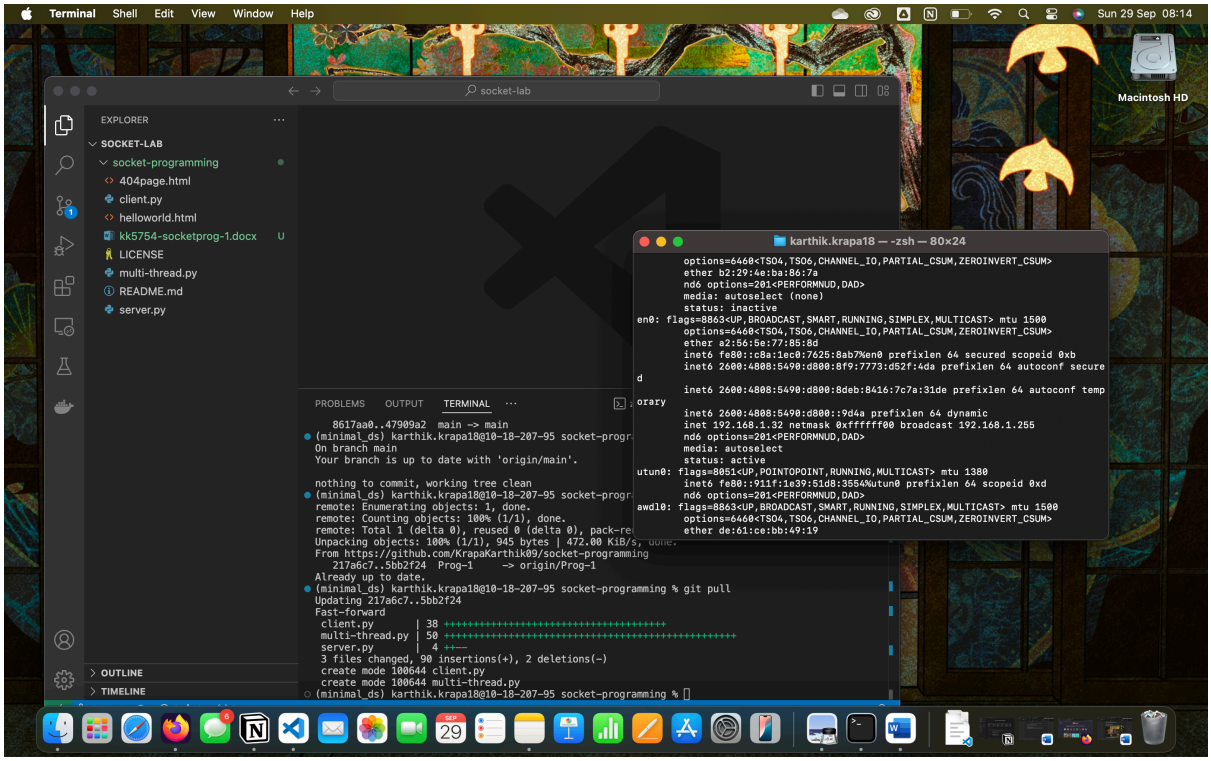
Name: Krapa Karthik **NYU ID:** N12039854 **NET ID:** kk5754

Computer Networking - Socket Programming Lab - 1

Identifying Lab Environment

The below image depicts my lab environment on my laptop, it uses MacOS.



Server.py File

Code with comment lines for explanation

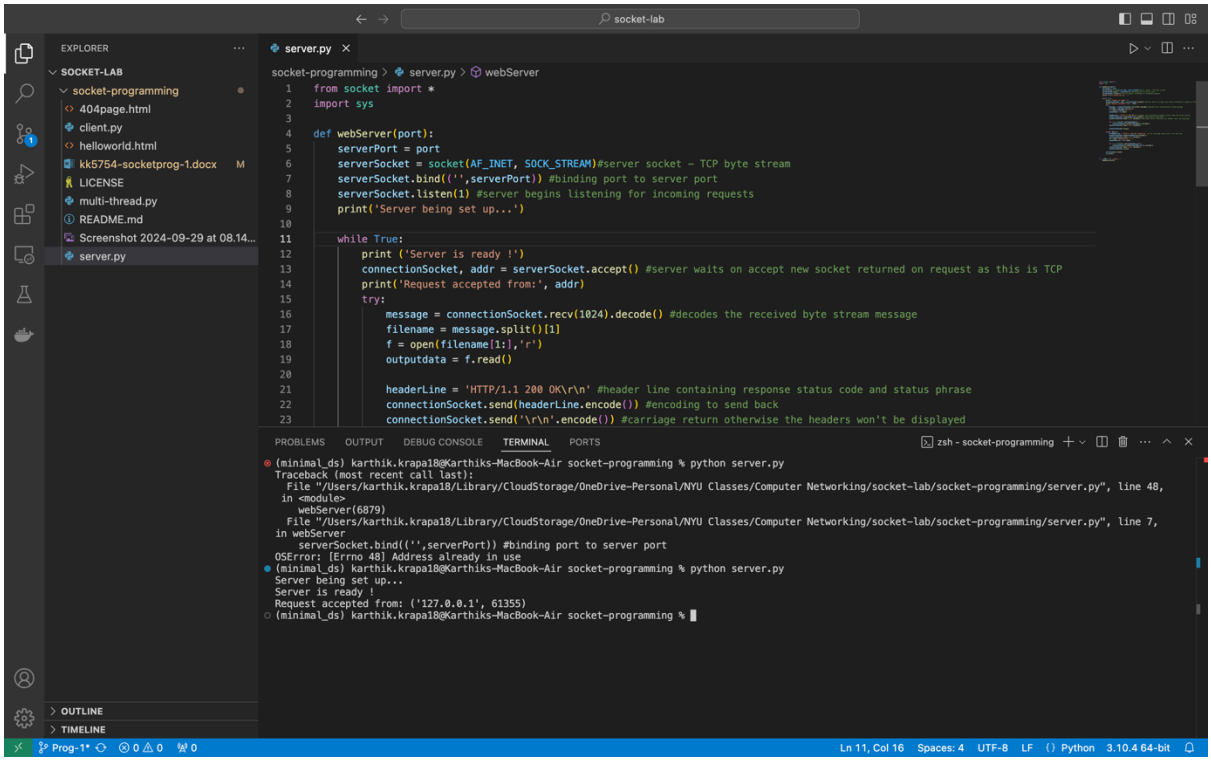
```
from socket import *
import sys

def webServer(port):
    serverPort = port
    serverSocket = socket(AF_INET, SOCK_STREAM)#server socket - TCP byte stream
    serverSocket.bind(('',serverPort)) #binding port to server port
    serverSocket.listen(1) #server begins listening for incoming requests
    print('Server being set up...')

    while True:
        print ('Server is ready !')
        connectionSocket, addr = serverSocket.accept() #server waits on accept new
socket returned on request as this is TCP
        print('Request accepted from:', addr)
        try:
```

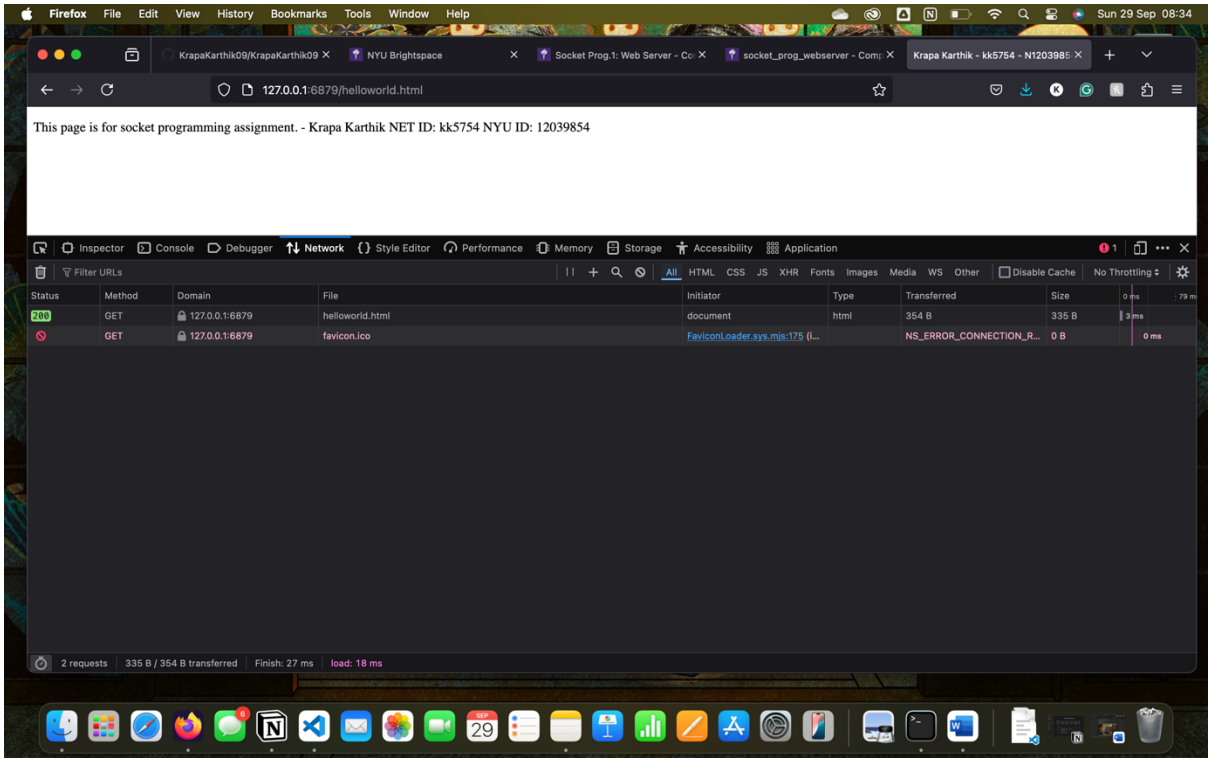
```
            message = connectionSocket.recv(1024).decode() #decodes the received
byte stream message
            filename = message.split()[1]
            f = open(filename[1:],'r')
            outputdata = f.read()

            headerLine = 'HTTP/1.1 200 OK\r\n' #header line containing response
status code and status phrase
            connectionSocket.send(headerLine.encode()) #encoding to send back
            connectionSocket.send('\r\n'.encode()) #carriage return otherwise the
headers won't be displayed

            for i in range(0, len(outputdata)):
                connectionSocket.send(outputdata[i].encode())
            connectionSocket.send('\r\n'.encode())

            connectionSocket.close()

        except IOError:
            errHeaderLine = 'HTTP/1.1 404 Not Found\r\n'  #r for carriage return
and n for new line
            connectionSocket.send(errHeaderLine.encode())
            connectionSocket.send('\r\n'.encode())
            f2 = open('404page.html','r')
            outputdata_err = f2.read()

            for i in range(0, len(outputdata_err)):
                connectionSocket.send(outputdata_err[i].encode())
            connectionSocket.send('\r\n'.encode())
            connectionSocket.close()

        serverSocket.close()
        sys.exit()


if __name__ == '__main__':
    webServer(6879)
```

Output:

helloworld.html 200

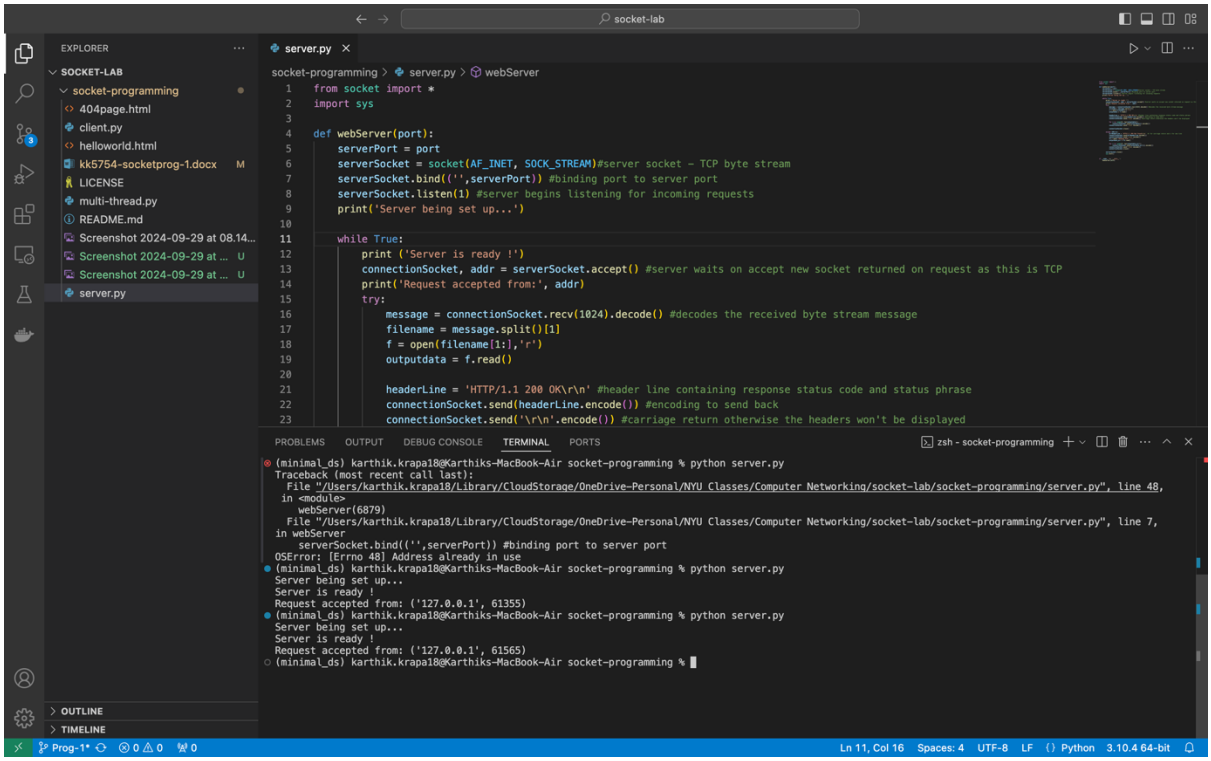
Name: Krapa Karthik **NYU ID:** N12039854 **NET ID:** kk5754

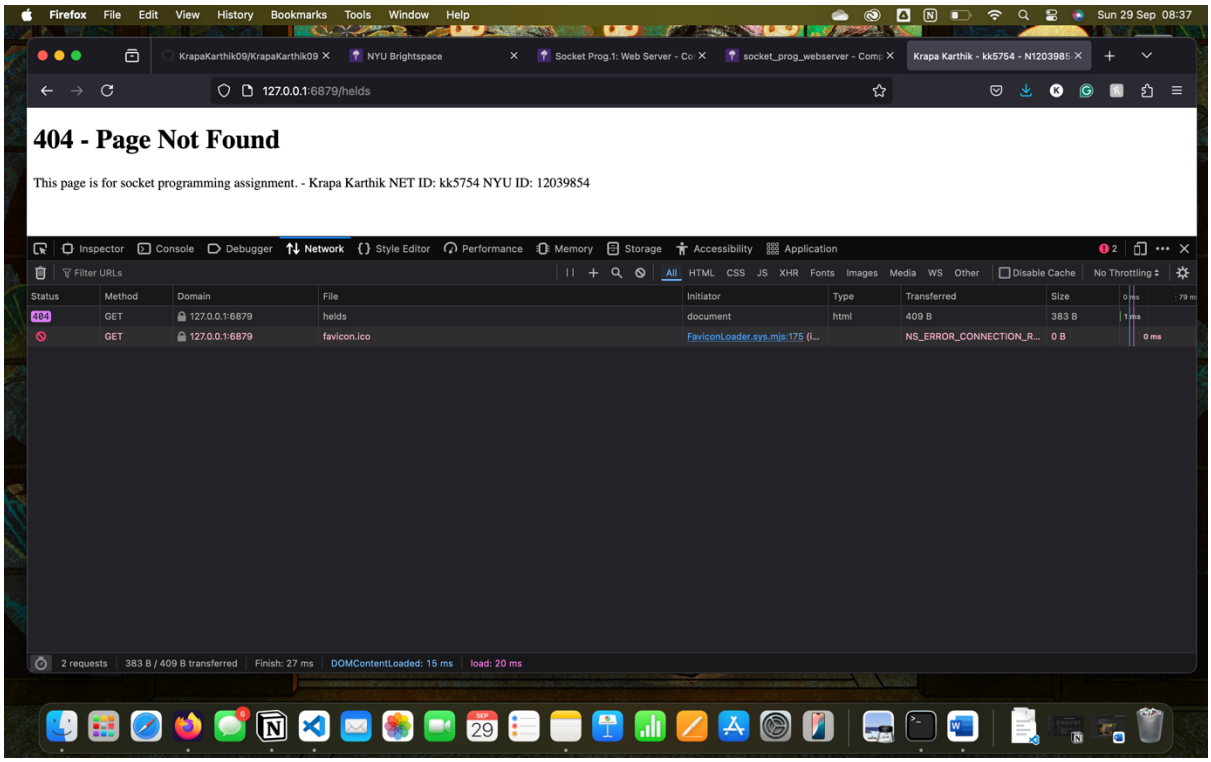




404page.html 404

Name: Krapa Karthik **NYU ID:** N12039854 **NET ID:** kk5754





Optional Exercises

1. Multi-thread.py

```
1.  from socket import *
2.  import sys
3.  import threading
4.
```

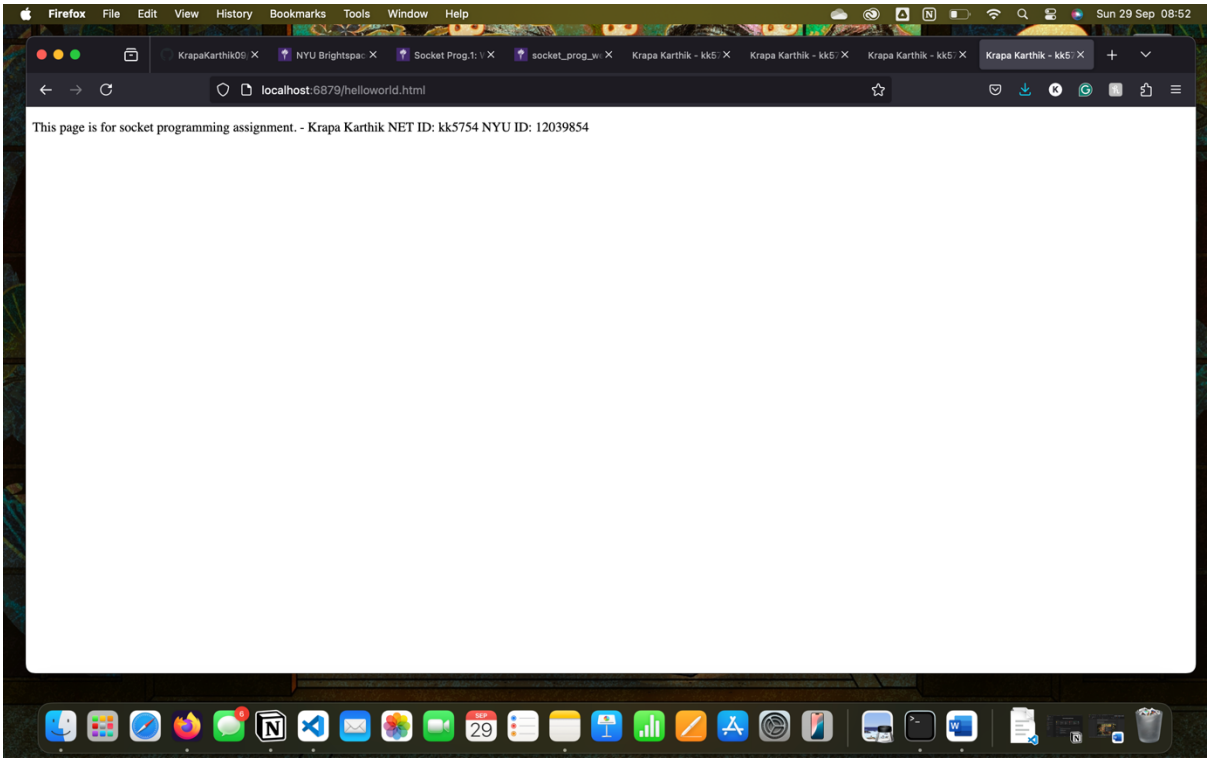
Name: Krapa Karthik **NYU ID:** N12039854 **NET ID:** kk5754

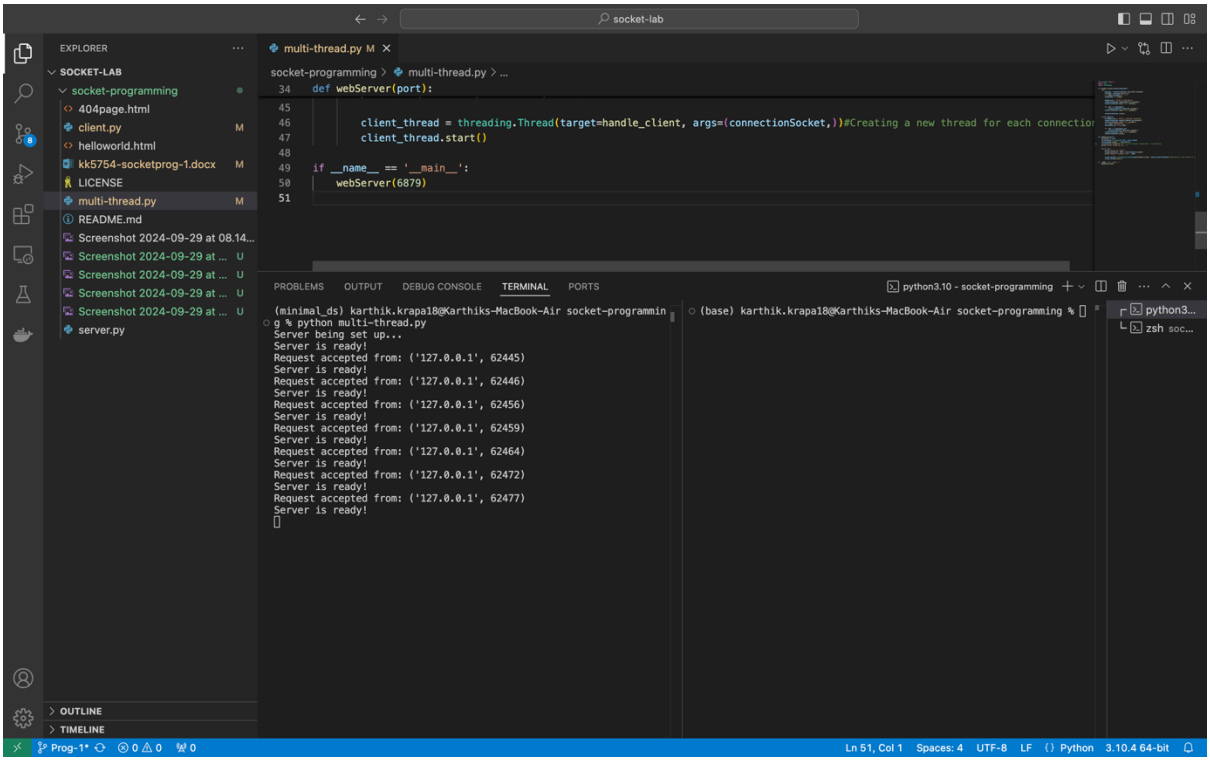
```
def handle_client(connectionSocket):
    try:
        message = connectionSocket.recv(1024).decode()
        filename = message.split()[1]
        f = open(filename[1:], 'r')
        outputdata = f.read()

        headerLine = 'HTTP/1.1 200 OK\r\n'
        connectionSocket.send(headerLine.encode())
        connectionSocket.send('\r\n'.encode())

        for char in outputdata:
            connectionSocket.send(char.encode())
        connectionSocket.send('\r\n'.encode())

        connectionSocket.close()

    except IOError:
        errHeaderLine = 'HTTP/1.1 404 Not Found\r\n'
        connectionSocket.send(errHeaderLine.encode())
        connectionSocket.send('\r\n'.encode())
        f2 = open('404page.html', 'r')
        outputdata_err = f2.read()

        for char in outputdata_err:
            connectionSocket.send(char.encode())
        connectionSocket.send('\r\n'.encode())
        connectionSocket.close()

def webServer(port):
    serverPort = port
    serverSocket = socket(AF_INET, SOCK_STREAM)
    serverSocket.bind(('', serverPort))
    serverSocket.listen(4) #this is for multiple connections - 4 connections
    print('Server being set up...')

    while True:
        print('Server is ready!')
        connectionSocket, addr = serverSocket.accept()
        print('Request accepted from:', addr)

        client_thread = threading.Thread(target=handle_client,
args=(connectionSocket,))#Creating a new thread for each connection
        client_thread.start()

if __name__ == '__main__':
    webServer(6879)

```

Output:

Name: Krapa Karthik **NYU ID:** N12039854 **NET ID:** kk5754

The below image indicates multi tabs open for indicating sessions with the server.





2. Client.py

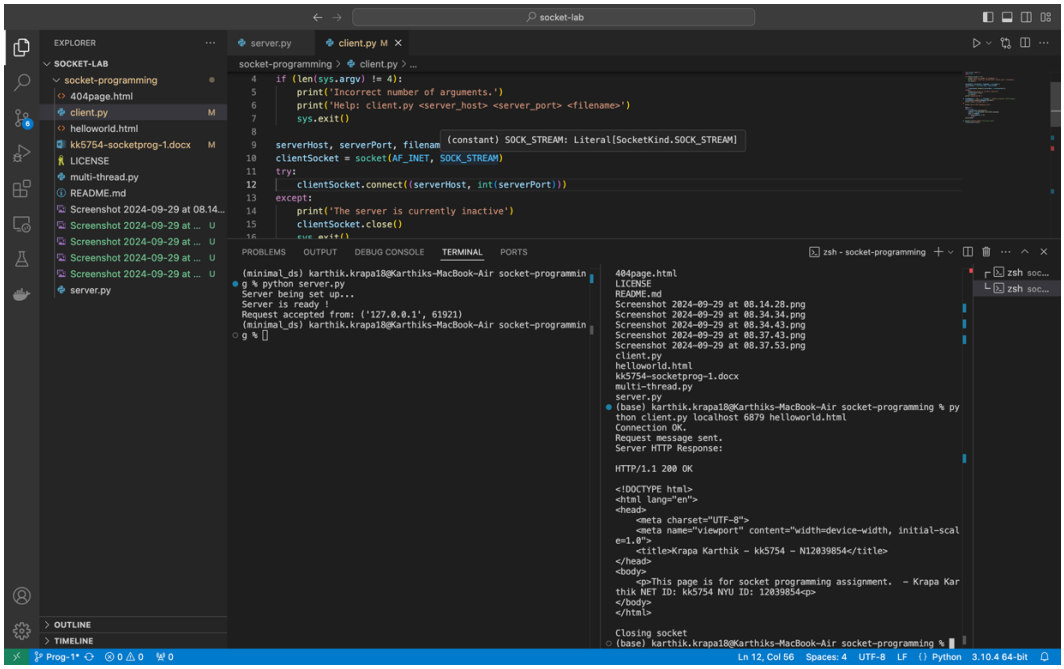
```
52. from socket import *
53. import sys
54.
```

Name: Krapa Karthik **NYU ID:** N12039854 **NET ID:** kk5754

```
55. if (len(sys.argv) != 4): #For arguments to the script so that client can
    connect to server
56.     print('Incorrect number of arguments.')
57.     print('Help: To run python client.py <server_host> <server_port>
    <filename>')
58.     sys.exit()
59.
60. serverHost, serverPort, filename = sys.argv[1:] #server arguments
61. clientSocket = socket(AF_INET, SOCK_STREAM)
62. try:
63.     clientSocket.connect((serverHost, int(serverPort))) #client connecting
    to server
64. except:
65.     print('The server is currently inactive')
66.     clientSocket.close() #close connection if inactive
67.     sys.exit()
68. print('Connection OK.')
69.
70. httpRequest = 'GET /' + filename + ' HTTP/1.1\r\n\r\n' #HTTP Request
71. clientSocket.send(httpRequest.encode())
72. print('Request message sent.')
73.
74. print('Server HTTP Response:\r\n')
75.
76. data = ""
77. while True:
78.     clientSocket.settimeout(5)
79.     newData = clientSocket.recv(1024).decode()
80.     data += newData
81.     if (len(newData) == 0):
82.         break
83. print(data)
84.
85. print('Closing socket') #Closing socket
86. clientSocket.close()
```

Name: Krapa Karthik **NYU ID:** N12039854 **NET ID:** kk5754

Output:



Please refer to the code from my Github Repo Link :
https://github.com/KrapaKarthik09/socket-programming/tree/Prog-1