**Critiques**

* When comparing the artefacts in our analysis stage to what we had with our result, we noticed that the layout of the project remained the same. However, with the modifications we made into our end result made the system helped us move into a better direction of what we wish to do with our project if we were to take this idea further.
* During our first iteration, we decided on the concept of hard-coding in our products and customers into the system. This idea was not efficient due to the fact that in later iterations, we would have the idea of adding more products or customers to the system, making this concept difficult to implement. We created separate text files to store all the information in regards to orders, customers, etc. This implementation of the text file databases lead to our project being more scalable and reliable come our end result.
* Originally, we were incorporating all of our classes into one default package. We were then made clear of that we needed to separate our system models from our UI classes. After going over the MVC approach, we then separated our system into 3 separate packages, each demonstrating the concept of the Model, View and Control layers respectively. The separation of responsibilities allows more flexibility with our code in later iterations and easier to then implement our design patterns into the system.
* In the beginning, the basis of our project was focused more on implementing the system based on the use cases. However, later on in the project, we then decided to focus more on including the design patterns into our system in order to reduce coupling and allowing the system to run more efficiently.
* Later on in the iterations, we included the concept of designing PCs and laptops. When purchasing the pieces to create these machines, the system requires certain subclasses in order create them. We thought of assigning each individual piece in the database their respected value, but this method would be too time consuming and complicated as the admin has the option of creating new products to be stored in the database. The Factory Design pattern was then implemented into the code to help of the assignment of the subclasses in the database. Using this design pattern demonstrated high cohesion and took away an unneeded tasks happening in the DataLayer class.