CIS 313 Project

Project objectives

- Conduct a project of significant complexity including all phases of the software development life cycle, from analysis and design to testing and debugging, and using object-oriented techniques.
- Develop and implement a program that requires GUI, use of data structures, text an/or file handling.
- Develop good collaboration skills using Git and GitHub to share work.

Project topic: You must create a quiz application that will allow teachers to enter quizzes and students to take the quizzes. Quiz questions can include multiple-choice, true-false of short answer questions. The quiz is automatically grades by the application, and results are stored for later review by teacher and students. Further specifications must be obtained by interviewing client.

1. Interview client (instructor) and complete specifications → functional requirement document
2. Complete the design phase → design specifications
3. Implementation and testing → Source code and unit tests
4. Executable file
5. Documentation → Final report
6. Project presentation
7. Individual report.

Project requirements

1. It allows you to demonstrate mastery of OOP, use of data structure, and GUI and programming.
2. It is complex enough without being overly ambitious.
3. The whole class will work on the same project, distributing tasks as appropriate.
4. You must use GitHub to host your project and collaborate.

Requirement document

You must perform a thorough analysis of your project that will result in the functional requirements document. This document must specify what the purpose of your program is and all its functionalities (see chapter 6). It must include all necessary use cases and class diagrams showing conceptual classes and their relationships.

Design specifications

You will perform an Object-Oriented Design (OOD). In this stage you identify the system structure and behavior. The analysis phase should have provided the major conceptual classes to the system. In this phase, you must identify the software classes, the relationship between classes, each class attributes and responsibilities, and interaction between objects. This is the process described in section 7.1 of your book. You will use sequence diagrams to assign responsibilities to classes, from which you deduce methods.

You design document will include the sequences diagrams and complete class diagrams for each class, including all attributes and methods. You should also include mock-up of your graphical user interface and indication of plans for data storage.

Source code - Testing and debugging
You must implement your design in Java. Your code must be efficient, use appropriately java construct, and object-oriented programming. You must include all necessary documentation comments. Your code must be well-formatted, the identifiers follow usual java convention. All authors and sources must be properly attributed.

Each group member must take responsibility for at least one non-trivial class. For this class, that group member must be named as the sole author. The author must have written the entire source. (You can, of course, discuss the classes and help each other, but the author must type in and understand all of the code of the class he/she is solely responsible for.) Additional classes may be authored by more than one group member. You must use GitHub to collaborate on the code.

Executable file
You must submit both the source code and an executable jar file.

Final group report
The group report should document your program, the team and the development process. It should contain at least

- a title
- a list of the group members, their roles and involvement and their responsibilities
- identify one class for each group member for grading purposes (the group member must be the sole author and may be interviewed about this class)
- a user level description of the program (*What does it do? How do I use it?*)
- a high-level implementation description of your program (design overview, important design decisions, some important implementation details)
- problems encountered (solved and unsolved)
- restrictions / limits (*What does the program not do?*)
- unimplemented features (*What was planned, but didn't get done because the time ran out?*)
- known bugs
- possible extension/improvement.

The group report should be between 6 and 8 pages long (12 point font, double-spaced, body of text, without appendices).

Final presentation
At the end of the project you will give a presentation and a demonstration of the running program to your clients.
The presentation must include a demonstration of the running program.
The time available for the presentation is 30 minutes. You should be well prepared and use presentation slides. All group members must participate in the presentation.

The purpose is to demonstrate what the program can do and what it looks like. It should emphasize all topics that may be interesting to the client and to users. This includes, obviously, a user level demonstration: what does the program do; what does it look like? You should also talk a bit about implementation structure: what objects are you using, what collections, what are their relationships, when/how are they created, etc.
It is expected that you produce a well-prepared, polished presentation. Visiting the Pearce Communication Lab is highly recommended.

Teamwork
I will create a GitHub assignment for the project on GitHub. Each team member is responsible to push her/his contributions to the repository. This will allow me to assess that every member of the team is contributing significantly to the project. You may use the different tools provided in GitHub such as issues and wiki to enhance your collaboration.
Each team might consider assigning specific roles to team members. Possible roles are project manager (keep the team on task, plan deadlines, etc.), liaison with instructor, "code checker" (responsible for merging code produced by different team members), etc.

The grade for this assignment is broken down as follows:

Requirement document	30 points
Project design	30 points
Project quality	50 points
Use of GitHub	10 points
Final presentation	15 points
Project Group report	20 points
Individual report	5 points
Total	**160 points**

Project quality includes issues such as project design, originality, usefulness, usability, functionality, difficulty and code quality.
Code quality is determined by correctness, coding style, appropriate use of language constructs (especially object-oriented constructs) and your understanding of the code.
Each group member must contribute substantially to the group work (including the full implementation of a non-trivial class). Every group member may be examined by an interview if the individual contribution to the group is not evident.

Deadlines
Saturday, November 4: Requirement document
Wednesday, November 15: design specifications
Saturday, December 9: Source code including unit test, executable file, final reports (group and individual)
Wednesday, December 13: presentation.
All work must be submitted via GitHub except for the individual report and the executable file which must be submitted on KC.

Late work penalty: Work submitted late will incur a 5% per calendar day penalty.