# Critical Thinking Assignment 3: Online Shopping System Use Case Diagram

Alexander Ricciardi

Colorado State University Global

CSC470: Software Engineering

Dr. Vanessa Cooper

January 5, 2025

## Critical Thinking Assignment 3: Online Shopping System Use Case Diagram

Unified Modeling Language (UML) Use-Case Diagrams are used early within the UML-base Development (Dolques et al, 2012). In Software Development (SD), they model the behavior of a system helping to capture the functional requirements of the system (IBM, 2023). They are structured around the concept of actors and use cases, the diagrams illustrate the interactions and relationships between them within a system, with the primary goal to describe the system's functionality (Helm n.d). What the system does. This essay explores an online shopping scenario by providing a UML Use-Case Diagram of a hypothetical online shopping system, analyzing its actors, use cases, and relationships, and providing two use case documentation examples describing flow and alternative flow steps.

## UML Use-Case Diagrams Overview

In the context of Software Engineering (SE), UML Use-Cases Digrams are part of the early stage of SD, and they play an important role in the software modeling of the Problem Space or Model of Process Space (MOPS). MOPS models "*what*" the business problem is and users are to understand and model the system or application requirements (Ricciardi, 2025). Use case modeling behind the identification of actors. An actor can be an individual or external/internal systems that interact with the system to fulfill a goal (Unhelkar, 2018 a). These are the main actors who benefit from the system—for example, The other components of a UML Use-Case Diagram are use cases and relationships. Use cases are illustrations of "*what*" a system does; however, they do not illustrate "*how*" the system does it. Relationships illustrate the interactions between actors, between use cases, and between actors and use cases. These relationships can be divided into four categories:

- Association relationships - illustrations of direct interactions between actors and use cases or between use cases.
- Include relationships - illustrations of interactions where use cases include the functionalities of other use cases.
- Extend relationships - illustrations of interactions where use cases are extended by other use cases under certain conditions.
- Generalization relationships - illustrations of interactions where use cases are generalized versions of other use cases or actors are generalized versions of other actors. This interaction can be defined as a parent-child relationship where the parent is the generalized version of a child.

(Ricciardi, 2025, p. 2)

On the next page sets Figure 2. It is a possible Use-Case Diagram for an online shopping scenario. It illustrates the actors, use cases, and their relationships within the online shopping system. The actors are depicted by a stick figure with a design starting with  "A" followed by a number and name description. Use cases are depicted as an eclipse container with a design starting with  "UC" followed by a number and name description. Relationships are depicted with lines and arrows with their name designation between "<< >>" if they represent an extend or include relationships, see Figure 1 for an illustration of them.

**Figure 1**

*Examples of Actors, Use Cases, and Relationships*

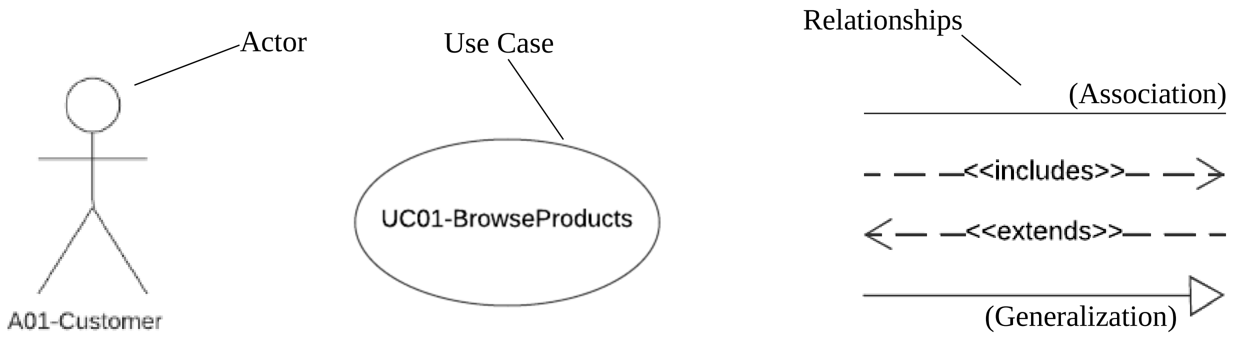
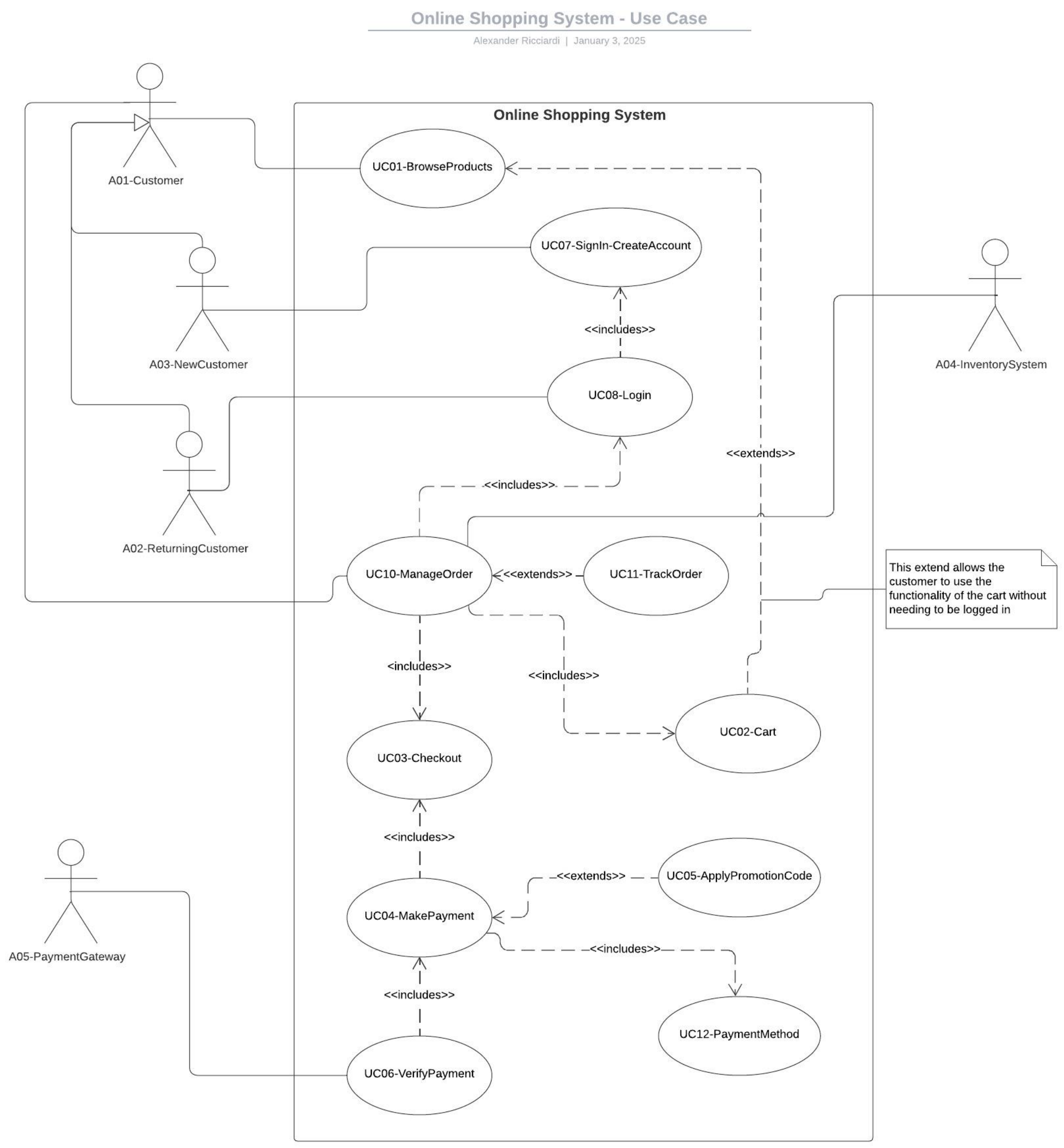


**Figure 2**

*Online Shopping System Use Case Diagram*



## Online Shopping System Analysis

Before crafting a USE-Case Diagram for the online shopping scenario, it is important to identify the actors.

> Use case modeling begins with the identification and documentation of users or actors" The main purpose of developing a software solution is to provide for the needs of these users. The actor also indicates how the system will be used (hence the term use cases). Actors provide the core starting point for the rest of modeling, design, and development in a software project.

(Unhelkar, 2018 a, p. 73).

Actors can be internal or external to the system, primary actors are the ones for whom the system is built, while secondary actors support the system to help primary actors to achieve their goals (Unhelkar, 2018). Direct actors interact with the system directly through an interface, for example, while indirect actors do not interact with the system interface but may receive output/date/goods from the system. Finally, abstract actors are generalizations of actors that can model the behavior of other actors, while concrete actors inherit from abstract actors. See Table 1 for a description of the actors.

**Table 1**

*Online Shopping System Actors*

| Actor | Description | Type |
|---|---|---|
| **A01-Customer** | A generic customer who interacts with the system. | Primary, Abstract, Direct |
| **A02-ReturningCustomer** | A registered customer with an account. | Primary, Concrete, Direct |
| **A03-NewCustomer** | A new customer who is using the system but needs to create an account. | Primary, Concrete, Direct |

| | | |
|---|---|---|
| **A04-InventorySystem** | An external actor in the online shopping system but part of the online business. It provides inventory information and updates. | Secondary, Concrete, Direct |
| **A05-PaymentGateway** | An external system that processes payments (e.g. credit card provider) | Secondary, Concrete, Direct |

After defining a list of potential actors, use cases can be identified (Unhelkar, 2018). The actor documentation can help identify use cases because it provides information on actor-to-use-case relationships. See Table 2 for a description of the online shopping system use cases.

**Table 2**

*Online Shopping System Use Cases*

| Use Case | Description |
|---|---|
| **UC01-BrowseProducts** | Allows customers to browse the available products. |
| **UC02-Cart** | Enables customers to add, remove, and manage items in their shopping cart. |
| **UC03-Checkout** | Enable the customer to finalize orders (reviewing items/prizes/quantity) |
| **UC04-MakePayment** | Handles the customer payment type and  promotion codes, if any are provided |
| **UC05-ApplyPromotionCode** | (Extends UC04) Allows the implementation of a discount code during payment. |
| **UC06-VerifyPayment** | Validates the payment with the A05-PaymentGateway. |
| **UC07-SignIn-CreateAccount** | Allows new customers to sign in by creating an account. |
| **UC08-Login** | (Included in UC07) implements authentication processes for registered customers (having an account). |
| **UC10-ManageOrder** | Allows customers to view and manage their orders. |
| **UC11-TrackOrder** | (Extends UC10) Allows customers to track order status. Potentially interacts with A04-InventorySystem for updates. |
| **UC12-PaymentMethod** | Manages customer payment information and method of payment. |

Use case documentation can help identify relationships. There are three main types of relationships:

- Actor-to-actor is always a generalization relationship or inheritance relationship.
- Actor to use case, called an association, is the relationship between an actor and a use case, also called communication as it can represent a communication between the actor and the system.
- Use case to use case are relationships relationships between two use cases that are defined by three specific types: include, extend, and inherit (generalization) relationships. These relationships were defined in the UML Use-Case Diagrams Overview section of this paper.

(Unhelkar, 2018 b).

The table below, Table 3, describes the relationships within the online shopping system diagram.

**Table 3**

*Online Shopping System Relationships*

| Relationship Type | Source | Target | Description |
|---|---|---|---|
| **Association** | A01-Customer | UC01-BrowseProducts | The customer browses the catalog of products. |
| **Association** | A01-Customer | UC10-ManageOrder | The customer uses the manager order use case to order/pay/track its orders |
| **Association** | A03-NewCustomer | UC07-SignIn-CreateAccount | A new customer signs in, creating an account. |
| **Association** | A02-ReturningCustomer | UC08-Login | The returning customer logs in using its credentials. |
| **Association** | A04-InventorySystem | UC02-ManageOrder | The inventory system updates or confirms cart items for availability. |
| **Association** | A05-PaymentGateway | UC06-VerifyPayment | An external payment gateway authorizes or declines payment transactions. |
| **<<includes>>** | UC08-Login | UC07-SignIn-CreateAccount | To log in the customer needs to have an account. Being signed in the system. |

| | | | |
|---|---|---|---|
| **<<includes>>** | UC10-ManageOrder | UC08-Login | To manage orders the customer needs to be logged in. |
| **<<includes>>** | UC10-ManageOrder | UC02-Cart | The Manage Order use case relies on the items in the customer cart to manage the customer orders |
| **<<includes>>** | UC10-ManageOrder | UC03-Checkout | The Manage Order use case relies on the functionality of the checkout use case the checkout the customer items. |
| **<<includes>>** | UC06-VerifyPayment | UC04-MakePayment | To verify the customer payment the customer needs to choose to make a payment. |
| **<<includes>>** | UC04-MakePayment | UC03-Checkout | To make a payment the customer needs to checkout first. |
| **<<includes>>** | UC04-MakePayment | UC12-PaymentMethod | The customer needs to choose a payment method. |
| **<<extends>>** | UC02-Cart | UC01-BrowseProducts | While browsing products the customer can add items to the cart. Note the customer does not need to be logged in to add items to the cart |
| **<<extends>>** | UC11-TrackOrder | UC10-ManageOrder | The customer has the option to track their orders. |
| **<<extends>>** | UC05-ApplyPromotion Code | UC04-MakePayment | The customer has the option to apply promotion codes to their purchases. |

## Examples of Flows and Alternative Flows

In use case documentation, flows (sometimes called “Main Flows” or “Basic Flows”) describe the happy path or the ideal flow (ProcessMaker, 2024). They document the usual or expected sequence of steps while Alternative Flows document exceptions, error conditions, or optional paths. Below are two examples of use case documentation describing the flows (main and alternative) for key use cases:

**Example: UC03–Checkout**

Main (Basic) Flow

1. Review Cart Items
    - The system displays items, quantities, and prices from UC02-Cart.
2. Confirm Order Details
    - The Customer checks shipping address, billing address, and any taxes or fees.
3. Proceed to Payment

    - The system transitions to UC04-MakePayment, where the Customer will choose payment options.

Alternative Flows

- A1: Cart Empty
    1. If the cart is empty, the system notifies the Customer and redirects them to UC01-BrowseProducts.
- A2: Session Timeout
    1. If the session expires during checkout, the system prompts the Customer to log in again via UC08-Login or create a new account if needed.

**Example: UC04–MakePayment**

Main (Basic) Flow

1. Select PaymentMethod
    - The system includes UC12-PaymentMethod, prompting the Customer to choose or enter credit card/billing details.
2. Verify Payment
    - The system calls UC06-VerifyPayment, which sends the payment request to A05-PaymentGateway.
3. Complete Payment
    - On successful authorization, the system finalizes the order, linking back to UC10-ManageOrder or concluding the checkout process.

Alternative Flows

- A1: Promotional Code
    1. The Customer opts to use UC05-ApplyPromotionCode (<<extends>>).

2. If valid, the system recalculates the total; if invalid, it notifies the Customer.

- A2: Payment Declined
    1. The gateway returns a “declined” response.
    2. The system offers the Customer a chance to retry or choose a different payment method (UC12).

For the seek of paper length, only two key use case documentation have been provided. However, the diagram illustrates several more use cases that could have been documented to provide examples of Main Flow and Alternative Flow scenarios. It is important to notice that “Use-Case Diagrams do not show any flow or dependencies in the system. They only provide a high-level picture of the system and have no features to represent the sequence of actions and alternative actions” (Unhelkar, 2018 b, p. 105)

## Conclusion

UML Use-Case Diagrams are a powerful tool for modeling the functional requirements of a system. In Software Development (SD), they are used early within the modeling process to capture the behavior and the functionality of a system. The primary components of a Use-Case Diagram are actors, use cases, and the relationships between them. The main goal of the components is to describe what the system does. For instance, the Online Shopping System Use-Case Diagram illustrates how actors, use cases, and their relationships capture the functional requirements of an online shopping system, defining what the system does. Additionaly, the two use case documentation examples illustrate samples of Main Flows and Alternative Flows describing how actors use specific use cases. In essence, the UML Use-Case Diagram illustrates “*what*” the system does, while the use case documentation describes “*how*” users may interact with specific system functionalities.

## References

Dolques, X., Huchard, M., Nebut, C., & Reitz, P. (2012). Fixing Generalization Defects in UML Use Case Diagrams. *Fundamenta Informaticae*, *115*(4), 327–356. https://doi-org.csuglobal.idm.oclc.org/10.3233/fi-2012-658cv

Helm, J. (n.d.). Business use-case model. *Rational unified process*. Fall 2023 SWEN 5135 Configuration Management course. University of Houston at Clear Lake. https://sceweb.uhcl.edu/helm/RUP_Folder/RationalUnifiedProcess/process/modguide/md_bucm.htm

IBM documentation (2023, September 21). Use-case diagrams. *IBM Rational Software Architect documentation*. IBM. https://www.ibm.com/docs/en/rational-soft-arch/9.7.0?topic=diagrams-use-case

ProcessMaker. (2024, August 20). *What is a happy path?* ProcessMaker. https://www.processmaker.com/blog/what-is-a-happy-path/

Ricciardi, A. S. (2025, January 2). UML Diagrams: A Guide for Software Engineers - Level up Coding. *Medium*. https://medium.com/gitconnected/uml-diagrams-a-guide-for-software-engineers-71220ffb775f

Unhelkar, B. (2018 a). Chapter 5 — Use case models-4: Actors and use cases. *Software engineering with UML*. CRC Press. ISBN 9781138297432

Unhelkar, B. (2018 b). Chapter 6 — Use case models-2: Use case diagrams and requirements modeling. *Software engineering with UML*. CRC Press. ISBN 9781138297432