Using the model-view-controller design pattern which helped divide theproject apart . The controller package consists of 6 classes from which the cart controller controls the shopping cart, the category controller manages the item categories, DB controller manages the CSV and storage aspect. The user controller takes care of the user and the store takes care of the store. The model part mainly has admin, category item store, and user. I have the view which is tables and most of the user interface is there. I created three separate packages and added the classes related to each in them. This helped me in creating a lot of helper functions and reusing code from different areas. In my midterm implementation, I mentioned I will want to use singleton and factory which I now realize is a bad decision as I feel that MVC is a hybrid pattern that builds on multiple patterns, compared to singleton as that's how I started first it became hard to keep track of them and it was taking a lot of time.

The MVC pattern is based on the observer where the view is updated by itself compared to the previous design, here if a modification is made it automatically changes the rest of it aswell. We are able to send data between objects and modify things at any time allowing more freedom.

I created a few helper methods as it helped in not repeating the same piece of code again and again. My midterm project was created in a time crunch, I feel that one of the things I cut back on in my original design was saying the cart is independent, which I realized was wrong, as the shopping cart is related to everything and precisely one of the most important things in the entire project. Just similar to how a normal shopping cart works the same way we have a user who can add, modify and delete their cart.

The main problems with the midterm design were that in the UML and in my report I treated shopping cart as an individual object which is actually wrong technically due to the way it should be implemented along with the items I followed the MVC architecture so its all realtaional and depended on each other.

My database was originally treated like an object but I found that using a database controller and putting functions inside that made much more sense. As this was my first time trying to create an MVC style application I feel that alot of things could have been improved and updated aswell.

I feel that I also implemented the search deature differently then what I had originally planned to create an aboject that searches through instead I made a searchable interface. Another thing I did different was that I noticed in the midterm diagram there was a flaw in the user class, so thats why i created the controller part and I tried my best to separate the UI and logic from each other as much as I could.