1. Use Case Diagram

Consider the following problem description: A mail-order company wants to automate its order processing. The initial version of the order processing system should be accessible to customers via the web. Customers can also call the company by phone and interact with the system via a customer representative. It is highly likely that the company will enhance this system in upcoming years with new features. The system allows customers to place orders, check the status of their orders, cancel an existing order and request a catalog. Customers may also return a product but this is only possible through the phone, not available on the web. When placing an order, the customer identifies himself by means of customer number (only for existing registered customers) or by means of his name and address. He then selects a number of products by giving the product number or by selecting products from the online catalogue. For each product, information such as price, a description and a picture (only on demand as they are usually high-resolution images of large size) are presented to the customer. Also, the availability of the product is obtained from the inventory. The customer indicates whether he wants to buy the product and in what quantity. When all desired products have been selected, the customer provides a shipping address and a credit card number and a billing address (if different from the shipping address). Then an overview of the ordered products and the total cost are presented. If the customer approves, the order is submitted. Credit card number, billing address and a specification of the cost of the order are used on the invoice, which is forwarded to the accounting system (an existing software module). Orders are forwarded to the shipping company, where they are filled and shipped. Customers who spent over a certain amount within the past year are promoted to be gold customers. Gold customers have additional rights such as being able to return products in an extended time period as well as earning more bonus points with each purchase. In addition, in cases where a product is on back order, gold customers have the option to sign up for an email notification for when the particular product becomes available. Identify actors and use cases for the system described above and show them on a UML Use Case Diagram.

Based on the problem description, we can identify the following actors and use cases for the system:

Actors:

1. Customer: Interacts with the system to place orders, check order status, cancel orders, request a catalog, and return products via phone.

2. Customer Representative: Represents the company and assists customers who interact with the system via phone.

Use Cases:

1. Place Order: Allows customers to select and purchase products, provide shipping and billing information, and submit the order.

2. Check Order Status: Enables customers to check the status of their orders.

3. Cancel Order: Allows customers to cancel an existing order.

4. Request Catalog: Allows customers to request a catalog.

5. Return Product (Phone Only): Allows customers to initiate a product return via phone.

6. Promote to Gold Customer: Automatically promotes customers who spent over a certain amount within the past year to gold customers.

7. Extend Return Period (Gold Customer): Allows gold customers to return products within an extended time period.

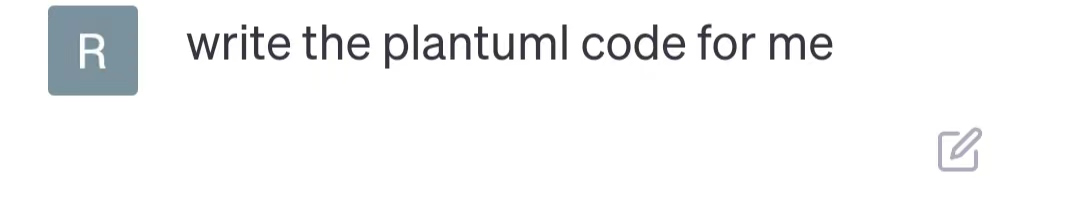
8. Earn Bonus Points (Gold Customer): Allows gold customers to earn bonus points with each purchase.

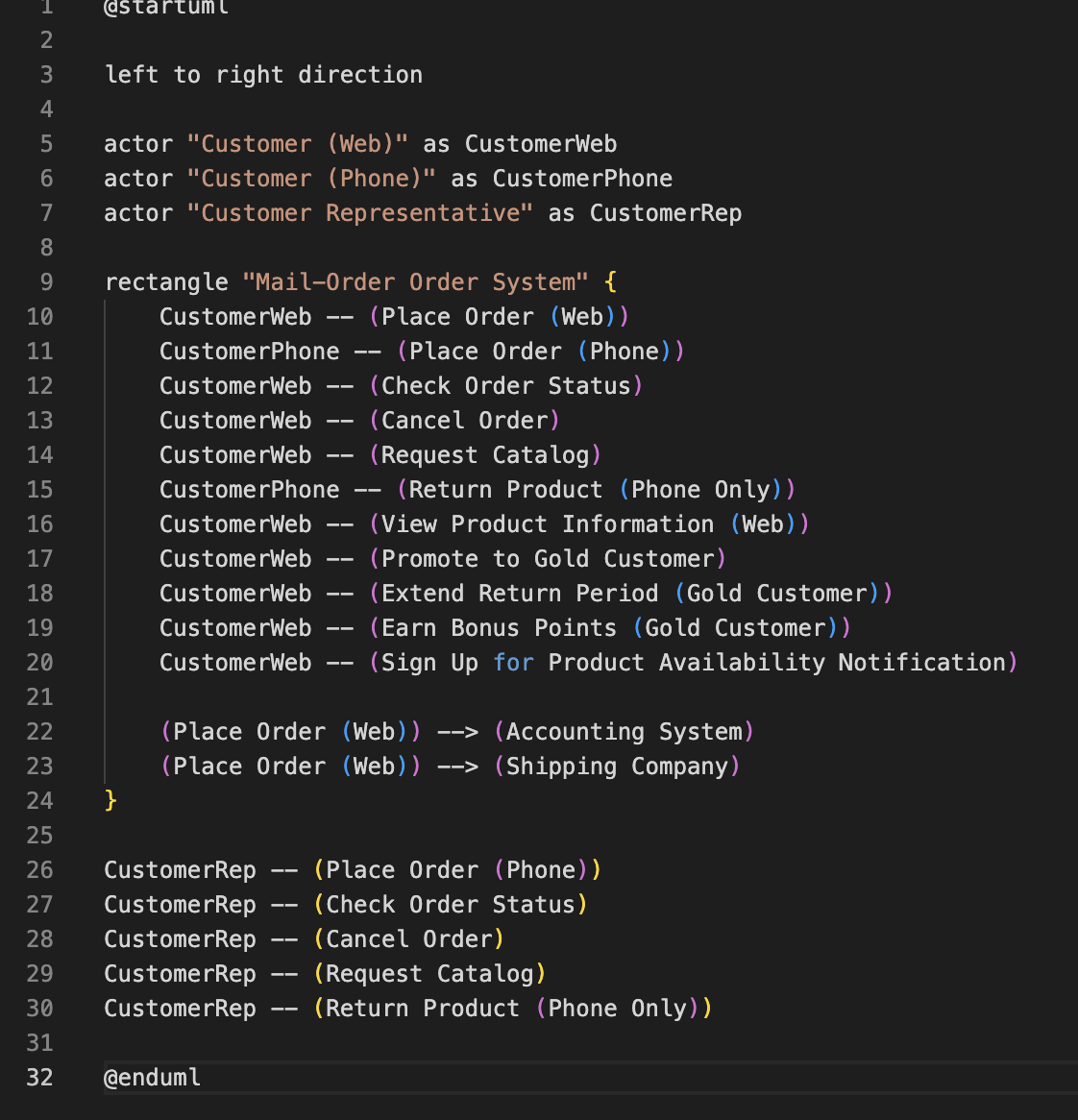
9. Sign Up for Product Availability Notification (Gold Customer): Enables gold customers to sign up for email notifications when back-ordered products become available.

Here's the UML Use Case Diagram representing the actors and use cases:

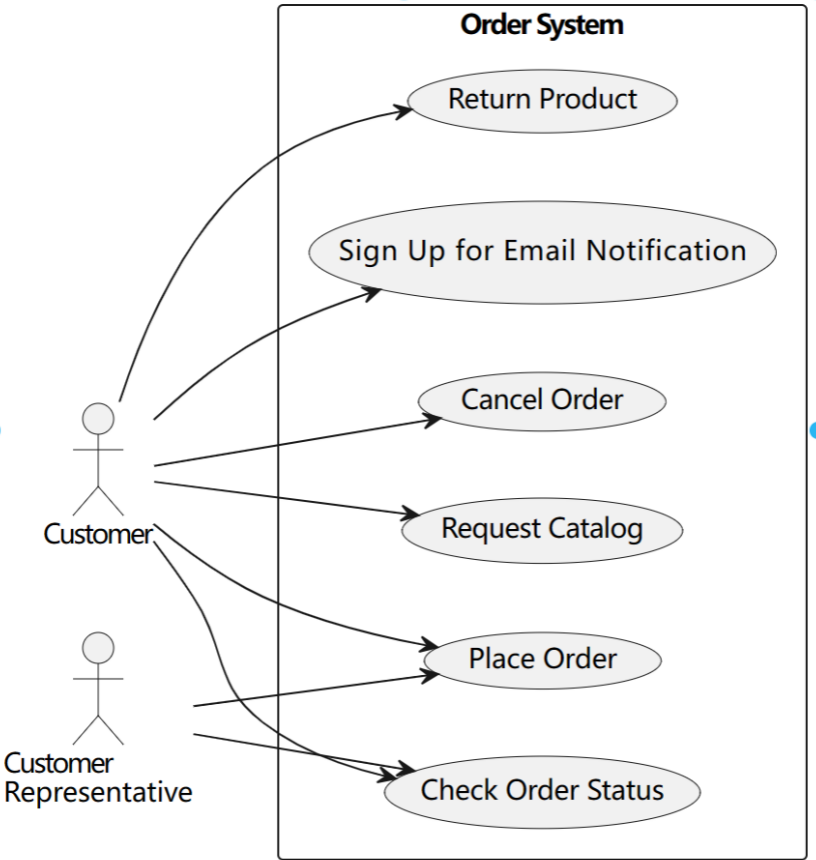


Note: The "Submit Order" and "Forward Invoice" use cases are not explicitly mentioned in the problem description but are implied as part of the order processing flow. The "Submit Order" use case represents the step where the order details are submitted, and the "Forward Invoice" use case represents the process of forwarding the invoice to the accounting system.





thus,we can get the use case diagram as follows:



2. Perform a quick application domain analysis to come up with an object model for the above system.

Based on the provided problem description, here's an initial object model for the order processing system:



In this object model:

- `Customer` represents a customer of the mail-order company with attributes like `customerId`, `name`, and `address`.

- `RegisteredCustomer` is a subclass of `Customer` and adds attributes like `username` and `password`.

- `Product` represents a product available for purchase, with attributes such as `productId`, `name`, `price`, `description`, and `picture`.

- `Inventory` manages the available products with an array or collection of `Product` objects.

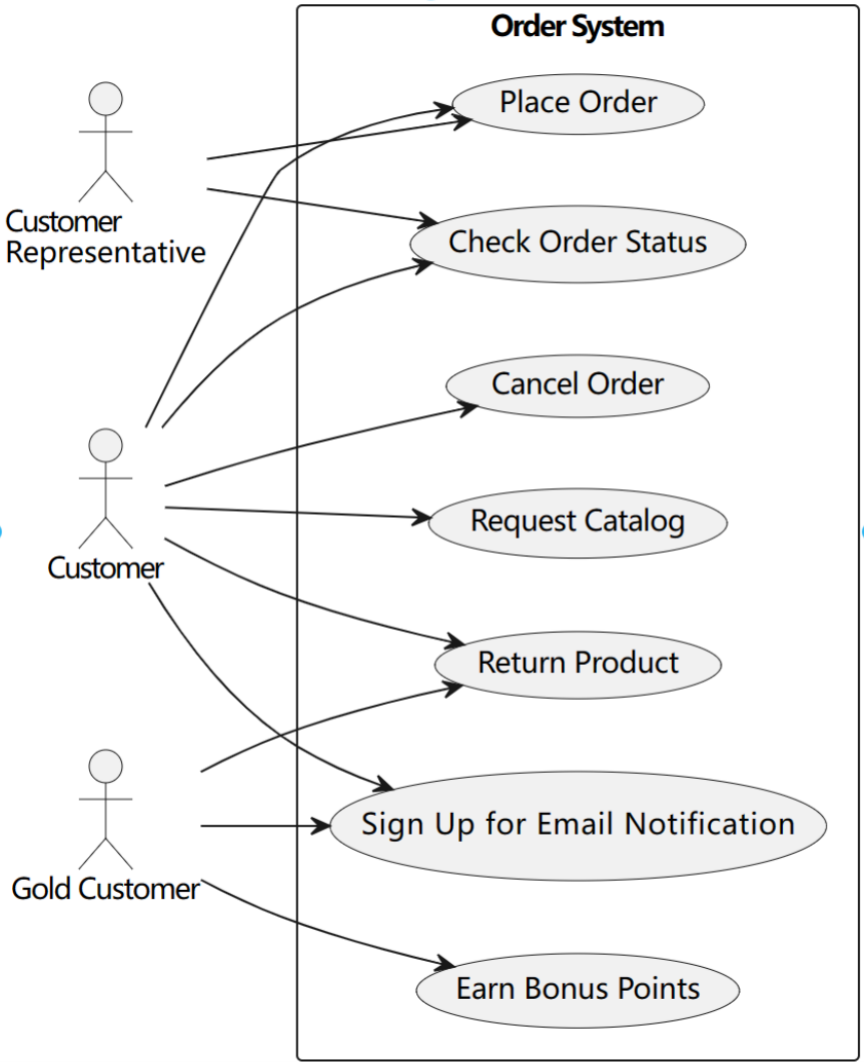
- `ShoppingCart` represents the shopping cart that holds selected products before placing an order, with an array or collection of `LineItem` objects.

- `LineItem` represents an individual item in the shopping cart, with a reference to the selected `Product` and the `quantity`.

- `Order` represents an order placed by a customer, with attributes like `orderId`, a reference to the `Customer`, an array or collection of `LineItem` objects, `shippingAddress`, `billingAddress`, and `creditCardNumber`.

This object model provides a basic representation of the main entities and relationships in the system. Additional classes and relationships may be required based on specific requirements and functionalities.

Then I improve it,I made the use case diagram:



For this part,I asked 3 questions totally,and the total time is 28 minutes.

1. Class Diagram:

Consider the following problem description: A mail-order company wants to automate its order processing. The initial version of the order processing system should be accessible to customers via the web. Customers can also call the company by phone and interact with the system via a customer representative. It is highly likely that the company will enhance this system in upcoming years with new features. The system allows customers to place orders, check the status of their orders, cancel an existing order and request a catalog. Customers may also return a product but this is only possible through the phone, not available on the web. When placing an order, the customer identifies himself by means of customer number (only for existing registered customers) or by means of his name and address. He then selects a number of products by giving the product number or by selecting products from the online catalogue. For each product, information such as price, a description and a picture (only on demand as they are usually high-resolution images of large size) are presented to the customer. Also, the availability of the product is obtained from the inventory. The customer indicates whether he wants to buy the product and in what quantity. When all desired products have been selected, the customer provides a shipping address and a credit card number and a billing address (if different from the shipping address). Then an overview of the ordered products and the total cost are presented. If the customer approves, the order is submitted. Credit card number, billing address and a specification of the cost of the order are used on the invoice, which is forwarded to the accounting system (an existing software module). Orders are forwarded to the shipping company, where they are filled and shipped. Customers who spent over a certain amount within the past year are promoted to be gold customers. Gold customers have additional rights such as being able to return products in an extended time period as well as earning more bonus points with each purchase. In addition, in cases where a product is on back order, gold customers have the option to sign up for an email notification for when the particular product becomes available.Express your findings with a UML Class Diagram, making sure to identify any critical operations of classes.

Based on the provided problem description, here's an updated UML class diagram representing the entities and relationships in the order processing system:

In this updated class diagram:



- `Customer` represents a customer of the mail-order company with attributes like `customerId`, `name`, and `address`.

- `RegisteredCustomer` is a subclass of `Customer` and adds attributes like `username` and `password`.

- `Product` represents a product available for purchase, with attributes such as `productId`, `name`, `price`, `description`, and `picture`.

- `Inventory` manages the available products with an array or collection of `Product` objects.

- `ShoppingCart` represents the shopping cart that holds selected products before placing an order, with an array or collection of `LineItem` objects.

- `LineItem` represents an individual item in the shopping cart, with a reference to the selected `Product` and the `quantity`.

- `Order` represents an order placed by a customer, with attributes like `orderId`, a reference to the `Customer`, an array or collection of `LineItem` objects, `shippingAddress`, `billingAddress`, and `creditCardNumber`.

Additionally, critical operations of the classes can include:

- `Customer`:

- `getId()`: Get the customer ID.

- `getName()`: Get the customer's name.

- `getAddress()`: Get the customer's address.

- `RegisteredCustomer` (inherits from `Customer`):

- `getUsername()`: Get the registered customer's username.

- `getPassword()`: Get the registered customer's password.

- `Product`:

- `getId()`: Get the product ID.

- `getName()`: Get the product's name.

- `getPrice()`: Get the product's price.

- `getDescription()`: Get the product's description.

- `getPicture()`: Get the product's picture.

- `Inventory`:

- `getAvailableProducts()`: Get the array/collection of available products.

- `ShoppingCart`:

- `getItems()`: Get the array/collection of items in the shopping cart.

- `addItem(item: LineItem)`: Add an item to the shopping cart.

- `removeItem(item: LineItem)`: Remove an item from the shopping cart.

- `Order`:

- `getId()`: Get the order ID.

- `getCustomer()`: Get the customer associated with the order.

- `getItems()`: Get the array/collection of items in the order.

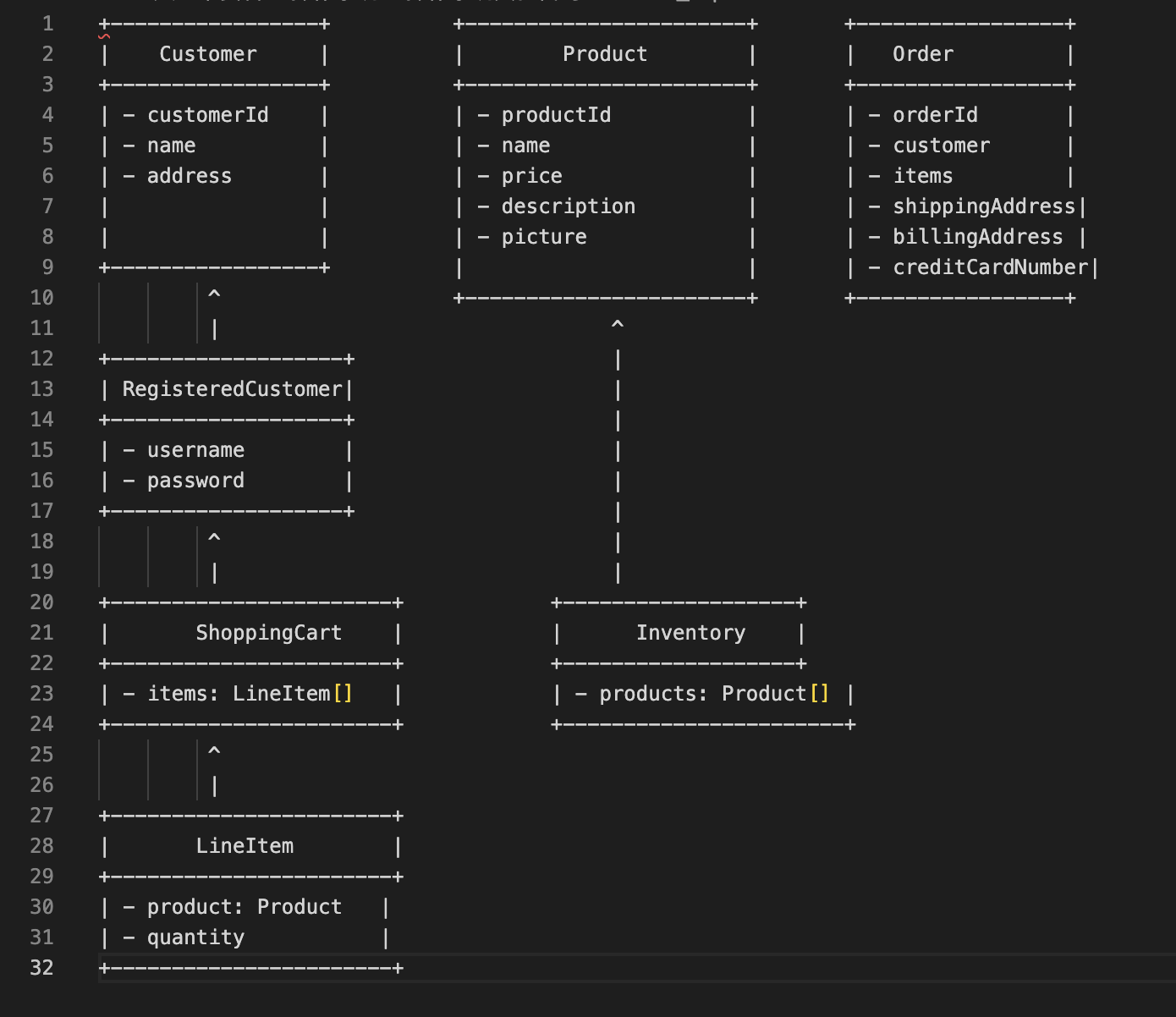
- `getShippingAddress()`: Get the shipping address for the order.

- `getBillingAddress()`: Get the billing address for the order.

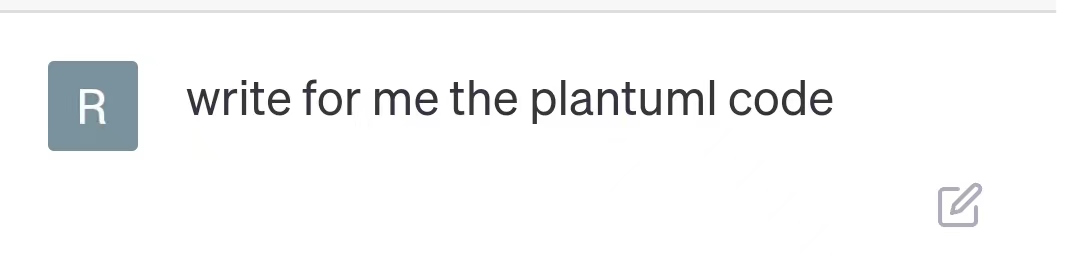
- `getCreditCardNumber()`: Get the credit card number for the order.

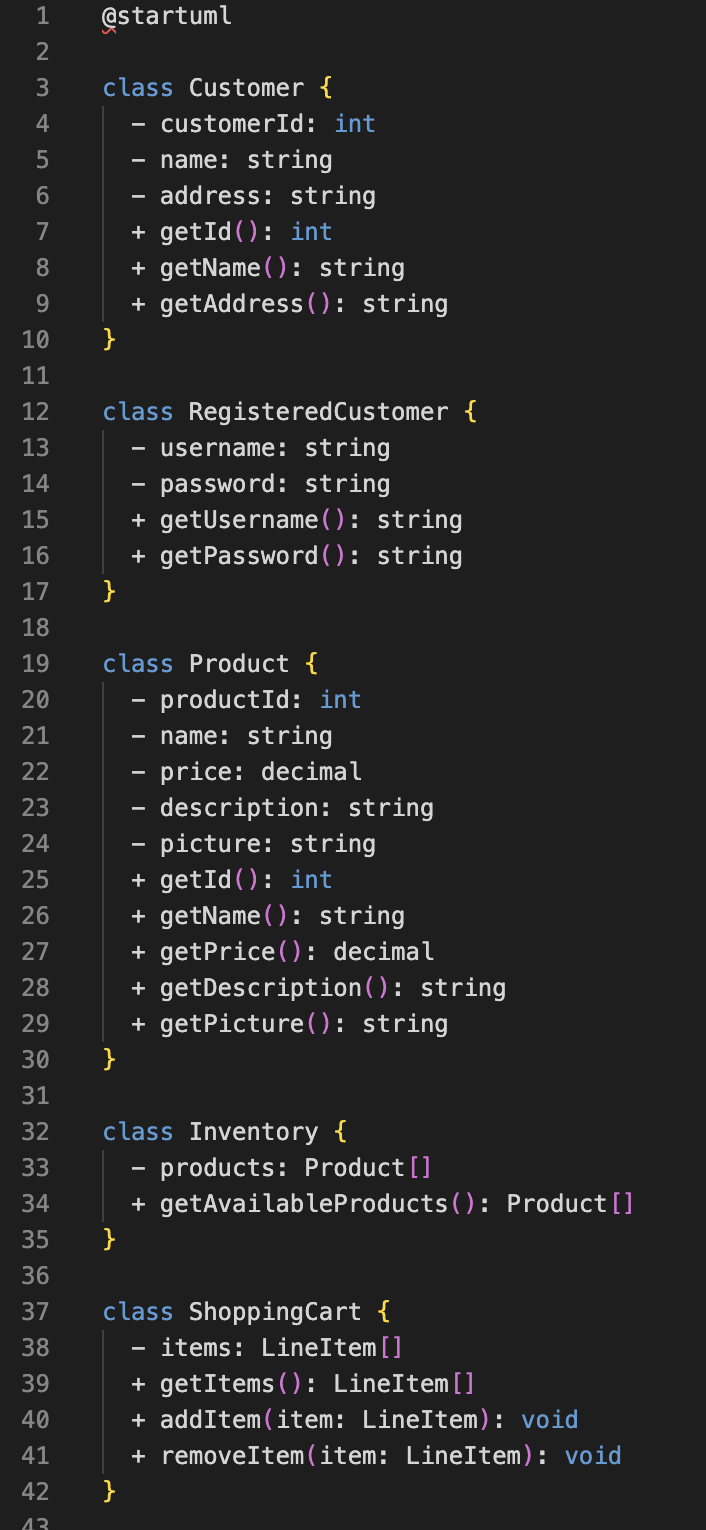
- `submitOrder()`: Submit the order for processing.

Now, let's add the critical operations to the class diagram:

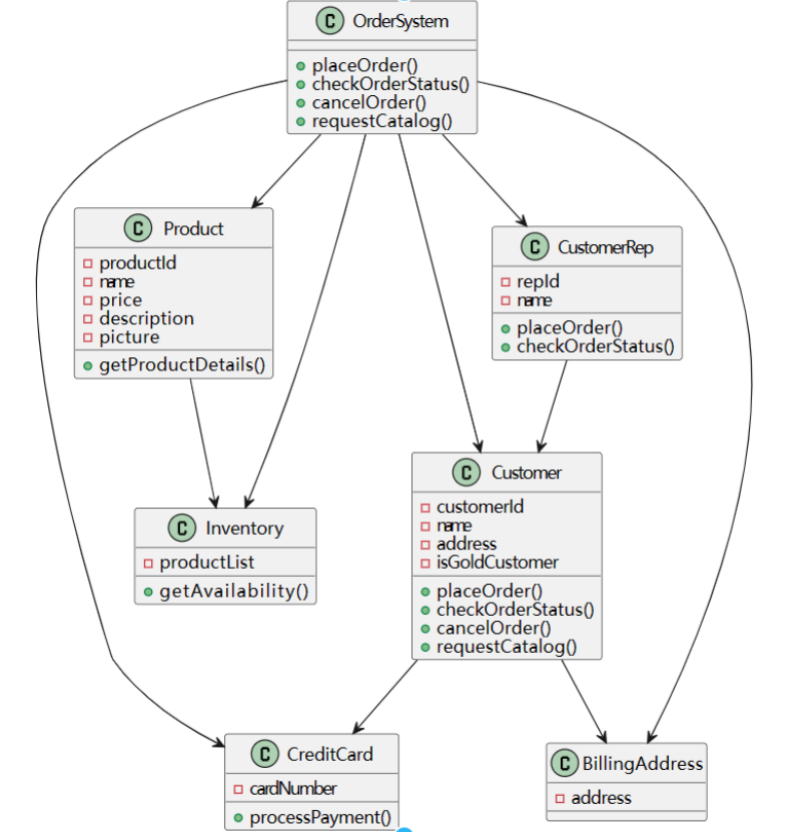


This updated class diagram includes the critical operations for each class, providing a clearer representation of the system's functionality.





Thus,the class diagram is as follows:

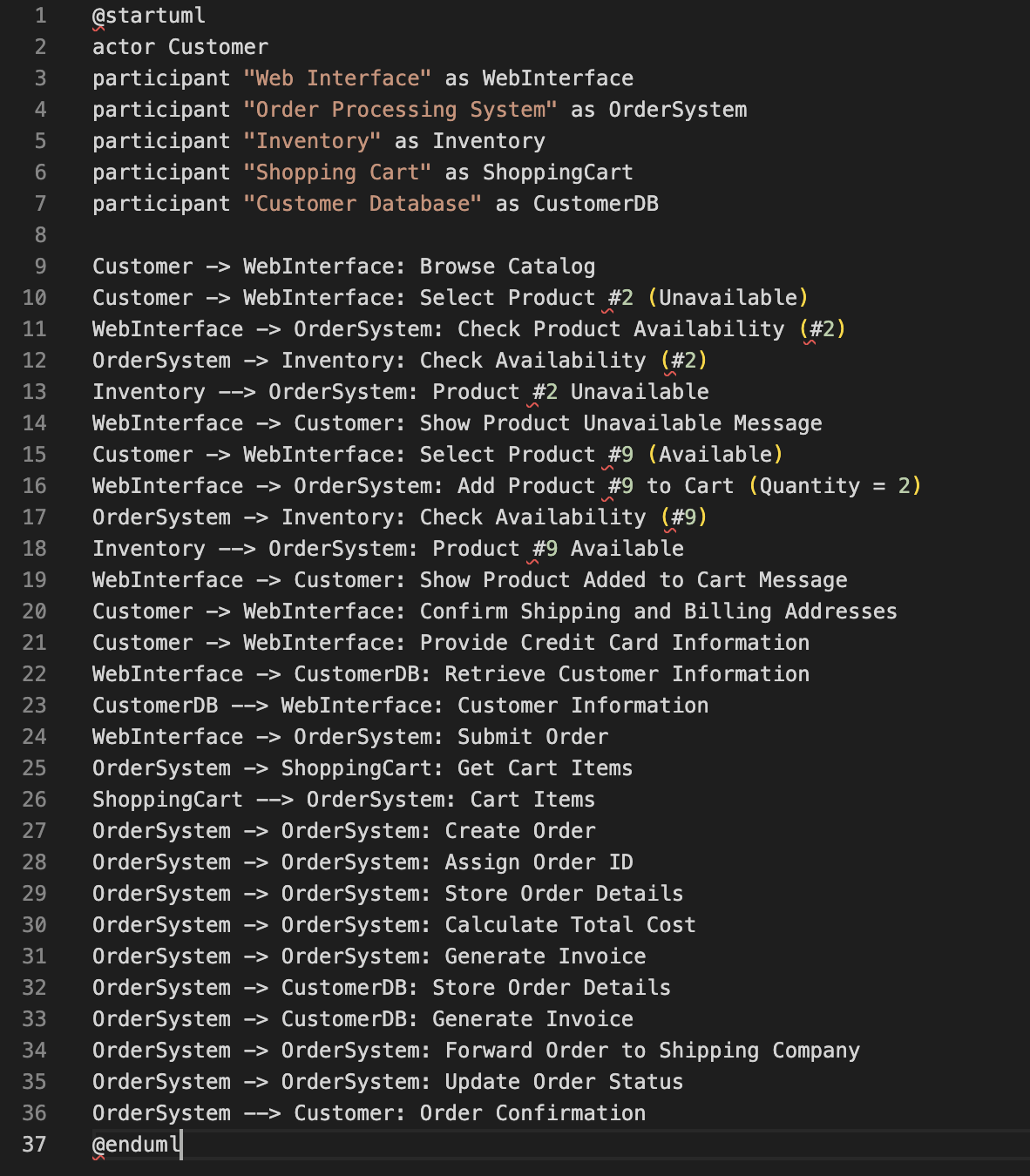


For this part,I asked 2 question totally,and the total time is 21minutes.

1. Sequence Diagram:

Consider the following use case scenario (for use case “place order”):

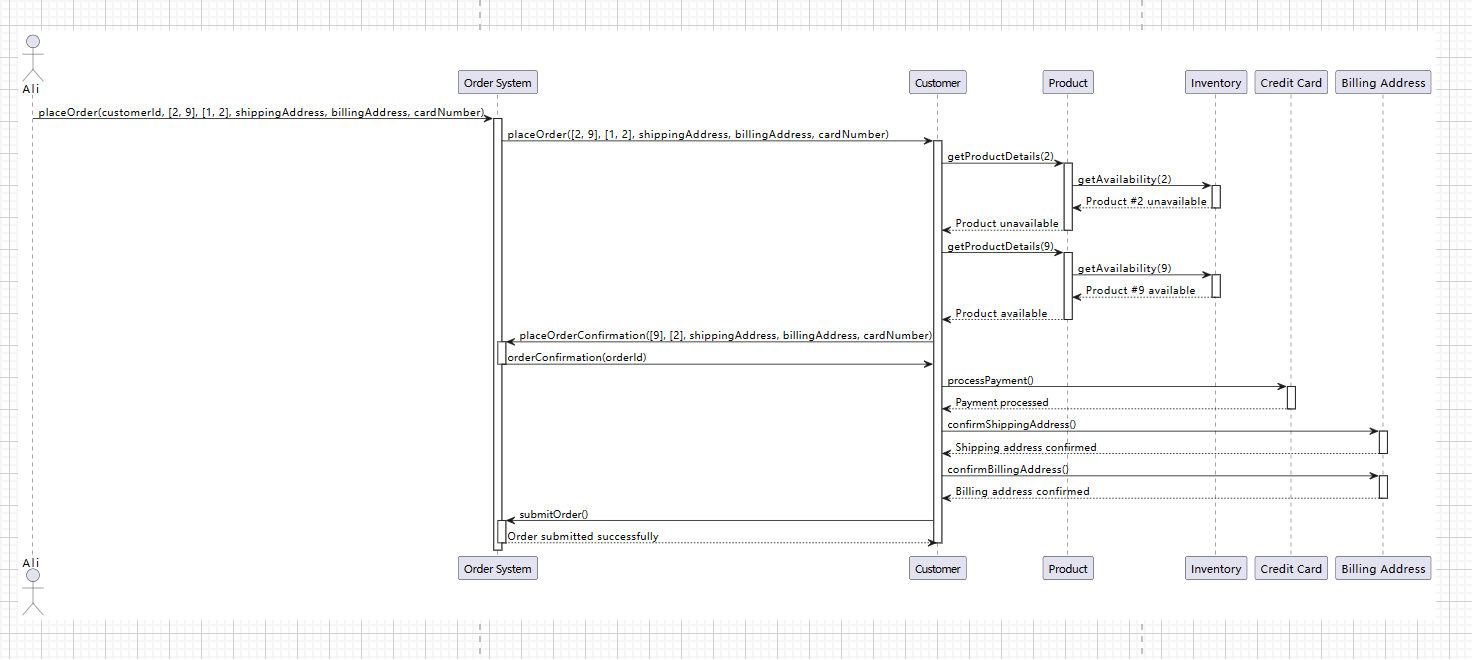
Ali is an existing customer of the order processing company described earlier, registered with their web site. Also assume that having browsed the printed catalogue he has, he already identified the two items (including their prices) he likes to buy from the company’s website using their product numbers (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.Draw a UML Sequence Diagram for this particular scenario. You may use any software/solution domain objects if needed as well.



In this sequence diagram, the interactions between the customer, web interface, order processing system, inventory, shopping cart, and customer database are shown. The steps include browsing the catalog, selecting products, checking availability, adding products to the cart, confirming addresses, providing credit card information, submitting the order, and generating an order confirmation.

Please note that the diagram assumes the existence of appropriate software/solution domain objects for handling the interactions and processes involved in the scenario.

Here is the sequence diagram:



For this part,I asked 1 question totally,and the total time is 13minutes.