Assignment 5: Software Design

In this assignment, you are requested to create a software design for *Reminders*, an application for task reminders whose requirements are listed below. To do so, you should follow the same approach that we presented in class, 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 used in the class example. 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 reminders and corresponding data.
To realize this requirement, I added to the design a class X with attributes Y and Z. Class X...
...
13. 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.

Requirements

1. A list consisting of reminders the users want to be aware of. The application must allow users to add reminders to a list, delete reminders from a list, and edit the reminders in the list.
2. The application must contain a database (DB) of *reminders* and corresponding *data*.

3. Users must be able to add reminders to a list by picking them from a hierarchical list, where the first level is the reminder type (e.g., Appointment), and the second level is the name of the actual reminder (e.g., Dentist Appointment).
4. Users must also be able to specify a reminder by typing its name. In this case, the application must look in its DB for reminders with similar names and ask the user whether that is the item they intended to add. If a match (or nearby match) cannot be found, the application must ask the user to select a reminder type for the reminder, or add a new one, and then save the new reminder, together with its type, in the DB.
5. The reminders must be saved automatically and immediately after they are modified.
6. Users must be able to check off reminders in the list (without deleting them).
7. Users must also be able to clear all the check-off marks in the reminder list at once.
8. Check-off marks for the reminder list are persistent and must also be saved immediately.
9. The application must present the reminders grouped by type.
10. The application must support multiple reminder lists at a time (e.g., “Weekly”, “Monthly”, “Kid’s Reminders”). Therefore, the application must provide the users with the ability to create, (re)name, select, and delete reminder lists.
11. The application should have the option to set up reminders with day and time alert. If this option is selected allow option to repeat the behavior.
12. **Extra Credit:** Option to set up reminder based on location.
13. 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 Blackboard.