Semester	T.E. Semester V – Computer Engineering
Subject	Software Engineering
Subject Professor In-charge	Dr. Sachin Bojewar
Assisting Teachers	Prof. Sneha Annappanavar
Laboratory	M313B

Student Name	Deep Salunkhe
Roll Number	21102A0014
TE Division	A

Title: Use-Case Model

Explanation:

A Use Case Model is a visual representation of the functional requirements of a system from the perspective of its users or external entities. It's an essential part of the Unified Modeling Language (UML) and is often used in software engineering and system design to capture and document the interactions between users (actors) and the system itself. Here's a detailed explanation of the Use Case Model:

Key Components of a Use Case Model:

1. **Use Case:** A use case represents a specific functionality or behavior of the system that provides value to an actor. Use cases are typically described in plain language and are not overly technical. They answer the question, "What can the system do for its users?" Each use case should have a unique name, such as "Register User" or "Place Order."
2. **Actor:** An actor is an external entity (such as a user, another system, or hardware device) that interacts with the system by participating in one or more use cases. Actors are represented as stick figures in use case diagrams.
3. **Relationships:** Use cases and actors are connected through various types of relationships:
    - **Association:** A straight line connecting an actor to a use case represents an association, indicating that the actor interacts with the use case. This is the most basic relationship.
    - **Inclusion:** An inclusion relationship (represented by a dashed arrow) indicates that one use case includes the functionality of another use case. For example, "Make Payment" might include "Select Payment Method."
    - **Extension:** An extension relationship (represented by a dotted arrow) indicates that one use case can extend another use case under certain conditions. For example, "Cancel Order" might extend "Place Order" if the user decides to cancel the order after placing it.
4. **System Boundary:** The system boundary, represented by a box or rectangle, encloses all the use cases of the system. It defines the scope of the system being modeled.

Benefits of a Use Case Model:

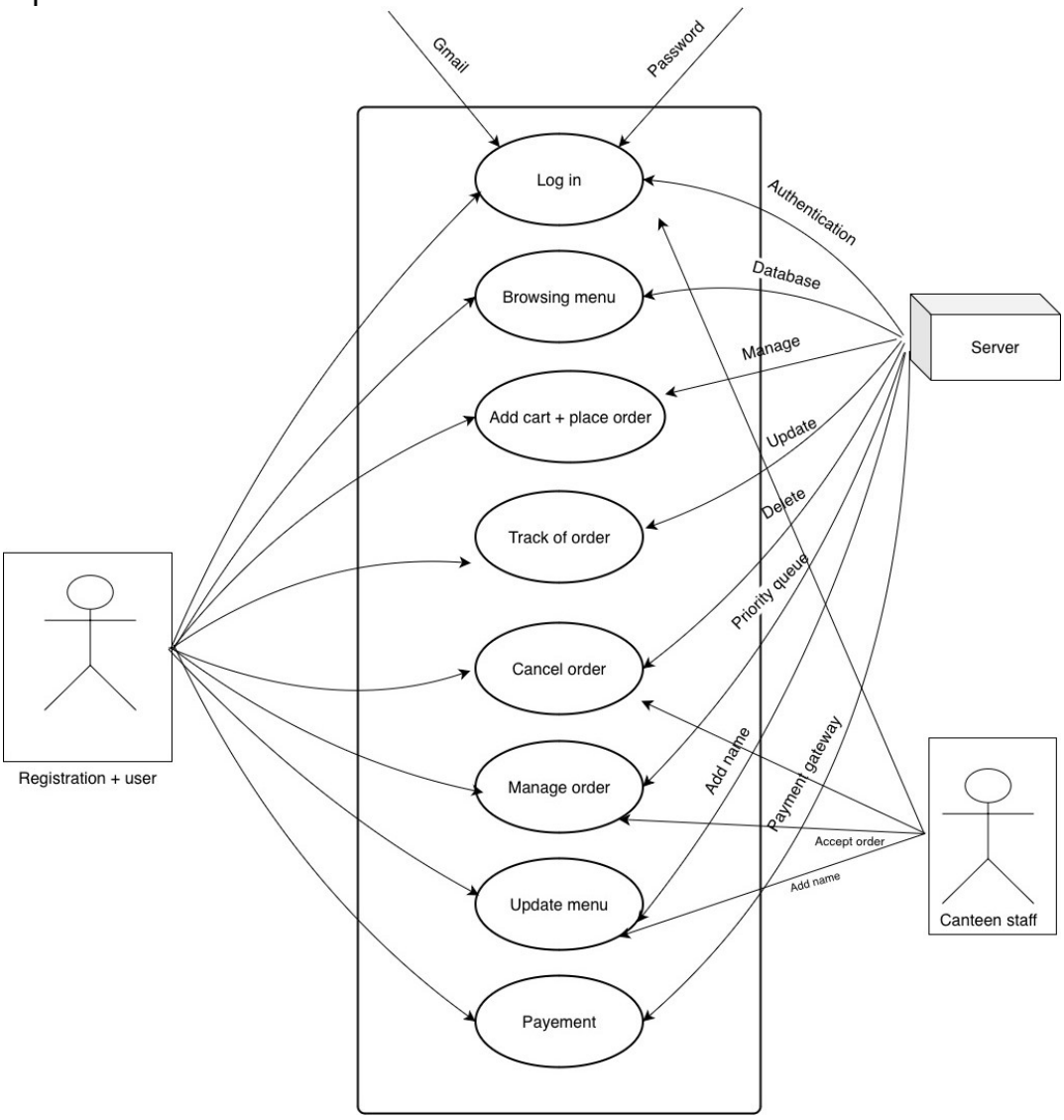
1. **Clarity:** Use case models provide a clear and concise way to document the functional requirements of a system, making it easier for stakeholders to understand what the system is supposed to do.

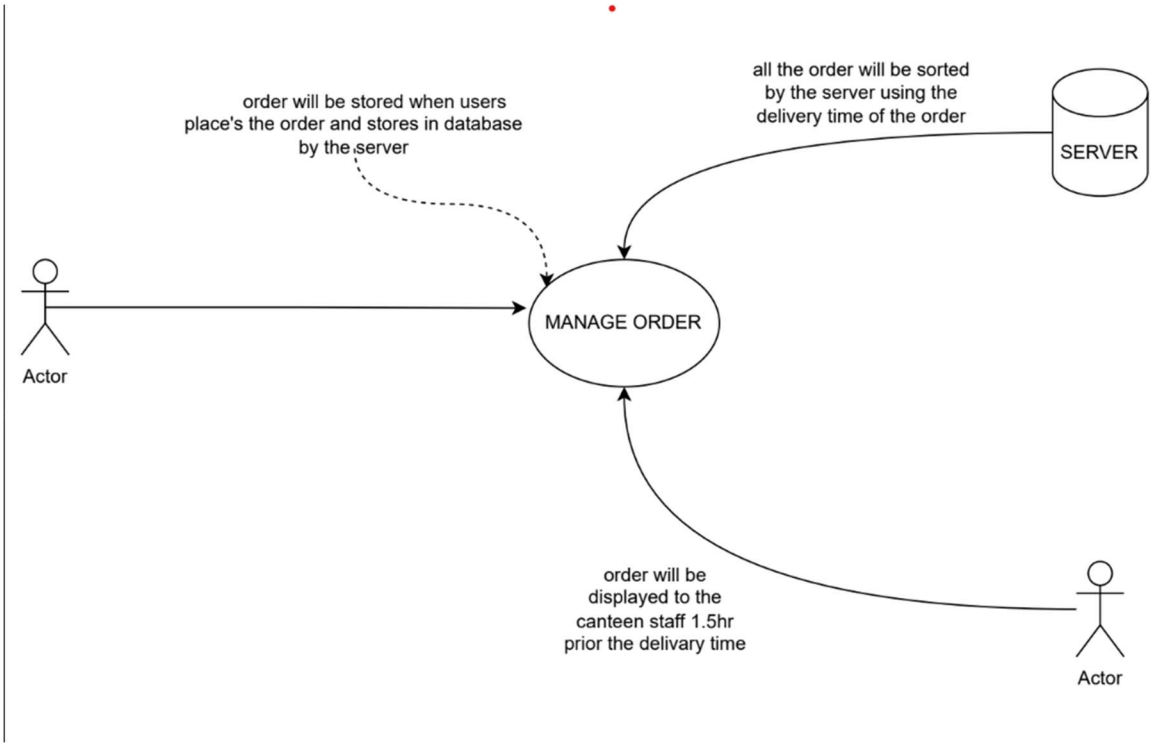
2. **Communication:** They serve as a communication tool between business analysts, developers, and other stakeholders. Non-technical stakeholders can easily grasp the system's functionality.
3. **Scope Definition:** Use cases help in defining the boundaries of the system by identifying what interactions occur between actors and the system.
4. **Use Case Prioritization:** It allows for the prioritization of use cases based on their importance to the business or project goals.
5. **Basis for Testing:** Use cases can be used as the basis for creating test cases to ensure that the system behaves as expected.
6. **Change Management:** They make it easier to manage changes to the system's functionality. When a new use case is added or an existing one is modified, its impact on the system can be assessed.

Creating a Use Case Model:

1. **Identify Actors:** Begin by identifying all the external entities (actors) that will interact with the system. These could be users, other systems, or even hardware devices.
2. **Define Use Cases:** For each actor, identify the use cases that represent the interactions they will have with the system. Describe these use cases in clear, user-centric language.
3. **Establish Relationships:** Connect actors to the use cases they participate in using associations. Use inclusion and extension relationships when one use case includes or extends another.
4. **Draw the System Boundary:** Draw a rectangle around all the use cases to define the system's scope.
5. **Detail Use Case Descriptions:** For each use case, provide a detailed description, including preconditions, postconditions, and any alternative flows or exceptional scenarios.
6. **Review and Iterate:** Review the use case model with stakeholders to ensure accuracy and completeness. Iterate as necessary based on feedback

Implementation:





Conclusion:

se case models are valuable in software development for several reasons:

- They help in understanding and communicating system functionality from a user's perspective.
- They serve as a foundation for requirement gathering and analysis.
- They can be used to derive test cases to ensure that the system meets its intended functionality.
- Use cases can evolve as the project progresses, making them a flexible tool for capturing and managing requirements.