Project 4: The Semester Project for OOAD – Part 1: Design Phase

Introduction

Project 4 is the first of three parts for your Semester Project. The plan right now is as follows:

- Project 4 – the Project Design deliverables – will be due on Wed 11/4 – worth 100 points
  - This part of the project is detailed below
- Project 5 – the first Semester Project Sprint is due Wed 11/18 – worth 75 points
  - This will be an interim report and demonstration to show where the project stands after your first two weeks of code development
- Project 6 – the second/final Semester Project Sprint is due Friday 12/4 – worth 100 points
  - This is the final delivery of your project with demonstrations

The Semester Project

Create a non-trivial set of classes and services that make use of some of the design patterns we cover to provide a nice set of functionality to the end-user; we like to see some UI design and data handling, but your project may vary. You may use any languages, frameworks, libraries, or utilities for your project as long as the project demonstrates object-oriented design.

Reminder on Project Scope Approach

Project 4 asks you (and your team) 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 provide 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.

While I am asking for a non-trivial amount of information, you should only generate the information you need to achieve your desired functionality given the size of your team. You will have a about four weeks to develop your system (2 iterations consisting of about 2 weeks each).

During each iteration, each member of your group should expect to work on one to two use cases (possibly with other teammates). As a result, if you are a member of a two-person team, you'll want to target 4-8 use cases for your system. A 3-person team is looking at 6-12 use cases (12 is a lot!).

Use these guidelines to scope the work of this assignment and really focus on generating information that will guide your implementation efforts during the last part of the semester.

Project 4 Deliverables

Your deliverable for Project 4 is a single PDF document that contains the information listed below:

Project Summary

- What is the Project called?
- Who is on your team (include all names)?
- 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?

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. 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 can include functional capabilities (what it can do), constraints (such as platforms, number of users, etc.), and non-functional characteristics (performance, security, usability, etc.).

Users and Tasks: Use Cases (Text or UML Diagrams)

- 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 by providing a use case (text or UML).
- Try to think of the various problems or variations that can occur while trying to accomplish these tasks and document them in the use case.
- Remember the WAVE rule when examining your use cases.

UML Activity Diagram

- Pick your most complex use case and create a UML Activity diagram that documents the possible paths through it.
- Typically, one activity diagram corresponds to a single use case, but if you decide to combine multiple use cases into a single activity diagram that's fine too.

Architecture Diagram

- Draw a (non-UML) 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.) at a high level. See examples of Architecture Diagrams in the UML lecture.
- If you're building a web service or application, what frameworks 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 elements you'll be using to implement your system and to think up front about what services in those elements are the most relevant to your application.
- Note: not all system information has to be conveyed in the diagram, you can augment the diagram with a text description or comments that go into the details more thoroughly if needed.

Data Storage

- Discuss how you will persist data in your application. What storage technology will you use? Text files? XML? CSV? MySQL? Where will the data be stored? Describe the classes that you will use to access this data at run-time. A diagram of tables and relations (like an ER diagram, or something less formal) or the classes that will manage data may be used here.

UI Mockups/Sketches

- 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.
- There is no fixed number of screen sketches. 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. For an automated tool, I highly recommend Balsamiq at https://balsamiq.com/

User Interactions/UML Sequence Diagrams

- Use your use cases and UI mockups to identify at least three interactions that your user will have with your application. (If your system is more about modelling internal interactions than external user ones, this should be used to model those interactions.)
- For each interaction provide both a text description of how your system will support it and a UML 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 normal branch and one that shows the off-normal branches.

UML Class Diagram & Pattern Use

- Create a detailed class diagram that documents everything you have discovered in your design activities with respect to what classes your system will contain: what relationships they have, what are their attributes and (public) methods, what design patterns are used, 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 application.
- Clearly this project is a great opportunity for looking for some experience reuse from the OO patterns library. Ideally, include at least three patterns as part of your project design. However, if you have difficulty identifying pattern use for your application, please reach out to class staff for guidance or further discussion of the requirement.
- For this delivery, you should include your concepts around using applicable patterns in your UML Class Diagram for the system above and highlight the patterns in the diagram.
- In your final delivery you will be asked to document and discuss what OO patterns or variations you considered or used, and how they impacted your design.

Grading Rubric

The point breakdown of this assignment is as follows:

Section	Points	Comments
Project Summary	10	Include your team's names!
Requirements	15	Main goals and other requirements
Use Cases	15	**UML or text use cases of users interacting with your program**
Activity Diagram	10	For most complex use case(s)
Architecture Diagram	10	Focus on components of system
Data Storage	10	Include tool and data description
UI Mockups/Sketches	10	Number of sketches varies
User Interactions	10	At least three sequence diagrams
Class Diagram	10	One or more describing system, note pattern use
Total	**100**	

- There is a **possible 10 point overall quality bonus for each submission**, based on assessment by the graders and professor.
- For the UML Diagrams, you can use a scan of a paper or whiteboard diagram, or use your favorite UML tools, such as Draw.IO. If done on paper/pencil or whiteboard, please be sure the diagrams are readable and clear.
- A single PDF from your team is Due Wednesday Nov 4 at 12 Noon on Canvas.
- Assignments will be accepted late for one week. There is no late penalty within 4 hours of the due date/time. In the next 44 hours, the penalty for a late submission is 5%. In the next 48 hours, the late penalty increases to 15% of the grade. After that point, assignments will not be accepted.
- E-mail or use Piazza to contact the class staff regarding homework questions or assistance. Please contact the class staff EARLY in the cycle for questions, clarifications, or variations for your project.