Project Name

Student Name and Number (one per line)

You can email me the milestone and send me the link to your fork of the GitHub repo you’re working with. (-1 mark if there is no link to your fork)

# Summary of Project

Cut and paste the 1 paragraph summary of what the project is about.

# Class Diagram of Actual System

(paste in conceptual diagram = ideal architecture from M2)

(create UML diagram of classes of interest, 2-4 pages, if I can’t read it you will lose 3 marks)

1 to 2 pages of text -> Describe the diagram and the relationships between the classes. Tell me the purpose of each important class and describe its relationships to other classes (4 to 10 classes). Map the conceptual classes to actual classes and describe why there are discrepancies and what the impact of the differences have on the system? Describe any reverse engineering tools used (ObjectAid UML Explorer, Enterprise Architect - Sparx Systems, ArgoUML, etc).

In maximum of one page, for two classes and the relationship between them: Copy-and-paste the class, method, and attributes **declarations** (and anything else that is necessary) directly from the source code. Do not include code that is unnecessary (be selective, you will lose marks for large dumps of source code.)

# Code Smells and System Level Refactorings

Be idealistic! Restructure the system to fix some of the code smells you identified above.

Describe the code smell(s) and how you will combine together a series of refactorings to fix the smells. Do not describe generic refactorings (eg do NOT do this: “Class X is a god class, so we should extract related methods”). Describe how the refactorings are interrelated and how they correct the problems you identified in the above. Be specific! For example, “First, I moved the methods X and fields Y and Z, to increase cohesion and to reduce feature envy. Second I renamed the class to reflect its new limited responsibilities. Third I removed the coupling to … Fourth, I created a superclass and pulled up methods …”

1 to 2 pages of text + 1 to 2 UML diagrams (diagrams should be at most 2 additional pages).

# Specific Refactorings that you will implement in Milestone 4

Be realistic, you have to implement at least two of these! Suggest two to four refactorings that you will implement in Milestone 4. In maximum of one page, for one of the refactorings you suggest: Copy-and-paste the class, method, and attributes **declarations** (and anything else that is necessary) directly from the source code. Do not include code that is unnecessary (be selective, you will lose marks for large dumps of source code.)

Notes:

1. Algorithmic changes are not acceptable refactorings. You must deal with the logic and relationships between the classes -- ideally real world or domain entities.
2. Stay away from the GUI classes as they often contain a lot of autogenerated non-domain code.
3. Just moving code around without modifying it is unacceptable.
4. Isolated changes that do not affect other parts of the system are unacceptable.
5. Example of possible refactorings: Refactoring 1, fix some complex if statements. Refactoring 2, introduce a strategy pattern and use delegation to keep the system running. Gradually remove the delegation on some of the cases.
6. You can use delegation to have the old methods call the new methods to keep the system running.