Software Design Specification
For
ThriftWerks: Point-of-Sale System

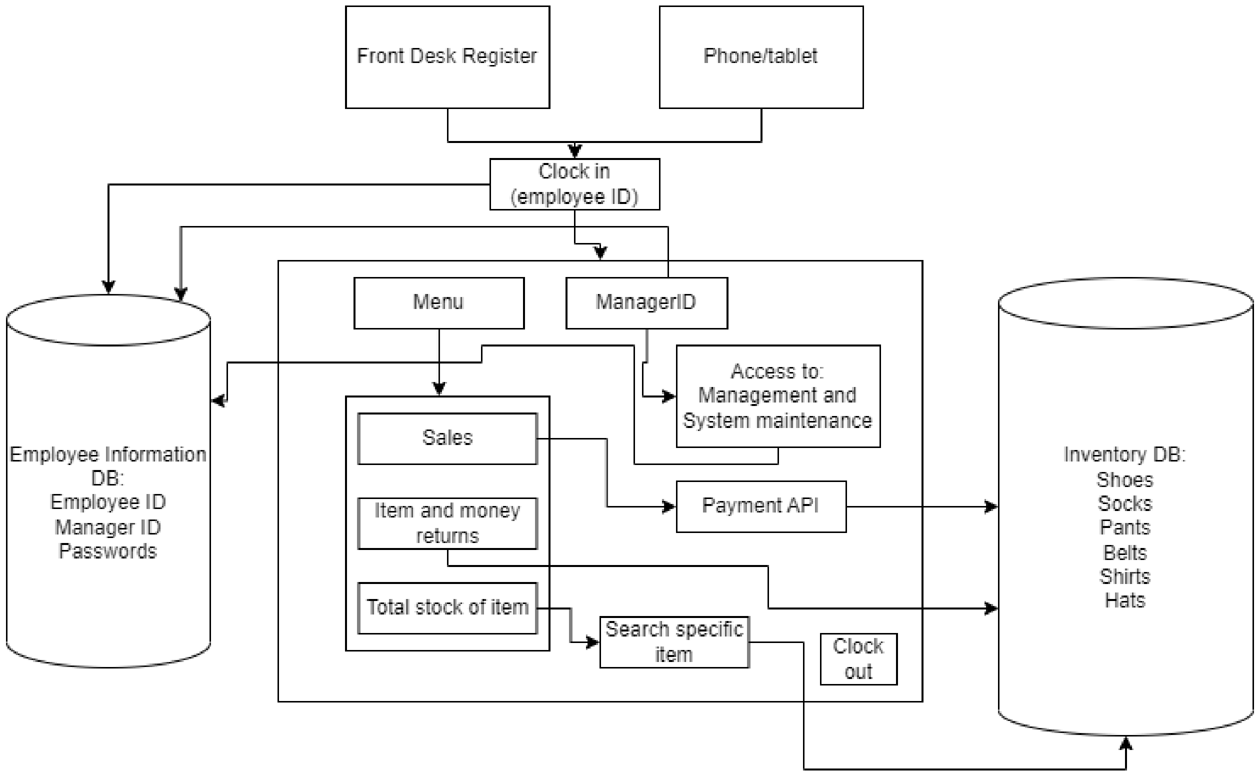
Team members:
Ricardo Lemus
Jane Ho
David Kaauwai

System Description

This software, which will be known as “ThriftWerks”, is a point-of-sale system that will facilitate the work of retail employees, most notably with processing sales, refunds, and inventory searching. The system can be accessed at front desk registers or through phones and tablets. From there, a login screen will appear where an employee ID and password must be inputted to proceed. The system will not only determine whether the account information matches what exists in the Employee Information Database, but will check if the ID matches that of a higher-level employee, such as a manager, so that more functions inaccessible by regular employees may be unlocked. Once the inputted information is confirmed to be valid, the system will direct the user to the main menu, which displays options of “Sales”, “Item and Money Returns”, and “Total Stock of Item”, all of which will utilize the Inventory Database. Higher-level employees will also have the options of “Management” and “System Maintenance”. “Sales” will use a payment API with the Inventory DB to carry out customer transactions when they purchase any item(s), and “Item and Money Returns” will directly access said DB to process customer refunds. “Total Stock of Item” will help search for specific items in the Inventory DB.

Software Architecture Overview

Architectural diagram of all major components:



The above software architecture diagram shows all the major components of the system we are designing. It shows what happens when a certain user logs in (manager or employee), two databases that store different information and specifically what information, and the different actions an employee or manager can do such as giving refunds or checking inventory on an item. Below are different characteristics that go more in depth on what each component does in the software architecture diagram.

Description of the software architecture diagram:

Front desk register: Connects to the "Clock in" screen which is to verify that the user is an employee of the store.

Phone/tablet: Also connects to the “Clock in” screen to verify they are an employee.

Clock in (Employee ID): Extends to the “Employee Information DB” to see if the ID and password match an account. Once logged in the “Menu” is accessible.

Employee Information DB: Stores employees/managers credentials.

Manager ID: Extends to the “Employee Information DB” to verify they are a manager and see if an ID and password match. Manager ID also extends to the special tools managers have.

Access to: Management and system maintenance: Allows managers to access moderations, employee management and system maintenance.

Inventory DB: Database where all stock of clothing is kept at.

Menu: Extends to sales, item and money returns (refunds) and total stock of an item (inventory).

Sales: Goes into “Payment API” which is used to process purchase transactions and it also extends into the “Inventory DB” to update the stock on an item.

Payment API: Used to process transactions.

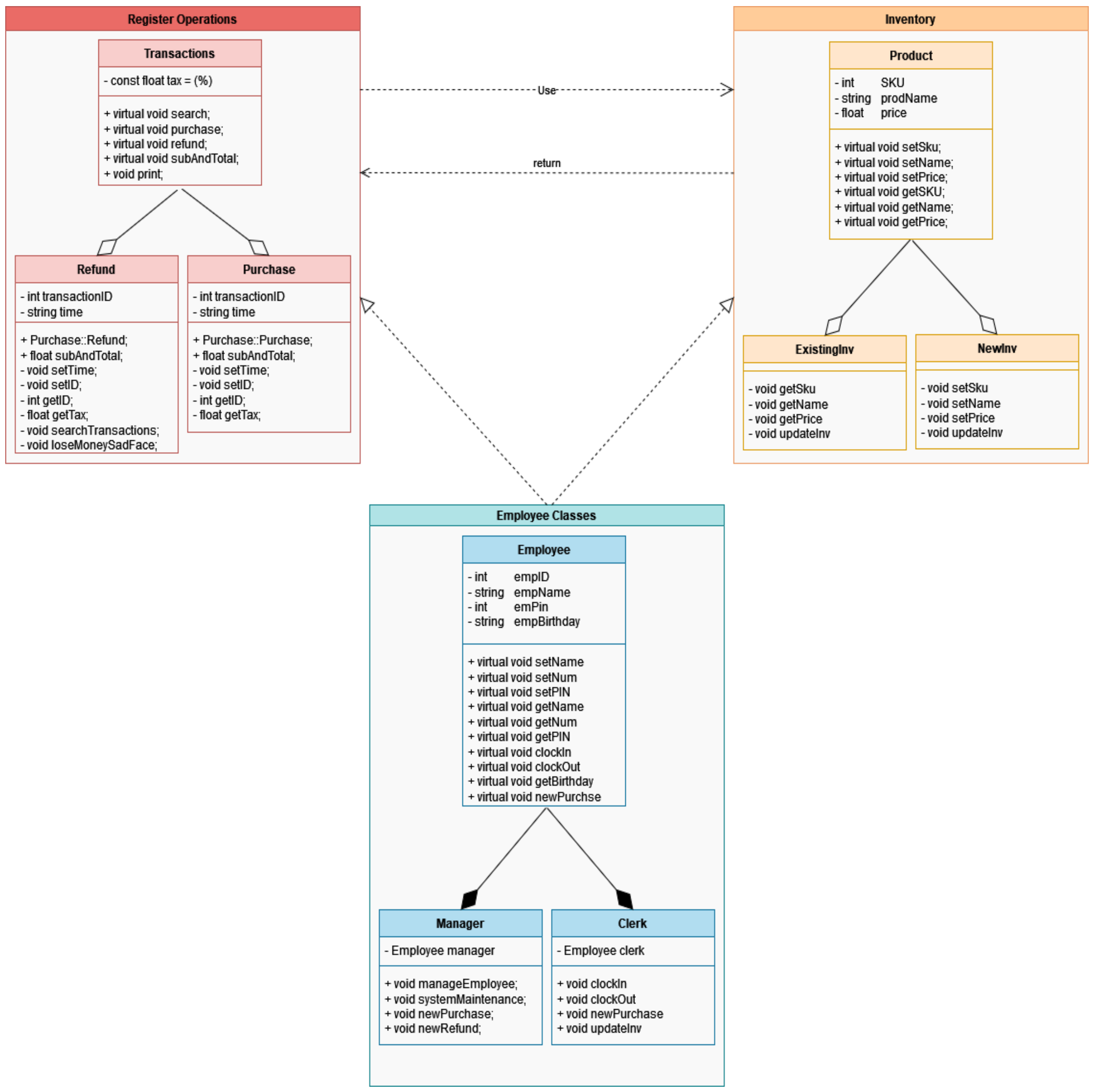
Item and money returns: Extend into “Inventory DB” to update the stock when an item is returned.

Total stock of an item: Goes into “Search specific item” which searches in the database of “Inventory DB” for a particular item.

Search specific item: Searches inside “Inventory DB” for an item the user wants to find.

Clock out: Logs a user out to prevent any non employee from tampering with the system.

UML Class Diagram



The image above is our UML diagram. The software will be divided into the three major sections of Register Operations, Inventory, and Employee Classes. Register Operations contains classes, attributes, and operations pertaining to calculating transaction totals and storing transaction information. Inventory serves to hold information for all products in the store, retrieving existing product information and creating new ones. Employee Classes will have employee information that is necessary for identification and account login, along with functions that they have access to.

Description of classes:

Transactions: Base class for common transactions made by employees, which includes processing purchases and refunds, searching inventory, and calculating purchase/refund totals.

Refund: Child class of Transactions, includes functions for searching refunds made, getting/setting IDs of refunds, and setting times for them.

Purchase: Child class of Transactions, includes functions for getting/setting IDs for purchases made and setting times for them.

Product: Base class for basic product information that is in the inventory, such as SKU codes, product names, and prices.

ExistingInv: Retrieves information of an existing inventory item, important for searches.

NewInv: Sets information for a new inventory item.

Employee: Base class for employee account information, which includes ID numbers, names, PINs, and birthdays.

Manager: Child class of Employee, gives access to manager-only functions like system maintenance and managing employee information.

Clerk: Child class of Employee, includes functions for clocking in and out for work.

Description of attributes:

const float tax (Transactions): Will be accessed to calculate purchase/refund totals with tax.

int transactionID (Refund/Purchase): The ID assigned to a specific refund/purchase.

string time (Refund/Purchase): The time at which the refund/purchase was made.

int SKU (Product): The product's identifying code, will mainly be used for inventory searches.

string prodName (Product): The name of a product, can also be used for searching.

float price (Product): The price of a product.

int empID (Employee): The ID number of an employee, will be needed for login.

string empName (Employee): The name of an employee.

int empPIN (Employee): The PIN/password of an employee, will be needed for login.

string empBirthday (Employee): The birth date of an employee.

Employee manager (Manager): An object of an employee, specifically for managers. Should have access to manager-specific functions.

Employee clerk (Clerk): An object of an employee, specifically for clerks.

Description of operations/connectors:

The software is composed of three categories of classes, the primary interaction of ThriftWerks is done through register operations by Employee classes, which instantiates all transactions. Transactions are either refunds or purchases. Those are implemented by employee accounts, which are instantiated and constructed from the Employee abstract class. Employees may also instantiate new products through the NewInv class. Transactions fetch and update existing inventory through the ExistingInv class.

Development Plan and Timeline

Partitioning of tasks:

Implementation (1-2 months): Each person will work on programming one of the three sections shown in the UML class diagram (Register Operations, Inventory, Employee Classes) and parts of the SWA diagram relating to it. Jane will work on Register Operations, the menu of options after login) and the manager-only functions like system maintenance and employee management. Adam will work on Inventory, the Inventory DB, and the item searching function. Ricardo will work on Employee Classes, the Employee DB, and the login/logout function.

Testing (2-3 months): Test the implementation for verification and validation, ideally each section is tested by all three people to garner as many unique perspectives as possible to minimize the amount of problems slipping through in the final product. (Alternatively: each person can do a different kind of testing, e.g. black-box, white-box)

Release and Maintenance (continuous): Send out the product for use and continuously fix bugs and add features when needed.

Team member responsibilities:

This project was split evenly between the three of us.

Ricardo Lemus:
- Work on the Employee class
- Finish the Employee DB and the login/logout functionality

Jane Ho:
- Work on the Register Operations
- Menu options and manager only functions

David Kaauwai:
- Will finish the Inventory
- Inventory Database and item searching functionality