CN Lab Report – Week 5

NAME	SRN	SECTION
ANKITH J RAI	PES1UG19CS069	B

TASK 1

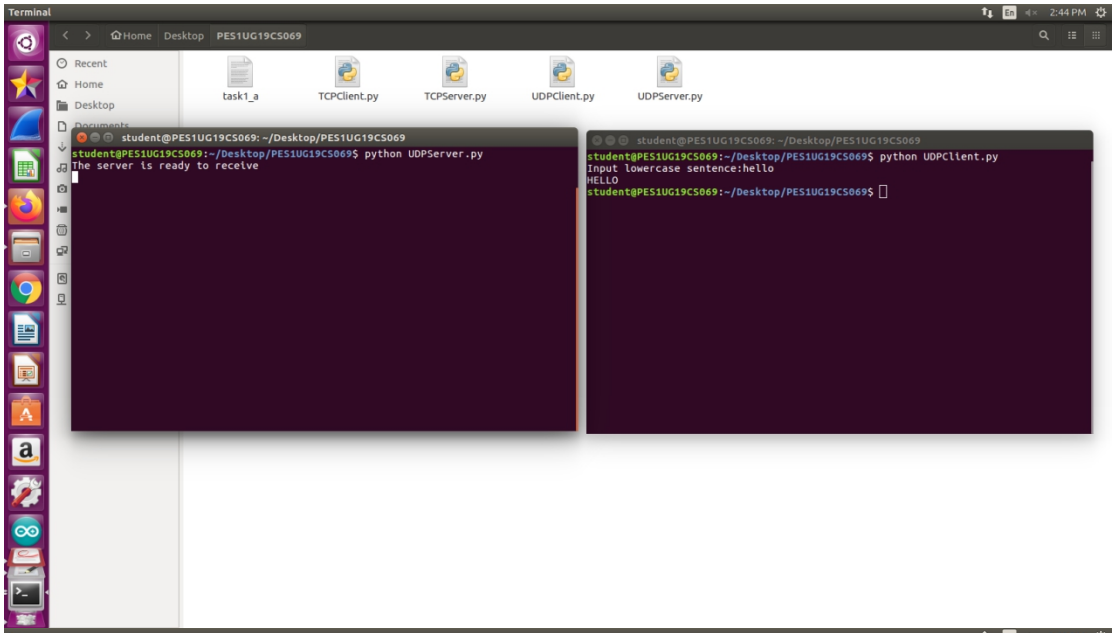
Client program for datagram service (UDP)

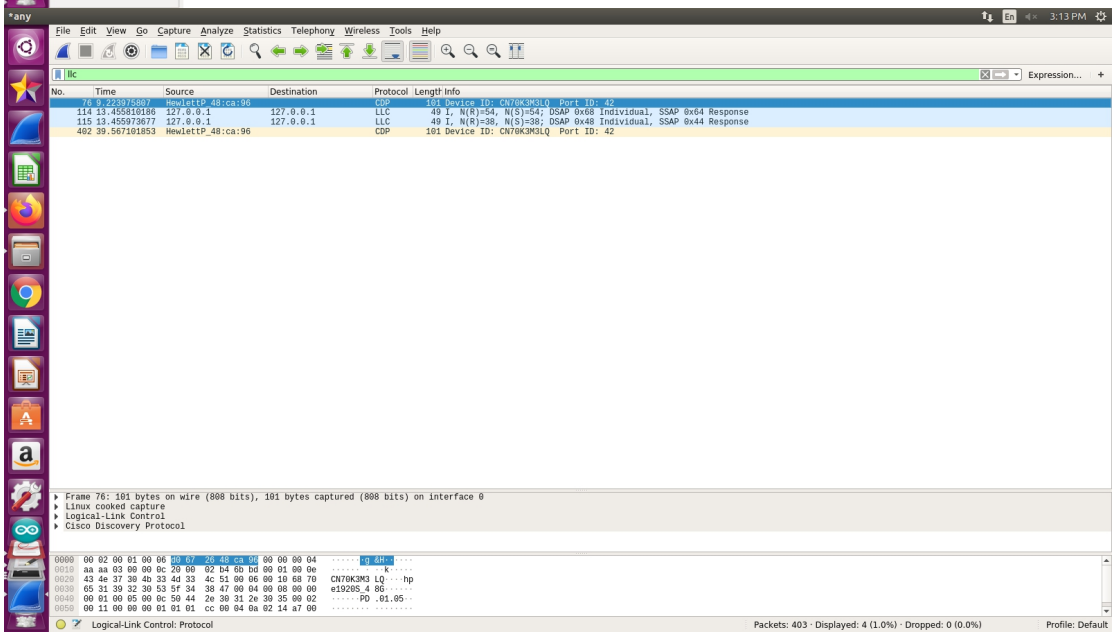
```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
clientSocket = socket(AF_INET,SOCK_DGRAM)
message = raw_input('Input lowercase sentence:')
clientSocket.sendto(message,(serverName, serverPort))
modifiedMessage, serverAddress = clientSocket.recvfrom(2048)
Print(modifiedMessage)
clientSocket.close()
```

Server program for datagram service (UDP)

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', serverPort))
print("The server is ready to receive")
while 1:
        message, clientAddress = serverSocket.recvfrom(2048)
        modifiedMessage = message.upper()
        serverSocket.sendto(modifiedMessage, clientAddress)
```

UDP - wireshark and terminal screenshots:





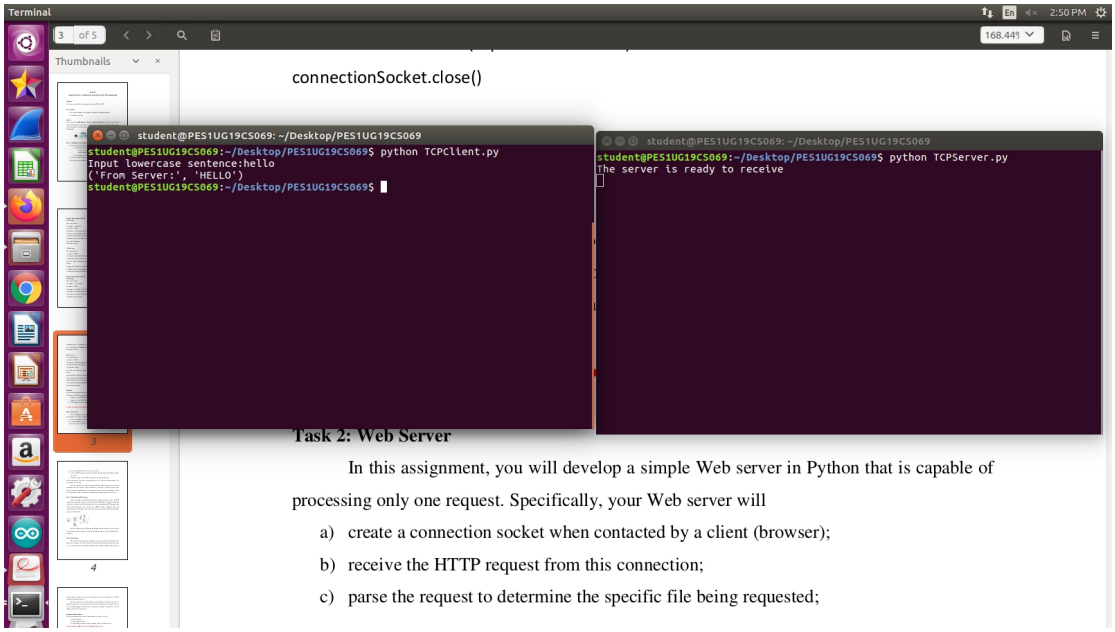
Client program for reliable byte-stream service (TCP)

```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName,serverPort))
sentence = raw_input('Input lowercase sentence:')
clientSocket.send(sentence)
modifiedSentence = clientSocket.recv(1024)
Print('From Server:', modifiedSentence)
clientSocket.close()
```

Server program for reliable byte-stream service (TCP)

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind((‘’,serverPort))
serverSocket.listen(1)
print('The server is ready to receive')
while 1:
        connectionSocket, addr = serverSocket.accept()
        sentence = connectionSocket.recv(1024)
        capitalizedSentence = sentence.upper()
        connectionSocket.send(capitalizedSentence)
        connectionSocket.close()
```

TCP - wireshark and terminal screenshots:



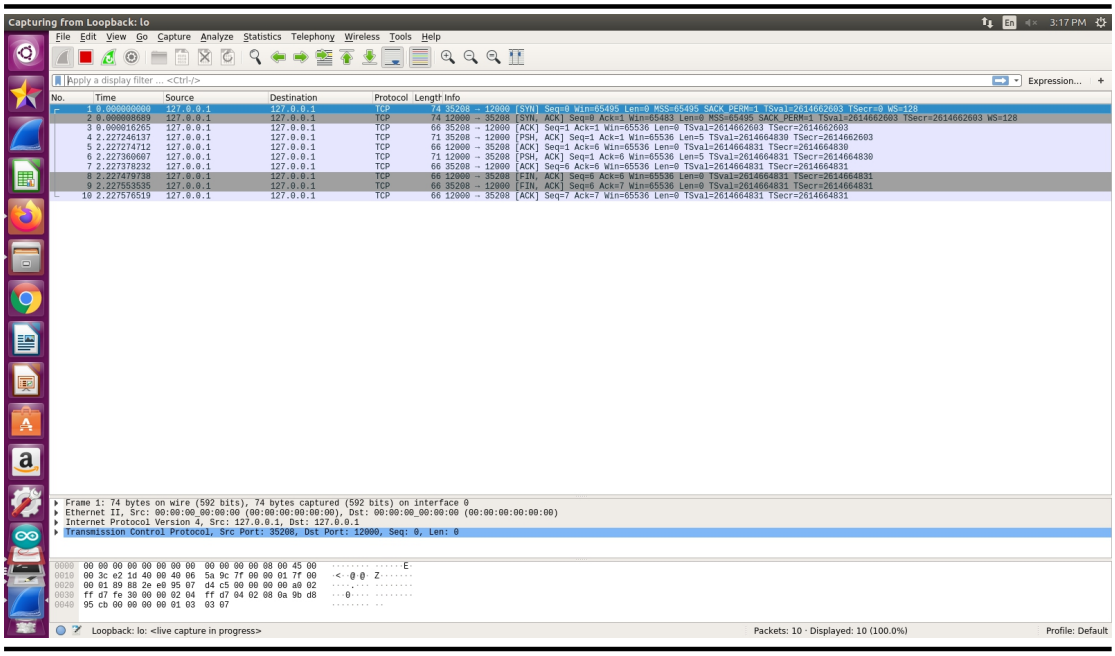
Task 2: Web Server

In this assignment, you will develop a simple Web server in Python that is capable of processing only one request. Specifically, your Web server will

a) create a connection socket when contacted by a client (browser);

b) receive the HTTP request from this connection;

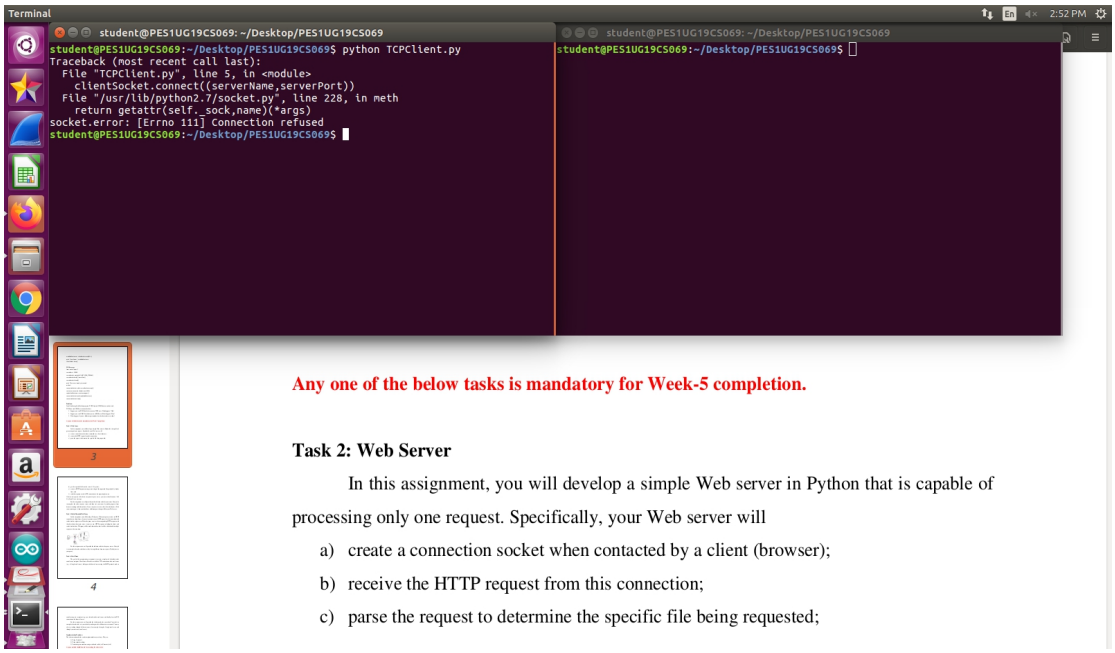
c) parse the request to determine the specific file being requested;



PROBLEMS:

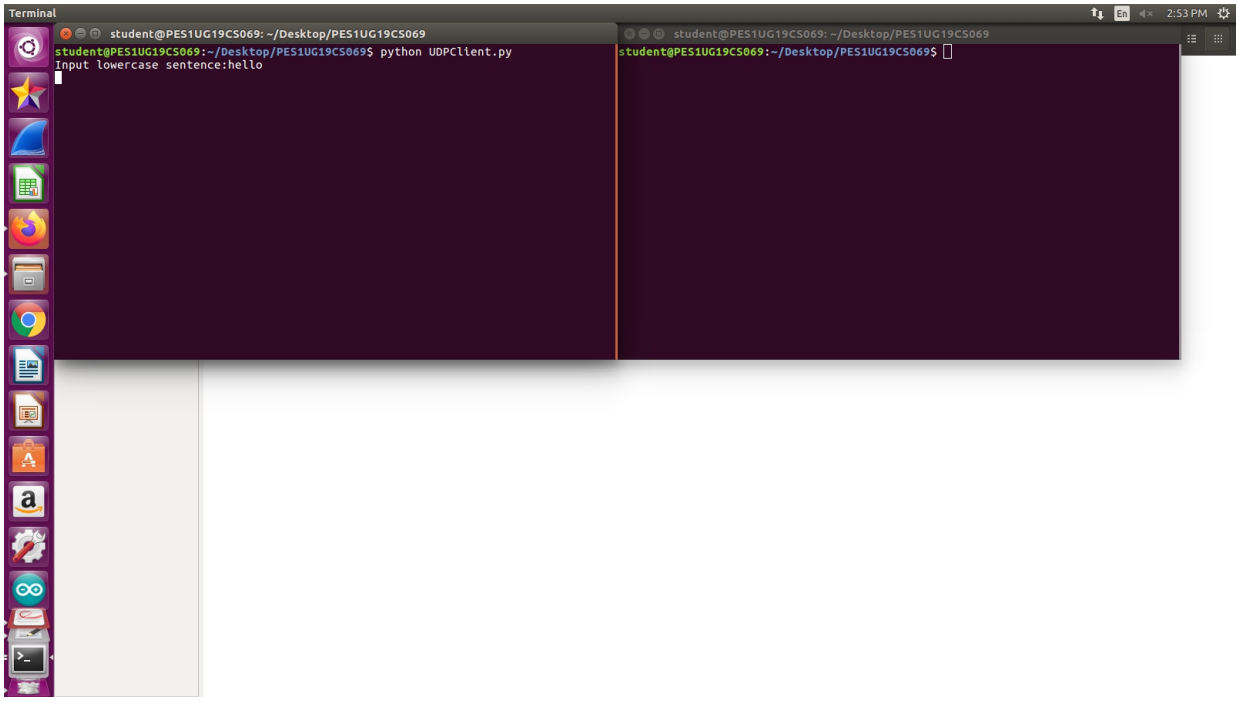
1. Suppose you run TCPClient before you run TCPServer. What happens? Why?

Ans)Suppose you run the TCPClient before running the TCPServer then the error is obtained is
socket error:connection refused



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

Ans)Suppose you run the UDPClient before UDPServer then the error obtained is that the packets are lost hence the UDPClient running terminal does not show any output.

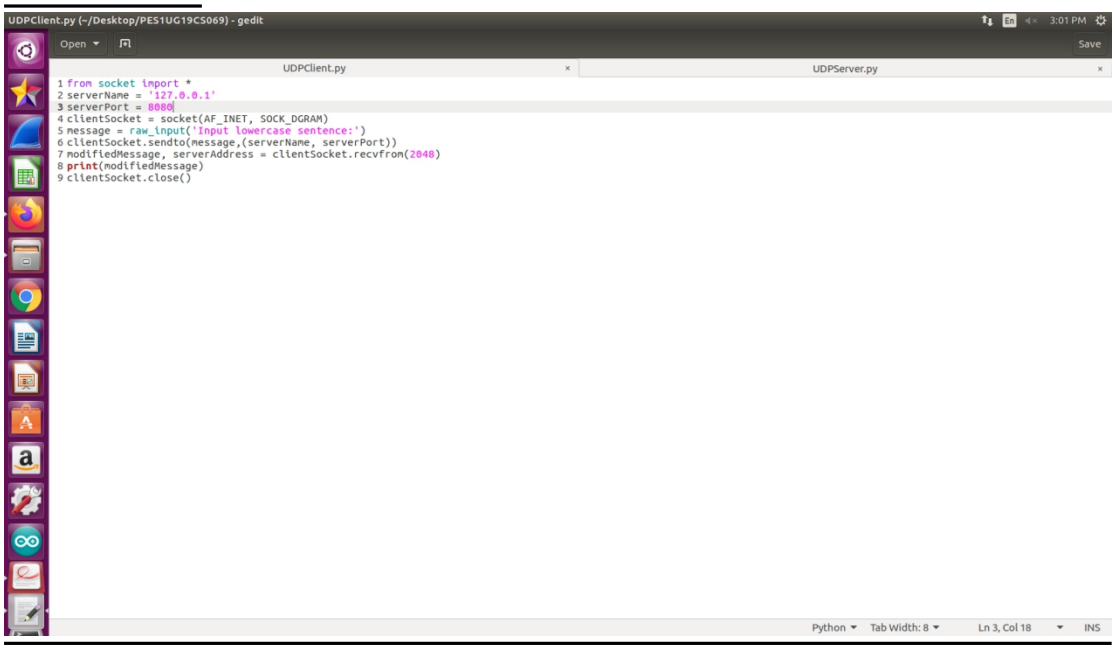


3. What happens if you use different port numbers for the client and server sides?

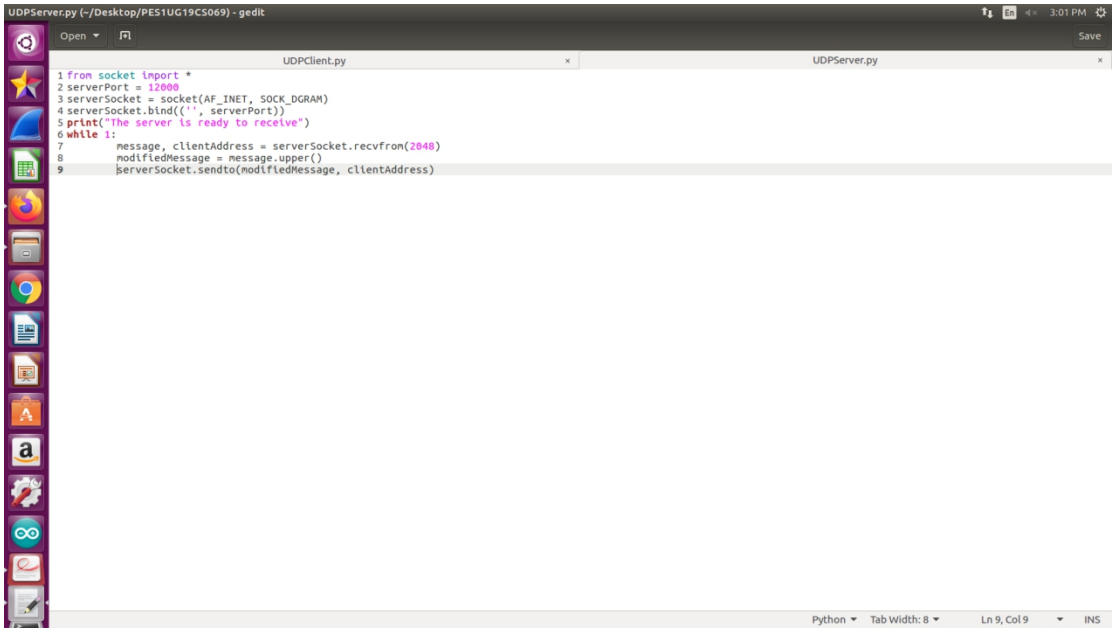
Ans) The following screenshots indicate the error obtained

a) <u>For UDP Case</u>

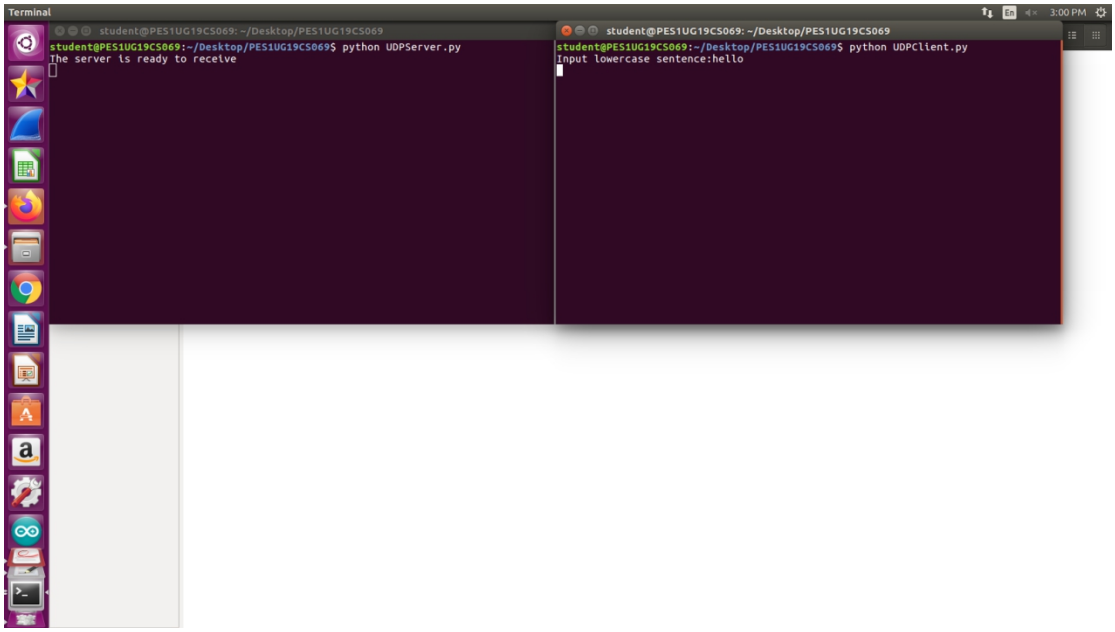
<u>UDPClient</u>



UDPServer



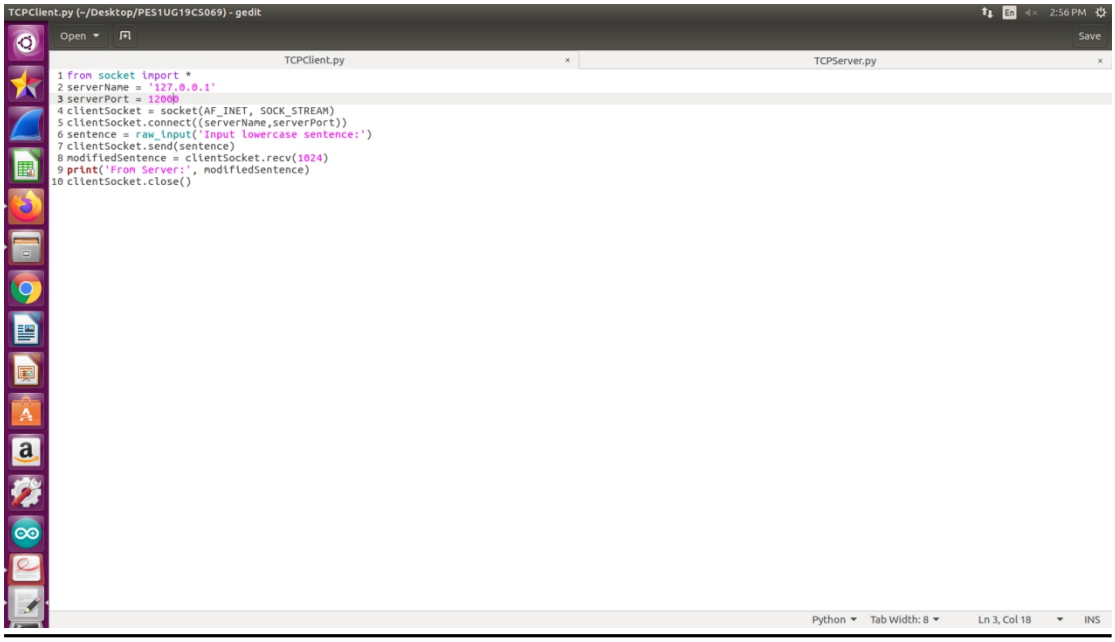
Terminal(screenshot)



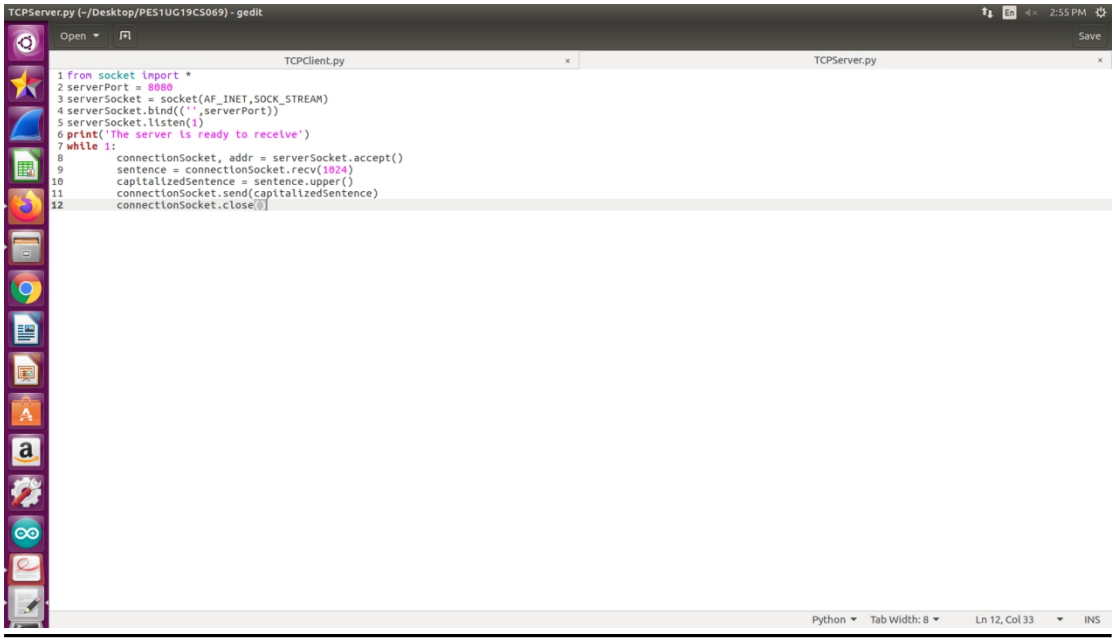
The error obtained is that the packets are lost hence the UDPClient running terminal does not show any output.

b) For TCP Case

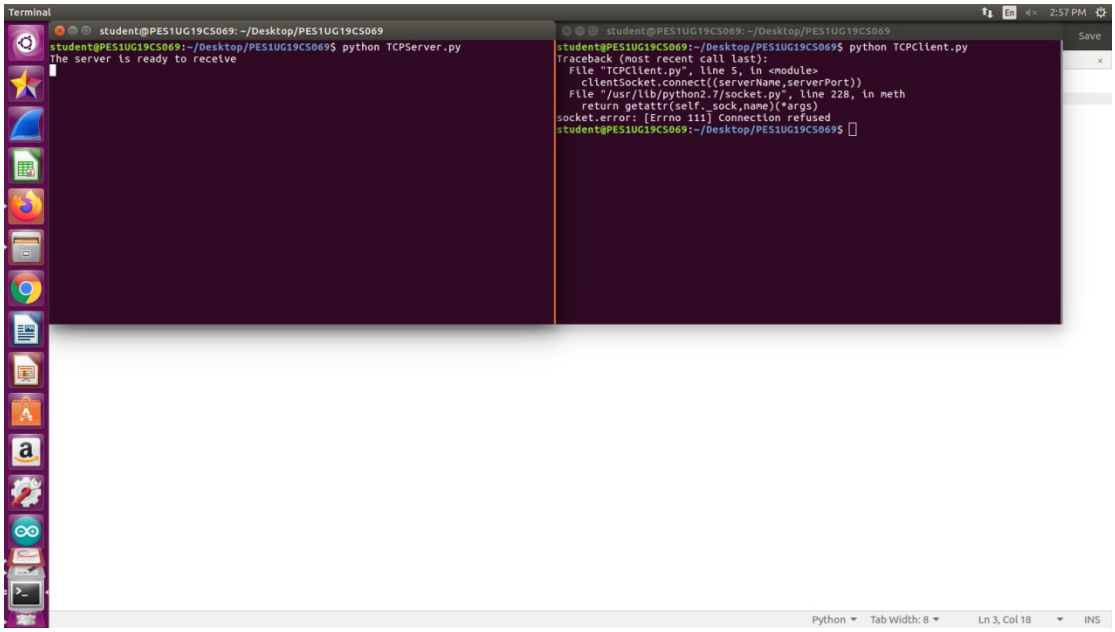
TCPClient



TCPServer



Terminal(screenshot)



The error is obtained is socket error:connection refused

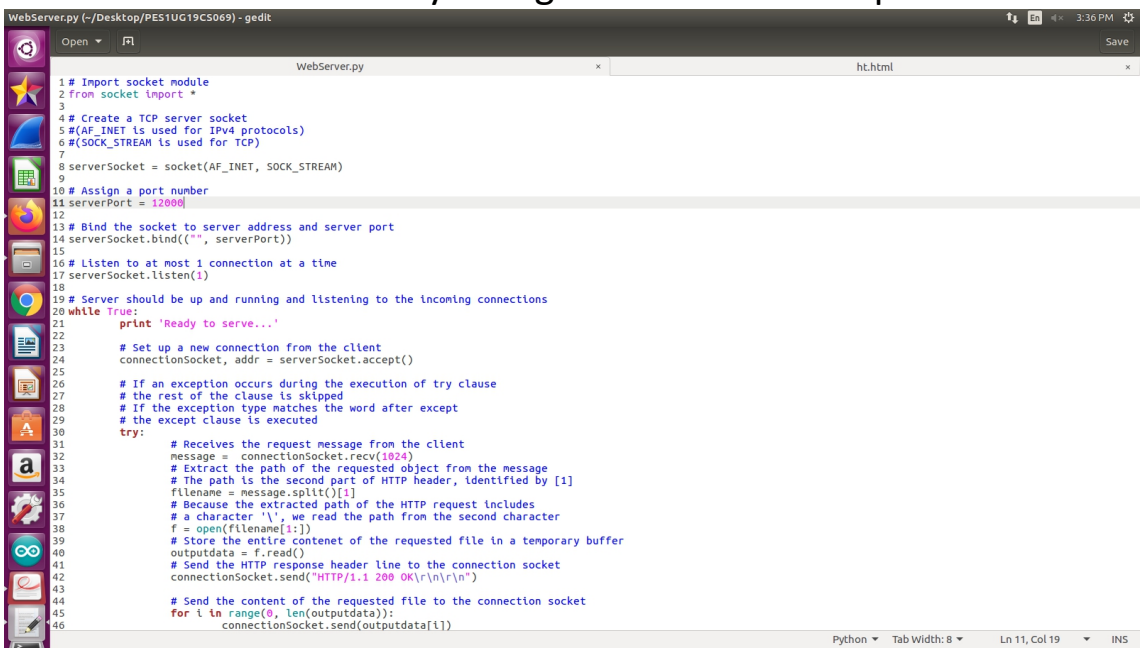
TASK 2

WEB SERVER

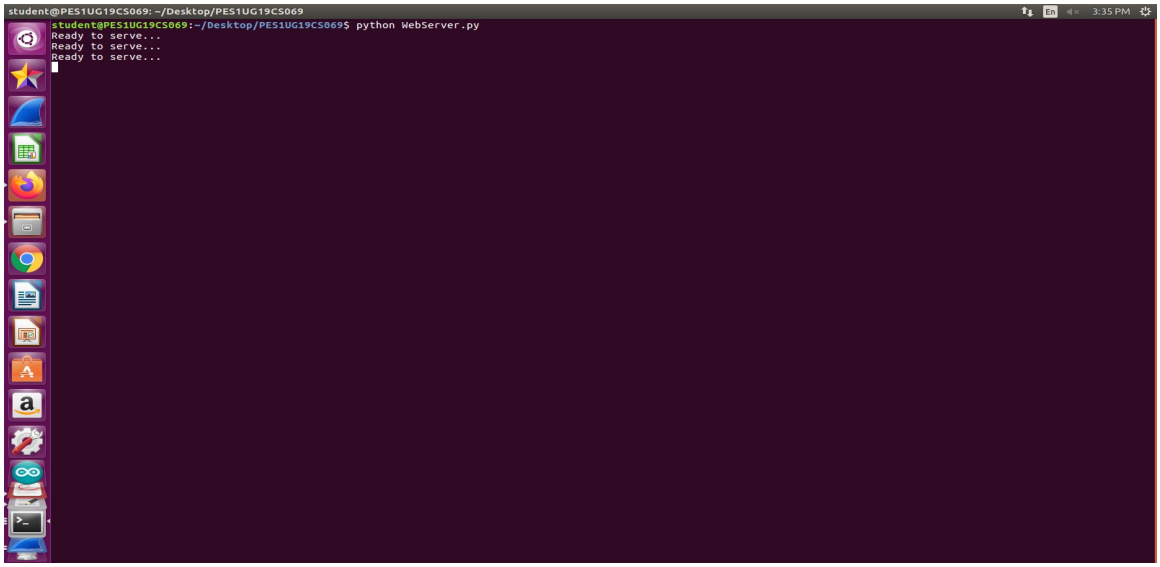
Test case 1(browser requesting a file that is present in the server):

Screenshot of code:

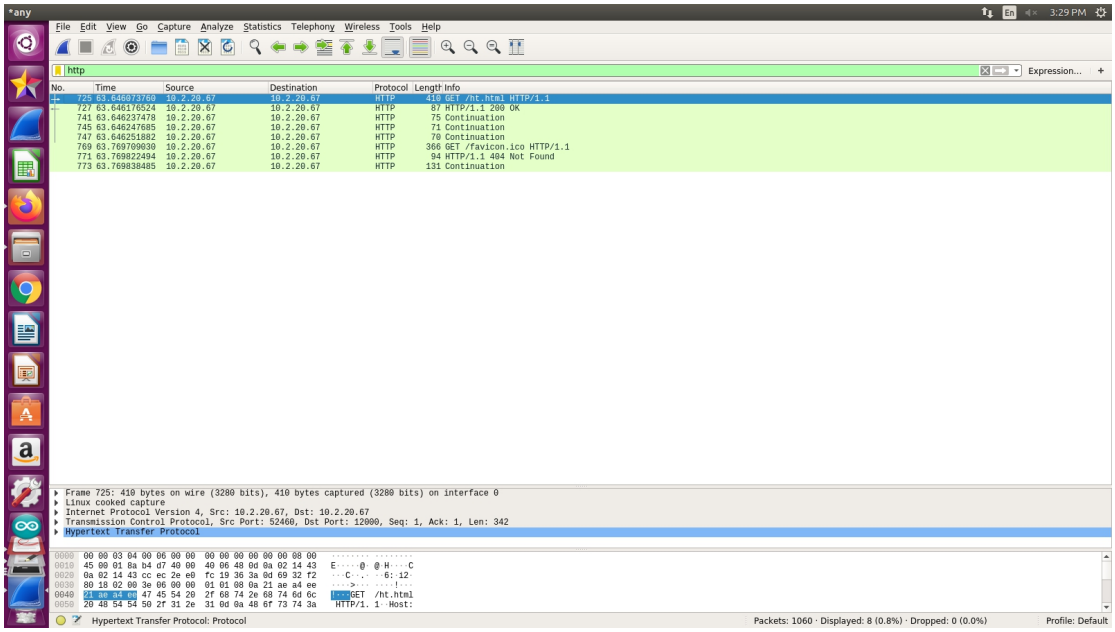
The html page I have taken is ht.html.
The ht.html contains only a single line code called pes.

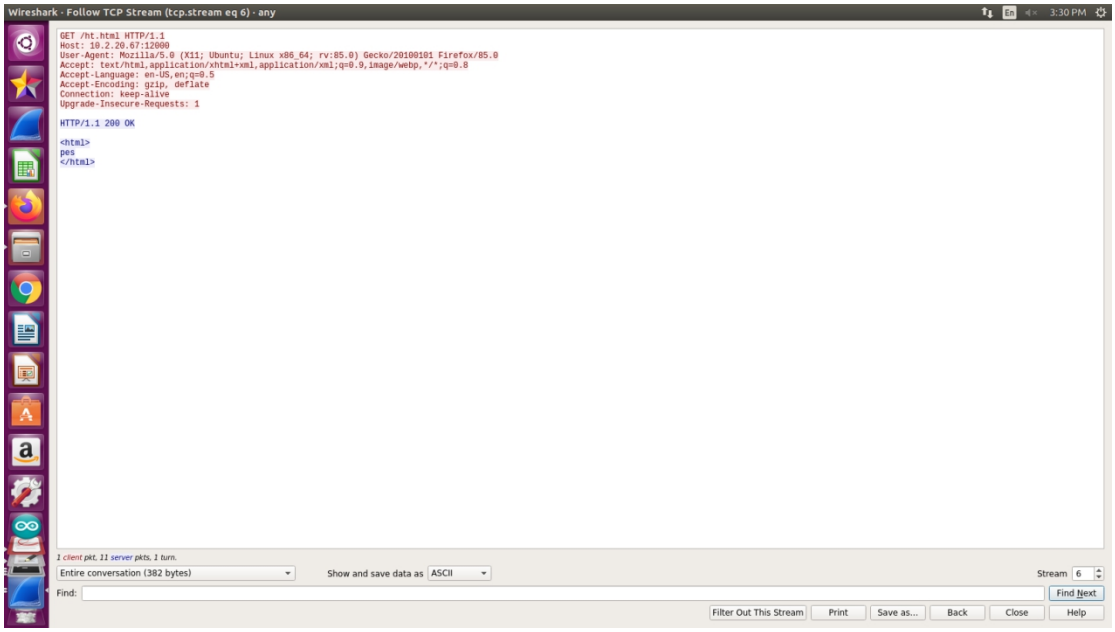


Screenshot of the server running:



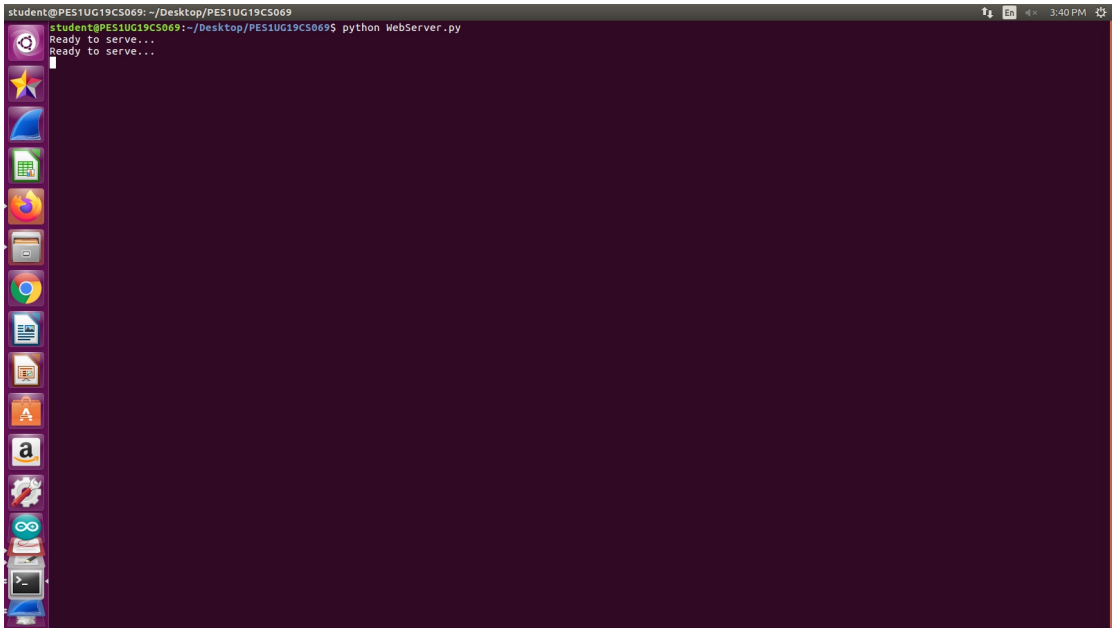
Wireshark screenshots:




Wireshark - Follow TCP Stream (tcp.stream eq 6) · any
3:30 PM
GET /ht.html HTTP/1.1
Host: 10.2.20.67:12000
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:85.0) Gecko/20100101 Firefox/85.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
HTTP/1.1 200 OK
<html>
pes
</html>
1 client pkt, 11 server pkts, 1 turn.
Entire conversation (382 bytes)
Show and save data as
ASCII
Stream
6
Find:
Find Next
Filter Out This Stream
Print
Save as...
Back
Close
Help

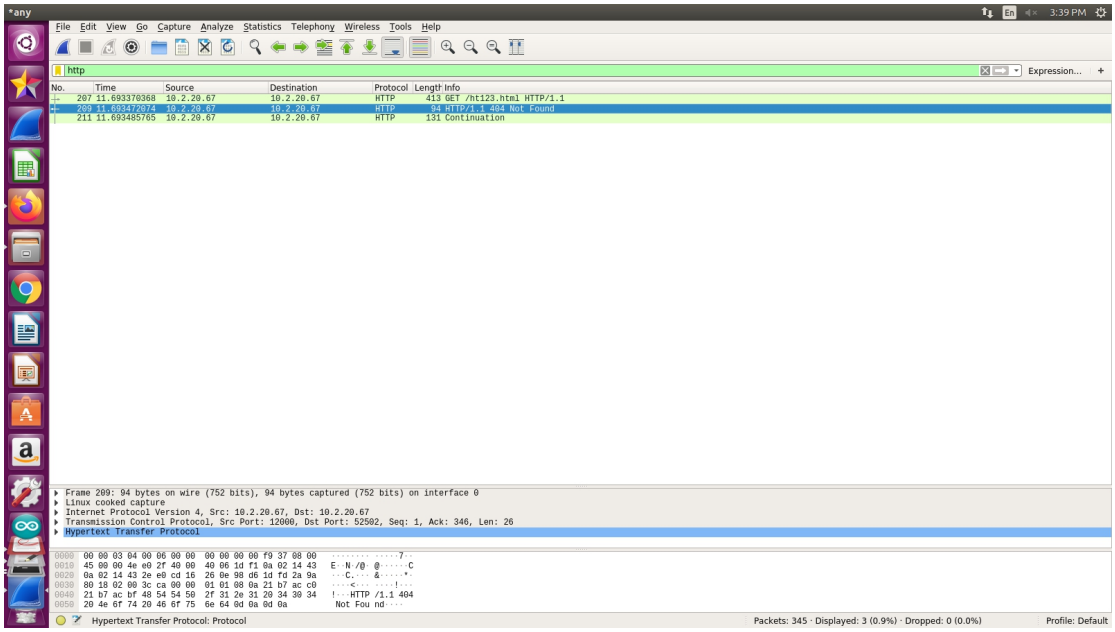
Test case 2(browser requesting a file that is not present in the server):

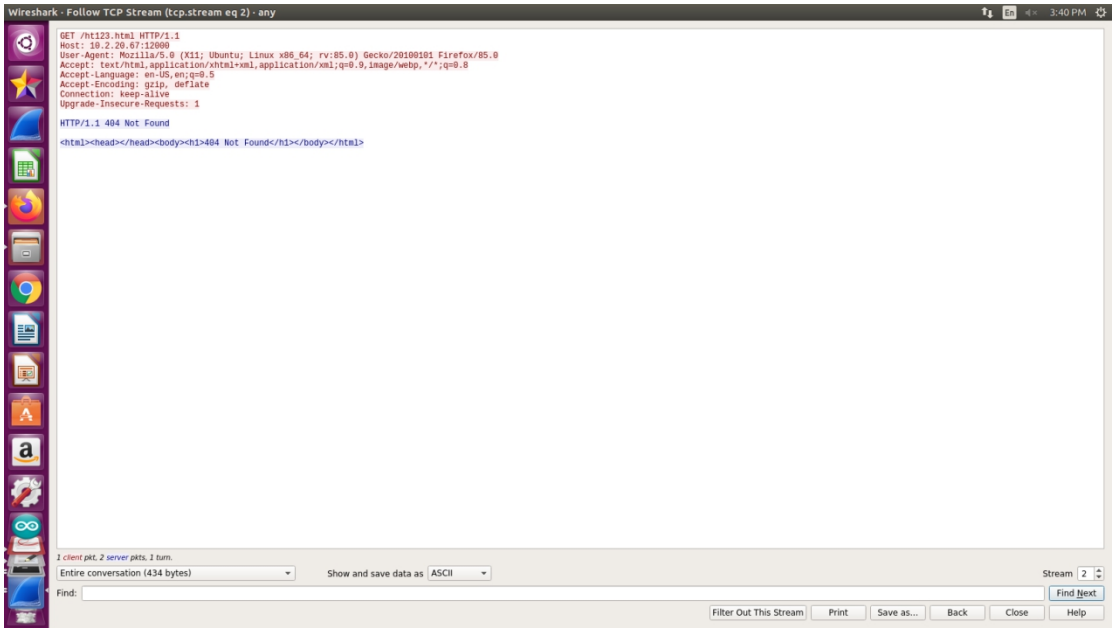
Screenshot of the server running:



Here I have used a file called ht123.html which is not present in the server.

Wireshark screenshots:





As I have used a file which is not present in the server hence I am getting an output 404 as seen in the above wireshark screenshots.