|  |  |
| --- | --- |
|  | JD |
| **JIANGSU UNIVERSITY** |

软件开发技术课程设计(II)

电子商城系统

专 业 班 级 ： 软件工程2102

学 号 ： 3210608047

学 生 姓 名 ： 许志恒

指 导 教 师 ： 林 琳

2023年 6 月 26日

目录

[第一章 概 述 2](#_Toc32339)

[1.1 课程设计目的 2](#_Toc1176)

[1.2 课程设计任务 2](#_Toc22397)

[1.3 使用技术及开发环境 3](#_Toc17446)

[1.3.1采用的开发技术 3](#_Toc19156)

[1.3.2使用框架 4](#_Toc6845)

[1.3.3开发工具 4](#_Toc24921)

[第二章 需求分析 4](#_Toc1005)

[2.1 功能分析 4](#_Toc11429)

[2.1.1用户注册登录模块 5](#_Toc26128)

[2.1.2订单管理查询模块 5](#_Toc12359)

[2.1.3管理员分类管理模块 6](#_Toc31402)

[2.1.4层次功能图设计 7](#_Toc15134)

[2.2 概念模型分析 7](#_Toc2783)

[2.2.1用户实体 7](#_Toc22640)

[2.2.2管理员实体 8](#_Toc10415)

[2.2.3商品实体 9](#_Toc32462)

[2.2.4总体e-r图 9](#_Toc4808)

[第三章 系统设计 9](#_Toc4872)

[3.1 数据库设计 9](#_Toc8239)

[3.1.1总表： 10](#_Toc18357)

[3.1.2管理员表（t\_admin）表 10](#_Toc14878)

[3.1.3商品（t\_goods）表 10](#_Toc26149)

[3.1.4购物车（t\_shopcar）表 11](#_Toc23993)

[3.1.5用户（t\_user）表 11](#_Toc13235)

[3.2 模块设计 12](#_Toc11407)

[3.2.1商城业务流程 12](#_Toc32465)

[3.2.2订单后台管理流程 12](#_Toc2560)

[3.2.3用户管理流程 13](#_Toc31041)

[第四章 系统实现 14](#_Toc29955)

[4.1 项目介绍 14](#_Toc29021)

[4.1.1idea文件夹： 14](#_Toc2269)

[4.1.2 build文件夹： 18](#_Toc19039)

[4.1.3. config文件夹： 19](#_Toc9414)

[4.1.4. node\_modules文件夹： 20](#_Toc21497)

[4.1.5. src文件夹： 20](#_Toc31127)

[4.1.6 static文件夹： 68](#_Toc25549)

[4.2 系统功能实现 69](#_Toc30186)

[4.2.1首页展示： 69](#_Toc26498)

[4.2.2注册登录功能展示： 71](#_Toc3169)

[4.2.3管理员功能展示： 74](#_Toc7294)

[4.2.4用户功能展示 80](#_Toc27879)

[第五章 总结 85](#_Toc17051)

第一章 概 述

## 1.1 课程设计目的

通过本课程设计的准备与总结，复习、领会、巩固和运用课堂上所学的软件开发方法和知识，为学生综合应用本专业所学习的多门课程知识创造实践机会，使每个学生了解软件工具与环境对于项目开发的重要性，并且重点深入掌握好几种较新或较流行的软件工具或计算机应用技术，提高学生今后参与开发稍大规模实际软件项目和探索未知领域的能力和自信心。

## 1.2 课程设计任务

假设某企业需要大学生创建一个电子商城系统（任选一个主题：手机、电器、图书、服装等），以便能够提高销售业绩，系统基本功能有：

1. 首页为商品展示页面，不需要登录。
2. 使用系统的用户包括一个管理员和多个用户，需要登陆才能操作系统，登录后可以修改个人信息。
3. 管理员可以管理商品（包括编号、名称、照片、价格等）、浏览用户、查看订单。
4. 用户可以注册（包括编号、姓名、照片、性别、地址等）。可以查询商品，可以查看商品详细信息。可以将商品加入购物车，可以进行结算（不用真实付款）。可以查看自己的订单。
5. 管理员统计：按消费金额倒排的十强用户列表，按销售量倒排序的十强商品列表。统计结果以柱状图或其他图形展示。
6. 选做：统计结果可以导出为excel文件。

## 1.3 使用技术及开发环境

1.3.1采用的开发技术

在该电子商城系统的开发中，我们选择了Vue2和Flask作为主要的开发技术。Vue2作为一种流行的JavaScript框架，为我们提供了构建动态用户界面的强大工具和组件。它具备响应式数据绑定、组件化开发和虚拟DOM等特性，使得开发单页应用程序变得更加简单高效。同时，Flask作为一种轻量级的Python Web框架，为我们提供了简洁灵活的设计，适用于构建中小型的Web应用程序和API。它提供了路由、请求处理和模板引擎等功能，并与数据库的集成相对便捷，使得我们能够快速开发出高性能的后端服务。

1.3.2使用框架

为了更好地实现该电子商城系统的功能，我们采用了Vue2和Flask作为开发框架。Vue2作为前端框架，帮助我们构建用户界面，实现了首页商品展示、用户登录、个人信息修改、商品查询、购物车和订单等功能。通过Vue Router进行前端路由管理，我们能够实现页面之间的导航和跳转。同时，使用Vue的组件化开发方式，我们将页面拆分为多个可复用的组件，提高了代码的可维护性和复用性。对于后端，我们选择了Flask作为Web框架，通过编写API接口，实现了用户管理、商品管理和订单管理等功能。Flask的简洁设计和灵活扩展的特性，使得我们能够快速构建出满足需求的后端服务。

1.3.3开发工具

在该电子商城系统的开发过程中，我们使用了PyCharm和VSCode作为主要的开发工具。PyCharm是一种强大的Python集成开发环境，提供了丰富的功能和工具，能够加速我们的开发过程。它具备代码自动完成、调试器、版本控制等特性，使得我们能够高效地编写和调试后端代码。同时，VSCode也是一种广受欢迎的轻量级代码编辑器，具备丰富的插件生态系统，支持多种编程语言和调试功能。我们可以使用VSCode进行前端代码的编写和调试，以及与团队协作和版本管理。通过PyCharm和VSCode这两个开发工具，我们能够更加便捷地进行开发、调试和版本控制，提高开发效率和代码质量。

# 需求分析

## 2.1 功能分析

分析系统功能，画功能结构图，并进行说明

系统需要实现一个简单的网上商城页面，可以总分为用户注册登录模块，订单管理查询模块和管理员分类管理模块，接下来我会分别介绍。

### 2.1.1用户注册登录模块

用户可分为未注册用户和已注册用户两种，在商城首页面有注册和登录按钮，用户可以根据需求来自主选择，选择注册则需要输入自己喜欢的账号和密码来进行注册，选择登录则需要输入已经完成注册保存在数据库中的账号和密码来实现登录。由图2.1可见

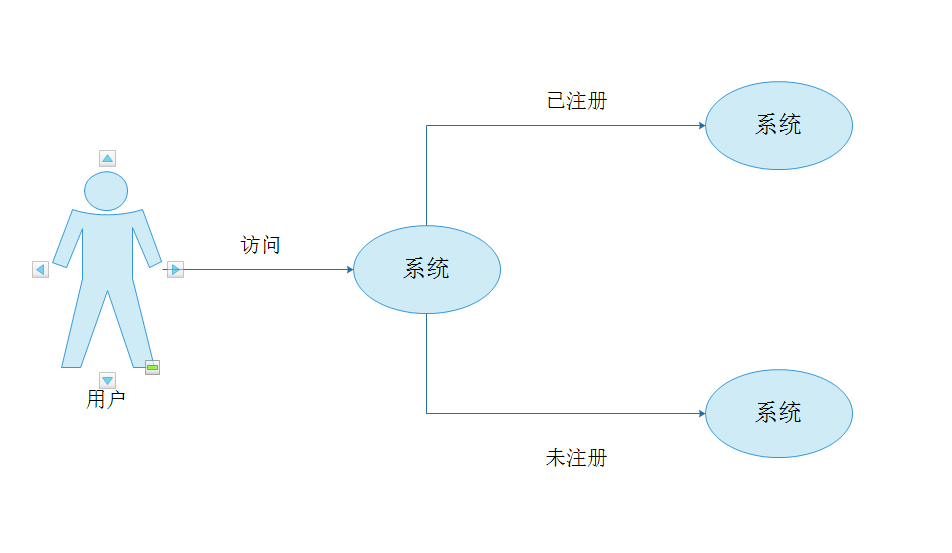


图2.1 用户模块

### 2.1.2订单管理查询模块

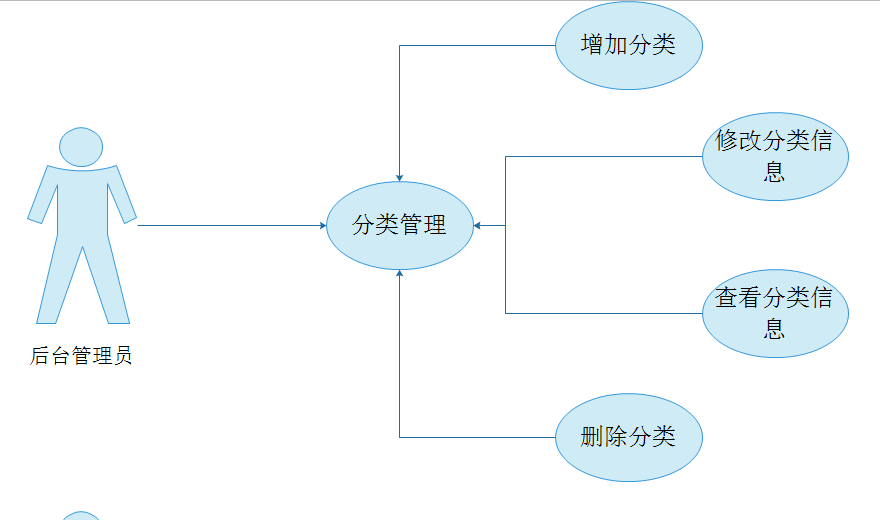
订单管理查询只包含两种用户，一类是后台管理员，一类是已注册用户，换而言之就是用户只有在注册以后才会被允许查询订单信息，已注册用户在商城前台确认收获和评价订单，后台管理员在商城后台查询订单信息和管理发货。由图2.2可见。



图2.2订单管理查询模块

### 2.1.3管理员分类管理模块

管理员可以在后台管理系统商城管理所有商品的分类，包括增删改查操作，不管是已注册用户还是未注册用户都可以在商城前台页面查看所有分类，但只能查看不能增删改查。



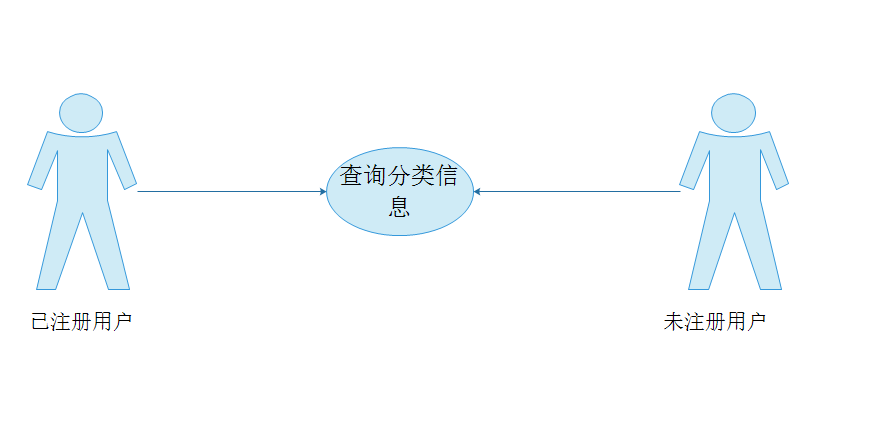


图2.3管理员分类管理模块

### 2.1.4层次功能图设计

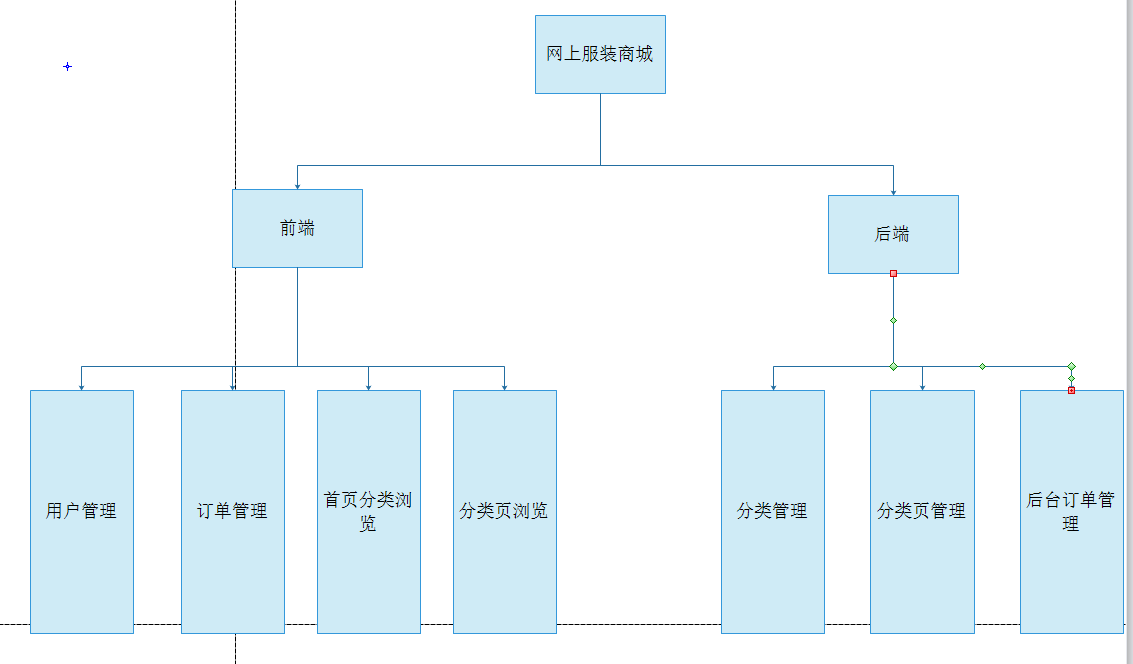


图2.4 系统层次功能图

## 2.2 概念模型分析

### 2.2.1用户实体

用户实体包含账号，密码，用户名，性别，地址等，可以选择性添加。如图2.5可见。

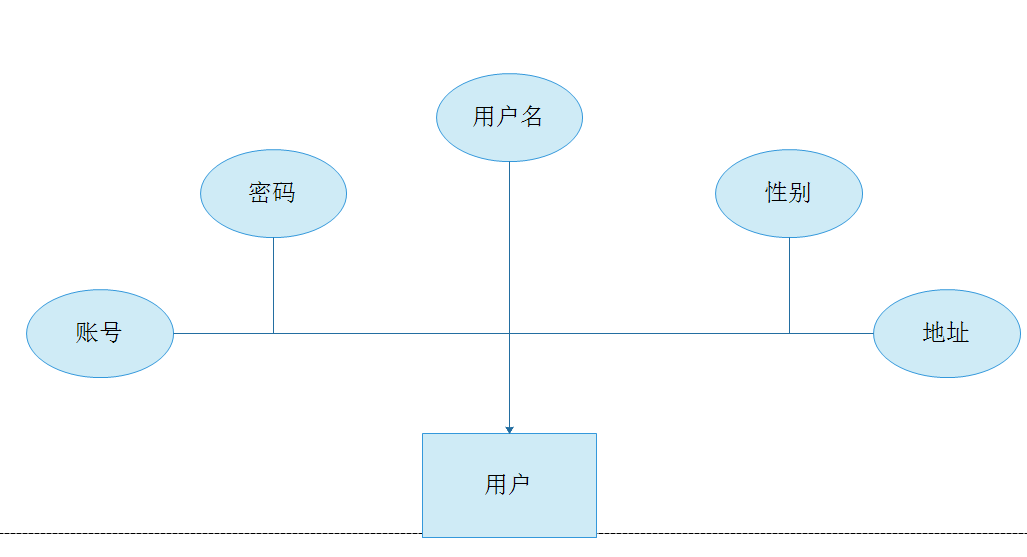


图2.5用户部分e-r图

### 2.2.2管理员实体

管理员实体包括账号，密码，用户名，性别等属性，可以选择性添加，如图2.6可见

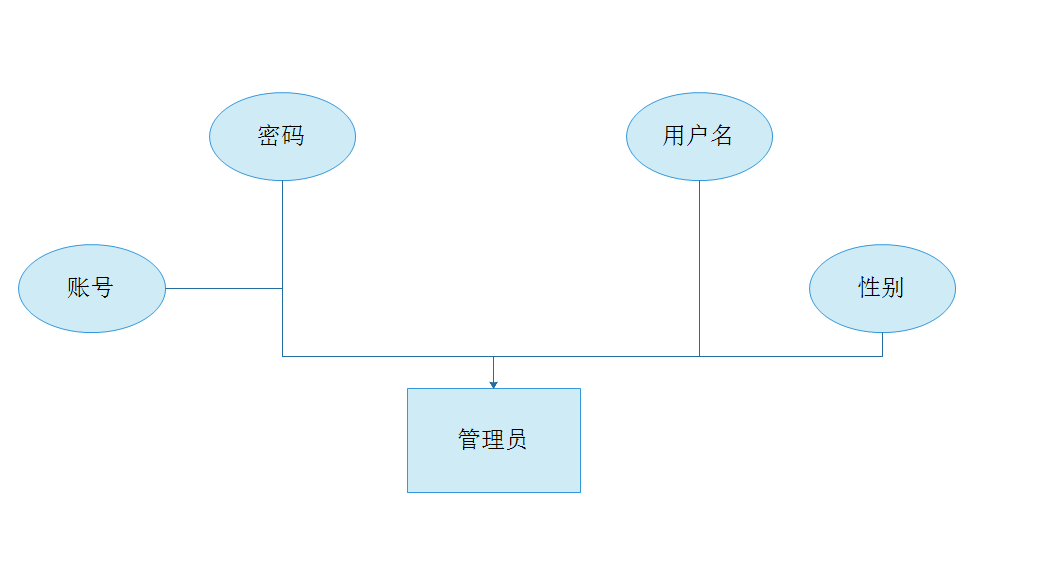


图2.6 管理员部分e-r图

### 2.2.3商品实体

商品实体包含编号，名称，价格，照片属性。如图2.7可见

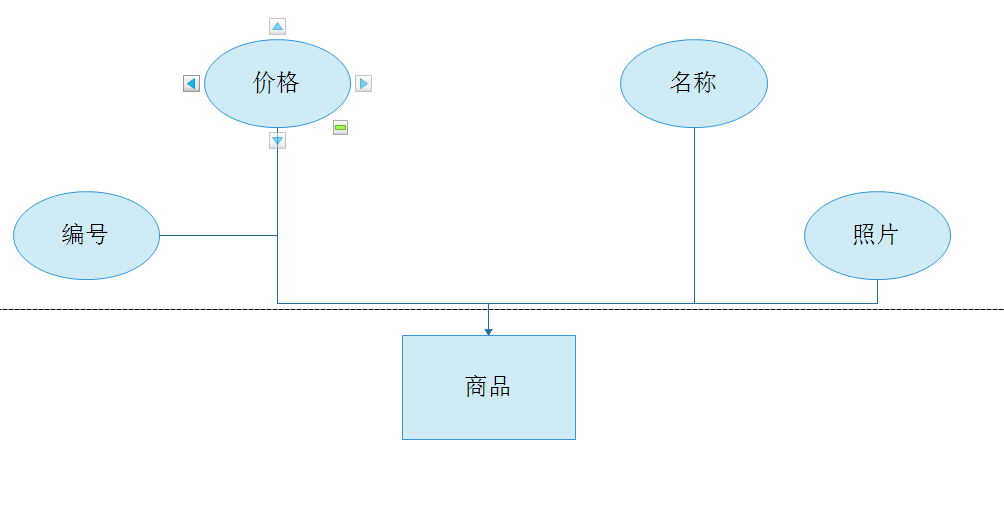


图2.7 商品部分e-r图

### 2.2.4总体e-r图

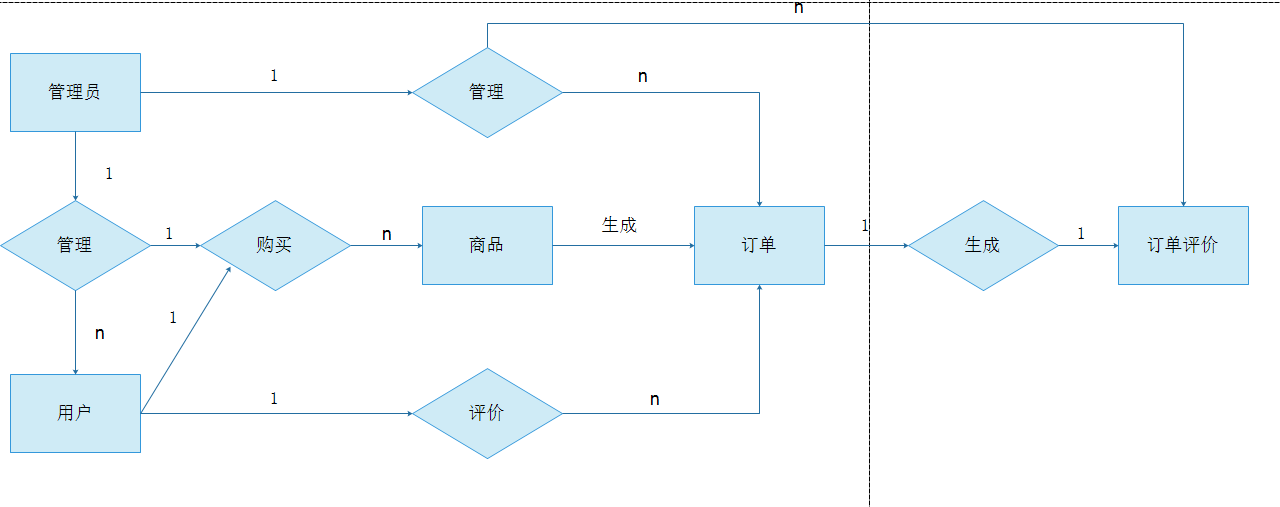


图2.9 总体e-r图

# 系统设计

## 3.1 数据库设计

本小姐将介绍系统开发所用到的数据库表，与三大模块紧密相连的五个数据库表分别是管理员表，商品表，购物车表，用户表。

### 3.1.1总表：

如图3.1为总表。



图3.1 总表

### 3.1.2管理员表（t\_admin）表

如图3.2管理员表

t\_admin表为管理员信息表。Id为int型，username为char型。Password为char型。

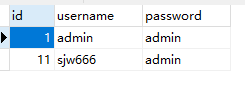


图3.2 管理员表

### 3.1.3商品（t\_goods）表

t\_goods表为商品信息表，id为char型，number为int型。Name为char型。Pic为指定的本地文件，price为int型。如图3.3商品表。

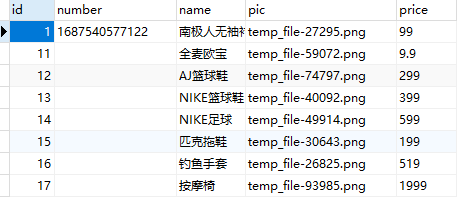


图3.3 商品表

### 3.1.4购物车（t\_shopcar）表

t\_goods表为购物车信息表，id为int型，goods为char型。Goodsid为int型。User为char型。Userid为int型，price为int型，number为int型，state为char型。如图3.4购物车表。



图3.4购物车表

### 3.1.5用户（t\_user）表

t\_user表主要为用户的数据表，其中id为int型，name为char型，username为char型，password为char型，sex为char型，pic为本地存储的图片，address为char型。如图3.5用户表。

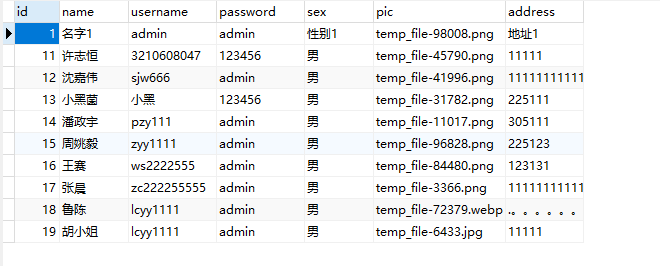


图3.5用户表

## 3.2 模块设计

### 3.2.1商城业务流程

受限用户可以登录商城首页浏览商品，此步不需要注册，如果选择商品信息则需要注册。系统会通过遍历数据库来检查用户是否完成注册，如果用户已经注册则可以将物品加入购物车，再选择购物车物品结算，填写收货地址，填写联系人，提交订单并收款，之后会产生一些订货信息到后台，管理员可以查看订单信息来处理发货，用户收到货后可以选择收获。如图3.6

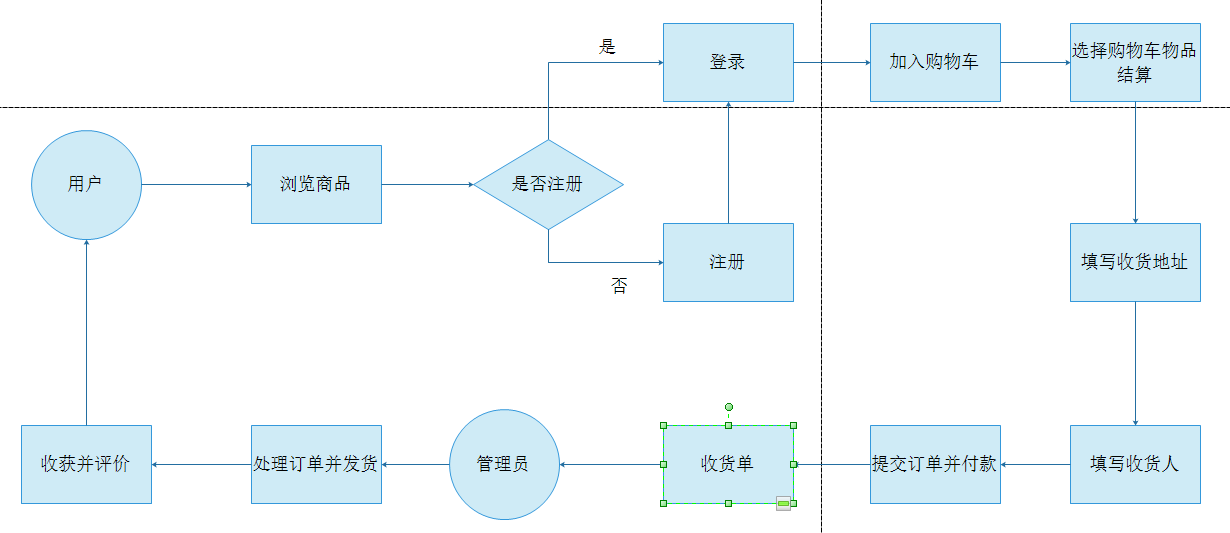


图3.6 商城业务流程图

### 3.2.2订单后台管理流程

首先管理员输入账号密码登陆后台，进入订单管理选项，一共有选择管理订单信息，管理管理员信息，管理用户信息四个选项。选择订单发货后会在用户确认收货后完成订单；查找订单需要输入订单在数据库中的编号，如果编号存在则可以查找到订单，不存在则会要求重新输入编号修改订单信息。管理用户信息可以根据自己的的需求来修改。如图3.7.

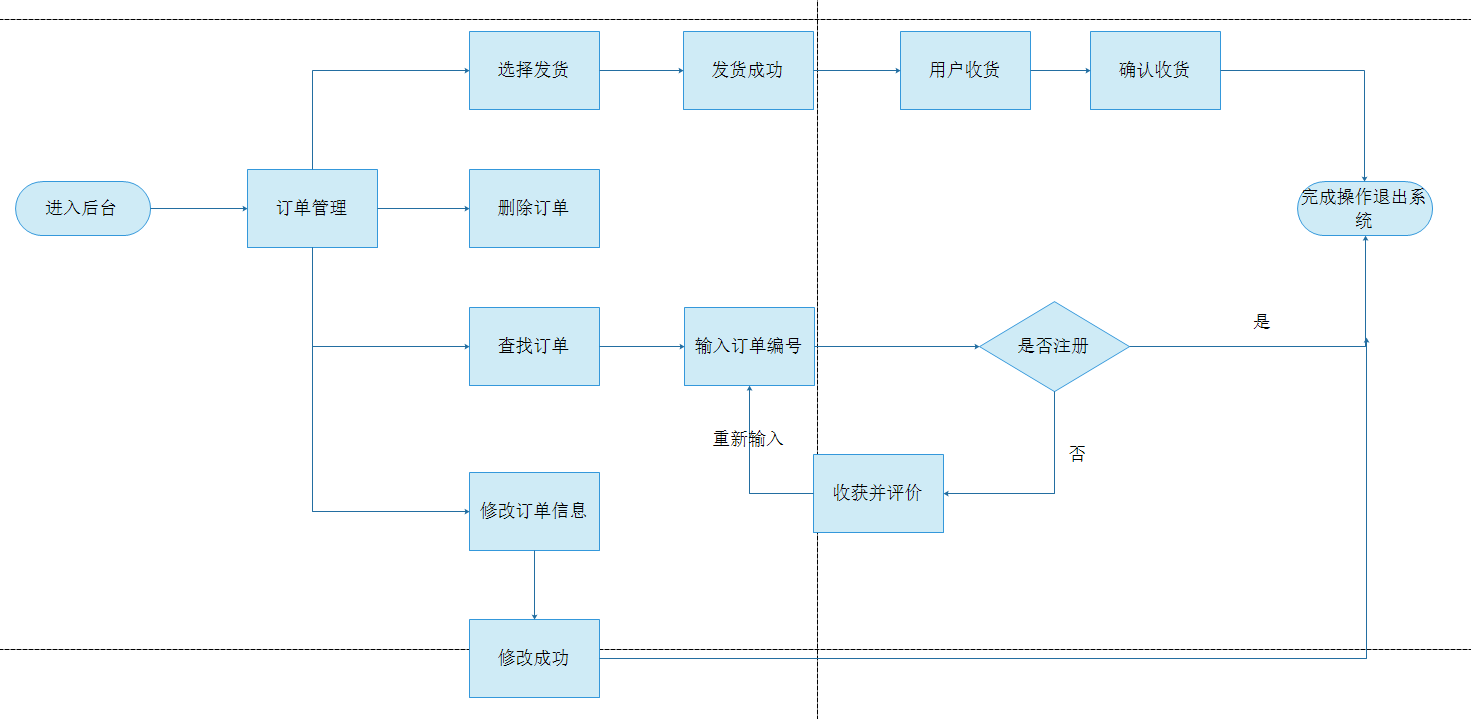


图3.7 订单后台管理流程

### 3.2.3用户管理流程

首先管理员输入账号密码登陆后台，进入用户管理选项，一共有添加新用户，删除用户，查找用户和修改用户信息四个选项。选择添加新用户则需要输入用户的信息；查找用户需要输入用户的账号来从数据库查询；修改用户信息则需要输入用户账号来查找修改的用户，再输入心得用户信息。从图3.8可见

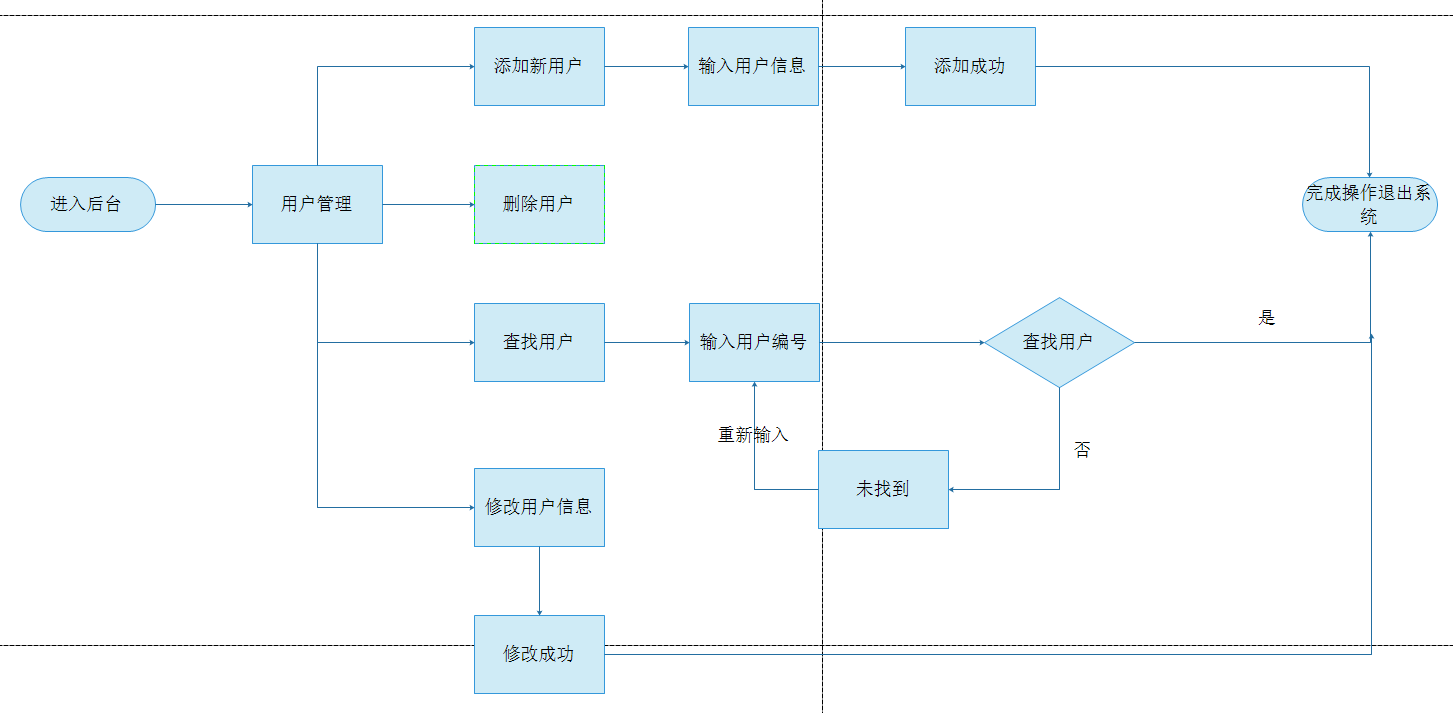


图3.8 用户管理流程

# 第四章 系统实现

## 4.1 项目介绍

### 4.1.1idea文件夹：

- 该文件夹是由JetBrains开发的IDE（集成开发环境）生成的，用于存储项目的配置文件和设置信息。这些文件包括项目的编码设置（encodings.xml）、JavaScript设置（misc.xml）和项目模块管理（modules.xml）等。

#### 4.1.1.1encodings.xml：

这个XML文件是一个项目配置文件，它包含了一个名为"Encoding"的组件，并设置了"addBOMForNewFiles"属性的值。

`<component name="Encoding">` 标签用于配置文件编码相关的设置。

- `addBOMForNewFiles="with NO BOM"` 设置了"addBOMForNewFiles"属性的值为"with NO BOM"。BOM代表字节顺序标记 (Byte Order Mark)，它是一个特殊的字符序列，用于指示文本文件的字节顺序和编码方式。"with NO BOM"表示在创建新文件时不添加BOM。

通过这个配置，项目设置了新文件的编码方式，指定了不在新文件中添加BOM。BOM的存在可能会影响文件的解析和处理，因此在某些情况下，特别是非Unicode编码的文件中，可以选择不添加BOM。

如图4.1encodings.xml

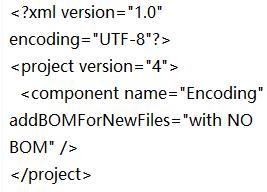


图4.1encodings.xml

#### 4.1.1.2misc.xml：

这个XML文件是一个项目配置文件，它包含了一些项目的设置信息。

1. `<component name="JavaScriptSettings">` 标签用于配置JavaScript相关的设置。

- `<option name="languageLevel" value="ES6" />` 指定了JavaScript的语言级别为ES6。这意味着项目使用ECMAScript 6标准的语法和特性。

2. `<component name="ProjectRootManager">` 标签用于配置项目的根目录管理器。

- `version="2"` 指定了根目录管理器的版本号。

- `project-jdk-name="Python 3.7 (GPT智能页面组合)"` 指定了项目使用的Python解释器的名称为"Python 3.7 (GPT智能页面组合)"。

- `project-jdk-type="Python SDK"` 指定了Python解释器的类型为Python SDK，表示使用Python开发环境。

通过这些配置，项目可以使用ES6标准的JavaScript语法和特性，并指定项目的根目录和所使用的Python解释器。这些配置可以帮助开发人员正确设置开发环境，以确保项目的正常运行和开发过程中的一致性。如图图4.2misc.xml

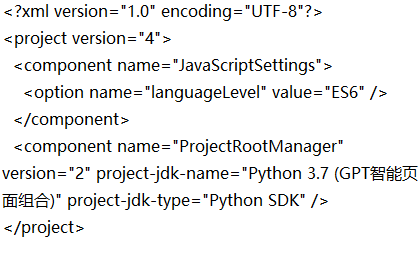


图4.2misc.xml

#### 4.1.1.3modules.xml：

这个文件在整体项目中的作用是描述项目模块的配置信息。它通过定义项目模块的路径和URL地址，提供了管理和组织项目结构的功能。

具体来说，这个文件中的 `<module>` 标签指定了一个模块文件的路径和URL地址，用于指示项目中存在的一个具体模块。通过这个配置，开发人员可以在项目中使用这个模块，并确保模块的正确加载和使用。

项目配置文件可以帮助开发人员更好地管理和组织项目结构，以及确保项目中所需的模块和组件被正确地加载和使用。这有助于提高开发效率，并确保项目的可维护性和可扩展性。如图4.3modules.xml

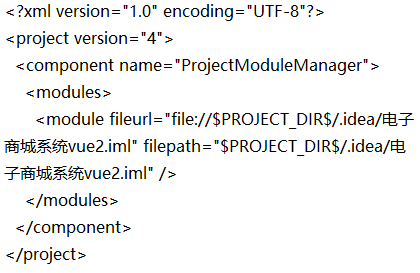


图4.3modules.xml

#### 4.1.1.4workspace.xml：

这个文件是一个项目配置文件，主要用于管理项目的变更列表、文件编辑器的状态以及项目视图等。它包含了多个组件和选项，每个组件都有不同的功能。

其中，`ChangeListManager` 组件用于管理项目的变更列表，可以添加、删除和管理代码变更记录。

`FileEditorManager` 组件用于管理文件编辑器的状态，包括打开的文件和当前编辑的位置。

`FindInProjectRecents` 组件用于记录最近的项目搜索和替换的字符串。

`IdeDocumentHistory` 组件记录了项目中的文件修改历史。

`ProjectFrameBounds` 组件用于记录项目窗口的位置和大小。

`ProjectView` 组件用于管理项目的视图结构，包括文件和文件夹的层次结构。

`PropertiesComponent` 组件用于存储和管理项目的属性配置。

`RecentsManager` 组件用于记录最近打开的文件和目录。

`RunDashboard` 组件用于管理运行配置的面板显示和规则。

这个文件的主要功能是配置和管理项目的各种属性、状态和视图，以便开发者可以方便地进行项目开发和管理。如图4.4workspace.xml

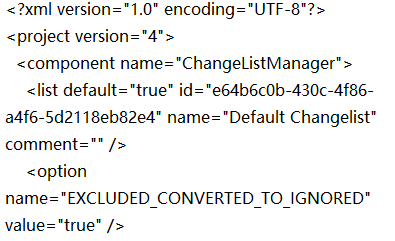


图4.4 workspace.xml

### 4.1.2 build文件夹：

- 该文件夹可能是用于存储项目构建相关的文件。通常在构建过程中，生成的文件会被输出到这个文件夹中。如图4.5build

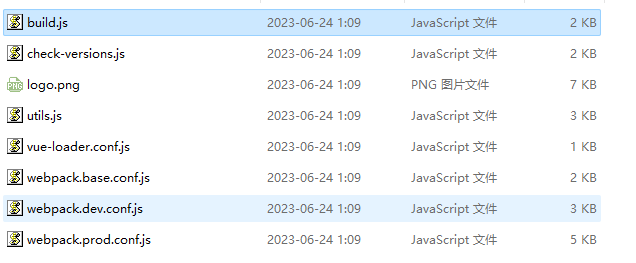


图4.5build

#### 4.1.2.1build.js：

这个文件是用来构建生产环境的脚本。它首先检查版本要求，然后使用webpack进行构建。它会清理之前构建的文件，并根据webpack配置进行打包。构建过程中会显示构建进度，并在构建完成后输出相应的信息。

4.1.2.2check-versions.js：

这个文件用于检查版本要求是否满足。它会检查当前环境中Node.js和npm的版本是否符合package.json中定义的要求。如果不满足要求，将显示警告信息并终止构建过程。

4.1.2.3utils.js：

这个文件包含了一些辅助函数，用于处理资源路径和加载器配置。其中包括了生成CSS加载器、样式加载器以及创建通知回调函数等功能。这些函数在构建过程中被其他文件引用，帮助配置和处理资源加载。

这些文件在构建过程中扮演了不同的角色，分别负责版本检查、构建过程和工具函数的处理。它们共同协作，确保生产环境的构建顺利进行。

### 4.1.3. config文件夹：

- 该文件夹包含一些项目的配置文件。这些文件可能包括构建配置、部署配置或其他项目相关的配置信息。如图4.6config文件夹

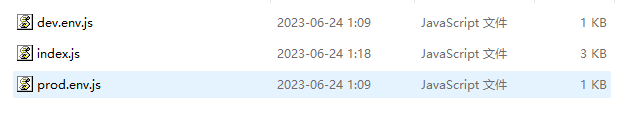


图4.6 config文件夹

4.1.3.1dev.env.js：

这个文件是用于开发环境的配置。它通过引入prod.env文件并合并配置，设置了NODE\_ENV为"development"，表示当前环境是开发环境。

4.1.3.2index.js：

这个文件是一个模板文件，用于配置webpack。它首先引入了一些依赖模块，包括path和os。然后定义了一个函数getIPAdress()，用于获取本机的IP地址。接下来，导出一个配置对象，包含了开发环境（dev）和生产环境（build）的相关配置信息。

在dev对象中，定义了一些开发服务器的设置，包括资源路径、代理表、主机、端口等。其中代理表指定了"/api"路径下的请求将被转发到"http://127.0.0.1:5000"，实现了前后端联调的代理功能。

在build对象中，定义了构建过程的相关配置信息，包括生成的index.html文件路径、静态资源路径、Source Maps的设置、gzip压缩等。

这两个文件分别用于开发环境和生产环境的配置，通过设置不同的参数和选项，为不同环境下的构建和开发提供了相应的支持和功能。

### 4.1.4. node\_modules文件夹：

- 该文件夹用于存储项目的依赖模块，这些模块通常是通过npm（Node.js的包管理工具）进行安装的。这些依赖模块包括Vue.js、Flask等项目所需的第三方库或框架。如图4.7node\_modules文件夹

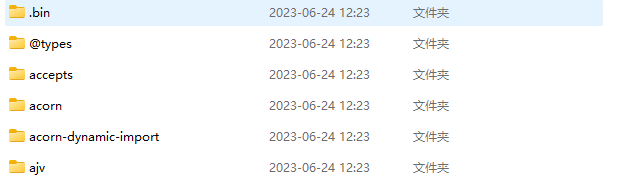


图4.7node\_modules文件夹

### 4.1.5. src文件夹：

- 该文件夹是项目的核心源代码目录，包含了前端Vue.js应用程序的开发代码。这里包含Vue组件、路由配置、API请求、页面模板等。如图4.8src文件夹

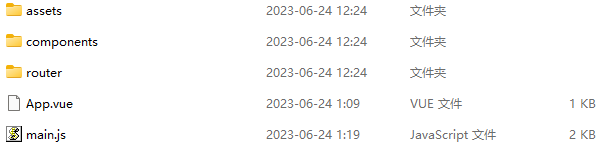


图4.8src文件夹

#### 4.1.5.1AddAdmin.vue

这段代码是一个Vue组件的代码，主要实现了一个添加管理员的功能页面。以下是该代码的主要功能和特点：

1. 页面结构：代码中使用了Vue的模板语法，定义了一个包含多个子组件的页面结构。页面包括顶部导航栏、侧边栏菜单、表单等元素。

2. 数据绑定：通过Vue的数据绑定语法，将页面中的输入框和数据对象（admin）进行了绑定，实现了双向数据绑定。输入框中的值变化会自动更新admin对象的对应属性的值。

3. 方法和事件处理：代码中定义了多个方法，用于处理表单的提交、文件上传等操作。例如，addadminact方法用于提交添加管理员的请求，uploadfile方法用于处理文件上传操作。

4. 组件引用：代码中引入了名为Htmenu的组件，用于渲染侧边栏菜单。

5. 生命周期钩子函数：代码中使用了Vue的生命周期钩子函数，例如created、beforeCreate和mounted，用于在不同阶段执行相应的逻辑。

6. 样式和脚本引用：代码中动态插入了样式和脚本引用的HTML代码，用于加载相关的CSS和JavaScript文件。

总体来说，该代码实现了一个添加管理员的功能页面，通过Vue的数据绑定和事件处理机制，实现了表单数据的提交和文件上传等功能。同时，通过组件化的方式，使页面结构清晰，易于维护和扩展。如图4.9AddAdmin.vue





图4.9AddAdmin.vue

#### 4.1.5.2AddGoods.vue:

这个代码片段是一个用于添加商品的页面组件，主要功能是提供一个表单，用户可以填写商品的名称、价格，并上传商品的照片。下面是页面的详细作用和一些特点：

1. 页面布局：

- 页面顶部有一个导航栏，显示网站的品牌标志。

- 左侧是一个垂直导航栏，显示不同的操作选项。

- 页面中间是一个卡片，包含商品添加的表单。

- 页面底部是一个版权信息。

2. 表单字段：

- 商品名称（name）：用户可以输入商品的名称。

- 商品照片（pic）：用户可以通过上传图片来设置商品的照片。使用了Element UI的el-upload组件实现上传功能，包括预览、下载和移除已上传的照片。

- 商品价格（price）：用户可以输入商品的价格

3. 功能特点：

- 图片上传：使用了el-upload组件实现图片的上传功能，用户可以通过点击加号图标选择本地的图片文件进行上传。

- 图片预览：用户可以点击照片列表中的预览按钮，弹出一个对话框显示大图预览。

- 图片下载：用户可以点击照片列表中的下载按钮，将图片文件下载到本地。

- 图片移除：用户可以点击照片列表中的删除按钮，从列表中移除已上传的照片。

- 表单提交：用户填写完商品信息后，可以点击"添加"按钮将商品信息提交到服务器进行保存。

4. 其他特点：

- 使用了Element UI库来实现表单组件和上传组件，提供了丰富的样式和交互功能。

- 通过axios库发送HTTP请求与服务器进行数据交互，包括上传图片和提交商品信息。如图4.11AddGoods.vue



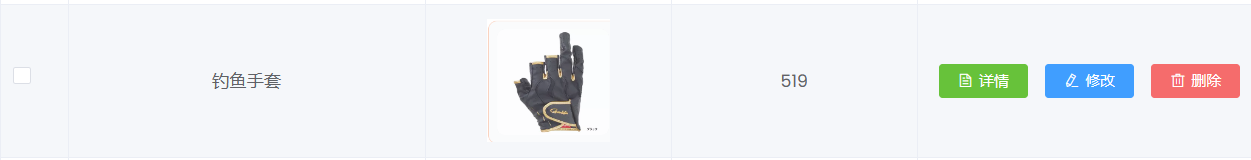


图4.11 AddGoods.vue

#### 4.1.5.3AddShopcar.vue:

功能：

1. 提供一个表单，用于添加购物车信息。

2. 实现了图片上传功能，用户可以上传商品照片。

3. 实现了商品和用户的选择功能，通过下拉选择框选择对应的商品和用户。

4. 支持填写价格、数量和状态等购物车信息。

5. 提供了添加购物车按钮，点击后将购物车信息提交到服务器。

页面详细作用：

1. 页面布局：包含导航栏、侧边栏、内容区域和页脚。

2. 导航栏：显示网站的品牌标志，点击可以返回首页。

3. 侧边栏：展示操作选项，其中的"htmenu"组件用于显示菜单项。

4. 内容区域：显示添加购物车的表单，包括商品、用户、价格、数量和状态等字段。

5. 页脚：显示版权信息。

特点：

1. 使用了Element UI库来实现表单组件和选择器，提供了丰富的样式和交互功能。

2. 使用了axios库发送HTTP请求与服务器进行数据交互，包括上传图片和提交购物车信息。

3. 通过监听文件选择框的变化，实现了图片的异步上传，并更新页面显示。

4. 使用了路由来跳转到购物车管理页面。

5. 通过sessionStorage来获取用户身份和用户信息，并根据需求进行相应的数据查询和处理。

如图4.12 AddShopcar.vue。





图4.12 AddShopcar.vue

#### 4.1.5.4AddUser.vue:

这段代码是一个名为"AddUser.vue"的Vue.js组件。让我们分析它的功能、页面细节和特点：

功能：

- 此组件用于向电子商城平台添加用户。

- 它提供一个表单，用于输入用户的详细信息，如姓名、用户名、密码、性别、照片和地址。

- 表单允许用户上传个人照片，并查看、下载或删除已上传的照片。

- 点击"添加"按钮后，组件将向"/adduseractjson"端点发送一个POST请求，包含用户数据，并根据响应处理成功或失败的情况。

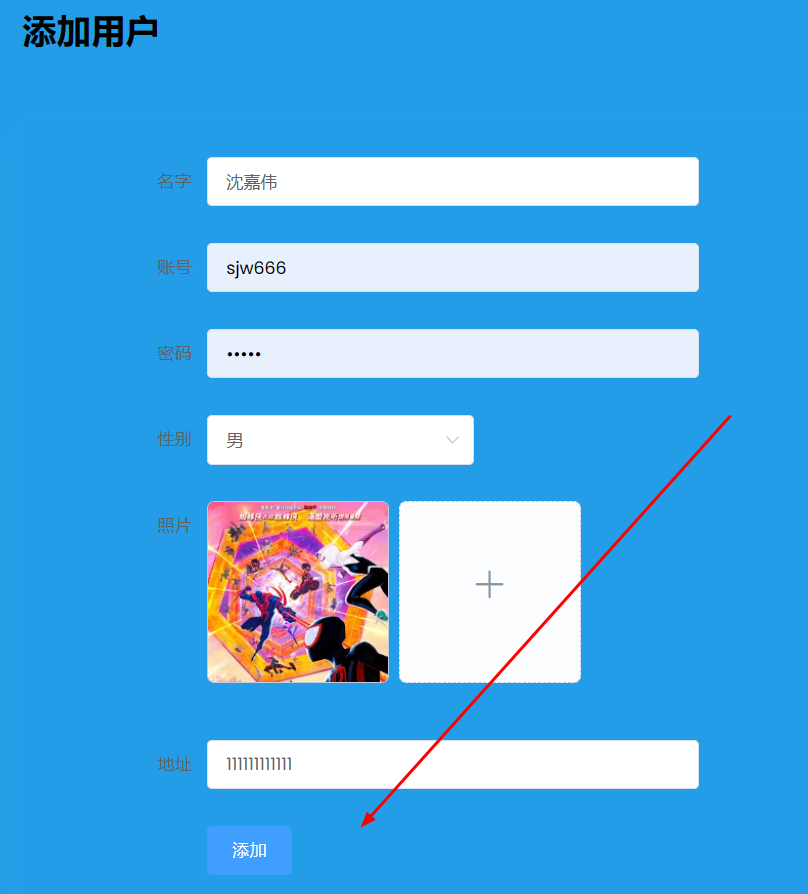
页面细节：

- 该组件使用了多个Vue.js和Element UI组件来创建表单并处理用户交互。

- 页面包括导航头部、侧边栏菜单和页脚，用于整体布局。

- 表单字段包括姓名、用户名、密码、性别（选择框）、照片（上传框）和地址的输入框。

- 提交表单会触发"adduseract"方法。如图4.13AddUser.vue



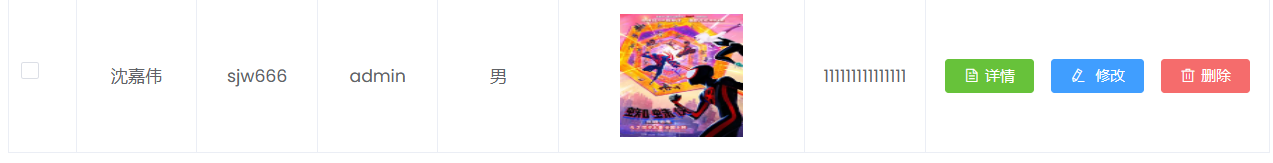


图4.13 AddUser.vue

#### 4.1.5.5AdminIndex.vue:

AdminIndex.vue是一个Vue.js组件，它用于管理员个人中心页面的展示和管理。让我们来分析它的功能、页面细节和特点：

功能：

- 此组件用于展示管理员个人中心的页面。

- 它提供了一个表单，显示和编辑管理员的账号和密码。

- 管理员可以修改密码并保存更改。

- 页面会通过Ajax请求获取管理员的详细信息并显示在表单中。

- 管理员可以上传文件，并将上传后的文件路径保存到表单中。

页面细节：

- 页面包括导航头部、侧边栏菜单和页脚，用于整体布局。

- 页面标题显示为"管理员个人中心"。

- 表单字段包括管理员id（隐藏字段）、账号（只读字段）和密码的输入框。

- 点击"修改"按钮会触发"updateadminact"方法，将修改后的管理员信息保存到数据库。

- 页面底部显示版权信息。

特点：

- 组件引入了名为Htmenu的子组件，用于展示侧边栏菜单。

- 使用Element UI库中的el-form、el-input和el-button等组件来创建表单和处理用户交互。

- 使用Axios库进行Ajax请求，获取和保存管理员信息。

- 页面加载时会执行一些初始化操作，如加载CSS和JS文件。

总体上，AdminIndex.vue组件提供了管理员个人中心页面的功能和布局，以及与后端服务器进行数据交互的能力。如图4.14AdminIndex.vue



图4.14 AdminIndex.vue

#### 4.1.5.6AdminManage.vue:

`AdminManage.vue` 组件是电子商城管理系统中的一个部分，主要用于管理员管理。以下是其功能、页面结构和特点的详细说明：

功能：

1. 显示管理员的表格，包含账号详细信息（用户名和密码），并提供查看、修改和删除管理员的操作。

2. 支持根据提供的条件（用户名和密码）搜索管理员。

3. 允许批量删除选定的管理员。

4. 提供分页功能，以便浏览管理员列表。

页面结构：

1. 模板部分包含页面的 HTML 结构，包括页眉、导航菜单、内容区域、表格和页脚。

2. 包含 `<htmenu>` 组件以呈现导航菜单。

3. 表格显示管理员列表，列出账号详细信息，并提供查看、修改和删除管理员的操作按钮。

4. 分页控件位于表格底部，用于浏览管理员列表。

特点：

1. 使用 Element UI 库（`el-` 组件）提供 UI 元素，如表单、按钮、表格和分页。

2. 使用数据属性（`adminall`、`adminDetails`、`pageSize` 等）来存储和操作与管理员相关的数据。

3. 使用 Axios（`this.$axios`）向服务器端 API 发起 HTTP 请求。

4. 定义各种方法来处理搜索管理员、选择多个管理员、删除管理员和浏览分页等操作。

5. 使用对话框（`<el-dialog>`）来显示管理员详细信息。

6. 使用 CSS 类进行样式设置和响应式设计（`col-12`、`row`、`card` 等）。

7. 包含一些自定义的 JavaScript 函数（`uploadfile`、`getNowTime`、`serializeForm`），用于文件上传、日期格式化和表单序列化。

8. 在 `beforeCreate` 和 `mounted` 生命周期钩子中初始化和配置各种外部 JavaScript 库（`jquery.dataTables`、`bootstrap-select`）。

总体而言，`AdminManage.vue` 组件提供了一个界面来管理电子商城管理系统中的管理员。它允许搜索、查看、修改和删除管理员，并提供了用户友好的表格视图和分页控制。如图4.15AdminManage.vue



图4.15 AdminManage.vue

#### 4.1.5.7AdminView.vue:

根据提供的代码，`AdminView.vue` 组件是电子商城管理系统中的一个部分，用于管理员的查看功能。以下是该组件的功能、页面结构和特点的详细分析：

功能：

1. 显示电子商城管理系统的导航菜单（`Htmenu` 组件）。

2. 允许管理员根据账号和密码进行搜索。

3. 显示管理员列表的表格，包括账号和密码信息，并提供查看详情的按钮。

4. 支持选择多个管理员，并提供批量删除操作。

5. 提供分页功能，允许管理员浏览不同页的管理员列表。

6. 显示管理员详细信息的对话框，并提供关闭按钮。

页面结构：

1. 页面顶部包含商城的品牌标志和导航菜单。

2. 表格和搜索框位于页面的主要内容区域。

3. 表格使用 Element UI 的 `el-table` 组件进行展示，显示管理员列表，并提供账号、密码和操作列。

4. 分页控件（`el-pagination`）位于表格下方，允许管理员切换页码和调整每页显示的管理员数量。

5. 管理员详细信息的对话框使用 Element UI 的 `el-dialog` 组件实现，包含账号和密码的展示，并提供关闭按钮。

特点：

1. 使用 Element UI 库提供丰富的 UI 组件，如表格、分页、输入框和对话框，增强了用户体验。

2. 使用 Axios 库向服务器端发起 HTTP 请求，实现搜索、删除等操作。

3. 通过监听选择事件实现批量删除功能。

4. 通过计算属性和方法对管理员列表进行分页展示。

5. 使用路由导航实现跳转到更新管理员信息页面（`/UpdateAdmin`）的功能。

6. 页面初始化前（`beforeCreate`）和初始化后（`mounted`）执行一些操作，如加载外部的 CSS 和 JavaScript 文件，以及其他初始化逻辑。

总体而言，`AdminView.vue` 组件提供了一个界面用于查看和管理电子商城管理系统中的管理员信息。它具有搜索、分页、删除和查看详情的功能，并通过使用 Element UI 和 Axios 等库提供了良好的用户界面和数据交互体验。

#### 4.1.5.8GoodsDetails.vue:

`GoodsDetails.vue` 是一个 Vue.js 组件，用于定义“商品详情”页面的模板、逻辑和样式。下面我们来分析它的功能、页面细节和特点：

功能：

- 该组件用于获取并显示特定商品的详细信息。

- 它允许用户查看商品的名称、图片、编号和价格。

- 根据用户的身份（存储在 `identity` 数据属性中），提供不同的操作选项。

- 如果用户身份为“用户”，则显示“购买”按钮，点击后会跳转到用户的购物车页面（`UserAddShopcar`），并将选定商品的ID和价格作为查询参数传递。

页面细节：

- 页面顶部包含一个导航栏，其中有一个 Logo 图标和主菜单导航项。

- 面包屑导航区域显示当前页面的路径，以及一个指向首页的链接。

- 商品详细信息以卡片形式展示，包括商品名称和图片。

- 下方是一个表格，显示商品的编号、名称和价格。

- 页面底部有一个版权信息区域，显示网站的 Logo 和简要描述。

特点：

- 采用了响应式设计，适应不同屏幕尺寸的设备。

- 使用了 Bootstrap 框架和其他一些样式文件，以实现页面布局和样式效果。

- 通过 Vue Router 实现了页面跳转和传递参数的功能。

- 使用 Axios 库发送异步请求，获取商品详细信息并进行展示。

以上是对 `GoodsDetails.vue` 组件的功能、页面细节和特点的简要分析。

如图4.16GoodsDetails.vue



图4.16 GoodsDetails.vue

#### 4.1.5.9GoodsManage.vue:

`GoodsManage.vue` 是一个 Vue.js 组件，用于定义“商品管理”页面的模板、逻辑和样式。下面是对该组件的功能、页面细节和特点的分析：

功能：

- 该组件用于管理商品，包括显示、搜索、删除、修改商品信息。

- 它提供了表格形式展示商品列表，可以进行多选删除和分页功能。

- 点击详情按钮可以查看商品的详细信息，点击修改按钮可以跳转到修改商品页面。

页面细节：

- 页面顶部有一个导航栏，包括网站 Logo 和主菜单导航项。

- 导航栏下方是一个垂直滚动的菜单栏，包含商品管理相关的选项。

- 页面中间是一个卡片式布局的商品列表，包括商品的名称、照片、价格和操作按钮。

- 每个商品列表项的操作按钮包括详情、修改和删除。

- 在底部是一个分页组件，用于切换页面和显示总共商品数量。

特点：

- 采用了响应式设计，适应不同屏幕尺寸的设备。

- 使用了 Element UI 框架，包括表格、分页和弹窗组件等。

- 利用 Axios 库发送异步请求，获取商品数据和进行删除操作。

- 使用了路由导航实现页面跳转。

- 通过条件搜索功能实现根据商品编号和名称进行商品筛选。如图4.17GoodsManage.vue

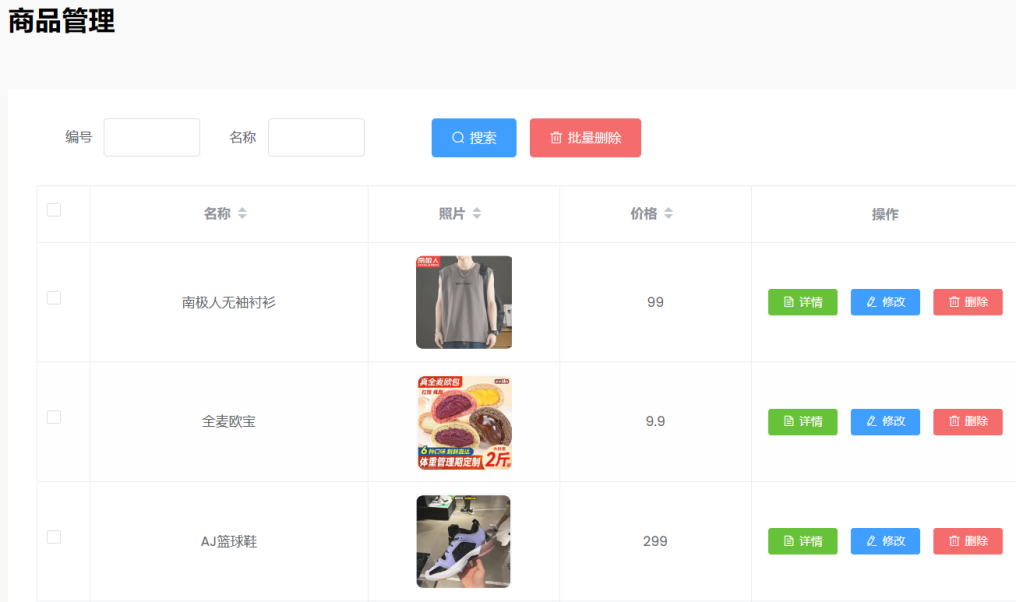


图4.17 GoodsManage.vue

#### 4.1.5.10GoodsView.vue:

这是一个名为"GoodsView"的Vue组件，主要用于展示商品信息和实现商品搜索、删除和查看详情等功能。下面对其功能、页面作用和特点进行详细分析：

功能：

1. 展示商品信息：通过调用接口获取商品数据并在表格中展示，包括商品编号、名称、照片、价格等字段。

2. 商品搜索：在页面顶部提供了编号和名称的搜索输入框，用户可以输入相应的条件进行商品搜索。

3. 商品删除：用户可以选择多个商品，在表格中勾选后点击删除按钮进行批量删除。

4. 商品详情查看：用户可以点击表格中的详情按钮，弹出对话框展示选中商品的详细信息，包括名称、照片、价格等字段。

页面作用：

1. 页面布局：通过HTML和CSS实现页面的整体布局，包括导航栏、侧边栏、内容区域和页脚等。

2. 商品列表展示：在内容区域的表格中展示商品信息，包括各个字段的列和分页功能。

3. 商品搜索和删除：通过搜索框和删除按钮实现商品搜索和删除功能。

4. 商品详情对话框：点击详情按钮弹出对话框展示商品的详细信息。

特点：

1. 使用了Element UI组件库：通过引入Element UI的组件，如el-form、el-table、el-pagination等，简化了页面开发和交互逻辑的实现。

2. 图片展示和处理：在表格中展示商品照片时，通过v-if和v-else指令判断是否有照片，并使用img标签显示图片。

3. 分页功能：使用el-pagination组件实现商品列表的分页功能，包括选择每页显示数量和跳转到指定页码。

4. 对话框组件：使用el-dialog组件实现商品详情对话框的展示和关闭功能。

5. 调用接口获取数据：通过使用axios库发送HTTP请求，调用接口获取商品数据，并在页面中进行展示和处理。

总体而言，该页面是一个简单的商品管理页面，提供了商品的展示、搜索、删除和查看详情等基本功能，使用了Vue和Element UI等技术实现。如图4.18GoodsView.vue



图4.18 GoodsView.vue

#### 4.1.5.11Htmenu.vue:

Htmenu.vue 是一个Vue组件，它实现了一个动态的导航菜单功能。根据用户的身份（identity），显示不同的菜单选项。

该组件的主要功能包括：

1. 根据用户身份（identity）显示不同的菜单：根据用户的身份（"管理员"或"用户"），该组件会动态地显示相应身份的菜单选项。如果用户身份是"管理员"，则显示管理员相关的菜单选项；如果用户身份是"用户"，则显示用户相关的菜单选项。

2. 点击菜单项进行导航：每个菜单项都包含一个链接，当用户点击菜单项时，会导航到相应的页面。点击"个人中心"菜单项时，会导航到个人中心页面；点击"管理员管理"菜单项时，会导航到管理员管理页面，依此类推。

3. 退出系统功能：在菜单的最后一个选项中，提供了"退出系统"功能。当用户点击该选项时，会清除sessionStorage中的用户信息，并导航到登录页面。

该组件的特点包括：

1. 根据身份动态显示菜单：根据用户的身份不同，会动态地显示不同的菜单选项。这样可以提供更好的用户体验，确保用户只能看到与其身份相关的菜单项。

2. 使用了Vue框架：Htmenu.vue 是一个Vue组件，利用了Vue框架的特性，例如数据绑定和组件生命周期钩子函数。

3. 使用了Vue Router进行页面导航：在点击菜单项时，通过Vue Router进行页面导航，实现了单页应用的无刷新跳转。

4. 使用了sessionStorage存储用户信息：用户身份和用户信息通过sessionStorage进行存储，以便在不同页面间进行共享和获取.

总的来说，Htmenu.vue 是一个根据用户身份显示不同菜单选项的Vue组件，提供了导航功能和退出系统功能，通过使用Vue框架和Vue Router实现了这些功能，并利用sessionStorage存储用户信息。如图4.19Htmenu.vue



图4.19 Htmenu.vue

#### 4.1.5.12Index.vue:

这是一个搜索商品的主页，具体功能如下：

1. 在页面顶部有一个导航栏，包含网站的logo和菜单项。

2. 页面顶部下方有一个面包屑导航，显示当前所在位置。

3. 页面中间是一个搜索栏，可以输入关键字进行商品搜索。点击搜索按钮可以触发搜索功能。

4. 页面下方是展示商品的区域，使用了循环（v-for）来遍历商品列表（goodsall），每个商品以图片、商品名称和详情按钮的形式展示。

5. 页面底部是网站的页脚部分，包含网站的logo和版权信息。

6. 页面使用了一些CSS和JavaScript库，用于样式和交互效果的实现。

页面的特点：

- 页面布局简洁清晰，采用了响应式设计，适应不同屏幕尺寸。

- 使用了Vue.js框架，通过组件化的方式管理页面的各个部分。

- 通过AJAX请求获取商品数据，实现动态展示和搜索功能。

- 使用了一些第三方库和插件，增加了页面的样式和交互效果。

总体而言，该页面提供了一个方便用户搜索商品的界面，展示了搜索结果并支持用户查看商品详情。如图4.20Index.vue





图4.20 Index.vue

#### 4.1.5.13Login.vue：

Login.vue 是一个登录页面的组件，用于用户进行登录操作。

该组件包含以下功能和特点：

1. 页面布局：通过 Bootstrap 样式库实现了响应式布局，适应不同屏幕大小的设备。

2. 表单输入：用户可以输入用户名和密码进行登录。输入框通过 `qw-c` 属性与组件内部的数据绑定，可以通过 `name` 属性获取表单数据。

3. 身份选择：用户可以通过下拉菜单选择登录身份，有两个选项：管理员和用户。选择的身份信息也与组件内部的数据绑定，可以通过 `name` 属性获取选择的身份。

4. 文件上传：通过 `uploadfile` 方法实现了文件上传功能。当用户选择文件后，调用该方法将文件发送到服务器进行上传，并更新组件内部的数据。

5. 数据处理：通过 `serializeForm` 方法将表单数据序列化为对象，并调用后端接口进行登录验证。验证结果会返回一个包含消息、用户信息、身份和令牌等数据的对象。

6. 路由跳转：根据登录结果，使用 Vue Router 实现页面的跳转。如果登录成功，根据身份的不同跳转到不同的页面。

7. 生命周期钩子：使用 Vue 的生命周期钩子函数，在组件的不同阶段执行相应的操作，例如在 `created` 钩子函数中获取浏览器缓存中的用户信息和身份。

8. CSS 和 JS 引入：通过动态操作 HTML 元素，动态引入了相关的 CSS 和 JS 文件，以样式和脚本的方式扩展页面功能。

总体来说，Login.vue 组件提供了用户登录的界面和相关功能，包括表单输入、身份选择、文件上传、数据处理和页面跳转等。它使用了现代的前端技术和框架，使用户可以方便地进行登录操作，并根据登录结果进行相应的处理。

如图4.21Login.vue



图4.21 Login.vue

#### 4.1.5.14Qtmenu.vue：

Qtmenu.vue 是一个导航菜单组件，用于在页面中显示导航链接和根据用户身份显示不同的菜单选项。

该组件具有以下功能和特点：

1. 导航菜单：通过 `<ul>` 和 `<li>` 元素创建一个无序列表，用于显示导航链接。

2. 菜单项：每个导航链接都位于一个 `<li>` 元素中，并通过 `qw-c` 属性与组件内部的数据绑定。

3. 身份判断：通过条件判断 (`v-if`) 根据用户的身份来显示不同的菜单选项。如果用户为管理员，显示个人中心链接；如果用户为普通用户，显示个人中心链接；如果用户未登录，则显示登录和注册链接。

4. 路由跳转：导航链接通过 `<a>` 标签的 `href` 属性设置路由路径，通过 Vue Router 实现页面之间的跳转。

5. 数据获取：在 `created` 生命周期钩子函数中，从浏览器缓存中获取用户身份和用户信息，并将其保存在组件的 `identity` 和 `userinfo` 数据中。

6. 退出系统：通过 `exitsystem` 方法实现用户退出系统的功能。调用该方法时，清空浏览器缓存中的数据，并跳转到登录页面。

总体来说，Qtmenu.vue 组件用于在页面中显示导航菜单，并根据用户身份显示不同的菜单选项。它通过获取用户信息、条件判断和路由跳转实现了不同菜单的显示和功能。该组件可以提供导航功能和用户操作的便利性，使用户可以方便地浏览和操作系统的不同页面和功能。

如图4.22Qtmenu.vue



图4.22Qtmenu.vue

#### 4.1.5.15Regist.vue：

Regist.vue 是一个注册页面组件，用于用户进行注册操作。

该组件具有以下功能和特点：

1. 注册表单：通过 `<form>` 元素创建一个注册表单，其中包含账号、密码、确认密码和身份等输入字段。

2. 注册按钮：通过 `<button>` 元素创建一个注册按钮，点击该按钮触发 `registact` 方法进行注册操作。

3. 注册请求：在 `registact` 方法中，通过使用 Axios 库发送 POST 请求，将表单数据提交给服务器进行注册。

4. 文件上传：通过 `uploadfile` 方法实现文件上传功能，用户可以选择文件并将其上传到服务器。

5. 路由跳转：根据注册结果，在 `registact` 方法中使用 Vue Router 进行页面跳转，跳转到登录页面或者显示注册成功的消息。

6. 页面样式：使用 Bootstrap 和自定义的 CSS 样式，实现页面的布局和样式设置。

7. 组件引入：在 `components` 中引入了 `Htmenu` 组件，用于显示导航菜单。

8. 数据获取：在 `created` 生命周期钩子函数中，从浏览器缓存中获取用户身份和用户信息，并将其保存在组件的 `identity` 和 `userinfo` 数据中。

9. 页面初始化：在 `beforeCreate` 生命周期钩子函数中，动态加载所需的 CSS 和 JavaScript 文件，用于初始化页面样式和功能。

总体来说，Regist.vue 组件提供了一个用户注册的界面，用户可以填写注册信息并提交进行注册操作。它具有用户表单输入、数据提交、文件上传和路由跳转等功能，通过与后端进行交互实现用户注册的功能。同时，它使用了样式框架和自定义样式，使页面呈现出美观的注册界面。

如图4.23Regist.vue



图4.23 Regist.vue

#### 4.1.5.16SearchGoods.vue：

这是一个搜索商品的页面组件，主要功能是允许用户通过输入关键字搜索商品。以下是该页面的主要部分：

1. 头部区域：包括网站的logo和导航菜单。

2. 面包屑区域：显示当前页面的路径导航，用户可以通过点击链接返回上一级页面。

3. 搜索栏区域：用户可以在输入框中输入关键字，并点击搜索按钮进行商品搜索。

4. 商品展示区域：根据搜索结果，以卡片形式展示符合条件的商品，每行显示3个商品。

5. 底部区域：显示网站的版权信息和相关链接。

该页面的特点包括：

- 使用了Bootstrap框架来实现响应式布局和样式。

- 通过Vue.js组件化开发，将导航菜单封装为独立的组件Qtmenu，并在页面中引用。

- 使用axios库进行HTTP请求，通过POST请求向服务器发送搜索商品的请求，并将结果展示在页面上。

- 通过v-for指令和goodsall数组，动态生成商品展示卡片。

- 通过点击商品链接，可以跳转到商品详情页面。

总体而言，该页面提供了一个用户友好的界面，方便用户搜索和浏览商品，并且使用了现代化的前端开发技术来实现交互功能。



如图4.24SearchGoods.vue

#### 4.1.5.17ShangpinTongji.vue：

这是一个名为ShangpinTongji的Vue组件，用于展示电子商城的商品统计信息。下面是对该组件的功能、页面作用和特点的详细描述：

功能：

- 获取十强商品的数据并展示在饼图和折线图中。

- 通过接口获取商品类别统计数据，并以饼图和折线图的形式展示。

- 提供上传文件的功能，并将上传的文件保存到服务器上。

- 实现添加管理员的功能。

页面作用：

- 作为电子商城后台管理系统的一部分，用于展示商品统计信息和进行管理员管理操作。

- 在页面顶部显示电子商城的品牌标志和导航菜单。

- 在页面内容区域展示商品统计图表。

- 在页面底部显示版权信息。

特点：

- 使用Vue框架构建，采用单文件组件的方式组织代码。

- 使用axios库进行HTTP请求和数据获取。

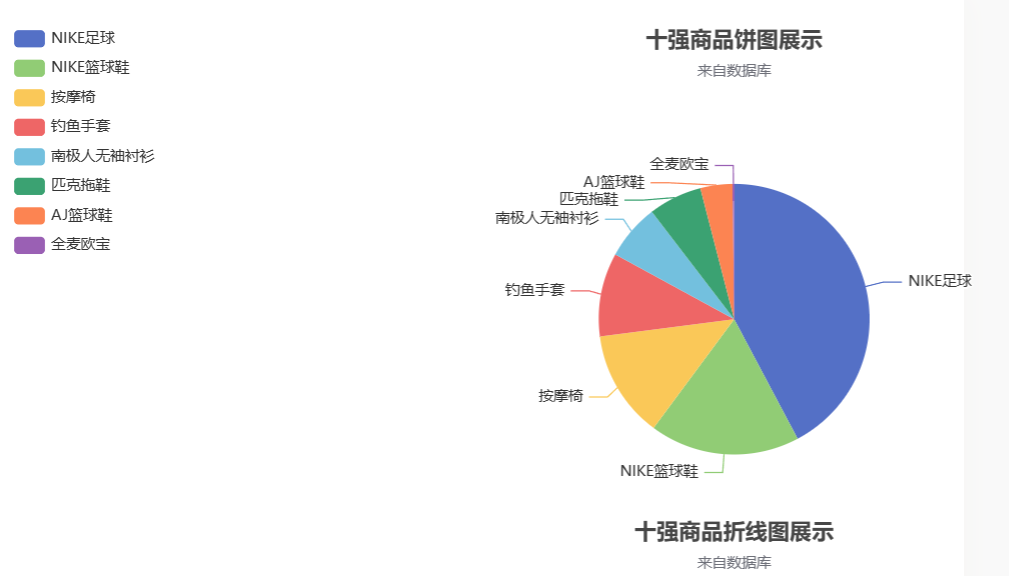
- 使用ECharts库实现饼图和折线图的展示。

- 通过调用后端接口实现数据的获取和保存。

- 使用Bootstrap和自定义的CSS样式来美化页面。

总体而言，该组件是一个用于展示商品统计信息和进行管理员管理的页面，它提供了直观的图表展示和数据操作功能，方便管理员对电子商城的运营进行分析和管理。

如图4.25ShangpinTongji.vue



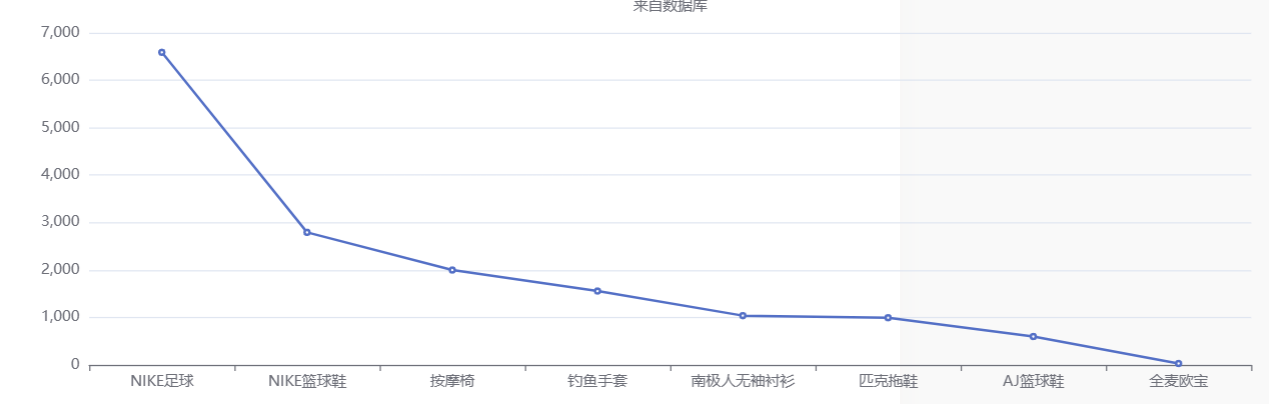


图4.25ShangpinTongji.vue

#### 4.1.5.18ShopcarManage.vue：

ShopcarManage.vue是一个购物车管理页面的Vue组件。它的主要功能是展示购物车列表，并提供搜索、删除和批量删除购物车的操作。

页面结构：

1. 页面顶部有一个导航栏，包含网站的Logo和标题。

2. 左侧是一个垂直滚动的导航菜单，显示其他功能模块的链接。

3. 页面内容部分包含一个标题和一个用于搜索的表单。

4. 下方是一个响应式的表格，展示购物车的数据。

5. 表格的每一行都有一个复选框，用于选择需要操作的购物车项。

6. 表格的每一列显示购物车的商品、用户、价格、数量和状态信息。

7. 每一行的最后一列包含三个操作按钮，分别是查看详情、修改和删除购物车项。

8. 表格的底部是一个分页器，用于切换页码和设置每页显示的数量。

功能特点：

1. 搜索功能：可以根据商品和用户进行购物车的搜索。

2. 删除功能：可以单个删除和批量删除购物车项。

3. 分页功能：可以切换购物车列表的页码和每页显示的数量。

4. 详情展示：点击详情按钮可以查看购物车项的详细信息。

5. 修改操作：点击修改按钮可以跳转到修改购物车项的页面。

总结：

ShopcarManage.vue是一个购物车管理页面，具有搜索、删除、分页和详情展示等功能，方便管理员对购物车进行管理和操作。

如图4.26ShopcarManage.vue

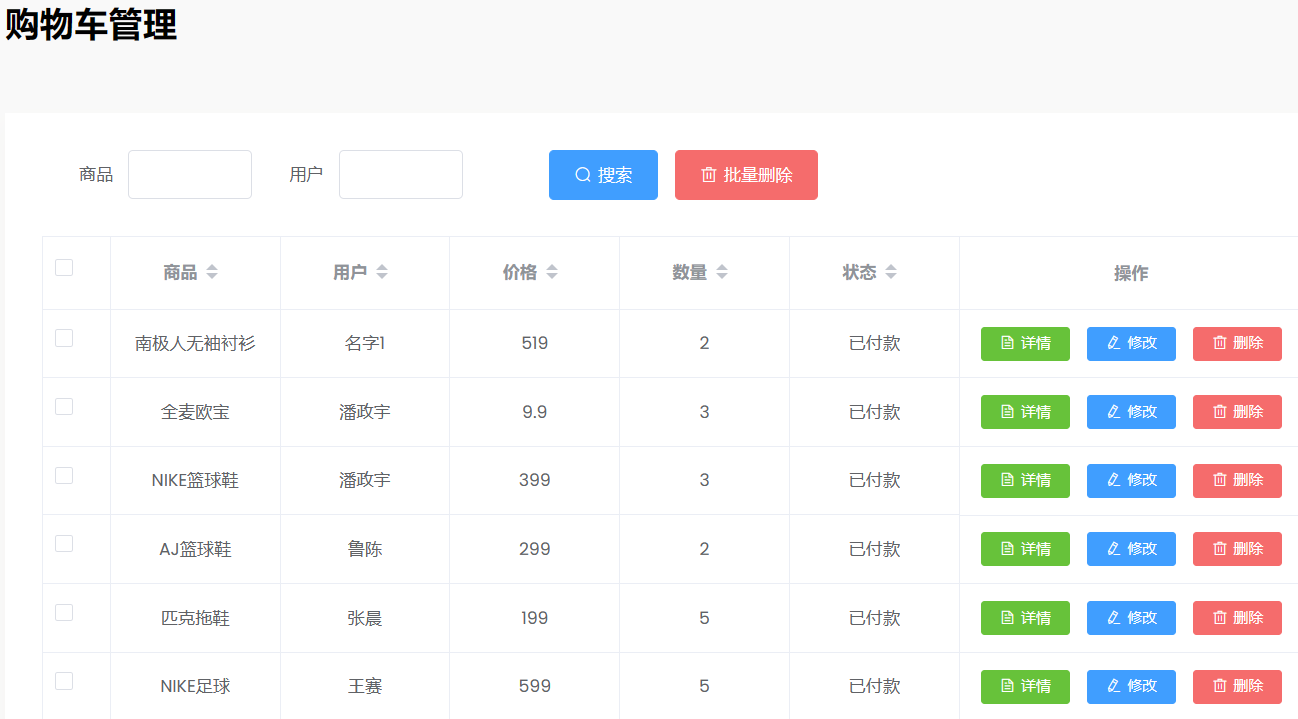


图4.26 ShopcarManage.vue

#### 4.1.5.19ShopcarView.vue：

该代码是一个Vue组件，名为"ShopcarView"，用于展示购物车信息并提供相关操作功能。以下是对其功能、页面作用和特点的详细分析：

功能：

1. 展示购物车信息：在页面上以表格形式展示购物车的各个字段，包括商品、用户、价格、数量、状态等。

2. 搜索购物车：提供搜索功能，可以根据商品和用户字段进行搜索过滤购物车信息。

3. 分页显示：根据设置的每页显示数量，将购物车信息进行分页展示，并提供翻页功能。

4. 查看购物车详情：点击表格中的详情按钮，弹出对话框显示选中购物车的详细信息。

5. 删除购物车：支持单个和批量删除购物车功能，可以选择一个或多个购物车进行删除操作。

页面作用：

该页面主要用于管理员或系统用户查看、管理和操作购物车信息。通过展示购物车信息和提供搜索、分页、删除等功能，方便用户对购物车数据进行查找、筛选和处理。

特点：

1. 使用了Element UI库：页面中使用了Element UI的组件，如表格(el-table)、分页(el-pagination)、输入框(el-input)、对话框(el-dialog)等，提供了丰富的交互和样式。

2. 响应式布局：页面采用了响应式布局，适应不同的屏幕尺寸和设备。

3. 弹窗显示详情：点击详情按钮会弹出一个对话框(el-dialog)，以清晰的方式显示选中购物车的详细信息。

4. 搜索和过滤功能：提供了根据商品和用户字段进行搜索过滤购物车信息的功能，方便用户快速找到目标数据。

5. 批量操作：支持批量选择购物车进行删除操作，提高操作效率。

6. 页面样式：使用了自定义的CSS样式和第三方库的样式，使页面具有较好的可视化效果和用户体验。

总体而言，该页面是一个功能完善的购物车管理界面，提供了丰富的操作和展示功能，便于管理员对购物车信息进行管理和处理。如图4.27ShopcarView.vue



图4.27 ShopcarView.vue

#### 4.1.5.20UpdateAdmin.vue

UpdateAdmin.vue是一个管理后台的页面组件，用于修改管理员信息。下面是该页面的主要功能、页面作用和特点的解析：

功能：

1. 修改管理员信息：通过填写表单字段，可以修改管理员的账号和密码。

2. 文件上传：支持上传文件，并将文件保存到服务器端。

页面作用：

1. 提供管理员信息修改功能：该页面提供了一个表单，管理员可以在表单中填写新的账号和密码，然后点击"修改"按钮进行修改操作。

2. 显示当前管理员信息：页面加载完成后，会根据管理员的ID获取管理员的详细信息，并将其显示在相应的表单字段中。

3. 提示修改结果：在修改操作完成后，会显示相应的提示信息，告知管理员修改结果。

特点：

1. 页面布局：使用了一种具有导航栏和侧边栏的典型管理后台布局，提供了良好的用户界面和交互体验。

2. 数据交互：通过Axios库与服务器进行数据交互，实现与后端的数据传输和处理。

3. 文件上传功能：支持管理员上传文件，并在上传完成后将文件保存到服务器中。

4. 样式定制：页面使用了自定义的样式，通过引入外部CSS文件和JavaScript库，实现了页面的样式和交互效果。

总结：

UpdateAdmin.vue是管理后台中负责修改管理员信息的页面组件。它提供了管理员信息的展示和修改功能，并支持文件上传操作。页面采用了典型的管理后台布局，通过与服务器的数据交互，实现了管理员信息的修改和保存。同时，通过样式定制和组件的使用，为管理员提供了良好的用户界面和交互体验。

如图4.28 UpdateAdmin.vue

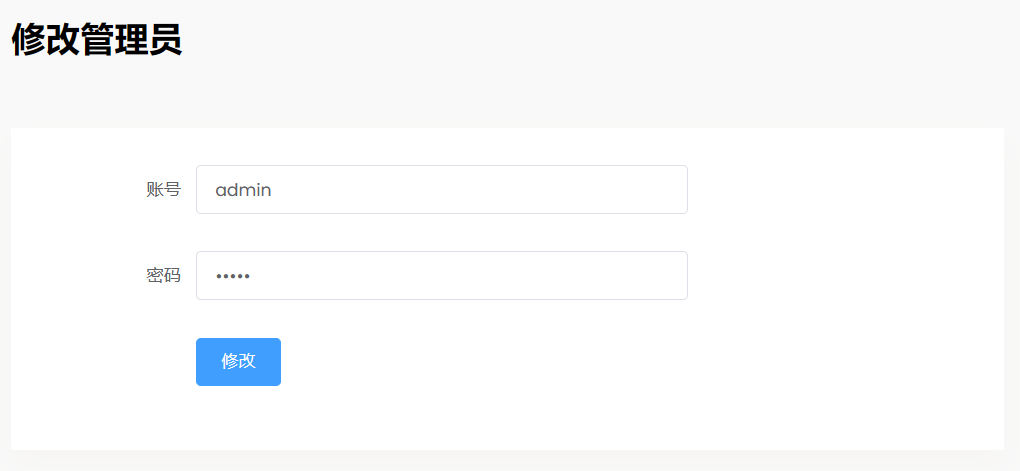


图4.28 UpdateAdmin.vue

#### 4.1.5.21UpdateGoods.vue

这是一个名为"UpdateGoods"的Vue组件，用于修改商品信息的页面。

该页面包含以下主要功能和特点：

1. 页面布局：页面由顶部导航栏、侧边栏菜单和内容区域组成。

2. 导航栏：顶部导航栏显示网站名称"电子商城"，点击可以返回首页。

3. 侧边栏菜单：通过导航菜单可以快速访问其他页面，其中使用了"Htmenu"组件来生成菜单项。

4. 商品信息表单：页面中间部分是一个表单，用于展示和修改商品的各个属性。

- 表单中使用了Element UI库的组件，如`el-form`、`el-input`、`el-upload`、`el-input-number`等，用于输入和展示商品的名称、照片、价格等信息。

- 商品照片可以通过上传、预览、下载和删除操作进行管理，使用了`el-upload`和`el-dialog`组件来实现。

5. 修改商品功能：页面底部有一个"修改"按钮，点击后将触发`updategoodsact`方法，将修改后的商品信息发送到服务器进行更新操作。

6. 其他功能：

- 页面加载时，通过调用`goodsdetailsjson`方法获取要修改的商品信息并显示在表单中。

- 在上传照片成功或失败时，会显示相应的提示信息。

总体而言，该页面提供了一个用户界面，用于修改商品信息，并通过与服务器的交互实现商品更新功能。如图4.29 UpdateGoods.vue

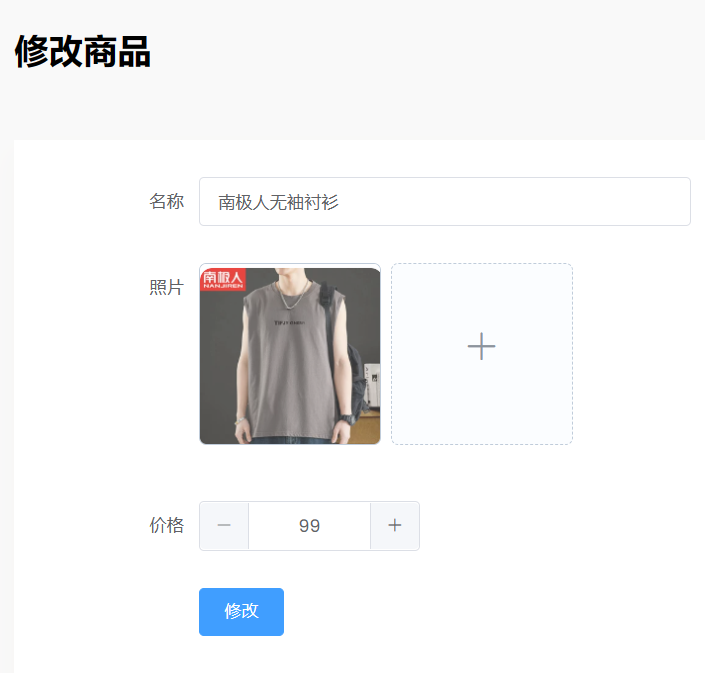


图4.29 UpdateGoods.vue

#### 4.1.5.22UpdateShopcar.vue

UpdateShopcar.vue是一个Vue组件，用于修改购物车信息的页面。以下是对该组件的功能、页面作用和特点的分析：

功能：

1. 展示当前购物车的信息，包括商品、用户、价格、数量和状态。

2. 可以选择修改商品、用户、价格、数量和状态。

3. 提供修改按钮，点击后可以将修改后的购物车信息提交到后端进行更新。

4. 可以上传文件，并将上传后的文件路径保存到购物车信息中。

页面作用：

UpdateShopcar.vue页面的主要作用是允许用户修改购物车的相关信息。用户可以通过表单输入或选择框选择需要修改的字段，然后点击修改按钮将修改后的信息提交给后端进行处理。

特点：

1. 使用了Element UI库中的表单组件（如el-form、el-input、el-select等）来构建表单界面，具有良好的交互性和可定制性。

2. 通过调用后端API接口实现数据的获取和更新，使用了axios库进行网络请求。

3. 页面使用了自定义的Htmenu组件作为导航栏，提供了用户导航功能。

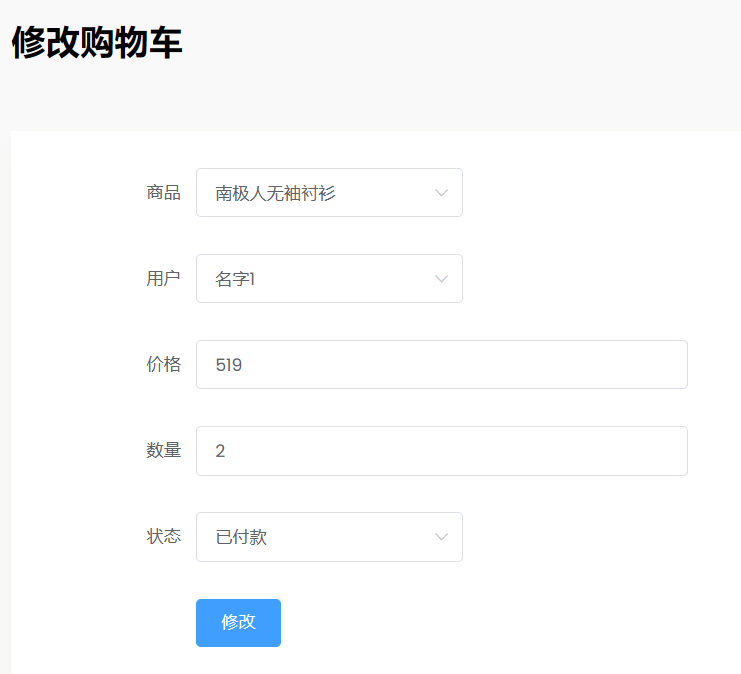
4. 通过调用后端API接口获取商品和用户的信息，并以下拉框的形式展示和选择。

5. 使用了Bootstrap和自定义的CSS样式来美化页面，使其具有良好的视觉效果。

总结：

UpdateShopcar.vue是一个用于修改购物车信息的Vue组件，提供了丰富的交互功能和界面元素，通过调用后端API实现数据的获取和更新。它具有良好的可扩展性和可定制性，可以根据实际需求进行功能的扩展和样式的定制。

如图4.30 UpdateShopcar.vue

如图4.30 UpdateShopcar.vue

#### 4.1.5.23UpdateUser.vue：

UpdateUser.vue是一个Vue组件，用于修改用户信息的页面。它包含了用户的各种信息字段，如名字、账号、密码、性别、照片和地址等。

该页面的特点和功能如下：

1. 页面布局：页面由顶部导航栏、侧边栏菜单、内容区和底部版权信息组成，整体布局简洁明了。

2. 表单提交：页面包含一个表单，用户可以在表单中修改用户的各种信息。

3. 图片上传：用户可以通过图片上传功能上传照片，并实时预览已上传的照片。用户可以移除、预览和下载已上传的照片。

4. 数据获取：页面在创建时通过发送异步请求获取指定用户的详细信息，并将信息填充到表单中进行展示。

5. 数据更新：用户点击修改按钮后，页面将获取表单中的数据，包括用户的各种信息和照片，通过异步请求将修改后的数据提交到服务器进行更新。

6. 页面跳转：更新成功后，页面将跳转到用户管理页面。

总结：UpdateUser.vue是一个用于修改用户信息的页面，具有用户信息展示、编辑和更新的功能。通过表单和图片上传组件实现了用户信息的修改和照片的上传功能，提供了良好的用户体验。如图4.31 UpdateUser.vue

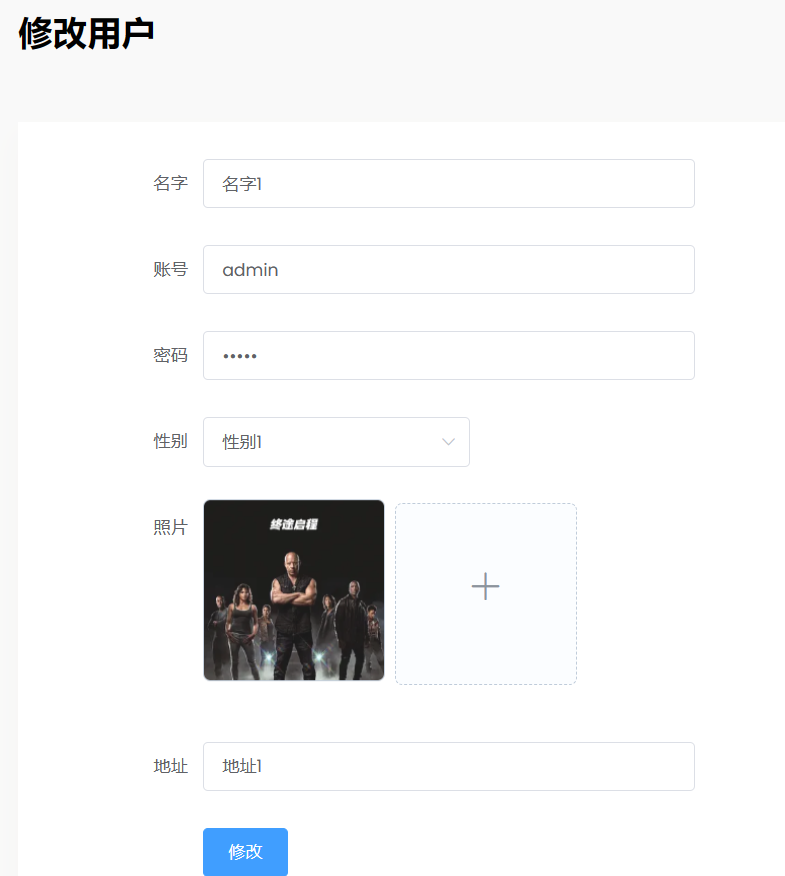


图4.31 UpdateUser.vue

#### 4.1.5.24UserAddShopcar.vue：

UserAddShopcar.vue是一个Vue组件，用于在电子商城网站中添加购物车功能的页面。以下是对其功能、页面作用和特点的详细分析：

功能：

1. 添加购物车：用户可以选择商品、填写数量和状态，然后点击添加按钮将商品添加到购物车中。

2. 商品选择：用户可以从商品列表中选择商品，并根据选择的商品动态显示对应的价格。

3. 文件上传：支持用户上传文件，用于商品图片等信息的展示。

4. 时间获取：提供了获取当前时间的函数。

5. 表单序列化：提供了将表单数据序列化为对象的方法。

6. 页面样式：使用了Bootstrap和自定义的样式文件，使页面具有美观的外观。

页面作用：

1. 显示网站的品牌logo和导航菜单，提供用户返回首页和导航功能。

2. 提供一个表单，用于用户输入购物车相关信息，如商品、数量和状态。

3. 提供商品选择功能，包括选择商品和显示对应价格。

4. 显示当前用户的身份和用户信息。

5. 提供添加购物车的按钮，点击后将表单数据提交到后端进行处理。

特点：

1. 使用Vue框架开发，结构清晰，易于维护和扩展。

2. 使用了Element UI组件库，提供了丰富的表单元素和交互组件，增强了用户体验。

3. 采用了前后端分离的开发方式，通过Ajax请求与后端进行数据交互。

4. 使用了Bootstrap和自定义样式文件，使页面具有响应式布局和美观的样式。

5. 页面中的数据通过双向数据绑定实现与用户输入的交互，使用户操作和数据同步更新。

6. 通过调用后端接口实现数据的获取和提交，实现了与后端的数据交互功能。

总之，UserAddShopcar.vue是一个用于添加购物车功能的页面，通过表单和交互组件实现用户输入购物车信息，并与后端进行数据交互，具有良好的用户体验和可扩展性。如图4.32 UserAddShopcar.vue



图4.32 UserAddShopcar.vue

#### 4.1.5.25UserIndex.vue：

这是一个名为UserIndex的Vue组件，用于展示用户个人中心页面。下面是对其功能、页面作用和特点的分析：

功能：

- 显示用户个人信息：包括用户ID、姓名、账号、密码、性别、照片、地址等信息。

- 修改用户个人信息：用户可以修改姓名、密码、性别、照片和地址等信息，并保存修改后的内容。

- 上传、预览和删除照片：用户可以上传个人照片，并在页面上进行预览和删除操作。

页面作用：

- 展示用户个人信息：通过表单控件将用户的个人信息显示在页面上，包括文本输入框、下拉选择框和上传照片组件等。

- 提供修改功能：用户可以修改个人信息，并通过点击"修改"按钮将修改后的信息保存到服务器。

- 支持照片上传：用户可以通过上传照片功能上传个人照片，并在页面上进行预览、下载和删除操作。

特点：

- 使用Element-UI库：页面中使用了Element-UI的组件，如el-form、el-input、el-select和el-upload等，提供了丰富的表单和上传功能。

- 采用响应式布局：页面使用了Bootstrap的栅格系统，具有响应式布局，适应不同屏幕尺寸的设备。

- 实时更新UI：通过监听数据的变化，可以实时更新UI界面，例如上传照片成功后会显示在页面上的照片列表中。

总结：

该页面是一个用户个人中心页面，提供了展示和修改用户个人信息的功能，同时支持上传、预览和删除照片。使用了Element-UI库和响应式布局，具有良好的用户界面和交互体验。

如图4.33 UserIndex.vue

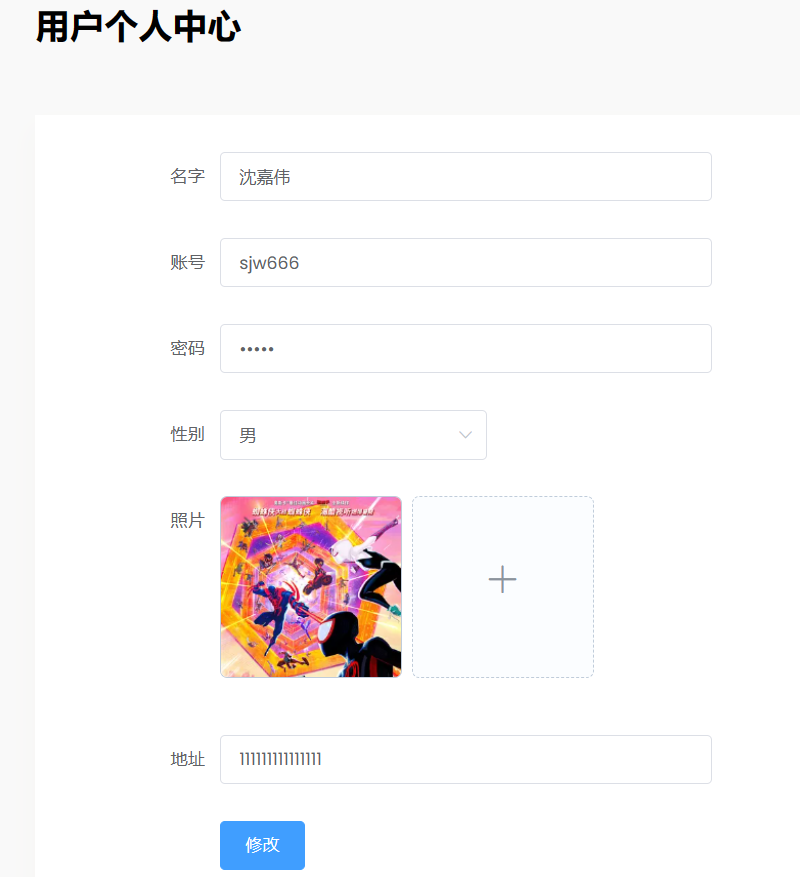


图4.33 UserIndex.vue

#### 4.1.5.26UserManage.vue：

该页面是一个用户管理页面，用于管理电子商城的用户信息。以下是该页面的主要功能和特点：

1. 页面布局：页面采用了响应式布局，适应不同屏幕尺寸的设备。页面包含顶部导航栏、侧边栏菜单、用户信息表格和分页器。

2. 用户搜索：页面提供了一个搜索表单，可以根据用户的名字、账号和密码进行用户搜索。

3. 用户信息表格：使用`el-table`组件展示用户信息。表格包含多列，包括名字、账号、密码、性别、照片、地址和操作等字段。表格支持排序、分页和多选功能。

4. 用户详情对话框：点击表格中的"详情"按钮可以打开用户详情对话框，展示选定用户的详细信息，包括名字、账号、密码、性别、照片和地址等字段。

5. 批量删除：页面提供了一个"批量删除"按钮，用于同时删除选中的多个用户。

6. 分页器：页面底部的分页器组件`el-pagination`用于切换不同的页码和调整每页显示的用户数量。

7. 文件上传：页面中的文件上传功能可以上传用户照片。

8. 样式和脚本引入：页面引入了一些样式和脚本文件，用于美化页面和实现一些交互功能。

总结：该页面提供了对用户信息的管理功能，包括搜索、查看详情、修改和删除用户等操作。同时，页面具有良好的交互性和响应式布局，适用于不同屏幕尺寸的设备。

#### 4.1.5.27UserOrderManage.vue：

该组件名为UserOrderManage.vue，是一个用于管理购物车的页面模板。

功能：

- 显示购物车列表：通过调用接口获取购物车数据，将数据展示在表格中，包括商品信息、用户信息、价格、数量、状态等。

- 搜索购物车：可以根据商品和用户进行搜索，输入关键词后点击搜索按钮即可筛选符合条件的购物车项。

- 批量删除购物车：可以选择多个购物车项，点击批量删除按钮进行删除操作。

- 查看购物车详情：点击详情按钮可以查看购物车项的详细信息，包括商品、用户、价格、数量和状态等。

- 分页功能：支持分页显示购物车列表，可以设置每页显示的数量，通过切换页码和每页显示数量来进行导航。

页面详细作用：

- 页面顶部有导航栏和品牌logo，点击logo可以返回首页。

- 页面左侧有导航菜单栏，展示其他页面链接，可以快速切换到其他功能页面。

- 页面中间部分是购物车管理的主要内容，包括搜索框、购物车列表、分页器。

- 每个购物车项包括商品信息、用户信息、价格、数量、状态和操作按钮。

- 点击操作按钮中的详情按钮可以弹出购物车项的详细信息对话框，显示商品、用户、价格、数量和状态。

- 右下角有页脚版权信息。

特点：

- 使用了Element UI组件库中的表格、分页器、对话框等组件，提供了美观和易用的界面。

- 通过Axios库发送HTTP请求，与后端进行数据交互。

- 实现了批量删除、搜索、分页等功能，提高了用户的操作效率。

- 使用了样式文件和脚本文件，实现了页面的样式和交互效果。

- 采用了Vue.js框架，通过组件化的方式构建页面，提高了代码的可维护性和复用性。

请注意，由于提供的代码片段中存在部分无法解析的内容（如htmenu组件、axios请求等），因此无法确切了解整个页面的完整功能和特点。以上回答仅基于提供的代码片段进行分析。如需更详细的分析，请提供完整的代码或提供更多相关信息。如图4.34 UserOrderManage.vue



图4.34 UserOrderManage.vue

#### 4.1.5.28UserShopcarManage.vue：

UserShopcarManage.vue是一个购物车管理页面的Vue组件。它包含了购物车列表、搜索功能、分页功能以及一些操作按钮。

该组件的主要功能有：

1. 展示购物车列表：通过el-table组件展示购物车的商品、用户、价格、数量、状态等信息。

2. 搜索功能：用户可以根据商品和用户进行搜索过滤购物车列表。

3. 批量删除购物车：用户可以选择多个购物车项进行批量删除操作。

4. 查看购物车详情：用户可以点击详情按钮查看购物车中某个商品的详细信息。

5. 结算购物车：用户可以点击结算按钮对购物车中的商品进行结算操作。

6. 单个购物车删除：用户可以点击删除按钮删除购物车中的某个商品。

7. 分页功能：展示购物车列表时进行分页展示，用户可以选择每页显示的数量并进行翻页。

页面特点：

1. 使用了element-ui组件库中的el-table、el-pagination、el-button、el-input、el-dialog等组件实现页面布局和交互功能。

2. 通过axios库进行与后端的数据交互，包括获取购物车列表、搜索购物车、删除购物车等操作。

3. 使用了vue-router实现页面的路由导航。

4. 通过sessionStorage存储和获取用户身份信息和用户信息。

总体来说，UserShopcarManage.vue是一个功能完善的购物车管理页面，用户可以对购物车进行搜索、删除、查看详情和结算等操作，方便管理和操作购物车中的商品信息。如图4.35 UserShopcarManage.vue



图4.35 UserShopcarManage.vue

#### 4.1.5.29UserUpdateShopcar.vue：

根据所提供的代码，这是一个用于修改购物车信息的Vue组件。以下是对代码功能、页面作用和特点的详细分析：

功能：

1. 显示顶部导航栏，包括电子商城的品牌标志和链接。

2. 显示侧边栏菜单，包含在`Htmenu`组件中。

3. 提供修改购物车信息的表单，包括商品选择、价格、数量和状态等字段。

4. 通过请求后端接口，获取商品列表和指定购物车的详细信息。

5. 支持上传文件并获取文件路径。

6. 实现修改购物车信息的功能。

页面作用：

该页面用于管理员或用户修改购物车的信息，包括商品、价格、数量和状态等字段。用户可以选择商品、输入价格和数量，并可以选择购物车的状态。在点击修改按钮后，将向后端发送请求，更新购物车信息，并给出相应的提示。

特点：

1. 使用了Element UI库中的组件，如`el-form`、`el-select`和`el-input`，实现了表单的布局和交互。

2. 通过Vue Router的`$route`对象获取路由参数，如购物车ID和商品ID，以便在页面加载时进行相关数据的查询和展示。

3. 使用了axios库发送异步请求，与后端进行数据交互。

4. 页面样式使用了自定义的CSS文件和第三方的CSS样式库，实现了整体的样式美化。

总结：

该页面是一个用Vue编写的购物车信息修改页面，提供了用户友好的界面和交互体验。它通过与后端接口的数据交互，实现了修改购物车信息的功能，方便管理员或用户对购物车信息进行修改操作。

#### 4.1.5.30UserView.vue

这是一个用户查看页面的Vue组件，主要用于展示和搜索用户信息。下面是该页面的主要功能和特点：

功能：

1. 显示用户列表：页面通过`el-table`组件展示用户信息，包括名字、账号、密码、性别、照片、地址等字段。

2. 分页功能：使用`el-pagination`组件实现用户列表的分页功能，可以选择每页显示的数量，并可以通过页码进行切换。

3. 搜索用户：通过输入名字、账号、密码等条件，在表格上方的搜索框中进行搜索，点击搜索按钮可以根据条件过滤用户列表。

特点：

1. 响应式布局：页面使用Bootstrap框架实现了响应式布局，在不同的设备上都能提供良好的用户体验。

2. 用户详情弹窗：点击用户列表中的详情按钮，弹出一个对话框显示选定用户的详细信息，包括名字、账号、密码、性别、照片和地址。

3. 删除用户：可以选择用户列表中的一个或多个用户，点击删除按钮进行批量删除操作，弹出确认对话框确认删除操作。

总体而言，该页面是一个简单的用户管理页面，提供了用户信息的展示、搜索和删除功能，以及用户详情的查看功能。如图4.36 UserView.vue



图4.36 UserView.vue

#### 4.1.5.31YonghuTongji.vue

这是一个名为"YonghuTongji"的Vue组件，用于展示十强用户的统计信息。下面是对该组件的功能、页面作用和特点的详细描述：

功能：

- 显示页面顶部的电子商城品牌标志和导航菜单。

- 在内容区域展示十强用户的统计数据，包括饼图和折线图。

- 提供上传文件的功能。

- 提供添加十强用户的功能。

页面作用：

- 展示电子商城品牌标志和导航菜单，使用户能够浏览网站的其他页面。

- 在内容区域以图表的形式展示十强用户的统计数据，便于用户了解用户分类和数量分布情况。

- 提供上传文件的功能，可能用于导入用户数据或其他文件的操作。

- 提供添加十强用户的功能，允许管理员向系统添加新的十强用户。

特点：

- 使用了Bootstrap框架和自定义的CSS样式，使页面具有良好的外观和响应式布局。

- 使用了Echarts图表库来绘制饼图和折线图，以直观的方式展示统计数据。

- 通过Ajax请求与后端服务器进行数据交互，实现数据的获取和提交操作。

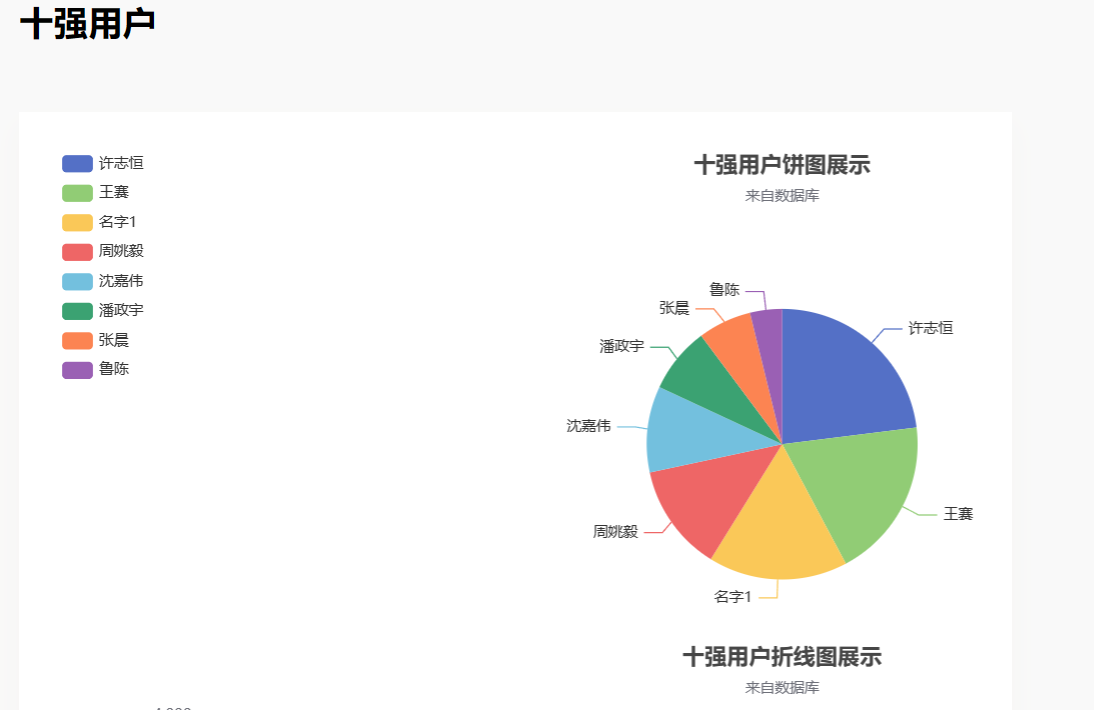
- 利用Vue组件化开发的方式，将页面拆分成多个组件，提高了代码的可维护性和复用性。

- 使用了Vue Router进行页面导航和路由管理。

- 通过Axios库发送HTTP请求，实现与后端接口的通信。

总体而言，这个组件是一个管理系统中展示和操作十强用户统计数据的页面，通过图表和表单交互等方式，使管理员能够方便地查看和管理用户数据。

如图4.37 YonghuTongji.vue



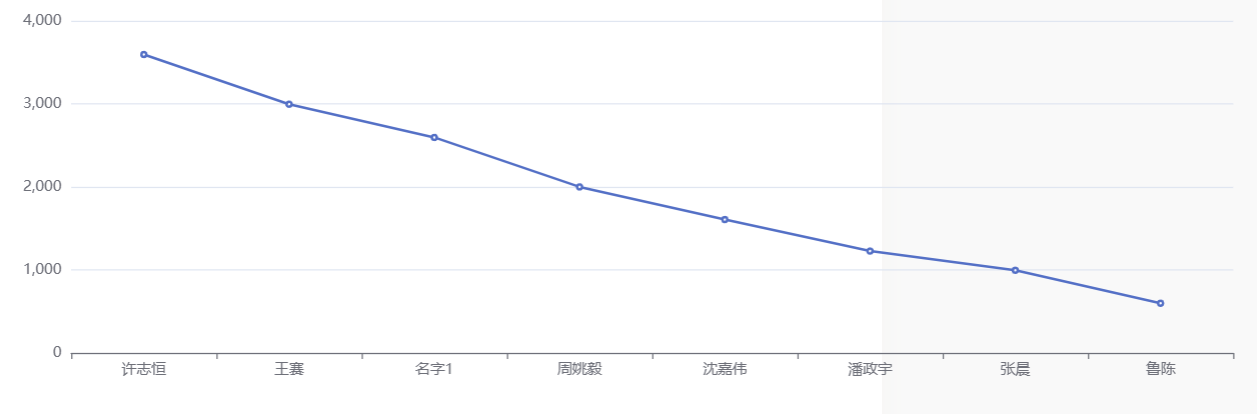


图4.37 YonghuTongji.vue

### 4.1.6 static文件夹：

- 该文件夹用于存储静态文件，例如图像、样式表和JavaScript文件等。这些文件在前端应用中被引用或加载，提供了项目所需的静态资源。

如图4.38 static文件夹

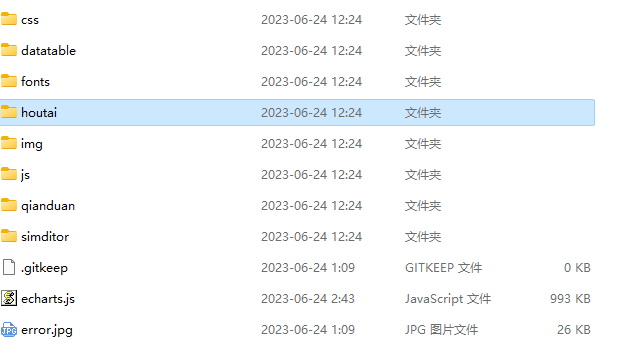


图4.38 static文件夹

## 4.2 系统功能实现

系统运行的截图及说明

1. 首页为商品展示页面，不需要登录。

### 4.2.1首页展示：

如图4.39 首页展示





图4.39 首页展示

这是一个搜索商品的主页，具体功能如下：

1. 在页面顶部有一个导航栏，包含网站的logo和菜单项。

2. 页面顶部下方有一个面包屑导航，显示当前所在位置。

3. 页面中间是一个搜索栏，可以输入关键字进行商品搜索。点击搜索按钮可以触发搜索功能。

4. 页面下方是展示商品的区域，使用了循环（v-for）来遍历商品列表（goodsall），每个商品以图片、商品名称和详情按钮的形式展示。

5. 页面底部是网站的页脚部分，包含网站的logo和版权信息。

6. 页面使用了一些CSS和JavaScript库，用于样式和交互效果的实现。

页面的特点：

- 页面布局简洁清晰，采用了响应式设计，适应不同屏幕尺寸。

- 使用了Vue.js框架，通过组件化的方式管理页面的各个部分。

- 通过AJAX请求获取商品数据，实现动态展示和搜索功能。

- 使用了一些第三方库和插件，增加了页面的样式和交互效果。

总体而言，该页面提供了一个方便用户搜索商品的界面，展示了搜索结果并支持用户查看商品详情。

### 4.2.2注册登录功能展示：

#### 4.2.2.1用户注册展示:

如用户注册展示



用户注册展示

Regist.vue 是一个注册页面组件，用于用户进行注册操作。

该组件具有以下功能和特点：

1. 注册表单：通过 `<form>` 元素创建一个注册表单，其中包含账号、密码、确认密码和身份等输入字段。

2. 注册按钮：通过 `<button>` 元素创建一个注册按钮，点击该按钮触发 `registact` 方法进行注册操作。

3. 注册请求：在 `registact` 方法中，通过使用 Axios 库发送 POST 请求，将表单数据提交给服务器进行注册。

4. 文件上传：通过 `uploadfile` 方法实现文件上传功能，用户可以选择文件并将其上传到服务器。

5. 路由跳转：根据注册结果，在 `registact` 方法中使用 Vue Router 进行页面跳转，跳转到登录页面或者显示注册成功的消息。

总体来说，Regist.vue 组件提供了一个用户注册的界面，用户可以填写注册信息并提交进行注册操作。它具有用户表单输入、数据提交、文件上传和路由跳转等功能，通过与后端进行交互实现用户注册的功能。同时，它使用了样式框架和自定义样式，使页面呈现出美观的注册界面。

注册成功展示：

如图4.41 注册成功展示



如图4.41 注册成功展示

#### 4.2.2.2用户/管理员登陆展示：

如图4.42 用户/管理员登陆展示



图4.42 用户/管理员登陆展示

Login.vue 组件提供了用户登录的界面和相关功能，包括表单输入、身份选择、文件上传、数据处理和页面跳转等。它使用了现代的前端技术和框架，使用户可以方便地进行登录操作，并根据登录结果进行相应的处理。

登陆成功展示：

如图4.43 登陆成功展示



图4.43 登陆成功展示

### 4.2.3管理员功能展示：

##### 4.2.3.1管理商品信息功能

如图4.44管理商品信息功能

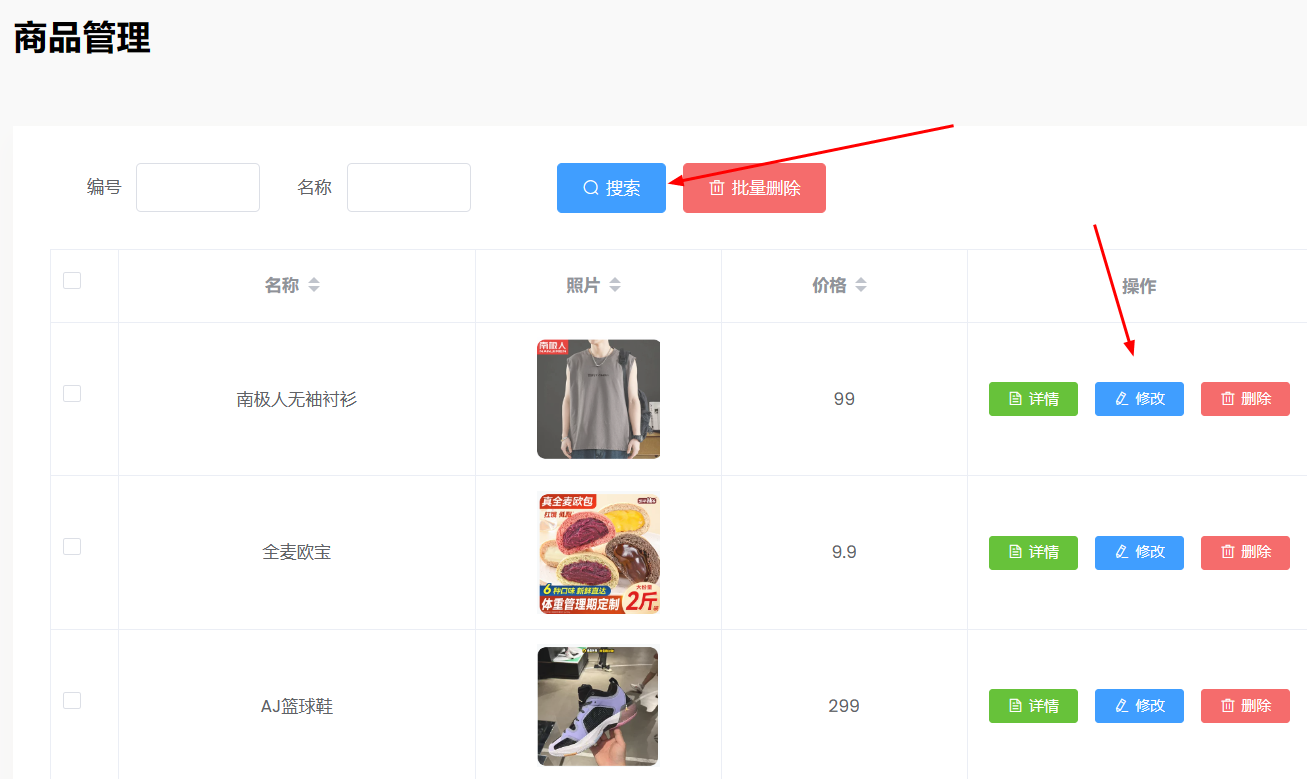


图4.44管理商品信息功能

功能：

- 该组件用于管理商品，包括显示、搜索、删除、修改商品信息。

- 它提供了表格形式展示商品列表，可以进行多选删除和分页功能。

- 点击详情按钮可以查看商品的详细信息，点击修改按钮可以跳转到修改商品页面。

##### 4.2.3.2管理用户功能

如图 4.45管理用户功能

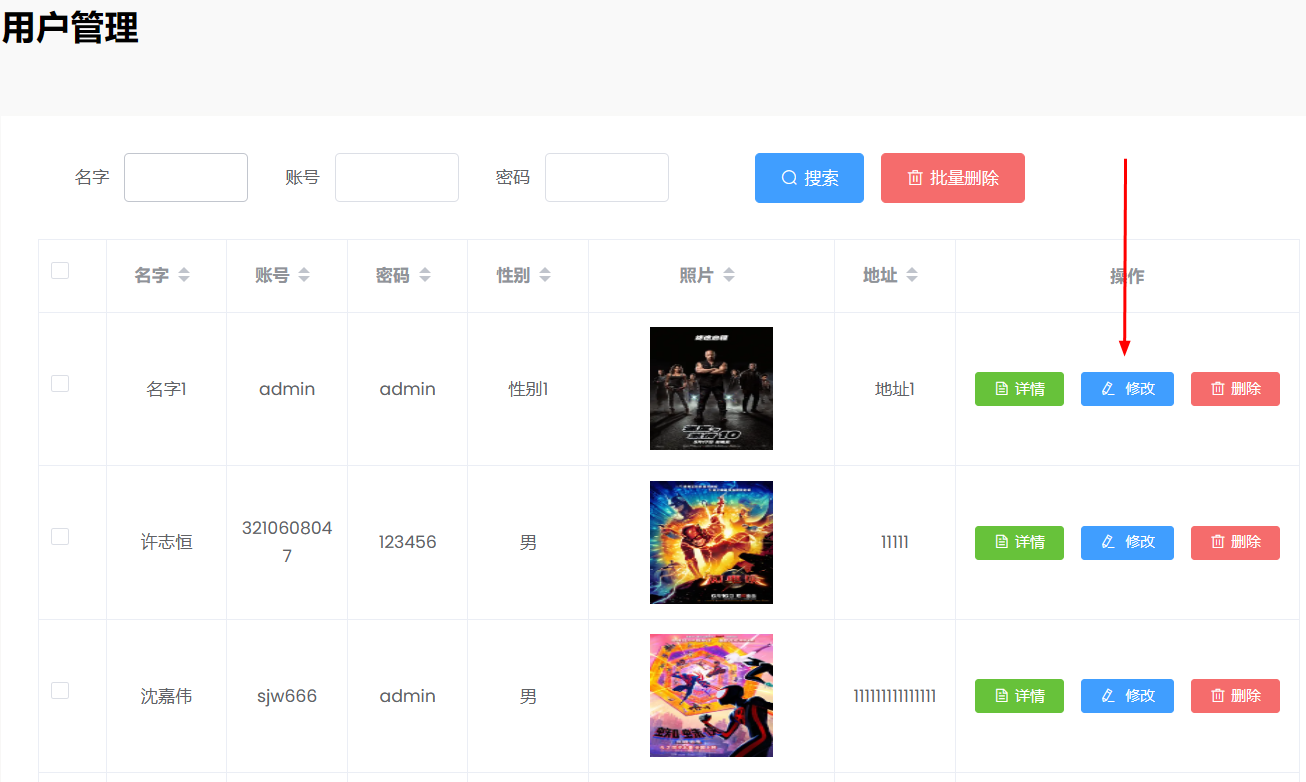


图 4.45管理用户功能

该页面是一个用户管理页面，用于管理电子商城的用户信息。以下是该页面的主要功能和特点：

1. 页面布局：页面采用了响应式布局，适应不同屏幕尺寸的设备。页面包含顶部导航栏、侧边栏菜单、用户信息表格和分页器。

2. 用户搜索：页面提供了一个搜索表单，可以根据用户的名字、账号和密码进行用户搜索。

3. 用户信息表格：使用`el-table`组件展示用户信息。表格包含多列，包括名字、账号、密码、性别、照片、地址和操作等字段。表格支持排序、分页和多选功能。

4. 用户详情对话框：点击表格中的"详情"按钮可以打开用户详情对话框，展示选定用户的详细信息，包括名字、账号、密码、性别、照片和地址等字段。

5. 批量删除：页面提供了一个"批量删除"按钮，用于同时删除选中的多个用户。

6. 分页器：页面底部的分页器组件`el-pagination`用于切换不同的页码和调整每页显示的用户数量。

7. 文件上传：页面中的文件上传功能可以上传用户照片。

##### 4.2.3.3查看订单/购物车功能

如图4.46 查看订单/购物车功能

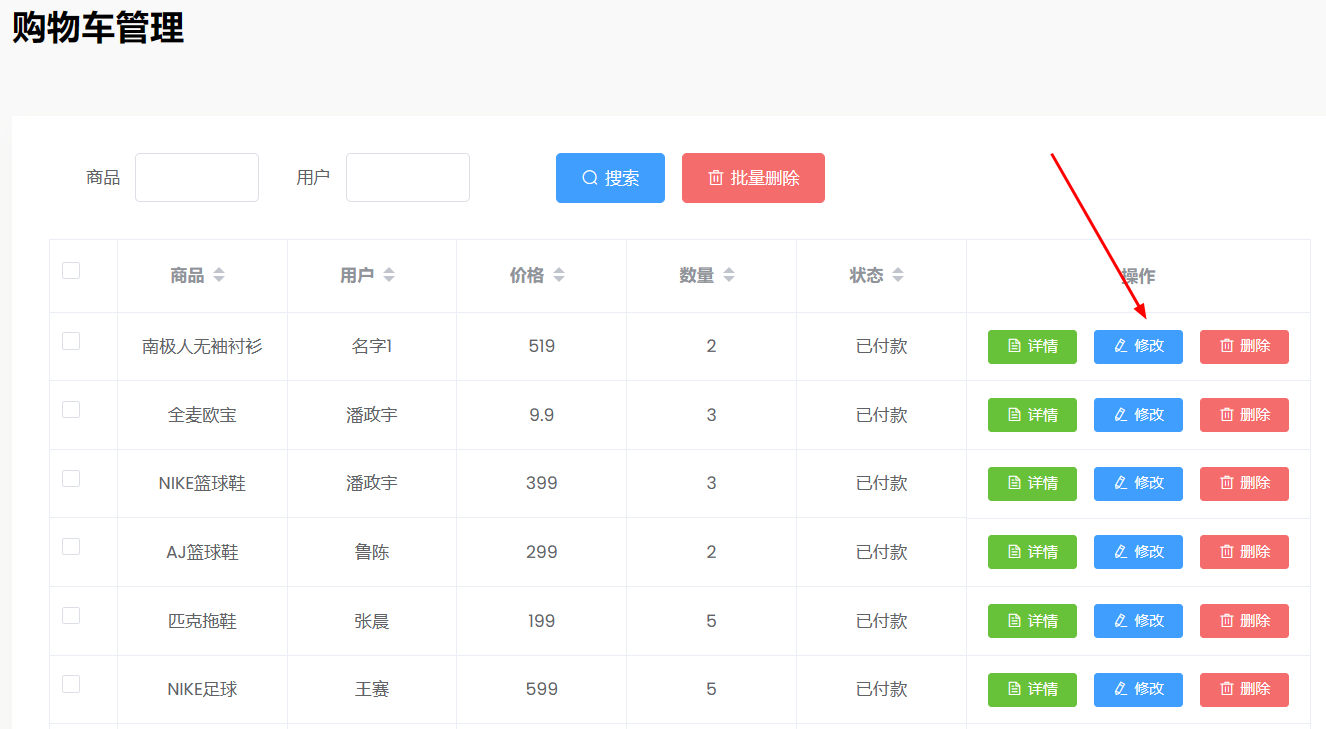


图4.46 查看订单/购物车功能

ShopcarManage.vue是一个购物车管理页面的Vue组件。它的主要功能是展示购物车列表，并提供搜索、删除和批量删除购物车的操作。

功能特点：

1. 搜索功能：可以根据商品和用户进行购物车的搜索。

2. 删除功能：可以单个删除和批量删除购物车项。

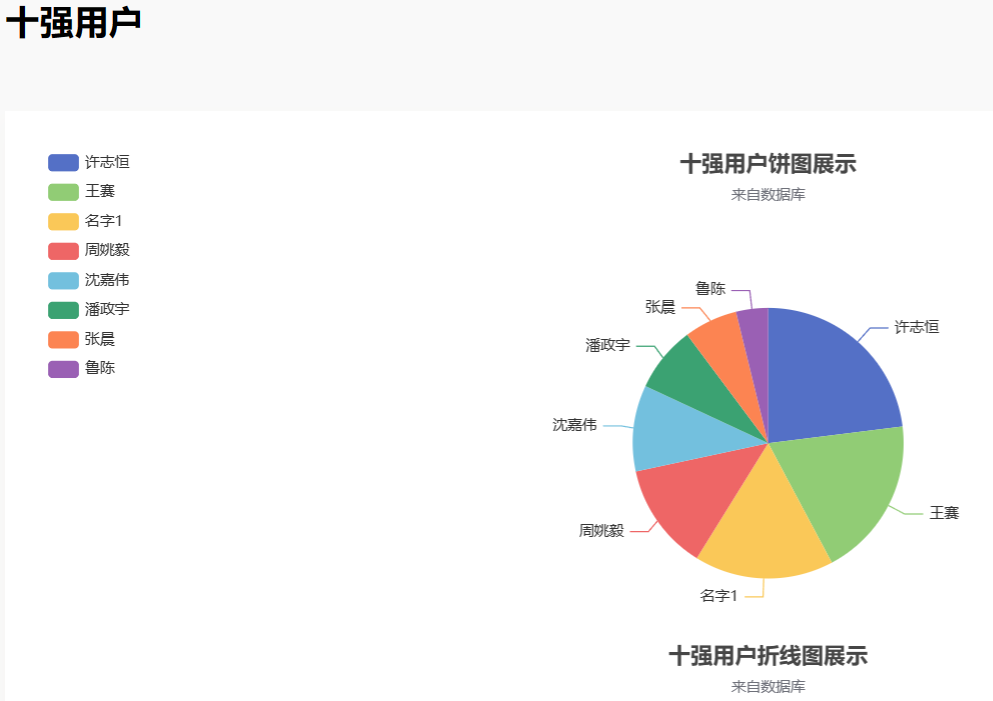
3. 分页功能：可以切换购物车列表的页码和每页显示的数量。

4. 详情展示：点击详情按钮可以查看购物车项的详细信息。

5. 修改操作：点击修改按钮可以跳转到修改购物车项的页面。

##### 4.2.3.4统计十强用户展示

如图 4.47统计十强用户展示



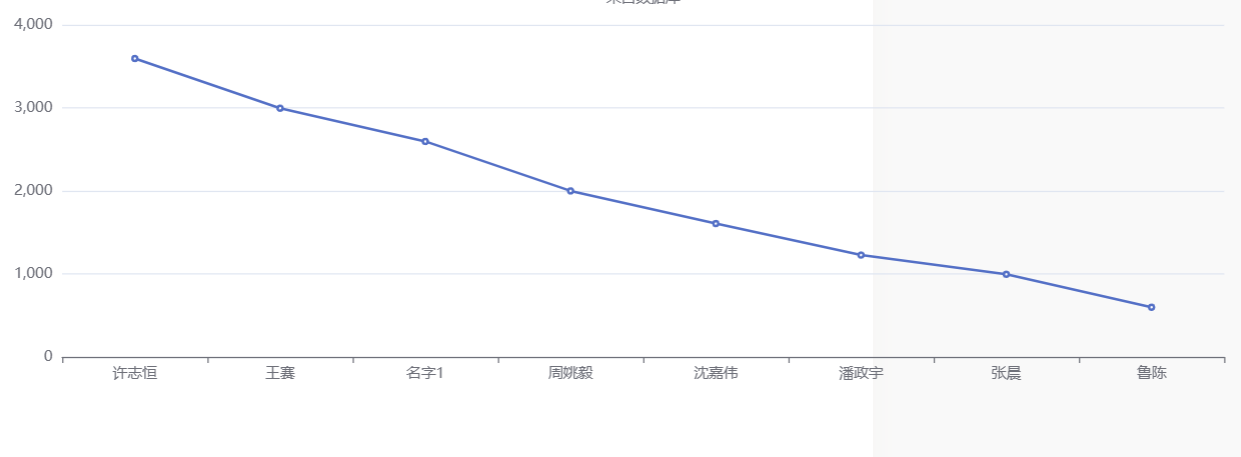


图 4.47统计十强用户展示

这是一个名为"YonghuTongji"的Vue组件，用于展示十强用户的统计信息。下面是对该组件的功能、页面作用和特点的详细描述：

功能：

- 显示页面顶部的电子商城品牌标志和导航菜单。

- 在内容区域展示十强用户的统计数据，包括饼图和折线图。

- 提供上传文件的功能。

- 提供添加十强用户的功能。

页面作用：

- 展示电子商城品牌标志和导航菜单，使用户能够浏览网站的其他页面。

- 在内容区域以图表的形式展示十强用户的统计数据，便于用户了解用户分类和数量分布情况。

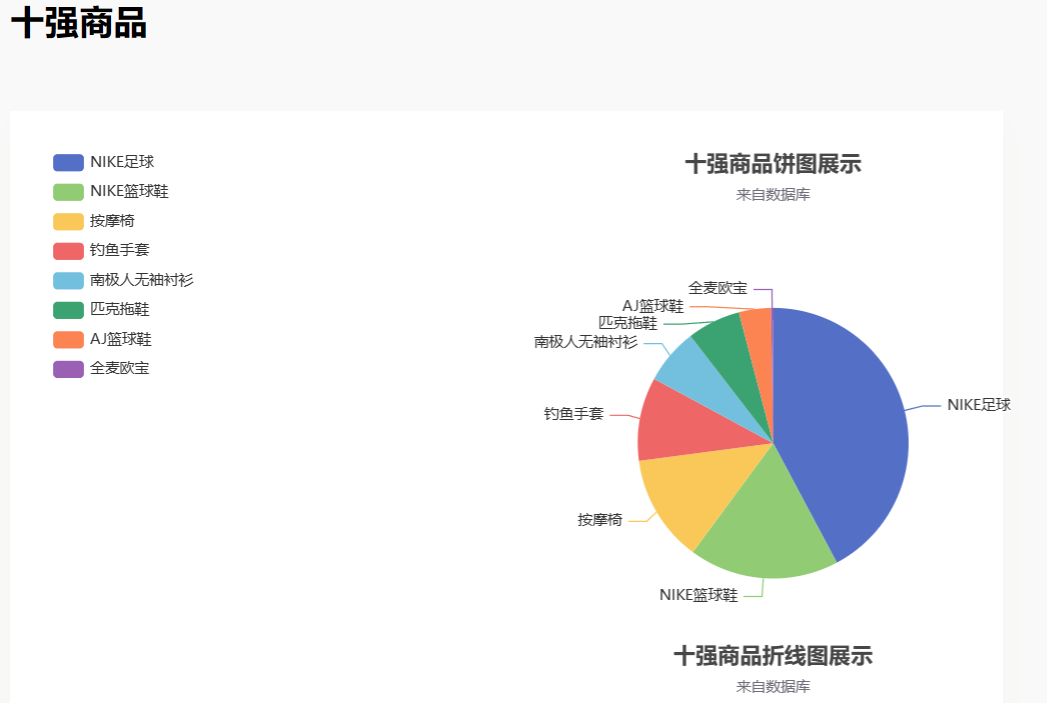
- 提供上传文件的功能，可能用于导入用户数据或其他文件的操作。

- 提供添加十强用户的功能，允许管理员向系统添加新的十强用户。

总体而言，这个组件是一个管理系统中展示和操作十强用户统计数据的页面，通过图表和表单交互等方式，使管理员能够方便地查看和管理用户数据。

##### 4.2.3.5统计十强商品展示

如图4.48 统计十强商品展示



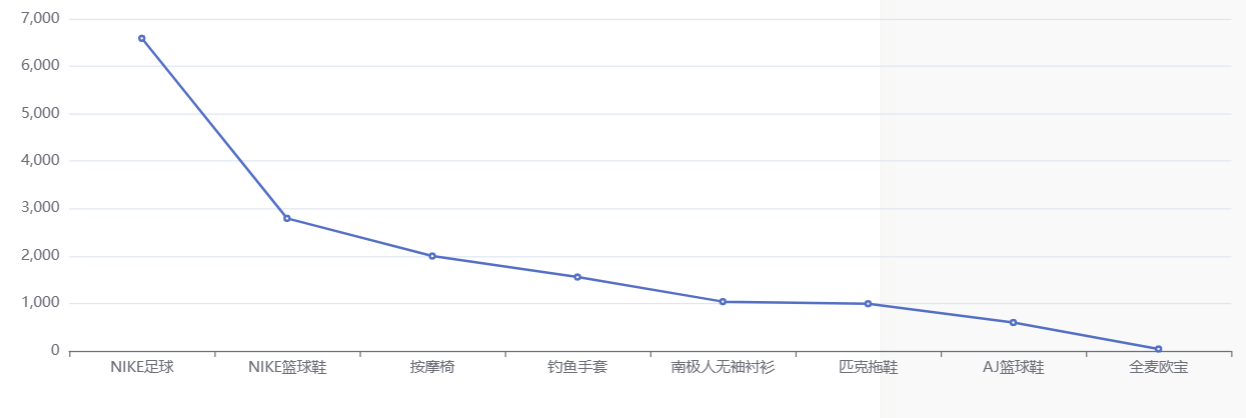


图4.48 统计十强商品展示

这是一个名为ShangpinTongji的Vue组件，用于展示电子商城的商品统计信息。下面是对该组件的功能、页面作用和特点的详细描述：

功能：

- 获取十强商品的数据并展示在饼图和折线图中。

- 通过接口获取商品类别统计数据，并以饼图和折线图的形式展示。

- 提供上传文件的功能，并将上传的文件保存到服务器上。

- 实现添加管理员的功能。

页面作用：

- 作为电子商城后台管理系统的一部分，用于展示商品统计信息和进行管理员管理操作。

- 在页面顶部显示电子商城的品牌标志和导航菜单。

- 在页面内容区域展示商品统计图表。

- 在页面底部显示版权信息。

总体而言，该组件是一个用于展示商品统计信息和进行管理员管理的页面，它提供了直观的图表展示和数据操作功能，方便管理员对电子商城的运营进行分析和管理。

### 4.2.4用户功能展示

##### 4.2.4.1修改个人信息

如图 4.49修改个人信息

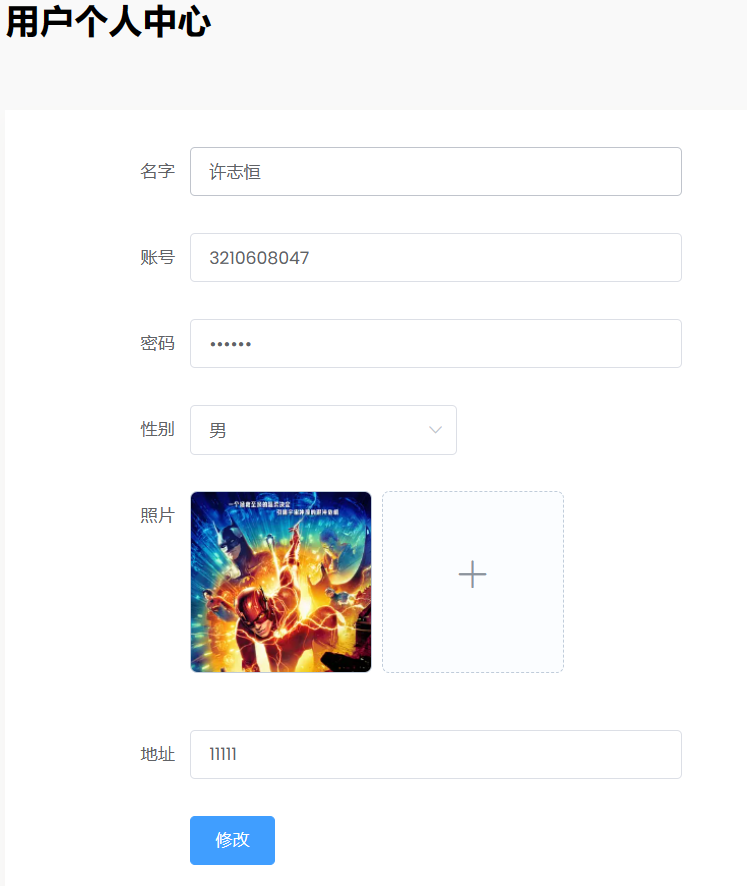


图4.49修改个人信息

这是一个名为UserIndex的Vue组件，用于展示用户个人中心页面。下面是对其功能、页面作用和特点的分析：

功能：

- 显示用户个人信息：包括用户ID、姓名、账号、密码、性别、照片、地址等信息。

- 修改用户个人信息：用户可以修改姓名、密码、性别、照片和地址等信息，并保存修改后的内容。

- 上传、预览和删除照片：用户可以上传个人照片，并在页面上进行预览和删除操作。

- 展示用户个人信息：通过表单控件将用户的个人信息显示在页面上，包括文本输入框、下拉选择框和上传照片组件等。

- 提供修改功能：用户可以修改个人信息，并通过点击"修改"按钮将修改后的信息保存到服务器。

- 支持照片上传：用户可以通过上传照片功能上传个人照片，并在页面上进行预览、下载和删除操作。

##### 4.2.4.2查询商品信息

如图4.50 查询商品信息



可以看出在这个页面的输入框里面输入想要得到的商品的信息并且点击搜索可以实现查询信息功能。

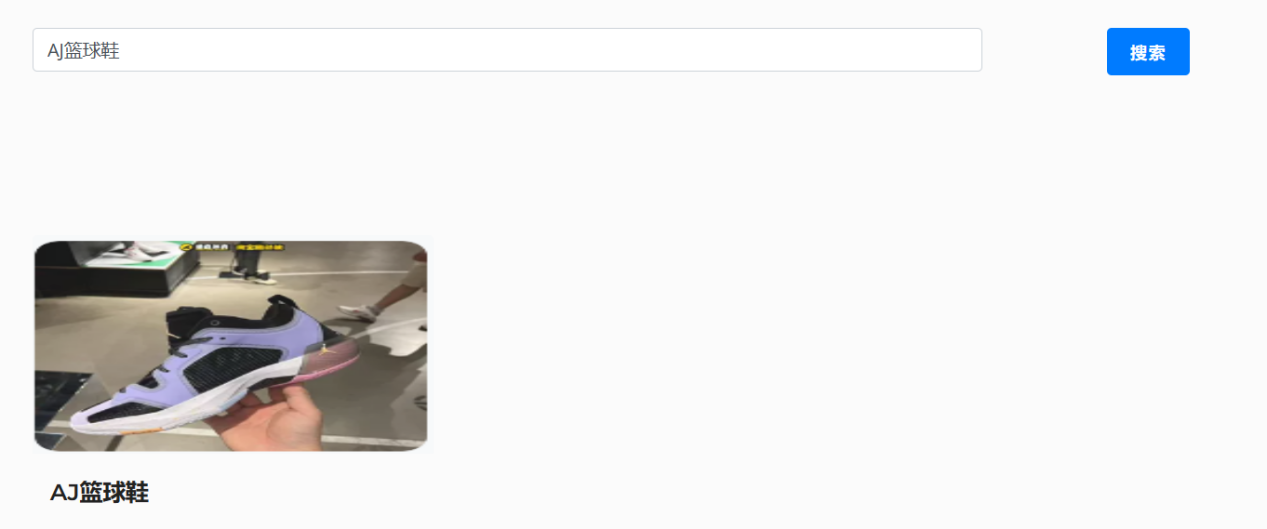


图4.50 查询商品信息

##### 4.2.4.3购物/加入购物车/查看订单信息

如图4.51 购物/加入购物车/查看订单信息



图4.51 购物/加入购物车/查看订单信息

UserAddShopcar.vue是一个Vue组件，用于在电子商城网站中添加购物车功能的页面。以下是对其功能、页面作用和特点的详细分析：

功能：

1. 添加购物车：用户可以选择商品、填写数量和状态，然后点击添加按钮将商品添加到购物车中。

2. 商品选择：用户可以从商品列表中选择商品，并根据选择的商品动态显示对应的价格。

3. 文件上传：支持用户上传文件，用于商品图片等信息的展示。

4. 时间获取：提供了获取当前时间的函数。

5. 表单序列化：提供了将表单数据序列化为对象的方法。



在购物车页面可以查看已经选择的商品信息，以及对他们进行搜索，批量删除等操作。

在下面也可以看到对于商品的管理操作，比如查看详情，结算操作以及删除操作，接下来我会展示商品的结算操作。

在购物车管理页面可以得到你已经购买的商品的信息，可以查看他们的详情信息。如图4.52 购物车管理页面



图4.52 购物车管理页面

1. **总结**

在设计Vue页面时，我面临了一些困难，主要是由于缺乏相关的专业知识和经验。为了克服这些困难，我积极利用互联网资源进行学习和搜索。我在Bilibili这个网站上找到了一些教学视频和教程，通过观看这些资源，我学习了Vue框架的基本概念、组件的使用以及前端页面的布局和设计技巧。这些学习资源对我的开发工作起到了很大的帮助。

另外，我还向老师请教了一些问题，并从老师那里学到了Vue与数据库连接的操作方法。老师的指导让我更好地理解了前后端的交互过程，以及如何将Vue前端与后端的数据库进行数据交互和处理。这对于实现系统的功能和数据管理非常重要。

然而，在开发过程中，我也遇到了一些问题，最主要的是Vue与数据库连接不成功的问题。在面对这个问题时，我通过在CSDN等技术社区进行搜索，找到了一些解决方法和实践经验。我逐步排查了代码中的错误和配置问题，并根据搜索到的解决方案进行了调整和修改。最终，我成功地实现了Vue与数据库的连接，确保了系统能够正常获取和处理数据。

在整个开发过程中，我感激老师和同学们的帮助和支持。老师给予了我很多宝贵的指导和建议，使我能够更好地理解和应用Vue框架。同时，同学们之间的互相交流和讨论也帮助我解决了一些问题，拓宽了我的视野。

通过这次开发经历，我不仅学到了Vue框架的应用和前后端交互的技巧，还提升了自己的问题解决能力和学习能力。我意识到，在面对困难和挑战时，积极主动地寻求帮助和学习是很重要的。通过不断地学习和尝试，我逐渐克服了困难，完成了系统的开发任务。

总结起来，这次开发经历让我深刻体会到了学习的重要性和持续学习的必要性。在未来的学习和工作中，我将继续保持积极的学习态度，不断提升自己的技能和知识水平，以应对各种挑战和困难。同时，我也会更加注重团队合作和互相学习的精神，共同进步，共同实现项目的成功。