Clothing Store Point of Sale System

Group 5

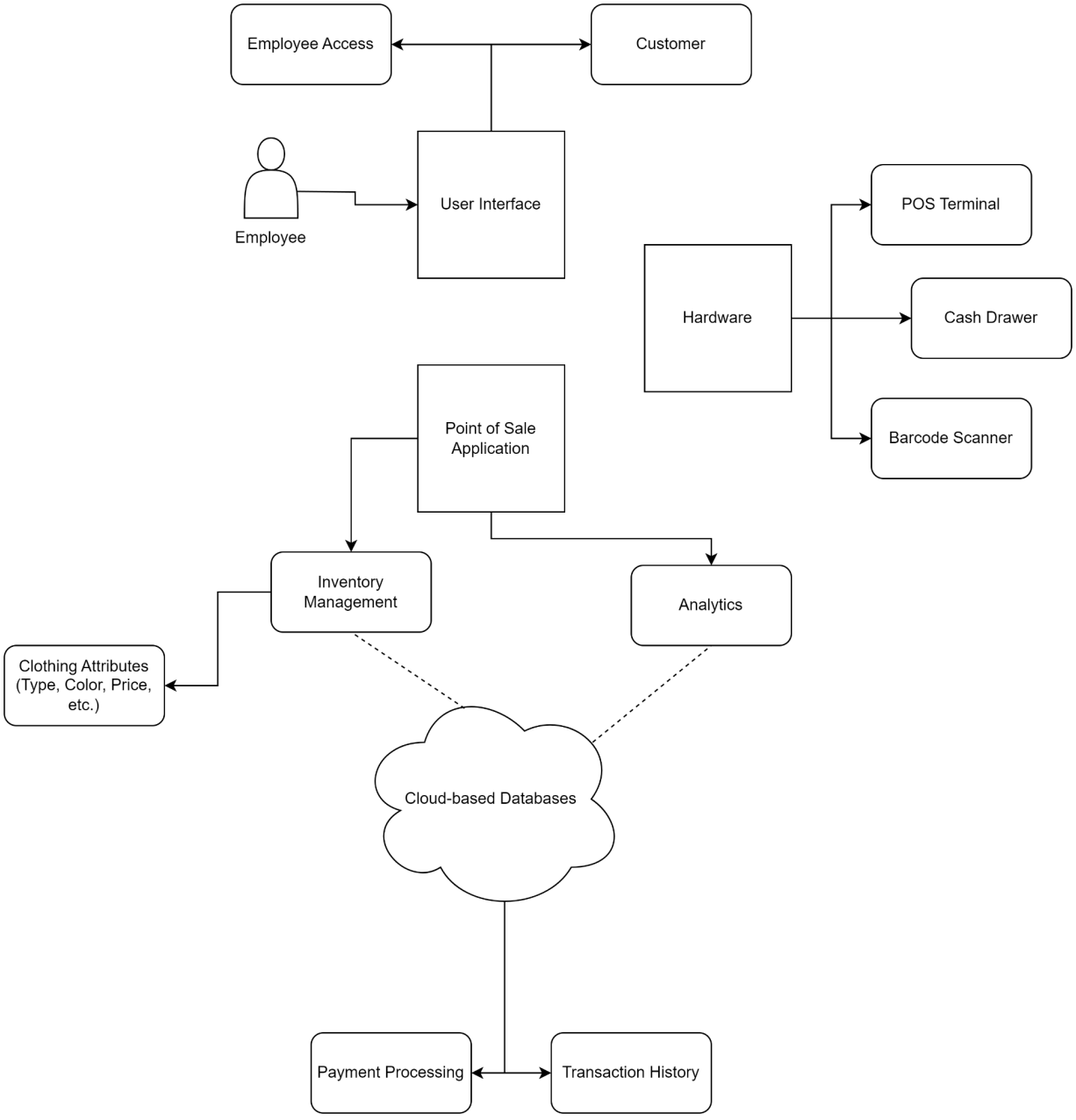
Miguel Melo Ochoa
Jose Hernandez Sanchez
Gabriel Sawaya

System Description

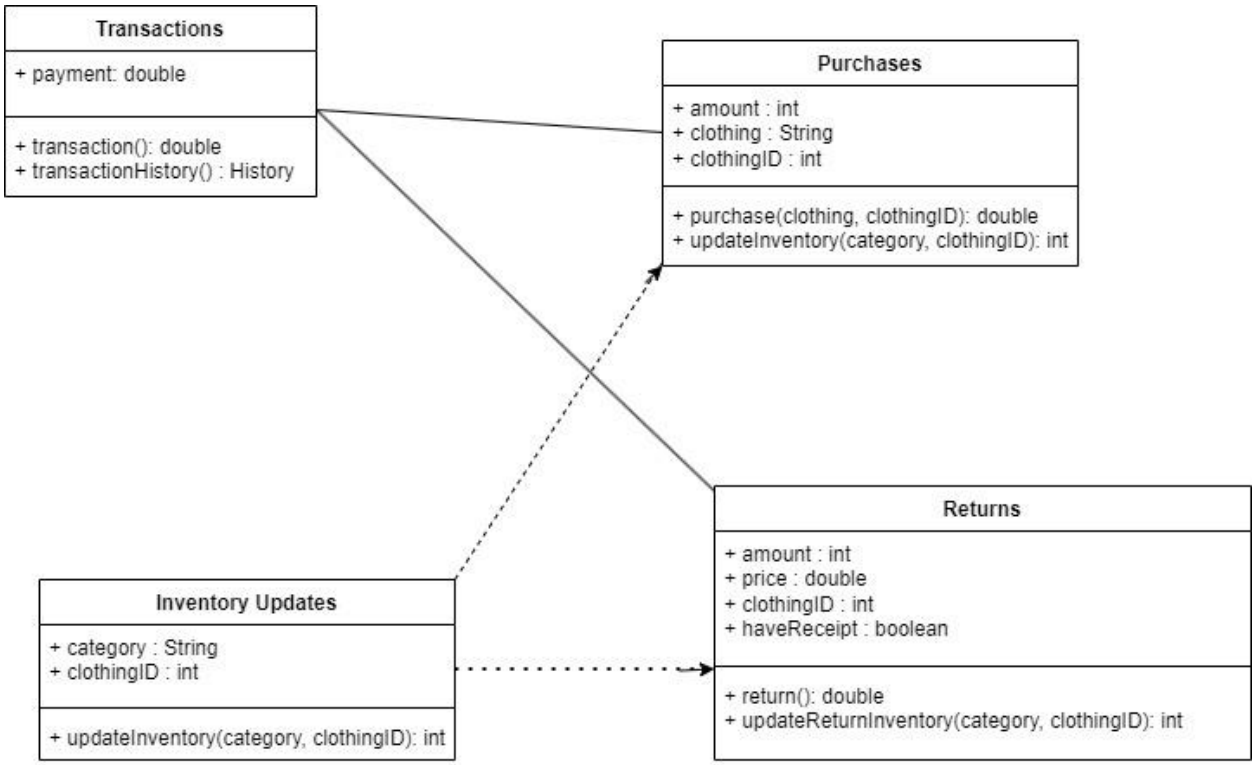
The Point of Sale (POS) system is designed to allow employees to accept payments, process transactions, and keep track of inventory throughout the store. The system is aimed at making sales efficient for employees, keeping items located in the designated store organized through inventory updates, and handling transactions such as purchases and returns. The POS system will also aid in keeping track of sales through analytical reports which can be accessed by higher-ranked employees. Employees will be able to scan clothing items via barcode scanner, once scanned, the item will be added towards the customers total transaction. Once all items are scanned, the employee can then process the payment via add-on card reader then can accept credit, debit, or cash, which will utilize a register to store/take out change. Once items have been scanned and purchased, the inventory of those items will be adjusted accordingly. Returns will work similarly, with the customer's returned item(s) being scanned as a return, payment being refunded, and inventory being updated.

Software Architecture Overview

Architectural diagram of all major components



UML Class Diagram



Description of classes

- Transactions
    - This class will be in charge of covering the transactions of purchase. Should be able to see transaction history. It should be able to access the Purchases and Returns class
- Inventory Updates
    - This class will cover the inventory update for the shop. This class will allow the employee to update the inventory. Will print a list of all the items in inventory. It will also be able to print out the amount of one item. Should have an add and delete function.
- Purchases
    - This will administer the purchase process. This includes processing payment, removing the purchased item from inventory, and processing the cost.
- Returns
    - This will administer the returns. Will process the return of payment and add the returned item to the Return Inventory.

Description of attributes

- Transactions:
    - payment : double, payment is the money used to purchase clothing which is why its type double.
- Inventory Updates:
    - -category : String, category is what the type of clothing is associated with.
    - -clothingID : int, clothingID is the ID/tag of the clothing to know what and which clothing it is.
- Purchases:
    - -amount : int, amount is how many purchases of clothing there are.
    - -clothing : String, clothing is the type of clothing(e.g.shirt,pants,jacket).
    - -clothingID : int, clothingID is the ID/tag of the clothing to know what and which clothing it is.
- Returns:
    - -amount : int, amount is how many pieces of clothing were purchased.
    - -price : double, price is the price of each clothing item.
    - -clothingID : int, clothingID is the ID/tag of the clothing to know what and which clothing it is.
    - -haveReceipt : boolean, haveReceipt checks if the customer has the receipt to show proof of purchasing the item.

Description of operations

- Transactions:
    - transaction(): double - Function used to process a transaction. When a transaction is complete, returns money used for the transaction (i.e. bill/receipt).
    - transactionHistory() - Stores transaction result in transaction history

- Inventory Updates:
    - updateInventory(category:String, clothingID:int): int - This function will take in the item's attribute(category) and ID as parameters and should return a count of the number of those item(s) that are left in inventory.

- Purchases:
    - purchase(clothing:String, clothingID:int): double - This function will take in the item's attribute(category) and ID as parameters and should return the total price of all item's as a double.
    - updateInventory(category:String, clothingID:int): int - This function will take in the item's attribute(category) and ID as parameters and should return a count of the number of those item(s) that are left in inventory.

- Returns:
    - return(): double - This function returns the dollar amount returned to the customer after a return
    - updateReturnInventory(category:String, clothingID:int): int - This function will take in the item's attribute(category) and ID as parameters and should return a count of the number of those item(s) that are left in inventory.

Development plan and timeline

Partitioning of Tasks

1. Gather Requirements
    a. First we speak with the client to obtain as much information about the system as possible. We want to collect all possible information, including functional and non-functional requirements for the proposed system. Not only would we need to obtain software related information, but also hardware requirements too. Thus an on-site meeting would be best in order to cover as much space as possible before we develop the system and know if we have any physical limitations produced by the location of the store or client. This phase of the project is estimated to take about two weeks.
2. Design/Develop the Software
    a. Next we would begin designing the system based on what we have gathered from the client. After finalizing the architecture of the system, we would develop the software and set up the necessary databases. We would also create safeguards to protect the databases from intrusions. This could take a month to a few months of work depending on the size of the team we are able to contract.
3. Implementation
    a. Once we have identified all core functionalities of the system, we will begin development of these functionalities. We will incorporate all of our design decisions, software implementations, and code structure into a fully functional system prototype, which will then be used for testing. This phase must take substantial time, as we want to make sure we are as thorough as possible with our code to ensure testing goes with minimal risks/failures to the system.
4. Verify & Validate the Software
    a. After the prototype has been developed, we will begin testing the product in its most practical use. We would execute all of our tests, including unit, integration,

and system testing. We will also conduct our user acceptance testing and gather feedback in order to refine the system. This phase of the project will take two weeks.

5. Maintenance/Other
    a. The system will be deployed for client use. The client will be able to obtain first-hand experience with the system and let us know if we have met their software expectations. We will also monitor the system after its initial rollout to make sure that there are no issues with the system. If the client would like us to continue working on the software then we could continue working with them and address any other concerns that they may have.

Team Member Responsibilities

Jose - Design/Develop the Software, implementation
Miguel - Implementation, Maintenance/Other
Gabriel - Gathering requirements, implementation, verify and validate the software

Verification Test Plan

Test Set 1:

Unit Test: Barcode Scanner Functionality - The barcode scanner functionality works to check if the barcode scanner for the hardware functions correctly when scanning a clothing's barcode. The barcode scanner takes in the input of the clothing's barcode and it should output all the details of the clothing such as the clothing ID, category, and price. The details of the clothing needs to have the correct data type as well. The clothing ID needs to be an int, category needs to be string, and the price needs to be an int. The expected output for the barcode scanner are the clothing's correct details and shouldn't be another clothing's details or any other invalid output. The barcode scanner test will make sure a clothing item's barcode displays the correct information for each one.

Functional Test: Payment Processing - The payment processing test is used to verify payment for a sale. The main features that will be tested here are the card reader, cash drawer, and POS terminal. Since the POS accepts both card and cash as forms of payment, both methods will be tested as part of the payment process test. In both cases, verifying the test will begin with the employee scanning all of the customers items via barcode scanner and/or manual entry of the items. Once all items have been scanned, the employee will proceed to the checkout screen on the POS terminal. The checkout screen will display to the employee which form of payment the customers will be using as either cash or credit/debit. For cash sales, the employee will select

cash. If cash is selected, the employee will input the amount of cash given by the customer. The cash drawer will open automatically and the POS terminal will then display the amount of change to give the customer back. For card sales, the employee will select credit/debit. If credit/debit is selected, the POS terminal will automatically connect with the payment processing database that will then connect with the databases linked with the customers bank that is associated with the card. Once the employee inserts the customer's card into the card reader, the clothing store's payment processing database will verify with the bank's database that there is money available on the card for the requested transaction. If there are sufficient funds, the POS terminal will display that the transaction is complete, and a receipt will be printed automatically for the customer's records. If there are not sufficient funds on the customer's card or the customer's card is denied, the transaction will not be processed and the POS terminal will prompt for another payment.

System Test: Complete Transaction - This test will demonstrate that the system that we have developed works accordingly. This will verify that all the individual pieces work together. This System test would also utilize the other tests in order to complete the system check. The test will start by running the Barcode Scanner Functionality unit test which will provide a boolean value that would either continue the test or stop it and notify the user that it has failed the first test. If the test passes we would continue and administer the Payment Process unit test. The test would then output a boolean value that will indicate if the test passed or failed. If it passed then it would continue the test and if it failed the test would stop and let the user know that it failed the Payment Process test. We would then verify the inventory updates properly. At this point we have scanned an object and processed the payment for it. For this test we are verifying that the inventory number decreased since the customer is buying an item. We will make sure that the correct item and amount of that item has decreased. The test will post and verify that the correct item is being updated. If it passes all the aforementioned tests, then it would post a successful test. If it fails it will give the user a notice that it failed this stage of the testing process or an earlier stage.

Test Set 2:

Unit Test: Price Calculation - The price calculation functionality works to check if the payment transactions are working correctly to allow a clothing product to be bought. Price calculation will be a test where it will check if the payment transaction from the buyer is the correct amount needed to purchase a clothing item. The price calculation will take input of the buyer’s payment to see if it was enough to purchase the clothing item. The test should output the price of the clothing item, the buyer's transaction payment, and the amount paid and if it was enough. Of course, the payment must be data type double as currency. The price calculation test will make sure purchasing clothing items function correctly.

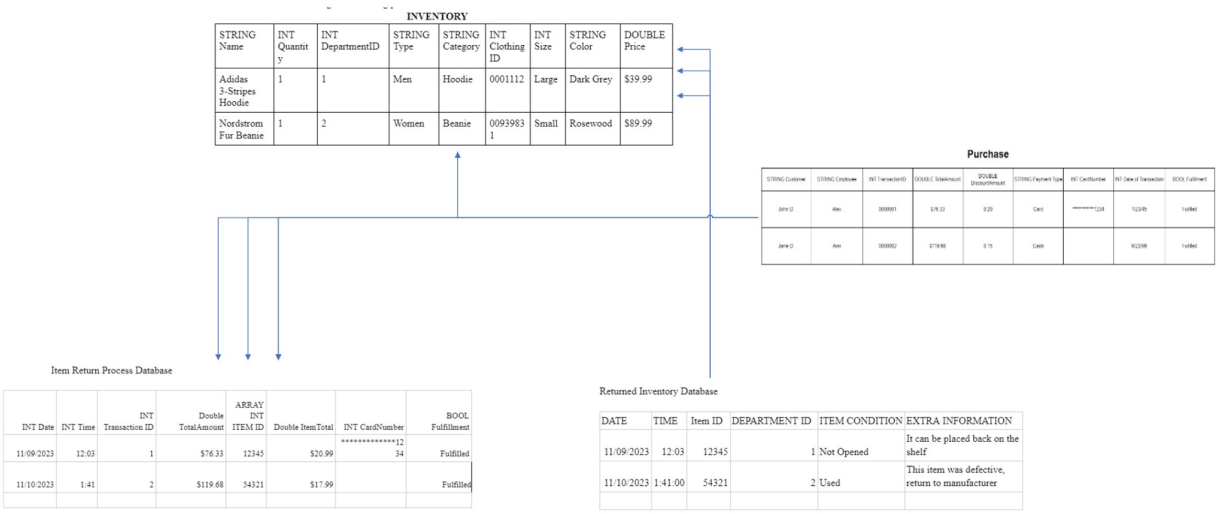
Functional Test: Discount Calculation - The discount calculation test verifies that discounts will be applied properly to sales. The store will determine which items will be at a discount and then obtain discount codes for those items. The discount codes will be obtained either from the discounted brand’s company or the marketing department of the clothing store. Discount codes will be applied either by scanning the code using the barcode scanner or entering in the discount

code manually through the POS terminal. After the employee has scanned all of the customers items, the employee will scan or enter in the discount code manually. Once the discount code has been scanned, the POS will ring up the discount linked with the item and subtract the appropriate amount from the item's original price.

System Test: Return Transaction - This test will ensure that the Return Transaction process works how it is supposed to. This will be accomplished by initiating the Barcode Scanner test which shall provide the correct information of the object being returned. If the test identifies that the wrong item is being shown then it has failed this portion of the test. If it passed the Barcode Scanner test then it will continue to the return payment stage. This stage will identify that the correct amount of money has been counted and being provided in the method in which it was initially made, i.e cash, card, etc. If it fails then the test stops here and posts that this portion fails. If both of these pass then we continue with the last portion of the system which involves updating the inventory. However it will not update the inventory database but instead add it to another database that will be for returned items. This test will make sure that the correct inventory is being updated with the correct categories, items, and amount. If it is identified that it isn't updating the correct inventory or not the correct item nor amount, then it has failed and will post that it failed at this stage. If it passes then its posts that it passed are all stages. Before any of this can begin we must make sure to provide a receipt which should be able to prove that the item scanned is on the receipt. If not then the test will automatically fail stating that the item cannot be found on receipt provided. The receipt has to be provided because it's how we will be able to know which item needs to be returned and how much money should be returned.

Data Management Plan

Data Management Strategy



Inventory Database

The inventory database is responsible for all stored inventory data in the clothing store. It organizes all clothing attributes such as the quantity, ID, color, and price. To be able to organize all the data regarding clothing inventory, the inventory database will efficiently keep track of all details of the inventory. The inventory database will make sure all clothing products are safe, secured, and organized in a perfectly structured inventory to ensure the clothing-point store will do their best in storing their clothing products.

INVENTORY

STRING Name	INT Quantity	INT DepartmentID	STRING Type	STRING Category	INT Clothing ID	INT Size	STRING Color	DOUBLE Price
Adidas 3-Stripes Hoodie	1	1	Men	Hoodie	0001112	Large	Dark Grey	$39.99
Nordstrom Fur Beanie	1	2	Women	Beanie	00939831	Small	Rosewood	$89.99

STRING Name - Name of the clothing product.

INT Quantity - Amount that is stored in the inventory.
INT Department Identification Number - This number will be used to identify what department the item that is being returned is a part of. This will be the key that connects the inventory database to the Return Database
STRING Type - Verification for what the clothing is designed for.
STRING Category - Displays what category the clothing is. (e.g. shirt, sweater, pants, etc).
INT ClothingID - Identifier for each piece of clothing.
INT Size - Size of the clothing product.
STRING Color - Color of the clothing product.
DOUBLE Price - Amount needed to purchase the clothing product.

Purchase Database

The Purchase database will track sales, including the details of the purchase such as the customer, employee who processed the purchase, the transaction ID, the total amount of the transaction, the discount applied to the purchase, the payment type, card information if the purchase was made with a credit/debit card, the date of the transaction, and status of fulfillment of the purchase.

Purchase

STRING Customer	STRING Employee	INT TransactionID	DOUBLE TotalAmount	DOUBLE DiscountAmount	STRING Payment Type	INT CardNumber	INT Date of Transaction	BOOL Fufillment
John D.	Alex	0000001	$76.33	0.20	Card	**********1234	1/23/45	Fufilled
Jane D.	Ann	0000002	$119.68	0.15	Cash		9/22/99	Fufilled

STRING Customer - Name of the customer
STRING Employee - Name of the employee who processed the purchase
INT TransactionID - Unique identifier of the purchase, also provided in receipt to reference the purchase
DOUBLE TotalAmount - Total dollar amount of purchase
DOUBLE DiscountAmount - Discount applied to the purchase
STRING PaymentType - Type of payment used (cash, credit/debit)
INT CardNumber - Card information if transaction was made with card. If a cash sale was made, the number will be negative to express that the transaction was made with cash as cash info cannot be stored in the database.
INT DateOfTransaction - Date that the purchase took place

BOOL Fulfillment - True if purchase was successful, false if purchase was unsuccessful. Unsuccessful purchase would be if the card was declined or the customer does not have any sufficient funds to complete the transaction.

Return Inventory Database

The Return Inventory database will store the information of returned items. This includes the item, date of return, condition, and any other details that may pertain to the subject for return. Before any item can be returned the customer will need to provide a receipt which will help find the reference to the purchase. Once the system verifies that the receipt is connected to a past purchase in our directory, we will scan the item(s) that will be returned. We then add those items to a return database that will be set up similar to the Inventory database. The returned items will later be pushed to another database that will work alongside the inventory database. As previously stated this database will have employees be able to see what the item is, when it was returned, its department, and if it can be put back on the shelf or not. If It can be put back on the shelf then it uses the department ID to move it to the correct Inventory database.

Returned Inventory Database

DATE	TIME	Item ID	DEPARTMENT ID	ITEM CONDITION	EXTRA INFORMATION
11/09/2023	12:03	12345	1	Not Used	It can be placed back on the shelf
11/10/2023	1:41:00	54321	2	Used	This item was defective, return to manufacturer

STRING Date - Date of return of the item
INT Time - Time of day that this return was processed
INT Item Identification Number - This number will be the way the system is able to know what exact Item is being processed.
INT Department Identification Number - This number will be used to identify what department the item that is being returned is a part of. This will be the key that connects the inventory database to the Return Database
STRING Item Condition - This will store the information that pertains to the state of the returned items.
String Extra Information - This will let the employees add any extra information about the item.

The Return Database will process the Return operation. Using the The TransactionID that will be acquired using the Receipt from the initial purchase the system will verify that the purchase was made within the allotted time and it was initially purchased at our store or associated stores. It will pull most of the information connected to the Transaction ID to this temporary database. Once the desired Items for return are scanned then we will add up the cost of each item and store it in the ItemTotal Database. We then process the return using the previous method. If it fails then it will prompt us.

Item Return Process Database

INT Date	INT Time	INT Transaction ID	Double TotalAmount	ARRAY INT ITEM ID	Double ItemTotal	INT CardNumber	BOOL Fulfillment
11/09/2023	12:03	1	$76.33	12345	$20.99	**********1234	Fulfilled
11/10/2023	1:41	2	$119.68	54321	$17.99		Fulfilled

STRING Date - Temporary Return Receipt Database
INT Time - Time of day that this return was processed
INT TransactionID - Unique identifier of the purchase, also provided in receipt to reference the purchase
DOUBLE TotalAmount - Total dollar amount of purchase
INT Item Identification Number - This number will be the way the system is able to know what exact Item is being processed.
DOUBLE ItemTotal - Total amount of the Scanned item(s).
INT CardNumber - Card information if transaction was made with card. If a cash sale was made, the number will be negative to express that the transaction was made with cash as cash info cannot be stored in the database.
BOOL Fulfillment - True if purchase was successful, false if purchase was unsuccessful. Unsuccessful purchase would be if the card was declined or the customer does not have any sufficient funds to complete the transaction.

Trade Off Discussion

SQL design is a relational database and NoSQL is a non-relational database. SQL design connects with each other. NoSQL design has connections with each other but not the same as a SQL design. Objectively, noSQL is easier to extend. Data can simply be added on, meaning the noSQL data can do better in that field. In the end, we agreed that SQL was the best way to store and manage our data since it's easier to be able to connect multiple databases using one or multiple pieces of information. It’s scaleable and so we can keep adding to the databases. Subjectively speaking, SQL is a lot more organized than other databases that we looked at.