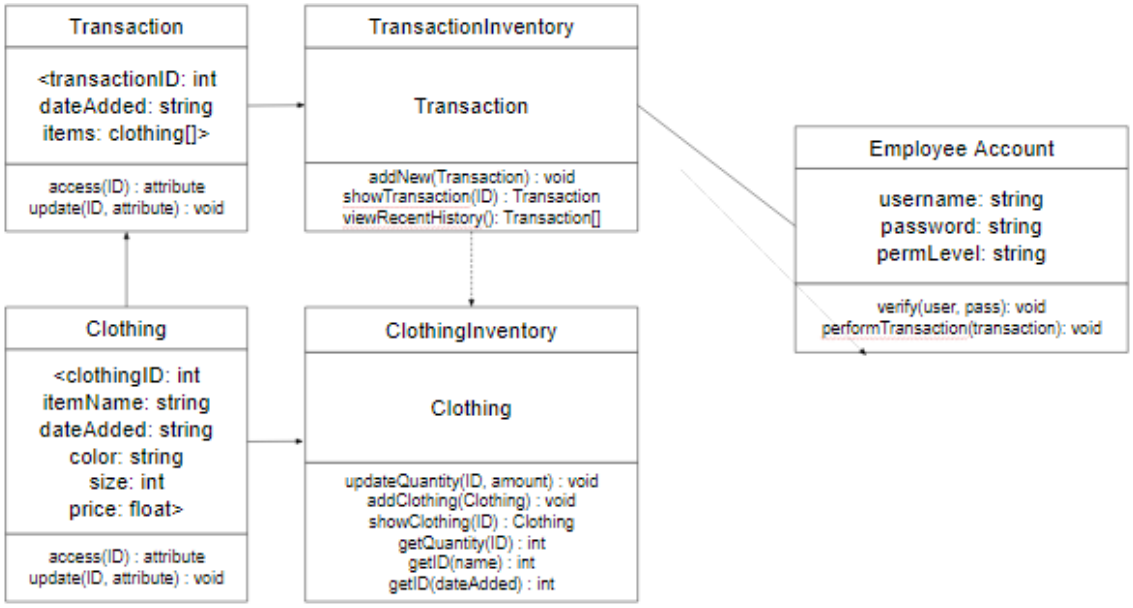
SDS: Test Plan

Software Design Specification Diagram:



Verification Test Plan:

Test Set 1:

This set of tests involves the clothing related functions and aspects of our software.
Unit Test:
We are testing the functionality of the getID function in the ClothingInventory class.

This function is meant to take a string as an input, and use a search algorithm to match what clothing item in the inventory has the name that was inputted.

We will be using a dummy clothing item for this test that has the name “Shoe” with ID 115.

By entering getID(“Shoe”), The program will handle the verification of this input being a string, and the function should return a value of “115”.

Functional Test:
We are testing the general functionality of Clothing and ClothingInventory.

This phase of testing will involve adding new clothing items to the inventory. We will add two new dummy items, “Sock” and “Hat” to the inventory, once the inventory already contains 150 items (starting at 0).

Both of the new items will contain all of their pertinent information excluding the ID and dateAdded.

Upon entering addClothing(“Sock”, (relevant input),), the ClothingInventory will have a new item with an ID of 150, the date added being the current date and time, and a name of “Sock”.

Upon entering addClothing(“Hat, (relevant input),), ClothingInventory will have a new item with an ID of 151, the date added, and a name of “Hat”.

System Test:
For the system test, we will be testing the locally stored ClothingInventory in the store’s primary computer and its connection to the tablets in the store that are operated by the employees.

Every time a piece of clothing is searched for by an employee or used in a transaction, the tablet will query the primary computer for the information it needs.

Whenever a transaction alters the quantity of clothing in the inventory, or if the inventory is otherwise changed by an employee during management, the tablet device informs the primary computer of the change, which updates its inventory, and (if necessary) updates information on all tablets currently displaying inventory quantities.

Test Set 2:

This is a test set focused on performing the transaction handling of our software.

Unit Test:

For our second test set, we will first do a unit test of a Transaction. We will make a new Transaction and call its update(). Transactions are randomly assigned a unique ID; for example, it might be 42839515 in this case. A clothing we might update it with is a “black and white dress” labeled “monochrome_dress”. We will take this ID, “42839515”, and call Transaction.update(42839515, monochrome_dress) and set it to a transaction with a dress.

Functional Test:

Now, we will test the functionality of the TransactionInventory. We will add the previously created Transaction with ID “42839515”. We will call TransactionInventory.add() and use Transaction.access(42839515) as a parameter. This should add this previously created Transaction into the TransactionInventory.

System Test:

For our system test, first we will log in using our admin credentials from an Employee account. This will be tested on the employee devices and they will log in. For instance, a set of employee credentials could be “user: yotsubabestgirl pass:quints25252” and they will log in

using that. Then, they will perform a transaction with clothing id “monochrome dress” and just input all the transaction info . They might want to verify it was added, so they will click on view history, which calls viewRecentHistory() and displays all that data. Finally, this test will finish with the employee logging out, which calls the logout() function finishing the test.