CS-342 Computer Networks Assignment Lab: 03

Group Number**: 61**
Group Members :
NAMAN GOYAL - 180123029
AYAZ ANIS - 180123007
Application Id: 07

There are 3 files in our zip folder.

1. Client.c
2. Server.c
3. DataItems.txt

<u>Steps to compile and run our source code in the Ubuntu environment.</u>

- **Starting the Server:**

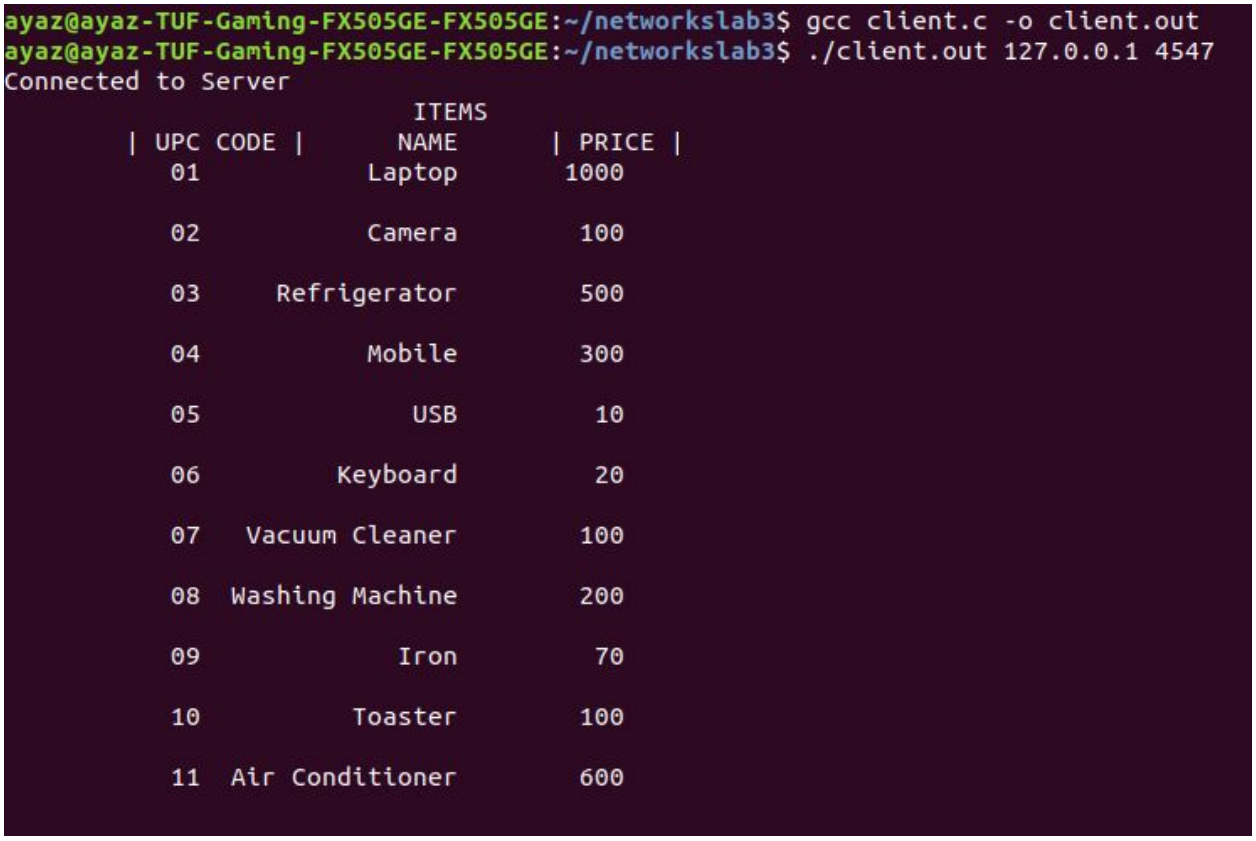
```
ayaz@ayaz-TUF-Gaming-FX505GE-FX505GE:~/networkslab3$ gcc -pthread server.c -o server.ou
t
ayaz@ayaz-TUF-Gaming-FX505GE-FX505GE:~/networkslab3$ ./server.out 4547
Listener on port 4547
Waiting for connections ...
New connection established
socket id : 4 , ip : 127.0.0.1 , port : 43716

New connection established
socket id : 5 , ip : 127.0.0.1 , port : 43788
```

-> We are using Multithreading during the process.

- **$gcc -pthread server.c -o server.out**: this command is to compile server.c file which generates an executable server.out file.
- To execute server.out file run the command : **$./server.out 4547** Here 4547 represents the port number.

- **Starting a Client :**



- **$gcc client.c -o client.out:** this command is to compile the client.c file which generates an executable client.out file.
- To execute the client.out file run the following command : **$./client.out 127.0.0.1 4547**.

Here 127.0.0.1 represents the local host's IP Address since we have the same machine for clients as well as servers. **Local Host's IP = 127.0.0.1** And **Port Number: 4547.**

- After running the client file, as soon as we are connected to the server, we can have the data items (shown in the above picture) which are being fetched from our database file: **DataItems.txt** which contains the 3-digit UPC no, Item Name, Item price for all the items listed.

```
          010          Toaster        100

          011  Air Conditioner        600



Type = 0 for adding items to cart
Type = 1 for closing connection with server and checking out

Enter request type:
```

- A request has been requested here by the client for the server. Here, 0 is for adding some item to our cart, 1 means to calculate the total billing for our cart which closes the connection with the server as well.

```
Type = 0 for adding items to cart
Type = 1 for closing connection with server and checking out

Enter request type: 0
Enter UPC code: 001
Enter Quantity: 20
-----------------------------
Name : Laptop | Price : 1000
-----------------------------

Enter request type: 0
Enter UPC code: 003
Enter Quantity: 50
-----------------------------
Name : Refrigerator | Price : 500
-----------------------------

Enter request type: 0
Enter UPC code: 011
Enter Quantity: 30
-----------------------------
Name : Air Conditioner | Price : 600
-----------------------------

Enter request type: 1
Total Bill = 63000
Closed from client side
ayaz@ayaz-TUF-Gaming-FX505GE-FX505GE:~/networkslab3$
```

- Items could be added to our cart and checked out as shown in the above picture.
- For a **0** request type: A **UPC Code** is being asked which is the ID of the item and the **Quantity** is being asked which represents the number of items that a client can request to add to the cart.
- For a **1** request type: The total billing amount is shown to the client and the server closes the connection.

- **Concurrent Server:**

```
ayaz@ayaz-TUF-Gaming-FX505GE-FX505GE:~/networkslab3$ ./server.out 4547
Listener on port 4547
Waiting for connections ...
New connection established
socket id : 4 , ip : 127.0.0.1 , port : 45028

New connection established
socket id : 5 , ip : 127.0.0.1 , port : 45186

New connection established
socket id : 6 , ip : 127.0.0.1 , port : 45192

New connection established
socket id : 7 , ip : 127.0.0.1 , port : 45198
```

- The above picture depicts when a server has been connected to various clients at the same time and each client is requesting from the server. We can see that we have 4 clients here having unique ids, and port numbers which are currently connected to the server.
- Hence the server is a "**Concurrent Server**", i.e., a server that accepts connections from multiple clients and serves all of them concurrently.

- **Errors:**

```
Type = 0 for adding items to cart
Type = 1 for closing connection with server and checking out

Enter request type: 0
Enter UPC code: 25
Enter Quantity: 50
Error Message: UPC is not found in database

Enter request type: 2
Enter request type either 0 or 1

Enter request type:
```

- Above we can see 2 errors regarding UPC Code and request type. If the UPC Code is not present in the data items list (or database), it will show an error message "**UPC is not found in database**" and again a request is generated. The request type accepted is only 0/1, if a user enters other than above, an error message is displayed showing to enter the request type which is either 0 or 1.

- **Terminating the Server**:

```
ayaz@ayaz-TUF-Gaming-FX505GE-FX505GE:~/networkslab3$ ./server.out 4547
Listener on port 4547
Waiting for connections ...
New connection established
socket id : 4 , ip : 127.0.0.1 , port : 45028

New connection established
socket id : 5 , ip : 127.0.0.1 , port : 45186

New connection established
socket id : 6 , ip : 127.0.0.1 , port : 45192

New connection established
socket id : 7 , ip : 127.0.0.1 , port : 45198

^CServer Exited Gracefully
ayaz@ayaz-TUF-Gaming-FX505GE-FX505GE:~/networkslab3$
```

- The above checks for the **CTRL+C Handler** in the case of the server. While connected, if the user presses CTRL+C, connection to the server is gracefully terminated. Then the server gracefully releases the open socket.