**对话部分**

总计开了六次对话，每次问题都在3-6个，总计耗时大约一个小时（忘记每次具体计时了）

以下为和ChatGPT的对话内容（只选取了相对最有效的对话，不然太多了。至于有一些方框错位了，原对话中本来就错位了，就不再修改了（嫌麻烦））。

**ASK：**

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. (1). Identify use cases for the system described above and show them on a UML Use Case Diagram.Caution: Customer is the only actor in the UML Use Case Diagram.

**REPLY：**

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

Place Order: The customer can place an order by selecting products, specifying quantities, providing shipping and billing address, and making payment.

Check Order Status: The customer can check the status of their order to track its progress and estimated delivery date.

Cancel Order: The customer can cancel an existing order if it hasn't been shipped yet.

Request Catalog: The customer can request a catalog to be sent to their address.

Return Product (Phone Only): The customer can initiate a product return by contacting the company via phone.

Register as a New Customer: New customers can register by providing their name and address.

Sign In (Existing Customer): Existing registered customers can sign in to access their account and place orders.

Upgrade to Gold Customer: Customers who meet the spending criteria become gold customers, earning additional rights and benefits.

Receive Back Order Email Notification (Gold Customers Only): Gold customers have the option to sign up for email notifications when back-ordered products become available.

Below is a UML Use Case Diagram illustrating the identified use cases with the customer as the only actor:

+-------------------+

| Customer |

+-------------------+

| Use Cases |

v

+----------------------------------+

| Order Processing |

+----------------------------------+

| - Place Order |

| - Check Order Status |

| - Cancel Order |

| - Request Catalog |

| - Return Product (Phone Only) |

| - Register as a New Customer |

| - Sign In (Existing Customer) |

| - Upgrade to Gold Customer |

| - Receive Back Order Email Notification (Gold Customers Only) |

+----------------------------------+

Note: The diagram shows a direct association between the customer and the system (Order Processing), representing the interaction between the customer and the system through the defined use cases.

因为ChatGPT总是给出很多莫名其妙的Actor，所以在此限定只有Customer一个参与者。之后ChatGPT给出了相对靠谱的答案。

**ASK：**

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.

**REPLY：**

In order to perform a quick application domain analysis and create an object model, we need to identify the main concepts and their relationships within the system. Based on the problem description, the following classes and their critical operations can be identified:

Customer

Attributes: customerNumber, name, address

Operations:

placeOrder(): Allows the customer to place an order.

checkOrderStatus(): Allows the customer to check the status of their orders.

cancelOrder(orderId: int): Allows the customer to cancel a specific order.

requestCatalog(): Allows the customer to request a catalog.

Product

Attributes: productNumber, price, description

Operations: None (No critical operations mentioned in the problem description, but additional operations can be added as required).

Order

Attributes: orderId, customer, orderDate, shippingAddress, billingAddress, orderStatus

Operations:

addItem(product: Product, quantity: int): Adds a product with the given quantity to the order.

removeItem(product: Product): Removes a product from the order.

calculateTotalCost(): Calculates the total cost of the order.

submitOrder(): Submits the order for processing.

ShippingCompany

Operations:

fillOrder(order: Order): Fills the given order by preparing it for shipping.

AccountingSystem

Operations:

generateInvoice(order: Order): Generates an invoice for the given order and forwards it to the accounting system.

GoldCustomer (subclass of Customer)

Operations:

returnProduct(order: Order, product: Product): Initiates the return process for a product in the given order.

NotificationSystem

Operations:

notifyCustomer(customer: Customer, product: Product): Sends an email notification to the customer when a back-ordered product becomes available.

Here is a UML Class Diagram representing the object model:

+-----------------+

| Customer |

+-----------------+

| - customerNumber: int

| - name: string

| - address: string

+-----------------+

| + placeOrder()

| + checkOrderStatus()

| + cancelOrder(orderId: int)

| + requestCatalog()

+-----------------+

|

|

|

| +-----------------+

+-----------------+ | Product |

| Order |<>-------------------------------+-----------------+

+-----------------+ | - productNumber: int

| - orderId: int | | - price: float

| - customer: Customer | - description: string

| - orderDate: date +-----------------+

| - shippingAddress: string

| - billingAddress: string

| - orderStatus: string

+-----------------+

| + addItem(product: Product, quantity: int)

| + removeItem(product: Product)

| + calculateTotalCost(): float

| + submitOrder()

+-----------------+

|

|

|

|

+------------------------+ +------------------------+

| ShippingCompany | | AccountingSystem |

+------------------------+ +------------------------+

| + fillOrder(order: Order) | + generateInvoice(order: Order)

+------------------------+ +------------------------+

| |

| |

| |

| |

+-----------------+ +-----------------------+

| GoldCustomer | | NotificationSystem |

+-----------------+ +-----------------------+

+ returnProduct(order: Order, product: Product) | + notifyCustomer(customer: Customer, product: Product)

+-----------------------+

Note: The class diagram presented here represents the basic structure of the object model. Depending on the specific requirements and further details of the system, additional attributes and operations may need to be included.

因为不太会写问题，感觉问的几次都不如最开始的这个。（比如让ChatGPT明确各个类之间的聚合、组合、继承等之类的关系等）不过它还是给出了两个场景中并没有提及的类。

**ASK：**

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.

**REPLY：**

Here is a UML Sequence Diagram for the given use case scenario "Place Order" with the described steps:

+------------------------+

| Customer |

+------------------------+

|

|

|

+------------------------------+

| Order |

+------------------------------+

| |

| |

| addItem() |

| (Product #2, 1) |

|------------------------->|

| |

| checkAvailability() |

| (Product #2) |

|------------------------------------>|

| |

| returnAvailabilityStatus(false) |

|<------------------------------------|

| |

| addItem() |

| (Product #9, 2) |

|------------------------->|

| |

| checkAvailability() |

| (Product #9) |

|------------------------->|

| |

| returnAvailabilityStatus(true) |

|<------------------------------------|

| |

| getShippingAddress() |

|------------------------->|

| |

| getBillingAddress() |

|------------------------->|

| |

| getCreditCardInfo() |

|------------------------->|

| |

| submitOrder() |

|------------------------->|

| |

| |

| |

+------------------------+ +------------------------+

| Shipping Company | | Accounting System |

+------------------------+ +------------------------+

| |

| |

| fillOrder() |

|<-------------------------|

| |

| |

| generateInvoice() |

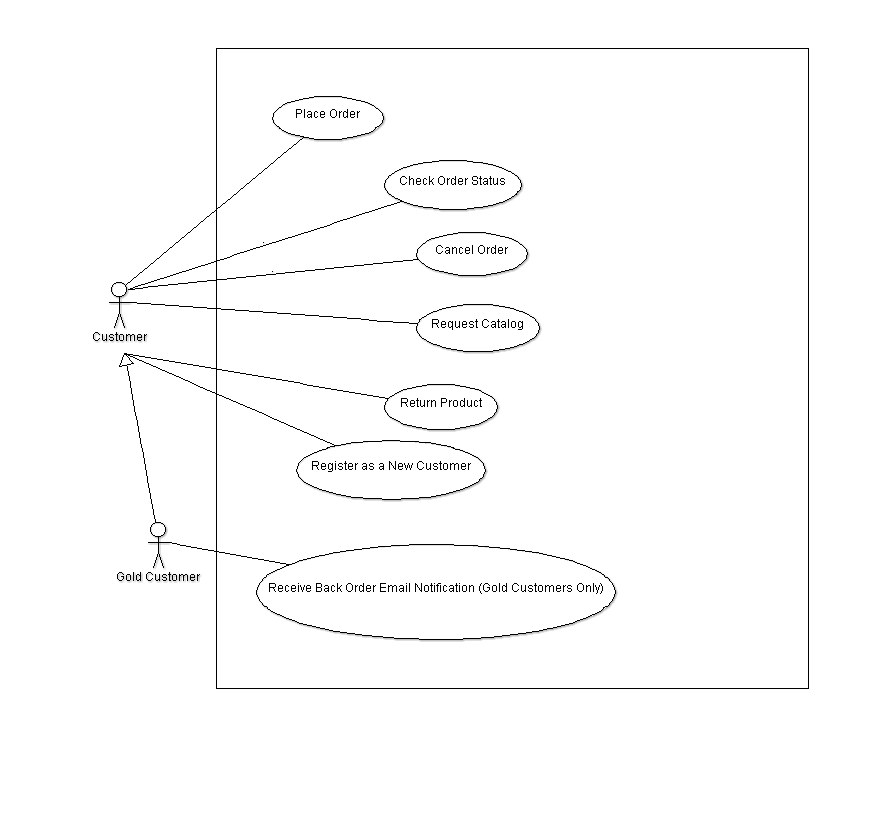
|<-------------------------|

| |

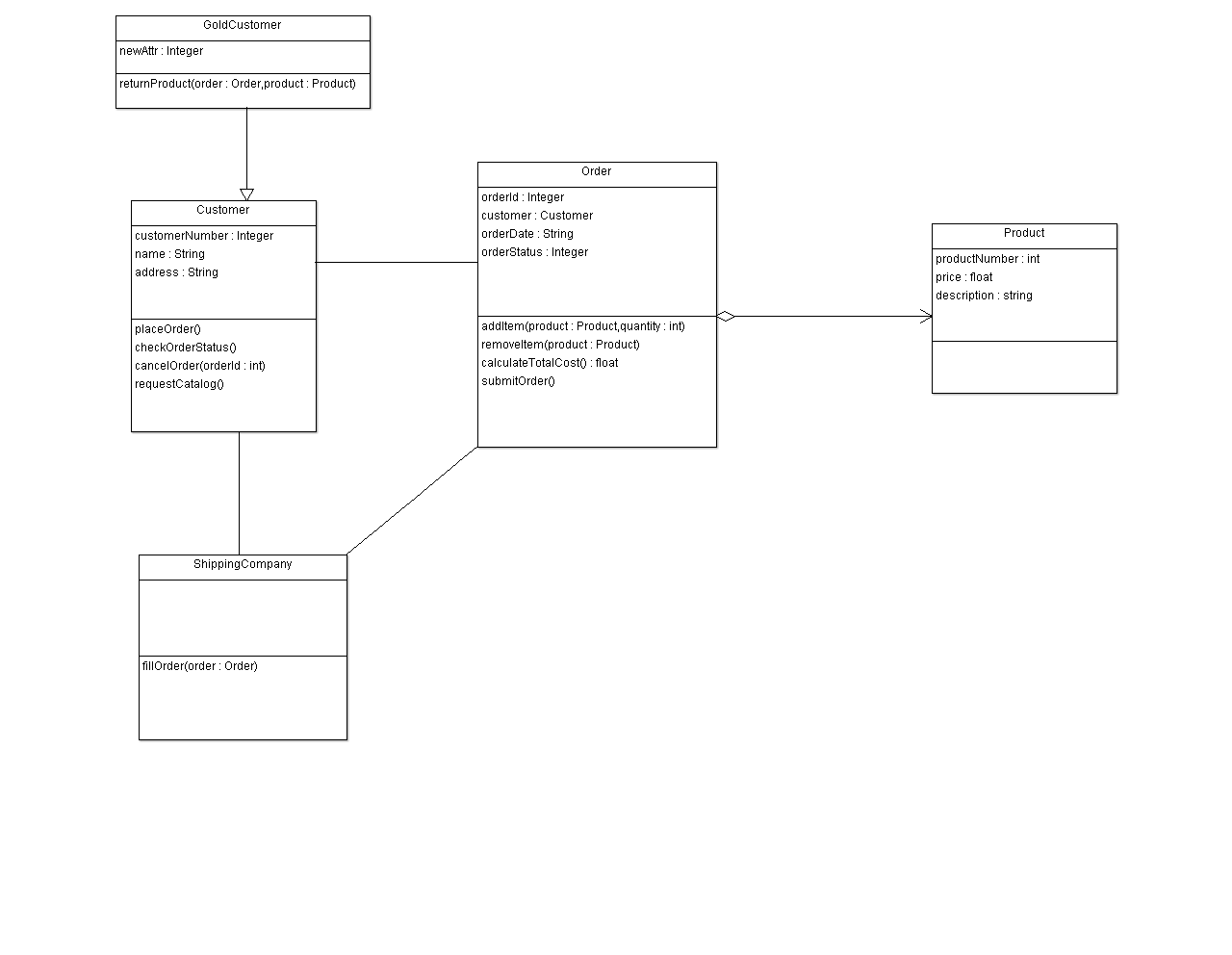
Note: This sequence diagram shows the interaction between the Customer, Order, Shipping Company, and Accounting System objects during the "Place Order" scenario. The steps of adding items to the order, checking availability, retrieving shipping and billing addresses, providing credit card information, and submitting the order are depicted. The interactions with the Shipping Company and Accounting System involve the fillOrder() and generateInvoice() operations, respectively.

顺序图其实挺正常的。不过它画了两个，感觉后一个没有什么必要。

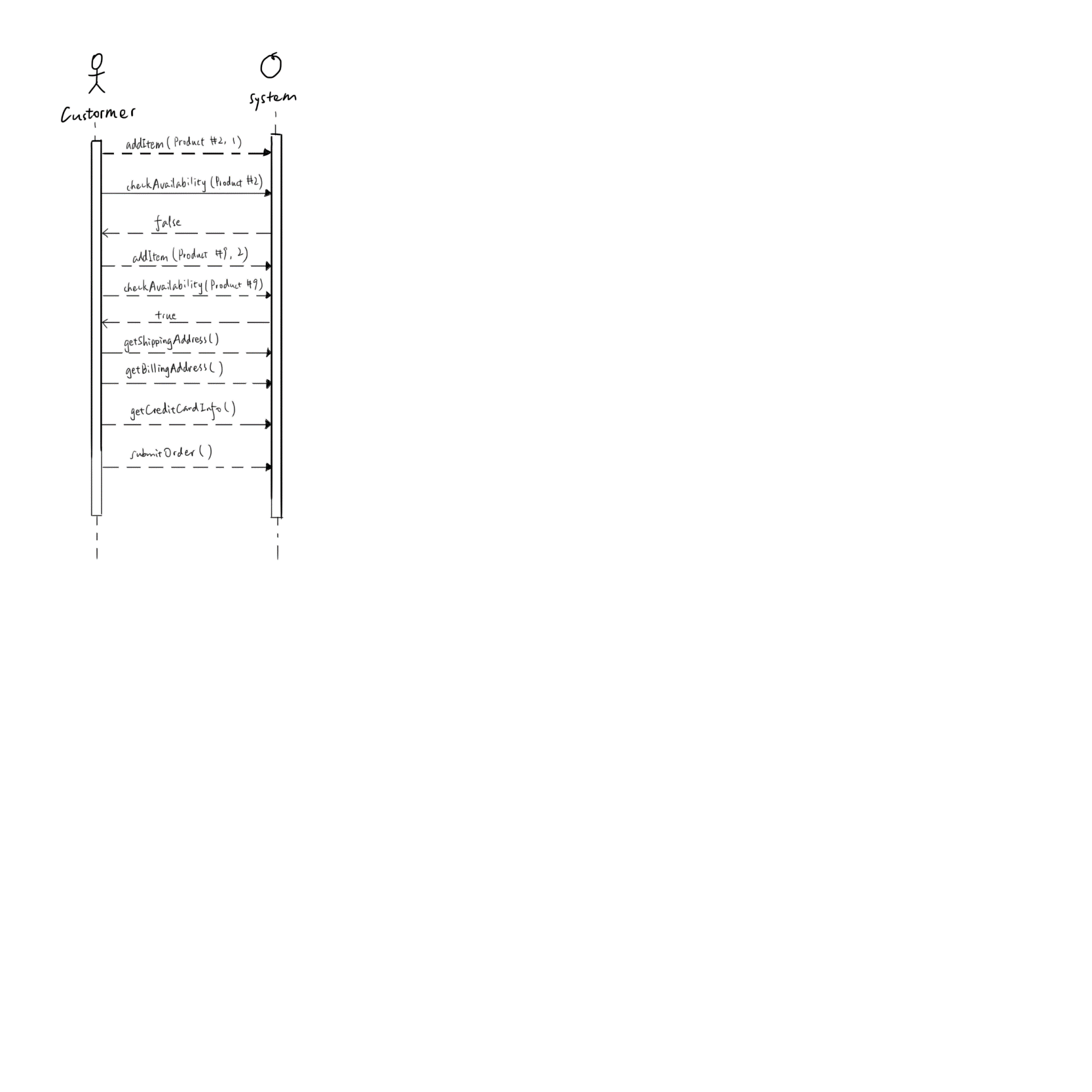
**改进之后的UML建模**



用例图：添加了GoldCustomer，继承自Customer。



类图：删除了AccountingSystem和NotificationSystem两个类，在Customer和GoldCustomer之间添加了继承关系，去掉了ShippingCompany和GoldCustomer之间的联系，并在ShippingCompany和Customer类之间添加关联关系。



主要修改的是一些顺序图的规范。