UNIVERSITY OF NORTH TEXAS
COLLLEGE OF ENGINEERING
COMPUTER SCIENCE & ENGINEERING DEPARTMENT

CSCE 5430 – Software Engineering
Spring 2023

Project 1 -Phase 2

Due Date for Project1 Phase2: Friday March 3, 2023, at 5:00 pm

1 Overview

In Project1 Phase1, we described the functionality of iFINANCE (the function model) using use cases. In this Project1-Phase2, we will refine the functional model and derive the analysis object model. The analysis object model focuses on the individual concepts that are manipulated by the system, their properties and their relationships. The analysis object model, is depicted with UML class diagrams, includes classes, attributes, and operations.

As you go through this Project1-Phase2, remember to keep up to date the project glossary and your project workbook revision history.

2 Goals

At the end of this Project1-Phase2, you should be able to:

- Identify analysis classes representing entity, boundary, and control objects.
- Identify relationships and multiplicities between analysis classes.
- Show analysis classes, relationships and multiplicities on a class diagram.
- Find analysis class attributes.
- Update the project glossary.

3 Identifying classes

3.1 Produce a List of Candidate Classes

Add a new section in your workbook titled 'List of Candidate Classes'. Read through the iFINANCE problem statement given in Project1-Phase1 and examine your (or the sample) solution for Phase1 (Section iFINANCE System Use cases) and list the nouns that you come across. This list forms your initial set of candidate classes.

Add any classes that you discovered during your earlier brainstorming. (Since you were thinking about the possible system when you produced the use case diagram during your activities in Project1-Phase1, they may well be relevant.)

3.2 Filter your Candidate List

Examine each candidate class and reject it if it is:

- The system itself, this will become many separate classes.
- Does not house a set of attributes that will take on different values.
- Would not have many instance objects.
- Outside the scope of the application domain.

- Trivial: This could be a trivial type (such as a string) or one that's likely to become an attribute (such as an UserID).
- Does not have a clear statement of purpose in the system.
- Does not house a set of operations (what does the class do?)

While you're considering which classes to keep and which to reject, further classes will occur to you: add these to the list of candidates and then apply the same filtering rules to them.

Re-examine the filtered list and classify the candidate classes into three types,

- entity,
- boundary or
- control classes.

Add a new section in your workbook titled 'Potential iFINANCE Classes'. For each class that has survived your filtering process, in this new section, tabulate its information as follows (this will make it easier to prototype your class diagram in the next step).

Class name	Class Type	Brief Description

4 Connecting classes using relationships

In this section you will consider ONLY the entity classes you identified above. Add a new section in your workbook titled 'Potential iFINANCE class diagram' to include your works as per the following requirements.

4.1 Connect Related Classes

Draw relationships between pairs of entity classes that seem to be strongly connected. At this stage, just show inheritance and association – you can refine the associations in the next step if appropriate.

If there is a candidate for inheritance, consider introducing abstract classes as necessary (for elegance and for future extension). Whenever you are tempted to use inheritance, make sure that it satisfies the following criteria:

- It makes the diagram clearer and therefore easier to understand (especially since nontechnical customers may see the diagram).
- It is not there to share implementation (sharing implementation is a design issue, not an analysis issue).

If you find a class that is associated with an existing relationship between two other classes, consider making it an association class.

Note that, you need to develop a separate class diagram (entity classes only) for each use case defined in Project1-Pahse1. Please consider the UCs given in the sample solution for Project1-Pahse1.

4.2 Refine the Associations

Some of your associations may be modelled more precisely as aggregation. Can you find any examples?

Do you think any of your aggregations qualify as compositions? (Hint: it's probably too early to tell at this stage.).

4.3 Describe the Relationships

For every relationship, except inheritance, you should now consider adding some kind of description:

- An association name, describing what the association represents.
- An association end names (role name), describing the source and/or target end names.

(You don't need to provide all three – provide just the association name or the role names, as you find appropriate)

4.4 Add Multiplicities

Specify the number of objects involved in each relationship (except inheritance, which is implicitly one-to-one). Specify multiplicities in one of the following ways:

- *m* – Exactly *m* objects.
- *m..n* – From*m* to *n* objects, inclusive.
- *m*..* – At least *m* objects.
- * – Any number of objects, including zero (it is shorthand for 0..*).

5 Update the project glossary

Once you have agreed upon your basic classes and class diagrams, some technical terms may be discovered. Add these new terms accompanied with short descriptions to your glossary.

6 Adding attributes

Add a new section in your workbook titled 'Revised iFINANCE class diagram". Now look for the attributes (properties) for each class of object (Entity classes only). Record each class's attribute as follows:

Attribute name	Attribute Type	Brief Description

Specify the type of each attribute, as appropriate. Stick to a simple set of types for your attributes, such as the Java primitives (int, float, char, boolean) and simple classes (e.g. String). If you're unfamiliar with other Java types, ask the TAs and your instructor.

Avoid recording attributes that can be derived from other attributes (for example, a Circle has a radius and a diameter, but we could pick either as an attribute and the other would be derived).

Revise your class diagram to include each class's attributes and add your new diagram in this workbook section.

7 Hand in

By completing all the requirements in this Project1-Phase2, submit one workbook per group through the Canvas by the due date mentioned above.

8 Grading

Description	Points
Update Table of Contents	5
List of Candidate classes	20
Potential iFINANCE classes	20
Potential iFINANCE class diagram	25
Adding Class attributes	20
Revised iFINANCE class diagram	10
Total	100