**Assignment 1, module 2**

This report outlines a modular (Bhatt & Pahade, 2021) design of a software system managing the self-service checkout at a supermarket, according to the assignment brief. This design includes relevant classes (section 1), their relationships (section 2), and their class diagram (section 3) generated via the industry-grade tool “PlantUML” (Hnatkowska *et al*., 2021; PlantUML, 2021).

**1. List of relevant classes and their rationale**

* ‘**VirtualShoppingBasket**’: to add/remove items to/from a shopping basket and compute their total price.
* ‘**BarcodeReader**’:to decode a barcode on an item’s image.
* ‘**Browse**’:interface to browse an item by its name, category, or bar code ID.
* ‘**IntegratedScale**’: to check whether an item’s weight matches the expected one.
* ‘**Customer**’: to pay for items.
* ‘**Item**’: to encapsulate and update items’ characteristics.
* ‘**AlcoholicItem**’:a sub-class of ‘Item’ for alcoholic items.
* ‘**TobaccoItem**’: a sub-class of ‘Item’ for tobacco-related items.
* ‘**LoyaltyCard**’: to encapsulate and update loyalty card-related data.
* ‘**PaymentMethod**’: to choose various payment methods and get cash back.
* ‘**StaffMember**’: to order items from suppliers, override transactions, restock shelves, and verify items’ availability.
* ‘**WarehouseStaffMember**’:a sub-class of ‘StaffMember’ with the assigned department.
* ‘**InStoreStaffMember**’: a sub-class of ‘StaffMember’ indicating whether they are assigned to a customer service kiosk.
* ‘**Transaction**’: to update the supermarket stock control system based on purchased items, alert the warehouse staff based on their availability, and request customers to get bagged paid items from the integrated scale.
* ‘**SupermarketStockControlSystem**’: to add or remove items, update their prices, check their availability, and order them.
* ‘**Shelf**’: to add or get the items on a shelf or remove them from it.
* ‘**Supplier**’: to check the order status and ship paid items.

**2. List of relationships between classes and their rationale**

The relationships (Phillips, 2018; Pine, 2019) among the classes in section 1 are:

* **Aggregation**:
  + ‘**Shelf**’ class has instances of ‘Item’, but ‘Item’ can exist without ‘Shelf’.
* **Associations**:
  + ‘**VirtualShoppingBasket**’class uses ‘Item’ objects and enables to interact with a ‘StaffMember’ object.
* **Compositions**:
  + ‘**VirtualShoppingBasket**’,‘**LoyaltyCard**’, ‘**PaymentMethod**’,and‘**Transaction**’ classes cannot exist without the ‘Customer’ class.
* **Implemented by**:
  + ‘**Browse**’interface is implemented by ‘SupermarketStockControlSystem’ and ‘BarcodeReader’.
* **Inheritance**:
  + ‘AlcoholicItem’ class extends the ‘Item’ class by the ‘is\_spirit’ Boolean attribute.
  + ‘TobaccoItem’ class extends the ‘Item’ class by the ‘is\_electronic’ Boolean attribute.
  + ‘WarehouseStaffMember’ class extends the ‘StaffMember’ class by the assigned department/s.
  + ‘InStoreStaffMember’ class extends the ‘StaffMember’ class by a Boolean attribute indicating whether they are assigned to the customer service kiosk.
* **Uni-directional associations** – only (uni-directionally)oneclass calls objects instantiated from other classes:
  + ‘**VirtualShoppingBasket** class with respect to ‘BarcodeReader’, ‘IntegratedScale’, ‘LoyaltyCard’, ‘PaymentMethod’, and ‘Transaction’.
  + ‘**BarcodeReader**’ class retrieves ‘Item’ objects based on the decoded bar code ID.
  + ‘**Customer**’class uses ‘BarcodeReader’ to browse for items based on their decoded bar code ID.
  + ‘**StaffMember**’class uses objects from ‘Supplier’, ‘Shelf’, ‘SupermarketStockControlSystem’, and ‘BarcodeReader’.
  + ‘**SupermarketStockControlSystem**’class with respect to ‘Item’ and ‘Supplier’.
  + ‘**Transaction**’class with respect to ‘WarehouseStaffMember’ and ‘SupermarketStockControlSystem’.
* **Used by**:
  + ‘**Browse**’interface is consumed by ‘VirtualShoppingBasket’, ‘Customer’, ‘StaffMember’, and ‘SupermarketStockControlSystem’.

**3. Class diagram**

**Fig. 1** illustrates the UML diagram of the classes in section 1 and their relationships in section 2.

Diagram

Description automatically generated

**Figure 1**. The class diagram of the proposed software system.

**References**

Bhatt, Y., & Pahade, P. (2021). Application of Python Programming and Its Future. In *Information and Communication Technology for Competitive Strategies* (ICTCS 2020) (pp. 849-857). Springer, Singapore.

Hnatkowska, B., Huzar, Z., & Tuzinkiewicz, L. (2021). Consistency Assessment of Datasets in the Context of a Problem Domain. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems* (pp. 112-125). Springer, Cham.

Phillips, D. (2018). *Python 3 object-oriented programming: Build robust and maintainable software with object-oriented design patterns in Python 3.8*. Packt Publishing Ltd.

Pine, D. J. (2019). *Introduction to Python for science and engineering*. CRC Press.

PlantUML (2021). *PlantUML Language Reference Guide*, version 1.2021.2. Retrieved from http://plantuml.com/ guide. Last accessed on Apr 13th, 2022.