

廈門大學



信息學院軟件工程系

面向對象分析與設計

題 目 售后和服务模块设计文档

班級組別 軟件工程2023級2班 2-7組

小組成員 伊木然、任課老師 邱明

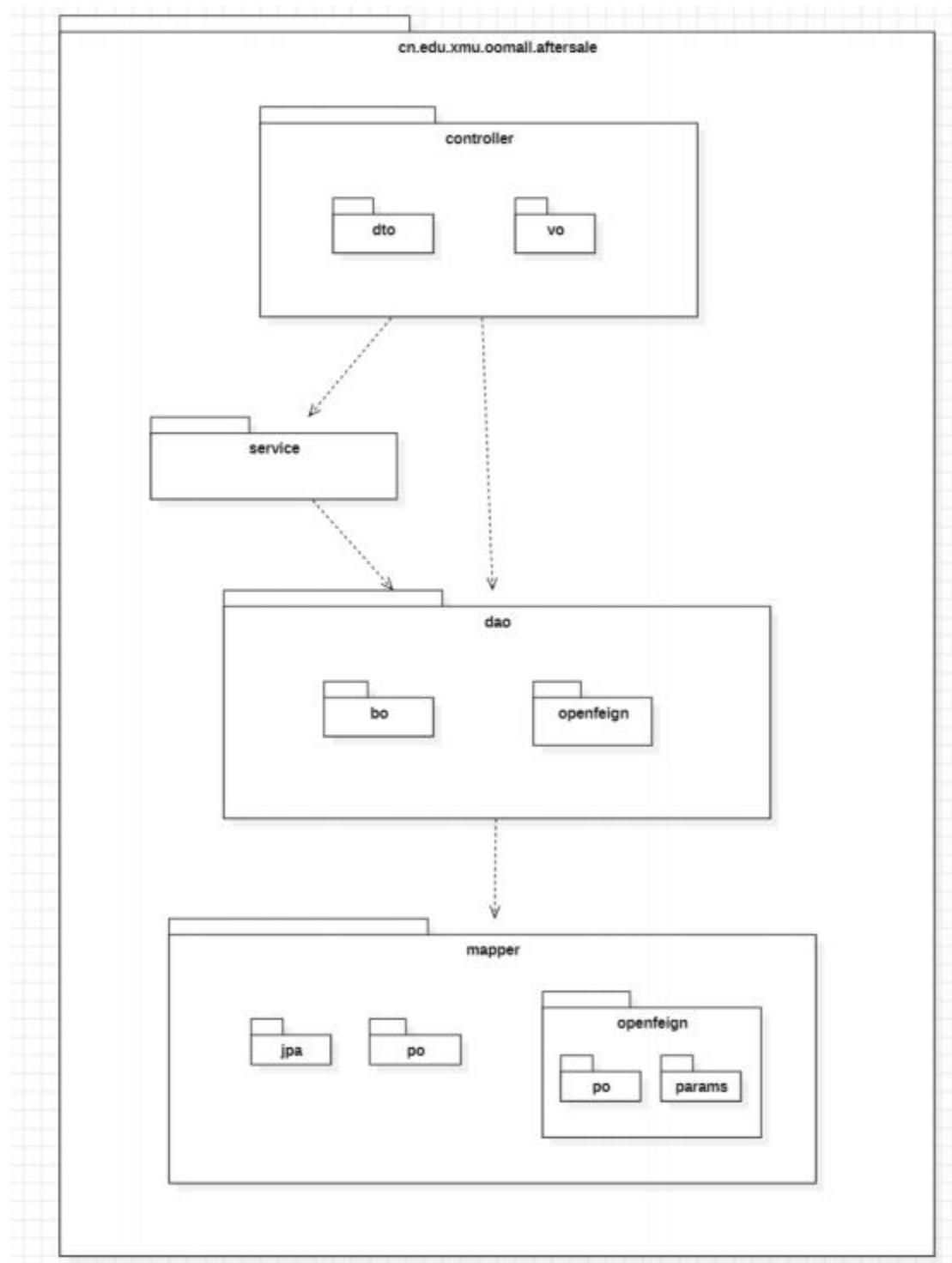
提交時間 2025年12月8日

目录

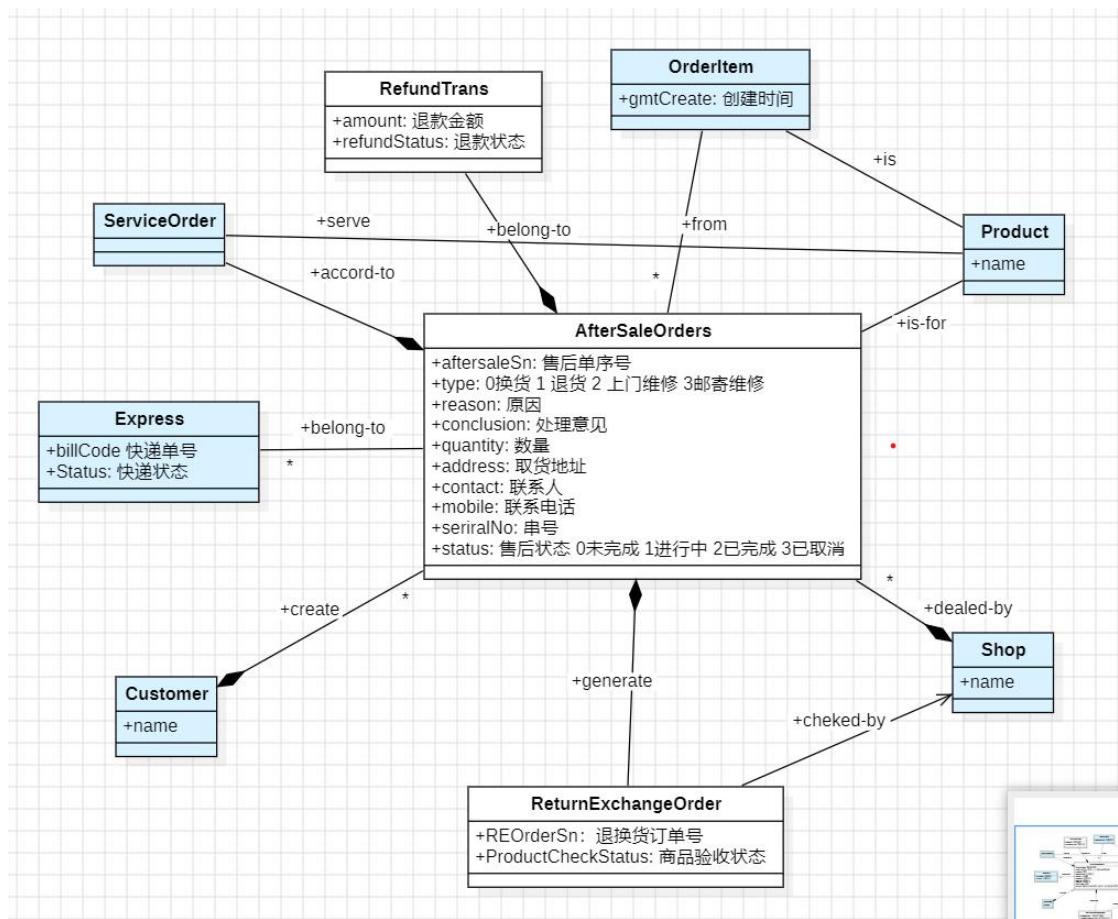
信息学院软件工程系	1
面向对象分析与设计	1
一、售后模块	3
(一) 包图	3
(二) 对象模型	4
(三) 类和接口	5
(四) 状态图	5
1. 售后单状态机	5
2. 仲裁状态机	错误! 未定义书签。
(五) 数据库	错误! 未定义书签。
(六) API与程序逻辑 (时序图)	7
1. 创建售后	错误! 未定义书签。
2. 查询售后	9
3. 取消售后	11
4. 处理退换货	错误! 未定义书签。

一、售后模块

(一) 包图

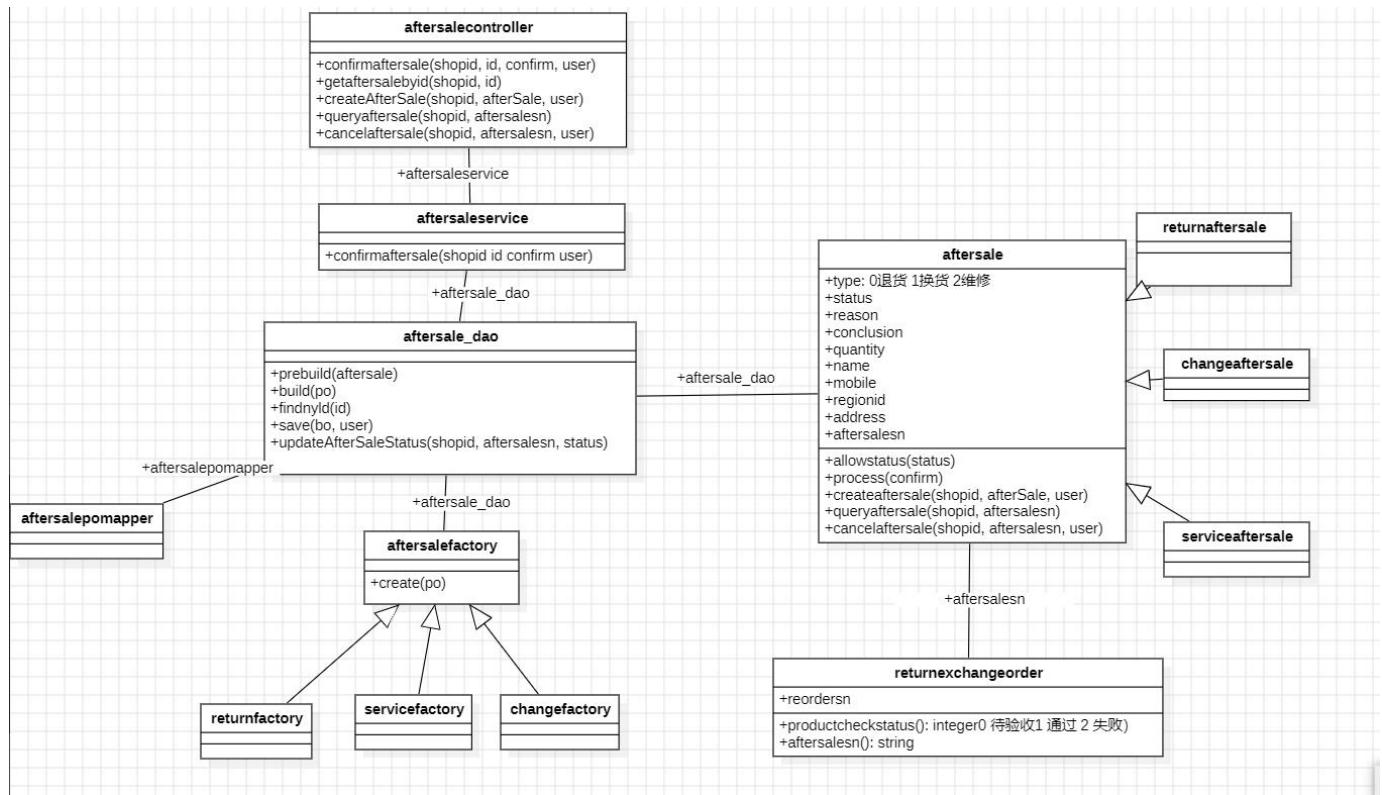


(二) 对象模型



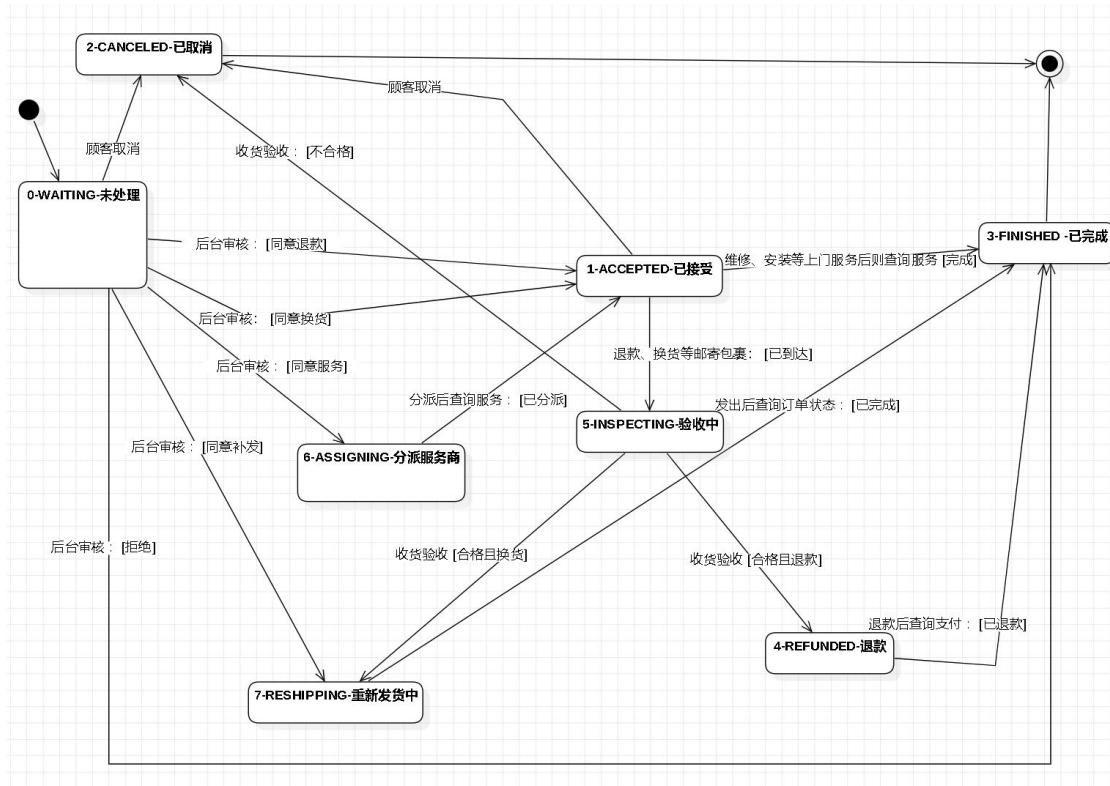
退货退款如果是仅退款则只有一个退款单，否则产生一个退货订单，如果是换货会根据商品再创建一个OrderItem的订单

(三) 类和接口



(四) 状态图

售后单状态机



(五) API与程序逻辑 (时序图)

1. 创建售后

<https://s.apifox.cn/82492a9f-ddfe-481b-a37cf4cad68ae2da/387349687e0>

POST /orderitems/{id}/aftersales

▶ 调试

需检查申请的数量小于订单明细中有效商品的数量（未退货换货）

请求参数

Path 参数

<code>id</code> integer 订单明细id	必填
--------------------------------	----

Header 参数

<code>authorization</code> string 用户token	必填
---	----

Body 参数 application/json 必填

<code>type</code> integer 0换货, 1退货, 2维修	可选	示例
<code>quantity</code> integer	可选	{ "type": 0,
<code>reason</code> string	可选	"quantity": 0,
<code>consignee</code> object (Consignee) Consignee	可选	"reason": "string",
<code>name</code> string 姓名	可选	"consignee": {
<code>mobile</code> string 电话	可选	"name": "string",
<code>regionId</code> integer 地区id	可选	"mobile": "string",
<code>address</code> string 详细地址	可选	"regionID": 0,
		"address": "string"
		}

请求示例代码

Shell JavaScript Java Swift Go PHP Python HTTP C C# Objective-C Ruby OCaml Dart ...

返回响应

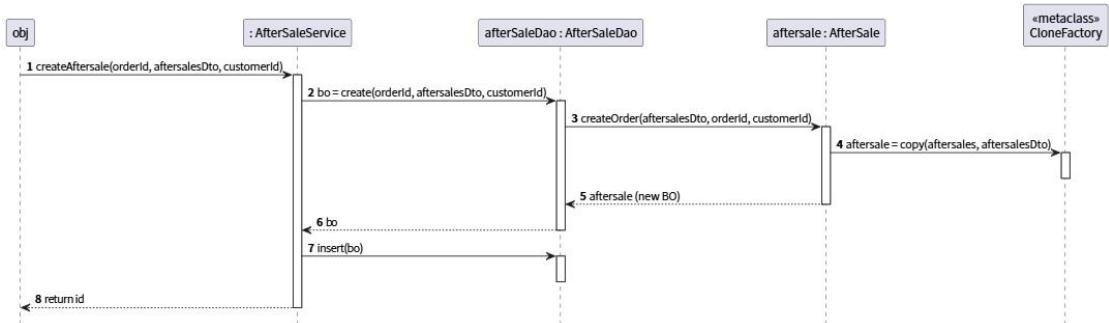
200 成功

application/json

成功

Body

<code>errno</code> integer 示例值: 0	可选	示例
<code>errmsg</code> string 示例值: 成功	可选	{ "errno": 0,
<code>data</code> object (SimpleAftersale) SimpleAftersale 售后服务	可选	"errmsg": "成功",
<code>id</code> integer 售后单id	可选	"data": {
<code>aftersaleSn</code> string 售后单序号	可选	"id": 0,
<code>type</code> integer 0换货, 1退货, 2维修	可选	"aftersaleSn": "string",
<code>conclusion</code> string 处理意见	可选	"type": 0,
<code>quantity</code> integer 货物数量	可选	"conclusion": "string",
<code>status</code> integer 状态	可选	"quantity": 0,
<code>inArbitrated</code> integer 是否仲裁中 0 未仲裁 1 仲裁中	可选	"status": 0,
		"inArbitrated": 0
		}



客户创建售后单时，是调用AfterSaleService的createAfterSale方法。

首先调用AfterSaleDao的create方法，该方法作为领域对象的工厂入口，负责组装上下文。

接着调用AfterSale领域对象的createOrder方法进行初始化。

在初始化过程中，调用CloneFactory的copy方法，将前端传入的DTO数据映射并填充到新的售后单BO对象中。

AfterSaleService拿到创建好且填充完毕的BO对象后，调用AfterSaleDao的insert方法，将这个新的售后单对象持久化保存到数据库中，最后返回生成的售后单ID。

2. 查询售后

<https://s.apifox.cn/82492a9f-ddfe-481b-a37c-f4cad68ae2da/380637113e0>

GET /shops/{id}/aftersales

管理员可通过售后单ID查看所有售后单

请求参数

Path 参数

<code>id</code> integer 店铺id	必需
------------------------------	----

Query 参数

<code>beginTime</code> string 开始时间	可选
<code>endTime</code> string 结束时间	可选
<code>page</code> integer 页码	可选
<code>pageSize</code> integer 每页数目	可选
<code>type</code> integer 售后类型	可选
<code>state</code> integer 售后状态	可选
<code>customerName</code> string 顾客名称	可选

Header 参数

<code>authorization</code> string 用户token	必需
---	----

请求示例代码

Shell JavaScript Java Swift Go PHP Python HTTP C C# Objective-C Ruby OCaml Dart :

返回响应

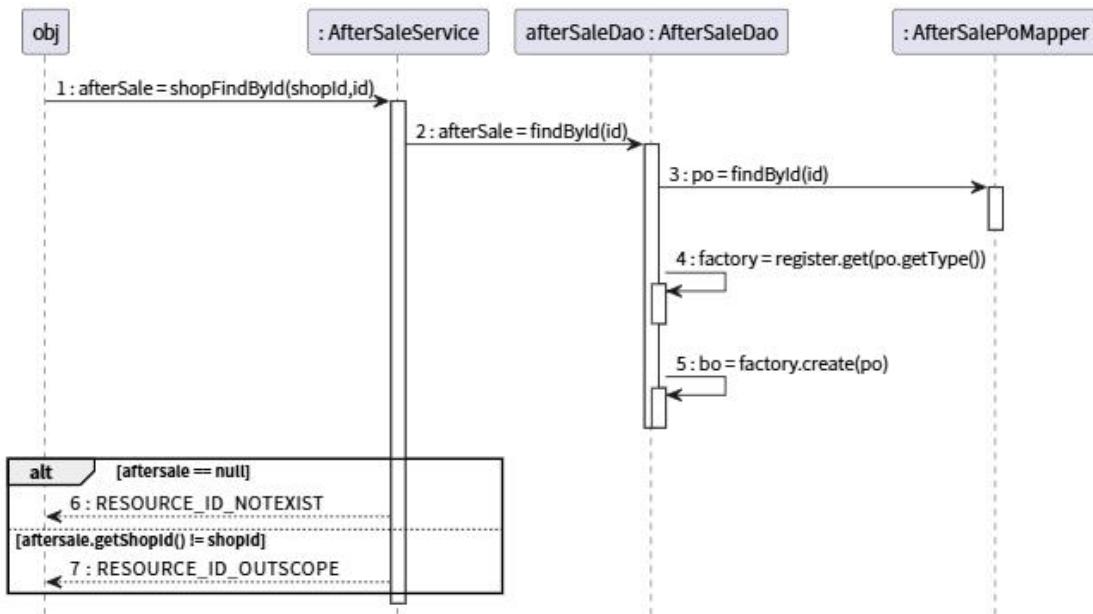
200 成功

application/json

<code>errno</code> integer	可选
<code>errmsg</code> string	可选
<code>data</code> object	可选
<code>page</code> integer 页码	可选
<code>pageSize</code> integer 每页数目	可选
<code>list</code> array[object (SimpleAftersale)]	可选
<code>id</code> integer 售后单id	可选
<code>aftersaleSn</code> string 售后单序号	可选
<code>type</code> integer	可选
0换货, 1退货, 2维修	
<code>conclusion</code> string 处理意见	可选
<code>quantity</code> integer 货物数量	可选
<code>status</code> integer 状态	可选
<code>inArbitrated</code> integer	可选
是否仲裁中 0 未仲裁 1 仲裁中	

示例

```
{  
  "errno": 0,  
  "errmsg": "string",  
  "data": {  
    "page": 0,  
    "pageSize": 0,  
    "list": [  
      {  
        "id": 0,  
        "aftersaleSn": "string",  
        "type": 0,  
        "conclusion": "string",  
        "quantity": 0,  
        "status": 0,  
        "inArbitrated": 0  
      }  
    ]  
  }  
}
```



店铺查看售后单，是调用AfterSaleService的shopFindById方法：

先调用 AfterSaleDao 的 findById方法找到对应的 aftersale，之后从 AfterSalePoMapper 当中得到 po 对象，调用 AfterSaleDao 的 register.get 方法得到工厂，再从工厂中由 po 对象得到对应的 bo 对象，返回之前进行判断：

如果aftersale为空，则返回RESOURCE_ID_NOTEEXIST

如果该aftersale不是该商铺的，则返回

RESOURCE_ID_OUTSCOPE判断合法之后返回bo对象

3. 取消售后

<https://s.apifox.cn/82492a9f-ddfe-481b-a37c-f4cad68ae2da/380646252e0>

PUT /shops/{shopId}/aftersales/{id}/cancel ▶ 调试

只可以在处理中的状态取消

请求参数

Path 参数

shopId	integer 店铺id	必需
id	integer 售后单id	必需

Header 参数 生成代码

authorization	string 用户token	必需
---------------	----------------	----

Body 参数 application/json 必填

reason	string null 取消原因	可选	示例
		{ "reason": "string" }	

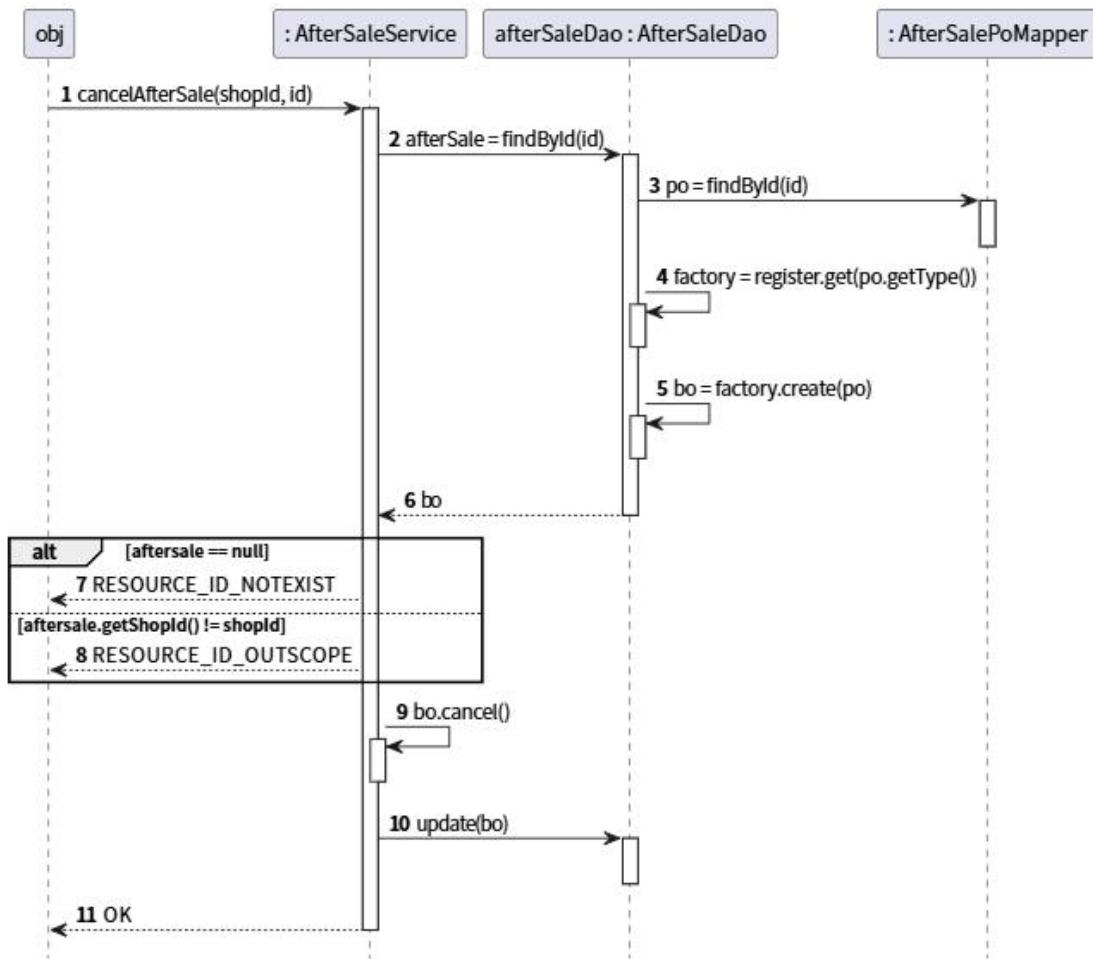
请求示例代码

Shell JavaScript Java Swift Go PHP Python HTTP C C# Objective-C Ruby OCaml Dart ...

返回响应

200 成功

application/json	示例		
成功	{ "errno": 0, "errmsg": "成功" }		
Body			
errno	integer 示例值: 0	可选	
errmsg	string 示例值: 成功	可选	



店铺取消售后单，是调用 AfterSaleService 的 cancelAfterSale 方法：

先调用 AfterSaleDao 的 findById 方法找到对应的 afterSale。从 AfterSalePoMapper 得到 po 对象，调用 AfterSaleDao 的 register.get 方法得到工厂，再由工厂通过 po 对象创建对应的 bo 对象。得到 bo 对象后进行判断：

如果 aftersale 为空，则返回 RESOURCE_ID_NOTEXIST。

如果该 aftersale 不是该商铺的（getShopId 不匹配），则返回 RESOURCE_ID_OUTSCOPE。

判断合法之后，调用 bo 对象的 cancel 方法，执行取消逻辑（例如更改状态为已取消）。

最后调用 AfterSaleDao 的 update 方法，将状态更新后的 bo 对象保存回数据库。

4. 审核售后

<https://s.apifox.cn/82492a9f-ddfe-481b-a37c-f4cad68ae2da/380637116e0>

PUT /shops/{shopId}/aftersales/{id}/confirm ▶ 调试

- 虚拟商品不需将商品退回，即可直接退款或者换货，虚拟商品的类别为313

请求参数

Path 参数

shopId	integer 店铺id	必需
id	integer 售后单id	必需

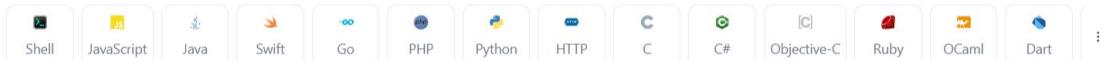
Header 参数

authorization	string 用户token	必需
---------------	----------------	----

Body 参数 application/json 必填

confirm	boolean	可选	示例
conclusion	string	可选	{
type	integer 修改类型, 0换货, 1退货, 2维修	可选	"confirm": true, "conclusion": "string", "type": 0 }

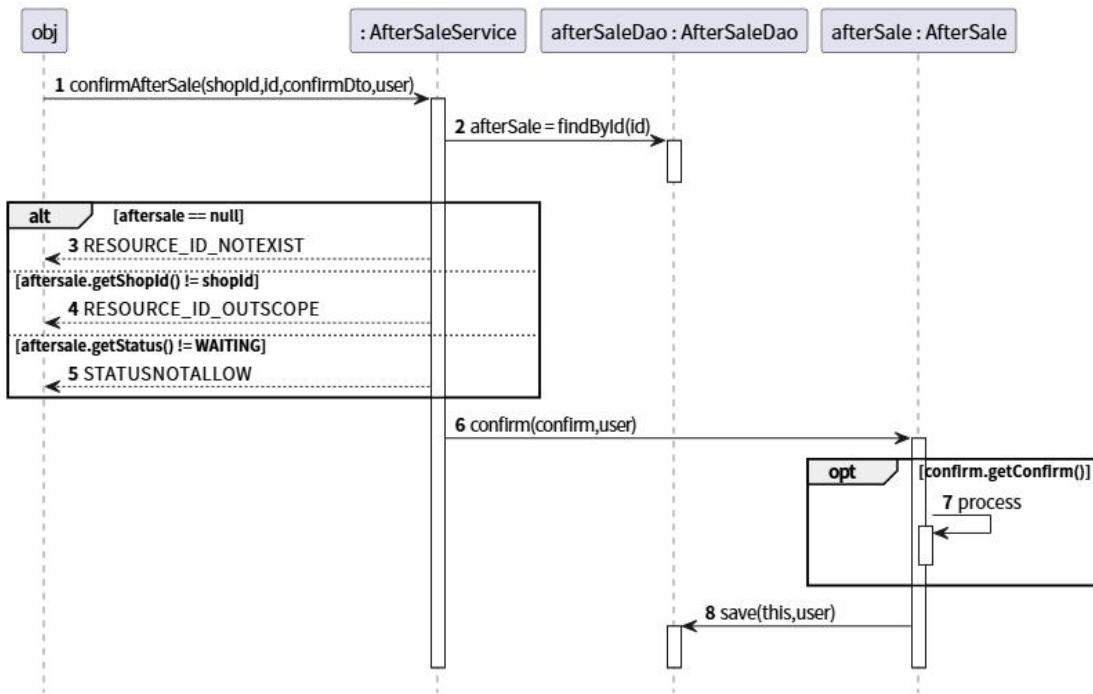
请求示例代码



返回响应

200 成功

application/json	示例	
成功	{ "errno": 0, "errmsg": "成功" }	
Body		
errno	integer 示例值: 0	可选
errmsg	string 示例值: 成功	可选



商户审核售后时，是调用AfterSaleService 的confirmAfterSale 方法。

其中，confirmAfterSale 先通过传进来的（售后） id，调用 AfterSaleDao 的findById 来找到对应的售后afterSale，然后进行判断：

如果afterSale 为空，则报错，返回RESOURCE_ID_NOTEXIST

如果该售后不属于该商铺，则报错，返回 RESOURSE_ID_OUTSCOPE

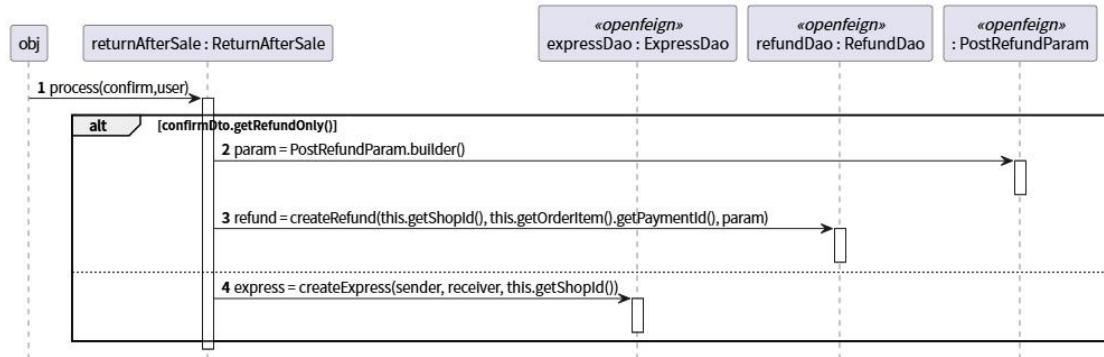
如果该售后不是待审核状态，则报错，返回STATUSNOTALLOW

检查合法之后根据confirm 调用AfterSale 的confirm 方法审核售后

如果不同意则直接结束，使用AfterSaleDao 的save 方法保存结果

如果同意售后，则有退货、换货、寄修， 分以下三种情况：

审核售后—退货



退货情况的售后，是调用 returnAfterSale 的process 方：

如果是仅退款，则只生成一个退款单

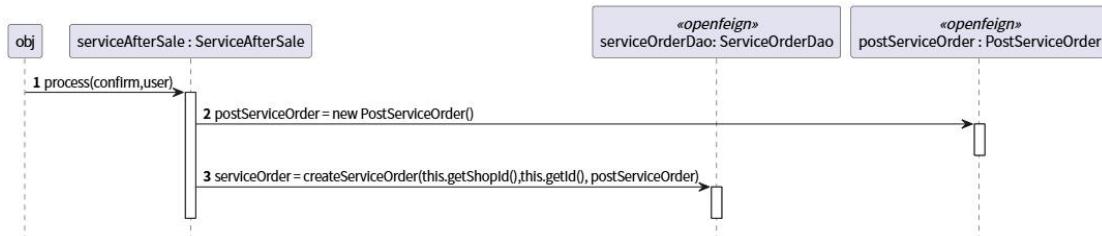
如果不是仅退款，还需要调用 ExpressDao 的create 方法创建物流单

审核售后—换货



换货情况的售后，需要用 ExpressDao 的 `create` 方法创建物流单

审核售后—寄修



寄修情况的售后，请求 ServiceOrder 得到数据，再调用 ServiceOrderDao 的 `create` 方法创建服务单