Outline of Files in Zip Submission

**Assignment #3 report**

Word document that contains the Class-responsibility-collaborator table. It also has the Domain model, UML diagram, and the use case diagram. It also contains the screen shots as required. It also contains the details of the design patterns (Command, Front controller, Template Pattern, factory/Abstract factory, Authorization process, Singleton, and observer) and where they were used. Please refer to this file to get the full scope of the design patterns and their implementations.

**Diagrams folder**

Contains the Domain model diagram, the UML diagram and use case diagram.

**Screenshots**

Contains the screenshots of some of the test runs as per requirement of the assignment.

**README.**

This file. Describes the files submitted.

**Onlinestore3: Actual folder of the program**

Contents:

Client.jar: The jar file to run the Customer side of the program.

ClientView Directory: package

Client.class: The class file of the Client.java program

Client.java: The file that handles the interaction of the customer with the client side.

Controller directory: package

A list of java and class files that uses a bunch of design patterns such as the Command pattern (Command.java and its subclasses), Front controller pattern (ServerFrontController.java, ClientFrontController.java, dispatcher.java, ClientDispatcher.java, the template design pattern, the authorization pattern (Authorization.java)

Makefile: To run the program.

Manifest-client.txt: The manifest file for creating the jar file for the Client.java

Manifest-server.txt: The manifest file for creating the jar file for the StoreServer.java

Model: A list of files that models a consistent data storage and manipulation such as Account, and its container. Products and its container. It uses the template design pattern.

Serverview: Houses the main file for the server side.

StoreServer.jar: the jar file for server side.

**How to run: Makefile.**

Server: enter in the command “make” to compile and then “make run-server” to run the server program

Client: enter in the command “make” to compile and then “make run-client” to run the client program.

Note: I used two registry “rmiregistry 2044 &” “and rmiregistry 2045 &”. One for the accounts container and another for the products container

**How to run: jar file.**

“Java -jar StoreServer.jar” to run the server.

“Java -jar Client.jar” to run the client.

Extra notes (important):

It is important that the server is up and running before the clients tries to perform any operation. It does not make sense that there is a customer in a store without an employee.

Another point is that, if a person enters the wrong information, they will still be able to enter the system, but they will be limited in what operation they can perform. This is done with the use of the Authorization pattern.