style
Referer: http://localhost:9966/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-24fe2247=1cd43fc4-83be-41c5-b33b-03405bf6eee7
```

Figure 20:Style HTTP request

**The HTTP request '/style.css' redirects the client to the css file stored on the computer
(The same path as the web server).**

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

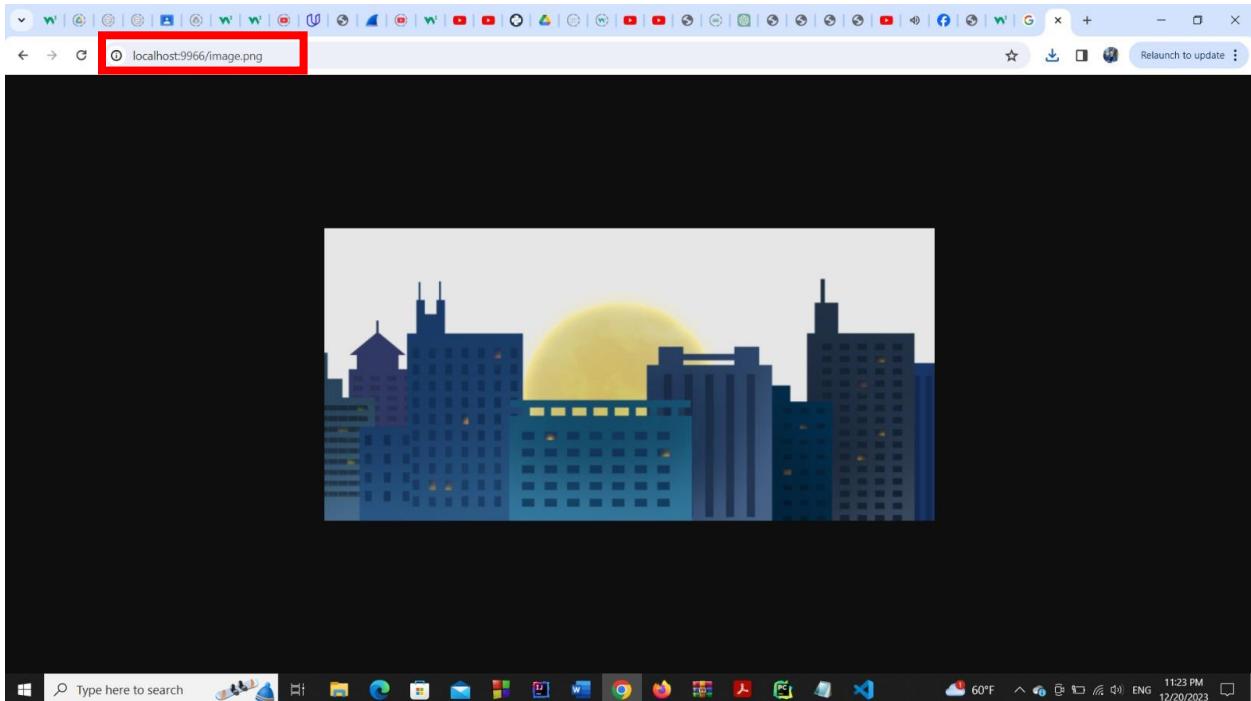


Figure 21:png image

A screenshot of a terminal window displaying an HTTP request log. The log shows a single entry for a GET request to "/image.png" from "localhost:9966". The log includes various headers such as Host, Connection, User-Agent, Accept, Sec-Fetch-Site, Sec-Fetch-Mode, Sec-Fetch-User, Sec-Fetch-Dest, Accept-Encoding, Accept-Language, and a Cookie. The terminal interface has a dark theme with light-colored text.

Figure 22:HTTP request for png

The HTTP request '/image.png' redirects the client to a png image stored on the computer (The same path as the web server).

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

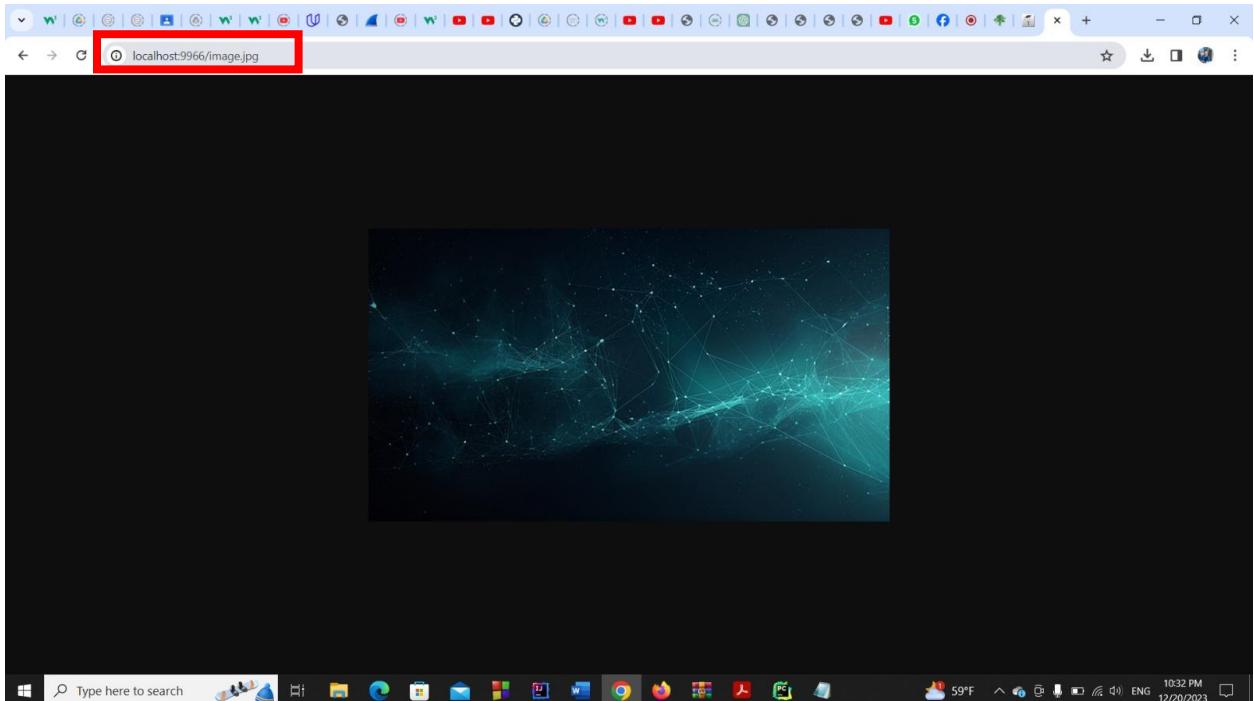


Figure 23:jpg image

```
↓ Received HTTP request: GET /image.jpg 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 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8,application/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-US,en;q=0.9
Cookie: Pycharm=24fe2247=1cd43fc4-83be-41c5-b33b-03405bf6eee7
```

Figure 24: HTTP request for ipa

The HTTP request ‘/image.jpg’ redirects the client to a jpg image stored on the computer (The same path as the web server).

7- Use the status code 307 Temporary Redirect to redirect the following

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

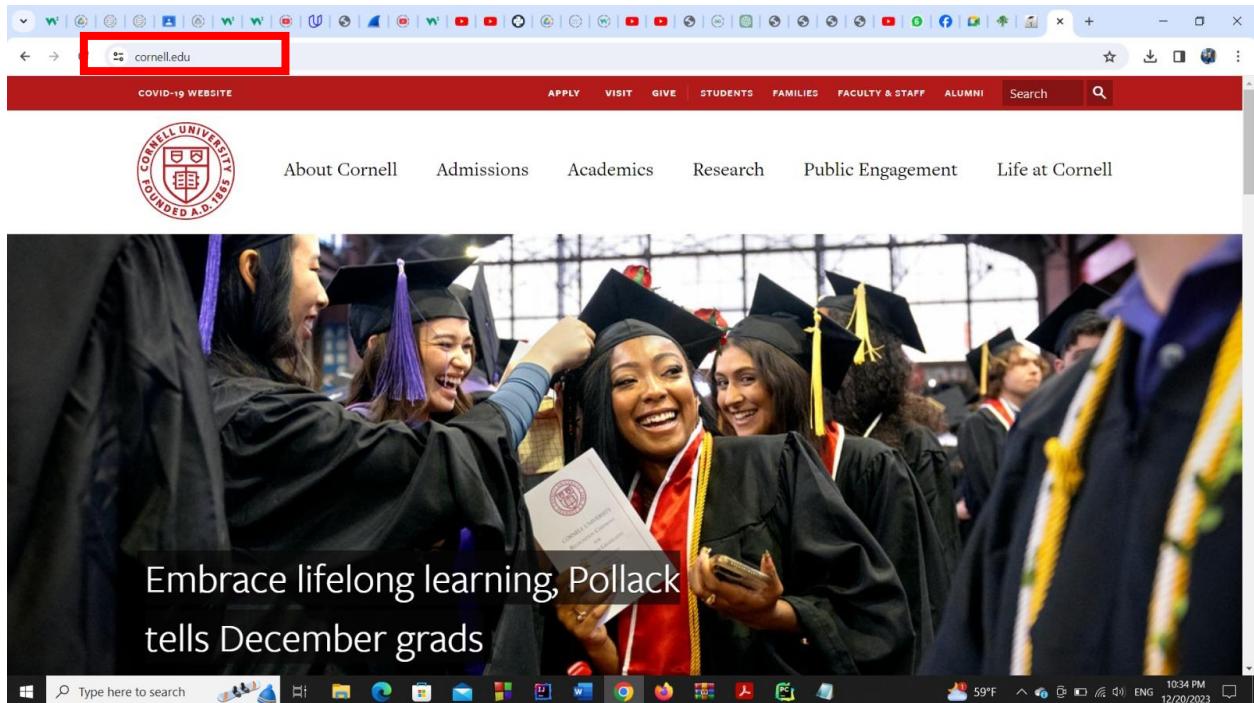


Figure 26:cornell.edu

A screenshot of a terminal window titled 'pythonProject > WebServer.py'. The window displays a log of an incoming HTTP request. The log entries include:

```
Received HTTP request: GET /cr 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 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8,application/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-US,en;q=0.9
Cookie: Pycharm-24fe2247=1cd43fc4-83be-41c5-b33b-03405bf6eee7
```

The terminal also shows the Python version and file path: 'Python 3.11 (pythonProject)'.

Figure 25:HTTP request for cornell

The HTTP request '/cr' redirects the client to cornell.edu website.

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

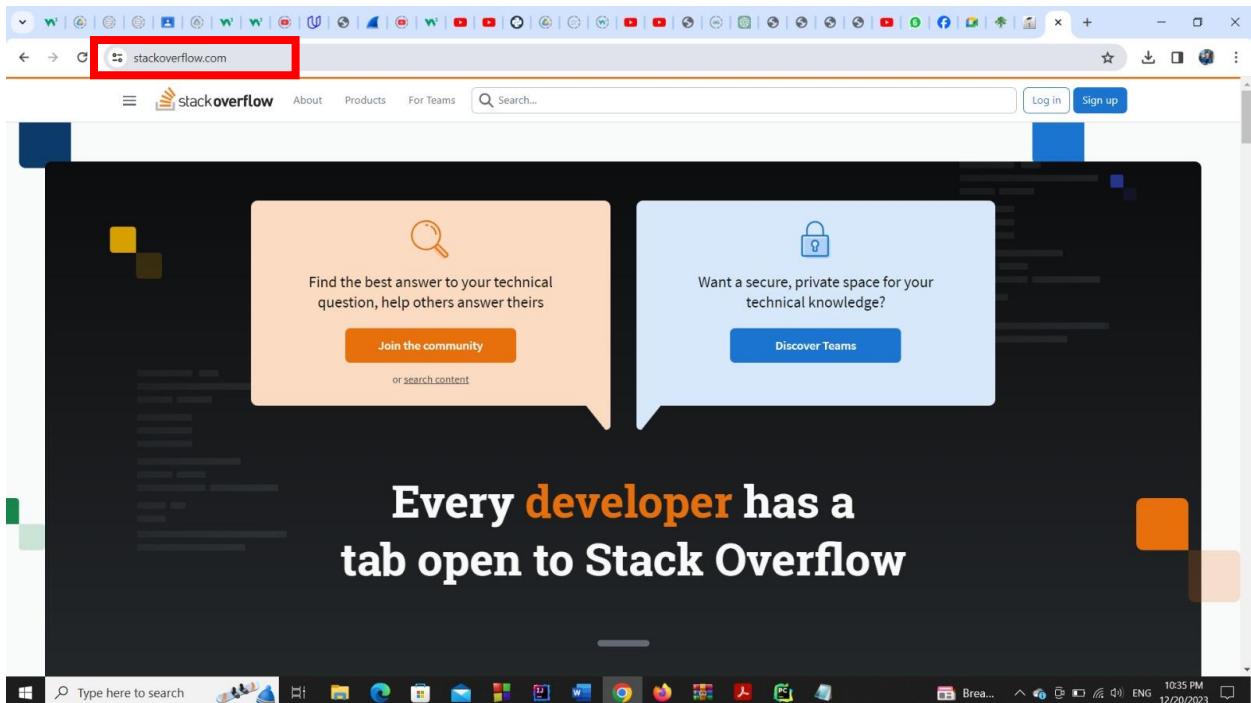


Figure 27:stackoverflow.com

```
↑ Received HTTP request: GET /so 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 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
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-24fe02247=1cd43fc4-83be-41c5-b33b-03405bf6eee7
```

Figure 28:HTTP request for stackoverflow

- If the request is /rt then redirect to ritaj website

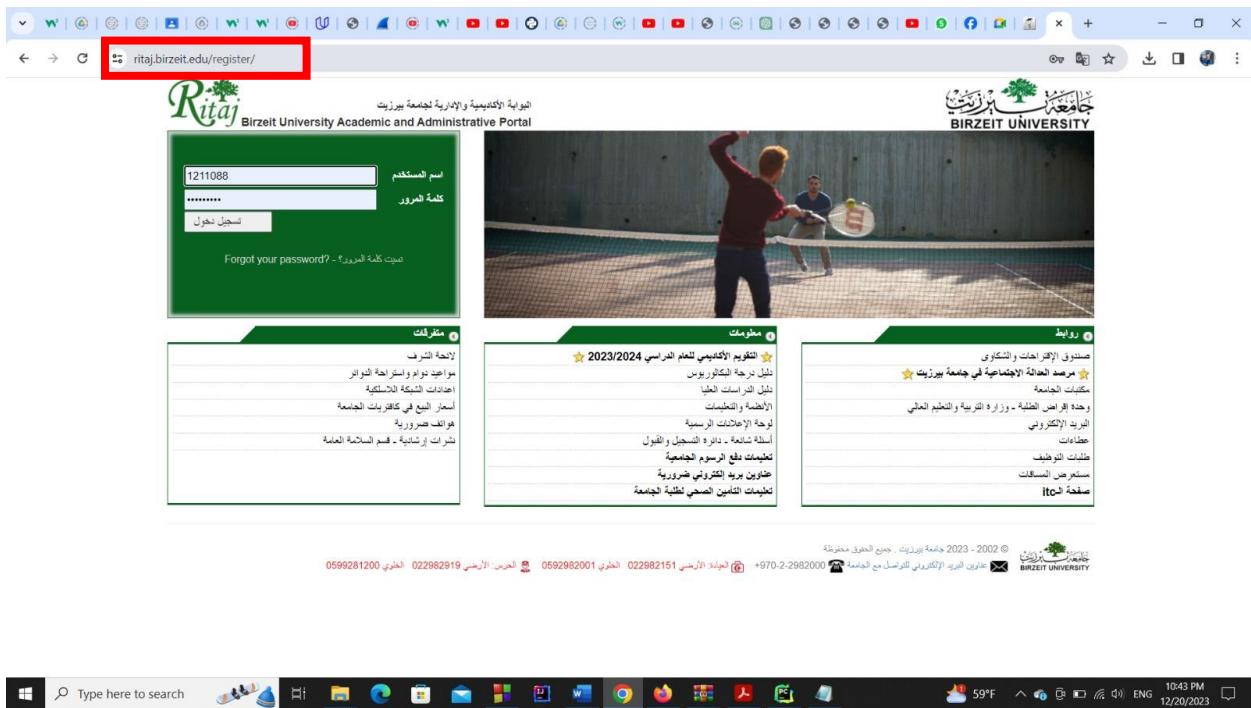


Figure 29:ritaj.edu

The HTTP request ‘/rt’ redirects the client to ritaj.edu website

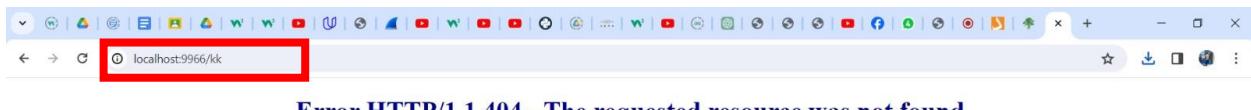
```

↑ Received HTTP request: GET /rt 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 Safari/537.36
Sec-Purpose: prefetch;prerender
Purpose: prefetch
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
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-24fe2247=1cd43fc4-83be-41c5-b33b-03405bf6eee7

```

Figure 30:HTTP request for ritaj

8- If the request is wrong or the file doesn't exist the server should return a simple HTML webpage that contains (Content-Type: text/html)



Client IP address and Port:127.0.0.1:50859



Figure 31:request is wrong or the file doesn't exist

The wrong HTTP request redirects the client to an Error page, indicates that the requested file is not found. It contains the client IP address and port, and our info.

A screenshot of a terminal window titled "pythonProject > WebServer.py". The window displays a log of a received HTTP request. The log entries are as follows:

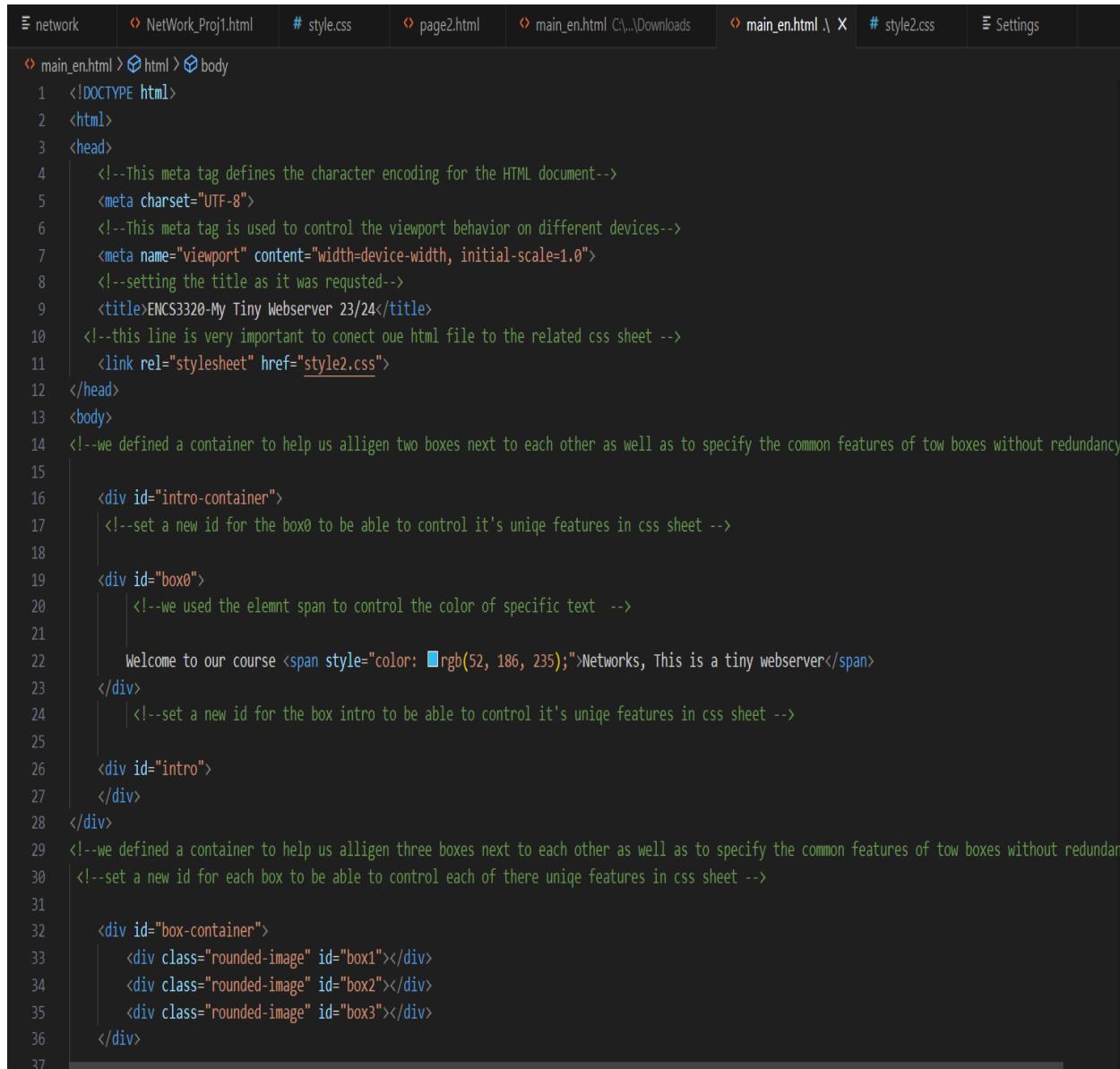
```
↑ Received HTTP request: GET /kk HTTP/1.1
↓ Host: localhost:9966
⇒ Connection: keep-alive
⇐ sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Google Chrome";v="120"
⇒ sec-ch-ua-mobile: ?0
⇐ sec-ch-ua-platform: "Windows"
⇒ Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/*q=0.8,application/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-US,en;q=0.9
Cookie: Pycharm-24fe2247=1cd43fc4-83be-41c5-b33b-03405bf6eee7
```

Figure 32:wrong HTTP request

Codes:

English Version

HTML (main_en.html)



```
network  NetWork_Proj1.html # style.css page2.html main_en.html C:\...\Downloads main_en.html \ X # style2.css Settings

<main_en.html>
  <!DOCTYPE html>
  <html>
    <head>
      <!--This meta tag defines the character encoding for the HTML document-->
      <meta charset="UTF-8">
      <!--This meta tag is used to control the viewport behavior on different devices-->
      <meta name="viewport" content="width=device-width, initial-scale=1.0">
      <!--setting the title as it was requested-->
      <title>ENCS3320-My Tiny Webserver 23/24</title>
      <!--this line is very important to connect our html file to the related css sheet -->
      <link rel="stylesheet" href="style2.css">
    </head>
    <body>
      <!--we defined a container to help us align two boxes next to each other as well as to specify the common features of two boxes without redundancy-->
      <div id="intro-container">
        <!--set a new id for the box0 to be able to control its unique features in css sheet -->
        <div id="box0">
          <!--we used the element span to control the color of specific text -->
          Welcome to our course <span style="color: #rgb(52, 186, 235);>Networks, This is a tiny webserver</span>
        </div>
        <!--set a new id for the box intro to be able to control its unique features in css sheet -->
        <div id="intro">
        </div>
      </div>
      <!--we defined a container to help us align three boxes next to each other as well as to specify the common features of two boxes without redundancy-->
      <!--set a new id for each box to be able to control each of them unique features in css sheet -->
      <div id="box-container">
        <div class="rounded-image" id="box1"></div>
        <div class="rounded-image" id="box2"></div>
        <div class="rounded-image" id="box3"></div>
      </div>
```

Figure 33:English html code

```

network NetWork_Proj1.html # style.css page2.html main_en.html C:\...\Downloads main_en.html X # style2.css Settings

main_en.html > html > body > div#boxinfo-container > div#boxinfo1.box-info > h5
39 <!--we defined a container to help us alligen three boxes next to each other as well as to specify the common features of tow boxes without redundant
40 <!--set a new id for each box to be able to control each of there unique features in css sheet -->
41
42 <div id="boxinfo-container">
43   <div class="box-info" id="boxinfo1">
44     <!--we used two elments span for the colors and u to underline the desired text -->
45     <!--we used h4,h5,h6 to control the size of the text as the number goes higher the text get smaller -->
46
47     <h4><u><span style="color: #rgb(161, 16, 115);"> Name</span></u> :Hala Qurt</h4>
48     <h4><u><span style="color: #rgb(161, 16, 115);"> ID</span></u>: 1220062</h4>
49
50     <h4><u><span style="color: #rgb(161, 16, 115);"> Hobbies</span></u> :</h4>
51     <!--we ul and li to create a list in html and introduce each item individualy -->
52
53     <h5>
54       <!--This is a list of Hala's hobbies -->
55
56       <ul>
57         <li>Reading</li>
58         <li>sport</li>
59         <li>singing</li>
60         <li>programming</li>
61       </ul>
62     </h5>
63
64     <h4><u><span style="color: #rgb(161, 16, 115);"> Project</span></u></h4>
65     <h5>
66       <!--and this is a list of Hala's projects -->
67
68       <ol>
69         <li>
70           <h5>- A System for booking train tickets (Java1 course)</h5>
71         </li>
72         <li>
73           <h5>- a system for pizza ordering and delivery (Java 2 )</h5>
74         </li>
75       </ol>

```

network NetWork_Proj1.html # style.css page2.html main_en.html C:\...\Downloads main_en.html X # style2.css Settings

```

    <main_en.html> <html> <body> <div#boxinfo-container> <div#boxinfo2.box-info> <h5>
    74 |     </li>
    75 |     <li>
    76 |         <h5>- a system that show and modify the internet usage in countries (Data structures course)</h5>
    77 |     </li>
    78 |     <li>
    79 |         <h5>-a system for Gaza electric sources (Data structures course)</h5>
    80 |     </li>
    81 |
    82     </ol>
    83 </h5>
    84 </div>
    85
    86
    87 <div class="box-info" id="boxinfo2">
    88     <h4><u><span style="color: #rgb(161, 16, 115);"> Name</span></u> :Doaa Hatu</h4>
    89     <h4><u><span style="color: #rgb(161, 16, 115);"> ID</span></u>: 1211088</h4>
    90
    91     <h4><u><span style="color: #rgb(161, 16, 115);"> Hobbies</span></u> :</h4>
    92     <h5>
    93         <!--This is a list of Doaa's hobbies -->
    94         <ul>
    95             <li>-Reading</li>
    96             <li>-Writing</li>
    97             <li>-Astronomy</li>
    98             <li>-Design</li>
    99         </ul>
    100    </h5>
    101
    102    <h4><u><span style="color: #rgb(161, 16, 115);"> Project</span></u>:</h4>
    103    <h5>
    104        <!--This is a list of Doaa's hobbies -->
    105        <ol>
    106            <li>
    107                <h5>-A delivery system for pizza restaurant (Java course)</h5>
    108            </li>
    109            <li>
    110                <h5>-School system for students (Data structures and algorithms course)</h5>

```

network NetWork_Proj1.html # style.css page2.html main_en.html C:\...\Downloads main_en.html \ X # style2.css Settings

```

100 <h5>-School system for students (Data structures and algorithms course)</h5>
101 </li>
102 <li>
103     <h5>-Simple calculator (Data structures and algorithms course)</h5>
104 </li>
105 <li>
106     <h5>-Scheduling algorithms (Operating Systems course)</h5>
107 </li>
108 <li>
109     <h5>Diode circuits simulation project (Electronics course)</h5>
110 </li>
111 </ol>
112 </h5>
113 </div>
114 <div class="box-info" id="boxinfo3">
115     <h4><u><span style="color: #rgb(161, 16, 115);"> Name</span></u> :Raghad Owaisi</h4>
116     <h4><u><span style="color: #rgb(161, 16, 115);"> ID</span></u>: 1200710</h4>
117
118     <h4><u><span style="color: #rgb(161, 16, 115);"> Hobbies</span></u> :</h4>
119     <h5>
120         <!--This is a list of Raghads hobbies -->
121         <ul>
122             <li>-programming</li>
123             <li>-video Production</li>
124             <li>-Communication & media</li>
125             <li>-Designing</li>
126         </ul>
127     </h5>
128
129     <h4><u><span style="color: #rgb(161, 16, 115);"> Project</span></u>:</h4>
130     <h5>
131         <!--This is a list of Raghad's hobbies -->
132         <ol>
133             <li>
134                 <h5>-Car Agency System (Data structures course)</h5>
135             </li>
136             <li>
```

```

    <!--Martyrs Data(Data structures and algorithms course)</h5>
</li>
<li>
    <h5>-Dental Clinic(Data Base Course)</h5>
</li>
<li>
    <h5>-Stack converting(Data structure course)</h5>
</li>
<li>
    <h5>-SM (Secure Media)(Usable security course)</h5>
</li>
</ol>
</h5>
</div>
</div>

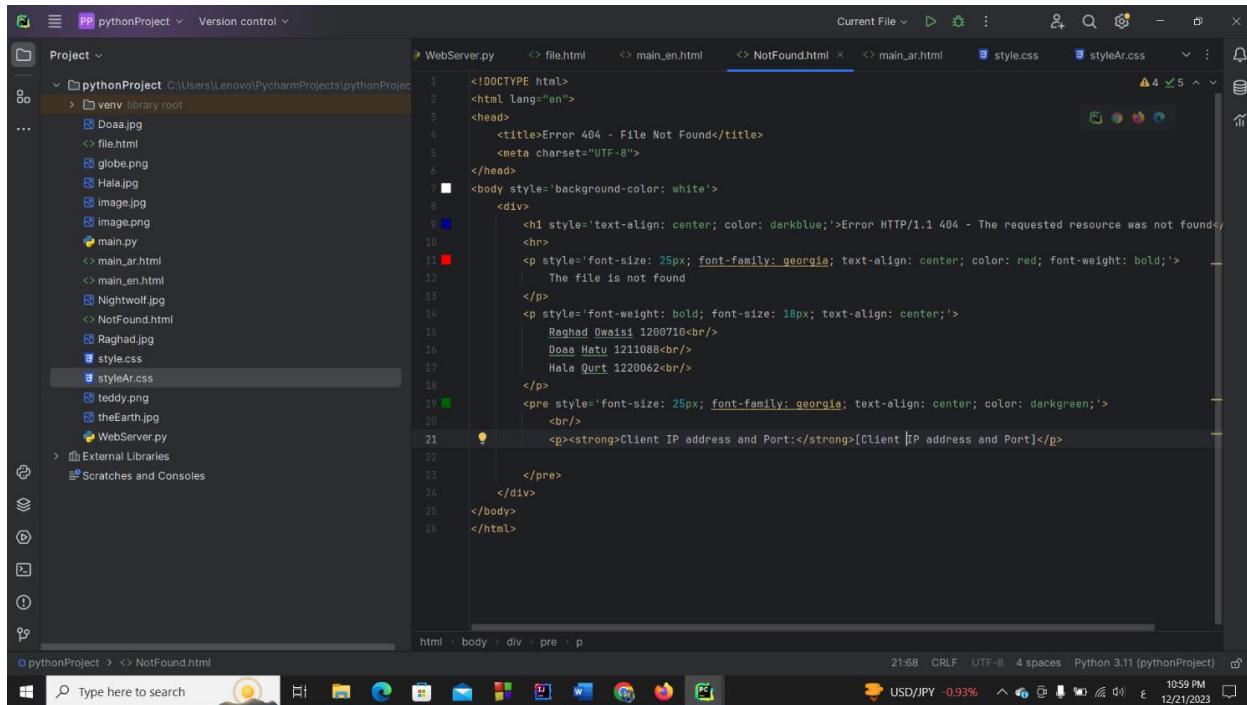
<div id="box4">
    <!--we uswd elemnt mark to highlight the text -->
    <h4><mark><u>Content-Type</u></mark></h4>
    <p>It is a header field that specifies the media type of the underlying data that specifies The entity format. We can obtain the referenced media type from decoding mechanisms done by Content-Encoding.</p>
    <h4><mark><u>We need it in the HTTP request because</u></mark></h4>
    <ol>
        <li>the HTTP uses the Content-Type header fields to provide, open, and extensible data typing and type negotiation.</li>
        <li>Also, it is crucial because it helps the recipient (client or server) to understand how to parse and process the content.</li>
    </ol>
</div>

<h2></h2>
<!-- the a element for hyper links -->
<a href="http://127.0.0.1:5500/" target="_blank">
    <button id = "button1"> html file</button>
</a>
</h2>
<h2>
    <a href="https://www.w3schools.com/python/python_strings.asp" target="_blank">
        <button id = "button2"> W3schools</button>

```

```
network  ◊ NetWork_Proj1.html # style.css ◊ page2.html ◊ main_en.html C:\...\Downloads ◊ main_en.html.\ X # style2.css Settings

◊ main_en.html > html > body > h2 > a > button#button2
</u>
161 </div>
162
163 <div id="box4">
164     <!--we uswd elemnt mark to highlight the text -->
165     <h4><mark><u>Content-Type</u></mark></h4>
166     <p>It is a header field that specifies the media type of the underlying data that specifies The entity format.
167     We can obtain the referenced media type from decoding mechanisms done by Content-Encoding.</p>
168     <h4><mark><u>We need it in the HTTP request because</u> :</mark></h4>
169     <ol>
170         <li>the HTTP uses the Content-Type header fields to provide, open, and extensible data typing and type negotiation.</li>
171         <li>Also, it is crucial because it helps the recipient (client or server) to understand how to parse and process the content.</li>
172     </ol>
173 </div>
174
175 <h2></h2>
176 <!-- the a element for hyper linkes -->
177 <a href="http://127.0.0.1:5500/" target="_blank">
178     <button id = "button1"> html file</button>
179 </a>
180 </h2>
181 <h2>
182     <a href="https://www.w3schools.com/python/python_strings.asp" target="_blank">
183         <button id = "button2"> W3schools</button>
184     </a>
185 </h2>
186 </body>
187 </html>
188
```



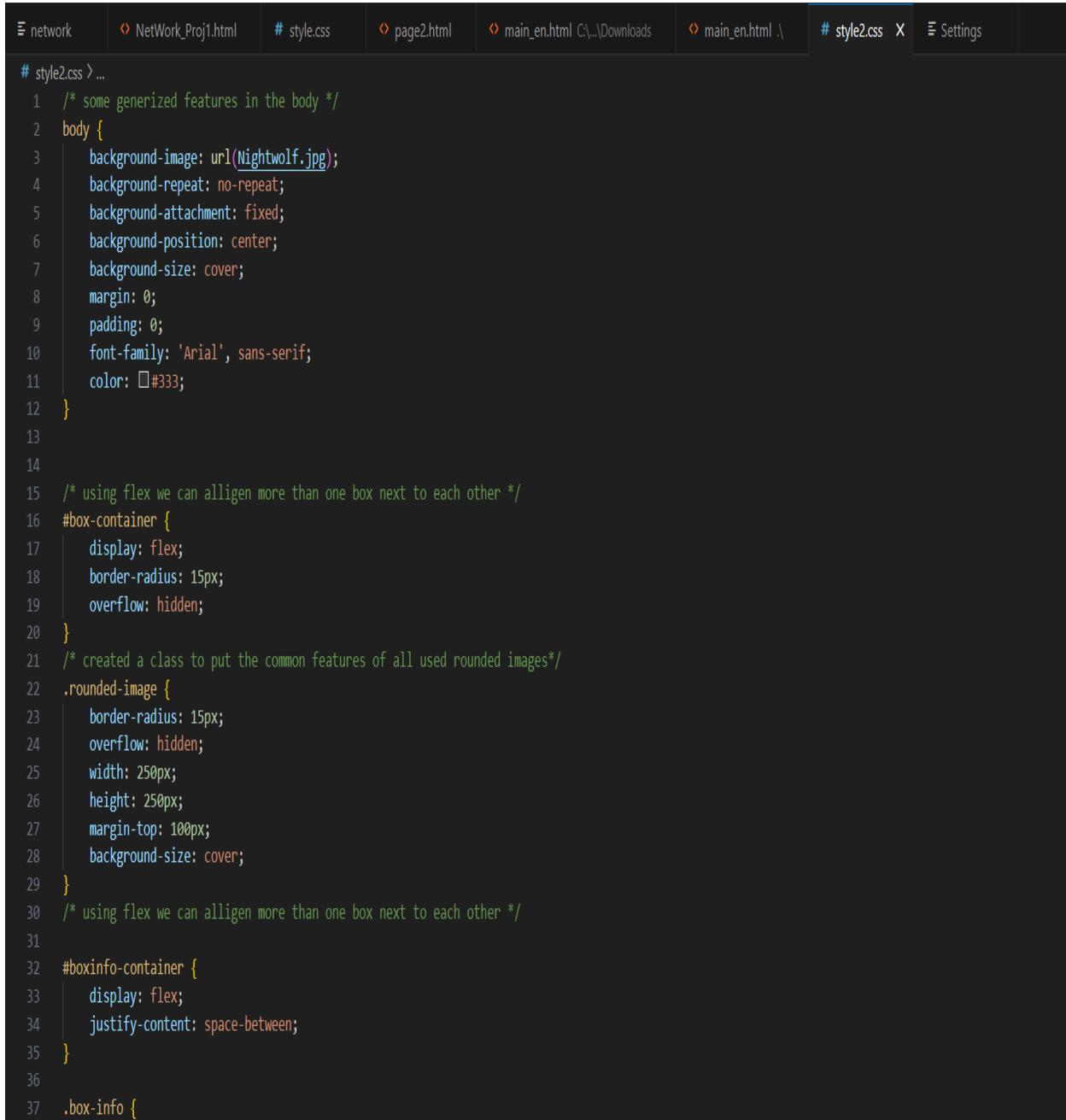
The screenshot shows the PyCharm IDE interface with the 'NotFound.html' file open in the editor. The code is as follows:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Error 404 - File Not Found</title>
    <meta charset="UTF-8">
</head>
<body style='background-color: white'>
    <div>
        <h1 style='text-align: center; color: darkblue;'>Error HTTP/1.1 404 - The requested resource was not found</h1>
        <br>
        <p style='font-size: 25px; font-family: georgia; text-align: center; color: red; font-weight: bold;'>
            The file is not found
        </p>
        <p style='font-weight: bold; font-size: 18px; text-align: center;'>
            Raghad Owaisi 1200710<br/>
            Doaa Hatu 1211088<br/>
            Hala Qurt 1220062<br/>
        </p>
        <pre style='font-size: 25px; font-family: georgia; text-align: center; color: darkgreen;'>
            <br/>
            <p><strong>Client IP address and Port:</strong>[Client IP address and Port]</p>
        </pre>
    </div>
</body>
</html>
```

Figure 34:error page HTML code

This HTML code is used to design a personalized 404 error page. It features a centrally positioned heading, a horizontal line to separate sections, a primary error message, the students' data, and the client's IP address and port. The page is carefully styled with particular fonts, colors, and alignments to enhance its visual appeal.

style.css Code



The screenshot shows a code editor window with the following tabs at the top: network, NetWork_Proj1.html, # style.css, page2.html, main_en.html C:\...\Downloads, main_en.html \, # style2.css X, and Settings. The current tab is # style.css.

```
# style2.css > ...
1 /* some generalized features in the body */
2 body {
3     background-image: url(Nightwolf.jpg);
4     background-repeat: no-repeat;
5     background-attachment: fixed;
6     background-position: center;
7     background-size: cover;
8     margin: 0;
9     padding: 0;
10    font-family: 'Arial', sans-serif;
11    color: #333;
12 }
13
14
15 /* using flex we can align more than one box next to each other */
16 #box-container {
17     display: flex;
18     border-radius: 15px;
19     overflow: hidden;
20 }
21 /* created a class to put the common features of all used rounded images*/
22 .rounded-image {
23     border-radius: 15px;
24     overflow: hidden;
25     width: 250px;
26     height: 250px;
27     margin-top: 100px;
28     background-size: cover;
29 }
30 /* using flex we can align more than one box next to each other */
31
32 #boxinfo-container {
33     display: flex;
34     justify-content: space-between;
35 }
36 .box-info {
```

Figure 35:Style css code

The screenshot shows a code editor window with a dark theme. At the top, there are several tabs: "network", "NetWork_Proj1.html", "# style.css", "page2.html", "main_en.html C:\...\Downloads", "main_en.html \\", "# style2.css X", and "Settings". The main area contains the following CSS code:

```
# style2.css > 4. box-info
37 .box-info {
38     padding: 10px;
39     border: 5px solid;
40     width: 30%;
41     font-size: 18px;
42     margin-top: 30px;
43     box-sizing: border-box;
44     border-radius: 15px;
45     overflow: hidden;
46 }
47 /* we used some features to make the web page look nicer and more organized */
48 #box4 {
49     font-family: "Times New Roman", Times, serif;
50     font-style: italic;
51     margin: 20px auto;
52     border: 0px ;
53     width: 80%;
54     font-size: 22px;
55     color: #rgb(63, 8, 77);
56     padding: 10px;
57     background-image: url(htmldesign.png);
58     background-size: cover;
59     border-radius: 15px;
60     overflow: hidden;
61     margin-bottom: 150px;
62 }
63 /* instead of repeating same features we used the three classes together */
64 #box1, #box2, #box3 {
65     border: 5px solid;
66     width: 250px;
67     height: 250px;
68     font-size: 22px;
69     margin-top: 100px;
70     margin-bottom: 30px;
71     border-radius: 15px;
72     overflow: hidden;
```

At the bottom right of the editor, there is status information: "Ln 37, Col 8 (4 selected) Spaces: 4 UTF-8 CRLF CSS".

```
# style2.css > #box1
75 .highlight-text {
76   color: #rgb(201, 182, 182);
77 }
78 /* specify the background image and margins for each box */
79
80 #box1 {
81   margin-left: 120px;
82   background-image: url(Hala.jpg);
83   background-size: cover;
84 }
85
86 #box2 {
87   margin-left: 250px;
88   background-image: url(Doaa.jpg);
89   background-size: cover;
90 }
91
92 #box3 {
93   margin-left: 280px;
94   background-image: url(Raghad.jpg);
95   background-size: cover;
96 }
97 /* insted of repeating same features we used the three classes togather */
98
99 #boxinfo1, #boxinfo2, #boxinfo3 {
100   font-family: "Times New Roman", Times, serif;
101   padding: 10px;
102   border: 5px solid;
103   width: 30%;
104   font-size: 18px;
105   margin-top: 30px;
106   margin-bottom: 150px;
107   box-sizing: border-box;
108   border-radius: 15px;
109   overflow: hidden;
110 }
111 /* to organize the list in modek/
```

```
# network      ◊ NetWork_Proj1.html   # style.css    ◊ page2.html    ◊ main_en.html C:\...\Downloads   ◊ main_en.html .\  # style2.css X  ☰ Settings

# style2.css > #boxinfo1
111 /* to oronize the lists we made*/
112 ul, ol {
113     list-style-type: none;
114     margin: 0;
115     padding: 0;
116 }
117
118 li {
119     margin-bottom: 0px;
120 }
121 /* insted of repeating same features we used the three classes togather */
122
123 #boxinfo1 {
124     font-style: italic;
125     border-color: ■rgb(201, 182, 182);
126     background-color: ■rgb(201, 182, 182);
127     margin-left: 30px;
128     color: □black;
129 }
130
131 #boxinfo2 {
132     font-style: italic;
133     border-color: ■rgb(201, 182, 182);
134     background-color: ■rgb(201, 182, 182);
135     margin-left: 70px;
136     color: □black;
137 }
138
139 #boxinfo3 {
140     font-style: italic;
141     border-color: ■rgb(201, 182, 182);
142     background-color: ■rgb(201, 182, 182);
143     margin-left: 100px;
144     color: □black;
145 }
146 /* insted of repeating same features we used the main class button */
147
```

The screenshot shows a code editor window with a dark theme. At the top, there are several tabs: "network", "NetWork_Proj1.html", "# style.css", "page2.html", "main_en.html C:\..\Downloads", "main_en.html \\", "# style2.css X", and "Settings". The main area contains the following CSS code:

```
148 button {
149     font-size: 30px;
150     margin-top: 20px;
151     font-style: italic;
152     font-weight: bold;
153     font-family: "Times New Roman", Times, serif;
154     padding: 8px;
155     border-radius: 10px; /* Adjust the border-radius as needed */
156     color: #rgb(25,25,112);
157     align-self: center;
158 }
160 #button1{
161     margin-left: 700px;
162 }
163 }
164 #button2{
165     margin-left: 680px;
166 }
167 }
168 #box0{
169     font-family:'fantasy',Papyrus;
170     margin-top: 150px;
171     margin-left: 10px;
172     margin-right: 20px;
173     border : 0px ;
174     width : 500px ;
175     height : 200px;
176     font-size: 50px;
177
178     padding : 30px;
179
180     color:#white;
181
182
183
184
```

At the bottom right of the editor, there are status indicators: "Ln 146, Col 45 (4 selected)", "Spaces: 4", "UTF-8", "CRLF", "CSS", and an empty circle icon.

This CSS code aims to create a visually appealing and organized layout by using flexbox, rounded images, specific fonts, and carefully chosen colors.

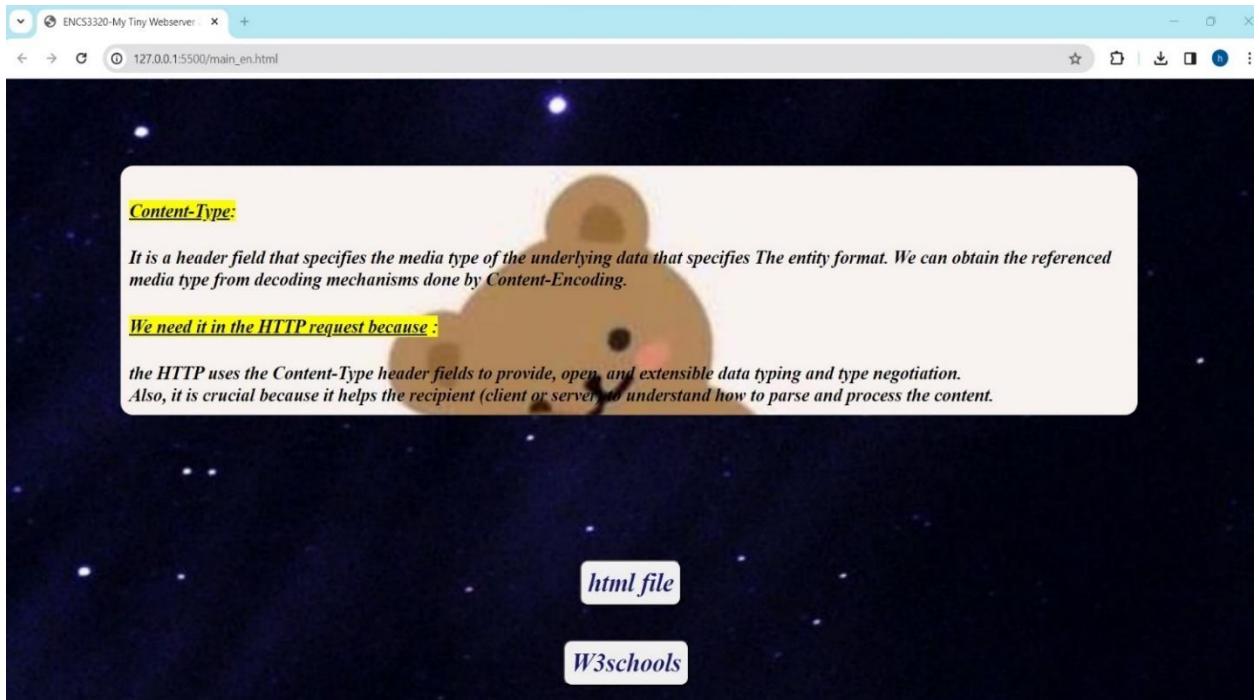
Web server:



Figure 36:English web server

A screenshot of a web browser window titled "ENCS3320-My Tiny Webserver". The URL in the address bar is "127.0.0.1:5500/main_en.html". The page displays three student profiles in a grid format. Each profile includes a photo, name, ID, hobbies, and projects.

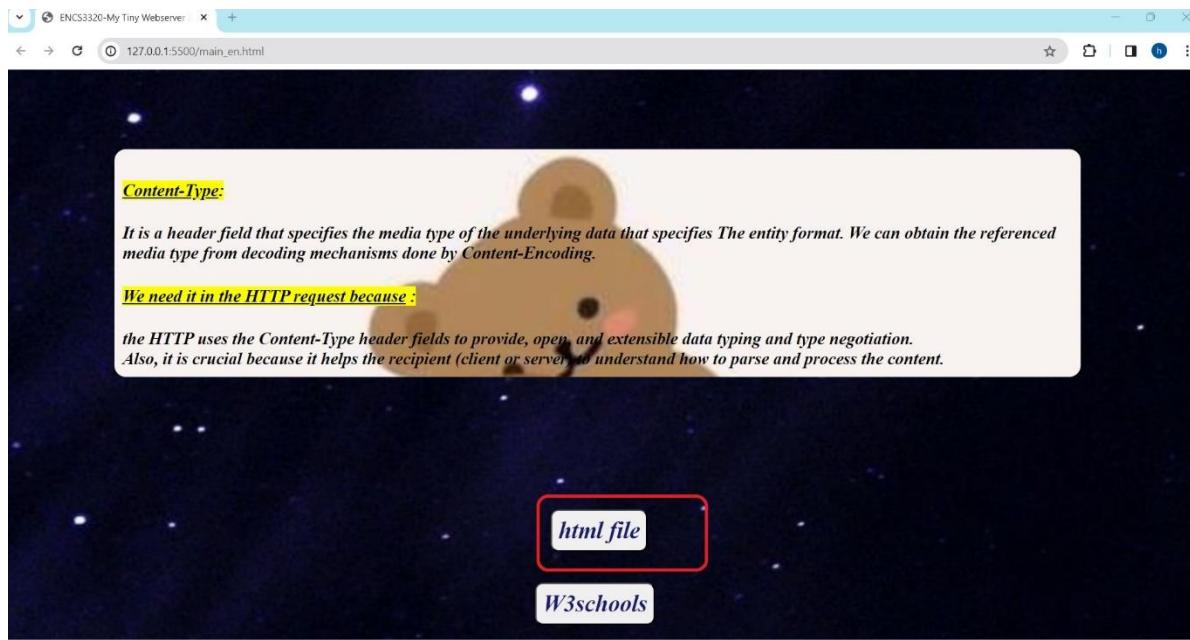
Name	ID	Hobbies	Projects
Hala Qurt	1220062	-Reading -sport -singing -programming	-A System for booking train tickets (Java1 course) - a system for pizza ordering and delivery (Java 2) - a system that show and modify the internet usage in countries (Data structures course)
Doaa Hatu	1211088	-Reading -Writing -Astronomy -Design	-A delivery system for pizza restaurant (Java course) -School system for students (Data structures and algorithms course) -Simple calculator (Data structures and algorithms course)
Raghad Owaisi	1200710	-programming -video Production -Communication & media -Designing	-Car Agency System (Data structures course) -Martyrs Data(Data structures and algorithms course) -Dental Clinic(Data Base Course)



In the figures above we made sure to follow the instructions as given in the project description by setting the title of the page, the welcome phrase with the right colors, the group member names, IDs, Hobbies and preformed projects.

We also used CSS to enhance the frontend part of the project in every HTML file.

In the following figures, we will show the pages which our main page redirect you (by using hyper links).



The html file button redirect you to the following html page (a local html page).

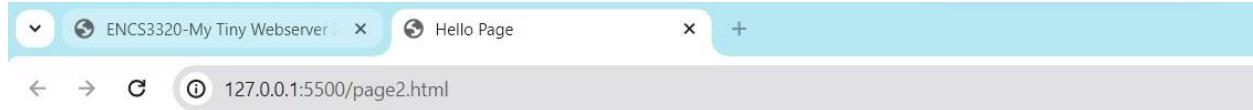
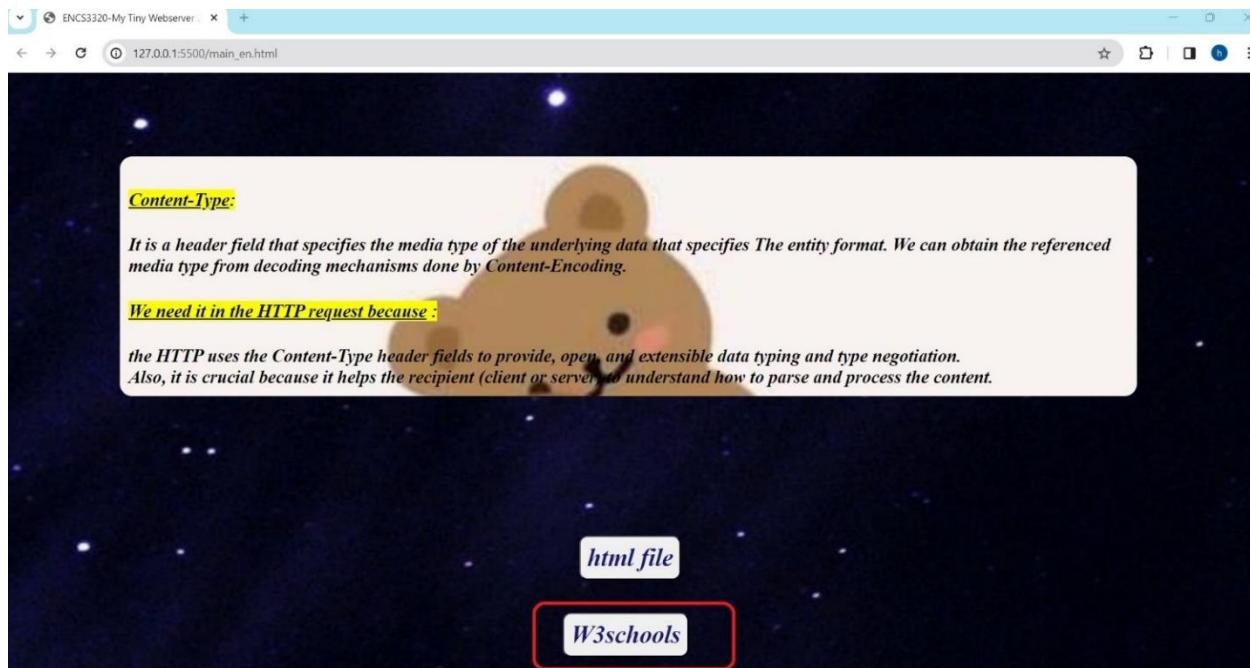


Figure 37:html file



The W3schools button redirect you to the following page (a link that was provided by the project).

The screenshot shows a web browser displaying the W3schools Python Strings page. The URL in the address bar is w3schools.com/python/python_strings.asp. The page content includes a sidebar with Python tutorial links like 'Python HOME', 'Python Intro', and 'Python Strings'. The main content area features a green banner for a 'Holiday Special' offer to 'Certify Your Skills! Boost Your Career' with a 'Save 775\$' discount. Below the banner, the title 'Python Strings' is displayed, followed by a 'Strings' section with text about Python strings and examples. A code editor shows the following Python code:

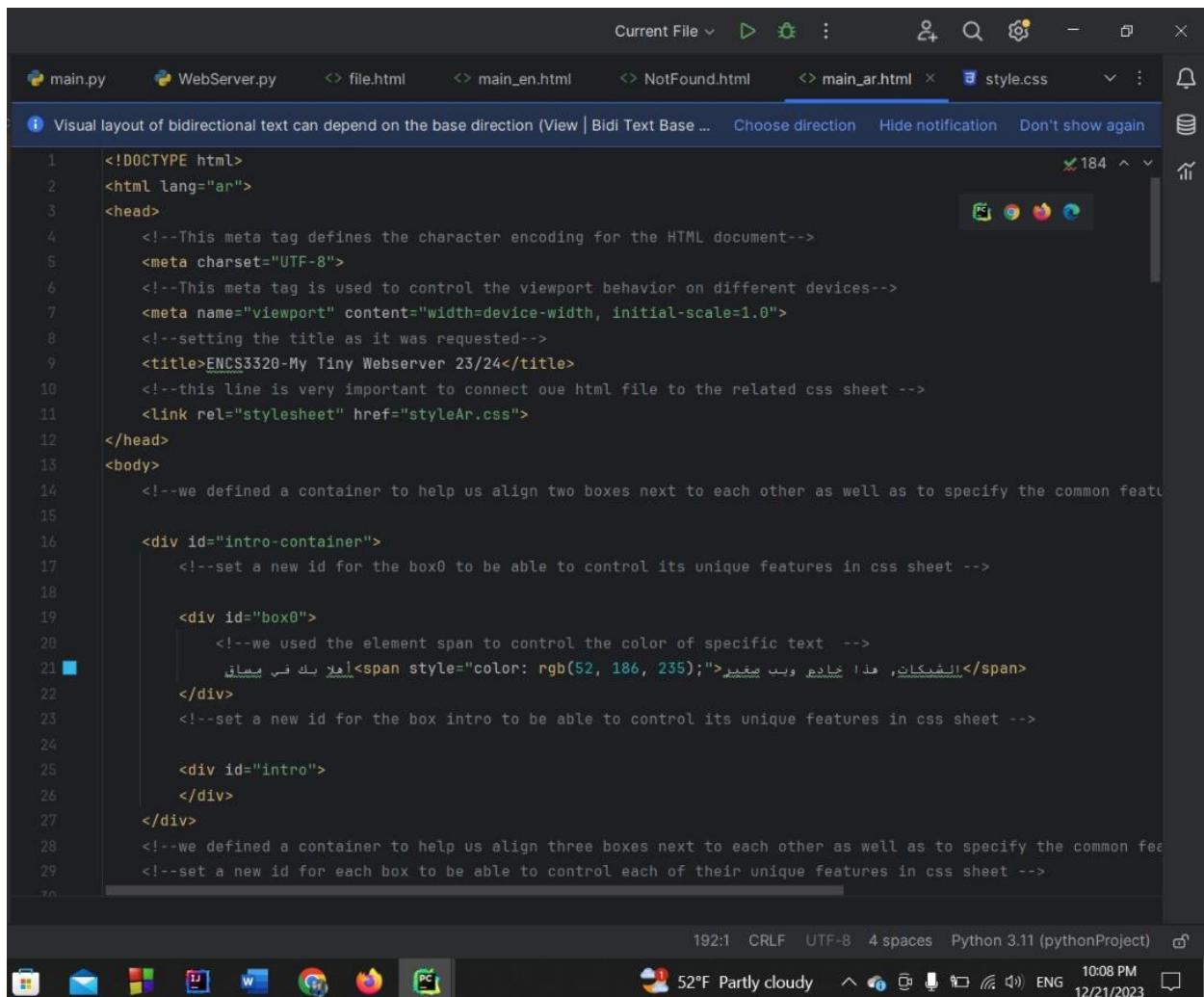
```
print("Hello")
print('Hello')
```

On the right side, there are two advertisements for 'Build Your Career!' offers with 'Save 775\$' discounts. One ad is for 'Lifelong access to all W3Schools existing and future courses, exams and certifications.' The W3schools logo is visible throughout the page.

Figure 38:W3schools file

Arabic Version

HTML code (main_ar.html):



The screenshot shows a code editor window with the following details:

- File Explorer:** Shows files: main.py, WebServer.py, file.html, main_en.html, NotFound.html, main_ar.html (current), style.css.
- Status Bar:** Current File, 192:1 CRLF, UTF-8, 4 spaces, Python 3.11 (pythonProject).
- Notification Bar:** Visual layout of bidirectional text can depend on the base direction (View | Bidi Text Base ...), Choose direction, Hide notification, Don't show again.
- Code Content:**

```
1  <!DOCTYPE html>
2  <html lang="ar">
3  <head>
4      <!--This meta tag defines the character encoding for the HTML document-->
5      <meta charset="UTF-8">
6      <!--This meta tag is used to control the viewport behavior on different devices-->
7      <meta name="viewport" content="width=device-width, initial-scale=1.0">
8      <!--setting the title as it was requested-->
9      <title>ENCS3320-My Tiny Webserver 23/24</title>
10     <!--this line is very important to connect our html file to the related css sheet -->
11     <link rel="stylesheet" href="styleAr.css">
12 </head>
13 <body>
14     <!--we defined a container to help us align two boxes next to each other as well as to specify the common features-->
15
16     <div id="intro-container">
17         <!--set a new id for the box0 to be able to control its unique features in css sheet -->
18
19         <div id="box0">
20             <!--we used the element span to control the color of specific text -->
21             إدخال بوك فون
22         </div>
23         <!--set a new id for the box intro to be able to control its unique features in css sheet -->
24
25         <div id="intro">
26             </div>
27     </div>
28     <!--we defined a container to help us align three boxes next to each other as well as to specify the common features-->
29     <!--set a new id for each box to be able to control each of their unique features in css sheet -->
```
- Bottom Taskbar:** Icons for File, Open, Save, Print, Copy, Paste, Find, Replace, Undo, Redo, and others.
- System Tray:** Icons for battery, signal strength, volume, and date/time (10:08 PM, 12/21/2023).

Figure 39:html Arabic code

The screenshot shows a code editor window with the following code:

```
59     <li><span style="color: #1a237e;">البرمجة</span></li>
60   </ul>
61
62
63   <h4><u><span style="color: #1a237e;">المشاريع</span></u></h4>
64
65   <!--and this is a list of Hala's projects -->
66
67   <ol>
68     <li>
69       <h5>-نظام لحجز تذاكر القطارات (مسارع جافا)</h5>
70     </li>
71     <li>
72       <h5>-نظام لطلب ووصول البضائع (مسارع جافا)</h5>
73     </li>
74     <li>
75       <h5>-نظام يظهر ويعمل استخدام الانترنت في الدول (مسارع هياكل البيانات)</h5>
76     </li>
77     <li>
78       <h5>-نظام لعمادار الكهرباء في غزة (مسارع هياكل البيانات)</h5>
79     </li>
80   </ol>
81
82
83
84   <div class="box-info" id="boxinfo2">
85     <h4><u><span style="color: #1a237e;">دعاة الختو</span></u></h4>
86     <h4><u><span style="color: #1a237e;">الاسم</span></u></h4>
87     <h4><u><span style="color: #1a237e;">الرقم الجامعي</span></u></h4>
88     <h4><u><span style="color: #1a237e;">1211088</span></u></h4>
89   </div>
```

The code includes Arabic text such as "البرمجة", "المشاريع", "نظام لحجز تذاكر القطارات", "نظام لطلب ووصول البضائع", "نظام يظهر ويعمل استخدام الانترنت في الدول", "نظام لعمادار الكهرباء في غزة", and "دعاة الختو". It also includes English text like "projects", "list", "ol", "li", "h5", "div", "u", and "span". The code is styled with colors like "#1a237e" and "rgb(161, 16, 115)". The code editor interface shows tabs for main.py, WebServer.py, file.html, main_en.html, NotFound.html, main_ar.html, and style.css. The status bar at the bottom shows file statistics (192:1 CRLF, 184 lines), Python version (Python 3.11 (pythonProject)), system information (52°F Partly cloudy, 10:09 PM, 12/21/2023), and system icons.

The screenshot shows a code editor window with a dark theme. The file being edited is named 'main_ar.html'. The code contains bidirectional text, with some parts in Arabic and some in English. A tooltip at the top of the editor bar indicates: 'Visual layout of bidirectional text can depend on the base direction (View | Bidi Text Base ...)'.

```
59     <li><u><span style="color: #800080; font-weight: bold;">البرمجةالمشاريعدعاة الختوالاسمالرقم الجامعي

The status bar at the bottom shows the following information: 192:1 CRLF UTF-8 4 spaces Python 3.11 (pythonProject) 52°F Partly cloudy 10:09 PM 12/21/2023 ENG


```


Visual layout of bidirectional text can depend on the base direction (View | Bidi Text Base ...)

```
117     </li>
118 </ol>
119
120     </div>
121     <div class="box-info" id="boxinfo3">
122         <h4><u><span style="color: #800080;"> Raghad عبدي المسمى الجامعي الهوايات المشاريع
```

The screenshot shows a code editor window with a dark theme. The top bar includes tabs for 'main.py', 'WebServer.py', 'file.html', 'main_en.html', 'NotFound.html', 'main_ar.html' (which is currently selected), and 'style.css'. A notification bar at the top indicates 'Visual layout of bidirectional text can depend on the base direction (View | Bidi Text Base ...)' with options to 'Choose direction', 'Hide notification', and 'Don't show again'. The main area displays the following code:

```
146     </li>
147     <li>
148         <h5>-عيادة الأسنان (يمان قواعد البيانات)</h5>
149     </li>
150     <li>
151         <h5>-تحويل السنن (يمان هيكل البيانات)</h5>
152     </li>
153     <li>
154         <h5>-SM (مساق الأبناء) (وسائل لابد الفاتحة للاستخدام)</h5>
155     </li>
156 </ol>
157
158 </div>
159 </div>
160
161 <div id="box4">
162     <!--we used element mark to highlight the text -->
163     <h4><mark><u> نوع المحتوى</u></mark></h4>
164     <p>
165         هذا هو حقل ترويسة يستخدم لتحديد نوع الوسائط للبيانات الكامنة في الرسالة، مما يحدد تنسيق الكيان.
166         يمكننا الحصول على نوع الوسائط العشار (البيه من خلال أليات فك الترميز التي تم ب بواسطة ترميز المحتوى.
167     </p>
168     <h4><mark><u> يفتح اليه في طلب HTTP لأن:</u></mark></h4>
169     <ol>
170         <li>
171             يستخدم HTTP مفهول رأس نوع المحتوى لتوفير تحديد البيانات وفتحها وجعلها قابلة للتعديل.
172         </li>
173         <li> فإنها أمر جيد لأنها تساعد الجهة المستلمة (العميل أو الخادم) على فهم كيفية تحليل ومعالجة المحتوى.
174     </ol>
175 </div>
```

The code is primarily in English, with some Arabic text highlighted in red. The Arabic text includes: 'عيادة الأسنان' (Dental Clinic), 'تحويل السنن' (Orthodontics), 'SM' (Masq al-Aban), 'نوع المحتوى' (Content Type), 'يفتح اليه في طلب HTTP لأن:' (It opens in when an HTTP request is made because:), and 'فإنها أمر جيد لأنها تساعد الجهة المستلمة' (It is good because it helps the receiving party). The code editor also shows a toolbar with icons for various applications like PC, Google Chrome, and Firefox.

```
Current File < main.py < WebServer.py < file.html < main_en.html < NotFound.html < main_ar.html < style.css < ... < Choose direction < Hide notification < Don't show again
```

184 ^
184 هذا هو حقل تبويبسة يُستخدم لتحديد نوع الوسائط للبيانات الكامنة في الرسالة، مما يحدد تنسيق الكيان.
184 على نوع الوسائط المفهوم الذي من خلال البيانات تلك التغذية التي تتم بواسطة توسيع المحتوى.
184

```
164 <p>  
165     هذا هو حقل تبويبسة يُستخدم لتحديد نوع الوسائط للبيانات الكامنة في الرسالة، مما يحدد تنسيق الكيان.  
166     على نوع الوسائط المفهوم الذي من خلال البيانات تلك التغذية التي تتم بواسطة توسيع المحتوى.  
167 </p>  
168 <h4><mark><u>نحتاج إليه</u></mark></h4>  
169 <ol>  
170     <li>  
171         يستخدم HTTP جنول رأس نوع المحتوى لتوفير تحديد البيانات وفتحها وجعلها قابلة للتتعديل.  
172     </li>  
173     <li>  
174         قاتلها أمر جنوي لأنها تساعد الجهة المستلمة (العميل أو الخادم) على فهم كيفية تحليل ومعالجة المحتوى.  
175     </ol>  
176 </div>  
177  
178 <h2>  
179     <span id="link1">  
180         <!-- the 'a' element for hyperlinks -->  
181         <button id="button1" onclick="window.open('http://127.0.0.1:5500/ar.html', '_blank')> HTML File<button>  
182     </span>  
183 </h2>  
184 <h2>  
185     <span id="link2">  
186         <button id="button2" onclick="window.open('https://www.w3schools.com/python/python_strings.asp', '_blank')> W3Schools<button>  
187     </span>  
188 </h2>  
189 </body>  
190 </html>
```

192:1 CRLF UTF-8 4 spaces Python 3.11 (pythonProject) 52°F Partly cloudy 10:10 PM 12/21/2023

This HTML document creates an Arabic webpage that introduces a course project related to (بخام ويب صغير). It provides information about the team, including names, IDs, hobbies, and projects. Additionally, it includes links to external resources, specifically a W3Schools page and an HTML file. The styling is externally handled by a CSS file.

styleAr.css code:

```
Current File < > file.html < > main_en.html < > NotFound.html < > main_ar.html style.css styleAr.css 4 5 19 ^ ^ ec
32 #boxinfo-container {
33     display: flex;
34     justify-content: space-between;
35 }
36
37 .box-info {
38     padding: 10px;
39     border: 5px solid;
40     width: 30%;
41     font-size: 18px;
42     margin-top: 30px;
43     box-sizing: border-box;
44     border-radius: 15px;
45     overflow: hidden;
46 }
47 /* we used some features to make the web page look nicer and more organized */
48
49 #box4 {
50     direction: rtl;
51     font-family: "Calibri", Times, serif;
52     font-style: italic;
53     margin: 20px auto;
54     border: 0px;
55     width: 80%;
56     font-size: 22px;
57     color: black;
58     padding: 10px;
59     background-image: url(teddy.png);
60     background-size: cover;
61     border-radius: 15px;
62     overflow: hidden;
}
#box4
```

```
WebServer.py      <> file.html      <> main_en.html      <> NotFound.html      <> main_ar.html      style.css      styleAr.css
62     overflow: hidden;
63     margin-bottom: 150px;
64     font-weight: bold;
65 }
66
67 /* insted of repeating same features we used the three classes togather */
68 #box1, #box2, #box3 {
69     border: 5px solid;
70     width: 250px;
71     height: 250px;
72     font-size: 22px;
73     margin-top: 100px;
74     margin-bottom: 30px;
75     border-radius: 15px;
76     overflow: hidden;
77 }
78
79 .highlight-text {
80     color: rgb(201, 182, 182);
81 }
82 /* specify the background image and margins for each box */
83
84 #box1 {
85     margin-left: 120px;
86     background-image: url(Hala.jpg);
87     background-size: cover;
88 }
89
90 #box2 {
91     margin-left: 250px;
92     background-image: url(Doaa.jpg);
#box4
```

62:22 CRLF UTF-8 4 spaces Python 3.11 (pythonProject) 10:46 PM
USD/JPY -0.92% ENG 12/21/2023 12/21/2023

The screenshot shows a code editor interface with the following details:

- File Explorer:** Shows files like WebServer.py, file.html, main_en.html, NotFound.html, main_ar.html, style.css, and styleAr.css.
- Code Editor:** Displays the content of styleAr.css. The code includes:
 - Styling for `li` elements with `margin-bottom: 0px;`.
 - A comment: `/* instead of repeating same features we used the three classes together */`.
 - Styling for `#boxinfo1` with `font-style: italic;`, `border-color: rgb(201, 182, 182);`, `background-color: rgb(201, 182, 182);`, `margin-left: 30px;`, and `color: black;`.
 - Styling for `#boxinfo2` with `font-style: italic;`, `border-color: rgb(201, 182, 182);`, `background-color: rgb(201, 182, 182);`, `margin-left: 70px;`, and `color: black;`.
 - Styling for `#boxinfo3` with `font-style: italic;`, `border-color: rgb(201, 182, 182);`, `background-color: rgb(201, 182, 182);`, `margin-left: 100px;`, and `color: black;`.
 - A final comment: `/* instead of repeating same features we used the main class button */`.
- Status Bar:** Shows the current time (62:22), file encoding (CRLF), character encoding (UTF-8), indentation (4 spaces), Python version (Python 3.11 (pythonProject)), and system information (Windows 10, 53°F Partly cloudy, ENG, 10:48 PM, 12/21/2023).

The screenshot shows a code editor interface with a dark theme. The top bar includes tabs for "WebServer.py", "file.html", "main_en.html", "NotFound.html", "main_ar.html", "style.css", and "styleAr.css". The status bar at the bottom displays "62:22 CRLF UTF-8 4 spaces Python 3.11 (pythonProject) 10:48 PM 12/21/2023". The main area contains the following CSS code:

```
154 button {  
155     font-size: 30px;  
156     margin-top: 20px;  
157     font-style: italic;  
158     font-weight: bold;  
159     font-family: "Calibri", Times, serif;  
160     padding: 8px;  
161     border-radius: 10px; /* Adjust the border-radius as needed */  
162     color: rgb(25,25,112);  
163     align-self: center;  
164 }  
165  
166 #button1 {  
167     margin-left: 700px;  
168 }  
169  
170 #button2 {  
171     margin-left: 680px;  
172 }  
173  
174 #box0 {  
175     direction: rtl;  
176     font-family: 'Calibri',Papyrus;  
177     margin-top: 150px;  
178     margin-left: 10px;  
179     margin-right: 20px;  
180     border: 0px;  
181     width: 500px;  
182     height: 200px;  
183     font-size: 50px;  
184     padding: 30px;  
#box4
```

A screenshot of a Windows desktop environment. In the foreground, a code editor window is open, displaying a CSS file named 'styleAr.css'. The code contains several CSS rules for elements like '#box0', '#intro', and '#intro-container'. The code editor has a dark theme with syntax highlighting. In the background, the Windows taskbar is visible, showing icons for File Explorer, Mail, Photos, Task View, Start, Edge, Google Chrome, and FileZilla. The system tray at the bottom right shows the date and time (12/21/2023, 10:49 PM), battery status (53°F), and connectivity information.

```
172
173
174 #box0 {
175     direction: rtl;
176     font-family: 'Calibri ', 'Papyrus';
177     margin-top: 150px;
178     margin-left: 10px;
179     margin-right: 20px;
180     border: 0px;
181     width: 500px;
182     height: 200px;
183     font-size: 50px;
184     padding: 30px;
185     color: white;
186 }
187
188 #intro {
189     background-image: url(globe.png);
190     background-size: cover;
191     width: 550px;
192     height: 550px;
193     margin-left: 10px;
194     margin-right: 250px;
195     margin-top: 100px;
196 }
197
198 #intro-container {
199     display: flex;
200 }
```

Web Server

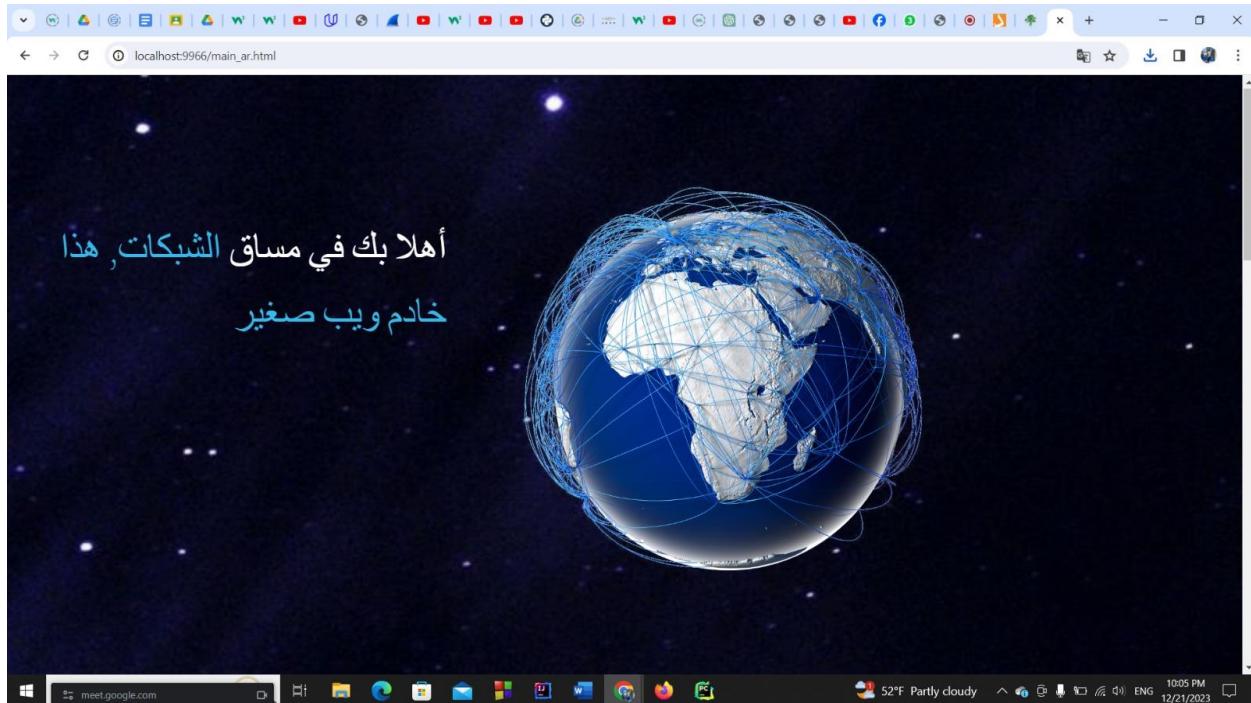


Figure 41:Arabic webserver

A screenshot of a web browser window showing an Arabic web server displaying three user profiles. The title bar says "localhost:9966/main_ar.html". Each profile card contains a photo of a person, their name, ID number, contact information, and a list of interests. The profiles are: 1. حلا قرط (ID 1220062), 2. دعاء الحنو (ID 1211088), and 3. رعد عويسى (ID 1200710). Each card also lists "المهارات" (Skills) and "المشاريع" (Projects). The bottom of the screen shows a Windows taskbar with various icons and system status.

المشاريع:

- نظام لحجز تذاكر العطار (مساف حافا)
- نظام لطلب ووصول البسرا (مساف حافا)
- نظام ظهر وعدل استخدام الإنترنط في الدول (مساف هياكل البيانات)
- نظام ظهر وعدل استخدام الإنترنط في غربة (مساف هياكل البيانات)
- نظام لمصادر الكهرباء في مصر (مساف هياكل البيانات)

المشاريع:

- نظام توصيل المطعم السيرا (مساف حافا)
- نظام مدرسي للطلاب (مساف هياكل البيانات والخوارزميات)
- الله حاسبي سبطه (مساف هياكل البيانات والخوارزميات)
- خوارزميات الجدوله (مساف أنظمة التشغيل)
- مشروع محاكاة دوائر الدايمود (مساف الألكترونيات)

المشاريع:

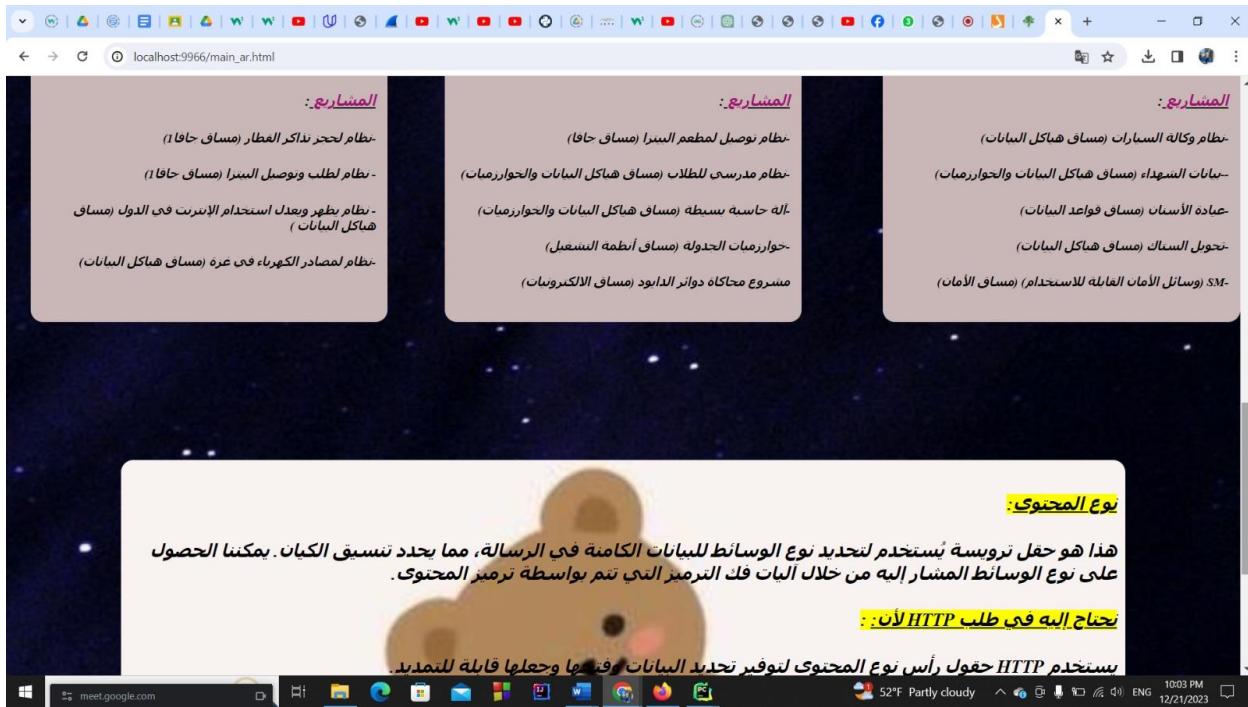
- نظام وكالة المسارات (مساف هياكل البيانات)
- بيانات الشهداء، (مساف هياكل البيانات والخوارزميات)
- عيادة الأسنان (مساف قواعد البيانات)
- تحويل السنات (مساف هياكل البيانات)
- SM- (وسائل الأمان القابلة لل باستخدام) (مساف الأمان)

نوع المحتوى:

هذا هو حقل ترويسة يستخدم لتحديد نوع الوسائط للبيانات الكامنة في الرسالة، مما يحدد تنسيق الكيان. يمكننا الحصول على نوع الوسائط المشار إليه من خلال الآيات فك الترميز التي تم بواسطة ترميز المحتوى.

نجاح إليه في طلب HTTP لأن:

يستخدم HTTP حقول رأس نوع المحتوى لتوفير تحديد البيانات، فرقها وجعلها قابلة للتمديد.



نوع المحتوى:

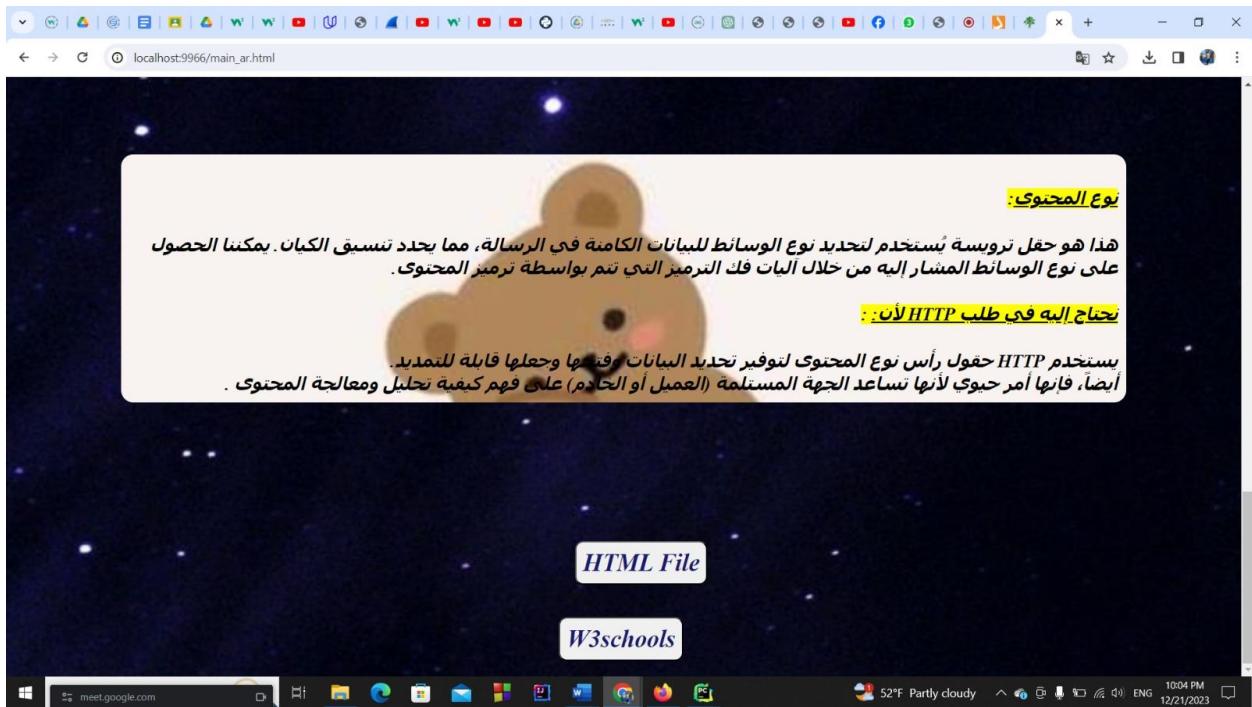
هذا هو حقل ترويسة يستخدم لتحديد نوع الوسائط للبيانات الكامنة في الرسالة، مما يحدد تنسيق الكيان. يمكننا الحصول على نوع الوسائط المشار إليه من خلال الآيات فك الترميز التي تم بواسطة ترميز المحتوى.

نجاح إليه في طلب HTTP لأن:

يستخدم HTTP حقول رأس نوع المحتوى لتوفير تحديد البيانات، فرقها وجعلها قابلة للتمديد. أيضا، فإنه أمر حيوي لأنها تساعد الجهة المستلمة (العميل أو الخادم) على فهم كيفية تحليل ومعالجة المحتوى.

HTML File

W3schools



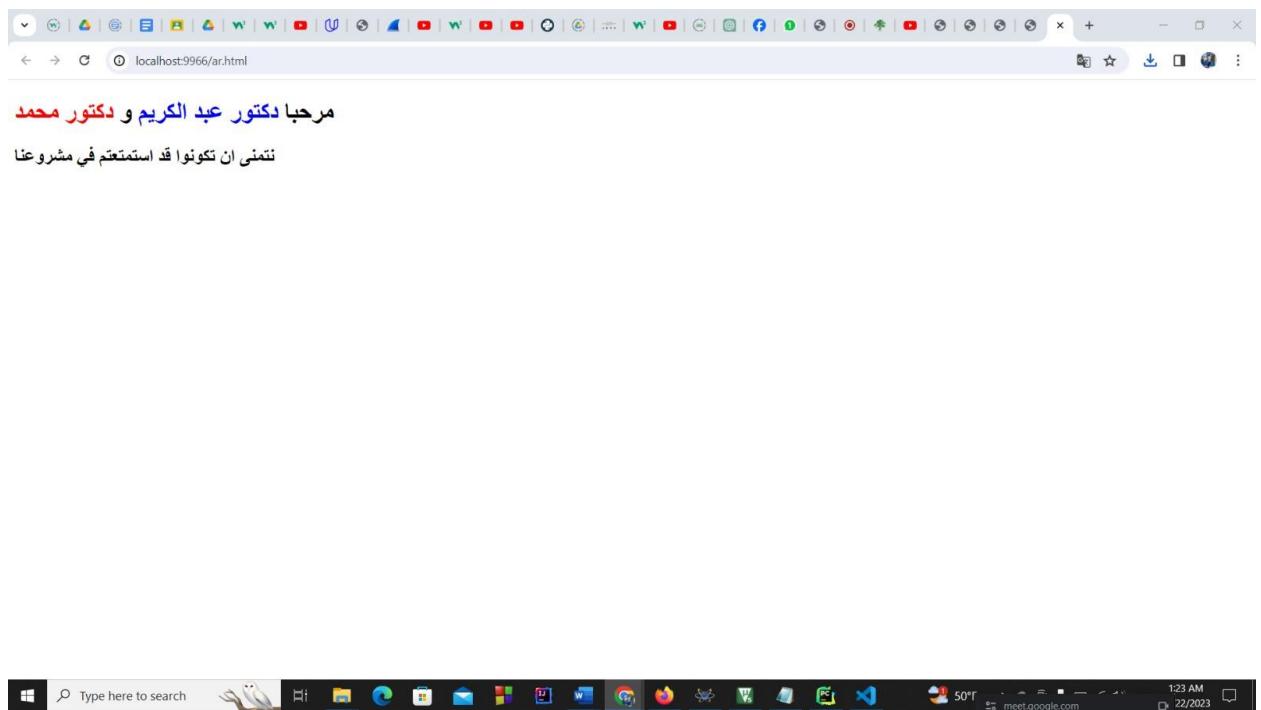


Figure 42:html Arabic file

Testing the website from another device

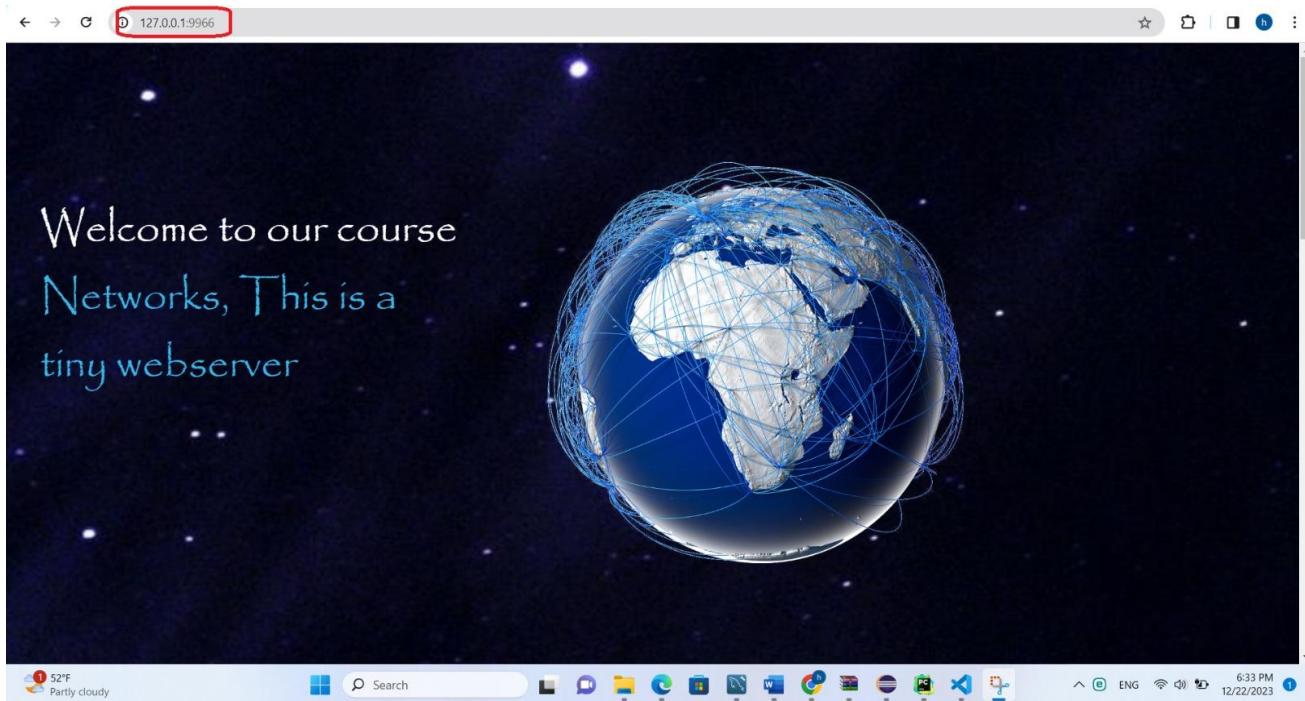


Figure 43:Test from another device