目录

[1 引言 3](#_Toc170206765)

[1.1 编写目的 3](#_Toc170206766)

[1.2 项目背景 3](#_Toc170206767)

[2 总体设计 3](#_Toc170206768)

[2.1 需求概述 3](#_Toc170206769)

[2.2 软件结构 4](#_Toc170206770)

[3 模块设计 4](#_Toc170206771)

[3.1 模块基本信息 4](#_Toc170206772)

[3.1.1 商品管理模块 4](#_Toc170206773)

[3.1.2 商品分类模块 5](#_Toc170206774)

[3.1.3 订单管理模块 5](#_Toc170206775)

[3.1.4用户管理模块 5](#_Toc170206776)

[3.1.5 购物车管理模块 5](#_Toc170206777)

[3.2 模块处理逻辑 5](#_Toc170206778)

[4 UML建模 8](#_Toc170206779)

[4.1 类图 8](#_Toc170206780)

[4.2 时序图 9](#_Toc170206781)

[4.3 用例图 9](#_Toc170206782)

[4.4 活动图 10](#_Toc170206783)

[4.5 组件图 10](#_Toc170206784)

[4.6 部署图 10](#_Toc170206785)

[5 数据库设计 11](#_Toc170206786)

[5.1 ER图表 11](#_Toc170206787)

[5.2 表设计 12](#_Toc170206788)

[5.2.1 地址表 12](#_Toc170206789)

[5.2.2 头像表 12](#_Toc170206790)

[5.2.3 轮播图表 13](#_Toc170206791)

[5.2.4 购物车表 13](#_Toc170206792)

[5.2.5 分类表 14](#_Toc170206793)

[5.2.6 商品表 14](#_Toc170206794)

[5.2.7 商品规格关联表 15](#_Toc170206795)

[5.2.8 图标表 16](#_Toc170206796)

[5.2.9 图标分类关联表 16](#_Toc170206797)

[5.2.10 订单商品关联表 16](#_Toc170206798)

[5.2.11 规格表 17](#_Toc170206799)

[5.2.12 系统文件表 17](#_Toc170206800)

[5.2.13 用户表 18](#_Toc170206801)

[5.2.14 订单表 19](#_Toc170206802)

[6 接口设计 20](#_Toc170206803)

[6.1 外部接口 20](#_Toc170206804)

[6.1.1 登录界面 20](#_Toc170206805)

[6.1.2 注册界面 21](#_Toc170206806)

[6.1.3 商城首页 22](#_Toc170206807)

[6.1.4 个人信息界面 23](#_Toc170206808)

[6.1.5 购物车界面 25](#_Toc170206809)

[6.1.6 地址信息界面 26](#_Toc170206810)

[6.1.7 订单界面 27](#_Toc170206811)

[6.1.8 管理员界面 29](#_Toc170206812)

[6.1.9 轮播图管理界面 30](#_Toc170206813)

[6.1.10 订单管理界面 31](#_Toc170206814)

[6.1.11 商品分类管理界面 33](#_Toc170206815)

[6.1.12 用户管理界面 35](#_Toc170206816)

[6.1.13 统计界面 37](#_Toc170206817)

[6.2 内部接口 38](#_Toc170206818)

[6.2.1 登录接口 38](#_Toc170206819)

[6.2.2 注册接口 38](#_Toc170206820)

[6.2.3 地址管理接口 38](#_Toc170206821)

[6.2.4 头像管理接口 40](#_Toc170206822)

[6.2.5 轮播图管理接口 41](#_Toc170206823)

[6.2.6 购物车管理接口 42](#_Toc170206824)

[6.2.7 分类管理接口 43](#_Toc170206825)

[6.2.8 文件管理接口 44](#_Toc170206826)

[6.2.9 商品管理接口 46](#_Toc170206827)

[6.2.10 图标管理接口 48](#_Toc170206828)

[6.2.11 收入分析管理接口 49](#_Toc170206829)

[6.2.12 订单管理接口 50](#_Toc170206830)

[6.2.13 角色管理接口 52](#_Toc170206831)

[6.2.14 用户管理接口 52](#_Toc170206832)

[7 性能 54](#_Toc170206833)

[7.1 精度 54](#_Toc170206834)

[7.2 时间特性 54](#_Toc170206835)

[7.3 灵活性 54](#_Toc170206836)

[8 测试 54](#_Toc170206837)

# 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 git版本库

我组创建了该项目的git版本库并上传至了github以下为网页：

**https://github.com/zx19834055688/2024Decalon.git**

# 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。

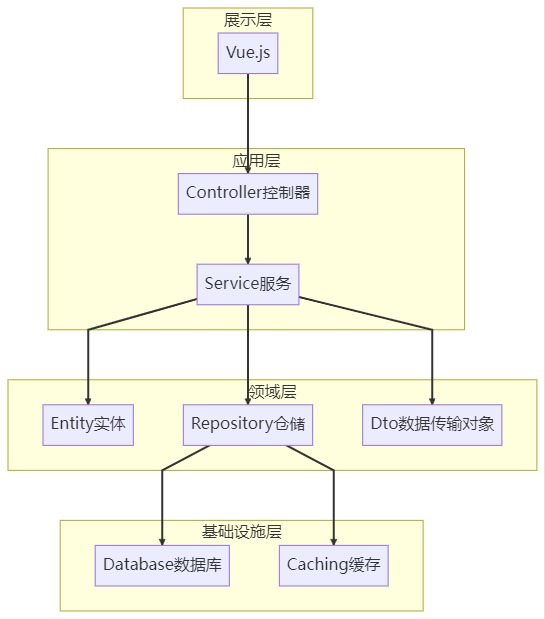
（2）商品分类模块：实现了分类的增删改查、分类层级管理、商品分类的关联等功能。

（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。

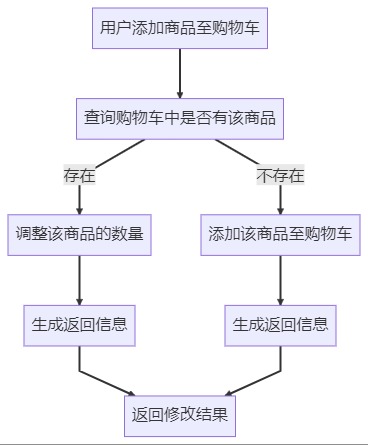
### 3.1.4用户管理模块

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。

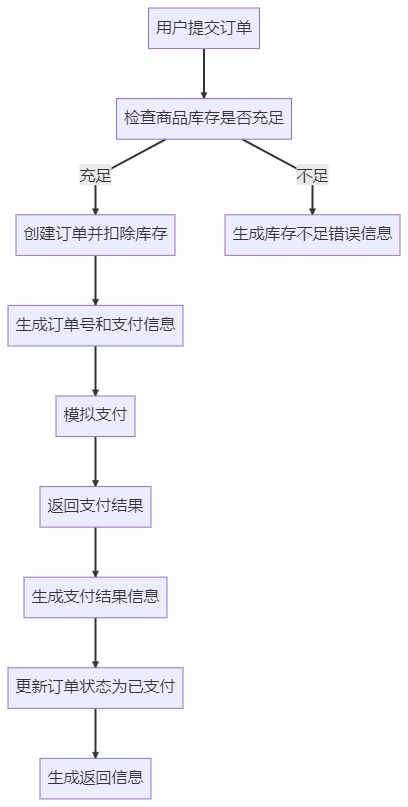
### 3.1.5 购物车管理模块

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。

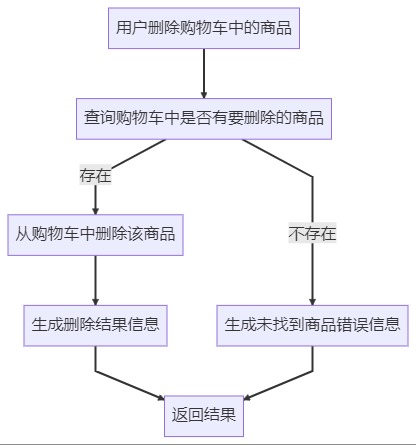
## 3.2 模块处理逻辑



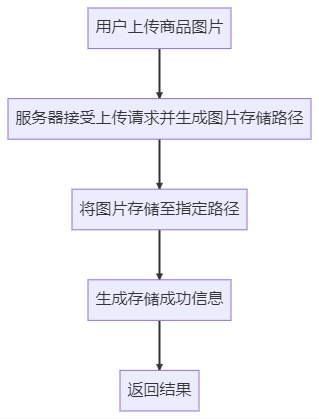
以上流程图是用户在购物车界面添加商品的操作流程。首先查询该商品是否已经在购物车中，若已存在，则调整该商品的数量，否则添加该商品至购物车。最后返回添加或调整的结果信息。这个流程设计简洁，易于理解。



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。



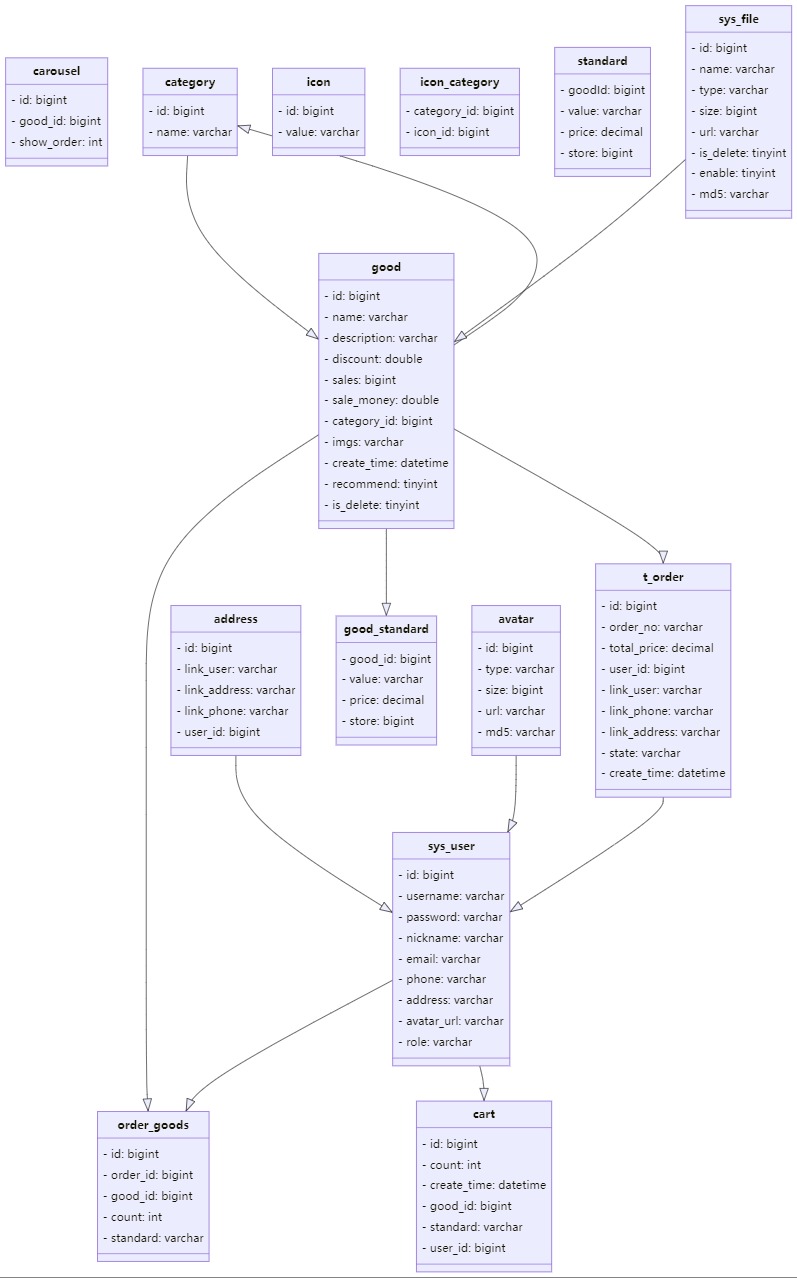
以上流程图是用户在购物车中删除商品的操作流程。首先查询该商品是否在购物车中，若存在则从购物车中删除该商品，否则返回未找到商品错误信息。最后生成删除结果信息并返回给用户。这个流程设计简洁明了，有效实现了购物车中删除商品的功能。



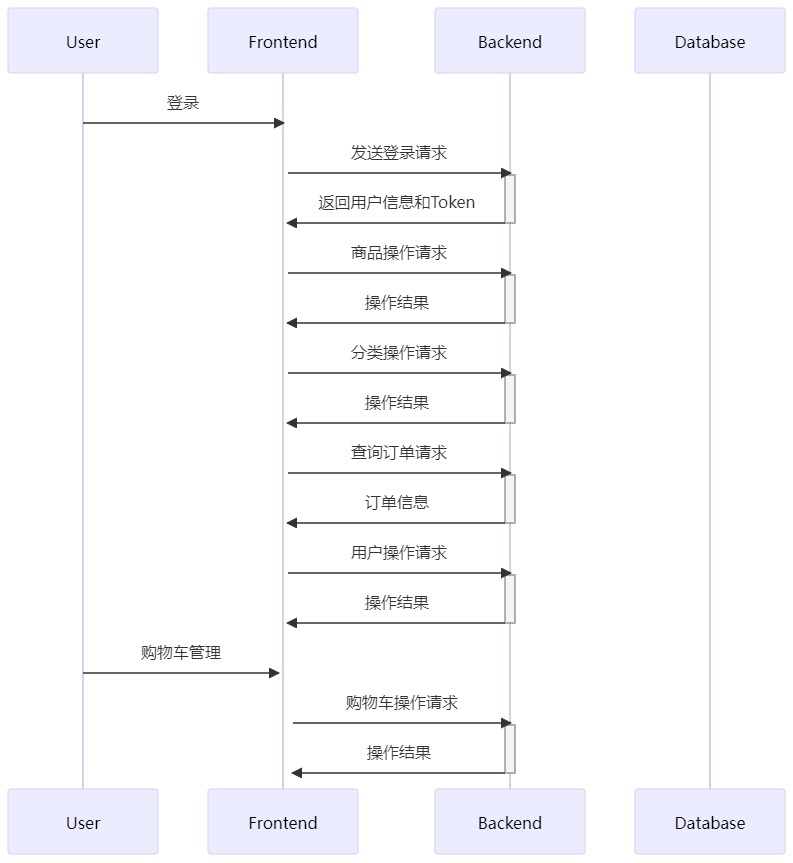
以上流程图是用户上传商品图片的操作流程。用户上传图片后，服务器接受到上传请求并根据图片大小和格式生成存储路径。然后将图片存储至指定路径，并生成存储成功信息返回给用户。这个流程设计简单清晰，有利于实现图片上传的功能。

# 4 UML建模

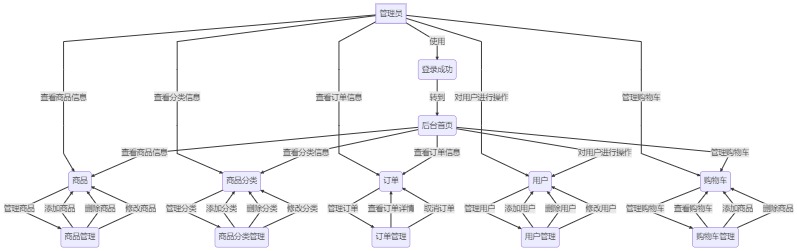
## 4.1 类图



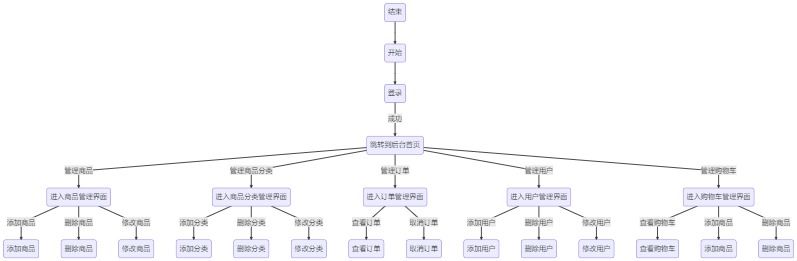
## 4.2 时序图



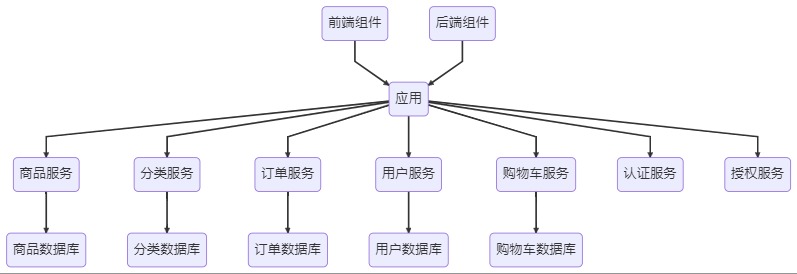
## 4.3 用例图



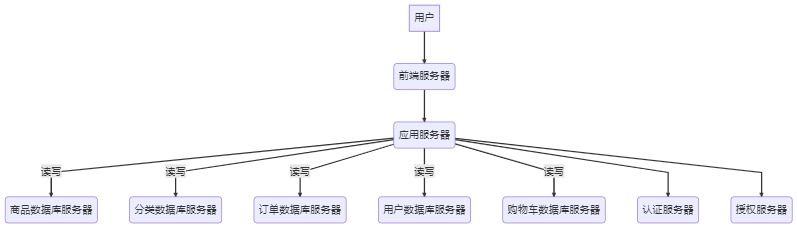
## 4.4 活动图



## 4.5 组件图

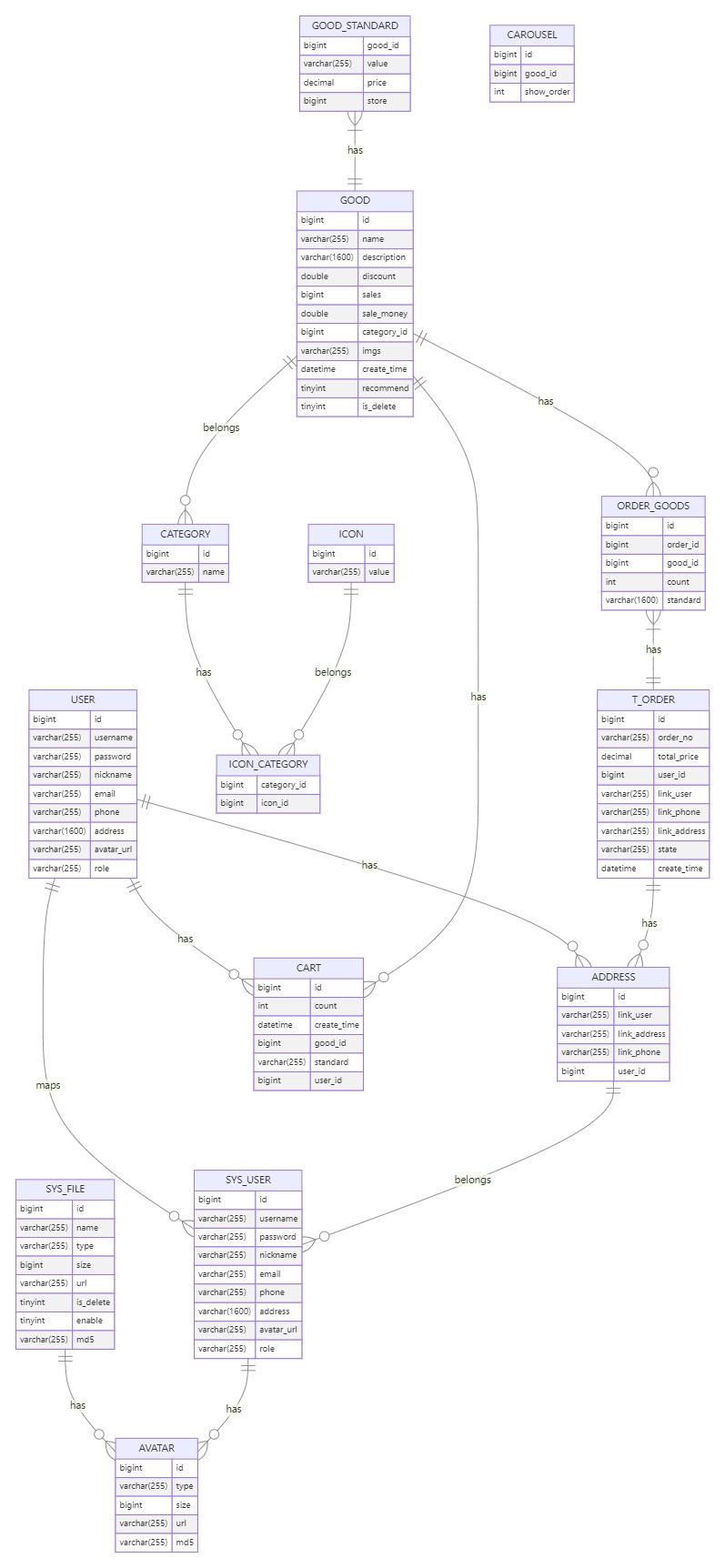


## 4.6 部署图



# 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 |
| 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 |
| 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 |
| 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 |
| 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 |
| 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>

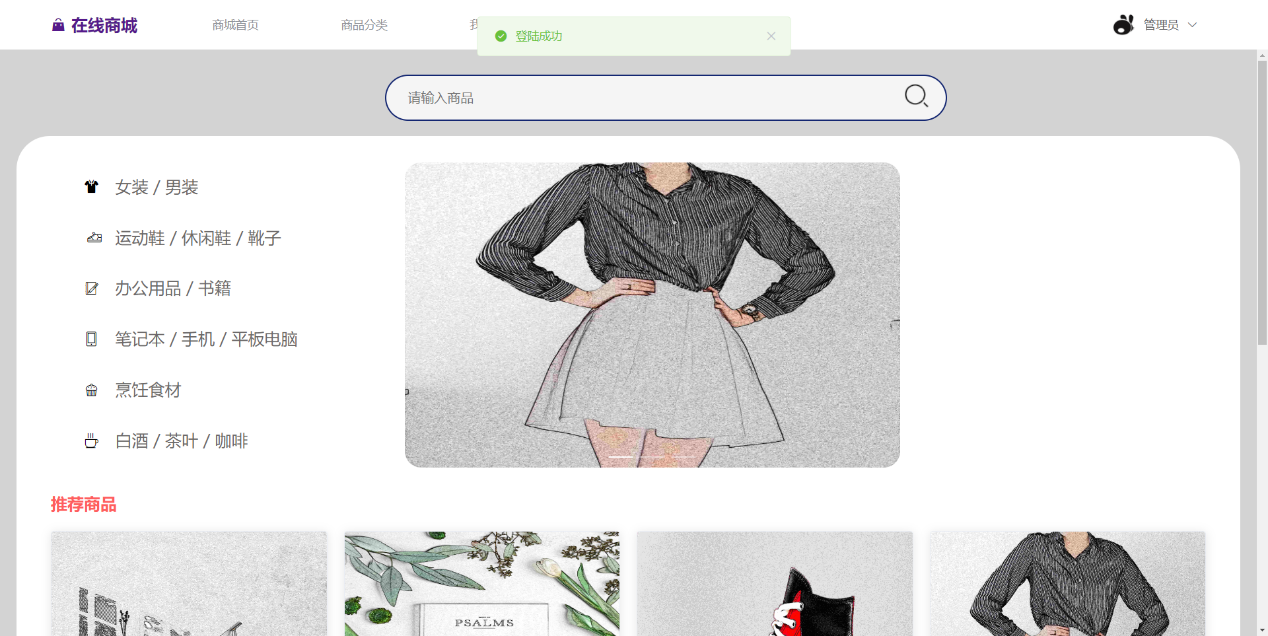
            <el-button @click="$router.push('/login')">返回</el-button>

          </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>

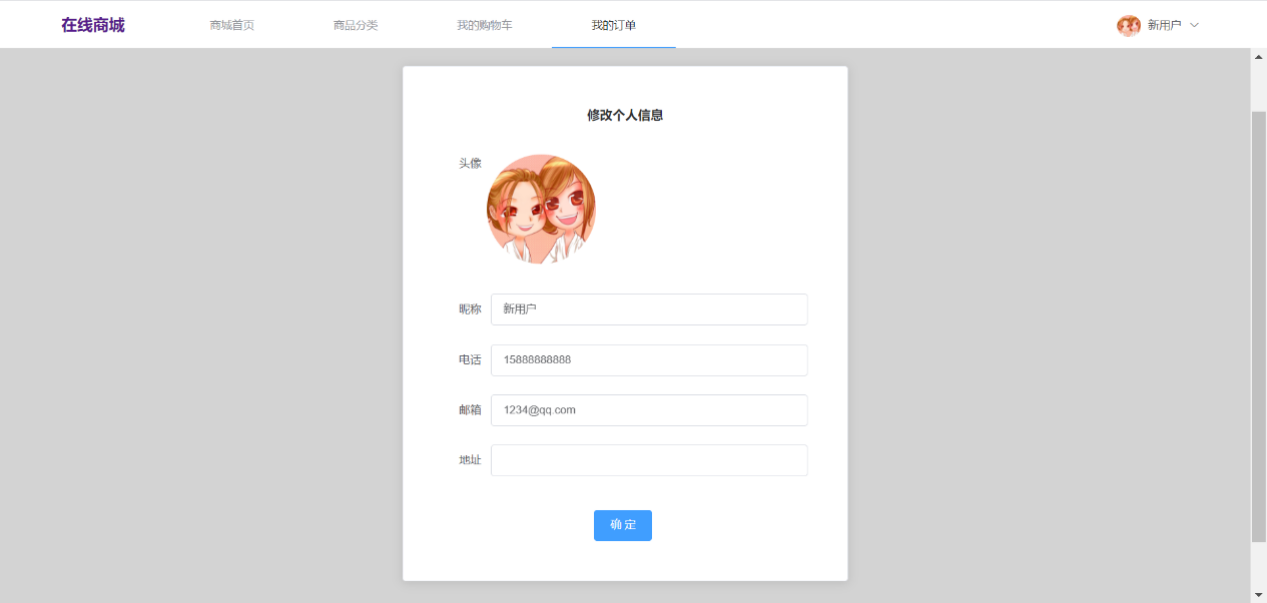
    <el-main style="background-color: lightgrey;width:100%;">

      <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: '' }"

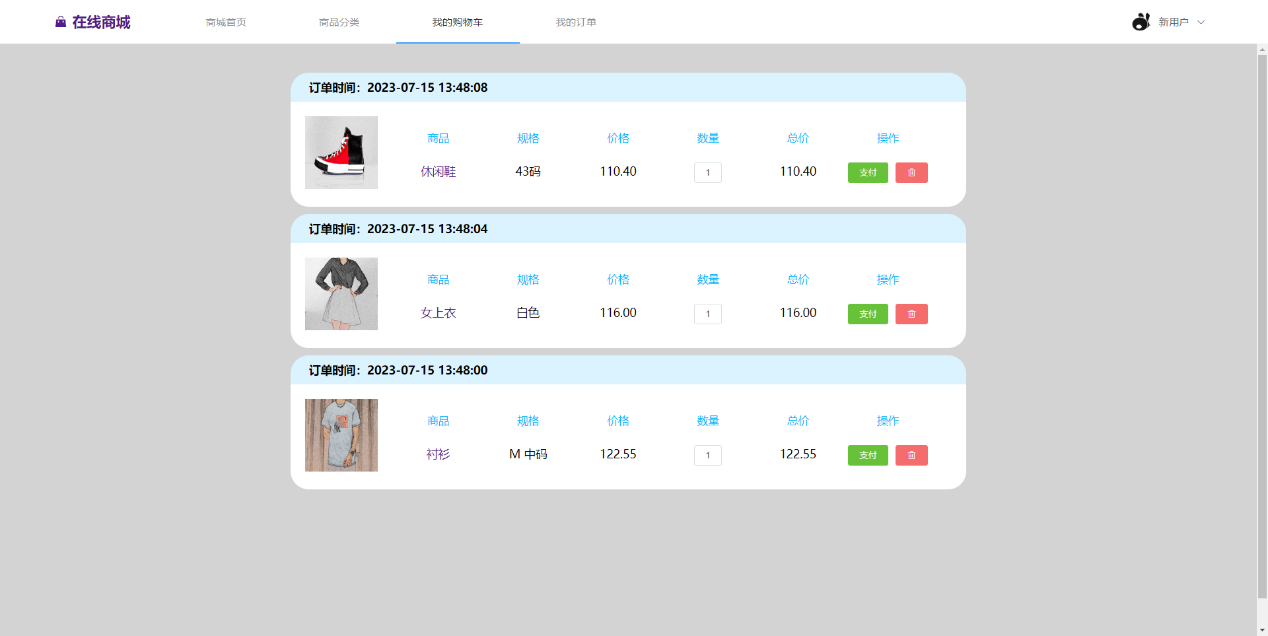
        >重置密码</el-button

      >

    </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>

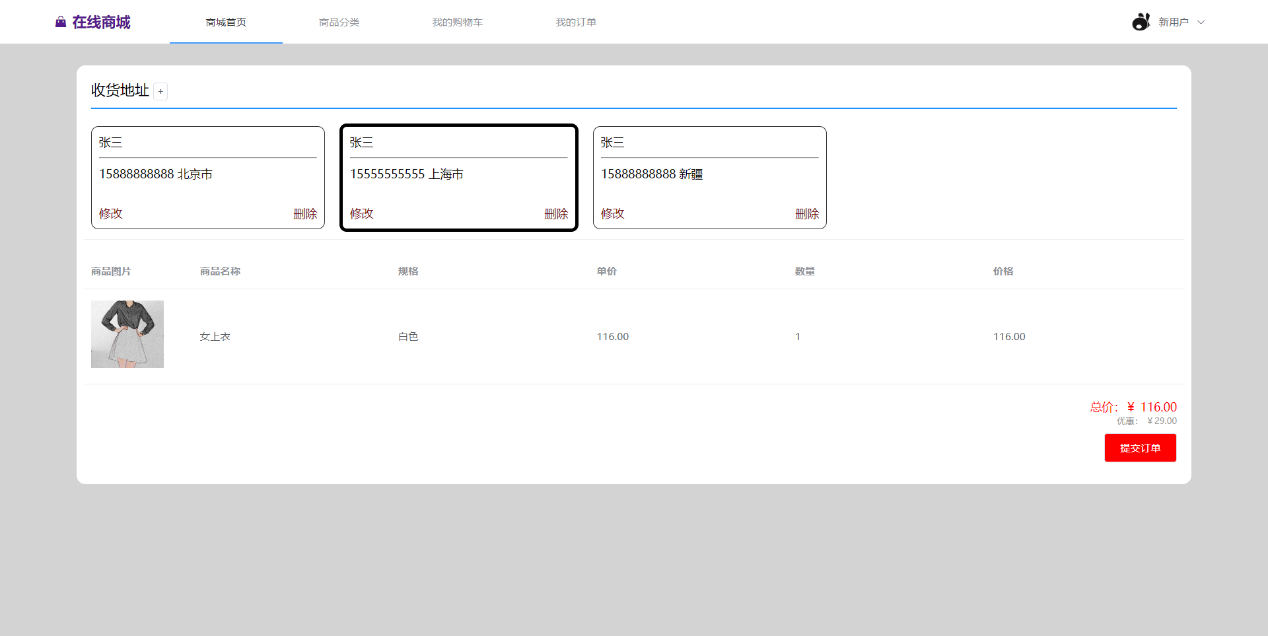
  <template v-for="cart in carts">

    <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">

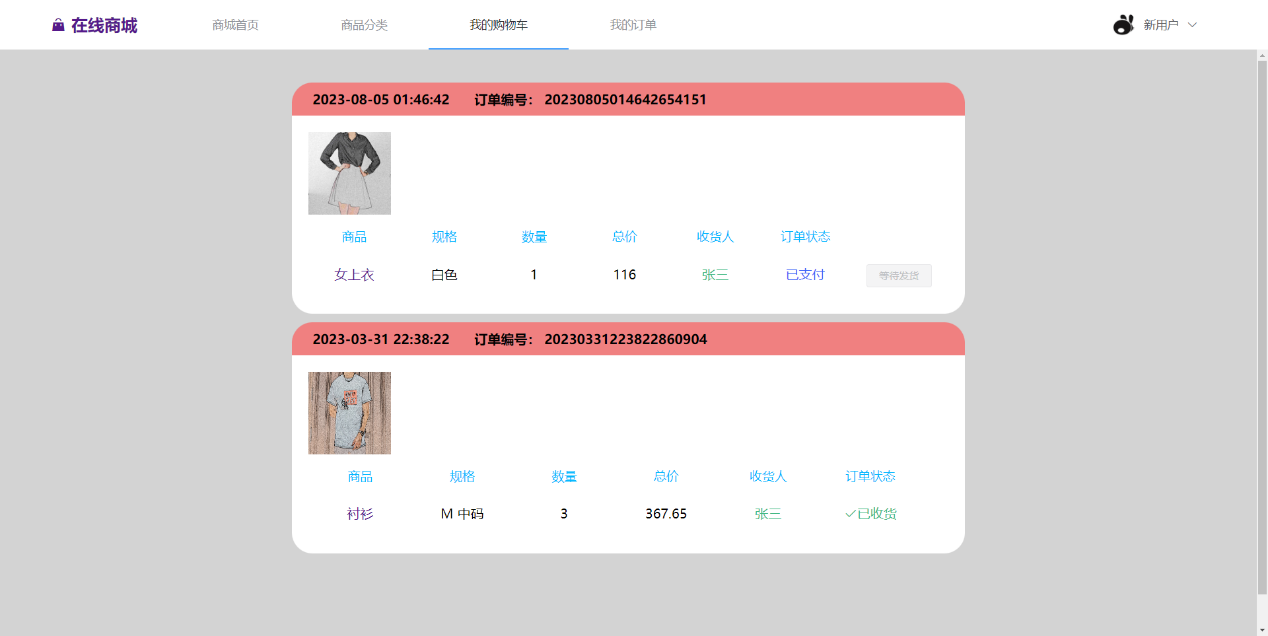
          <el-button @click="dialogFormVisible = false">取 消</el-button>

          <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" />

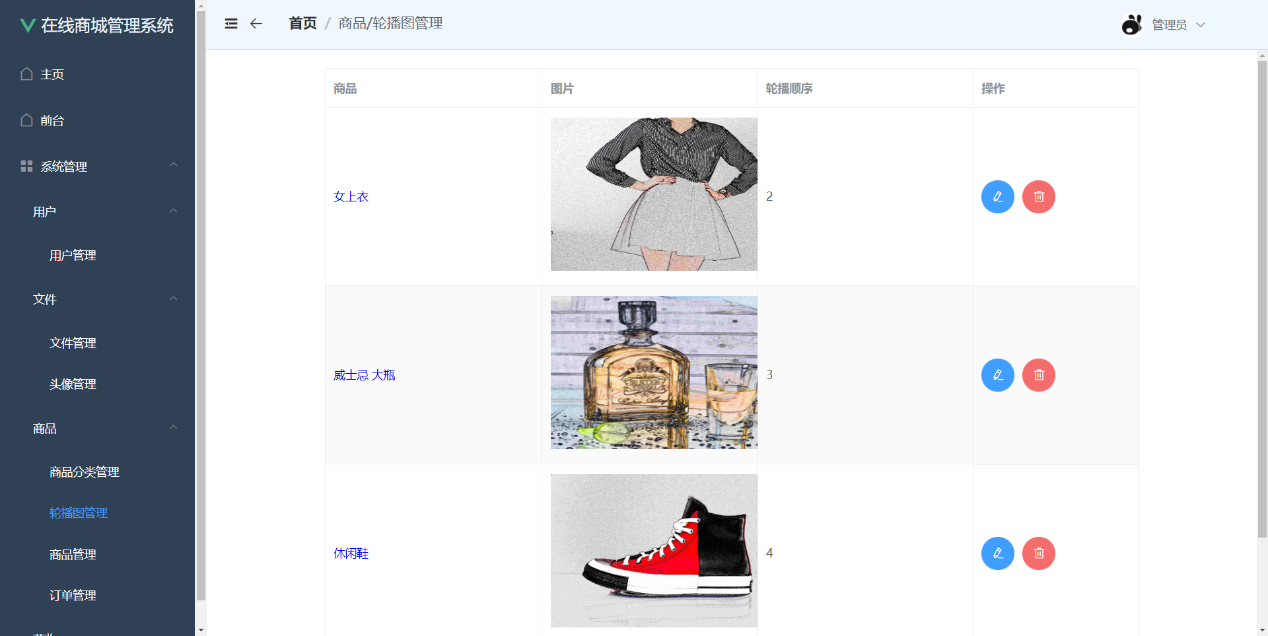
        </el-main>

      </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.goodName}}</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>

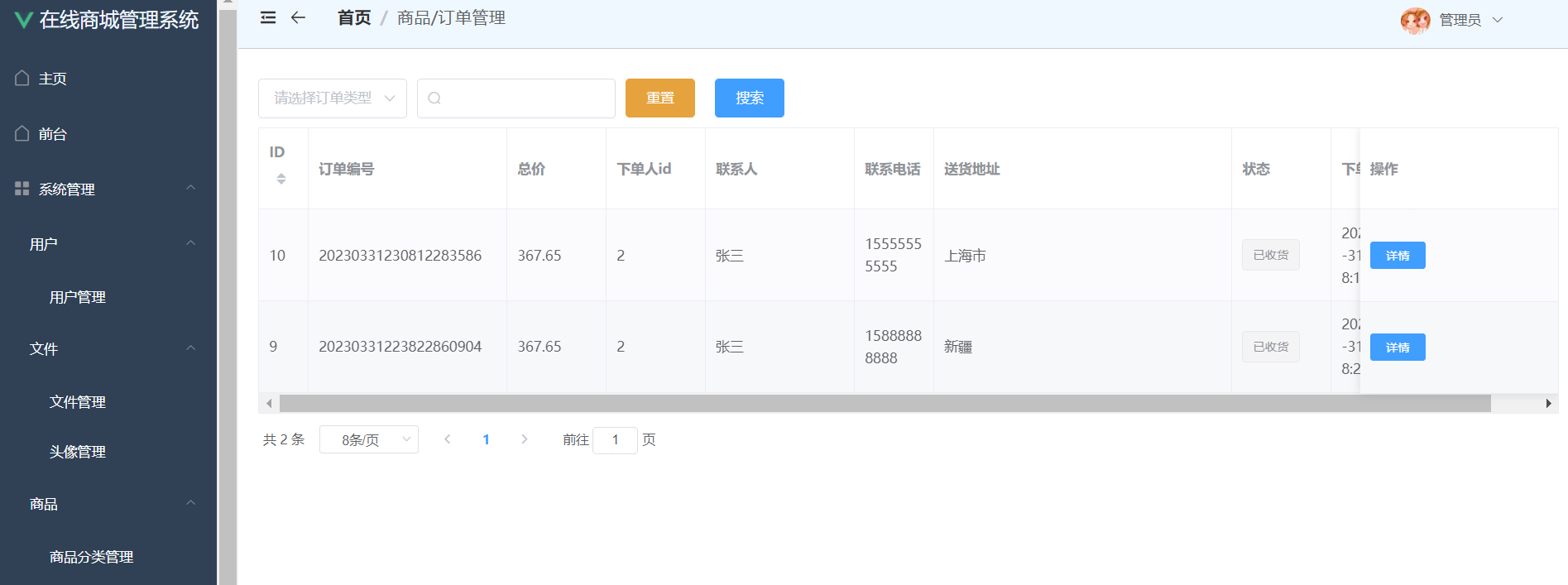
          </template>

        </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>

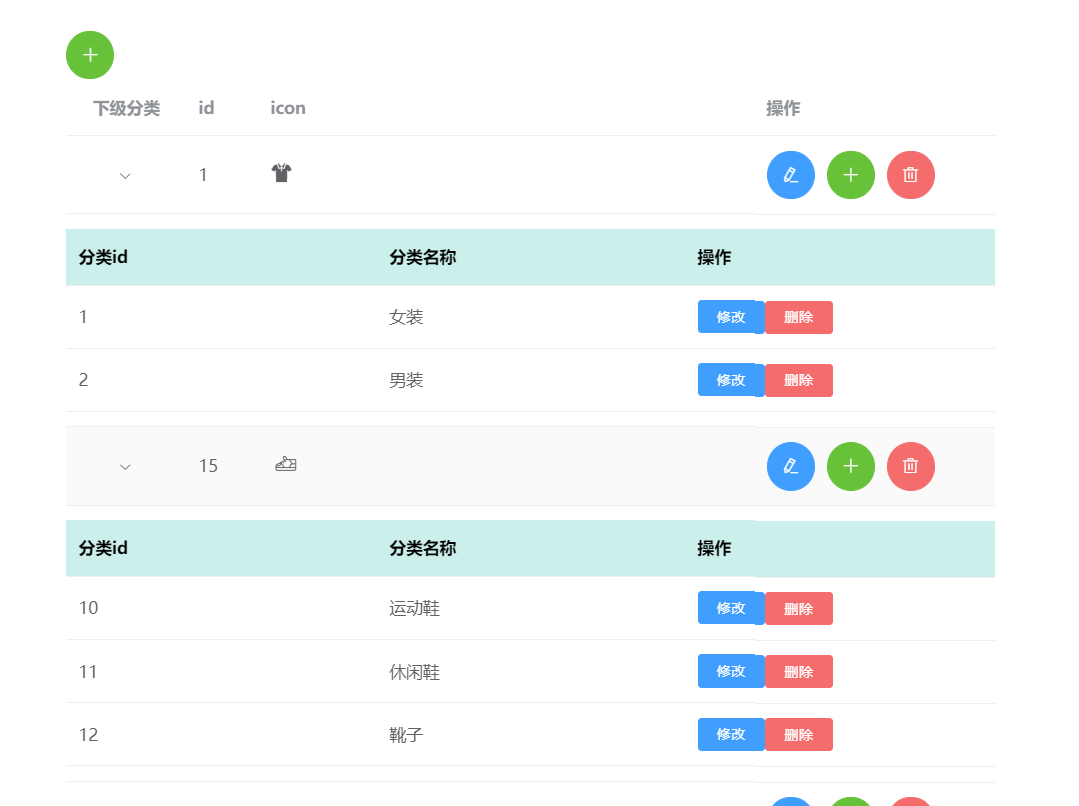
          </el-popconfirm>

        </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>

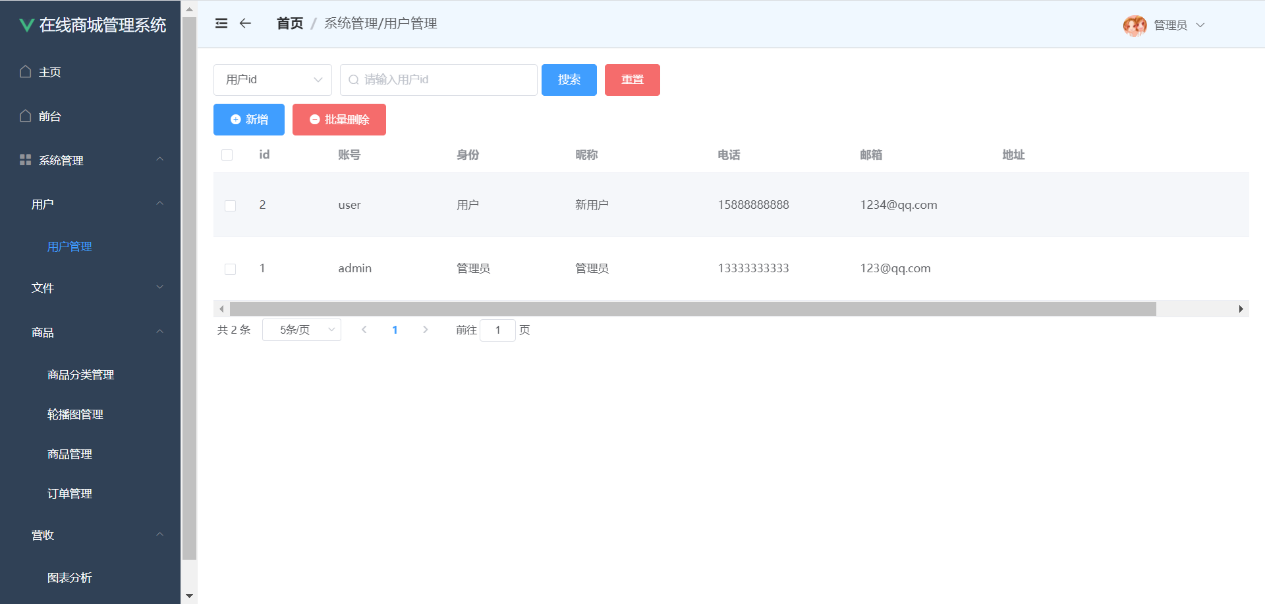
            </el-popconfirm>

          </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,categoryId));  
}  
@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,categoryId));  
}

### 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 |
| 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 |
| 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 |