Table of Contents

1. Lab Objectives:

- Familiarize with the concept underlining Use Case Diagram for requirement analysis.
- Explore all elements grouped under UML Use Case Diagram.
- Analyze functional requirements and problem statements of an intended system to be able to translate into use case modeling.

2. Tools Used in the Lab

We will use Star UML to model our Use case diagrams.

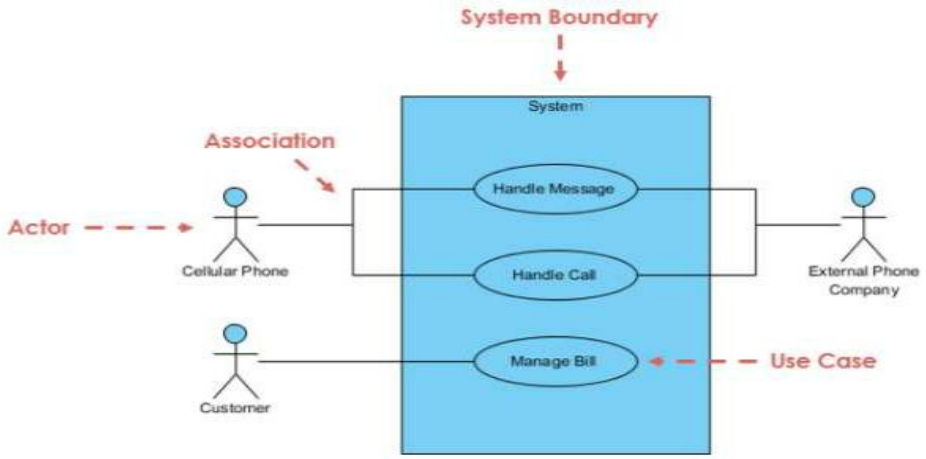
3. Use Case Diagram

A use case diagram is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.

When to use: UML use case diagrams are ideal for:

- Representing the goals of system-user interactions
- Defining and organizing functional requirements in a system
- Specifying the context and requirements of a system
- Modeling the basic flow of events in a use case

4. Use Case Diagram Component:

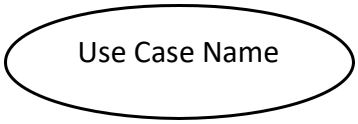


4.1 Actors:

The users that interact with a system. An actor can be a person, an organization, or an outside system that interacts with your application or system. Actors can be primary or secondary. Primary actors are actor(s) using the system to achieve a goal. Secondary Actors are actors that the system needs assistance from to achieve the primary actor's goal.

4.2 Use cases:

A Use Case is a set of scenarios describing an interaction between a user and a system. It displays (in symbolic form) the relationship between actor and use cases. Besides these it also represents the system's functionality and the requirements of the system from the user's perspective.



4.3 Boundary of system:

The system boundary is potentially the entire system as defined in the requirements document. For large and complex systems, each module may be the system boundary. For example, in an organization, each of the modules such as personnel, payroll, accounting, etc.

5. Relations

A. Association

A line between actors and use cases. In complex diagrams, it is important to know which actors are associated with which use cases.

B. Extend

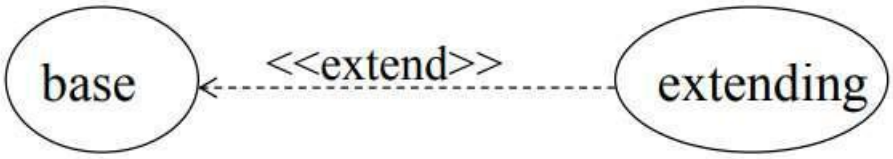
As the name implies it extends the base use case and adds more functionality to the system. Here are a few things to consider when using the <> relationship.

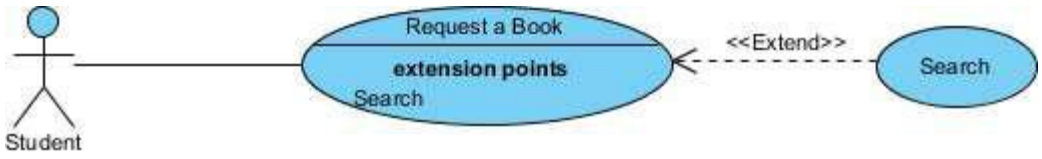
- The extending use case is dependent on the extended (base) use case.
- The extending use case is usually optional and can be triggered conditionally.
- The extended (base) use case must be meaningful on its own.

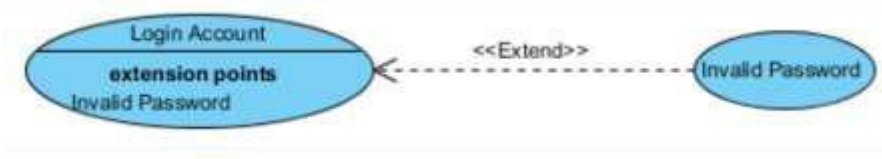
The extend relationships are important because they show optional functionality or system behavior. The <<extend>> relationship is used to include optional behavior from an extending use case in an extended use case. Take a look at the use case diagram example below. It shows an extend connector and an extension point "Search".

For example, a "Make Payment" use case may have different payment options such as credit card or PayPal. If the "Pay with PayPal" option is optional and not always required, then it should be an extension of the "Make Payment" use case.

The tip of arrowhead points to the base use case and the child use case is connected at the base of the arrow.







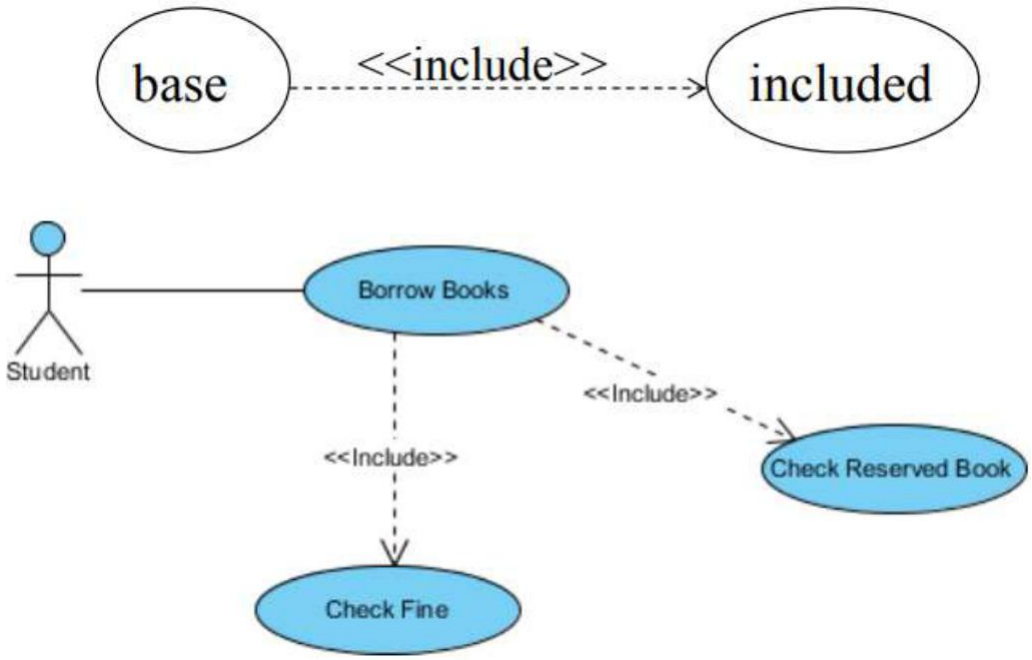
C. Include

A standard case linked to a mandatory use case. If a use case is a basic, standalone functionality that is required by other use cases, then it should be included in those use cases. This means that the behavior of the included use case is performed within the context of the including use case.

The <<Include>> relationship is used to include common behavior from an included use case into a base use case in order to support the reuse of common behavior.

For example, a "Login" use case is a basic functionality that is required by other use cases such as "View Profile" and "Manage Account". Therefore, "Login" should be included in those use cases.

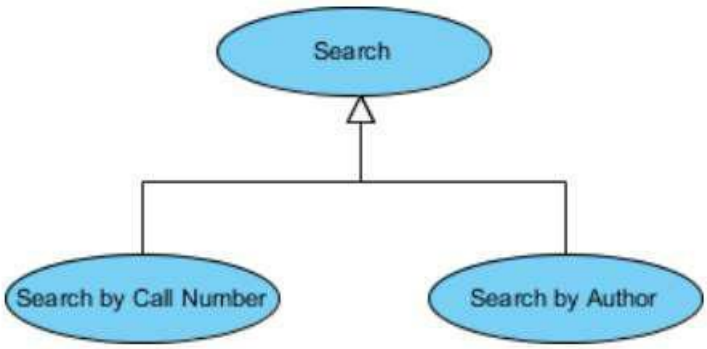
A uses relationship from base use case to child use case indicates that an instance of the base use case will include the behavior as specified in the child use case.



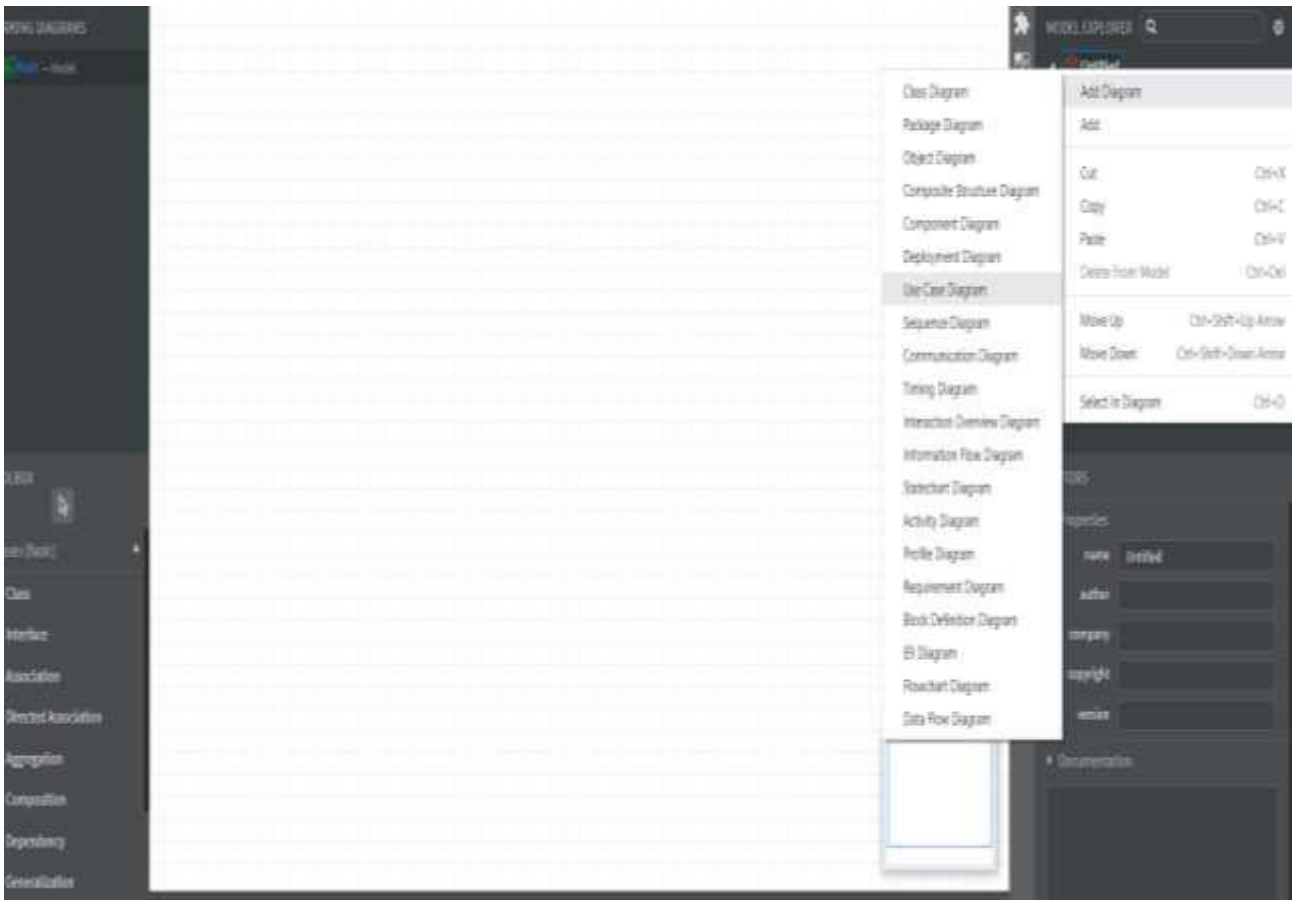
D. Generalization

A generalization relationship is a parent-child relationship between use cases. The child use case is connected at the base of the arrow

In a use case diagram, a generalization relationship is represented by a directed arrow with a triangle arrowhead, pointing from the child use case to the parent use case. The arrow is connected to the base of the child use case and the tip of the arrow is connected to the parent use case.

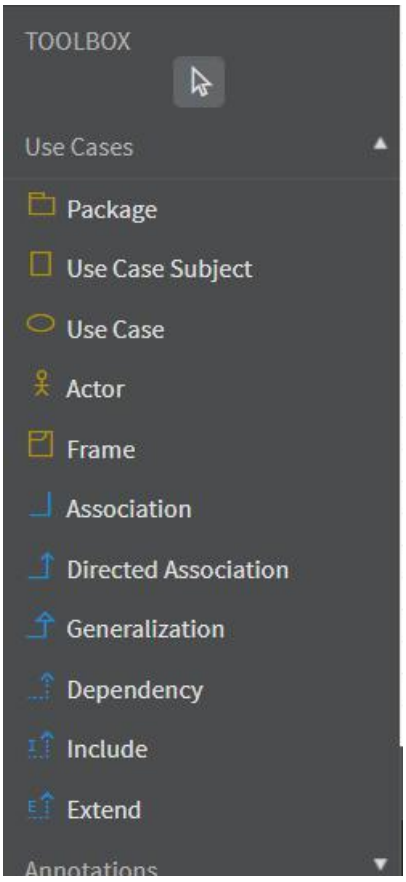


6. StarUML: Adding a Use Case Diagram



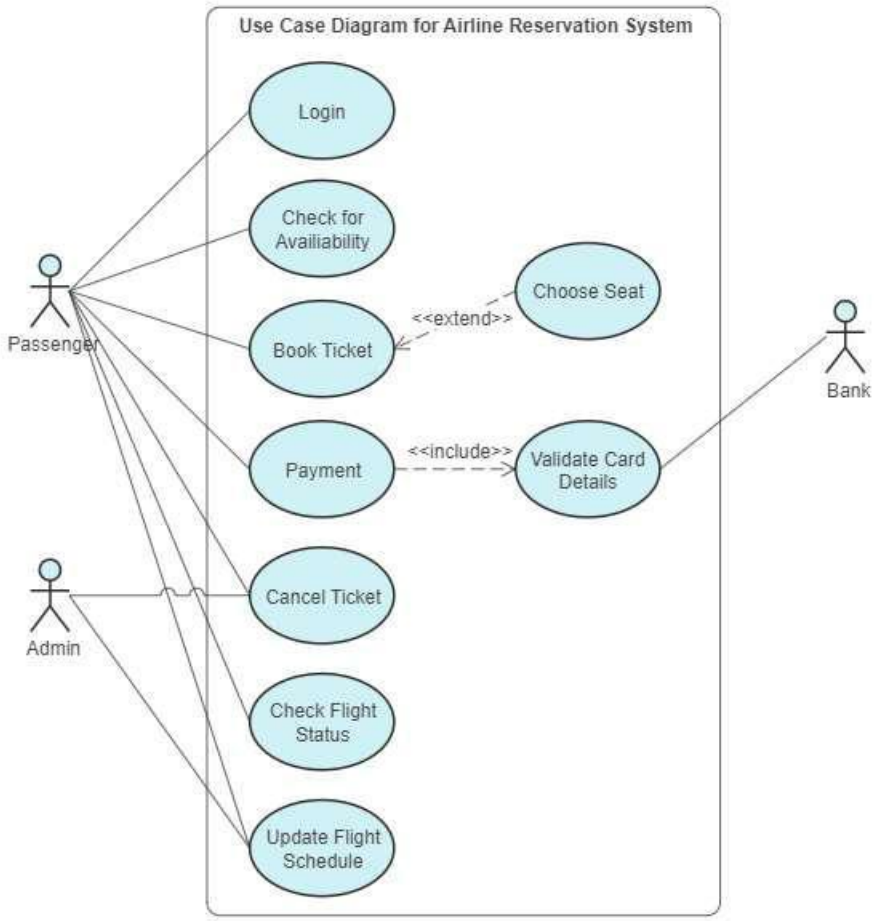
7. StarUML: Object diagram notations

All notations of Use Case Diagram are available in the toolbox on the left side of StarUML



Sample Task

This system is the subsystem of the airline reservation system. The actors are passengers, admins, and the banks that are the organizations. The passenger is concerned with multiple use cases like login, check for availability, book ticket, etc. The book ticket use case is in relation to the choose seat use case. The admin cancels tickets, updates flight schedules. The bank sees the payment use cases.



Lab Task 1:

Customer actor uses bank ATM to Check Balances of his/her bank accounts, Deposit Funds, Withdraw Cash and/or Transfer Funds use case. ATM Technician provides Maintenance and Repairs. All these use cases also involve Bank actor whether it is related to customer transactions or to the ATM servicing.

Lab Task 2:

Draw the Use case diagram for online shopping system where a customer can login to account, view product with search functionality, make purchase with complete checkout process and show all possible actor include in this system.

Lab Task 3:

A Corporate Housing Management System (CHMS) helps multinational companies efficiently manage employee accommodations across multiple office locations. Employees can request company-provided housing based on availability, job roles, and seniority while also registering complaints for maintenance issues or requesting upgrades. The system operates across different cities and time zones, ensuring compliance with various local tenancy laws, tax regulations, and safety policies. Housing eligibility depends on employment contracts, marital status, and tenure, with tiered preferences allowing employees to choose from different accommodation categories such as standard, premium, or executive housing.

To streamline processes, a multi-level approval system ensures that housing requests go through HR, the finance department, and regional housing managers before final allocation. The system also tracks lease agreements, automates lease renewals and terminations, and prevents overbooking while ensuring fair distribution based on company policies. Budget allocation and cost tracking are integrated, allowing admins to oversee expenses, optimize leasing agreements, and allocate funds per department efficiently. Additionally, CHMS connects with payroll and HR systems to adjust housing deductions or allowances accordingly.

The system enhances employee satisfaction by providing personalized housing options while offering real-time data analytics on occupancy rates, maintenance history, cost per employee, and overall housing utilization. AI-driven room allocation ensures optimal distribution based on employee preferences, lease agreements, and real-time availability. In case of emergencies such as safety concerns or natural disasters, employees can request emergency relocations, and the system assists in crisis management. Automated notifications keep employees informed about policy changes, room assignments, and lease updates, significantly reducing administrative workload. Ultimately, CHMS ensures an efficient, compliant, and cost-effective approach to corporate housing while optimizing employee experience and resource allocation.

Lab Task 4:

A **Military Barracks Management System (MBMS)** helps defense organizations efficiently manage accommodations for soldiers and officers across multiple bases. Personnel are assigned barracks based on rank, unit, and operational status, ensuring fair allocation without overbooking. The system factors in deployment schedules, temporary duty assignments, and emergency housing needs, dynamically adjusting room allocations. Maintenance requests for infrastructure issues, security concerns, or facility upgrades can be submitted and tracked in real-time.

Security and compliance are critical, so MBMS ensures that only authorized personnel can access specific accommodations, with biometric authentication and strict clearance levels in place. Automated notifications alert soldiers about room assignments, policy changes, or upcoming facility inspections. The system integrates with military logistics to track personnel movements and preemptively allocate housing for incoming deployments. Additionally, it generates reports on barrack occupancy, maintenance history, and budget utilization for facility upgrades. AI-driven analytics help predict accommodation needs, optimize resource allocation, and reduce operational inefficiencies. By automating approvals and streamlining housing logistics, MBMS enhances efficiency while maintaining security and discipline within military facilities.

Lab Task 5

An **International Student Housing System (ISHS)** assists universities in managing accommodations for students from diverse countries, ensuring fair room assignments while considering cultural, dietary, and language preferences. Students apply for rooms based on factors like program duration, scholarship eligibility, and financial status. Housing is allocated dynamically, ensuring that students with disabilities or special needs receive appropriate accommodations.

The system integrates with visa and immigration departments to verify residency statuses, preventing unauthorized stays and ensuring compliance with university policies. Maintenance requests, roommate conflict resolution, and upgrade requests are handled through a structured workflow, with automated notifications keeping students informed. Since international students often arrive at different times due to varying academic calendars, ISHS dynamically adjusts room allocations to ensure availability. AI-driven insights analyze housing preferences, optimize room matching, and predict seasonal demand. Additionally, the system generates reports on occupancy rates, student satisfaction, and budget utilization, helping universities efficiently manage housing resources while enhancing student experiences.