



Electrical and Computer Engineering Department
Computer Networks – ENCS3320
Project #1

Student name: Kareem Qutob **ID:1211756**

Partner name: Ghandy Ghanem **ID: 1212386**

Partner name: Qais Ayyash **ID:1213205**

Instructor name: Dr. Mohammed Jubran

Section: 2

Abstract

This group project explores fundamental networking concepts through hands-on socket programming. Divided into three parts, it encompasses understanding network utilities, implementing a TCP client-server application for screen locking, and creating a web server capable of serving dynamic content. Using Python, the team executes commands like ping, tracert, and nslookup, capturing DNS messages with Wireshark. The TCP application responds differently based on student IDs, initiating a screen lock for group members. The web server handles diverse content types, implements redirects, and displays custom error pages.

Content table :

Abstract.....	I
Part 1.....	1
What are ping, tracert , nslookup and telnet.....	1
Commands screenshot.....	1
Ping a device in the same network	1
ping www.cornell.edu	2
tracert www.cornell.edu	3
nslookup www.cornell.edu	4
wireshark to capture some DNS messages	5
Part 2 Implementation of a server-client applications	6
Server code:.....	6
Clinet code:.....	7
Code run:.....	7
Code result:.....	8
Part 3: Web server application.....	9
Server Code:.....	9
Testing on the same computer.....	11
Main_en on command line.....	11
Main_en on browser.....	13
main_en html code.....	14
Main_en CSS code.....	14
Clicking “our first website” link.....	16
Local Html page request on command line.....	16
Local html file code.....	17

Local html file css code.....	18
Main_ar request.....	19
Main_ar request on command line.....	19
Main_ar on browser.....	21
Main_ar html code.....	22
Main_ar css code.....	22
.html request.....	24
html request on command line.....	24
.Html request on browser.....	25
.CSS request.....	25
css request on command line.....	25
.css request on browser.....	26
PNG request.....	26
PNG request on command line.....	26
PNG request on browser.....	27
JPG request.....	27
JPG request on command line.....	27
JPG request on browser.....	28
Page redirecting.....	28
/cr on command line.....	28
/cr on browser.....	29
/rt on command line.....	29
/rt on browser.....	30
/so on command line.....	30
/so on browser.....	31
ERROR webpage.....	31
Error on command line.....	31
Error page on browser.....	32

Error code segment.....	32
Test from another device.....	33
From another laptop.....	33
From another phone.....	33
Conclusion.....	34

List of figures:

Figure 1: pinging a device on the same network.....	1
Figure 2:ping www.cornell.edu.....	2
Figure 3: tracert www.cornell.edu.....	3
Figure 4: nslookup www.cornell.edu	4
Figure 5: wireshark DNS captured packets.....	5
Figure 6:TCP server code.....	6
Figure 7:Client code.....	7
Figure 8:client-server testcases	7
Figure 9:server-clinet code run	8
Figure 10:Web server code	10
Figure 11: request messege for / or /en or /index.html or /main_en.html	12
Figure 12: main_en.html on browser	13
Figure 13:main_en html code	14
Figure 14: main_en css code	15
Figure 15: local html file link	16
Figure 16:local html page	16
Figure 17:local html file request	17
Figure 18: local html file code	17
Figure 19: local html file css code	18
Figure 20:python strings link	18
Figure 21: w3school python strings tutorial	18
Figure 22:/ar request message.....	20
Figure 23:main_ar on browser	21
Figure 24:main_ar html code	22
Figure 26:.html request	24
Figure 27: /.html request on browser	25
Figure 28: .css request message	25
Figure 29:.css request on browser.....	26
Figure 30: png request message	26
Figure 31:>PNG on browser.....	27
Figure 32:JPG request message	27
Figure 33:JPG on browser	28
Figure 34:/cr on command line	28
Figure 35:/cr on browser	29
Figure 36: /rt request message	29
Figure 37:/rt on browser.....	30
Figure 38:/so message request	30
Figure 39:/so on browser	31
Figure 40: error on command line.....	31
Figure 41:error on browser	32
Figure 42:error code.....	32
Figure 43:test from another laptop.....	33
Figure 44:test from a phone	33

Part 1:

1) What are ping, tracert , nslookup and telnet

ping: a command line utility that tests if a networked device is connected / responsive by measuring the RTT time of a packet sent to it

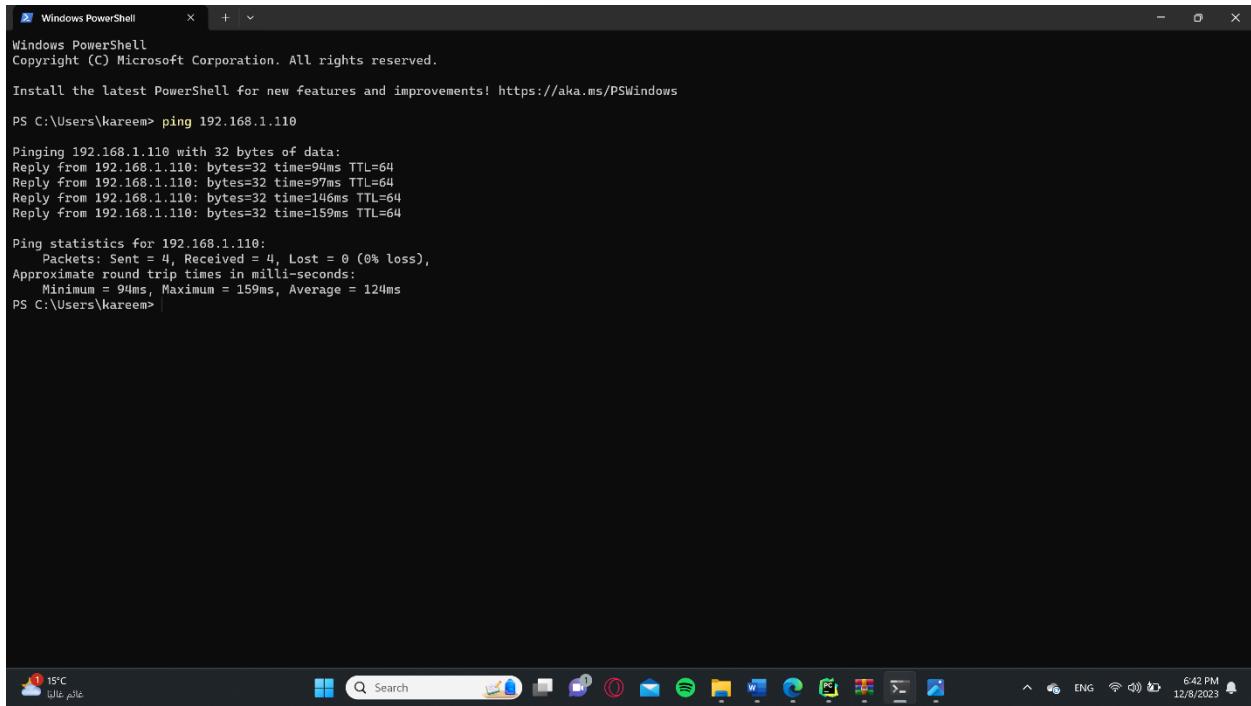
tracert: a command line utility that traces the route for a data packet from a source to a specified destination

nslookup: a command line utility to retrieve domain names, IP address

telnet: a command line protocol to establish a text-based connection between two devices over a network

2) Commands screenshot

A) Ping a device in the same network



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\kareem> ping 192.168.1.110

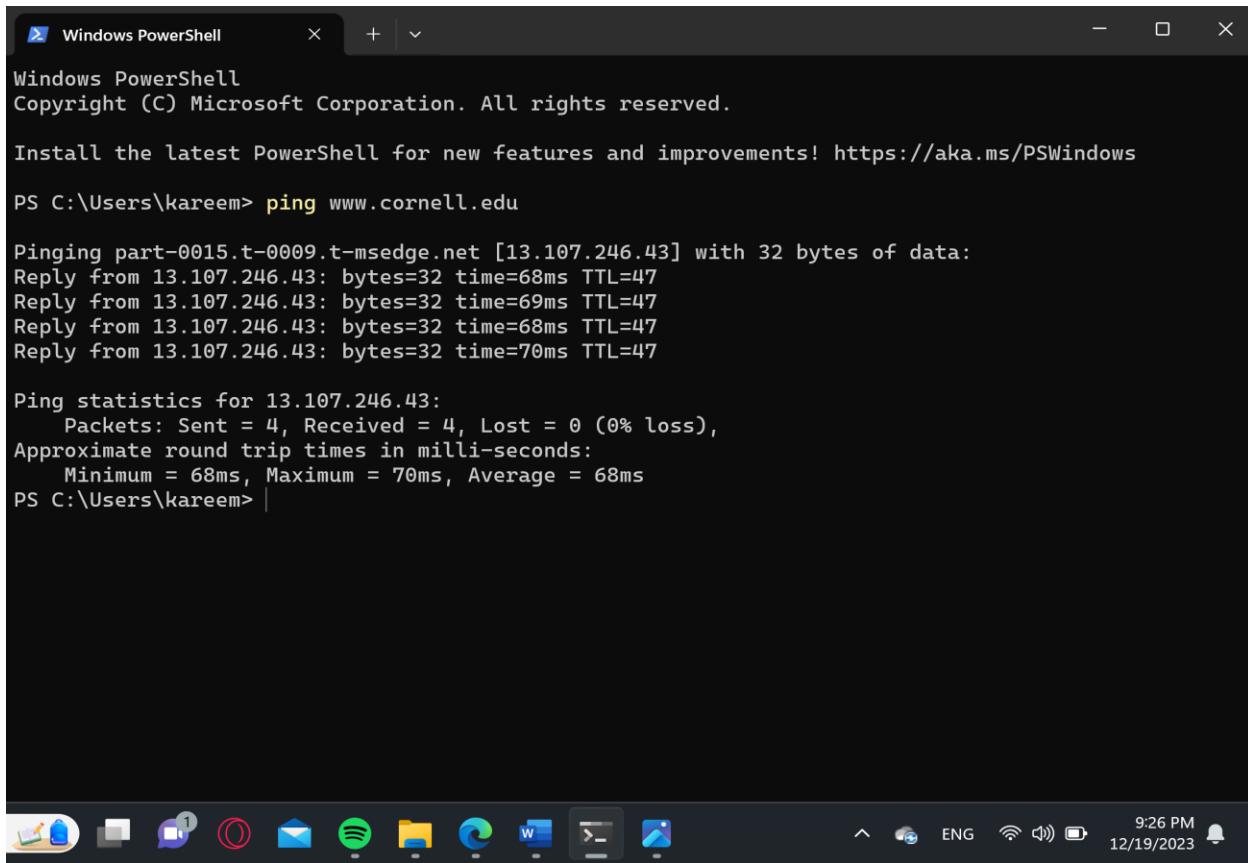
Pinging 192.168.1.110 with 32 bytes of data:
Reply from 192.168.1.110: bytes=32 time=94ms TTL=64
Reply from 192.168.1.110: bytes=32 time=97ms TTL=64
Reply from 192.168.1.110: bytes=32 time=146ms TTL=64
Reply from 192.168.1.110: bytes=32 time=159ms TTL=64

Ping statistics for 192.168.1.110:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 94ms, Maximum = 159ms, Average = 124ms
PS C:\Users\kareem>
```

Figure 1: pinging a device on the same network

we pinged a phone with IP 192.168.1.110 on the same network as the computer that we run the command in, and a successful reply was received, four packets were received with an average RTT time of 124ms. %0 percent lost indicates a stable and responsive connection which is expected because we are working on the same network.

B) ping www.cornell.edu



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\kareem> ping www.cornell.edu

Pinging part-0015.t-0009.t-msedge.net [13.107.246.43] with 32 bytes of data:
Reply from 13.107.246.43: bytes=32 time=68ms TTL=47
Reply from 13.107.246.43: bytes=32 time=69ms TTL=47
Reply from 13.107.246.43: bytes=32 time=68ms TTL=47
Reply from 13.107.246.43: bytes=32 time=70ms TTL=47

Ping statistics for 13.107.246.43:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 68ms, Maximum = 70ms, Average = 68ms
PS C:\Users\kareem> |
```

Figure 2:ping www.cornell.edu

The device used successfully communicated with the server that has IP 13.107.246.43, which is associated with the domain www.cornell.com, the 4 packets were received successfully with an average delay of 68ms. The "0% loss" suggests a stable connection, and all hardware such as cables, the network card, the router, and the modem are all working correctly, including the internet service provider.

Based on the ping results it is hard to tell if the response is from the USA the response times are relatively low which could indicate that the server is close demographically the internet's routing and infrastructure complexities make it difficult to pinpoint the exact location. The IP address 13.107.213.43 after searching is associated with Microsoft Edge servers and Microsoft uses CDNs to distribute content globally which means that server placements are in a place nearer than the USA but that does not mean that the website is not based in the USA.

C) [tracert www.cornell.edu](#)

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\kareem> tracert www.cornell.edu

Tracing route to part-0015.t-0009.t-msedge.net [13.107.246.43]
over a maximum of 30 hops:

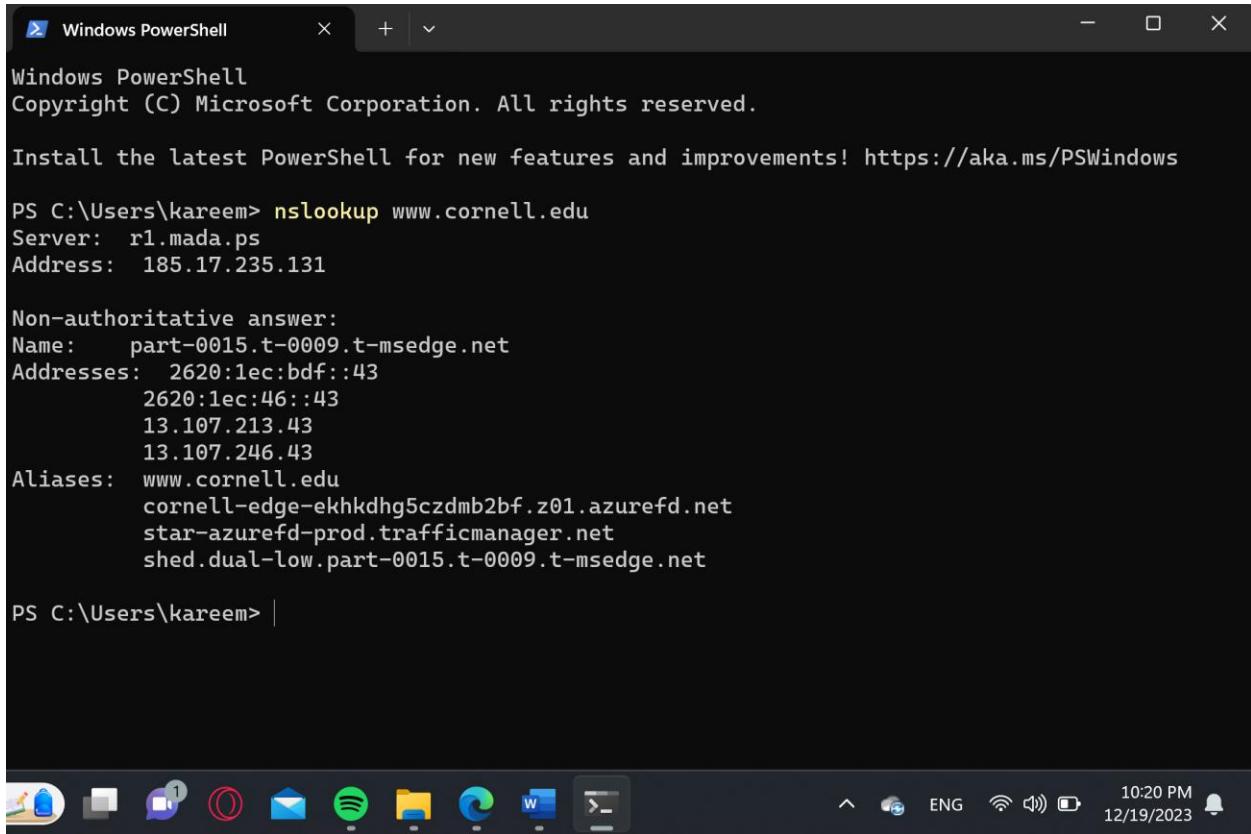
 1      6 ms      3 ms      4 ms  192.168.1.254
 2      9 ms      9 ms      7 ms  ADSL-185.17.235.203.mada.ps [185.17.235.203]
 3      5 ms      5 ms      5 ms  172.16.250.145
 4     12 ms      7 ms      6 ms  172.16.250.209
 5     63 ms     58 ms     58 ms  ix-xe-7-3-0-0.tcore2.av2-amsterdam.as6453.net [80.231.152.77]
 6     59 ms     55 ms     62 ms  if-be-25-2.ecore1.fr0-frankfurt.as6453.net [80.231.65.21]
 7     63 ms     76 ms     60 ms  195.219.223.1
 8     65 ms     57 ms     64 ms  ae31-0.icr02.fra21.ntwk.msn.net [104.44.230.18]
 9     62 ms     59 ms     63 ms  ae25-0.ier04.dus30.ntwk.msn.net [104.44.235.132]
10     63 ms     66 ms     61 ms  13.104.140.164
11     *         *         * Request timed out.
12     *         *         * Request timed out.
13     *         *         * Request timed out.
14     *         *         * Request timed out.
15    76 ms     70 ms     70 ms  13.107.246.43

Trace complete.
PS C:\Users\kareem>
```

Figure 3: [tracert www.cornell.edu](#)

The tracert command is used to trace the route of data packets from our device to a specified destination in this case [www.cronell.edu](#) showing IP addresses of hops and routers with their RTT, as shown in Figure 3 there are 15 hops with the first one being the local router then the second one is an ISP , the * means that the request timed out which mean the hops did not respond to the tracert request

D) nslookup www.cornell.edu



A screenshot of a Windows PowerShell window titled "Windows PowerShell". The command "nslookup www.cornell.edu" is run, showing the following output:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\kareem> nslookup www.cornell.edu
Server:  r1.mada.ps
Address: 185.17.235.131

Non-authoritative answer:
Name:    part-0015.t-0009.t-msedge.net
Addresses: 2620:1ec:bdf::43
          2620:1ec:46::43
          13.107.213.43
          13.107.246.43
Aliases:  www.cornell.edu
          cornell-edge-ekhkdhg5czdmb2bf.z01.azurefd.net
          star-azurefd-prod.trafficmanager.net
          shed.dual-low.part-0015.t-0009.t-msedge.net

PS C:\Users\kareem> |
```

The taskbar at the bottom shows various icons and the system tray indicates the date and time as 10:20 PM on 12/19/2023.

Figure 4: nslookup www.cornell.edu

nslookup command reveal DNS information of www.cornell.com , a list of all IPs associated with the domain part, and alternative names with the same IP address

3) wireshark to capture some DNS messages.

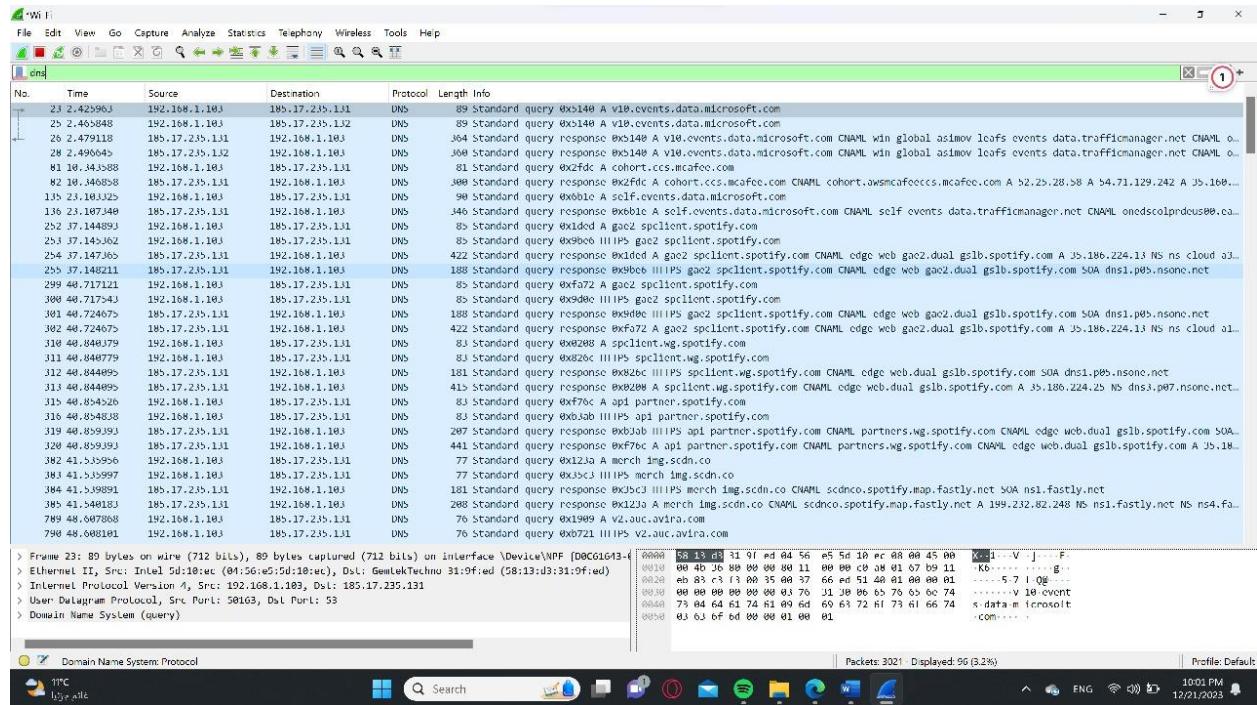


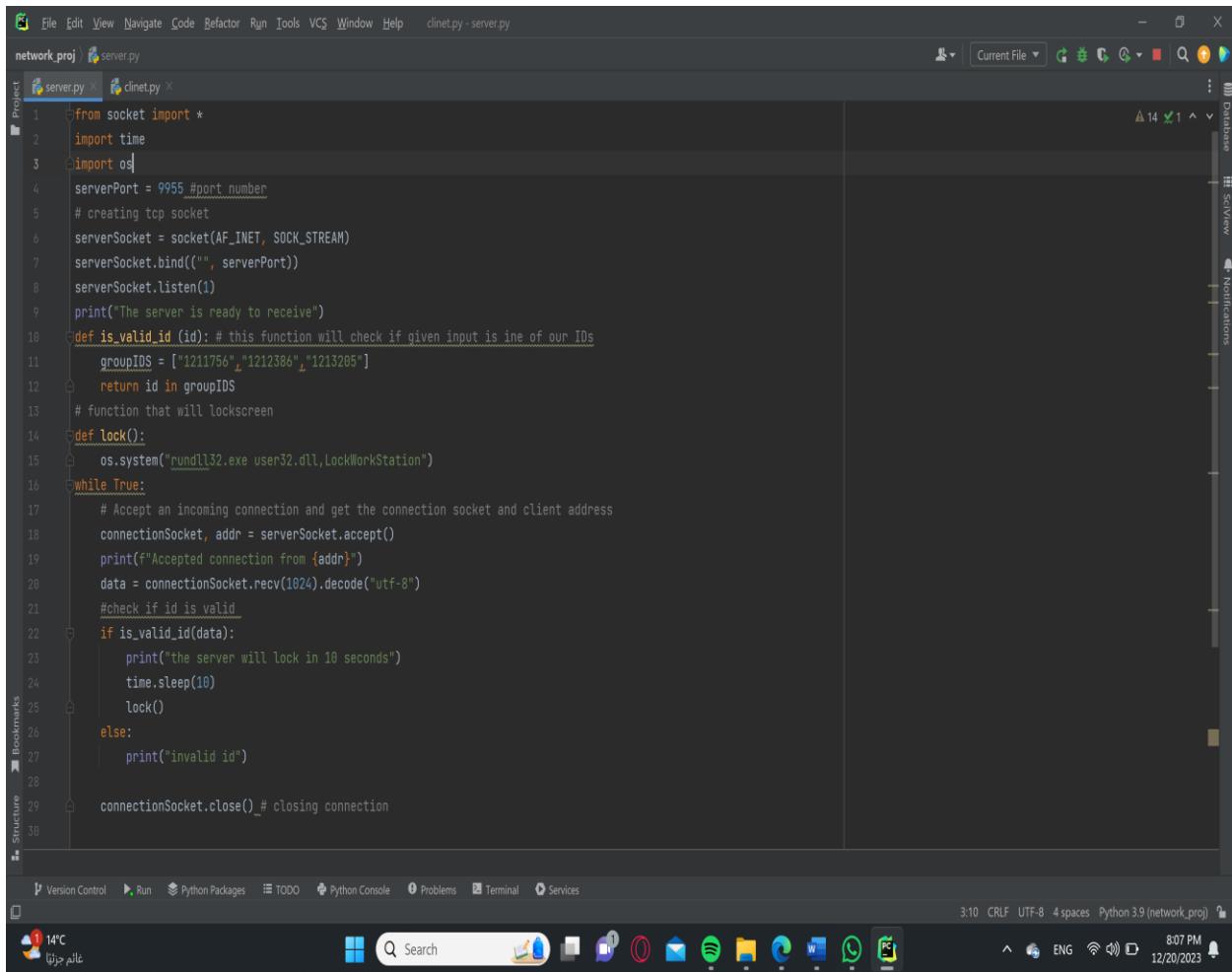
Figure 5: wireshark DNS captured packets

The capture paints a picture of internet activity, with a focus on DNS resolution and potential Spotify usage(which I was using at that time).and many other dns queries that suggest the computer was actively resolving domain names for various online activities.

Part 2: Implementation of a server-client applications

Using socket programming in python we implemented a simple TCP server-client application that listen on port 9955, the server will wait message from client if message has one of our group IDs it will display a message that the OS will close the lock screen in 10 seconds, otherwise it will display an error message.

Server code:



The screenshot shows a Python development environment with the following details:

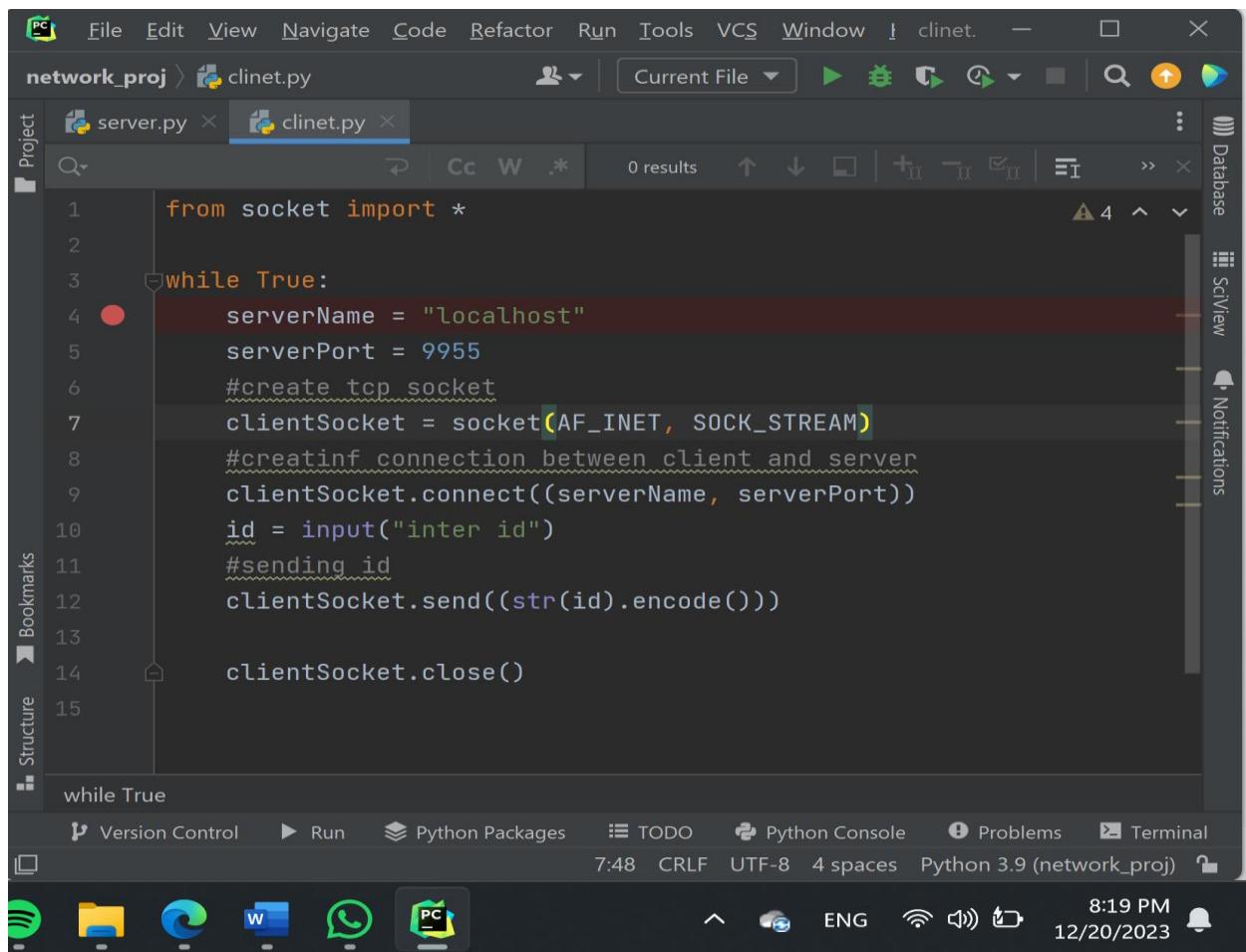
- Project:** network_proj
- Files:** server.py (the current file), clinet.py, and server.py (another instance).
- Code Content:**

```
from socket import *
import time
import os
serverPort = 9955 #port number
# creating top socket
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind(("0.0.0.0", serverPort))
serverSocket.listen(1)
print("The server is ready to receive")
def is_valid_id(id): # this function will check if given input is one of our IDs
    groupIDs = ["1211756", "1212386", "1213205"]
    return id in groupIDs
# function that will lockscreen
def lock():
    os.system("rundll32.exe user32.dll,LockWorkStation")
while True:
    # Accept an incoming connection and get the connection socket and client address
    connectionSocket, addr = serverSocket.accept()
    print(f"Accepted connection from {addr}")
    data = connectionSocket.recv(1024).decode("utf-8")
    #check if id is valid
    if is_valid_id(data):
        print("the server will lock in 10 seconds")
        time.sleep(10)
        lock()
    else:
        print("invalid id")
    connectionSocket.close() # closing connection
```
- IDE Features:** The interface includes tabs for Version Control, Run, Python Packages, TODO, Python Console, Problems, Terminal, Services, and a bottom toolbar with various icons.
- System Status:** The taskbar at the bottom shows the date (12/20/2023), time (8:07 PM), battery level (14%), and system status (ENG).

Figure 6: TCP server code

A simple server code that create a TCP connection and wait for a client to send input to it

Client code:



The screenshot shows the PyCharm IDE interface with the project 'network_proj' open. The 'clinet.py' file is the active editor. The code implements a simple TCP client. It starts by importing the socket module. A while loop continues to run. Inside the loop, it sets the server name to 'localhost' and the port to 9955. It then creates a TCP socket using socket(AF_INET, SOCK_STREAM). After creating the socket, it connects to the specified server. It prompts the user for an ID and sends it to the server via the socket. Finally, it closes the client socket. The code is annotated with comments explaining each step.

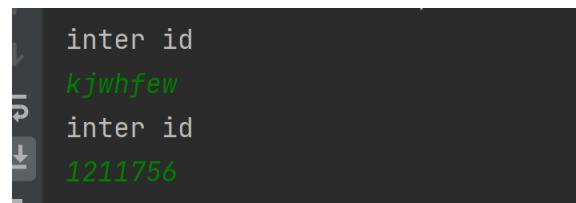
```
from socket import *
while True:
    serverName = "localhost"
    serverPort = 9955
    #create tcp socket
    clientSocket = socket(AF_INET, SOCK_STREAM)
    #creatinf connection between client and server
    clientSocket.connect((serverName, serverPort))
    id = input("inter id")
    #sending id
    clientSocket.send((str(id).encode()))
    clientSocket.close()
```

Figure 7:Client code

A simple client code that will create a TCP connection and send input for the server to process

Code run:

For testing we will send two inputs the first input will be a random string and the second will be a an ID for one of group members

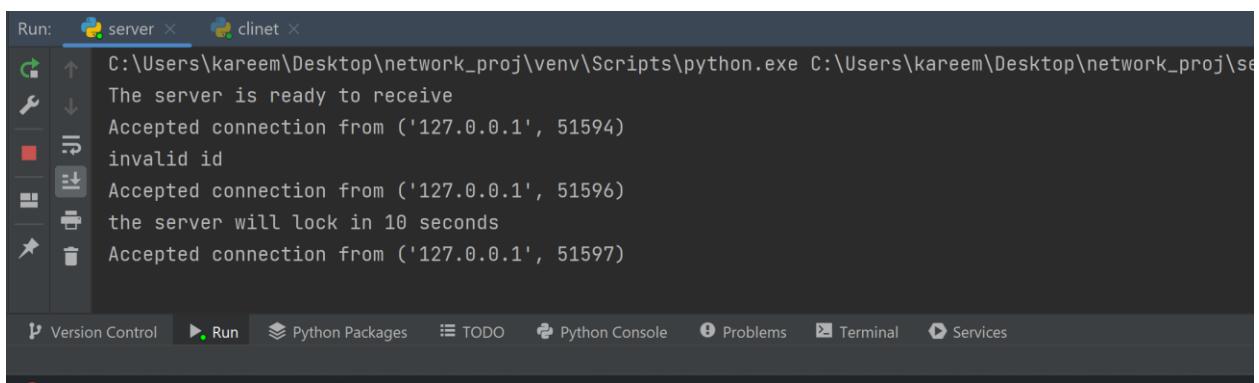


The terminal window shows the interaction between the client and the server. The client sends two messages: a random string 'kjwhfew' and an ID '1211756'. The server's response is not visible in the provided screenshot.

```
inter id
kjwhfew
inter id
1211756
```

Figure 8:client-server testcases

Code result:



The screenshot shows a terminal window with two tabs: "server" and "client". The "server" tab is active and displays the following output:

```
C:\Users\kareem\Desktop\network_proj\venv\Scripts\python.exe C:\Users\kareem\Desktop\network_proj\se
The server is ready to receive
Accepted connection from ('127.0.0.1', 51594)
invalid id
Accepted connection from ('127.0.0.1', 51596)
the server will lock in 10 seconds
Accepted connection from ('127.0.0.1', 51597)
```

The terminal interface includes standard navigation keys (up, down, left, right) and icons for copy, paste, and delete. Below the terminal are several tabs: Version Control, Run, Python Packages, TODO, Python Console, Problems, Terminal, and Services. The "Run" tab is highlighted.

Figure 9:server-clinet code run

In the two test cases, the server accepted the connection, the first input was decalerd invalid, the second activated the lock screen after 10 seconds .

Part 3: Web server application

In this part we will build a simple but a complete web server using python listening on port 9966

Server Code:

```
1  import socket
2  from socket import * # importing socket programming library
3  serverPort=9966 # user port number
4  serverSocket = socket(AF_INET,SOCK_STREAM) # creating tcp socket
5  serverSocket.bind(("",serverPort))
6  serverSocket.listen(1) # server begin listing to TCP request
7  print("The server is ready to receive")
8  while True:
9      conn, addr = serverSocket.accept()
10     sentence = conn.recv(2048).decode()
11     print(addr)
12     print(sentence)
13     #spliting the request and taking the first line
14     request = sentence.split("\n")
15     #taking ip and address
16     ip = addr[0]
17     port = addr[1]
18
19     #cases for diffrent requests
20     if request[0].__contains__("GET /index.html HTTP/1.1") or request[0].__contains__("GET /main_in.html HTTP/1.1") or (request[0].__contains__("GET /en HTTP/1.1") or request[0].
21         conn.send("HTTP/1.1 200 OK\r\n".encode())
22         conn.send("Content-Type: text/html \r\n".encode())
23         conn.send("\r\n".encode())
24         f1 = open('main_en.html', "rb")
25         conn.send(f1.read())
26     elif request[0].__contains__("/ar HTTP/1.1"):
27         conn.send("HTTP/1.1 200 OK\r\n".encode())
28         conn.send("Content-Type: text/html \r\n".encode())
29         conn.send("\r\n".encode())
30         f1 = open('main_ar.html', "rb")
31
32     elif request[0].__contains__("main_ar.html HTTP/1.1"):
33         conn.send("HTTP/1.1 200 OK\r\n".encode())
34         conn.send("Content-Type: text/html \r\n".encode())
35         conn.send("\r\n".encode())
36         f1 = open('main_ar.html', "rb")
37         conn.send(f1.read())
38     elif request[0].__contains__("first.html HTTP/1.1"):
39         conn.send("HTTP/1.1 200 OK\r\n".encode())
40         conn.send("Content-Type: text/html \r\n".encode())
41         conn.send("\r\n".encode())
42         f1 = open('first.html', "rb")
43         conn.send(f1.read())
44     elif request[0].__contains__("main_en.html HTTP/1.1"):
45         conn.send("HTTP/1.1 200 OK\r\n".encode())
46         conn.send("Content-Type: text/html \r\n".encode())
47         conn.send("\r\n".encode())
48         f1 = open('main_en.html', "rb")
49         conn.send(f1.read())
50
51     elif request[0].__contains__("main_en.css HTTP/1.1"):
52         conn.send("HTTP/1.1 200 OK\r\n".encode())
53         conn.send("Content-Type: text/css \r\n".encode())
54         conn.send("\r\n".encode())
55         f1 = open("main_en.css", "rb")
56         conn.send(f1.read())
57     elif request[0].__contains__("main_ar.css HTTP/1.1"):
58         conn.send("HTTP/1.1 200 OK\r\n".encode())
59         conn.send("Content-Type: text/css \r\n".encode())
60         conn.send("\r\n".encode())
61         f1 = open("main_ar.css", "rb")
```

```
conn.send(f1.read())
elif request[0]__contains__(“s.css HTTP/1.1”):
    conn.send(“HTTP/1.1 200 OK\r\n”.encode())
    conn.send(“Content-Type: text/css \r\n”.encode())
    conn.send(“\r\n”.encode())
    f1 = open(“s.css”, “rb”)
    conn.send(f1.read())
elif request[0]__contains__(“palestine.png HTTP/1.1”):
    conn.send(“HTTP/1.1 200 OK\r\n”.encode())
    conn.send(“Content-Type: image/png \r\n”.encode())
    conn.send(“\r\n”.encode())
    f1 = open(“palestine.png”, “rb”)
    conn.send(f1.read())
elif request[0]__contains__(“nature.png HTTP/1.1”):
    conn.send(“HTTP/1.1 200 OK\r\n”.encode())
    conn.send(“Content-Type: image/png \r\n”.encode())
    conn.send(“\r\n”.encode())
    f1 = open(“nature.png”, “rb”)
    conn.send(f1.read())
elif request[0]__contains__(“pal.jpeg HTTP/1.1”):
    conn.send(“HTTP/1.1 200 OK\r\n”.encode())
    conn.send(“Content-Type: image/png \r\n”.encode())
    conn.send(“\r\n”.encode())
    f1 = open(“pal.jpeg”, “rb”)
    conn.send(f1.read())
elif request[0]__contains__(“qais.jpg HTTP/1.1”):
    conn.send(“HTTP/1.1 200 OK\r\n”.encode())
    conn.send(“Content-Type: image/png \r\n”.encode())
    conn.send(“\r\n”.encode())
    f1 = open(“qais.jpg”, “rb”)
while True : elif request[0]__contains__(“l..
```

```
conn.send(fi.read())
elif request[0].__contains__("(dog.jpg HTTP/1.1"):
    conn.send("HTTP/1.1 200 OK\r\n".encode())
    conn.send("Content-Type: image/png \r\n".encode())
    conn.send("\r\n".encode())
    f1 = open("dog.jpg", "rb")
    conn.send(f1.read())
elif request[0].__contains__("(ghandy.jpg HTTP/1.1"):
    conn.send("HTTP/1.1 200 OK\r\n".encode())
    conn.send("Content-Type: image/png \r\n".encode())
    conn.send("\r\n".encode())
    f1 = open("ghandy.jpg", "rb")
    conn.send(f1.read())
elif request[0].__contains__("(kareem.jpg HTTP/1.1"):
    conn.send("HTTP/1.1 200 OK\r\n".encode())
    conn.send("Content-Type: image/png \r\n".encode())
    conn.send("\r\n".encode())
    f1 = open("kareem.jpg", "rb")
    conn.send(f1.read())
elif request[0].__contains__("(cr HTTP/1.1"):
    conn.send("HTTP/1.1 307 Temporary Redirect\r\n".encode())
    conn.send("Location: https://www.cornell.edu/\r\n".encode())
    conn.send("\r\n".encode())
elif request[0].__contains__("(so HTTP/1.1"):
    conn.send("HTTP/1.1 307 Temporary Redirect\r\n".encode())
    conn.send("Location: https://stackoverflow.com/\r\n".encode())
    conn.send("\r\n".encode())
elif request[0].__contains__("(rt HTTP/1.1"):
    conn.send("HTTP/1.1 307 Temporary Redirect\r\n".encode())
    conn.send("Location: https://catalinawiezeit.edu/\r\n".encode())
while True : elif request[0].__contains__("k...  
Version Control Run Python Packages TODO Python Console Problems Terminal Services
```

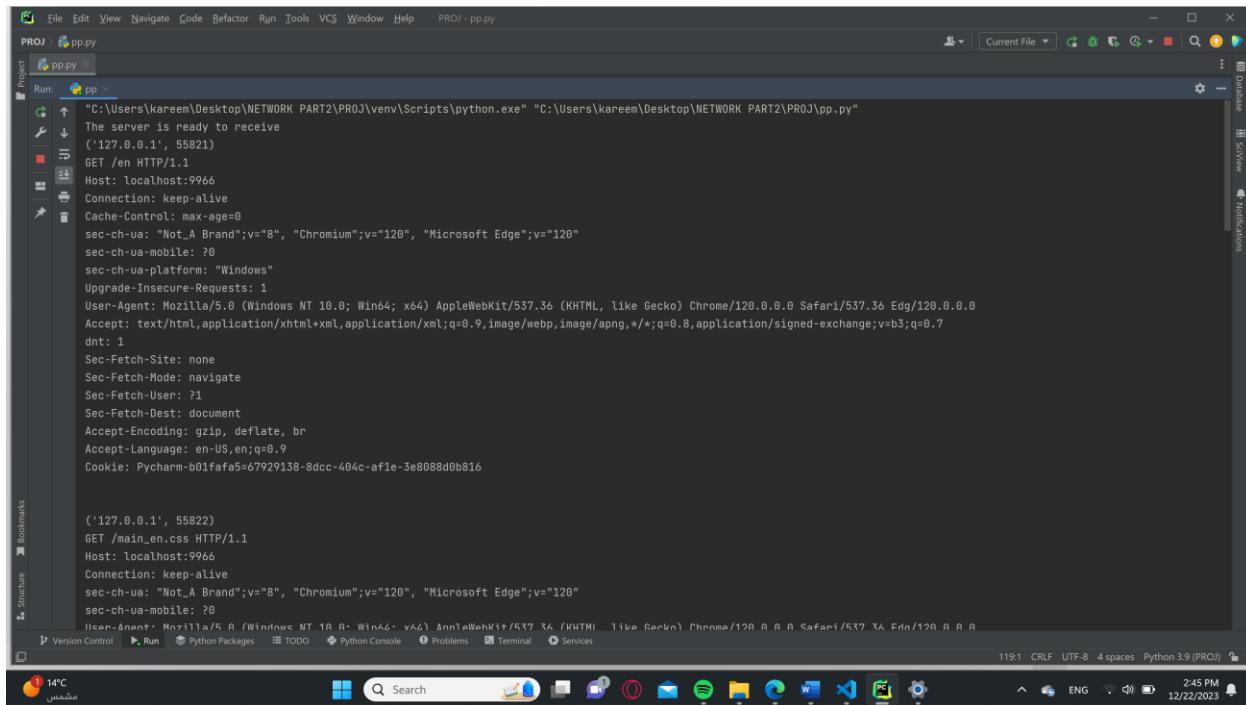
```
122     conn.send("\r\n".encode())
123 else: # this handle error cases
124     conn.send("HTTP/1.1 404 Not Found \r\n".encode())
125     # content type
126     conn.send("Content-Type: text/html; charset=utf-8\r\n".encode())
127     conn.send("\r\n".encode())
128     # s contain the html page of the error 404
129     s = "<html><head><title>Error 404</title><style type='text/css'> h1{ font-size:4em; text-align:center; margin:0; padding:0; </style><body> IP: " + ip + " Port=" + str(port) + "</em></p></body></html>\r\n"
130     data = s.encode()
131     conn.send(data)
132
133
134 conn.close()#closing connections
```

Figure 10: Web server code

Testing on the same computer:

- 1- if the request is / or /index.html or /main_en.html or /en (for example localhost:9966/ or localhost:9966/en) then the server should send main_en.html file with Content-Type: text/html.

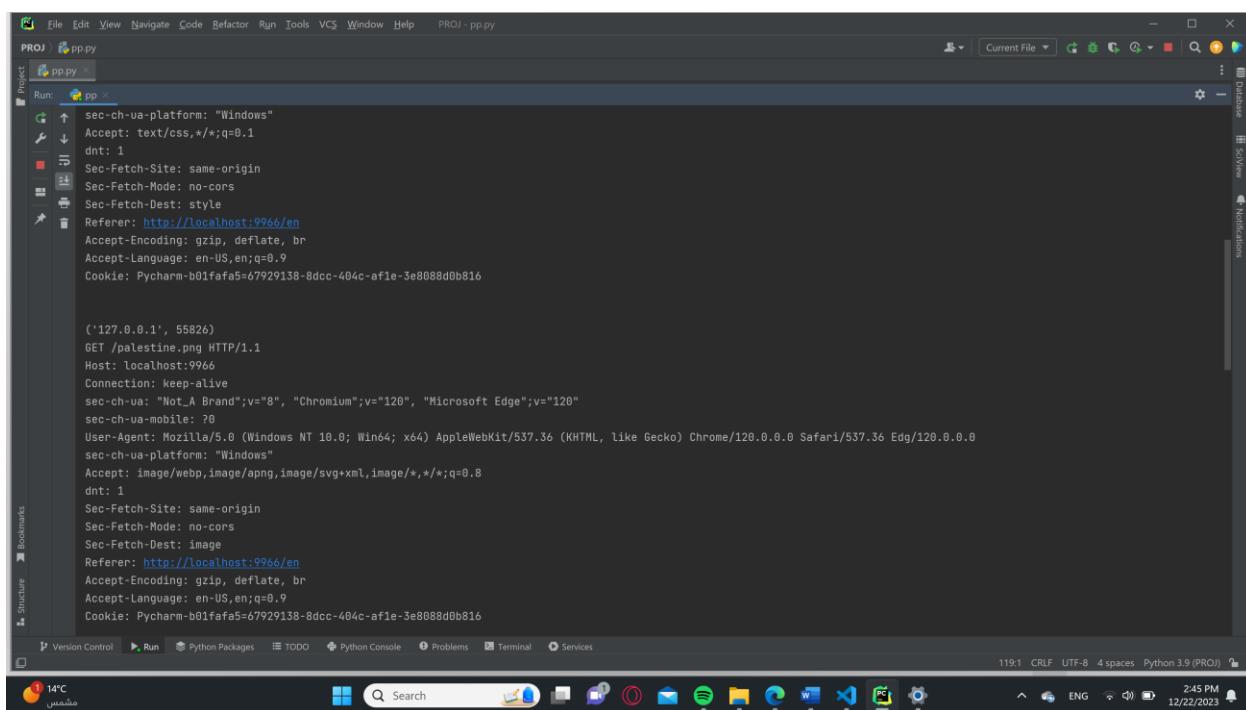
Main_en on command line:



```
"C:\Users\kareem\Desktop\NETWORK PART2\PROJ\venv\Scripts\python.exe" "C:\Users\kareem\Desktop\NETWORK PART2\PROJ\pp.py"
Run: pp.py
('127.0.0.1', 55822)
The server is ready to receive
GET /main_en.css HTTP/1.1
Host: localhost:9966
Connection: keep-alive
Cache-Control: max-age=0
sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="120"
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/120.0.0.0 Safari/537.36 Edg/120.0.0.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
dnt: 1
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
Cookie: Pycharm-b01fafaf5=67929138-8dcc-404c-af1e-3e8088d0b816

('127.0.0.1', 55822)
GET /main_en.css HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="120"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36 Edg/120.0.0.0
Accept: text/css,*/*;q=0.1
dnt: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: style
Referer: http://localhost:9966/en
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-b01fafaf5=67929138-8dcc-404c-af1e-3e8088d0b816

('127.0.0.1', 55826)
GET /palestine.png HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="120"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36 Edg/120.0.0.0
sec-ch-ua-platform: "Windows"
Accept: image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
dnt: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: image
Referer: http://localhost:9966/en
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-b01fafaf5=67929138-8dcc-404c-af1e-3e8088d0b816
```



```
"C:\Users\kareem\Desktop\NETWORK PART2\PROJ\venv\Scripts\python.exe" "C:\Users\kareem\Desktop\NETWORK PART2\PROJ\pp.py"
Run: pp.py
('127.0.0.1', 55826)
GET /palestine.png HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="120"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36 Edg/120.0.0.0
sec-ch-ua-platform: "Windows"
Accept: image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
dnt: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: image
Referer: http://localhost:9966/en
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-b01fafaf5=67929138-8dcc-404c-af1e-3e8088d0b816
```

```

('127.0.0.1', 55827)
GET /kareem.jpg HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A_Brand";v="8", "Chromium";v="120", "Microsoft_Edge";v="120"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36 Edg/120.0.0.0
sec-ch-ua-platform: "Windows"
Accept: image/webp,image/apng,image/svg+xml,image/*,*;q=0.8
dnt: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: image
Referer: http://localhost:9966/en
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-b01fafaf5=67929138-8dcc-404c-af1e-3e8088d0b816

('127.0.0.1', 55828)
GET /ghandy.jpg HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A_Brand";v="8", "Chromium";v="120", "Microsoft_Edge";v="120"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36 Edg/120.0.0.0
sec-ch-ua-platform: "Windows"
Accept: image/webp,image/apng,image/svg+xml,image/*,*;q=0.8
dnt: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: image
Referer: http://localhost:9966/en
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-b01fafaf5=67929138-8dcc-404c-af1e-3e8088d0b816

```



```

('127.0.0.1', 55827)
GET /quais.jpg HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A_Brand";v="8", "Chromium";v="120", "Microsoft_Edge";v="120"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36 Edg/120.0.0.0
sec-ch-ua-platform: "Windows"
Accept: image/webp,image/apng,image/svg+xml,image/*,*;q=0.8
dnt: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: image
Referer: http://localhost:9966/en
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-b01fafaf5=67929138-8dcc-404c-af1e-3e8088d0b816

('127.0.0.1', 55828)
GET /qais.jpg HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A_Brand";v="8", "Chromium";v="120", "Microsoft_Edge";v="120"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36 Edg/120.0.0.0
sec-ch-ua-platform: "Windows"
Accept: image/webp,image/apng,image/svg+xml,image/*,*;q=0.8
dnt: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: image
Referer: http://localhost:9966/en
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-b01fafaf5=67929138-8dcc-404c-af1e-3e8088d0b816

```

Figure 11: request message for / or /en or /index.html or /main_en.html

note that the html request will request other objects as pngs and jpgs and css file , “keep alive” is shown I request which mean it is a persistent connection and all other obejcts will have on RRT request time.

Main_en on browser:

When requesting any main_en.html or any equivalent request like / this html page will appear on the browser

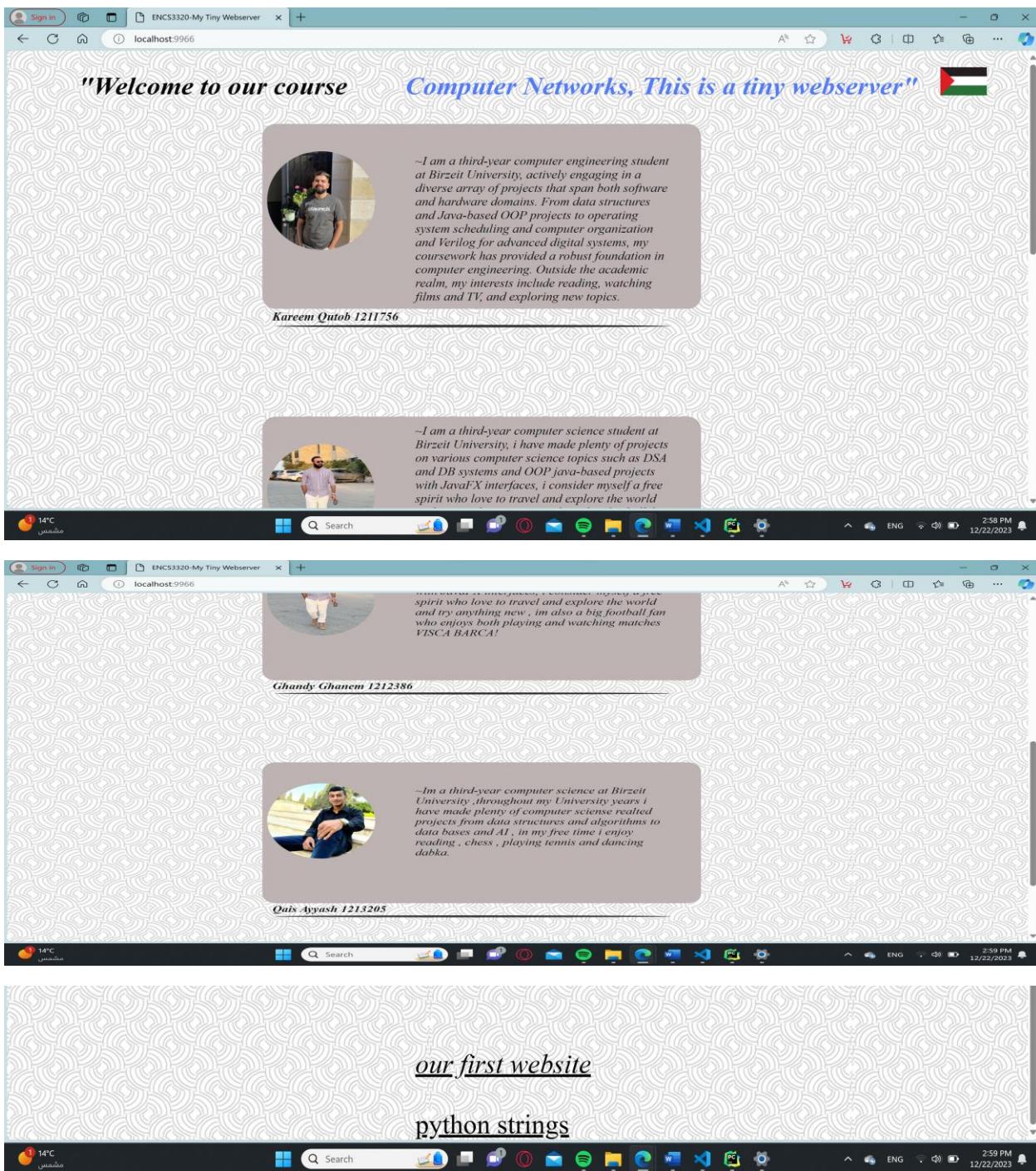
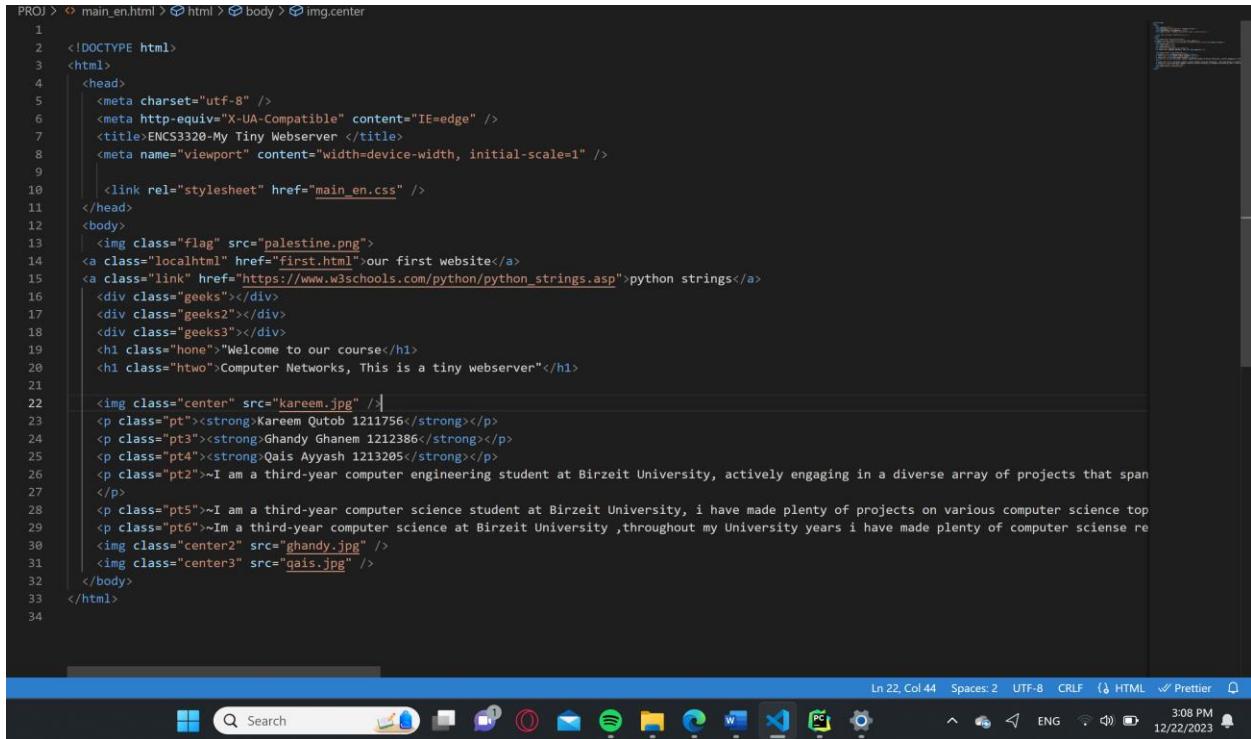


Figure 12: main_en.html on browser

This web page check all the boxes requested for page design using html and css for styling , with brief information about us.

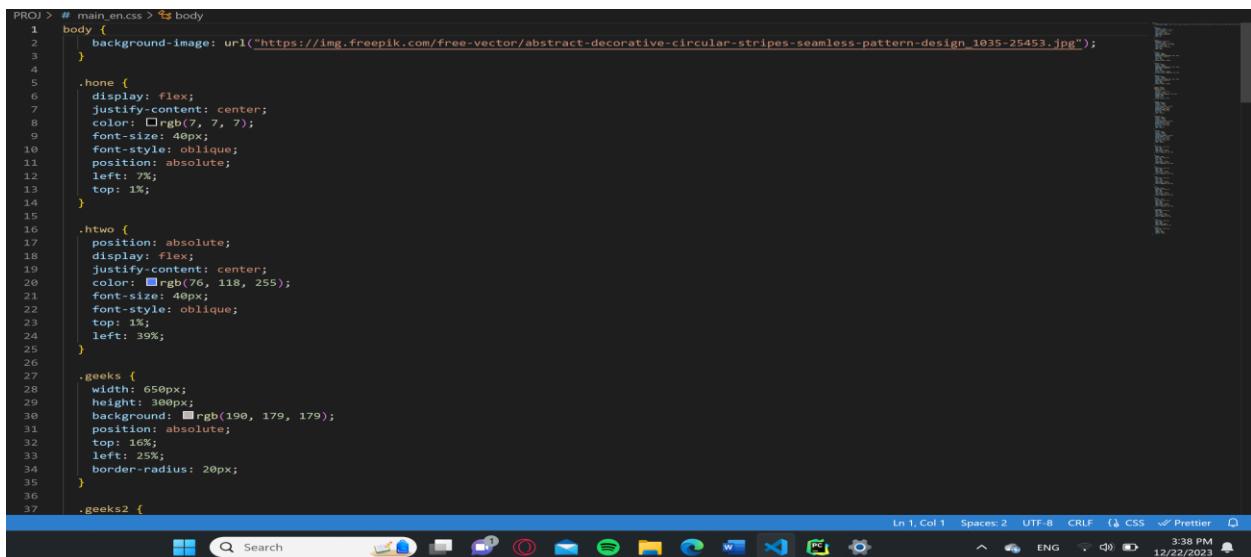
main_en html code:



```
PROJ > main_en.html > html > body > img.center
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <meta charset="utf-8" />
5      <meta http-equiv="X-UA-Compatible" content="IE=edge" />
6      <title>ENCS3320-My Tiny Webserver</title>
7      <meta name="viewport" content="width=device-width, initial-scale=1" />
8
9
10     <link rel="stylesheet" href="main_en.css" />
11   </head>
12   <body>
13     
14     <a class="localhtml" href="first.html">our first website</a>
15     <a class="link" href="https://www.w3schools.com/python/python_strings.asp">python strings</a>
16     <div class="geeks"></div>
17     <div class="geeks2"></div>
18     <div class="geeks3"></div>
19     <h1 class="hone">"Welcome to our course</h1>
20     <h1 class="htwo">Computer Networks, This is a tiny webserver</h1>
21
22     
23     <p class="pt"><strong>Kareem Qutob 1211756</strong></p>
24     <p class="pt3"><strong>Ghandy Ghanem 1212386</strong></p>
25     <p class="pt4"><strong>Qais Ayash 1213205</strong></p>
26     <p class="pt2">~I am a third-year computer engineering student at Birzeit University, actively engaging in a diverse array of projects that span</p>
27     <p class="pt5">~I am a third-year computer science student at Birzeit University, i have made plenty of projects on various computer science top</p>
28     <p class="pt6">~Im a third-year computer science at Birzeit University ,throughout my University years i have made plenty of computer science re</p>
29     
30     
31   </body>
32 </html>
33
34
```

Figure 13:main_en html code

Main_en CSS code:



```
PROJ > main_en.css > body
1  body {
2    background-image: url("https://img.freepik.com/free-vector/abstract-decorative-circular-stripes-seamless-pattern-design_1035-25453.jpg");
3  }
4
5  .hone {
6    display: flex;
7    justify-content: center;
8    color: #rgb(7, 7, 7);
9    font-size: 40px;
10   font-style: oblique;
11   position: absolute;
12   left: 7%;
13   top: 15%;
14 }
15
16 .htwo {
17   position: absolute;
18   display: flex;
19   justify-content: center;
20   color: #rgb(76, 118, 255);
21   font-size: 40px;
22   font-style: oblique;
23   top: 15%;
24   left: 39%;
25 }
26
27 .geeks {
28   width: 650px;
29   height: 300px;
30   background: #rgb(190, 179, 179);
31   position: absolute;
32   top: 16%;
33   left: 25%;
34   border-radius: 20px;
35 }
36
37 .geeks2 {
```

```
PROJ > # main_en.css > body
36 .geeks1 {
37     width: 650px;
38     height: 300px;
39     background: #rgb(190, 179, 179);
40     position: absolute;
41     top: 80px;
42     left: 25px;
43     border-radius: 20px;
44     border-color: #rgb(145, 40, 40);
45 }
46
47 .geeks2 {
48     width: 650px;
49     height: 300px;
50     background: #rgb(190, 179, 179);
51     position: absolute;
52     top: 80px;
53     left: 25px;
54     border-radius: 20px;
55 }
56
57 .geeks::after,
58 .geeks2::after,
59 .geeks3::after {
60     content: "";
61     position: absolute;
62     bottom: -30px;
63     background: #rgb(43, 42, 42);
64     width: 90%;
65     height: 4px;
66     left: 5px;
67     border-radius: 50%;
68 }
69
70 .center {
71     width: 160px;
72 }
```

Ln 1, Col 1 Spaces: 2 UTF-8 CRLF CSS Prettier 3:38 PM 12/22/2023

```
PROJ > # main_en.css > .center2
72 .width: 160px;
73 .height: 160px;
74 .display: flex;
75 .justify-content: center;
76 .align-items: center;
77 .position: absolute;
78 .border-radius: 50px;
79 .position: absolute;
80 .top: 22px;
81 .left: 25.5px;
82 }
83
84 .center2 {
85     width: 160px;
86     height: 160px;
87     display: flex;
88     justify-content: center;
89     align-items: center;
90     position: absolute;
91     border-radius: 50px;
92     position: absolute;
93     top: 86px;
94     left: 25.5px;
95 }
96
97 .center3 {
98     width: 160px;
99     height: 160px;
100    display: flex;
101   justify-content: center;
102   align-items: center;
103   position: absolute;
104   border-radius: 50px;
105   position: absolute;
106   top: 150px;
107   left: 25.5px;
108 }
```

Ln 89, Col 25 Spaces: 2 UTF-8 CRLF CSS Prettier 3:41 PM 12/22/2023

```
110 .pt {
111     position: absolute;
112     top: 54px;
113     left: 26px;
114     font-size: larger;
115     font-style: italic;
116     color: #rgb(20, 23, 23);
117 }
118
119 .pt2 {
120     width: 10cm;
121     position: absolute;
122     top: 20px;
123     left: 46px;
124     font-size: larger;
125     font-style: italic;
126     color: #rgb(20, 23, 23);
127 }
128
129 .pt3 {
130     position: absolute;
131     top: 118px;
132     left: 26px;
133     font-size: larger;
134     font-style: italic;
135     color: #rgb(20, 23, 23);
136 }
137
138 .pt4 {
139     position: absolute;
140     top: 182px;
141     left: 26px;
142     font-size: larger;
143     font-style: italic;
144     color: #rgb(20, 23, 23);
145 }
```

Ln 132, Col 15 Spaces: 2 UTF-8 CRLF CSS Prettier 3:41 PM 12/22/2023

Figure 14: main_en.css code

Clicking “our first website” link:

As asked in the main_en design there are a link that will redirect to local html page of our choice



Figure 15: local html file link

When clicked thid page will appear:

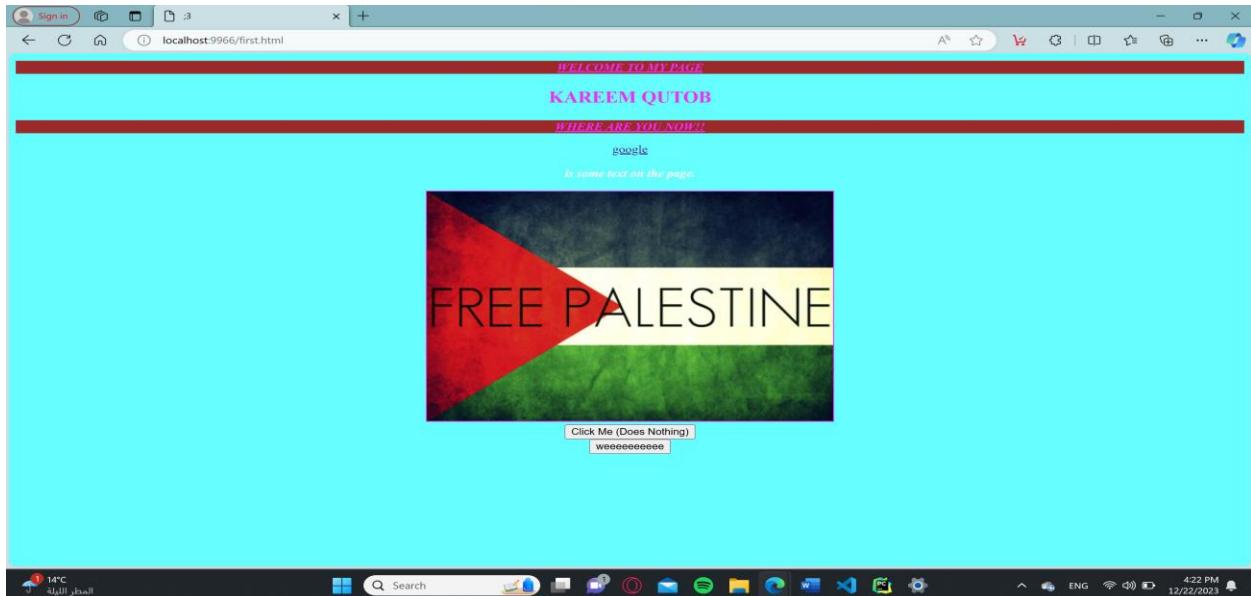


Figure 16:local html page

This our first attempt of HTML/CSS coding so it was only an attempt of trying different objects and styles

Local Html page request on command line:

```
↑ ('127.0.0.1', 58375)
↓ GET /first.html HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A_Brand";v="8", "Chromium";v="120", "Microsoft_Edge";v="120"
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/120.0.0.0 Safari/537.36 Edg/120.0.0.0
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
dnt: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://localhost:9966/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-bo1faFa5=67929158-8dcc-404c-af1e-3e8088d0b816

('127.0.0.1', 58377)
GET /s.css HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A_Brand";v="8", "Chromium";v="120", "Microsoft_Edge";v="120"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36 Edg/120.0.0.0
```

```

dnt: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: style
Referer: http://localhost:9966/first.html
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-b01fafa5=67929138-8dcc-404c-a1fe-3e8088d0b816

('127.0.0.1', 58378)
GET /pal.jpeg HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A_Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="120"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36 Edg/120.0.0.0
sec-ch-ua-platform: "Windows"
Accept: image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
dnt: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: image
Referer: http://localhost:9966/first.html
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-b01fafa5=67929138-8dcc-404c-a1fe-3e8088d0b816

```

Control Run Python Packages TODO Python Console Problems Terminal Services

133.25 CRLF UTF-8 4 spaces Python 3.9 (PROJ) 4:28 PM ENG 12/22/2023

Figure 17:local html file request

the request of html file will include the jpg and css code attached to the website

Local html file code:

```

PROJ > ⌂ firsthtml > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <title>3</title>
5      <meta charset="utf-8">
6      <link rel="stylesheet" href="s.css">
7
8  </head>
9
10 <body>
11     <h1 class="a n">Welcome to My Page</h1>
12     <h2 class="a"> KAREEM QUTOB</h2>
13     <h2 class="n a">where are you now!!</h3>
14     <a href="https://www.google.com/"> google</a>
15     <br>
16
17     <p><b><em>is some text on the page.</em></b></p>
18
19     
20     <br>
21     <button type="button">Click Me (Does Nothing)</button>
22     <br>
23     <button type="button">weeeeeeeeee</button>
24
25 </body>
26 </html>
27

```

Figure 18: local html file code

Local html file css code:

```
PROJ > # s.css > img
  1 <img{
  2
  3     border: solid;
  4     cursor: pointer;
  5     box-shadow: inset;
  6   }
  7 <p{
  8     color: ■rgb(255, 255, 255);
  9   }
10 <body{
11     color: ■magenta;
12     text-align: center;
13     background-color: ■aqua;
14   }
15 <.n{
16     background-color: ■brown;
17     font-style: italic;
18     text-decoration: underline;
19     text-transform: uppercase;
20     font-weight: bold;
21     font-size: medium;
22 }
```

Figure 19: local html file css code

Clicking python string link:

As asked in the main_en design there are a link that will redirect to w3schools page about python strings



Figure 20:python strings link

A screenshot of a web browser displaying the 'Python Strings' tutorial from w3schools. The URL in the address bar is https://www.w3schools.com/python/python_strings.asp. The page has a sidebar on the left with various Python topics, and the main content area is titled 'Python Strings'. It contains sections for 'Strings', 'Example', and 'Try it Yourself». The right side of the page includes a sidebar with links like 'Jobs', 'Spaces', 'Get Certified', 'Sign Up', and 'Log in'. The bottom of the screen shows a Windows taskbar with several pinned icons.

Figure 21: w3school python strings tutorial

Main_ar request:

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

Main_ar request on command line:

```
"C:\Users\kareem\Desktop\NETWORK PART2\PROJ\venv\Scripts\python.exe" "C:\Users\kareem\Desktop\NETWORK PART2\PROJ\pp.py"
The server is ready to receive
('127.0.0.1', 61898)
GET /ar HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="120"
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/120.0.0.0 Safari/537.36 Edg/120.0.0.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
dnt: 1
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
Cookie: Pycharm-b01fafaf5=67929138-8dcc-404c-afile-3e8088d0b816

('127.0.0.1', 61930)
GET /main_ar.css HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="120"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36 Edg/120.0.0.0
sec-ch-ua-platform: "Windows"
n Control Run Python Packages TODO Python Console Problems Terminal Services
118:1 CRLF UTF-8 4 spaces Python 3.9 (PROJ)
4:41 PM 12/22/2023
```

```
pp x
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36 Edg/120.0.0.0
sec-ch-ua-platform: "Windows"
Accept: text/css,*/*;q=0.1
dnt: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: style
Referer: http://localhost:9966/ar
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-b01fafaf5=67929138-8dcc-404c-afile-3e8088d0b816

('127.0.0.1', 61931)
GET /palestine.png HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="120"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36 Edg/120.0.0.0
sec-ch-ua-platform: "Windows"
Accept: image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
dnt: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: image
Referer: http://localhost:9966/ar
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Version Control Run Python Packages TODO Python Console Problems Terminal Services
118:1 CRLF UTF-8 4 spaces Python 3.9 (PROJ)
4:41 PM 12/22/2023
```

```

pp >
↑ ('127.0.0.1', 61932)
↓ GET /kareem.jpg HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="120"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36 Edg/120.0.0.0
sec-ch-ua-platform: "Windows"
Accept: image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
dnt: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: image
Referer: http://localhost:9966/ar
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-b01fafaf5=67929138-8dcc-404c-af1e-3e8088d0b816

('127.0.0.1', 61933)
GET /ghandy.jpg HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="120"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36 Edg/120.0.0.0
sec-ch-ua-platform: "Windows"
Accept: image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
dnt: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: image
Referer: http://localhost:9966/ar
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-b01fafaf5=67929138-8dcc-404c-af1e-3e8088d0b816

dnt: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: image
Referer: http://localhost:9966/ar
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-b01fafaf5=67929138-8dcc-404c-af1e-3e8088d0b816

('127.0.0.1', 61934)
GET /qais.jpg HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="120"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36 Edg/120.0.0.0
sec-ch-ua-platform: "Windows"
Accept: image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
dnt: 1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: image
Referer: http://localhost:9966/ar
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-b01fafaf5=67929138-8dcc-404c-af1e-3e8088d0b816

```

Figure 22:/ar request message

Similar to main_en request, a persistent tcp connection that request an html file with objects like pictures and css code

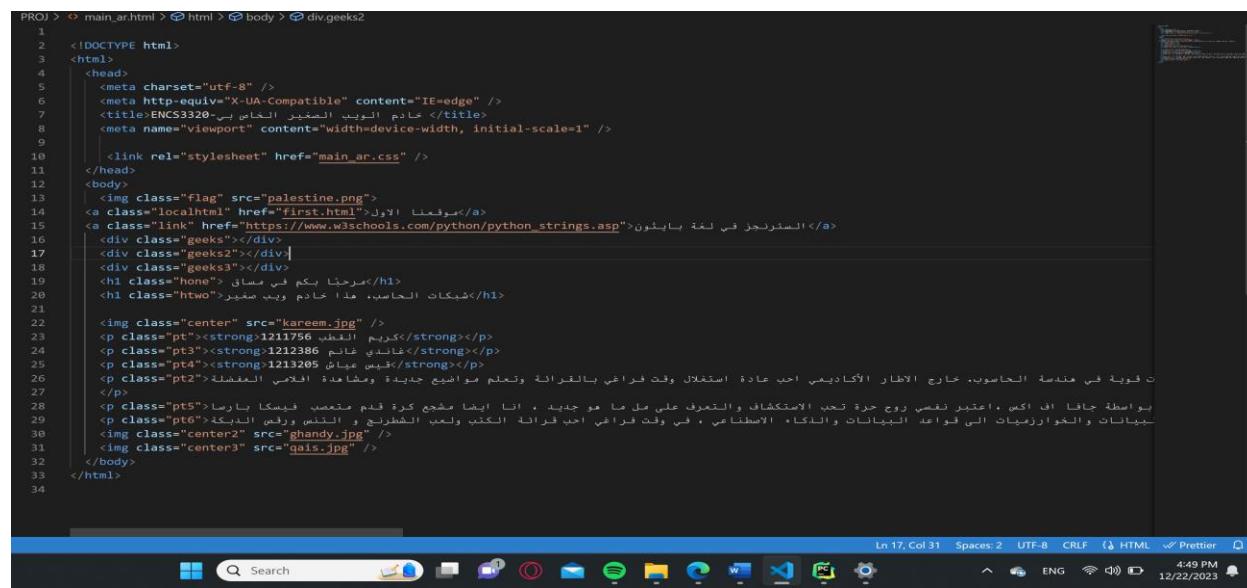
Main_ar on browser:



Figure 23:main_ar on browser

An Arabic version of main_en website

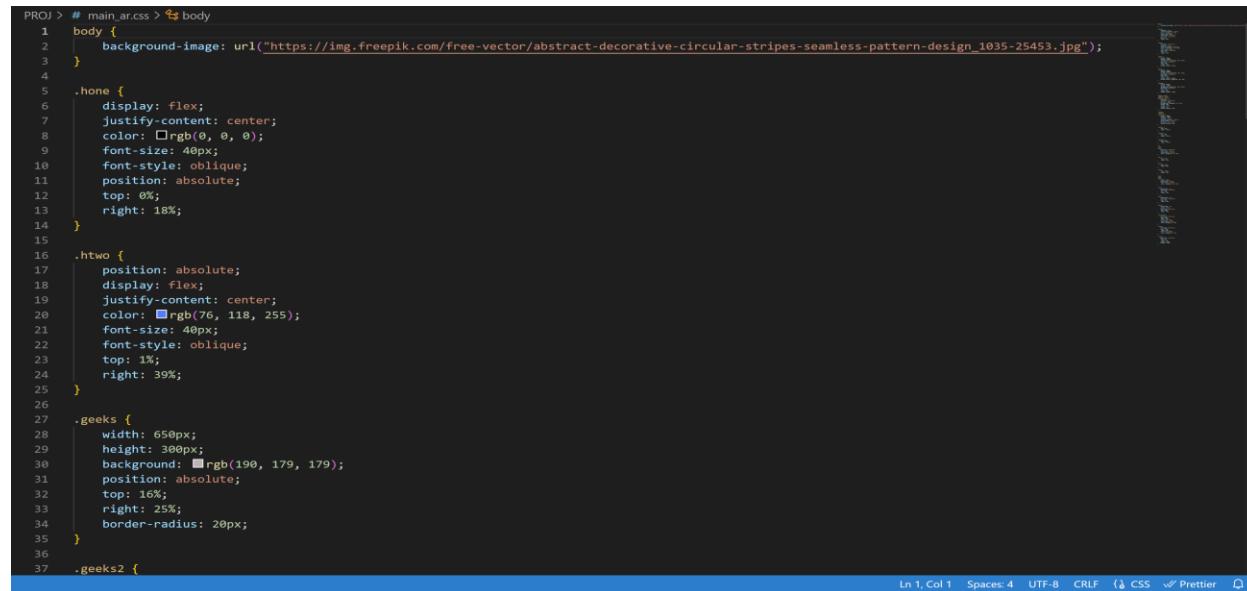
Main_ar html code:



```
PROJ > main_ar.html > html > body > div.geeks2
1
2  <!DOCTYPE html>
3  <html>
4    <head>
5      <meta charset="utf-8" />
6      <meta http-equiv="X-UA-Compatible" content="IE=edge" />
7      <title>ENCS3320- خادم الموبيل المخادر بـ</title>
8      <meta name="viewport" content="width=device-width, initial-scale=1" />
9
10     <link rel="stylesheet" href="main_ar.css" />
11   </head>
12   <body>
13     
14     <a class="localhtml" href="first.html">موقعنا الالكتروني</a>
15     <a class="link" href="https://www.w3schools.com/python/python_strings.asp">المسترجع في لغة بـPython</a>
16     <div class="geeks"></div>
17     <div class="geeks2"></div>
18     <div class="geeks3"></div>
19     <h1 class="home">مرحبا بك في مساق</h1>
20     <h1 class="htwo">ذيليات الحاسوب، هنا خادم ويب مفتوحة</h1>
21
22     
23     <p class="pt1"><strong>1211756</strong><br/>عزمي الخطيب</p>
24     <p class="pt3"><strong>1212386</strong>-عابد غاظم</p>
25     <p class="pt4"><strong>1213205</strong>-كيس مهاش</p>
26     <p class="pt2"></p>
27
28     <p class="pt5"> بواسطة جافا اف اكس ، اعتبر نفس روح حرة تحب الاستكشاف والتعرف على كل ما هو جديد ، انا ايضاً منجع كرة قدم متعمق كريستيان برادا</p>
29     <p class="pt6">بيانات والخوارزميات الى قواعد البيانات والذكاء الاصطناعي ، في وقت فراغي احب القراءة والتكتب ولعب الشطرنج وتعلم البرمجة</p>
30     
31     
32
33   </body>
34 </html>
```

Figure 24:main_ar html code

Main_ar css code:



```
PROJ > main_ar.css > body
1  body {
2    background-image: url("https://img.freepik.com/free-vector/abstract-decorative-circular-stripes-seamless-pattern-design_1035-25453.jpg");
3  }
4
5 .home {
6   display: flex;
7   justify-content: center;
8   color: #rgb(0, 0, 0);
9   font-size: 40px;
10  font-style: oblique;
11  position: absolute;
12  top: 0%;
13  right: 18%;
14 }
15
16 .htwo {
17  position: absolute;
18  display: flex;
19  justify-content: center;
20  color: #rgb(76, 118, 255);
21  font-size: 40px;
22  font-style: oblique;
23  top: 1%;
24  right: 39%;
25 }
26
27 .geeks {
28  width: 650px;
29  height: 300px;
30  background: #rgb(190, 179, 179);
31  position: absolute;
32  top: 16%;
33  right: 25%;
34  border-radius: 20px;
35 }
36
37 .geeks2 {
```

```
PROJ > # main_ar.css > body
  38   width: 650px;
  39   height: 300px;
  40   background: #rgb(190, 179, 179);
  41   position: absolute;
  42   top: 80%;
  43   right: 25%;
  44   border-radius: 20px;
  45   border-color: #rgb(145, 40, 40);
  46 }
  47
  48 .geeks3 {
  49   width: 650px;
  50   height: 300px;
  51   background: #rgb(190, 179, 179);
  52   position: absolute;
  53   top: 144%;
  54   right: 25%;
  55   border-radius: 20px;
  56 }
  57
  58 .geeks::after,
  59 .geeks2::after,
  60 .geeks3::after {
  61   content: "";
  62   position: absolute;
  63   bottom: -30px;
  64   background: #rgb(43, 42, 42);
  65   width: 90%;
  66   height: 4px;
  67   right: 3%;
  68   border-radius: 50%;
  69 }
  70
  71 .center,
  72 .center2,
  73 .center3 {
  74   width: 160px;
```

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF (CSS ✓ Prettier

```
PROJ > # main_ar.css > body
  74   width: 160px;
  75   height: 160px;
  76   display: flex;
  77   justify-content: center;
  78   align-items: center;
  79   position: absolute;
  80   border-radius: 50%;
  81 }
  82
  83 .center {
  84   top: 22%;
  85   right: 25.5%;
  86 }
  87
  88 .center2 {
  89   top: 86%;
  90   right: 25.5%;
  91 }
  92
  93 .center3 {
  94   top: 150%;
  95   right: 25.5%;
  96 }
  97
  98 .pt,
  99 .pt3,
  100 .pt4 {
  101   position: absolute;
  102   font-size: larger;
  103   font-style: italic;
  104   color: #rgb(20, 23, 23);
  105 }
  106
  107 .pt {
  108   top: 51%;
  109   right: 29%;
```

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF (CSS ✓ Prettier

```
PROJ > # main_ar.css > body
 111
 112 .pt3 {
 113   top: 115%;
 114   right: 29%;
 115 }
 116
 117 .pt4 {
 118   top: 179%;
 119   right: 29%;
 120 }
 121
 122 .pt2,
 123 .pt5,
 124 .pt6 {
 125   width: 10cm;
 126   font-size: larger;
 127   font-style: italic;
 128   color: #rgb(20, 23, 23);
 129 }
 130
 131 .pt2 {
 132   direction: rtl;
 133   position: absolute;
 134   top: 20%;
 135   right: 40%;
 136 }
 137
 138 .pt5 {
 139   direction: rtl;
 140   position: absolute;
 141   top: 84%;
 142   right: 40%;
 143 }
 144
 145 .pt6 {
 146   direction: rtl;
 147   width: 10cm;
```

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF (CSS ✓ Prettier

```

145 .pt6 {
146     direction: rtl;
147     width: 10cm;
148     position: absolute;
149     top: 148%;
150     right: 40%;
151 }
152
153 .localhtml {
154     position: absolute;
155     top: 208%;
156     right: 40%;
157     font-size: 40px;
158     font-style: italic;
159     color: □rgb(0, 0, 0);
160 }
161
162 .link {
163     position: absolute;
164     top: 220%;
165     right: 40%;
166     font-size: 40px;
167     color: □rgb(0, 0, 0);
168 }
169
170 .flag {
171     position: absolute;
172     top: 2%;
173     right: 4%;
174     width: 70px;
175     height: 70px;
176 }
177

```

Figure 25:main_ar.css code

.html request:

- 3- 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.

.html request on command line:

```

"C:\Users\kareem\Desktop\NETWORK PART2\PROJ\venv\Scripts\python.exe" "C:\Users\kareem\Desktop\NETWORK PART2\PROJ\pp.py"
The server is ready to receive
('127.0.0.1', 6519)
GET /first.html HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="120"
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/120.0.0.0 Safari/537.36 Edg/120.0.0.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
dnt: 1
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
Cookie: Pycharm-b01fa5a5=67929138-8dcc-404c-af1e-3e8088d0b816

```

Figure 26:.html request

For convenience I reused an html file from previous parts

.Html request on browser



Figure 27: .html request on browser

Our code will return the html file for any request with \filename.html request

.CSS request:

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

.css request on command line:

```
"C:\Users\kareem\Desktop\NETWORK PART2\PROJ\venv\Scripts\python.exe" "C:\Users\kareem\Desktop\NETWORK PART2\PROJ\pp.py"
The server is ready to receive
HTTP/1.1 65196
GET /s.css HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A_Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="120"
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/120.0.0.0 Safari/537.36 Edg/120.0.0.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/*,*;q=0.8,application/signed-exchange;v=b3;q=0.7
dnt: 1
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
Cookie: Pycharm-b01fafaf5=67929138-8dcc-404c-afile-3e8088d0b816
```

Figure 28: .css request message

.css request on browser:

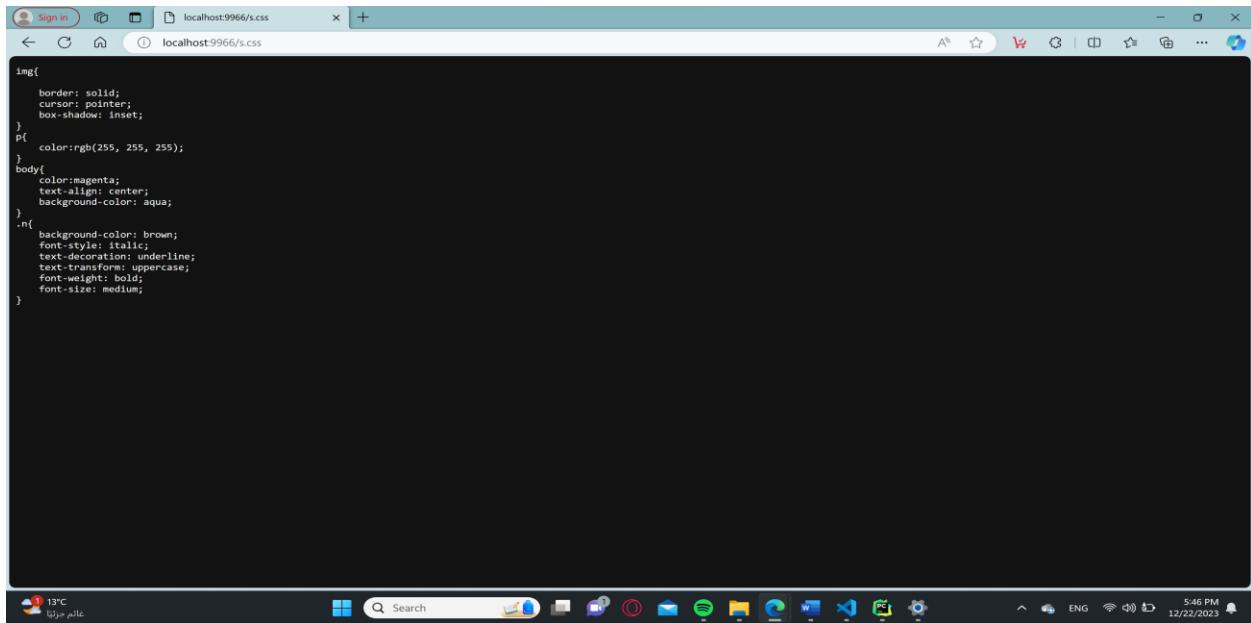


Figure 29:.css request on browser

PNG request:

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

PNG request on command line:

```
"C:\Users\kareem\Desktop\NETWORK PART2\PROJ\venv\Scripts\python.exe" "C:\Users\kareem\Desktop\NETWORK PART2\PROJ\pp.py"
The server is ready to receive
('127.0.0.1', 65217)
GET /palestine.png HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="120"
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/120.0.0.0 Safari/537.36 Edg/120.0.0.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
dnt: 1
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-b01faf5=67929138-8dcc-404c-af1e-3e8088d0b816
Version Control Run Python Packages TODO Python Console Problems Terminal Services
23:1 CRLF UTF-8 4 spaces Python 3.9 (PROJ) 549 PM 12/22/2023
```

Figure 30: png request message

.PNG request on browser:

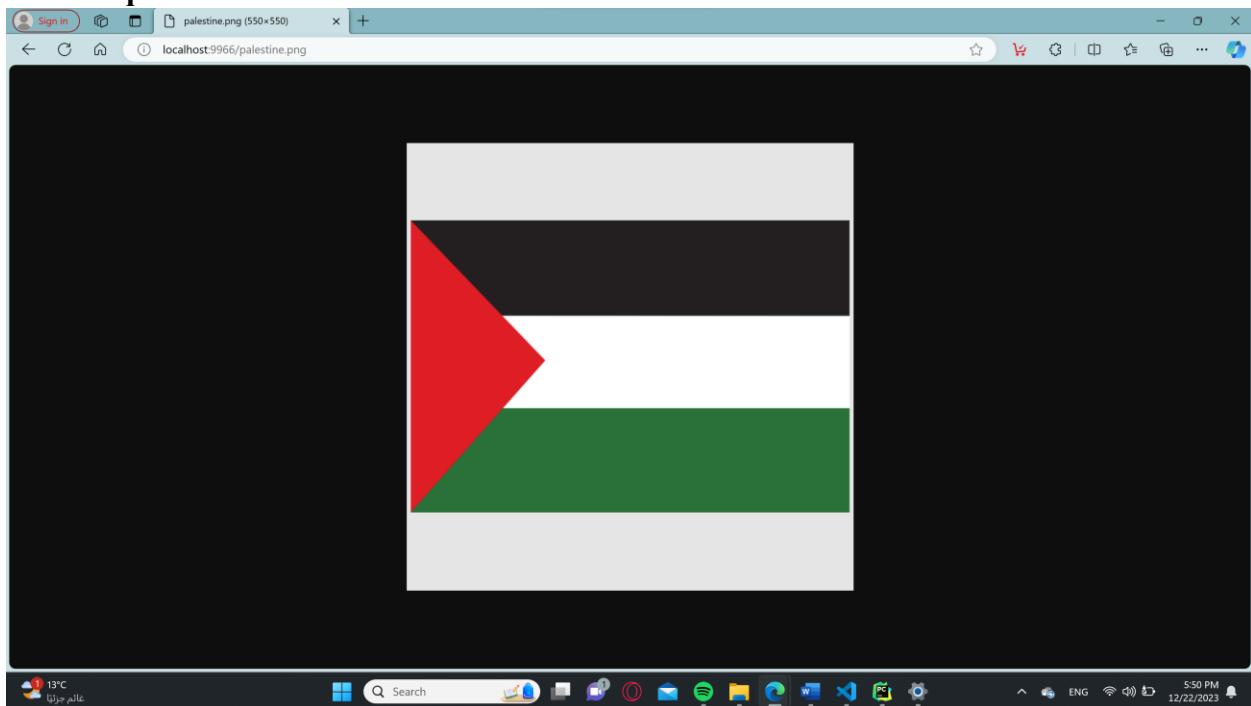


Figure 31:>PNG on browser

.JPG request:

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

JPG request on command line:

```
"C:\Users\kareem\Desktop\NETWORK PART2\PROJ\venv\Scripts\python.exe" "C:\Users\kareem\Desktop\NETWORK PART2\PROJ\pp.py"
The server is ready to receive
('127.0.0.1', 65246)
GET /dog.jpg HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="128"
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/120.0.0.0 Safari/537.36 Edg/120.0.0.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
dnt: 1
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
Cookie: Pycharm-b01faf5=67929138-8dcc-404c-af1e-3e8088d0b816
23:1 CRLF UTF-8 4 spaces Python 3.9 (PROJ) 5:53 PM ENG 12/22/2023
```

The terminal window shows a Python script named 'pp.py' running on port 9966. It receives a GET request for '/dog.jpg' from a Chromium-based browser on '127.0.0.1'. The browser's user agent string is visible, along with other standard headers like 'Content-Type' and 'Accept'. The terminal also shows the Python version and encoding settings.

Figure 32:JPG request message

.JPG on browser:

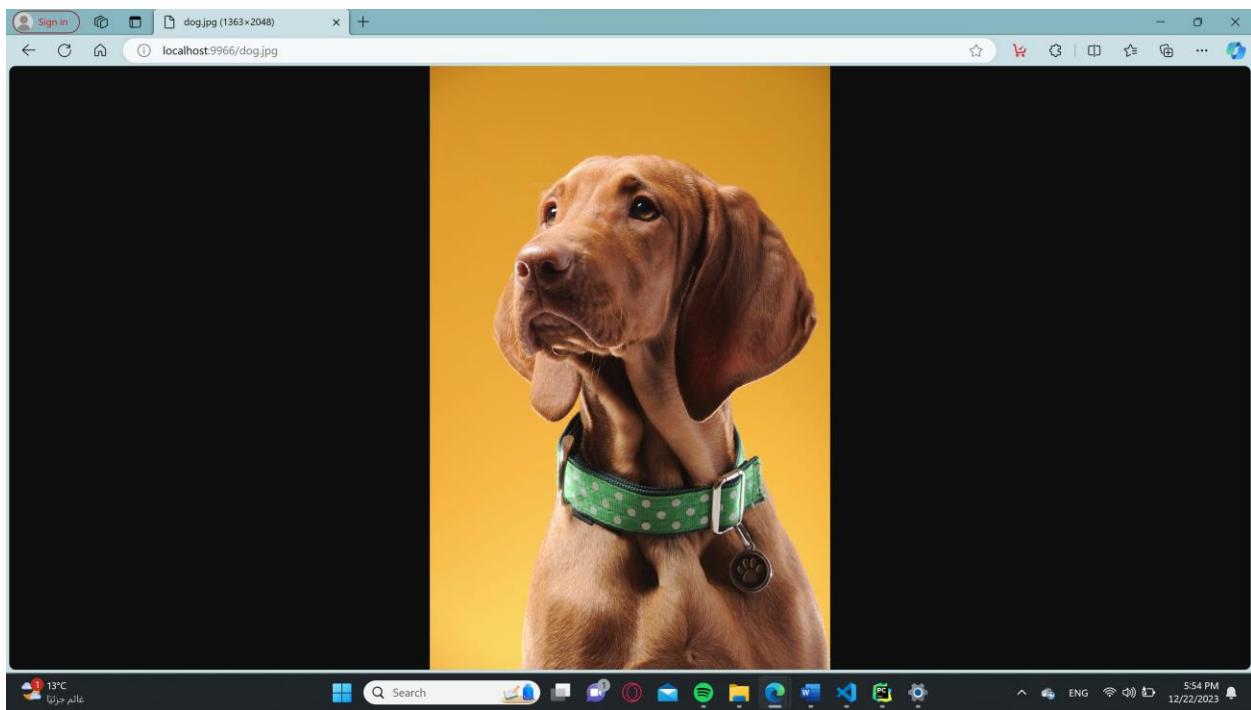


Figure 33: JPG on browser

Page redirecting:

Use the status code **307 Temporary Redirect** to redirect the following

- If the request is **/cr** then redirect to cornell.edu website
- If the request is **/so** then redirect to stackoverflow.com website
- If the request is **/rt** then redirect to [ritaj](http://ritaj.com) website

/cr on command line:

```
"C:\Users\kareem\Desktop\NETWORK PART2\PROJ\venv\Scripts\python.exe" "C:\Users\kareem\Desktop\NETWORK PART2\PROJ\pp.py"
The server is ready to receive
('127.0.0.1', 65269)
GET /cr HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A_Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="120"
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/120.0.0.0 Safari/537.36 Edg/120.0.0.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
dnt: 1
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
Cookie: Pycharm-b01fafaf5=67929138-8dcc-404c-aef1-3e8088d0b816
Version Control Run Python Packages TODO Python Console Problems Terminal Services
23:1 CRLF UTF-8 4 spaces Python 3.9 (PROJ) 5:58 PM
12/22/2023
```

Figure 34: /cr on command line

/cr on browser:

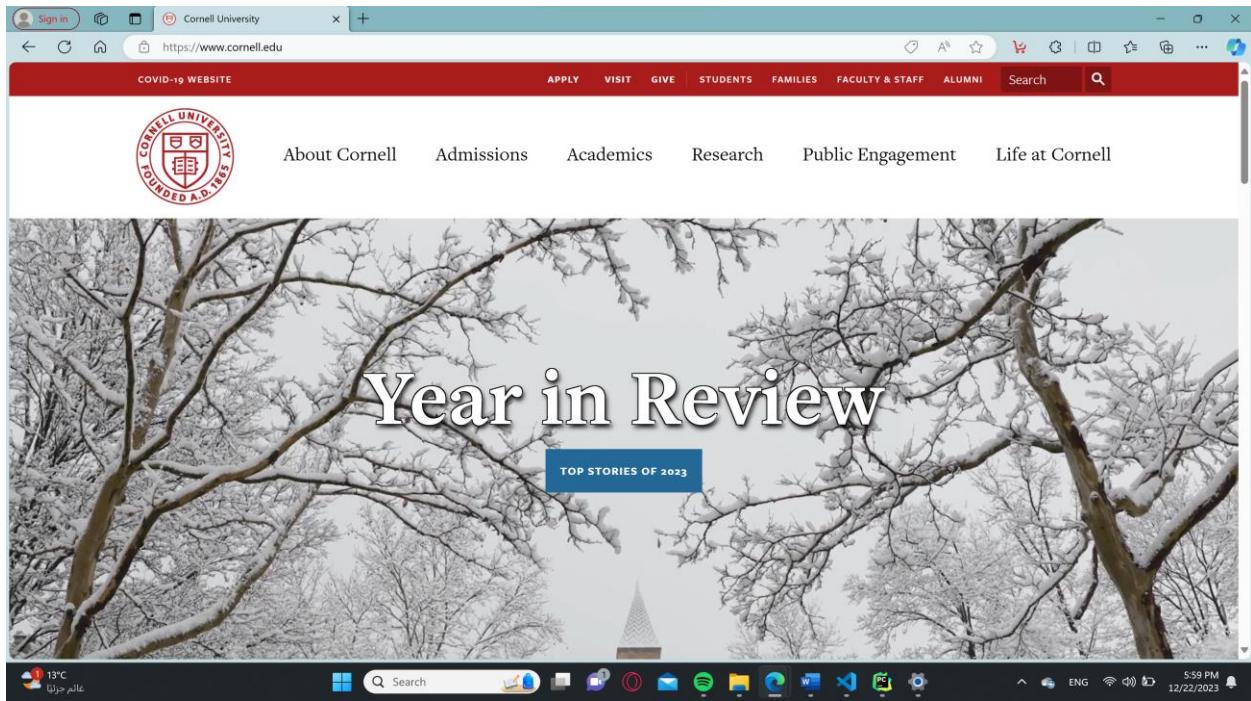


Figure 35:/cr on browser

It will redirect to www.cornell.edu

/rt on command line:

A screenshot of a terminal window titled "pp" showing an incoming HTTP request. The command used was "python -m http.server". The request details are as follows:

```
C:\Users\kareem\Desktop\NETWORK PART2\PROJ\venv\Scripts\python.exe "C:\Users\kareem\Desktop\NETWORK PART2\PROJ\pp.py"
The server is ready to receive
(127.0.0.1, 65318)
GET /rt HTTP/1.1
Host: localhost:9966
Connection: keep-alive
sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="120"
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/120.0.0.0 Safari/537.36 Edg/120.0.0.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
dnt: 1
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
Cookie: Pycharm-b01fafa5=67929158-8dcc-404c-a1f1-3e8088d0b816
```

The terminal also shows the Python environment and system status at the bottom.

Figure 36: /rt request message

/rt on browser :

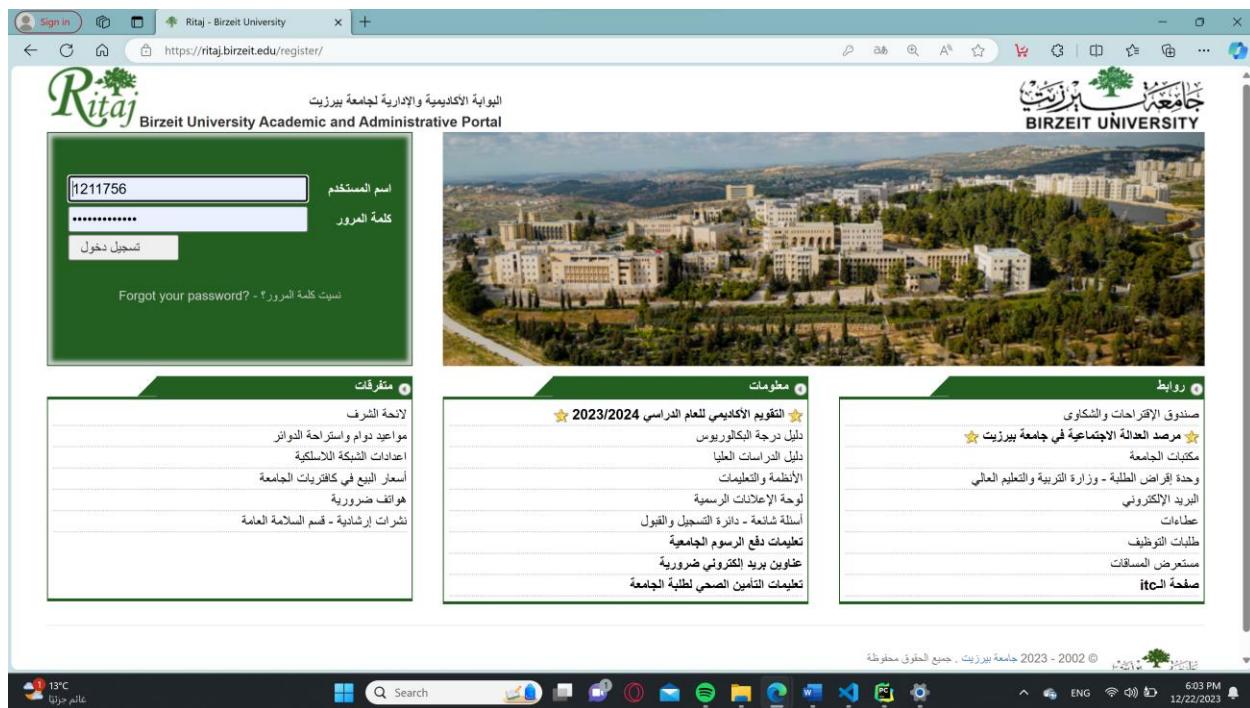


Figure 37:/rt on browser

/so on command line:

```
('127.0.0.1', 65347)
GET /so HTTP/1.1
Host: localhost:9966
Connection: keep-alive
Cache-Control: max-age=0
sec-ch-ua: "Not_A Brand";v="8", "Chromium";v="120", "Microsoft Edge";v="120"
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/120.0.0.0 Safari/537.36 Edg/120.0.0.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/*,*q=0.8,application/signed-exchange;v=b3;q=0.7
dnt: 1
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
Cookie: Pycharm-b01fafaf5=67929138-8dcc-404c-af1e-3e8088d0b816

File Control Run Python Packages TODO Python Console Problems Terminal Services
51:1 CRLF UTF-8 4 spaces Python 3.9 (PROJ)
6:04 PM 12/22/2023
```

Figure 38:/so message request

/so on browser:

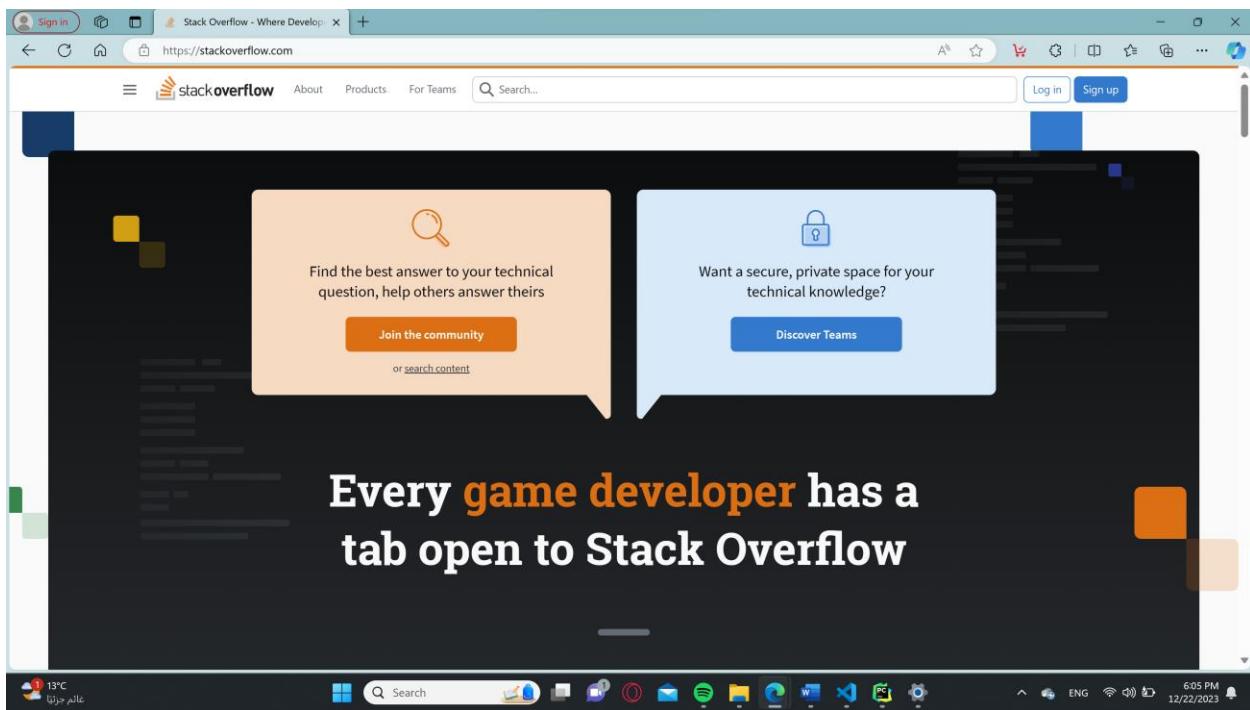


Figure 39:/so on browser

ERROR webpage:

If the request is wrong or the file doesn't exist the server should return a simple HTML webpage that contains (Content-Type: text/html)

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

Error on command line:

A screenshot of a terminal window titled 'ipy' showing an error message. The message is: "GET /rrrr HTTP/1.1", "Host: localhost:9966", "Connection: keep-alive", "sec-ch-ua: \"Not_A Brand\";v=\"8\", \"Chromium\";v=\"120\", \"Microsoft Edge\";v=\"120\"", "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/120.0.0.0 Safari/537.36 Edg/120.0.0.0", "Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7", "dnt: 1", "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", "Cookie: Pycharm-b01fafa5=67929138-8dcc-404c-af1e-3e8088d0b816". The terminal window also shows a Python environment with icons for Version Control, Run, Python Packages, TODO, Python Console, Problems, Terminal, Services, and a status bar indicating 108:35 CRLF UTF-8 4 spaces Python 3.9 (PROJ).

Figure 40: error on command line

Error page on browser:

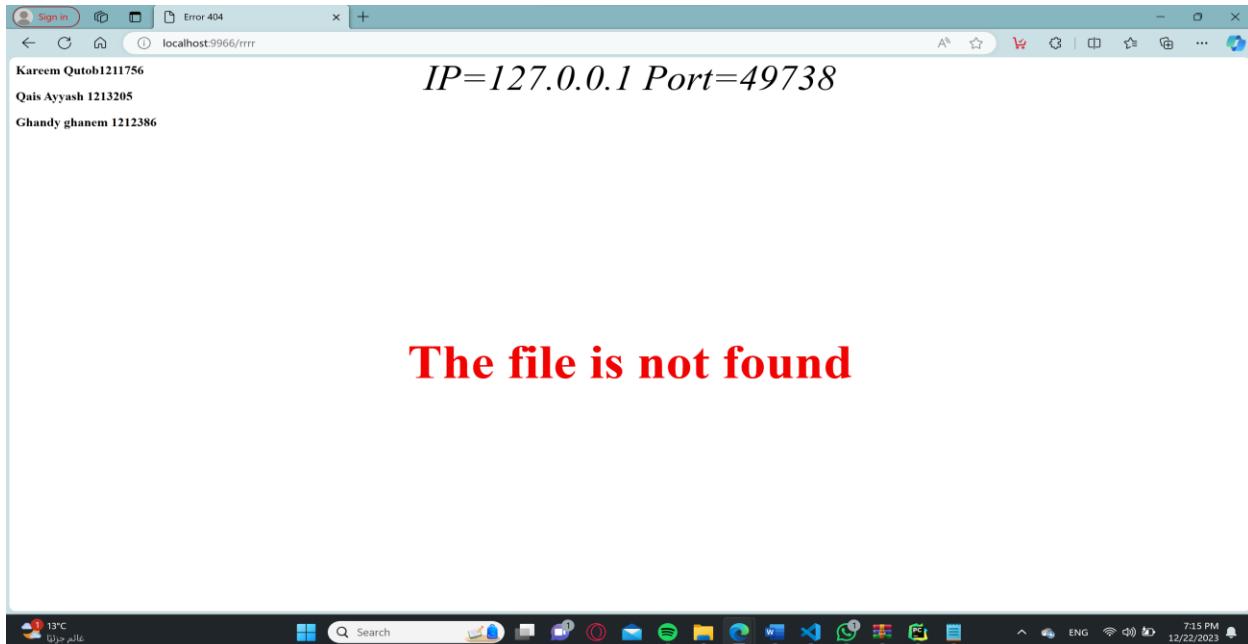


Figure 41:error on browser

In this part the html code was part of python code so we can add the id and port number for every client

Error code segment:

```
else:# this handle error cases
    conn.send("HTTP/1.1 404 Not Found \r\n".encode())
    # content type
    conn.send("Content-Type: text/html; charset=utf-8\r\n".encode())
    conn.send("\r\n".encode())
    # s contain the html page of the error 404
    s = "<html><head><title>Error 404</title><style type=\"text/css\">           h1{           font-size:4em;           text-align:center;           margin:0;           padding:0;
        ip) + " Port=" + str(port) + "</em></p></body></html>\r\n"
    data = s.encode()
    conn.send(data)
```

Figure 42:error code

The html file test will be stored in variable sentence so we can open different pages based on ip and port number

Test from another device:

From another laptop:

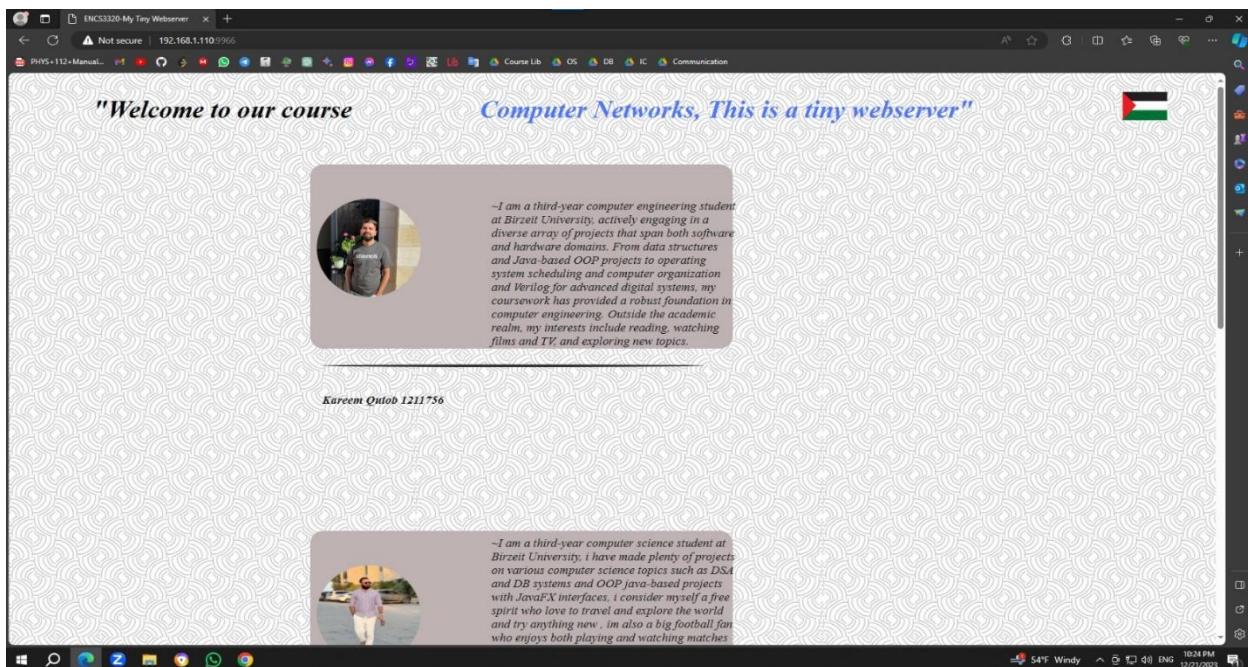


Figure 43: test from another laptop

From a phone:

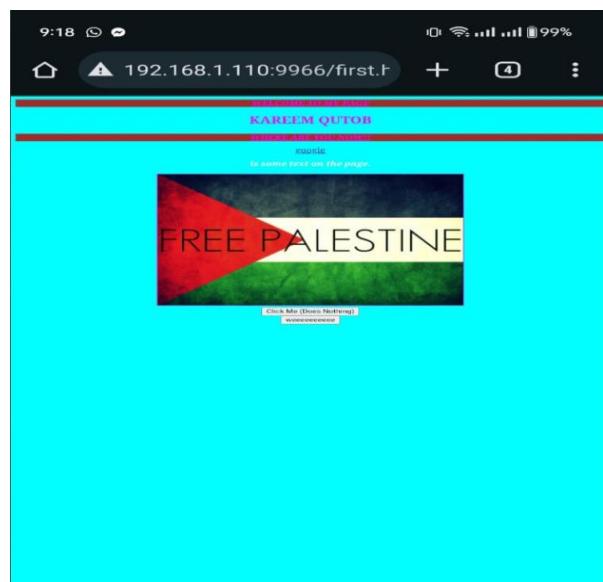


Figure 44: test from a phone

Conclusion:

In conclusion, this comprehensive computer networks and socket programming project requires us to demonstrate our understanding and practical application of network utilities, TCP client-server communication, and web server development. The project, divided into three parts, covers executing network commands, implementing a TCP client-server application with screen-locking functionality based on messages, and creating a web server capable of handling diverse HTTP requests, redirects, and error responses. Emphasizing the use of socket programming and forbidding external libraries. Successfully completing the project involves a holistic grasp of networking concepts, programming proficiency, and rigorous testing to validate the functionality of implemented solutions.