Software Engineering
Experiment 4: Implement Class Diagram

Learning Objective: To implement UML class diagrams for the project.

Tools: MS Word, draw.io or any other UML tool.

Theory:

Class Diagrams:

Classes are the structural units in object oriented system design approach, so it is essential to know all the relationships that exist between the classes, in a system. All objects in a system are also interacting to each other by means of passing messages from one object to another.

Elements in class diagram

Class diagram contains the system classes with its data members, operations and relationships between classes.

Class

A set of objects containing similar data members and member functions is described by a class. In UML syntax, class is identified by solid outline rectangle with three compartments which contain

● **Class name**

A class is uniquely identified in a system by its name. It lies in the first compartment in the class rectangle.

● **Attributes**

Property shared by all instances of a class. It lies in the second compartment in class rectangle.

● **Operations**

An execution of an action can be performed for any object of a class. It lies in the last compartment in the class rectangle.

Example

To build a structural model for an Educational Organization, 'Course' can be treated as a class which contains attributes 'courseName' & 'courseID' with the operations 'addCourse()' & 'removeCourse()' allowed to be performed for any object to that class.
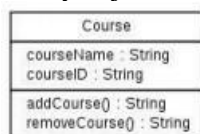
● **Generalization/Specialization**

It describes how one class is derived from another class. Derived class inherits the properties of its parent class.
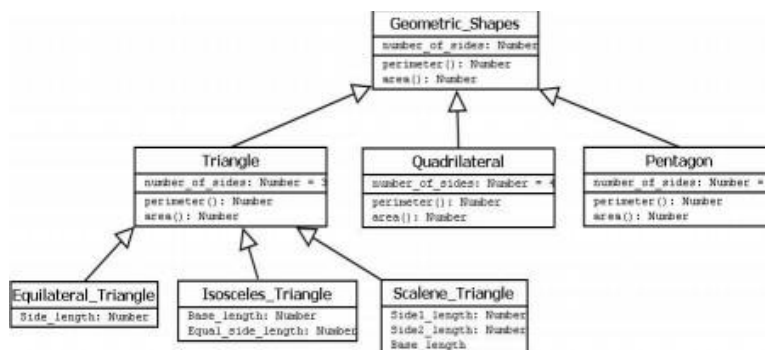
Example

Figure-02:

Geometric_Shapes is the class that describes how many sides a particular shape has. Triangle, Quadrilateral and Pentagon are the classes that inherit the property of the Geometric_Shapes class. So the relations among these classes are generalization. Now Equilateral_Triangle, Isosceles_Triangle and Scalene_Triangle, all these three classes inherit the properties of Triangle class as each one of them has three sides. So, these are specialization of Triangle class.

Relationships

Existing relationships in a system describe legitimate connections between the classes in that system.

Example

In structure model for a system of an organization an employee (instance of 'Employee' class) is always assigned to a particular department (instance of 'Department' class) and the association can be shown by a line connecting the respective classes.
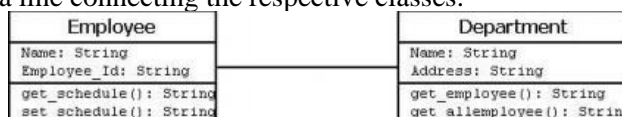
Figure-03:

- **Aggregation**

It is a special form of association which describes a part-whole relationship between a pair of classes. It means, in a relationship, when a class holds some instances of related class, then that relationship can be designed as an aggregation.

Example

For a supermarket in a city, each branch runs some of the departments they have. So, the relation among the classes 'Branch' and 'Department' can be designed as aggregation. In UML, it can be shown as in the fig. below.
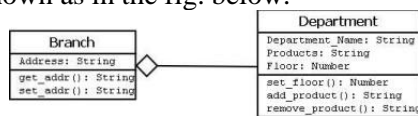
Figure-04:

- Composition

It is a strong from of aggregation which describes that whole is completely owns its part. Life cycle of the part depends on the whole.

Example

Let consider a shopping mall has several branches in different locations in a city. The existence of branches completely depends on the shopping mall as if it is not exist any branch of it will no longer exists in the city. This relation can be described as composition and can be shown as below

Figure-05:

- **Multiplicity**

It describes how many numbers of instances of one class is related to the number of instances of another class in an association.

Notation for different types of multiplicity:

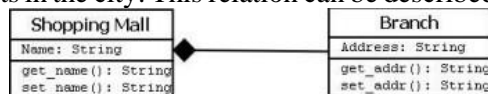|	
Single instance	1
Zero or one instance	0..1
Zero or more instance	0..*
One or more instance	1..*
Particular range(two to six)	2..6

Figure-06:

Example
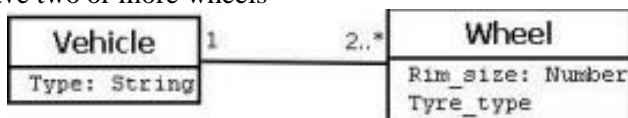
One vehicle may have two or more wheels

Figure-07:

Procedure:

When required to describe the static view of a system or its functionalities, we would be required to draw a class diagram. Here are the steps you need to follow to create a class diagram.

Step 1: Identify the class names.

The first step is to identify the primary objects of the system.

Step 2: Distinguish relationships.

Next step is to determine how each of the classes or objects are related to one another. Look out for commonalities and abstractions among them; this will help you when grouping them when drawing the class diagram.

Step 3: Create the Structure

First, add the class names and link them with the appropriate connectors. You can add attributes and functions/ methods/ operations later.

Result and Discussion:

Q.1) What is a Class diagram?

Ans Classes are the structural units in object-oriented system design approach, so it is essential to know all the relationships that exist between the classes, in a system. All objects in a system are also interacting to each other by means of passing messages from one object to another.

Q.2) Draw the class diagram for your project.

Role class

+role_id :int

+role_title : string

+role _description:string

+add Role()

+edit role()

+delete role()

+search role()

+assign role ()

Permission class

+permission_id:int

+permission_role_id:int

+permission_module:string

+permission_title:string
+permission_description:string

+addPermission()

+editPermission()

+deletePermission()

+searchPermission()

Payment class

+payment_id:int

+payment_customer_id:int

+payment_description :string
+payment_amount:string
+payment_date:date

+addPayment()

+editPayment()

+deletePayment()

+searchPayment()

---

user class

+user id :int

+user_role_id:int
+user_name:string
+user_email:string
+user_dob:date
+user_address:string

+addUser()

+editUser()

+deleteUser()

+searchUser()

Customer Class

+customer_id:int

+customer_name:string

+customer_mobile:string

+customer_email:string
+customer_password:string
+customer_username:string
+customer_address:string

+addCustomer()

+editCustomer()

+deleteCustomer()

+searchCustomer()

Inventory Class

+inventory_id:int

+inventory_description:string

+inventory_number :string

+inventory_type:string

+inventory_items:string

+addInventory()

+editInventory()

+deleteInventory()

+searchInventory()

Stock Class

+stock_id:int

+permission_role_id:int

+permission_module:string

+permission_title:string
+permission_description:string

+addStock()

+editStock()

+deleteStock()

+searchStock()

Supplier class

+Supplier_id:int

+supplier_name:string

+supplier_email:string

+supplier_address:string
+supplier_password:string
+supplier_username:string

+addSupplier()

+editSupplier()

+deletesupplier()

+SearchSupplier()

Learning Outcomes: The student should have the ability to:

LO 1: Identify the importance of class diagrams.

LO 2: Draw class diagrams for a given scenario.

Course Outcomes: Upon completion of the course students will be able to understand and demonstrate class diagrams.

Conclusion: Through this experiment, we were able to learn about the UML class diagram and its implementation for the Footwear Inventory Management System. The importance of class diagrams was also discussed, and all the scenarios were taken into consideration while constructing the class diagram.

For Faculty Use

Correction Parameters	Formative Assessment [40%]	Timely completion of Practical [40%]	Attendance / Learning Attitude [20%]	
Marks Obtained				