NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

SL-2002 – Software Design & Architecture Lab

LAB Instructor: Rubab Manzar and
Noureen Fatima

Lab 02

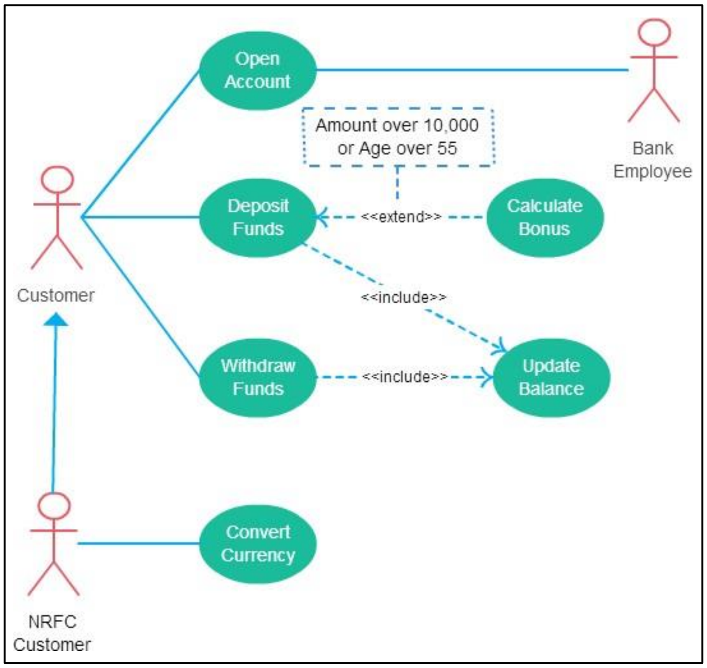
Outline

- Getting to know the Use Case diagram
- Introducing the Use Case Diagram
- Where to find the Use Case Diagram
- Elements of Use Case Diagram
- Partial Use Case Context Diagram
- Extends
- Generalization
- Exercise

Getting to know the Use Case diagram

The Use Case diagram is one of the Unified Modeling Language (UML) Behavioral diagrams that can be used to describe the goals of the users and other systems that interact with the system that is being modeled.

They are used to describe the functional requirements of a system, subsystem or entity.



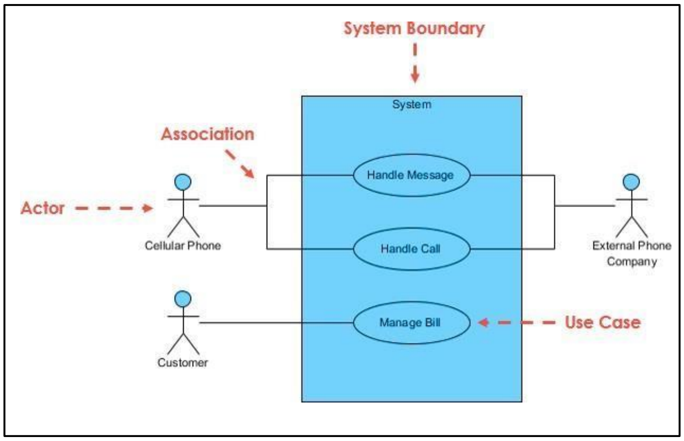
Usage of the Use Case Diagram

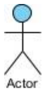
The Use Case diagram is used to describe the goals that users or other systems want to achieve from interacting with the system.

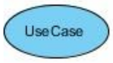
Where to find the Use Case Diagram

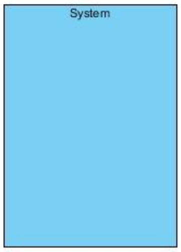
- Ribbon: File > Project> papyrus project > UML Behavioral > Use Case

A standard form of use case diagram is defined in the Unified Modeling Language as shown in the Use Case Diagram example below:







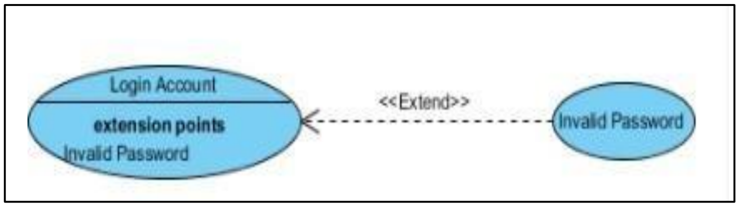


Boundary of system

Communication Link

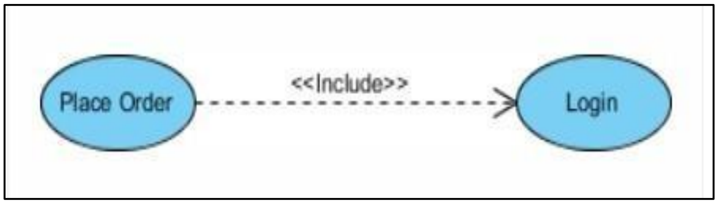
Extends

Indicates that an **"Invalid Password"** use case may include (subject to specified in the extension) the behavior specified by base use case **"Login Account"**.



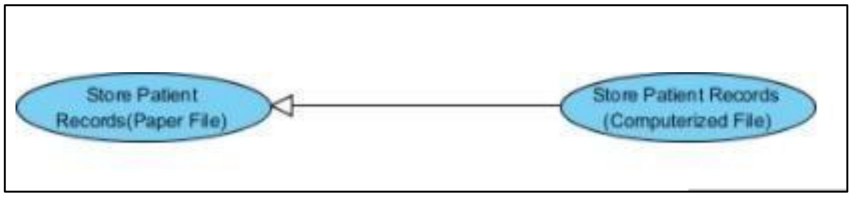
Include

- A uses relationship from base use case to child use case indicates that an instance of the base use case will include the behavior as specified in the child use case.



Generalization

A generalization relationship is a parent-child relationship between use cases.

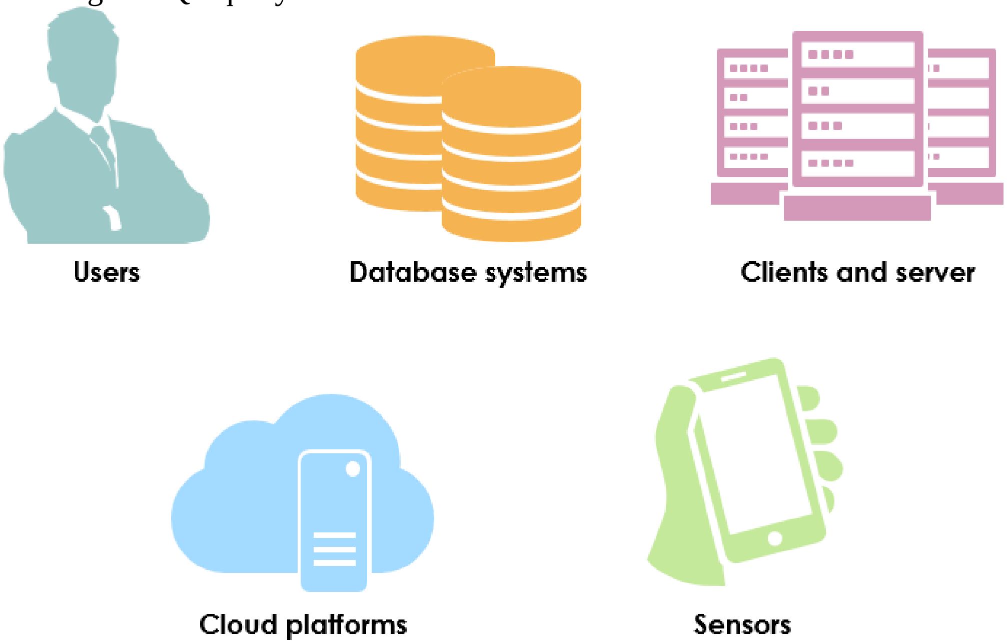

Store Patient Records(Paper File)
Store Patient Records (Computerized File)

Use Case Description example:

A user clicks the search button on an application's user interface. The application sends an SQL query to a database system. The database system responds with a result set. The application formats and displays the result set to the user.

In this scenario:

- The user is a primary actor because he initiates the interaction with the system (application).
- The database system is a secondary actor because the application initiates the interaction by sending an SQL query.



Use Case Examples

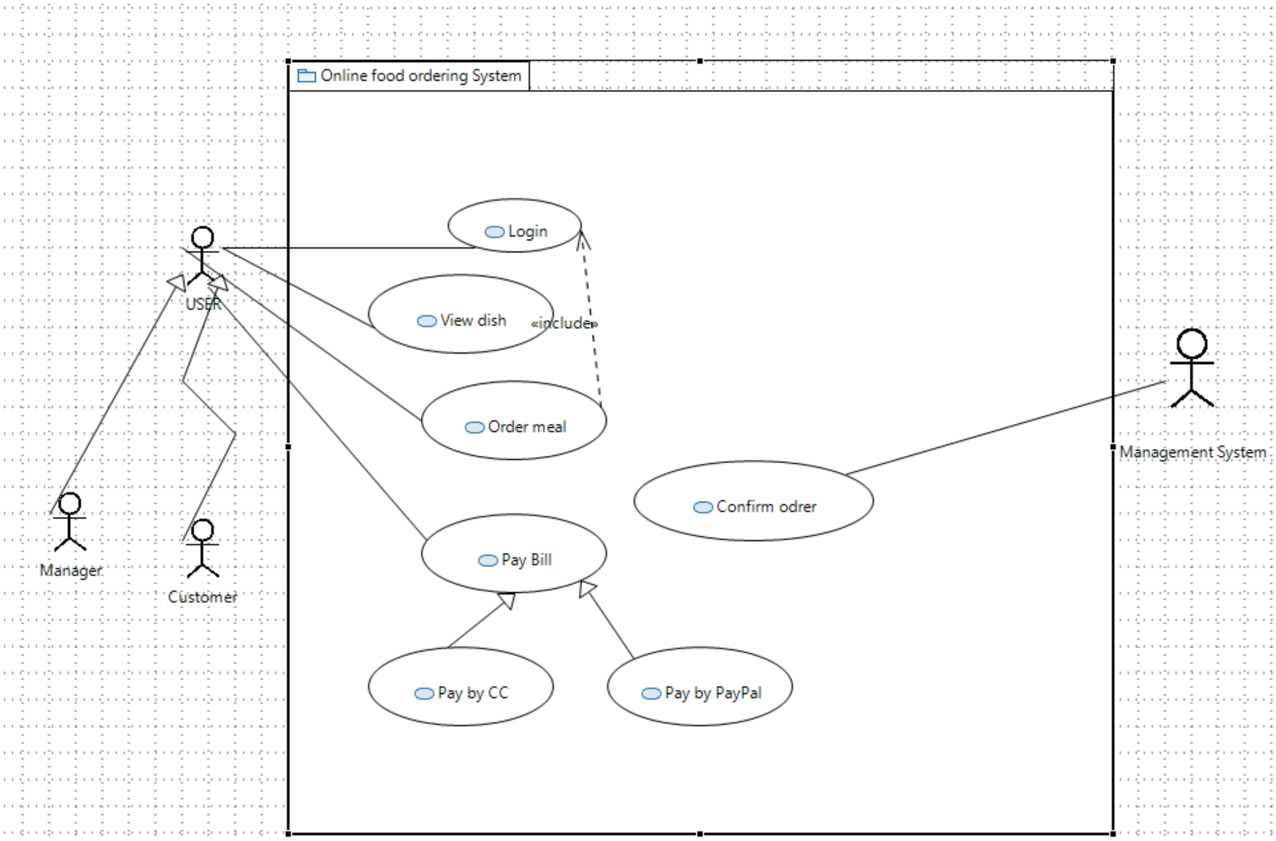
Scenario

Online Ordering System is a modern platform that enables you to manage food delivery business activity efficiently. Restaurant takeout and delivery are taking a bite out of dining traffic, making it easier for customers to search for a favorite restaurant, filter cuisines, choose from listed items, and choose doorstep delivery or pick-up. Online ordering systems generally consists of 2 main components. First is a website or mobile app for hungry customers to view the restaurant's dishes and place an online order. Second is an admin management interface for the restaurants to receive and manage the customer's orders. A customer is able to order meal from Online ordering systems. This online food can be used by customer or manager as user that behave as a particular Actor. There are several key steps, for ordering the favorite meal. First they need to login their account with web application using their email id and they see the multiple options in different category and from the listed category, they can choose food item as many as they want. There are very important restrictions, user cannot order the meal without logged in, that means logged in activity must be included with associated actor. System will take a proper confirmation regarding address and payment details. In this Food Online System there are two modes of payment, one is by credit card and other is via PayPal, user can pay in either way. In this system user cannot proceed their order without selecting particular payment method.

Basic steps for making use case on Eclipse-Papyrus.

Go to File other and create Papyrus Project with selective use case diagram.

1. Select “Package” from node palate.
2. Drag three different Actor
3. Drag Four different use cases
4. Create Relation between use cases and Actor.
5. Identify Include and Extends relation nature.


Online food ordering System
Login
USER
View dish
«include»
Order meal
Management System
Confirm odrer
Pay Bill
Manager
Customer
Pay by CC
Pay by PayPal