Task 3

At first this is the complete code that I created:

```
# import socket module
from socket import *
import sys  # In order to terminate the program
def main():
    serverSocket = socket(AF_INET, SOCK_STREAM)
    # Prepare a sever socket
    serverPort = 12000
    serverSocket.bind(('', serverPort))
    serverSocket.listen(1)
    # Fill in start
    # Fill in end
    while True:
        # Establish the connection
        print('Ready to serve...')
        connectionSocket, addr = serverSocket.accept()
                        # Fill in start              #Fill in end
        try:
            message = connectionSocket.recv(2048).decode()  # Fill in
start           #Fill in end
            print(message)
            filename = message.split()[1]
            f = open(filename[1:])
            outputdata = f.read()  # Fill in start       #Fill in end
            connectionSocket.send('\nHTTP/1.1 200 OK\n\n'.encode())
            print(outputdata)
            # Fill in start
            # Fill in end
            # Send the content of the requested file to the client
            for i in range(0, len(outputdata)):
                connectionSocket.send(outputdata[i].encode())
            connectionSocket.send("\r\n".encode())

            connectionSocket.close()
        except IOError:
    # Send response message for file not found
            print('HTTP/1.1 404 Not found')
            connectionSocket.send('HTTP/1.1 404 Not found'.encode())
    # Fill in start
    # Fill in end
    # Close client socket
            connectionSocket.close()
    # Fill in start
    # Fill in end
    serverSocket.close()
    sys.exit()  # Terminate the program after sending the
corresponding data
if __name__ == '__main__':
    main()
```

So after running the code to the TCP server I printed the message and I got the a response this is the response:

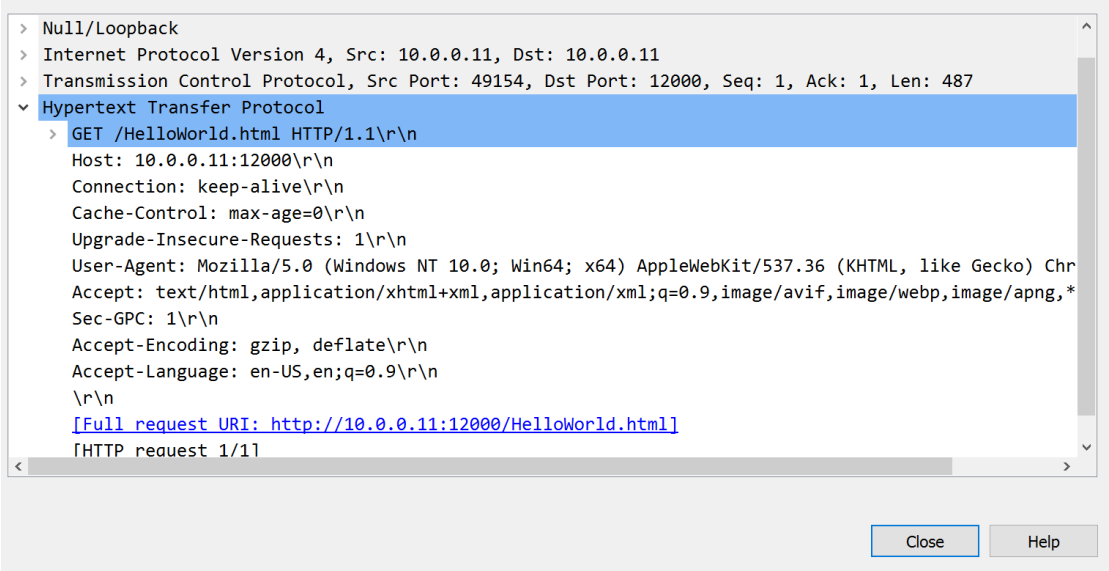
As we can see in the picture the host is my Wi-Fi Ip 10.0.0.11 and the port that I gave to the socket (12000 it doesn't matter this is the pore that we saw on practice it can be another number).

The next picture will present how it looks like in the Wireshark:

The Wireshark main window(The message is the first green) :

4	0.095395	127.0.0.1	127.0.0.1	TCP	44 55102 → 55101 [ACK] Seq=1 Ack=2 Win=10223 Len=0
5	3.332396	10.0.0.11	10.0.0.11	TCP	56 49154 → 12000 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
6	3.332452	10.0.0.11	10.0.0.11	TCP	56 12000 → 49154 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
7	3.332516	10.0.0.11	10.0.0.11	TCP	44 49154 → 12000 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
8	3.332665	10.0.0.11	10.0.0.11	HTTP	531 GET /HelloWorld.html HTTP/1.1
9	3.332687	10.0.0.11	10.0.0.11	TCP	44 12000 → 49154 [ACK] Seq=1 Ack=488 Win=2619648 Len=0
10	3.333445	10.0.0.11	10.0.0.11	HTTP	62 Continuation
11	3.333484	10.0.0.11	10.0.0.11	TCP	44 49154 → 12000 [ACK] Seq=488 Ack=19 Win=2619648 Len=0
12	3.333515	10.0.0.11	10.0.0.11	TCP	45 12000 → 49154 [PSH, ACK] Seq=19 Ack=488 Win=2619648 Len=1 [TCP segment of a reassembled PDU]
13	3.333522	10.0.0.11	10.0.0.11	TCP	44 49154 → 12000 [ACK] Seq=488 Ack=20 Win=2619648 Len=0
14	3.333530	10.0.0.11	10.0.0.11	TCP	45 12000 → 49154 [PSH, ACK] Seq=20 Ack=488 Win=2619648 Len=1 [TCP segment of a reassembled PDU]
15	3.333536	10.0.0.11	10.0.0.11	TCP	44 49154 → 12000 [ACK] Seq=488 Ack=21 Win=2619648 Len=0

This is the message in the Wireshark:



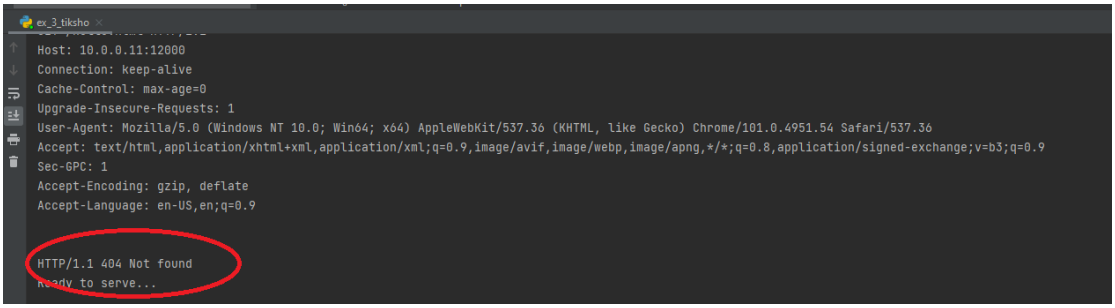
This is the recording:

WiresharkHttpgood.link.pcapng

This picture presents the html fie that I opened in my browser as requested in the task.

My HelloWorld file for the task

After running the http file, I tried to run a file which is not in the server directory, I tried the same thing but with Hello.html instead of HelloWorld.html and I got the not found message that I wrote in the exception in the code.



This is how the Wireshark looks:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	127.0.0.1	127.0.0.1	TCP	45	55101 → 55102 [PSH, ACK] Seq=1 Ack=1 Win=10233 Len=1
2	0.000065	127.0.0.1	127.0.0.1	TCP	44	55102 → 55101 [ACK] Seq=1 Ack=2 Win=10223 Len=0
3	11.815985	10.0.0.11	10.0.0.11	TCP	56	49687 → 12000 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
4	11.816086	10.0.0.11	10.0.0.11	TCP	56	12000 → 49687 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
5	11.816155	10.0.0.11	10.0.0.11	TCP	44	49687 → 12000 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
6	11.816642	10.0.0.11	10.0.0.11	HTTP	526	GET /Hello.html HTTP/1.1
7	11.816667	10.0.0.11	10.0.0.11	TCP	44	12000 → 49687 [ACK] Seq=1 Ack=483 Win=2619648 Len=0
8	11.816995	10.0.0.11	10.0.0.11	TCP	66	12000 → 49687 [PSH, ACK] Seq=1 Ack=483 Win=2619648 Len=22 [TCP segment of a reassembled PDU]
9	11.817025	10.0.0.11	10.0.0.11	TCP	44	49687 → 12000 [ACK] Seq=483 Ack=23 Win=2619648 Len=0
10	11.817052	10.0.0.11	10.0.0.11	TCP	44	12000 → 49687 [FIN, ACK] Seq=23 Ack=483 Win=2619648 Len=0
11	11.817063	10.0.0.11	10.0.0.11	TCP	44	49687 → 12000 [ACK] Seq=483 Ack=24 Win=2619648 Len=0
12	11.817640	10.0.0.11	10.0.0.11	TCP	44	49687 → 12000 [FIN, ACK] Seq=483 Ack=24 Win=2619648 Len=0
13	11.817662	10.0.0.11	10.0.0.11	TCP	44	12000 → 49687 [ACK] Seq=24 Ack=484 Win=2619648 Len=0
14	18.391821	127.0.0.1	127.0.0.1	TCP	45	55104 → 55105 [PSH, ACK] Seq=1 Ack=1 Win=10233 Len=1
15	18.391888	127.0.0.1	127.0.0.1	TCP	44	55105 → 55104 [ACK] Seq=1 Ack=2 Win=10232 Len=0
16	18.467800	127.0.0.1	127.0.0.1	TCP	45	55104 → 55105 [PSH, ACK] Seq=2 Ack=1 Win=10233 Len=1
17	18.467863	127.0.0.1	127.0.0.1	TCP	44	55105 → 55104 [ACK] Seq=1 Ack=3 Win=10232 Len=0

This is the recording:

WiresharknotFound.link.pcapng

And this is what I got in the browser: