UML

# Objective:

To create the following UML:

* sequence diagram

for a real project or system.

# Theory:

**UML (UNIFIED MODELING LANGUAGE):**

**UML** (Unified Modeling Language) is a modeling language used by software developers. UML can be used to develop diagrams and provide users with ready-to-use, expressive modeling examples. Some UML tools generate program language code from UML. UML can be used for modeling a system independent of a platform language. UML is a graphical language for visualizing, specifying, constructing, and documenting information about software-intensive systems. UML gives a standard way to write a system model, covering conceptual ideas. With an understanding of modeling, the use and application of UML can make the software development process more efficient.

There are two categories of UML:

* behavior diagrams
* use case diagram
* sequence diagram
* structured diagrams
* class diagram
* **CLASS DIAGRAM:**

A **class diagram** is an illustration of the relationships and source code dependencies among **classes** in the Unified Modeling Language (UML). In this context, a **class** defines the methods and variables in an object, which is a specific entity in a program or the unit of code representing that entity.

**COMPONENTS OF A CLASS DIAGRAM:**

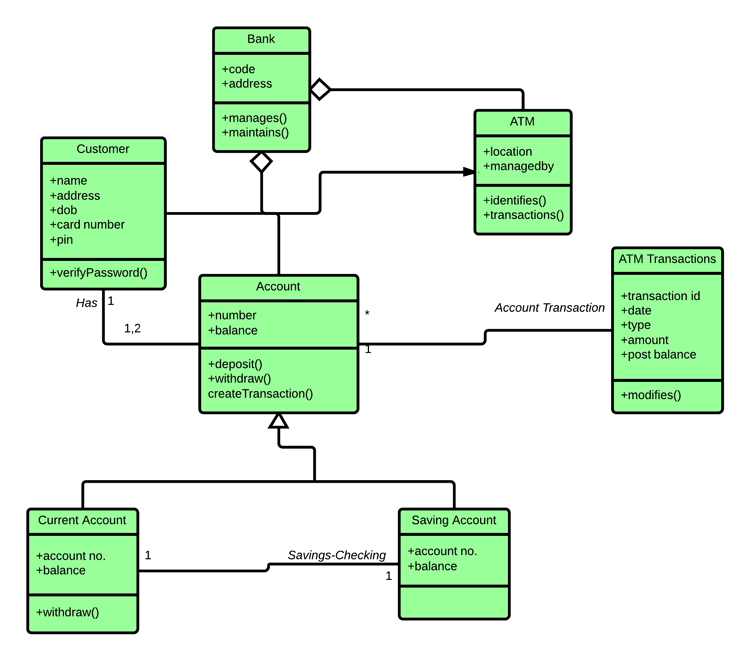
The standard class diagram is composed of three sections:

* **Upper section:**Contains the name of the class. This section is always required, whether you are talking about the classifier or an object.
* **Middle section:**Contains the attributes of the class. Use this section to describe the qualities of the class. This is only required when describing a specific instance of a class.
* **Bottom section:**Includes class operations (methods). Displayed in list format, each operation takes up its own line. The operations describe how a class interacts with data.

**HOW TO CREATE A CLASS DIAGRAM:**

The term "interactions" refers to the various relationships and links that can exist in class and object diagrams. Some of the most common interactions include:

* **Inheritance:** The process of a child or sub-class taking on the functionality of a parent or superclass, also known as generalization. It's symbolized with a straight connected line with a closed arrowhead pointing towards the superclass.
* **Bidirectional association:** The default relationship between two classes. Both classes are aware of each other and their relationship with the other. This association is represented by a straight line between two classes.
* **Unidirectional association:** A slightly less common relationship between two classes. One class is aware of the other and interacts with it. Unidirectional association is modeled with a straight connecting line that points an open arrowhead from the knowing class to the known class.



**Exercise**

Draw class diagram for the following scenario

Implement class diagrams with C#

* Hotel management system

**Problem statement:**

A hotel has various types of rooms Dimensions of various: price, number of single beds, number of double beds. A database with a listing of all the rooms of the hotel is supplied. This database includes when the rooms have been booked.

People can look for availability on a website for certain types of room (room price), for a certain time of span. The systems check availability and returns the proposition that fits the reservation. If no exact match is found, something similar is proposed with at least the same person capacity.

* Airline reservation system

**Problem Statement:**

To develop a computerized meeting, the rising customer interest in booking online air travel reservations. The system should be convenient, user friendly and available via the internet.

The system should allow the users to view entire flights information of the airline, book tickets, view or, if required cancel current reservations and create member login for standalone users as well as agents.