ASSIGNMENT 5

OBJECT ORIENTED DESIGN AND PROGRAM

TOSHA KULKARNI

**PROBLEM STATEMENT**: The main goal of the project is to design an online marketplace handling various events like:

Registration and Login

Browsing Items

Updating Items

Removing Items

Adding Items

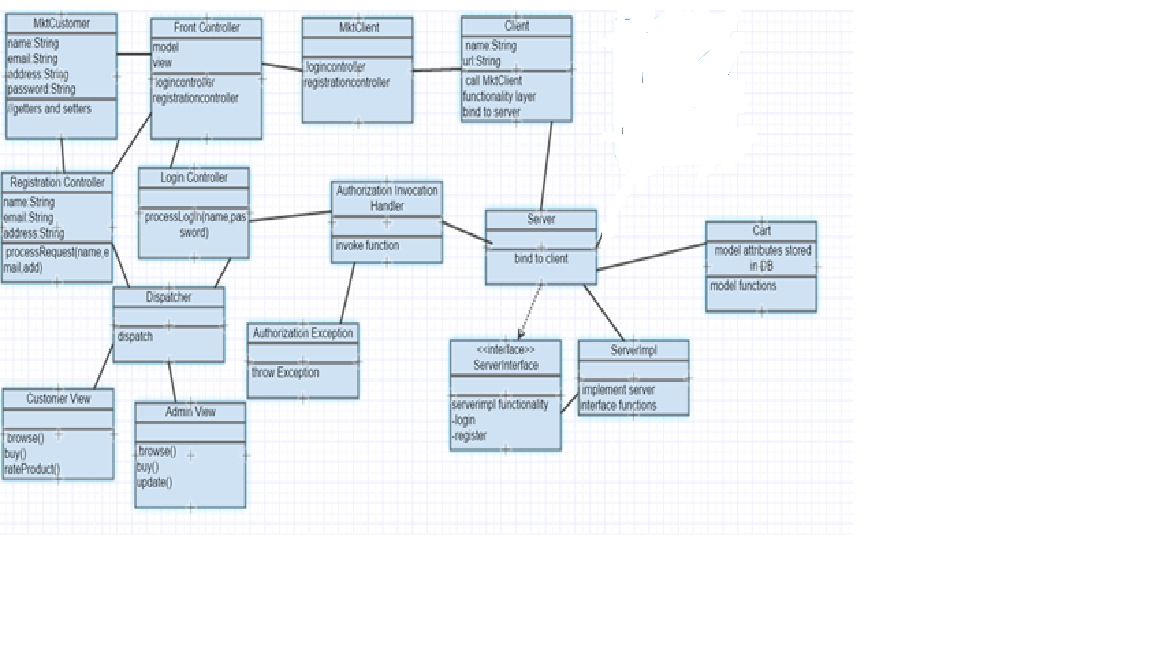
Purchasing Items

In this document for the code provided, the focus is on extending framework using Model View Controller design pattern and Java RMI. We specifically focus on the login functionality in this assignment using Front controller, Command pattern and Abstract Factory pattern.

**DOMAIN MODEL:** A domain model is a representation of the high level functionalities of the system.

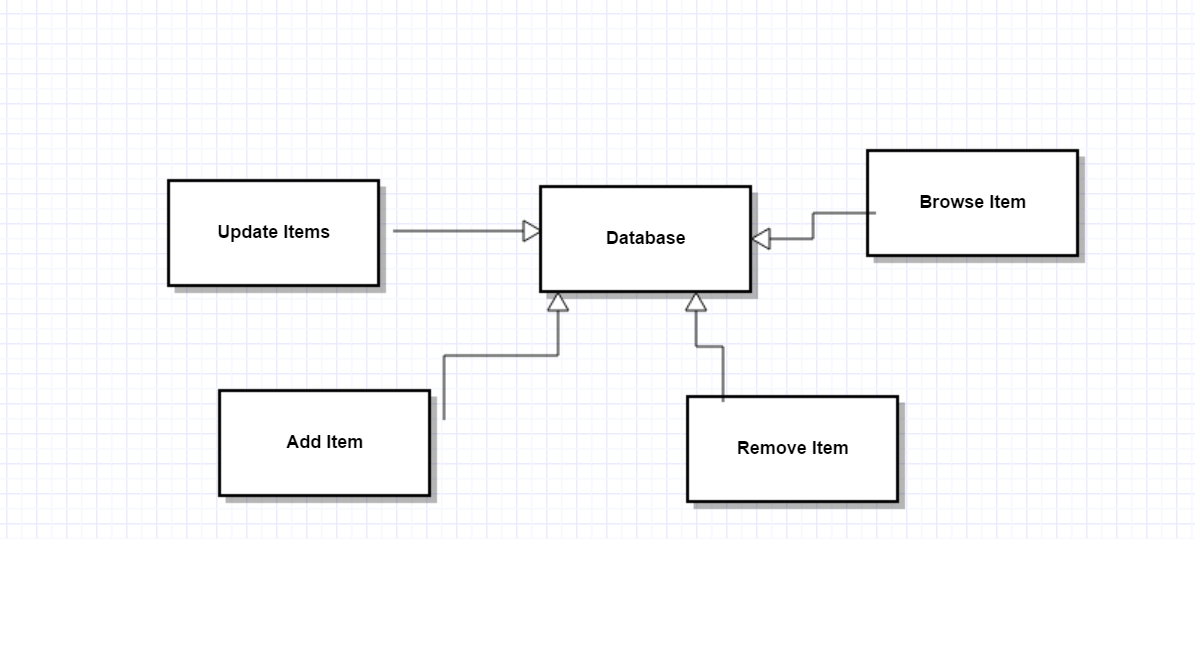
In this assignment, the online market place provides the services of login, registration, browsing, adding, removing, purchasing.

**CLASS DIAGRAM:** Class diagram is a useful representation of attributes and behavior of the system. It is very useful because it can be directly mapped with object oriented language.

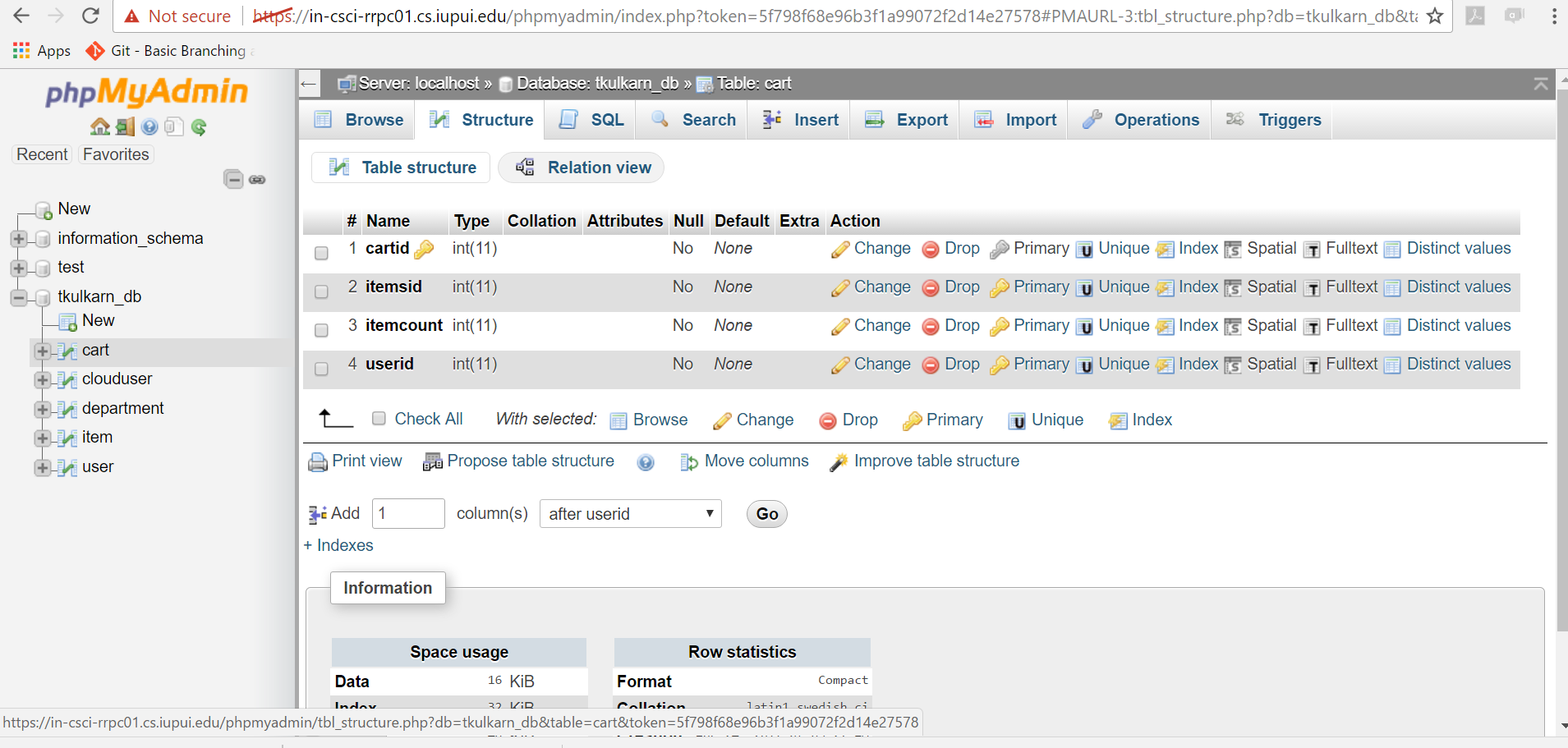


The

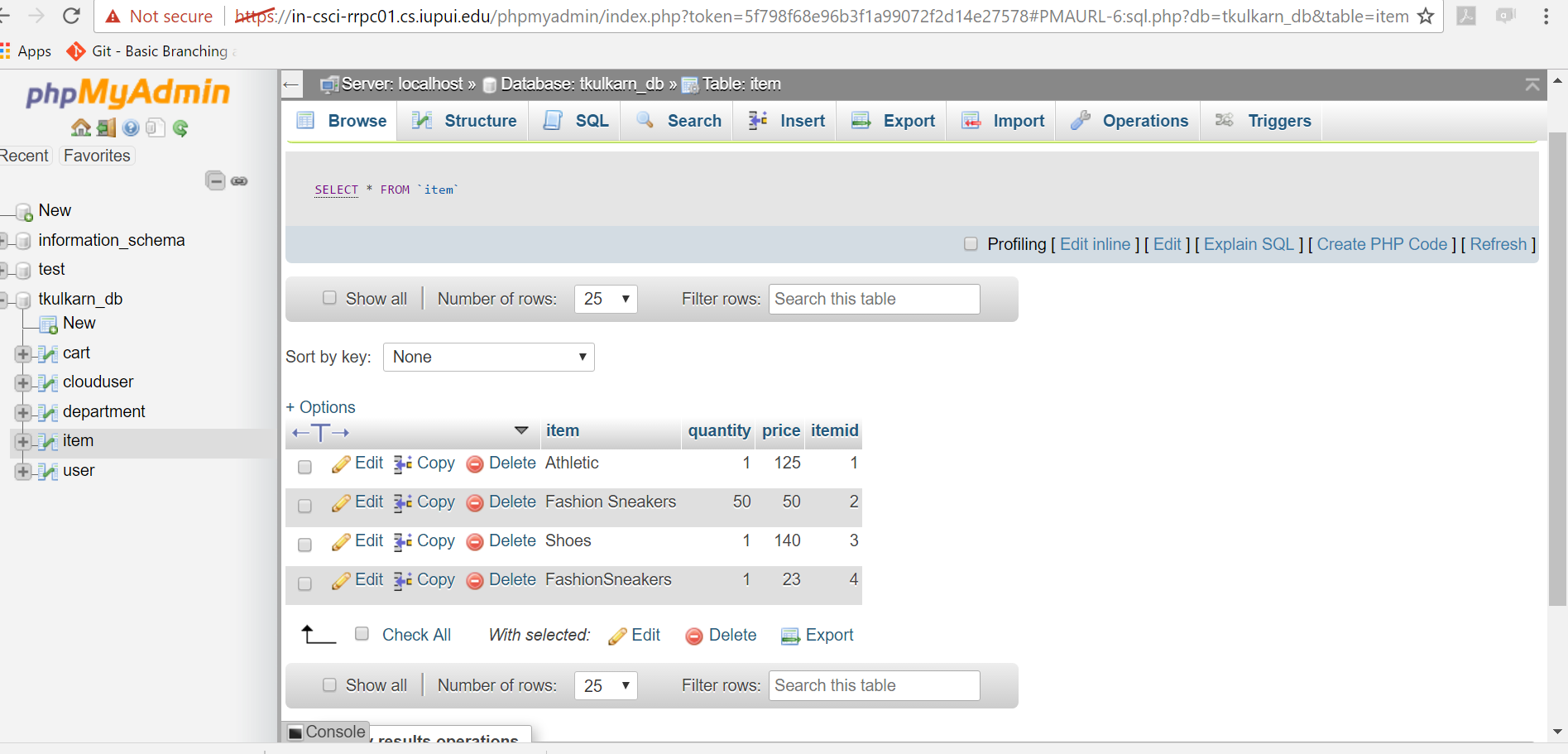
Interaction with the database:



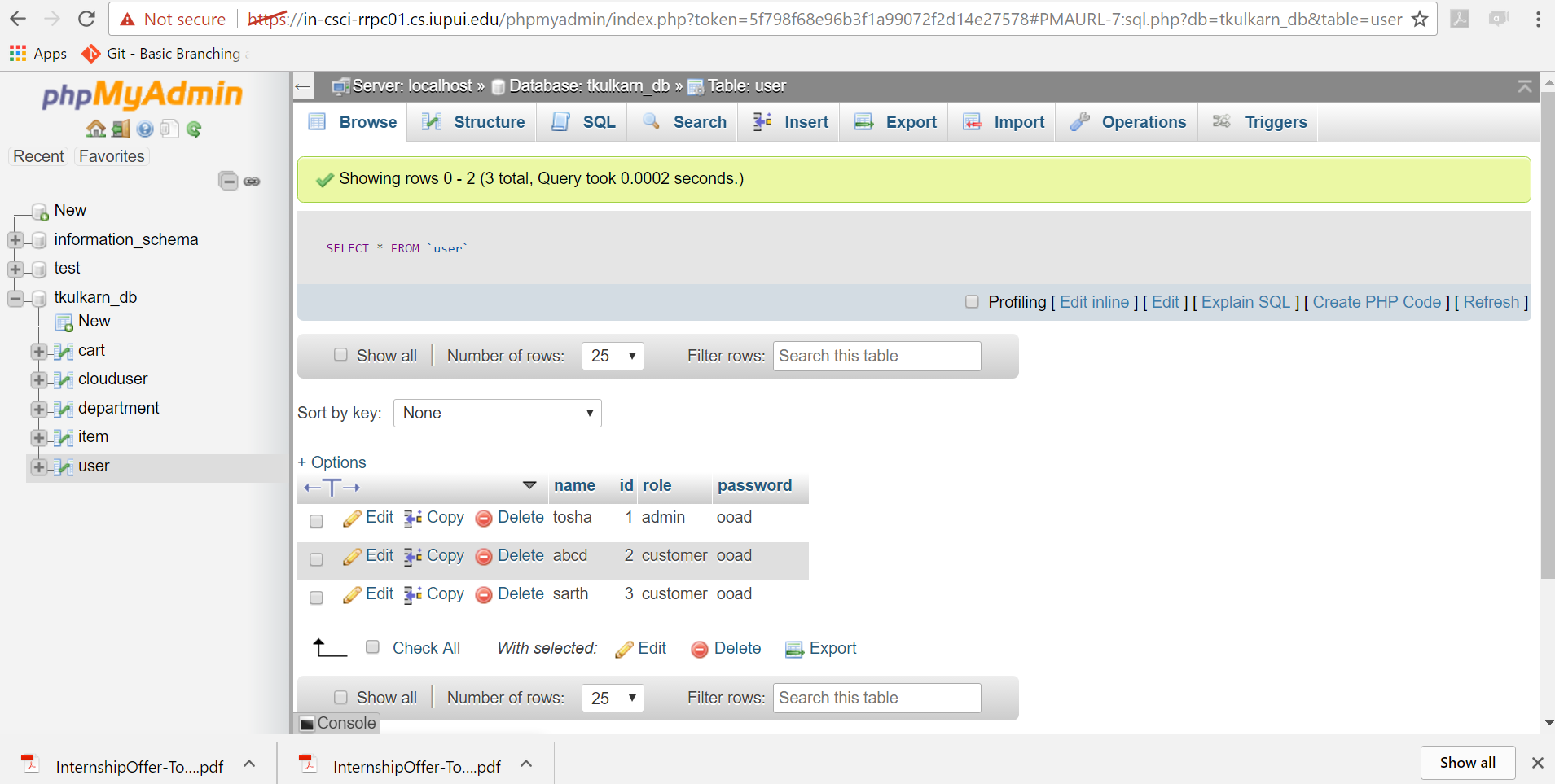
Cart table:



Item table:



User Table:



**DESIGN PATTERN:** The architectural design pattern used is model-view-controller where the view deals with visibility to the client, controller is the transfer logic and model deals with business data.

The data flow goes from view to the controller. The controller then accesses the model data and the behavior and returns the result to the controller. The controller then passes the results to the view. Apart from MVC, Front controller, Command Pattern ,Authorization pattern, Abstract Factory Pattern, Reflection and proxy pattern for various functionality.

The Server implements the interface registration which is defined in the interface Market.

**JAVA RMI:** Java RMI helps invoke remote methods. In this assignment, Market, Server and MktCustomer are components of the Java RMI implementation. Market is the interface which calls the remote function registration**.** Multithreading is an important feature of RMI which helps processing multiple requests.

**FRONT CONTROLLER:** Front Controller is the single point handler of all the requests. It serves as the entry point of the requests. The FrontController redirects the tasks to various controllers based on the action.The other controllers to which the request are delegated are Registration Controller and Login Controller

**COMMAND PATTERN:** Command pattern falls under behavioural category.In this the invoker gets the request as object and according to the command an action is executed.Like in this assignment, the end user just chooses to login and appropriately the actions are taken, and its checked if the user is admin or customer.

**ABSTRACT FACTORY PATTERN:** In this an interface is responsible for creating valid objects. The data flow is hidden from the end user. It is basically a creational design pattern. In this the client framework is not concerned about how closely the objects in the families are configured or deployed but only on the services that they provide.

**REFLECTION PATTERN:** It is used to support unanticipated changes in the system.It is used to describe code in the same system or itself.It exposes the methods in the system package.

**AUTHORIZATION PATTERN:** Role based access is used along with java annotations to ensure that only authentic users are allowed in the system.The AuthorzationInvocation handler is invoked everytime an annotation is found.

**PROXY PATTERN:** This is used to pass an instance as a real object.It is structural type and the class represents the functionality of another class.

The Client binds to the server and then calls the marketclient. The MarketClient displays the options to the user. The input from the user is transferred to the frontcontroller. The frontcontroller passes the control to the registration controller or login controller based on the choice.The registration controller accesses the data from the model class. The new user is then added and the result is passes to the frontcontroller.The parameters considered in this assignment for registration are only name,email id and address; which are all of String data type.The login controller after invocation accesses the server implementation function processlogin taking the username and password and accesses the model attributes and the comparison result is passed to the frontcontroller. The frontcontroller then knowing the role of the user through the dispatcher calls the appropriate view.In this assignment the values for login are hardcored, however eventually it will be verified against the database.The admin doesnot have the right to rate a product and the customer is not allowed to update an item.

**THREAD SAFE PATTERN:**

The Thread-safe Interface pattern ensures that intracomponent method calls avoid self-deadlock and minimize locking overhead.(Referred : <http://www.cs.wustl.edu/~schmidt/PDF/locking-patterns.pdf>)

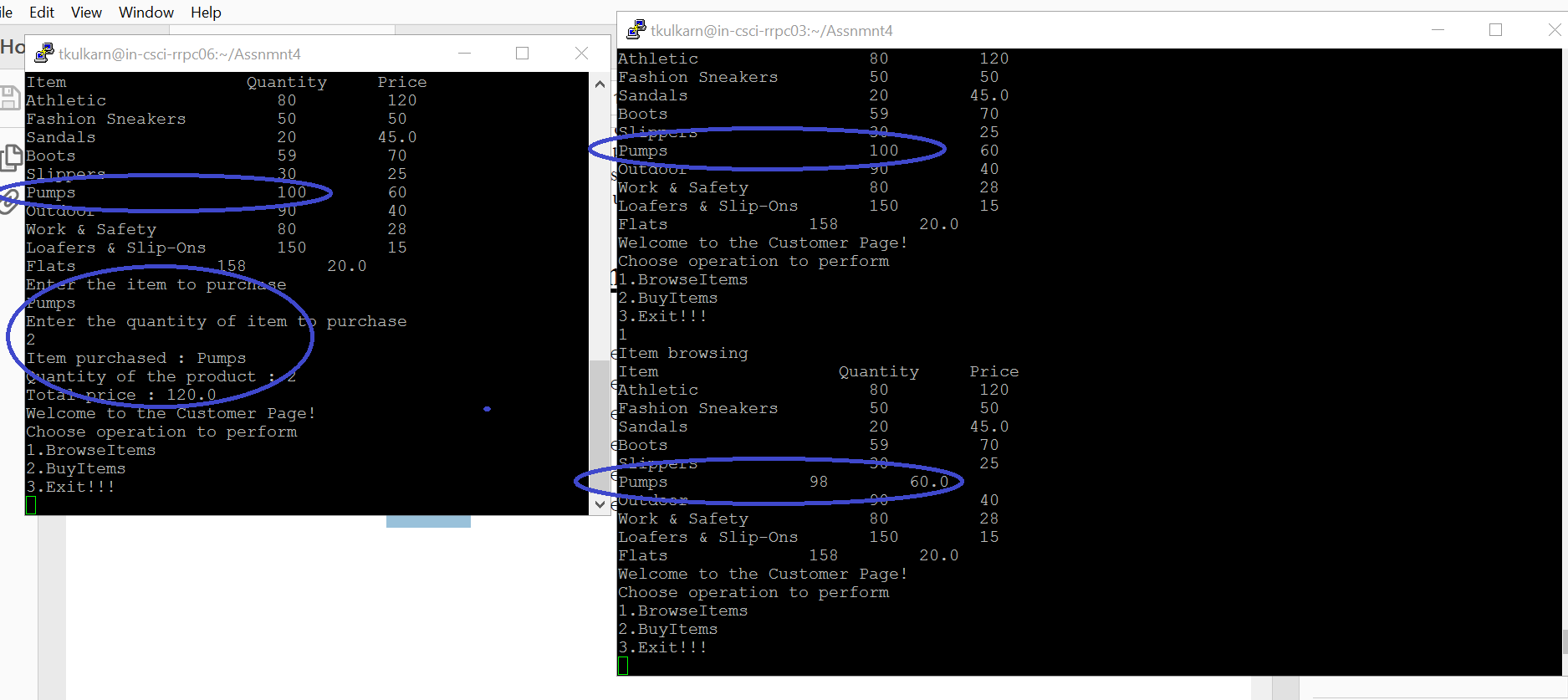
In the online marketplace the updation of items ,deletion of items, addition of items,purchase,checkout is synchronized with each other. Monitoring the objects created help in achieving this.

**JAVA RMI CONCURRENCY DISCUSSION:**

Theability of Java to run several parts of a program in parallel is called concurrency. A program achieves high performance and throughput by utilizing unused parts of underlying Operating Systems. Java RMI achieves concurrency using the mechanism of underlying OS. Thread Scheduling is done to decide how to share the processor resource among threads. However, the threading logic is supplied by the JVM which also contains the thread scheduler. In Windows, the threads are time sliced whereas in Linux based system the higher priority thread is run till completion.

JDK releases significantly onwards the release of JDK 1.5 includes multiple concurrency utilities.Executor,mutex,semaphore,latches,ForkJoinPool,Phaser,streams facility, lambda expressions are few examples of how Java evolved for supporting multiple threads.

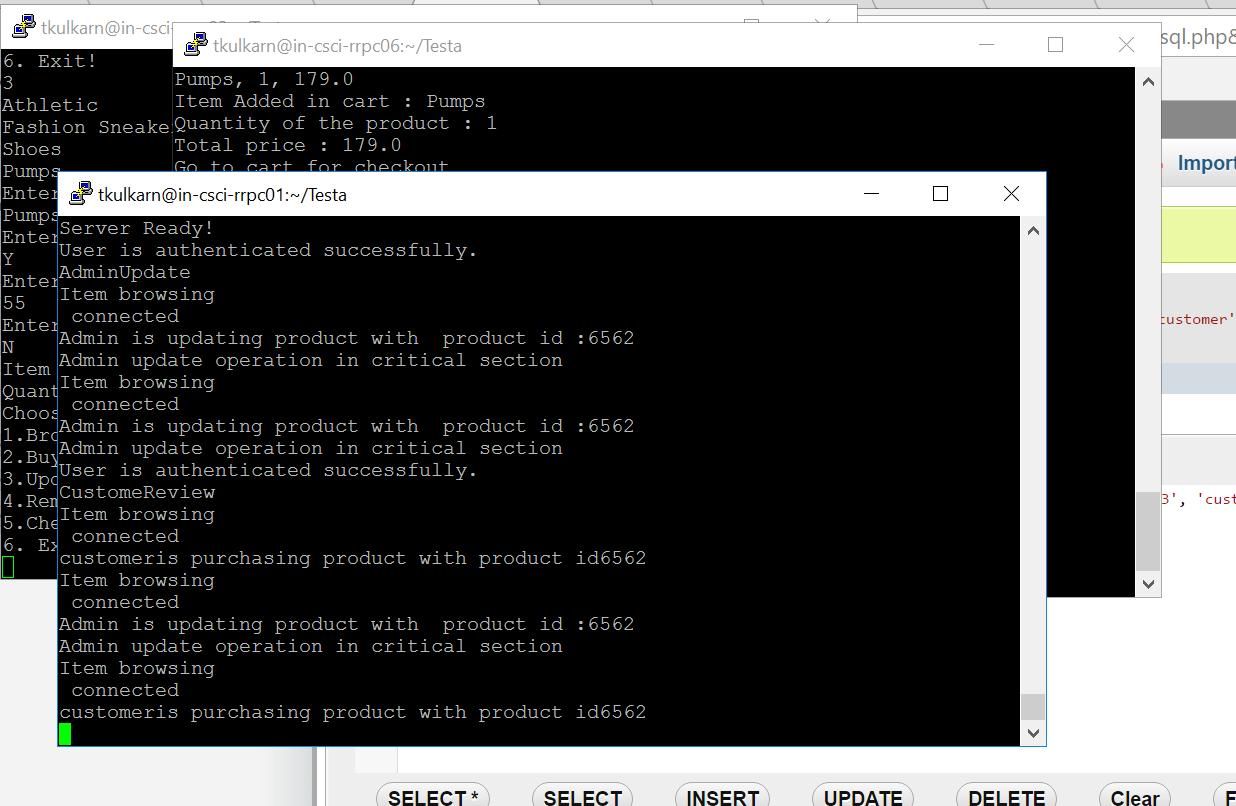
This is essentially to achieve better resource utilization and produce more responsive programs. In this assignment, whenever there is any update or addition to the marketplace done by one client on the server, it is simultaneously reflected to the other client as well accessing the same functionality.



Consider the example where client rrpc03 and rrpc06 are both using the server. In a case when both the clients first access the browse functionality, they see the same store products.

Now client rrpc06 buys the item Pumps in quantity of 2.In the scenario when client rrpc03 tries to browse and buy the product he sees the updated and valid online marketplace with Pumps item available in 98 quantity, thereby reflecting the concurrency.

**JAVA SYNCRONIZATION DISCUSSION:**



Java Synchronization is implemented in all the basic functionalities.

Consider the scenario:

One customer is browsing an item to buy the product. And the admin is trying to update the same product. The product id is 6562.

The thread safe interface ensures that the product is updated first by the admin. The change is then reflected to the customer also.

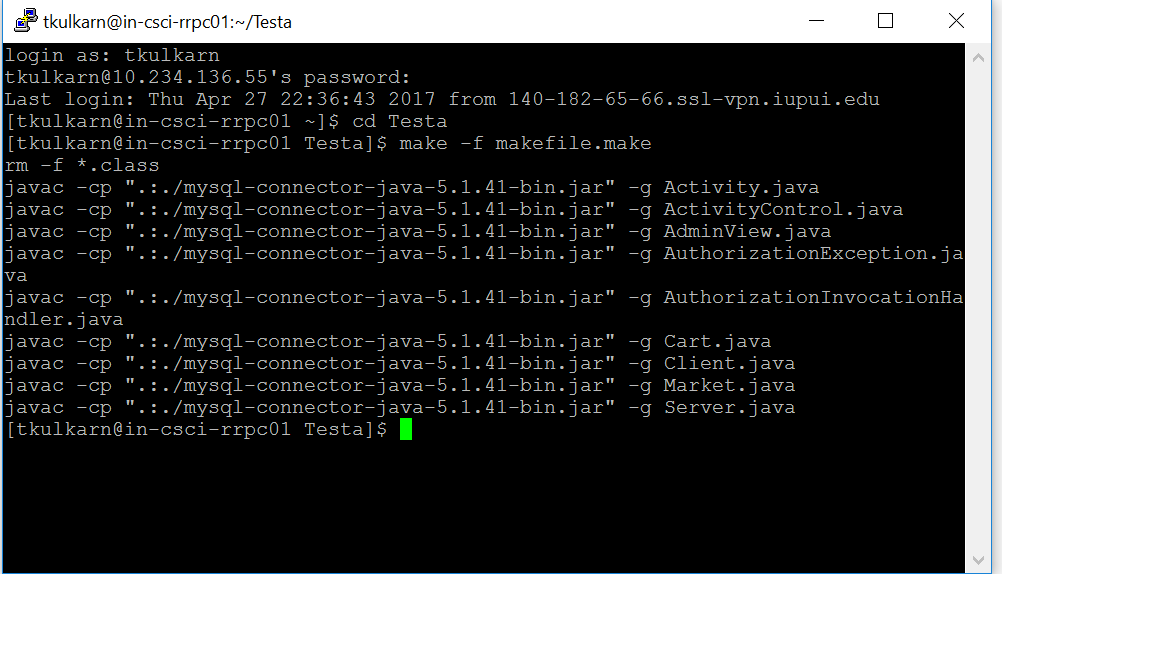
So synchronization makes sure that the product update which is critical is done first and then the customer can buy it.

**GITHUB+REPORT DISCUSSION:**

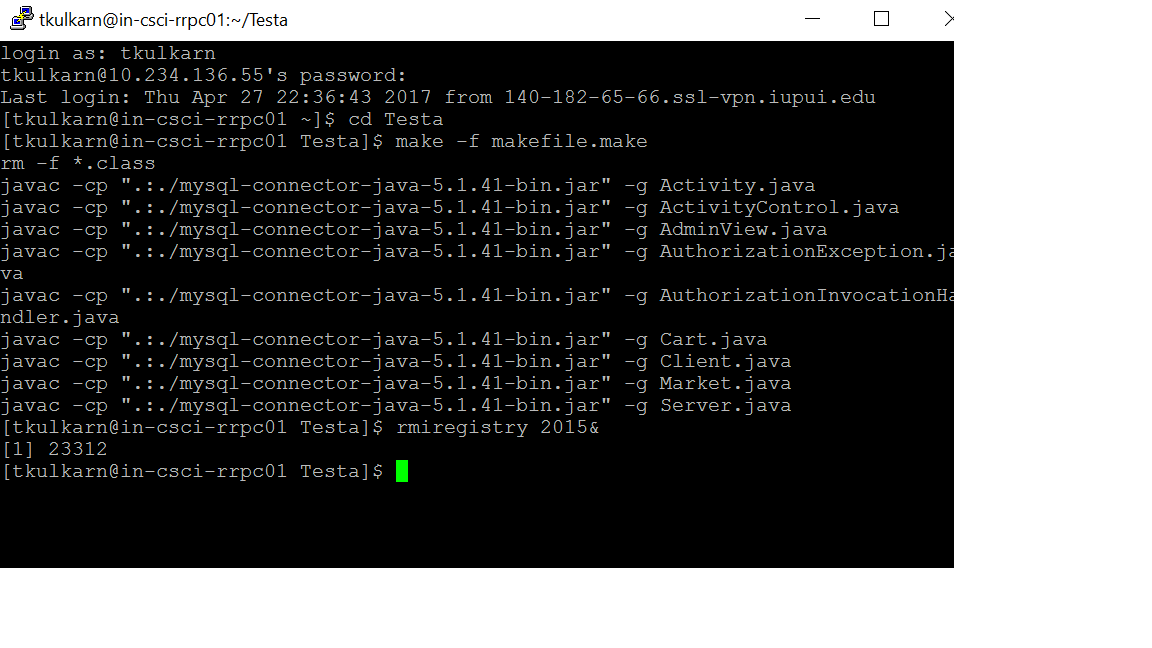
In the previous assignment, I was asked to provide useful comments in each file.I have included comments in this code.

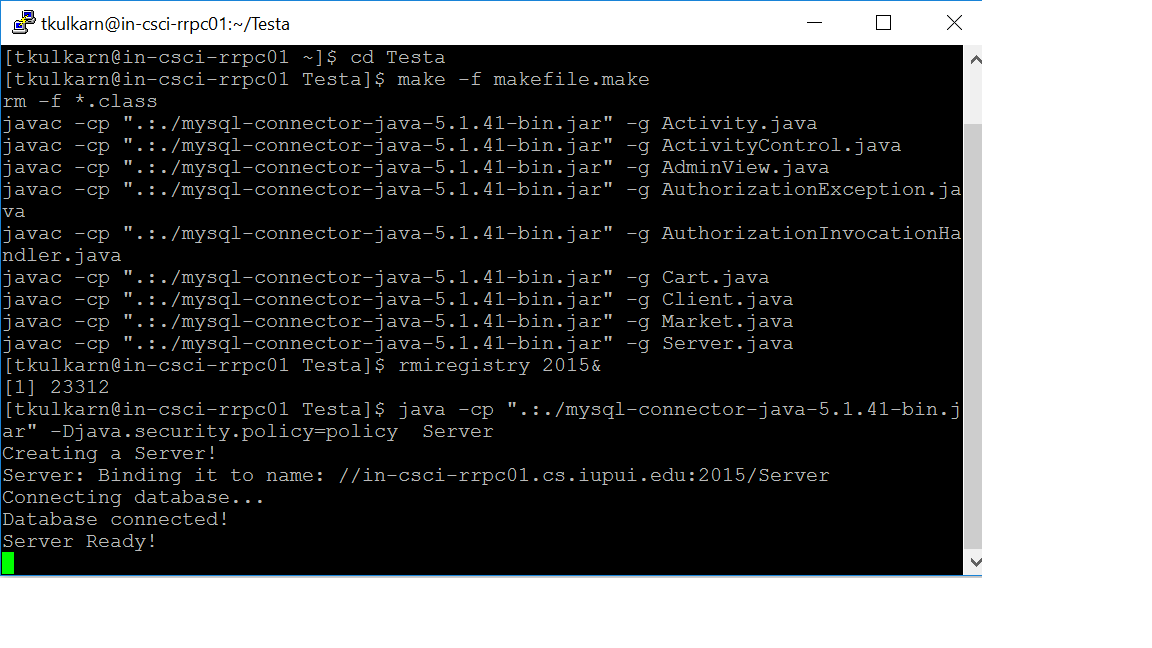
**RESULTS:**

1. Compiling all the java files using the makefile



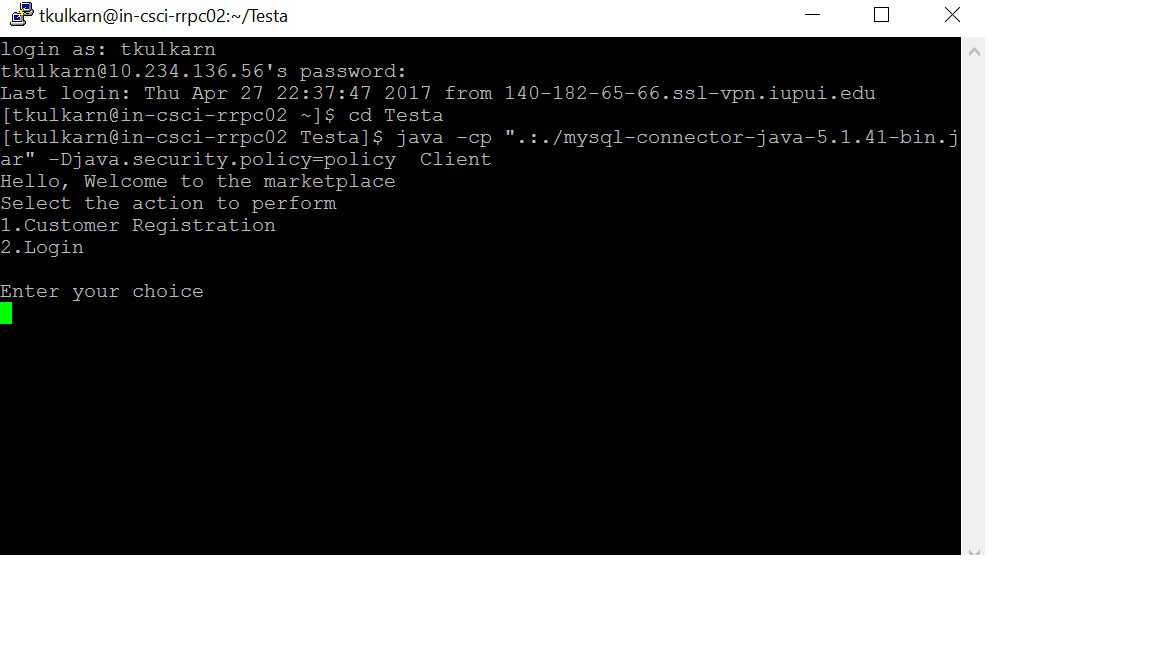
1. Starting the server



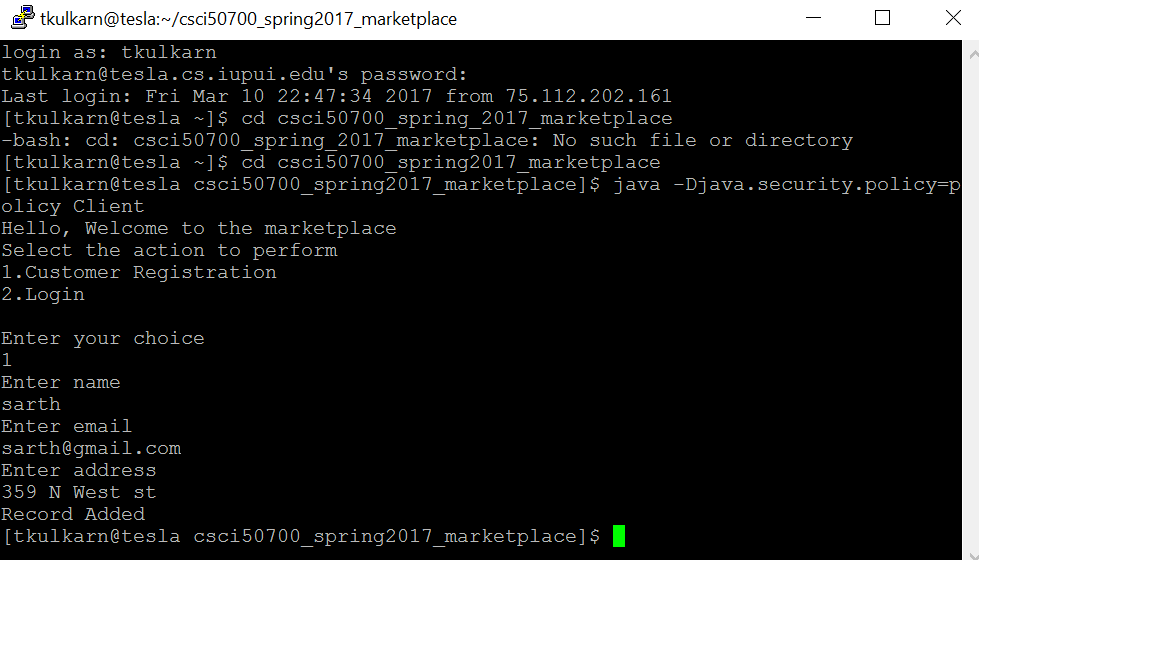


The port used is 2015

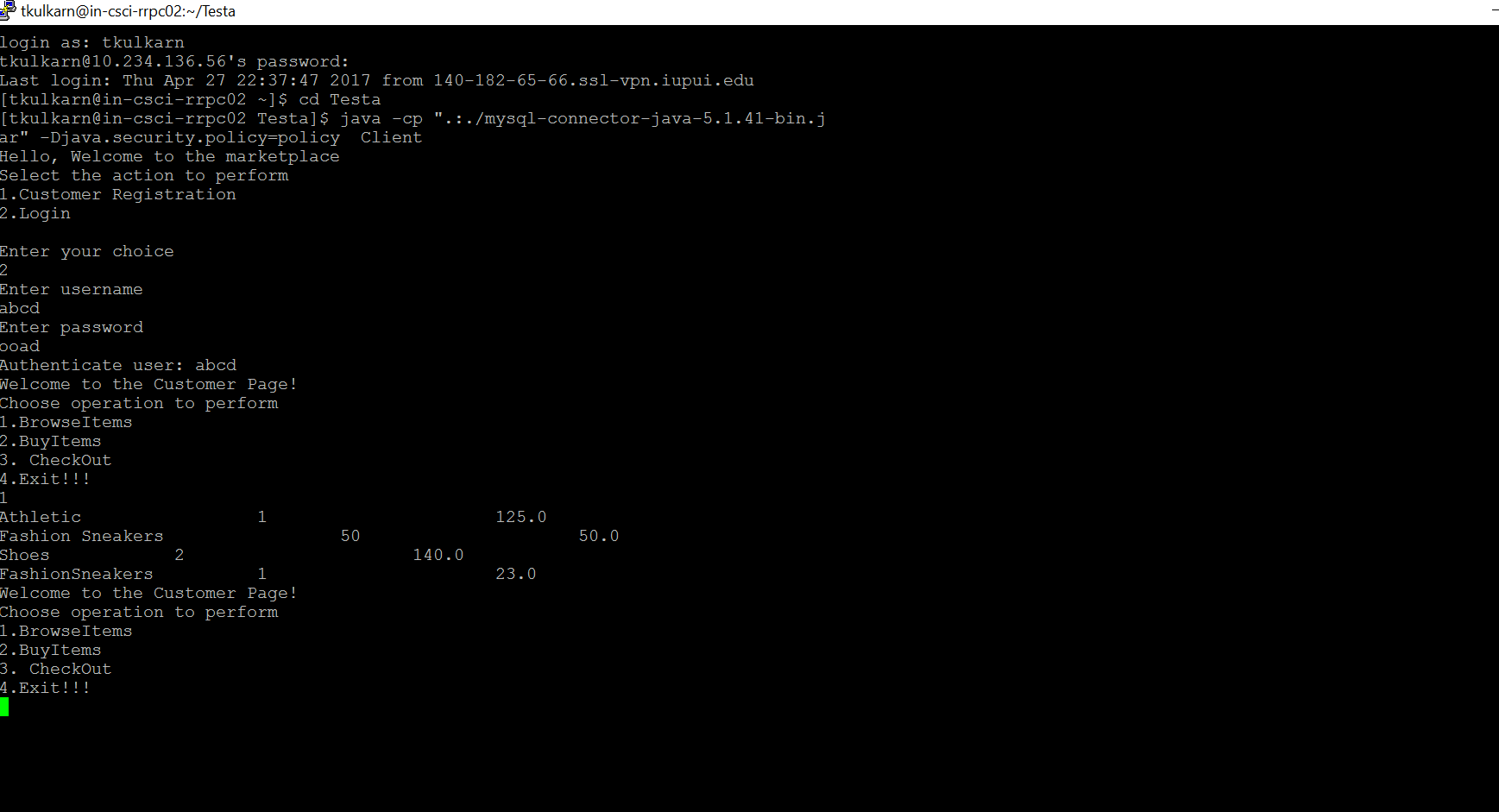
3.Starting another instance of putty for running the client



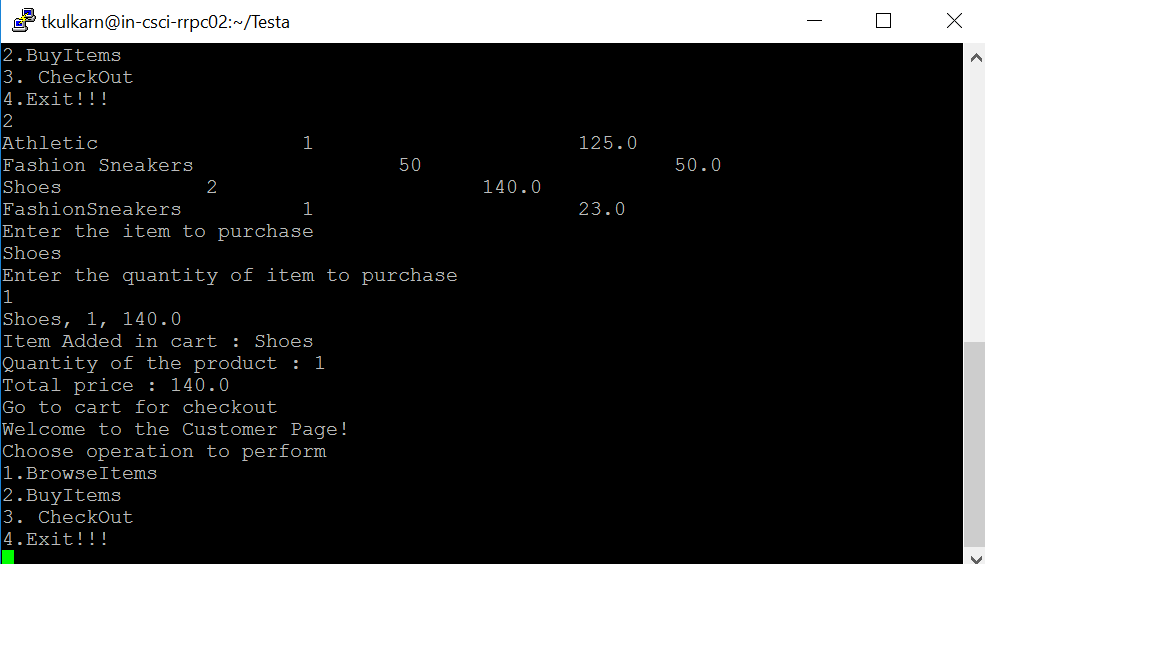
The option 1 is registration which adds a new user.



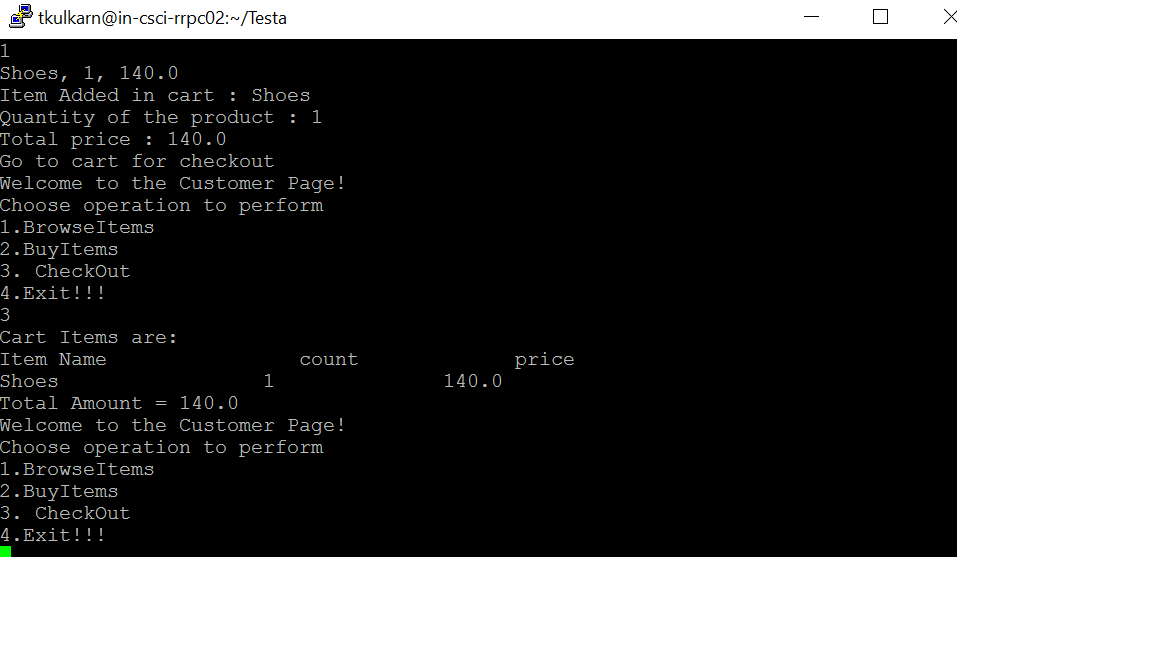
The option 2 is login. Consider customer login and browse function



Consider customer buy



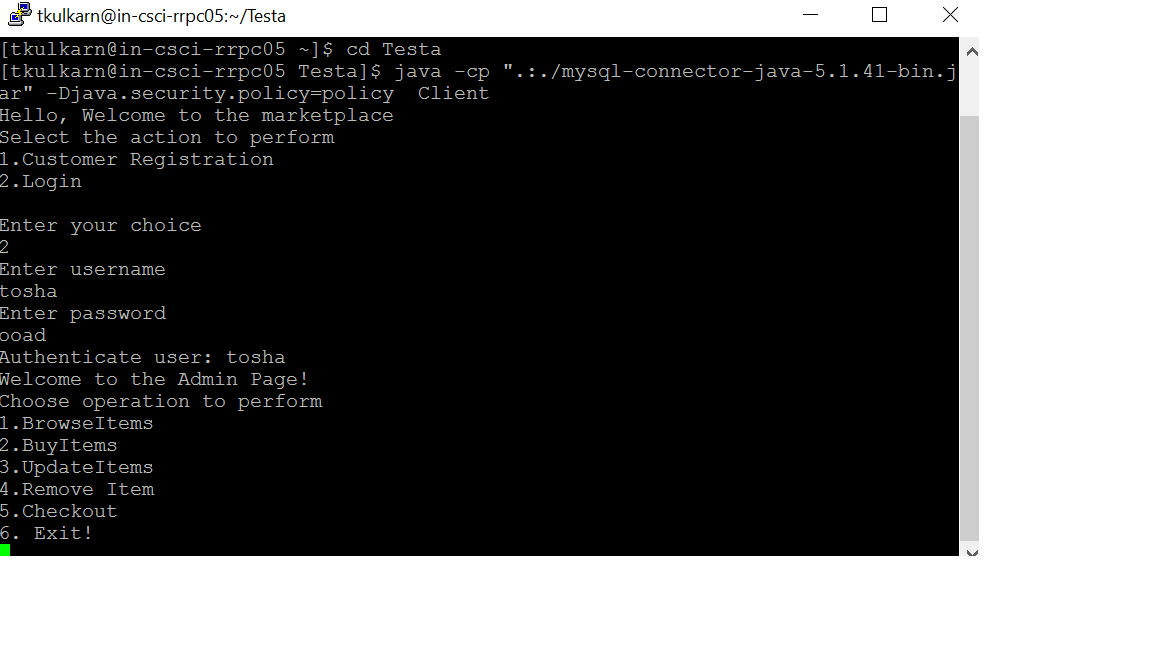
Customer checkout:



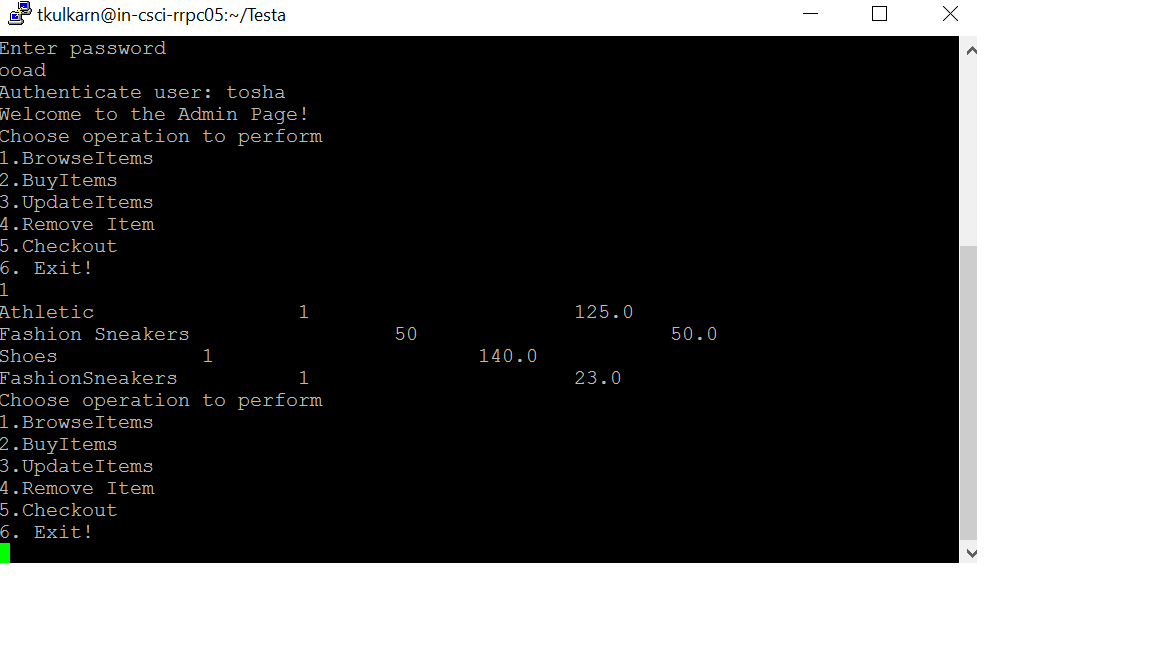
Cart empty after checkout:



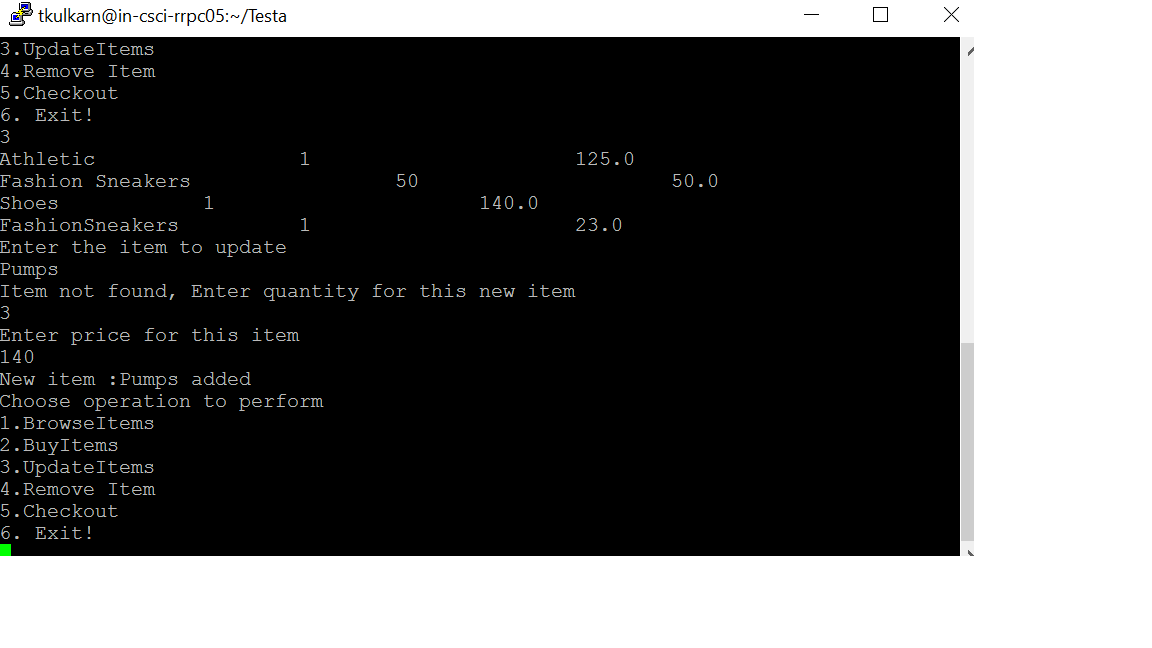
Admin Login

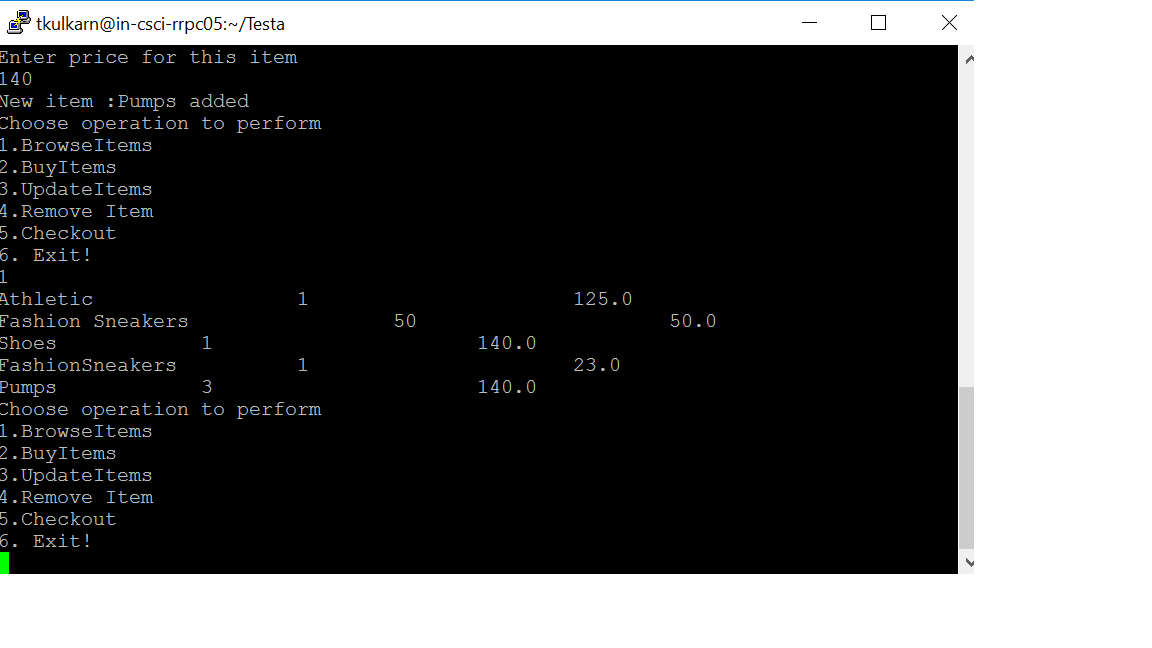


Admin Browse

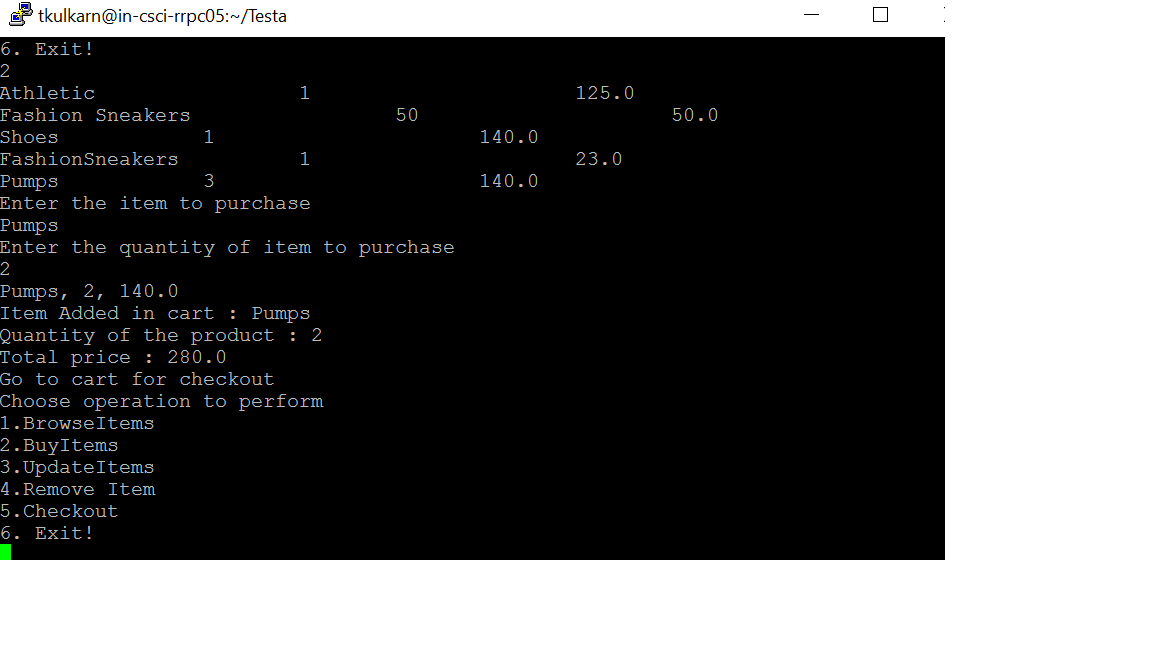


Add Item :Admin

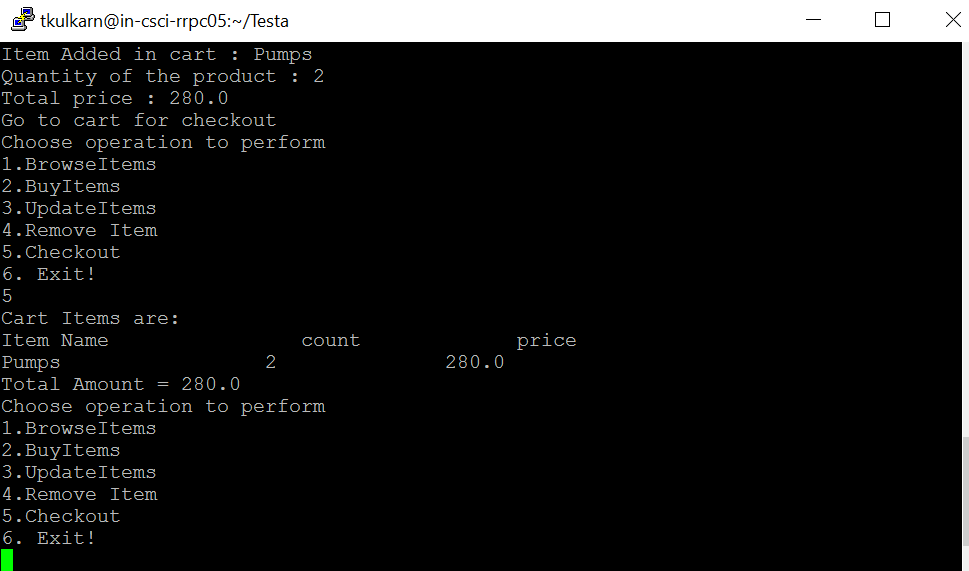
Item added:



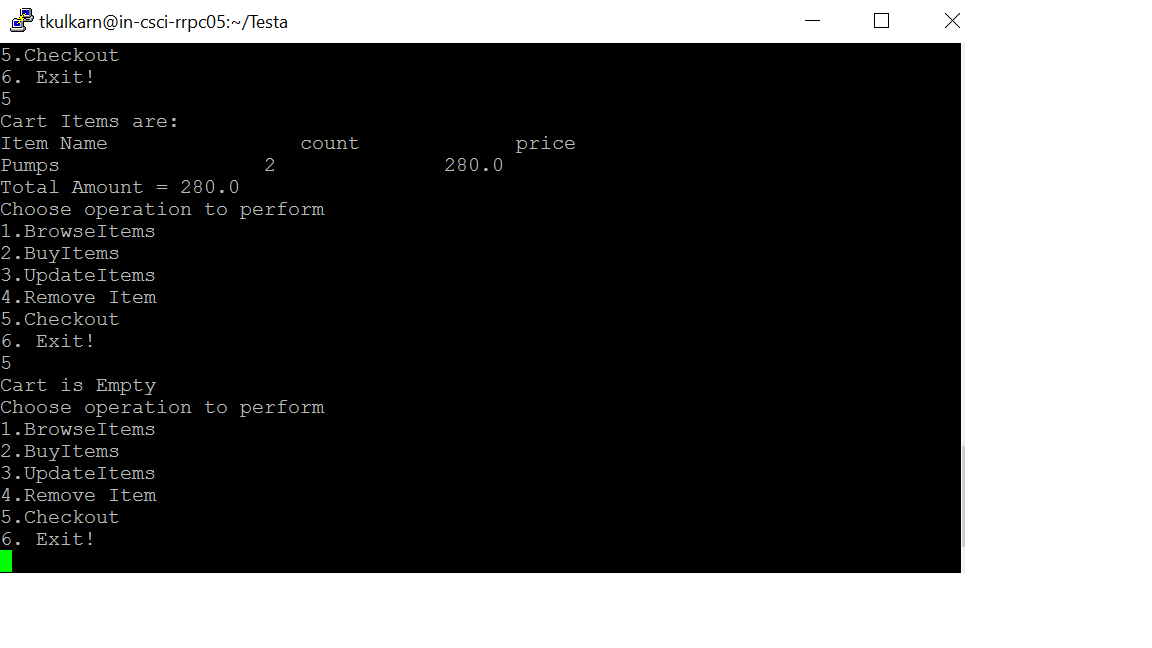
Admin Buy



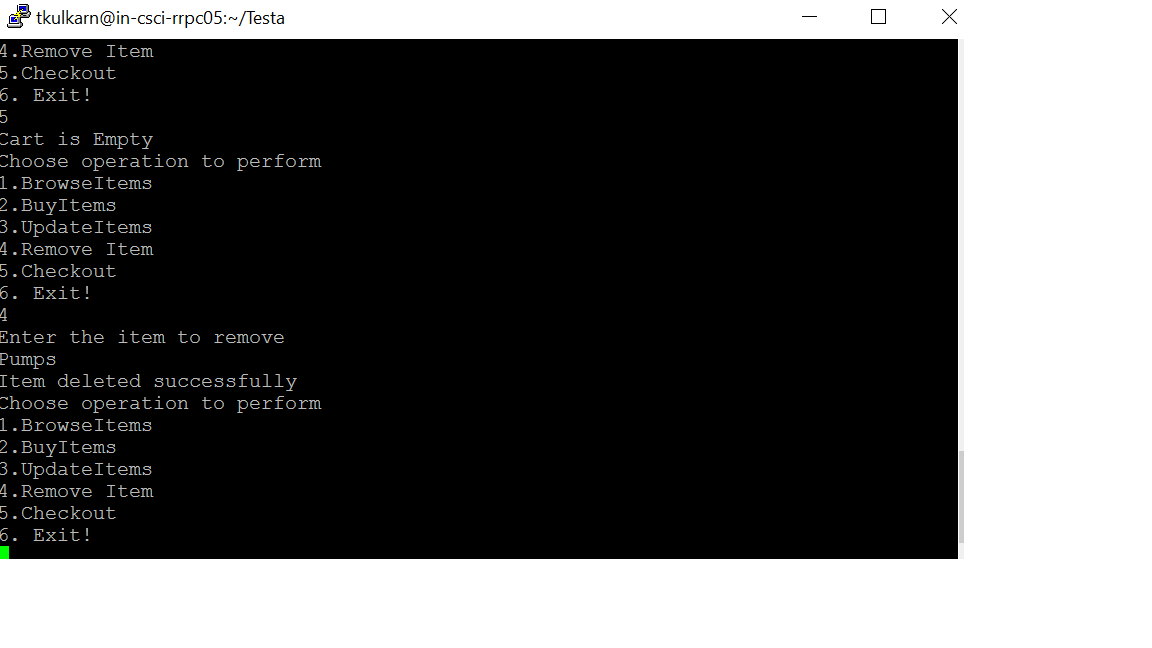
Admin Checkout:



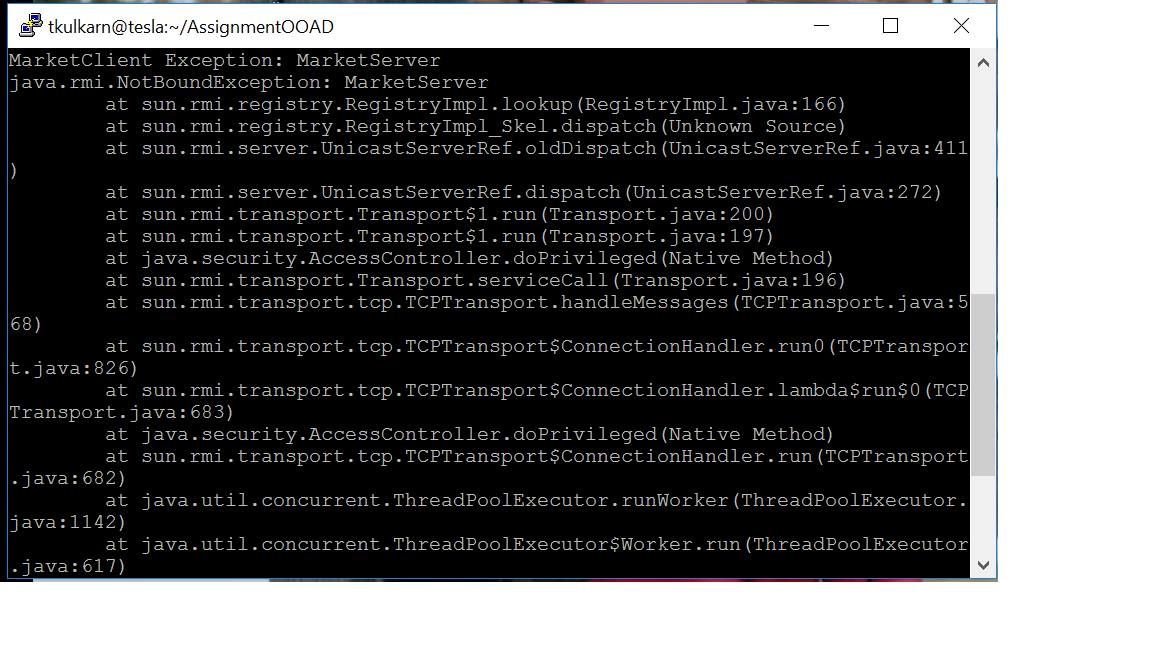
Admin Cart empty after Checkout:



Admin Delete Item



**POSSIBLE ERRORS:**When the client application is started without the server



When the server is attempted to start without starting the rmiregistry

