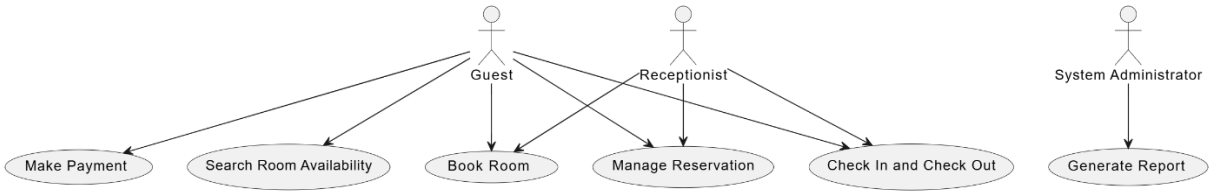
Use Case Diagrams



Use case diagrams are a key part of UML (Unified Modeling Language) and are primarily used to represent the **functional requirements** of a system. Unlike sequence diagrams, which emphasise the order of interactions in a specific scenario, use case diagrams focus on *who* interacts with *what functionalities* within a system. These diagrams provide a high-level view of the system and help clarify the system's intended use, user goals, and the interactions users have with the system.

Key Components of a Use Case Diagram

Use case diagrams are built around the main functionalities of a system, called *use cases*, which are represented as ovals. Here are the essential components:

1. **Actors**: Entities (often users) who interact with the system to achieve a goal.
2. **Use Cases**: Functionalities or services the system provides, each depicted as an oval labelled with a specific task (e.g., "Book Room").
3. **System Boundary**: A rectangle that encapsulates the use cases, representing the scope of the system.
4. **Relationships**:
    - **Associations**: Solid lines connecting actors to use cases, indicating interaction.
    - **Include**: A dotted arrow with the <<include>> stereotype, used to show that a use case includes another use case.
    - **Extend**: A dotted arrow with the <<extend>> stereotype, used to show optional or conditional use cases.

Developing a Use Case Diagram for a Hotel Room Booking System

Step 1: Identify Actors and Use Cases

A hotel room booking system typically has a variety of users and functionalities. For this system, we might consider the following actors:

- **Guest**: A primary user of the system who needs to search, book, and manage reservations.
- **Receptionist**: A user who assists with reservations, updates bookings, and checks in/out guests.
- **System Administrator**: A user who manages system configurations, handles maintenance, and oversees user accounts.

With these actors in mind, the use cases include:

- **Search Room Availability**: Allows users to find available rooms based on dates and requirements.
- **Book Room**: Enables users to reserve a room for specified dates.
- **Make Payment**: Facilitates the payment process to secure a booking.
- **Manage Reservation**: Allows users to update or cancel existing reservations.
- **Check In** and **Check Out**: Supports the guest's entry and exit processes.
- **Generate Report**: An administrative use case for reports, e.g., for occupancy or revenue.

Step 2: Determine Relationships Between Use Cases

Some use cases depend on others, or may even include or extend additional functionality:

- The **Book Room** use case could include **Make Payment** because payment is a part of booking.
- **Check In** may extend from **Book Room**, as only a booked guest would check in.
- The **Generate Report** use case might include data on reservations, requiring association with **Manage Reservation**.

Step 3: Draw the Use Case Diagram

Here's how these components translate into a use case diagram:

1. **System Boundary**: Draw a rectangle to define the boundary, labelled "Hotel Room Booking System."
2. **Actors**: Place actors (e.g., Guest, Receptionist, System Administrator) outside the boundary.
3. **Use Cases**: Inside the boundary, draw ovals for each use case (e.g., Book Room, Search Room Availability, Make Payment).

4. **Relationships**:
    - Draw solid lines between each actor and the use cases they interact with.
    - Use <<include>> arrows to show included processes (e.g., Book Room <<include>> Make Payment).
    - Use <<extend>> arrows to represent optional processes (e.g., Check In <<extend>> Book Room).

Example of Relationships in the Diagram

- The **Guest** actor has an association with **Search Room Availability**, **Book Room**, **Make Payment**, and **Manage Reservation**.
- The **Receptionist** is associated with **Check In**, **Check Out**, and **Generate Report**.
- The **System Administrator** can access **Generate Report** and **Manage Reservation** to monitor and maintain the system.

Practical Use of the Diagram

This use case diagram allows developers, stakeholders, and analysts to visualise the main functionalities needed for a hotel room booking system, identify user roles, and understand how each user will interact with the system. Each component of the diagram plays a role in creating a clear understanding of the system's requirements before diving into specific implementation details like sequence diagrams or class diagrams.

Conclusion

Use case diagrams are an effective tool for capturing the primary functionalities of a system and clarifying user interactions and expectations. In a hotel room booking system, they give an overview of the essential tasks for guests, receptionists, and administrators, helping everyone involved gain a shared understanding of system requirements and scope.