Market Place

Assignment Report 1

Virinchi Sainath Nalluri
vnalluri@iupui.edu

30th January, 2018

Table of Contents

Objective

The objective of this assignment is to provide skeleton framework for online marketplace, make a domain model, and classes necessary for MVC design pattern and also implement Java RMI for the project.

Assignment Discussion

Role of **Java RMI**

Java RMI helps in interaction between two different JVMs. One JVM can remotely call a method in another JVM. Hence, the name Remote Method Invocation (RMI). It contains an interface which extends the java.rmi.Remote class. In our case it is Model Interface. This interface contains all the methods according to our use case. In the marketplace project, ModelImpl class implements the Model Interface. In the ModelImpl, we bind a String name to the ModelImpl, which is a Remote Object (as Model is extending Remote). Later the binded string can be used to get this remote object.

This is done in ControllerImpl class. The string is used to pick up the remote object (ModelImpl). Using this object, we call the required methods according to our use case. For the first assignment, we simply are displaying that they are connected.

Role of **MVC Pattern**

The MVC Pattern essential contains Model View Controller.

Model is the component where all the logic and processing instructions reside.

View is the front-end, with which the user interacts with.

Controller is the middleware between view and model. Model and view doesn't know of each other existence. However, controller is the part which acts as mediator between View and Model.

Working : Based on the user's interaction with view, this interaction message is sent to the controller. And controller then invokes the appropriate method in the model, which does the logic processing and displays the result.

In this assignment, ModelImple class acts as the Model, ControllerImpl class acts as the Controller and View class acts as View.

MVC and Java RMI Architecture

The Model component of the project, Model interface extends interface. Hence, ModelImpl which implements Model, is also remote. So, when the user interacts with the view, the instruction is sent to controller. The controller then captures the remote ModelImpl object using Naming.loopUp. The captured object has the different methods such as addProduct, updatePrice etc. Using the object reference variable we invoke the appropriate method depending on the user's interaction with view.

Domain Model

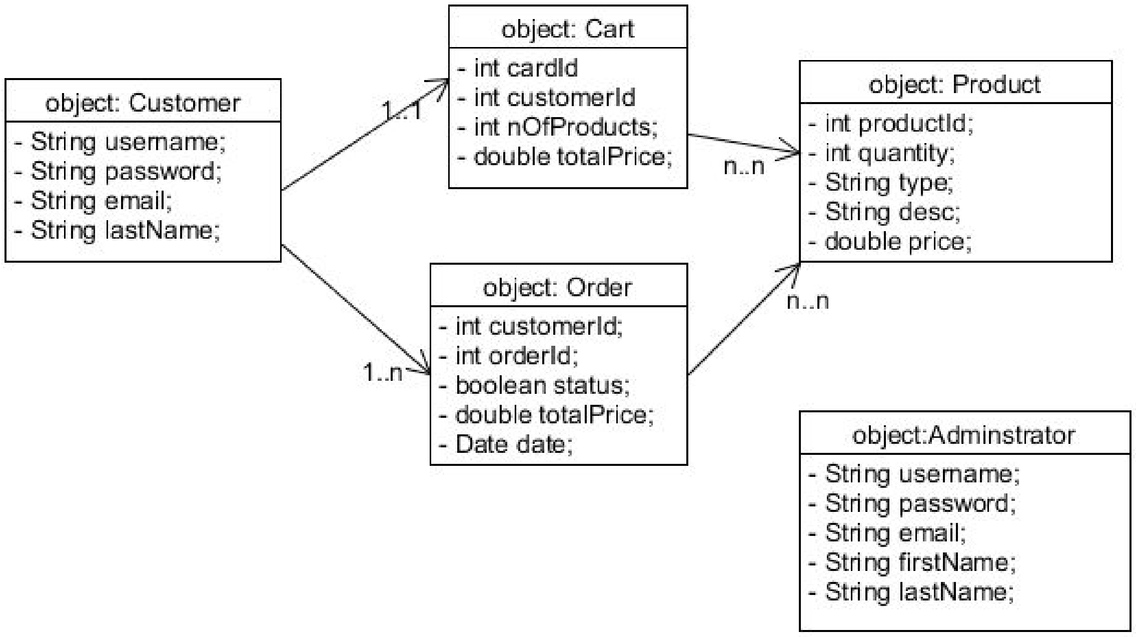


Fig Domain Model

Associations:
Customer has a Cart. A Cart has many products. An Order has many products. However, there is no association between Administrator and other entities, because, he merrily edits the products and deletes the Customers.

__Classes Created Related to the Domain Model:__
Customer.class, Administrator.class, Cart.class, Order.class, Product.class.

All these entity classes have their attributes as shown in the domain model and have their behavior as methods.

__Other Classes Created :__

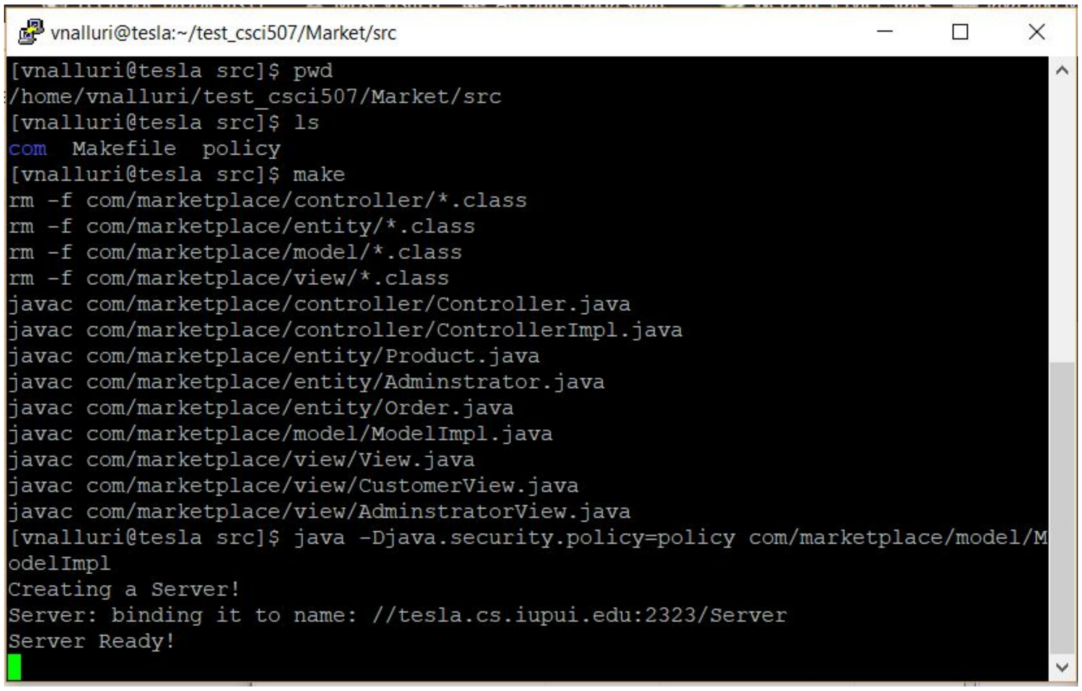
Model — Model.class (Interface), ModelImpl.class
Controller — Controller.class (Interface), ControllerImpl.class
View — ViewInterface.class (Interface), View.class

Also, I have AdmistratorView.class and CustomerView.class, just in case in the future if we want to do it separately. But for now, we are only connecting to Server through View.

AdminstratorView.class and CustomerView.class has functionalities which the application offers to users depending on the user type. Hence sepeate interfaces have been given for Adminstartor and Customer.

Sample Runs

1. Creating the server and making it ready for the user to connect.

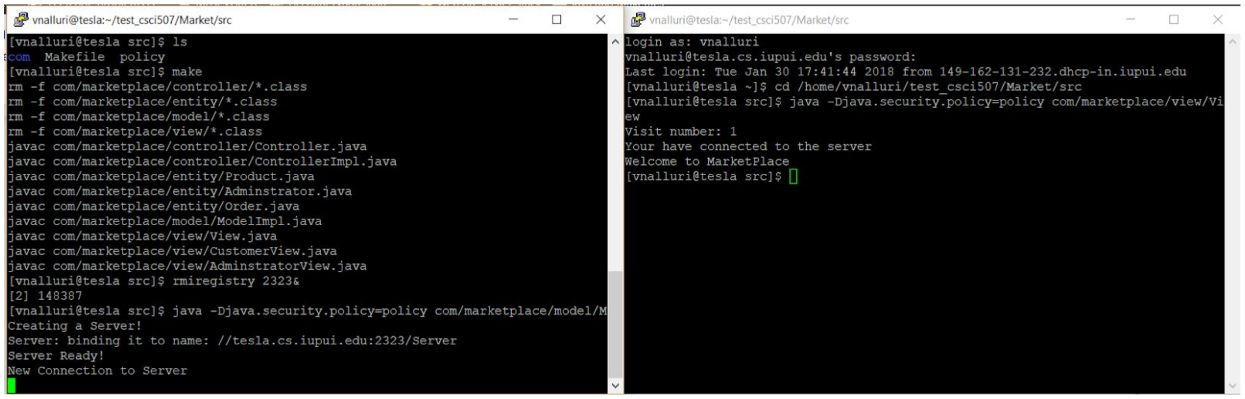


```
[vnalluri@tesla src]$ pwd
/home/vnalluri/test_csci507/Market/src
[vnalluri@tesla src]$ ls
com  Makefile  policy
[vnalluri@tesla src]$ make
rm -f com/marketplace/controller/*.class
rm -f com/marketplace/entity/*.class
rm -f com/marketplace/model/*.class
rm -f com/marketplace/view/*.class
javac com/marketplace/controller/Controller.java
javac com/marketplace/controller/ControllerImpl.java
javac com/marketplace/entity/Product.java
javac com/marketplace/entity/Adminstrator.java
javac com/marketplace/entity/Order.java
javac com/marketplace/model/ModelImpl.java
javac com/marketplace/view/View.java
javac com/marketplace/view/CustomerView.java
javac com/marketplace/view/AdminstratorView.java
[vnalluri@tesla src]$ java -Djava.security.policy=policy com/marketplace/model/M
odelImpl
Creating a Server!
Server: binding it to name: //tesla.cs.iupui.edu:2323/Server
Server Ready!
```

Fig: Server Window

The Server Window shows that server has been created and is ready.

2. Making user 1 connect to the server



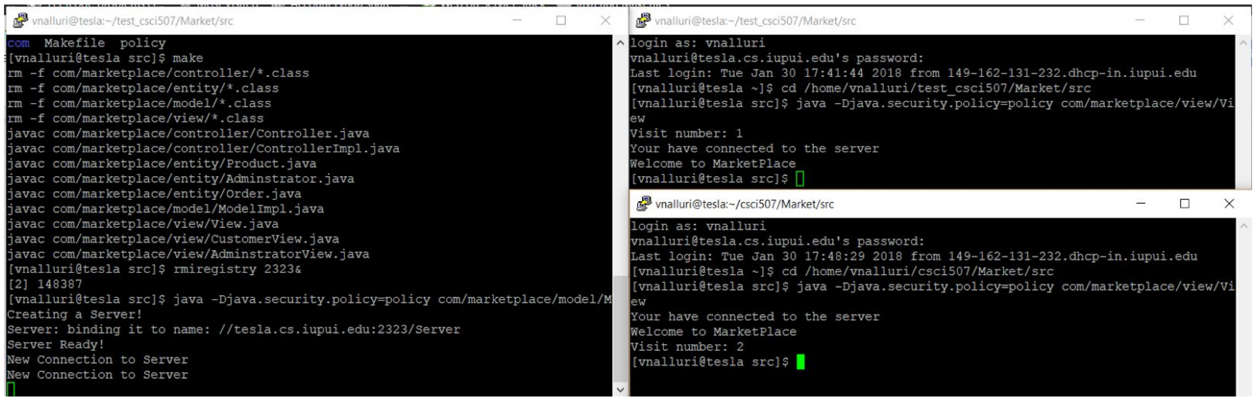
```
vnalluri@tesla:~/test_csci507/Market/src
[vnalluri@tesla src]$ ls
com  Makefile  policy
[vnalluri@tesla src]$ make
rm -f com/marketplace/controller/*.class
rm -f com/marketplace/entity/*.class
rm -f com/marketplace/model/*.class
rm -f com/marketplace/view/*.class
javac com/marketplace/controller/Controller.java
javac com/marketplace/controller/ControllerImpl.java
javac com/marketplace/entity/Product.java
javac com/marketplace/entity/Adminstrator.java
javac com/marketplace/entity/Order.java
javac com/marketplace/model/ModelImpl.java
javac com/marketplace/view/View.java
javac com/marketplace/view/CustomerView.java
javac com/marketplace/view/AdminstratorView.java
[vnalluri@tesla src]$ rmiregistry 2323&
[2] 148387
[vnalluri@tesla src]$ java -Djava.security.policy=policy com/marketplace/model/M
Creating a Server!
Server: binding it to name: //tesla.cs.iupui.edu:2323/Server
Server Ready!
New Connection to Server
```

```
vnalluri@tesla:~/test_csci507/Market/src
login as: vnalluri
vnalluri@tesla.cs.iupui.edu's password:
Last login: Tue Jan 30 17:41:44 2018 from 149-162-131-232.dhcp-in.iupui.edu
[vnalluri@tesla ~]$ cd /home/vnalluri/test_csci507/Market/src
[vnalluri@tesla src]$ java -Djava.security.policy=policy com/marketplace/view/Vi
ew
Visit number: 1
Your have connected to the server
Welcome to MarketPlace
[vnalluri@tesla src]$
```

Fig: (Left to Right) Server window and User 1 window

As soon as user 1 connects, in the server window "New Connection to Server" message is printed. And in user 1 window, welcome message is printed with his visitor number.

3. Making user 2 connect to the server



```
vnalluri@tesla:~/test_csci507/Market/src
com  Makefile  policy
[vnalluri@tesla src]$ make
rm -f com/marketplace/controller/*.class
rm -f com/marketplace/entity/*.class
rm -f com/marketplace/model/*.class
rm -f com/marketplace/view/*.class
javac com/marketplace/controller/Controller.java
javac com/marketplace/controller/ControllerImpl.java
javac com/marketplace/entity/Product.java
javac com/marketplace/entity/Adminstrator.java
javac com/marketplace/entity/Order.java
javac com/marketplace/model/ModelImpl.java
javac com/marketplace/view/View.java
javac com/marketplace/view/CustomerView.java
javac com/marketplace/view/AdminstratorView.java
[vnalluri@tesla src]$ rmiregistry 2323&
[2] 148387
[vnalluri@tesla src]$ java -Djava.security.policy=policy com/marketplace/model/M
Creating a Server!
Server: binding it to name: //tesla.cs.iupui.edu:2323/Server
Server Ready!
New Connection to Server
New Connection to Server
```

```
vnalluri@tesla:~/test_csci507/Market/src
login as: vnalluri
vnalluri@tesla.cs.iupui.edu's password:
Last login: Tue Jan 30 17:41:44 2018 from 149-162-131-232.dhcp-in.iupui.edu
[vnalluri@tesla ~]$ cd /home/vnalluri/test_csci507/Market/src
[vnalluri@tesla src]$ java -Djava.security.policy=policy com/marketplace/view/Vi
ew
Visit number: 1
Your have connected to the server
Welcome to MarketPlace
[vnalluri@tesla src]$
```

```
vnalluri@tesla:~/csci507/Market/src
login as: vnalluri
vnalluri@tesla.cs.iupui.edu's password:
Last login: Tue Jan 30 17:48:29 2018 from 149-162-131-232.dhcp-in.iupui.edu
[vnalluri@tesla ~]$ cd /home/vnalluri/csci507/Market/src
[vnalluri@tesla src]$ java -Djava.security.policy=policy com/marketplace/view/Vi
ew
Your have connected to the server
Welcome to MarketPlace
Visit number: 2
[vnalluri@tesla src]$
```

Fig: Left: Server Window; Top Right: User 1 window; Bottom Right: User 2 window

As soon as user 2 connects, again a "New Connection to server" message is printed and in user 2 window, a welcome message and his visitor number is printed.

Conclusion And Analysis

With this assignment, a skeleton framework has been make for online marketplace. A domain model has been make for the application. And classes required for running of the MVC software design pattern whose implementation is done using Java RMI.

With this assignment, focus has been laid on Java RMI, the basis on which enterprise systems are made. And MVC pattern, which depicts a pattern in which different aspects of work has been distributed into different components.

References

1. For writing make file : https://www.cs.swarthmore.edu/~newhall/unixhelp/howto_makefiles.html
2. Java RMI : Bank example in Canvas
3. MVC Pattern : Calculator Example in Canvas
4. Technologies Used : Eclipse IDE, WinScp, PUTTY, Tesla, Java