WESTERN UNIVERSITY
FACULTY OF ENGINEERING
DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

SE2203 –Software Design

Assignment 3

Due Date: March 20, 2020

1 Overview

It is highly recommended to use the given sample solution for assignment2 (the workbook and the Java code) and consider it as the base for your works in this new assignment. Don't forget to change the header/footer information and the document copyright to your information.

In assignment 2, we described the functionality of iCLINIC (the function model) using use cases. In this assignment, 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 assignment, remember to keep up to date the project glossary and your project workbook revision history.

2 Goals

At the end of this assignment, 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.
- Generate the database scheme from the Entity class model.
- Build JavaFX prototypes for a number of data entry forms.
- Update the project glossary.

3 Assignment works

3.1 Question 1 - Produce a List of Candidate Classes (assesses the ET1 CEAB indicator)

Add a new section in your workbook titled 'List of Candidate Classes'. Read through the iCLINIC problem statement given in assignment2 and examine your (or the sample) solution for assignment2 (Section iCLINIC 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 assignment2, they may well be relevant.)

3.2 Question 2 – Filter your Candidate List (assesses the ET1 CEAB indicator)

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 iCLINIC 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

3.3 Question 3 – Connect Related Classes (assesses the PA3 CEAB indicator)

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

3.3.1 Develop UML Class diagram

Start by drawing the 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.

3.3.2 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.).

3.3.3 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)

3.3.4 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..*).

3.4 Question 4 - 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.

3.5 Question 5 - Adding attributes (assesses the PA3 CEAB indicator)

Add a new section in your workbook titled 'Revised iCLINIC 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). 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). Use your class diagram to create a new diagram to include **each class's attributes** and add your new diagram in this workbook section.

3.6 Question 6 – Enhance/develop your iCLINIC prototype (assesses the PA5 CEAB indicator)

In this assignment, you will consider the provided sample code solution for assignment2 and build extra functionalities on it as per the following requirements.

3.6.1 A mapping between the analysis object model and the database schema

- For each class in your entity class diagram, write a model class ~~and an adapter class as we did for the tables Teams and Matches in Lab5~~. The model class should define all the attributes included in your class diagram in addition to the required set and get functions. ~~The adapter classes are responsible to create the corresponding tables in the JavaDB database, and provide the actual read and write methods to the database.~~

3.6.2 A data entry forms for Manage iCLINIC Workers

- Use the skills you gained from Lab5 activities to develop two JavaFX forms and their controllers. The first form for add new worker data and save it into the database, the second form is to edit/delete a selected worker from the database.
- Use your imagination to determine the proper look-and-feel of your forms in terms of the number/type of the fields and the required buttons.
- Attach the first form to the menu option: Manage Users → Add New Worker.
- Attach the second form to the menu option: Manage Users → Edit/Delete Worker.

~~3.6.3 A data entry forms for Manage Patients Records~~

- ~~Develop two JavaFX forms and their controllers. The first form for adding add a new patient record data and save it into the database, the second form for modifying an existing record or delete an existing patient record from the database.~~
- ~~Design the proper look-and-feel of your forms in terms of the number/type of the fields and the required buttons.~~
- ~~Attach the first form to the menu option: Patients Records → Add ne Patient.~~
- ~~Attach the second form to the menu option: Manage Employee → Edit/Delete Patient.~~

4 Hand In

1. Using IntelliJ IDE export the iCLINIC project to ZIP and name it *yourUwoId*_Assignment3Prototype.zip.
2. Submit this zip file along with your solution workbook using OWL assignment link at the due date mentioned above.