Department of Electrical and Computer Engineering

ENCS3320 - Computer Networks

Summer Semester 2023/2024

Project#1: Socket Programming

Prepared by:

Mouath Masalmah - 1220179

Zaid Mousa - 1221833

Bahaa Bani Shamsaa - 1220252

Instructor: Dr.Ibrahim Nemer

Section: SECTION_1

Table of Contents

Table of Figures

Theory:

Task 1: Commands & Wireshark

1. **Ping:**
    - Ping is a network diagnostic tool used to test the reachability of a host on an IP network. It works by sending ICMP Echo Request packets to the target host and waiting for an Echo Reply. This helps in measuring the round-trip time and identifying network issues.
2. **Tracert:**
    - Tracert (or traceroute) is a utility that traces the path that a packet takes from the source to the destination across multiple routers. It helps in identifying the route, delays, and any points of failure along the path.
3. **Nslookup:**
    - Nslookup is used to query the Domain Name System (DNS) to obtain domain name or IP address mappings. It's a helpful tool for troubleshooting DNS-related issues and understanding the DNS structure.
4. **Telnet:**
    - Telnet is a protocol that allows you to connect to another machine on the network, typically for managing servers or network devices. It's an older protocol and lacks security, making it less commonly used today.

Task 2: Socket Programming (TCP and UDP)

11. **TCP Client-Server Application:**
    - TCP is a connection-oriented protocol that ensures reliable data transmission between client and server. In this task, the client sends a string to the server, which processes it (replacing vowels with #) and sends it back.
12. **UDP Client-Server Application:**
    - UDP is a connectionless protocol that allows fast data transmission with no guarantee of delivery. The server listens on a specific port, and clients send messages to it. The server keeps track of the last message received from each client.

Task 3: Web Server

15. **Web Server Application:**
    - A web server listens for HTTP requests from clients (browsers) and serves them content based on the request. The server needs to handle various content types (HTML, CSS, images) and redirect or return error pages as necessary.

Procedure:

Task 1: Commands & Wireshark

5. **Ping a device in the same network:**
    - Connect your laptop and smartphone to the same Wi-Fi network.
    - Open the command prompt on your laptop.
    - Type ping <smartphone IP> to test connectivity.
6. **Ping www.ox.ac.uk:**
    - In the command prompt, type ping www.ox.ac.uk.
    - Note the response times and analyze the location based on the IP.
7. **Tracert www.ox.ac.uk:**
    - Type tracert www.ox.ac.uk in the command prompt.
    - Observe the hops and response times along the route to Oxford University.
8. **Nslookup www.ox.ac.uk:**
    - Type nslookup www.ox.ac.uk in the command prompt.
    - Record the IP address returned and verify its ownership.
9. **Telnet www.ox.ac.uk:**
    - Type telnet www.ox.ac.uk in the command prompt.
    - Attempt to establish a connection and note the result.
10. **Wireshark DNS Capture:**
    - Open Wireshark and start capturing packets.
    - Perform some DNS queries (e.g., visit a website).
    - Stop the capture and analyze the DNS messages.
    - Take screenshots showing the time and date.

Task 2: Socket Programming (TCP and UDP)

13. **TCP Client-Server:**
    - Write a server program that listens on a port.
    - Write a client program that connects to the server and sends a string.
    - The server replaces vowels with # and sends the modified string back.
    - The client prints the received string.
14. **UDP Client-Server:**
    - Write a UDP server that listens on a port.
    - Write UDP clients that send messages to the server.

- The server logs the messages and the clients print their received messages.
- Test the communication between multiple clients and the server.

Task 3: Web Server

16. **Web Server Implementation:**
    - Write a server program that listens on a specific port.
    - Handle different URL requests by serving the appropriate HTML or CSS files.
    - Implement logic to return custom error pages for invalid requests.
    - Include images, links, and redirects as specified.
    - Test the server by making various requests from a browser (e.g., http://localhost:1515/en).
17. **HTML and CSS:**
    - Create HTML files (main_en.html, main_ar.html, etc.) that include information about the team, images, and links.
    - Style the pages using CSS, ensuring the layout is visually appealing.
    - Store all files in the appropriate directories for the server to access.
18. **Testing and Documentation:**
    - Run the server and access it through a browser.
    - Document the theory, procedure, and results with explanations and screenshots.

Result&Discussion

Task1: Commands & Wireshark

Part 1: In your words, what are ping, tracert, nslookup, and telnet?

Ping: is a network tool that tests the possibility of sending messages between devices by transmitting small data packets. It also measures the time taken for these packets to be sent and received, helping to identify any network issues that need to be resolved.

Tracert: It is a network tool that displays the path that data takes to reach a destination by showing all the routers it passes through. It is useful for seeing the path and checking for any network delays along the way.

Nslookup: It is a command line tool used in network administration to find the domain name or IP address associated with a website. It helps diagnose DNS problems by examining and retrieving DNS records.

Telnet: It is a protocol that allows users to connect to remote computers over the Internet or a local network, allowing them to interact with the remote device as if they were directly in front of it, issuing commands and controlling the system through a text interface.

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

Point A: ping a device in the same network, e.g. from a laptop to a smartphone.

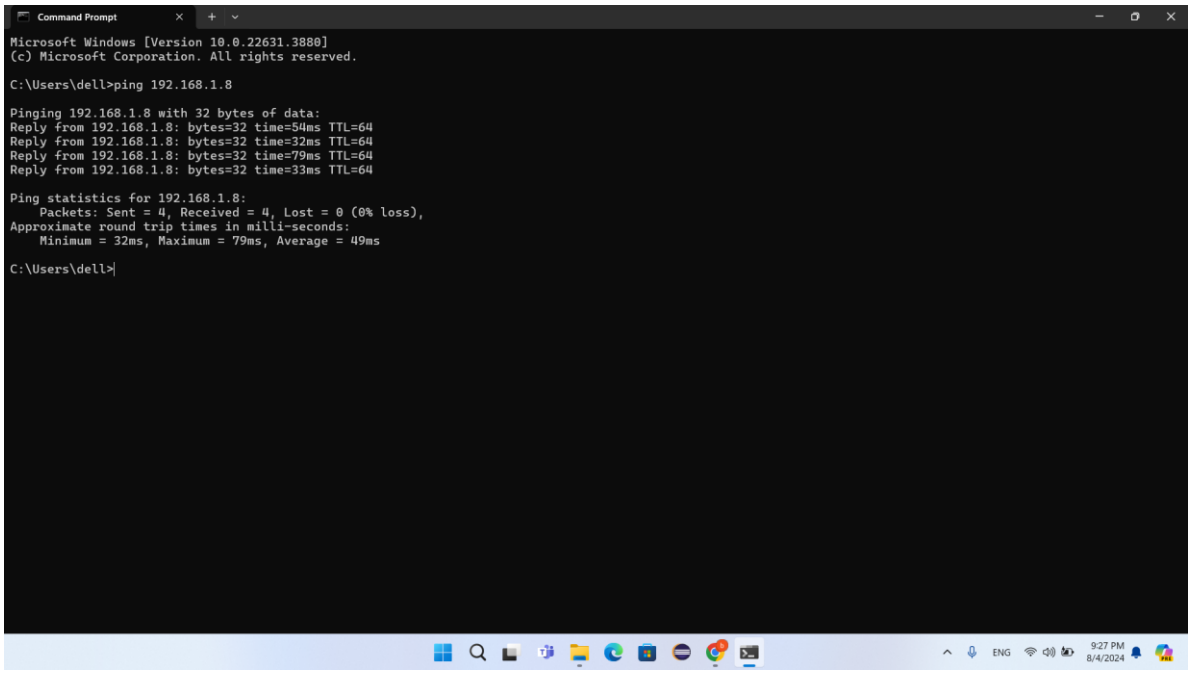


Figure 1: pinging my network

Here when we ping a device in the same network, my laptop sends small data packets to your smartphone to check if it's online. If the smartphone is connected, it replies, and we get the response time, showing that the device is reachable.

Point B: ping *www.ox.ac.uk.*

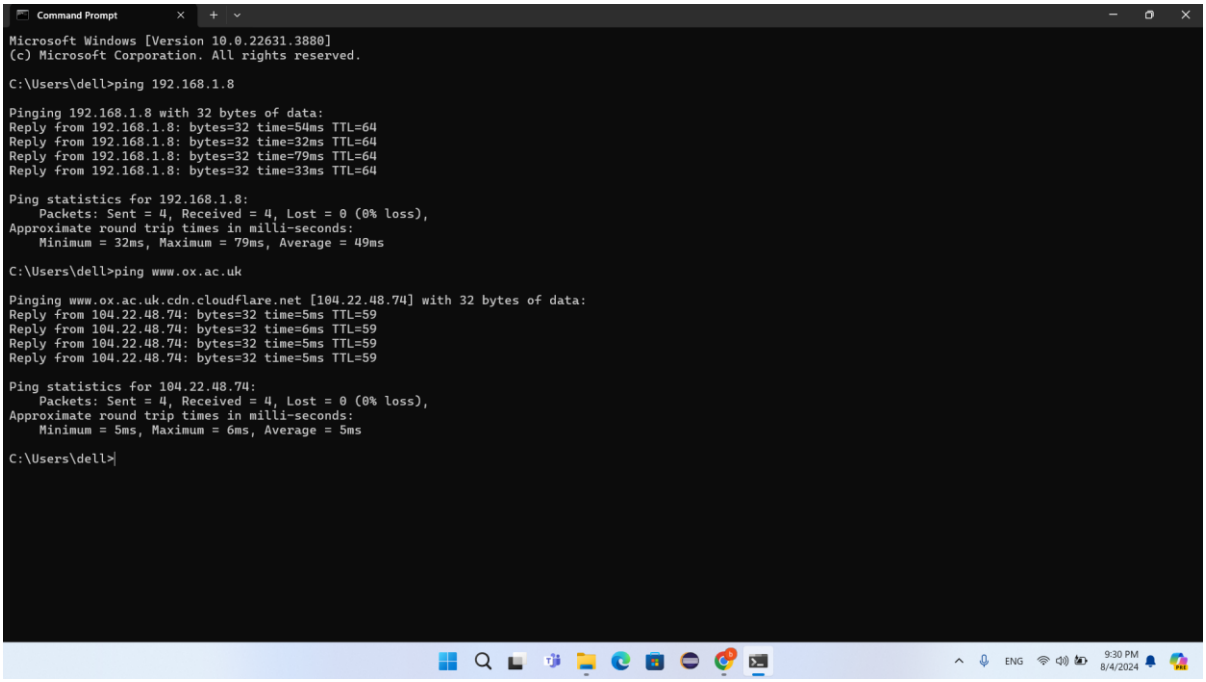


Figure 2: pinging www.ox.ac.uk

When we ping `www.ox.ac.uk`, my computer sends data packets to the Oxford University website's server. The server replies if it's online, and you see the response time. This helps to check if the website is reachable and how quickly it responds.

Point C: From the ping results, specify the location of the server from where you got the response? Explain your answer briefly.

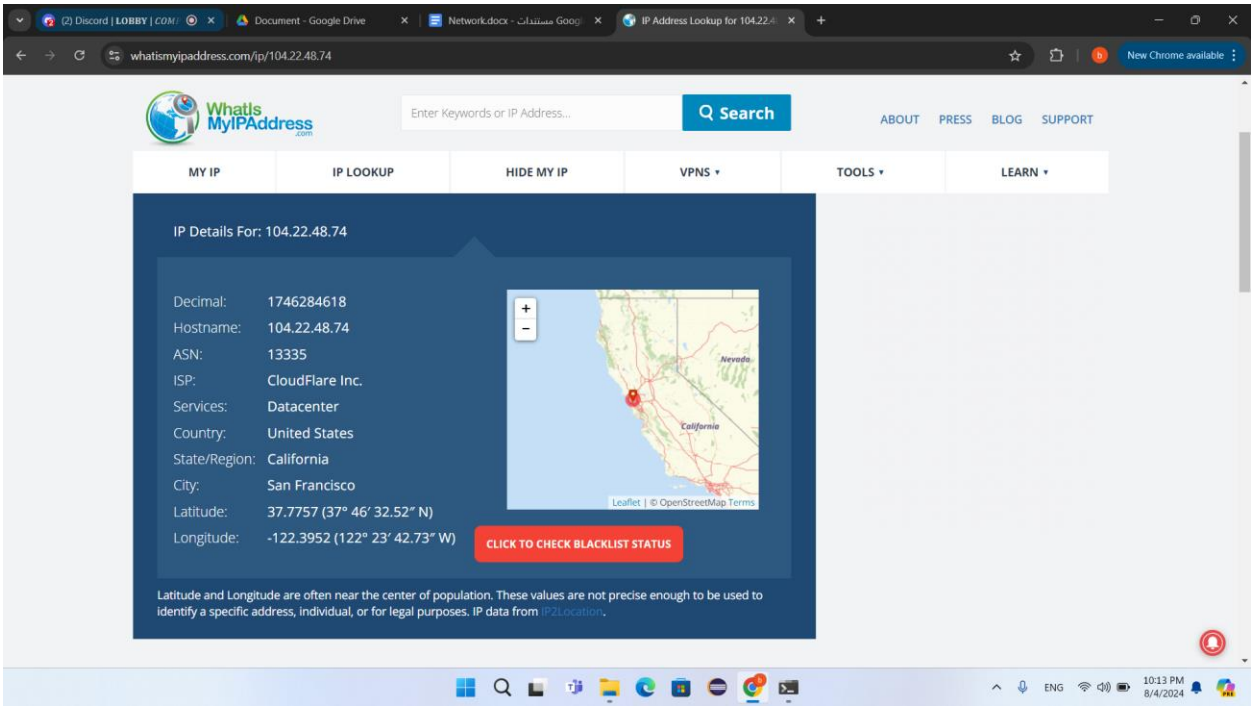


Figure 3:location server response

From the ping results alone, we typically cannot directly determine the exact physical location of the server. The ping command shows me the IP address of the server and the round-trip time (RTT), but it doesn't provide geographic information.

However, we can use the IP address obtained from the ping results and perform an IP geolocation lookup to estimate the server's location. This method isn't perfectly accurate, but it gives a general idea of where the server might be located.

Point D: tracert *www.ox.ac.uk*.

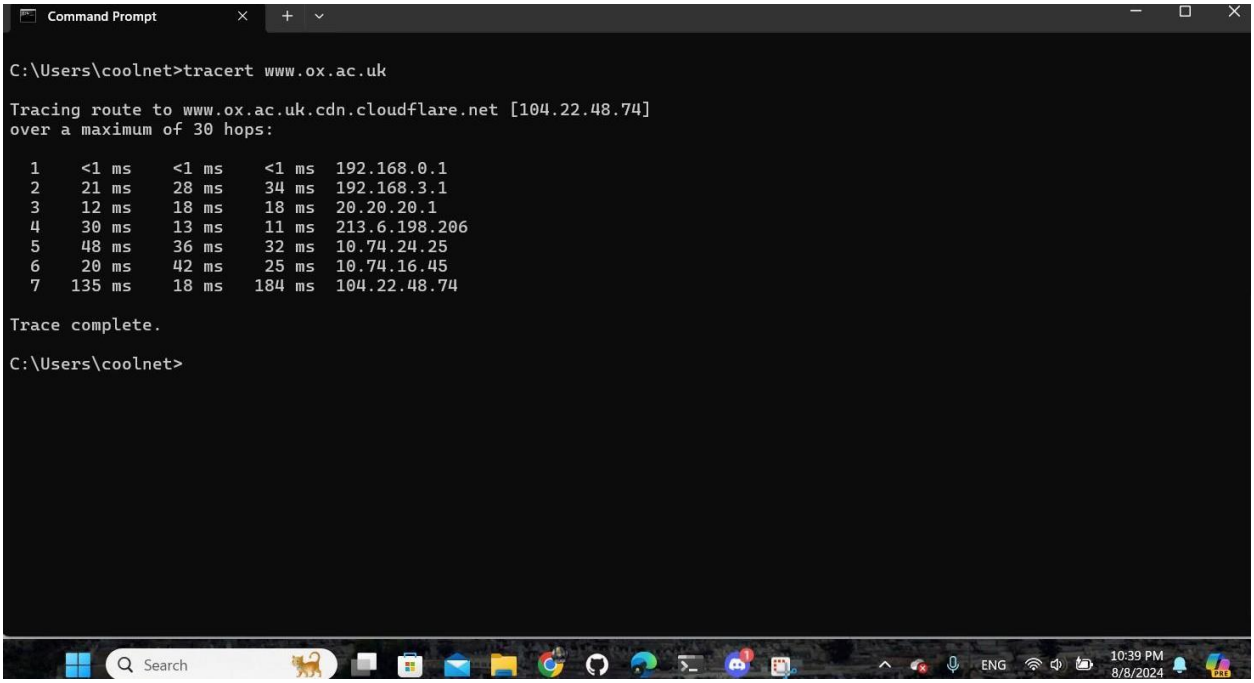


Figure 4: tracert www.ox.ac.uk

That shows me the path my data takes from my device to Oxford University's server, listing each router (hop) along the way and how long it takes to reach each one.

Point E: nslookup *www.ox.ac.uk.*

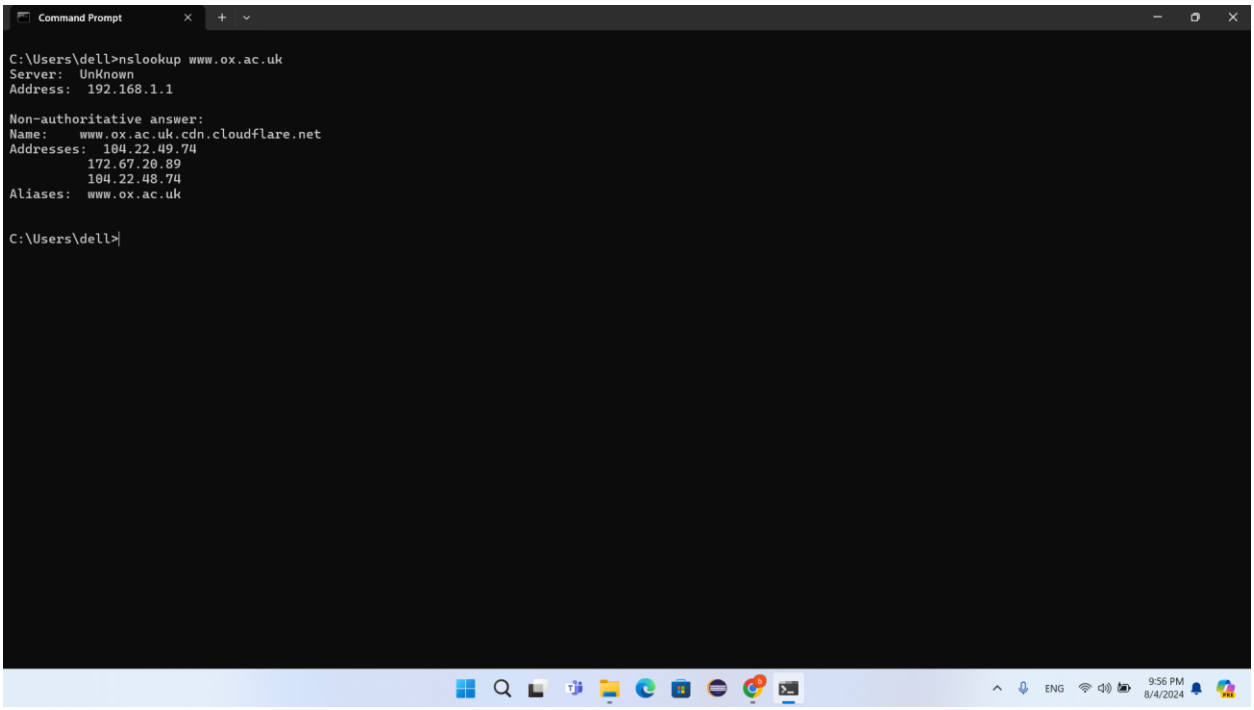


Figure 5: nslookup www.ox.ac.uk

It retrieves the IP address of the Oxford University website by querying a DNS server.

The point F: telnet *www.ox.ac.uk*.

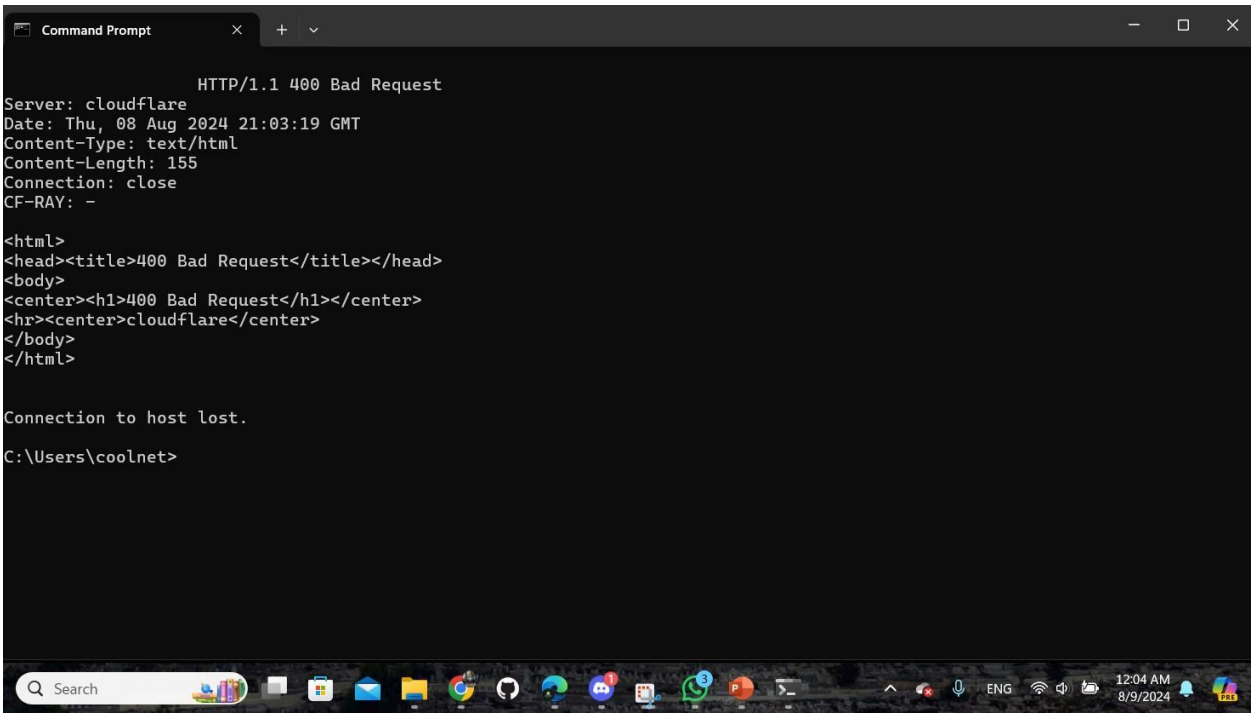


Figure 6: telnet www.ox.ac.uk

The "Bad Request" error happens because Telnet sends an incomplete or improperly formatted request that the server can't understand. The server is reachable, but the request isn't correct.

Part 3: Give some details about autonomous system (AS) number, number of IPs, prefixes, peers, name of Tier1-ISP of *www.ox.ac.uk*.

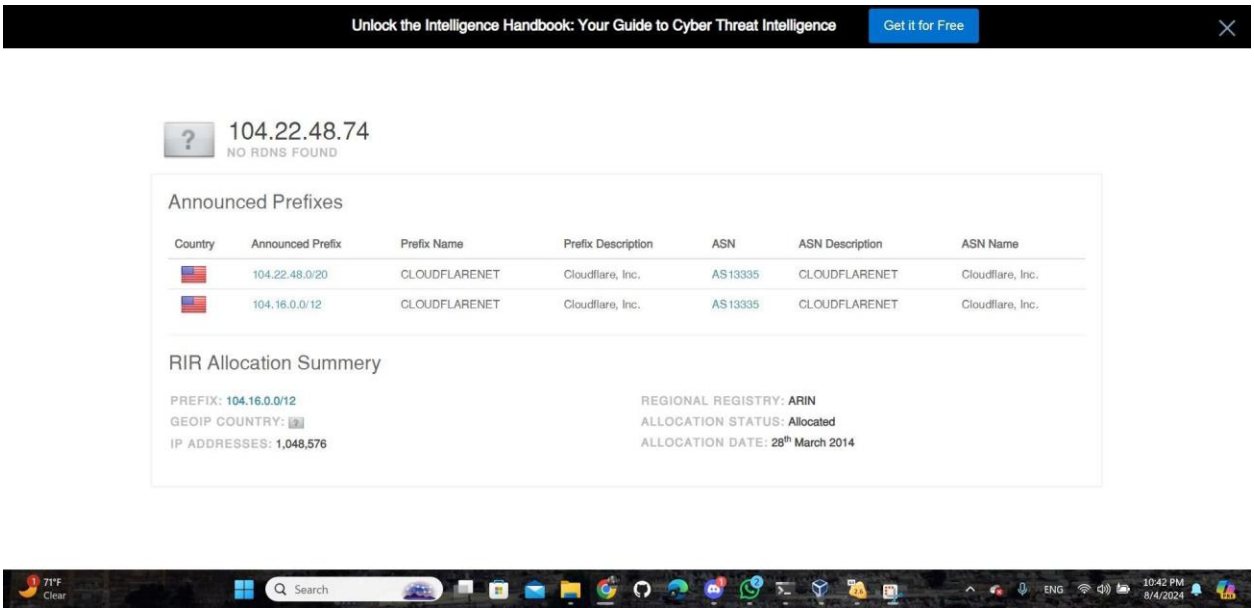


Figure 7: Details about tier1 ISP for www.ox.ac.uk

Here we use this website to find all details we need about "www.ox.ac.uk" that we need.

Part 4: *Wireshark*

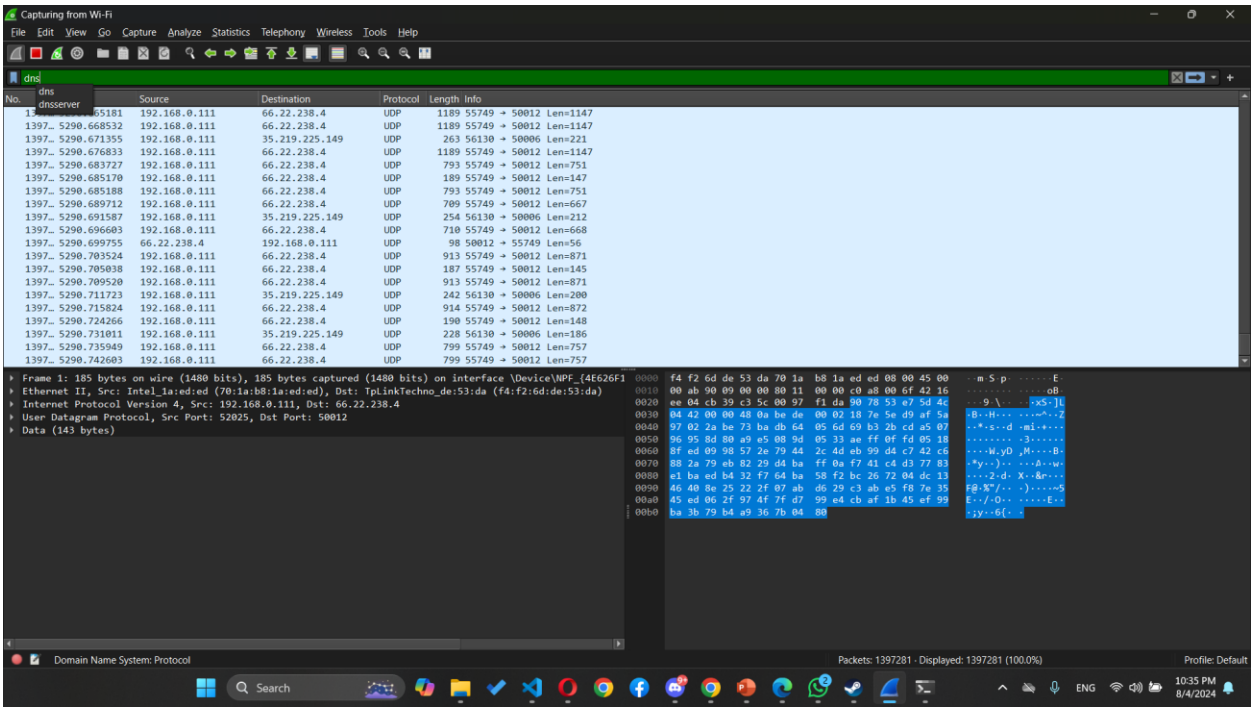


Figure 8: dns on wire shark

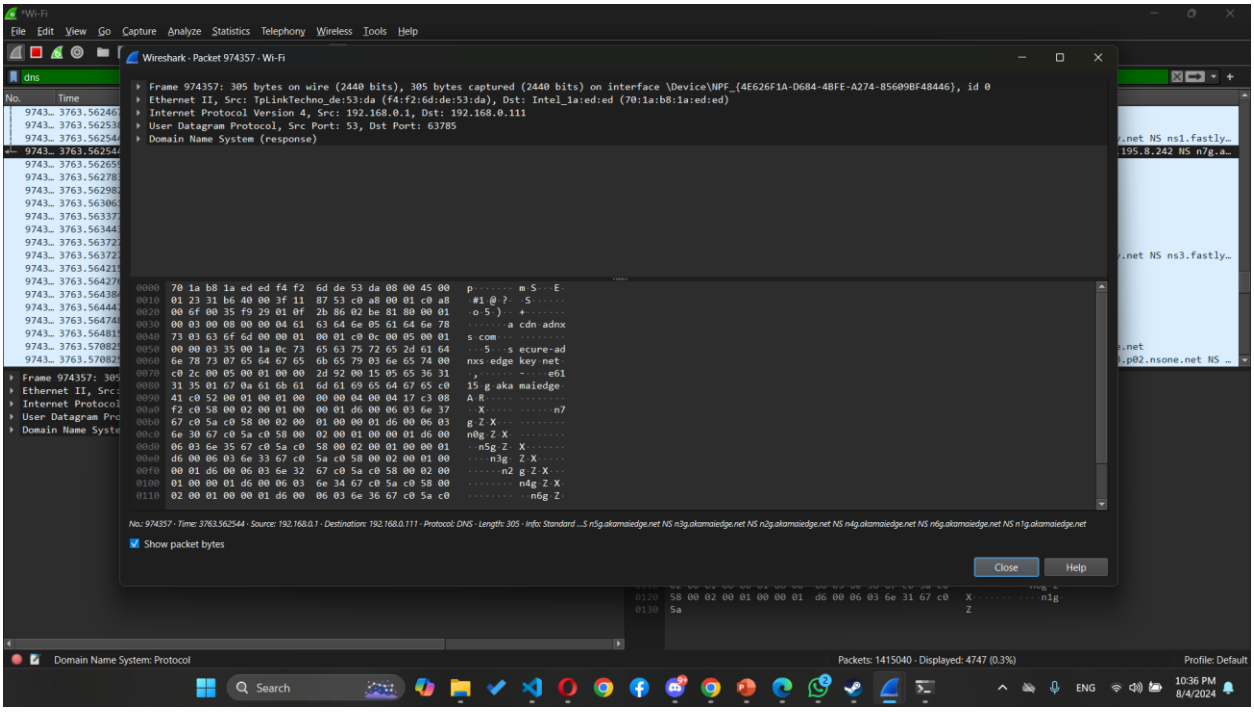


Figure 9: details for one of the packets

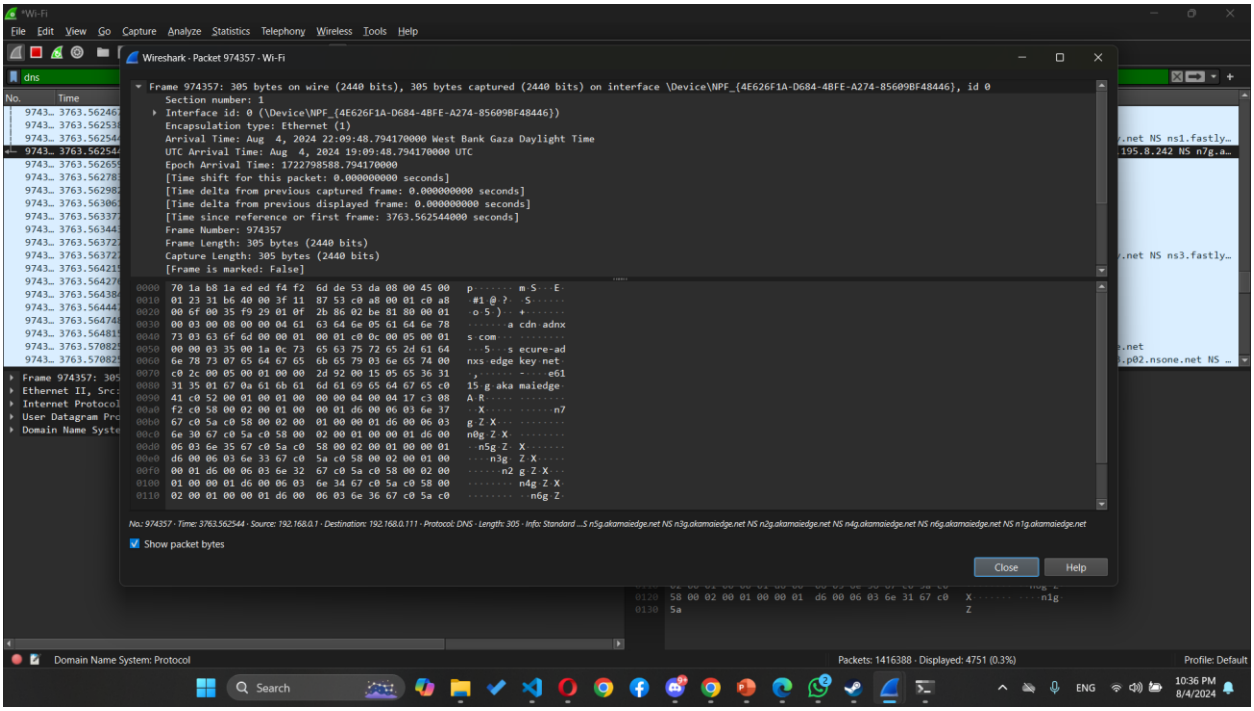


Figure 10: frame details on the packet

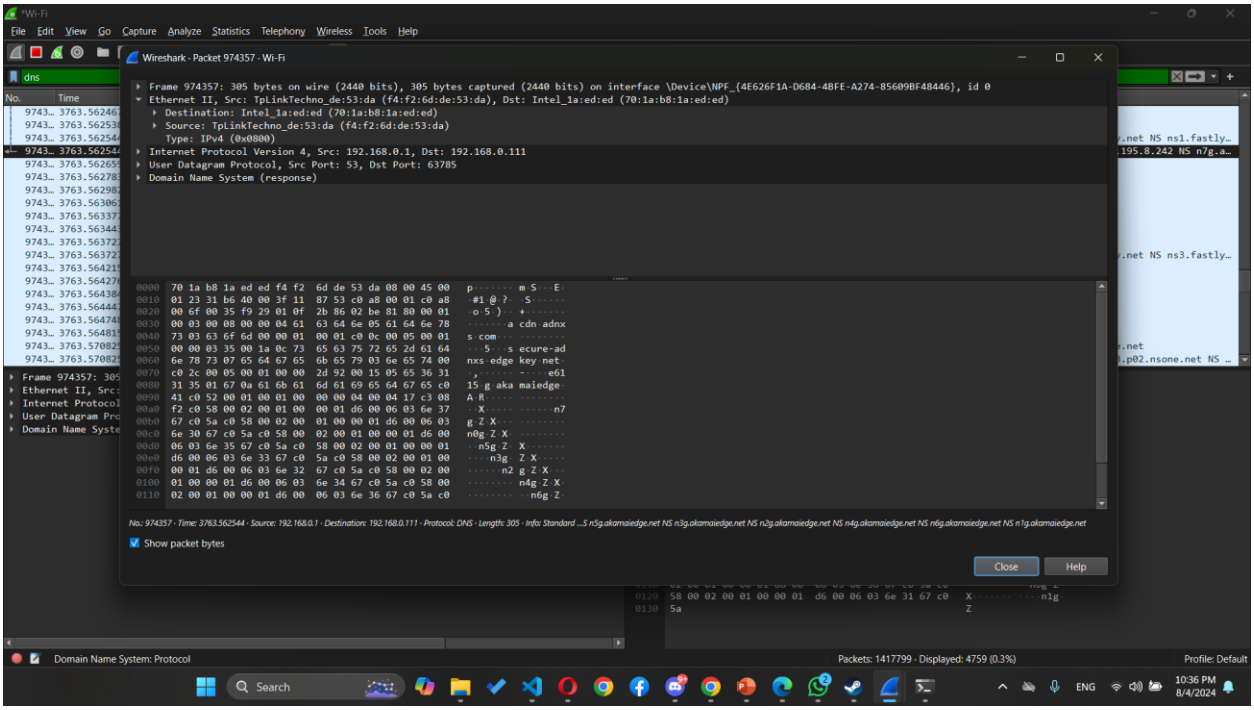


Figure 11: ethernet II details

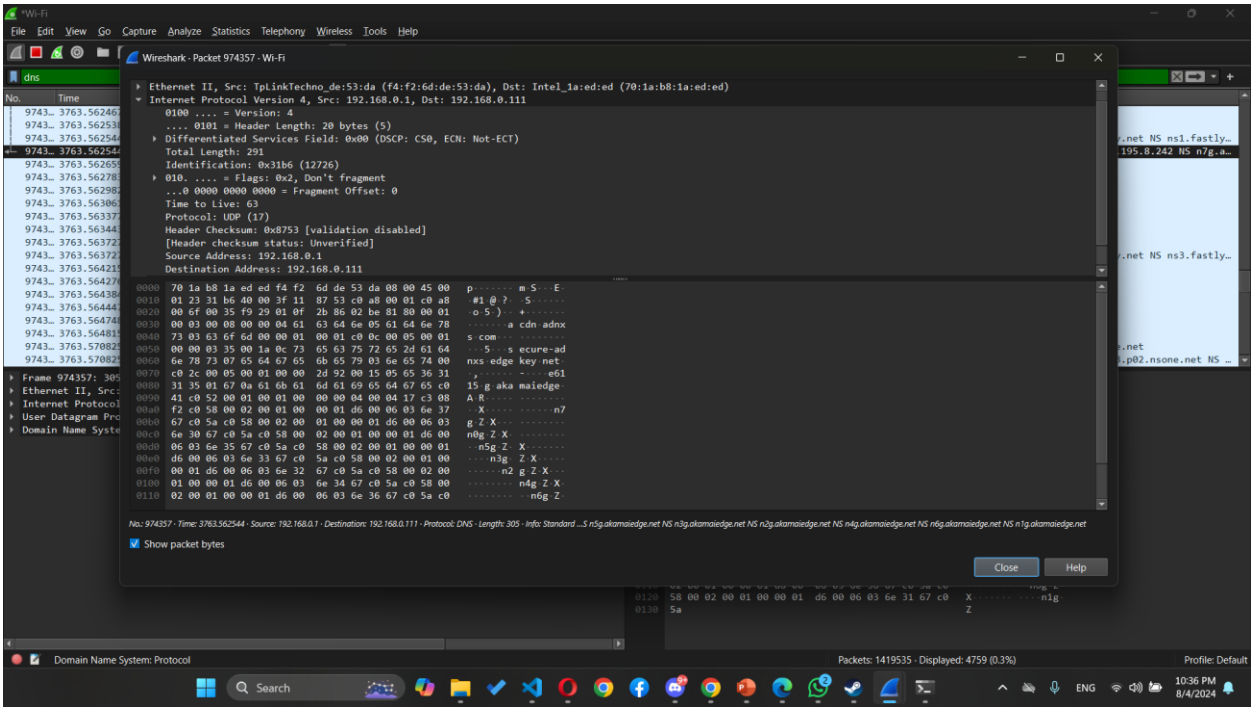


Figure 12: details for the source and destination for it

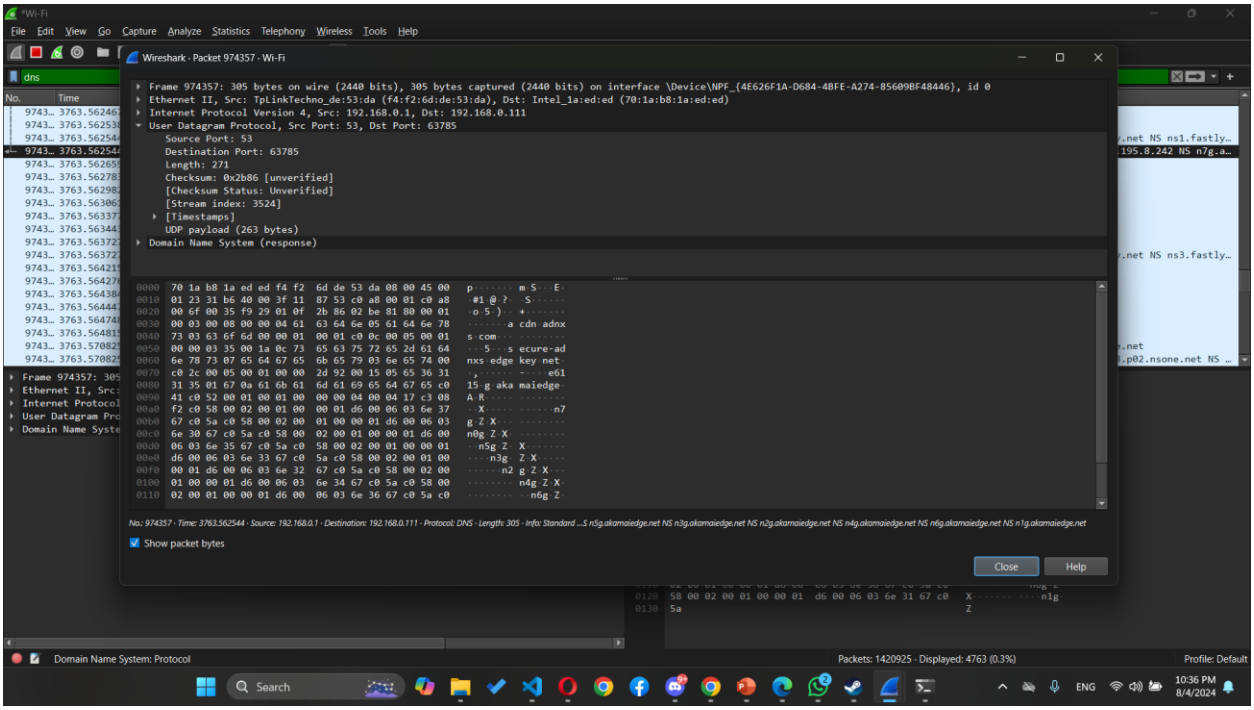


Figure 13: user datagram protocal

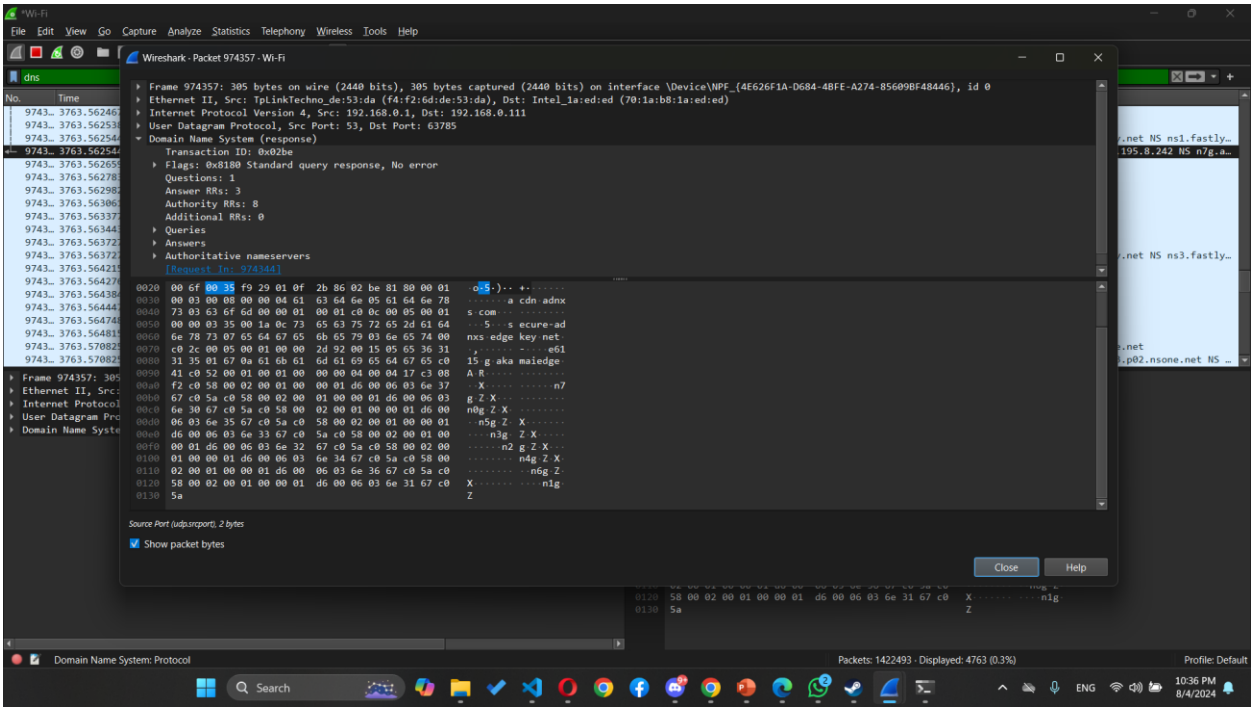


Figure 14: domain name system

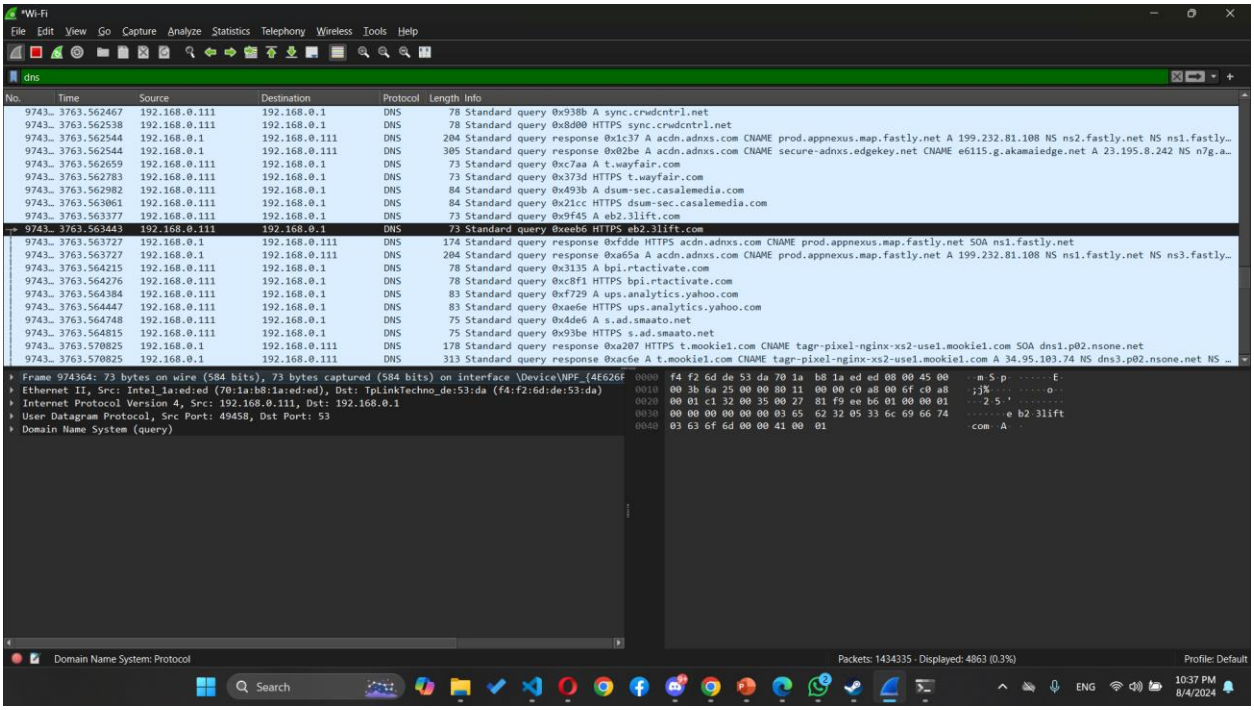


Figure 15: The packet that we show the details about it

- Capturing DNS Messages:
- 1. Start Wireshark: Open Wireshark, choose your network interface, and begin capturing packets.
- 2. Filter DNS Traffic: Type `dns` in the display filter to show only DNS packets.
-
- Examining DNS Queries:
- Query Packets: Displays DNS queries where the client requests the IP address for a domain name from the DNS server.
-
- Analyzing DNS Responses:
- Response Packets: Shows DNS responses containing the IP address corresponding to the requested domain.
-
- Detailed Packet Analysis:
- Packet Details: Click on a DNS packet to see details such as:
- Transaction ID: Matches queries to responses.
- Flags: Indicates if the packet is a query or response and other info.
- Questions: Domain name requested.
- Answers: IP address provided by the server.
- Interpreting Data:
- Hex Data: The raw packet data is shown in hexadecimal at the bottom, with Wireshark translating it for easier understanding.
- Common DNS Operations:
- Standard Query: Client request for a domain's IP address.
- Response: DNS server's reply with the resolved IP address.

Task 2:

Part 1:

- The Server Code:

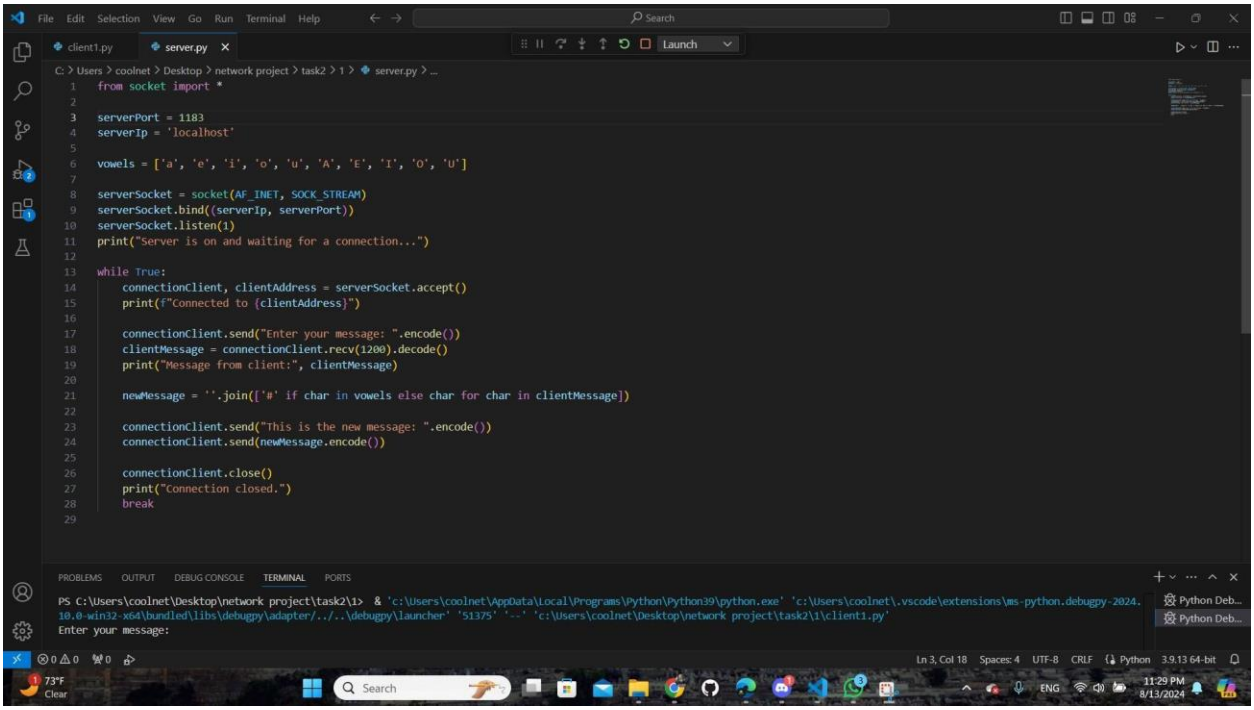


Figure 16: server code

The server code simply opens a connection and waits for only one client. After that, the server receives a message from the client, converts the upper and lower case vowels to #, then sends it back to the client and stops the connection and print message about this.

- The Client Code:

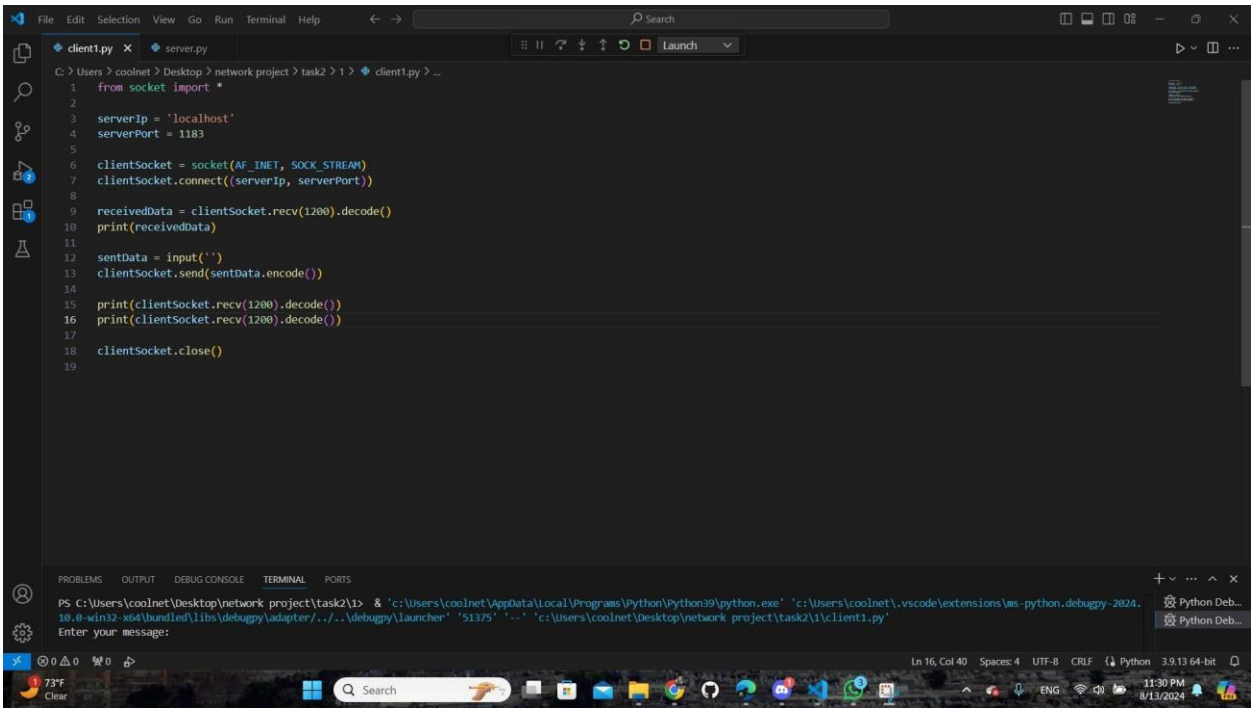


Figure 17: client code

The client code simply sends a connection request to the server. After the server accepts the connection, the client sends a message to the server, which the server returns in the new format, prints it, and then disconnects from the server.

- The First Server Output Message:

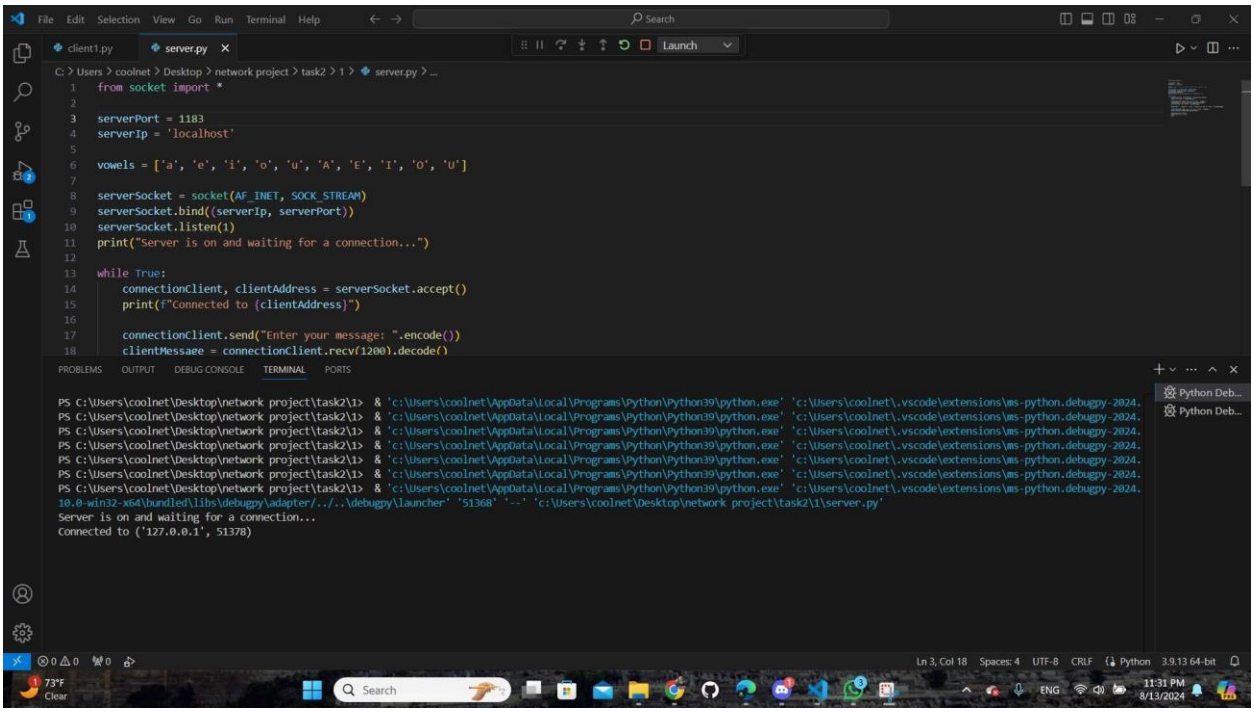


Figure 18: first server output message

The server prints a message that it is up and running and is waiting for a client to connect to it (one client, no more). After connecting to that client, it prints the client's IP and its port to confirm that it is connected.

- The Client Output Message:

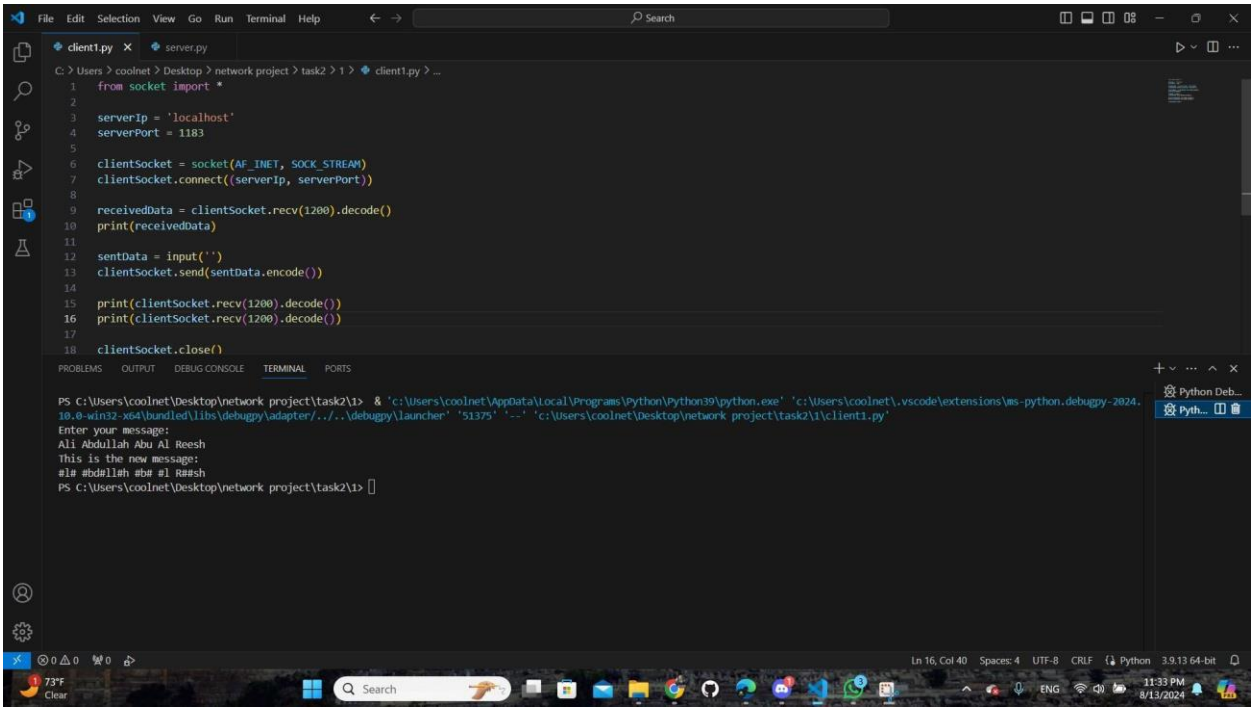


Figure 19: Client output message

After connecting to the server, the client is asked to enter a sentence to be sent to the server, which in turn converts the vowels to # and then returns it to the client, and the final sentence is printed as we see above.

- The Second Server Output Message:

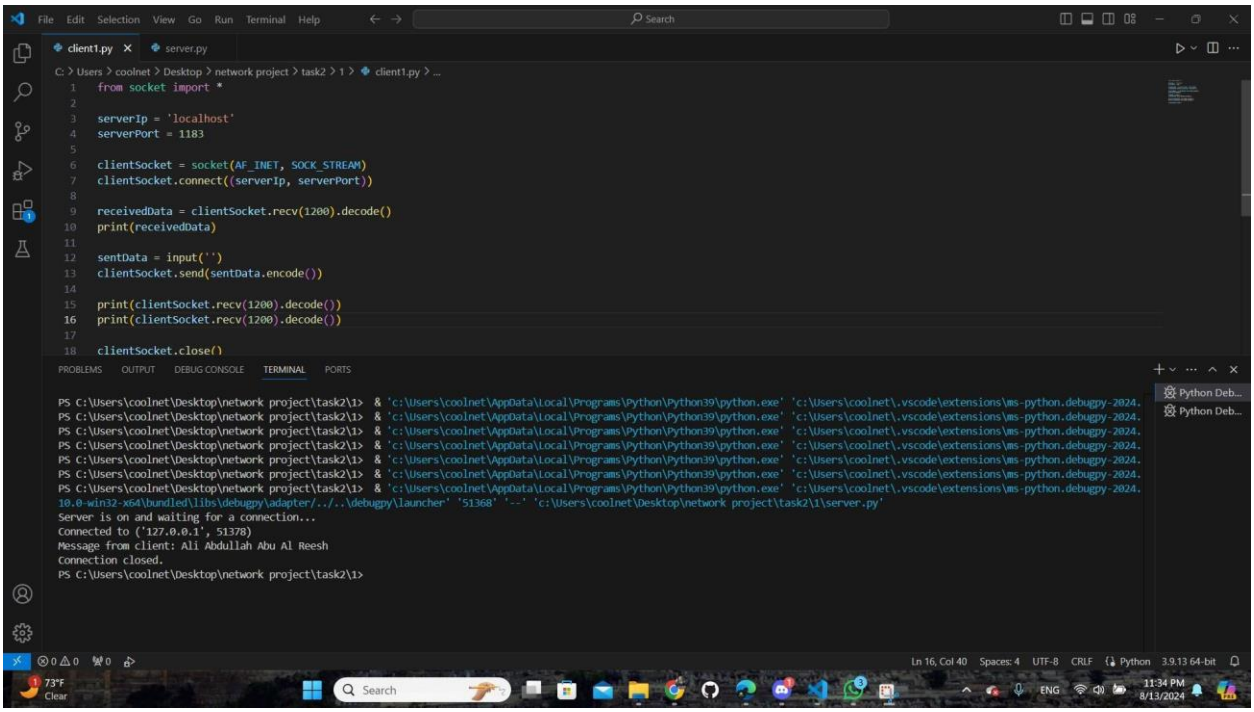


Figure 20: Second server output

In this image, the server shows the sentence it received from the client by printing it, then performs the necessary operations on it, sends it, then stops the connection and prints the sentence "Connection closed."

Part 2:

- The Server Code:

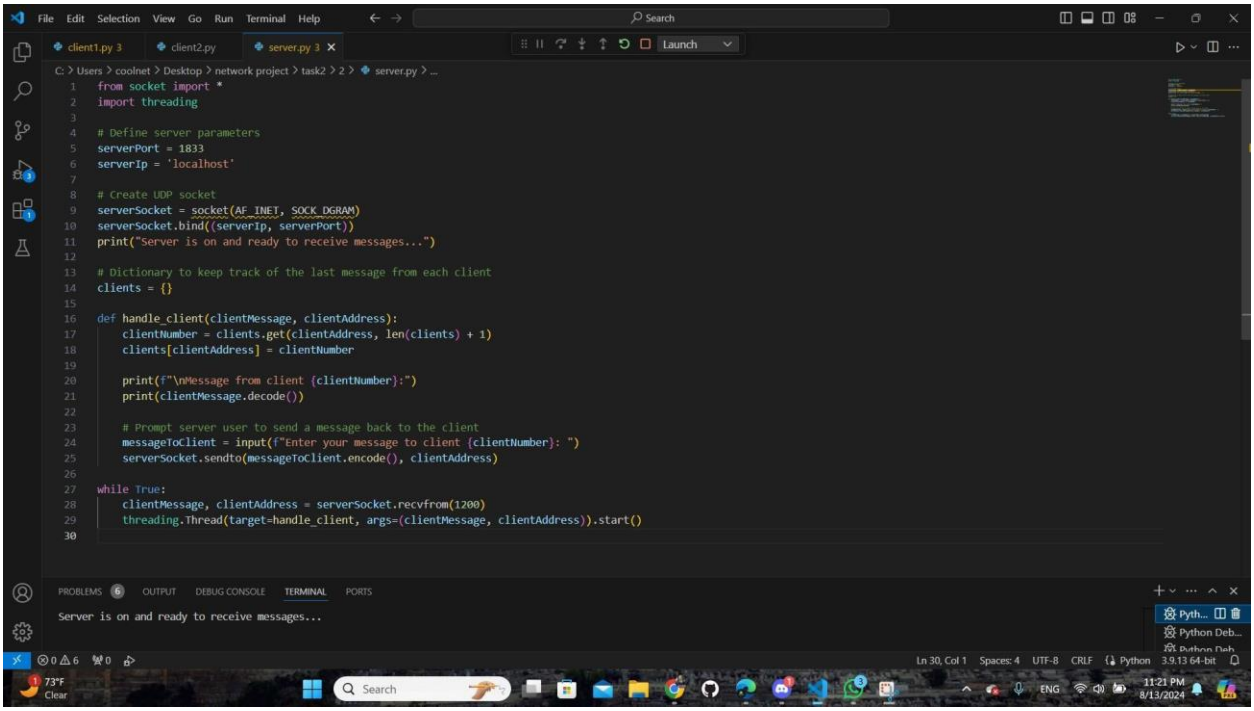


Figure 21: server code for part2

This image shows the server code, which is different from the previous task, because in this task the server will open a connection with the rest of the clients, and the server can respond to messages (it works as a client, not a server).

- The Client1 && Client2 Code:

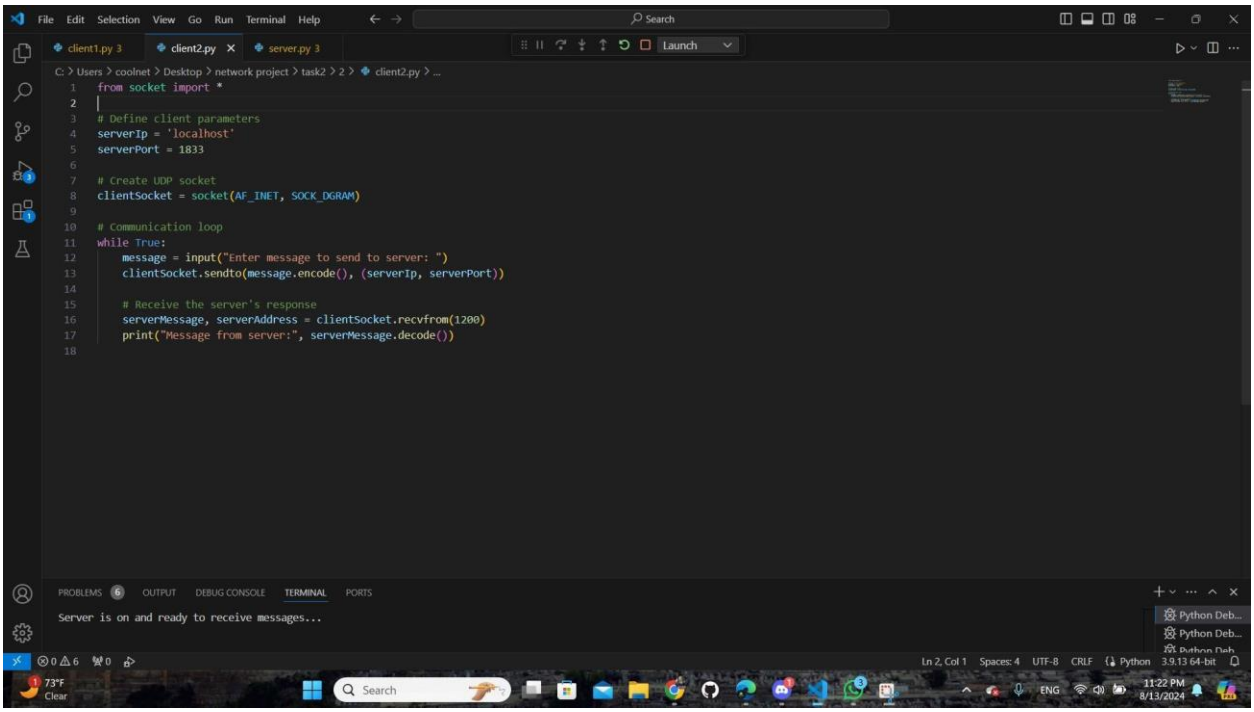


Figure 22: client 2 code

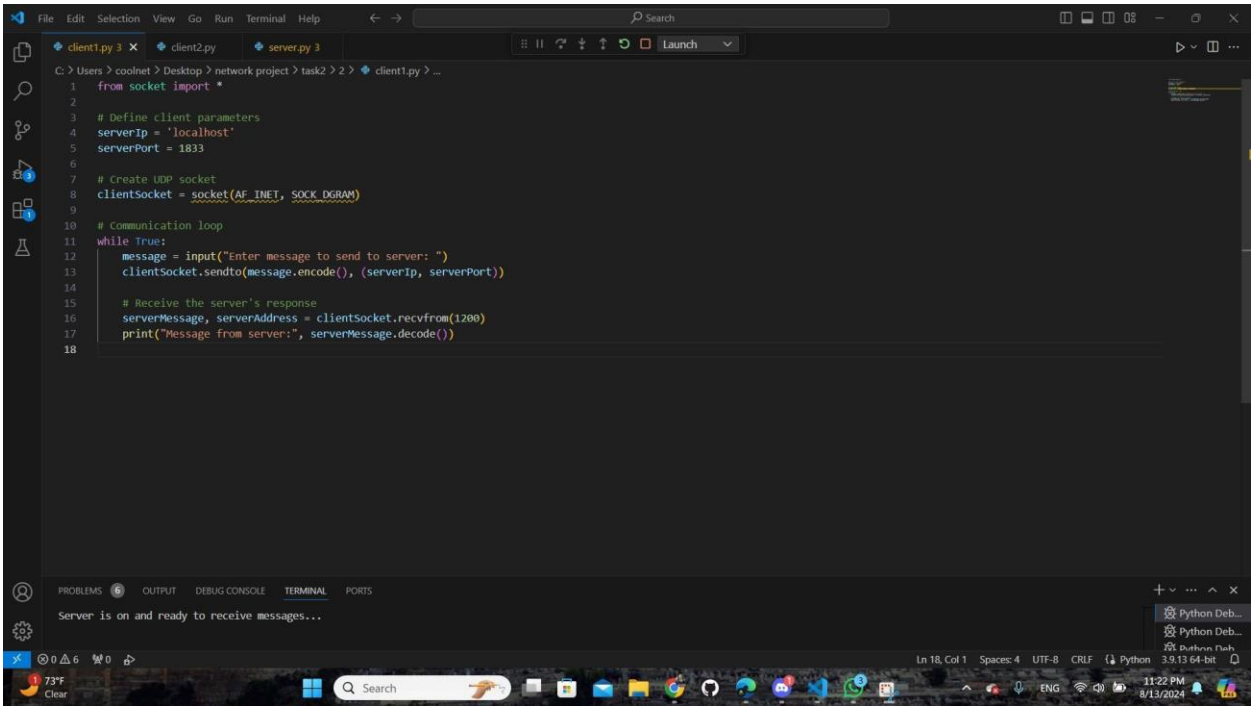


Figure 23: client 1 code

At the same time, the client, after connecting to the server, can send and receive (it works as a server and a client at the same time), and the client cannot send to other clients.

- The First Server Output Message:

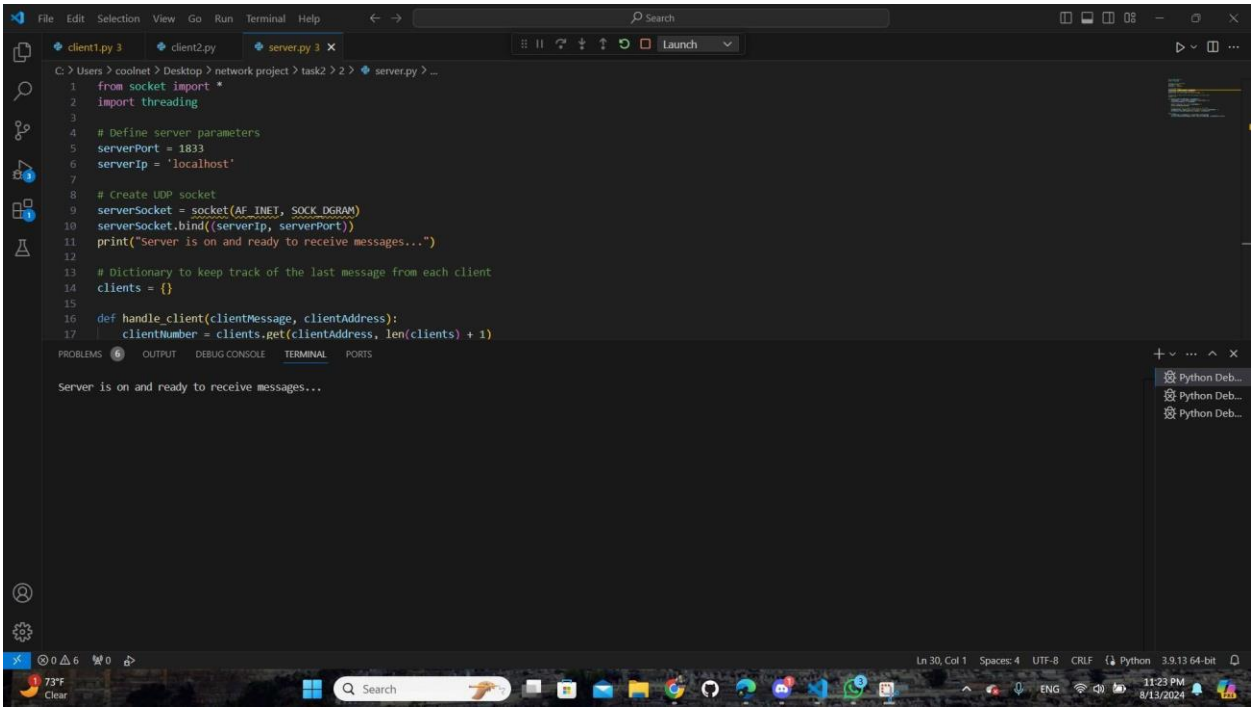


Figure 24:starting the server

After the server is started and configured to connect to more than one client, it prints a message that it is ready to connect with clients.

- The Client Output:

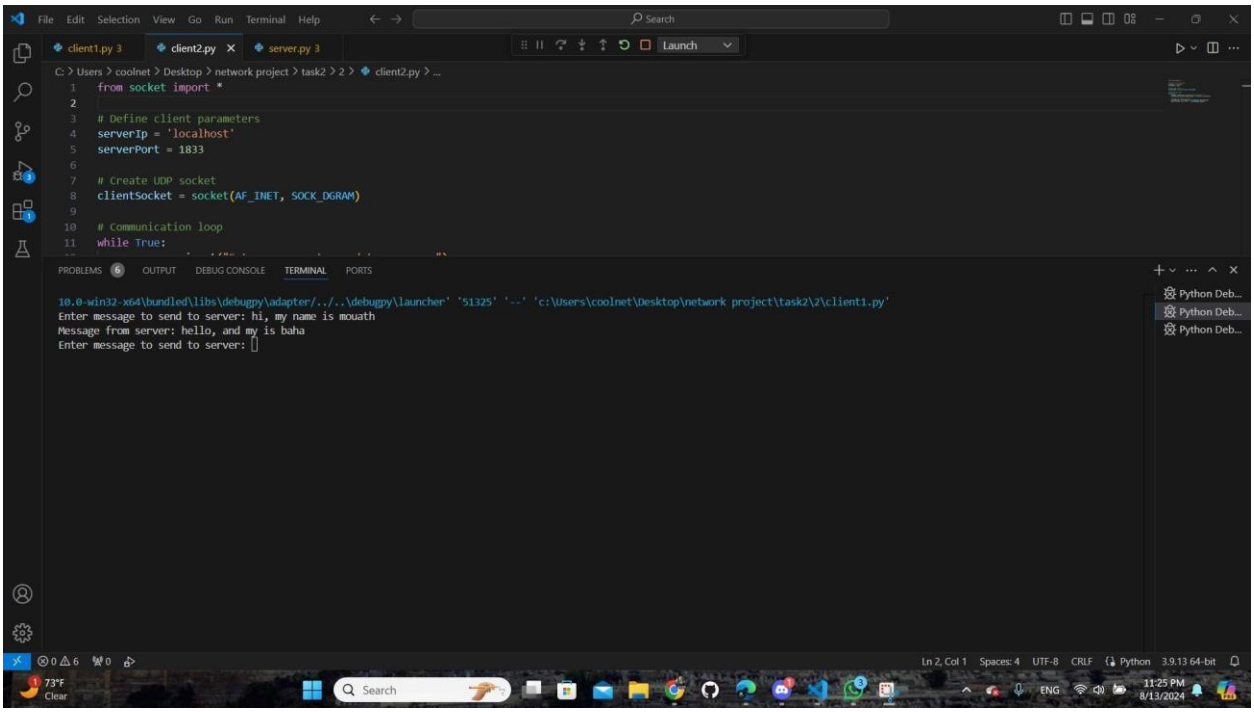


Figure 25: client 1 message

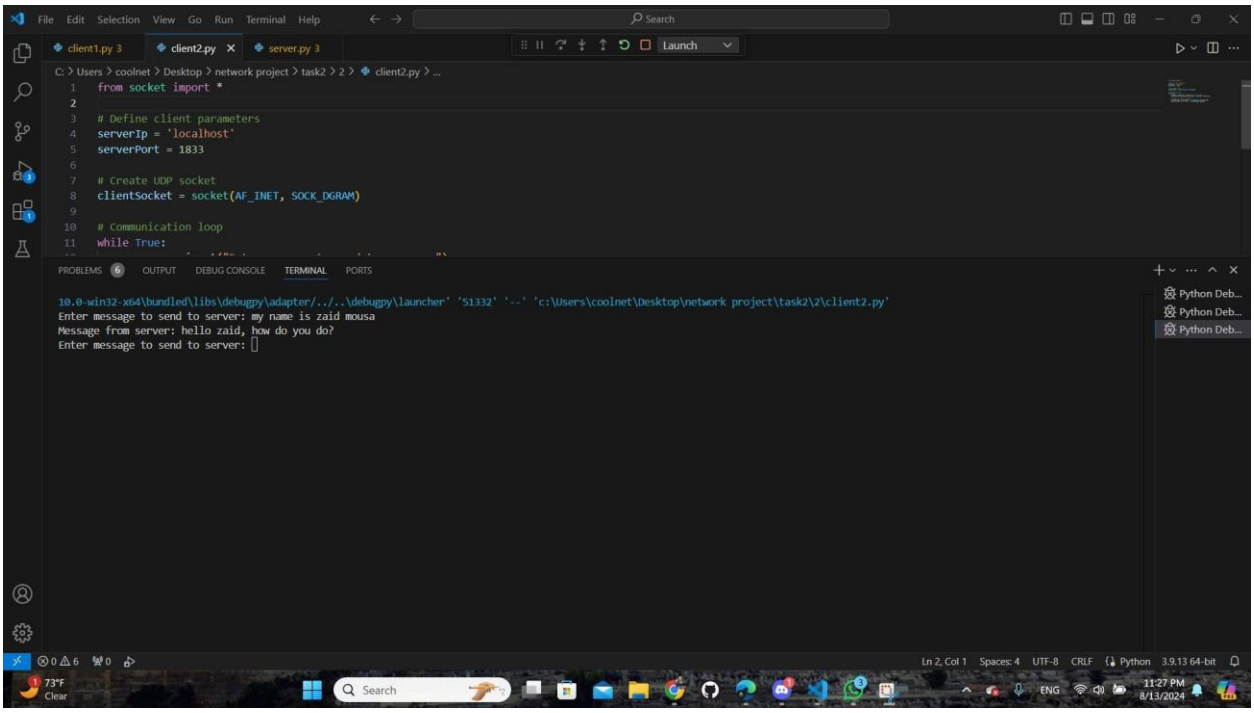


Figure 26: client 2 message

After the client connects to the server, it requests the server's message, and when it arrives there, the server responds to the message (exchanging messages like Messenger, etc.).

- The Second Server Output Message:

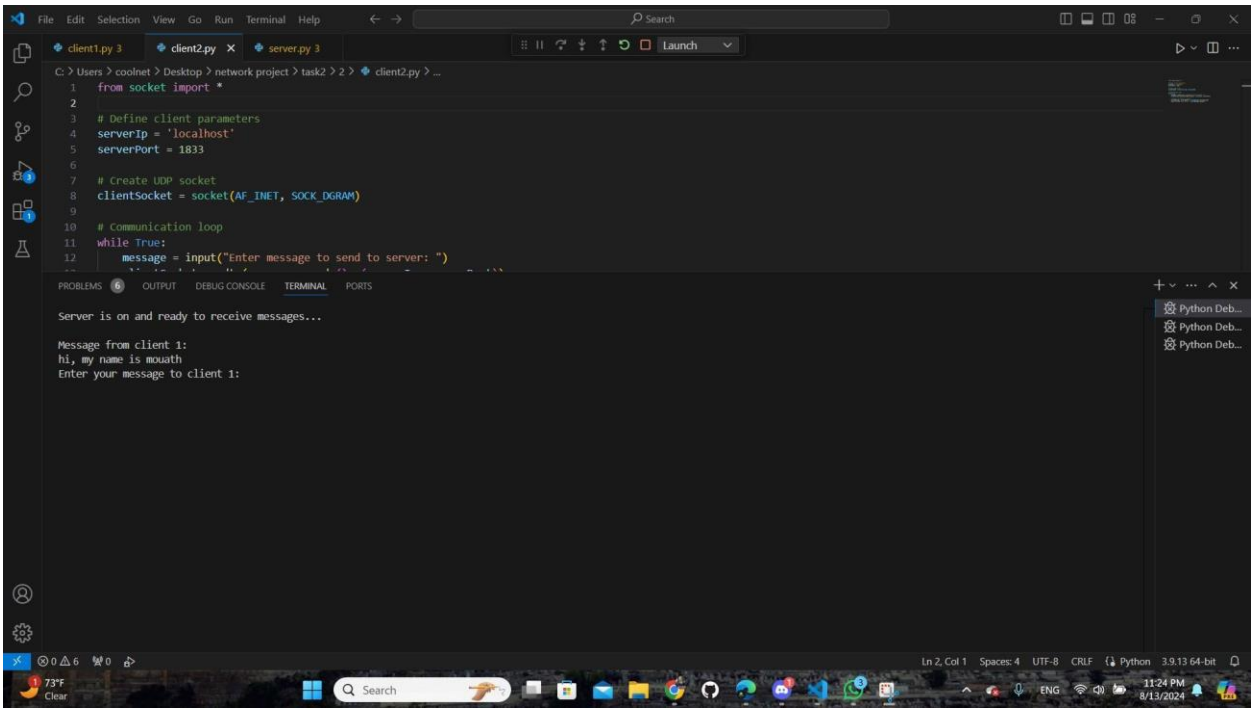


Figure 27: server receive client 1 message

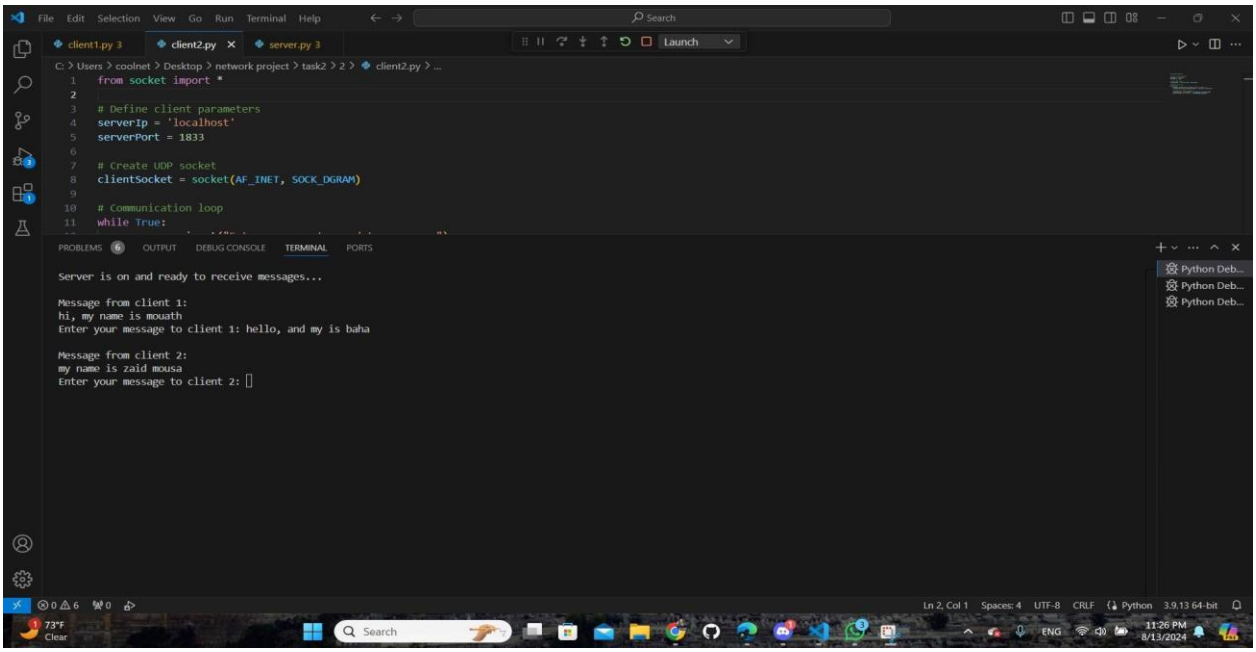


Figure 28:server receive the client 2 message

After the message arrives from the client, it is printed who is the client who sent the message and what is the message sent, and he can reply to the sent message, but the client who sent the message cannot resend it unless the server replies to him.

Task 3:

1-

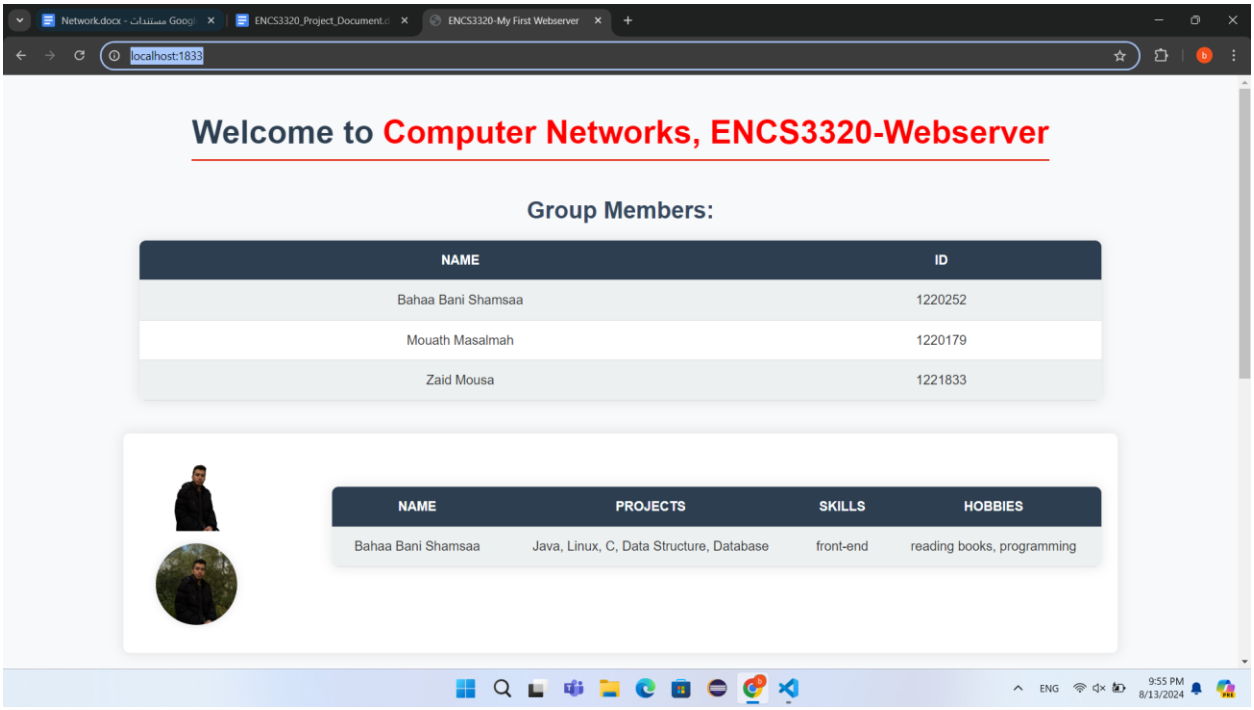


Figure 29:request /

As we see here when I type link http://localhost:1833/ it goes to the english page

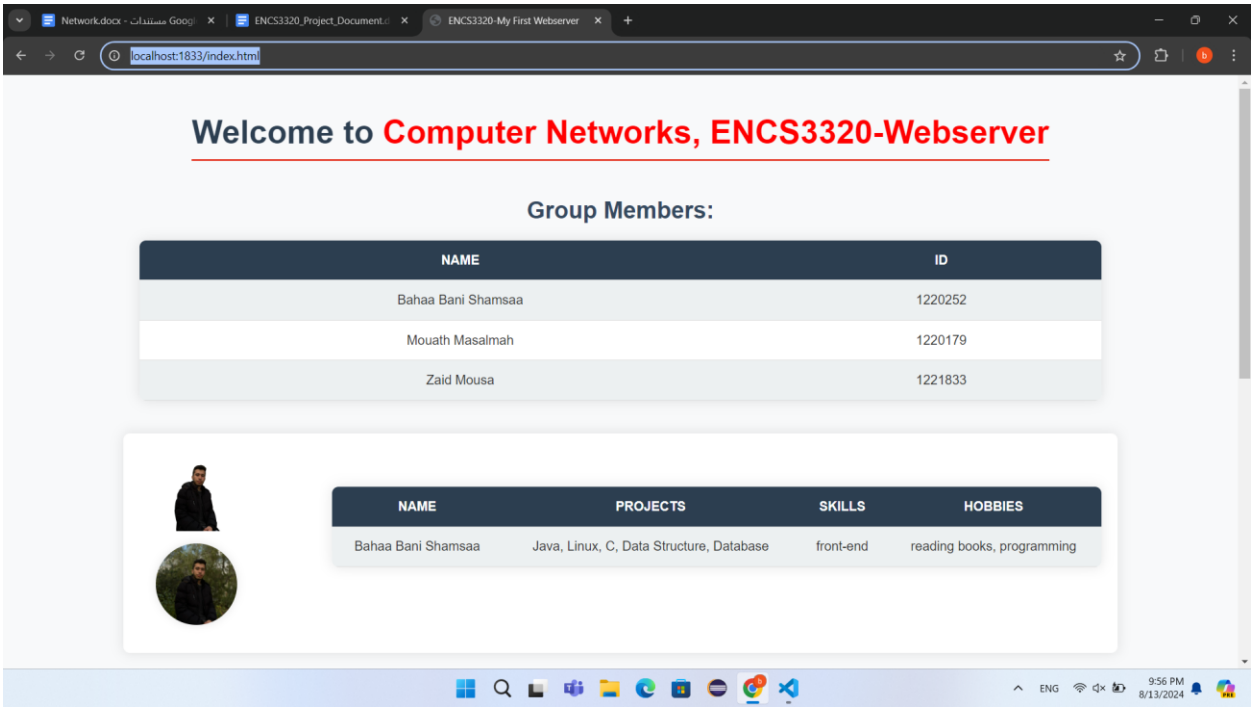


Figure 30: index.html request

As we see here when I type link http://localhost:1833/index.html it goes to the english page

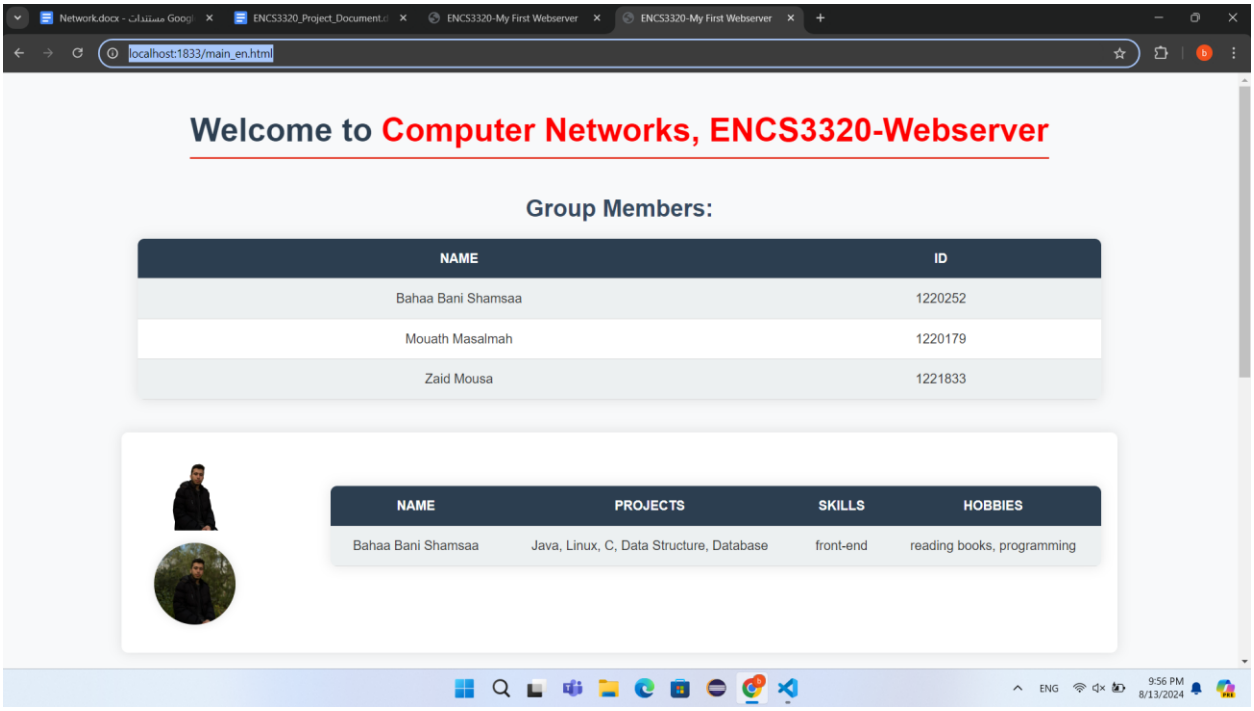


Figure 31: main_en.html request

As we see here when I type link http://localhost:1833/main_en.html it goes to the english page

a.

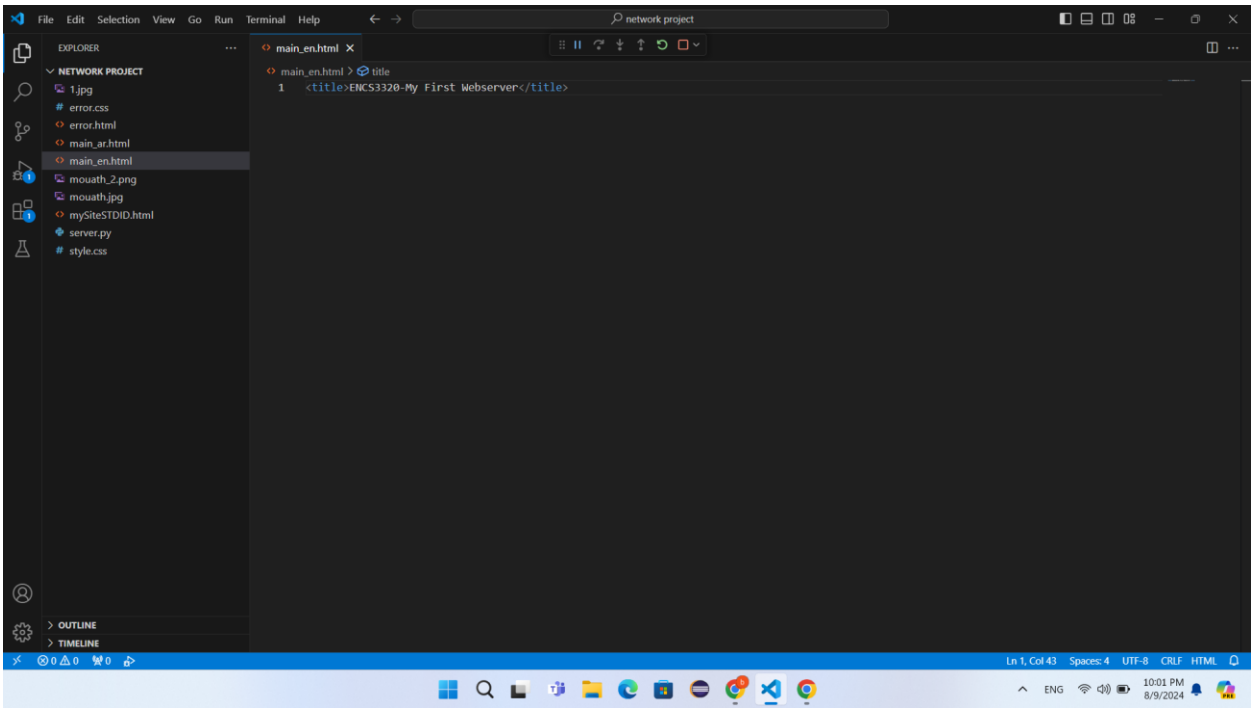


Figure 32: title of the page

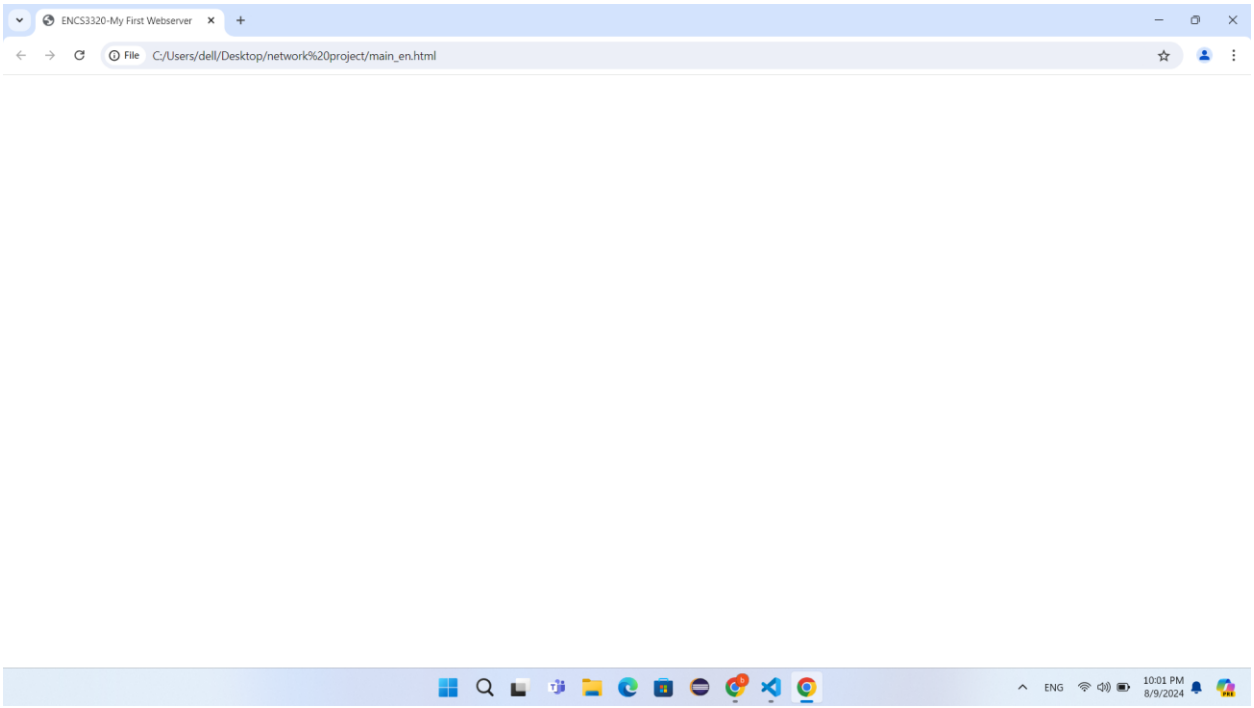


Figure 33:browser title

Here we put the title ENCS-My First Webserver

b.

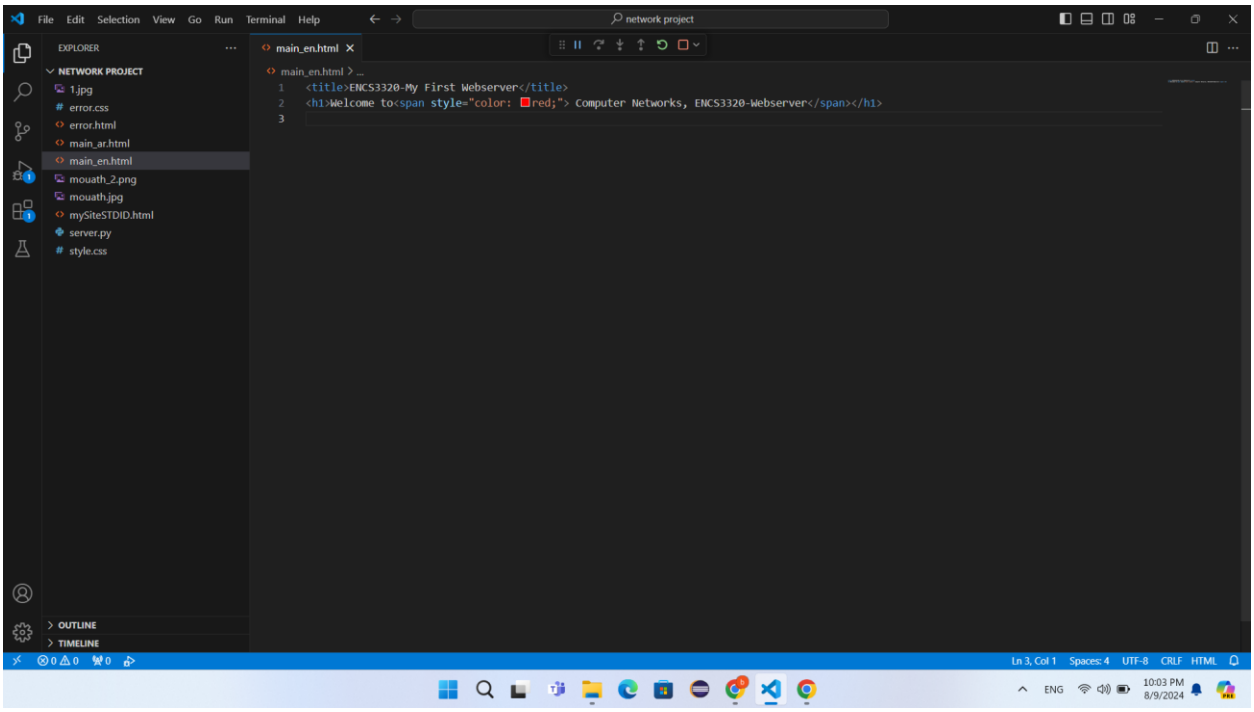


Figure 34: welcome message

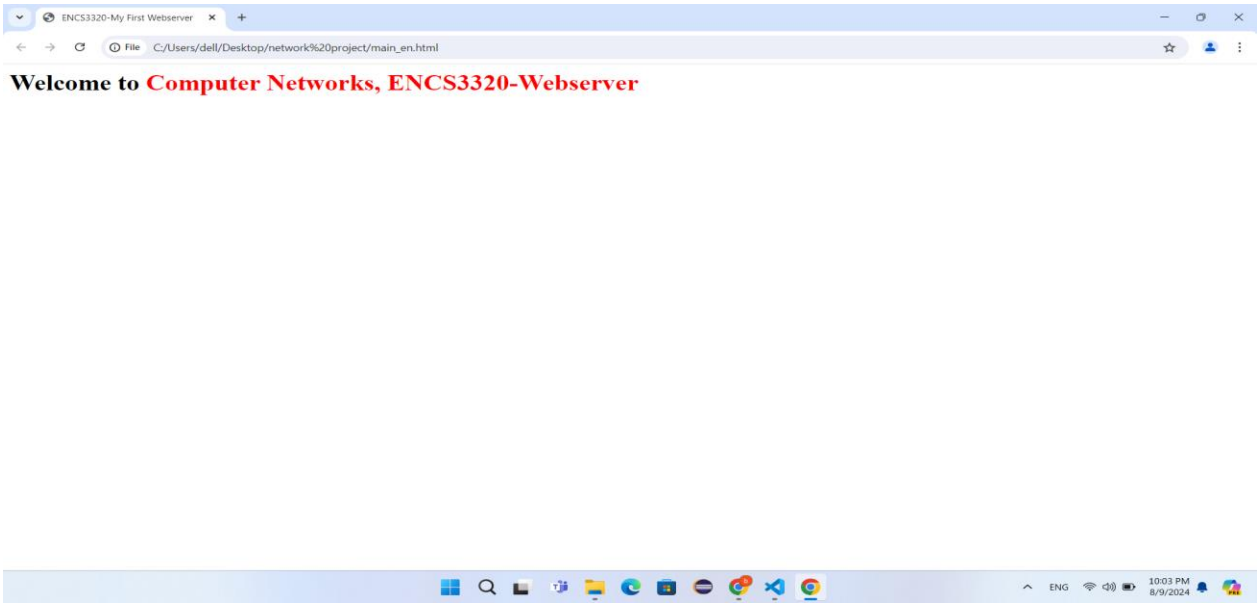


Figure 35:browser welcome

Here I use span to make the words with different colors

c.

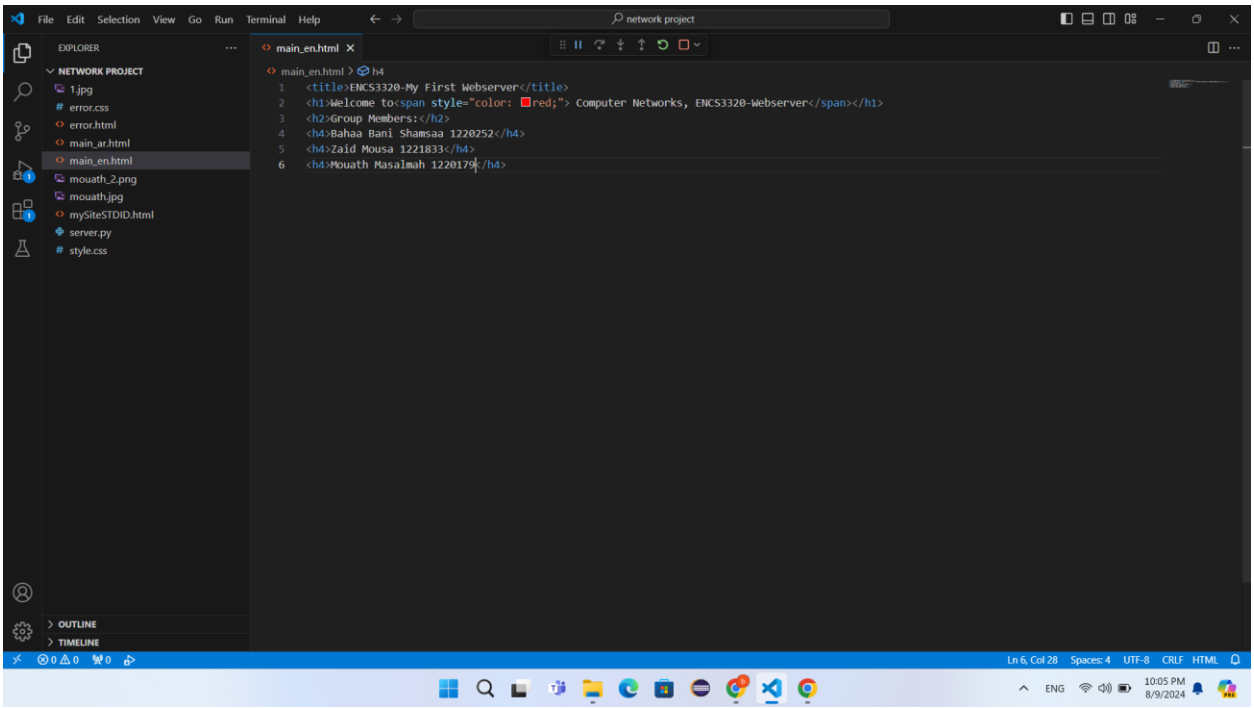


Figure 36:group members

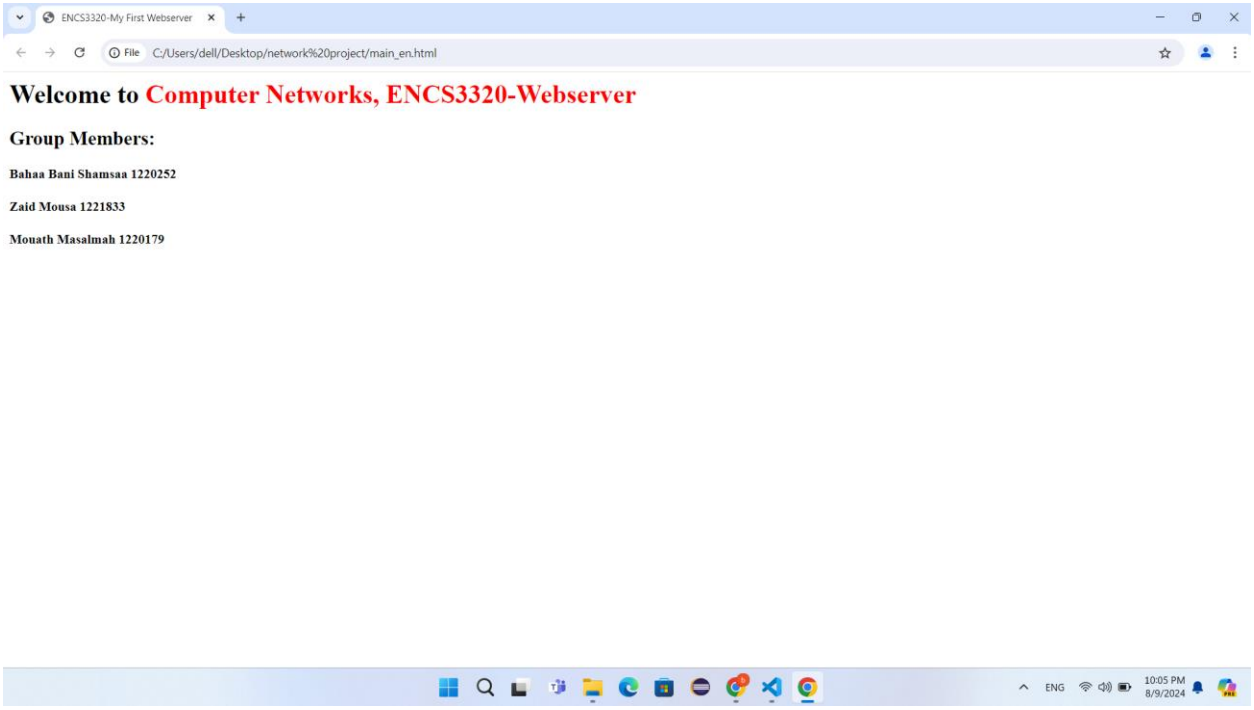


Figure 37: browser group members

Here we type the group members names with the numbers

d.

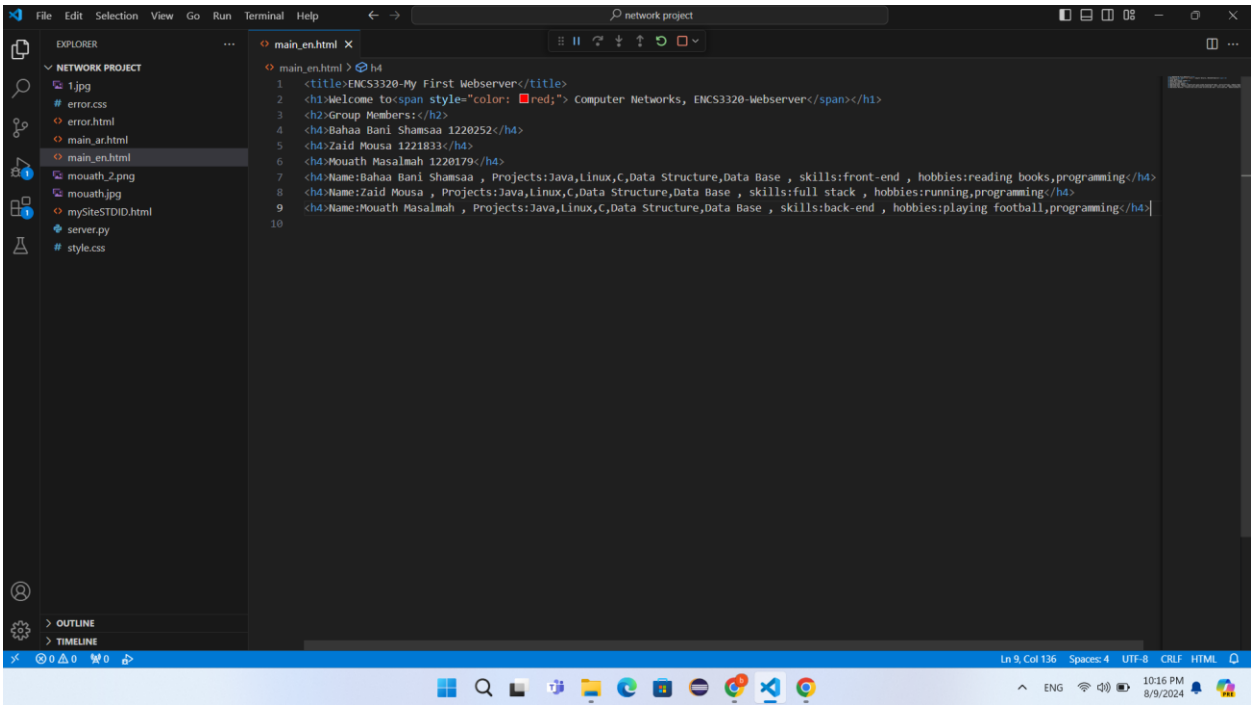


Figure 38: info about group members

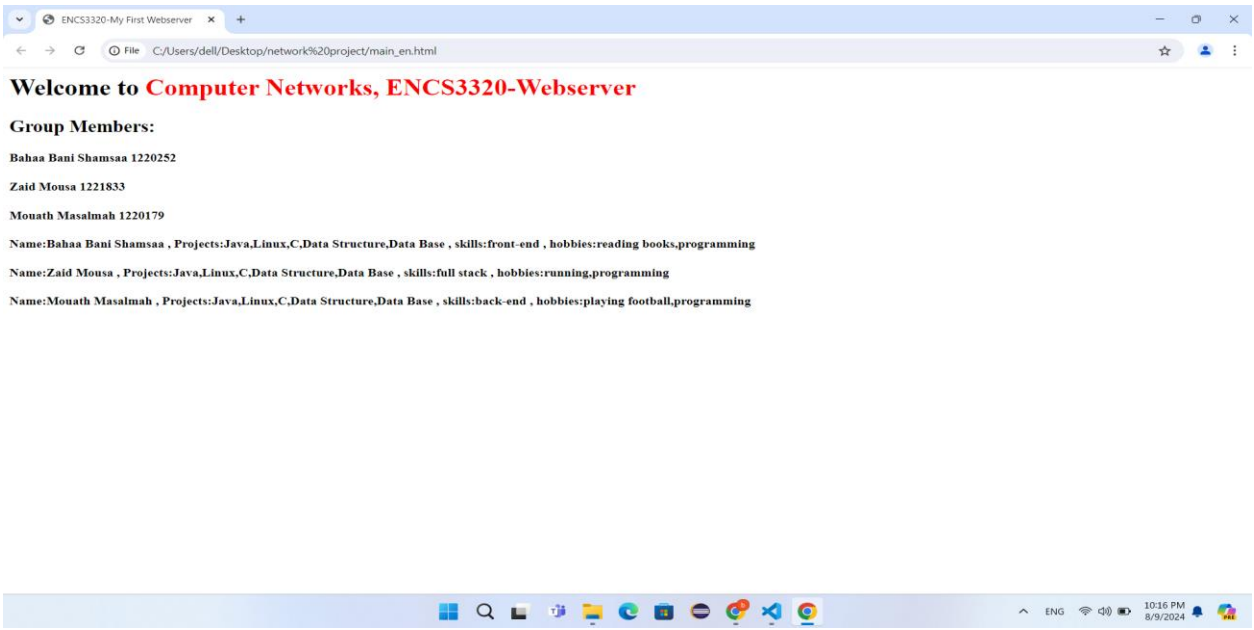


Figure 39:browser info group members

Here we type the names, projects, skills and hobbies

e and f:

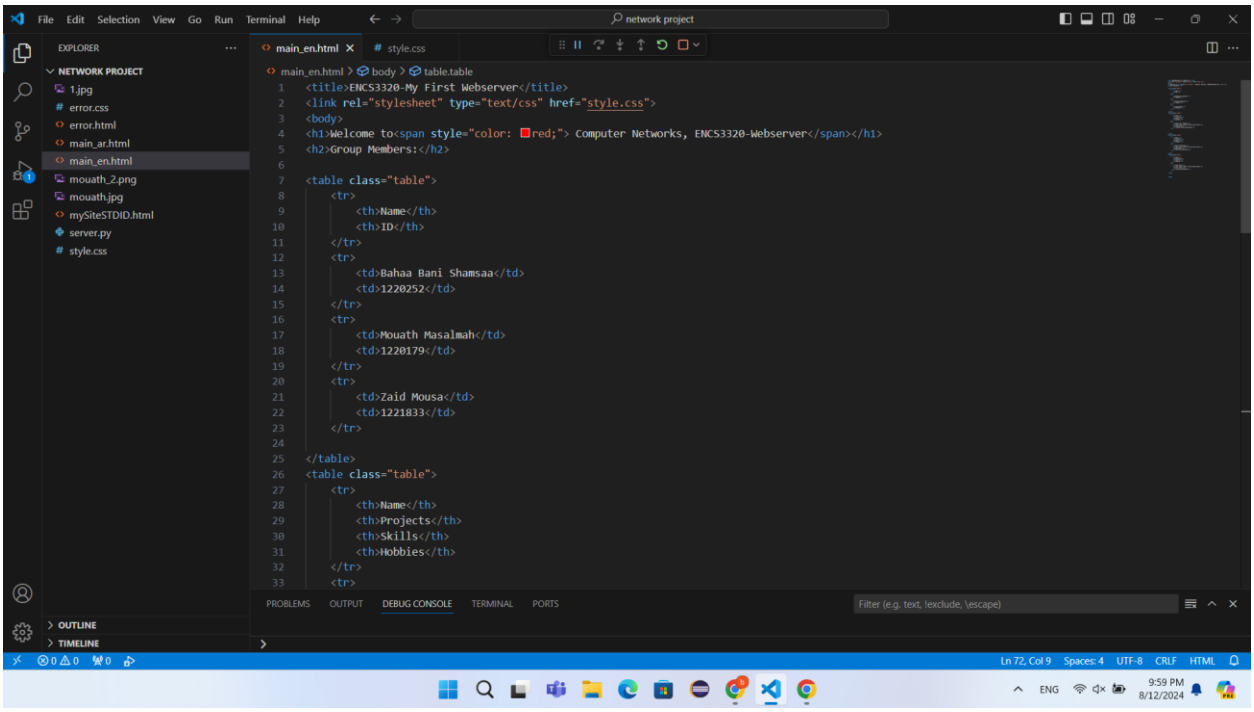


Figure 40: table 1

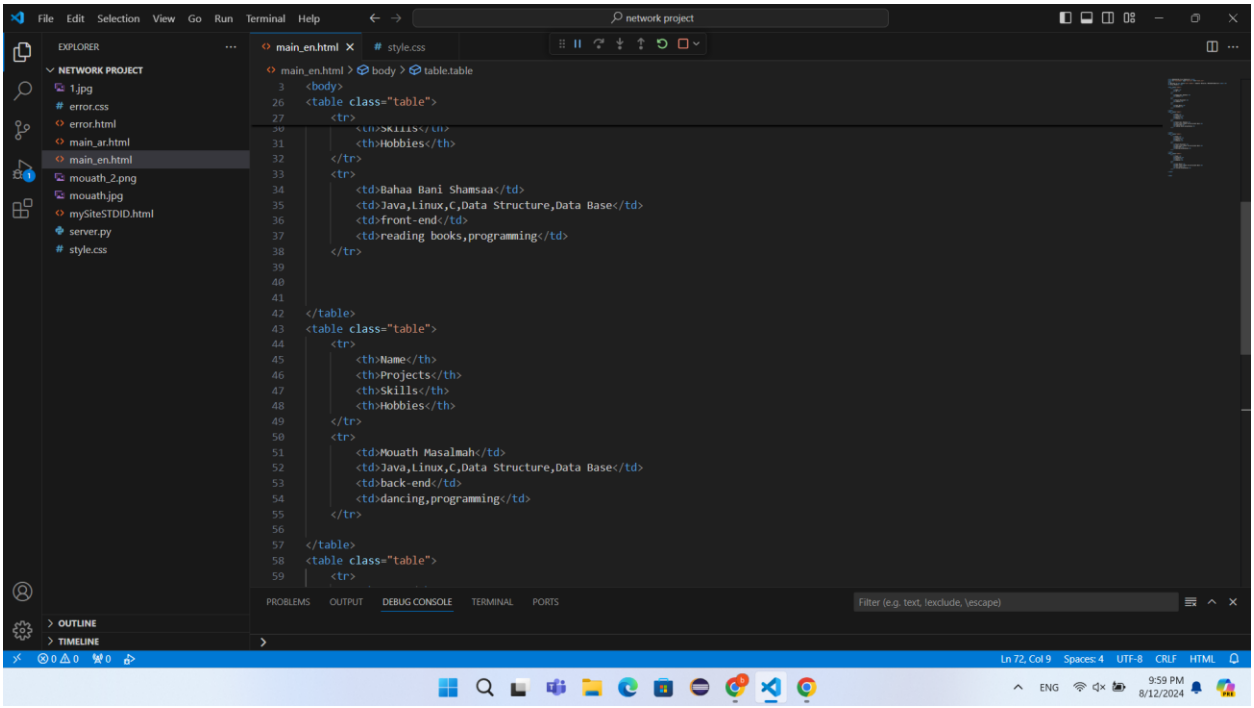


Figure 41: table 2

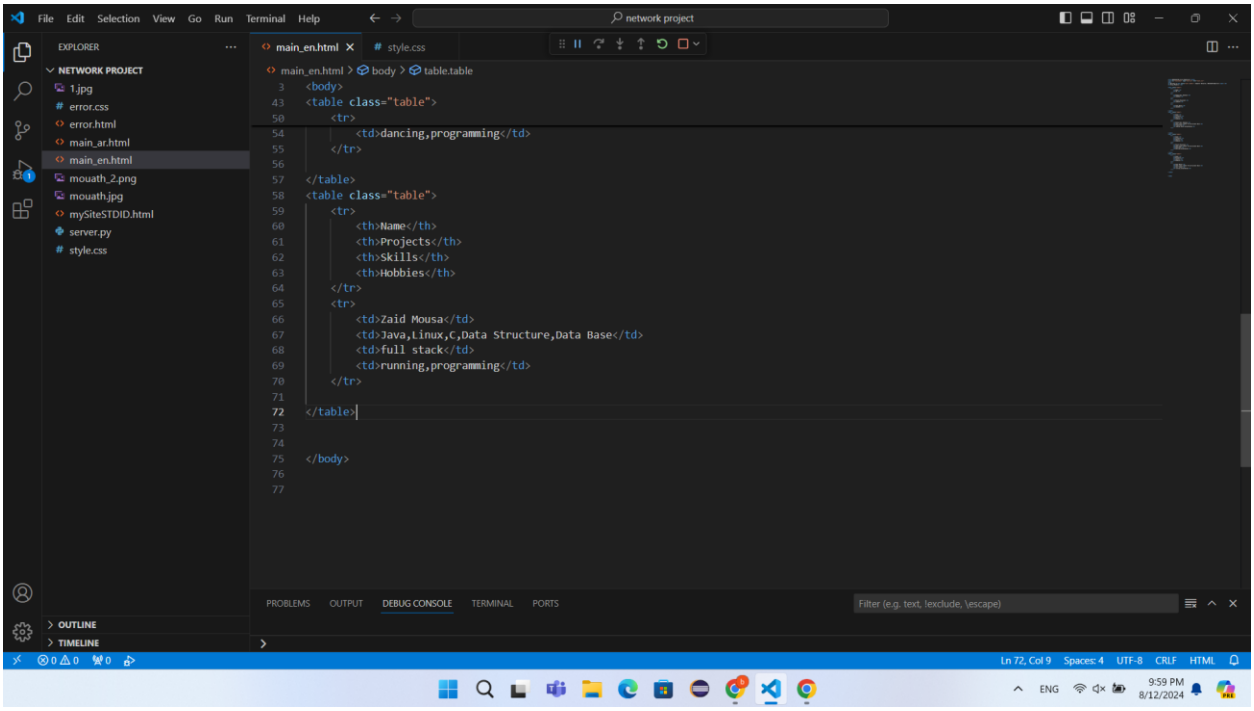


Figure 42: table 3

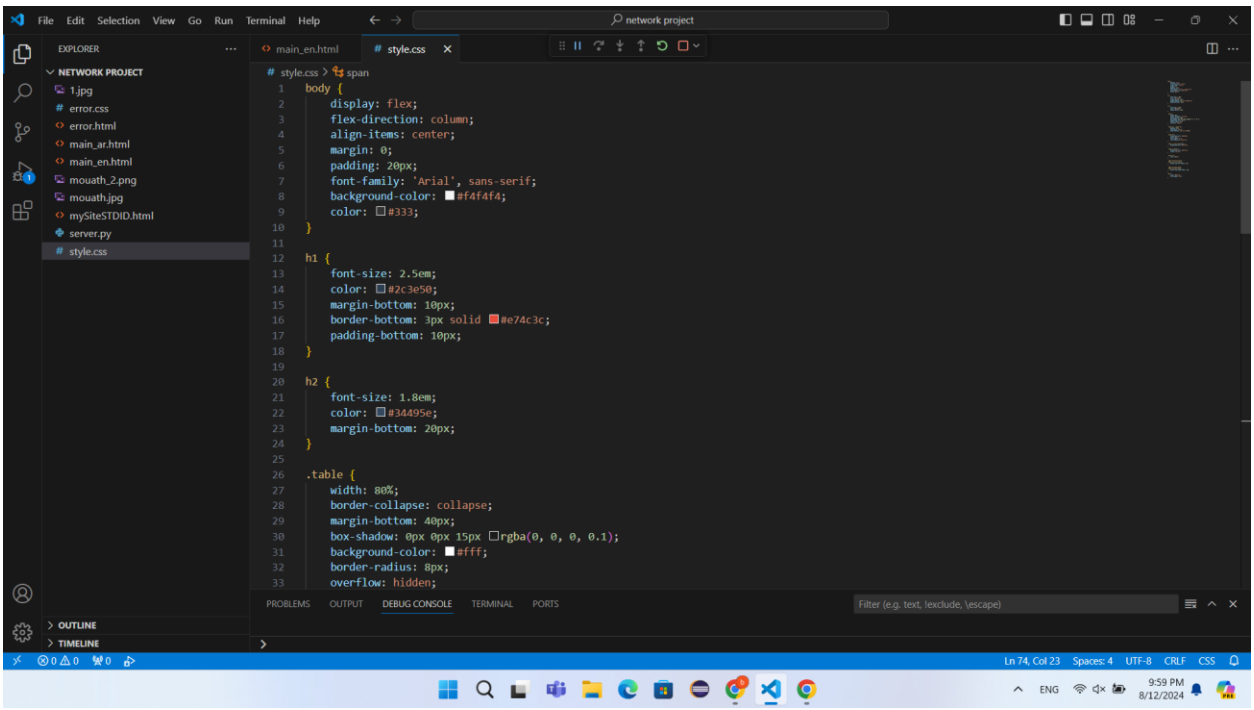


Figure 43: css for table 1

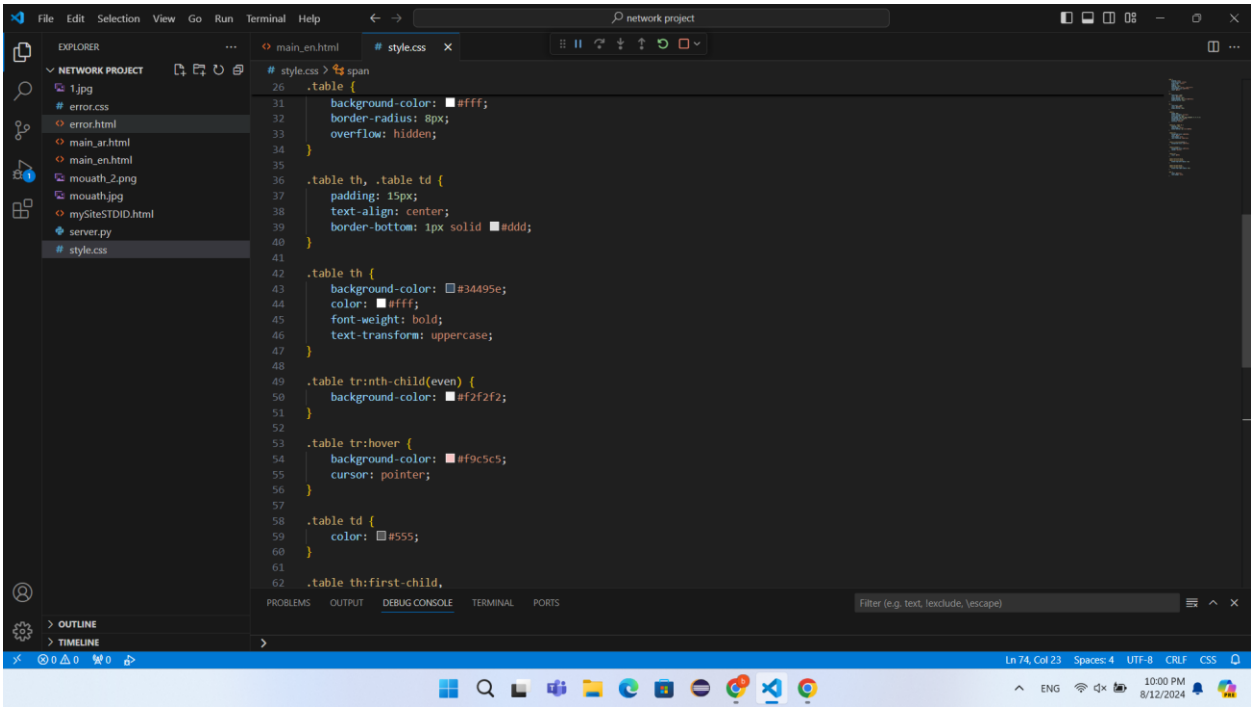


Figure 44: css for table 2

2

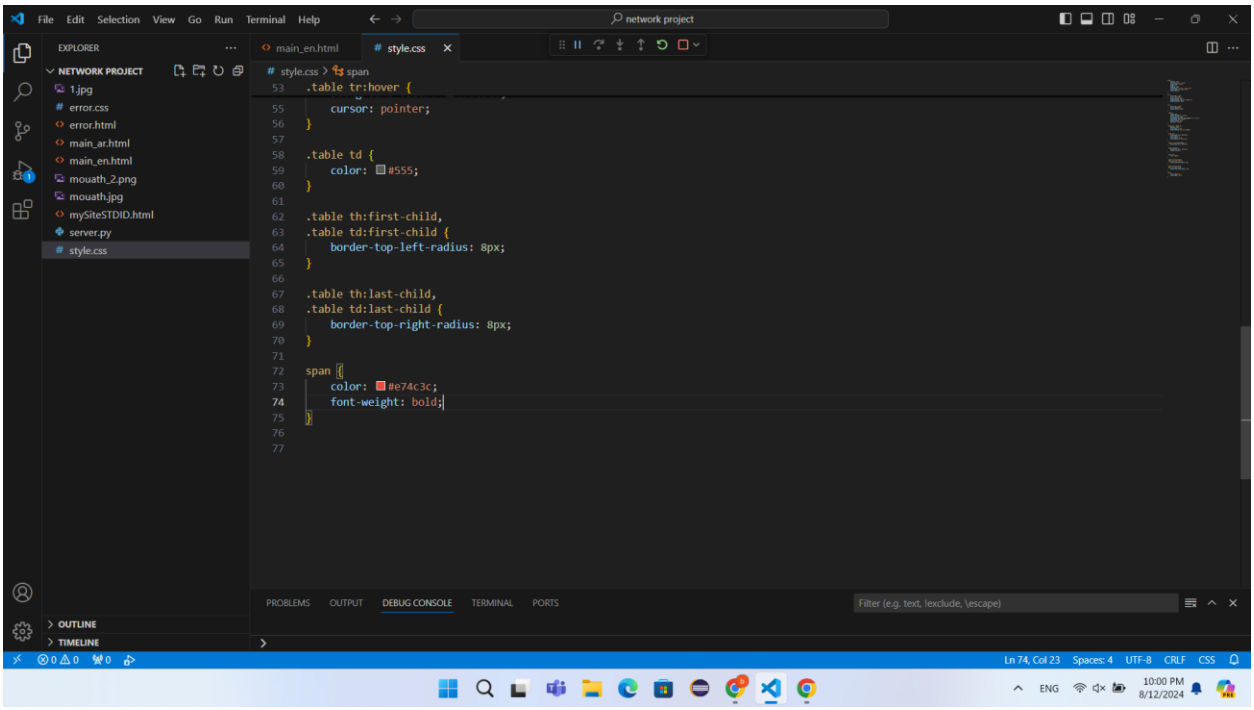


Figure 45: css for table 3

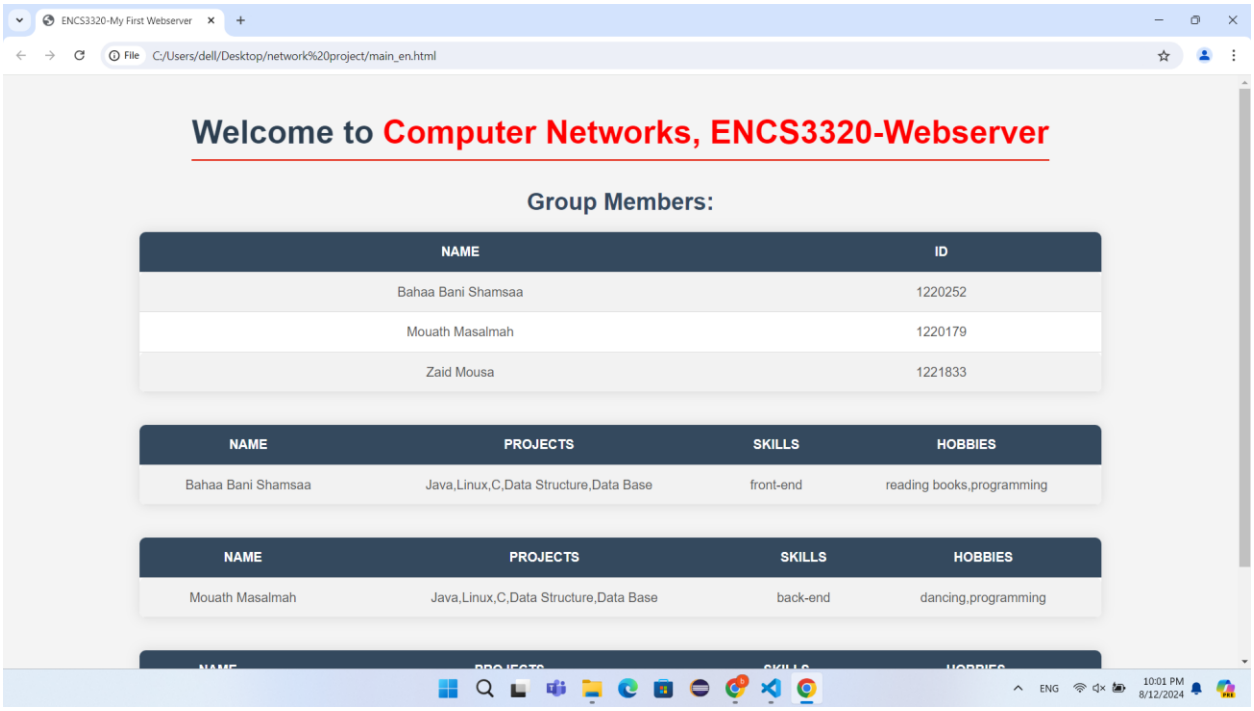


Figure 46:browser for table 1

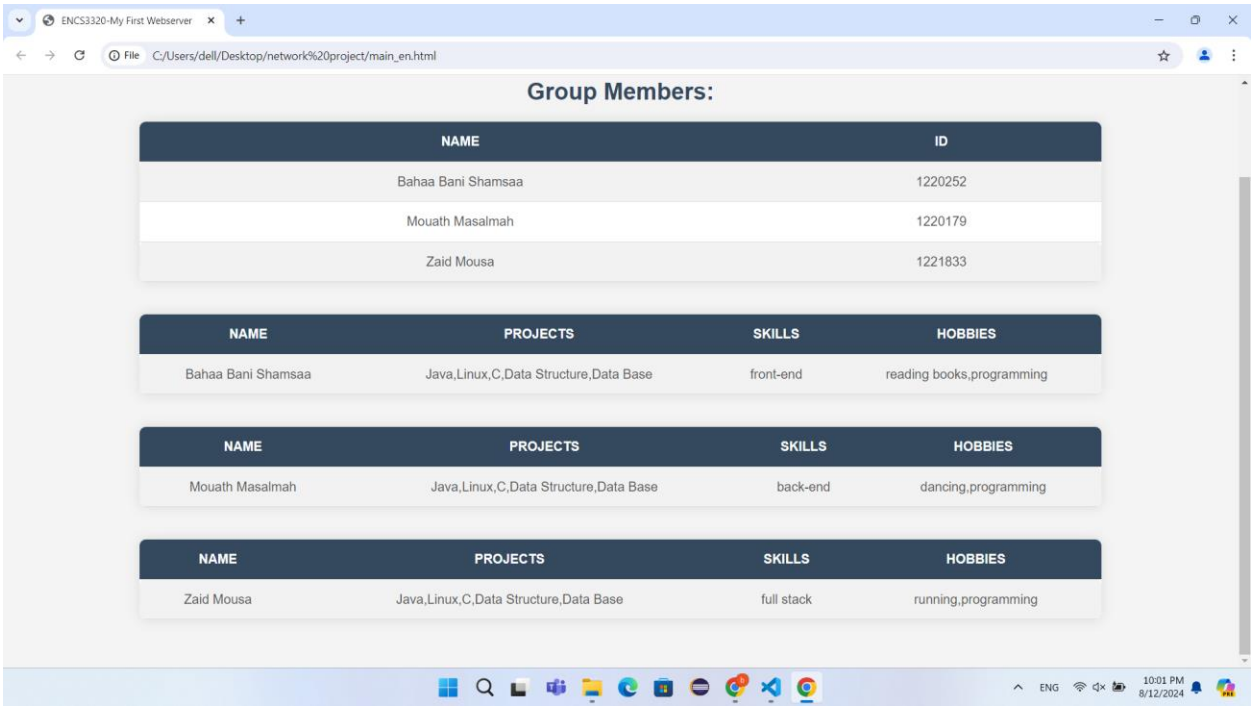


Figure 47: browser for table 2

Here we divide the group members names and IDs in table and each member with his information in tables, and we use a good CSS to make the page look nice, we use things like hover, background, border

h.

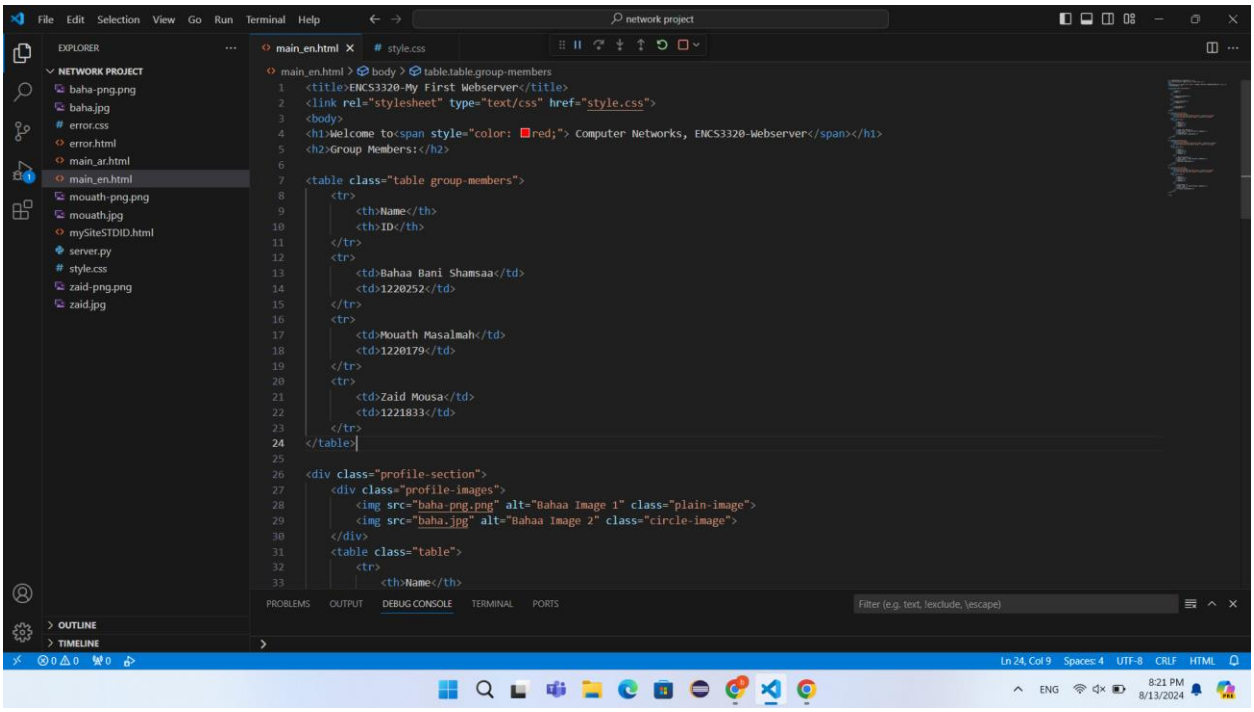


Figure 48:png and jpg 1

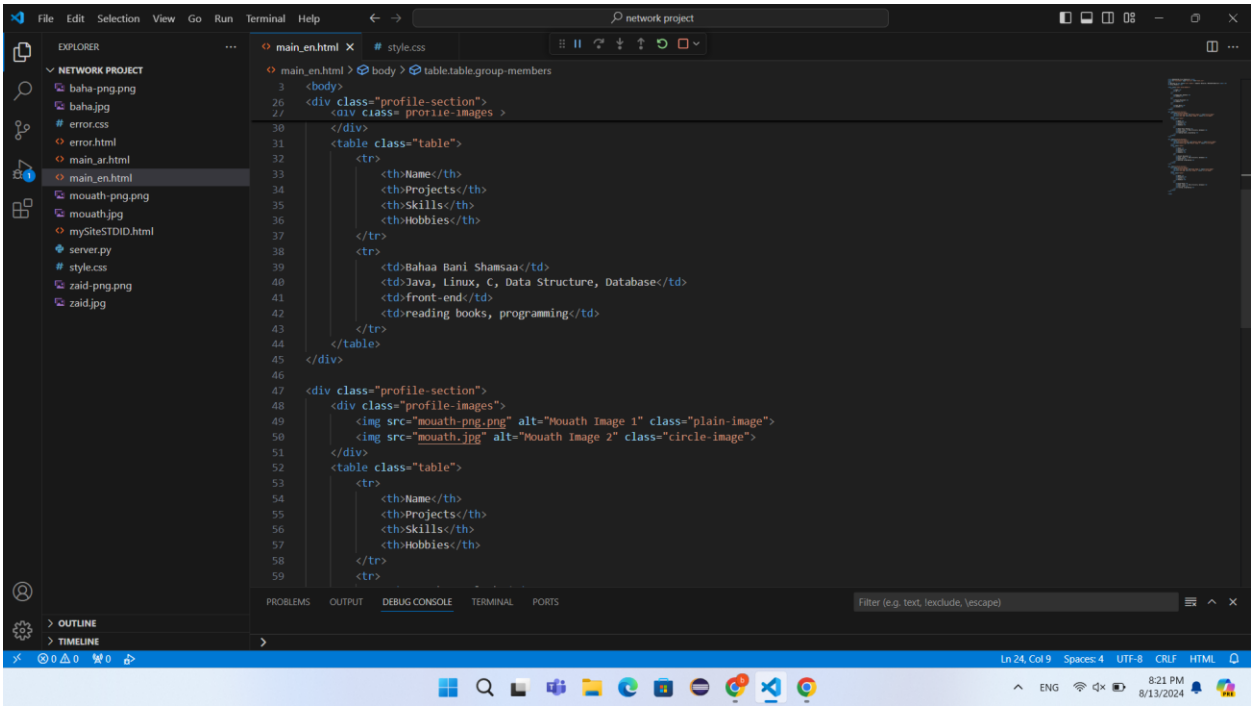


Figure 49:png and jpg 2

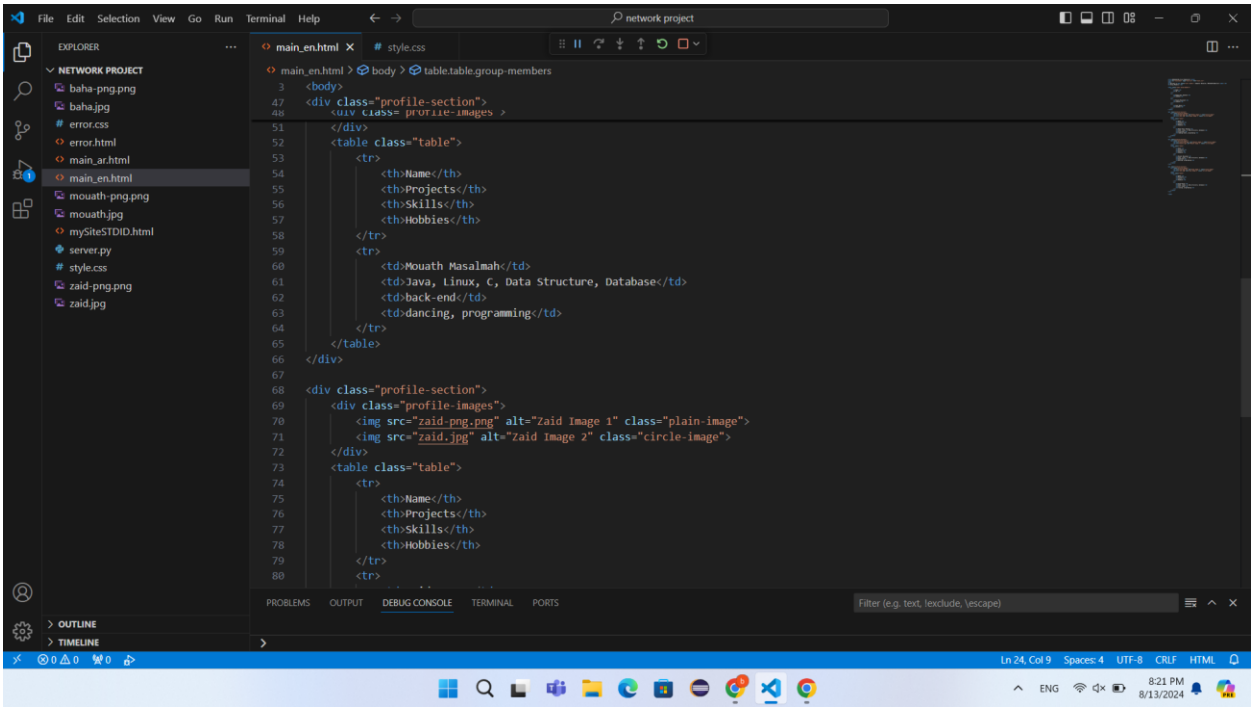


Figure 50:png and jpg 3

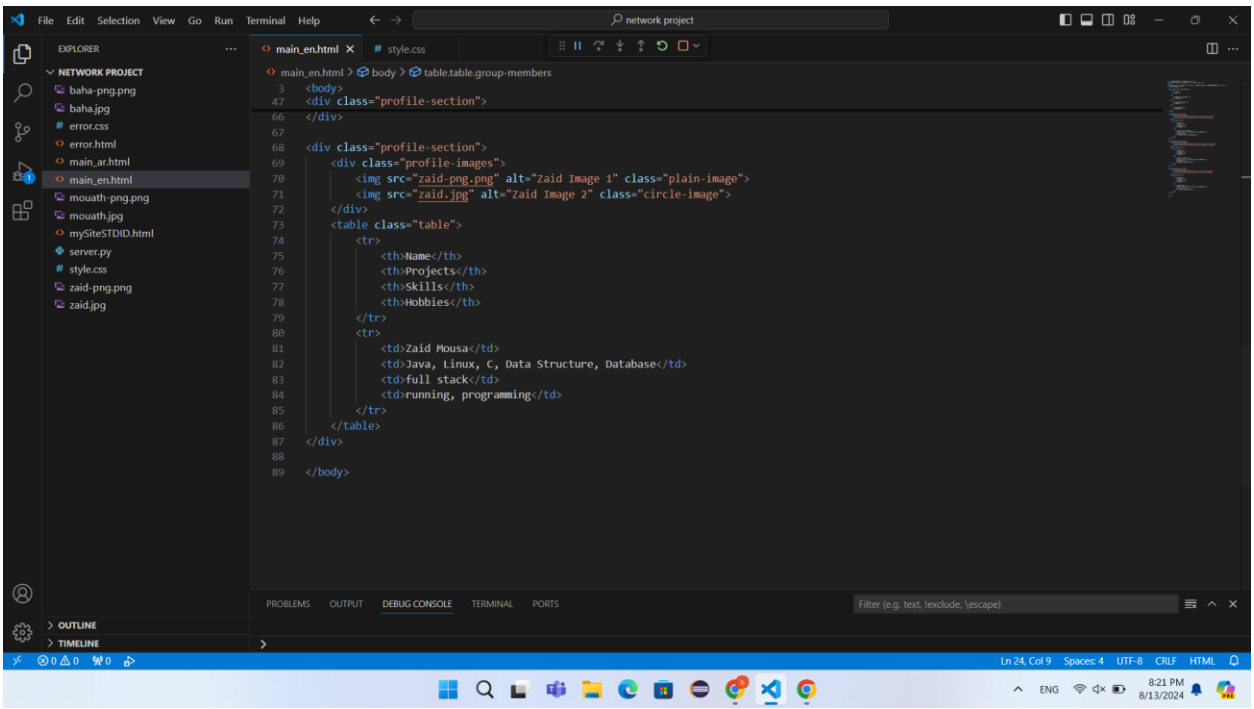


Figure 51:png and jpg 4

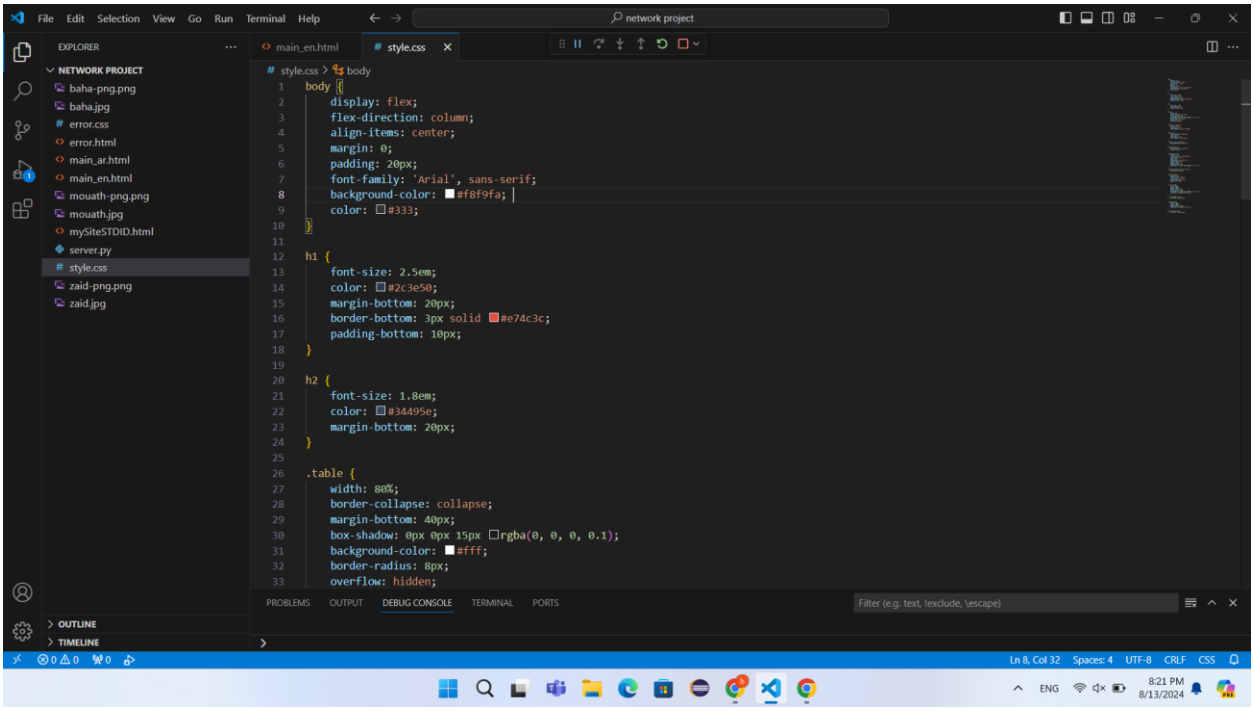


Figure 52:css png and jpg 1

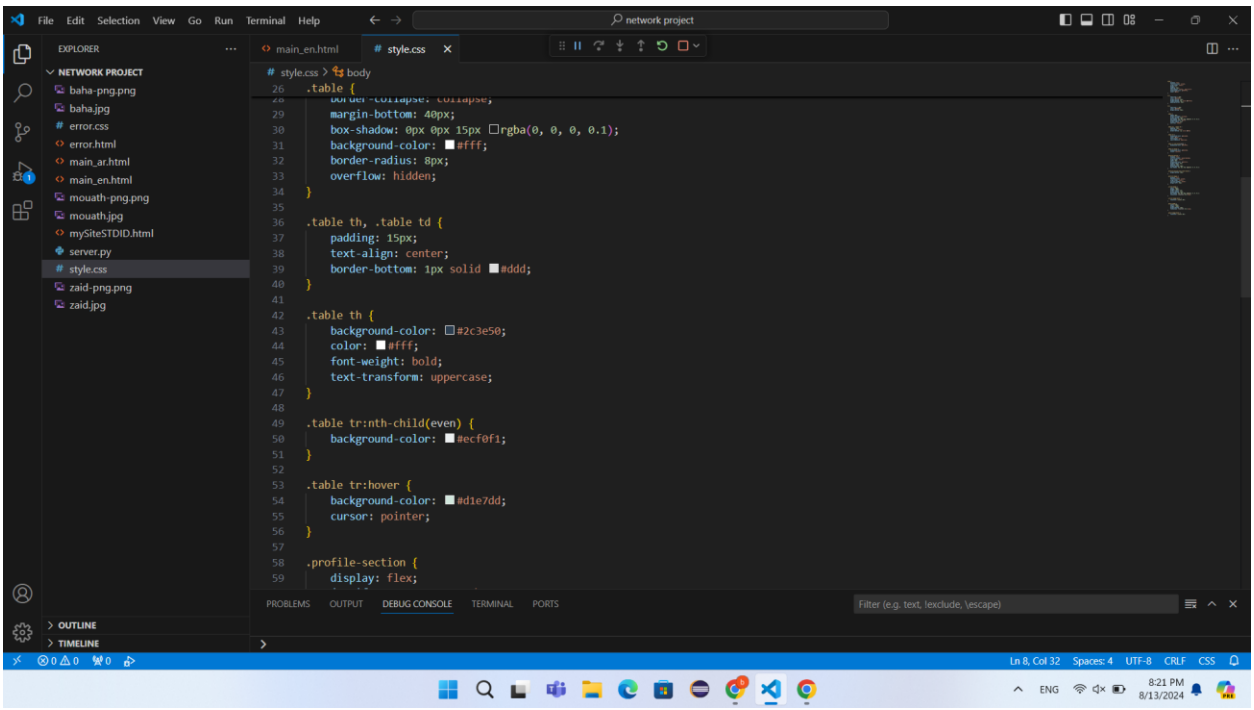


Figure 53:css png and jpg 2

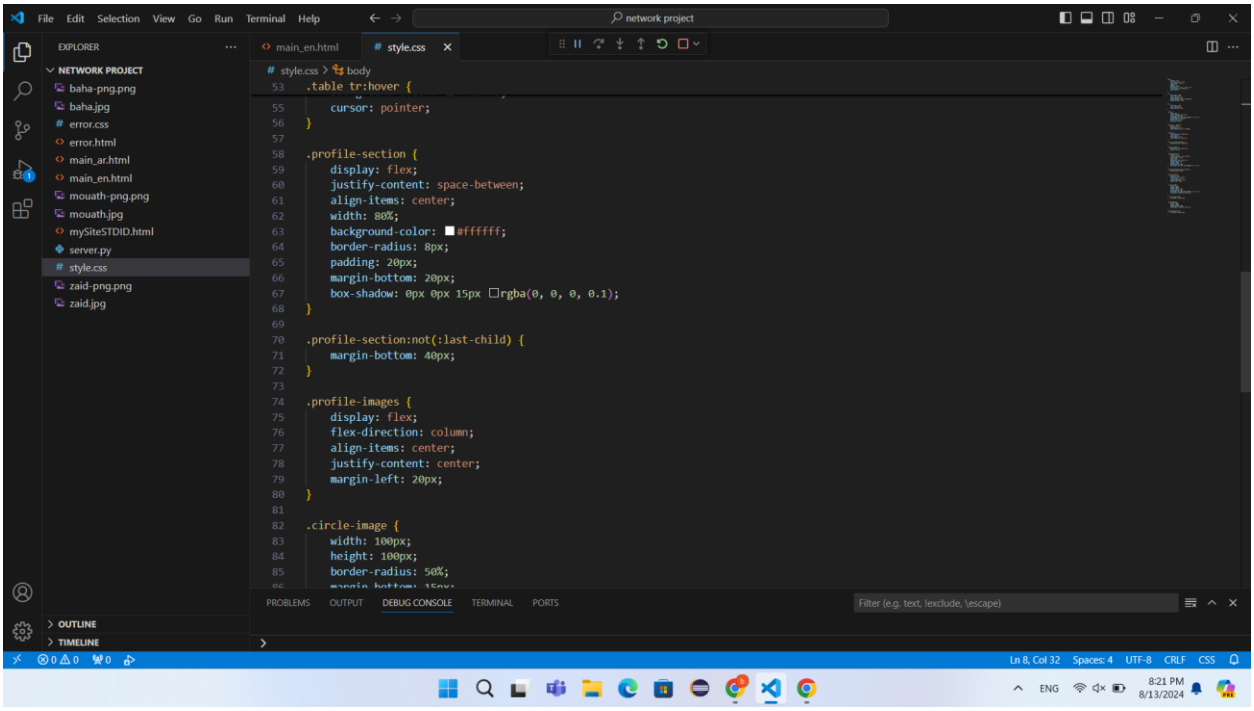


Figure 54:css png and jpg 3

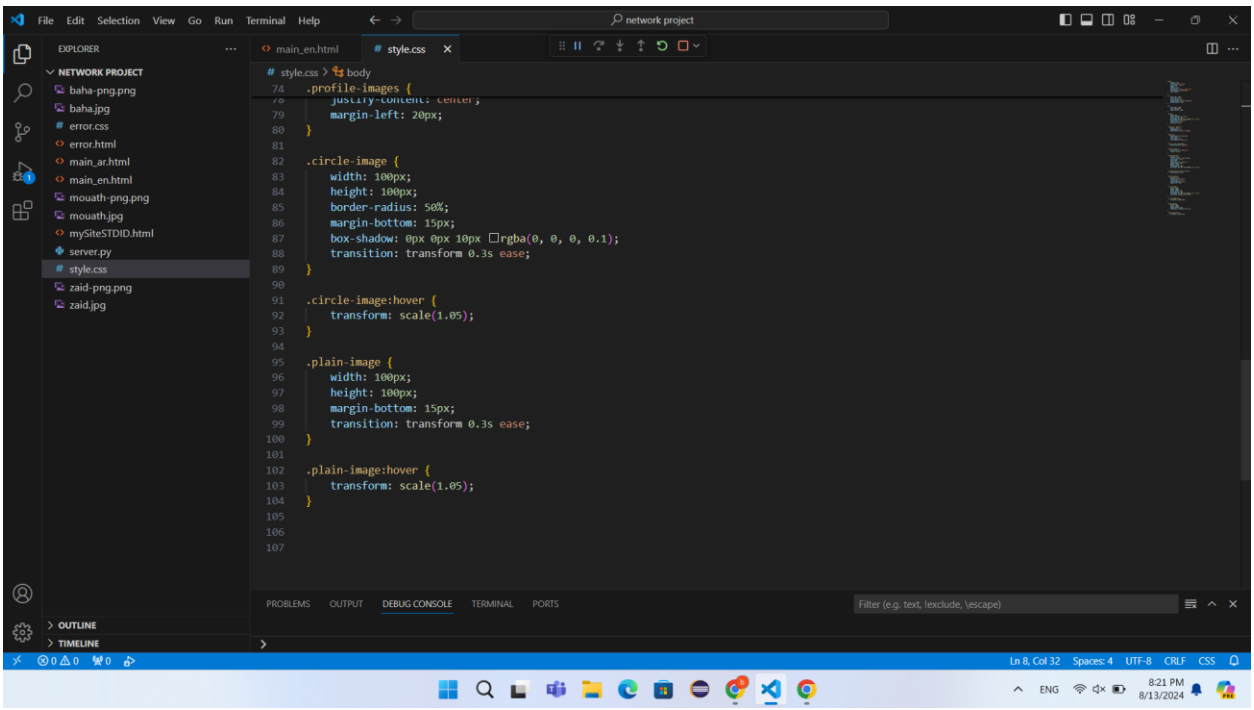


Figure 55:css png and jpg 4

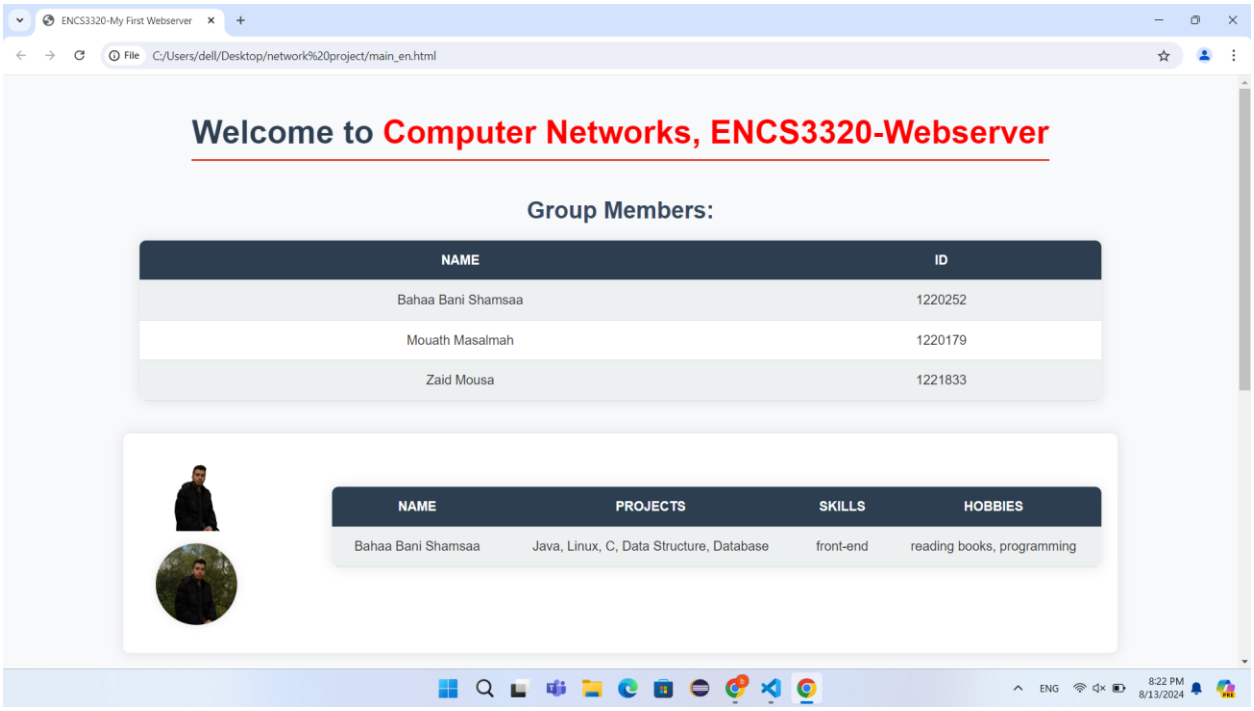


Figure 56:browser for png and jpg 1

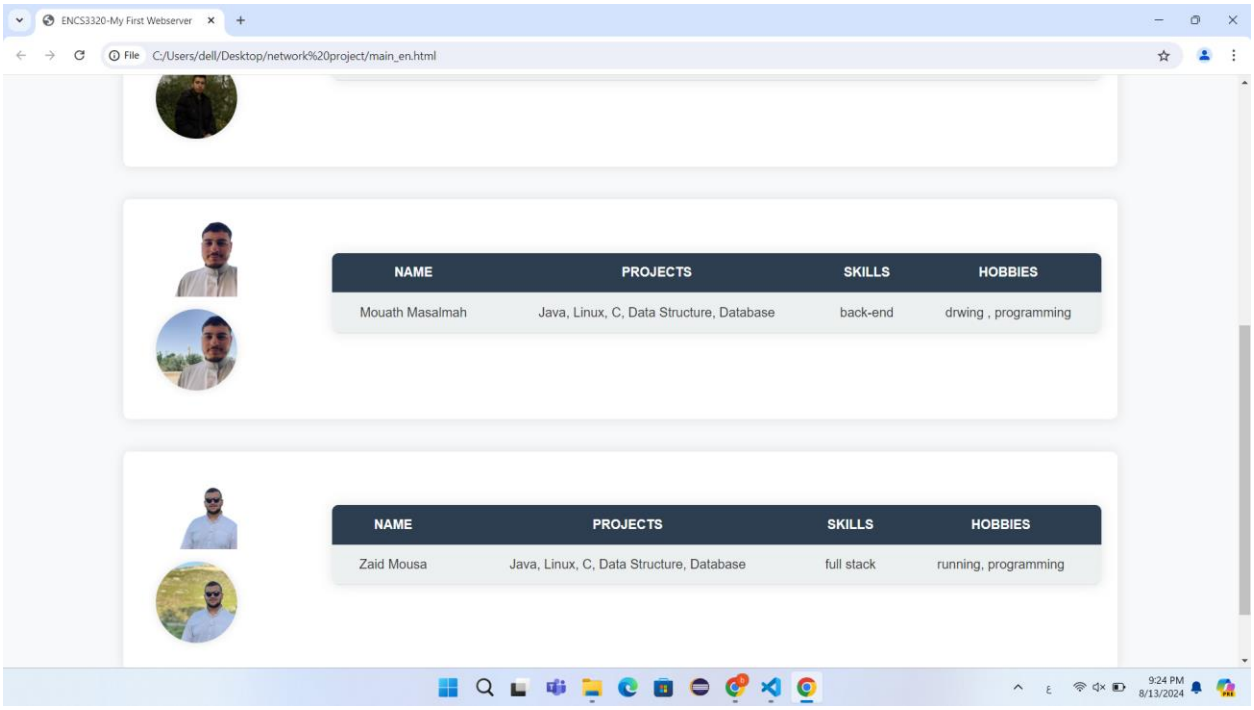


Figure 57:browser for png and jpg 2

As we see here, we put 6 images 3 .jpg and 3 .png, each member we put his images at his table , we use border-radius to make the img circle

i and j:

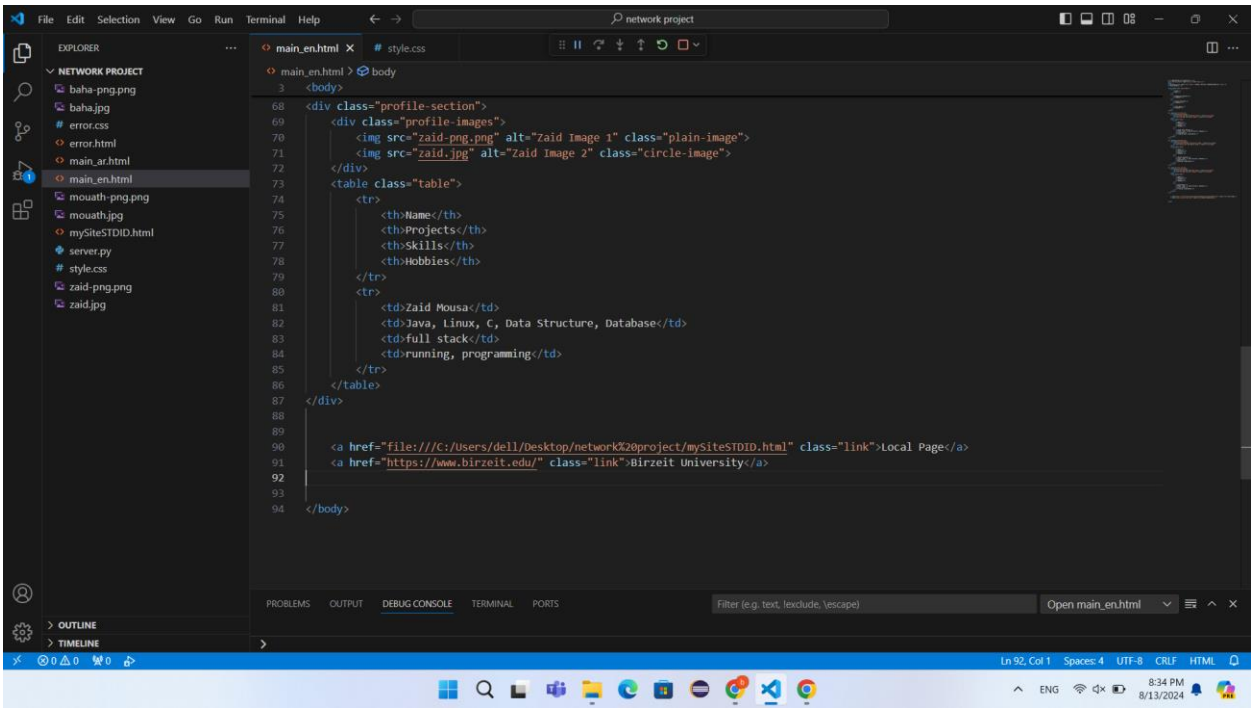


Figure 58: link to local html and Birzeit web

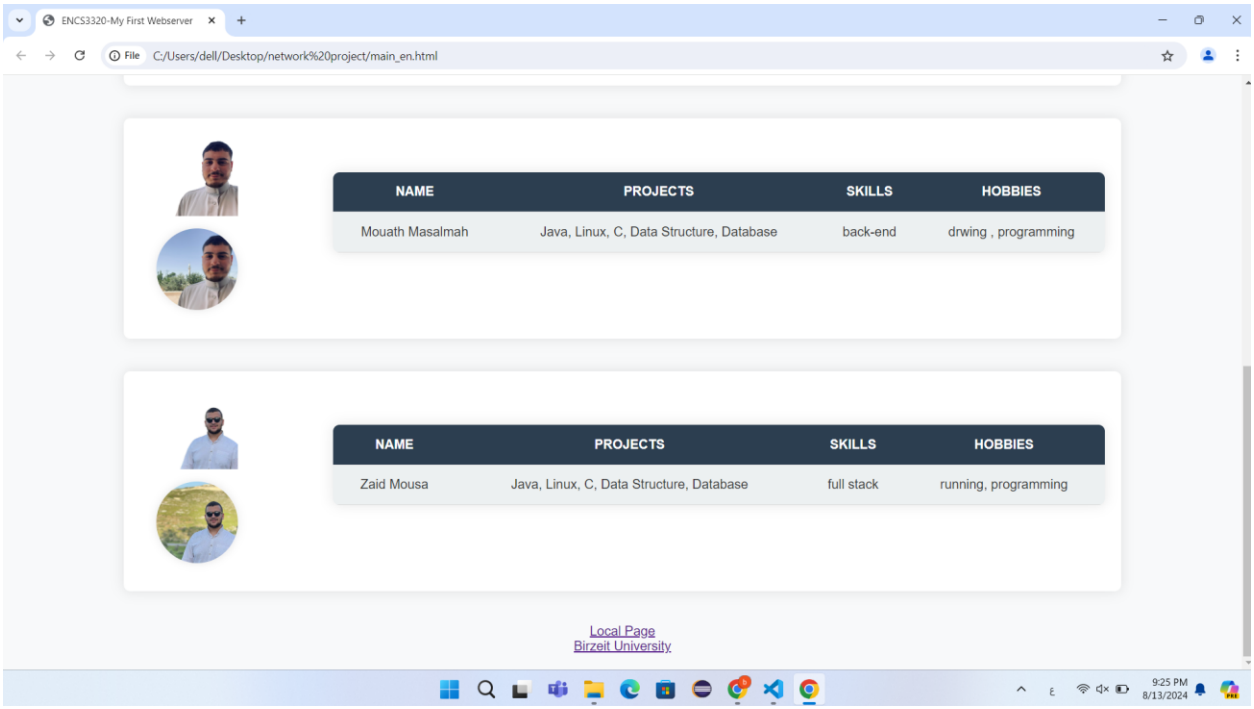


Figure 59:browser for link to local html and Birzeit web

Here we put 2 links one for Birzeit and another one for a local page that it is mySiteSTDID.html

2.

Figure 60: /ar request

Figure 61: main_ar.html request

As we see here when I search using this link http://localhost:1833/ar it shows the Arabic version of main_en.html which is main_ar.html

3.

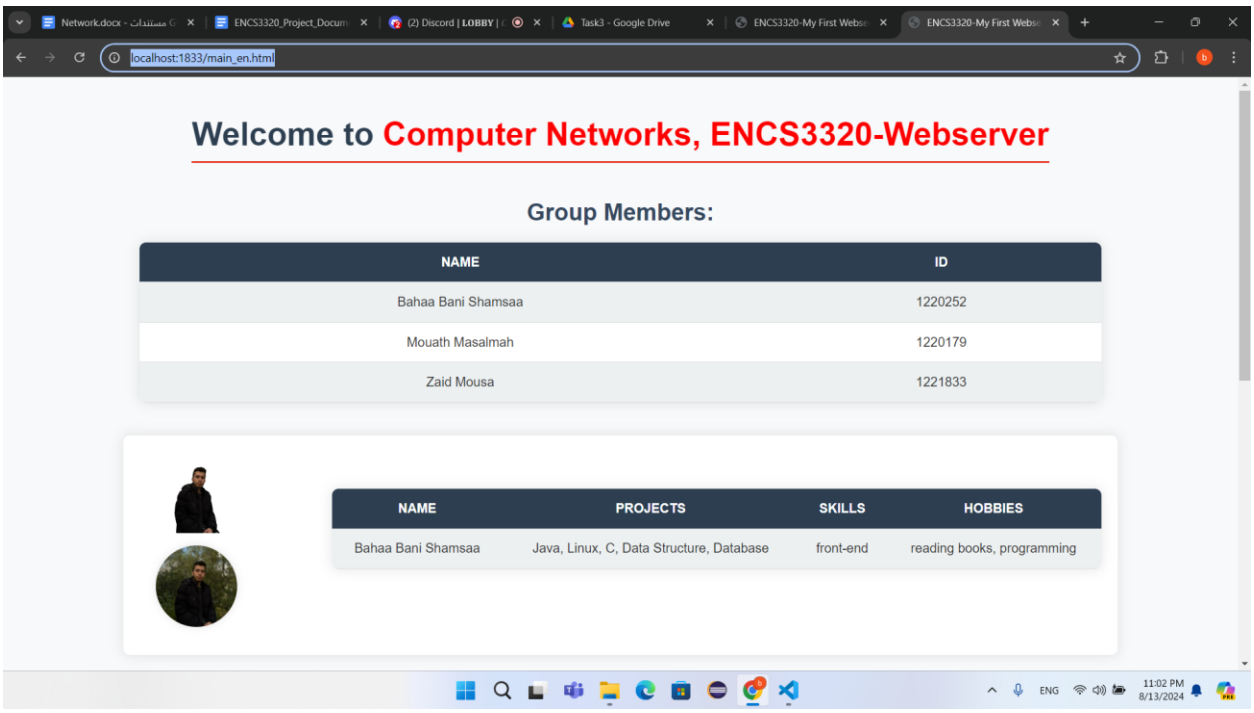


Figure 62: main_en.html request

Here we tried to search for an .html file and we search for main_en.html using the link http://localhost:1833/main_en.html and it shows the contents of main_en.html file

4.

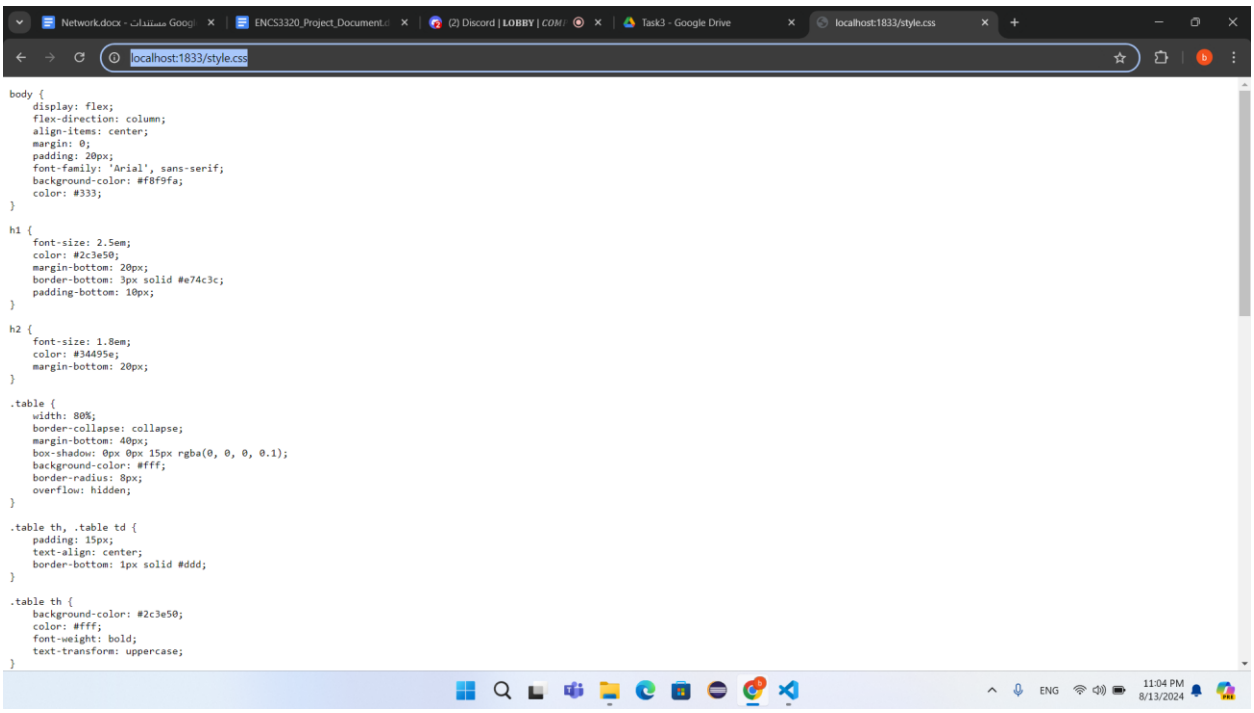


Figure 63: style.css request

Here we tried to search for and .css file and we search for style.css using the link http://localhost:1833/style.css and it shows the content of style.css file

5.

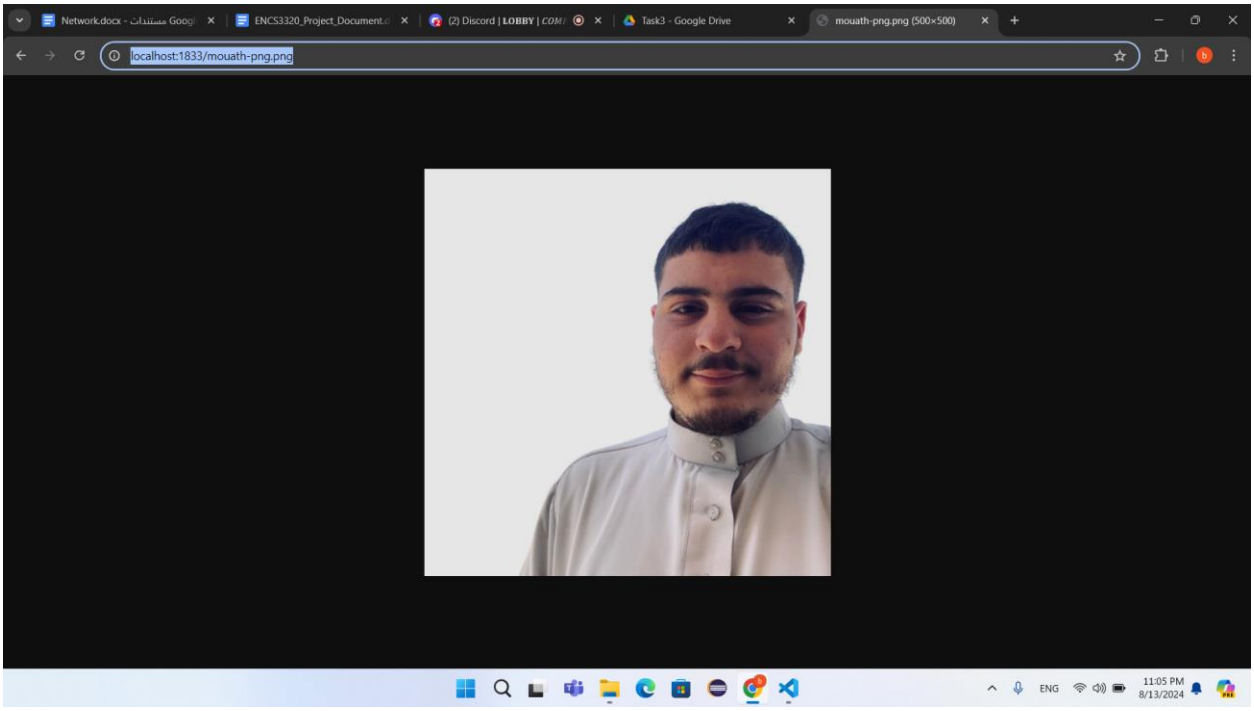


Figure 64: mouath.png request

Here we tried to search for .png img and we search for mouath-png.png using the link http://localhost:1833/mouath-png.png and it shows the content of mouath-png.png

6.

Figure 65: zaid.jpg request

Here we tried to search for .jpg img and we search for zaid.jpg using the link http://localhost:1833/zaid.jpg and it shows the content of zaid.jpg

7 and 8:

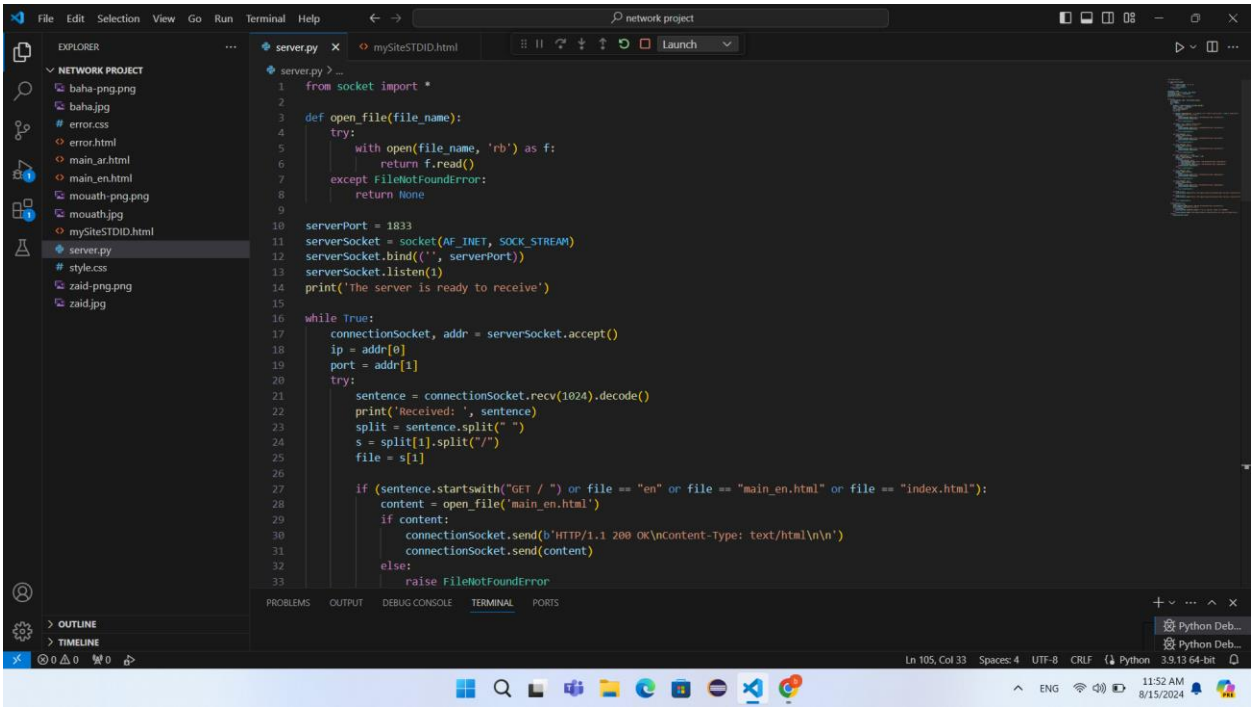


Figure 66: server 1

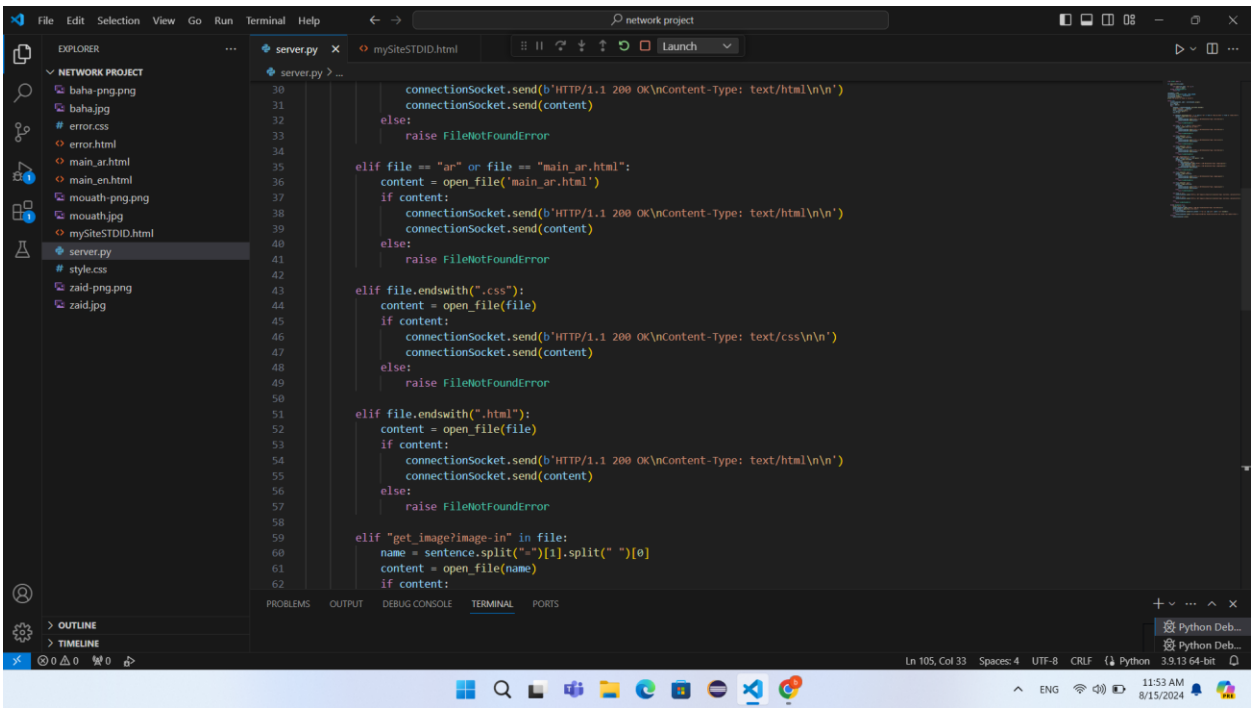


Figure 67: server 2

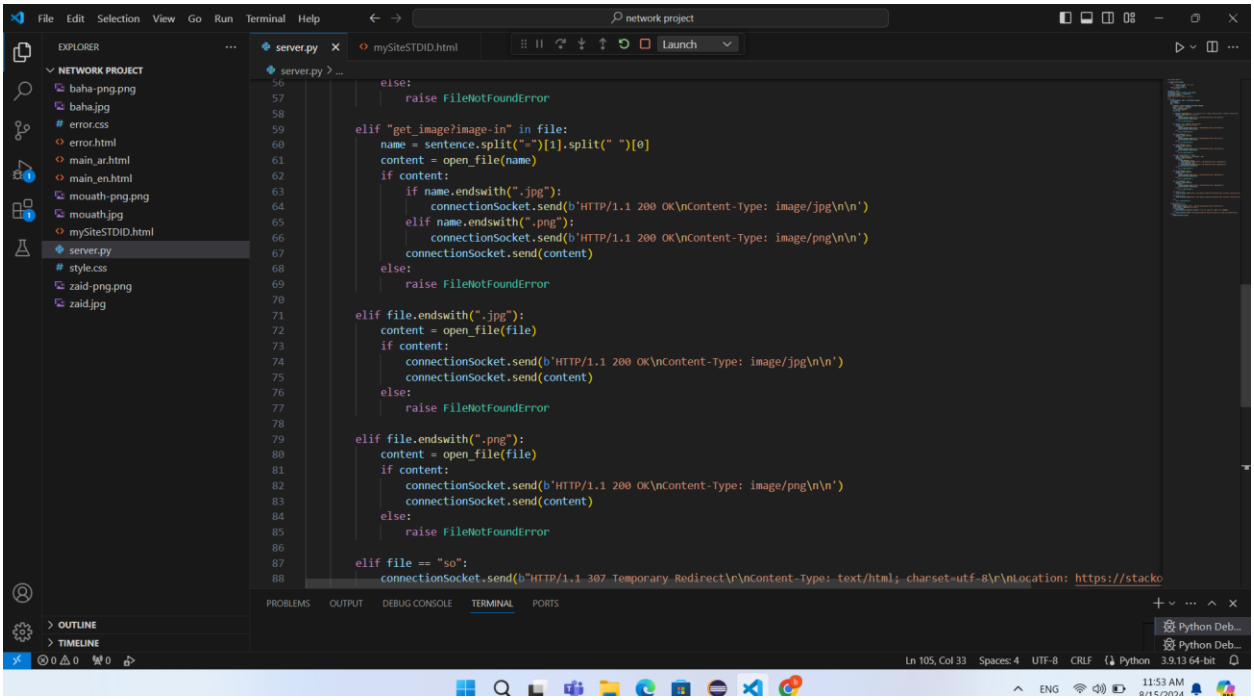


Figure 68: server 3

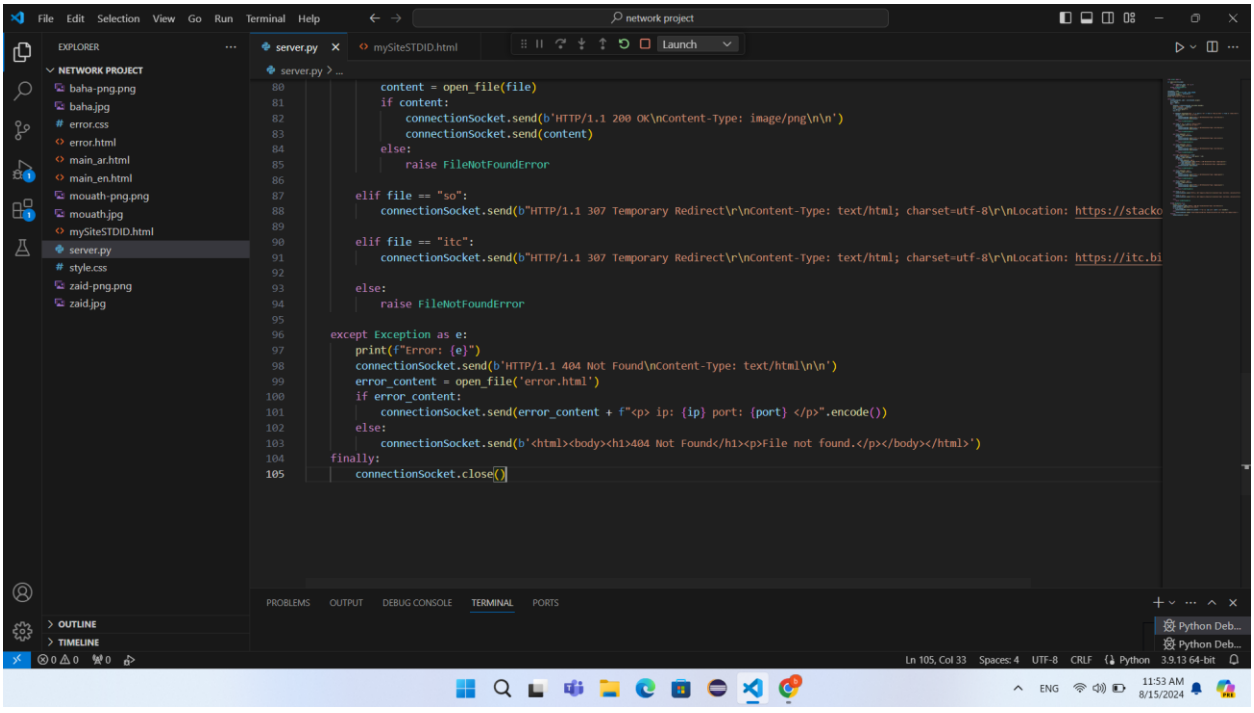


Figure 69: server 4

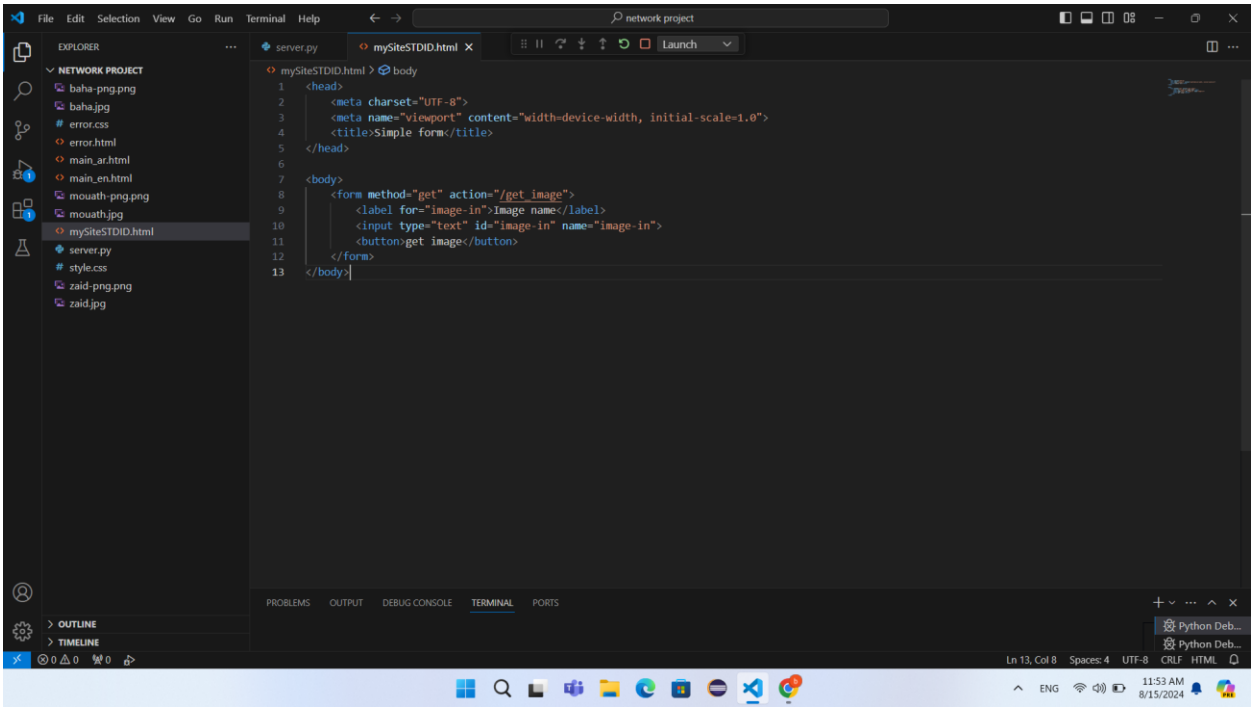


Figure 70: mySiteSTDID.html

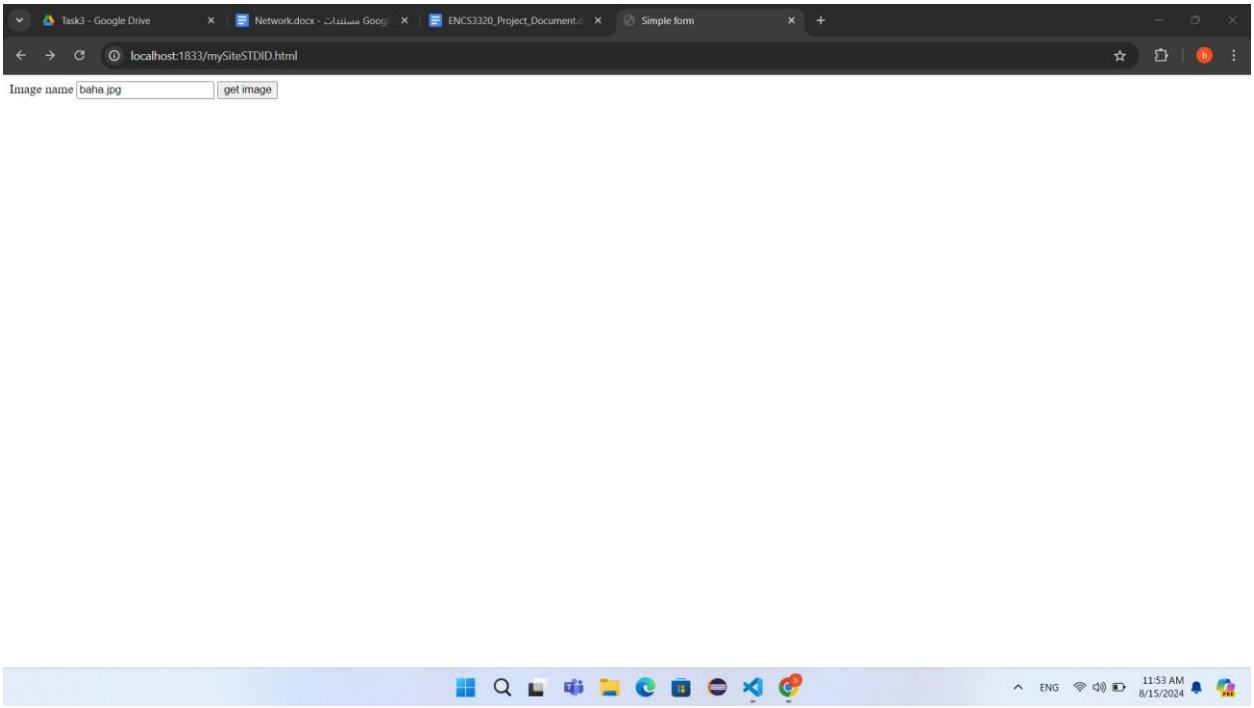


Figure 71: browesr for mySiteSTDID.html

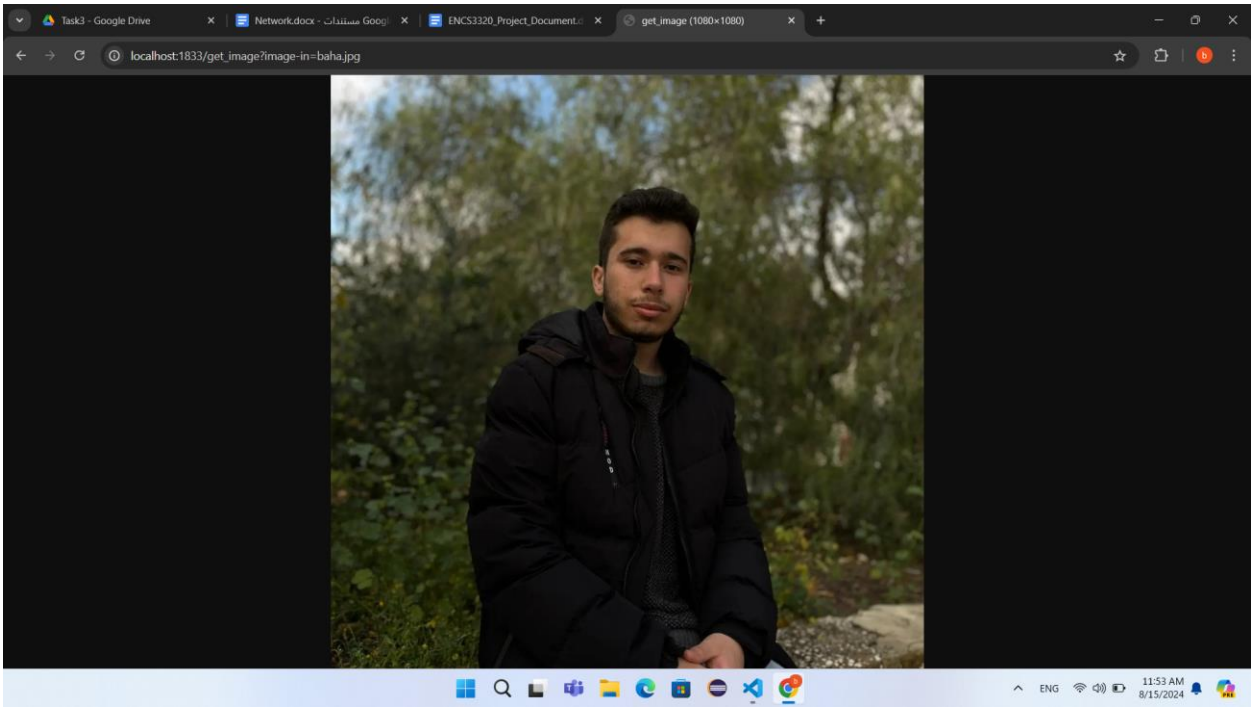


Figure 72: image browser for mySiteSTDID.html

Aswe see in images we use zaid ID which is 1221833 and we use last 4 numbers 1833 as a port number then we create socket then we set it up , then we see if the sentence we have start with / or en or main_en.html or index.html then open file main_en.html, else if sentence start with ar or main_ar.html so open man_ar.html file , else if sentence end with .css so it open the file requested .css , else if sentence end with .html it open the file requested .html , else if sentence contains get_image?image_in this asking for an specific image , else if file ends with .jpg serarch for that file , else if file end with .png search for that file

9.

a.

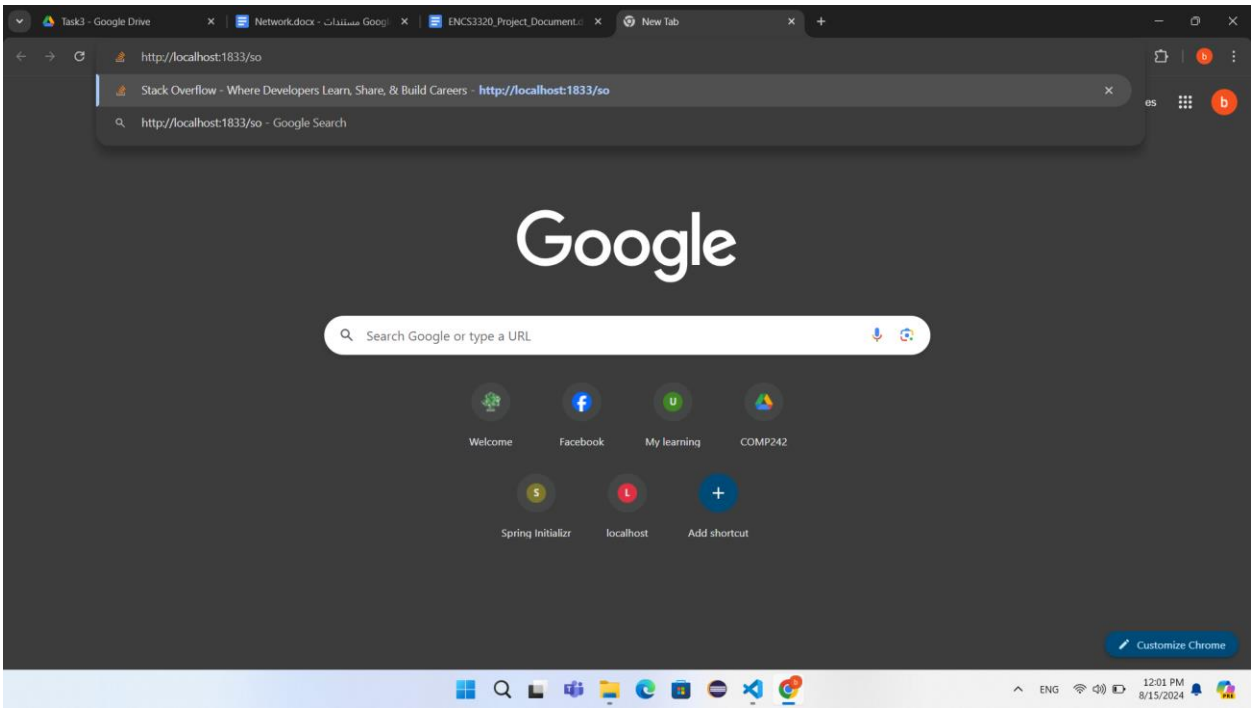


Figure 73: /so request

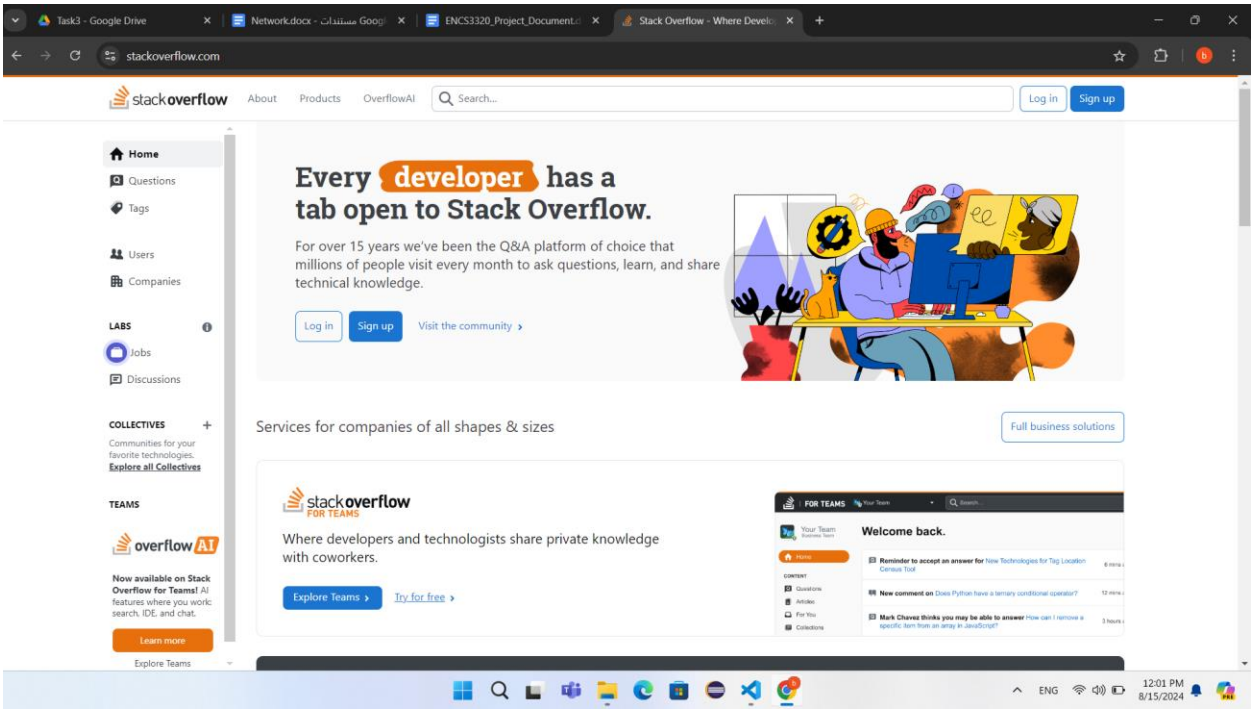


Figure 74: solution for /so

Here we search for http://localhost:1833/so and it goes to stackoverflow.com

b.

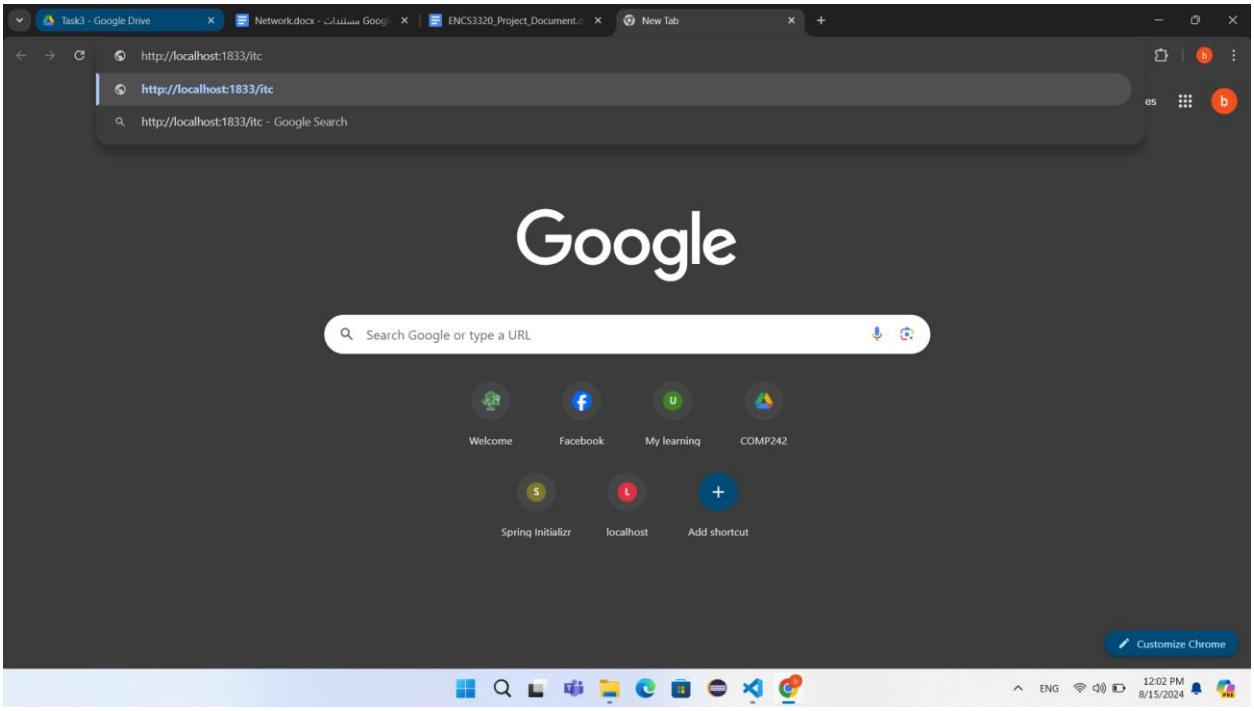


Figure 75: /itc request

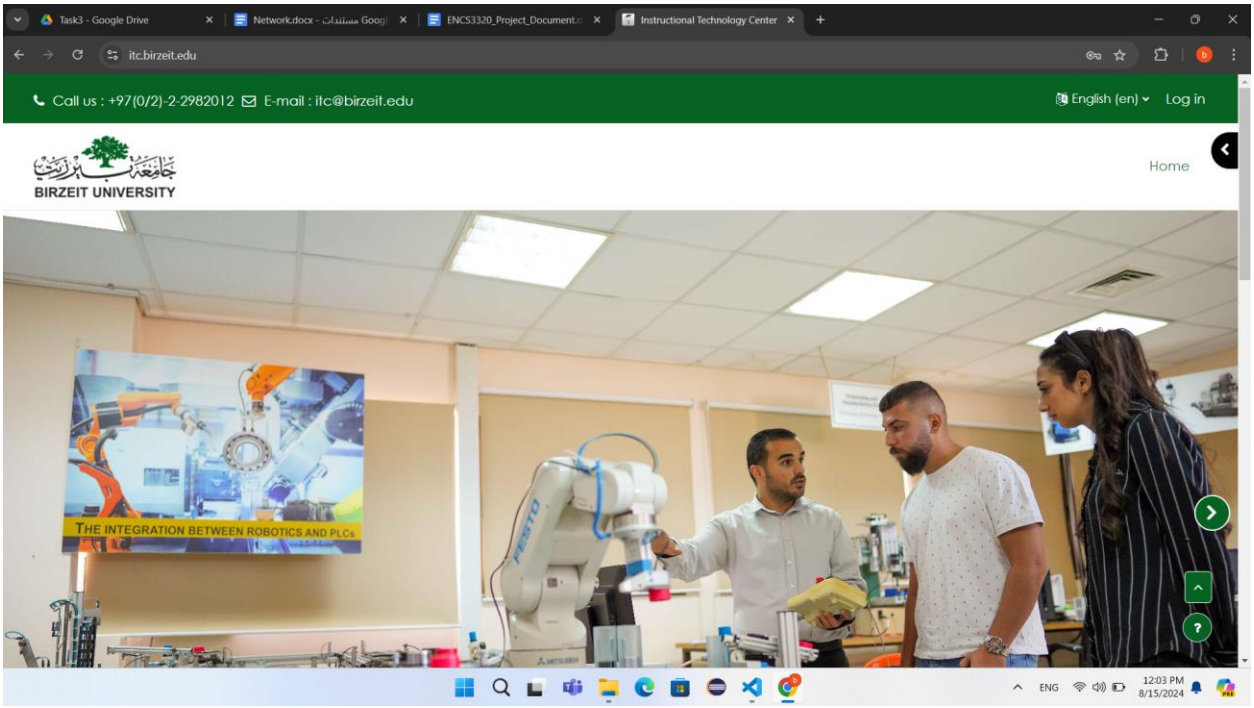


Figure 76: solution for /itc

Here we search for http://localhost:1833/itc and it goes to itc.birzeit.edu

10.

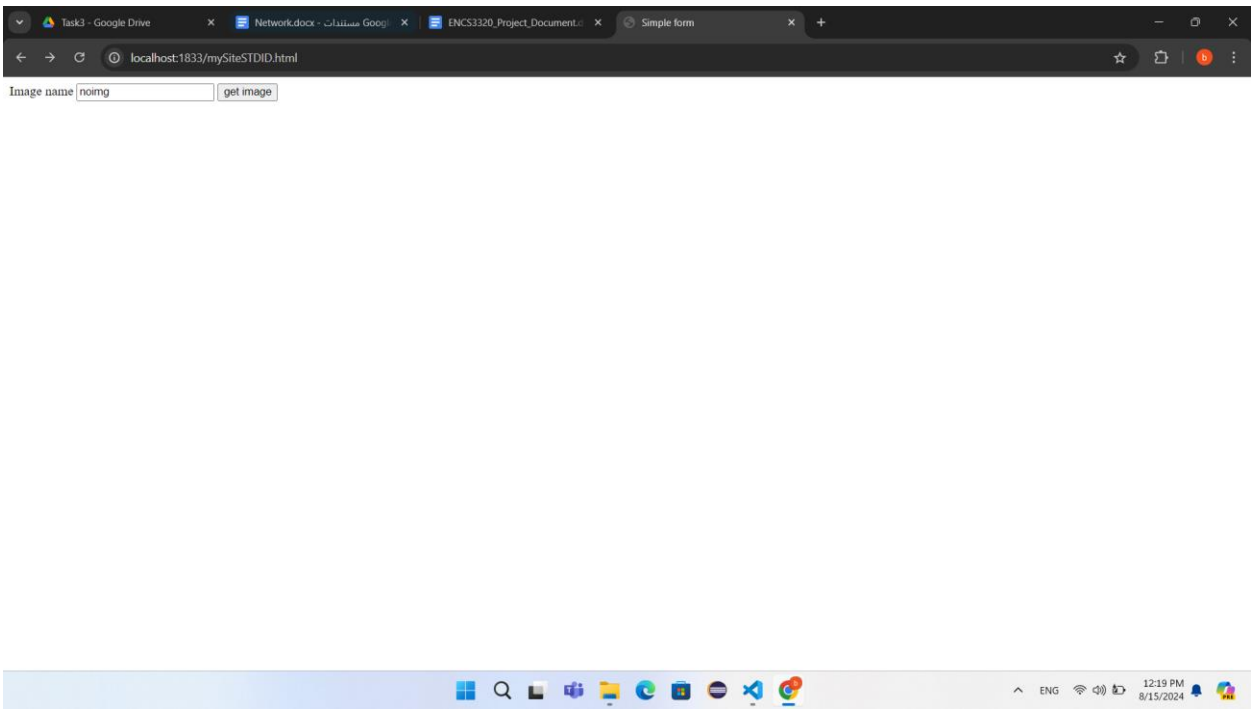


Figure 77: wrong image

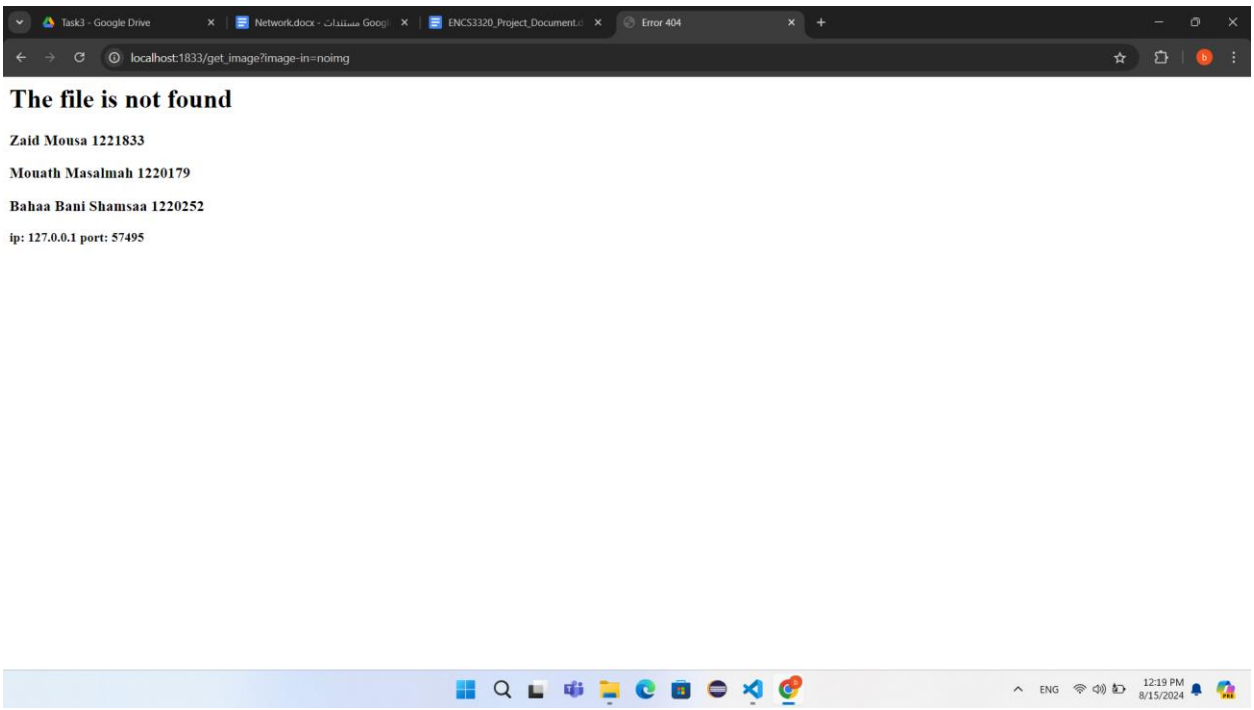


Figure 78: error.html browser

Here when we search for an image does not exist the error.html page shows

11.

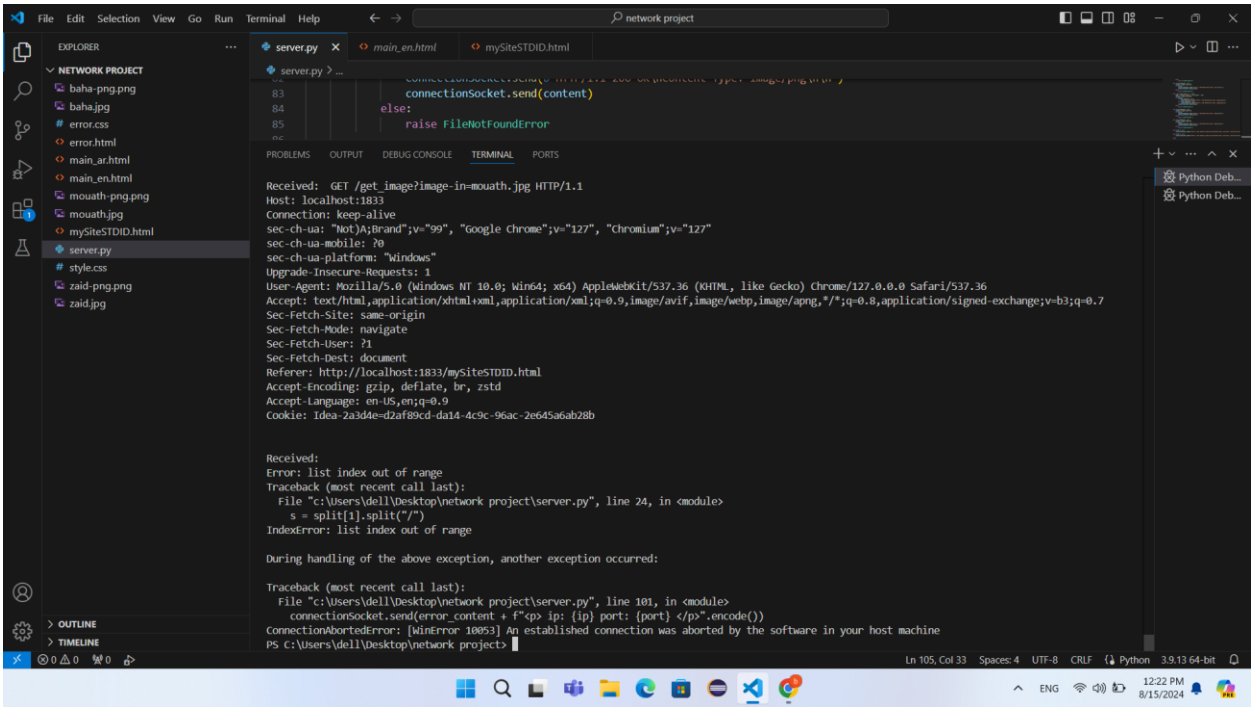


Figure 79: http request on terminal

Here first it is asking the server for a file named mouath.jpg then the request sent to the server then it tells the server what browser and operating system are used

Problems and challenges:

We had a problem with the arabic page when we search for it , it shows like that:

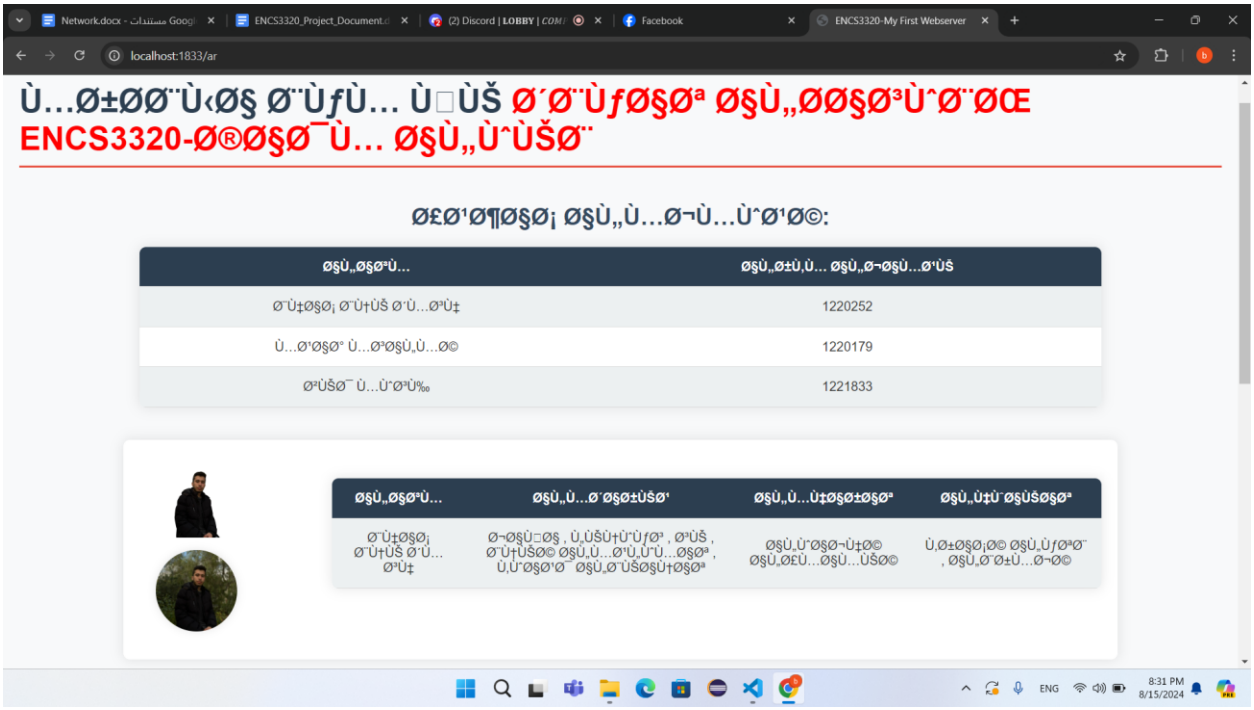
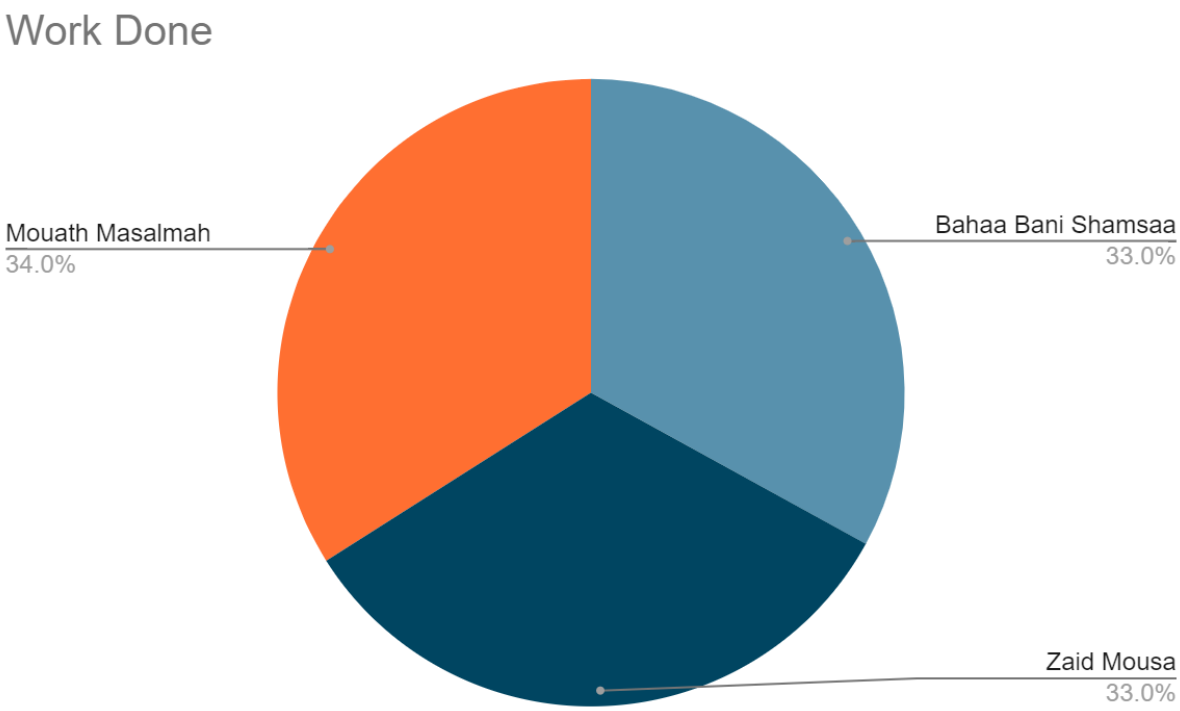


Figure 79: problem on /ar

and we solve it by adding `<meta charset="UTF-8">`

Work Done:



Code and report:

Task1:

1.Mouath

2.A.Bahaa

B.Bahaa

C.Bahaa

D.Mouath

E.Zaid

3.Mouath

4.Zaid

Task 2:

1. Zaid , Mouath and Bahaa

2.Zaid and Mouath

Task 3:

1.Bahaa and Zaid

A.Bahaa

B.Bahaa

C.Bahaa

D.Bahaa

E.Bahaa

F.Bahaa

H.Bahaa

I.Bahaa

J.Bahaa

2.Mouath

3.Mouath

4.Zaid

5.Zaid

6.Bahaa

7.Bahaa

8.Mouath

9.A.Zaid

B.Zaid

10.Mouath

11.Mouath

References

DNS : https://www.cloudflare.com/learning/dns/what-is-dns/

Round-trip time (RTT): https://www.cloudflare.com/learning/cdn/glossary/round-trip-time-rtt/

Hop (networking): https://en.wikipedia.org/wiki/Hop_(networking)

"Bad Request" error: https://www.semrush.com/blog/400-bad-request/

vowels: https://www.grammarly.com/blog/vowels/

port: https://www.cloudflare.com/learning/network-layer/what-is-a-computer-port/

html, css & python: https://www.w3schools.com/

wireshark:https://www.comptia.org/content/articles/what-is-wireshark-and-how-to-use-it#:~:text=Wireshark%20is%20a%20network%20protocol,packet%20sniffer%20in%20the%20world.