



BIRZEIT UNIVERSITY

Faculty of Engineering and Technology

Electrical and Computer Engineering

Department

Computer Network – ENCS3320

Prepared by:

Malak Nassar	ID:1200757	section:1
Tala Jebrini	ID: 1200493	section:1
Aya Dahbour	ID: 1201738	section:2
Hadeel Froukh	ID: 1201585	section:1

Instructor: Dr. Abdalkarim Awad

BIRZEIT

Date: May 31, 2023

Table of Contents

List of Figures.....	3
1. Part 1:.....	1
1.1. Definitions:.....	1
1.2. Commands :.....	1
2. Part 2:.....	4
3. Part 3:.....	8
4. Part 4:.....	33

List of Figures

Figure 1:Ping a device in the same network.....	1
Figure 2:ping www.yale.edu.....	2
Figure 3:tracert www.harvard.edu.....	3
Figure 4:nslookup www.harvard.edu.....	3
Figure 5: server code	4
Figure 6:client Aya	5
Figure 7:client Malak	5
Figure 8:client Tala	6
Figure 9:client Hadeel.....	6
Figure 10:UDP client and server implementation output	7
Figure 11:request ‘/’	8
Figure 12:HTTP response for ‘/’	8
Figure 13:request index.html	9
Figure 14:HTTP response for request index.html.....	9
Figure 15:request ‘main_en.html’	10
Figure 16:HTTP response request ‘/main_en.html’	10
Figure 17:Figure 15:request ‘en’	11
Figure 18:HTTP response request ‘en’	11
Figure 19:request ‘ar’	12
Figure 20:HTTP response request ‘ar’	12
Figure 21:request ‘.html’	13
Figure 22:HTTP response request ‘.html’	13
Figure 23:request ‘.css’	14
Figure 24:HTTP response request ‘.css’	14
Figure 25:request ‘.png’	15
Figure 26:HTTP response request ‘.png’	15
Figure 27:request ‘.jpg’	16
Figure 28:HTTP response request ‘.jpg’	16
Figure 29:request ‘yt’	17
Figure 30:HTTP response request ‘yt’	17
Figure 31:HTTP response request ‘so’	18
Figure 32:entire arabic page.....	31
Figure 33:entire english page	32

1. Part 1:

1.1. Definitions:

1.1.1. What is ping:

Ping is a simple computer network software utility that is used to test and validate a host's reachability on an Internet Protocol (IP) network.

1.1.2. What is tracert:

Traceroute is a command that allows you to 'trace' the path that a packet takes on its way to its destination.

1.1.3. What is nslookup:

nslookup is a command-line tool for querying IP addresses and DNS records, including reverse DNS lookup.

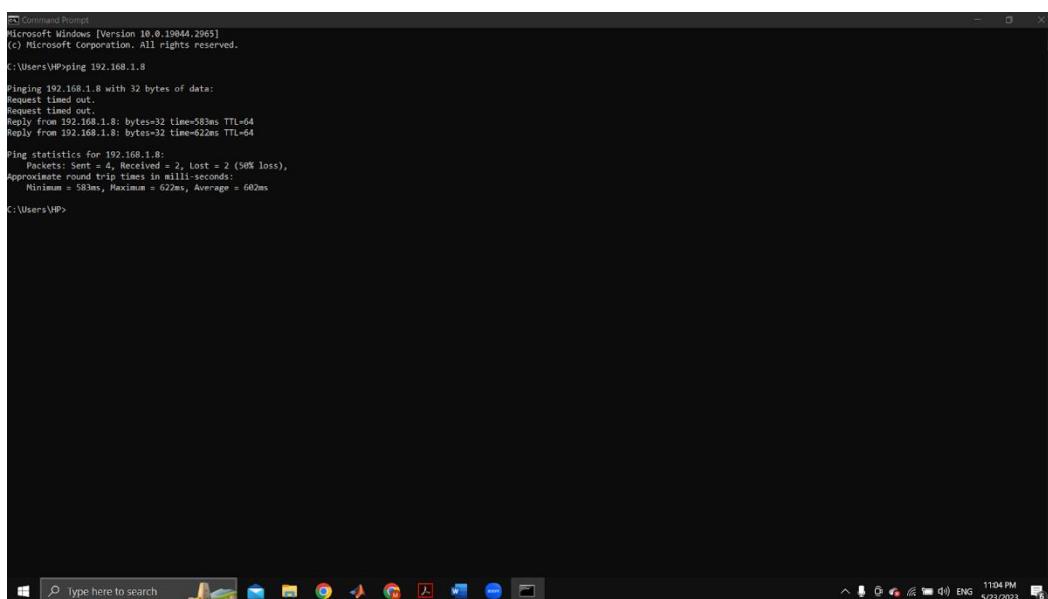
1.1.4. What is telnet:

Telnet is a TCP/IP-based network protocol for remote text-based communication between computers.

1.2. Commands :

1.2.1. Ping a device in the same network from Laptop to a smartphone:

The device is a smart phone of IP address 192.168.1.8.



```
C:\Users\VP>ping 192.168.1.8

Pinging 192.168.1.8 with 32 bytes of data:
Request timed out.
Request timed out.
Reply from 192.168.1.8: bytes=32 time=583ms TTL=64
Reply from 192.168.1.8: bytes=32 time=622ms TTL=64

Ping statistics for 192.168.1.8:
    Packets: Sent = 4, Received = 2, Lost = 2 (50% loss),
Approximate round trip times in milli-seconds:
    Minimum = 583ms, Maximum = 622ms, Average = 602ms

C:\Users\VP>
```

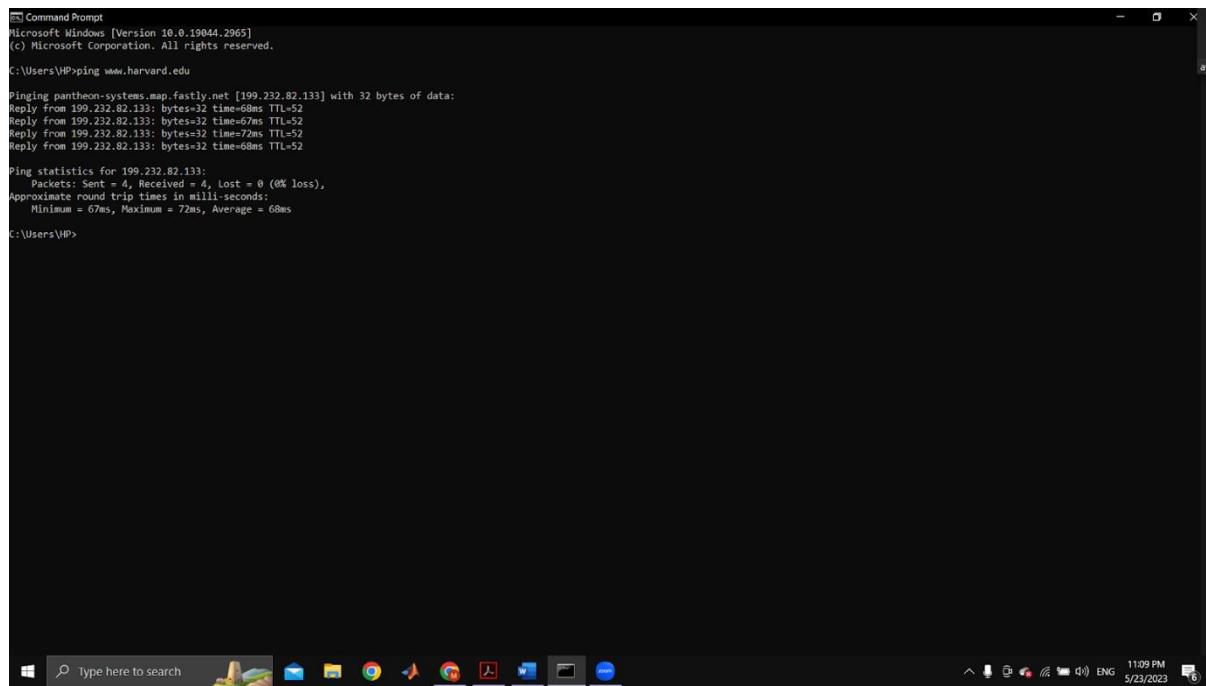
Figure 1:Ping a device in the same network

As shown in figure 1 that we received a response from phone IP. When four packets are sent, each

with the same period through TTL, they are all received without any loss, because of the small destination between the devices.

1.2.2. Ping www.harvard.edu:

As seen in figure2, four packets were transmitted with the same TTL but various time delays of Minimum = 67ms, Maximum = 72ms, and Average = 68ms. We can see that the smartphone response time is shorter than the www.yale.edu response time, Since the two servers are not in the same network, the time for each packet to travel to and from its destination is more than when they were in the same network in the previous part.



```
Command Prompt
Microsoft Windows [Version 10.0.19044.2965]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>ping www.harvard.edu

Pinging pantheon-systems.map.fastly.net [199.232.82.133] with 32 bytes of data:
Reply from 199.232.82.133: bytes=32 time=68ms TTL=52
Reply from 199.232.82.133: bytes=32 time=67ms TTL=52
Reply from 199.232.82.133: bytes=32 time=72ms TTL=52
Reply from 199.232.82.133: bytes=32 time=68ms TTL=52

Ping statistics for 199.232.82.133:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 67ms, Maximum = 72ms, Average = 68ms

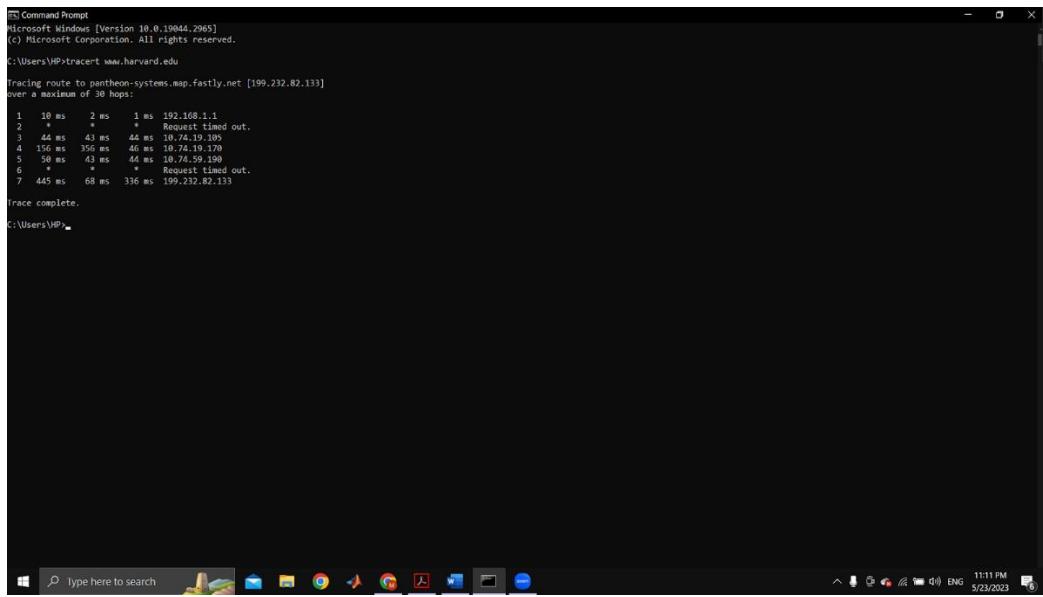
C:\Users\HP>
```

Figure 2:ping www.yale.edu

1.2.3. Tracert www.harvard.edu:

Tracert is used to trace the path that an Internet Protocol packet takes to the destination. Using this command (tracert), three messages are sent to each router, followed by a wait for the router's response, and the procedure is repeated until you reach the correct IP address. The measurements grow as we move down the line because the router rotates farther. If the request fails, the packet is marked * The packet is unable to reach the targeted destination due to a problem along the path or location.

Figure3 shows how much time the packets take to reach the next server and the path was used.



```
Command Prompt
Microsoft Windows [Version 10.0.19044.2965]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>tracert www.harvard.edu

Tracing route to pantheon-systems.map.fastly.net [199.232.82.133]
over a maximum of 30 hops:
1  10 ms   2 ms   1 ms  192.168.1.1
2  *         Request timed out.
3  44 ms   43 ms   44 ms  10.74.19.170
4  156 ms  356 ms  46 ms  10.74.59.190
5  50 ms   43 ms   44 ms  10.74.59.190
6  *         *         *         Request timed out.
7  445 ms  68 ms   336 ms  199.232.82.133

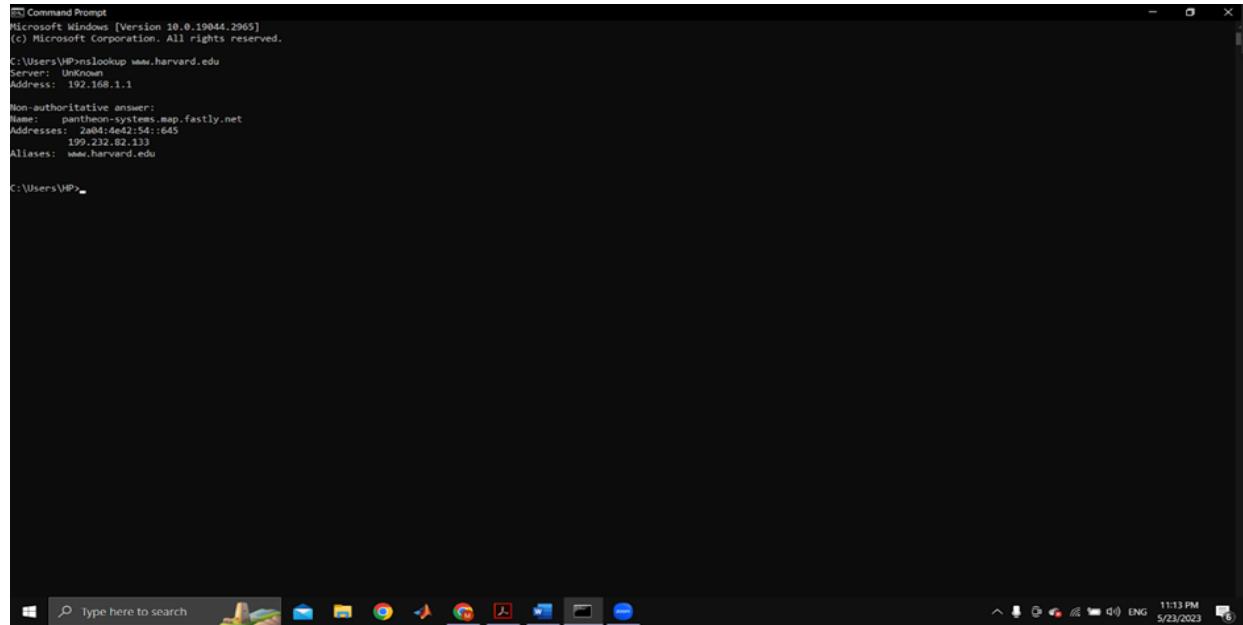
Trace complete.

C:\Users\HP>
```

Figure 3:tracert www.harvard.edu

1.2.4. nslookup www.harvard.edu:

nslookup is used to display the IP address and other information, such as the name of the server As shownen in figure 4:



```
Command Prompt
Microsoft Windows [Version 10.0.19044.2965]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>nslookup www.harvard.edu
Server:  Unknown
Address: 192.168.1.1

Non-authoritative answer:
Name:  pantheon-systems.map.fastly.net
Address: 2408:4442:54::645
199.232.82.133
Aliases: www.harvard.edu

C:\Users\HP>
```

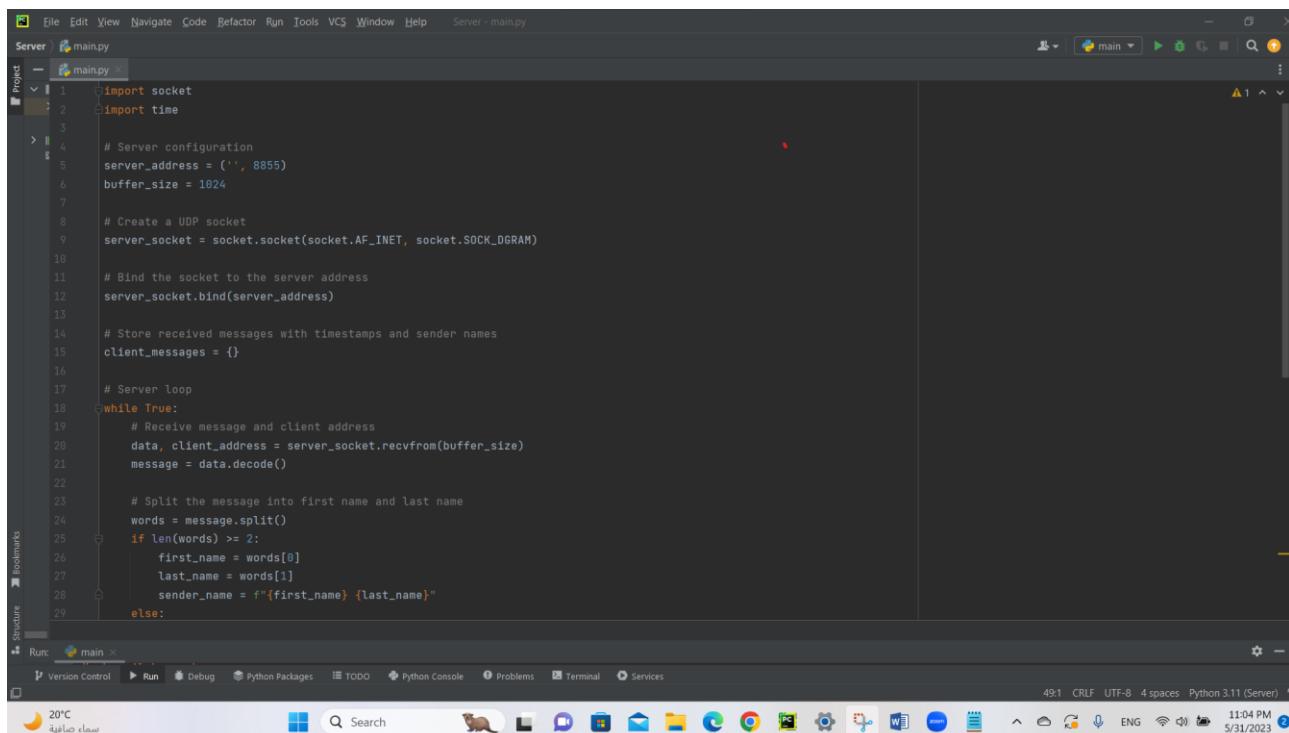
Figure 4:nslookup www.harvard.edu

2. Part 2:

Implement the following server and client application both for UDP. The client sends broadcast messages every 2 seconds to a server listening on port 8855. The server counts the received messages. Run the programs.

-On 4 different computers connected through the same network

A screenshot for the server:



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

- File Menu:** File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Project Bar:** Server > main.py
- Code Editor:** The main.py file contains Python code for a UDP server. The code imports socket and time, sets up a server socket on port 8855, binds it to the server address, and enters a loop to receive messages from clients. It splits each message into first name and last name, stores them in a dictionary, and then prints the last received message for each client.
- Toolbars:** Version Control, Run, Debug, Python Packages, TODO, Python Console, Problems, Terminal, Services.
- Status Bar:** 49:1 CRLF UTF-8 4 spaces Python 3.11 (Server), 20°C, 11:04 PM, 5/31/2023.

Figure 5: server code

The UDP server code sets up a server that listens on a specific port (e.g., port 8855). It waits for incoming messages from clients. When a message is received, the server extracts the sender's name and IP address, and stores the last received message from each client. It then displays the last received messages from all clients, including the sender's name and the time of reception.

A screenshots for 4 clients:

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

- Project:** Client
- File:** main.py
- Code Content:**

```
1 import socket
2 import time
3
4 # Client configuration
5 server_address = ('172.19.26.22', 8855)
6 buffer_size = 1024
7 broadcast_interval = 2 # in seconds
8 student_name = "Aya Dahboun"
9
10 # Create a UDP socket
11 client_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
12 client_socket.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)
13
14 # Client loop
15 while True:
16     # Send broadcast message
17     message = student_name.encode()
18     client_socket.sendto(message, server_address)
19
20     # Wait for the next broadcast interval
21     time.sleep(broadcast_interval)
22
23 # Close the socket (unreachable in this example)
24 client_socket.close()
```

- Run Tab:** Shows a run configuration for 'main'.
- Bottom Status Bar:** Displays system information including temperature (20°C), battery level, and date/time (11:06 PM, 5/31/2023).

Figure 6:client Aya

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

- Project:** pythonProject14
- File:** main.py
- Code Content:**

```
1 import socket
2 import time
3
4 # Client configuration
5 server_address = ('172.19.26.22', 8855)
6 buffer_size = 1024
7 broadcast_interval = 2 # in seconds
8 student_name = "Malak Nassar"
9
10 # Create a UDP socket
11 client_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
12 client_socket.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)
13
14 # Client loop
15 while True:
16     # Send broadcast message
17     message = student_name.encode()
18     client_socket.sendto(message, server_address)
19
20     # Wait for the next broadcast interval
21     time.sleep(broadcast_interval)
22
23 # Close the socket (unreachable in this example)
24 client_socket.close()
```

- Run Tab:** Shows a run configuration for 'main'.
- Bottom Status Bar:** Displays system information including temperature (20°C), battery level, and date/time (11:07 PM, 5/31/2023).

Figure 7:client Malak

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

- Project:** A project named "client" is open, containing a file "main.py".
- Code Content:**

```
1 import socket
2 import time
3
4 # Client configuration
5 server_address = ('172.19.26.22', 8855)
6 buffer_size = 1024
7 broadcast_interval = 2 # in seconds
8 student_name = "Tala Jibrini"
9
10 # Create a UDP socket
11 client_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
12 client_socket.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)
13
14 # Client loop
15 while True:
16     # Send broadcast message
17     message = student_name.encode()
18     client_socket.sendto(message, server_address)
19
20     # Wait for the next broadcast interval
21     time.sleep(broadcast_interval)
22
23 # Close the socket (unreachable in this example)
24 client_socket.close()
```
- Toolbars and Status Bar:** The toolbar includes "Run", "Version Control", "Run", "Python Packages", "TODO", "Python Console", "Problems", "Terminal", and "Services". The status bar at the bottom right shows "2:1 CRLF UTF-8 4 spaces Python 3.9 (pythonProject)" and the date/time "5/31/2023 11:08 PM".

Figure 8:client Tala

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

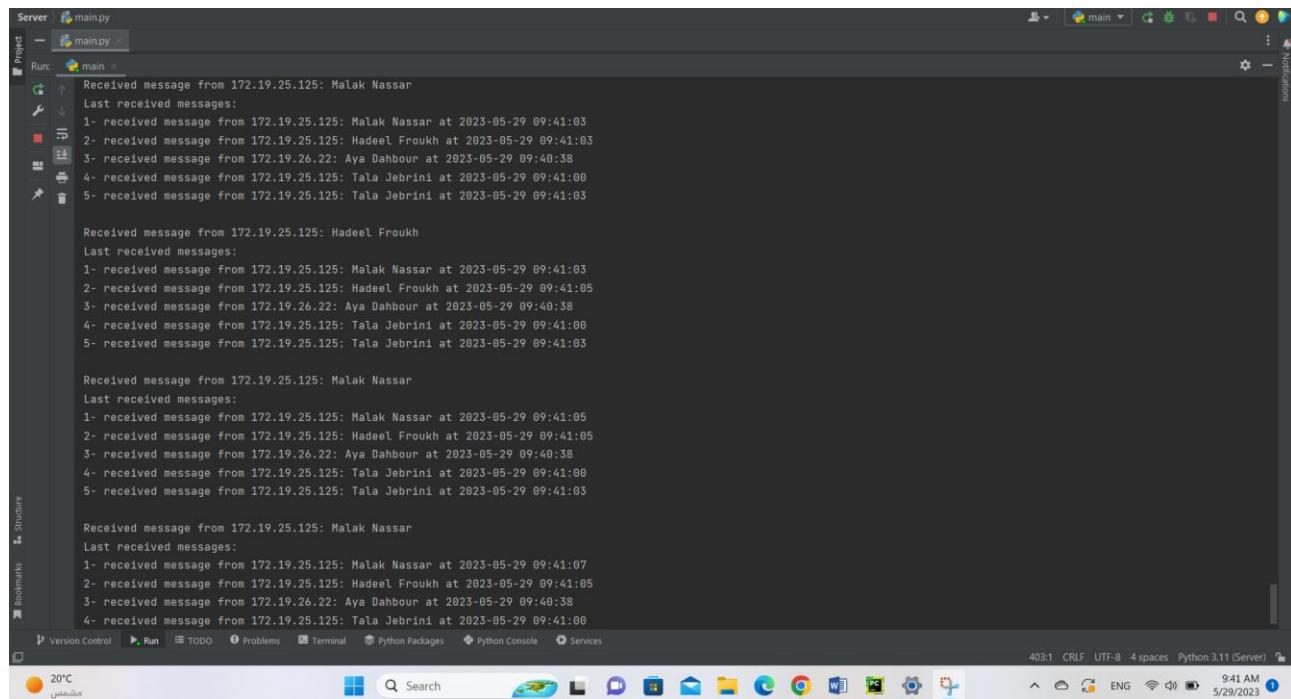
- Project:** A project named "client" is open, containing a file "main.py".
- Code Content:**

```
1 import socket
2 import time
3
4 # Client configuration
5 server_address = ('172.19.26.22', 8855)
6 buffer_size = 1024
7 broadcast_interval = 2 # in seconds
8 student_name = "Hadeel Froukh"
9
10 # Create a UDP socket
11 client_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
12 client_socket.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)
13
14 # Client loop
15 while True:
16     # Send broadcast message
17     message = student_name.encode()
18     client_socket.sendto(message, server_address)
19
20     # Wait for the next broadcast interval
21     time.sleep(broadcast_interval)
22
23 # Close the socket (unreachable in this example)
24 client_socket.close()
```
- Toolbars and Status Bar:** The toolbar includes "Run", "Version Control", "Run", "Python Packages", "TODO", "Python Console", "Problems", "Terminal", and "Services". The status bar at the bottom right shows "Activate Windows Go to Settings to activate Windows.", "16:29 CRLF UTF-8 4 spaces Python 3.9 (client)", and the date/time "31/05/2023 11:08 PM".

Figure 9:client Hadeel

The UDP client code creates clients that send broadcast messages every 2 seconds. Each message includes the client's name. The clients broadcast the messages to the network, allowing the server to receive and process them. Multiple clients can run simultaneously, and the server differentiates between them using their IP addresses. The clients continuously send messages, allowing the server to display the last received message from each client.

When we run them:



```
Server main.py
Project - main.py
Run: main x
Received message from 172.19.25.125: Malak Nassar
Last received messages:
1- received message from 172.19.25.125: Malak Nassar at 2023-05-29 09:41:03
2- received message from 172.19.25.125: Hadeel Froukh at 2023-05-29 09:41:03
3- received message from 172.19.26.22: Aya Dahbour at 2023-05-29 09:40:38
4- received message from 172.19.25.125: Tala Jebrini at 2023-05-29 09:41:03
5- received message from 172.19.25.125: Tala Jebrini at 2023-05-29 09:41:03

Received message from 172.19.25.125: Hadeel Froukh
Last received messages:
1- received message from 172.19.25.125: Malak Nassar at 2023-05-29 09:41:03
2- received message from 172.19.25.125: Hadeel Froukh at 2023-05-29 09:41:03
3- received message from 172.19.26.22: Aya Dahbour at 2023-05-29 09:40:38
4- received message from 172.19.25.125: Tala Jebrini at 2023-05-29 09:41:03
5- received message from 172.19.25.125: Tala Jebrini at 2023-05-29 09:41:03

Received message from 172.19.25.125: Malak Nassar
Last received messages:
1- received message from 172.19.25.125: Malak Nassar at 2023-05-29 09:41:03
2- received message from 172.19.25.125: Hadeel Froukh at 2023-05-29 09:41:03
3- received message from 172.19.26.22: Aya Dahbour at 2023-05-29 09:40:38
4- received message from 172.19.25.125: Tala Jebrini at 2023-05-29 09:41:03
5- received message from 172.19.25.125: Tala Jebrini at 2023-05-29 09:41:03

Received message from 172.19.25.125: Malak Nassar
Last received messages:
1- received message from 172.19.25.125: Malak Nassar at 2023-05-29 09:41:03
2- received message from 172.19.25.125: Hadeel Froukh at 2023-05-29 09:41:03
3- received message from 172.19.26.22: Aya Dahbour at 2023-05-29 09:40:38
4- received message from 172.19.25.125: Tala Jebrini at 2023-05-29 09:41:03
5- received message from 172.19.25.125: Tala Jebrini at 2023-05-29 09:41:03
```

Figure 10: UDP client and server implementation output

3. Part 3:

Using socket programming, implement a simple but a complete web server in go, python, java or C that is listening on port 9977. 3.1. if the request is / or /index.html or /main_en.html or /en (for example localhost:9977/ or localhost:9977/en) then the server should send main_en.html file with Content-Type: text/html. 1.if the request is '/':

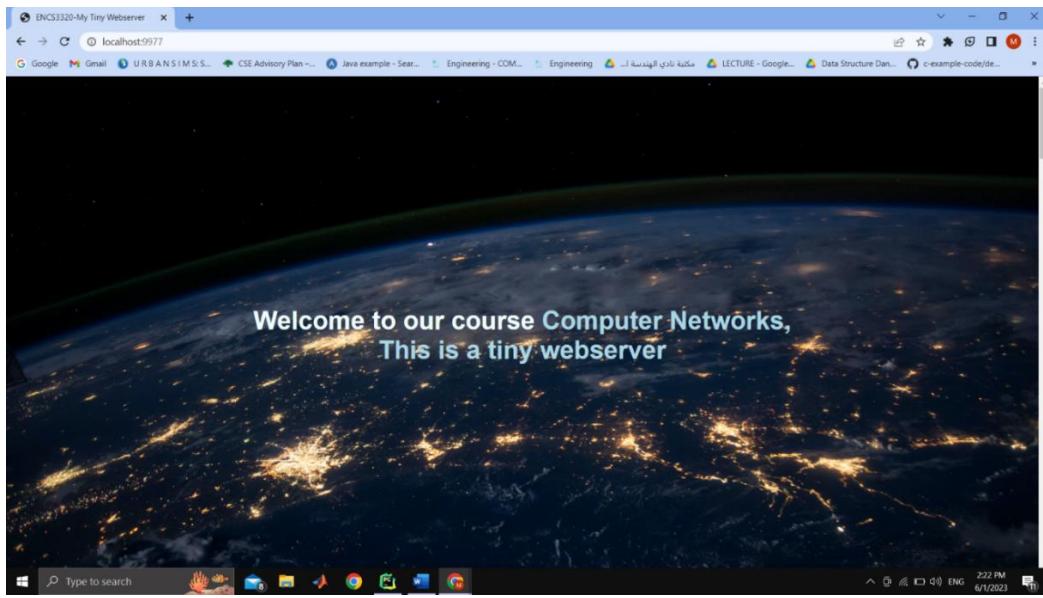


Figure 11:request '/'

The HTTP response:

```
pythonNetworkProject | main.py
File Edit View Navigate Code Refactor Run Tools VCS Window Help pythonNetworkProject - main.py
Run: main >
the [Server] is ready to receive .....
GET /style.css HTTP/1.1
Host: localhost:9977
Connection: keep-alive
sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "Not:A-Brand";v="99"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
sec-ch-ua-platform: "Windows"
Accept: text/css,*/*;q=0.1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: style
Referer: http://localhost:9977/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.8,ar;q=0.8,pt;q=0.7
Cookie: Pycharm-66859dc94a51bcf-df73-45a6-be8f-6ec316406b51; amp_a0683b=qNx09m7kG-67fEaExzfzEh.NmpxNTYzD0M5aTF4bQ=.1hisIntqn.1his44k1r.0.0.0

the [Server] is ready to receive .....
GET /images/nasa-017b3SHj8-unplash.jpg HTTP/1.1
Host: localhost:9977
Connection: keep-alive
sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "Not:A-Brand";v="99"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
sec-ch-ua-platform: "Windows"
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*;q=0.8
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: image
Referer: http://localhost:9977/style.css
Accept-Encoding: gzip, deflate, br
```

Figure 12:HTTP response for '/'

2.if request '/index.html':

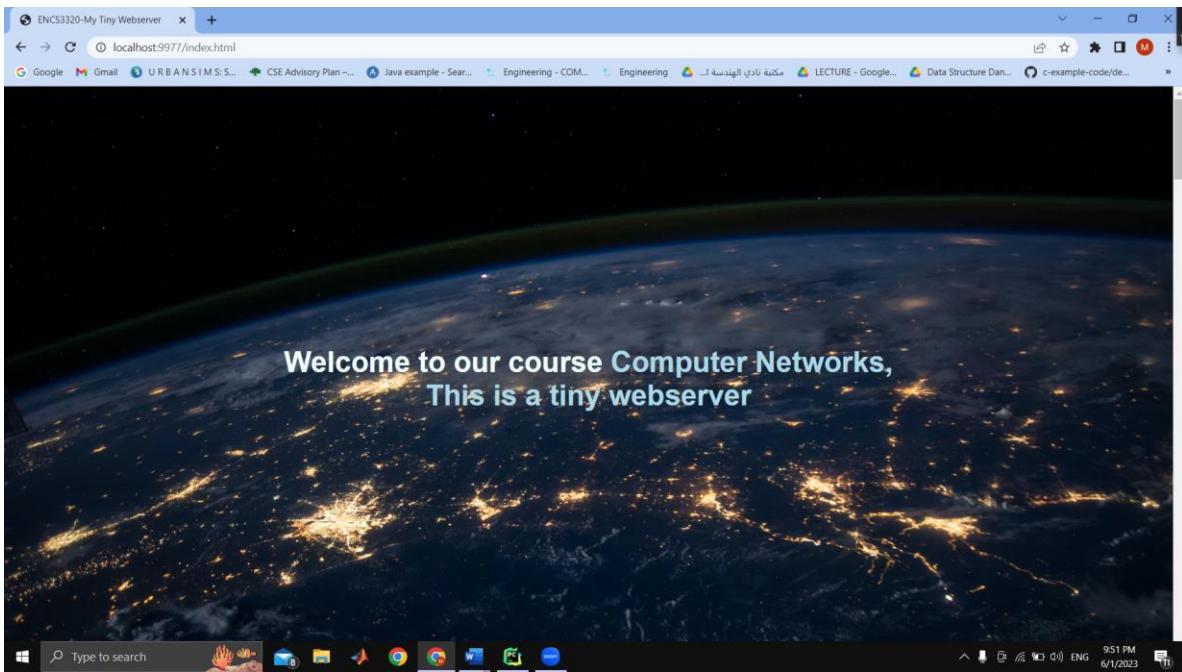


Figure 13:request index.html

The http response:

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help pythonNetworkProject - main.py
pythonNetworkProject main.py
Project Project Run main.py main_any.html main_en.html style.css main_ar.html error.html
Run main
the [Server] is ready to receive .....
GET /style.css HTTP/1.1
Host: localhost:9977
Connection: keep-alive
sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "Not-A-Brand";v="99"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
sec-ch-ua-platform: "Windows"
Accept: text/css,*/*;q=0.1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: style
Referer: http://localhost:9977/index.html
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,ar;q=0.8,pt;q=0.7
Cookie: Pycharm-66859dbc=94a51bcf-df73-45a6-be8f-6ec316486b31; amp_a0683b=qNx09m7kG-67fEaExzfzEh.NmpxNTYzd0M5aTF4bQ=..1h1s1ntqn.1h1s4jq47.0.0.0

the [Server] is ready to receive .....
GET /images/nasa-Q1p7bh3SHJ8-unplash.jpg HTTP/1.1
Host: localhost:9977
Connection: keep-alive
sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "Not-A-Brand";v="99"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
sec-ch-ua-platform: "Windows"
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
```

Figure 14:HTTP response for request index.html

4. if request '/main_en.html':

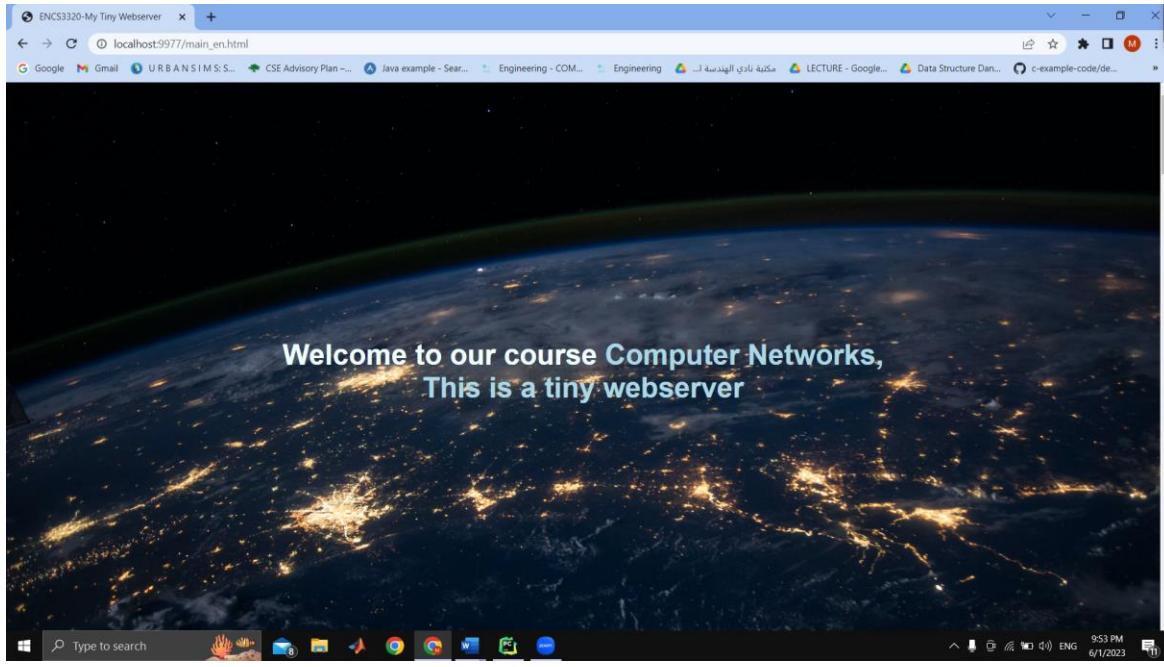


Figure 15:request 'main_en.html'

The HTTP response:

```
the [Server] is ready to receive .....  
GET /style.css HTTP/1.1  
Host: localhost:9977  
Connection: keep-alive  
sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "Not:A-Brand";v="99"  
sec-ch-ua-mobile: ?0  
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36  
sec-ch-ua-platform: "Windows"  
Accept: text/css,*/*;q=0.1  
Sec-Fetch-Site: same-origin  
Sec-Fetch-Mode: no-cors  
Sec-Fetch-Dest: style  
Referer: http://localhost:9977/main_en.html  
Accept-Encoding: gzip, deflate, br  
Accept-Language: en-US,en;q=0.9,ar;q=0.8,pt;q=0.7  
Cookie: Pycharm-66859dbc-94a51bcf-df73-45a6-be8f-6ec316486b31; amp_a0683b=qNx09m7Kg-67fEaExzfzEh.NmpxNTYzdDM5aTF4bQ==..1h1s1ntqn.1h1s4qcgu.0.0.0  
  
the [Server] is ready to receive .....  
GET /images/nasa-Qip7bh3Shj8-unsplash.jpg HTTP/1.1  
Host: localhost:9977  
Connection: keep-alive  
sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "Not:A-Brand";v="99"  
sec-ch-ua-mobile: ?0  
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36  
sec-ch-ua-platform: "Windows"
```

Figure 16:HTTP response request '/main_en.html'

4.if the request '/en' :

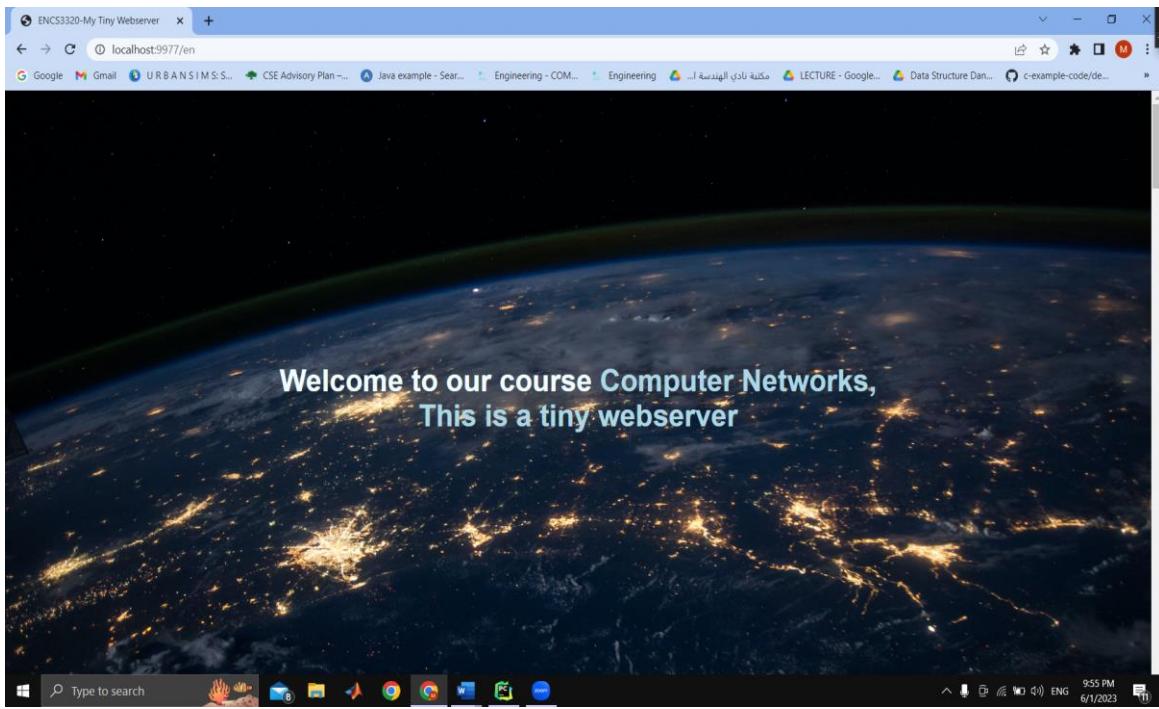


Figure 17:Figure 15:request 'en'

The HTTP response:

```

pythonNetworkProject  main.py
File Edit View Navigate Code Behavior Run Tools VCS Window Help pythonNetworkProject - main.py
Project  main.py  main_any.html  main_en.html  style.css  main_ar.html  error.html
Run: main ×
GET /style.css HTTP/1.1
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,ar;q=0.8,pt;q=0.7
Cookie: Pycharm-66859dbc=94a51bcf-df73-45a6-be8f-6ec316486b31; amp_a0683b=qNx09m7kG-67fEaExzfzEh.NmpxNTYzd0MSaTf4bQ=..1h1s1ntqn.1h1s4tpeo.0.0.0
the [Server] is ready to receive .....
GET /style.css HTTP/1.1
Host: localhost:9977
Connection: keep-alive
sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "Not:A-Brand";v="99"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
sec-ch-ua-platform: "Windows"
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/*,*;q=0.8,application/signed-exchange;v=b3;q=0.7
Purpose: prefetch
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: style
Referer: http://localhost:9977/en
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,ar;q=0.8,pt;q=0.7
Cookie: Pycharm-66859dbc=94a51bcf-df73-45a6-be8f-6ec316486b31; amp_a0683b=qNx09m7kG-67fEaExzfzEh.NmpxNTYzd0MSaTf4bQ=..1h1s1ntqn.1h1s4tpeo.0.0.0

the [Server] is ready to receive .....
GET /images/nasa-Qip7hJSHj8-unsplash.jpg HTTP/1.1
Host: localhost:9977
Connection: keep-alive
sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "Not:A-Brand";v="99"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
sec-ch-ua-platform: "Windows"

```

Figure 18:HTTP response request 'en'

3.2: when the request is /ar:

Then the server response with main_ar.html which is an Arabic version of main_en.htm

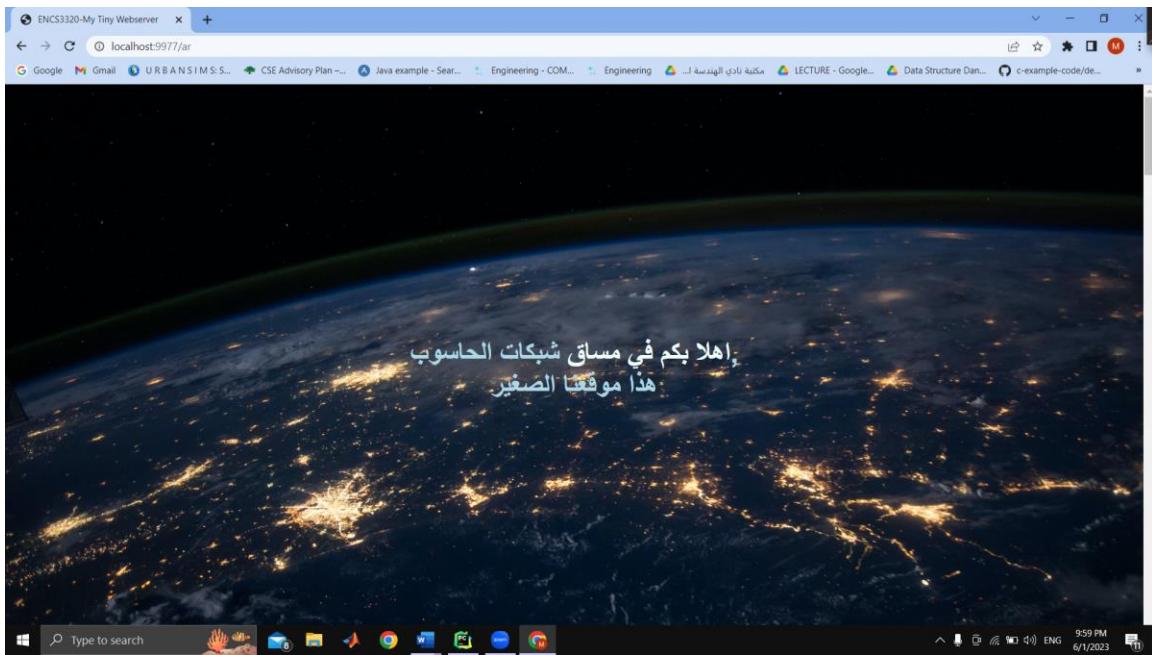


Figure 19:request 'ar'

The http response:

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help pythonNetworkproject - main.py
pythonNetworkproject main.py
Project main
Run: main
Cookie: Pycharm-66859dbc=94a51bcf-df73-45a6-be8f-6ec316486b31; amp_a0683b=qNx09m7kG-67fEaExzfzEh.NmpxNTYzdDM5aTF4bQ==..1h1s1ntqn.1h1s50rj2.0.0.0

the [Server] is ready to receive .....
GET /style.css HTTP/1.1
Host: localhost:9977
Connection: keep-alive
sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "Not:A-Brand";v="99"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
sec-ch-ua-platform: "Windows"
Accept: text/css,*/*;q=0.1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: style
Referer: http://localhost:9977/ar
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,ar;q=0.8,pt;q=0.7
Cookie: Pycharm-66859dbc=94a51bcf-df73-45a6-be8f-6ec316486b31; amp_a0683b=qNx09m7kG-67fEaExzfzEh.NmpxNTYzdDM5aTF4bQ==..1h1s1ntqn.1h1s50rj2.0.0.0

the [Server] is ready to receive .....
GET /images/nasa-Q1p7bhJSHj8-unsplash.jpg HTTP/1.1
Host: localhost:9977
Connection: keep-alive
sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "Not:A-Brand";v="99"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
sec-ch-ua-platform: "Windows"
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*/*;q=0.8
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
```

A screenshot of the PyCharm IDE showing the "Run" tool window. It displays the HTTP request log for the "/ar" endpoint. The log shows two requests: one for the CSS file "style.css" and another for the image "nasa-Q1p7bhJSHj8-unsplash.jpg". Both requests are from a user agent identified as "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36". The requests include standard headers like "Host", "Connection", "sec-ch-ua", "User-Agent", "Accept", "Sec-Fetch-Site", "Sec-Fetch-Mode", and "Sec-Fetch-Dest". The cookie "Pycharm-66859dbc" is also present in both requests.

Figure 20:HTTP response request 'ar'

3.3:ends with .html:

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.

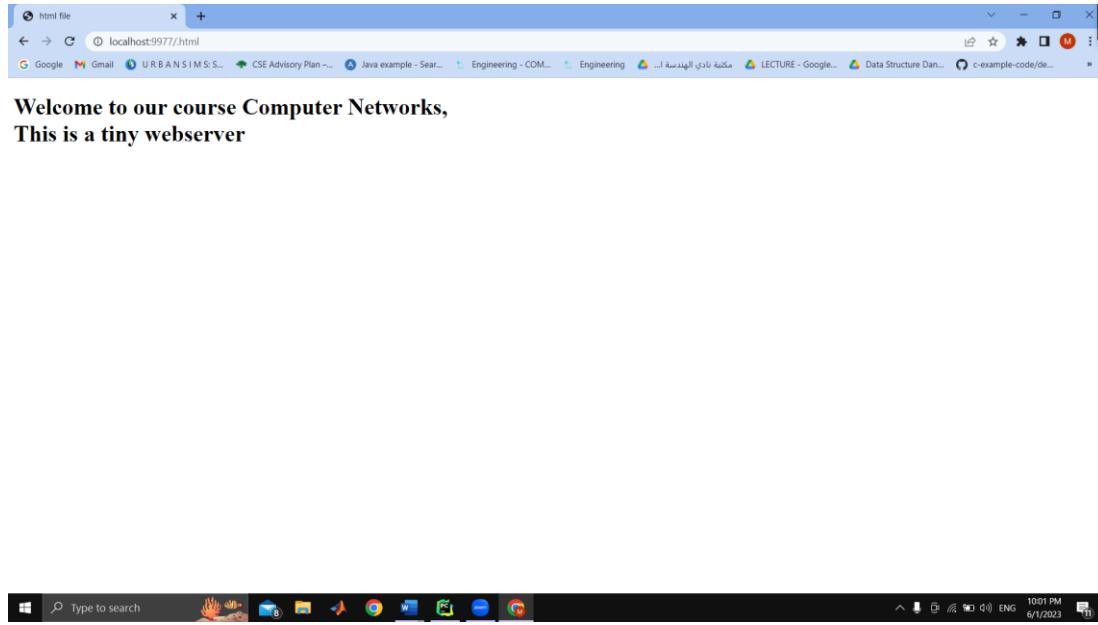


Figure 21:request '.html'

The http response:

The screenshot shows a terminal window titled 'pythonNetworkproject - main.py'. It displays the server's log output and the client's request headers. The server logs show 'the [Server] is ready to receive'. The client's request headers include 'GET /html HTTP/1.1', 'Host: localhost:9977', 'Connection: keep-alive', 'Cache-Control: max-age=0', 'Sec-Ch-Ua: "Chromium";v="112", "Google Chrome";v="112", "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/112.0.0.0 Safari/537.36', 'Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7', 'Sec-Fetch-Site: none', 'Sec-Fetch-Mode: navigate', 'Sec-Fetch-User: ?1', 'Sec-Fetch-Dest: document', 'Accept-Encoding: gzip, deflate, br', 'Accept-Language: en-US,en;q=0.9,ar;q=0.8,pt;q=0.7', and 'Cookie: Pycharm-66859dbc=94a51bcf-df73-45a6-be8f-6ec316486b31; amp_a0683b=qNx09m7kG-67fEaExzfzEh.NmpxNTYzdDM5aTF4bQ==..1h1s1ntqn.1h1s5asp8.0.0.0'. The terminal window has a dark theme and includes a status bar at the bottom.

Figure 22:HTTP response request '.html'

Ends with.css:

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

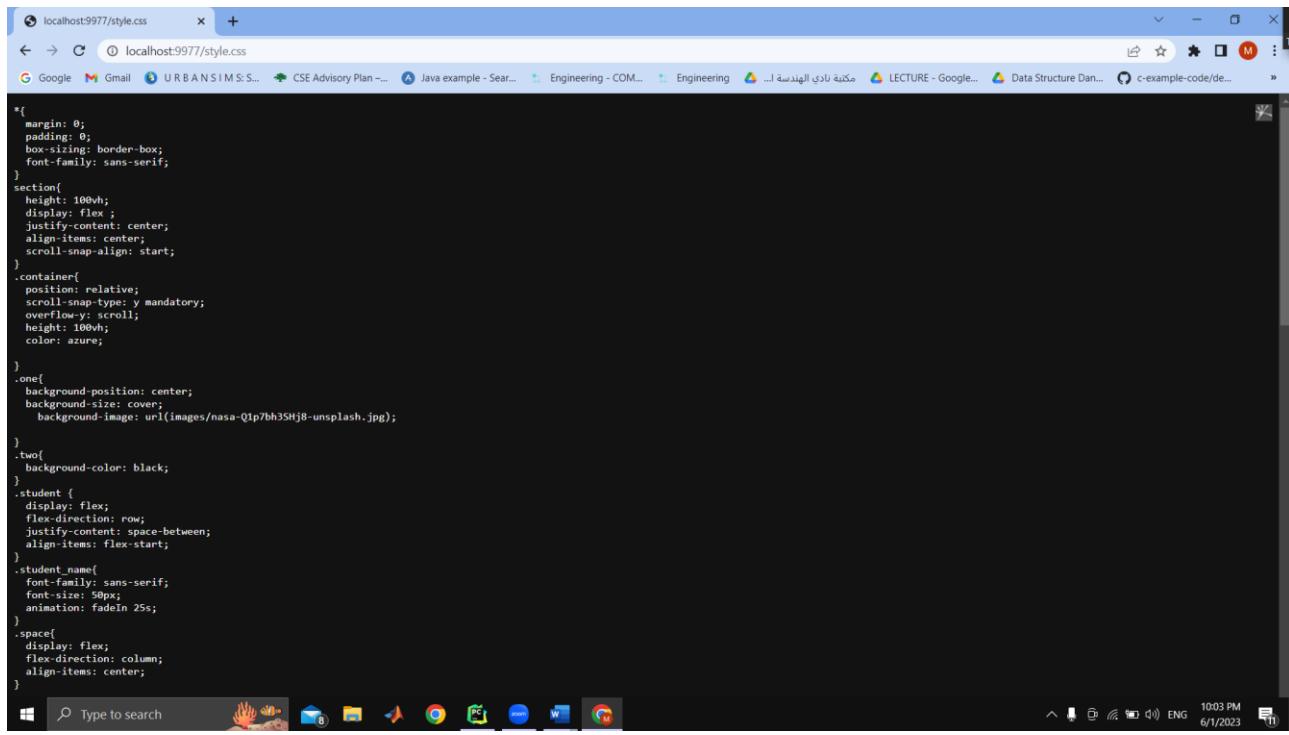


Figure 23::request '.css'

The http response:

The screenshot shows a PyCharm IDE with a terminal window displaying an HTTP request log. The log shows a GET request for '/style.css' from 'localhost:9977'. The request headers include 'Sec-Fetch-Mode: navigate', 'Sec-Fetch-User: ?1', 'Sec-Fetch-Dest: document', 'Accept-Encoding: gzip, deflate, br', 'Accept-Language: en-US,en;q=0.9,ar;q=0.8,pt;q=0.7', and a cookie. The response message in the terminal says 'the [Server] is ready to receive'. The status bar at the bottom right indicates the date as 6/1/2023 and the time as 10:04 PM.

```
the [Server] is ready to receive ......

the [Server] is ready to receive ......

GET /style.css HTTP/1.1
Host: localhost:9977
Connection: keep-alive
Cache-Control: max-age=8
sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "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/112.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,ar;q=0.8,pt;q=0.7
Cookie: Pycharm-66859dbc=94a51bcf-df73-45a6-be8f-6ec316486b31; amp_a0683b=qNx09m7kG-67fEaExzfzEh.NmpxNTYzdDM5aTF4bQ==..1hls1ntqn.1hls5eb0a.0.0.0

the [Server] is ready to receive .....
```

Figure 24:HTTP response request '.css'

Ends with .png:

if the request is a .png then the server should send the png image with Content-Type: image/png.

You can use any image.

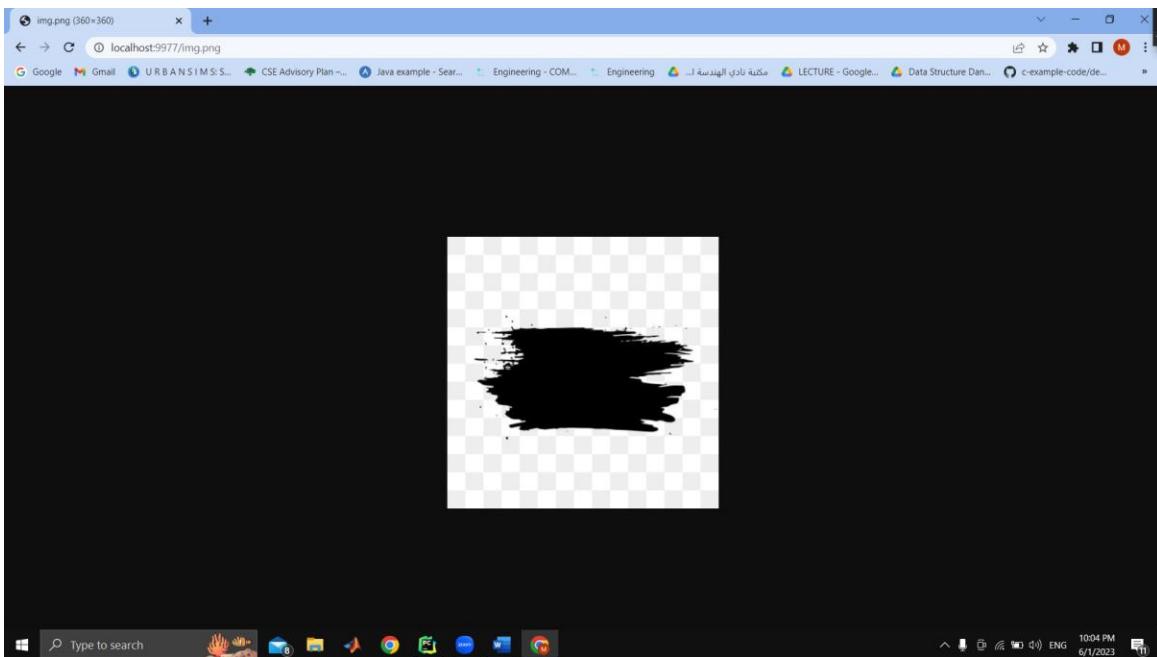


Figure 25:request '.png'

```
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,ar;q=0.8,pt;q=0.7
Cookie: Pycharm-66859dbc94a51bcf-df73-45a6-be8f-6ec316486b31; amp_a0683b=qNx09m7kG-67fEaExzfzEh.NmpxNTYzd0M5aTF4bQ==..1h1s1ntqn.1h1s5fk5v.0.0.0

the [Server] is ready to receive .....
GET /img.png HTTP/1.1
Host: localhost:9977
Connection: keep-alive
sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "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/112.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,ar;q=0.8,pt;q=0.7
Cookie: Pycharm-66859dbc94a51bcf-df73-45a6-be8f-6ec316486b31; amp_a0683b=qNx09m7kG-67fEaExzfzEh.NmpxNTYzd0M5aTF4bQ==..1h1s1ntqn.1h1s5h77a.0.0.0

the [Server] is ready to receive .....
|
```

The screenshot shows the PyCharm IDE's terminal or logs pane displaying an HTTP request for a .png file. The request includes standard headers like User-Agent, Accept, and Sec-Fetch-Mode, along with a cookie. The response message "the [Server] is ready to receive" is shown twice, indicating the server is listening and has received the request.

Figure 26:HTTP response request '.png'

Ends with jpg:

XV | Page

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

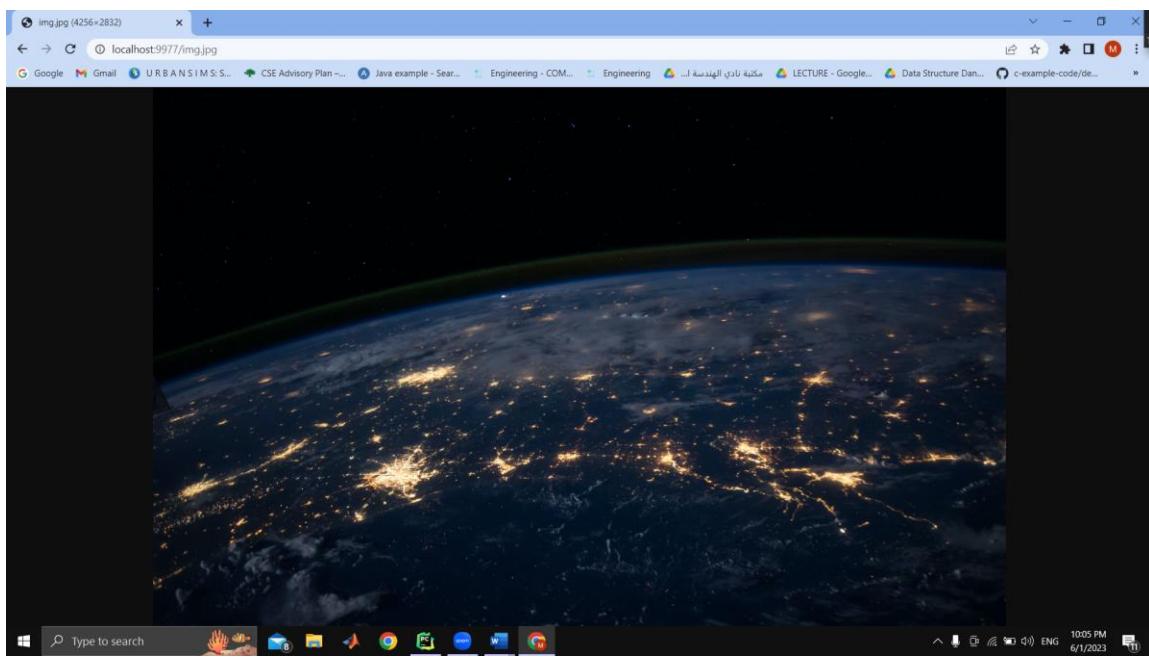


Figure 27:request '.jpg'

The HTTP response:

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help pythonNetworkProject - main.py
pythonNetworkProject | main.py
Project | Run: main | X
  Sec-Fetch-Site: none
  Sec-Fetch-Mode: navigate
  Sec-Fetch-User: ?1
  Sec-Fetch-Dest: document
  Accept-Encoding: gzip, deflate, br
  Accept-Language: en-US,en;q=0.9,ar;q=0.8,pt;q=0.7
  Cookie: Pycharm-66859dbc94a51bcf-df73-45a6-be8f-6ec316486b31; amp_a0683b=qNx09m7kG-67fEaExzfzEh.NmpxNTYzdDM5aTF4bQ=..1h1s1ntqn.lhls5h77a.0.0.0

the [Server] is ready to receive ......

the [Server] is ready to receive .....
GET /img.jpg HTTP/1.1
Host: localhost:9977
Connection: keep-alive
sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "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/112.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,ar;q=0.8,pt;q=0.7
Cookie: Pycharm-66859dbc94a51bcf-df73-45a6-be8f-6ec316486b31; amp_a0683b=qNx09m7kG-67fEaExzfzEh.NmpxNTYzdDM5aTF4bQ=..1h1s1ntqn.lhls5i44q.0.0.0

the [Server] is ready to receive .....
```

Figure 28:HTTP response request '.jpg'

Use the status code 307 Temporary Redirect to redirect the following

- If the request is /yt then redirect to youtube website

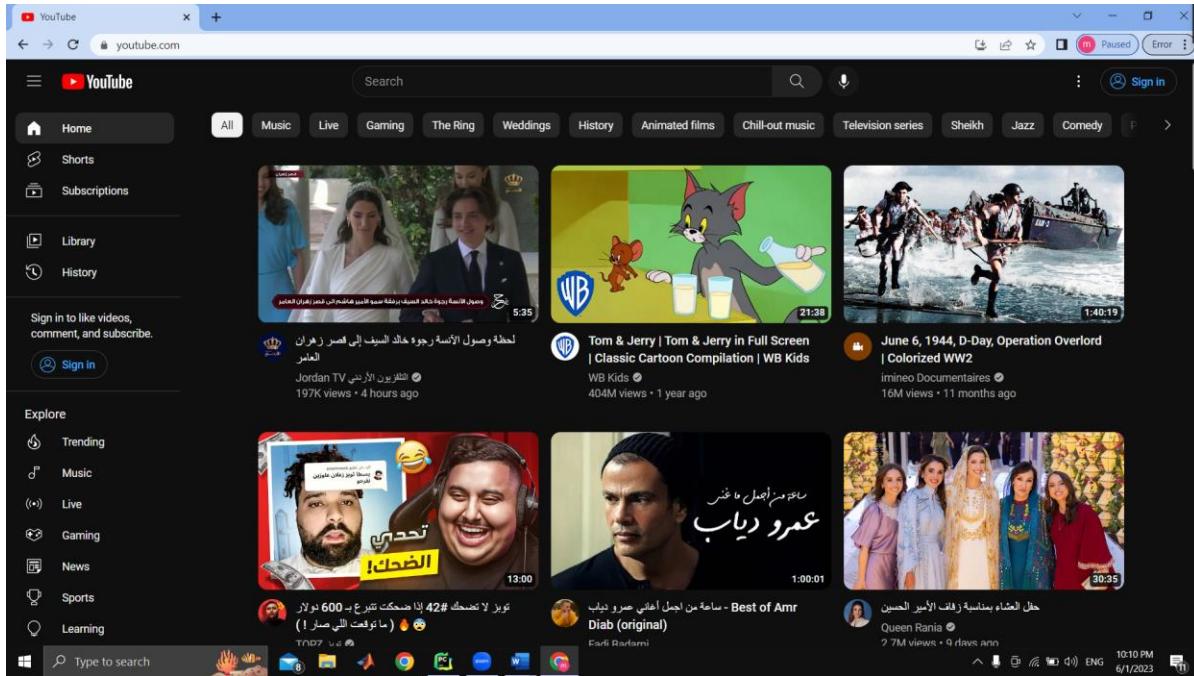
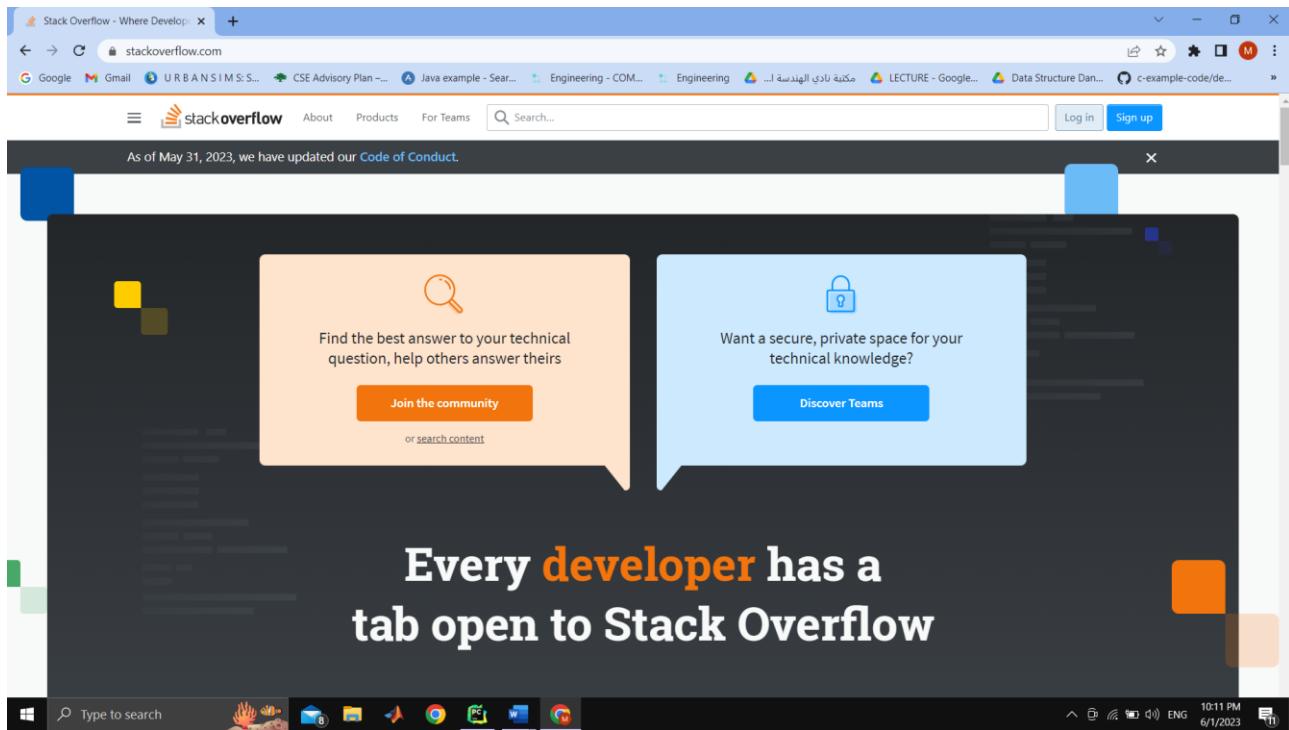


Figure 29:request 'yt'

A screenshot of a terminal window titled 'main.py'. The window shows a list of files including 'main.py', 'main_any.html', 'main_en.html', 'style.css', 'main_ar.html', and 'error.html'. The main area of the terminal displays an incoming HTTP request. The request starts with 'GET /yt HTTP/1.1', followed by headers: 'Host: localhost:9977', 'Connection: keep-alive', 'sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "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/112.0.0.0 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=0.9'. The response message 'the [Server] is ready to receive' is repeated multiple times, indicating a temporary redirect loop.

Figure 30:HTTP response request 'yt'

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



The http response:

```
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,ar;q=0.8,pt;q=0.7
Cookie: Pycharm-66859dbc=94a51bcf-df73-45a6-be8f-6ec316486b31; amp_a0683b=qNx09m7K6-67fEaExzfzEh.NmpxNTYzdDM5aTF4bQ=..1h1s1ntqn.1h1s5v06r.0.0.0

the [Server] is ready to receive .....
GET /so HTTP/1.1
Host: localhost:9977
Connection: keep-alive
sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "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/112.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,ar;q=0.8,pt;q=0.7
Cookie: Pycharm-66859dbc=94a51bcf-df73-45a6-be8f-6ec316486b31; amp_a0683b=qNx09m7K6-67fEaExzfzEh.NmpxNTYzdDM5aTF4bQ=..1h1s1ntqn.1h1s5v7gv.0.0.0

the [Server] is ready to receive .....
```

Figure 31:HTTP response request 'so'

c. If the request is /rt then redirect to ritaj website



HTTP response:

```

Type to search
File Edit View Navigate Code Refactor Run Tools VCS Window Help pythonNetworkproject - main.py
pythonNetworkproject main.py
Project main.py - main.html - main_ar.html - style.css - main_en.html - error.html
Run: main
the [Server] is ready to receive .....  

the [Server] is ready to receive .....  

GET /rt HTTP/1.1  

Host: localhost:9977  

Connection: keep-alive  

sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "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/112.0.0.0 Safari/537.36  

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7  

Sec-Fetch-Site: none  

Sec-Fetch-Mode: navigate  

Sec-Fetch-User: ?1  

Sec-Fetch-Dest: document  

Accept-Encoding: gzip, deflate, br  

Accept-Language: en-US,en;q=0.9,ar;q=0.8,pt;q=0.7  

Cookie: Pycharm-66859dbc=94a51bcf-df73-45a6-be8f-6ec316486b31; amp_a0683b=qNx09m7k6-67fEaExzfzEh.NmpxNTYzdDM5aTF4bQ=..1h1s1ntqn.1h1s5v7gv.0.0.0  

the [Server] is ready to receive .....  

the [Server] is ready to receive .....  


```

Error page :

If the request is wrong or the file doesn't exist



The http:

A screenshot of the PyCharm IDE. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help, and the current file "pythonNetworkproject - main.py". The bottom status bar shows "10:14 PM 6/1/2023 ENG".

the [Server] is ready to receive

the [Server] is ready to receive

the [Server] is ready to receive

```
GET /ssssss HTTP/1.1
Host: localhost:9977
Connection: keep-alive
sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112", "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/112.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,ar;q=0.8,pt;q=0.7
Cookie: Pycharm-66859dbc=94a51bcf-df73-45a6-be8f-6ec316486b31; amp_a0683b=qNx09m7kG-67fEaExzfzEh.NmpxNTYzdDM5aTF4bQ==..1h1s1ntqn.1h1s5v7gv.0.0.0
```

the [Server] is ready to receive

the [Server] is ready to receive

The socket program:

```
from socket import *

# server_address=(' ',9977)# assign the port
serverport = 9977
serverSocket = socket(AF_INET, SOCK_STREAM) # create a tcp server(create the
socket)
serverSocket.bind((' ', serverport)) # connected the server to the port
serverSocket.listen(1) # listen= every to see every request and accept it
while True:
    print("the [Server] is ready to receive ..... ")
    connectionSocket, client_address = serverSocket.accept() # accept, opens a
new socket
    data = connectionSocket.recv(1048).decode() # reads the data from the
client
    # print("IP:"+client_address[0]+ "port" + str(client_address))
    print(data)
    IP = client_address[0]
    port = client_address[1]
    browser_lines = data.split('\n')
    browser = browser_lines[0]
    client_req_url = ""
    if len(browser_lines) > 1:
        client_req_url = browser.split()[1]
        try:
            if client_req_url == '/' or client_req_url == '/index.html' or
client_req_url == '/main_en.html' or client_req_url == '/en':
                connectionSocket.send('HTTP/1.1 200 OK \r\n'.encode())
                connectionSocket.send("Content-Type:text/html\r\n".encode())
                connectionSocket.send("\r\n".encode())
                html_english = open("main_en.html", encoding="utf-8").read()
                connectionSocket.send(html_english.encode())
            elif client_req_url == '/ar' or
client_req_url.endswith('main_ar.html'):
                connectionSocket.send('HTTP/1.1 200 OK \r\n'.encode())
                connectionSocket.send("Content-Type:text/html\r\n".encode())
                connectionSocket.send("\r\n".encode())
                html_arabic = open("main_ar.html", encoding="utf-8").read()
                connectionSocket.send(html_arabic.encode())
            elif client_req_url == '/.html':
                connectionSocket.send("HTTP/1.1 200 OK \r\n".encode())
                connectionSocket.send("Content-Type:text/html\r\n".encode())
                connectionSocket.send("\r\n".encode())
                any_html = open("main_any.html", "rb").read() # open english
html file
                connectionSocket.send(any_html)
            elif client_req_url == '/style.css' or
client_req_url.endswith('.css'):
                connectionSocket.send("HTTP/1.1 200 OK \r\n".encode())
                connectionSocket.send("Content-Type:text/css\r\n".encode())
                connectionSocket.send("\r\n".encode())
                style_css = open("style.css", "rb").read() # open english html
file
                connectionSocket.send(style_css)

            elif client_req_url.endswith(".png"):
                connectionSocket.send("HTTP/1.1 200 OK \r\n".encode())
        
```

```

connectionSocket.send("Content-Type:image/png\r\n".encode())
connectionSocket.send("\r\n".encode())
png_image = open("images/black.png", "rb").read() # open
english html file
    connectionSocket.send(png_image)
elif client_req_url.endswith(".jpg"):
    connectionSocket.send("HTTP/1.1 200 OK \r\n".encode())
    connectionSocket.send("Content-Type:image/jpeg\r\n".encode())
    connectionSocket.send("\r\n".encode())
    jpg_image = open("images/nasa-Q1p7bh3SHj8-unsplash.jpg",
"rb").read() # open english html file
    connectionSocket.send(jpg_image)
elif client_req_url == '/yt':
    redirect_link=b'HTTP/1.1 307 Temporary
Redirect\nLOCATION:https://www.youtube.com/\n\n'
    connectionSocket.sendall(redirect_link)
elif client_req_url == '/so':
    redirect_link=b'HTTP/1.1 307 Temporary
Redirect\nLOCATION:https://stackoverflow.com/\n\n'
    connectionSocket.sendall(redirect_link)
elif client_req_url == '/rt':
    redirect_link=b'HTTP/1.1 307 Temporary
Redirect\nLOCATION:https://ritaj.birzeit.edu/register/\n\n'
    connectionSocket.sendall(redirect_link)
else:
    # Send a 404 Not Found response
    connectionSocket.send("HTTP/1.1 404 Not Found\r\n".encode())
    connectionSocket.send('Content-Type: text/html\r\n'.encode())
    connectionSocket.send('\r\n'.encode())
    connectionSocket.send('\r\n'.encode())
    f = """
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8"/>
    <title>Error-404</title>
</head>
<style>
    body {{
        background: #bcefff;
        background-image: -webkit-gradient(radial, 50% 50%, 0,
50% 50%, 800);
        padding-top: 5%;
        padding-left: 40%;
    }}
    img {{
        width: 250px;
        height: 250px;
    }}
    .names {{
        margin-top: 50px;
        margin-bottom: 50px;
        font: bold;
    }}
</style>
<body>

```

```


# The file is not found



Malak Nassar - 1200757



Tala Jebrini - 1200493



Aya Dahbour - 1201738



Hadeel Froukh - 1201585



IP: {ip} Port: {port}


```

```

response = f.format(ip=IP, port=port)
connectionSocket.send(response.encode())

```

```

finally:
    connectionSocket.close()

```

the HTML code:

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>ENCS3320-My Tiny Webserver</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <div class="container">
        <section class="one">
            <div class="header">
                <h1>Welcome to our course <span class="highlight"> Computer Networks,</span><br>
                <span class="centered-text">This is a tiny webserver</span></h1>
            </div>
        </section>
        <section class="one">
            <span class="pa">
                <p> <br> Our group members<br><br><hr/><br>Malak Nassar - 1200757<br><br><br>Tala Jebrini - 1200493<br><br><br>Aya Dahbour - 1201738<br><br><br>Hadeel Froukh - 1201585</p>
            </span>
        </section>
        </section>
        <span class="studnet">

```



```

student<br><br>Some the projects that I have done during different
courses:<br><br>Java Project.
<br><br>Data Structure Projects.<br><br>My hobbies are reading
and coding</span>
</section>

</section>
<section class="six">
<span class="li">

    <p>   <a
    href="https://www.w3schools.com/python/gloss_python_multi_line_strings.asp">
<h4> Python MultiLine Strings <br><br></h4></a></p>
        <p> <a href="main_ar.html"> <h4> To arabic HTML file</h4></a></p>
    </span>

</section>

</div>

</body>
</html>

```

The Arabic html:

```

<!DOCTYPE html>
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>ENCS3320-My Tiny Webserver</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>

    <!-- content goes here -->

    <!-- content goes here -->

    <div class="container">
        <section class="one">
            <div class="header">
                <h1> مساق في بكم اهلاً <span class="highlight"> شبكات </span><br>
                <span class="centered-text">الصغير موقعنا هذا</span></h1>
            </div>
        </section>
        <section class="one">
            <span class="pa">
                - نصار ملاك <br> الفريق أعضاء <br><br><br><br>
                1200757<br><br><br> 1200493<br><br><br>- جبريني تالا -
                1201738<br><br><br> 1201585</p>
            </span>
        </section>
    </div>

```


p;
المشاريع بعض

 حاسوب هندسة طالبة
جافا مشروع

 بها قمت التي
هو اياتي

 الخوارزميات و المعلومات بنية شاريع
البرمجه و القراءة
</section>

</section>

<section class="six">

<p>

<h4> الخطوط متعدد بايثون نص

</h4></p>

باللغة المموقع ملف الى <h4> الانجليزيه</h4></p>

</section>

</div>

</body>

</html>

The css :

```
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    font-family: sans-serif;
}

section{
    height: 100vh;
    display: flex ;
    justify-content: center;
    align-items: center;
    scroll-snap-align: start;
}

.container{
    position: relative;
    scroll-snap-type: y mandatory;
    overflow-y: scroll;
    height: 100vh;
    color: azure;
}

.one{
    background-position: center;
    background-size: cover;
    background-image: url(images/nasa-Q1p7bh3SHj8-unsplash.jpg);
}

.two{
    background-color: black;
}

.student {
    XXVII | Pa
```

```

display: flex;
flex-direction: row;
justify-content: space-between;
align-items: flex-start;
}
.student_name{
font-family: sans-serif;
font-size: 50px;
animation: fadeIn 25s;
}
.space{
display: flex;
flex-direction: column;
align-items: center;
}

.student_ID {
font-family: sans-serif;
font-size: 50px;
animation: fadeIn 20s;
margin-top: 10px; /* Add some spacing between student_name and student_ID */
}

.student_info {
font-family: sans-serif;
font-size: 50px;
animation: fadeIn 20s;
display: flex;
flex-direction: column-reverse;
align-items: flex-end; /* Align to the right */
margin-right: 100px; /* Add some right margin for spacing */
}
@keyframes fadeIn {
0% { opacity: 0; }
100% { opacity: 1; }
}
.highlight{
/* color: cornflowerblue; */
color: lightblue;
/* color: darkslateblue; */
}
.header{
font-size: 25px;
font-family: cursive;
}
.centered-text {
/* color: darkslateblue; */
color: lightblue;

display: block;
text-align: center;
}
.three{
background-color: black;
}

```

```
.four{
    background-color: black;
}

.five{
    background-color: black;
}

.six{
    background-position: center;
    background-size: cover;
    background-image: url(images/black.png);
}

.p@ {
font-size: 15px;
font-family: sans-serif;
color: bisque;
background-color: cadetblue;
width: 300px;
height: 300px;
text-align: center;
}

.li {
text-align: center;
}

.bpa {
    font-size: 25px;
    font-family: sans-serif;
    color: bisque;
    background-color: rgb(142, 178, 190);
    width: 600px;
    height: 400px;
    text-align: center;
}
```

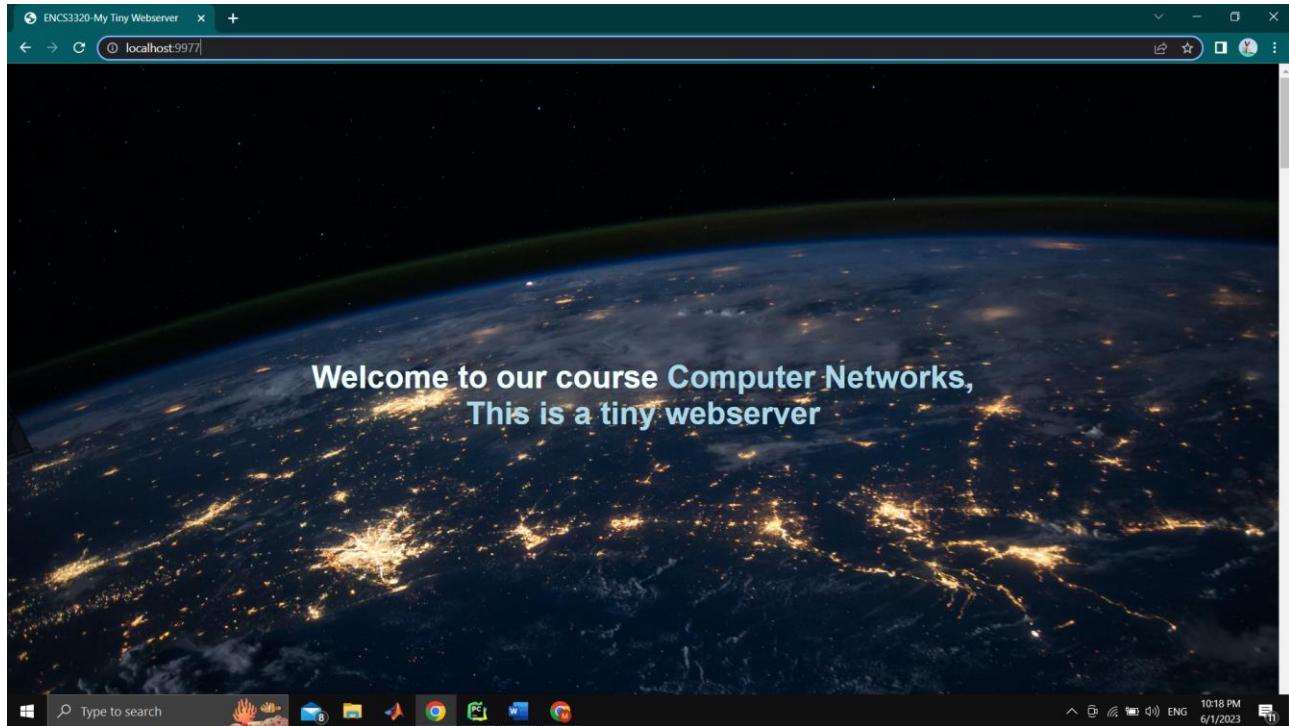
The any html file :

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>html file</title>

</head>
<body>
<h1>Welcome to our course    <span class="highlight"> Computer
Networks,</span><br>
        <span class="centered-text">This is a tiny webserver</span></h1>

</body>
</html>
```

Using another device -> Tala's computer :



The entire page :

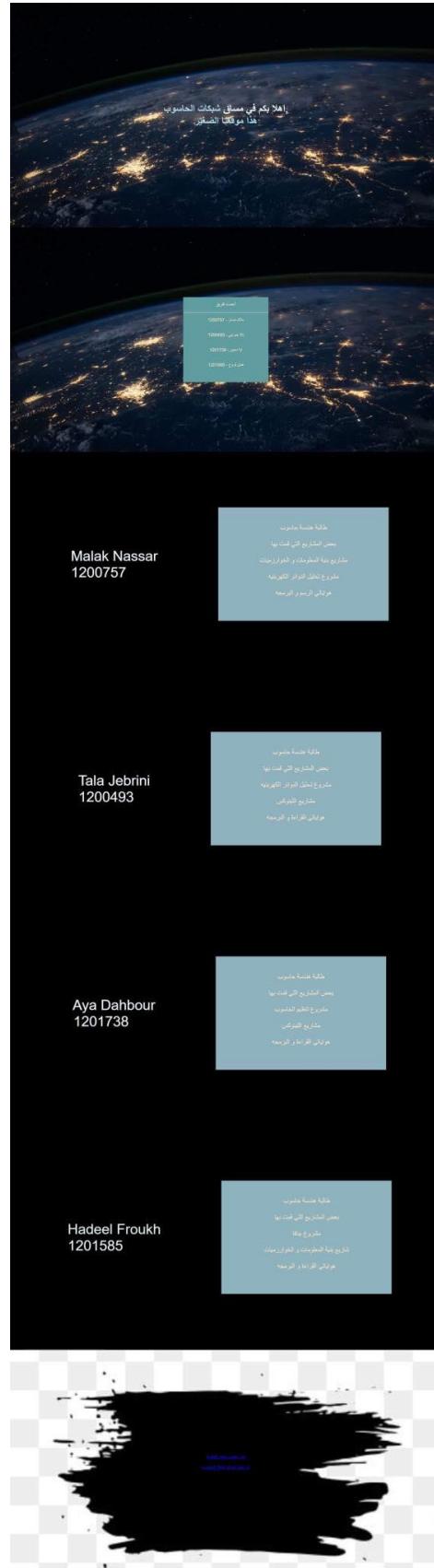


Figure 32:entire arabic page



Figure 33:entire english page

4. Part 4:

Developed a messaging system for seamless peer-to-peer communication. Displays sender's name and message date. Enter line number followed by "D" to access specific content. Smooth setup with documented screenshots.

Server output:

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help peer_to_peer_server - client.py
peer_to_peer_server C:\Users\venu
  venu
    cl.py
    client.py
    main.py
  External Libraries
  Scratches and Consoles

main.py x client.py x cl.py x
1 import threading
2 import socket
3 import time
4
5     # Function to receive messages from the server
6 def receive_messages():
7     while True:
8         data = client_socket.recv(1024).decode()
9         sender_name, message = data.split(':'. 1)

Run: client X main X cl X
C:\Users\HP\PycharmProjects\peer_to_peer_server\venv\bin\python.exe C:\Users\HP\PycharmProjects\peer_to_peer_server\main.py
Server listening on ('localhost', 12345)
New connection from ('127.0.0.1', 50215)
New connection from ('127.0.0.1', 50216)

Bookmarks
Status bar: Type to search 11:40 PM 6/1/2023
```

Peer #1:

The screenshot shows the PyCharm IDE interface with a project named "peer_to_peer_server". The "client.py" file is open in the editor, containing Python code for a peer client. The code imports threading, socket, and time, defines a receive_messages function, and enters a loop where it receives data from a client socket. The run configuration "client" is selected, and the output window shows the program's execution:

```
C:\Users\HP\PycharmProjects\peer_to_peer_server\venv\bin\python.exe C:\Users\HP\PycharmProjects\peer_to_peer_server\client.py
Enter your first name: TALA
Enter your last name: JEBRINI
1- Send a message
2- Display a received message
Enter your choice:
```

Peer #2:

The screenshot shows the PyCharm IDE interface with a project named "peer_to_peer_server". The "client.py" file is open in the editor, containing Python code for a peer client. The code imports threading, socket, and time, defines a receive_messages function, and enters a loop where it receives data from a client socket. The run configuration "client" is selected, and the output window shows the program's execution:

```
C:\Users\HP\PycharmProjects\peer_to_peer_server\venv\bin\python.exe C:\Users\HP\PycharmProjects\peer_to_peer_server\client.py
Enter your first name: MALAK
Enter your last name: NASSAR
1- Send a message
2- Display a received message
Enter your choice: |
```

Message received from the first peer:

The screenshot shows a PyCharm project titled "peer_to_peer_server" with three files: main.py, client.py, and cl.py. The main.py file contains code for a client socket. The client.py file is currently selected and running, displaying a terminal window with the following interaction:

```
C:\Users\HP\PycharmProjects\peer_to_peer_server\venv\bin\python.exe C:\Users\HP\PycharmProjects\peer_to_peer_server\client.py
Enter your first name: TALA
Enter your last name: JEBRINI
1- Send a message
2- Display a received message
Enter your choice: Received a message from MALAK NASSAR at 23:41:18
```

The taskbar at the bottom shows various icons for system functions like search, file explorer, and browser.

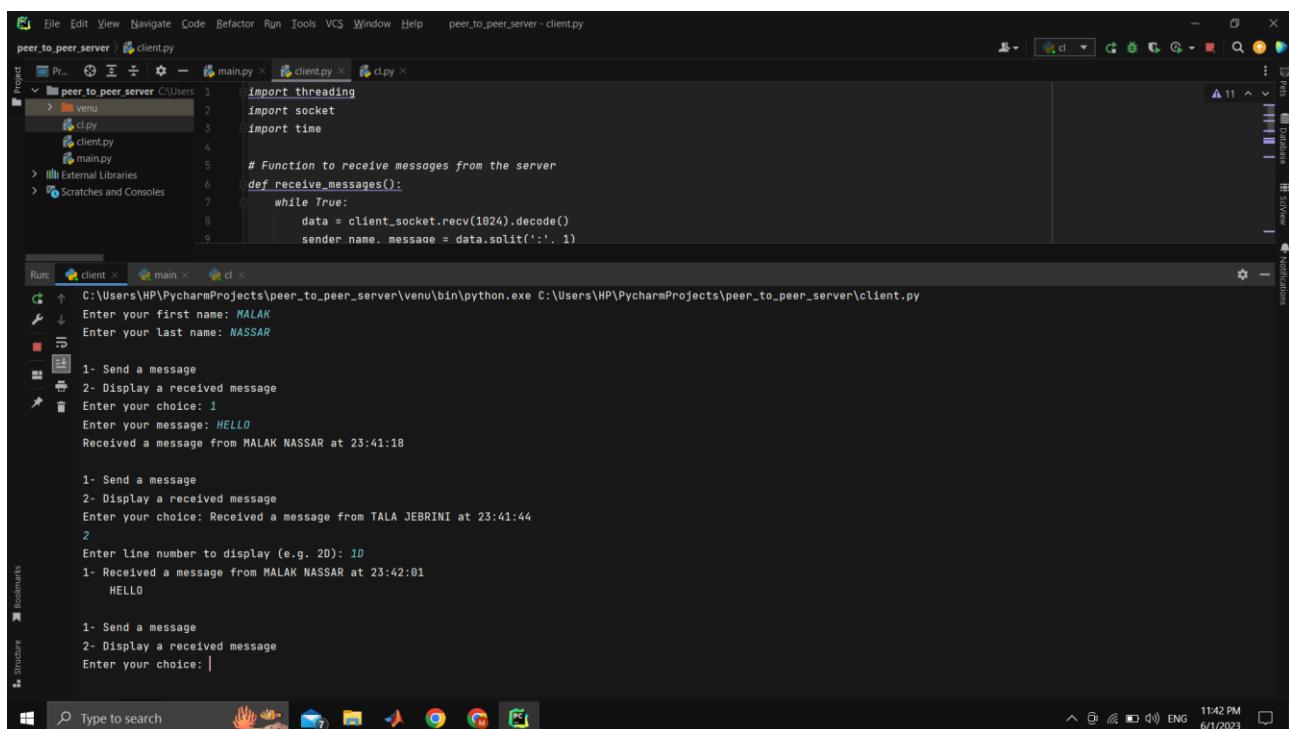
Message received from the second peer:

The screenshot shows the same PyCharm project and setup as the previous one. The client.py file is running, and the terminal window shows:

```
C:\Users\HP\PycharmProjects\peer_to_peer_server\venv\bin\python.exe C:\Users\HP\PycharmProjects\peer_to_peer_server\client.py
Enter your first name: MALAK
Enter your last name: NASSAR
1- Send a message
2- Display a received message
Enter your choice: 1
Enter your message: HELLO
Received a message from MALAK NASSAR at 23:41:18

1- Send a message
2- Display a received message
Enter your choice: Received a message from TALA JEBRINI at 23:41:44
```

The displayed of the reserved messages:

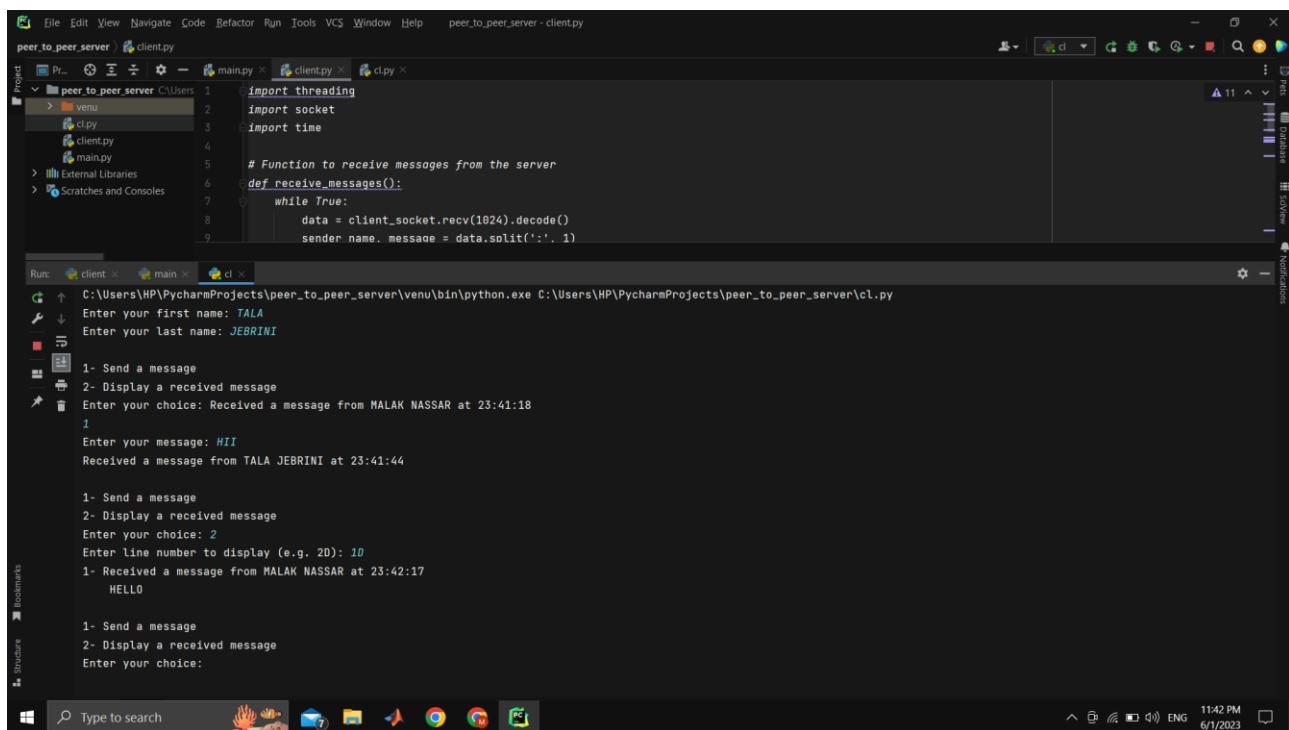


```
File Edit View Navigate Code Refactor Run Tools VCS Window Help peer_to_peer_server - client.py
peer_to_peer_server client.py
Project Py_... main.py clientpy cl.py
peer_to_peer_server C:\Users\venu
> venu
cl.py
client.py
main.py
> External Libraries
> Scratches and Consoles
import threading
import socket
import time
# Function to receive messages from the server
def receive_messages():
    while True:
        data = client_socket.recv(1024).decode()
        sender_name, message = data.split(':'. 1)
Run Run Client > main > cl
C:\Users\HP\PycharmProjects\peer_to_peer_server\venv\bin\python.exe C:\Users\HP\PycharmProjects\peer_to_peer_server\client.py
Enter your first name: MALAK
Enter your last name: NASSAR
1- Send a message
2- Display a received message
Enter your choice: 1
Enter your message: HELLO
Received a message from MALAK NASSAR at 23:41:18

1- Send a message
2- Display a received message
Enter your choice: Received a message from TALA JEBRINI at 23:41:44
2
Enter line number to display (e.g. 20): 10
1- Received a message from MALAK NASSAR at 23:42:01
    HELLO

1- Send a message
2- Display a received message
Enter your choice: |

```



```
File Edit View Navigate Code Refactor Run Tools VCS Window Help peer_to_peer_server - client.py
peer_to_peer_server client.py
Project Py_... main.py clientpy cl.py
peer_to_peer_server C:\Users\venu
> venu
cl.py
client.py
main.py
> External Libraries
> Scratches and Consoles
import threading
import socket
import time
# Function to receive messages from the server
def receive_messages():
    while True:
        data = client_socket.recv(1024).decode()
        sender_name, message = data.split(':'. 1)
Run Run Client > main > cl
C:\Users\HP\PycharmProjects\peer_to_peer_server\venv\bin\python.exe C:\Users\HP\PycharmProjects\peer_to_peer_server\cl.py
Enter your first name: TALA
Enter your last name: JEBRINI
1- Send a message
2- Display a received message
Enter your choice: Received a message from MALAK NASSAR at 23:41:18
1
Enter your message: HI
Received a message from TALA JEBRINI at 23:41:44

1- Send a message
2- Display a received message
Enter your choice: 2
Enter line number to display (e.g. 20): 10
1- Received a message from MALAK NASSAR at 23:42:17
    HELLO

1- Send a message
2- Display a received message
Enter your choice: |

```

The screenshot shows the PyCharm IDE interface with a project named 'peer_to_peer_server'. The 'main.py' file is open, displaying Python code for a client application. The code uses threading, socket, and time modules to receive messages from a server. A message from 'MALAK NASSAR' is received at 23:41:18. The user then sends a message to 'TALA JEBRINI' at 23:41:44. The application then receives a message from 'TALA JEBRINI' at 23:42:17, which is displayed as 'HELLO'. The user then sends another message to 'TALA JEBRINI' at 23:42:33, which is displayed as 'HII'. The application continues to loop, receiving and sending messages.

```
import threading
import socket
import time

# Function to receive messages from the server
def receive_messages():
    while True:
        data = client_socket.recv(1024).decode()
        sender_name, message = data.split(': ', 1)
```

Cilent and server code:

```
import socket
import threading

# Create a TCP socket
server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Bind the socket to a specific address and port
server_address = ('localhost', 8855)
server_socket.bind(server_address)

# Listen for incoming connections
server_socket.listen(5)
print('Server listening on', server_address)

# List to store connected clients
connected_clients = []

# Function to handle a client connection
def handle_client(client_socket, client_address):
    print('New connection from', client_address)

    # Receive the client's first name and last name
    client_info = client_socket.recv(1024).decode()
    first_name, last_name = client_info.split()

    # Add the client to the list of connected clients
    connected_clients.append((client_socket, first_name, last_name))

    while True:
        try:
```

```

data = client_socket.recv(1024).decode()
if data:
    # Send the message to all connected clients
    for client in connected_clients:
        sender = f"{first_name} {last_name}"
        message = f"{sender}: {data}"
        client[0].send(message.encode())
except ConnectionResetError:
    # Handle client disconnection
    print('Client', client_address, 'disconnected')
    connected_clients.remove((client_socket, first_name, last_name))
    client_socket.close()
    break

# Accept client connections
while True:
    client_socket, client_address = server_socket.accept()
    client_thread = threading.Thread(target=handle_client, args=(client_socket,
client_address))
    client_thread.start()

```

```

import threading

import socket

import time


# Function to receive messages from the server
def receive_messages():

    while True:

        data = client_socket.recv(1024).decode()

        sender_name, message = data.split(':', 1)

        received_messages.append((sender_name, message))

        print(f'Received a message from {sender_name.strip()} at
{time.strftime('%H:%M:%S')}')

# Function to send a message to the server
def send_message(message):

    client_socket.send(message.encode())

```

```

# Function to display the content of a received message

def display_message(line_number):

    if line_number >= 1 and line_number <= len(received_messages):

            sender_name, message = received_messages[line_number - 1]

            print(f"{line_number}- Received a message from {sender_name.strip()} at
{time.strftime('%H:%M:%S')})"

            print(f"    {message})")

    else:

            print("Invalid line number")

# Read peer's first name and last name

first_name = input("Enter your first name: ")

last_name = input("Enter your last name: ")

# Create a TCP socket

client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Connect to the server

server_address = ('localhost', 8855)

client_socket.connect(server_address)

client_socket.send(f"{first_name} {last_name}".encode())

# Start a thread to receive messages from the server

receive_thread = threading.Thread(target=receive_messages)

receive_thread.start()

# List to store received messages

received_messages = []

```

```
# Main loop to send and display messages

while True:

    print("\n1- Send a message")

    print("2- Display a received message")

    choice = input("Enter your choice: ")




    if choice == '1':

        message = input("Enter your message: ")

        send_message(message)

    elif choice == '2':

        line_input = input("Enter line number to display (e.g. 2D): ")

        if len(line_input) >= 2 and line_input[-1].upper() == 'D':



            line_number = int(line_input[:-1])

            display_message(line_number)

        else:

            print("Invalid input. Please enter a line number followed by 'D'.")

    else:

        print("Invalid choice")



    time.sleep(0.1)
```