兔子大学北京校区

课程设计报告

基于SpringBoot + Vue的电子商城的设计与实现

学　　院：　计算机学院

专　业　名　称：　软件工程

学　生　姓　名：　忧伤大白兔

学　　号：　00000001

完成日期：　xxxx 年 xx 月 xx 日

目录

一、需求分析

1. 引言

1.1 编写目的

该系统需求分析文档的编写目的为明确该电子商城系统的业务需求、功能需求等方面的要求和规划，包括系统的定义、范围、特征、过程、用户需求、控制需求、界面需求、性能需求等。本文档旨在提供详细和清晰的系统需求，以指导系统的设计和实现，有效保障系统的可靠性、安全性、易用性和高效性，最终满足用户的期望和需求，并使整个系统开发的过程更加明确、高效和自然。该文档的目的是为了确保开发出高品质的系统，顺利地完成了各种业务需求和用户需求的实现。

1.2 项目说明

本项目是针对电子商城系统的需求分析文档，包括业务需求、功能需求和非功能需求等方面的内容。该文档是本项目的重要成果之一，旨在全面地描述该电子商城系统的需求，明确系统的定义、范围、特性、过程、用户需求、控制需求、界面需求、性能需求等方面的要求和规划，为系统的设计和实现提供指导和依据。在编写该文档时，需按照标准的格式和规范进行撰写，注重需求的详细描述和正确性，以确保系统的功能和性能的正确实现，同时还需要充分考虑用户的需求和反馈，确保满足用户的需求和期望。该文档同时还需要与其他项目文档进行协同，配合系统开发的进度和任务，使开发过程更加清晰和有条理。

2. 产品介绍

2.1 产品概要说明

该电子商城系统旨在为商家和消费者提供一个直观、易用的购物平台，通过该平台销售商品和宣传品牌，消费者通过该平台购买商品，享受更便捷的购物体验。在该电子商城系统中，商家可以上传和管理商品信息、进行销售统计、订单管理和客户管理等功能，消费者则可以通过购物车等功能方便地购买商品。本系统的目标是提供一个用户体验良好、商家易于管理的电商平台，以实现系统商业化运作和持续增长。

在编写需求分析文档时，本系统的业务目标和产品方向明确，需求从用户的角度和企业利益的角度出发，全面深入地分析了该系统对商家和消费者的需求，包括产品定义、范围、特性、过程、用户需求、控制需求、界面需求、性能需求等方面的内容。在需求分析文档中，详细描述了系统架构、界面设计、交互逻辑、数据流、数据库设计、安全性和可靠性等主要内容，便于开发者理解和实现系统需求。本系统将充分考虑人性化和易用性，为用户提供提高使用体验的功能，如搜索、推荐功能等。同时，为保障系统的安全性和稳定性，系统还设计了具有可扩展性的开放接口和良好的系统性能，并依照标准文件格式和规范进行编写，最终确保系统的质量和用户体验的可持续性。

2.2 产品用户定位

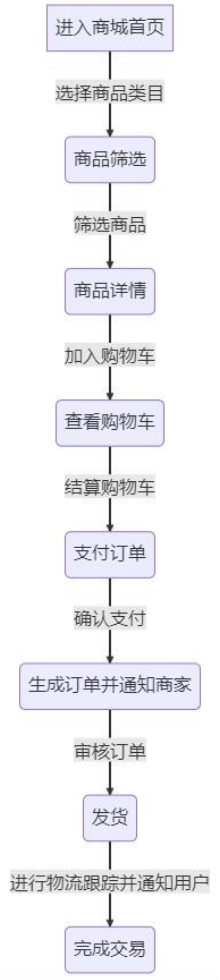
本系统的产品用户包括商家和消费者。商家是系统的提供者，他们通过该平台发布商品信息并销售商品，通过该平台增加销售渠道和宣传曝光度来扩大业务影响和盈利水平。消费者则是系统的使用者，通过该平台购买商品、付款、跟踪订单等进行网上购物。

因此，该系统需要满足商家和消费者的不同需求，为其提供便利、快捷、安全的使用体验，增强商家和消费者之间的沟通和信任，达到互利互惠的共赢局面。在系统需求分析文档中，就针对商家和消费者的需求分别列出了相应的需求和特性，明确了各个用户在该电子商城系统中所需的具体功能和实现要求，以提高产品的实用性和用户满意度。

2.3 产品中的角色

商城管理员、消费者

3. 产品总体业务流程图



该流程图是该电子商城系统的主要业务流程图，展示了消费者浏览和选择商品、加入购物车、结算支付和商家发货等流程。具体流程及相关功能如下所示:

消费者进入商城首页，选择商品类目;

根据需求，在商品页面进行筛选和详情查看;

选择商品后，加入购物车，随时查看购物车中商品;

在购物车中结算购物车，输入订单信息并确认支付;

订单信息提交成功后，通知平台管理员审核订单;

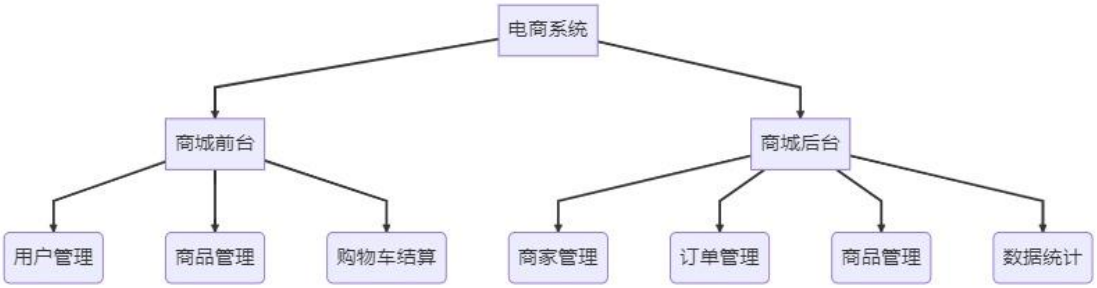
管理员审核订单通过后，通知商家发货;

商家根据订单发货，并通过系统进行物流跟踪;

用户收到货物并确认交易完成。

该流程图可快速展示电子商城系统的主要业务流程和逻辑，便于用户理解和对系统进行需求分析。

4. 产品功能结构图



说明:

商城前台包括用户管理、商品管理、购物车结算等，供消费者使用;

商城后台包括商家管理、订单管理、商品管理、数据统计等，供平台管理员和商家使用;

电商系统是这两个部分的总体，实现电商交易的整个过程。

通过该产品功能结构图，可以清晰地了解每个模块的基本功能，并知道它们之间的关系，使得系统开发者和用户能够更好地理解该电商平台的主要结构和功能。

5.任务管理

5.1 项目现状

该电子商城系统是一项新的电商平台建设项目，目前处于规划和设计阶段，还没有开始投入具体的开发工作。本项目由开发小组负责开发和实现，开发周期为四个月。在该项目中，将采用目前流行的 Java Web、Vue 等技术，结合底层的 Web 技术和框架，以实现该电子商城的核心功能和创新点。根据传统的需求工程流程，该项目将会完成需求分析、系统设计、开发实现、测试、上线的五个阶段，本文档即是项目启动的第一个阶段：需求分析。

5.2 项目目标

该电子商城系统的目标是创建一个功能丰富、用户友好和易于管理的电商平台，以方便商家上传和管理商品信息，为消费者提供简单快捷的购物体验。在此基础上，通过使用现代化技术、强化平台数据支持、持续改进用户体验，最终实现电商交易的成功转化，帮助商家提升销售业绩和品牌知名度，进而实现持续盈利。项目的目标是为全球用户提供更好的电商服务，为合作伙伴带来更具竞争力的各项优势，为整个行业带来更多发展的机遇。

6.功能需求

6.1 功能概述

商品管理、商品分类、订单管理、用户管理、购物车管理等基本功能。其中，商品管理包括商品添加、编辑、删除等操作；商品分类包括商品分类添加、编辑、删除，以及商品分类的显示和维护；订单管理包括订单的添加操作；用户管理包括用户注册、登录、个人信息管理，以及管理员对用户的管理；购物车管理包括购物车的添加、修改、删除等操作。通过这些功能，系统可以实现一个完整的电商平台。

6.2 功能点清单

功能模块	功能点
商品管理	商品添加
	商品编辑
	商品删除
商品分类	商品分类添加
	商品分类编辑
	商品分类删除
	商品分类显示和维护
订单管理	订单添加
用户管理	用户注册

功能模块	功能点
	用户登录
	个人信息管理
	管理员对用户的管理
购物车管理	购物车添加
	购物车修改
	购物车删除

6.3 功能点描述

6.3.1 商品管理

此模块负责提供对商品资源的管理。具体包括添加商品（如商品基本信息，商品展示图片，库存量等），对商品进行编辑（如修改销售价格，更新库存等）以及删除商品。对于电商平台而言，商品是核心资源，因此商品管理模块的健康运作对于整个平台都具有重要意义。在实现时，该模块需要根据实际业务需求进行具体设计和实现。

6.3.2 商品分类管理

此模块负责对商品进行分类管理。具体包括商品分类添加、编辑、删除，以及商品分类的显示和维护。商品分类为用户在浏览或搜索商品时的重要依据，因此该模块的设计和实现需要充分考虑到用户的使用体验及便捷性。

6.3.3 订单管理

此模块负责提供订单的生成与管理。具体包括订单添加、修改、删除，以及订单状态更改等。订单的管理在电商平台中是至关重要的，而订单状态更改等功能则为用户提供安全可靠的购物环境，是电商平台优化用户体验的重要手段。

6.3.4 用户管理

此模块负责用户的注册、登录、个人信息以及管理员对用户的管理。通过用户管理模块，用户可以注册成为平台会员，登录到自己的账户进行购物、查看订单、管理个人信息等操作，同时管理员可以通过该模块对用户账户进行管理，如进行用户的黑/白名单管理，账户密码管理等操作。对于电商平台而言，用户管理模块的健康运作对于保障平台安全和用户购物体验至关重要。

6.3.5 购物车管理

此模块负责提供购物车功能，包括购物车的添加、商品数量修改和删除等操作。购物车是电商平台中一个常用的功能模块，通过对购物车的管理，用户可以方便地添加多个商品并记录在购物车中，在需要时进行快速结算。同时，该模块的健康运作还能为平台提供收入增长的机会。

二、可行性分析

1. 引言

1.1 标识

ECA-20230501-001

1.2 背景

随着互联网技术的发展和普及，电子商务在全球范围内得到了迅猛的发展，已经成为了一种重要的商业模式和生活方式。电子商城是电子商务的重要组成部分，是一个基于互联网的商业模式和交易平台，通过网络进行产品和服务的销售。电子商城已成为了线上交易和消费的主要方式，人们可以方便快捷地买到自己想要的商品和服务，并在家中享受线上购物的便利性和舒适性。

基于这个背景，我们决定开发一个功能强大、易于使用、可靠性高的商城系统，以满足用户在线购物的需求。这个商城系统将支持多种商品和服务的销售、多种支付方式和多种物流配送方式，提供优惠券和积分等扩展功能，为用户提供舒适的购物体验和服务。同时，我们还将采用最新的技术和开发模式，在保证系统稳定运行的同时，不断推出新的功能和特性，以满足用户不断增长的需求和期望。

1.3 项目概述

该系统一个基于 Web 的电子商城系统，可以实现在线购物、订单管理、用户管理等功能。用户可以在平台上浏览商品、选择规格、加入购物车，然后通过结账付款生成订单，并在个人中心进行订单的查看操作。管理员可以对商品分类、商品信息、订单信息进行管理和维护。

该电子商城系统的目标是建立一种快速和方便的在线购物平台，提供个性化、智能化、安全可靠的服务体验，为消费者提供丰富的商品选择，以及一个公平、透明和可靠的交易环境。该系统的收入主要来自于商品销售和提供广告服务，是一种充满商机和发展前景的商业模式。

该电子商城系统的开发涉及了多个技术、团队协作、需求分析、测试等多个方面的工作，是一项大型的软件开发项目。互联网公司、商业公司和投资人都可以是该项目的开发者和投资人，共同推进该电子商城系统的建设，创造更多的商业机遇和社会价值。

1.4 文档概述

本文档将对基于 SpringBoot+Vue 的电商系统进行可行性分析。在分析的过程中，将探讨该系统的市场需求、技术可行性、商业模式、资金投资、利润预测等方面的问题。通过本文档的分析，可为有意开发该类型电商系统的团队提供一定的参考和决策支持，并为投资者对项目的可行性做出评估。该电商系统具备商品管理、分类管理、订单管理、用户管理、购物车管理等基础功能，为用户提供购买商品的便利。在未来的发展中，该系统可以加入优惠券红包等进一步提高用户使用体验，增加用户黏性。

2. 可行性分析的前提

2.1 项目的要求

项目目标及定位：本项目旨在提供一个全功能的电子商城系统，面向所有需要在线购物的用户，包括但不限于日常用品、家居装饰、数码电器等，为商家和消费者提供一个直观、易用的购物平台。

功能需求：系统需要拥有完善的商品管理功能，包括商品的新增、编辑、删除等；商品分类管理功能，方便用户快速定位需要的商品；订单管理，包括订单状态、物流信息等；用户管理，包括用户的注册、登录、个人信息管理等，同时需要支持管理员对普通用户的管理；购物车管理，支持用户加入购物车并统计订单金额。系统需要支持至少 10 万用户同时在线，能够承受高并发访问和数据访问的负载。

技术要求：系统基于 SpringBoot+Vue 进行开发，数据库使用 MySQL，服务器端能够支持 Linux 环境部署、Tomcat 服务器。同时需要支持 RESTful API 接口规范，确保系统的高可维护性和扩展性。

人员需求：系统开发团队需包括至少 3 名开发人员，其中包括至少 1 名 Java 开发人员和 1 名前端开发人员。所有团队成员需要具备一定的项目开发经验和技术水平，能够高效协作。

资金需求：项目预算需要包括开发、测试、部署、运维、市场推广、客服等方面的成本，预算总额为 50 万人民币。其中开发成本最高，预计需要占该总预算的 60%。在项目运营期间，需要额外投入一定资金进行市场推广和用户获取，以确保项目的正常运行。

商业模式：系统主要收入来源为商家的广告投放费用和商品销售服务费用，商家投放广告可获得免费推荐商品、排名优先、关键词置顶等特权，而平台则从中抽取一定比例的广告费和交易服务费。

营销策略：在项目运营期间，需要通过多种方式获取用户，包括线上和线下的各种广告投放和推广活动，加强用户体验和口碑传播，例如免费配送、用户礼赠等方式。并根据数据分析和用户反馈进行精准营销策略的制定。

2.2 项目的目标

提供一个全功能的电子商城系统，以解决用户在线购物的需求，为商家和消费者提

供一个直观、易用的购物平台。

实现商家与消费者的信息对称化，为商家提供一个销售商品和宣传品牌的渠道，为消费者提供一个安全、便捷的购物环境。

通过优化用户体验、提高交易效率、完善售后服务等方式，增加用户对该系统的粘性和信赖度，从而实现平台的盈利目标。

为商家提供一个低成本、高效率的销售渠道，从而提高商家的盈利能力，并通过平台的流量导入和用户转化为商家带来更多的商业机会。

不断优化用户体验、拓宽服务范围和合作伙伴，为消费者带来更多选择，从而增加系统的市场份额和竞争力。

这些目标是该电子商城系统开发的出发点，也是开发者在项目中需要始终关注和追求的目标。在可行性分析中，我们需要从多方面考虑这些目标的实现可行性，确保项目能够符合实际需求并创造积极的商业价值。

2.3 项目的环境、条件、假定和限制

1、环境：该系统将在互联网环境下运行，需要考虑数据安全、网络稳定性、访问速度等问题。

2、条件：开发人员需具备相关技术能力，需要有足够的开发时间和开发成本支持，项目需要有足够的用户和商家参与。

3、假定：该项目的假设条件包括，用户需求为在线购物，市场潜力和竞争环境适合开发该项目，商家愿意选择该平台销售商品，用户对该平台有较高的信赖度。

4、限制：在项目开发和运营期间，需要遵守相关法律法规，包括但不限于广告法、电商法等，需要制定和执行严格的隐私保护和数据安全政策。

这些环境、条件、假定和限制是该系统可行性分析的前提，需要在整个项目计划和执行过程中加以考虑和遵守，以确保项目的顺利实施和长期稳定。

2.4 进行可行性分析的方法

1、技术可行性分析：从技术角度考虑该系统是否可行，包括开发语言、框架、数据库、服务器等方面，以及技术人员是否具备该系统开发所需的技术水平。

2、经济可行性分析：从经济角度考虑该系统是否可行，包括项目开发和运营成本、预期收益等方面，以及市场需求和竞争环境是否足以支持该系统盈利。

3、商业模式可行性分析：研究该系统的收入模式、用户获取和留存策略等商业模式是否可行，并估算市场的需求、用户和收益潜力。

4、社会环境可行性分析：考虑政策、法律、文化等方面对该系统的影响，以评估各种因素对系统可行性的潜在影响。

在本项目的可行性分析中，需要综合运用以上方法，进行综合评估，以确保项目在技术、经济、商业模式和社会环境等方面的可行性。同时，需注意对各个分析方法中得出的最终结论进行合理综合和协调，以获得最终的可行性分析结论。

3. 可选的方案

3.1 方案 A

采用增量式开发模型，根据系统的功能需求和用户反馈，逐步开发和完善系统。在开发过程中，需要进行充分的交流和协作，及时发现和解决问题。此外，周期性地对开发进展和开发质量进行审核和评估，确保开发进展和预期质量双重可行性。

3.2 方案 B

采用敏捷开发模型，将系统需求和开发联系起来。在短周期内进行开发、测试、部署和反馈，充分倾听用户反馈和需求，快速响应市场变化和用户变化，在快速满足市场需求和用户的同时，保证开发进度和开发质量。

3.3 方案 C

采用原型开发模型，制作初期的原型版本，进行用户测试和反馈，从而完善原型，并在此基础上完善系统。在原型测试中，满足用户的需求和交互体验至关重要，可以通过原型评测表对用户的需求和反馈进行收集和整理，以便在开发初期提供有利的参考依据。

3.3 选用的方案

采用方案 B：采用敏捷开发模型。原因如下：

1、敏捷开发模型不需要大量的人力资源，可以适应人数较少的团队。

2、敏捷开发模型强调快速响应市场变化和用户反馈，可以快速推出 MVP 版本，不需要大量的功能开发和前期规划，减少了开发时间和成本。

3、敏捷开发模型将开发、测试和部署等环节相集成，可以加速开发进度，减少团队协调与沟通的时间。

4、敏捷开发模型注重用户反馈，可以快速获取用户需求和意见，和根据反馈迭代开发，确保开发和市场的双重可行性，以此提高产品的质量和用户体验。

5、敏捷开发模型强调团队协作和沟通，鼓励开放式的沟通，可以将团队成员间的合作和沟通紧密结合，从而更快地达成共识，推进项目进展。

综上，因为敏捷开发模型对团队人数要求不高，能够充分协调人员之间的工作，可以快速地响应市场和用户反馈，更好地满足客户需求，推出高质量的产品和服务，所以在人员比较少的情况下，采用敏捷开发模型是更好的选择。

4. 所建议的系统

4.1 对所建议的系统的说明

该系统提供一个全功能的电子商城系统，以解决用户在线购物的需求，为商家和消费者提供一个直观、易用的购物平台。商家通过该平台销售商品和宣传品牌，消费者通过该平台购买商品享受更便捷的购物体验。在该电子商城系统中，用户可以购买各种商品，并获得相应的购物体验，包括购物车、收藏夹等功能，同时平台支持商家的商品上传和管理、销售统计、订单管理和客户管理等功能。本着用户至上和质量至上的理念，在开发该系统过程中，将充分考虑用户的需求和反馈，不断完善电子商城的功能和性能。同时，开发者还将积极探索技术创新和商业模式创新，在不断提升用户体验和商业价值的同时，持续增长和扩大该系统的市场份额和竞争力。由此，本系统是一个能够满足用户需求，具有强有力的市场竞争力的功能完整的电商系统。

4.3 影响

4.3.1 设备

开发人员所有的设备和网络设备。

4.3.2 软件

visual studio code、MySQL、Intellij IDEA。

4.3.3 运行

运行环境：Linux 系统。

4.3.4 开发

平台后端基于 Spring Boot 构建。平台前端基于 Vue.js 构建。

4.3.5 环境

开发环境：Windows10，MySQL 8.0.19，Intellij IDEA，visual studio code。

4.4 局限性

处理时间较长，随着对于未来日益增 长用户数量增加。无法满足用户数据收集，在数据资源更新和存储上必须扩容。

5. 经济可行性

5.1 投资

除了 idea 开发工具需要收费，其他开发工具均可以免费使用，均免费面向开发者，这是我们经济可行性最为关键的要素，但后期项目上线的云服务器是需要付费的，整体开发成本是比较低的，所涉及的技术栈也是开源免费的，几乎 0 成本，在开发工作结束后，是以出租的方式给第三方机构使用，一旦我们可以获得较高的客源的话，收入也是非常的可观。

5.2 预期的经济效益

用户增长：随着用户数量的增加和平均每人消费金额的提高，该系统将增加用户数量和用户消费额，从而增加年收入。

低成本高效：本系统将通过互联网等手段，实现低成本的销售渠道和营销手段，从而降低运营成本和获客成本，并提高营收效率。

品牌价值提升：通过平台的品牌优势和商家的品牌合作，相互提升品牌知名度，进而提高系统的市场竞争力和商业价值。

商家的营业额提高：我们的电商平台将帮助商家宣传、推广物品，从而提高了商家的知名度以及营业额，为商家打造更多财富增值的机会。

这些经济效益体现了该电子商城系统在市场需求和经济利益方面的优势，同时说明该系统的盈利模式和商业潜能。同时，需要注意到经济效益是估算值，具体测试所得可能会出现误差和不确定性，需结合实际数据进行修正和调整。

5.3 市场预测

在该系统经济可行性的市场预测中，根据当前电子商务市场的规模和增长趋势，预计电子商城系统市场将保持持续增长，尤其是在线购物的市场需求，根据相关统计数据和研究报告预测出必定呈现稳步增长态势。因此，该系统的市场前景很好，并有望获得更大的市场份额和竞争优势。预计该系统将在未来几年内迅速获得用户和市场份额，并在快速成长的电商市场中取得良好的增长。

6. 技术可行性

6.1 Java 语言

Java 作为企业级开发最常用的编程语言，其作为该系统的后端开发是非常合适的，加上 Java 代码的可维护性及运行效率，搭配 SpringBoot 框架和 MyBatis 框架协助开发，可以让开发效率大大翻倍。

6.2 MySQL

MySQL 作为一款经久不衰的数据库，作为该系统的数据持久化工具是比较合理的，加上其强大的技术生态圈，以及编程延伸出来的各种 ORM 框架，对 MySQL 的操作是比较友好的。

6.3 HTML、JavaScript

HTML 作为传统网站页面标记语言，用来建设网站是必不可少的。搭配 javaScript 编写的脚本可以完美的与后端服务器完成数据操作，这里再借助 Vue.js 框架针对系统进行模块化控制，大大加强代码效率及可维护性。

7 社会可行性

促进传统商业的转型和创新。 通过该电子商城系统，传统商家可以更加便捷地进入电商领域，拓展营销渠道和销售渠道，而且还能借此提高品牌曝光度和知名度。这将帮助更多的传统商业成功转型并加速创新。

满足用户在线购物的需求。 该电子商城系统通过提供丰富的商品和优质的客户服务，大大简化了消费者购物的流程。这方便了那些没有足够时间进行实体购物或者面对疫情或天气等因素无法离开家门的消费者，同时也能满足那些外地用户的购物需求，为社会提供了更方便的购物体验。

综上，该电子商城系统在促进传统商业转型和创新以及满足用户在线购物需求方面，具有显著的社会可行性。该系统的功能和服务将进一步提高并改善社会的生活质量，同时也将带动电商市场进一步的发展和繁荣。

8. 结论

针对该系统进行经济可行性、技术可行性、社会可行性等进行分析，加上各种调研，该系统的设计与实现是着实可行的。

三、概要设计

1. 引言

1.1 目的

该文档旨在确保系统需求得到完整、一致和准确的表达，通过对用户需求进行概括与分析，确定系统的功能和特性，提高系统的设计质量，并对系统各模块进行整体规划和分解。

1.2 范围

1.2.1 系统目标

本系统的设计目标是提供一个完整的购物体验，包括商品浏览、下单、付款、发货等流程。系统的前端使用 Vue 框架实现，后端使用 Spring Boot 框架实现。系统主要功能包括商品管理、商品分类、订单管理、用户管理（分管理员和普通用户）、购物车管理等。在这个系统中，管理员可以管理商品和订单，普通用户可以注册、登录、查看商品并下单，购物车可以方便用户暂存购物商品。系统实现了基本的商品库存管理，商品分类管理和订单管理，且支持用户通过前台下单并在后台管理系统中查看订单状态。整个系统致力于提供一个简单易用、高效实用的电子商城平台，帮助商家更好地提供商品服务，圆满完成交易。

1.2.2 主要软件需求

该电子商城系统需要实现管理员、普通用户两个个主要角色的功能。管理员需要实现商品和订单管理，包括添加、修改、删除商品信息和查看、删除订单。普通用户需要实现注册、登录、商品浏览、购物车管理、下单、订单查询等功能。购物车需要支持商品添加、删除等操作。对于前端，需要实现简洁易用的页面和优秀的用户体验。对于后端，需要实现高效稳定的服务和容错机制。此外，还需要实现基于业务问题的数据分析和统计，以便于商家识别销售趋势和计划精细化营销策略。最终为商家和用户提供优质的电子商务服务。。

1.2.3 软件设计约束、限制

采用 Spring Boot 框架进行后端开发，前端采用 Vue 框架进行开发；需要实现基本的用户权限控制和数据安全保护机制；必须遵循开发过程中的测试、文档管理、流程管控等规范性要求。

1.3 参考资料

[1] 陈春雷.软件工程导论[M].北京：清华大学出版社,2019.

[2] 董家辉.软件需求工程[M].北京：国防工业出版社,2017.

[3] 韩震.软件系统分析与设计方法[M].北京：电子工业出版社,2018.

[4] Roger S. Pressman.软件工程：实践者的研究方法（第八版）[M].北京：机械工业出版社,2018.

1.4 版本信息

修改编号	修改日期	修改后版本	修改位置	内容概述

2.数据设计

本部分主要描述软件所涉及的外部数据的结构描述。如果数据以数据库文件呈现，则描述表的名称和表字段结构；如果数据以外部文件形式呈现，则要描述文件的内部结构。

2.1 数据对象和形成的数据结构

2.1.1 地址表（address）

描述：地址表，用于存储用户的收货地址信息。
字段：id（主键）, link_user, link_address, link_phone, user_id。其中，link_user 表示联系人，link_address 表示具体地址，link_phone 表示联系电话，user_id 表示所属用户。

2.1.2 头像表（avatar）

描述：头像表，用于存储用户的头像。
字段：id（主键）, type, size, url, md5。其中，type 表示文件类型，size 表示文件大小，url 表示文件路径，md5 表示文件的 md5 值。

2.1.3 轮播图表（carousel）

描述：轮播图表，用于存储商品的轮播图信息。
字段：id（主键）, good_id, show_order。其中，good_id 表示对应的商品 id，show_order 表示播放顺序。

2.1.4 购物车表（cart）

描述：购物车表，用于存储商品加入购物车的信息。
字段：id（主键）, count, create_time, good_id, standard, user_id。其中，count 表示数量，create_time 表示加入时间，good_id 表示商品 id，standard 表示规格，user_id 表示用户 id。

2.1.5 分类表（category）

描述：分类表，用于存储商品分类的信息。
字段：id（主键）, name。其中，name 表示类别名称。

2.1.6 商品表（good）

描述：商品表，用于存储商品的信息。
字段：id（主键）, name, description, discount, sales, sale_money, category_id, imgs, create_time, recommend, is_delete。其中，name 表示商品名称，description 表示描述，discount 表示折扣，sales 表示销量，sale_money 表示销售额，category_id 表示分类 id，imgs 表示商品图片，create_time 表示创建时间，recommend 表示是否推荐，is_delete 表示是否删除。

2.1.7 商品-规格关联表（good_standard）

描述：商品规格表，用于存储商品规格信息。
字段：good_id, value, price, store。其中，good_id 表示商品 id，value 表示规格名称，price 表示该规格的价格，store 表示该规格的库存。

2.1.8 图标表（icon）

描述：图标表，用于存储图标的信息。

字段：id（主键）, value。其中，value 表示图标的识别码。

2.1.9 商品分类-图标关联表（icon_category）

描述：商品分类 - 图标关联表，用于存储商品分类与图标的关联信息。
字段：category_id, icon_id。其中，category_id 表示分类 id，icon_id 表示图标 id。

2.1.10 订单-商品关联表（order_goods）

描述：订单商品表，用于存储订单与商品的关联信息。
字段：id（主键）, order_id, good_id, count, standard。其中，order_id 表示订单 id，good_id 表示商品 id，count 表示数量，standard 表示规格。

2.1.11 规格表（standard）

描述：规格表，用于存储商品规格信息。
字段：goodId（主键）, value, price, store。其中，goodId 表示商品 id，value 表示商品规格，price 表示该规格的价格，store 表示该规格的库存。

2.1.12 系统文件表（sys_file）

描述：系统文件表，用于存储系统文件的信息。
字段：id（主键）, name, type, size, url, is_delete, enable, md5。其中，name 表示文件名称，type 表示文件类型，size 表示文件大小，url 表示文件路径，is_delete 表示是否删除，enable 表示是否启用，md5 表示 md5 值。

2.1.13 用户表（sys_user）

描述：用户表，用于存储用户的信息。
字段：id（主键）, username, password, nickname, email, phone, address, avatar_url, role。其中，username 表示用户名，password 表示密码，nickname 表示昵称，email 表示邮箱，phone 表示手机号码，address 表示地址，avatar_url 表示头像链接，role 表示角色。

2.1.14 订单表（t_order）

描述：订单表，用于存储用户的订单信息。
字段：id（主键）, order_no, total_price, user_id, link_user, link_phone, link_address, state, create_time。其中, order_no 表示订单号, total_price 表示总价, user_id 表示用户 id, link_user

表示联系人，link_phone 表示联系电话，link_address 表示地址，state 表示订单状态，create_time 表示创建时间。

2.2 全局数据

该用户相关数据：用户表（sys_user）、地址表（address）、头像表（avatar）。

商品相关数据：商品表（good）、商品规格表（good_standard）、规格表（standard）、分类表（category）、商品分类 - 图标关联表（icon_category）、订单商品表（order_goods）。

系统文件相关数据：系统文件表（sys_file）。

订单相关数据：订单表（t_order）。

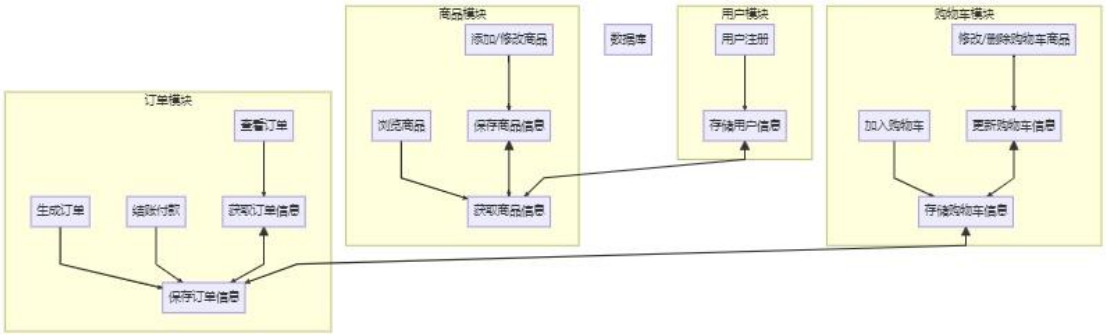
其他：购物车表（cart）、图标表（icon）与上述表格有关联关系。

以上数据包括用户信息、商品信息、系统文件等各方面的数据，是该系统的核心数据。

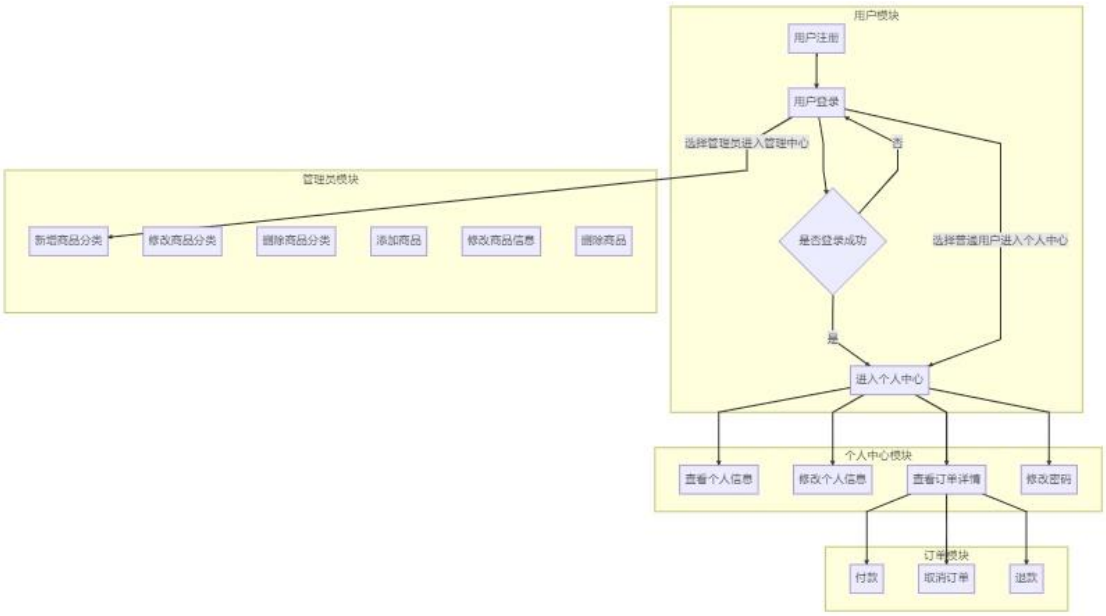
3 体系结构设计

3.1 数据和控制流复审

3.1.1 数据流图

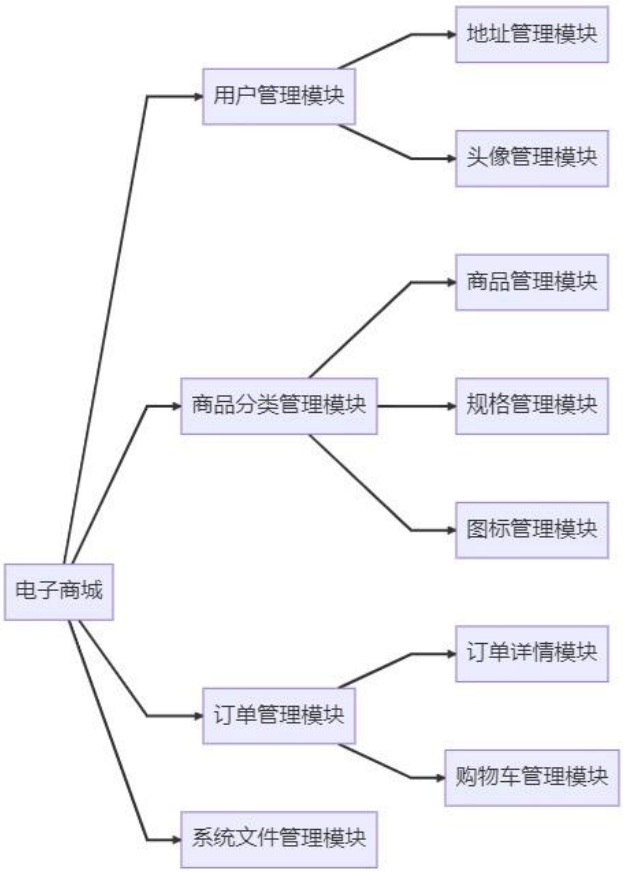


3.1.2 控制流图



3.2 得出的程序结构

根据复审的数据流图，逐步得出软件的逻辑结构组成。利用优化思想，对软件结构图进行优化设计，得出模块层次结构适中的软件结构图。如下所示：



4.界面设计

4.1 登录页面

提供用户名和密码的输入框

提供登录按钮，点击登录后跳转到主页

4.2 注册页面

提供用户名、密码、确认密码的输入框

提供注册按钮，点击后将用户信息提交到服务器进行注册

提供已有账号，跳转登录的入口

4.3 主页

顶部有电商网站名称和搜索框

左侧提供商品分类栏，点击某一项可以查看该分类下的商品列表

右侧有购物车按钮和用户个人中心按钮

展示不同的海报或页面推荐的商品

4.4 商品列表页

显示该分类下所有商品的列表

展示商品的图片、名称、价格等

4.5 商品详情页

显示商品的封面、名称、规格、价格、销量等详细信息

提供立即购买、加入购物车的按钮

4.6 购物车页面

展示购物车中所有的商品、数量、价格、规格等信息

提供添加、移除、修改商品数量的操作

提供提交订单按钮

4.7 订单页面

提供订单列表和订单详情展示

展示订单的基本信息、商品、数量、价格、收货地址、下单时间、支付时间等信息

提供查看订单详情等操作

4.8 个人中心页面

展示用户信息，如头像、用户名、邮箱、手机号码等

提供修改个人信息、修改密码、退出登录的入口

展示用户的订单列表和相关操作

5、模块过程设计

5.1 用户管理模块

用户注册：用户输入用户名、密码，后台进行校验，若校验通过则保存用户信息到数据库

用户登录：用户输入用户名和密码，后台验证用户名和密码是否正确，若验证通过则跳转到主页

个人信息管理：用户可以查看和修改自己的个人信息，如修改密码、更换头像等

地址管理：用户可以添加、修改和删除自己的地址信息

5.2 商品分类管理模块

商品分类添加：管理员选择添加商品分类类型并输入分类名称、添加分类图标等信息，后台保存分类信息到数据库

商品分类修改：管理员选择修改商品分类类型，根据分类名称或分类 id 进行查询，后台更新分类信息到数据库

商品分类删除：管理员选择删除商品分类类型，根据分类名称或分类 id 进行查询，后台从数据库删除对应分类信息

商品管理：管理员可以添加、修改和删除商品，对商品进行分类、输入商品名称、价格、数量等信息，并添加商品图片和规格信息

规格管理：管理员可以添加、修改和删除商品规格类型，如颜色、尺寸等信息

5.3 订单管理模块

生成订单：用户在购物车页面提交订单，后台根据订单商品信息进行计算，生成订单并保存到数据库

订单详情：用户可以查看订单详情，如订单商品信息、订单状态、订单号等

购物车管理：用户可以添加、修改和删除购物车中的商品信息

5.4 系统文件管理模块

系统文件管理模块主要用来处理用户头像图片、商品图片等文件的上传、存储、读取和删除等操作

四、详细设计

1 引言

1.1 编写目的

根据需求分析文档确定的需求，为系统开发提供更为详细、精确、可行的设计方案。在该文档中，我们将具体阐述系统各个阶段的设计方案，包括系统架构、模块划分、数据库设计、界面设计等细节内容，以确保开发人员能够准确理解和实现每一个阶段的开发。同时，该文档也需要提供详细的系统运行流程，通讯协议，算法实现、数据结构的说明等，以方便开发人员写出可靠的代码。另外，在系统维护时，该文档也作为重要依据，提供系统各模块的设计原则、开发要求、版本升级策略等，以确保系统可维护、可扩展、可重用性、安全性等方面要求得以满足。因此，该文档的编写应该全面考虑因素，并尽可能确保其具有清晰性、可读性和易于理解性，以便于开发人员理解、协作和实现。

1.2 项目背景

该电子商城系统是针对线上购物市场的一款应用系统，通过结合 Spring Boot 和 Vue 技术实现了商品管理、商品分类、订单管理、用户管理、购物车管理等相关功能。系统旨在为买家提供一个方便快捷、高效、可靠的购物服务。

1.3 参考资料

[1] 林晓斌. Java 企业级开发实战[M]. 第二版. 北京: 人民邮电出版社, 2017.
[2] 邓俊辉. 数据结构[M]. 北京: 清华大学出版社, 2018.
[3] 刘伟. SpringBoot 企业级应用开发实战[M]. 北京: 电子工业出版社, 2017.
[4] 张磊，马俊昭，王丽华，等. Vue.js 实战[M]. 北京: 人民邮电出版社, 2017.
[5] 郑海波，金丹华，张曼. 电子商务系统设计[M]. 北京: 清华大学出版社, 2009.

2 总体设计

2.1 需求概述

（1）商品管理模块：实现了商品的基本信息录入、图片上传、状态管理等相关功能。

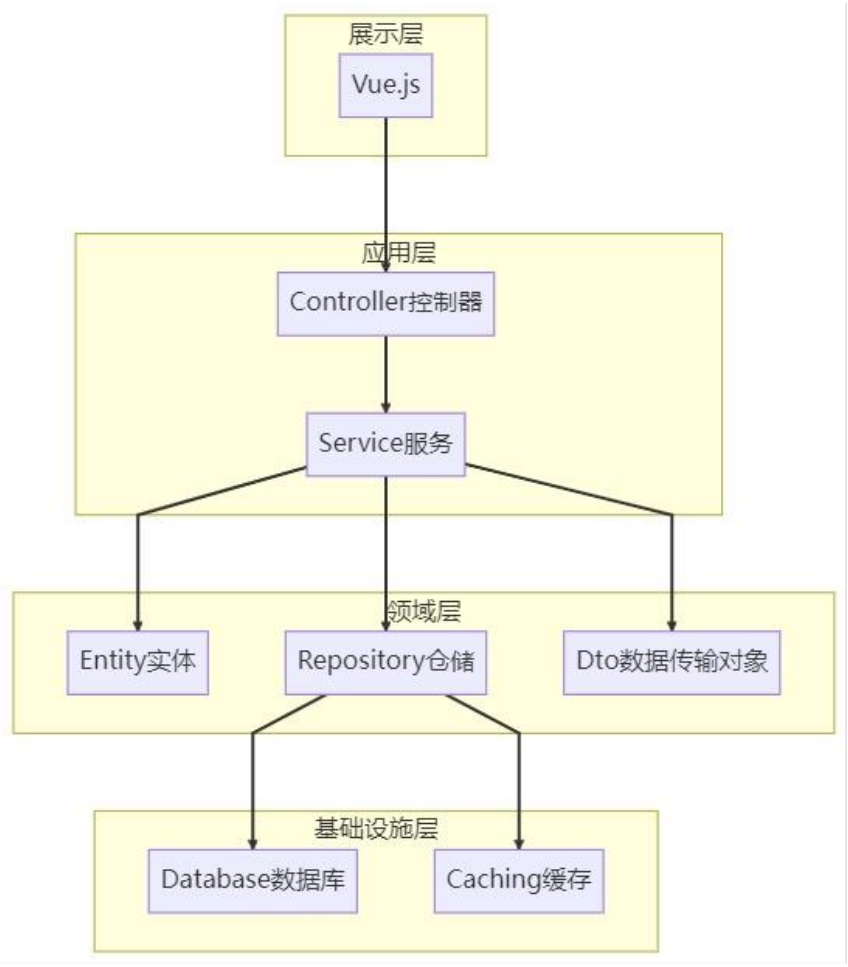
(2) 商品分类模块：实现了分类的增删改查、分类层级管理、商品分类的关联等功能。

(3) 订单管理模块：实现了订单的查询、创建、删除、退货等功能。

(4) 用户管理模块：实现了用户的注册、登录、密码找回、个人信息修改和查询等功能。

(5) 购物车管理模块：实现了添加商品、删除商品、调整数量、结算等功能。

2.2 软件结构



描述：以上的软件结构图，采用了四层架构设计，不同层之间通过依赖关系进行交互。展示层采用 Vue.js 框架来实现，应用层处理各种请求并调用服务层来处理具体业务逻辑，领域层负责封装业务逻辑，基础设施层负责管理底层资源，如数据库和缓存等。该软件结构图体现了高内聚低耦合的设计原则，有利于系统的可维护性和可扩展性。

3 模块设计

3.1 模块基本信息

3.1.1 商品管理模块

模块描述：负责对商品进行增删改查等基本操作，包括商品信息的录入、维护和展示，商品图片的上传、展示和删除，商品状态的管理等功能。

3.1.2 商品分类模块

模块描述：负责商品分类的管理，包括商品分类的层级关系维护、新增、修改、删除等基本操作。

3.1.3 订单管理模块

模块描述：负责订单相关的基础操作，包括订单的查询、创建、删除等功能。

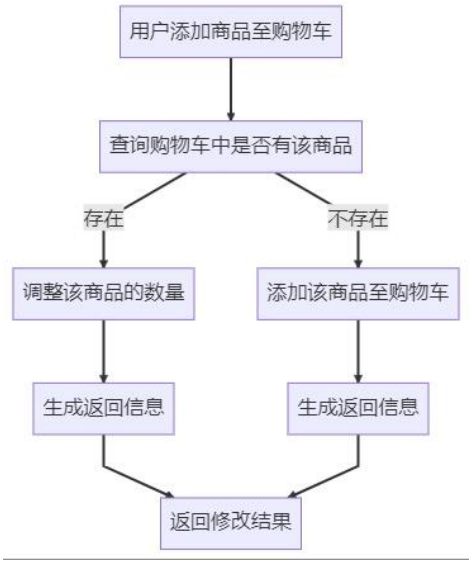
3.1.4 用户管理模块

模块描述：负责用户信息相关功能，包括用户注册、登录、个人信息维护等操作。

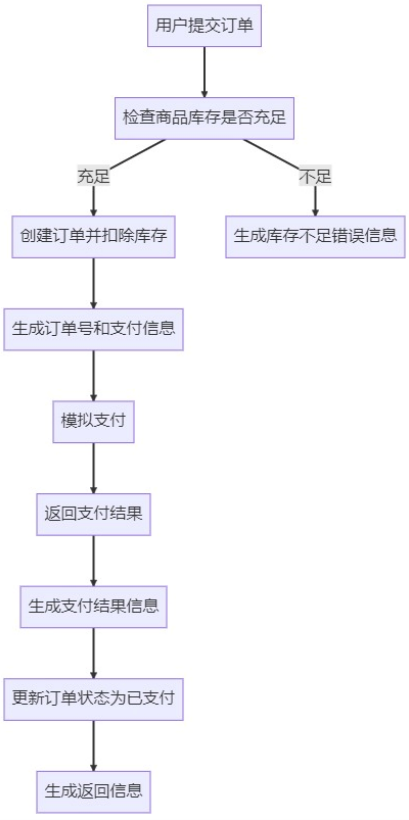
3.1.5 购物车管理模块

模块描述：负责购物车的管理，包括加入购物车、调整商品数量、删除商品、结算等基本操作。

3.2 模块处理逻辑

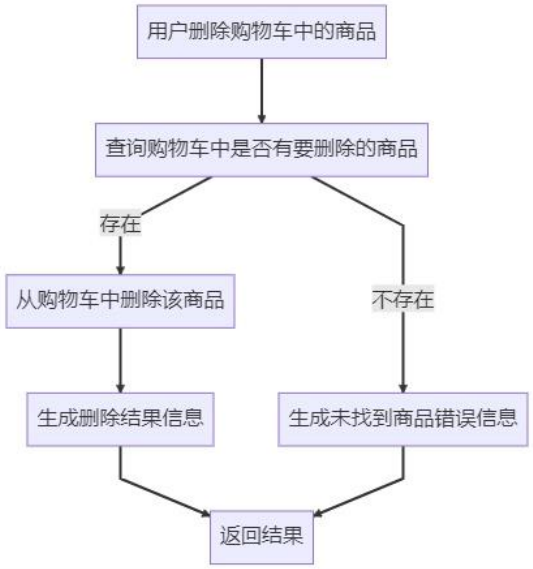


以上流程图是用户在购物车界面添加商品的操作流程。首先查询该商品是否已经在购物车中，若已存在，则调整该商品的数量，否则添加该商品至购物车。最后返回添加或调整的结果信息。这个流程设计简洁，易于理解。

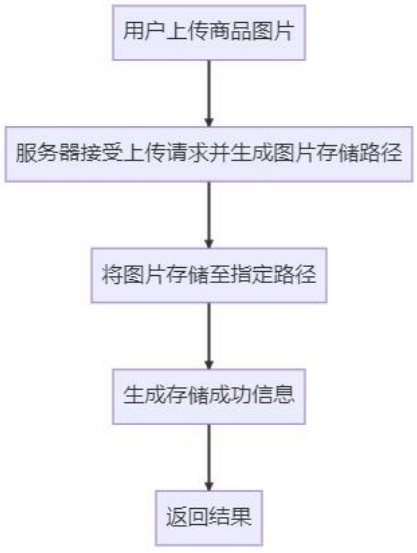


以上流程图是用户提交订单的操作流程。首先检查所有商品的库存是否充足，若充足则创建订单并扣除库存，否则返回库存不足错误信息。然后生成订单号和支付信息，完成支付，

返回支付结果，更新订单状态为已支付，最后生成返回信息。这个流程设计清晰明了，覆盖了电商系统中订单的基本操作。



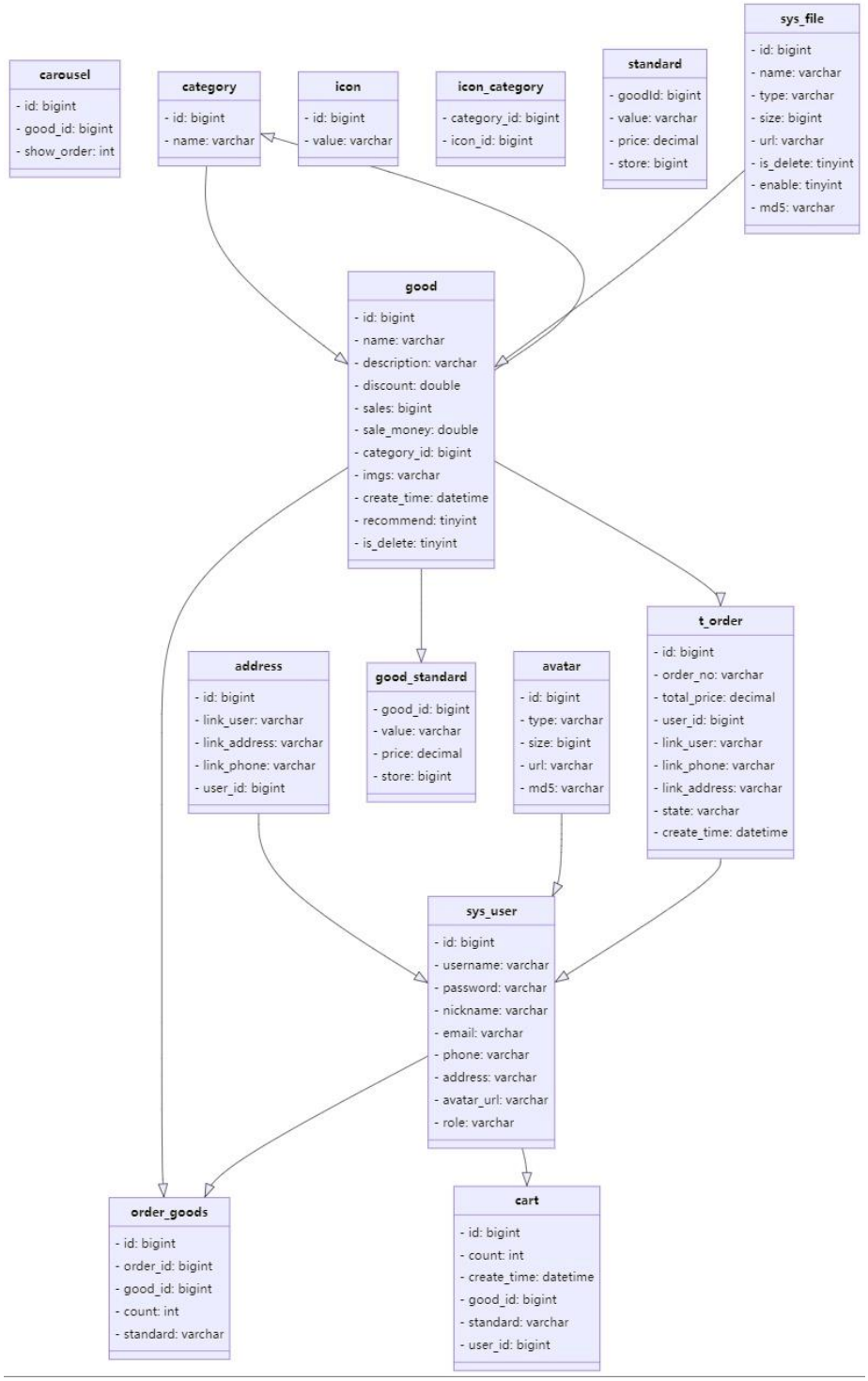
以上流程图是用户在购物车中删除商品的操作流程。首先查询该商品是否在购物车中，若存在则从购物车中删除该商品，否则返回未找到商品错误信息。最后生成删除结果信息并返回给用户。这个流程设计简洁明了，有效实现了购物车中删除商品的功能。



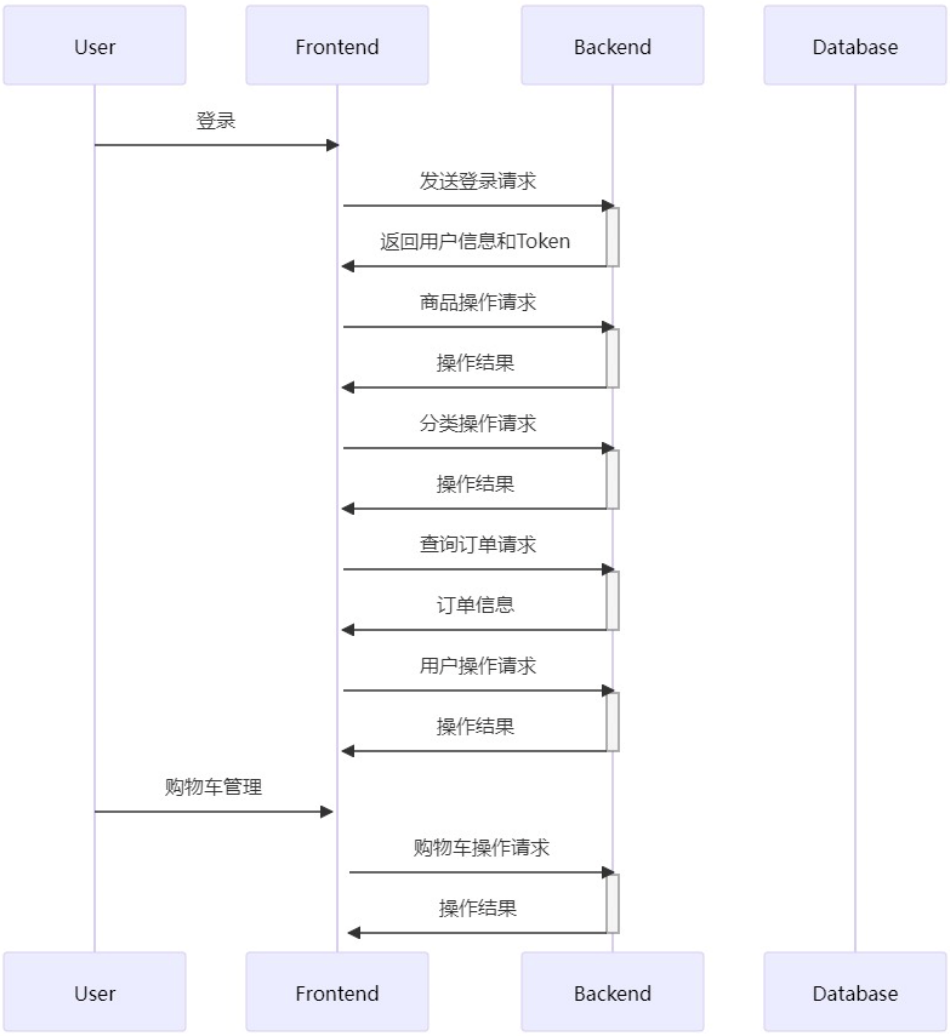
以上流程图是用户上传商品图片的操作流程。用户上传图片后，服务器接受到上传请求并根据图片大小和格式生成存储路径。然后将图片存储至指定路径，并生成存储成功信息返回给用户。这个流程设计简单清晰，有利于实现图片上传的功能。

4 UML 建模

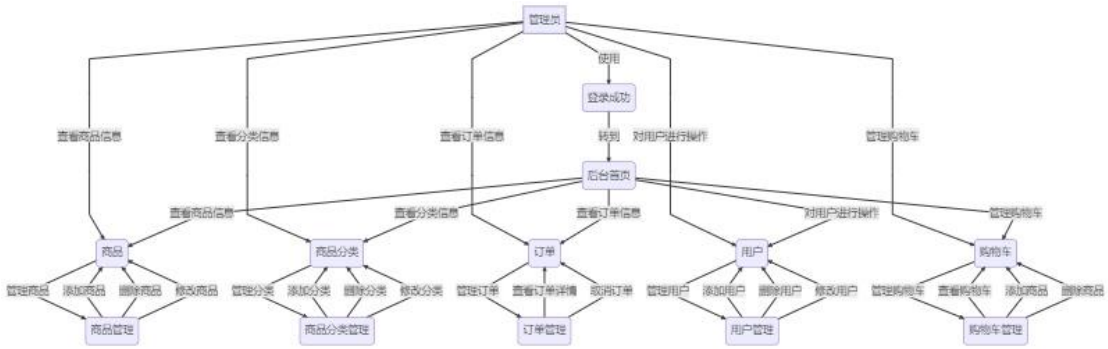
4.1 类图


carousel
- id: bigint
- good_id: bigint
- show_order: int
category
- id: bigint
- name: varchar
icon
- id: bigint
- value: varchar
icon_category
- category_id: bigint
- icon_id: bigint
standard
- goodId: bigint
- value: varchar
- price: decimal
- store: bigint
sys_file
- id: bigint
- name: varchar
- type: varchar
- size: bigint
- url: varchar
- is_delete: tinyint
- enable: tinyint
- md5: varchar
good
- id: bigint
- name: varchar
- description: varchar
- discount: double
- sales: bigint
- sale_money: double
- category_id: bigint
- imgs: varchar
- create_time: datetime
- recommend: tinyint
- is_delete: tinyint
address
- id: bigint
- link_user: varchar
- link_address: varchar
- link_phone: varchar
- user_id: bigint
good_standard
- good_id: bigint
- value: varchar
- price: decimal
- store: bigint
avatar
- id: bigint
- type: varchar
- size: bigint
- url: varchar
- md5: varchar
t_order
- id: bigint
- order_no: varchar
- total_price: decimal
- user_id: bigint
- link_user: varchar
- link_phone: varchar
- link_address: varchar
- state: varchar
- create_time: datetime
sys_user
- id: bigint
- username: varchar
- password: varchar
- nickname: varchar
- email: varchar
- phone: varchar
- address: varchar
- avatar_url: varchar
- role: varchar
order_goods
- id: bigint
- order_id: bigint
- good_id: bigint
- count: int
- standard: varchar
cart
- id: bigint
- count: int
- create_time: datetime
- good_id: bigint
- standard: varchar
- user_id: bigint

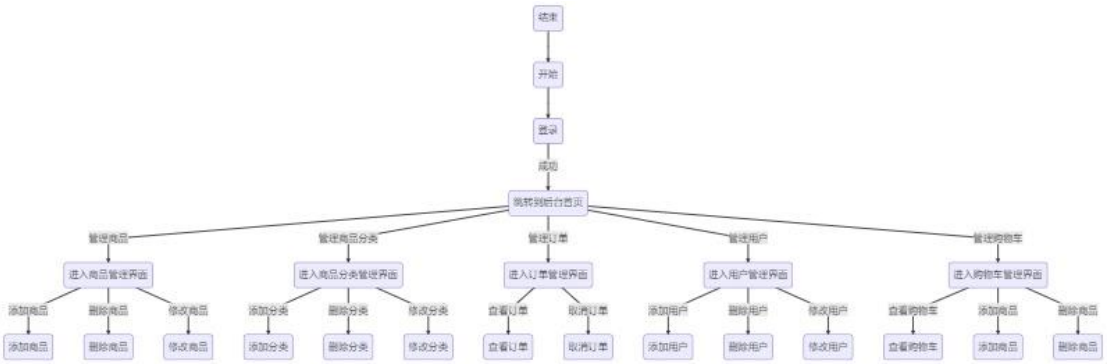
4.2 时序图


User
Frontend
Backend
Database
登录
发送登录请求
返回用户信息和Token
商品操作请求
操作结果
分类操作请求
操作结果
查询订单请求
订单信息
用户操作请求
操作结果
购物车管理
购物车操作请求
操作结果
User
Frontend
Backend
Database

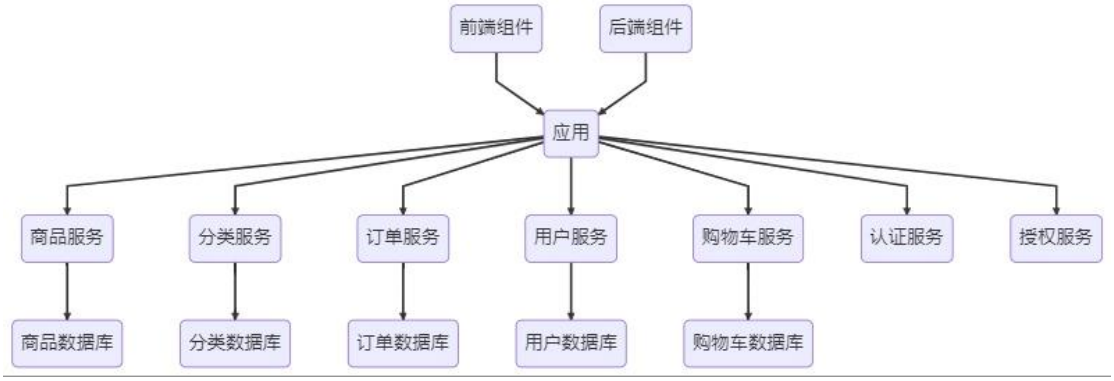
4.3 用例图


管理员
使用
登录成功
查看商品信息
查看分类信息
查看订单信息
转到
对用户进行操作
管理购物车
后台首页
查看商品信息
查看分类信息
查看订单信息
对用户进行操作
管理购物车
商品
商品分类
订单
用户
购物车
管理商品
添加商品
删除商品
修改商品
管理分类
添加分类
删除分类
修改分类
管理订单
查看订单详情
取消订单
管理用户
添加用户
删除用户
修改用户
管理购物车
查看购物车
添加商品
删除商品
商品管理
商品分类管理
订单管理
用户管理
购物车管理

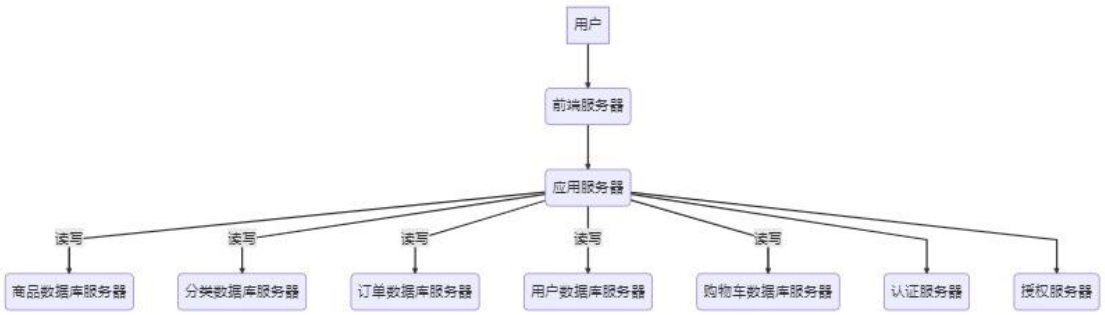
4.4 活动图



4.5 组件图

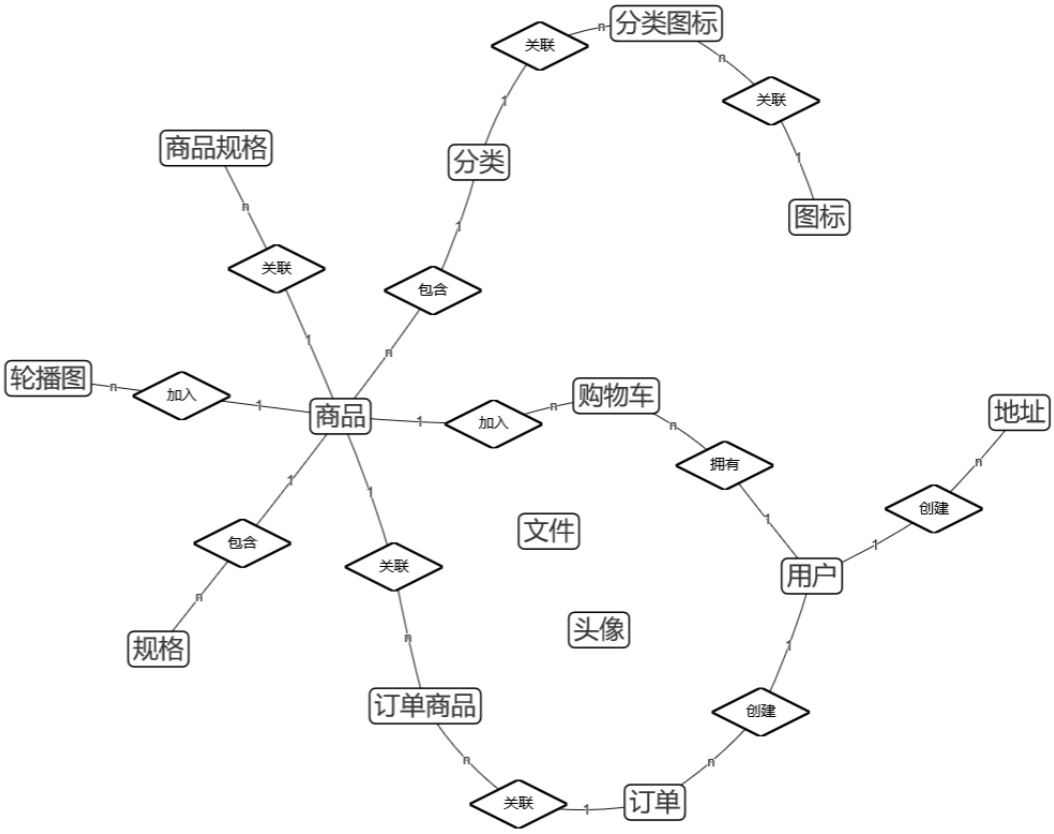


4.6 部署图

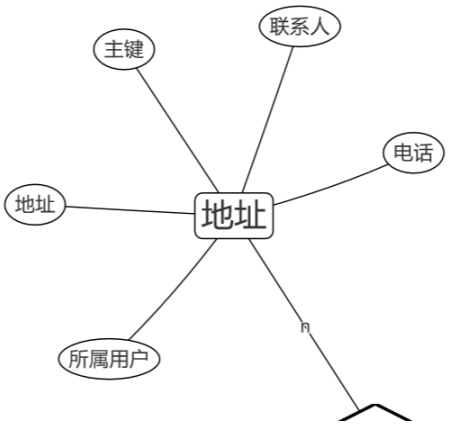


5 数据库设计

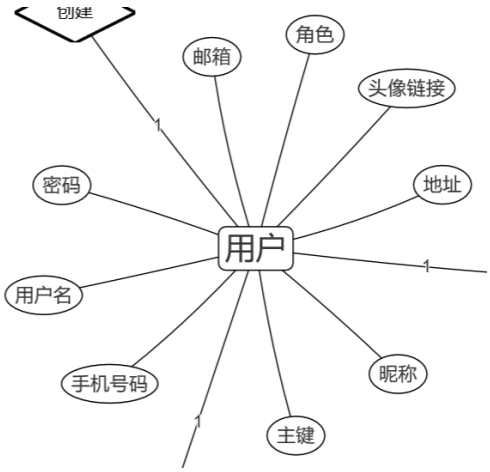
5.1 ER 图



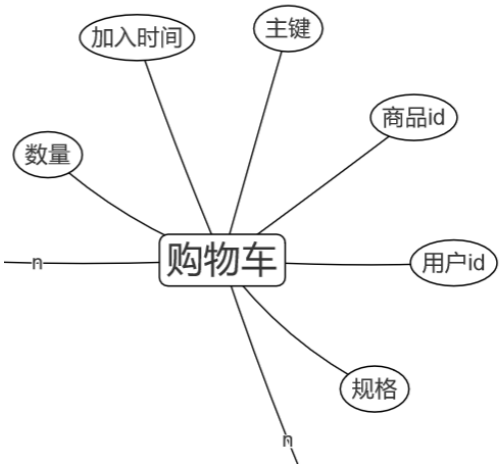
实体关联表



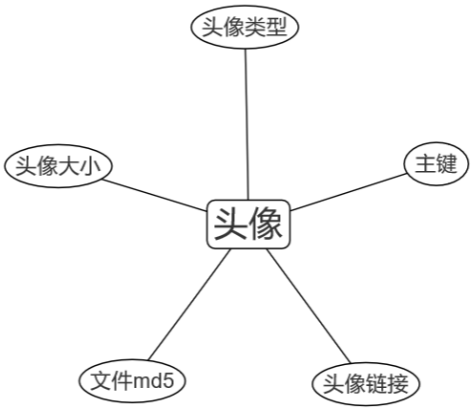
地址表


创建
邮箱
角色
头像链接
1
密码
地址
用户
1
用户名
手机号码
昵称
1
主键

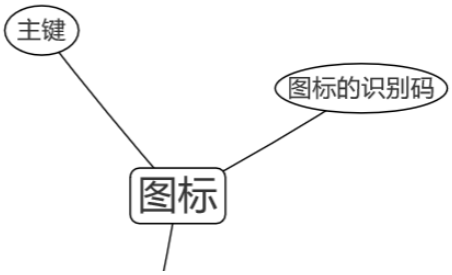
用户表


加入时间
主键
商品id
数量
n
购物车
用户id
规格
n

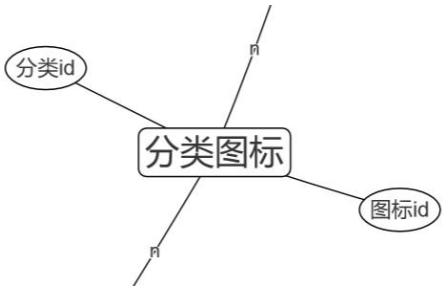
购物车表


头像类型
头像大小
主键
头像
文件md5
头像链接

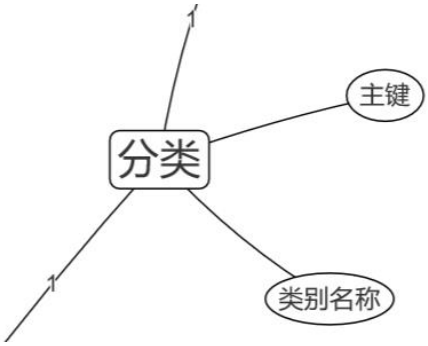
头像表



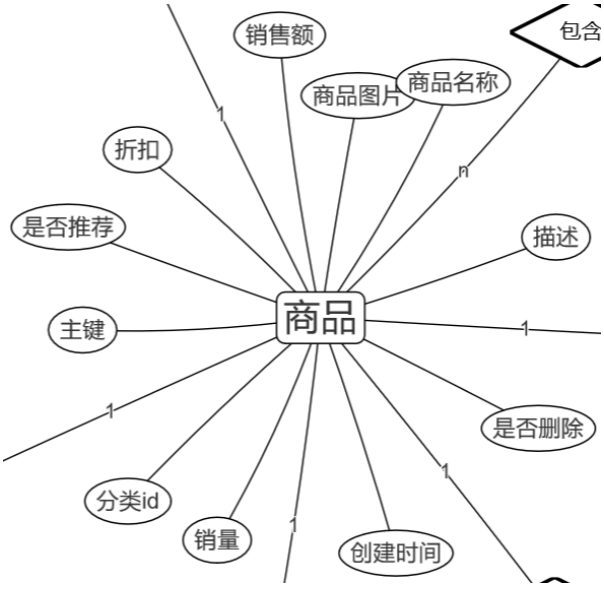
图标表



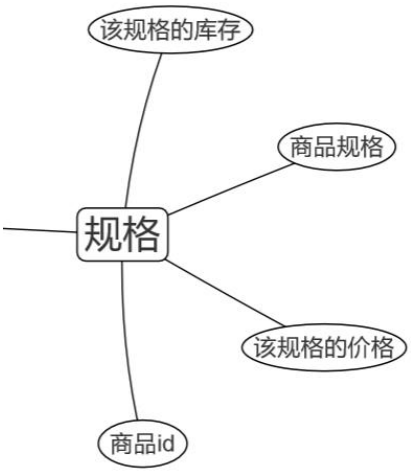
图标分类表



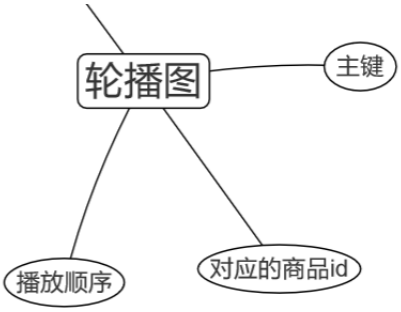
分类表



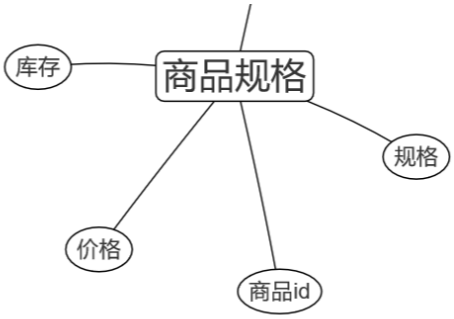
商品表



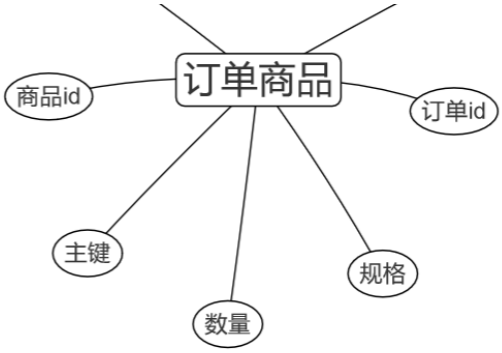
规格表



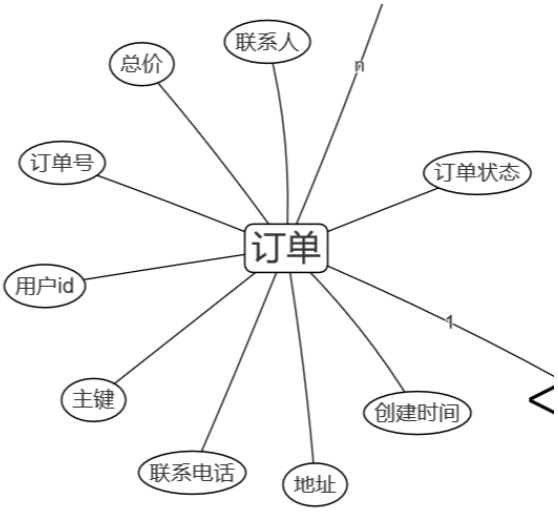
轮播图表



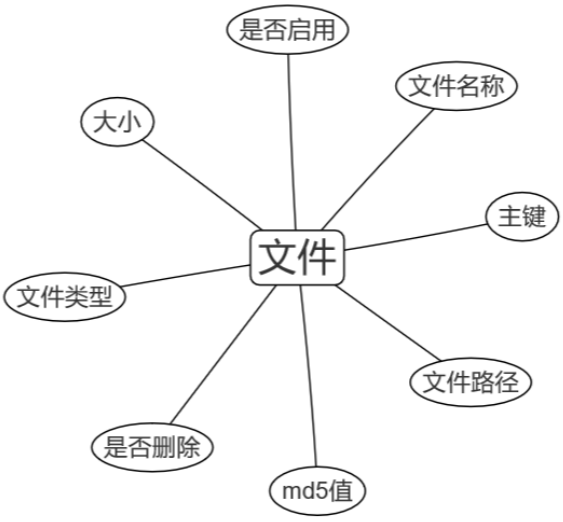
商品规格表



订单商品表



订单表



文件表

5.2 表设计

5.2.1 地址表

数据名称	数据类型	数据描述
id	bigint(0)	主键
link_user	varchar(255)	联系人
link_address	varchar(255)	地址
link_phone	varchar(255)	电话
user_id	bigint(0)	所属用户

5.2.2 头像表

数据名称	数据类型	数据描述
id	bigint(0)	主键
type	varchar(255)	类型
size	bigint(0)	大小
url	varchar(255)	地址

md5	varchar(255)	Md5 值

5.2.3 轮播图表

数据名称	数据类型	数据描述
id	bigint(0)	主键
good_id	bigint(0)	对应的商品 id
show_order	int(0)	播放顺序

5.2.4 购物车表

数据名称	数据类型	数据描述
id	bigint(0)	主键
count	int(0)	数量
create_time	datetime(0)	加入时间
good_id	bigint(0)	商品 id
standard	varchar(255)	规格

数据名称	数据类型	数据描述
user_id	bigint(0)	用户 id

5.2.5 分类表

数据名称	数据类型	数据描述
id	bigint(0)	主键
name	varchar(255)	类别名称

5.2.6 商品表

数据名称	数据类型	数据描述
id	bigint(0)	主键
name	varchar(255)	商品名称
description	varchar(1600)	描述
discount	double(10, 2)	折扣
sales	bigint(0)	销量

数据名称	数据类型	数据描述
sale_money	double(10, 2)	销售额
category_id	bigint(0)	分类 id
imgs	varchar(255)	商品图片
create_time	datetime(0)	创建时间
recommend	tinyint(1)	是否推荐。0 不推荐，1 推荐
is_delete	tinyint(1)	是否删除，0 未删除，1 删除

5.2.7 商品规格关联表

数据名称	数据类型	数据描述
good_id	bigint(0)	商品 id
value	varchar(255)	规格
price	decimal(10, 2)	价格
store	bigint(0)	库存

5.2.8 图标表

数据名称	数据类型	数据描述
id	bigint(0)	主键
value	varchar(255)	图标的识别码

5.2.9 图标分类关联表

数据名称	数据类型	数据描述
category_id	bigint(0)	分类 id
icon_id	bigint(0)	图标 id

5.2.10 订单商品关联表

数据名称	数据类型	数据描述
id	bigint(0)	主键
order_id	bigint(0)	订单 id
good_id	bigint(0)	商品 id
count	int(0)	数量

数据名称	数据类型	数据描述
standard	varchar(1600)	规格

5.2.11 规格表

数据名称	数据类型	数据描述
goodId	bigint(0)	商品 id
value	varchar(255)	商品规格
price	decimal(10, 2)	该规格的价格
store	bigint(0)	该规格的库存

5.2.12 系统文件表

数据名称	数据类型	数据描述
id	bigint(0)	主键
name	varchar(255)	文件名称
type	varchar(255)	文件类型

数据名称	数据类型	数据描述
size	bigint(0)	大小
url	varchar(255)	文件路径
is_delete	tinyint(1)	是否删除
enable	tinyint(1)	是否启用
md5	varchar(255)	md5 值

5.2.13 用户表

数据名称	数据类型	数据描述
id	bigint(0)	主键
username	varchar(255)	用户名
password	varchar(255)	密码
nickname	varchar(255)	昵称
email	varchar(255)	邮箱
phone	varchar(255)	手机号码

数据名称	数据类型	数据描述
address	varchar(1600)	地址
avatar_url	varchar(255)	头像链接

5.2.14 订单表

数据名称	数据类型	数据描述
id	bigint(0)	主键
order_no	varchar(255)	订单号
total_price	decimal(10, 2)	总价
user_id	bigint(0)	用户 id
link_user	varchar(255)	联系人
link_phone	varchar(255)	联系电话
link_address	varchar(255)	地址
state	varchar(255)	订单状态
create_time	datetime(0)	创建时间

6 接口设计

6.1 外部接口

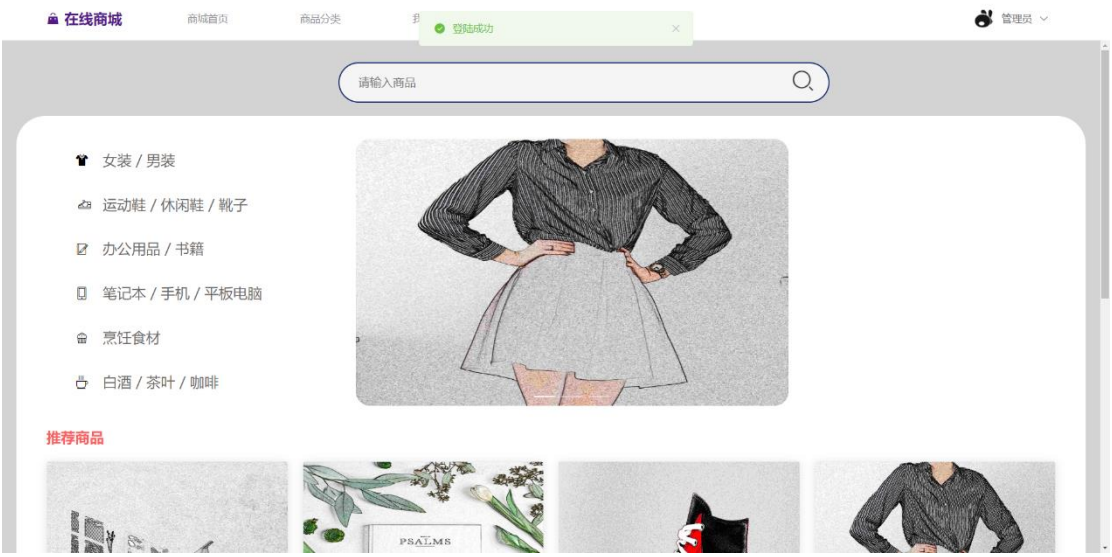
6.1.1 登录界面


```
<div class="title">
        <b>登录在线商城</b>
      </div>
      <div style="margin-top: 30px">
        <el-form label-width="70px">
          <el-form-item label="用户名">
            <el-input v-model.trim="user.username" aria-
required="true"></el-input>
          </el-form-item>
          <el-form-item label="密码" style="margin-top: 25px">
            <el-input v-model.trim="user.password" show-password aria-
required="true"></el-input>
          </el-form-item>
          <el-form-item style="margin: 30px 80px">
            <el-button type="success" @click="onSubmit">登录</el-button>
            <el-button @click="$router.push('/register')">注册</el-
button>
          </el-form-item>
        </el-form>
      </div>
```

6.1.2 注册界面


```
<div class="title">
        <b>注 册</b>
      </div>
      <div style="margin-top: 30px">
        <el-form label-width="70px">
          <el-form-item label="用户名">
            <el-input v-model.trim="user.username" aria-
required="true"></el-input>
          </el-form-item>
          <el-form-item label="密码" style="margin-top: 25px">
            <el-input v-model.trim="user.password" show-password aria-
required="true"></el-input>
          </el-form-item>
          <el-form-item label="确认密码" style="margin-top: 25px">
            <el-input v-model.trim="user.confirmPassword" show-password
aria-required="true"></el-input>
          </el-form-item>
          <el-form-item style="margin: 30px 80px">
            <el-button type="success" @click="onSubmit">注册</el-button>
            <el-button @click="$router.push('/login')">返回</el-button>
          </el-form-item>
        </el-form>
      </div>
```

6.1.3 商城首页



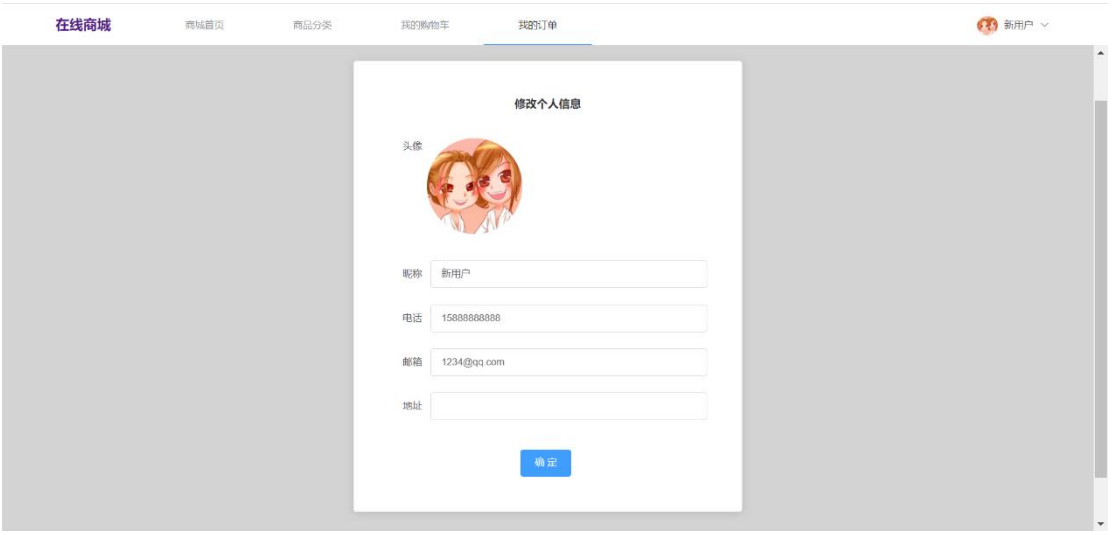

```
<el-container style="height: 100%;width:100%;">
  <el-header style="background-color: white">
    <Navagation :user="user"
                :role="role"
                :login-status="loginStatus"
    ></Navagation>
  </el-header>


  <el-main style="background-color: lightgrey;width:100%;">

    <router-view />
  </el-main>

</el-container>
```

6.1.4 个人信息界面



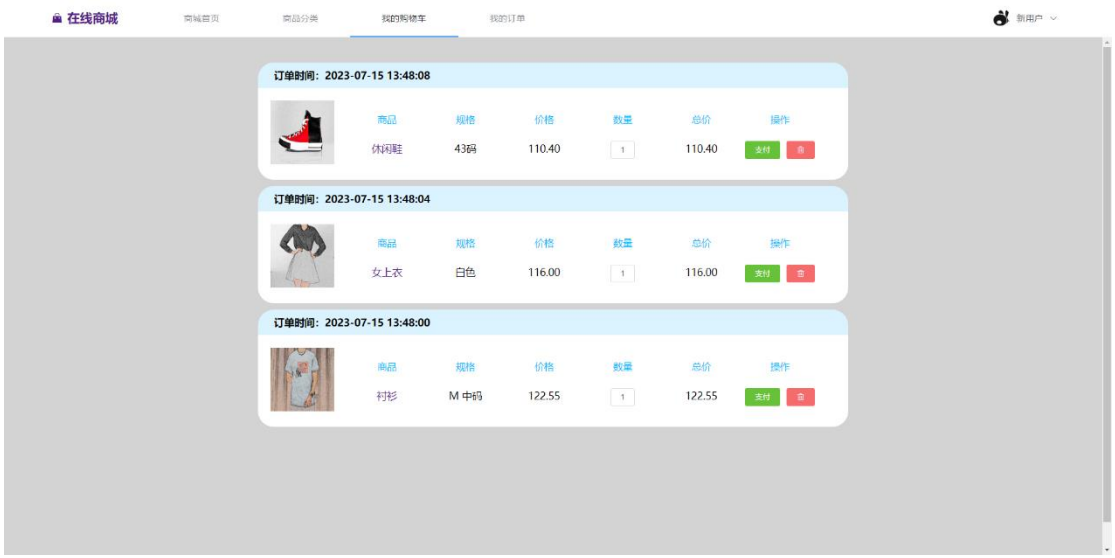

```
<el-card class="card">
    <div style="text-align: center; margin-bottom: 30px">
      <b>修改个人信息</b>
    </div>

    <el-form label-width="60px">
      <el-form-item label="头像">
        <el-upload
          class="avatar-uploader"
          :action="baseApi + '/avatar'"
          :headers="token"
          :show-file-list="false"
          :on-success="handleAvatarSuccess"
        >
          <img
            v-if="form.avatarUrl"
            :src="baseApi + form.avatarUrl"
            class="avatar"
          />
          <i v-else class="el-icon-plus avatar-uploader-icon"></i>
        </el-upload>
      </el-form-item>

      <el-form-item label="昵称">
        <el-input v-model="form.nickname" autocomplete="off"></el-input>
      </el-form-item>
      <el-form-item label="电话">
        <el-input v-model="form.phone" autocomplete="off"></el-input>
      </el-form-item>
      <el-form-item label="邮箱">
```

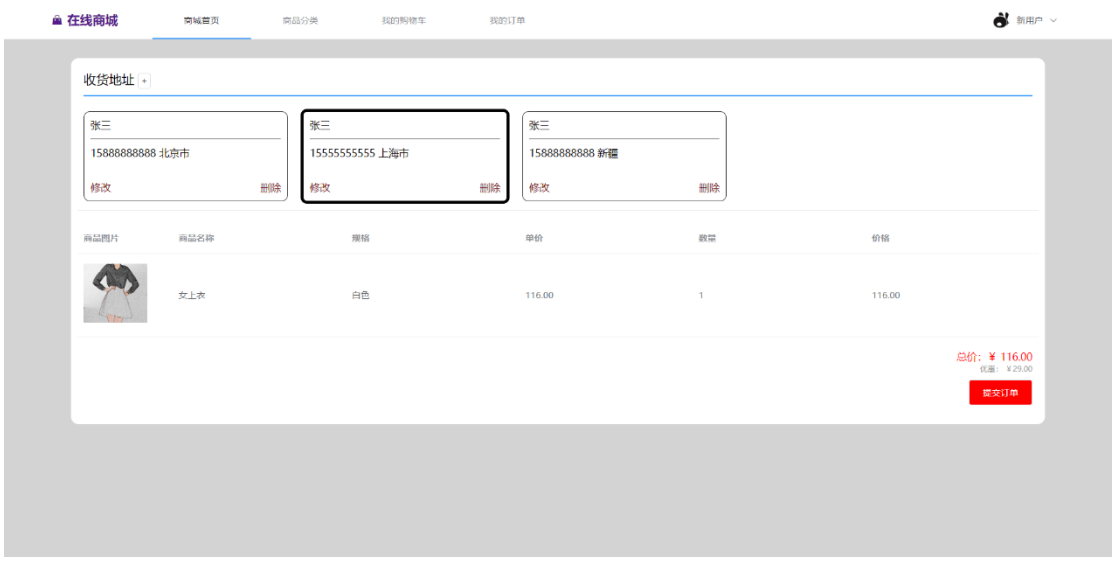
```
      <el-input v-model="form.email" autocomplete="off"></el-input>
    </el-form-item>
    <el-form-item label="地址">
      <el-input v-model="form.address" autocomplete="off"></el-input>
    </el-form-item>
    <el-button
      type="primary"
      style="margin-left: 190px; margin-top: 20px"
      @click="save"
      >确 定</el-button
    >
  </el-form>
  <el-popover placement="right" width="200" trigger="click">
    <el-form>
      <el-form-item label="新密码">
        <el-input
          type="password"
          v-model="resetPsw.newPassword"
          autocomplete="off"
        ></el-input>
      </el-form-item>
      <el-form-item label="确认密码">
        <el-input
          type="password"
          v-model="resetPsw.confirmPassword"
          autocomplete="off"
        ></el-input>
      </el-form-item>
      <el-button size="mini" type="primary" @click="toResetPassword"
        >确 定</el-button
      >
    </el-form>
    <el-button
      slot="reference"
      type="warning"
      style="margin-left: 190px; margin-top: 20px"
      @click="resetPsw = { newPassword: '', confirmPassword: '' }"
      >重置密码</el-button
    >
  </el-popover>
</el-card>
```

6.1.5 购物车界面



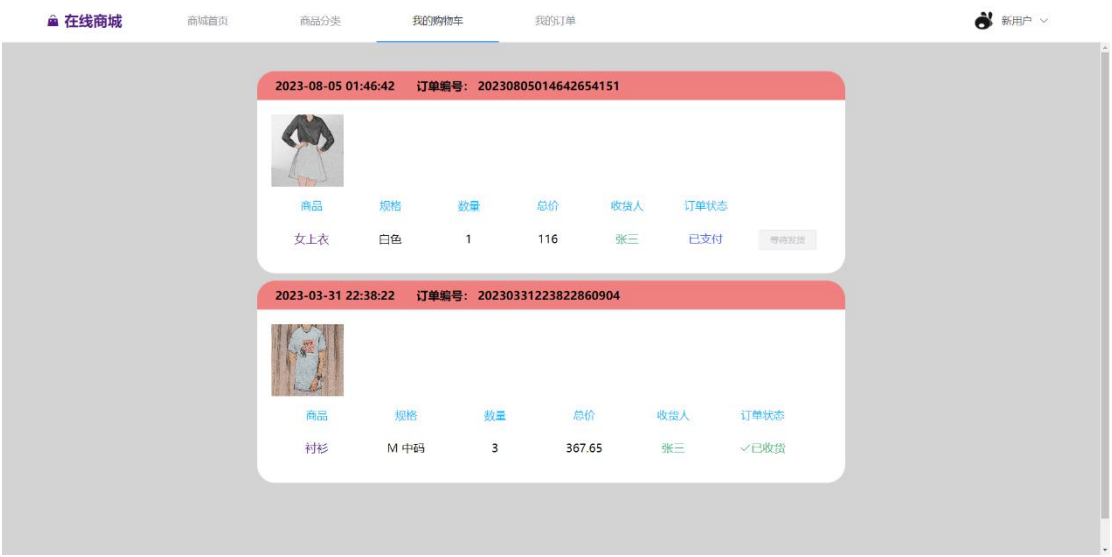

```
<div style="width: 55%;height:100%;margin: 20px auto">
  <div v-if="carts.length === 0" class="empty-box">
    <span style="font-family: 华文彩云;font-size: 40px" >购物车是空的哦
</span>
  </div>
  <template v-for="cart in carts">
    <cart-item :cart="cart" @delete="delItem" :key="cart.id"
style="margin-bottom: 10px"></cart-item>
  </template>
</div>
```

6.1.6 地址信息界面



```
<el-dialog title="地址信息" :visible.sync="dialogFormVisible">
        <el-form label-width="90px" style="padding: 0 60px">
          <el-form-item label="联系人">
            <el-input v-model="address.linkUser"
autocomplete="off"></el-input>
          </el-form-item>
          <el-form-item label="联系电话">
            <el-input v-model="address.linkPhone"
autocomplete="off"></el-input>
          </el-form-item>
          <el-form-item label="地址">
            <el-input
              v-model="address.linkAddress"
              autocomplete="off"
            ></el-input>
          </el-form-item>
        </el-form>
        <div slot="footer" class="dialog-footer">
          <el-button @click="dialogFormVisible = false">取 消</el-
button>
          <el-button type="primary" @click="saveAddress">确 定</el-
button>
        </div>
      </el-dialog>
```

6.1.7 订单界面



```
<div class="header" style="padding-left: 25px;">
    <span style="line-height: 40px">{{order.create_time}}</span>
    <span style="line-height: 40px;margin-left: 30px">订单编号：
{{order.order_no}}</span>
  </div>
  <div class="body">
    <div style="display: inline-block;margin-right: 20px">
      <router-link :to="'goodview/'+order.good_id">
        <img :src="baseApi + order.imgs" style="width:
100px;height:100px">
      </router-link>
    </div>
    <div style="display: inline-block;line-height: 40px" >
      <table>
        <tr>
          <th>商品</th>
          <th>规格</th>
          <th>数量</th>
          <th>总价</th>
          <th>收货人</th>
          <th>订单状态</th>
        </tr>
        <tr>
          <a :href="'goodview/'+order.good_id">
            <td>{{order.good_name}}</td>
          </a>
          <td>{{order.standard}}</td>
          <td>{{order.count}}</td>
          <td>{{order.total_price}}</td>
          <el-popover
              placement="bottom-start"
              width="200"
              trigger="hover"
              :content=address>
            <td slot="reference" style="color:
#42b983">{{ order.link_user }}</td>
          </el-popover>
<!--          订单状态-->
          <template v-if="order.state==='已发货'">
            <td style="color: #42b983">{{order.state}}</td>
            <td>
              <el-button style="margin-left: 20px;" size="mini"
type="primary" @click="receive">确认收货</el-button>
```

```
            </td>
          </template>

          <template v-else-if="order.state==='已收货'">
            <td style="color: #42b983"><a class="el-icon-
check"></a>{{order.state}}</td>
          </template>

          <template v-else-if="order.state==='已支付'">
            <td style="color: #3b62f8"> {{order.state}}</td>
            <td>
              <el-button size="mini" type="info" plain disabled>等待发货
</el-button>
            </td>
          </template>

          <template v-else>
            <td>{{order.state}}</td>
            <td>
              <el-button style="margin-left: 20px" size="mini"
type="success" @click="pay">去支付</el-button>
            </td>
          </template>

        </tr>
      </table>
    </div>
  </div>
```

6.1.8 管理员界面

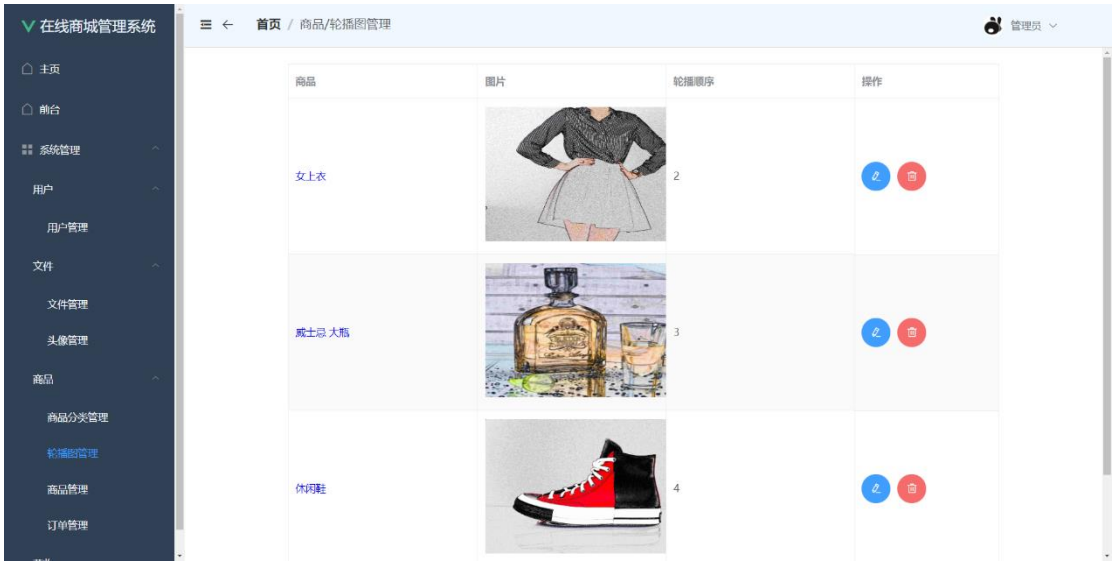

```
  <div style="height: 100%">
    <el-container style="height: 100%">
      <!--      侧边栏-->
      <el-aside
        :width="sideWidth + 'px'"
        style="background-color: rgb(238, 241, 246); height: 100%"
      >
        <Aside :is-collapse="isCollapse"></Aside>
      </el-aside>

      <el-container>
        <!--        导航栏-->
        <el-header
          style="border-bottom: 1px solid #ccc; background-color:
aliceblue"
        >
          <Header
            :collapse-icon="collapseIcon"
            :collapse-title="collapseTitle"
            @collapse="handleCollapse"
            :user="user"
          ></Header>
        </el-header>

        <el-main :class="{bk: $route.path=='/manage/home'}">
          <router-view @refresh="getUser" />
        </el-main>
      </el-container>
    </el-container>
```

```
  </div>
```

6.1.9 轮播图管理界面

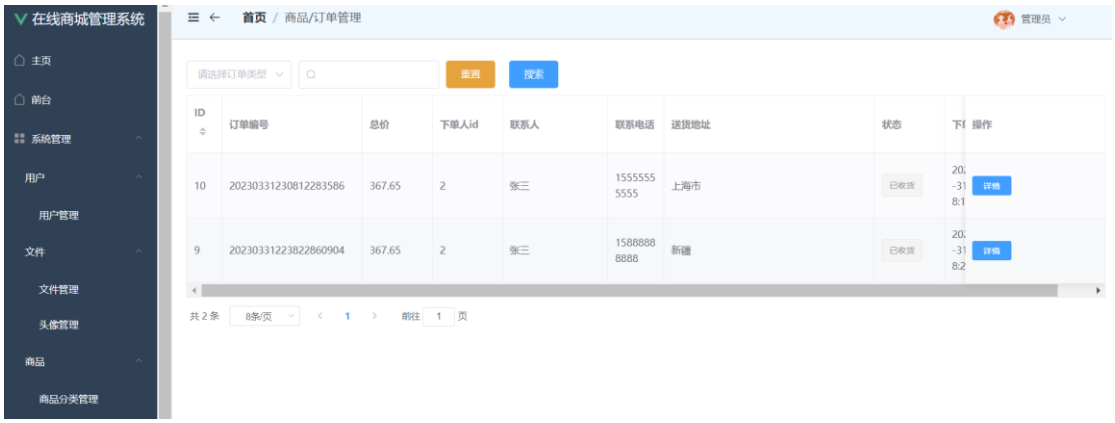



```
<div>
      <el-table :data="tableData" border stripe style="width:
80%;margin: 2px auto">
        <el-table-column label="商品">
          <template slot-scope="scope">
            <a :href="'/goodView/'+scope.row.goodId">{{scope.row.goodNam
e}}</a>
          </template>
        </el-table-column>
        <el-table-column  label="图片" >
          <template   slot-scope="scope">
            <img :src="baseApi + scope.row.img" width="300" height="185"
/>
          </template>
        </el-table-column>
        <el-table-column prop="showOrder" label="轮播顺序"></el-table-
column>

        <el-table-column
            fixed="right"
            label="操作"
            width="200">
          <template slot-scope="scope">
```

```
            <el-button type="primary" icon="el-icon-edit"
circle  @click="edit(scope.row)"></el-button>
            <el-popconfirm
                @confirm="del(scope.row.id)"
                title="确定删除？"
            >
              <el-button type="danger" icon="el-icon-delete" circle
slot="reference" style="margin-left: 10px"></el-button>
            </el-popconfirm>
          </template>
        </el-table-column>
      </el-table>
    </div>
```

6.1.10 订单管理界面




```
<el-table :data="tableData" border stripe style="width: 100%">
      <el-table-column prop="id" label="ID" width="50" sortable> </el-
table-column>
      <el-table-column prop="orderNo" label="订单编号" width="200"></el-
table-column>
      <el-table-column prop="totalPrice" label="总价" width="100"></el-
table-column>
      <el-table-column prop="userId" label="下单人 id" width="100"></el-
table-column>
      <el-table-column prop="linkUser" label="联系人" width="150"></el-
table-column>
      <el-table-column prop="linkPhone" label="联系电话"></el-table-
column>
      <el-table-column prop="linkAddress" label="送货地址"
width="300"></el-table-column>
      <el-table-column prop="state" label="状态" width="100">
```

```
        <template slot-scope="scope">
          <el-tag type="success" v-if="scope.row.state==='已支付
'">{{scope.row.state}}</el-tag>
          <el-tag type="primary" v-if="scope.row.state==='已发货
'">{{scope.row.state}}</el-tag>
          <el-tag type="info" v-if="scope.row.state==='已收货
'">{{scope.row.state}}</el-tag>
        </template>
      </el-table-column>
      <el-table-column prop="createTime" label="下单时间"></el-table-
column>
      <el-table-column
          fixed="right"
          label="操作"
          width="200">
        <template slot-scope="scope">
          <el-button type="primary"
size="mini"  @click="showDetail(scope.row)">详情</el-button>
          <el-popconfirm
              @confirm="delivery(scope.row)"
              title="确定发货吗？"
              v-if="scope.row.state==='已支付'"
          >
            <el-button type="primary" size="mini" slot="reference"
style="margin-left: 10px">发货</el-button>
          </el-popconfirm>
        </template>
      </el-table-column>
    </el-table>
```

6.1.11 商品分类管理界面


```
<el-table :data="icons" stripe>
        <!-- 下级分类表-->
        <el-table-column type="expand" label="下级分类" width="100px">
          <template slot-scope="scope">
            <el-table
              :data="scope.row.categories"
              :header-cell-style="{ background: '#cbefea', color:
'black' }"
            >
              <el-table-column label="分类 id" prop="id"></el-table-
column>
              <el-table-column label="分类名称" prop="name"></el-table-
column>
              <el-table-column label="操作">
                <template slot-scope="scope">
                  <el-button
                    type="primary"
                    size="mini"
                    @click="handleEditCategory(scope.row)"
                    >修改</el-button
                  >

                  <el-popconfirm
```

```
                  @confirm="deleteCategory(scope.row)"
                  title="确定删除？"
                >
                  <el-button
                    type="danger"
                    size="mini"
                    slot="reference"
                    >删除</el-button
                  >
                </el-popconfirm>
              </template>
            </el-table-column>
          </el-table>
        </template>
        <!---->
      </el-table-column>
      <el-table-column label="id" prop="id" width="60px"></el-table-
column>
      <el-table-column label="icon">
        <template slot-scope="scope">
          <i class="iconfont" v-html="scope.row.value"></i>
        </template>
      </el-table-column>

      <el-table-column fixed="right" label="操作" width="200">
        <template slot-scope="scope">
          <el-button
            type="primary"
            icon="el-icon-edit"
            circle
            @click="handleEditIcon(scope.row)"
          ></el-button>
          <el-button
            type="success"
            icon="el-icon-plus"
            circle
            @click="handleAddCategory(scope.row)"
          ></el-button>

          <el-popconfirm
            @confirm="deleteIcon(scope.row.id)"
            title="确定删除？"
          >
            <el-button
```

```
                type="danger"
                icon="el-icon-delete"
                circle
                slot="reference"
                style="margin-left: 10px"
              ></el-button>
            </el-popconfirm>
          </template>
        </el-table-column>
      </el-table>
```

6.1.12 用户管理界面


```
<el-table
      :data="tableData"
      background-color="black"
      @selection-change="handleSelectionChange"
    >
      <el-table-column type="selection"></el-table-column>
      <el-table-column prop="id" label="id" width="100"></el-table-
column>
      <el-table-column
        prop="username"
        label="账号"
        width="150"
      ></el-table-column>
      <el-table-column label="身份" width="150">
        <template slot-scope="scope">
```

```
          <span v-if="scope.row.role === 'user'">用户</span>
          <span v-if="scope.row.role === 'admin'">管理员</span>
        </template>
      </el-table-column>
      <el-table-column
        prop="nickname"
        label="昵称"
        width="180"
      ></el-table-column>
      <el-table-column prop="phone" label="电话" width="180"></el-table-
column>
      <el-table-column prop="email" label="邮箱" width="180"></el-table-
column>
      <el-table-column
        prop="address"
        label="地址"
        width="350"
      ></el-table-column>
      <el-table-column label="操作">
        <template slot-scope="scope">
          <el-button size="mini" type="success"
@click="handleEdit(scope.row)"
            >编辑</el-button
          >
          <el-button
            size="mini"
            type="danger"
            @click="handleDelete(scope.row.id)"
            >删除</el-button
          >
        </template>
      </el-table-column>
    </el-table>
```

6.1.13 统计界面


```
<el-tabs v-model="activeName" @tab-click="handleClick">
      <el-card
        style="
          display: inline-block;
          margin-left: 50px;
          margin-top: 30px;
          font-weight: bold;
          font-size: 22px;
          color: #ffb02a;
        "
        >￥总计：{{ total | numFilter }}</el-card
      >
      <!--      柱状图-->
      <el-tab-pane label="各类收入柱状图" name="bar">
        <div
          id="bar"
          style="width: 1200px; height: 500px; margin: auto auto"
        ></div>
      </el-tab-pane>
      <!--      饼图-->
      <el-tab-pane label="各类收入饼图" name="pie">
        <div
          id="pie"
          style="width: 600px; height: 600px; margin: 10px auto"
        ></div>
      </el-tab-pane>
      <!--  本周收入折线图-->
      <el-tab-pane label="本周收入" name="line1">
        <div
          id="weekLine"
```

```
          style="width: 900px; height: 500px; margin: 10px auto"
        ></div>
      </el-tab-pane>
      <!-- 本月收入折线图-->
      <el-tab-pane label="本月收入" name="line2">
        <div
          id="monthLine"
          style="width: 1500px; height: 500px; margin: 10px auto"
        ></div>
      </el-tab-pane>
    </el-tabs>
```

6.2 内部接口

6.2.1 登录接口

```
@PostMapping("/login")
public Result login(@RequestBody LoginForm loginForm) {
    UserDTO dto = userService.login(loginForm);
    return Result.success(dto);
}
```

6.2.2 注册接口

```
@PostMapping("/register")
public Result register(@RequestBody LoginForm loginForm) {
    User user = userService.register(loginForm);
    return Result.success(user);
}
```

6.2.3 地址管理接口

```
/*
查询
*/
@GetMapping("/{userId}")
public Result findAllById(@PathVariable Long userId) {
    return Result.success(addressService.findAllById(userId));
```

```
}

@GetMapping
public Result findAll() {
    List<Address> list = addressService.list();
    return Result.success(list);
}


/*
保存
*/
@PostMapping
public Result save(@RequestBody Address address) {
    boolean b = addressService.saveOrUpdate(address);
    if(b){
        return Result.success();
    }else{
        return Result.error(Constants.CODE_500,"保存地址失败");
    }

}

@PutMapping
public Result update(@RequestBody Address address) {
    addressService.updateById(address);
    return Result.success();
}

/*
删除
*/
@DeleteMapping("/{id}")
public Result delete(@PathVariable Long id) {
    addressService.removeById(id);
    return Result.success();
}
```

6.2.4 头像管理接口

```
//上传头像
@PostMapping()
public Result uploadAvatar(@RequestParam MultipartFile file){
```

```
    System.out.println("uploadAvatar====>");
    String url = avatarService.upload(file);
    return Result.success(url);
}
//根据文件名下载文件，即文件的 url
@GetMapping("/{fileName}")
public void download(@PathVariable String fileName,
HttpServletResponse response){
    avatarService.download(fileName,response);
}
//根据文件 id 删除文件
@Authority(AuthorityType.requireAuthority)
@DeleteMapping("/{id}")
public Result deleteById(@PathVariable int id){
    int i = avatarService.delete(id);
    if(i == 1){
        return Result.success();
    }else{
        return Result.error(Constants.CODE_500,"删除失败");
    }
}
//查询
@GetMapping("/page")
public Result selectPage(@RequestParam int pageNum,
                         @RequestParam int pageSize){
    int index = (pageNum - 1) * pageSize;
    List<Avatar> avatars = avatarService.selectPage(index, pageSize);
    int total = avatarService.getTotal();
    HashMap<String, Object> map = new HashMap<>();
    map.put("records",avatars);
    map.put("total",total);
    return Result.success(map);
}
```

6.2.5 轮播图管理接口

```
/*
查询
*/
@GetMapping("/{id}")
public Result findById(@PathVariable Long id) {
    return Result.success(carouselService.getById(id));
```

```
}

@GetMapping
public Result findAll() {
    List<Carousel> list = carouselService.getAllCarousel();
    return Result.success(list);
}

/*
保存
*/
@Authority(AuthorityType.requireAuthority)
@PostMapping
public Result save(@RequestBody Carousel carousel) {
    Good good = goodService.getById(carousel.getGoodId());
    if(good == null) {
        return Result.error("400", "商品 id 错误，未查询到商品 id = " +
carousel.getGoodId());
    }
    carouselService.saveOrUpdate(carousel);
    return Result.success();
}
@Authority(AuthorityType.requireAuthority)
@PutMapping
public Result update(@RequestBody Carousel carousel) {
    Good good = goodService.getById(carousel.getGoodId());
    if(good == null) {
        return Result.error("400", "商品 id 错误，未查询到商品 id = " +
carousel.getGoodId());
    }
    carouselService.updateById(carousel);
    return Result.success();
}

/*
删除
*/
@Authority(AuthorityType.requireAuthority)
@DeleteMapping("/{id}")
public Result delete(@PathVariable Long id) {
    carouselService.removeById(id);
    return Result.success();
}
```

6.2.6 购物车管理接口

```
/*
查询
*/
//根据购物车 id 查询
@GetMapping("/{id}")
public Result selectById(@PathVariable Long id) {
    return Result.success(cartService.getById(id));
}
//查找所有用户的购物车
@GetMapping
public Result findAll() {
    List<Cart> list = cartService.list();
    return Result.success(list);
}
//查找某个用户的购物车
@GetMapping("/userid/{userId}")
public Result selectByUserId(@PathVariable Long userId) {
    return Result.success(cartService.selectByUserId(userId)) ;
}

/*
保存
*/
@PostMapping
public Result save(@RequestBody Cart cart) {
    cart.setCreateTime(DateUtil.now());
    cartService.saveOrUpdate(cart);
    return Result.success();
}

@PutMapping
public Result update(@RequestBody Cart cart) {
    cartService.updateById(cart);
    return Result.success();
}

/*
删除
*/
@DeleteMapping("/{id}")
public Result delete(@PathVariable Long id) {
```

```
    cartService.removeById(id);
    return Result.success();
}
```

6.2.7 分类管理接口

```
/*
查询
*/
@GetMapping("/{id}")
public Result findById(@PathVariable Long id) {
    return Result.success(categoryService.getById(id));
}

@GetMapping
public Result findAll() {
    List<Category> list = categoryService.list();
    return Result.success(list);
}

/*
保存
*/
@PostMapping
public Result save(@RequestBody Category category) {
    categoryService.saveOrUpdate(category);
    return Result.success();
}

/**
 *  新增下级分类 + 上下级分类关联
 *
 * @param category 下级分类
 * @return 结果
 */
@PostMapping("/add")
public Map<String, Object> add(@RequestBody Category category) {
    categoryService.add(category);
    return BaseApi.success();
}

@Authority(AuthorityType.requireAuthority)
```

```
@PutMapping
public Result update(@RequestBody Category category) {
    categoryService.updateById(category);
    return Result.success();
}


/**
 * 删除分类
 *
 * @param id id
 * @return 结果
 */
@Authority(AuthorityType.requireAuthority)
@GetMapping("/delete")
public Map<String, Object> delete(@RequestParam("id") Long id) {
    return categoryService.delete(id);
}
```

6.2.8 文件管理接口

```
//上传文件
@PostMapping("/upload")
public Result upload(@RequestParam MultipartFile file){
    String url = fileService.upload(file);
    return Result.success(url);
}

//根据文件名下载文件，即文件的url
@GetMapping("/{fileName}")
public void download(@PathVariable String fileName,
HttpServletResponse response){
    fileService.download(fileName,response);
}
//根据文件id删除文件
@Authority(AuthorityType.requireAuthority)
@DeleteMapping("/{id}")
public Result deleteById(@PathVariable int id){
    int i = fileService.fakeDelete(id);
    if(i == 1){
        return Result.success();
    }else{
```

```
            return Result.error(Constants.CODE_500,"删除失败");
        }
    }
    //批量删除文件
    @Authority(AuthorityType.requireAuthority)
    @PostMapping("/del/batch")
    public Result deleteBatch(@RequestBody List<Integer> ids){
        for (Integer id : ids) {
            int i = fileService.fakeDelete(id);
            if(i != 1){
                return Result.error(Constants.CODE_500,"删除文件：
    "+fileService.getById(id).getName()+"时失败，删除已终止");
            }
        }
        return Result.success();
    }

    @Authority(AuthorityType.requireAuthority)
    @GetMapping("/enable")
    public Result changeEnable(@RequestParam int id,@RequestParam boolean
    enable){
        int i = fileService.changeEnable(id, enable);
        if(i == 0){
            return Result.error(Constants.CODE_500,"修改失败");
        }else {
            return Result.success();
        }

    }
    //查询
    @GetMapping("/page")
    public Result selectPage(@RequestParam int pageNum,
                             @RequestParam int pageSize,
                             @RequestParam(required = false) String fileName){

        IPage<MyFile> myFileIPage = fileService.selectPage(pageNum,
    pageSize, fileName);
        return Result.success(myFileIPage);
    }
```

6.2.9 商品管理接口

```
@Authority(AuthorityType.requireAuthority)
@PostMapping
public Result save(@RequestBody Good good) {
    System.out.println(good);
    return Result.success(goodService.saveOrUpdateGood(good));
}

@Authority(AuthorityType.requireAuthority)
@PutMapping
public Result update(@RequestBody Good good) {
    goodService.update(good);
    return Result.success();
}

@Authority(AuthorityType.requireAuthority)
@DeleteMapping("/{id}")
public Result delete(@PathVariable Long id) {
    goodService.deleteGood(id);
    return Result.success();
}

@GetMapping("/{id}")
public Result findById(@PathVariable Long id) {
    return Result.success(goodService.getGoodById(id));
}

//获取商品的规格信息
@GetMapping("/standard/{id}")
public Result getStandard(@PathVariable int id) {
    return Result.success(goodService.getStandard(id));
}
//查询推荐商品，即 recommend=1
@GetMapping
public Result findAll() {

    return Result.success(goodService.findFrontGoods());
}
//查询销量排行
@GetMapping("/rank")
public Result getSaleRank(@RequestParam int num){
    return Result.success(goodService.getSaleRank(num));
}
//保存商品的规格信息
@PostMapping("/standard")
```

```
public Result saveStandard(@RequestBody List<Standard> standards,
@RequestParam int goodId) {
    //先删除全部旧记录
    standardService.deleteAll(goodId);
    //然后插入新记录
    for (Standard standard : standards) {
        standard.setGoodId(goodId);
        if(!standardService.save(standard)){
            return Result.error(Constants.CODE_500,"保存失败");
        }
    }
    return Result.success();
}

//删除商品的规格信息
@Authority(AuthorityType.requireAuthority)
@DeleteMapping("/standard")
public Result delStandard(@RequestBody Standard standard) {
    boolean delete = standardService.delete(standard);
    if(delete) {
        return Result.success();
    }else {
        return Result.error(Constants.CODE_500,"删除失败");
    }
}

//修改商品的推荐字段
@Authority(AuthorityType.requireAuthority)
@GetMapping("/recommend")
public Result setRecommend(@RequestParam Long id,@RequestParam
Boolean isRecommend){
    return Result.success(goodService.setRecommend(id,isRecommend));
}

@GetMapping("/page")
public Result findPage(
                    @RequestParam(required = false, defaultValue =
"1") Integer pageNum,
                    @RequestParam(required = false, defaultValue =
"10") Integer pageSize,
                    @RequestParam(required = false, defaultValue =
"") String searchText,
                    @RequestParam(required = false) Integer
categoryId) {
```

```
    return
Result.success(goodService.findPage(pageNum,pageSize,searchText,categ
oryId));
}
@GetMapping("/fullPage")
public Result findFullPage(
       @RequestParam(required = false, defaultValue = "1") Integer
pageNum,
       @RequestParam(required = false, defaultValue = "10") Integer
pageSize,
       @RequestParam(required = false, defaultValue = "") String
searchText,
       @RequestParam(required = false) Integer categoryId) {

    return
Result.success(goodService.findFullPage(pageNum,pageSize,searchText,c
ategoryId));
}
```

6.2.10 图标管理接口

```
/*
查询
*/
@GetMapping("/{id}")
public Result findById(@PathVariable Long id) {
    return Result.success(iconService.getById(id));
}

@GetMapping
public Result findAll() {
    List<Icon> list = iconService.getIconCategoryMapList();
    return Result.success(list);
}


/*
保存
*/
@Authority(AuthorityType.requireAuthority)
@PostMapping
```

```
public Result save(@RequestBody Icon icon) {
    iconService.saveOrUpdate(icon);
    return Result.success();
}

@Authority(AuthorityType.requireAuthority)
@PutMapping
public Result update(@RequestBody Icon icon) {
    iconService.updateById(icon);
    return Result.success();
}

/*
 *删除
 */
@Authority(AuthorityType.requireAuthority)
@GetMapping("/delete")
public Map<String, Object> delete(@RequestParam("id") Long id) {
    return iconService.deleteById(id);
}
```

6.2.11 收入分析管理接口

```
@GetMapping("/chart")
public Result getChart(){
    return Result.success(incomeService.getChart());
}
@GetMapping("/week")
public Result getWeekIncome(){
    return Result.success(incomeService.getWeekIncome());
}

@GetMapping("/month")
public Result getMonthIncome(){
    return Result.success(incomeService.getMonthIncome());
}
```

6.2.12 订单管理接口

```
/*
查询
```

```
*/
@GetMapping("/userid/{userid}")
public Result selectByUserId(@PathVariable int userid) {
    return Result.success(orderService.selectByUserId(userid));
}
@GetMapping("/orderNo/{orderNo}")
public Result selectByOrderNo(@PathVariable String orderNo) {
    return Result.success(orderService.selectByOrderNo(orderNo));
}
@GetMapping
public Result findAll() {
    List<Order> list = orderService.list();
    return Result.success(list);
}

/*
分页查询
*/
@GetMapping("/page")
public Result findPage(@RequestParam int pageNum,
                       @RequestParam int pageSize,
                       String orderNo,String state){
    IPage<Order> orderPage = new Page<>(pageNum,pageSize);
    QueryWrapper<Order> orderQueryWrapper = new QueryWrapper<>();
    orderQueryWrapper.ne("state","待付款");
    if(!Util.isEmptyString(state)){
        orderQueryWrapper.eq("state",state);
    }
    if(!Util.isEmptyString(orderNo)){
        orderQueryWrapper.like("order_no",orderNo);
    }

    orderQueryWrapper.orderByDesc("create_time");
    return
Result.success(orderService.page(orderPage,orderQueryWrapper));
}
/*
保存
*/
@PostMapping
public Result save(@RequestBody Order order) {
    String orderNo = orderService.saveOrder(order);
    return Result.success(orderNo);
```

```
}
//支付订单
@GetMapping("/paid/{orderNo}")
public Result payOrder(@PathVariable String orderNo){
    orderService.payOrder(orderNo);
    return Result.success();
}
//发货
@Authority(AuthorityType.requireAuthority)
@GetMapping("/delivery/{orderNo}")
public Result delivery(@PathVariable String orderNo){
    orderService.delivery(orderNo);
    return Result.success();
}
//确认收货
@GetMapping("/received/{orderNo}")
public Result receiveOrder(@PathVariable String orderNo){
    if(orderService.receiveOrder(orderNo)){
        return Result.success();
    }
    else {
        return Result.error(Constants.CODE_500,"确认收货失败");
    }
}

@PutMapping
public Result update(@RequestBody Order order) {
    orderService.updateById(order);
    return Result.success();
}

/*
删除
*/
@DeleteMapping("/{id}")
public Result delete(@PathVariable Long id) {
    orderService.removeById(id);
    return Result.success();
}
```

6.2.13 角色管理接口

```
@PostMapping("/role")
public Result getUserRole(){
    User currentUser = TokenUtils.getCurrentUser();
    return Result.success(currentUser.getRole());
}
```

6.2.14 用户管理接口

```
@GetMapping("/userinfo/{username}")
public Result getUserInfoByName(@PathVariable String username) {
    User one = userService.getOne(username);
    return Result.success(one);
}

@GetMapping("/userid")
public long getUserId() {
    return TokenUtils.getCurrentUser().getId();
}

@GetMapping("/user/")
public Result findAll() {
    List<User> list = userService.list();
    return Result.success(list);
}

@PostMapping("/user")
public Result save(@RequestBody User user) {

    return userService.saveUpdate(user);
}

@Authority(AuthorityType.requireAuthority)
@DeleteMapping("/user/{id}")
public Result deleteById(@PathVariable int id) {
    boolean isSuccessful = userService.removeById(id);
    if (isSuccessful) {
        return Result.success();
    } else {
        return Result.error(Constants.CODE_500, "删除失败");
    }
}
```

```
@Authority(AuthorityType.requireAuthority)
@PostMapping("/user/del/batch")
public Result deleteBatch(@RequestBody List<Integer> ids) {
    boolean isSuccessful = userService.removeBatchByIds(ids);
    if (isSuccessful) {
        return Result.success();
    } else {
        return Result.error(Constants.CODE_500, "删除失败");
    }
}

@GetMapping("/user/page")
public Result findPage(@RequestParam int pageNum,
                       @RequestParam int pageSize,
                       String id,
                       String username,
                       String nickname) {
    IPage<User> userPage = new Page<>(pageNum, pageSize);
    QueryWrapper<User> userQueryWrapper = new QueryWrapper<>();
    if (!Util.isEmptyString(id)) {
        userQueryWrapper.like("id", id);
    }
    if (!Util.isEmptyString(username)) {
        userQueryWrapper.like("username", username);
    }
    if (!Util.isEmptyString(nickname)) {
        userQueryWrapper.like("nickname", nickname);
    }
    userQueryWrapper.orderByDesc("id");
    System.out.println("============" + TokenUtils.getCurrentUser());
    return Result.success(userService.page(userPage,
userQueryWrapper));
}

/**
 * 重置密码
 *
 * @param id          用户id
 * @param newPassword 新密码
 * @return 结果
 */
@GetMapping("/user/resetPassword")
public Result resetPassword(@RequestParam String id, @RequestParam
String newPassword) {
```

```
    userService.resetPassword(id, newPassword);
    return Result.success();
}
```

7 性能

7.1 精度

要按照严格的数据格式输入，不能输入非法字符，否则系统不给予响应进行处理，查询时要保证准确率为 100%，所有包含查询关键字的书籍都应能查到，不能有遗漏。

7.2 时间特性

（1）响应时间：用户任意操作后 5 秒内系统给予反馈信息。
（2）更新处理时间：由系统运行状态来决定。
（3）数据的转换和传送时间：能够在 20 秒内完成。

7.3 灵活性

当需求发生某些变化时，该软件的基本操作、数据结构、运行环境等等基本不会发生变化，只是对系统的数据库的文件和记录进行处理，就可以满足需求。

8 测试

功能点	测试用例	输出结果
商品管理	添加商品成功	商品成功添加到商品列表
商品管理	编辑商品信息成功	商品信息更新成功
商品管理	删除商品成功	商品成功从商品列表中删除

功能点	测试用例	输出结果
商品管理	删除商品失败（该商品已被订单引用）	提示该商品已被订单引用，无法删除
商品分类	添加分类成功	分类成功添加到分类列表
商品分类	添加分类失败（分类名重复）	提示分类名已存在
商品分类	编辑分类信息成功	分类信息更新成功
商品分类	编辑分类信息失败（分类名重复）	提示分类名已存在
商品分类	删除分类成功	分类成功从分类列表中删除
商品分类	删除分类失败（该分类下有商品）	提示该分类下有商品，无法删除
订单管理	查看订单详细信息	成功显示订单的详细信息
用户管理	添加管理员成功	管理员成功添加到管理员列表
用户管理	添加管理员失败（用户名已存在）	提示用户名已存在
用户管理	编辑管理员信息成功	管理员信息更新成功
用户管理	编辑管理员信息失败（用户名已存在）	提示用户名已存在
用户管理	删除管理员成功	管理员成功从管理员列表中删除

功能点	测试用例	输出结果
用户管理	删除管理员失败（该管理员是唯一管理员）	提示该管理员是唯一管理员，无法删除
用户管理	添加普通买家用户成功	用户成功添加到用户列表
用户管理	添加普通买家用户失败（用户名已存在）	提示用户名已存在
用户管理	编辑普通买家用户信息成功	用户信息更新成功
用户管理	删除普通买家用户成功	用户成功从用户列表中删除
用户管理	查看用户购物车商品	成功显示用户购物车商品
购物车管理	添加商品到购物车成功	商品成功添加到购物车
购物车管理	添加商品到购物车失败（该商品已经在购物车中）	提示该商品已经在购物车中
购物车管理	从购物车删除商品成功	商品成功从购物车中删除
购物车管理	从购物车删除商品失败（该商品不在购物车中）	提示该商品不在购物车中