**Experiment 03- Implement UML Use-case**

**Learning Objective:** To implement UML use-case for the project.

**Tools:**  MS Word, draw.io

**Theory:**

**Use case diagrams**

**Use case diagrams belong to the category of behavioral diagram** of UML diagrams. They present a graphical overview of the functionality provided by the system. It consists of a set of actions (referred to as use cases) that the concerned system can perform one or more actors, and dependencies among them.

**Actor**

An actor can be defined as an object or set of objects, external to the system, which interacts with the system to get some meaningful work done. Actors could be human, devices, or even other systems.

For example, consider the case where a customer *withdraws cash* from an ATM. Here, customer is a human actor.

Actors can be classified as below:

* **Primary actor**: They are principal users of the system, who fulfill their goal by availing some service from the system. For example, a customer uses an ATM to withdraw cash when he needs it. A customer is the primary actor here.
* **Supporting actor**: They render some kind of service to the system. "Bank representatives", who replenishes the stock of cash, is such an example. It may be noted that replenishing stock of cash in an ATM is not the prime functionality of an ATM.

In a use case diagram primary actors are usually drawn on the top left side of the diagram.

**Use Case**

A use case is simply a functionality provided by a system.

Continuing with the example of the ATM, *withdraw cash* is a functionality that the ATM provides. Therefore, this is a use case. Other possible use cases include, *check balance*, *change PIN*, and so on.

Use cases include both successful and unsuccessful scenarios of user interactions with the system. For example, authentication of a customer by the ATM would fail if he enters wrong PIN. In such case, an error message is displayed on the screen of the ATM.

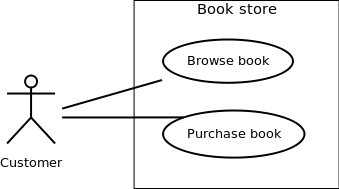
**Subject**

Subject is simply the system under consideration.

**Graphical Representation**

An actor is represented by a stick figure and name of the actor is written below it. A use case is depicted by an ellipse and name of the use case is written inside it. The subject is shown by drawing a rectangle. Label for the system could be put inside it. Use cases are drawn inside the rectangle, and actors are drawn outside the rectangle, as shown in figure - 01.

Figure - 01: A use case diagram for a book store



**Association between Actors and Use Cases**

A use case is triggered by an actor. Actors and use cases are connected through binary associations indicating that the two communicates through message passing.

An actor must be associated with at least one use case. Similarly, a given use case must be associated with at least one actor. Association among the actors is usually not shown. However, one can depict the class hierarchy among actors.

**Use Case Relationships**

Three types of relationships exist among use cases:

* Include relationship
* Extend relationship
* Use case generalization

**Include Relationship**

Include relationships are used to depict common behavior that are shared by multiple use cases. This could be considered analogous to writing functions in a program in order to avoid repetition of writing the same code. Such a function would be called from different points within the program.

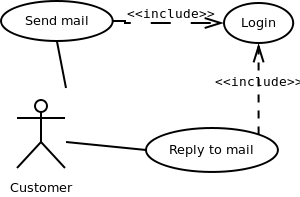


Figure - 02: Include relationship between use cases

**Notation**

Include relationship is depicted by a dashed arrow with a «include» stereotype from the including use case to the included use case.

**Extend Relationship**

Use case extensions are used to depict any variation to an existing use case. They are used to specify the changes required when any assumption made by the existing use case becomes false.

**Example**

Let's consider an online bookstore. The system allows an authenticated user to buy selected book(s). While the order is being placed, the system also allows specifying any special shipping instructions, for example, call the customer before delivery. This *Shipping Instructions* step is optional, and not a part of the main *Place Order* use case. Figure - 03 depicts such relationship.

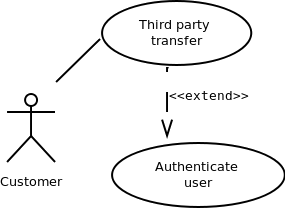


Figure - 03: Extend relationship between use cases

**Notation**

Extend relationship is depicted by a dashed arrow with a «extend» stereotype from the extending use case to the extended use case.

**Generalization Relationship**

Generalization relationship is used to represent the inheritance between use cases. A derived use case specializes some functionality it has already inherited from the base use case.

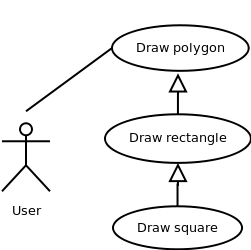


Figure - 04: Generalization relationship among use cases

**Identifying Use cases**

Once the primary and secondary actors have been identified, we have to find out their goals i.e. what the functionality they can obtain from the system is. Any use case name should start with a verb like, "Check balance".

Guidelines for drawing Use Case diagrams

Following general guidelines could be kept in mind while trying to draw a use case diagram:

* Determine the system boundary
* Ensure that individual actors have well-defined purpose
* Use cases identified should let some meaningful work done by the actors
* Associate the actors and use cases -- there shouldn't be any actor or use case floating without any connection
* Use include relationship to encapsulate common behavior among use cases, if any.

**Output :**

Use case diagram for music recommender system



**Result and Discussion:**

Q.1) What is a use-case diagram?

A.1)A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has and will often be accompanied by other types of diagrams as well.

Q.2) What are the components in a use-case diagram?

* Actors: The users that interact with a system. An actor can be a person, an organization, or an outside system that interacts with your application or system.
* System: A specific sequence of actions and interactions between actors and the system.
* Goals: The end result of most use cases.

**Learning Outcomes:** The student should have the ability to:

LO 1: Identify the importance of use-case diagrams.

LO 2: Draw use-case diagrams for a given scenario.

**Course Outcomes:** Upon completion of the course students will be able to understand and demonstrate use-case diagrams.

**Conclusion:**

In this experiment, we studied about components of use-case diagram and we implemented the same for our topic.

**Viva Questions:**

1. What is use-case diagrams used for?

2. What is extends and include? Differentiate between the two.

3. What is a generalization relationship in a use-case?

4. “Use case diagrams belong to the category of behavioral diagram”. What does the statement mean?

For Faculty Use:

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