Report-Assignment 3

Supriya Cirimoni

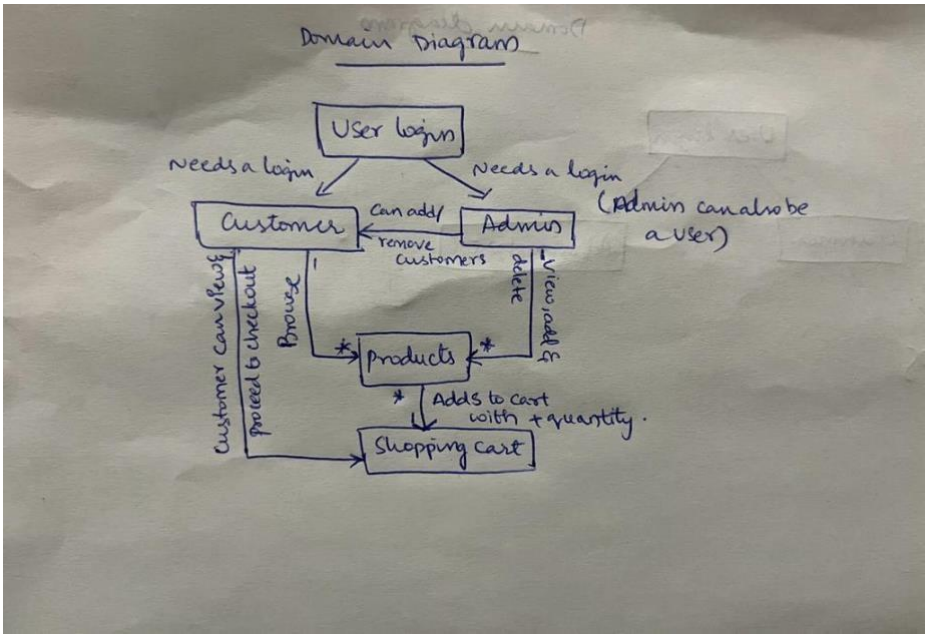
Introduction

The focus of this assignment is on Application Control Patterns that may be applied to the domain model, including the Model-View-Control architecture developed in the previous assignment. Assignment 3 is to implement the Front Controller pattern as well as the authorization pattern to provide a comprehensive login and authorization pattern for both administrators and customers.
Here, we demonstrate the Command and Factory/Abstract Factory design patterns, as well as the Template design pattern, in combination with the Front Controller and Authorization patterns.

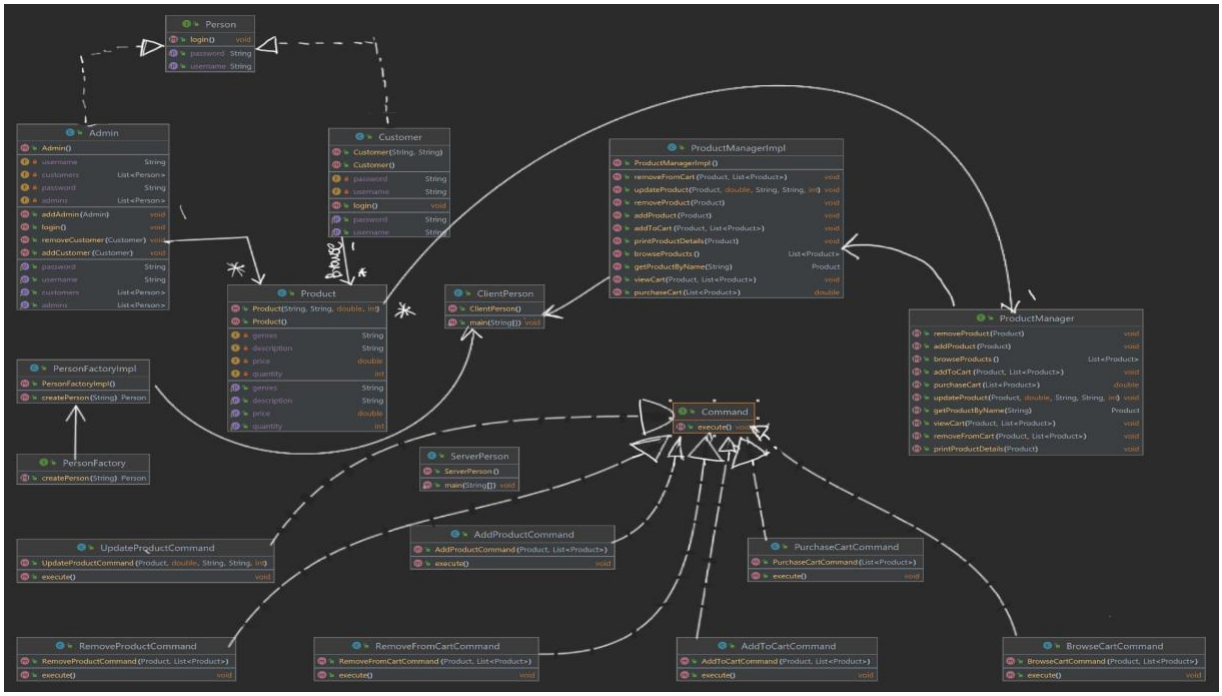
This report aims to improve the previous assignment's online store's operation by strengthening its control mechanisms and offering better authorization and access control using JavaRMI. The online store has browsing items, updating items, removing items, adding items, and purchasing items. Additionally, it supports user and administrator login and registration.
The system is constructed in Java, providing a customer-facing view and an administrator view. This assignment aims to familiarize us with translating customer requirements to a domain model and identifying the classes, class responsibilities, and operations.

Domain Model and Class Diagram

The domain model is a conceptual representation of real-world entities and their relationships within the system.



The class diagram represents the classes, their attributes, and their relationships with one another.



Implementing Authorization Pattern:
The Authorization pattern is used to verify that the online store's resources are secure. This pattern ensures that users are authenticated and approved to access resources and conduct system operations. Here, in this assignment, we implemented the Authorization pattern to offer administrators and customers secure access to the online shop.

To implement the pattern, here, we first defined the roles and permissions for admin and customers. I have authenticated users based on their credentials to guarantee that only authorized users have access to the system's resources.

Roles and Permissions:
Admin: They have complete system access and can perform actions such as adding and removing items, adjusting the prices of the items, and managing customer accounts.
Customer: They have limited access to the system. Customers can only view products, add them to their shopping cart, and purchase items.

Implementing Command Pattern:
The command pattern is a data-driven design pattern that belongs to the behavioral pattern category. A request is wrapped in an object and sent to the invoker object as a command. The

Invoker object searches for an object that can handle this command and sends it on to the matching object, which performs the command.

Here, I have implemented the pattern with the help of the following class.

Command: This is just a simple interface that contains a declaration of just one method called **execute**.

Implementing Abstract Factory Pattern:

The Factory Pattern is a creational design pattern that allows objects to be formed without specifying the specific type of item to be created. It enables the generation of objects to be delegated to a factory class that understands which object to build based on the input. On the server side, we created a remote object that implements the Factory interface. The client will call the method on the remote object to create a specific type of object.

Implementing MVC Architecture:
MVC stands for model, view, and controller. It is a design pattern that is used to create user interfaces, data, and control logic, which is achieved by separating the components.

Design
The design of this online store system consists of three main components: the server, the client, and the common objects.

Server
The server component is responsible for maintaining the state of the system and providing the business logic for the online store. The server is implemented using Java RMI, which allowed the client to communicate with the server and make remote method invocations.

The "PersonFactoryImpl" and "ProductManagerImpl" classes implement the "PersonFactory" and "ProductManager" interfaces respectively, which are defined in the "common" package.

Client
The client communicates with a remote server that provides access to a list of products and allows customers to browse through the products, add them to a cart, remove them from the cart, and purchase the items in the cart. This has both customer and Admin interfaces to manage.
Here, I used the PersonFactory and ProductManager interfaces, which are defined in the common package, to interact with the remote server.
PersonFactory- is used to create instances of the Customer and Admin classes. ProductManager- is used to access and manage the product list.

Here, in the end, I have also created a JAR file.
I have also created a make file to compile and run the code.

Use Cases

1. User/Administrator: The user and administrator both will be able to register for an account and log in to their account. However, Administrators will also be able to add and remove customer and administrator accounts.

2. Browsing: The customer will be able to browse items in the store.

3. Updating: Administrators will be able to change the description, price, and quantity of an item in the system.

4. Delete: Administrators can delete items from the system.

5. Adding: Administrators can add new items to the system.

6. Purchasing: Customers can purchase items from the shopping cart. The system will prevent the customer from buying more items than are currently available.

Note that multiple users can simultaneously log in and use the application. Following is the list of tasks/actions the user can perform using this application.

Admin:

- Admin Registration
- Admin Login
- Add Customer
- Remove Customer
- Update product • Remove product.
- Add product.

Customer

- Customer Registration
- Customer Login
- Browse product.
- Add the product to the cart.

- Purchase product
- Remove the product from the cart.
- Purchase the product.

Implementation

It is tested by running the program on JavaRMI by making a connection between Server and the Client.

Once, the connection is established, Select the desired option, and continue shopping. Happy Shopping!

Results (Screenshots on Tesla Machine):

Please find the below-attached screenshots of the results after securing the Client-Server connection:

Server :

```
scirimo@in-csci-rrpc03:~/sampleAssign3$ make server
java Server.ServerPerson
Server started
PersonFactory Object Created
Rebind Done!
ProductManager Object Created
Rebind Done!
Server running...
```

Client:

Login –

```
Welcome to online Shopping store!
Are you Customer or Admin:
customer
1.Login 2.Register
1
Enter Username:
customer1
Enter Password:
Pass@1234
Login Successful
Select 1.Add to cart
2.Remove from cart
3.View Items in cart
4.Purchase Items in cart
5View Products
6.Logout
```

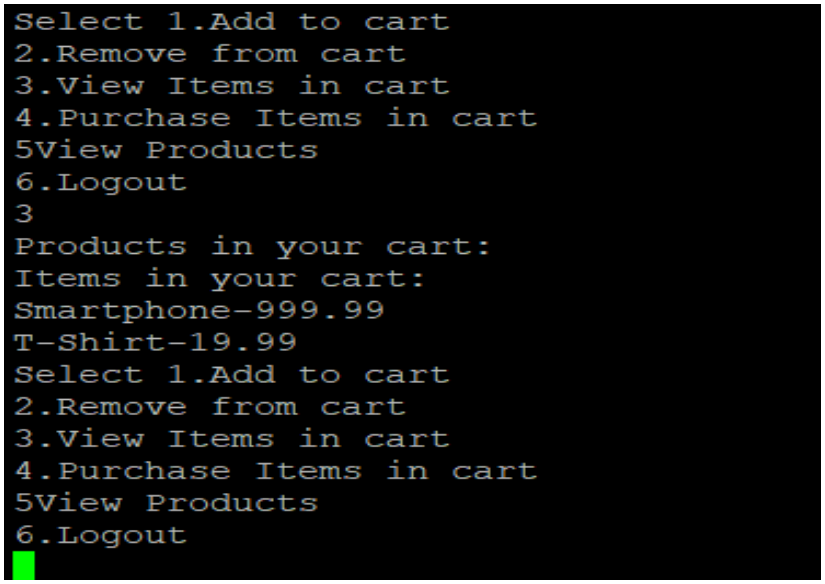
View Products –

```
Select 1.Add to cart
2.Remove from cart
3.View Items in cart
4.Purchase Items in cart
5View Products
6.Logout
5
Products:
Electronics - Smartphone - $999.99 - 5 in stock
Clothing - T-Shirt - $19.99 - 10 in stock
Home - Coffee Maker - $49.99 - 3 in stock
Select 1.Add to cart
2.Remove from cart
3.View Items in cart
4.Purchase Items in cart
5View Products
6.Logout
```

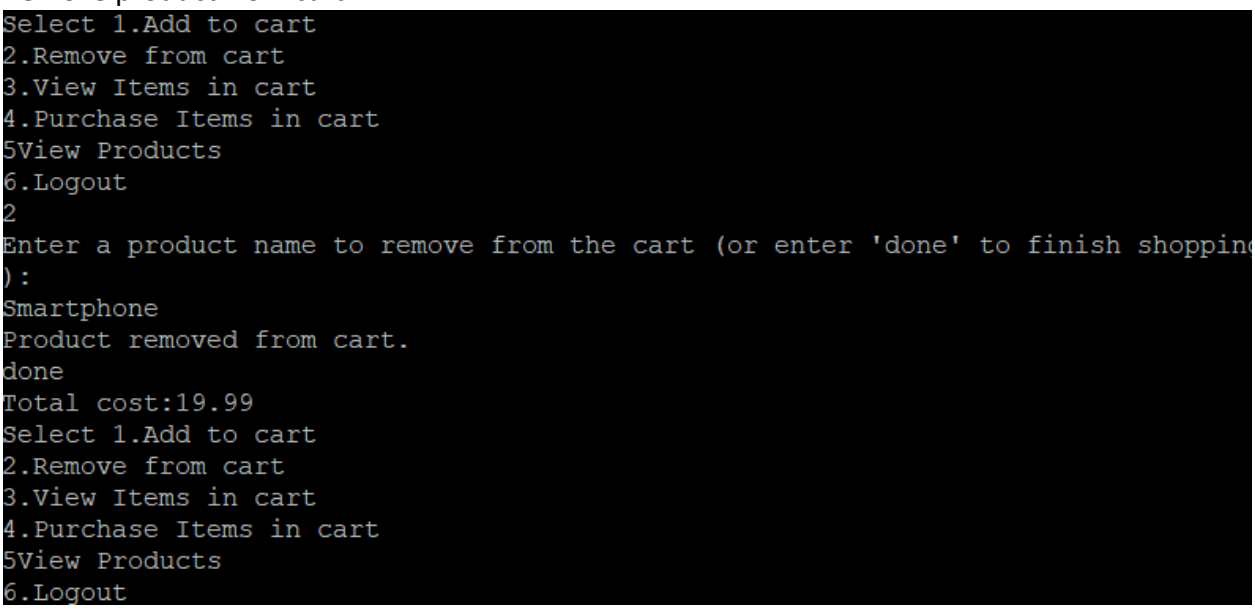
Add to Cart -

```
Products:
Electronics - Smartphone - $999.99 - 5 in stock
Clothing - T-Shirt - $19.99 - 10 in stock
Home - Coffee Maker - $49.99 - 3 in stock
Select 1.Add to cart
2.Remove from cart
3.View Items in cart
4.Purchase Items in cart
5View Products
6.Logout
1
Products:
Electronics - Smartphone - $999.99 - 5 in stock
Clothing - T-Shirt - $19.99 - 10 in stock
Home - Coffee Maker - $49.99 - 3 in stock
Enter a product name to add to your cart (or enter 'done' to finish shopping):
Smartphone
Product Added!
T-Shirt
Product Added!
done
Total cost:1019.98
Select 1.Add to cart
2.Remove from cart
3.View Items in cart
4.Purchase Items in cart
5View Products
6.Logout
```

View cart-



Remove product from cart –



Purchase:

```
Products in your cart:
Items in your cart:
T-Shirt-19.99
Select 1.Add to cart
2.Remove from cart
3.View Items in cart
4.Purchase Items in cart
5View Products
6.Logout
4
Purchase Successful with total cost:19.99
Select 1.Add to cart
2.Remove from cart
3.View Items in cart
4.Purchase Items in cart
5View Products
6.Logout
```

Logout –

```
Select 1.Add to cart
2.Remove from cart
3.View Items in cart
4.Purchase Items in cart
5View Products
6.Logout
6
Logged out!
scirimo@in-csci-rrpc06:~/sampleAssign3$
```

Admin –

Admin login or register

```
scirimo@in-csci-rrpc06:~/sampleAssign3$ make clientPerson
java Client.ClientPerson
Welcome to online Shopping store!
Are you Customer or Admin:
Admin
1.Login 2.Register
```

Login

```
1.Login 2.Register
1
Enter Username:
admin1
Enter Password:
Pass@1234
Login Successful
Select 1.view products
2.Add product
3.Remove product
4.Update product
5.Add admin
6.Add customer
7.Remove customer
8.Logout
```

Register:

```
Welcome to online Shopping store!
Are you Customer or Admin:
Admin
1.Login 2.Register
2
Enter Username:
Spandana
Enter Password:
Cirimoni
Registration Successful
Select 1.view products
2.Add product
3.Remove product
4.Update product
5.Add admin
6.Add customer
7.Remove customer
8.Logout
```

View products at admin

```
1
Products:
Electronics - Smartphone - $999.99 - 5 in stock
Clothing - T-Shirt - $19.99 - 10 in stock
Home - Coffee Maker - $49.99 - 3 in stock
Select 1.view products
2.Add product
3.Remove product
4.Update product
5.Add admin
6.Add customer
7.Remove customer
8.Logout
```

Admin add product

```
2
Product Type:
laptop
Product Description:
mac book pro
Product price:
599.99
Product quantity:
3
Product Added
List of products:
Smartphone - 999.99
T-Shirt - 19.99
Coffee Maker - 49.99
mac book pro - 599.99
Select 1.view products
2.Add product
3.Remove product
4.Update product
5.Add admin
6.Add customer
7.Remove customer
8.Logout
```

Removing Product:

```
Select 1.view products
2.Add product
3.Remove product
4.Update product
5.Add admin
6.Add customer
7.Remove customer
8.Logout
3
Enter the product name to remove:
T-Shirt
Product is removed
```

Viewing Products after viewing:

```
Products:
Electronics - Smartphone - $999.99 - 5 in stock
Home - Coffee Maker - $49.99 - 3 in stock
mac - book - $999.99 - 5 in stock
Select 1.view products
2.Add product
3.Remove product
4.Update product
5.Add admin
6.Add customer
7.Remove customer
8.Logout
```

Admin adding new customer.

```
6
Enter Customer uname:
supriya11
Enter customer password:
customer
Customer added
Select 1.view products
2.Add product
3.Remove product
4.Update product
5.Add admin
6.Add customer
7.Remove customer
8.Logout
```

Admin Logout

```
8
Logged out!
scirimo@in-csci-rrpc06:~/sampleAssign3$
```

Conclusion:

In conclusion, I designed and implemented an online store that meets the client's specifications. I identified the domain-level classes and attributes, created a domain model and class diagram, and developed the classes and use cases required to demonstrate a working Java RMI

application. I've also included a client application that allows the user to interact with the system.