Networks Programming Assignment 1

Badr Elsayed - 22010664
AbdelRahman Amgad - 22010871

The code:

```
from socket import *
import sys

if len(sys.argv) <= 1:
    print('Usage : "python ProxyServer.py server_ip"\n[server_ip : It
is the IP Address Of Proxy Server')
    sys.exit(2)

# Create a server socket, bind it to a port and start listening
tcpSerSock = socket(AF_INET, SOCK_STREAM)

# Fill in start.
tcpSerSock.bind((sys.argv[1], 8888))
tcpSerSock.listen(5)
# Fill in end.

while 1:
    # Start receiving data from the client
    print('Ready to serve...')
    tcpCliSock, addr = tcpSerSock.accept()
    print('Received a connection from:', addr)

    # Fill in start.
    message = tcpCliSock.recv(4096).decode()
    # Fill in end.
    # print(message)

    # Bonus
    method = message.split()[0]

    # Extract the filename from the given message
    print(message.split()[1])
    filename = message.split()[1].partition("/")[2]
```

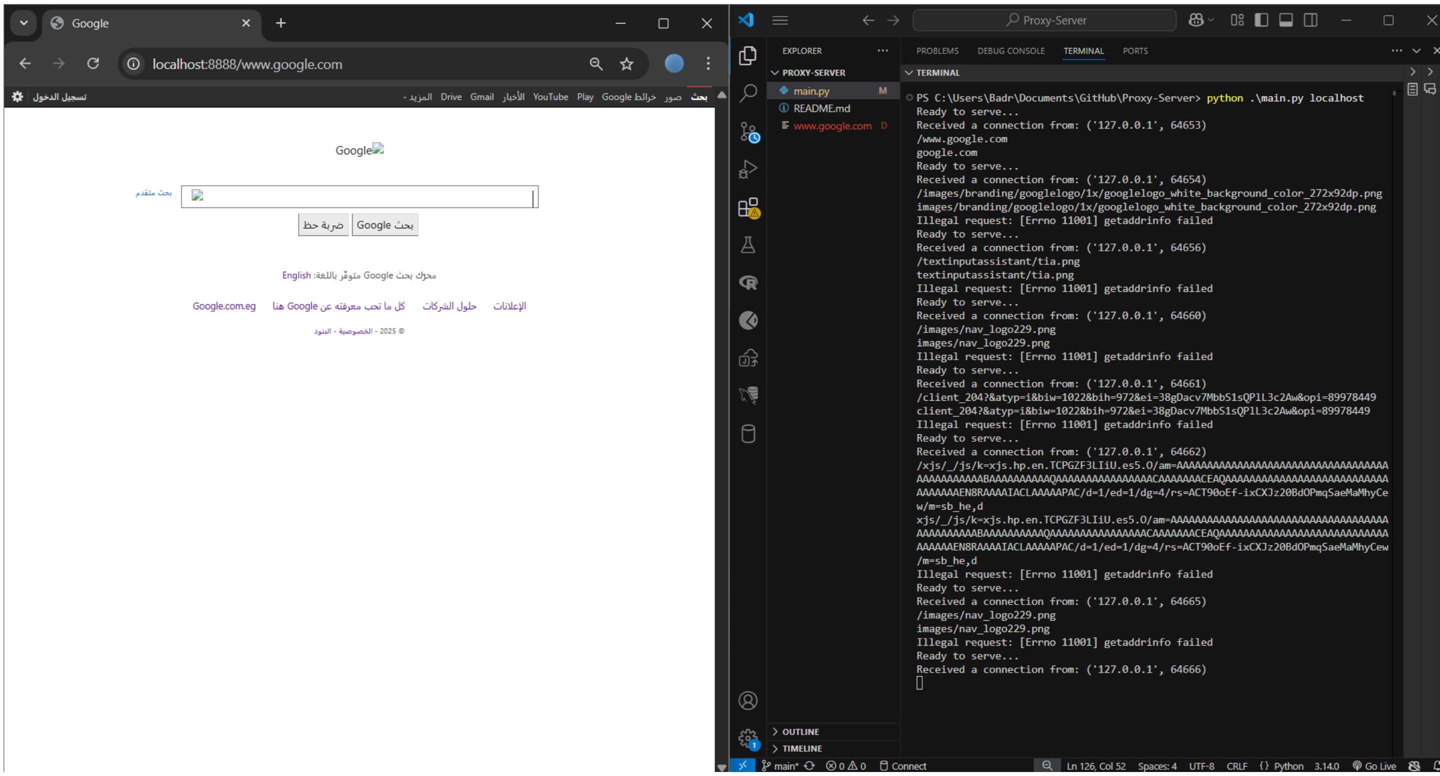
```
    # print(filename)

    # Bonus
    post_body = ""
    if method == "POST" and "\r\n\r\n" in message:
        post_body = message.split("\r\n\r\n", 1)[1]

    fileExist = "false"
    filetouse = "/" + filename
    # print(filetouse)

    # Bonus
    if method == "GET":
        try:
            # Check whether the file exist in the cache
            f = open(filetouse[1:], "r")
            outputdata = f.readlines()
            fileExist = "true"

            # ProxyServer finds a cache hit and generates a response
message
            tcpCliSock.send("HTTP/1.0 200 OK\r\n".encode())
            tcpCliSock.send("Content-Type:text/html\r\n".encode())

            # Fill in start.
            for line in outputdata:
                tcpCliSock.send(line.encode())
            # Fill in end.
            print('Read from cache')

        # Error handling for file not found in cache
        except IOError:
            pass

    # Error handling for file not found in cache
    if fileExist == "false":
        # Create a socket on the proxyserver
        # Fill in start.
        c = socket(AF_INET, SOCK_STREAM)
```

```
        # Fill in end.

        hostn = filename.replace("www.", "", 1)
        print(hostn)

        try:
            # Connect to the socket to port 80
            # Fill in start.
            c.connect((hostn, 80))
            # Fill in end.

            # Create a temporary file on this socket and ask port 80
            # for the file requested by the client
            fileobj = c.makefile('rwb', 0)

            # Bonus: Build request based on method
            if method == "POST":
                content_length = len(post_body.encode())
                request = f"POST /{filename.split('/', 1)[1] if '/'
in filename else ''} HTTP/1.0\r\n"
                request += f"Host: {hostn}\r\n"
                request += f"Content-Length: {content_length}\r\n"
                request += "Content-Type:
application/x-www-form-urlencoded\r\n\r\n"
                request += post_body
            else:
                request = "GET " + "http://" + filename + "
HTTP/1.0\r\nHost: " + hostn + "\r\n\r\n"

            fileobj.write(request.encode())

            # Read the response into buffer
            # Fill in start.
            buffer = fileobj.readlines()
            # Fill in end.

            # Create a new file in the cache for the requested file.
            # Also send the response in the buffer to client socket
يand the corresponding file in the cache
```

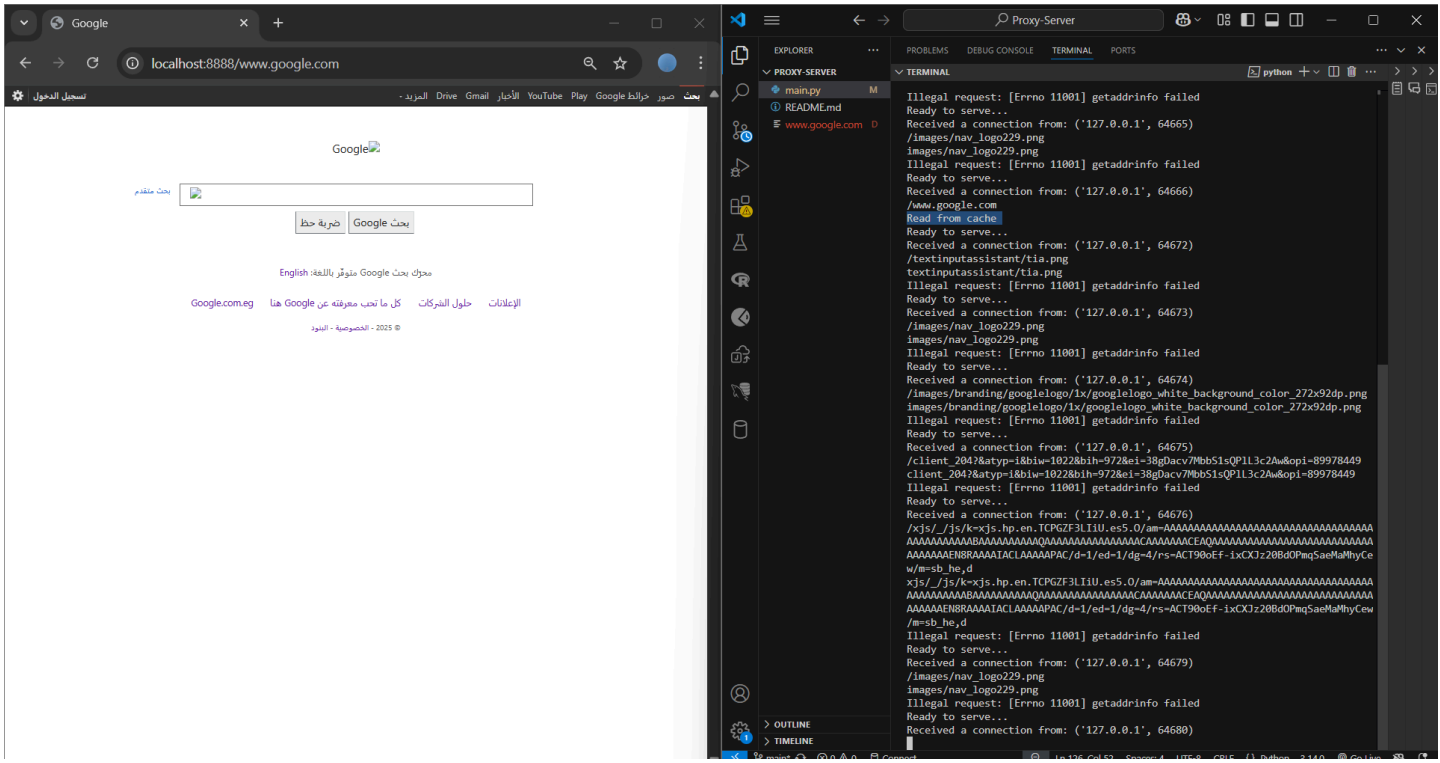
```
            # Bonus we only cache GET requests
            if method == "GET":
                tmpFile = open("./" + filename, "wb")

                # Fill in start.
                for line in buffer:
                    tmpFile.write(line)
                    tcpCliSock.send(line)
                tmpFile.close()
                # Fill in end.
            else:
                # else we just send POST response without caching
                for line in buffer:
                    tcpCliSock.send(line)

        except Exception as e:
            print("Illegal request:", str(e))
            # HTTP response message for error
            # Fill in start.
            tcpCliSock.send("HTTP/1.0 404 Not Found\r\n".encode())

tcpCliSock.send("Content-Type:text/html\r\n\r\n".encode())
            # Fill in end.

    # Close the client and the server sockets
    tcpCliSock.close()

# Fill in start.
# Fill in end.
```

The screenshots:

First time accessing a site (google), resulting in a miss



Accessing the site again, resulting in a hit as shown



Note we found:

There are no images displayed (relevant to “Illegal Request” in the terminal), this is because most images are sent over HTTPS and a different port number than the one needed for HTML/HTTP.