Project Milestone #1: Domain Model

Due Jan 17 by 4:59pm **Points** 5 **Submitting** on paper

M1: Domain Model (Conceptual UML Class Diagram)

In preparation for the class project, construct a ***conceptual*** UML Class Diagram that models the Domain Concepts described in the **Course Project** overview (*Users*, *Statuses*, *Feeds, etc.)*. Capture attributes that might exist in each class. You should lean towards associations (instead of attributes) for anything that is not a simple, primitive value. You do **NOT** need to include operations for classes. Capture the relationships between classes. Label the UML Class diagram correctly with appropriate symbols to represent association, multiplicity/cardinality, composition, aggregation, etc as needed. Include a note for every class and complex relationship that needs explaining to describe what it is to another person that wants to review your diagram.

Use an appropriate software choice to create the diagram. For example, LucidChart, yUML, etc. We recommend using LucidChart **(https://www.lucidchart.com/)** . It provides the correct symbols and styles we are looking for. You can sign up for an education account that will allow your account to have sufficient space to create documents for this class.

For definitions of each term, refer to the Project Overview Documentation.

Submission:

- Submit a paper copy of your diagram to the TA in room 1058 in the basement of the TMCB
- (Optionally) You may resubmit your class diagram with any corrections within 72 hours of getting the graded diagram back, for the possibility of getting half of the points you lost back.

Requirements:

The following checklist of requirements will be used by the TA in reviewing your UML diagram.

- Have appropriate classes: User, Status, Feed, etc.
  - Classes can have attributes but favor associations for non-basic types (eg, int, string)
  - Anything that is not apparently clear has a note describing its purpose
- Has a relationship or class capturing the *follows* relationship.
- Has associations between classes that are related
  - Each association has a name (with a direction on that name label, if it is helpful)
  - Each association has multiplicities/cardinalities
- Includes at least one composition or aggregation relationship.
  - Each relationship uses the correct symbol
  - Each relationship uses the correct direction

- Is well organized, neat, and readable