Project 5 Report

Ella Budack, Joseph Getachew, Hunter Specht, Juan Rodriguez

December 11, 2023

0.1 Project Description

Project Structure

For Project 5, our team was tasked with implementing a guided user interface (GUI), concurrency, and socket programming for our marketplace from project 4. In the marketplace, users are sorted into two separate accounts: sellers and buyers. Based on the user's account choice, they have different options and actions available to them. Our marketplace is a computer parts marketplace, and our target audience is computer users.

This marketplace uses files to save data between program uses. This means that a user can still log on and see their data even if the application closes. Each account can also be created, edited, and deleted. For our marketplace, all that is required is a unique username and password. Our marketplace requires users to sign up or create an account before accessing the marketplace.

It is during account creation that the user selects if they want to be a buyer or a seller. Sellers place their products in stores, and they can have any number of stores they like. Customers are allowed to browse and access any store on the marketplace. Some of the core features of our marketplace include sellers, customers, and the marketplace itself. The marketplace displays all of the products that are available, listing the store selling the product, the product's name, and the price of the product. Selecting a specific product will allow the user to see the product's page, which displays the description of the product, and the quantity available. As with other marketplaces, once a customer purchases an item, the quantity available decreases for all other customers.

In the marketplace, sellers have a distinct role from the customers, and thus have different options/actions. Their actions include creating, editing, and deleting products in their stores and viewing each store's sales. Viewing the sales by store also allows sellers to see customer information and revenues from the sales. Customers, on the other hand, can view the marketplace to view product listings. They also can search for certain products by name, store, or description. The ability to sort the marketplace by price or quantity available is also a feature available to the user. Customers also have the option to purchase items from the product's page and review their purchase history.

Along with these core features, our group also chose two additional features. Our first additional feature is a shopping cart. This feature allows customers to add products from different stores and purchase all the items in their cart at once. The shopping cart feature allows sellers to view the number of products that are currently in a customer's shopping cart. They can also see the details of the product in the customer's shopping cart.

The second feature we chose to implement was the files feature. This feature allows customers to export their purchase history to a file titled [username]PurchaseHistory.csv. A .csv file is a comma separated value file and allows for easy imports, exports, and storage of data. Sellers have two additional abilities through the files feature: import and export. Sellers are able specify their file path and upload a .csv file of products for their store(s). Our feature supports file uploads for one store at a time, meaning that all the products in a .csv file must all be placed in the same store. The export feature for sellers allows them to export a list of their store's products. Both customers and sellers import and/or export using .csv files and each file has one line/row per product.

In addition to all of the above, we have also implemented a GUI, concurrency, and socket programming. All of the features and marketplace abilities now appear as pop up windows for the user to interact with. Most of our GUI feature choices and selections as drop down items. Our GUI allows for easier visualization of our marketplace. Users no longer have to look at plain text in the terminal. For instance, when the user first launches the application a window pops up welcoming them to the HardwareHarbor Marketplace. After the user acknowledges this message (clicking "ok" button), they will be prompted to select one of three options from a drop down menu. They can select "create account", "sign in", or "exit". From here the user selects their choice and GUI's corresponding to the selection appear. Concurrency is what allows multiple users to access the marketplace at the same time. Threading is used to allow our program to handle each user and the actions they select. This means that two users can look at the marketplace simultaneously, without having to wait for the first user to log out. Our server can handle multiple clients at once. To prevent race conditions (for example, two customers trying to buy the last of an item), we have implemented gate-keeping. This means that certain actions in our program are limited so only one user can perform the action at a time (such as two customers trying to purchase one item). Socket programming (Network IO) allows the marketplace to be accessed from any computer/machine. This allows users to access our marketplace from anywhere, meaning they are not just limited to the machine the code was written on. This is very essential for allowing everyone to access the HardwareHarbor marketplace.

Structure Justification

Functionality was a primary focus of our design choices in regard to installing a GUI. This, coupled with ease of implementation, guided many of the decisions that were made.

Hence, as opposed to creating complex GUIs for every process in our program, we decided to use the simpler, less beautiful JOptionPane GUIs. This is one of the primary design decisions that we made as a group, as it would save time to focus on actually integrating GUIs into the program at the cost of appearance. However, when necessary, our program still uses a few complex GUIs where we as a group found to be most beneficial. Thus, we coupled ease of implementation and the quality of the final product in order to create the best overall program.

Furthermore, in regard to network IO and concurrency, our program models Professor Dunsmore's EchoServer class, where the server creates a new thread each time that a new client connects. This allows for both functions to be integrated easily, and it allowed our group to allot an increased amount of time for other tasks. Our client receives short Strings, which are sent from the methods called by the server's processing, and then displays particular GUIs depending on the String.

Lastly, the methods of our program retain a very logical structure. The first method handles sign in, where a user's username and password are entered, processed, and matched to an account that has been written to a file. It then returns a user. After that, our program determines whether the returned user is a buyer or a seller and proceeds to display the corresponding menu. Each menu displays a menu of actions, such as adding an item to a store, to which the user is invited to select one. Each action is performed, and the menu is re-displayed. This continues until the user decides to logout.

Our program is structured around three core .txt files: one to save sellers' data (such as their stores and listings), one to save buyers' data (such as their carts and purchase histories), and one to store login data (such as usernames and passwords). As users traverse through the program, the files are updated each time a new change is made that would impact other users. Thus, this allows the program to update whenever a page is reloaded, as it now reads from an updated document. This functionality is coupled with our methods, which handle all of the processing on the server side so that the client's computer is not burdened with any sort of processing. Lastly, the methods, when necessary, send simple Strings to the client, which then are assigned to a certain GUI which is soon displayed. These three integral parts of our program form its structure, which is spread across several different classes, each assigned to network IO, processing, or displaying a GUI. Our program is much better structured now than it was at the end of Project 4, which was something that all of us had mentioned when discussing ways to improve heading into Project 5.

0.2 Team Member Contributions

0.2.1 Ella Budack

My Contributions

My contributions include setting up the repository for project 5, starting the concurrency and socket/network IO programming, helping organize the division of work, and writing the report, readme file, and a few test cases. Also, I did some general debugging. Again, I was designated team leader by my team.

My first contributions included meeting with my team and GTA to discuss project 4 communication issues. After hearing the group's input on how to improve communication and work-load and flow for project 5, I texted the group chat to set up a timeline and work division. I suggested having the debugging and revisions for project 4 done by the Monday we returned from Thanksgiving break. While we did not get as much done as we would like, we did get to work the week we got back.

I started designing a few of the GUI's we needed, namely the 'welcome' and 'create, sign up, exit' GUI's. I first created these to use when making sure the client server (socket programming) code I was writing worked. I also helped start coding the client, server, and server thread files. While this way of coding was not used (we just used a client and server file with various other classes), it helped set some of the project work in motion.

In addition, I did some general debugging and created the report and read me files. I reused the report from project 4 and updated it with the project 5 information. I did the same with the read me file.

What I Would Do Differently

If given the opportunity to redo this project, I would start earlier. I was not as productive as I wanted to be over Thanksgiving break. I would also have suggested more, but shorter, meetings where we discuss what we are working on and what we plan to do for the next few days. I would also like to have started earlier so I could better think out how our program was going to work. If I had more time, I would have made more complex GUIs for the user to interact with. I also would have gone back and made more thorough comments and test cases if I had the time. Getting my group to meet more regularly would have also been a priority. Overall, there are some minor things I would like to change if given the chance (such as time management and communication), however, for the first big college project I have worked on I think it went pretty well. I will take my experiences and regrets from this project and keep them in the back of my mind as I continue to collaborate in the future.

0.2.2 Joseph Getachew

My Contributions

One of my contributions to Project 5 was firstly being an outstanding team member and communicating efficiently as well as being punctual to meetings. I also proposed the idea of how we could split up work so that git merges and commits could happen seamlessly. Another achievement was developing the buyer GUI that allowed for concurrent client messages to be sent to the server. Additionally, I also provided test cases for the buyer and seller menu so that we can ensure our program is reliable. To enhance the user-friendly experience, I dedicated time to making a refined login GUI where I aimed to make for a smooth entry point for users. Another one of my contributions involves the presentation in which I had made the FAQ sections providing realistic questions that a recruiter might ask, the testing slide where I justified the logic of our test cases, the project pitch which is the most influential for potential recruiters, and the introduction of the team. My involvement in both the technical aspect and the presentation goes to show how committed I am to even splitting the workload and having success in our project.

What I Would Do Differently

If I could go back I would have our group meeting before break and finish all the issues with Project 4 then so when we came back for break we would be ready for Project 5 and have 2 weeks to complete it. We should have assigned work more efficiently, but we did well as it is hard to split up work without knowing how you can split it up and not interfere with one another. I believe we should have also tried for a more aesthetically pleasing interface. I also believe that we should have tried to resolve some problems that were also occurring in Project 4 with communication as that was a major problem when trying to divide up code. A better way of communicating like regular checkups and commits would have allowed us to work fluidly so there was no confusion and everyone would have time to understand the uploaded code.

0.2.3 Hunter Specht

My Contributions

My primary roles in the development of Project 5 were coding new functionality, revising old functionality, and debugging. I am responsible for the structure of our program, and I coded many of the classes that are present in the project, and I dedicated a large amount of time to fixing Project 4, as it did not perform well under specific circumstances, such as edge cases, that have now been rectified. I am responsible for the way that our server connects and communicates with the client, as well as the fact that the server creates a new thread upon bearing a new user. Lastly, I designed the way in which the program communicates with the GUI, using short, simple Strings to tell the client to display a certain frame.

In addition to the physical project, I have also helped with a small part of the project report and the presentation. I wrote the part of the report that details the structure and provides justification for it, as I am the most familiar with the inner workings of our program. Lastly, I contributed by staying in contact with the other members of my group and attending regular meetings where we discussed the progress of the project.

What I Would Do Differently

Looking back, the only improvement that I wish I had made would have been leaving more time to debug and improve the final product. I was not extremely stretched for time, but rather just enough to where I was forced to code when I would have rather been doing something else, like study for finals. In regard to our team's communication and planning, they were both significantly better than Project 4. I think that the very structured nature of Project 5 (Network IO, concurrency, GUI) allowed us to determine a much fairer division of labor that ultimately resulted in the best possible project. Thus, I would not change much, though it would have been smart for me to start coding some of the major aspects of Project 5 over Thanksgiving Break, when I was home relaxing and not doing any other schoolwork, as opposed to this past week, when I had finals to study for.

0.2.4 Juan Rodriguez

My Contributions

Throughout Project 5, I assisted the group in many ways, whether it be coding, or even other external things throughout the project. When it comes to coding, I mainly collaborated on GUIs with other members of the group, as well as debugging parts of what was left over to fix from Project 4. For example, I worked on ViewStoresGUI, PhGUI, MarketplaceGUI, CartItemsGUI and I also helped debug other GUIs, including my original Statistics Class, and then I also helped debug parts of the entire project as a whole. Just like all the other members of the group, one of more important roles I played was simply being a member and good teammate to the others, that role means that I made sure attend meetings when I was available, I aided in helping plan out what was to be done or deciding when to meet, and just talking to other members in general and discussing the code/project. While this is expected of being a single member within a bigger group, it is still important to do so. Also just like all other members of the group, we worked on the project report, presentation and the other non-coding parts of the project.

What I Would Do Differently

There is maybe some things I would have done differently in Project 5, just like with Project 4. I think that there was still a slight bit of the same problems as with project 4 regarding time management and communication, but I can definitely attribute those problems to being of my own fault, and not the group's. These were personal problems, that I was able to overcome, and not distract from the rest of the member's ability to work properly. These problems were also much smaller than before, and while some of these problems still pertained, I also think that the group ended up working much better together this time, as compared to previously stated in the other report. There could have maybe been some better splitting up of work as always but overall, this project didn't have many extreme or crazy problems or other things I would have done differently.