# Victor Ngetich

# CS 255

# 3/29/25

**Interpret the object model for the new online storefront by responding to the following prompts:**

o **What are the different functions of the online storefront? How are they represented in this type of model?**

The UML class diagram has a few different functions of the online storefront, each represented by distinct classes and their associated methods. The Customer class manages the customer accounts, with methods like register (), login (), and updateProfile(), enabling customers to register, login and modify their information. The User class controls the verification of users, with the verifyLogin() method used to authenticate users. Administrative product management is represented by the Administrator class, which uses the updateCatalog() method to modify the product listings. Shopping cart functionality is supported by the Shopping Cart class, which uses methods like addCartItem(), updateQuantity(), viewCartDetails(), and checkOut() to control customer shopping cart contents. The Order class handles order processing, by using the placeOrder() function. The Shipping Info class, manages shipping details by using the updateShippingInfo() method. Lastly, the Order Details class handles order details management and uses the calcPrice () method to calculate subtotals for items within an order. All these functionalities are encapsulated

within their respective classes, with attributes (data) and methods (functions) defining their behavior and interactions within the system.

**What are the different classes of “users” represented by this object model? What are the associations between these classes?**

The different classes of users of the UML Class Diagram include Customer who is the end user of the ecommerce system (the person who makes the purchase), The User class which represents a general consumer and Administrator who manages the systems product catalog.

The association between these classes includes.

A Customer is directly associated with placing an Order, symbolizing the customer- buying relationship. A User is involved with the system through their association with a Shopping Cart, symbolizing their ability to add, modify, and view products before payment. Lastly, an Administrator is associated with modifying the Catalog, symbolizing their role in maintaining and updating the product listings in the e-commerce system.

**How would the objects “use” their respective variables and functions?**

The Customer object, holding variables like customer name, address, and credit card info, uses functions like register () to open an account, login () to access it, and updateProfile() to modify personal data. The User object, having user ID, password, and logged in variables, uses the verifyLogin() function to check login. The Administrator object, having an admin name and email variables, uses the updateCatalog() function to change product catalogs. The Shopping Cart object, which holds cart ID, product ID, and quantity, uses addCartItem(), updateQuantity(), viewCartDetails(), and checkout() functions to manage the shopping activities. Order object, holding order ID, date, and customer details, calls the placeOrder() function to complete the order. Shipping Info object, holding shipping ID, type, and shipping cost, calls the updateShippingInfo() function to update shipping details. Lastly, the Order Details object, holding product ID, name, quantity, and unit price, calls the calcPrice() function to compute the subtotals.

**Does this object model capture all of Hamp Crafts’ desired functionality? Why or why not?**

The UML class diagram captures many of the most important features of an e-commerce system, such as customer handling, shopping carting, and ordering. But not all the potential requirements of Hamp Crafts are included. Customer ratings, discounting facilities, payment handling, order tracking, and handling returns are some of the facilities omitted. In addition, more advanced admin features such as customer account management or system reports are lacking. Thus, though the diagram addresses the overall outline, it can probably be improved upon meeting Hamp Crafts' intended functionality.

**The above diagram uses a solid diamond shape to represent a form of aggregation. What type of aggregation does this represent? What does it imply about the relationship between the classes? Why is a solid diamond the appropriate choice here?**

The solid diamond shape in the UML Diagram represents composition aggregation. This implies a whole-part relationship where the lifetime of the contained object depends on the lifetime of the containing object. That is, if the parent object (e.g., Customer or Shopping Cart) is removed, then the child object (e.g., Order or Order Details) is removed.

The solid diamond is appropriate because the entities contained (such as Order Details in an Order or Shipping Info with an Order) do not exist in themselves without the parent entity. An Order Details entity, for example, is irrelevant without its own Order, and a Shipping Info record is automatically linked with an Order. Therefore, composition is an accurate model of the dependency where the containing entity is integral to the overall entity.

**Finally, think about the two different models you’ve explored for Hamp Crafts’ systems: a process model and an object model. Then compare these models by responding to the following prompts:**

**o How well do you think a process model describes the system? What information does it make easier to understand? What aspects of the system are more difficult to understand or are not represented?**

A process model (e.g., a flowchart) excels at showing the flow of activity through the system, e.g., how a customer would order something or how an administrator would update the catalog. It is excellent for showing the sequence of things, decision points, and the overall workflow. This makes the "how" of the operations in the system easy to follow.

It struggles to represent data structures and relationships among data. It does not show the structure of the data of the system or how different parts of the system relate in data terms.

o **How well do you think an object model describes the system? What information does it make easier to understand? What aspects of the system are more difficult to understand or are not represented?**

An object model (like a UML class diagram) accurately represents the system structure, displaying the classes, attributes, and data entity relationships.

It describes the organization of the data, the relationships between different components of the system, and the techniques and characteristics associated with each. It is an easy matter to understand the "what" of the system's architecture. It is not so apparent in depicting the dynamic character of operations or occurrences. It's not as effective in visualizing the sequential steps in the system.