**Vending Machine MVC model explanation of implementation**

1. Model – This is the data-storage of the program and in the vending machine this is the ArrayList of Item objects that stores every item that is read/written into the program or updated when the user uses the business logic. The arraylist is used consequently throughout the entire program to be able to have a precise count of the items in the vending machine at the start of the program as well as the end. The price of the item in the Arraylist is very important to be able to calculate the correct change for the user. The stock is also important to be able to let the user know if they can purchase an item and remove an item from the stock count each time that happens.
2. View – The view in this program is the Main.java class that will display the entire vending machine using the methods from ItemController class. It will allow the user to be able to see the items in the vending machine, be able to input some money and then pick a item to buy. The program will then decide what to do depending on how the user interacts with it by calling methods from the ItemController class. This view is simply for abstracting the logic from the user and letting them only interact with the front-end of the program. Everything the user does is checked and tested with the other classes to ensure accurate data is manipulated.
3. Controller class – this is the class that implements most of the business logic of the program and acts as a middle-ground between the user and the data storage. This is one of the most important classes that will allow the program to read files,write files, create items, select and buy items, update the stock, display the items and display an audit log with DateTime of when the Vending machine starts reading the new item stock and ends by writing the updated stock to the file.

Service Layer: This part behaves like the controller class and handles most of the logic of the program. Service layer allows for abstraction and it is the controller interface that is implemented for being able to create the functionality of the program that the user will be interacting with. It allows for ease of access to the data and the user. Service layer is used to display the items from the arraylist, used to read the items into a file and write them, it is used to check the stock of an item and decide whether a user can buy the specified program or not.

Unit testing: Unit testing is for isolating code methods into different tests to verify if they have worked as intended. For example, in my program I tested if the stock was updated accordingly. If items can be bought and change can be given etc. It makes it easier to verify what functionality works in the program and what doesn’t, sort-of like debugging or proving that the program can work as a whole. But the isolation part enables that each part can work on its own before being connected with the other parts. So that if you need to fix or update any code you can know exactly what must be updated without wasting time verifying the entire program and checking what must be fixed.