Group Project Assignment

- You will work on the class project as a member of a team. The optimal team size is 3 people.
- The problem statement can be found in the file, “ProjectDescription.pdf.”
- Your team will produce five documents and one presentation. These will be submitted using the Discussion tool for the rest of the class to view. If you want to revise your documents, this may be accomplished using the Discussion tool as well. Most teams submit their documents as .doc, .docx, .rtf, or .pdf attachments.
    - Team charter
    - Project plan
    - Vision document
    - Requirements document
    - Design document
- There will be one presentation to the class during the last week of the term. This is the final project demonstration. Your team can use Adobe Connect, or any other available tool, to create a video of your presentation, the link to which you will then submit using the Discussions tool. The rest of the class will be able to then view your presentation.
- Each of the documents and the final demo presentation will be graded on a scale of 1 to 10. This amounts to 60 percent of your project grade. Everyone on the team receives the same grade on these documents and presentation.
- 40 percent of your project grade will be based on a peer review. You will have the opportunity to provide feedback on your team partners. (See the rubric in the "TeammateEvaluationForm.pdf" file.) During the last week of the course we will send an e-mail asking you to return the forms by return email. We will average together the feedback from your teammates for this peer evaluation.
- You will have to code and test the project.
    - You may use any programming language(s) you want.
    - You may use any CASE (Computer Aided Software Engineering) tools that are available to you.
- You may use Adobe Connect for your team collaboration and to produce your videos.
- There are other team collaboration tools provided in Blackboard.

The project is described in the file, "ProjectDescription.pdf."

The purpose of the class project is to exercise the skills and knowledge you gain in the class. Don't think of the project as a programming project (although there is programming involved at the end). Think of it as a software engineering project. You will be applying software engineering approaches. Your grade will depend on how well you do with the software engineering. Coding is just a small part of the entire project. In fact, it is possible for a reasonable skillful programmer to hack a solution in a weekend, so the programming part of the project shouldn't be your main concern. Once again, you must show evidence of how you have applied software engineering skills and knowledge to the project.

There are several milestones along the way:

1. Team Charter
2. Project Plan
3. Vision Document
4. Software Requirements Specification (SRS)
5. Skeletal System (internal)
6. Minimal System (internal)
7. Software Design
8. Project Demo of the Target System

The Team Charter

This document describes your team. You should choose a team name and produce a team logo. (Team motto, team flag, team song, secret handshake, etc. are optional.) The team charter document should contain the team name and logo, and biographies of the team members. It would be a good idea to include photos of the members, since we won't be seeing you in person. It might a bit early to decide on these things, but your charter should include a first cut at who is going to do what on the project. Jobs include project manager, lead architect, lead programmer, lead tester, lead SQA (Software Quality Assurance) engineer, lead CM (Configuration Management) engineer, etc. Of course these people just lead. Everybody works. You might want to decide who will be the editor for each of the deliverables you have to produce. You might also consider who will be doing the presentations. One person can do the whole presentation, or you can use the "tag team" approach. Your charter must also describe how the team will make decisions and how it will handle and resolve conflicts and issues.

The Project Plan

Your plan should have at least the following sections:

- Work-breakdown structure (WBS).
- The features of the following four increments:
    - Skeletal — architecture.
    - Minimal — most important functionality.
    - Target — what you can expect to produce in one semester.
    - Dream — what you could do if you had enough time.
- Schedule.
- Risk assessment plan.
- Quality plan.
    - Quality Assurance.
    - Testing.
    - Configuration Management.

The Three Increments:

Skeletal System

- This is an internal milestone. There is no deliverable.
- This increment validates the architecture.
- May require stubs and drivers for testing.
- All subsystems are defined and their requirements are specified in SRS documents.
- All messages between subsystems are defined.
- From here, the individual subsystems may be developed independently.

Minimal System

- This is also an internal milestone. There is no deliverable.
- Purpose of the Minimal system:
    - Minimize risk.
    - Prove out the architecture.
    - All remaining requirements are fleshed out.
    - Update the Project Plan.
    - Revise the Software Requirements Specification.
    - Continue the Design.

Target System

- This is the final increment of capability for the semester.
- It's what will be demonstrated on the last day of the class.
- The target system is the version that should be described in the all of the remaining deliverables (HCI Prototype, SRS, and Design document).

Vision Document

- Documents the vision of the product from the stakeholder viewpoint.
- Documents the problem.
- Identifies stakeholder groups and their needs.
- You can modify the template used in the lecture and remove any items that aren't really applicable to this project.

Software Requirements Specification (SRS)

- This sets out the requirements for the target system.
- Describe the following:
    - Glossary - define key terms.
    - Software architecture — identify the subsystems. What are the functions of each subsystem? What information does the subsystem encapsulate? What are the interfaces required? This may be a separate document.
    - Document applicable functional requirements with use cases in a use case document
    - Document any functional requirements that are not amenable to use cases in a supplementary specification document.
    - Non-functional requirements and design/implementation constraints shall be documented in a supplementary specification document.

Software Design Document

- This is the design of the target system.
- Class diagram — show the classes and their connections (associations, aggregations, specializations).
- For each class, document its attributes operations, and connections to other classes.
- For each of the major scenarios, document the interactions between objects of these classes using a dynamic model (e.g., sequence or activity diagram).

Project Demo

This will be due by the last day of the class. You will record a demonstration of your system.