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

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.

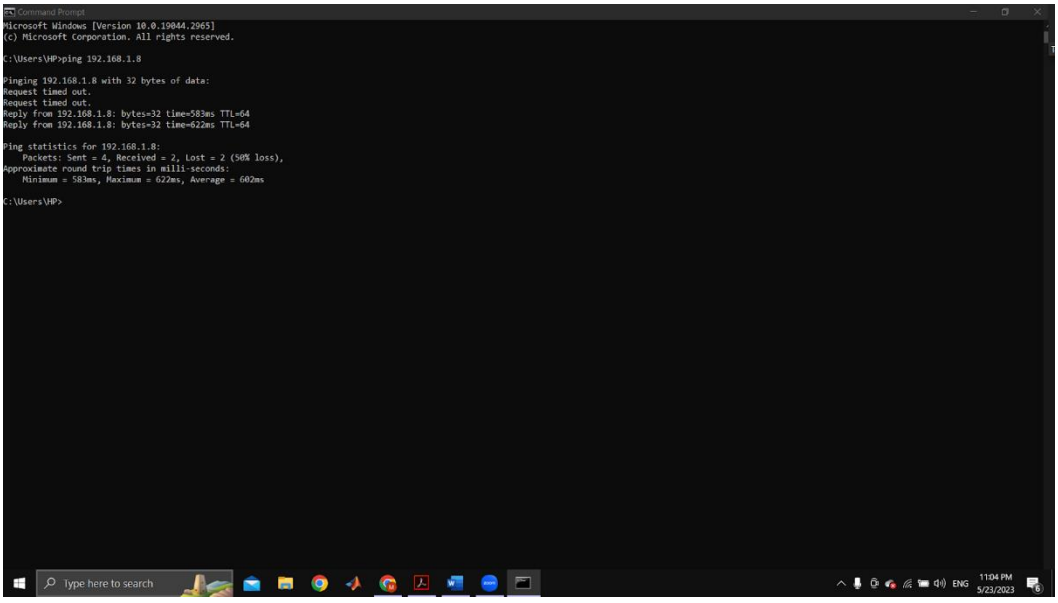


Figure 1:Ping a device in the same network

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 Min imum = 67ms, Maximum = 72ms, and Average = 68ms.We can see that the smartphone response ti me 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.

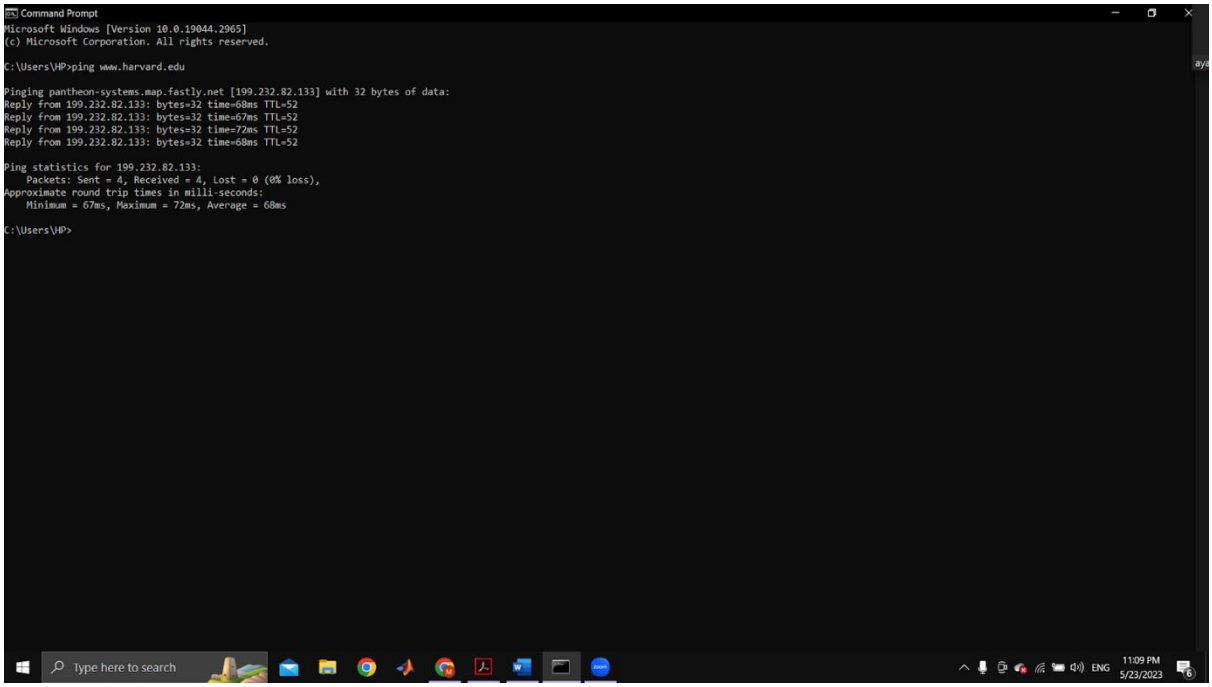


Figure 2:ping www.yale.edu

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 comment (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.

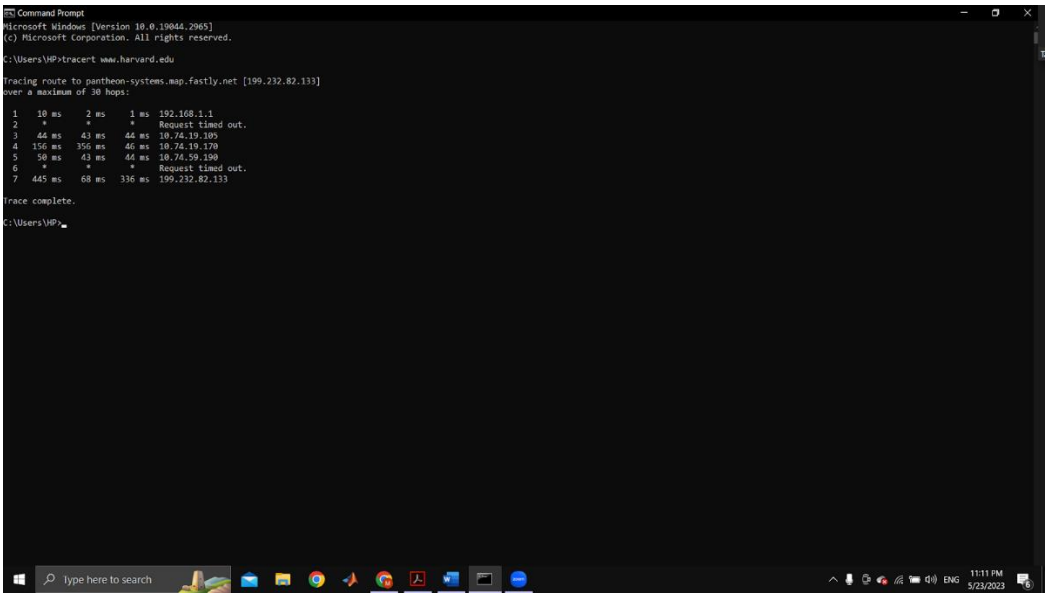


Figure 3:tracert www.harvard.edu

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 showen in figure 4:

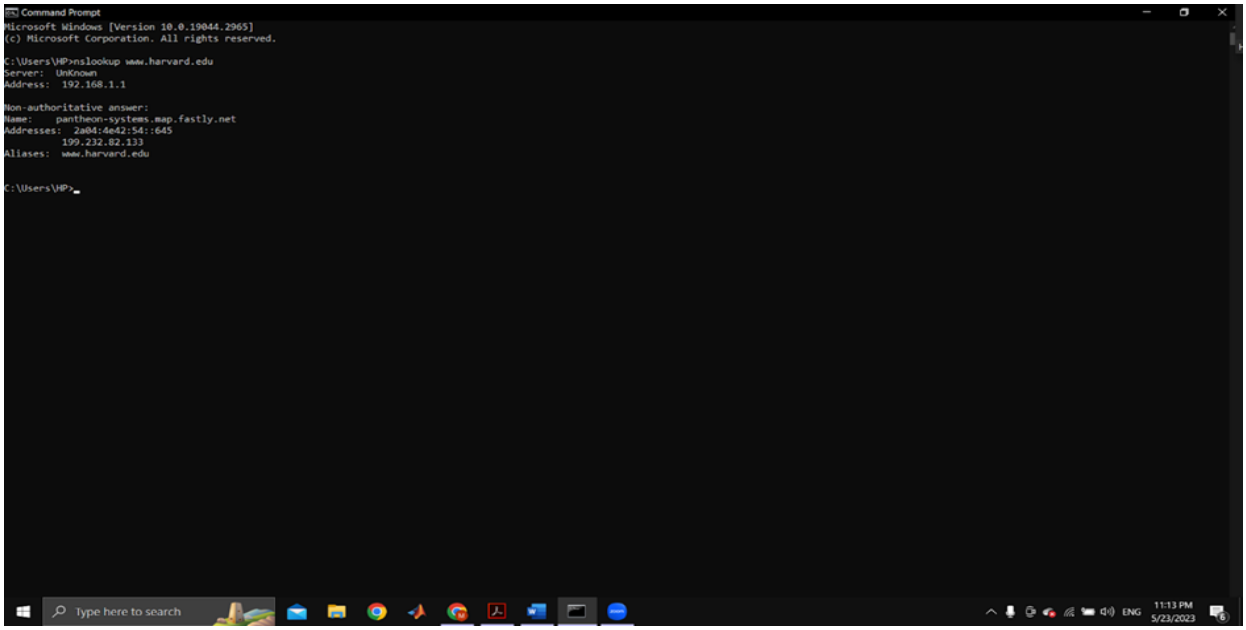


Figure 4:nslookup www.harvard.edu

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:

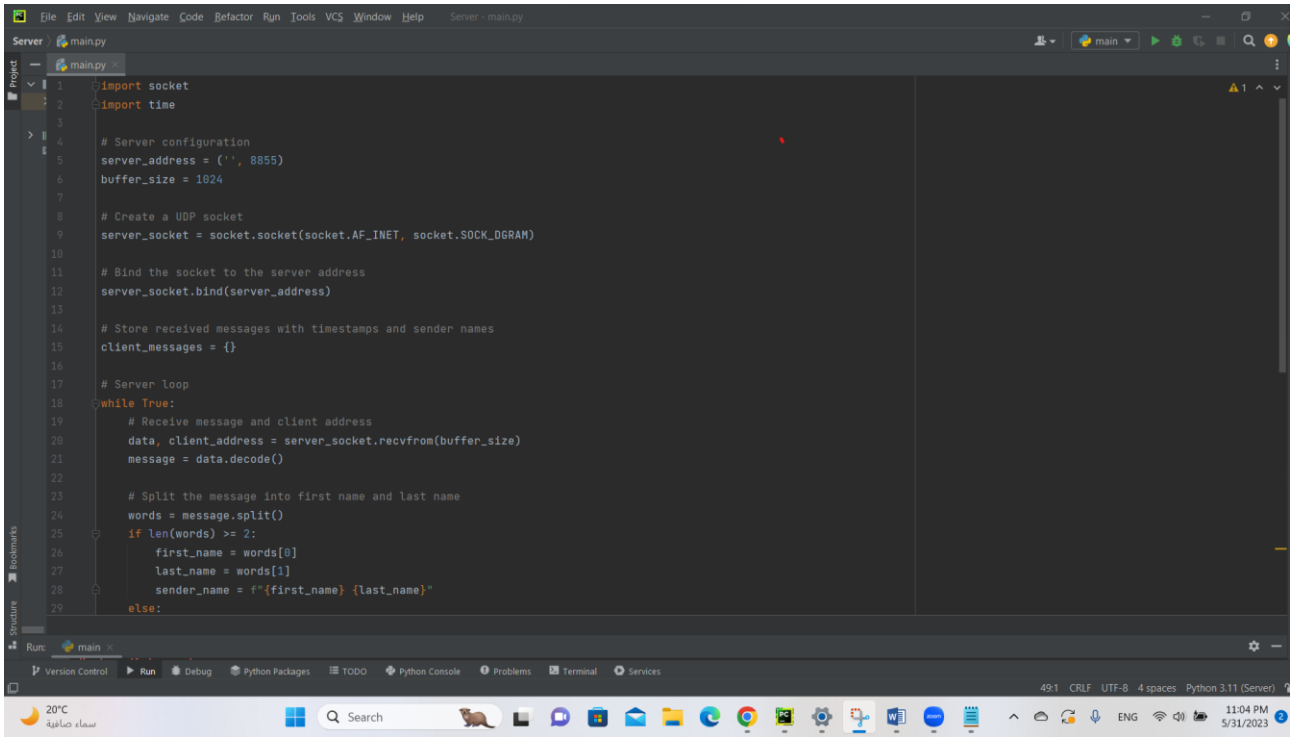


Figure 5: server code

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:

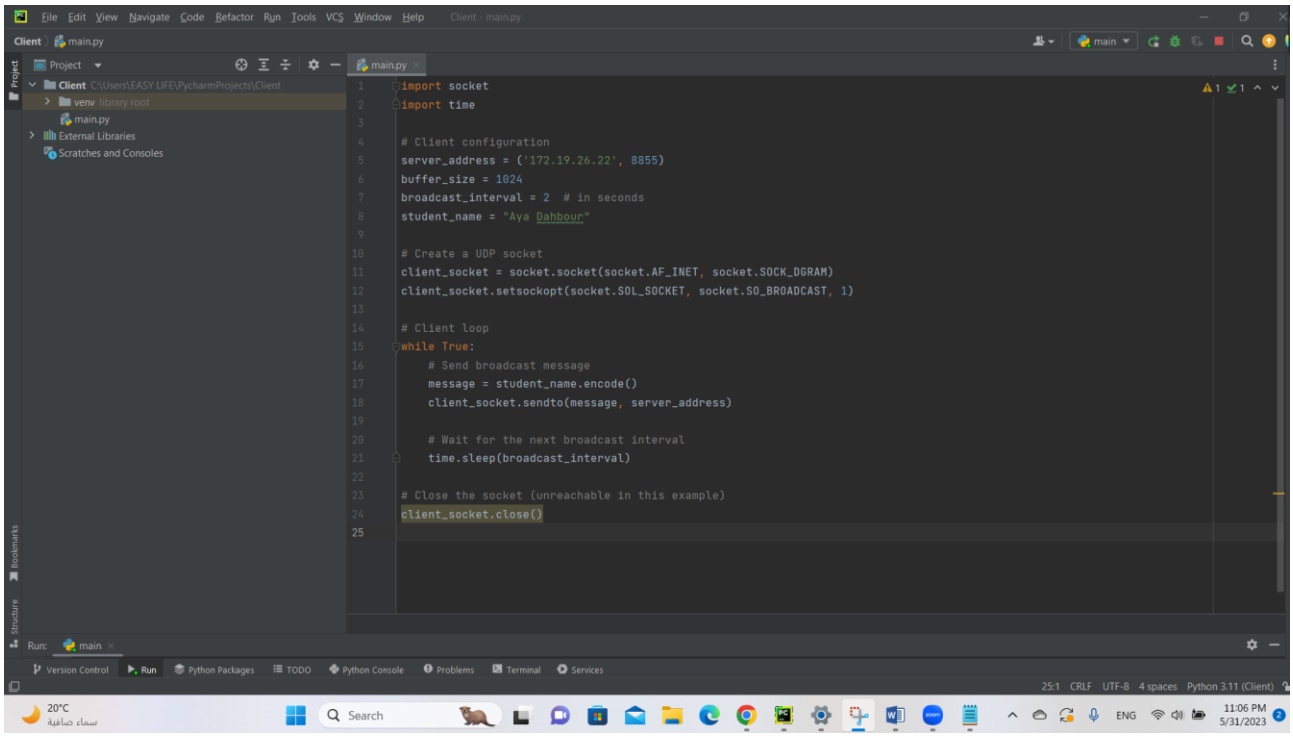


Figure 6:client Aya

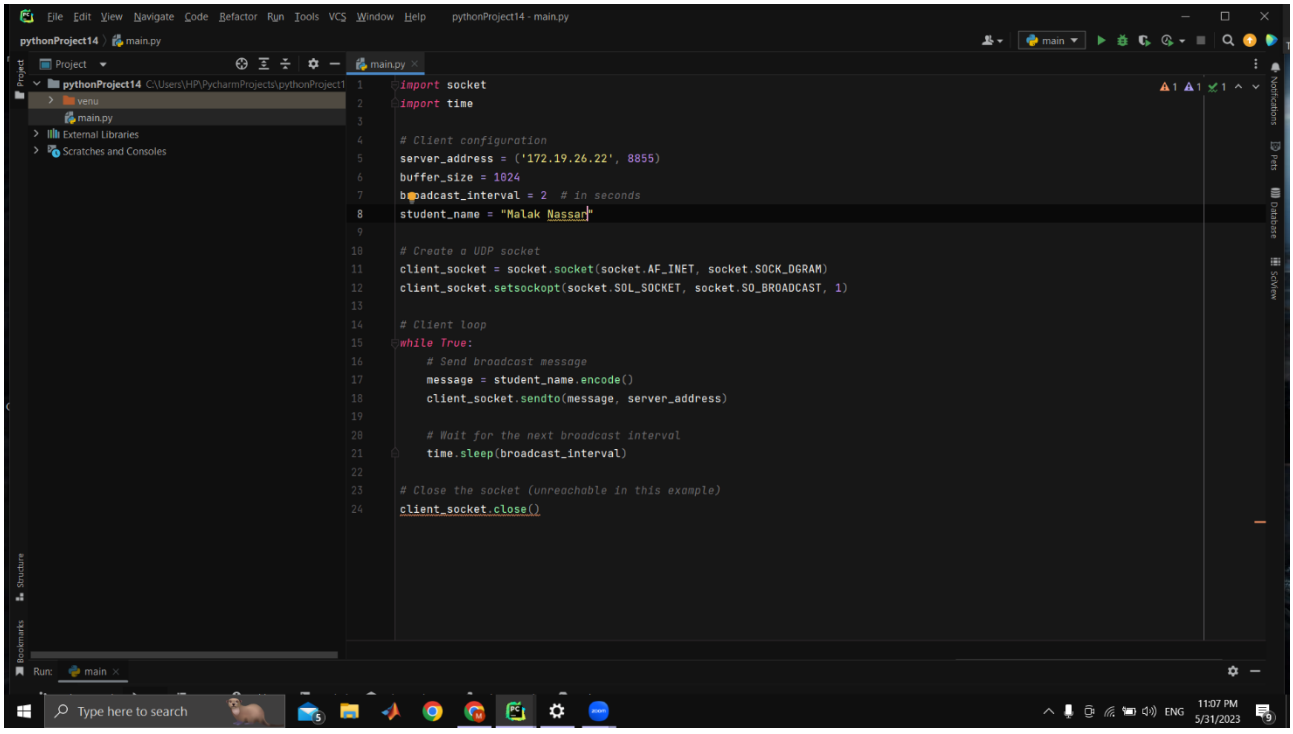
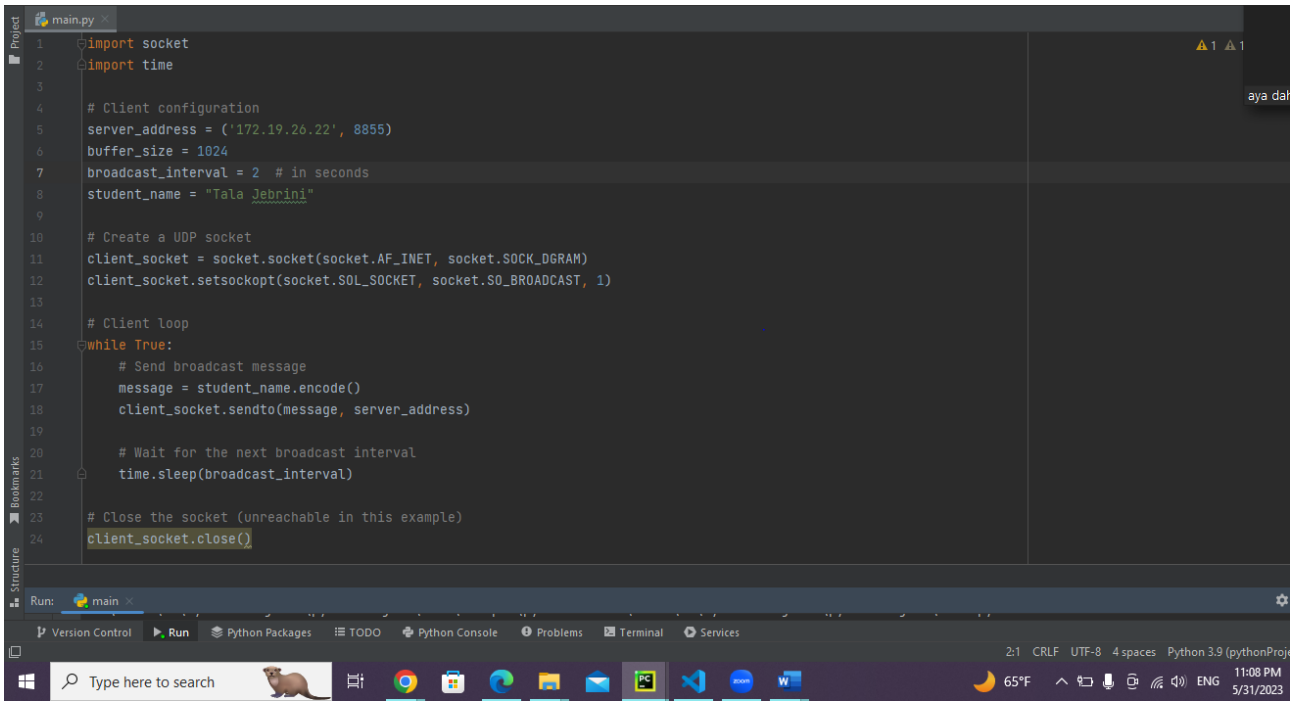
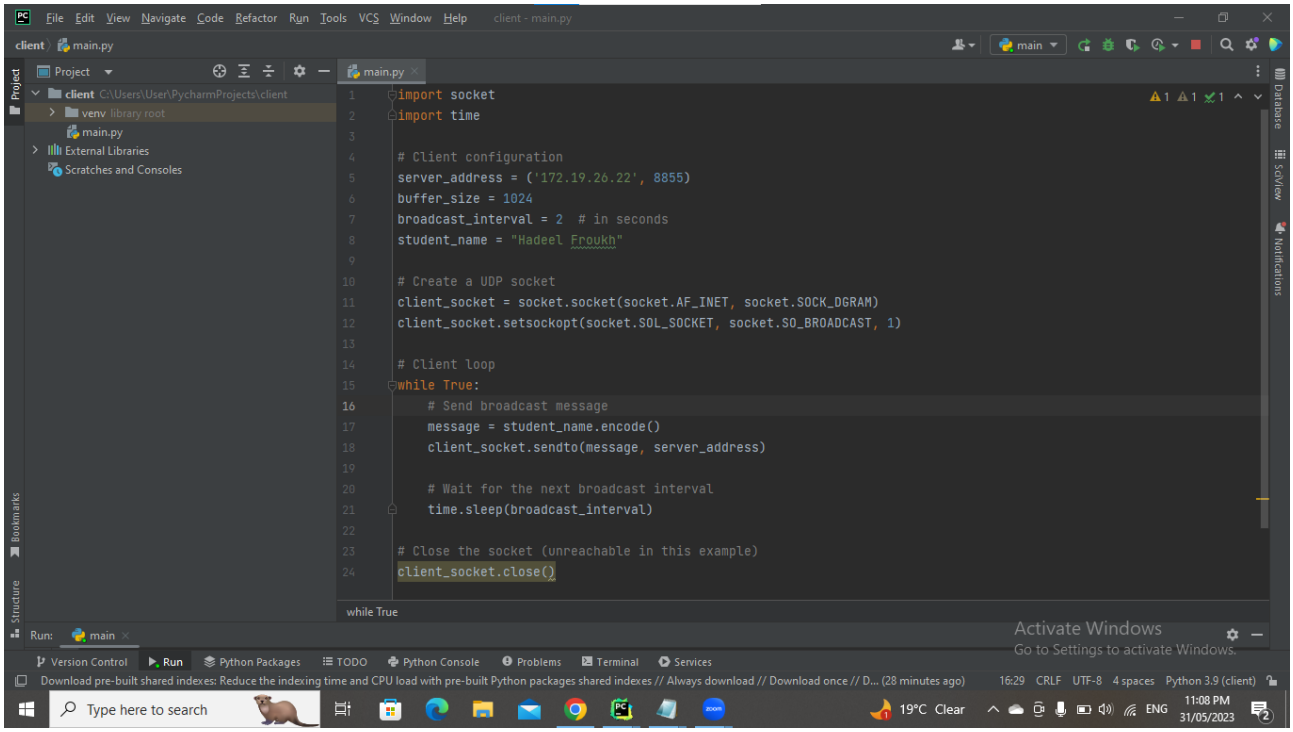


Figure 7:client Malak



```python
import socket
import time

# Client configuration
server_address = ('172.19.26.22', 8855)
buffer_size = 1024
broadcast_interval = 2  # in seconds
student_name = "Tala Jebrini"

# Create a UDP socket
client_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
client_socket.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)

# Client loop
while True:
    # Send broadcast message
    message = student_name.encode()
    client_socket.sendto(message, server_address)

    # Wait for the next broadcast interval
    time.sleep(broadcast_interval)

# Close the socket (unreachable in this example)
client_socket.close()
```

Figure 8:client Tala



```python
import socket
import time

# Client configuration
server_address = ('172.19.26.22', 8855)
buffer_size = 1024
broadcast_interval = 2  # in seconds
student_name = "Hadeel Froukh"

# Create a UDP socket
client_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
client_socket.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)

# Client loop
while True:
    # Send broadcast message
    message = student_name.encode()
    client_socket.sendto(message, server_address)

    # Wait for the next broadcast interval
    time.sleep(broadcast_interval)

# Close the socket (unreachable in this example)
client_socket.close()
```

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:

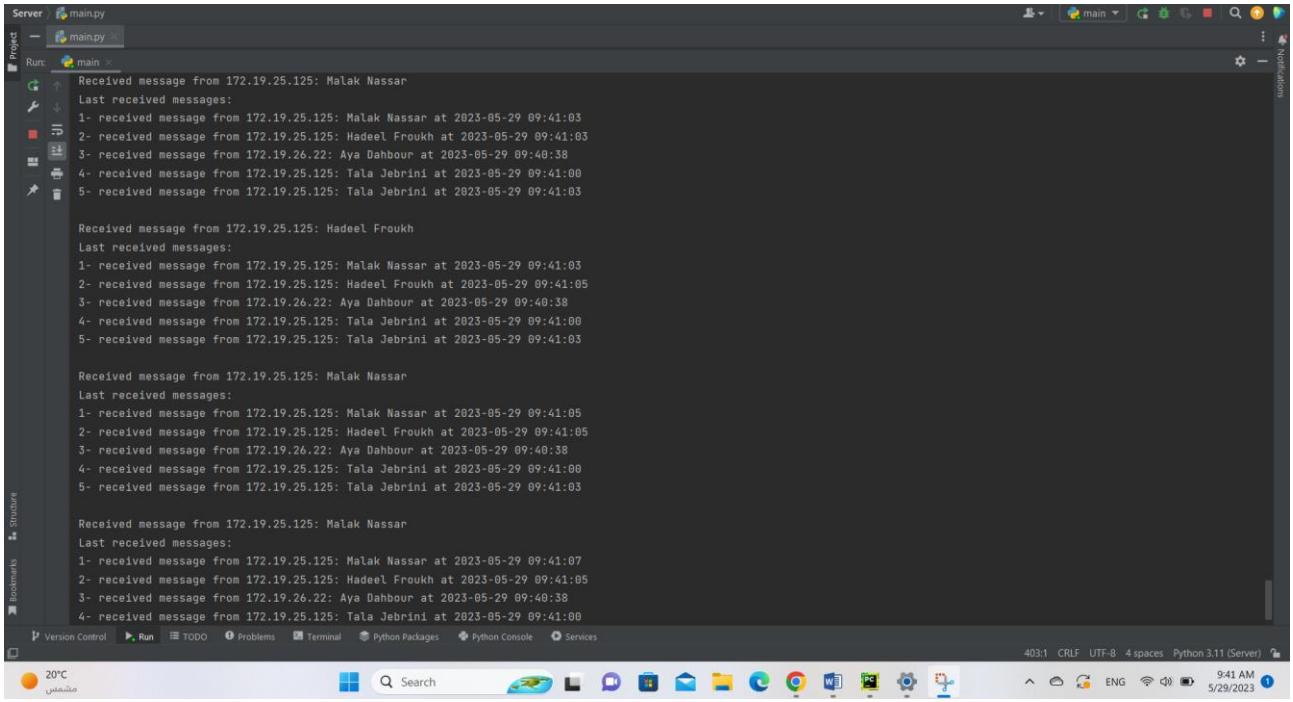


Figure 10:UDP client and server implementation output

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 '/':

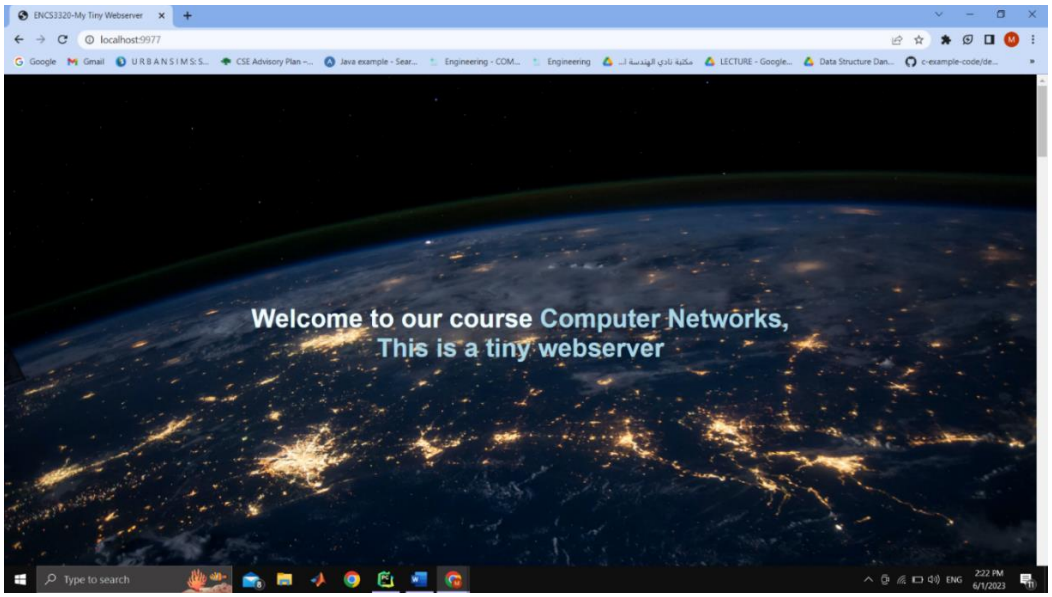


Figure 11:request '/'

The HTTP response:

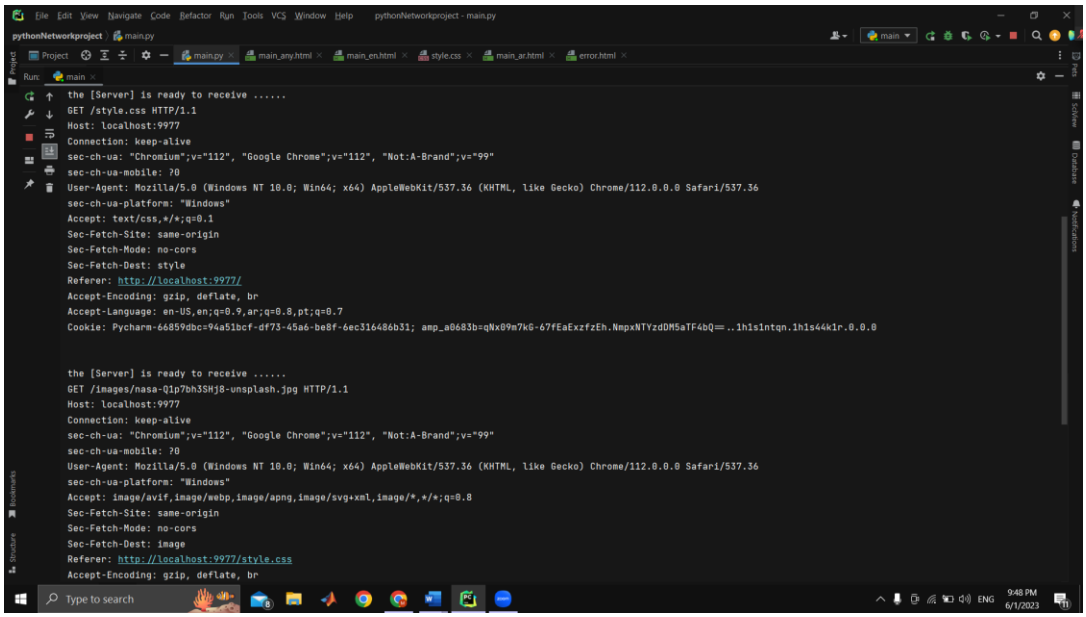


Figure 12:HTTP response for '/'

2.if request **‘/index.html’:**

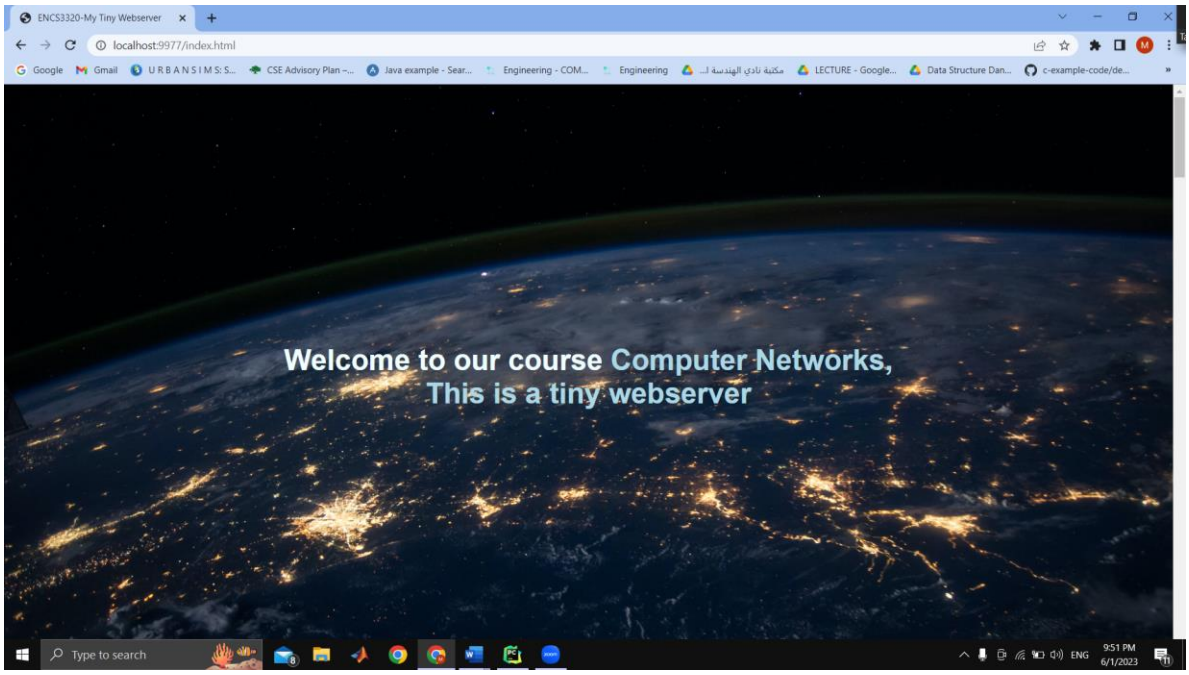


Figure 13:request index.html

The http response:

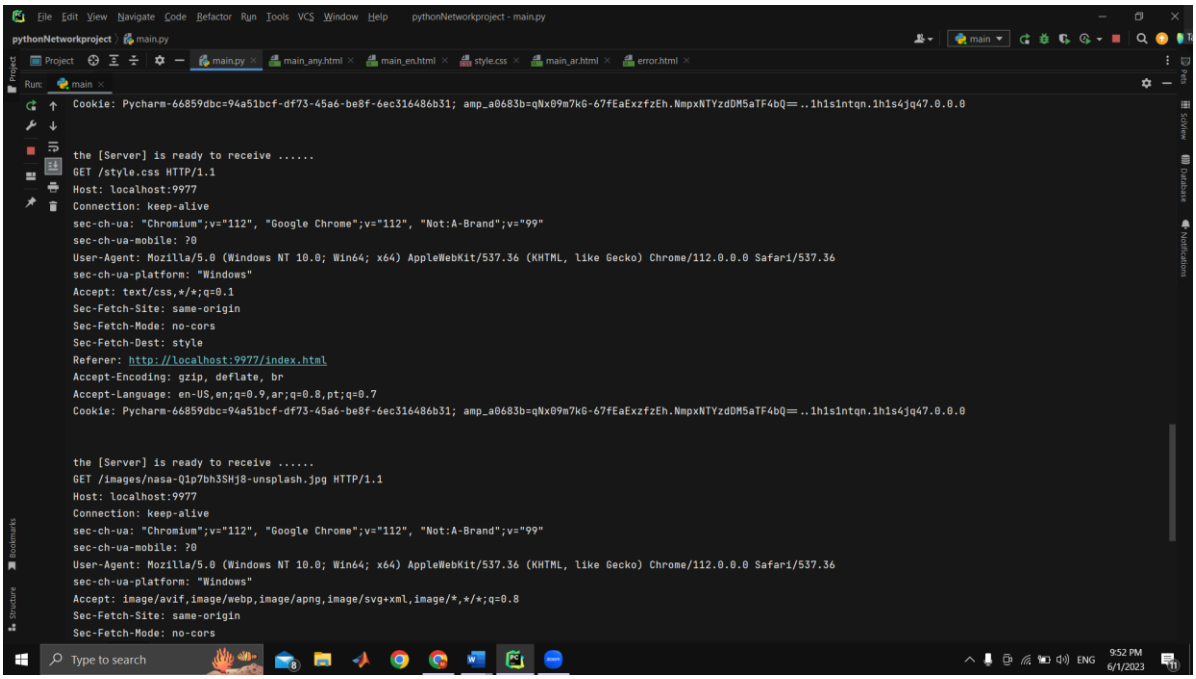


Figure 14:HTTP response for request index.html

4. if request ‘/main_en.html’:

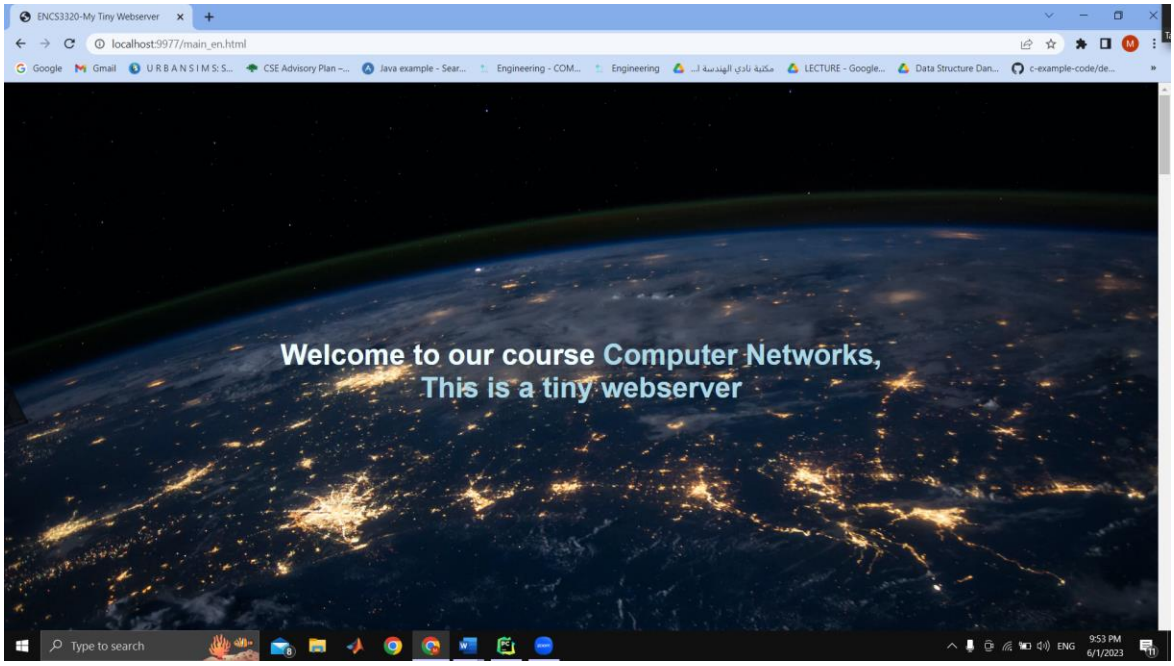


Figure 15:request ‘main_en.html’

The HTTP response:

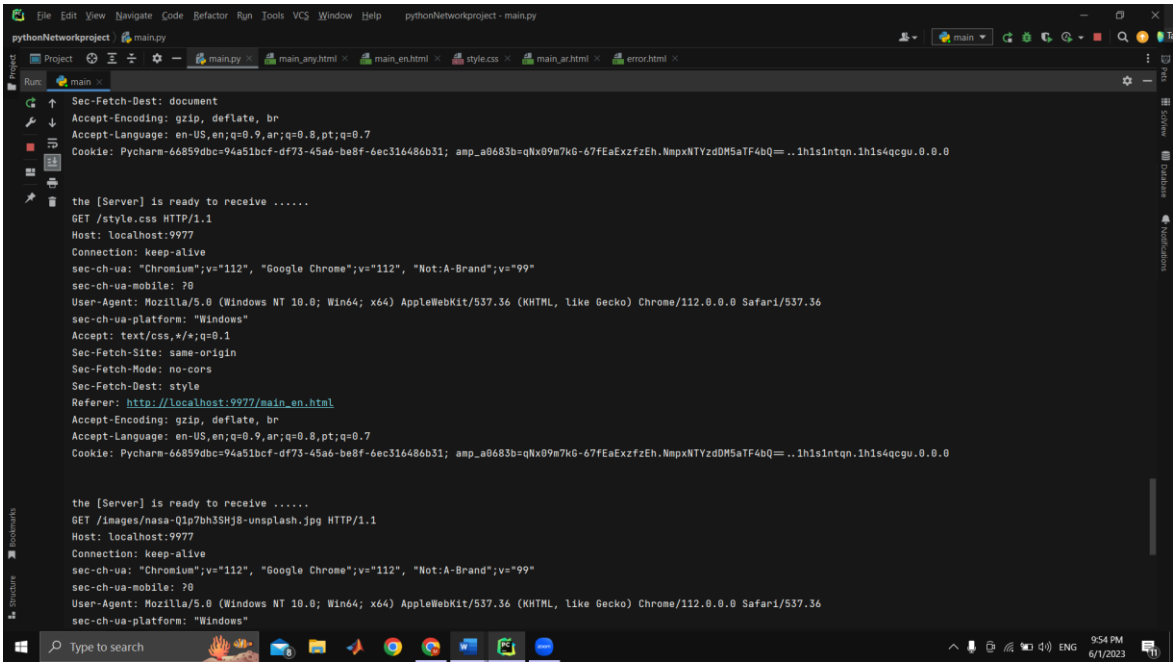


Figure 16:HTTP response request ‘/main_en.html’

4.if the request ‘/en’ :

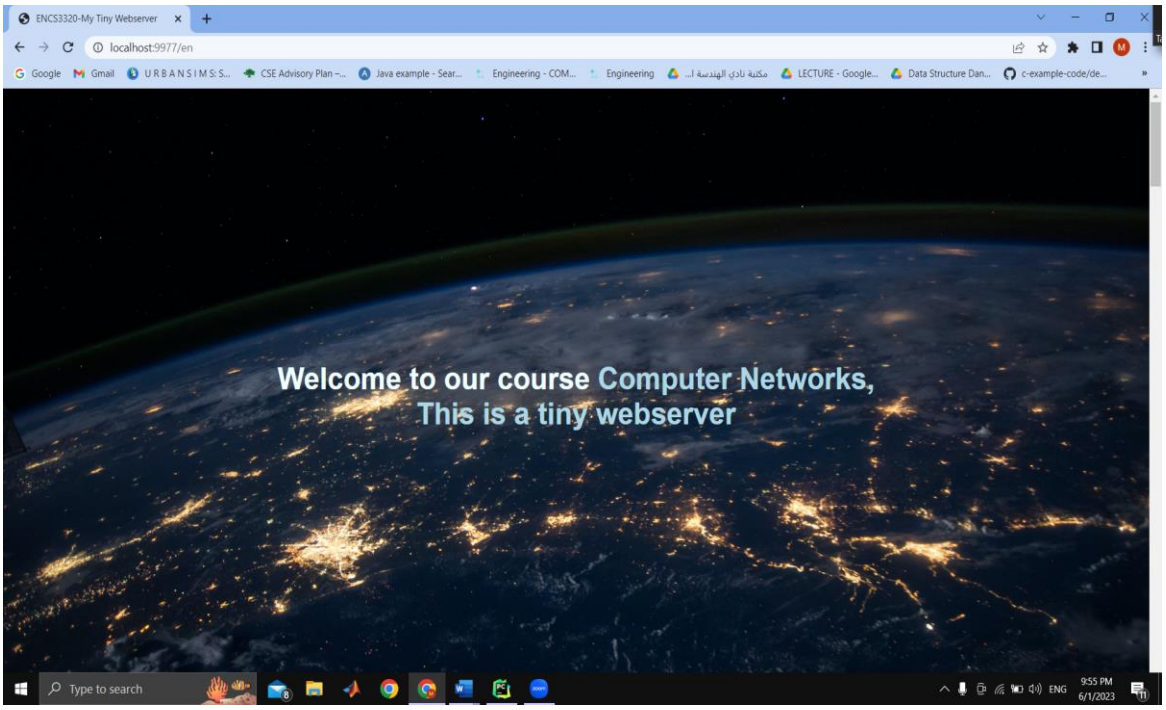


Figure 17:Figure 15:request 'en'

The HTTP response:

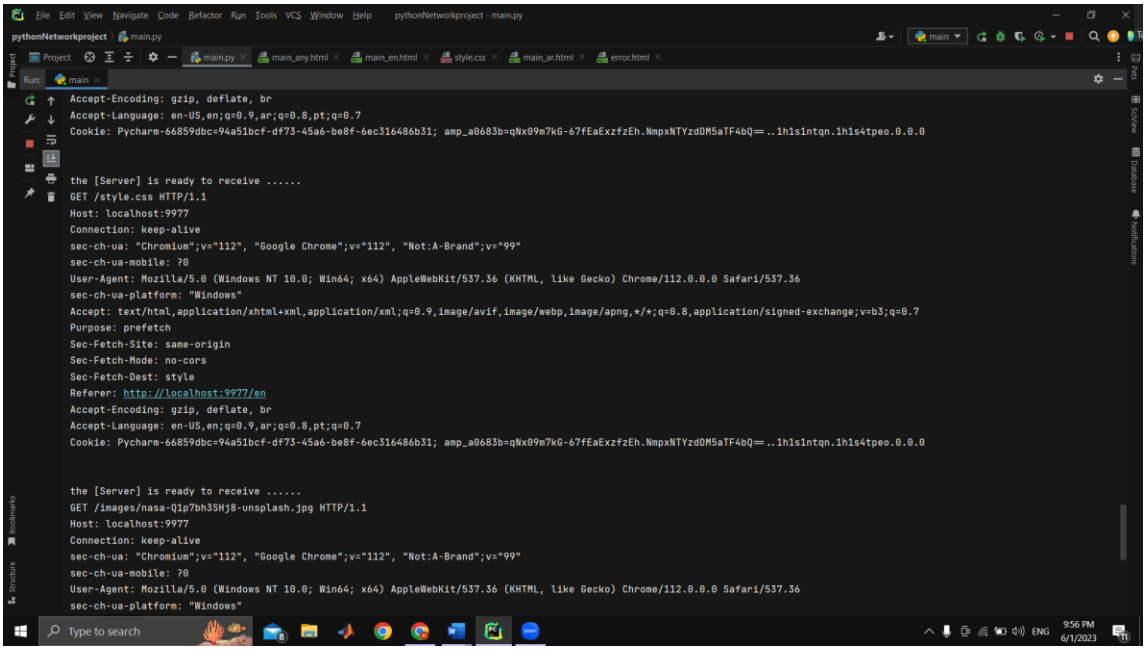


Figure 18:HTTP response request 'en'

3.2: when the request is /ar:

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

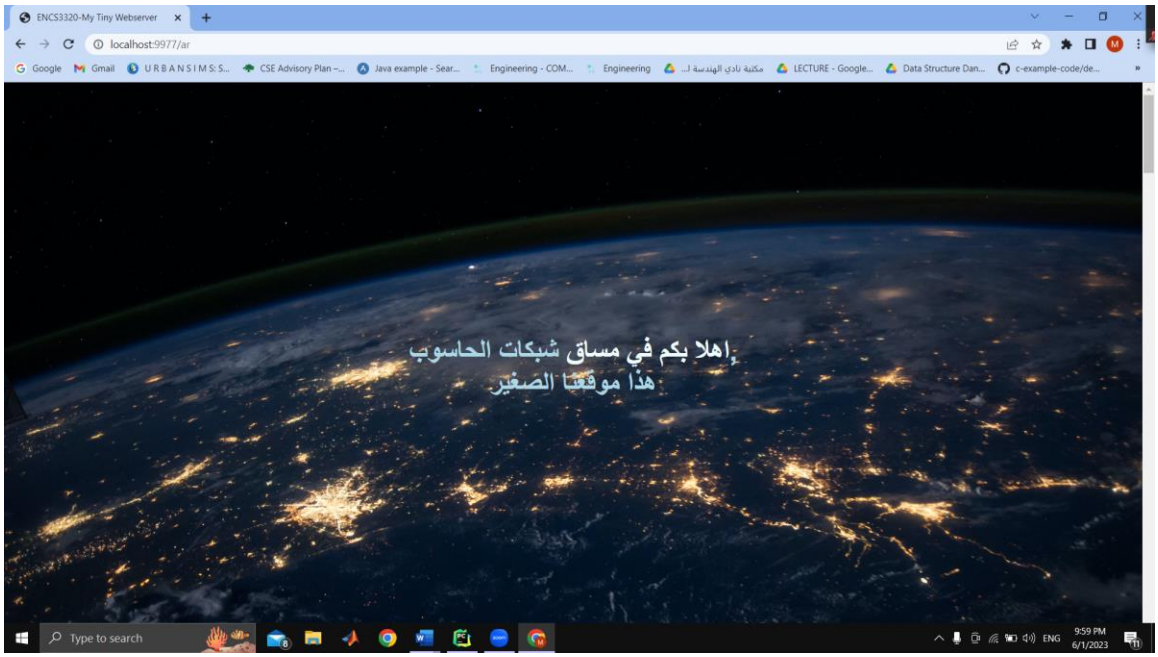


Figure 19:request 'ar'

The http response:

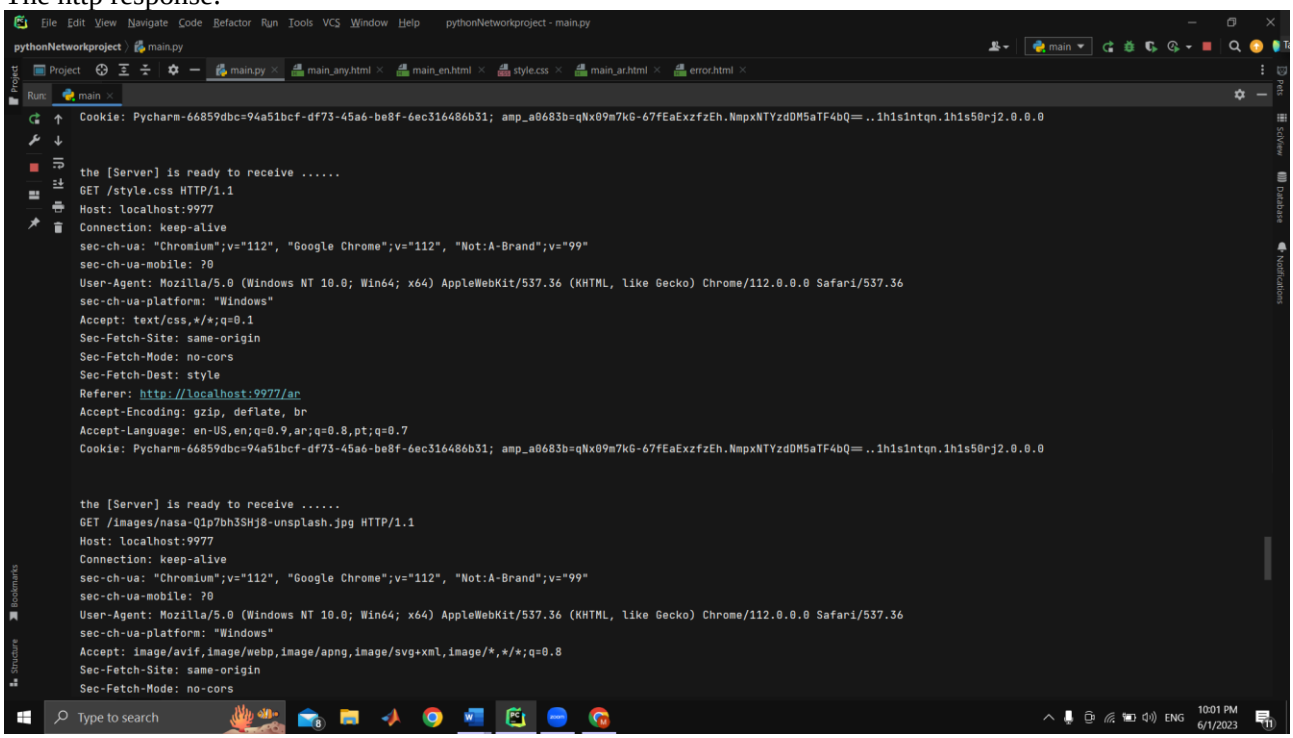


Figure 20:HTTP response request 'ar'

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.

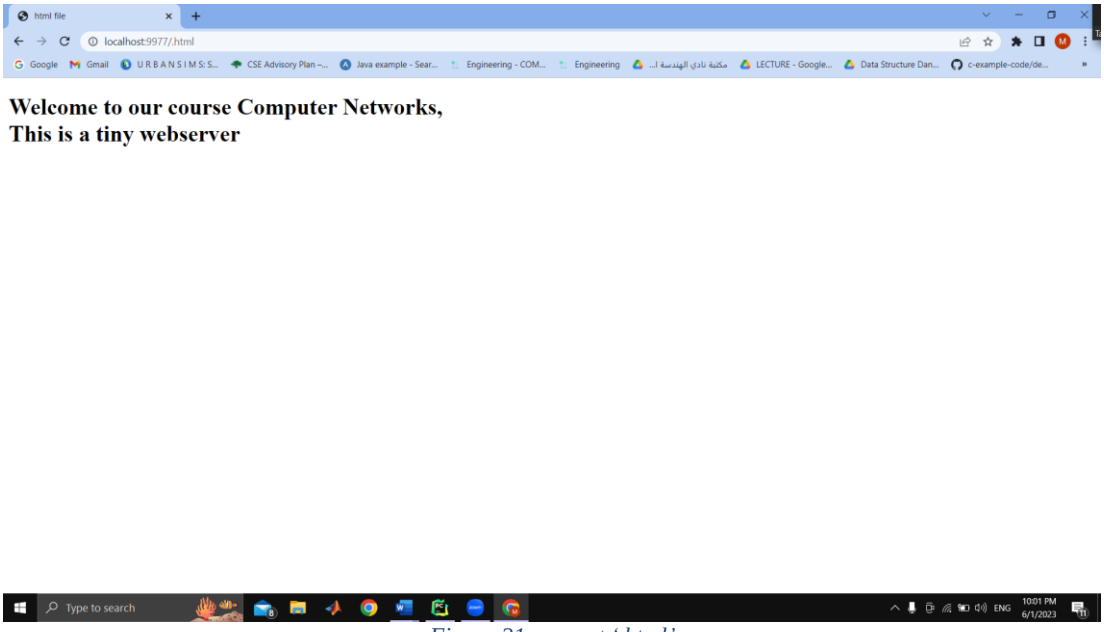


Figure 21:request '.html'

The http response:

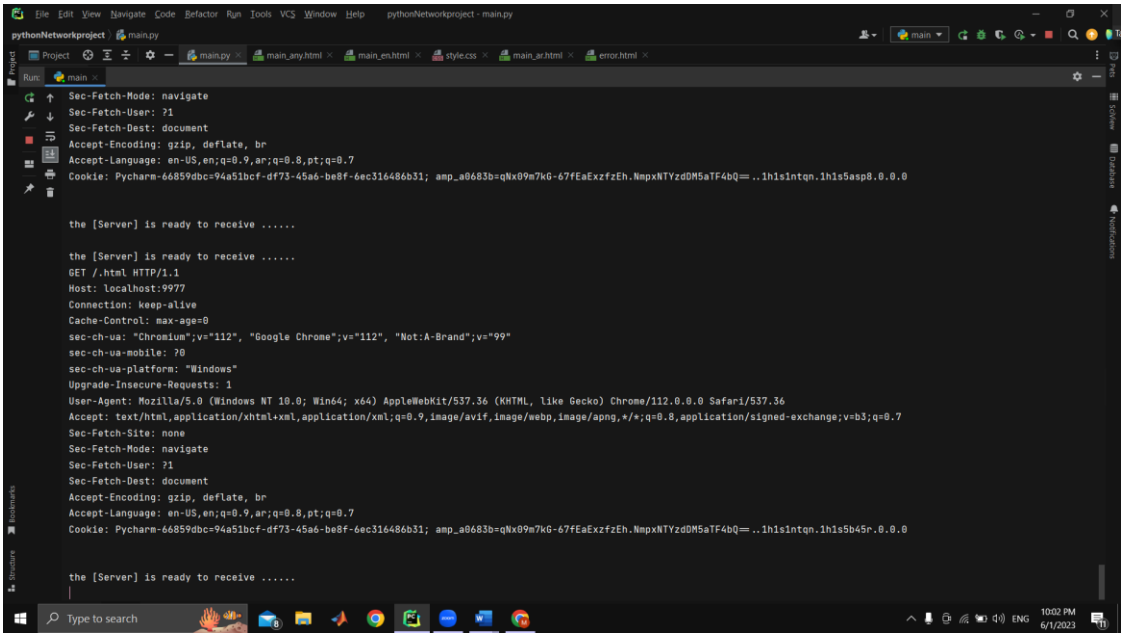


Figure 22:HTTP response request '.html'

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.

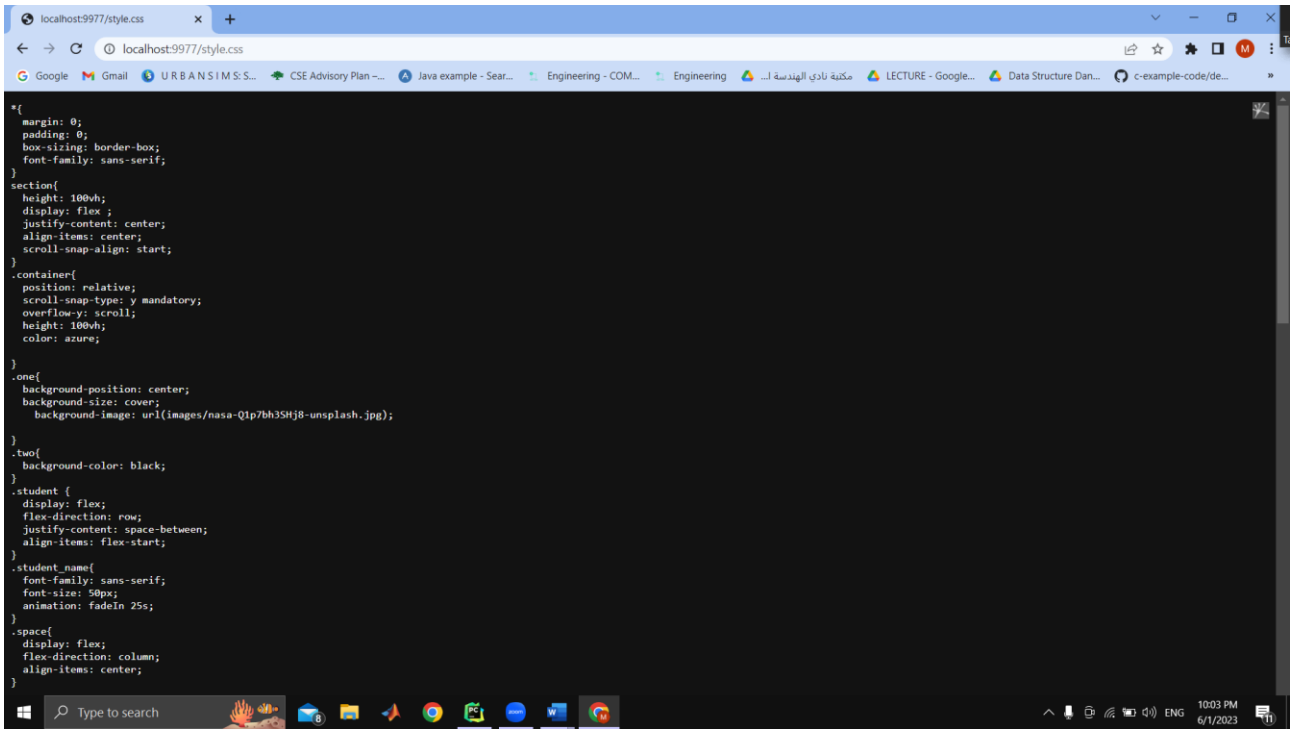


Figure 23::request '.css'

The http response:

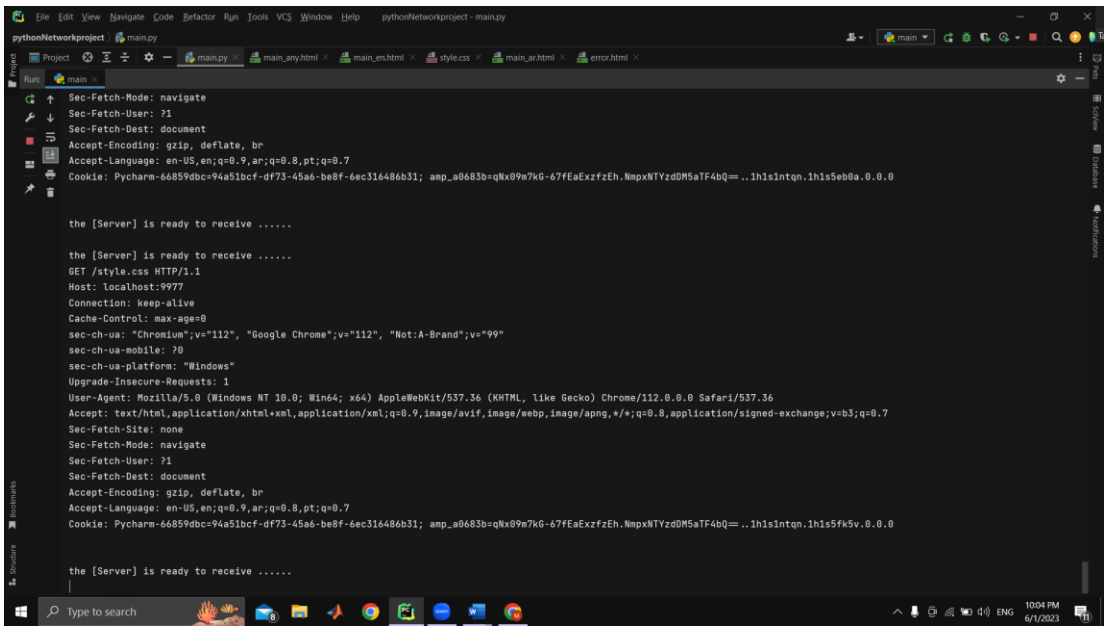


Figure 24:HTTP response request '.css'

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 imag.

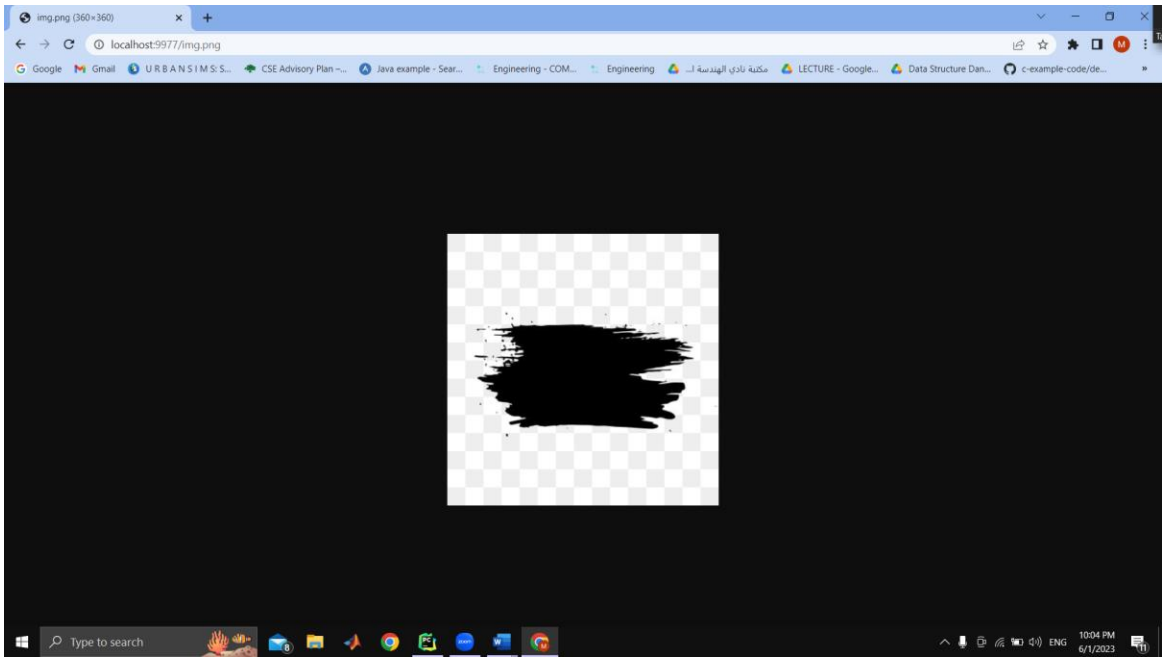


Figure 25:request '.png'

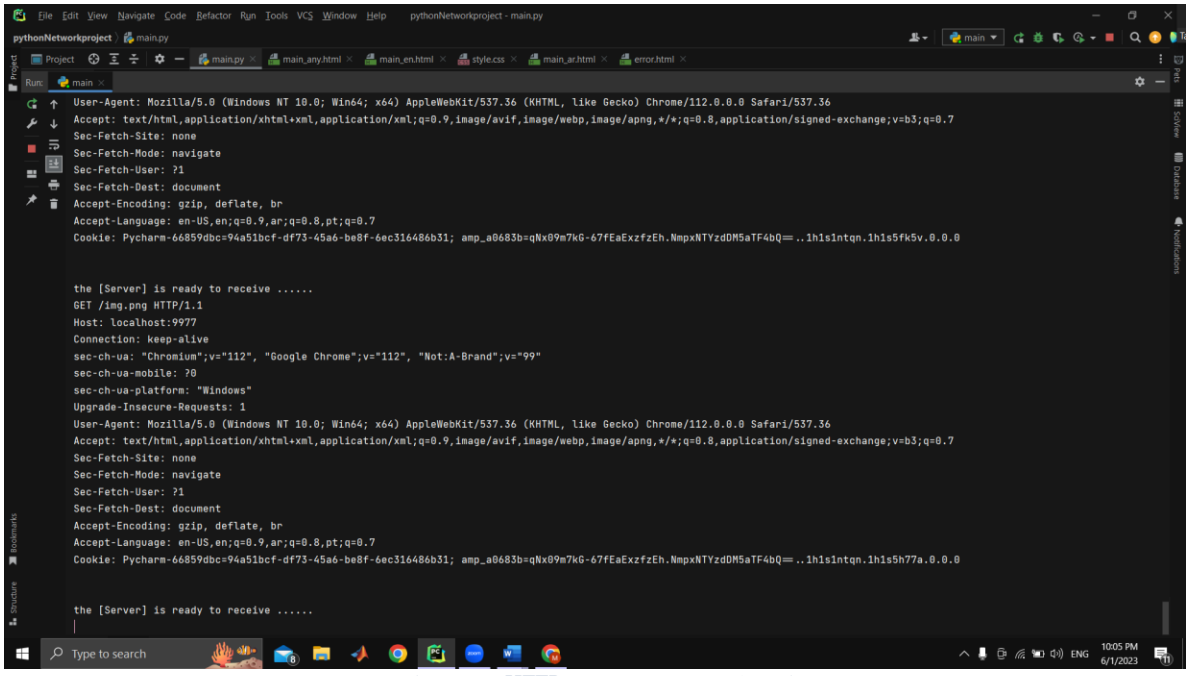


Figure 26:HTTP response request '.png'

Ends with jpg:

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

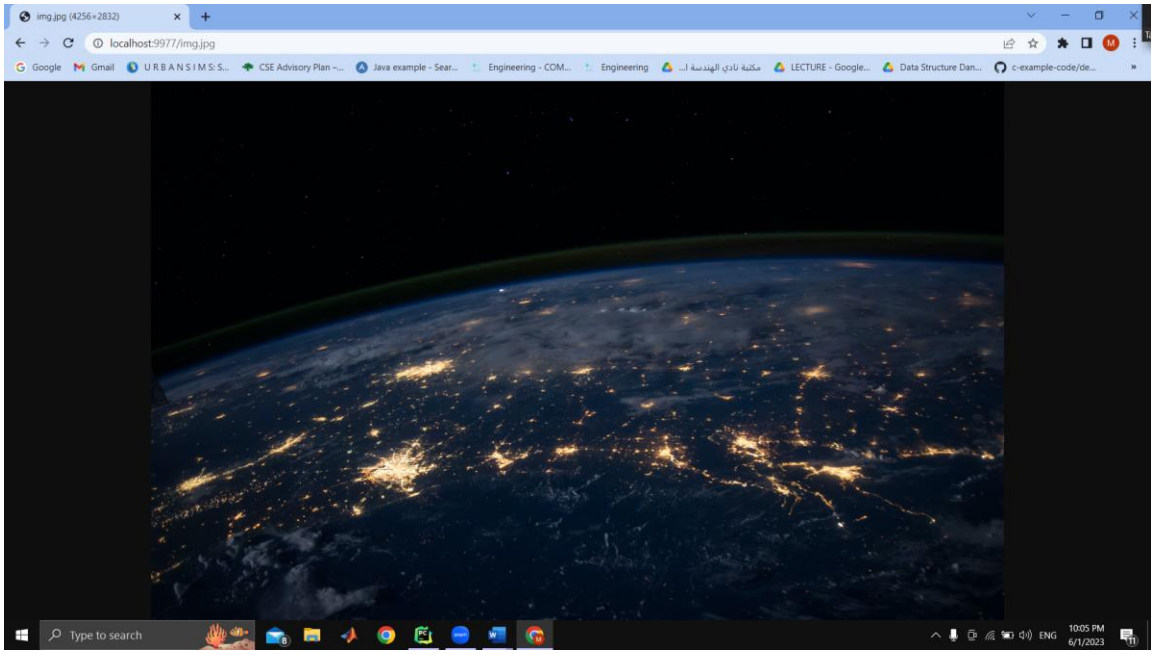


Figure 27:request '.jpg'

The HTTP response:

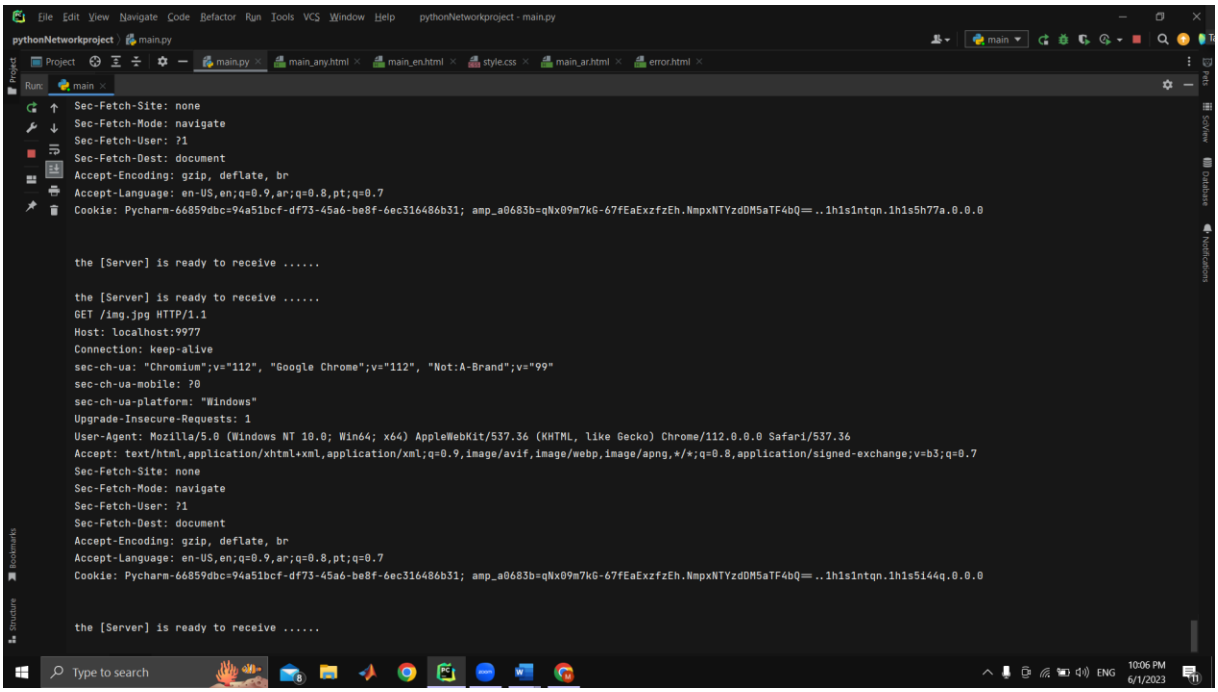


Figure 28:HTTP response request '.jpg'

Use the status code 307 Temporary Redirect to redirect the following

a. If the request is /yt then redirect to youtube website

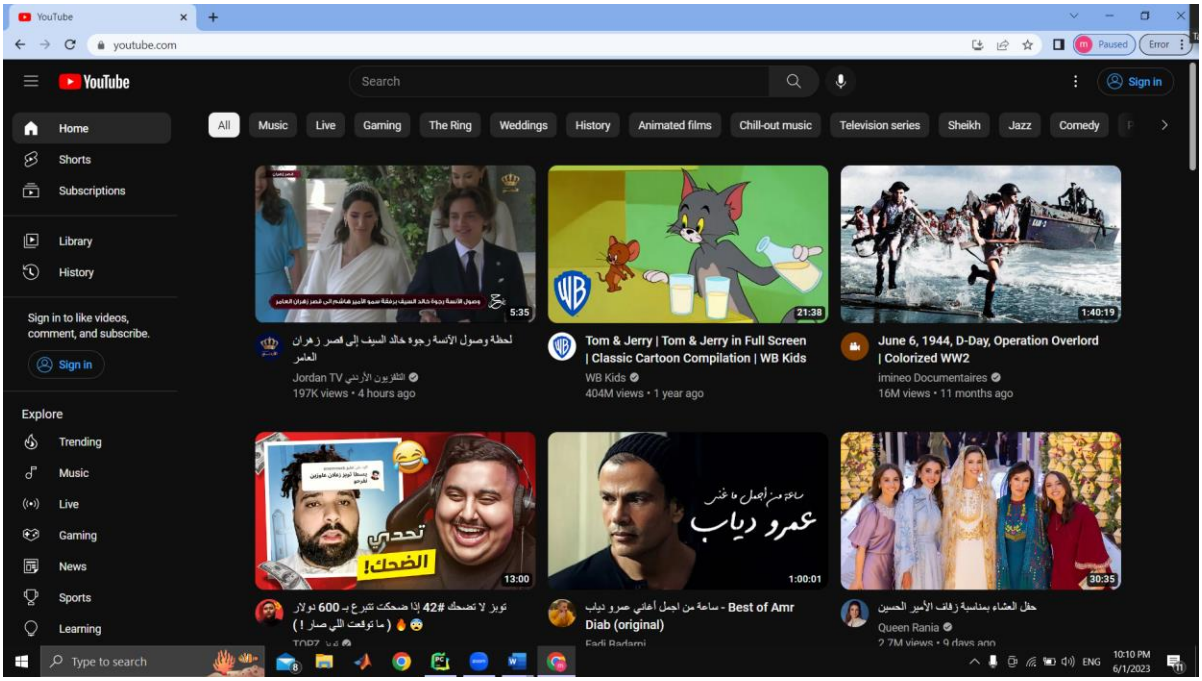


Figure 29:request 'yt'

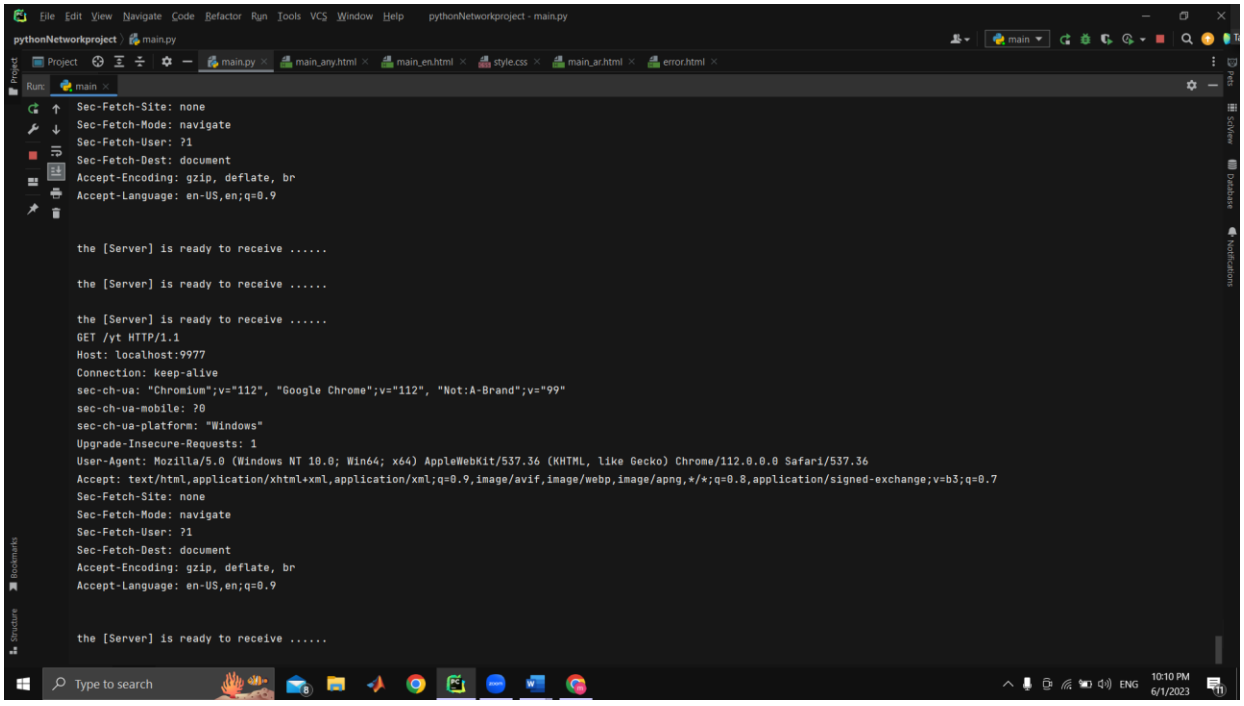
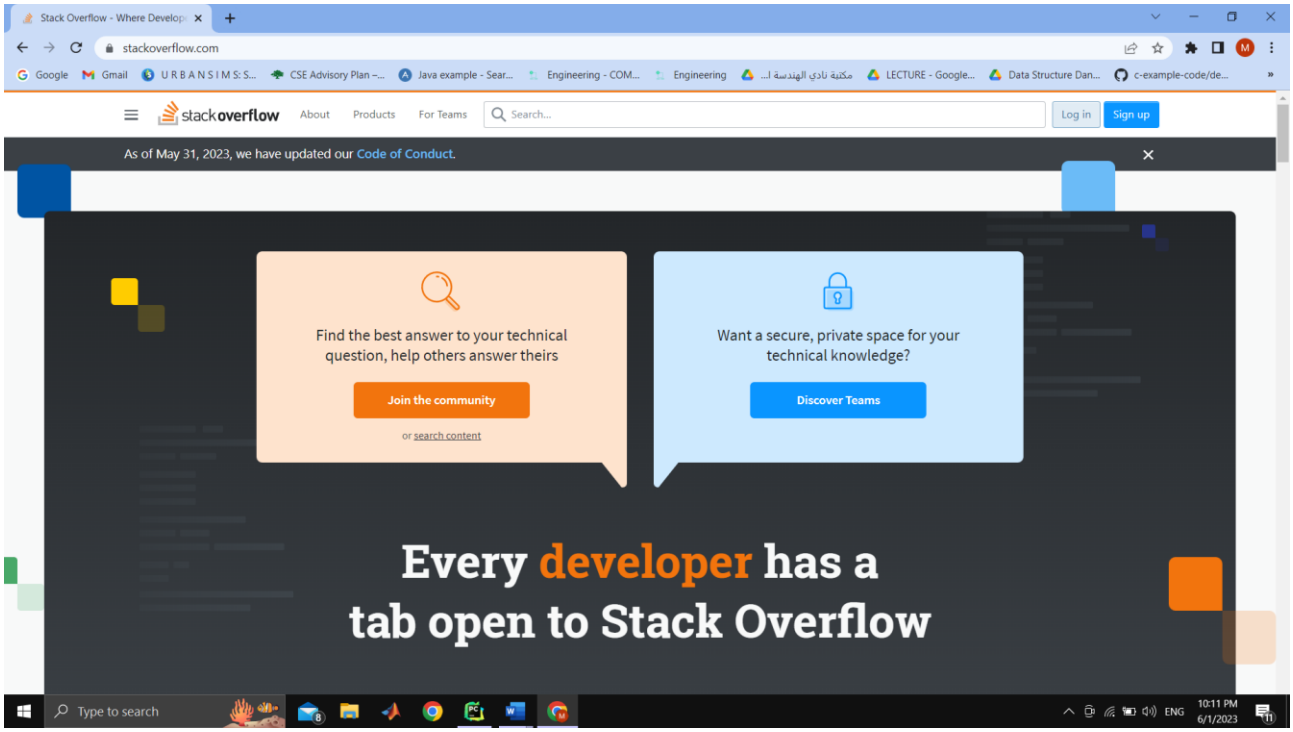


Figure 30:HTTP response request 'yt'

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



The http response:

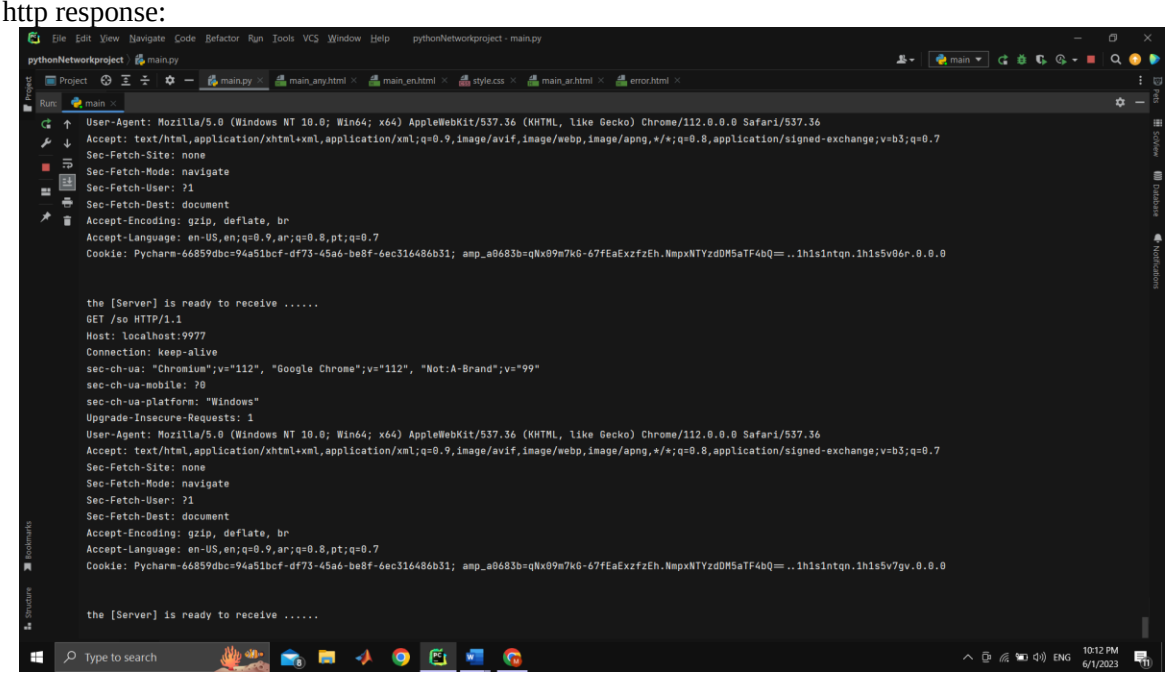
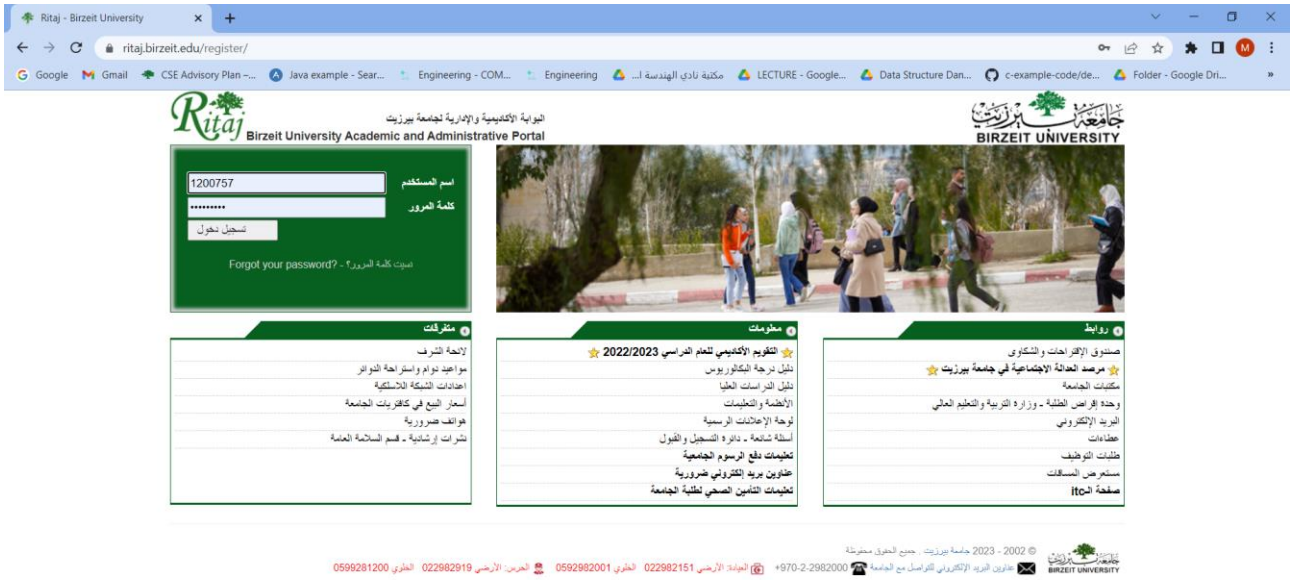
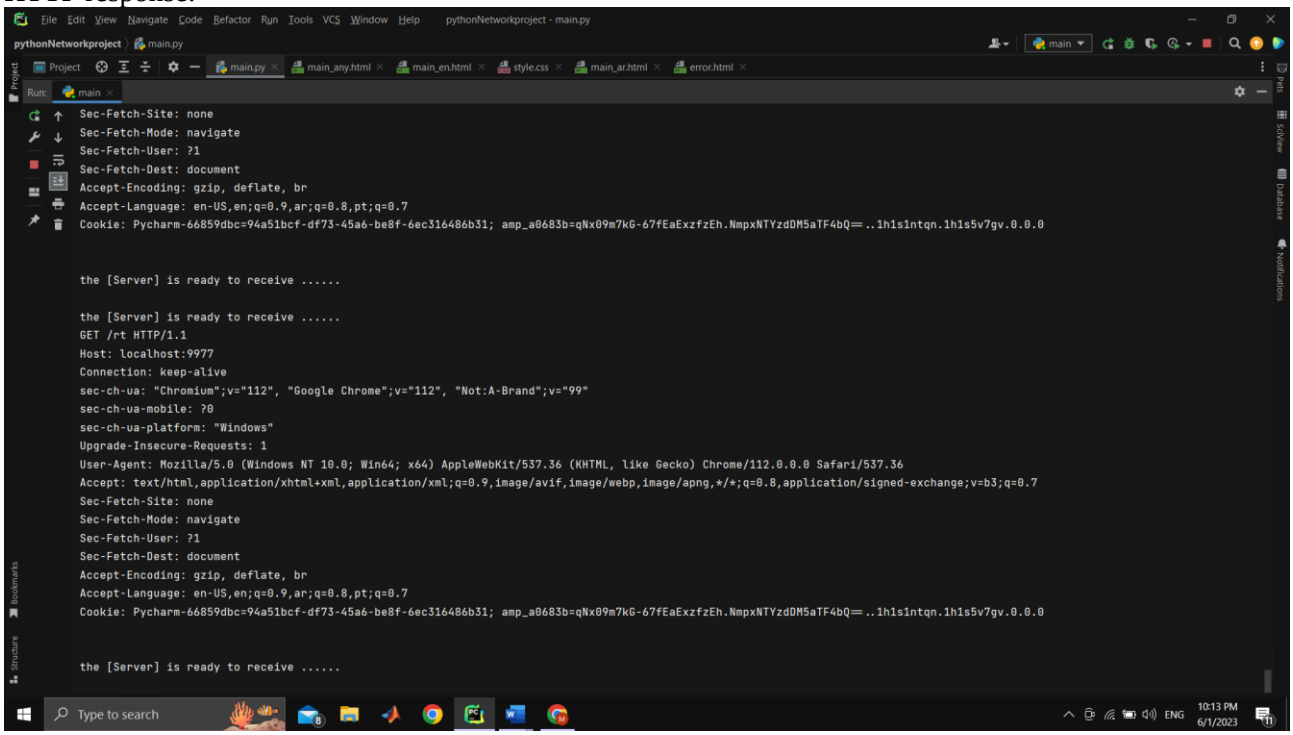


Figure 31:HTTP response request 'so'

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

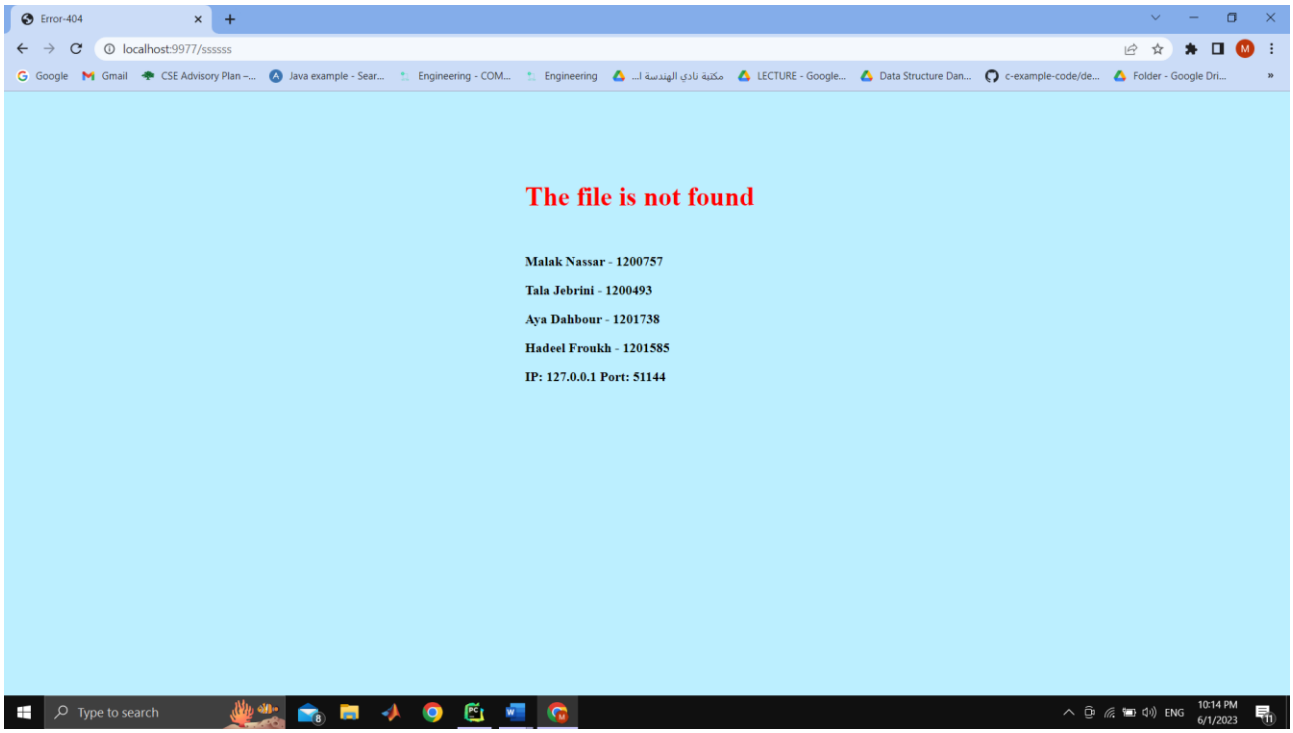


HTTP response:

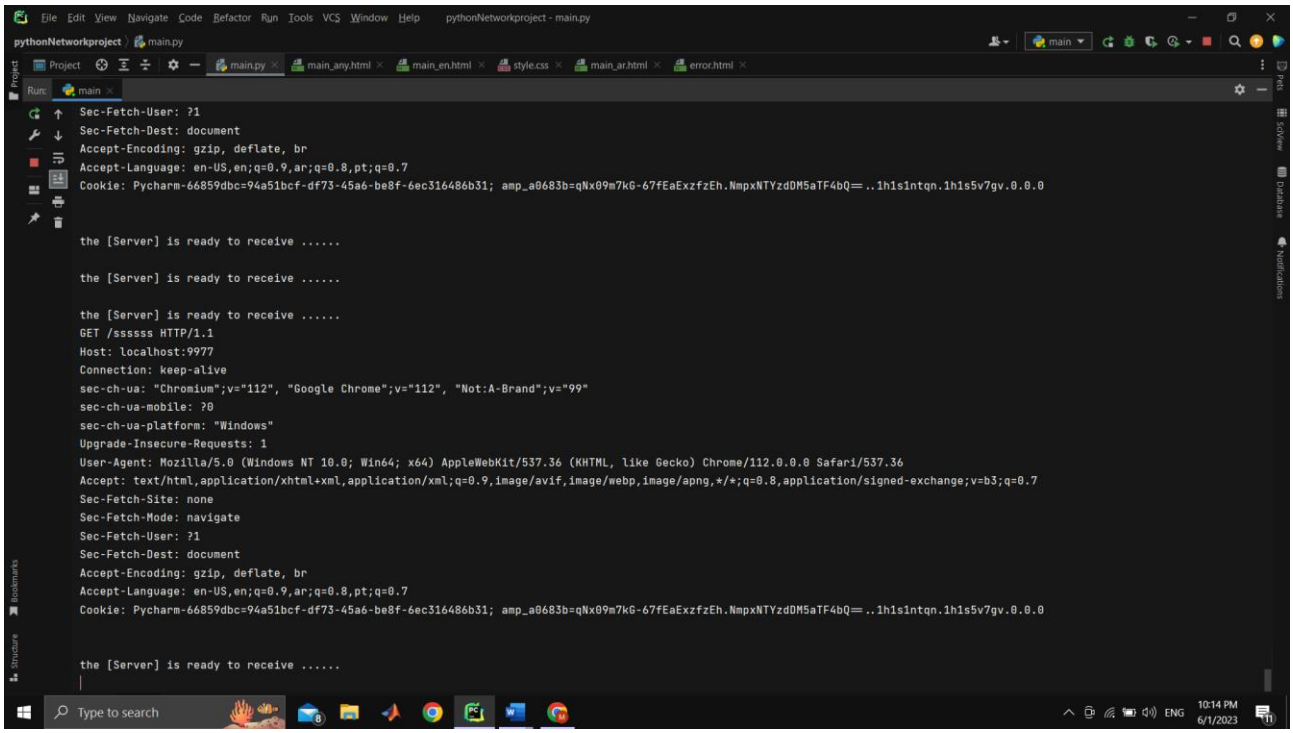


Error page :

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



The http:



The socket program:

```
from socket import *

# server_address=('',9977)# assign the port
serverport = 9977
serverSocket = socket(AF_INET, SOCK_STREAM)  # create a tcp server(craete 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>
```

```
                <div class="error_page">
                    <h1 style="color: red">The file is not found</h1>
                    <div class="names">
                        <p><b>Malak Nassar - 1200757</b></p>
                        <p><b>Tala Jebrini - 1200493</b></p>
                        <p><b>Aya Dahbour - 1201738</b></p>
                        <p><b>Hadeel Froukh - 1201585</b></p>
                        <p><b>IP: {ip} Port: {port}</b></p>
                    </div>
                </div>
                </body>
                </html>
                """

                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>
        <section class="two">
            <span class="studnet">
```

```
                <span class="student_name">  Malak Nassar</span>
            <span class="student_ID"><br>  1200757</span>
            </span>
            <span
class="space">           
             &n
bsp;            &nbs
p;           </span>
            <span  class="bpa"><br><br>Computer system engineering
student<br><br>Some the projects that I have done during different
courses:<br><br>Data Structure Projects.
               <br><br>Electrical Circuit Analysis project.<br><br>My hobbies
are drawing and coding</span>

        </section>
        <section class="three">
            <span class="studnet">
                <span class="student_name">  Tala Jebrini</span>
            <span class="student_ID"><br>  1200493</span>
            </span>
            <span
class="space">           
             &n
bsp;            &nbs
p;           </span>
            <span  class="bpa"><br><br>Computer system engineering
student<br><br>Some the projects that I have done during different
courses:<br><br>Electrical Circuit Analysis project.
                <br><br>Linux projects.<br><br>My hobbies are reading and
coding</span>
        </section>
        <section class="four">
            <span class="studnet">
                <span class="student_name">  Aya Dahbour</span>
            <span class="student_ID"><br>  1201738</span>
            </span>
            <span
class="space">           
             &n
bsp;            &nbs
p;           </span>
            <span  class="bpa"><br><br>Computer system engineering
student<br><br>Some the projects that I have done during different
courses:<br><br>Computer Organization Project.
                <br><br>Linux projects.<br><br>My hobbies are reading and
coding</span>
        </section>
        <section class="five">
            <span class="studnet">
                <span class="student_name">  Hadeel Froukh </span>
            <span class="student_ID"><br>  1201585</span>
            </span>
            <span
class="space">           
             &n
bsp;            &nbs
p;           </span>
            <span  class="bpa"><br><br>Computer system engineering
```

```
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>اهلا بكم في مساق   <span class="highlight"> شبكات
الحاسوب,</span><br>
            <span class="centered-text">هذا موقعنا الصغير</span></h1>
            </div>


    </section>
    <section class="one">
        <span class="pa">

            <p> <br> أعضاء الفريق<br><br><hr/><br>نصار ملاك -
1200757<br><br><br>جبريني تالا - 1200493<br><br><br>دحبور اية-
1201738<br><br><br>فروخ هديل - 1201585</p>
```

```
        </span>

    </section>

        </section>
        <section class="two">
            <span class="studnet">
                <span class="student_name">  Malak Nassar</span>
            <span class="student_ID"><br>  1200757</span>
            </span>
            <span
class="space">           
             &n
bsp;            &nbs
p;          </span>
            <span  class="bpa"><br><br>حاسوب هندسة طالبة<br><br>المشاريع بعض
بها قمت التي<br><br>الخوارزميات و المعلومات بنية مشاريع
                <br><br>الكهربئيه الدوائر تحليل مشروع<br><br>و الرسم هواياتي
البرمجه</span>
        </section>
        <section class="three">
            <span class="studnet">
                <span class="student_name">  Tala Jebrini</span>
            <span class="student_ID"><br>  1200493</span>
            </span>
            <span
class="space">           
             &n
bsp;            &nbs
p;          </span>
            <span  class="bpa"><br><br>حاسوب هندسة طالبة<br><br>المشاريع بعض
بها قمت التي<br><br>الكهربئيه الدوائر تحليل مشروع
                <br><br>اللينوكس مشاريع<br><br>البرمجه و القراءة هواياتي</span>
</section>
        <section class="four">
            <span class="studnet">
                <span class="student_name">  Aya Dahbour</span>
            <span class="student_ID"><br>  1201738</span>
            </span>
            <span
class="space">           
             &n
bsp;            &nbs
p;          </span>
            <span  class="bpa"><br><br>حاسوب هندسة طالبة<br><br>المشاريع بعض
بها قمت التي<br><br>الحاسوب تنظيم مشروع
                <br><br>اللينوكس مشاريع<br><br>البرمجه و القراءة هواياتي</span>
        </section>
        <section class="five">
            <span class="studnet">
                <span class="student_name">  Hadeel Froukh </span>
            <span class="student_ID"><br>  1201585</span>
            </span>
            <span
class="space">           
             &n
bsp;            &nbs
```

```
p;         </span>
            <span  class="bpa"><br><br>حاسوب هندسة طالبة<br><br>المشاريع بعض
بها قمت التي<br><br>جافا مشروع
                <br><br>الخوارزميات و المعلومات بنية شاريع<br><br>هواياتي
البرمجه و القراءة</span>
        </section>

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

            <p>  <a
href="https://www.w3schools.com/python/gloss_python_multi_line_strings.asp">
<h4> الخطوط متعدد بايثون نص <br><br></h4></a></p>
             <p> <a href="main_ar.html"> <h4> باللغة الموقع ملف الى
الانجليزيه</h4></a></p>
        </span>

    </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 {
```

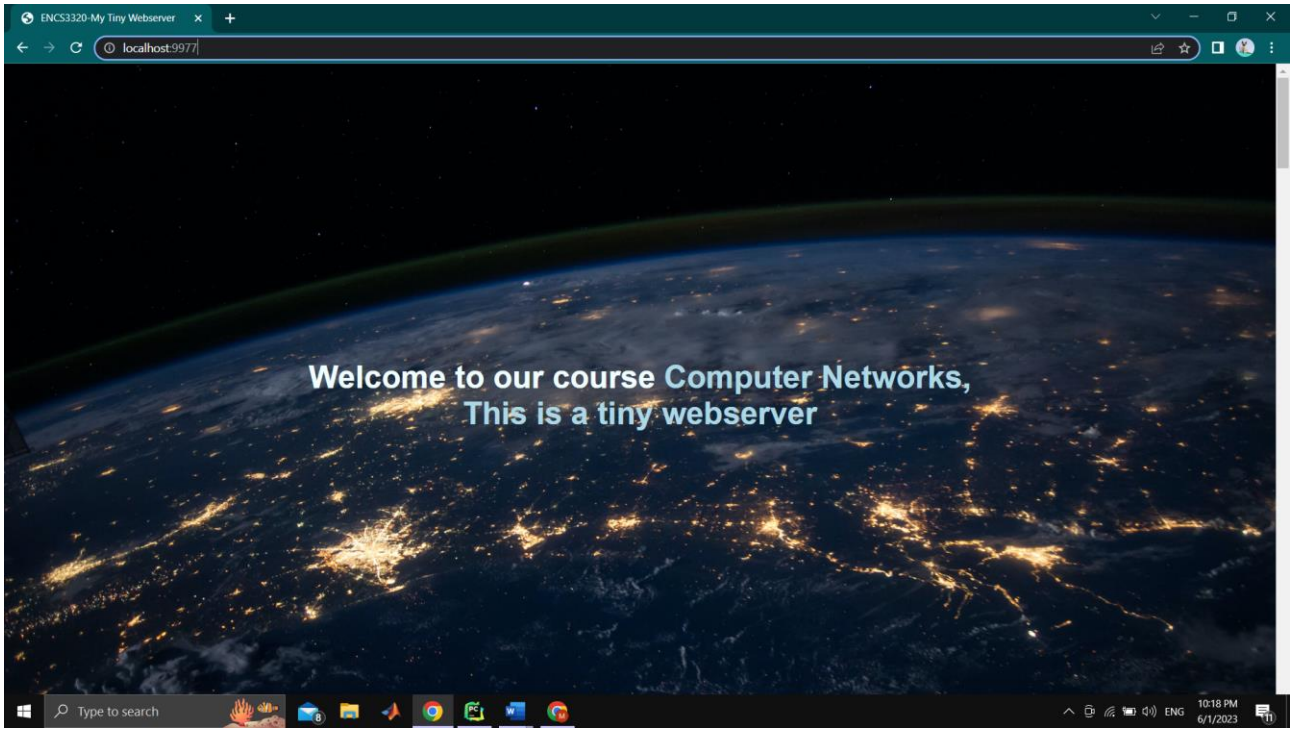
```
  display: flex;
  flex-direction: row;
  justify-content: space-between;
  align-items: flex-start;
}
.student_name{
  font-family: sans-serif;
  font-size: 50px;
  animation: fadeIn 25s;
}
.space{
  display: flex;
  flex-direction: column;
  align-items: center;
}

.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);
}



.pa {
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 :

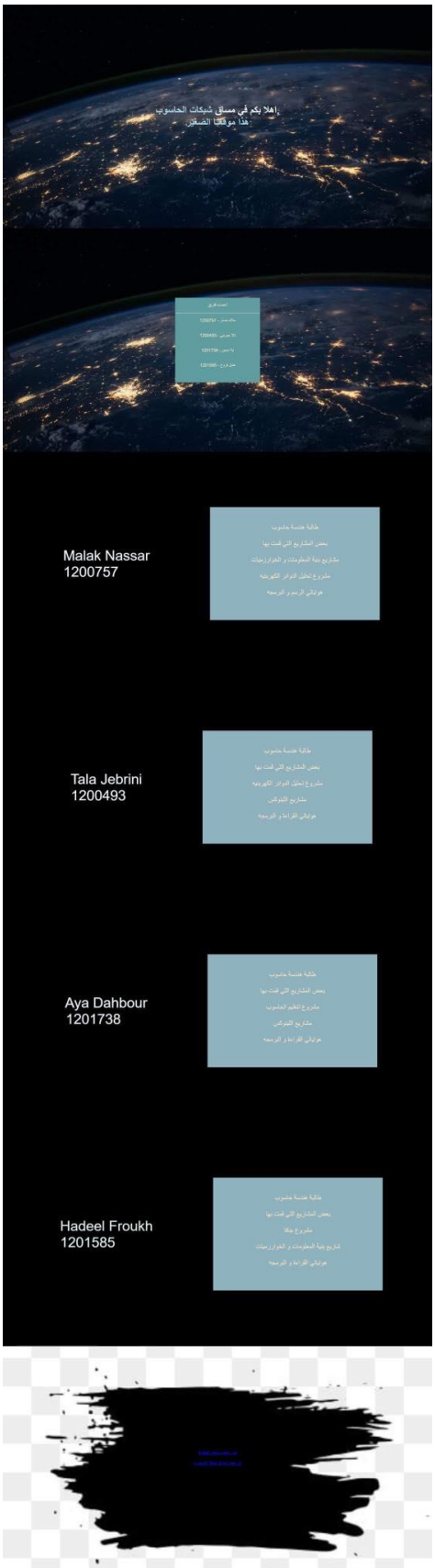


Figure 32:entire arabic page

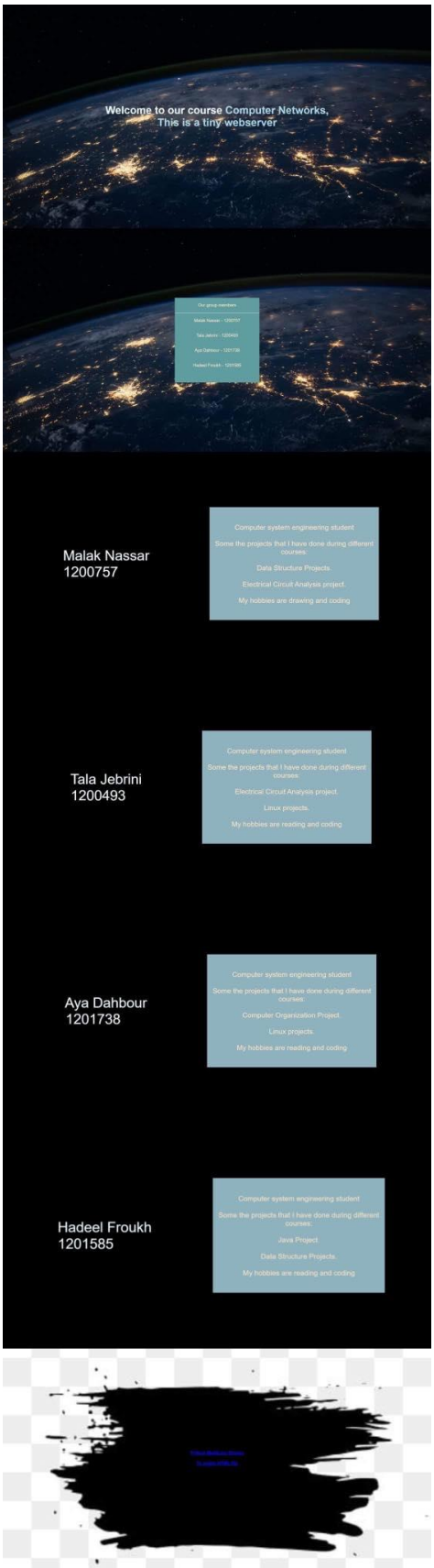
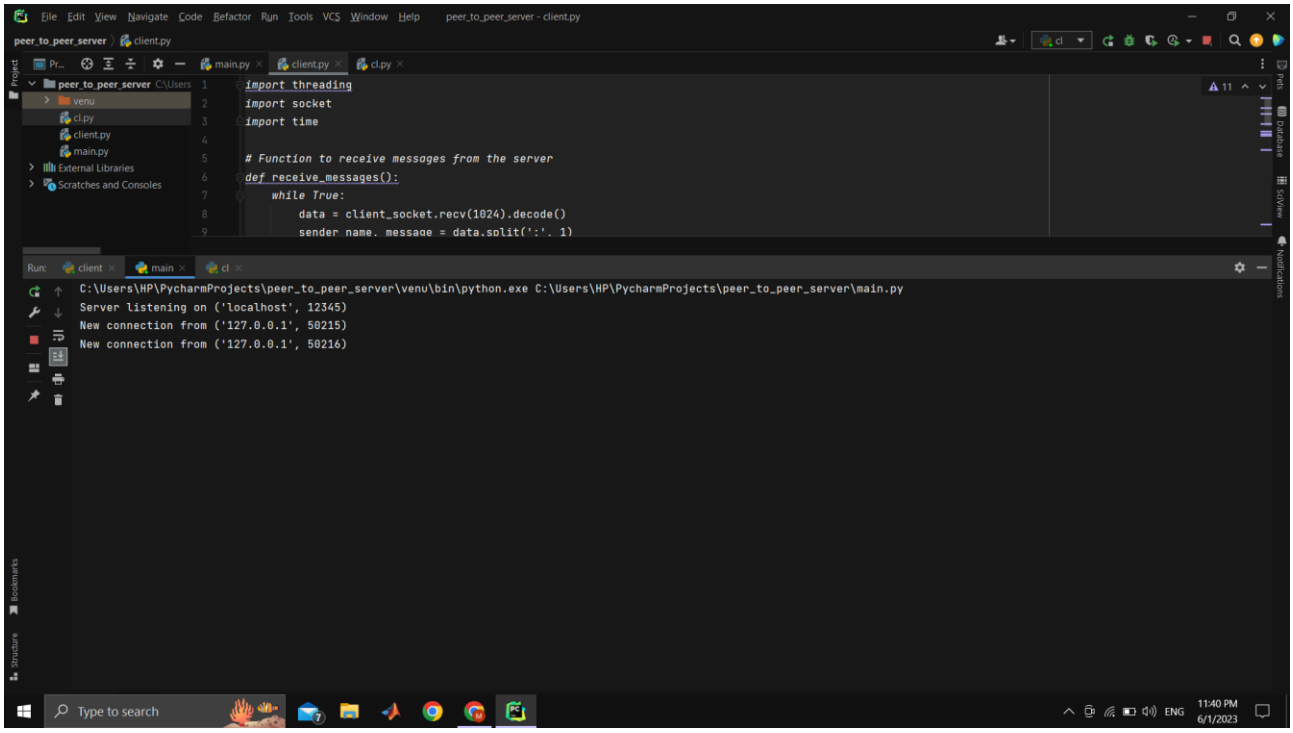


Figure 33:entire english page

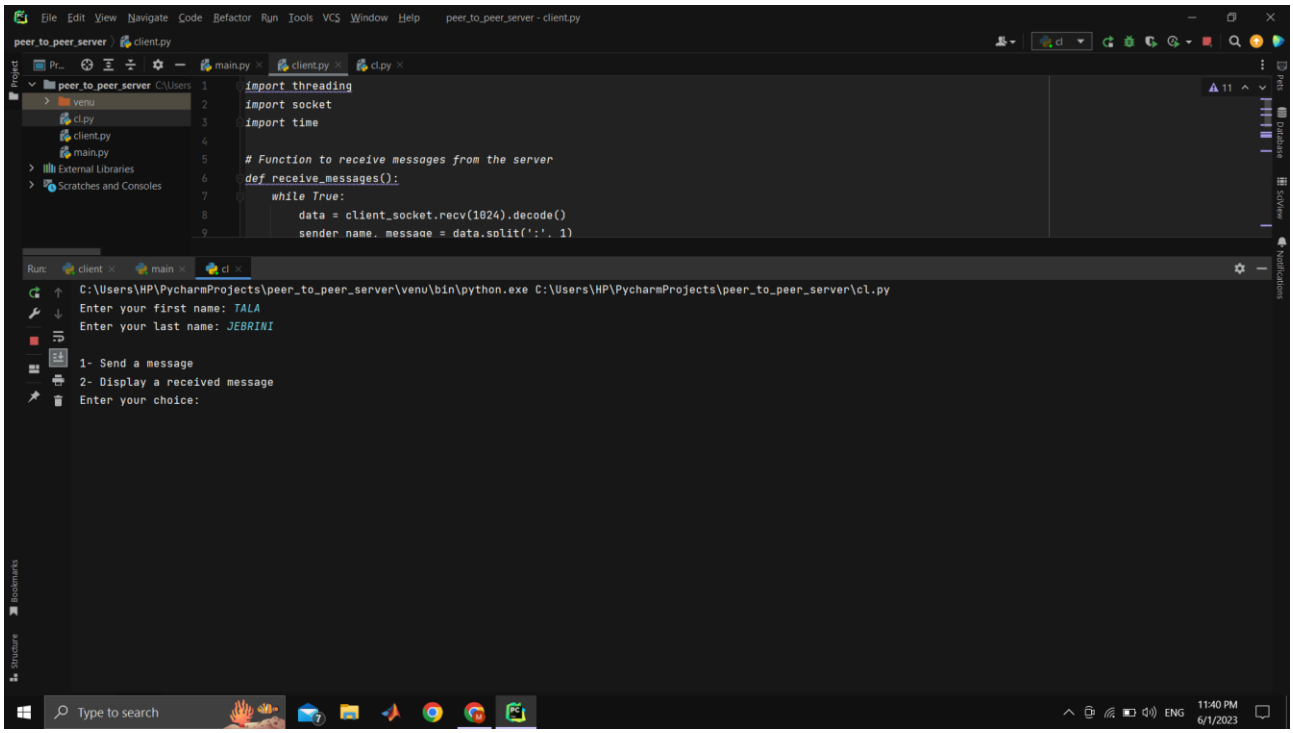
4. Part 4:

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.

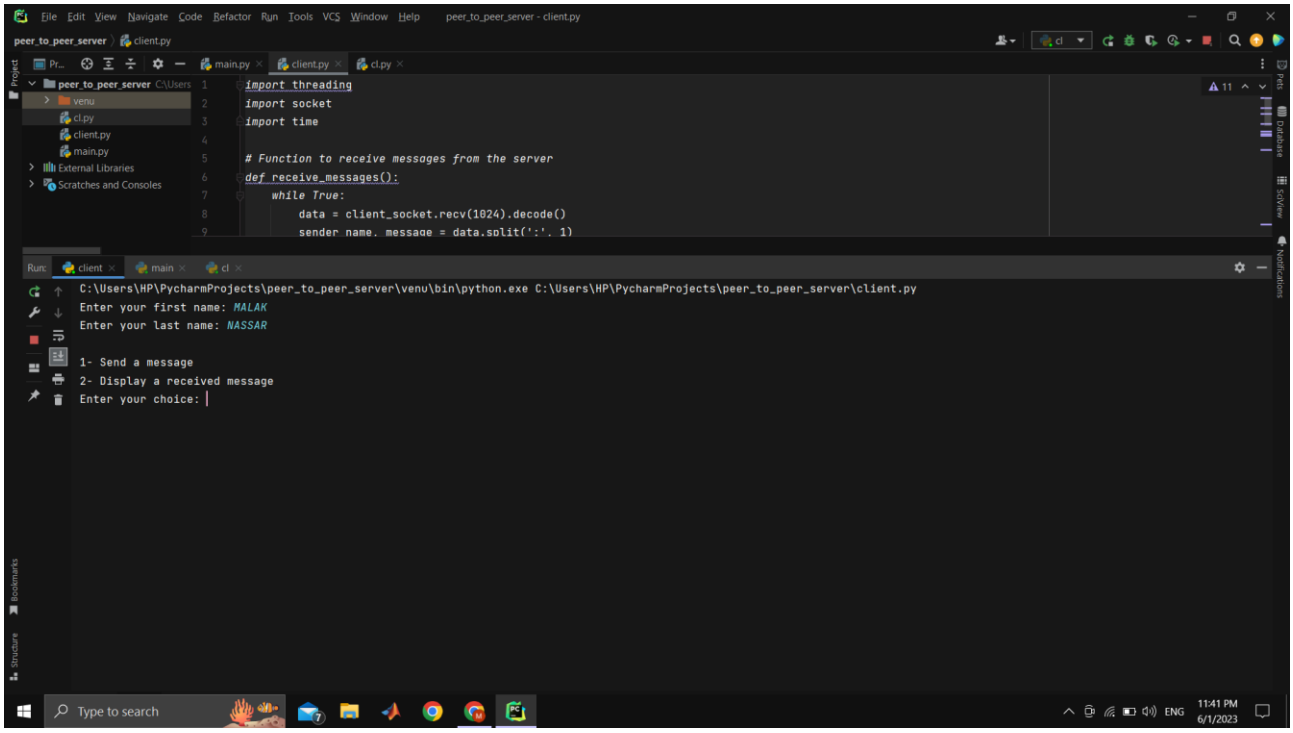
Server output:



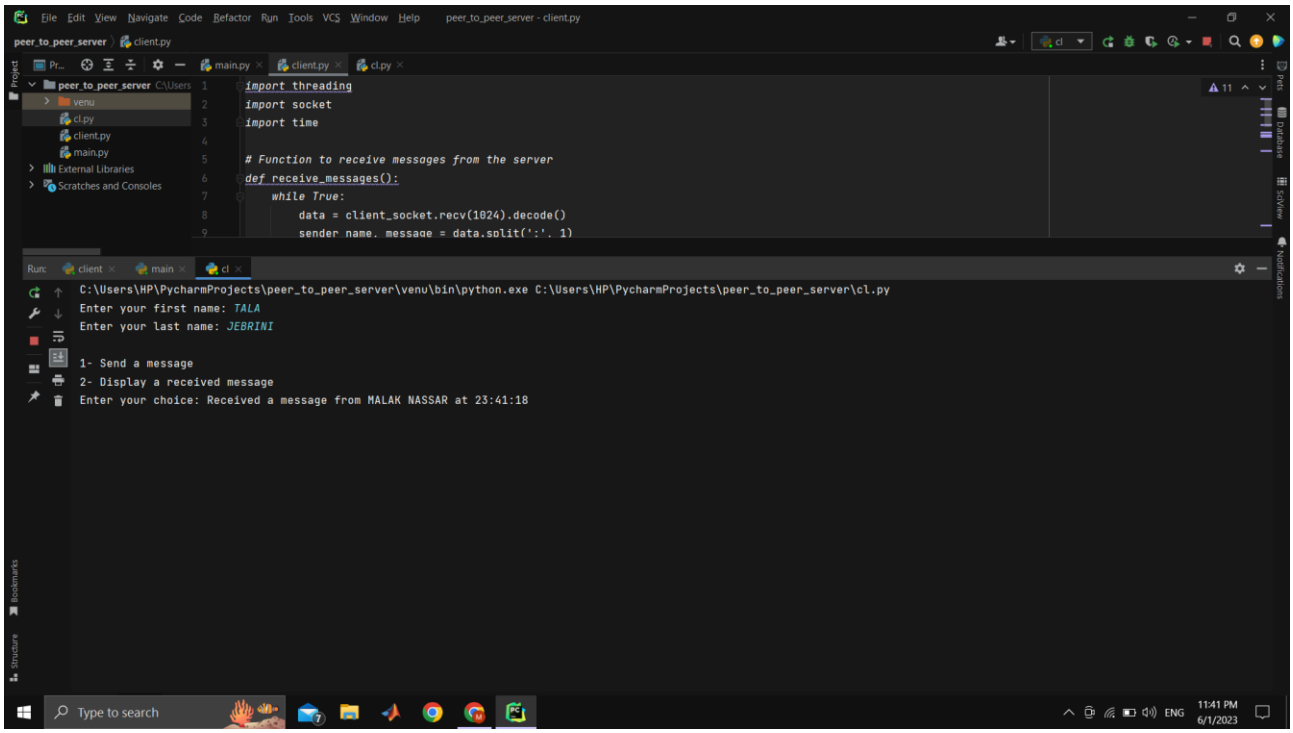
Peer #1:



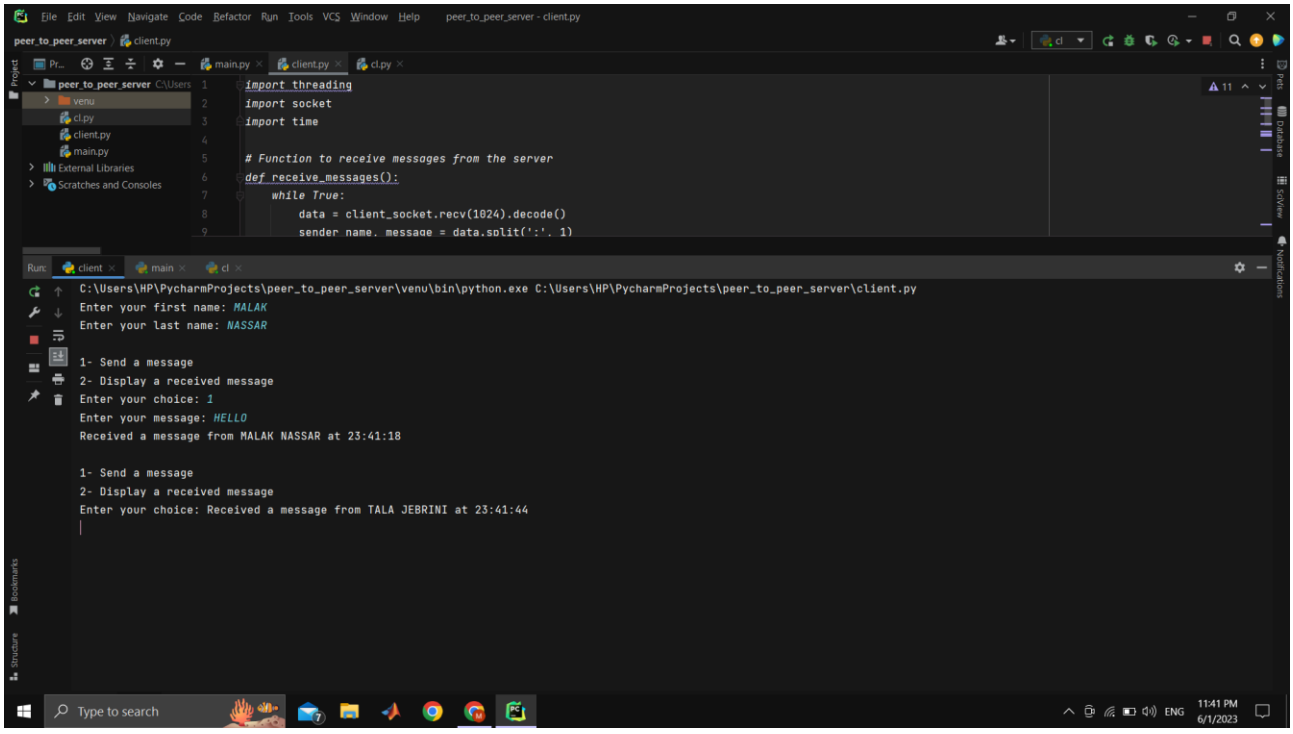
Peer #2:



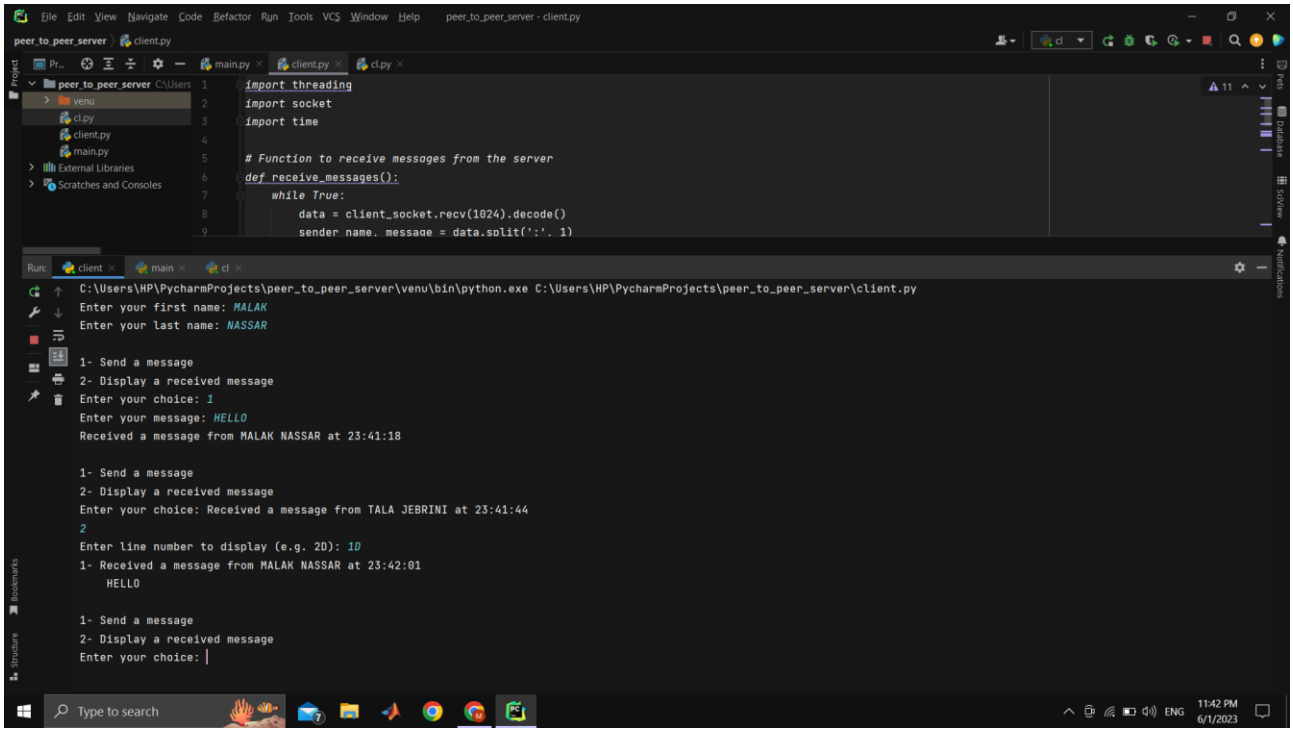
Massege receved from the first peer:

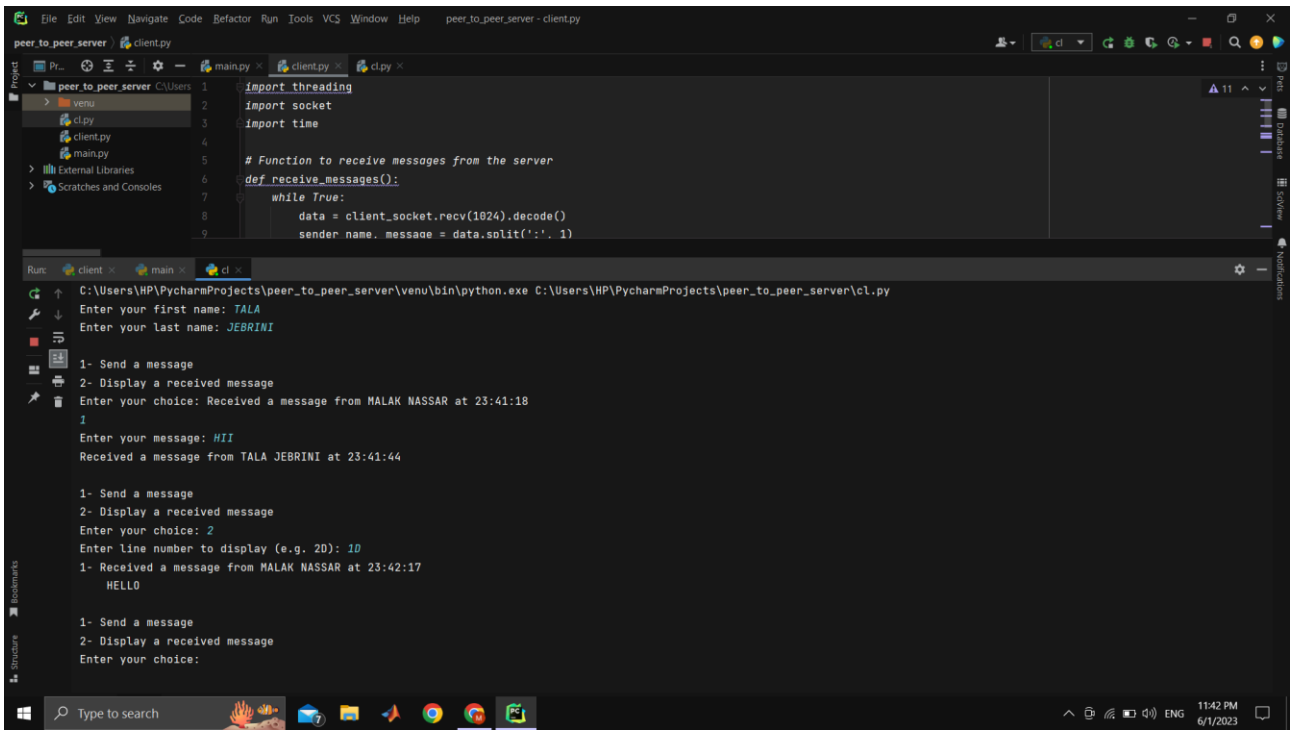


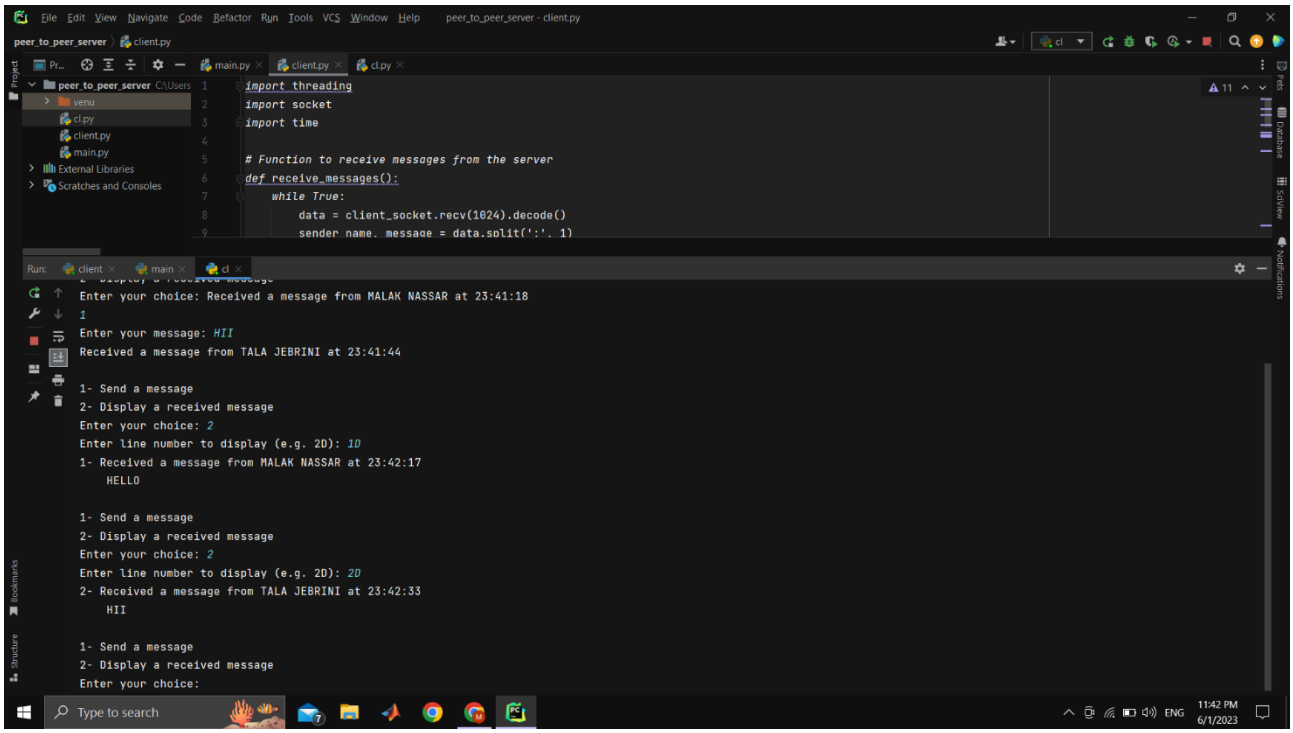
Massege receved from the secound peer:



The displayed of the reseved messages:


File Edit View Navigate Code Refactor Run Tools VCS Window Help peer_to_peer_server - client.py
peer_to_peer_server client.py
main.py client.py cl.py
peer_to_peer_server C:\Users
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: client main cl
C:\Users\HP\PycharmProjects\peer_to_peer_server\venu\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. 2D): 1D
1- Received a message from MALAK NASSAR at 23:42:01
HELLO
1- Send a message
2- Display a received message
Enter your choice:
Type to search
ENG
11:42 PM
6/1/2023


File Edit View Navigate Code Refactor Run Tools VCS Window Help peer_to_peer_server - client.py
peer_to_peer_server client.py
main.py client.py cl.py
peer_to_peer_server C:\Users
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: client main cl
C:\Users\HP\PycharmProjects\peer_to_peer_server\venu\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: HII
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. 2D): 1D
1- Received a message from MALAK NASSAR at 23:42:17
HELLO
1- Send a message
2- Display a received message
Enter your choice:
Type to search
ENG
11:42 PM
6/1/2023



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)
```