



**ebook**

# SOFTWARE ENGINEERING DSE

**SOFTWARE DESIGN USING OBJECT ORIENTED  
ANALYSIS (USE CASE DIAGRAMS)**

## Lesson 06 – Software Design Using Object Oriented Analysis (Use Case Diagrams)

### Object Oriented Analysis and Design

Object-Oriented Programming concepts originate from a broader approach to the entire software development life cycle known as object-oriented analysis and design (OOAD).

**OOAD is a technical method of analyzing and designing an application based on that system's object models (the logical components of the system that interact with one another).**

Object-Oriented analysis is a process that groups items that interact with one another, typically by class, data or behavior, to create a model that accurately represents the intended purpose of the system as a whole.

### Unified Modeling Language (UML)

- The Unified Modeling Language (UML) is a graphical language for Object-Oriented analysis **that gives a standard way to write a software system's blueprint.** It helps to visualize, specify, construct, and document the artifacts of an object-oriented system.
- It is used to depict the structures and the relationships in a complex system.
- UML model describes what a system is supposed to do.

### Types of UML Diagrams

The two broadest categories that encompass all other types are Behavioral UML diagram and Structural UML diagram.

#### Behavioral UML Diagram

- Activity Diagram
- **Use Case Diagram**
- Interaction Overview Diagram
- Timing Diagram

## Software Engineering

- State Machine Diagram
- Communication Diagram
- **Sequence Diagram**

### Structural UML Diagram

- **Class Diagram**
- Object Diagram
- Component Diagram
- Composite Structure Diagram
- Deployment Diagram
- Package Diagram
- Profile Diagram

### Behavioral UML Diagram

UML behavioral diagrams visualize, specify, construct, and document the dynamic aspects of a system. The most important behavioral diagrams are **Use Case Diagrams and Sequence Diagrams**.

#### Use Case Diagram

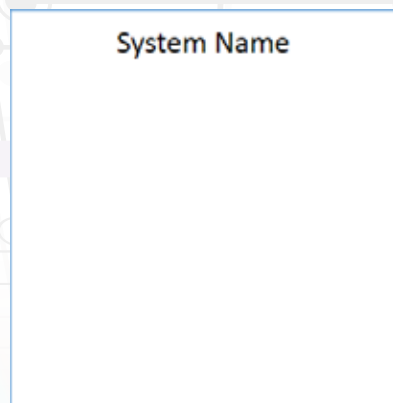
- A use case describes the sequence of **actions (methods)** a system performs yielding visible results. It shows the interaction of things outside the system with the system itself.
- Use case diagram help to understand **how a system should behave**.
- This helps you to gather **requirements from users' point of view**.

## Software Engineering

## Use Case Diagram Notations and Symbols

## System

Draw your system's boundaries using a rectangle that contains use cases. Place actors outside the system's boundaries.



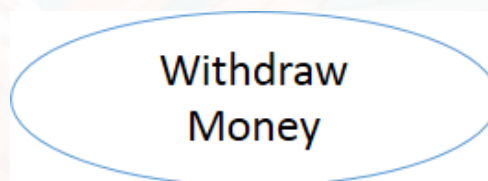
## Use Case

System function (automated process or manual process)

Named by verb + Noun (or Noun Phrase).

E.g.: Withdraw Money

Each Actor must be linked to a use case, while some use cases may not be linked to actors.



## Actor

Someone interacts with use case (system function).

Named by noun. Ex: Customer or Supplier

Actor triggers use case(s).

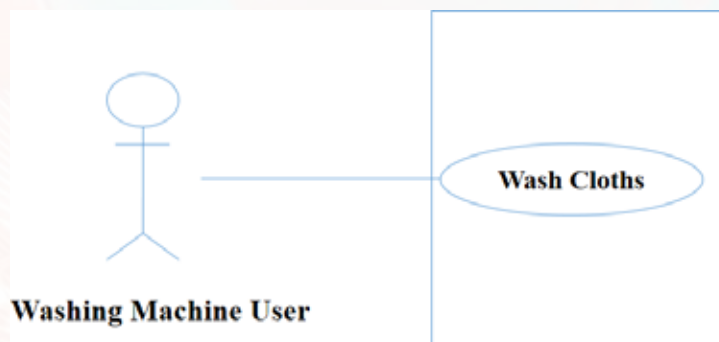
Actor has a responsibility toward the system (inputs), and Actor has expectations from the system (outputs).



## Software Engineering

**Customer****Relationship (Communication Link)**

The participation of an actor in a use case is shown by connecting an actor to a use case by a solid link.

**Use Case Diagram Example***Figure 6.0.1 Use Case Diagram Example***Different types of Relationship**

- Association between Actor and the Use Case
- Include between two Use Cases
- Extends between two Use Cases
- Generalization of a User Case
- Generalization of an Actor

## Software Engineering

### Association between Actor and the Use Case

This one is straightforward and present in every use case diagram.

- An actor must be associated with at least one use case.
- An actor can be associated with multiple use cases.
- Multiple actors can be associated with a single use case.

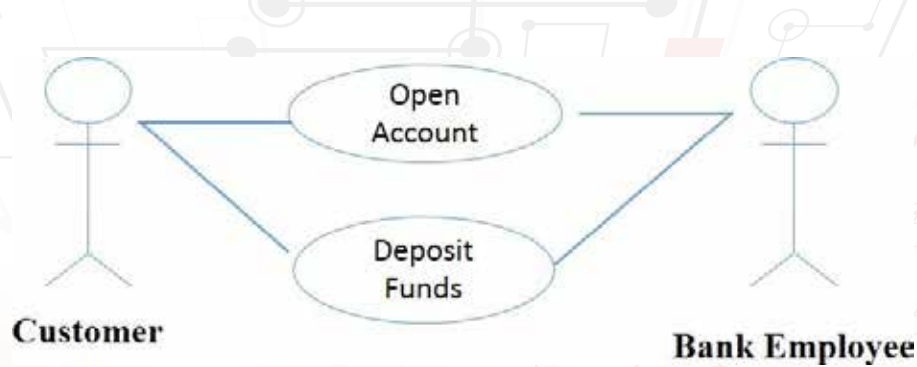


Figure 6.0.2 Association between Actor and the Use Case

### Include

When a use case is depicted as using the functionality of another use case. 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.

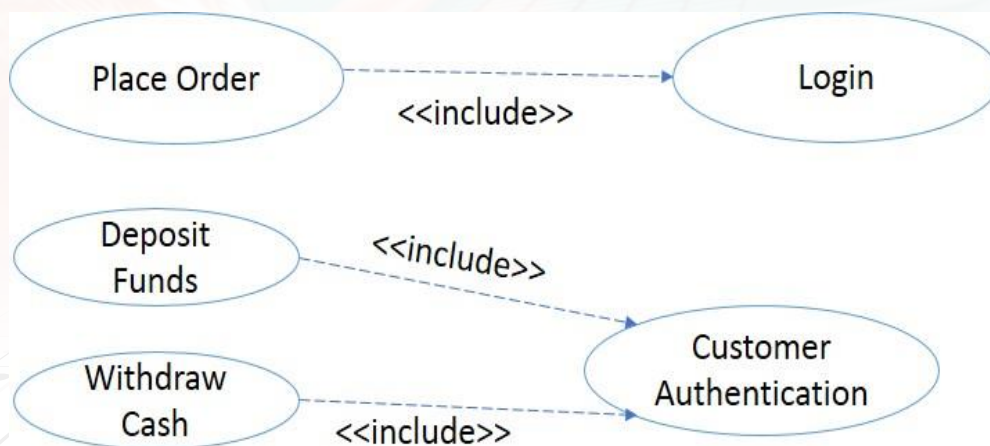


Figure 6.0.3 Include

## Software Engineering

### Extend

Is a directed relationship that specifies how and when the behavior defined in usually supplementary (optional)? The extending use case is dependent on the extended (base) use case.

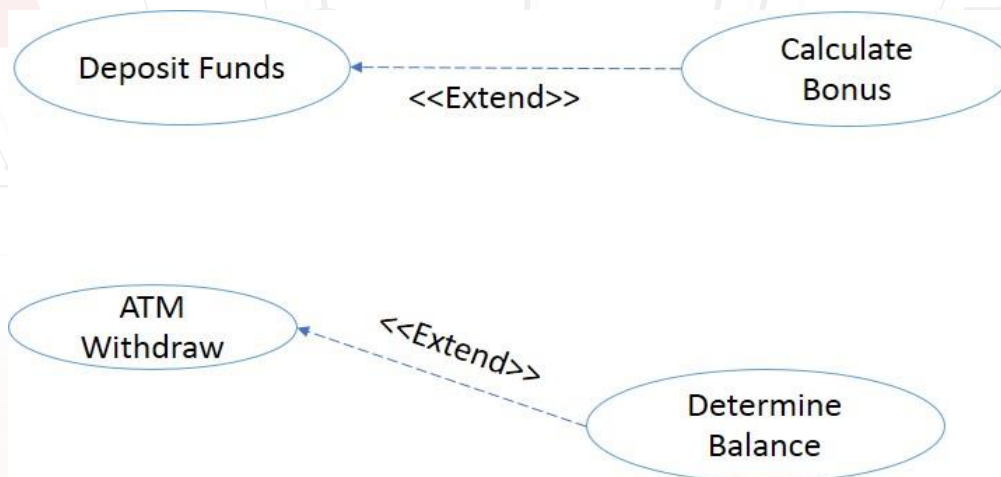


Figure 6.0.4 Extend

### Generalization of a Use Case

The relationship which can exist between two use cases and which shows that one Use Case (child) inherits the structure, behavior, and relationships of another actor (parent).

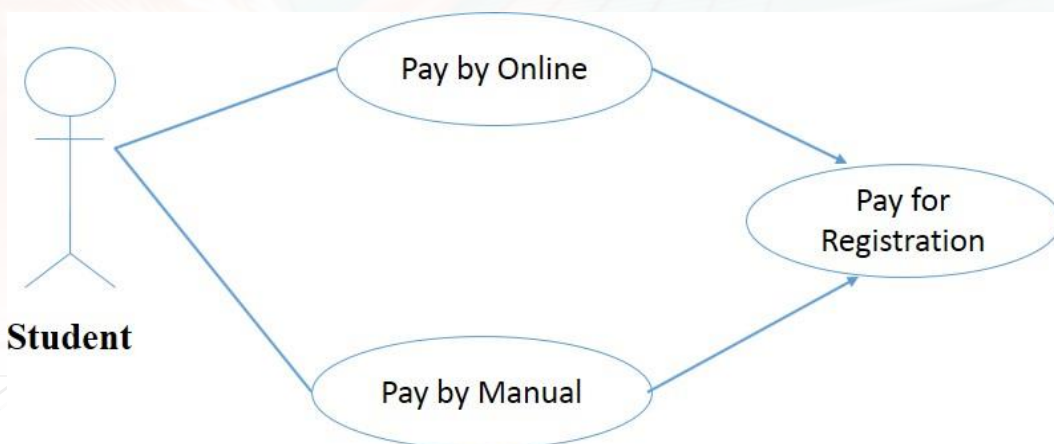


Figure 6.0.5 Generalization of a Use Case

## Software Engineering

### Generalization of an Actor

Generalization of an actor means that one actor can inherit the role of the other actor.

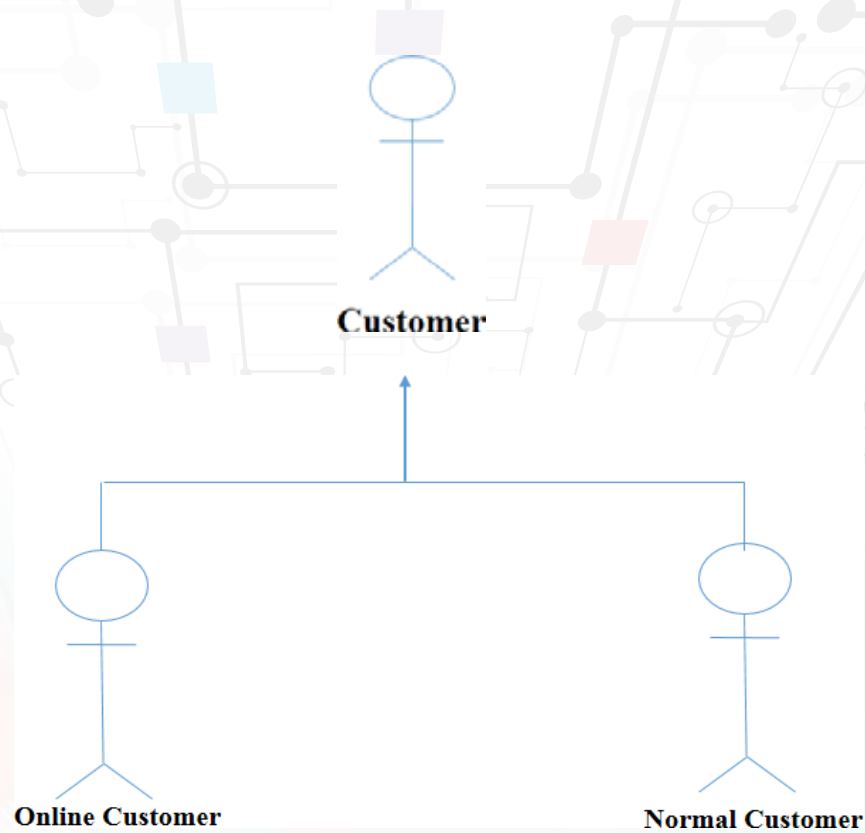


Figure 6.0.6 Generalization of an Actor

### Include Relationship

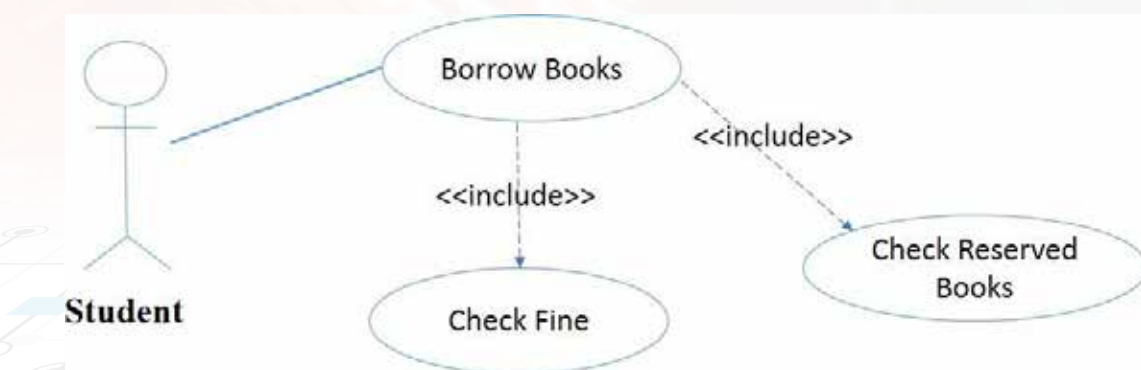


Figure 6.0.7 Include Relationship



### Extend Relationship



Figure 6.0.8 Extend Relationship

### Generalization Relationship

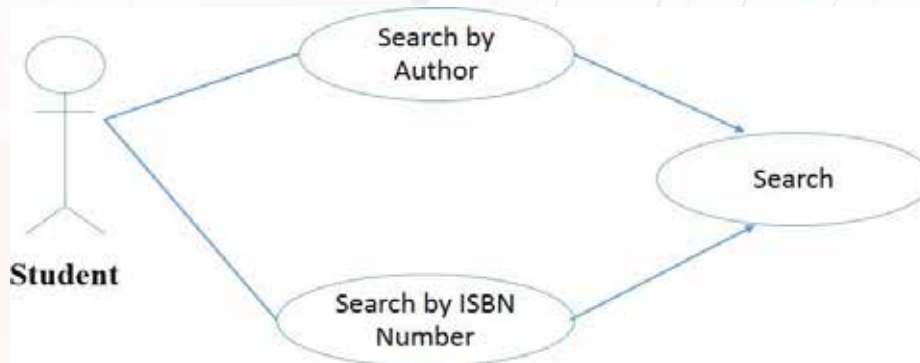


Figure 6.0.9 Generalization Relationship

### How to Identify Actor

- Who uses the system?
- Who installs the system?
- Who starts up the system?
- Who maintains the system?
- Who shuts down the system?
- What other systems use this system?
- Who gets information from this system?

## Software Engineering

- Who provides information to the system?

### How to Identify Use Case?

- What functions will the actor want from the system?
- Does the system store information? What actors will create, read, update or delete this information?
- Does the system need to notify an actor about changes in the internal state?
- Are there any external events the system must know about? What actor informs the system of those events?

### Use Case Diagrams RULES

- **An actor initiates a use case**, and an actor receives something of value from the use case.
- **Initiating actor is on the left of the use case, and receiving actor is on the right.**
- Actors name appears just below the actor.
- Name of the use case appears either inside the ellipse or just below it.
- An association lines connect an actor to the use case and represent the communication between actor and the use case.
- Actors are outside of the system and use cases are inside the system.
- Use a rectangle to represent the system boundary.

### Exercise 01: Banking System

A Customer open a Savings Account with the help of Bank Employee. Customer can Deposit funds as well as Withdraw funds. Customer can Check the Account Balance and later customer can Close the Account if he/she wants. Customer can obtain the Loans with the help of Manager.

## Software Engineering

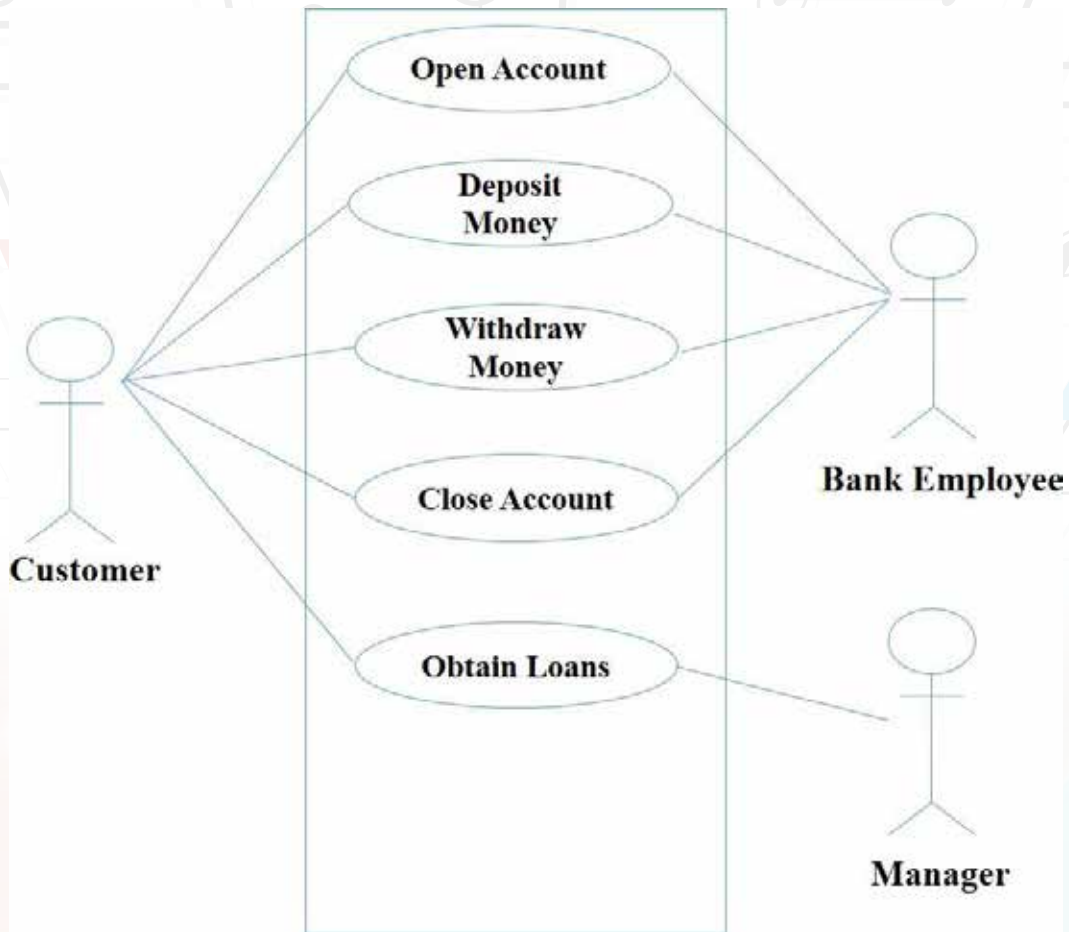


Figure 6.0.10 Use case diagram – Banking system example 1

### Exercise 02: Banking System

There are three Customer Types: Online Customer, Walking Customer and NRFC Customer. Walking Customer open a Savings Account with the help of Bank Employee, But Online Customer and NRFC customer can open a savings account Online without Bank Employee.

Customer can Deposit funds as well as Withdraw funds. When deposits funds Bank calculate the Bonus if the amount is over 10,00 and if the customer's age over 45. When Depositing and Withdrawing the money Updated Balance is a mandatory function.

Customer can Check the Account Balance.

Customer can obtain the Loans with the help of Manager.

NRFC Customer Can Convert the Currency.

## Software Engineering

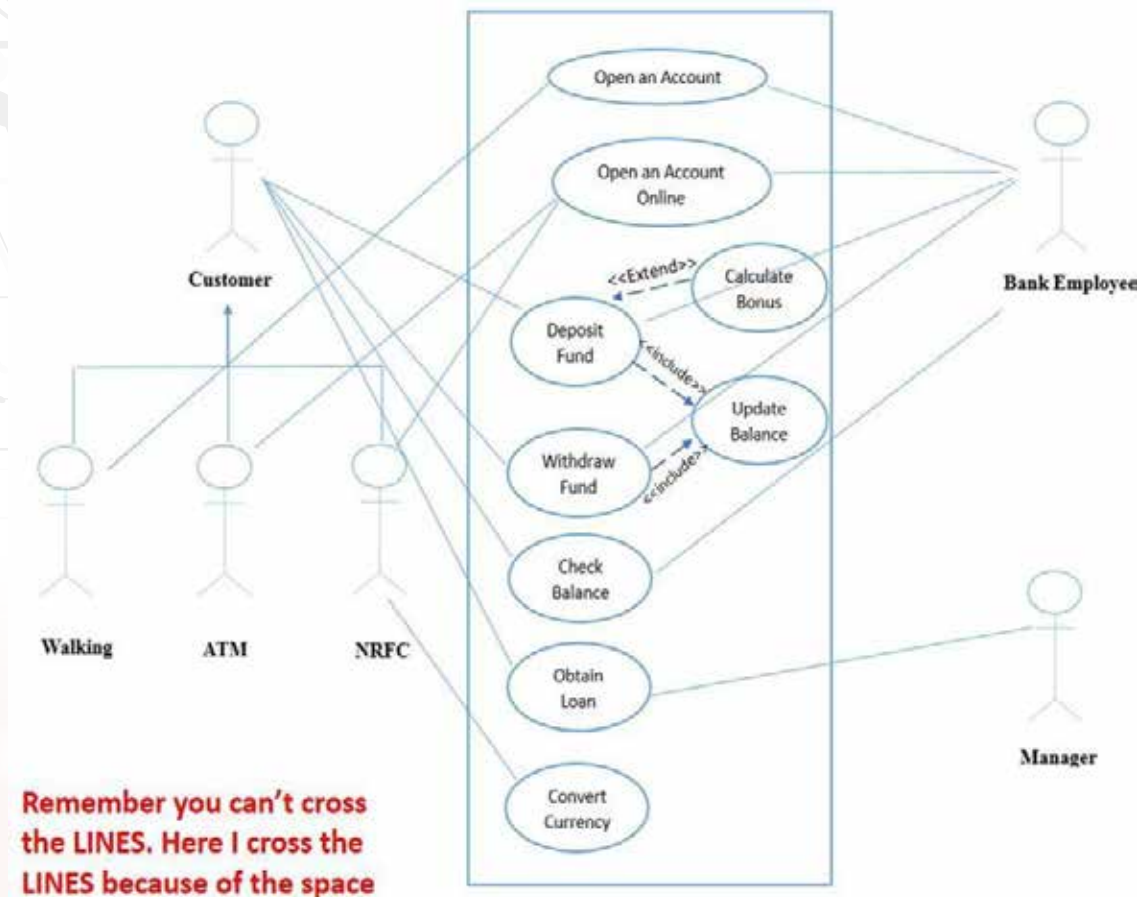


Figure 6.0.11 Use case diagram – Banking system example 2

### Exercise 03: Student LMS System

The university wants to develop an automated system that would enable Program Office Staff to add the new Course details to the System. Also they can update and delete the course details. Before adding the new course details, staff account should be verified.

Lecturer can add modules to the course. Students can view all the details of all modules offered for a particular semester. Student can drop any module which he/she already registered and submit any assignment online. The system should also enable the lectures to check lecture schedules and access the submitted work of the students.

When submitting the Assignments, students can view the Lecturer name if they needed.



## Software Engineering

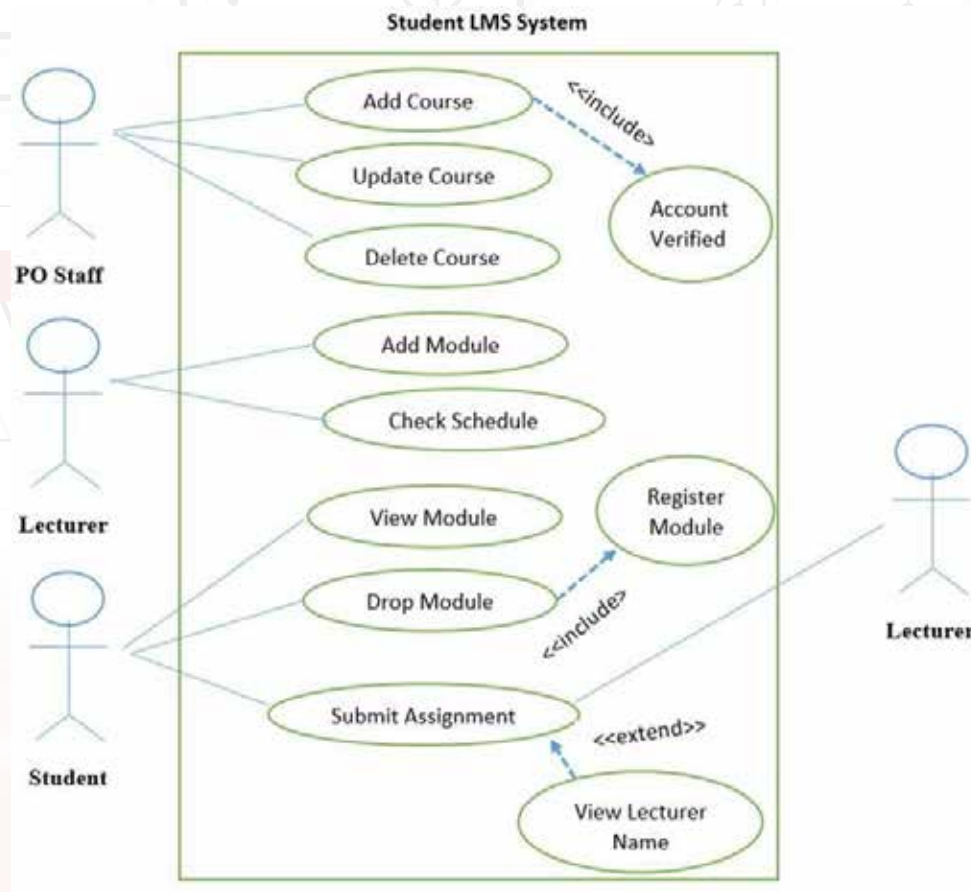


Figure 6.0.12 Use case diagram – Student LMS system

### Exercise 04: Online Shopping System

Food City wants to develop an Online Shopping System where Customer can register first, then Customer Can Place his/her order. Before place the order customer **must check the availability of the item**. After placing the order customer can view the Order as well.

Customer Service of the Food City can view the order of the customer and update the order status. Also they can add, update, delete items from the system. When item is not available purchasing department create a purchase order and send it to the Supplier. Before that supplier should be register in the system. Once the items delivered, purchasing department maintain a GRN and update the inventory.

Customer can do the payments using credit cards or cheque. System automatically generates Monthly Sales Report and Daily Stock Report which can be view by only Branch Manager.

## Software Engineering

When viewing the reports Manager can change the options of the display report (Example: Monthly to Quarter, or Daily to Week)

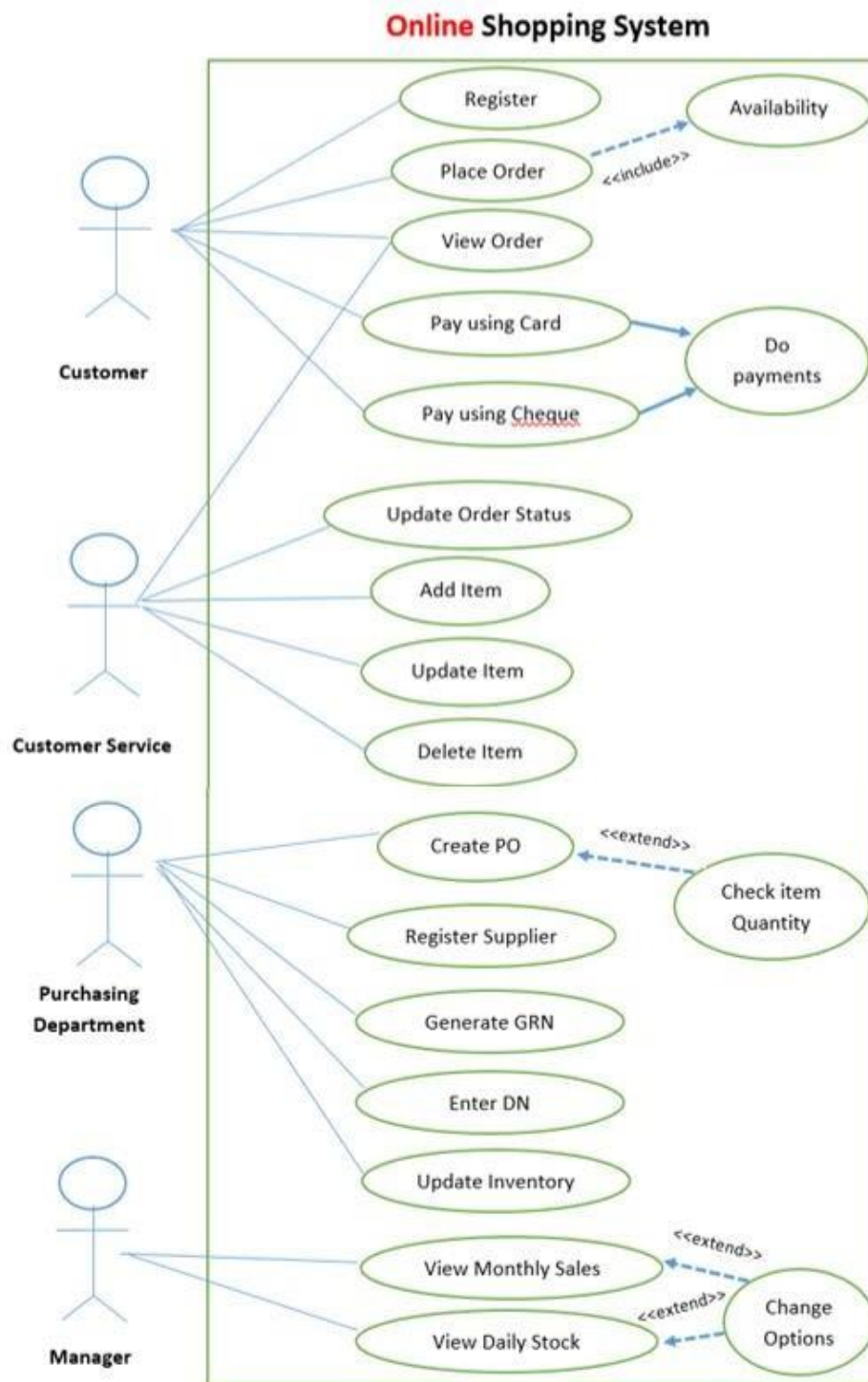


Figure 6.0.13 Use case diagram – Online shopping system

## Software Engineering

### UML Diagrams Tools

Since it is very difficult to draw Use Case and other diagrams in word document you can use following web site to sign up for free UML designs.

You can use **Microsoft VISIO** tool instead of Word. Or else you can use following Online Tools for UML design.

<https://www.lucidchart.com/>

<https://www.smartdraw.com/>