Assignment 03 Report

(ChatGPT Powered by gpt-3.5-turbo.)

1. UML Use Case Diagram

Q: ...(the case content) Identify actors and use cases for the system described above and show them on a UML Use Case Diagram.

A:

Sure, here's a UML Use Case Diagram for the Order Processing System:

+---------------+ | Customer | +---------------+ | | | | | |+-------------------+ +-----------------------+| Customer | | Customer || Representative |<---------------------| Web Interface || | | |+-------------------+ +-----------------------+ | | | | | | +---------------+ | Shipping | | Company | +---------------+

The actors in this system are:

1. Customer
2. Customer Representative
3. Web Interface
4. Shipping Company

The use cases in this system are:

1. Place Order
2. Check Order Status
3. Cancel Order
4. Request Catalog
5. Return Product (via phone only)
6. Register as a Customer
7. Authenticate as a Customer
8. Upgrade to Gold Customer
9. Receive Email Notification (for back order products)

The Use Case Diagram shows the relationships between the actors and the use cases in the system.

Q:

You should list which actor do these use case belong to

A:

Sure, here are the use cases and the actors they belong to in the Order Processing System:

1. Place Order - Customer
2. Check Order Status - Customer
3. Cancel Order - Customer
4. Request Catalog - Customer
5. Return Product (via phone only) - Customer Representative
6. Register as a Customer - Customer
7. Authenticate as a Customer - Customer
8. Upgrade to Gold Customer - Customer
9. Receive Email Notification (for back order products) - Gold Customer

Q:

But you said The actors in this system are:

Customer  
Customer Representative  
Web Interface  
Shipping Company

A:

You are correct. I apologize for the confusion in my previous response. Here is an updated list of use cases with their respective actors in the Order Processing System:

1. Place Order - Customer, Web Interface
2. Check Order Status - Customer, Web Interface
3. Cancel Order - Customer, Web Interface
4. Request Catalog - Customer, Web Interface
5. Return Product (via phone only) - Customer Representative
6. Register as a Customer - Customer
7. Authenticate as a Customer - Customer
8. Upgrade to Gold Customer - Customer
9. Receive Email Notification (for back order products) - Gold Customer, Web Interface

Q:

So where is Shipping Company? And what's Shipping Company doing?

A:

The Shipping Company is not directly involved in any of the use cases as an actor. Instead, it is a system component that receives orders from the Order Processing System and fulfills them by shipping the products to the customers.

In the context of the use cases, after an order is placed and approved, the Order Processing System forwards the order information to the Shipping Company for fulfillment. The Shipping Company then retrieves the products from inventory, packages them, and ships them to the customer's specified address. The Shipping Company's role in the system is more of a backend process rather than an actor that interacts with the system's users directly.

Q:

And why is Gold Customer not included in the actor? He appeared in the use cases you list

A:

You are correct, the Gold Customer should also be included as an actor in the system since it has a specific role in some of the use cases. The Gold Customer is a special type of customer that has additional rights and benefits, such as extended return period and bonus points.

Therefore, the updated list of actors in the Order Processing System is:

1. Customer
2. Customer Representative
3. Web Interface
4. Shipping Company
5. Gold Customer

Thank you for bringing this to my attention and apologies for any confusion caused.

Q:

So why don't you draw this Use Case Diagram again since it's changed a lot?

A:

Sure, here's the updated UML Use Case Diagram for the Order Processing System:

+---------------+ | Customer | +---------------+ | | | | | |+-----------------------+ +-----------------------+| Customer | | Customer || Representative |<----------------| Web Interface || | | |+-----------------------+ +-----------------------+ | | | | | | +---------------+ | Shipping | | Company | +---------------+

| | +---------------+ | Gold | | Customer | +---------------+

The actors in this system are:

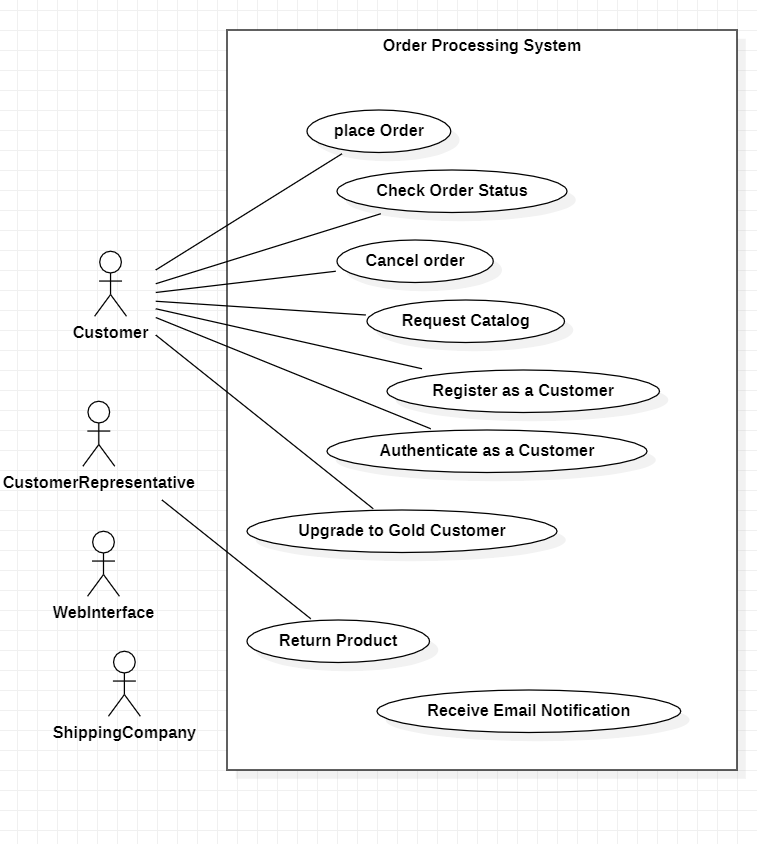
1. Customer
2. Customer Representative
3. Web Interface
4. Shipping Company
5. Gold Customer

The use cases in this system are:

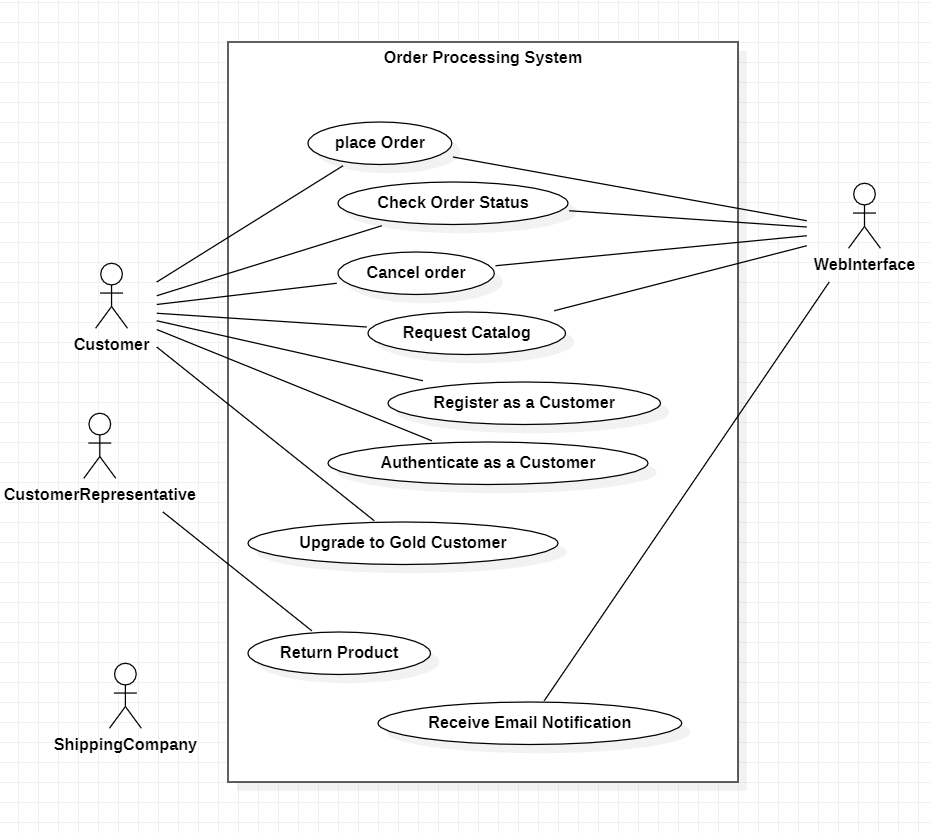
1. Place Order - Customer, Web Interface
2. Check Order Status - Customer, Web Interface
3. Cancel Order - Customer, Web Interface
4. Request Catalog - Customer, Web Interface
5. Return Product (via phone only) - Customer Representative
6. Register as a Customer - Customer
7. Authenticate as a Customer - Customer
8. Upgrade to Gold Customer - Customer
9. Receive Email Notification (for back order products) - Gold Customer, Web Interface

The Use Case Diagram shows the relationships between the actors and the use cases in the system.

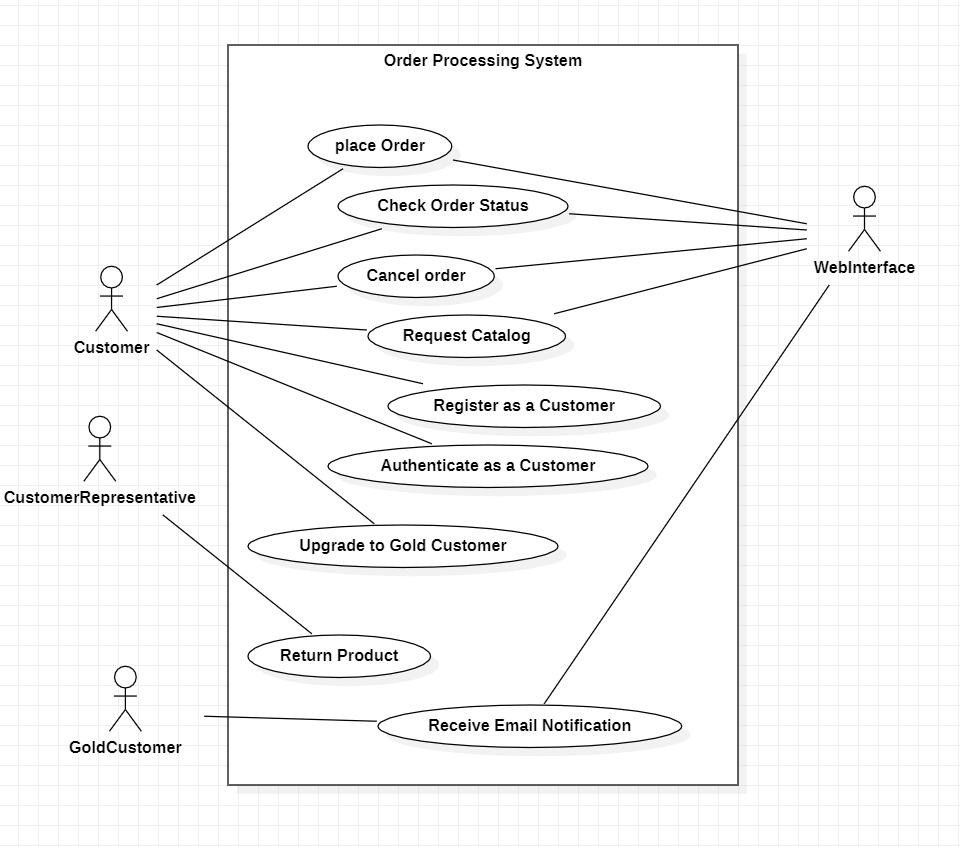
We can see that ChatGPT produced version 1:



Because there’s one use case’s actor is not included in its early analysis, and web interface has nothing to do. So here comes version 2:



But Gold Customer’s still not included in actors. So AI made another adjustment. By the way, AI explained that the Shipping Company is not directly involved in any of the use cases as an actor. Instead, it is a system component that receives orders from the Order Processing System and fulfills them by shipping the products to the customers. So it’s the final version:



1. Class Diagram

Q:

Perform a quick application domain analysis to come up with an object model for the above system. Express your findings with a UML Class Diagram, making sure to identify any critical operations of classes.  
Consider the following use case scenario (for use case “place order”):  
Ali is an existing customer of the order processing company described earlier, registered with their web site. Also assume that having browsed the printed catalogue he has, he already identified the two items (including their prices) he likes to buy from the company’s website using their product numbers (i.e. #2 and #9). First, he tries to buy one of product #2, but it is listed as unavailable in the inventory. Then, he adds two quantities of product #9, which turns out to be available, to his basket. He is then asked to confirm his registered shipping and billing addresses and credit card information from the customer database. He completes the order by clicking the Submit button. You may ignore processing of customer authentication.

A:

Based on the problem description, we can identify the following objects in the Order Processing System:

1. Customer
2. Customer Order
3. Product
4. Inventory
5. Shipping Address
6. Billing Address
7. Credit Card
8. Catalog
9. Order Status
10. Accounting System
11. Email Notification

These objects can be represented using a UML Class Diagram as follows:

+-----------------+ +----------------+ +------------+| Customer | | Customer | | Product |+-----------------+ | Order | +------------+| - customerID | +----------------+ | - productID || - name | | - orderID | | - name || - address | | - orderDate | | - price || - creditCard | | - orderStatus | | - quantity || | | - shippingAddr | | - status |+-----------------+ | - billingAddr | +------------+ | - productList | +----------------+ | +addProduct() | | +removeProduct()| | +calculateTotal()| | +submitOrder() | +----------------+

^ | +------------------+ | Inventory | +------------------+ | - productID | | - quantity | +------------------+ | +checkAvailability()| +------------------+

^ | +------------------+ | Shipping Address | +------------------+ | - streetAddress | | - city | | - state | | - zipCode | +------------------+

^ | +------------------+ | Billing Address| +------------------+ | - streetAddress | | - city | | - state | | - zipCode | +------------------+

^ | +-------------------+ | Credit Card | +-------------------+ | - cardNumber | | - expirationDate | | - cvv | +-------------------+

^ | +-------------------+ | Catalog | +-------------------+ | - productList | +-------------------+

^ | +-------------------+ | Order Status | +-------------------+ | - status | +-------------------+

^ | +-------------------+ | Accounting System | +-------------------+ | - invoice | +-------------------+

^ | +-------------------+ | Email Notification| +-------------------+ | - customerEmail | | - productID | | - notificationMsg | +-------------------+

The critical operations of some of the classes are:

Customer Order:

* addProduct(): adds a product to the customer's order.
* removeProduct(): removes a product from the customer's order.
* calculateTotal(): calculates the total cost of the order.
* submitOrder(): submits the order to the Order Processing System.

Product:

* checkAvailability(): checks if the product is available in the inventory.

Inventory:

* checkAvailability(): checks the availability of a product in the inventory.

Email Notification:

* sendNotification(): sends an email notification to a gold customer regarding a back order product becoming available.

For the use case scenario of placing an order, the following sequence of events can be identified:

1. The customer logs in to the web interface.
2. The customer searches for products using their product numbers.
3. The web interface retrieves the product information, including price and availability, from the Product and Inventory classes.
4. The customer adds the available products to their order using the Customer Order class.
5. The customer confirms their shipping and billing addresses and credit card information.
6. The Customer Order class calculates the total cost of the order.
7. The customer submits the order using the submitOrder() method of the Customer Order class.
8. The Order Processing System forwards the order to the Accounting System for invoicing and to the Shipping Company for fulfillment.

Q:

can you show me the relationship between these classes you draw? Like association, aggregation, generalization and so on

A:

Certainly! Here is an updated UML Class Diagram that includes the relationships and cardinalities between the classes:

+-----------------+ 1..\* +----------------+ +------------+| Customer |<>----------| Customer | | Product |+-----------------+ 1 | Order | +------------+| - customerID | +----------------+ | - productID || - name | | - orderID | | - name || - address | | - orderDate | | - price || - creditCard | | - orderStatus | | - quantity || | | - shippingAddr | | - status |+-----------------+ | - billingAddr | +------------+ | - productList | +----------------+ | +addProduct() | | +removeProduct()| | +calculateTotal()| | +submitOrder() | +----------------+

^ | +------------------+ | Inventory | +------------------+ | - productID | | - quantity | +------------------+ | +checkAvailability()| +------------------+

^ | +------------------+ | Shipping Address | +------------------+ | - streetAddress | | - city | | - state | | - zipCode | +------------------+

^ | +------------------+ | Billing Address| +------------------+ | - streetAddress | | - city | | - state | | - zipCode | +------------------+

^ | +-------------------+ | Credit Card | +-------------------+ | - cardNumber | | - expirationDate | | - cvv | +-------------------+

^ | +-------------------+ | Catalog | +-------------------+ | - productList | +-------------------+

^ | +-------------------+ | Order Status | +-------------------+ | - status | +-------------------+

^ | +-------------------+ | Accounting System | +-------------------+ | - invoice | +-------------------+

^ | +-------------------+ | Email Notification| +-------------------+ | - customerEmail | | - productID | | - notificationMsg | +-------------------+

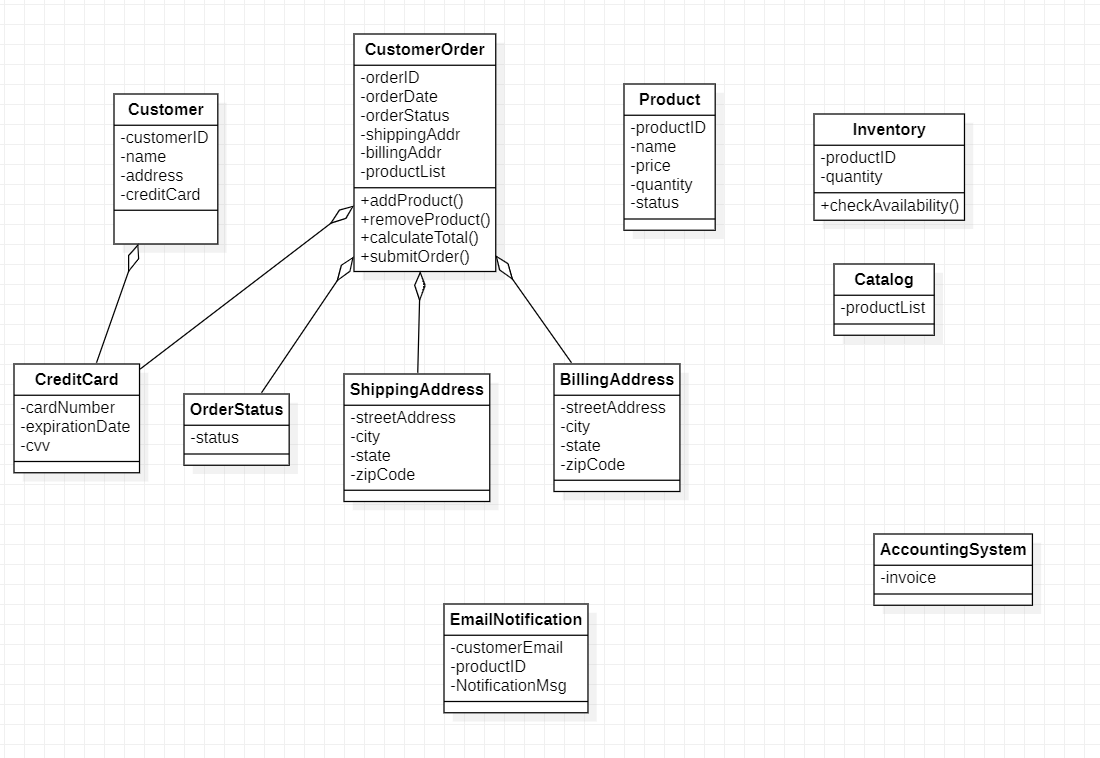
The relationships between the classes are as follows:

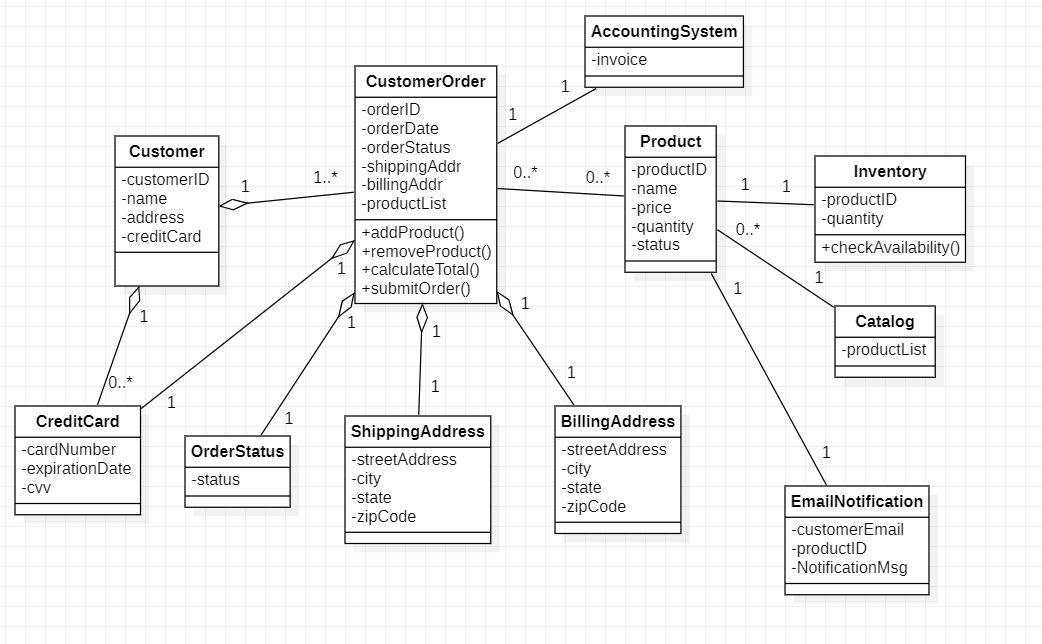
1. Customer and Customer Order: there is a one-to-many association between a Customer and their Customer Orders.
2. Customer Order and Product: there is a many-to-many association between a Customer Order and the Products they have ordered.
3. Product and Inventory: there is a one-to-one association between a Product and its inventory.
4. Customer Order and Shipping Address/Billing Address/Credit Card: there is a one-to-one association between a Customer Order and the Shipping Address, Billing Address, and Credit Card used for the order.
5. Catalog and Product: there is a one-to-many association between a Catalog and its Products.
6. Customer Order and Order Status: there is a one-to-one association between a Customer Order and its Order Status.
7. Customer Order and Accounting System: there is a one-to-one association between a Customer Order and the Accounting System.
8. Email Notification and Product: there is a one-to-one association between an Email Notification and the Product that the notification is related to.

Additionally, there are some composition relationships:

1. Customer Order and Shipping Address/Billing Address/Credit Card: a Customer Order is composed of a Shipping Address, Billing Address, and Credit Card.
2. Customer Order and Order Status: a Customer Order is composed of an Order Status.
3. Customer Order and Accounting System: a Customer Order is composed of an invoice from the Accounting System.

So version 1 like this(because I don’t know the relationship between some classes):

But after I asked ChatGPT for this, it became much clearer:



1. Sequence Diagram

Q:

draw a UML Sequence Diagram for this particular scenario.

A:

Sure! Here is a UML Sequence Diagram for the use case scenario of placing an order:

gherkin

Copy

+--------------------------+ +----------------------+ +-------------------+| Customer | | Customer Order | | Product |+--------------------------+ +----------------------+ +-------------------+| | | - productList | | || | | | | || | | | | || | | | | || searchForProduct() | | | | ||------------------------>| | | | || | | | | || | | | | || | | | | checkAvailability()|| selectProduct() | | | |------------------>||------------------------>| | | | || | | | | || | | | | || confirmOrder() | | | | ||------------------------>| | calculateTotal() | | || | |----------------------->| | || | | | | || | | | | || | | | | || submitOrder() | | | | ||------------------------>| | | | || | | | | || | | | | || | | | | || | | | | || | | | | || | | | | || | | | | || | | | | || | | | | || displayOrder() | | | | ||<------------------------| | | | |+--------------------------+ +----------------------+ +-------------------+

In this sequence diagram, the steps involved in the use case scenario are shown in order from top to bottom. The arrows indicate the direction of the message flow between the objects.

The sequence diagram starts with the Customer searching for a product by calling the searchForProduct() method. The Customer then selects a product by calling the selectProduct() method. The Product object responds by calling the checkAvailability() method to check if the product is available in the inventory.

Once the Customer has added all desired products to their order, they call the confirmOrder() method to confirm their order details. The Customer Order object then calculates the total cost of the order by calling the calculateTotal() method.

The Customer then submits the order by calling the submitOrder() method. This triggers a process that involves forwarding the order to the Accounting System and the Shipping Company for fulfillment.

Finally, the Customer calls the displayOrder() method to display the details of their order.

The result is like below:

