

# Use Case Model

Section 4

# Use cases

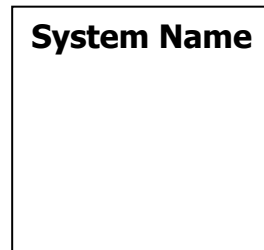
Use cases are descriptions of the functionality of a system from a user perspective.

- Describes the **functionality of a system** from the user's perspective
- Shows how a **user (actor)** interacts with the system to achieve a goal
- Focuses on **what** the system does, not **how** it does it
- Highlights the **relationships** between actors and use cases
- Clarifies the **developer's understanding** of user requirements

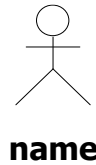
# Components of Use-case Model

The components of a Use-case model are:

- System boundary



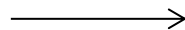
- Actor



- Use case

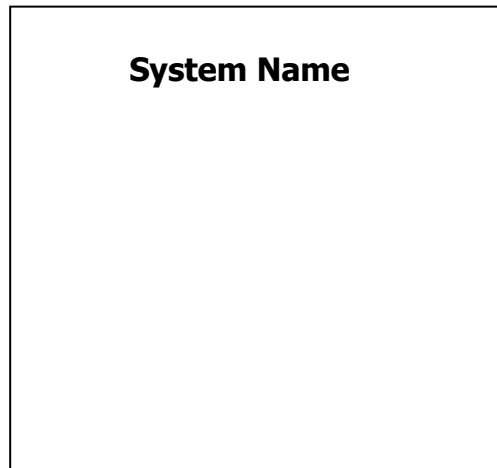


- Relationship



# System

- System in the use-case diagram is represented as a rectangle with the name appearing on the top.
- It helps identify what is external versus internal, and what the responsibilities of the system are.
- The external environment is represented only by actors



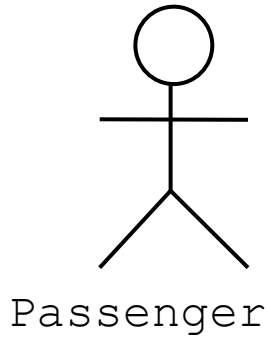
# Step 1: Identify the System

Before modeling, you need to clearly define **the system you are modeling**.

- Ask yourself: “*What system am I describing?*”
- Example: For a train ticket distributor, the system is the **Train Ticket System**.

Draw a **system boundary box** (helps clarify what's inside/outside).

# Actors



- An actor models an external entity which communicates with the system:
  - User
  - External system
  - Physical environment
- An actor has a unique name and an optional description.
- Represented by stick figures outside the system boundary

**An actor defines a role that an external entity assumes during interaction with the system**

# Actors

✗ Wrong:

- *Alex is purchasing a ticket*
- *Pierre is using the ATM*

✓ Correct (use roles):

- *Customer purchases a ticket*
- *Client uses the ATM*

**Actor is a role not an individual instance.**

# Actors

- A single person may be represented by multiple actors if they have multiple roles.
- If several people play the same role, they are represented by one actor.
- An actor is not always human, actors can also be:
  - Devices manipulated by the system (e.g., printers)
  - Existing software integrated into the project
  - External IT systems that interact with the system;
  - ....



# Finding Actors

- The actors of a system can be identified by answering a number of questions:
  - Who will use the functionality of the system?
  - Who will maintain the system?
  - What other system does this system need to interact?
  - ....

# Step 2: Identify Potential Actors

Actors are **entities outside the system** that interact with it. They can be:

- **Human users:** anyone who uses the system directly.
  - Example: Traveler, Ticket Clerk.
- **External systems:** software or hardware that communicates with your system.
  - Example: Central Computer System for updating tariffs.
- **Organizations or devices:** sometimes organizations (like a Payment Gateway) or devices (like a Ticket Printer) can be actors if they interact with the system.

**Always:** Ask: “Who triggers actions in the system or receives information from it?”

# Step 3: Identify Actors

For each potential actor, check:

- Does this actor **initiate a use case**?
- Does this actor **receive information** from the system?

If yes → it is an actor.

**=> Name the Actors:** Use **singular nouns**: Traveler, Central Computer System, Payment Gateway, etc. Keep it simple and **descriptive**.

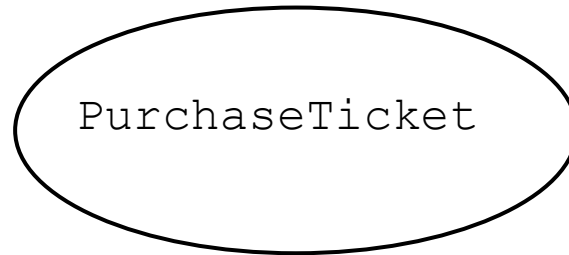
# Exercise – Online Food Ordering System

“A university wants to provide an online food ordering system (OFOS) for students and staff to order meals from campus cafeterias. Users can browse the menu of available cafeterias, place an order selecting multiple items, and pay online using credit/debit card or a campus wallet. Users receive order confirmation and estimated delivery time. Cafeteria staff can view incoming orders and update order status (preparing, ready, delivered). Administrators can generate daily and weekly sales reports. The system should process payments securely and be available 24/7. It should also handle up to 500 concurrent orders without crashing.”

# Use Case

- A **use case** is a pattern of behavior the system exhibits
- Represents a **dialogue between an actor and the system**
- Always **initiated by an actor**
- Must be connected to **at least one actor**

## Example



# Creating Use Cases

## 1 Identify the actors

- Determine the users or external systems that will interact with the system.

## 2 Analyze how each actor will use the system

- Understand how each actor will exploit the system.
- Capture the goals and needs of the actor.

## 3 List candidate use cases

- Enumerate all possible interactions between actors and the system.

## 4 Select the most relevant use cases

- Retain the essential use cases for the system.
- Filter out secondary or optional cases.