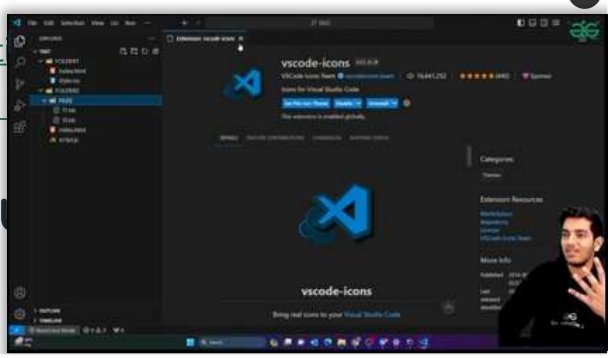
Use Case Diagrams | Unified Modeling Language (UML)

Last Updated : 15 Jul, 2024

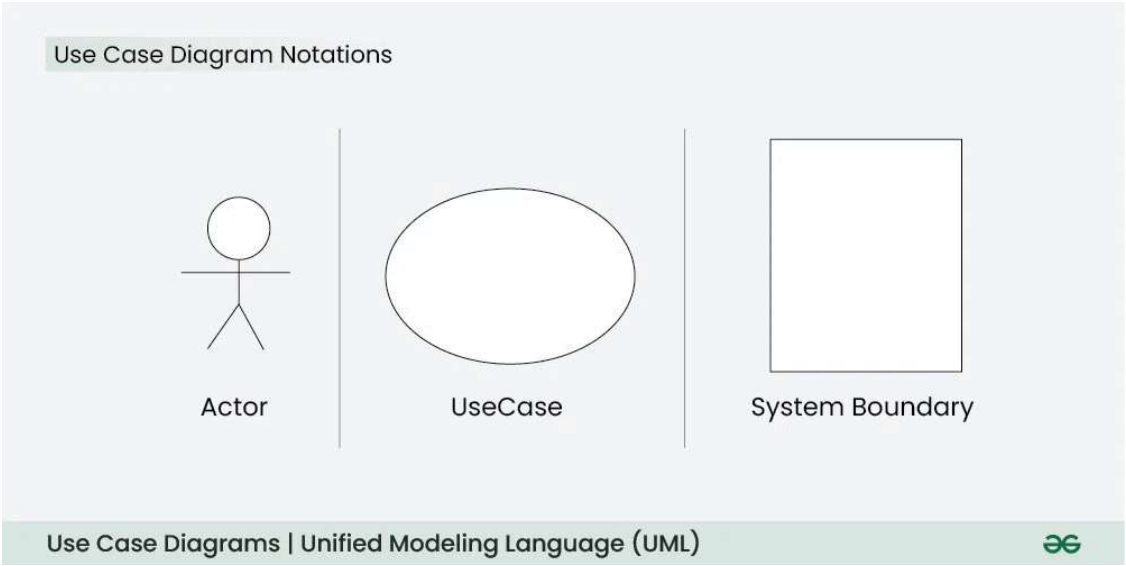
A Use Case Diagram is a vital tool in system design, it provides a visual representation of how users interact with a system. It serves as a blueprint for understanding the functional requirements of a system from a user's perspective, aiding in the communication between stakeholders and guiding the development process.

Important Topics for the Use Case Diagrams



1. What is a Use Case Diagram in

A Use Case Diagram is a type of Unified Modeling Language (UML) diagram that represents the interaction between actors (users or external systems) and a system under consideration to accomplish specific goals. It provides a high-level view of the system's functionality by illustrating the various ways users can interact with it.



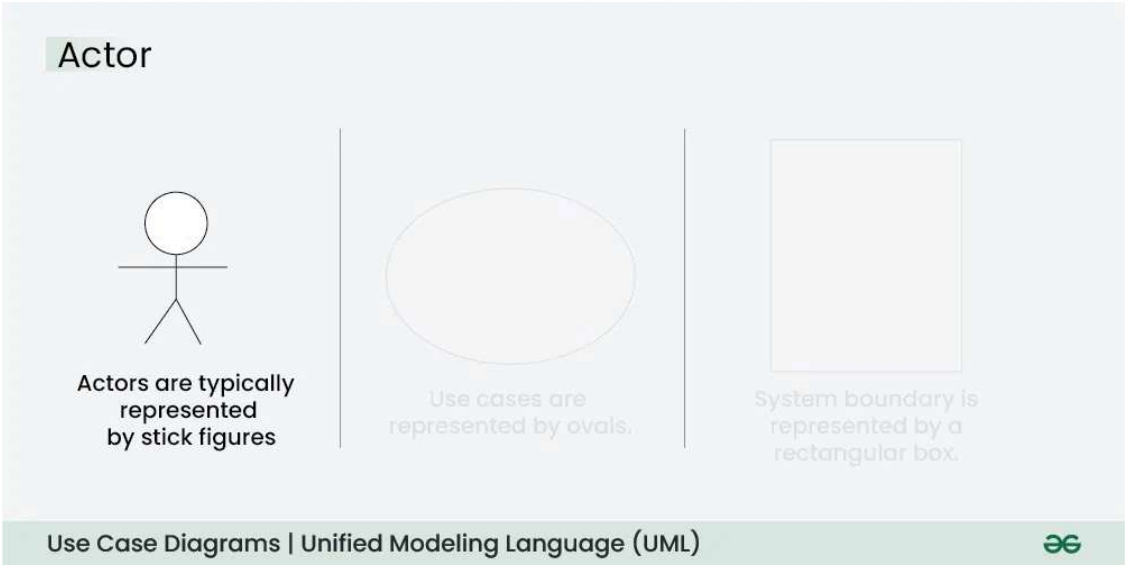
2. Use Case Diagram Notations

UML notations provide a visual language that enables software developers, designers, and other stakeholders to communicate and document system designs, architectures, and behaviors in a consistent and understandable manner.

1.1. Actors

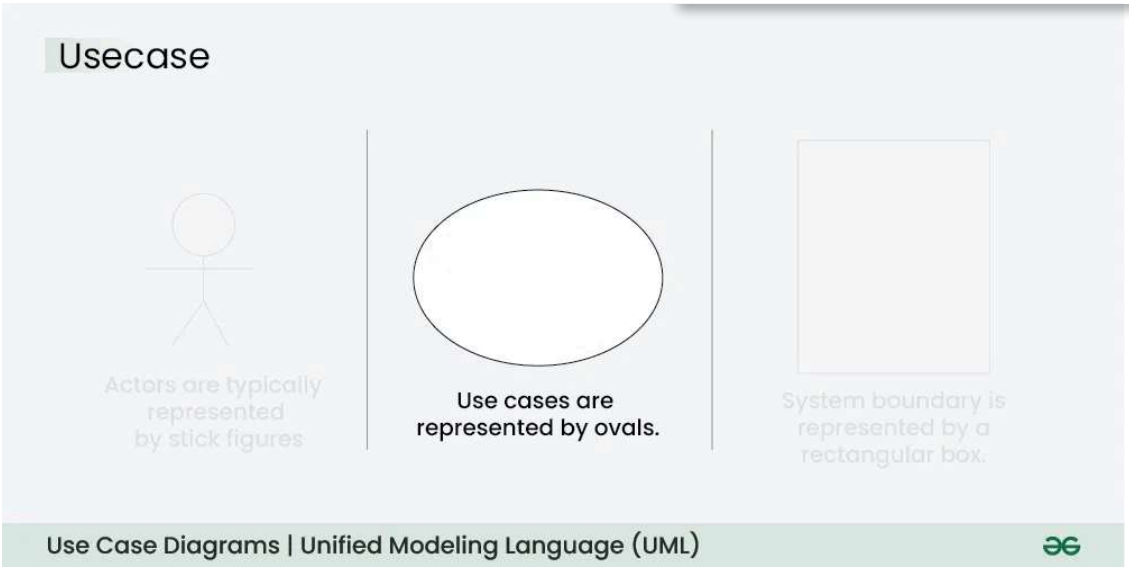
Actors are external entities that interact with t
users, other systems, or hardware devices. In t
Diagram, actors initiate use cases and receive

identification and understanding of actors are crucial for accurately modeling system behavior.



1.2. Use Cases

Use cases are like scenes in the play. They represent specific things your system can do. In the online shopping system, examples of use cases could be “Place Order,” “Track Delivery,” or “Update Product Information”. Use cases are represented by ovals.

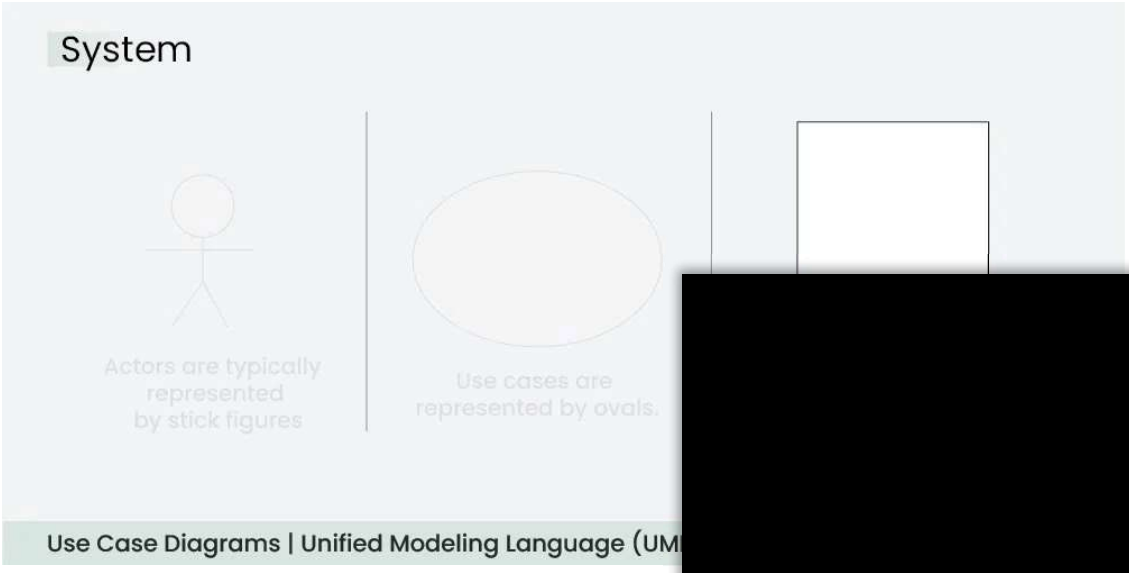


1.3. System Boundary

The system boundary is a visual representation of the scope or limits of the system you are modeling. It defines what is inside the system and what is outside. The boundary helps to establish a clear distinction between the elements that are part of the system and those that are external to it. The system boundary is typically represented by a rectangular box that surrounds all the use cases of the system.

Purpose of System Boundary:

- **Scope Definition:** It clearly outlines the boundaries of the system, indicating which components are internal to the system and which are external actors or entities interacting with the system.
- **Focus on Relevance:** By delineating the system's scope, the diagram can focus on illustrating the essential functionalities provided by the system without unnecessary details about external entities.



3. Use Case Diagram Relationships

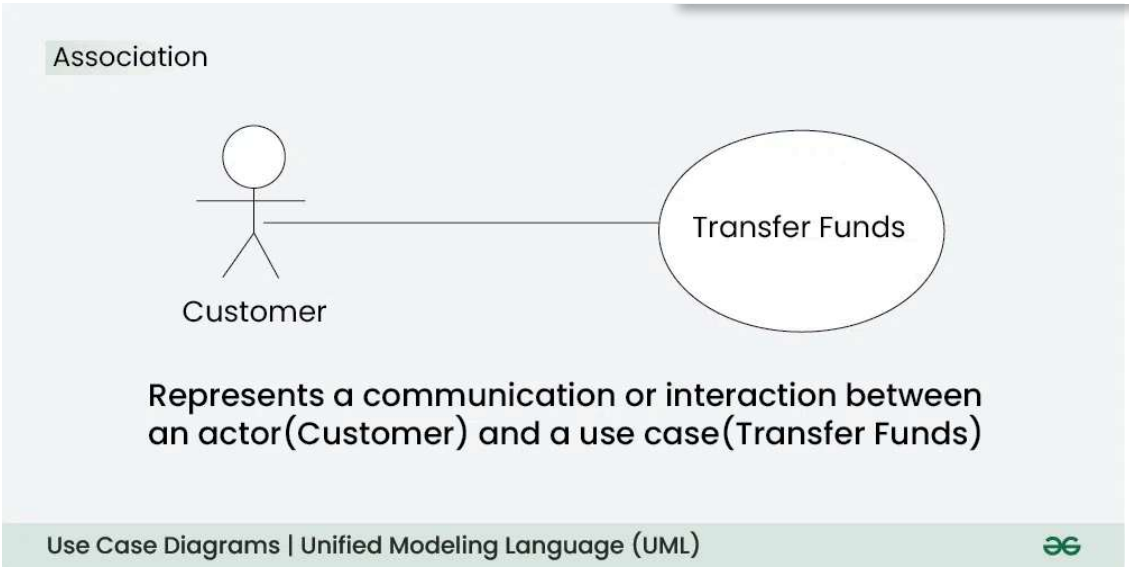
In a Use Case Diagram, relationships play a crucial role in depicting the interactions between actors and use cases. These relationships provide a comprehensive view of the system's functionality and its various scenarios. Let's delve into the key types of relationships and explore examples to illustrate their usage.

3.1. Association Relationship

The Association Relationship represents a communication or interaction between an actor and a use case. It is depicted by a line connecting the actor to the use case. This relationship signifies that the actor is involved in the functionality described by the use case.

Example: Online Banking System

- **Actor:** Customer
- **Use Case:** Transfer Funds
- **Association:** A line connecting the "Customer" actor to the "Transfer Funds" use case, indicating the customer's involvement in the funds transfer process.

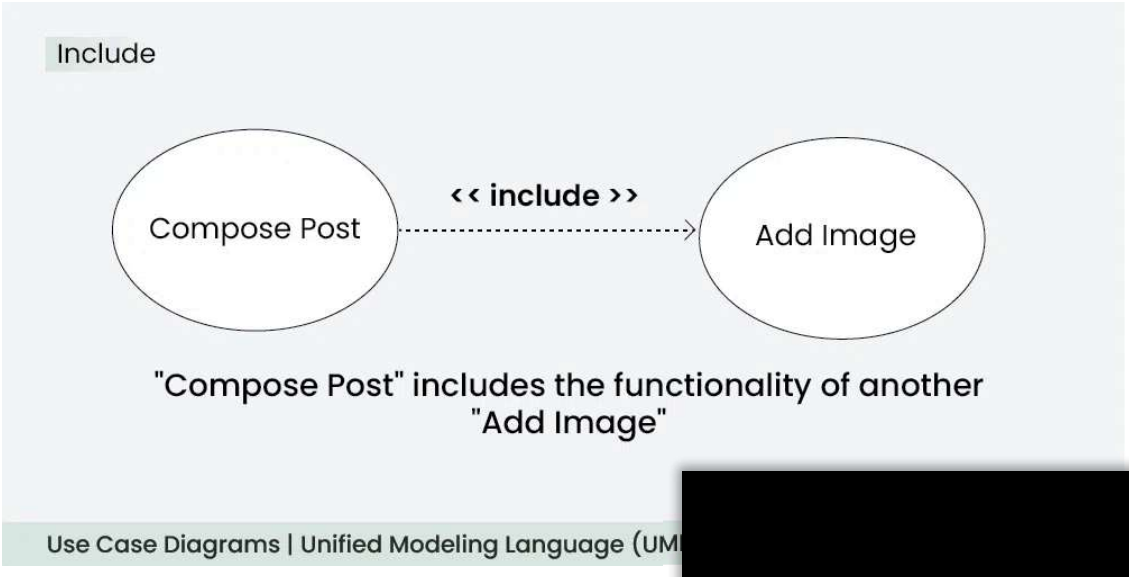


3.2. Include Relationship

The Include Relationship indicates that a use case includes the functionality of another use case. It is denoted by a dashed arrow pointing from the including use case to the included use case. This relationship promotes modular and reusable design.

Example: Social Media Posting

- **Use Cases:** Compose Post, Add Image
- **Include Relationship:** The “Compose Post” use case includes the functionality of “Add Image.” Therefore, composing a post includes the action of adding an image.

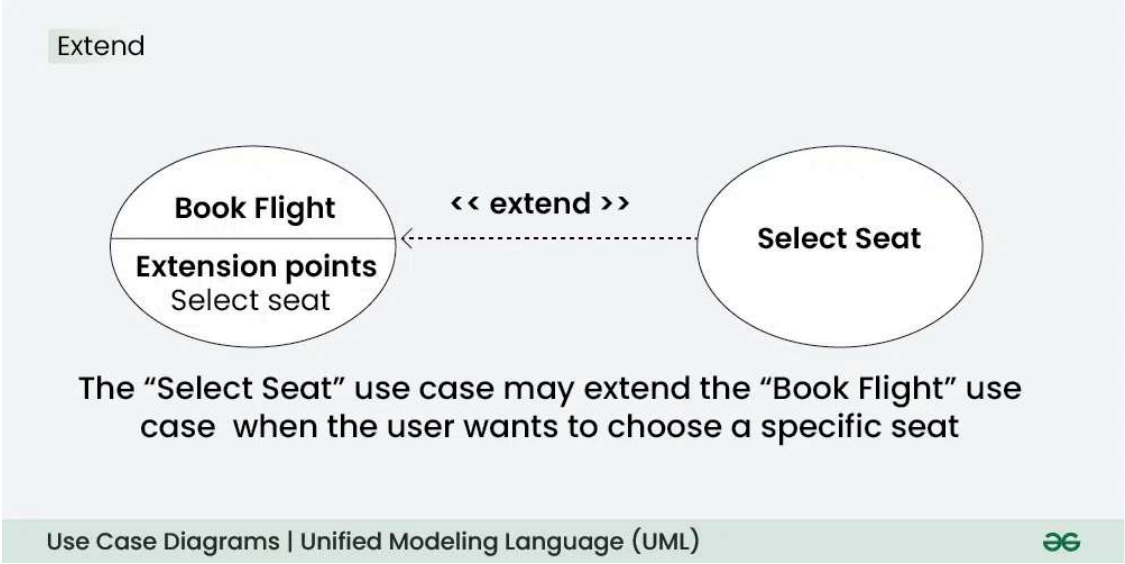


3.3. Extend Relationship

The Extend Relationship illustrates that a use case can be extended by another use case under specific conditions. It is represented by a dashed arrow with the keyword “extend.” This relationship is useful for handling optional or exceptional behavior.

Example: Flight Booking System

- **Use Cases:** Book Flight, Select Seat
- **Extend Relationship:** The “Select Seat” use case may extend the “Book Flight” use case when the user wants to choose a specific seat, but it is an optional step.

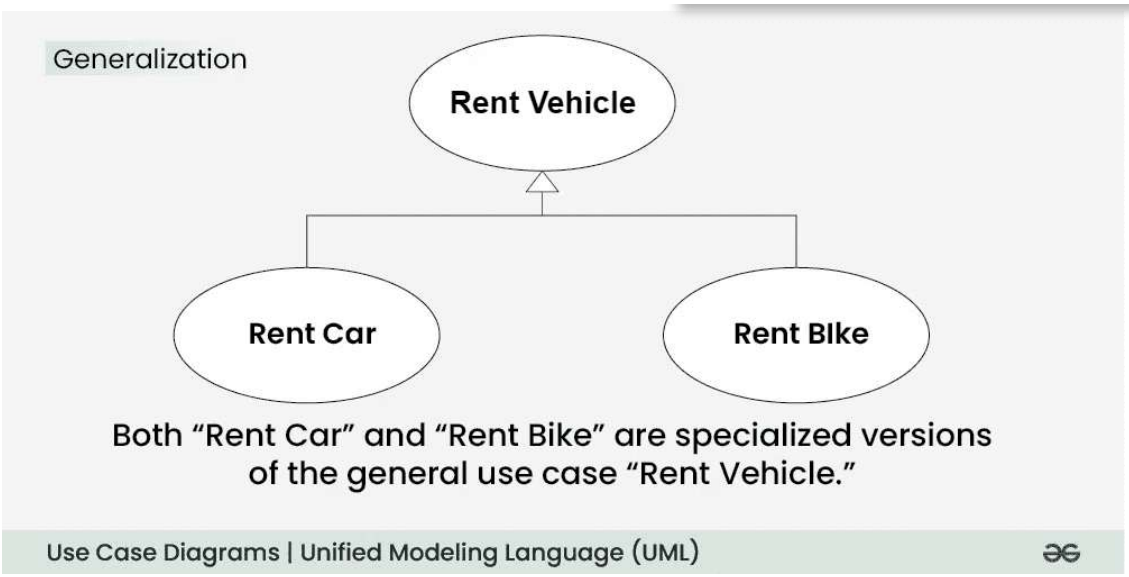


3.4. Generalization Relationship

The Generalization Relationship establishes an “is-a” connection between two use cases, indicating that one use case is a specialized version of another. It is represented by an arrow pointing from the specialized use case to the general use case.

Example: Vehicle Rental System

- **Use Cases:** Rent Car, Rent Bike
- **Generalization Relationship:** Both “Rent Car” and “Rent Bike” are specialized versions of the general use case “Rent Vehicle”



Generalization Relationship

4. How to draw a Use Case diagram in UML?

Step 1: Identify Actors

Determine who or what interacts with the system. These are your actors. They can be users, other systems, or external entities.

Step 2: Identify Use Cases

Identify the main functionalities or actions the system must perform. These are your use cases. Each use case should represent a specific piece of functionality.

Step 3: Connect Actors and Use Cases

Draw lines (associations) between actors and the use cases they are involved in. This represents the interactions between actors and the system.

Step 4: Add System Boundary

Draw a box around the actors and use cases to represent the system boundary. This defines the scope of your syste

Step 5: Define Relationships

If certain use cases are related or if one use ca
you can indicate these relationships with appr

Step 6: Review and Refine

Step back and review your diagram. Ensure that it accurately represents the interactions and relationships in your system. Refine as needed.

Step 7: Validate

Share your use case diagram with stakeholders and gather feedback. Ensure that it aligns with their understanding of the system's functionality.

Let's understand how to draw a Use Case diagram with the help of an Online Shopping System:

1. Actors:

- Customer
- Admin

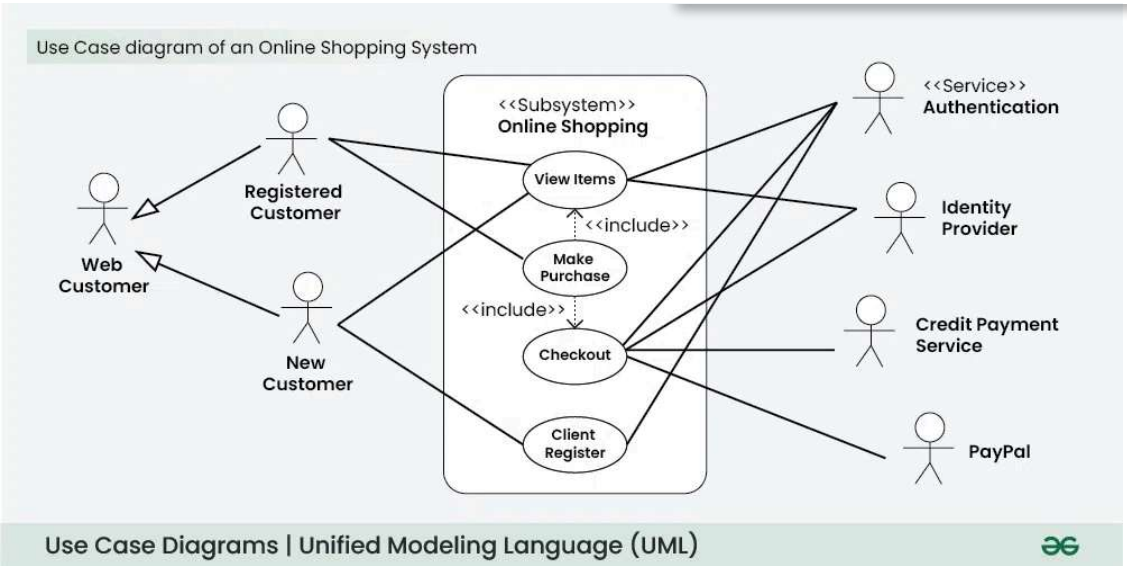
2. Use Cases:

1. Browse Products
2. Add to Cart
3. Checkout
4. Manage Inventory (Admin)

3. Relations:

- The Customer can browse products, add to the cart, and complete the checkout.
- The Admin can manage the inventory.

Below is the usecase diagram of an Online Shopping System:



5. What are common Use Case Diagram Tools and Platforms?

Several tools and platforms are available to create and design Use Case Diagrams. These tools offer features that simplify the diagram creation process, facilitate collaboration among team members, and enhance overall efficiency. Here are some popular Use Case Diagram tools and platforms:

6.1. Lucidchart

- Cloud-based collaborative platform.
- Intuitive drag-and-drop interface.
- Real-time collaboration and commenting.
- Templates for various diagram types.
- Integration with other tools like Jira and Confluence.

6.2. draw.io

- Free, open-source diagramming tool.
- Works offline and can be integrated with Google Drive, Dropbox, and others.
- Offers a wide range of diagram types, including Use Case Diagrams.
- Customizable shapes and themes.

6.3. Microsoft Visio

- Part of the Microsoft Office suite.
- Supports various diagram types, including U

- Integration with Microsoft 365 for collaborative editing.
- Extensive shape libraries and templates.

6.4. SmartDraw

- User-friendly diagramming tool.
- Templates for different types of diagrams, including Use Case Diagrams.
- Integration with Microsoft Office and Google Workspace.
- Auto-formatting and alignment features.

6.5. PlantUML

- Open-source tool for creating UML diagrams.
- Text-based syntax for diagram specification.
- Integrates with various text editors and IDEs.
- Supports collaborative work using version control systems.

6. What are Common Mistakes and Pitfalls while making Use Case Diagram?

Avoiding common mistakes ensures the accuracy and effectiveness of the Use Case Diagram. Here are key points for each mistake:

6.1. Overcomplication:

- **Mistake:** Including excessive detail in the diagram.
- **Impact:** Confuses stakeholders and complicates understanding.
- **Prevention:** Focus on essential use cases and maintain an appropriate level of abstraction.

6.3. Ambiguous Relationships:

- **Mistake:** Unclear relationships between actors and use cases.
- **Impact:** Causes misinterpretation of system interactions.
- **Prevention:** Clearly define and label relatio

6.3. Inconsistent Naming Conventions:

- **Mistake:** Inconsistent naming of actors and
- **Impact:** Causes confusion and hinders com

- **Prevention:** Establish and adhere to a consistent naming convention.

6.4. Misuse of Generalization:

- **Mistake:** Incorrect use of generalization relationships.
- **Impact:** Misrepresentation of the “is-a” relationship between use cases or actors.
- **Prevention:** Ensure accurate usage to represent specialization relationships.

6.5. Overlooking System Boundaries:

- **Mistake:** Not clearly defining the system boundary.
- **Impact:** Challenges understanding of the system’s scope.
- **Prevention:** Clearly enclose relevant actors and use cases within a system boundary.

6.6. Lack of Iteration:

- **Mistake:** Treating the diagram as a static artifact.
- **Impact:** May become outdated and not reflect the current state of the system.
- **Prevention:** Use an iterative approach, updating the diagram as the system evolves.

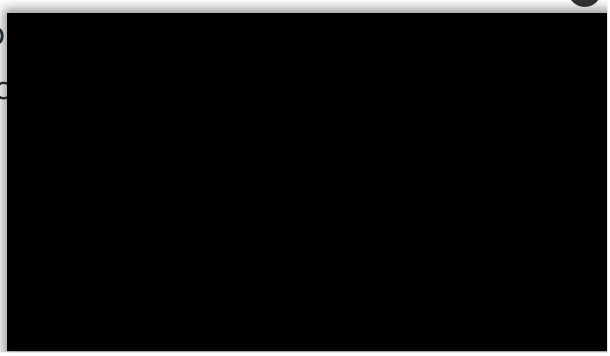
7. What can be Use Case Diagram Best Practices?

Creating effective and clear Use Case Diagrams is crucial for communicating system functionality and interactions. Here are some best practices to follow:

7.1 Keep it Simple:

- **Focus on High-Level Functionality:** Avoid unnecessary details and concentrate on representing the system’s p
- **Use Concise Language:** Use clear and conc
  actor names to enhance readability.

7.2 Consistency:



- **Naming Conventions:** Maintain a consistent naming convention for use cases and actors throughout the diagram. This promotes clarity and avoids confusion.
- **Formatting Consistency:** Keep a consistent format for elements like ovals (use cases), stick figures (actors), and lines to maintain a professional look.

7.3. Organize and Align:

- **Logical Grouping:** Organize use cases into logical groups to represent different modules or subsystems within the system.
- **Alignment:** Maintain proper alignment of elements to make the diagram visually appealing and easy to follow.

7.4. Use Proper Notation:

- **Consistent Symbols:** Adhere to standard symbols for actors (stick figures), use cases (ovals), and relationships to ensure understanding.
- **Proper Line Types:** Clearly distinguish between association, include, extend, and generalization relationships using appropriate line types.

7.5. Review and Iterate:

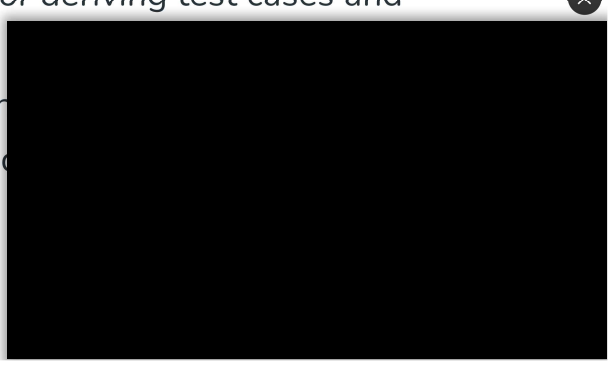
- **Feedback Loop:** Regularly review the diagram with stakeholders to ensure accuracy and completeness.
- **Iterative Process:** Use an iterative process, updating the diagram as the system evolves or more information becomes available.

By following these best practices, you can create Use Case Diagrams that effectively communicate the essential aspects of a system, fostering a shared understanding among stakeholders and facilitating the development process.

8. What are the Purpose and Benefits of Use Case Diagrams?

The Use Case Diagram offers numerous benef
development process. Here are some key adva
Diagrams:

- **Visualization of System Functionality:**
    - Use Case Diagrams provide a visual representation of the system's functionalities and interactions with external entities.
    - This visualization helps stakeholders, including non-technical ones, to understand the system's high-level behavior.
- **Communication:**
    - Use Case Diagrams serve as a powerful communication tool, facilitating discussions between stakeholders, developers, and designers.
    - They provide a common language for discussing system requirements, ensuring a shared understanding among diverse team members.
- **Requirement Analysis:**
    - During the requirements analysis phase, Use Case Diagrams help in identifying, clarifying, and documenting user requirements.
    - They capture the various ways users interact with the system, aiding in a comprehensive understanding of system functionality.
- **Focus on User Goals:**
    - Use Case Diagrams center around user goals and scenarios, emphasizing the perspective of external entities (actors).
    - This focus on user interactions ensures that the system is designed to meet user needs and expectations.
- **System Design:**
    - In the system design phase, Use Case Diagrams aid in designing how users (actors) will interact with the system.
    - They contribute to the planning of the user interface and help in organizing system functionalities.
- **Testing and Validation:**
    - Use Case Diagrams are valuable for deriving test cases and validating system behavior.
    - Testers can use the diagrams to er
      scenarios, including alternative an
      considered during testing.



9. Conclusion

In conclusion, a Use Case Diagram in UML serves as a powerful tool for capturing and visualizing the functional requirements and interactions within a system. By representing actors, use cases, and their relationships in a clear and concise manner, this diagram provides a high-level overview of the system's behavior.

srinam

Previous Article

Activity Diagrams | Unified Modeling Language (UML)

Next Article

Sequence Diagrams | Unified Modeling Language (UML)

Similar Reads

Class Diagrams vs Object Diagrams | Unified Modeling Language(UML)

UML class diagrams and object diagrams are the key tools for understanding the structure of a system, yet they serve distinct purposes. The differences...

3 min read

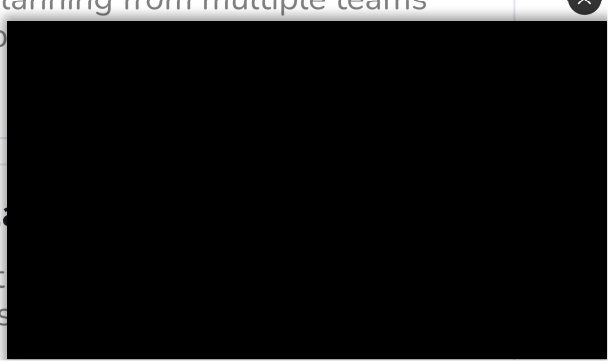
Behavioral Diagrams | Unified Modeling Language(UML)

Complex applications need collaboration and planning from multiple teams and hence require a clear and concise way to co

7 min read

State Machine Diagrams | Unified Modeling La

A State Machine Diagram is used to represent t
part of the system at finite instances of time. It's



7 min read

Activity Diagrams | Unified Modeling Language (UML)

Activity Diagrams are used to illustrate the flow of control in a system and refer to the steps involved in the execution of a use case. It is a type of...

9 min read

Sequence Diagrams | Unified Modeling Language (UML)

Unified Modelling Language (UML) is a modeling language in the field of software engineering that aims to set standard ways to visualize the design o...

10 min read

View More Articles

Article Tags : Geeks Premier League | System Design | Geeks Premier League 2023

GeeksforGeeks
Sanchhaya Education Private Limited
Corporate & Communications Address:- A-143, 9th Floor, Sovereign Corporate Tower, Sector- 136, Noida, Uttar Pradesh (201305) | Registered Address:- K 061, Tower K, Gulshan Vivante Apartment, Sector 137, Noida, Gautam Buddh Nagar, Uttar Pradesh, 201305