Terna Engineering College

Computer Engineering Department

Program: Sem VI

Course: Software Engineering Lab

LAB Manual

PART A

(PART A: TO BE REFERRED BY STUDENTS)

Experiment No.05

A.1 Aim:

Develop UML Class Diagram for selected mini project

A.2 Prerequisite:

Requirement Modelling

A.3 Outcome:

After successful completion of this experiment, students will be able to

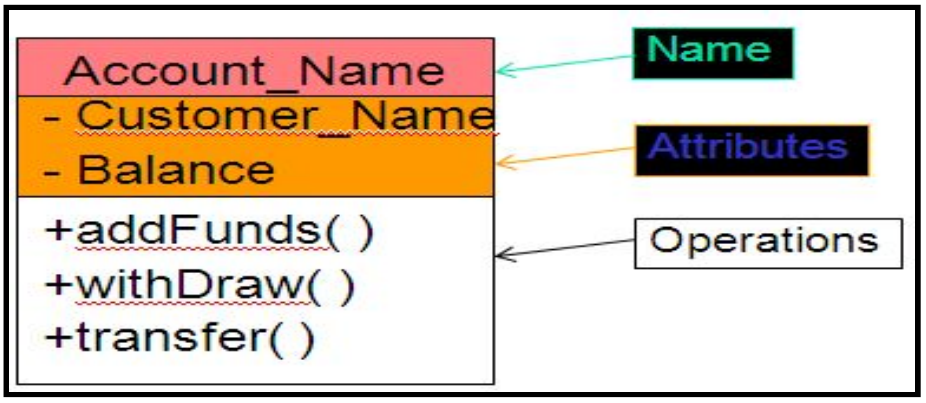
- ✔ Refine requirements from a set of preliminary requirements and able to identify actors and possible use cases.
- ✔ Able to model requirements using UML

A.4 Theory:

- The class diagram is a collection of static elements such as classes and their relationships connected like a graph to each other.
- It shows the dependency between the classes that can be used in our system.
- Used for describing structure and behaviour in the use cases
- Provide a conceptual model of the system in terms of entities and their relationships
- Used for requirement capture, end-user interaction
- Detailed class diagrams are used for developers
- The Class Diagram shows a set of classes, interfaces, and collaborations and their relationships.

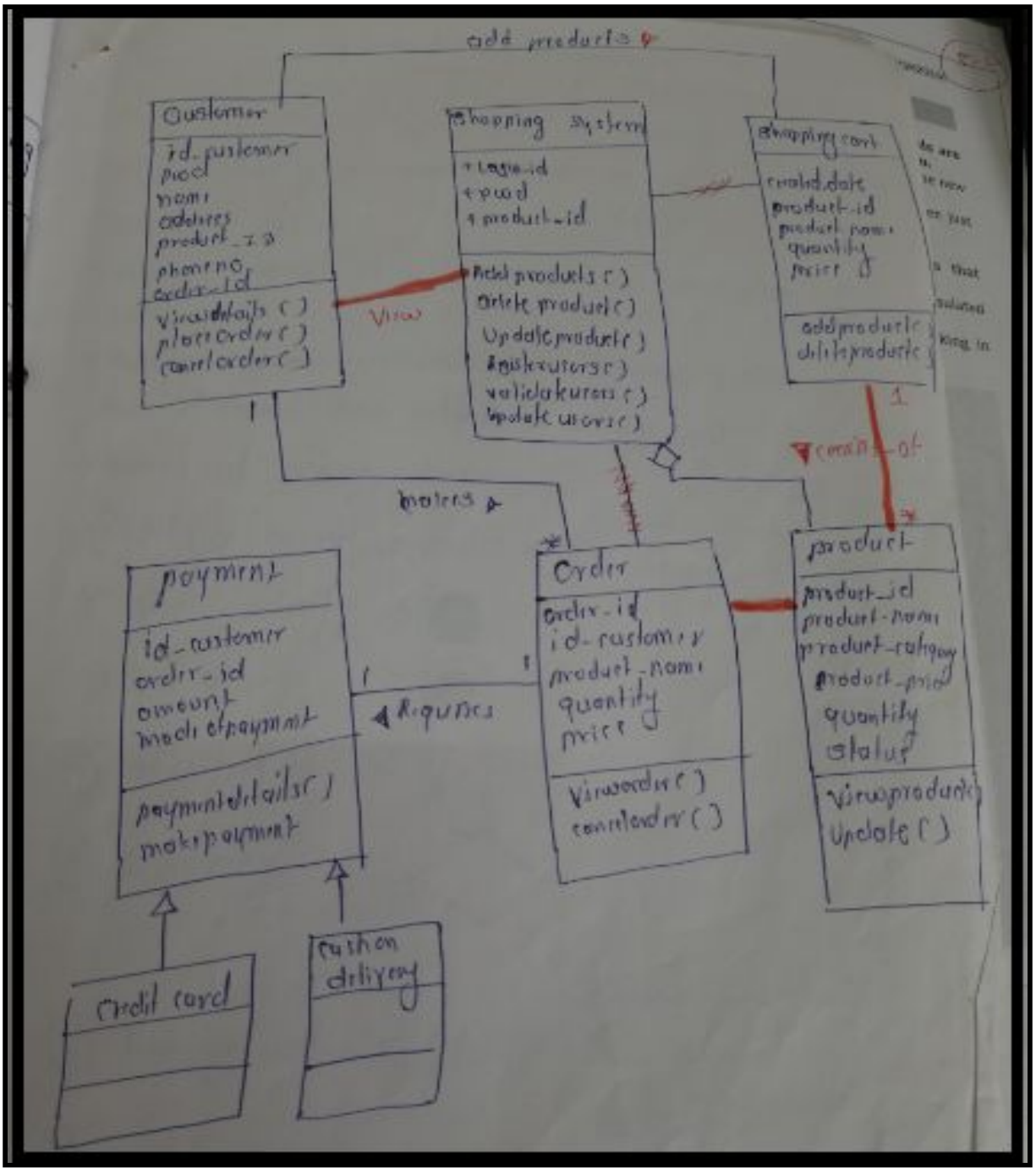
- There is the most common diagram in modelling object-oriented systems and are used to give the static view of a system.
- UML Class Diagram consists of;
    - Classes
    - Relationships

- **Class:**
    - **Each class is represented by a rectangle subdivided into three compartments**
        - Name
        - Attributes
        - Operations
    - Modifiers are used to indicate the visibility of attributes and operations.
    - '+' is used to denote *Public* visibility (everyone)
    - '#' is used to denote *Protected* visibility (friends and derived)
    - '-' is used to denote *Private* visibility (no one)



- **Relationships:**
    - There are three kinds of Relationships
        - **Generalization** (parent-child relationship)
        - **Association** (student enrols in course)
        - **Dependency** ---------

■ **Example:** **Class Diagram for Online Shopping System**



PART B

(PART B: TO BE COMPLETED BY STUDENTS)

(Students must submit the soft copy as per the following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case there is no Blackboard access available)

Roll No. 50	**Name:** AMEY THAKUR
Class: Comps TE B	**Batch:** B3
Date of Experiment: 03/03/2021	**Date of Submission:** 03/03/2021
Grade:	

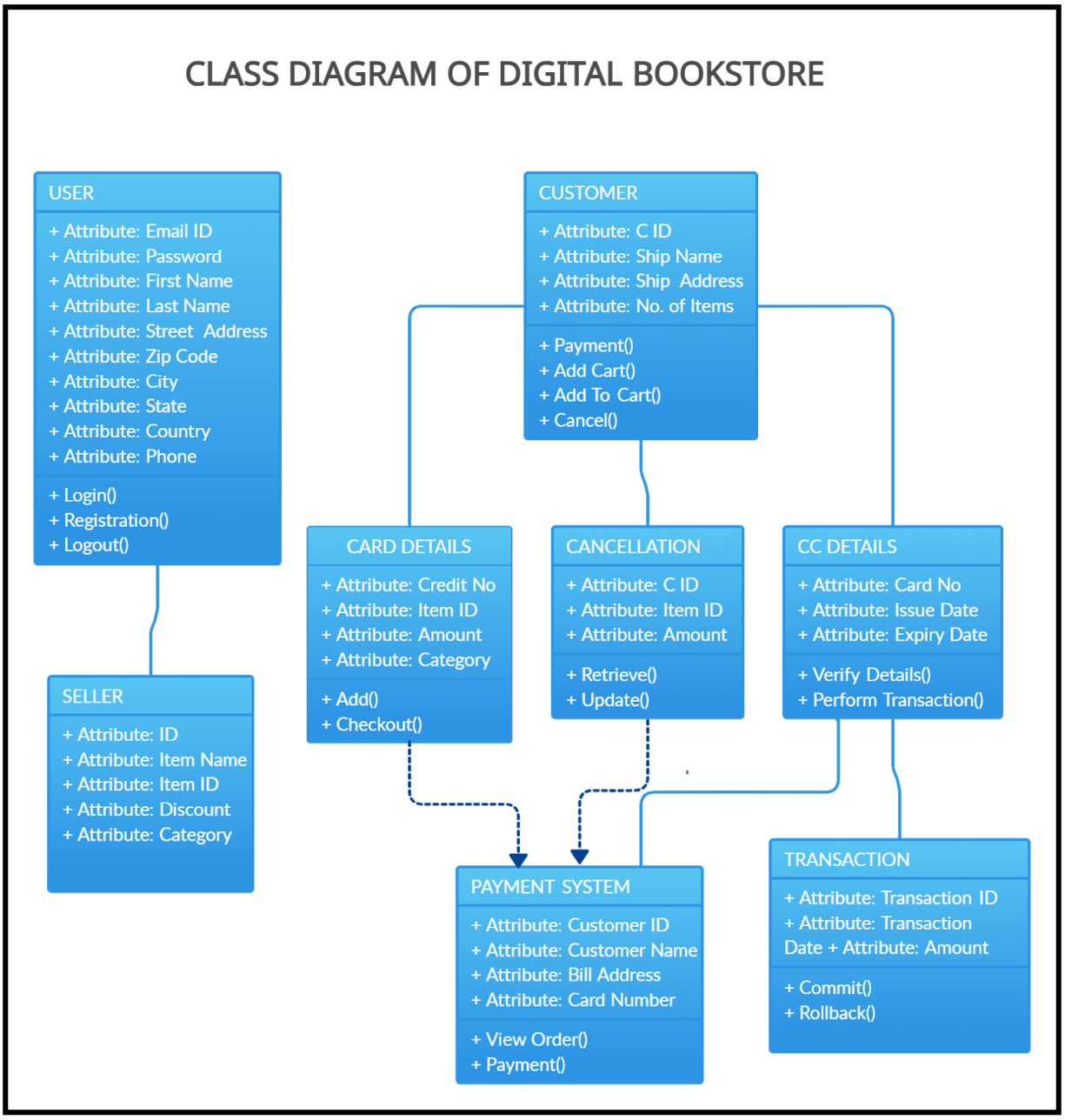
B.1 Draw Class Diagram for selected mini project

What is a Class diagram?

- ➔ The class diagram is static. It represents the static view of an application. The class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.
- ➔ The class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modelling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.
- ➔ The class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

Purpose of Class Diagrams

- ➔ The purpose of the class diagram is to model the static view of an application. Class diagrams are the only diagrams that can be directly mapped with object-oriented languages and thus widely used at the time of construction.
- ➔ UML diagrams like activity diagram, sequence diagrams can only give the sequence flow of the application, however, the class diagram is a bit different. It is the most popular UML diagram in the coder community.
- ➔ The purpose of the class diagram can be summarized as –
    - Analysis and design of the static view of an application.
    - Describe the responsibilities of a system.
    - The base for component and deployment diagrams.
    - Forward and reverse engineering.



B.2 Conclusion:

(Students must write the conclusion)

UML class diagrams are useful when modelling business data. By accurately modelling attributes and associations of class entities, we can easily map these class diagram specifications to entity beans with CMP. Class attributes map to abstract access methods for persistent fields, and association roles map to abstract access methods for relationship fields. Navigability determines whether relationship access methods appear in both related entity beans or just one. Furthermore, multiplicity notation determines the correct type for relationship fields, life cycle issues, and cascading delete characteristics.

B.3 Question of Curiosity

1. Explain Composition and aggregation with suitable examples.

Ans:

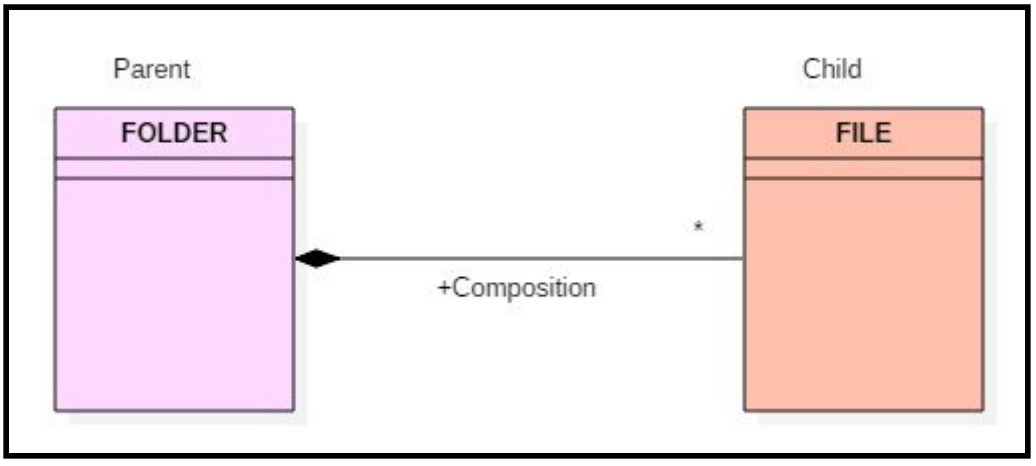
UML Composition

- ➔ It is not a standard UML Relationship, but it is still used in various applications.
- ➔ Composite aggregation is a subtype of aggregation relation with characteristics as:
  - It is a two-way association between the objects.
  - It is a whole/part relationship.
  - If a composite is deleted, all other parts associated with it are deleted.
- ➔ Composite aggregation is described as a binary association decorated with a filled black diamond at the aggregate (whole) end.

- ➔ The folder could contain many files, while each File has exactly one Folder parent. If a folder is deleted, all contained files are removed as well.
- ➔ In a composite aggregation, an object may be a part of only one composite at a time.

UML Composition Example:

- ➔ For example, in a windowing system, a Frame belongs to precisely one Window. In a composite aggregation, the whole system is responsible for the disposition of its parts, which means that the composite must manage the creation and destruction of its parts.



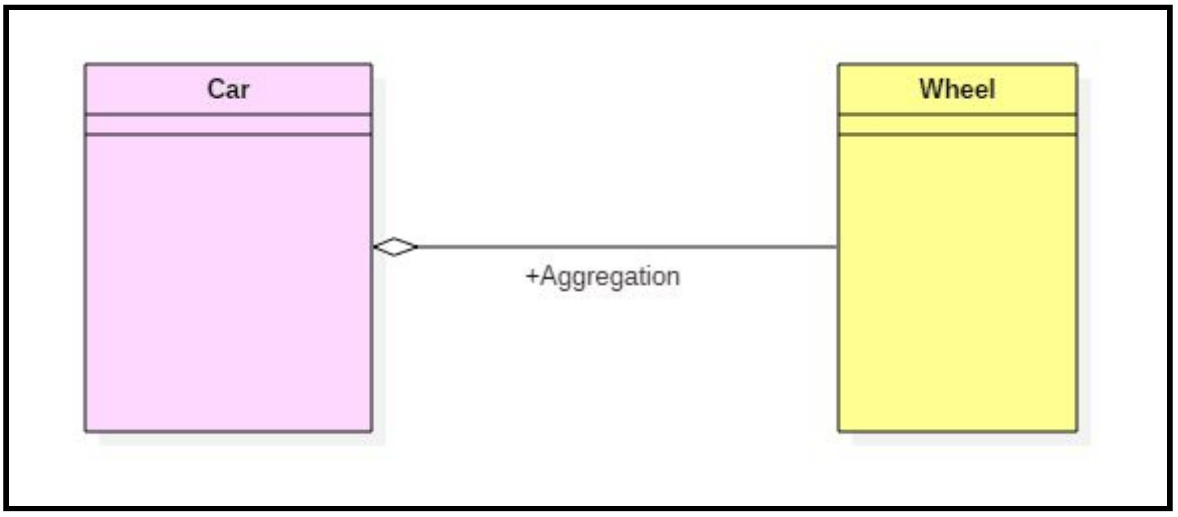
Composition

UML Aggregation

- ➔ An aggregation is a subtype of an association relationship in UML. Aggregation and composition are both the types of association relationship in UML. An aggregation relationship can be described in simple words as "an object of one class can own or access the objects of another class."
- ➔ In an aggregation relationship, the dependent object remains in the scope of a relationship even when the source object is destroyed.

UML Aggregation Example:

- ➔ Let us consider an example of a car and a wheel.
- ➔ A car needs a wheel to function correctly, but a wheel doesn't always need a car. It can also be used with the bike, bicycle, or any other vehicles but not a particular car. Here, the wheel object is meaningful even without the car object. Such a type of relationship is called the UML Aggregation relation.



Aggregation

2. Explain various elements used in association relationships.

Ans:

Association

It is a structural relationship that represents objects that can be connected or associated with another object inside the system. Following constraints can be applied to the association relationship.

- ➔ {implicit} – Implicit constraints specify that the relationship is not manifest; it is based upon a concept.
- ➔ {ordered} – Ordered constraints specify that the set of objects at one end of an association are in a specific way.

- ➔ {changeable} – Changeable constraint specifies that the connection between various objects in the system can be added, removed, and modified as per the requirement.
- ➔ {addOnly} – It specifies that the new connections can be added from an object which is situated at the other end of an association.
- ➔ {frozen} – It specifies that when a link is added between two objects, then it cannot be modified while the frozen constraint is active on the given link or a connection.

We can also create a class that has association properties; it is called an association class.

****