TOI-OHOMAI
Institute of Technology

COMP.6215 – GUI Programming

Assessment 01

Due Date: September 12th 4pm

Assessment	Type of Assessment	Due Date & Time	Marks	Weighting
Assign #1	Coding Assignment	12th September 2019 4pm	See Rubric	40%
Learning outcomes assessed:				
1. Demonstrate competency in the use of a GUI programming language and environment. 2. Demonstrate the use of a GUI language for the creation of an effective front end. 3. Demonstrate the testing, debugging and documentation of GUI programs.				

Marks:

Task Number	Your Mark	Total Mark
1		
2		
3		
Total		

Reminder

The assignment must be a product of your own work, except for the use of resources supplied with the course, discussions conducted with the lecturers, and other assistance shown as acceptable in the section *Assistance to Other Students* below.

Assistance to other students

Students themselves can be excellent resources to assist the learning of fellow students, but there are issues that arise in assessments that relate to the type and amount of assistance given by students to other students. It is important to recognise what types of assistance are beneficial to another's learning and also what types of assistance are acceptable in an assessment.

Beneficial Assistance

- Study Groups
- Discussion
- Sharing reading material

Unacceptable Assistance

- Working together on one copy of the assessment and submitting it as your own work
- Giving another student your work
- Copying someone else's work
- Changing or correcting another student's work
- Copying from books, the Internet etc. and submitting it as your own work

Helpful hints

Although you don't need to submit the diagram designs, it is strongly recommended that you make use of flow diagrams and UML diagrams to design your application before you start coding. This will save you a lot of time.

Assessment outcome

Upon successful completion of this assessment you will have a good understanding on how to separate UI logic from the business logic of the application. You will also have a good understanding of the solid principles.

Individual Assessment

This assessment is an individual assessment. You are to create a product of your own, but are able to work with other students as per the guidelines above.

Submission Type

You will receive a link to GitHub Classroom where you can submit your assignment. You will need to put all your apps inside this single repository.

Deliverables

You are to create 6 applications (3 x 2) where the 3 applications have are executed 2 times using a different UI Framework.

1 x Using a Console Interface
1 x Using a basic WPF Interface

The logic of the app needs to be coded using the SOLID principles and you are to use the same code for each of the application types that you create. You are only allowed to change the code that is used for the User Interface.

Tasks

For all the tasks you have complete freedom when it comes to the layout within the following restrictions:

- You are make the app look good, look for contrast, spacing and other user elements that make a good UI.
- Any data stored in the app, will disappear when you quit the app, none of the data is persistent.
- You are to do your best to implement the first 4 of the SOLID principles. In the event where you feel this cannot be done, create a little **notes.md** explaining why you couldn't do this.

Task 1 – User Registration Form (32 Marks)

You are to create a registration form for an application. The application needs to ask for the following parameters:

- First Name
- Last Name
- Email
- Username (generated automatically)
  (Format is 1st letter of first name followed by the last name. The username has to be in lowercase)
- Password (with confirmation)

At a later stage your manager asked you to add this field to your application.

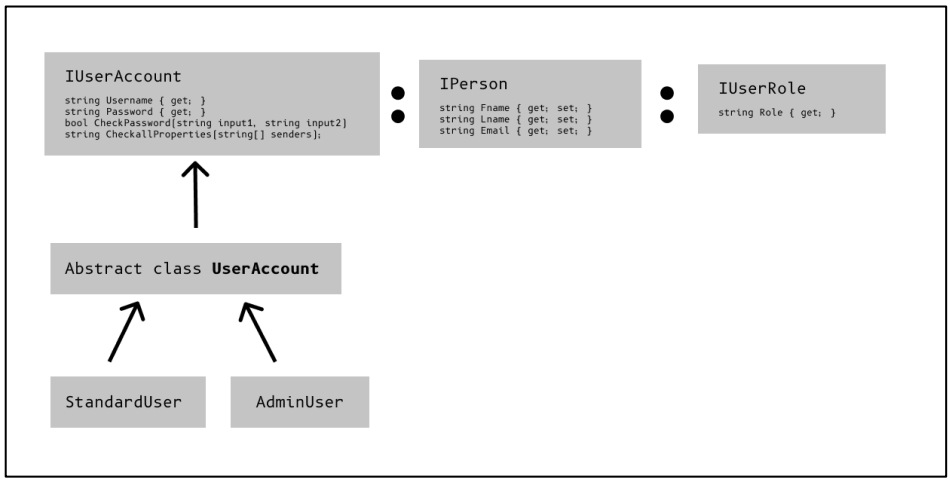
- User's Role
  (Create 2 roles, an admin and a standard user. By default a user is a standard user).

The application needs to validate that all fields have been completed.
The data needs to be stored in a collection and printed to the screen.

There should also be an option to list all the registered users.

EXTRA INFORMATION:

The above task must be modelled after the following UML diagram.



The **StandardUser** and **AdminUser** must inherit from **UserAccount** and only implement the value of the Role property – this can not be typed in by the user.

Task 2 - Phonebook (64 Marks)

You are to create a contact list that displays 5 users. Display the contacts in either a table (for console) or use labels to display the data.

DO NOT USE A DATAGRID
You will need to display the following data:

- First Name
- Last Name
- Phone Number
- Address (street and city)

At a later stage your manager decided to add this field to your application.

- Email

Everyone can see the first name, last name, phone number and email, but only supervisors can see the address (street and city)

You have to use the data from this link:

- https://jsonplaceholder.typicode.com/users?_limit=5

It is your choice to hardcode the information or pull in the API into the app if you want to extend yourself.

EXTRA INFORMATION:

This task will test you on "smarter" programming. You will not be given an UML diagram for this task since it will reveal the answer.

You are allowed to some DRY programming (don't repeat yourself) in the **program.cs**, otherwise you have to create to many interfaces and classes as is done in the Tim Corey OCP Video.

You do not need to submit a UML for this task, but do create one to plan your application well.

Task 3 - Random Number Game (32 Marks)

You are to create a little game where you need to guess the number that the app has selected between 2 values that are pre set per level.

You get 3 chances.

All your guesses need to be recorded in a collection so that you can see your previous guesses.

If you guess the number correctly you score points.

At first there is a single level of difficulty [Easy], it will allow the player to pick any number between 01 and 05:

- Easy – game between 01 and 5

You need to extend the game to in several levels of difficulty

- Med – game between 01 and 10
- Hard – game between 01 and 20

A player can score points for number of guesses, which are as follows

1	10
2	6
3	2

You will need to record the points in a separate collection and display them on the screen. Remember to include at what level they scored the points and give user the option to type in their name with a maximum of 5 letters.

EXTRA INFORMATION:

This task will test you on your decisions to see what really is code for the GUI and what is code for the business logic – you will find that the console app is significantly harder than the WPF app, so plan it carefully and well.