Assignment 5: Software Design

In this assignment, you are requested to create a software design for *GroceryListManager*, an application for managing grocery lists whose requirements are listed below. To do so, you should follow the same approach that we present in the P3L2 lesson, that is, analyze the requirements to identify and refine (1) classes, (2) attributes, (3) operations, and (4) relationships in your design. Just to be completely clear, **your task is to design the system, not to implement it**. Please note that not all requirements will necessarily affect your design in a direct way. For example, you do not have to do anything about the GUI in your design. For another example, the fact that a DB is mentioned does not mean that you have to actually model a DB.

Your design should be expressed using a UML class diagram, and the level of detail of the design should be analogous to the level of detail we used in the P3L2 lesson. Basically, you have to provide enough details for the design to be self contained and for allowing us to assess whether the design is suitably realizing the system requirements. To help with this, you must also provide a "design information" document in which you **concisely** describe, for each of the requirements listed below, how that requirement is either realized in your design or not considered because it does not affect it. For example:

…
2. The application must contain a database (DB) of items and corresponding item types.
To realize this requirement, I added to the design a class X with attributes Y and Z. Class X...
...
11. The User Interface (UI) must be intuitive and responsive.
Not considered because it does not affect the design directly.

Optionally, you can also provide in the document additional information about your design, such as assumptions or rationale for some design decisions.

You can use any UML tool to create your design. If you are not familiar with any specific tool, we recommend that you ask on Piazza for suggestions. In fact, on Piazza there is already some discussion about that.

Requirements

1. A grocery list consists of items the users want to buy at a grocery store. The application must allow users to add items to a list, delete items from a list, and change the quantity of items in the list (e.g., change from one to two pounds of apples).
2. The application must contain a database (DB) of *items* and corresponding *item types*.

3. Users must be able to add items to a list by picking them from a hierarchical list, where the first level is the item type (e.g., cereal), and the second level is the name of the actual item (e.g., shredded wheat). After adding an item, users must be able to specify a quantity for that item.
4. Users must also be able to specify an item by typing its name. In this case, the application must look in its DB for items with similar names and ask the users, for each of them, whether that is the item they intended to add. If a match cannot be found, the application must ask the user to select a type for the item and then save the new item, together with its type.
5. Lists must be saved automatically and immediately after they are modified.
6. Users must be able to check off items in a list (without deleting them).
7. Users must also be able to clear all the check-off marks in a list at once.
8. Check-off marks for a list are persistent and must also be saved immediately.
9. The application must present the items in a list grouped by type, so as to allow users to shop for a specific type of products at once (i.e., without having to go back and forth between aisles).
10. The application must support multiple lists at a time (e.g., “weekly grocery list”, “monthly farmer’s market list”). Therefore, the application must provide the users with the ability to create, (re)name, select, and delete lists.
11. The User Interface (UI) must be intuitive and responsive.

To submit your assignment, you should do the following:

- Create a directory called Assignment5 in the usual **personal GitHub repository we assigned to you**.
- Save your UML class diagram in the Assignment5 directory as a PDF file named “`design.pdf`”.
- Save the “design information” document in the same directory, in markdown format, and name it “`design-information.md`”.
- Commit and push your file(s) to your remote repository.
- Submit the commit ID for the files on T-Square.