# CSCI4448: Project Part 2

## Spring 2014: Due Friday, March 21 at 8:00 pm

**Note*: start early – you have time now, and this is due 3 days after your mid-term!***

**Semester Project Overview**
The project is divided up into 5 parts.
Notice that the focus is on *Object-Oriented Analysis and Design,* as per the title of the course!

| Part | Points | Due Date |
|---|---|---|
| 1: Topic and Team | 10 | Tue, Feb 18 |
| 2: Requirements & Analysis | 150 | Fri, Mar 21 |
| 3: Progress Report | 50 | Fri, Apr 11 |
| 4: Finished Project, Report | 50 | Fri, Apr 25 |
| 5: Demo | 50 | Week of Apr 28 (sign-up for spot) |

**Project**
Part 2 asks you to engage in analysis and design activities for your semester project over the next two weeks. You will generate a detailed set of tasks that can be accomplished with your proposed system and a comprehensive design of that system. The goal of these analysis and design activities is to generate information that will allow you to start implementing the system with confidence.

Your deliverable for Project 2 is a document that contains the information listed below:

- **Project Summary:** What is the high-level overview of your semester project. What are you trying to accomplish? What will your system do when you are done? *This can be mostly copied from Part 1 that you submitted.*
- **Project Requirements:** Based on the project summary, what are the requirements and responsibilities for your system? List the requirements and their associated responsibilities in this section. In generating the requirements, you want to identify the main goals of the system and its associated responsibilities. This list may be short but it should attempt to convey the "big picture" view of your system and what it is trying to accomplish. The requirements in this list can shift from being descriptions of functional capabilities to discussing constraints (such as platforms, number of users, etc.) to discussing other non-functional characteristics of the proposed system.
  You can submit this as a list or table. Each item should be numbered for reference.

- **Users and Tasks:** How many different types of users will your system have? What tasks do they need to accomplish with your system. Document how the system will support each task via a use case. Try to think of the various problems that can occur while trying to accomplish these tasks and document them in the use case.
- **Activity Diagram**: Pick your most complex use case and create an activity diagram that documents all of the possible paths through it. Typically, one activity diagram corresponds to a single use case, but if you can think of a way to combine multiple use cases into a single activity diagram that's fine too.
- **Architecture Diagram**: Draw a diagram that shows the architecture of your system. Is it a web app? A mobile app? Does your mobile app interact with a web service? Does your web app interact with both desktop and mobile clients? etc. This diagram will be a boxes-and-arrows diagram that shows the various architectural components of your system (devices, databases, 3rd party services, etc.) If you have questions about this part of the assignment, you can take a stab at creating the diagram and then send it to us for review. Include in this diagram more information about the structure of your application and what frameworks it will be using. For example, if you're building an Android application, describe the activities you'll be creating and what Android services you'll be using (Media, Networking, Location, etc.). If you're building a web service or application, what framework will you be using, how will your service be structured, what request-response cycles will your service be supporting, etc. The goal of this task is to get you to delve more deeply into the frameworks you'll be using to implement your system and to think up front about what classes and services in that framework are the most relevant to your application. Note: not all of this information has to be conveyed in the diagram, you can augment the diagram with a textual description that goes into the details more deeply.
- **Data Storage:** Discuss how you will persist data in your application. What storage technology will you use? Text files? XML? sqlite? Where will the data be stored? Describe the classes that you will use to access this data at run-time.
- **UI Mockups:** Create screen mockups for the user interface of various parts of your application. What will a user see as they work through the tasks identified in your use cases? What is the overall organization of your user interface? How will data be displayed? How will the user navigate from screen to screen? Use this task as a means for focusing your thoughts about what you will actually be creating... iterate now on how your screens will be layed out and then include your final sketches in this section. I'm not going to place a limit on the number of screens you need to create for this section. Include what you think is needed to convey an overall sense of your application. Note: it is okay to work on paper for this task and then scan in your work to include in your document.
- **User Interactions:** Use your use cases and UI mockups to identify at least three interactions that your user will have with your application. For each interaction, describe how your system will support it. Then show a sequence

diagram of the objects that will participate in the interaction; some of these objects will represent UI widgets, some will be model objects used to access/update persistent data, and some will be objects that sit in between the UI and the persistent data. This latter class of object is known as a controller and they contain application logic that decides how to respond to events, updating both model objects and UI widgets to represent the new state of the system. Recall that sequence diagrams do not contain conditional constructs, so be sure to clearly describe the interaction that is being displayed in the sequence diagram. Do not try to show if-then-else scenarios in a single sequence diagram. If you find yourself needing to show such a situation, simply create two sequence diagrams, one that shows the true branch and one that shows the false branch.

- **Class Diagram:** Create a class diagram that documents everything you have gleaned from the other activities above with respect to what classes your system will contain: what relationships they have, what are their attributes and (public) methods, what design patterns are present in your design, etc. If it is too difficult to fit everything into a single diagram, you can split your classes across more than one diagram in whatever way makes the most sense for your particular application.

**How to Scope**

While I have asked for a non-trivial amount of information above, you should only generate the information you need to achieve your desired functionality given the size of your team. You will have four weeks to develop your system prototype (2 iterations consisting of 2 weeks each). During each iteration, each member of your group should expect to work on one to two use cases (possibly with other team mates). As a result, if you are a member of a two-person team, you'll want to target a 4-8 use cases for your system. A 5-person team is looking at a 10-20 use cases, although I would be surprised if any team identifies 20 tasks that their system can perform. Use these guidelines to scope the work of this project part and really focus on generating information that will guide your implementation efforts during the last four weeks of the semester.

The point breakdown of this project part is as follows:

| **Section** | **Points** |
|---|---|
| **Project Summary** | 10 |
| **Requirements** | 20 |
| **Use Cases** | 20 |
| **Activity Diagram** | 10 |
| **Architecture Diagram** | 20 |
| **Data Storage** | 10 |
| **UI Mockups** | 20 |
| **User Interactions** | 20 |
| **Class Diagram** | 20 |
| **Total:** | 150 |

## Assessment

In general,  you will be graded on the thoughtfulness and completeness of answering the questions above, along with the overall quality of the work.

## Submission

Please submit a single PDF document to Moodle containing all of your work.
Your PDF should be named Project2_identikey.pdf, where you replace identikey with your identikey.  All group members must submit the file.