**详细设计**

—— SCUT网上购物系统

目录

**[详细设计](#_Toc530601776)** [1](#_Toc530601776)

[—— SCUT网上购物系统 1](#_Toc530601777)

**[0. 修订历史](#_Toc530601778)** [2](#_Toc530601778)

**[1. 概述](#_Toc530601779)** [2](#_Toc530601779)

[1.1 系统概述 2](#_Toc530601780)

[1.2 软件设计目标 2](#_Toc530601781)

**[2. 术语表](#_Toc530601782)** [3](#_Toc530601782)

**[3. 设计概述](#_Toc530601783)** [4](#_Toc530601783)

[3.1 系统的复用计划 4](#_Toc530601784)

[3.2 系统接口设计 4](#_Toc530601785)

[3.2.1 提供给用户的界面 4](#_Toc530601786)

[3.2.2 系统外部接口设计 8](#_Toc530601787)

[3.2.3 系统内部接口设计 8](#_Toc530601788)

[3.3 对象模型设计 9](#_Toc530601789)

[3.3.1 对象模型 9](#_Toc530601790)

[3.3.2 前端页面结构 9](#_Toc530601791)

[3.3.3 对象描述 9](#_Toc530601792)

[3.4 系统功能（用例）实现详细设计 11](#_Toc530601793)

[3.4.1 顺序图 11](#_Toc530601794)

[3.4.2 状态图 16](#_Toc530601795)

[3.5 系统非功能设计 26](#_Toc530601796)

[3.5.0 约束矩阵 26](#_Toc530601797)

[3.5.1 性能设计 27](#_Toc530601798)

[3.5.2 维护性设计 27](#_Toc530601799)

[3.5.3 易用性设计 27](#_Toc530601800)

[3.5.4 兼容性设计 27](#_Toc530601801)

[3.5.5 可拓展性设计 27](#_Toc530601802)

[3.5.6 可靠性设计 27](#_Toc530601803)

[3.5.7 灾备设计 28](#_Toc530601804)

[3.5.8 安全性设计 28](#_Toc530601805)

# 0. 修订历史

|  |  |  |  |
| --- | --- | --- | --- |
| **版 本** | **日 期** | **描 述** | **作 者** |
| **v1.0** | **2018.10.22** | 完成概述部分 | **林少辉** |
| **V1.1** | **2018.10.24** | 系统复用计划 | **刘斌** |
| **v1.2** | **2018.10.26** | 系统内外部接口设计 | **刘斌** |
| **V1.3** | **2018.10.26** | 对象模型设计 | **游增** |
| **V1.4** | **2018.10.30** | 系统顺序图 | **林水泉** |
| **v1.5** | **2018.10.30** | 系统状态图 | **汪喆** |
| **V1.6** | **2018.10.29** | 非功能性需求 | **林烁** |
| **v1.6** | **2018.10.31** | 根据各部分，补充术语表 | **林少辉** |
| **v2.0** | **2018.10.31** | 小组讨论，对概要设计进行整体修订 | **全体成员** |
| **v2.1** | **2018.11.21** | 格式审校，调整排版，  添加目录页码页眉 | **林烁** |
|  |  |  |  |

# **1. 概述**

## 1.1 系统概述

SCUT Mall是一个模仿京东的购物网站，分为客户端和管理员端。客户端主要用于让用户进行购物、操作购物车、查询商品信息、打折促销活动以及用户购物统计功能；而管理员端用于系统管理员、业务员进行商品分类管理和销售数据统计等场景。

## 1.2 软件设计目标

其软件设计目标是实现一个采用JavaEE开发、包括账户模块、商品管理模块、查询模块、购物模块、系统管理模块、统计模块等六大模块的Java Web网页。其中账户模块是许多IT系统的基本模块，用户可以注册、登录账户，并编辑修改账户的身份、地址信息。商品管理模块是SCUT网上购物系统的核心模块之一，商品管理是指商品上推到系统中，呈现在用户的可浏览视图中以供购买，商品下架则是不开放该商品的购买权限以及移出用户的可浏览识图。查询模块是SCUT网上购物系统的核心模块之一，用户可以查询在售商品（即已上架的商品）、查询购物车中的商品和查询已购买的商品。购物模块是SCUT网上购物系统的核心模块之一，用户对心仪的在售商品或购物车中的商品进行购买产生结算，主要有订单结算、积分结算和打折结算。系统管理模块是SCUT网上购物系统的支撑模块之一，管理员通过管理界面查看用户信息、业务员信息、商品信息、业务员/统计人员授权、用户封号等操作，以及开启和关闭系统。统计模块是SCUT网上购物系统的核心模块之一，统计人员根据系统的业务数据进行相关的用户、订单、营业额的统计，为商业决策提供建议。

而在具体的开发上，我们选择了springboot框架，使得开发更加容易入手；而MVC模式的使用，有利于将代码分解，逻辑更加清晰；此外，我们还将使用Tomcat作为服务器，租用阿里云的入门级服务器，这些资源都是不可或缺的！

在非功能性需求方面，系统响应速度：用户操作系统平均响应时间小于3秒。支持最大并发数为50。在维护方面整个应用软件系统应能够连续7\*24小时不间断工作，应用软件中的任一模块更新、加载时，在不更新与上下模块的接口的前提下，以不影响业务运转和服务为原则；应用软件具有较高的自动化程度，如：自动异常调度、自动故障告警、自动任务恢复等；应用软件具备相应容错手段，能容许操作人员的某些失误操作。在安全性上包括用户和权限管理，数据范围权限的控制，对支付通讯进行加密，防止数据篡改、攻击。

# 2. 术语表

|  |  |
| --- | --- |
| 术语 | 术语解释 |
| HTML | 超文本标记语音，标准通用标记语言下的一个应用。“超文本”就是指页面内可以包含图片、链接，甚至音乐、程序等非文字元素 |
| CSS | 层叠样式表是一种用来表现HTML（标准通用标记语言的一个应用）或XML（标准通用标记语言的一个子集）等文件样式的计算机语言。 |
| javascript | JavaScript一种直译式脚本语言，是一种动态类型、弱类型、基于原型的语言，内置支持类型。它的解释器被称为JavaScript引擎，为浏览器的一部分，广泛用于客户端的脚本语言，最早是在HTML（标准通用标记语言下的一个应用）网页上使用，用来给HTML网页增加动态功能 |
| Tomcat | Tomcat 服务器是一个免费的开放源代码的Web 应用服务器，属于轻量级应用[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/tomcat/_blank)，在中小型系统和并发访问用户不是很多的场合下被普遍使用，是开发和调试JSP 程序的首选 |

# 3. 设计概述

## 3.1 系统的复用计划

即库、框架、模式、构件等方面的复用。

本系统采用SpringBoot框架，创建独立的Spring应用程序。直接嵌入了Tomcat,自动配置Spring和第三方库，不需要XML配置。

MVC模式，Model View Controller，是模式-视图-控制器，用一种业务逻辑、数据、界面显示分离的方法组织代码，将业务逻辑聚集到一个部件里面，在改进和个性化定制界面及用户交互的同时，不需要重新编写业务逻辑。MVC用于映射传统的输入、处理和输出功能在一个逻辑的图形化用户界面的结构中。Model（模型）表示应用程序核心（比如数据库记录列表）。View（视图）显示数据（数据库记录）。Controller（控制器）处理输入（写入数据库记录）。

MVC 模式同时提供了对 HTML、CSS 和 JavaScript 的完全控制。Model（模型）是应用程序中用于处理应用程序数据逻辑的部分。通常模型对象负责在数据库中存取数据。View（视图）是应用程序中处理数据显示的部分。通常视图是依据模型数据创建的。Controller（控制器）是应用程序中处理用户交互的部分。通常控制器负责从视图读取数据，控制用户输入，并向模型发送数据。

说明：在分析、设计、编码、评审等活动中未嵌入与复用有关的活动；软件过程中也未定义复用设计师、复用工程师的角色，故

创建可复用构件：无

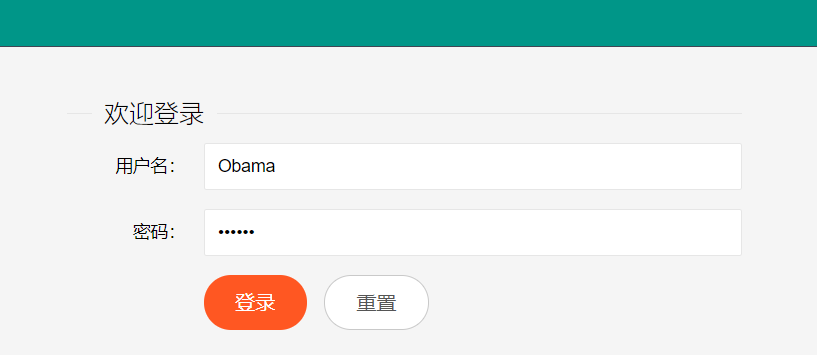
使用可复用构件：无

说明：在分析、设计、编码、评审等活动中未嵌入与复用有关的活动；软件过程中也未定义复用设计师、复用工程师的角色；项目规模较小，无复用需求；功能模块较为独立，无需复用构件支持。

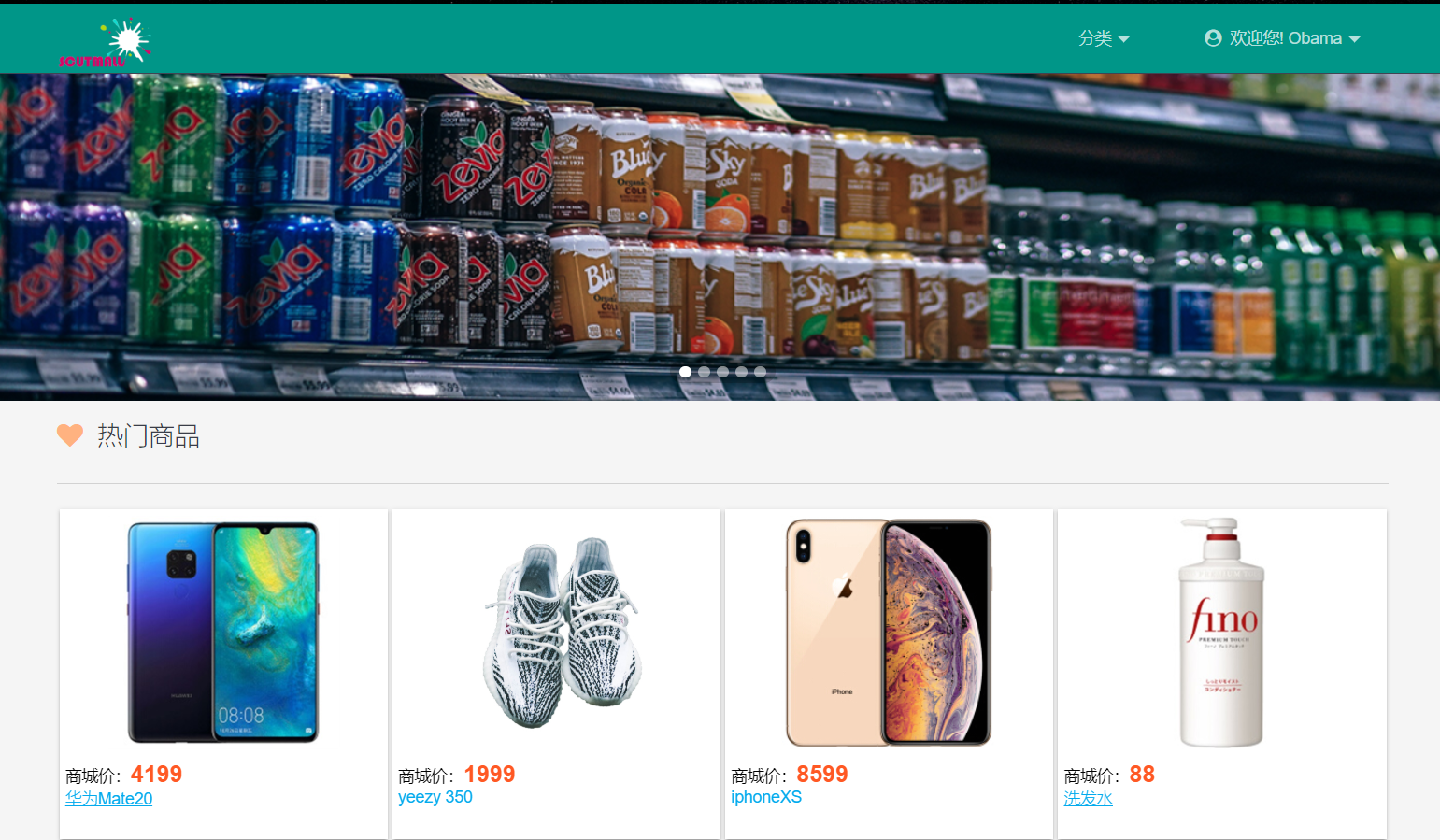
## 3.2 系统接口设计

### 3.2.1 提供给用户的界面

1. 用户点击登录，输入用户名和密码



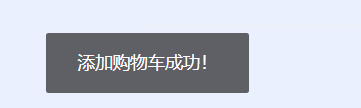
1. 用户进入商品主页



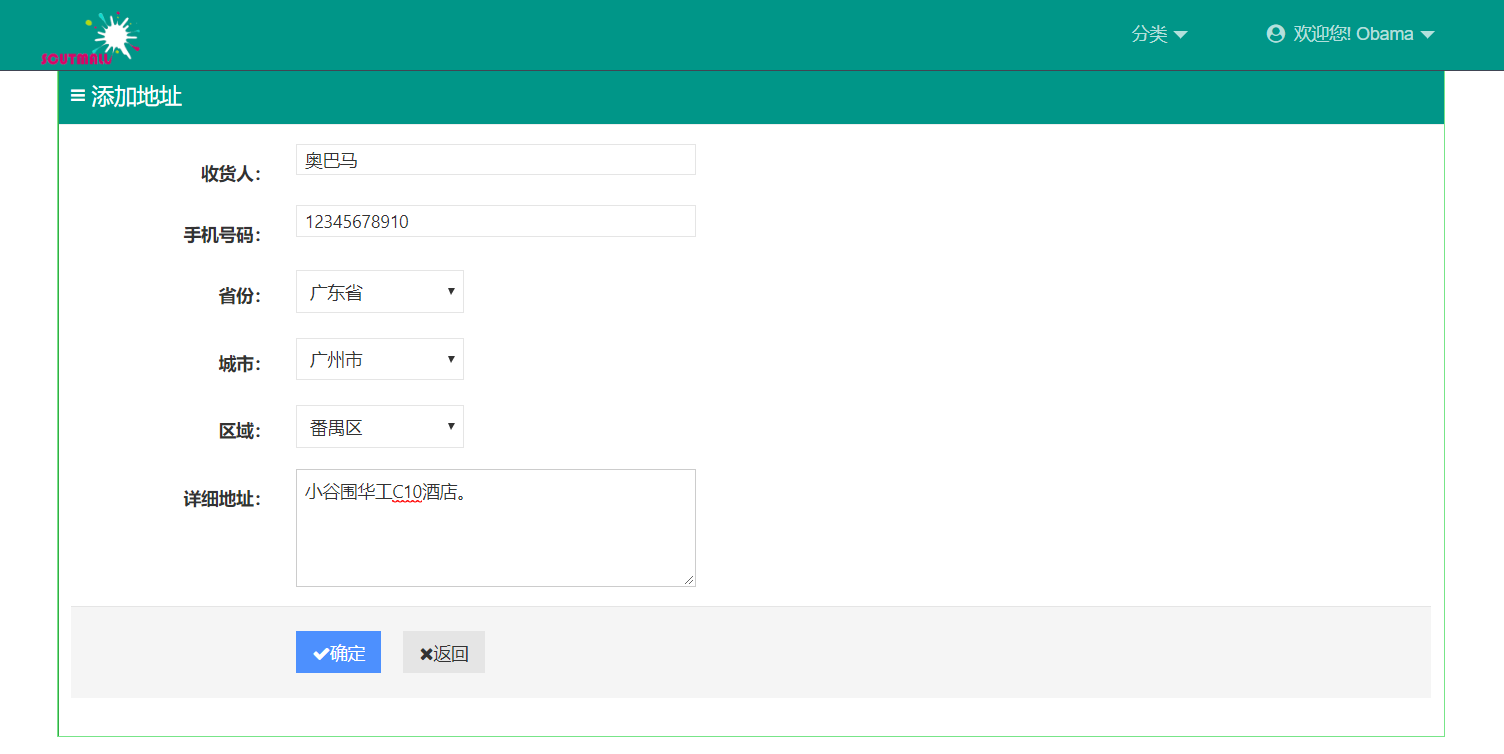
1. 用户点击商品进入商品详情页



1. 用户点击加入购物车



1. 用户点击我的地址，添加地址信息





1. 用户点击我的购物车，进入购物车页面



1. 页面显示购物成功



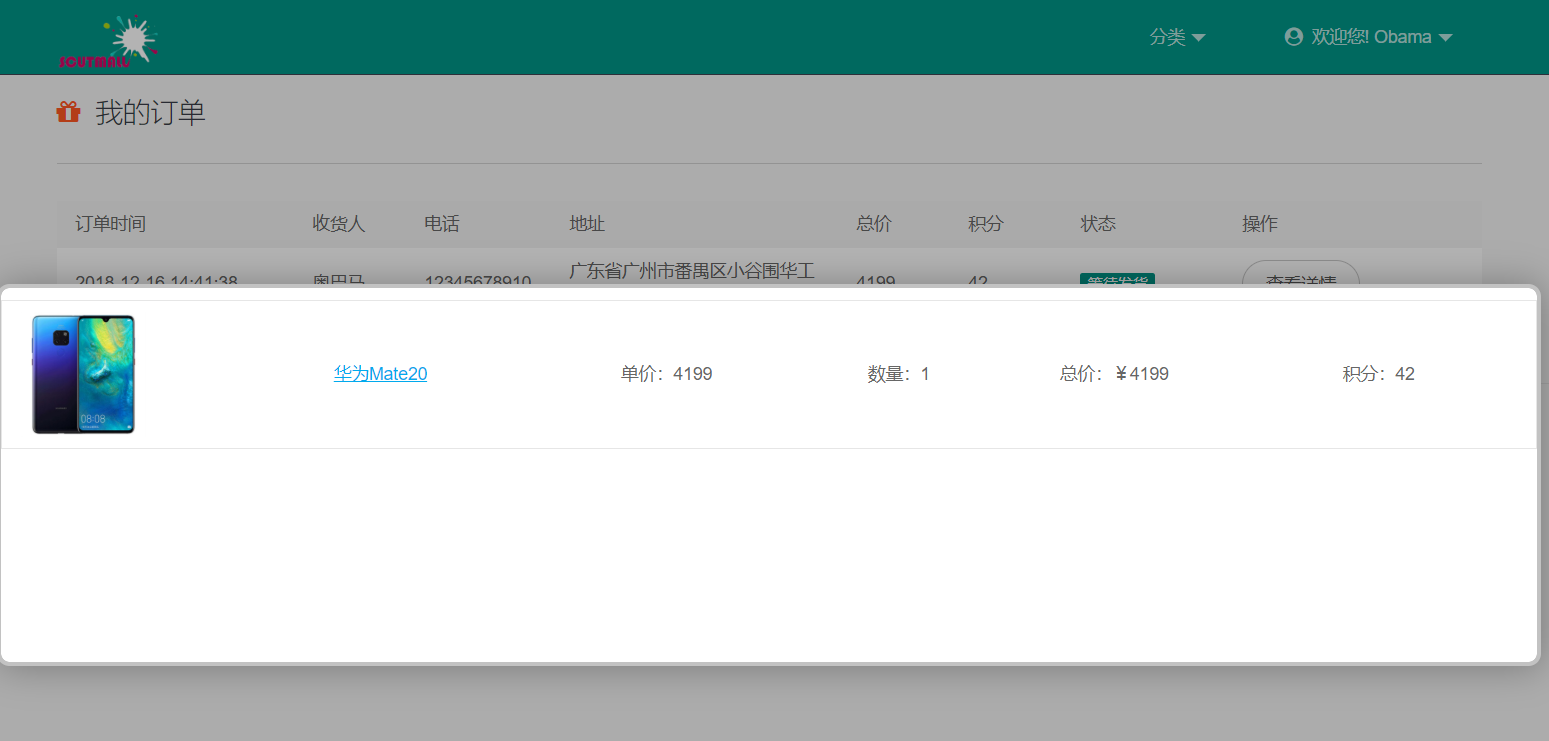
1. 用户点击我的订单，查看订单信息



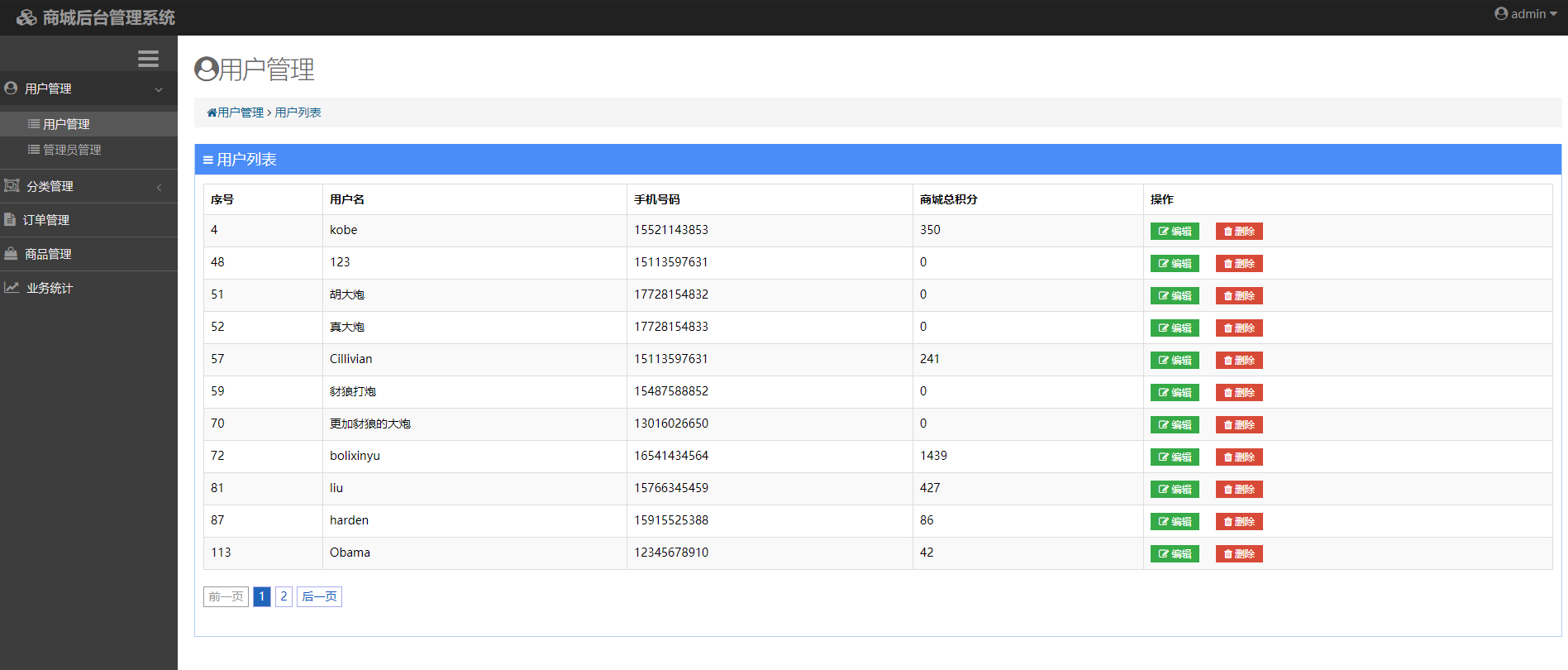
1. 用户点击付款，显示付款成功后订单状态显示为等待发货

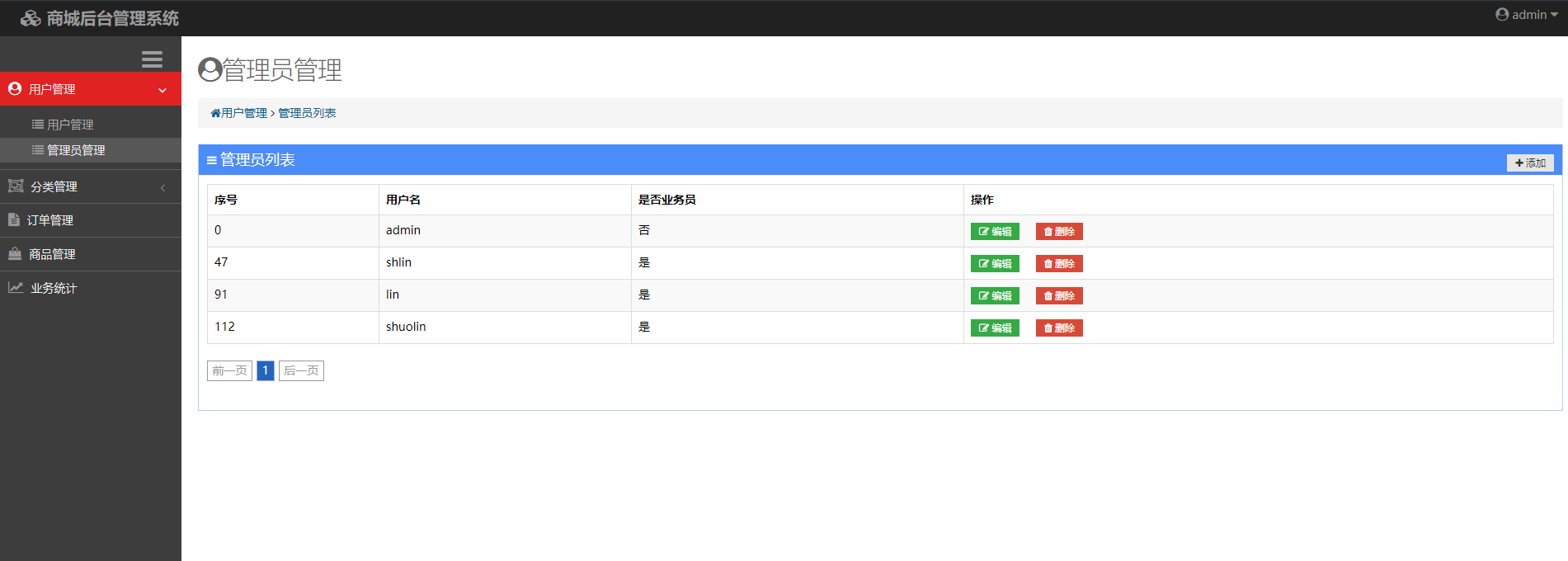


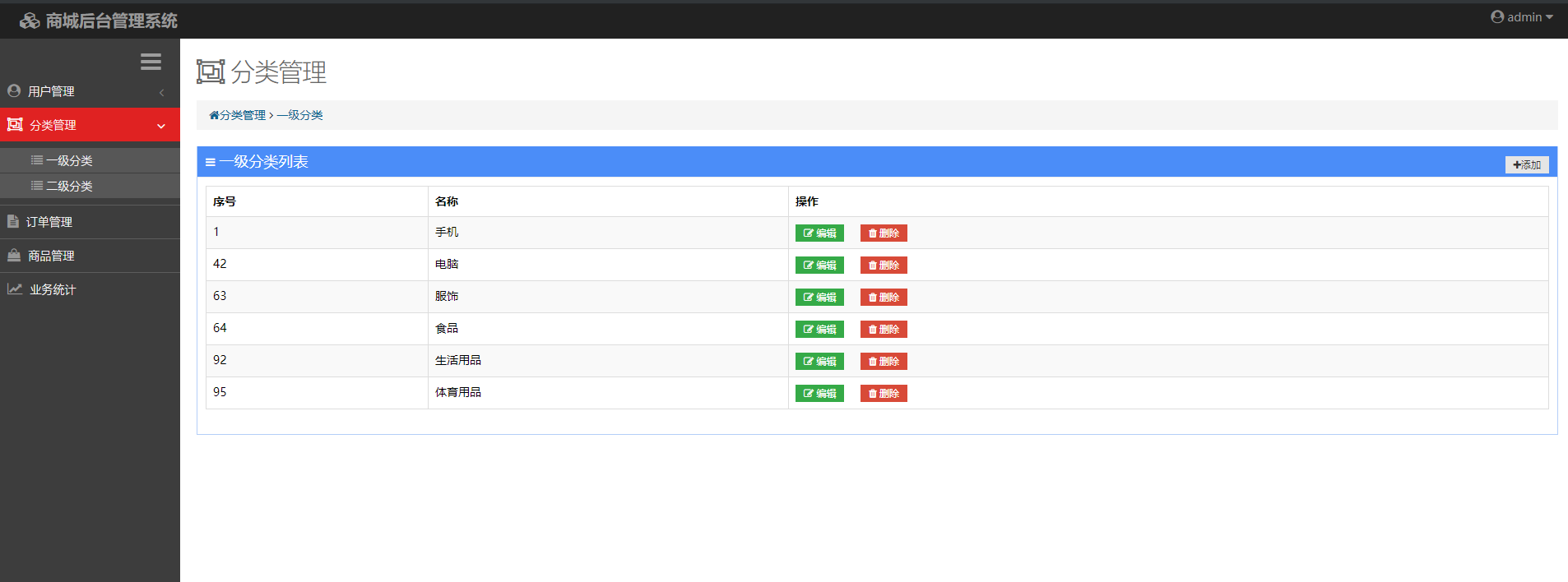
1. 点击查看详情，查看订单所购商品的信息



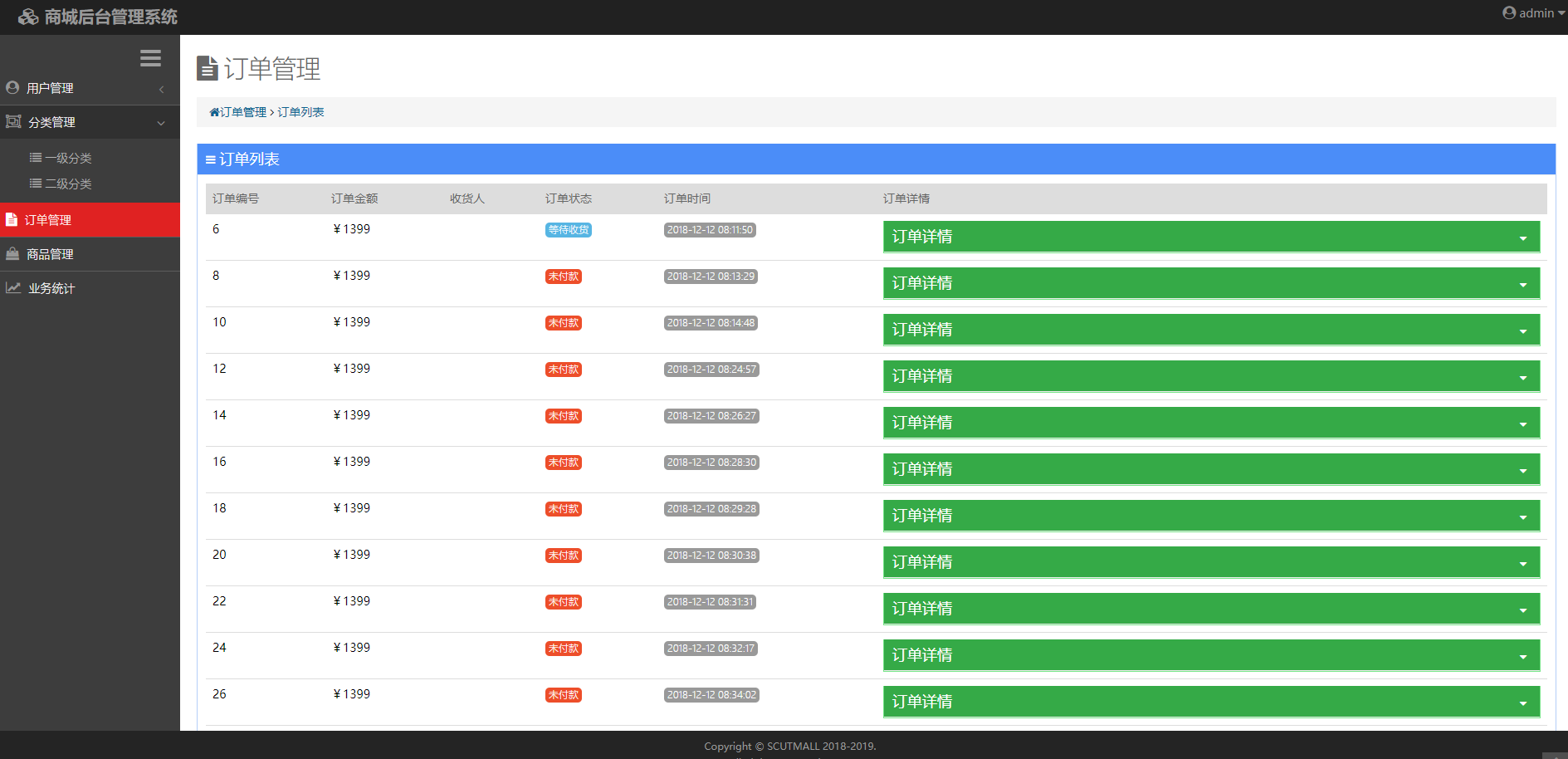
11.后台管理系统













### 3.2.2 系统外部接口设计

与外部系统的交互设计，无。

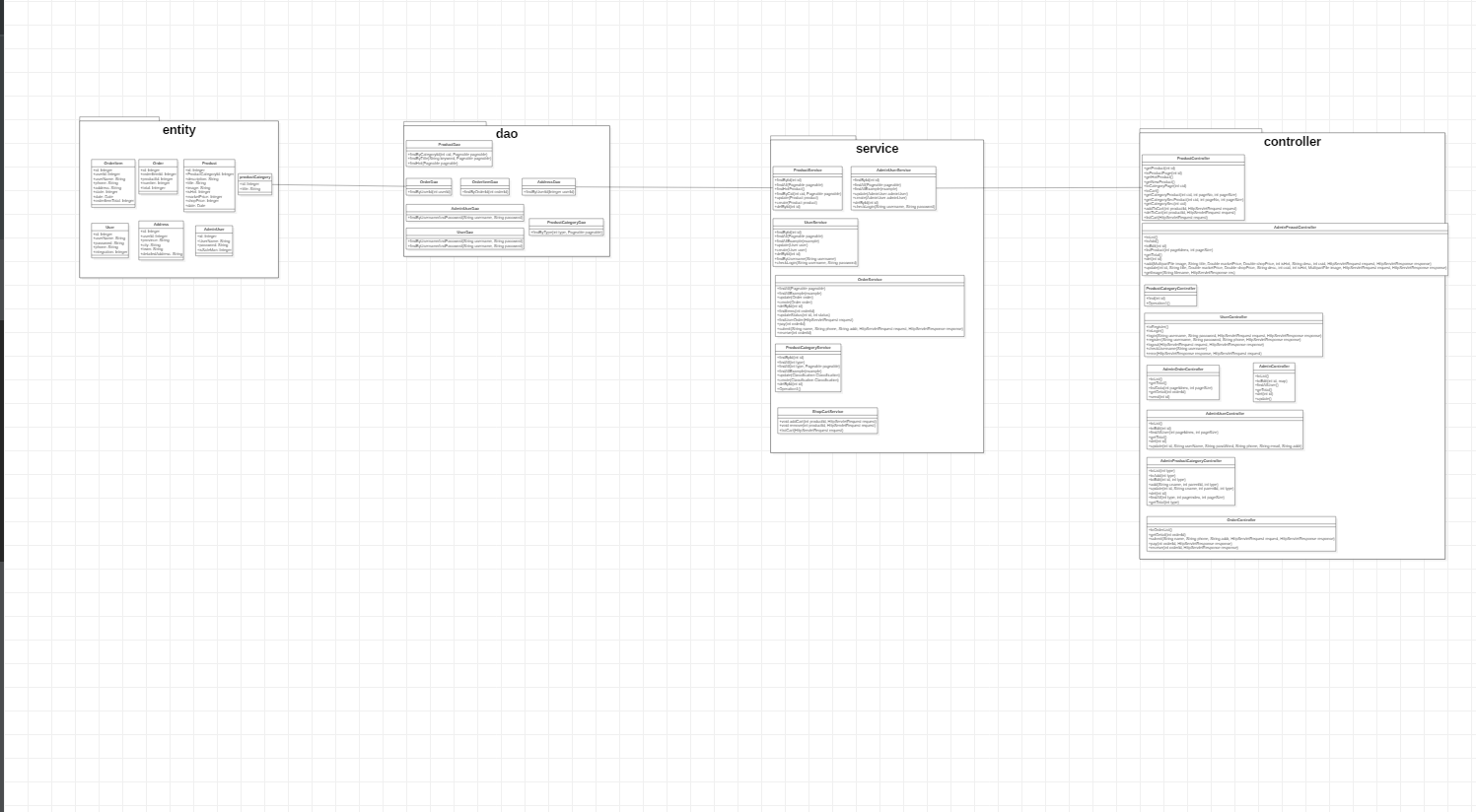
### 3.2.3 系统内部接口设计

各子系统、各模块间的接口设计

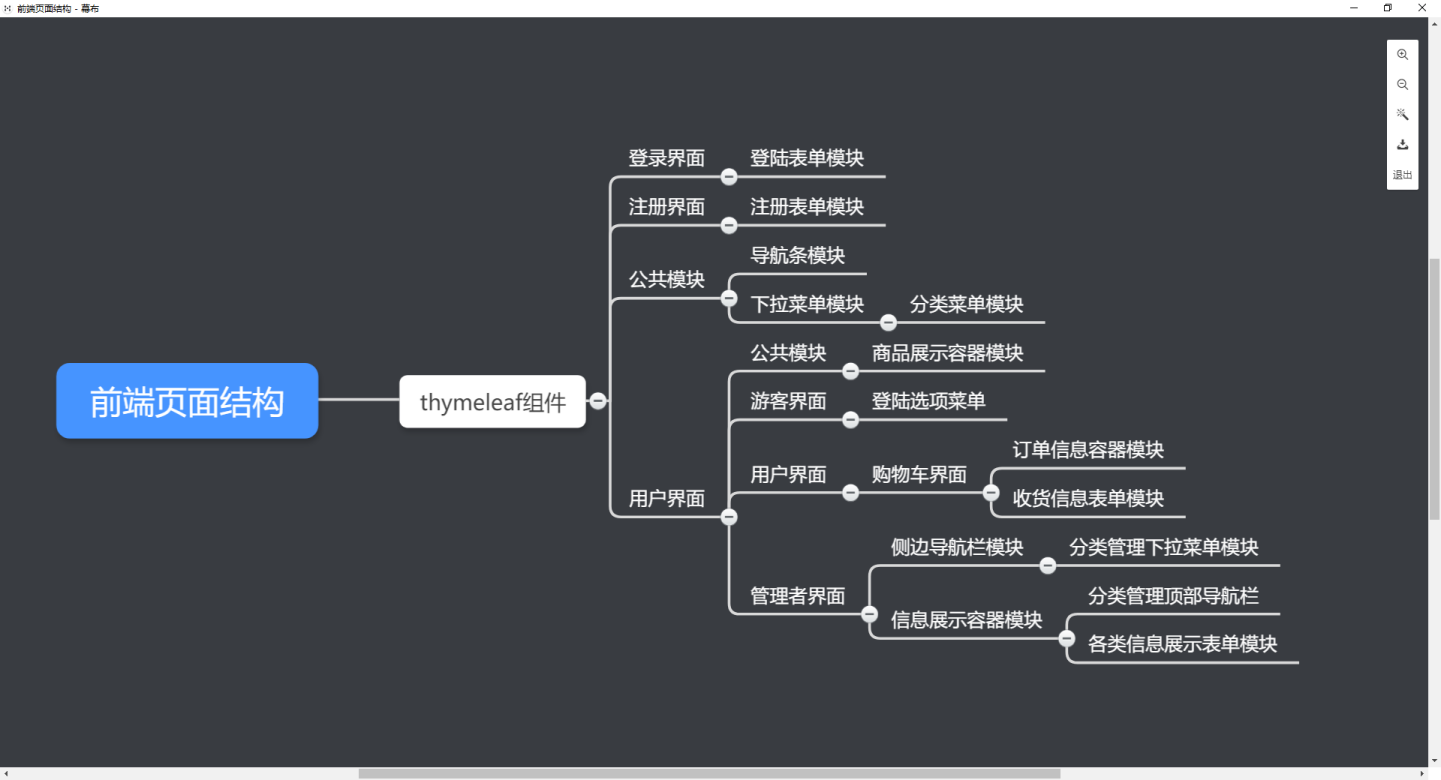
首先，系统运用的是JavaEE+MySQL+js的结构，结构上分为web前端模块包括html、css和js模块，后端模块采用MVC设计模式。前后端的连接交互采用ajax。

## 3.3 对象模型设计

### **3.3.1 对象模型**



### 3.3.2 前端页面结构



### 3.3.3 对象描述

(均省略get set方法)

* **lobalExceptionHandler：用于统一的错误处理**

属性：serialVersionUID：数据类型: long, 作用：序列化版本id

LOGGER: 数据类型：Logger，作用：用于记录错误时间处理日志

方法: ResultBean<String> defaultErrorHandler(HttpServletRequest req, Exception e)： 记录下错误信息和造成错误信息的访问源

void runtimeExceptionHandler(HttpServletRequest req, HttpServletResponse res, Exception e)：处理运行时错误

ResultBean<String> validationExceptionHandler(HttpServletRequest req, ConstraintViolationException e)：处理验证错误

* **WebLogAspect: 用于记录请求信息**

属性：LOGGER：数据类型：Logger，作用：用于记录请求信息

startTime：数据类型：ThreadLocal<Long>，作用：用于记录请求开始时间

方法：void doBefore(JoinPoint joinPoint)：接受到请求处理前记录请求内容

void doAfterReturning(Object ret)：处理完请求后记录下返回的内容

* **ResultBean<T>：返回数据类型的接口**

属性：serialVersionUID：数据类型：long，作用：序列化版本号

SUCCESS 数据类型：int，作用：定义成功代号为0

FAIL 数据类型：int，作用：定义失败代号为1

NO\_LOGIN\_MSG：数据类型：String，作用：定义未登录返回信息

NO\_PERMISSION\_MSG：数据类型：String，作用：定义无权限返回信息

SUCC\_MSG：数据类型：String，作用：定义成功返回信息

FAIL\_MSG：数据类型：String，作用：定义失败返回信息

message：数据类型：String，作用：返回信息

state：数据类型：int， 作用：当前状态

data：数据类型：T，作用：返回的数据

* **实体类共同具有方法与属性：**

属性：serialVersionUID：数据类型：long，作用：序列化版本号

id：数据类型：Integer，作用：实体id号

方法：boolean equals(Object that)：用于对象之间进行比较，重载Object的equals函数

int hashCode()：获取该实体对象的hash值，先用质数31+id的hash值，在用31乘以这个结果，在+该实体类中另一个属性的hash值，依次处理完整个属性

String toString()：将该实体对象信息以json字符串形式给出

* **service接口(除了购物车服务外)共同具有方法：用于访问数据库**

方法：findById(int id)：通过实体类对象id查询并返回要查询的实体类对象，

findAll(Pageable pageable)：分页查询所有的对应的实体类对象

List<T> findAllExample(Example<T> example)：按条件查询对应的实体类对象

update(T t)：更新该对象数据

create(T t)：将对象添加至数据库

delById(int id)：按照id号删除从数据库中删除该对象

* **AdminUserService：用于访问数据库管理者表的接口**

特有方法：AdminUser checkLogin(HttpServletRequest request,String username, String pwd)：验证登陆并返回该管理者在数据库的信息

* **ClassificationService：用于访问数据库商品种类表的接口**

特有方法：List<Classification> findByParentId(int cid)：利用该品类的上一级分类id查询商品并返回列表

* **UserService：用于访问数据库用户表的接口**

特有方法：List<User> findByUsername(String username)：通过用户名查询并返回用户数据列表

* **OrderService：用于访问数据库用户表的接口**

特有属性：STATE\_NO\_PAY：数据类型：int，作用：定义未支付状态代号为1

STATE\_WAITE\_SEND：数据类型：int，作用：定义等待发货状态代号为2

STATE\_WAITE\_RECEIVE：数据类型：int，作用：定义等待签收状态代号为3

STATE\_COMPLETE：数据类型：int，作用：定义订单完成状态为4

特有方法：List<OrderItem> findItems(int orderId)：通过订单项id查询到该笔订单中特定的订单项信息

pay(int orderId)：支付该订单，并更新订单信息状态

updateStatus(int id, int status)：更新订单状态信息

List<Order> findUserOrder(HttpServletRequest request)：查询用户的订单列表

submit(String name, String phone, String addr, HttpServletRequest request, HttpServletResponse response)：确认订单信息

receive(int orderId)：确认收货，更新订单状态信息

* **ProductService：用于访问数据库产品表的接口**

特有方法：List<Product> findHotProduct()：查询热卖商品并返回列表

List<Product> findNewProduct(Pageable pageable)：查询最新商品并返回列表

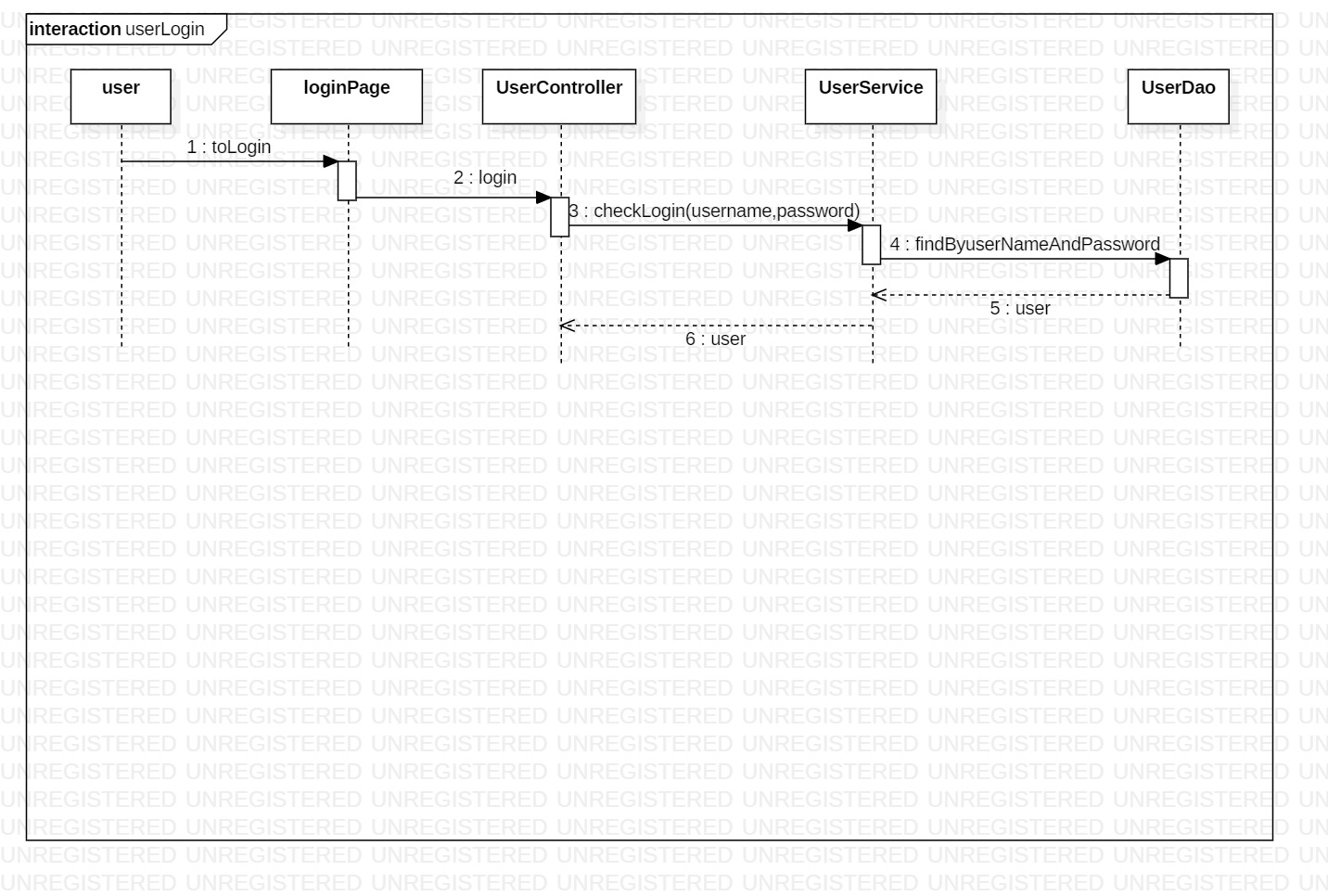
List<Product> findByCid(int cid,Pageable pageable)：利用一级分类id查询商品并返回列表

List<Product> findByCsid(int csid,Pageable pageable)：利用二级分类id查询商品并返回列表

## 系统功能（用例）实现详细设计

### 3.4.1 顺序图

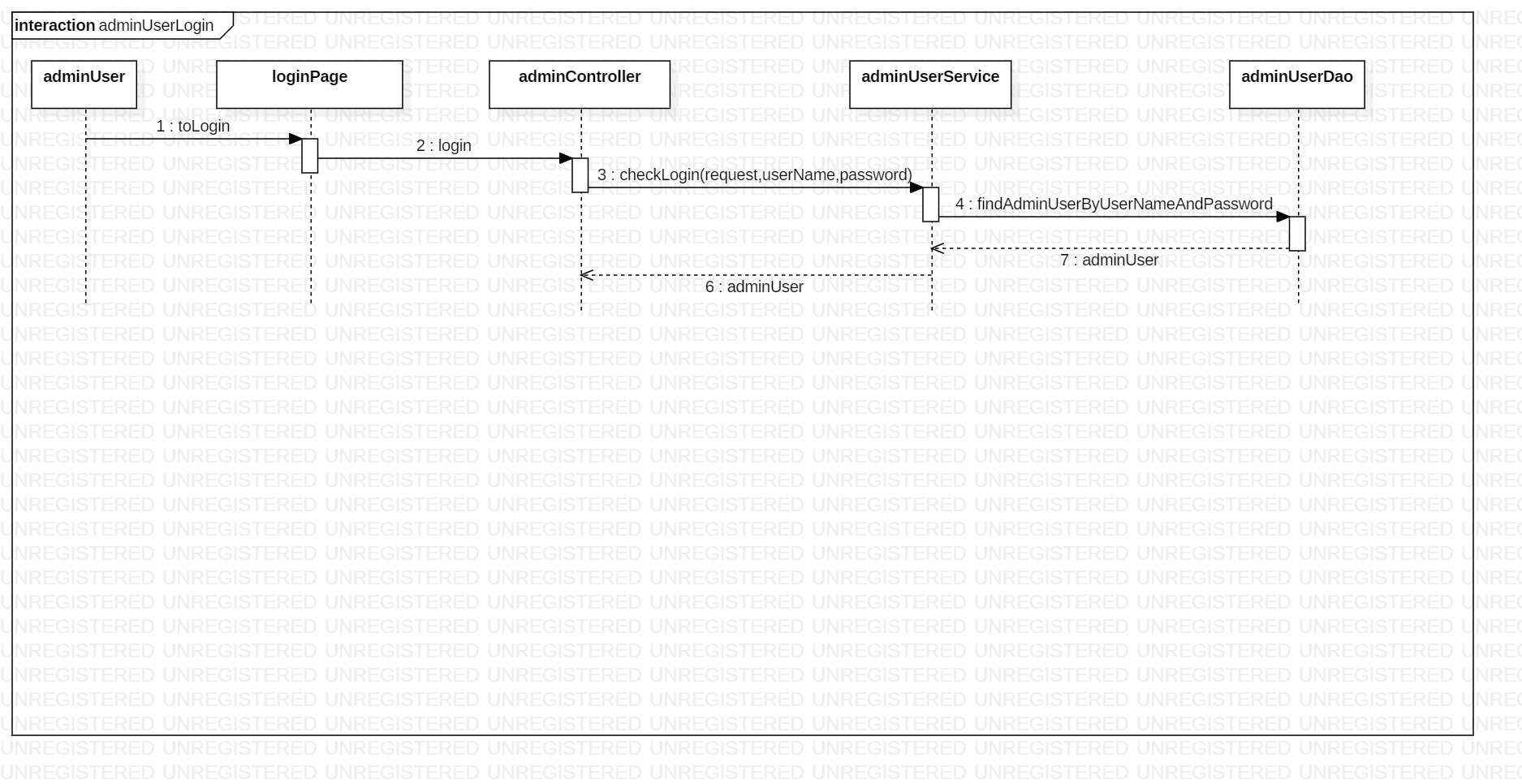
用户登录



备注：

1. login(String userName,String password,HttpServletRequest request,HttpServletResponse response)

2. findByUserNameAndAndPassword(String userName,String password)

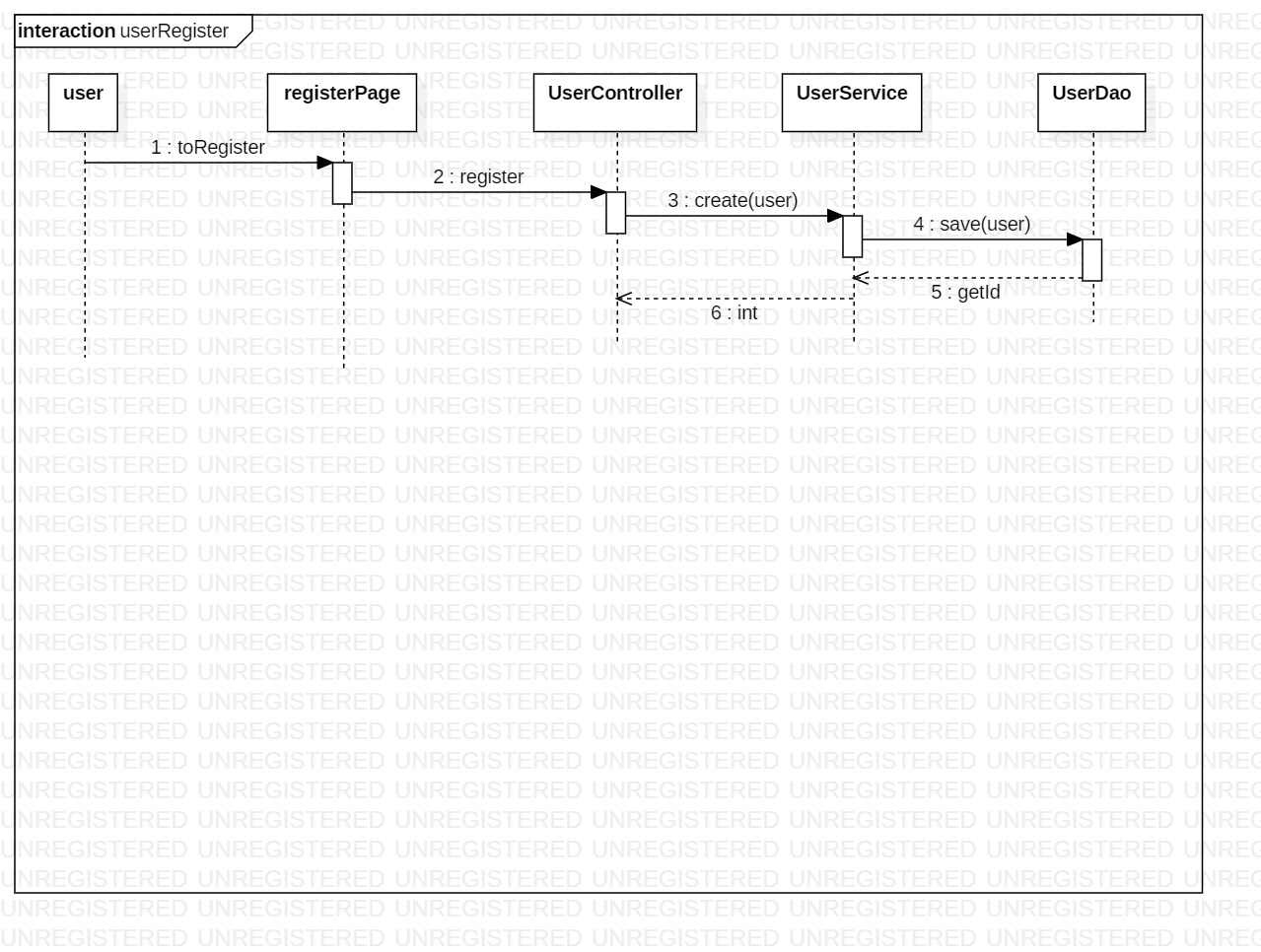
管理员登录  
  


备注：

1. login(String userName,String password,HttpServletRequest request,HttpServletResponse response)

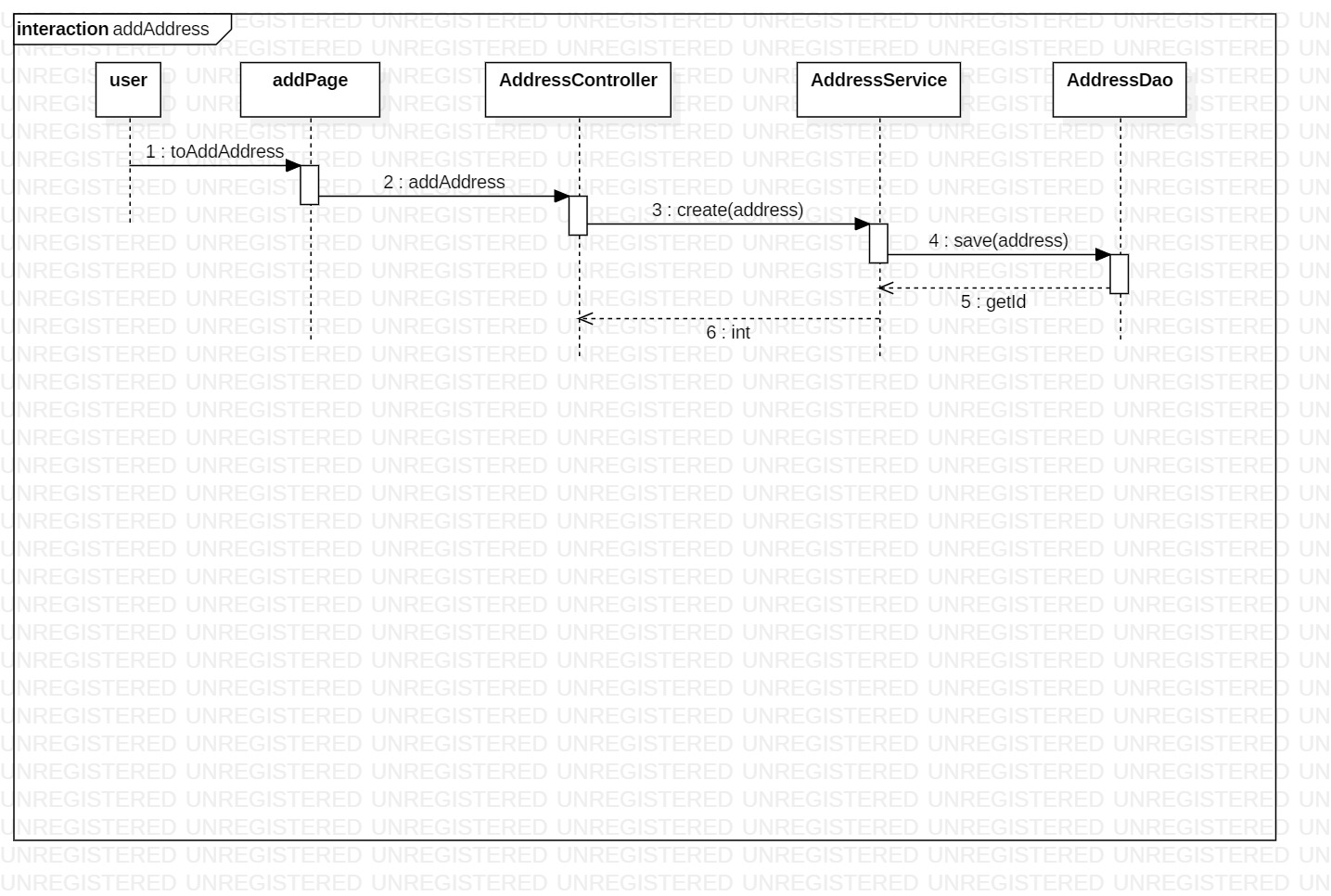
2. findAdminUserByUserNameAndAndPassword(String userName,String password)

用户注册



备注：1. register(String userName,String password,String phone,HttpServletResponse response ,HttpServletRequest request)

1. 用户增加地址



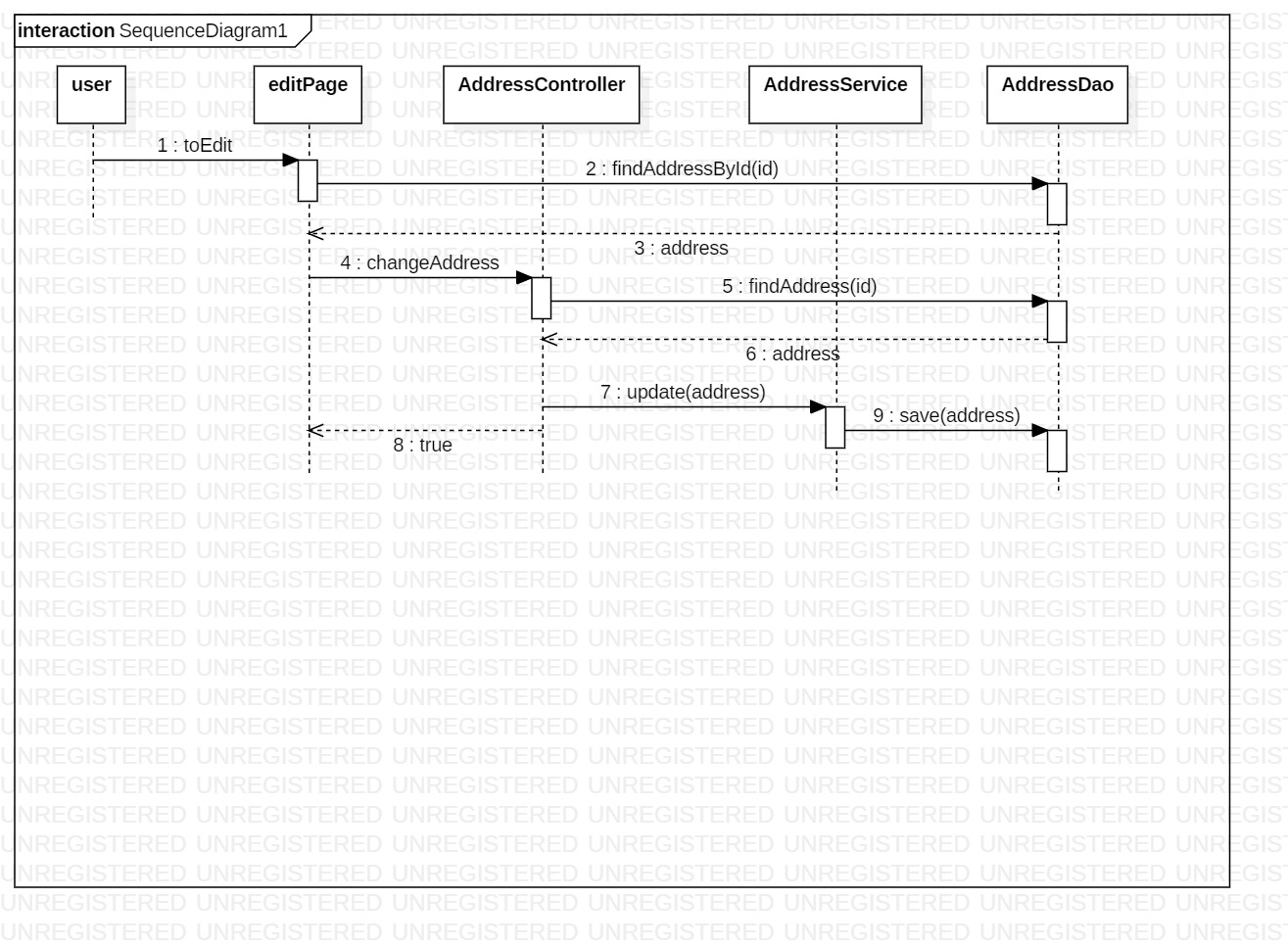
备注：

1.addAddress(String province,String city,String district,String detail,

String receiver,String phone,HttpServletRequest request,

HttpServletResponse response)

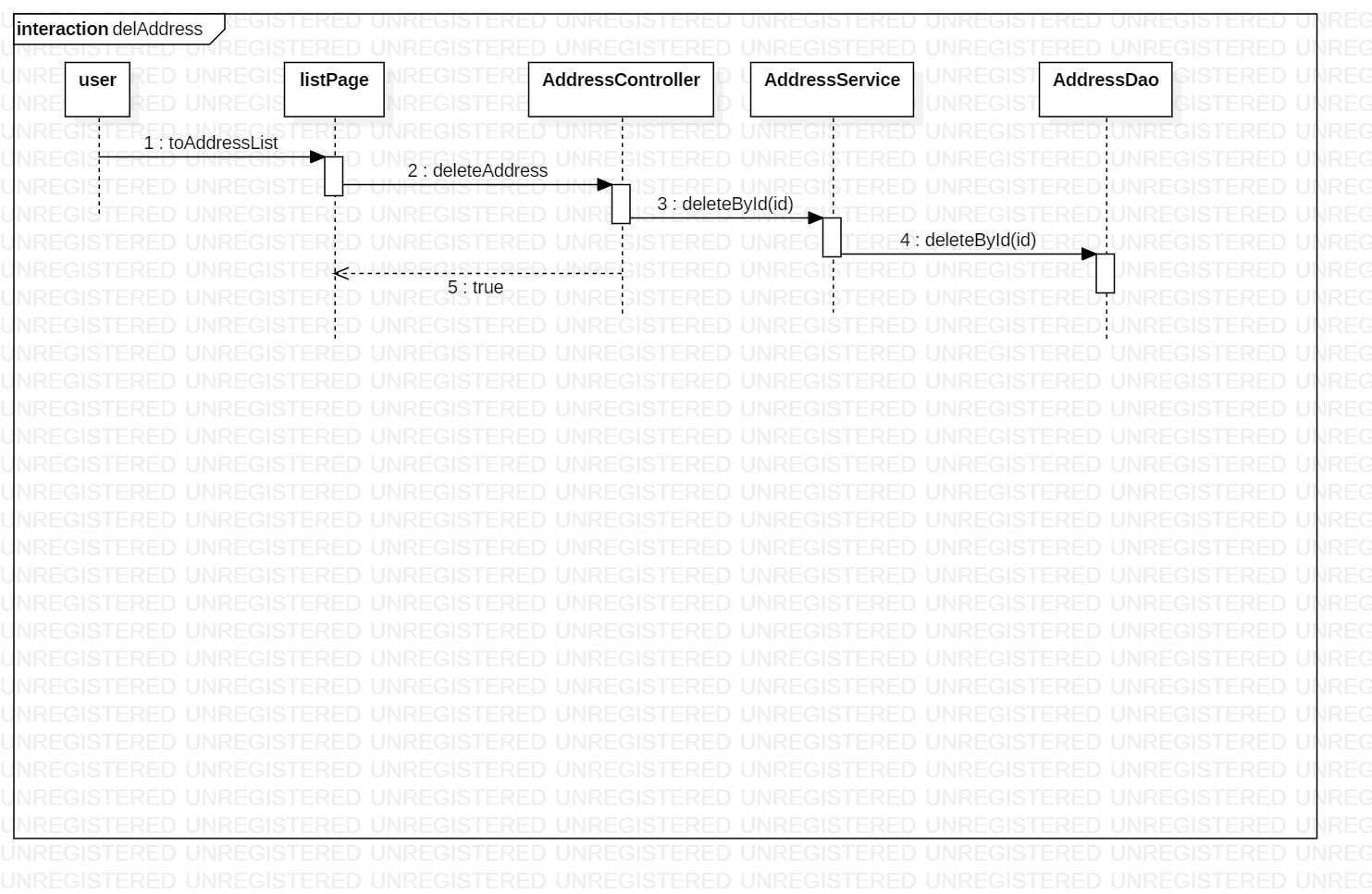
1. 用户编辑地址



备注：

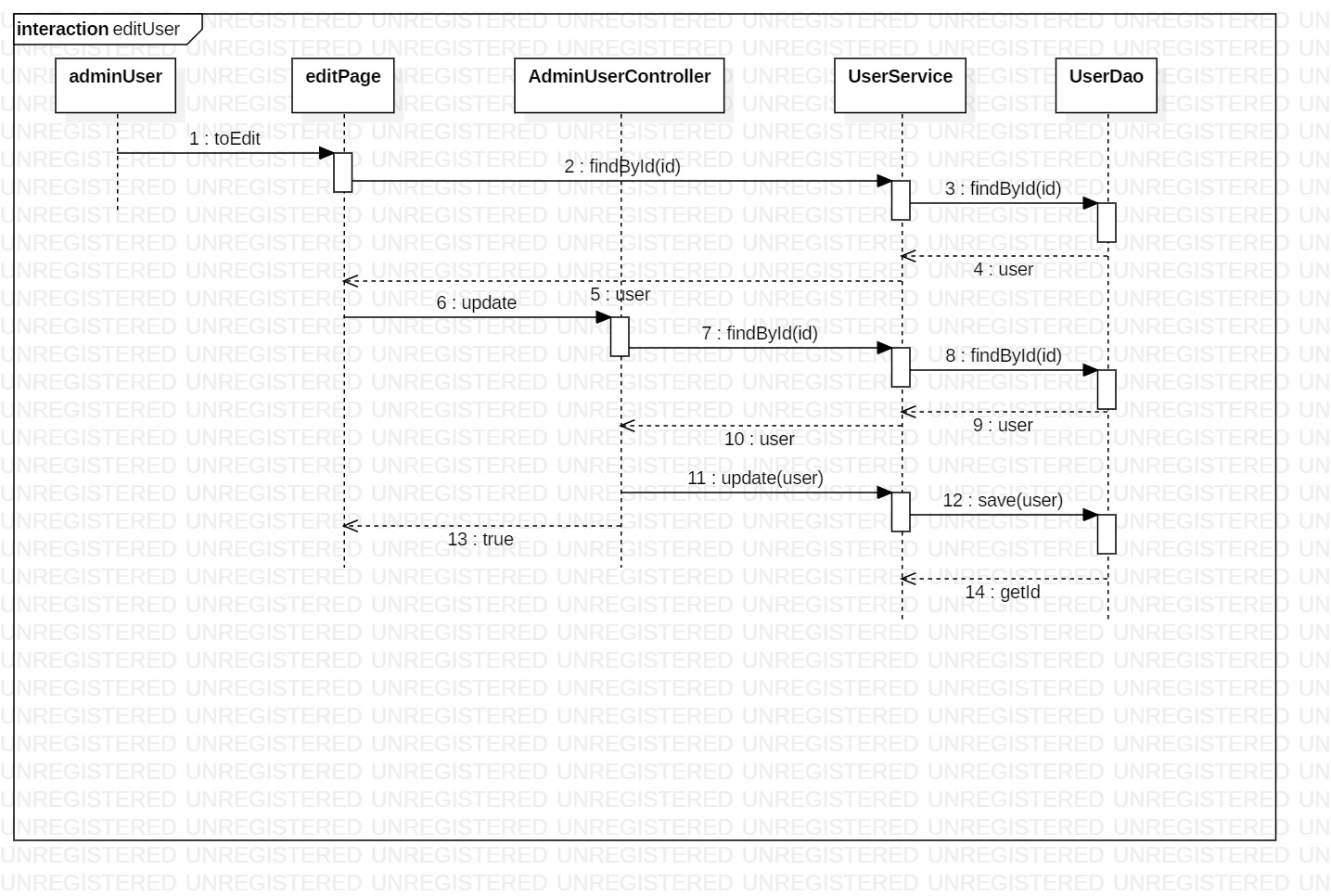
1.changeAddress(Integer id , String province,String city,String district,String detail,String receiver,String phone,HttpServletRequest request, HttpServletResponse response)

用户删除地址



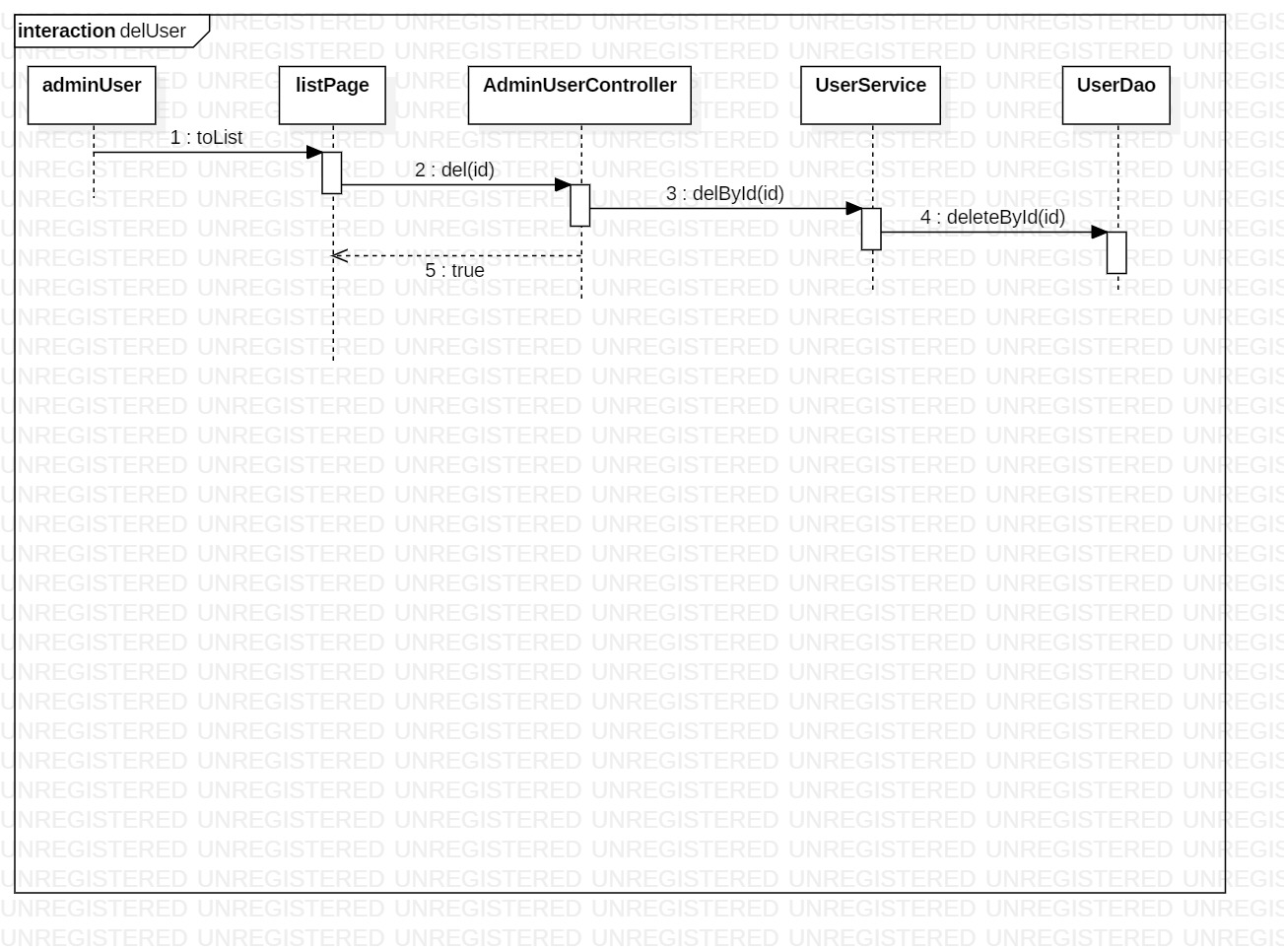
备注：

1. deleteAddress(Integer id,HttpServletResponse response)  
     
     
   管理员删除用户

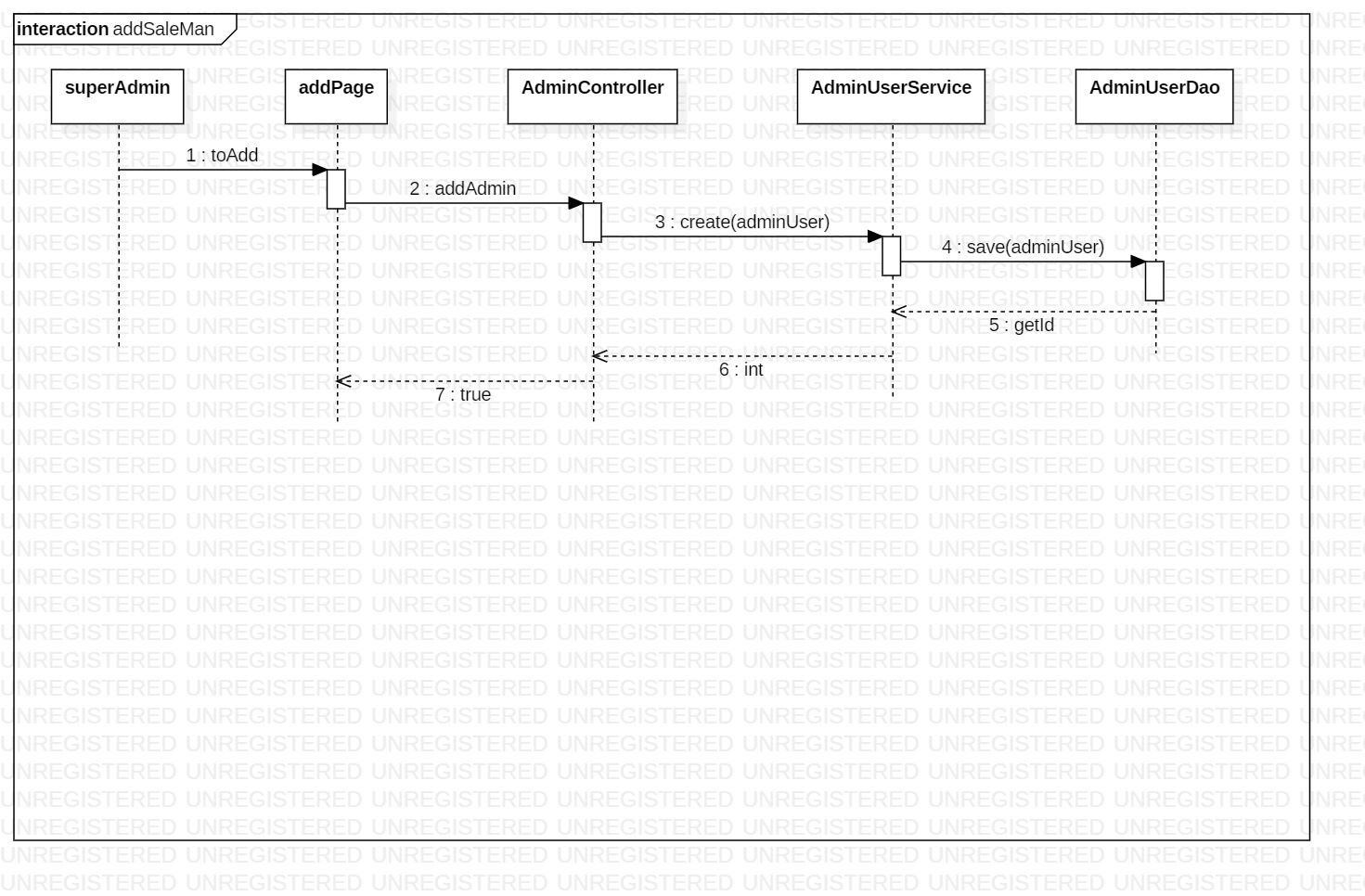


备注

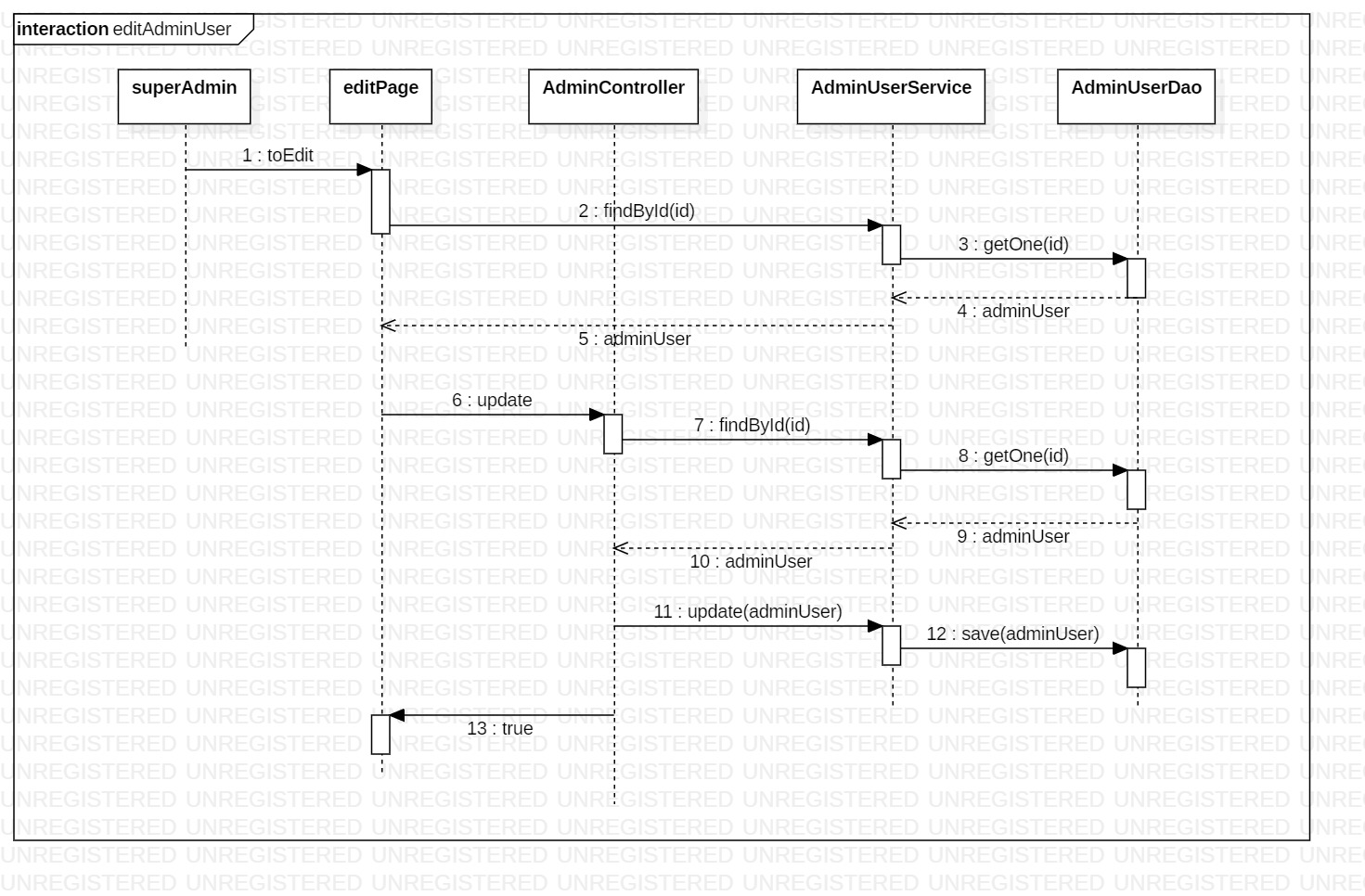
1.update(int id,String userName,String phone,String password,int integration,HttpServletRequest request, HttpServletResponse response)   
  
管理员删除用户



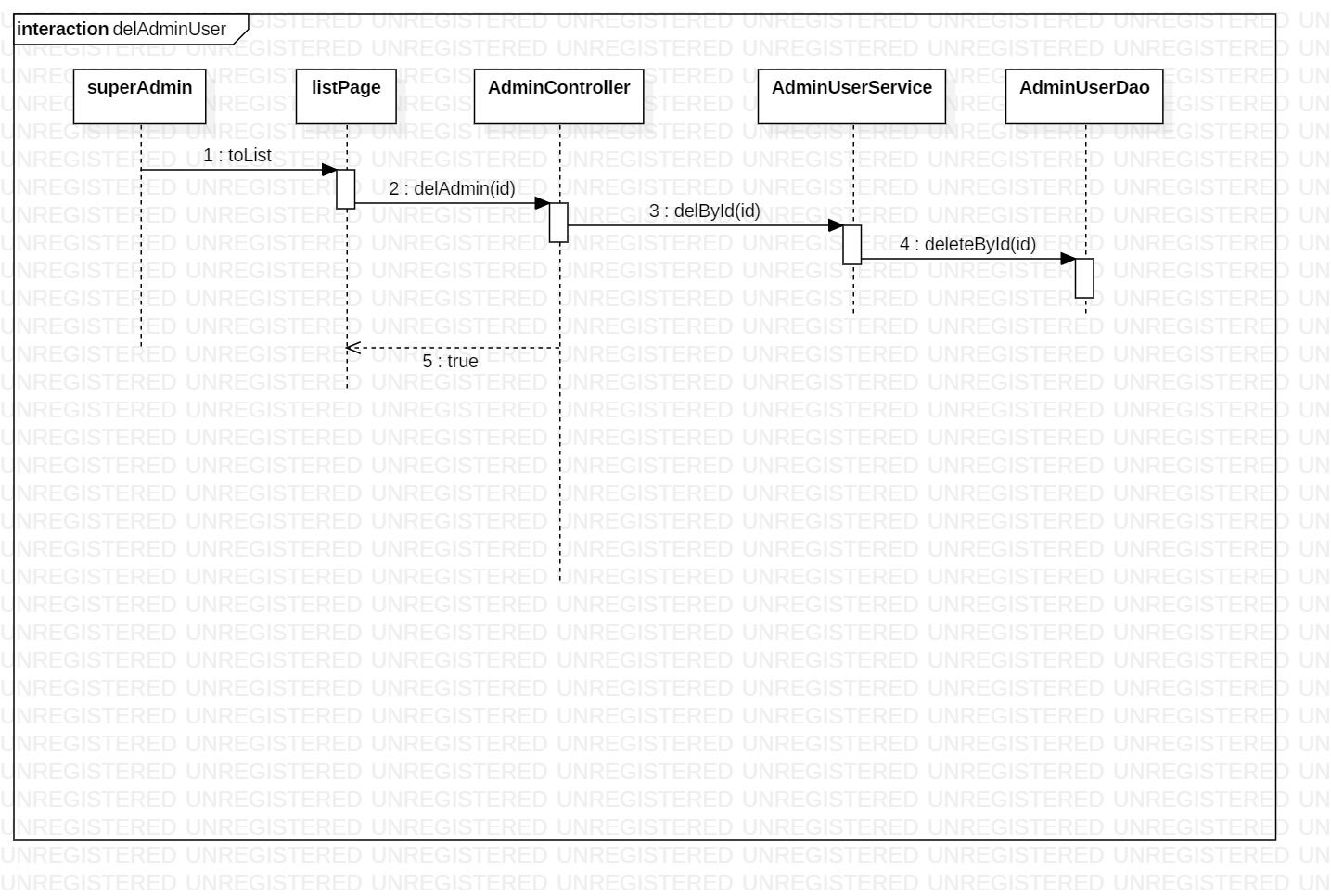
超级管理员增加业务员



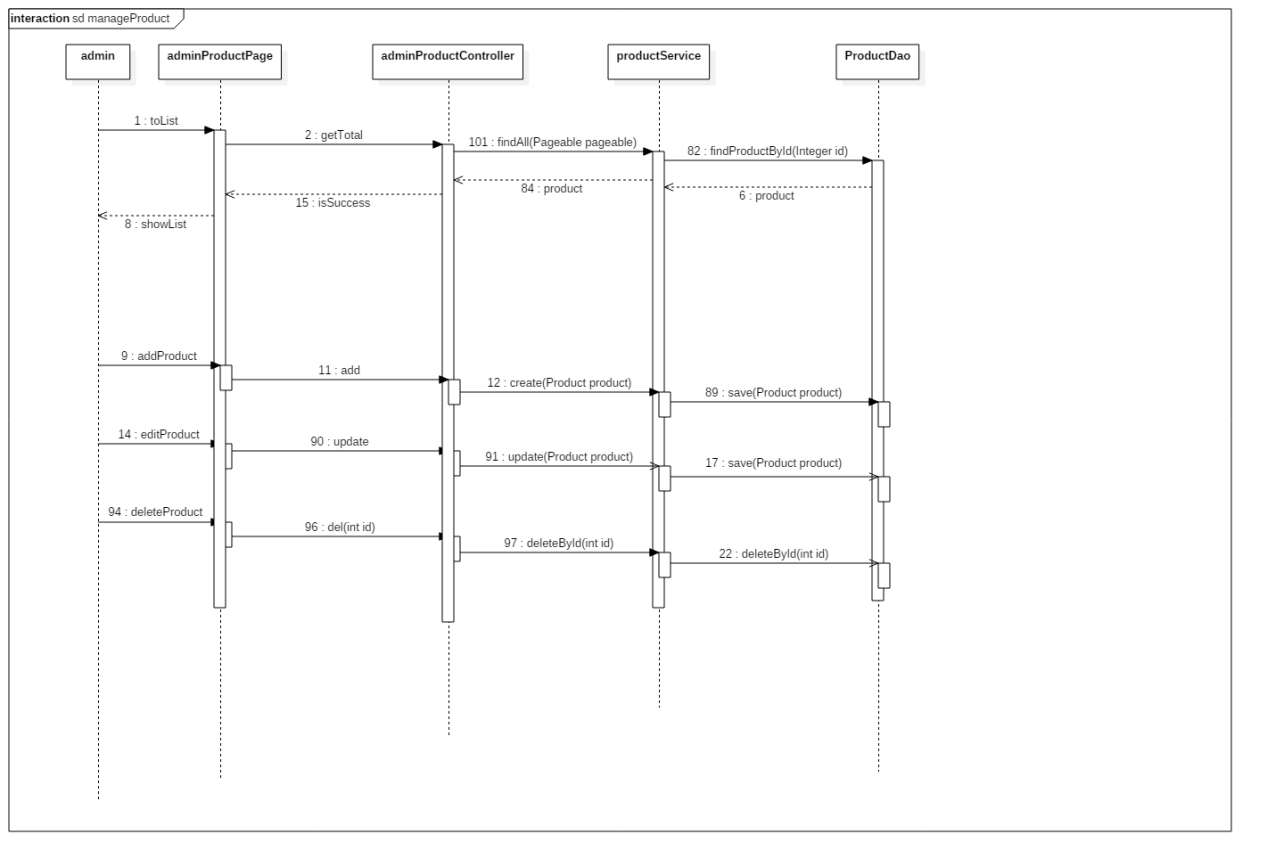
备注：  
1.addAdmin(String userName,String password, HttpServletRequest request,HttpServletResponse response)  
  
超级管理员编辑业务员



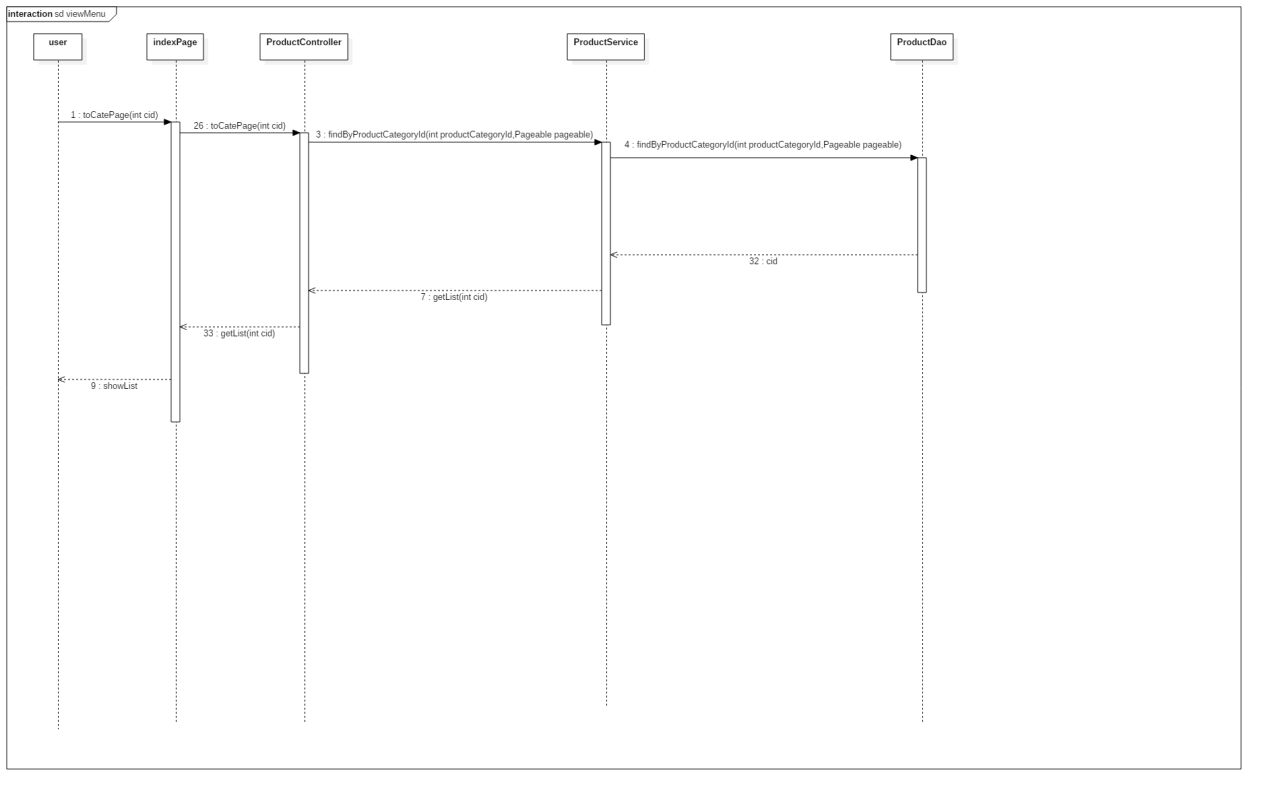
备注：

1. update(int id,String userName,String password, Integer isSaleMan,HttpServletRequest request, HttpServletResponse response)   
     
   超级管理员删除业务员  
   

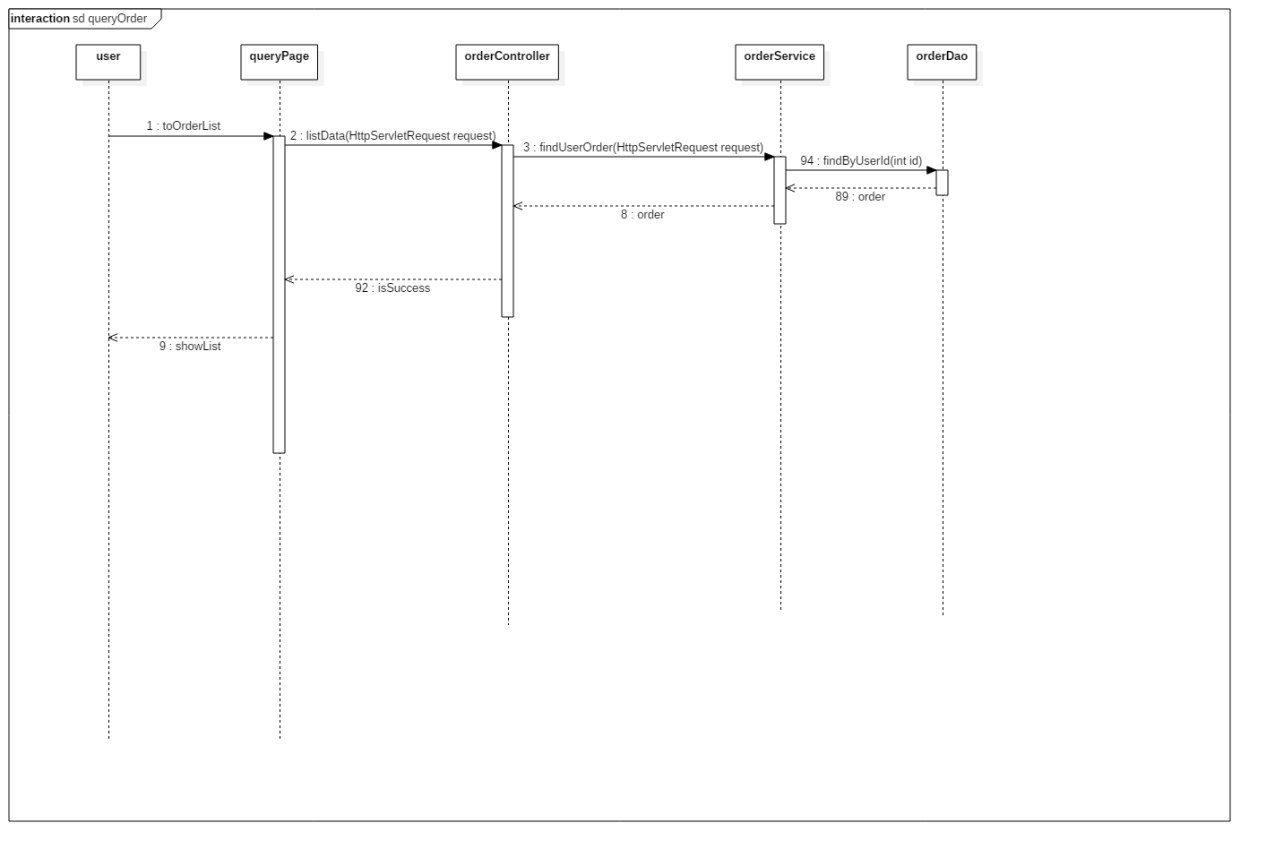
管理商品



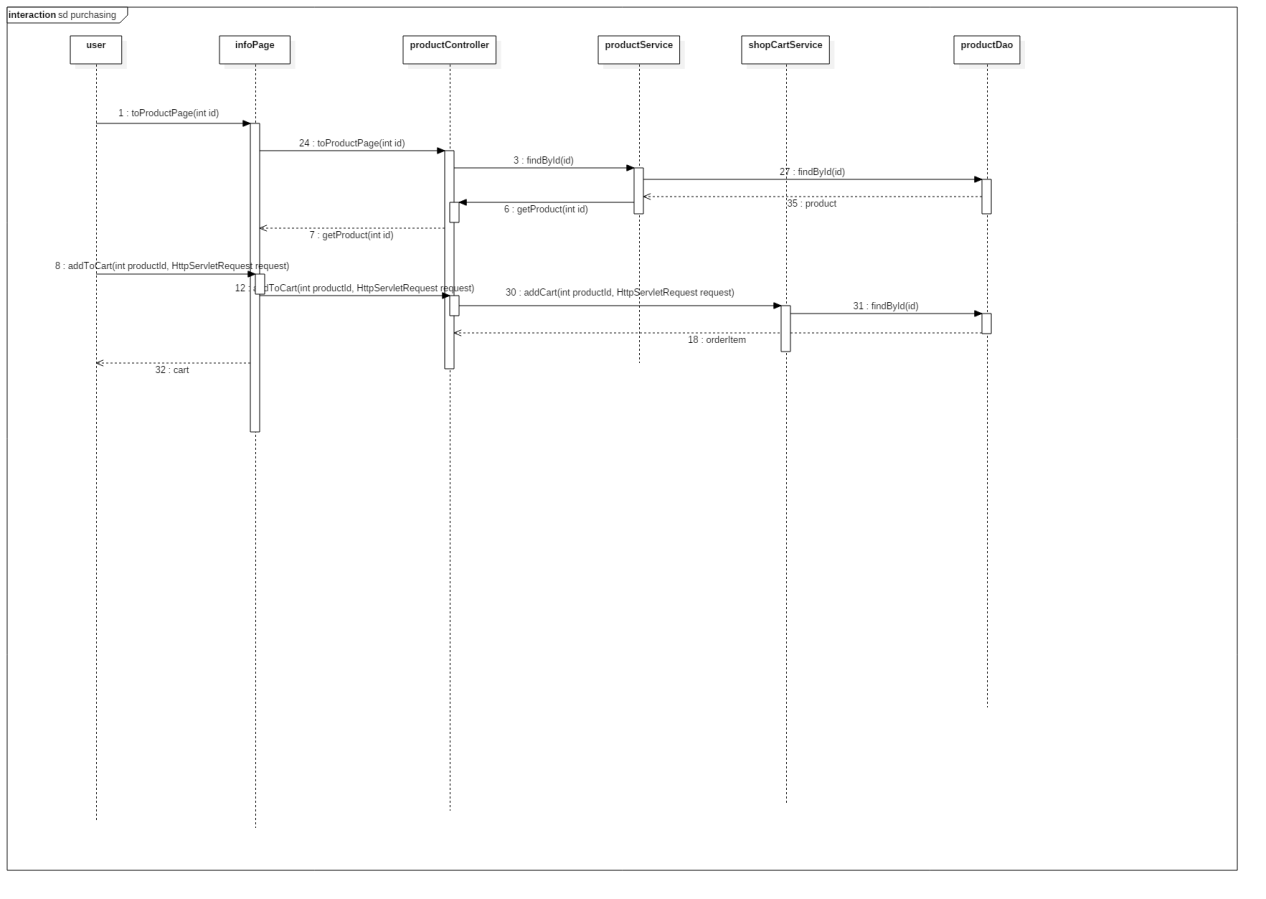
用户查看商品



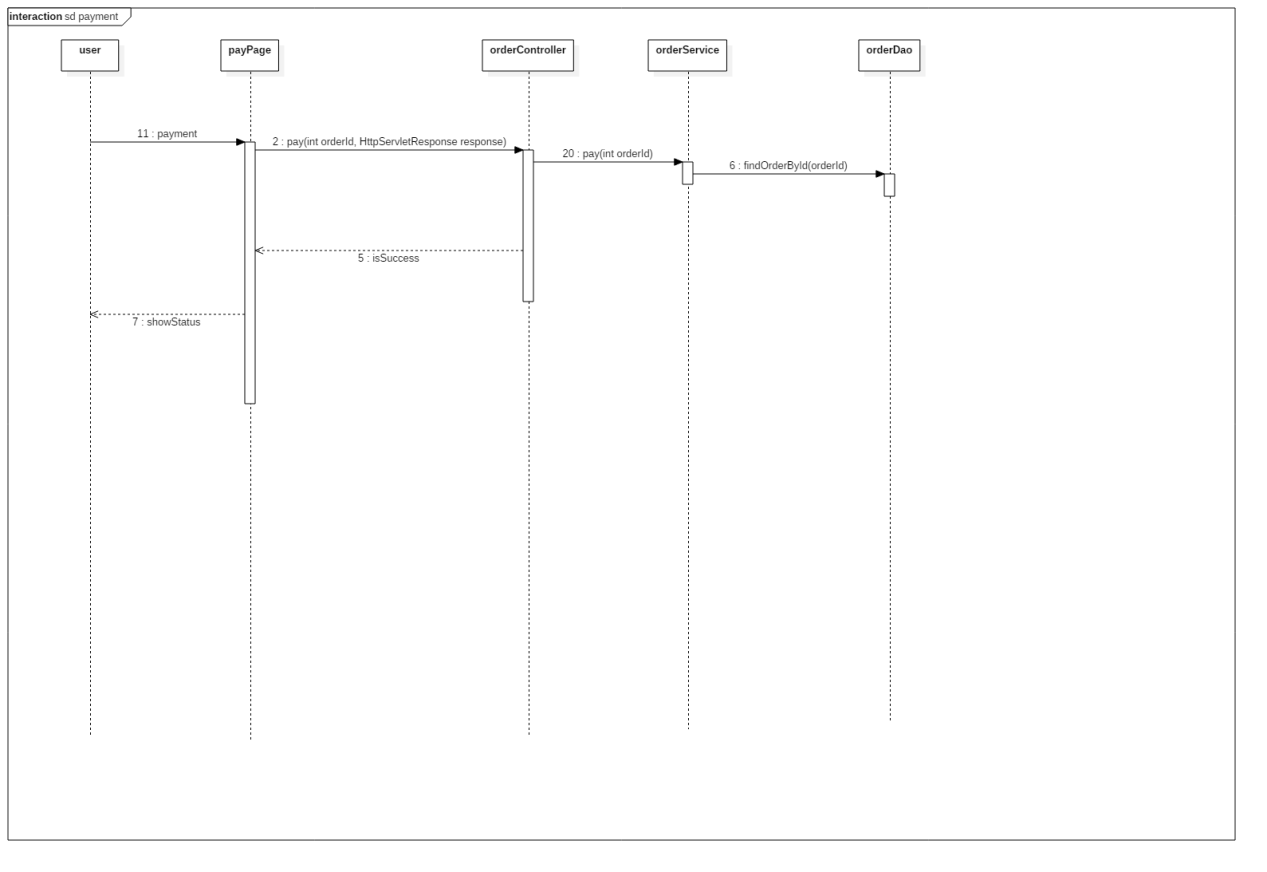
用户查询订单



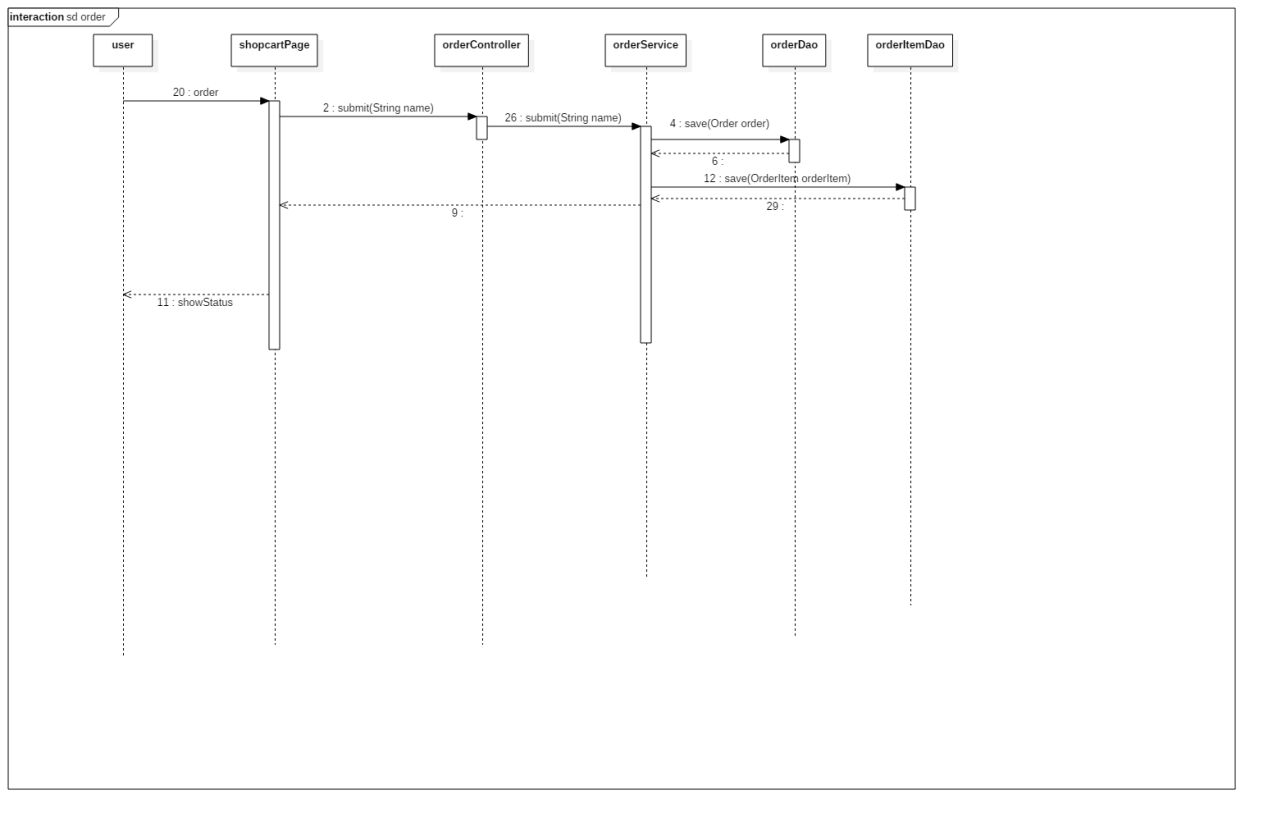
用户加入购物车



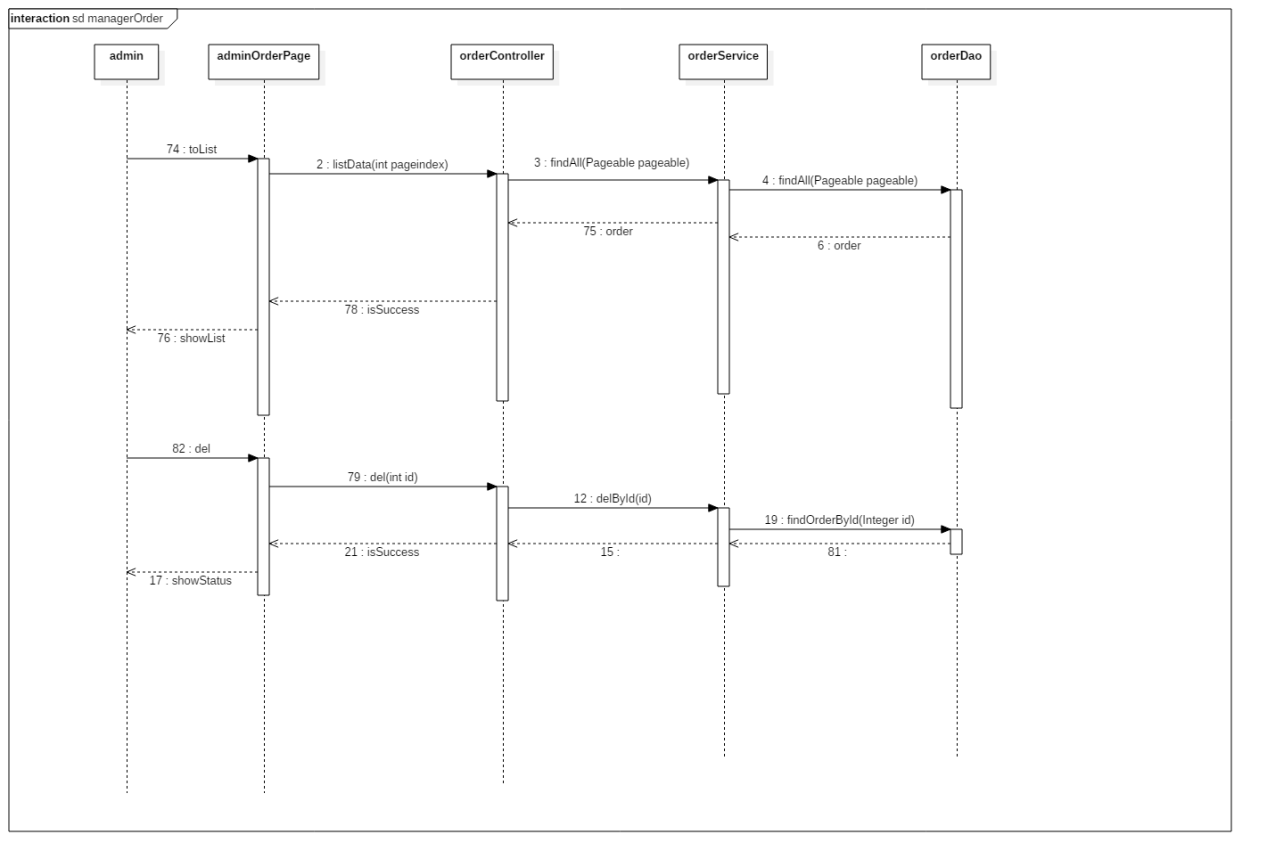
用户订单付款



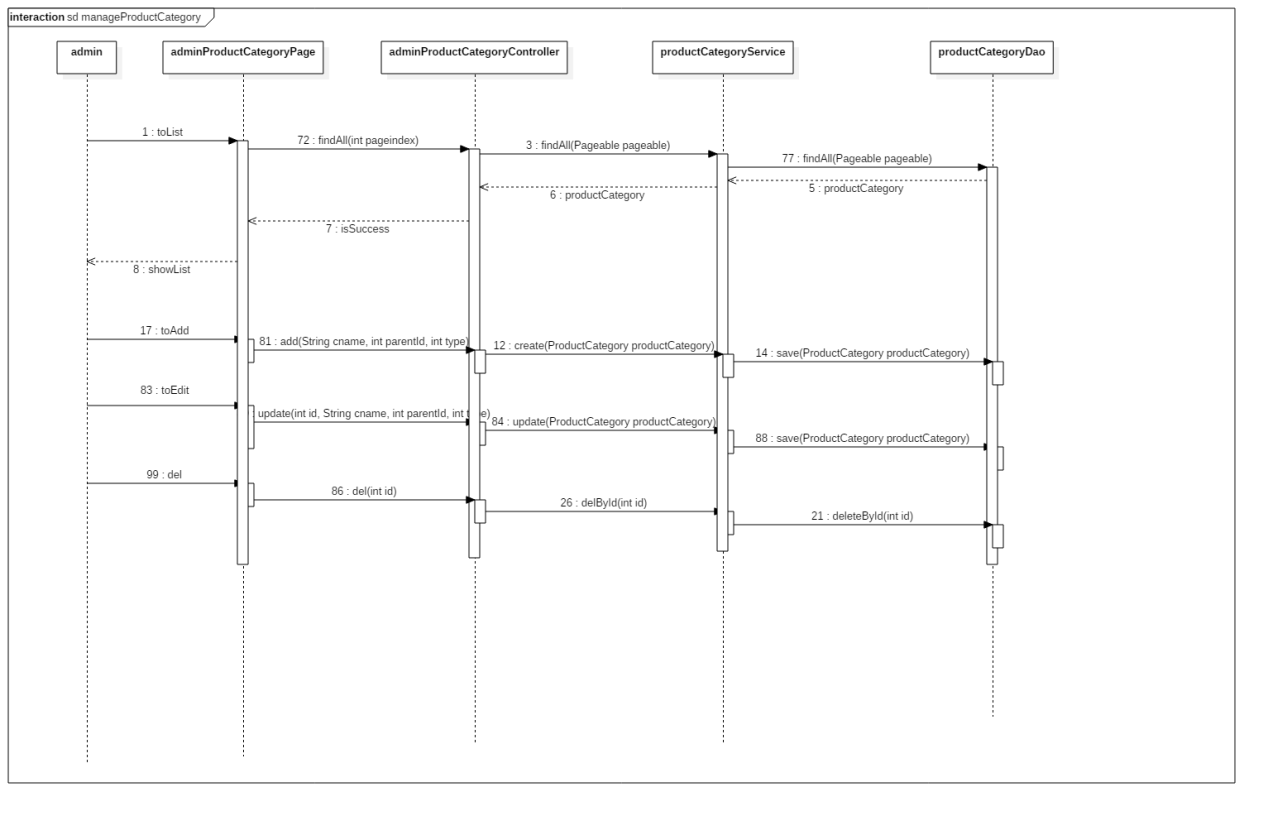
用户下单



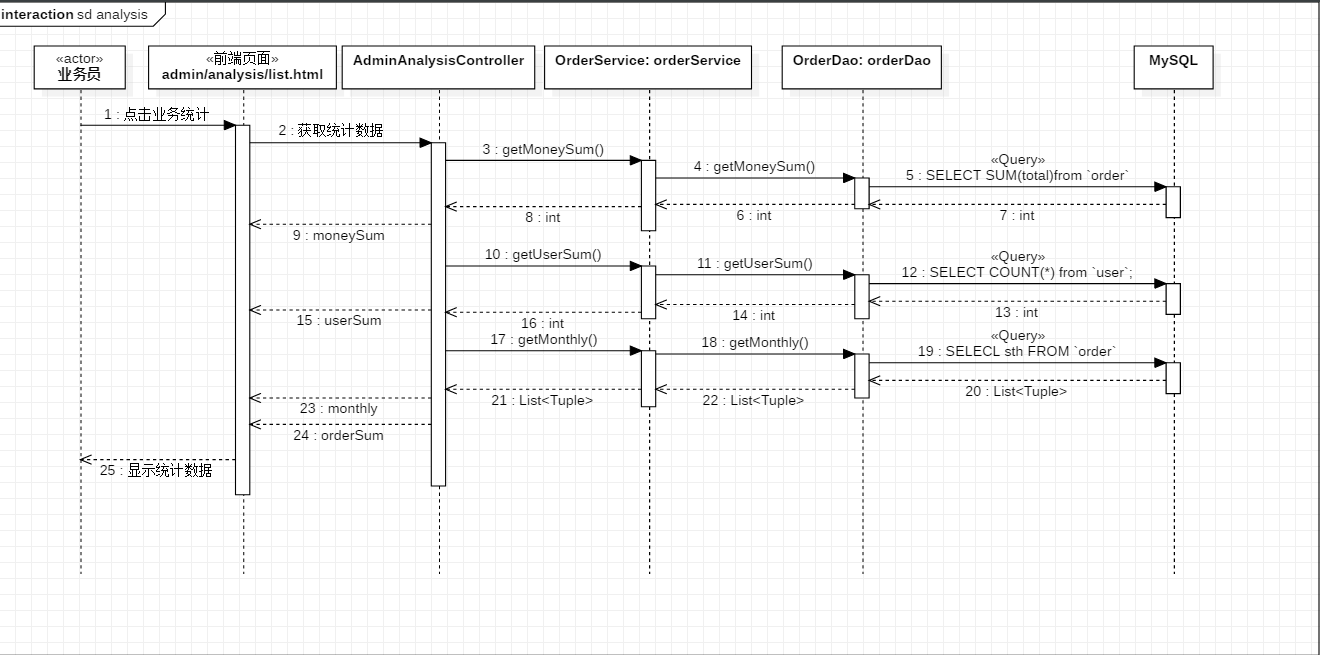
后台管理订单



管理商品分类



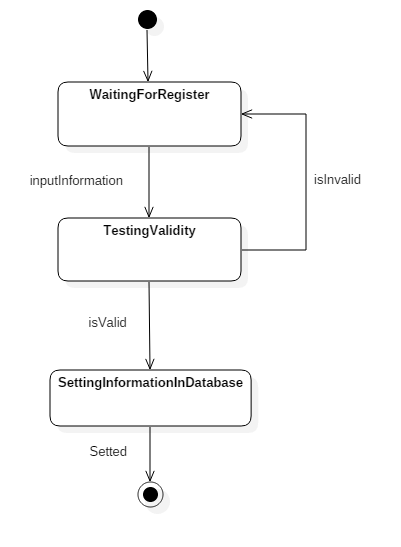
统计业务



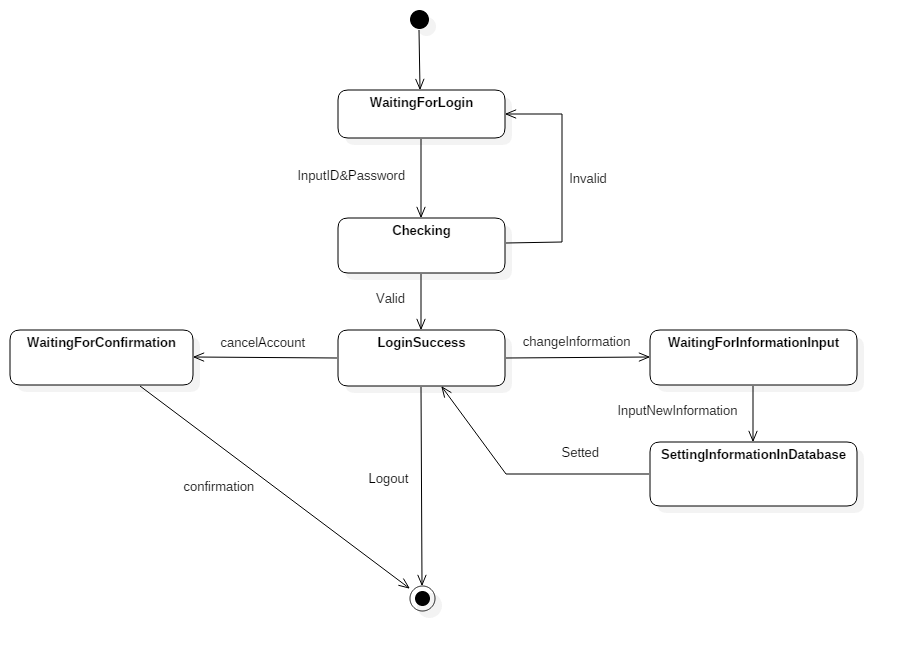
### 3.4.2 状态图

#### 3.4.2.1 用户相关

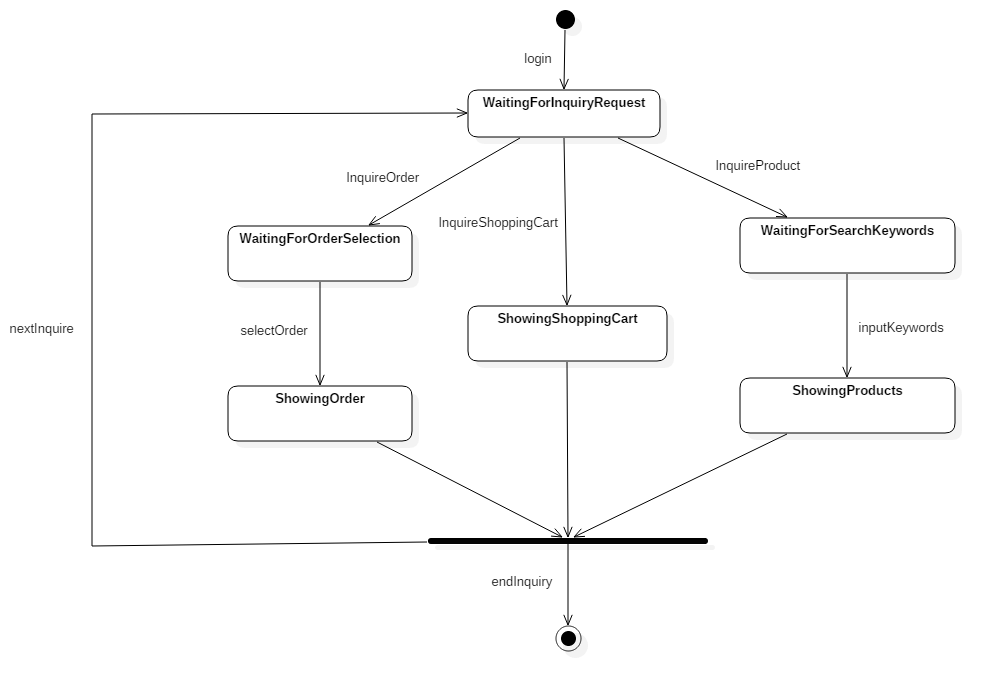
* + **用户\_注册**



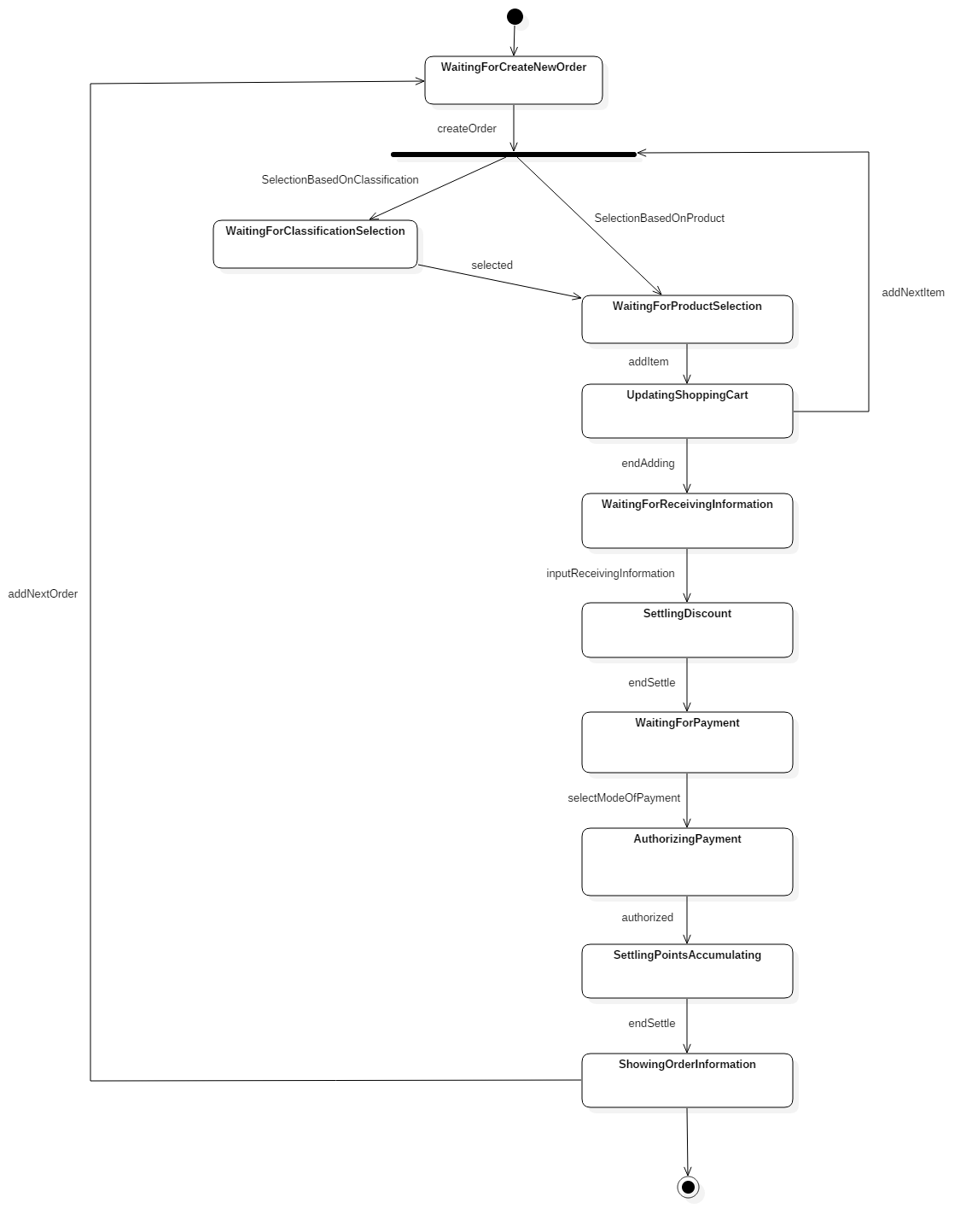
* + **用户\_账号操作**



* + **用户\_查询操作**

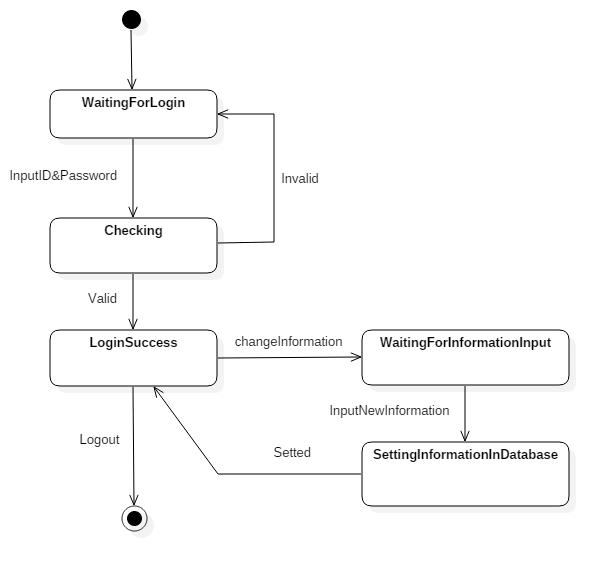


* + **用户\_购物操作**



#### 3.4.2.2 业务员相关

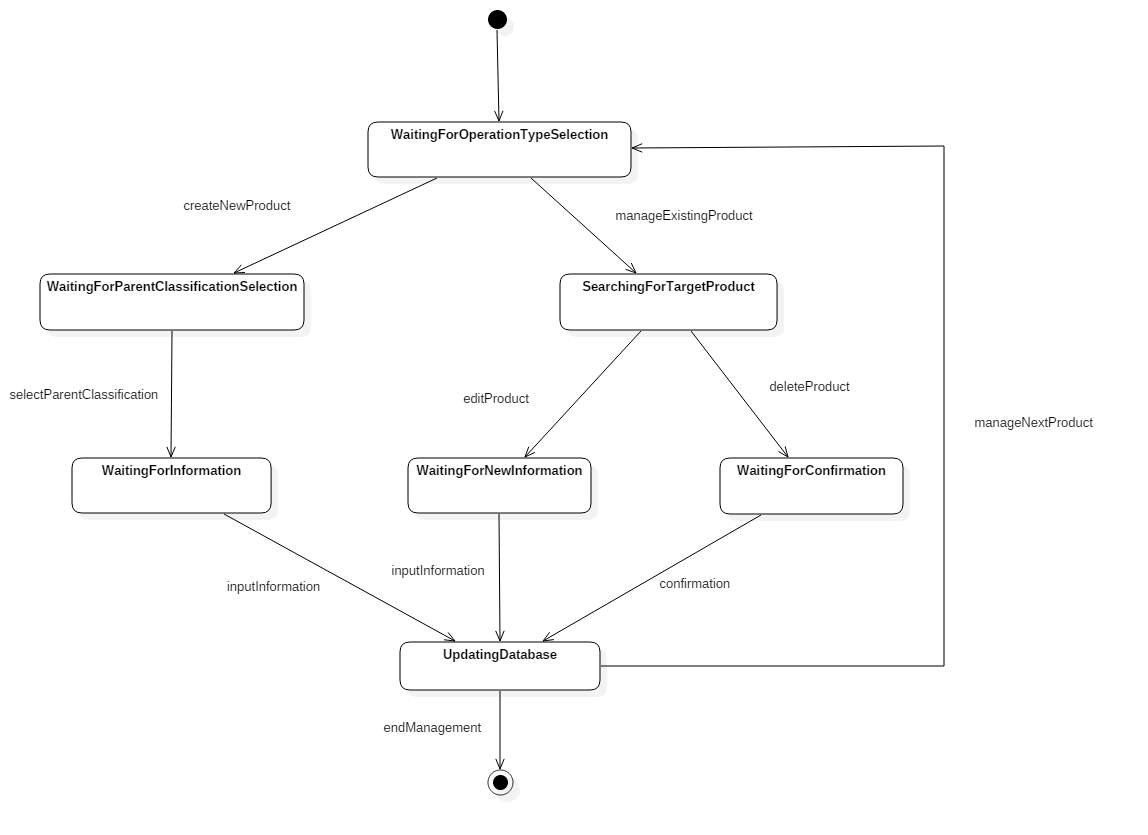
* **业务员\_账号操作**



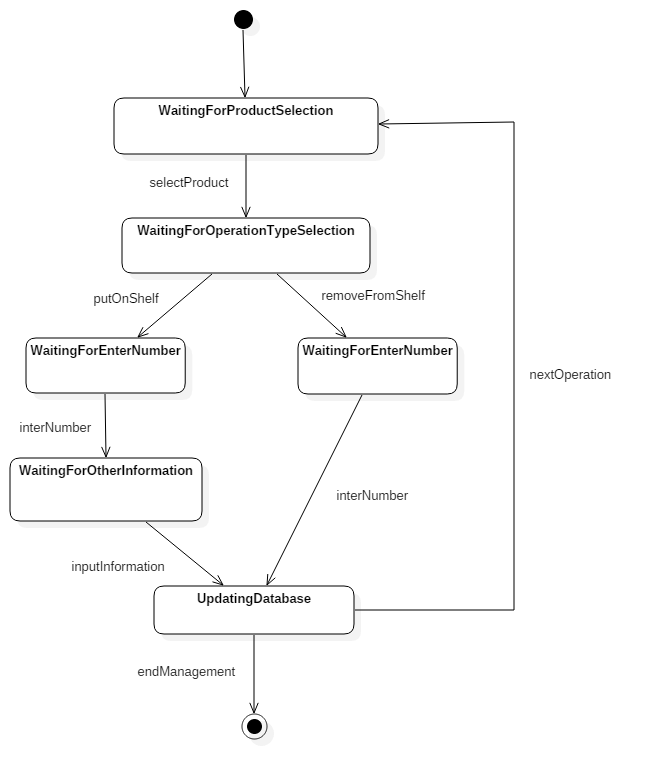
* **业务员\_品类管理**



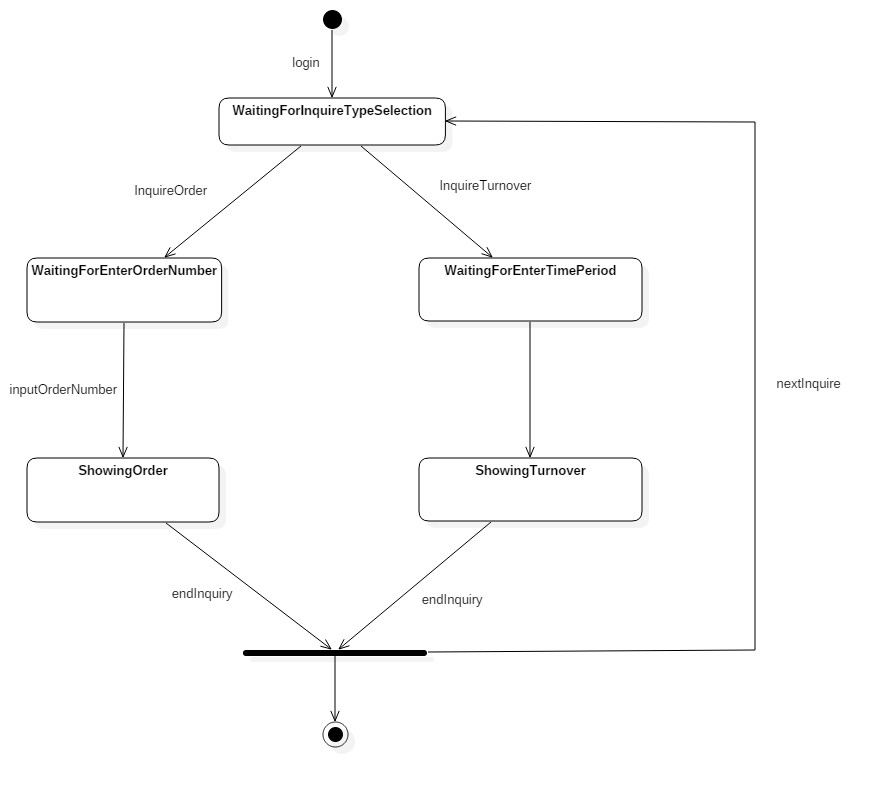
* **业务员\_产品管理**



* **业务员\_货物上下架**

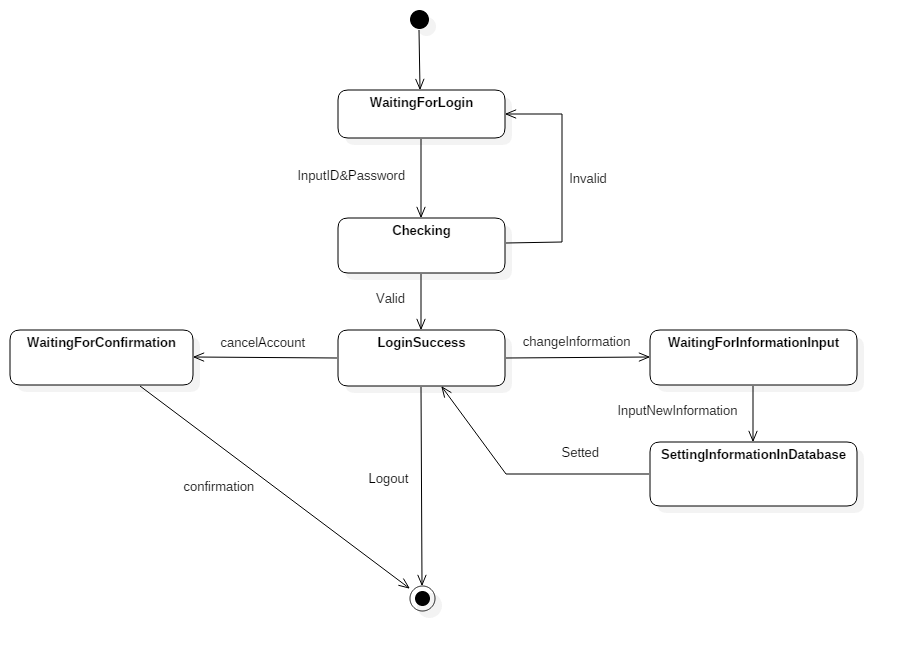


* **业务员\_查询操作**

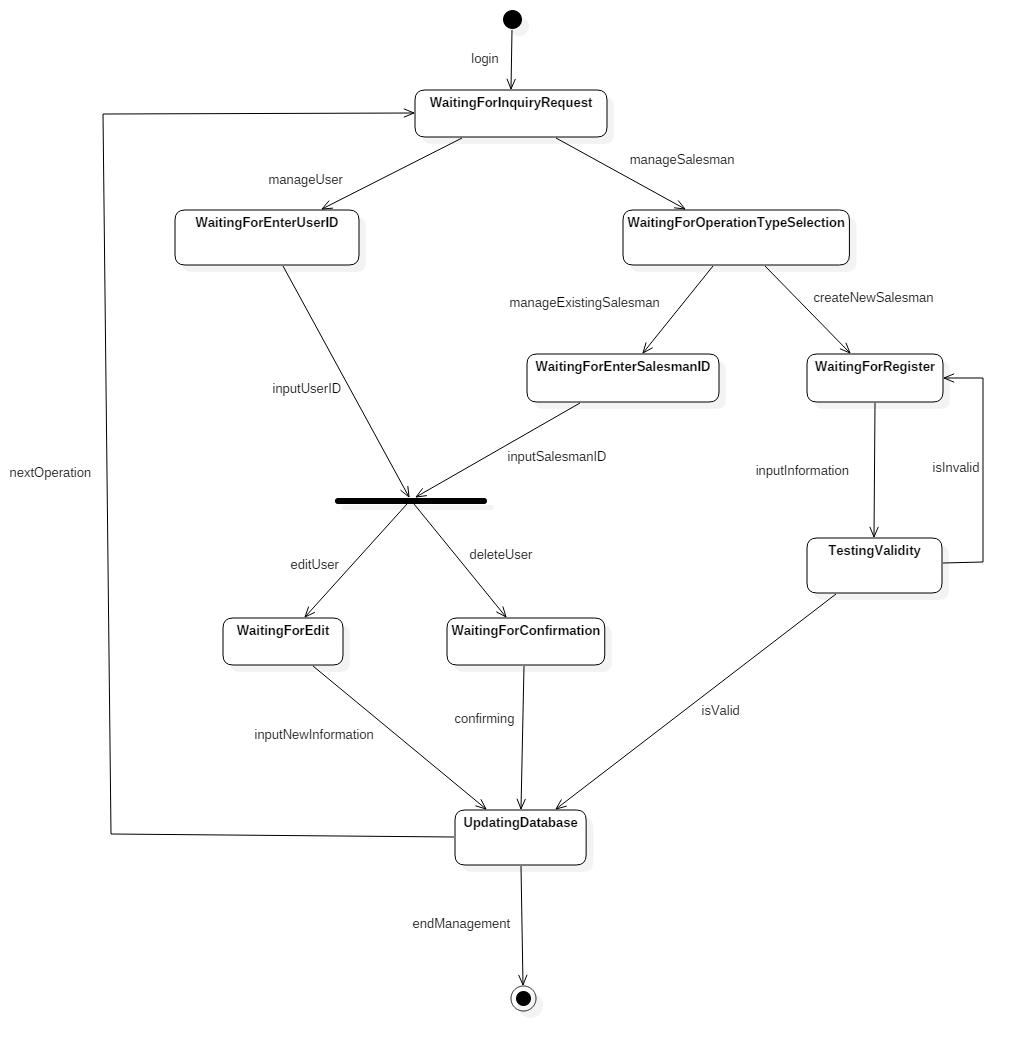


#### 3.4.2.3 系统管理员相关

* **系统管理员\_账户操作**



* **系统管理员\_用户与业务员管理**



## 系统非功能设计

### 3.5.0 约束矩阵

**各个模块进行实现时需要同时考虑到相应非功能性设计的约束，以下为模块与非功能性设计的约束矩阵。**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **模块**  **约束** | **账户模块** | **商品管理模块** | **查询模块** | **购物模块** | **系统管理模块** | **统计模块** |
| **性能设计** | **√** | **√** | **√** | **√** |  |  |
| **维护性设计** | **√** | **√** | **√** | **√** | **√** | **√** |
| **易用性设计** | **√** | **√** | **√** | **√** | **√** |  |
| **兼容性设计** | **√** |  | **√** | **√** |  |  |
| **可拓展性设计** | **√** | **√** | **√** | **√** | **√** | **√** |
| **可靠性设计** | **√** | **√** | **√** | **√** | **√** |  |
| **灾备设计** |  | **√** |  |  | **√** |  |
| **安全性设计** | **√** |  |  | **√** | **√** |  |

### 3.5.1 性能设计

* + - 系统响应速度：用户操作系统平均响应时间小于3秒。
    - 支持最大并发数：100

### 3.5.2 维护性设计

* 整个应用软件系统应能够连续7\*24小时不间断工作，应用软件中的任一模块更新、加载时，在不更新与上下模块的接口的前提下，以不影响业务运转和服务为原则；
* 应用软件具有较高的自动化程度，如：自动异常调度、自动故障告警、自动任务恢复等；
* 应用软件具备相应容错手段，能容许操作人员的某些失误操作。

### 3.5.3 易用性设计

* 该系统提供基于图形化的友好管理界面；
* 符合IE界面风格：

 系统风格协调一致，体现人性化的友好性管理界面；

 符合用户习惯或者容易被用户接受的管理风格；

 具有容错能力，包括错误诊断和提示，并具有快速的系统反应的管理界面。

* 具有详细的、易懂的联机帮助，协助用户使用。

### 3.5.4 兼容性设计

* 对Chrome, Firefox, 360浏览器的进行兼容

### 3.5.5 可拓展性设计

* 对后期用户数量增大时的可拓展性的支持
* 对积分规则、活动结算的业务可拓展性的支持

### 3.5.6 可靠性设计

* 系统满足7\*24稳定运行需要。系统平均无故障时间（MTTF）大于4320小时（180天）；
* 系统容错性强，在外系统故障、用户非法操作、数据内容/格式出错的情况下，仍可正常运行平均失效间隔（MTBF）大于4320小时（180天）；
* 系统故障状态下恢复时间短，平均恢复时间（MTTR）小于10分钟；
* 对人工输入的数据以及来自不同接口的数据进行合法性检查，对错误数据进行自动纠错处理。

### 3.5.7 灾备设计

* 系统做到能支持定期的自动数据备份操作，还提供人工备份的操作。
* 提供多种数据输出格式，该输出格式可以方便快捷的倒回原来系统中。
* 真正做到使系统能在数据损坏，丢失等情况下将备份数据倒回，实现数据恢复。

### 3.5.8 安全性设计

* 用户和权限管理
* 数据范围权限的控制
* 对支付通讯进行加密
* 防止数据篡改、攻击