|  |
| --- |
| 目 录  [摘 要 I](#_Toc129554601)  [**Abstract** II](#_Toc129554602)  [第1章 前 言 1](#_Toc129554603)  [1.1 本项目的研究背景及意义 1](#_Toc129554604)  [1.2 关于本项目的国内外研究现状分析（二级标题） 2](#_Toc129554605)  [1.3 项目的开发范围及目标（二级标题） 2](#_Toc129554606)  [1.4 论文结构简介（二级标题） 2](#_Toc129554607)  [第2章 技术与原理 3](#_Toc129554608)  [2.1 系统开发相关技术介绍 3](#_Toc129554609)  [2.2 系统开发工具介绍 4](#_Toc129554610)  [第3章 网吧管理系统的需求建模 6](#_Toc129554611)  [3.1 章节介绍 6](#_Toc129554612)  [3.2 网吧管理系统角色定义 6](#_Toc129554613)  [3.3 网吧管理系统功能性需求分析 7](#_Toc129554614)  [3.4 网吧管理系统非功能性需求分析 15](#_Toc129554615)  [第4章 概要设计 16](#_Toc129554616)  [4.1 系统架构设计及原理 16](#_Toc129554617)  [4.2 功能模块设计 17](#_Toc129554618)  [4.3 数据库设计 21](#_Toc129554619)  [第5章 模块设计 27](#_Toc129554620)  [5.1 登录注册模块的设计与实现 27](#_Toc129554621)  [5.2 用户管理模块的设计与实现 30](#_Toc129554622)  [5.3 会员卡管理模块的设计与实现 32](#_Toc129554623)  [5.4 清单管理模块的设计与实现 35](#_Toc129554624)  [5.5 设备管理模块的设计与实现 38](#_Toc129554625)  [5.6 商品管理模块的设计与实现 40](#_Toc129554626)  [5.7 电脑管理模块的设计与实现 43](#_Toc129554627)  [第6章 部署和应用 46](#_Toc129554628)  [6.1 本章参考写法 46](#_Toc129554629)  [6.1 XXX系统的部署 46](#_Toc129554630)  [6.2 XXX系统的应用 48](#_Toc129554631)  [6.3 XXX系统功能性测试 55](#_Toc129554632)  [第7章 总结与展望 60](#_Toc129554633)  [7.1 本章参考写法 60](#_Toc129554634)  [参考文献 61](#_Toc129554635)  [致 谢 62](#_Toc129554636) |

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；

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

**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.

**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）**

网吧就是向社会公众开放的营利性上网服务提供场所,社会公众可利用  
网吧内的计算机及上网接入设备等进行网页浏览、学习、网游、聊天、视频  
听音乐或其她活动,网吧经营者通过收取使用费或提供其她增值服务获得收  
入。网吧就是目前阶段最适合年轻人消费的休闲场所,也就是培养网络应用的  
最好的平台。

网吧计费管理系统就是在国内外先进计费思想与管理理念基础上进行创新,  
经多年努力造就而成的专业计费管理系统,主要面向网吧、机房、电子阅览室由于  
其在计费管理上所体现的突出的优越性,可满足您既想实现轻松管理,又想实现效  
益最大效益可持续化的多重需要,成为了网吧机房经营管理人员的理想选择,就是  
打造品牌网吧与机房的前提。

人工计时收费就是非常繁琐的一项工作,耗费人力物力,而极容易出错  
后的统计工作更就是一项费时费力的苦差事,甚至不能统计。而且,上机过程中  
每个网吧都需要值班人员值班,耗费了管理人员的很多时间。计算机硬件的发展再  
快也离不开计算机软件的支持计算机软件的滞后,必然会形成使计算机硬件的发  
展成为泡沫性发展!面对如此庞大的消费群体,网吧必然要面对有条不素的收费  
管理的困扰,如何保证网吧内部的计算机能够正常的运行,并且满足不同人群的需  
求,并且保证不同区域不同时段的环境下,如何针对不同的计费价格准确无误的统  
一完成收费。开发一个合适的网吧计费系就成了必须。  
本系统主要针对网吧业主使用的方便为到网吧上网的人员提供方便,以便吸  
引更多的顾客,应做到各种会员卡的设计、可用资源的开放与计费方式的合理,全  
自动管理网吧内部的所有事件,让网吧业主轻松自如的经营具体应表现在下面几  
个方面:

* 减少值班人员、维护人员,降低管理人员费用
* 增加网吧的开放时间,提高设备的利用率。
* 通过计费系统代替的手工计费,而且实现了对数据的自动统计与查询功  
  能,使网吧的管理基本上实现了自动化
* 提供工作人员账号,对没有账号码或无效的用户则拒绝使用,开放的目的  
  就是为了方便网管人员对于某些计算机的调试与整理。

系统通用性,兼顾手工、会员卡等其它收费方式。

**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）

本文从网吧管理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，依据该网吧管理系统的需求分析，完成了XXX网吧管理系统的设计与开发，实现了XXXX功能。本论文以软件开发的大致流程来介绍本网吧管理系统，具体内容如下：

第一章节是前言，讲述网吧管理系统的开发的背景和意义，综述了国内外关于网吧管理网站的现状，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 技术与原理

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)）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)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)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)，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)支持、各类版本工具([git](https://baike.baidu.com/item/git/12647237?fromModule=lemma_inlink)、[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

3.1 章节介绍

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、网管和用户。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；网管进入登录页面，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、网管管理、清单列表等用例，用例图如图3.2.1所示：

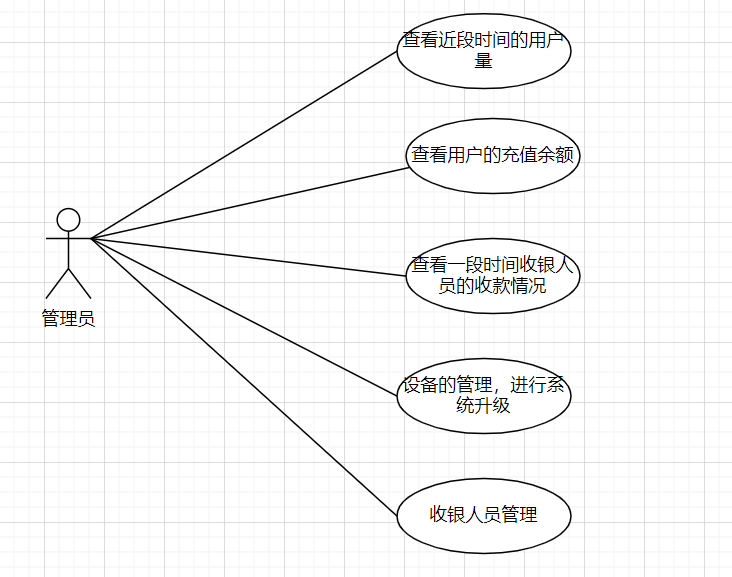


图3.2.1 管理员用例图

2．网管：主要包括开卡/充值、用户管理、会员卡管理等用例，用例图如图3.2.2所示：



图3.2.2 网管用例图

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3.2.3所示：

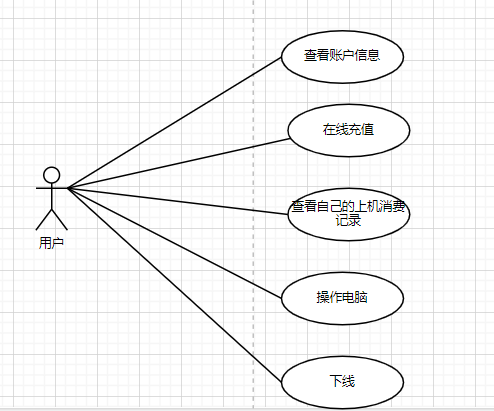


图3.2.3 用户用例图

3.3 网吧管理系统功能性需求分析

**3.3.1 用户登录和注册功能的需求分析**

用户登录和注册包含网管和用户两个角色，故分两部分分别分析。其中网管包含注册、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 3.3.1 所示。

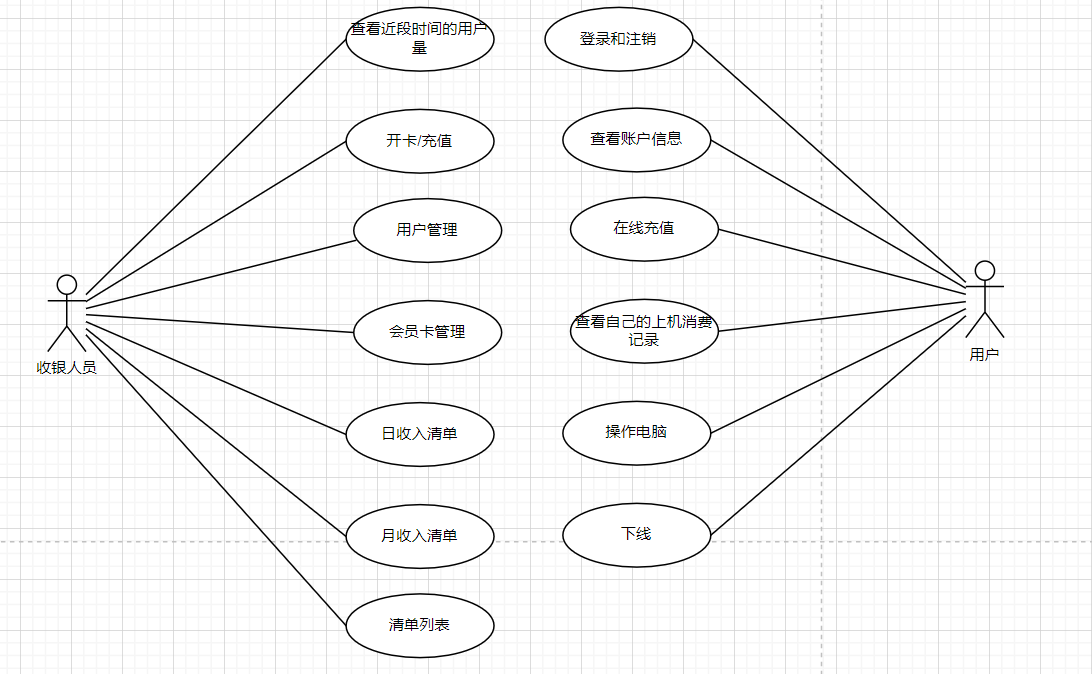


图3.3.1 用户登录注册用例

用户登录和注册功能需求分析如下：

1. 网管管理功能。
2. 网管注册和注销功能：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3-3-1 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 | 2023/03/12 |
| 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 | | |

**3.3.2 用户管理功能的需求分析**

用户管理包含对用户的搜索、用户的详情展示、用户的编辑、用户的删除、用户的启用/禁用等功能，而这些功能都必须已登录的情况下才能操作。

用户管理的用例图如图 3.3.2 所示。

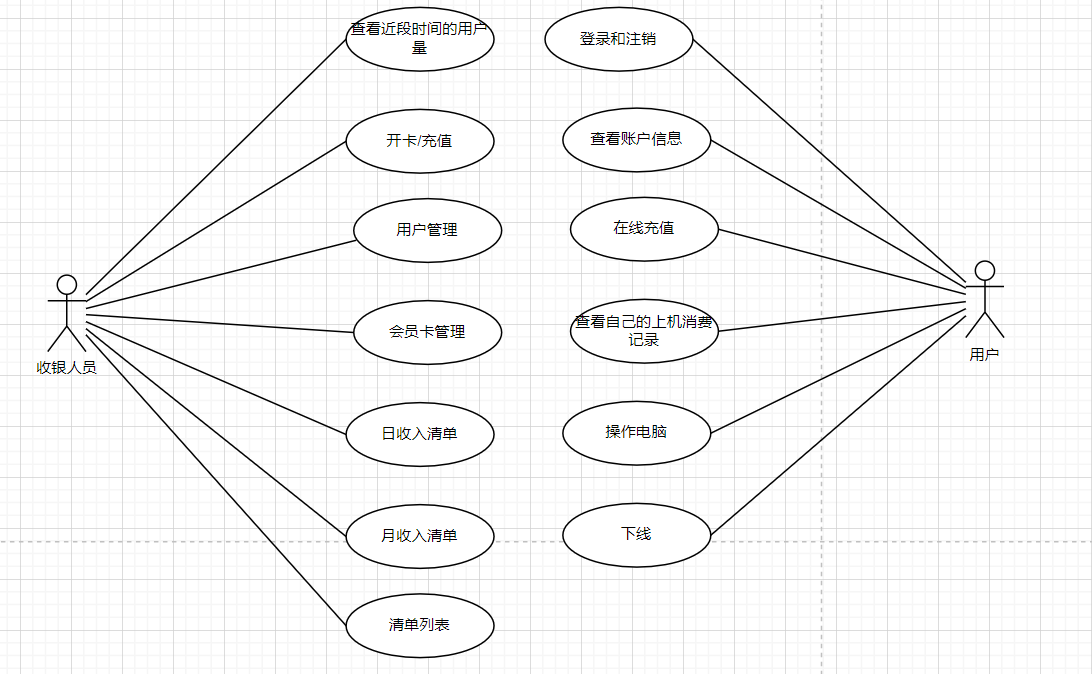


图3.3.2 用户管理用例

用户管理功能需求分析如下：

（1）网管管理功能。

1）用户的搜索功能：网管登录后，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）用户详细信息的展示：网管在登录之后，只能粗略的查看用户信息，详细的用户账户信息需要选择指定用户才能查看。

3）用户的启动/禁用：网管在登录后，可以对用户的信息进行操作，如果对用户账户禁用，则用户登录时，会阻断式弹窗，用户账户被禁用，请联系网管，启用后才能正常登录。

表3-3-2 用户登录用例描述

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | ZYZ01 | 用例名称 | 用户管理 |
| 编制人 |  | 编制日期 | 2022/02/08 |
| 用例描述 | 此用例主要描述网管对用户的管理 | | |
| 参与者 | 网管 | | |
| 前置条件 | 网管已登录 | | |
| 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. 网管进入登陆界面 2. 网管输入账号密码后点击登录 | | | |
| 可选事件流  2a. 网管对用户账户的修改  2b. 网管对用户账户的查阅  网管对用户账户的禁用 | | | |
| 特殊需求 | 无 | | |
| 扩展点 | 无 | | |

**3.3.3 会员卡管理功能的需求分析**

会员卡管理包含对用户上机、用户下机、会员卡充值、会员卡退费、用户卡详情、会员卡的搜索等功能，而这些功能都必须已登录的情况下才能操作

会员卡管理功能的用例图如图 3-3-3 所示。

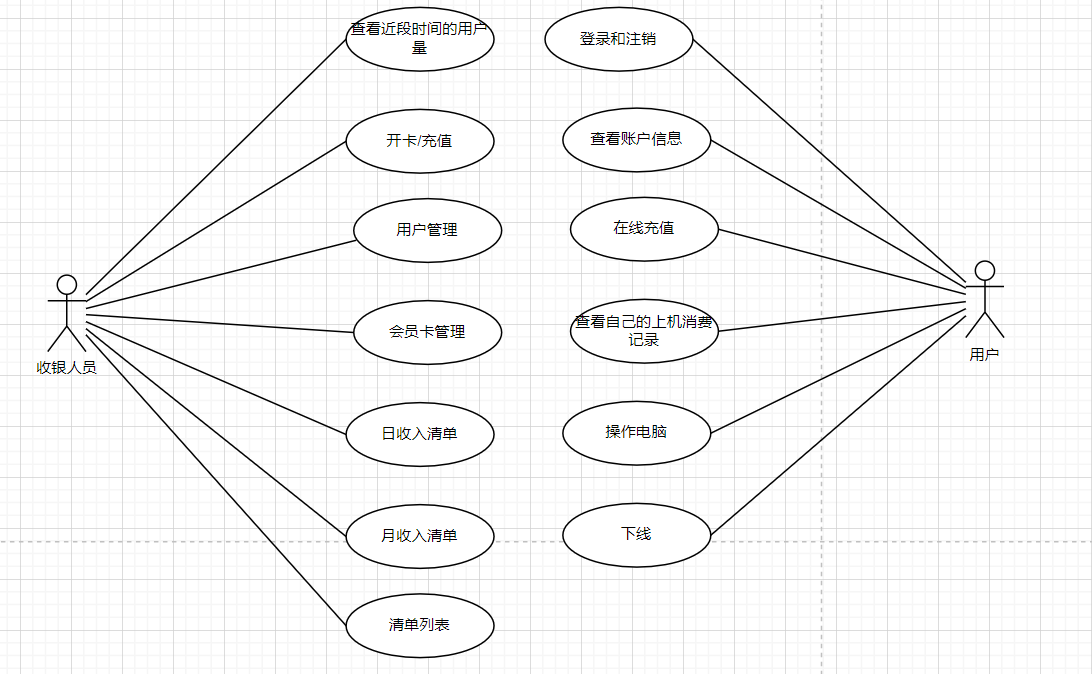


图3.3.3 XXX用例

会员卡管理功能需求分析如下：

（1）会员卡管理功。

1）用户上机：普通用户可以自行上机、下机，而临时卡用户没有上机权限，只能由管理员替临时卡用户下机。

2）会员卡充值：网管在登录之后，可以帮用户充值，用户也可以自行充值。

3）会员卡退费：用户可以联系网管进行退费，可以退回剩余余额的全部现金

表3.3.3 用户登录用例描述

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | ZYZ01 | 用例名称 | 用户管理 |
| 编制人 |  | 编制日期 | 2022/02/08 |
| 用例描述 | 此用例主要描述网管对会员卡的管理 | | |
| 参与者 | 网管 | | |
| 前置条件 | 网管已登录 | | |
| 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. 网管进入登陆界面 2. 网管输入账号密码后点击登录 | | | |
| 可选事件流  2a. 网管对用户上机  2b. 网管对用户的上机  2c. 会员卡充值  2d.会员卡退费  2c.用户卡详情  2d.会员卡的搜索 | | | |
| 特殊需求 | 无 | | |
| 扩展点 | 对用户的充值 | | |

**3.3.4 商品管理功能的需求分析**

商品管理包含对展示商品、商品上新、移除商品、更新商品等功能，这些功能是网管登录后才能操作的，用户可以花费指定金币购买商品，由网管配送。

商品管理功能的用例图如图 3.3.4 所示。

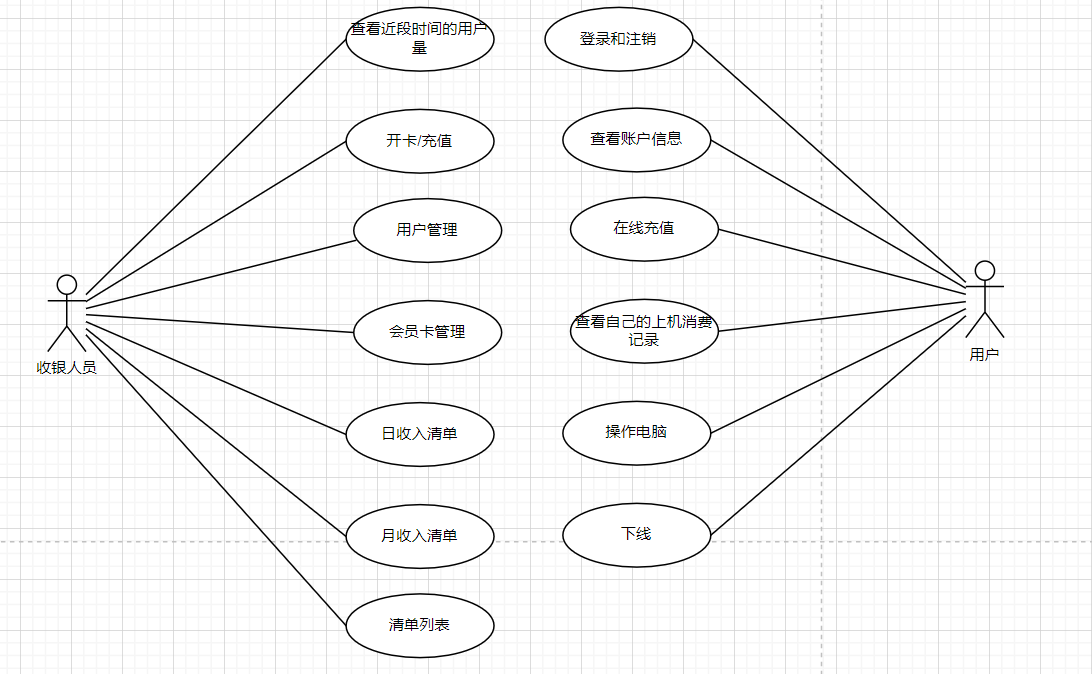


图3.3.4 XXX用例

商品管理功能需求分析如下：

1. 商品管理功能。
2. 商品上新：网管登录后，可以填写商品后，新增商品。
3. 商品更新：网管登录后，做活动需要更新商品，修改商品的数据

表3.3.4 用户登录用例描述

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | ZYZ01 | 用例名称 | 用户管理 |
| 编制人 |  | 编制日期 | 2022/02/08 |
| 用例描述 | 此用例主要描述网管对会员卡的管理 | | |
| 参与者 | 网管 | | |
| 前置条件 | 网管已登录 | | |
| 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. 网管进入登陆界面 2. 网管输入账号密码后点击登录 | | | |
| 可选事件流  2a. 展示商品  2b. 商品上新  2c. 移除商品  2d. 更新商品 | | | |
| 特殊需求 | 无 | | |
| 扩展点 | 无 | | |

**3.3.5 清单管理功能的需求分析**

清单管理包含对日收入清单，月收入清单，清单列表等功能，这些功能是网管登录后才能查阅的，网管可以查看当天的收入或者当月的收入情况，以及清单列表。

清单管理功能的用例图如图 3.3.5 所示。，否则

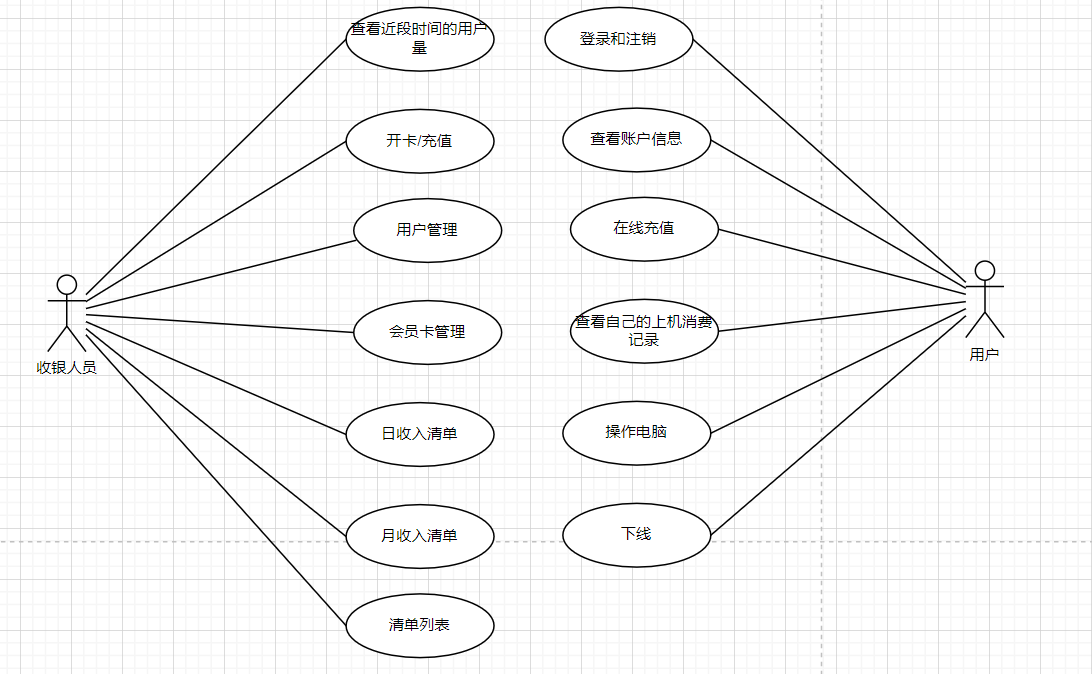


图3.3。5 XXX用例

清单管理功能需求分析如下：

1. 清单管理功能。
2. 日收入清单：网管登录后，可以查看当天为用户充值的记录。
3. 月收入清单：网管登录后，可以查看当月为用户充值的记录
4. 清单列表：网管登录后，可以查看所有为用户充值的记录

表3.3.5 用户登录用例描述

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | ZYZ01 | 用例名称 | 用户管理 |
| 编制人 |  | 编制日期 | 2022/02/08 |
| 用例描述 | 此用例主要描述网管对清单子系统的管理 | | |
| 参与者 | 网管 | | |
| 前置条件 | 网管已登录 | | |
| 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. 网管进入登陆界面 2. 网管输入账号密码后点击登录 | | | |
| 可选事件流  2a. 展示商品  2b. 商品上新  2c. 移除商品  2d. 更新商品 | | | |
| 特殊需求 | 无 | | |
| 扩展点 | 无 | | |

3.4 网吧管理系统非功能性需求分析

网吧管理系统是管理员和网管对网吧的管理和用户操作的电脑的系统。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。用户对软件质量属性、运行环境、资源约束、外部接口等方面的要求或期望，包括：

1.性能需求：用户在软件响应速度、结果精度、运行时资源消耗量等方面的要求。

2.可靠性需求：用户在软件失效的频率、严重程度、易恢复性，以及故障可预测性等方面的要求。

3.易用性需求：用户在界面的易用性、美观性，以及对面向用户的文档和培训资料等方面的要求。

4.安全性需求：用户在身份认证、授权控制、私密性等方面的要求。

5.运行环境约束：用户对软件系统运行环境的要求。

6.外部接口：用户对待开发软件系统与其他软件系统或硬件设备之间的接口的要求。

第4章 概要设计

本章基于在网吧管理系统的需求分析，给出网吧管理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 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：

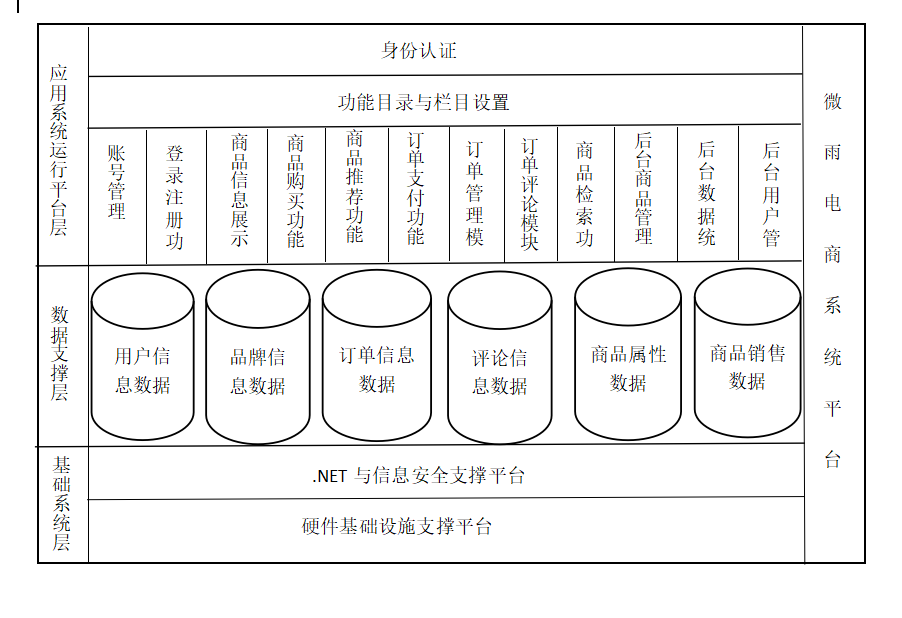


图4-1 系统架构设计

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

网吧管理系统主要的功能模块包括用户登录注册模块、用户管理模块、会员卡管理模块、清单管理模块、设备管理模块，商品管理模块、电脑管理模块等，除此之外还有后台数据可视化模块。以下是系统核心功能模块结构图如图4-2所示：

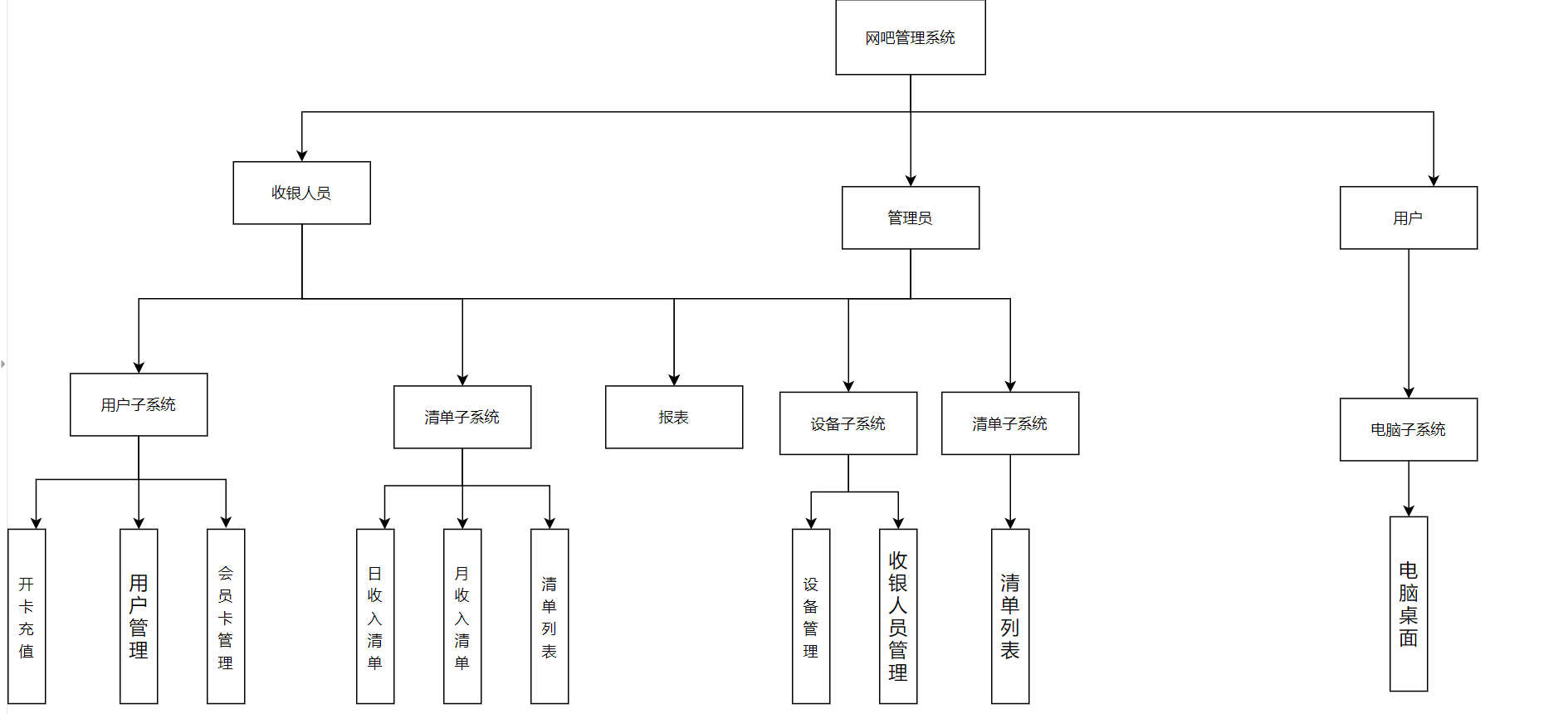


图4-2 核心功能模块结构图

**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。

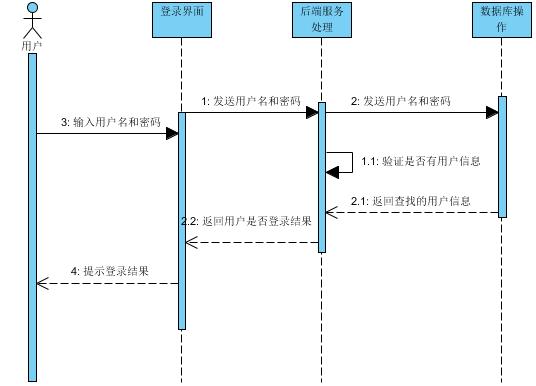


图4-3 系统人员管理顺序图

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：

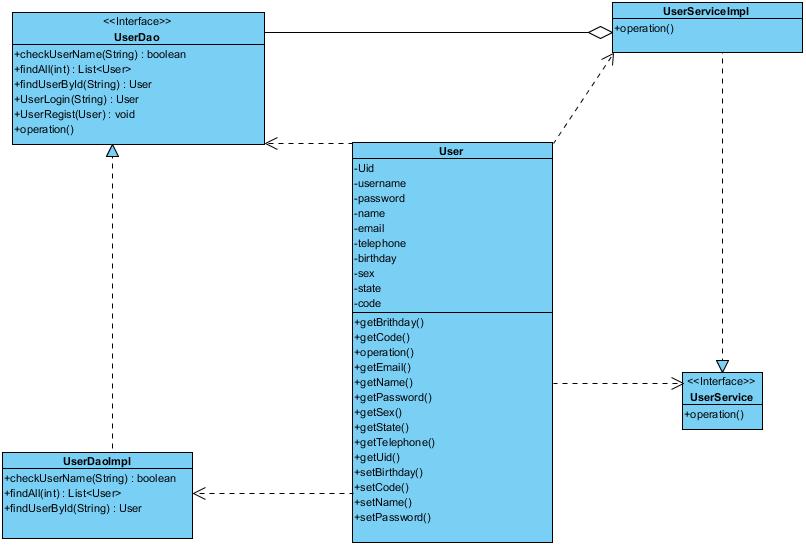


图4-4 系统用户管理类图

**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：

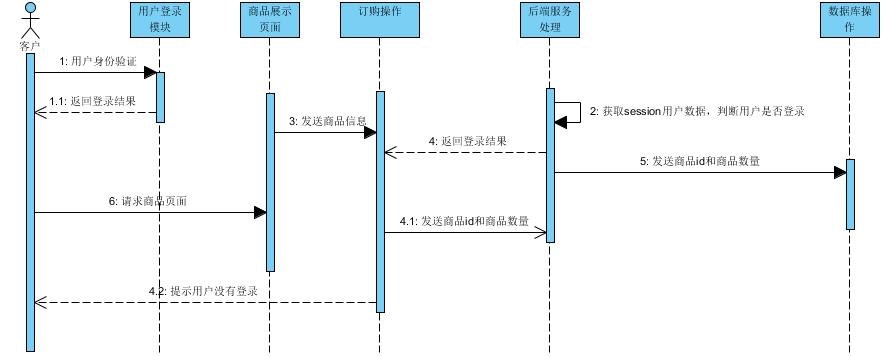


图 4-5 订单管理时序图

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：

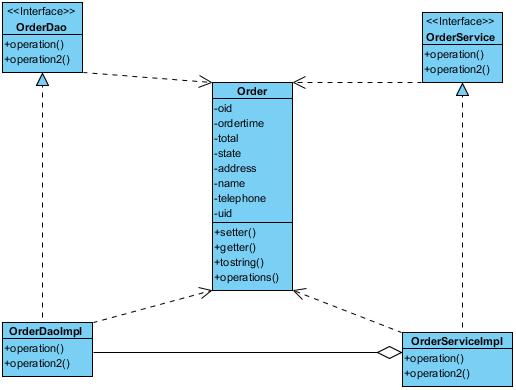


图 4-6 订单管理模块类图设计

**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：

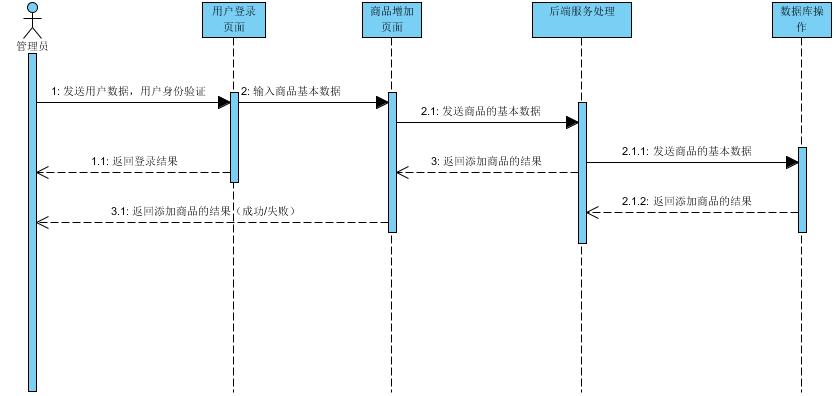


图4-7 商品管理时序图

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：

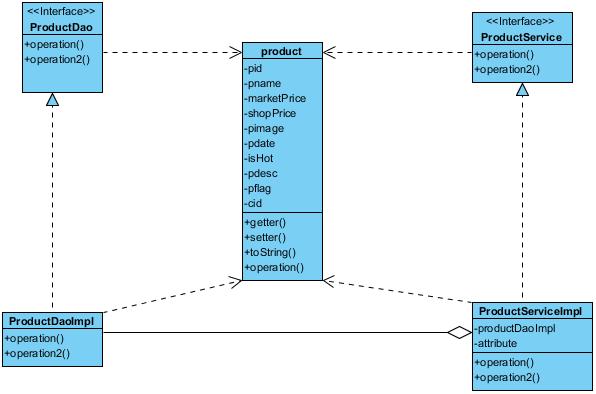


图 4-8 商品管理主类图设计

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用户模块、会员卡模块、充值模块、设备模块、退费模块、消费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：

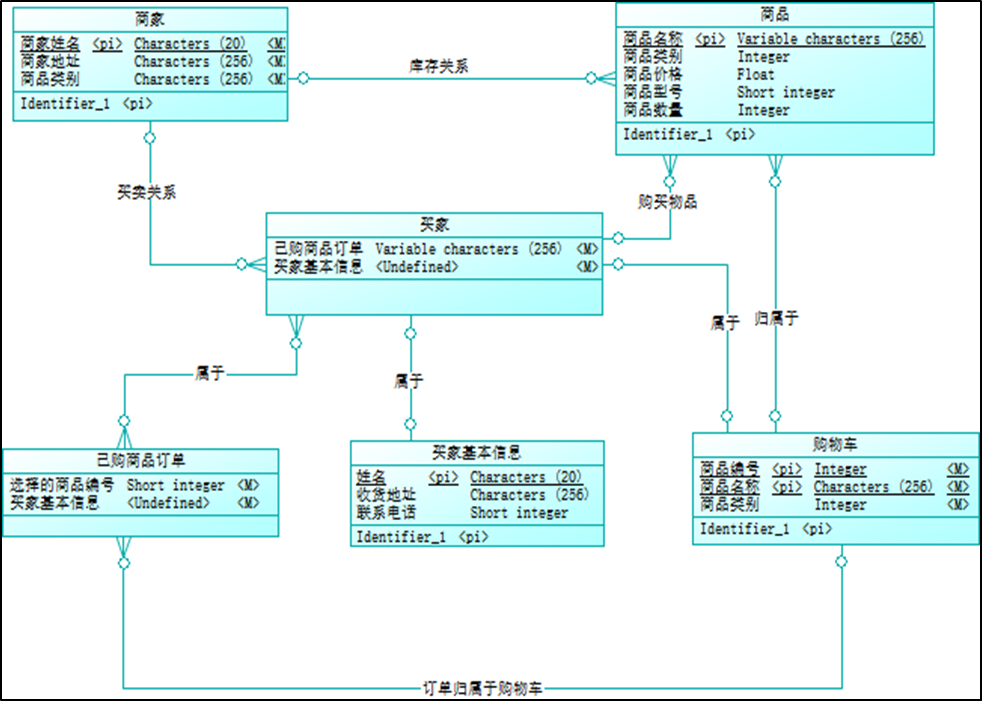


图4-9 系统数据库ER图

**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user表、card表、equip表、consumer表、refund表、top\_up表、goods表等。主要实体化用户、会员卡、设备、消费、退费、充值的设计。

表汇总

|  |  |  |  |
| --- | --- | --- | --- |
| **序号** | 表名 | 中文名称 | 功能说明 |
| 1 | user | 用户表 | 用户表包含三种角色 |
| 2 | card | 会员表 | 会员表包含会员卡和临时卡 |
| 3 | equip | 设备表 | 网吧电脑设备的数据 |
| 4 | consumer | 消费表 | 用户消费的记录 |
| 5 | refund | 退费表 | 用户的退费记录 |
| 6 | top\_up | 充值表 | 用户充值后会充值记录会存在放在数据库 |
| 7 | goods | 商品表 | 网吧的提供的商品 |

表4-1 类别信息表（user）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **字段名称** | **字段描述** | **字段类型** | **长度** | **允许空** | **缺省值** |
| 1 | user\_id | 用户ID | int | 11 |  |  |
| 2 | user\_type | 用户类型 | varchar | 255 |  |  |
| 3 | nick\_name | 昵称 | varchar | 255 |  |  |
| 4 | id\_card | 身份证 | decimal | 11 |  |  |
| 5 | pwd | 密码 | int | 11 |  |  |
| 6 | phone | 手机号 | varchar | 255 |  |  |
| 7 | qq | qq | varchar | 255 |  |  |
| 8 | wxchat | 微信 | varchar | 255 |  |  |
| 9 | email | 邮箱 | varchar | 255 |  |  |
| 10 | sex | 性别 | int | 11 |  |  |
| 11 | create\_user | 创建人 | int | 11 |  |  |
| 12 | type | 用户类型 | int | 11 |  |  |
| 13 | create\_time | 创建时间 | datetime | 0 |  |  |
| 14 | update\_user | 更新人 | int | 11 |  |  |
| 15 | update\_time | 更新时间 | datetime | 0 |  |  |
| 16 | delete\_flag | 是否逻辑删除 | int | 11 |  |  |
| 17 | state | 状态 | int | 11 |  |  |

表4-1 类别信息表（goods）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **字段名称** | **字段描述** | **字段类型** | **长度** | **允许空** | **缺省值** |
| 1 | ID | 商品ID | int | 11 |  |  |
| 2 | item\_name | 商品名 | varchar | 255 |  |  |
| 3 | item\_description | 商品描述 | varchar | 255 |  |  |
| 4 | price | 商品价格 | decimal | 255 |  |  |
| 5 | Status | 商品状态 | int | 11 |  |  |
| 6 | pic\_url | 商品URL | varchar | 255 |  |  |

表4-1 类别信息表（card）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **字段名称** | **字段描述** | **字段类型** | **长度** | **允许空** | **缺省值** |
| 1 | card\_id | 卡id | int | 11 |  |  |
| 2 | card\_type | 卡类型 | int | 11 |  |  |
| 3 | user\_id | 用户id | Int | 11 |  |  |
| 4 | level | 会员等级 | int | 11 |  |  |
| 5 | phone | 手机号码 | varchar | 255 |  |  |
| 6 | sex | 性别 | int | 11 |  |  |
| 7 | total\_money | 总充值金额 | double | 12 |  |  |
| 8 | spend\_money | 已消费金额 | double | 12 |  |  |
| 9 | create\_use | 创建人 | int | 11 |  |  |
| 10 | create\_time | 创建时间 | datetime | 255 |  |  |
| 11 | update\_user | 更新人 | int | 11 |  |  |
| 12 | update\_time | 更新时间 | datetime | 255 |  |  |
| 13 | delete\_flag | 是否逻辑删除 | int | 11 |  |  |
| 14 | state | 是否开机 | Int | 11 |  |  |
| 15 | configure | 配置 | Int | 11 |  |  |
| 16 | postion | 位置 | int | 11 |  |  |

表4-1 类别信息表（consumer）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **字段名称** | **字段描述** | **字段类型** | **长度** | **允许空** | **缺省值** |
| 1 | consume\_id | 消费id | int | 11 |  |  |
| 2 | user\_id | 用户id | int | 11 |  |  |
| 3 | card\_id | 卡id | Int | 11 |  |  |
| 4 | start\_time | 上机时间 | datetime | 255 |  |  |
| 5 | end\_time | 下机时间 | datetime | 255 |  |  |
| 6 | consume\_money | 消费金额 | double | 11 |  |  |
| 7 | create\_user | 创建人 | int | 12 |  |  |
| 8 | create\_time | 创建时间 | datetime | 255 |  |  |
| 9 | update\_user | 更新人 | int | 11 |  |  |
| 10 | update\_time | 更新时间 | datetime | 255 |  |  |
| 11 | delete\_flag | 是否逻辑删除 | int | 11 |  |  |

表4-1 类别信息表（equip）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **字段名称** | **字段描述** | **字段类型** | **长度** | **允许空** | **缺省值** |
| 1 | equip\_id | 设备id | int | 11 |  |  |
| 2 | name | 设备版本 | varchar | 255 |  |  |
| 3 | type | 设备类型 | varchar | 255 |  |  |
| 4 | number | 设备数量 | Int | 11 |  |  |
| 5 | vip\_price | Vip价格 | double | 11 |  |  |
| 6 | no\_vip\_price | 非vip价格 | double | 11 |  |  |
| 7 | reduce | 剩余机型数量 | int | 12 |  |  |

表4-1 类别信息表（refund）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **字段名称** | **字段描述** | **字段类型** | **长度** | **允许空** | **缺省值** |
| 1 | refund\_id | 退费id | int | 11 |  |  |
| 2 | user\_id | 用户id | int | 11 |  |  |
| 3 | card\_id | 卡id | int | 11 |  |  |
| 4 | create\_user | 创建人 | Int | 11 |  |  |
| 5 | create\_time | 创建时间 | datetime | 255 |  |  |
| 6 | update\_user | 更新人 | int | 11 |  |  |
| 7 | update\_time | 更新时间 | datetime | 255 |  |  |
| 8 | delete\_flag | 是否删除 | Int | 11 |  |  |

表4-1 类别信息表（top\_up）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **字段名称** | **字段描述** | **字段类型** | **长度** | **允许空** | **缺省值** |
| 1 | top\_up\_id | 退费id | int | 11 |  |  |
| 2 | card\_id | 用户id | int | 11 |  |  |
| 3 | User\_id | 用户id | Int | 11 |  |  |
| 4 | Money | 卡id | int | 11 |  |  |
| 5 | way | 创建人 | Int | 11 |  |  |
| 6 | create\_user | 创建时间 | datetime | 255 |  |  |
| 7 | create\_time | 更新人 | int | 11 |  |  |
| 8 | update\_user | 更新时间 | datetime | 255 |  |  |
| 9 | update\_time | 是否删除 | Int | 11 |  |  |
| 10 | delete\_flag |  |  |  |  |  |
|  |  |  |  |  |  |  |

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 |  |  |

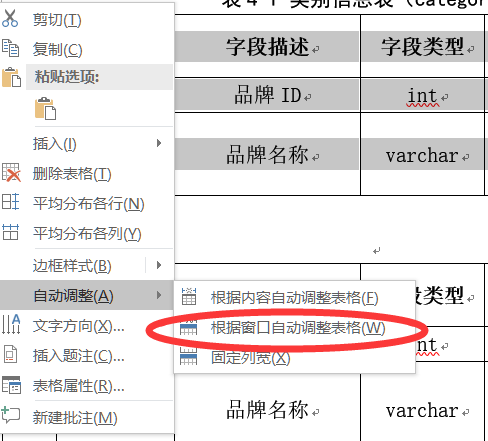
表4-1 类别信息表（category）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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 登录注册模块的设计与实现

**5.1.1 登录注册模块的功能描述**

登录注册模块的业务逻辑设计包含管理员登录、网管的注册、登录以及用户的注册、登录、注销、个人信息管理。用户注册需要提供自己的身份证、手机号，输入信息后进行身份证、手机号、密码的校验，校验完成后才能注册成功。之后，用户可以通过输入身份证和密码的方式登录。系统不提供手机号的方式登录，主要是网吧里为了方便用户登录，统一使用身份证登录的方式，刚注册的用户，密码是统一的123456。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。

**5.1.2 登录注册模块操作流程**

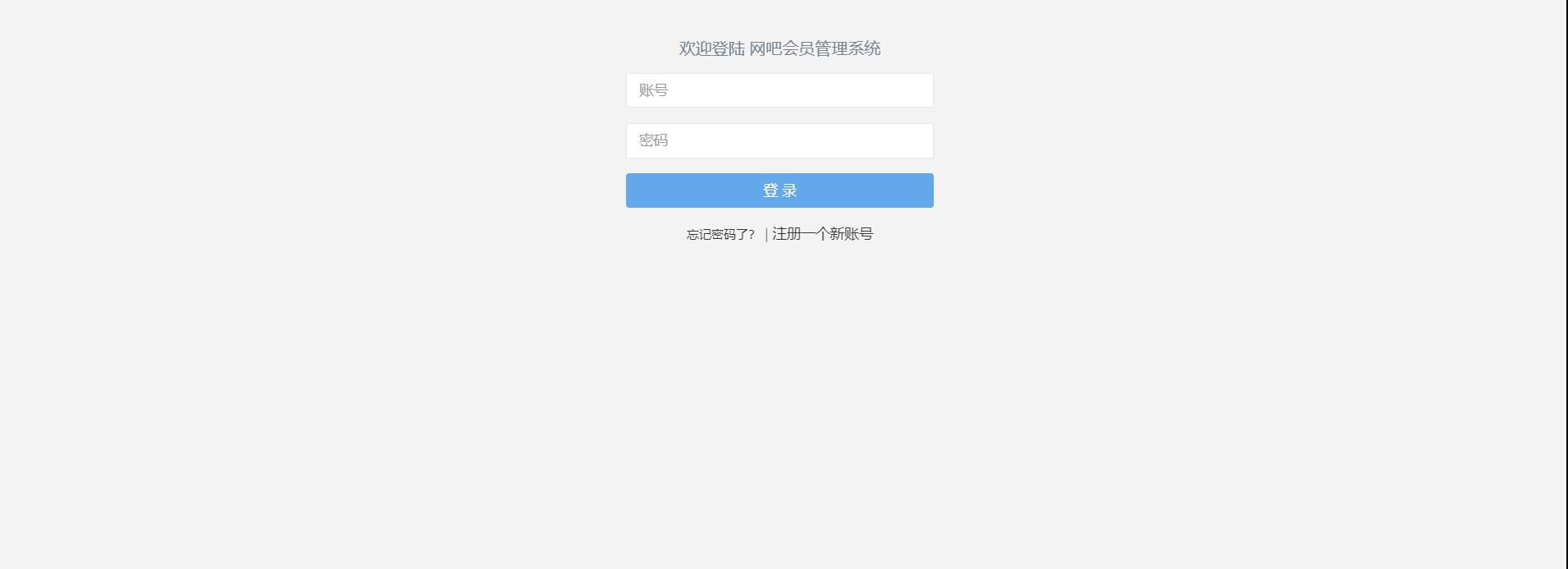
租房用户子模块的界面设计实现用户登录、注册、个人信息的查看与修改等界面，用户登录、注册时，界面需要给与用户必要的提示，同时在输入存在错误时给与反馈。在个人信息查看页面，用户可以查看自己的用户名、手机号、头像等信息，并且在修改之后实时更新显示数据。



**图5.1 登录注册的流程**

**5.1.3 登录注册功能界面设计**

不同的用户，有不同的登录注册界面，管理员和网管都具备管理功能，使用同一个登录页面，而用户的登录界面在电脑桌面上，值需要输入身份证和密码即可解锁电脑，享受其电脑服务，当身份证或者密码错误时，会提示出错误提示。



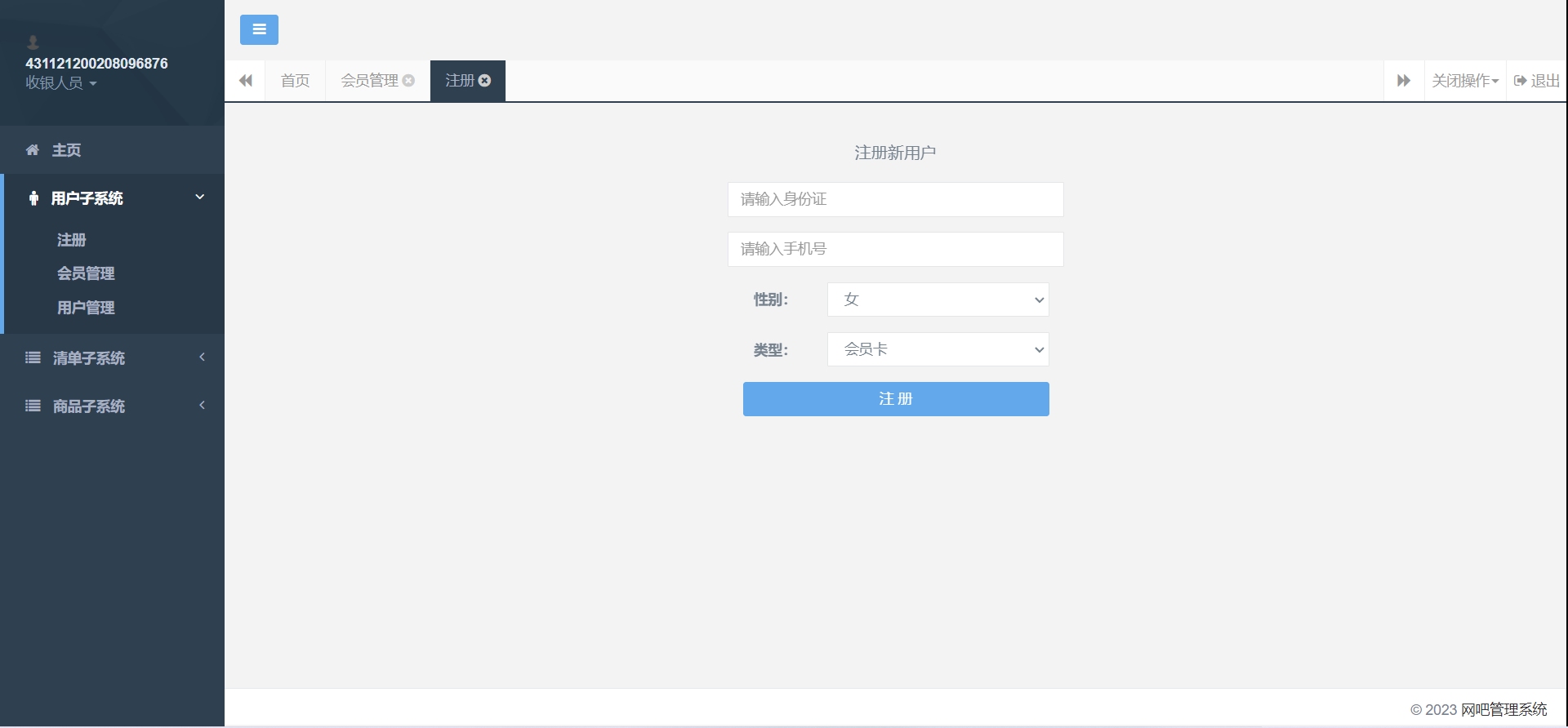


图5-2 用户注册界面设计图

**5.1.4 登录注册功能类图结构**

图 3-5 展示了登录注册子模块的类图结构，其中主要由 UserLogin类、

UserController 类 和 UserServiceImpl类 负 责 实 现 登录注册的 管 理，SpringSecurityConfig类、UserDetailsServiceImpl类、MyAuthenticationFailureHandler类、LogoutSuccessHandlerImpl类是SpringSecurity的认证配置类。UserController类中有修改手机号、用户名和头像等方法，它依赖于 UserServiceImpl 等接口。



图 3-5 登录注册子模块类图

**5.1.5 XXX功能核心代码**

@GetMapping("/userLogin")

public ResultBean userLogin(HttpServletRequest request, HttpServletResponse response) {

//获取req的参数

String idCard = request.getParameter("account");

String pwd = request.getParameter("pwd");

ResultBean resultBean = null;

// 用户验证

Authentication authentication = null;

try {

UsernamePasswordAuthenticationToken authenticationToken = new UsernamePasswordAuthenticationToken(idCard, pwd);

// 该方法会去调用UserDetailsServiceImpl.loadUserByUsername

authentication = authenticationManager.authenticate(

authenticationToken);

//返回登录的用户信息

UserInfo userInfo = (UserInfo) authentication.getPrincipal();

if(userInfo != null){

throw new Exception("用户不存在");

}

//查询用户的数据并保存在Session中

User user = userService.getOne(new QueryWrapper<User>().eq("id\_card", idCard));

HttpSession session = request.getSession();

session.setAttribute("user", JSON.toJSONString(user));

resultBean = new ResultBean();

resultBean.setData("2");

} catch (Exception e) {

e.printStackTrace();

resultBean = new ResultBean(e);

if (e.getMessage().equals("用户不存在"))

resultBean.setData("0");

else if (e.getMessage().equals("密码不正确"))

resultBean.setData("1");

else resultBean.setData("3");

} finally {

return resultBean;

}

}}

5.2 用户管理模块的设计与实现

**5.2.1 用户管理模块的功能描述**

用户管理模块的业务逻辑设计是包含用户的用户的搜索、用户的详情展示、用户的编辑、用户的删除、用户的启用/禁用等功能。该模块是在网管登录之后才能具有的功能，非该角色无法进行操作。

**5.2.2 用户管理模块操作流程**

网管在登录之后，进入管理页面的主页，管理页面具有用户管理功能，包含用户的用户数据的增删改查等功能。



**图5.1 租房用户的注册流程**

**5.2.3 用户管理功能界面设计**

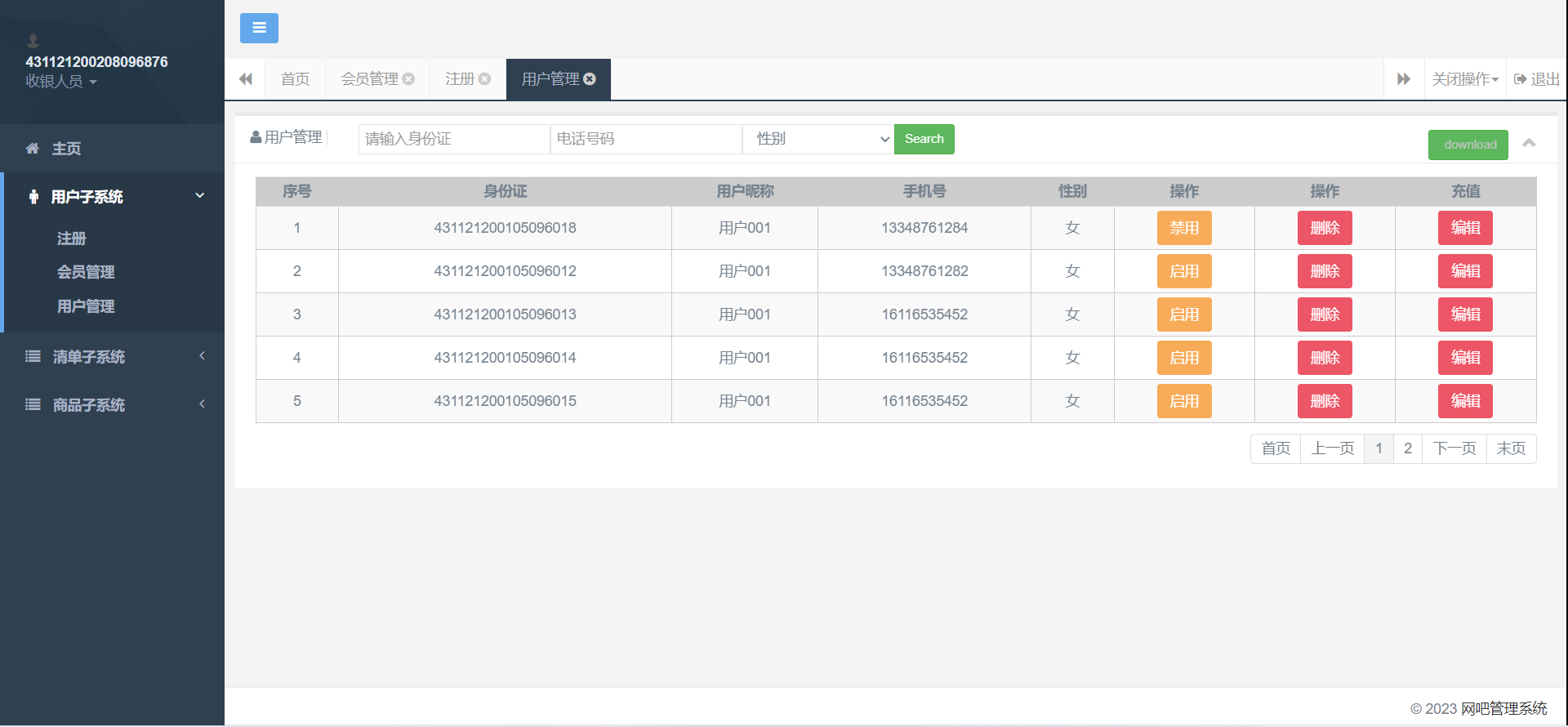
网管对用户的管理，必须包含增删功能，在此基础上，为了让用户有更好的体验，对数据进行了搜索和分页操作。

图5-2 用户管理界面设计图

**5.2.4 登录注册功能类图结构**

图 3-5 展示了用户管理子模块的类图结构，其中主要由 UserController类、

CardController类 和 UserController类 负 责 实 现对用户的 管 理。UserController类中有获取用户列表、用户的增删改查等方法，它依赖于 UserServiceImpl 等接口。



图 3-5 租房用户子模块类图

**5.2.4 用户管理功能核心代码**

//用户列表的获取

@GetMapping("/userList")

public IPage<User> userList(HttpServletRequest request, HttpServletResponse response) {

//获取req的请求参数

String idCard = request.getParameter("idCard2");

String phone = request.getParameter("phone2");

String sex = request.getParameter("sex2");

String userId = request.getParameter("userId");

//获取当前登录用户的信息

HttpSession session = request.getSession();

User user = JSON.parseObject((String)session.getAttribute("user"),User.class);

String param = request.getParameter("current");

//mybatisplus提供的分页查和讯

Integer page = 1;

if(param != null) page = Integer.parseInt(param);

Page pages = new Page<>(page, 5);

IPage<User> iPage = null;

try {

iPage = userService.selectJoinListPage(pages, User.class,

new MPJLambdaWrapper<User>()

.selectAll(User.class)

.eq(User::getDeleteFlag, 0)

.like(StringUtils.isNotEmpty(idCard),User::getIdCard,idCard)

.like(StringUtils.isNotEmpty(phone),User::getPhone,phone)

.eq(StringUtils.isNotEmpty(sex),User::getSex,sex)

.eq(StringUtils.isNotEmpty(userId),User::getUserId,userId)

.eq(user != null,User::getUserType,user != null ?user.getUserType()+1:0));

} catch (Exception e) {

e.printStackTrace();

}finally {

return iPage;

}

}

5.3 会员卡管理模块的设计与实现

**5.3.1 会员卡管理模块的功能描述**

会原卡管理模块的业务逻辑设计是包含用户上机、用户下机、会员卡充值、会员卡退费、用户卡详情、会员卡的搜索、会员的退费等功能。该模块是在网管登录之后才能具有的功能，非该角色无法进行操作。

**5.3.2 会员卡管理模块操作流程**

网管在登录之后，进入管理页面的主页，管理页面具有会员卡管理功能，包含用户的上下机和会员卡的一系列管理功能。



**图5.1 租房用户的注册流程**

**5.3.3 会员卡管理功能界面设计**

网管对会员卡的管理，必须包含对会员卡的增删改查功能，在此基础上，为了更好的管理会员，增加了用户的上机、用户下机和用户充值等功能。

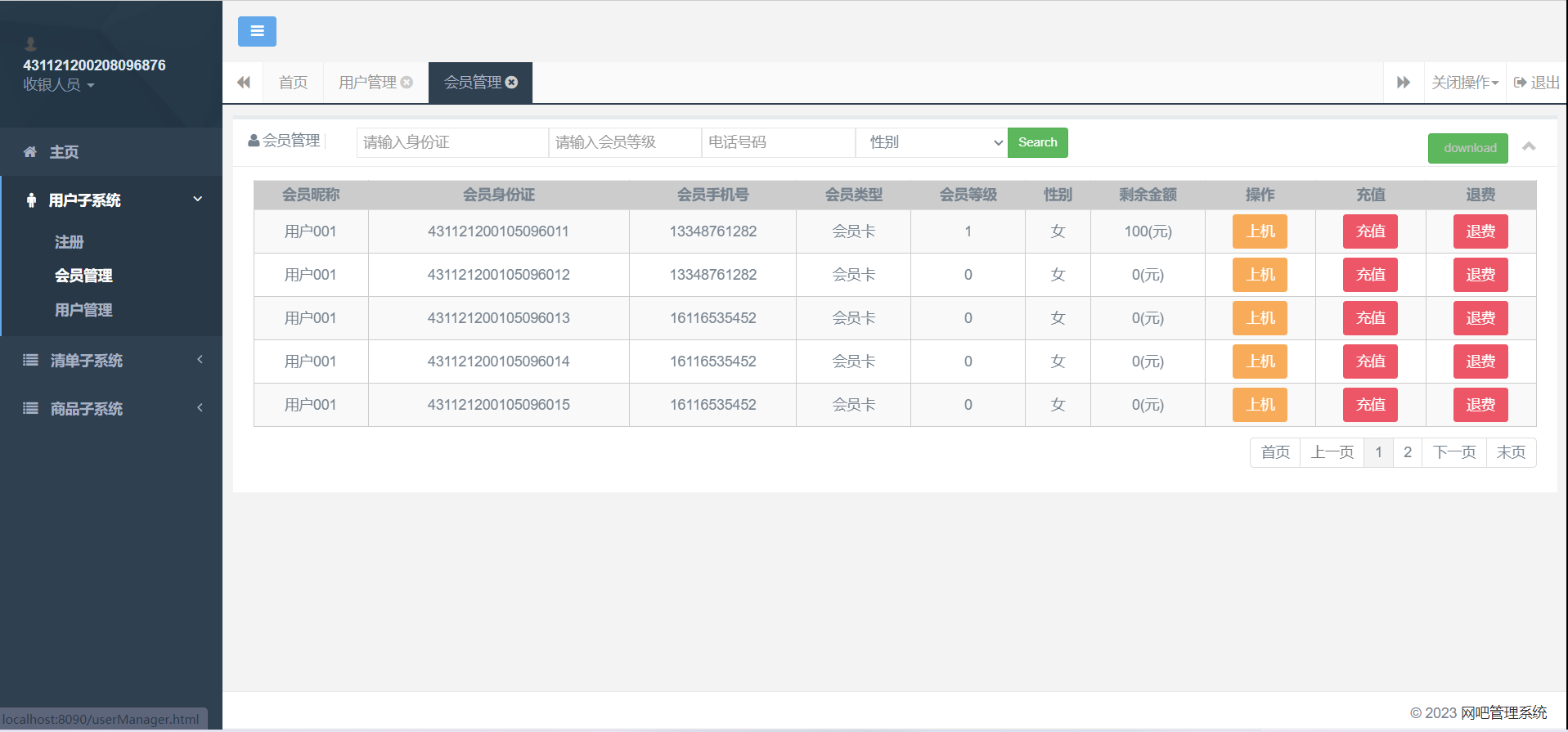


图5-2 前台登录界面设计图

**5.3.4 登录注册功能类图结构**

图 3-5 展示了用户管理子模块的类图结构，其中主要由 CardController类、

CardController类 和 RefundController类 负 责 实 现对会员卡的 管 理。CardController类中有获取会员列表、会员的增删改查、会员的退费等方法，它依赖于 CardServiceImpl、UserServiceImpl等接口。



图 3-5 租房用户子模块类图

**5.3.5 用户管理功能核心代码**

//清单列表

@GetMapping("/statistics")

public List<TopUpDto> statistics(HttpServletRequest request, HttpServletResponse response){

//获取req的参数

String type = request.getParameter("type");

String phone = request.getParameter("phone2");

String idCard = request.getParameter("idCard2");

String way = request.getParameter("way2");

//对比参数是日收入清单还是月收入清单

String day = DateUtil.getNowday();

String nowMonth = DateUtil.getNowMonth();

List<TopUp> list = null;

ArrayList<TopUpDto> topUpDtos = new ArrayList<TopUpDto>();

try {

if (type.equals("day")){

//日收入

list = topUpService.list(new QueryWrapper<TopUp>().like("create\_time", day).eq(StringUtils.isNotEmpty(way),"way",way));

}else {

//月收入

list = topUpService.list(new QueryWrapper<TopUp>().like("create\_time", nowMonth).eq(StringUtils.isNotEmpty(way),"way",way));

}

for (TopUp topUp : list) {

boolean flag = true;

User user = userService.getById(topUp.getUserId());

//变量的深度复制，用于页面展示的实体类

TopUpDto topUpDto = new TopUpDto();

PropertyUtils.copyProperties(topUpDto, topUp);

topUpDto.setCreateUserName(user.getNickName());

topUpDto.setIdCard(user.getIdCard());

topUpDto.setPhone(user.getPhone());

if(StringUtils.isNotEmpty(idCard)&& !topUpDto.getIdCard().contains(idCard))flag = false;

if(StringUtils.isNotEmpty(phone) && !topUpDto.getPhone().contains(phone))flag = false;

if(flag)

topUpDtos.add(topUpDto);

}

} catch (Exception e) {

e.printStackTrace();

}finally {

return topUpDtos;

}

}

5.4 清单管理模块的设计与实现

**5.4.1 清单管理模块的功能描述**

清单管理模块的业务逻辑设计是包含对日收入清单，月收入清单，清单列表等功能。该模块是在网管登录之后才能具有的功能，非该角色无法进行操作。

**5.4.2 用户管理模块操作流程**

网管在登录之后，进入管理页面的主页，管理页面具有清单管理的功能，包含网管的当天收入和月收入记录等功能。



**图5.1 租房用户的注册流程**

**5.4.3 清单管理功能界面设计**

网管对清单的管理，必须包含对清单列表功能，在此基础上，为了更好的管理收入清单，日收入和月收入清单列表的功能。



图5-2 前台登录界面设计图

**5.4.4 清单管理功能类图结构**

图 3-5 展示了用户管理子模块的类图结构，其中主要由 StatisticsController类、UserController、CardController类 、ConumerContoller和 TopupController类 负 责 实 现对清单的 管 理。StatisticsController类中有获取日收入清单和月收入清单等方法，它依赖于 CardServiceImpl、UserServiceImpl、ConsumerServiceImpl等接口。



图 3-5 租房用户子模块类图

**5.4.5 用户管理功能核心代码**

/\*

\* 提交退费申请

\* \*/

@PostMapping("/card/refund")

public ResultBean refund(HttpServletRequest request, HttpServletResponse response) {

Integer userId = Integer.parseInt(request.getParameter("userId"));

Integer cardId = Integer.parseInt(request.getParameter("cardId"));

ResultBean resultBean = null;

try {

//改变用户状态

User user = userService.getById(userId);

user.setState(1);

userService.updateById(user);

//退费

Refund refund = new Refund();

refund.setUserId(userId);

refund.setCardId(cardId);

refundService.save(refund);

//清空用户余额

Card card = cardService.getById(cardId);

card.setTotalMoney(0.0);

card.setSpendMoney(0.0);

card.setLevel(0);

cardService.updateById(card);

resultBean = new ResultBean();

resultBean.setData(true);

} catch (Exception e) {

resultBean = new ResultBean(e);

resultBean.setData("操作失败");

e.printStackTrace();

} finally {

return resultBean;

}

}

**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")

}

5.5 设备管理模块的设计与实现

**5.5.1 设备管理模块的功能描述**

设备管理模块的业务逻辑设计是包含对网吧电脑设备的管理功能。该模块是在管理员登录之后才能具有的功能，非该角色无法进行操作。

**5.5.2 设备管理模块操作流程**

管理员在登录之后，进入管理页面的主页，管理页面具有设备管理的功能，包含展示网吧设备的不同类型，改变网吧不同级别电脑的收费情况以及网吧设备的升级等功能。



**图5.1 租房用户的注册流程**

**5.5.3 设备管理功能界面设计**

管理员对网吧设备的管理，必须包含对设备的展示功能，在此基础上，为符合当前社会的需求，具有更新换代的功能，不容易被淘汰，所以设计了网吧的设备升级功能。

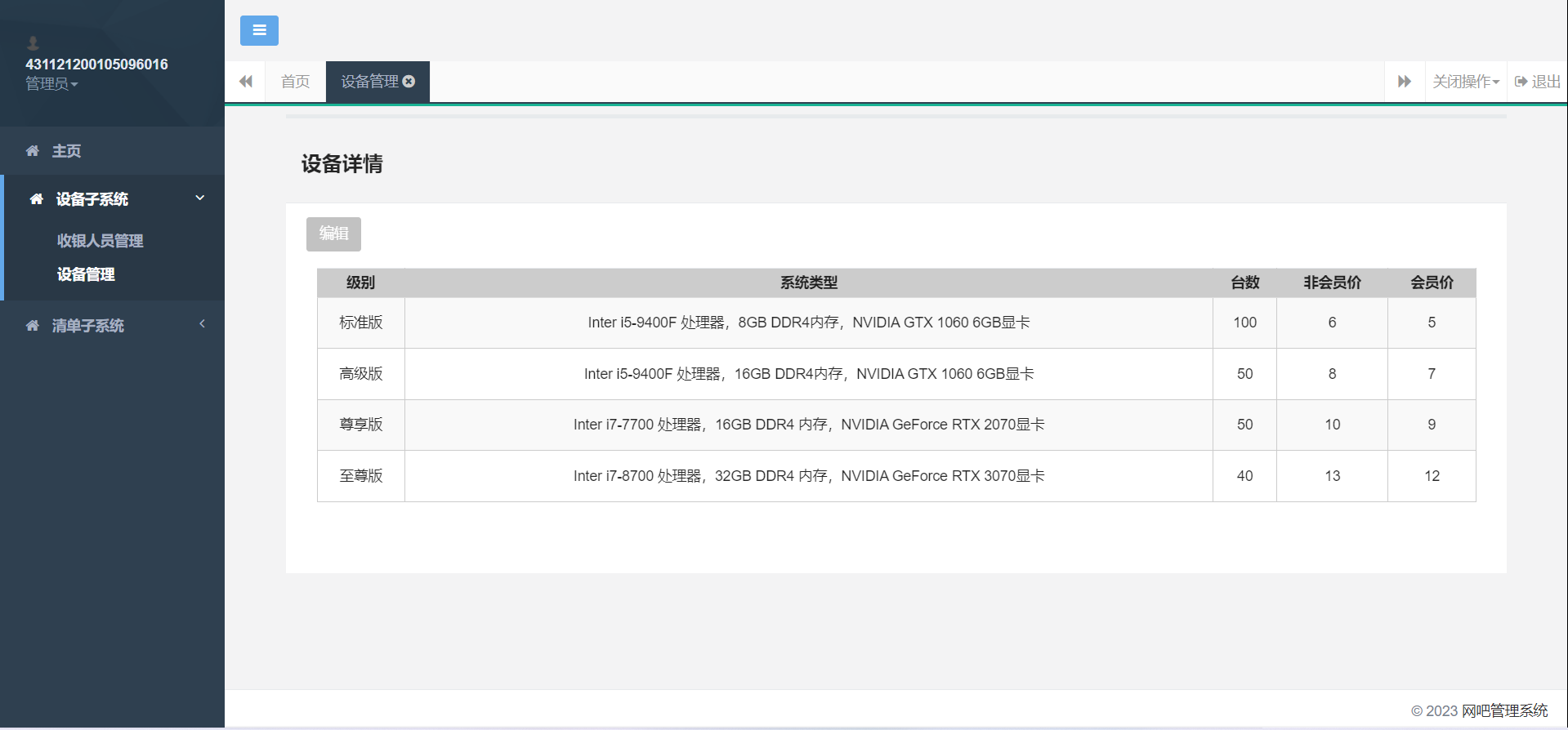


图5-2 前台登录界面设计图

**5.5.4 清单管理功能类图结构**

图 3-5 展示了用户管理子模块的类图结构，其中主要由 StatisticsController类、UserController、CardController类 、ConumerContoller和 TopupController类 负 责 实 现对清单的 管 理。StatisticsController类中有获取日收入清单和月收入清单等方法，它依赖于 CardServiceImpl、UserServiceImpl、ConsumerServiceImpl等接口。



图 3-5 租房用户子模块类图

**5.5.4 用户管理功能核心代码**

/\*

\* 提交退费申请

\* \*/

@PostMapping("/card/refund")

public ResultBean refund(HttpServletRequest request, HttpServletResponse response) {

Integer userId = Integer.parseInt(request.getParameter("userId"));

Integer cardId = Integer.parseInt(request.getParameter("cardId"));

ResultBean resultBean = null;

try {

//改变用户状态

User user = userService.getById(userId);

user.setState(1);

userService.updateById(user);

//退费

Refund refund = new Refund();

refund.setUserId(userId);

refund.setCardId(cardId);

refundService.save(refund);

//清空用户余额

Card card = cardService.getById(cardId);

card.setTotalMoney(0.0);

card.setSpendMoney(0.0);

card.setLevel(0);

cardService.updateById(card);

resultBean = new ResultBean();

resultBean.setData(true);

} catch (Exception e) {

resultBean = new ResultBean(e);

resultBean.setData("操作失败");

e.printStackTrace();

} finally {

return resultBean;

}

}

5.6 商品管理模块的设计与实现

**5.6.1 商品管理模块的功能描述**

商品管理模块的业务逻辑设计是包含展示商品、商品上新、移除商品、更新商品等功能。该模块是在网管登录之后才能具有的功能，非该角色无法进行操作。

**5.6.2 商品管理模块操作流程**

网管在登录之后，进入管理页面的主页，管理页面具有商品管理的功能，可以对商品进行一系列操作。



**图5.1 商品管理的流程**

**5.6.3 商品管理功能界面设计**

网管对商品的管理，必须包含对商品列表功能，在此基础上，为了让用户有更好的体验，用户购买后能查看订单信息，由网管进行配送商品。

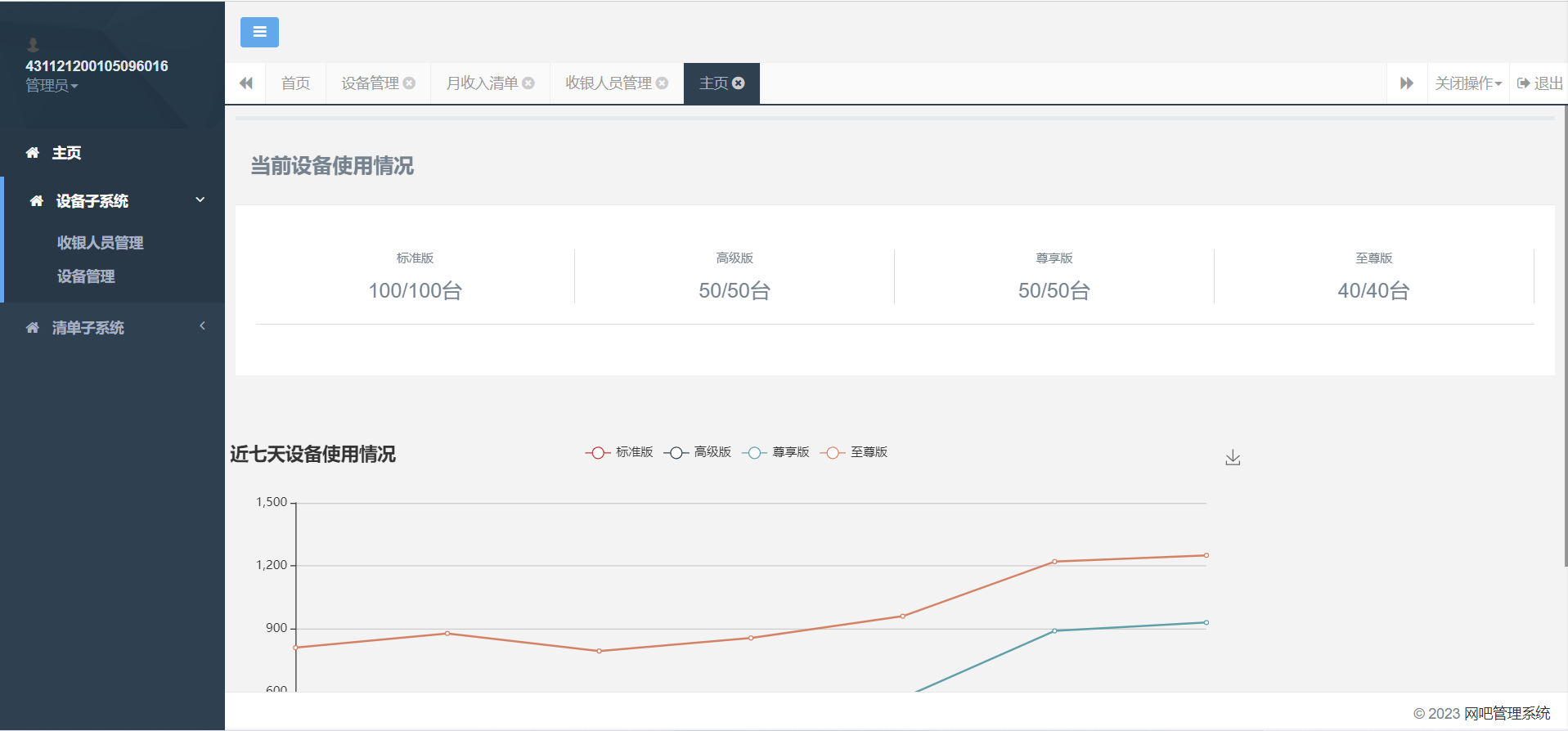


图5-2 商品界面设计图

**5.6.4 清单管理功能类图结构**

图 3-5 展示了用户管理子模块的类图结构，其中主要由 StatisticsController类、UserController、CardController、GoodsController类 、ConumerContoller和 TopupController类 负 责 实 现对清单的 管 理。StatisticsController类中有获取日收入清单和月收入清单等方法，它依赖于 CardServiceImpl、UserServiceImpl、ConsumerServiceImpl等接口。



图 3-5 租房用户子模块类图

**5.6.5 用户管理功能核心代码**

/\*

\* 商品管理

\* \*/

@PostMapping("/card/refund")

public ResultBean refund(HttpServletRequest request, HttpServletResponse response) {

Integer userId = Integer.parseInt(request.getParameter("userId"));

Integer cardId = Integer.parseInt(request.getParameter("cardId"));

ResultBean resultBean = null;

try {

//改变用户状态

User user = userService.getById(userId);

user.setState(1);

userService.updateById(user);

//商品类型

Refund refund = new Refund();

refund.setUserId(userId);

refund.setCardId(cardId);

refundService.save(refund);

//商品的购买

Card card = cardService.getById(cardId);

card.setTotalMoney(0.0);

card.setSpendMoney(0.0);

card.setLevel(0);

cardService.updateById(card);

resultBean = new ResultBean();

resultBean.setData(true);

} catch (Exception e) {

resultBean = new ResultBean(e);

resultBean.setData("操作失败");

e.printStackTrace();

} finally {

return resultBean;

}

}

5.7 电脑管理模块的设计与实现

**5.7.1 电脑管理模块的功能描述**

电脑管理模块的业务逻辑设计实现了模拟电脑的基本功能，用户在浏览器中可以正常操作电脑，包括切换壁纸、浏览器搜索、打开文件、打开图片、充值网费、下机、查看当前设备详情以及呼叫网管是等功能。该模块是在用户登录之后才能具有的功能，非该角色无法进行操作。

**5.7.2 网吧管理模块操作流程**

用户在登录之后，解锁电脑的所有功能，否则提示出对应的错误信息，重新登录。电脑具有切换壁纸、浏览器搜索、打开文件、打开图片、充值网费、下机、呼叫网管以及查看当前设备详情等功能。



**图5.1 租房用户的注册流程**

**5.7.3 电脑管理功能界面设计**

模仿window做了一台电脑，具有电脑的基本功能，包括单击、双击、打开文件、切换壁纸等功能，和真实的电脑没有多大区别，以下就是电脑的界面设计。



图5-2 前台登录界面设计图

**5.7.4 电脑管理功能类图结构**

图 3-5 展示了用户管理子模块的类图结构，其中主要由 StatisticsController类、UserController、CardController类 、ConumerContoller和 TopupController类 负 责 实 现对清单的 管 理。StatisticsController类中有获取日收入清单和月收入清单等方法，它依赖于 CardServiceImpl、UserServiceImpl、ConsumerServiceImpl等接口。



图 3-5 租房用户子模块类图

**5.7.5 用户管理功能核心代码**

var \_ID = 0;

/\*当前所在目录的id，每当视图发生改变的时候，也就是view方法执行的时候，

需要同步该值\*/

setFile(\_ID);

// 渲染初始化数据，显示pid为0的数据

var back = document.querySelector('.back');

back.addEventListener('click',function(){

var info = getInfo(\_ID);// 返回上一级：获取当前目录的父级的子目录

console.log(data.list,info,\_ID);

if(info) {

setFile(info.pid);

}

});

/\*\*

\* 添加新数据

\* @param newData

\*/

function addData(newData){

newData.id = getMaxId() + 1;

data.list.push(newData);

}

/\*

\* 获取数据中最大的id

\* \*/

function getMaxId() {

var maxId = 0;

data.list.forEach(function(item){

if(item.id > maxId) {

maxId = item.id

}

});

return maxId;

}

//音视频播放时禁用右键菜单；

fileDetail.oncontextmenu = function(e){

e.stopPropagation();

e.preventDefault();

}

fileDetail.onmousedown = function(e){

e.stopPropagation();

}

fileDetail.onclick = function(e){

e.stopPropagation();

}

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：

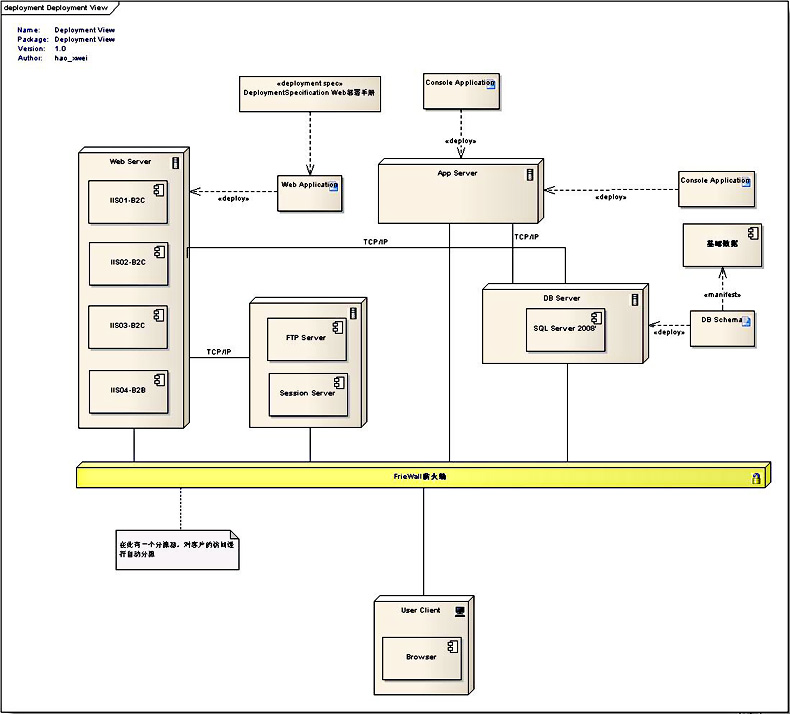


图6-2 系统UML部署图

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：

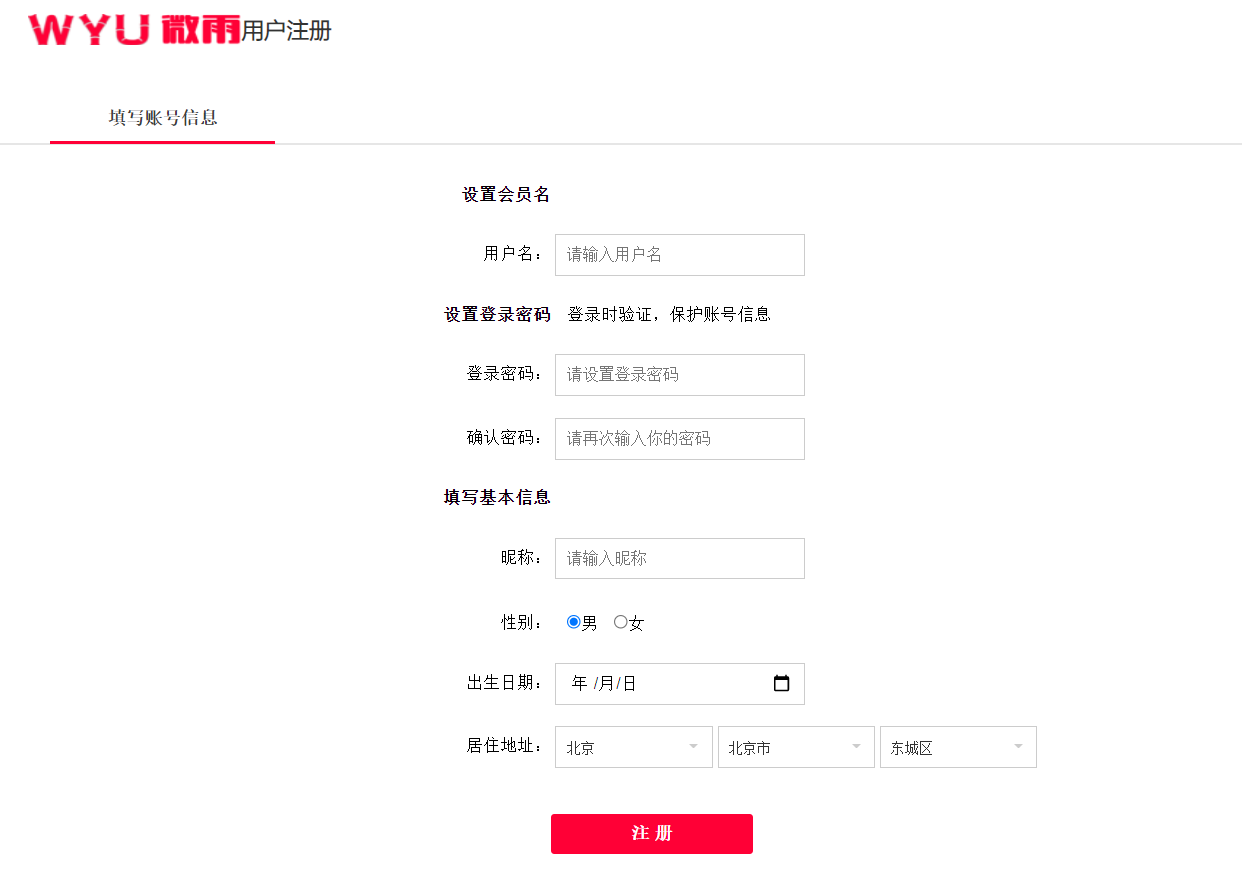


图2-3 前台用户注册

（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。

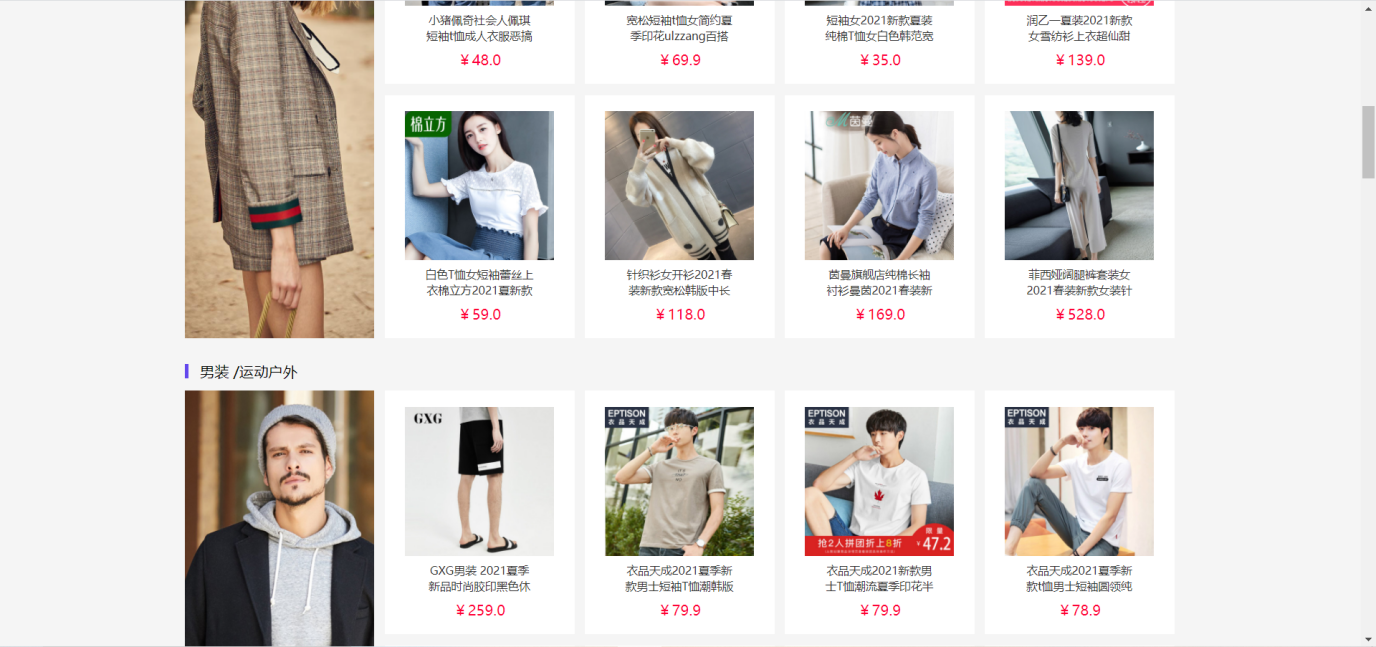


图2-4 商品分类展示

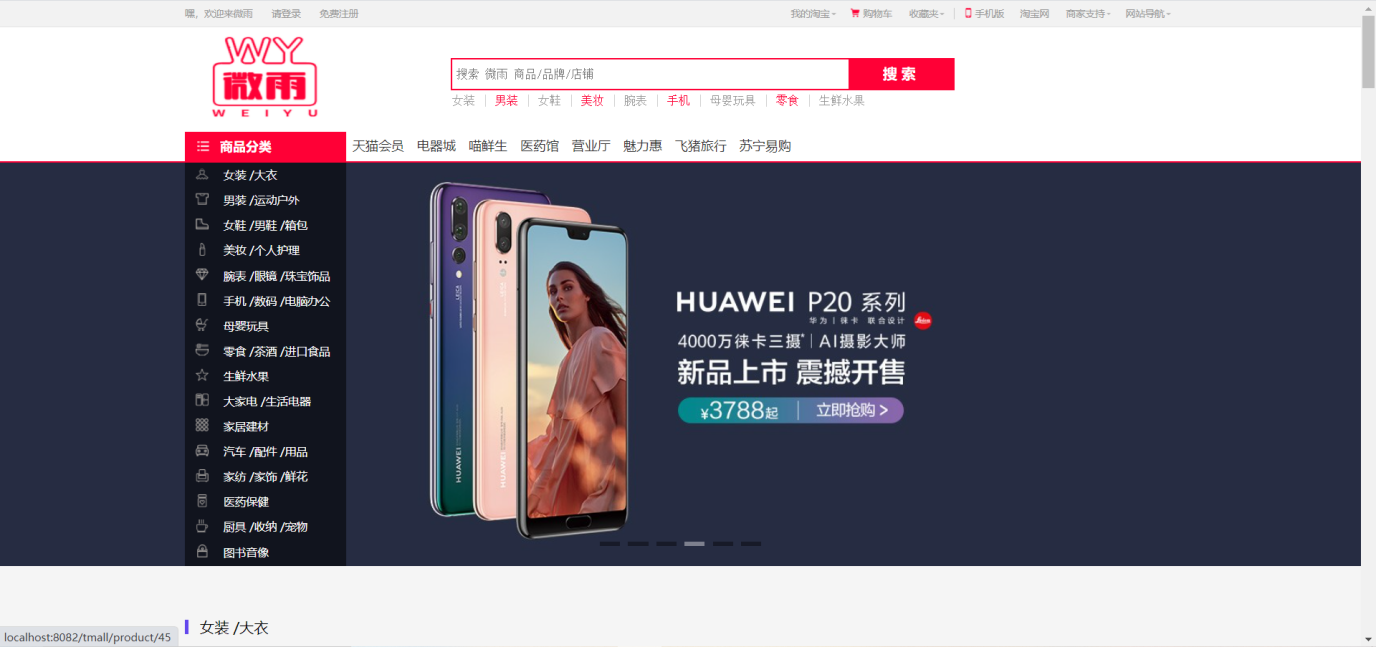


图 2-5 商品分类展示界面

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：

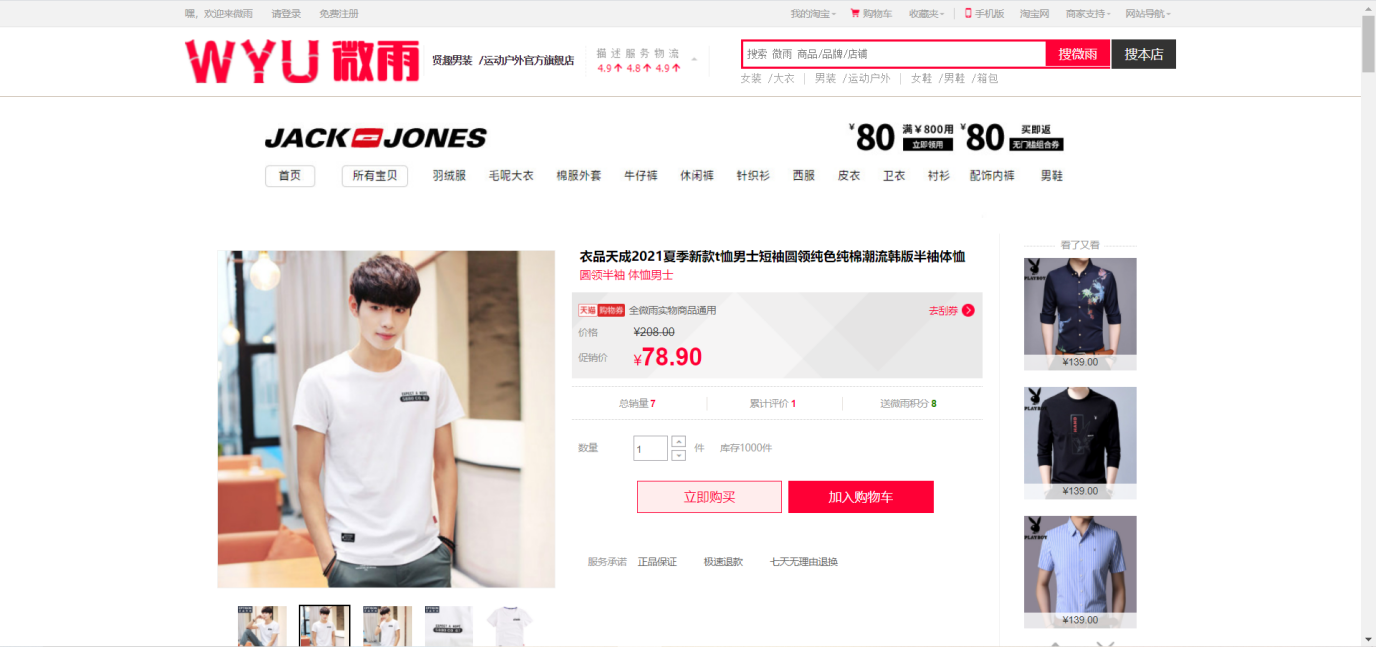


图2-6 商品详情页展示

（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：

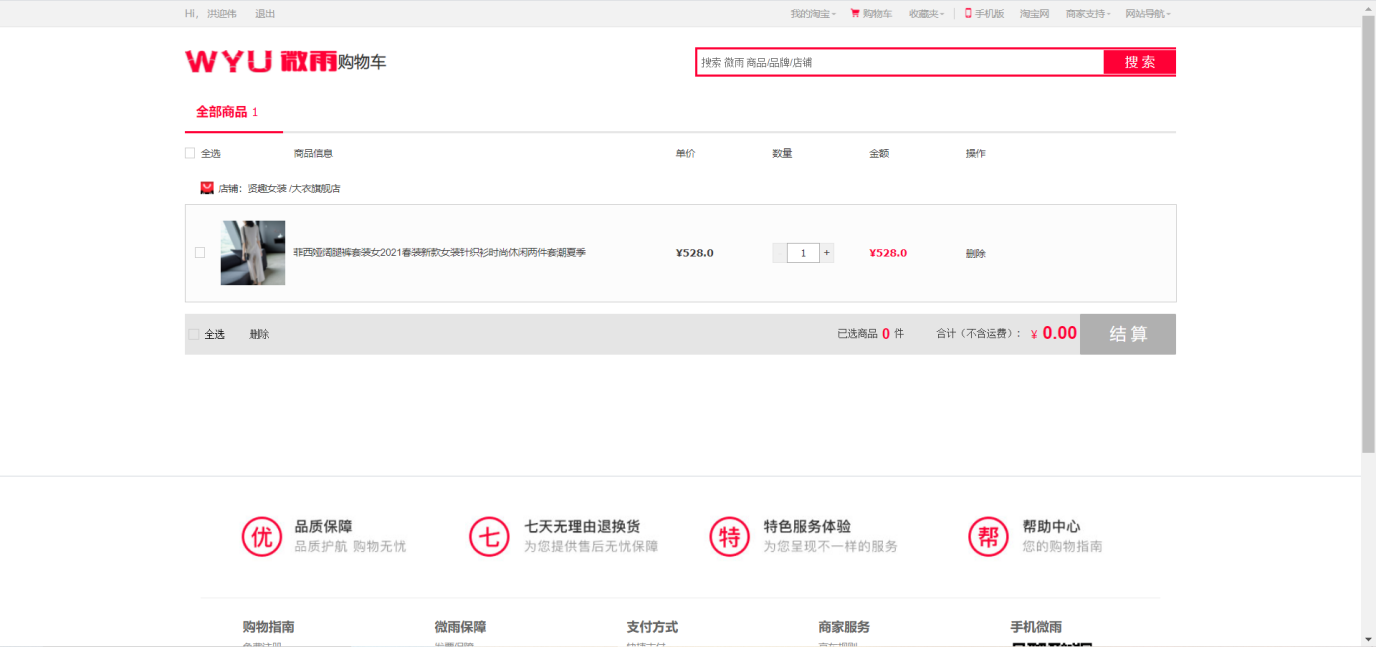


图2-7 购物车界面

（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。

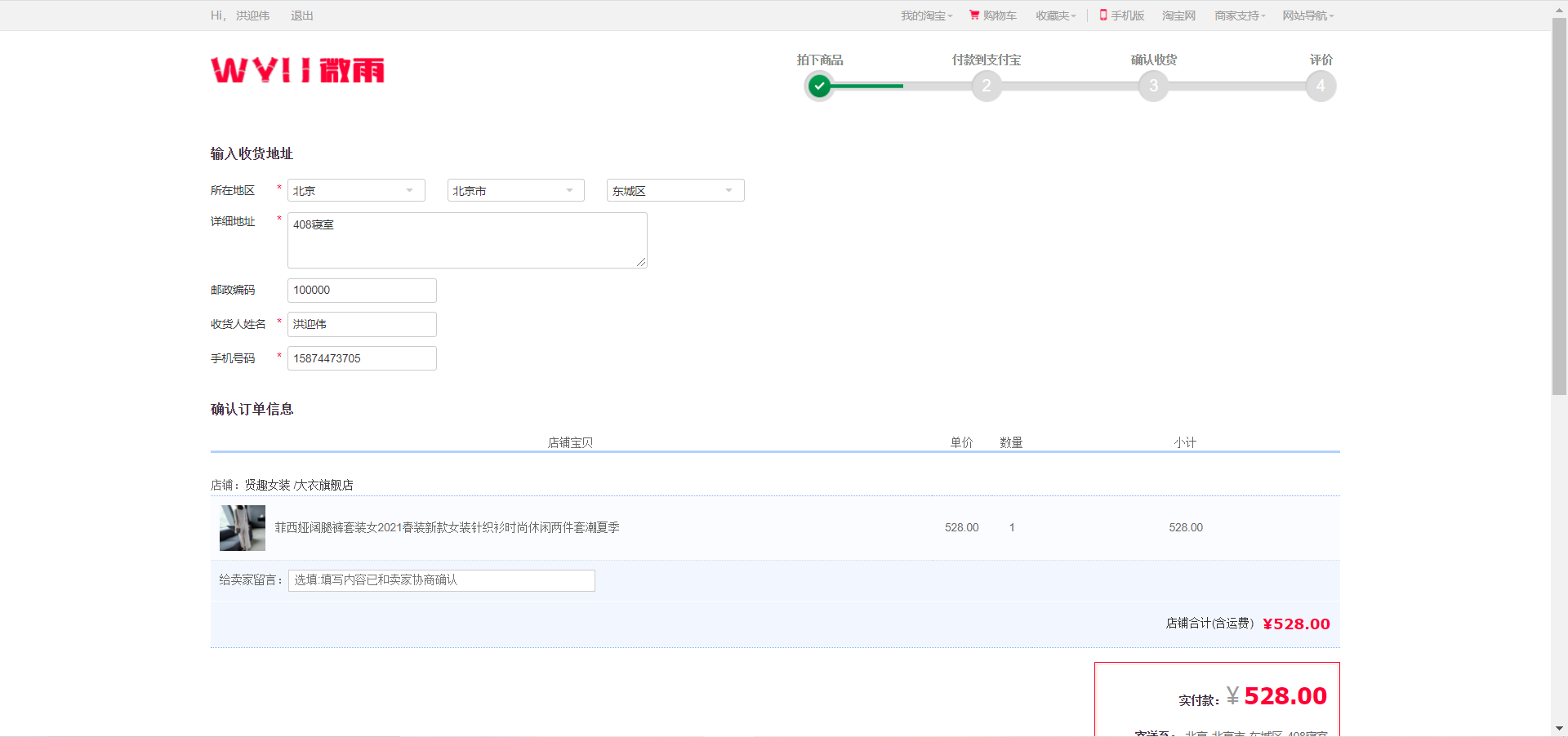


图2-8 生成订单页面

（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。

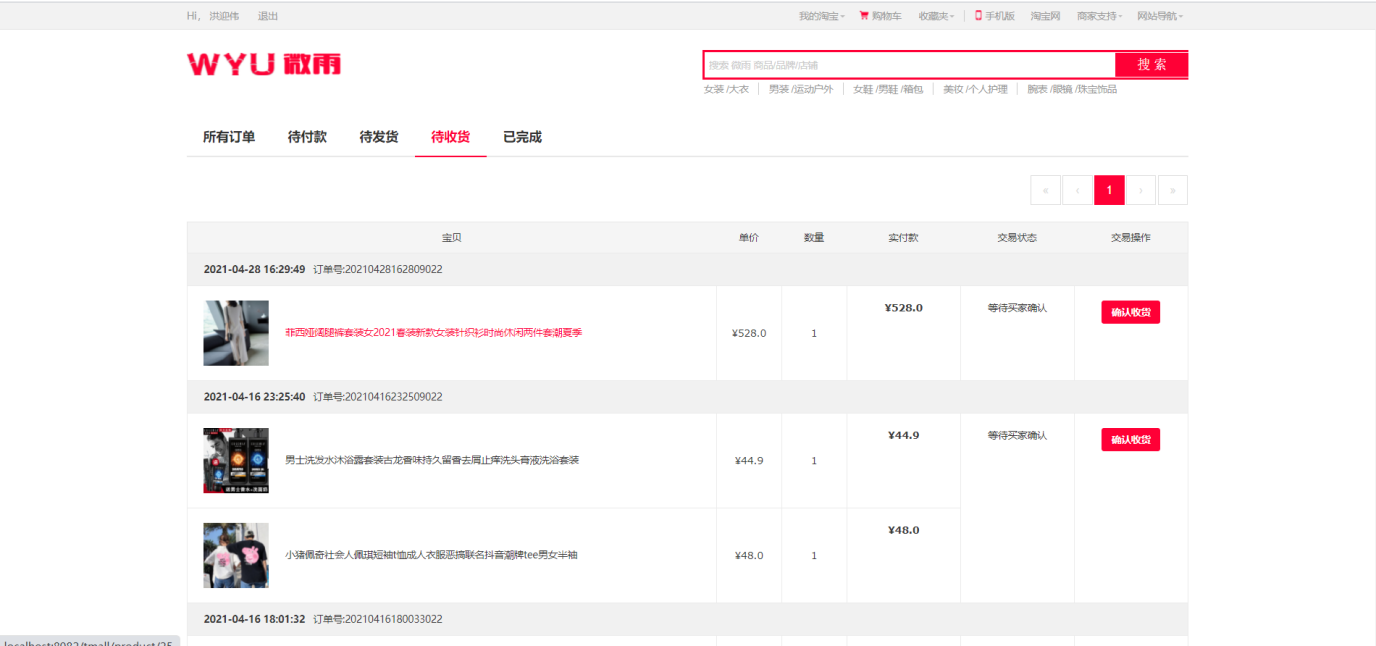


图2-9 客户的订单状态

（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。

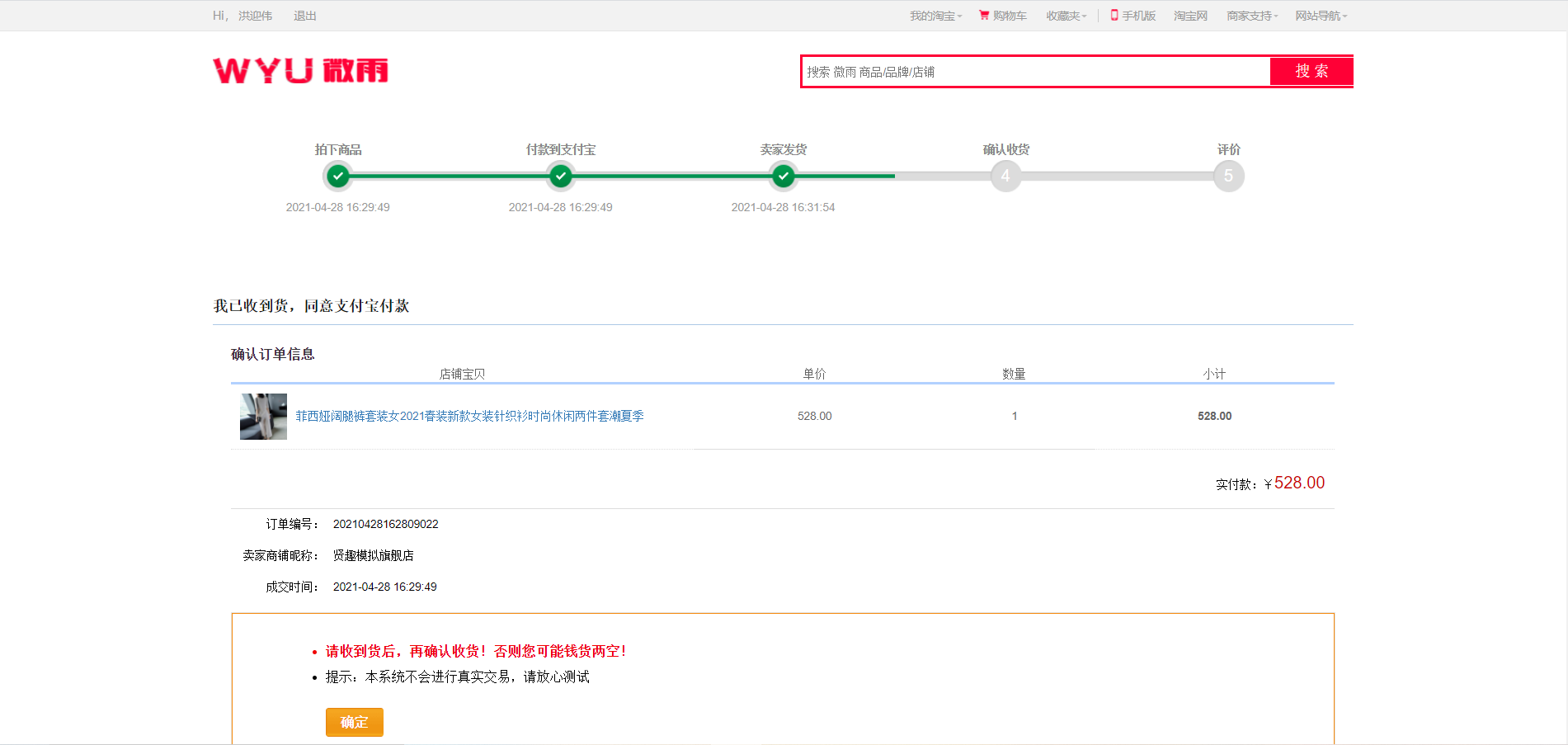


图2-10 商品评价界面

**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：

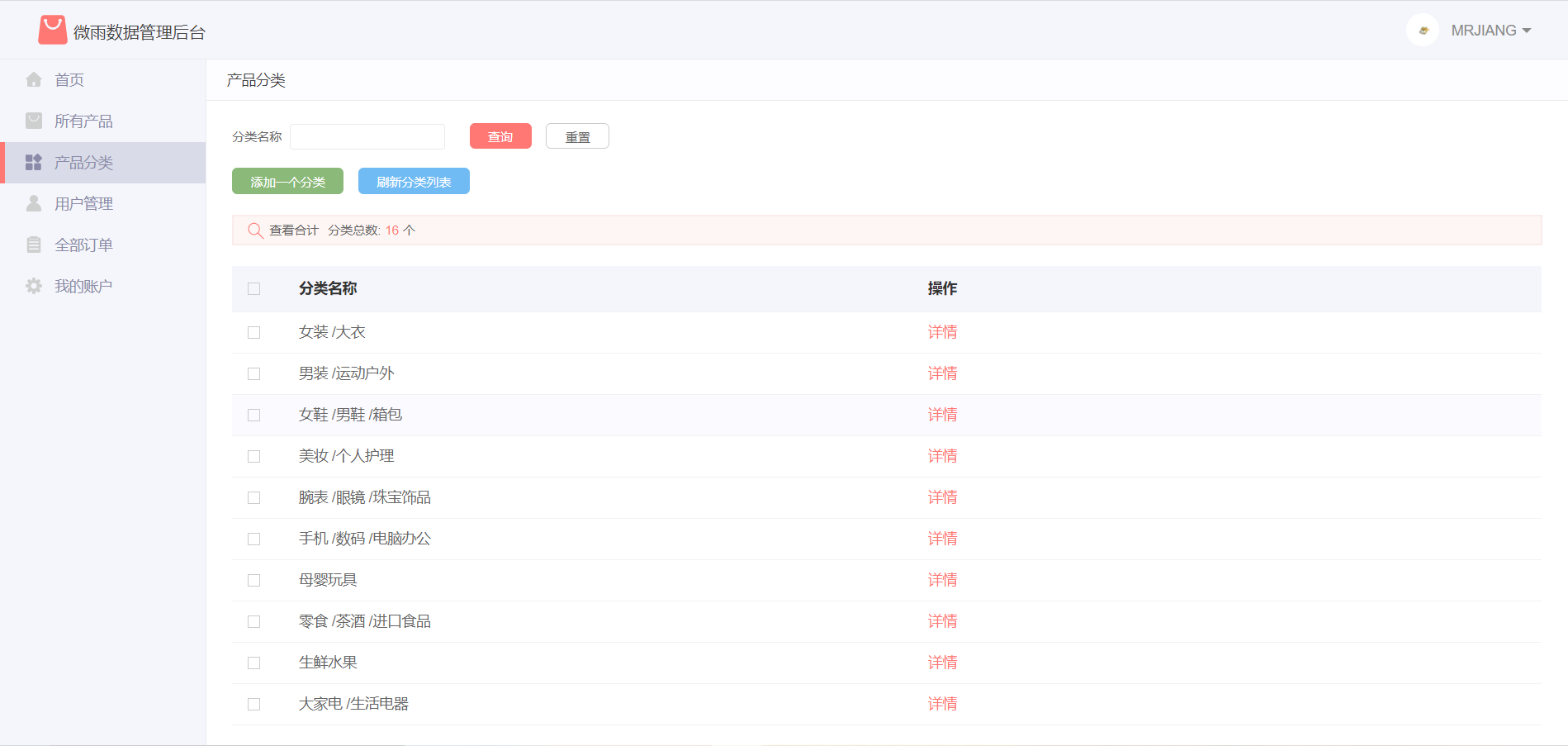


图2-12 商品类别展示界面

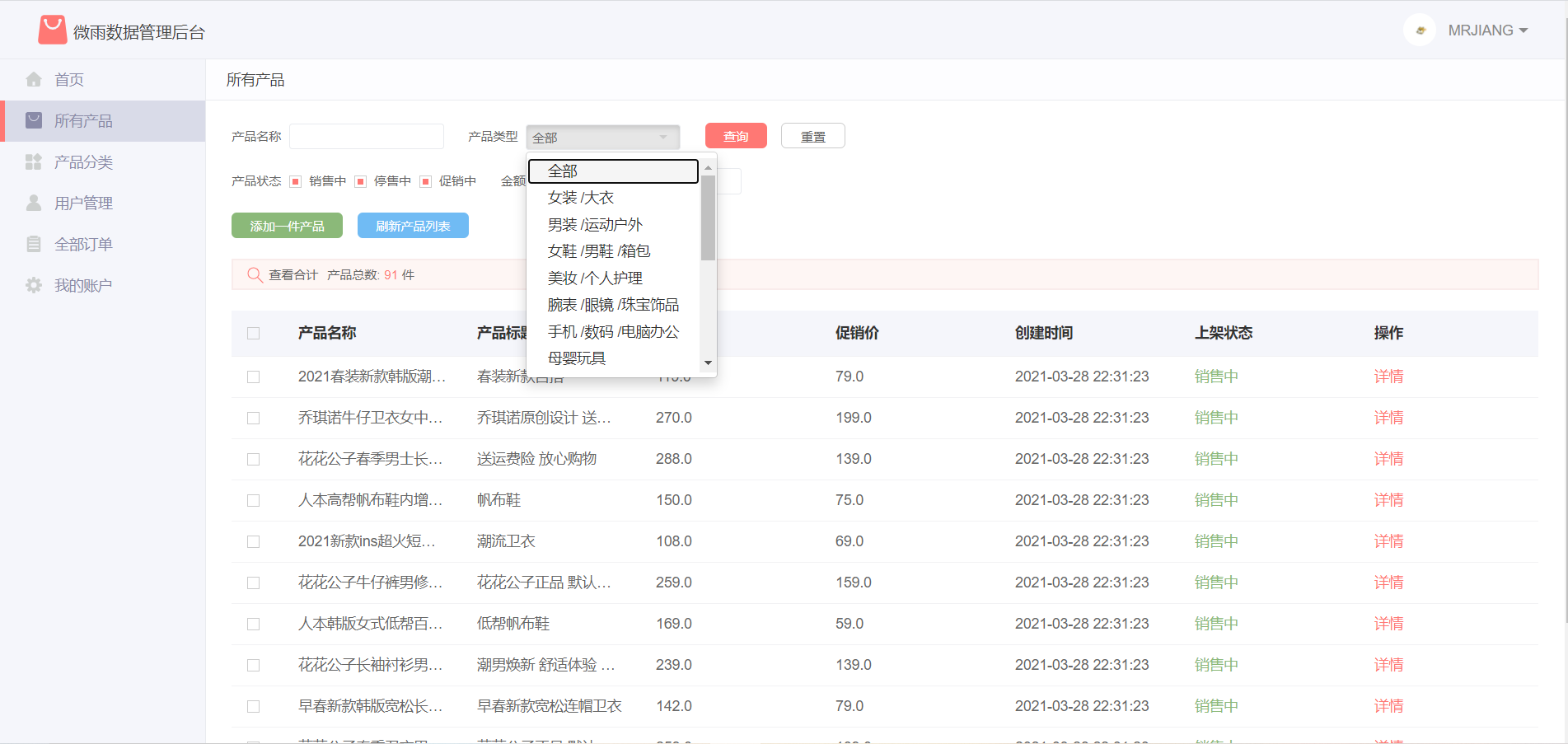


图2-13 商品展示界面

（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：

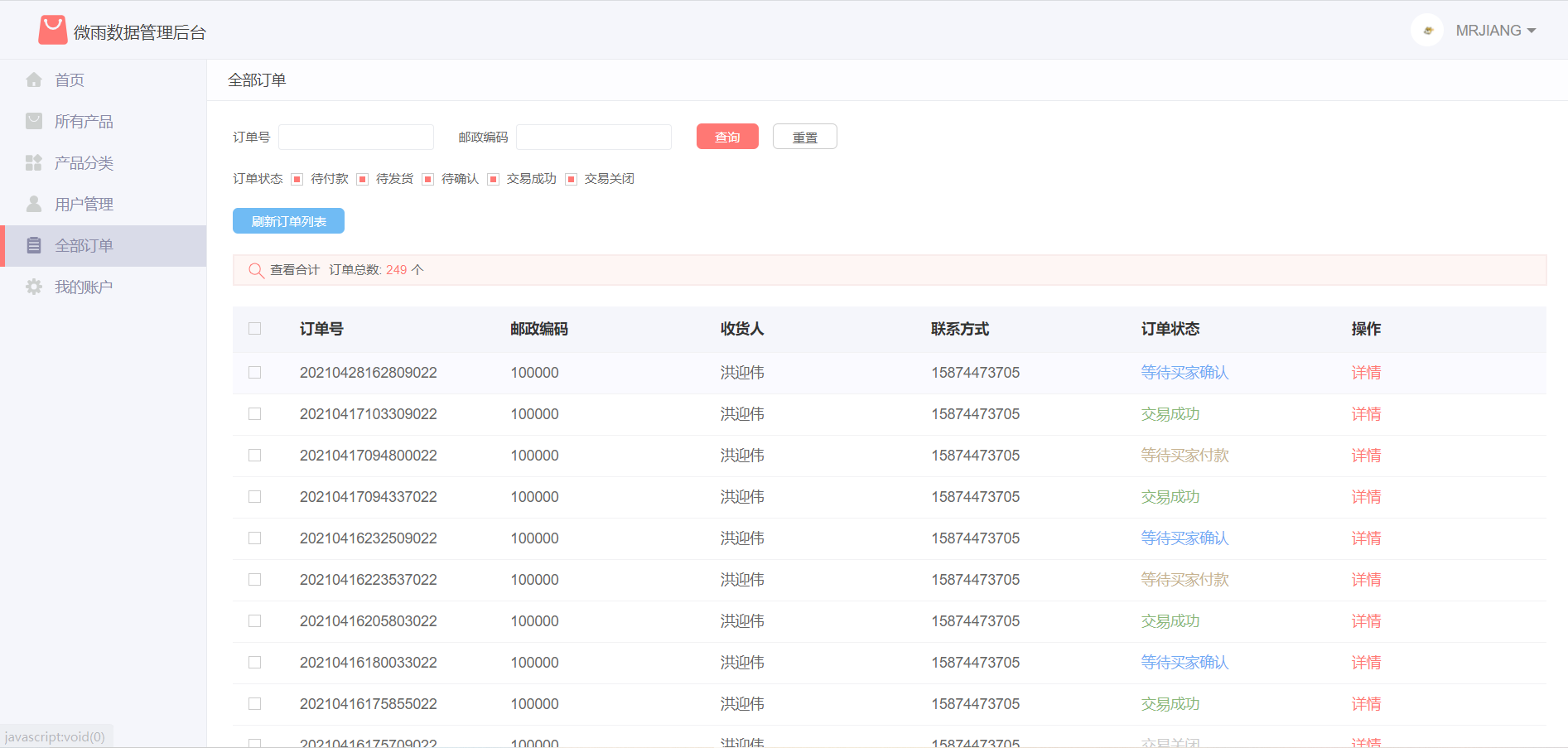


图2-15 订单管理界面

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 |