جامعة بيرزيت

BIRZEIT UNIVERSITY

# ENCS3320: Computer Networks
# 2023/2024
# Section 1
# Project #1 Report

|  |  |  |
|---|---|---|
| Prepared By: | Tala Dabbagh | **1200620** |
|  | Anwaar Qasem | **1200609** |
|  | Oday Khallaf | **1190546** |

Instructor: Abdalkarim Awad

Date: 21/12/2023

# Contents

Part one: ... 3
Part Two : ... 11
TCP Client Code : ... 11
TCP Server Code : ... 12
lock screen code : ... 15
Part Three : ... 16
Code Server.py: ... 16
main_en.html code : ... 19
main_ar.html code : ... 26
404.html code : ... 31
Style.css: ... 33
Style.css 404 : ... 42
main_en page : ... 45
... 45
Main_ ar  page : ... 47
Not Found Page: ... 49
Runs : ... 50
Cr: ... 51
So: ... 51
Rt: ... 52

# Part one:

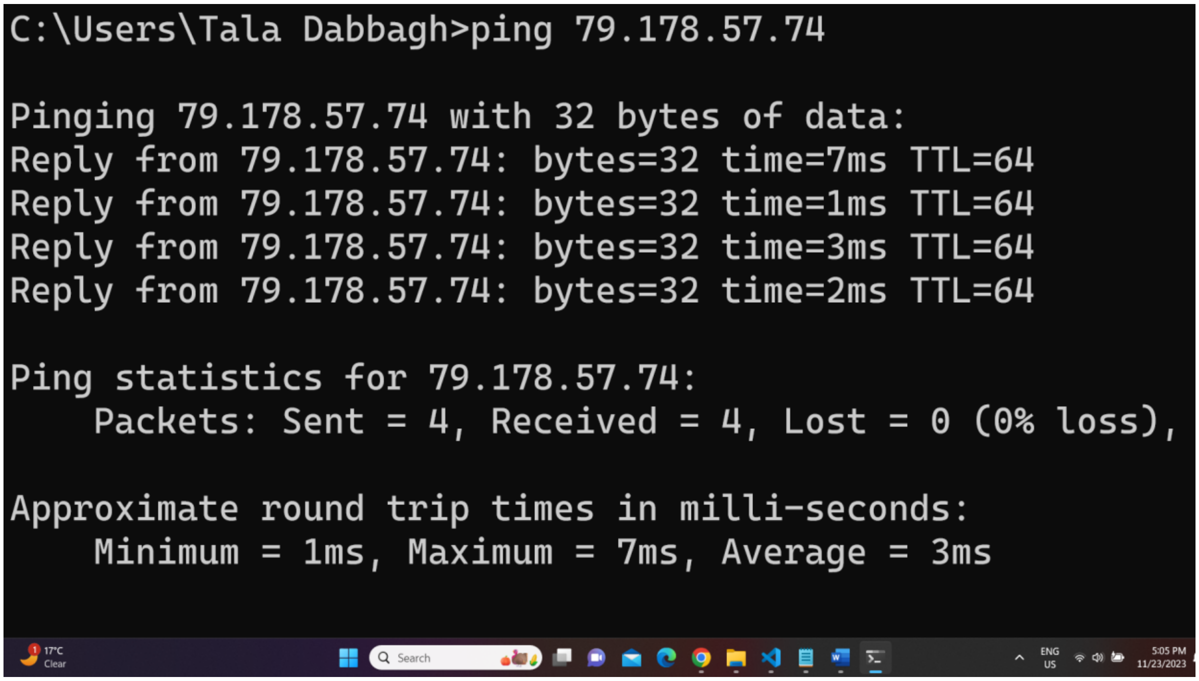
1. Ping:is a network utility that checks the connection between two devices by sending an Internet Control Message Protocol (ICMP) echo request and waiting for a response.

   Tracert: (or traceroute) traces the route that packets take from your computer to a destination, showing the network hops in between.

   Nslookup: is a command-line tool for querying Domain Name System (DNS) to obtain domain name or IP address information.

   Telnet: is a protocol that allows you to connect to and interact with a remote computer as if you were physically present, but it's less secure compared to SSH.re

   2.1.

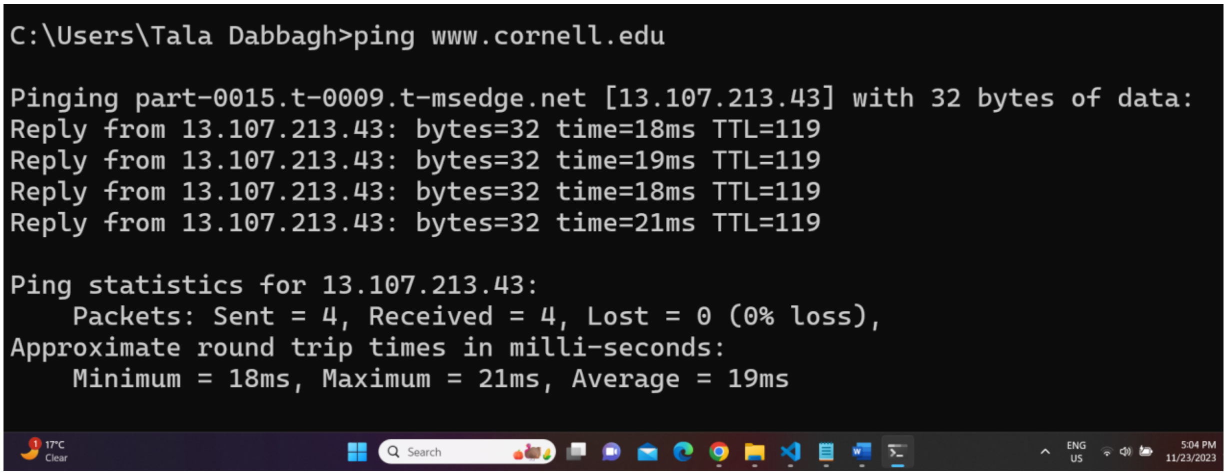


```
C:\Users\Tala Dabbagh>ping 79.178.57.74

Pinging 79.178.57.74 with 32 bytes of data:
Reply from 79.178.57.74: bytes=32 time=7ms TTL=64
Reply from 79.178.57.74: bytes=32 time=1ms TTL=64
Reply from 79.178.57.74: bytes=32 time=3ms TTL=64
Reply from 79.178.57.74: bytes=32 time=2ms TTL=64

Ping statistics for 79.178.57.74:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 7ms, Average = 3ms
```

The results indicate a successful communication with the destination, with all four packets sent receiving replies. The round-trip times, ranging from 1ms to 7ms, with an average of 3ms, suggest a low-latency connection, characteristic of a local or nearby network. The absence of packet loss (0%) further supports the reliability of the connection.

Based on the ping results, and my IP address 79.178.57.74 is assigned to my device in Palestine, it is evident that the response is not from the USA. The low round-trip times (1ms to 7ms) indicate a relatively low-latency connection, which is consistent with a local or nearby network. Therefore, the response is from Palestine, not the USA.

2.2



```
C:\Users\Tala Dabbagh>ping www.cornell.edu

Pinging part-0015.t-0009.t-msedge.net [13.107.213.43] with 32 bytes of data:
Reply from 13.107.213.43: bytes=32 time=18ms TTL=119
Reply from 13.107.213.43: bytes=32 time=19ms TTL=119
Reply from 13.107.213.43: bytes=32 time=18ms TTL=119
Reply from 13.107.213.43: bytes=32 time=21ms TTL=119

Ping statistics for 13.107.213.43:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 18ms, Maximum = 21ms, Average = 19ms
```

The ping command tested the connectivity to the domain www.cornell.edu, which resolved to myIP address 13.107.213.43. The results indicate successful communication with the destination, with all four packets sent receiving replies. The round-trip times range from 18ms to 21ms, with an average of 19ms, suggesting a moderate latency connection. The absence of packet loss (0%) further supports the reliability of the connection.

The distance from Palestine to the USA is 10,853 km.

To estimate the round-trip time (RTT) based on distance:

One-way latency = Distance / Propagation Speed

Let's assume the propagation speed is close to the speed of light in fiber optic cables, which is approximately 2×10^8 meters per second.

One-way latency = 10,853,000 (m) / 2×10^8 (m/s)

One-way latency ≈ 0.054265(s)

to estimate round trip time:

RTT = 2×One-way latency

RTT ≈ 2×0.054265 seconds

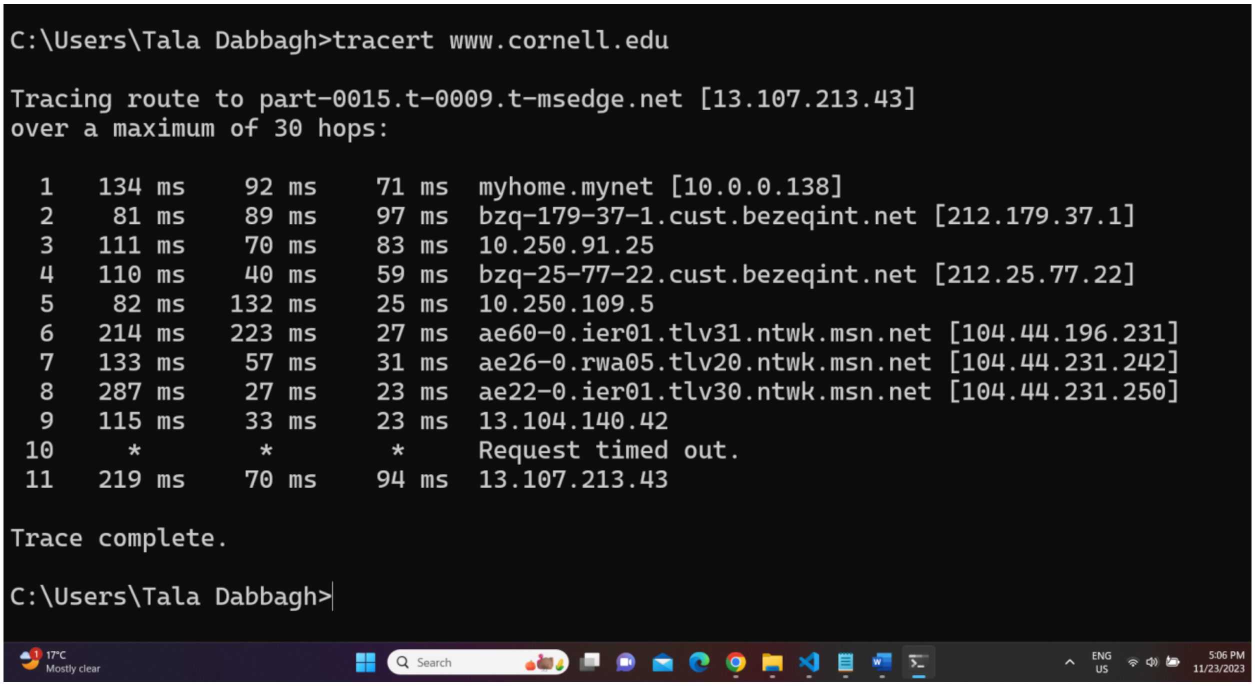
RTT ≈ 0.10853 seconds

RTT ≈ 108.53 milliseconds

So, based on the simplified model, the estimated round-trip time for a distance of 10,853 km is approximately 108.53 milliseconds, which is for sure not accurate but could give us an idea.

The observed round-trip time of 19 milliseconds is relatively low, indicating a quick response and low latency, which suggests that the destination server is physically closer than the USA to us.

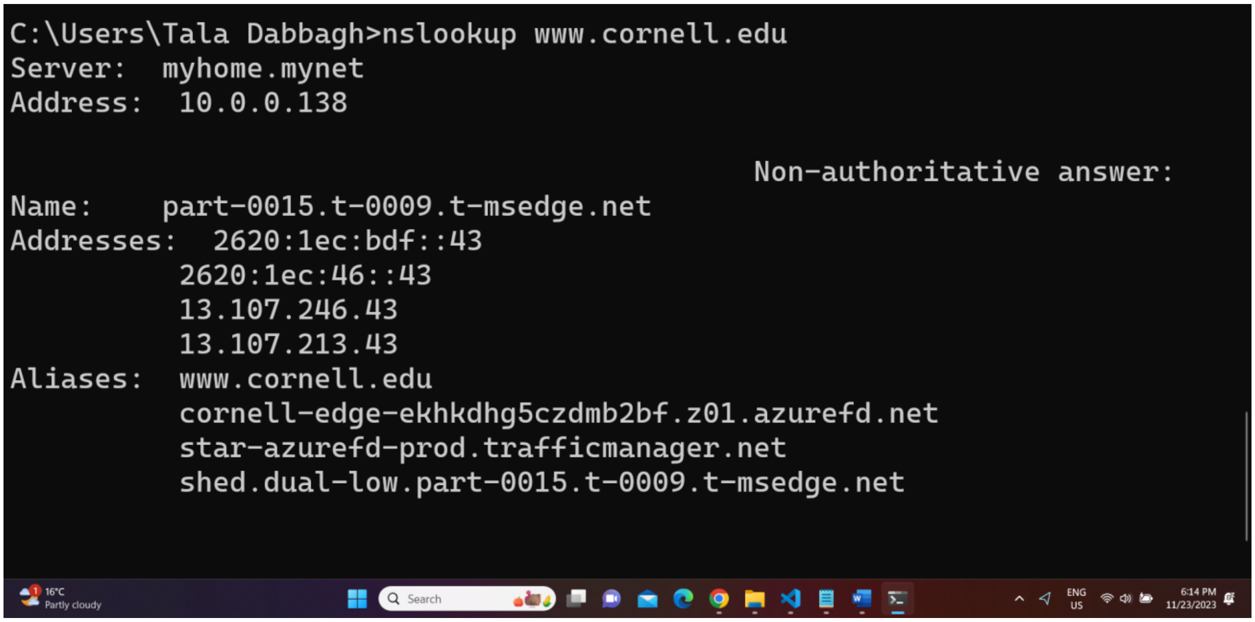
2.3.



```
C:\Users\Tala Dabbagh>tracert www.cornell.edu

Tracing route to part-0015.t-0009.t-msedge.net [13.107.213.43]
over a maximum of 30 hops:

  1   134 ms    92 ms    71 ms  myhome.mynet [10.0.0.138]
  2    81 ms    89 ms    97 ms  bzq-179-37-1.cust.bezeqint.net [212.179.37.1]
  3   111 ms    70 ms    83 ms  10.250.91.25
  4   110 ms    40 ms    59 ms  bzq-25-77-22.cust.bezeqint.net [212.25.77.22]
  5    82 ms   132 ms    25 ms  10.250.109.5
  6   214 ms   223 ms    27 ms  ae60-0.ier01.tlv31.ntwk.msn.net [104.44.196.231]
  7   133 ms    57 ms    31 ms  ae26-0.rwa05.tlv20.ntwk.msn.net [104.44.231.242]
  8   287 ms    27 ms    23 ms  ae22-0.ier01.tlv30.ntwk.msn.net [104.44.231.250]
  9   115 ms    33 ms    23 ms  13.104.140.42
 10     *        *        *     Request timed out.
 11   219 ms    70 ms    94 ms  13.107.213.43

Trace complete.

C:\Users\Tala Dabbagh>
```

The tracert output traces the route from a local computer to www.cornell.edu. It begins at the local router, passes through an ISP's network, navigates Microsoft's network, and reaches the destination server at 13.107.213.43. The varying response times indicate the duration for each hop. "Request timed out" suggests some routers didn't respond, which is common and may not indicate an issue.

2.4.



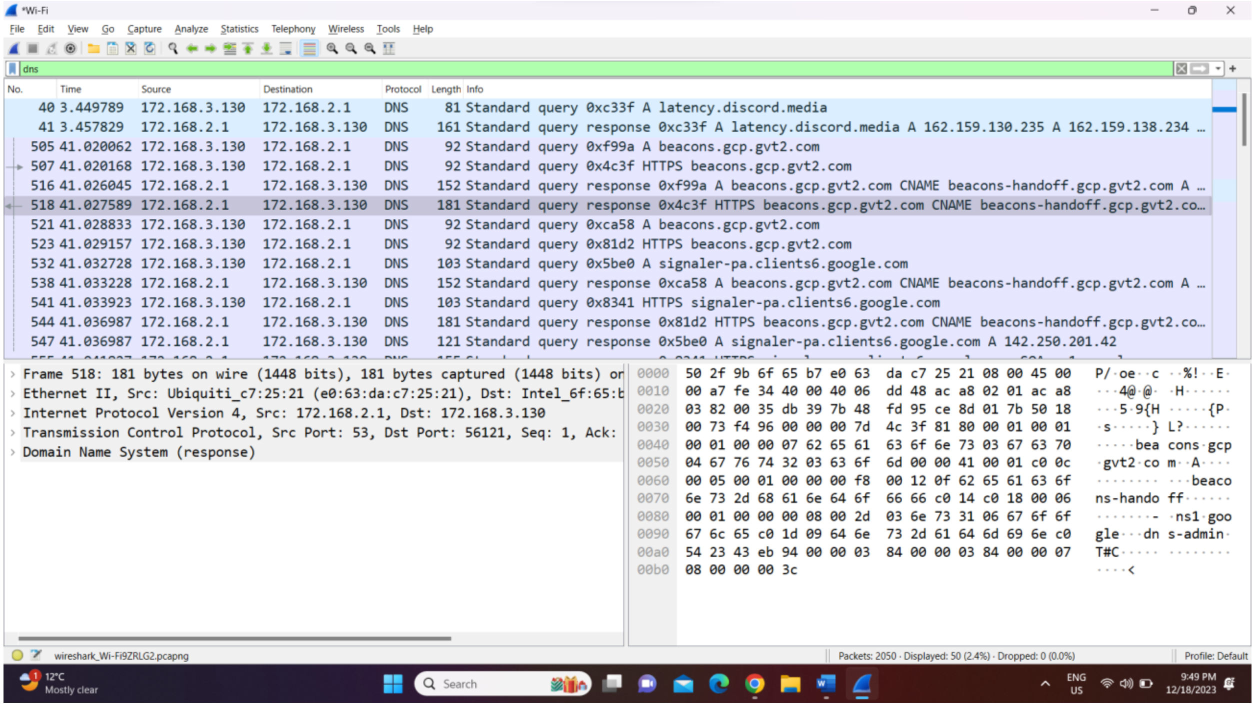
```
C:\Users\Tala Dabbagh>nslookup www.cornell.edu
Server:  myhome.mynet
Address:  10.0.0.138

                                            Non-authoritative answer:
Name:    part-0015.t-0009.t-msedge.net
Addresses:  2620:1ec:bdf::43
          2620:1ec:46::43
          13.107.246.43
          13.107.213.43
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
```

The nslookup output shows the domain resolution information for www.cornell.edu. The DNS server used is myhome.mynet with the IP address 10.0.0.138. The “non-authoritative answer” section provides the resolved IP addresses for www.cornell.edu:

- IPv6 Addresses: 2620:1ec:bdf::43, 2620:1ec:46::43
- IPv4 Addresses: 13.107.246.43, 13.107.213.43

Additionally, it shows “aliases” associated with www.cornell.edu, including cornell-edge-ekhkdhg5czdmb2bf.z01.azurefd.net and star-azurefd-prod.trafficmanager.net, suggesting the involvement of Microsoft Azure's content delivery network (CDN) infrastructure in serving Cornell University's website. The shed.dual-low.part-0015.t-0009.t-msedge.net alias points to the specific server handling the request within the Microsoft Edge server network.

3. General look


*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
dns
Search
12°C
Mostly clear
ENG
US
9:49 PM
12/18/2023

```
No.  Time        Source         Destination    Protocol Length Info
  40 3.449789    172.168.3.130  172.168.2.1    DNS       81 Standard query 0xc33f A latency.discord.media
  41 3.457829    172.168.2.1    172.168.3.130  DNS      161 Standard query response 0xc33f A latency.discord.media A 162.159.130.235 A 162.159.138.234 …
 505 41.020062   172.168.3.130  172.168.2.1    DNS       92 Standard query 0xf99a A beacons.gcp.gvt2.com
 507 41.020168   172.168.3.130  172.168.2.1    DNS       92 Standard query 0x4c3f HTTPS beacons.gcp.gvt2.com
 516 41.026045   172.168.2.1    172.168.3.130  DNS      152 Standard query response 0xf99a A beacons.gcp.gvt2.com CNAME beacons-handoff.gcp.gvt2.com A …
 518 41.027589   172.168.2.1    172.168.3.130  DNS      181 Standard query response 0x4c3f HTTPS beacons.gcp.gvt2.com CNAME beacons-handoff.gcp.gvt2.co…
 521 41.028833   172.168.3.130  172.168.2.1    DNS       92 Standard query 0xca58 A beacons.gcp.gvt2.com
 523 41.029157   172.168.3.130  172.168.2.1    DNS       92 Standard query 0x81d2 HTTPS beacons.gcp.gvt2.com
 532 41.032728   172.168.3.130  172.168.2.1    DNS      103 Standard query 0x5be0 A signaler-pa.clients6.google.com
 538 41.033228   172.168.2.1    172.168.3.130  DNS      152 Standard query response 0xca58 A beacons.gcp.gvt2.com CNAME beacons-handoff.gcp.gvt2.com A …
 541 41.033923   172.168.3.130  172.168.2.1    DNS      103 Standard query 0x8341 HTTPS signaler-pa.clients6.google.com
 544 41.036987   172.168.2.1    172.168.3.130  DNS      181 Standard query response 0x81d2 HTTPS beacons.gcp.gvt2.com CNAME beacons-handoff.gcp.gvt2.co…
 547 41.036987   172.168.2.1    172.168.3.130  DNS      121 Standard query response 0x5be0 A signaler-pa.clients6.google.com A 142.250.201.42
```

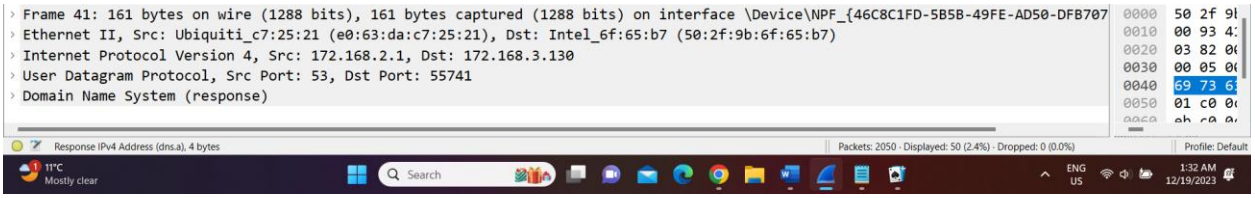
```
> Frame 518: 181 bytes on wire (1448 bits), 181 bytes captured (1448 bits) or
> Ethernet II, Src: Ubiquiti_c7:25:21 (e0:63:da:c7:25:21), Dst: Intel_6f:65:b
> Internet Protocol Version 4, Src: 172.168.2.1, Dst: 172.168.3.130
> Transmission Control Protocol, Src Port: 53, Dst Port: 56121, Seq: 1, Ack:
> Domain Name System (response)
```

```
0000  50 2f 9b 6f 65 b7 e0 63  da c7 25 21 08 00 45 00   P/·oe··c ··%!··E·
0010  00 a7 fe 34 40 00 40 06  dd 48 ac a8 02 01 ac a8   ···4@·@· ·H······
0020  03 82 00 35 db 39 7b 48  fd 95 ce 8d 01 7b 50 18   ···5·9{H ·····{P·
0030  00 73 f4 96 00 00 00 7d  4c 3f 81 80 00 01 00 01   ·s·····} L?······
0040  00 01 00 00 07 62 65 61  63 6f 6e 73 03 67 63 70   ·····bea cons·gcp
0050  04 67 76 74 32 03 63 6f  6d 00 00 41 00 01 c0 0c   ·gvt2·co m··A····
0060  00 05 00 01 00 00 00 f8  00 12 0f 62 65 61 63 6f   ········ ···beaco
0070  6e 73 2d 68 61 6e 64 6f  66 66 c0 14 c0 18 00 06   ns-hando ff······
0080  00 01 00 00 00 08 00 2d  03 6e 73 31 06 67 6f 6f   ·······- ·ns1·goo
0090  67 6c 65 c0 1d 09 64 6e  73 2d 61 64 6d 69 6e c0   gle···dn s-admin·
00a0  54 23 43 eb 94 00 00 03  84 00 00 03 84 00 00 07   T#C····· ········
00b0  08 00 00 00 3c                                     ····<
```

```
wireshark_Wi-Fi9ZRLG2.pcapng        Packets: 2050 · Displayed: 50 (2.4%) · Dropped: 0 (0.0%)        Profile: Default
```

- 172.168.2.1   =  represents the DNS Server of my wifi which is in student housing.
- 172.168.3.130  = represents the IP address of my device
- "Src Port" represents the standard DNS which is 53.
- The Destination Port is 55741, where the DNS response is being sent.

## Closer look( capture 1):

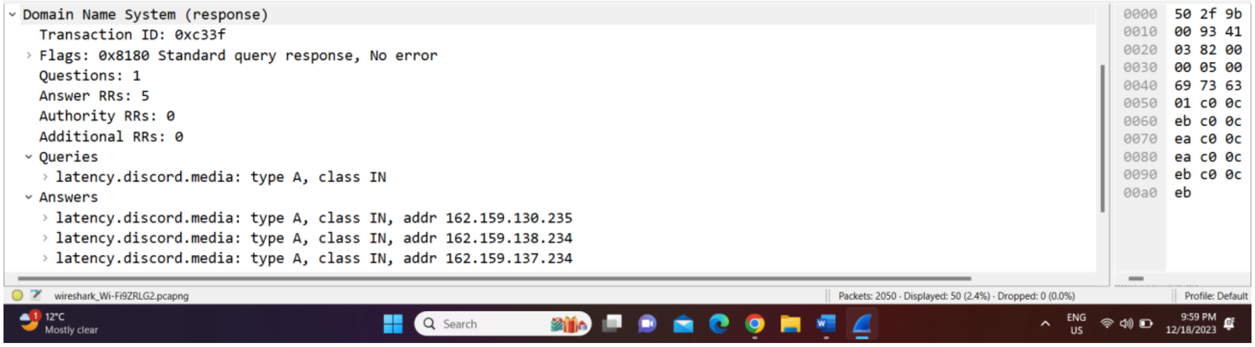


```
> Frame 41: 161 bytes on wire (1288 bits), 161 bytes captured (1288 bits) on interface \Device\NPF_{46C8C1FD-5B5B-49FE-AD50-DFB707
> Ethernet II, Src: Ubiquiti_c7:25:21 (e0:63:da:c7:25:21), Dst: Intel_6f:65:b7 (50:2f:9b:6f:65:b7)
> Internet Protocol Version 4, Src: 172.168.2.1, Dst: 172.168.3.130
> User Datagram Protocol, Src Port: 53, Dst Port: 55741
> Domain Name System (response)
```

```
0000  50 2f 9b
0010  00 93 4:
0020  03 82 0(
0030  00 05 0(
0040  69 73 6:
0050  01 c0 0(
```

```
Response IPv4 Address (dns.a), 4 bytes        Packets: 2050 · Displayed: 50 (2.4%) · Dropped: 0 (0.0%)        Profile: Default
```

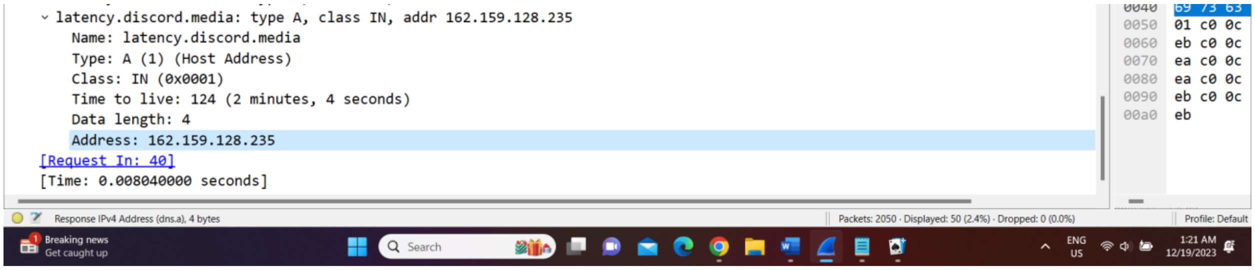
In this DNS capture (Frame 41), a 161-byte packet was sent from source IP 172.168.2.1 to destination IP 172.168.3.130 using UDP ports source 53 and destination 55741. The packet is a DNS response, originating from MAC address e0:63:da:c7:25:21. And, the network packet is using Internet Protocol version 4 (IPv4).



```
˅ Domain Name System (response)
    Transaction ID: 0xc33f
  › Flags: 0x8180 Standard query response, No error
    Questions: 1
    Answer RRs: 5
    Authority RRs: 0
    Additional RRs: 0
  ˅ Queries
    › latency.discord.media: type A, class IN
  ˅ Answers
    › latency.discord.media: type A, class IN, addr 162.159.130.235
    › latency.discord.media: type A, class IN, addr 162.159.138.234
    › latency.discord.media: type A, class IN, addr 162.159.137.234
```

```
0000  50 2f 9b
0010  00 93 41
0020  03 82 00
0030  00 05 00
0040  69 73 63
0050  01 c0 0c
0060  eb c0 0c
0070  ea c0 0c
0080  ea c0 0c
0090  eb c0 0c
00a0  eb
```

wireshark_Wi-Fi9ZRLG2.pcapng | Packets: 2050 · Displayed: 50 (2.4%) · Dropped: 0 (0.0%) | Profile: Default



```
    ˅ latency.discord.media: type A, class IN, addr 162.159.128.235
        Name: latency.discord.media
        Type: A (1) (Host Address)
        Class: IN (0x0001)
        Time to live: 124 (2 minutes, 4 seconds)
        Data length: 4
        Address: 162.159.128.235
  [Request In: 40]
  [Time: 0.008040000 seconds]
```

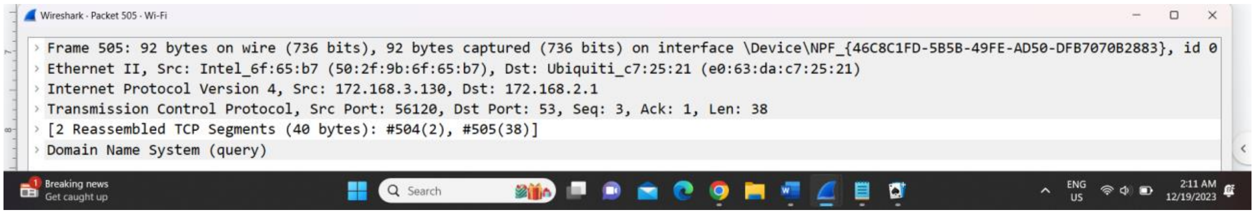
```
0040  69 73 63
0050  01 c0 0c
0060  eb c0 0c
0070  ea c0 0c
0080  ea c0 0c
0090  eb c0 0c
00a0  eb
```

Response IPv4 Address (dns.a), 4 bytes | Packets: 2050 · Displayed: 50 (2.4%) · Dropped: 0 (0.0%) | Profile: Default

Domain Name System (DNS) Response:

This DNS response contains information related to the DNS query made between the source and destination IP addresses. Every packet is given a transaction ID, in this case it’s “0xc33f”. The flags indicate a successful standard query response. The server can perform queries, and there's one question and five answer resource records for "latency.discord.media" with associated IP addresses. Each answer has a Time to Live of 124 seconds. The response, which took 0.00804 seconds, shows no errors in the query process.

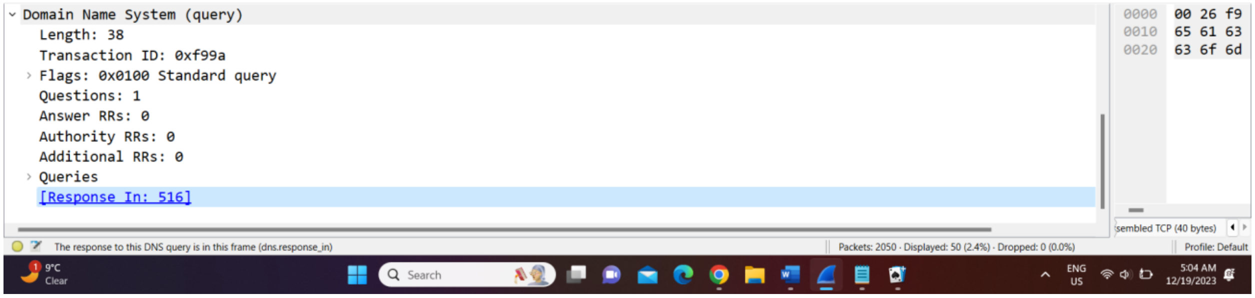
Closer look( capture 2):



Wireshark · Packet 505 · Wi-Fi

```
› Frame 505: 92 bytes on wire (736 bits), 92 bytes captured (736 bits) on interface \Device\NPF_{46C8C1FD-5B5B-49FE-AD50-DFB7070B2883}, id 0
› Ethernet II, Src: Intel_6f:65:b7 (50:2f:9b:6f:65:b7), Dst: Ubiquiti_c7:25:21 (e0:63:da:c7:25:21)
› Internet Protocol Version 4, Src: 172.168.3.130, Dst: 172.168.2.1
› Transmission Control Protocol, Src Port: 56120, Dst Port: 53, Seq: 3, Ack: 1, Len: 38
› [2 Reassembled TCP Segments (40 bytes): #504(2), #505(38)]
› Domain Name System (query)
```

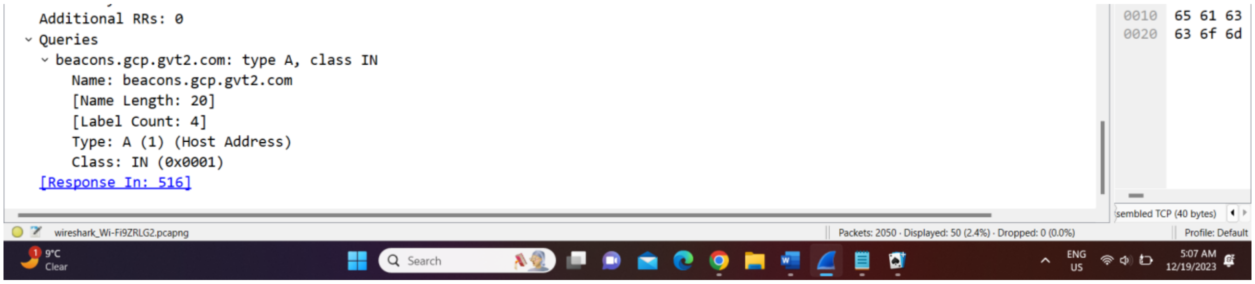
In this DNS capture (Frame 505), a 92-byte packet was sent from source IP 172.168.3.130 (my computer)  to destination 172.168.2.1 using Transmission Control Protocol (TCP) source port 56120, destination port 53, a sequence number of 3, acknowledgment number of 1, and a length of 38 bytes.



```
˅ Domain Name System (query)
    Length: 38
    Transaction ID: 0xf99a
  › Flags: 0x0100 Standard query
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  › Queries
    [Response In: 516]
```

```
0000  00 26 f9
0010  65 61 63
0020  63 6f 6d
```

```
The response to this DNS query is in this frame (dns.response_in)    Packets: 2050 · Displayed: 50 (2.4%) · Dropped: 0 (0.0%)    Profile: Default
```

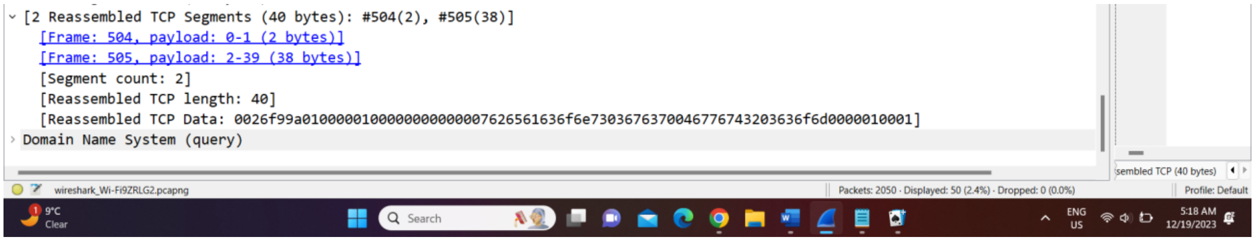


```
    Additional RRs: 0
  ˅ Queries
    ˅ beacons.gcp.gvt2.com: type A, class IN
        Name: beacons.gcp.gvt2.com
        [Name Length: 20]
        [Label Count: 4]
        Type: A (1) (Host Address)
        Class: IN (0x0001)
    [Response In: 516]
```

```
0010  65 61 63
0020  63 6f 6d
```

```
wireshark_Wi-Fi9ZRLG2.pcapng    Packets: 2050 · Displayed: 50 (2.4%) · Dropped: 0 (0.0%)    Profile: Default
```

This DNS query has a length of 38 bytes, and a transaction ID of 0xf99a. The flags indicate a standard query, with one question, zero answer resource records (RRs), zero authority RRs, and zero additional RRs. It requests the IPv4 address for "beacons.gcp.gvt2.com" using the standard query format within the IN class. The response is expected to be in Frame 516.



```
˅ [2 Reassembled TCP Segments (40 bytes): #504(2), #505(38)]
    [Frame: 504, payload: 0-1 (2 bytes)]
    [Frame: 505, payload: 2-39 (38 bytes)]
    [Segment count: 2]
    [Reassembled TCP length: 40]
    [Reassembled TCP Data: 0026f99a01000001000000000000076265616364f6e7303676370046776743203636f6d0000010001]
› Domain Name System (query)
```

```
wireshark_Wi-Fi9ZRLG2.pcapng    Packets: 2050 · Displayed: 50 (2.4%) · Dropped: 0 (0.0%)    Profile: Default
```

This information indicates the reconstruction of TCP segments within the captured frames. Specifically, two TCP segments from Frame 504 (2 bytes) and Frame 505 (38 bytes) have been combined, forming a total reconstructed TCP length of 40 bytes. The hexadecimal data at the end represents the content of the rebuilt TCP data, which is likely containing encoded information related to a DNS query.

# Part Two :

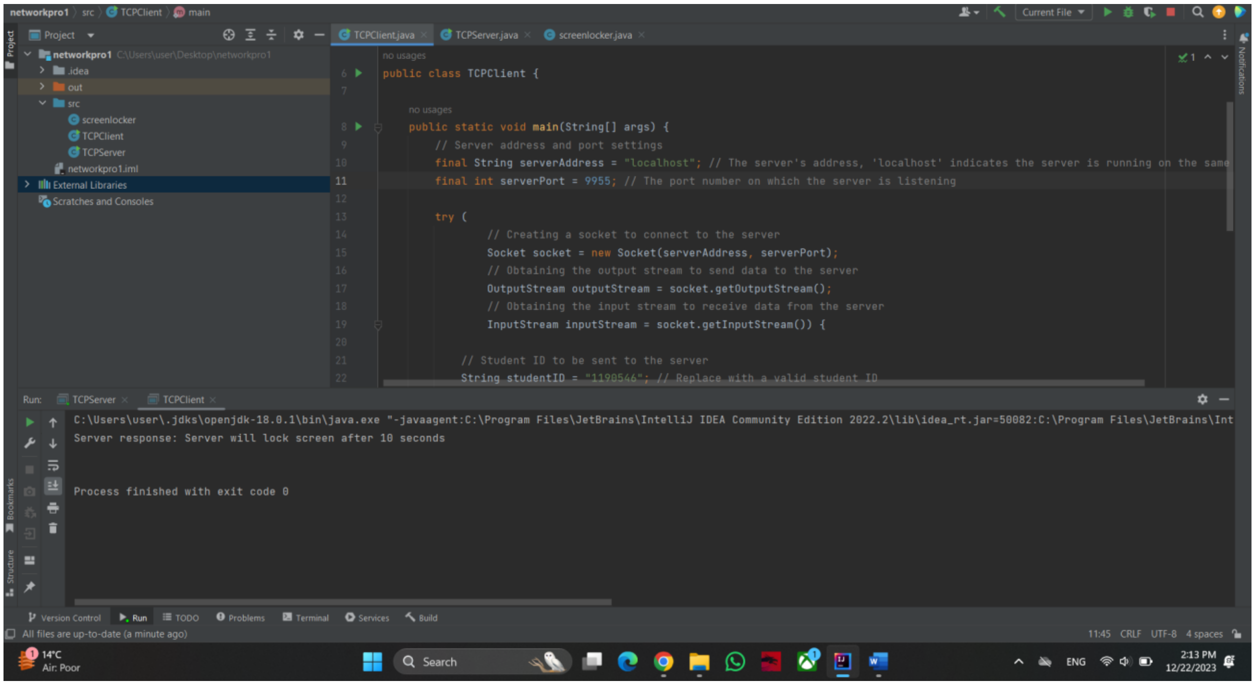
This is the TCP client and server code to implement the request:

## TCP Client Code :

This code is a TCP client. It is designed to connect to a TCP server running on the local machine (localhost) at port 9955. The client establishes a socket connection to the server and then obtains both output and input streams for sending and receiving data, respectively.
The client sends a student ID, which is a string "1190546", to the server via the output stream. It then waits for a response from the server, reading it through the input stream into a buffer. Once the response is received, the client reads the bytes from the buffer up to the number of bytes read, converts them into a string, and prints out the server's response to the console.

```java
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.Socket;

public class TCPClient {

    public static void main(String[] args) {
        // Server address and port settings
        final String serverAddress = "localhost"; // The server's address,
'localhost' indicates the server is running on the same machine
        final int serverPort = 9955; // The port number on which the server
is listening

        try (
                // Creating a socket to connect to the server
                Socket socket = new Socket(serverAddress, serverPort);
                // Obtaining the output stream to send data to the server
                OutputStream outputStream = socket.getOutputStream();
                // Obtaining the input stream to receive data from the server
                InputStream inputStream = socket.getInputStream()) {

            // Student ID to be sent to the server
            String studentID = "1190546"; // Replace with a valid student ID

            // Sending the student ID to the server
            outputStream.write(studentID.getBytes());

            // Buffer to store the response from the serverك
            byte[] buffer = new byte[1024];
            // Reading the server's response;
            int bytesRead = inputStream.read(buffer);
            // Print out the server's response
            System.out.println("Server response: " + new String(buffer, 0,
bytesRead));

        } catch (IOException e) {
            e.printStackTrace();
        }
    }


}
```

Run:



```
no usages
public class TCPClient {

    no usages
    public static void main(String[] args) {
        // Server address and port settings
        final String serverAddress = "localhost"; // The server's address, 'localhost' indicates the server is running on the same
        final int serverPort = 9955; // The port number on which the server is listening

        try (
                // Creating a socket to connect to the server
                Socket socket = new Socket(serverAddress, serverPort);
                // Obtaining the output stream to send data to the server
                OutputStream outputStream = socket.getOutputStream();
                // Obtaining the input stream to receive data from the server
                InputStream inputStream = socket.getInputStream()) {

            // Student ID to be sent to the server
            String studentID = "1190546"; // Replace with a valid student ID
```

```
C:\Users\user\.jdks\openjdk-18.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.2\lib\idea_rt.jar=50082:C:\Program Files\JetBrains\Int
Server response: Server will lock screen after 10 seconds


Process finished with exit code 0
```

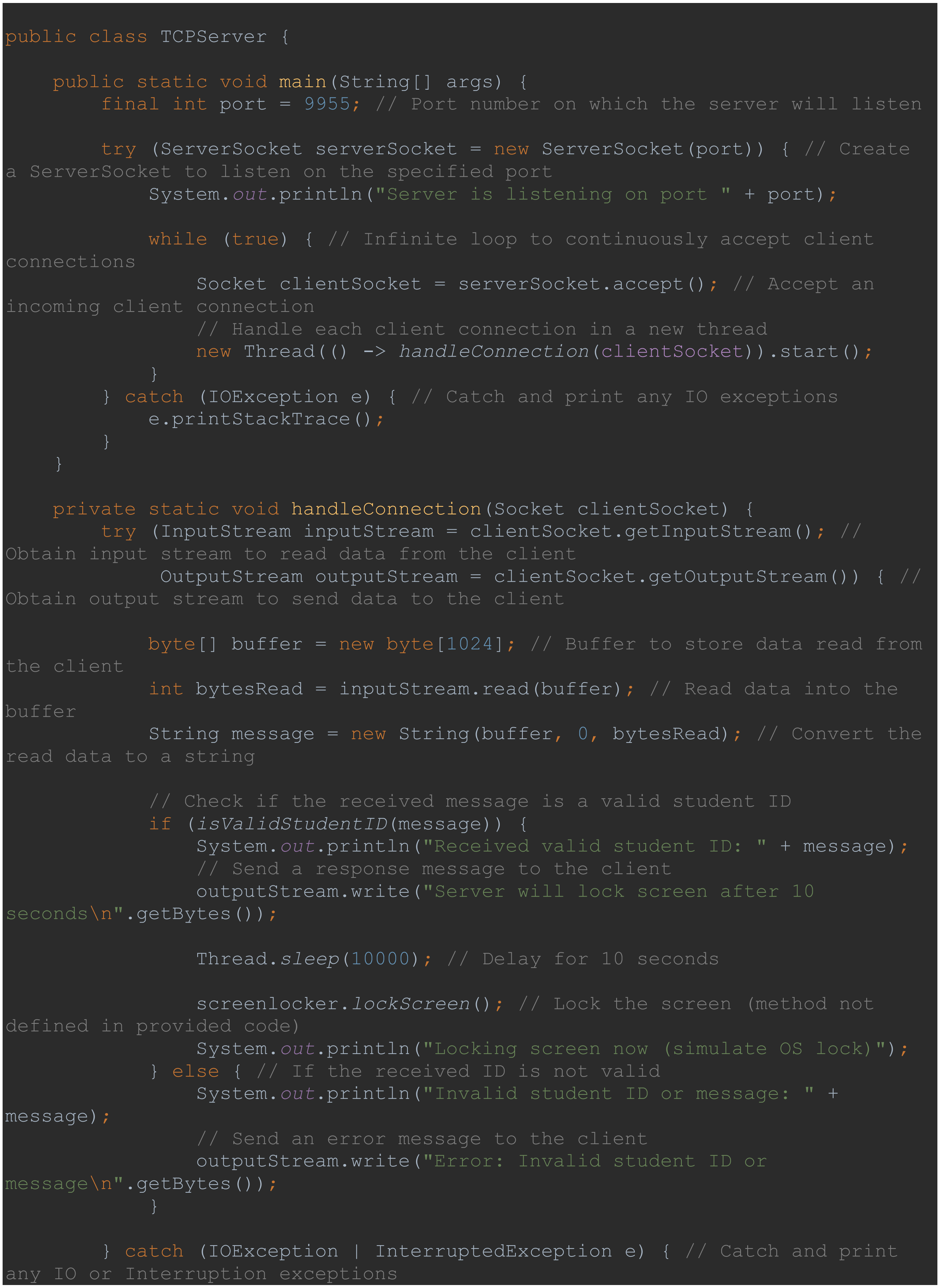
# TCP Server Code :

This Java code represents a TCP server that listens for client connections on a specified port, in this case, port 9955. The server is designed to run indefinitely, constantly listening for new connections. Upon accepting a connection from a client, it spawns a new thread to handle the client-server communication without blocking other incoming connections.

Each client connection is processed by the handle Connection method. This method reads data from the client through an input stream into a buffer, expecting to receive a student ID as a string. The received ID is checked against a list of valid IDs in the isValidStudentID method.

If the ID is valid, the server prints a confirmation message to its console, sends a response to the client indicating that the screen will lock after 10 seconds, and then simulates a screen lock (using a hypothetical screenlocker.lockScreen() method not implemented in this snippet). If the ID is not valid, the server sends an error message back to the client.

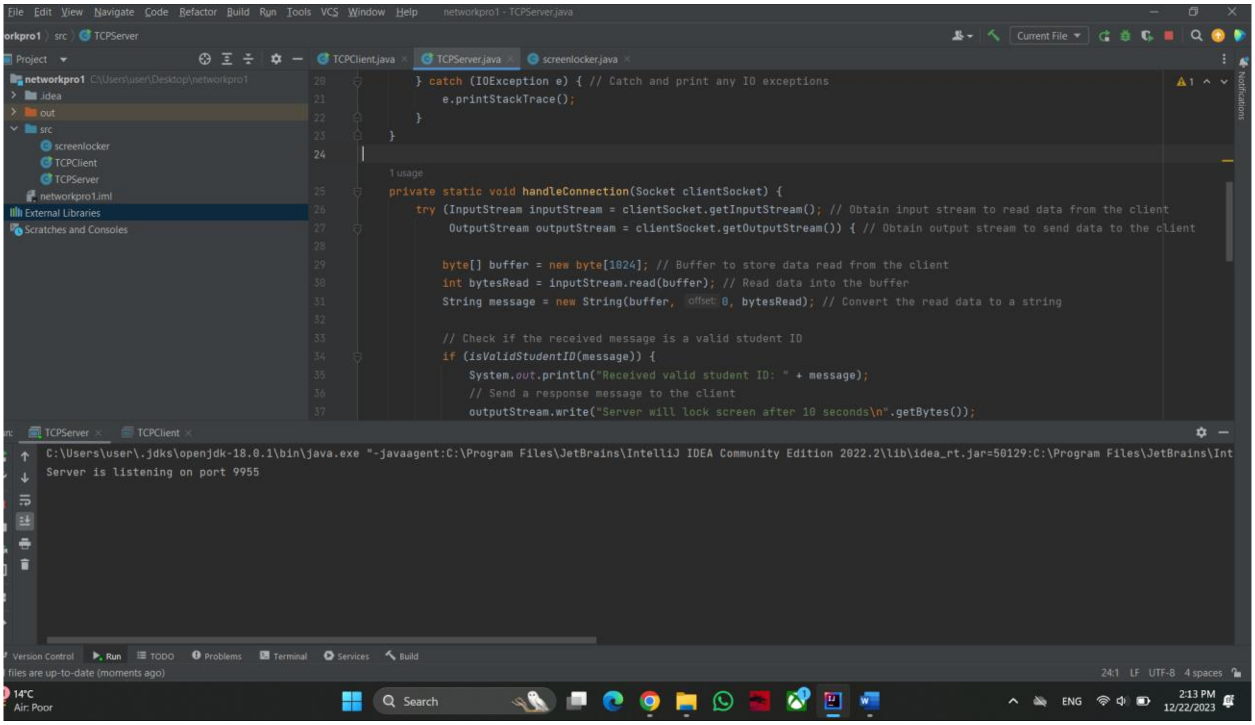
```
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.ServerSocket;
import java.net.Socket;
```



```java
public class TCPServer {

    public static void main(String[] args) {
        final int port = 9955; // Port number on which the server will listen

        try (ServerSocket serverSocket = new ServerSocket(port)) { // Create
a ServerSocket to listen on the specified port
            System.out.println("Server is listening on port " + port);

            while (true) { // Infinite loop to continuously accept client
connections
                Socket clientSocket = serverSocket.accept(); // Accept an
incoming client connection
                // Handle each client connection in a new thread
                new Thread(() -> handleConnection(clientSocket)).start();
            }
        } catch (IOException e) { // Catch and print any IO exceptions
            e.printStackTrace();
        }
    }

    private static void handleConnection(Socket clientSocket) {
        try (InputStream inputStream = clientSocket.getInputStream(); //
Obtain input stream to read data from the client
             OutputStream outputStream = clientSocket.getOutputStream()) { //
Obtain output stream to send data to the client

            byte[] buffer = new byte[1024]; // Buffer to store data read from
the client
            int bytesRead = inputStream.read(buffer); // Read data into the
buffer
            String message = new String(buffer, 0, bytesRead); // Convert the
read data to a string

            // Check if the received message is a valid student ID
            if (isValidStudentID(message)) {
                System.out.println("Received valid student ID: " + message);
                // Send a response message to the client
                outputStream.write("Server will lock screen after 10
seconds\n".getBytes());

                Thread.sleep(10000); // Delay for 10 seconds

                screenlocker.lockScreen(); // Lock the screen (method not
defined in provided code)
                System.out.println("Locking screen now (simulate OS lock)");
            } else { // If the received ID is not valid
                System.out.println("Invalid student ID or message: " +
message);
                // Send an error message to the client
                outputStream.write("Error: Invalid student ID or
message\n".getBytes());
            }

        } catch (IOException | InterruptedException e) { // Catch and print
any IO or Interruption exceptions
```

```java
            e.printStackTrace();
        }
    }

    private static boolean isValidStudentID(String studentID) {
        // List of valid student IDs
        String[] validStudentIDs = {"1190546", "1190547", "1190548"}; //
Expand the list as needed

        // Check if the received student ID is in the list of valid IDs
        for (String validID : validStudentIDs) {
            if (studentID.equals(validID)) {
                return true; // The ID is valid
            }
        }

        return false; // The ID is invalid
    }
}
```

Run:



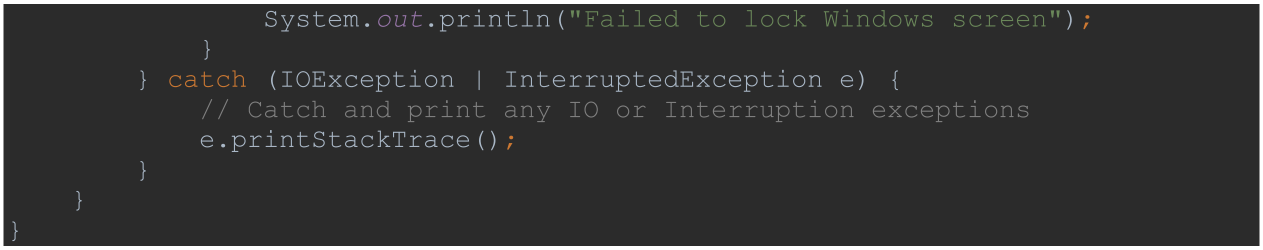
```java
        } catch (IOException e) { // Catch and print any IO exceptions
            e.printStackTrace();
        }
    }

    1 usage
    private static void handleConnection(Socket clientSocket) {
        try (InputStream inputStream = clientSocket.getInputStream(); // Obtain input stream to read data from the client
             OutputStream outputStream = clientSocket.getOutputStream()) { // Obtain output stream to send data to the client

            byte[] buffer = new byte[1024]; // Buffer to store data read from the client
            int bytesRead = inputStream.read(buffer); // Read data into the buffer
            String message = new String(buffer, offset: 0, bytesRead); // Convert the read data to a string

            // Check if the received message is a valid student ID
            if (isValidStudentID(message)) {
                System.out.println("Received valid student ID: " + message);
                // Send a response message to the client
                outputStream.write("Server will lock screen after 10 seconds\n".getBytes());
```

```
C:\Users\user\.jdks\openjdk-18.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.2\lib\idea_rt.jar=50129:C:\Program Files\JetBrains\Int
Server is listening on port 9955
```

# lock screen code :

This Java code snippet defines a class named screen locker with a static method lock Screen() that locks the user's screen. The method determines the operating system of the user's machine by checking the system property os.name. If the operating system is identified as Windows, it proceeds to lock the screen by calling another static method within the class, lockWindowsScreen().
The lockWindowsScreen() method uses the Runtime.getRuntime().exec() function to execute a system-specific command, rundll32.exe user32.dll,LockWorkStation, which is the command in Windows for locking the workstation/screen. After executing the command, the method waits for the process to complete and checks the exit code to determine if the operation was successful. An exit code of 0 indicates success, and it prints a confirmation message to the console. If the process does not succeed, it prints an error message.

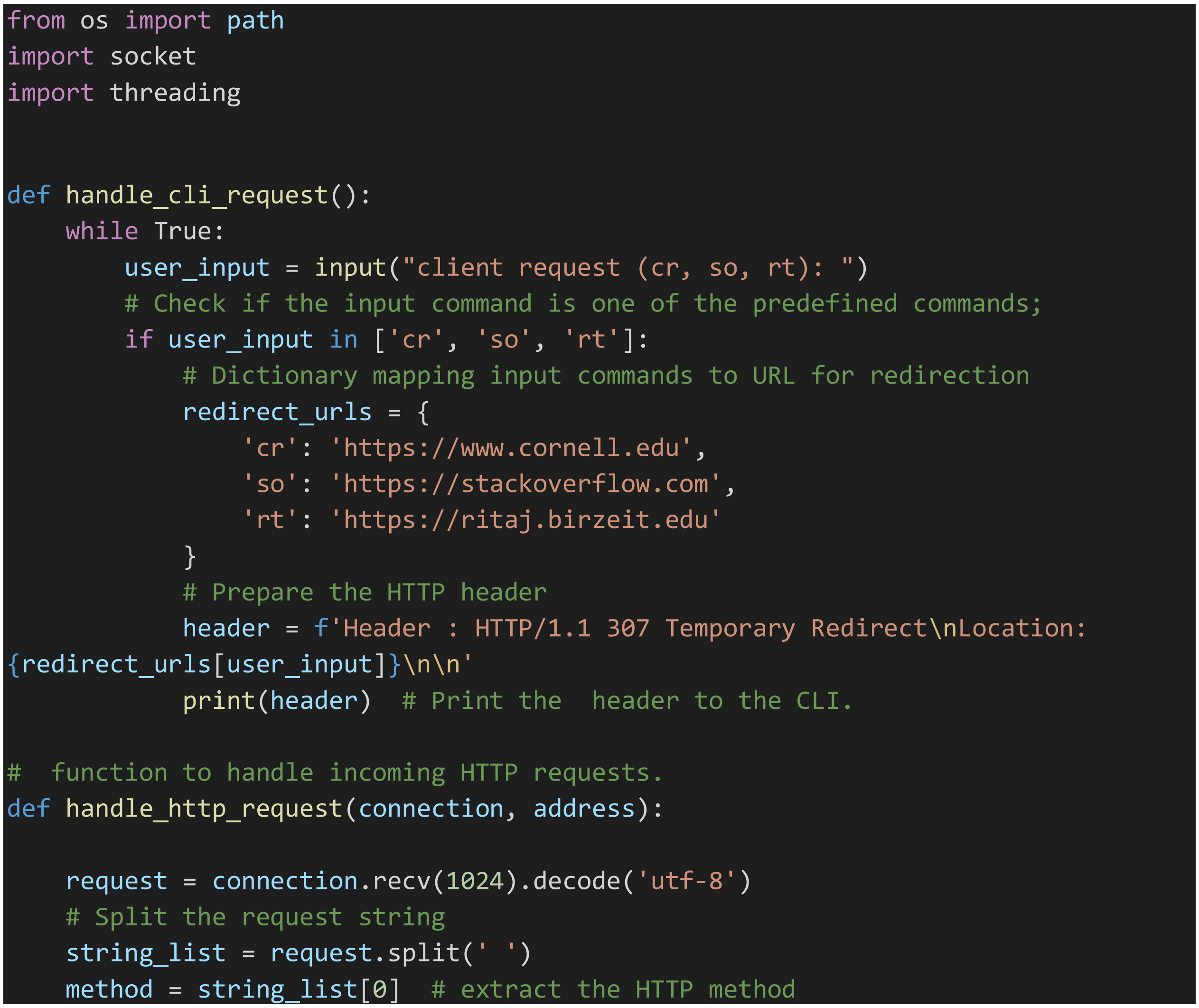
```
import java.io.IOException;

public class screenlocker {

    public static void lockScreen() {
        // Retrieve the name of the operating system
        String os = System.getProperty("os.name").toLowerCase();

        // Check if the operating system is Windows
        if (os.contains("win")) {
            // If it's Windows, lock the Windows screen
            lockWindowsScreen();
        } else {
            // If the OS is not Windows, print a message indicating that
screen lock is not implemented
            System.out.println("Screen lock not implemented for this OS");
        }
    }

    private static void lockWindowsScreen() {
        try {
            // Execute the command to lock the Windows screen using
rundll32.exe
            Process process = Runtime.getRuntime().exec("rundll32.exe
user32.dll,LockWorkStation");
            // Wait for the command to complete and get the exit code
            int exitCode = process.waitFor();

            // Check if the command was successful (exit code 0)
            if (exitCode == 0) {
                System.out.println("Windows screen locked");
            } else {
                // If the command failed, print an error message
```



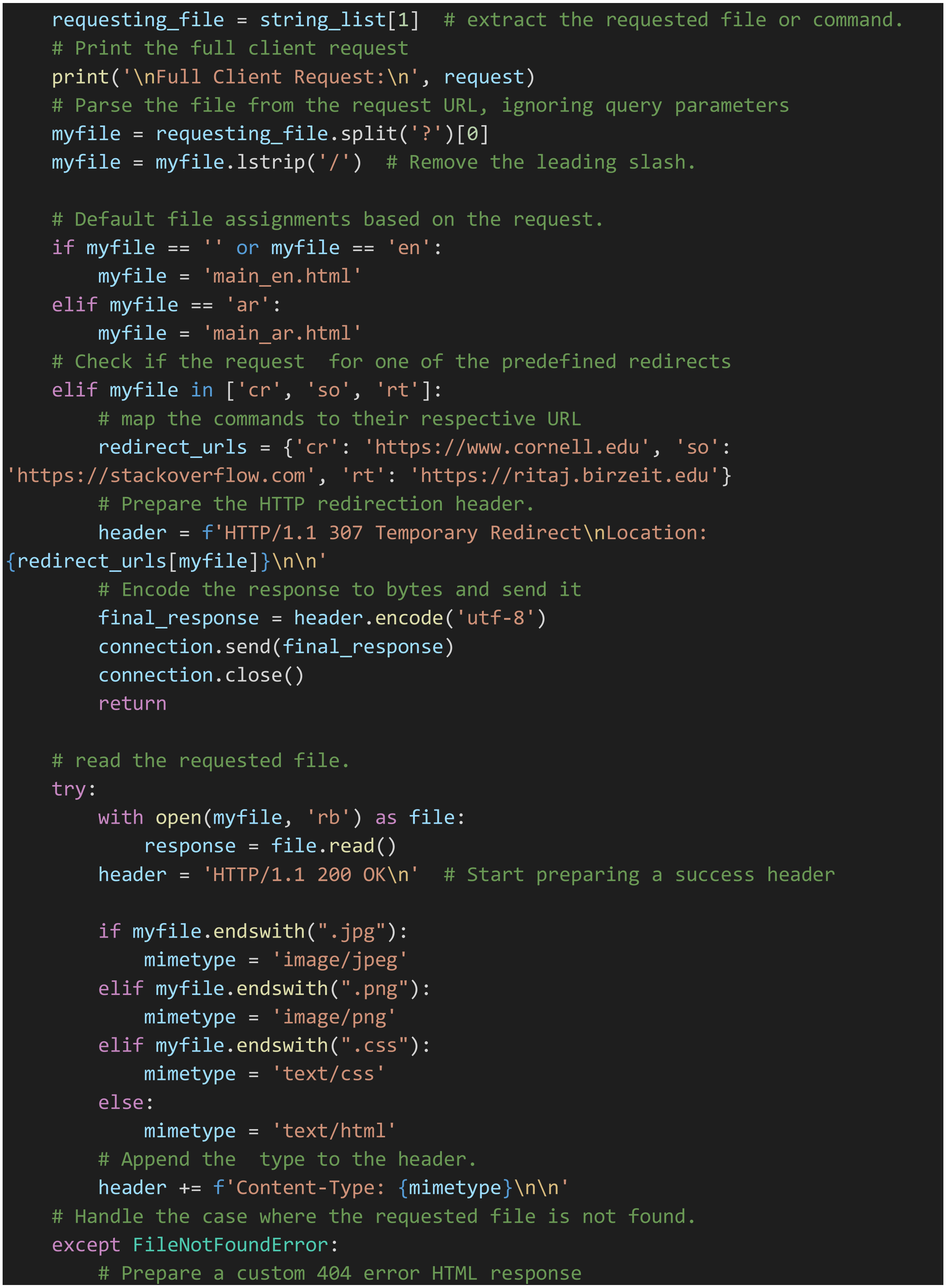
```
                System.out.println("Failed to lock Windows screen");
            }
        } catch (IOException | InterruptedException e) {
            // Catch and print any IO or Interruption exceptions
            e.printStackTrace();
        }
    }
}
```

## Part Three :

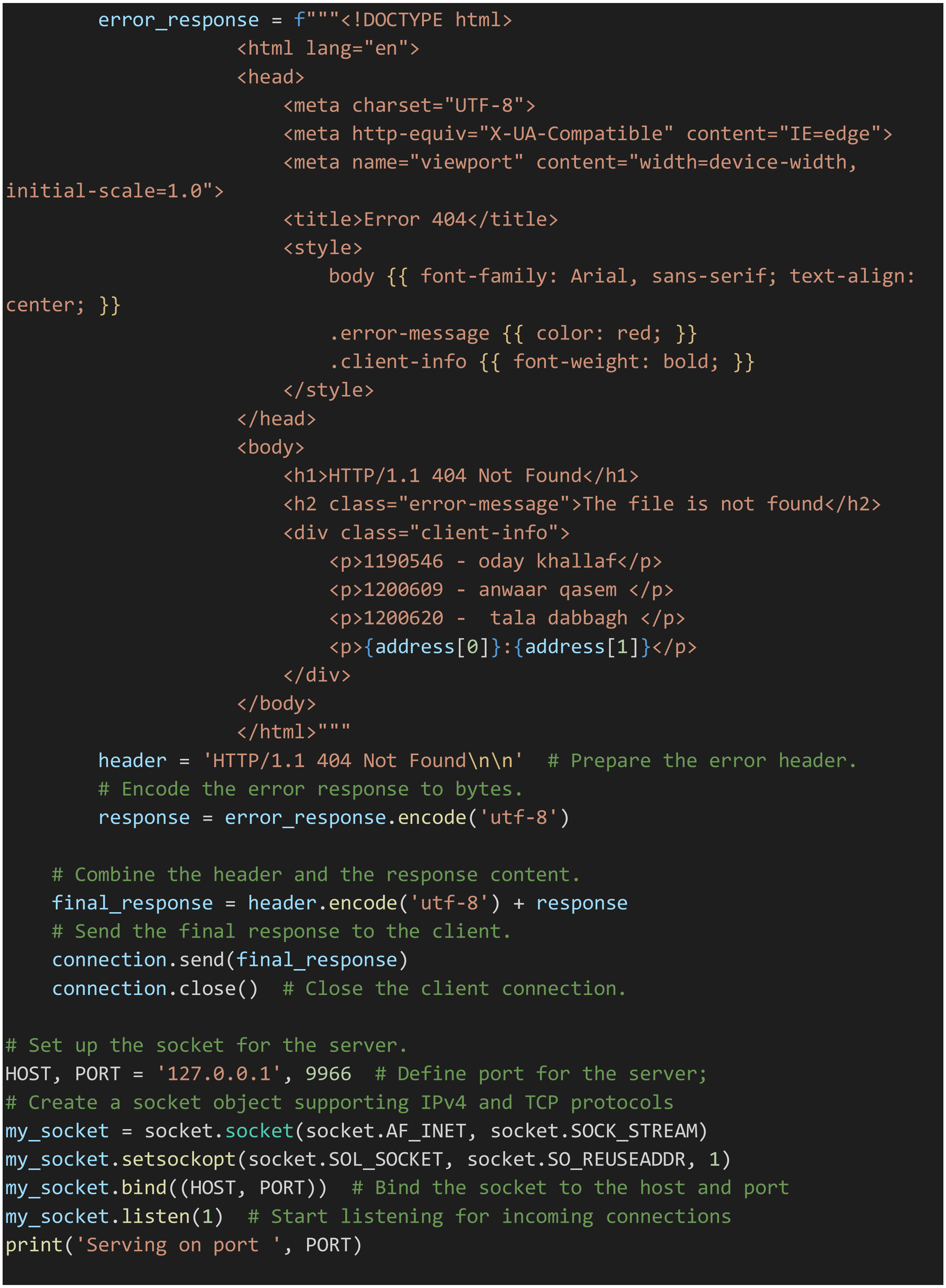
### Code Server.py:



```
from os import path
import socket
import threading


def handle_cli_request():
    while True:
        user_input = input("client request (cr, so, rt): ")
        # Check if the input command is one of the predefined commands;
        if user_input in ['cr', 'so', 'rt']:
            # Dictionary mapping input commands to URL for redirection
            redirect_urls = {
                'cr': 'https://www.cornell.edu',
                'so': 'https://stackoverflow.com',
                'rt': 'https://ritaj.birzeit.edu'
            }
            # Prepare the HTTP header
            header = f'Header : HTTP/1.1 307 Temporary Redirect\nLocation: 
{redirect_urls[user_input]}\n\n'
            print(header)  # Print the  header to the CLI.

#  function to handle incoming HTTP requests.
def handle_http_request(connection, address):

    request = connection.recv(1024).decode('utf-8')
    # Split the request string
    string_list = request.split(' ')
    method = string_list[0]  # extract the HTTP method
```



```
    requesting_file = string_list[1]  # extract the requested file or command.
    # Print the full client request
    print('\nFull Client Request:\n', request)
    # Parse the file from the request URL, ignoring query parameters
    myfile = requesting_file.split('?')[0]
    myfile = myfile.lstrip('/')  # Remove the leading slash.

    # Default file assignments based on the request.
    if myfile == '' or myfile == 'en':
        myfile = 'main_en.html'
    elif myfile == 'ar':
        myfile = 'main_ar.html'
    # Check if the request  for one of the predefined redirects
    elif myfile in ['cr', 'so', 'rt']:
        # map the commands to their respective URL
        redirect_urls = {'cr': 'https://www.cornell.edu', 'so':
'https://stackoverflow.com', 'rt': 'https://ritaj.birzeit.edu'}
        # Prepare the HTTP redirection header.
        header = f'HTTP/1.1 307 Temporary Redirect\nLocation:
{redirect_urls[myfile]}\n\n'
        # Encode the response to bytes and send it
        final_response = header.encode('utf-8')
        connection.send(final_response)
        connection.close()
        return

    # read the requested file.
    try:
        with open(myfile, 'rb') as file:
            response = file.read()
        header = 'HTTP/1.1 200 OK\n'  # Start preparing a success header

        if myfile.endswith(".jpg"):
            mimetype = 'image/jpeg'
        elif myfile.endswith(".png"):
            mimetype = 'image/png'
        elif myfile.endswith(".css"):
            mimetype = 'text/css'
        else:
            mimetype = 'text/html'
        # Append the  type to the header.
        header += f'Content-Type: {mimetype}\n\n'
    # Handle the case where the requested file is not found.
    except FileNotFoundError:
        # Prepare a custom 404 error HTML response
```

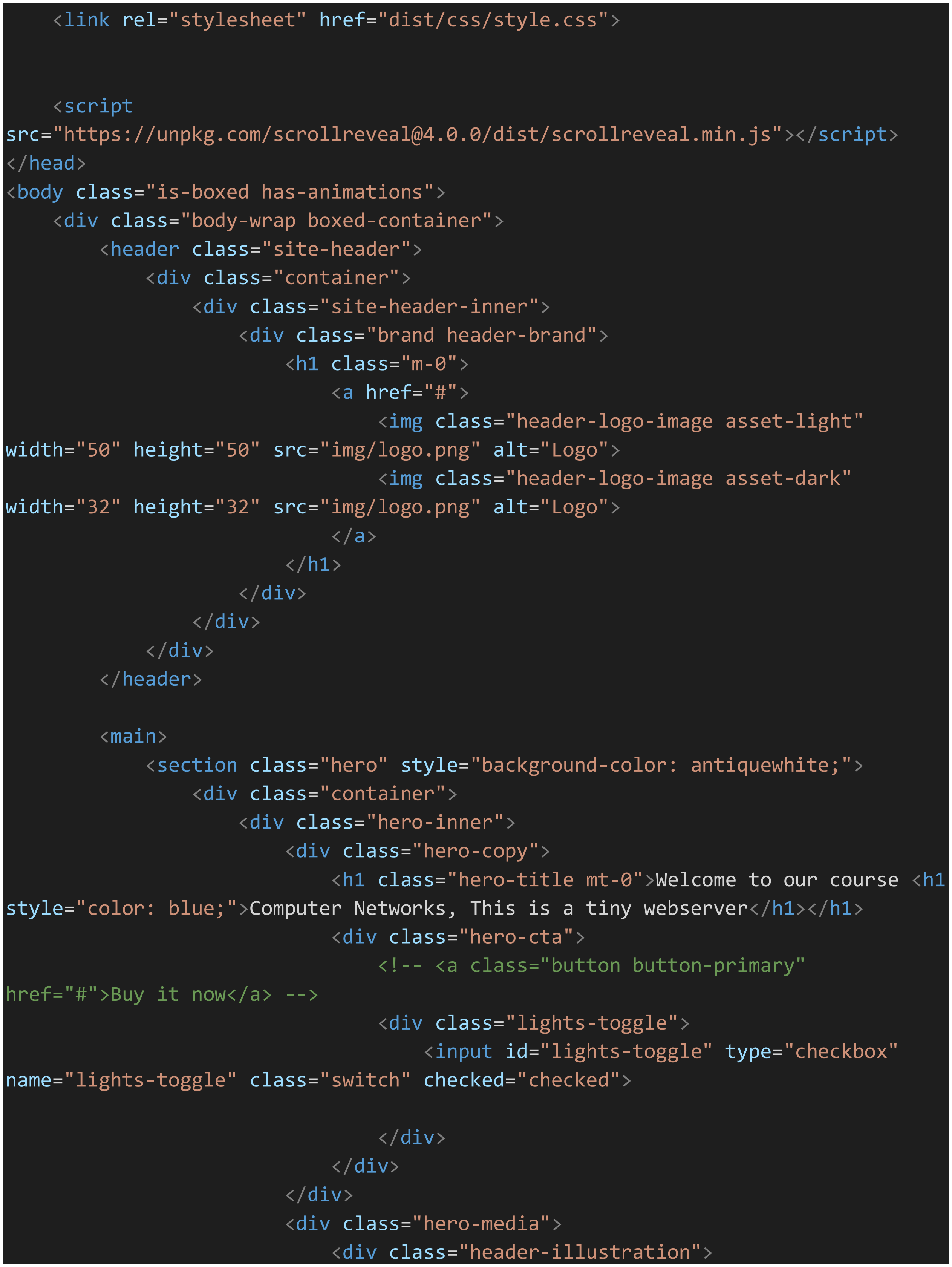


```
        error_response = f"""<!DOCTYPE html>
                    <html lang="en">
                    <head>
                        <meta charset="UTF-8">
                        <meta http-equiv="X-UA-Compatible" content="IE=edge">
                        <meta name="viewport" content="width=device-width,
initial-scale=1.0">
                        <title>Error 404</title>
                        <style>
                            body {{ font-family: Arial, sans-serif; text-align:
center; }}
                            .error-message {{ color: red; }}
                            .client-info {{ font-weight: bold; }}
                        </style>
                    </head>
                    <body>
                        <h1>HTTP/1.1 404 Not Found</h1>
                        <h2 class="error-message">The file is not found</h2>
                        <div class="client-info">
                            <p>1190546 - oday khallaf</p>
                            <p>1200609 - anwaar qasem </p>
                            <p>1200620 -  tala dabbagh </p>
                            <p>{address[0]}:{address[1]}</p>
                        </div>
                    </body>
                    </html>"""
        header = 'HTTP/1.1 404 Not Found\n\n'  # Prepare the error header.
        # Encode the error response to bytes.
        response = error_response.encode('utf-8')

    # Combine the header and the response content.
    final_response = header.encode('utf-8') + response
    # Send the final response to the client.
    connection.send(final_response)
    connection.close()  # Close the client connection.

# Set up the socket for the server.
HOST, PORT = '127.0.0.1', 9966  # Define port for the server;
# Create a socket object supporting IPv4 and TCP protocols
my_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
my_socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
my_socket.bind((HOST, PORT))  # Bind the socket to the host and port
my_socket.listen(1)  # Start listening for incoming connections
print('Serving on port ', PORT)
```

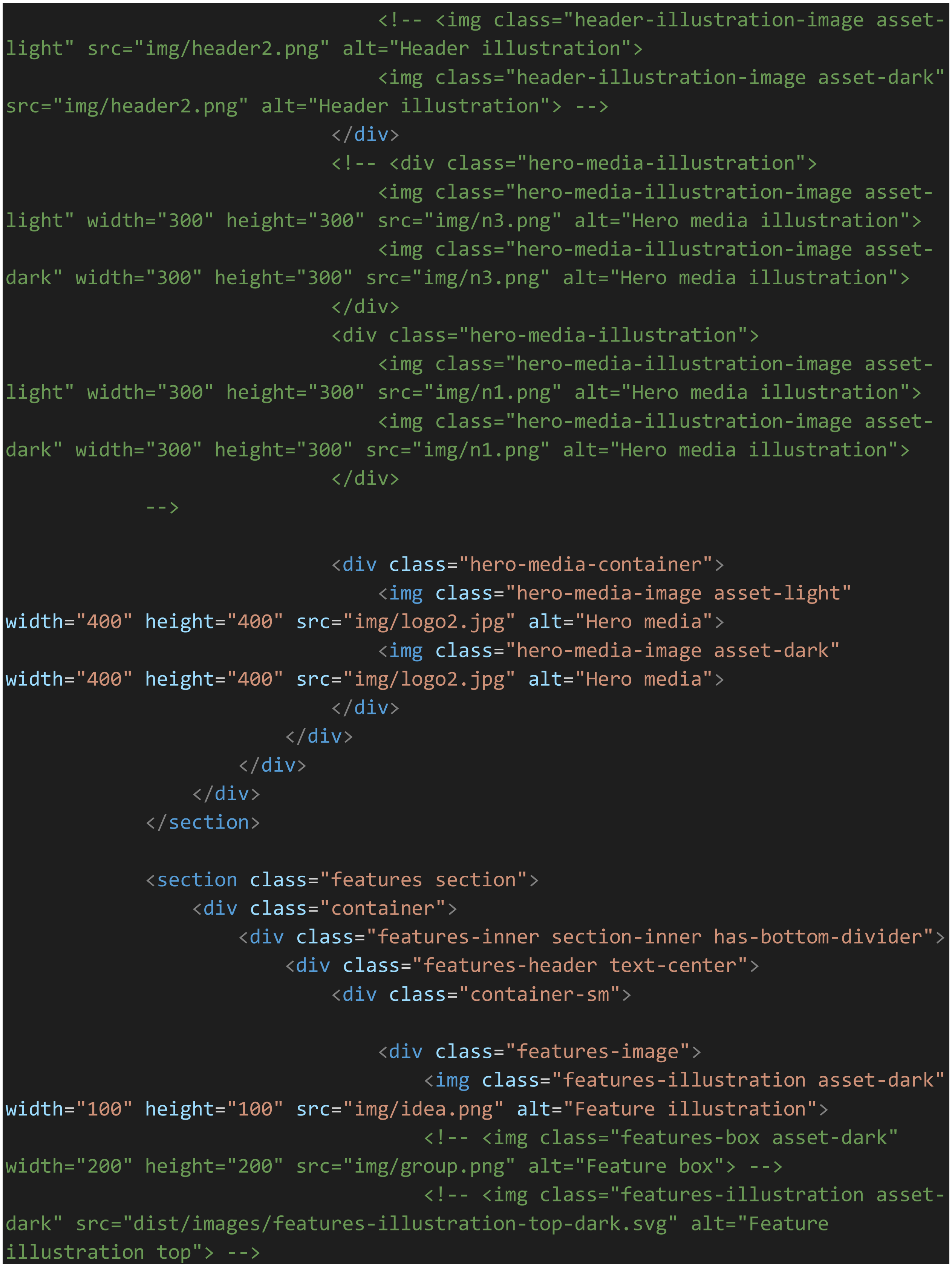
```python
# Start the CLI request
cli_thread = threading.Thread(target=handle_cli_request)
cli_thread.start()

# start listening for network requests.
try:
    while True:
        # Accept an incoming connection
        connection, address = my_socket.accept()
        # Start a new thread to handle the HTTP request.
        thread = threading.Thread(target=handle_http_request, args=(connection,
address))
        thread.start()
except KeyboardInterrupt:
    # Handle a keyboard interrupt to shut down the server gracefully.
    print("Server is shutting down.")
    my_socket.close()
```

## main_en.html code :

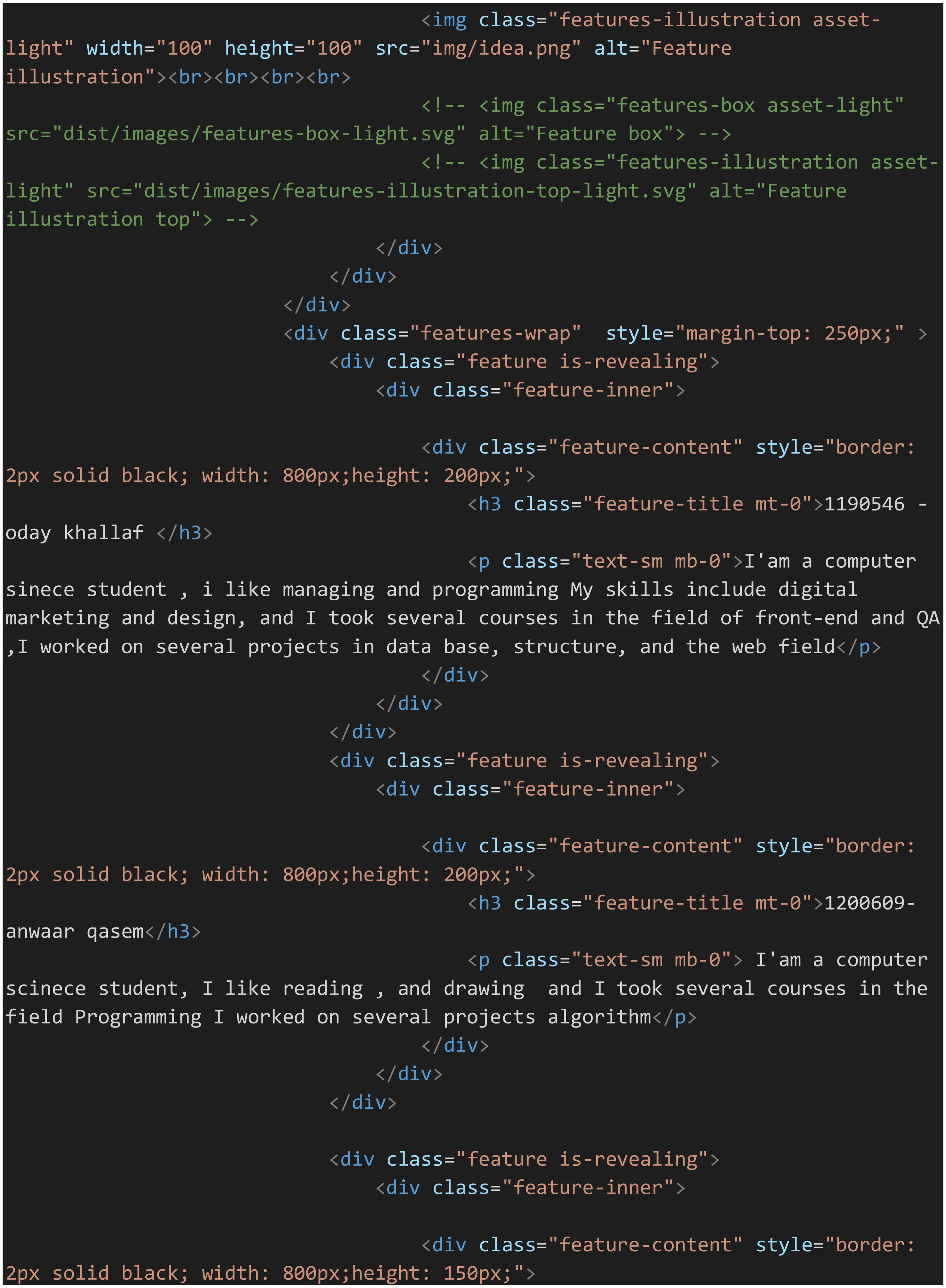
```html
<!DOCTYPE html>
<html lang="en" class="no-js">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>ENCS3320-Simple Webserver</title>
    <link
href="https://fonts.googleapis.com/css?family=Heebo:400,700|IBM+Plex+Sans:600"
rel="stylesheet">
```



```
    <link rel="stylesheet" href="dist/css/style.css">


    <script
src="https://unpkg.com/scrollreveal@4.0.0/dist/scrollreveal.min.js"></script>
</head>
<body class="is-boxed has-animations">
    <div class="body-wrap boxed-container">
        <header class="site-header">
            <div class="container">
                <div class="site-header-inner">
                    <div class="brand header-brand">
                        <h1 class="m-0">
                            <a href="#">
                                <img class="header-logo-image asset-light"
width="50" height="50" src="img/logo.png" alt="Logo">
                                <img class="header-logo-image asset-dark"
width="32" height="32" src="img/logo.png" alt="Logo">
                            </a>
                        </h1>
                    </div>
                </div>
            </div>
        </header>

        <main>
            <section class="hero" style="background-color: antiquewhite;">
                <div class="container">
                    <div class="hero-inner">
                        <div class="hero-copy">
                            <h1 class="hero-title mt-0">Welcome to our course <h1
style="color: blue;">Computer Networks, This is a tiny webserver</h1></h1>
                            <div class="hero-cta">
                                <!-- <a class="button button-primary"
href="#">Buy it now</a> -->
                                <div class="lights-toggle">
                                    <input id="lights-toggle" type="checkbox"
name="lights-toggle" class="switch" checked="checked">

                                </div>
                            </div>
                        </div>
                        <div class="hero-media">
                            <div class="header-illustration">
```



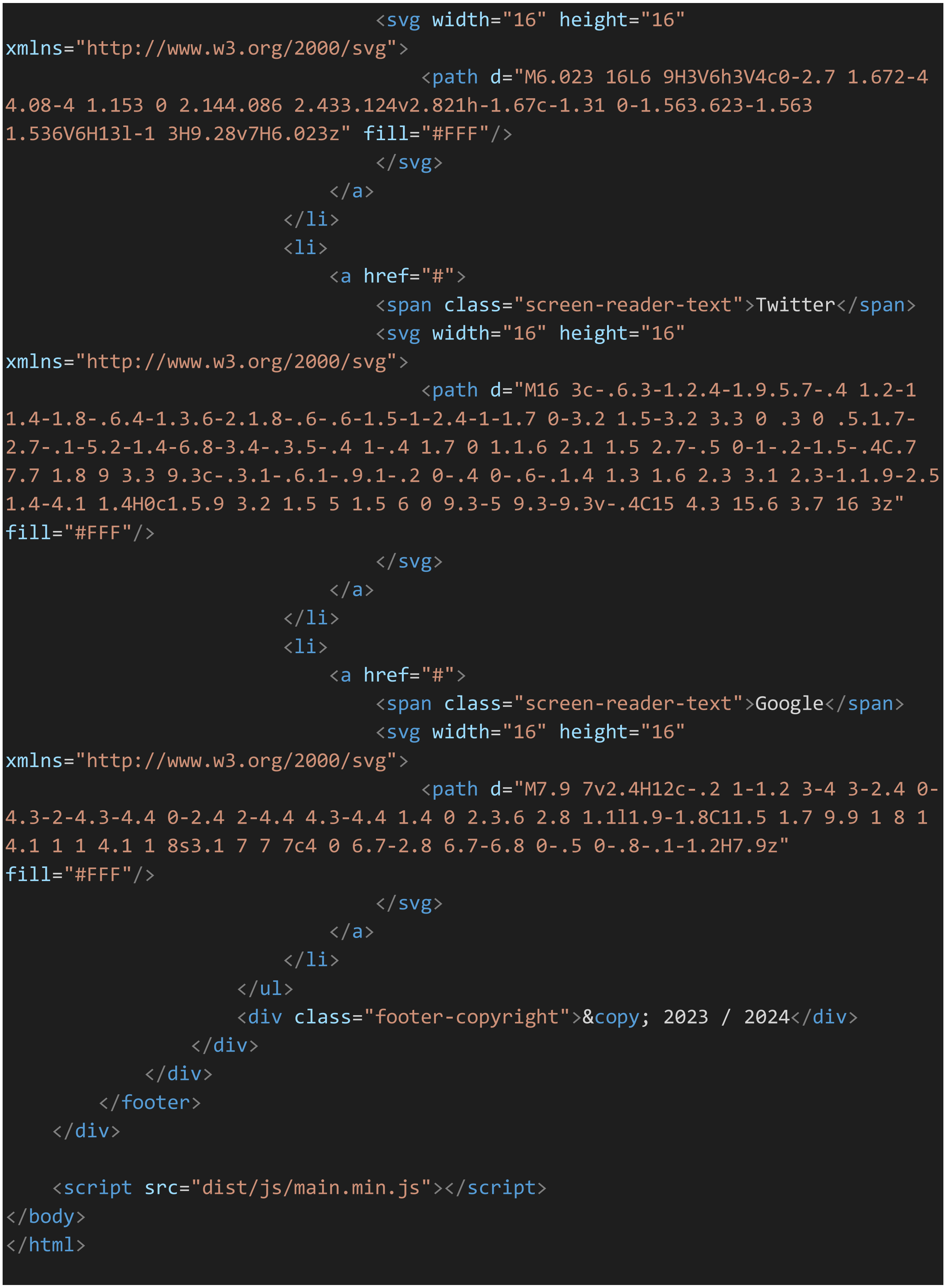
```
                                <!-- <img class="header-illustration-image asset-
light" src="img/header2.png" alt="Header illustration">
                                <img class="header-illustration-image asset-dark"
src="img/header2.png" alt="Header illustration"> -->
                            </div>
                            <!-- <div class="hero-media-illustration">
                                <img class="hero-media-illustration-image asset-
light" width="300" height="300" src="img/n3.png" alt="Hero media illustration">
                                <img class="hero-media-illustration-image asset-
dark" width="300" height="300" src="img/n3.png" alt="Hero media illustration">
                            </div>
                            <div class="hero-media-illustration">
                                <img class="hero-media-illustration-image asset-
light" width="300" height="300" src="img/n1.png" alt="Hero media illustration">
                                <img class="hero-media-illustration-image asset-
dark" width="300" height="300" src="img/n1.png" alt="Hero media illustration">
                            </div>
            -->

                            <div class="hero-media-container">
                                <img class="hero-media-image asset-light"
width="400" height="400" src="img/logo2.jpg" alt="Hero media">
                                <img class="hero-media-image asset-dark"
width="400" height="400" src="img/logo2.jpg" alt="Hero media">
                            </div>
                        </div>
                    </div>
                </div>
            </section>

            <section class="features section">
                <div class="container">
                    <div class="features-inner section-inner has-bottom-divider">
                        <div class="features-header text-center">
                            <div class="container-sm">

                                <div class="features-image">
                                    <img class="features-illustration asset-dark"
width="100" height="100" src="img/idea.png" alt="Feature illustration">
                                    <!-- <img class="features-box asset-dark"
width="200" height="200" src="img/group.png" alt="Feature box"> -->
                                    <!-- <img class="features-illustration asset-
dark" src="dist/images/features-illustration-top-dark.svg" alt="Feature
illustration top"> -->
```



```
                                    <img class="features-illustration asset-
light" width="100" height="100" src="img/idea.png" alt="Feature
illustration"><br><br><br><br>
                                    <!-- <img class="features-box asset-light"
src="dist/images/features-box-light.svg" alt="Feature box"> -->
                                    <!-- <img class="features-illustration asset-
light" src="dist/images/features-illustration-top-light.svg" alt="Feature
illustration top"> -->
                                </div>
                            </div>
                        </div>
                        <div class="features-wrap"  style="margin-top: 250px;" >
                            <div class="feature is-revealing">
                                <div class="feature-inner">

                                    <div class="feature-content" style="border:
2px solid black; width: 800px;height: 200px;">
                                        <h3 class="feature-title mt-0">1190546 -
oday khallaf </h3>
                                        <p class="text-sm mb-0">I'am a computer
sinece student , i like managing and programming My skills include digital
marketing and design, and I took several courses in the field of front-end and QA
,I worked on several projects in data base, structure, and the web field</p>
                                    </div>
                                </div>
                            </div>
                            <div class="feature is-revealing">
                                <div class="feature-inner">

                                    <div class="feature-content" style="border:
2px solid black; width: 800px;height: 200px;">
                                        <h3 class="feature-title mt-0">1200609-
anwaar qasem</h3>
                                        <p class="text-sm mb-0"> I'am a computer
scinece student, I like reading , and drawing  and I took several courses in the
field Programming I worked on several projects algorithm</p>
                                    </div>
                                </div>
                            </div>

                            <div class="feature is-revealing">
                                <div class="feature-inner">

                                    <div class="feature-content" style="border:
2px solid black; width: 800px;height: 150px;">
```

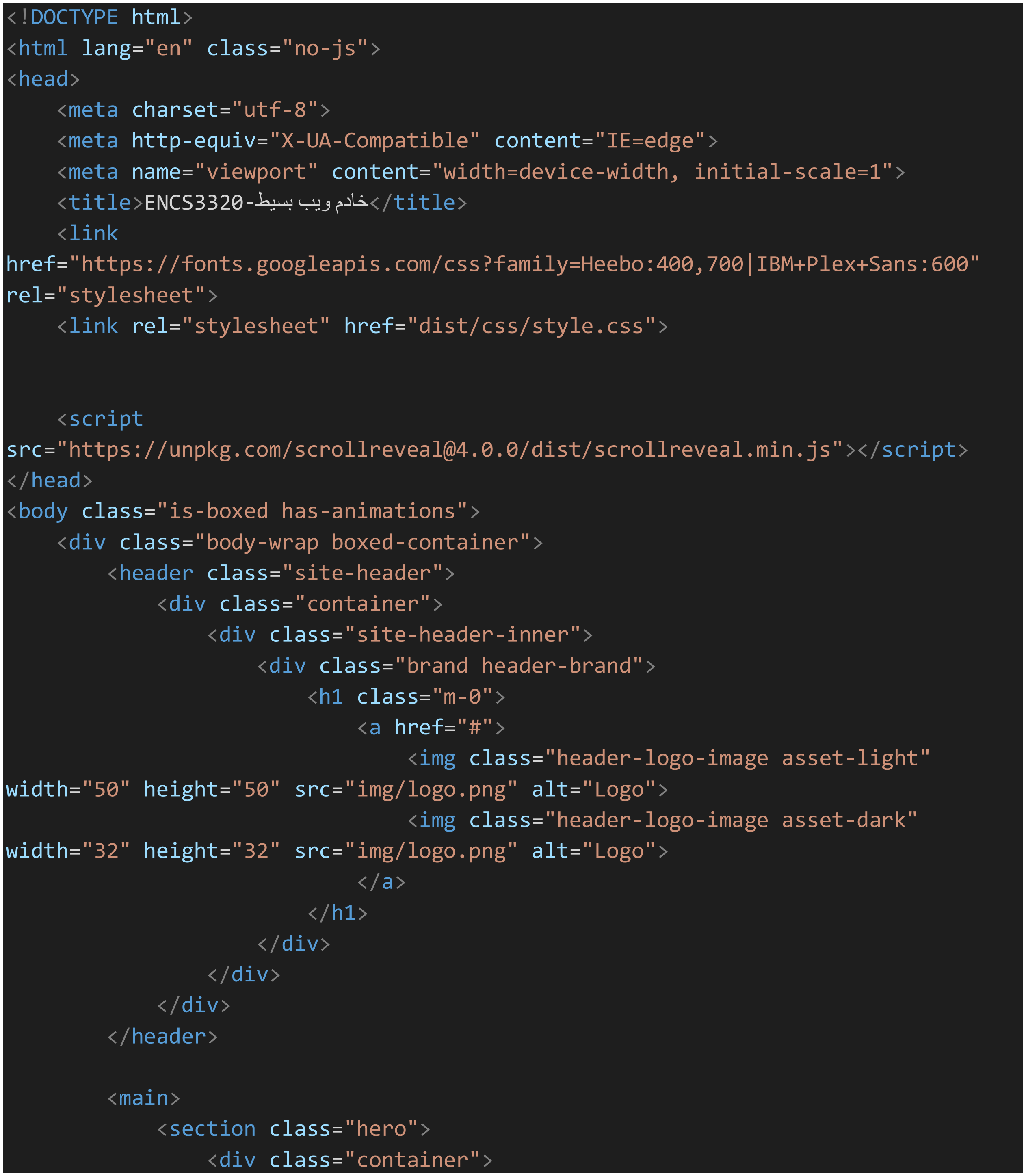
```
                                        <h3 class="feature-title mt-0">1200620-
tala dabbagh</h3>
                                        <p class="text-sm mb-0"> I'am a computer
sinece student  I like Photography , My skills are problem solving and time
management, and I took courses in Backend</p>
                                    </div>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
            </section>

            <section class="cta section">
                <div class="container-sm">
                    <div class="cta-inner section-inner">
                        <div class="cta-header text-center "style="border: 2px
solid black; width: 800px;height: 800px;">
                            <h2 class="section-title mt-0">Content-Type
header</h2>
                            <p class="section-paragraph">The Content-Type header
in an HTTP request, as specified in RFC 2616 (which covers HTTP/1.1), is used to
indicate the media type of the body of the request. This header is important
because it tells the server what the data actually is and how it should be
processed. In technical terms, the Content-Type header specifies the MIME type of
the content.

                                Here are a few key points about the Content-Type
header:

                                Identification of Data Format: It helps the
server understand the format of the data sent in the request. This could be a
format like text/html, application/json, image/jpeg, etc.

                                Data Processing: Based on the Content-Type, the
server can correctly process the data. For example, if the Content-Type is
application/json, the server can parse the body as a JSON object.

                                Ensuring Compatibility: It ensures that the data
is compatible with the receiving end. If the server expects a certain type of
content and receives a different type, it can lead to errors in data processing.

                                Client-Server Agreement: It's part of the HTTP
protocol's way of ensuring that both the client and the server are in agreement
about the type of data being communicated.
```

```
                                In summary, the Content-Type header in an HTTP
request is crucial for correctly conveying the type of the content in the request
body, enabling the server to process it appropriately.</p>
                            <div class="cta-cta">
                                <a class="button button-primary"
href="https://datatracker.ietf.org/doc/html/rfc2616" target="_blank">Learn
more</a>
                            </div>
                        </div>
                    </div>
                </div>
            </section>
        </main>

        <footer class="site-footer has-top-divider"style="background-
color:antiquewhite;">
            <div class="container">
                <div class="site-footer-inner">
                    <div class="brand footer-brand">
                        <a href="#">
                            <img class="asset-light" height="100" width="100"
src="img/logo2.jpg" alt="Logo">
                            <img class="asset-dark" height="100" width="100"
src="img/logo2.png" alt="Logo">
                        </a>
                    </div>
                    <ul class="footer-links list-reset">
                        <li>
                            <a href="#">Contact</a>
                        </li>
                        <li>
                            <a href="#">About us</a>
                        </li>
                        <li>
                            <a href="#">FAQ's</a>
                        </li>
                        <li>
                            <a href="#">Support</a>
                        </li>
                    </ul>
                    <ul class="footer-social-links list-reset" >
                        <li>
                            <a href="#">
                                <span class="screen-reader-text">Facebook</span>
```

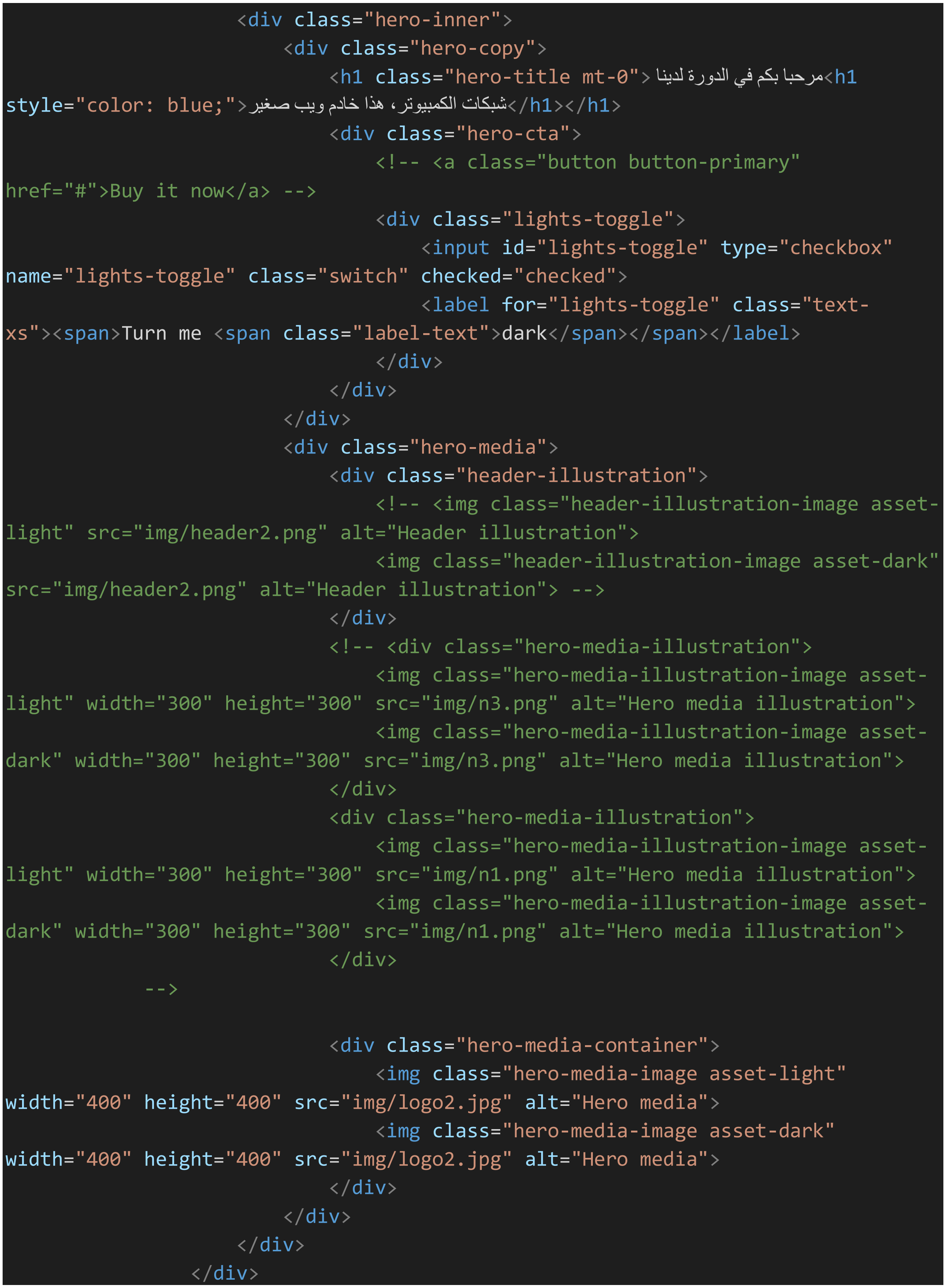


```
                                <svg width="16" height="16"
xmlns="http://www.w3.org/2000/svg">
                                    <path d="M6.023 16L6 9H3V6h3V4c0-2.7 1.672-4
4.08-4 1.153 0 2.144.086 2.433.124v2.821h-1.67c-1.31 0-1.563.623-1.563
1.536V6H13l-1 3H9.28v7H6.023z" fill="#FFF"/>
                                </svg>
                            </a>
                        </li>
                        <li>
                            <a href="#">
                                <span class="screen-reader-text">Twitter</span>
                                <svg width="16" height="16"
xmlns="http://www.w3.org/2000/svg">
                                    <path d="M16 3c-.6.3-1.2.4-1.9.5.7-.4 1.2-1
1.4-1.8-.6.4-1.3.6-2.1.8-.6-.6-1.5-1-2.4-1-1.7 0-3.2 1.5-3.2 3.3 0 .3 0 .5.1.7-
2.7-.1-5.2-1.4-6.8-3.4-.3.5-.4 1-.4 1.7 0 1.1.6 2.1 1.5 2.7-.5 0-1-.2-1.5-.4C.7
7.7 1.8 9 3.3 9.3c-.3.1-.6.1-.9.1-.2 0-.4 0-.6-.1.4 1.3 1.6 2.3 3.1 2.3-1.1.9-2.5
1.4-4.1 1.4H0c1.5.9 3.2 1.5 5 1.5 6 0 9.3-5 9.3-9.3v-.4C15 4.3 15.6 3.7 16 3z"
fill="#FFF"/>
                                </svg>
                            </a>
                        </li>
                        <li>
                            <a href="#">
                                <span class="screen-reader-text">Google</span>
                                <svg width="16" height="16"
xmlns="http://www.w3.org/2000/svg">
                                    <path d="M7.9 7v2.4H12c-.2 1-1.2 3-4 3-2.4 0-
4.3-2-4.3-4.4 0-2.4 2-4.4 4.3-4.4 1.4 0 2.3.6 2.8 1.1l1.9-1.8C11.5 1.7 9.9 1 8 1
4.1 1 1 4.1 1 8s3.1 7 7 7c4 0 6.7-2.8 6.7-6.8 0-.5 0-.8-.1-1.2H7.9z"
fill="#FFF"/>
                                </svg>
                            </a>
                        </li>
                    </ul>
                    <div class="footer-copyright">&copy; 2023 / 2024</div>
                </div>
            </div>
        </footer>
    </div>

    <script src="dist/js/main.min.js"></script>
</body>
</html>
```

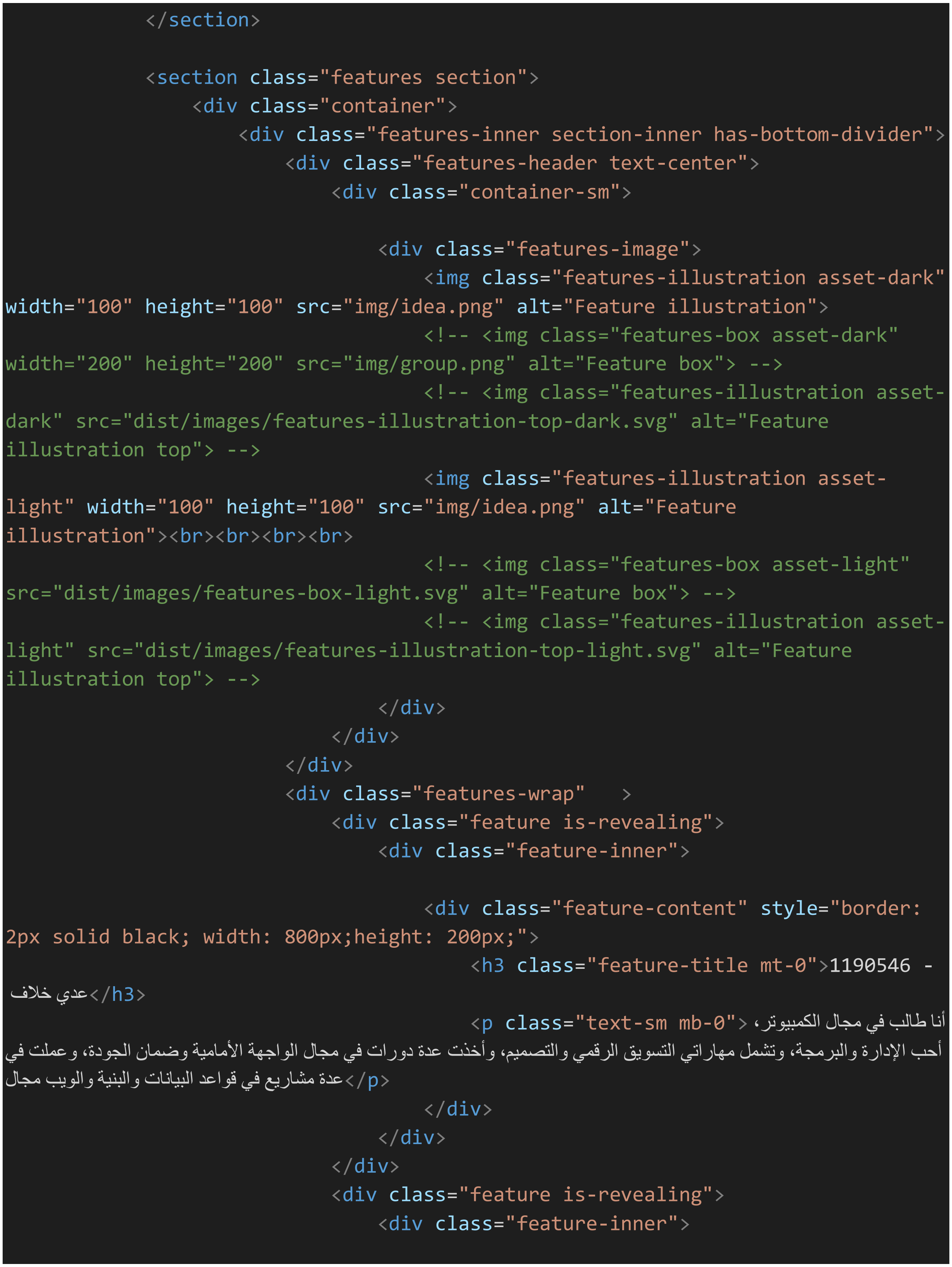
main_ar.html code :



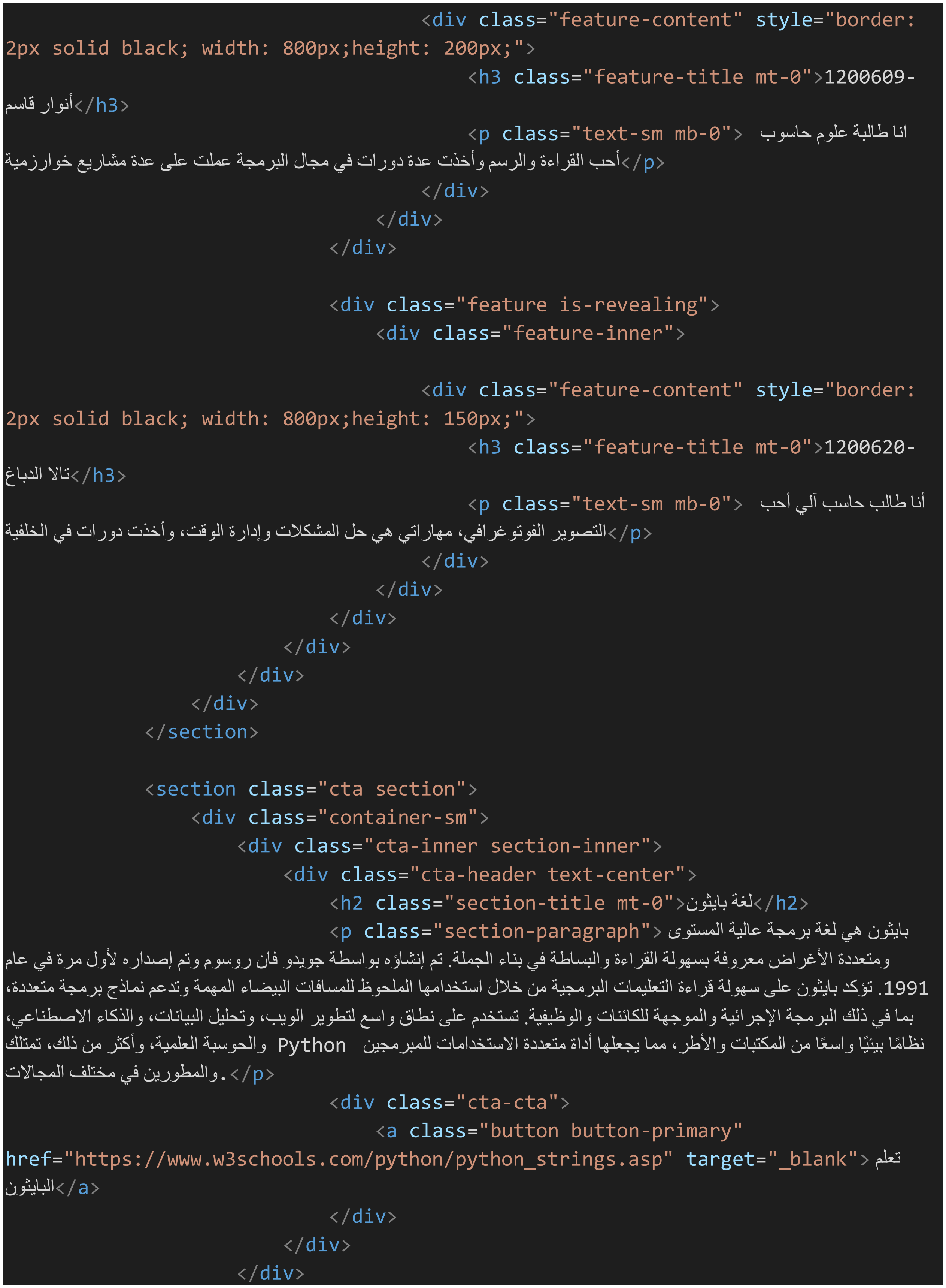
```
<!DOCTYPE html>
<html lang="en" class="no-js">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>ENCS3320-خادم ويب بسيط</title>
    <link
href="https://fonts.googleapis.com/css?family=Heebo:400,700|IBM+Plex+Sans:600"
rel="stylesheet">
    <link rel="stylesheet" href="dist/css/style.css">


    <script
src="https://unpkg.com/scrollreveal@4.0.0/dist/scrollreveal.min.js"></script>
</head>
<body class="is-boxed has-animations">
    <div class="body-wrap boxed-container">
        <header class="site-header">
            <div class="container">
                <div class="site-header-inner">
                    <div class="brand header-brand">
                        <h1 class="m-0">
                            <a href="#">
                                <img class="header-logo-image asset-light"
width="50" height="50" src="img/logo.png" alt="Logo">
                                <img class="header-logo-image asset-dark"
width="32" height="32" src="img/logo.png" alt="Logo">
                            </a>
                        </h1>
                    </div>
                </div>
            </div>
        </header>

        <main>
            <section class="hero">
                <div class="container">
```



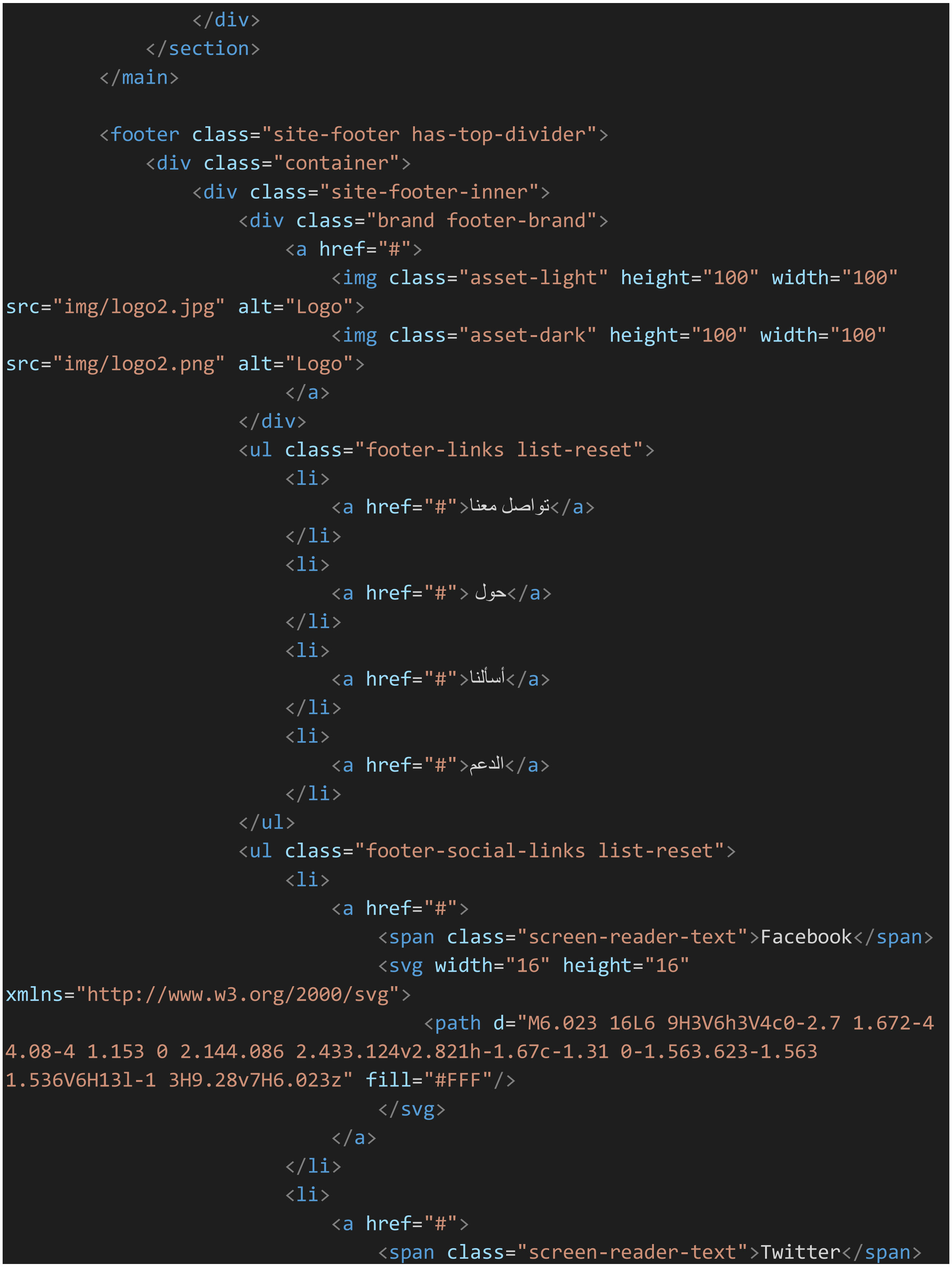
```
                    <div class="hero-inner">
                        <div class="hero-copy">
                            <h1 class="hero-title mt-0">مرحبا بكم في الدورة لدينا<h1
style="color: blue;">شبكات الكمبيوتر، هذا خادم ويب صغير</h1></h1>
                            <div class="hero-cta">
                                <!-- <a class="button button-primary"
href="#">Buy it now</a> -->
                                <div class="lights-toggle">
                                    <input id="lights-toggle" type="checkbox"
name="lights-toggle" class="switch" checked="checked">
                                    <label for="lights-toggle" class="text-
xs"><span>Turn me <span class="label-text">dark</span></span></label>
                                </div>
                            </div>
                        </div>
                        <div class="hero-media">
                            <div class="header-illustration">
                                <!-- <img class="header-illustration-image asset-
light" src="img/header2.png" alt="Header illustration">
                                <img class="header-illustration-image asset-dark"
src="img/header2.png" alt="Header illustration"> -->
                            </div>
                            <!-- <div class="hero-media-illustration">
                                <img class="hero-media-illustration-image asset-
light" width="300" height="300" src="img/n3.png" alt="Hero media illustration">
                                <img class="hero-media-illustration-image asset-
dark" width="300" height="300" src="img/n3.png" alt="Hero media illustration">
                            </div>
                            <div class="hero-media-illustration">
                                <img class="hero-media-illustration-image asset-
light" width="300" height="300" src="img/n1.png" alt="Hero media illustration">
                                <img class="hero-media-illustration-image asset-
dark" width="300" height="300" src="img/n1.png" alt="Hero media illustration">
                            </div>
            -->

                            <div class="hero-media-container">
                                <img class="hero-media-image asset-light"
width="400" height="400" src="img/logo2.jpg" alt="Hero media">
                                <img class="hero-media-image asset-dark"
width="400" height="400" src="img/logo2.jpg" alt="Hero media">
                            </div>
                        </div>
                    </div>
                </div>
```



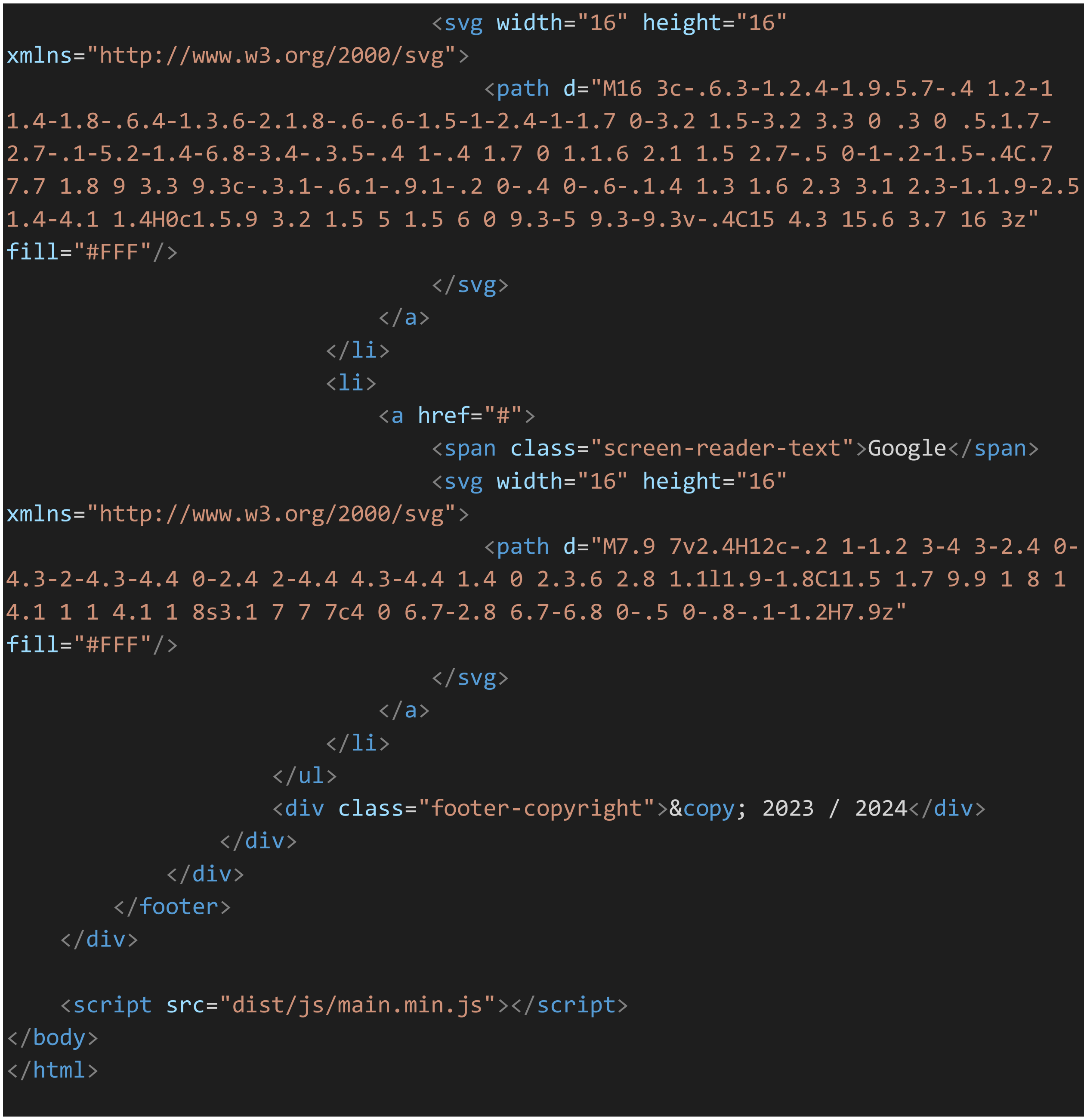
```
            </section>

            <section class="features section">
                <div class="container">
                    <div class="features-inner section-inner has-bottom-divider">
                        <div class="features-header text-center">
                            <div class="container-sm">

                                <div class="features-image">
                                    <img class="features-illustration asset-dark"
width="100" height="100" src="img/idea.png" alt="Feature illustration">
                                    <!-- <img class="features-box asset-dark"
width="200" height="200" src="img/group.png" alt="Feature box"> -->
                                    <!-- <img class="features-illustration asset-
dark" src="dist/images/features-illustration-top-dark.svg" alt="Feature
illustration top"> -->
                                    <img class="features-illustration asset-
light" width="100" height="100" src="img/idea.png" alt="Feature
illustration"><br><br><br><br>
                                    <!-- <img class="features-box asset-light"
src="dist/images/features-box-light.svg" alt="Feature box"> -->
                                    <!-- <img class="features-illustration asset-
light" src="dist/images/features-illustration-top-light.svg" alt="Feature
illustration top"> -->
                                </div>
                            </div>
                        </div>
                        <div class="features-wrap"   >
                            <div class="feature is-revealing">
                                <div class="feature-inner">

                                    <div class="feature-content" style="border:
2px solid black; width: 800px;height: 200px;">
                                        <h3 class="feature-title mt-0">1190546 -
عدي خلاف</h3>
                                        <p class="text-sm mb-0">أنا طالب في مجال الكمبيوتر،
أحب الإدارة والبرمجة، وتشمل مهاراتي التسويق الرقمي والتصميم، وأخذت عدة دورات في مجال الواجهة الأمامية وضمان الجودة، وعملت في
مجال الويب والبنية وقواعد البيانات في عدة مشاريع</p>
                                    </div>
                                </div>
                            </div>
                            <div class="feature is-revealing">
                                <div class="feature-inner">
```



```
                                    <div class="feature-content" style="border:
2px solid black; width: 800px;height: 200px;">
                                        <h3 class="feature-title mt-0">1200609-
أنوار قاسم</h3>
                                        <p class="text-sm mb-0"> انا طالبة علوم حاسوب
أحب القراءة والرسم وأخذت عدة دورات في مجال البرمجة عملت على عدة مشاريع خوارزمية</p>
                                    </div>
                                </div>
                            </div>

                            <div class="feature is-revealing">
                                <div class="feature-inner">

                                    <div class="feature-content" style="border:
2px solid black; width: 800px;height: 150px;">
                                        <h3 class="feature-title mt-0">1200620-
تالا الدباغ</h3>
                                        <p class="text-sm mb-0"> أنا طالب حاسب آلي أحب
التصوير الفوتوغرافي، مهاراتي هي حل المشكلات وإدارة الوقت، وأخذت دورات في الخلفية</p>
                                    </div>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
            </section>

            <section class="cta section">
                <div class="container-sm">
                    <div class="cta-inner section-inner">
                        <div class="cta-header text-center">
                            <h2 class="section-title mt-0">لغة بايثون</h2>
                            <p class="section-paragraph"> بايثون هي لغة برمجة عالية المستوى
ومتعددة الأغراض معروفة بسهولة القراءة والبساطة في بناء الجملة. تم إنشاؤه بواسطة جويدو فان روسوم وتم إصداره لأول مرة في عام
1991. تؤكد بايثون على سهولة قراءة التعليمات البرمجية من خلال استخدامها الملحوظ للمسافات البيضاء المهمة وتدعم نماذج برمجة متعددة،
بما في ذلك البرمجة الإجرائية والموجهة للكائنات والوظيفية. تستخدم على نطاق واسع لتطوير الويب، وتحليل البيانات، والذكاء الاصطناعي،
والحوسبة العلمية، وأكثر من ذلك، تمتلك Python نظامًا بيئيًا واسعًا من المكتبات والأطر، مما يجعلها أداة متعددة الاستخدامات للمبرمجين
والمطورين في مختلف المجالات.</p>
                            <div class="cta-cta">
                                <a class="button button-primary"
href="https://www.w3schools.com/python/python_strings.asp" target="_blank"> تعلم
البايثون</a>
                            </div>
                        </div>
                    </div>
```

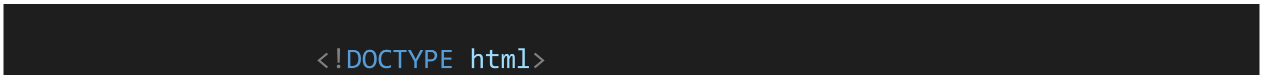


```
                </div>
            </section>
        </main>

        <footer class="site-footer has-top-divider">
            <div class="container">
                <div class="site-footer-inner">
                    <div class="brand footer-brand">
                        <a href="#">
                            <img class="asset-light" height="100" width="100"
src="img/logo2.jpg" alt="Logo">
                            <img class="asset-dark" height="100" width="100"
src="img/logo2.png" alt="Logo">
                        </a>
                    </div>
                    <ul class="footer-links list-reset">
                        <li>
                            <a href="#">تواصل معنا</a>
                        </li>
                        <li>
                            <a href="#"> حول</a>
                        </li>
                        <li>
                            <a href="#">أسألنا</a>
                        </li>
                        <li>
                            <a href="#">الدعم</a>
                        </li>
                    </ul>
                    <ul class="footer-social-links list-reset">
                        <li>
                            <a href="#">
                                <span class="screen-reader-text">Facebook</span>
                                <svg width="16" height="16"
xmlns="http://www.w3.org/2000/svg">
                                    <path d="M6.023 16L6 9H3V6h3V4c0-2.7 1.672-4
4.08-4 1.153 0 2.144.086 2.433.124v2.821h-1.67c-1.31 0-1.563.623-1.563
1.536V6H13l-1 3H9.28v7H6.023z" fill="#FFF"/>
                                </svg>
                            </a>
                        </li>
                        <li>
                            <a href="#">
                                <span class="screen-reader-text">Twitter</span>
```

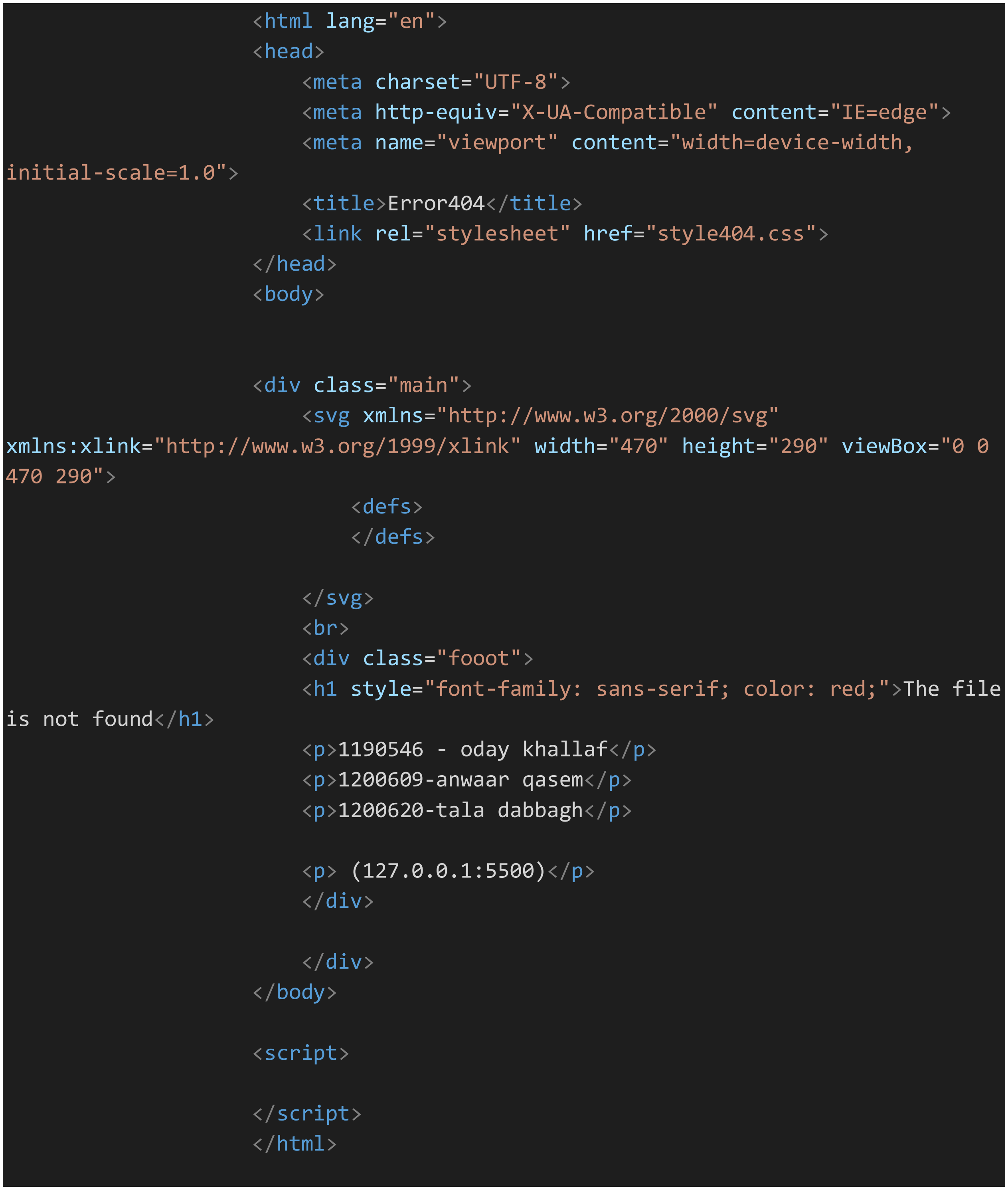


```
                                <svg width="16" height="16"
xmlns="http://www.w3.org/2000/svg">
                                    <path d="M16 3c-.6.3-1.2.4-1.9.5.7-.4 1.2-1
1.4-1.8-.6.4-1.3.6-2.1.8-.6-.6-1.5-1-2.4-1-1.7 0-3.2 1.5-3.2 3.3 0 .3 0 .5.1.7-
2.7-.1-5.2-1.4-6.8-3.4-.3.5-.4 1-.4 1.7 0 1.1.6 2.1 1.5 2.7-.5 0-1-.2-1.5-.4C.7
7.7 1.8 9 3.3 9.3c-.3.1-.6.1-.9.1-.2 0-.4 0-.6-.1.4 1.3 1.6 2.3 3.1 2.3-1.1.9-2.5
1.4-4.1 1.4H0c1.5.9 3.2 1.5 5 1.5 6 0 9.3-5 9.3-9.3v-.4C15 4.3 15.6 3.7 16 3z"
fill="#FFF"/>
                                </svg>
                            </a>
                        </li>
                        <li>
                            <a href="#">
                                <span class="screen-reader-text">Google</span>
                                <svg width="16" height="16"
xmlns="http://www.w3.org/2000/svg">
                                    <path d="M7.9 7v2.4H12c-.2 1-1.2 3-4 3-2.4 0-
4.3-2-4.3-4.4 0-2.4 2-4.4 4.3-4.4 1.4 0 2.3.6 2.8 1.1l1.9-1.8C11.5 1.7 9.9 1 8 1
4.1 1 1 4.1 1 8s3.1 7 7 7c4 0 6.7-2.8 6.7-6.8 0-.5 0-.8-.1-1.2H7.9z"
fill="#FFF"/>
                                </svg>
                            </a>
                        </li>
                    </ul>
                    <div class="footer-copyright">&copy; 2023 / 2024</div>
                </div>
            </div>
        </footer>
    </div>

    <script src="dist/js/main.min.js"></script>
</body>
</html>
```

## 404.html code :

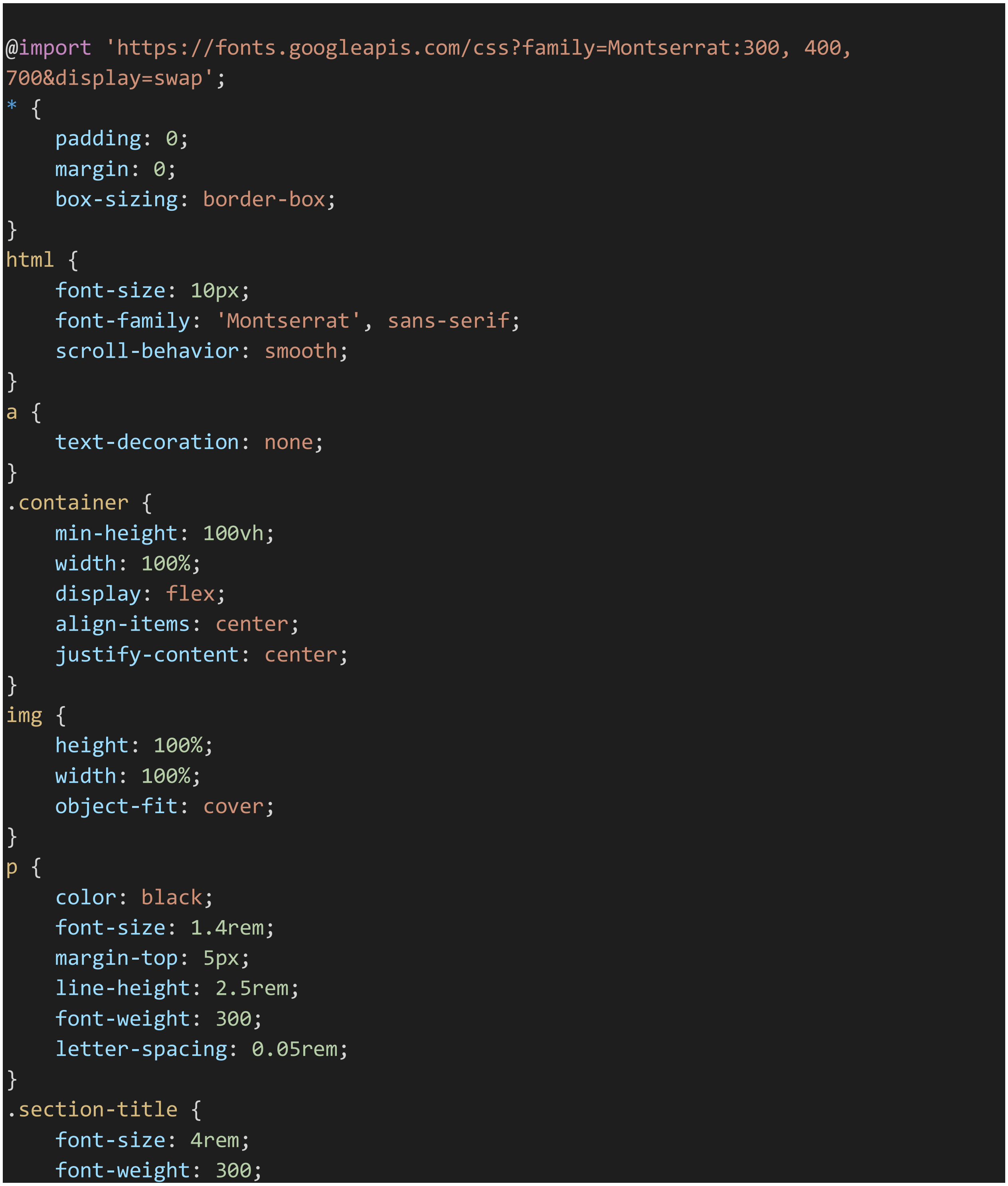


```
                    <!DOCTYPE html>
```

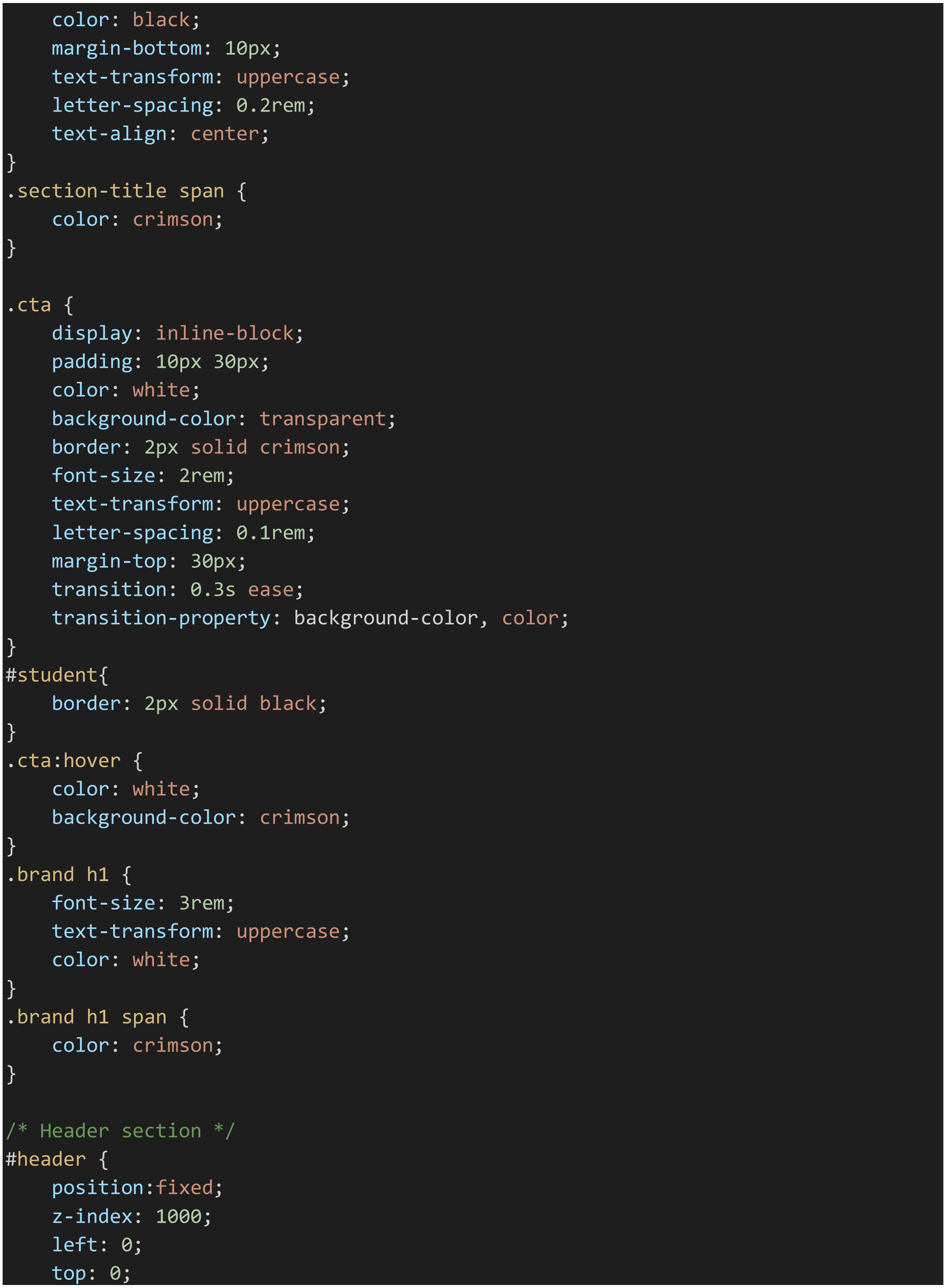


```
                    <html lang="en">
                    <head>
                        <meta charset="UTF-8">
                        <meta http-equiv="X-UA-Compatible" content="IE=edge">
                        <meta name="viewport" content="width=device-width, 
initial-scale=1.0">
                        <title>Error404</title>
                        <link rel="stylesheet" href="style404.css">
                    </head>
                    <body>


                    <div class="main">
                        <svg xmlns="http://www.w3.org/2000/svg" 
xmlns:xlink="http://www.w3.org/1999/xlink" width="470" height="290" viewBox="0 0 
470 290">
                            <defs>
                            </defs>

                        </svg>
                        <br>
                        <div class="fooot">
                        <h1 style="font-family: sans-serif; color: red;">The file 
is not found</h1>
                        <p>1190546 - oday khallaf</p>
                        <p>1200609-anwaar qasem</p>
                        <p>1200620-tala dabbagh</p>

                        <p> (127.0.0.1:5500)</p>
                        </div>

                        </div>
                    </body>

                    <script>

                    </script>
                    </html>
```

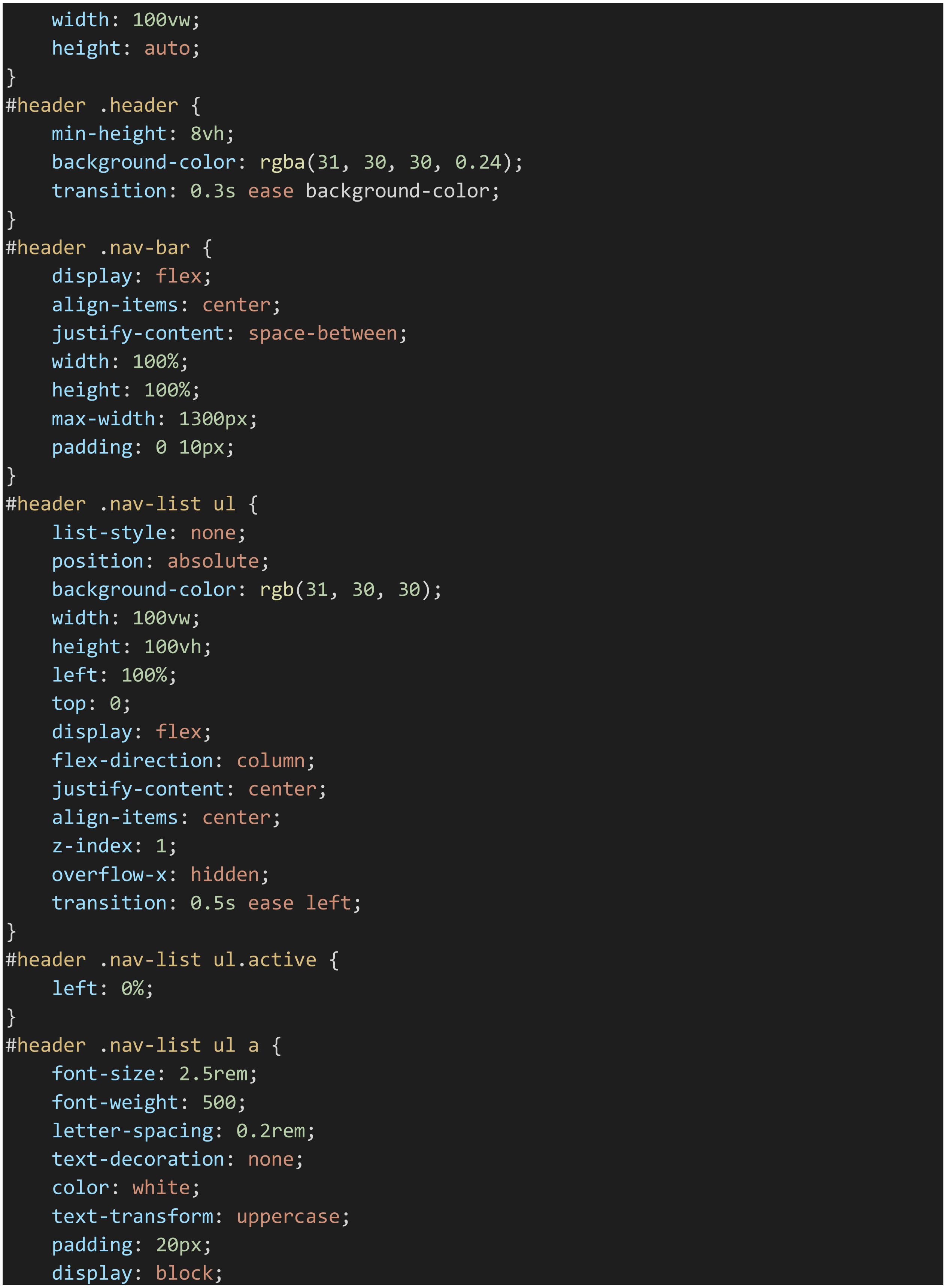
Style.css:



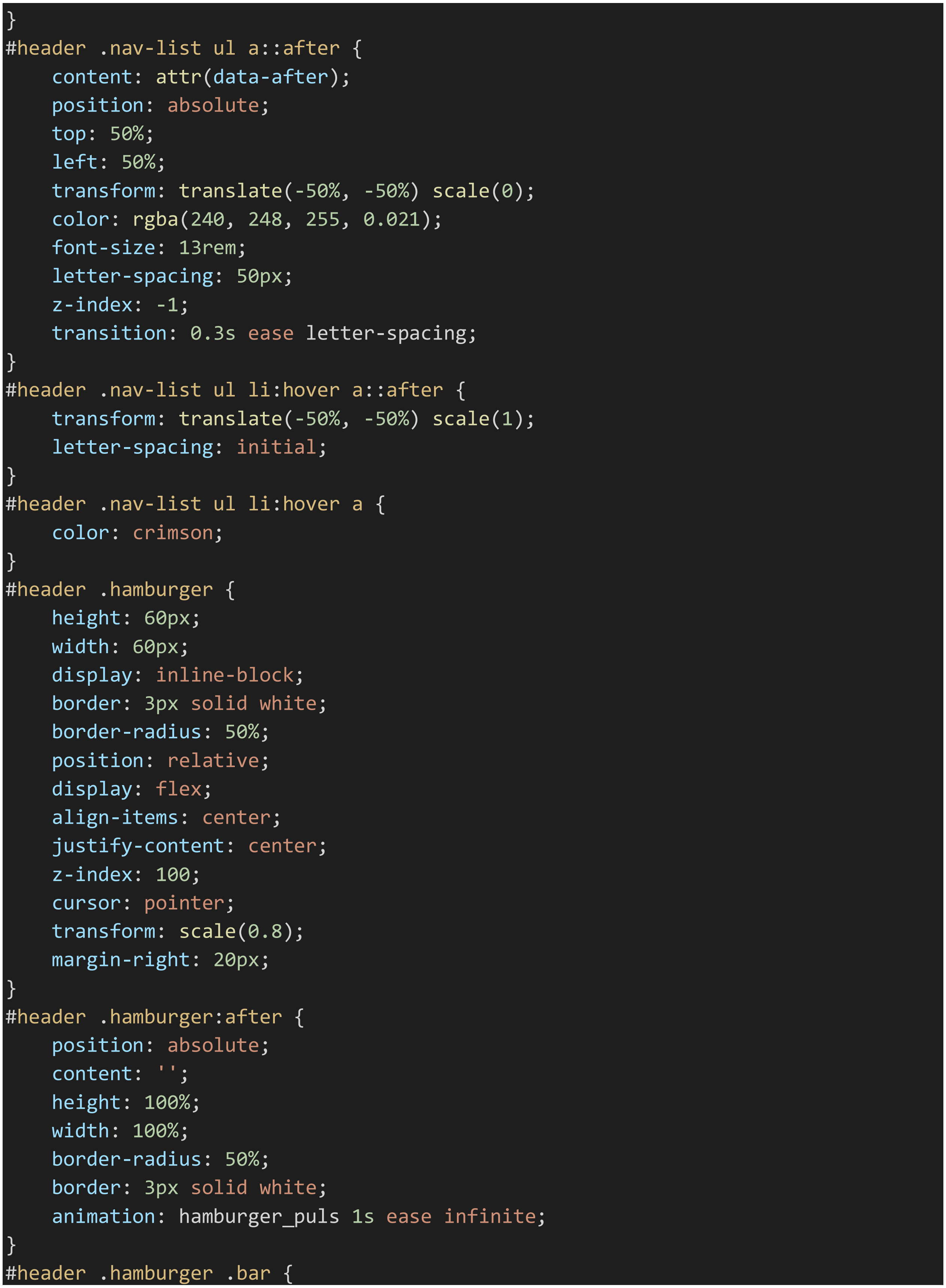
```css
@import 'https://fonts.googleapis.com/css?family=Montserrat:300, 400,
700&display=swap';
* {
    padding: 0;
    margin: 0;
    box-sizing: border-box;
}
html {
    font-size: 10px;
    font-family: 'Montserrat', sans-serif;
    scroll-behavior: smooth;
}
a {
    text-decoration: none;
}
.container {
    min-height: 100vh;
    width: 100%;
    display: flex;
    align-items: center;
    justify-content: center;
}
img {
    height: 100%;
    width: 100%;
    object-fit: cover;
}
p {
    color: black;
    font-size: 1.4rem;
    margin-top: 5px;
    line-height: 2.5rem;
    font-weight: 300;
    letter-spacing: 0.05rem;
}
.section-title {
    font-size: 4rem;
    font-weight: 300;
```



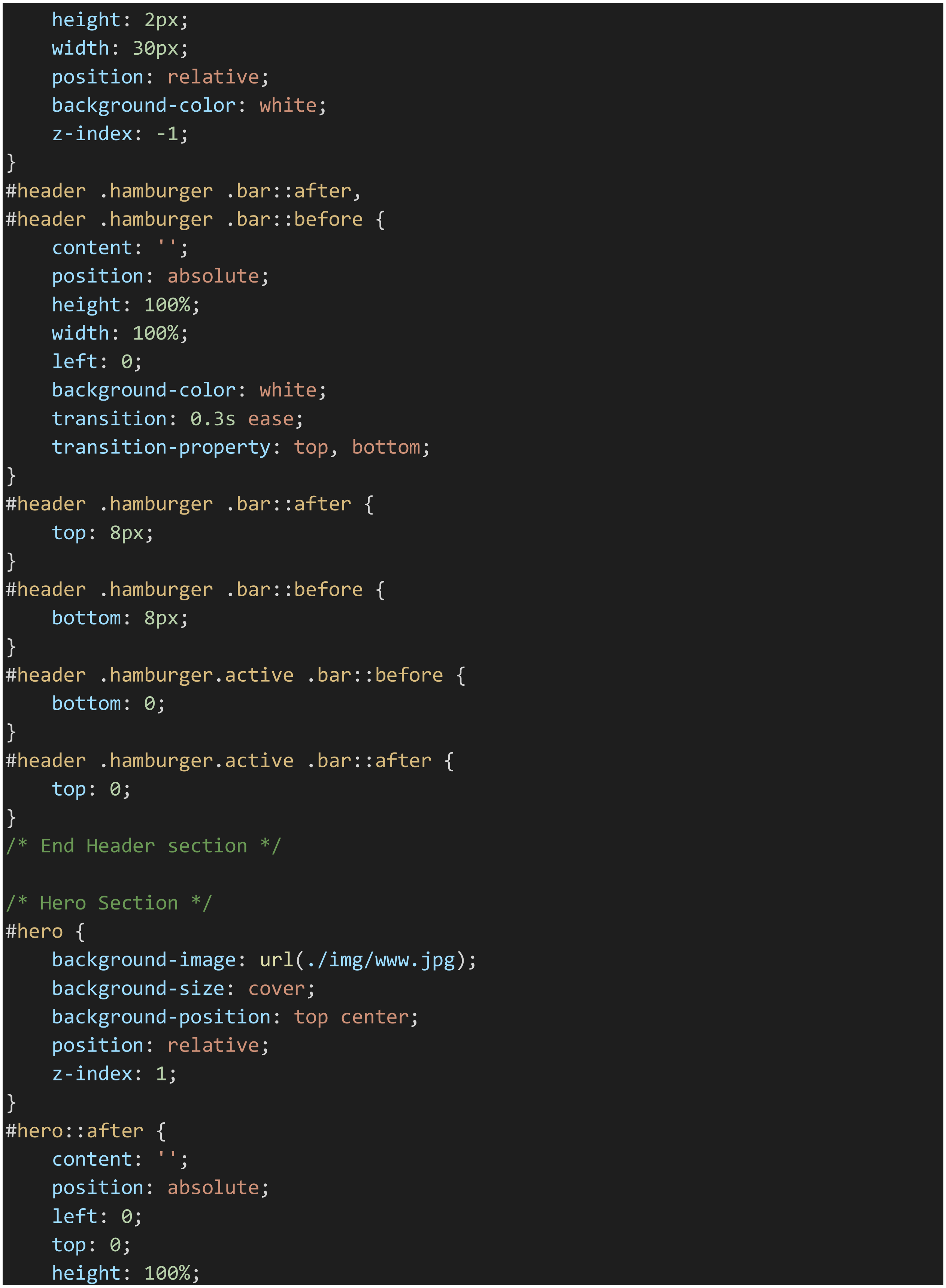
```css
    color: black;
    margin-bottom: 10px;
    text-transform: uppercase;
    letter-spacing: 0.2rem;
    text-align: center;
}
.section-title span {
    color: crimson;
}

.cta {
    display: inline-block;
    padding: 10px 30px;
    color: white;
    background-color: transparent;
    border: 2px solid crimson;
    font-size: 2rem;
    text-transform: uppercase;
    letter-spacing: 0.1rem;
    margin-top: 30px;
    transition: 0.3s ease;
    transition-property: background-color, color;
}
#student{
    border: 2px solid black;
}
.cta:hover {
    color: white;
    background-color: crimson;
}
.brand h1 {
    font-size: 3rem;
    text-transform: uppercase;
    color: white;
}
.brand h1 span {
    color: crimson;
}

/* Header section */
#header {
    position:fixed;
    z-index: 1000;
    left: 0;
    top: 0;
```



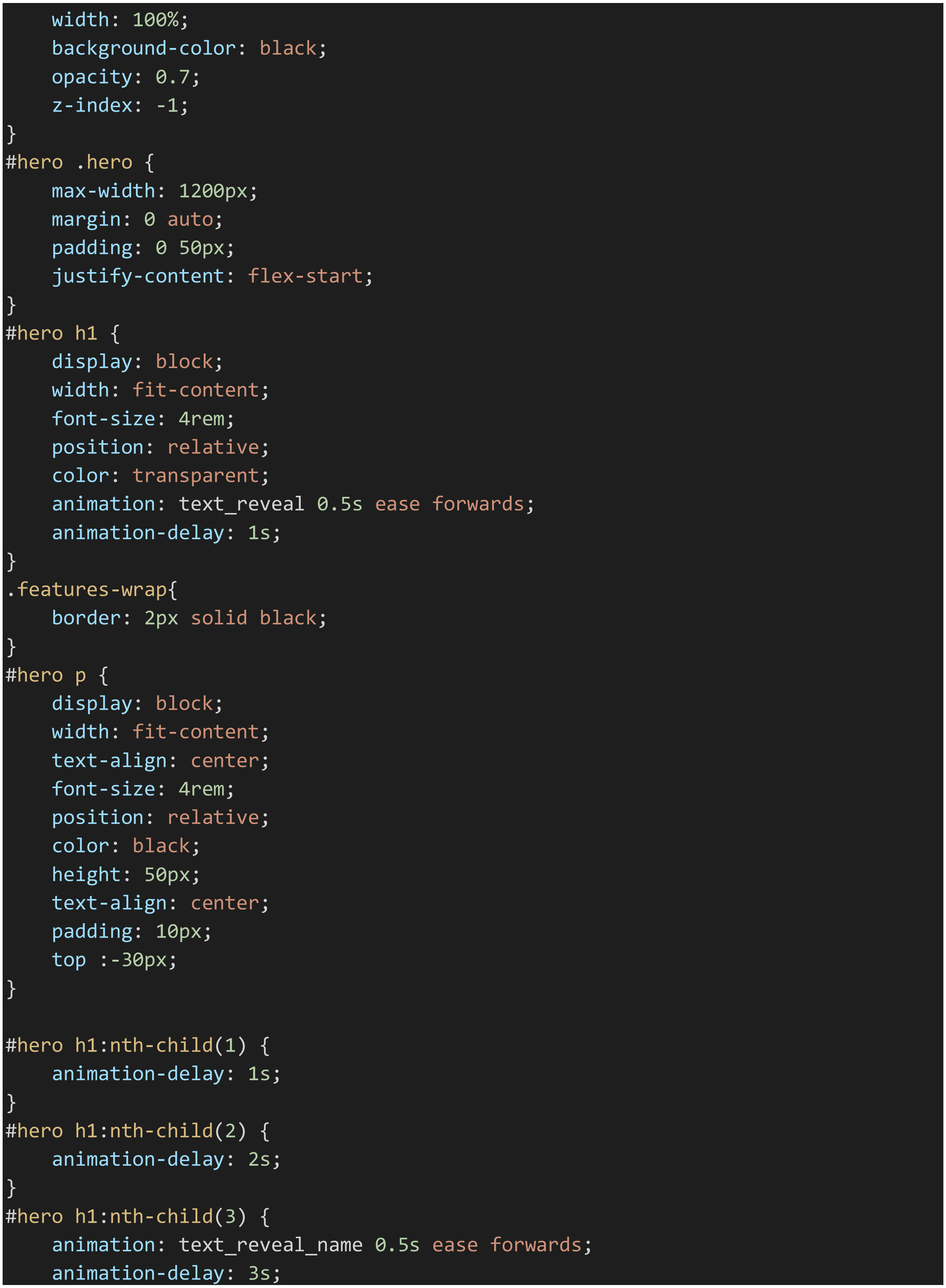
```css
    width: 100vw;
    height: auto;
}
#header .header {
    min-height: 8vh;
    background-color: rgba(31, 30, 30, 0.24);
    transition: 0.3s ease background-color;
}
#header .nav-bar {
    display: flex;
    align-items: center;
    justify-content: space-between;
    width: 100%;
    height: 100%;
    max-width: 1300px;
    padding: 0 10px;
}
#header .nav-list ul {
    list-style: none;
    position: absolute;
    background-color: rgb(31, 30, 30);
    width: 100vw;
    height: 100vh;
    left: 100%;
    top: 0;
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    z-index: 1;
    overflow-x: hidden;
    transition: 0.5s ease left;
}
#header .nav-list ul.active {
    left: 0%;
}
#header .nav-list ul a {
    font-size: 2.5rem;
    font-weight: 500;
    letter-spacing: 0.2rem;
    text-decoration: none;
    color: white;
    text-transform: uppercase;
    padding: 20px;
    display: block;
```



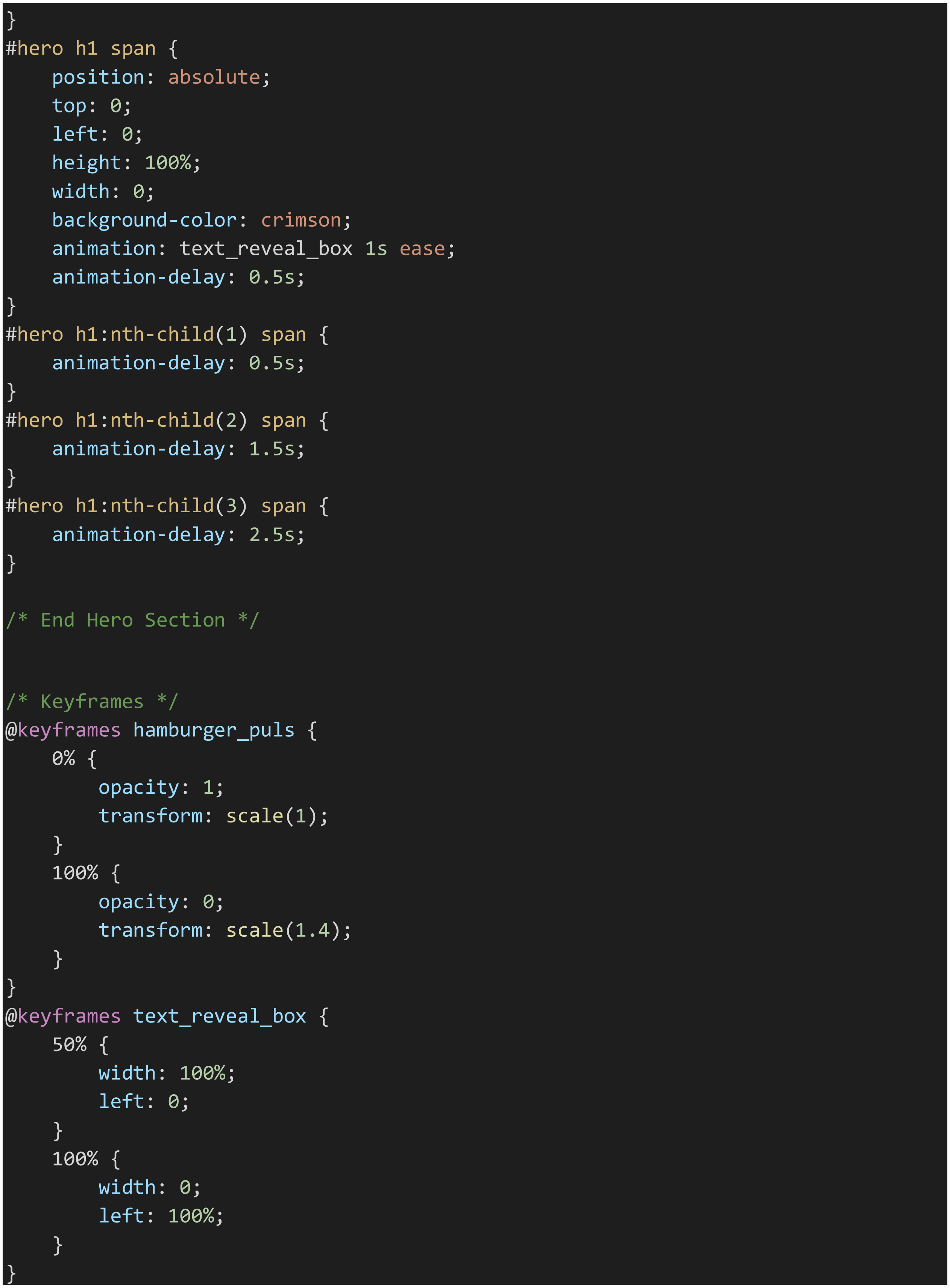
```
}
#header .nav-list ul a::after {
    content: attr(data-after);
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%) scale(0);
    color: rgba(240, 248, 255, 0.021);
    font-size: 13rem;
    letter-spacing: 50px;
    z-index: -1;
    transition: 0.3s ease letter-spacing;
}
#header .nav-list ul li:hover a::after {
    transform: translate(-50%, -50%) scale(1);
    letter-spacing: initial;
}
#header .nav-list ul li:hover a {
    color: crimson;
}
#header .hamburger {
    height: 60px;
    width: 60px;
    display: inline-block;
    border: 3px solid white;
    border-radius: 50%;
    position: relative;
    display: flex;
    align-items: center;
    justify-content: center;
    z-index: 100;
    cursor: pointer;
    transform: scale(0.8);
    margin-right: 20px;
}
#header .hamburger:after {
    position: absolute;
    content: '';
    height: 100%;
    width: 100%;
    border-radius: 50%;
    border: 3px solid white;
    animation: hamburger_puls 1s ease infinite;
}
#header .hamburger .bar {
```



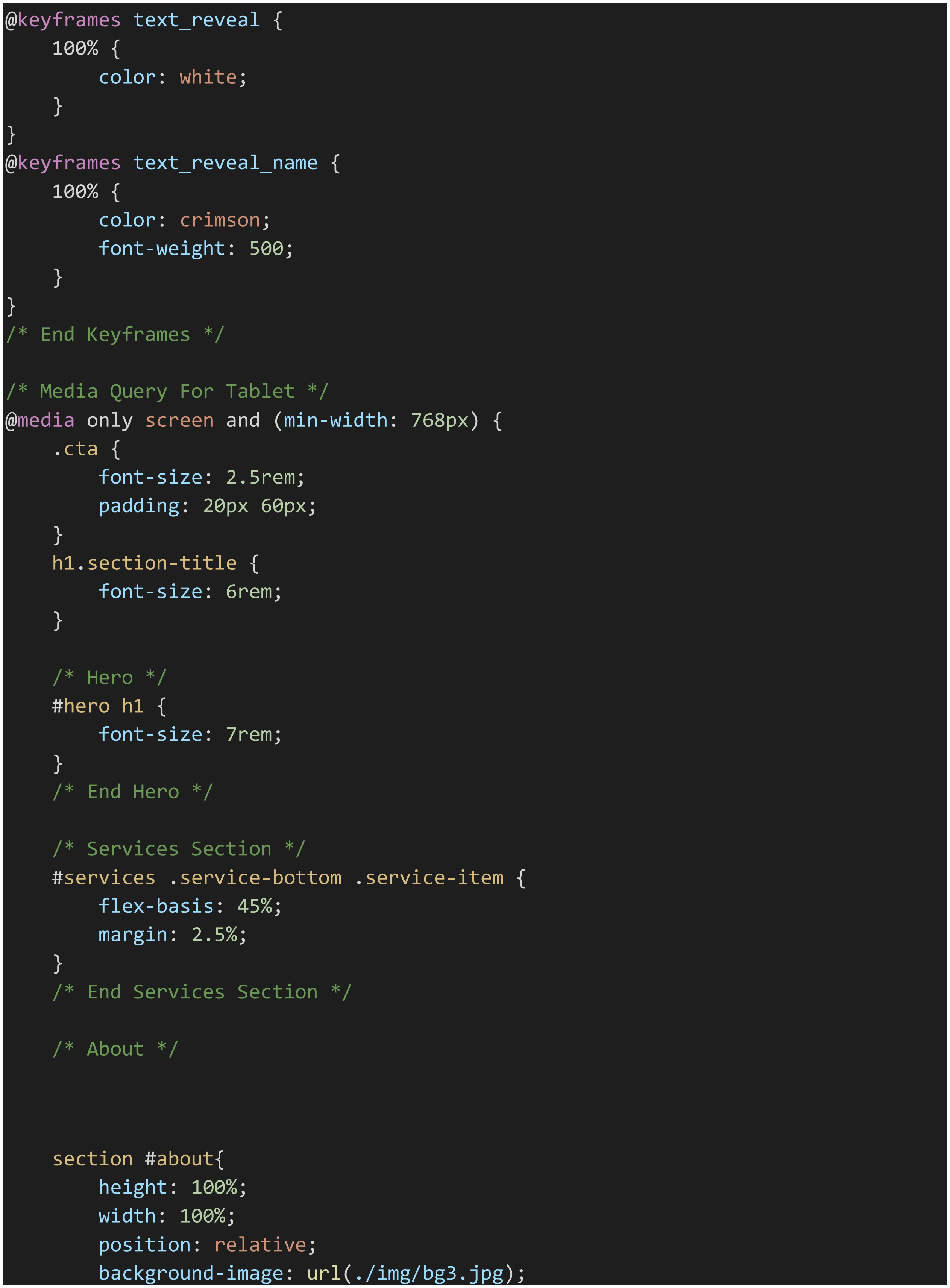
```
    height: 2px;
    width: 30px;
    position: relative;
    background-color: white;
    z-index: -1;
}
#header .hamburger .bar::after,
#header .hamburger .bar::before {
    content: '';
    position: absolute;
    height: 100%;
    width: 100%;
    left: 0;
    background-color: white;
    transition: 0.3s ease;
    transition-property: top, bottom;
}
#header .hamburger .bar::after {
    top: 8px;
}
#header .hamburger .bar::before {
    bottom: 8px;
}
#header .hamburger.active .bar::before {
    bottom: 0;
}
#header .hamburger.active .bar::after {
    top: 0;
}
/* End Header section */

/* Hero Section */
#hero {
    background-image: url(./img/www.jpg);
    background-size: cover;
    background-position: top center;
    position: relative;
    z-index: 1;
}
#hero::after {
    content: '';
    position: absolute;
    left: 0;
    top: 0;
    height: 100%;
```



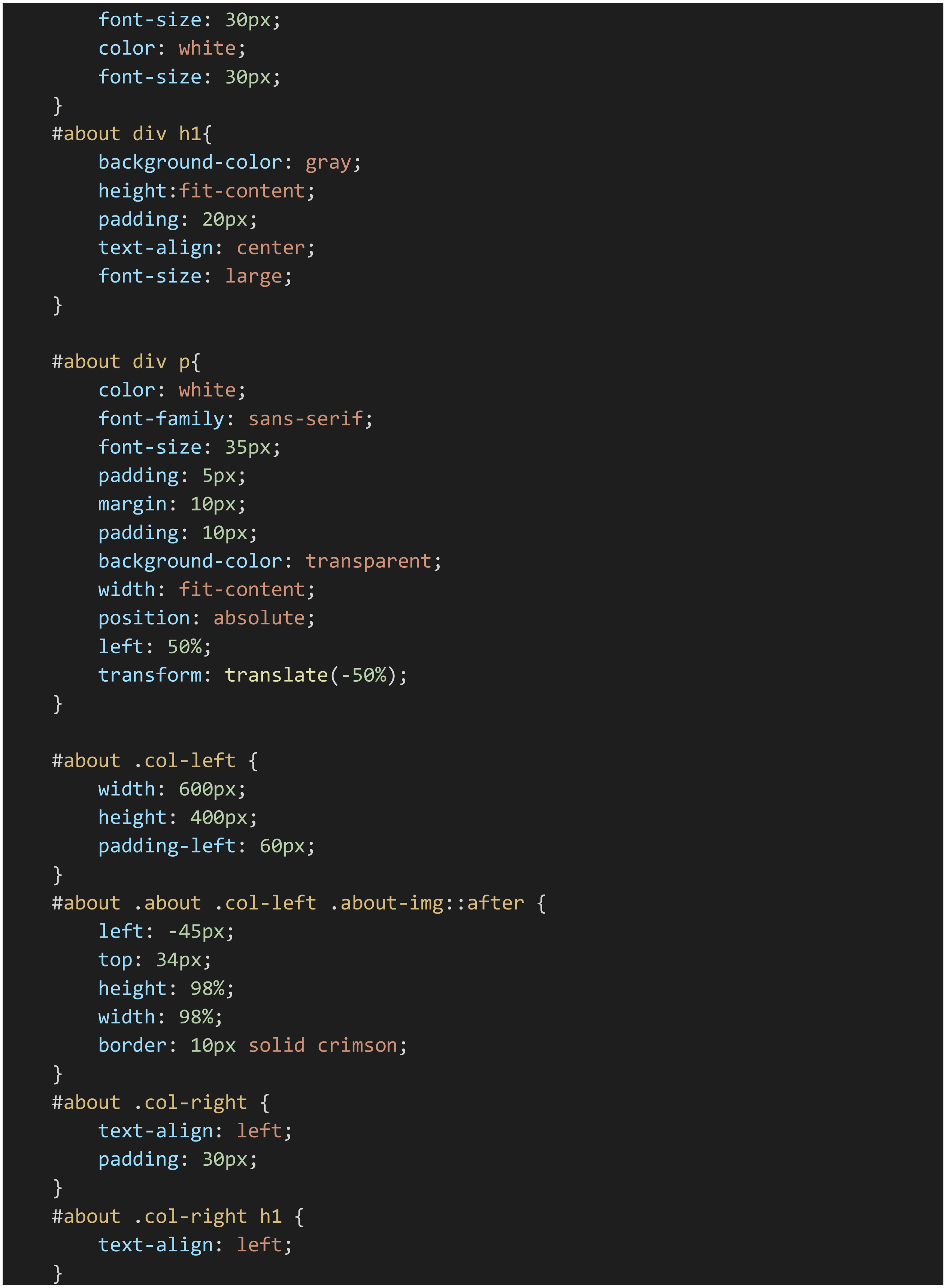
```css
    width: 100%;
    background-color: black;
    opacity: 0.7;
    z-index: -1;
}
#hero .hero {
    max-width: 1200px;
    margin: 0 auto;
    padding: 0 50px;
    justify-content: flex-start;
}
#hero h1 {
    display: block;
    width: fit-content;
    font-size: 4rem;
    position: relative;
    color: transparent;
    animation: text_reveal 0.5s ease forwards;
    animation-delay: 1s;
}
.features-wrap{
    border: 2px solid black;
}
#hero p {
    display: block;
    width: fit-content;
    text-align: center;
    font-size: 4rem;
    position: relative;
    color: black;
    height: 50px;
    text-align: center;
    padding: 10px;
    top :-30px;
}

#hero h1:nth-child(1) {
    animation-delay: 1s;
}
#hero h1:nth-child(2) {
    animation-delay: 2s;
}
#hero h1:nth-child(3) {
    animation: text_reveal_name 0.5s ease forwards;
    animation-delay: 3s;
```



```css
}
#hero h1 span {
    position: absolute;
    top: 0;
    left: 0;
    height: 100%;
    width: 0;
    background-color: crimson;
    animation: text_reveal_box 1s ease;
    animation-delay: 0.5s;
}
#hero h1:nth-child(1) span {
    animation-delay: 0.5s;
}
#hero h1:nth-child(2) span {
    animation-delay: 1.5s;
}
#hero h1:nth-child(3) span {
    animation-delay: 2.5s;
}

/* End Hero Section */


/* Keyframes */
@keyframes hamburger_puls {
    0% {
        opacity: 1;
        transform: scale(1);
    }
    100% {
        opacity: 0;
        transform: scale(1.4);
    }
}
@keyframes text_reveal_box {
    50% {
        width: 100%;
        left: 0;
    }
    100% {
        width: 0;
        left: 100%;
    }
}
```



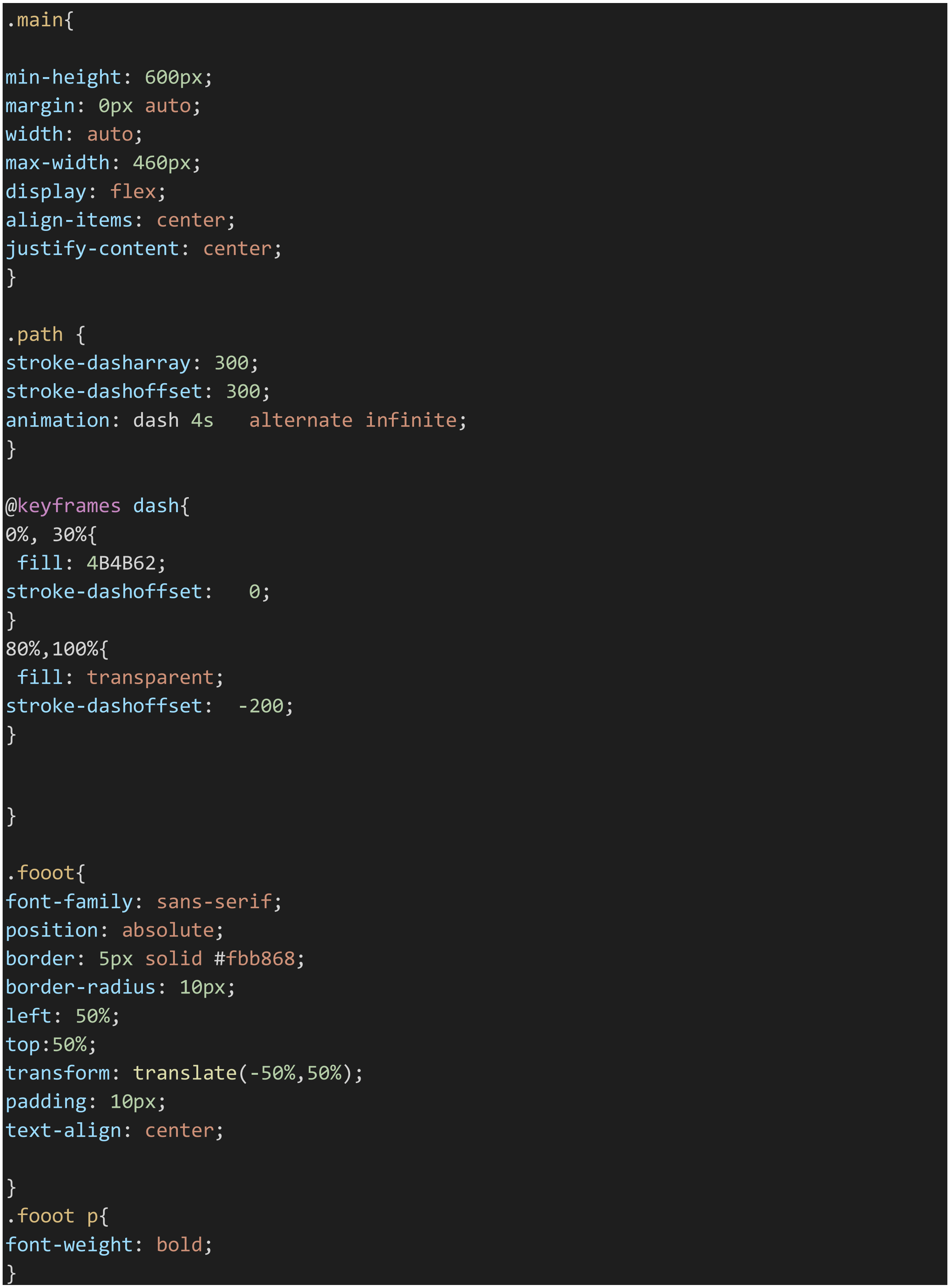
```css
@keyframes text_reveal {
    100% {
        color: white;
    }
}
@keyframes text_reveal_name {
    100% {
        color: crimson;
        font-weight: 500;
    }
}
/* End Keyframes */

/* Media Query For Tablet */
@media only screen and (min-width: 768px) {
    .cta {
        font-size: 2.5rem;
        padding: 20px 60px;
    }
    h1.section-title {
        font-size: 6rem;
    }

    /* Hero */
    #hero h1 {
        font-size: 7rem;
    }
    /* End Hero */

    /* Services Section */
    #services .service-bottom .service-item {
        flex-basis: 45%;
        margin: 2.5%;
    }
    /* End Services Section */

    /* About */



    section #about{
        height: 100%;
        width: 100%;
        position: relative;
        background-image: url(./img/bg3.jpg);
```



```
        font-size: 30px;
        color: white;
        font-size: 30px;
    }
    #about div h1{
        background-color: gray;
        height:fit-content;
        padding: 20px;
        text-align: center;
        font-size: large;
    }

    #about div p{
        color: white;
        font-family: sans-serif;
        font-size: 35px;
        padding: 5px;
        margin: 10px;
        padding: 10px;
        background-color: transparent;
        width: fit-content;
        position: absolute;
        left: 50%;
        transform: translate(-50%);
    }

    #about .col-left {
        width: 600px;
        height: 400px;
        padding-left: 60px;
    }
    #about .about .col-left .about-img::after {
        left: -45px;
        top: 34px;
        height: 98%;
        width: 98%;
        border: 10px solid crimson;
    }
    #about .col-right {
        text-align: left;
        padding: 30px;
    }
    #about .col-right h1 {
        text-align: left;
    }
    /* End About */
}
/* End Media Query For Tablet */

/* Media Query For Desktop */
@media only screen and (min-width: 1200px) {
    /* header */
    #header .hamburger {
        display: none;
    }
    #header .nav-list ul {
        position: initial;
        display: block;
        height: auto;
        width: fit-content;
        background-color: transparent;
    }
    #header .nav-list ul li {
        display: inline-block;
    }
    #header .nav-list ul li a {
        font-size: 1.8rem;
    }
    #header .nav-list ul a:after {
        display: none;
    }
    /* End header */

    #services .service-bottom .service-item {
        flex-basis: 22%;
        margin: 1.5%;
    }
}
/* End  Media Query For Desktop */
```

## Style.css 404 :

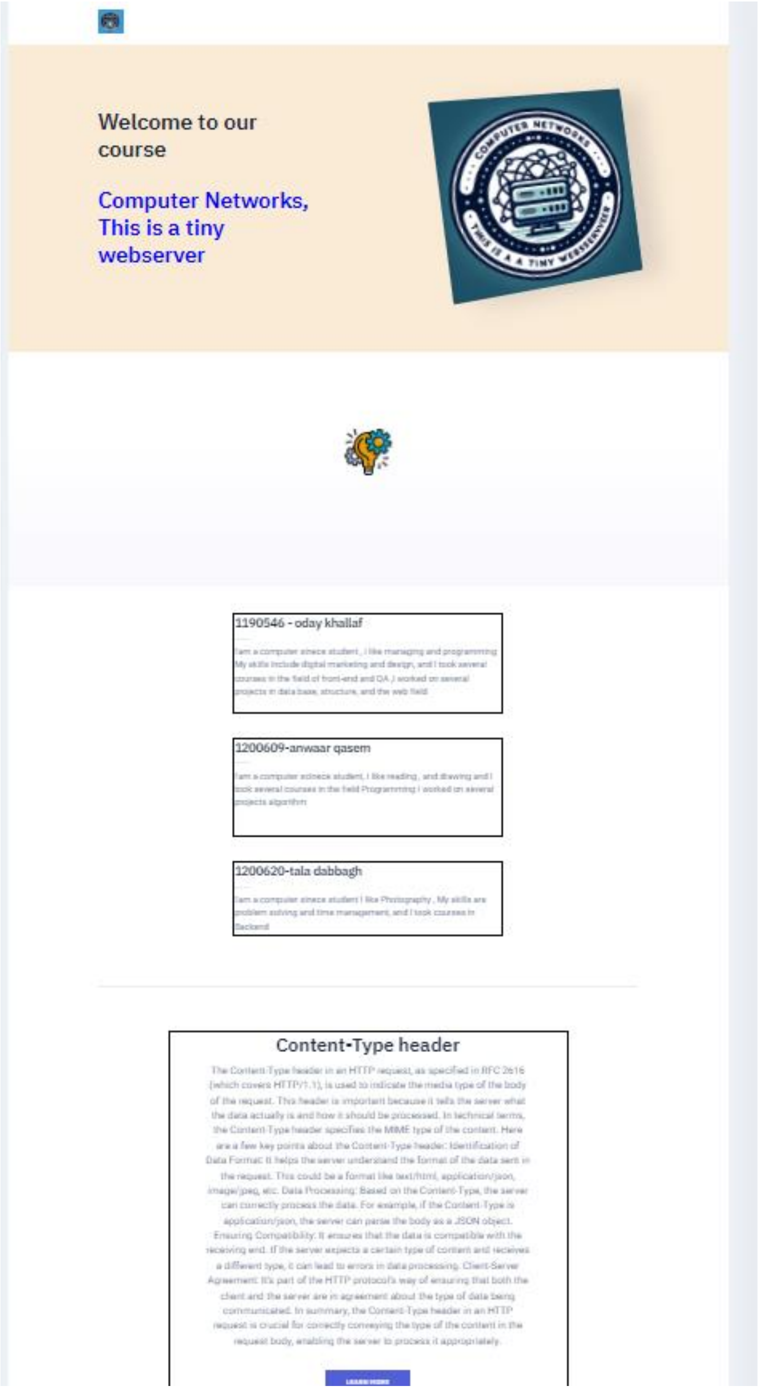
```
.fundo{
  animation: scales 3s alternate  infinite;
  transform-origin: center;

}
```



```
.main{

min-height: 600px;
margin: 0px auto;
width: auto;
max-width: 460px;
display: flex;
align-items: center;
justify-content: center;
}

.path {
stroke-dasharray: 300;
stroke-dashoffset: 300;
animation: dash 4s   alternate infinite;
}

@keyframes dash{
0%, 30%{
 fill: 4B4B62;
stroke-dashoffset:   0;
}
80%,100%{
 fill: transparent;
stroke-dashoffset:  -200;
}


}

.fooot{
font-family: sans-serif;
position: absolute;
border: 5px solid #fbb868;
border-radius: 10px;
left: 50%;
top:50%;
transform: translate(-50%,50%);
padding: 10px;
text-align: center;

}
.fooot p{
font-weight: bold;
}
```

[redacted]

main_en page :



Welcome to our course

Computer Networks, This is a tiny webserver

1190546 - oday khallaf

I am a computer sinece student , I like managing and programming My skills include digital marketing and design, and I took several courses in the field of front-end and QA ,I worked on several projects in data base, structure, and the web field

1200609-anwaar qasem

I am a computer scineck student, I like reading , and drawing and I took several courses in the field Programming I worked on several projects algorithm

1200620-tala dabbagh

I am a computer sinece student I like Photography , My skills are problem solving and time management, and I took courses in Backend

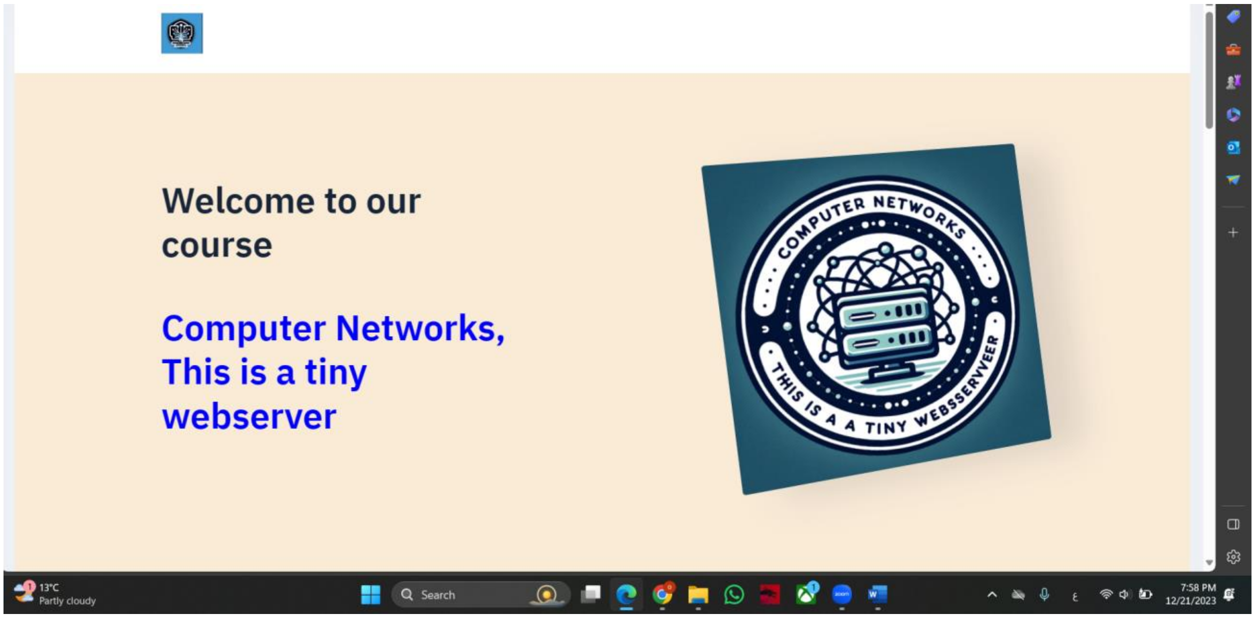
Content-Type header

The Content-Type header in an HTTP request, as specified in RFC 2616 (which covers HTTP/1.1), is used to indicate the media type of the body of the request. This header is important because it tells the server what the data actually is and how it should be processed. In technical terms, the Content-Type header specifies the MIME type of the content. Here are a few key points about the Content-Type header: Identification of Data Format: It helps the server understand the format of the data sent in the request. This could be a format like text/html, application/json, image/jpeg, etc. Data Processing: Based on the Content-Type, the server can correctly process the data. For example, if the Content-Type is application/json, the server can parse the body as a JSON object. Ensuring Compatibility: It ensures that the data is compatible with the receiving end. If the server expects a certain type of content and receives a different type, it can lead to errors in data processing. Client-Server Agreement: It's part of the HTTP protocol's way of ensuring that both the client and the server are in agreement about the type of data being communicated. In summary, the Content-Type header in an HTTP request is crucial for correctly conveying the type of the content in the request body, enabling the server to process it appropriately.

## Content-Type header

The Content-Type header in an HTTP request, as specified in RFC 2616 (which covers HTTP/1.1), is used to indicate the media type of the body of the request. This header is important because it tells the server what the data actually is and how it should be processed. In technical terms, the Content-Type header specifies the MIME type of the content. Here are a few key points about the Content-Type header: Identification of Data Format: It helps the server understand the format of the data sent in the request. This could be a format like text/html, application/json, image/jpeg, etc. Data Processing: Based on the Content-Type, the server can correctly process the data. For example, if the Content-Type is application/json, the server can parse the body as a JSON object. Ensuring Compatibility: It ensures that the data is compatible with the receiving end. If the server expects a certain type of content and receives a different type, it can lead to errors in data processing. Client-Server Agreement: It's part of the HTTP protocol's way of ensuring that both the client and the server are in agreement about the type of data being communicated. In summary, the Content-Type header in an HTTP request is crucial for correctly conveying the type of the content in the request body, enabling the server to process it appropriately.

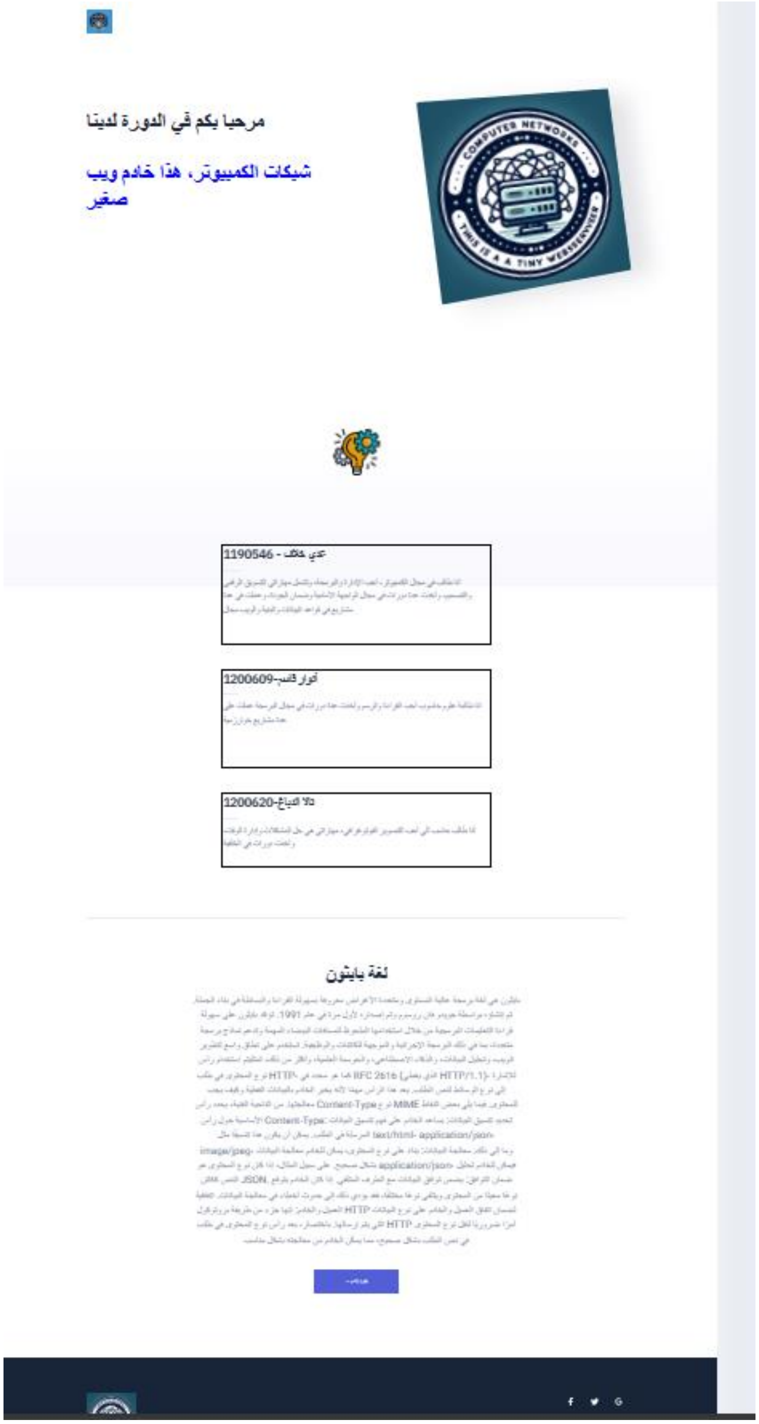
LEARN MORE



Welcome to our course

Computer Networks, This is a tiny webserver

Main_ ar page :



مرحبا بكم في الدورة لدينا

شبكات الكمبيوتر، هذا خادم ويب صغير

**عدي خالد - 1190546**

انا طالب في مجال الكمبيوتر، احب الإدارة والبرمجة وتشمل مهاراتي التسويق الرقمي والتصميم، وأخذت عدة دورات في مجال البرمجة الأساسية وضمان الجودة، وعملت في عدة مشاريع في قواعد البيانات [illegible]

**أنوار قاسم-1200609**

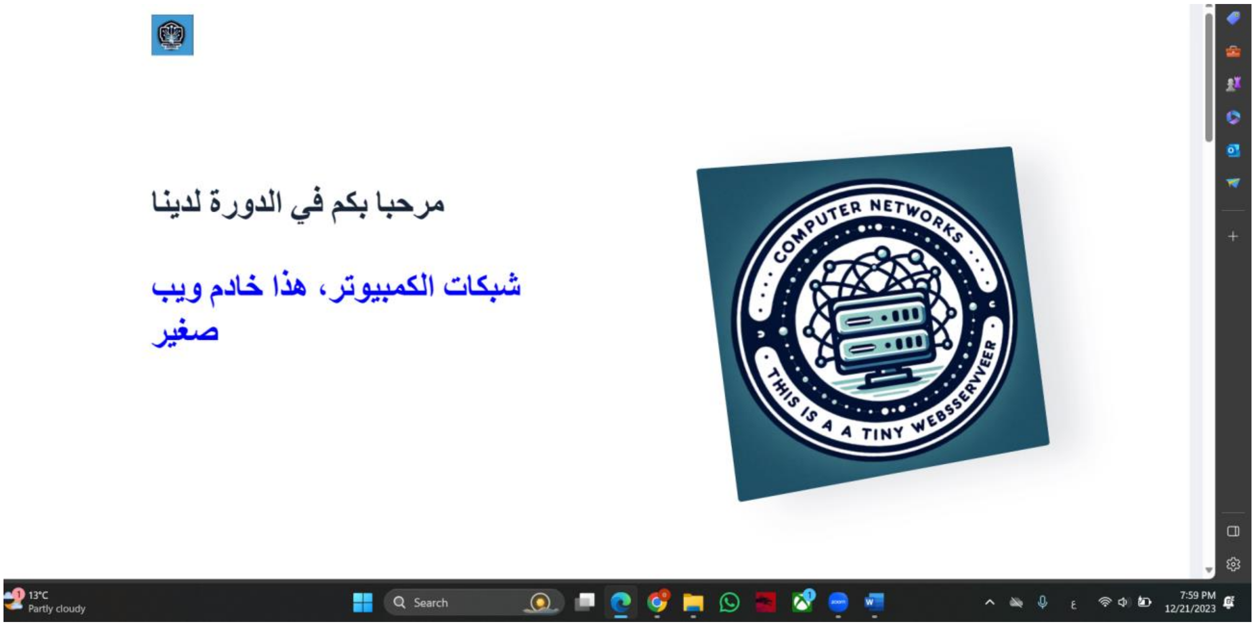
انا طالبة علوم حاسوب احب القراءة والرسم وأخذت عدة دورات في مجال البرمجة عملت على عدة مشاريع خوارزمية

**دالا الدباغ-1200620**

انا طالب حاسب الي احب التصوير الفوتوغرافي، مهاراتي هي حل المشكلات وإدارة الوقت وأخذت دورات في الثقافة

## لغة بايثون

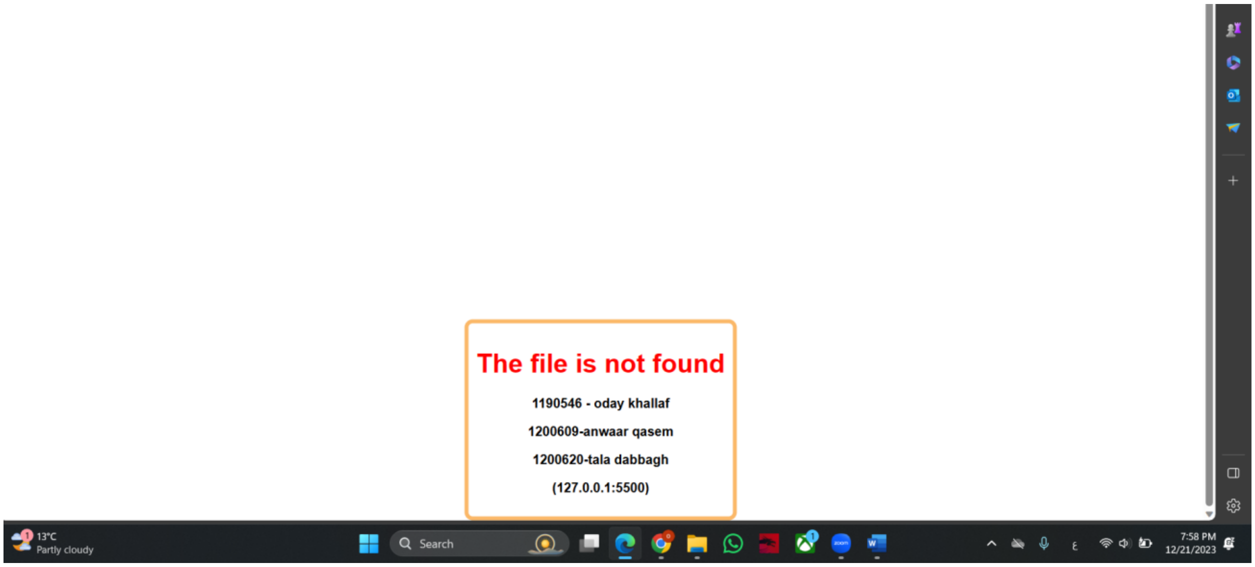
بايثون هي لغة برمجة عالية المستوى ومتعددة الأغراض معروفة بسهولة القراءة والبساطة في بناء الجملة تم إنشاؤها بواسطة جويدو فان روسوم وتم إصدارها لأول مرة في عام 1991. تؤكد بايثون على سهولة قراءة التعليمات البرمجية من خلال استخدامها الملحوظ للمسافات البيضاء المهمة وتدعم نماذج برمجة متعددة، بما في ذلك البرمجة الإجرائية والموجهة للكائنات والوظيفية. تستخدم على نطاق واسع لتطوير الويب وتحليل البيانات والذكاء الاصطناعي والحوسبة العلمية، والكثير من ذلك. [illegible] استخدام رأس للإشارة إلى{HTTP/1.1 الذي يعني} RFC 2616 كما هو محدد في HTTP، نوع المحتوى في طلب إلى نوع الوسائط لنص الطلب. يعد هذا الرأس مهمًا لأنه يخبر الخادم بالبيانات الفعلية وكيف يجب المحتوى هنا يلي بعض النقاط MIME نوع Content-Type معالجتها. من الناحية الفنية، يحدد رأس تحديد تنسيق البيانات: يساعد الخادم على فهم تنسيق البيانات :Content-Type الأساسية حول رأس المرسلة في الطلب. يمكن أن يكون هذا تنسيقًا مثل text/html، application/json، ربما إلى تكامل معالجة البيانات: بناءً على نوع المحتوى، يمكن للخادم معالجة البيانات. image/jpeg، فيمكن للخادم تحليل application/json بشكل صحيح. على سبيل المثال، إذا كان نوع المحتوى هو ضمان التوافق: يضمن توافق البيانات مع الطرف المتلقي. إذا كان الخادم يتوقع JSON، النص. كان نوعًا معينًا من المحتوى ويتلقى نوعًا مختلفًا، فقد يؤدي ذلك إلى حدوث أخطاء في معالجة البيانات. الفعلية لضمان اتفاق العميل والخادم على نوع البيانات HTTP العميل والخادم: إنها جزء من طريقة بروتوكول أمرًا ضروريًا لنقل نوع المحتوى HTTP التي يتم إرسالها. باختصار، يعد رأس نوع المحتوى في طلب في نص الطلب بشكل صحيح، مما يمكن الخادم من معالجته بشكل مناسب.



مرحبا بكم في الدورة لدينا

شبكات الكمبيوتر، هذا خادم ويب صغير

Not Found Page:



The file is not found

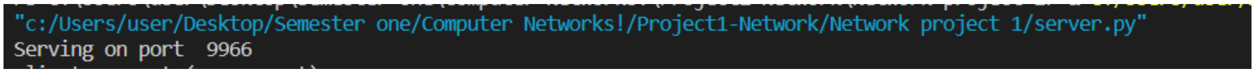
1190546 - oday khallaf

1200609-anwaar qasem

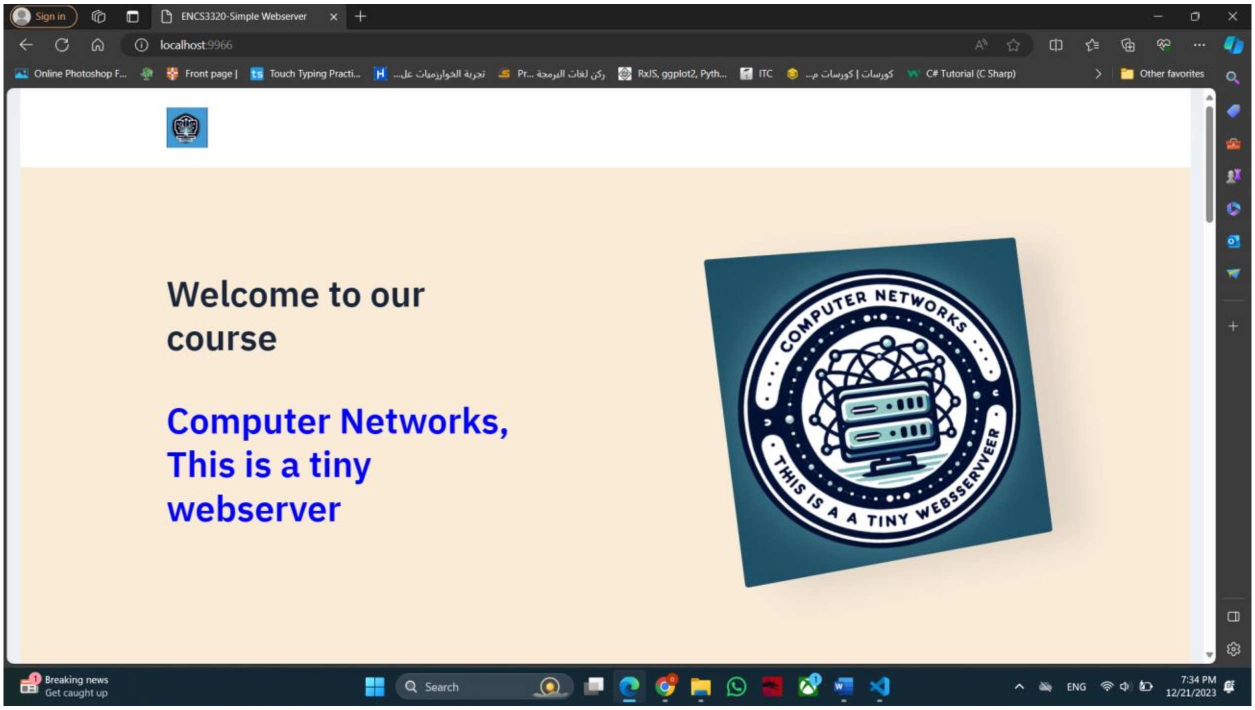
1200620-tala dabbagh

(127.0.0.1:5500)

# Runs :



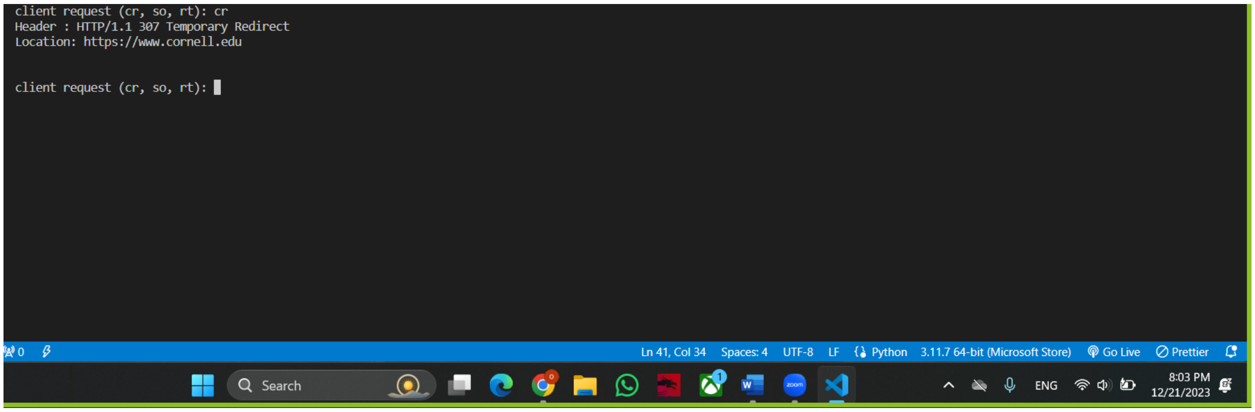
```
"c:/Users/user/Desktop/Semester one/Computer Networks!/Project1-Network/Network project 1/server.py"
Serving on port  9966
```



Welcome to our course

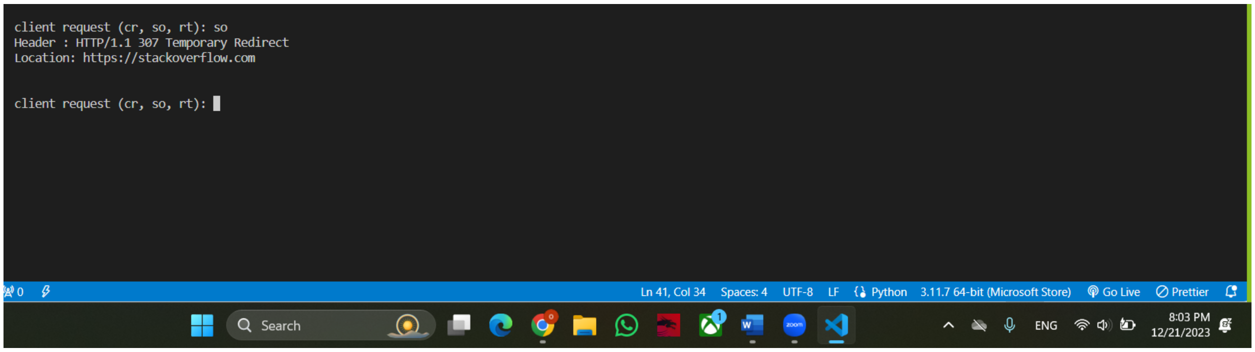
Computer Networks, This is a tiny webserver

Cr:



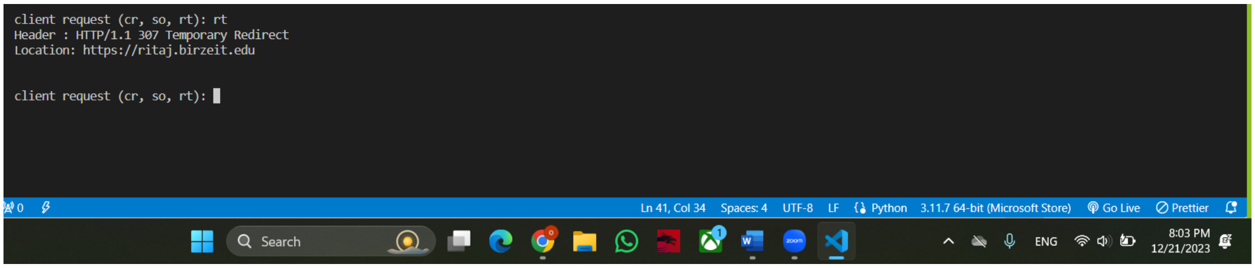
```
client request (cr, so, rt): cr
Header : HTTP/1.1 307 Temporary Redirect
Location: https://www.cornell.edu


client request (cr, so, rt): 
```

So:



```
client request (cr, so, rt): so
Header : HTTP/1.1 307 Temporary Redirect
Location: https://stackoverflow.com


client request (cr, so, rt): 
```

Rt:



```
client request (cr, so, rt): rt
Header : HTTP/1.1 307 Temporary Redirect
Location: https://ritaj.birzeit.edu


client request (cr, so, rt):
```