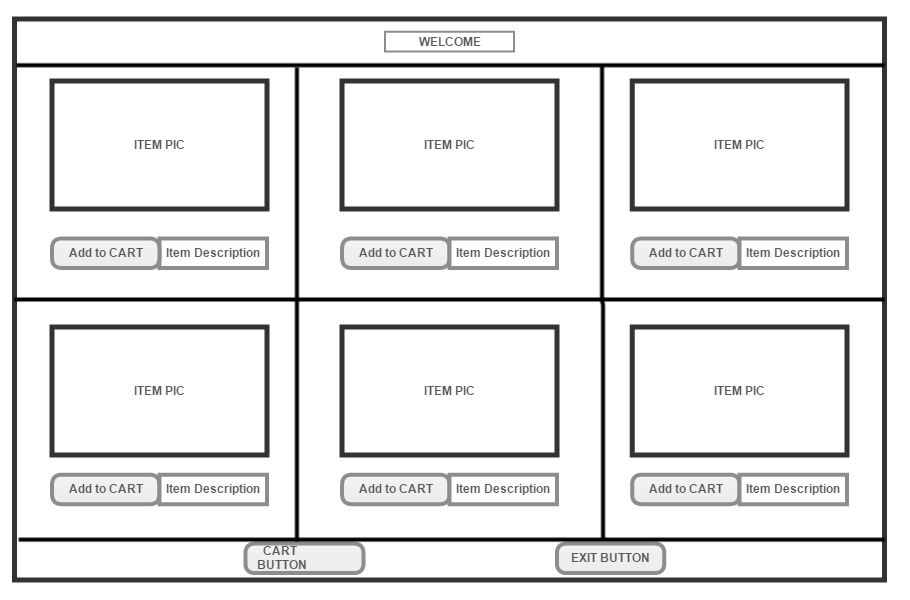
Design & Layout of G.U.I

When planning my GUI for this assessment I went through a few of different ideas before finally deciding on the finished layout that we can see in the wireframe diagram below.

****

The GUI above is a JFrame split into twelve different Panels, I am going to describe how I built the GUI and the different layouts that were used. The finished layout took a little time to get right, many different iterations of code were used before I finally got the panels, buttons and labels where I wanted them in the GUI.

The default layout for the JFrame is Border Layout, so I decided to split the North, Centre and South of the frame into three panels. I did not use the East and West of the JFrame instead I chose to split the centre panel into another three panels of three columns which were laid out using Grid Layout. Each of these panels in the centre were then in turn split into another two panels also using Grid Layout but with two rows and no columns, which gave me the layout I desired. Once the layout of my JFrame was complete I was then ready to start inserting the different elements into the panels.

I had a label in the north panel for a welcome message on the GUI and two buttons in the south panel, the first button was to display the shopping cart and the second button was a button which would exit the GUI.

The main centre panel was now split evenly into six even areas, each area would have a picture of the item for sale, a button to add the item to the shopping cart and also a label with text stating the amount of items left in stock.

The default layout for a panel is flow layout, so I decided this would be fine for the six areas where I would be displaying the items. I made the jpeg images the correct width for each of the panels so that it would help in the layout of the buttons and labels, it automatically pushed the buttons under the images which was perfect for what I wanted. Once I had the layout perfect, my images and buttons all correct I decided to fix the size of the JFrame and disable the resize option, this is simply done with “frame.setResizable(false);”. I decided to do this because after all the hours given getting it right I felt it was better left at the one size, the elements looked good and symmetrical if you like, I think changing the size would be detrimental to the look of the GUI.

My shopping cart was another JFrame independent from the main shop JFrame. I kept it basic enough due to time constraints, the cart had one main panel with eight labels on it to display a thank you note, the total amount of money due and labels to display each different item purchased.

Events handled in my code

In total my GUI has eight buttons, each of these buttons had to have event handling in place, so that when the user clicked the button, something actually happens.

I will describe one of the “add to cart” buttons first as I have six of them in total. Each of the “add to cart” buttons have the same code because they all have the same functionality. However the labels associated with each button have different text due to being different items.

The six “add to cart” buttons had quite a bit of code inside the Action Listener class because there was a lot that needed to happen with every click of the button. First the number of items in stock would decrease by one with every click, also every click would increase my click counter for that button. I needed a click counter so that I could display the numbers of a particular item that was purchased.

There is also some if statements in the button listeners, the first set of if statements are in place to add up the total number of clicks therefore adding the total amount of $ due whilst also reducing the amount of items in stock and setting the correct text to the labels.

The second set of if statements in the listeners were counting the clicks of each button and assigning text to each of the labels in the shopping cart which were displaying the name of items purchased.

I had another two buttons in my GUI. The first was simply an exit button so that when it was clicked the GUI would close, done quite easily with one line of code inside the Action Listener “System.exit(0);”

The final button I had was “Display Shopping Cart”, it is obvious enough what this button does. Inside the handling code for this button I created a new JFrame which is an instance of my cart class which I had earlier created in my program. I add the panel to my frame and the eight labels get added to the panel when the button is clicked.

