



**Faculty of Engineering & Technology – Computer Systems
Engineering Department**

Second Semester 2023-2024

Computer Network ENCS3320

Project – 1-

Prepared by: Rasha Daoud - 1210382

: Nadia Thaer - 1210021

Section: 2

Instructor: Abdalkarim Awad

Date: 5th May 202

Table of Content :

Part 1.....	3
1.1Ping.....	3
1.2Tracert.....	3
1.3.NSLookup.....	3
1.4.Telnet.....	3
Part one 2	4
2.1.Ping a device in the same network (from the laptop to the smartphone).	4
2.2.Ping www.stanford.edu	5
2.3.From the ping results, do you think the response you got is from USA? Explain your answer briefly	6
2.4.Tracert www.stanford.com.....	7
2.5 nslookup www.stanford.com.....	8
Part one 3. use wireshark to capture some DNS messages.....	9
Part 2.....	10
Part 3.....	14
3.0. Entity Tag Cache Validators in The HTTp protocol	14
3.1 Code and Explaination	14
3.1.1 The main.py	14
3.1. if the request is / or /index.html or /main_en.html or /en (for example localhost:6060/ or localhost:6060/en)	22
3.1.1 If I request /index.html	22
3.1.2 request / main_en.html	23
3.1.3 request /en	24
3.1.4 . request /	25
3.2. Requested /ar and main_ar.....	26
3.2.1 requeste the /ar	26
3.2.2 request the /main_ar	27
3.3 if the request Was rasha.html	28
3.4 .if request c.css	29
3.5 if I request p.png	30
3.6 if I request j.jpg	31
3.7 MyForm	32
3.8 if I request /so	33
3.9 if I request /itc	34
3.10. Request any File Does Not Contain in The Server.....	35

List of Figure

Figure 1:Ping a device in The Same Network from Labtop To Smart Phone	4
Figure 2:Ping www.stanford.....	5
Figure 3:Tracert www.standford.com	7
Figure 4:Nslookup www.standford.com	8
Figure 5:Result From Wireshark.....	9
Figure 6:part 2 Code	10
Figure 7:part2 code	10
Figure 8:part 2 Code.	11
Figure 9:part 2 code.	11
Figure 10:part 2 Code.	12
Figure 11:part(3)→Code of mainServer (1).....	14
Figure 12:part (3)→code of mainSer (2).....	14
Figure 13: part 3→mainSer code (3).	15
Figure 14: part 3->the mainServer code (4).....	16
Figure 15:part (3) -->mainSer Code (4).....	17
Figure 16:Main_en Code→> Part 1	18
Figure 17:Maine_en Cod -->Part 2	18
Figure 18:Main_ar.HTml Code 1.....	19
Figure 19:Main_ar.html code 2.....	20
Figure 20:css Code 1.....	20
Figure 21css code 2:	21
Figure 22:My Form -->1	21
Figure 23:MyFor code-->2	21
Figure 24:request /index.html Result	22
Figure 25:request /index.html Http request.....	22
Figure 26:request / main_en.html Result	23
Figure 27:request / main_en.html Http request	23
Figure 28:request /en Result	24
Figure 29:request /en Http Request Massge	24
Figure 30:request / Result	25
Figure 31:request / Http request massge	25
Figure 32:requeste the /ar Result	26
Figure 33:The Http request For -->requeste the /ar	26
Figure 34: request the /main_ar Result	27
Figure 35: request the /main_ar Http request massge	27
Figure 36:the request Was rasha.html Result	28
Figure 37:the request Was rasha.html Request Massge	28
Figure 38:request c.css Result	29
Figure 39:request c.css Http Massge Requested	29
Figure 40:request p.png Result	30
Figure 41:request p.png Massge Request	30
Figure 42:request j.jpg Result	31
Figure 43:request j.jpg HTTP requested Massge	31
Figure 44:My Form Result	32
Figure 45:MyForm Result HTTP request	32
Figure 46:request /so Result	33
Figure 47:request /so HTTP requested Massge	33
Figure 48:request /itc Result	34
Figure 49:request /itc Requested massge Http	34
Figure 50:Request any File Does Not Contain in The Server Result	35
Figure 51:Request any File Does Not Contain in The Server Http requested massge	35

Part 1

1.1Ping

Ping is a network utility used to test host reachability on IP networks. It measures the round-trip time for messages sent from one host to another and back, derived from sonar terminology.

1.2Tracert

Tracert, or traceroute, maps the path data takes on the internet from its origin to its destination. It helps find connectivity problems and network slowdowns.

1.3.NSLookup

NSLookup, short for "Name Server Lookup," is a command-line tool used to query Domain Name System (DNS) servers to obtain domain name or IP address information.

1.4.Telnet

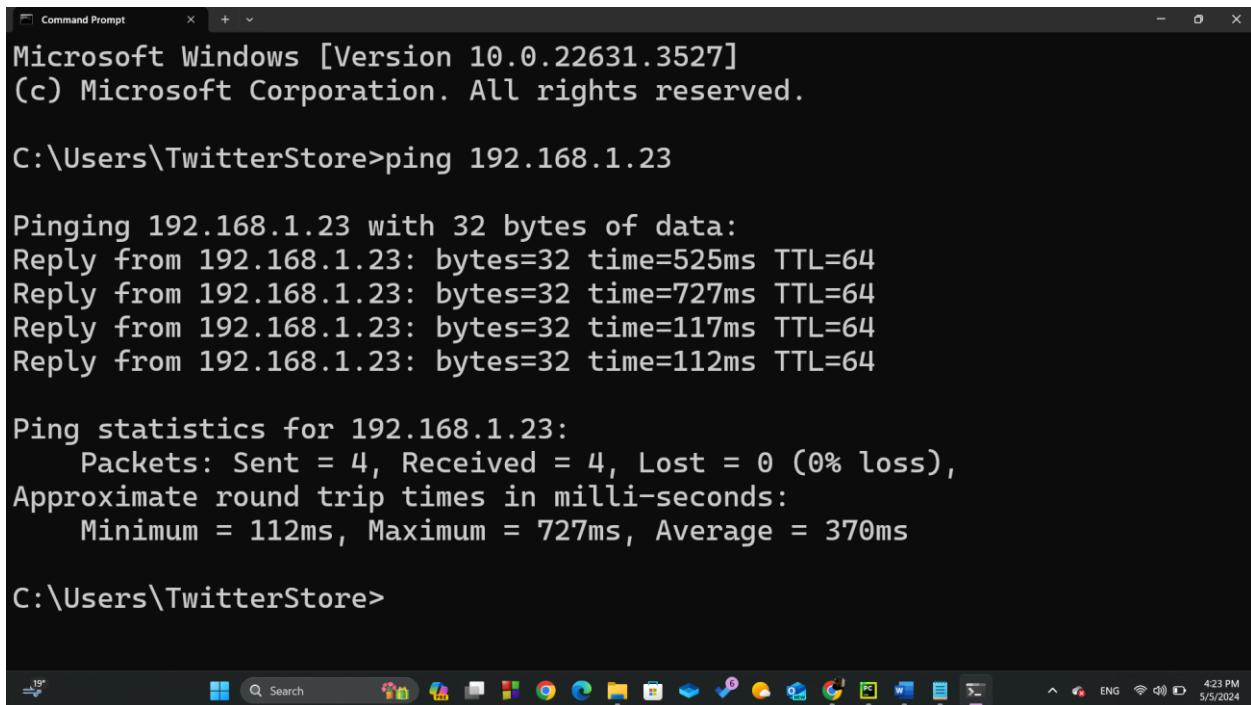
Telnet is a network protocol and command-line tool that allows users to establish a connection to a remote computer or server over a network. It provides a bidirectional interactive text-oriented communication channel between the local and remote systems.

Part one 2

2.1.Ping a device in the same network (from the laptop to the smartphone).

IPv4 Address (Laptop): 192.168.56.1

IPv4 Address (smartphone): 192.168.1.23



```
Microsoft Windows [Version 10.0.22631.3527]
(c) Microsoft Corporation. All rights reserved.

C:\Users\TwitterStore>ping 192.168.1.23

Pinging 192.168.1.23 with 32 bytes of data:
Reply from 192.168.1.23: bytes=32 time=525ms TTL=64
Reply from 192.168.1.23: bytes=32 time=727ms TTL=64
Reply from 192.168.1.23: bytes=32 time=117ms TTL=64
Reply from 192.168.1.23: bytes=32 time=112ms TTL=64

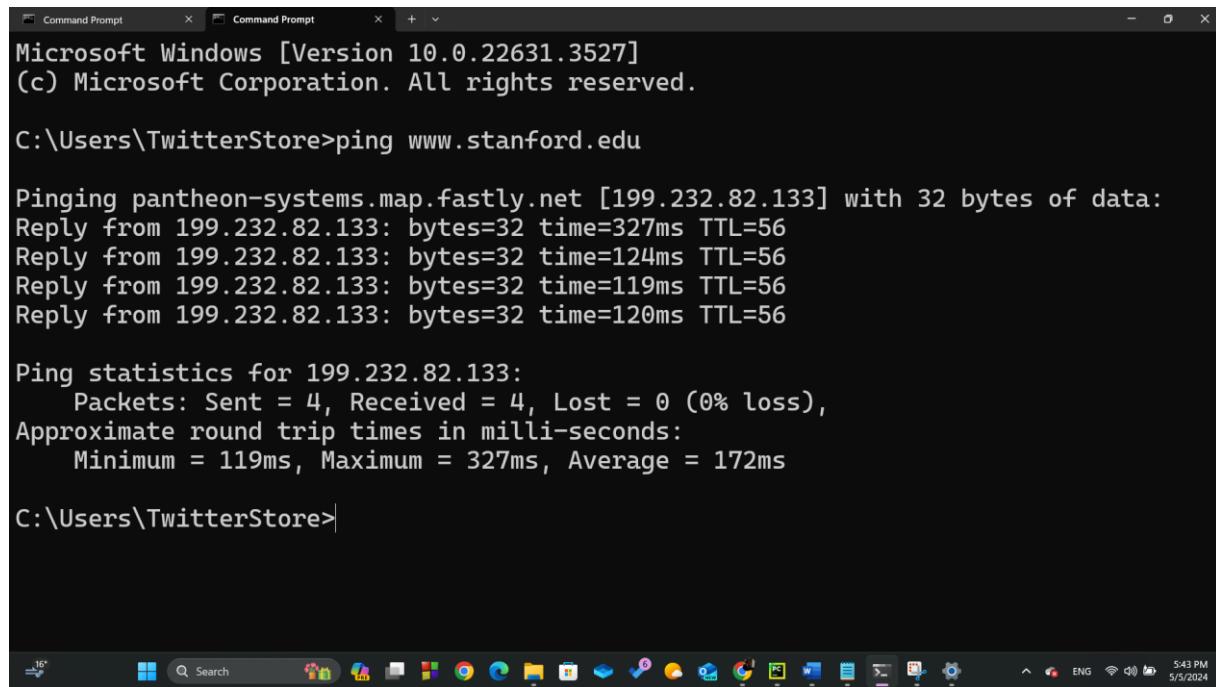
Ping statistics for 192.168.1.23:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 112ms, Maximum = 727ms, Average = 370ms

C:\Users\TwitterStore>
```

Figure 1:Ping a device in The Same Network from Labtop To Smart Phone.

- The ping command is sending ICMP echo requests to the IP address 192.168.1.23 with packets of size 32 bytes.
- Then a reply line show the reply from the IP address 192.168.1.23, indicating that the device at that address responded to the ping request. The time parameter indicates the round-trip time in, and TTL (Time To Live) represents the maximum number of hops the packet can traverse before being discarded.
- And in the next section provides statistics about the ping operation. It shows that 4 packets were sent, all 4 were received successfully (0% loss), and no packets were lost.

2.2.Ping www.stanford.edu



```
Microsoft Windows [Version 10.0.22631.3527]
(c) Microsoft Corporation. All rights reserved.

C:\Users\TwitterStore>ping www.stanford.edu

Pinging pantheon-systems.map.fastly.net [199.232.82.133] with 32 bytes of data:
Reply from 199.232.82.133: bytes=32 time=327ms TTL=56
Reply from 199.232.82.133: bytes=32 time=124ms TTL=56
Reply from 199.232.82.133: bytes=32 time=119ms TTL=56
Reply from 199.232.82.133: bytes=32 time=120ms TTL=56

Ping statistics for 199.232.82.133:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 119ms, Maximum = 327ms, Average = 172ms

C:\Users\TwitterStore>
```

Figure 2:Ping www.stanford.edu.

- The ping command is sending ICMP echo requests to the IP address 199.232.82.133, which corresponds to the domain www.stanford.edu. The IP address is resolved from the domain name using DNS (Domain Name System) resolution.

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

According to the information given in the ping results, the response is highly probable to be from the USA.

The domain www.stanford.edu is hosted on the Fastly content delivery network (CDN), as indicated by the hostname "pantheon-systems.map.fastly.net" in the ping results. Fastly is a globally distributed CDN service provider that accelerates content delivery by caching data closer to end-users worldwide.

Also, if we use a website that check the location of the IP address 199.232.82.133, we get the following details:

Hostname: 199.232.82.133

ASN: 54113

ISP: Fastly Inc.

Services: Datacenter

Assignment: [Likely Static IP](#)

Country: United States

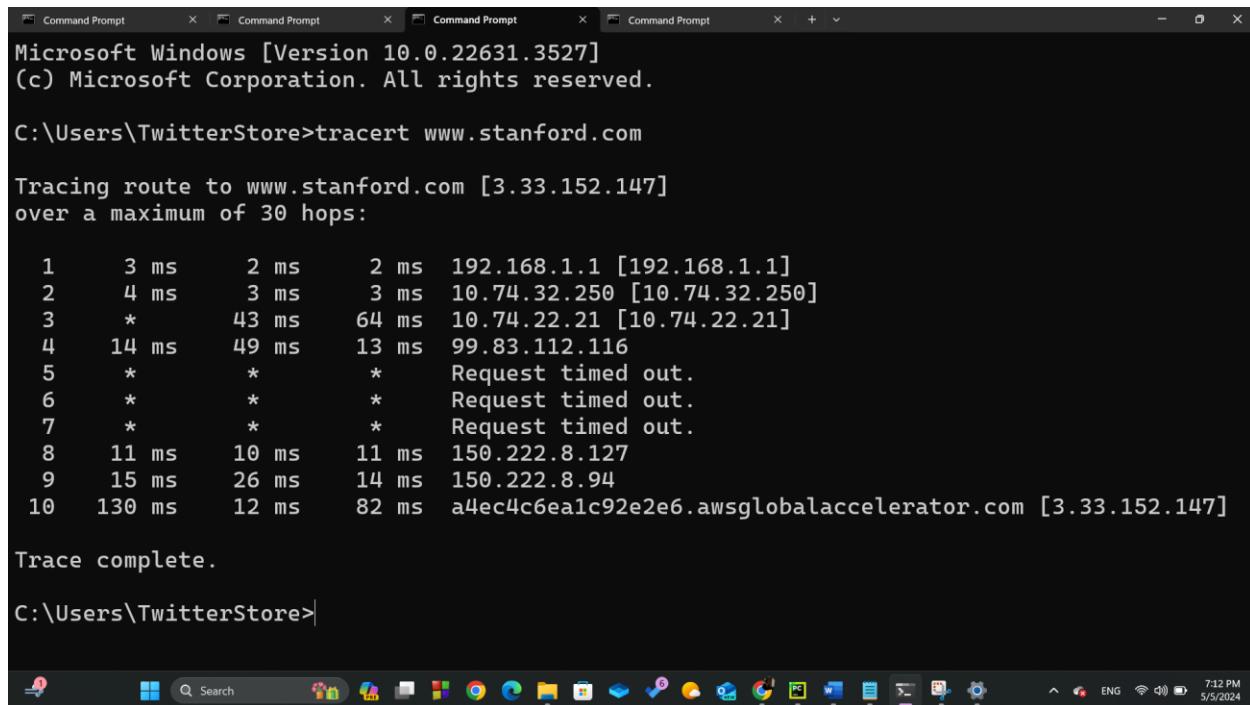
State/Region: California

City: San Francisco

Thus, we can determine that this IP address is allocated in USA.

2.4.Tracert www.stanford.com

The traceroute shows the path taken by network packets from my computer to the destination server at IP address 3.33.152.147. along with the round-trip time (latency) for each hop.



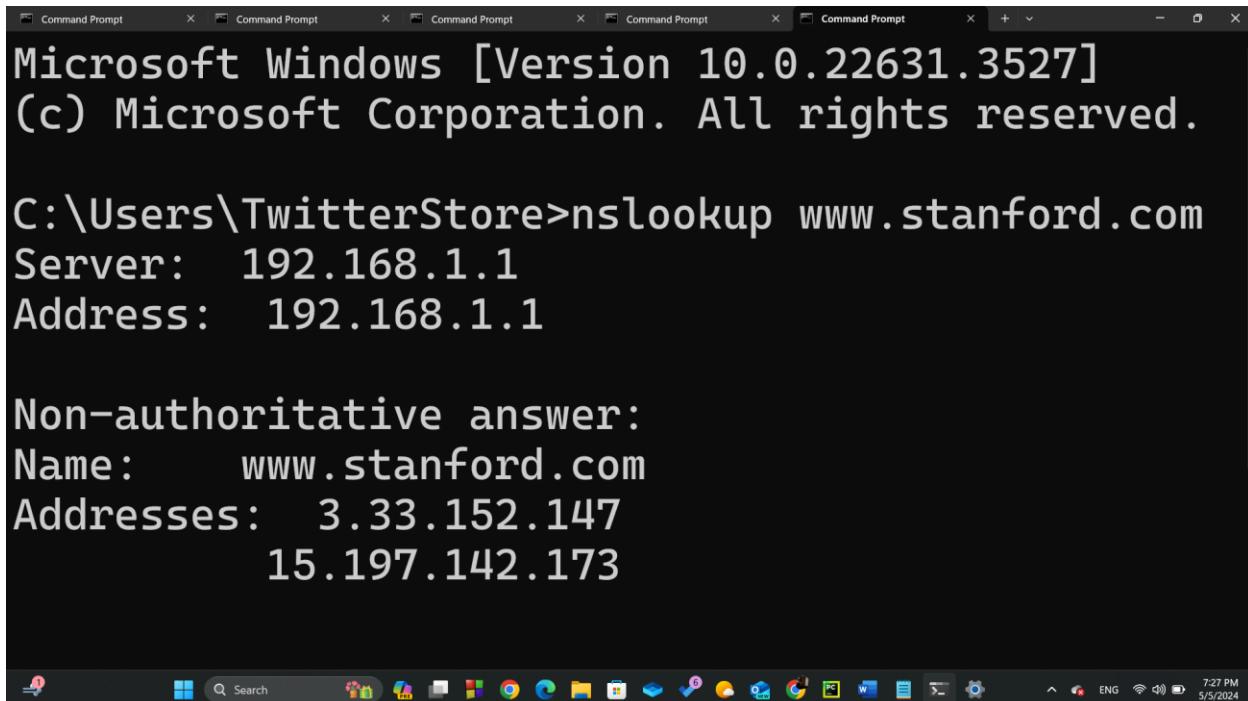
Microsoft Windows [Version 10.0.22631.3527]
(c) Microsoft Corporation. All rights reserved.
C:\Users\TwitterStore>tracert www.stanford.com
Tracing route to www.stanford.com [3.33.152.147]
over a maximum of 30 hops:
1 3 ms 2 ms 2 ms 192.168.1.1 [192.168.1.1]
2 4 ms 3 ms 3 ms 10.74.32.250 [10.74.32.250]
3 * 43 ms 64 ms 10.74.22.21 [10.74.22.21]
4 14 ms 49 ms 13 ms 99.83.112.116
5 * * Request timed out.
6 * * Request timed out.
7 * * Request timed out.
8 11 ms 10 ms 11 ms 150.222.8.127
9 15 ms 26 ms 14 ms 150.222.8.94
10 130 ms 12 ms 82 ms a4ec4c6ea1c92e2e6.awsglobalaccelerator.com [3.33.152.147]
Trace complete.
C:\Users\TwitterStore>

Figure 3:Tracert www.stanford.com

In some cases, the traceroute may encounter hops that do not respond to the traceroute request, resulting in "Request timed out" messages. This could be due to network congestion, firewall configurations, or other reasons.

The last line of the traceroute output displays the IP address of the destination server (3.33.152.147) and its hostname (a4ec4c6ea1c92e2e6.awsglobalaccelerator.com), indicating that the traceroute reached its final destination.

2.5 nslookup www.stanford.com



```
Microsoft Windows [Version 10.0.22631.3527]
(c) Microsoft Corporation. All rights reserved.

C:\Users\TwitterStore>nslookup www.stanford.com
Server: 192.168.1.1
Address: 192.168.1.1

Non-authoritative answer:
Name: www.stanford.com
Addresses: 3.33.152.147
           15.197.142.173
```

Figure 4:Nslookup www.stanford.com

The nslookup command for www.stanford.com provides information about the domain name and its associated IP addresses.

Server: 192.168.1.1: This line indicates the DNS server used for the nslookup. And the DNS server at IP address 192.168.1.1 is providing the response.

Non-authoritative answer: This message indicates that the response comes from a DNS server other than the authoritative DNS server for the domain.

www.stanford.com resolves to two IP addresses: 3.33.152.147 and 15.197.142.173. These are the destinations that the computer would communicate with when accessing www.stanford.com over the internet.

Part one 3. use wireshark to capture some DNS messages

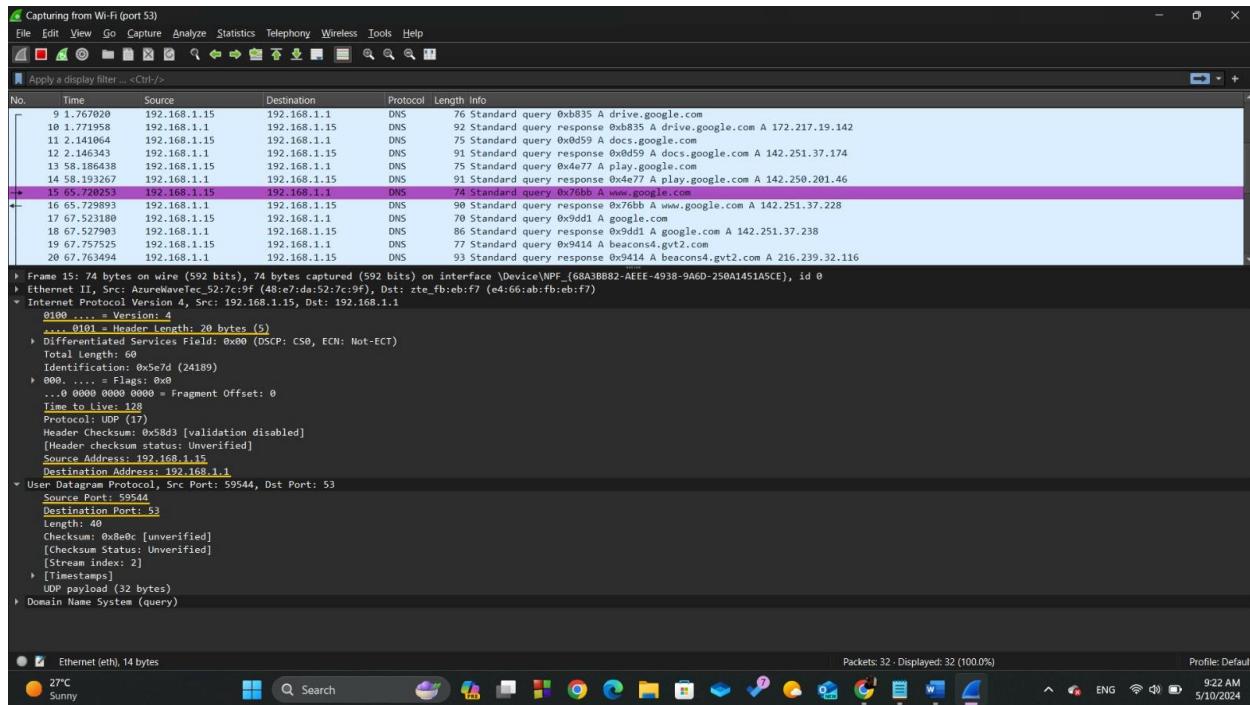
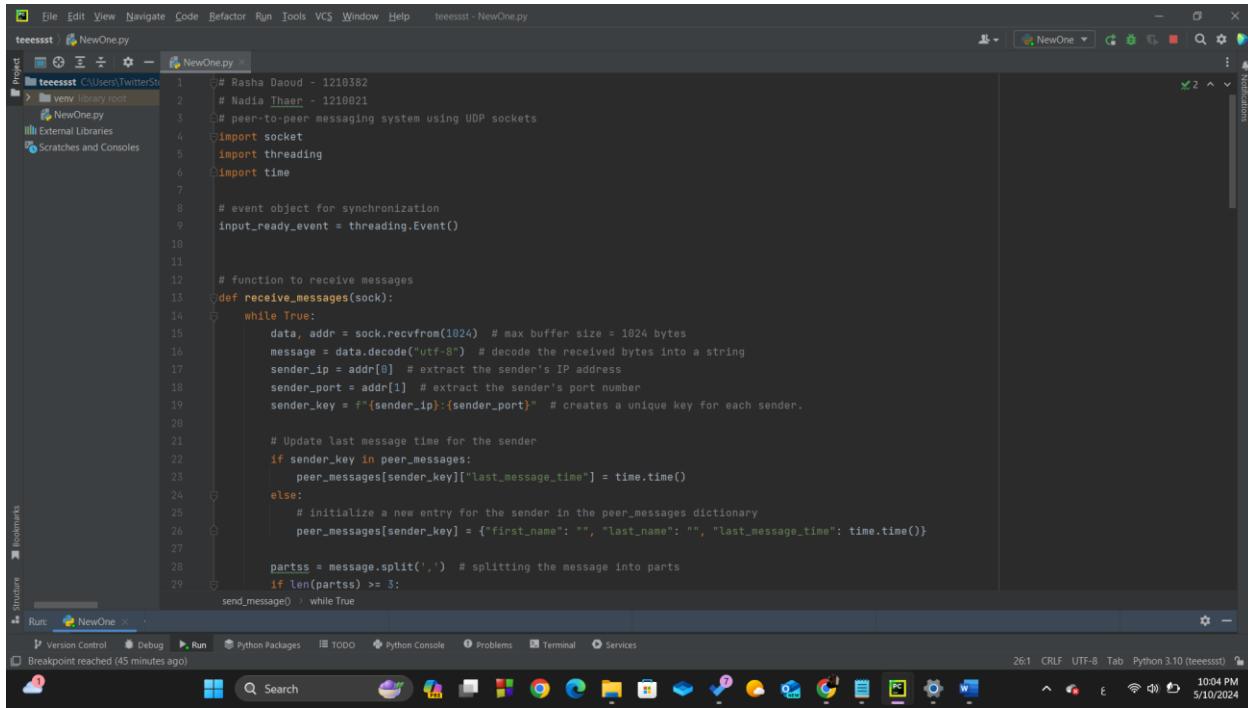


Figure 5:Result From Wireshark.

1. Version: the internet protocol addressing version is IPV4.
2. Header Length: the total length of the header is 20 bytes.
3. TTL: The packet can travel through 128 routers.
4. Source Address: the local IP address of my PC = 192.168.1.15.
5. Destination Address: the IP address of the destination = 192.168.1.1.
6. Source Port: the source port of the packets = 59544
7. Destination Port: the destination port of the packets = 53.

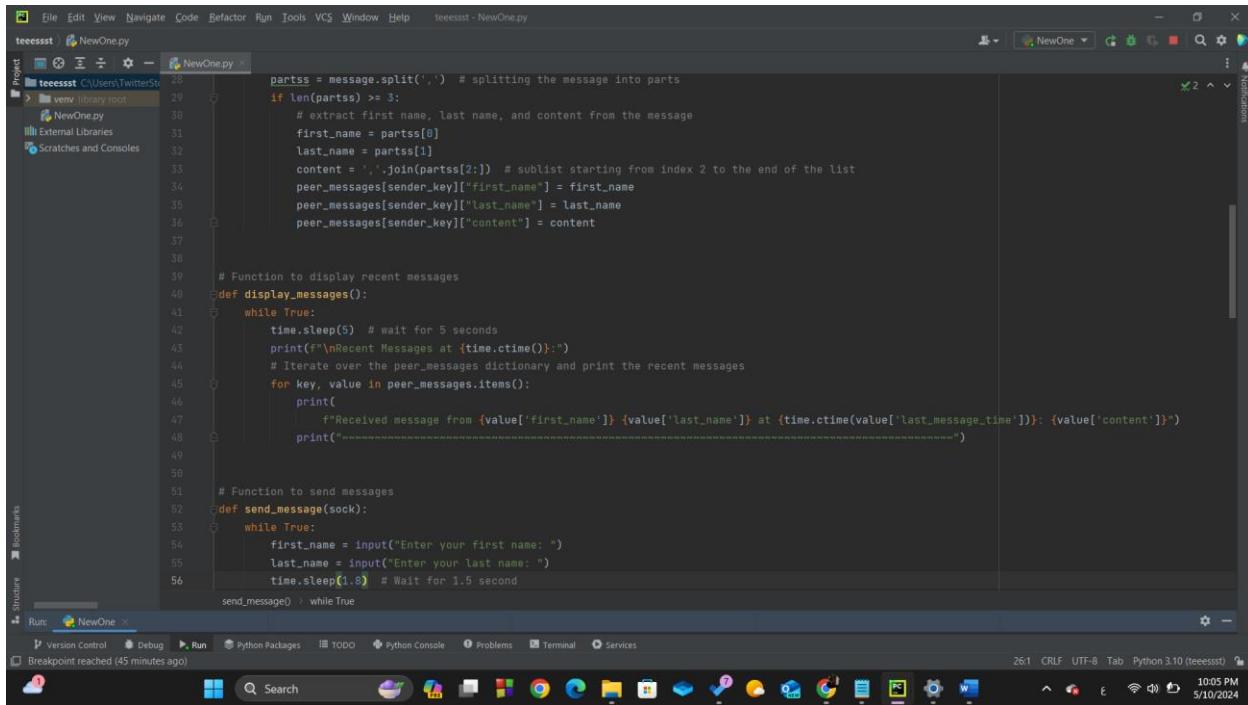
Part 2



The screenshot shows the PyCharm IDE interface with the following details:

- File Menu:** File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Project:** teessst, NewOne.py.
- Code Editor:** The file NewOne.py contains Python code for a peer-to-peer messaging system using UDP sockets. The code includes functions for receiving messages, displaying recent messages, and sending messages.
- Toolbars:** Version Control, Debug, Run, Python Packages, TODO, Python Console, Problems, Terminal, Services.
- Status Bar:** Breakpoint reached (45 minutes ago), 26:1 CRLF, UTF-8 Tab, Python 3.10 (teessst), 10:04 PM, 5/10/2024.

Figure 6:part 2 Code .



The screenshot shows the PyCharm IDE interface with the following details:

- File Menu:** File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help, teessst - NewOne.py.
- Project:** teessst, NewOne.py.
- Code Editor:** The file NewOne.py contains Python code for a peer-to-peer messaging system using UDP sockets. The code includes functions for receiving messages, displaying recent messages, and sending messages.
- Toolbars:** Version Control, Debug, Run, Python Packages, TODO, Python Console, Problems, Terminal, Services.
- Status Bar:** Breakpoint reached (45 minutes ago), 26:1 CRLF, UTF-8 Tab, Python 3.10 (teessst), 10:05 PM, 5/10/2024.

Figure 7:part2 code

```
last_name = input("Enter your last name: ")
time.sleep(1.8) # Wait for 1.5 second
message = input("Enter your message: ")
time.sleep(1.5) # Wait for 1.5 second
full_message = f'{first_name},{last_name},{message}' # create the full message
message_bytes = full_message.encode("utf-8")
broadcast_addr = ('192.168.1.255', 5051) # Broadcast address for peers
sock.sendto(message_bytes, broadcast_addr) # send the message to all peers

# Function to display the content of a message from a peer based on line number
def display_message_content():
    while True:
        input_ready_event.wait() # Wait until input is ready
        try:
            line_input = input("Enter the line number of the message to display (e.g., 20): ")
            line_number, action = line_input.split('D') # Split the input into line number and action
            line_number = int(line_number.strip()) # Extract the line number
            if action.strip().lower() != 'd': # Check if the action is 'D' (display)
                print("Invalid action!!! Please enter 'D' to display the message.")
                continue
            if line_number < 1 or line_number > len(peer_messages): # Check if the line number is valid
                print("Invalid line number!!! Please enter a valid line number.")
                continue
            key, value = list(peer_messages.items())[line_number - 1] # Get the message based on line number
            print(f"\nMessage content from {value['first_name']} {value['last_name']}:\n{value['content']}\n")
        except ValueError, IndexError:
            print("Invalid input. Please enter a valid line number followed by 'D'.")
        finally:
            input_ready_event.clear() # Reset the event after processing input

# Main function
def main():
    # Create a UDP socket
    sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    sock.bind(('0.0.0.0', 5051)) # Bind the socket to all available interfaces on port 5051

    # Thread for receiving messages
    receive_thread = threading.Thread(target=receive_messages, args=(sock,))
    receive_thread.start()

    # Thread for displaying messages
    display_thread = threading.Thread(target=display_messages)
    display_thread.start()

    # Thread for sending messages
    send_thread = threading.Thread(target=send_message, args=(sock,))
    send_thread.start()

    # Thread for displaying message content
    display_content_thread = threading.Thread(target=display_message_content)
    display_content_thread.start()

    # Keep the main thread alive
    send_message() > while True
```

Figure 8:part 2 Code.

```
except (ValueError, IndexError):
    print("Invalid input. Please enter a valid line number followed by 'D'.")
finally:
    input_ready_event.clear() # Reset the event after processing input

# Main function
def main():
    # Create a UDP socket
    sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    sock.bind(('0.0.0.0', 5051)) # Bind the socket to all available interfaces on port 5051

    # Thread for receiving messages
    receive_thread = threading.Thread(target=receive_messages, args=(sock,))
    receive_thread.start()

    # Thread for displaying messages
    display_thread = threading.Thread(target=display_messages)
    display_thread.start()

    # Thread for sending messages
    send_thread = threading.Thread(target=send_message, args=(sock,))
    send_thread.start()

    # Thread for displaying message content
    display_content_thread = threading.Thread(target=display_message_content)
    display_content_thread.start()

    # Keep the main thread alive
    send_message() > while True
```

Figure 9:part 2 code.

The screenshot shows the PyCharm IDE interface with the following details:

- File Menu:** File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Project Tree:** A tree view showing the project structure with files like `NewOne.py`, `NewOne.py` (under venv library root), External Libraries, and Scratches and Consoles.
- Code Editor:** The main window displays the `NewOne.py` file content. The code uses threads to handle message receiving, displaying, sending, and content display. It includes a main loop and a dictionary to store messages from peers.
- Run Tab:** Shows the configuration for running the application, with the current tab being `NewOne`.
- Bottom Bar:** Includes icons for Version Control, Debug, Run, Python Packages, TODO, Python Console, Problems, Terminal, and Services. It also shows the current Python version as Python 3.10 (reeessst).
- System Tray:** Shows standard Windows icons for battery, network, and system status.

Figure 10:part 2 Code.

Explanation The Whole Code :

the provided code creates a simple peer-to-peer messaging system using UDP sockets in Python. It allows users to send and receive messages between multiple peers over a local network. The code consists of four main functions: receiving messages, displaying recent messages, sending messages, and displaying the content of a specific message. Each function runs in its own thread to handle concurrent operations. When a message is received, the sender's information and message content are stored in a dictionary. The program continuously displays recent messages and prompts users to send new messages or view specific message contents. Additionally, it binds to a specific IP address and port to listen for incoming messages and uses a broadcast address to send messages to all peers on the network.

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help teessst - NewOne.py
NewOne.py
Project C:\Users\TwitterStore\PycharmProjects\teessst\venv\Scripts\python.exe C:/Users/TwitterStore/PycharmProjects/teessst/NewOne.py
NewOne.py
Visual layout of bidirectional text can depend on the base direction (View | Bidi Text Base Direction)
Choose direction Hide notification Don't show again
53     while True:
54         first_name = input("Enter your first name: ")
55         last_name = input("Enter your last name: ")
send_message() > while True
Run: NewOne x
C:/Users/TwitterStore/PycharmProjects/teessst/venv\Scripts\python.exe C:/Users/TwitterStore/PycharmProjects/teessst/NewOne.py
Enter your first name: Rasha
Enter your last name: Daoud
Recent Messages at Fri May 10 21:40:37 2024:
Enter your message: dfd
Enter your first name:
Recent Messages at Fri May 10 21:40:42 2024:
Received message from Rasha Daoud at Fri May 10 21:40:40 2024: Hi
-----
nadiaff
Enter your last name: ffd
Recent Messages at Fri May 10 21:40:47 2024:
Received message from Rasha Daoud at Fri May 10 21:40:40 2024: Hi
-----
Enter your message: ffd
Enter your first name:
Recent Messages at Fri May 10 21:40:52 2024:
Received message from nadiaff at Fri May 10 21:40:51 2024: dfd
-----
Recent Messages at Fri May 10 21:40:57 2024:
Received message from nadiaff at Fri May 10 21:40:51 2024: dfd
-----
```

Figure 11: The output of part 2.

Part 3

3.0. Entity Tag Cache Validators in The HTTP protocol

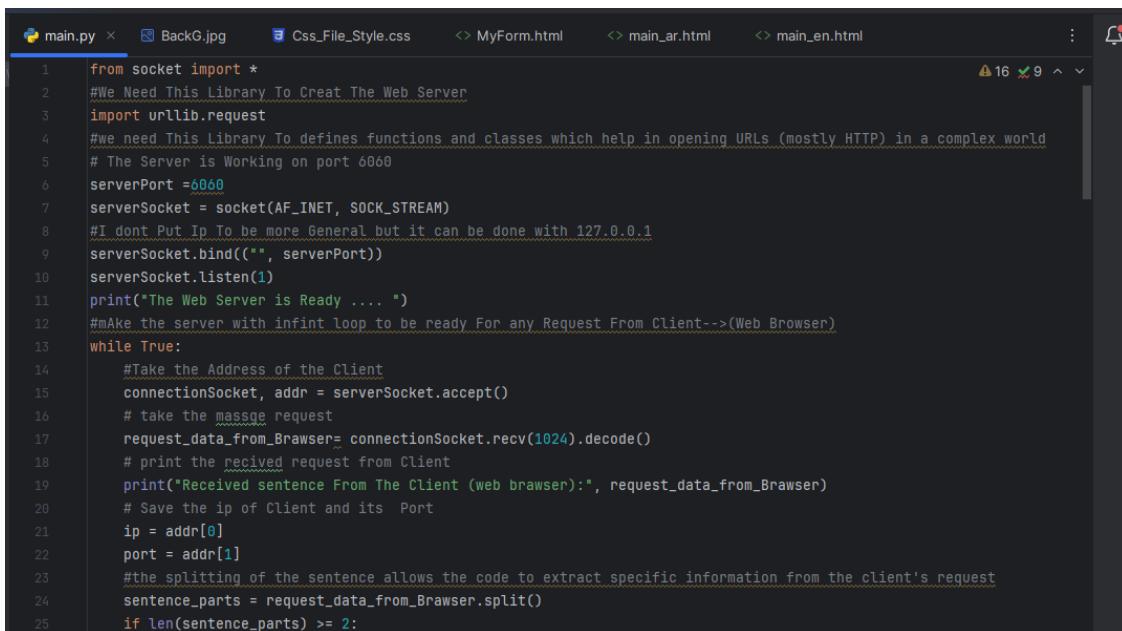
Entity Tag (ETag) cache validators in the HTTP protocol are mechanisms used for cache validation. They provide a way for a client to check if its cached representation of a resource is still valid. ETags are opaque identifiers generated by the server to represent a specific version of a resource.

So According to This information , Entity Tag cache validators enhance the efficiency and reliability of caching mechanisms in the HTTP protocol, improving performance and reducing the load on servers by minimizing unnecessary data transfer.

3.1 Code and Explanation

3.1.1 The main.py

Code For the web Server → server process in Socket at server:



```
 1  from socket import *
 2  #We Need This Library To Create The Web Server
 3  import urllib.request
 4  #we need This Library To defines functions and classes which help in opening URLs (mostly HTTP) in a complex world
 5  # The Server is Working on port 6060
 6  serverPort =6060
 7  serverSocket = socket(AF_INET, SOCK_STREAM)
 8  #I dont Put Ip To be more General but it can be done with 127.0.0.1
 9  serverSocket.bind(("*", serverPort))
10  serverSocket.listen(1)
11  print("The Web Server is Ready .... ")
12  #mAke the server with infinit loop to be ready For any Request From Client-->(Web Browser)
13  while True:
14      #Take the Address of the Client
15      connectionSocket, addr = serverSocket.accept()
16      # take the message request
17      request_data_from_Browser= connectionSocket.recv(1024).decode()
18      # print the received request from Client
19      print("Received sentence From The Client (web browser):", request_data_from_Browser)
20      # Save the ip of Client and its Port
21      ip = addr[0]
22      port = addr[1]
23      #the splitting of the sentence allows the code to extract specific information from the client's request
24      sentence_parts = request_data_from_Browser.split()
25      if len(sentence_parts) >= 2:
```

Figure 12:part(3)→Code of mainServer (1).

```
if len(sentence_parts) >= 2:
    #so the object will be -->/resources-->the place where the server will give me the response
    object = sentence_parts[1]
    print("Requested object:", object)
```

Figure 13:part (3)→code of mainSer (2).

Notice : Each Line in The Code There is An Explanation .

```

30     if object == '/' or object == '/index.html' or object == '/main_en.html' or object == '/en':
31         connectionSocket.send("HTTP/1.1 200 OK \r\n".encode())
32         connectionSocket.send("Content-Type: text/html \r\n".encode())
33         connectionSocket.send("\r\n".encode())
34         file1 = open("main_en.html", "rb")
35         connectionSocket.send(file1.read())
36
37     elif object == '/ar' or object=='/main_ar.html':
38         connectionSocket.send("HTTP/1.1 200 OK \r\n".encode())
39         connectionSocket.send("Content-Type: text/html \r\n".encode())
40         connectionSocket.send("\r\n".encode())
41         file2 = open("main_ar.html", "rb")
42         connectionSocket.send(file2.read())
43
44     elif object.endswith('.html'):
45         connectionSocket.send("HTTP/1.1 200 OK \r\n".encode())
46         connectionSocket.send("Content-Type: text/html \r\n".encode())
47         connectionSocket.send("\r\n".encode())
48         file3 = open("main_en.html", "rb")
49         connectionSocket.send(file3.read())

```

Figure 14: part 3 → mainSer code (3).

- In the first if statement → the request is **/ or /index.html or /main_en.html or /en (for example localhost:6060/ or localhost:6060/en)** then the server should send **main_en.html** file with Content-Type: text/html.
- In the second if statement → if the request is **/ar or main_ar** then the server send the main_ar file
- In the third if statement → if the request end with **.html then** the server will display the main_en.html this mean any request Ended with .html it will display the same file and this appear in our result if we request an **rasha.html** the file response is main_en.

```

50     elif object.endswith('.css'):
51         connectionSocket.send("HTTP/1.1 200 OK \r\n".encode())
52         connectionSocket.send("Content-Type: text/css \r\n".encode())
53         connectionSocket.send("\r\n".encode())
54         file4 = open("Css_File_Style.css", "rb")
55         connectionSocket.send(file4.read())
56
57     elif object.endswith('.png'):
58         url = "https://e7.pngegg.com/pngimages/423/224/png-clipart-desktop-high-definition-television-forest-1080p-1"
59         headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)'}
60         requestFromClient = urllib.request.Request(url, headers=headers)
61         content=urllib.request.urlopen(requestFromClient).read()
62         connectionSocket.send("HTTP/1.1 200 OK \r\n".encode())
63         connectionSocket.send("Content-Type: image/png \r\n".encode())
64         connectionSocket.send("\r\n".encode())
65         connectionSocket.send(content)

```

Figure 15: part 3->the mainServer code (4)

- the fourth if statement → if the request end with **.css** then the server will send response the **Css_File_Style.css** this mean any request Ended with .css it will send the same response of file and this appear in our result if we request an **c.css** the opened file will be the **Css_File_Styel.css** in This Methode it will be the request By The Client (web browser) be more General .
- The Fifth if statement → if the request end with **.png** then the server will send response the **image will take from the url website** this mean any request Ended with .png it will send the same response of image and this appear in our result if we request an **p.png** the opened image will be the **image from the url** in This Methode it will be the request By The Client (web browser) be more General .
- **Explaine the Code of The Fifth if statement :**

This code fetches an image from a specified URL and sends it as a response to a client's request. It sets up a request to mimic a web browser, retrieves the image content, and sends it back to the client over a network connection i.

- **Notice** :In Our Code we need to use the set up the request to mimic web browser When I do it Like jpg , it Appear Error we Don't why This Error Happen .

```

67
68     elif object.endswith('.jpg'):
69         url = "https://www.w3schools.com/w3css/img_forest.jpg"
70         connectionSocket.send("HTTP/1.1 200 OK\r\n".encode())
71         connectionSocket.send("Content-Type: image/jpg \r\n".encode())
72         connectionSocket.send("\r\n".encode())
73         urllib.request.urlretrieve(url, filename="image.jpg")
74         # Open the local image file and send its content
75         file6 = open("image.jpg", "rb")
76         connectionSocket.send(file6.read())
77
78     elif object == '/so':
79         connectionSocket.send("HTTP/1.1 307 Temporary Redirect \r\n".encode())
80         connectionSocket.send("Content-Type: text/html \r\n".encode())
81         connectionSocket.send("Location: https://stackoverflow.com \r\n".encode())
82         connectionSocket.send("\r\n".encode())
83
84     elif object == '/itc':
85         connectionSocket.send("HTTP/1.1 307 Temporary Redirect \r\n".encode())
86         connectionSocket.send("Content-Type: text/html \r\n".encode())
87         connectionSocket.send("Location: https://ritaj.birzeit.edu// \r\n".encode())
88         connectionSocket.send("\r\n".encode())

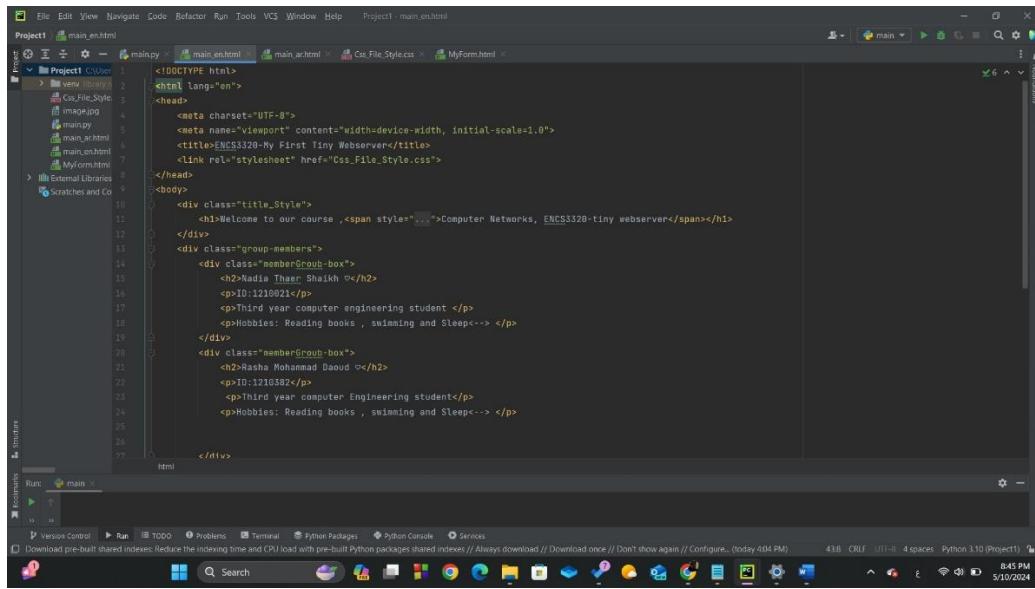
```

Figure 16:part (3) -->mainSer Code (4)

- In the Sixth If Statement → if the request end with **.jpg** then the server will send response the **image will take from the url website** this mean any request Ended with .jpg it will send the same response of image and this appear in our result if we request an **j.jpg** the opened image will be Download locally **from the url** and Save it in File Called →**image** in This Methode it will be the request By The Client (web browser) be more General .
- In the Seventh If Statement →it will Display an the Stackovelflow .com
- In Eight if Statement →it will Display The Ritage

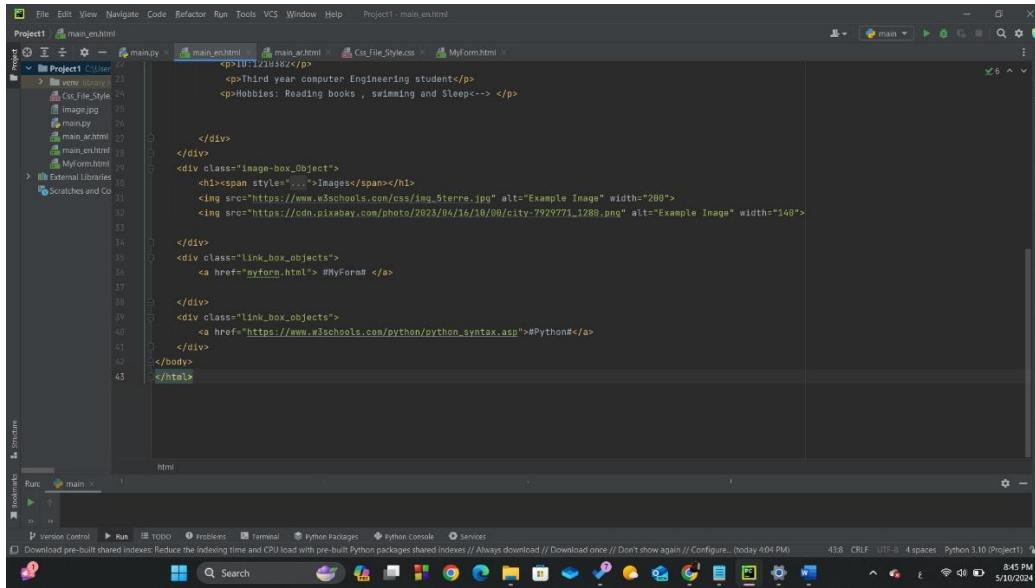
3.1.2 HTML and css Code

Maine_en HTML Code :



```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>ENCS3320-My First Tiny Webserver</title>
        <link rel="stylesheet" href="Css_File_Style.css">
    </head>
    <body>
        <div class="title_Style">
            <h1>Welcome to our course ,<span style="...>Computer Networks, ENCS3320-tiny webserver</span></h1>
        </div>
        <div class="group-members">
            <div class="member_group_box">
                <h2>Nadia Thareen Shaikh </h2>
                <p>ID:121021c</p>
                <p>Third year computer engineering student </p>
                <p>Hobbies: Reading books , swimming and Sleep<--> </p>
            </div>
            <div class="member_group_box">
                <h2>Rasha Mohammad Daoud </h2>
                <p>ID:1210302</p>
                <p>Third year computer Engineering student</p>
                <p>Hobbies: Reading books , swimming and Sleep<--> </p>
            </div>
        </div>
    </body>
</html>
```

Figure 17:Main_en Code →> Part 1 .



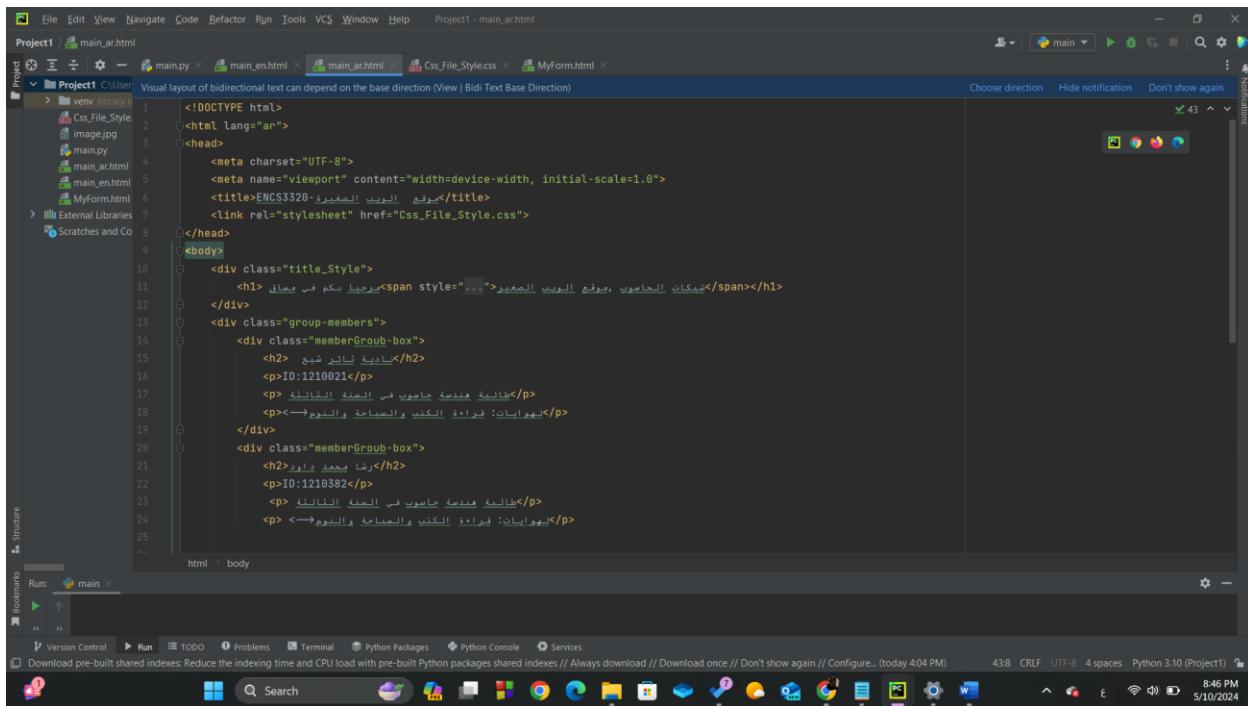
```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>ENCS3320-My First Tiny Webserver</title>
        <link rel="stylesheet" href="Css_File_Style.css">
    </head>
    <body>
        <div class="group-members">
            <div class="member_group_box">
                <h2>Nadia Thareen Shaikh </h2>
                <p>ID:121021c</p>
                <p>Third year computer Engineering student </p>
                <p>Hobbies: Reading books , swimming and Sleep<--> </p>
            </div>
            <div class="member_group_box">
                <h2>Rasha Mohammad Daoud </h2>
                <p>ID:1210302</p>
                <p>Third year computer Engineering student</p>
                <p>Hobbies: Reading books , swimming and Sleep<--> </p>
            </div>
        </div>
        <div class="image_box_Object">
            <div style="text-align: center; margin-bottom: 10px;">
                
            </div>
            <div style="text-align: center; margin-bottom: 10px;">
                
            </div>
        </div>
        <div class="link_box_objects">
            <a href="#">#Myform</a>
            <a href="https://www.w3schools.com/python/python_syntax.asp">#Python#</a>
        </div>
    </body>
</html>
```

Figure 18:Maine_en Cod -->Part 2

code is a basic HTML webpage that displays information about a course and its members. It includes elements like headings, paragraphs, and images. The title of the webpage is "ENCS3320-My First Tiny Webserver". It lists two group members with their names, IDs, and hobbies. There are also images displayed, along with links to other pages like "myform.html" and a Python syntax tutorial on W3Schools. The webpage uses simple styling with CSS to make it visually appealing. Overall, it's a

simple webpage showcasing information about a course and its members, with links to additional resources.

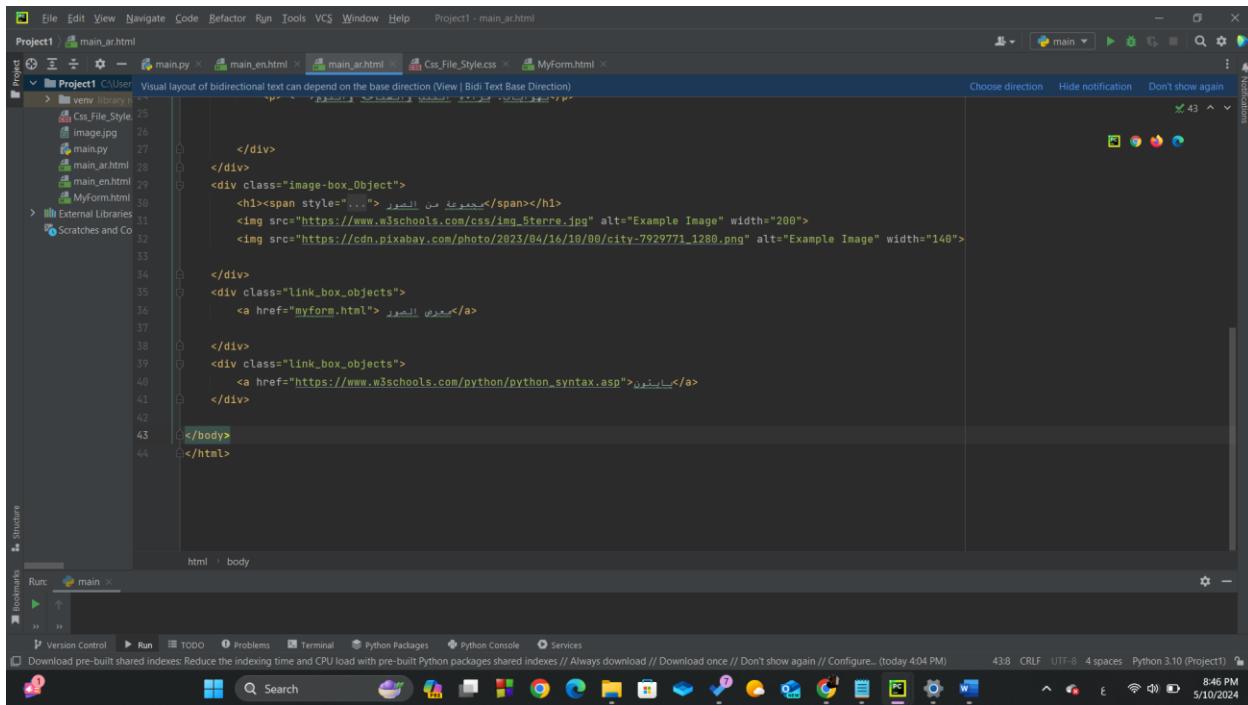
Maine_ar:



The screenshot shows the PyCharm IDE interface with the file `main_ar.html` open. The code is written in HTML and includes Arabic text. It features two sections of member profiles, each with a name, ID, and a brief description. The code uses CSS classes like `title_Style`, `group-members`, and `memberGroup_box`. The PyCharm interface includes a sidebar with project files like `main.py` and `Css_File_Style.css`, and a bottom toolbar with various icons.

```
<!DOCTYPE html>
<html lang="ar">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>ENCS3320 - موقع الموارد المفتوحة</title>
    <link rel="stylesheet" href="Css_File_Style.css">
</head>
<body>
    <div class="title_Style">
        <h1>مكتبات الحاسوب، موقع الموارد المفتوحة</h1>
    </div>
    <div class="group-members">
        <div class="memberGroup_box">
            <h2>بيانات قيادي تأثير قيادي</h2>
            <p>ID:12108021</p>
            <p>كلية هندسة حاسوب في السنة الرابعة</p>
            <p><img alt="arrow icon" style="vertical-align: middle;">معلومات: قراءة الكتب والرسائل والرسائل</p>
        </div>
        <div class="memberGroup_box">
            <h2>بيانات قيادي قيادي</h2>
            <p>ID:12108382</p>
            <p>كلية هندسة حاسوب في السنة الرابعة</p>
            <p><img alt="arrow icon" style="vertical-align: middle;">معلومات: قراءة الكتب والرسائل والرسائل</p>
        </div>
    </div>
</body>
```

Figure 19:Main_ar.HTML Code 1



This screenshot shows the continuation of the `main_ar.html` code. It includes several `img` tags with URLs pointing to external images. The code structure remains consistent with the previous screenshot, featuring the same header, body sections, and member profiles. The PyCharm interface is identical, with the same sidebar and bottom toolbar.

```
        <div class="image_box_Object">
            <h1><span style="color: #0000ff; font-size: 2em; margin-right: 10px; position: relative; top: -5px; left: -5px; border-radius: 50%; padding: 2px; background-color: white; border: 1px solid #0000ff; transition: all 0.3s ease; cursor: pointer; user-select: none; ">+</span>بيانات من المقرر</h1>
            
            
        </div>
        <div class="link_box_objects">
            <a href="myform.html">بيانات المقرر</a>
        </div>
        <div class="link_box_objects">
            <a href="https://www.w3schools.com/python/python_syntax.asp">بيانات</a>
        </div>
</body>
</html>
```

Figure 20:Main_ar.html code 2

Css Code :

```
Project1 > Css_File_Style.css
Project1 > Project1 C:\User\venv\library\main.py x main_en.html x main_ar.html x Css_File_Style.css x MyForm.html x
*.css files are supported in other JetBrains IDEs
1  /*
2   Nadia 1210021
3   Rasha 1210382
4   */
5  /* First Style The Body of The page */
6  body {
7      background-repeat: no-repeat;
8      background-color:#B9D9EB;
9  }
10 /* Style The Title*/
11 .title_Style {
12     text-align: center;
13     margin-bottom: 20px;
14 }
15 /* Style The Shape of Group Member Box */
16 .group-members {
17     display: flex;
18     /* Change flex-direction to column */
19     flex-direction: column;
20     /* Make All Object on The Left of The Web page */
21     align-items: left;
22     margin-bottom: 20px;
23 }
24 .memberGroup-box {
25     width: 40%; /* Set width to 100% to occupy the entire width */
26     padding: 20px;
27 }
```

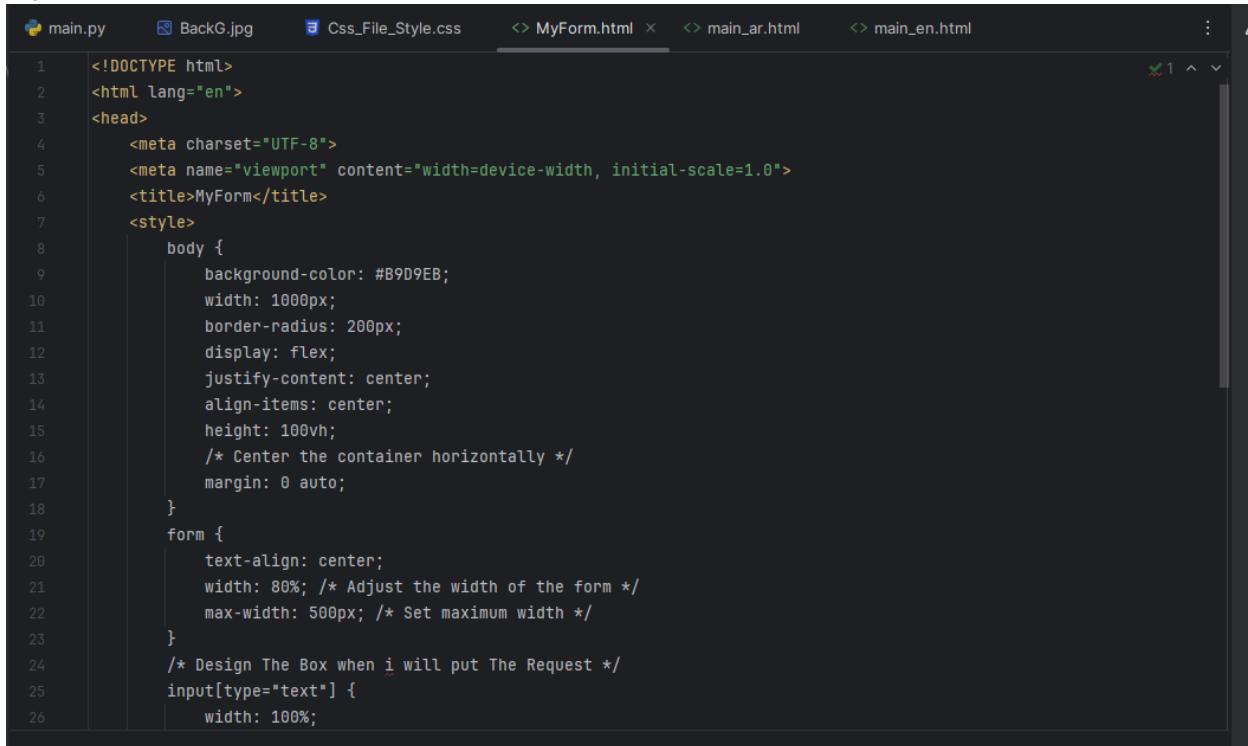
Figure 21:css Code 1

The screenshot shows the PyCharm IDE interface with the following details:

- File Bar:** File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help, Project1 - Css_File.css
- Project Bar:** Project1, Css_File.css, main.py, main_en.html, main_ar.html, Css_File_Style.css, MyForm.html
- Left Sidebar:** Project (expanded to show venv library, Css_File.Style, image.jpg, main.py, main_ar.html, main_en.html, MyForm.html), External Libraries, Scratches and Co.
- Central Area:** CSS code editor for `Css_File_Style.css`. The code defines styles for various HTML elements like `.memberGroup-box`, `.image-box_Object`, `.link_box_objects`, and `.link-box a`. A yellow circle highlights the background color declaration in the `.link_box_objects` rule.
- Right Sidebar:** Notifications for Try WebStorm and Try PyCharm Professional, Dismiss button, and a checkmark icon with "2".
- Bottom Navigation:** Run (main), Version Control, Run, TODO, Problems, Terminal, Python Packages, Python Console, Services.
- Bottom Status:** Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download once // Don't show again // Configure... (today 4:04 PM), 421 CRLF, UTF-8, 4 spaces, Python 3.10 (Project1), 847 PN, 5/10/2024.

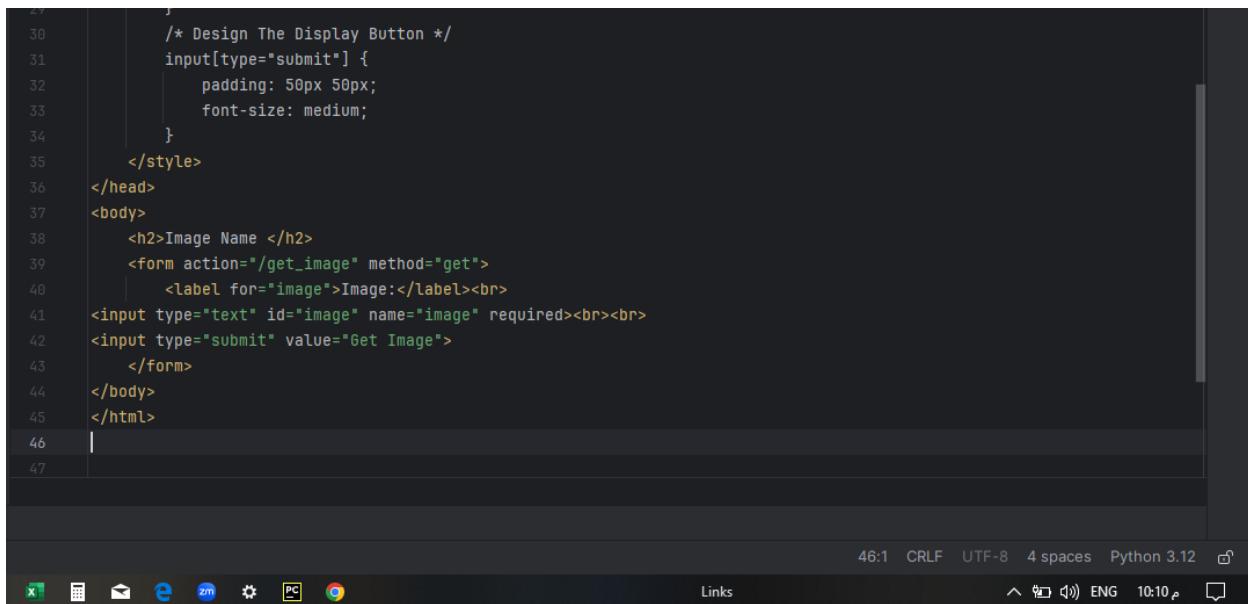
Figure 22css code 2:

My Form Code



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>MyForm</title>
7     <style>
8         body {
9             background-color: #B9D9EB;
10            width: 1000px;
11            border-radius: 200px;
12            display: flex;
13            justify-content: center;
14            align-items: center;
15            height: 100vh;
16            /* Center the container horizontally */
17            margin: 0 auto;
18        }
19        form {
20            text-align: center;
21            width: 80%; /* Adjust the width of the form */
22            max-width: 500px; /* Set maximum width */
23        }
24        /* Design The Box when i will put The Request */
25        input[type="text"] {
26            width: 100%;
```

Figure 23: My Form -->1



```
27     }
28     /* Design The Display Button */
29     input[type="submit"] {
30         padding: 50px 50px;
31         font-size: medium;
32     }
33 </style>
34 </head>
35 <body>
36     <h2>Image Name </h2>
37     <form action="/get_image" method="get">
38         <label for="image">Image:</label><br>
39         <input type="text" id="image" name="image" required><br><br>
40         <input type="submit" value="Get Image">
41     </form>
42 </body>
43 </html>
44 |
45
46
47
```

46:1 CRLF UTF-8 4 spaces Python 3.12

Links ENG 10:10 PM

Figure 24: MyFor code-->2

3.2. if the request is / or /index.html or /main_en.html or /en (for example localhost:6060/ or localhost:6060/en)

3.2.1 If I request /index.html

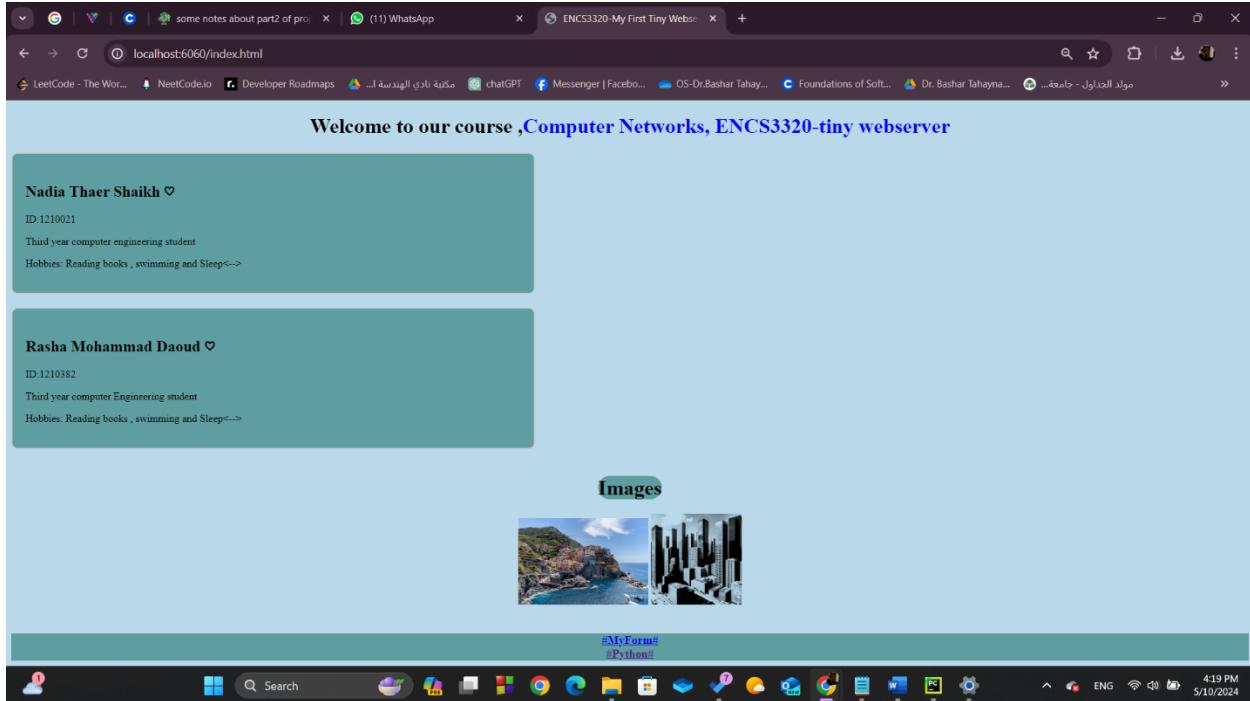


Figure 25:request /index.html Result.

The request message :

```
Run: main
C:\Users\TwitterStore\PycharmProjects\Project1\venv\Scripts\python.exe C:/Users/TwitterStore/PycharmProjects/Project1/main.py
The Web Server is Ready ....
Received sentence From The Client (web browser): GET /index.html HTTP/1.1
Host: localhost:6060
Connection: keep-alive
sec-ch-ua: "Chromium";v="124", "Google Chrome";v="124", "Not-A.Brand";v="99"
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/124.0.0.0 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, zstd
Accept-Language: ar,en-US;q=0.9,en;q=0.8
```

A screenshot of the PyCharm IDE showing the terminal window. The terminal output shows a successful HTTP request from a web browser to the local host at port 6060 for the '/index.html' endpoint. The response includes standard headers like 'Content-Type', 'Content-Length', and 'Connection'. The PyCharm interface includes a top navigation bar with tabs for Run, Version Control, Problems, Terminal, Python Packages, Python Console, and Services. The bottom of the screen shows the Windows taskbar with various pinned application icons.

Figure 26:request /index.html Http request.

3.2.2 request / main_en.html

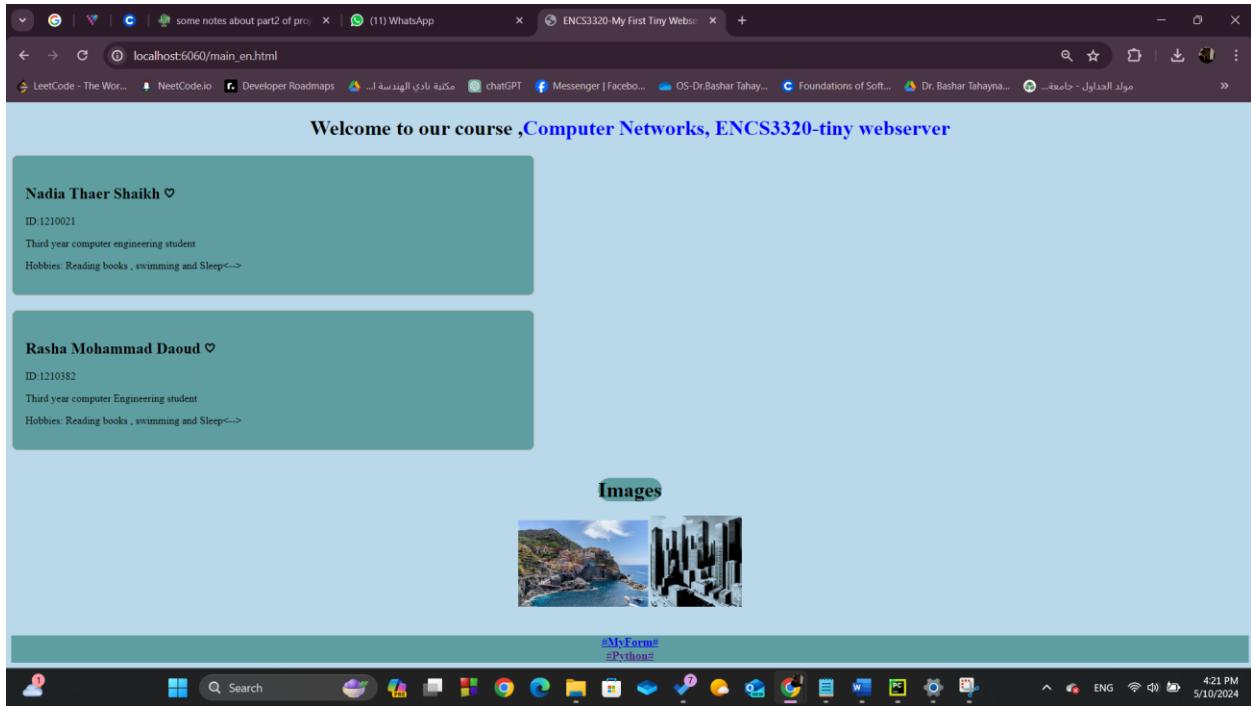


Figure 27:request / main_en.html Result.

The requasred masse HTTP:

```
Run: C:\Users\TwitterStore\PycharmProjects\Project1\venv\Scripts\python.exe C:/Users/TwitterStore/PycharmProjects/Project1/main.py
G ↑ C:\Users\TwitterStore\PycharmProjects\Project1\main.py
↓ The Web Server is Ready ....
Received sentence From The Client (web browser): GET /main_en.html HTTP/1.1
Host: localhost:6060
Connection: keep-alive
Cache-Control: max-age=0
sec-ch-ua: "Chromium";v="124", "Google Chrome";v="124", "Not-A.Brand";v="99"
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/124.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,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, zstd
Accept-Language: ar,en-US;q=0.9,en;q=0.8
```

Figure 28:request / main_en.html Http request .

3.2.3 request /en

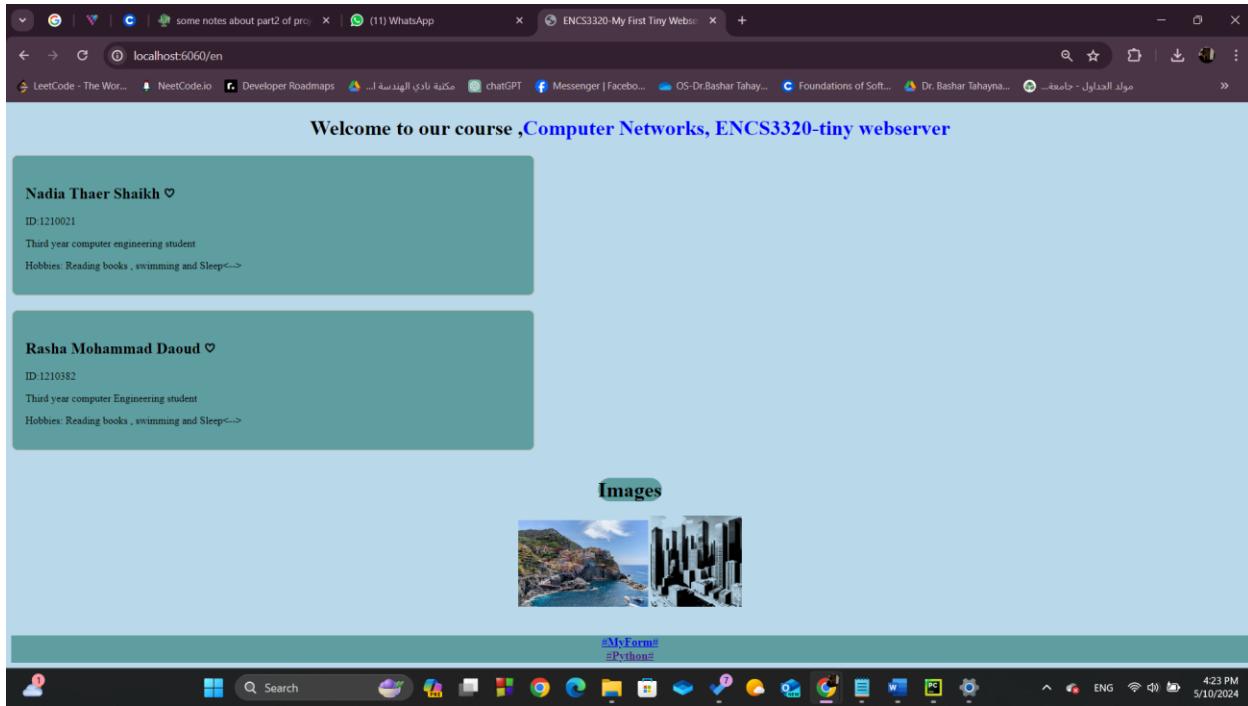


Figure 29:request /en Result .

The HTTP request :

```
Run: main x
G ↑ C:\Users\TwitterStore\PycharmProjects\Project1\venv\Scripts\python.exe C:/Users/TwitterStore/PycharmProjects/Project1/main.py
The Web Server is Ready ....
Received sentence From The Client (web browser): GET /en HTTP/1.1
Host: localhost:6666
Connection: keep-alive
Cache-Control: max-age=0
sec-ch-ua: "Chromium";v="124", "Google Chrome";v="124", "Not-A.Brand";v="99"
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/124.0.0.0 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, zstd
Accept-Language: ar,en-US;q=0.9,en;q=0.8
```

Figure 30:request /en Http Request Massge .

3.2.4 . request /

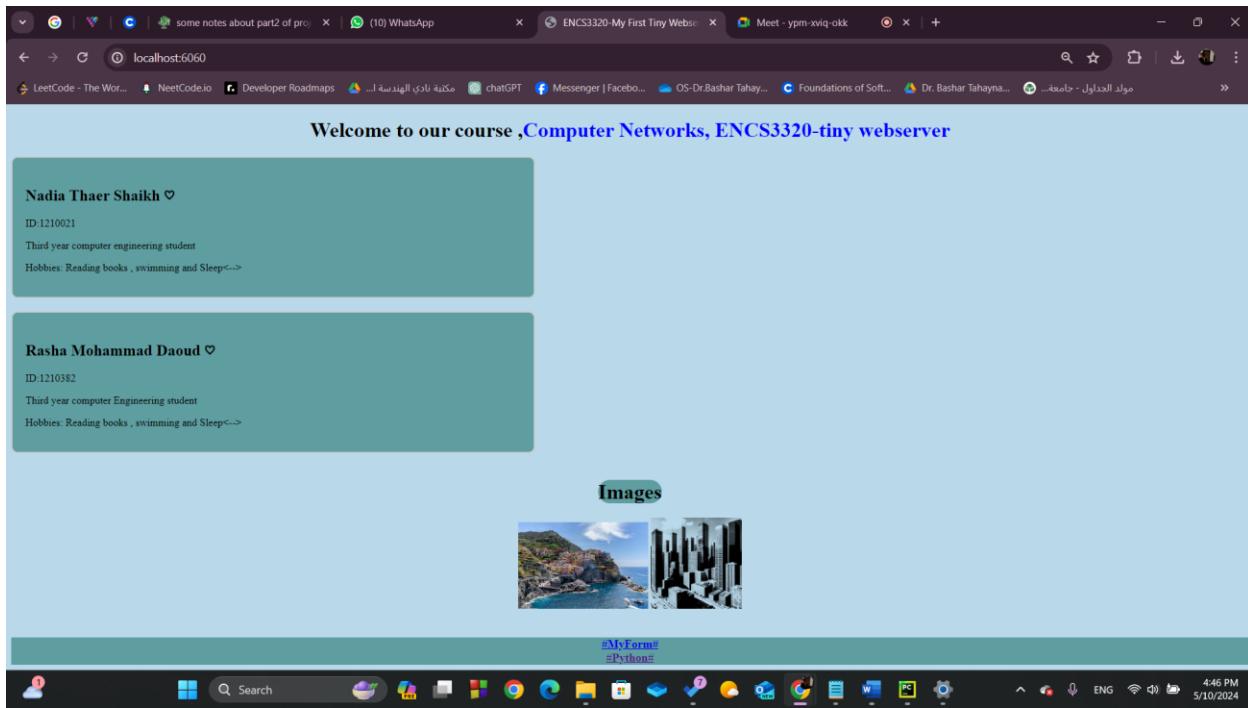


Figure 31:request / Result

The HTTP request :

```
Run: main
C:\Users\TwitterStore\PycharmProjects\Project1\venv\Scripts\python.exe C:/Users/TwitterStore/PycharmProjects/Project1/main.py
The Web Server is Ready ....
Received sentence From The Client (web browser): GET / HTTP/1.1
Host: localhost:6060
Connection: keep-alive
Cache-Control: max-age=0
sec-ch-ua: "Chromium";v="124", "Google Chrome";v="124", "Not-A.Brand";v="99"
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/124.0.0.0 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=0.8
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br, zstd
Accept-Language: ar,en-US;q=0.9,en;q=0.8
```

Figure 32:request / Http request massge .

3.3. Requested /ar and main_ar

3.3.1 requeste the /ar

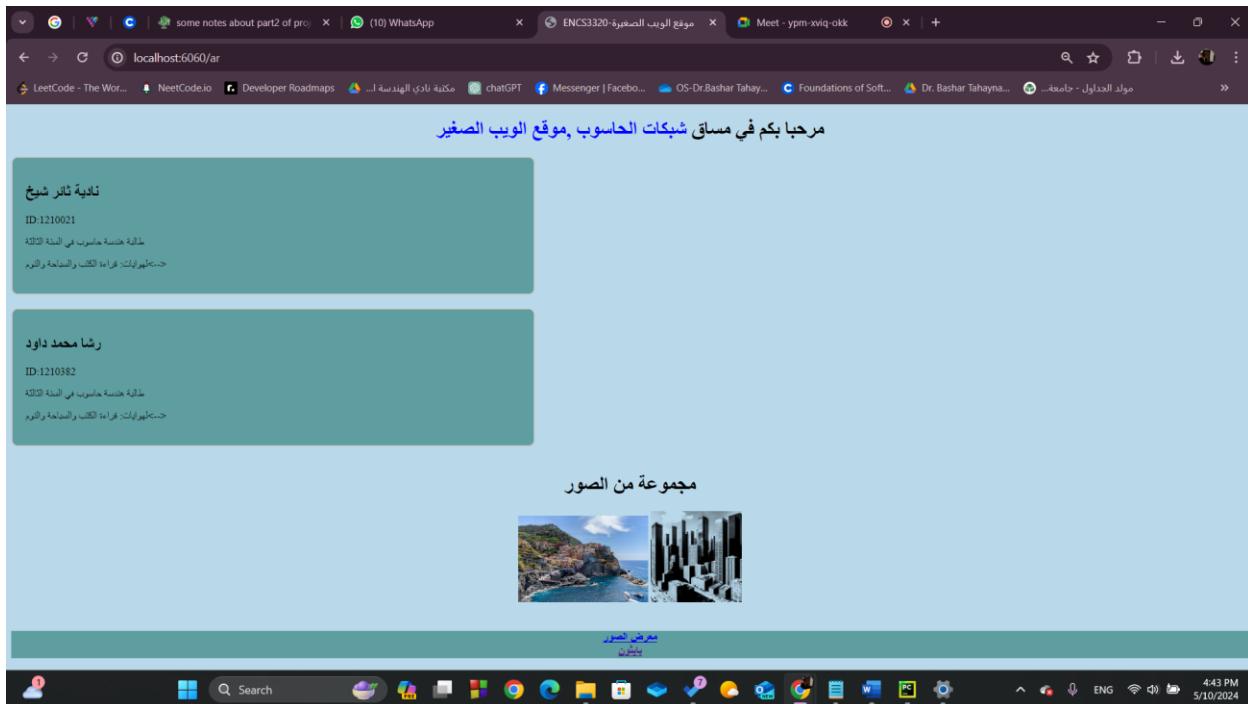


Figure 33:requeste the /ar Result .

The HTTP request:

A screenshot of a terminal window titled "Run: main". The window displays a raw HTTP request. The "Request object" is "/ar". The "Received sentence From The Client (web browser)" is "GET /Css_File_Style.css HTTP/1.1". The "Host" is "localhost:6060". The "Connection" is "keep-alive". The "User-Agent" is "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/124.0.0.0 Safari/537.36". The "Accept" header is "text/css,*/*;q=0.1". The "Sec-Fetch-Site" is "same-origin". The "Sec-Fetch-Mode" is "no-cors". The "Sec-Fetch-Dest" is "style". The "Referer" is "http://localhost:6060/ar". The "Accept-Encoding" is "gzip, deflate, br, zstd". The "Accept-Language" is "ar,en-US;q=0.9,en;q=0.8". The terminal also shows other tabs like "Version Control", "Run", "TODO", "Problems", "Terminal", "Python Packages", "Python Console", and "Services". The bottom of the screen shows the Windows taskbar.

Figure 34:The Http request For -->requeste the /ar

3.3.2 request the /main_ar



Figure 35: request the /main_ar Result.

The HTTP request

```
Run: main
C:\Users\TwitterStore\PycharmProjects\Project1\venv\Scripts\python.exe C:/Users/TwitterStore/PycharmProjects/Project1/main.py
The Web Server is Ready ....
Received sentence From The Client (web browser): GET /main_ar.html HTTP/1.1
Host: localhost:6060
Connection: keep-alive
Cache-Control: max-age=0
sec-ch-ua: "Chromium";v="124", "Google Chrome";v="124", "Not-A.Brand";v="99"
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/124.0.0.0 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, zstd
Accept-Language: ar,en-US;q=0.9,en;q=0.8
```

Figure 36: request the /main_ar Http request massge .

3.4 if the request Was rasha.html

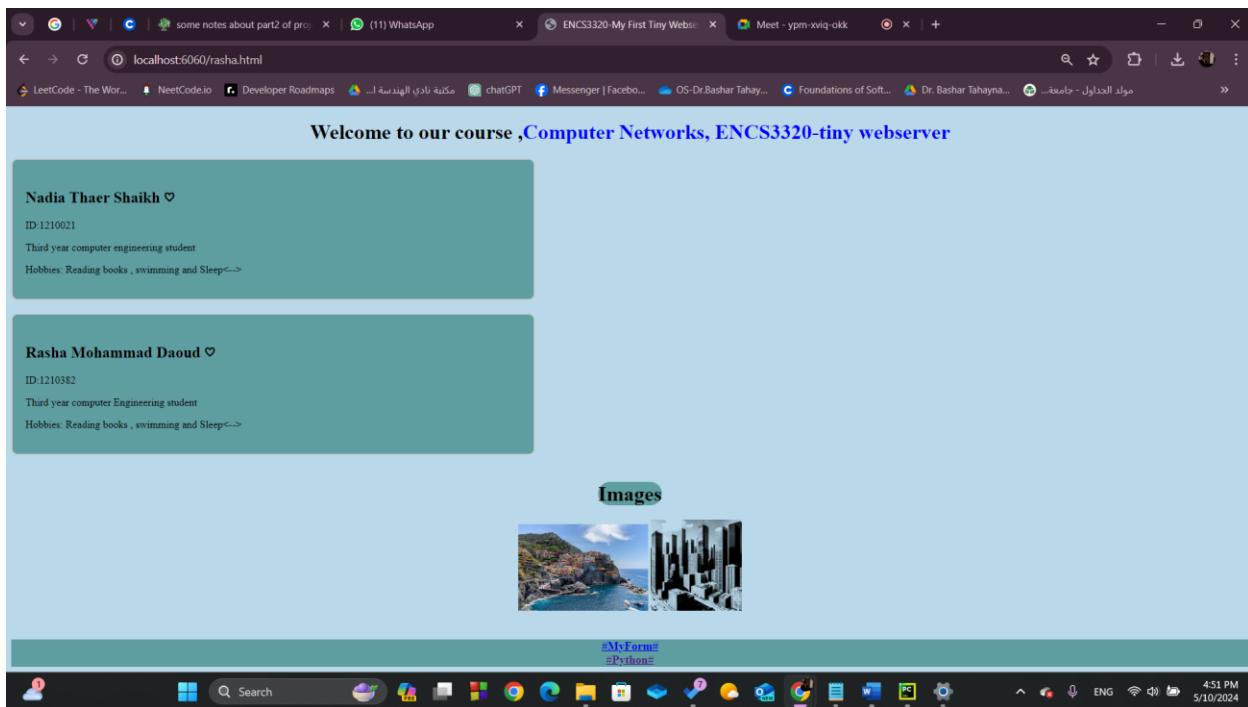


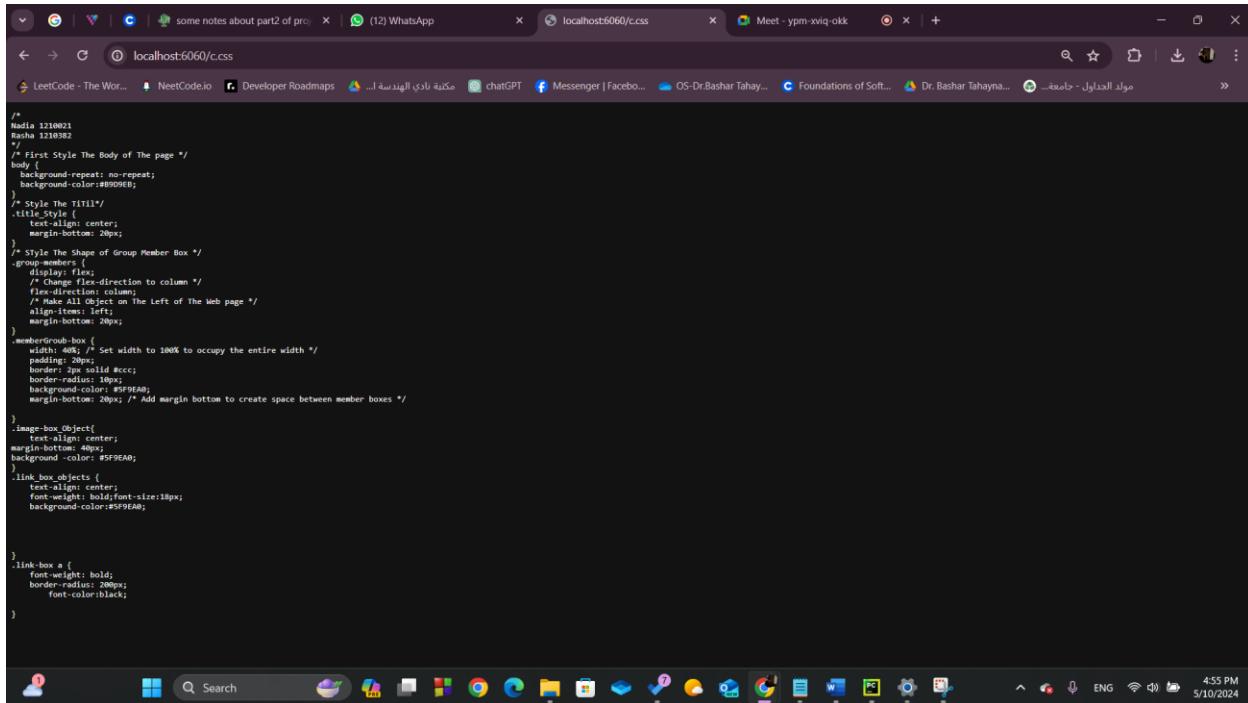
Figure 37:the request Was rasha.html Result .

The Http Request :

A screenshot of a terminal window titled 'main'. The window displays the raw HTTP request message. The 'Requested object' is '/rasha.html'. The 'Received sentence From The Client (web browser)' is 'GET /Css_File_Style.css HTTP/1.1'. The 'Host' header is 'localhost:6060'. The 'Connection' header is 'keep-alive'. The 'sec-ch-ua' header is 'Chromium";v="124", "Google Chrome";v="124", "Not-A.Brand";v="99"'. The 'sec-ch-ua-mobile' header is '?0'. The 'User-Agent' header is 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/124.0.0.0 Safari/537.36'. The 'sec-ch-ua-platform' header is 'Windows'. The 'Accept' header is 'text/css,*/*;q=0.1'. The 'Sec-Fetch-Site' header is 'same-origin'. The 'Sec-Fetch-Mode' header is 'no-cors'. The 'Sec-Fetch-Dest' header is 'style'. The 'Referer' header is 'http://localhost:6060/rasha.html'. The 'Accept-Encoding' header is 'gzip, deflate, br, zstd'. The 'Accept-Language' header is 'ar,en-US;q=0.9,en;q=0.8'. The terminal window has a dark theme and includes standard Windows taskbar icons at the bottom.

Figure 38:the request Was rasha.html Request Massge.

3.5 .if request c.css



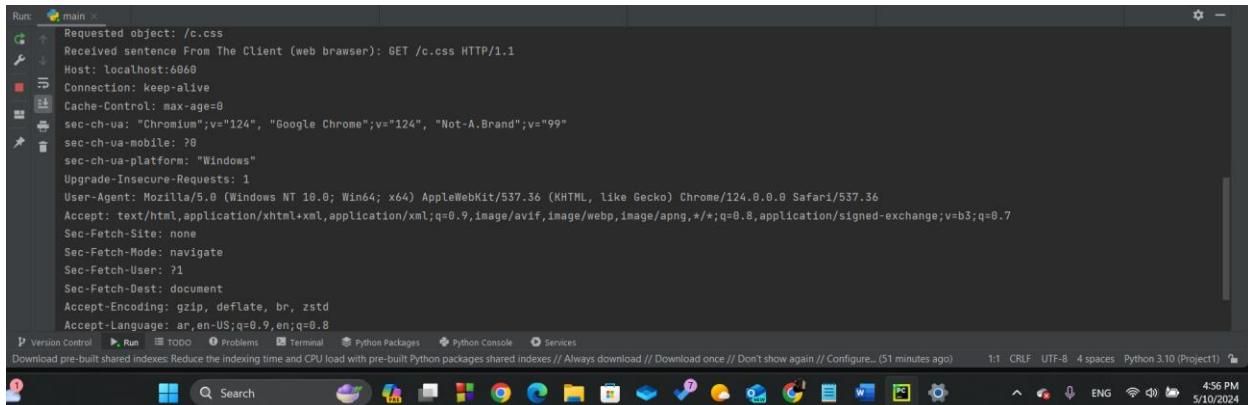
The screenshot shows a Microsoft Edge browser window with multiple tabs open. The active tab displays the CSS code for 'c.css'. The code is a stylized template for a group member box, featuring a white background with a black border and rounded corners. It includes sections for 'body', 'group-members', 'membersGroupBox', 'image-box_Object', 'link_box_objects', and 'link-box_a'. The code uses various CSS properties like 'background-color', 'border-radius', and 'font-weight'.

```
/*
Nadia 12108021
Rasha 12109382
*/
/* First Style The Body of The page */
body {
    background-repeat: no-repeat;
    background-color:#B0B0B0;
}
/* Style The Title */
.title {
    text-align: center;
    margin-bottom: 20px;
}
/* Style The Shape Of Group Member Box */
.group-members {
    display: flex;
    /* Change flex-direction to column */
    flex-direction: column;
    /* Make All Object on The Left of The Web page */
    align-items: flex-start;
    margin-bottom: 20px;
}
.membersGroupBox {
    width: 40%; /* Set width to 100% to occupy the entire width */
    padding: 20px;
    border: 2px solid #ccc;
    border-radius: 10px;
    background-color: #5F9EA0;
    margin-bottom: 20px; /* Add margin bottom to create space between member boxes */
}
.image-box_Object{
    text-align: center;
    margin-bottom: 40px;
    background-color: #5F9EA0;
}
.link_box_objects {
    text-align: center;
    font-weight: bold;font-size:18px;
    background-color:#5F9EA0;
}

.link-box_a {
    font-weight: bold;
    border-radius: 20px;
    font-color:black;
}
}
```

Figure 39:request c.css Result .

The Http request



The screenshot shows a terminal window from the PyCharm IDE. The 'Run' tab is selected, and the output pane shows the following log entry:

```
Run: Requested object: /c.css
Received Sentence From The Client (web browser): GET /c.css HTTP/1.1
Host: localhost:6060
Connection: keep-alive
Cache-Control: max-age=0
sec-ch-ua: "Chromium";v="124", "Google Chrome";v="124", "Not-A.Brand";v="99"
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/124.0.0.0 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, zstd
Accept-Language: ar,en-US;q=0.9,en;q=0.8
```

Figure 40:request c.css Http Massge Requested .

3.6 if I request p.png

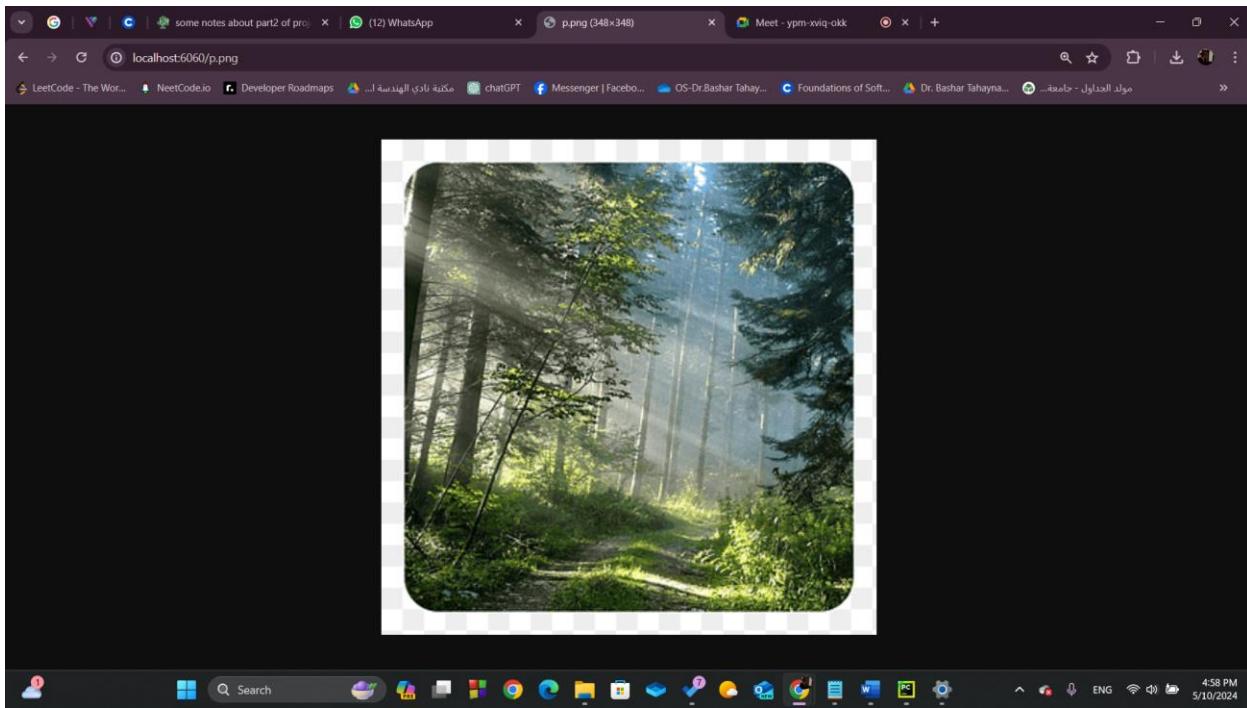


Figure 41:request p.png Result

The Http request :

```
Run: main <-->
Requested object: /p.png
Received sentence From The Client (web browser): GET /p.png HTTP/1.1
Host: localhost:6060
Connection: keep-alive
Cache-Control: max-age=0
sec-ch-ua: "Chromium";v="124", "Google Chrome";v="124", "Not-A.Brand";v="99"
sec-ch-ua-mobile: ?
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/124.0.0.0 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, zstd
Accept-Language: ar,en-US;q=0.9,en;q=0.8
```

Figure 42:request p.png Massge Request .

3.7 if I request j.jpg

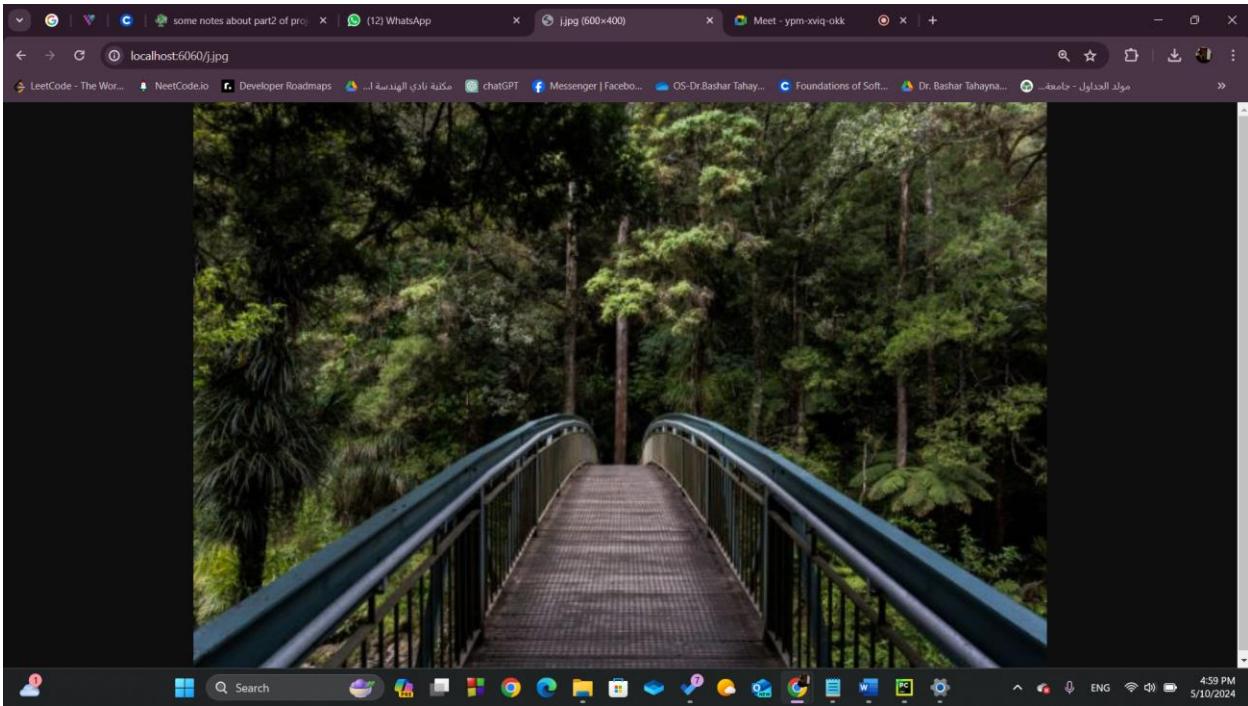
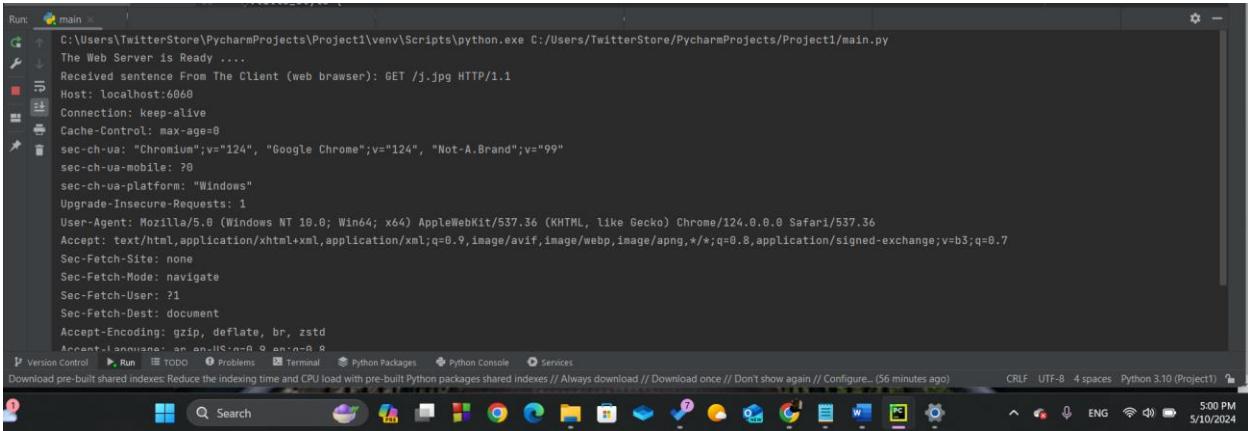


Figure 43:request j.jpg Result.

The Http request :



```
Run: main.py
C:\Users\TwitterStore\PycharmProjects\Project1\venv\Scripts\python.exe C:/Users/TwitterStore/PycharmProjects/Project1/main.py
The Web Server is Ready ....
Received sentence From The Client (web browser): GET /j.jpg HTTP/1.1
Host: localhost:6060
Connection: keep-alive
Cache-Control: max-age=0
sec-ch-ua: "Chromium";v="124", "Google Chrome";v="124", "Not-A.Brand";v="99"
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/124.0.0.0 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, zstd
Accept-Language: en-US,en;q=0.8

```

Version Control Run TODO Problems Terminal Python Packages Python Console Services

Download pre-built shared indexes. Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download once // Don't show again // Configure... (56 minutes ago)

CRLF UTF-8 4 spaces Python 3.10 (Project1) 5:00 PM 5/10/2024

Figure 44:request j.jpg HTTP requested Massge.

3.8 MyForm

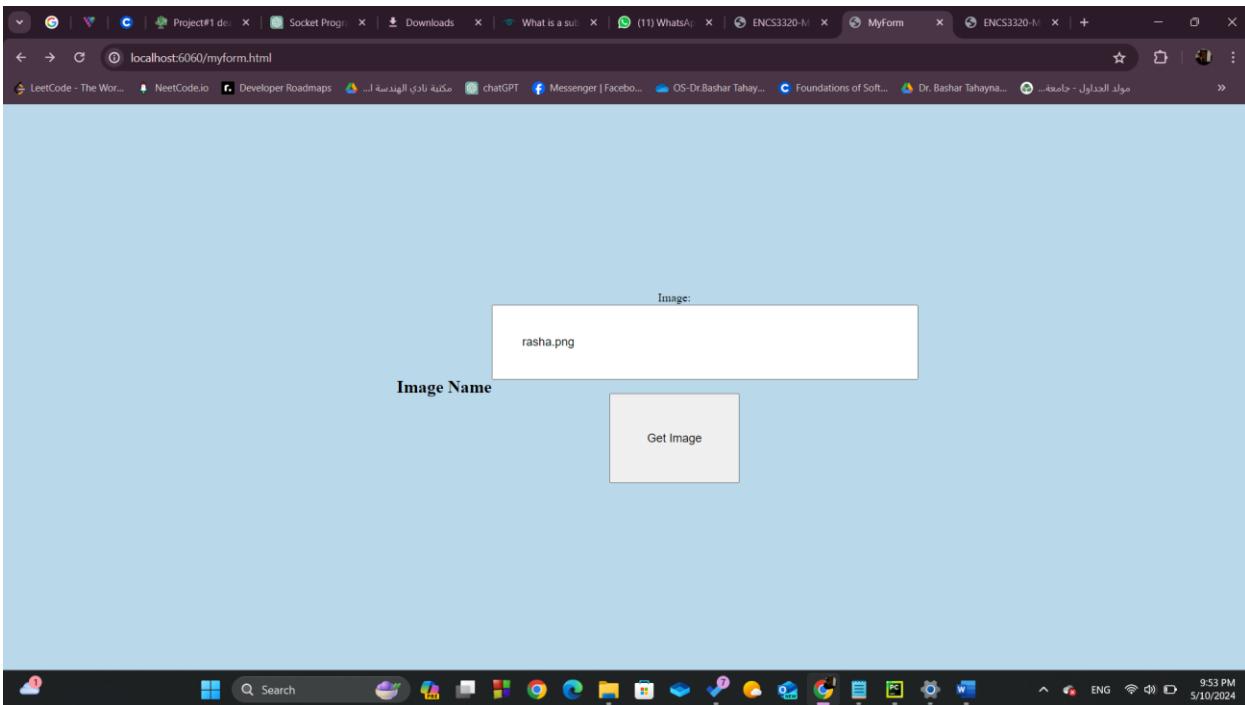


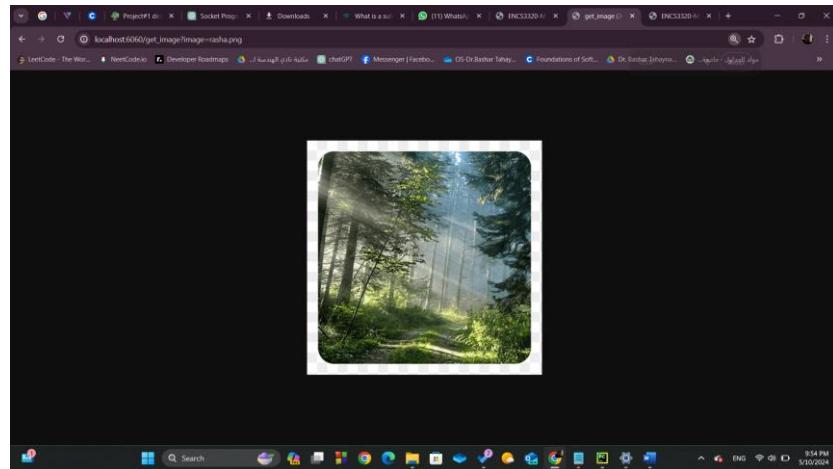
Figure 45: My Form Result

HTTP request:

A screenshot of a terminal window titled 'main'. The log shows an incoming GET request for '/myform.html' from 'localhost:6060'. The request includes standard headers like Host, Connection, Cache-Control, and User-Agent, which identifies the browser as Google Chrome version 124.0.0.0. The terminal also shows the Python environment and system status at the bottom.

Figure 46: MyForm Result HTTP request.

Result of photo result:



3.9 if I request /so

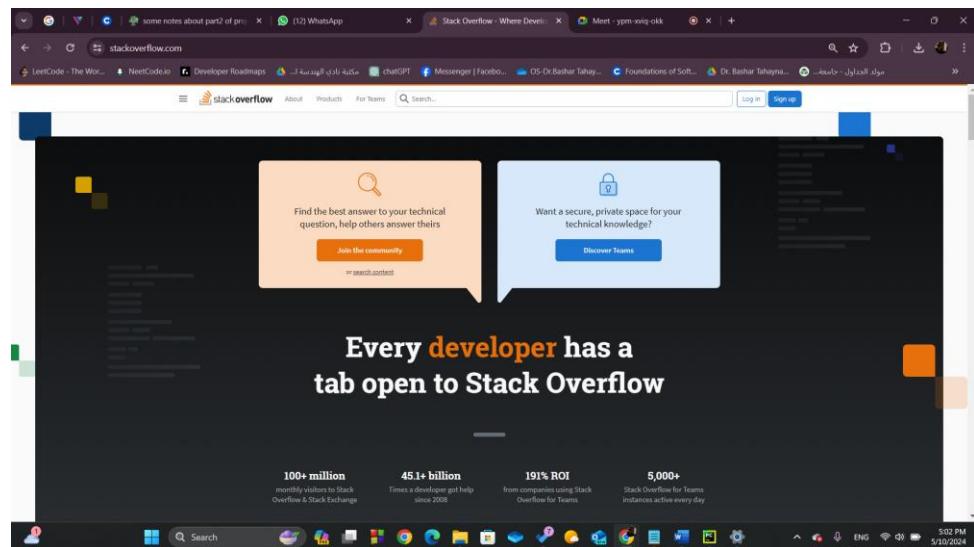


Figure 47:request /so Result.

The Http request :

```
Run: main.py
C:\Users\TwitterStore\PycharmProjects\Project1\venv\Scripts\python.exe C:/Users/TwitterStore/PycharmProjects/Project1/main.py
The Web Server is Ready ....
Received sentence From The Client (web browser): GET /so HTTP/1.1
Host: localhost:6060
Connection: keep-alive
sec-ch-ua: "Chromium";v="124", "Google Chrome";v="124", "Not-A-Brand";v="99"
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/124.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/png,*/*;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, zstd
Accept-Language: ar,en-US;q=0.9,en;q=0.8
```

Figure 48:request /so HTTP requested Massge .

3.10.if I request /itc

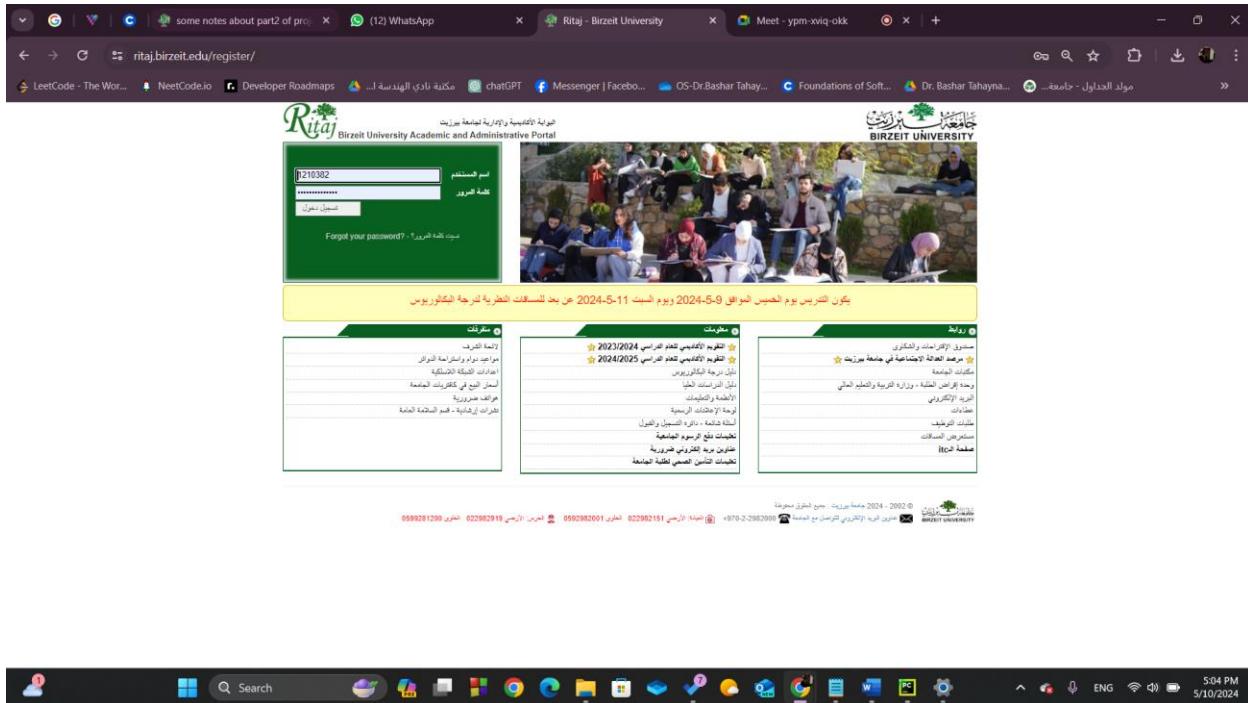


Figure 49:request /itc Result .

The Http request:

```
Run: main
C:\Users\TwitterStore\PycharmProjects\Project1\venv\Scripts\python.exe C:/Users/TwitterStore/PycharmProjects/Project1/main.py
The Web Server is Ready ....
Received sentence From The Client (web browser): GET /itc HTTP/1.1
Host: localhost:6060
Connection: keep-alive
sec-ch-ua: "Chromium";v="124", "Google Chrome";v="124", "Not-A.Brand";v="99"
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/124.0.0.0 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, zstd
Accept-Language: ar,en-US;q=0.9,en;q=0.8
Download pre-built shared indexes. Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download once // Don't show again// Configure... (1 hour ago)
```

Figure 50:request /itc Requested message Http .

3.11. Request any File Does Not Contain in The Server

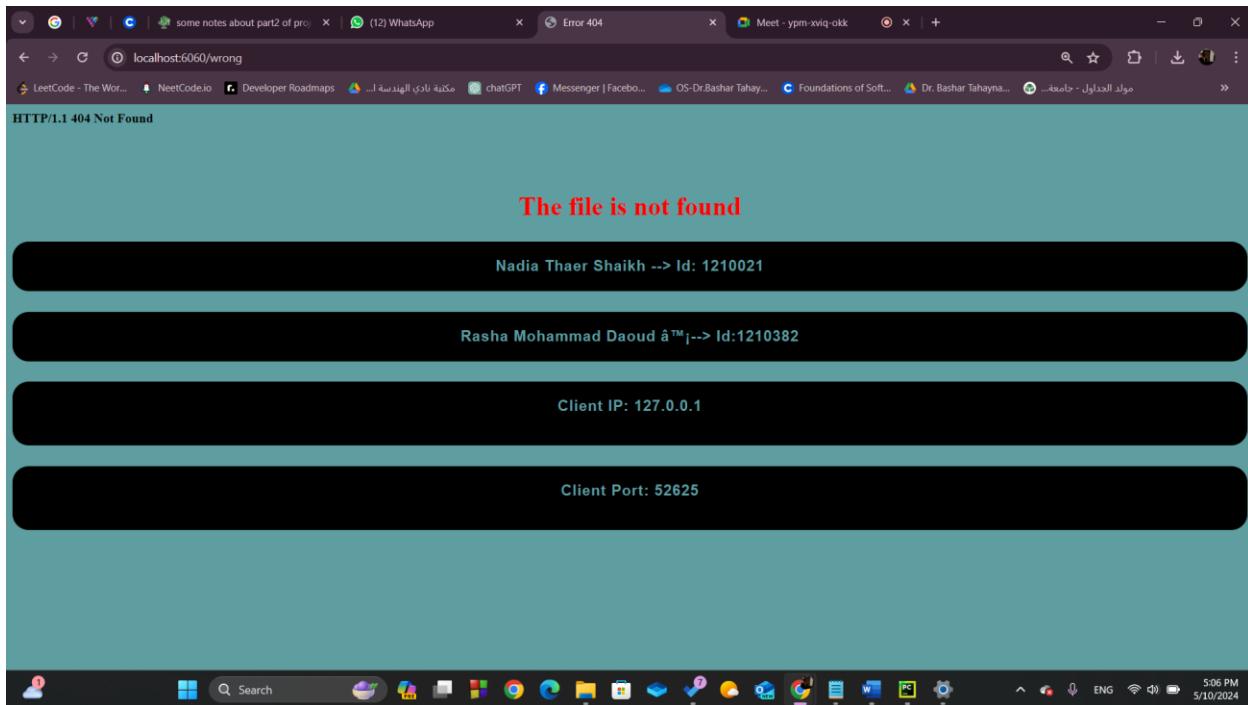


Figure 51:Request any File Does Not Contain in The Server Result.

The HTTP request:

A screenshot of the PyCharm IDE. The "Run" tab is active, showing the command "C:\Users\TwitterStore\PycharmProjects\Project1\venv\Scripts\python.exe C:/Users/TwitterStore/PycharmProjects/Project1/main.py". The terminal pane displays the following log output:

```
Run: main x
C:\Users\TwitterStore\PycharmProjects\Project1\venv\Scripts\python.exe C:/Users/TwitterStore/PycharmProjects/Project1/main.py
The Web Server is Ready ....
Received sentence From The Client (web browser): GET /wrong HTTP/1.1
Host: localhost:6060
Connection: keep-alive
Cache-Control: max-age=0
sec-ch-ua: "Chromium";v="124", "Google Chrome";v="124", "Not-A.Brand";v="99"
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/124.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,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, zstd
Accept-Language: ar,en-US;q=0.9,en;q=0.8
```

The status bar at the bottom indicates "Download pre-built shared indexes. Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download once // Don't show again // Configure... (today 4:04 PM)" and "22:1 CRLF UTF-8 4 spaces Python 3.10 (Project1)".

Figure 52:Request any File Does Not Contain in The Server Http requested massge .