Object Oriented Design and Programming

CSCI 50700 Assignment-6

Love Modi
Department of Computer Information and Science
Indiana University Purdue University Indianapolis, USA

TABLE OF CONTENTS

1 Introduction

The client desires an online marketplace where they can sell goods (and possibly services) to customers geographically dispersed around the world. Think Amazon but on a smaller scale and budget. Their desire is to have a system that is constructed in a portable language (Java) and makes use of their existing network. The system itself should present a view for the customer to interact with as well as a view for the employees or administrators of the company to interface with. For the customer there is a need for them to be able to browse available products – this should present the customer with the type, description and price of the item with the options to add to their shopping cart. If the customer attempts to add a quantity of the item more than the current supply the system should prevent the customer from adding these and prompt them with a message on the availability of the item. The customer should be able to also purchase their items from the shopping cart. This shopping cart should maintain state and be persistent through interactions with the application. The administrators should be able to update an item's description within the system, update its price, and update its quantity. The administrator should also be able to remove items from the system if so desired. Administrators should be able to add other administrators as well as add/remove customer accounts. On the other hand, a customer should be able to initially register for their account by themselves. The system should handle any faults or unexpected scenarios gracefully.

1.1 Requirements

Below are the requirements, segregated from client's application description

- Separate views for customer and administrator, so separate interface for customer and admin
- Customer should be able to browse available products
    - type, description and price of the item
    - the options to add to their shopping cart
- If the customer attempts to add a quantity of the item more than the current supply the system should prevent the customer from adding these and prompt them with a message on the availability of the item.
- The customer should be able to also purchase their items from the shopping cart
- Shopping cart should have persistent storage
- The administrators should be able to update an item's description within the system, update its price, and update its quantity
- The administrator should also be able to remove items from the system if so desired.
- Administrators should be able to add other administrators as well as add/remove customer accounts
- administrators cannot register for accounts.
- customer should be able to initially register for their account by themselves.
- Administrator cannot purchase items in that role
- The system should handle any faults or unexpected scenarios gracefully

2 Assignment #1 Overview

A conscious decision to use the Model View Control (MVC) architecture for this application to provide separation of concerns and apply the layers design pattern. This helped in developing a low coupled system. Separating the page views from the method implementation helped in keeping the client-side light weight. I have kept view on client side and model on server side and controllers on both client and server sides (marketClientController and marketServerController) so that RMI could be absorbed within the MVC architecture. Market interface has been implemented for the RMI. This separation of concerns ensures that changes in one part of the system do not need changes beyond that subsystem.

3 Assignment #2 Overview

Front Controller pattern was implemented in assignment# 2. It is the centralized point of entry for all the requests from the client. A dispatcher class was implemented to fetch the required view for the client. All the requests from the views are configured to reach the client controller only through the front controller. Front Controller becomes the pipe line between the client and rest of the system. This has given a great way to control and navigate requests regarding different views.

4 Assignment #3 Overview

Role Based Access Control (RBAC) was the crux for the assignment# 3. RBAC guarantees that a method will be accessed by the user (customer/admin) who has the permission to access that method. This provides security against any malicious attacks trying to access methods which they don't have access to. Authorization Invocation handler is used to get this implemented. It checks whether the user role allows access to a specific method.

5 Assignment #4 Overview

Concurrency was the focus for assignment# 4. Marketplace application is accessed by multiple clients on a single server. This removes the scenarios when dirty data gives the user incorrect information when two or more threads are working concurrently to modify data. To ensure that it is important to implement synchronization patterns. In my application as per my design I have used thread-safe synchronization and monitor object synchronization. There was no need for future pattern in my application, so it was not implemented.

6 Assignment #5 Overview

Synchronization patterns were addressed in assignment# 5. 5 important synchronization related patterns were discussed. I tried to check the feasibility and requirement of all them within the scope of my project. I could not use Future pattern but applied Monitor Object, Guarded Suspension, Scoped Locking, Thread-Safe Interface via the use of Synchronization keyword. In-depth discussion of how these patterns work and applied in the project was discussed within assignment #6.

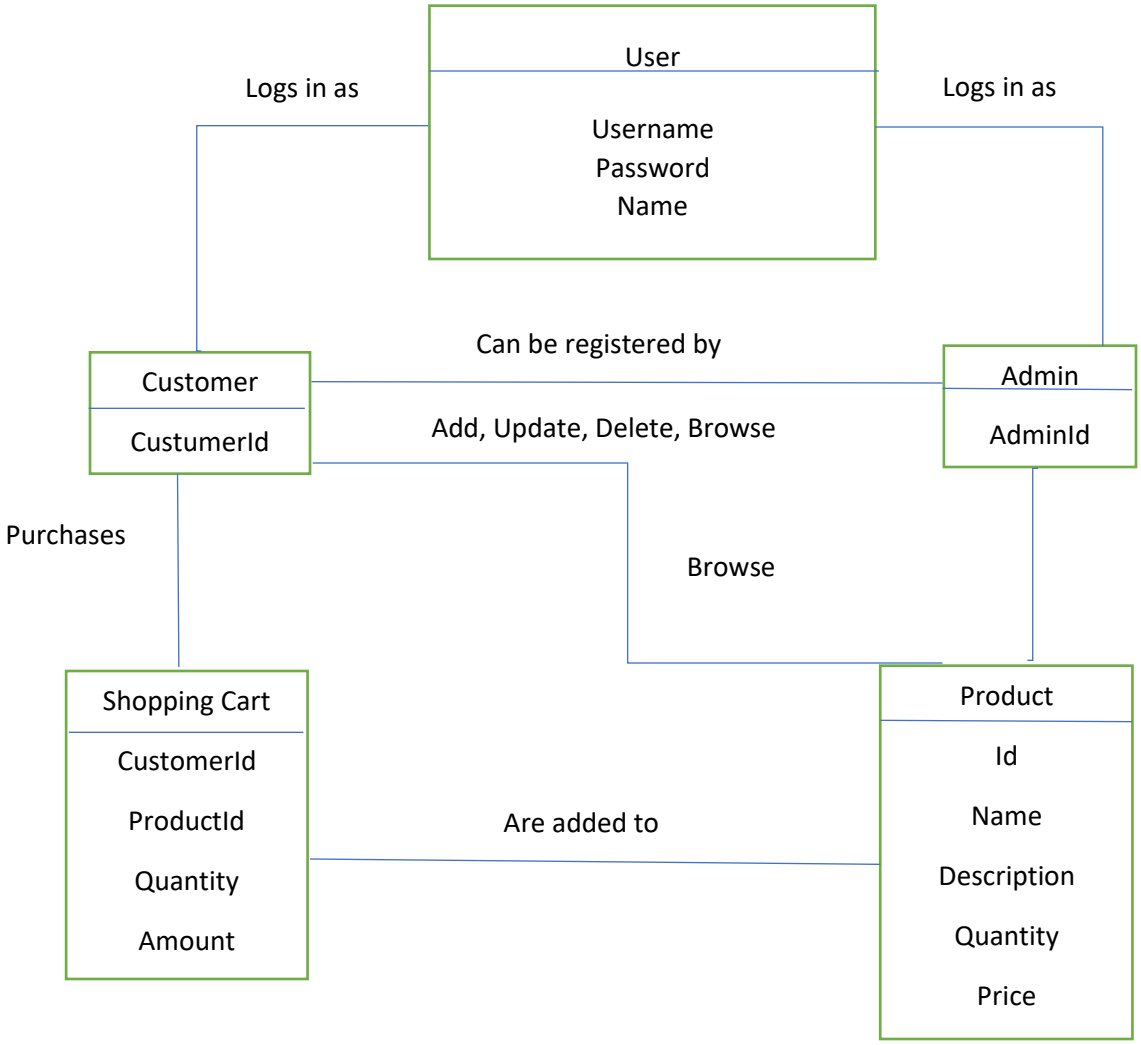
7 Database Access Layer Pattern

To shield the MarketModel from the database, a data mapper layer has been placed between the two layers. The idea is to decouple the persistent storage space from the application logic (within the Market Model). This is more like implementing another MVC application where the database acts as the model, market model acts as the view and the data mapper layer acts as the model. Hence by introducing a layer of separation, we are efficiently being able to access our database and handle distributed operations due to the synchronization patterns added in the earlier assignment.

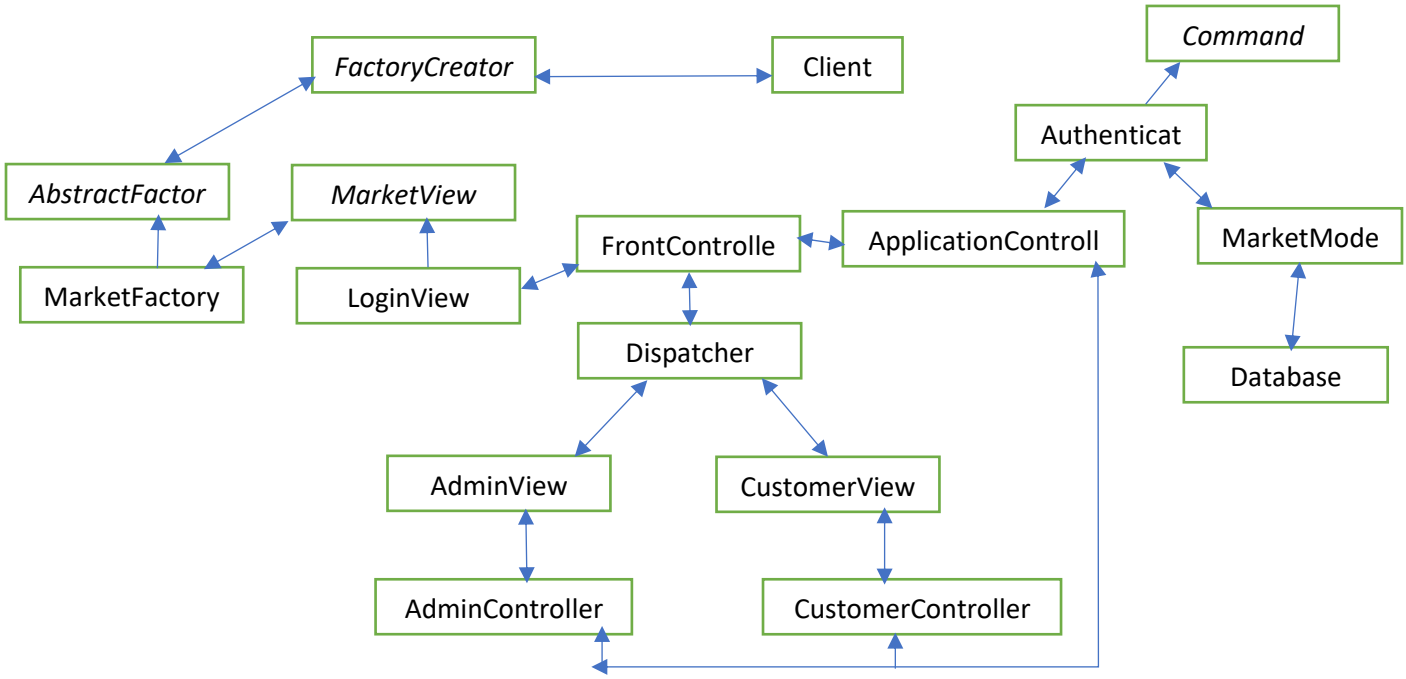
This method tries to address the problem of the disconnect between how to map objects and their relationships – inter object navigation and inheritance in such tables. Hence, we introduce this separate layer between application and database, so that mapping operations can be done in this layer and all the business logic can be provided in the applications end layer. For me this approach has helped in lowering the coupling and increasing the cohesion which has resulted in a smooth application.

8 Updated (Final) Documentation

- Domain Model:

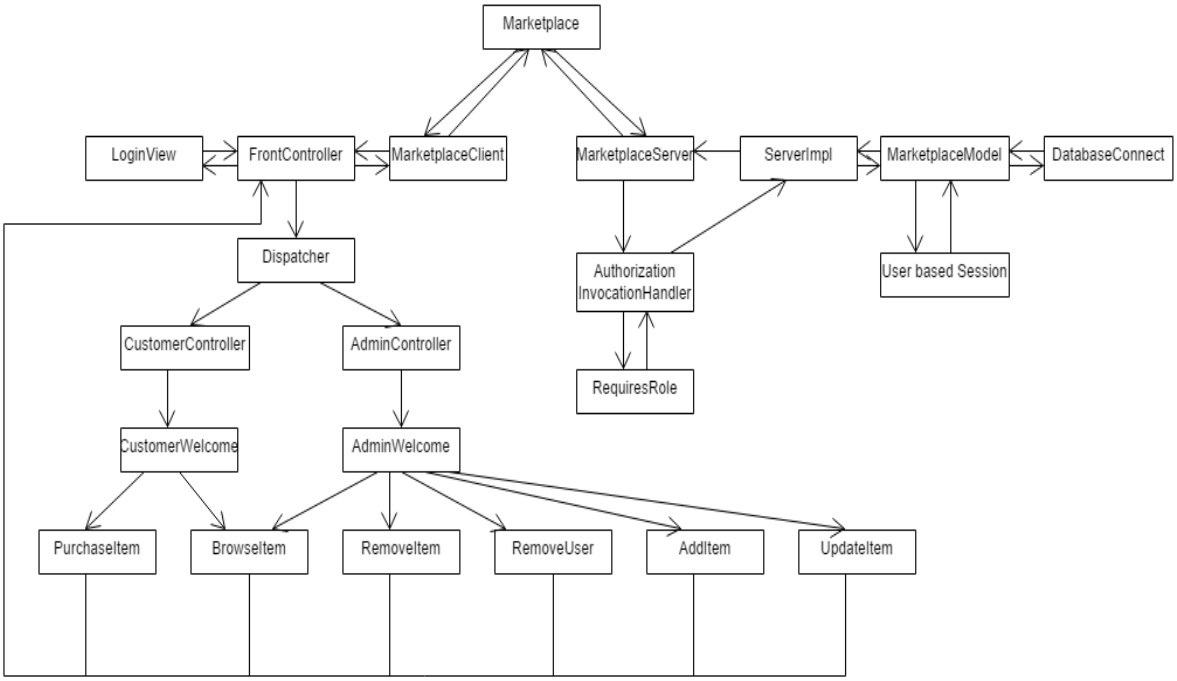


- Software Design



9 Complete Functionality Implementation

Below is a simplified software design model as of this moment:



All the required functionalities have now been implemented. The sample runs for all function can be found in the next section.

10 Sample runs highlighting full functionality

Please find below the snapshots to verify that all the functionalities are working:

Signup

-Customer

```
Welcom to Market App...
What would you like to do today?
Please enter your selection:
1. Signup as a new customer
2. Already a user? Signin
1

Please enter a First Name: Mona

Please enter a Last Name: Mona

Please enter a username: mona

Please enter a password: mona
User registered successfully!!

Please enter your Username and Password to sign in.

Please enter a username:
```

Edit Copy Delete lm	lm	lmodi	lmodi	admin	
Edit Copy Delete monica	s	mon	mon	customer	
Edit Copy Delete Mona	Mona	mona	mona	customer	
Edit Copy Delete Monu	Monu	monu	monu	customer	

Login:

-Admin

```
Please enter your Username and Password to sign in.

Please enter a username: admin

Please enter a password: admin
Admin Authenticated Successfully.
Welcome Admin, Below are the items on sale...
```

-Customer

```
Please enter your Username and Password to sign in.

Please enter a username: customer

Please enter a password: customer
Customer Authenticated Successfully.
Welcome Customer, Below are the items on sale...
```

Admin functionalities:

-Manage user accounts

- Add admin

```
1
Please make your selection:
1. Add Account
2. Remove Account
3. Exit
1

First Name: Bran

Last Name: Upp

username: bran

password: upp

Roletype (customer or admin): admin

 Add more users? Please enter 'y' or 'n': n

Users added successfully!!...
```

	firstname	lastname	username	password	roleType
Edit Copy Delete	admin	admin	admin	admin	admin
Edit Copy Delete	Bran	Upp	bran	upp	admin

- Add customer

```
First Name: Brice

Last Name: Ran

username: brice

password: brice

Roletype (customer or admin): customer

 Add more users? Please enter 'y' or 'n': n

Users added successfully!!...
```

- Remove customer

```
username: brice

 Delete more users? Please enter 'y' or 'n': n

Users deleted successfully!!...
```

-Browse Products

```
Please make your selection:
1. Manage Admin and Customers
2. Browse Products
3. Add Products
4. Update Products
5. Delete Products
6. Exit
2

Item id: 8
Name: Dell Simple
Description: Nice Laptop
Quantity: 9
Price per item: 900.0


Item id: 9
Name: VR Laptop
Description: VR Lap
Quantity: 1
Price per item: 900.0


Item id: 10
Name: Dell Laptop
Description: Cool Laptop
Quantity: 10
Price per item: 900.0
```

-Add Products

```
Please make your selection:
1. Manage Admin and Customers
2. Browse Products
3. Add Products
4. Update Products
5. Delete Products
6. Exit
3

Please enter the name of the product: Fly Laptop

Please enter the description of the product:Nice Product

Please enter the quantity of the product:10

Please enter the price of the product:950

 Add more products? Please enter 'y' or 'n': n
\Added products successfully...Browse available products\n
```

```
Item id: 9
Name: VR Laptop
Description: VR Lap
Quantity: 1
Price per item: 900.0


Item id: 10
Name: Dell Laptop
Description: Cool Laptop
Quantity: 10
Price per item: 900.0


Item id: 11
Name: Dell Laptop
Description:  Cool Laptop
Quantity: 58
Price per item: 800.0


Item id: 12
Name: ABC
Description: hghjg
Quantity: 6
Price per item: 600.0


Item id: 13
Name: Fly Laptop
Description: Nice Product
Quantity: 10
Price per item: 950.0
```

-Update Products

```
Quantity: 58
Price per item: 800.0

Item id: 12
Name: ABC
Description: hghjg
Quantity: 6
Price per item: 600.0

Please make your selection:
1. Manage Admin and Customers
2. Browse Products
3. Add Products
4. Update Products
5. Delete Products
6. Exit
4
Please enter the Id of the product you want to update: 10
Please enter the name of the product: Dell Laptop
Please enter the description of the product:Cool Laptop
Please enter the quantity of the product:10
Please enter the price of the product:900
 Update more products? Please enter 'y' or 'n': y
Please enter the Id of the product you want to update: 8
Please enter the name of the product: Dell Simple
Please enter the description of the product:Nice Laptop
Please enter the quantity of the product:9
Please enter the price of the product:900
 Update more products? Please enter 'y' or 'n': n
Update Successful...Browse available products
```

```
Please make your selection:
1. Manage Admin and Customers
2. Browse Products
3. Add Products
4. Update Products
5. Delete Products
6. Exit
2

Item id: 8
Name: Dell Simple
Description: Nice Laptop
Quantity: 9
Price per item: 900.0


Item id: 9
Name: VR Laptop
Description: VR Lap
Quantity: 1
Price per item: 900.0


Item id: 10
Name: Dell Laptop
Description: Cool Laptop
Quantity: 10
Price per item: 900.0
```

-Delete Products

```
Please make your selection:
1. Manage Admin and Customers
2. Browse Products
3. Add Products
4. Update Products
5. Delete Products
6. Exit
5

Please enter the Id of the product you want to delete: 12

 Delete more products? Please enter 'y' or 'n': n
Successfuly deleted Items...
```

```
Item id: 10
Name: Dell Laptop
Description: Cool Laptop
Quantity: 10
Price per item: 900.0


Item id: 11
Name: Dell Laptop
Description:  Cool Laptop
Quantity: 58
Price per item: 800.0


Item id: 13
Name: Fly Laptop
Description: Nice Product
Quantity: 10
Price per item: 950.0
```

Customer Functionalities

-Add item to cart

```
Please make your selection:
1. Add item to Cart
2. Purchase Item in your Cart
3. Browse Products
4. View Products in your Cart
5. Exit
1

Enter the product Id you want to add to cart: 8

Enter the Quantity you want to add: 2
Items in your Cart:

Item id: 8
Name: Dell Simple
Description: Nice Laptop
Quantity: 2
Price per item: 900.0
```

-Purchase Items in cart

```
Please make your selection:
1. Manage Admin and Customers
2. Browse Products
3. Add Products
4. Update Products
5. Delete Products
6. Exit
2

Item id: 8
Name: Dell Simple
Description: Nice Laptop
Quantity: 9
Price per item: 900.0


Item id: 9
Name: VR Laptop
Description: VR Lap
Quantity: 1
Price per item: 900.0


Item id: 10
Name: Dell Laptop
Description: Cool Laptop
Quantity: 10
Price per item: 900.0
```

```
Please make your selection:
1. Add item to Cart
2. Purchase Item in your Cart
3. Browse Products
4. View Products in your Cart
5. Exit
2
=======Your shopping cart Items have been purchased=========
Congrats your order has been shipped...
View more products to buy

Item id: 8
Name: Dell Simple
Description: Nice Laptop
Quantity: 7
Price per item: 900.0
```

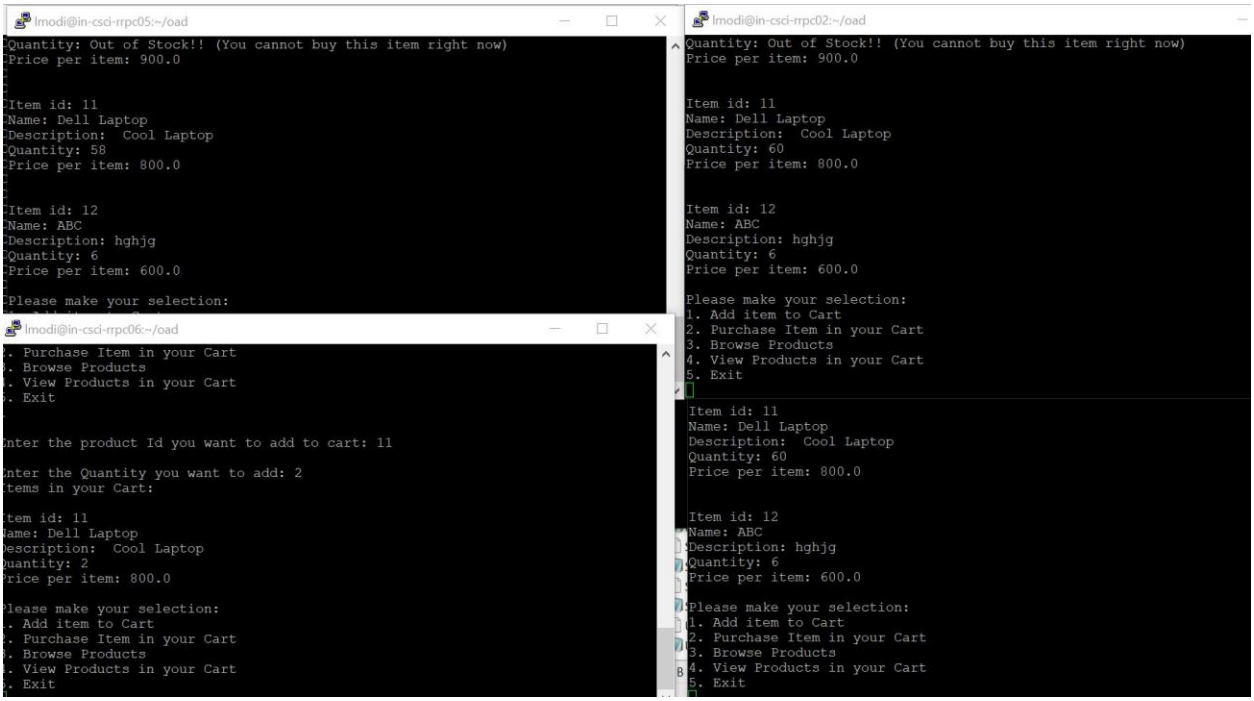
-Browse items

```
Please make your selection:
1. Add item to Cart
2. Purchase Item in your Cart
3. Browse Products
4. View Products in your Cart
5. Exit
3

Item id: 8
Name: Dell Simple
Description: Nice Laptop
Quantity: 7
Price per item: 900.0


Item id: 9
Name: VR Laptop
Description: VR Lap
Quantity: 1
Price per item: 900.0
```

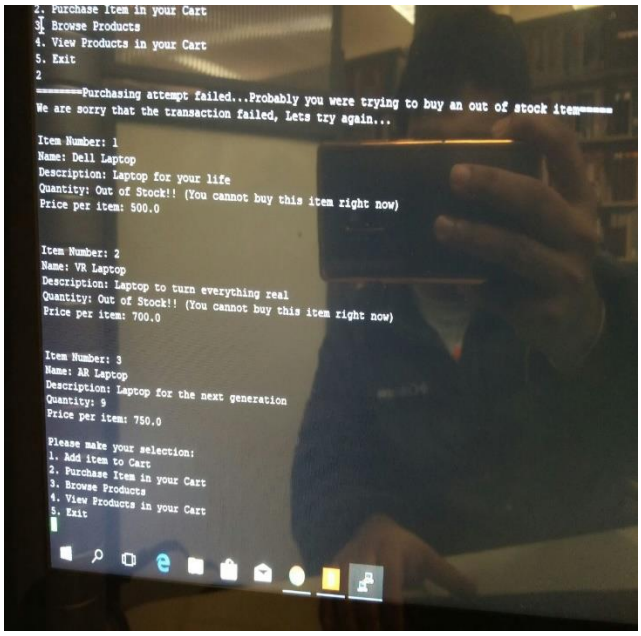
-View Products in cart

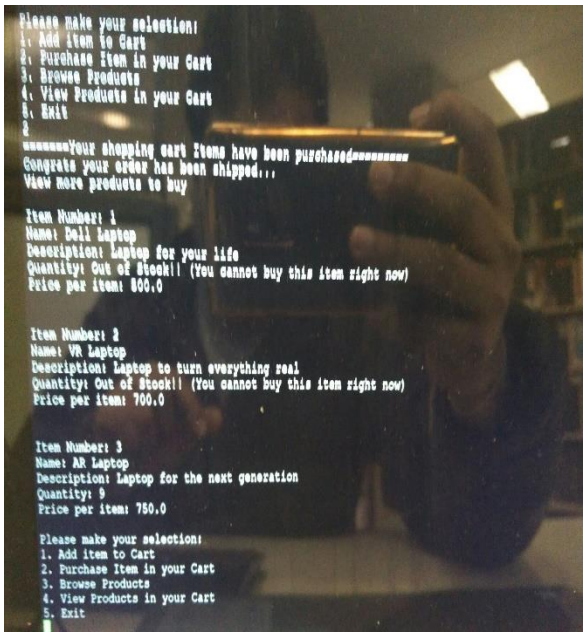
```
Please make your selection:
1. Add item to Cart
2. Purchase Item in your Cart
3. Browse Products
4. View Products in your Cart
5. Exit
4
Items in your Cart:

Item id: 8
Name: Dell Simple
Description: Nice Laptop
Quantity: 2
Price per item: 900.0
```

-Concurrency (4 Clients in parallel)



-Synchronization (Only 1 product left to buy)





11 Conclusion

This has been by far one of the most interesting assignments that I've worked on. Apart from Java RMI, every that happens behind the scenes in any framework like Spring or Struts was understood and implemented. Starting with the big ball of mud to this point has been a real interesting journey. It was all blurry when I started with this application. It was challenging at times to take decisions which could affect my future design and may mean rework in the next assignments. Hence decisions were always made with a foresight about the next parts that I would be implementing in the next assignments. I liked how this project helped me think about application construction like that in industry. I also liked the usage of so many design patterns in such a practical way that it gets embedded in your mind. It was challenging as well as fascinating to think about the product and incrementally move towards it with every assignment.

I'd say that through the course of this project I learned that the industry does not use Java RMI as a tool these days for building distributed application. It might be because of the uncertainties and scaling issues (during concurrency and synchronization) that come along with the framework. So, one thing that I'd want to change regarding this assignment is the use of relevant technologies which have direct usage in the industry at the present juncture of time. This would help us showcase this application as a project on our resume and really enforce our understanding on how to go about building such application if we would be making this for real world application. Also, usage of Java frameworks like Spring or Struts after assignment 2 or 3 would make this application more interesting. This would mean that we would know behind the scene tasks that Java does, but we would also be able to utilize upon the great work that other people have done rather than starting from scratch. Hence, I'd be able to focus on other things (such as increasing scalability, improving security and robustness) and let the frameworks deal with the lower level trivial tasks that I'd to program myself.

If I'd to change something in my design, it would be the following. I think I've a lot of components on the client side (the front controller, market Client controller). I really liked the approach that one of the class mates described during the class. He explained that he kept only the frontend display on client end (very light client part) and everything else on the server-side (starting with the dispatcher). I think by keeping the client as light as possible and doing all the operation on the server end makes more sense rather than having a lot of parts and some computation at the client side. A light weight client means that all types of clients (even with less processing power) can seamlessly work by making web service calls to client side where the dispatcher can collect the request, parse it and delegate it to the required controller. I read somewhere during this project that the SpringMVC framework works in a similar way which is one of the most famous Java based frameworks. So, I'd try to put more components on server side to improve the performance of the system.

Overall this project has been interesting and entertaining to work on. I really enjoyed building small parts connecting them to have a full-fledged system.