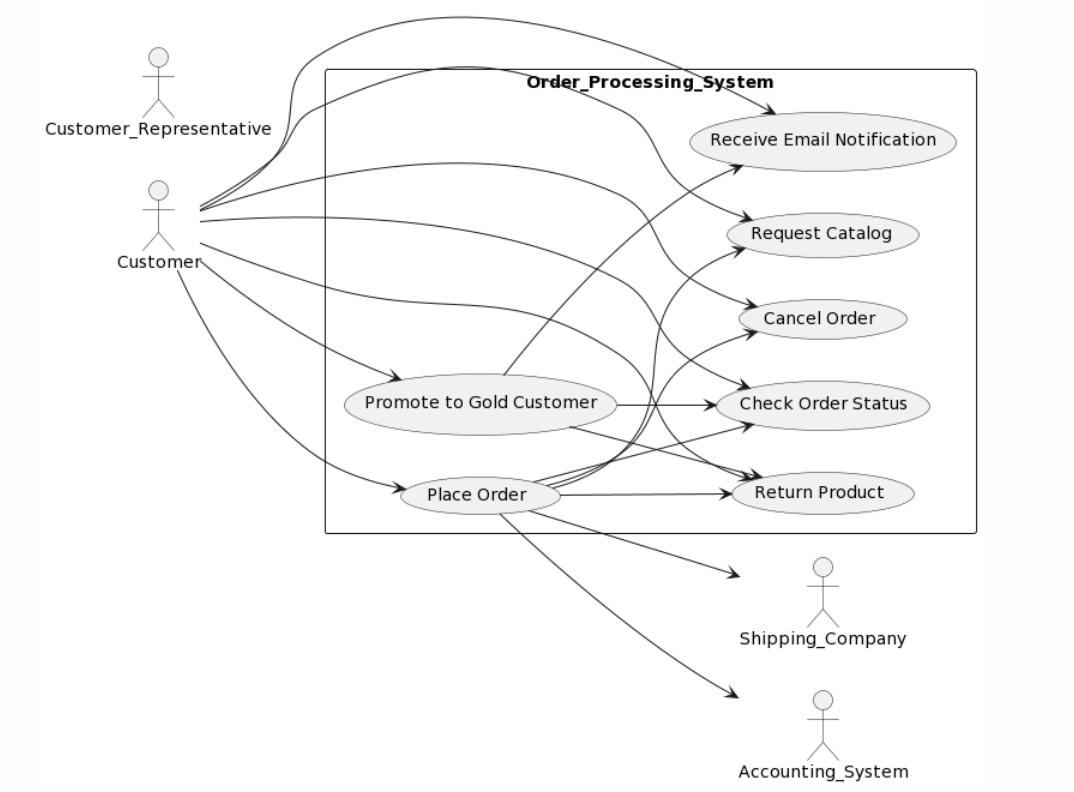
**Case study: Order Processing System**

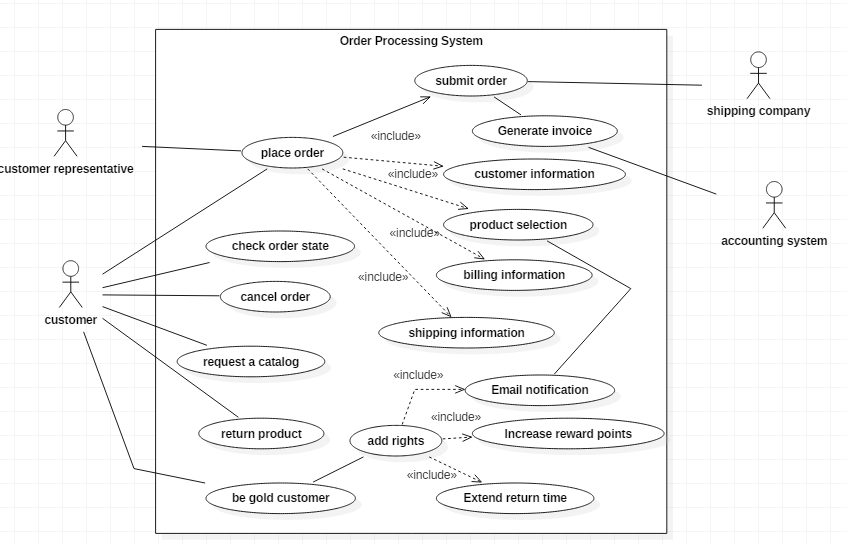
Consider the following problem description: A mail-order company wants to automate its order processing. The initial version of the order processing system should be accessible to customers via the web. Customers can also call the company by phone and interact with the system via a customer representative. 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 actors and use cases for the system described above and show them on a UML Use Case Diagram.

AI给出的用例图



完善后给出的用例图（40分钟前后，没具体记）



参与者方面参考了AI给出的用例图，但是在对于用例的识别方面AI可能并不完善，无法识别相应的成为黄金会员与增加的相关权限的用例，同时在对于参与者与用例之间的关系也存在不足。

下面的提问记录：

第一次提问：直接将上述要求提交给AI（内容较多，不放上来了）

回答

：Actors:

* Customer
* Customer Representative
* Accounting System
* Shipping Company Use Cases:
* Place Order
* Check Order Status
* Cancel Order
* Request Catalog
* Return Product
* Promote to Gold Customer
* Receive Email Notification

其中存在图片无法生成。

第二次提问：生成用例图的相关plantuml代码

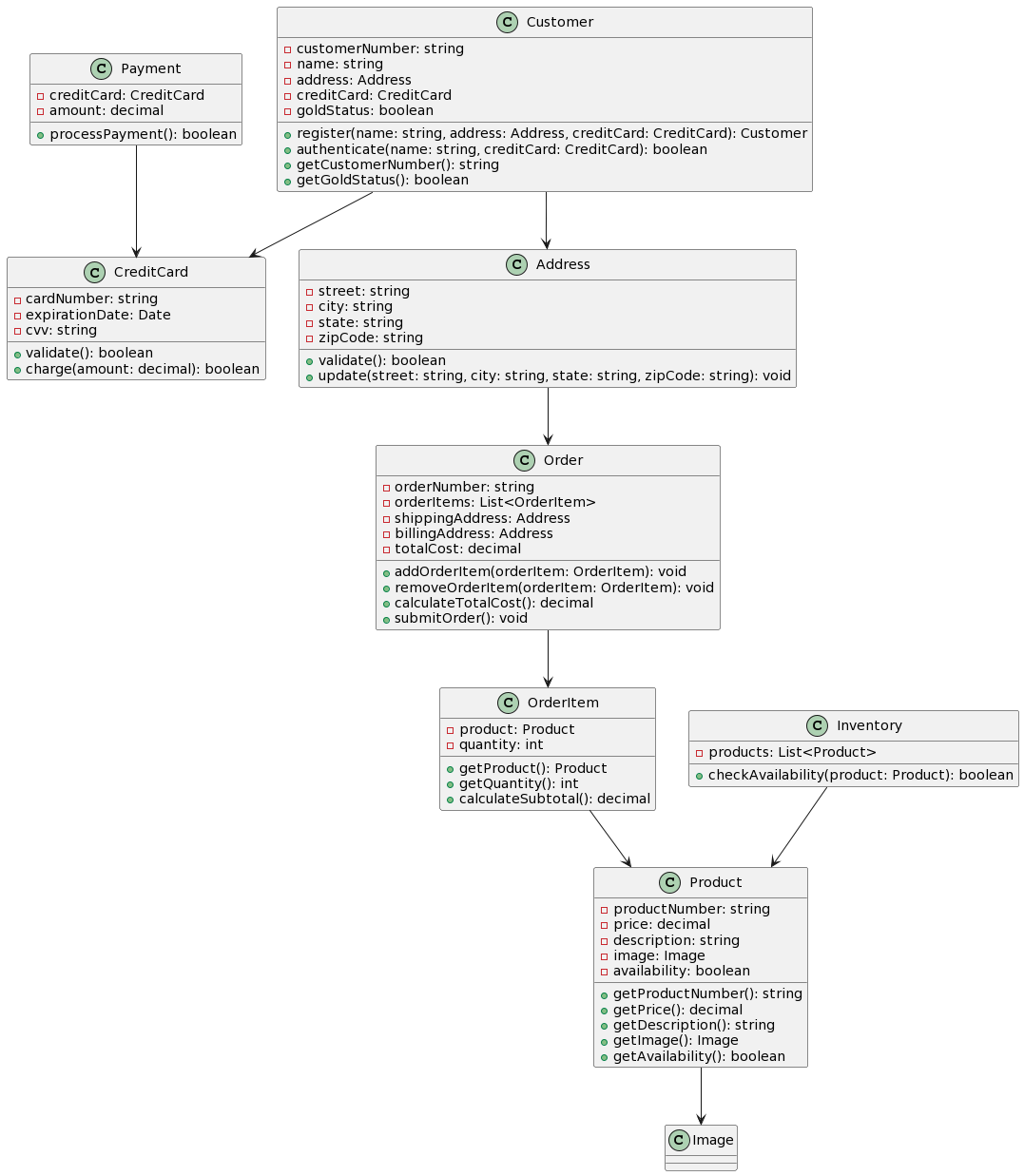
回答：具体代码就不放上来了，具体生成的图片就是上面AI生成的图像。

(2) 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.

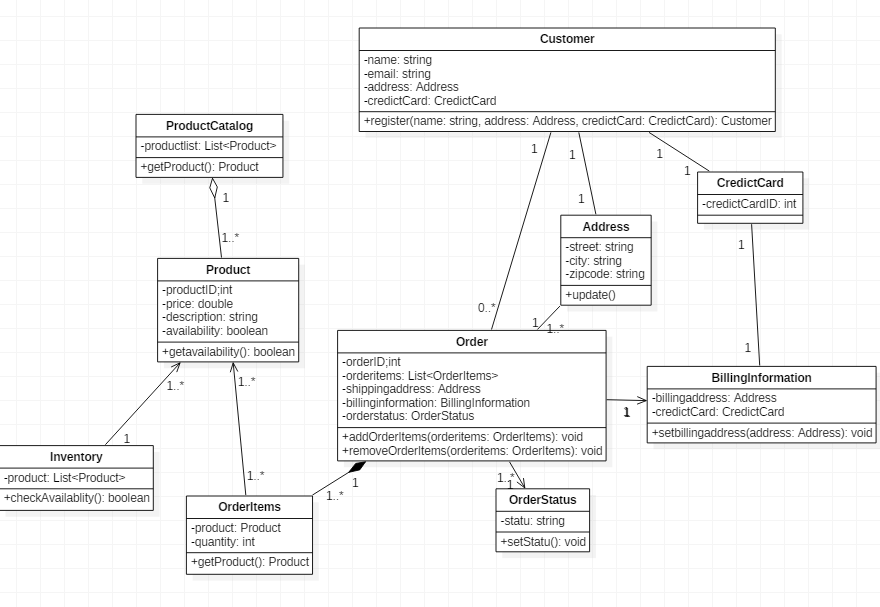
Consider the following use case scenario (for use case “place order”):

*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.*

AI给出的类图：



我具体实现的类图（1小时左右）：



个人感觉AI生成的类图还是比较完整的，其中相应的get之类的方法我只书写了一部分在我自己的类图上，大多数还是参考了AI的类图，但是AI类图中没具体考虑到产品目录，订单的状态等相对于的类，在其类图上做了一定的补充。

下面是具体的问答：

第一次提问：直接给出了本次大作业的要求

回答：

Classes:

* Customer
* Product
* Order
* OrderItem
* Inventory
* Payment
* CreditCard
* Address Critical Operations:
* **Customer**: register, authenticate, getCustomerNumber, getGoldStatus
* **Product**: getProductNumber, getPrice, getDescription, getImage, getAvailability
* **Order**: addOrderItem, removeOrderItem, calculateTotalCost, submitOrder
* **OrderItem**: getProduct, getQuantity, calculateSubtotal
* **Inventory**: checkAvailability
* **Payment**: processPayment
* **CreditCard**: validate, charge
* **Address**: validate, update The use case scenario "place order" involves the creation of an **Order** object, which contains one or more **OrderItem** objects, each of which corresponds to a **Product** object. The **Inventory** class is used to check the availability of each product before it is added to the order. The **Order** class calculates the total cost of the order and submits it for processing. The **Payment** class is responsible for processing the payment using the customer's **CreditCard** object. The **Address** class is used to validate and update the shipping and billing addresses for the order. The **Customer** class is responsible for customer registration, authentication, and checking the customer's gold status.

第二次提问：能给出类具体的属性与方法吗？

回答：

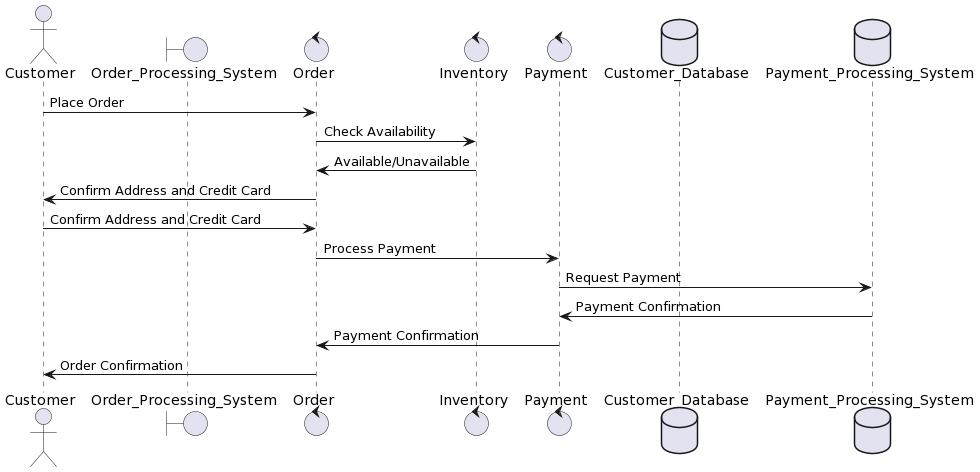
下面是每个类的属性和方法： **Customer** class:

* Attributes:
  + **customerNumber**: string
  + **name**: string
  + **address**: Address
  + **creditCard**: CreditCard
  + **goldStatus**: boolean
* Methods:
  + **register(name: string, address: Address, creditCard: CreditCard): Customer**
  + **authenticate(name: string, creditCard: CreditCard): boolean**
  + **getCustomerNumber(): string**
  + **getGoldStatus(): boolean** **Product** class:
* Attributes:
  + **productNumber**: string
  + **price**: decimal
  + **description**: string
  + **image**: Image
  + **availability**: boolean
* Methods:
  + **getProductNumber(): string**
  + **getPrice(): decimal**
  + **getDescription(): string**
  + **getImage(): Image**
  + **getAvailability(): boolean** **Order** class:
* Attributes:
  + **orderNumber**: string
  + **orderItems**: List
  + **shippingAddress**: Address
  + **billingAddress**: Address
  + **totalCost**: decimal
* Methods:
  + **addOrderItem(orderItem: OrderItem): void**
  + **removeOrderItem(orderItem: OrderItem): void**
  + **calculateTotalCost(): decimal**
  + **submitOrder(): void** **OrderItem** class:
* Attributes:
  + **product**: Product
  + **quantity**: int
* Methods:
  + **getProduct(): Product**
  + **getQuantity(): int**
  + **calculateSubtotal(): decimal** **Inventory** class:
* Attributes:
  + **products**: List
* Methods:
  + **checkAvailability(product: Product): boolean** **Payment** class:
* Attributes:
  + **creditCard**: CreditCard
  + **amount**: decimal
* Methods:
  + **processPayment(): boolean** **CreditCard** class:
* Attributes:
  + **cardNumber**: string
  + **expirationDate**: Date
  + **cvv**: string
* Methods:
  + **validate(): boolean**
  + **charge(amount: decimal): boolean** **Address** class:
* Attributes:
  + **street**: string
  + **city**: string
  + **state**: string
  + **zipCode**: string
* Methods:
  + **validate(): boolean**
  + **update(street: string, city: string, state: string, zipCode: string): void**

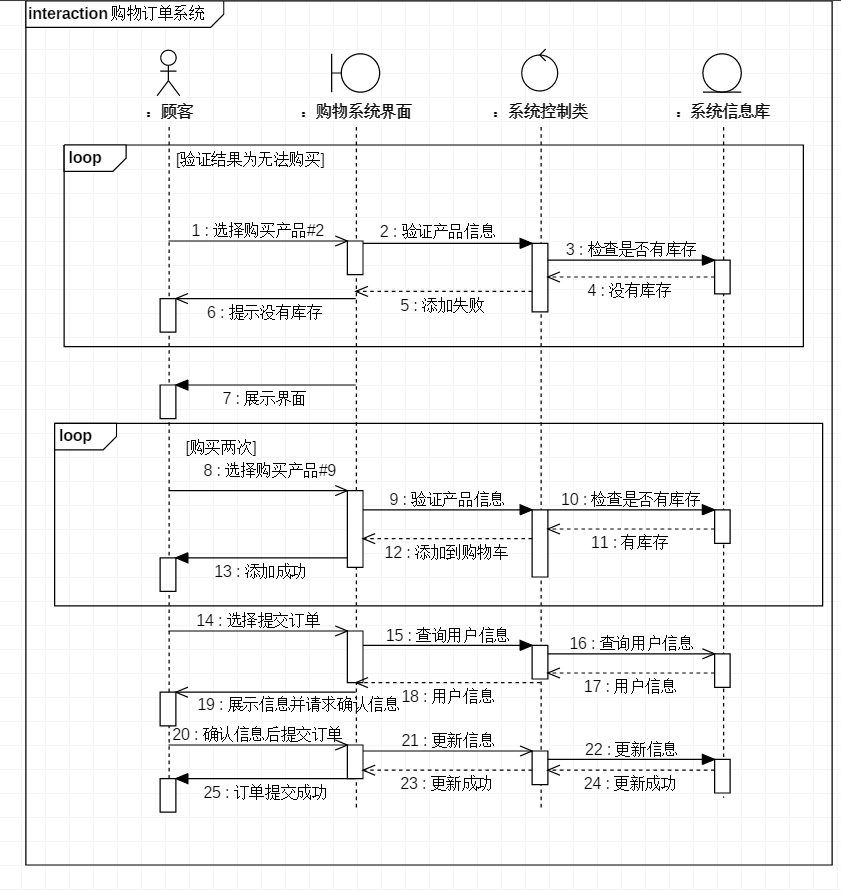
第三次提问：给出相对应的plantuml代码（具体回答就直接参照上面AI的类图）

(3) draw a UML Sequence Diagram for this particular scenario. You may use any software/solution domain objects if needed as well.

AI给出的顺序图：



我给出具体的顺序图（20分钟左右完成）

****

这次基本上没有怎么参照AI给出的顺序图，第一是觉得AI给出的相对于的类过于繁琐，我直接将其涵盖在了一起，其次则是AI给出的顺序图，完全无法参照具体的要求来进行实现，可以说完成的比较抽象。

第一次提问：直接放要求

回答：

以下是该场景的UML序列图： 说明：

* **Customer**对象通过web页面向**Order**对象提交订单。
* **Order**对象检查**Inventory**对象中的库存，以确保每个**OrderItem**对象中的**Product**对象都有足够的库存。
* **Order**对象向**Customer**对象请求确认已注册的**Address**对象和**CreditCard**对象。
* **Customer**对象向**Order**对象确认**Address**对象和**CreditCard**对象。
* **Order**对象使用**Payment**对象来处理支付，该对象在其**CreditCard**对象中具有客户的信用卡信息。
* **Payment**对象使用客户的信用卡信息向支付处理系统发送支付请求。
* 支付处理系统确认支付，并向**Payment**对象返回确认。
* **Payment**对象向**Order**对象返回支付确认。
* **Order**对象向**Customer**对象返回订单确认。

第二次提问：生成对应的plantuml代码

回答：参照上面AI生成的图