Development Report

|  |  |
| --- | --- |
| **Student** | Shubham Satish Kumar Mishra |
| **ID** | W9641416 |
| **Module** | Object Oriented Programming (CIS4037-N-FJ1-2022) |
| **Course** | MSc |

Table of Contents

[1. Discussion of GUI 3](#_Toc134586414)

[1.1 Overview 3](#_Toc134586415)

[1.2 GUI 3](#_Toc134586416)

[1.3 ASCTableModel 4](#_Toc134586417)

[1.5 Testing 5](#_Toc134586418)

[2. Discussion of Enhanced Application 6](#_Toc134586419)

[3. Discussion of Improvement 9](#_Toc134586420)

[4.1 Overview 9](#_Toc134586421)

[4.2 Requirements 9](#_Toc134586422)

[4.3 Implementation 10](#_Toc134586423)

[5. References 11](#_Toc134586424)

# 1. Discussion of GUI

## 1.1 Overview

During the process of creating a GUI version, certain methods in the programme were carried over from the prior work, while others were eliminated since they were no longer useful for the GUI application, as shown in Table 1.

Table 1. Use of Console version components in GUI Version

|  |  |
| --- | --- |
| Console Method | Change for GUI Version |
| Load() | No change, same as task 3. |
| displayMenu() | Completely removed. |
| viewItem() | Completely removed. |
| buyItem() | Totally different, but logic same. |
| addStock() | Totally different, but logic same. |
| Save() | No change, same as task 3 |
| Quit() | No change, same as task 3 |

## 

## 1.2 GUI

The given figure below depicts the basic structure of the GUI after transiting from task 3

The main components of the GUI are:

* Jframe which is the container for GUI and is the main class
* Jtable named ascStockItem, used to display the list of stock items.
* JLabel named photoLabel, used to display the image of the item selected from the list.
* Jpanel named photoPanel, used to display the name of the item selected in the list
* 5 different action buttons used to add stocks or add multiple stock, buy stocks or buy multiple stocks and quit button.

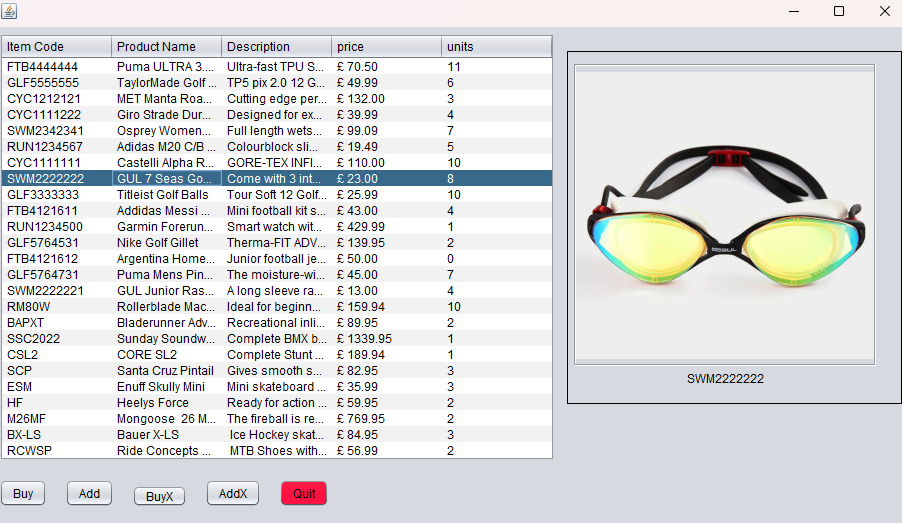


Figure 1: ASC GUI

## 1.3 ASCTableModel

A Jtable is created within the code with the custom data model(ASCTableModel) which imports a data from an Arraylist of ASCStockItem objects. The JTable displays the header names “Item Code”, “Product Name”, “Description”, “Price” and “Units”.

Graphical user interface, text

Description automatically generated

Figure 2

The table also contains a photoPanel and buttons to buy or add items in the stock. The code also reads data from a file and handles exceptions.

## 1.5 Testing

In this, the program is tested whether it stores information of the stock items and keeps it updated after every modification.

The below figure helps to understand the same.

**A picture containing background pattern

Description automatically generated**

Figure 3: Input data File

Figure 10 depicts the data displayed by the JTable. As can be observed the first lines in data file is displayed as first row in the JTable.

Graphical user interface, application, table

Description automatically generated

Figure 4: Populated JTable

The saveFile method is used to save the contents of the array list to a note file. Initially a path object is created for the output of the file and deletes the previous record(if there are any). After that a buffered output stream object is created which uses a loop to iterate through the array list and appends the contents to a string. Additionally, it also formats the price data to include pounds and pences and saves them to the string.

# 2. Discussion of Enhanced Application

2.1 Overview

The enhanced version of the application consist of three additional features.

1. photoPanel
2. buyXButton
3. addXButton

1)photoPanel:

The photoPanel is created with a LineBoard which contains a photoLabel and an itemLabel. Both Labels are given the width of 302 pixels. The photoLabel is where the images of the selected item from the table list will be displayed. Whereas the itemLabel will display the name of the selected item.

Graphical user interface, text, application

Description automatically generated

Figure 5

The displayImage method is used to display the image of the selected item in the list on the GUI. It clears ant text or images in the labels, gets the buffered image for the selected item, and sets the photo label using the photo array list. It also displays the name of the selected stock item on to the itemLabel.

2) buyXItem

To buy a single item unit from the Jtable, buyItem method is used. As soon as a particular row in the table is selected, it retrieves the relevant information about the stock item, such as item code, title, price, and quantity. If the quantity of the selected item is greater than 0 then it decreases the quantity by 1 and updates the table model accordingly. After this, a confirmation message is displayed with information about the item purchased, the price and the remaining stock. If the quantity of the selected row in the table is equal to 0 then it displays a warning message that the item is out of stock. Similarly if quantity is less than 5, it will display a warning message that stock is running low.

Text

Description automatically generated with medium confidence

Figure 6

Coming to the buyXItem method where the only difference when compared to buyItem method is that it allows the user to buy multiple units of the product. When clicked on BuyX button, it displays an input dialog box with a drop down menu to select the number of quantity of products available to purchase. It will decrease the quantity of the selected item by the value which is chosen by the user and update the table model.

3) addXStock.

The addStock method allows the user to add a single unit to the stock of the selected Item in the table. First it checks whether a row is selected in the table or not, and if it is, the method will retrieve all the product related information such as product code, title, price, and quantity. After this, it increases the quantity by 1 and updates the table model with the new quantity value, and displays a message dialog confirming the added stock.

Text

Description automatically generated

Figure 7

Similar to addStock method, addXStock allows the user to add quantity but in multiple units. After the item is selected in the table, it then displays an input dialog prompting the user to enter the number of units to add to the stock.

# 3. Discussion of Improvement

## 4.1 Overview

To improve the security of this application, a login feature could be added to the program. The proposed login feature can also add an element of personalised experience for the users.

## 4.2 Requirements

To implement this feature, it would be necessary to create a user database to store login credentials and access permissions.

## 4.3 Implementation

The ideal way of implementing this feature would be to create a separate login window with dialog box that prompts users for their login details. The program would then identify those details and then match it with user database. The user would get access to the system if the login details match and if not, the user would be prompted to try again.

The given figure below gives a little snippet of how the implementation would look like.

Graphical user interface, text, application, email

Description automatically generated

Figure 8

To test the enhanced version of the program, we could make several user accounts with different access levels and permissions. Then, we could simply try logging in with each of the log in details stored in the database. To verify and check any errors, we could try logging in with invalid credentials to ensure that there are no loopholes or authentication errors in the program.

# 5. References

Minh N.H. (2019) *How to create hyperlink with JLabel in Java Swing*. Available at: https://www.codejava.net/java-se/swing/how-to-create-hyperlink-with-jlabel-in-java-swing (Accessed: 01 May 2023)