

CN Report

Assignment 3

1. Server1.py

The screenshot displays a Python client-server application. On the left, a Command Prompt window shows the client's execution. The user navigates to the directory 'E:\chinmay\6th Sem\CN\Asgn 3' and runs 'python client.py'. The client prompts for a request string, and the user enters 'HTTP/1.1\r\n'. The client then attempts to connect to the server at 192.168.56.1:5000. A connection error occurs, displaying a traceback: 'ConnectionRefusedError: [WinError 10061] No connection could be made because the target machine actively refused it'. On the right, the PyCharm IDE shows the server code (server1.py) and the Run console output. The server code imports the socket module, sets the port to 5000, and listens for connections. The Run console shows the server starting and listening on 192.168.56.1:5000. A new connection is established from 192.168.56.1 at port 62357.

```
Client 1
```

```
Client 2
```

```
Server
```

Client 1 initially established connection, hence client 2 wont be able to connect.

2. Server2.py

```
Command Prompt - python client.py
E:\chinmay\6th Sem\CN\Asgn 3>python client.py <- 1
Enter request string : 3+5 <- 3
Answer is 8
Enter request string : quit() <- 5
Closing Connection 192.168.56.1

E:\chinmay\6th Sem\CN\Asgn 3>python client.py <- 7
Enter request string : 

Command Prompt
E:\chinmay\6th Sem\CN\Asgn 3>python client.py <- 2
Enter request string : 4+5 <- 4
Answer is 9
Enter request string : quit() <- 6
Closing Connection 192.168.56.1

E:\chinmay\6th Sem\CN\Asgn 3>

server2.py
16
17
18
19
20
21
22
23
24
25
26
27
print(response.encode(FORMAT))
conn.sendall(response.encode(FORMAT))
print('response sent')
req = conn.recv(1024).decode(FORMAT).strip()

if req == 'quit()':
    conn.sendall('quit()'.encode(FORMAT))
    break

PORT = 5000
SERVER = socket.socket(socket.AF_INET, socket.SOCK_STREAM())
```

```
Run: server2.py
[STARTING] Server is starting...
[NEW CONNECTION] at ('192.168.56.1', 62370) connected <- 1
[ACTIVE CONNECTIONS] 1
[NEW CONNECTION] at ('192.168.56.1', 62371) connected <- 2
[ACTIVE CONNECTIONS] 2
Client ('192.168.56.1', 62370) sent -> 3+5 <- 3
Response sent to ('192.168.56.1', 62370)
Client ('192.168.56.1', 62371) sent -> 4+5 <- 4
Response sent to ('192.168.56.1', 62371)
[CONNECTION CLOSED] at ('192.168.56.1', 62370) <- 5
[CONNECTION CLOSED] at ('192.168.56.1', 62371) <- 6
[NEW CONNECTION] at ('192.168.56.1', 62374) connected <- 7
[ACTIVE CONNECTIONS] 1
```

Order 1 and 2 shows connection of **client 1** and **Client 2** respectively.

Order 3 and 4 shows request and response between clients and server.

3 -> req = 3+5 [Client_1]

Response = 8

4 -> req = 4+5 [Client_2]

Response = 9

5 -> req == quit() [Connection is closed between client 1 and server]

6 -> req == quit() [Connection is closed between client 2 and server]

(Server is still open to listen connection)

7 -> establishing connection between client 3 and server.

3. Server3.py

The screenshot shows a Python IDE with two files: `client.py` and `server3.py`. The `client.py` file contains a loop that sends requests and receives responses. The `server3.py` file contains a loop that receives requests and sends responses. The Run console shows the output of the server, including connection logs and response messages.

```
client.py
17 req_string += '\r\n'
18
19 client.sendall(req_string.encode(FORMAT))
20 response = client.recv(1024)
21 response = response.decode(FORMAT, errors='ignore')
22 if response == 'quit()':
23     client.close()
24     print(f'Closing Connection {SERVER}')
25     break
26
27 while True:
28     if response == 'quit()':
```

```
server3.py
1 [SERVER LISTNING] at ('192.168.56.1', 5000) <- 0
2 [INCOMING CONNECITON] from ('192.168.56.1', 62385) <- 1
3 [INCOMING CONNECITON] from ('192.168.56.1', 62386) <- 2
4 Client ('192.168.56.1', 62386) sent -> 4+5 <- 3
5 Client ('192.168.56.1', 62386) sent -> 6+8 <- 4
6 Client ('192.168.56.1', 62386) sent -> quit() <- 5
7 [Closing connection] on client ('192.168.56.1', 62386) request
8 Client ('192.168.56.1', 62386) sent -> quit() <- 6
9 [Closing connection] on client ('192.168.56.1', 62386) request
10 [INCOMING CONNECITON] from ('192.168.56.1', 62387) <- 7
11 Client ('192.168.56.1', 62387) sent -> exit() <- 8
12
13 Process finished with exit code 0
```

Setblocking is “True” by-default, hence it is set to “False” so that control over user must be entirely on user rather than kernel

```
server.setblocking(False)
```

Lists for inputs and outputs are used to store connection requests and their respective outputs

```
inputs = [server]
outputs = []
```

Order **1** and **2** for establishing connection.

Order **3** and **4** indicates requests which are then evaluated by server.

Order **5** and **6** indicates closing of connection between clients and server3.

Order **7** shows new connection from client3.

Order **8** shows `exit()` request which closed the **connection and halted** the server socket.

4. Server4.py

```
Command Prompt
E:\chinmay\6th Sem\CN\Asgn 3>python client.py <- 1
Enter request string : Hello <-3
Request recieved as : Hello
Enter request string : How are you <- 5
Request recieved as : How are you
Enter request string : quit() <- 7
Closing Connection 192.168.56.1
E:\chinmay\6th Sem\CN\Asgn 3>

Command Prompt
E:\chinmay\6th Sem\CN\Asgn 3>python client.py <- 2
Enter request string : Hi <- 4
Request recieved as : Hi
Enter request string : I am fine :) <- 6
Request recieved as : I am fine :)
Enter request string : quit() <- 8
Closing Connection 192.168.56.1
E:\chinmay\6th Sem\CN\Asgn 3>

server4.py
22 if response == 'quit()':
23     client.close()
24     print(f'Closing Connection {SERVER}')
25     break
26 elif response == 'exit()':
27     client.close()
28     print(f'Closing Connection {SERVER}')
29     exit(0)
30     print(response, end='')
while True:
    if response == 'quit()':
```

```
Run: server4
C:\Users\ACER\AppData\Local\Programs\Python\Python38-32\python.exe "E:\chinmay\6th S
[SERVER LISTNING] at ('192.168.56.1', 5000) <- 0
[INCOMMMING CONNECITON] from ('192.168.56.1', 62419) <- 1
[INCOMMMING CONNECITON] from ('192.168.56.1', 62420) <- 2
Client ('192.168.56.1', 62419) sent -> Hello <- 3
Client ('192.168.56.1', 62420) sent -> Hi <- 4
Client ('192.168.56.1', 62419) sent -> How are you <- 5
Client ('192.168.56.1', 62420) sent -> I am fine :) <- 6
Client ('192.168.56.1', 62419) sent -> quit() <- 7
[Closing connection] on client ('192.168.56.1', 62420) request <- 7
Client ('192.168.56.1', 62420) sent -> quit() <- 8
[Closing connection] on client ('192.168.56.1', 62420) request <- 8
```

Server_4 differs from Server_3 by request handle, unlike server_3 it does not evaluate the request but rather just respond with the same string.

Beside Order 1 and 2 (connection), Order 3 to 8 shows set of request-response pair between clients and the server.