



Scrum Masters, Inc.

Submission #4 - Beta Release

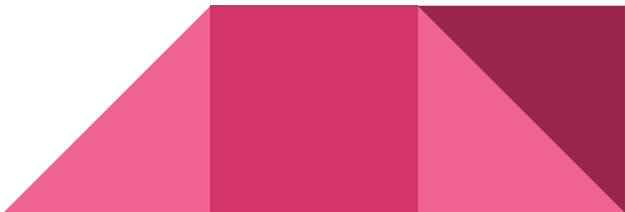
Lessons Learned - Design Patterns

- Before taking this course, not many of us were familiar with Design Patterns at all
 - Previously, we were all used to just getting an assignment and immediately starting to write the practical Java code
 - The use of Design Patterns makes programming projects, especially in a group situation, much easier to manage and complete
 - Learning about and working with the different classes of patterns, such as Creational, Structural and Behavioral really helped us to become better programmers and better understand how to handle specific tasks
- 

Lessons Learned - Don't Switch Designs

- Our original plan for Submission #4 was to switch to design utilizing the Abstract Factory Design Pattern
 - We planned on having an abstract Transaction class and utilizing that class to create the functionality for our Sale, Return and Rental classes
 - However, that was much easier said than done...
 - We realized that the way we originally designed our project was already effective enough
 - Switching the design at the last phase would have been catastrophic and would honestly have ruined our project
- 

Lessons Learned - Coding in a Team

- We learned that using repositories like Git are more than convenience, they are basically a necessity
 - The only way to efficiently have multiple people working on the same code at the same time is to exercise heavy version control
 - You need to consider how small changes in your code will affect the project as a whole and even small bug fixes and problem solving can involve large changes between different versions of the code
 - Dividing the work up so that there is little overlap is a great way to allow people to fit coding into their schedules
- 


Lessons Learned - Use Cases

- Use cases are a great way of outlining all of the necessities and requirements that a project must meet
- They provide a good way to divide up work assignments and allow team members to focus on one area at a time
- They are also useful for exploring and outlining all of the errors and incorrect uses that the project must cover through the alternative scenarios
- Use cases are in a simple and easily readable format so clients and people with no computer science background can understand the processes



User Manual - Rental

After logging in, click on the **Rental** button in the main screen. You will be brought to a menu that will allow you to add items to the order. Enter the ID of the item, the quantity, and the number of days the item will be rented for in the text boxes. If the item is being returned, please also enter the transaction ID that is provided on the customer's receipt. Note that while you must provide a quantity and number of days when adding an item to an order, if you are processing a rental return, the numbers will not matter and only item ID and transaction ID will be used. However, you still must enter a number for these two fields. Once all the information is entered, click the **Add Order** button to enter the item into the order list. Repeat for all necessary items. Once the order list is complete, click the **Submit Order** button to place the order or the **Return Rental** button to return an item already purchased. Once the **Return Rental** button is pressed, the item is returned and no further information is required. The **Submit Order** button will bring you to the payment window where a customer may pay with cash or credit. If a customer pays with cash, enter the amount given into the window and click the **Cash** button. The change will be automatically calculated and displayed and click **End Sale** to complete the transaction. If a customer pays with credit, click the **Credit** button to bring up a new window. The new window will prompt you for the customer's credit card number which should be entered and then click **Submit Info**. The credit card will be checked against the database that contains all credit cards for validity. If the card is invalid, the system will prompt you for a valid card number. Once a valid number has been entered, the window will close and you will be returned to the previous screen with the payment options. Click **End Sale** to finish the transaction.



User Manual - Database

Our database was made within netbeans. The database was a local apache derby database, accessed through a jdbc driver. to replicate our database, following steps.

- Go to services tab in netbeans
- Open databases subtab
- Right click on java DB
- Select properties
- Change the directory settings to Shane's DB stuff
- Right click on scrumDB and click connect
- Username and Password is "scrum"
- Open and run the script DB.sql in netbeans to populate the data

Your database should now be created properly and populated properly.



Beta Release - Changes

- Added support for backorders and displays message to cashier indicating an item has been backordered for a customer
- Numerous minor bug fixes
- Ensuring the program still functions with improper/incorrect inputs
- Fixed the Receipt to also print for credit card transactions

