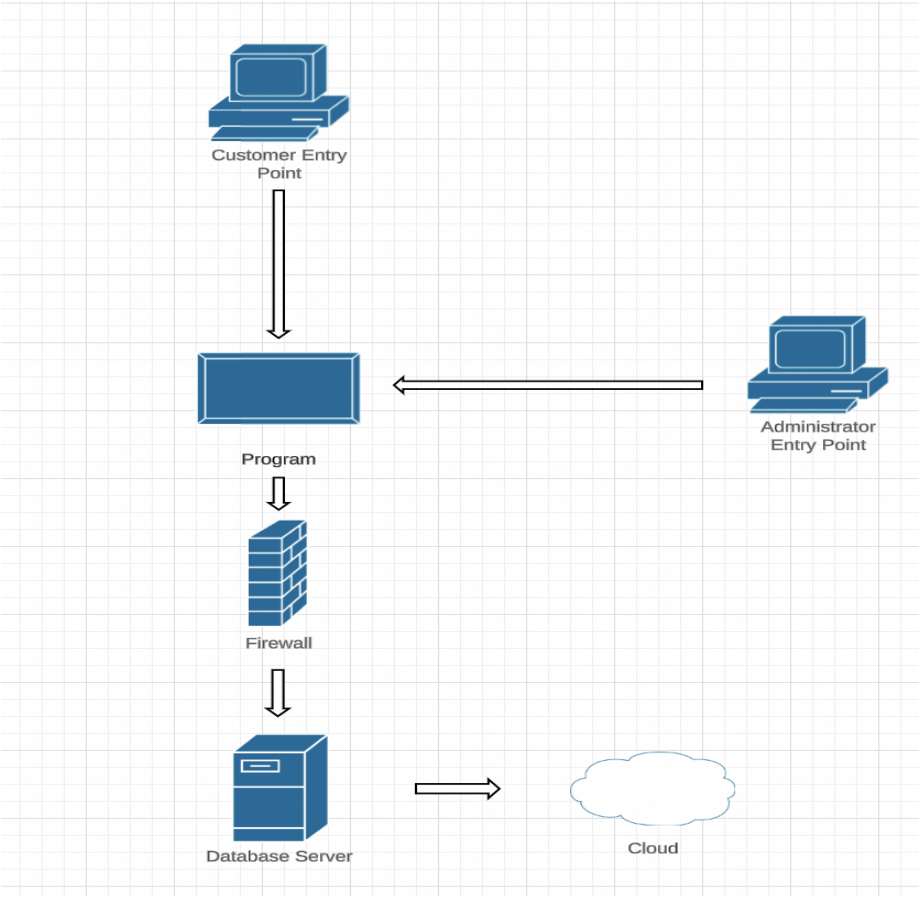
Software Requirement Specification for Clothing Store POS System

By: Shreyas Basu and Salvador De La Torre

1 Introduction

This document outlines the design and requirements for a storewide kiosk system that will initially be launched in one store and eventually spread store-wide. The purpose of this system is to make it easier for customers, employees, and the owners to shop/work in the store. It's needed because the current system is inefficient, not helpful/intuitive, and an upgrade is long overdue. This system-wide change will affect employees in that it will make their job easier. It will also affect customers by making the shopping experience next-level, more fun, and unique. Overall, it will be in the best interest for the company to get this system. This document is outlined into four sections, the introduction, user requirements, system requirements, and other information.

2 Architectural Diagram:

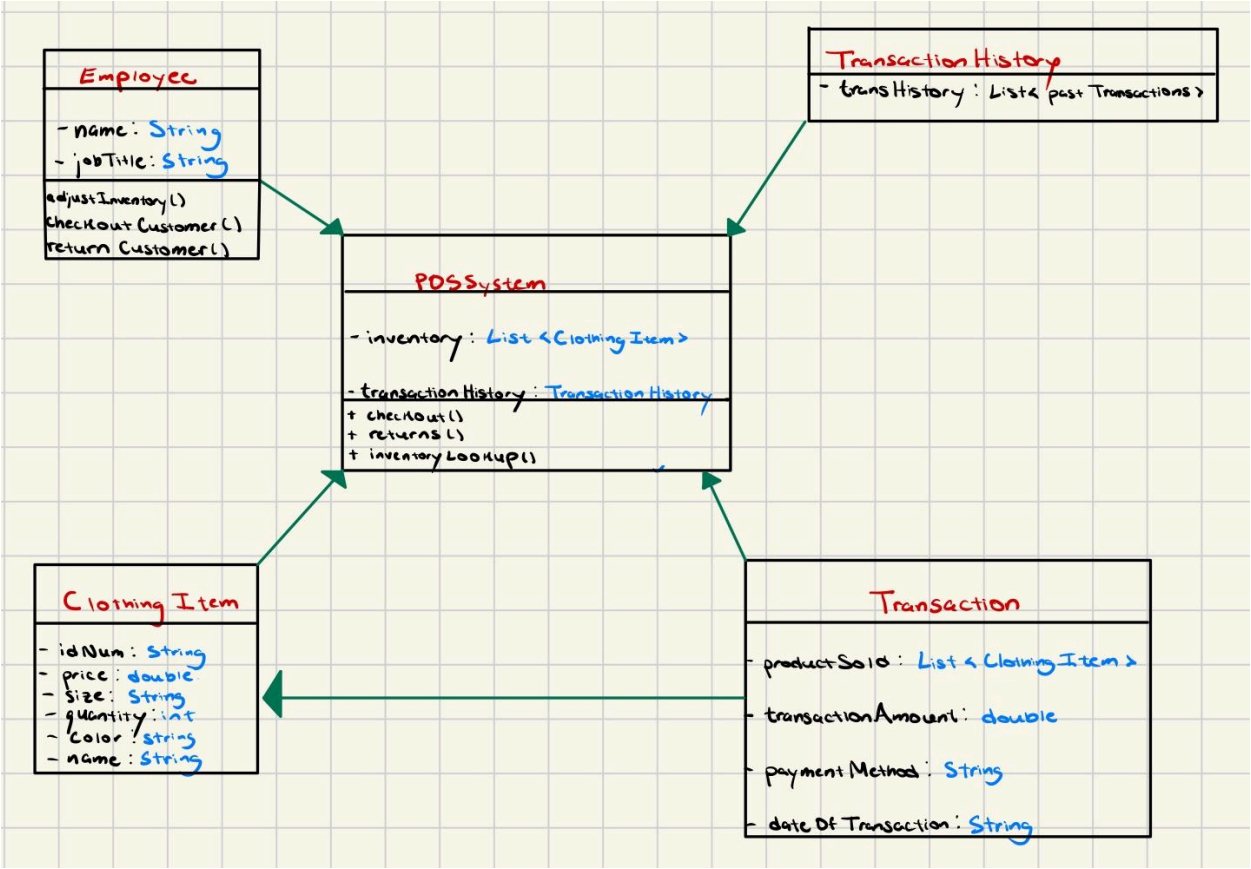


Description

This Software Architectural Diagram is precise and concise. There are two entry points, the user and administrator entries. Once correct credentials are entered, it goes to the

program where they do what they need to do. For customers, it would be handing the items to the employee for them to be scanned and checked out. For administrators, it would be actions such as price matching, returns, account setup, seeing purchase history, etc. These programs store data in a database server which is then backed up by the cloud.

3 UML Diagram:



Description

The UML Class diagram for this system is quite simple and does not contain too many components.

The main class is the **POSSystem** class, which is meant to represent the main system and is the center of our diagram. This class contains two variables, inventory and transactionHistory. inventory is a variable that will hold in a list, all the available

inventory at the store and the transactionHistory variable is an object of the TransactionHistory class which is meant to hold all the transaction history. The checkout() function is used to process any transactions where a customer is making a purchase. The return function is used to process any transactions where a customer is making a return. And lastly, the inventoryLookup() function is meant to be able to lookup what that store or other stores using the same system have available in their inventory.

The next class is the **TransactionHistory** class, which only has one variable which is transHistory that holds in a list all the information related to past transactions.

Next would be the **Employee** class, which has variable name and jobTitle. Both variables in this class are string variables, and the name variable will hold the name of the employee and jobTitle will hold the jobTitle that that specific employee has. In this class are the adjustInventory(), checkoutCustomer(), and returnCustomer() functions. The adjustInventory() function will allow the employee to adjust the store's inventory after either a purchase or a return. The checkoutCustomer() function will allow the employee to ring up customers when they are ready to checkout, and the returnCustomer() function will allow the employee to process returns that a customer may have.

The **Transaction** class has the productSold variable which has a list of all clothingItems that have been sold. The transactionAmount variable is a double variable that represents the total cost of a particular transaction. The paymentMethod variable is a string variable that is meant to represent the type of payment that a customer is using for their transaction. Lastly the dateOfTransaction variable is a string variable that will hold the date that a particular transaction was done.

The last class being the **ClothingItem** class is a class that holds all the descriptive properties of the clothing items. The idNum variable is a string that holds the unique ID number of a certain product. The size variable is a string variable that represents what size that particular product is. The price variable is a double variable that represents the

cost of that particular product. The quantity variable is an int variable that holds the total number of that particular product in the store's inventory. The color variable is a string variable that holds the color of that particular product. And lastly, the name variable is a string variable that holds the name of that particular product.

4 Development Plan and Timeline

Partitioning of Tasks

As a group we partitioned the tasks to an equal 50/50 to make it as equal and fair as possible. The tasks that we needed to complete included the development of a title page, description of the system, creation of an architecture and UML diagram, describing with detail the diagrams that we created, and lastly our development plans and timeline for this assignment.

Team Member Responsibilities

Shreyas worked on the introduction portion of the assignment, creating the architecture diagram, and describing the architecture diagram that was created.

Salvador worked on the creation of the UML diagram, describing the UML diagram that was created, and the development plan and timeline portion of the assignment

Timeline

Each team member's task responsibilities as well as the assignment as a whole needs to be completed by March 8th @11:59 PM, the assigned deadline.