

Socket Programming and Web Development

Project No One

Friday, May 10, 2024

Group Members

leelyan karajah 1201191

Dareen muhanna 1202963

Section :4

Instructor : Dr. Ibrahim Nemer

Abstract

This project will teach us a variety of skills. We will track packets by using the "Command Prompt" program to run various commands, and we will use Wireshark to examine various DNS messages.

Then, since they communicate with one another over TCP connections, we will utilize socket programming to create a basic client-server system.

Ultimately, we will use socket programming to build a web server that can process many files and respond to various HTTP requests from the client. In addition to building an HTML and CSS web page with the details of the group members on it.

Contents

Abstract	2
Part 1:	6
1. In your own words, what are ping, tracert, nslookup, and telnet (write one sentence for each one).	6
2. Make sure that your computer is connected to the internet and then run the following commands:	6
A. Ping a device in the same network, e.g. from a laptop to a smartphone	6
B. ping www.stanford.edu.....	7
C. From the ping results, do you think the response you got is from USA? Explain your answer briefly.....	8
D. tracert www.stanford.edu.....	8
E. nslookup www.stanford.edu	9
3. use wireshark to capture some DNS messages.	10
Part 2:	11
Server Code.....	11
Explanation of server code	12
Client Code	13
Explanation of client code.....	14
Client Input code	15
Server Output code.....	15
Part 3:	16
Web Server.....	16
0. From rfce2616, what is Entity Tag Cache Validators in the HTTP protocol and why do we need it?.....	19

if 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.....	20
1. If the request is /ar then the server response with main_ar.html which is an Arabic version of main_en.html.....	26
2. 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. Make it general (not only for specific filename)	27
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. Make it general (not only for specific filename).....	28
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. Make it general (not only for specific filename).....	29
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. Make it general (not only for specific filename).....	30
7. Use myform.html to get image by typing the name of the image in a box.	31
8. Use the status code 307 Temporary Redirect to redirect the following	32
9. 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).....	34
10. Test the project from a browser on the same computer and from a different computer or phone.....	35
Codes.....	36
English version	36
Arabic version	47

Figure 1: ping a device in the same network	7
Figure 2: www.stanford.edu.....	7
Figure 3 : www.stanford.edu.....	8
Figure 4: nslookup www.standford.edu.....	9
Figure 5: capture some DNS messages	10
Figure 6:server codes	11
Figure 7:client code	13
Figure 8:client input.....	15
Figure 9:server output code.....	15
Figure 10:web server	16
Figure 11:The HTTP request ‘/index.html’	20
Figure 12: The HTTP request ‘/main_en.html’	20
Figure 13:The HTTP request ‘/en’	21
Figure 14:HTTP request for ‘/en’	21
Figure 15:HTTP request for ‘/main_en.html’	22
Figure 16:HTTP request for ‘/index.html’	22
Figure 17:The HTTP request ‘/main_ar.html’	26
Figure 18:The HTTP request ‘/file.html’ redirects the client to a webpage designed by html.....	27
Figure 19:The HTTP request ‘/style.css’ redirects the client to the css file stored on the computer (The same path as the web server).	28
Figure 20:The HTTP request ‘/image.png’ redirects the client to a png image stored on the computer (The same path as the web server).	29
Figure 21:The HTTP request ‘/image.jpg’ redirects the client to a jpg image stored on the computer (The same path as the web server).	30
Figure 22:The HTTP request ‘/Myform.html’	32
Figure 23:stackoverflow.com website	32
Figure 24:itc.birzeit.edu website	33
Figure 25:HTML english version.....	36
Figure 26:cssenglish version	39
Figure 27:HTMLARABIC version	47
Figure 28:CSS arabic version.....	49

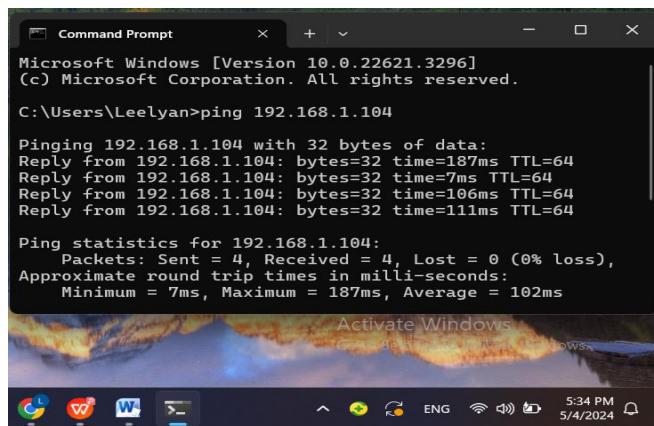
Part 1:

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 popular utility tool for troubleshooting. Help to test host/server reachability, internet connection, network interface card, and DNS difficulties.
 - **Traceroute:** It is used to determine the exact path that data takes on its way to the destination by sending Internet Control Message Protocol (ICMP) echo packets to it.
 - **NSLOOKUP:** It is a utility that finds 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 enables two-way collaborative and text-based communication between two machines, as well as the ability 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:

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



```
Microsoft Windows [Version 10.0.22621.3296]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Leelyan>ping 192.168.1.104

Pinging 192.168.1.104 with 32 bytes of data:
Reply from 192.168.1.104: bytes=32 time=187ms TTL=64
Reply from 192.168.1.104: bytes=32 time=7ms TTL=64
Reply from 192.168.1.104: bytes=32 time=106ms TTL=64
Reply from 192.168.1.104: bytes=32 time=111ms TTL=64

Ping statistics for 192.168.1.104:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 7ms, Maximum = 187ms, Average = 102ms

Activate Windows
Copyright © 2023 Microsoft Corporation. All rights reserved.

C:\Users\Leelyan>
```

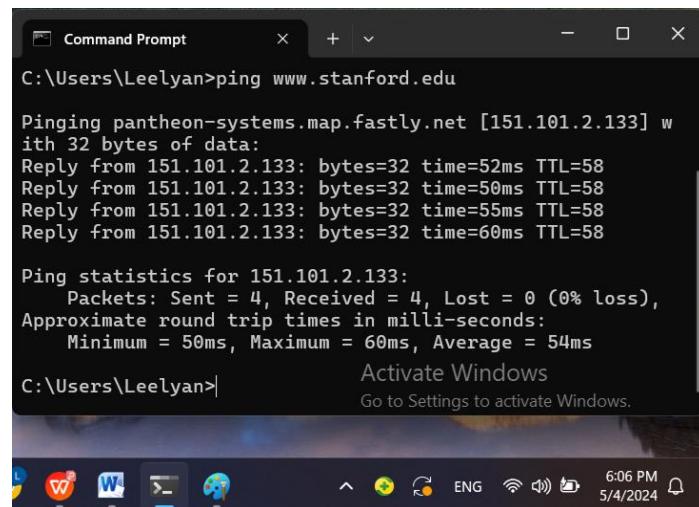
Figure 1: ping a device in the same network

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 were successfully sent and received, the average round-trip time was 102 milliseconds

B. ping www.stanford.edu



A screenshot of a Windows Command Prompt window titled "Command Prompt". The window shows the command "ping www.stanford.edu" being run from the directory "C:\Users\Leelyan". The output of the ping command is displayed, showing four replies from the IP 151.101.2.133 with times ranging from 50ms to 60ms. Below the ping output, the command prompt shows "Activate Windows" and "Go to Settings to activate Windows." The taskbar at the bottom of the screen includes icons for File Explorer, Edge, Word, and others, along with system status indicators like battery level and signal strength. The date and time are shown as 5/4/2024 and 6:06 PM respectively.

Figure 2: www.stanford.edu

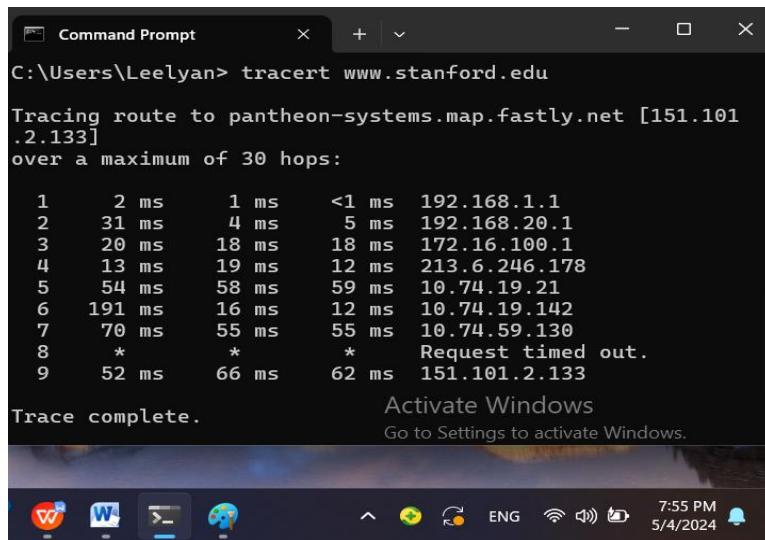
We used cmd to perform ping www.stanford.edu, we saw the round-trip time (in milliseconds) for each packet, the IP address of "pantheon systems.map.fastly.net" is 151.101.2.133. It also shows the replay of each message

containing the size of each packet in bytes, the time from sending to receiving this packet and TTL which is the number of nodes each packet has to pass through.

- C. From the ping results, do you think the response you 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.stanford.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 (151.101.2.133) 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.

D. tracert www.stanford.edu



A screenshot of a Windows Command Prompt window titled "Command Prompt". The window shows the command "tracert www.stanford.edu" being run. The output displays the tracing route to the server pantheon-systems.map.fastly.net with an IP of 151.101.2.133, over a maximum of 30 hops. The route consists of nine nodes, with the last node being the destination. The command concludes with "Trace complete." and a message to activate Windows.

```
C:\Users\Leelyan> tracert www.stanford.edu
Tracing route to pantheon-systems.map.fastly.net [151.101.2.133]
over a maximum of 30 hops:
1    2 ms      1 ms    <1 ms   192.168.1.1
2    31 ms     4 ms      5 ms   192.168.20.1
3    20 ms     18 ms     18 ms   172.16.100.1
4    13 ms     19 ms     12 ms   213.6.246.178
5    54 ms     58 ms     59 ms   10.74.19.21
6    191 ms    16 ms     12 ms   10.74.19.142
7    70 ms     55 ms     55 ms   10.74.59.130
8    *         *         *       Request timed out.
9    52 ms     66 ms     62 ms   151.101.2.133
Trace complete.
```

Figure 3 : www.stanford.edu

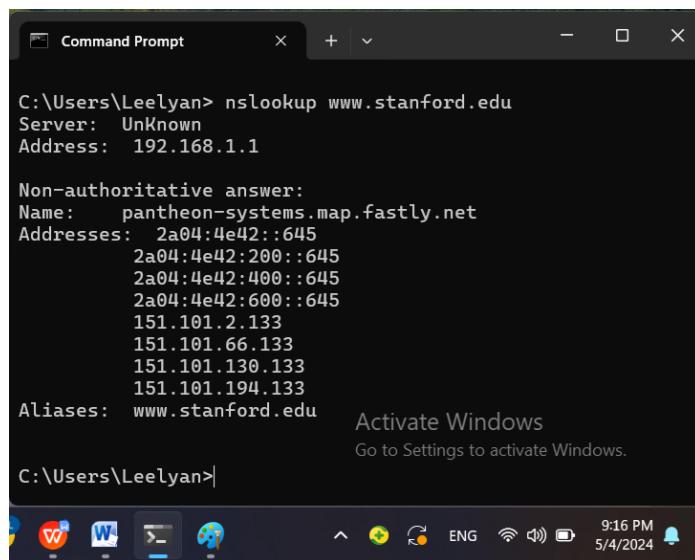
We used Cmd to perform traceroute, the first line shows the target IP address or domain being traced and its corresponding IP address. The Column Headers:

- **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 represents 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.

For the "Request timed out" in a traceroute result, it means that the ICMP (Internet Control Message Protocol) echo request sent to the specific hop did not receive a response within the timeout period.

E. nslookup www.stanford.edu



The screenshot shows a Windows Command Prompt window titled "Command Prompt". The command entered is "nslookup www.stanford.edu". The output displays the following information:

```
C:\Users\Leelyan> nslookup www.stanford.edu
Server: Unknown
Address: 192.168.1.1

Non-authoritative answer:
Name: pantheon-systems.map.fastly.net
Addresses: 2a04:4e42::645
           2a04:4e42:200::645
           2a04:4e42:400::645
           2a04:4e42:600::645
           151.101.2.133
           151.101.66.133
           151.101.130.133
           151.101.194.133
Aliases: www.stanford.edu

C:\Users\Leelyan>
```

The window also includes a watermark for "Activate Windows" and the date and time "5/4/2024 9:16 PM".

Figure 4: nslookup www.stanford.edu

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 "192.168.1.1,"

- **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.stanford.edu" resolves to "pantheon-systems.map.fastly.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.

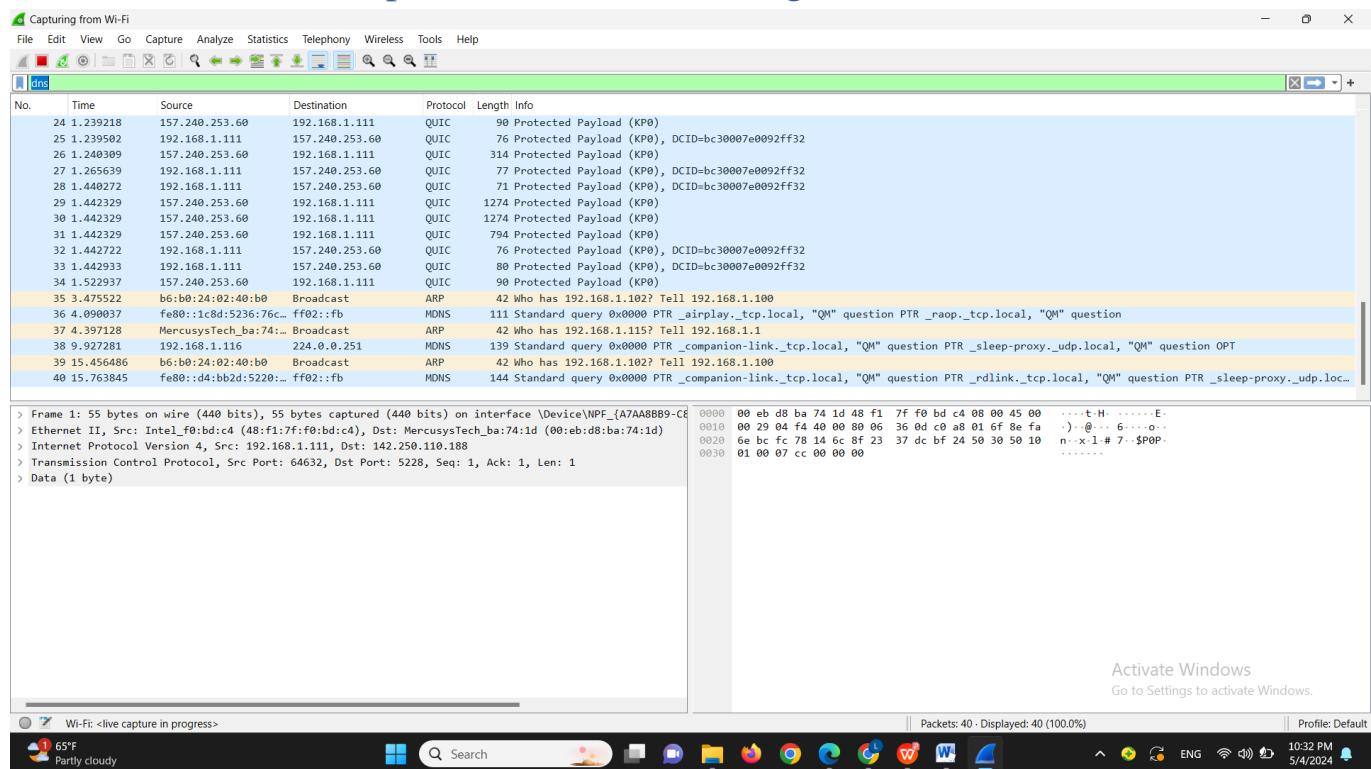


Figure 5: capture some DNS messages

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.

Part 2: Server Code

Figure 6:server codes

A screenshot of a code editor window titled "parttwoProject". The current file is "server.py". The code defines a class "Display" with an __init__ method that initializes first_name, last_name, message, and timestamp. It also contains a handle_client method that listens on port 1024, receives messages from clients, and prints them to the console. A "display_messages" method is used to print all received messages. The main function creates a socket, binds it to ("0.0.0.0", 5051), and starts a thread to handle client connections. If the name is "__main__", it calls the main() function.

```
# server.py
import socket
import threading
import time

class Display:
    def __init__(self, first_name, last_name, message, timestamp):
        self.first_name = first_name
        self.last_name = last_name
        self.message = message
        self.timestamp = timestamp

    messages = {}
    message_lock = threading.Lock()

    def handle_client(sock, addr):
        while True:
            data, _ = sock.recvfrom(1024)
            message = data.decode().split(' ', 2)
            if len(message) != 3:
                print("Invalid message format")
                continue
            first_name, last_name, msg = message
            timestamp = time.strftime("%a %b %d %H:%M:%S %Y", time.localtime())
            with message_lock:
                messages[(first_name, last_name)] = Display(first_name, last_name, msg, timestamp)
            print(f"Received message from {first_name} {last_name} at {timestamp}")
            display_messages()

    def display_messages():
        if not messages:
            print("No messages to display.")
            return
        print("Messages:")
        for i, (_, display) in enumerate(messages.items()):
            print(f"{i + 1}- From {display.first_name} {display.last_name} at {display.timestamp}: {display.message}")

    def main():
        sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        sock.bind(("0.0.0.0", 5051))
        print("Server listening on port 5051")

        while True:
            data, addr = sock.recvfrom(1024)
            client_thread = threading.Thread(target=handle_client, args=(sock, addr))
            client_thread.start()

    if __name__ == "__main__":
        main()
```

A screenshot of a code editor window titled "parttwoProject". The current file is "server.py". The code is identical to the one in the previous screenshot, but it includes a "display_messages" method that prints all received messages. The main function creates a socket, binds it to ("0.0.0.0", 5051), and starts a thread to handle client connections. If the name is "__main__", it calls the main() function.

```
# server.py
import socket
import threading
import time

class Display:
    def __init__(self, first_name, last_name, message, timestamp):
        self.first_name = first_name
        self.last_name = last_name
        self.message = message
        self.timestamp = timestamp

    messages = {}
    message_lock = threading.Lock()

    def handle_client(sock, addr):
        while True:
            data, _ = sock.recvfrom(1024)
            message = data.decode().split(' ', 2)
            if len(message) != 3:
                print("Invalid message format")
                continue
            first_name, last_name, msg = message
            timestamp = time.strftime("%a %b %d %H:%M:%S %Y", time.localtime())
            with message_lock:
                messages[(first_name, last_name)] = Display(first_name, last_name, msg, timestamp)
            print(f"Received message from {first_name} {last_name} at {timestamp}")
            display_messages()

    def display_messages():
        if not messages:
            print("No messages to display.")
            return
        print("Messages:")
        for i, (_, display) in enumerate(messages.items()):
            print(f"{i + 1}- From {display.first_name} {display.last_name} at {display.timestamp}: {display.message}")

    def main():
        sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        sock.bind(("0.0.0.0", 5051))
        print("Server listening on port 5051")

        while True:
            data, addr = sock.recvfrom(1024)
            client_thread = threading.Thread(target=handle_client, args=(sock, addr))
            client_thread.start()

    if __name__ == "__main__":
        main()
```

Explanation of server code

This simple Python script functions, as a UDP server, monitoring and showing messages. It can handle clients simultaneously with each client being handled through threads.

Using Sockets for Networking: The script creates a UDP server to receive incoming data. This makes it lighter, but less dependable than TCP.

Threading for concurrency:

The script uses threads to allow multiple clients to transmit messages concurrently without causing problems. Each incoming message is handled by its own thread, allowing the server to process numerous messages concurrently.

Data Storage:

The server saves receiving messages in a messages dictionary with a key that includes the sender's first and last name. This storage allows the server to keep track of every message and display them

Threading occurs when many threads simultaneously ask the messages dictionary. The lock is used to ensure that only one thread can edit a dictionary at a time. This prevents from data loss and keeps thread safety.

The handle_client function listens for messages decodes them. Gathers the senders first and last names, along, with the message content. It also creates a timestamp to show when the message was received.

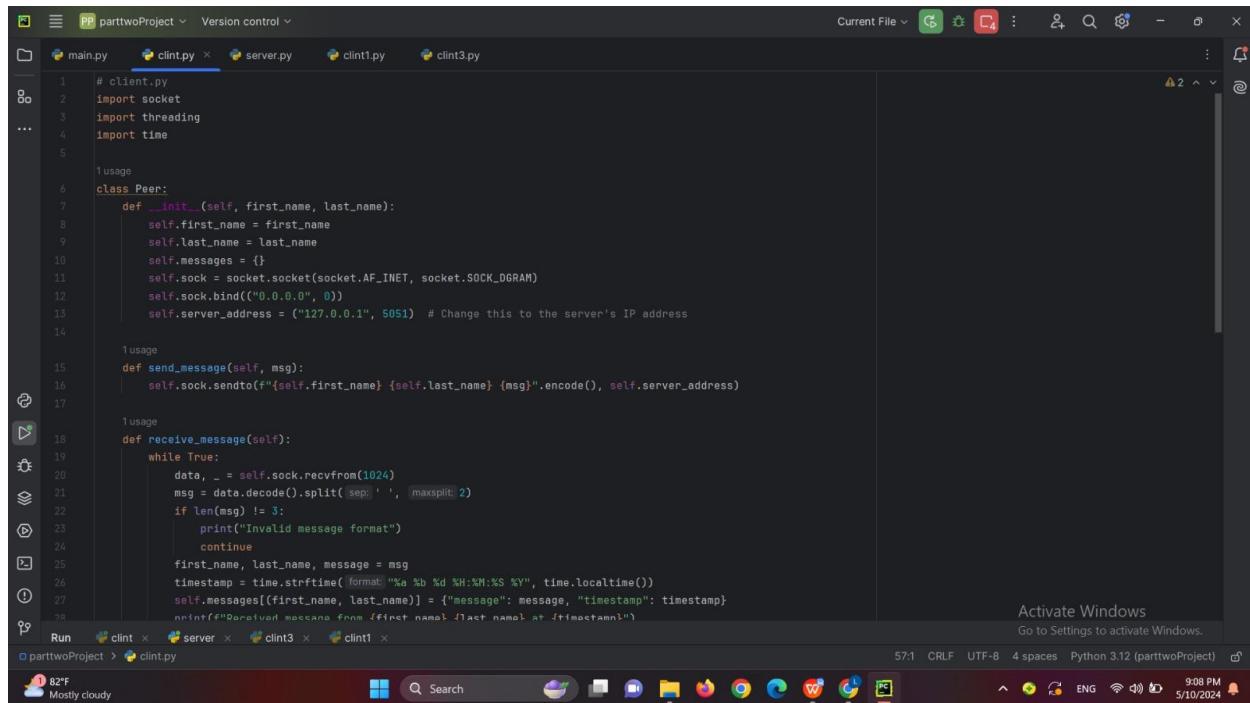
After ensuring thread safety using a lock the function adds the message to the messages dictionary. Then calls display_messages() to show the updated message list.

Showing messages;

The display_messages function outputs all saved messages on the screen. If there are no messages it indicates that the message list is empty.

Client Code

Figure 7:client code



```
# client.py
import socket
import threading
import time

# Usage
class Peer:
    def __init__(self, first_name, last_name):
        self.first_name = first_name
        self.last_name = last_name
        self.messages = {}
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        self.sock.bind(("0.0.0.0", 0))
        self.server_address = ("127.0.0.1", 5051) # Change this to the server's IP address

    def send_message(self, msg):
        self.sock.sendto(f"{self.first_name} {self.last_name} {msg}".encode(), self.server_address)

    def receive_message(self):
        while True:
            data, _ = self.sock.recvfrom(1024)
            msg = data.decode().split(' ', maxsplit=2)
            if len(msg) != 3:
                print("Invalid message format")
                continue
            first_name, last_name, message = msg
            timestamp = time.strftime("%a %b %d %H:%M:%S %Y", time.localtime())
            self.messages[(first_name, last_name)] = {"message": message, "timestamp": timestamp}
            print(f"Received message from {first_name} {last_name} at {timestamp}")

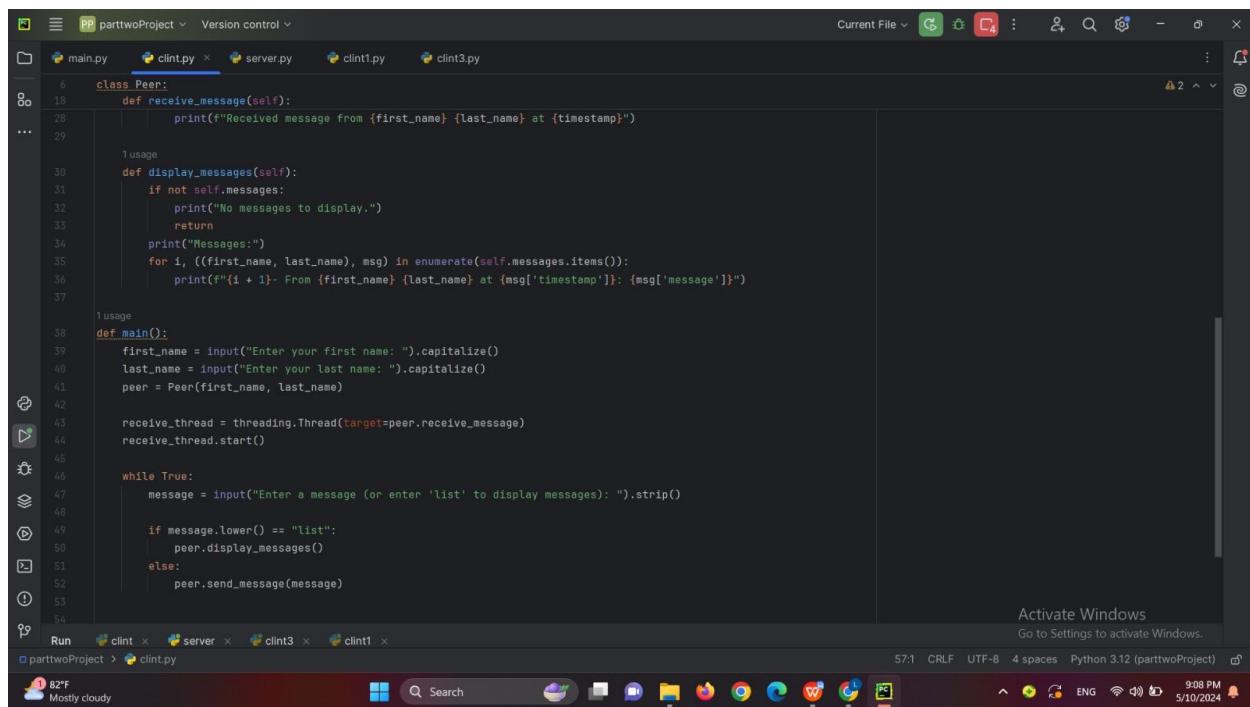
# Usage
def main():
    first_name = input("Enter your first name: ").capitalize()
    last_name = input("Enter your last name: ").capitalize()
    peer = Peer(first_name, last_name)

    receive_thread = threading.Thread(target=peer.receive_message)
    receive_thread.start()

    while True:
        message = input("Enter a message (or enter 'list' to display messages): ").strip()

        if message.lower() == "list":
            peer.display_messages()
        else:
            peer.send_message(message)

if __name__ == "__main__":
    main()
```



```
class Peer:
    def receive_message(self):
        print(f"Received message from {first_name} {last_name} at {timestamp}")

    def display_messages(self):
        if not self.messages:
            print("No messages to display.")
            return
        print("Messages:")
        for i, ((first_name, last_name), msg) in enumerate(self.messages.items()):
            print(f"{i + 1}- From {first_name} {last_name} at {msg['timestamp']}: {msg['message']}")

    def main():
        first_name = input("Enter your first name: ").capitalize()
        last_name = input("Enter your last name: ").capitalize()
        peer = Peer(first_name, last_name)

        receive_thread = threading.Thread(target=peer.receive_message)
        receive_thread.start()

        while True:
            message = input("Enter a message (or enter 'list' to display messages): ").strip()

            if message.lower() == "list":
                peer.display_messages()
            else:
                peer.send_message(message)

if __name__ == "__main__":
    main()
```

```

6     class Peer:
7         def __init__(self, first_name, last_name):
8             self.first_name = first_name
9             self.last_name = last_name
10            self.messages = {}
11
12            self.sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
13            self.sock.bind((self.last_name, 0))
14
15            self.server_address = ('localhost', 5051)
16
17            self.peer_name = f'{self.first_name} {self.last_name}'
18
19
20        def send_message(self, message):
21            msg = {'first_name': self.first_name, 'last_name': self.last_name, 'message': message}
22            self.sock.sendto(str.encode(json.dumps(msg)), self.server_address)
23
24
25        def receive_message(self):
26            data, address = self.sock.recvfrom(1024)
27            msg = json.loads(data.decode())
28            self.messages[msg['timestamp']] = msg['message']
29
30
31        def display_messages(self):
32            for i, (first_name, last_name), msg in enumerate(self.messages.items()):
33                print(f'{i + 1}- From {first_name} {last_name} at {msg["timestamp"]}: {msg["message"]}')
34
35
36    def main():
37        first_name = input("Enter your first name: ").capitalize()
38        last_name = input("Enter your last name: ").capitalize()
39        peer = Peer(first_name, last_name)
40
41        receive_thread = threading.Thread(target=peer.receive_message)
42        receive_thread.start()
43
44        while True:
45            message = input("Enter a message (or enter 'list' to display messages): ").strip()
46
47            if message.lower() == "list":
48                peer.display_messages()
49            else:
50                peer.send_message(message)
51
52
53
54
55    if __name__ == "__main__":
56        main()
57

```

Explanation of client code

The Peer Class

Initializes with the peer's first and last name, plus an empty dictionary for storing messages.

Creates a UDP socket and assigns it a random port.

Defines the server address to which messages should be sent (in this scenario, localhost at port 5051 is assumed).

`send_message(msg)`: Provides a message to the server. The message includes the sender's first and last name, and also the message content.

`receive_message()`: Responds regularly for incoming messages from the server.

When a message is received, it takes the sender's name, message content, and timestamp before storing it in the peer's message dictionary.

`display_messages()` shows all messages sent by the peer.

In The main method :The script creates a Peer class object, listens for incoming messages, and enters a loop for sending and displaying received messages. If the user enters "list", the script displays the received messages.

Client Input code

Figure 8:client input

The screenshot shows the PyCharm IDE interface with the 'parttwoProject' project open. The 'clint3.py' file is the active file, displaying Python code for a client application. The code defines a main function that starts a receive thread and enters a loop to read user input. If the input is 'list', it calls a peer's display_messages method; otherwise, it sends the message to the peer. The terminal window below shows the execution of the script and interactions with the user:

```
C:\Users\Leelyan\PycharmProjects\parttwoProject\.venv\Scripts\python.exe C:\Users\Leelyan\PycharmProjects\parttwoProject\clint3.py
Enter your first name: dareen
Enter your last name: muhanna
Enter a message (or enter 'list' to display messages): kk
Enter a message (or enter 'list' to display messages):
```

The status bar at the bottom indicates Python 3.12 (parttwoProject) and the system date and time.

Server Output code

Figure 9:server output code

The screenshot shows the PyCharm IDE interface with the 'parttwoProject' project open. The 'server.py' file is the active file, displaying Python code for a server application. The code defines a main function that starts a receive thread and enters a loop to read messages from clients. It prints received messages to the terminal. The terminal window below shows the execution of the script and the printed messages from multiple clients:

```
C:\Users\Leelyan\PycharmProjects\parttwoProject\.venv\Scripts\python.exe C:\Users\Leelyan\PycharmProjects\parttwoProject\server.py
Server listening on port 5051
Received message from Dareen Muhanna at Fri May 10 21:05:05 2024
Messages:
1- From Daireen Muhanna at Fri May 10 21:05:05 2024: kk
Received message from Leelyan Karajah at Fri May 10 21:05:48 2024
Messages:
1- From Daireen Muhanna at Fri May 10 21:05:05 2024: kk
2- From Leelyan Karajah at Fri May 10 21:05:48 2024: kk
Received message from Mohamed Ahd at Fri May 10 21:05:56 2024
Messages:
1- From Daireen Muhanna at Fri May 10 21:05:05 2024: kk
2- From Leelyan Karajah at Fri May 10 21:05:48 2024: kk
3- From Mohamed Ahd at Fri May 10 21:05:56 2024: kk
```

The status bar at the bottom indicates Python 3.12 (parttwoProject) and the system date and time.

Part 3:

Using socket programming, implement a simple but a complete web server in go, python, java or C that is listening on port 6060. Make the code as general as possible

Web Server

Figure 10:web server

The screenshot shows the PyCharm IDE interface. The project tree on the left contains files like ar.html, dareen.jpeg, file.html, image.jpg, image.png, leelyan.jpg, main.py, main_ar.html, main_en.html, Myform.html, network.png, NotFound.html, sky.jpg, style.css, styleAr.css, teddy.png, tset.py, and webserver.py. The main editor window displays the code for `webserver.py`. The code defines a `Serve` function that handles file requests and a `HandleRequests` function that processes HTTP requests. It also includes a `ProcessRequests` function and a check for GET requests. The status bar at the bottom shows Python 3.12 (InetworkProject).

```
def Serve(path, type):
    try:
        with open(path, 'rb') as file:
            # Read the content of the file
            content = file.read()
            response = f"HTTP/1.1 200 OK\nContent-Type: {type}\n\n"
            response += content
    except FileNotFoundError:
        response = "HTTP/1.1 404 Not Found\n\nPage not found".encode('utf-8')

    return response

def HandleRequests(clientSocket,clientAddress):
    # Handle the request and send a response
    # Receive client requests
    request = clientSocket.recv(1024).decode('utf-8')
    print(f"Received HTTP request: {request}")

    # Return response
    response = ProcessRequests(request,clientAddress)

    # Send response
    clientSocket.sendall(response)

    # Close the connection with the client
    clientSocket.close()

# A Function to handle the client requests
"""
This function takes the HTTP request and extracts
the method and path from it to process the the request based on them
"""

def ProcessRequests(request,clientAddress):
    """
    # Check if the method is GET
    
```

The screenshot shows the PyCharm IDE interface. The project tree on the left contains files like ar.html, dareen.jpeg, file.html, image.jpg, image.png, leelyan.jpg, main.py, main_ar.html, main_en.html, Myform.html, network.png, NotFound.html, sky.jpg, style.css, styleAr.css, teddy.png, tset.py, and webserver.py. The main editor window displays the code for `webserver.py`. The code defines a `Serve` function that attempts to open a file and read its content, constructs an HTTP response, and returns it. It also includes a `HandleRequests` function and a `ProcessRequests` function. The status bar at the bottom shows Python 3.12 (InetworkProject).

```
import socket
import threading

# Define the IP address(for host) and Port
HOST = socket.gethostname()
# HOST = "127.0.0.1"
PORT = 6060

# Creating socket
serverSocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Bind the socket to the specific host and port
serverSocket.bind(("0.0.0.0", PORT))

# A Function to serve HTML files
"""
This function reads the content of the file,
constructs the HTTP response
and return the response
"""
6 usages
def Serve(path, type):
    # Attempt to open the file
    try:
        with open(path, 'rb') as file:
            # Read the content of the file
            content = file.read()
            response = f"HTTP/1.1 200 OK\nContent-Type: {type}\n\n"
            response += content
    except FileNotFoundError:
        response = "HTTP/1.1 404 Not Found\n\nPage not found".encode('utf-8')

    return response

def HandleRequests(clientSocket,clientAddress):
    # Handle the request and send a response
    # Receive client requests
    request = clientSocket.recv(1024).decode('utf-8')
    print(f"Received HTTP request: {request}")

    # Return response
    response = ProcessRequests(request,clientAddress)

    # Send response
    clientSocket.sendall(response)

    # Close the connection with the client
    clientSocket.close()

# A Function to handle the client requests
"""
This function takes the HTTP request and extracts
the method and path from it to process the the request based on them
"""

def ProcessRequests(request,clientAddress):
    """
    # Check if the method is GET
    
```

The screenshot shows the PyCharm IDE interface with the 'Project' tool window on the left and the code editor on the right. The code editor displays the 'webserver.py' file, which contains Python code for a web server. The code defines a function 'ProcessRequests' that handles HTTP requests based on the path. It checks for GET methods, extracts paths, prints received requests, and serves files like 'main_en.html', 'main_ar.html', 'style.css', and various image files ('dareen.jpeg', 'image.jpg', etc.). The code also includes logic for redirecting to 'stackoverflow.com' or 'itc.birzeit.edu' if the requested path is '/so' or '/itc'. A comment at the bottom indicates a default 404 error message. The status bar at the bottom shows system information including weather (93°F, Partly sunny), network, battery, and time (11:17 AM, 5/10/2024).

```
def ProcessRequests(request,clientAddress):
    # Check if the method is GET
    if request.startswith("GET /"):

        # Extract the path from the request
        path = request.split()[1]

        # Print the received HTTP request to the terminal
        print(f"Received HTTP request: {request}")

        # Send main_en.html file with Content-Type: text/html
        if path == '/' or path == '/index.html' or path == '/main_en.html' or path == '/en':
            return Serve(path='main_en.html', type='text/html')

        # Response with main_ar.html which is an Arabic version of main_en.html
        elif path == '/ar':
            return Serve(path='main_ar.html', type='text/html')

        # Send the requested html file with Content-Type: text/html
        elif path.endswith('.html'):
            return Serve(path[1:], type='text/html')

        # Send the requested css file with Content-Type: text/css
        elif path.endswith('.css'):
            return Serve(path[1:], type='text/css')

        # Send the png image with Content-Type: image/png
        elif path.endswith('.png'):
            return Serve(path[1:], type='image/png')

        # Send the jpg image with Content-Type: image/jpeg
        elif path.endswith('.jpg'):
```

This screenshot shows the same PyCharm environment with the 'webserver.py' file open. The code has been modified to include a 'usage' section and an 'errorMSG' function. The 'usage' section provides a template for generating an error page with client information. The 'errorMSG' function constructs a response for a 404 error, replacing placeholder '_old' with the client's IP address and port. The rest of the code remains the same, handling paths and serving files. The status bar at the bottom shows the same system information as the first screenshot.

```
def ProcessRequests(request,clientAddress):
    # Send the jpg image with Content-Type: image/jpeg
    elif path.endswith('.jpg'):
        return Serve(path[1:], type='image/jpeg')

    # Redirect to stackoverflow.com website
    elif path == '/so':
        return "HTTP/1.1 307 Temporary Redirect\nLocation: https://stackoverflow.com\n\n".encode('utf-8')
    # Redirect to itc website
    elif path == '/itc':
        return "HTTP/1.1 307 Temporary Redirect\nLocation: https://itc.birzeit.edu\n\n".encode('utf-8')

    # Default: 404 Not Found
    return errorMSG(clientAddress)

usage
def errorMSG(clientAddress):
    response = "HTTP/1.1 404 Not Found\nContent-Type: text/html\n\n"

    try:
        with open("NotFound.html", "r") as errorFile:
            error_content = errorFile.read()
            error_content = error_content.replace("_old: [Client IP address and Port]", _new=f"{clientAddress[0]}")
            response += error_content
    except FileNotFoundError:
        response += "<html><body><h1>404 Not Found</h1><p>The requested resource was not found.</p></body></html>"

    return response.encode('utf-8')
```

The screenshot shows the PyCharm IDE interface. The left sidebar displays the project structure for '1networkProject' containing files like 'ar.html', 'dareen.jpg', 'file.html', etc., and a 'webserver.py' file which is currently selected. The main editor window shows the code for 'webserver.py'. The code defines a class 'HandleRequests' with methods 'errorMSG' and 'start'. It uses the 'socket' and 'threading' modules to handle incoming connections and serve files. A conditional block checks if the script is run as the main module to start the server. The status bar at the bottom indicates the file is saved in 'C:\Users\Leelyan\PycharmProjects\1networkProject\webserver'. The taskbar at the bottom shows various application icons.

```
def errorMSG(clientAddress):
    try:
        response += "<html><body><h1>404 Not Found</h1><p>The requested resource was not found.</p></body></html>"
    except FileNotFoundError:
        response += "<html><body><h1>404 Not Found</h1><p>The requested resource was not found.</p></body></html>"
```

○ Explanation :

- ✓ **Importing Modules:** The code imports the necessary modules `socket` and `threading`.
- ✓ **Defining Host and Port:** It sets the host to the local machine's IP address and the port to 6060.
- ✓ **Creating and Binding Socket:** It creates a TCP socket and binds it to the specified host and port.
- ✓ **Function to Serve HTML Files (Serve()):** Reads and serves HTML files, returning appropriate HTTP responses.
- ✓ **Function to Handle Client Requests (HandleRequests()):** Receives client requests, processes them, and sends responses accordingly.

- ✓ **Function to Process Client Requests (ProcessRequests()):** Processes HTTP requests based on the requested path, serving files or handling redirection.
- ✓ **Function to Handle Error Messages (errorMSG()):** Constructs HTTP responses for "404 Not Found" errors.
- ✓ **Main Server Loop (start()):** Listens for incoming connections, accepts them, and handles each connection in a separate thread.
- ✓ **Main Execution Block:** Starts the server by calling the start() function.

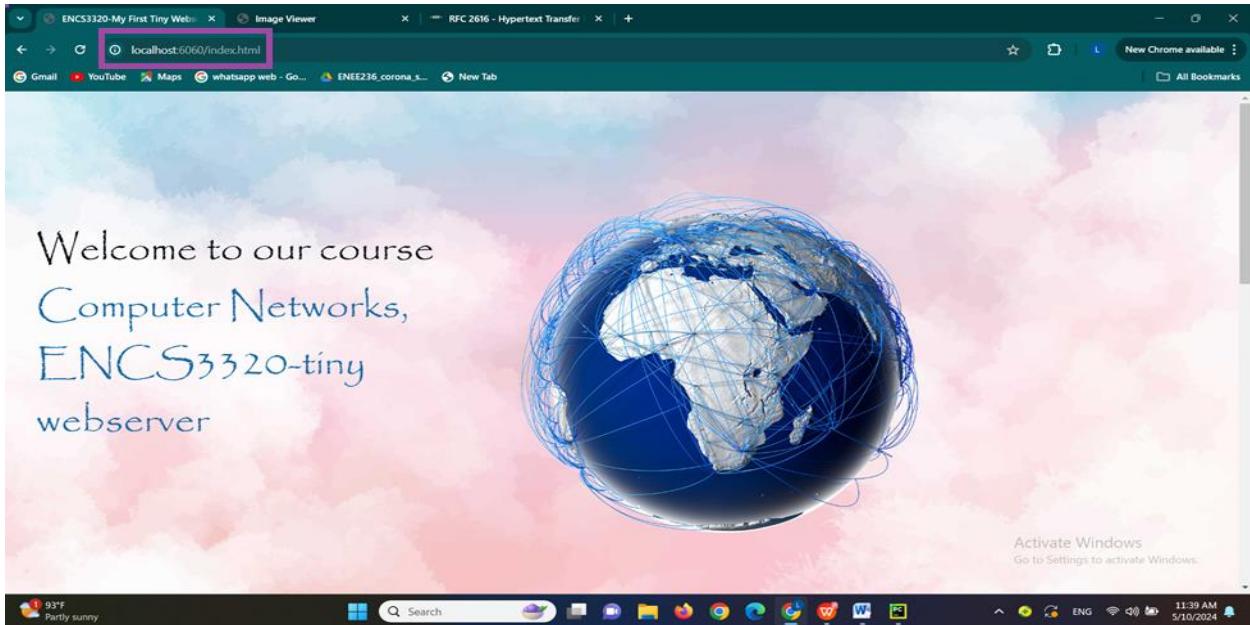
This code essentially creates a basic HTTP server that serves files and handles client requests, all while using multithreading to handle multiple clients simultaneously.

0. From rfce2616, what is Entity Tag Cache Validators in the HTTP protocol and why do we need it?

The ETag response-header field value, an entity tag, provides for an "opaque" cache validator. This might allow more reliable validation in situations where it is inconvenient to store modification dates, where the one-second resolution of HTTP date values is not sufficient, or where the origin server wishes to avoid certain paradoxes that might arise from the use of modification dates

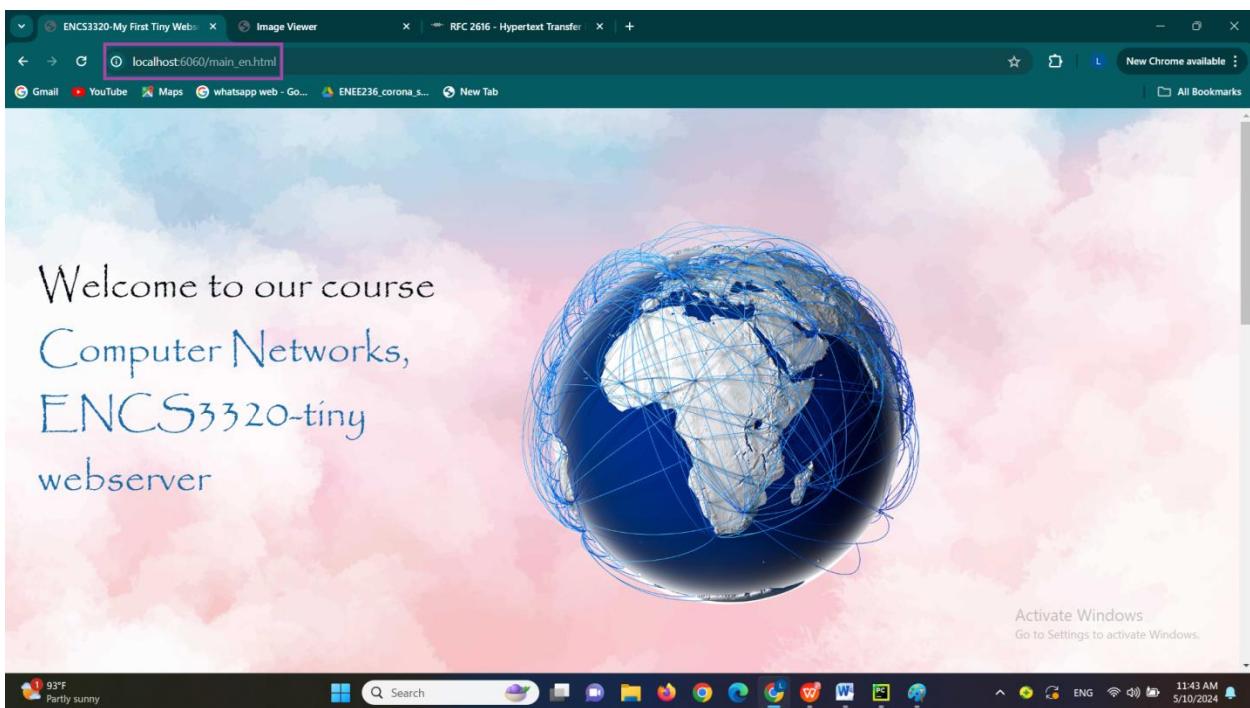
if the request is / or /index.html or /main_en.html or /en (for example localhost:6060/ or localhost:6060/en) then the server should send main_en.html file with Content-Type: text/html.

Figure 11: The HTTP request '/index.html'



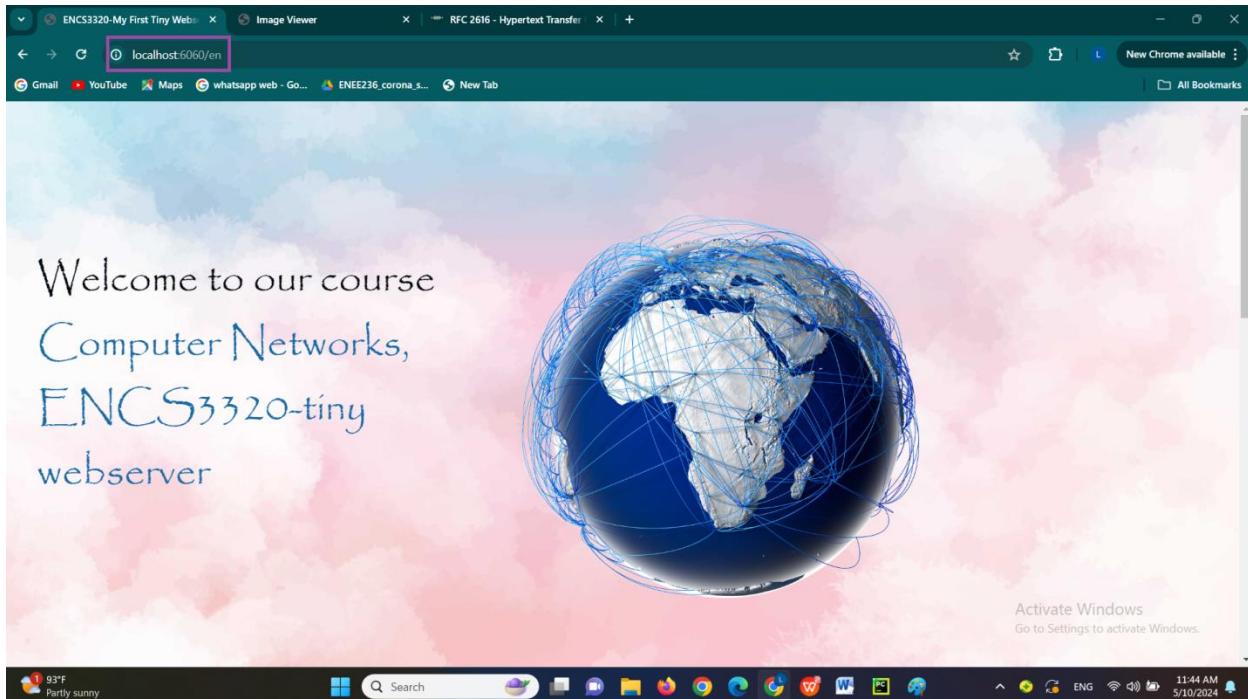
The HTTP request '/index.html' redirects the client to our webpage.

Figure 12: The HTTP request '/main_en.html'



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

Figure 13:The HTTP request '/en'



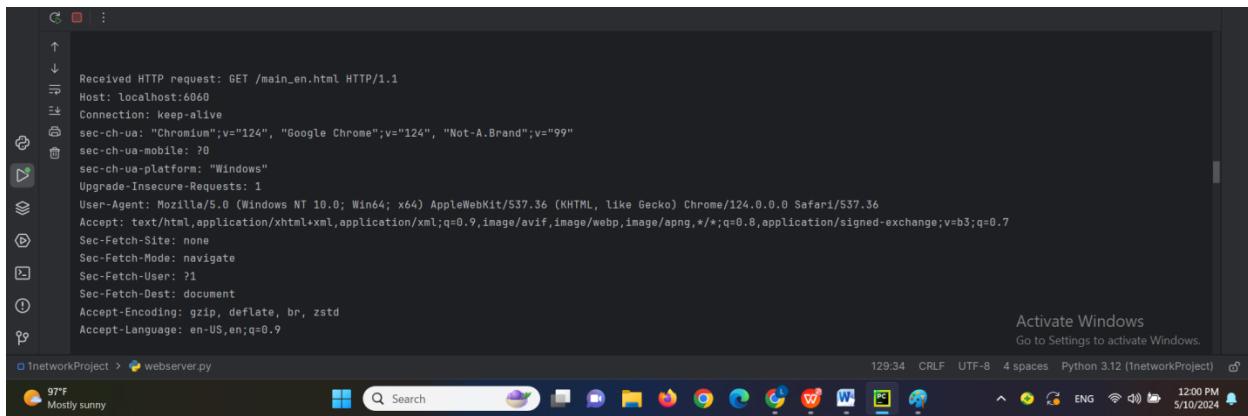
The HTTP request '/en' redirects the client to our webpage.

Figure 14:HTTP request for '/en'

A screenshot of a terminal window titled 'InetworkProject > webserver.py'. The window displays an incoming HTTP request from a client connected to 'localhost:6060'. The request details include the method (GET), path ('/en'), version (HTTP/1.1), host header, connection type (keep-alive), cache control, user agent (Mozilla/5.0), accept header (text/html, application/xhtml+xml, etc.), and other headers like Sec-Ch-Ua and User-Agent. The terminal also shows the current file path and some system status information at the bottom.

HTTP request for '/en'.

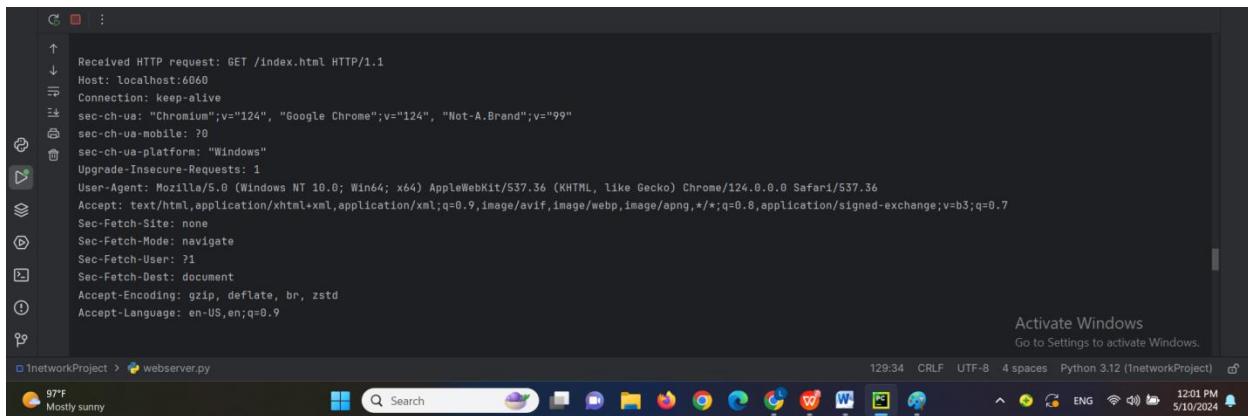
Figure 15:HTTP request for '/main_en.html'



```
Received HTTP request: GET /main_en.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,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: en-US,en;q=0.9
```

HTTP request for '/main_en.html'.

Figure 16:HTTP request for '/index.html'



```
Received HTTP request: 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,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: en-US,en;q=0.9
```

HTTP request for '/index.html'.

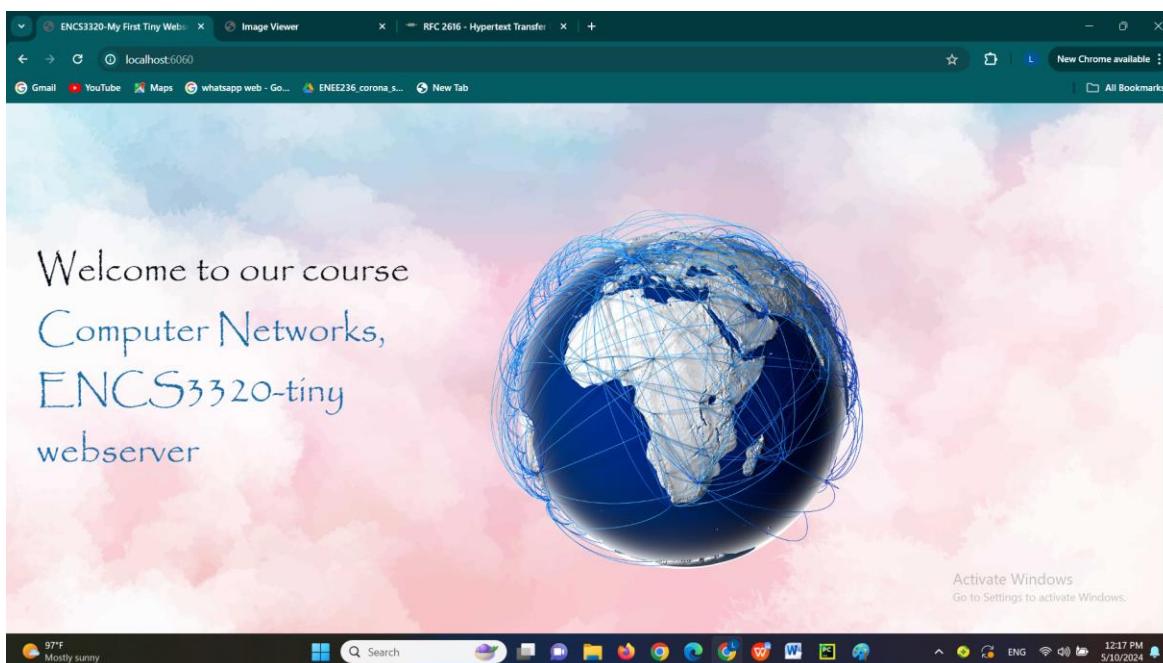
○ Explanation:

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

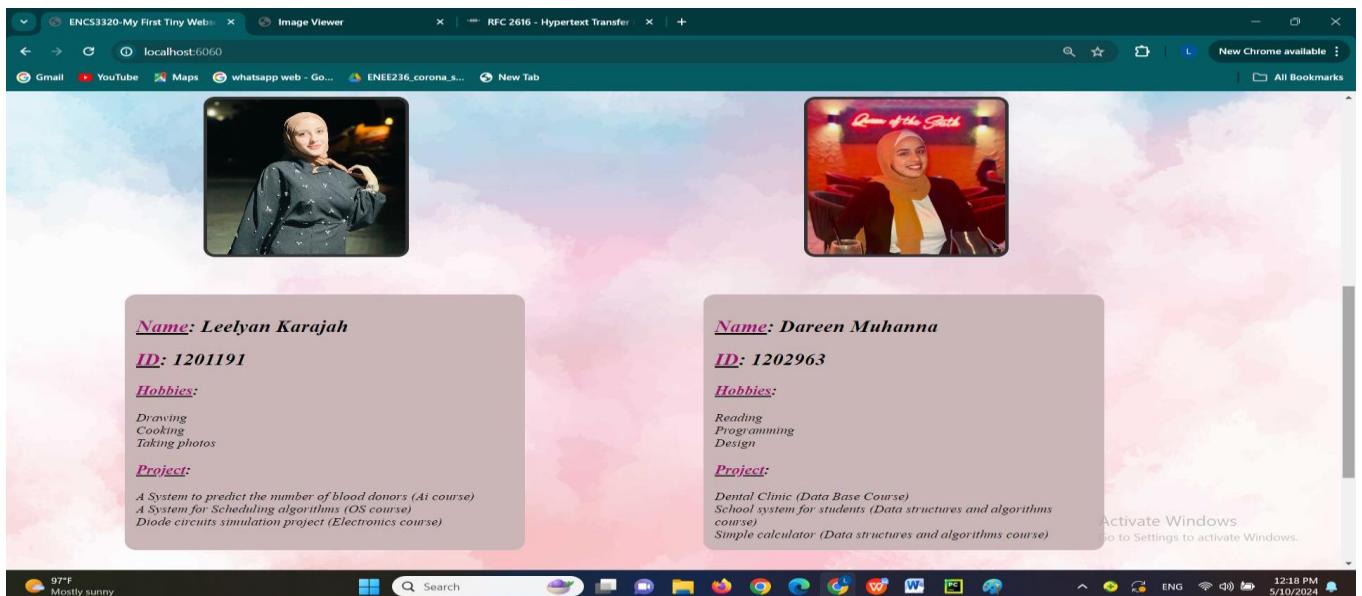
The request line: “GET /main_en.html HTTP/1.1”.

- ✓ **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** the port number: (6060)
- ✓ **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.

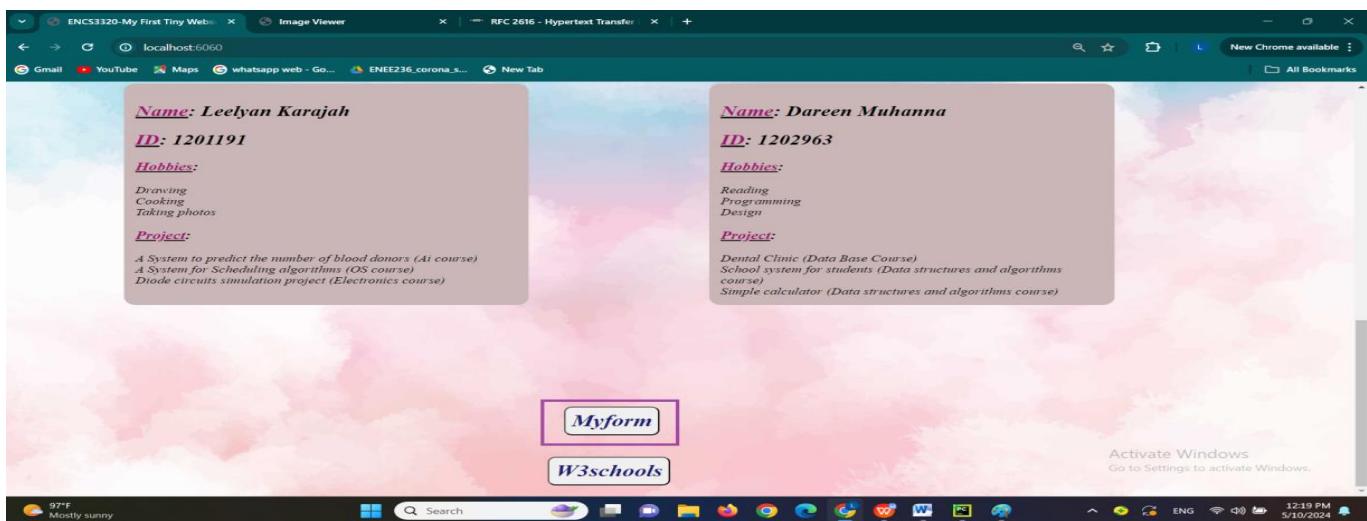
- The **main_en.html** file should contain HTML webpage that contains :
 - a. “ENCS3320-My First Tiny Webserver” in the title.
 - b. “Welcome to our course **Computer Networks, ENCS3320-tiny webserver**” (part of the phrase is in **Blue**).



- c. Group members names and IDs
- d. Some information about the group members. For instance, projects you have done during different course (programming, electrical, math, etc), skills, hobbies, etc.
- e. Use CSS to make the page looks nice
- f. Divide the page in different boxes and put student's information in the different boxes
- g. Include CSS as a separate file
- h. The page should contain at least An image with extention.jpg and an image with extension .png



i. A link to a local html file (myform.html)



j. a link to https://www.w3schools.com/python/python_syntax.asp

Name: Leelyan Karajah
ID: 1201191
Hobbies:
Driving
Cooking
Taking photos
Project:
A System to predict the number of blood donors (Ai course)
A System for Scheduling algorithms (OS course)
Diode circuits simulation project (Electronics course)

Name: Dareen Muhanna
ID: 1202963
Hobbies:
Reading
Programming
Design
Project:
Dental Clinic (Data Base Course)
School system for students (Data structures and algorithms course)
Simple calculator (Data structures and algorithms course)

Myform
W3schools

Activate Windows
Go to Settings to activate Windows.

When you click the button it will take you to
https://www.w3schools.com/python/python_syntax.asp.

Python Tutorial

- Python HOME
- Python Intro
- Python Get Started
- Python Syntax**
- Python Comments
- Python Variables
- Python Data Types
- Python Numbers
- Python Casting
- Python Strings
- Python Booleans
- Python Operators
- Python Lists
- Python Tuples
- Python Sets
- Python Dictionaries
- Python If...Else

Python Syntax

[Previous](#) [Next](#)

Execute Python Syntax

As we learned in the previous page, Python syntax can be executed by writing directly in the Command Line:

```
>>> print("Hello, World!")
Hello, World!
```

Or by creating a python file on the server, using the .py file extension, and running it in the Command Line:

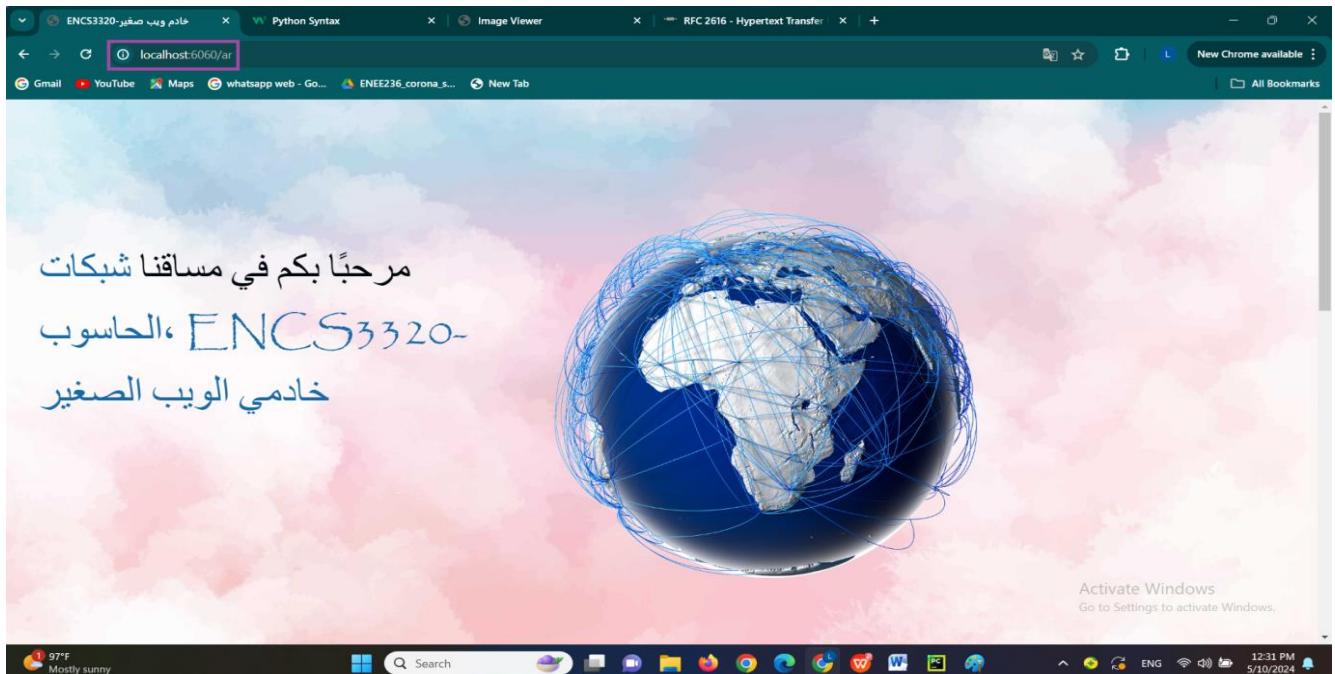
On this page

- Execute Python Syntax
- Python Indentation
- Python Variables
- Python Comments
- Exercises

Activate Windows
Go to Settings to activate Windows.

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

Figure 17: The HTTP request `'/main_ar.html'`



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

```

Received HTTP request: GET /ar 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,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

```

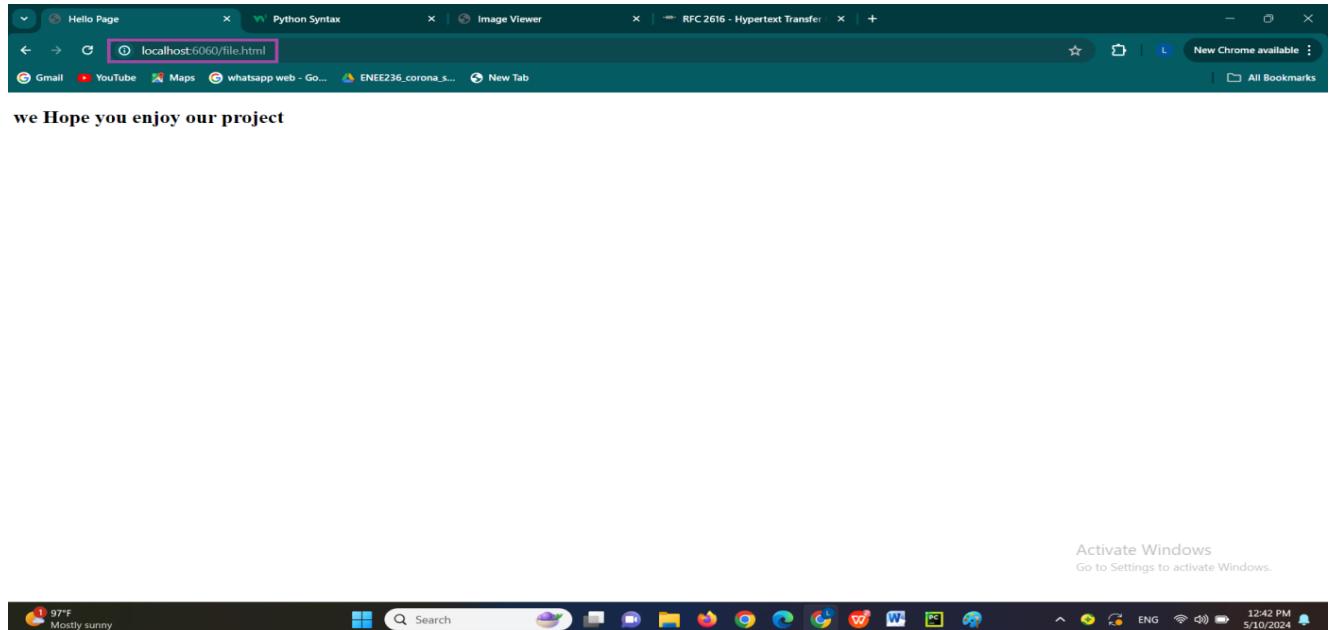
Activate Windows
Go to Settings to activate Windows.

1networkProject > main_ar.html

97°F Mostly sunny

- 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. Make it general (not only for specific filename)

Figure 18:The HTTP request '/file.html' redirects the client to a webpage designed by html.



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

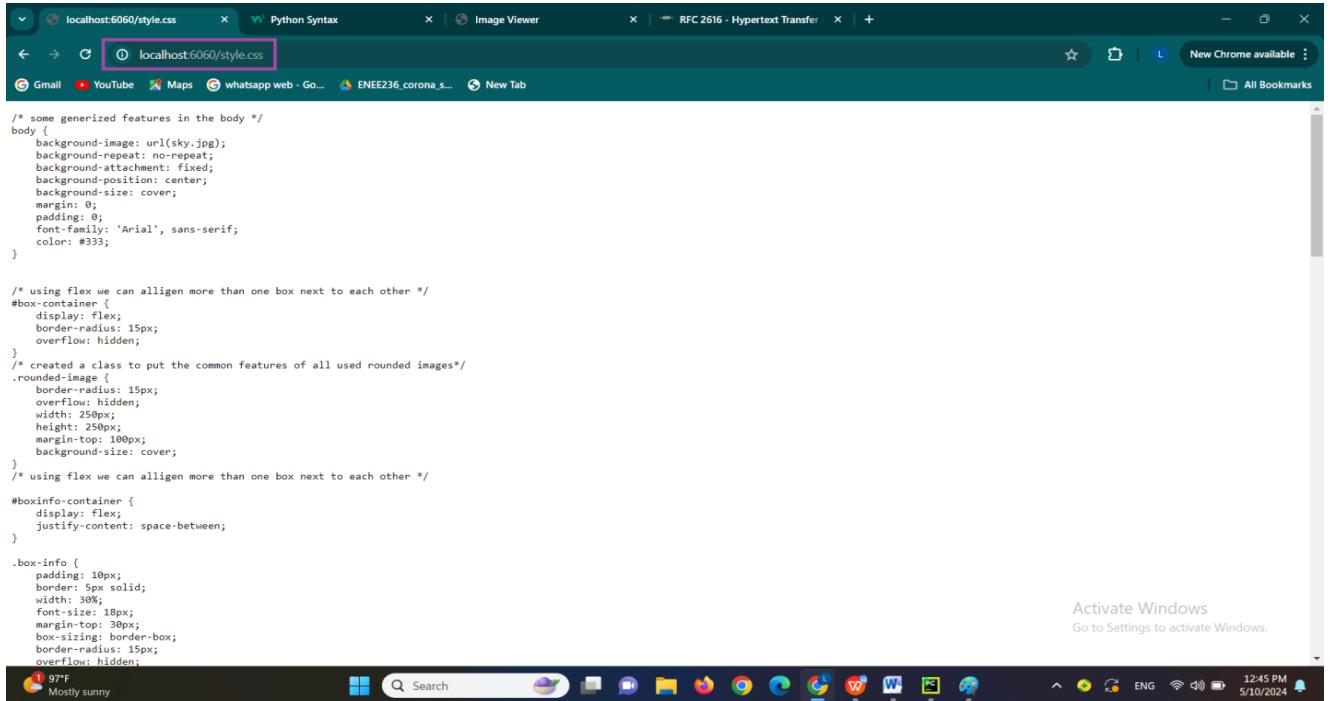
The screenshot shows a terminal window with a dark background. It displays an incoming HTTP request for "/file.html" via GET protocol. The request includes headers such as Host, Connection, User-Agent, Accept, Sec-Fetch-Site, Sec-Fetch-Mode, Sec-Fetch-User, and Sec-Fetch-Dest. The terminal also shows the current working directory as "TnetworkProject" and the file "main_ar.html". At the bottom, there is a Windows taskbar with pinned icons and system status indicators.

```

Received HTTP request: GET /file.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: ?
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
  
```

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. Make it general (not only for specific filename)

Figure 19:The HTTP request '/style.css' redirects the client to the css file stored on the computer (The same path as the web server).



The screenshot shows a Windows desktop environment. In the center is a Microsoft Edge browser window displaying a CSS file's content. The code includes various CSS rules for body, box-container, rounded-image, and boxinfo-container elements. To the right of the browser, the Windows taskbar is visible, showing icons for various applications like File Explorer, OneDrive, and a weather widget indicating 97°F and 'Mostly sunny'. At the bottom right, the system tray shows the date (5/10/2024), time (12:45 PM), battery level, and network status.

```

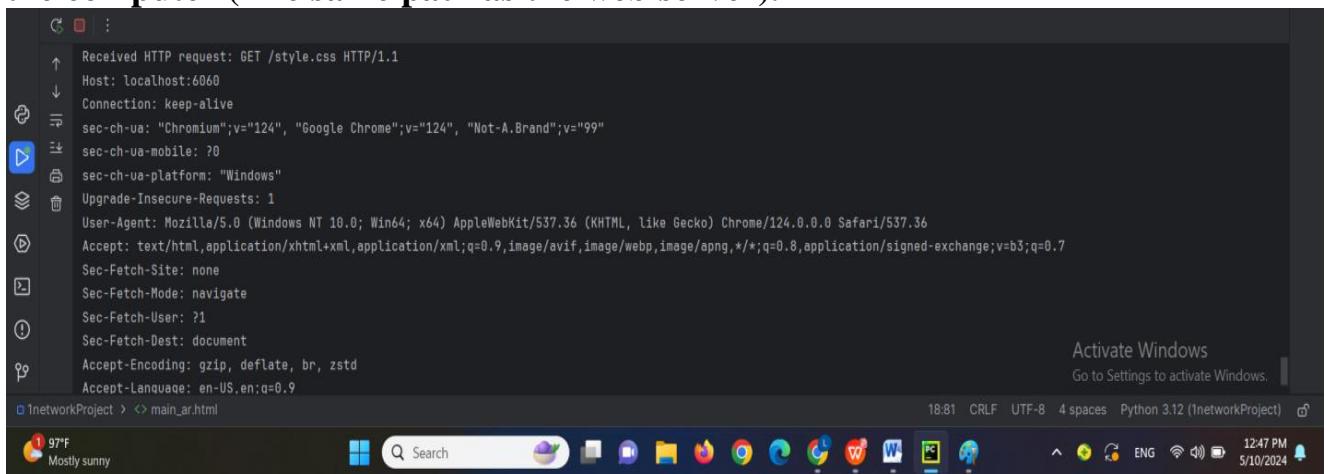
/*
 * some generalized Features in the body */
body {
    background-image: url(sky.jpg);
    background-repeat: no-repeat;
    background-attachment: fixed;
    background-position: center;
    background-size: cover;
    margin: 0;
    padding: 0;
    font-family: 'Arial', sans-serif;
    color: #333;
}

/* using Flex we can align more than one box next to each other */
.box-container {
    display: flex;
    border-radius: 15px;
    overflow: hidden;
}
/* created a class to put the common features of all used rounded images*/
.roundedImage {
    border-radius: 15px;
    overflow: hidden;
    width: 250px;
    height: 250px;
    margin-top: 100px;
    background-size: cover;
}
/* using flex we can align more than one box next to each other */
#boxinfo-container {
    display: flex;
    justify-content: space-between;
}

.box-info {
    padding: 10px;
    border: 5px solid;
    width: 300px;
    font-size: 18px;
    margin-top: 30px;
    box-sizing: border-box;
    border-radius: 15px;
    overflow: hidden;
}

```

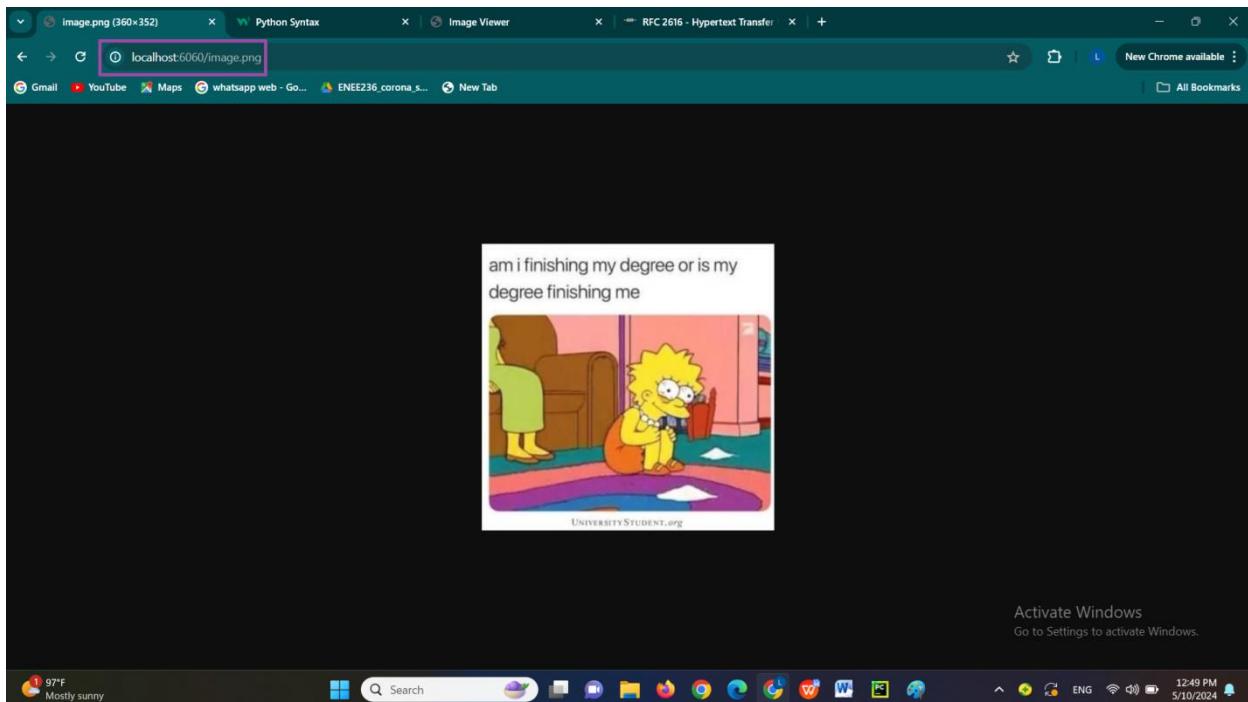
The HTTP request '/style.css' redirects the client to the css file stored on the computer (The same path as the web server).



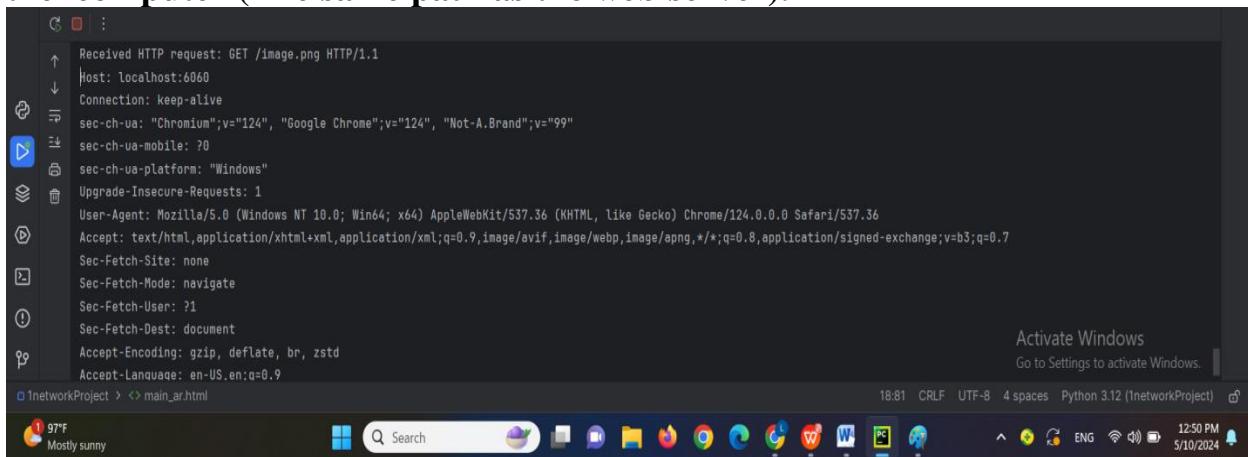
The screenshot shows a Windows desktop environment. In the center is a terminal window from the Windows Start menu showing an expanded view of an incoming HTTP request. The request details include the method (GET), URL (/style.css), version (HTTP/1.1), host (localhost:6060), connection type (keep-alive), user agent (Chromium; v=124, Google Chrome; v=124, Not-A-Brand; v=99), mobile indicator (sec-ch-ua-mobile: 0), platform (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 (?), sec-fetch-dest (document), accept-encoding (gzip, deflate, br, zstd), and accept-language (en-US,en;q=0.9). To the right of the terminal, the Windows taskbar is visible, showing icons for various applications like File Explorer, OneDrive, and a weather widget indicating 97°F and 'Mostly sunny'. At the bottom right, the system tray shows the date (5/10/2024), time (12:47 PM), battery level, and network status.

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. Make it general (not only for specific filename)

Figure 20: The HTTP request '/image.png' redirects the client to a png image stored on the computer (The same path as the web server).

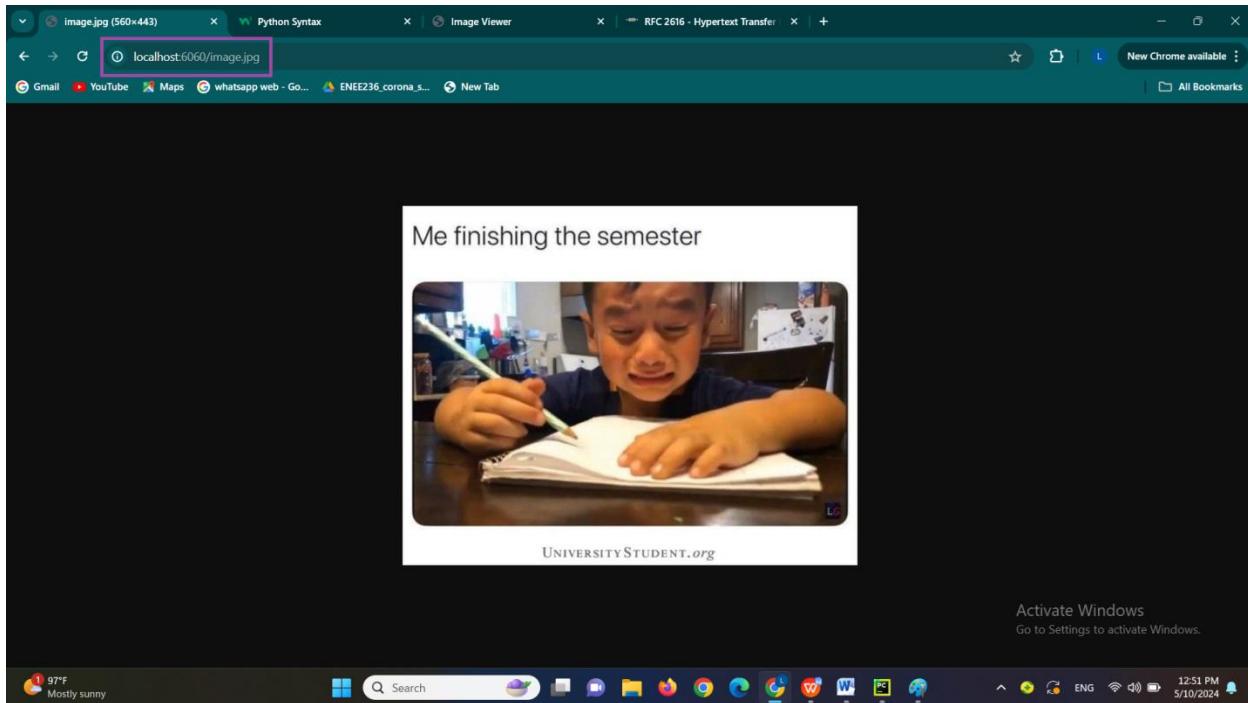


The HTTP request '/image.png' redirects the client to a png image stored on the computer (The same path as the web server).

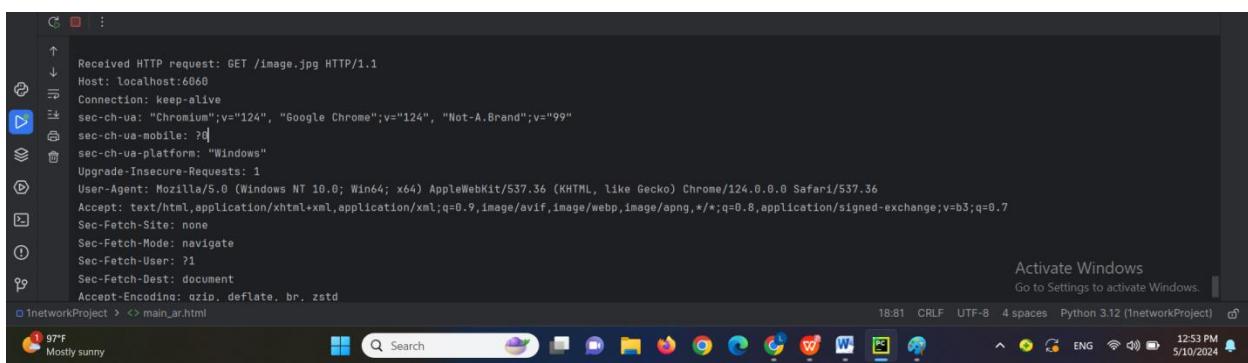


6. if the request is a **.jpg** then the server should send the jpg image with Content-Type: image/jpeg. You can use any image. Make it general (not only for specific filename)

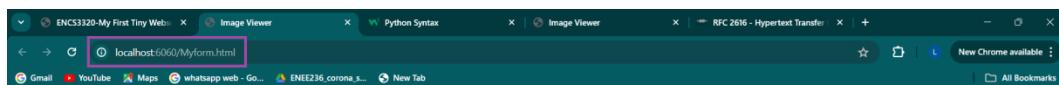
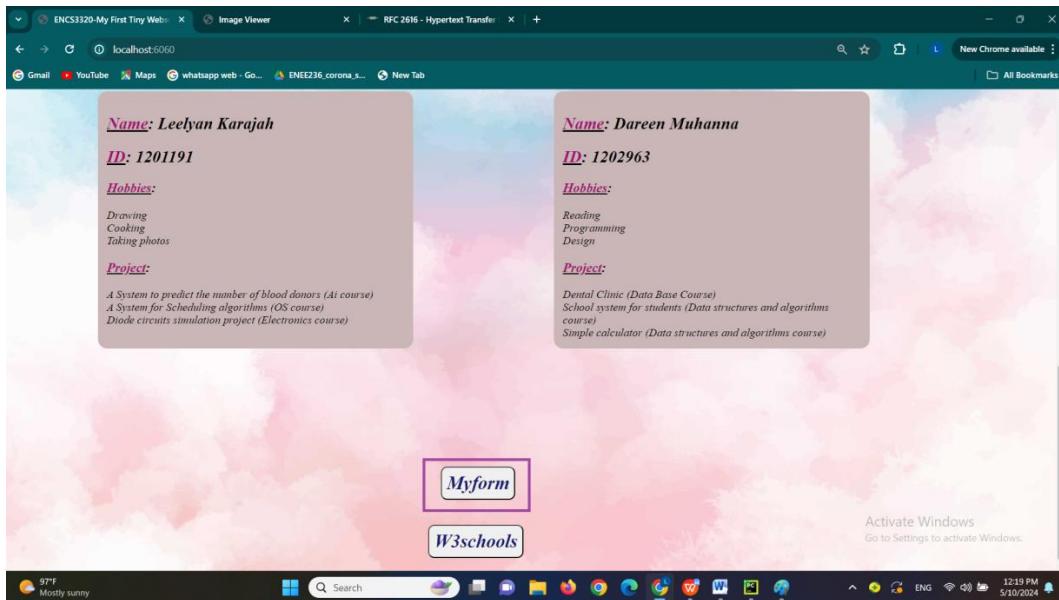
Figure 21: The HTTP request '/image.jpg' redirects the client to a jpg image stored on the computer (The same path as the web server).



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

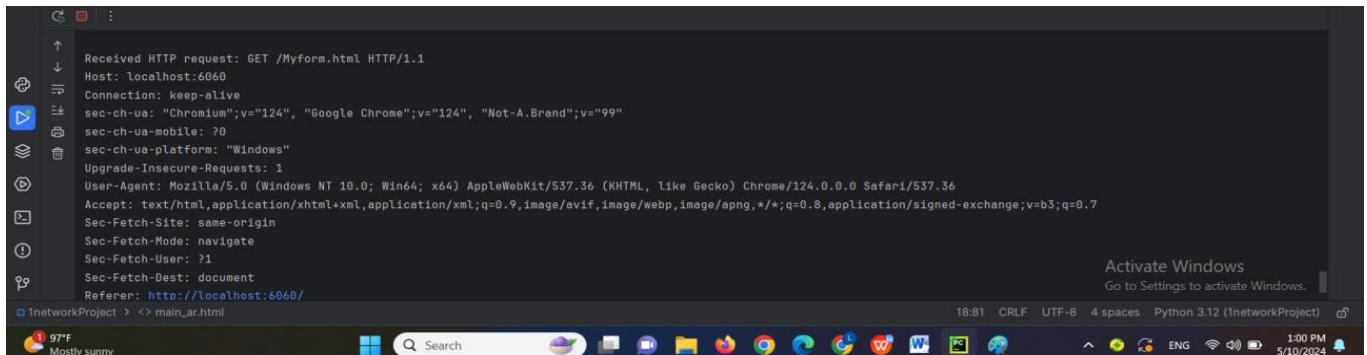


7. Use myform.html to get image by typing the name of the image in a box.



The HTTP request '/Myform.html'

Figure 22:The HTTP request '/Myform.html'

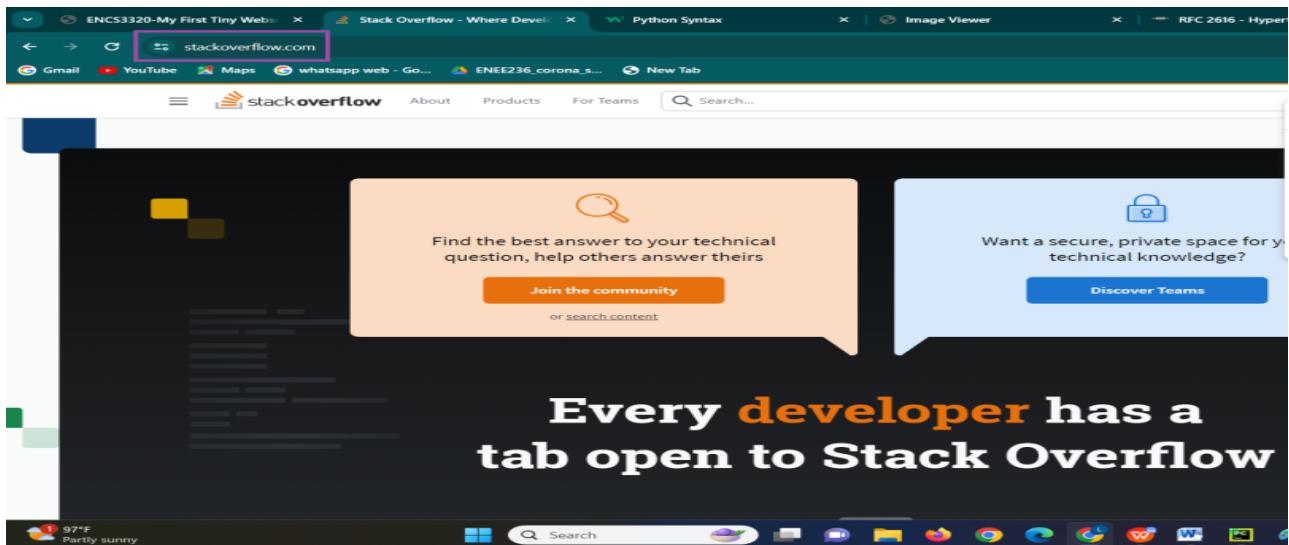


```
Received HTTP request: GET /Myform.html HTTP/1.1
Host: localhost:6060
Connection: keep-alive
sec-ch-ua: "Chromium";v="124", "Google Chrome";v="124", "Not-A.Brand";v="99"
sec-ch-ua-mobile: ?
sec-ch-ua-platform: "Windows"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; 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=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?
Sec-Fetch-Dest: document
Referer: http://localhost:6060/
1networkProject > <> main_ar.html
```

8. Use the status code 307 Temporary Redirect to redirect the following

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

Figure 23:stackoverflow.com website



```

Received HTTP request: 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/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: en-US,en;q=0.9

```

Activate Windows
Go to Settings to activate Windows.

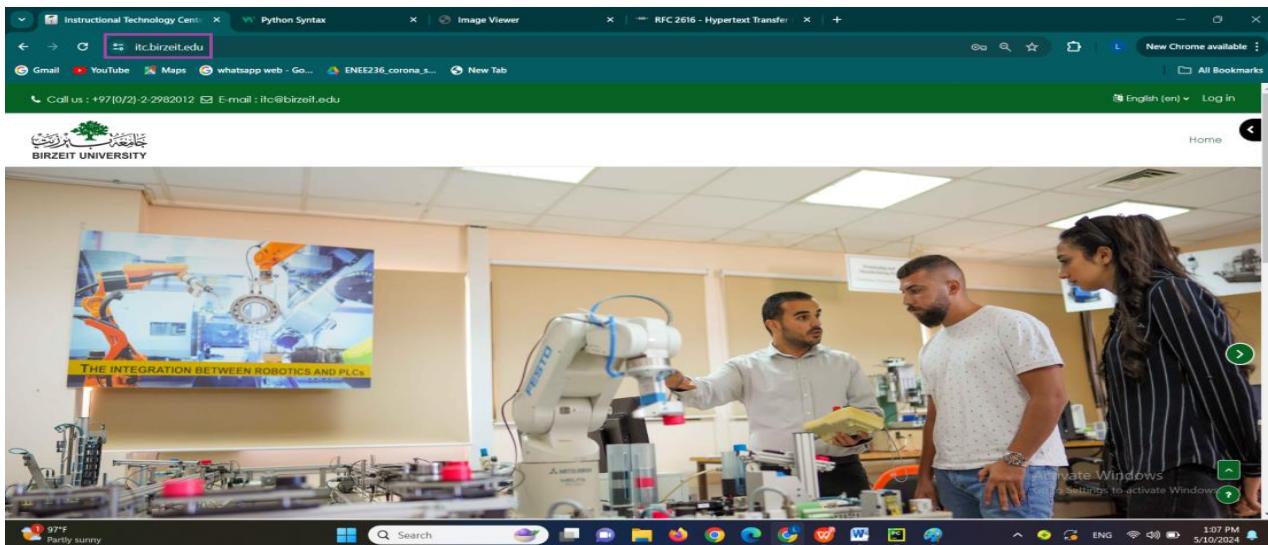
18:81 CRLF UTF-8 4 spaces Python 3.12 (InetworkProject)

Upcoming Earnings Search

106 PM 5/10/2024

- ❖ If the request is **/itc** then redirect to **itc.birzeit.edu** website

Figure 24:itc.birzeit.edu website



```

Received HTTP request: 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,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: en-US,en;q=0.9

```

Activate Windows
Go to Settings to activate Windows.

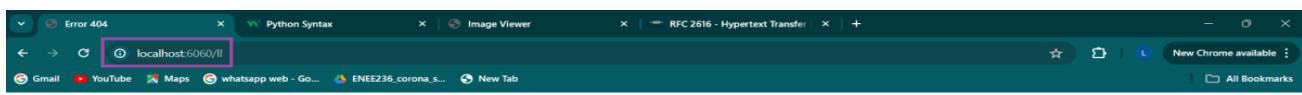
18:81 CRLF UTF-8 4 spaces Python 3.12 (InetworkProject)

97°F Partly sunny Search

108 PM 5/10/2024

9. 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)

1. "HTTP/1.1 404 Not Found" in the response status
2. "Error 404" in the title
3. "**The file is not found**" in the body in **red**
4. Your names and IDs in **Bold**
5. The IP and port number of the client



Error HTTP/1.1 404 - The requested resource was not found

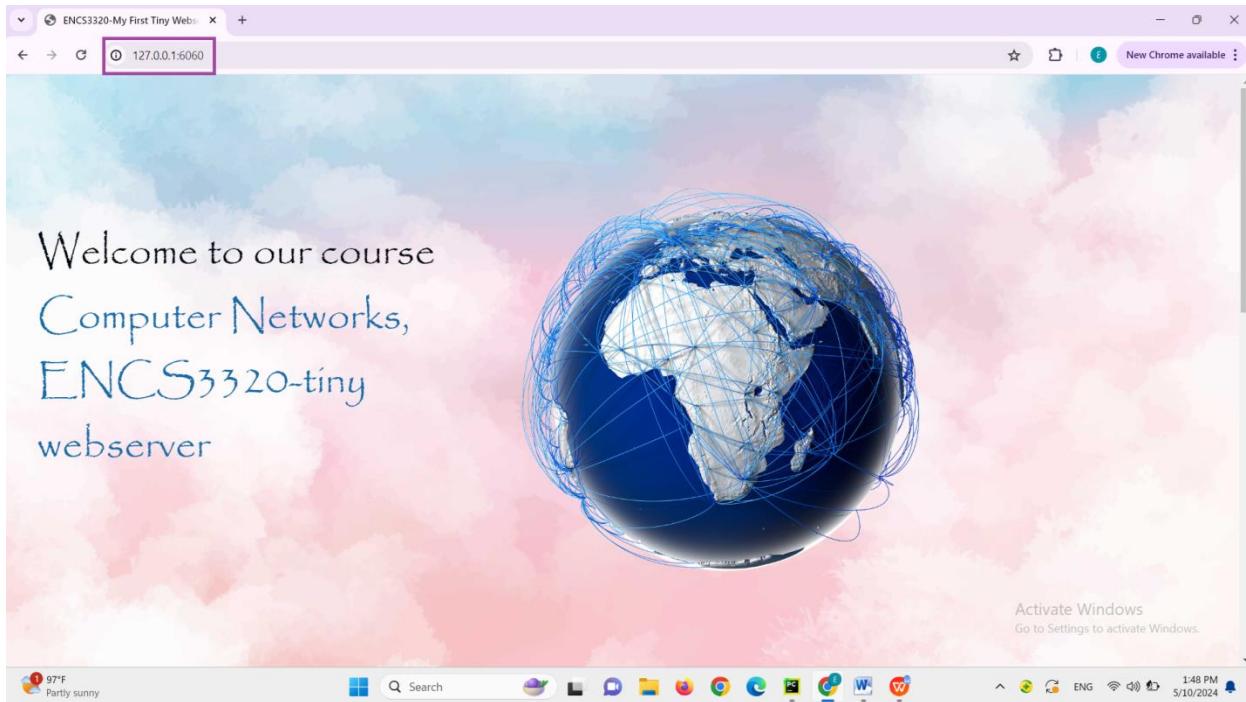
The file is not found

Leeyan karajah 1201191
Dareen muhanna 1202963

Client IP address and Port:127.0.0.1:63694

A screenshot of a terminal window titled 'Received HTTP request: GET /ll HTTP/1.1'. The window displays a detailed list of HTTP headers from a client. The headers include 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, and Accept-Language: en-US,en;q=0.9. The terminal also shows the path '1networkProject > <> main_ar.html'. The system tray at the bottom shows the same battery and weather information as the previous screenshot.

10. Test the project from a browser on the **same computer** and from a **different computer or phone**.

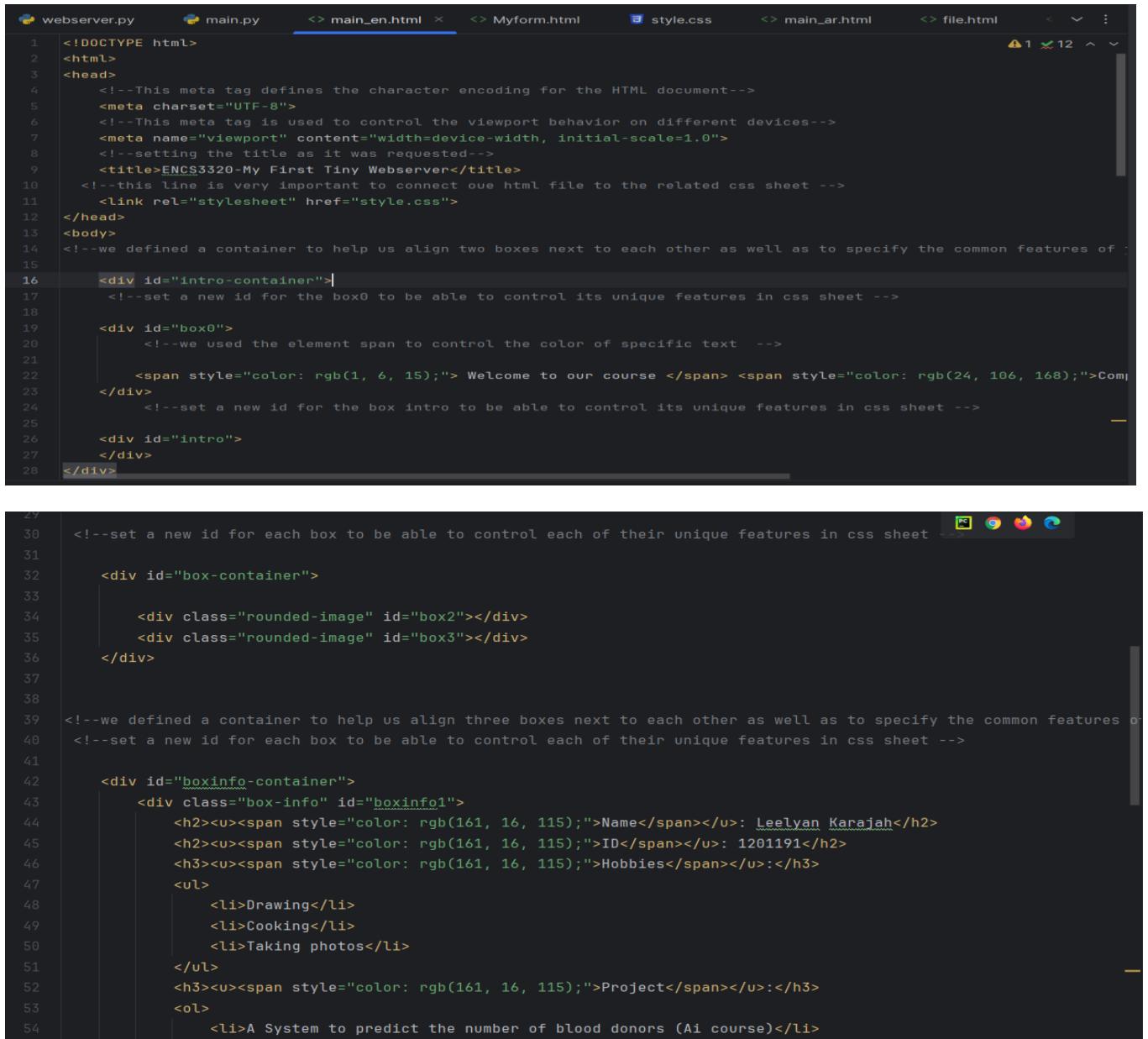


Codes

English version

- HTML (main_en.html)

Figure 25:HTML english version



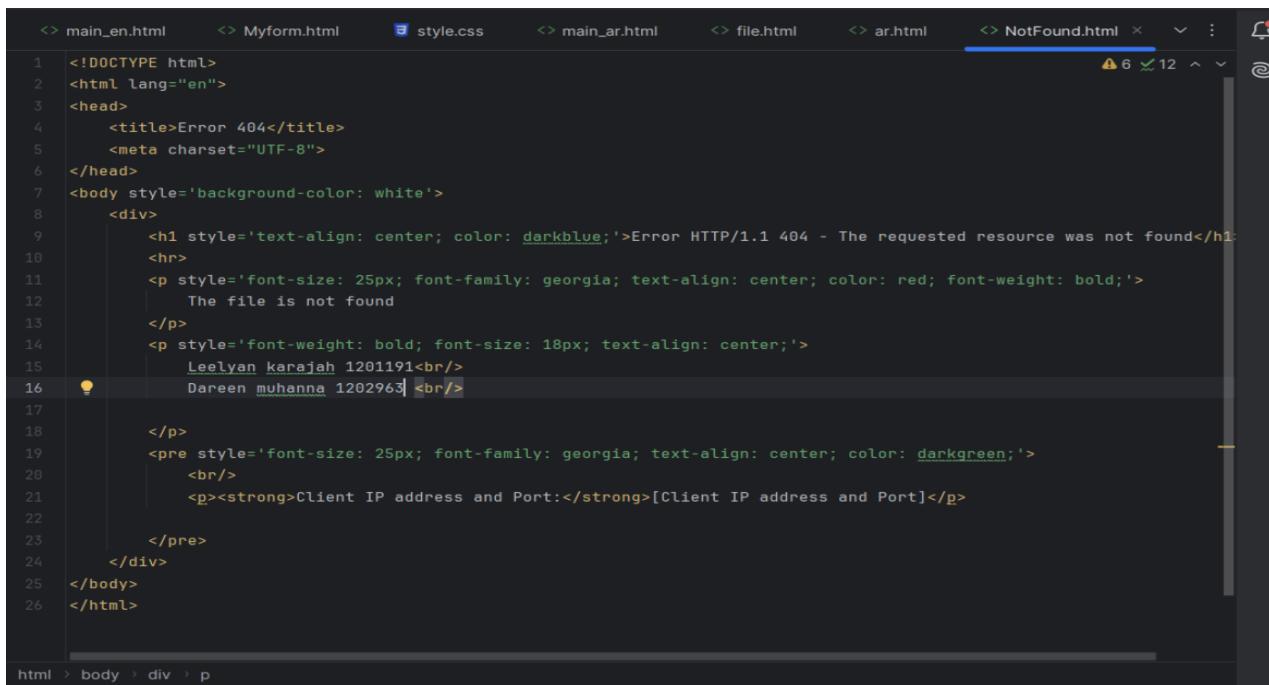
```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <!--This meta tag defines the character encoding for the HTML document--&gt;
5     &lt;meta charset="UTF-8"&gt;
6     <!--This meta tag is used to control the viewport behavior on different devices--&gt;
7     &lt;meta name="viewport" content="width=device-width, initial-scale=1.0"&gt;
8     <!--setting the title as it was requested--&gt;
9     &lt;title&gt;ENCS3320-My First Tiny Webserver&lt;/title&gt;
10    &lt!--this line is very important to connect our html file to the related css sheet --&gt;
11    &lt;link rel="stylesheet" href="style.css"&gt;
12 &lt;/head&gt;
13 &lt;body&gt;
14     <!--we defined a container to help us align two boxes next to each other as well as to specify the common features of both boxes--&gt;
15
16     &lt;div id="intro-container"&gt;
17         &lt!--set a new id for the box0 to be able to control its unique features in css sheet --&gt;
18
19         &lt;div id="box0"&gt;
20             &lt!--we used the element span to control the color of specific text --&gt;
21             &lt;span style="color: rgb(1, 6, 15);&gt; Welcome to our course &lt;/span&gt; &lt;span style="color: rgb(24, 106, 168);&gt;Com-
22             &lt;/span&gt;
23             &lt;!--set a new id for the box intro to be able to control its unique features in css sheet --&gt;
24
25         &lt;div id="intro"&gt;
26             &lt;/div&gt;
27     &lt;/div&gt;
28
29
30     &lt!--set a new id for each box to be able to control each of their unique features in css sheet --&gt;
31
32     &lt;div id="box-container"&gt;
33
34         &lt;div class="rounded-image" id="box2"&gt;&lt;/div&gt;
35         &lt;div class="rounded-image" id="box3"&gt;&lt;/div&gt;
36     &lt;/div&gt;
37
38
39     &lt!--we defined a container to help us align three boxes next to each other as well as to specify the common features of all three boxes--&gt;
40     &lt;!--set a new id for each box to be able to control each of their unique features in css sheet --&gt;
41
42     &lt;div id="boxinfo-container"&gt;
43         &lt;div class="box-info" id="boxinfo1"&gt;
44             &lt;h2&gt;&lt;u&gt;&lt;span style="color: rgb(161, 16, 115);&gt;Name&lt;/span&gt;&lt;/u&gt;: Lealyan Karajah&lt;/h2&gt;
45             &lt;h2&gt;&lt;u&gt;&lt;span style="color: rgb(161, 16, 115);&gt;ID&lt;/span&gt;&lt;/u&gt;: 1201191&lt;/h2&gt;
46             &lt;h3&gt;&lt;u&gt;&lt;span style="color: rgb(161, 16, 115);&gt;Hobbies&lt;/span&gt;&lt;/u&gt;:&lt;/h3&gt;
47             &lt;ul&gt;
48                 &lt;li&gt;Drawing&lt;/li&gt;
49                 &lt;li&gt;Cooking&lt;/li&gt;
50                 &lt;li&gt;Taking photos&lt;/li&gt;
51             &lt;/ul&gt;
52             &lt;h3&gt;&lt;u&gt;&lt;span style="color: rgb(161, 16, 115);&gt;Project&lt;/span&gt;&lt;/u&gt;:&lt;/h3&gt;
53             &lt;ol&gt;
54                 &lt;li&gt;A System to predict the number of blood donors (Ai course)&lt;/li&gt;</pre>
```

```
55         <li>A System for Scheduling algorithms (OS course)</li>
56         <li>Diode circuits simulation project (Electronics course)</li>
57     </ol>
58 </div>
59 <div class="box-info" id="boxinfo3">
60     <h2><u><span style="color: rgb(161, 16, 115);>Name</span></u>: Daireen Muhanna</h2>
61     <h2><u><span style="color: rgb(161, 16, 115);>ID</span></u>: 1202963</h2>
62     <h3><u><span style="color: rgb(161, 16, 115);>Hobbies</span></u>:</h3>
63     <ul>
64         <li>Reading</li>
65         <li>Programming</li>
66         <li>Design</li>
67     </ul>
68     <h3><u><span style="color: rgb(161, 16, 115);>Project</span></u>:</h3>
69     <ol>
70         <li>Dental Clinic (Data Base Course)</li>
71         <li>School system for students (Data structures and algorithms course)</li>
72         <li>Simple calculator (Data structures and algorithms course)</li>
73     </ol>
74 </div>
75 </div>
76
77
```

```
78
79     <h2></h2>
80     <!-- the element for hyper links -->
81     <a href="/Myform.html" target="_blank">
82         <button id = "button1"> Myform</button>
83     </a>
84 </h2>
85 <h2>
86     <a href="https://www.w3schools.com/python/python_syntax.asp" target="_blank">
87         <button id = "button2"> W3schools</button>
88     </a>
89 </h2>
90 </body>
91 </html>
```

html > body > div#intro-container

- HTML (notfound.html)



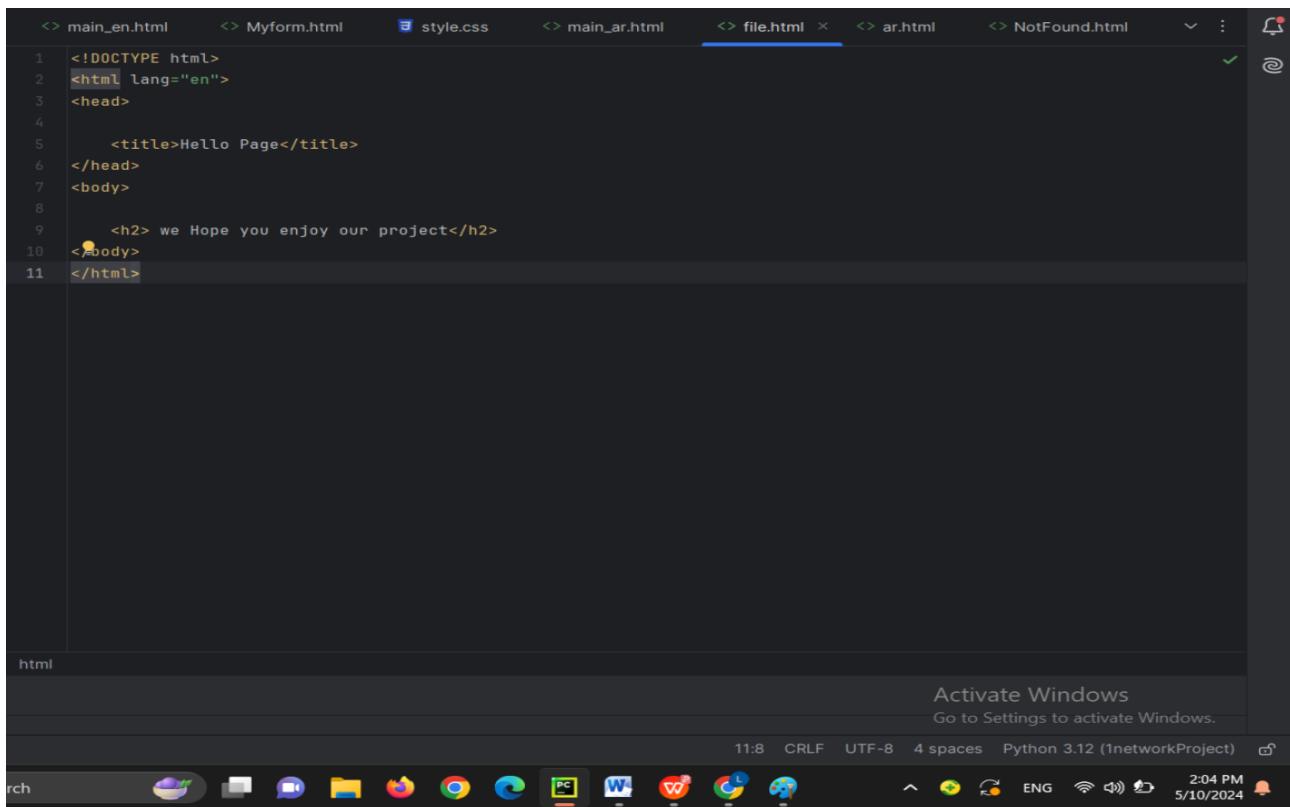
```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <title>Error 404</title>
5   <meta charset="UTF-8">
6 </head>
7 <body style='background-color: white'>
8   <div>
9     <h1 style='text-align: center; color: darkblue;'>Error HTTP/1.1 404 - The requested resource was not found</h1>
10    <hr>
11    <p style='font-size: 25px; font-family: georgia; text-align: center; color: red; font-weight: bold;'>
12      The file is not found
13    </p>
14    <p style='font-weight: bold; font-size: 18px; text-align: center;'>
15      Leelyan karajah 1201191<br/>
16      Dareen muhanna 1202963<br/>
17    </p>
18    <pre style='font-size: 25px; font-family: georgia; text-align: center; color: darkgreen;'>
19      <br/>
20      <p><strong>Client IP address and Port:</strong>[Client IP address and Port]</p>
21    </pre>
22  </div>
23 </body>
24 </html>

```

html > body > div > p

- HTML (file.html)



```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <title>Hello Page</title>
5 </head>
6 <body>
7   <h2> we Hope you enjoy our project</h2>
8 </body>
9 </html>

```

Activate Windows
Go to Settings to activate Windows.

11:8 CRLF UTF-8 4 spaces Python 3.12 (1networkProject) 2:04 PM 5/10/2024

- o style.css

Figure 26:cssenglish version

```

1  /* some generalized features in the body */
2  body {
3      background-image: url(sky.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 alligen more than one box next to each other */
16 #box-container {
17     display: flex;
18     border-radius: 15px;
19     overflow: hidden;
20 }
21
22 /* created a class to put the common features of all used rounded images*/
23 .rounded-image {
24     border-radius: 15px;
25     overflow: hidden;
26     width: 250px;
27     height: 250px;
28     margin-top: 100px;
29     background-size: cover;
30 }
```

Activate Windows
Go to Settings to activate Windows.

49:1 CRLF UTF-8 4 spaces Python 3.12 (1networkProject) 2:09 PM 5/10/2024

```

30 /* using flex we can alligen more than one box next to each other */
31
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
48 /* we used some features to make the web page look nicer and more organized */
49
50 /* instead of repeating same features we used the three classes together */
51 #box2, #box3 {
52     border: 5px solid;
53     width: 250px;
54     height: 250px;
55     font-size: 22px;
56     margin-top: 100px;
57     margin-bottom: 30px;
58     border-radius: 15px;
59 }
```

Activate Windows
Go to Settings to activate Windows.

49:1 CRLF UTF-8 4 spaces Python 3.12 (1networkProject) 2:10 PM 5/10/2024

```
<> main_en.html    <> Myform.html    <> style.css    <> main_ar.html    <> file.html    <> ar.html    <> NotFound.html    <> 36    <> 36

58     border-radius: 15px;
59     overflow: hidden;
60 }
61
62 .highlight-text {
63     color: rgb(201, 182, 182);
64 }
65 /* specify the background image and margins for each box */
66
67
68 #box2 {
69     margin-left: 250px;
70     background-image: url(leelyan.jpg);
71     background-size: cover;
72 }
73
74 #box3 {
75     margin-right: 10px;
76     margin-left: 500px;
77     background-image: url(Dareen.jpg);
78     background-size: cover;
79 }
80 /* insted of repeating same features we used the three classes togather */
81
82 #boxinfo1, #boxinfo3 {
83     font-family: "Times New Roman", Times, serif;
84     padding: 10px;
85     border: 5px solid;
86     width: 30%;
87     float: right;
88 }

Activate Windows
Go to Settings to activate Windows.

49:1 CRLF UTF-8 4 spaces Python 3.12 (1networkProject) ⚡
ch 🍇 📄 💬 📁 🌐 🎯 📈 📤 🎯 ENG 🔔 2:11 PM 5/10/2024
```

```
<> main_en.html <> Myform.html <> style.css <> main_ar.html <> file.html <> ar.html <> NotFound.html <> : 36 ^ v @
115 #boxinfo3 {
116     font-style: italic;
117     border-color: rgb(201, 182, 182);
118     background-color: rgb(201, 182, 182);
119     margin-right: 300px;
120     color: black;
121 }
122 /* instead of repeating same features we used the main class button */
123
124 button {
125     font-size: 30px;
126     margin-top: 20px;
127     font-style: italic;
128     font-weight: bold;
129     font-family: "Times New Roman", Times, serif;
130     padding: 8px;
131     border-radius: 10px; /* Adjust the border-radius as needed */
132     color: rgb(25,25,112);
133     align-self: center;
134 }
135
136 #button1{
137     margin-left: 700px;
138 }
139
140 #button2{
141     margin-left: 680px;
142 }
```

Activate Windows
Go to Settings to activate Windows.

49:1 CRLF UTF-8 4 spaces Python 3.12 (1networkProject) ⌂

```
<> main_en.html <> Myform.html <> style.css <> main_ar.html <> file.html <> ar.html <> NotFound.html <> : 36 ^ v @
135 }
136 #button1{
137     margin-left: 700px;
138 }
139
140 #button2{
141     margin-left: 680px;
142 }
143
144 #box0{
145     font-family:'fantasy',Papyrus;
146     margin-top: 150px;
147     margin-left: 10px;
148     margin-right: 20px;
149     border : 0px ;
150     width : 500px ;
151     height : 200px;
152     font-size: 50px;
153
154     padding : 30px;
155
156     color:white;
157 }
158
159
160 #intro{
161     background-image: url(network.png);
162     background-size: cover;
163     width: 550px;
```

Activate Windows
Go to Settings to activate Windows.

49:1 CRLF UTF-8 4 spaces Python 3.12 (1networkProject) ⌂



```
<> main_en.html <> Myform.html <> style.css <> main_ar.html <> file.html <> ar.html <> NotFound.html <> : 36 <> @

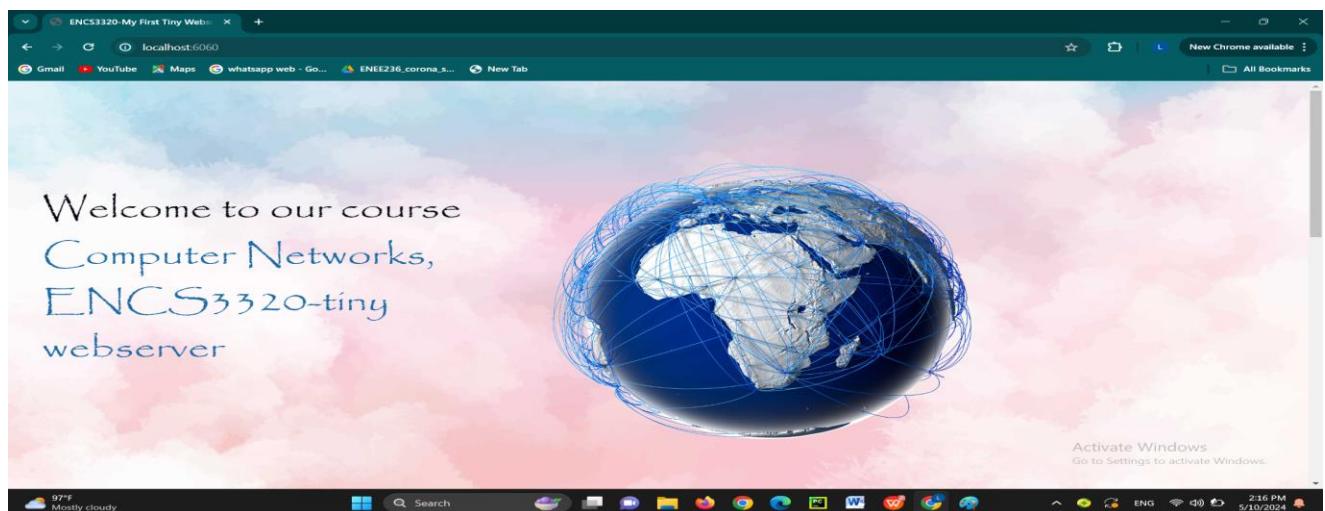
149 border : 0px ;
150 width : 500px ;
151 height : 200px;
152 font-size: 50px;
153
154 padding : 30px;
155
156 color:white;
157
158 }
159
160 #intro{
161 background-image: url(network.png);
162 background-size: cover;
163 width: 550px;
164 height: 550px;
165 margin-left: 10px;
166 margin-right: 250px;
167 margin-top: 100px;
168 }
169 #intro-container{
170 display: flex;
171 }
172 }

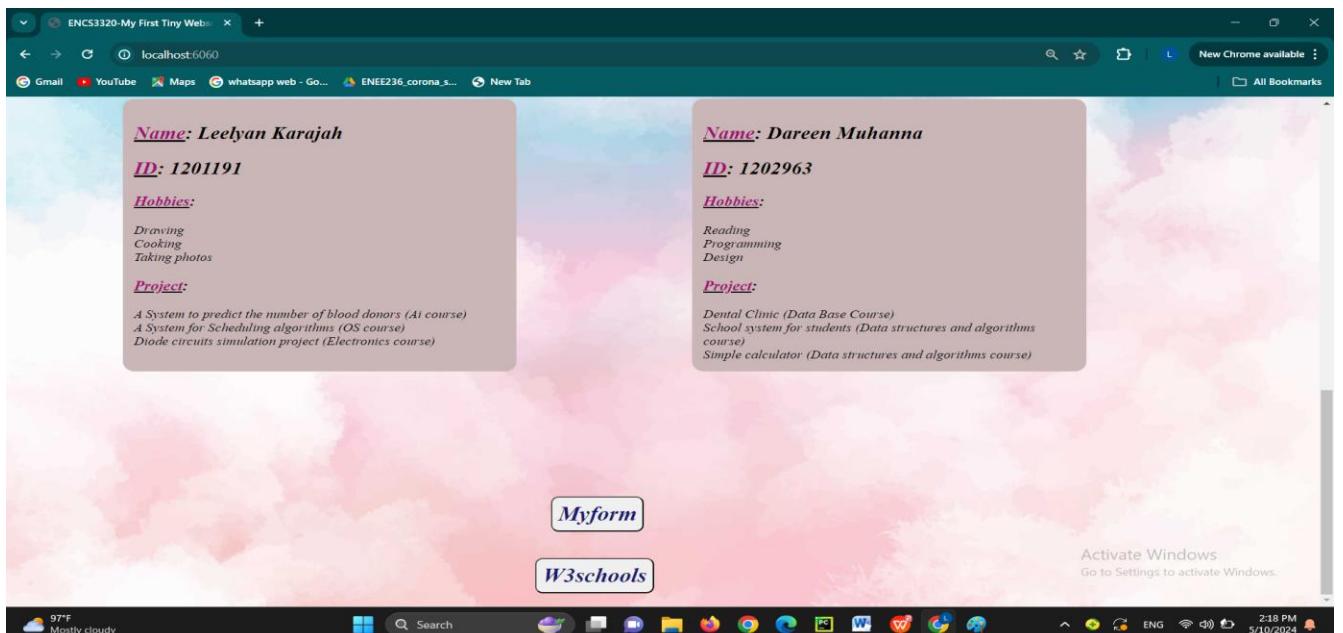
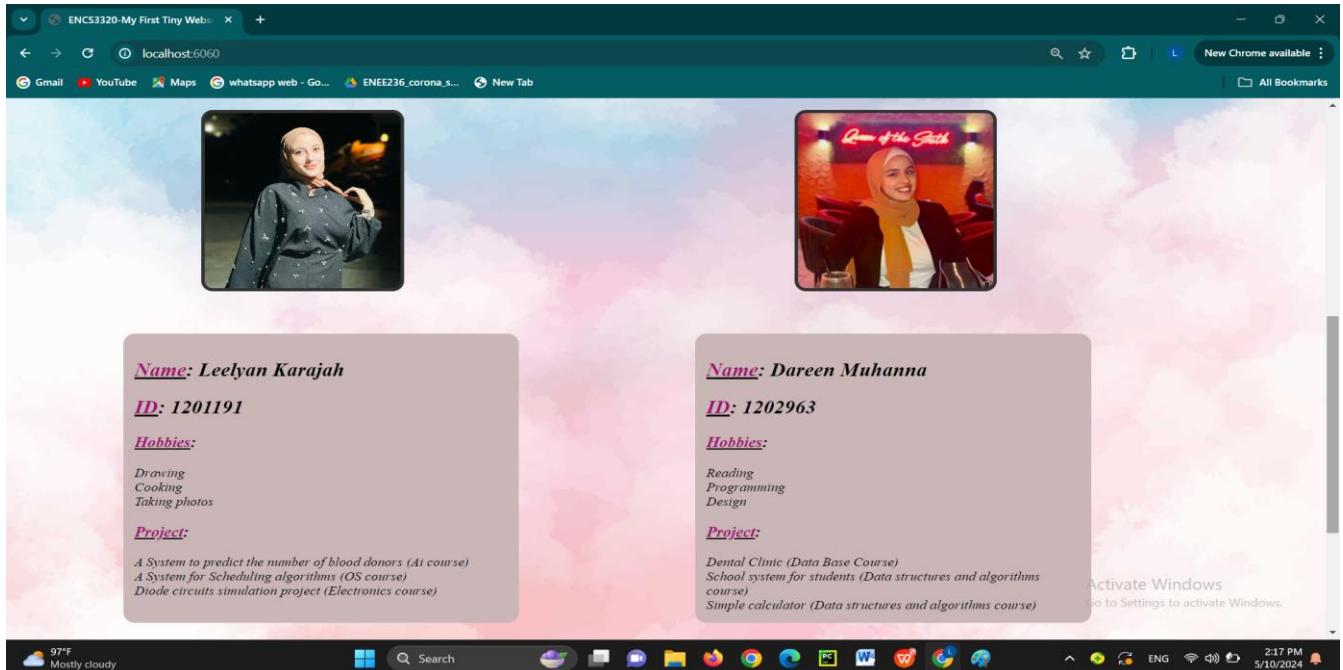
Activate Windows
Go to Settings to activate Windows.

49:1 CRLF UTF-8 4 spaces Python 3.12 (1networkProject) 2:14 PM 5/10/2024
```

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

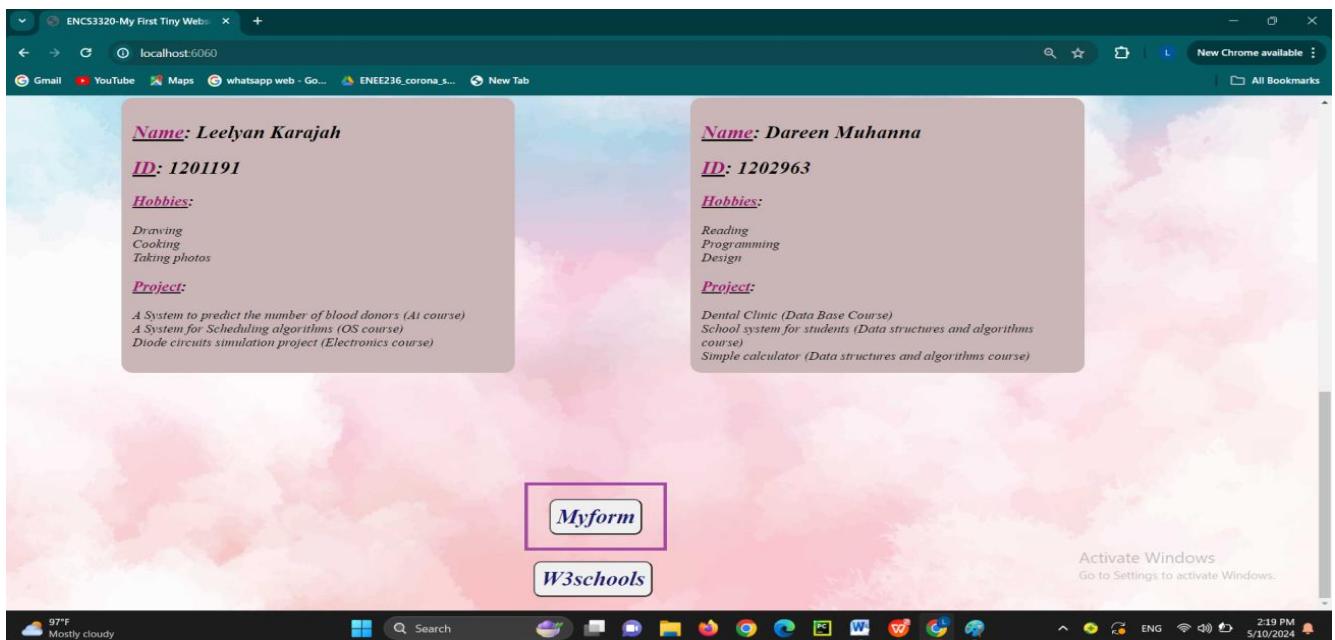
- o Web server



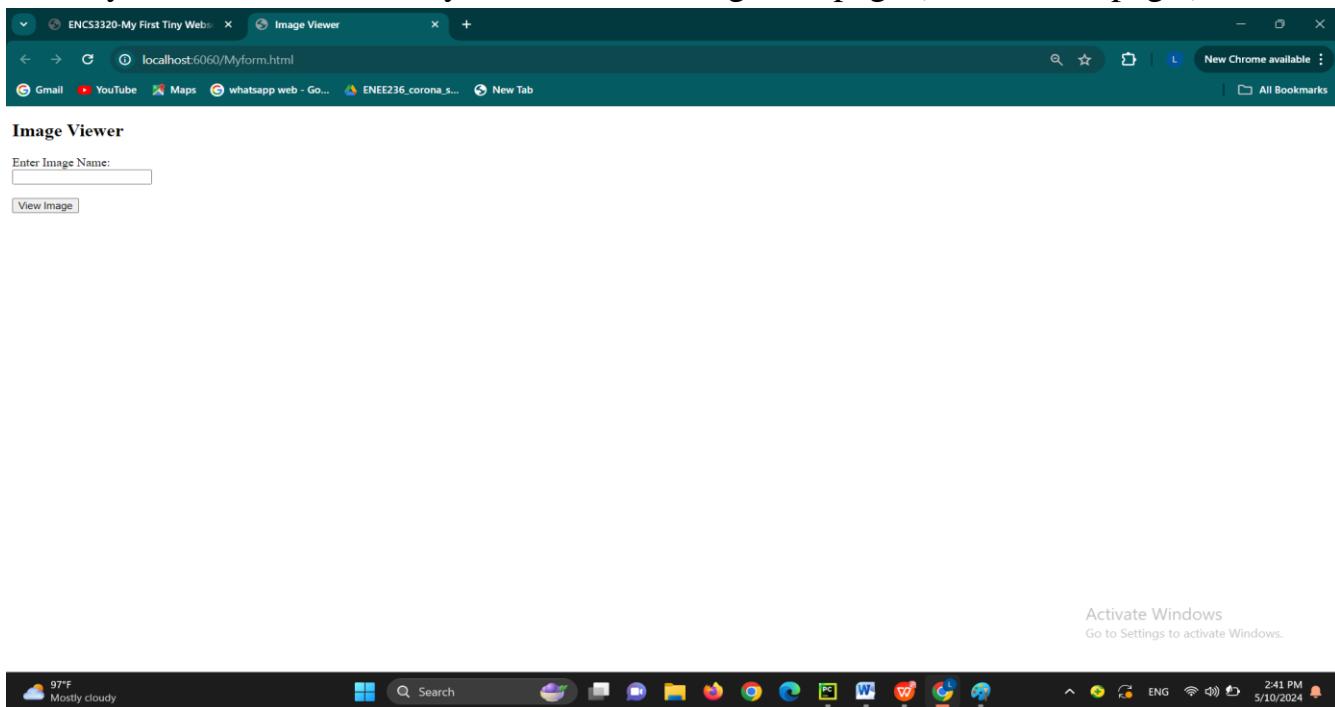


In the figures above we made sure to follow the instructions as given in the project description by setting the title of the page, the welcome phrase with the right colors, the group member names, IDs, Hobbies and preformed projects.

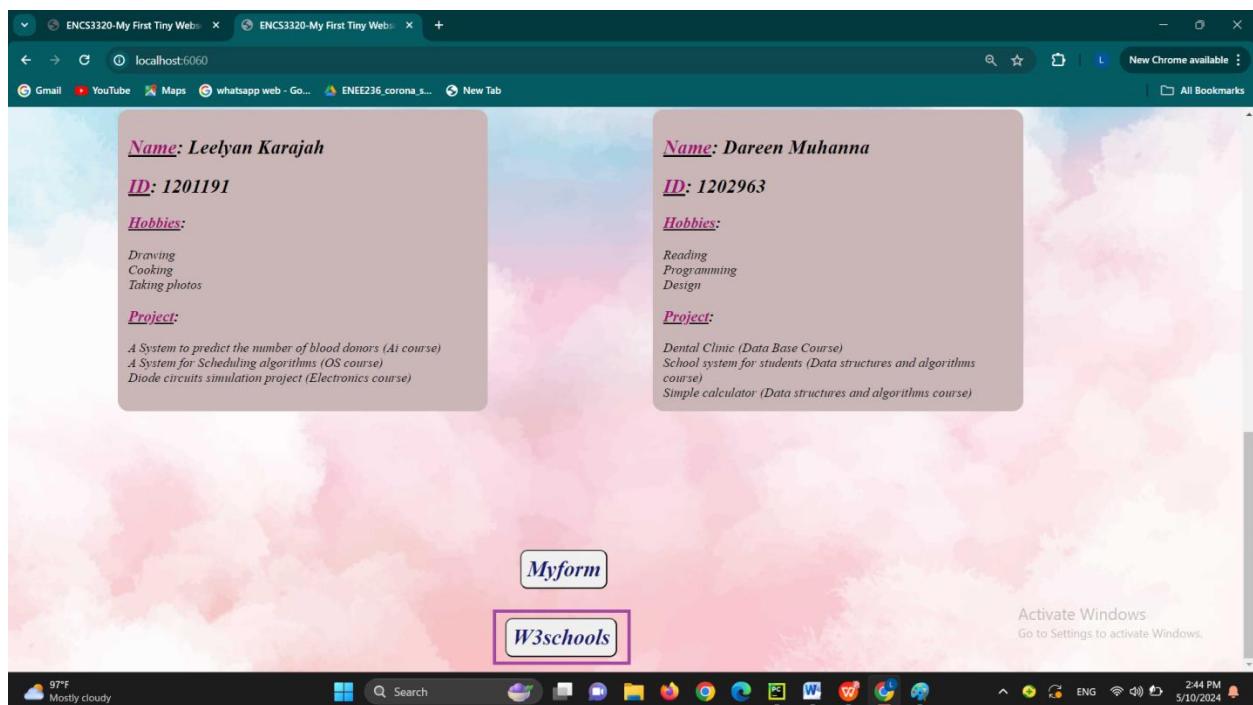
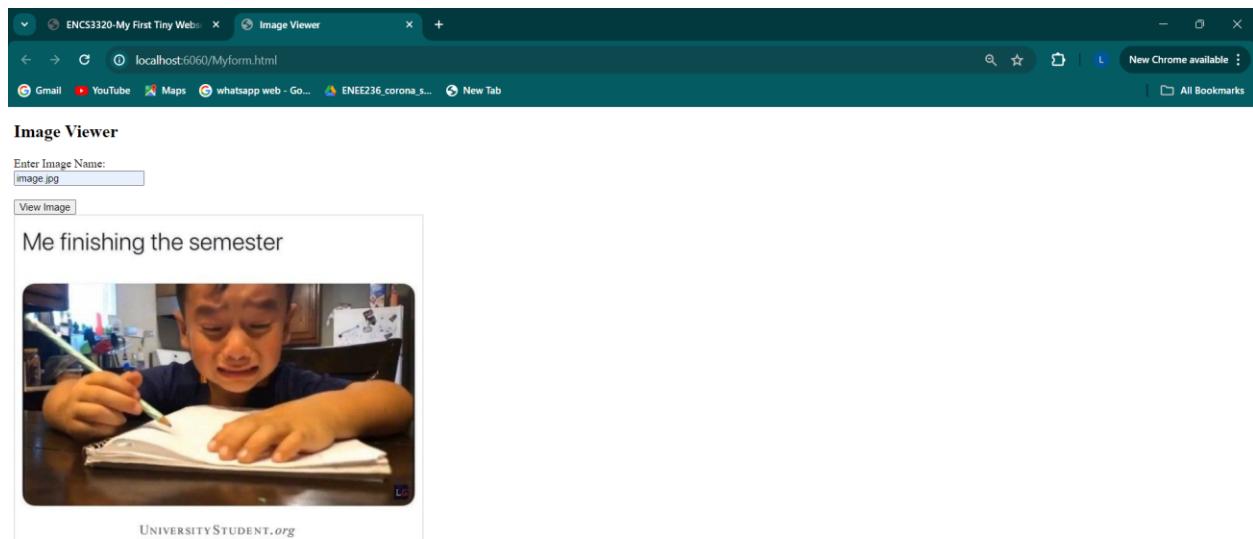
We also used CSS to enhance the frontend part of the project in every HTML file. **In the following figures, we will show the pages which our main page redirect you (by using hyper links).**



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



In the box you can but any image name ant it will appears .



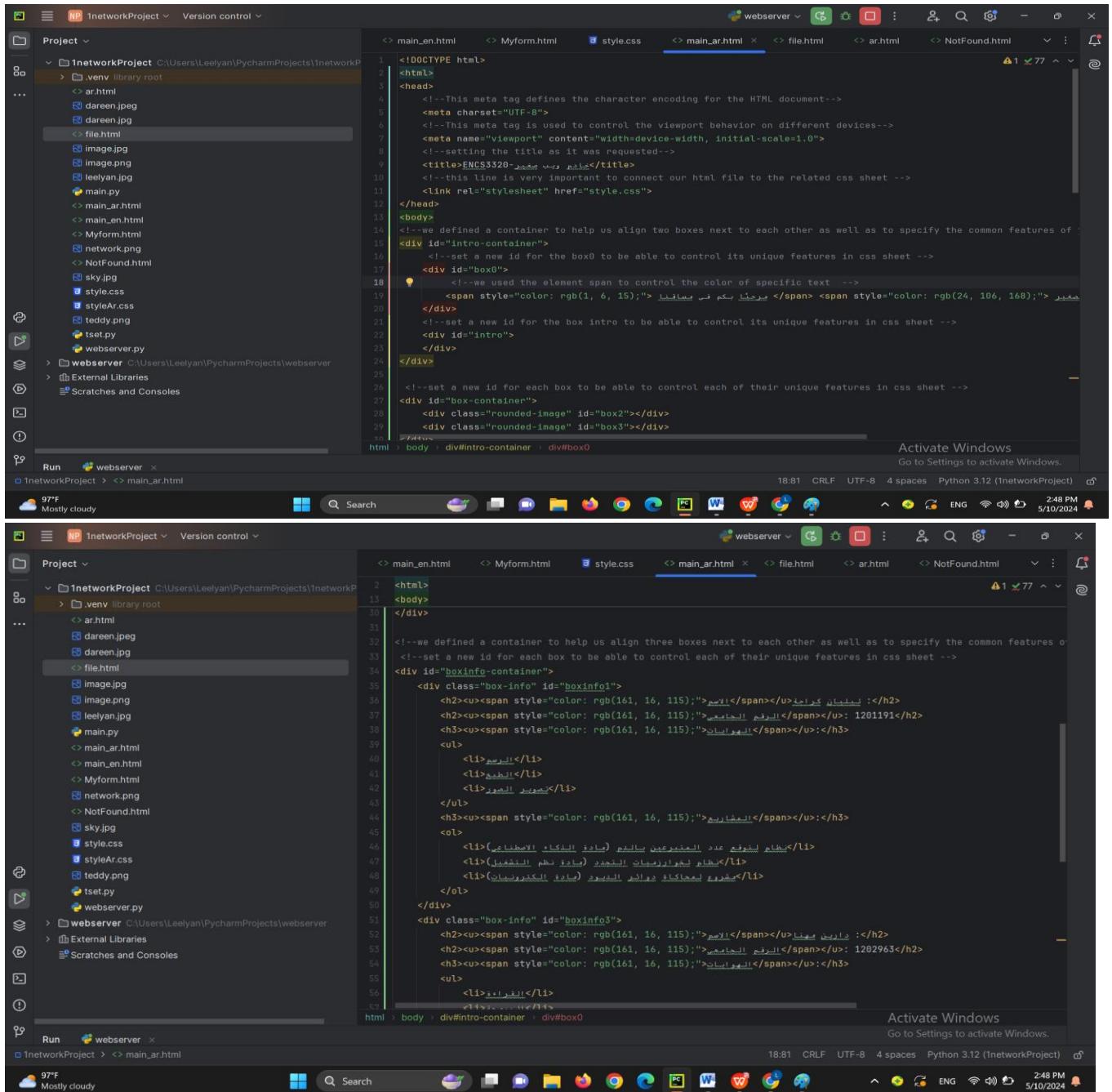
The W3schools button redirect you to the following page (a link that was provided by the project).

The screenshot shows a Google Chrome browser window with the address bar displaying "w3schools.com/python/python_syntax.asp". The main content area shows the "Python Syntax" page from w3schools. The left sidebar has a tree view of Python topics, with "Python Syntax" selected and highlighted in green. The main content area has a title "Python Syntax" with "Execute Python Syntax" below it. It shows a command-line example: `>>> print("Hello, World!")` followed by the output "Hello, World!". Below this, text says "As we learned in the previous page, Python syntax can be executed by writing directly in the Command Line:" and "Or by creating a python file on the server, using the .py file extension, and running it in the Command Line:". A sidebar on the right lists "On this page" topics: Execute Python Syntax, Python Indentation, Python Variables, Python Comments, and Exercises. At the bottom, there's a Windows taskbar with various icons and a weather widget showing "97°F Mostly cloudy".

Arabic version

- HTML (main_ar.html)

Figure 27:HTMLARABIC version



```
<!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>ENCS320-إندام دبى مفدى</title>
    <!--this line is very important to connect our html file to the related css sheet -->
    <link rel="stylesheet" href="style.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 both boxes-->
    <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 -->
            <span style="color: rgb(1, 6, 15);>العنوان يكتب بمفرد</span> <span style="color: rgb(24, 106, 168);>جدة</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 id="box0">
                <!--set a new id for each box to be able to control each of their unique features in css sheet -->
                <div id="box-container">
                    <div class="rounded-image" id="box2"></div>
                    <div class="rounded-image" id="box3"></div>
                </div>
            </div>
        </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 all three boxes-->
    <div id="boxInfo-container">
        <div class="box-info" id="boxinfo1">
            <h2><span style="color: rgb(161, 16, 115);>العنوان</span></h2>
            <h2><span style="color: rgb(161, 16, 115);>العنوان</span></h2> 1201193</h2>
            <h3><span style="color: rgb(161, 16, 115);>العنوان</span></h3>
            <ul>
                <li><span style="color: #0000ff; font-weight: bold;">العنوان</span></li>
                <li><span style="color: #0000ff; font-weight: bold;">العنوان</span></li>
                <li><span style="color: #0000ff; font-weight: bold;">العنوان</span></li>
            </ul>
            <h3><span style="color: rgb(161, 16, 115);>العنوان</span></h3>
            <ol>
                <li><span style="color: #0000ff; font-weight: bold;">العنوان</span></li>
                <li><span style="color: #0000ff; font-weight: bold;">العنوان</span></li>
                <li><span style="color: #0000ff; font-weight: bold;">العنوان</span></li>
            </ol>
        </div>
        <div class="box-info" id="boxinfo2">
            <h2><span style="color: rgb(161, 16, 115);>العنوان</span></h2>
            <h2><span style="color: rgb(161, 16, 115);>العنوان</span></h2> 1202963</h2>
            <h3><span style="color: rgb(161, 16, 115);>العنوان</span></h3>
            <ul>
                <li><span style="color: #0000ff; font-weight: bold;">العنوان</span></li>
                <li><span style="color: #0000ff; font-weight: bold;">العنوان</span></li>
                <li><span style="color: #0000ff; font-weight: bold;">العنوان</span></li>
            </ul>
        </div>
        <div class="box-info" id="boxinfo3">
            <h2><span style="color: rgb(161, 16, 115);>العنوان</span></h2>
            <h2><span style="color: rgb(161, 16, 115);>العنوان</span></h2> 1203042</h2>
            <h3><span style="color: rgb(161, 16, 115);>العنوان</span></h3>
            <ul>
                <li><span style="color: #0000ff; font-weight: bold;">العنوان</span></li>
                <li><span style="color: #0000ff; font-weight: bold;">العنوان</span></li>
                <li><span style="color: #0000ff; font-weight: bold;">العنوان</span></li>
            </ul>
        </div>
    </div>

```

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

- Project:** 1networkProject
- File:** main_ar.html
- Code Content:**

```
<html>
<body>
<div id="boxinfo-container">
<div class="box-info" id="boxinfo3">
<ul>
<li><a href="#">العنوان</a></li>
<li><a href="#">المدونة</a></li>
<li><a href="#">الاتصال</a></li>
</ul>
<span style="color: #800080;">نحوه- Toolbars and Status Bar: Shows the file path C:\Users\Leilyan\PycharmProjects\1networkProject\main_ar.html, status bar with 18:81, CRLF, UTF-8, 4 spaces, Python 3.12 (InetworkProject), and system icons.

```

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

- Project:** 1networkProject
- File:** main_ar.html
- Code Content:**

```
<html>
<body>
<div id="boxinfo-container">
<div class="box-info" id="boxinfo3">
<ul>
<li><a href="#">نظام مدرسي للطلاب (مقدمة مبادئ البيانات والخوارزميات)</a></li>
<li><a href="#">بيان المدرسة العالمية</a></li>
</ul>
</div>
<h2></h2>
<!-- the element for hyper links -->
<a href="#" target="_blank">
<button id = "button1">نحوه</button>
</a>
</h2>
<a href="https://www.w3schools.com/python/python_syntax.asp" target="_blank">
<button id = "button2">بيان المدرسة العالمية</button>
</a>
</h2>
</body>

```

- Toolbars and Status Bar:** Shows the file path C:\Users\Leilyan\PycharmProjects\1networkProject\main_ar.html, status bar with 18:81, CRLF, UTF-8, 4 spaces, Python 3.12 (InetworkProject), and system icons.

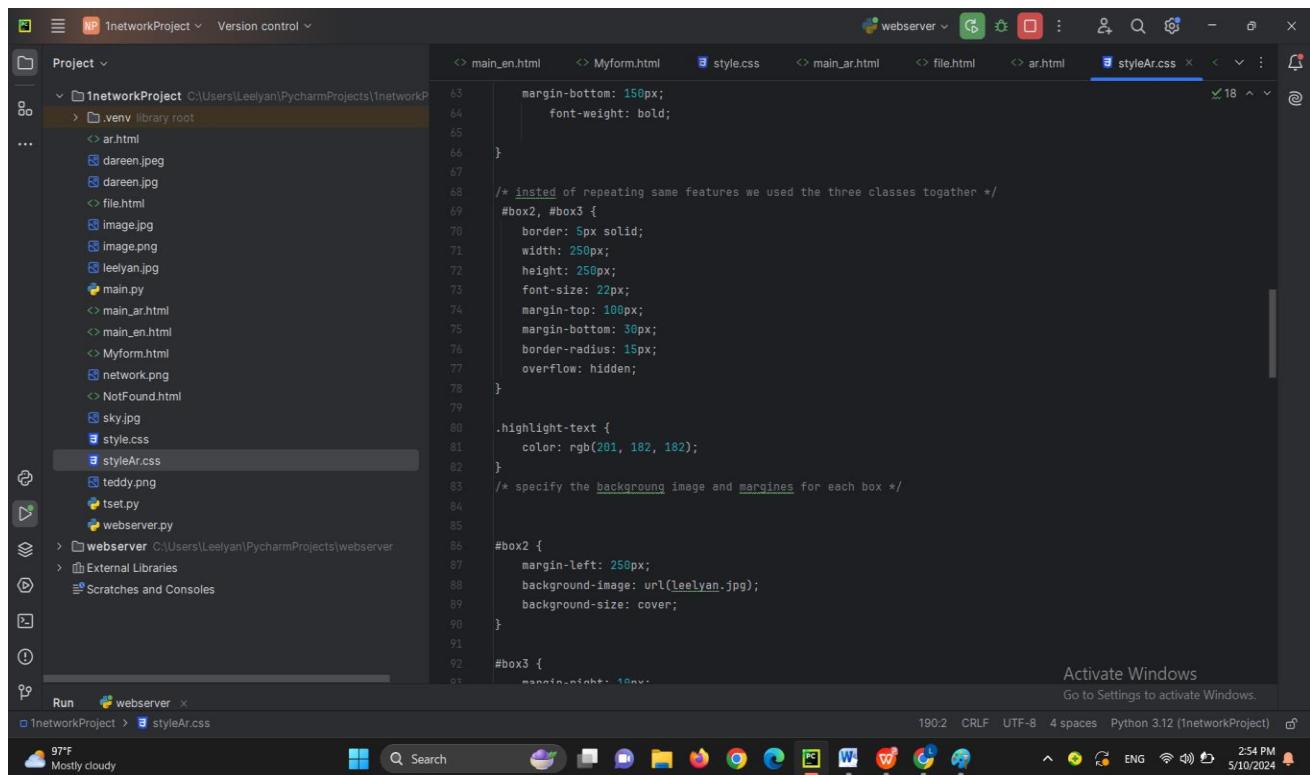
○ styleAr.css

Figure 28:CSS arabic version

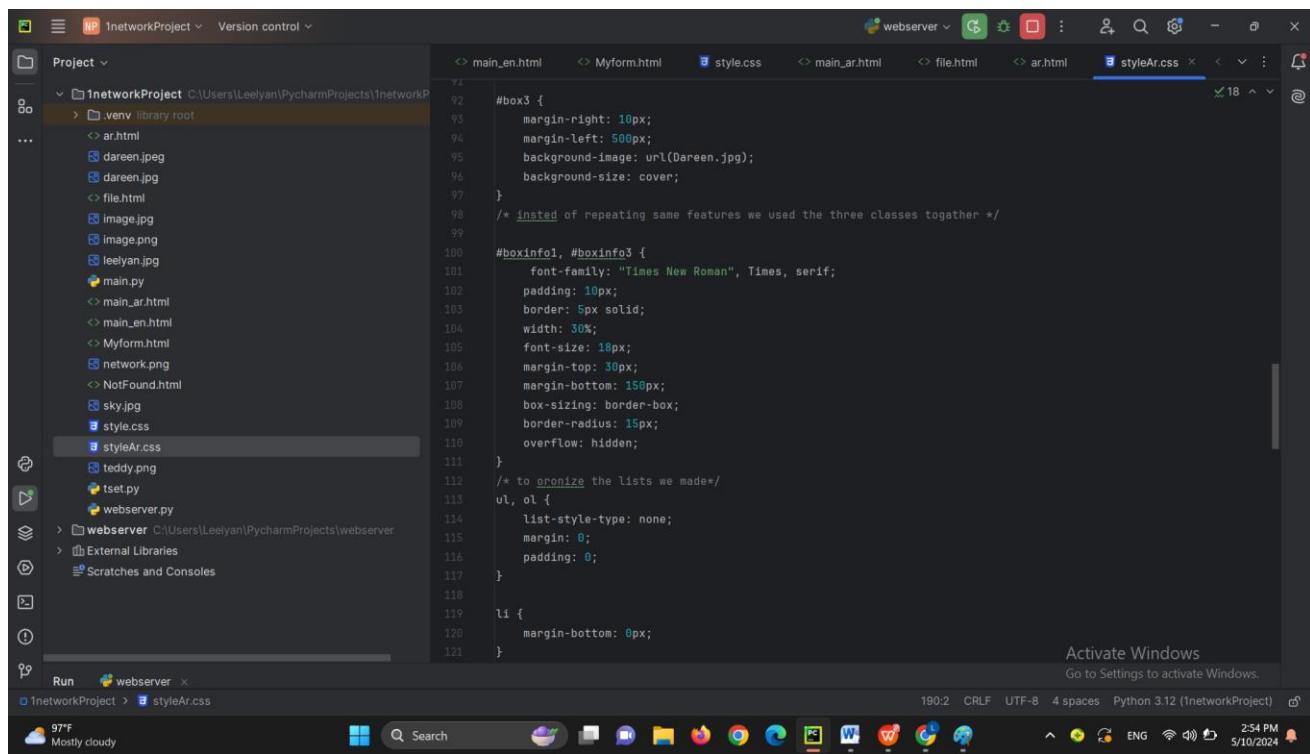
```
1  /* some generalized features in the body */
2  body {
3      background-image: url(sky.jpg);
4      background-repeat: no-repeat;
5      background-attachment: fixed;
6      background-position: center;
7      background-size: cover;
8      margin: 0;
9      padding: 0;
10     font-family: 'Arial', 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
22 /* created a class to put the common features of all used rounded images*/
23 .rounded-image {
24     border-radius: 15px;
25     overflow: hidden;
26     width: 250px;
27     height: 250px;
28     margin-top: 100px;
29     background-size: cover;
30 }
31
32 /* using flex we can align more than one box next to each other */
33
```

The screenshot shows the PyCharm IDE interface. The top bar displays the project name "1networkProject" and "Version control". The left sidebar shows the project structure under "Project". The main area is a code editor with the file "styleAr.css" open. The code defines CSS styles for ".boxinfo-container" and "#box4". The bottom status bar shows "Activate Windows Go to Settings to activate Windows.", the file path "1networkProject > styleAr.css", and system information like "190:2 CRLF UTF-8 4 spaces Python 3.12 (1networkProject)". The bottom navigation bar includes icons for Run, Stop, and Build.

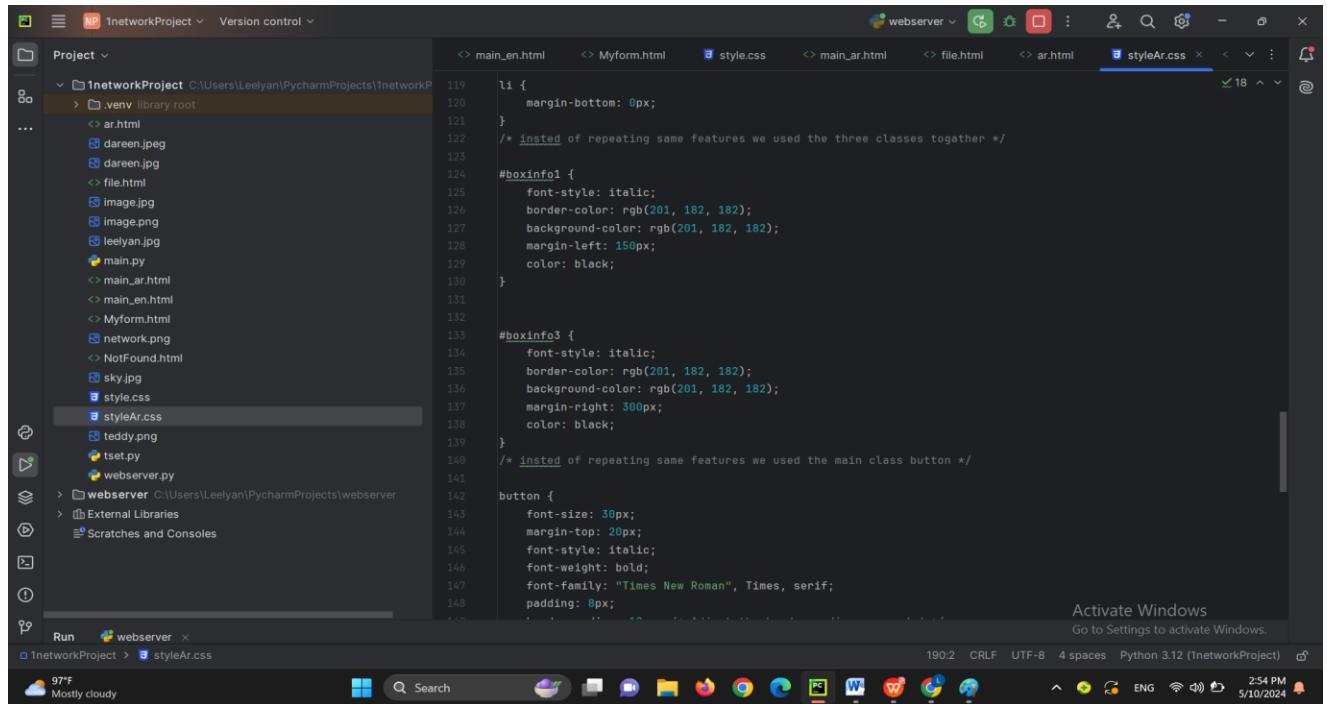
```
32 #boxinfo-container {  
33     display: flex;  
34     justify-content: space-between;  
35 }  
36  
37 .box-info {  
38     padding: 10px;  
39     border: 5px solid;  
40     width: 30%;  
41     font-size: 18px;  
42     margin-top: 30px;  
43     box-sizing: border-box;  
44     border-radius: 15px;  
45     overflow: hidden;  
46 }  
47 /* we used some features to make our web page look nicer and more organized */  
48  
49 #box4 {  
50     font-family: "Times New Roman", Times, serif;  
51     font-style: italic;  
52     margin: 20px auto;  
53     border: 0px ;  
54     width: 80%;  
55     font-size: 22px;  
56     color: black;  
57  
58     padding: 10px;  
59     background-image: url(teddy.png);  
60     background-size: cover;  
61     border-radius: 15px;  
62     overflow: hidden;  
63 }
```



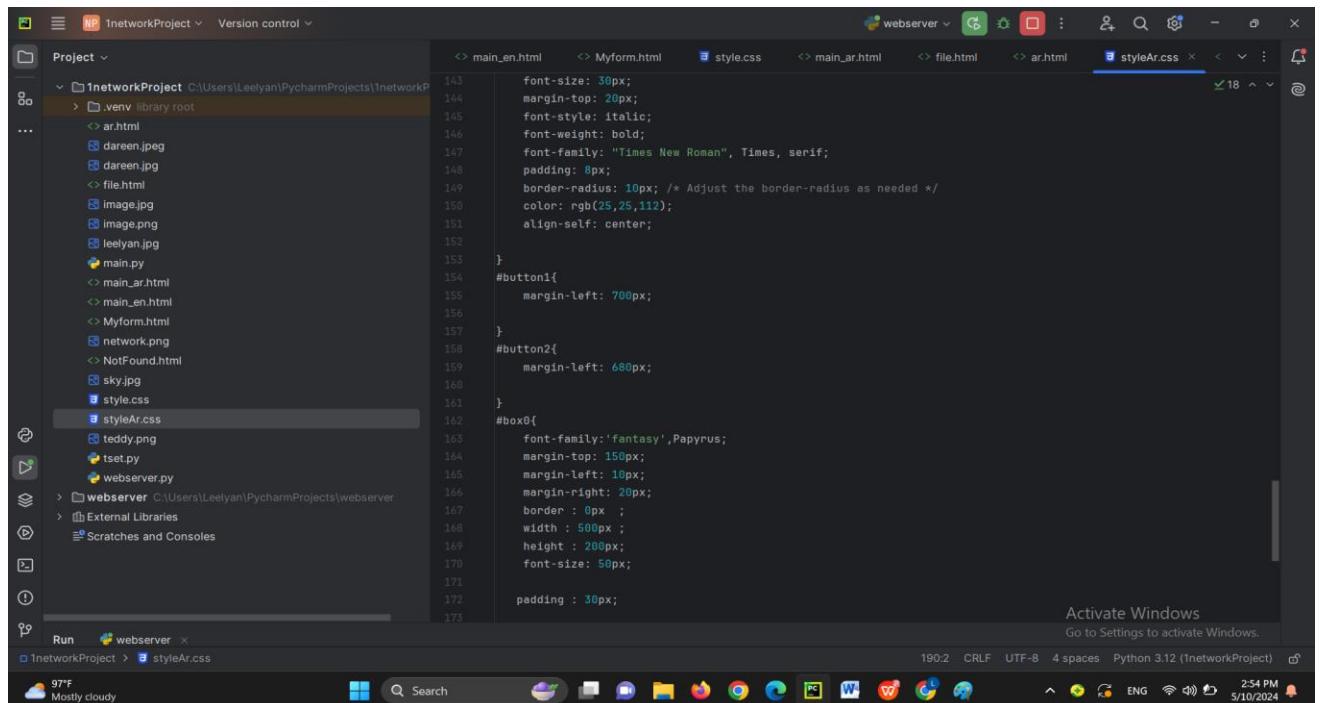
```
63 margin-bottom: 150px;
64 font-weight: bold;
65 }
66 /* instead of repeating same features we used the three classes together */
67 #box2, #box3 {
68 border: 5px solid;
69 width: 250px;
70 height: 250px;
71 font-size: 22px;
72 margin-top: 100px;
73 margin-bottom: 30px;
74 border-radius: 15px;
75 overflow: hidden;
76 }
77
78 .highlight-text {
79 color: rgb(201, 182, 182);
80 }
81 /* specify the background image and margins for each box */
82
83 #box2 {
84 margin-left: 250px;
85 background-image: url(leelyan.jpg);
86 background-size: cover;
87 }
88
89 #box3 {
90 margin-right: 100px;
91 }
```



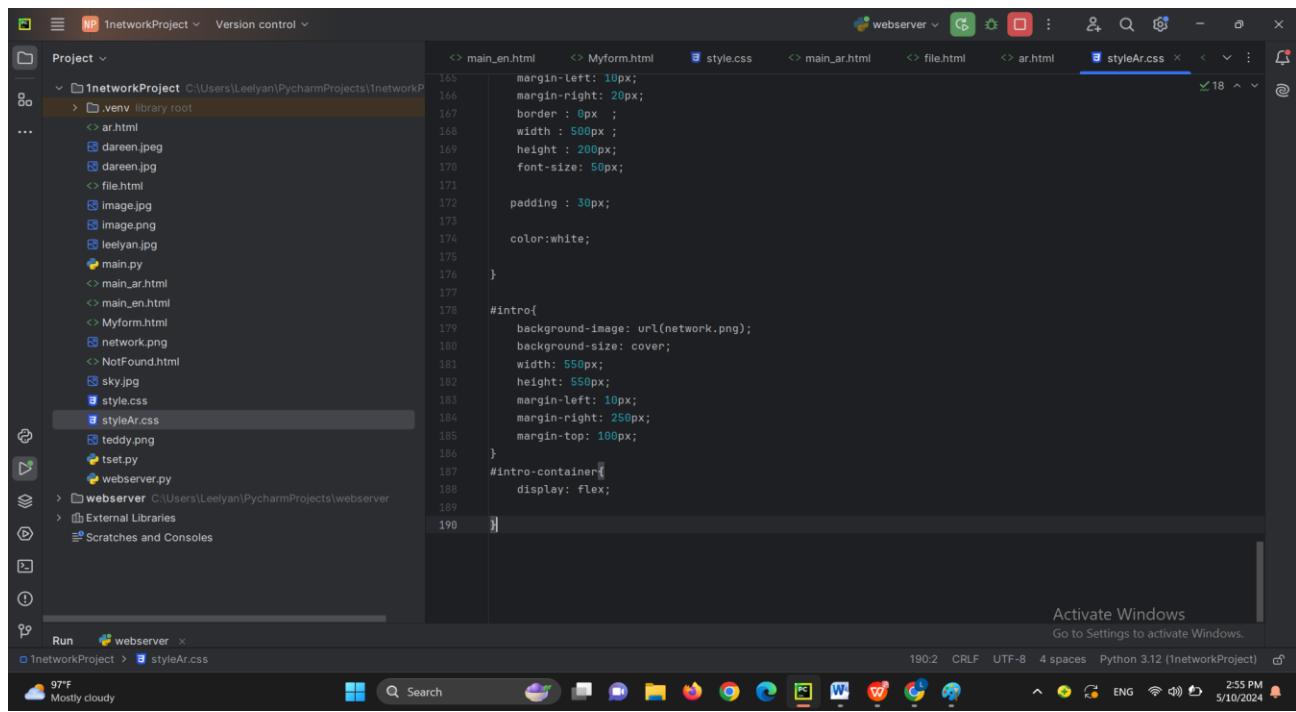
```
92 #box3 {
93 margin-right: 10px;
94 margin-left: 500px;
95 background-image: url(Dareen.jpg);
96 background-size: cover;
97 }
98 /* instead of repeating same features we used the three classes together */
99
100 #boxinfo1, #boxinfo3 {
101 font-family: "Times New Roman", Times, serif;
102 padding: 10px;
103 border: 5px solid;
104 width: 30%;
105 font-size: 18px;
106 margin-top: 30px;
107 margin-bottom: 150px;
108 box-sizing: border-box;
109 border-radius: 15px;
110 overflow: hidden;
111 }
112 /* to organize the lists we made*/
113 ul, ol {
114 list-style-type: none;
115 margin: 0;
116 padding: 0;
117 }
118
119 li {
120 margin-bottom: 8px;
121 }
```



```
119 li {
120     margin-bottom: 0px;
121 }
122 /* Instead of repeating same features we used the three classes together */
123
124 #boxinfo1 {
125     font-style: italic;
126     border-color: rgb(201, 182, 182);
127     background-color: rgb(201, 182, 182);
128     margin-left: 150px;
129     color: black;
130 }
131
132 #boxinfo3 {
133     font-style: italic;
134     border-color: rgb(201, 182, 182);
135     background-color: rgb(201, 182, 182);
136     margin-right: 300px;
137     color: black;
138 }
139
140 /* Instead of repeating same features we used the main class button */
141
142 button {
143     font-size: 30px;
144     margin-top: 20px;
145     font-style: italic;
146     font-weight: bold;
147     font-family: "Times New Roman", Times, serif;
148     padding: 8px;
149 }
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
```



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

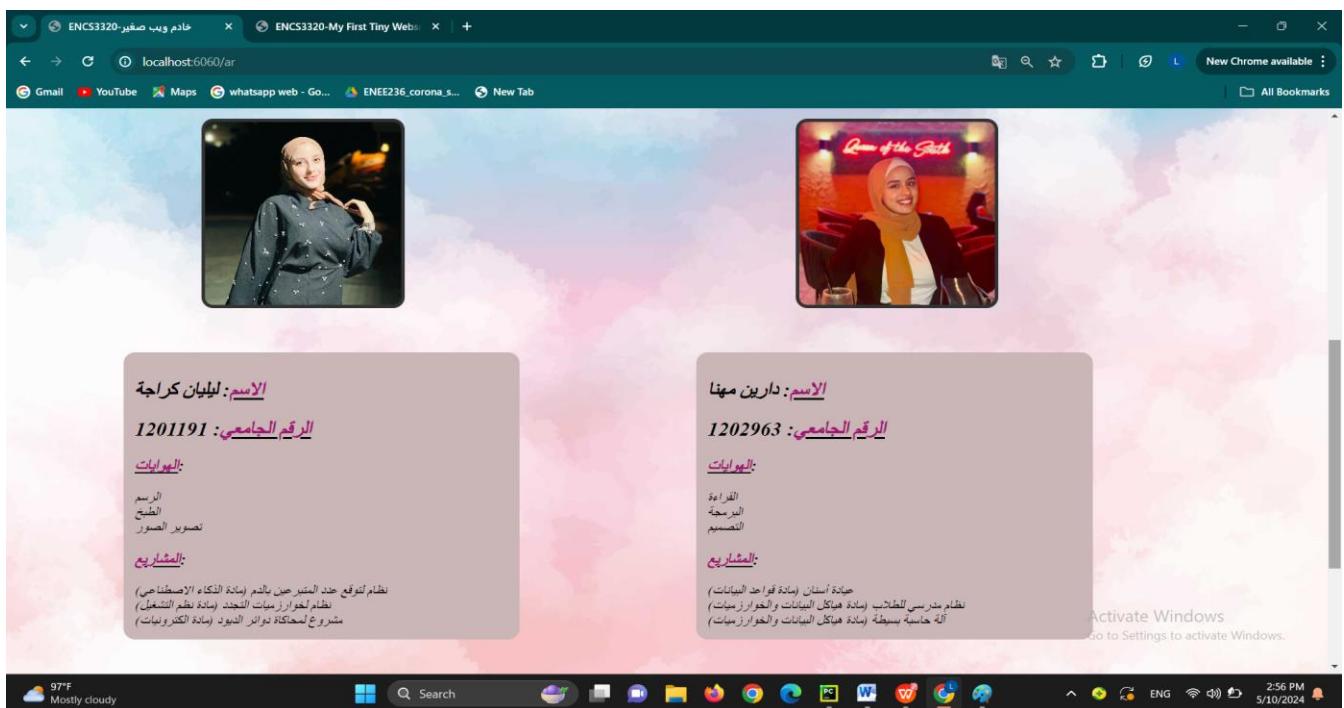
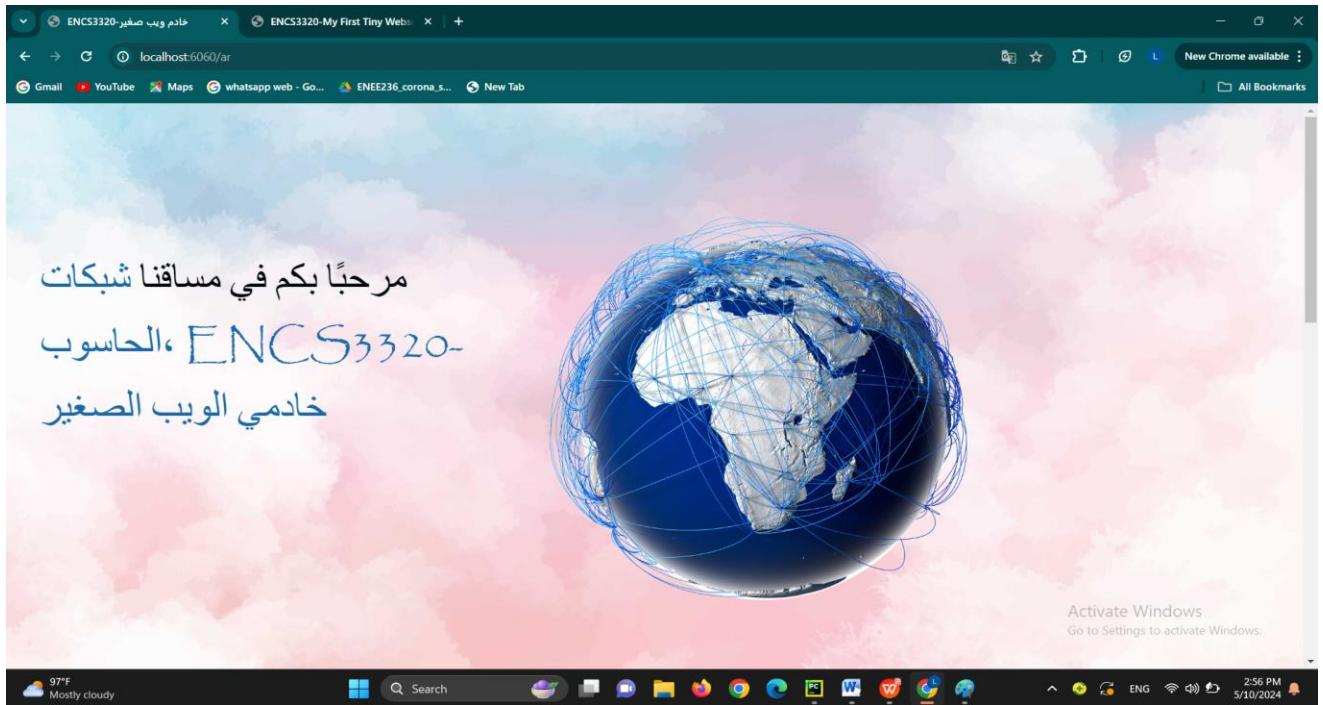


The screenshot shows the PyCharm IDE interface with the following details:

- Project View:** The left sidebar displays the project structure under "1networkProject". It includes files like "ar.html", "dareen.jpg", "file.html", "image.jpg", "image.png", "leelyan.jpg", "main.py", "main_ar.html", "main_en.html", "MyForm.html", "network.png", "NotFound.html", "sky.jpg", "style.css", and "styleAr.css".
- Code Editor:** The main window shows the content of the "styleAr.css" file. The code defines styles for "main_en.html" and "Myform.html".

```
165 margin-left: 10px;
166 margin-right: 20px;
167 border: 0px;
168 width: 500px;
169 height: 200px;
170 font-size: 50px;
171 padding: 30px;
172 color: white;
173 }
174 #intro {
175 background-image: url(network.png);
176 background-size: cover;
177 width: 550px;
178 height: 550px;
179 margin-left: 10px;
180 margin-right: 250px;
181 margin-top: 100px;
182 }
183 #intro-container {
184 display: flex;
185 }
186 }
187 }
188 }
189 }
190 }
```
- Status Bar:** At the bottom, the status bar shows "Activate Windows", "Go to Settings to activate Windows.", file statistics ("190:2 CRLF, UTF-8, 4 spaces"), and Python version ("Python 3.12 (1networkProject)").
- System Tray:** The bottom right corner shows system icons for weather (97°F Mostly cloudy), search, taskbar, and various application icons.

○ Web server



The screenshot shows a web browser window with two student profiles displayed against a pink and blue gradient background.

Student Profile 1 (Left):

- الاسم:** ليلىان كراجة
- الرقم الجامعي:** 1201191
- الموابات:**
 - الرسم
 - المطبخ
 - تصوير الصور
- المشاريع:**
 - نظام لتوفيق عدد المترحبين باسم (مادة النكاء الإصطناعي)
 - نظام لخوارزميات التعدد (مادة نظم التشغيل)
 - مشروع لمحاكاة دوائر الدايموند (مادة الكترونيات)

Student Profile 2 (Right):

- الاسم:** دارين مهنا
- الرقم الجامعي:** 1202963
- الموابات:**
 - القراءة
 - البرمجة
 - التصميم
- المشاريع:**
 - عجلة أسنان (مادة قواعد البيانات)
 - نظام مدريسي للطاحب (مادة هيكل البيانات والخوارزميات)
 - آلة حاسوبية بسيطة (مادة هيكل البيانات والخوارزميات)

Buttons:

- نوعي (blue button)
- وثائق المدرسة العالمية 3 (button with a black border)

Browser Status Bar:

- Activate Windows
- Go to Settings to activate Windows.
- 2:56 PM
- ENG
- 5/10/2024

The screenshot shows a web browser window with a file upload interface.

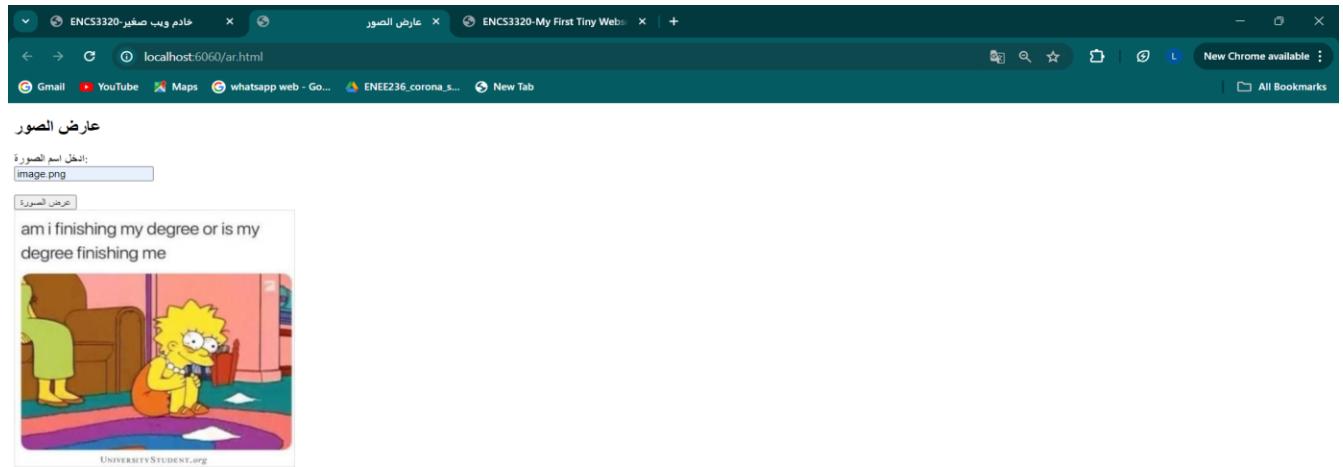
Section Title: عرض الصور

Input Fields:

- اندخل اسم الصورة:
- عرض الصورة:

Browser Status Bar:

- Activate Windows
- Go to Settings to activate Windows.
- 2:57 PM
- ENG
- 5/10/2024



Activate Windows
Go to Settings to activate Windows.

