**Lab 1 :**

I have a server.py implemented from the given skeleton code. I also have a HelloWorld.html containing the text *‘Hello World’.*

>server.py

#import socket module

from socket import \*

import sys # In order to terminate the program

#Prepare a sever socket

serverSocket = socket(AF\_INET, SOCK\_STREAM)

serverPort = 6789

serverSocket.bind(('', serverPort))

serverSocket.listen(1)

while True:

print('Ready to serve...')

connectionSocket, addr = serverSocket.accept() #Establish the connection

try:

message = connectionSocket.recv(1024)

filename = message.split()[1]

f = open(filename[1:], "rb")

outputdata = f.read()

#Send one HTTP header line into socket

header = '\nHTTP/1.1 200 OK\n\n'

connectionSocket.send(header.encode())

for i in range(2, len(outputdata)):

connectionSocket.send(outputdata[i:i+1])

connectionSocket.send("\r\n".encode())

connectionSocket.close()

except IOError:

connectionSocket.send('HTTP/1.1 404 Not Found\r\n\r\n'.encode())

errorMessage = '<html><head></head><body><h1>404 Not Found</h1></body></html>\r\n'

connectionSocket.send(errorMessage.encode())

connectionSocket.send(b'\r\n')

connectionSocket.close()

serverSocket.close()

sys.exit()#Terminate the program after sending the corresponding data

>HelloWorld.html

⁠<html>

<head>

<title>Hello World</title>

</head>

<body >

Hello World

<iframe name="hiddenIFrame" style="display:none;"></iframe>

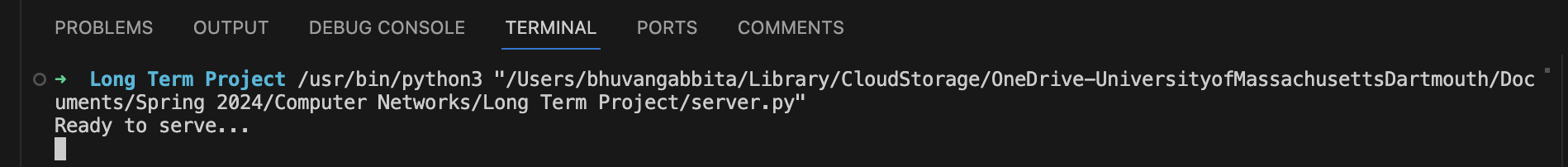
</body>

</html>

(Ignore the iframe)  
  
**Scenario 1:**

**Requesting HelloWorld.html.**

* Run server.py



* Request for HelloWorld.html in the URL bar.
* A screenshot of a computer

  Description automatically generated
* Since we are running the server on our system(Mac), we request using localhost from client browser.
* Now I tried requesting from a Windows laptop which has a completely different IP address and was able to access the file contents.

A computer screen shot of a black screen

Description automatically generated

**Scenario 2:**

**Requesting cutecat.html (File doesn’t exist at the server side)**

A screen shot of a computer

Description automatically generated

**Optional Exercise:**

1. I have written a server script that runs and creates a thread for every client connection .

>multiserverthread.py

from socket import \*

import sys, threading # In order to terminate the program

class ConsumerThread(threading.Thread):

def \_\_init\_\_(self, addr,connectionSocket):

threading.Thread.\_\_init\_\_(self)

self.address = addr

self.csocket = connectionSocket

def run(self):

try:

message = connectionSocket.recv(1024)

filename = message.split()[1]

f = open(filename[1:], "rb")

outputdata = f.read()

#Send one HTTP header line into socket

header = '\nHTTP/1.1 200 OK\n\n'

connectionSocket.send(header.encode())

#Send the content of the requested file to the client

for i in range(2, len(outputdata)):

connectionSocket.send(outputdata[i:i+1])

connectionSocket.send(b'\r\n\r\n')

connectionSocket.close()

except IOError:

#Send response message for file not found

connectionSocket.send('HTTP/1.1 404 Not Found\r\n\r\n'.encode())

errorMessage = '<html><head></head><body><h1>404 Not Found</h1></body></html>\r\n'

connectionSocket.send(errorMessage.encode())

connectionSocket.send(b'\r\n\r\n')

#Close client socket

connectionSocket.close()

serverSocket = socket(AF\_INET, SOCK\_STREAM)

serverPort = 6789

#Prepare a sever socket

serverSocket.bind(('', serverPort))

serverSocket.listen(5)

while True:

#Establish the connection

print('Ready to serve...')

connectionSocket, addr = serverSocket.accept()

#pass clientsock to the ConsumerThread thread object being created

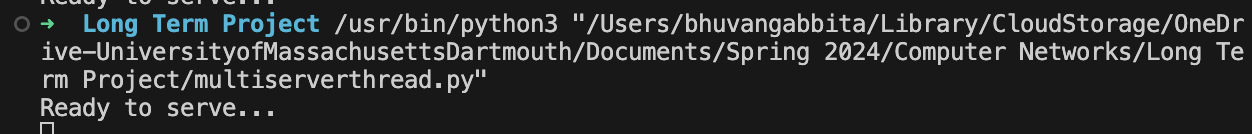
newthread = ConsumerThread(addr , connectionSocket)

newthread.start()

serverSocket.close()

sys.exit()#Terminate the program after sending the corresponding data

Opening multiple clients for the same server.



On windows laptop:

A screenshot of a computer

Description automatically generated

On Phone:

A black rectangle with white dots

Description automatically generated

Server after multiple client requests.  
A screen shot of a computer

Description automatically generated

1. Wrote a client.py script that performs a GET request for the file and retrieves the contents.

>client.py

from socket import \*

import sys

serverName = sys.argv[1]

serverPort = int(sys.argv[2])

file\_name = sys.argv[3]

clientSocket = socket(AF\_INET, SOCK\_STREAM)

clientSocket.connect((serverName, serverPort))

message = 'GET /' + file\_name

clientSocket.send(message.encode())

# header = repr(clientSocket.recv(1024))#.decode('utf-8')

# messageReceived = repr(clientSocket.recv(1024))#.decode('utf-8')

# finalMessage = ''

# while messageReceived:

# finalMessage += messageReceived

# messageReceived = repr(clientSocket.recv(1024)) #.decode('utf-8')

response = b" "

while True:

part = clientSocket.recv(1024)

if not part:

break

response += part

print(response)

clientSocket.close()

Upon running the client.py with server, port and filename as commandline arguments.

