COMPUTER NETWORKS LABORATORY

WEEK #5

NAME : SIRI S

SEMESTER : 4

SECTION :H

SRN : PES1UG19CS485

Objective:

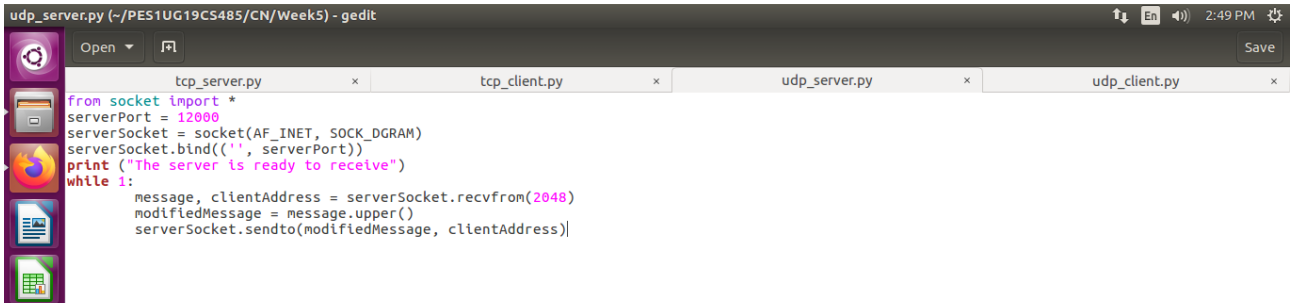
To develop a simple Client-Server application using TCP and UDP.

Task 1:

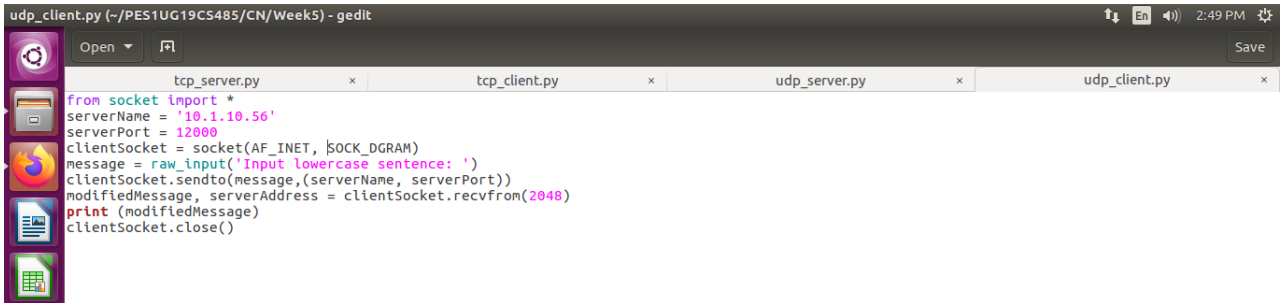
1. Create an application that will
a. Convert lowercase letters to uppercase
• e.g. [a...z] to [A...Z]
• code will not change any special characters, e.g. &*!
b. If the character is in uppercase, the program must not alter
2. Create Socket API both for client and server.
3. Must take the server address and port from the Command Line Interface (CLI).

1. UDP CONNECTION:

UDP Server:

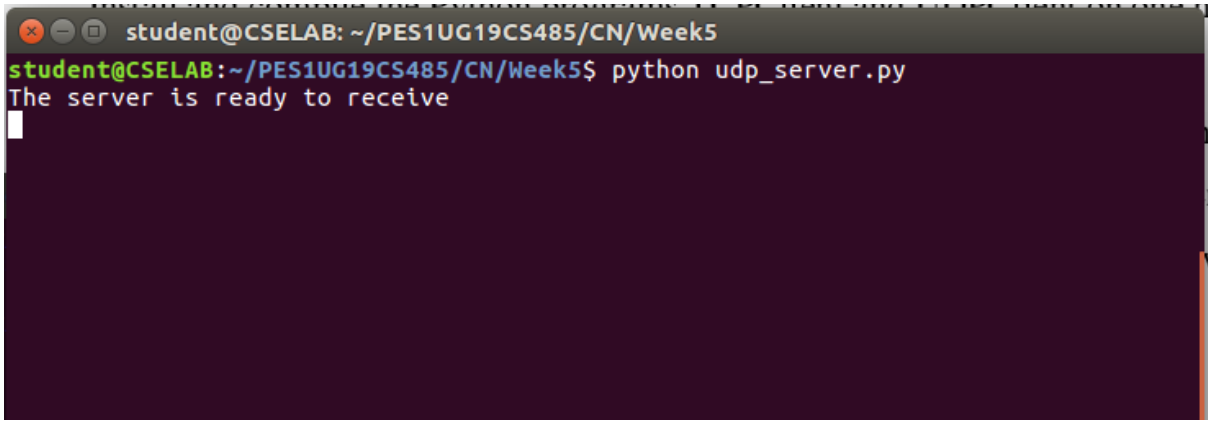


UDP Client:

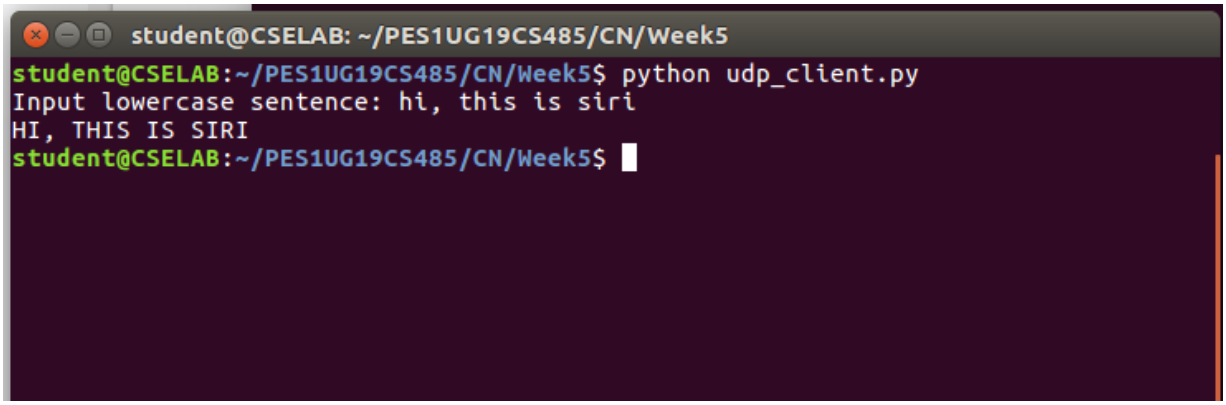


UDP CONNECTION BETWEEN SERVER AND CLIENT:

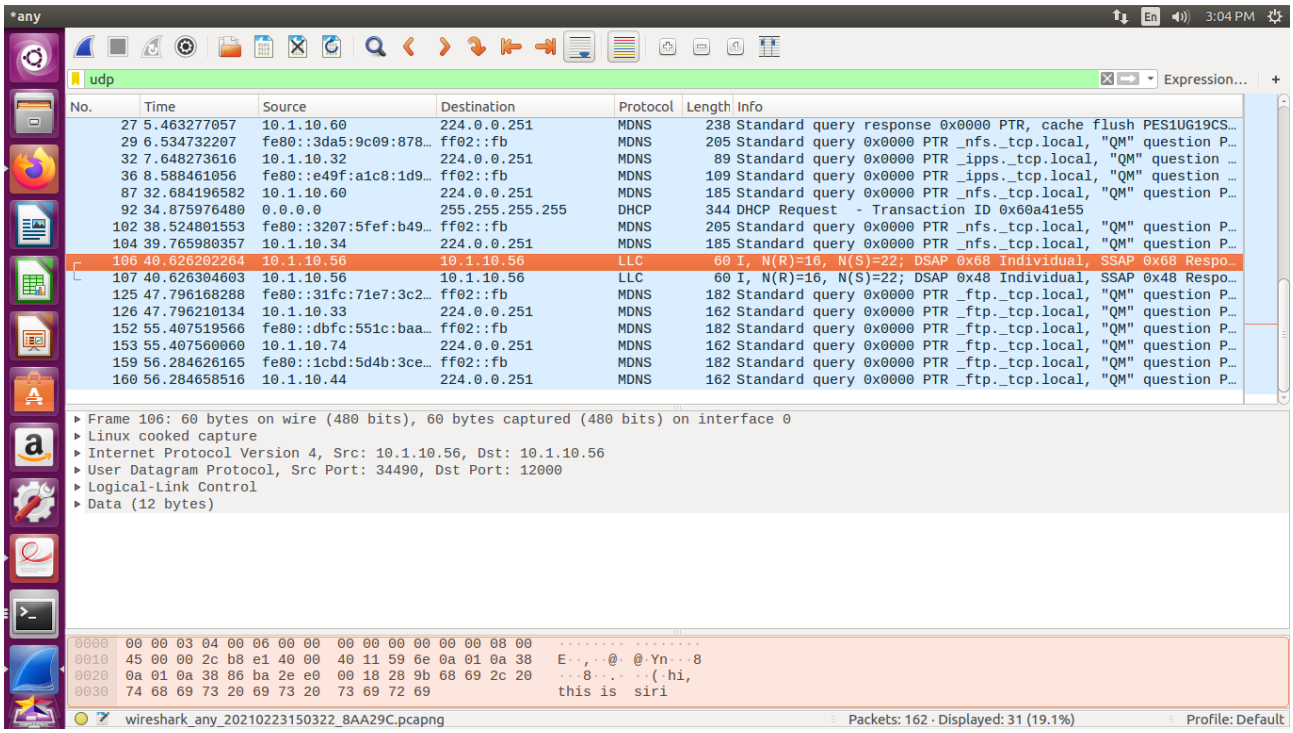
UDP Server:



UDP Client:

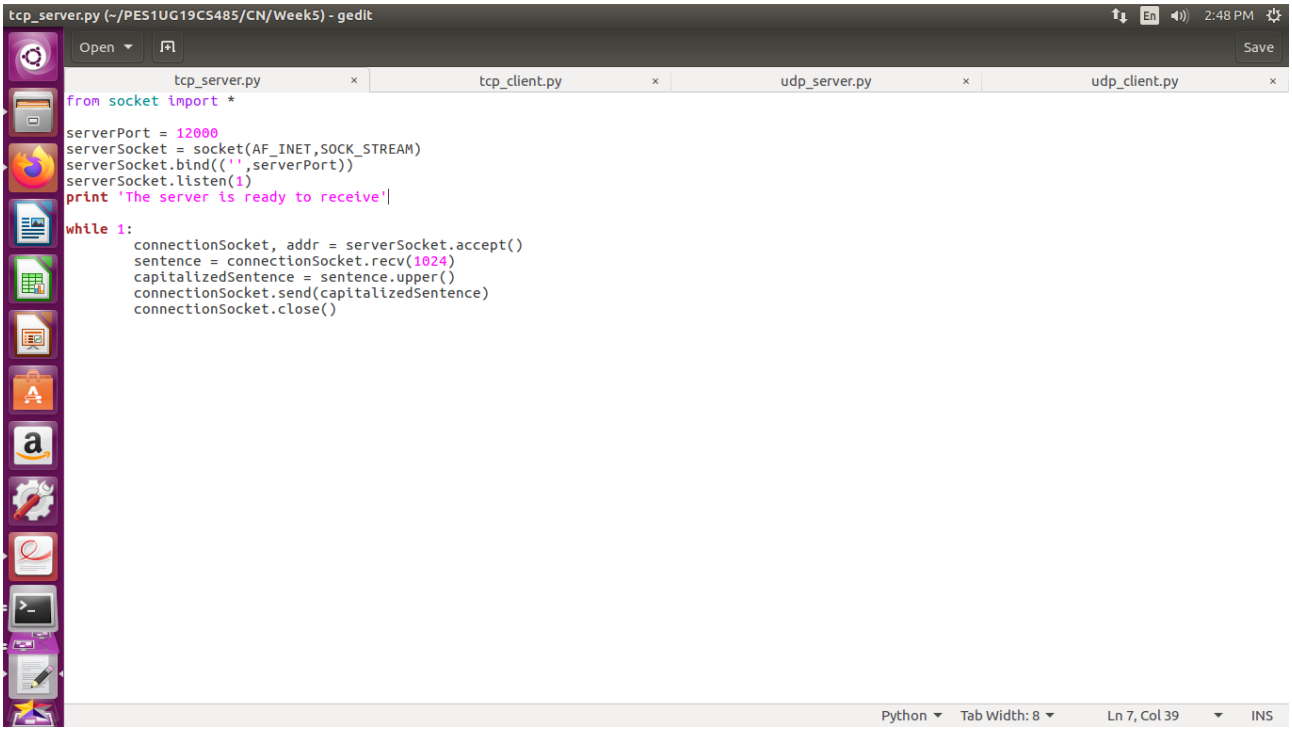


WIRESHARK CAPTURE FOR UDP CONNECTION:

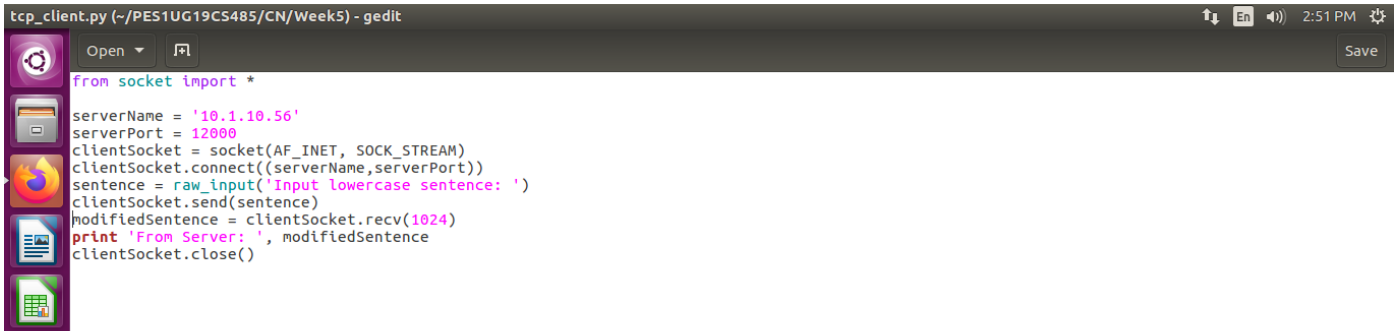


2. TCP CONNECTION:

TCP Server:

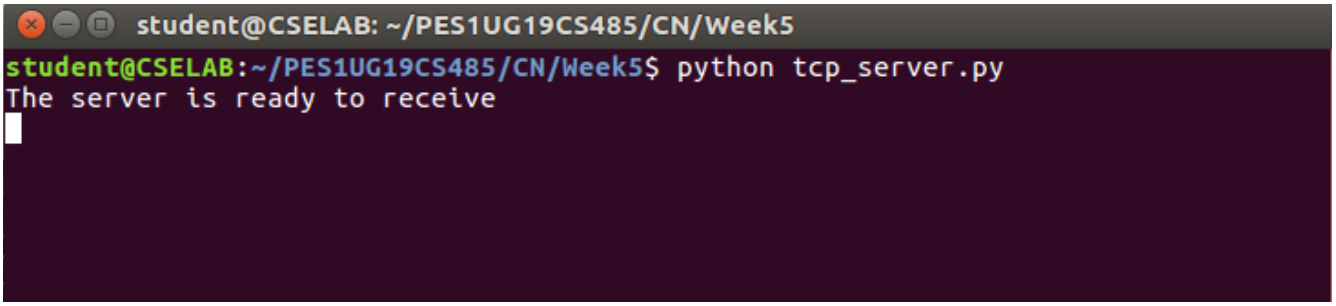


TCP Client:

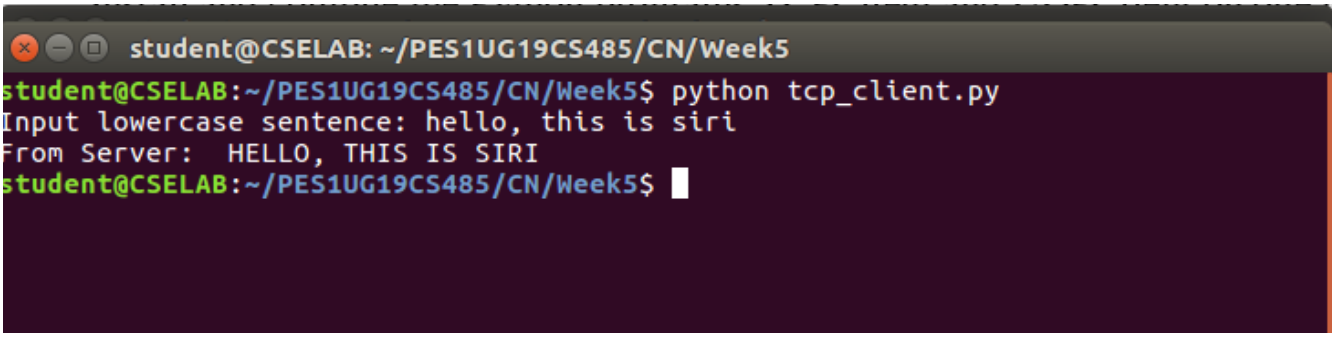


TCP CONNECTION BETWEEN SERVER AND CLIENT:

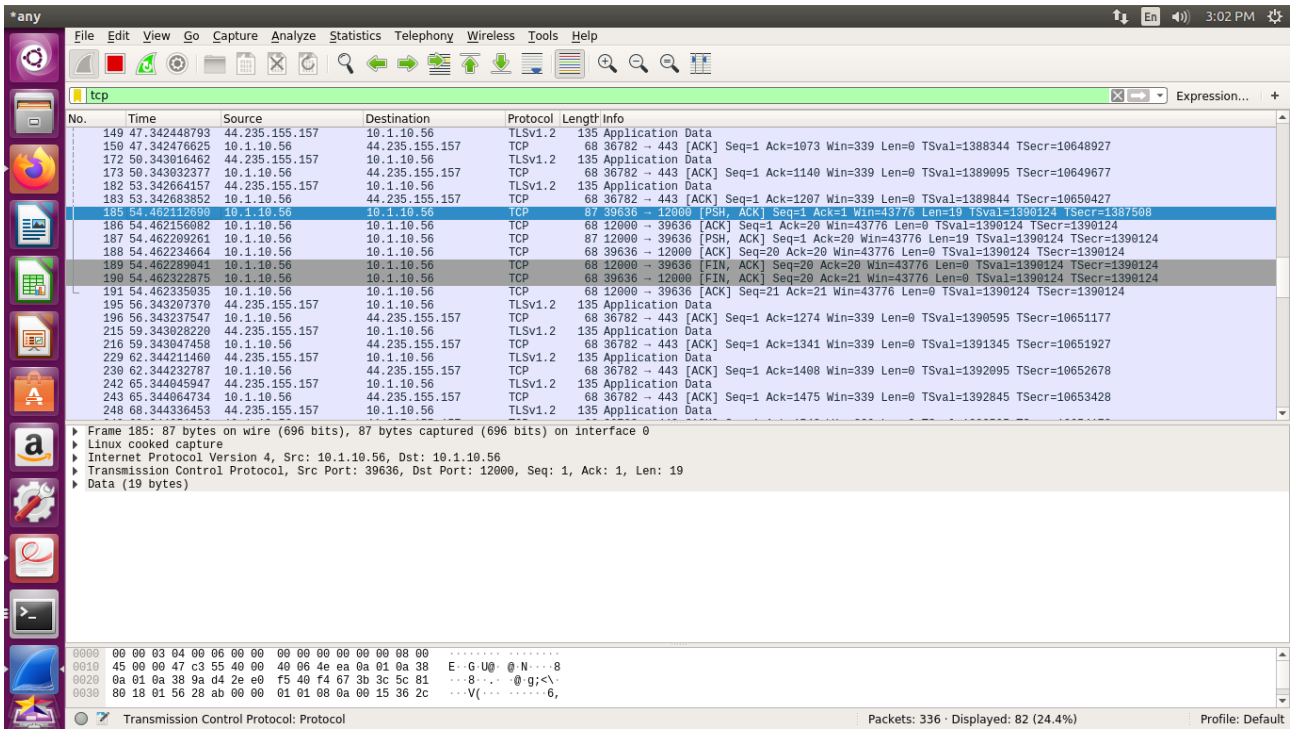
TCP Server:



TCP Client:

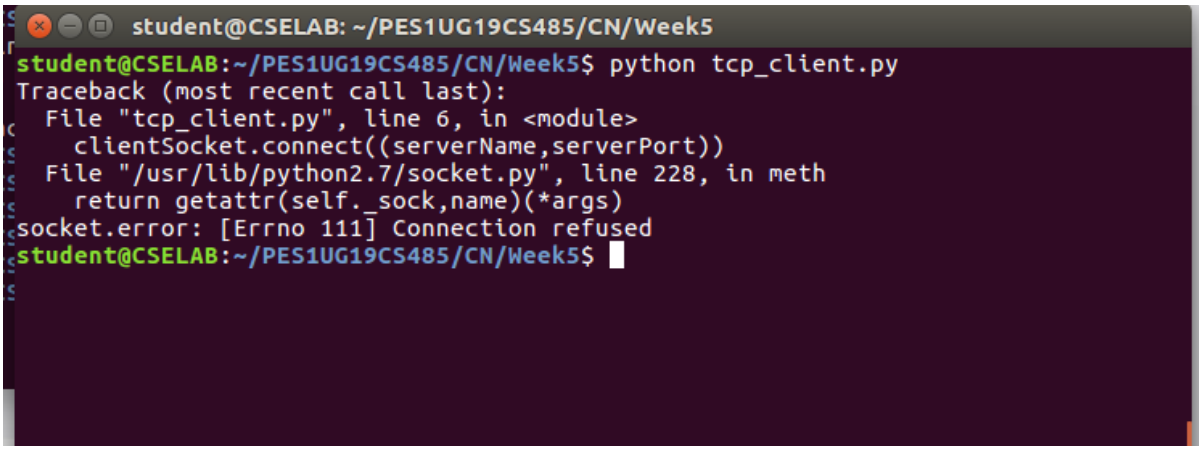


WIRESHARK CAPTURE FOR TCP CONNECTION:



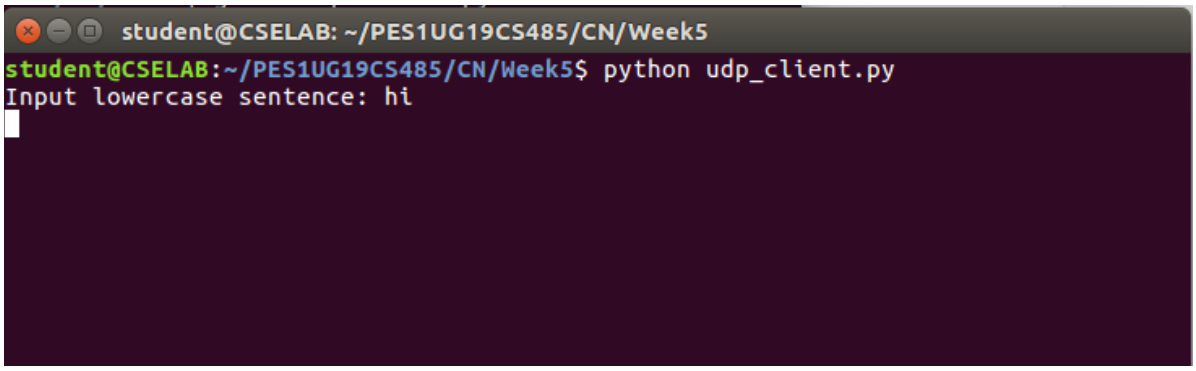
3. PROBLEMS:

1. Suppose you run TCPClient before you run TCPServer. What happens? Why?



This is because a TCP Connection between two socket interfaces only when a host machine listens to requests on a given IP address and port number and accepts connections made by another machine at the same address and port.

2. **Suppose you run UDPClient before you run UDPServer. What happens? Why?**



No error occurs because UDP does not require a prior connection to be established between the host machines for data transfer to begin.

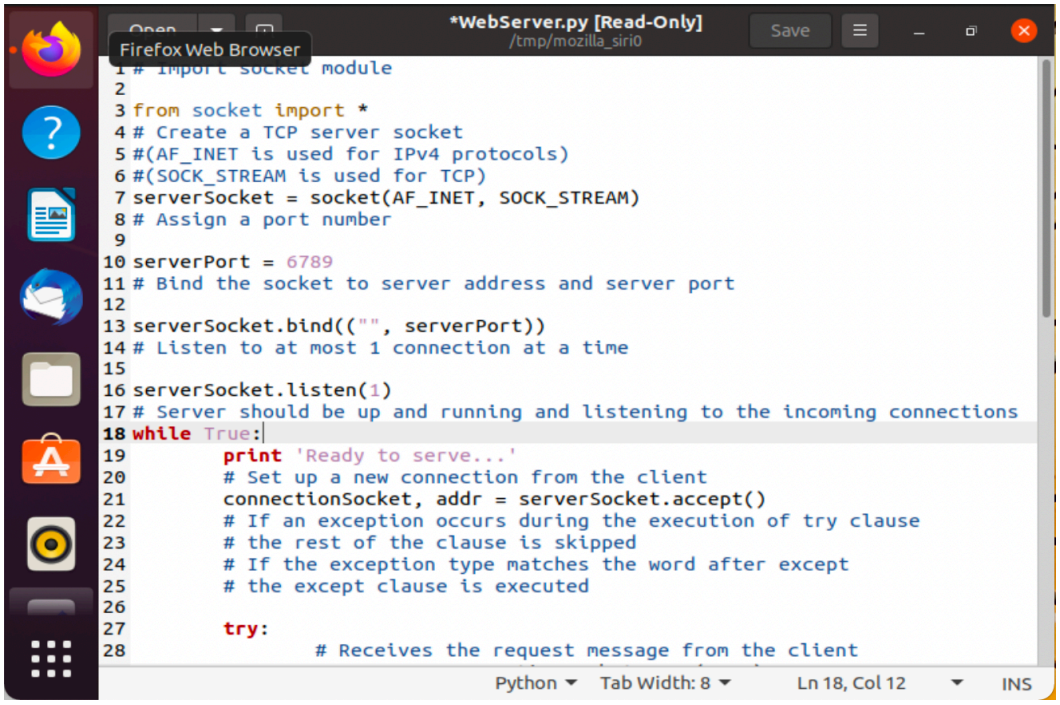
3. **What happens if you use different port numbers for the client and server sides?**

In case of TCP Connection, it leads to a ConnectionRefusedError because the server socket application is not listening to requests at the same port number as the client socket is trying to connect with.

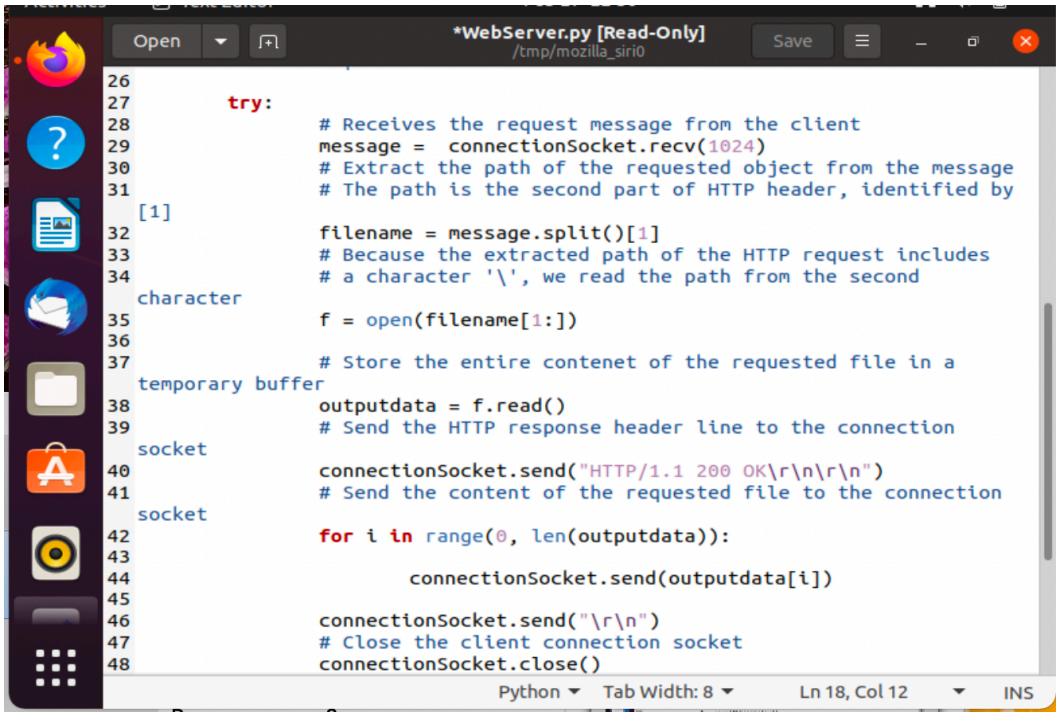
In case of UDP Connection, no error is obtained as no prior connection is required to b established .

Task 2: WEB SERVER

WebServer.py



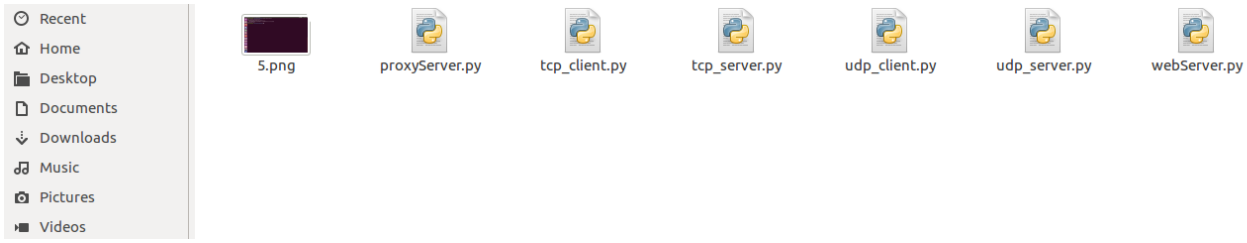

```
1 # Import socket module
2
3 from socket import *
4 # Create a TCP server socket
5 #(AF_INET is used for IPv4 protocols)
6 #(SOCK_STREAM is used for TCP)
7 serverSocket = socket(AF_INET, SOCK_STREAM)
8 # Assign a port number
9
10 serverPort = 6789
11 # Bind the socket to server address and server port
12
13 serverSocket.bind(("", serverPort))
14 # Listen to at most 1 connection at a time
15
16 serverSocket.listen(1)
17 # Server should be up and running and listening to the incoming connections
18 while True:
19         print 'Ready to serve...'
20         # Set up a new connection from the client
21         connectionSocket, addr = serverSocket.accept()
22         # If an exception occurs during the execution of try clause
23         # the rest of the clause is skipped
24         # If the exception type matches the word after except
25         # the except clause is executed
26
27         try:
28                 # Receives the request message from the client
```




```
26
27         try:
28                 # Receives the request message from the client
29                 message =  connectionSocket.recv(1024)
30                 # Extract the path of the requested object from the message
31                 # The path is the second part of HTTP header, identified by
   [1]
32                 filename = message.split()[1]
33                 # Because the extracted path of the HTTP request includes
34                 # a character '\', we read the path from the second
   character
35                 f = open(filename[1:])
36
37                 # Store the entire contenet of the requested file in a
   temporary buffer
38                 outputdata = f.read()
39                 # Send the HTTP response header line to the connection
   socket
40                 connectionSocket.send("HTTP/1.1 200 OK\r\n\r\n")
41                 # Send the content of the requested file to the connection
   socket
42                 for i in range(0, len(outputdata)):
43
44                         connectionSocket.send(outputdata[i])
45
46                 connectionSocket.send("\r\n")
47                 # Close the client connection socket
48                 connectionSocket.close()
```

```
49         except:
50
51                 # Send HTTP response message for file not found
52                 connectionSocket.send("HTTP/1.1 404 Not Found\r\n\r\n")
53                 connectionSocket.send("<html><head></head><body><h1>404 Not
   Found</h1></body></html>\r\n")
54                 # Close the client connection socket
55                 connectionSocket.close()
56 serverSocket.close()
57
58
59
```

Saving an image in the same directory as WebServer.py



On accessing localhost with the port number specified in the code from Firefox: