Unit 3

Fundamentals of Object-Oriented Design (Practical)

Practical Activity - Creating an object diagram

Summary

I really enjoyed this unit because it challenged me about existing UML modelling skills; caused me to think *why* should (or not) this or that relationship be used? Overall, it was a great activity to participate in because it has practical application I real-world. I admit that though this is unit 3, I referenced reading material from Unit 5 (UML Distilled by Martin Fowler) and other UML resources to understand the correct application of object diagrams versus class diagrams.

I feel that, since the requirement for this unit was to develop a *UML* object diagram, that UML knowledge should be presented *first* as opposed to delaying it to Unit 5. I understand the need to move incrementally from object-orientated concepts to modelling those objects, so perhaps this unit ought to leave off UML modelling requirements until Unit 5.

Diagram Description

The UML diagram presented is my first attempt at representing a fictitious supermarket. The practical activity was designed to get students to consider UML conventions and types of relationships; to think in terms of **objects** and their **relationships** such as **Association**, **Specialization**, **Composition** and **Aggregation** relationships as well as the **multiplicities** (and names) of relationships.

Modelling Requirements

For the practical activity, requirements stated as:

Create an object model to represent a supermarket, for example, you might like to consider some but exclusively all the following:

- *Staff*
- *Products*
- *Customers*
- *Online orders*
- *Loyalty schemes*

Consider how you might represent inheritance within your model and where you might use composition.

It was not immediately clear from the outset that the diagram was to be a simple *conceptual* diagram devoid of attributes and operations. So, based on the first posting of an object diagram by another student, I interpreted the tutor's feedback to mean "class diagram" and set about delivering on a class diagram.

Method to Develop the Object Model

In considering the objects of the fictitious supermarket, I took the following steps to produce an object model diagram. First, I looked for relevant system requirements. Second, I formed **sentences** to help identify all objects of the system. Then I generated one or more **conceptual** diagrams to focus on the requirements of the model. And lastly, I used the outputs of the previous steps—after a revision or two—and moved to producing a UML object model to observe the model at a single point in time.

I find this approach works for the modelling activity because it helps to slowly uncover the information required to meet all requirements.

Sentences used to identify objects:

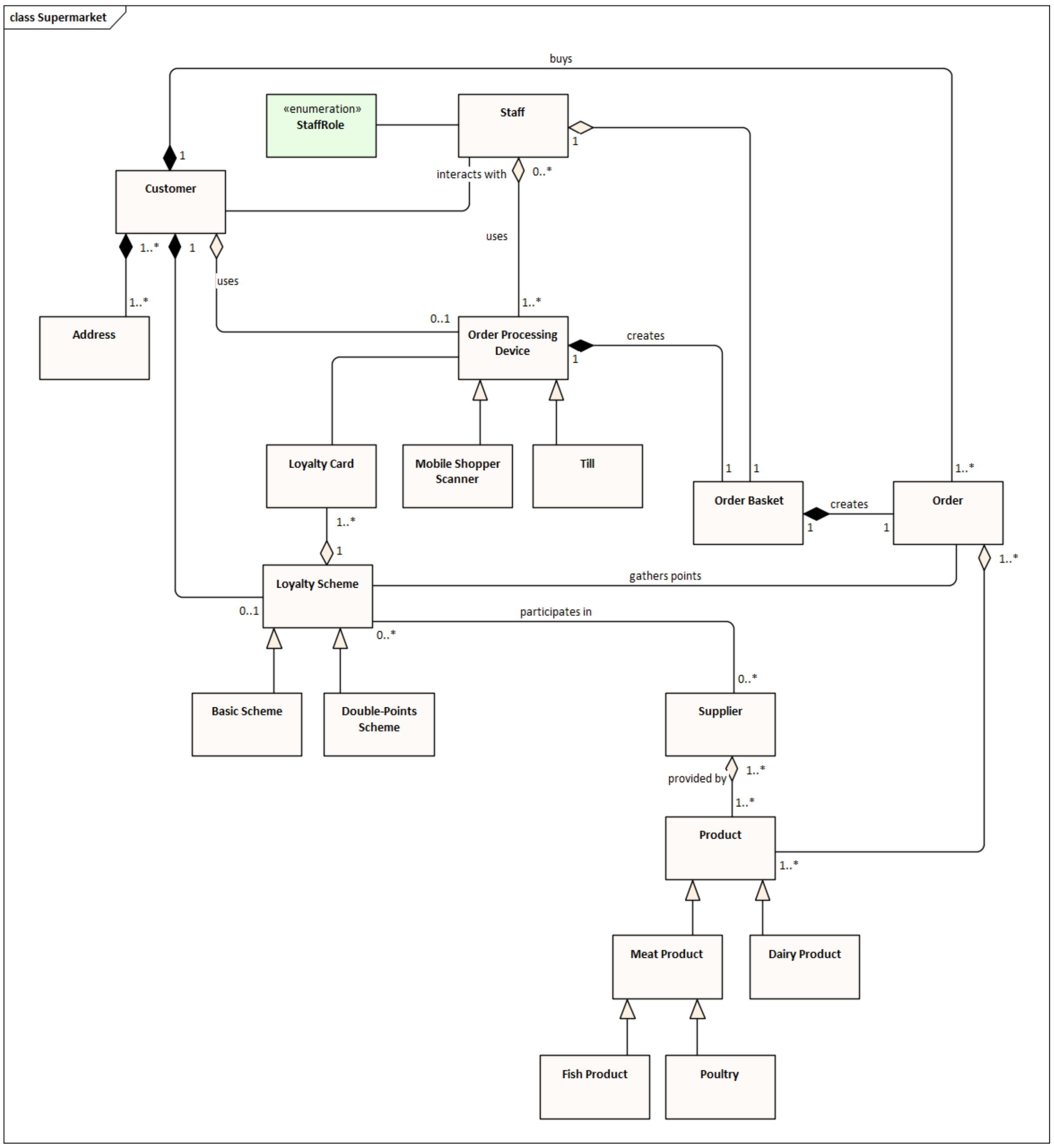
- `Staff` works with one or more `Order Processing Device`s.
- An Order Processing Device is either a `Mobile Shopper Scanner` or a `Till`.
- An Order Processing Device scans a `Customer`'s `Loyalty Card`.
- A Loyalty Card is part of a `Loyalty Scheme`.
- A Loyalty Scheme is either Basic or provides Double Points.
- A `Supplier` can provide one or more `Product`s.
- A Customer participates in zero or one Loyalty Schemes.
- A Customer resides at a single `Address`.

- A Customer will perform a checkout operation either by themself OR by interacting with a Staff member (who acts in a particular `Role`).
- An Order Processing Device is used to process a Customer's `Order` which is an `Order Basket`.
- An Order consists of one or more Products.

First is presented the object diagram for this activity. Following are the modelling attempts taken to arrive at the final object diagram.

Attempt #1

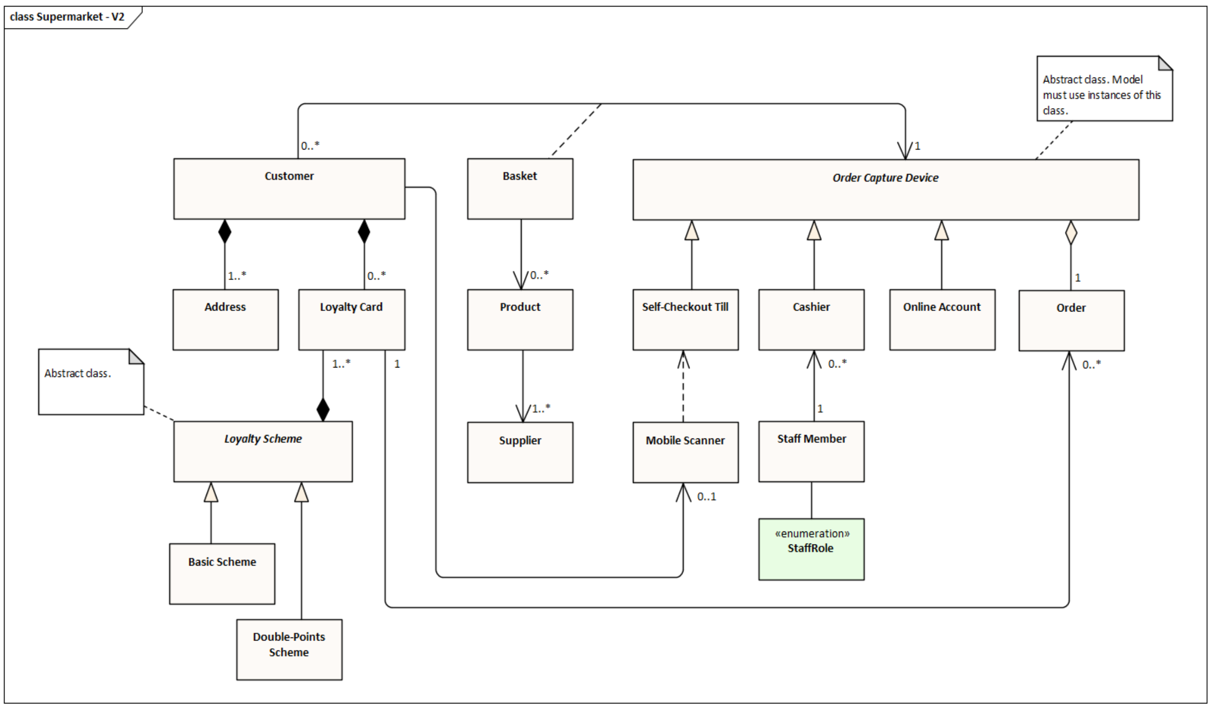
The following diagram is based on the sentences described:



UML object model (1st draft)

Attempt #2

Reviewing this model, I noted that the first draft does not quite follow my real-world experience. So, modelling my experience of using **Sainbury's (UK)** as inspiration, resulted in a second revision shown below:



UML object model (2nd draft)

I like the second attempt a lot more than the first attempt, because of the following:

- **Association Classes**.

  `Basket` is an association class between `Customer` and `Order Capture Device`. This implies that an `Order` **must** exist when a `Customer` is associated with an `Order Capture Device`.

- **Online Account.**

  `Online Account` is a type of `Order Capture Device` that is instantiated whenever a `Customer` accesses the shopping website portal. I like this idea because `Online Account` becomes just another means to capture a collection of `Product` instances without paying attention to the *method by which* the `Online Account` is accessed.

- **Order Capture Device.**

  The model shows that a `Customer` can use three different methods to process their `Order`, namely online (Online Account) or in-person (Self-Checkout, Cashier).

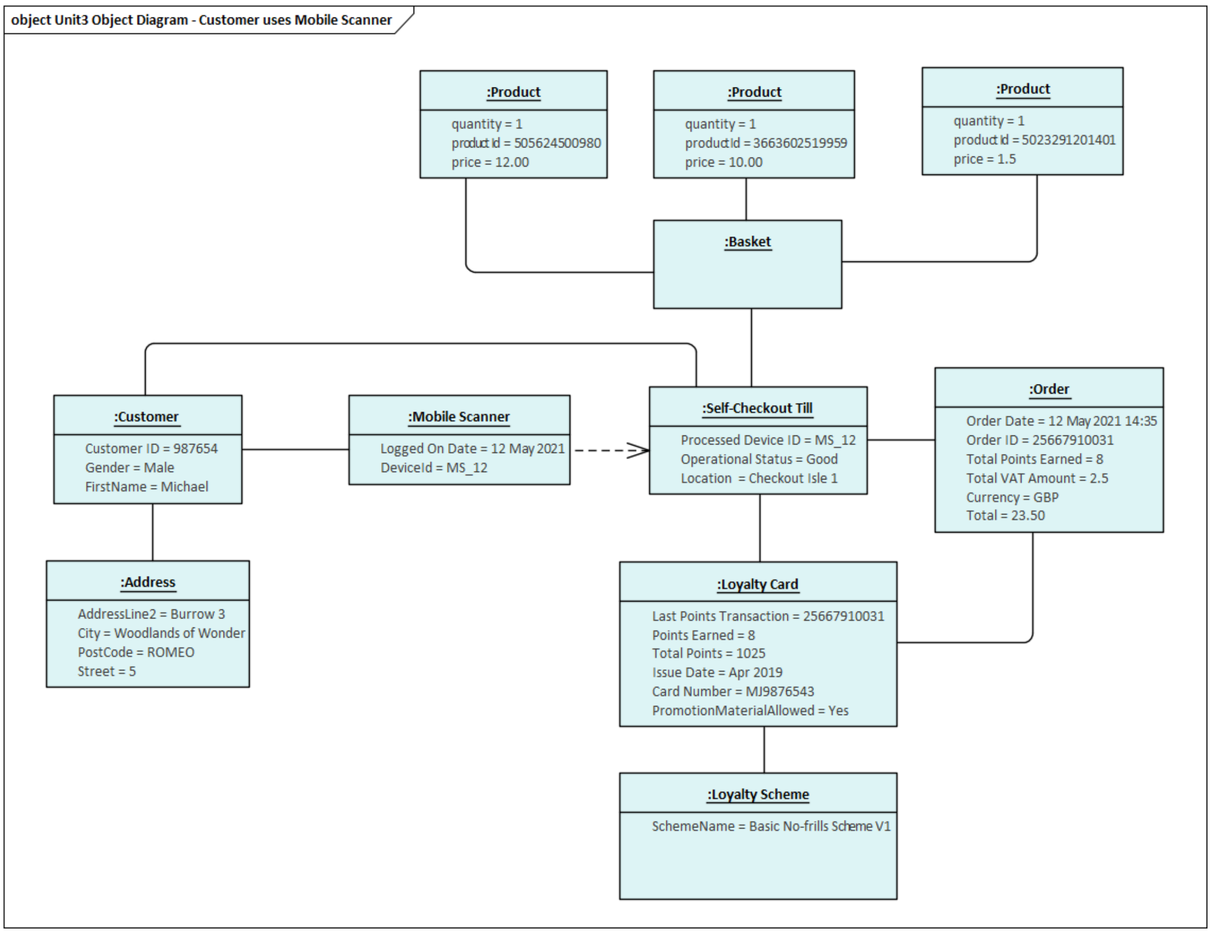
- **Mobile Scanner.**

  This represents a handheld scanner that can scan a collection of `Products` into the `Customer's` basket. I found interesting to model that the `Mobile Scanner` does not generate `Orders` as such, and **depends on** the service of another class, `Self-Checkout`, to complete an `Order`. Whether or not this is correct, I am not fully certain, but I do believe the relations are sufficient.

Final Object Diagram

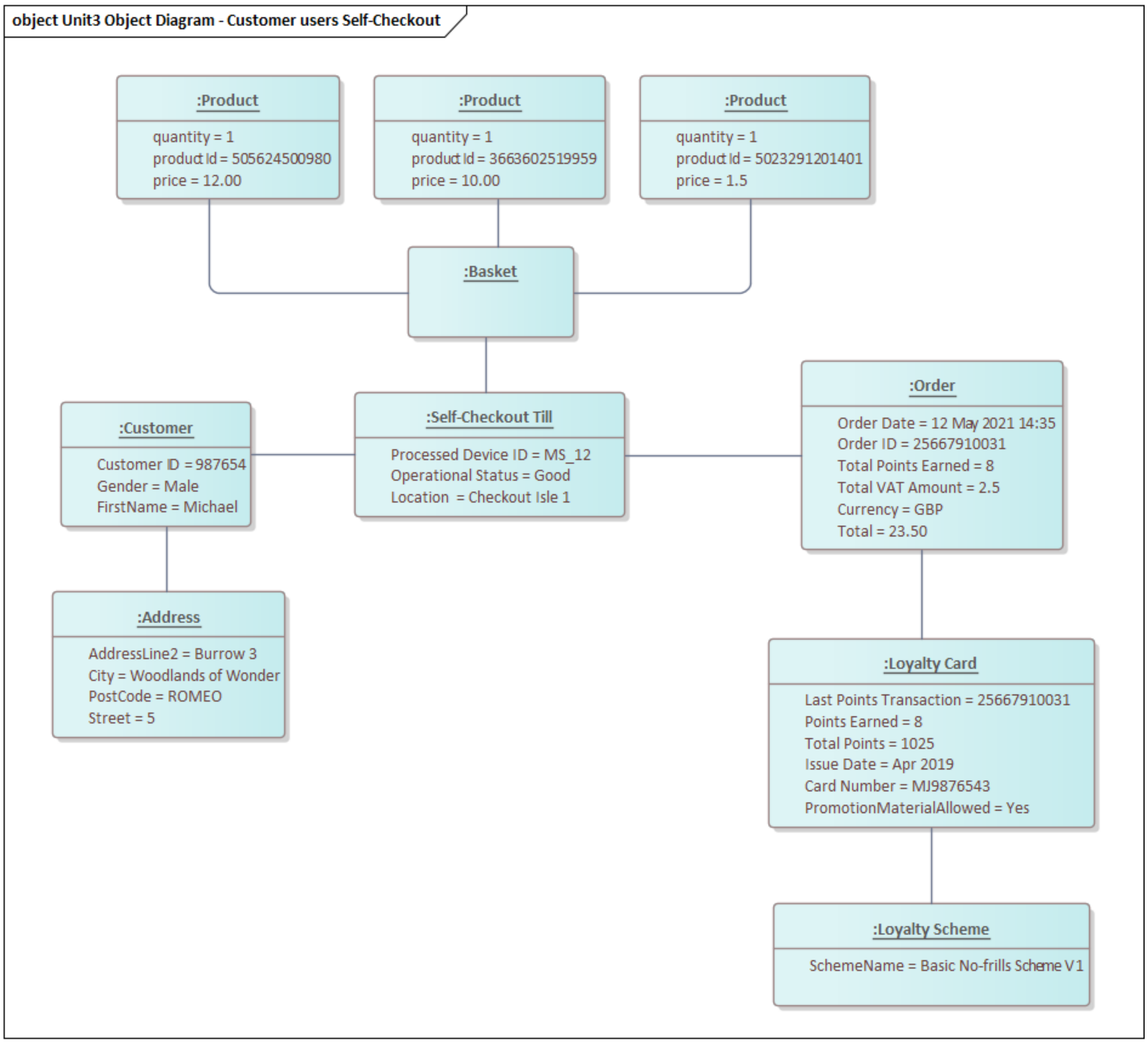
I generated a few object diagrams to test the assumptions of the final object model based on each potential **use case** the model must be able to handle.

- Customer uses the `Mobile Scanner`



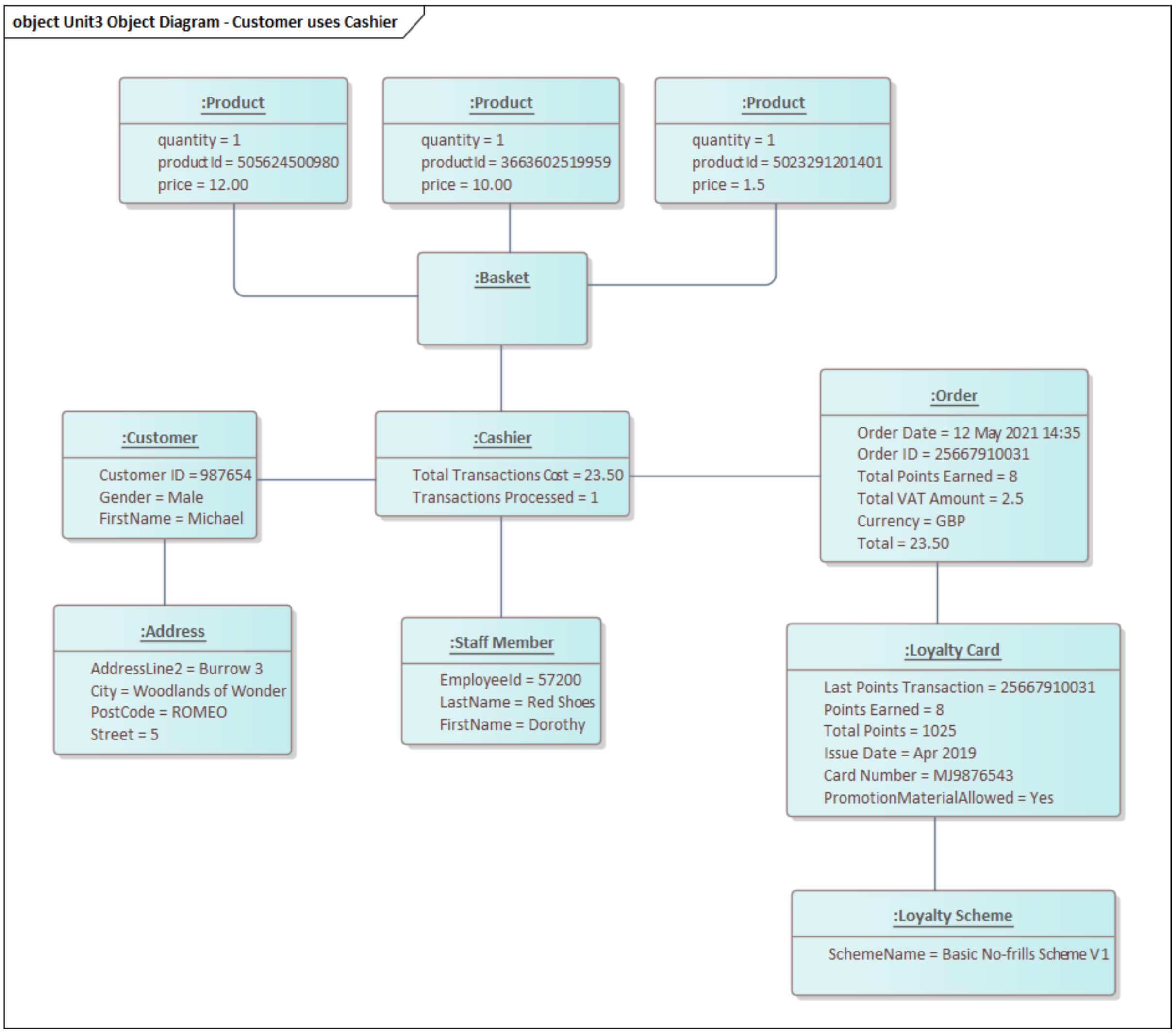
UML object model

- Customer uses the `Self-Checkout Till`



UML object model

- Customer uses the `Cashier`



UML object model