CSCI-4448/5448: Project Description
Fall 2015

Objectives

- CSCI-4448 Work in a group of size 4-5 people
- CSCI-5448 Work in a group of size 2-3 people
- Develop an object-oriented software product

Semester Project Overview

All students in OOA&D are required to complete a group project. Time will be provided during class to discuss project ideas. Teams will be formed using catme.org based on schedules. You will receive an email from catme to fill out.

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
1: Topic and Team	5
2: Requirements & Analysis	60
3: Presentation	10
4: Finished Project, Report	25
5: Peer Evaluation	[*incorporated*]

Project Part 1

This is the initial part to your projects.
The goal for this part is to find your group, figure out your topic, and scope out what you intend to do. The instructor will then help ensure the scope of your project is achievable, or guide you towards an achievable project.

Submit to the course website a .pdf document with the following details (should be about 1 page or so, in this order, with these headings):

a. **Team:** Full name of each member in the group
b. **Title:** The title for your project (A very brief description)
c. **Description:** A longer description
d. Intended **platform/environment:** (Ruby on Rails, Mobile, etc.)
e. **Programming languages** to be used (and how familiar your team is with each)
   Note: You must use object-oriented language(s)!
f. **Functionality:** A bullet-list of functionality you intend to complete. For example, if you want to implement Google, you just don't have time or

resources to do so. But you could do parts of Google, such as indexing web pages and rudimentary search algorithms or the such.

Example (Use this template):

Team: Liz Boese, Cookie Monster, Superman

Title: Delivery Service Website as SaaS

Description: A delivery service website that clients can lease access to which keeps track of their customers, what gets shipped and how often. We will set up for one client only.

Platform/Environment: PHP Yii Framework integrated with MySQL

Programming Languages:

Language	Liz Boese	Cookie Monster	Superman
PHP	Expert	Beginner	Beginner
JavaScript	Expert	Intermediate	Guru
HTML	Guru	Expert	Guru
CSS	Expert	Intermediate	Guru

Functionality:

- Customers can sign up
- Customers can log in
- Admin can log in
- Admin can set up what products get delivered each week
- Admin can add products
- Admin can add customers

[OPTIONAL] **Stretch Functionality:**

- End world hunger
- Learn how to fly

Submission

Submit a single .PDF document named **TitleOfYourProject_Part1.pdf** to you group GitHub account containing all of your work.
Add a link to your group's GitHub project on the Google sites course website under "Your Projects".

Project Part 2

Part 2 asks you to engage in analysis and design activities for your semester project. 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 Part 2 is a single .PDF that contains the information listed below:

- **Team:** Liz Boese, Cookie Monster, Superman
- **Title:** Delivery Service Website as SaaS
- **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 a table. 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. Be sure to break them down into small manageable separate requirements.

  These need to be in a table. Each item must be numbered for reference.

  You can write the requirements in short sentences or in the Agile format.
  - 4 Separate tables for the requirements:
    - Business Requirements
    - User Requirements
    - Functional Requirements
    - Non-Functional Requirements

  If there are no business requirements then state so instead of a table.
  - [optional] you can add priority (Critical, High, Med, Low, Nice-to-have)
  - [optional] you can add topic/area (e.g., Login, Profile, DB, etc.)
  - [optional] you can add user(s)/actor(s) involved in each requirement

  Example

Business Requirements				
ID	**Requirement**	**Topic Area**	**User**	**Priority**
BR-001	All login userids must be the user's corporate email address.	Authentication	All	Critical
BR-002	Employee payments are distributed on the first of each month.	Payroll	Employees	High
BR-003	Only Admin can change a an employee's internal ID number	Admin Dashboard	Admin	Medium

- **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.
  Provide both use case diagrams and use case documents *(see below for approximately how many to do).*

- **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. *Note you must use UML 2.0 version with "swim lanes".*

- **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 laid 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. There are also free wire-framing tools online.*

- **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 you may already know about 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. Be sure to show the visibility modifiers and relationships between the classes (use either notation presented).

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 weeks to develop your system prototype (4 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). A 5-person team is looking at 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.

The point breakdown of this project part is as follows:

Section	Points
Project Summary	2
Requirements	5
Use Cases	10
Activity Diagram	10
Data Storage	3
UI Mockups	5
User Interactions	10
Class Diagram	15
Total:	60

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

Submit a single .PDF document named **TitleOfYourProject_Part2.pdf** to your GitHub containing all of your work.

Project Part 3: Presentation
Present and demo your final system.

A **demo** of the final system during the last week of class – to the rest of the class. You will have a limited number of minutes for your group, including set-up *(as soon as a group ends, your time begins).* Your demo should consist of two parts:

- A brief demo walk-through demo of one or two use-cases.
- Then focus on a particular design pattern or two, or the diagrams that you did to define the project. Describe what you applied and why, or what you could apply and how.
    - Consider what is interesting about your project that the rest of the class can learn from.
    - Do not try to show everything, just focus on some part of the whole to share with the group.

General presentation tips:

- Use 20-point font minimum
- *Pictures say a thousand words* much easier to convey information than lots of words on each slide
- Face the audience when you speak
- Make sure we can read your content. Don't show the whole class diagram; zoom in to some classes showing a particular design pattern to discuss.
- Include title and group member names!
- Everyone must speak during presentation.

You must also record a demo of your software and post it to your GitHub. You can use this video during your presentation.

Submission
Sign up for a time to demo in Moodle.
Submit a single .PDF document named **TitleOfYourProject_Part3.pdf** with your slides and a link to a demo of your software to your group's GitHub.

Project Part 4: Final Report
Project Part 4 asks you to complete the work on your semester project.

The required deliverable is a **report** that documents the final state of your system:

1. What features were implemented and a class diagram showing the final set of classes and relationships of the system. *(This may have changed from what you originally anticipated from earlier submissions).* Discuss what changed in your class diagram and why it changed, or how it helped doing the diagrams first before coding if you did not need to change much.
2. Did you make use of any design patterns in the implementation of your final prototype? If so, how? If not, where could you make use of design patterns in your system?
3. In addition, the report must discuss how the final system changed from the design you presented in Project Part 2. In particular, include the class diagram you submitted for Project Part 2 and use it to compare and contrast with the class diagram that represents the final state of the system.
4. What have you learned about the process of analysis and design now that you have stepped through the process to create, design and implement a system?

Submission
Submit a single .PDF document named **TitleOfYourProject_Part4.pdf** to your group's GitHub containing all of your work.

Project Part 5: Peer Evaluation and Team Dynamics
Each person individually fills out the peer evaluation form (see Moodle) and submits into Moodle as an Excel spreadsheet (do not convert to .PDF). This will be incorporated into the individual project grade for groups that have someone not helping much at all or someone who put considerable extra effort helping others and/or developing the project as a good team player. *However, taking over the project and not being a good/helpful team player is bad!*

Please note: Giving everyone the same number does not count as filling out this form!
You will need to fill in the comments to support your evaluation, as well as the Team Dynamics question. For Team Dynamics, enter the letter a, b, or c for "Which one" to specify which question you are answering.