Reflective Report

Catshop

Vickshan Vicknakumaran | CI553 | Student ID: 20824389

Contents

[Introduction 2](#_Toc105155888)

[Tools 2](#_Toc105155889)

[IDE (Integrated Development Environment) 2](#_Toc105155890)

[Cloning via GitHub 2](#_Toc105155891)

[WhiteStar UML 4](#_Toc105155892)

[Project Management 4](#_Toc105155893)

[Development 4](#_Toc105155894)

[Merge Quantity of same products 4](#_Toc105155895)

[Adding Color to buttons 5](#_Toc105155896)

[Adding Sounds 5](#_Toc105155897)

[Sort Products by its ID 6](#_Toc105155898)

[Remove Button 6](#_Toc105155899)

[Testing 7](#_Toc105155900)

[Junit 7](#_Toc105155901)

[UML Diagram 8](#_Toc105155902)

[Use-Case Diagram 8](#_Toc105155903)

[Sequence Diagram 8](#_Toc105155904)

[Documentation 9](#_Toc105155905)

[JavaDoc 9](#_Toc105155906)

# Introduction

The aim of this coursework is improved onto the Catshop model program, which was provided by Rodger through GitHub. Catshop is a digital store system which delivers customer orders using multiple points. There is seven windows Customer Client, two Cashier Client, BackDoor Client, Pick Client, Display Client, and Collect Client. In the cashier model, Customers were able to find and check product price and quantity using the catalogue number. If the quantity did exist, they had to buy the item. If they wanted to remove an item, a remove option was available for them by entering the product ID of the product they would like to remove. If they wanted to place an order, they would get a unique order number which would be displayed to them. Once their order has been picked, they would be able to collect their order.

Github link to the code : https://github.com/vickshan001/2021-CI553-catshop

# Tools

### IDE (Integrated Development Environment)

At the start of the coursework, I preferred to use “IntelliJ” as it is more familiar to the IDE. Unfortunately, I was not able to continue using this IDE as the program did not function at all. Therefore, I had moved to a different IDE called “Eclipse” as the program was created on Eclipse and the program was functioning.

### Cloning via GitHub

To clone via GitHub, I had to fork Catshop from “2021-CI553-catshop”. After this I had used eclipses import function to clone the project file from GitHub.

Graphical user interface, text, application

Description automatically generated

Figure 1 – Import window

Figure 1 shows the first step of cloning which is File > Import > Git > Projects from Git.

Graphical user interface, text, application

Description automatically generated

Figure 2 – Repository Source

After selecting Git with smart import, I used clone URI. To get the URI, I had to go to the forked ci553-catshop under my name form GitHub and gets its URL.

Graphical user interface, text, application, email

Description automatically generated

Figure 3 – Git R epositiry

I had input the forked URL into the URI which would fill the other input boxes automatically except the Authentication section. For Authentication, I had to input my username on GitHub and for the password we use our personal access token. In order get the personal access token, I had use GitHub developer setting and generate a personal access token which we can decide the expiration for the token to work.

### WhiteStar UML

WhiteStar UML is to software modeler aimed to support agile and concise modeling. WhiteStar UML was used to create use-case diagram and sequence diagram which allowed me to demonstrate the program.

# Project Management

When deciding on the features which I wanted to add onto the program. It was important that these features are useful for the client as well as feature which would be achieved based on my skillset. I decided to do a priority list were the most priority feature would be first to be done and vice versa. This priority list allowed me to work more efficiently and organised.

# Development

### Merge Quantity of same products

This feature merges the quantity of the same product ordered. To do this, I had made changes in BetterBasket where I had made changes to the add method. An if statement is being used to run the sortbasket method to sort the ordernumber then a for loop to check if there are any product that are the same and add one to those which do.

When the cashier adds the same product again the quantity will change and displayed to the user. With the code which provided at the start of the coursework, if the same scenario occurred then it would make a new line with the same product listed.

Text

Description automatically generated

Figure – Merge Quantity in Better Basket

### Adding Color to buttons

I had added color to each button so GUI for the users would be more easy access. To detect the movement the mouse over the button, I had to use an mouseListener function as well as set background color and text

A screenshot of a computer

Description automatically generated with medium confidence

Figure - MouseListner in View Class

### Adding Sounds

To add this function, I had made sound folder where it would consist of the sound files and a java class where it would open the sound and sound would be played when the class was called. To use it in other classes, you would have to call the class and then add this function, with sound file name, to code which you would like the sound to be played.

Timeline

Description automatically generated

Figure - Sound Class in Sound Folder

### Sort Products by its ID

In the original code, when a person buys an item, it unorganised. Therefore, I had created a function called sortBasket in the BetterBasket Class, where it would compare the previous item to its current item.

Text

Description automatically generated

Figure - Sort Products in BetterBasket Class

### Remove Button

Originally, when a user bought an item, they did not have the ability to remove an item. Therefore, I had included a remove function in the CashierModel Class. The function takes the productid of the item which is needed to be removed. Then loops through each item in the basket to find the product in the basket. When the item is found, the quantity is subtracted. If the Quantity equals to zero then the item is removed from the list.

Text

Description automatically generated

Figure - Remove Function in BetterBasket Class

# Testing

### Junit

When working on this project I decided to test the program through Junit testing, as it’s easy to implement into eclipse.

Graphical user interface, text, application

Description automatically generated

Figure - Junit Testing

Table

Description automatically generated with medium confidence

Figure - Junit Testing

# UML Diagram

### Use-Case Diagram

Diagram

Description automatically generated

Figure - UseCase Diagram of catshop

### Sequence Diagram

Diagram

Description automatically generated

Figure - Customer Client Sequence Diagram

A picture containing diagram

Description automatically generated

Figure - Cashier Client Sequence Diagram

# Documentation

### JavaDoc

The JavDoc was made using the plugin which was provided by eclipse.

Graphical user interface, text, application, email

Description automatically generated

Figure - Java Documentation

[Figure 1 – Import window 2](#_Toc105156307)

[Figure 2 – Repository Source 3](#_Toc105156308)

[Figure 3 – Git R epositiry 3](#_Toc105156309)

[Figure 4 – Merge Quantity in Better Basket 5](#_Toc105156310)

[Figure 5 - MouseListner in View Class 5](#_Toc105156311)

[Figure 6 - Sound Class in Sound Folder 6](#_Toc105156312)

[Figure 7 - Sort Products in BetterBasket Class 6](#_Toc105156313)

[Figure 8 - Remove Function in BetterBasket Class 7](#_Toc105156314)

[Figure 9 - Junit Testing 7](#_Toc105156315)

[Figure 10 - Junit Testing 8](#_Toc105156316)

[Figure 11 - UseCase Diagram of catshop 8](#_Toc105156317)

[Figure 12 - Customer Client Sequence Diagram 9](#_Toc105156318)

[Figure 13 - Cashier Client Sequence Diagram 9](#_Toc105156319)

[Figure 14 - Java Documentation 10](#_Toc105156320)