**学位论文独创性声明**

本人郑重声明：

1.坚持以“求实、创新”的科学精神从事研究工作。

2.本论文是我本人在导师指导下进行的研究工作和取得的成果。

3.本论文中除引文外，所有实验、数据和有关材料均是真实的。

4.本论文中除引文和致谢的内容外，不含其他人或者其它机构经发 表或撰写过的研究成果。

5.其他同志对本研究所做的贡献均已在论文中作了声明并表达了谢意。

作者签名：

日  期:2022年4月20日

**摘 要**

随着互联网的快速发展，人们生活发生了巨大的转变，非常多的业务转移到了网络上进行办理，其中最受大家欢迎的就是进行网络购物。网络购物给人们生活带来了极大的便利，人们可以足不出户就能完成商品购买，在线购物受到了大家的一致追捧。

本文设计并开发了一款基于SpringBoot框架的在线购物平台，前端采用Vue框架实现，前后端分离通过HTTP的方式进行调用，数据存放在MySQL数据库中，系统实现前后台两个界面，前台功能包括商品查询、商品购买、购物车、订单管理、收藏夹、收货地址管理等功能，后台由管理员使用对系统信息进行管理和维护，系统集成Interceptor拦截器来进行Token验证，集成E-Charts框架来实现数据统计功能。

【关键词】：在线购物；SpringBoot框架；MySQL数据库；Interceptor拦截器；

Abstract

With the rapid development of the Internet, great changes have taken place in people's lives. A lot of business has been transferred to the Internet for processing, among which the most popular is online shopping. Online shopping has brought great convenience to people's life. People can buy goods without leaving home. Online shopping has been unanimously sought after by everyone.

This paper designs and develops an online shopping platform based on springboot framework. The front end is realized by Vue framework. The front and back ends are separated and called by HTTP. The data is stored in MySQL database. The system realizes the front and back interfaces. The front functions include commodity query, commodity purchase, shopping cart, order management, favorites, receiving address management and so on, In the background, the administrator manages and maintains the system information, integrates the interceptor interceptor to verify the token, and integrates the e-charts framework to realize the data statistics function.

【Keywords】:Online shopping; Springboot framework; Mysql database; Interceptor interceptor;

**目 录**

[1 绪论 1](#_Toc1516)

[1.1 研究背景 1](#_Toc994)

[1.2 研究意义 1](#_Toc21806)

[1.3 研究现状 1](#_Toc24304)

[1.3.1 国内研究现状 1](#_Toc27433)

[1.3.2 国外研究现状 2](#_Toc8881)

[1.4 论文结构 2](#_Toc7532)

[1.5 本章小结 2](#_Toc26486)

[2 技术路线分析 3](#_Toc16407)

[2.1 开发工具 3](#_Toc8612)

[2.1.1 Navicat 3](#_Toc19358)

[2.1.2 IntelliJ IDEA 3](#_Toc28412)

[2.1.3 Tomcat 3](#_Toc14704)

[2.2 开发技术 3](#_Toc13083)

[2.2.1 Interceptor拦截器 3](#_Toc5879)

[2.2.1 Spring Boot框架 4](#_Toc9467)

[2.2.2 Mybatis框架 4](#_Toc2454)

[2.2.3 Vue框架 4](#_Toc15743)

[2.3 本章小结 5](#_Toc5002)

[3 系统分析 6](#_Toc14154)

[3.1 可行性分析 6](#_Toc10455)

[3.1.1 技术可行性 6](#_Toc4727)

[3.1.2 经济可行性 6](#_Toc23727)

[3.1.3 法律可行性 6](#_Toc25326)

[3.1.4 操作可行性 6](#_Toc29423)

[3.2 需求分析 7](#_Toc9823)

[3.2.1 功能性需求分析 7](#_Toc16040)

[3.2.2 非功能性需求分析 8](#_Toc1843)

[3.3 本章小结 9](#_Toc15822)

[4 系统设计 10](#_Toc23542)

[4.1 系统架构设计 10](#_Toc9092)

[4.2 功能模块设计 11](#_Toc19868)

[4.2.1 前台功能模块 11](#_Toc14422)

[4.2.2 后台功能模块 12](#_Toc18244)

[4.3 业务流程设计 13](#_Toc17495)

[4.3.1 用户登录注册 13](#_Toc18939)

[4.3.2 购买商品 14](#_Toc4906)

[4.4.3 修改密码 14](#_Toc30525)

[4.4.4 商品收藏 15](#_Toc25172)

[4.4系统数据库设计 16](#_Toc6284)

[4.3.1 数据库概念设计 16](#_Toc5915)

[4.3.2 数据库表设计 18](#_Toc24765)

[4.4 本章小结 20](#_Toc31000)

[5 系统功能实现 21](#_Toc16564)

[5.1 前台功能实现 21](#_Toc9867)

[5.1.1 注册功能实现 21](#_Toc13041)

[5.1.2 登录功能实现 22](#_Toc14381)

[5.1.3 收藏功能实现 24](#_Toc15520)

[5.1.4 商品下单功能 26](#_Toc30854)

[5.2后台功能模块 29](#_Toc6150)

[5.2.1 分类商品销量统计功能 29](#_Toc29468)

[5.2.2 商品管理模块 31](#_Toc23430)

[5.2.3 用户管理功能 33](#_Toc501)

[5.2.4 权限管理功能 35](#_Toc18831)

[5.3 本章小结 37](#_Toc29758)

[6运行系统和测试 38](#_Toc6446)

[6.1测试规划 38](#_Toc32533)

[6.2测试用例 38](#_Toc17447)

[6.3测试结果 40](#_Toc29135)

[6.4 本章小结 40](#_Toc3762)

[总 结 41](#_Toc12584)

[参考文献 42](#_Toc24151)

[致 谢 43](#_Toc22852)

**1 绪论**

## 1.1 研究背景

随着互联网技术的不断发展，信息产业的不断扩张，网络普及到了各行各业中去。在信息化如此发达的今天，人们方方面面都需要网络，人们可以足不出户进行点餐，外送派送到家，在网上进行购物，挑选心仪的衣服，或者预约网约车来进行出行。网络已经深入人们生活的衣食住行中去。同时，国家鼓励大力发展科技产业，出台多项政策扶持互联网新型产业，传统行业结合互联网进行变更已经成为时代发展的潮流。

截至2021年12月，中国网民规模达10.32亿，较2020年12月增长4296万，互联网普及率达73.0%，互联网在线办理业务的方式逐渐替代了传统行业，数字化、信息化和网络化是21世纪最重要的特征，利用网络技术结合传统行业完成转变已经成为当下市场发展的主旋律。现如今，网络已经覆盖到人们生活的方方面面，电子商务得到了巨大的发展，“互联网+”思维结合电子商务有了非常巨大的空间。网络购物就是在这样的大环境下快速兴起的，并且取得了巨大的成就，网络购物拥有巨大的前景。

## 1.2 研究意义

开发一款线上购物商品平台可以解决人们出现购物问题，省去线下购物的时间，给人们带来极大的便利。同时对于管理人员来说，完善的后台管理系统可以最大程度上省去人力物力的投入，传统需要三个人进行统计记录的工作现在一个人就能轻松完成，计算机管理系统的介入让管理更为高效快捷。

从技术层面来说，本文设计并开发出一款基于Spring Boot框架的前后端分离的在线购物平台，弥补当下SpringBoot框架同商城产业结合较少的空缺[1]，增强对该技术框架的文献研究，采用SpringBoot框架的前后端分离系统目的在于提高商城系统的开发效率、让代码逻辑更加清晰，更加容易应对变化不断的需求，更容易维护，综合减少开发成本[2]。

## 1.3 研究现状

### 1.3.1 国内研究现状

对于国内来说，国内互联网技术的发展时间也相对较短，管理系统出现也晚于世界领先水平，同时对于网上商城系统来说，更是在05年才开始得到发展，开发本系统的Spring框架技术在07年才诞生出来，真正在2010年以后才被国内大量的使用起来，并且得到了迅速的发展。截至目前市面上运用Spring框架开发的系统已经非常多了，但是对于SpringBoot框架开发的网上购物商城则并不常见，已知的著名的网上购物商城Web端淘宝是采用SSH框架进行开发，已经较为落后，但是淘宝的功能实现上面还是非常值得本系统进行参考。

### 1.3.2 国外研究现状

对于国外来说，电商系统发展相对较早，欧美国家最早出现，比较著名的就是亚马逊购物网站，后来陆续出现了不少购物平台，这些系统页面较为整洁优美，功能也十分完善，对于本系统的开发可以有非常不错的参考价值。但是跟本系统采用一样技术的电商系统在国外同样很难查询到，由于搜索条件有限，获取到相关领域的外文文献也较为稀少，已知的系统获取源码的难度也较大，相关开发文档也没有开放出来，只能通过UI来进行设计，从而进行需求分析。所以，目前电商系统研究现状已经有较长的发展历史，但是新型框架和技术采用的较少，需要进行研究和拓展。

## 1.4 论文结构

第一章绪论，从背景到意义介绍了开发线上购物平台的必要性，然后从研究现状进行分析，发现国内外同类型系统的优点和不足，从而来进行借鉴和避免。

第二章技术路线分析，对开发线上购物系统的工具和技术进行介绍，并且介绍选取这些框架和技术的原因。

第三章系统分析，系统分析模块主要是对系统的需求以及可行性进行分析和介绍，介绍线上购物系统的需求转换的过程，并且分析该系统的开发可行性。

第四章系统设计，系统设计为本文非常重要的章节，包括系统功能模块设计、业务模块设计以及数据库设计等。

第五章系统实现，本章节主要是对线上购物平台的前端、后端的主要功能实现流程进行介绍，结合页面和代码来讲解实现流程。

第六章系统测试，系统测试是每个系统开发结束前都必须要经历的过程，该章节主要是对系统测试流程和用例以及结果进行介绍。

## 1.5 本章小结

本章详细介绍了时代背景和技术背景，指出了当下存在的问题，然后阐述了本文的主要研究内容以及章节安排，使得读者能够了解本文的总体架构。

# 2 技术路线分析

线上购物平台开发采用IntelliJ IDEA为开发工具，数据库采用Navicat工具进行可视化展示，系统部署在Tomcat容器上，系统后端框架采用Spring Boot轻量级框架，前端采用Vue框架进行渲染。

## 2.1 开发工具

### 2.1.1 Navicat

本系统开发选取采用Navicat的原因是因为它是一款轻量级的可视化软件，没有厚重的环境要求，可随时进行变更以及替换版本[3]。其次，它支持多个不同数据库的连接，除常用的mysql外，还支持Oracle、PostgreSQL以及SQL Server等不同类型的数据库，方便系统更换数据库类型时不需要再替换软件，同时对于国内开发人员来说，Navicat是为数不多支持中文的数据库可视化软件，方便国内开发人员上手开发，最后对于设计表格来说非常便利，提供手动设计表格与修改表字段类型的界面，方便修改数据库表格，基于以上原因，本系统开发选用Navicat软件进行管理数据库。

### 2.1.2 IntelliJ IDEA

IntelliJ IDEA简称Idea，是Java编程语言开发的集成环境，本系统选取Idea软件进行开发带来了非常大的便利，首先，IntelliJ IDEA软件页面比较优美，开发体验良好，支持多语言多种颜色选择，十分人性化。其次，IntelliJ IDEA可以集成多个插件，比如Mybatis的Plus插件，支持自动生成增删改查配置文件，支持lombok插件，免去生成实体类Get、Set方法，为开发系统提供了极大的便利。最后，IntelliJ IDEA软件支持编码辅助和智能编辑等重要功能，可以在代码输入过程中，自动补充方法或类，极大程度上减少开发人员重复工作，非常适用于本系统的开发[4]。基于以上原因，本系统代码的开发采用IntelliJ IDEA软件进行。

### 2.1.3 Tomcat

本系统选取Tomcat作为部署容器是因为它具有非常便捷的部署操作，并且架构属于轻量级，不需要占用大量的内存空间，这样确保了项目的快速部署和启动。并且Tomcat属于开源的项目，不需要额外支付费用。基于以上原因，本系统部署容器选择Tomcat服务器上。

## 2.2 开发技术

### 2.2.1 Interceptor拦截器

Interceptor拦截器主要是基于SpringMVC框架实现，本系统主要是需要拦截器进行权限验证，本系统基于安全考虑，在用户登录后的每一次功能操作的时候需要对请求进行Token验证，解析Token来获取权限情况，分不同的状态，1、"401", "未获取到token, 请重新登录"。 2、"402""权限验证失败, 请重新登录" 。3、"403", "用户不存在, 请重新登录"。以此就可以清楚的了解用户信息。

### 2.2.1 Spring Boot框架

Spring Boot完全迁移利用了Spring的所有特性[5]，一大特性就是控制反转和依赖注入，音乐系统中依赖注入部分就是利用到该特性，后台之间模块关联的调用就是靠依赖注入直接引用，不需要每次都进行新建对象，比如在用户管理中，只需要在服务层注入一个用户对象，每个方法都可以调用，不需要每个方法重新新建一遍，可以释放系统内存，并且Spring框架会对该Bean的生命周期进行管理，在不使用的时候还会自动Kill掉，释放内存。

Spring Boot框架另外一大优势就是进行解耦，本系统集合了框架解耦的特性，将用户管理模块、商品管理模块等等模块解耦出来，各个模块之间关联较少，在一个模块出现问题后最小程度上减少对系统的影响性，同时继承了Spring框架轻量级的优点，整个系统打完War包部署到服务器上只有40兆大小，不占用内存空间[6]。

此外，SpringAOP也是该框架的核心，SpringAOP为面向切面编程，将维度划分为切面和点，采用该模块可以对日志和事务进行很好的集成，非常方便进行日志管理，本系统日志管理就是基于SpringAOP的基础上实现的。Spring框架拥有这些良好的框架和特性，为本系统的开发提供了诸多的便利，这也是本系统后端开发采用Spring Boot框架的原因之一。

### 2.2.2 Mybatis框架

MyBatis 本是apache的一个开源项目iBatis，Mybatis框架主要是用于系统和数据库之间交互使用，Mybatis框架同Spring框架进行集成，主要原理是采用了ORM的思想，将数据库字段和实体类字段进行一一映射，系统操作实体类数据，然后通过映射来传入到数据库中进行修改，本系统中在resource目录下配置好Mapper.xml映射，将实体类和系统的持久层代码结合起来，这样系统可以通过改变实体类的赋值来传递到数据库中，完成数据库的读写查操作。

### 2.2.3 Vue框架

本系统前端开发框架选取Vue框架进行开发，页面选取Html页面，通过CSS框架来调整页面样式，并且通过Javascript框架来进行方法调用。

Vue框架是一款轻量级的框架非常轻量级的框架，非常适合Web开发，并且同后端调用十分便利，Vue页面十分优美，并且市面上拥有非常多好看的模板，适合系统开发快速集成，本系统就基于这些优点所以决定选取Vue框架作为本系统前端框架选型。

## 2.3 本章小结

本章节对开发线上购物系统的工具和技术进行介绍，并且介绍选取这些框架和技术的原因，主要包括对开发工具Navicat、IntelliJ IDEA、Tomcat等工具进行介绍，对SpringBoot、Mybatis、Vue等框架进行了分析，便于读者理解。

# 3 系统分析

系统分析模块主要是对系统的需求以及可行性进行分析和介绍，介绍线上购物系统的需求转换的过程，并且分析该系统的开发可行性。

## 3.1 可行性分析

### 3.1.1 技术可行性

开发本系统所使用到的技术是否超出开发人员所掌握的临界是判断技术可行性是否达标的重要依据，如果超出则存在开发风险，如果符合开发人员技术栈则可以进行开发。本系统采用SpringBoot框架进行后台搭建，前端采用Vue框架来进行对接接口，这些框架是开发人员学习比较久且掌握的比较深的技术框架，并且对于Spring Boot框架来说，搭建起来非常便利，极大的简化了配置，让开发人员可以更加专注于功能开发上面，并且，本系统前后端分离，代码分割开来，互相不干扰，耦合性非常低，非常适合前后端一起进行开发，这样做可以让代码看起来更为清晰，逻辑结构更加清楚，不管是开发速度还是运维对接都非常便利，并且也没有打破开发人员技术壁垒，不会在技术上面造成困扰。

### 3.1.2 经济可行性

经济可行性是指系统开发成本和预计收益之间的关系，如果开发成本远大于预计收益时，就需要认真考虑该系统是否要进行立项开发，反之同样要客观对收益情况进行评估。对应到本系统的中就是，首先考虑本系统的开发成本，系统开发环境较为简易，不需要占用大量硬件设备，同时采用Spring Boot框架来进行搭建，极大程度上简化了搭建步骤，这样可以最大程度上减少开发成本。从预估收益上来说，本系统可以极大程度上弥补市场当前的空缺，吸引商家入驻，下发广告来获取利润，在巨大的市场环境下可以获取很大的收益，综合考虑来看，系统的经济可行性分析结果是可行的。

### 3.1.3 法律可行性

网上购物系统在法律上是有专门规定的，必须对卖家工商营业执照进行审核后才能进行售卖非违禁物品。所以本系统在进行使用的时候需要对商家信息进行审核就可以符合法律规定，不会触犯法律所禁止的。

### 3.1.4 操作可行性

操作可行性是指系统在上线和发布后系统操作难度和系统使用复杂度。本系统通过Spring Boot框架进行搭建，所以可以打包成Jar包直接部署在服务器上运行，对于运维人员是非常友好的，不需要再配置Tomcat环境以及JDK环境，直接运行Jar包即可，省去了非常多的操作。系统基于B/S架构运行在Web端上，用户只需要打开界面就可以使用，支持在线购物，即使不用登录也能很好的使用，并且页面进行了良好的UI设计，使用性上面符合正常人操作，适用年龄非常广泛。并且对于系统性能来说，本系统对后台进行了优化，可以确保多人在线时系统也不会进行卡顿，并且各自分离开来，不会影响彼此的操作，对于操作上面非常容易。

对管理者和用户两方面来说，本系统基于B/S架构，只需要对系统简单的部署调试就能够正常使用，系统界面美观大气，不需要经过单独的学习就能够进行功能模块上的操作，所以在使用层面上是没有难度的。对于运维人员来说，系统需要打好war包部署到Tomcat容器上即可，并且后期运维只需要进行War包替换，操作上节省了很多步骤。

## 3.2 需求分析

需求分析是系统开发前非常重要的阶段，线上购物系统需求分析主要从两方面进行，分别是功能性需求和非功能性需求。

### 3.2.1 功能性需求分析

线上购物系统根据功能不同，将系统分为两个角色，分别是普通用户角色和管理员角色，普通用户角色主要是线上购物系统前台的使用者，主要功能点包括登录注册、个人中心、选购商品、收藏夹、我的订单、地址管理等功能。功能用例图如下图3-1所示。

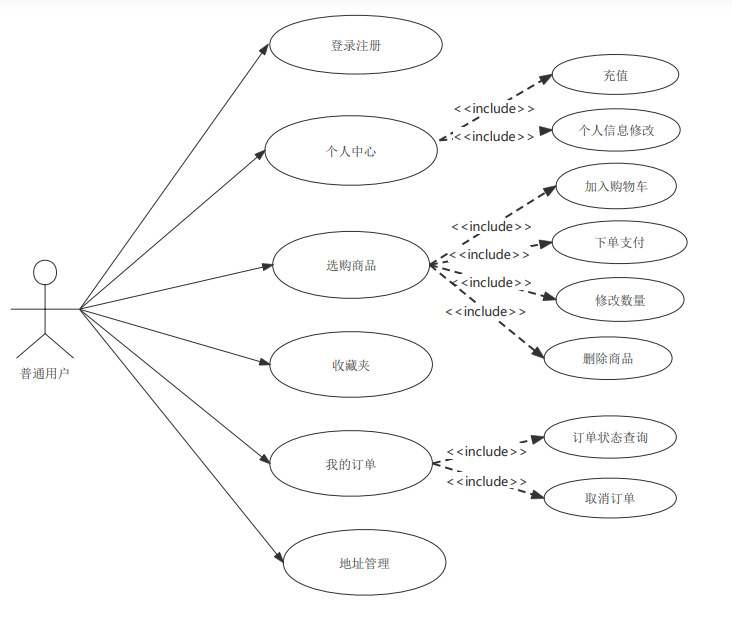


图3-1 普通用户用例图

管理员用户是线上购物系统的后台使用者，功能主要是对前台信息以及用户信息进行管理操作，主要功能包括用户管理、权限管理、收货地址管理、公告管理、日志管理、商品管理、订单管理。其中权限管理又分别包括角色管理和菜单管理，两者结合组成本系统的权限管理功能。商品管理又分别是对商品信息管理以及商品分类管理两方面。管理员功能用例图如下图3-2所示。

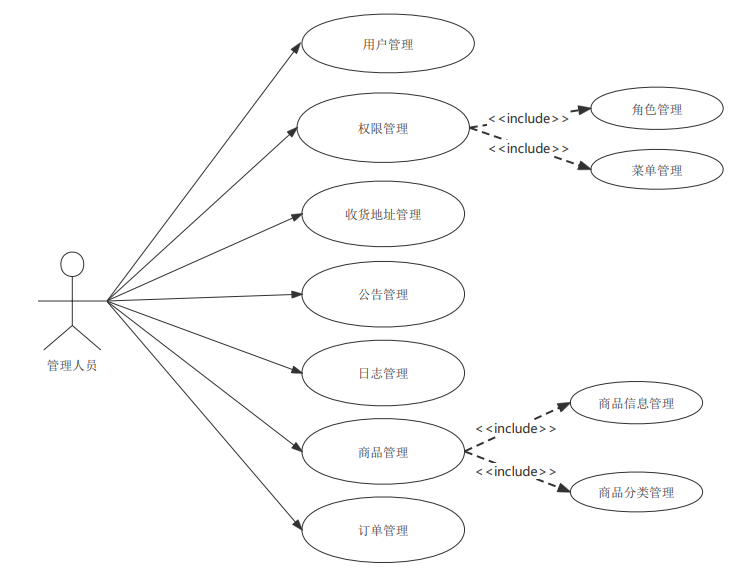


图3-2 管理员用例图

### 3.2.2 非功能性需求分析

非功能需求不仅与软件系统本身有关，还与系统的开发过程有关。非功能性需求主要是从系统的正确性、健壮性、性能、安全性、兼容性等方面进行分析和考虑。

正确性：正确性要求系统不允许出现业务上的错误，并且在数据发生多次变动后扔保持数据不发生实质上的错误，确保系统的正确运行。在本系统中，系统集成事务管理，在数据因意外断开时可以发生回滚操作，同时Mybatis框架确保了数据多次变动后的前后顺序，确保数据正确性。

健壮性：健全性要求系统具有一定的应对错误的能力，健壮性强的系统可以应对突发的错误情况，例如突然的断电、断网以及人为的进行错误数据攻击等情形扔保持程序不会突然的崩溃，拥有正确的处理和适当的回避机制。

安全性：安全性要求系统必须严格控制访问权限，确保用户必须通过认证后才能获取对应的权限，阻拦企图绕开安全防线破坏系统的操作。本系统在前端登录中对密码进行了防护，确保数据的安全性，同时Session设置失效时间，避免时间过长导致密码泄露。

## 3.3 本章小结

本章节从可行性分析进行介绍，系统分析模块主要是对系统的需求以及可行性进行分析和介绍，介绍线上购物系统的需求转换的过程，并且分析该系统的开发可行性。

# 4 系统设计

系统设计主要包括系统功能模块设计、业务模块设计以及数据库设计等。根据系统需求分析来进行模块设计。

## 4.1 系统架构设计

线上购物商城系统的架构图如下图4-1所示。

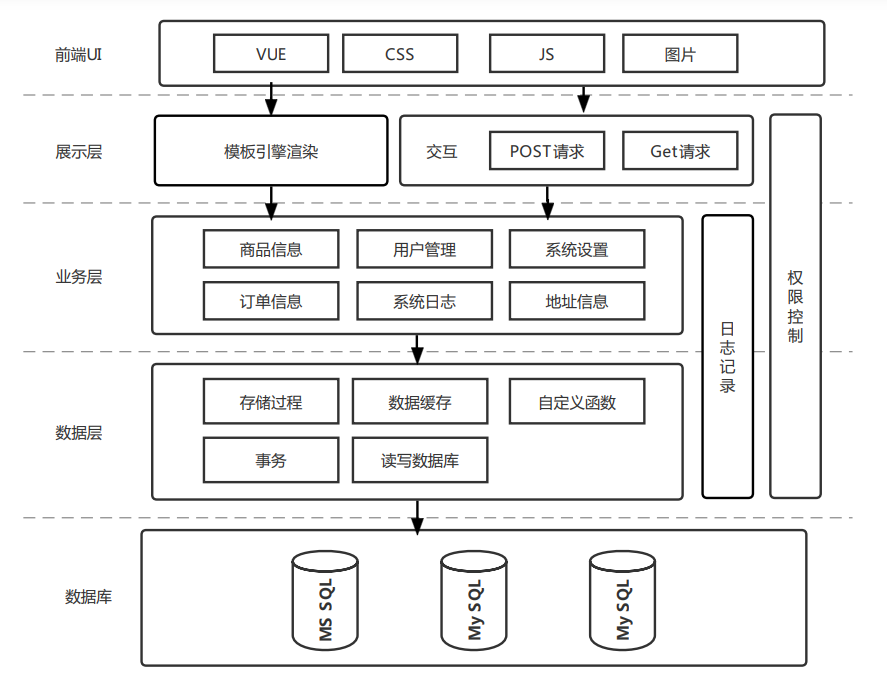


图4-1 系统架构图

前端的UI和展示层组成的系统前端结构，本系统前端开发框架选取Vue框架进行开发，页面选取Html页面，通过CSS框架来调整页面样式，并且通过Javascript框架来进行方法调用。

系统后端采用MVC设计思想，将后端框架进行分层，展示层拦截前端请求进行下发，业务层进行业务处理，数据层与数据进行交互。数据库采用MySql数据库来存储信息，通过MyBatis框架来完成映射和系统完成数据交互。

并且系统集成日志记录和权限控制能功能，采用SpringBoot框架继承Spring框架AOP控制切面编程的组件，来对日志信息进行管理，通过切面切点来集成到系统后端中，完成日志的管理，可以详细记录每一步系统的操作和类的对象，既方便运维人员跟踪又方便开发人员进行调试。权限控制则是采用Security安全控制框架集成SpringBoot框架来进行实现，通过获取Token来进行权限管理，增加系统的安全性，避免恶意攻击。

## 4.2 功能模块设计

功能模块设计在需求分析的基础上进行延伸，线上购物系统根据界面不同将功能模块分为两个模块，分别是前台功能模块和后台功能模块。

### 4.2.1 前台功能模块

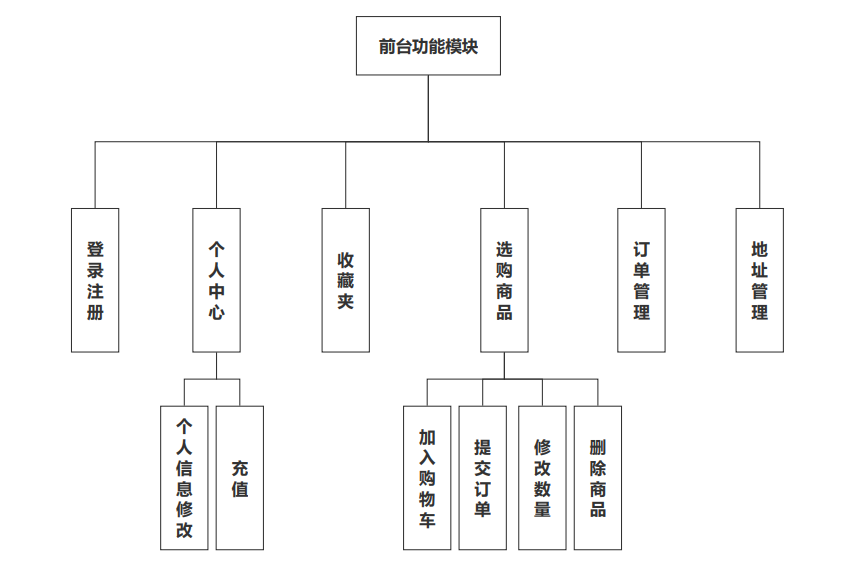


图4-2 前台功能模块图

登录注册：包括注册和登录功能，没有注册之前只能以游客形式访问本系统的前端页面，并且不支持选购商品、订单管理、地址管理等一系列功能，注册后才可以获取账户并且登录使用。

个人中心：个人中心由普通用户登录后使用，包括个人信息修改和充值功能，充值功能可以给自己账户进行充值，个人信息修改包括头像上传和修改密码等功能。

收藏夹：在用户登录后可以在收藏夹中收藏商品信息，并且可以在收藏夹里直接跳转到商品详情页面中去，还可以取消收藏等功能。

选购商品：选购商品是对购买商品包含的一系列流程，包括加入购物车、提交订单、修改数量、删除商品等功能。

订单管理：在用户登录后可以在我的订单中对订单信息进行管理，包括查询订单信息，对未付款订单进行付款，删除订单以及取消订单等功能，在商家发货后买家可以点确认收货来改变订单状态。

地址管理：对地址信息进行统一管理。

### 4.2.2 后台功能模块

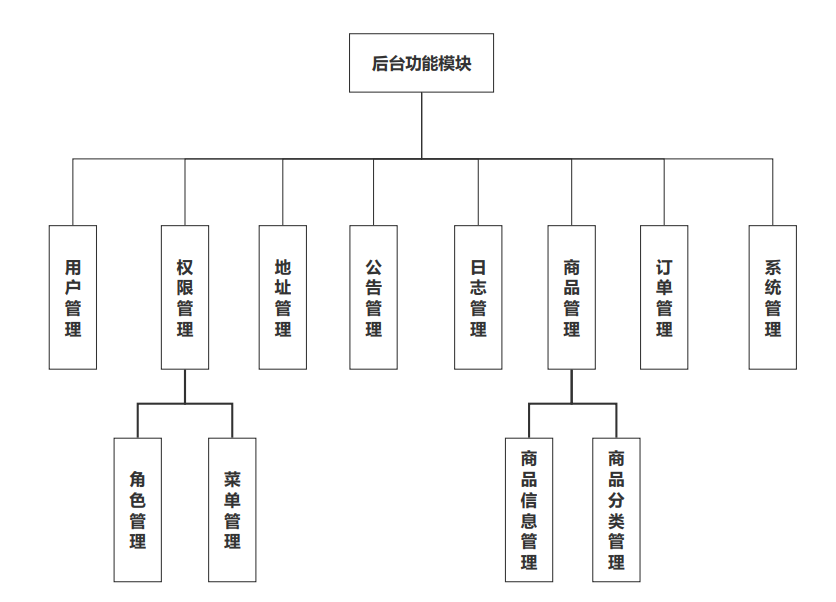


图4-3 后台功能模块图

1、用户管理：主要包括对用户信息进行维护，包括管理员用户信息和普通用户信息。在该界面中修改用户信息包括可以对角色也进行修改。

2、权限管理：权限管理包括角色管理和菜单管理，对系统功能菜单进行管理，在该模块中可以对系统功能菜单信息进行维护，可以开启和关闭功能权限。对系统角色进行划分，可以在该模块中新建和删除角色信息，并且根据角色在菜单管理中分配权限。  
 3、地址管理：对收货地址信息进行管理，包括对所有用户的地址信息进行管理。

4、公告管理：公告管理由后台发布和使用，本公告信息仅有后台管理人员进行查看，以及对公告信息进行维护。

5、日志管理：对系统操作日志进行查看和查询，在后台中的操作功能都会在日志中记录下来。

6、商品管理：包括商品分类以及商品信息进行管理。

7、订单管理：对用户产生的订单信息进行校验，并且根据订单状态安排发货。

8、系统管理：对系统轮播图以及管理员信息进行维护。

## 4.3 业务流程设计

### 4.3.1 用户登录注册

用户进入注册页面后输入信息可以进行注册，但是用户名不能相同，否则会注册失败，注册后的账户可以进行登录获取预定权限。

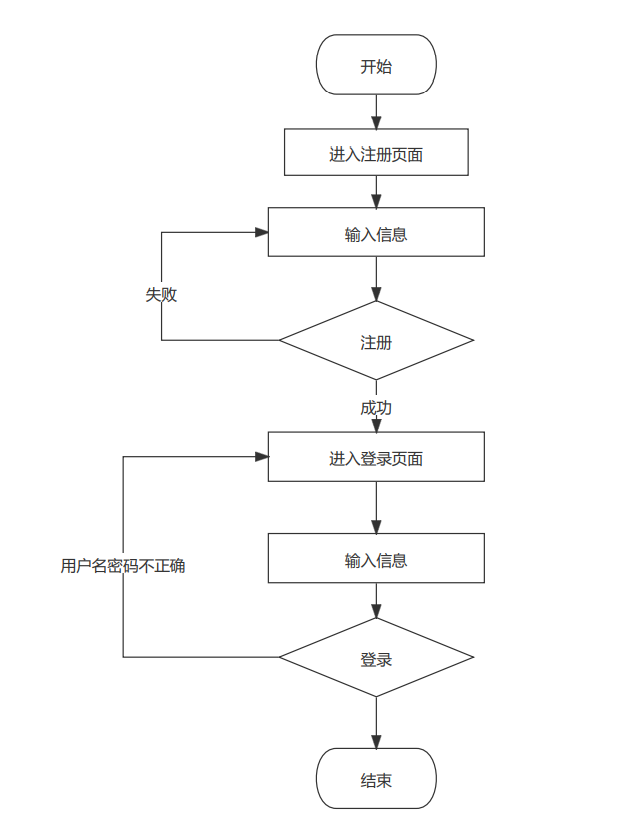


图4-2 注册登录流程图

注册流程主要是系统会对用户的账户进行相似性校验，确保每个账户只能有一个账号，避免问题发生。用户第一次使用系统需先进行注册操作后，待注册完成输入注册的用户名和密码就能对系统进行登录。

### 4.3.2 购买商品

用户登录后进入到商品信息页面，点击购买进行付款就可以进行购物，购买成功后会生成对应的订单信息，业务流程图如下。

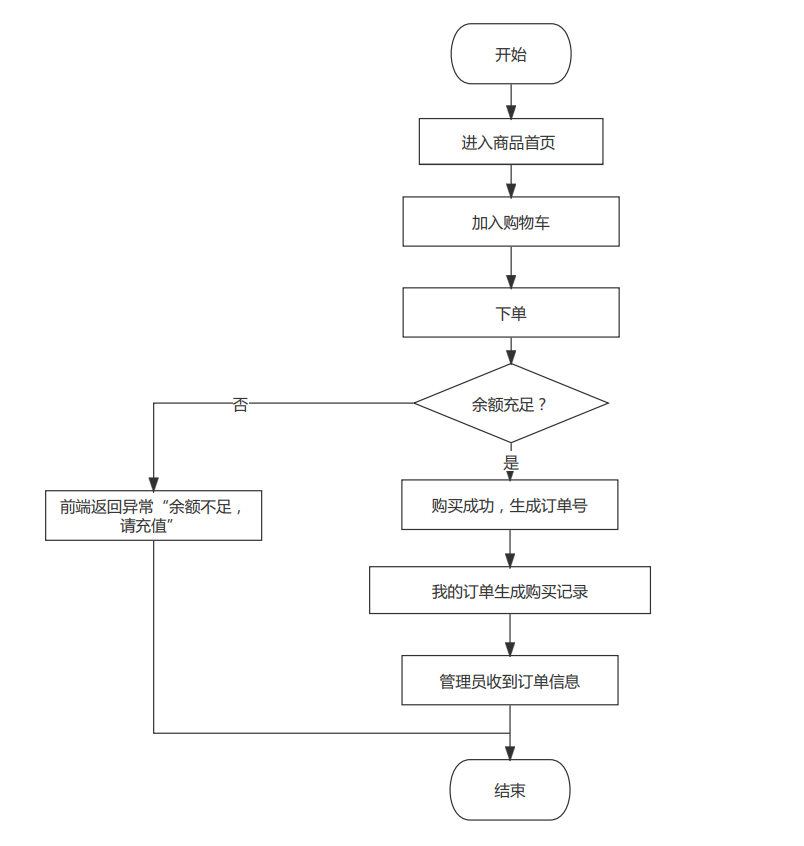


图4-3 购买商品流程图

### 4.4.3 修改密码

用户在登录后进入到个人中心后可以点击修改密码，并且输入的密码必须确保两次不一致才能修改成功。业务流程图如下。

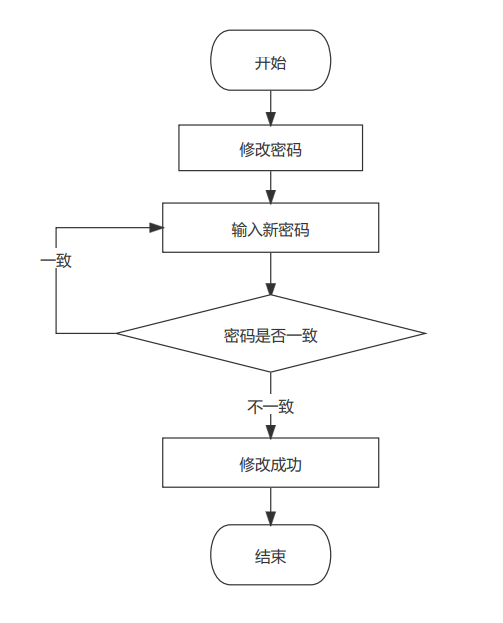


图4-4 修改密码流程图

### 4.4.4 商品收藏

用户登录后进入到商品详情页面中可以对商品信息进行收藏，流程如下图所示。

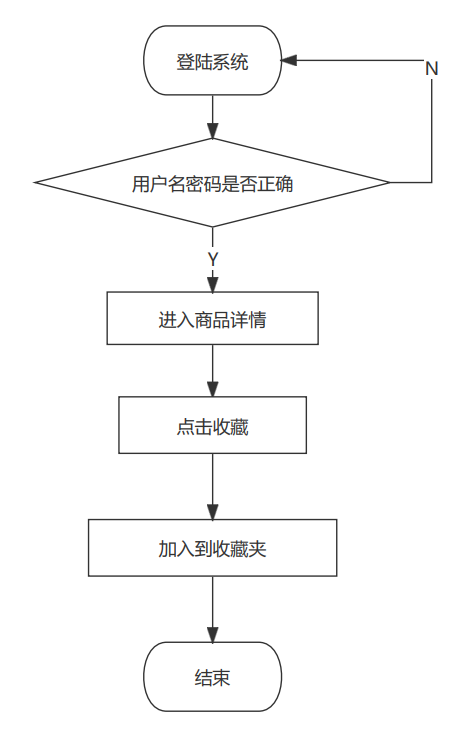


图4-5 商品收藏流程图

## 4.4系统数据库设计

### 4.3.1 数据库概念设计

线上购物平台的E-R图如下图所示：

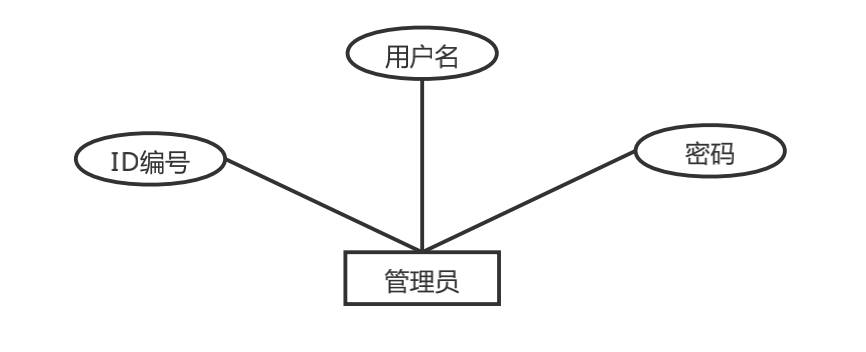


图4-6 管理员实体属性图

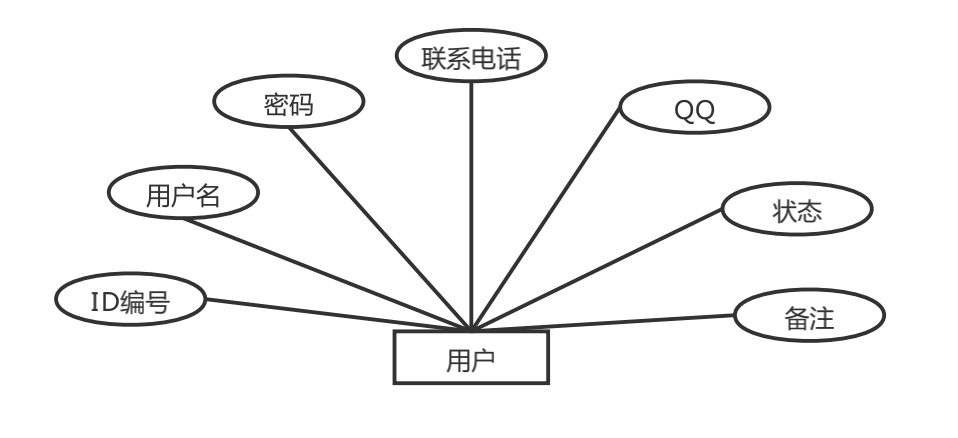


图4-7 用户实体属性图

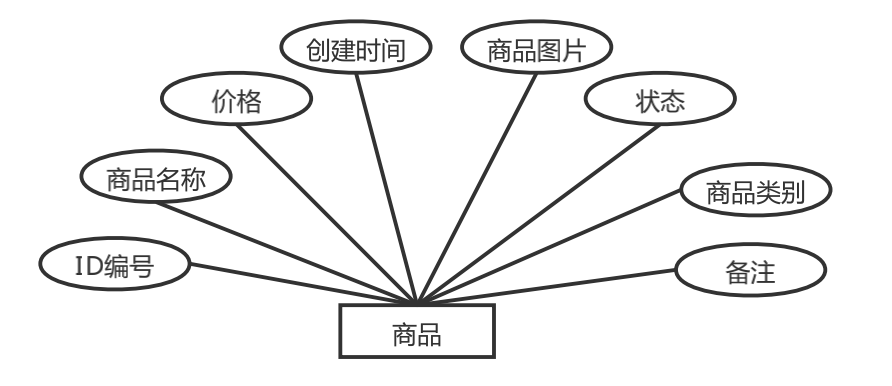


图4-8 商品实体属性图

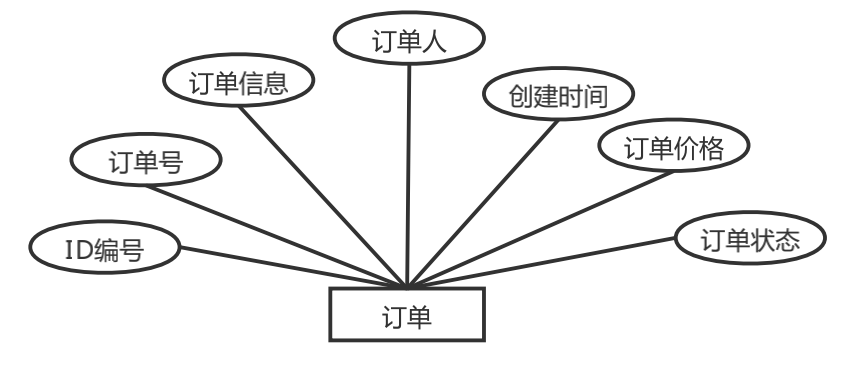


图4-9订单实体属性图

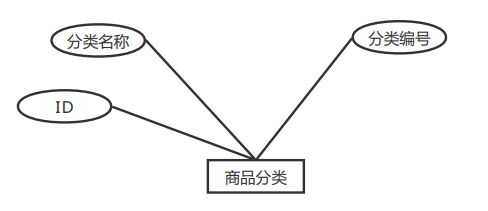


图4-10 商品分类实体属性图

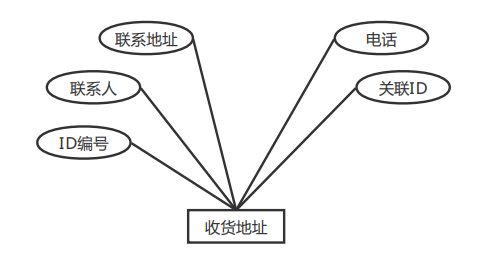


图4-11 收货地址实体属性图

本系统的总体ER图如下图4-12所示：

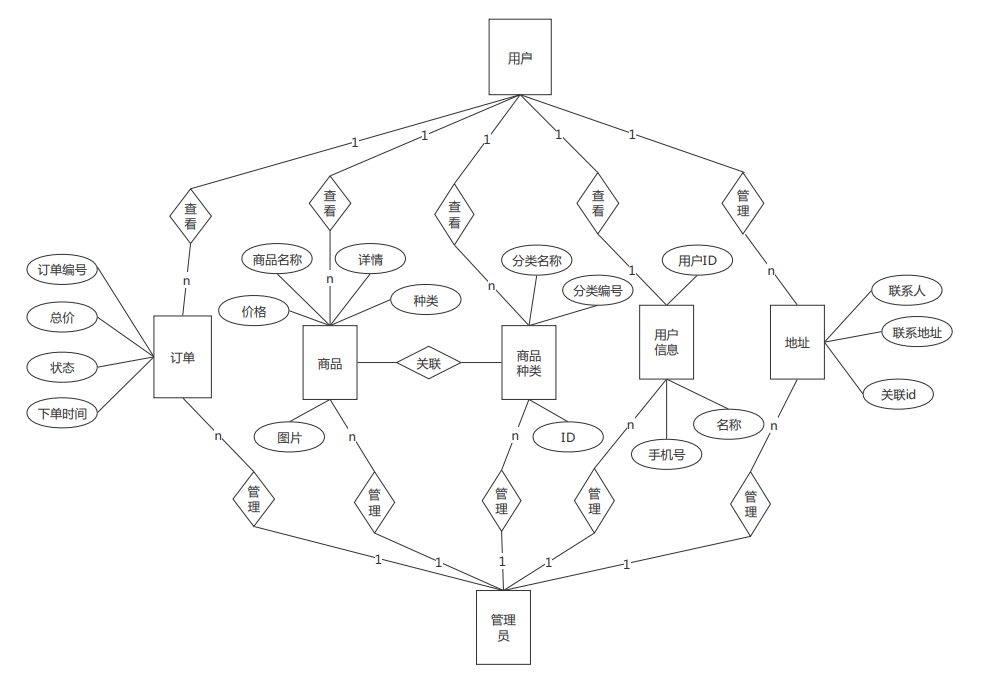


图4-12 系统总体E-R图

### 4.3.2 数据库表设计

1.管理员表

管理员表主要存储着管理员信息，id为表主键，唯一且不为空，属于自增序列，主要字段包括用户名、密码等字段。

表4-1 管理员表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 是否为主键 | 字段解释 |
| id | int | 是 | 主键 |
| username | varchar | 否 | 用户名 |
| password | varchar | 否 | 密码 |

2 用户表

用户表主要存储着用户信息，id为表主键，唯一且不为空，属于自增序列，主要字段包括用户名、密码等字段。其中用户状态字段1表示正常 0表示已停用。

表4-2 用户表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 是否为主键 | 字段解释 |
| user\_id | int | 否 | 主键 |
| user\_qq | varchar | 否 | QQ号 |
| user\_status | varchar | 否 | 用户状态 |
| user\_phone | varchar | 否 | 电话 |
| user\_beizhu | varchar | 否 | 备注 |
| user\_username | varchar | 否 | 用户名 |
| user\_pwd | varchar | 否 | 密码 |

3 商品表

商品表主要存储着商品信息，id为表主键，唯一且不为空，属于自增序列，主要字段包括商品名称、商品类别、备注、状态等字段。其中商品状态字段1表示正常 0表示已下架。

表4-3 商品表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 是否为主键 | 字段解释 |
| goods\_id | int | 否 | 主键 |
| goods\_img | varchar | 是 | 商品图片 |
| goods\_name | varchar | 否 | 商品名称 |
| goods\_category | varchar | 否 | 商品类别 |
| goods\_beizhu | varchar | 否 | 备注 |
| goods\_price | varchar | 否 | 原价 |
| goods\_price\_new | varchar | 否 | 现价 |
| goods\_time | varchar | 否 | 上架时间 |
| goods\_stauts | varchar | 否 | 商品状态 |

4 地址表

地址表主要存储着地址信息，id为表主键，唯一且不为空，属于自增序列，主要字段包括联系人、联系地址、关联人等字段。

表4-4 地址表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 是否为主键 | 字段解释 |
| id | int | 是 | 主键 |
| link\_user | varchar | 否 | 联系人 |
| link\_address | varchar | 否 | 联系地址 |
| link\_phone | varchar | 否 | 联系电话 |
| user\_id | varchar | 否 | 所属用户 |

5.订单表

订单表主要存储着订单信息，id为表主键，唯一且不为空，属于自增序列，主要字段包括用户ID、订单信息、订单状态等字段。

表4-5 订单表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段类型 | 是否为主键 | 字段解释 |
| order\_id | int | 是 | 主键 |
| order\_person\_id | varchar | 否 | 用户ID |
| order\_shop\_id | varchar | 否 | 购买者ID |
| order\_message | varchar | 否 | 订单信息 |
| order\_addtime | varchar | 否 | 新增时间 |
| order\_stauts | varchar | 否 | 订单状态 |
| order\_status\_pay | varchar | 否 | 订单付款状态 |

## 4.4 本章小结

本章节主要是对系统架构以及功能模块进行分析，然后对数据库概念设计以及数据库表结构设计进行描述。

# 5 系统功能实现

线上购物系统主要分为前台后台两个模块，功能实现上面也分别从这两个模块进行介绍。

## 5.1 前台功能实现

### 5.1.1 注册功能实现

注册功能是本系统前端用户获取账户的唯一途径，没有账户是不能对本系统进行使用，本系统拦截器会拦截下来所有没有登录的请求，注册页面如下图所示。

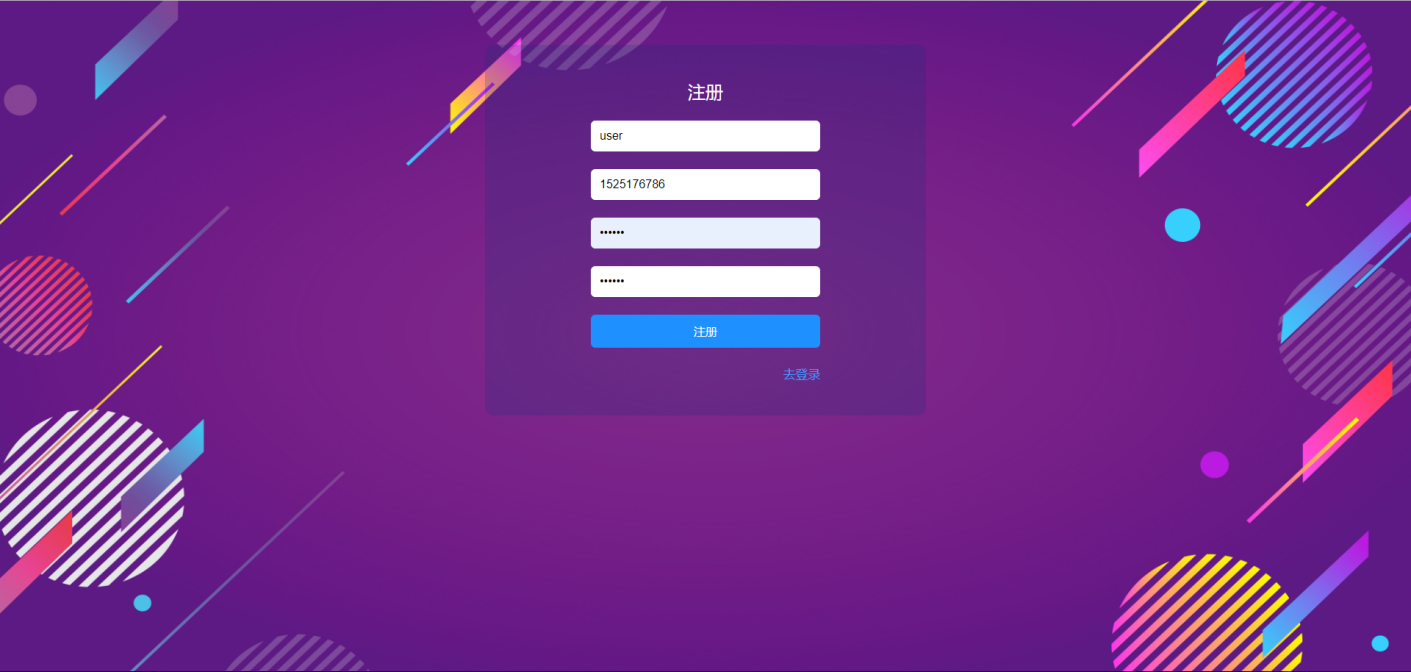


图5-1 注册页面

代码如下：

@PostMapping("/register")

public Result<User> register(@RequestBody User user, HttpServletRequest request) {

if (user.getPassword() == null) {

user.setPassword("123456");

}

User dbUser = userService.register(user);

request.getSession().setAttribute("user", user);

logService.log(user.getUsername(), StrUtil.format("用户 {} 注册账号成功", user.getUsername()));

return Result.success(dbUser);

}

实现流程：

1、前端将输入的信息通过Form表单的形式传递到后端后，会被后端接口拦截下来，然后分配到用户相关的控制层中的register方法中去，用户信息会被@RequestBody User user实体类接收。

2、首先会对密码信息进行验证，如果密码为空则为其输入默认密码“123456”，如果密码不为空则继续对用户信息进行判断。

3、系统调用userService.register(user);业务层中的方法，业务层会对用户名先进行判断，查询是否已经存在该用户，如果存在则返回用户名已存在，请重新注册的提示。如果不存在则新增。

4、调用user.setRole(CollUtil.newArrayList(2L))方法将用户默认为普通用户角色，然后对数据库进行交互，将信息保存。

5、注册流程主要是系统会对用户的账户进行相似性校验，确保每个账户只能有一个账号，避免问题发生。用户第一次使用系统需先进行注册操作后，待注册完成输入注册的用户名和密码就能对系统进行登录。

### 5.1.2 登录功能实现

注册后的账户可以在登录界面进行登录，本系统只有进行登录后才能获取使用权限。登录页面如下图所示。

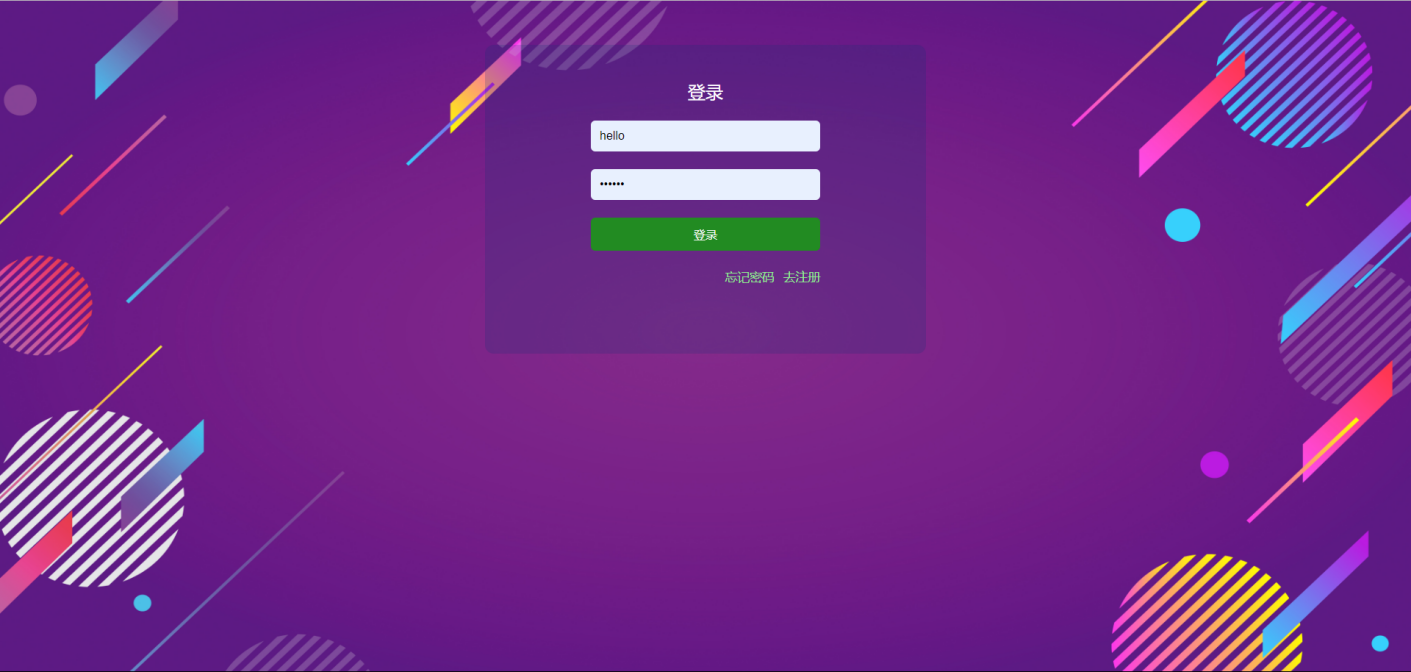


图5-2 登录页面

代码如下：

@PostMapping("/login")

public Result<User> login(@RequestBody User user) {

User res = userService.login(user);

// 生成token

String token = JWT.create().withAudience(res.getUsername()).sign(Algorithm.HMAC256(res.getPassword()));

res.setToken(token);

logService.log(user.getUsername(), StrUtil.format("用户 {} 登录系统", user.getUsername()));

return Result.success(res);

}

实现流程：

1、用户输入账户名密码后前端将信息传递到后端去，系统通过HTTP的方式进行调用，前端查询路径为/login的API接口。

2、后端接口请求首先会被UserController的login()拦截下来，该方法主要是对登录信息进行验证，首先会先通过用户名来到数据库查询数据，获取到该用户的用户名和密码。获取到用户名密码后调用getOne信息来对数据进行解析。

3、解析后的数据调用SecurityUtils.getSubject()根据安全框架来进行验证，对密码进行解密，然后同数据库进行比对。

4、比对无误后返回结果，如果不一致则返回登陆失败和异常信息，本系统对异常信息进行了统一处理。如果一致则根据用户名和密码生成Token然后返回到前端中去，设置Token失效时间，接下来每一步操作都会对Token进行验证。

对于Token生成，本系统集成JWT来实现Token生成，采用JWT.create()方法来生成Token然后重写拦截器来拦截所有请求验证Token。

在用户登录后的每一次功能操作的时候需要对请求进行Token验证，解析Token来获取权限情况，分不同的状态，1、"401", "未获取到token, 请重新登录"。 2、"402""权限验证失败, 请重新登录" 。3、"403", "用户不存在, 请重新登录"。以此就可以清楚的了解用户信息。

登录成功后系统会跳转到前台首页去。页面如下图所示。



图5-3 前台首页

### 5.1.3 收藏功能实现

用户登录后进入到商品详情页面中可以对商品信息进行收藏，页面如下图所示。



图5-4 商品详情页面

点击右上交的收藏按钮，可以将商品信息加入到收藏夹中，点击点赞按钮，可以对商品信息进行点赞，点赞多的商品会优先进去到商品推荐模块中去。点击收藏后可以在用户收藏夹中查看到商品信息，页面如下图所示。

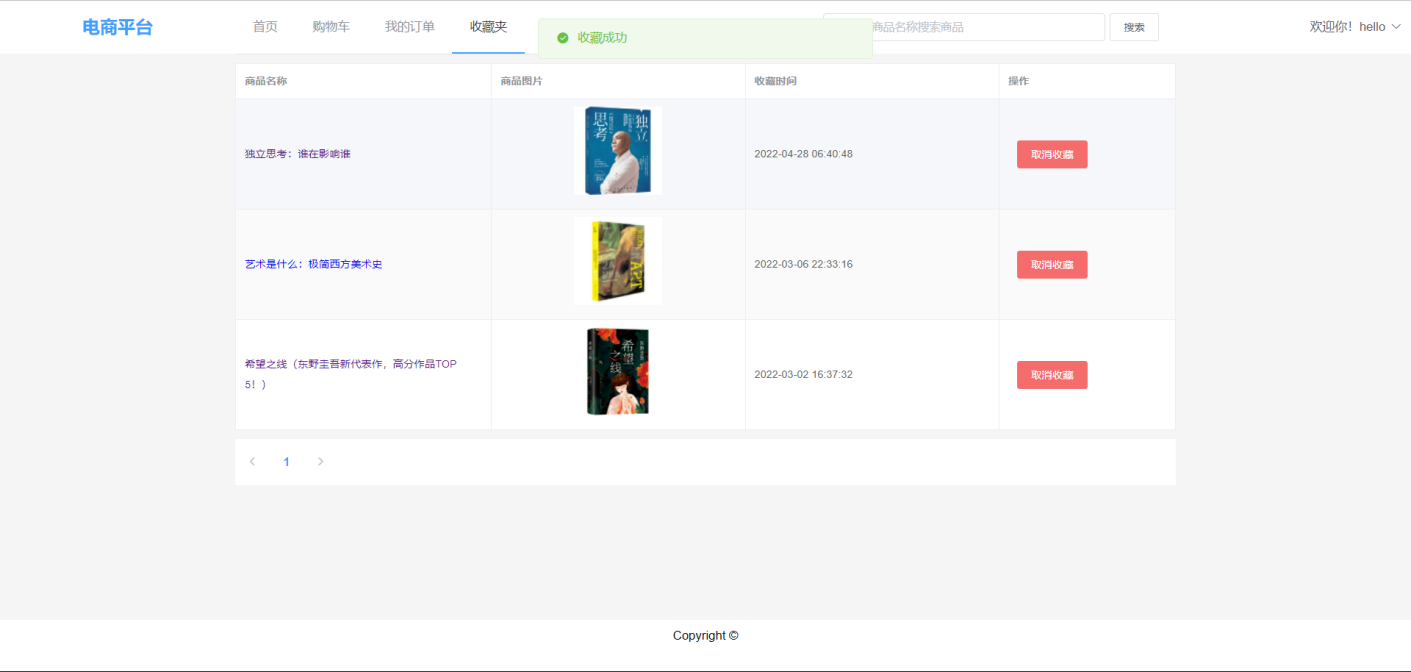


图5-5 收藏夹列表

代码如下：

@PostMapping

public Result<?> save(@RequestBody Collect collect) {

List<Collect> list = collectService.list(Wrappers.<Collect>lambdaQuery().eq(Collect::getGoodsId, collect.getGoodsId())

.eq(Collect::getUserId, getUser().getId()));

if (CollUtil.isNotEmpty(list)) {

throw new CustomException("-1", "您已收藏该商品");

}

collect.setCreateTime(DateUtil.now());

collectService.save(collect);

return Result.success();

}

实现流程：

收藏商品信息前会先对商品信息进行查询判断，通过调用collectService.list方法来查询出所有的商品信息，采用Java1.8以上才独有的Stream()流的方法来对商品信息和收藏夹信息进行比对，判断该商品是否已经存在，已经存在则返回"您已收藏该商品"，如果不存在则对数据进行处理，然后调用业务层collectService.save(collect);方法来进行新增。

### 5.1.4 商品下单功能

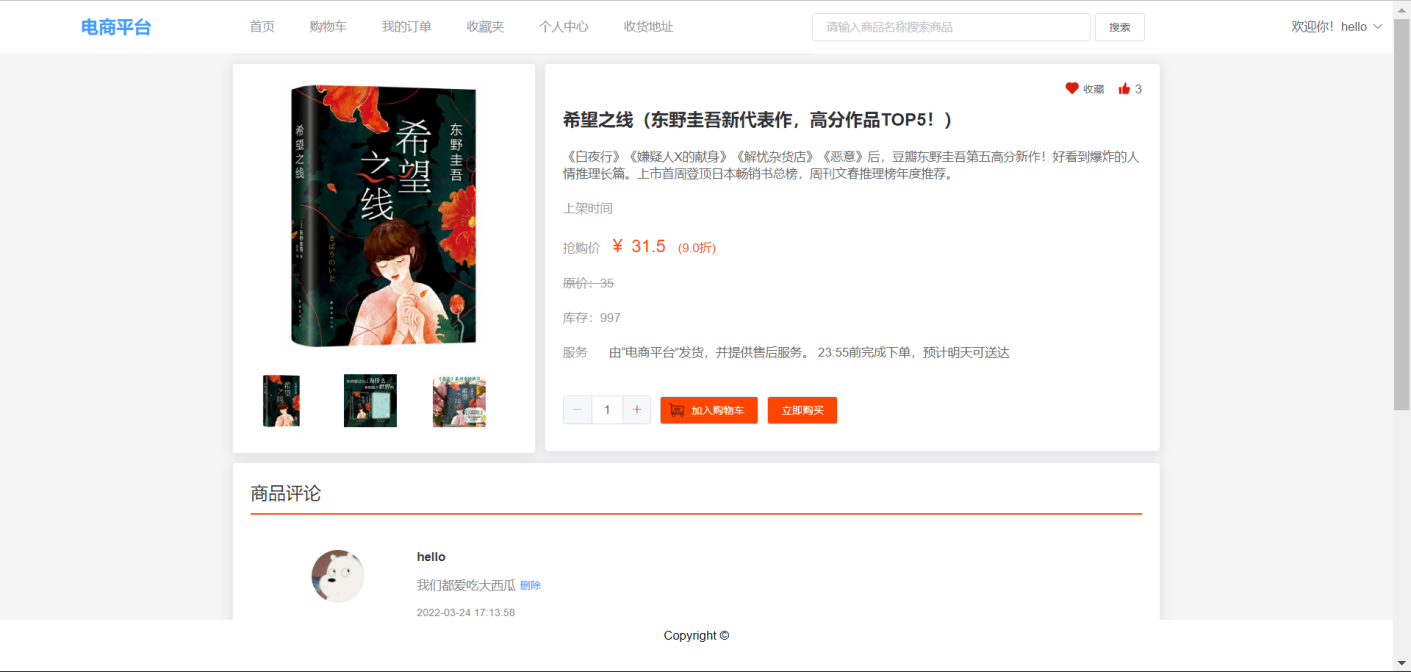


图5-6 商品详情页面

商品下单功能分两种方式实现，第一点击立即购买，系统会直接生成订单，可以进行付款提交订单。第二先加入到购物车中，然后在购物车中统一进行结算，提交订单，进行付款，最后在生成订单信息。

点击立即购买后页面如下图所示。

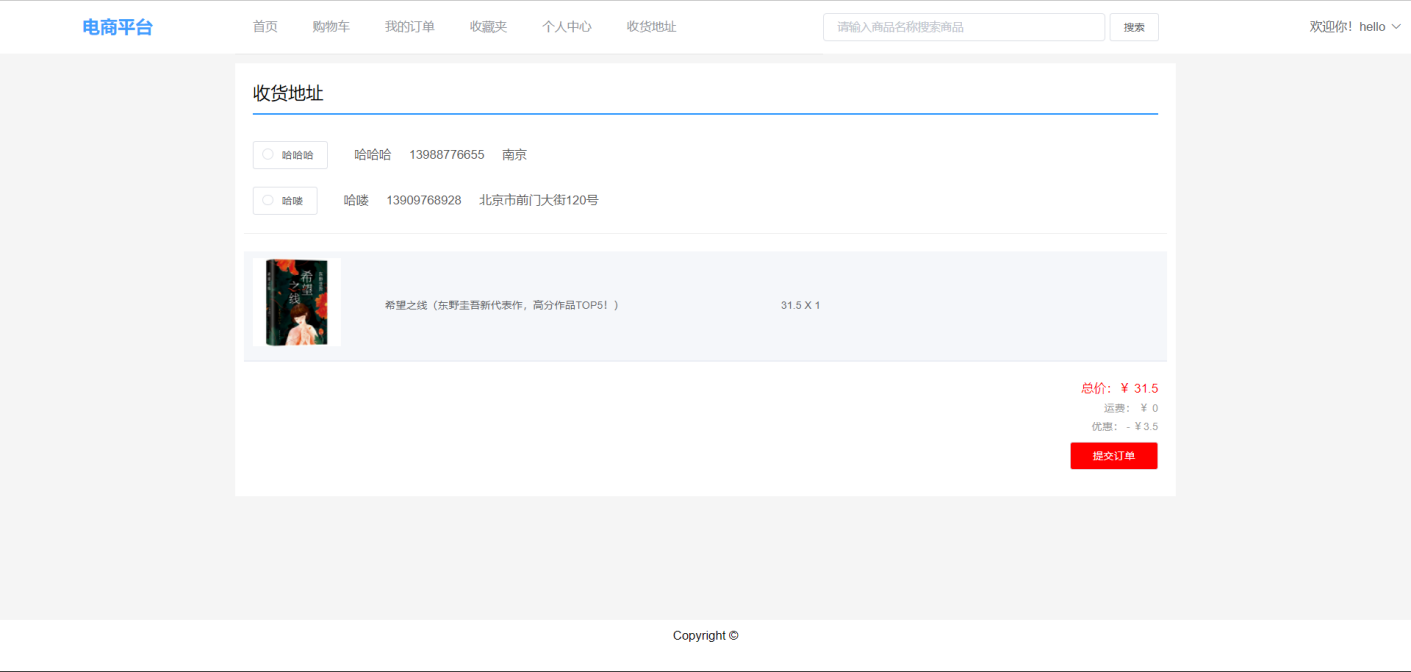


图5-7 提交订单页面

代码如下：

List<Cart> carts = JSONUtil.toBean(cartsStr, new TypeReference<List<Cart>>() {

}, true);

orderService.save(order);

for (Cart cart : carts) {

Integer count = cart.getCount(); //计算购物车中商品数量

Long goodsId = cart.getGoodsId(); //保存商品信息

// 扣库存

Goods goods = goodsService.getById(goodsId); //根据商品ID来查询商品信息

if(goods.getStore() - cart.getCount() < 0) { //判断库存是否充足

throw new CustomException("-1", "库存不足"); //库存不足，返回购买失败

}

goods.setStore(goods.getStore() - cart.getCount()); //库存充足则减去库存信息

goods.setSales(goods.getSales() + cart.getCount());

goodsService.updateById(goods);//更新商品数量

OrderGoods orderGoods = new OrderGoods(); //模型转换

orderGoods.setOrderId(order.getId());

orderGoods.setGoodsId(goodsId);

orderGoods.setCount(count);

orderGoodsService.save(orderGoods);

}

if (order.getType() == 1) { // 1表示购物车，0表示直接购买

// 提交订单时清空个人的购物车商品

cartService.remove(Wrappers.<Cart>lambdaUpdate().eq(Cart::getUserId, getUser().getId()));

}

return Result.success(order);

除去直接下单外，还有一种购买方式是先加入到购物车中，在商品详情页面中可以将商品信息添加到购物车中，可以添加数量来加入到购物车，数量不能超过库存总量。加入后会在购物车页面查看到商品信息，页面如下图所示。

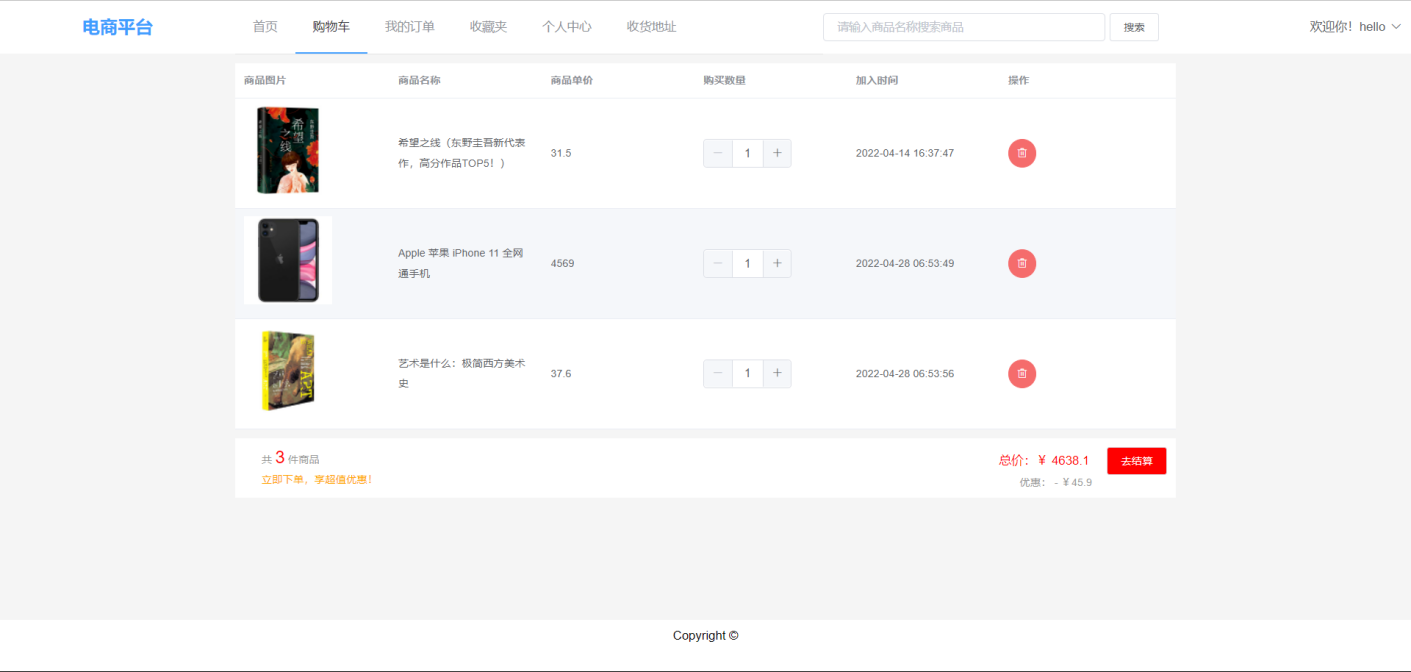


图5-8 购物车页面

在购物车中可以对商品总价进行计算，查询购物车信息代码如下：

@GetMapping("/page")

public Result<?> findPage(@RequestParam(required = false, defaultValue = "") String name,

@RequestParam(required = false, defaultValue = "1") Integer pageNum,

@RequestParam(required = false, defaultValue = "10") Integer pageSize) {

LambdaQueryWrapper<Category> query = Wrappers.<Category>lambdaQuery().orderByDesc(Category::getId);

if (StrUtil.isNotBlank(name)) {

query.like(Category::getName, name);

}

IPage<Category> page = categoryService.page(new Page<>(pageNum, pageSize), query);

return Result.success(page);

}

实现流程：

点击购物车后会将请求传递到后端中，被购物车@RequestMapping标签拦截下来，分配到对应的方法中去。

拦截下来后会调用cartService.list方法来进行查询，最后将信息返回。

点击马上付款可以跳转到付款界面，在付款界面可以选择或者新增收货地址。页面如下图所示。

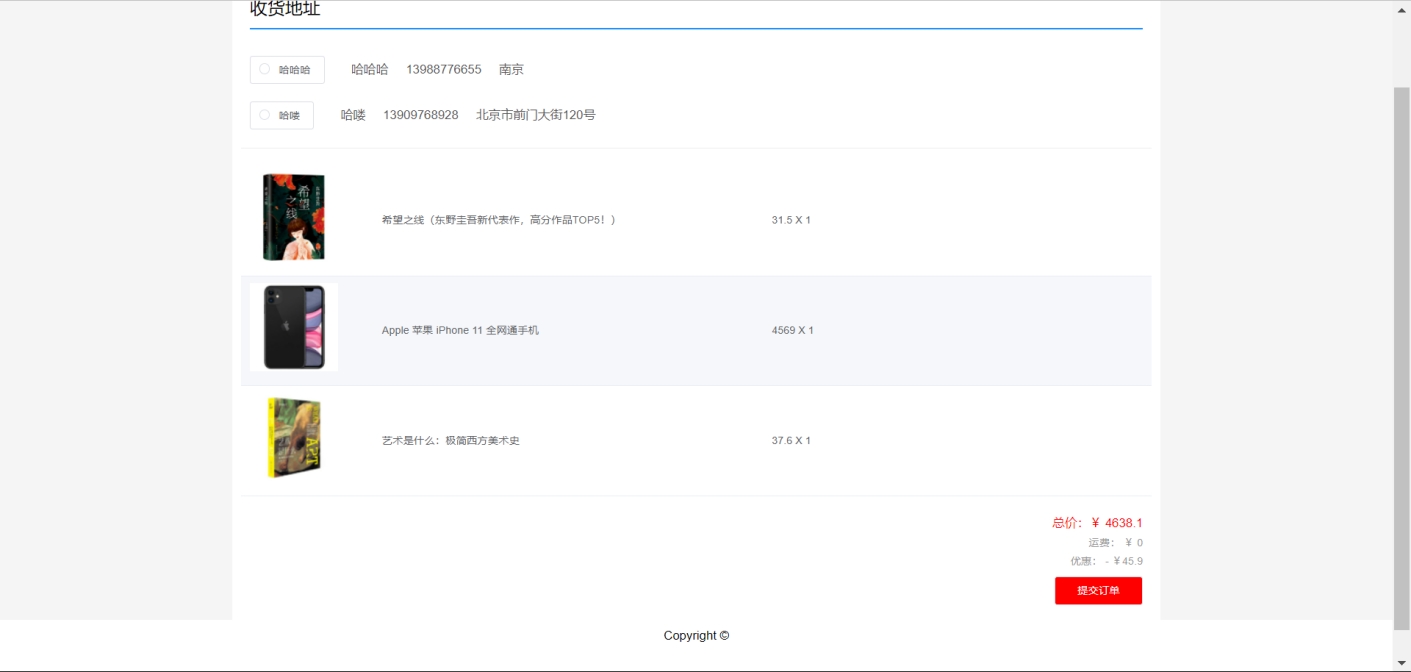


图5-9 待付款界面

点击提交订单后可以完成购物。页面如下图所示。



图5-10 订单列表

## 5.2后台功能模块

### 5.2.1 分类商品销量统计功能

本系统集成E-Charts框架来实现数据统计功能，页面如下图所示。

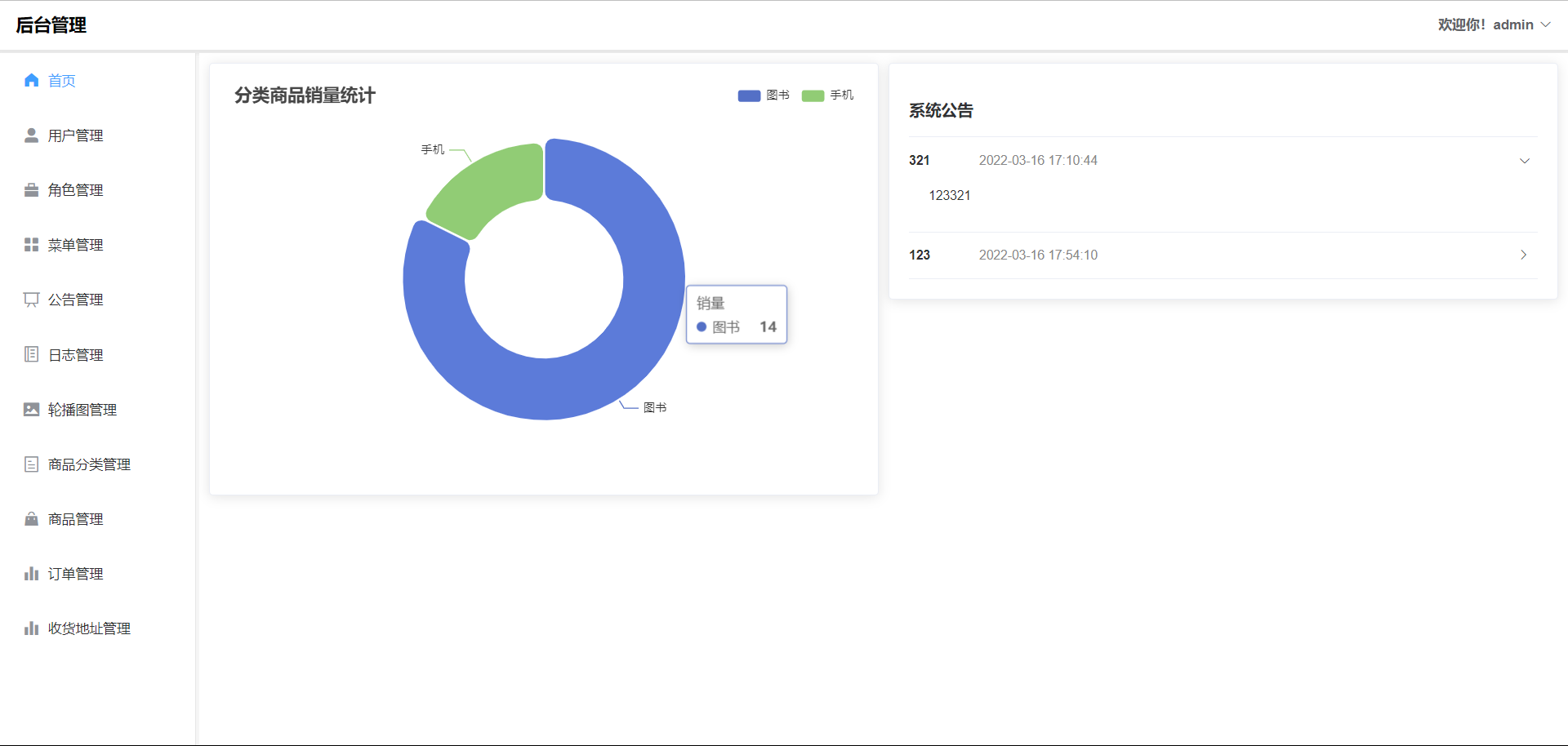


图5-11 分类商品销量统计

实现方式是集成E-Chars框架来进行图表展示，首先前端通过定义函数方法将系统的数据进行获取，查询数据的流程跟其他的功能模块类似，将数据从数据库中取出后设置对应的维度信息，、以及横纵坐标信息，然后通过E-Chars框架来形成表格页面呈现出来[15]。

**主要代码：**

@GetMapping("/page")

//获取分类名称参数

public Result<?> findPage(@RequestParam(required = false, defaultValue = "") String name,

//获取分页页数参数

@RequestParam(required = false, defaultValue = "1") Integer pageNum,

//获取分页显示数量参数

@RequestParam(required = false, defaultValue = "10") Integer pageSize) {

//调用分类业务的查询方法进行查询

IPage<Goods> page = goodsService.findPage(new Page<>(pageNum, pageSize), name);

return Result.success(page);

}

**实现流程：**

1. 前端发起请求后AJAX将携带届数数据传入到后端中，被前端控制器拦截后分配给后端控制层，后端主要进行三步处理，数据查询，数据建模，数据可视化。
2. 根据用户查询数据，查询分类信息、商品信息、订单信息、分类信息后将数据存放到对应实体类中，完成数据查询。
3. 数据查询通过控制层访问业务层到持久层进行数据交互，查询出来的信息存放到list列表中，进行数据建模，前端建立好图表所需模型，将存放好的数据进行模型转换，对建立好的图表信息数据进行一一填充，填充后完成数据建模工作。
4. 初始化将要存放E-Charts图表的DOM对象，编写E-Charts图表需要的属性，最后将属性注入图表，然后返回到前端页面，就可以将数据可视化显示出来，并且可以根据CSS样式来自行进行调整，调整图表类型和样式。

### 5.2.2 商品管理模块

进入商品管理页面后会对用户信息进行全局查询，具体的商品管理页面如下图5-12所示。

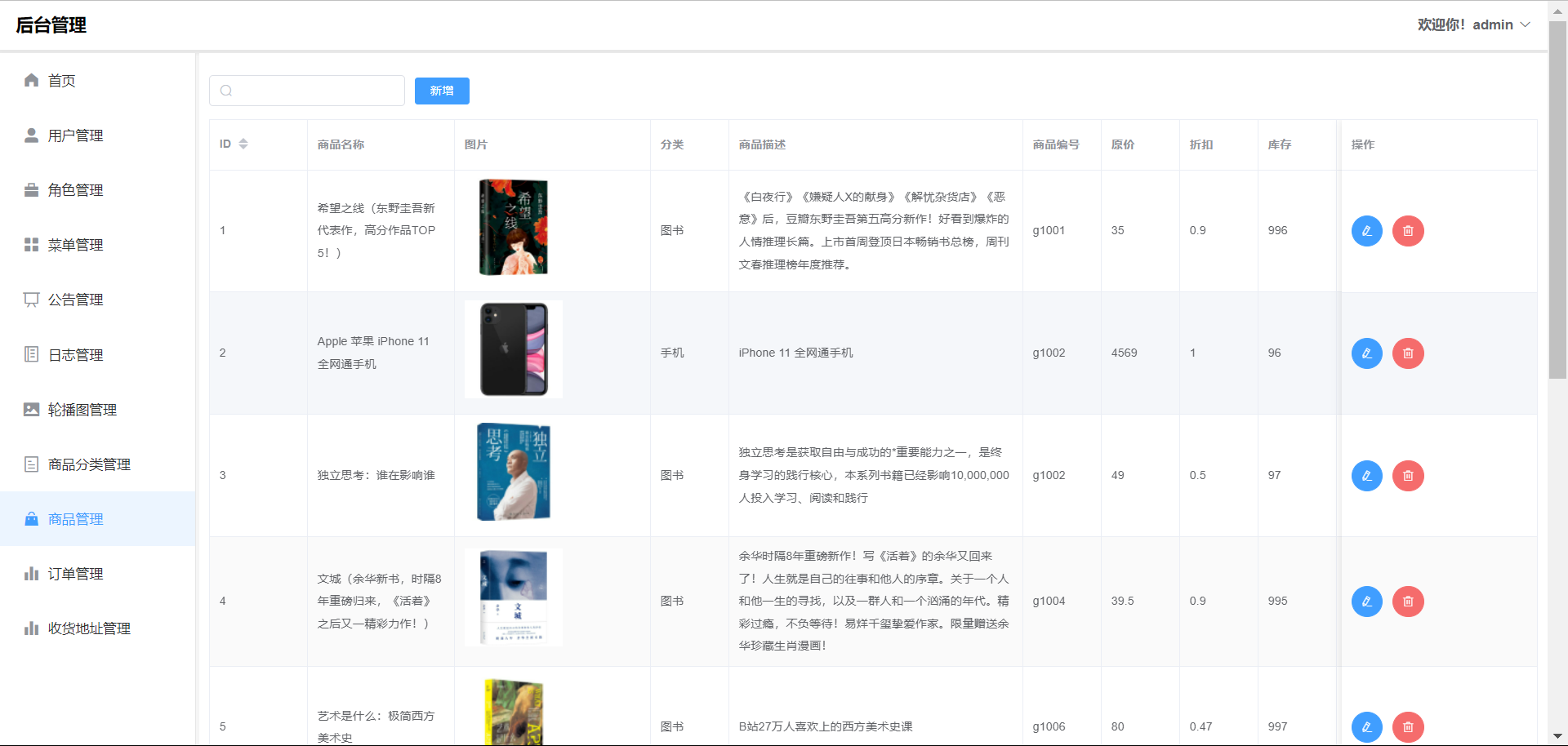


图5-12 商品管理页面

在商品管理中，可以对商品信息进行一系列操作，主要是对商品信息进行维护，除却基本的增删改查外，商品管理中还可以对用户图片来进行修改，页面如下图所示。

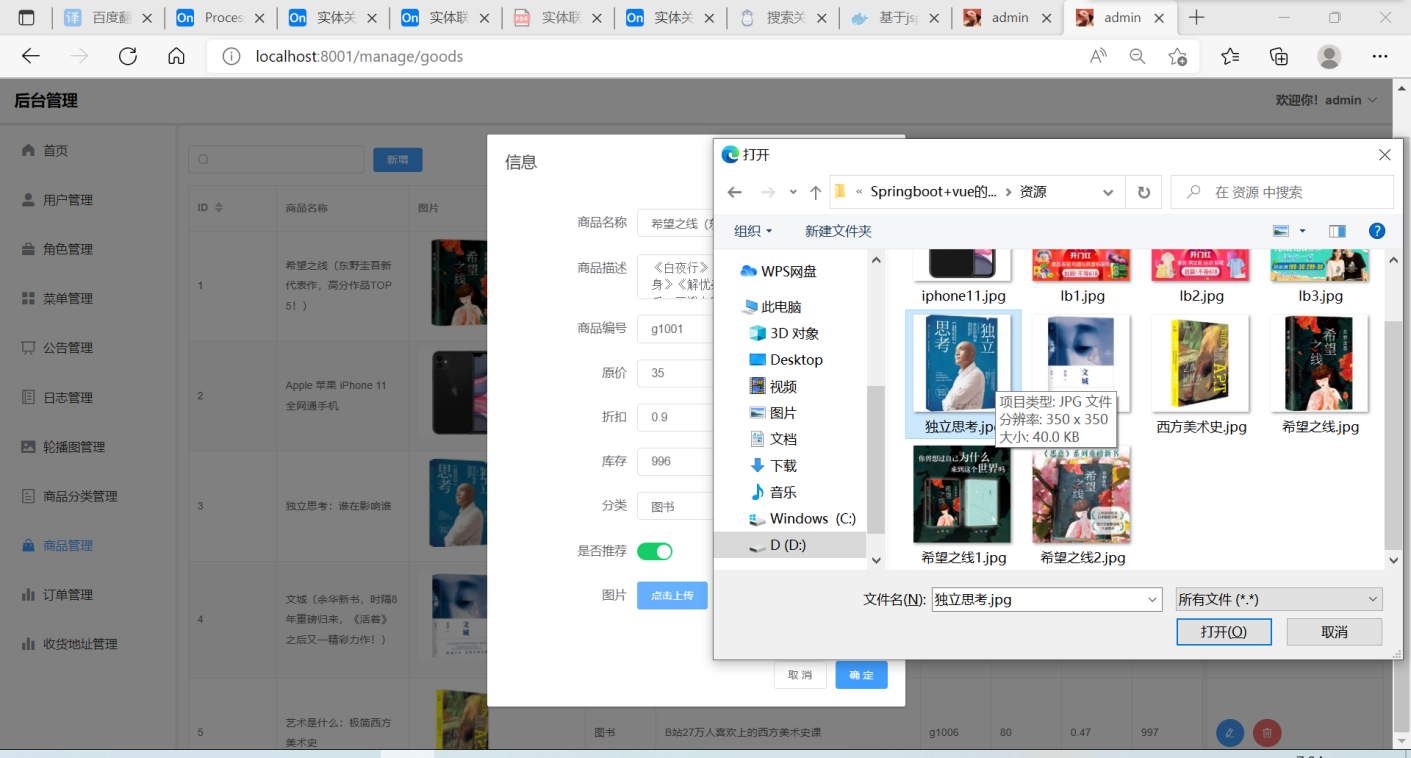


图5-13 图片上传页面

图片上传通过MultipartFile类将图片上传到后端中，然后后端将图片文件上传到对应的路径中，然后将图片路径保存到数据库中，每次读取的时候只用根据路径来读取图片信息即可，不需要再进行下载。

**主要代码**：

@PostMapping("/upload")

public Result<String> upload(MultipartFile file) {

String filePath = System.getProperty("user.dir") + "/src/main/resources/static/file/";

String flag = System.currentTimeMillis() + "";

String fileName = file.getOriginalFilename();

try {

FileUtil.writeBytes(file.getBytes(), filePath + flag + "-" + fileName);

System.out.println(fileName + "--上传成功");

Thread.sleep(1L);

} catch (Exception e) {

System.err.println(fileName + "--文件上传失败");

}

return Result.success(flag);

}

**实现流程：**

1. 头像上传支持多种格式，包括.png格式以及.jpg和.jpeg等格式，基本上涵盖了图片的常用格式，系统首先会将上传上来的图片信息进行出来，判断其后缀格式。  
    2、如果是.png格式则拼接路径为data:image/png;base64，如果是.jpg和.jpeg格式则拼接路径为data:image/jpeg;base64。  
    3、创建InputStream输入流和OutputStream输出流来对图片信息进行读写操作，将图片信息上传到指定位置，上传通过BASE64形式进行传输。  
    4、上传后将URL信息返回出来，存放到用户信息中，下次直接通过读取路径来查找和显示图片信息。

### 5.2.3 用户管理功能

进入用户管理页面后会对用户信息进行全局查询，并且将查询出来的信息进行分页处理，每页有十个信息，避免一个页面信息冗余，在页面里会对用户的主要信息进行展示，如果要查询用户的全部详细信息，需要点击进入到对应的用户信息里面，该模块主要功能是对用户尽行管理，主要包括用户的增删改查。

进行全局查询时，控制层获取到页面请求后会将信息下发到业务层中进行逻辑处理，用户列表功能是进行分页的全局查询，所以除了分页的rows和total字段外不会再传入其他信息。然后会将信息分页展示出来。如下图所示。

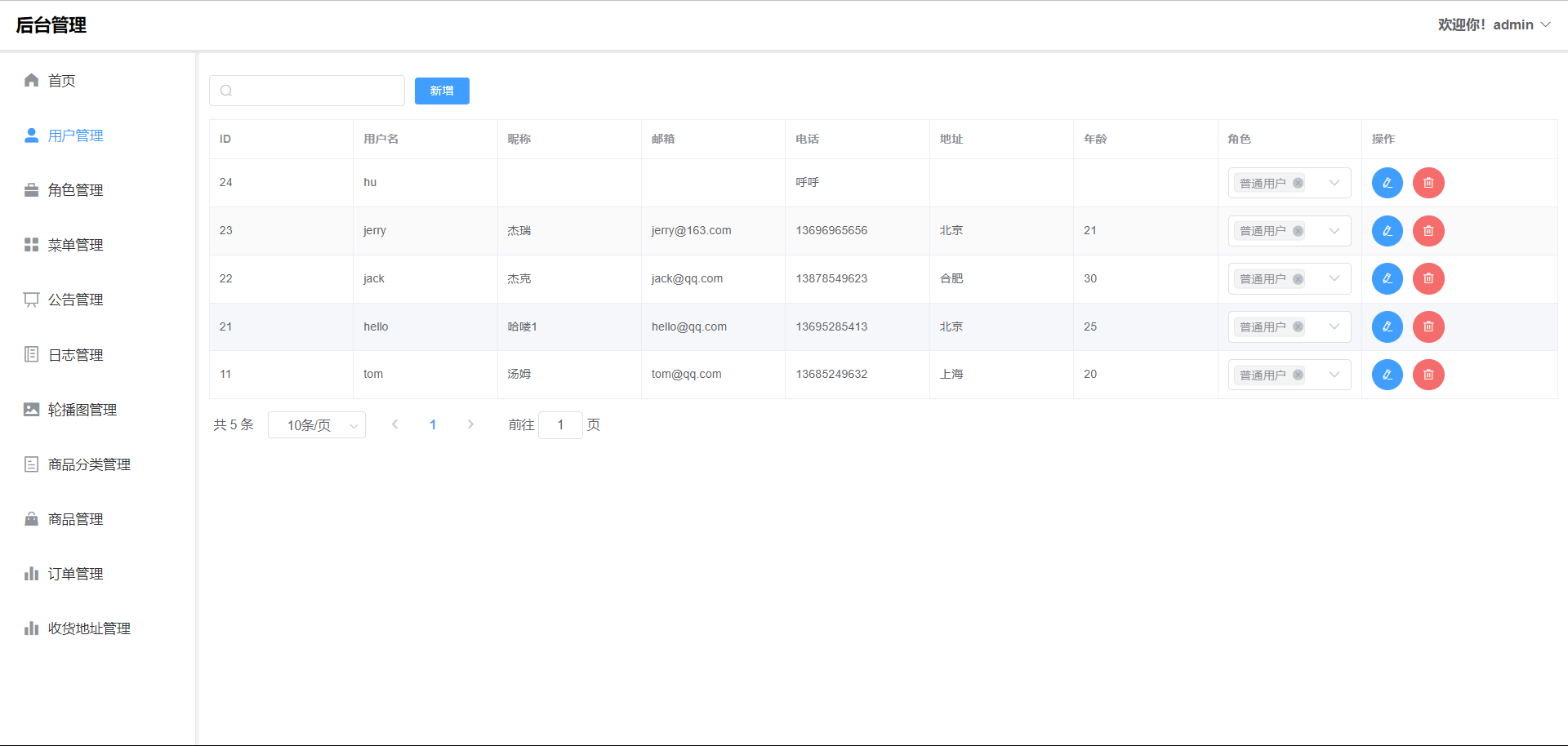


图5-14 用户列表

代码如下：

@GetMapping("/page")

public Result<IPage<User>> findPage(@RequestParam(required = false, defaultValue = "") String name,

@RequestParam(required = false, defaultValue = "1") Integer pageNum,

@RequestParam(required = false, defaultValue = "10") Integer pageSize) {

LambdaQueryWrapper<User> wrapper = Wrappers.<User>lambdaQuery().ne

(User::getUsername, "admin").like(User::getUsername, name).orderByDesc(User::getId);

return Result.success(userService.page(new Page<>(pageNum, pageSize), wrapper));

}

进入用户管理页面后点击新增用户就进入用户新增页面，页面如下图所示。

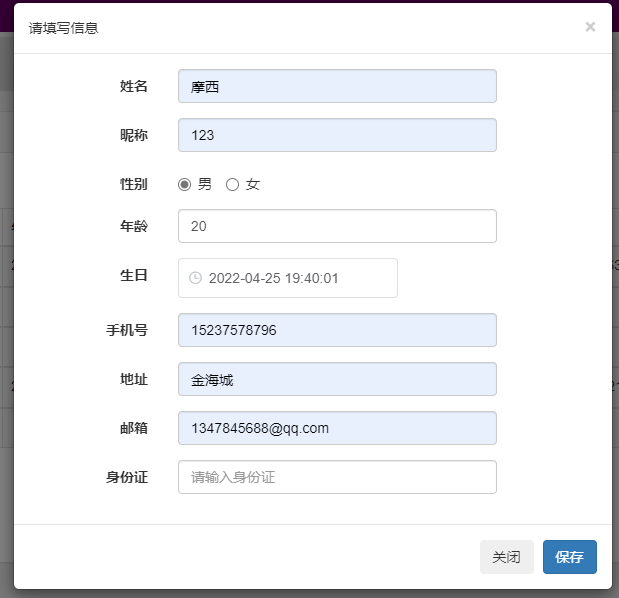


图5-11 用户新增页面

代码如下：

@PostMapping

public Result<?> save(@RequestBody User user) {

if (user.getPassword() == null) {

user.setPassword("123456");

}

logService.log(StrUtil.format("新增用户：{} ", user.getUsername()));

return Result.success(userService.save(user));

}

实现流程如下：

1、用户输入信息后前端页面会将信息通过Form表单的形式进行保存提交，然后JS会调用function($)方法来通过URL对后端的接口进行访问。

2、前端通过URL调用后端接口请求首先会被UserController的save()方法拦截下来，然后对信息进行解析。

3、UserService是用户管理的服务层，UserService.save()主要是对数据库中是否存在该用户做检验，返回类型为Boolean类型，返回True表示用户名已存在，返回False表示用户名不存在，可以进行注册。UserService进行逻辑处理后会将数据调用Mapper层来和数据库进行交互。  
 4、交互后的数据在逐层进行返回，新增成功后会在页面上返回成功的提示，新增失败后同样会进行错误提示。

### 5.2.4 权限管理功能

权限管理包括角色管理和菜单管理，对系统功能菜单进行管理，在该模块中可以对系统功能菜单信息进行维护，可以开启和关闭功能权限。对系统角色进行划分，可以在该模块中新建和删除角色信息，并且根据角色在菜单管理中分配权限。页面如下图所示。



图5-12 角色管理页面

在角色管理中可以对角色信息进行维护，其中最主要的是对角色功能进行分配，页面如下图所示。



图5-13 分配权限页面

代码如下：

@PutMapping

public Result<?> update(@RequestBody Role role) {

logService.log(StrUtil.format("更新角色：{}", role.getName()));

return Result.success(roleService.updateById(role));

}

可分配的权限则来自菜单管理中，页面如下图所示。

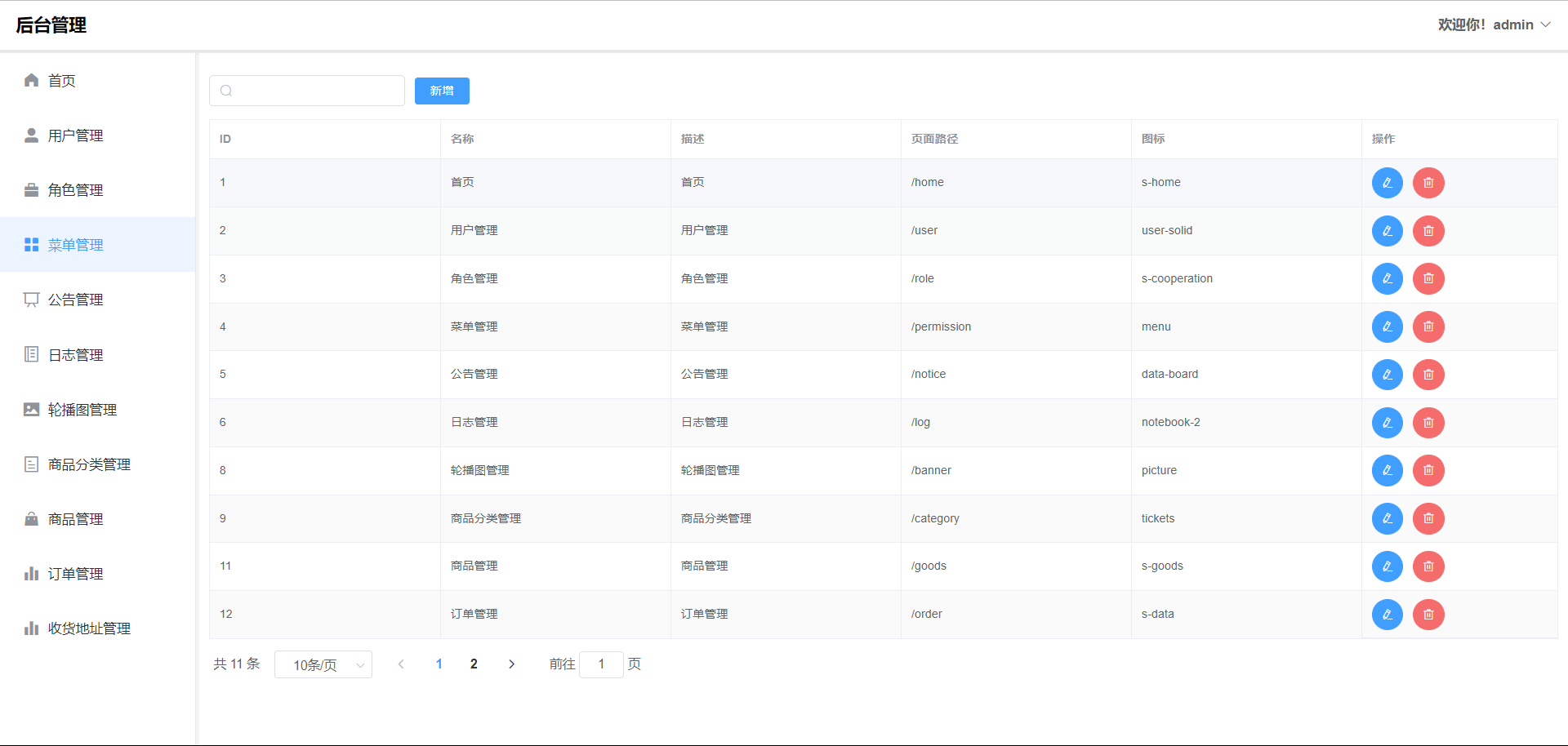


图5-14 菜单管理页面

在菜单管理中可以对菜单列表信息进行维护，包括新增、修改和删除功能，对菜单信息进行修改后也会对角色和用户信息进行更改。

修改页面如下图所示。

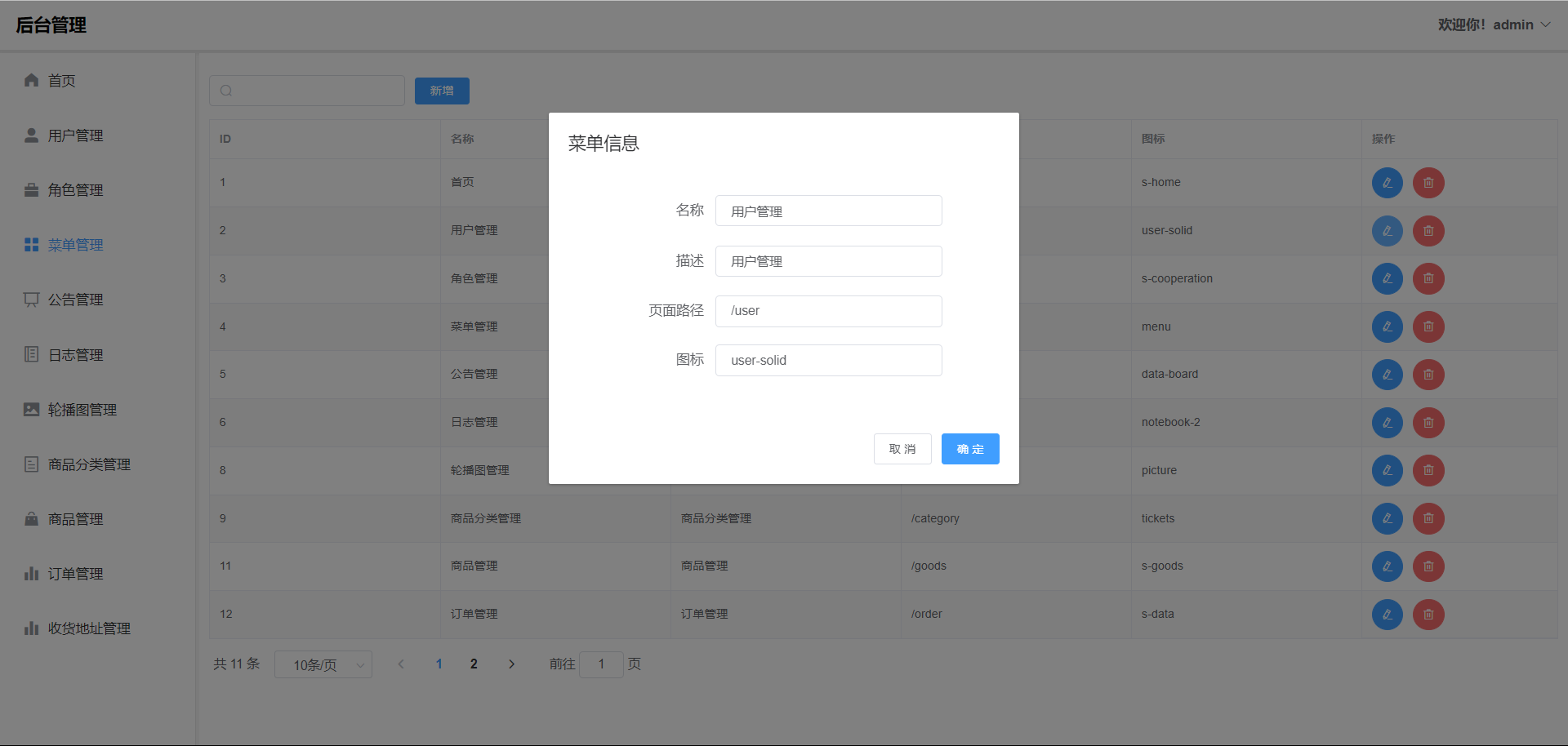


图5-15 菜单修改页面

代码如下：

@PutMapping

public Result<?> update(@RequestBody Permission permission) {

logService.log(StrUtil.format("更新权限菜单：{}", permission.getName()));

return Result.success(permissionService.updateById(permission));

}

在菜单管理中管理功能列表，在角色管理中进行功能分配，这样就实现了权限控制功能的实现流程。

## 5.3 本章小结

本章节分别从系统前台功能模块的设计与实现到系统后台功能模块的设计与实现进行了介绍，介绍本系统的功能实现。

# 6运行系统和测试

## 6.1测试规划

系统测试是指对整个系统进行全方面的测试，由于本系统不牵涉硬件设施，所以不需要进行硬件测试，本系统全面测试主要是对系统进行功能测试和性能测试以及安全测试等方面进行，功能测试方面需要先对系统开发功能说明书来进行分析，然后测试功能上是否有不符合功能说明书的要求，如果有需要记录下来并且督促开发人员进行修改。性能测试则是对系统运行速度以及进行压力测试，通过大量数据来模拟系统的高压下的运行情况，一般分别从系统响应速度和并发量来测试系统的临界值和平均耗时。对于安全测试来说，主要是对系统安全性来进行测试，对系统数据的安全性，关键信息的防护能力，以及进行sql注入来查看系统是否能防范，主要是对系统安全措施和抵御非法入侵的能力。

本系统的测试规划：

1. 分析并记录系统的功能说明书，并且将系统模块进行整理分类；
2. 根据系统功能模块来编写单元测试用例，并且根据测试用例来完成黑盒测试；
3. 将测试结果记录并反馈给相关开发人员进行修改，修改后针对对应的模块进行反复复测。
4. 对系统进行压力测试，测试最大并发量和系统不同并发量下的反应速度，并且记录临界值。
5. 对系统进行安全性测试，通过模拟非法入侵来测试系统的安全性。
6. 汇总编写测试报告。

## 6.2测试用例

表6-2测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试案例 | 操作 | 预期结果 | 实际结果 | 结论 |
| 用户注册测试 | 1.用户进入注册界面  2.用户输入注册的基本信息用户名：user、登陆密码：123123手机号：1526565712、确认密码：123123  3.点击注册按钮 | 注册成功 | 注册成功 | 测试达标 |
| 用户登陆功能测试 | 1.点击登陆页面；  2.输入账户：user，密码：1234563.点击登陆按钮。 | 登录失败，提示”账号或密码错误” | 登录失败，提示”账号或密码错误” | 测试达标 |
| 用户登陆功能测试 | 1.点击登陆页面；  2.输入账户：user，密码：123123 .点击登陆按钮。 | 登录成功 | 登录成功 | 测试达标 |
| 订单删除功能测试 | 1.管理员输入账号/密码：admin/admin 点击登陆；  2.登陆成功点击订单管理；  3.选中订单；  4.点击删除订单；  5.点击确认删除； | 删除成功 | 删除成功 | 测试达标 |
| 商品查询功能测试 | 1.管理员输入账号/密码：admin/admin 点击登陆；  2.登陆成功点击商品管理；  3.输入框中输入：Apple 苹果 iPhone 11  4.按回车搜索 | 查询成功 | 查询成功 | 测试达标 |
| 角色新增功能测试 | 1.管理员输入账号/密码：admin/admin 点击登陆；  2.登陆成功点击角色管理；  3.点击新增角色；  4.输入角色名称：商家，描述：无  5.点击新增按钮 | 新增成功 | 新增成功 | 测试达标 |

## 6.3测试结果

通过对本系统进行黑盒测试，系统达到可以运行、可以正常的完成系统中的功能，项目确定的范围、策略以及对应计划已经基本实现。

## 6.4 本章小结

本章主要介绍了软件测试的定义，对系统的功能经行了黑盒测试并写明了测试的用例。通过测试，本系统功能基本实现，运行较为成功，基本符合预期。

# **总 结**

本文设计并开发了一款基于SpringBoot框架的在线购物平台，前端采用Vue框架实现，前后端分离通过HTTP的方式进行调用，数据存放在MySQL数据库中，系统实现前后台两个界面，前台功能包括商品查询、商品购买、购物车、订单管理、收藏夹、收货地址管理等功能，后台由管理员使用对系统信息进行管理和维护，系统集成Interceptor拦截器来进行Token验证，集成E-Charts框架来实现数据统计功能。

开发一款线上购物商品平台可以解决人们出现购物问题，省去线下购物的时间，给人们带来极大的便利。同时对于管理人员来说，完善的后台管理系统可以最大程度上省去人力物力的投入，传统需要三个人进行统计记录的工作现在一个人就能轻松完成，计算机管理系统的介入让管理更为高效快捷。

本系统对数据查询进行了优化，增加了索引，提高了查询速度，增加了用户的体验。虽然本系统对已有的系统进行了改进，但是由于自身技术原因问题，本系统还存在着不少的问题。在系统中没有将模块单独罗列出来，对进度进行统一的管理，没有设计音乐排行榜功能来进行实现，这是不小的遗憾。

本系统的创新点：

（1）本系统采用Spring Boot框架+Vue框架开发，前后端分离，前后端代码分离开来，通过Http进行调用，实现前后端解耦，让代码更为清晰。

（2）本系统增加数据统计与分析模块，把分类商品销量进行统计与分析，系统集成E-Charts框架，将统计的数据进行可视化展示。

（3）本系统优化图片上传和预览功能，由于系统需要大量的图片上传和显示的地方，所有将图片上传进行优化，配置MultipartConfigFactory来实现图片上传。

总得来说，本系统还是有需要迭代和拓展的地方，伴随着自身技术的不断提高，可以将线上购物商城系统功能集成微服务框架，将功能点上传到阿里云上，通过微服务框架来实现模块之间功能的完全解耦，需要用到那些功能只用调用该功能模块本身即可。

最后，希望通过开发本系统可以给人们提供一款更好用的Web端商城购物平台，在生活中在可以随时随地的进行线上购物。其次还希望有关管理方面的系统软件可以得到推广，增加对数量信息的统计、集成推荐算法等，采取更方便的技术对系统进行完善，提高用户的使用体验。

# 参考文献

[1]陈继磊. 高并发下购物平台系统的设计与实现[D].山东大学,2021.

[2]杨文豪. 基于微服务的网上商城系统的设计与实现[D].北京邮电大学,2021.

[3]杨众. 基于Spring Boot的校园商城系统设计与实现[J]. 信息技术与信息化,2020,(09):81-82.

[4]王柯梦,王瑞平. 基于B2C网上商城前台系统数据库设计与实现[J]. 广西质量监督导报,2020,(09):70-71.

[5]薄志强. 基于SSM框架的网上商城系统的设计与实现[D].西安电子科技大学,2020.

[6]耿庆阳. 基于Spring Boot与Vue的电子商城设计与实现[D].西安石油大学,2020.

[7]刘彤月. 基于SSM的分布式商城的设计与实现[D].中国地质大学(北京),2020.

[8]Nadeem Fakhar Malik,Aamer Nadeem,Muddassar Azam Sindhu. Achieving State Space Reduction in Generated Ajax Web Application State Machine[J]. Intelligent Automation & Soft Computing,2022,33(1).

[9]唐小玲.基于AJAX异步交互技术的门户网站构建[J].电脑知识与技术,2021,17(32):165-166.DOI:10.14004/j.cnki.ckt.2021.3256.

[10] Selakovic M, Pradel M. Performance issues and optimizationsin JavaScript: an empirical study[C]// Ieee/acm, InternationalConference on Software Engineering. IEEE, 2017: 61-72.

[11] 国家发展改革委.“十四五”循环经济发展规划[Z].发改环资〔2021〕969 号，2021（7）

[12] 邱琼. 十四五我国电子商务发展趋势研判与政策思路[J]. 商业经济研究，2021（3）

[13]吕来明,陈天舒.二手物品网络交易平台的运行机制及义务承担规则[J].商业经济研究,2021(24):111-114.

[14]张洋洋,王丹,越豪杰,范政,林浩. 校园二手物品交易平台研究与设计[J]. 无线互联科技,2021,18(10):48-49.

# 致 谢

行文至此，我的文章趋于结尾，当然也代表着我的官方学习生涯告一段落，但学海无涯，我将永远怀着一颗学徒的心，以谦虚、兼爱、仁慈的心去迎接这个属于我的另一段生活。当我满含热泪回首过往，四年来尽是无数的希望和感动，我收获了令我足以生存无忧的知识、教会我更好的与人相处的友谊、令我毕生难忘的师生情节。

首先要感谢我敬爱的指导老师，他作为我的本科论文指导老师，他有着近乎完美的教学方法以及认真负责的态度，从我最初与他相识到论文的题目定题、开题报告的撰写、初稿的审查最后到论文的定稿等等，兢兢业业、任劳任怨、恪尽职守是我对他最真实的评价，他用他那丰富的经验与技巧帮助我修改论文不合理的地方以及升华论文，在此由衷地感谢于延老师在百忙之中抽出宝贵的时间给我的论文提出宝贵的指导意见。祝我最敬爱的老师身体健康，工作顺利！

当然还有大学四年以来我最亲密的同窗们，当我孤身一人背井离乡前往心中的象牙塔，对我来说，这是无疑是一个充满孤独的地方，安全感的缺乏让我不禁背生凉意，是他们亲切的问候让我心升暖阳。四年以来，我们相互帮助，共同分享生活的喜悦，在考研期间我一次次想放弃也是他们一次次耐心的安慰，我才能扛到现在。最后，他们宝贵的建议也是我能顺利完成论文必不可少的因素。天下没有不散的宴席，没有人的青春永远十八岁但总有人满怀十八岁的梦想，愿他们在今后的人生旅途中长风破浪，直挂云帆！