Mid-Module Assignment System Design

Selecting Candidate Classes (List of Classes)

I started this part by looking at the scenario and by going through the nouns to see if they fall into groups (Liang, 2003) and then highlighting these.

When an **item** is scanned, it is added to the **customers'** purchases in a **virtual shopping basket**.

Items can either be scanned with the **barcode reader** or they can be **weighed** using the **integrated scales**.

When the customer has finished scanning their items, they will be given the option to scan their **loyalty card**, if they have one.

The customer then purchases their items using several different **payment methods** (e.g., **cash, card, ApplePay, AndroidPay, vouchers, loyalty points**).

Supermarket staff will have the ability to **override transactions** where required, for example if a **price** has been scanned incorrectly, they will also be responsible for checking the **age** of a customer for restricted items.

A **completed transaction** will update the supermarkets **stock control systems** and generate an **alert** for **warehouse staff** if either the **shelves should be replenished** or if **stock needs to be reordered**.

I then separated out the candidate classes and decided to shorten some names as this would make the coding element potentially easier.

Candidate Classes 1	Candidate Classes 2
• Customer • Items • Virtual Shopping Basket • Barcode Reader • Price • Age • Weight • Integrated Scales • Loyalty Card • Loyalty Points • Payment Methods • Payment • Cashcard • Applepay • Androidpay • Vouchers • Staff • Supermarket Staff	• **Customer** • **Items** • **Virtual Shopping Basket - shortened to Basket** • **Barcode Reader** • ~~Price~~ – An instance of items • ~~Age~~ - An instance of items • ~~Weight~~ - An instance of Integrated Scales • **Integrated Scales** • ~~Loyalty Card~~ - An instance of Payment • ~~Loyalty Points~~ - An instance of Loyalty Card • ~~Payment Methods~~ - An instance of Payment • **Payment** • **Cash** • **Cashcard** • **Applepay**

• Completed Transaction • Supermarket • Override Transactions • Stock Control System • Alert • Warehouse Staff • Shelves • Stock	• **Androidpay** • **Vouchers** • **Staff** • ~~Supermarket Staff~~ - An instance of Staff • **Completed Transaction – shortened to Transaction** • ~~Supermarket~~ – Outside the scope of the system • ~~Override Transactions -~~ An instance of Transactions • ~~Stock Control System~~ - An instance of Staff • ~~Alert~~ - An instance of Stock Control System • ~~Warehouse Staff~~ - An instance of Staff • ~~Refill Shelves~~ - An instance of Stock Control System • ~~Re-order Stock~~ - An instance of Stock Control System

Classes

- **Customer**
- **Items**
- **Basket**
- **Barcode Reader**
- **Integrated Scales**
- **Payment**
- **Cash**
- **Cashcard**
- **Applepay**
- **Androidpay**
- **Vouchers**
- **Transaction**
- **Staff**
- **Stock**

List of the Relationships between the Classes

Customer

- A **customer** can **scan** many items using t**he Barcode Reader**
- A **customer** can **weigh** many items item using the **Integrated Scales**
- A **customer** can choose to scan and use their **Loyalty Card**
- A **customer** can choose to pay using many different **Payment Methods**

Staff

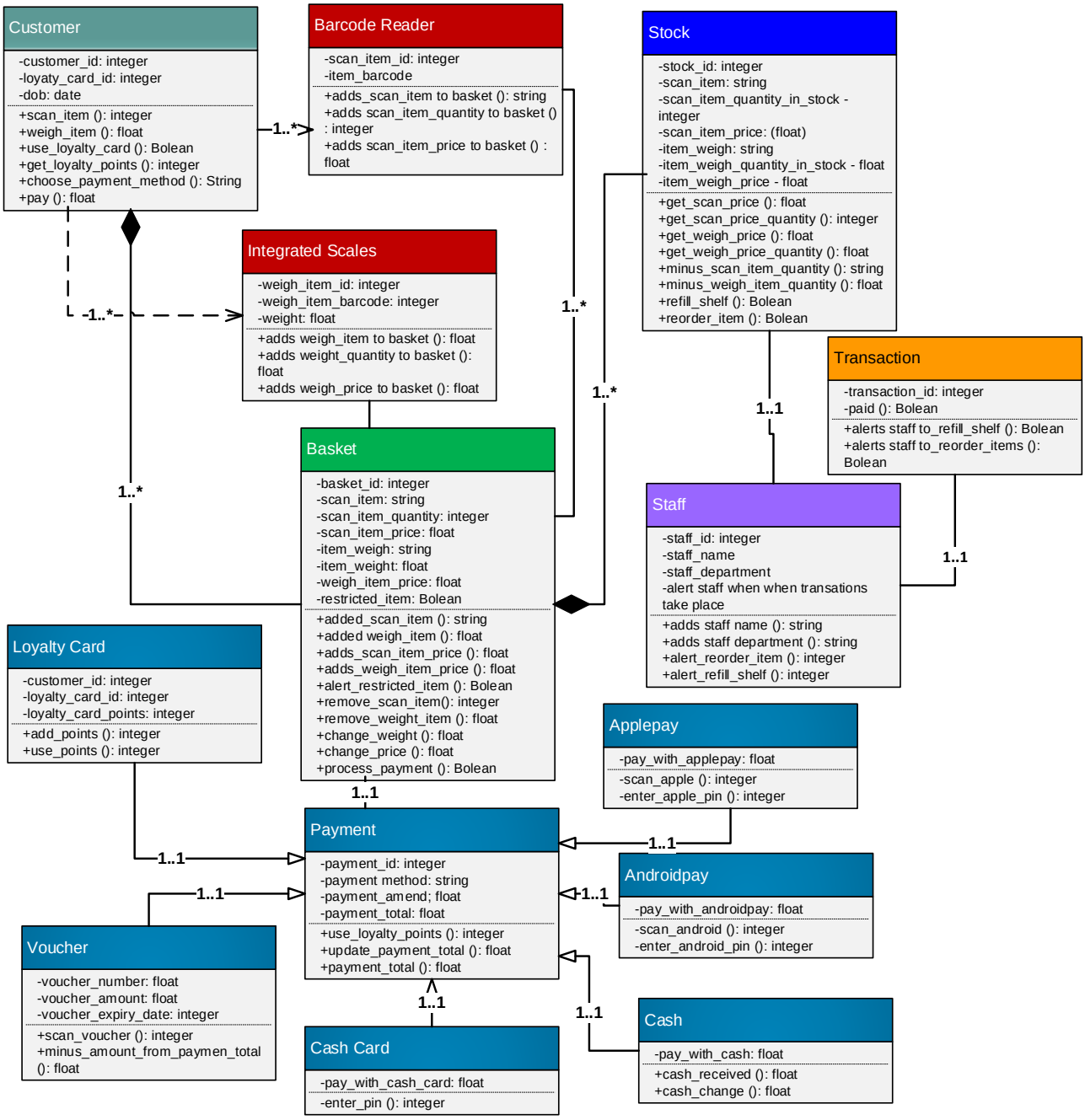
- **Supermarket Staff** can **Override Transactions**
    - **Supermarket Staff** will **check the customer's age** if the item is restricted then okay to add item or remove item.

- **Supermarket Staff** will **check a price is correct** and override it to the correct price if required.
- **Supermarket Staff** will check an **item has been weighed correctly** and override it to the correct weight if required.

Warehouse Staff

- **Warehouse Staff** are **alerted** after a **Completed Transaction** when **stock** needs to be **re-ordered.** There needs to be a minimum stock level to trigger an alert to re-order stock.
- **Warehouse Staff** are **alerted** after a **Completed Transaction** when **stock** needs to **refilled** on the **shelves.** There needs to be a minimum stock level to trigger an alert to refill the shelves.

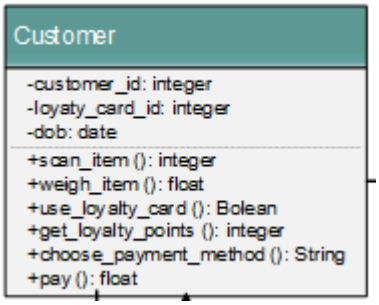
Class Diagram



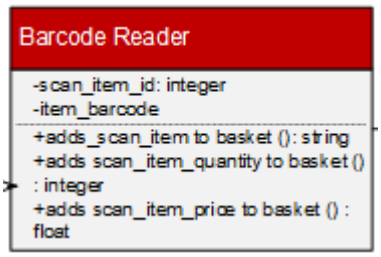
Attributes and Operations for Classes and Rationale for the System Design

Customers can choose to pay via the self-service terminal. There is an ID for the customer and loyalty card and their date of birth is required to purchases restricted items.

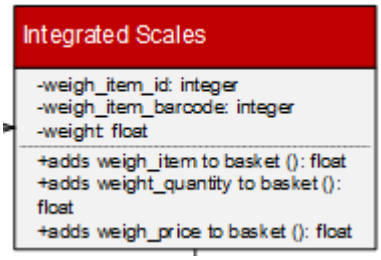
The customer can scan barcode items and weigh and select an item. They can choose to use their loyalty card by selecting yes or no and pay using different methods.



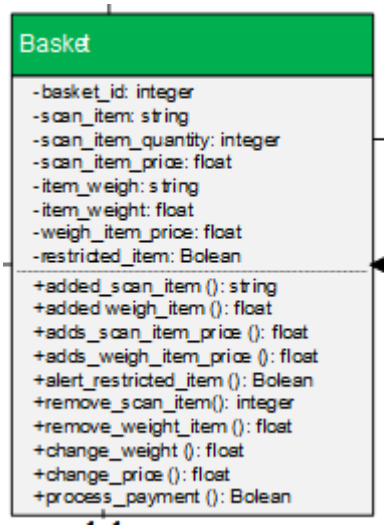
The Barcode Reader will scan any item that the customer puts through it.
When scanned the item is identified as well as the quantity and the price, these details are added to the Basket.



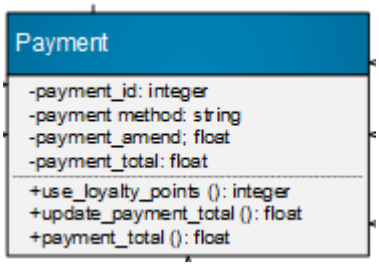
The Integrated Scales weigh relevant items that the customer puts on it.
When weighed, the item is identified and the quantity and the price are added to the Basket.



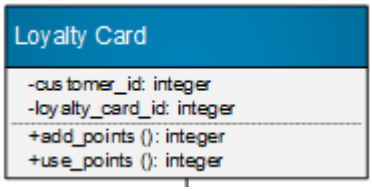
The Basket contains items the Customer scans or weighs as well as the quantities and prices.
Supermarket Staff, can access the basket to add, remove items, amend prices and quantities.
When Supermarket staff are alerted of a restricted item they verify the Customers' age and click yes or no if they are 18 or older, if not, the item can be removed.
The Customer can click to process the payment for the items in the Basket, using yes or no.



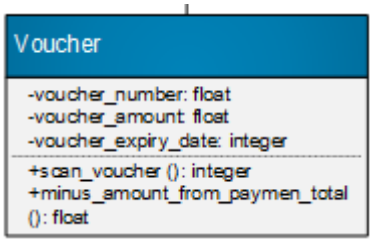
When a payment is processed the Customer is given a list payment options. There will be an option to use the Loyalty Card points, this updates the total to be paid.



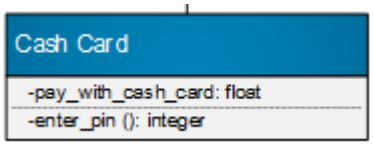
A Customer can choose whether to use their Loyalty Card, they can also decide whether to use existing points or had points added.



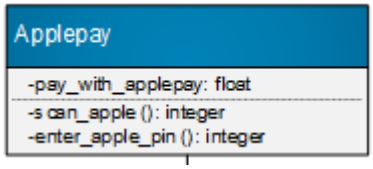
Vouchers can be selected as a Payment Method, this will check the amount and expiry date as well as using the voucher amount to pay for/towards the goods in the Basket.



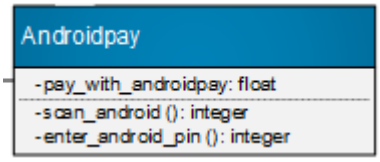
A Cash Card can pay for goods in the Basket, the Customer will be required to enter a pin.



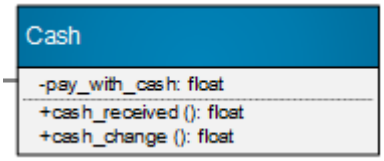
Applepay can pay for goods in the Basket, the Customer will scan their device and enter a pin.



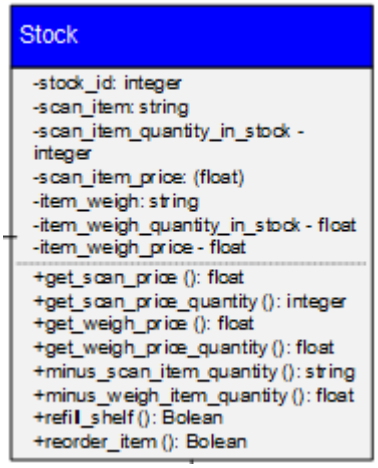
Androidpay can pay for goods in the Basket, the Customer will scan their device and enter a pin.



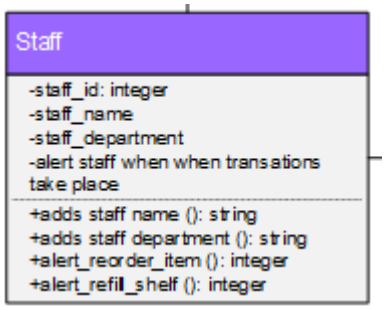
Cash can pay for goods in the Basket, the Customer will feed this into the checkout and be given the correct change.



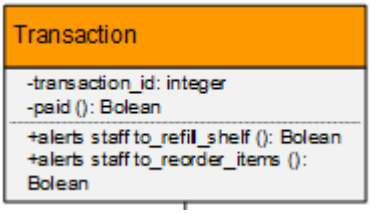
Stock are items scanned or weighed. The system will have the quantities in stock and deduct stock when a transaction is completed. When the quantity gets below a certain level it triggers an alert to the Warehouse Staff to place an order or refill shelves based on the level of stock a shelf holds.



There are two different staff the system impacts; Supermarket and Warehouse Staff, they are identified by Department and access to what they can do in the system is governed by this. Supermarket Staff can override items, quantities, prices and alert them to check restricted items. Warehouse Staff reorder items when prompted and refill shelves.



The completed Transaction triggers an alert to fill shelves or reorder items.



References

Fowler M. (2003) UML Distilled: A Brief Guide to the Standard Object Modeling Language. 3rd ed. Boston, MA: Addison Wesley

Liang, Y., 2003. From use cases to classes: a way of building object model with UML. Information and Software technology, 45(2), pp.83-93.

Seidle M, Scholz M. Huemer C. and Kappel G. (2014) UML @ Classroom: An Introduction to Object-Oriented Modeling, Springer Nature

Stevens P. (2000) Using UML: Software engineering with objects and components. Harlow, Essex: Addison Wesley