**黑龙江东方学院**

**大作业**

题目：项目题目描述说明

|  |  |
| --- | --- |
| 组 别 | 第一组 |
| 项目经理 | 许伟杰 |
| 组 员 | 潘帅、曹一凡、王哲源 |
| 专 业 | 2021级1班 |
| 指导教师 |  |
| 学 院 | 信息工程与学院 |
| 日 期 | 2024年5月26日 |

目　　录

[第1章 概述 1](#_Toc169331947)

[1.1 项目背景 1](#_Toc169331948)

[1.3 相关技术 1](#_Toc169331951)

[第2章 系统分析/现状分析 2](#_Toc169331952)

[2.1 业务需求概述 2](#_Toc169331953)

[2.2 未来发展趋势 2](#_Toc169331954)

[第3章 总体设计思路 4](#_Toc169331955)

[3.1 总体思路 4](#_Toc169331956)

[3.2 总体架构 4](#_Toc169331957)

[第4章 设计与实现 5](#_Toc169331958)

[4.1 前端界面设计 5](#_Toc169331959)

[4.2 后端接口设计 14](#_Toc169331977)

[4.3 业务功能设计 23](#_Toc169332065)

[4.4 数据库设计 24](#_Toc169332077)

[4.5 创新点设计 30](#_Toc169332091)

[第5章 经验总结 31](#_Toc169332094)

[5.1 项目经验总结 31](#_Toc169332095)

题目（标题已经定义好）

# 概述

## 项目背景

随着人们生活水平的提高，对于高品质珠宝的需求也在不断增加。越来越多的人愿意在珠宝上投资，不仅仅是作为装饰品，更是一种投资和传承。因此，一个方便的在线平台能够满足消费者日益增长的需求，并为他们提供更多选择，将会受到欢迎。

## 建设意义

## 这个项目旨在创建一个在线平台，专门用于销售各种类型的珠宝产品。这个平台将提供一个方便、安全和可靠的购物体验，让用户能够轻松地浏览、选择和购买他们喜欢的珠宝。也为各种小型珠宝商户提供一个销售平台。

## 相关技术

一、后端（Spring Boot）:

1. Spring Boot：用于构建基于Java的后端服务。
2. JWT鉴权登录
3. MySQL数据库

二、前端（vue.js+微信小程序开发工具）：

1. Vue.js：用于构建用户界面的JavaScript框架。
2. Vue Router：用于在单页应用中实现路由管理。
3. Axios：用于与后端API进行HTTP通信。
4. Element Plus: 基于 Vue 3，面向设计师和开发者的组件库。
5. 微信小程序开发工具：用于开发、调试和发布微信小程序。
6. 微信小程序框架：了解微信小程序的开发规范和API调用方式。
7. TDesign：适配微信端的组件库。

# 系统分析/现状分析

## 业务需求概述

过网页销售珠宝，可以突破地域限制，覆盖全球市场，扩大潜在客户群体；网页售卖珠宝不受时间和地域的限制，可以24小时全天候在线销售，提高销售效率和利润；相对于传统实体店铺，网页售卖珠宝的项目通常需要较低的成本来进行运营和维护，包括租金、人工成本等；过数据分析和用户行为追踪，可以实现精准的营销策略，针对用户的喜好和需求进行个性化推荐，提高销售转化率通过不断优化网站设计、购物流程、客户服务等方面，提升用户体验，增强用户黏性和忠诚度；通过预售、定制等方式，可以有效降低库存风险，减少资金占用和滞销产品的损失；通过网页售卖珠宝，可以加强品牌的线上曝光度，提升品牌知名度和影响力；通过数据分析和报告，可以更好地了解用户行为和市场趋势，为业务决策提供数据支持和指导；通过网站上的评论、评分、社交分享等功能，可以与客户进行互动，获取反馈和建议，不断改进产品和服务。

## 未来发展趋势

增强的数字体验： 随着技术的日益发展，未来网页售卖珠宝的项目将提供更加丰富的数字体验，例如增强现实（AR）和虚拟现实（VR）技术，使顾客可以更生动地体验珠宝产品，提高购买决策的互动性；个性化定制服务： 未来的趋势将更加强调个性化定制的珠宝产品。网页售卖珠宝的项目将会积极推动定制体验，通过在线设计工具和交互式界面，使客户可以根据自己的喜好和需求定制珠宝产品，从而提高用户参与度和忠诚度；可持续发展： 环保和社会责任意识不断增强。网页售卖珠宝的项目将更多地关注可持续发展，并且提供符合环保标准和社会责任的产品。同时，通过教育和透明度，项目可以向消费者传达其产品的可持续性；区块链技术应用： 区块链技术可以被用来跟踪珠宝的来源和品质，从而确保产品的真实性、可追溯性以及打击造假。这将有利于提高消费者对产品的信任度；更强的安全和隐私保护： 随着网络安全和数据隐私日益受到关注，网页售卖珠宝的项目将不断加强网络安全和数据保护措施，以确保顾客的信息安全和隐私；社交化购物体验： 未来的网页售卖珠宝项目可能更加注重社交化购物体验，在网站上集成社交分享和互动功能，通过用户生成的内容和社交推荐提高产品的曝光和销售；跨境电商和本地化服务： 随着全球化趋势的加速，网页售卖珠宝的项目也将更多地关注跨境电商，并提供更符合当地文化和偏好的本地化服务。这些趋势将影响未来网页售卖珠宝项目的发展方向，并将对项目的业务模式、技术应用和客户体验等方面产生深远的影响。

# 总体设计思路

## 总体思路

市场调研和定位： 首先，进行市场调研，了解珠宝行业的发展趋势、竞争对手、目标受众和消费者需求。根据调研结果，明确定位并确定自身项目的特色和优势，为后续的运营和推广奠定基础；建立品牌形象： 开发一个吸引人的品牌形象，其中包括命名、标识、品牌故事和理念。确保品牌与目标受众的需求和品味相符，并在网站上突出展示品牌价值观；供应链与产品采购： 确保与可靠的供应商合作，确保珠宝的品质和真实性。建立良好的供应链，确保产品来源可追溯，可持续性，以及符合标准；产品展示和描述： 开发专业且吸引人的产品页面，包括高质量的珠宝图片、详细的产品描述和规格，以及相关的珠宝知识和价格信息；在线交易平台： 搭建安全可靠的在线支付系统，并提供多种支付方式和货币选择，以适应不同地区客户的需求。同时，确保交易过程的安全性和可信度；个性化定制服务： 提供个性化定制服务，允许客户根据自己的喜好和预算定制珠宝产品，包括在线设计工具和定制咨询；用户体验优化： 重视用户体验，确保网站界面友好、易于导航，提供给客户良好的浏览和购物体验。以提升品牌知名度和吸引潜在客户；客户服务与售后： 提供高品质的客户服务和售后支持，包括在线咨询、退换货政策和售后维护等，以确保客户满意度和忠诚度；数据分析和优化： 运用数据分析工具追踪网站流量、用户行为以及销售数据，通过数据洞察不断优化网站内容、产品和营销策略。这些是建立网页售卖珠宝项目的一般思路，关键在于深入了解市场需求，提供优质产品和服务，以及持续优化运营和用户体验。

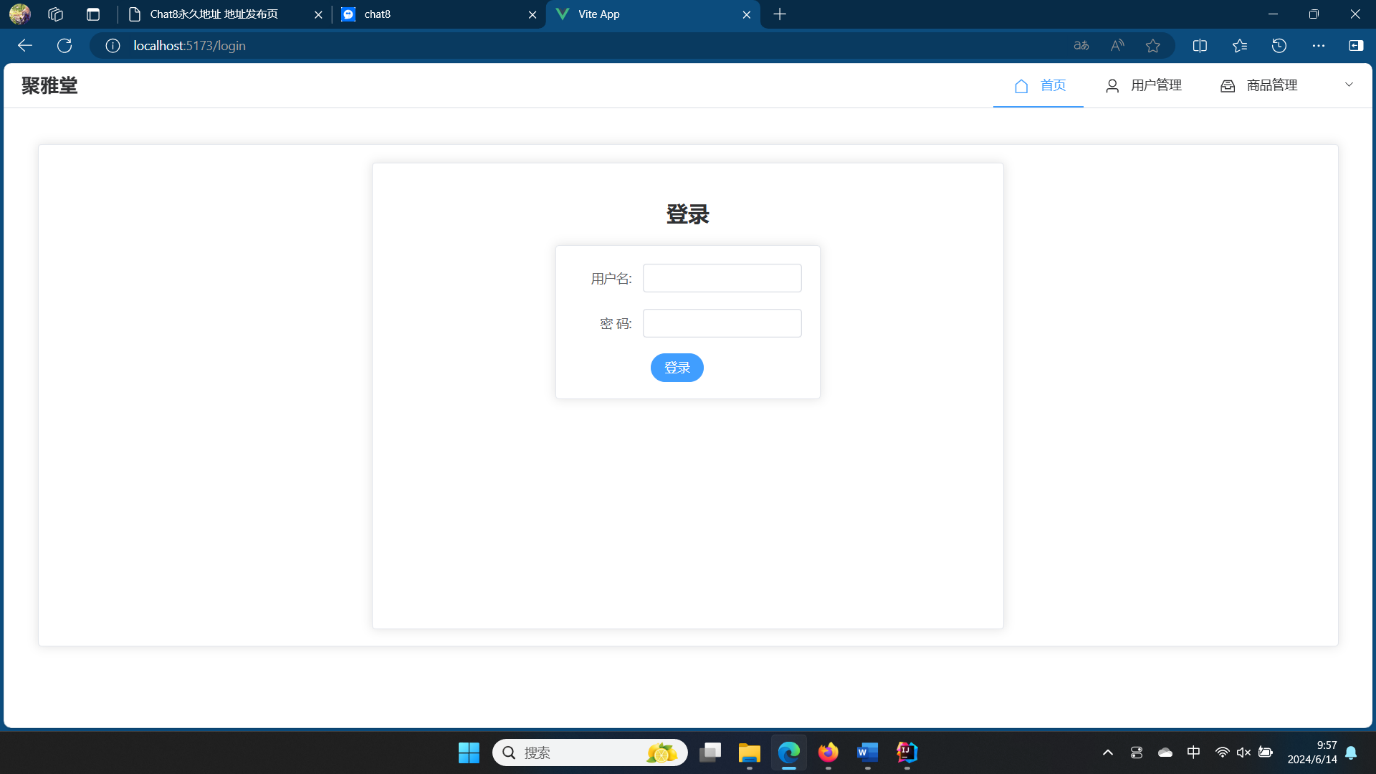
## 总体架构

1. 前端
   1. 管理员前端（Vue.js）
   2. 用户前端（微信小程序开发工具）
2. 后端（Spring Boot）
   1. 控制器 (Controller):负责处理前端发送的HTTP请求，并将请求转发给对应的服务处理，然后返回响应给前端。
   2. 数据处理(Mapper):Mybatis框架中定义SQL语句和映射关系的配置文件。
   3. 数据库(MySQL):本项目使用MySQL数据库存储数据。
   4. 服务 (Service)：包含业务逻辑的服务层，负责处理具体的业务需求，例如处理用户登录、查询数据等。

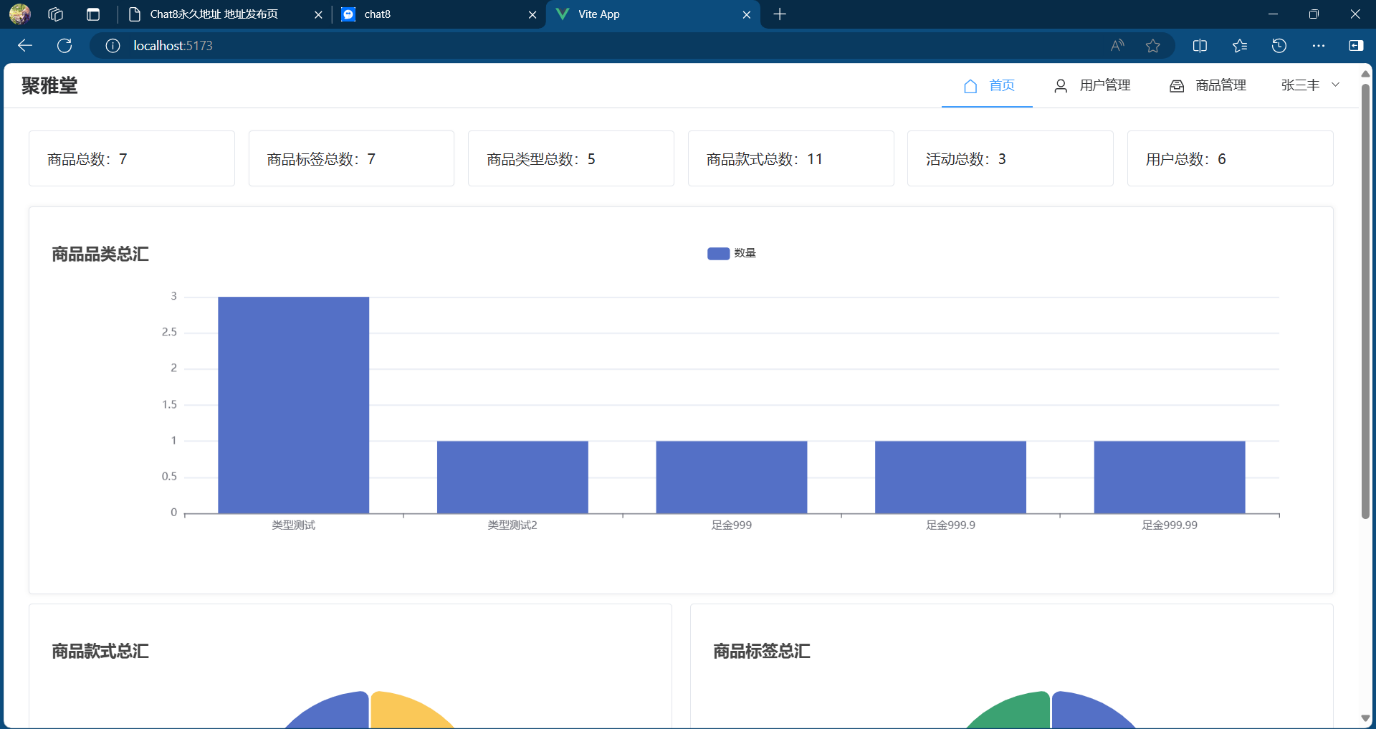
# 设计与实现

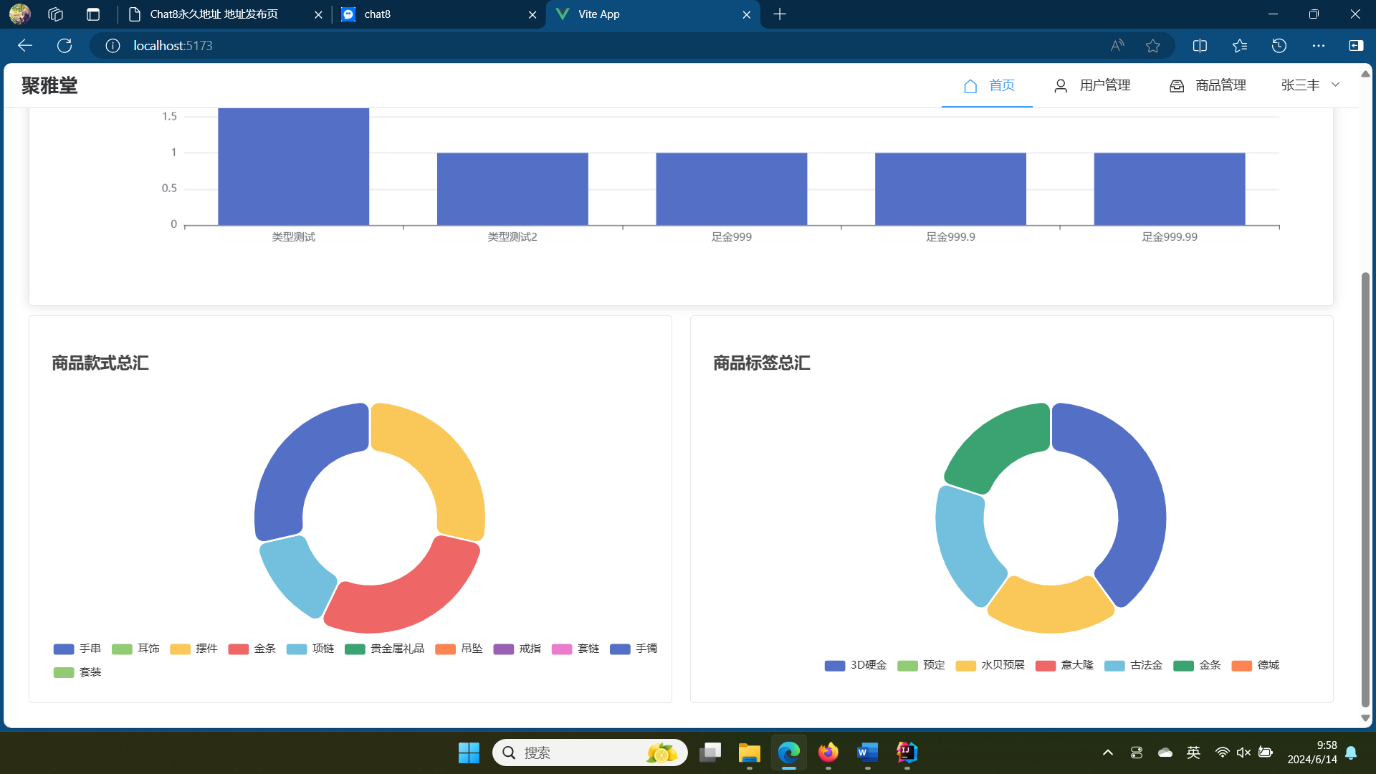
## 前端界面设计

### 管理员登录界面设计

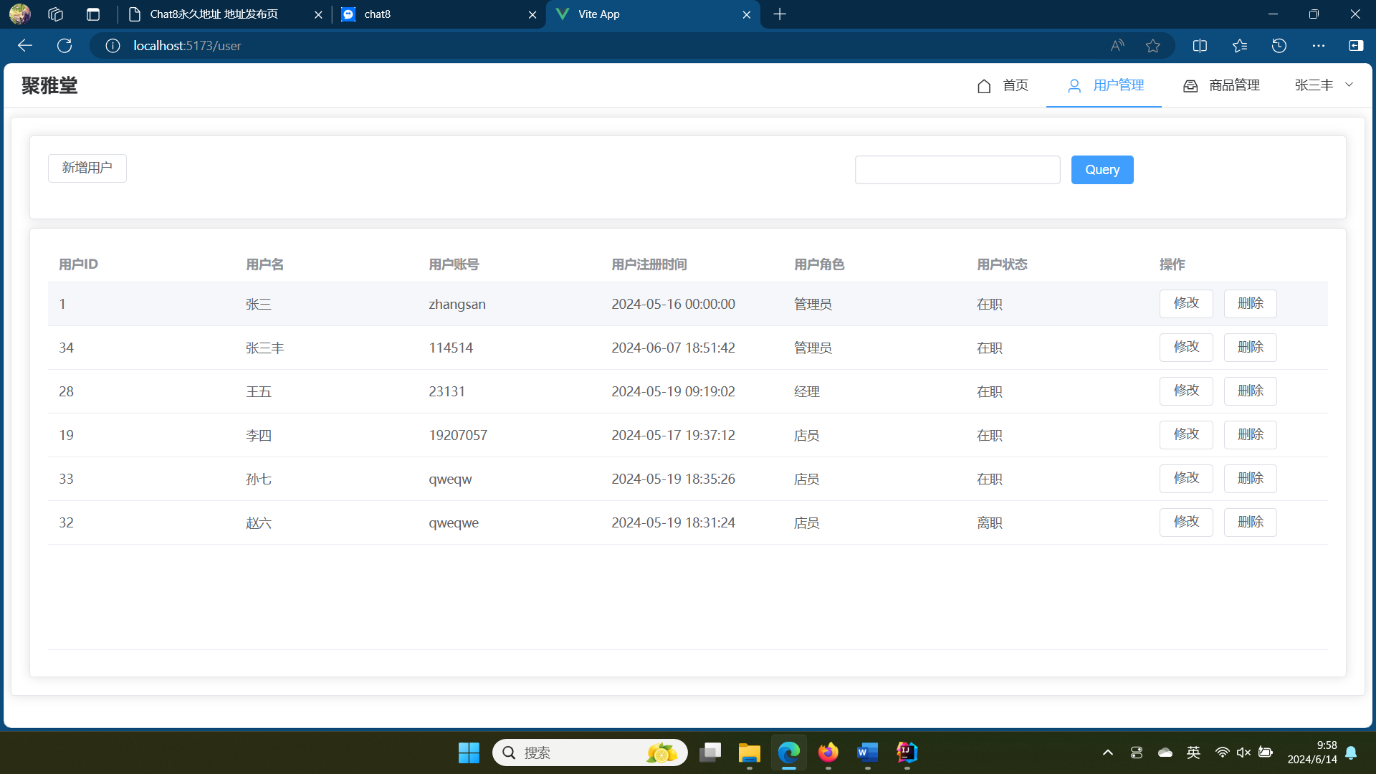


### 管理员首页界面设计

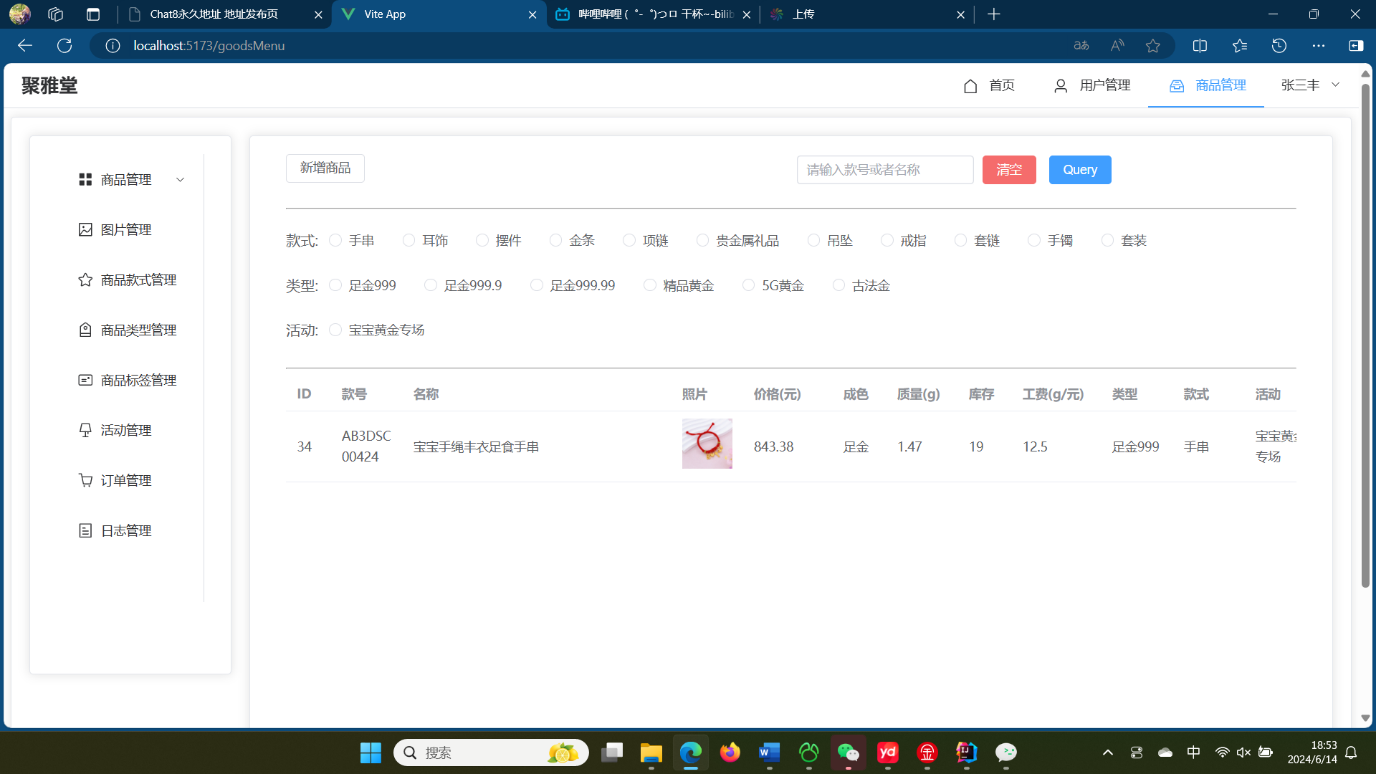




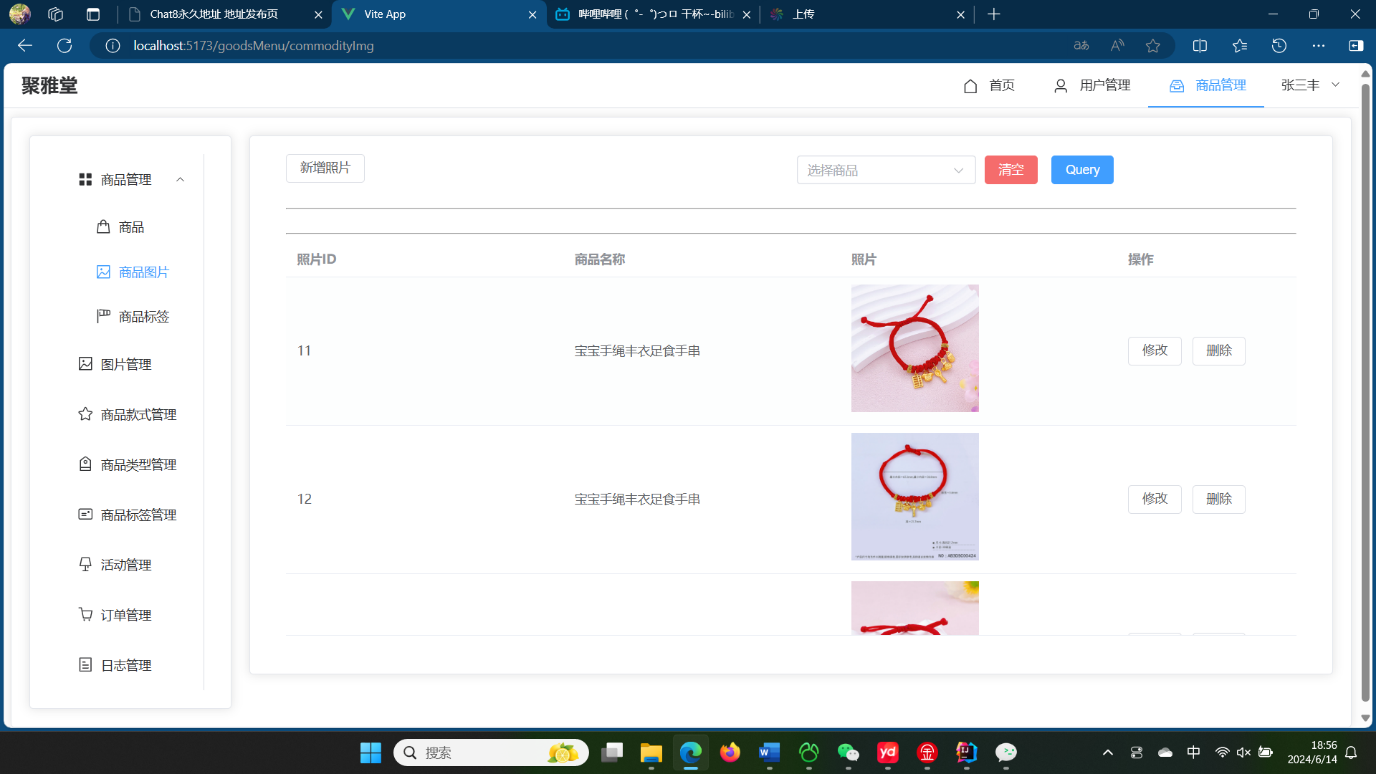
### 管理员用户管理页面设计



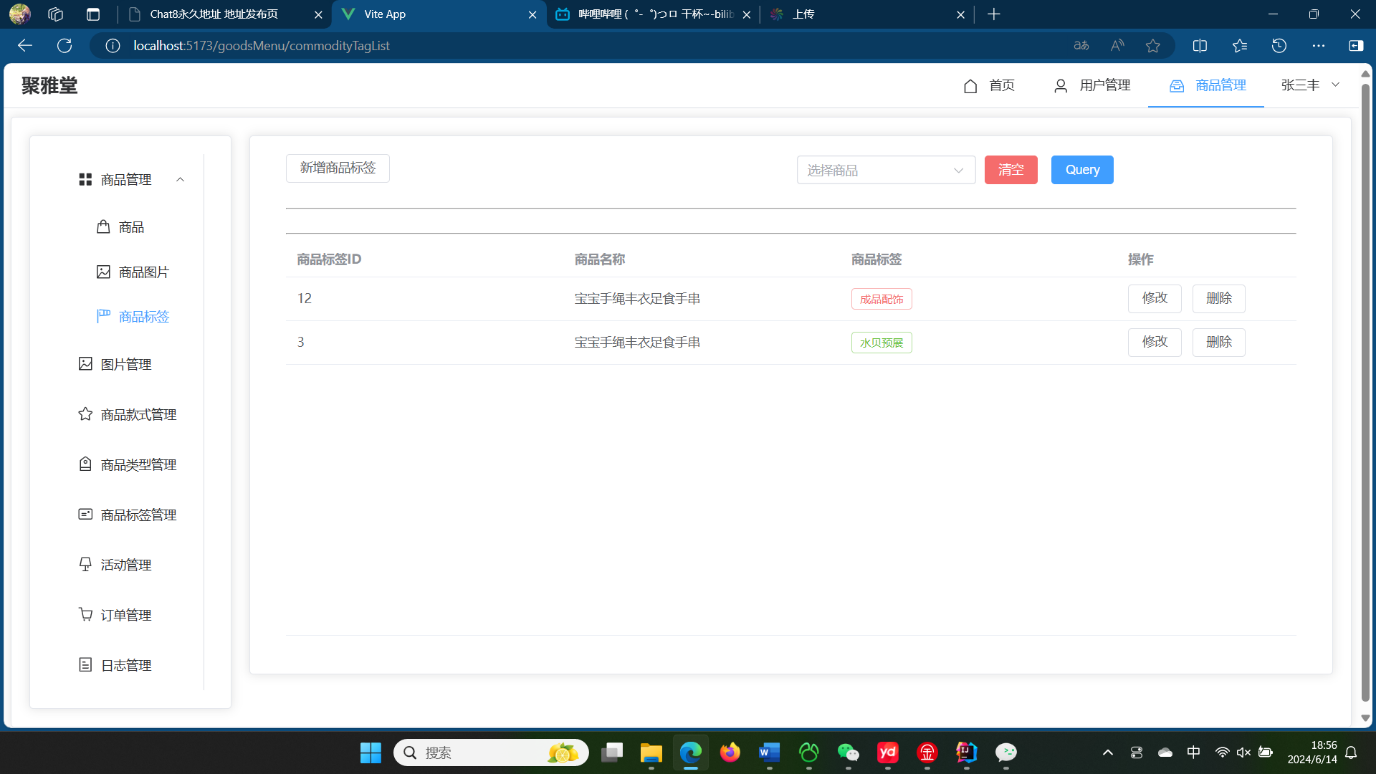
### 管理员商品管理界面设计



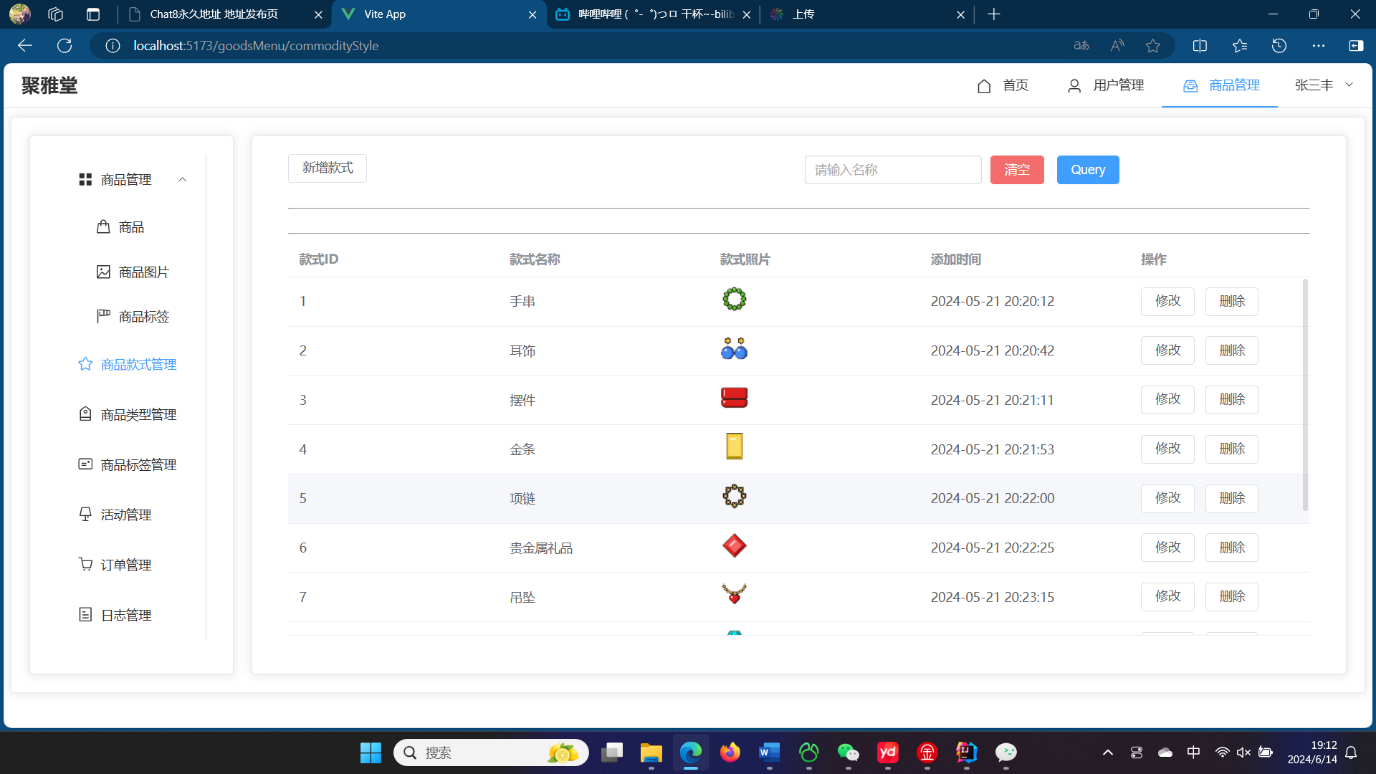
### 管理员商品图片管理界面设计



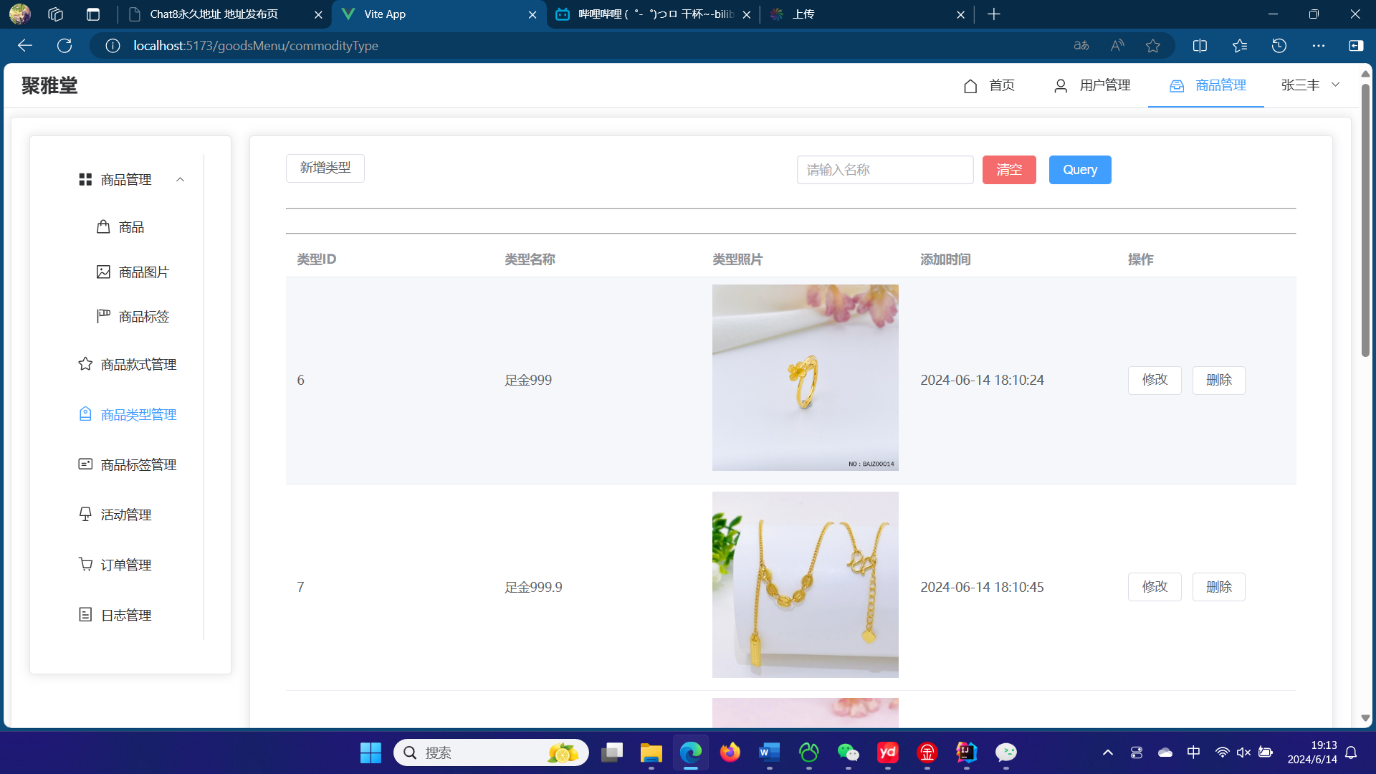
### 管理员商品标签管理界面设计



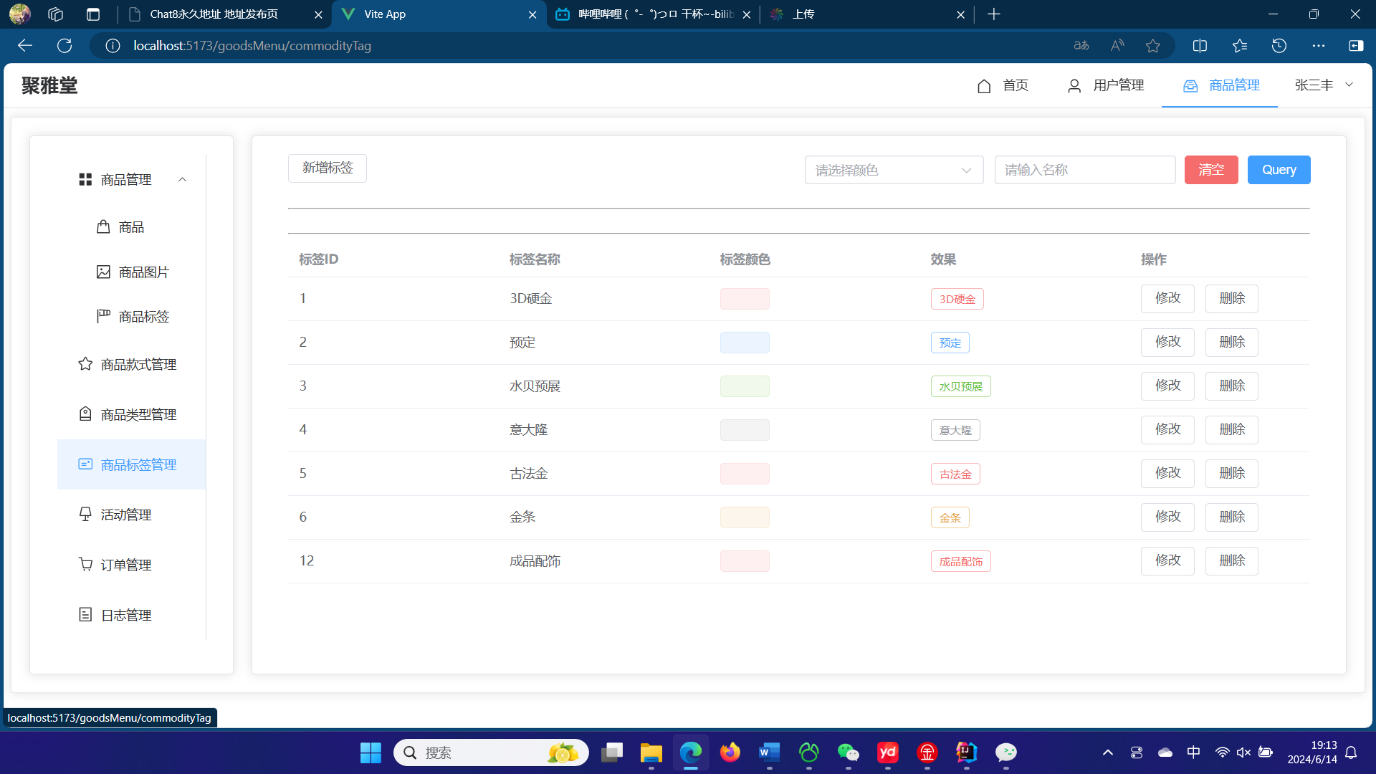
### 管理员商品款式管理界面设计



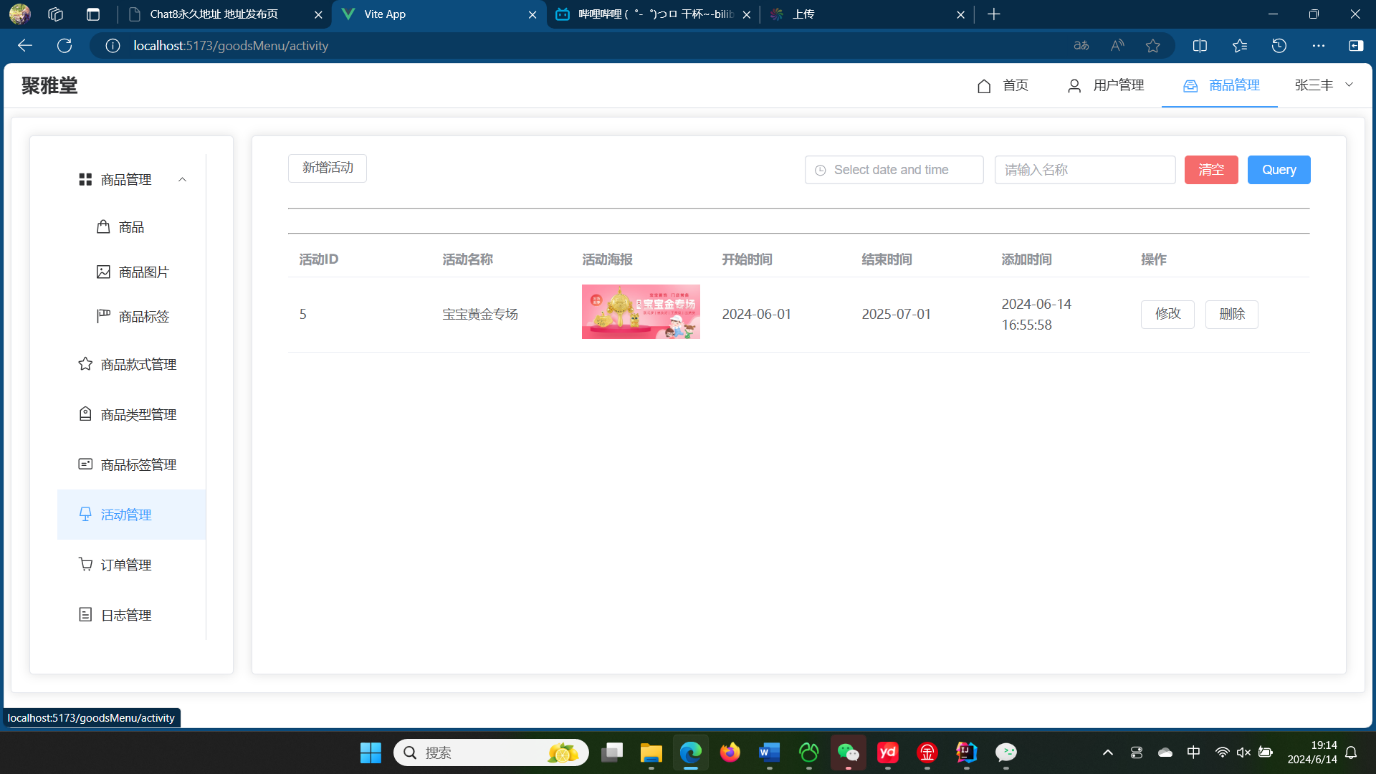
### 管理员类型管理界面设计



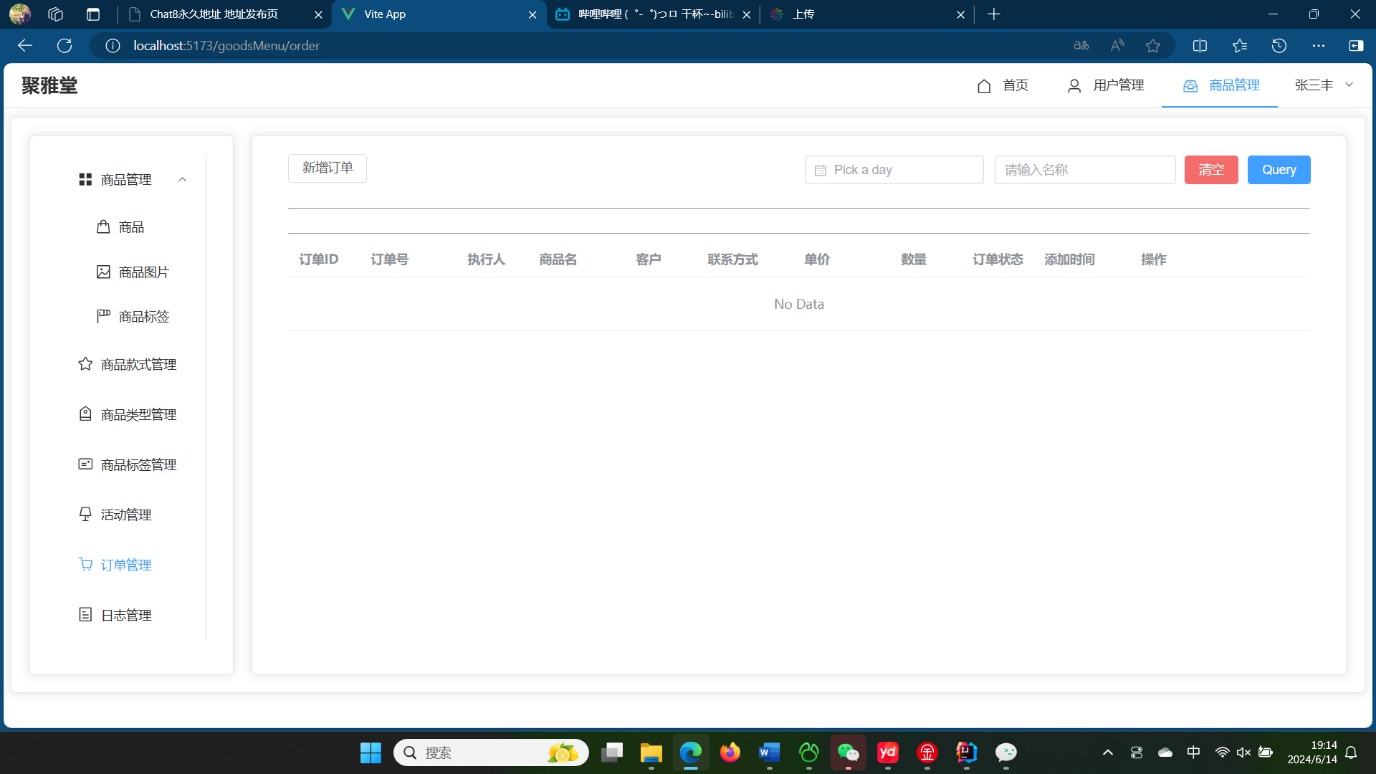
### 管理员标签管理界面设计



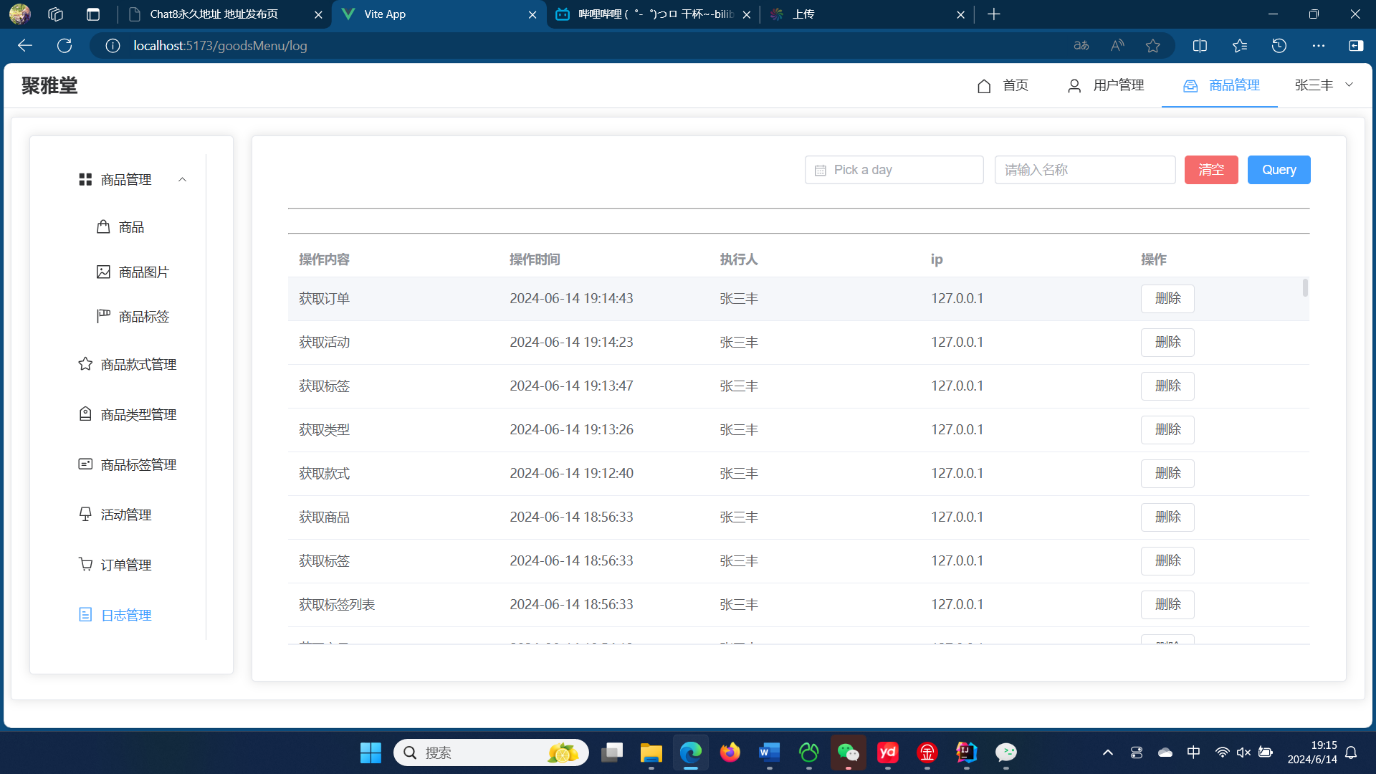
### 管理员活动管理界面设计



### 管理员订单管理界面设计



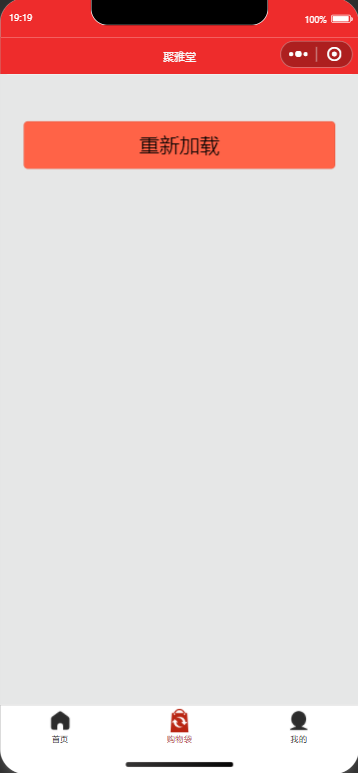
### 管理员日志管理界面设计



### 微信端首页界面设计

### 微信端订单界面设计

### 微信端个人中心界面设计

### 微信端商品界面设计



### 微信端商品详情界面设计



## 后端接口设计

加粗字体为必填项

### 获取活动接口设计

http://127.0.0.1:2001/api/ getActivity

返回活动表中的数据。

### 获取活动数量接口

http://127.0.0.1:2001/api/ getActivityCount

返回活动表中的活动总数。

### 按照名称获取活动接口

http://127.0.0.1:2001/api/ getActivityName/

传入activityId、activityName、activityBeginTime，返回对应活动的数据。

### 更新活动接口设计

http://127.0.0.1:2001/api/ updateActivity

传入activityName、 activityImgUrl、activityBeginTime、activityEndTime、**activityId**，返回更新状态。

### 新增活动接口设计

http://127.0.0.1:2001/api/ addActivity

传入**activityName**、**activityImgUrl**、**activityBeginTime**、**activityEndTime**、**activityAddTime**，返回新增状态。

### 删除活动接口设计

http://127.0.0.1:2001/api/ deleteActivityById/

传入**activityId**，返回删除状态。

### 获取商品接口设计

http://127.0.0.1:2001/api/ getCommodity

返回商品信息。

### 按照名称获取商品接口设计

http://127.0.0.1:2001/api/ getCommodityName/

传入commodityId、commodityNum、commodityTagId、commodityTypeId、commodityStyleId、commodityActivityId，返回对应的数据。

### 修改商品信息接口设计

http://127.0.0.1:2001/api/ updateCommodity

传入commodityNum、commodityName、commodityUrl、commodityPrice、commodityFineness、commodityWeight、commodityInventory、commodityWage、commodityTagId、commodityTypeId、commodityStyleId、commodityActivityId、**commodityId**，返回更新状态。

### 新增商品接口设计

http://127.0.0.1:2001/api/ addCommodity

传入**commodityNum**、**commodityName**、**commodityUrl**、**commodityPrice**、**commodityFineness**、**commodityWeight**、**commodityInventory**、**commodityWage**、**commodityTagId**、**commodityTypeId**、**commodityStyleId**、**commodityActivityId**、**commodityAddTime**，返回新增状态。

### 删除商品接口设计

http://127.0.0.1:2001/api/ deleteCommodityById/

传入**commodityId**，返回删除状态。

### 获取图片接口设计

http://127.0.0.1:2001/api/ getImg

返回图片数据。

### 按照名称获取图片接口设计

http://127.0.0.1:2001/api/ getImgName/

传入commodityName，返回查询到的数据。

### 更新图片接口设计

http://127.0.0.1:2001/api/ updateImg

传入commodityImgUrl、**commodityImgId**，返回更新状态。

### 新增图片接口设计

http://127.0.0.1:2001/api/ addImg

传入**commodityImgUrl**、**commodityImgId**，返回新增状态。

### 删除图片接口设计

http://127.0.0.1:2001/api/ deleteImgById/

传入**commodityImgId**，返回删除状态。

### 获取款式接口设计

http://127.0.0.1:2001/api/ getStyle

返回款式数据。

### 按照名称获取款式接口设计

http://127.0.0.1:2001/api/ getStyleName/

传入commodityStyleName，返回查询接口。

### 更新款式接口设计

http://127.0.0.1:2001/api/ updateStyle

传入commodityStyleName、commodityStyleImgUrl、**commodityStyleId**，返回更新状态。

### 新增款式接口设计

http://127.0.0.1:2001/api/ addStyle

传入**commodityStyleName**、**commodityStyleImgUrl**、**commodityStyleAddTime**，返回新增状态。

### 删除款式接口设计

http://127.0.0.1:2001/api/ deleteStyleById/

传入**commodityStyleId**，返回删除状态

### 获取标签列表接口设计

http://127.0.0.1:2001/api/ getTagList

返回标签列表数据

### 查询标签列表接口设计

http://127.0.0.1:2001/api/ getTagListName/

传入commodityTagName、commodityTagName，返回查询数据

### 更新标签列表接口设计

http://127.0.0.1:2001/api/ updateTagList

传入commodityTagId，**commodityTagListId**，返回更新状态

### 新增标签列表接口设计

http://127.0.0.1:2001/api/ addTagList

传入**commodityId**、**commodityTagId**，返回新增状态

### 删除标签列表接口设计

http://127.0.0.1:2001/api/ deleteTagListById/

传入**commodityTagListId**，返回删除状态

### 获取标签接口设计

http://127.0.0.1:2001/api/ getTag

返回标签数据

### 查询标签接口设计

http://127.0.0.1:2001/api/ getTagName/

传入commodityTagId、commodityTagName、commodityTagColor，返回查询数据

### 更新标签接口设计

http://127.0.0.1:2001/api/ updateTag

传入commodityTagName、commodityTagColor、**commodityTagId**，返回更新状态

### 新增标签接口设计

http://127.0.0.1:2001/api/ addTag

传入**commodityTagName**、**commodityTagColor**，返回新增状态

### 删除标签接口设计

http://127.0.0.1:2001/api/ deleteTagById/

传入**commodityTagId**，返回删除状态

### 获取类型接口设计

http://127.0.0.1:2001/api/ getType

返回类型数据

### 查询类型接口设计

http://127.0.0.1:2001/api/ getTypeName/

传入commodityTypeName，返回查询结果

### 更新类型接口设计

http://127.0.0.1:2001/api/ updateType/

传入commodityTypeName、commodityTypeImgUrl、**commodityTypeId**，返回更新状态

### 新增类型接口设计

http://127.0.0.1:2001/api/ addType

传入**commodityTypeName**、**commodityTypeImgUrl**、**commodityTypeAddTime**，返回新增状态

### 删除类型接口设计

http://127.0.0.1:2001/api/ deleteTypeById/

传入**commodityTypeId**，返回删除状态

### 获取商品数量接口设计

http://127.0.0.1:2001/api/ getcCount

返回商品数量

### 获取类型数量接口设计

http://127.0.0.1:2001/api/ getctyCount

返回类型数量

### 获取款式数量接口设计

http://127.0.0.1:2001/api/ getcsCount

返回款式数量

### 获取活动数量接口设计

http://127.0.0.1:2001/api/ getaCount

返回活动数量

### 获取用户数量接口设计

http://127.0.0.1:2001/api/ getuCount

返回用户数量

### 获取标签数量接口设计

http://127.0.0.1:2001/api/ getctCount

返回标签数量

### 获取全部类型接口设计

http://127.0.0.1:2001/api/ getTypeTotal

返回全部类型数据

### 获取全部款式接口设计

http://127.0.0.1:2001/api/ getStyleTotal

返回全部款式数据

### 获取全部标签接口设计

http://127.0.0.1:2001/api/ getTagTotal

返回全部标签数据

### 获取日志接口设计

http://127.0.0.1:2001/api/ getLog

返回日志数据

### 新增日志接口设计

http://127.0.0.1:2001/api/ addLog

传入**id**、**name**、**time**、**userName**、**ip**，返回新增状态

### 删除日志接口设计

http://127.0.0.1:2001/api/ deleteLogById/

传入**id**，返回删除状态

### 获取订单接口设计

http://127.0.0.1:2001/api/ getOrder

返回订单数据

### 查询订单接口设计

http://127.0.0.1:2001/api/ getOrderName/

传入orderNum、addTime，返回查询结果

### 按照执行人查询订单接口设计

http://127.0.0.1:2001/api/ getOrderUserName/

传入userId，返回查询结果

### 查询已完成订单接口设计

http://127.0.0.1:2001/api/getOrderYwc/

传入userId，返回查询结果

### 查询未完成订单接口设计

http://127.0.0.1:2001/api/getOrderWwc/

传入user Id，返回查询结果

### 更新订单接口设计

http://127.0.0.1:2001/api/updateOrder

传入userId、commodityId、client、tel、num、orderStateId、**orderId**，返回更新状态

### 新增订单接口设计

http://127.0.0.1:2001/api/addOrder

传入**orderNum**、**userId**、**commodityId**、**client**、**tel**、**num**、**addTime**，返回新增状态

### 删除订单接口设计

http://127.0.0.1:2001/api/deleteOrderById/

传入**orderId**，返回删除状态

### 后台登录接口设计

http://127.0.0.1:2001/api/login

传入**userAccount**，**userPassword**，返回token

### 微信登录接口设计

http://127.0.0.1:2001/api/userPassword

传入userAccount，userPassword，返回token

### 获取用户接口设计

http://127.0.0.1:2001/api/ getUser

返回用户数据

### 查询用户接口设计

http://127.0.0.1:2001/api/ getUserName/

传入userName，返回查询结果

### 更新用户接口设计

http://127.0.0.1:2001/api/ updateUser

传入userPassword、userRoleId、userStateId、**userId**，返回更新状态

### 新增用户接口设计

http://127.0.0.1:2001/api/ addUser

传入**userName**、**userAccount**、**userPassword**、**userRoleId**、**userRegister**，返回新增状态

### 删除用户接口设计

http://127.0.0.1:2001/api/ deleteUserById/

传入**userId**，返回删除状态

### 微信端获取标签接口设计

http://127.0.0.1:2001/api/ getWxTag/

返回标签数据

### 微信端获取商品接口设计

http://127.0.0.1:2001/api/ getWxCommodity/

返回商品数据

### 微信端获取类型接口设计

http://127.0.0.1:2001/api/ getWxType/

返回类型数据

### 微信端获取商品图片接口设计

http://127.0.0.1:2001/api/ getWxCommodityImg/

返回商品图片

### 微信端获取类型价格升序接口设计

http://127.0.0.1:2001/api/ getWxTypePriceUp

返回类型数据，按照价格升序

### 微信端获取类型价格降序接口设计

http://127.0.0.1:2001/api/ getWxTypePriceDown

返回类型数据，按照价格降序

### 微信端获取类型重量升序接口设计

http://127.0.0.1:2001/api/ getWxTypeWageUp/

返回类型数据，按照重量升序

### 微信端获取类型重量降序接口设计

http://127.0.0.1:2001/api/ getWxTypeWageDown/

返回类型数据，按照重量降序

### 微信端获取类型时间升序接口设计

http://127.0.0.1:2001/api/getWxTypeTimeUp/

返回类型数据，按照时间升序

### 微信端获取类型时间降序接口设计

http://127.0.0.1:2001/api/getWxTypeTimeDown/

返回类型数据，按照时间降序

### 微信端获取款式接口

http://127.0.0.1:2001/api/getWxStyle

返回款式数据

### 微信端获取款式价格升序接口设计

http://127.0.0.1:2001/api/getWxStylePriceUp/

返回款式数据，按照价格升序

### 微信端获取款式价格降序接口设计

http://127.0.0.1:2001/api/getWxStylePriceDown/

返回款式数据，按照价格降序

### 微信端获取款式重量升序接口设计

http://127.0.0.1:2001/api/getWxStyleWageUp/

返回款式数据，按照重量升序

### 微信端获取款式重量降序接口设计

http://127.0.0.1:2001/api/getWxStyleWageDown

返回款式数据，按照重量降序

### 微信端获取款式时间升序接口设计

http://127.0.0.1:2001/api/getWxStyleTimeUp

返回款式数据，按照时间升序

### 微信端获取款式时间降序接口设计

http://127.0.0.1:2001/api/getWxStyleTimeDown

返回款式数据，按照时间降序

### 微信端获取活动接口

http://127.0.0.1:2001/api/getWxActivity

返回活动数据

### 微信端获取活动价格升序接口设计

http://127.0.0.1:2001/api/getWxActivityPriceUp

返回活动数据，按照价格升序

### 微信端获取活动价格降序接口设计

http://127.0.0.1:2001/api/getWxActivityPriceDown

返回活动数据，按照价格降序

### 微信端获取活动重量升序接口设计

http://127.0.0.1:2001/api/getWxActivityWageUp

返回活动数据，按照重量升序

### 微信端获取活动重量降序接口设计

http://127.0.0.1:2001/api/getWxActivityWageDown

返回活动数据，按照重量降序

### 微信端获取活动时间升序接口设计

http://127.0.0.1:2001/api/getWxActivityTimeUp

返回活动数据，按照时间升序

### 微信端获取活动时间降序接口设计

http://127.0.0.1:2001/api/getWxActivityTimeDown

返回活动数据，按照时间降序

## 业务功能设计

## 管理员登录功能

用于管理员登录，验证是否为管理员身份，生成密钥，用于验证是否有登录。

## 用户管理功能

用于管理用户，可以查看用户的数据，新增、修改、删除用户数据，控制用户权限。

## 管理员首页展示功能

用于统计各种数据并展示。

## 商品管理功能

用于管理商品数据。

## 商品款式管理功能

用于管理商品款式。

## 商品标签管理功能

用于管理商品标签。

## 商品类型管理功能

用于商品类型管理

## 活动管理功能

用于管理活动

## 订单管理功能

用于管理订单

## 日志管功能

用于记录和管理日志

## 订单功能

用于用户订购商品，可以记录下顾客的姓名和联系方式。

## 数据库设计

* + 1. Activity表

CREATE TABLE `activity` (

`activity\_id` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '活动ID',

`activity\_name` varchar(255) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci NOT NULL COMMENT '活动名',

`activity\_img\_url` varchar(255) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci NOT NULL COMMENT '活动图片路径',

`activity\_begintime` date NOT NULL COMMENT '活动开始时间',

`activity\_endtime` date NOT NULL COMMENT '活动结束时间',

`activity\_addtime` datetime NOT NULL COMMENT '活动添加时间',

PRIMARY KEY (`activity\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci COMMENT='活动';

## Commodity表

CREATE TABLE `commodity` (

`commodity\_id` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '商品ID',

`commodity\_num` varchar(500) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '商品款号',

`commodity\_name` varchar(500) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci NOT NULL COMMENT '商品名称',

`commodity\_url` varchar(255) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '商品封面地址',

`commodity\_price` decimal(10,2) NOT NULL COMMENT '商品价格',

`commodity\_fineness` varchar(255) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci NOT NULL COMMENT '商品成色',

`commodity\_weight` decimal(10,2) NOT NULL COMMENT '商品重量',

`commodity\_inventory` varchar(255) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci NOT NULL COMMENT '商品库存',

`commodity\_tag\_id` bigint(20) DEFAULT NULL COMMENT '商品标签ID',

`commodity\_wage` decimal(10,2) NOT NULL COMMENT '商品工费',

`commodity\_addtime` datetime NOT NULL COMMENT '商品添加时间',

`commodity\_type\_id` int(11) DEFAULT NULL COMMENT '商品类Id',

`commodity\_style\_id` int(11) DEFAULT NULL COMMENT '商品款式Id',

`commodity\_activity\_id` int(11) DEFAULT NULL COMMENT '商品活动Id',

PRIMARY KEY (`commodity\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=35 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci COMMENT='商品';

## commodity\_img表

CREATE TABLE `commodity\_img` (

`commodity\_img\_id` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '商品图片ID',

`commodity\_img\_url` varchar(500) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '商品图片地址',

`commodity\_id` bigint(20) DEFAULT NULL COMMENT '商品Id',

PRIMARY KEY (`commodity\_img\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=16 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

## commodity\_img\_list表

CREATE TABLE `commodity\_img\_list` (

`commodity\_img\_id` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '商品图片ID',

`commodity\_id` bigint(20) DEFAULT NULL COMMENT '商品ID',

`commodity\_img\_url\_id` bigint(20) DEFAULT NULL COMMENT '商品图片ID',

PRIMARY KEY (`commodity\_img\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

## commodity\_style表

CREATE TABLE `commodity\_style` (

`commodity\_style\_id` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '商品款式ID',

`commodity\_style\_name` varchar(255) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci NOT NULL COMMENT '商品款式名称',

`commodity\_style\_img\_url` varchar(255) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci NOT NULL COMMENT '商品款式照片路径',

`commodity\_style\_addtime` datetime NOT NULL COMMENT '商品款式添加时间',

PRIMARY KEY (`commodity\_style\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=22 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci COMMENT='商品款式';

## commodity\_tag表

CREATE TABLE `commodity\_tag` (

`commodity\_tag\_id` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '商品标签ID',

`commodity\_tag\_name` varchar(255) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '商品标签名称',

`commodity\_tag\_color` varchar(255) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '商品标签颜色',

PRIMARY KEY (`commodity\_tag\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=13 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

## commodity\_tag\_list表

CREATE TABLE `commodity\_tag\_list` (

`commodity\_tag\_list\_id` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '商品标签列表ID',

`commodity\_tag\_id` bigint(20) DEFAULT NULL COMMENT '商品标签ID',

`commodity\_id` bigint(20) DEFAULT NULL COMMENT '商品ID',

PRIMARY KEY (`commodity\_tag\_list\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=10 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

## commodity\_type表

CREATE TABLE `commodity\_type` (

`commodity\_type\_id` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '商品类ID',

`commodity\_type\_name` varchar(255) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci NOT NULL COMMENT '商品类名',

`commodity\_type\_img\_url` varchar(255) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci NOT NULL COMMENT '商品类图片路径',

`commodity\_type\_addtime` datetime NOT NULL COMMENT '商品类添加时间',

PRIMARY KEY (`commodity\_type\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=12 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci COMMENT='商品类';

## log表

CREATE TABLE `log` (

`id` int(11) NOT NULL AUTO\_INCREMENT COMMENT '主键ID',

`name` varchar(255) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '操作内容',

`time` datetime DEFAULT NULL COMMENT '操作时间',

`username` varchar(255) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT '微信端执行' COMMENT '操作人',

`ip` varchar(255) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '操作人IP',

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=707 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

## order表

CREATE TABLE `order` (

`order\_id` int(11) NOT NULL AUTO\_INCREMENT COMMENT '订单ID',

`order\_num` varchar(255) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '订单号',

`user\_id` int(11) DEFAULT NULL COMMENT '执行人',

`commodity\_id` int(255) NOT NULL COMMENT '商品ID',

`client` varchar(255) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '客户名',

`tel` varchar(255) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '联系方式',

`num` int(11) DEFAULT '1' COMMENT '数量',

`addTime` datetime DEFAULT NULL COMMENT '添加时间',

`order\_state\_id` int(11) NOT NULL DEFAULT '2' COMMENT '订单状态',

PRIMARY KEY (`order\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=9 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

## order\_state表

CREATE TABLE `order\_state` (

`order\_state\_id` int(11) NOT NULL AUTO\_INCREMENT COMMENT '订单状态ID',

`state\_name` varchar(255) COLLATE utf8\_unicode\_ci NOT NULL COMMENT '订单状态',

PRIMARY KEY (`order\_state\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=3 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

## role表

CREATE TABLE `role` (

`role\_id` int(11) NOT NULL COMMENT '角色ID',

`role` varchar(255) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '角色',

PRIMARY KEY (`role\_id`) USING BTREE

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

## state表

CREATE TABLE `state` (

`state\_id` int(11) NOT NULL COMMENT '状态ID',

`state` varchar(255) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '状态',

PRIMARY KEY (`state\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

## user表

CREATE TABLE `user` (

`user\_id` bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '用户ID',

`user\_name` varchar(255) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '用户名',

`user\_account` varchar(255) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '用户账号',

`user\_password` varchar(255) COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '用户密码',

`user\_register` datetime DEFAULT NULL COMMENT '用户注册时间',

`user\_role\_id` int(255) DEFAULT NULL COMMENT '用户角色ID',

`user\_state\_id` int(255) DEFAULT '1' COMMENT '用户状态ID',

PRIMARY KEY (`user\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=35 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci COMMENT='用户';

## 创新点设计

## 使用了JWT权鉴功能，使用户登录更加安全。

## 使用AOP切面编程，对每一个操作进行环绕通知，可以清晰的记录用户的操作记录，以及快速的定位程序的问题。

# 经验总结

## 项目经验总结