Assignment 5: Software Design

In this assignment, you will design an application for students to practice taking vocabulary quizzes. A student can take vocabulary quizzes created by students, or create their own quiz to be taken by other students. The application will also keep track of quiz score statistics by student and by quiz. The detailed requirements for the application are provided below.

To create your design, 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**. The requirements for the application are listed below, in the "Requirements" section. Please note that not every requirement will be fully and directly represented in your design. For instance, at this level of detail, you do not have to show any purely GUI specific classes, if they are *only* doing user display and input and not performing any significant business logic. Similarly, any database support layer may be left out, if it is purely doing persistence tasks and not performing any data manipulation.

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 throughout the **whole** P3L2 lesson (i.e., do not limit your design to only the elements focused on in the final video). Specifically, **you must provide enough details for the design to be self contained and for us to be able to assess whether the design suitably realizes all 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 why it does not directly affect the design and is not shown. Copy the list of requirements, and add your explanation for each one of them. For example:

…
2. The application must contain a list of items in object 'X'. Object 'Y' must use the list for action 'A'.
To realize this requirement, I added to the design a class 'X' with list 'Z' of items. Class 'Y' has a relationship to class 'X' and uses it in method 'A' to….
...
16. The User Interface (UI) must be intuitive and responsive.
This is not represented in my design, as it will be handled entirely within the GUI implementation.

This explanation should be clear enough to allow us to follow the rationale behind your design and **how it will fulfill each specified requirement, including any that are not directly depicted in your class diagram.** 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, but do not hand draw it. 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. When starting the application, a user can choose whether to (1) log in as a specific *student* or (2) register as a new *student*.
    a. To register as a new student, the user must provide the following student information:
        i. A unique username
        ii. A major
        iii. A seniority level (i.e., freshman, sophomore, junior, senior, or grad)
        iv. An email address
    b. The newly added *student* is immediately created in the system.
    c. For simplicity, there is no password creation/authentication; that is, selecting or entering a *student* username is sufficient to log in as that *student*.
    d. Also for simplicity, *student* and quiz information is local to a device.
2. The application allows *students* to (1) add a quiz, (2) remove a quiz they created, (3) practice quizzes, and (4) view the list of quiz score statistics.
3. To add a quiz, a *student* must enter the following quiz information:
    a. Unique name
    b. Short description
    c. List of N words, where N is between 1 and 10, together with their definitions
    d. List of N * 3 incorrect definitions, not tied to any particular word, where N is the number of words in the quiz.
4. To remove a quiz, *students* must select it from the list of the quizzes they created. Removing a quiz must also remove the score statistics associated with that quiz.
5. To practice a quiz, *students* must select it from the list of quizzes created by other students.
6. When *a student* is practicing a quiz, the application must do the following:
    a. Until all words in the quiz have been used **in the current practice session**:
        i. Display a random word in the quiz word list.
        ii. Display four definitions, including the correct definition for that word (the other three definitions must be randomly selected from the union of (1) the set of definitions for the other words in the quiz and (2) the set of incorrect definitions for the quiz.
        iii. Let the *student* select a definition and display “correct” (resp., “incorrect”) if the definition is correct (resp., incorrect).
    b. After every word in the quiz has been used, the *student* will be shown the percentage of words they correctly defined, and this information will be saved in the quiz score statistics for that quiz and *student*.

7. The list of quiz score statistics for a *student* must list all quizzes, ordered based on when they were last played by the *student* (most recent first). Clicking on a quiz must display (1) the student's first score and when it was achieved (date and time), (2) the student's highest score and when it was achieved (date and time), and (3) the names of the first three students to score 100% on the quiz, ordered alphabetically.
8. The user interface must be intuitive and responsive.
9. The performance of the game should be such that students do not experience any considerable lag between their actions and the response of the application.

Submission Instructions

To submit your assignment, you should do the following:

- Create a directory called `Assignment5` in the usual **personal GitHub repository we assigned to you**. This is an **individual assignment**; do **not** use your new team repositories.
- Save your UML class diagram in the `Assignment5` directory as a PDF file named "`design.pdf`". **Important:** Make sure to open your PDF after generating it and double check it, as we had a number of cases of students not realizing that the conversion to PDF had not worked as expected.
- 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 your solution on Canvas.