**Assignment 4**

**Class Diagram**

**AIM -** Design and implementation of analysis model - Class Model.

**PROBLEM STATEMENT -**

Identify Analysis Classes and assign responsibilities.

Implement Analysis class Model-class diagram with a

suitable object-oriented language.

**OBJECTIVE -**

To Identify Analysis Classes and assign responsibilities.

To Draw Analysis class Model.

To Implement Analysis class Model-class diagram.

**THEORY -**

**Class Diagram :**

The Class diagram shows the building blocks of any object-orientated system. Class diagrams depict the static view of the model or part of the model, describing what attributes and behaviors it has rather than detailing the methods for achieving operations. Class diagrams are most useful to illustrate relationships between classes and interfaces. Generalizations, aggregations, and associations are all valuable in reflecting inheritance, composition or usage, and connections, respectively.

**Classes :**

A class is an element that defines the attributes and behaviors that an object is able to generate. The behavior is described by the possible messages the class is able to understand along with operations that are appropriate for each message. Classes may also contain definitions of constraints tagged values and stereotypes.

**Interfaces :**

An interface is a specification of behavior that implementers agree to meet. It is a contract. By realizing an interface, classes are guaranteed to support a required behavior, which allows the system to treat non-related elements in the same way – i.e. through the common interface.

**Association :**

An association implies two model elements have a relationship - usually implemented as

an instance variable in one class. This connector may include named roles at each end, multiplicity or cardinality, direction and constraints.

Association is the general relationship type between elements. For more than two elements, When code is generated for class diagrams, associations become instance variables in the target class.

**Generalizations :**

A generalization is used to indicate inheritance.

Drawn from the specific classifier to a general classifier, the generalized implication is that the source inherits the target's characteristics.

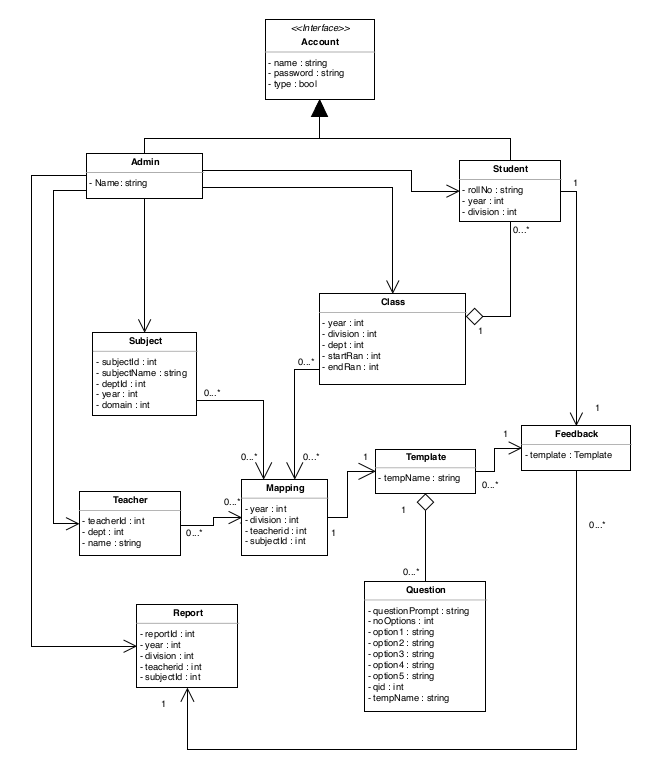
**Aggregations :**

Aggregations are used to depict elements that are made up of smaller components.

Aggregation relationships are shown by a white diamond-shaped arrowhead pointing

towards the target or parent class.

A stronger form of aggregation - a composite aggregation - is shown by a black diamond-shaped arrowhead and is used where components can be included in a maximum of one composition at a time. If the parent of a composite aggregation is deleted, usually all of its parts are deleted with it however, a part can be individually removed from a composition without having to delete the entire composition. Compositions are transitive, asymmetric relationships and can be recursive.

****

There are 9 classes in the aforementioned Class diagram

For the Feedback management system,

Classes included being - Account, Admin, Class, Subject, Student, Teacher, Question, Template, and Feedback.

**Abstraction relationships :**

An abstraction relationship is a dependency between model elements that represent the same concept at different levels of abstraction or from different viewpoints. The account class is further inherited into Admin and Student classes.

**Aggregation relationships :**

In UML models, an aggregation relationship shows a class as a part of or subordinate to

another class. An aggregation is a special type of association in which objects are assembled or configured together to create a more complex object. Aggregation relationship is established between classes Question and Template.

In the Feedback management system, we are using objects of all the classes. The whole system is controlled and objects are created by Admin. For example, the Admin object login should be true. Then for all the subsequent operations performed by the Admin, like AddStudent, AddSubject, Addteacher, etc. we need to give the call to all the operations.

**CONCLUSION -**

Thus we have prepared a Design Model from Analysis Model and studied in detail the working of the system/project. Identified Design classes/ Evolve Analysis Model and used advanced relationships. We drew a Design class Model using OCL and UML2.0 Notations and implemented the design model with a suitable object-oriented language.