Tutorial/Practical Class 6 in Week 7

In this tutorial, you will answer some questions regarding Class Diagram and Design Principles

Relevant Course Materials: Week 6: Lecture and Lectorial (Class Diagram and Design Principles)

Relevant Assignments: Assignment 2

[Individual] Part A: Class Diagram Concepts (25 mins)

1. Answer the following questions
    a. What are objects?

       A discrete entity with a well-defined boundary that encapsulates state and behaviour. An object is an instance of a class

    b. What is the difference between a class and an instance?

       Class is the descriptor for a set of objects that share the same attributes, operations, methods, relationships, and behaviour. So, we can create several instances of a class.

    c. What is encapsulation?

       We “encapsulate” (hide) certain things from other objects. The data inside an object is hidden and can only be manipulated by invoking one of the object's functions.

    d. What is the difference between Aggregation and Composition?

       Aggregation is a weak Whole-Part relationship (like a computer system and its peripherals). The aggregate can sometimes exist independently of the parts, sometimes not; The parts may exist independently of the aggregate.

       Composition is a strong form of aggregation (like a tree and its leaves). The parts belong to exactly one composite at a time. If the composite is destroyed, it must destroy all its parts or give responsibility for them over to some other object.

        i. Provide an example of Aggregation and Composition in the Library System.

           The relation between the Book class and the Chapter class can be a Composition.

           The relation between the BookType class and Book class can be an Aggregation.

    e. What is the difference between cohesion and coupling?

       Coupling measures how closely connected two classes or modules are.

       Cohesion measures the degree to which the elements inside a module/class belong together.

[Team-based] Part B: Class Diagram of the Online Bookstore Platform (80 mins)

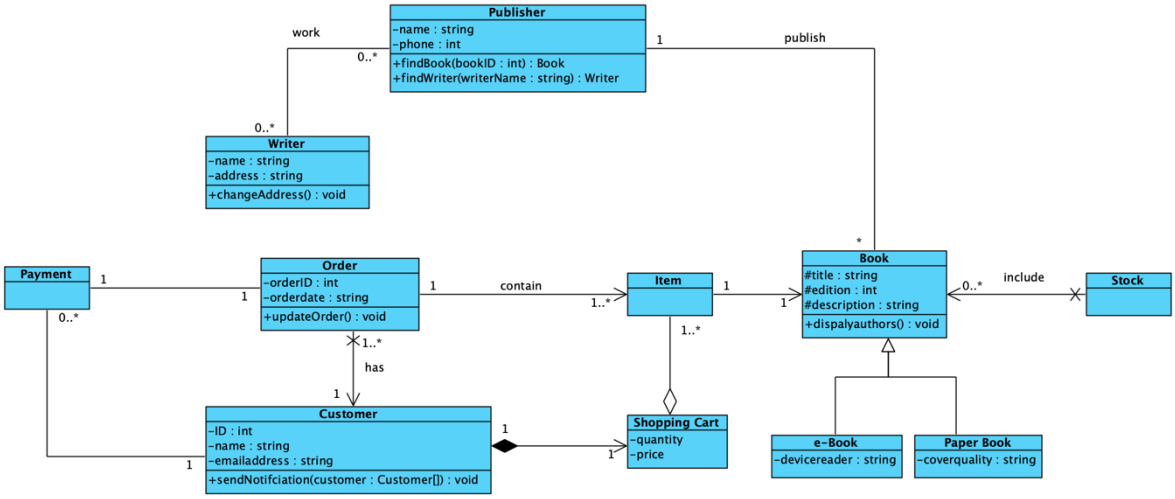
Note: You need to team up with your group members and work on the following tasks.

Imagine you (students) are a software engineer working in a software development organization and want to develop an Online Bookstore Platform. The Online Bookstore Platform stores and sells different types of books. In the following parts, you are expected to work on the design process of the Online Bookstore Platform. You have freedom and have to add more features, services, requirements, etc. to the Online Bookstore Platform.

1. Use the information that you gathered from previous tutorials about the stakeholders, functional/non-functional requirements, and use case modelling of the Online Bookstore Platform and create a Class Diagram for this system using SOLID and GRASP principles (**45 minutes**).

   Note 1: Your class diagram should have at least **10 classes**: all classes should have two attributes and two operations, multiplicity. Your class diagram should include 4 diverse relationships such as association, generalization, aggregation, and composition relationships

   Note 2: Use the Visual Paradigm[1,2] software tool to create the class diagram



2. Write the skeleton code of the designed class diagram in Java (**35 minutes**).

   Note: Please respect the following conventions when implementing the code:

   - Create a basic Java project in Eclipse.
   - Methods should be empty or return a dummy value (empty String “”, 0, or 0.0 depending on its return value).
     - No need to provide a concrete implementation of methods
   - Implement the visibility as presented in the class diagram.
   - If required by multiplicity, use java.util.List (e.g. “private List<User> books;”).

Book Class

```
import java.util.List;
public class Book {
```

[1] https://ap.visual-paradigm.com/rmit-university/
[2] https://www.rmit.edu.au/students/support-services/it-support-systems/software-apps

```
        private Publisher publisher;
        protected String title;
        protected int edition;
        protected String description;
        public void displayAuthors()
        {

        }
}
```

eBook Class

```
import java.util.List;
public class eBook extends Book{
        private String devicereader;
}
```

PaperBook Class

```
import java.util.List;
public class PaperBook extends Book {
        private String coverquality;

}
```

Customer Class

```
import java.util.List;
public class Customer {
        private ShoppingCart shoppingCart;
   private List <Payment> payments;
        private int ID;
        private String name;
        private String emailladdress;
        public void sendNotifcation(Customer []customer)
        {

        }
}
```

Item Class

```
public class Item {

}
```

Order Class

```
import java.util.List;
```

```
public class Order {
	private List <Item> items;
	private Payment payment;
	private Customer customer;
	private int orderID;
	private String orderdate;
	public void updateOrder()
	{

	}
}
```

Publisher Class

```
import java.util.List;
public class Publisher {
	private List <Writer> writers;
	private List <Book> books;
	private String name;
	private int phone;
	public Book findBook(int bookId)
	{
		Book book= new Book();
		return book;
	}
	public Writer findWriter(String writerName)
	{
		Writer writer= new Writer();
		return writer;

	}
}
```

ShoppingCart Class

```
import java.util.List;
public class ShoppingCart {
	private List <Item> items;
}
```

Stock Class

```
import java.util.List;

public class Stock {
	private List <Book> books;
}
```

Writer Class

```
import java.util.List;
public class Writer {
	private List <Publisher> publishers;
	private String name;
	private String address;
	public void changeAddress()
	{

	}
}
```