



**BIRZEIT UNIVERSITY**  
Faculty of Engineering and Technology

**Department of Electrical and Computer Engineering**

**Computer Network ENCS3320  
Project Report**

**Prepared by:**

<b>Tala Flaifel</b>	<b>ID: 1201107</b>	<b>Section: 1</b>
<b>Hajar Al Souqi</b>	<b>ID: 1203257</b>	<b>Section: 2</b>
<b>Hazar Michael</b>	<b>ID: 1201838</b>	<b>Section: 2</b>

**Instructors: Dr. Abdalkarim Awad - Dr. Ibrahim Nimer**

**Date: 31/8/2023**

## Table of Contents

Table of figures.....	3
1. Part 1.....	4
1.1. Definitions.....	4
1.1.1 What is ping.....	4
1.1.2 What is tracert.....	4
1.1.3 What is nslookup.....	4
1.1.4 What is telnet.....	4
1.2 Commands.....	5
1.2.1 Ping www.amazon.de.....	5
1.2.2 Tracert www.amaon.de.....	6
1.2.3 nslookup www.amaon.de.....	7
2. Part 2.....	8
2.1 / , /index.html , /main_en.html and /en requests.....	8
2.2 /ar request.....	14
2.2 Text File for Laptop Data.....	15
2.4 Sorting by Name (/SortByName) request.....	16
2.5 Sorting by Price (/SortByPrice) request.....	16
2.6 HTML File Request Handling.....	17
2.7 CSS File Request Handling.....	18
2.8 .PNG Image Request Handling.....	19
2.9 JPG Image Request Handling.....	19
2.10 Temporary Redirects.....	20
2.10.1 /azn request redirect to amazon website.....	20
2.10.2 /so request redirects to stackoverflow.com website.....	20
2.10.3 /bzu request redirect to birzeit website.....	21
2.11 404 Not Found Handling.....	21
2.12 HTTP Request Logging.....	22
Appendices.....	23
Appendix A: main code.....	23
Appendix B: styles.css code.....	26
Appendix C: Arabic page HTTP code.....	28

Appendix D: English page HTTP code.....	29
Appendix E: Error404 HTTP code.....	31

## Table of figures

Figure 1: Ping www.amazon.de	5
Figure 2: Tracert www.amaon.de	6
Figure 3: nslookup www.amaon.de	7
Figure 4: '/' request	8
Figure 5: '/index.html' request	12
Figure 6: '/main_en.html' request	12
Figure 7: '/en.html' request	13
Figure 8: Language-Specific Page Delivery	13
Figure 9: '/ar' request	14
Figure 10: Arabic Version Page Delivery	14
Figure 11 :text file contents	15
Figure 12: Loading Laptop Data from File Code	15
Figure 13:/SortByName request	16
Figure 14:/ SortByPrice request	16
Figure 15: /.html file request	17
Figure 16: HTML File Request Handling Code	17
Figure 17: /.css request	18
Figure 18: /.png request	19
Figure 19: /.jpg request	19
Figure 20: /azn request	20
Figure 21: /so request	20
Figure 22: /bzu request	21
Figure 23: any invalid request output	21
Figure 24: http request example	22

## **1. Part 1**

### **1.1. Definitions**

#### **1.1.1 What is ping**

Ping is a network utility that tests the connection and measures the response time between two computers or devices in an IP network.

#### **1.1.2 What is tracert**

Tracert (short for "traceroute") is a network diagnostic tool used to trace the route that data packets take from your computer to a destination host on the Internet. It shows the sequence of routers and intermediate nodes through which the data passes, helping to identify potential network issues or delays along the way.

#### **1.1.3 What is nslookup**

Nslookup is a command-line tool used to query and retrieve information from Domain Name System (DNS) servers. It's commonly used to find the IP address associated with a domain name or to obtain other DNS-related information.

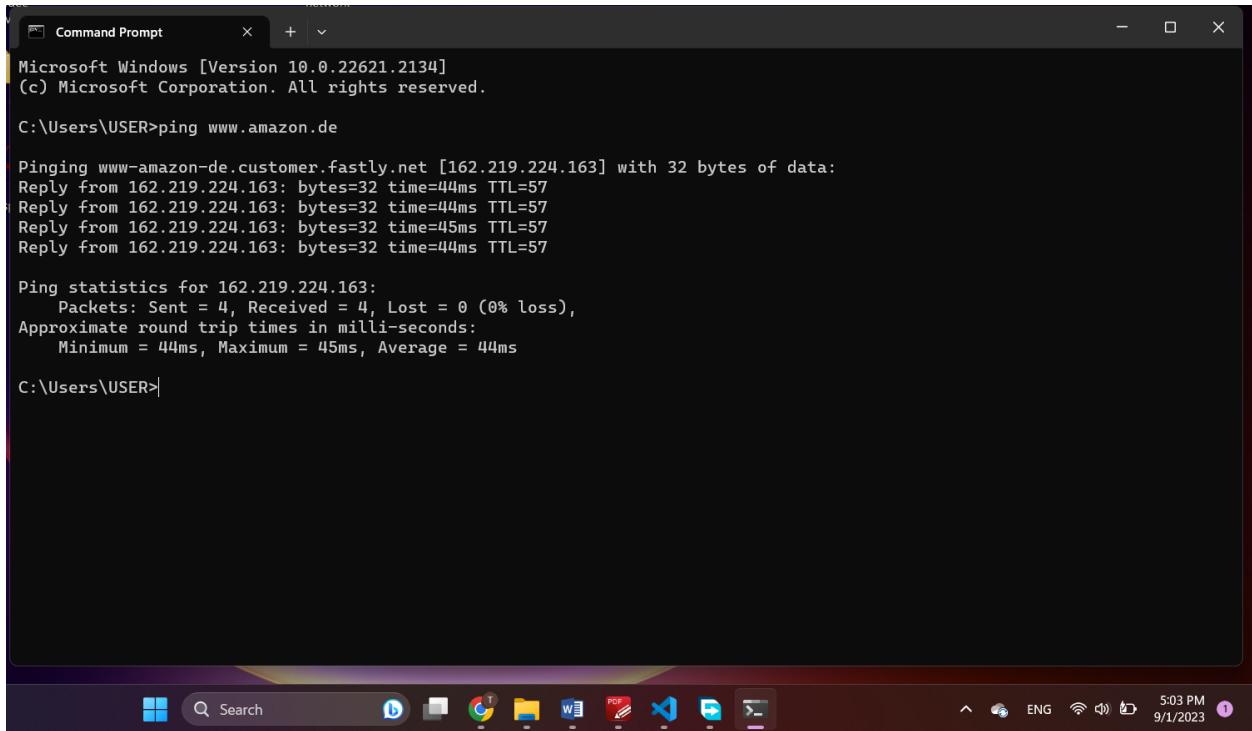
#### **1.1.4 What is telnet**

Telnet is a network protocol and command-line tool that allows you to establish a remote terminal session with another computer over a network, typically the internet. It enables you to access the command-line interface of a remote server or device as if you were physically present at that machine. However, telnet is not secure and transmits data, including passwords, in plaintext, which makes it unsuitable for sensitive information. As a result, its usage has largely been replaced by more secure alternatives like SSH (Secure Shell).

## 1.2 Commands

### 1.2.1 Ping [www.amazon.de](http://www.amazon.de)

As seen in figure 1, four packets were transmitted with the same TTL but various time delays of Minimum = 23ms, Maximum = 24ms, and Average = 23ms. We can see that the smartphone response time is shorter than the www.yale.edu response time, Since the two servers are not in the same network, the time for each packet to travel to and from its destination is more than when they were in the same network in the previous part.



```
Command Prompt
Microsoft Windows [Version 10.0.22621.2134]
(c) Microsoft Corporation. All rights reserved.

C:\Users\USER>ping www.amazon.de

Pinging www-amazon-de.customer.fastly.net [162.219.224.163] with 32 bytes of data:
Reply from 162.219.224.163: bytes=32 time=44ms TTL=57
Reply from 162.219.224.163: bytes=32 time=44ms TTL=57
Reply from 162.219.224.163: bytes=32 time=45ms TTL=57
Reply from 162.219.224.163: bytes=32 time=44ms TTL=57

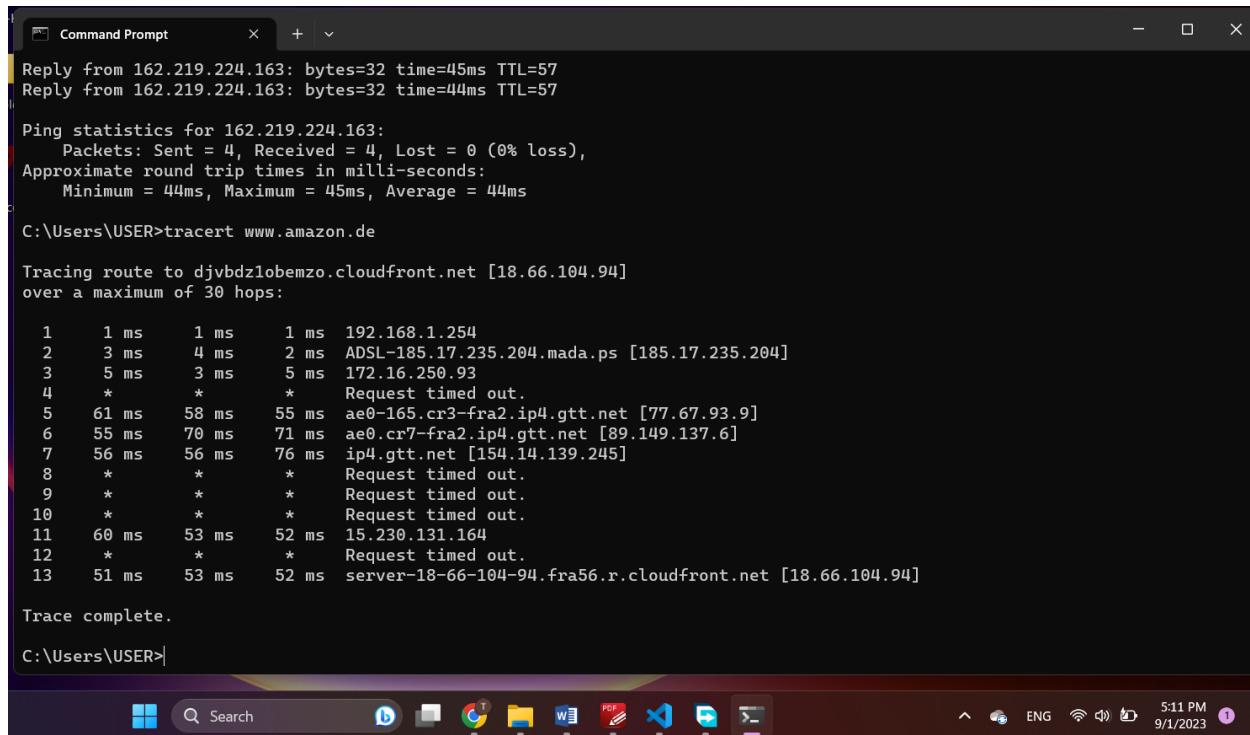
Ping statistics for 162.219.224.163:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 44ms, Maximum = 45ms, Average = 44ms

C:\Users\USER>
```

Figure 1: Ping [www.amazon.de](http://www.amazon.de)

## 1.2.2 Tracert [www.amazon.de](http://www.amazon.de)

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



```
Command Prompt
Reply from 162.219.224.163: bytes=32 time=45ms TTL=57
Reply from 162.219.224.163: bytes=32 time=44ms TTL=57

Ping statistics for 162.219.224.163:
  Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
  Approximate round trip times in milli-seconds:
    Minimum = 44ms, Maximum = 45ms, Average = 44ms

C:\Users\USER>tracert www.amazon.de

Tracing route to djvbdzlobemzo.cloudfront.net [18.66.104.94]
over a maximum of 30 hops:

 1  1 ms    1 ms    1 ms  192.168.1.254
 2  3 ms    4 ms    2 ms  ADSL-185.17.235.204.mada.ps [185.17.235.204]
 3  5 ms    3 ms    5 ms  172.16.250.93
 4  *        *        *      Request timed out.
 5  61 ms   58 ms   55 ms  ae0-165.cr3-fra2.ip4.gtt.net [77.67.93.9]
 6  55 ms   70 ms   71 ms  ae0.cr7-fra2.ip4.gtt.net [89.149.137.6]
 7  56 ms   56 ms   76 ms  ip4.gtt.net [154.14.139.245]
 8  *        *        *      Request timed out.
 9  *        *        *      Request timed out.
10  *        *        *      Request timed out.
11  60 ms   53 ms   52 ms  15.230.131.164
12  *        *        *      Request timed out.
13  51 ms   53 ms   52 ms  server-18-66-104-94.fra56.r.cloudfront.net [18.66.104.94]

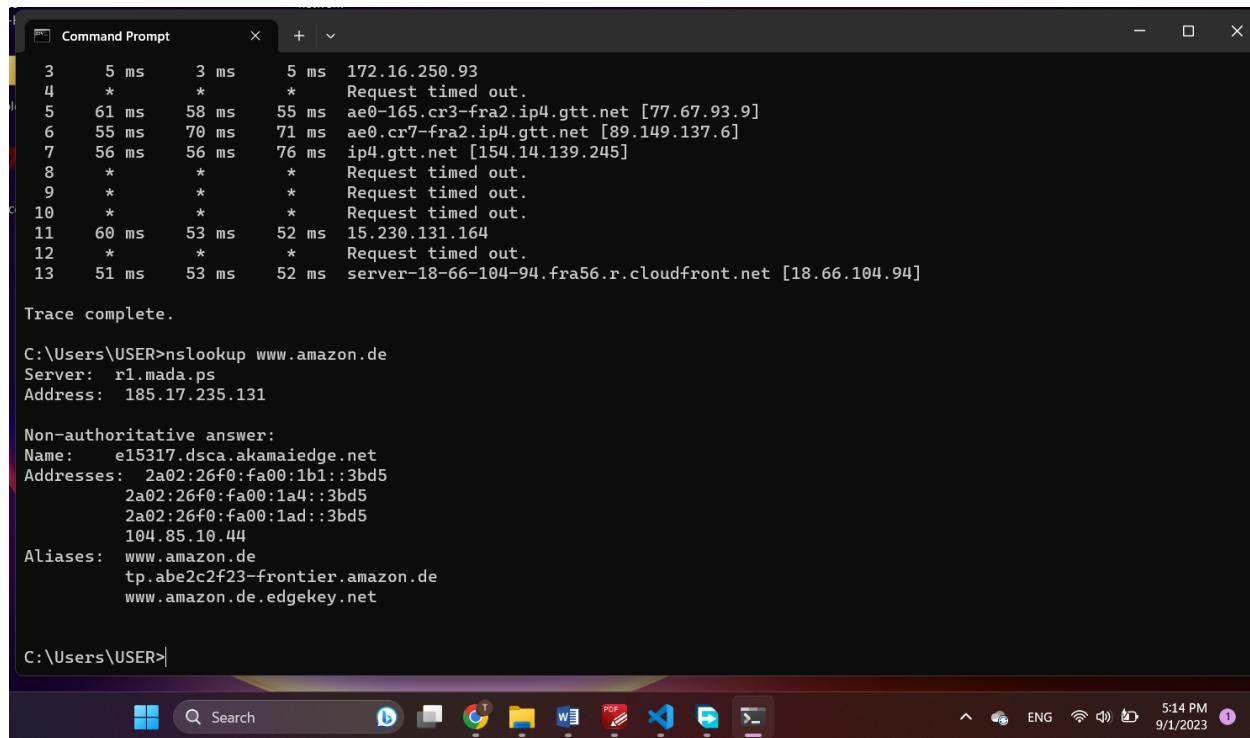
Trace complete.

C:\Users\USER>
```

Figure 2: Tracert [www.amazon.de](http://www.amazon.de)

### 1.2.3 nslookup [www.amazon.de](http://www.amazon.de)

nslookup is used to display the IP address and other information, such as the name of the server As shown in figure 3



```
Command Prompt
3      5 ms    3 ms    5 ms  172.16.250.93
4      *          *          * Request timed out.
5     61 ms    58 ms   55 ms  ae0-165.cr3-fra2.ip4.gtt.net [77.67.93.9]
6     55 ms    70 ms   71 ms  ae0.cr7-fra2.ip4.gtt.net [89.149.137.6]
7     56 ms    56 ms   76 ms  ip4.gtt.net [154.14.139.245]
8      *          *          * Request timed out.
9      *          *          * Request timed out.
10     *          *          * Request timed out.
11    60 ms    53 ms   52 ms  15.230.131.164
12     *          *          * Request timed out.
13    51 ms    53 ms   52 ms  server-18-66-104-94.fra56.r.cloudfront.net [18.66.104.94]

Trace complete.

C:\Users\USER>nslookup www.amazon.de
Server:  r1.mada.ps
Address: 185.17.235.131

Non-authoritative answer:
Name:  e15317.dsca.akamaiedge.net
Addresses:  2a02:26f0:fa00:1b1::3bd5
           2a02:26f0:fa00:1a4::3bd5
           2a02:26f0:fa00:1ad::3bd5
           104.85.10.44
Aliases:  www.amazon.de
          tp.abe2c2f23-frontier.amazon.de
          www.amazon.de.edgekey.net

C:\Users\USER>
```

Figure 3: nslookup www.amazon.de

## 2. Part 2

In order to achieve the project requirements, a simple web server was implemented using socket programming. The primary objective of this server is to listen on port 12345 and respond to specific URL requests with appropriate content.

The server was configured to listen on port 12345. This involves setting up a socket connection and binding it to the specified port. The server is designed to listen for incoming connections and process client requests.

### 2.1 / , /index.html , /main\_en.html and /en requests

Beyond the initial requirements of "/", "/index.html", "/main\_en.html", and "/en", the server accommodated an array of possible scenarios.

- if the request is '/':

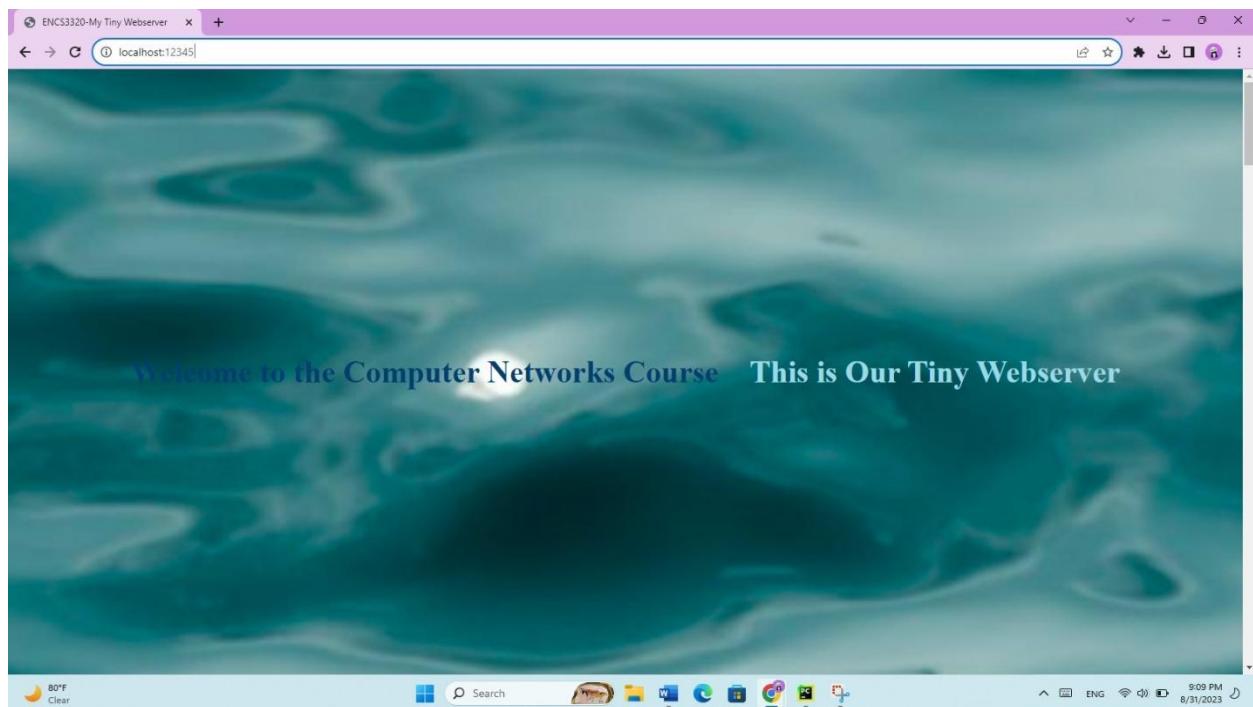
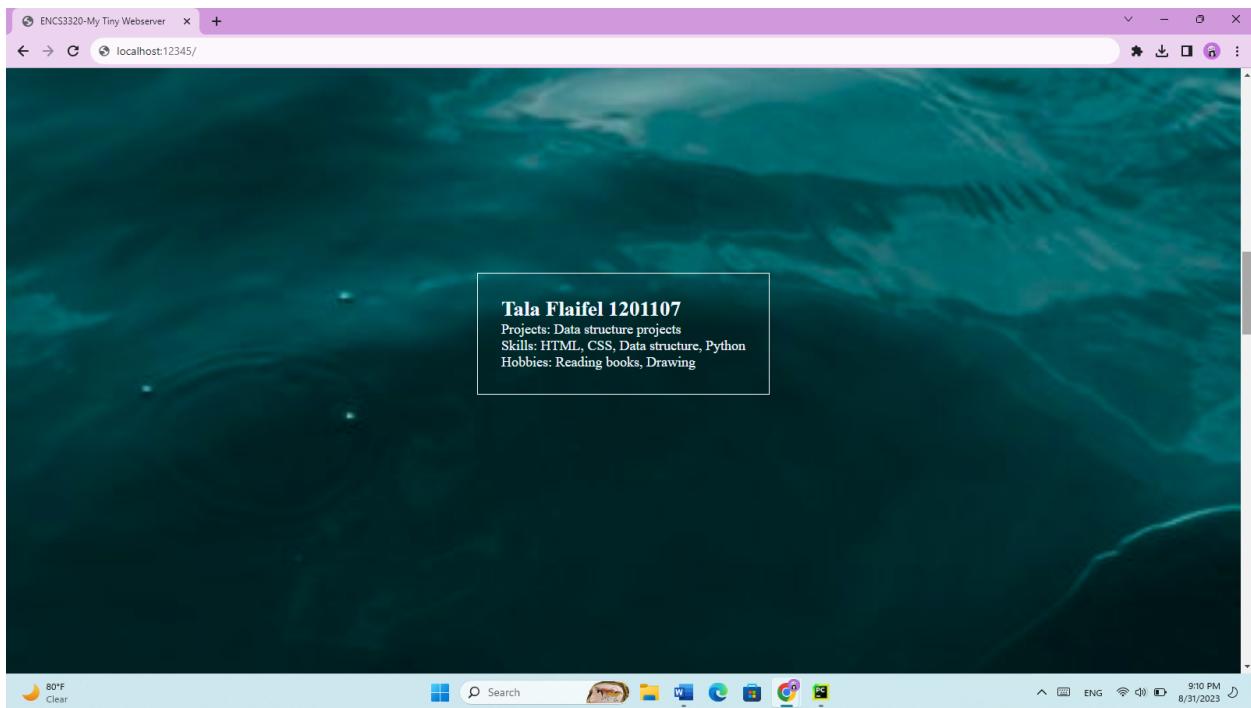
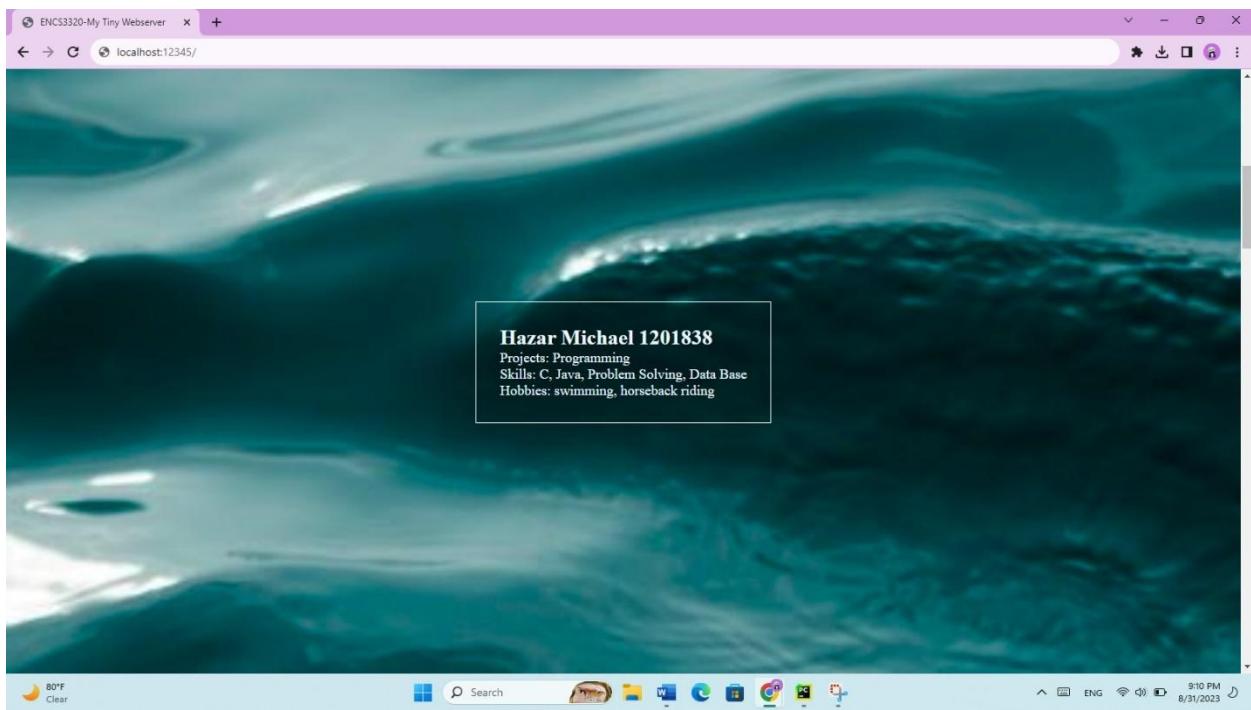
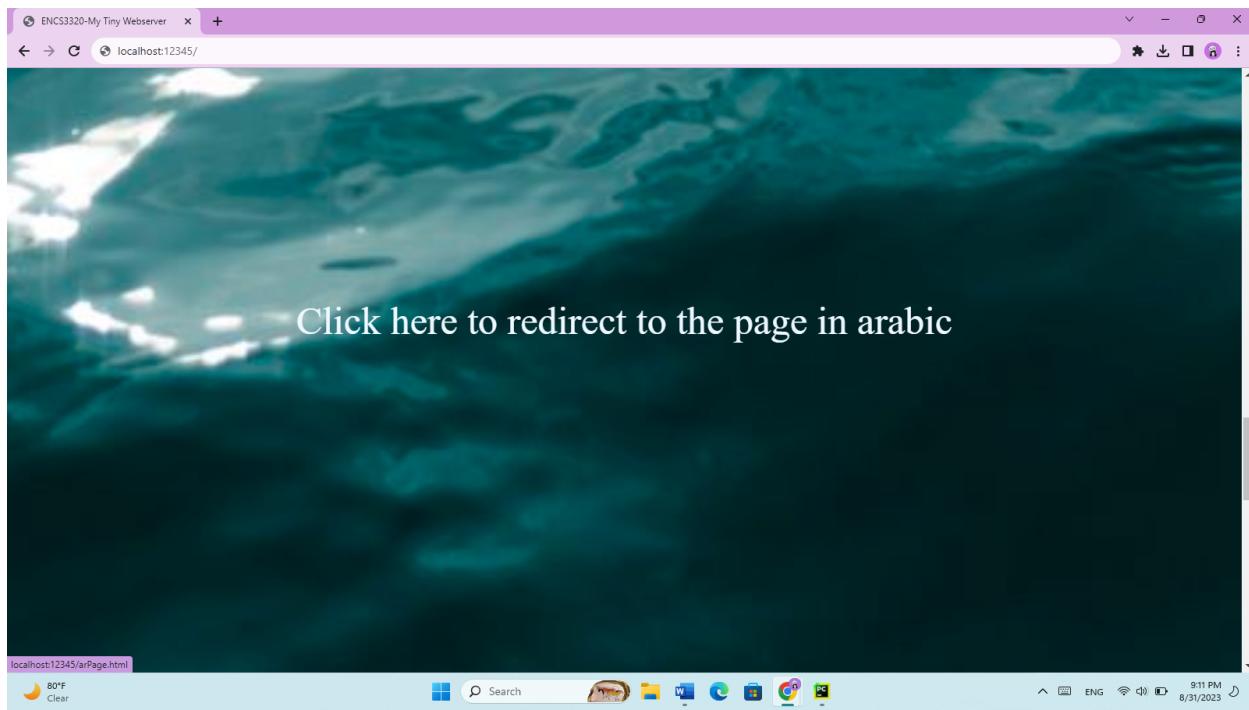
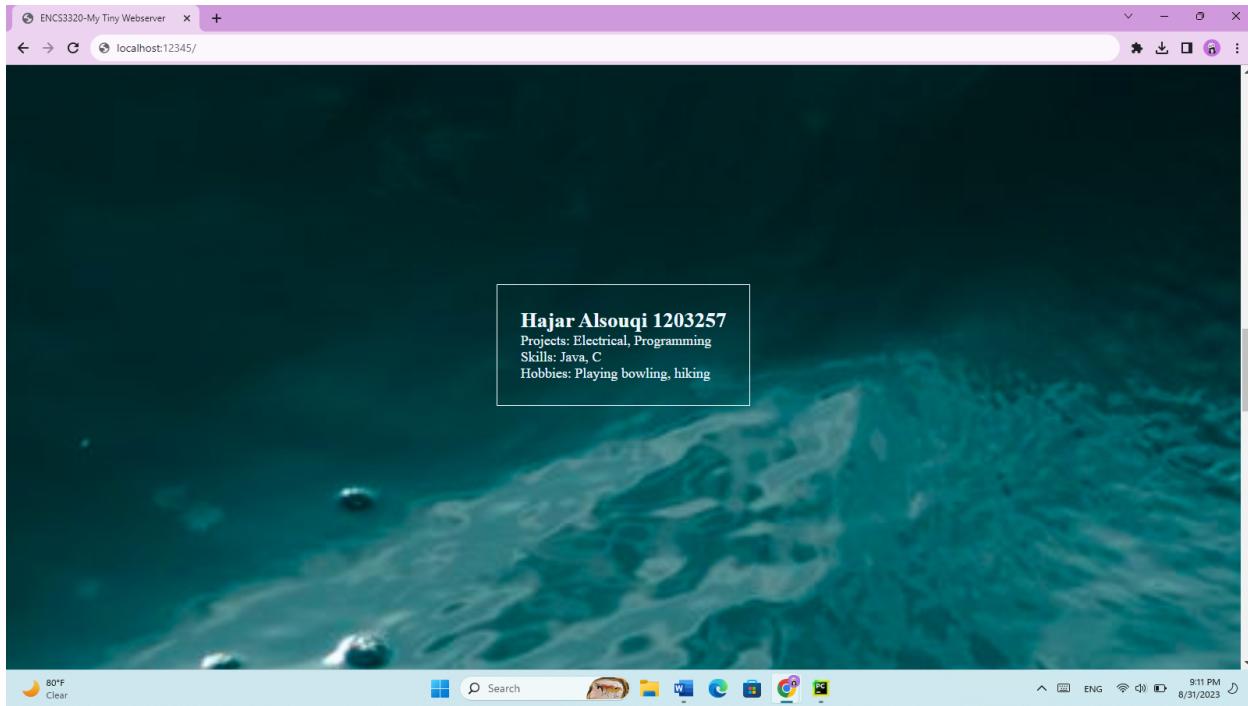


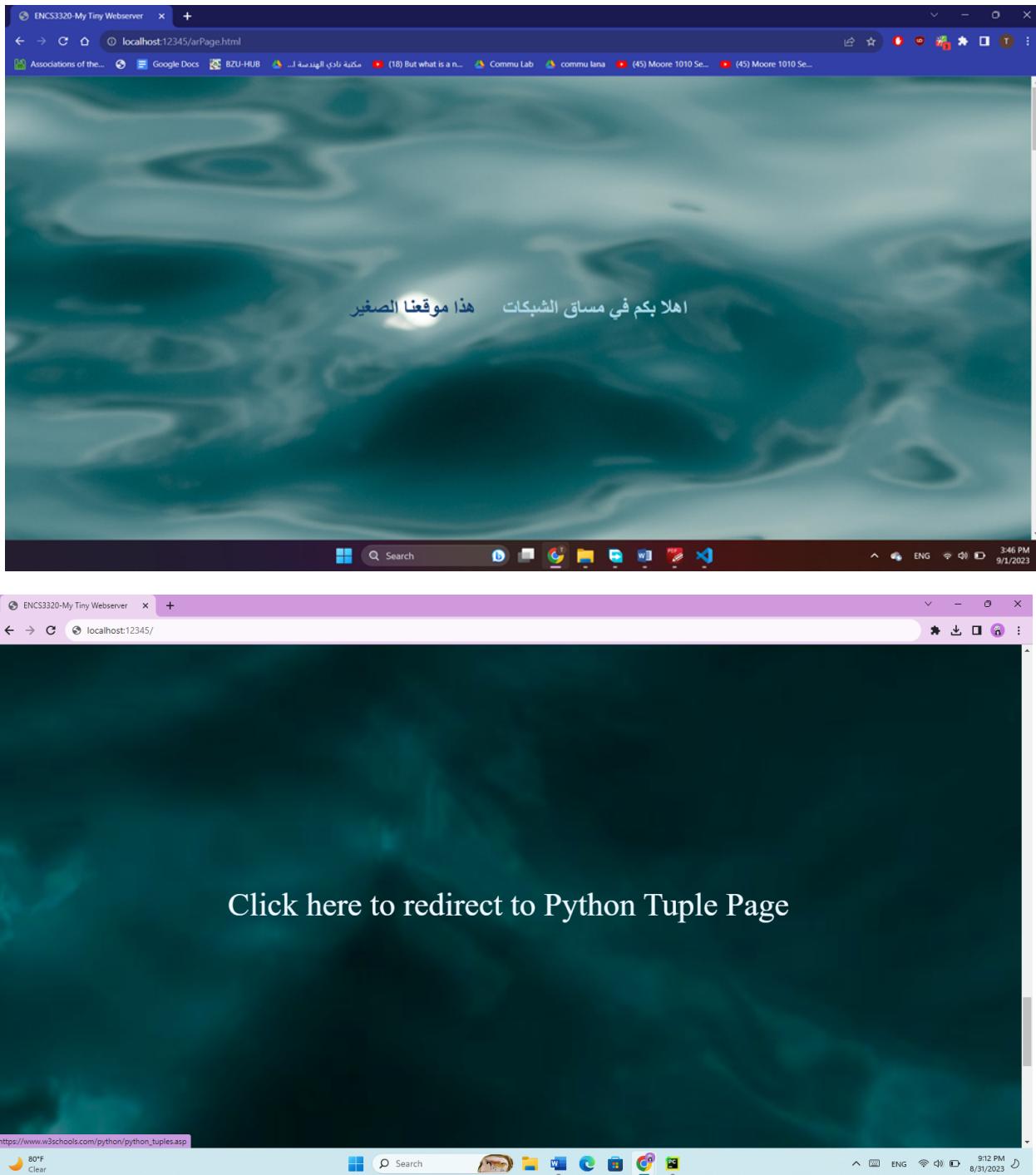
Figure 4: '/' request





The above image shows a hyperlink labeled "Click here to redirect to the page in Arabic," offering a user-friendly option for visitors to seamlessly transition to the Arabic version of the web page upon clicking.

## Redirected page



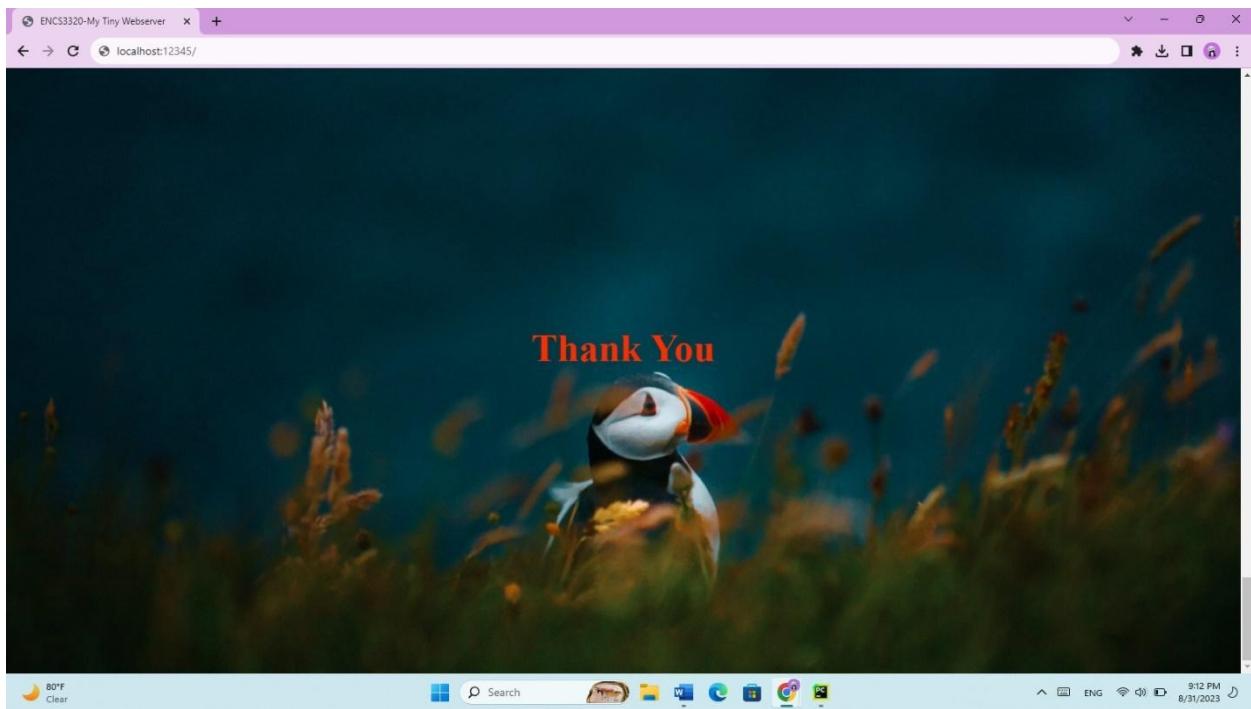
The above image shows a hyperlink reading "Click here to redirect to Python Tuple Page," serving as a direct link to the URL [https://www.w3schools.com/python/python\\_tuples.asp](https://www.w3schools.com/python/python_tuples.asp) as specified. This feature offers users swift access to the relevant educational content on Python tuples.

## Redirected page

The screenshot shows a web browser window with the title "Python Tuples". The URL in the address bar is [w3schools.com/python/python\\_tuples.asp](https://www.w3schools.com/python/python_tuples.asp). The page content is titled "Python Tuples" and includes a code example: 

```
mytuple = ("apple", "banana", "cherry")
```

. To the right of the main content, there is a sidebar with a promotional banner for "Lifelong Access To All Current And Future Courses" and a "Get the deal" button. Below the banner are links for "COLOR PICKER" and social media icons for Facebook, Instagram, LinkedIn, and GitHub. At the bottom of the page, there is a "Get your own Python Server" button.



The forthcoming images illustrate that the following requests: "/", "/index.html", "/main\_en.html", or "/en" (e.g., "localhost:12345/" or "localhost:12345/en"), all direct us to the central main web server. This unified response mechanism ensures consistent access to the main server content regardless of the specified path or language parameter.

- if request ‘/index.html’:



Figure 5: '/index.html' request

- if request ‘/main\_en.html’:



Figure 6: '/main\_en.html' request

- if the request '/en':

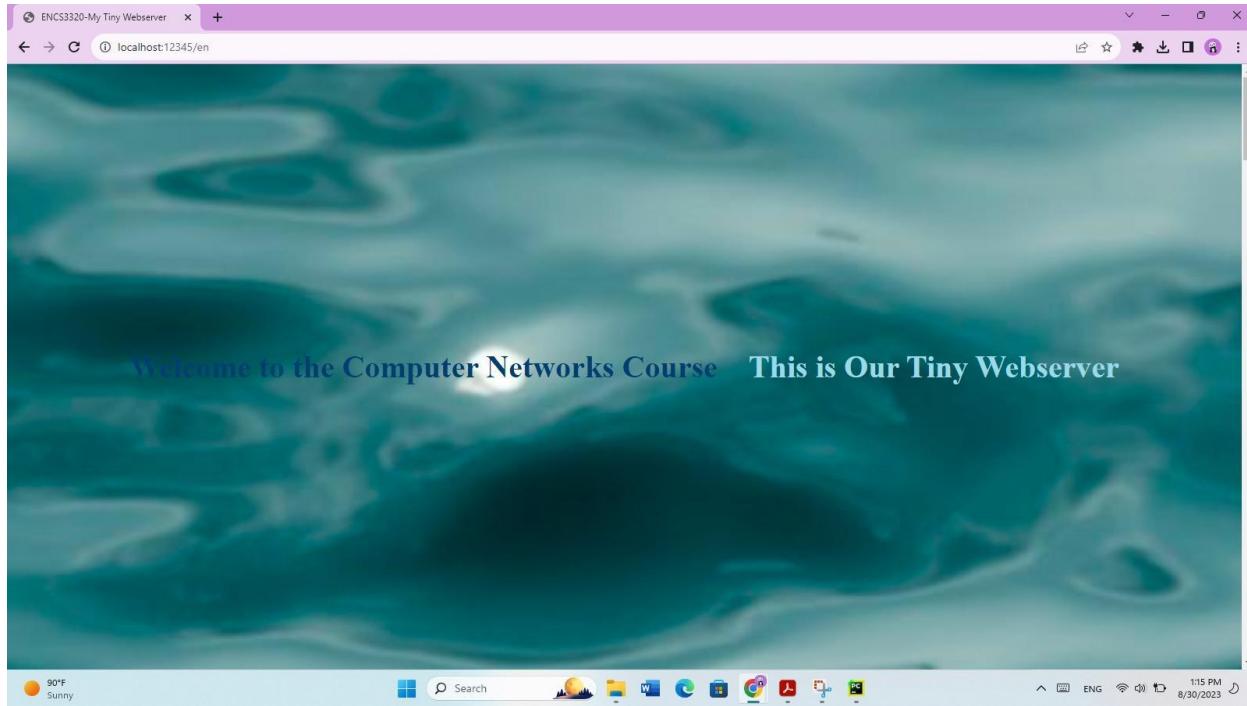


Figure 7: '/en.html' request

```

while True:
    print("The [Server] is ready to receive ...")
    connectionSocket, client_address = serverSocket.accept()
    data = connectionSocket.recv(1048).decode()
    print(data)
    browser_lines = data.split('\n')
    browser = browser_lines[0]

    if len(data) > 0:

        request_line = data.split('\n')[0]
        request_path = request_line.split()[1]
        client_req_url = browser.split()[1]

        if request_path in ('/', '/index.html', '/main_en.html', '/en', '/enPage.html'):
            connectionSocket.send('HTTP/1.1 200 OK\r\n'.encode())
            connectionSocket.send('Content-Type: text/html; charset=UTF-8\r\n'.encode())
            connectionSocket.send('\r\n'.encode())
            with open("enPage.html", "rb") as file:
                connectionSocket.send(file.read())

```

Figure 8: Language-Specific Page Delivery

This code snippet handles the first request by checking if the requested path matches "/", "/index.html", "/main\_en.html", "/en", or "/enPage.html". If the match occurs, the server sends a "200 OK" response header along with the "Content-Type: text/html; charset=UTF-8" header.

## 2.2 /ar request

In response to the "/ar" request, the server provides "main\_ar.html," an Arabic counterpart of "main\_en.html." This facilitates language-specific content delivery based on user preferences, enhancing accessibility and user experience.

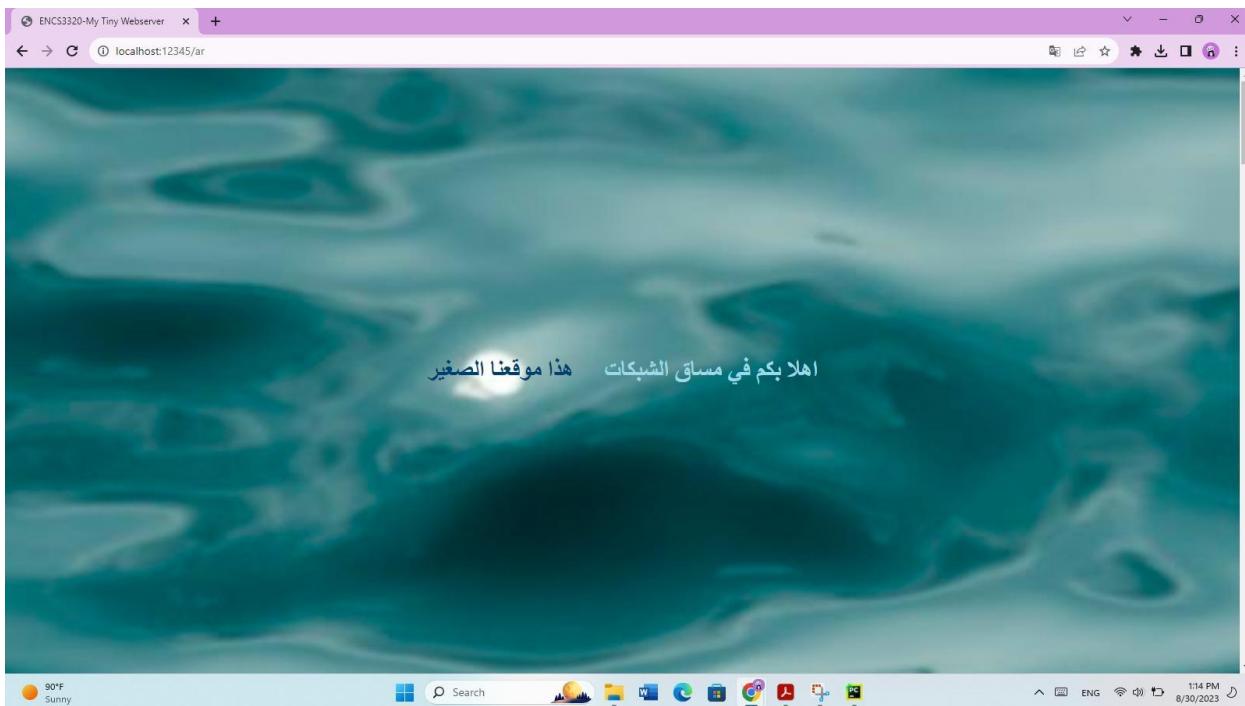
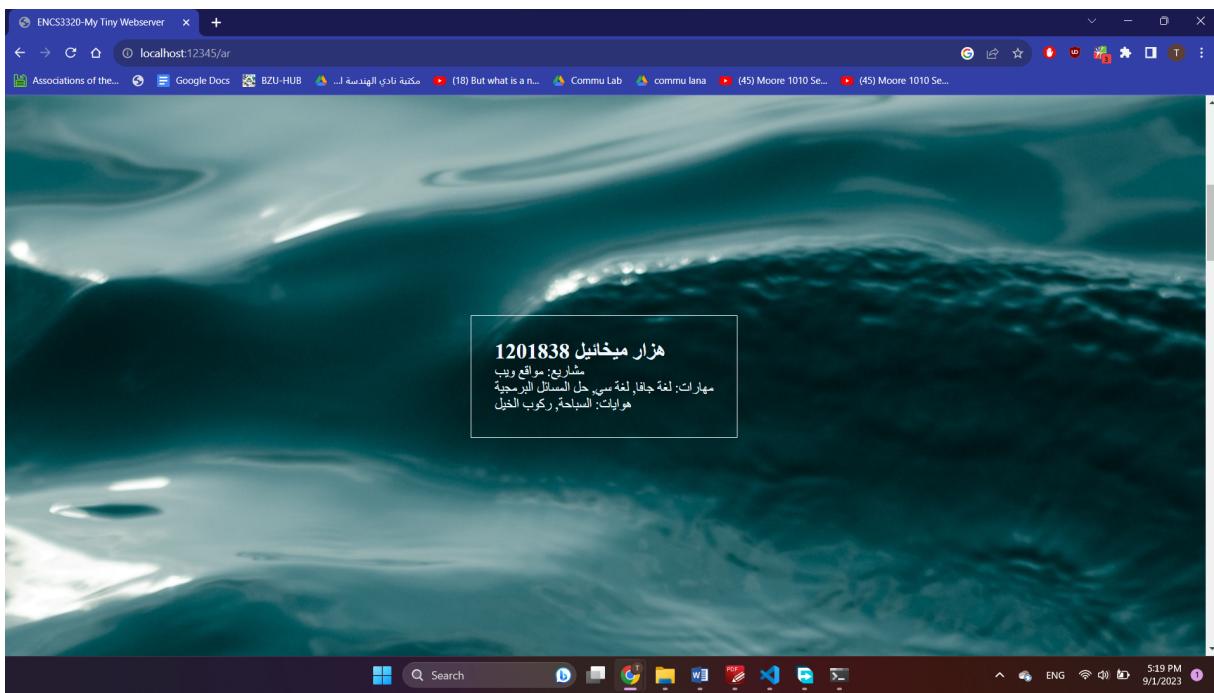
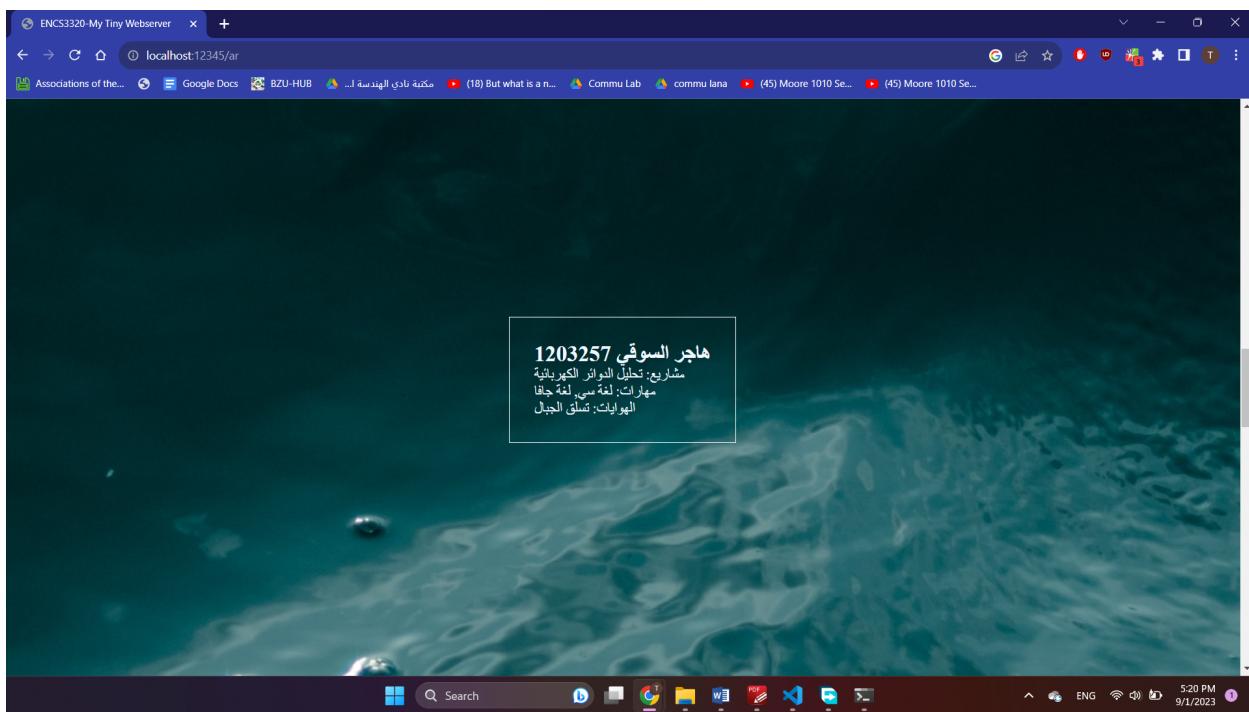
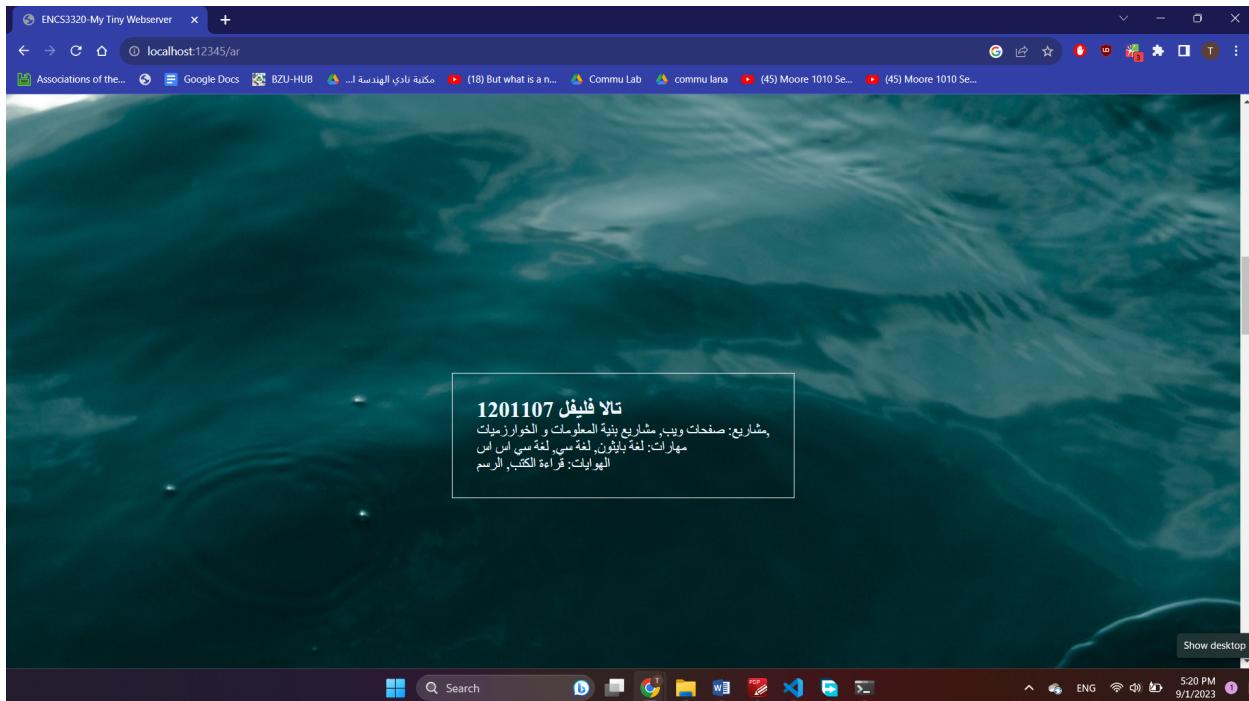
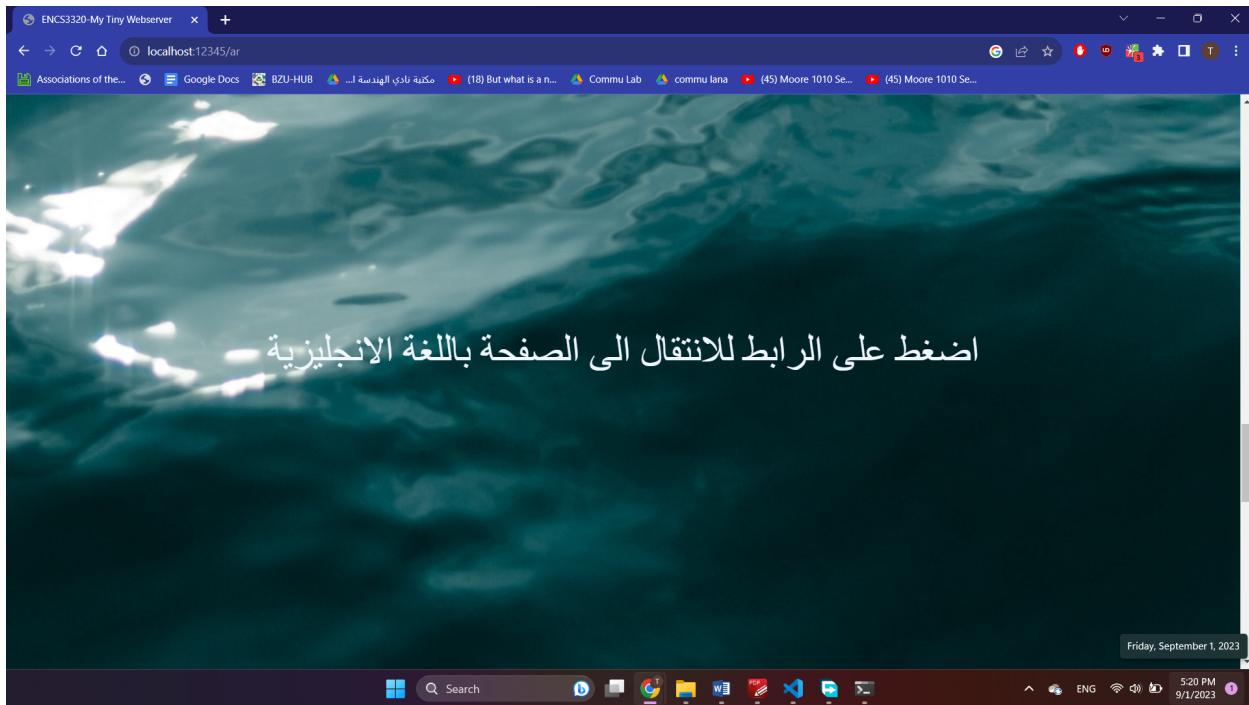


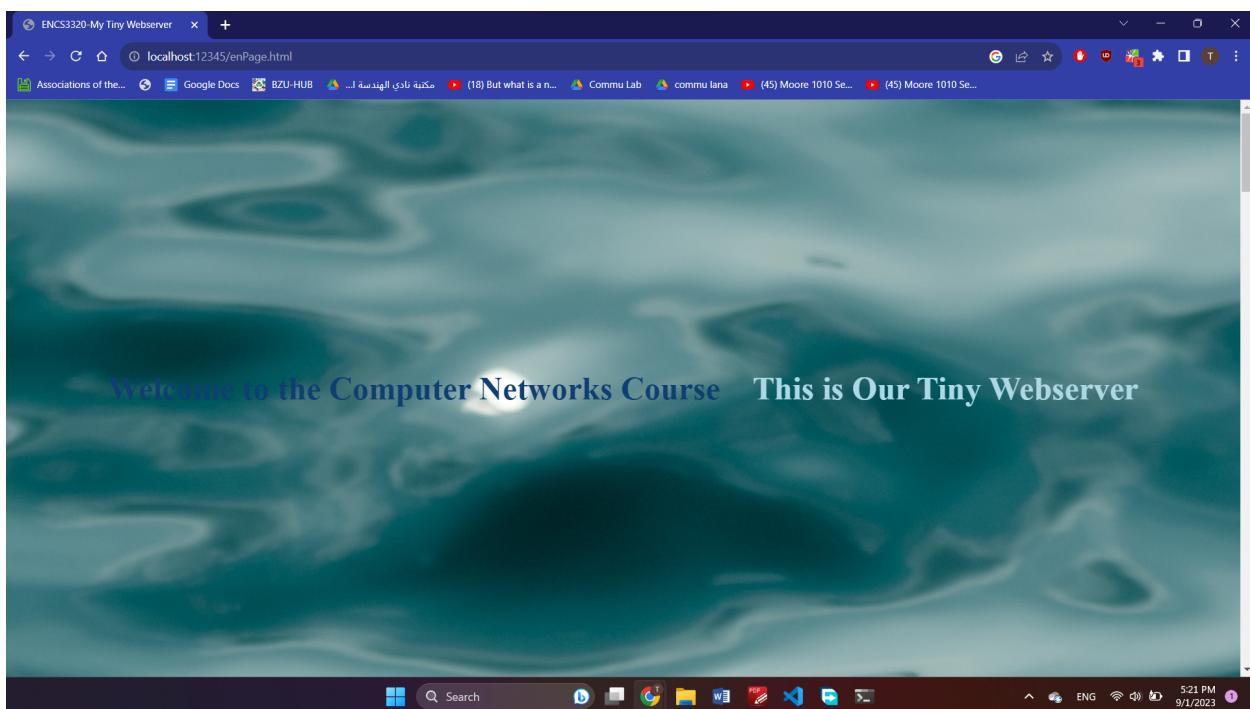
Figure 9: '/ar' request

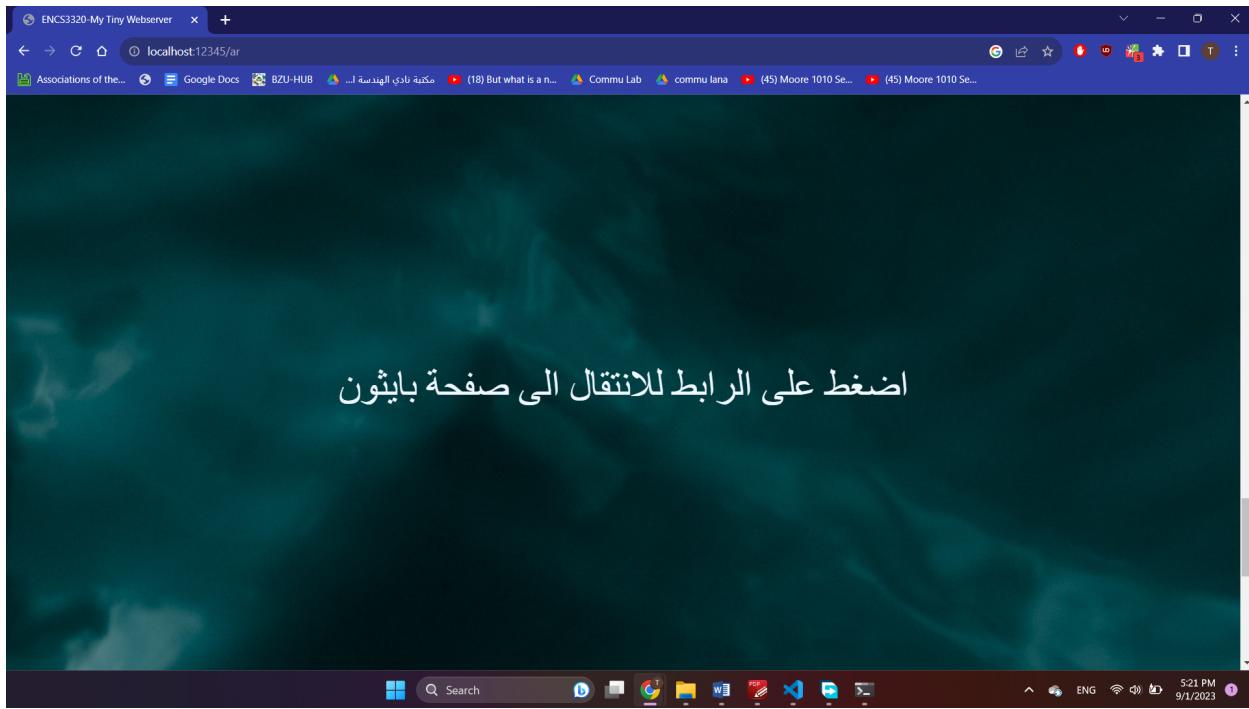






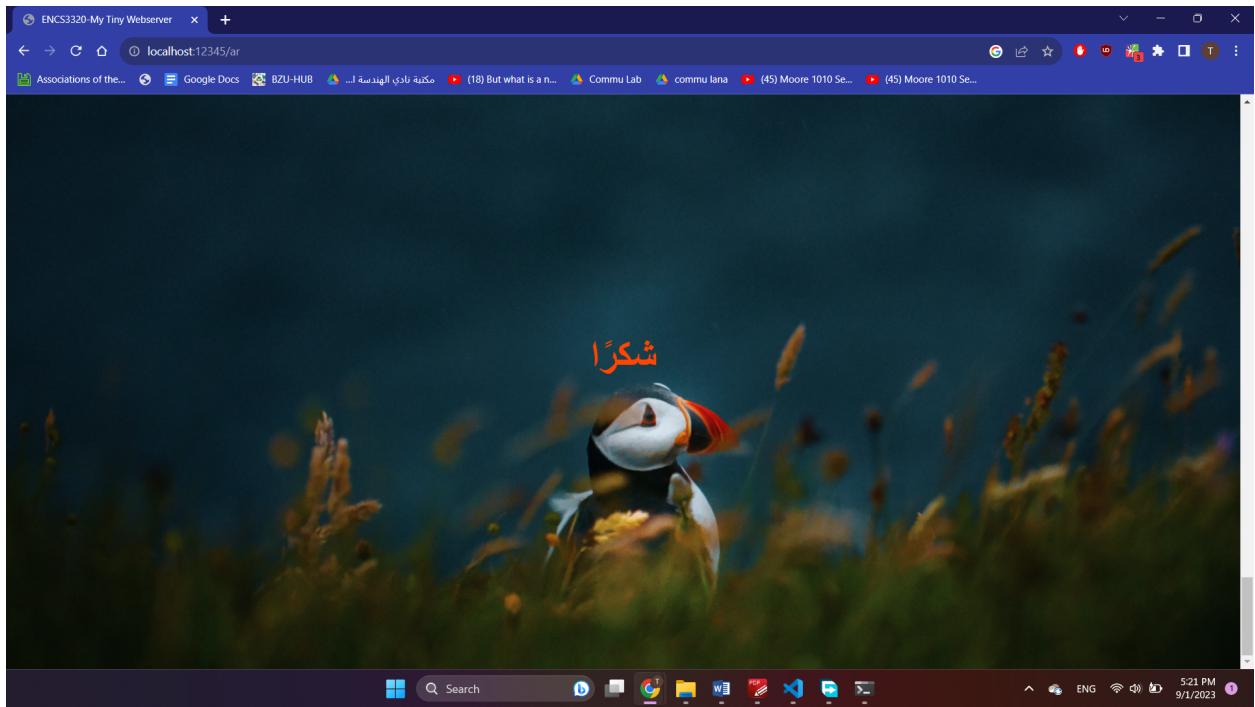
## Redirected page





## Redirected page

A screenshot of a web browser displaying the W3Schools Python Tuples tutorial. The page title is "Python Tuples". On the left, there is a sidebar with a navigation menu for Python tutorials, where "Python Tuples" is currently selected. The main content area contains text about tuples, code examples, and an "Example" section. A sidebar on the right features the W3Schools logo, a "COLOR PICKER" icon, and social media links. The browser's address bar shows "w3schools.com/python/python\_tuples.asp". The taskbar at the bottom shows the system clock indicating 5:23 PM on 9/1/2023.



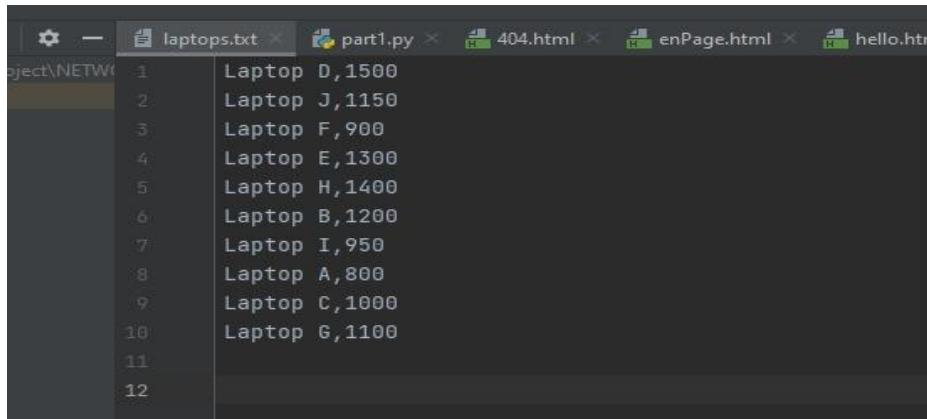
```
    connectionSocket.send('HTTP/1.1 200 OK\r\n'.encode())
elif request_path in ('/ar', '/arPage.html'):
    connectionSocket.send('HTTP/1.1 200 OK\r\n'.encode())
    connectionSocket.send('Content-Type: text/html; charset=UTF-8\r\n'.encode())
    connectionSocket.send('\r\n'.encode())
    with open("arPage.html", "rb") as file:
        connectionSocket.send(file.read())
```

Figure 10: Arabic Version Page Delivery

This code section corresponds to the demand of Arabic Version Page Delivery. When the requested path matches "/ar" or "/arPage.html", the server responds with the "arPage.html" file. The response headers are set to "200 OK" and "Content-Type: text/html; charset=UTF-8", and the server sends the content of the Arabic page file to the client, ensuring language-specific content delivery.

## 2.2 Text File for Laptop Data

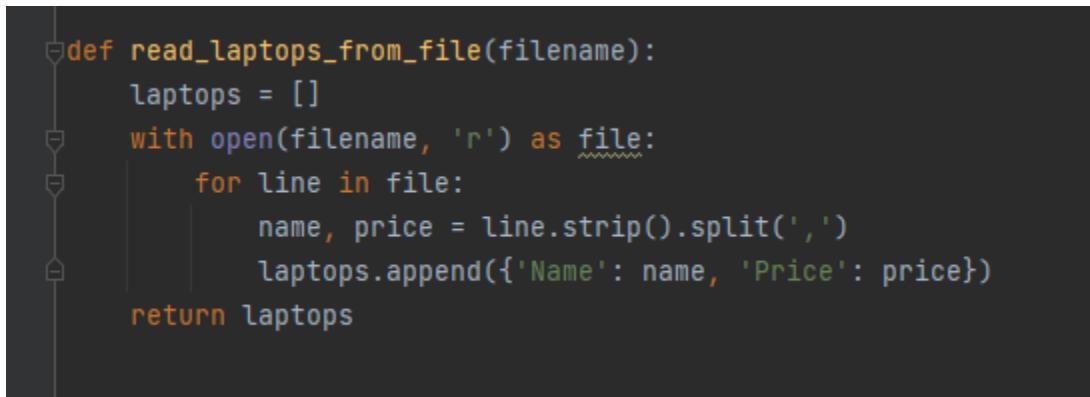
A laptop data file, containing names and prices of a minimum of 10 laptops, has been included. This file serves as the data source for subsequent server operations.



The screenshot shows a code editor interface with multiple tabs at the top: 'laptops.txt', 'part1.py', '404.html', 'enPage.html', and 'hello.htm'. The 'laptops.txt' tab is active, displaying the following content:

Line Number	Laptop Name, Price
1	Laptop D,1500
2	Laptop J,1150
3	Laptop F,900
4	Laptop E,1300
5	Laptop H,1400
6	Laptop B,1200
7	Laptop I,950
8	Laptop A,800
9	Laptop C,1000
10	Laptop G,1100
11	
12	

Figure 11 :text file contents



```
def read_laptops_from_file(filename):
    laptops = []
    with open(filename, 'r') as file:
        for line in file:
            name, price = line.strip().split(',')
            laptops.append({'Name': name, 'Price': price})
    return laptops
```

Figure 12: Loading Laptop Data from File Code

This code segment corresponds to Loading Laptop Data from a File. The function `read_laptops_from_file` reads data from a specified file containing laptop names and prices. It processes each line of the file, splitting it into name and price components based on a comma delimiter.

## 2.4 Sorting by Name (/SortByName) request

Upon receiving a "/SortByName" request, the server presents laptop names and prices in uppercase, sorted alphabetically. The output is displayed as a text page, enhancing readability and accessibility.

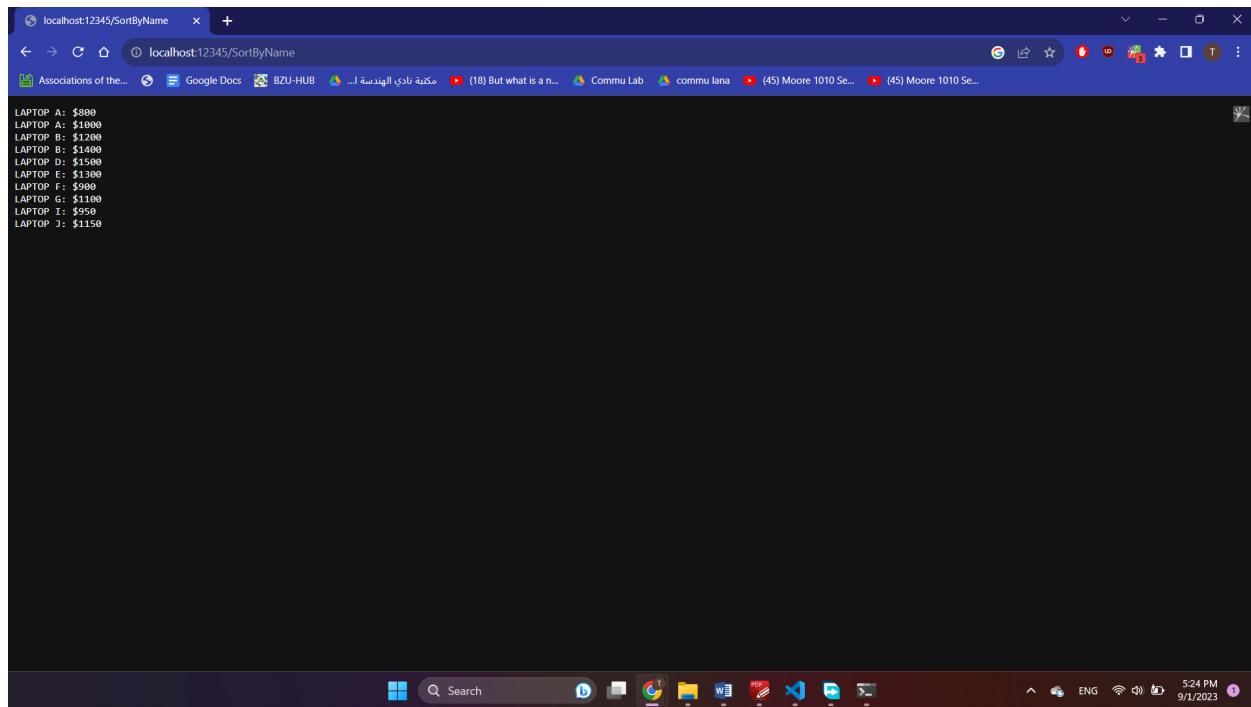


Figure 13:/SortByName request

## 2.5 Sorting by Price (/SortByPrice) request

For a "/SortByPrice" request, the server showcases laptop names and prices, sorted by price. Additionally, the total sum of all laptop prices is provided. The results are presented in a text page, facilitating easy comprehension.

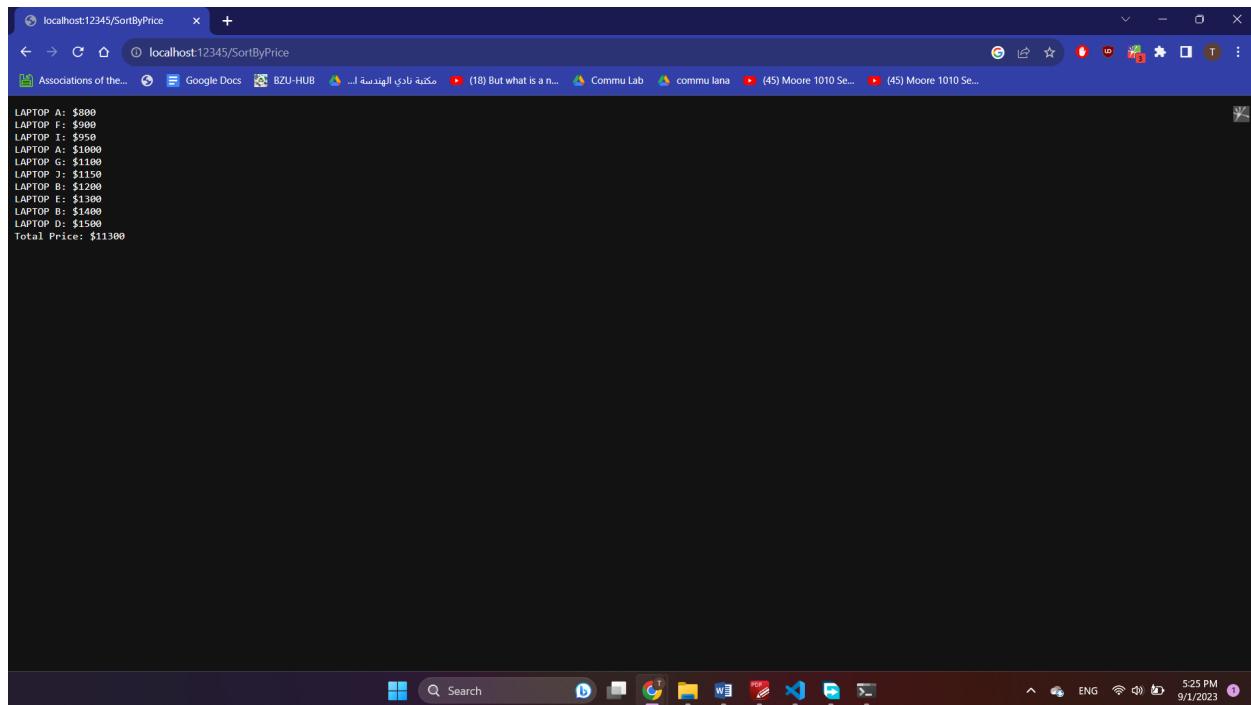


Figure 14:/SortByPrice request

## 2.6 HTML File Request Handling

if the request is an .html file then the server should send the requested html file with Content-Type: text/html.

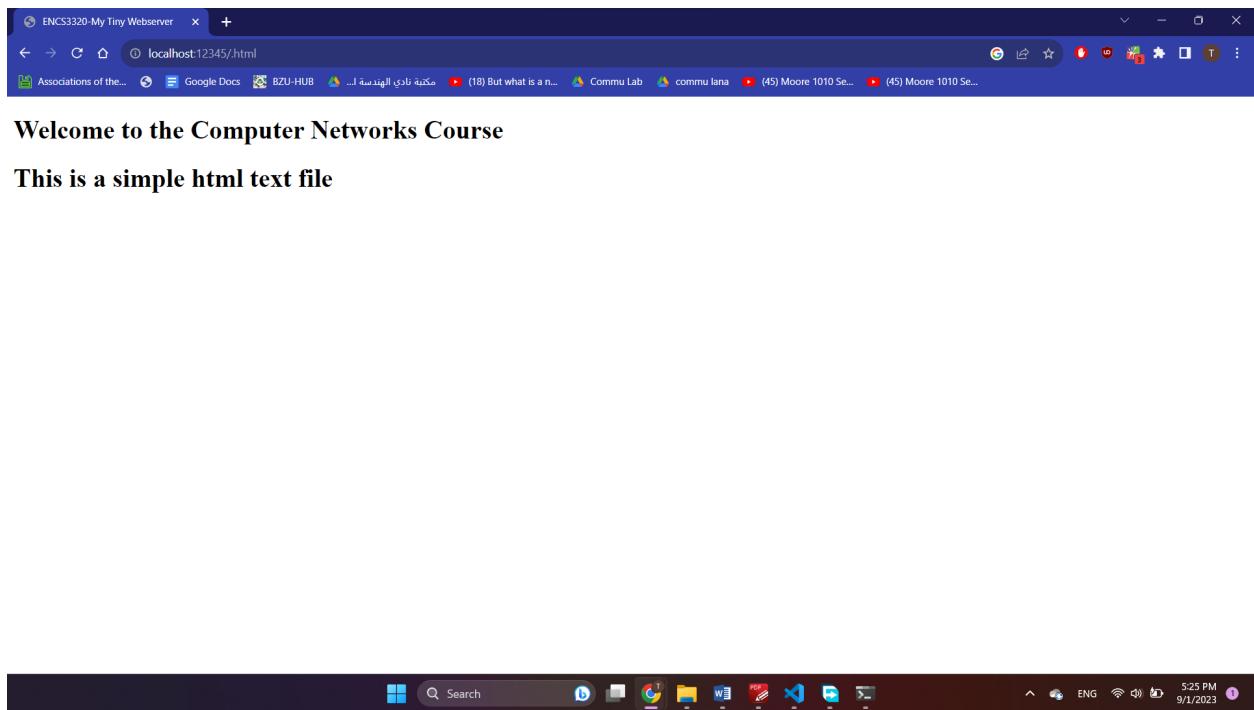


Figure 15: /.html file request

```
    connectionSocket.send(jpg_image)
elif client_req_url.endswith('/.html'):
    connectionSocket.send('HTTP/1.1 200 OK\r\n'.encode())
    connectionSocket.send('Content-Type: text/html; charset=UTF-8\r\n'.encode())
    connectionSocket.send('\r\n'.encode())
    with open("hello.html", "rb") as file:
        connectionSocket.send(file.read())
elif client_req_url.endswith('.css'):
```

Figure 16: HTML File Request Handling Code

This code snippet corresponds to HTML File Request. When the client's requested URL ends with ".html", the server responds with a "200 OK" status, setting the "Content-Type" header to "text/html; charset=UTF-8".

## 2.7 CSS File Request Handling

Requests for .css files prompt the server to provide the designated CSS content, tailored for "text/css; charset=UTF-8" content type.

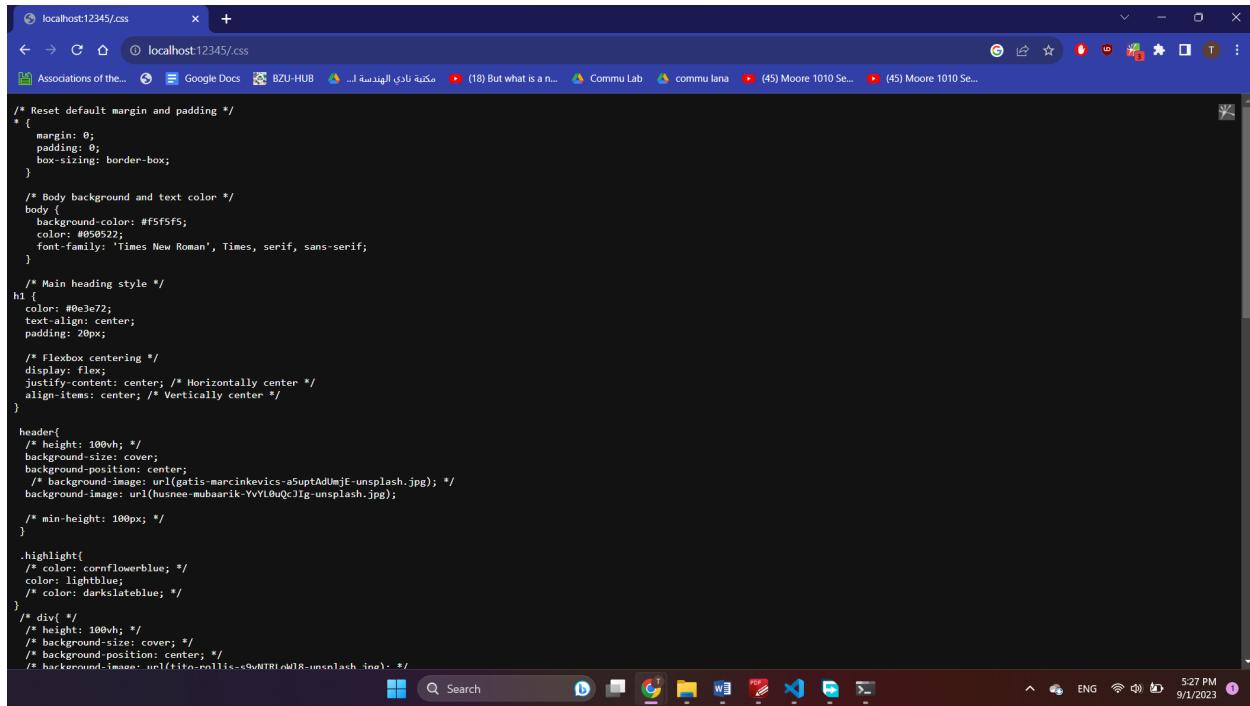


Figure 17: /.css request

## 2.8 .PNG Image Request Handling

For .png file requests, the server transmits the requested PNG image while ensuring the correct "Content-Type: image/png" setting. This technique guarantees accurate image rendering.

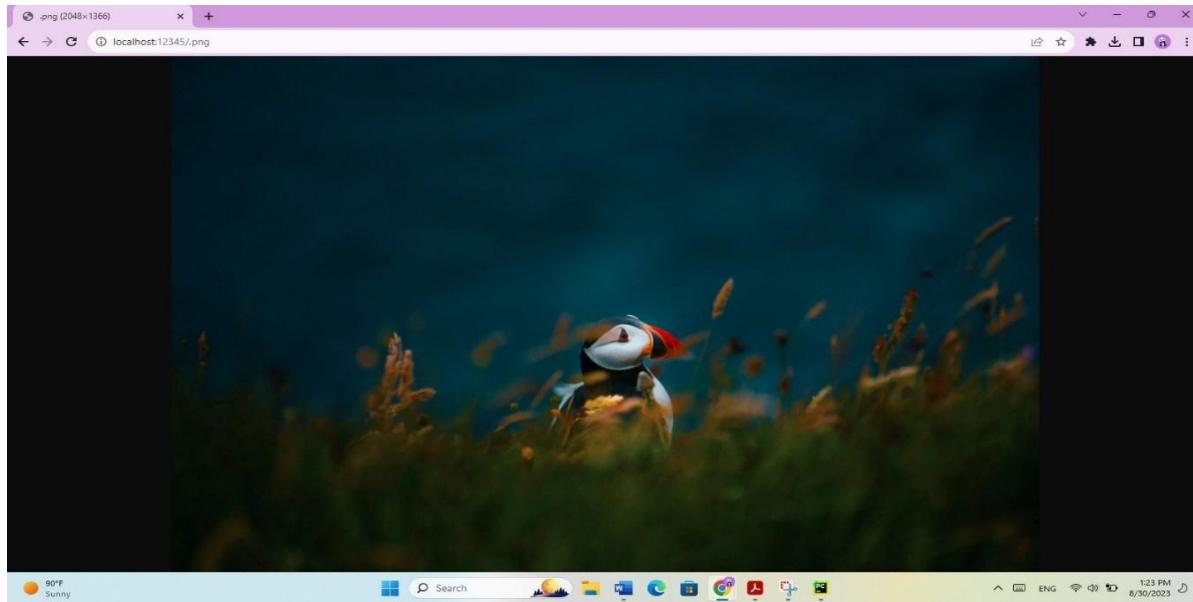


Figure 18: /.png request

## 2.9 JPG Image Request Handling

Requests for .jpg files lead the server to transmit the specified JPG image, accompanied by "Content-Type: image/jpeg." This protocol ensures optimal display of requested images in the browser.

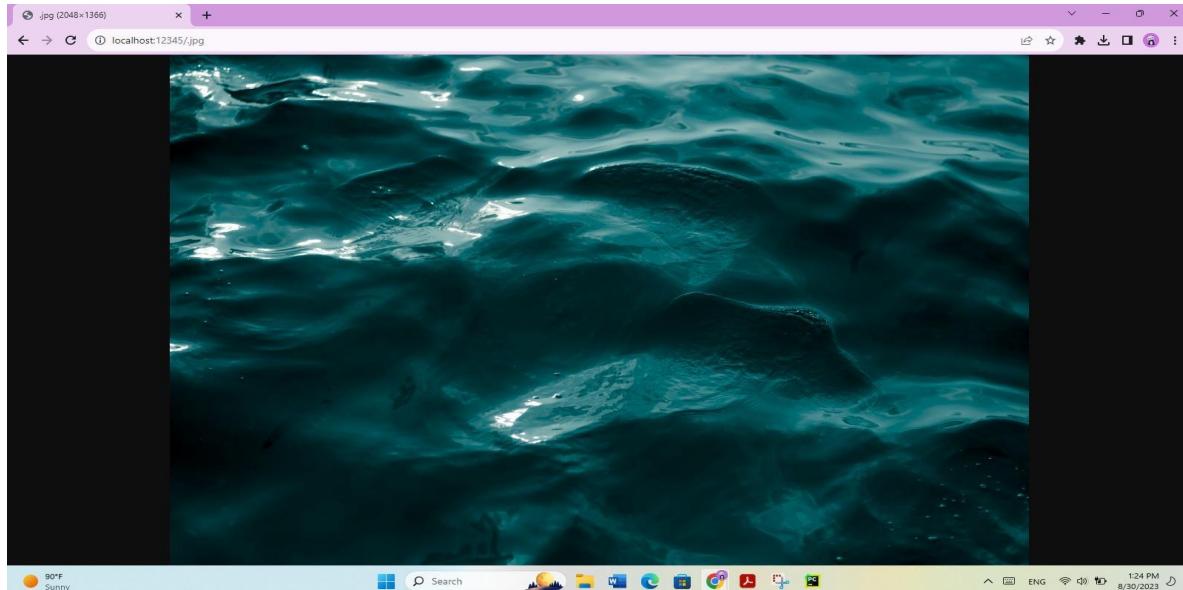


Figure 19: /.jpg request

## 2.10 Temporary Redirects

Utilizing the 307 Temporary Redirect status code, the server redirects these three requests.

### 2.10.1 /azn request redirect to amazon website

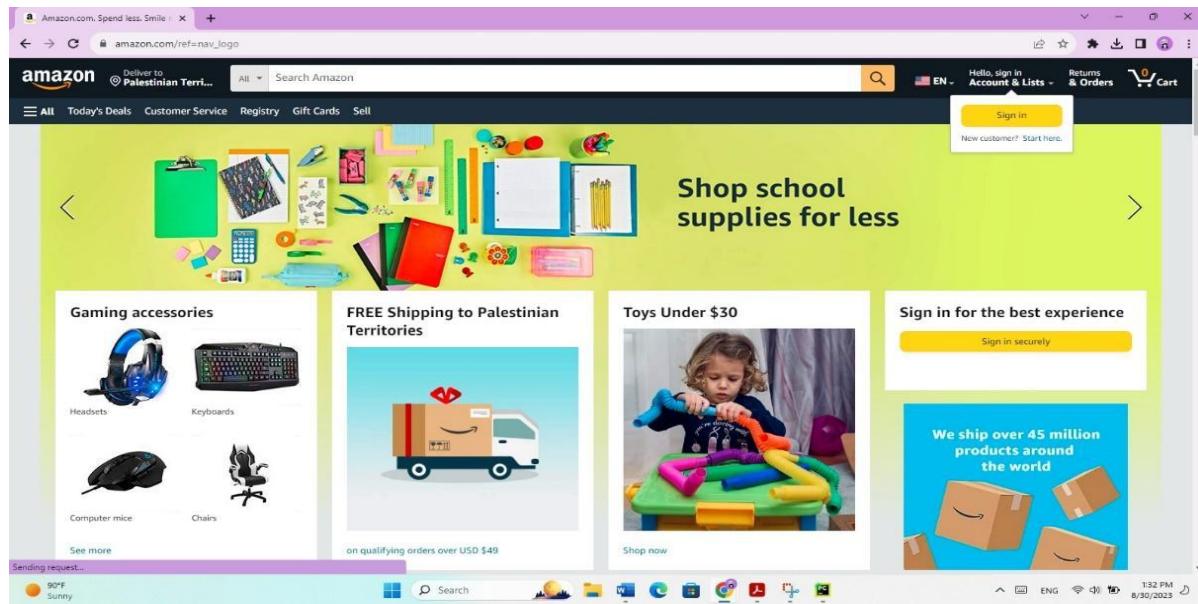


Figure 20: /azn request

### 2.10.2 /so request redirects to stackoverflow.com website

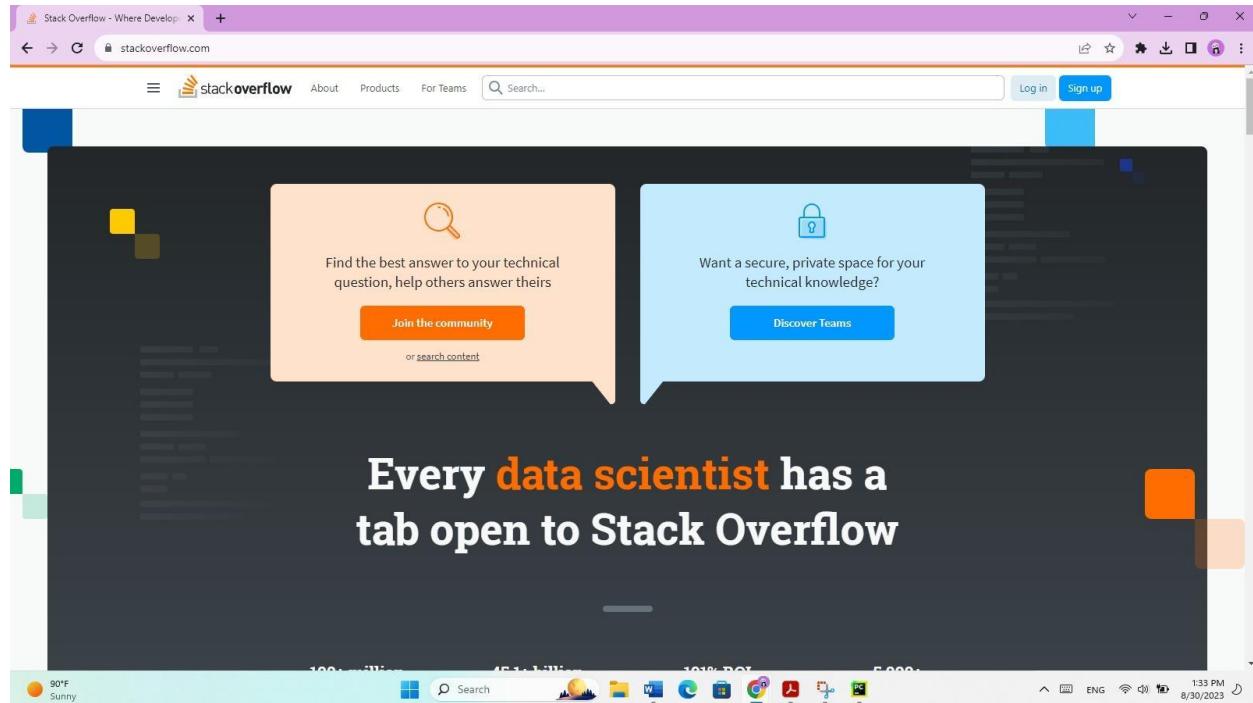


Figure 21: /so request

## 2.10.3 /bzu request redirect to birzeit website

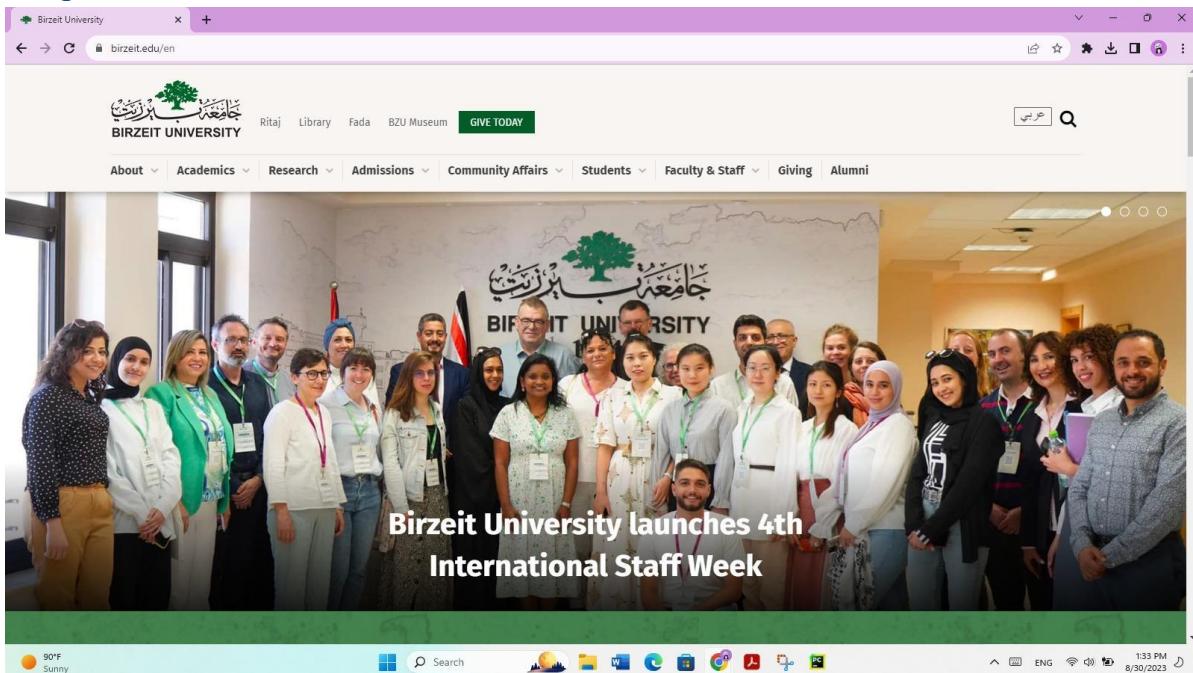


Figure 22: /bzu request

## 2.11 404 Not Found Handling

In case of incorrect or missing requests, the server generates a simple HTML page with the "HTTP/1.1 404 Not Found" status, "Error 404" in the title, and a noticeable "The file is not found" message in red. The page also includes bold names and IDs, as well as the client's IP and port for reference.

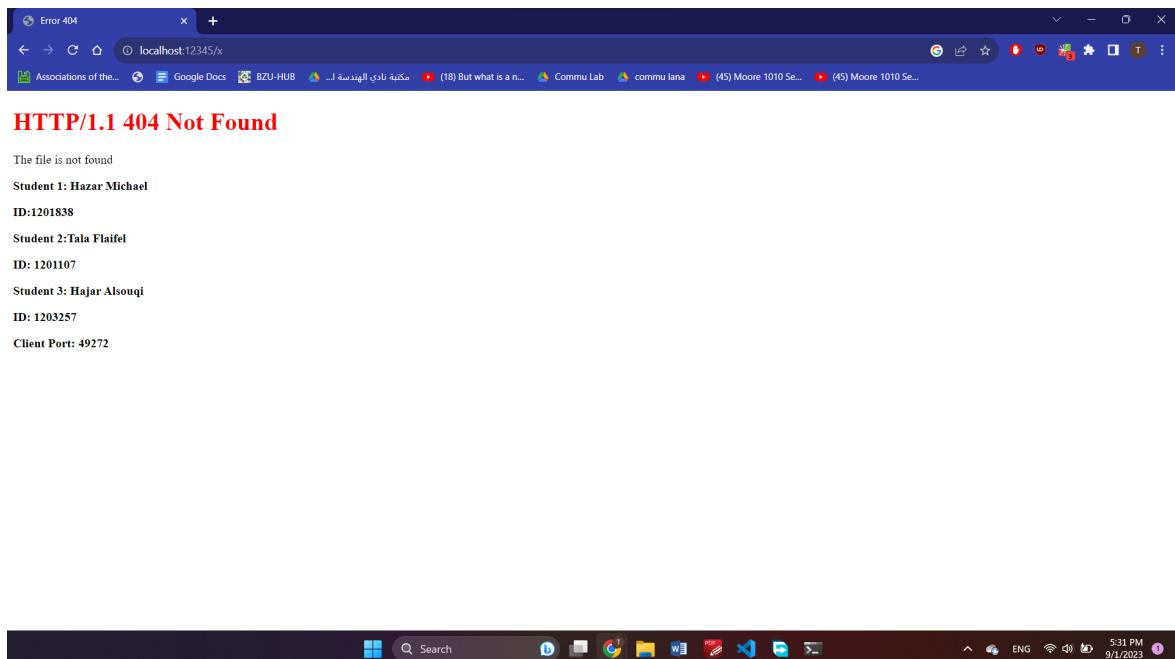
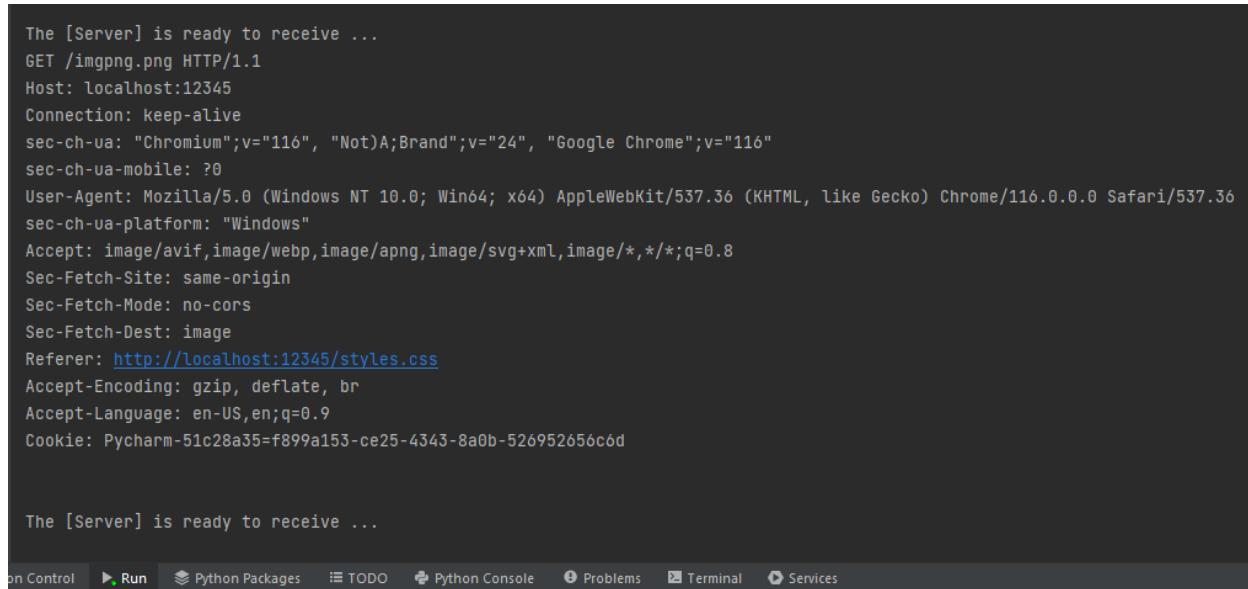


Figure 23: any invalid request output

## 2.12 HTTP Request Logging

The program logs incoming HTTP requests on the terminal window, enhancing monitoring and debugging capabilities. This feature aids in tracking server-client interactions and diagnosing potential issues.



```
The [Server] is ready to receive ...
GET /imgpng.png HTTP/1.1
Host: localhost:12345
Connection: keep-alive
sec-ch-ua: "Chromium";v="116", "Not)A;Brand";v="24", "Google Chrome";v="116"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36
sec-ch-ua-platform: "Windows"
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: image
Referer: http://localhost:12345/styles.css
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: Pycharm-51c28a35=f899a153-ce25-4343-8a0b-526952656c6d

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

The screenshot shows a terminal window with the following content:

- The first part of the terminal shows an incoming HTTP request from a browser (Chrome) on port 12345. The request is for the file "/imgpng.png". The browser's User-Agent header indicates it's running on Windows 10.0, using AppleWebKit/537.36 (KHTML, like Gecko) and Chrome/116.0.0.0.
- The second part of the terminal shows the server's response, which includes a cookie named "Pycharm-51c28a35" with the value "f899a153-ce25-4343-8a0b-526952656c6d".

The terminal window has a dark background and a light-colored text area. At the bottom, there is a navigation bar with icons and labels for "Run Control", "Run", "Python Packages", "TODO", "Python Console", "Problems", "Terminal", and "Services".

Figure 24: http request example

# Appendices

## Appendix A: main code

```
from socket import *

serverport = 12345
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind(('', serverport))
serverSocket.listen(1)

def generate_response_header(status, content_type):
    response = f'HTTP/1.1 {status}\r\n'
    response += f'Content-Type: {content_type}\r\n'
    response += '\r\n'
    return response.encode()

def sort_laptops_by_name(laptops):
    return sorted(laptops, key=lambda laptop: laptop['Name'].upper())

def sort_laptops_by_price(laptops):
    return sorted(laptops, key=lambda laptop: int(laptop['Price']))

def format_laptop_list(laptops):
    formatted_list = []
    for laptop in laptops:
        formatted_list.append(f'{laptop['Name'].upper()}: ${laptop['Price']}')
    return '\n'.join(formatted_list)

def read_laptops_from_file(filename):
    laptops = []
    with open(filename, 'r') as file:
        for line in file:
            name, price = line.strip().split(',')
            laptops.append({'Name': name, 'Price': price})
    return laptops

    laptops = read_laptops_from_file('laptops.txt')
while True:
    print("The [Server] is ready to receive ...")
    connectionSocket, client_address = serverSocket.accept()
    data = connectionSocket.recv(1048).decode()
    print(data)
    browser_lines = data.split('\r\n')
    browser = browser_lines[0]

    if len(data) > 0:

        request_line = data.split('\r\n')[0]
```

```

request_path = request_line.split()[1]
client_req_url = browser.split()[1]

        if request_path in ('/', '/index.html', '/main_en.html', '/en',
'/'enPage.html'):
            connectionSocket.send('HTTP/1.1 200 OK\r\n'.encode())
                                connectionSocket.send('Content-Type: text/html;
charset=UTF-8\r\n'.encode())
            connectionSocket.send('\r\n'.encode())
            with open("enPage.html", "rb") as file:
                connectionSocket.send(file.read())
elif request_path in ('/ar', '/arPage.html'):
    connectionSocket.send('HTTP/1.1 200 OK\r\n'.encode())
                                connectionSocket.send('Content-Type: text/html;
charset=UTF-8\r\n'.encode())
    connectionSocket.send('\r\n'.encode())
    with open("arPage.html", "rb") as file:
        connectionSocket.send(file.read())
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("styles.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("imgpng.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("husnee-mubaarak-YvYL0uQcJIG-unsplash.jpg", "rb").read()
# open english html file
    connectionSocket.send(jpg_image)
elif client_req_url.endswith('/.html'):
    connectionSocket.send('HTTP/1.1 200 OK\r\n'.encode())
                                connectionSocket.send('Content-Type: text/html;
charset=UTF-8\r\n'.encode())
    connectionSocket.send('\r\n'.encode())
    with open("hello.html", "rb") as file:
        connectionSocket.send(file.read())
elif client_req_url.endswith('.css'):
    connectionSocket.send('HTTP/1.1 200 OK\r\n'.encode())
                                connectionSocket.send('Content-Type: text/css;
charset=UTF-8\r\n'.encode())
    connectionSocket.send('\r\n'.encode())
    with open("styles.css", "rb") as file:
        connectionSocket.send(file.read())

try:
    with open(request_path[1:], "rb") as file:
        connectionSocket.send(file.read())
except FileNotFoundError:
    # Handle 404 here...

```

```

    pass

    elif client_req_url == '/SortByName':
        response = generate_response_header('200 OK', 'text/plain; charset=UTF-8')
        sorted_laptops = sort_laptops_by_name(laptops)
        laptop_list = format_laptop_list(sorted_laptops)
        connectionSocket.send(response)
        connectionSocket.send(laptop_list.encode())
    elif client_req_url == '/SortByPrice':
        response = generate_response_header('200 OK', 'text/plain; charset=UTF-8')
        sorted_laptops = sort_laptops_by_price(laptops)
        laptop_list = format_laptop_list(sorted_laptops)
        total_price = sum(int(laptop['Price']) for laptop in sorted_laptops)
        laptop_list += f"\nTotal Price: ${total_price}"
        connectionSocket.send(response)
        connectionSocket.send(laptop_list.encode())
    elif client_req_url == '/azn':
        redirect_link = b'HTTP/1.1 307 Temporary Redirect\nLOCATION:https://www.amazon.com/ref=nav_logo\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 == '/bzu':
        redirect_link = b'HTTP/1.1 307 Temporary Redirect\nLOCATION:https://www.birzeit.edu/en\n\n'
        connectionSocket.sendall(redirect_link)
    else:
        # Handle other cases (404 or redirects) here...
        response = generate_response_header('404 Not Found', 'text/html; charset=UTF-8')
        with open("404.html", "rb") as file:
            html_content = file.read().replace(b'{client_ip}', str(client_address[0]).encode()).replace(
                b'{client_port}', str(client_address[1]).encode())
            connectionSocket.sendall(response + html_content)
        pass

connectionSocket.close()

```

## Appendix B: styles.css code

```

/* Reset default margin and padding */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

/* Body background and text color */
body {
    background-color: #f5f5f5;
    color: #050522;
    font-family: 'Times New Roman', Times, serif, sans-serif;
}

/* Main heading style */
h1 {
    color: #0e3e72;
    text-align: center;
    padding: 20px;

    /* Flexbox centering */
    display: flex;
    justify-content: center; /* Horizontally center */
    align-items: center; /* Vertically center */
}

header{
    /* height: 100vh; */
    background-size: cover;
    background-position: center;
    /* background-image: url(gatis-marcinkevics-a5uptAdUmjE-unsplash.jpg); */
    background-image: url(husnee-mubaarik-YvYL0uQcJIg-unsplash.jpg);

    /* min-height: 100px; */
}

.highlight{
    /* color: cornflowerblue; */
    color: lightblue;
    /* color: darkslateblue; */
}

/* div{ */
    /* height: 100vh; */
    /* background-size: cover; */
    /* background-position: center; */
    /* background-image: url(tito-rollis-s9vNTRLoWl8-unsplash.jpg); */
    /* background-image: url(adrien-olichon-MsEWtfunmEc-unsplash.jpg); */

    /* min-height: 100px; */
/* } */
/* Student information box */
.student-box {
    /* background-color: rgb(106, 195, 227); */
    /* border: 1px solid #ccc; */
    padding: 20px;
    margin: 20px;
}

```

```

.student-box {
    /* background-color: rgb(106, 195, 227); */
    /* border: 1px solid #ccc; */
    border: 1px solid #ffffff;
    padding: 30px 30px;
    color: aliceblue;
    transition: 0.5 ease;
    font-size: 18px;
    margin: 20px;
}

/* Blue text style */
.blue-color {
    color: blue;
}
h2{
    color: aliceblue;
}
h3{
    color: orangered;
}
/* Default styling for the image container */
.image-container {
    background-size: cover;
    background-position: center;
    background-image: url(imgpng.png);
}

/* Introduction paragraph */
.intro {
    text-align: center;
    padding: 20px;
}
.black-color {
    color: black;
}
.center-content {
    display: flex;
    justify-content: center;
    align-items: center;
    min-height: 100vh;
    /* background-color: #f5f5f5; Match the body background */
}
.mid-content {
    display: flex;
    justify-content: center;
    align-items: center top;
    min-height: 100vh;
    /* background-color: #f5f5f5; Match the body background */
}
.above-center {
    position: relative;
    top: -30px; /* Adjust this value to move the text above the center */
}
a {

```

```

        display: block;
        color: rgb(52, 52, 150);
        text-decoration: none;
        margin-bottom: 10px;
    }

```

## Appendix C: Arabic page HTTP code

```

<!DOCTYPE html>
<html>
<head>
    <link rel="stylesheet" type="text/css" href="styles.css">
    <meta charset="UTF-8">
</head>

<body>
<header>
<title>ENCS3320-My Tiny Webserver</title>

<div class="center-content">
    <h1 class="title">هذا موقعنا الصغير</h1>
    <h1 class="title highlight">اهلا بكم في مساق الشبكات</h1>
</div>
<!-- <h1><span class="black-color">Welcome to our course</span><samp>Computer Networks, This is a tiny webserver</samp></h1>
<p class="intro">Here are the group members and some information about them:</p> -->

<div class="center-content">
<div class="student-box">
    <h2>1201838 هزار ميخائيل </h2>
    <p>مشاريع: موقع ويب</p>
    <p>مهارات: لغة جافا, لغة سي, حل المسائل البرمجية</p>
    <p>هوايات: السباحة, ركوب الخيل</p>
</div>
</div>

<div class="center-content">
<div class="student-box">
    <h2>1201107 تالا فليفل </h2>
    <p>, مشاريع: صفحات ويب, مشاريع بنية المعلومات و الخوارزميات</p>
    <p>مهارات: لغة بايثون, لغة سي, لغة سى اس</p>
    <p>الهوايات: قراءة الكتب, الرسم</p>
</div>
</div>

<div class="center-content">
<div class="student-box">
    <h2>1203257 هاجر السوقى </h2>
    <p>مشاريع: تحليل الدوائر الكهربائية</p>
    <p>مهارات: لغة سى, لغة جافا</p>
</div>
</div>

```

```

<p>الهوايات: تسلق الجبال</p>
</div>
</div>

<div class="center-content">
    <p><a style="font-size: 50px;color: aliceblue" href="enPage.html" >اضغط على الرابط للاتصال الى الصفحة باللغة الانجليزية</a>
    </p>
</div>

<div class="center-content">
    <p><a style="font-size: 50px;color: aliceblue" href="https://www.w3schools.com/python/python_tuples.asp" >اضغط على الرابط للاتصال الى صفحة بيانات</a>
    </p>
</div>

</div>
</header>
<div class="image-container">
    <div class="center-content">
        <h3 class="above-center" style="font-size: 50px;">شكراً</h3>
    </div>
</div>

<header></header>
</div></body>
</html>

```

## Appendix D: English page HTTP code

```

<!DOCTYPE html>
<html>
<head>
    <link rel="stylesheet" type="text/css" href="styles.css">
</head>

<body>
<header>
<title>ENCS3320-My Tiny Webserver</title>

<div class="center-content">
    <h1 style="font-size: 40px;" class="title">Welcome to the Computer Networks Course</h1>
    <h1 style="font-size: 40px;" class="title highlight">This is Our Tiny Webserver</h1>
</div>
<!-- <h1><span class="black-color">Welcome to our course</span><samp class="blue-color"> Computer Networks, This is a tiny webserver</samp></h1>
<p class="intro">Here are the group members and some information about them:</p> -->

<div class="center-content">

```

```

<div class="student-box">
    <h2>Hazar Michael      1201838</h2>
    <p>Projects: Programming</p>
    <p>Skills: C, Java, Problem Solving, Data Base</p>
    <p>Hobbies: swimming, horseback riding </p>
</div>
</div>

<div class="center-content">
<div class="student-box">

    <h2>Tala Flaifel      1201107</h2>
    <p>Projects: Data structure projects </p>
    <p>Skills: HTML, CSS, Data structure, Python</p>
    <p>Hobbies: Reading books, Drawing</p>
</div>
</div>

<div class="center-content">
<div class="student-box">
    <h2>Hajar Alsouqi 1203257</h2>
    <p>Projects: Electrical, Programming</p>
    <p>Skills: Java, C</p>
    <p>Hobbies: Playing bowling, hiking </p>
</div>
</div>

<div class="center-content">
    <p><a style="font-size: 50px;color: aliceblue" href="arPage.html" >Click here to redirect to the page in arabic</a>
        </p>
</div>

<div class="center-content">
    <p><a style="font-size: 50px;color: aliceblue"
    href="https://www.w3schools.com/python/python_tuples.asp " >Click here to redirect to Python Tuple Page</a>
        </p>
</div>

</div>
</header>
<div class="image-container">
    <div class="center-content">
        <h3 class="above-center" style="font-size: 50px;">Thank You</h3>
    </div>
</div>
</header></header>
</body>
</html>

```

## Appendix E: Error404 HTTP code

```
<!DOCTYPE html>
<html>
<head>
    <title>Error 404</title>
</head>
<body>
    <h1 style="color: red;">HTTP/1.1 404 Not Found</h1>
    <p>The file is not found</p>
    <p><strong>Student 1: Hazar Michael</strong></p>
    <p><strong>ID:1201838</strong></p>
    <p><strong>Student 2:Tala Flaifel </strong></p>
    <p><strong>ID: 1201107</strong></p>
    <p><strong>Student 3: Hajar Alsouqi</strong></p>
    <p><strong>ID: 1203257</strong></p>
    <p><strong>Client Port: {client_port}</strong></p>
</body>
</html>
```