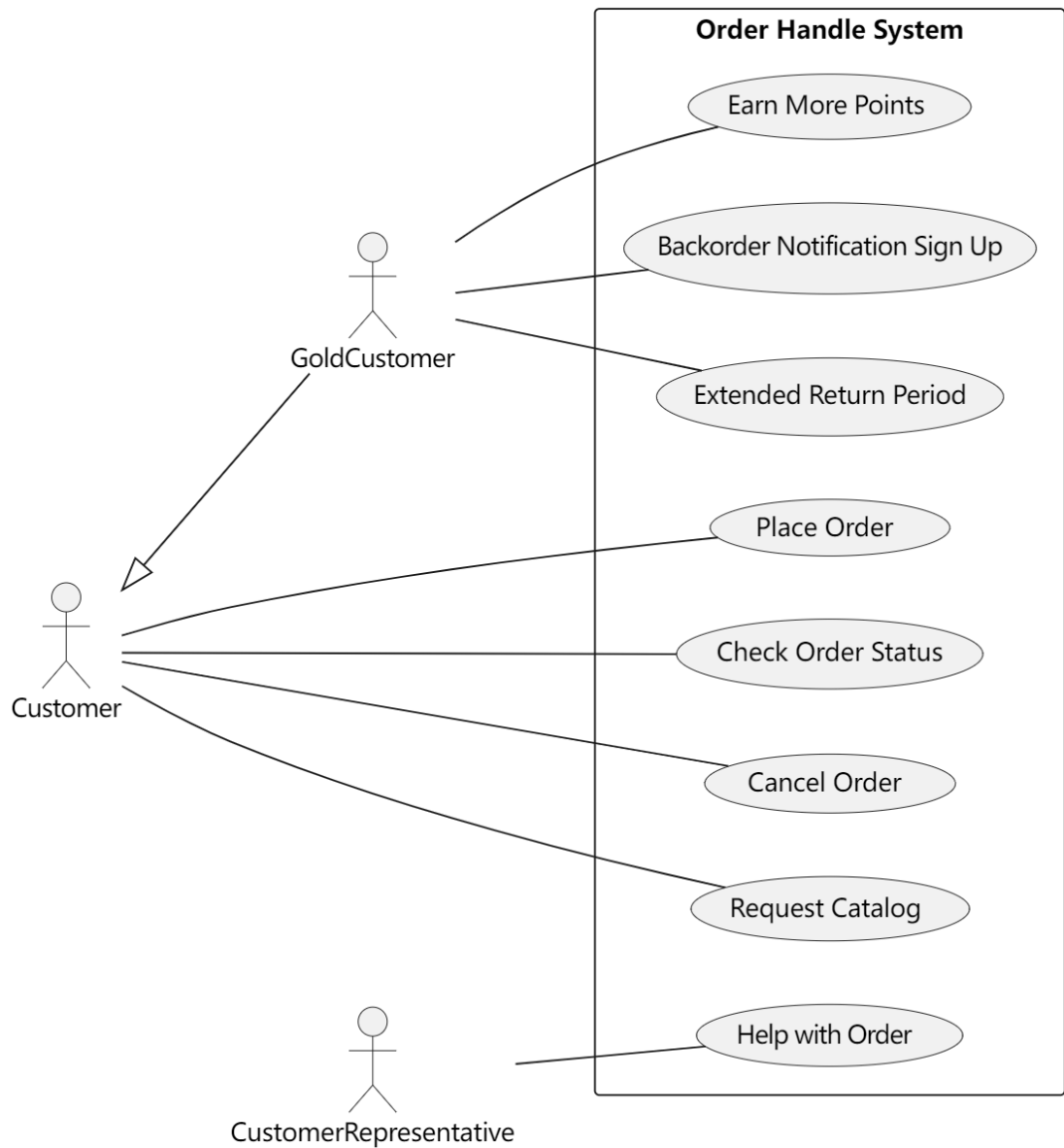
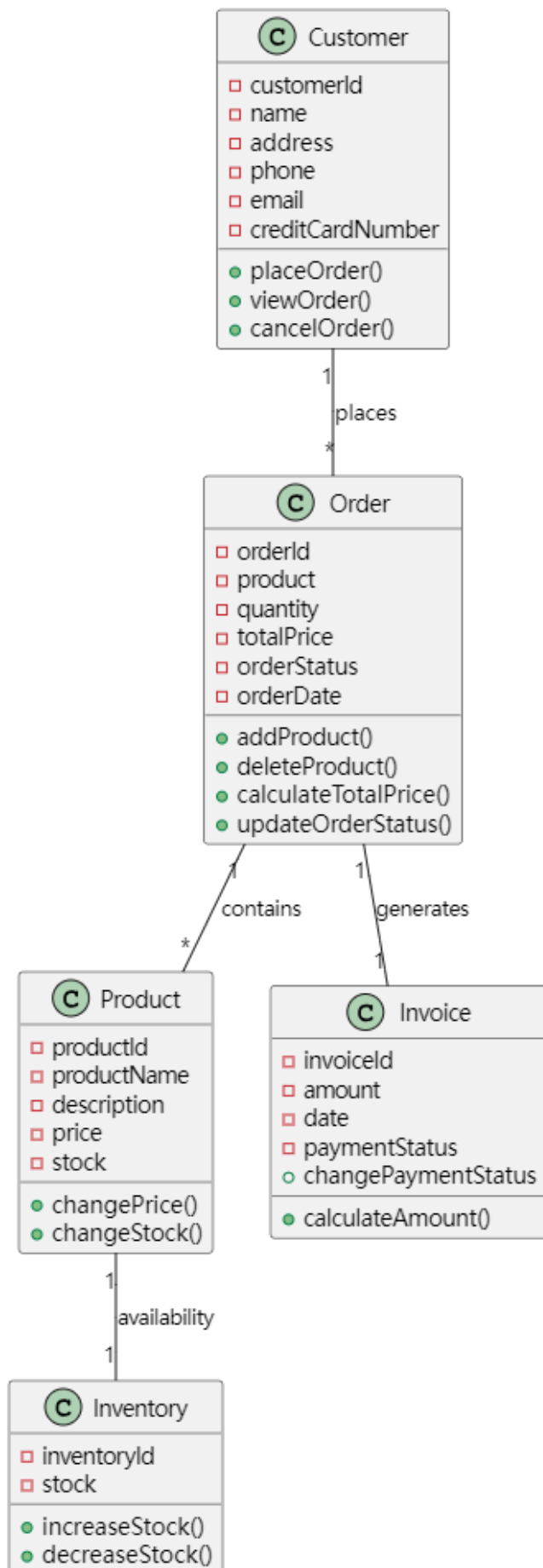


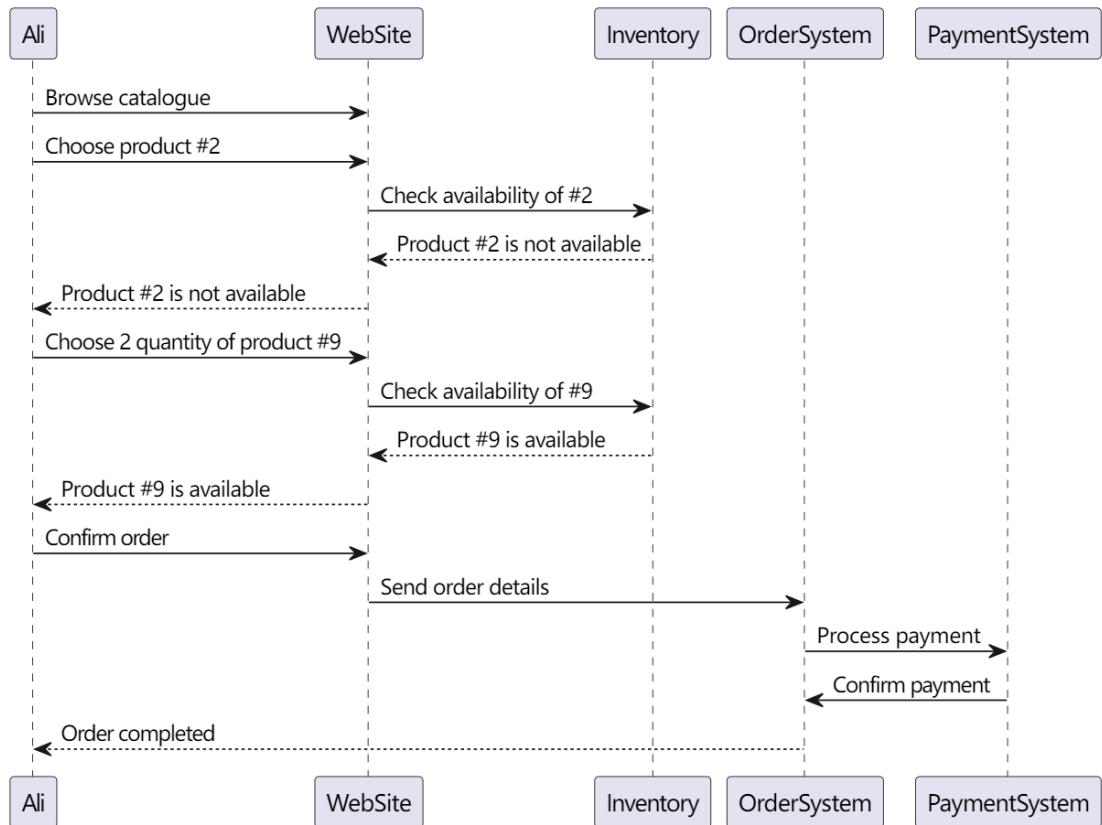
软件需求与建模第二次实验报告

杨曦 2022302121427

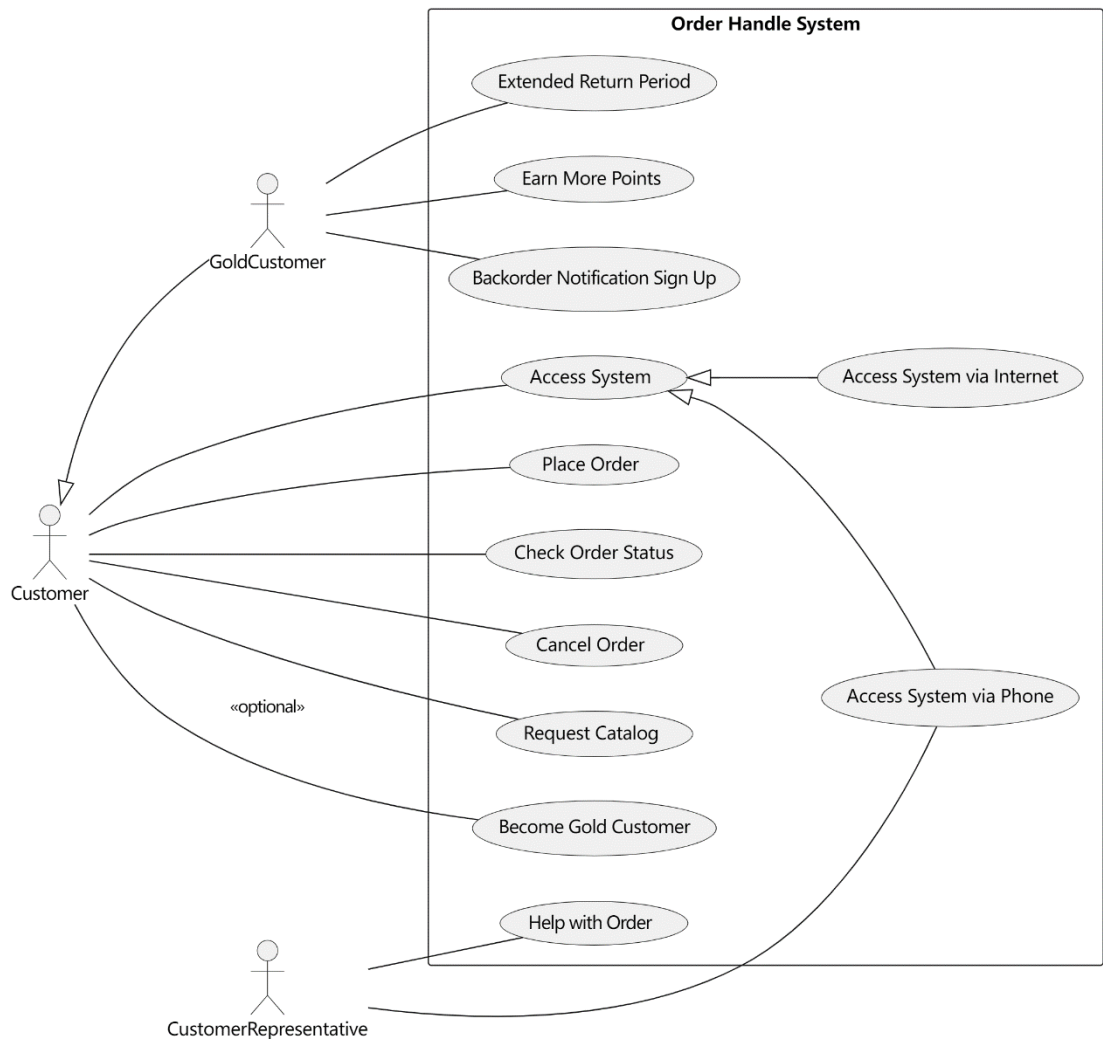
初始方案(依次是用例图、类图和顺序图)







优化后方案(只有用例图被优化)



方案对比

1. 在第一个图中，对于如何访问系统并没有明确的区分，用户（无论金牌客户或者普通客户）都可以进行下单、查看订单状态、取消订单、请求目录的操作。而对于客服代表，他们的主要任务被定义为帮助处理订单。

在第二个图中，"Access System"用例被明确分为了"Access System via Internet"和 "Access System via Phone"，这两种访问方式定义得更加清晰，也将这两种不同的访问方式的特点和差异表现出来。

2. 在优化的用例图中， 添加了可选的 "Become Gold Customer" 用例，表示用户可以成为金牌客户，同时不是所有用户都会成为金牌客户，这是对真实业务场景的一个更深入的理解和表示。

AI 辅助过程

一. 问题建模

1. 翻译与阅读题面 <10分钟>

[IN1] <本次实验的英文题面>

[OUT1] <本次实验的中文题面>

2. 问题领域分析 <2分钟>

[IN2] 对该问题进行问题领域分析：1. 了解业务概况 2. 整理业务目标

[OUT2] 对该问题进行问题领域分析：

1. 了解业务概况：业务是邮购公司，主要通过互联网和电话呼叫进行产品销售，客户可以下订单，检查订单状态，取消订单并请求产品目录。并且公司可能会在未来几年增加系统的新功能。此外，还引入了金牌客户机制，即消费超过一定金额的客户可以成为金牌客户，享受更多服务。

2. 整理业务目标：自动化订单处理，提供客户在线访问，电话互动，确保订单的准确与及时，提高客户满意度和购买率。引入金牌客户制度，提高客户忠诚度。在未来几年内提升和增加新的系统功能，以符合发展和客户需求变化。

3. 涉众分析 <4分钟>

[IN3] 对该问题进行涉众分析，给出涉众的简要定义

[OUT3] 对该问题进行涉众分析，涉众可能包括以下几类：

1. 客户：他们是邮购公司的主要用户，他们通过公司的在线平台或电话调用及客服代理人订购产品，查看订单状态，取消订单，索取目录，退换货物

等。他们对系统的易用性，响应速度和准确性有很高的要求。

2. 公司管理层：他们是要求将订单处理自动化的人，希望这种系统能提高业务效率，提升客户满意度，并带来更高的营业额。他们也可能决定未来的系统升级和新功能添加。

3. 客服代表：他们通过电话为客户提供服务，与订单处理系统进行交互，他们希望系统界面井然有序，实时更新，并能快速准确地处理订单。

4. 会计部门：他们处理由系统生成的发票和付款信息，他们需要系统为他们提供准确而全面的订单信息。

5. 运输公司：他们根据系统生成的购货单来填充和运送商品，他们希望获得清晰，详细的订单信息，从而高效、准时地完成货物的递送。

6. 供应商：他们需要从系统获取销售与库存信息，以便调整生产和供货计划。

7. 金牌客户：他们是公司的高价值客户，除基础邮购服务外，还享有额外权益与服务。例如扩展的退货期，每次购买赚取更多积分等，他们对优质、特权服务的体验有更高需求。

每个涉众群体都有不同的需求和期望，对系统的需求和使用方式也会有所不同。涉众分析可以帮助我们更好地了解和满足这些需求。

二.用例建模

4. 参与者与用例识别 <5分钟>

[IN4] 识别该问题的参与者和用例

[OUT4] 以下是该问题的参与者和用例：

参与者：

1. 客户：他们可以访问网站或电话下订单、查询订单状态、取消订单、请求产品目录及退货。
2. 客服代表：他们通过电话与客户沟通，代表客户操作系统。
3. 会计系统（存在的软件模块）：处理订单发票信息。
4. 运输公司：负责接收订单信息并处理产品的配送。
5. 系统管理人员：对系统进行升级和新增功能的调整。

用例：

1. 下订单：客户可以通过网站或电话向公司下订单。
2. 查询订单状态：客户可以查看他们的订单状态。
3. 取消订单：当客户不再需要产品或其他原因时，他们可以取消订单。
4. 请求产品目录：要查看公司的产品，客户可以请求公司的产品目录。
5. 产品退货：客户如果对购买的产品不满意，他们可以申请退货，但需通过电话操作。
6. 金牌客户额外服务：已满足一定消费额度的客户有权获取金牌客户服务，如更长的退货时间、购物时获得更多积分、对缺货产品设置邮件通知等。

5. 绘制用例图 <15分钟>

非AI过程

6. 优化用例图 <8分钟>

[IN5] 用户以不同方式访问系统的用例如何表示？

[OUT5] 在此情况下，用户可以通过不同的方式访问系统，具体表示如下：

1. 通过网络：用户可以通过互联网访问系统，进行在线购物，查看订单状态，取消订单等操作。

2. 通过电话：用户也可以通过电话联系客服，通过人工进行订单处理、商品退回等服务。

这两种访问方式可能涉及到不同的操作接口和用户体验，因此在设计系统时需要分别进行考虑。同时，由于存在可能增加新功能的需求，系统设计应具备一定的可扩展性。

为了满足这些需求，可以设计如下的数据结构和功能来支撑：

用户数据：包括客户编号，姓名，地址，电话，电子邮件等信息。

产品数据：包括产品编号，名称，描述，图片，库存，价格等信息。

订单数据：包括订单号，客户编号，产品编号，数量，收货地址，信用卡信息，订单状态等信息。

服务功能：包括订购产品，查看订单状态，取消订单，请求目录，通知备货产品等。

连接不同服务的接口：例如与会计系统和运输公司的接口，以完成订单处理。

权限与等级系统：对于普通客户和金牌客户的不同权益做出规定，如退货期限，积分获取等。

[IN6] 使用 plantUML 绘制这一部分的用例

[OUT6] @startuml

left to right direction

actor Customer

actor "Customer Representative" as CR

rectangle "Order Processing System" {

Customer -- (Access System via Internet)

CR -- (Access System via Telephone)

(Order Products) .> (Check Order Status) : <<

extend >>

(Order Products) .> (Cancel Order) : << extend >>

(Order Products) .> (Request Catalog) : << extend >>

Customer -- (Order Products)

CR -- (Process Product Return)

Customer -- (Become Gold Customer) : << optional >>

}

@enduml

三.设计类图

7. 绘制类图 <8分钟>

[IN7] 进行快速的应用领域分析，为上述系统建立一个对象模型。用UML类图表示你的发现，确保识别任何类的关键操作。

[OUT7] 很抱歉，由于我作为文本类AI，无法为你提供直接显示的UML图

像，但是我可以文字描述一下你的UML类图需要包含哪些类以及这些类的关键属性和操作，然后你可以依据这些描述在自己的图形工具中绘制UML类图。

在订单处理系统中，可能需要包含以下类：

1. 客户 (Customer) 类：

- 属性：客户ID，姓名，地址，电话，电子邮件，信用卡号；
- 操作：下订单，查看订单，取消订单；

2. 订单 (Order) 类：

- 属性：订单ID，商品，数量，总价，订单状态，订单日期；
- 操作：添加商品，删除商品，计算总价，更新订单状态；

3. 商品 (Product) 类：

- 属性：商品ID，商品名，描述，价格，库存量；
- 操作：修改价格，修改库存；

4. 目录 (Catalog) 类：

- 属性：目录ID，包含的商品；
- 操作：添加商品，删除商品；

5. 库存 (Inventory) 类：

- 属性：库存ID，商品，库存量；
- 操作：增加库存，减少库存；

6. 客服代表 (CustomerRepresentative) 类：

- 属性：员工ID，姓名；
- 操作：通过电话处理订单；

7. 发票 (Invoice) 类:

- 属性: 发票ID, 金额, 日期, 付款状态;
- 操作: 计算金额, 更改付款状态;

8. 运输公司 (ShippingCompany) 类:

- 属性: 公司名, 地址, 电话;
- 操作: 处理订单, 运输订单;

9. 会计系统 (AccountingSystem) 类:

- 属性: 系统ID;
- 操作: 处理发票;

以上提供了一个基础的对象模型, 你可以根据实际情况进行扩充和优化, 如添加更多的属性和操作或增加新的类。

[IN8] 使用 plantUML 绘制类图

[OUT8] 很抱歉, 作为AI助手, 我无法直接创作出PlantUML格式的类图。

但我能为你提供一个使用PlantUML编写的示例代码, 你可以根据此代码在你的PlantUML编辑器里创作出类图。这是一个包含`Customer`、`Order`、`Product`、`Catalog`、`Inventory`、`CustomerRepresentative`、`Invoice`、`ShippingCompany`和`AccountingSystem`类的简单类图的代码:

```
```plantUML
```

```
@startuml
```

```
class "Customer" as customer {

 -customerId

 -name

 -address

 -phone

 -email

 -creditCardNumber

 +placeOrder()

 +viewOrder()

 +cancelOrder()

}
```

```
class "Order" as order {

 -orderId

 -product

 -quantity

 -totalPrice

 -orderStatus

 -orderDate

 +addProduct()

 +deleteProduct()

 +calculateTotalPrice()

}
```

```
+updateOrderStatus()
}
```

```
class "Product" as product {
 -productId
 -productName
 -description
 -price
 -stock
 +changePrice()
 +changeStock()
}
```

```
class "Catalog" as catalog {
 -catalogId
 -products
 +addProduct()
 +deleteProduct()
}
```

```
class "Inventory" as inventory {
 -inventoryId
```

```
-product

-stock

+increaseStock()

+decreaseStock()

}
```

```
class "CustomerRepresentative" as
customerRepresentative {

 -employeeId

 -name

 +handleOrderViaPhone()

}
```

```
class "Invoice" as invoice {

 -invoiceId

 -amount

 -date

 -paymentStatus

 +calculateAmount()

 +changePaymentStatus

}
```

```

class "ShippingCompany" as shippingCompany {

 -companyName

 -address

 -phone

 +handleOrder()

 +shipOrder()

}

class "AccountingSystem" as accountingSystem {

 -systemId

 +handleInvoice()

}

@enduml

...

```

这段代码表示的类图中有九个类，这些类的属性和方法分别对应我在前一个答案中描述的那些类的属性和方法。根据系统具体情况，这个类图可能需要做出适当修改，比如添加类或者属性等。

## 四.场景建模

[IN9] 为该特定场景画一个UML序列图，给出 plantUML 代码

考虑以下用例场景（用于用例"下订单"）： **Ali** 是前面描述的订单处理公

司的现有客户，已在他们的网站上注册。同时假设，他已经浏览了他拥有的印刷品目录，以及他希望通过公司网站使用他们的产品编号（即#2和#9）购买的两项产品（包括价格）。首先，他试图购买一件产品#2，但在库存中显示不可用。然后，他将两份可用的产品#9加入购物车。然后他被要求确认他在客户数据库中注册的收货和账单地址以及信用卡信息。他点击“提交”按钮完成订单。你可以忽略处理客户认证。

[OUT9] 在以下的PlantUML代码中，描述了这个特定场景的序列图：

```
``plantUML
```

```
@startuml
```

```
participant "Ali" as A
```

```
participant "WebSite" as W
```

```
participant "Inventory" as I
```

```
participant "OrderSystem" as O
```

```
participant "PaymentSystem" as P
```

```
A -> W : Browse catalogue
```

```
A -> W : Choose product #2
```

```
W -> I : Check availability of #2
```

```
I --> W : Product #2 is not available
```

```
W --> A : Product #2 is not available
```

```
A -> W : Choose 2 quantity of product #9
```

```
W -> I : Check availability of #9
```



I --> W : Product #9 is available

W --> A : Product #9 is available

A -> W : Confirm order

W -> O : Send order details

O -> P : Process payment

P -> O : Confirm payment

O --> A : Order completed

@endum1

...

在这个场景中，**Ali**在网站上浏览了他拥有的目录，并选择了产品。网站将请求发送到库存以检查产品的可用性。当产品**#2**不可用，**Ali**选择了2份产品**#9**。库存确认后，**Ali**最终确认了订单。订单详情发送到订单系统，并向支付系统处理支付。支付确认后，订单完成。你可以将这段代码复制到PlantUML，以绘制序列图。