CS 542 Final Project [100 points]
(Due by 12pm on May 19th)

Objectives

Practice object-oriented design with design patterns

Have some fun being creative!

Overview

The final project is the delivery of an executable software system that makes use of at least **5** design patterns and is implemented in C++.

General Requirements

You may either work individually, or in two-person teams.

You are asked to submit both your code and a final project document.

The final project document must include the description of the real-life problem your team is going to solve. The document must describe the intended use of the system, its overall functionality and its main architectural components. Also note in the document that whether you work by yourself, or in a team of two (with names of the member given) in this project. The final project document should be 1-2 pages.

The final project documents must include the detailed analysis and design, and demonstrate the appropriate use of object-oriented design techniques and design patterns, in particular

a. The design should identify all classes and objects for the software project, and their relationship to each other. The use of Unified Modeling Language (UML) Class Diagrams is required.

b. The design should demonstrate the use of **5** or more design patterns in different situations. Among the patterns, you need to use at least one Structural pattern and one or more Behavioral patterns. For each additional appropriate use of pattern beyond the required **5** patterns, there will be 2% bonus toward your final total score.

c. Each decision, the use of class, objects, and design patterns, and the relation between these items, should be explained. You should also include with a brief discussion of how these patterns participate in an intelligent solution to the original problem. For example, you should come up with a legitimate business concept that will benefit from your selection of patterns.

Implementation

Implement the main functionality of the system in C++. Separate the class declarations in the header files and the implementation details in the source files. Include in your code appropriate comments.

Submission

SUBMIT ALL THE FILES IN A ZIP FILE TO COUGAR COURSES:

Always make sure the files you submit can be compiled on empress.csusm.edu.

- Program1: Submit **all the source code files** in a zip file to cougar course.

Note: Compress all the above files in a zip file, name it with your name e.g. FirstLast.zip, and submit the zip file on Cougar Course.

Grading

1. On Cougar Course, submit all the files in a zip file with your name. Otherwise, we will not grade it.
2. Your code should be compiled on Cougar Course. If there is a compilation error, you will get 0 points.
3. Your code counts for **80%** of the total points.
4. Your project document counts for **20%** of the total points.
5. Late submissions that past the due date are not acceptable.
6. Duplicate submissions, except from your team member, are not acceptable and will get 0 points.
7. If you use any code, design, and other materials from others, you should explicitly reference and give credit to the original resources in your project document. Otherwise, you will get 0 points.

Sample Projects

1. Library Management System:

If an admin logs into the system, he should be able to access the database with all the privileges such as addition/modification/deletion of book & its details, assigning the book to any student, renew and entry of return, etc.. If a student logs into the system, he should be able to search the books and know its details like author, publisher, publishing date, availability, location at the library, etc.

2. Personal Finance Manager:

The system will allow any number of users to be created with login credentials. Each user will have separate data storage space. User will enter his daily spending to the system with details like date of purchase, shop name, product name rate etc. And he should be able to get the summary of his spending by month wise or year wise, etc. He should be able to analyze between months or the same month of two years, etc. Usage of database is preferable. But still you can use flat file. If you do this now, you can enhance your application with your “Data Mining” knowledge later.

3. Online Banking System:

The system will be accessed by two different people – Admin and account holder. Admins should be able to open new account (both current & savings) with customer details. Account holders will be able to see the current balance providing the account number. The customer should be able to check the balance, add a beneficiary, transfer money to a beneficiary, etc. You can think of all the necessary functionalities to be implemented.

4. Personal Calendar:

The application will get the event details with time and store it in a file or database. You should also get the remainder details like when the user wants the remainder for this event. Viewing, modification and deletion of events are obvious requirements. You need to write a background service

which will pop up the remainder at the right time.

5. Plagiarism Calculator:

The application will accept a document and place it in a folder. For any new incoming document, it will compare with each file in that folder and tells the user how much it resembles. You can have a threshold limit like 30%. If it is less than the threshold, the application will not do anything. If it crosses, it will alert the user with the document title which resembles the new one. The complete system will involve information retrieval concepts. But you can design a simple one but up to the standards.