Project 5: The Semester Project for OOAD – Design Phase

Introduction

Project 5 is the first of three deliverables for your Semester Project. The plan right now is as follows:

- Project 5 – the Project Design deliverables – will be due on Wed 3/30 – worth 100 points (with possible bonus)
  - This part of the project is detailed below
- Project 6 – the first Semester Project Sprint is due Wed 4/13 – worth 75 points
  - This will be an interim submission including an in-person demonstration and review to show where the project code stands after your first two weeks of code development
  - Expect to deliver at least 50% of the functionality of your code here
- Project 7 – the second/final Semester Project Sprint is due Wed 4/27 – worth 100 points (with possible bonus)
  - This is the final delivery of your project code and support with recorded demonstrations
- Note that Projects 5, 6, and 7 are not divided into Part 1 and 2 deliveries – one delivery per project

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 functionalities to an end-user; we like to see some UI design, business logic, and data handling (MVC?), but your project may vary per your proposed topic. 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 5 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 four weeks to develop the code for your system (2 iterations consisting of two weeks each).

During each iteration, each member of your group should expect to work on one or maybe two major functionalities or use cases (possibly with other teammates). As a result, if you are a member of a two-person team, you will want to target 4 to 8 use cases for your system. A three-person team is looking at 6-12 use cases (12 is a lot!). To be clear, I would rather see quality designed and delivered code deliveries than a larger volume of poor work.

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 5 Deliverables

Your deliverable for Project 5 is a single PDF document that contains the information listed below. Clearly indicate each section in your PDF as they are graded separately:

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 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 goals, 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 all major user tasks by providing one or more use cases (using a text template or UML).
- Try to think of the various problems or variations that can occur while trying to accomplish these tasks and document those alternative situations in the use cases.
- Remember the WAVE rule when examining your use cases.

UML Activity or State Diagram

- Depending on your application, select use of either an activity or a state diagram to show your system's operations and the paths through it.
- Highlight or annotate in your diagram where your use cases are being serviced by the functionality being described.

Architecture Diagram

- Draw a (non-UML) diagram that shows the layered architecture and components 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.
- If you are 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.
- 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 typical Architecture Diagrams in the UML lecture.

- The goal of this task is to get you to delve more deeply into all the components 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? Hibernate? Where will the data be stored? Describe the classes and libraries that you will use to access this data at run-time.
- Note that this section is required even if data is simply in memory.
- Any reasonable diagram of data details, tables and relations (like a Logical Data Model or Entity-Relationship data diagram), or a diagram of 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 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 or a whiteboard for this task and then scan in your work to include in your document. For an automated mockup tool, I highly recommend Balsamiq at https://balsamiq.com/
- If you do not have a UI in your application, see Bruce for alternate design activities

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 responsibilities and relationships they have, what are their attributes and methods, what design patterns are used, etc.
- Please include return and argument types as well as accessibility in classes.
- 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. **Your system must include use of at least four patterns** as part of your project design. If you have difficulty identifying pattern use for your application, please reach out to class staff for guidance or further discussion of the requirement.
- Highlight or annotate the patterns in the UML class diagram.

User Interactions/UML Sequence Diagrams

- Use your class diagram, use cases, and UI mockups to identify at least three primary 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.
- For an MVC style system, some of these objects could represent UI widgets, some could be model objects used to access/update persistent data, and some could be objects that sit in between the UI and the persistent data as controllers. The controllers 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 parts for the sequence diagrams, one that shows the normal (happy path) flow and one that shows one or more off-normal flows.

Grading Rubric

The point breakdown of this assignment is as follows:

Section	**Points**	**Comments**
Project Summary	10	Include your team's names!
Requirements	10	Main goals and other requirements
Use Cases	10	UML or text use cases of users interacting with your program
Activity or State Diagram	10	At least 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	15	Number of sketches varies
User Interactions	10	At least three interactions in sequence diagrams
Class Diagram	15	One or more describing system, note pattern use
Total	**100**	

- There is a **possible 5 or 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/diagrams.net. 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 3/30 at 8 PM on Canvas.
- Assignments will be accepted late as follows: there is no late penalty within 4 hours of the due date/time. In the next 48 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 are not 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.