**Comp 730 + 830 Final Software Project Update 3**

Team 1: Sean Burwen, Greeshma Srinidhi

**Past Project Progress:**

In the last update, the database part of the project was created and connected to the existing code. This allowed for the listings to be stored persistently and for the “create listings” functionality to be completed. This week, the next step was to begin working on the cart functionality. So far, the cart has been created in Cart.java and functionality is in place for a user to add and remove any items from the cart. The cart will also generate a total price based on the items inside of it. As of right now, the total only updates when the cart initially opens, and this will need to be improved so that it can update when items are removed. When an item is in the cart, it has a different visual appearance thanks to the class methods CartModeOn() and CartModeOff() in Listing.java.

Worked on implementing user authentication and created GUI for the login screen and user should be able to get login screen prompted when we click on login button and hardcoded the username and password in code to check the functionality.

**Sean’s Activities:**

* Created Cart.java
  + Created visual design for the cart, roughly based on the existing design of Browser.java.
  + Dynamically adds listings to the screen just like Browser.java. Listings will only be added if their “carted amount” is not 0.
  + Listings are designed to look different when they are viewed in the cart.
* Updated Listing.java
  + Changed all visual elements from local variables to private class variables. This way they can be dynamically changed by other class methods.
  + Implemented CartModeOn() method to change which visual elements are shown when the cart is open.
  + Implemented CartModeOff() method to do the opposite when the cart is closed.
  + Implemented functionality for “Add to cart” button which increases the “carted amount” by 1.
  + Implemented functionality for “Remove from cart” button which sets the “carted amount” to 0.
* Updated Browser.java
  + Implemented “View Cart” button which opens the cart as a dialog window. The database and listings array are passed through to the cart.
  + When the cart is opened, every listing is put into cart mode. When the cart is closed, every listing is taken out of cart mode.

**Greeshma’s Activities:**

* Updated Browser.java
* Implemented GUI for login screen.
* Implemented “User authentication” without clicking on login button.
* Modified user authentication code in the way that should prompt login screen when clicked on login button.
* Hardcoded username and password to check the functionality.

**Next Steps:**

1. Improve Cart.java
   1. Allow for dynamic updating of the total calculation.
   2. Allow for dynamic updating of the listings visually. Currently, if one is removed, the rest do not fall into place as expected.
2. Begin work on the checkout page.
   1. The checkout page should be a dialog box which acts similarly to the create listings dialog box.
   2. Once the checkout page has been completed, a receipt should be generated and returned. Functionality per-user will come later.
   3. Once the checkout is complete, all purchased items should have their quantity reduced by 1 and the cart should be cleared.
3. Improve User Authentication
   1. Force a user to login before the program can be used. This can be done with a dialog box.
      1. In this dialog box, allow a user to fill out a form to sign up as well.
   2. Store the user account information in another database table.
   3. Add functionality for users to edit their information.

**Difficulties:**

One major difficulty this week was with trying to make the visual elements of listings visible or invisible based on where they were viewed. Creating the CartModeOn and Off methods ended up being the solution to this. Another major difficulty was with trying to get the cart to dynamically update when the listings were removed. This has still not been fixed and will have to be addressed in the coming weeks. Properly handling authentication errors and providing meaningful error messages to users without revealing sensitive information is important for usability and security.