Data Communication and Computer Networks III (EC345)

Simple Online e-Shopper Server and a Customer Program

Assignment 2019-June

Due date: 22nd May 2018

The task of the assignment is to implement a simple Online Supermarket Server (e-ShopperServer) and Customer (e-ShopperCustomer) programs to conduct online buying and selling.

The System, as a whole, consists of major three components:

1. A Server program (e-ShopperServer) which is capable of list down available product information to the customer from a stock database (Stock.txt), calculate total bill based on unit price and send to customer, and reduce number of items that customer purchased from the stock database after customer pay the bill.

2. A Customer program (e-ShopperCustomer) which is capable of requesting available product information, receiving product information, displaying product information in the terminal, sending chosen product and purchasing quantity to the server, finally displaying the total bill in the terminal of the customer and ask customer to pay the bill.

3. A Simple text file that works as the stock database (please refer the given stock.txt for the format). However, note that you are free to use any format.

One instance of the system is illustrated in Figure 1.

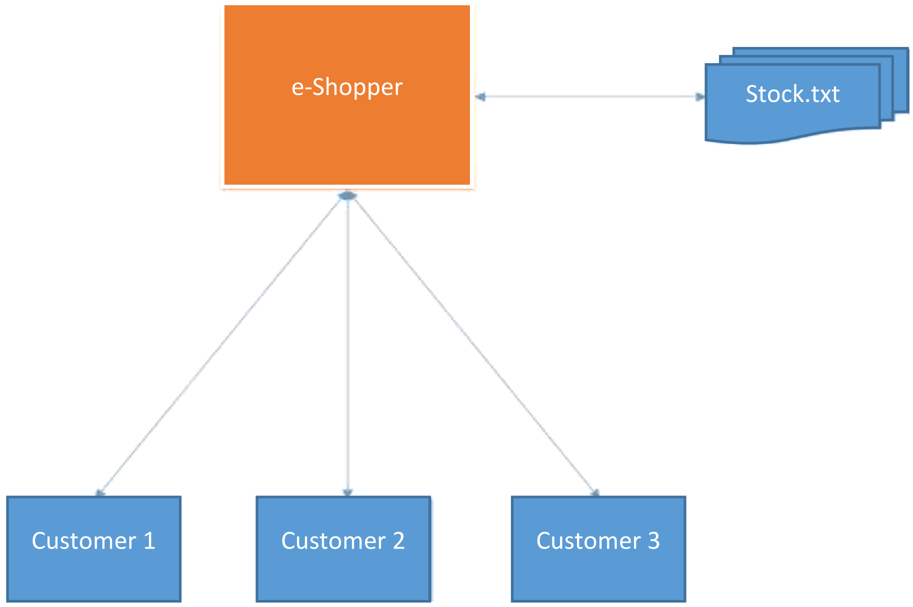


Figure 1

Objectives you should achieve in e-ShopperServer.

1. Listen to the customers' connections.
2. When a new customer connects to the server, the server should create either thread or a process to handle the customer.
3. As soon as new Thread or a Process created, server should send 'Welcome message' to the client as follows "Welcome to e-Shopper. Buy best electronic goods under one roof. Find latest electronic goods available with us: <You can hard-code Product type or get Product type by reading values from stock.txt file >.
   You can buy following products through e-Shopper today:
   Blender
   Camera
   Heater
   Please find best brand by giving first 4 letters of the product."

4. Then server should wait until "Customer" sends first 4 letters of the product type.
   Ex: **blen** for Blender, **came** for Camera
5. When server received a **product type** message from the Customer, the server should then send product brand name list for requested product type to the "Customer" by reading the "stock.txt". Product brand name list should include product id along with the brand name, available quantity and unit price.
   Please use the "stock.txt" provided with the assignment.
6. Upon receiving product number and purchasing quantity from the e-ShopperCustomer as buy message (i.e., with "b" tag), the server should refer the "stock.txt" and calculate the total cost for customer selected product and send back to the customer.
   Note: If purchasing quantity is higher than available quantity an error message should send to the customer
7. After receiving the pay message (i.e., with "p" tag) along with product number and purchased quantity from the e-ShopperCustomer, the server should update the stock database in "stock.txt" by reducing purchased quantity from available quantity. Then display, "Available quantity is : #" and send notification "Your order processed successfully!" to the customer.

Guidelines to design the e-ShopperServer

1. The server should listen on PORT 10000.
2. Server should be a concurrent server
3. Moreover, Servers should not make any zombie processes (if you use fork () to implement to the concurrent server) upon e-ShopperCustomer termination.
4. Server should handle EINTR interrupts
5. The server should display the Customer's information: IP address and the PORT number, in the Server's terminal (refer Figure 3). If there are multiple customers at the same time, Server should display the information of all customers.
6. The server should maintain separate file descriptors for the listening socket, connection sockets, and for the stock.txt.
7. You are free to use either C or C++ languages to implement the e-ShopperServer program.
8. You are free to use any file from your practical programs to implement your e-ShopperServer Program.

Objectives you should achieve in e-ShopperCustomer.

1. Make a connection with the server by giving the host name or the IP address.

   E.g., e-ShopperCustomer <IP ADDRESS>

   or

   ./ e-ShopperCustomer <Computer name>
2. Wait until e-ShopperCustomer receives the welcome message from the server
3. Then type ‘**blen**’ (first 4 letters of the product type) and send the request message to the e-ShopperServer. Then the customer should wait until server sends the product brand information (Please refer the scenario depicted in Figure 2)
4. When e-ShopperCustomer received product information from the server, Customer should display it in the terminal as follows:

   ```
   PID        Brand name        Available quantity     Unit price
   001        Vitamix5200       10                     3000.00
   002        Vitamix750        4                      7000.00
   003        Nutribullet       10                     2500.00
   004        MagicBullet       10                     3000.00
   ```
5. After the user select a product to buy, the customer should send the message to the server as three values: <message type> <product number > <purchasing quantity >, separated by “space.”

   e.g., **b 001 2**
6. Then the customer should wait for the server’s response (Total cost) and display it in the terminal.
7. The customer should send a message to the server to pay the bill and purchase selected product as three values :

   <message type> <product number > <purchasing quantity >, separated by “space.”

   e.g., **p 001 2**
8. Finally, the customer should wait for the server’s notification, “Your order processed successfully” after successful database update.

   One instance of the e-ShopperCustomer could be as follows:

```
e-ShopperPC$ ./ e-ShopperCustomer 127.0.0.1
Welcome to e-Shopper. Buy best electronic goods under one roof. Find latest electronic
goods available with us.
You can buy following products through e-Shopper today:
        Blender
        Camera
        Heater
Please find best brand by giving first 4 letters of the product
blen
```

```
PID     Brand name      Available quantity      Unit price
001     Vitamix5200     10                       3000.00
002     Vitamix750       4                       7000.00
003     Nutribullet     10                       2500.00
004     MagicBullet     10                       3000.00
b 001 2
Total Cost  : 6000.00 Press "P" to confirm the appointment
p 001 2
Your order processed successfully!
```

Please use the scenario illustrated in Figures 2,3 and 4 as a reference to implement the server.

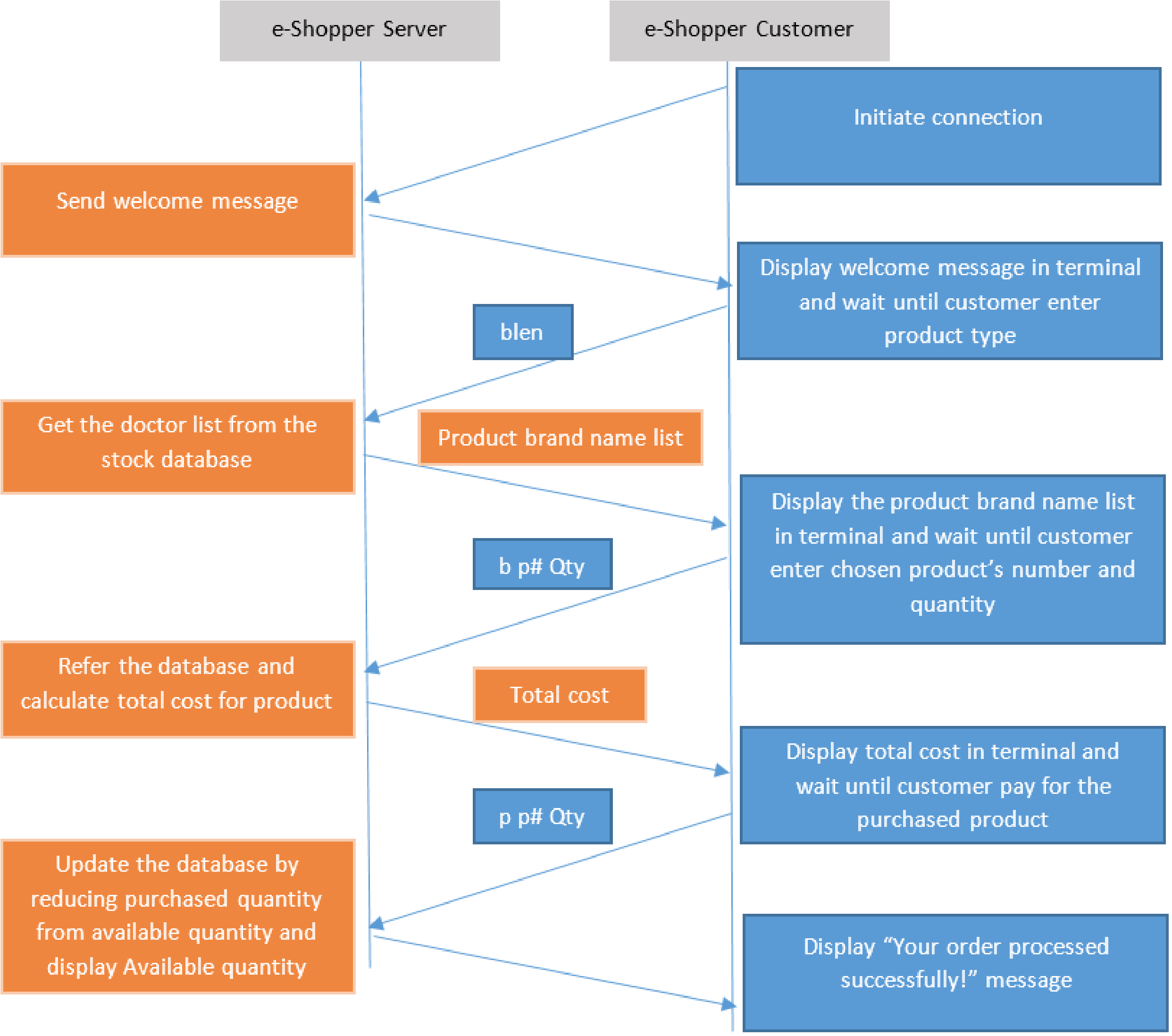
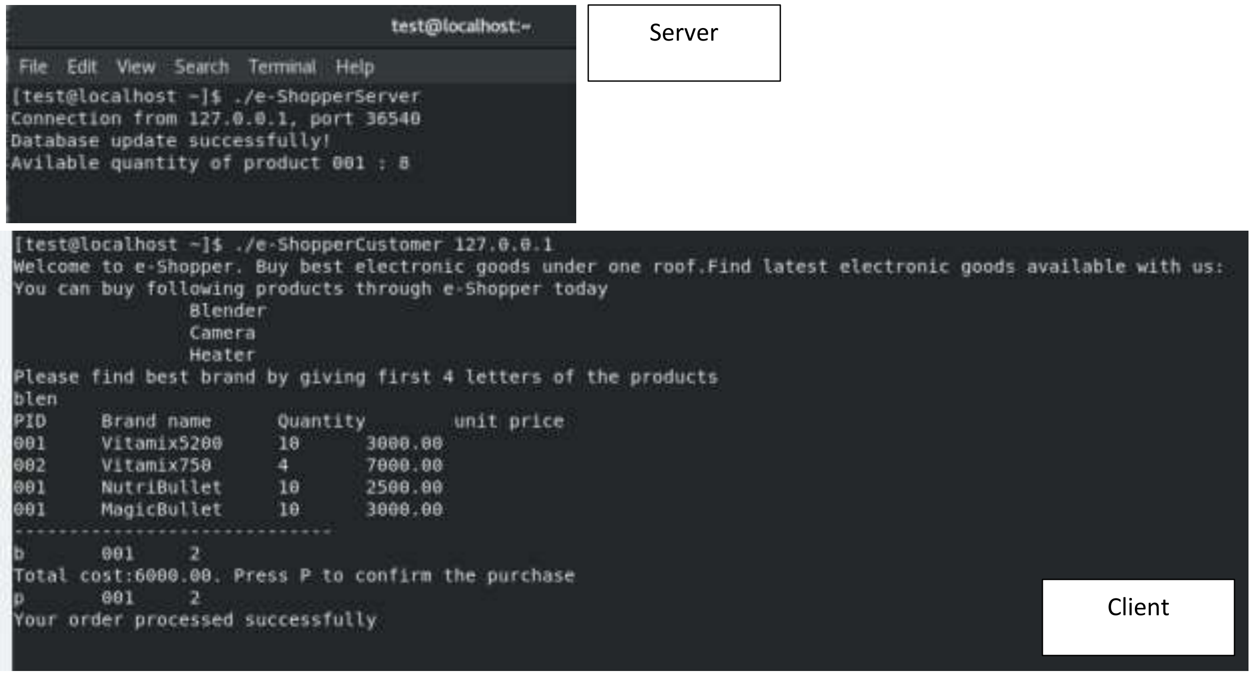

e-Shopper Server
e-Shopper Customer
Initiate connection
Send welcome message
Display welcome message in terminal and wait until customer enter product type
blen
Get the doctor list from the stock database
Product brand name list
Display the product brand name list in terminal and wait until customer enter chosen product's number and quantity
b p# Qty
Refer the database and calculate total cost for product
Total cost
Display total cost in terminal and wait until customer pay for the purchased product
p p# Qty
Update the database by reducing purchased quantity from available quantity and display Available quantity
Display "Your order processed successfully!" message

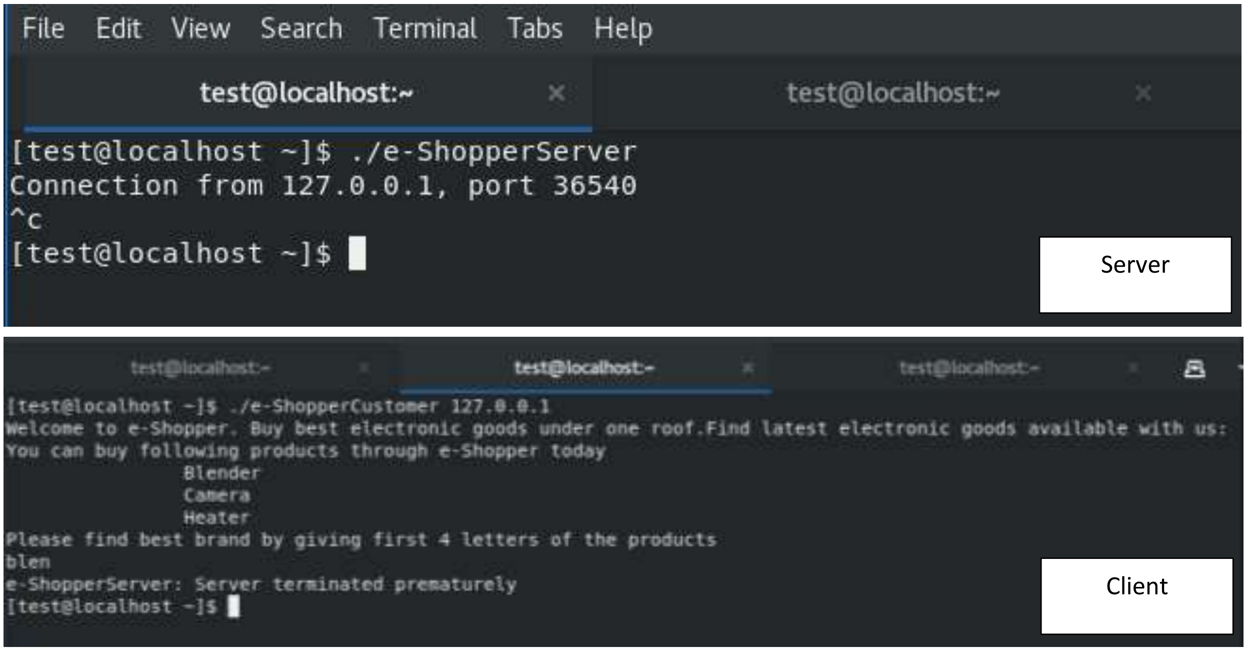
Figure 2



```
[test@localhost ~]$ ./e-ShopperServer
Connection from 127.0.0.1, port 36540
Database update successfully!
Avilable quantity of product 001 : 8
```

```
[test@localhost ~]$ ./e-ShopperCustomer 127.0.0.1
Welcome to e-Shopper. Buy best electronic goods under one roof.Find latest electronic goods available with us:
You can buy following products through e-Shopper today
                Blender
                Camera
                Heater
Please find best brand by giving first 4 letters of the products
blen
PID     Brand name      Quantity        unit price
001     Vitamix5200     10      3000.00
002     Vitamix750      4       7000.00
001     NutriBullet     10      2500.00
001     MagicBullet     10      3000.00
-----------------------------
b       001     2
Total cost:6000.00. Press P to confirm the purchase
p       001     2
Your order processed successfully
```

Figure 3: Server and client at successful execution



```
[test@localhost ~]$ ./e-ShopperServer
Connection from 127.0.0.1, port 36540
^c
[test@localhost ~]$ 
```

```
[test@localhost ~]$ ./e-ShopperCustomer 127.0.0.1
Welcome to e-Shopper. Buy best electronic goods under one roof.Find latest electronic goods available with us:
You can buy following products through e-Shopper today
                Blender
                Camera
                Heater
Please find best brand by giving first 4 letters of the products
blen
e-ShopperServer: Server terminated prematurely
[test@localhost ~]$ 
```

Figure 4: Server and client when handling server process termination

Guidelines to design the Handling Server Process Termination

1. You should use I/O Multiplexing to handle the STDIN and the Sockets in the e-ShopperCustomer
2. You should implement all e-ShopperCustomer to exit when the server crashes (Hint: use select() system call).
3. You are free to use either C or C++ languages to implement the e-ShopperCustomer program.
4. You are free to use any file from your Practical programs to implement your e-ShopperCustomer program.
5. You should implement applicable (necessary) methods discussed in chapter 5 of the reference book.

Guidelines to draft the Stock.txt (Stock Database)

1. You are free to use any format to design the Stock database
2. The file should include Product Number, Product, Brand name, Available quantity, and the unit price of a product.
3. Please use the given Stock.txt as necessary. This file was created by separating the Product Number, Product, Available quantity, and the unit price by using "tabs."
4. Your server program should be capable of separating the Product Number, Brand name, Product, Available quantity, and the unit price accordingly.

Guidelines for the submission, demonstration, and viva session.

1. If you download, copy, or substitute some examples from the module reference book, module practical sheets, online repositories, GitHub, StackOverflow, or some third party websites, you should add a comment and mention the place you copied the coding segment.
2. The implemented e-Shopper server and the customers, along with the necessary header files should be compiled and run on the laboratory PCs.
3. Your completed program files and all other necessary files and makefiles must be placed in a directory named:
   DCCNIII_<ID:GroupLeader>_<ID:Member1>_<ID:member2>
   After that, you should create a .zip file of that folder and upload it to the link provided in the course web.
4. The group leader should submit the .zip file representing all you members.
5. **No late submissions will be accepted under any circumstance.**
6. If your assignment is incomplete due to illness or other circumstances on or near the submission date you must submit the partial solution you have completed. The lecturer and/or the Department's Board of Examiners will take any exceptional circumstance, supported by a medical certificate.
7. You may be required to demonstrate your program to the LIC and the assigned instructors during the decided practical session.
8. The programs should run on any machine in the allocated laboratory.
9. **The members of the panel will evaluate you all based on the**
   a. **design guidelines of both server and customer given above,**

b. percentage you completed,

c. percentage of the codes you copied from third party materials (practicle worksheets excluded),

d. Clarity of the answers you provide.

Duties to Cover

1. The leader is responsible for dividing the duties among the members.
2. It is also the duty of the leader to present at the viva session about the status of the final program, i.e., finished the task or the percentage you all finished the task.
3. Each member will get a time slot to explain the part he/she received from the leader and the percentage he/she achieved the given task.
4. You are encouraged to write your own coding without copying from third party sources, e.g., websites, blogs, and online examples.
5. **In case if you caught copying the codings from third party sources without giving proper acknowledgments, as explained in the section "guidelines to submit and demonstrate," the entire team will get zero marks.**
6. In case if you get some codings from the third party sources, you will be subject to questioning by the evaluation panel about the logic behind the copied codes, and you should be able to answer the questions to get marks.

Schedule for the Viva Sessions

Final submission 17th May 2018

Viva and Demo Session 22ne May 2018 on A405 from 5.30 P.M. to 7.30 P.M.

Failure to meet these requirements may result in the assignment not being marked.