|  |
| --- |
| 目 录 |

[摘 要 I](#_Toc103669314)

[Abstract II](#_Toc103669315)

[第1章 前 言 1](#_Toc103669316)

[1.1 本项目的研究背景及意义 1](#_Toc103669317)

[1.2 关于本项目的国内外研究现状分析（二级标题） 1](#_Toc103669318)

[1.3 项目的开发范围及目标（二级标题） 1](#_Toc103669319)

[1.4 论文结构简介（二级标题） 1](#_Toc103669320)

[第2章 技术原理 3](#_Toc103669321)

[2.1 系统开发相关技术介绍 3](#_Toc103669322)

[2.2 系统开发工具介绍 4](#_Toc103669323)

[第3章 XXX系统的需求建模 5](#_Toc103669324)

[3.2 XXX系统角色定义 5](#_Toc103669325)

[3.2 XXX系统功能性需求分析 6](#_Toc103669326)

[3.3 XXX系统非功能性需求分析 7](#_Toc103669327)

[3.5 XXX系统的数据流图 8](#_Toc103669328)

[第4章 概要设计 11](#_Toc103669329)

[4.1 本章参考写法 11](#_Toc103669330)

[4.1 系统架构设计及原理 11](#_Toc103669331)

[4.2 功能模块设计 12](#_Toc103669332)

[4.3 数据库设计 17](#_Toc103669333)

[第5章 模块设计 21](#_Toc103669334)

[5.1 XXX模块的设计与实现（租房用户模块的设计与实现） 21](#_Toc103669335)

[5.2 XXX模块的设计与实现 24](#_Toc103669336)

[5.3 XXX模块的设计与实现 25](#_Toc103669337)

[第6章 部署和应用 27](#_Toc103669338)

[6.1 本章参考写法 27](#_Toc103669339)

[6.1 XXX系统的部署 27](#_Toc103669340)

[6.2 XXX系统的应用 29](#_Toc103669341)

[6.3 XXX系统功能性测试 36](#_Toc103669342)

[第7章 总结与展望 41](#_Toc103669343)

[7.1 本章参考写法 41](#_Toc103669344)

[参考文献 42](#_Toc103669345)

[致 谢 43](#_Toc103669346)

网吧管理系统设计与实现

摘 要

随着经济的发展，社会的进步，计算机越来越深入到我们日常的工作学习及生活中，成为我们日常生活中不可缺少的辅助工具。网吧也随之在现代生活中越来越流行，但随着发展越来越壮大，需要的机器增多，人工管理也有很多不足的地方，所以需要开发网吧管理系统。

但只考虑到以上问题是远远不够的，现实中的网吧是由显示器和高配置主机构成的，而云网吧则是本地只留显示器和一个低配置主机，然后云端机房经过互联网的加持后充当真正的高配置主机，用户玩大型游戏的时候，其实“烧”的是云端机房的配置，基本上网吧能做的事情，云网吧都能做得到。而云网吧的可弹性伸缩的算力和存储能力，极大减少网吧的硬件投入和经营风险，一站式解决上网服务经营者的投入难题，当今云网吧才是主流。

关键词：云网吧；云端机房；可弹性伸缩的能力；存储能力；

关键词：关键词1；关键词2；……关键词n

(\*关键词一般为论文中出现的重要的、描述主题的词；

中英文关键词不少于三个，不多于七个。\*)

关键字部分：小四号、黑体、1.15倍行距，段前0行、段后0行，顶格（无首航缩进）、

多个关键词用中文分号隔开，最后一个关键词无需标点

**Design and Implementation of Internet Bar Management System**

中英文摘要的页脚：5号字体、分别罗马数字 I II

**Abstract**

With the development of economy and the progress of society, computers have become more and more involved in our daily work, study and life, and become an indispensable auxiliary tool in our daily life. Internet cafes are also becoming more and more popular in modern life. However, with the development of Internet cafes, more and more machines are needed, and there are many shortcomings in manual management. Therefore, it is necessary to develop an Internet cafes management system.

But it is far from enough to only consider the above problems. In reality, Internet cafes are made up of displays and high-configuration hosts, while cloud Internet cafes are local only with displays and a low-configuration host, and then the cloud computer room acts as a real high-configuration host after being supported by the Internet. When users play large games, what they "burn" is the configuration of the cloud computer room. Basically, what the Internet cafes can do is what the cloud Internet cafes can do. The flexible computing power and storage capacity of cloud Internet cafes greatly reduce the hardware investment and business risks of Internet cafes, and solve the investment problems of Internet service operators in a one-stop way. Today, cloud Internet cafes are the mainstream.

(\* 英文摘要严格使用英文标点符号。中、英文摘要意思要基本相同。在撰写完英文摘要后，请执行一次自动拼写检查，以减少英文拼写错误的可能性。\*)

**Key words: cloud Internet cafe; Cloud machine room; The ability to flex; Storage capacity**

|  |
| --- |
| 第1章 前 言 |

1.1 本项目的研究背景及意义

**1.1.1 项目研究背景（三级标题）**

(\*写好内容后用格式刷\*)

**1.1.2 项目开发意义（三级标题）**

(\*写好内容后用格式刷\*)

1.2 关于本项目的国内外研究现状分析（二级标题）

(\*写好内容后用格式 刷\*)

1.3 项目的开发范围及目标（二级标题）

(\*写好内容后用格式刷\*)

1.4 论文结构简介（二级标题）

本文从传统租房行业中存在的问题入手，对目标用户进行需求调研、与国内XXX租房系统，国外XXX系统进行分析对比，并且对开发技术进行分析、选型，依据该租房系统的需求分析，完成了XXX在线租房系统的设计与开发，实现了XXXX功能。本论文以软件开发的大致流程来介绍本在线租房系统，具体内容如下：

第一章节是前言，讲述在线租房系统的开发的背景和意义，综述了国内外关于在线租房网站的现状，确定了本系统的开发范围及目标。

第二章节是技术与原理，是讲述该项目需要用到的技术以及该项技术实现的原理，以及讲述开发的开发工具，还有系统开发环境的介绍。

第三章节是需求建模，首先讲述了系统可行性分析，然后对系统的需求做了个详细的分析，然后对列出系统角色的定义，最后画出系统类图。

第四章节是概要设计，首先讲述了系统技术架构，然后介绍了系统的物理架构包括服务器的选择和数据库的选择，最后是讲述了对数据库架构的设计 。

第五章节是模块设计，本章节是详细讲述了系统的各个模块设计，分别从类目模块，商品模块，订单模块，微信授权登录及推送模块，支付模块等进行了讲述，让读者能更清晰的了解本系统。

第六章节是部署与应用，分别从部署环境，以及更改环境所需要注意的事项都完整的讲述了。

第七章是本论文的尾声，是本人对此次项目的总结。

本科生工程论文应该注重标准与规范，首先要注意以下问题：（1）工程论文不是软件文档的堆砌；（2）不应包含代码、大量界面等细节。

|  |
| --- |
| 第2章 技术原理 |

2.1 系统开发相关技术介绍

该系统是基于Vue、Layui、Bootstrap、JQuery、Echarts实现的，界面样式是由Layui+Bootstrap展示的，Vue提供动态属性绑定，实现无刷新动态更新数据，请求接口使用Ajax异步请求，使用Echarts首页展示数据报表，让管理员快速简洁地查看近几天的数据变化，后端使用SpringBoot+MybatisPlus的技术架构进行构建，应用关系型数据库MySQL和非关系型数据库Redis，简化了整体的开发过程。利用MySQL进行数据的持久化存储，结合Druid高效可管理的连接器进行连接；利用Redis进行数据缓存，用于后台管理员登录的时效性控制功能；使用SpringSecurity进行授权和认证，拦截无效请求，增加系统安全性。

**2.1.1 Vue.js**

Vue基于MVVM模式，即数据驱动，采用数据劫持结合发布订阅模式实现数据的双向绑定，并且使用了虚拟DOM，即将真实DOM节点用JS对象来表示，底层通过diff算法对新旧虚拟节点进行比较，实现最小量更新，由此避免了开发者直接操作DOM，同时也提供了更好的性能。

**2.1.2 SpringBoot框架**

SpringBoot是对Spring的升级改进，它基于Spring4设计，去除了Spring中繁琐的配置项，自动装配了很多配置，在使用到这些配置时会自动调用。SpringBoot通过Maven或Gradle等工具，可以生成可直接执行的jar包和war包；SpringBoot内置了Tomcat，省去了Spring中需要配置Tomcat的麻烦；SpringBoot提供了非常多的starter，以此来简化Maven配置。

**2.2.4 SpringSecurity框架**

Spring Security是一个能够为基于Spring的企业应用系统提供声明式的安全访问控制解决方案的安全框架。它提供了一组可以在Spring应用上下文中配置的Bean，充分利用了Spring IoC，DI（控制反转Inversion of Control ,DI:Dependency Injection 依赖注入）和AOP（面向切面编程）功能，为应用系统提供声明式的安全访问控制功能，减少了为企业系统安全控制编写大量重复代码的工作。

**2.2.5 Echarts**

ECharts是一款基于[JavaScript](https://baike.baidu.com/item/JavaScript/321142?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/ECharts/_blank)的[数据可视化](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%8F%AF%E8%A7%86%E5%8C%96/1252367?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/ECharts/_blank)图表库，提供直观，生动，可交互，可个性化定制的数据可视化图表。ECharts最初由[百度](https://baike.baidu.com/item/%E7%99%BE%E5%BA%A6/6699?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/ECharts/_blank)团队开源，并于2018年初捐赠给[Apache](https://baike.baidu.com/item/Apache/6265?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/ECharts/_blank)基金会，成为ASF孵化级项目。

**2.2.6 MyBtisPlus框架**

MyBatis-Plus（简称 MP）是一个基于 MyBatis 的增强工具，它对 Mybatis 的基础功能进行了增强，但未做任何改变。使得我们可以可以在 Mybatis 开发的项目上直接进行升级为 Mybatis-plus，正如它对自己的定位，它能够帮助我们进一步简化开发过程，提高开发效率。

2.2.7 RabbitMQ

RabbitMQ是实现了高级[消息队列](https://baike.baidu.com/item/%E6%B6%88%E6%81%AF%E9%98%9F%E5%88%97/4751675?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/rabbitmq/_blank)协议（[AMQP](https://baike.baidu.com/item/AMQP/8354716?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/rabbitmq/_blank)）的开源消息代理软件（亦称面向消息的[中间件](https://baike.baidu.com/item/%E4%B8%AD%E9%97%B4%E4%BB%B6/452240?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/rabbitmq/_blank)）。RabbitMQ服务器是用[Erlang](https://baike.baidu.com/item/Erlang?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/rabbitmq/_blank)语言编写的，而集群和[故障转移](https://baike.baidu.com/item/%E6%95%85%E9%9A%9C%E8%BD%AC%E7%A7%BB/14768924?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/rabbitmq/_blank)是构建在[开放电信平台](https://baike.baidu.com/item/%E5%BC%80%E6%94%BE%E7%94%B5%E4%BF%A1%E5%B9%B3%E5%8F%B0/15696499?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/rabbitmq/_blank)框架上的。所有主要的[编程语言](https://baike.baidu.com/item/%E7%BC%96%E7%A8%8B%E8%AF%AD%E8%A8%80/9845131?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/rabbitmq/_blank)均有与代理接口通讯的客户端库。

2.2 系统开发工具介绍

**2.2.1 开发工具Idea**

IDEA 全称 IntelliJ IDEA，是[java](https://baike.baidu.com/item/java/85979?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/IntelliJ%20IDEA/_blank)编程语言的[集成开发环境](https://baike.baidu.com/item/%E9%9B%86%E6%88%90%E5%BC%80%E5%8F%91%E7%8E%AF%E5%A2%83/298524?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/IntelliJ%20IDEA/_blank)。IntelliJ在业界被公认为最好的[Java](https://baike.baidu.com/item/Java/85979?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/IntelliJ%20IDEA/_blank)[开发工具](https://baike.baidu.com/item/%E5%BC%80%E5%8F%91%E5%B7%A5%E5%85%B7/10464557?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/IntelliJ%20IDEA/_blank)，尤其在智能代码助手、代码自动提示、重构、[JavaEE](https://baike.baidu.com/item/JavaEE/3066623?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/IntelliJ%20IDEA/_blank)支持、各类版本工具([git](https://baike.baidu.com/item/git/12647237?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/IntelliJ%20IDEA/_blank)、[svn](https://baike.baidu.com/item/svn/3311103?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/IntelliJ%20IDEA/_blank)等)、[JUnit](https://baike.baidu.com/item/JUnit/1211849?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/IntelliJ%20IDEA/_blank)、[CVS](https://baike.baidu.com/item/CVS/405463?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/IntelliJ%20IDEA/_blank)整合、代码分析、 创新的GUI设计等方面的功能可以说是超常的

**2.2.2 数据库MySQL**

MySQL 是一款安全、跨平台、高效的，并与 PHP、Java 等主流编程语言紧密结合的数据库系统。体积小，速度快，而且功能丰富强大。使用Mysql作为数据库可以极大的降低开发成本。

**2.2.3 InnoDB**

InnoDB是 MySQL 上第一个提供外键约束的数据存储引擎，除了提供事务处理外，InnoDB 还支持行锁，提供和 Oracle 一样的一致性的不加锁读取，能增加并发读的用户数量并提高性能，不会增加锁的数量。InnoDB 的设计目标是处理大容量数据时最大化性能，它的 CPU 利用率是其他所有基于磁盘的关系数据库引擎中最有效率的。

2.2.4 非关系型数据库Redis

Redis是一个开源(BSD许可)的，用C语言编写的基于内存的数据结构存储系统（是一个高性能的 key-value存储系统）。而且会周期性的把更新的数据写入磁盘或者把修改操作写入追加的记录文件，实现数据的持久化。Redis可以用在数据库，缓存和消息中间件。

2.2.5 Maven

Maven的本质是一个项目管理工具，(Maven是java语言编写的，所以Maven管理的东西，都已面向对象的方式进行设计)，将项目开发和管理过程抽象成一个项目模型(POM)

|  |
| --- |
| 第3章 网吧管理系统的需求建模 |

本章基于网吧管理系统设计与开发，接下来对系统进行详细的需求分析。主要包括功能需求分析和性能需求分析两个方面来介绍网吧管理系统，其中用例分析采用UML图例来阐述。

3.2 网吧管理系统角色定义

本网吧管理系统主要有三个角色，分别为管理员、收银人员和用户。用户在登录桌面上输入身份证和密码进行登录，登录后即可解锁电脑；收银人员进入登录页面，分别输入用户名和密码进行登录，登录后可以对用户、清单进行管理；管理员进入登录页面，分别输入用户名和密码进行登录，登录后可以对设备、清单进行管理。用例图分别为：

1．管理员：主要包括 设备管理、收银人员管理、清单列表等用例，用例图如图3.x所示：

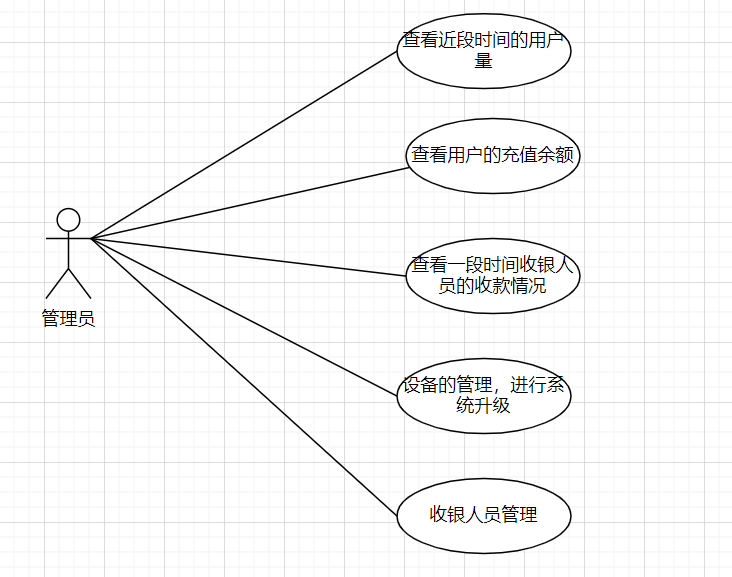


图3.x XXX用例图

2．收银人员：主要包括开卡/充值、用户管理、会员卡管理XXX等用例，用例图如图3.x所示：

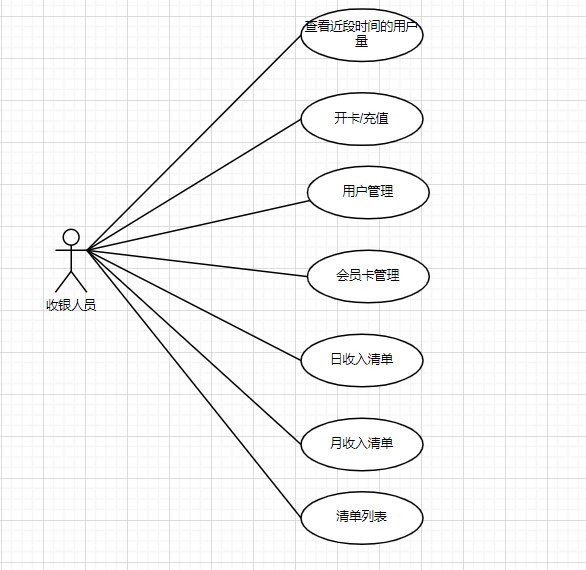


图3.x XXX用例图

3．用户：主要包括在线充值、操作电脑、查看自己的上机消费记录等用例，用例图如图3.x所示：

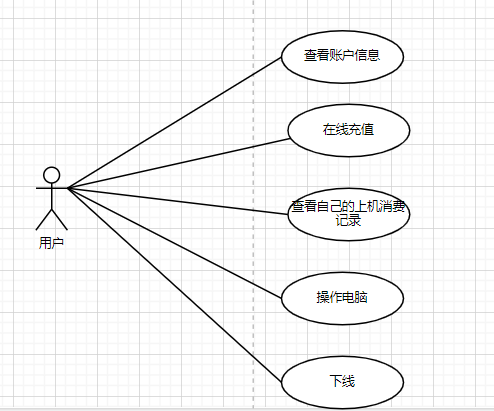


图3.x XXX用例图

3.2 XXX系统功能性需求分析

**3.2.1 用户登录和注册功能的需求分析**

用户登录和注册包含收银人员和用户两个角色，故分两部分分别分析。其中收银人员包含注册、登录、注销、用户信息管理等功能，而用户有登录功能，但没用注册功能，只能由网管进行注册。系统不为管理员提供用户管理功能。用户管理的用例图如图 2-4 所示。

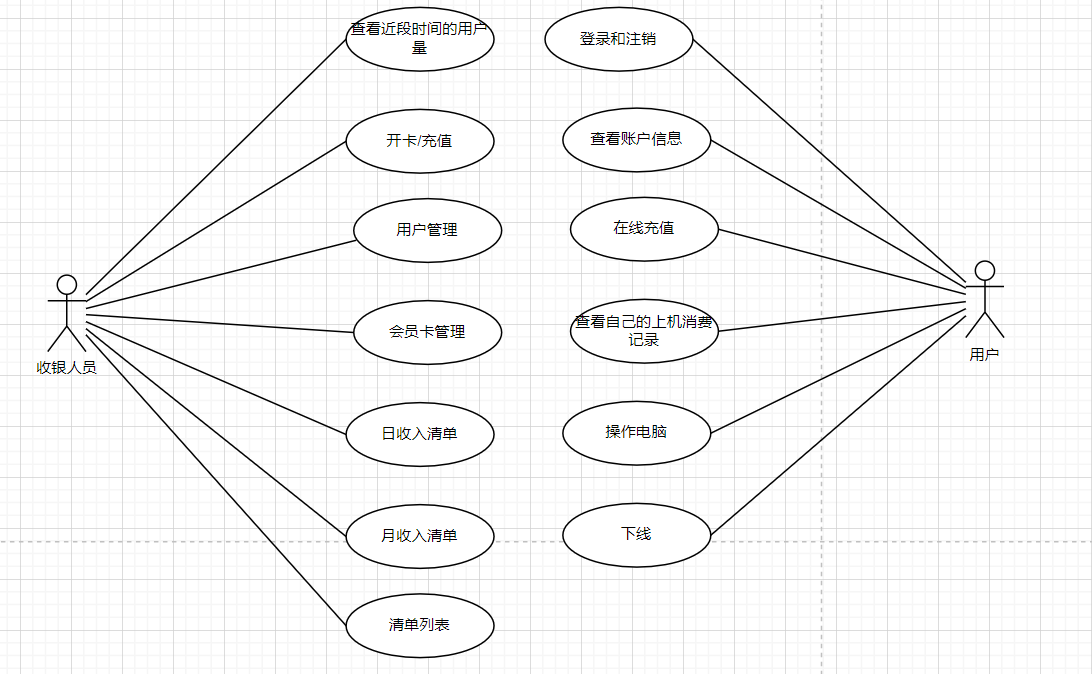


图3.x XXX用例

用户登录和注册功能需求分析如下：

1. 收银人员管理功能。
2. 收银人员注册和注销功能：用户进入到注册页面，需要输入注册手机号和密码以及系统验证码。所有信息校验完毕后，才发送注册请求，以防止用户填错手机号或恶意注册。用户注册时提供的信息需要记录到服务器并实现持久化的保存，以供后续的读取显示和其他作用，注册同理。
3. 用户登录与注销功能：用户在注册之后，可以凭借注册时的身份证登录，在校验完毕后便可登录。出于安全性和技术上的考量，用户登录只通过身份证和密码登录。登录后，网页端或者 App 客户端能够记录用户的登录状态，保证用户能在一段时间内自动登录。注销功能则需要清除用户的登录状态，保障用户的隐私安全。
4. 用户个人信息编辑功能：用户在登录后，可以查看个人信息，包括用户名、身份证、手机号、邮箱等。用户能够更改这些信息，更新后的信息需要保存到服务器。其中手机号的更改流程需要和注册时一致，要经过验证流程。
5. 用户可以在线呼叫网管：用户在登录后，出现了任何需求问题，都可以呼叫网管，网管收到信息后会回应用户。

表3-1 用户登录用例描述

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | ZYZ01 | 用例名称 | 登录 |
| 编制人 |  | 编制日期 | 2022/02/08 |
| 用例描述 | 此用例主要描述用户进行系统的登录 | | |
| 参与者 | 用户 | | |
| 前置条件 | 用户已经注册系统 | | |
| 后置条件 | 根据用户身份来决定用户进入系统后可以看见的功能模块 | | |
| 优先级别 | 高 | | |
| 基本事件流   1. 用户进入登陆界面 2. 用户输入账号密码后点击登录 | | | |
| 可选事件流  2a. 用户输入错误的账号密码，提示用户进行注册  2b. 用户输入正确的账号密码，进入到系统中，并且根据用户的身份不同展示不同的功能模块 | | | |
| 特殊需求 | 无 | | |
| 扩展点 | 登录的滑块验证 | | |

**3.2.1 用户管理功能的需求分析**

用户管理包含对用户的搜索、用户的详情查看、用户的编辑、用户的删除、用户的启用/禁用

的用例图如图 2-4 所示。

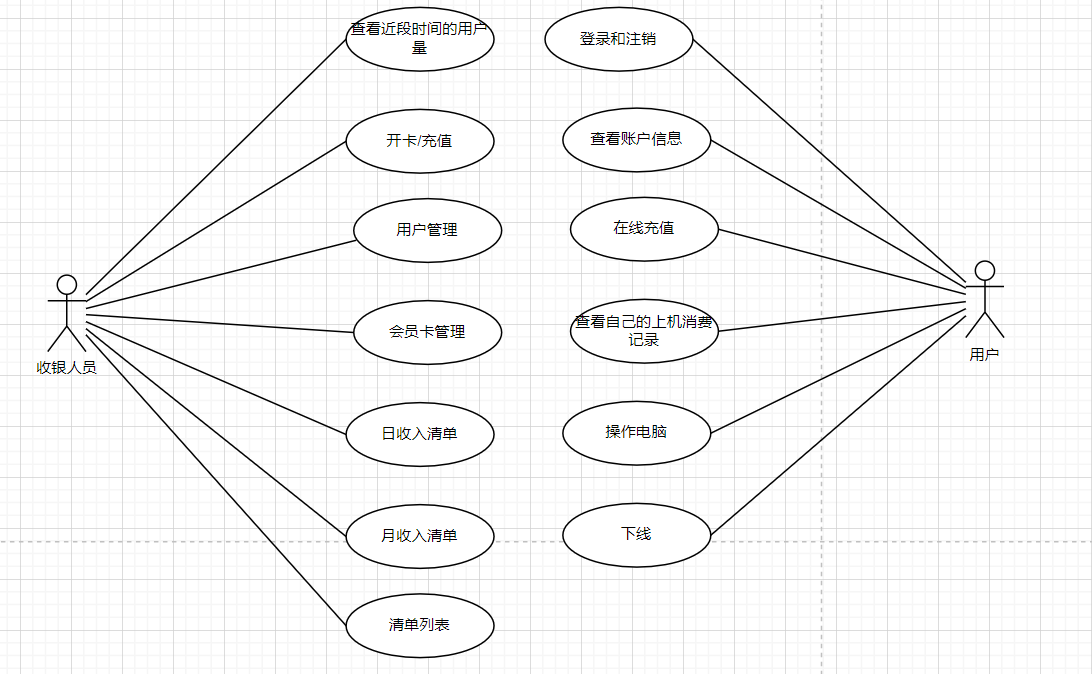


图3.x XXX用例

用户管理功能需求分析如下：

（1）收银人员管理功能。

1）用户的搜索功能：收银人员登录后，如果想查看某一用户的账户信息，可以通过用户的身份证、用户的手机号、用户的性别来搜索到指定用户，并对其操作。

2）用户的详情查看功能：用户在注册之后，可以凭借注册时的身份证登录，在校验完毕后便可登录。出于安全性和技术上的考量，用户登录只通过身份证和密码登录。登录后，网页端或者 App 客户端能够记录用户的登录状态，保证用户能在一段时间内自动登录。注销功能则需要清除用户的登录状态，保障用户的隐私安全。

3）用户个人信息编辑功能：用户在登录后，可以查看个人信息，包括用户名、身份证、手机号、邮箱等。用户能够更改这些信息，更新后的信息需要保存到服务器。其中手机号的更改流程需要和注册时一致，要经过验证流程。

4）用户可以在线呼叫网管：用户在登录后，出现了任何需求问题，都可以呼叫网管，网管收到信息后会回应用户。

表3-1 用户登录用例描述

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | ZYZ01 | 用例名称 | 登录 |
| 编制人 |  | 编制日期 | 2022/02/08 |
| 用例描述 | 此用例主要描述用户进行系统的登录 | | |
| 参与者 | 用户 | | |
| 前置条件 | 用户已经注册系统 | | |
| 后置条件 | 根据用户身份来决定用户进入系统后可以看见的功能模块 | | |
| 优先级别 | 高 | | |
| 基本事件流   1. 用户进入登陆界面 2. 用户输入账号密码后点击登录 | | | |
| 可选事件流  2a. 用户输入错误的账号密码，提示用户进行注册  2b. 用户输入正确的账号密码，进入到系统中，并且根据用户的身份不同展示不同的功能模块 | | | |
| 特殊需求 | 无 | | |
| 扩展点 | 登录的滑块验证 | | |

**3.2.2 XXX功能的需求分析**

(\*写好内容后用格式刷\*)

**3.2.2 XXX功能的需求分析**

(\*写好内容后用格式刷\*)

3.3 XXX系统非功能性需求分析

在线租房系统是为租房用户提供房源信息查看和房源租赁的系统，也是为第三方租房平台提供房源展示的信息推广系统。一个系统如果想拥有良好的用户体验，就要拥有良好的性能特性。下面就针对数据精确度、可使用性及兼容性、时间特性、安全保密性、可维护性几个方面的性能需求来详细介绍本在线租房系统。

**3.2.2 数据精确度**

在线租房系统在房源或者用户资料等信息录入时，需要保证填写的信息经过数据格式和现实逻辑的校验后才录入系统。同时需要保证数据显示的格式符合现实认知，必要数据的精确程度不会过繁过简，而是符合现实常识和逻辑需要。

**3.2.2 可使用性及兼容性**

(\*写好内容后用格式刷\*)

**3.2.2 时间特性**

(\*写好内容后用格式刷\*)

**3.2.2 安全保密性**

(\*写好内容后用格式刷\*)

**3.2.2 可维护性**

(\*写好内容后用格式刷\*)

3.5 XXX系统的数据流图

数据流图也称为数据流程图（DFD），它可以让系统分析者更加直观清楚地明白这个系统是做什么的可解决什么问题的一种图形工具。

**3.4.2 顶层层数据流图**

(\*写好内容后用格式刷\*)

第0层数据流图展示系统的参与角色及对应角色间对系统进行的操作。从大体上展现了本系统的基本功能。



图3-x 顶层数据流图

**3.5.3 第一层数据流图**

(\*写好内容后用格式刷\*)

第一层的数据流图中，实体类包括微信用户和商家。微信用户的数据流程只概括成了点餐，点餐首先需要从商品表中读出商品信息，用户下订单后将订单信息存进订单表。商家的数据流程概括为处理订单，当有用户下单了，本系统会及时通知商家，商家处理订单首先要从订单表中读取出订单信息，执行相应的操作修改订单状态。商家对订单执行的操作优惠存入订单表中。



图3-9 一层数据流图

**3.5.3 第2层数据流图**

第二层数据流图如图3-10所示。第二层的数据流图中，更加详细的描述了本系统的业务逻辑。首先微信用户通过公众号进入系统执行查看商品信息流程，该流程需要从商品表和类目表中分别查出商品信息和类目信息。接着用户将商品加入购物车后进行结算操作，进入下一个页面，在这个页面用户可以查看所选购的菜品，用户在这个页面填写收货地址，填写完成后提交订单，提交订单这个流程会将订单信息存入订单表和订单详情表。最后用户付款时付款的金额又需要从订单表中查询出来。用户付款成功后又会先从订单表中查询出来这个定案，修改这个订单的支付状态，再把修改完的数据存入订单表中。商家通过网页端登录界面输入账号和密码来登录该系统的后台。商家可以添加和删除类目，这个流程会将类目信息存入类目中。同时商家添加和修改商品信息也会将商品信息存入商品表中。商家在处理订单时，首先要从订单表中查询出订单信息，执行相应的操作后，把订单信息再次存入订单表中。



图3-10 二层数据流图

|  |
| --- |
| 第4章 概要设计 |

本章基于在线租房系统的需求分析，给出在线租房系统的总体设计架构、功能模块的具体细节设计以及数据库的设计等。

4.1 本章参考写法

（\* 4.1 系统架构及原理

Web项目经常采用的架构形式是MVC和分层架构，按照这些架构给出系统的架构图，图中标明各部分的主要部件及部件之间的关系。对照架构图以通俗的语言阐明系统的实现原理。（嵌入软件应包括系统原理，硬件框架）

架构图可以使用自定义的示意图，但注意图中一种符号只能表达一种涵义，避免二义性。也可使用包图、构件图等更精确的UML图示表达系统架构。

4.2 业务用例的实现

用顺序图或协作图描述用例的实现，除领域概念（类/对象）外，加入用户界面、技术支撑环境（持久化、通讯、进程管理等相关概念）等技术类概念（类/对象）。注意：一个用例可对应若干个顺序图/协作图，若用例较多，只描述主要用例的实现即可。

4.4 数据库设计

使用ER图进行概念设计，并得出关系模式（图）。只需要简要对核心数据描述

\*）以上标题供参考

4.1 系统架构设计及原理

本系统的架构图主要从应用系统的运行平台层、数据支撑层、基础系统层来展示系统的功能模块架构。其中应用系统运行平台层包括用户认证、功能目录与栏目设置以及系统的主要功能模块；数据支撑层包括用户信息数据、品牌信息数据、订单信息数据、评论信息数据、商品属性数据、商品销售数据等持久化数据存储表；基础系统层包括系统的安全信息支撑设施以及硬件设施的支撑平台。系统的架构图如下图4-1：

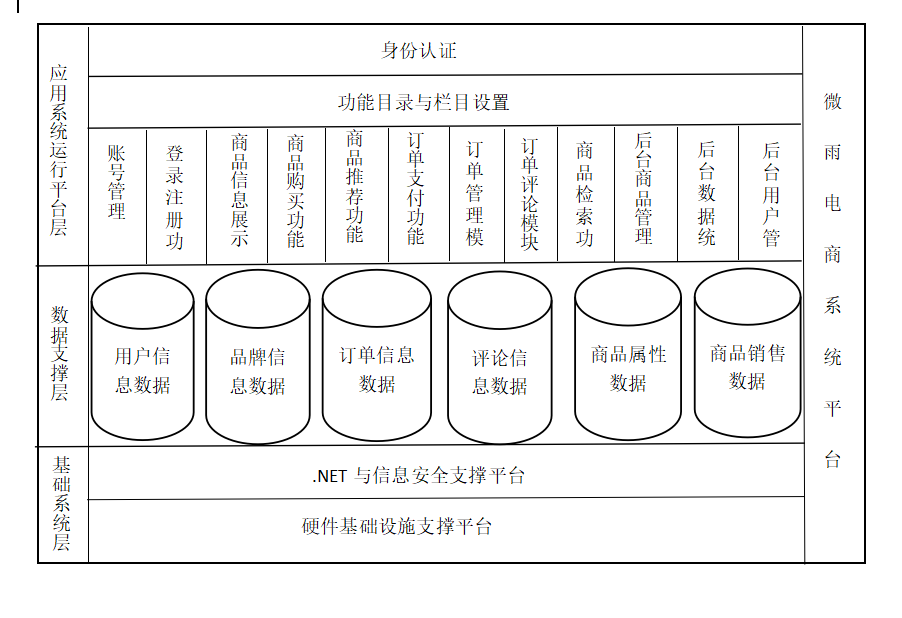


图4-1 系统架构设计

系统架构实现原理：一个比较完善的系统可以根据功能点分成几个相对独立的子系统。每一个子系统都可以被称为一个单独的节点，同时每一个节点可以配置多个集群。子系统之间可以进行直接的通信、协调和协作，从而共同完成整个系统的业务流程。它的体系结构是表示层->业务层->持久层->数据库。整个项目将部署于Tomcat。理论上，Tomcat服务器支持400个并发，800个并发数则需要2个及以上的Tomcat，8000个并发需要20个服务器才能完成Tomcat群集。当然，可以通过不断添加Tomcat服务器来解决高并发性问题。例如，每一个项目都需要用户登录这个功能点，这不可避免地会带来会话复制问题，Tomcat本身具有共享会话的能力。因此，整个系统使用分布式服务器集群的体系结构来设计不同模块之间的接口数据请求。这将有助于系统维护并提高数据安全性。

4.2 功能模块设计

微雨电商系统主要的功能模块包括客户订单管理模块、商品管理模块、消费者用户管理模块、订单管理模块、前台购物车管理模块等，除此之外还有后台数据可视化模块、商品的评论管理模块以及支付模块和商品智能推荐模块。以下是系统核心功能模块结构图如图4-2所示：

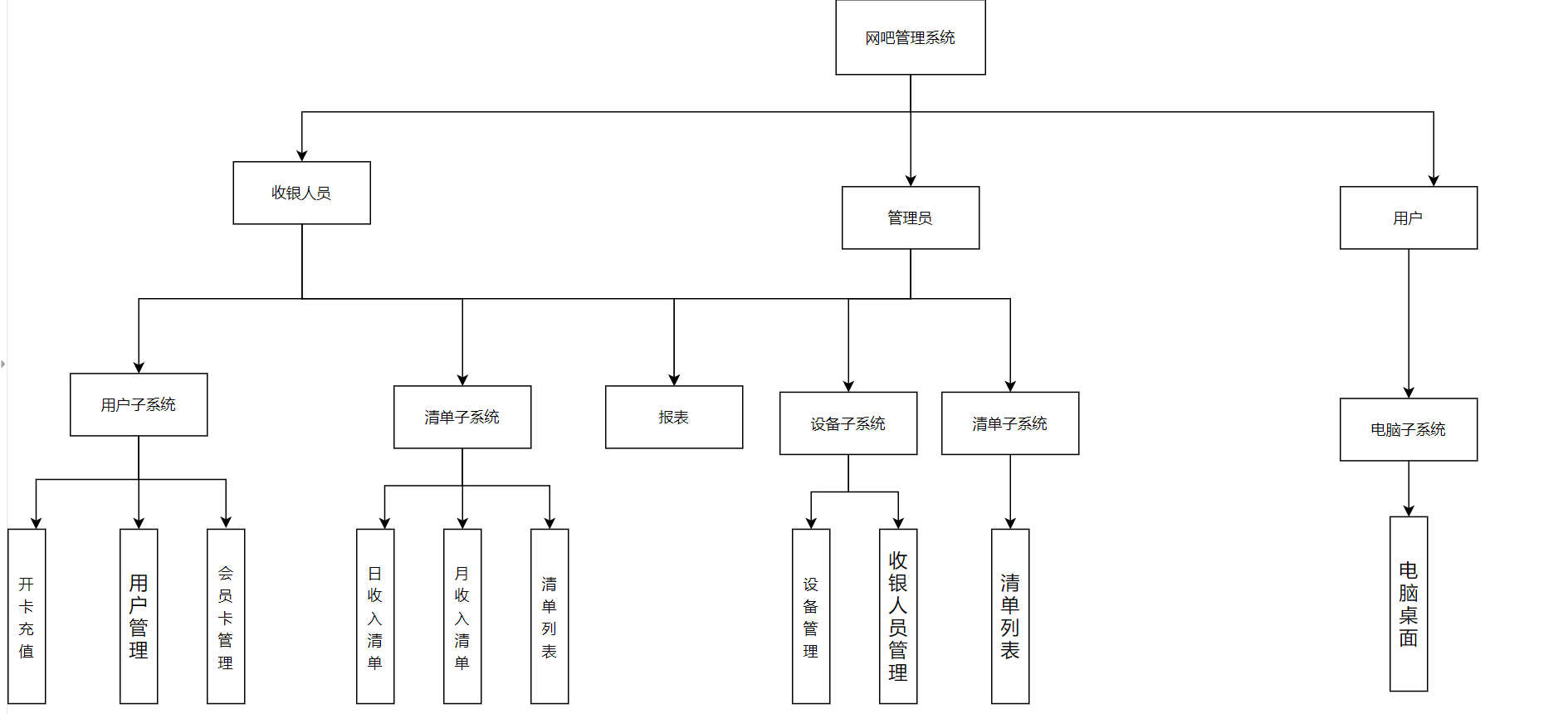


图4-2 核心功能模块结构图

**4.2.1 消费者用户管理模块**

1.时序图设计

用户先通过浏览器查看服务平台的登录界面，然后输入自己的账号和密码，前端接收到数据，封装好了，通过json数据传到后端，后端经过MVC三层架构进行处理，之后通过操作数据库，查找是否有该用户，然后把查找的结果，通过页面返回给用户。具体的“消费者用户管理模块”时序图如图4-3所示。

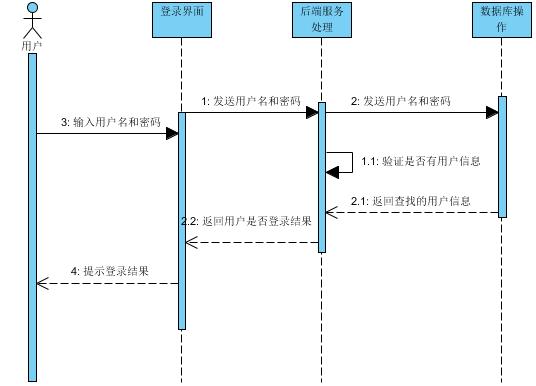


图4-3 系统人员管理顺序图

2.消费者用户管理模块核心类图设计

该类图主要是用户类的设计。主要包括消费者用户主类、用户数据库操作的实现接口、用户操作的业务接口和实现类。用户管理类图设计如图4-4所示：

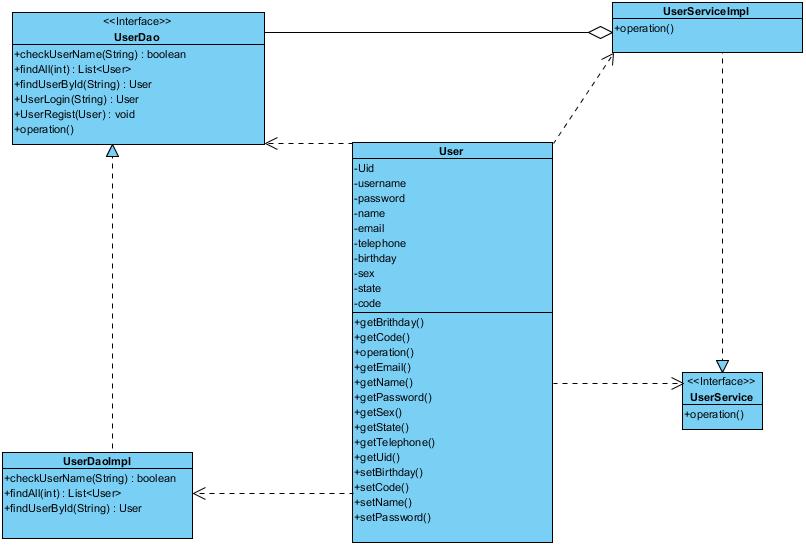


图4-4 系统用户管理类图

**4.2.2 订单管理模块**

1.时序图设计

用户通过浏览器，可以浏览商品展示界面，如果用户想看商品详细信息，还可以点击商品图标，直接可以查看商品的详细信息。如果用户想点击订购，那么平台会提示你要登录，登录之后，才可以进行订购，然后系统会获取你的基本信息，进而生成订单信息，发送到后端进行处理，之后对数据库进行操作，然后存储。之后，返回给用户一个提示信息，表示下订单成功。具体的“订单管理模块”时序图设计如下图4-5：

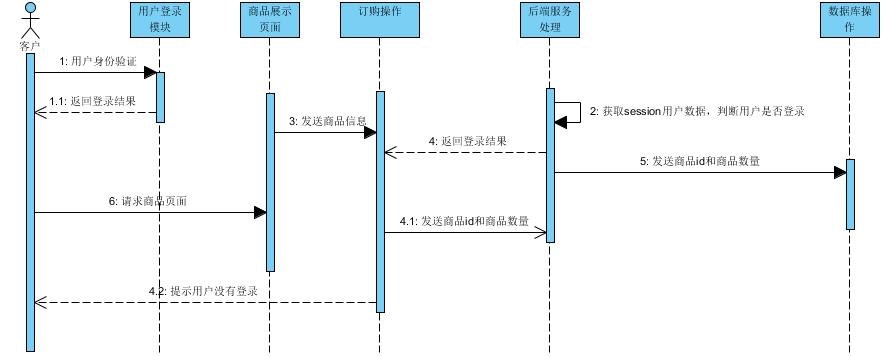


图 4-5 订单管理时序图

2.该类图主要是订单模块的主类以及对应的数据库操作接口，还有订单业务的接口和实现类。该类图主要展示了订单管理模块中的持久层、业务层、控制层之间的设计关系。订单管理模块主类图设计如图 4-6所示：

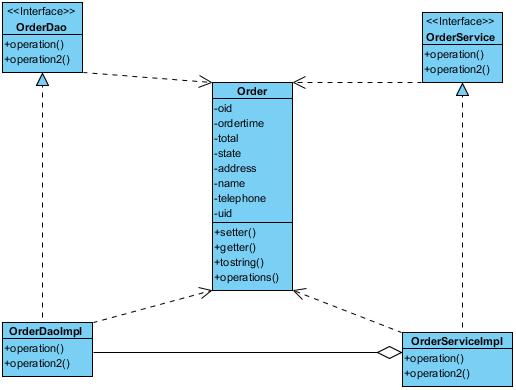


图 4-6 订单管理模块类图设计

**4.2.3 商品管理模块**

1.时序图设计

管理员查看后台管理系统，会提示管理员登录系统，登录之后，管理员可以输入商品基本信息数据，然后前端对数据进行封装，之后发送到后端，后端接收到数据，对数据进行相应的处理。然后储存在数据库中，然后通过页面返回提示信息给管理员。商品管理模块时序图如图4-7所示：

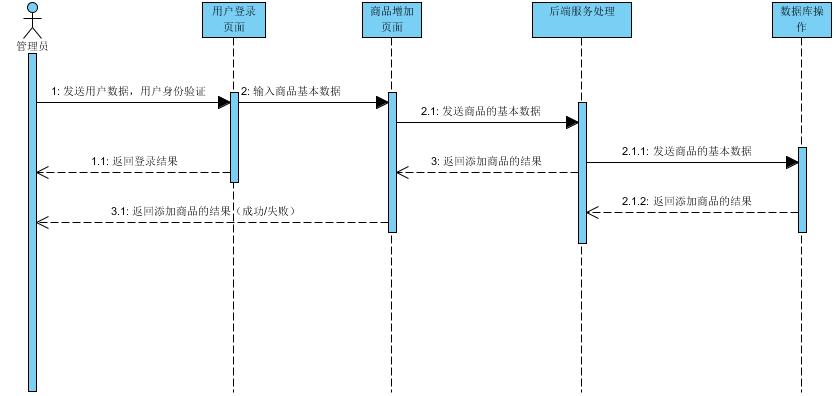


图4-7 商品管理时序图

2.该类图主要是商品类的设计以及其业务类之间的设计关系，主要包括商品类的数据库操作接口、业务模块接口和实现类。商品管理模块主类图设计如图4-8：

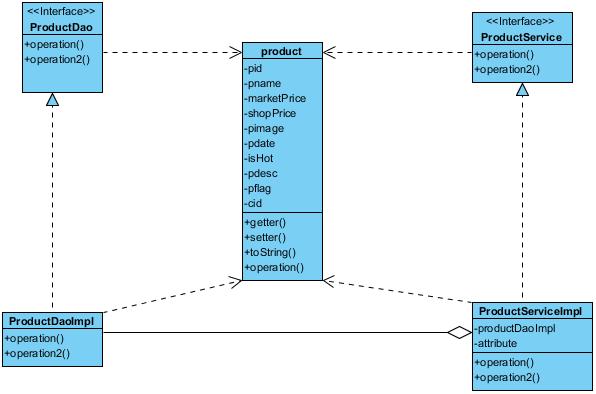


图 4-8 商品管理主类图设计

4.3 数据库设计

**4.3.1 数据库概念设计**

该系统的数据库概念设计主要包括商家、商品、买家、订单管理、订单的基本信息以及购物车模块之间的关系设计，通过模块之间的关系来体现系统的功能架构，从而便于数据库表的设计。系统数据库ER图概念数据模型如图4-9 所示：

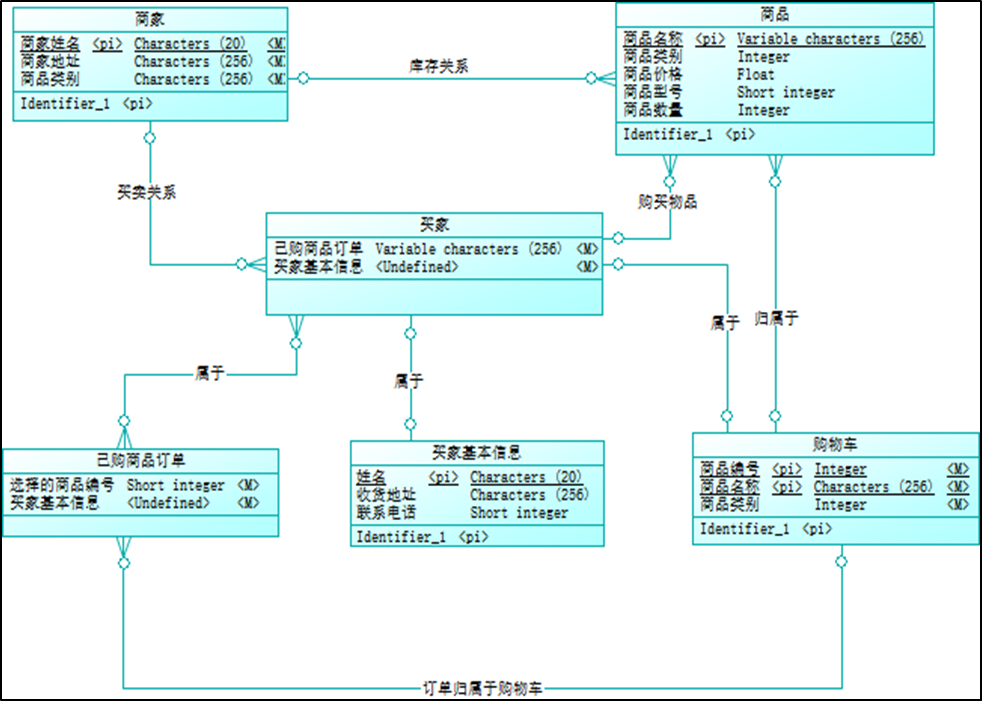


图4-9 系统数据库ER图

**4.3.2 数据库主要表设计**

系统数据库持久化层面，主要设计的表有address表、admin表、category表、product表、productimage表、productorder表、Productorderitiem表、property表、review表、user表等。主要实体化消费者用户、商家用户、商品、订单、产品分类、评论、地址等信息的设计。

表4-1 类别信息表（category）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **字段名称** | **字段描述** | **字段类型** | **长度** | **允许空** | **缺省值** |
| 1 | ID | 品牌ID | int | 11 |  |  |
| 2 | Name | 品牌名称 | varchar | 255 |  |  |

以下是集中常见的格式错误

表4-1 类别信息表（category）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **字段名称** | **字段描述** | **字段类型** | **长度** | **允许空** | **缺省值** |
| 1 | ID | 品牌ID | int | 11 |  |  |
| 2 | Name | 品牌名称 | varchar | 255 |  |  |

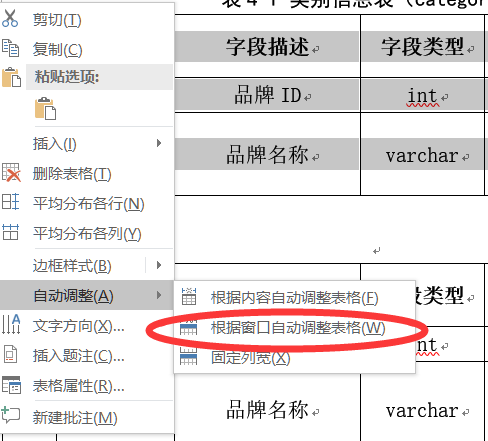
表4-1 类别信息表（category）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **字段名称** | **字段描述** | **字段类型** | **长度** | **允许空** | **缺省值** |
| 1 | ID | 品牌ID | int | 11 |  |  |
| 2 | Name | 品牌名称 | Varchar  varchar varchar | 255 |  |  |

表4-1 类别信息表（category）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **字段名称** | **字段描述** | **字段类型** | **长度** | **允许空** | **缺省值** |
| 1 | ID | 品牌ID | int | 11 |  |  |
| 2 | Name | 品牌名称 | varchar | 255 |  |  |

1. 表的宽度大小不一 ，必须设置为“根据窗口自动调整”



1. 表中各行的高度不一（尽量做到高度一致）
2. 表的标题不是在表的上方
3. 表的标题的序号、字体不正确
4. 表的标题没有正确的居中
5. 表中内容的字体是否为“五号”字
6. 表中字段名是否“加粗”
7. 表之间的空行较多
8. 以上格式规定同样适用于其他表格

|  |
| --- |
| 第5章 模块设计 |

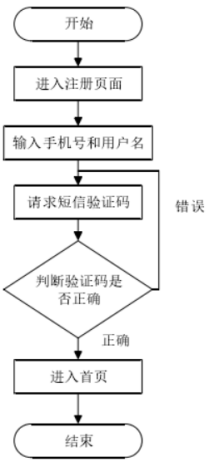
5.1 XXX模块的设计与实现（租房用户模块的设计与实现）

**5.1.1 XXX模块的功能描述**

租房用户子模块的业务逻辑设计包含租房用户注册、登录、注销、个人信息管理、反馈系统改进意见和查看租房资讯功能。用户注册需要提供自己的手机号，在收到验证短信并核实了手机号之后，则注册成功。之后，用户可以通过获取短信验证码的方式登录。这里没有提供通过密码登录的功能，因为如果要在数据库中存储用户密码的话，会需要引入更多安全方面的技术例如数据的加密和解密等以维护用户的信息安全，这样无疑增加了系统的复杂度和实现难度。在使用完网站或者 App 提供的服务之后，用户可以选择直接关闭网页或者退出应用，这样浏览器中的Cookie 或者 App 中的状态记录会继续保存用户的登录状态。在 Cookie 或者状态记录过期之前，用户都可以直接登录而不需要经过验证。如果用户选择在退出之前选择注销账户，那么用户下次使用服务之前就必须再次登录以验证自己的身份。用户在登录之后，可以更改自己绑定的手机号和用户名，并且向租房系统反馈改进意见。用户可以点击 App 首页的图片链接查看与租房相关的新闻资讯。

**5.1.2 租房用户用户操作流程**

租房用户子模块的界面设计实现用户登录、注册、个人信息的查看与修改、意见反馈和查看租房资讯等界面，用户登录、注册时，界面需要给与用户必要的提示，同时在输入存在错误时给与反馈。在个人信息查看页面，用户可以查看自己的用户名、手机号、头像等信息，并且在修改之后实时更新显示数据。用户可以在意见反馈界面提交系统改进意见；在点击资讯图片后跳转到新闻详情页。



**图5.1 租房用户的注册流程**

**5.1.3 租房用户功能界面设计**

购物网站有前台和商家后台，前台有注册和登录页面，商家后台只有登录页面，商家都是管理者，直接在数据库中添加，对于前台的用户注册，通过邮箱的形式注册，同时提供邮箱激活功能，设置邮箱激活的时间限制，保证注册安全进行，用户登录后方可参与购物及查看等操作。用户登录需填写后台生成的验证码。



图5-2 前台登录界面设计图

**4.1.4 XXX功能类图结构**

图 3-5 展示了租房用户子模块的类图结构，其中主要由 CUserController 类、

CUserLoginManager 类 和 CUserServiceImpl 类 负 责 实 现 租 房 用 户 管 理。CUserController 类中有修改手机号、用户名和头像等方法，它依赖于 CUserService和 CommonService 等接口。CUserLoginManager 类提供第三方平台提供的短信验证方法 getCaptcha()，它需要调用方提供获取短信验证码的手机号 作为参数。CUserServiceImpl 类 则 实 现 租 房 用 户 登 录 、 注 册 等 业 务 的 具 体 逻 辑 ， 以 供CUserController 类调用，它实现了 CUserService 接口

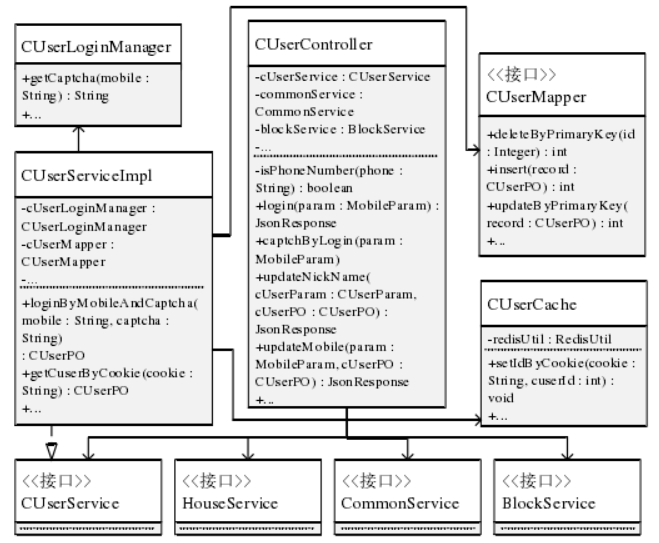


图 3-5 租房用户子模块类图

**4.1.4 XXX功能核心代码**

@GetMapping("/dishesList")

@Cacheable(cacheNames = "dishes",key = "111")

public ResultVO dishesList() {

//1. 查询所有的菜品（上架）

List<DishesInfo> dishesInfoList = dishesService.findAllDishes();

//2. 查询类目(一次性查询)

//java8, lambda表达式

List<Integer> typeList = dishesInfoList.stream()

.map(e -> e.getType().getType())

.collect(Collectors.toList());

List<Type> typeList2 = categoryService.findByCategoryTypeIn(typeList);

//3. 封装数据

List<DishesVO> dishesVOList = new ArrayList<>();

for (Type type : typeList2) {

DishesVO dishesVO = new DishesVO();

dishesVO.setCategoryType(type.getCategoryType());

dishesVO.setCategoryName(type.getCategoryName());

List<DishesInfoVO> dishesInfoVOList = new ArrayList<>();

//将菜品放到对应的类目下

for (DishesInfo dishesInfo : dishesInfoList) {

if (dishesInfo.getCategoryType().equals(type.getCategoryType())) {

DishesInfoVO dishesInfoVO = new DishesInfoVO();

BeanUtils.copyProperties(dishesInfo, dishesInfoVO);

dishesInfoVOList.add(dishesInfoVO);

}

}

dishesVO.setDishesInfoVOList(dishesInfoVOList);

dishesVOList.add(dishesVO);

}

return ResultVOUtil.success(dishesVOList);

}

5.2 XXX模块的设计与实现

**5.2.1 XXX模块的功能描述**

(\*写好内容后用格式刷\*)

**5.2.2 租房用户用户操作流程**

(\*写好内容后用格式刷\*)

**图5.1 租房用户的注册流程**

**5.2.3 租房用户功能界面设计**

(\*写好内容后用格式刷，保持与本注释段落相同的段落格式\*)

图5-2 前台登录界面设计图

**5.2.4 XXX功能类图结构**

(\*写好内容后用格式刷\*)

图 3-5 租房用户子模块类图

**5.2.4 XXX功能核心代码**

@GetMapping("/dishesList")

@Cacheable(cacheNames = "dishes",key = "111")

}

5.3 XXX模块的设计与实现

**5.3.1 XXX模块的功能描述**

(\*写好内容后用格式刷\*)

**5.3.2 租房用户用户操作流程**

(\*写好内容后用格式刷\*)

**图5.1 租房用户的注册流程**

**5.3.3 租房用户功能界面设计**

(\*写好内容后用格式刷\*)

图5-2 前台登录界面设计图

**5.3.4 XXX功能类图结构**

(\*写好内容后用格式刷\*)

(\*写好内容后用格式刷\*)

图 3-5 租房用户子模块类图

**5.3.4 XXX功能核心代码**

@GetMapping("/dishesList")

@Cacheable(cacheNames = "dishes",key = "111")

}

|  |
| --- |
| 第6章 部署和应用 |

6.1 本章参考写法

（\*

1）描述系统运行环境，可以使用UML的部署图；

2）选择若干典型界面说明系统的重要输入/输出；

3）与已有系统进行多方面的对比说明先进性或特色，或与本系统投入使用前状况对比说明应用效果。

4）对系统关键功能点或重要输入/输出的测试分析

\*）

6.1 XXX系统的部署

**6.1.1 系统运行环境**

操作系统：Windows 2000/XP/7/10

开发工具：IntelliJ IDEA Ultimate 2020

数据库：MySQL 3.5.6 Redis 3.2.1

JDK: 1.8

Maven:3.6.3

硬件配置：CPU:Pentium200MHz以上

内存：32MB RAM以上

硬盘：320MB以上磁盘空间

显示分辨率：最低800\*600像素建议1024\*768像素及以上

浏览器：Google、火狐、IE等。

**6.1.2 系统部署**

（1）系统构件图：不同的系统构件之间，他们存在着彼此唯一的关系则是构件之间的依赖性。构件依赖性指一个单独的构件依赖于另一个单独的构件。构件依赖性之间用虚线箭头表示。如下图所示，箭头指向的构件表示被依赖的，这说明Cart、Eshop、Checkout都依赖于ShoppingServlet。下图描述是在购物系统中几个构件之间的依赖关系构件图。系统构件图如图1-1所示：

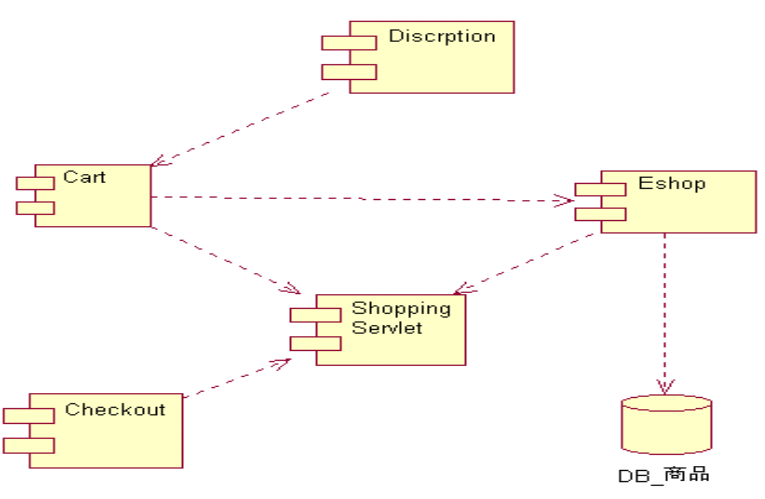


图6-1 系统构件图

（2）系统UML部署图

1、Web Server服务器：一台Web服务器预装4个Linux操作系统的虚拟机，来模拟由于客户访问量大而导致的并发数据处理的流程，需要借助分布式服务器请求的访问方式，同时也是采用前后端分离。1个用来部署B2C Web的前端系统, 1个App Server用来部署B2C Web的后端系统，1个DB服务器用于存储MySQL数据库，1个FTP服务器用于存储系统的图片数据。

2、Web访问量分流设备：根据网站的并发度和用户流量，整个系统架构必须做到对用户访问的流量进行引流。这样做，可以避免由于某个模块故障，而导致的整个项目系统瘫痪的情况。

3、FTP Server And Session Server服务器：网站前台和后台的图片都上传到一个单独的FTP服务器上，并且基于B2C Web下的Session也要备份存储于这个单独的服务器上。

4、DB Server服务器：数据库服务器

5、App Server服务器：控制台程序的服务器，用于与前端系統进行请求交互。

6、Firewall防火墙：防火墙需要对所有的服务器进行过滤保护。

7、User Clien：用户自己的PC端需要预安装浏览器，兼容Google、火狐、IE等浏览器。

整个系统UML部署图如1-2所示：

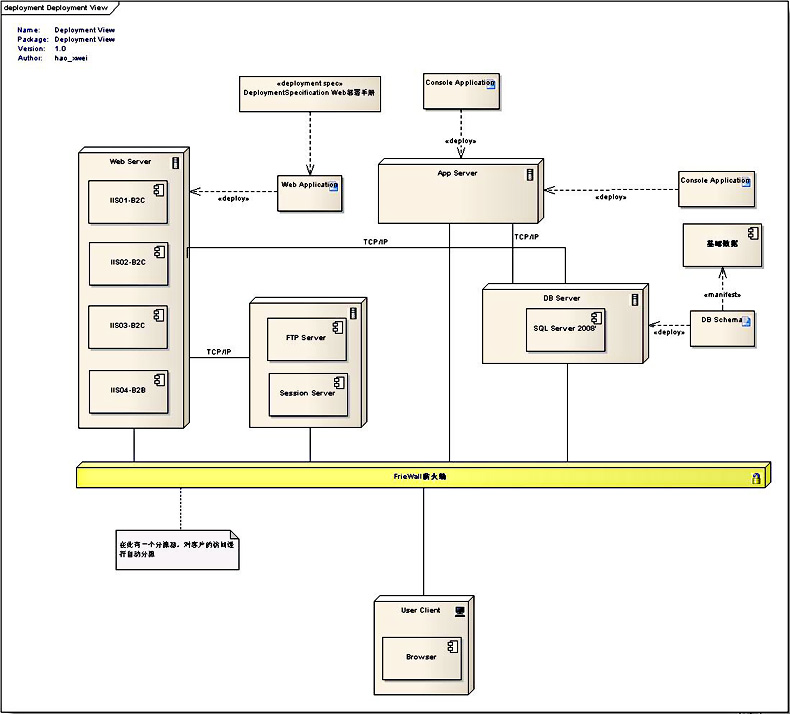


图6-2 系统UML部署图

6.2 XXX系统的应用

**6.2.1 前台购物系统**

（1）登录与注册

1）登录：第一种方式，客户可以通过注册的账号和密码进行登录，同时系统会提示没有注册过的用户进行注册，而对于密码错误的，可以通过手机号进行密码找回（即修改密码）如图2-1。第二种方式，客户也可以通过手机扫码进行第三方登录，避免了注册的流程，使得操作更加的方便，如图2-2所示。

图2-1 账号密码登录 图2-2 手机扫码第三方登录

2）注册：通过用户名进行注册，稍后进行手机号绑定用于找回密码。需要填写自己的用户名昵称，出生年月以及居住地址，然后通过手机号短信进行激活。如图2-3所示：

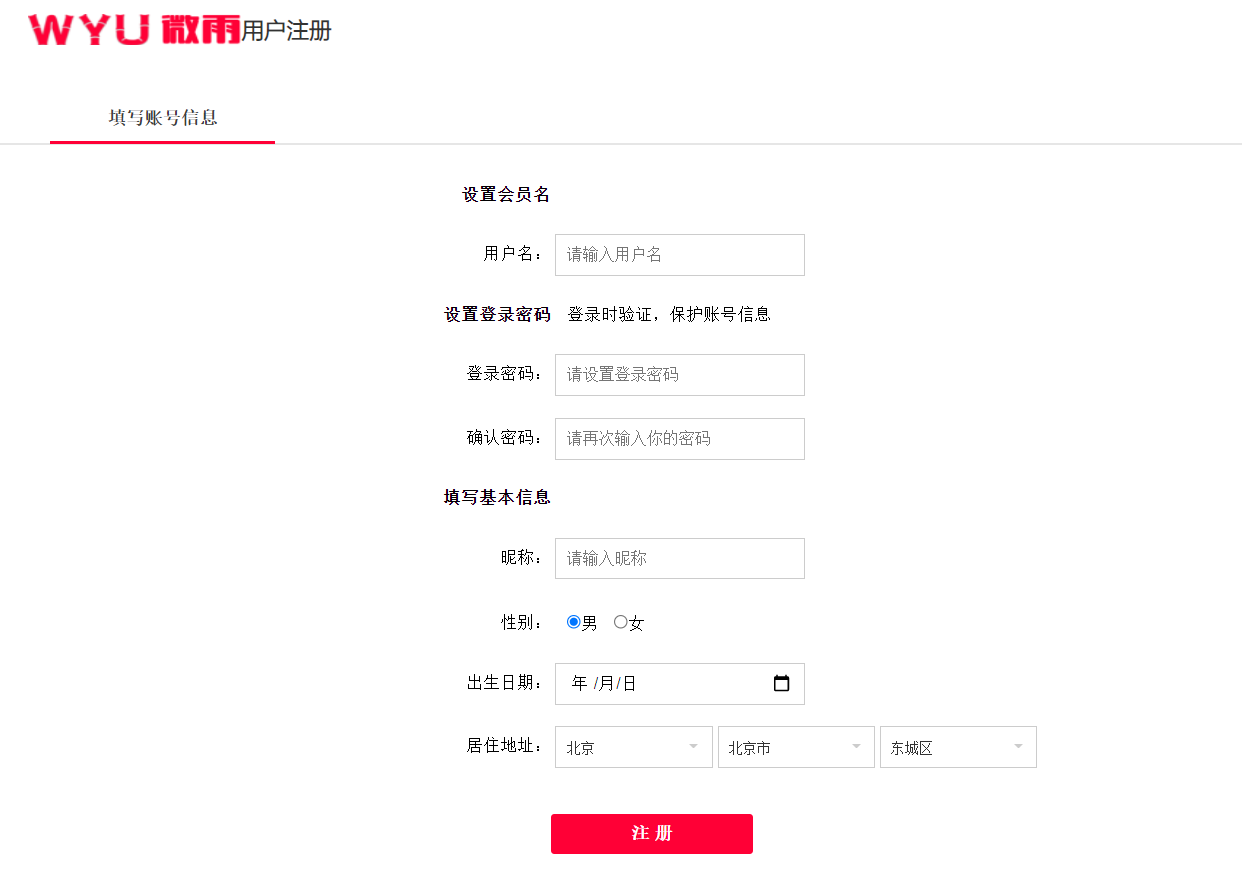


图2-3 前台用户注册

（2）前台商品展示

1）首页展示：首页主要展示商品的分类以及各类商品的品牌，同时也可以通过搜索栏进行全文检索，商品主要显示商品的名称、品牌、价格等信息。如图2-4所示、图2-5所示。

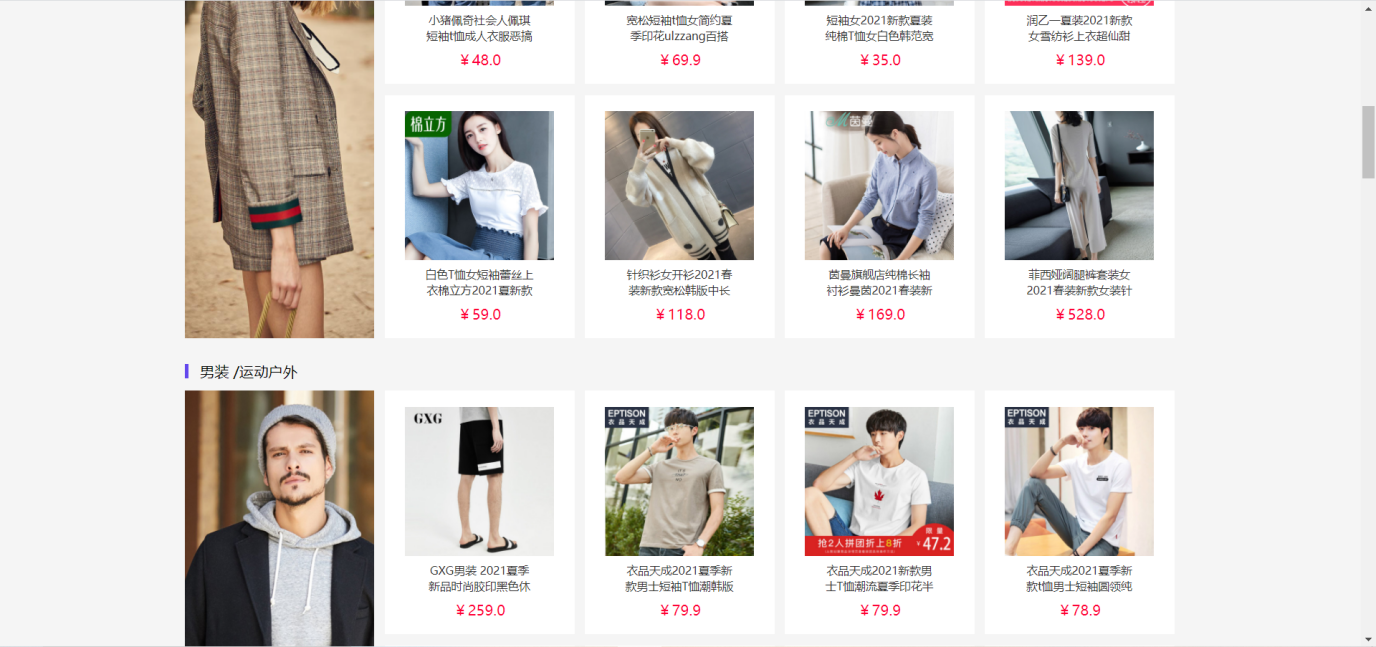


图2-4 商品分类展示

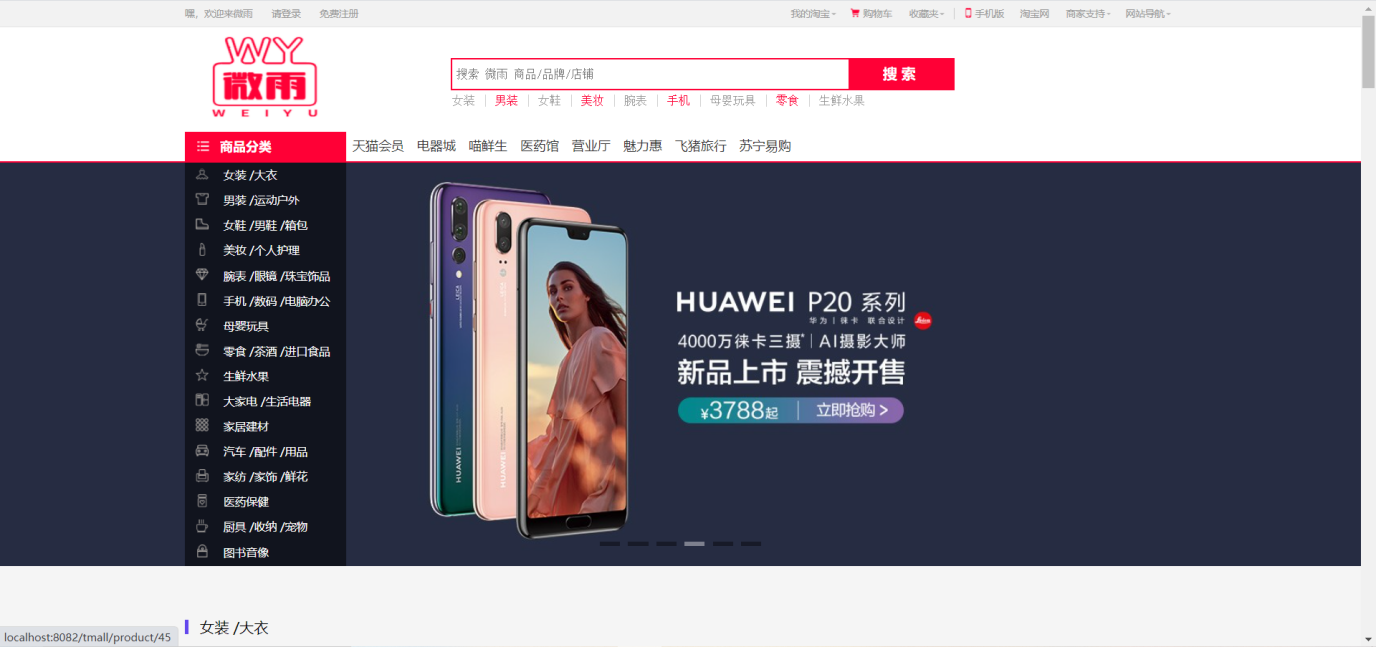


图 2-5 商品分类展示界面

2）商品详情：详情页展示商品的产品种类、属性值、价格、优惠卷、商品的名称还有其他买家对该商品的评价，还有商品的智能推荐模块的展示。如图2-6所示：

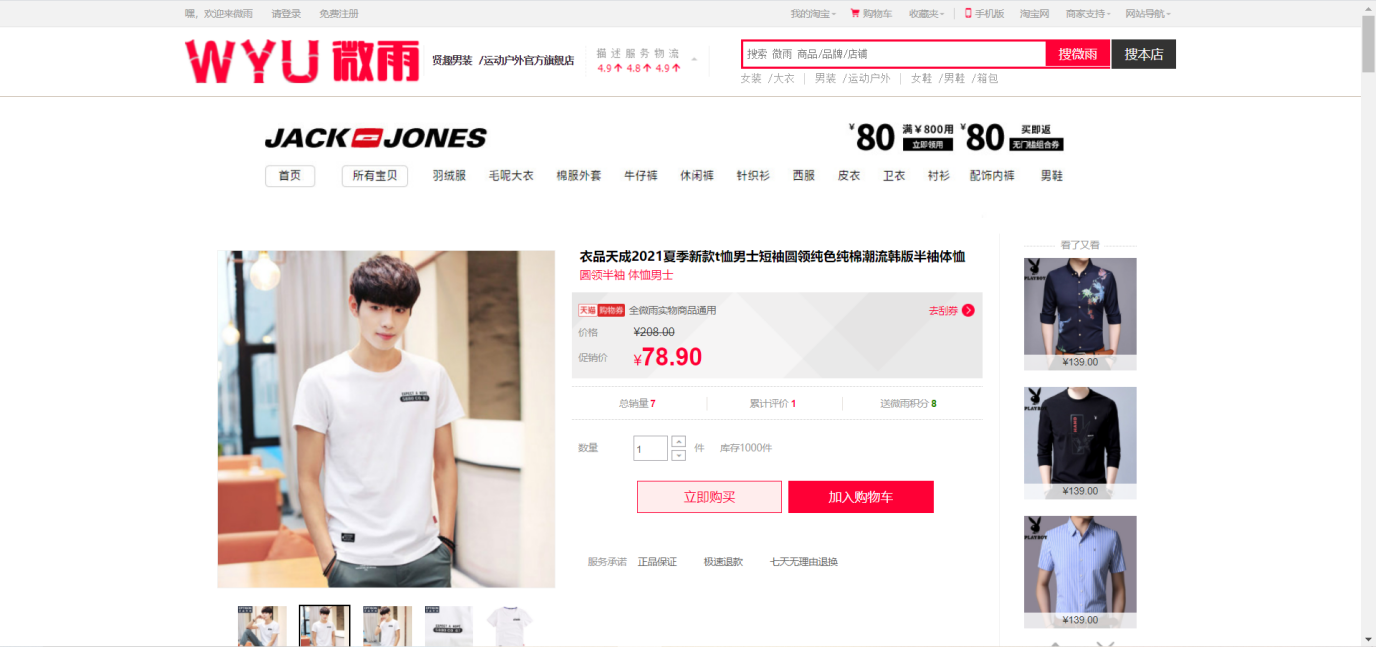


图2-6 商品详情页展示

（3）购物车：可以从主页或者其他地方添加商品到购物车，也可以从购物车删除过个商品或者清空购物车，然后可以从购物车中选中商品生成订单。如图2-7所示：

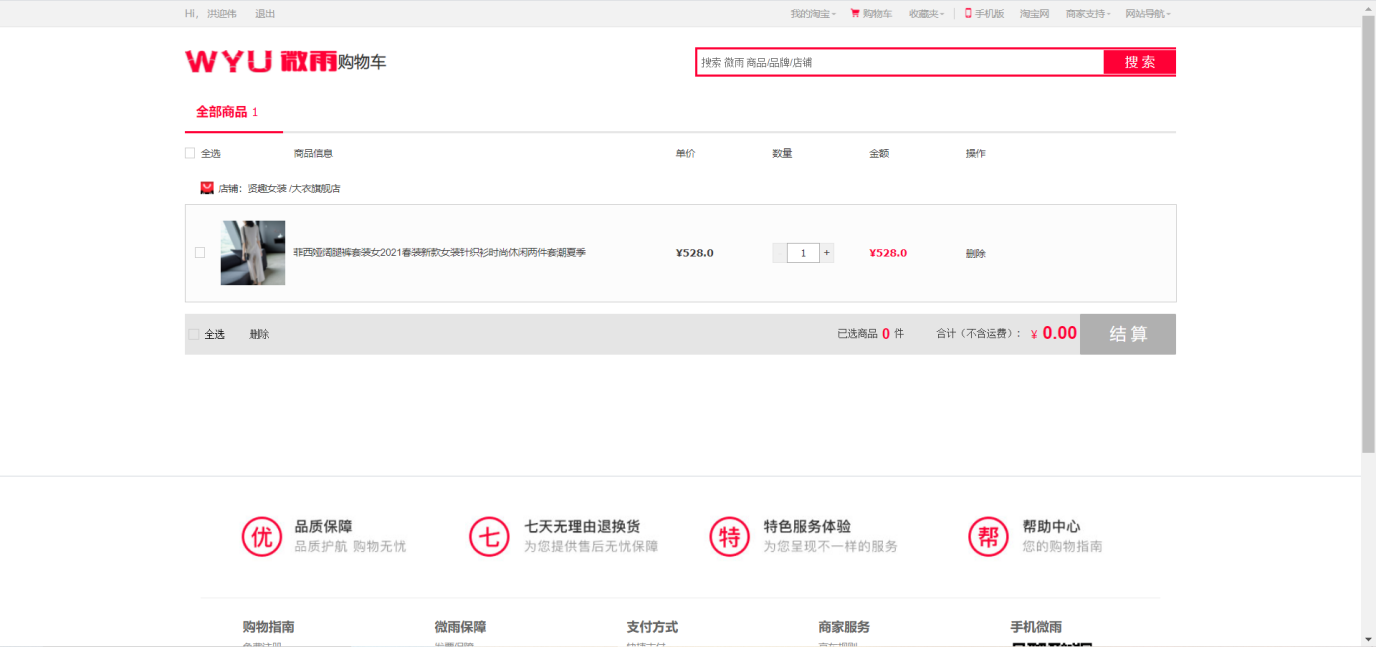


图2-7 购物车界面

（4）生成订单到支付功能（省略支付界面）：填写订单的基本信息，包括签收地址、详细地址、邮政编码、签收人信息、手机号。最后可以选填给商家留言。如图2-8所示。

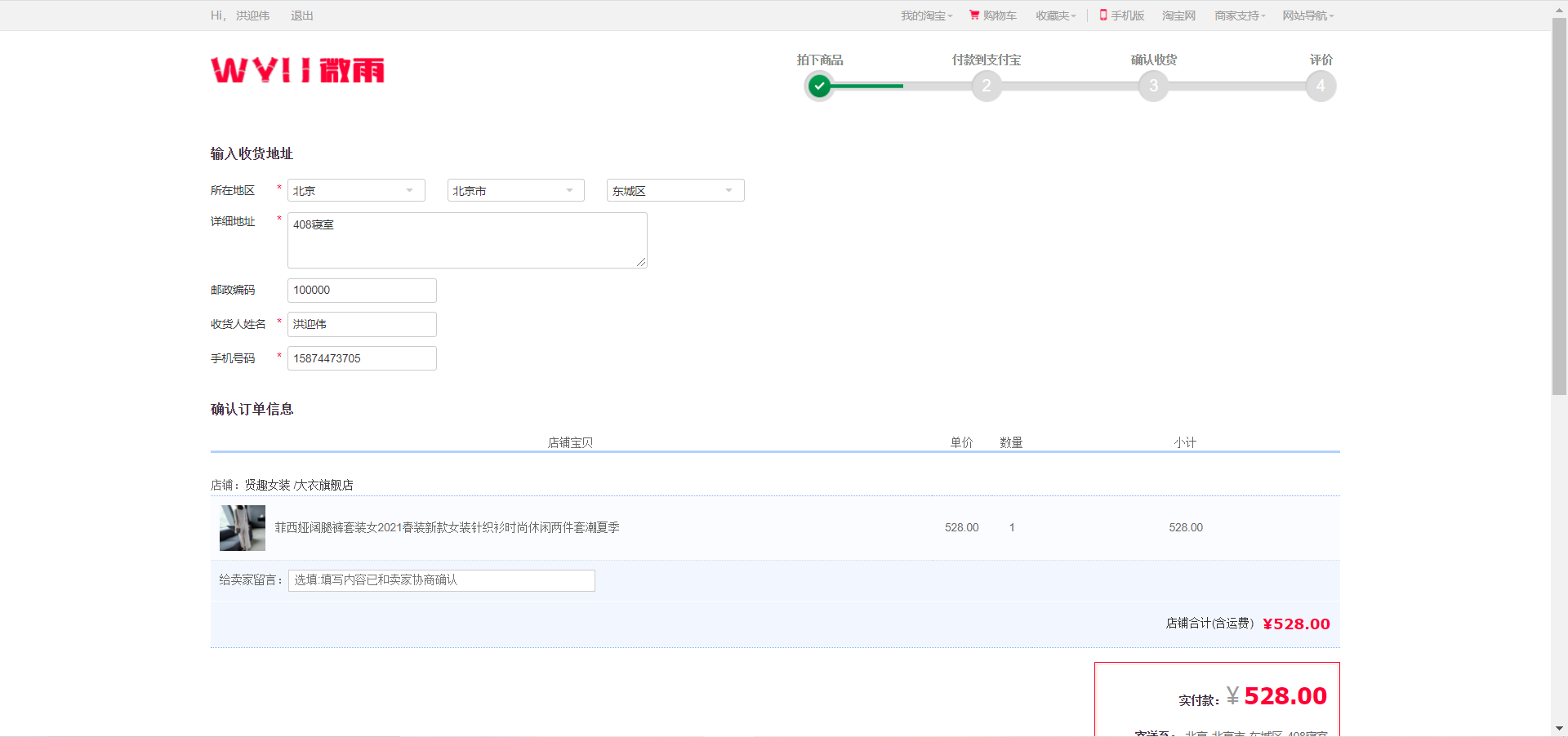


图2-8 生成订单页面

（5）客户的订单状态：包括待付款、待发货、待收货、已完成四个状态。其中已完成中包括待评价以及交易成功的状态。图图2-9所示。

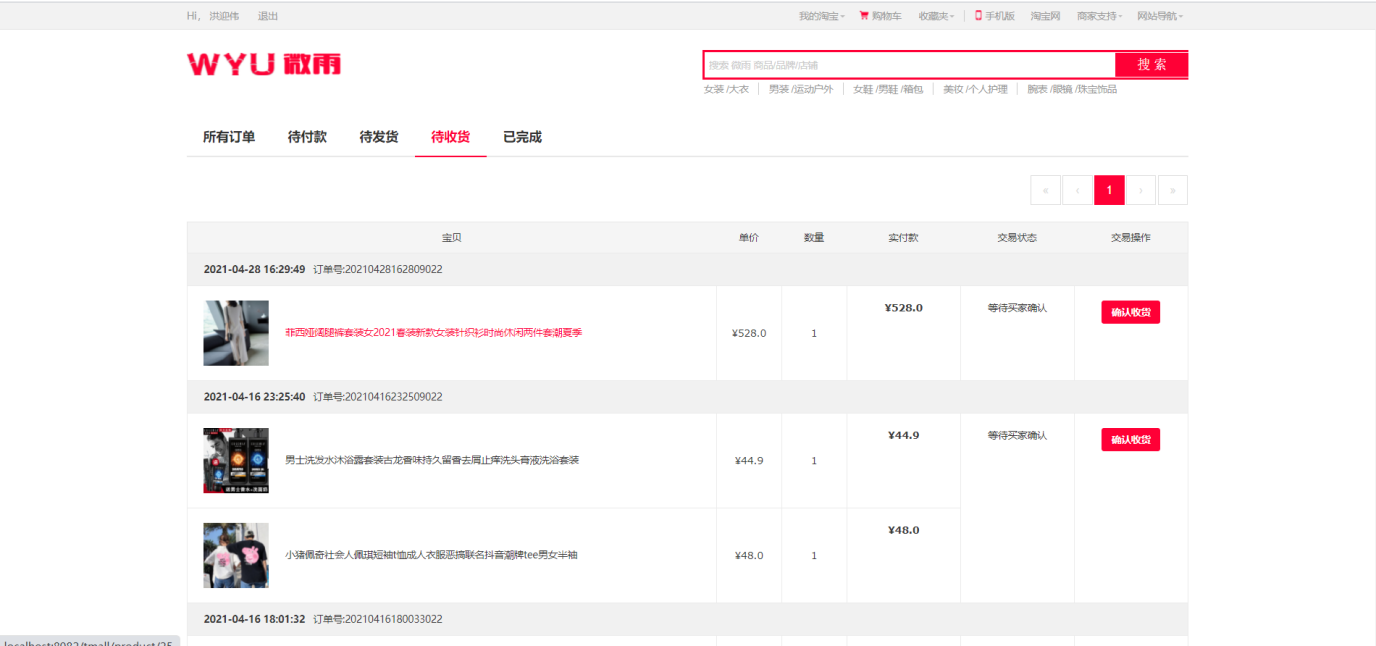


图2-9 客户的订单状态

（6）订单交易成功评价功能：订单交易成功之后，买家可以对商品进行评价，评价信息会更新到商品的详情页中。如图2-10所示。

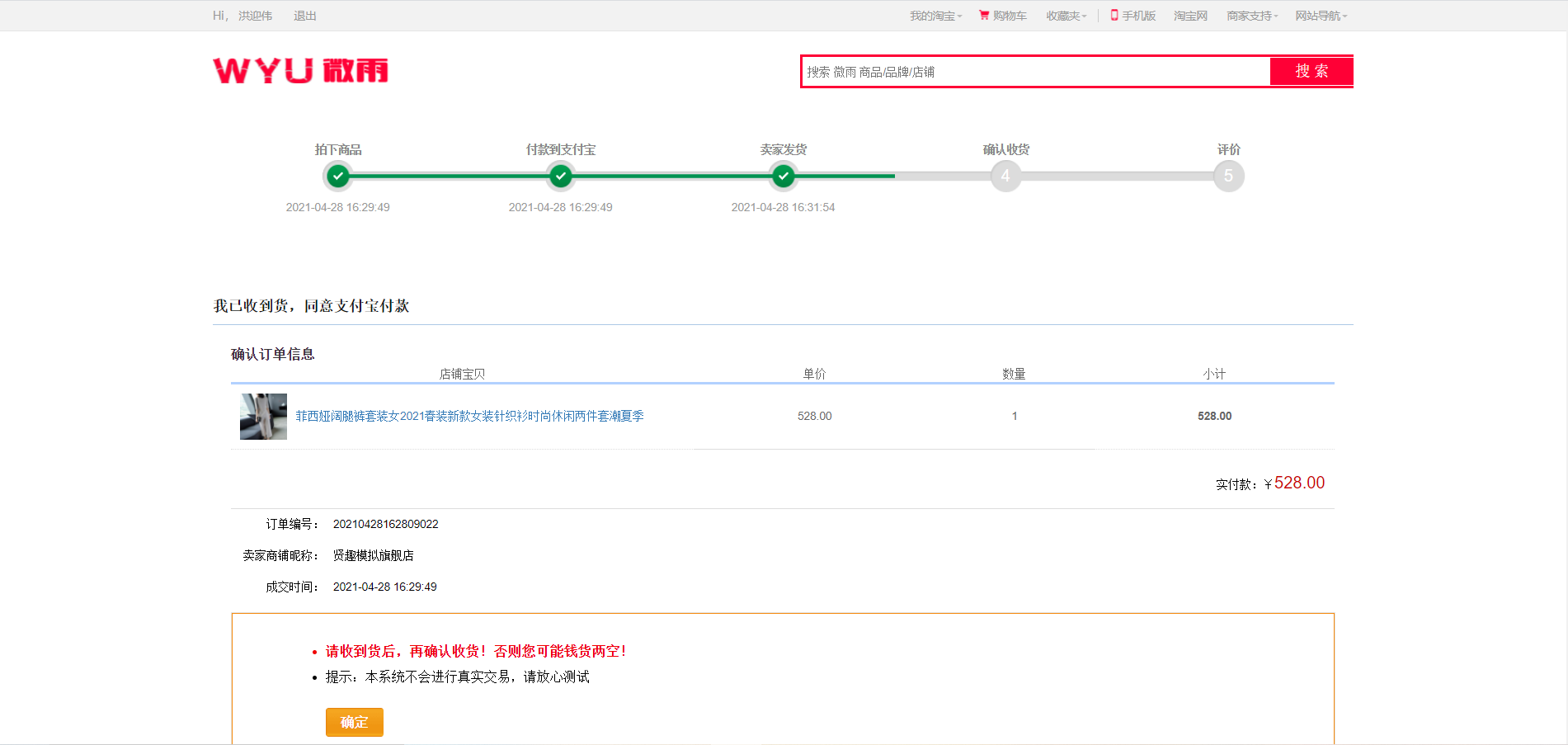


图2-10 商品评价界面

**6.2.2 后台管理系统**

（1）后台数据可视化：数据库可视化主要对上架商品、注册用户、成交订单数进行统计。同时通过线性统计表可视化一周之内的每一天总成交额、交易完成订单数、等待卖家发货、等待买家付款的订单数据的变换趋势。如图2-11所示：

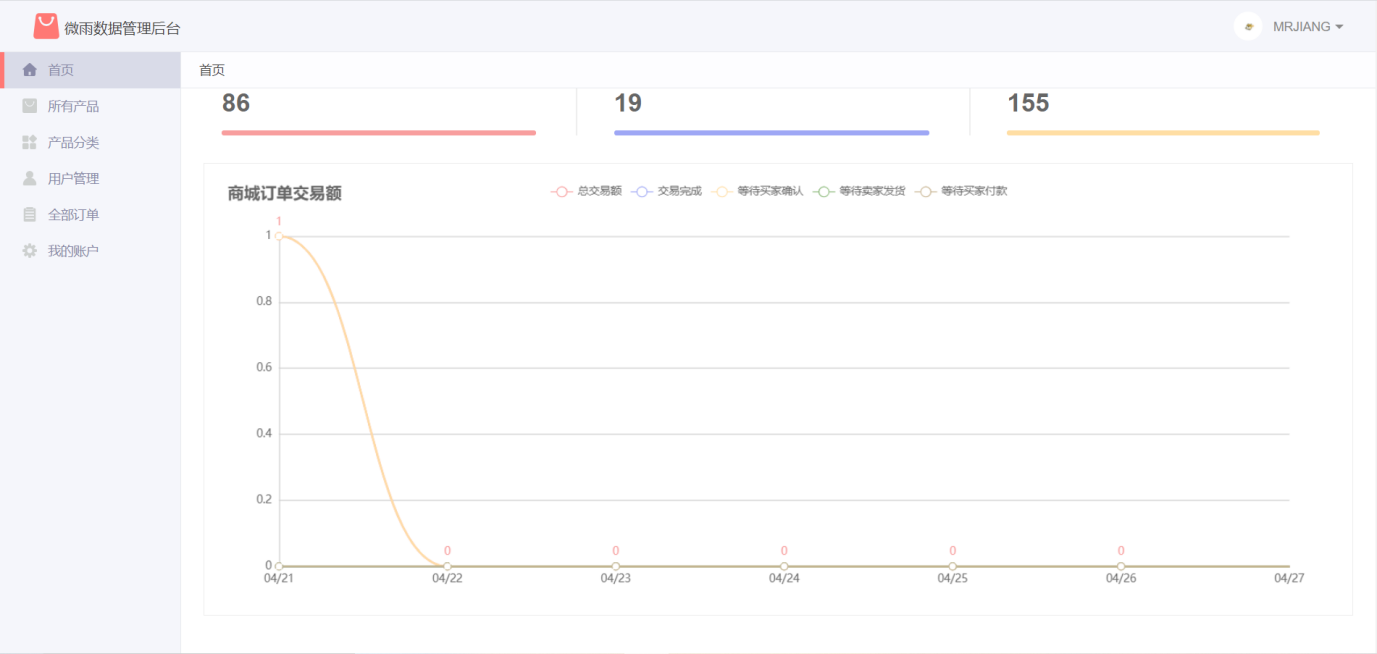


图2-11 后台数据可视化界面

（2）商品管理和产品分类：可以添加商品的类别，或者添加商品的品牌。如图2-12.可以添加商品，或者批量导入商品。如图2-13所示：

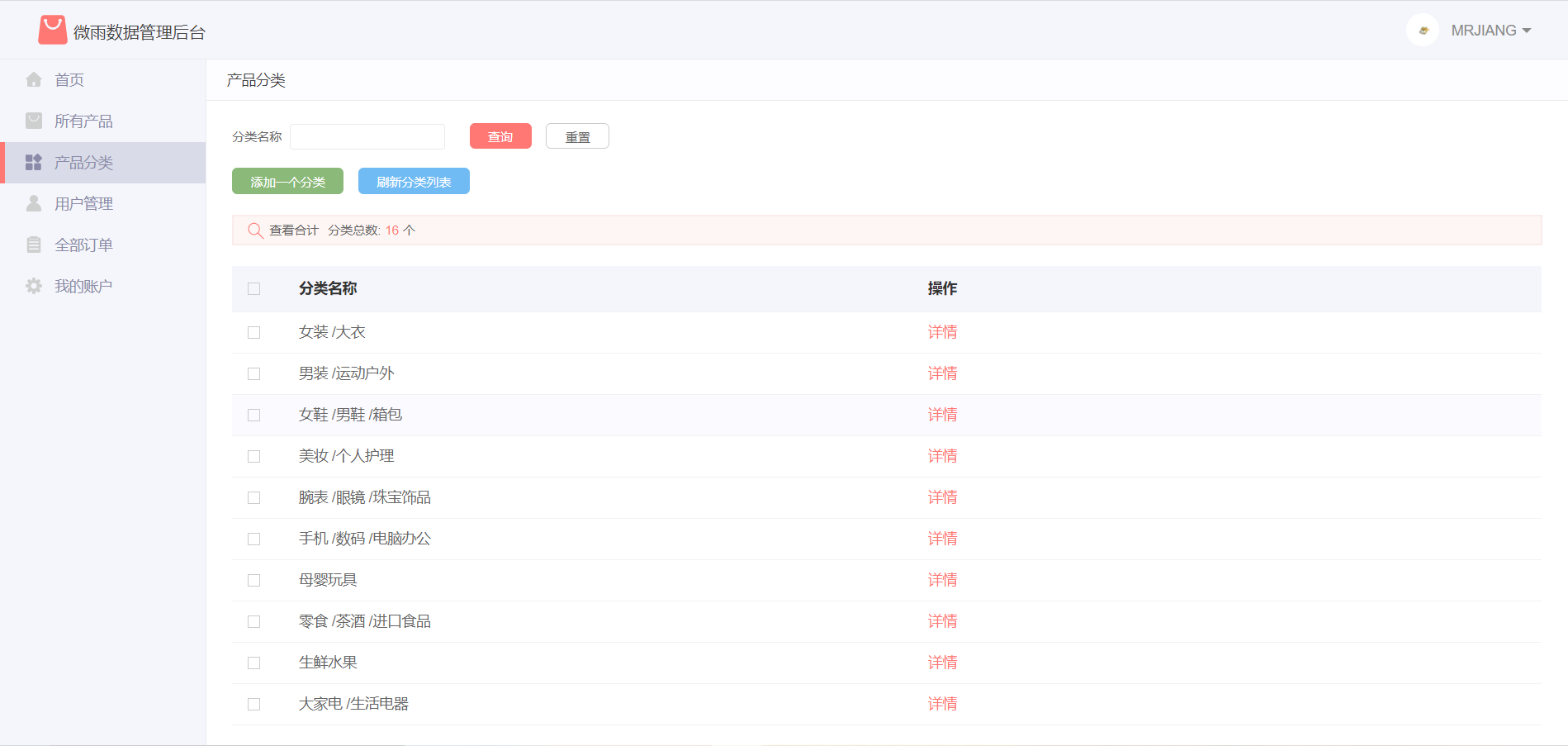


图2-12 商品类别展示界面

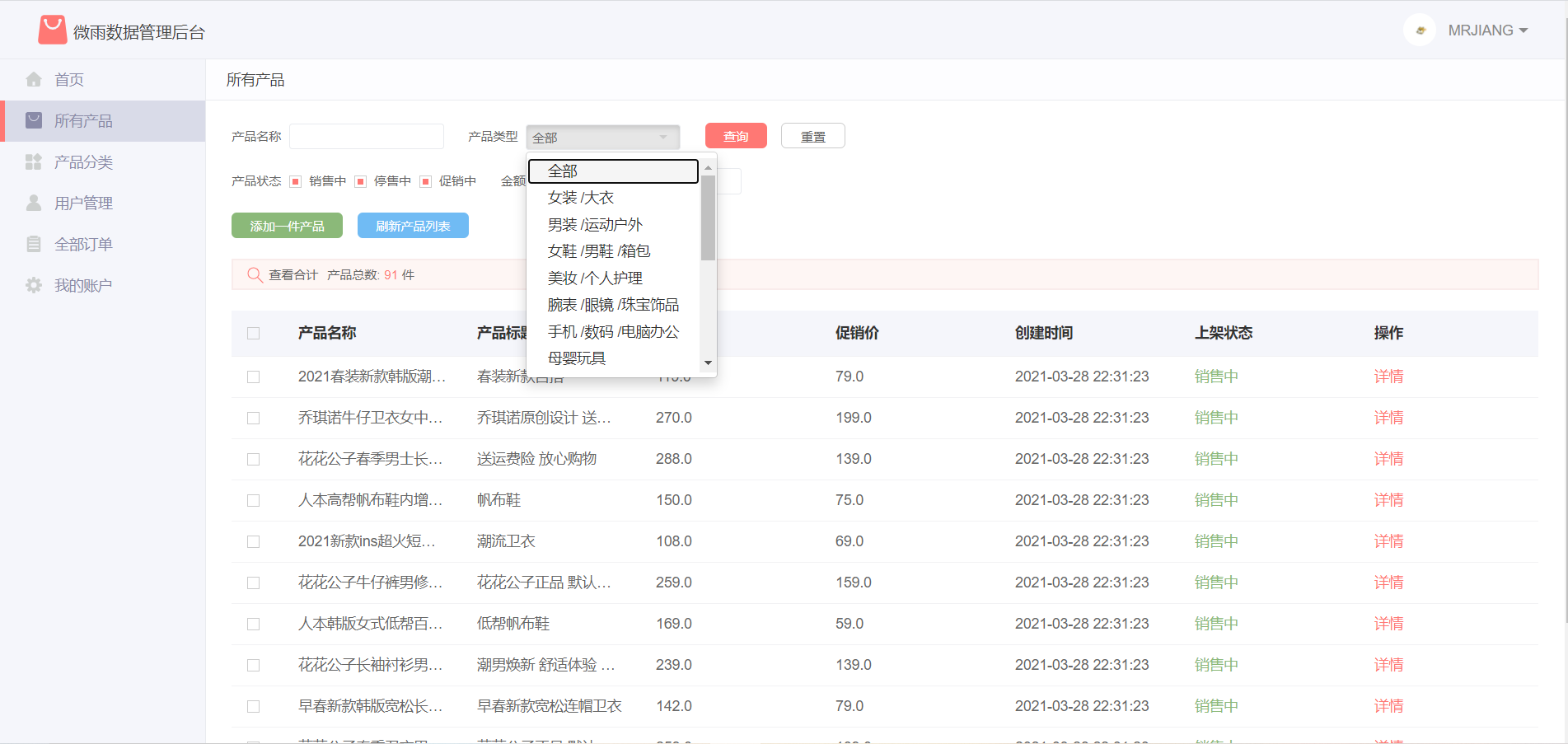


图2-13 商品展示界面

（3）用户管理：对用户的购买情况进行查看，或者对用户的订单进行评价。同时也可以查看用户购物车的情况。如图2-14所示：



图2-14 用户管理界面

（4）订单管理：可以搜索查看订单状态，更新订单状态（对订单进行发货）；可以多次筛选订单。如图2-15所示：

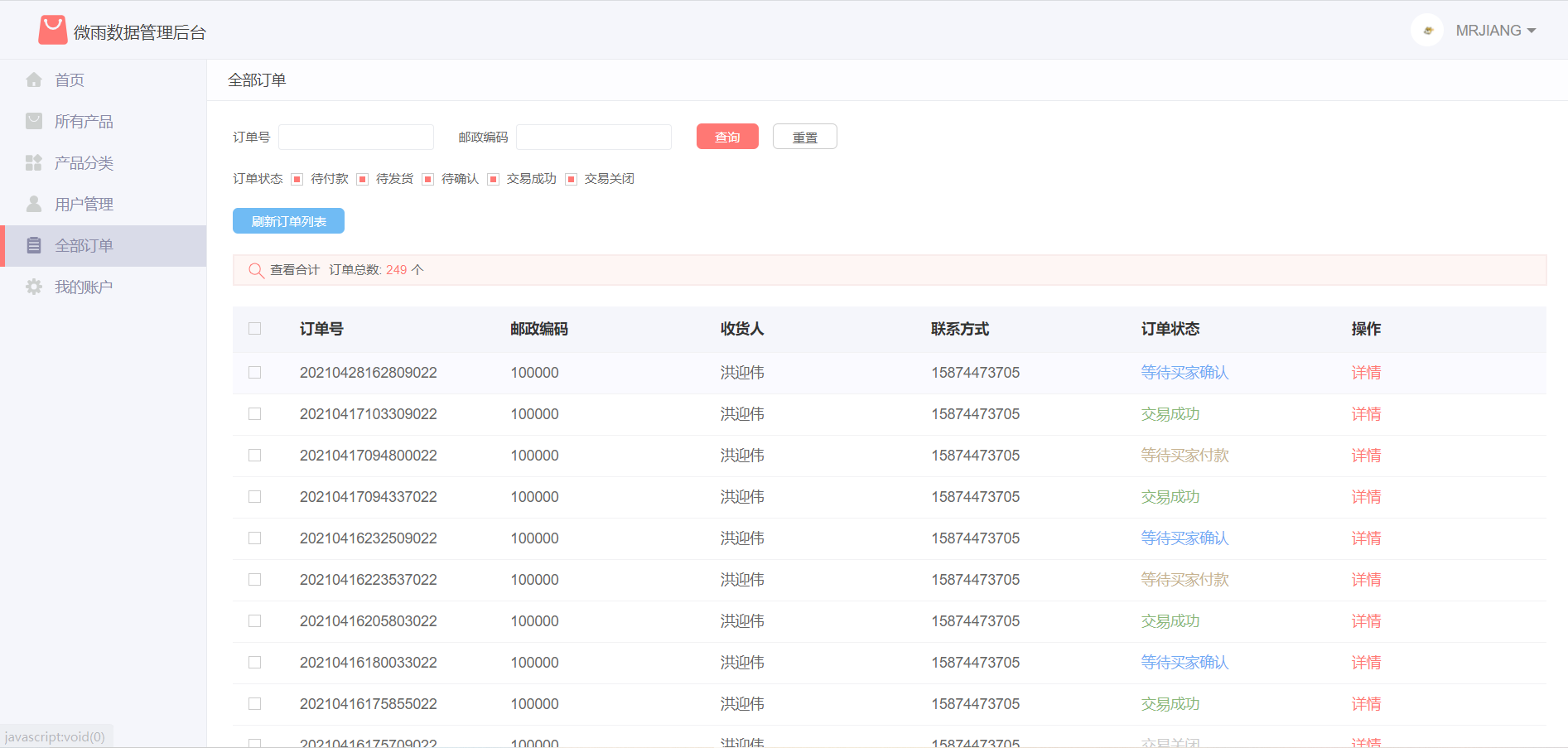


图2-15 订单管理界面

6.3 XXX系统功能性测试

系统功能性测试主要是测试系统中业务流程是否正常、业务过程中的bug或者不足、可接受的并发度，主要包括前台用户注册登录模块、前台用户搜索模块、前台用户添加购物车提交订单模块、用户交易成功的评论模块、支付功能模块。

**6.4.1 前台用户注册登录模块**

用户登录包括账号密码登录和第三方扫描登录，测试由于账号不对是否登录成功、测试由于密码不对是否登录成功、测试验证码不对是否登录成功；测试第三方登录是否获取正确的账号和密码以及账号的基本信息是否有差错。前台用户登录测试用例表见表4-1。前台用户注册测试包括注册用户名是否被应用、设置的密码是否两次都正确、出生年月是否勾选错误、居住地址是否勾选并确认。前台用户注册测试用例表见表4-2。

表4-1 前台用户登录测试用例表

|  |  |
| --- | --- |
| 用例编号 | 001 |
| 用例描述 | 前台用户登录 |
| 测试人员 | 洪迎伟 |
| 用例目的 | 验证用户登录功能 |
| 用例 | 1. 输入未注册的用户名和密码 2. 输入注册的用户名和错误密码 3. 输入注册的用户名、正确密码、错误验证码 4. 输入注册的用户名、正确密码、正确验证码（包括不区分大小写的测试用例） 5. 测试第三方登录时，获取不到账号 6. 测试第三方登录获取的账号信息 |
| 预期结果 | 登录成功，顺利进入商城首页，若登录失败，错误提示 |
| 测试结果 | 1. 无法顺利进入首页，提示用户名未注册，请点击注册 2. 无法顺利进入首页，提示用户名密码错误 3. 无法顺利进入首页，提示验证码错误 4. 顺利进入首页，登录成功 5. 第三方登录加载中，无法登录 6. 顺利进入首页，登录成功 |
| 状态 | 通过 |

表4-2 前台用户注册测试用例表

|  |  |
| --- | --- |
| 用例编号 | 002 |
| 用例描述 | 前台用户注册 |
| 测试人员 | 洪迎伟 |
| 用例目的 | 验证用户注册是否正常 |
| 用例 | 1. 填写已注册的用户名 2. 填写第二次密码错误 3. 填写已注册昵称 4. 选择2021年之前的出生日期 5. 选择居住地没有勾选确认 6. 填写用户名验证通过、密码填写验证通过、出生日期选择正常、居住地选择验证通过 |
| 预期结果 | 注册成功，顺利进入商城登录界面，若注册不通过，错误提示 |
| 测试结果 | 1. 系统发起Ajax验证，用户名已被注册 2. 系统发起捆绑验证，第二次密码错误 3. 系统发起Ajax验证，昵称已被使用 4. 提示日期错误 5. 提示居住地没有确认 6. 系统注册成功，顺利进入登录界面，下一步进行登录 |
| 状态 | 通过 |

**6.4.2 前台用户搜索模块**

测试系统按商品类别、商品品牌、商品名称进行搜索。按类别测试，查看系统是否有其他类别的穿插；按商品品牌测试，查看商品的品牌是否正确，品牌下的商品是否齐全；按商品名称搜索，查看商品的名称是否完全，以及商品之间是否有穿插。前台用户搜索模块测试用例表见表4-3所示。

表4-3 前台用户搜索模块测试用例表

|  |  |
| --- | --- |
| 用例编号 | 003 |
| 用例描述 | 前台用户搜索模块 |
| 测试人员 | 洪迎伟 |
| 用例目的 | 验证前台的搜索功能是否正常 |
| 用例 | 1. 输入“手机”这个类别 2. 输入“手机 华为”这个品牌 3. 输入“华为Pro 30” 4. 输入“华为 Pro40” 5. 输入“华为 RedMi” |
| 预期结果 | 搜索成功，顺利进入搜索商品列表，若搜索不成功，提示没有该类商品。 |
| 测试结果 | 1. 商品列表全是“手机”，出现的条数与数据库录入的该类别一样多 2. 搜索成功，商品列表全是“华为”品牌的手机 3. 搜索成功 4. 搜索成功 5. 搜索错误，提示该品牌不存在该商品 |
| 状态 | 通过 |

**6.4.3 前台用户添加购物车生成订单模块**

主要测试商品加入购物车和从购物车生成订单是否正常。加入购物车测试商品的库存和规格值添加到购物车时是否异常；测试生成订单主要包括生成订单总价格是否正确、购物车是否减少商品、生成的订单商品是否正确等测试用例。前台用户添加购物车和生成订单模块测试用例表见表4-4所示。

表4-4 前台用户添加购物车和生成订单模块测试用例表

|  |  |
| --- | --- |
| 用例编号 | 004 |
| 用例描述 | 前台用户添加购物车生成订单模块 |
| 测试人员 | 洪迎伟 |
| 用例目的 | 测试加入购物车是否正常和提交订单是否正常 |
| 用例 | 1. 选择超过库存的商品数量 2. 选择不足1件的商品数量 3. 选择小数1.5件的商品数量 4. 选择正常的商品规格值 5. 选择正常的商品数量 6. 从购物车中勾选两件商品，生成订单 7. 从购物车中勾选0件商品，生成订单 8. 从购物车中删除商品 9. 清空购物车 |
| 预期结果 | 订单生成成功，顺利进入提交订单页面，若生成不成功，提示相关警告或者错误 |
| 测试结果 | 1. 加入购物车失败，提示库存不足 2. 加入购物车失败，提示商品数量错误 3. 加入购物车失败，提示商品数量错误 4. 加入购物车成功，请查看购物车 5. 加入购物车成功，请查看购物车 6. 生成订单成功，订单总价也正确 7. 生成订单不成功，请选择商品 8. 商品删除成功 9. 购物车已清空，请添加商品 |
| 状态 | 通过 |

**6.4.4 用户交易成功的评论模块**

主要测试系统的用户交易成功之后，对商品进行评论是否更新到商品详情页以及更新到的商品是否正确，与此同时查看评论信息的时间是否正确，内容是否正确（有无乱码），评论对用的用户是否正确。用户交易成功的评论模块用例图表见表4-5所示。

表4-5 用户交易成功的评论模块用例图表

|  |  |
| --- | --- |
| 用例编号 | 005 |
| 用例描述 | 用户交易成功的评论模块 |
| 测试人员 | 洪迎伟 |
| 用例目的 | 用户交易成功之后是否评论商品正常 |
| 用例 | 1、首页添加商品到购物车，从购物车中勾选商品生成订单，然后提交订单，支付完成，待发货，待收货，最后签收，签收成功之后，对商品进行评价。 |
| 预期结果 | 商品评价成功，查看商品详情，商品调加的评论信息正常，评论用户正常，评论时间正常，评论应对的商品也正常。 |
| 测试结果 | 查看商品详情，添加的评论信息无误、评论的用户账号无误、用户评论的时间无误、用户评论的信息也无误  对于评论信息是否乱码测试结果：   1. 英文字母评论通过 2. 中文评论信息通过 3. 其他语言暂未考虑 |
| 状态 | 部分通过，有些功能点有待完善 |

**6.4.5 支付功能模块**

支付功能模块主要是测试用户的支付账号是否获取成功，支付过程中提交的订单信息是否正常，支付过程回调是否正常，支付失败是否会造成订单异常。支付功能模块用例表见表4-6所示。

表4-6 支付功能模块用例表

|  |  |
| --- | --- |
| 用例编号 | 006 |
| 用例描述 | 支付功能模块用例表 |
| 测试人员 | 洪迎伟 |
| 用例目的 | 用户支付成功之后，订单状态是否正常更新 |
| 用例 | 1. 点击支付，查看生成的支付二维码的订单信息是否正常 2. 点击支付，中途断网，扫描失败 3. 点击支付，扫码成功 4. 点击支付，输入账号错误 5. 点击支付，输入账号和密码正确 6. 点击支付，付款成功，回调失败 7. 点击支付，付款成功，回调成功 8. 点击支付，付款成功，查看订单状态 |
| 预期结果 | 付款成功，订单状态为待发货，若付款失败，订单状态不变，订单需重新支付，买家账号内的钱不会扣除。 |
| 测试结果 | 1. 生成的订单信息正常 2. 付款失败，订单状态为待付款 3. 订单支付成功，支付功能进行回调 4. 付款失败，账号密码错误 5. 订单支付成功，支付功能进行回调 6. 付款成功，订单为待发货状态，只是没有完成页面跳转 7. 付款成功，支付成功，订单状态为待发货 8. 付款成功，支付成功，订单状态为待发货 |
| 状态 | 通过，部分功能有点添加或者完善 |

|  |
| --- |
| 第7章 总结与展望 |

7.1 本章参考写法

(\*总结、比较与展望。这一部分要总结自己工作的优点在哪里、不足之处在哪里、进一步的改进方案如何等。如果有类似应用背景的系统，还须比较你的工作与他人工作的优、劣。 \*)

|  |
| --- |
| 参考文献 |

1. 何焕春, 杨怿. 基于MVVM构架的Web前端框架研究[J]. 电脑知识与技术, 2017, 013(024):59-60.
2. 刘义忠, 张伟. 基于SSM框架的后台管理系统设计与实现[J]. 软件导刊, 2019, 18(02):74-77.
3. 胡峰. 基于JAVA的B2C电子商城设计与实现[D]. 西安: 西安电子科技大学, 2017.
4. 刘金昇. 基于SSH的校园电子商城设计与实现[D]. 内蒙古: 内蒙古大学,2017.
5. 王昶太. 基于Spring与Hibernate的智能电子商城系统的设计与实现[D]. 北京:北京邮电大学, 2015.
6. 赵贝贝, 梅一垚.基于MVC的校园电子商城的设计与实现[J]. 电子技术与软件工程, 2015(01): 85-88.

|  |
| --- |
| 致 谢 |