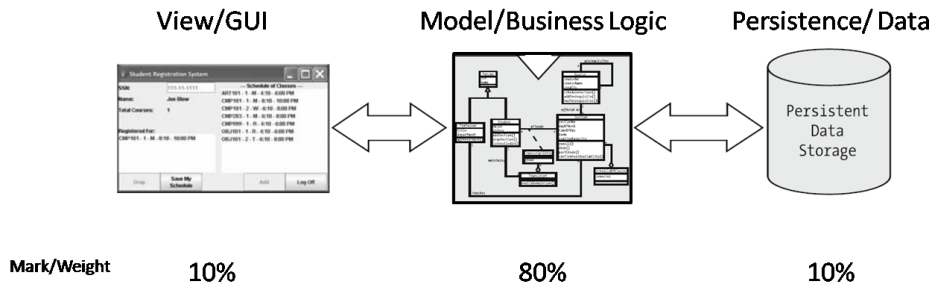
# Guide to Completing COMP6213 OOP The Final Project

First of all, you should think about how the marking of the project will be conducted. The following diagram shows the breakdown of the marking/scoring.




If you focus your effort in developing GUI and even produce the application having impressive look & feel, you have miss a chance to get a high score. Impressive GUI with poor Model will result in a lower score that having a good model with no GUI at all.

The reason for having this marking breakdown is to ensure that you develop proper Object-oriented applications. If you properly designed the model, this model can be re-used in many other application be desktop, web or enterprise without changing the model at all.

That's why the better approach in completing the project should follow these steps:

1. STEP 1: Define the Model

   Model comprises domain classes. Domain classes are classes that exist in the problem, not in the implemented application. All GUI classes are implementation classes. Examples of domain classes include Customer, Book, Product, Transaction, etc. The class diagram represents the Model. Therefore, you should start with identifying the relevant classes, their attributes and methods. Since the problems are very common, you can compare your class diagram with some sources on internet or you can consult with me.

2. STEP 2: Implement the model in Java programming language

   The class diagram does not only show the classes, but it also includes the relationships among the classes. There are two common class relationships: Association and Inheritance. Association relationships (including Composition and Aggregation but don't worry about these two at the moment) actually represents Business Rules, for example "A student can borrow many Book". You will have to code the classes and the relationships properly, otherwise your application may not work as required. The model is actually the also the representation of the Business Rules or Business Logic. It is the core or the engine of your application. Please, make sure that you consult with me about the coding. You can either send the code via email then I can make some comments, or your can make an appointment to do face-to-face consultation.

3. STEP 3: Hard-coded data

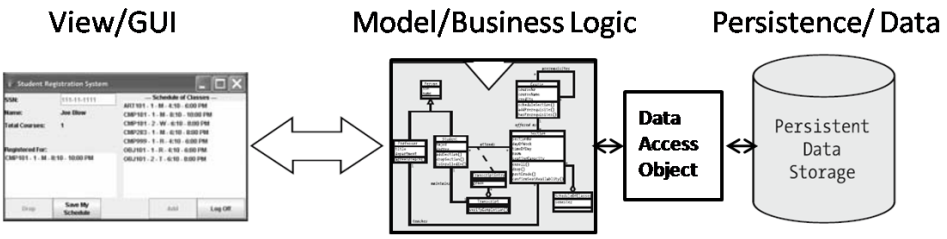
With regard to the Final Project, in the first stage of your design and coding, you can ignore any classes or interface related to Administrator. You should focus on the Users and how they interact with the application.

| Project | User |
|---|---|
| Bookstore | Customer |
| Publisher | Customer (Bookstore) |
| Travel Agent | Customer |
| Soccer Manager | Fan |
| Library | Member (Borrower) |
| CD store | Customer |
| NGO | Member |
| Auction | Bidder |

At this stage, ignore the Administrator and its functions such as Add/Edit/Delete data. How your application gets the data? For the moment, it is better that you simply hard-code in the main function of the Driver program. This is to ensure that your Model works at the first place. It is also useful for testing every single class, and the whole Model. You're always welcomed to consult with me to know exactly what I mean with the hard-coded data.

4. STEP 4: Data Access Object

Having successfully implemented and tested the Model with hard-coded data, it is time now to provide persistence or stored data, in a form of a text file, database, or xml.




DAO is a class that contains statements to read/write data into/from files, database or xml. Please, consult with me to see how this class looks like.

5. STEP 5: Interface or GUI

Creating the GUI after all the Model and DAO classes have been coded, tested and simulated to work properly is far easier than coding it before or during the Model/DAO development. This will ensure that your Mode is separated from the GUI. Having a clear separation between Model and GUI or View will make sure that we can easily change the GUI without touching the Model classes. This also makes sure that the functionality/business logic of your application is embedded in the Model, not in the GUI. If you wrongly put the functionality in the GUI, your program will not be object-oriented. It will look more like C than Java program. You can consult with me to help understanding how to get the GUI working while it is separated from the Model.

6. STEP 5: The Administrator Class (Add/Edit/Delete data)

   Having the program worked, adding Administrator function to Add/Edit/Delete data should be straight forward. Actually, when you use a text editor to add, edit or delete textual data, you play the role as an Administrator. Now, instead of using a text editor, you will create a class that acts similar to it, that is to add, edit and delete data into or from files, database or xml. You may have a different GUI classes (Frame) for this Administrator.

Talking about the scoring again and relate it to these 6 steps, we may trace your accumulated final project score as follows:

| Step | Score |
|---|---|
| STEP 1: Define the Model | 20 |
| STEP 2: Implement the model in Java programming language | 20+40 = 60 |
| STEP 3: Hard-coded data | 20+40+10 = 70 |
| STEP 4: Data Access Object | 20+40+10+10 = 80 |
| STEP 5: Interface or GUI | 20+40+10+10+10 = 90 |
| STEP 5: The Administrator Class (Add/Edit/Delete data) | 20+40+10+10+10+10 = 100 |

good luck!

If you think about all the specification of the final projects, you will recognize that there are similarity across the projects.