Online Marketplace Application Using Role Based Access Control

In this project, the online marketplace is implemented in the Front Controller design pattern which is built upon the basis of Model-View-Controller (System Architectural pattern) with the RMI interface. In this application, the Front Controller will handle all the requests with the single handler by means of the centralized request handling mechanism. In this project the role of the Front Controller is to Authenticate and Authorize the requests made by the Customer/Client and the handler will track the requests when the login credentials are provided by means of request to the handler.

Functions of the Components of the Front Controller:

1. Front Controller:(MarketplaceFrontController) It handles all the requests made by the Client or Customer during the login and Front Controller has Single handler for handling all the requests.
2. Dispatcher: (MarketplaceDispatcher) The dispatcher object will be used for the sending the request to the specific handler.
3. View: (CustomerView,AdministratorView) the view acts an object for making a request.

Abstract Factory Pattern

In this project the abstract factory pattern is created to effectively use the factory objects without separate creating the classes for it. AbstractFactory is an abstract class that the all the factory classes will extends property of AbstractFactory class.

1.The AbstractFactory class is an abstract class that contains all the abstract methods.

2. The factory classes (Concrete Class) will extends the methods of AbstractFactory classes and it then creates the objects of all the factory class.

3. MarketplaceGadget is an interface that responsible for creating the class object that are invoked by the AbstractFactory upon the request of the FactoryProducer class.

4. The GadgetFactory is the class that extends the abstract class from the AbstractFactory class.

5. The Abstractfactory sends the method information to the GadgetFactory for the appropriate object that returns the item of type gadget.

6. The AbstractFactory class is an abstract class that passes the information to the interfaces like MarketplaceGadgets and the MarketplaceFurniture interfaces takes the information from the GadgetFactory and FurnitureFactory respectively and sends the appropriate object for the FactoryProducer class.

Command Pattern:

1. In this project the command pattern is used to invoke the appropriate Concrete command class for the request given by the CommandRequest class.

2. Invoker passes the request given by the CommandRequest class and it sends to the commandInterface for the getting required object.

3. Command pattern is used as data handling in application, it is meant for the responsible for behavior of the system.

4. the Command Interface then passes the request from the Invoker class and fetches the exact object by using the Concrete Command Class.

5. The role of the Invoker class is to Invoke the messages and wraps it as an object.

6. After getting the exact object from the invoker the it is then sent to the Client to display it as a view.

The Purpose of the application and the design patterns is to check for the user authentication and to take to the respective page to the user. That is, Customer page for the customers and Administrator page for the Administrators.

Reflection Pattern:

In this application, I have used the reflection pattern to modify or to extend the functionality of the application. As the application need to be changed upon the requirements change. So, the reflection pattern is well suited for handling sudden change in the functionality of the application without impacting the other part of the application. This pattern is used to minimize the dependency of the one class on the other class, which separates the concerns. The pattern also maintains the abstraction level by not letting the object to know the operation of request that has passed to it and maintains the consistency of the application.