Passion Project - 15% of your final grade

Sec A - due Feb 12 @ 11:59pm
Sec B - due Feb 15 @ 11:59pm

Content Management Systems (CMS) are incredibly flexible, and are the basis of most web applications. Your task is to create a CMS-based project which reflects your passion. You will be evaluated on these key areas:

- Project Scope
- Data Architecture
- Code Hygiene
- Presentation

Project Requirements

Project Scope - 4% of final grade

- Your idea is inspired by something that interests you. Choose something interesting, but don't feel pressured to do something which has never been done before.
- Your choice of features for your project are challenging but not overwhelming given the time.
- You have a clean and descriptive wireframe of exactly what your project flow will be like (before you build it).
- You have some ideas of what you could do with more time, and your project is built with extensibility in mind.

Data Architecture - 4% of final grade

- Your system should have a minimum of two tables (Models) to represent your content.
- Your system has ties to relational data architecture. These concepts are commonly referred as: One to One, One to Many, Many to Many. I am expecting a minimum of two tables (Models) programmed with this architecture in mind.
- The system is semantic (i.e. If you are linking two tables by a One to Many relationship, you should be able to offer evidence supporting your decision).

Code Hygiene - 4% of final grade

- Code is hosted and maintained by a repository.
- **Code is not copied, in any way, from: online references; peers; and examples done in class.**
- Variables are appropriately named.
- Model-View-Controller (MVC) architecture pattern is followed.
- Use of Entity Framework to represent data.
- Comments are used to describe how the code works in your own words.

Presentation - 3% of final grade

- You attend class on presentation day and present your project.
- Avoid pacing, rushing, backtracking, mumbling, shouting, etc.
- Project is aesthetically pleasing.
- There are concrete use cases (a.k.a. “Narrative”) which are supportive of your tool.
- Ability to handle unexpected occurrences (like a bug in your code) well.
- “Show that you know code without showing code”.