Lab 1: Requirements Engineering & UML Use-Case Modelling

Objective:

From a short, instructor-provided scenario, elicit key functions and constraints; write clear, verifiable functional (FR) and nonfunctional (NFR) requirements; then translate them into a UML use-case diagram and one use-case flow.

Duration:

90 minutes

Software Requirements:

Draw.io, LucidChart, StarUML or any other online UML tool.

Learning Outcomes for Lab 1

By the end of this lab, students will be able to:

1. **Elicit and document requirements**:
    - Write clear, testable **functional requirements (FRs)** and **nonfunctional requirements (NFRs)** from a given scenario.
    - Define measurable **acceptance criteria** and justify priorities.
2. **Model system behaviour**:
    - Identify **actors** (e.g., Customer, Admin) and **use cases** for a software system.
    - Construct a **UML use-case diagram** with correct syntax (associations, «include»/«extend» relationships).
3. **Specify use-case flows**:
    - Draft a **main success scenario** and **alternate flows** for a key use case (e.g., payment processing).
4. **Apply industry standards**
    - Format requirements in a structured table (e.g., Req ID, Type, Description) akin to a **Software Requirements Specification (SRS)**.

Note: Students may be randomly called for a presentation after completing the lab. Please be prepared to discuss your code, bugs you found, and how you fixed them.

Introduction:

Use case Modelling

Describes the **interaction of users and the system**

- Describes **what functionality a system provides to its users**.
- Use case models have **two important elements** - **actors** and **use cases**.

Actor/s: One or set of objects who directly interacts with the system. Every actor has a defined purpose while interacting with the system. An actor can be a person, device or another system.

Use case: A piece of functionality that a system offers to its users. Set of all use cases defines the entire functionality of the system. Also define the error conditions that may occur while interacting with the system

How to create Use case diagrams?

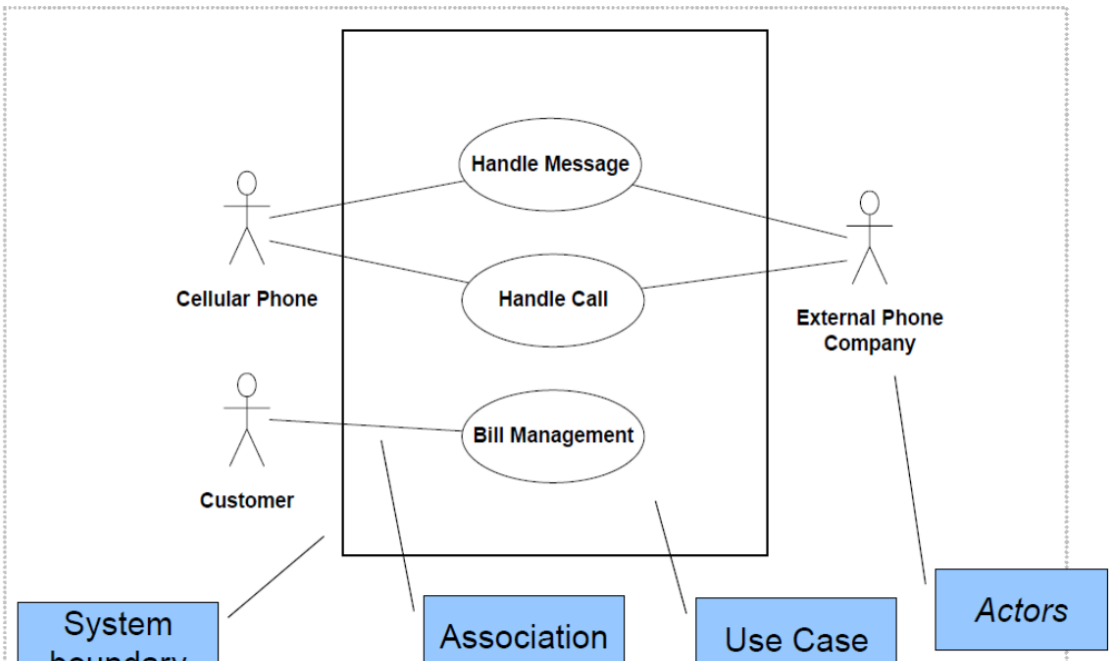
- List main system functions (use cases) in a column
- Draw ovals around the function labels
- Draw system boundary
- Draw actors and connect them with use cases
- Specify include and extend relationships between use cases

"Include" is used when a use case **always calls another use case** as part of its behavior (e.g., *"Place Order"* always **includes** *"Process Payment"*).

"Extend" is used when a use case **optionally adds behaviour** to another (e.g., *"Make Payment"* can be **extended** by *"Apply Discount"*, if applicable).

Use **include** for reuse, and **extend** for optional or conditional behaviour.

Example Use Case Diagram for a simple Telecom System:



Example Use Case Flow for a Simple Telecom System:

Use Case: Recharge Mobile Balance

Main Success Scenario:

1. **Customer** selects the **"Recharge Balance"** option from the Telecom self-service kiosk or mobile app.System displays a form to enter **mobile number** and **amount**.
2. Customer inputs a valid mobile number and recharge amount.
3. System validates the inputs.
4. System requests payment method selection (credit card/debit card/mobile wallet).
5. Customer selects a payment method.
6. System connects to the **Payment Gateway**.
7. Payment is authorized.
8. System updates the **customer balance** in the backend system.
9. A **confirmation message** is shown and a **receipt is printed or emailed**.
10. Use case ends successfully.

Alternate Flow

8a. Payment Declined

- 8a1. System displays a **payment failed** message.
- 8a2. Customer is prompted to try another **payment method**.
- 8a3. If payment fails again (after 2 attempts), the recharge process is **cancelled** and the customer is notified.

Scenario

Self-Service Coffee Kiosk

You are designing the software for a **self-service coffee kiosk** in a busy café. The kiosk must allow customers to:

1. Select a coffee (Espresso, Americano, Latte, Cappuccino).
2. Choose size (Small, Medium, Large).
3. Customize add-ons (Extra shot, Soy/Almond milk, Syrups).
4. Pay via credit/debit card or mobile wallet.
5. Print a receipt with order details and loyalty points added.

The kiosk screen should be responsive and straightforward. Customers must complete a single order in under 60 seconds. Password-protected admin mode allows café staff to refill ingredients, update prices, and view sales reports.

Deliverables

1. **Requirements Table** (Word/Excel):
    - Exactly **five FRs, out of which two are given** (FR-001) and **two NFRs** out of which **one** is already given. (NFR-001), each with:
        1. Req ID
        2. Type (Functional/Nonfunctional)
        3. Description ("The system shall…")
        4. Priority (High/Medium/Low)
        5. Acceptance Criteria (measurable pass/fail)
        6. Rationale (short justification)

2. **UML Use-Case Diagram** (PDF)
   Showing all actors, use cases, and at least one «include» or «extend» relation.

3. **Use-Case Flow Document** (Word, one page):
    - For one key use case (e.g., "Place Order"), provide:
        - **Main Success Scenario** (step-by-step).
        - **Alternate Flow, at least one** (if payment is declined, for example).

Lab 1 Steps (90 min)

1. **Introduction & Scenario Review (5 min)**

2. **Elicit & Draft Requirements (35 min)**
    - **Brainstorm (10 min):**
        - List all the possible functions (e.g., “Select coffee type”) and constraints (e.g., “Order < 60 s”).
    - **Populate Requirements Table (25 min):**
        - Use the template below.

        - Fill in exactly **five FRs** and **two NFRs**.

        - After drafting, each table row must include **measurable** acceptance criteria.
        - **Example FRs/NFRs** (for guidance; students must write their own phrasing):

Req ID	Type	Description	Priority	Acceptance Criteria	Rationale (short)
FR-001	Functional	The system shall allow a customer to select a coffee type (Espresso, etc.)	High	When a customer taps “Espresso,” the screen highlights Espresso and displays “Size”	Core ordering functionality
FR-002	Functional	The system shall allow the customer to select a drink size.	High	When a size is selected, it is shown as chosen and the next step is prompted.	Required for price calculation and preparation
FR-003	Functional				
FR-004	Functional				
FR-005	Functional				
NFR-001	Nonfunc.	The system shall complete any order (selection to payment) in under 60 s.	High	A timing log shows < 60 s from first tap to transaction approval	Ensures quick service during peak hours
NFR-002	Nonfunc.				

3. **Extract Actors & Use Cases (10 min)**
    - From the approved requirements, list:
        - **Actors (at least three)**
        - **Use Cases (at least five)**

4. **Draw an UML Use-Case Diagram (20 min)**
    - Open a UML tool (e.g., **draw.io** or **Lucidchart**).
    - **Actors** on left:
        - Draw stick figures: Actors
    - **Use Cases** (ovals): label them UC-01…UC-05 with titles.
    - **Associations**
        - (At least one «extend»/<<include>> relationship is required: see step 6 below.)
5. **Write One Use-Case Flow (20 min)**
    - Pick a **UC:** In a separate page, draft:

        - **Main Success Scenario**
        - **Alternate Flow**
6. **Finalize & Export (5 min)**
    - **Use-Case Diagram**: Export as PDF.
    - **Use-Case Flow**: Export as PDF.
    - Submit both files plus the revised Requirements Table.