SOFTWARE ENGINEERING
DESIGN PATTERNS PROJECT

Directions

You will work in teams of 4.
Pick 4 design patterns (1 for each team member) that are **not** from any I covered in the design patterns lecture (Singleton, Observer, Command, Flyweight, or Factory).
You additionally may not choose the Façade or the Lock patterns -- they are too easy for this assignment.

For each pattern:

- Name the pattern.
- Identify the pattern category (creational, structural, behavioral, or the mystical concurrency).

In YOUR OWN words…

- Describe the pattern in natural human language.
  Describe the theory, components, and nitty gritty of how the pattern works.
  Use metaphors/analogies or examples.
  Make it accessible, make it interesting, but most of all make it yours.
- Describe the situation(s) in which you might use the pattern.
- Identify advantages to using the pattern (at least one).
- Identify disadvantages to using the pattern (at least one).

- Implement a small example of the pattern in C# in original code, also commented by you:
  - The code must compile and run, although it may not do anything particularly interesting. I'm only concerned with you showing me you understand the pattern with a "toy" implementation.
  - If you use Visual Studio and want to include your entire solution and corresponding projects, that is fine. Otherwise just including your .cs files is fine.
  - Place your implementations in separate files.
  - Note that this must be **your** implementation! I'm not interested if you can copy and paste an example from online. If you need to learn from another implementation, great, but write YOUR own example from scratch.

- Provide UML Diagrams:
  - A general UML diagram of the pattern.
  - A specific UML diagram of the pattern **based upon the code you wrote**. Use the general UML diagram as a guideline to ensure you implemented it correctly, and verify it follows the premise of the general diagram.
  - Note that different design patterns may need different diagrams. A Class diagram showing the user defined types (aka UDTs) involved as well as their members and relationships will suffice for most patterns. However, some patterns are better explained with a Sequence or even an Activity diagram. Choose one of those diagram types to show the pattern.
  - You must draw the diagram yourself instead of relying on tools to generate it for you. Either draw it (neatly) on paper and take a picture of it or use software to help create it.

        - Some software can create the diagram completely for you from your code. I don’t want an auto-generated diagram in your submission.

Note
You are to submit original work – your own descriptions, implementation, and diagrams.
Teammates, your name is going on this. Verify one anothers’ 100% original work.

There are thousands of existing examples to learn from for these patterns.
I support and even endorse you accessing and learning from these works – O’Reilly Media publishes source code for all of their books, and it is my go to when learning anything new.
However, do NOT grab and submit this content.
I want 100% original work in your submission, completed by you and your team.

Put everything in your own words.
You will have to learn complicated material and present it in accessible terms that you think up.

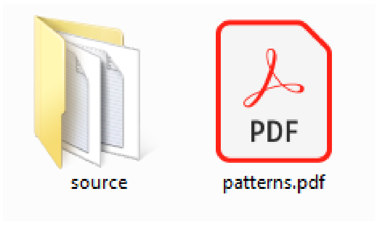
I’m not interested in reading code or text copied and pasted from ANY source.
Create your OWN code from scratch so I know you can implement the pattern.
These are patterns (they show up everywhere), so feel free to read all of the code in the world.

Then write your own.
I reserve the right to call you into my office to explain every line of code.

Draw your OWN UML diagram based on the code you wrote.
Only the general diagram should look anything like another source, and even then, I’ll be looking with a microscope.
Any attempts to copy and paste from ANY online source will result in a zero for that part of the rubric.

Submission
- Submit a single pdf for your group called "patterns.pdf".
- In this pdf, show everything for the first pattern followed by everything for the second (i.e. don't interweave them them)
- Be sure to label each part. For example:
    Name: Singleton
    Category: Creational
    Description: …
    When to Use: …
    Advantages: …
    Disadvantages: …
    General UML: [include diagram]
    Specific UML: [include diagram]

- Save patterns.pdf in a folder next to the folder containing your C# code. It should look like the following:



- The “source” folder should contain code for each of the 4 design patterns, clearly separated so I know which goes with which pattern.
    - For example: having a file “MVC.cs” and a file “Multiton.cs” or having a directory “MVC” containing all Singleton code and another directory “Multiton” containing all Multiton code.
- Zip up the folder containing the patterns.pdf and source directory and submit that zipped folder to Moodle.

Rubric (25pts per pattern, 100pts total)

- Per pattern:
    - (1pts) The name of the pattern. That’s right. You get a point for just picking any pattern.
    - (1pts) Stating the category
    - (3pts) Your plain English description of the pattern with an analogy or metaphor
    - (2pts) When to use the pattern
    - (2pts) Advantages
    - (2pts) Disadvantages
    - (3pts) General UML diagram
    - (5pts) Specific UML diagram
    - (6pts) C# Implementation
- For 4 total patterns – 100 points, a shared group grade.