**基于Tomcat集群与Redis分布式的企业级电商系统的设计与实现**

|  |  |  |  |
| --- | --- | --- | --- |
| 学 院：  专 业：  姓 名：  指导老师： | 计算机学院 | | |
| 软件工程 | | |
| 徐秋冰 | 学 号：  职 称： | 150202102539 |
| 魏志军 | 讲师 |

中国·珠海

二○一九年五月

**诚信承诺书**

**本人郑重承诺：**本人承诺呈交的毕业设计《基于Tomcat集群与Redis分布式的企业级电商系统的设计与实现》是在指导教师的指导下，独立开展研究取得的成果，文中引用他人的观点和材料，均在文后按顺序列出其参考文献，设计使用的数据真实可靠。

本人签名：

日期： 年 月 日

**基于Tomcat集群与Redis分布式的企业级**

**电商系统的设计与实现**

**摘 要**

谈起互联网，就让人不得不联想到电商。在中国，互联网发展已有将近二十年之久了。可以说，最初的互联网，便是伴随着电商而诞生的。现如今，随着互联网发展愈来愈迅猛，对于那些传统企业亦或是准备向电商业务进行拓展的个人或企事业单位，如果想好好利用互联网这个工具，借助这个工具、借势互联网发展浪潮而屹立于行业之巅，那么，能否熟悉当下前沿的开发技术以及掌握核心的产品设计理念，便自然成为了关键因素。

本次研究选题以电商为业务对象，面向现在互联网环境下电商业务日益丰富的场景，旨在提供一种在普通电商系统架构的基础上更加稳定、可靠的架构出来，希望能够解决电商业务在UAD与QPS开始增长的同时，系统性能和响应效率逐渐受到影响，进而阻碍了业务进一步拓宽发展的问题。

本次毕业设计，便是通过以Spring+SpringMVC+Mybatis为基础框架，同时通过对Tomcat容器进行集群，配合Nginx的反向代理以及负载均衡能力，最后依靠Redis作为中间件进行会话的缓存存储，实现一套高可用、高性能的企业级电商系统的目的，来真真正正做到企业级开发，体会现有市场上电商企业在面对日常大量QPS压力的情况下，是如何对现有系统项目进行重构、做二次架构升级的，以此来让自己对电商项目的研发有一个更深入、更深刻的理解。

**关键词：**互联网；电商；高可用高性能；SSM；Tomcat；Nginx

**The Design and Implementation of**

**Enterprise E-commerce System**

**Abstract**

When talking about the Internet, people have to think of e-commerce. In China, the Internet has been developing for nearly two decades. It can be said that the original Internet was born with e-commerce. Nowadays, as the Internet develops more and more rapidly, for those traditional enterprises or individuals or enterprises that are ready to expand into the e-commerce business, if you want to make good use of the Internet, use this tool to develop the Internet. And standing in the footsteps of the industry, then, whether you can become familiar with the current cutting-edge development technology and master the core product design concept, it has naturally become a key factor.

This research topic is based on e-commerce, and it is aimed at providing an increasingly stable and reliable architecture based on the common e-commerce system architecture. It is intended to provide a more stable and reliable architecture based on the existing e-commerce system architecture. While the UAD and QPS began to grow, the system performance and response efficiency were gradually affected, which hindered the further expansion of the business.

This graduation project is based on Spring+SpringMVC+Mybatis, through clustering Tomcat containers, Nginx's reverse proxy and load balancing capabilities, and finally relying on Redis as a middleware for session cache storage. The purpose of a high-availability, high-performance enterprise-class e-commerce system is to truly achieve enterprise-level development, and to understand how existing e-commerce companies in the market are facing the daily pressure of QPS. System project reconstruction and secondary architecture upgrade, in order to let yourself have a deeper and deeper understanding of the development of e-commerce projects.

**Keywords**：Internet; e-commerce; high availability and high performance; SSM; Tomcat; Nginx

**目 录**

[1 引言 1](#_Toc6607118)

[1.1 研究背景 1](#_Toc6607119)

[1.2 研究意义 1](#_Toc6607120)

[1.3 研究内容 1](#_Toc6607121)

[1.4 可行性分析 2](#_Toc6607122)

[1.4.1 技术可行性 2](#_Toc6607123)

[1.4.2 社会可行性 2](#_Toc6607124)

[1.5 论文结构 2](#_Toc6607125)

[2 需求分析 2](#_Toc6607126)

[2.1 非功能性需求 2](#_Toc6607127)

[2.1.1 可靠性 2](#_Toc6607128)

[2.1.2 健壮性 3](#_Toc6607129)

[2.1.3 可维护性 3](#_Toc6607130)

[2.1.4 可扩充性 3](#_Toc6607131)

[2.2 功能性需求分析 3](#_Toc6607132)

[2.2.1 门户端 3](#_Toc6607133)

[2.2.2管理员端 6](#_Toc6607134)

[2.3 详细需求分析 7](#_Toc6607135)

[2.3.1 系统参与者 7](#_Toc6607136)

[2.3.2 系统用例图 8](#_Toc6607137)

[2.3.3 系统用例 9](#_Toc6607138)

[2.3.4系统领域模型 20](#_Toc6607139)

[3 概要设计 21](#_Toc6607140)

[3.1 功能模块设计 21](#_Toc6607141)

[3.1.1 功能模块概述 21](#_Toc6607142)

[3.1.2 功能模块结构图 23](#_Toc6607143)

[3.2 技术选型 24](#_Toc6607144)

[3.2.1 后端设计 24](#_Toc6607145)

[3.2.2 前端设计 25](#_Toc6607146)

[3.3 逻辑视图 26](#_Toc6607147)

[3.4 开发视图 29](#_Toc6607148)

[3.5 物理视图 30](#_Toc6607149)

[3.6 数据视图 30](#_Toc6607150)

[3.6.1 数据库环境说明 30](#_Toc6607151)

[3.6.2 数据库命名规则 30](#_Toc6607152)

[3.6.3 数据库E-R图 31](#_Toc6607153)

[3.6.4 数据字典 31](#_Toc6607154)

[4 详细设计与实现 37](#_Toc6607155)

[4.1 普通用户端 37](#_Toc6607156)

[4.1.1 用户模块 37](#_Toc6607157)

[4.1.2 商品模块 45](#_Toc6607158)

[4.1.3 购物车模块 48](#_Toc6607159)

[4.1.4 收货地址模块 51](#_Toc6607160)

[4.1.5 订单模块 53](#_Toc6607161)

[4.1.6 支付模块 56](#_Toc6607162)

[4.2 管理员端 60](#_Toc6607163)

[4.2.1 用户模块 60](#_Toc6607164)

[4.2.2 商品模块 62](#_Toc6607165)

[4.2.3 订单模块 67](#_Toc6607166)

[4.2.4 品类模块 71](#_Toc6607167)

[5 系统测试 76](#_Toc6607168)

[5.1 测试环境 76](#_Toc6607169)

[5.2 测试范围 76](#_Toc6607170)

[5.3 测试用例 77](#_Toc6607171)

[5.3.1 门户端 77](#_Toc6607172)

[5.3.2 管理员端 82](#_Toc6607173)

[5.4 测试总结 85](#_Toc6607174)

[6 总结与展望 85](#_Toc6607175)

[参考文献 87](#_Toc6607176)

[谢 辞 88](#_Toc6607177)

[附 录 89](#_Toc6607178)

1 引言

1.1 研究背景

当今21世纪，借助互联网+的趋势，能够发现机遇遍地都是。随之互联网发展逐渐趋于成熟，当初伴随着互联网诞生之初的电商，也日渐成熟，成为了我们社会、国家经济快速发展的坚实建设基础。现今依托于互联网之上而搭建起来的电商系统，优秀卓越的能抵抗每秒上亿级别的QPS以及通过其稳健、高性能高可用的独特架构（犹如聚石塔一般），除了带给用户流畅般的使用体验的同时，还能为商家带来丰厚的收益。那么我们如何能实现这种高可用、高性能以及高稳健性的系统架构呢？这便是本次毕设作品需要探索、达成的目标。

1.2 研究意义

现今互联网发展迅猛，任何一款应用、系统或程序，在得到良好的运营的同时，也就意味着能够引入可观的用户流量，而能否快速、稳定地处理这些流量，也是我们互联网软件研发工程师在职业规划道路上能力能否得到不断提升、经验能否得到不断丰富的先决条件。互联网软件研发工程师的水平能力，已经不能单单从会基本的CRUD（增删改查）操作而满足于基本现状了，优秀的工程师更具备出色的架构能力，全局把控能力与部署运维能力。世界著名电商零售巨头美国亚马逊公司，他们公司内部一直信仰这么一句话“Operation Is Excellent”，即“运维即是一切”，这再一次体现出优秀的软件研发工程师，是需要具备上述所述的架构分析、全局把控与稳健运维的能力的。而本次的毕设选题，原因也正是想在满足基本的企业级电商业务能够正常运作的情况下，对其进行架构升级，除了实现QPS的提高以及系统的高性能、稳健性之外，更重要的是对毕设作者自我能力的一次探索，一次深掘、一次突破。相信通过本次毕设作品的设计与实现，作者本人的技术架构能力能够更上一层楼，能够洞悉到技术的本质，去真正享受科技带来的乐趣。

1.3 研究内容

按照电商业务的基本内容，本次毕设采用SSM框架搭建电商系统的基础架构，采用Tomcat作为Web容器，Mysql作为数据库，Redis作为会话缓存中间件，旨在构造出一套高可用、高性能的电商系统，从用户注册、登录（用户模块）、访问网站商城到选择心仪商品（商品模块）放入购物车（购物车模块），然后填写收货地址（收货地址模块）、确认订单（订单模块），到最后的扫码付款（支付模块），整个操作都契合当下市场上面现有的电商业务流程，这样便能通过模拟真实的环境，对企业级电商业务在运作过程中可能遇到的困难以及有哪些可行的解决方案有一个深刻的印象，明确的思路。

1.4 可行性分析

1.4.1 技术可行性

本系统在开源框架的使用上，目前网上有大量的现成资料以及教学视频可供参考学习，同时各种框架以及部署技术在相对应的社区上都有广泛的讨论，因此客观来说可行度非常高。与此同时，本次系统的开发人员也有过一定的实际项目开发经验。

1.4.2 社会可行性

随着社会的进步，人类科学水平的提高，计算机已经成为了人们日常生活中的一部分。现今互联网的快速发展，用户量的指数级上升，对电商系统的响应能力、可用性以及稳健性的要求也越来越高。构建一个高性能、高可用的电商系统，也正是契合了当下电商从业者的需求，继而迎合了这个时代快速发展的需求。

1.5 论文结构

本论文分为六个章节进行撰写。第一章节为引言，介绍本次毕设的研究背景、研究意义和可行性分析等。第二章为需求分析，用来描述本次毕设作品的功能性需求分析和详细需求分析，第三章为概要设计，着重围绕功能模块设计与细节、技术选型、逻辑视图等基于UML建模的视图而展开，第四章便是对本次毕设系统详细设计与实现的撰写，第五章是对系统的测试描写，第六章是本次毕业设计总结与未来展望。

2 需求分析

2.1 非功能性需求

一个合格的、称得上符合软件工程规范的系统，除了满足基本的功能性需求之外，还要满足一些非功能性需求。针对本次系统的研发，将通过可靠性、健壮性、可维护性和可扩充性四个方面进行描述。

2.1.1 可靠性

由于本系统是围绕电商主题进行研发，而电商则意味着较高负载的QPS和PV，为了能够研发出一个在一定时期内能够较好地负载高并发、高流量的系统，系统在基本框架能够提供基本功能的前提下，还要使用Web容器集群与会话缓存服务器分布式部署的方式，依托Web服务器Nginx的反向代理和负载均衡特性，将请求分流到不同的Web容器中，以此来达到系统高可用的特性。

2.1.2 健壮性

面对高流量的请求，如果系统只是单一台服务器部署，那么很可能会因为某一段时间的高并发而导致服务器无法负载过多因而发生宕机现象。因此，考虑到此种情况，使用Nginx的负载均衡——权重调度算法，对不同的服务器定义不同的权重，让请求压力能够尽可能地均匀散布到每一台机器上，以此来提高整个系统的健壮性。

2.1.3 可维护性

一个良好的系统工程，除了满足基本的功能性需求之外，如若发生需求变更、运维计划重置等情况，那么在维护的时候，是否能够较便利、便捷与安全地进行，便成了必要的衡量标准。因此，本次系统采用前后端分离的方式，以及在服务器上采用编写自动化发布脚本的方式，来达到系统在部署、维护时的便捷与简易。

2.1.4 可扩充性

如果在后期，系统做了SEO优化，然后PV和QPS又逐渐提高，那么在进行扩展的时候，只需要在Web服务器中配置新的服务节点，或者在新增节点的基础上采用其他的负载均衡调度算法，那么对于系统来说，既做到了横向扩展需求，也满足了系统性能的提高。

2.2 功能性需求分析

电商系统面向的人群主要有两类：一类是普通用户，他们便是电商业务的主要营销对象，这类用户在我们的电商系统中浏览商品下单商品，与我们形成交易关系；另一类是商城管理员用户，管理员用户负责商品的上下架、订单发货和用户管理等功能。因此本次电商系统会划分为两个子系统进行开发，它们分别是：电商系统门户端和电商系统管理员端。

2.2.1 门户端

门户端面向互联网广大用户，该子系统主要提供在线浏览商品和下订单购买商品的功能，具体功能需求设计如下所示：

（1）用户模块

该模块提供用户与本电商系统交互的功能，具体有用户登录、用户注册、登录前的忘记密码操作、登录状态下的重置密码操作、获取自己在本电商系统中注册的信息操作、更新自己的注册信息操作以及退出登录操作。

【用户登录】 用户在登录界面输入自己已经注册过的用户名和密码，然后点击登录按钮完成登录操作。默认情况下不强制用户必须先登录才能访问网站上的商品信息，只有当用户点击“购物车”或“我的订单”或“我的MMALL”或者“关于MMALL”时，系统才会在当前页面跳转到登录页面，强制用户必须先登录后才能执行之前的操作。

【用户注册】 用户在注册页面，输入合法的用户名、密码、联系电话、邮箱、密码的提示问题和答案后便可点击注册按钮，完成注册。

【忘记密码（登录前）】 如若用户在登录的时候忘记密码，那么便可选择忘记密码按钮，然后在忘记密码提示页面，输入当初注册信息时填写的密码提示问题和密码提示答案，若和当初注册时填写的内容是一致的，那么就允许用户重置自己的密码，用户再次输入新的密码后，便可完成密码重置操作。

【重置密码（登录后）】 用户在成功登录进电商系统后，如若想修改现有的登录密码，那么可以在不用输入密码提示问题和答案的情况下，直接输入新的密码便可完成登录状态下的重置密码操作。

【获取用户信息】 用户在登录状态下，点击“我的MMALL”按钮，便可在个人信息主页上查看当初自己注册时的一些信息，包括用户名、手机、邮箱、密码提示问题和答案。

【更新用户信息】 在登录的状态下，用户可以修改自己当初注册时的一些信息，例如手机、邮箱等。

【退出登录】 在登录状态下，用户可以选择退出登录，以匿名访客的身份浏览本电商系统，但如果想要完成一次购物操作，那么依旧需要先登录成功后，方可执行。

（2）商品模块

该模块提供商品的信息给互联网用户，供其浏览，搜索与查看每一件商品的详情。

【商品搜索】 用户可以在目前现存的成百上千种商品中，搜索自己感兴趣的商品。只要输入商品的关键词便能搜索到。

【动态排序商品列表】商品显示的时候可以按照价格升序或者降序的方式展现出来。

【查看商品详情】 在商品列表里面，用户可以点击某一件商品，进入到该商品的详情页面，查看该商品更加详细的内容。

（3）购物车模块

该模块提供购物车功能，帮助互联网用户将自己心仪的商品加入购物车，然后通过购物车可以查看自己已经选择的商品，需要的时候也可以直接增加（或移除）某件已选择的商品，然后通过勾选想要付款的商品，完成商品的购买。

【添加商品】 用户在某件商品的详情页面内，如果确认自己想要购买这件商品，那么便可以将该件商品添加进购物车，方便后续的下订单操作。

【修改商品数量】 用户可以修改目前已添加进购物车的商品的数量，增加或者减少某样商品的数量。

【移除商品】 在购物车内，用户也可以选择移除已经添加进来的某件商品。

【列表展示】 可以展示用户目前已经添加进购物车中的所有商品，包括商品基本信息、已选数量和价格。

【选中/取消选中商品】 在购物车内，用户可以通过勾选商品的方式来决定最终要购买的商品到底有哪些。选中分为单个商品选中和商品全选；取消选中也分为单个商品取消选中和所有商品取消选中。

（4）收货地址模块

该模块提供用户在完成订单时也需要填写自己的收货地址的功能。允许用户新增收货地址、修改收货地址和删除收货地址。

【新增收货地址】 用户可以在订单确认页新增自己的收货地址，填写收件人姓名、所在省市、详细地址、收货人电话和邮政编码即可。

【编辑收货地址】 用户可以编辑已经保存过的收货地址，可编辑的包括收件人姓名、所在省市、详细地址、收货人电话和邮政编码。

【删除收货地址】 用户可以选择删除已经添加的一条收货地址。

（5）订单模块

该模块为本系统与用户建立直接的交易关系。该模块提供用户创建订单、查看订单详情（列表）与取消订单的功能。

【创建订单】 用户在订单确认页面，确认自己所要购买的商品信息（包括数量和价格）和收货地址后，点击创建订单按钮，即可创建一张订单。

【查看订单列表】 用户在“我的订单”页面，可以查看到自己迄今为止所创建过的所有订单，包括成功支付过的订单和取消支付过的订单。

【查看订单详情】 在订单列表里面，用户可以选择查看某一笔订单的详细内容，包括该笔订单的创建时间、订单号、收件人、订单支付状态和支付方式等，还可以看到该笔订单对应的商品清单。

【取消订单】 对于仍未支付的订单，用户有权取消与本系统已经建立的交易关系，选择取消订单便可。

（6）支付模块

该模块作为互联网用户购买商品的最后一个业务流程，当跳转到订单支付页面时，系统会在页面上显示一个二维码，用户使用支付宝扫码支付，然后在手机端输入支付密码完成支付。

【订单支付】 在订单支付页面，系统会显示出一个支付的二维码，用户使用手机进行扫码支付。

2.2.2管理员端

管理员端是面向小范围的管理员用户的，该子系统帮助他们管理自己商城的商品信息以及上下架情况，同时承担着确认用户购买的商品是否已经发货的职责。该子系统的功能模块设计如下所示：

（1）用户模块

该模块提供给管理员查看当前电商系统中已注册的用户的信息的功能。

【用户列表】 管理员可以查看到当前系统中已经注册的用户的信息。

（2）商品模块

该模块提供给管理员查看当前商城中存在的所有商品的功能。这些商品包括上下架的，同时还提供给管理员新增商品、查看指定商品详情与编辑商品内容的功能。

【商品列表】 给管理员展示当前商城中存在的所有商品，包括已上架的和已下架的。同时支持管理员按照商品ID或者商品名称进行查询。

【商品搜索】 管理员可以根据商品名称或者商品Id，查找特定的商品信息。

【商品详情】 提供给管理员查看某一件商品详细内容的功能。

【商品编辑】 提供给管理员编辑某一件商品内容的功能。管理员可以修改商品名称、价格、数量、上传图片及富文本等。

【商品添加】 提供给管理员新增商品的功能。管理员需要填写商品名称、商品描述、所属分类、商品价格和商品库存、商品图片以及商品详情（富文本内容）。

【商品上下架】 根据实际情况，提供给管理员上下架某件商品的功能。管理员可以让已经上架的商品转变为下架状态，这样在商城中便不再显示该件商品；管理员也可以让已经下架的商品转变为上架状态，这样在商城中该件商品便又能被重新检索到。

（3）订单模块

该模块提供给管理员查看商城系统中迄今为止已创建过的所有订单的功能。系统以分页的形式展现给管理员订单列表，管理员可以查看某笔订单的详情内容，必要的时候还可以修改订单状态。

【订单列表】 系统展示给管理员迄今为止已创建过的所有订单，按照分页的形式展现给管理员。管理员也可以通过订单号进行订单检索。

【订单搜索】管理员可以根据订单号，查询指定的订单。

【订单详情】 管理员可以查看某一笔订单的详情内容。包括该笔订单的订单号、创建时间、收件人、订单状态和商品列表等。

【订单发货】 对于已经付款的订单，管理员可以把该订单状态从已付款修改为已发货的状态。

（4）品类管理模块

该模块提供给管理员维护商品种类的功能。管理员可以新添加一个品类，或者在目前现存的品类列表中修改某个品类的名称，抑或是查看某个品类下面的子品类。

【品类列表】 管理员可以查看某个品类下面的所有子品类。默认情况下系统显示根节点下面的子品类内容。

【品类名称修改】 管理员可以修改某个品类的名称。

【查看子品类】 管理员可以查看某个品类下面的所有子品类列表。

【添加品类】 管理员可以选择在指定的品类节点下面新增一个子品类。

2.3 详细需求分析

2.3.1 系统参与者

该系统的参与者分为三类，一类是匿名访客用户，一类是普通用户，一类是管理员用户。每种用户可以执行的行为都不一样。参与者的具体信息如表2.1所示：

表2.1 参与者信息表

|  |  |
| --- | --- |
| 参与者 | 行为 |
| 匿名访客用户 | 浏览商品列表、查看商品详情、注册 |
| 普通用户 | 用户登录、浏览（搜索）商品列表、查看商品详情、添加商品进购物车、填写收货地址、确认订单、支付订单 |
| 商城管理员 | 用户登录、查看普通用户信息列表、查看（搜索）商品列表、查看商品详情、编辑商品信息、新增商品、商品种类名称修改、商品种类新增、商品种类浏览 |

【匿名访客用户】 只要是未注册本电商系统的访客，都属于匿名访客用户。匿名访客用户可以查看商城中的商品列表，抑或是查看某件商品的详细信息，但无法购买商品。注册本电商系统，然后成功登陆后方可购买商品。

【普通用户】 登录本电商系统门户端的用户即是普通用户。普通用户可以浏览（搜索）商品，查看某件商品的详细内容、对心仪的商品，可以添加进购物车，然后填写收货地址确认订单，最后支付订单，完成一次购物流程。

【商城管理员】 登录本电商系统管理员端的用户即是商城管理员。商城管理员可以对商城目前现有的商品进行维护，然后还可查看注册商城的用户信息。

2.3.2 系统用例图

门户端用例图，如下图2.1所示：

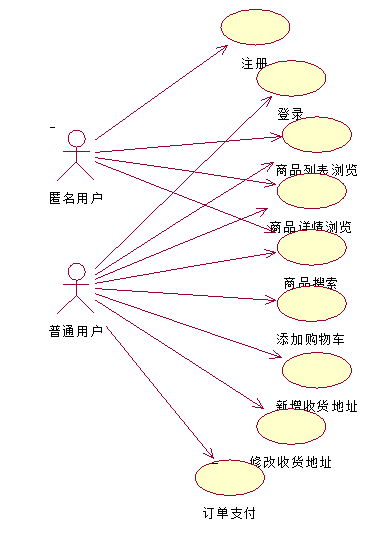


图2.1 匿名用户用例图

管理员端用例图，如下图2.2所示：

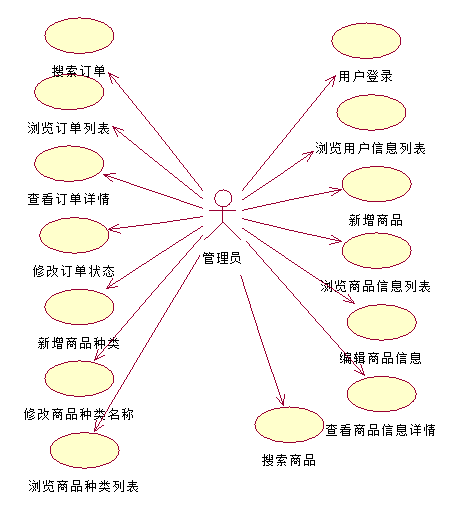


图2.2 管理员端用户图

2.3.3 系统用例

2.3.3.1 门户端主要用例

2.3.3.1.1 <UC1 用户注册>

**范围：**门户端

**级别：**用户目标

**主要参与者：**匿名用户

**涉众及其关注点：**

匿名用户：通过输入合法的用户名、密码、联系电话、邮箱、密码提示问题和密码提示答案，成功注册成为本电商系统的普通用户。

**前置条件：**匿名用户知晓电商系统的访问地址并且能够成功访问到电商系统。

**主要流程：**

1. 匿名用户访问电商系统网址；
2. 在电商系统首页的左上角，点击“注册”按钮；
3. 进入到注册页面，输入合法的信息；
4. 信息输入完毕，点击注册按钮，等待系统校验信息；
5. 系统校验通过，本次注册操作成功。

**扩展：**

3a. 若匿名用户输入的用户名或者邮箱已被别人使用过，页面上会提示相关信息，引导用户重新填写；

3b. 若匿名用户有些必填信息没有输入，点击“注册”按钮时，系统进行相关提示，提示必填项不能为空；

5a. 系统校验失败，弹出提示相关信息，并引导用户进行相关操作。

2.3.3.1.2 <UC2 浏览商品列表>

**范围：**门户端

**级别：**用户目标

**主要参与者：**匿名用户、普通用户

**涉众及其关注点：**

匿名用户：在未注册、登录的情况下能够访问商城系统首页，浏览到商品列表。

普通用户：在登录的情况下，进入商城系统首页，能够浏览到商品列表。

**前置条件：**用户知晓商城系统的访问地址。

**主要流程：**

1. 用户在浏览器上输入商城的访问地址；
2. 成功访问到商城系统首页，然后能浏览到商品列表。

**扩展：**

2a. 普通用户执行登录操作成功后，进入商城系统首页；

2b. 匿名用户不需要经过登录操作，直接进入商城系统首页。

2.2.3.1.3 <UC3 浏览商品详情>

**范围：**门户端

**级别：**用户目标

**主要参与者：**匿名用户、普通用户

**涉众及其关注点：**

匿名用户：在未注册、未登录的情况下能够访问商城系统，在商品列表中点击某个商品，能够进入到该件商品的详情页面内，浏览该件商品的详情内容。

普通用户：在登录的情况下，进入商城系统首页，在商城列表中随意点击某件商品，能够进入到该件商品的详情页面内，浏览该件商品的详情内容。

**前置条件：**用户知晓商城系统的访问地址。

**主要流程：**

1. 用户在浏览器上输入商城的访问地址；
2. 成功访问到商城系统首页，查看商品列表；
3. 随意点击商品列表中的某个商品，然后进入该件商品的详情内容页面。

**扩展：**

2a. 普通用户执行登录操作成功后，进入商城系统首页；

2b. 匿名用户不需要经过登录操作，直接进入商城系统首页。

2.3.3.1.4 <UC4 搜索商品>

**范围：**门户端

**级别：**用户目标

**主要参与者：**匿名用户、普通用户

**涉众及其关注点：**

匿名用户：在未注册、未登录的情况下能够访问电商系统，在商城首页通过关键字能够执行搜索商品的操作。

普通用户：在登录的情况下，在商城首页通过关键字能够执行搜索商品的操作。

**前置条件：**用户知晓商城系统的访问地址。

**主要流程：**

1. 用户在浏览器上输入商城的访问地址；
2. 成功访问到商城系统首页，然后在首页搜索框里面输入关键字，搜索想要查看的商品；
3. 点击搜索按钮进行搜索，系统将返回携带搜索关键字的商品列表给用户。

**扩展：**

2a. 普通用户执行登录操作成功后，进入商城系统首页；

2b. 匿名用户不需要经过登录操作，直接进入商城系统首页；

3a. 如果用户想要搜索的商品不存在本商城中，系统会在搜索结果页面内进行相关的文字提示。

2.3.3.1.5 <UC5 登录>

**范围：**门户端

**级别：**用户目标

**主要参与者：**普通用户

**涉众及其关注点：**

普通用户：在成功访问到电商系统的情况下，输入合法的用户名和密码，成功登录电商系统。

**前置条件：**用户已注册

**主要流程：**

1. 用户访问电商系统的登录页面；
2. 在登录页面上输入合法的用户名和密码；
3. 点击“登录按钮”，等待系统校验登录信息；
4. 系统校验成功，页面自动跳转到系统首页。

**扩展：**

2a. 若用户输入的用户名或者密码为空，点击“登录”按钮时，系统进行相关提示，提示用户名或密码不能为空；

4a. 若系统校验登录信息校验失败，返回错误信息，并提示给用户。

2.3.3.1.6 <UC6 添加购物车>

**范围：**门户端

**级别：**用户目标

**主要参与者：**普通用户

**涉众及其关注点：**

普通用户：在登录状态下，对心仪的商品点击“加入购物车”按钮，系统能正确跳转到成功加入购物车的页面。

**前置条件：**用户已经登录进本电商系统。

**主要流程：**

1. 选择某件商品，进入该件商品的内容详情页；
2. 在内容详情页点击“加入购物车”按钮；
3. 系统进行页面跳转，跳转到商品成功添加进购物车页面。

**扩展：**

3a. 当商品成功添加进购物车后，用户可以选择直接进入结算流程或者返回商品首页继续购物

2.3.3.1.7 <UC7 新增收货地址>

**范围：**门户端

**级别：**用户目标

**主要参与者：**普通用户

**涉众及其关注点：**

普通用户：在订单确认页面，选择新增收货地址，然后填上相关地址信息，点击“保存”按钮，系统能够成功保存。

**前置条件：**用户已经登录，有选择心仪的商品并加入购物车

**主要流程：**

1. 在我的购物车页面，商品信息确认无误，点击“去结算”按钮；
2. 在订单结算页面，选择新增收货地址操作；
3. 在弹出框中填写相关的地址信息；
4. 填写完成后，点击“保存收货地址”按钮，等待系统校验信息；
5. 系统校验成功，收货地址新增成功。

**扩展：**

3a. 收货地址信息中，若用户没有输入必填项，系统提示相关信息，引导用户必须输入必填项后才能保存收货地址。

2.3.3.1.8 <UC8 修改收货地址>

**范围：**门户端

**级别：**用户目标

**主要参与者：**普通用户

**涉众及其关注点：**

普通用户：在订单确认页面，能够对某个收货地址信息进行修改。

**前置条件：**用户已经登录，有选择心仪的商品并加入了购物车。

**主要流程：**

1. 在订单结算页面，选择先前已创建好的一个收货地址；
2. 点击“编辑”按钮，然后在弹出框中重新编辑相关信息；
3. 编辑完成后，点击“保存收货地址”按钮；
4. 等待系统提交编辑信息，提交完成后弹出框自动消失，本次编辑操作成功。

**扩展：**

3a. 若编辑时有必填项没有填，在点击“保存收货地址”按钮时，系统弹出提示框，提示用户必填项不能为空。

2.3.3.1.9 <UC9 支付订单>

**范围：**门户端

**级别：**用户目标

**主要参与者：**普通用户

**涉众及其关注点：**

普通用户：确认订单无误，提交订单后，能看到系统显示的支付二维码，然后通过手机扫码，能成功支付此次订单交易。

**前置条件：**用户确认订单无误，准备进行订单提交

**主要流程：**

1. 用户点击“提交订单”按钮，进入订单支付页面；
2. 订单支付页面会显示出一张二维码，供用户扫码支付；
3. 用户使用手机进行扫码支付；
4. 等待支付宝发送回调过来，系统进行支付状态确认。
5. 支付宝发送支付成功的回调过来，系统收到回调后，页面进行跳转，显示支付成功。

**扩展：**

5a. 若支付宝5分钟内没有发送回调过来，系统进行异常信息提示，引导用户重新下订单并支付。

2.3.3.2 管理员端主要用例

2.3.3.2.1 <UC1 登录>

**范围：**管理员端

**级别：**用户目标

**主要参与者：**管理员

**涉众及其关注点：**

管理员：使用合法的账号密码能够成功登录电商系统的管理员端。

**前置条件：**管理员知晓电商系统管理端的访问地址

**主要流程：**

1. 浏览器窗口输入电商系统管理员端地址，进入到电商系统管理员端的登录界面；
2. 输入正确的账号和密码；
3. 点击“登录”按钮，等待系统校验登录信息；
4. 登录信息系统校验成功，页面跳转到管理端首页。

**扩展：**

3a. 若有账号或者密码没有输入，当执行登录操作时，系统弹出信息框，提示账号或者密码不能为空；

4a. 若输入的账号或者密码是错误的，系统弹出信息框，提示账号或者密码错误。

2.3.3.2.2 <UC2 浏览用户信息列表>

**范围：**管理员端

**级别：**用户目标

**主要参与者：**管理员

**涉众及其关注点：**

管理员：登录成功进入操作系统首页，点击“用户管理”导航按钮，能够浏览到网站迄今为止已注册过的所有用户信息（id、用户名、邮箱、联系方式、注册时间）。

**前置条件：**管理员已成功登录进电商管理系统。

**主要流程：**

1. 管理员点击左侧导航栏目里面的“用户管理”按钮；

2. 页面显示用户列表。

2.3.3.2.3 <UC3 新增商品>

**范围：**管理员端

**级别：**用户目标

**主要参与者：**管理员

**涉众及其关注点：**

管理员：成功登陆进电商管理系统后，能够执行新增商品操作，并且操作能够成功。

**前置条件：**管理员已成功登陆进电商管理系统。

**主要流程：**

1. 管理员点击左侧导航栏目里面的“商品管理”按钮；
2. 在商品管理页面右上角点击“添加商品”按钮；
3. 添加新商品的相关信息；
4. 点击“提交”按钮，等待系统校验信息；
5. 系统校验成功，页面跳转回商品列表页面。

**扩展：**

4a. 若有必填项（比如商品名称、商品描述等）没有填，在执行保存操作时，系统弹出提示框，提示必填项不能为空。

2.3.3.2.4 <UC4 编辑商品>

**范围：**管理员端

**级别：**用户目标

**主要参与者：**管理员

**涉众及其关注点：**

管理员：成功登陆进电商管理系统后，管理员能执行编辑商品操作，并且新的商品信息能编辑保存成功。

**前置条件：**管理员已成功登录进电商管理系统。

**主要流程：**

1. 管理员点击左侧导航栏目里面的“商品管理”按钮；
2. 在商品列表中，选择需要编辑的商品，在该商品行中点击“编辑”按钮，进入编辑页面；
3. 编辑商品的相关信息；
4. 点击“保存”按钮，等待系统校验编辑信息；
5. 信息校验通过，页面跳转回商品列表页面。

**扩展：**

5a. 若有必填项（比如商品名称、商品描述等）没有填，在执行保存操作时，系统弹出提示框，提示必填项不能为空。

2.3.3.2.5 <UC5 浏览商品列表>

**范围：**管理员端

**级别：**用户目标

**主要参与者：**管理员

**涉众及其关注点：**

管理员：成功登陆进电商管理系统后，点击“商品管理”导航按钮，能查看到目前商城中已存在的所有商品（包括已上架和已下架的）。

**前置条件：**管理员已成功登陆进电商管理系统

**主要流程：**

1. 管理员点击左侧导航栏目里面的“商品管理”按钮；
2. 导航栏右侧页面会显示目前商城中已存在的所有商品信息列表，例如商品Id、商品名称、价格、上下架状态等（包括已上架和已下架的）。

2.3.3.2.6 <UC6 查看商品详情>

**范围：**管理员端

**级别：**用户目标

**主要参与者：**管理员

**涉众及其关注点：**

管理员：在商品信息列表里，随意选择某个商品，点击“详情”按钮，能够查看到该件商品的详情内容。

**前置条件：**管理员已成功登录进电商管理系统。

**主要流程：**

1. 管理员点击左侧导航栏目里面的“商品管理”按钮；
2. 在导航栏右侧，显示出来商品信息列表后，随意选择某个商品，点击该行商品右侧的“详情”按钮；

3. 导航栏右侧页面跳转到该件商品的详情内容页。

2.3.3.2.7 <UC7 浏览订单列表>

**范围：**管理员端

**级别：**用户目标

**主要参与者：**管理员

**涉众及其关注点：**

管理员：在商城系统首页，点击左侧导航栏的“订单管理”按钮，能够查看到商城迄今为止创建过的所有订单信息列表。

**前置条件：**管理员已成功登录进电商管理系统。

**主要流程：**

1. 管理员点击左侧导航栏目里面的“订单管理”按钮；

2. 导航栏目右侧页面会显示商城迄今为止创建过的所有订单信息列表（包括订单号、收件人、订单状态、订单总价等）。

2.3.3.2.8 <UC8 查看订单详情>

**范围：**管理员端

**级别：**用户目标

**主要参与者：**管理员

**涉众及其关注点：**

管理员：在商城系统首页，点击左侧导航栏的“订单管理”按钮后，能够查看到商城迄今为止所创建过的所有订单信息列表。

**前置条件：**管理员已成功登录进电商管理系统。

**主要流程：**

1. 点击页面左侧导航栏的“订单管理”按钮；

2. 导航栏右侧页面会显示商城迄今为止所创建过的所有订单信息列表（包括订单号、收件人、订单状态等）。

2.3.3.2.9 <UC9 修改订单状态>

**范围：**管理员端

**级别：**用户目标

**主要参与者：**管理员

**涉众及其关注点：**

管理员：对于已经支付成功的订单，管理员能够修改该笔订单的状态，从已支付的状态修改为已发货状态。

**前置条件：**管理员已成功登录进电商管理系统。

**主要流程：**

1. 点击页面左侧导航栏的“订单管理”按钮；
2. 等待导航栏右侧订单信息列表显示完全后，选择订单状态为已支付的订单，进入该笔订单的详情内容页面；
3. 在订单详情内容页面，定位到订单状态一行，点击“发货”按钮，等待系统执行修改订单状态的操作；
4. 系统操作成功，页面弹出一个信息提示框，提示订单发货成功，点击“确认”按钮后，页面自动跳转到订单信息列表页。

**扩展：**

4a. 若系统操作失败，页面会弹出提示框，进行相关的错误提示。

2.3.3.2.10 <UC10 浏览商品种类列表>

**范围：**管理员端

**级别：**用户目标

**主要参与者：**管理员

**涉众及其关注点：**

管理员：能够查看商品种类信息列表，包括每一种种类的子种类的列表信息。

**前置条件：**管理员已成功登陆进电商管理系统。

**主要流程：**

1. 点击页面左侧导航栏的“品类管理”按钮；
2. 导航栏右侧显示品类列表，默认显示根品类节点下面的子品类信息；
3. 点击品类列表中某一行品类的子品类信息；

4. 页面能够显示该品类节点的子品类信息列表。

2.3.3.2.11 <UC11 新增商品种类>

**范围：**管理员端

**级别：**用户目标

**主要参与者：**管理员

**涉众及其关注点：**

管理员：能新增商品种类。

**前置条件：**管理员已成功登陆进电商管理系统。

**主要流程：**

1. 在“品类列表”页面，点击右上角的“添加品类”按钮；
2. 选择所属分类，然后输入新的品类名称；
3. 点击“提交”按钮，等待系统校验信息；
4. 系统校验成功，页面弹出操作成功的信息提示框，点击“确认”按钮后，页面自动跳转到品类列表页面。

**扩展：**

2a. 若没有输入新的品类名称就点击“提交”按钮，页面会弹出提示框，提示用户品类名称不能为空。

2.3.3.2.12 <UC12 修改品类名称>

**范围：**管理员端

**级别：**用户目标

**主要参与者：**管理员

**涉众及其关注点：**

管理员：能对现有的品类名称进行修改。

**前置条件：**管理员已成功登陆进电商管理系统。

**主要流程：**

1. 在“品类列表”页面，选中需要修改名称的品类节点；
2. 点击左侧的“修改名称”按钮，这时页面会弹出一个输入框，供我们输入新的品类名称；
3. 输入新的品类名称，然后点击“确认”按钮，等待系统执行修改操作；
4. 系统执行操作成功，页面弹出提示框，提示用户修改品类名称成功。

**扩展：**

3a. 若在修改框中什么也没有输入的话，系统不做任何响应。

2.3.3.2.13 <UC13 搜索商品>

**范围：**管理员端

**级别：**用户目标

**主要参与者：**管理员

**涉众及其关注点：**

管理员：能够根据商品Id或者商品名称，搜索得到相关的商品信息。

**前置条件：**管理员已成功登陆进电商管理系统。

**主要流程：**

1. 在“商品列表”页面，选择按商品Id（或按商品名称）查询；
2. 输入准确的商品Id（或者商品名称关键字），点击“搜索”按钮；
3. 等待系统执行搜索操作，然后返回相关的结果；

**扩展：**

2a. 若选择查询条件后，没有在查询框中输入任何内容，系统则是默认查询全部商品信息列表出来。

2.3.3.2.14 <UC14 搜索订单>

**范围：**管理员

**级别：**用户目标

**主要参与者：**管理员

**涉众及其关注点：**

管理员：能够根据订单号，查询出指定的订单。

**前置条件：**管理员已成功登陆进电商管理系统。

**主要流程：**

1. 在“订单列表”页面，在订单号输入框中输入订单号，然后点击“搜索”按钮；
2. 等待系统执行搜索操作，然后将指定的订单记录返回回来。

**扩展：**

1a. 如果管理员没有输入订单号，却点击了“搜索”按钮，那么系统默认是查找全部的订单记录出来。

2.3.4系统领域模型

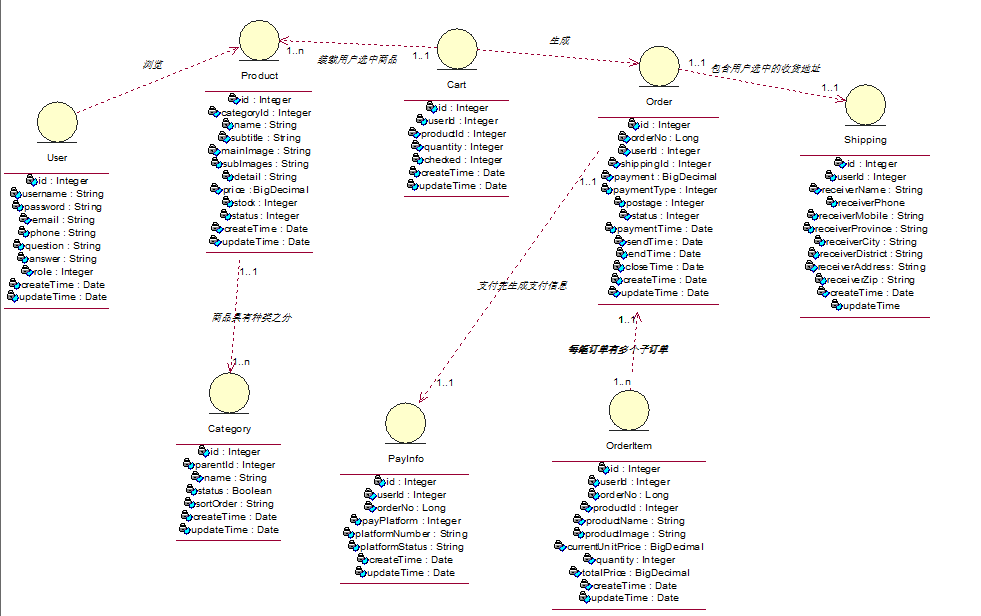


图2.3 系统领域模型图

3 概要设计

3.1 功能模块设计

3.1.1 功能模块概述

本次电商系统实际分为两个子系统进行研发。这两个子系统分别是电商系统门户端和电商系统管理员端，面向的人群分别是普通互联网用户和管理员。

电商系统门户端的功能模块设计如下表所示：

表3.1 电商系统门户端的功能模块设计表

|  |  |  |
| --- | --- | --- |
| 功能模块 | 功能名称 | 描述 |
| 用户模块 | 注册 | 注册本电商系统，成为其普通用户，然后可以在电商系统里面购物。 |
| 用户模块 | 登录 | 登录本电商系统，然后可以在里面购物。 |
| 用户模块 | 重置密码(登录前) | 登录前忘记密码，通过密码提示问题，输入问题答案后可以重置自己的登录密码。 |
| 用户模块 | 重置密码(登录后) | 在登录状态下，直接修改自己的登录密码。 |
| 用户模块 | 获取用户信息 | 在登录状态下，可以查看到自己当初注册电商系统时填写的相关信息。 |
| 用户模块 | 修改用户信息 | 在登录状态下，可以修改自己当初注册电商系统时填写的相关信息。 |
| 用户模块 | 退出登录 | 在登录状态下，退出电商系统。以匿名用户的身份访问电商系统。 |
| 商品模块 | 搜索商品 | 能够按照商品关键字对某些特定商品进行搜索。 |
| 商品模块 | 动态排序商品列表 | 能够按照价格对商品列表进行升序或者降序排序。 |
| 商品模块 | 查看商品详情 | 能够查看某个商品的详情内容。 |
| 购物车模块 | 添加商品 | 在登录状态下，可以把心仪的商品添加进购物车，方便结算。 |
| 购物车模块 | 修改商品数量 | 在登录状态下，可以修改购物车中商品的数量。数量最小值为1，最大值为该商品当前的库存量。 |
| 购物车模块 | 移除商品 | 在登录状态下，可以从购物车里面移除某件商品，使得该件商品的价格不计入结算流程。 |
| 购物车模块 | 列表展示 | 在登录状态下，可以查看目前已经加入购物车中的所有商品。 |
| 购物车模块 | 选中/取消选中商品 | 在登录状态下，可以进一步选择是否把商品纳入结算流程。 |
| 收货地址模块 | 新增收货地址 | 在订单确认页面，可以新增自己的收货地址（填写地址的相关信息即可）。 |
| 收货地址模块 | 编辑收货地址信息 | 在订单确认页面，可以编辑目前已存在的收货地址的相关信息。 |
| 收货地址模块 | 删除收货地址 | 在订单确认页面，可以删除目前已存在的收货地址信息。 |
| 订单模块 | 创建订单 | 用户可以创建一张订单。 |
| 订单模块 | 查看订单列表 | 用户可以查看迄今为止自己所创建过的所有订单。 |
| 订单模块 | 查看订单详情 | 用户可以查看某一张订单的详情内容。 |
| 订单模块 | 取消订单 | 在订单还未成功支付时，用户可以取消订单。 |
| 支付模块 | 支付订单 | 通过手机支付宝扫码支付，用户可以为创建的订单进行支付操作。 |

电商系统管理员端的功能模块设计如下表所示：

表3.2 电商系统管理员端的功能模块设计表

|  |  |  |
| --- | --- | --- |
| 功能模块 | 功能名称 | 描述 |
| 用户模块 | 浏览用户列表 | 管理员可以查看迄今为止商城已注册的所有用户信息。 |
| 商品模块 | 浏览商品列表 | 管理员可以查看到商城中现存的所有商品信息。 |
| 商品模块 | 搜索商品 | 管理员可以根据商品名称或者商品Id搜索特定的商品 |
| 商品模块 | 查看商品详情 | 管理员可以查看某个商品的详情内容信息。 |
| 商品模块 | 编辑商品 | 管理员可以编辑某个商品的具体信息。 |
| 商品模块 | 添加商品 | 管理员可以添加新的商品。 |
| 商品模块 | 修改商品上/下架状态 | 管理员可以对商品进行上下架状态的修改。 |
| 订单模块 | 浏览订单列表 | 管理员可以浏览迄今为止商城中存在的所有订单。 |
| 订单模块 | 搜索订单 | 管理员可以根据订单号查询指定的订单 |
| 订单模块 | 查看订单详情 | 管理员可以查看某一笔订单的详细内容。 |
| 订单模块 | 修改订单状态 | 管理员可以修改订单状态，从已支付的状态修改为已发货状态。 |
| 品类模块 | 浏览品类列表 | 管理员可以查看商城中存在的所有品类信息，包括品类Id和品类名称。 |
| 品类模块 | 查看子品类列表 | 管理员可以查看某一品类的所有子品类信息列表 |
| 品类模块 | 添加品类信息 | 管理员可以新增品类信息。 |
| 品类模块 | 修改品类名称 | 管理员可以修改某个品类的名称。 |

3.1.2 功能模块结构图

本电商系统分为两个子系统，分别是电商系统门户端和电商系统管理员端，系统划分如下图所示：

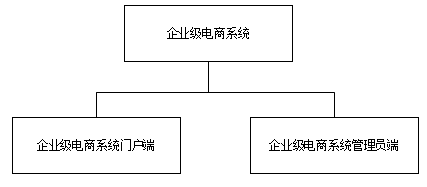


图3.1 电商系统划分图

电商系统门户端的功能结构图如下图所示：

图3.2 电商系统门户端功能结构图



电商系统管理员端的功能结构图如下图所示：

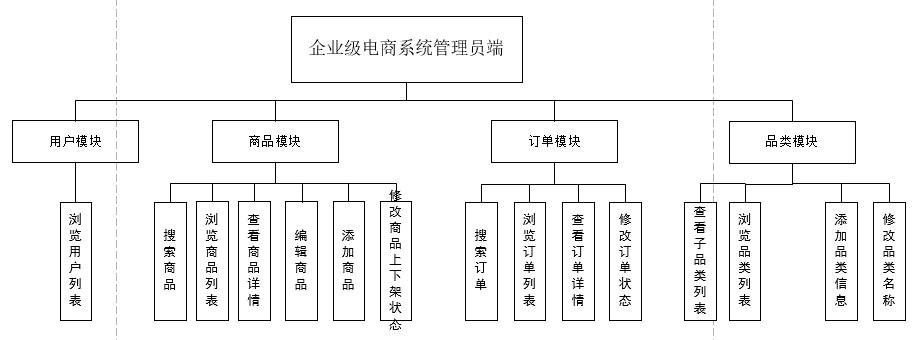


图3.3 电商系统管理员端功能结构图

3.2 技术选型

本电商系统是基于B/S架构，前后端分离的方式进行开发的。门户端的前端采用原生html+css+Node.js+jQuery组合进行开发，管理员端的前端采用React+React-Router+Node.js进行开发；门户端和管理员端的后端统一采用目前市场上较流行的SSM（Spring+SpringMVC+Mybatis）框架进行开发，该框架组合也是秉承了MVC分层架构的理念。

3.2.1 后端设计

电商系统门户端和管理员端统一采用目前市场上较为流行的SSM框架，用来搭建项目的基础环境。针对MVC分层架构的设计理念，springMVC在本次项目中充当了Controller的角色，MyBatis在本次项目中充当了Model的角色。由于本电商系统是前后端分离的，后端响应客户端请求的方式是基于RESTFul风格响应的，因此后端不会返回任何视图（即View）给前端，其只会返回某次请求对应的操作数据库而产生的结果（即Data）回去。除此之外，考虑到Web容器是集群部署的，为了解决集群环境下如何对用户请求产生的会话共享这一问题，本次项目还使用到了中间件Redis，作为用户会话的高速缓存，来应对高并发、高QPS情况下后台因需要校验身份而频繁请求会话缓存的问题。最后，由于本电商系统门户端和管理员端两个子系统是部署在同一台服务器上的，为了确保不同的请求能够对应到不同的系统服务，这里也采用了Nginx，进行反向代理服务以及Web容器集群下的负载均衡服务。具体信息如下表所示：

表3.3 后端开发环境信息表

|  |  |
| --- | --- |
| 开发语言 | Java (jdk1.8) |
| 开发框架 | Spring 4.0.3.RELEASE + SpringMVC 4.0.3 RELEASE + Mybatis 3.4.1 |
| 开发工具 | Intellij IDEA 15.0.6 |
| 系统环境 | CentOS 6.7, 64 bit |
| 数据库 | MySQL 5.7.17-log Community Server |
| 中间件 | Redis 2.8 |
| 服务器 | Nginx 1.10.2 |

3.2.2 前端设计

本次电商系统分为了两个子系统：一个是电商系统门户端，一个是电商系统管理员端。针对这两个子系统，它们的前端设计也是不同的。

3.2.2.1 电商系统门户端

电商系统门户端的前端设计，采用了原生开发。使用html+css+jQuery，用来构建页面视图；使用Node.js+Ajax，用来编写页面与用户的交互逻辑以及与后台进行请求交互。项目构建使用到了Webpack，项目中的依赖采用npm进行管理。具体信息如下表所示：

表3.4 电商系统门户端——前端开发环境信息表

|  |  |
| --- | --- |
| 开发语言 | HTML5 + CSS3 + Node.js (CommonJS) v6.12.3 |
| 所使用到的技术 | Jquery、Ajax |
| 开发工具 | Sublime Version 3.0 Build 3143 |
| 系统环境 | Windows / Linux |
| 浏览器 | Chrome、Firefox、Safari等主流浏览器 |

3.2.2.2 电商系统管理员端

电商系统管理员端的前端设计，采用了目前市场上主流的React框架进行搭建。考虑到管理员端的受众使用范围没有门户端的受众使用范围广，因此也借助于React-Router的路由特性，进行单页应用开发。除此之外，对于项目代码工程化构建与打包，使用的是Webpack3.0系列，依赖管理本次使用了Yarn。具体信息如下表所示：

表 3.5 电商系统管理员端——前端开发环境信息表

|  |  |
| --- | --- |
| 开发语言 | HTML5 + SASS + Node.js (ES6) v6.12.3 |
| 开发框架 | React16 React-Router4 |
| 开发工具 | Sublime Version 3.0 Build 3413 |
| 系统环境 | Windows / Linux |
| 浏览器 | Chrome、Firefox、Safari等主流浏览器 |

3.3 逻辑视图

本系统前后端分离，电商系统门户端和管理员端的后端架构是统一的；但它们的前端架构是不一样的，后面会对这两个子系统的前端逻辑视图分别进行阐述。

电商系统门户端和管理员端的后端都采用SSM框架进行搭建，遵循标准的MVC架构设计理念，可以分为中间件层（Common）、控制层（Controller）、业务层（Service）、数据访问层（Dao）以及模型层（Model）：

* 中间件层（Common）：用来对请求真正抵达业务层之前做一些预处理。比如AuthorityInterceptor，用来对身份权限进行统一校验；SessionExpireFilter，用来重置会话过期时间的过滤器。
* 控制层（Controller）：负责对请求进行分发调度，委派给不同的Service去处理，同时对Service层的执行结果以JSON格式返回给前端。
* 业务层（Service）：负责处理真正的电商业务逻辑，必要时通过Dao层与数据库进行交互，然后将查询结果返回给上一层。
* 数据访问层（Dao）：负责与数据库交互，然后将交互结果返回给上一层。
* 模型层（Model）：维护业务实体与数据库表之间的映射关系。

其逻辑视图如下图所示：

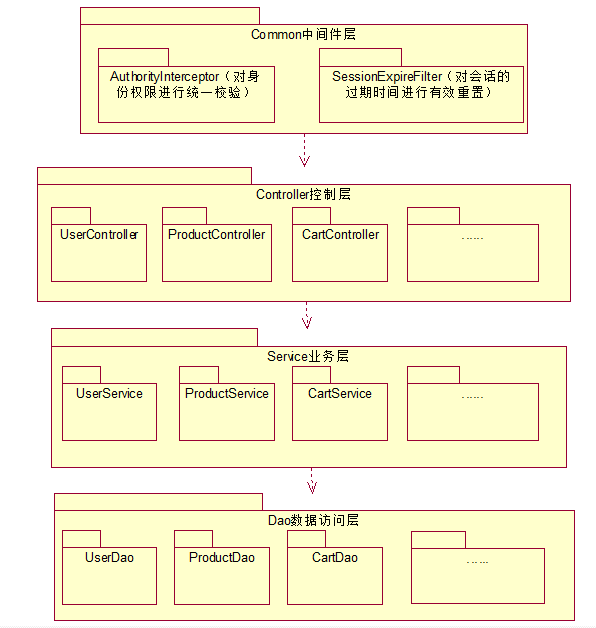


图 3.4 电商系统门户端和管理员端的后端逻辑视图

对于电商系统门户端的前端设计，由于采用的是原生开发，因此我们把门户端的开发框架思维分为交互逻辑层（Page）、视图层（View）、工具层（Util）以及服务层（Service）：

* 交互逻辑层（Page）：用于编写与用户交互的逻辑代码，诸如点击按钮、自动加载、填写文本、选择列表等事件触发时会发生的逻辑。
* 视图层（View）：使用HTML5和CSS对页面的样式进行绘制，设计。
* 工具层（Util）：提供Ajax请求、分页等功能的支持。
* 服务层（Service）：针对不同业务模块，对具体的业务操作进行封装。

其逻辑视图如下图所示：

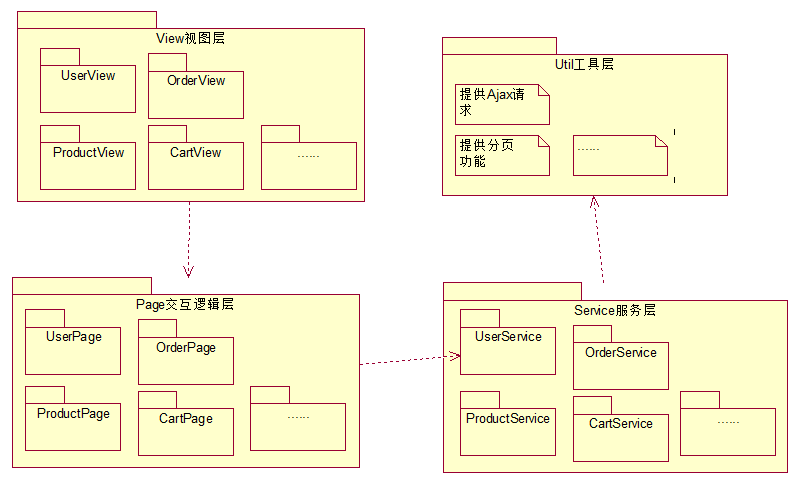


图 3.5 电商系统门户端的前端逻辑视图

对于电商系统管理员端的前端设计，我们使用了主流的前端开发框架React。由于管理员端的前端项目是单页应用，因此针对单页应用，我们把项目的开发架构分为组件层（Component）、逻辑层（Page）、服务层（Service）与工具层（Util）：

* 组件层（Component）：比如构成单页页面的导航栏、头标、布局、公共区域等。
* 逻辑层（Page）：用于编写与用户交互的逻辑代码，诸如点击按钮、自动加载、填写文本、选择列表等事件触发时会发生的逻辑。
* 服务层（Service）：针对不同业务模块，对具体的业务操作进行封装。
* 工具层（Util）：提供Ajax请求、分页、文件上传以及富文本上传等功能的支持。

其逻辑视图如下图所示：

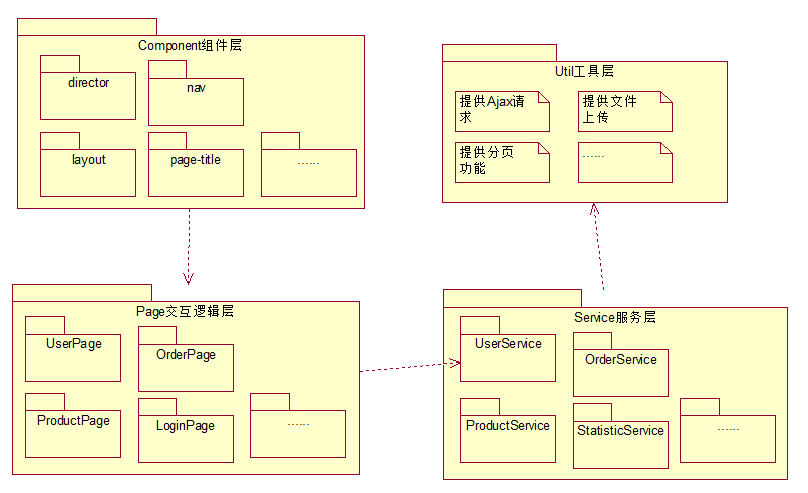


图 3.6 电商系统管理员端的前端逻辑视图

3.4 开发视图

根据3.3 的逻辑视图，本系统的开发视图可以由如下图所示：

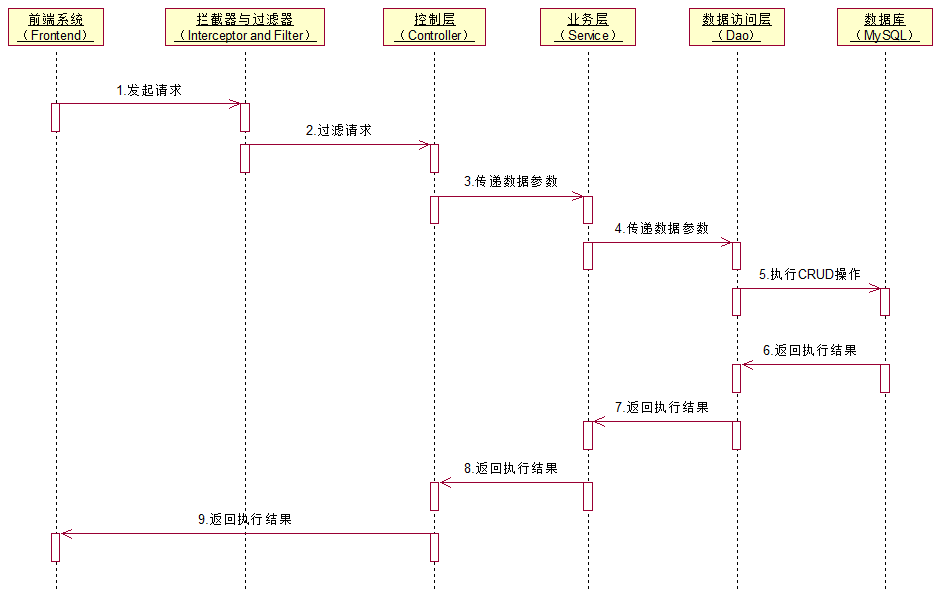


图 3.7 系统开发视图

根据此系统开发视图，当用户（普通用户亦或是管理员用户）在前端（Frontend）通过Ajax的异步方式发起请求时，请求首先被过滤器（Filter）捕获，过滤器被用来对已经鉴权的用户的会话进行过期时间重置，然后被放行。接着，如果请求涉及到管理员相关的操作，那么还会被拦截器（Interceptor）所拦截，用来验证此次请求是否是身份为管理员的用户发起的，确认是管理员后才会放行。紧接着，真正的请求就由控制器（Controller）获取了，控制器根据请求信息，把请求再次转发到业务层（Service），业务层接收到请求携带的参数后，对参数进行必要的处理，然后通过数据访问层（Dao），与数据库（Database）进行交互。然后再把执行的结果以JSON格式、按照原先的请求传递路径传递回去，最后在前端把JSON数据渲染出来，异步刷新页面。

3.5 物理视图

本系统是基于B/S（Browser/Server，即浏览器/服务器）架构进行开发的。在物理视图中，共分为两个终端。

第一个终端是客户端，第二个终端是服务端。客户端通过浏览器与服务端建立http连接，然后在有效的连接期间内，彼此以Request-Response的形式通信。服务端采用all in one 的形式，将Web服务器、Web容器（集群形态）、数据库以及缓存服务器都部署在同一台机子上。具体的部署图如下图所示：

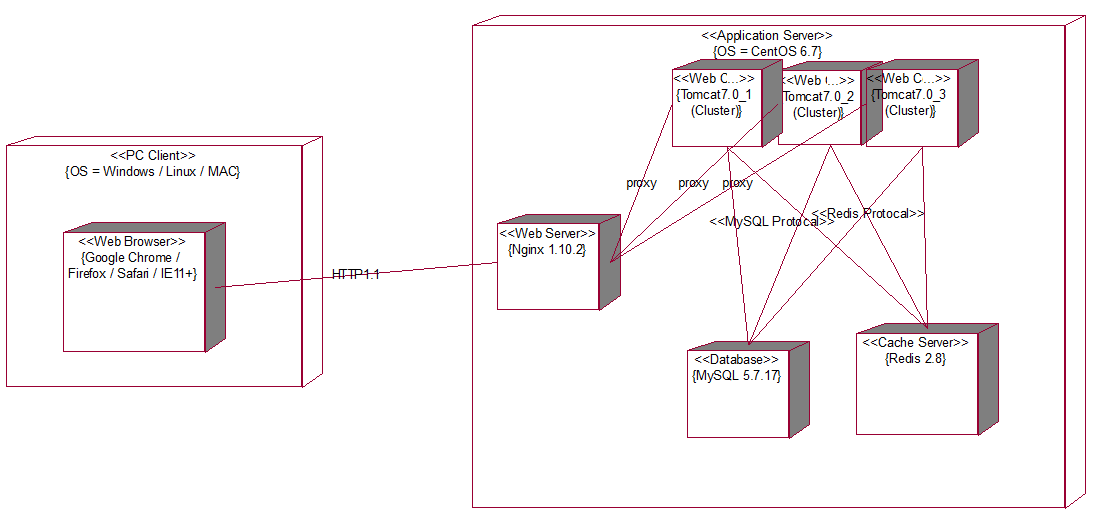


图 3.8 系统物理视图

3.6 数据视图

3.6.1 数据库环境说明

数据库名称：mmall

数据库类型及版本：MySQL 5.1.17-log Community Server

3.6.2 数据库命名规则

数据库命名规则如下表所示：

表3.6 数据库命名规则表

|  |  |
| --- | --- |
| 缩写、术语 | 描述 |
| mmall | 数据库名称 |
| mmall\_user | 用户表 |
| mmall\_product | 商品表 |
| mmall\_category | 种类表 |
| mmall\_shipping | 收货地址表 |
| mmall\_cart | 购物车表 |
| mmall\_pay\_info | 支付信息表 |
| mmall\_order | 订单表 |
| mmall\_order\_item | 订单详情表 |

3.6.3 数据库E-R图

本系统的数据库实体关系图如下图所示：

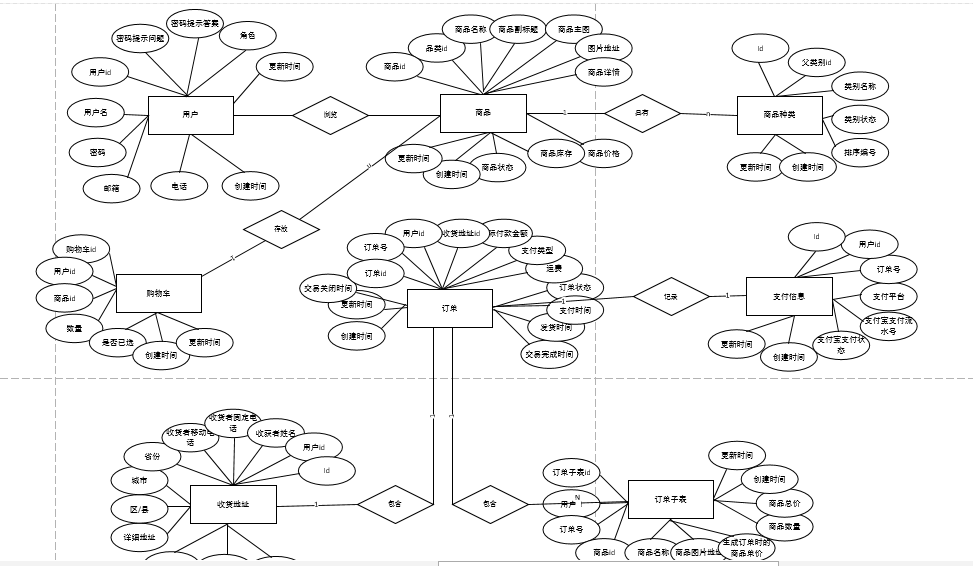


图 3.9 数据库实体关系E-R图

3.6.4 数据字典

表3.7 数据库总表

|  |  |  |
| --- | --- | --- |
| 表名 | 中文名 | 描述 |
| mmall\_user | 用户表 | 存储用户的基本信息，比如账号、密码（MD5加密）、姓名、邮箱等。 |
| mmall\_product | 商品表 | 存储商品的基本信息，比如商品名称、单价、库存等。 |
| mmall\_category | 品类表 | 存储品类的基本信息，比如品类名称、Id、父品类Id等。 |
| mmall\_shipping | 收货地址表 | 存储用户收货地址的基本信息，比如收货人姓名、地址、联系电话等。 |
| mmall\_cart | 购物车表 | 存储用户的购物车中的商品信息，比如某个用户的某次购物车中某个商品的选购数量，是否有勾选购买等。 |
| mmall\_pay\_info | 支付信息表 | 存储用户支付订单的基本信息，比如订单号、支付方式、支付状态等。 |
| mmall\_order | 订单表 | 存储用户的订单信息，比如订单号、该笔订单的总金额、支付时间等。 |
| mmall\_order\_item | 订单详情表 | 存储订单对应的详情信息，比如某笔订单下面包括了哪些商品，这些商品的名称、选购数量、总价等。 |

表3.8 用户表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **表名** | **用户表 – mmall\_user** | | | | | |
| **字段名** | **数据类型** | **长度** | **非空** | **约束条件** | **中文描述** | **备注** |
| id | int | 11 | 是 | 唯一，标识 | 用户ID | 主键，自增 |
| username | varchar | 50 | 是 |  | 用户名 |  |
| password | varchar | 50 | 是 |  | 用户密码，MD5加密 |  |
| email | varchar | 50 |  |  | 用户邮箱 |  |
| phone | varchar | 20 |  |  | 联系电话 |  |
| question | varchar | 100 |  |  | 找回密码问题 |  |
| answer | varchar | 100 |  |  | 找回密码答案 |  |
| role | int | 4 | 是 |  | 角色 0-管理员，1-用户 |  |
| create\_time | datetime | 0 | 是 |  | 创建时间 |  |
| update\_time | datetime | 0 | 是 |  | 最后一次更新时间 |  |

表3.9 商品表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **表名** | **商品表 – mmall\_product** | | | | | |
| **字段名** | **数据类型** | **长度** | **非空** | **约束条件** | **中文描述** | **备注** |
| id | int | 11 | 是 | 唯一，标识 | 商品ID | 主键，自增 |
| category\_id | int | 11 | 是 |  | 分类id,对应mmall\_category表的主键 |  |
| name | varchar | 100 | 是 |  | 商品名称 |  |
| subtitle | varchar | 200 |  |  | 商品副标题 |  |
| main\_image | varchar | 500 |  |  | 产品主图,url相对地址 |  |
| sub\_images | text | 0 |  |  | 图片地址,json格式,扩展用 |  |
| detail | text | 0 |  |  | 商品详情 |  |
| price | decimal | 20 | 是 |  | 价格,单位-元保留两位小数 |  |
| stock | int | 11 | 是 |  | 库存数量 |  |
| status | int | 6 |  |  | 商品状态.1-在售 2-下架 3-删除 |  |
| create\_time | datetime | 0 |  |  | 创建时间 |  |
| update\_time | datetime | 0 |  |  | 最后一次更新时间 |  |

表3.10 品类表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **表名** | **品类表 – mmall\_category** | | | | | |
| **字段名** | **数据类型** | **长度** | **非空** | **约束条件** | **中文描述** | **备注** |
| id | int | 11 | 是 | 唯一，标识 | 类别ID | 主键，自增 |
| parent\_id | int | 11 |  |  | 父类别id当id=0时说明是根节点,一级类别 |  |
| name | varchar | 50 |  |  | 品类名称 |  |
| status | tinyint | 1 |  |  | 类别状态1-正常,2-已废弃 |  |
| sort\_order | int | 4 |  |  | 排序编号,同类展示顺序,数值相等则自然排序 |  |
| create\_time | datetime | 0 |  |  | 创建时间 |  |
| update\_time | datetime | 0 |  |  | 最后一次更新时间 |  |

表3.11 收货地址表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **表名** | **收货地址表 – mmall\_shipping** | | | | | |
| **字段名** | **数据类型** | **长度** | **非空** | **约束条件** | **中文描述** | **备注** |
| id | int | 11 | 是 | 唯一，标识 | 收货地址ID | 主键，自增 |
| user\_id | int | 11 |  |  | 用户id |  |
| receiver\_name | varchar | 20 |  |  | 收货姓名 |  |
| receiver\_phone | varchar | 20 |  |  | 收货固定电话 |  |
| receiver\_mobile | varchar | 20 |  |  | 收货移动电话 |  |
| receiver\_province | varchar | 20 |  |  | 省份 |  |
| receiver\_city | varchar | 20 |  |  | 城市 |  |
| receiver\_district | varchar | 20 |  |  | 区/县 |  |
| receiver\_address | varchar | 200 |  |  | 详细地址 |  |
| receiver\_zip | varchar | 6 |  |  | 邮编 |  |
| create\_time | datetime | 0 |  |  | 创建时间 |  |
| update\_time | datetime | 0 |  |  | 最后一次更新时间 |  |

表3.12 购物车表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **表名** | **购物车表 – mmall\_cart** | | | | | |
| **字段名** | **数据类型** | **长度** | **非空** | **约束条件** | **中文描述** | **备注** |
| id | int | 11 | 是 | 唯一，标识 | 购物车ID | 主键，自增 |
| user\_id | int | 11 | 是 |  | 用户id |  |
| product\_id | int | 11 |  |  | 商品id |  |
| quantity | int | 11 |  |  | 数量 |  |
| checked | int | 11 |  |  | 是否选择,1=已勾选,0=未勾选 |  |
| create\_time | datetime | 0 |  |  | 创建时间 |  |
| update\_time | datetime | 0 |  |  | 最后一次更新时间 |  |

表3.13 支付信息表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **表名** | **支付信息表 – mmall\_pay\_info** | | | | | |
| **字段名** | **数据类型** | **长度** | **非空** | **约束条件** | **中文描述** | **备注** |
| id | int | 11 | 是 | 唯一，标识 | 收货地址ID | 主键，自增 |
| user\_id | int | 11 |  |  | 用户id |  |
| order\_no | bigint | 20 |  |  | 订单号 |  |
| pay\_platform | int | 10 |  |  | 支付平台:1-支付宝,2-微信 |  |
| platform\_number | varchar | 200 |  |  | 支付宝支付流水号 |  |
| platform\_status | varchar | 20 |  |  | 支付宝支付状态 |  |
| create\_time | datetime | 0 |  |  | 创建时间 |  |
| update\_time | datetime | 0 |  |  | 最后一次更新时间 |  |

表3.14 订单表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **表名** | **订单表 – mmall\_order** | | | | | |
| **字段名** | **数据类型** | **长度** | **非空** | **约束条件** | **中文描述** | **备注** |
| id | int | 11 | 是 | 唯一，标识 | 收货地址ID | 主键，自增 |
| order\_no | bigint | 20 |  |  | 订单号 |  |
| user\_id | int | 11 |  |  | 用户id |  |
| shipping\_id | int | 11 |  |  | 收货地址id |  |
| payment | decimal | 20 |  |  | 实际付款金额,单位是元,保留两位小数 |  |
| payment\_type | int | 4 |  |  | 支付类型,1-在线支付 |  |
| postage | int | 10 |  |  | 运费,单位是元 |  |
| status | int | 10 |  |  | 订单状态:0-已取消-10-未付款，20-已付款，40-已发货，50-交易成功，60-交易关闭 |  |
| payment\_time | datetime | 0 |  |  | 支付时间 |  |
| send\_time | datetime | 0 |  |  | 发货时间 |  |
| end\_time | datetime | 0 |  |  | 交易完成时间 |  |
| close\_time | datetime | 0 |  |  | 交易关闭时间 |  |
| create\_time | datetime | 0 |  |  | 创建时间 |  |
| update\_time | datetime | 0 |  |  | 最后一次更新时间 |  |

表3.15 订单详情表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **表名** | **订单详情表 – mmall\_order\_item** | | | | | |
| **字段名** | **数据类型** | **长度** | **非空** | **约束条件** | **中文描述** | **备注** |
| id | int | 11 | 是 | 唯一，标识 | 收货地址ID | 主键，自增 |
| user\_id | int | 11 |  |  | 用户id |  |
| order\_no | bigint | 20 |  |  | 订单号 |  |
| product\_id | int | 11 |  |  | 商品id |  |
| product\_name | varchar | 100 |  |  | 商品名称 |  |
| product\_image | varchar | 500 |  |  | 商品图片地址 |  |
| current\_unit\_price | decimal | 20 |  |  | 生成订单时的商品单价，单位是元,保留两位小数 |  |
| quantity | int | 10 |  |  | 商品数量 |  |
| total\_price | decimal | 20 |  |  | 商品总价,单位是元,保留两位小数 |  |
| create\_time | datetime | 0 |  |  | 创建时间 |  |
| update\_time | datetime | 0 |  |  | 最后一次更新时间 |  |

4 详细设计与实现

4.1 普通用户端

4.1.1 用户模块

4.1.1.1 注册

注册是匿名用户成为本电商系统普通用户的唯一途径。匿名用户访问到本电商系统的首页后，在首页左上角点击“注册”按钮，进入注册页面（如图4.1所示），输入合法的用户名、密码、联系电话等相关信息，然后点击“立即注册”按钮，然后完成注册。值得一提的是，用户在输入用户名、联系电话和邮箱的时候，页面会执行Ajax异步验证逻辑，及时验证用户输入的用户名等是否会和其他用户重复，如果重复，页面会弹出提示信息，引导用户重新填写（如图4.2所示）；否则进入正常的注册逻辑。



图4.1 系统注册页面

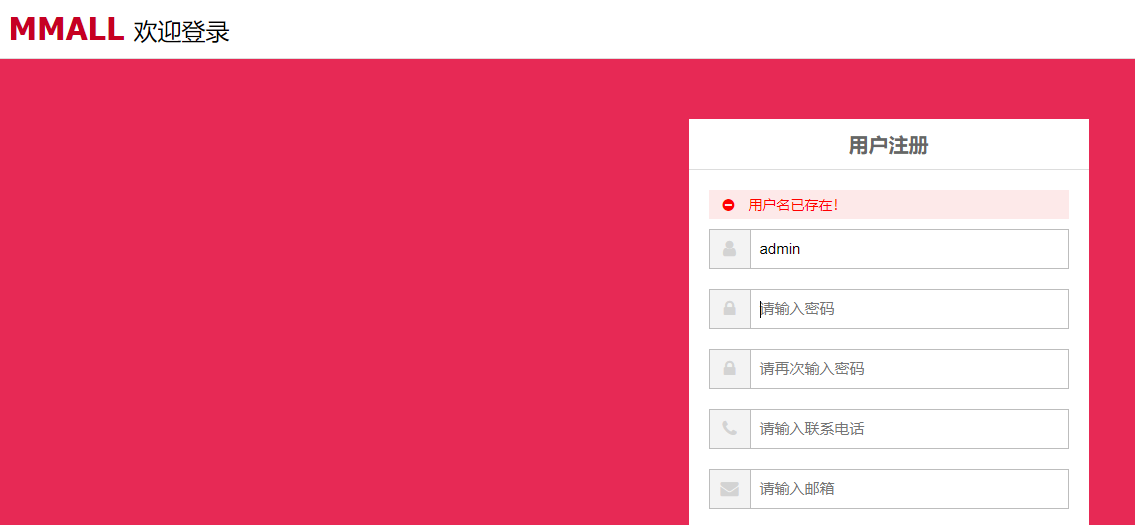


图4.2 注册时错误提示

4.1.1.2 登录

匿名用户成功登录本电商系统后，便可以真正购买自己心仪的商品。在登录页面（如图4.3所示），输入合法的账号密码，点击“登录”按钮，等待系统后台进行身份校验，如果身份校验成功，便自动跳转到首页（亦或是先前访问系统的某个页面）；否则在登录窗口弹出错误信息。错误信息分为两种：

* 用户输入了原本就不存在系统数据库里面的信息，也即用户不存在错误；
* 用户输入的账号存在系统数据库里面，但输入的密码和数据库里面已经存有的密码无法匹配上，也即密码错误。

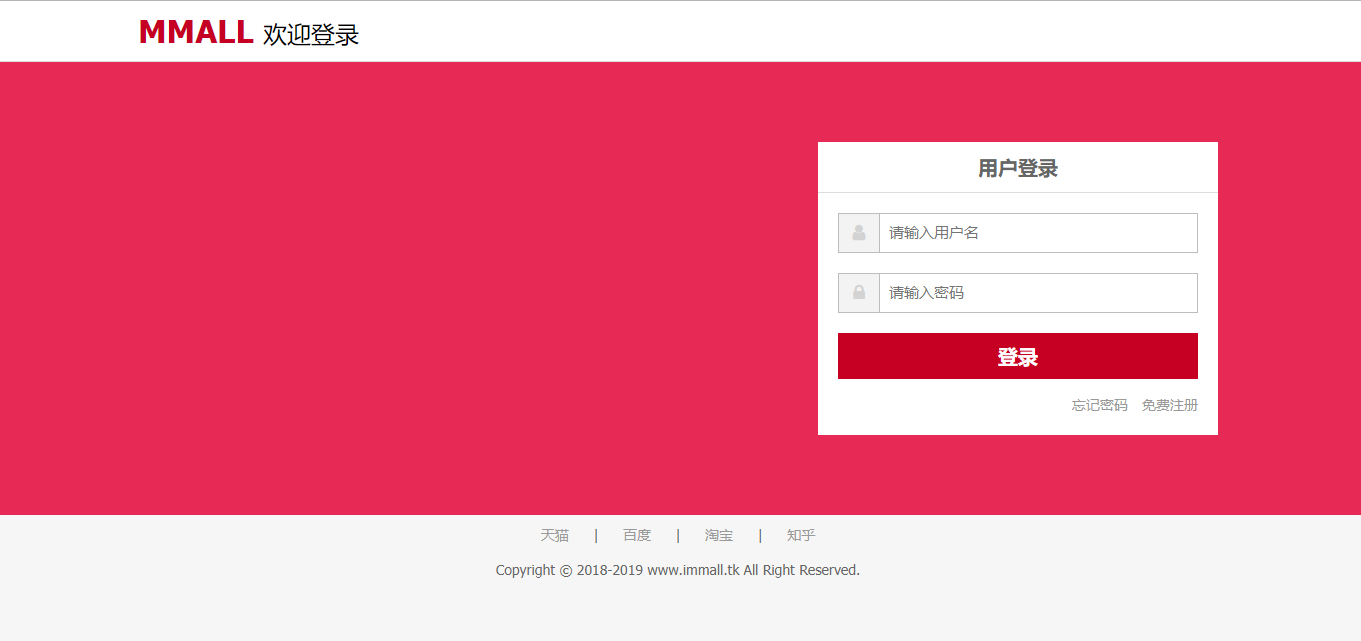
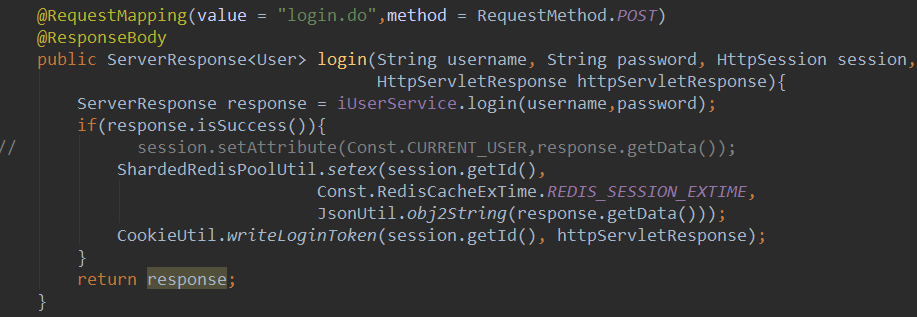


图4.3 用户登录页面

登录完成后，系统后端会把该用户本次请求产生的SessionId连同该用户信息一起存储进Redis里面（以分布式的方式存储），在Redis中，采用String的数据结构来维护用户请求产生的会话与自身信息之间的对应关系，也即SessionId作为key值，用户自身的信息作为value值。与此同时，还会对该key值设置过期时间，默认是30mins。相关代码如下：



4.1.1.3 重置密码（登录前）

如果用户在登录本电商系统的时候忘记了登录密码，那么便可以点击登录窗口下面的“忘记密码”按钮，开始进行密码找回操作。进入到密码找回页面（如图4.4所示），输入用户名，然后点击“下一步”按钮，这个时候找回密码窗口会显示密码提示问题（如图4.5所示），然后根据密码提示问题输入密码提示答案，如果答案输入正确，接着会要求用户输入新的密码，确认输入的密码不小于6位后，点击“修改密码”按钮，等待系统执行密码修改的操作，执行完后页面再次跳转，提示用户密码修改成功（如图4.6所示）。



图4.4 密码找回页面

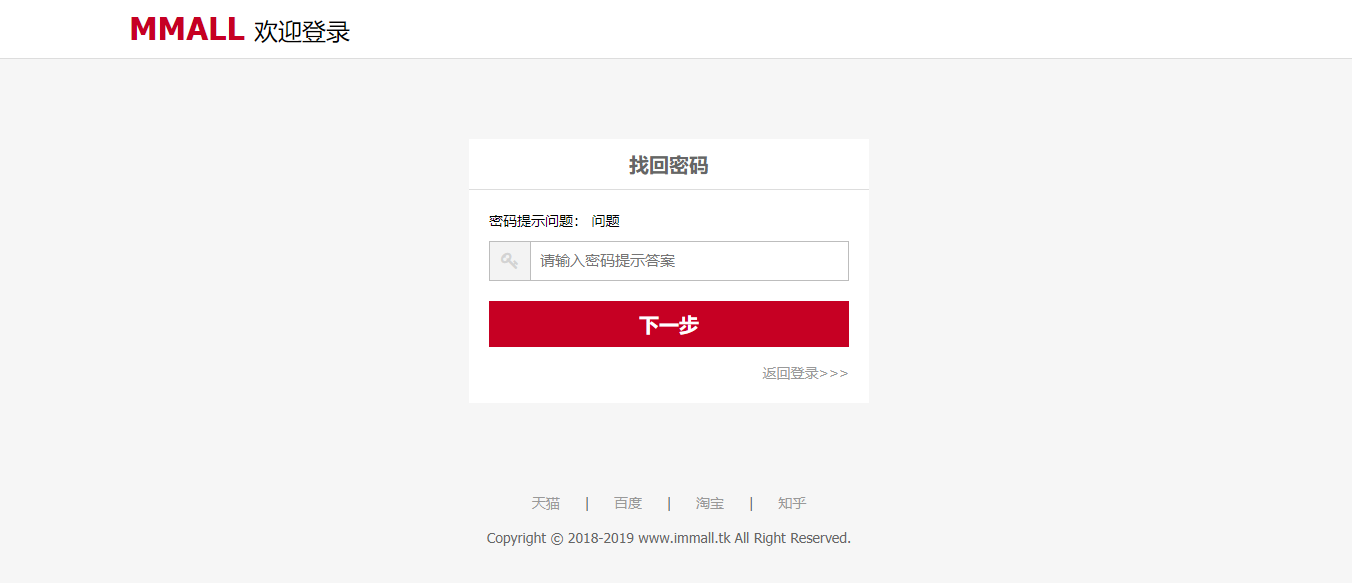
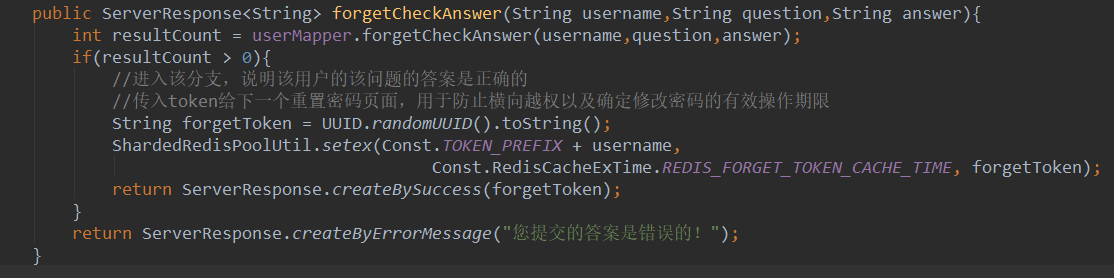


图4.5 输入密码提示答案



图4.6 密码修改成功

用户在修改密码的过程中，系统会生成一个token码，该token码会传入到重置密码页面，用来防止用户横向越权，恶意修改其他用户的密码。该token码是独一无二的，在用户提交新的密码的时候，系统会拿到用户提交过来的token码与Redis中事先存好的token码进行匹配。而在Redis中，我们采用String类型的数据结构，key值为token\_prefix与用户名拼接而成，value值即是token码。如果Redis中存储的token码与前端用户传过来的token码匹配不上，则说明当前用户存在恶意修改其他用户密码的行为，但因为token码匹配不上，因此本次修改密码操作失败。另外，token码也约束用户需要在固定的时间内（默认12h）修改密码，一旦token码过期了，那么用户需要重新走一遍修改密码的流程。以下为核心代码：





该用例的时序图如下图所示：



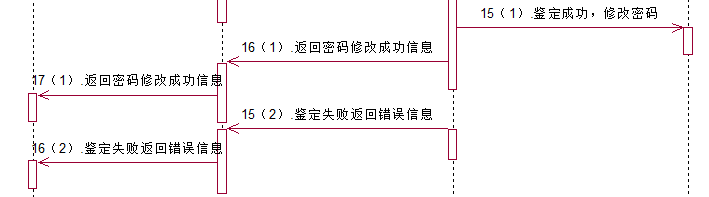


图4.7 重置密码（登录前）时序图

4.1.1.4 重置密码（登录后）

当成功登录进本电商系统后，如果这个时候想修改的自己的登录密码，可以在首页右上角，点击“我的MMALL”，进入到个人信息中心页面（如图4.7所示），然后在左侧，选择“修改密码”选项，然后进入到修改密码页面（如图4.8所示）。接着，在修改密码页面输入旧密码和新密码，然后点击“提交”按钮。等待系统校验信息，如果校验成功，页面会弹出“密码重置成功！”的信息提示，否则会提示相关错误信息：

* 如果旧密码错误，页面提示“旧密码错误，请重试！”；
* 如果新的密码和确认密码不一致。页面提示“两次输入的密码不一致，请重试！”。

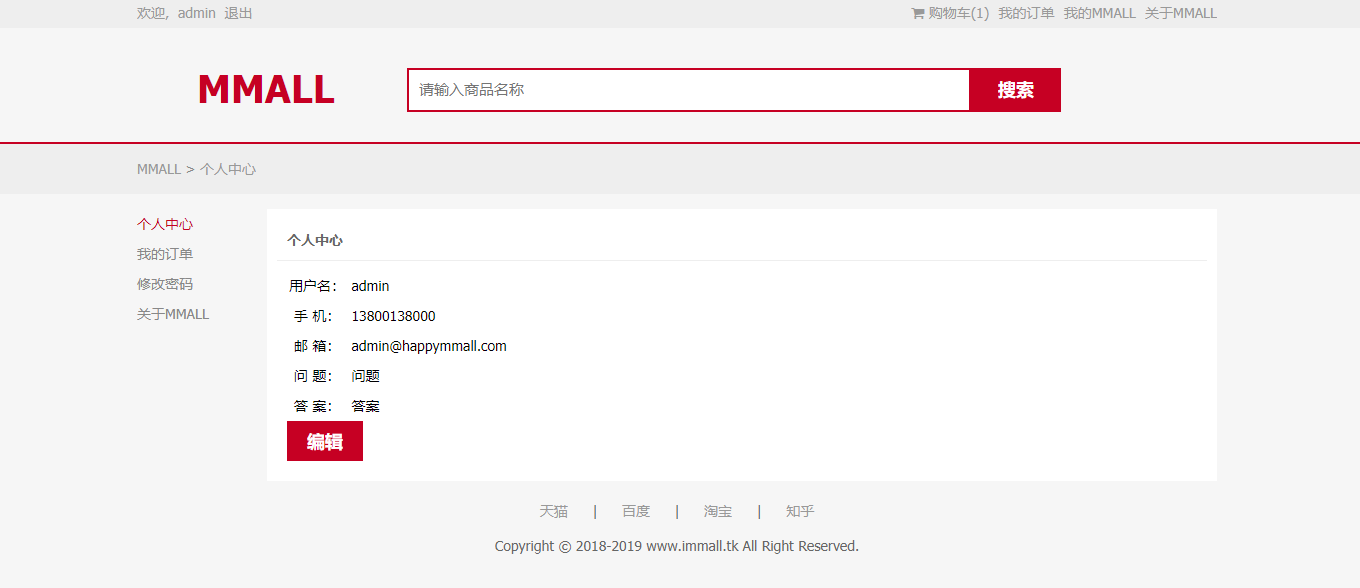


图4.8 个人信息中心页面

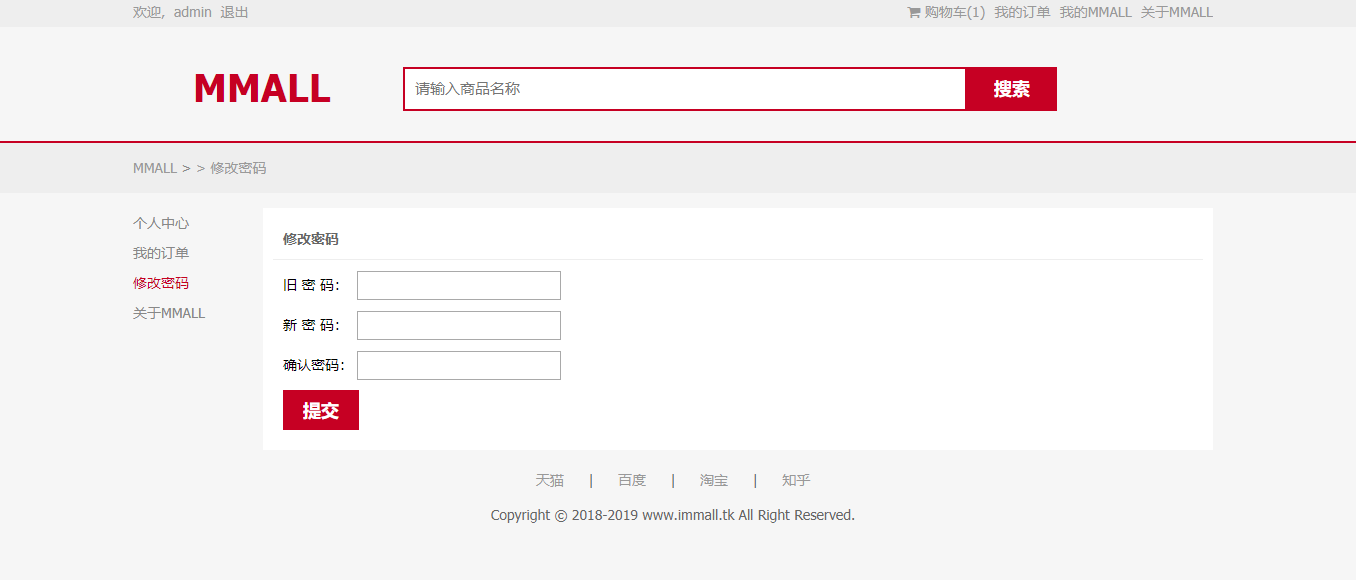


图4.9 密码修改页面

4.1.1.5 获取个人信息

在登录进本电商系统后，用户可以查看自己的个人信息，也即是当初注册时填写的信息。

在商城首页的右上角，点击“我的MMALL”，便可进入到个人中心页面（如图4.9所示）。在该页面内，可以查看到自己当初注册电商系统时所填写的用户名、手机、邮箱、密码提示问题与密码提示答案。

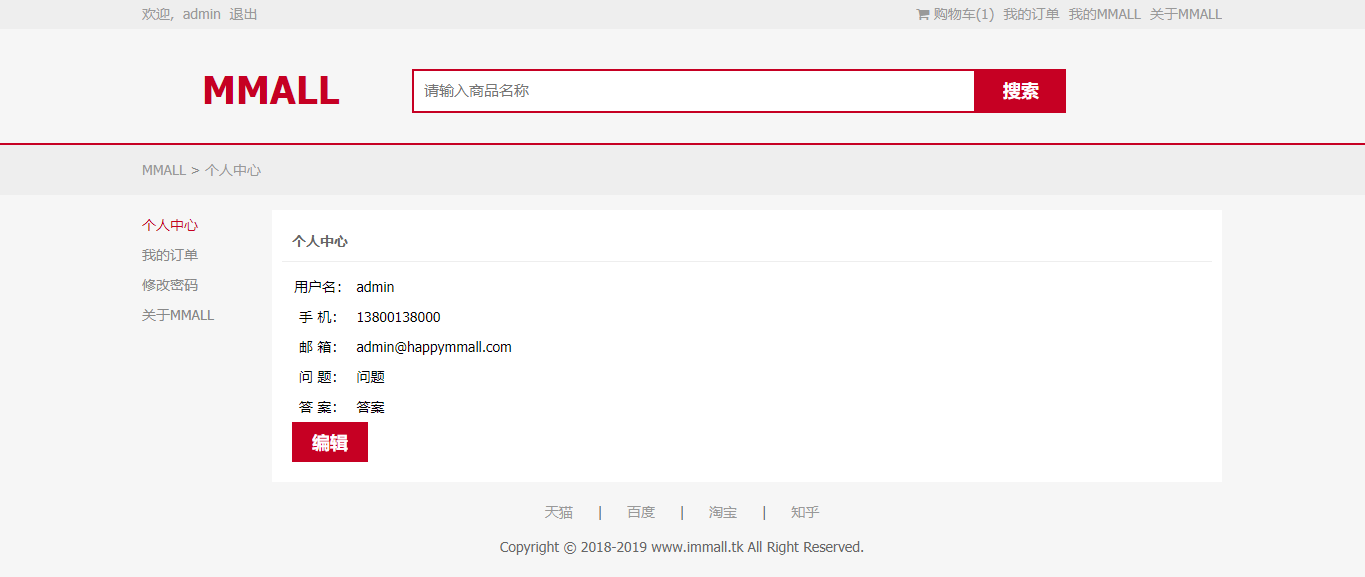


图4.10 个人中心页面

4.1.1.6 修改个人信息

在登录进电商系统后，如果想要修改自己当初注册时填写的信息，在进入个人中心页面后，可以点击个人中心页面上的“编辑”按钮，然后解锁个人信息栏目为可编辑状态（如图4.10所示），接着便可以进行编辑操作。编辑完成后，点击“提交”按钮，页面会对提交内容进行校验，如果校验通过，那么本次编辑操作成功；否则页面上会弹出信息提示框，提示相关的错误信息：

* 如果手机没填或者填的格式不对，那么页面会提示“手机号格式不正确”；
* 如果邮箱没填或者填的格式不对，那么页面会提示“邮箱格式不正确”；
* 如果密码提示问题没有填，那么页面会提示“密码提示问题不能为空”；
* 如果密码提示问题答案没有填，那么页面会提示“密码提示问题答案不能为空”。

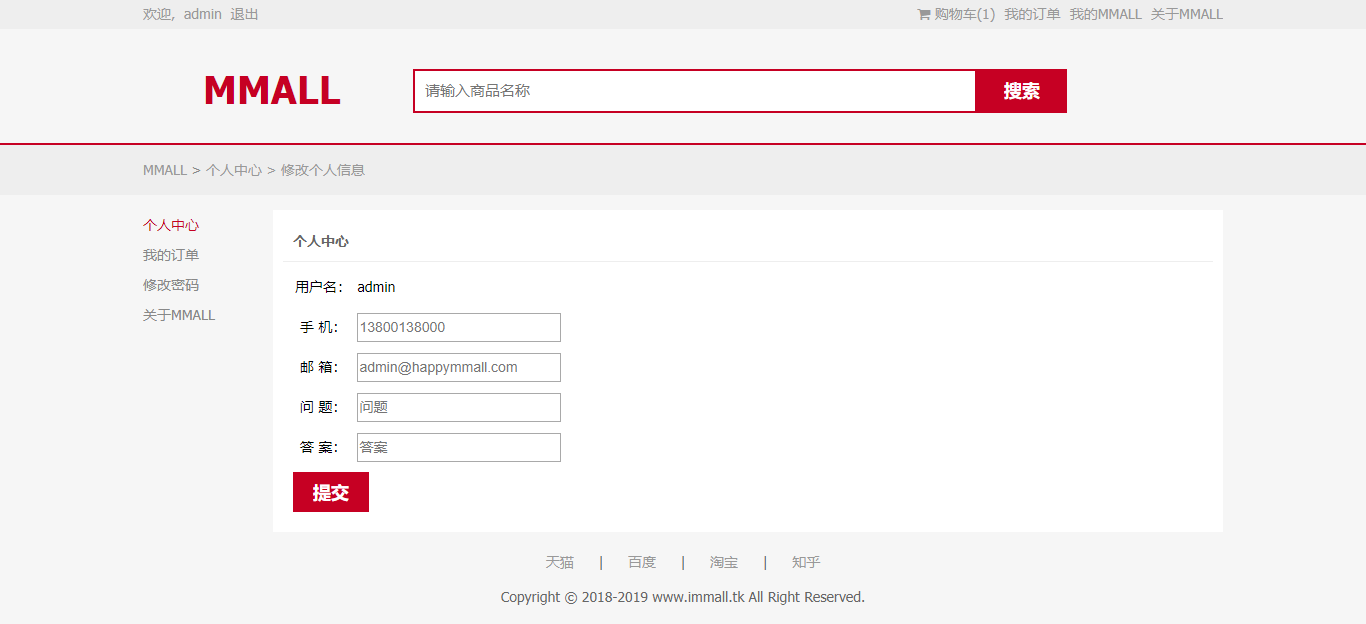


图4.11 编辑个人信息

4.1.1.7 退出登录

在已经登录进电商系统的前提下，用户可以退出当前登录状态。在导航头部的左上角，点击“退出”按钮（如图4.11所示），即可完成退出操作。

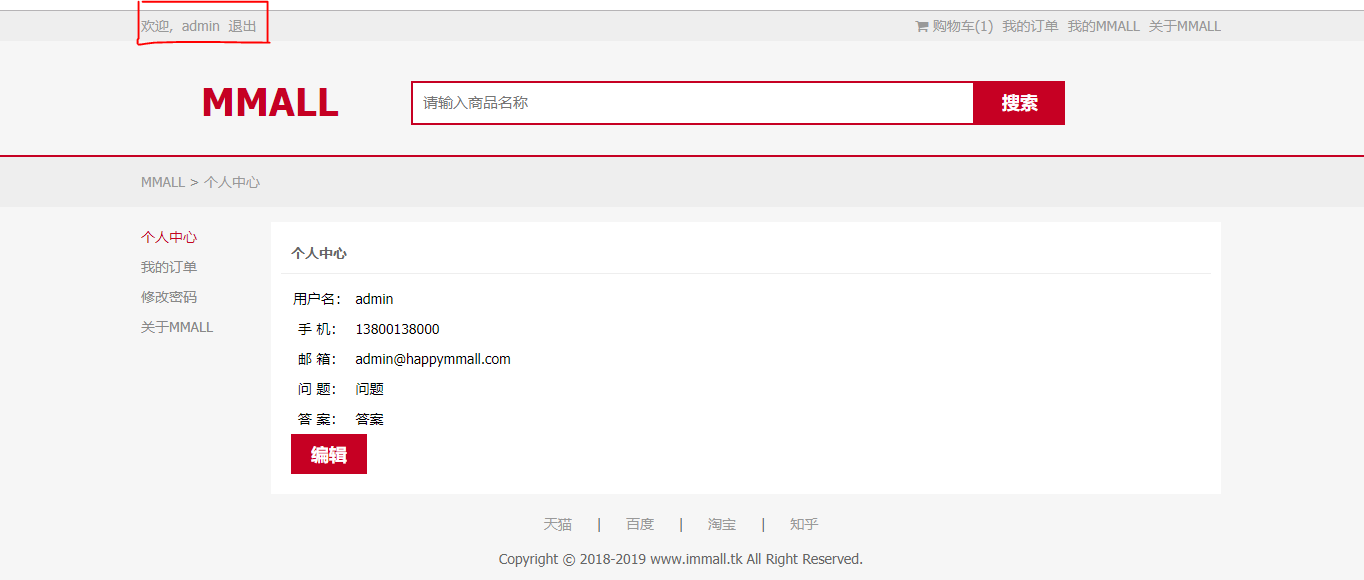
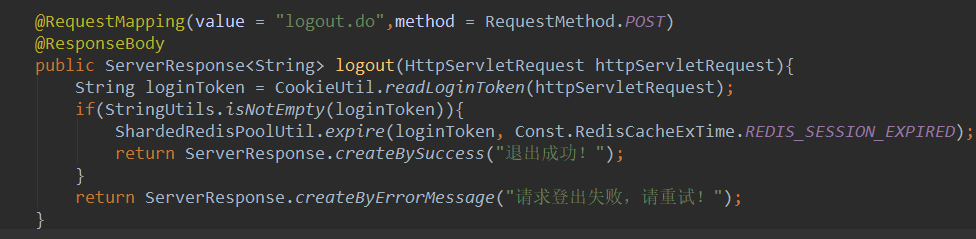


图4.12 退出登录

退出登录时，系统后端也会把该用户对应的存储在Redis里面的相关信息删除，也即是把Redis中的键的存活时长设为-2（过期处置），关键代码如下：



4.1.2 商品模块

4.1.2.1 搜索商品

成功访问到本电商系统的首页后，在首页的搜索框中，可以输入商品的关键字（如图4.12所示），然后点击“搜索”按钮，可以搜索到与关键字相关的商品（如图4.13所示）。如果输入的关键字系统数据库里面没有相关记录，那么页面会提示用户查找不到相关商品的信息（如图4.14所示）。

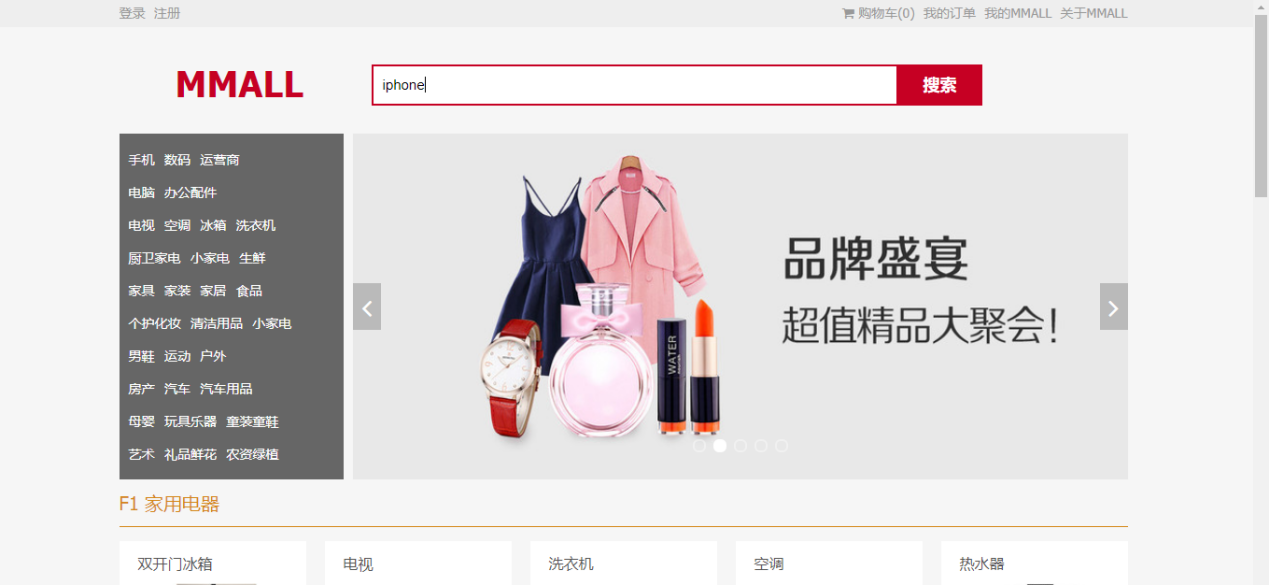


图4.13 在商品搜索框中输入关键字进行搜索

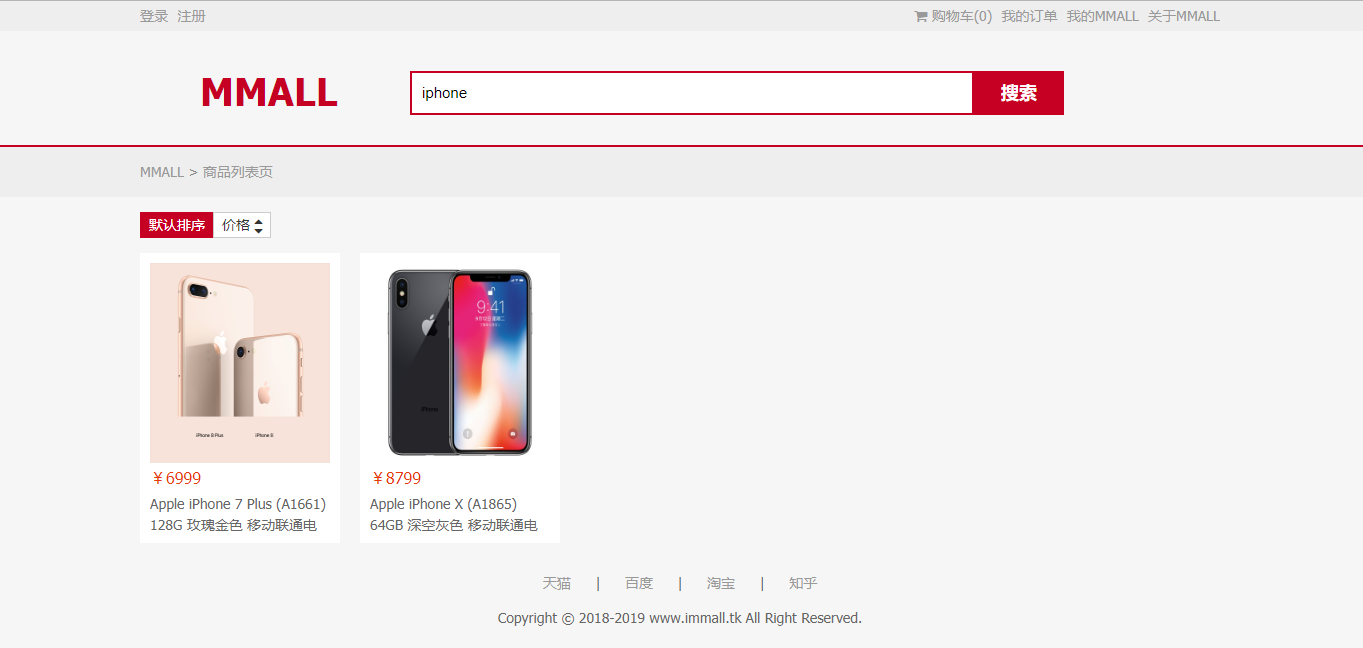
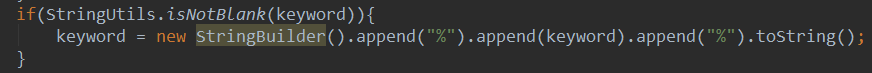


图4.14 搜索到与关键字相关的商品

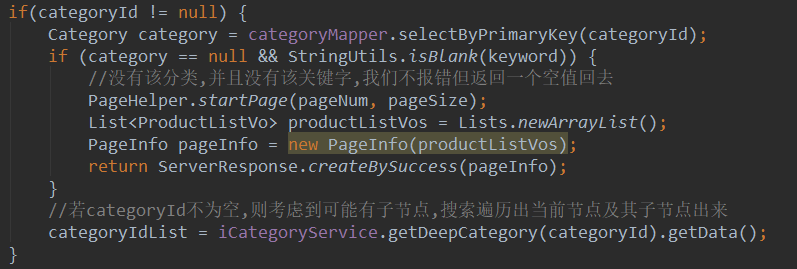


图4.15 系统查找不到与输入关键字相关的商品记录

为了保证搜索出来的商品能够尽可能地覆盖关键字，目前系统后端对于商品的搜索是按模糊方式进行查询的，关键代码如下：



与此同时，在获取商品的时候，如果传入了商品的品类Id，那么我们还得把该品类下面的所有子品类给检索出来，通过递归的方式，不断检索出属于该品类的子品类，关键代码如下：



然后再在系统数据库中查找属于这些品类的所有商品，最后返回给前端，展现出来。

4.1.2.2 动态排序商品列表

搜索出来的商品，默认都是按价格进行升序排序的（如图4.15所示）。当然，有时候为了方便，我们也可以对搜索出来的商品按照价格进行降序排序（如图4.16所示）。目前系统只支持按照价格进行升序或者降序排序。

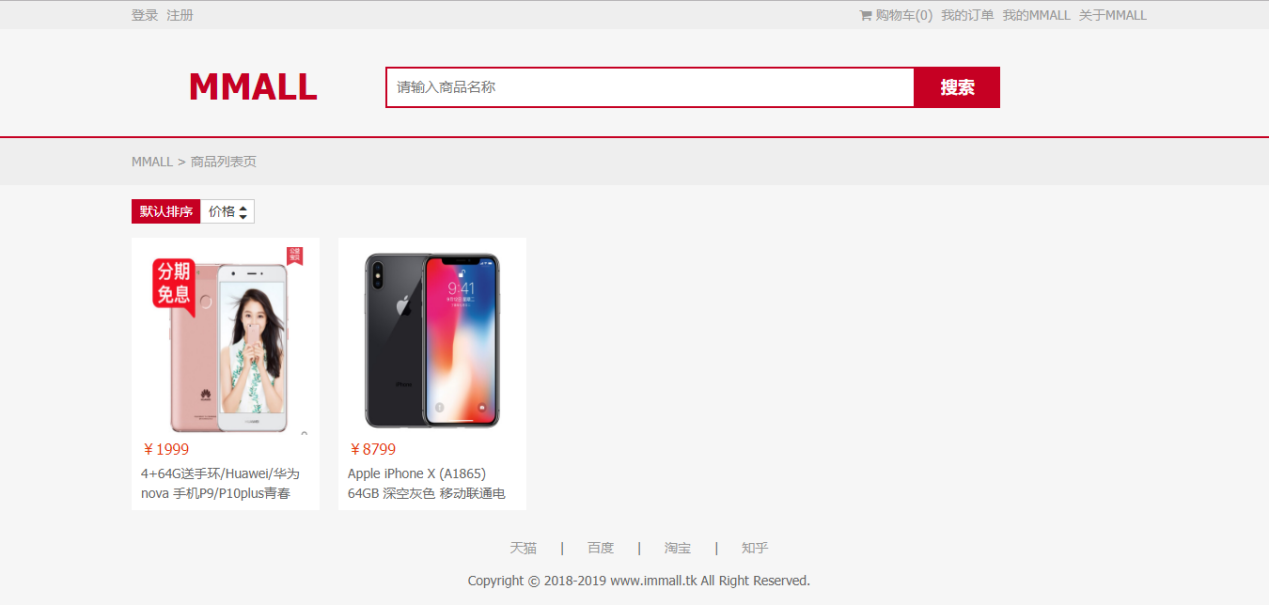


图4.17 搜索出来的商品默认按价格升序排序

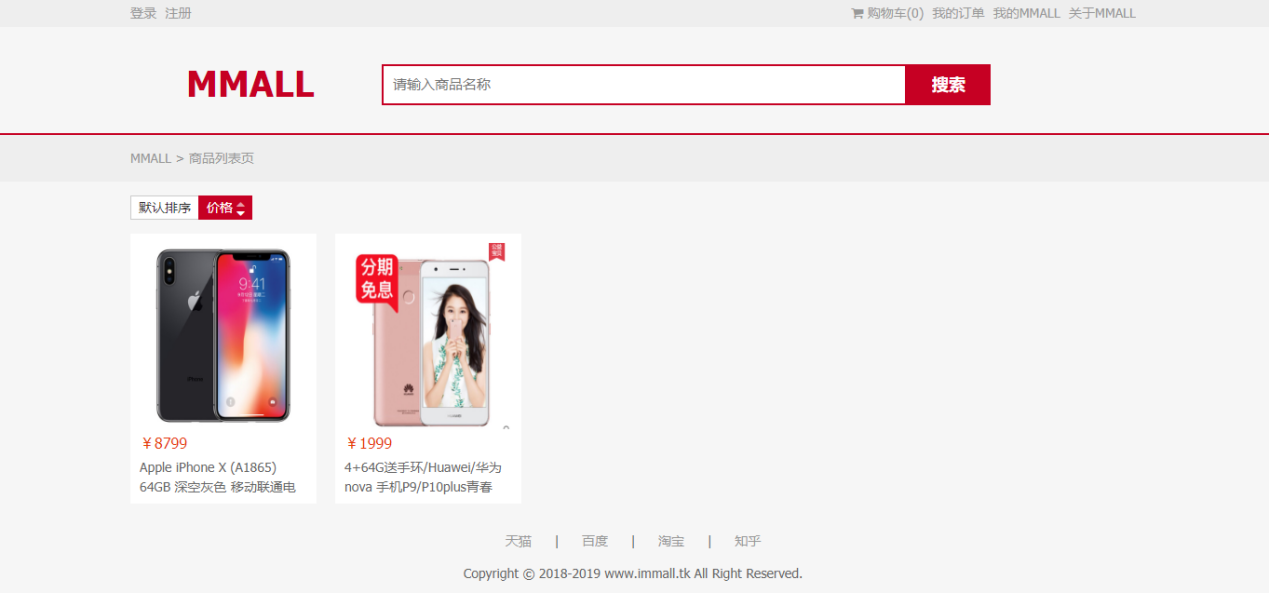


图4.18 按照价格降序的方式排序商品

4.1.2.3 查看商品详情

用户可以在商城中，选择心仪的商品，点击查看其详情内容（如图4.19所示）。

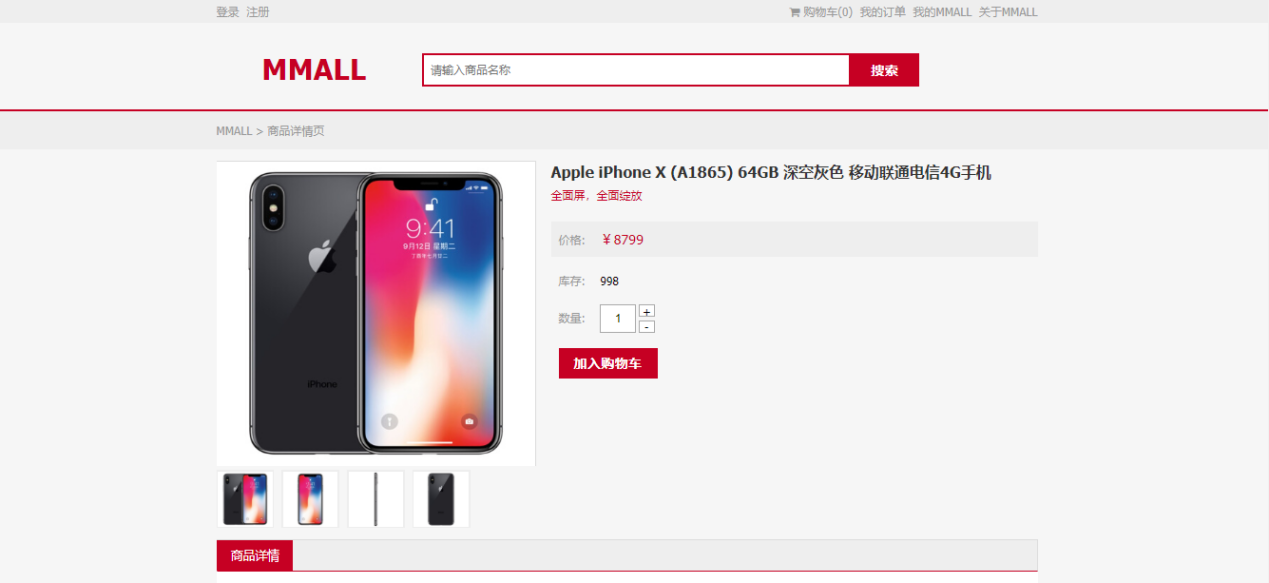


图4.19 查看商品详情

4.1.3 购物车模块

4.1.3.1 添加商品

用户在登录的情况下，如果有心仪的想要购买的商品，可以在该商品的详情页面内，点击“加入购物车”按钮（如图4.20所示），将该商品添加进购物车，然后可以前往购物车，查看先前添加的商品（如图4.21所示）。

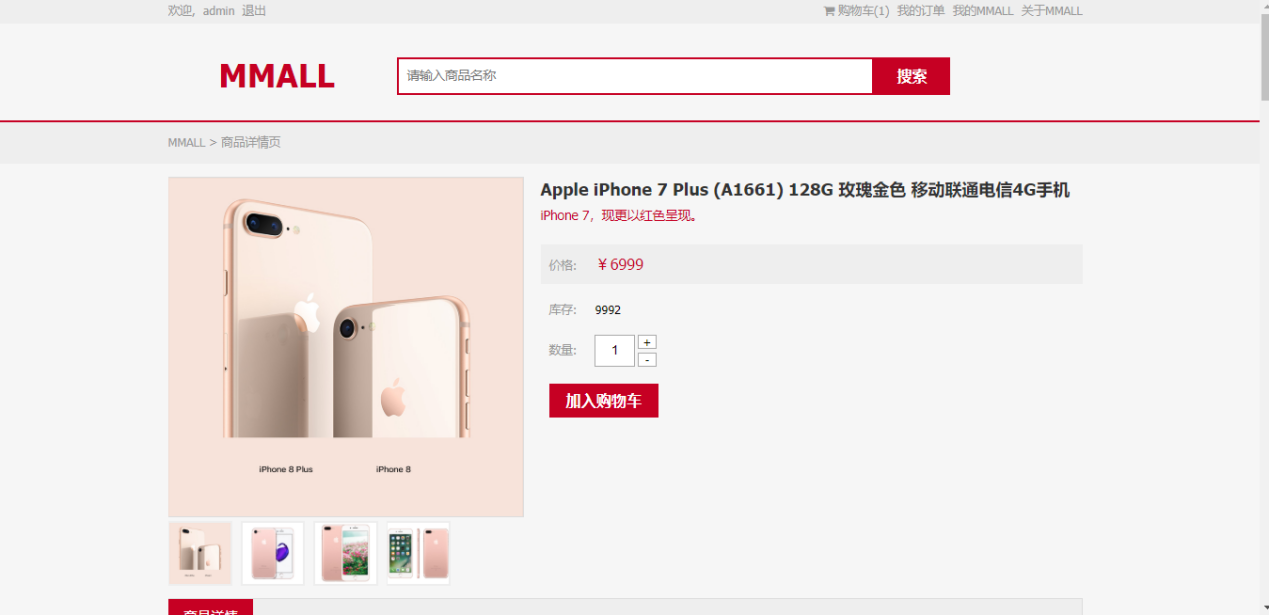


图4.20 在商品详情页面内，点击“加入购物车”按钮



图4.21 在购物车中查看添加的商品

4.1.3.2 修改商品数量

用户可以修改已经添加进购物车中的商品数量。可以对商品数量进行增减操作，数量最少为1（如图4.21所示）。购物车中的商品数量如果超过库存数量，页面会弹出信息提示框，提示用户该商品购买量已经超过该商品的库存数量。

4.1.3.3 移除商品

如果用户对自己选择添加进购物车中的商品不满意，可以去到购物车页面，然后选择移除购物车中的商品（如图4.22所示）。



图4.22 选择移除购物车中的商品

4.1.3.4 列表展示

用户可以查看目前已经添加进购物车中的所有商品。在导航头部的右侧，点击“购物车”图标（如图4.23所示），即可进入购物车商品列表页面（如图4.21所示）。



图4.23 点击进入“我的购物车”商品列表页面

4.1.3.5 选中/取消选中商品

用户在购物车列表页面，可以通过勾选购物车中商品的方式，决定最终是否把某件商品计入结算流程。如图4.21所示，勾选上的商品（默认处于选中状态）会计入总价；否则不会计入总价的范畴（如图4.24所示）。



图4.24 取消选中的商品不会计入总价的范畴

4.1.4 收货地址模块

4.1.4.1 新增收货地址

用户在订单结算页面，可以添加新的收货地址。点击“使用新地址”图标后，页面会弹出一个地址填写的表单（如图4.25所示），用户在表单上面填写相应的收件人姓名、选择所在城市（省市级别）、详细地址等信息，然后点击“保存收货地址”按钮，等待系统校验信息，若信息无误，系统接着执行新增收货地址记录的操作；若信息有误，页面会弹出信息提示框，提示用户相关的错误信息：

* 如果收件人姓名未填，页面弹出提示框“请输入收件人姓名！”；
* 如果所在城市没有选择，页面弹出提示框“请选择所在城市！”；
* 如果详细地址未填，页面弹出提示框“请输入收货人的详细地址！”；
* 如果联系方式未填，页面弹出提示框“请输入收货人的联系方式！”。

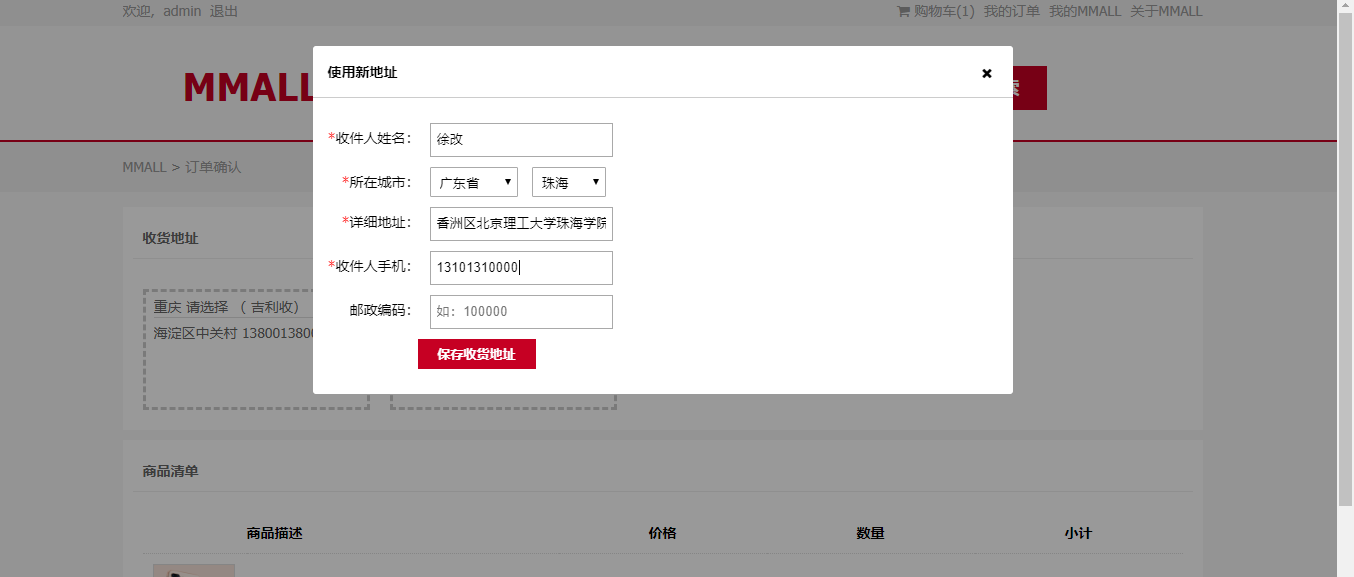


图4.25 地址填写表单

4.1.4.2 编辑收货地址

如果用户对收货地址有需要修改的地方，可以在订单确认页面，选择已经存在的收货地址进行编辑。如图4.26所示，在已经存在的收货地址右下角，点击“编辑”按钮，页面会弹出一张地址编辑的表单（如图4.27所示），用户在该表单上面即可对类如收件人姓名、详细地址等信息进行更改。

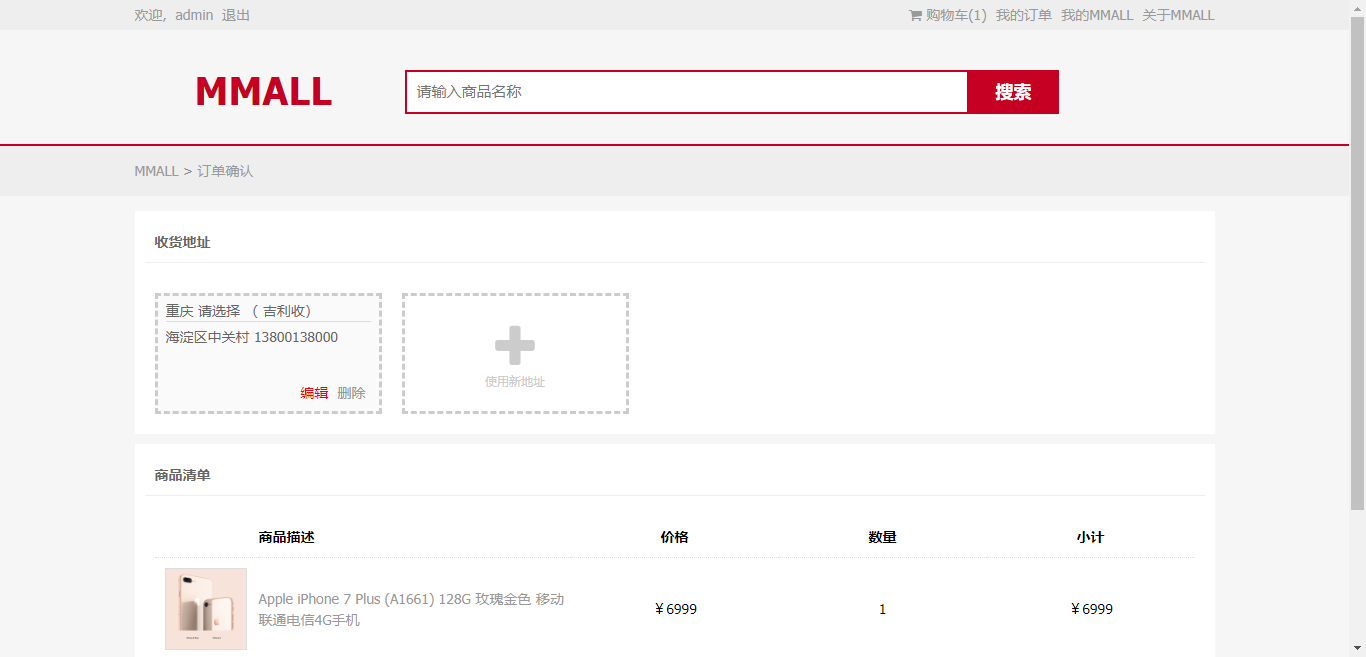


图4.26 选择编辑收货地址

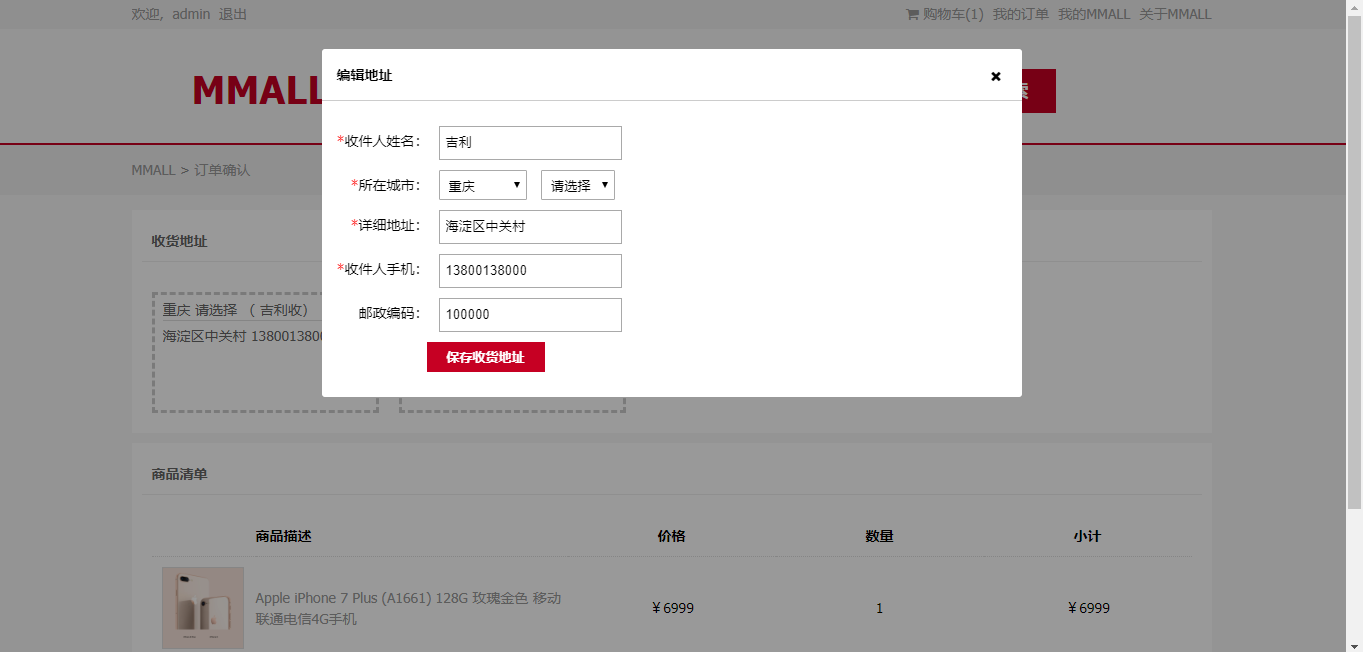


图4.27 编辑收货地址

4.1.4.3 删除收货地址

如果自己的收货地址过多，用户可以选择删除收货地址。在订单确认页面，确认自己想要删除的收货地址记录，在该收货地址的右下角，点击删除按钮（如图4.28所示），页面会弹出信息提示框，询问是否确认删除该条收货地址记录，点击“确认”按钮，即可完成收货地址记录删除的操作。

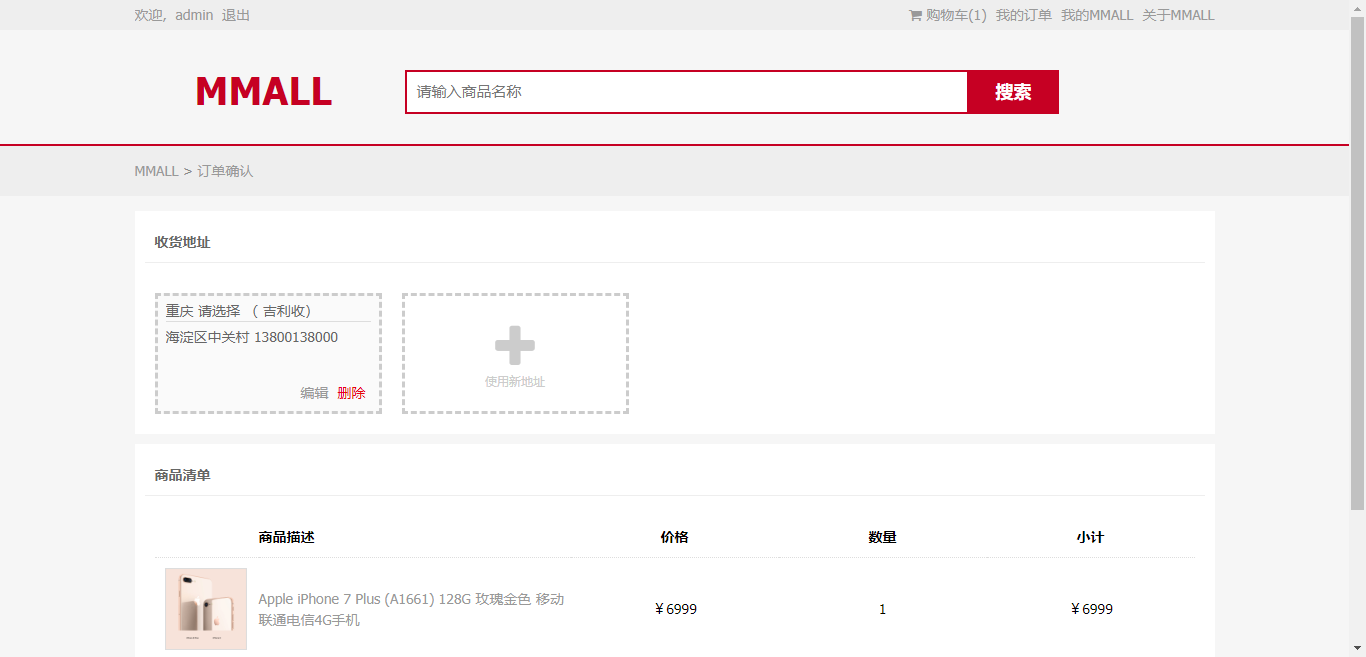


图4.28 删除收货地址记录

4.1.5 订单模块

4.1.5.1 创建订单

当用户在订单确认页面，确认收货地址以及自己选中的商品信息无误后，点击“提交订单”按钮，然后系统会自动生成一张订单。

4.1.5.2 查看订单列表

在登录状态下，用户可以点击导航栏头部中的“我的订单”选项（如图4.29所示），进入个人订单列表页面。系统会以分页的形式，将用户所创建过的所有订单按批次返回回来（如图4.30所示）。

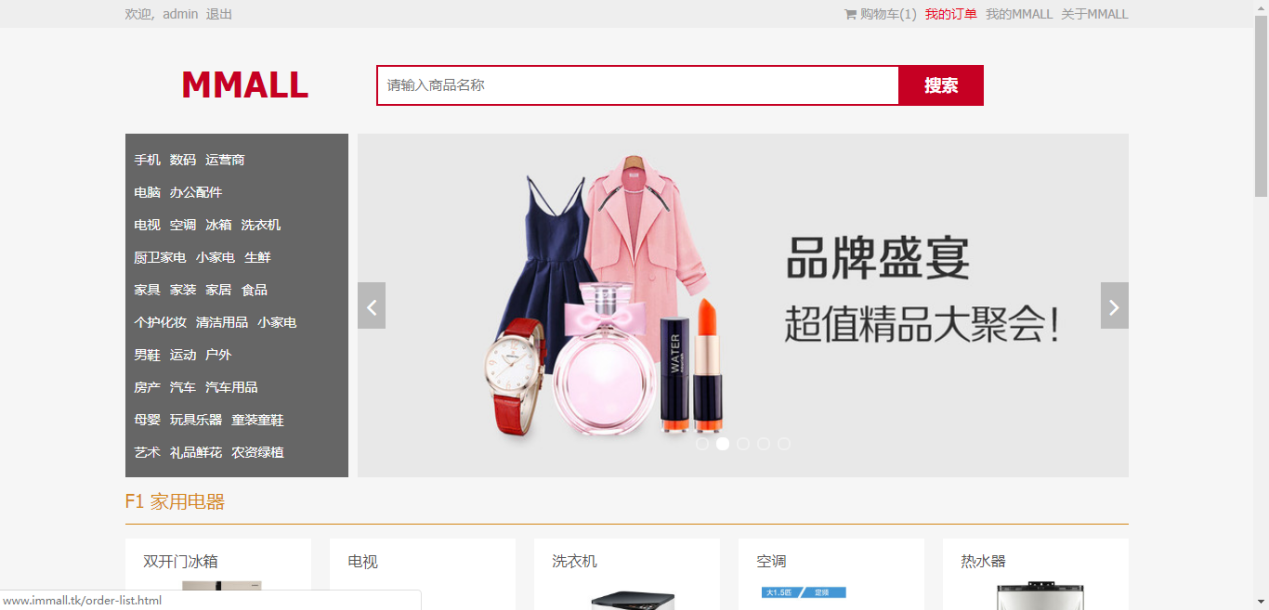


图4.29 点击“我的订单”，进入订单列表页面

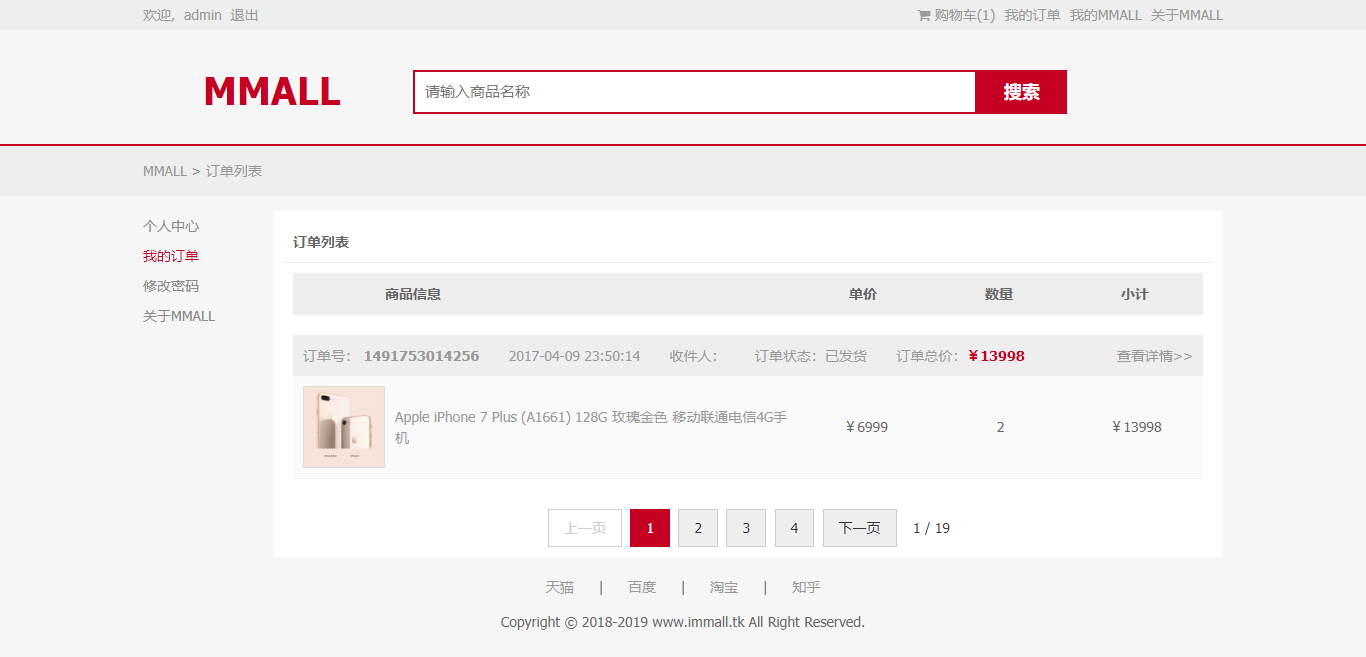


图4.30 查看订单列表

4.1.5.3 查看订单详情

在订单列表页面，用户可以选择查看某笔订单的详情内容。如图4.31所示，在订单列表页面中，选择某笔订单，然后点击“查看详情”按钮，这时页面便会跳转到该笔订单的详情页面（如图4.32所示）。

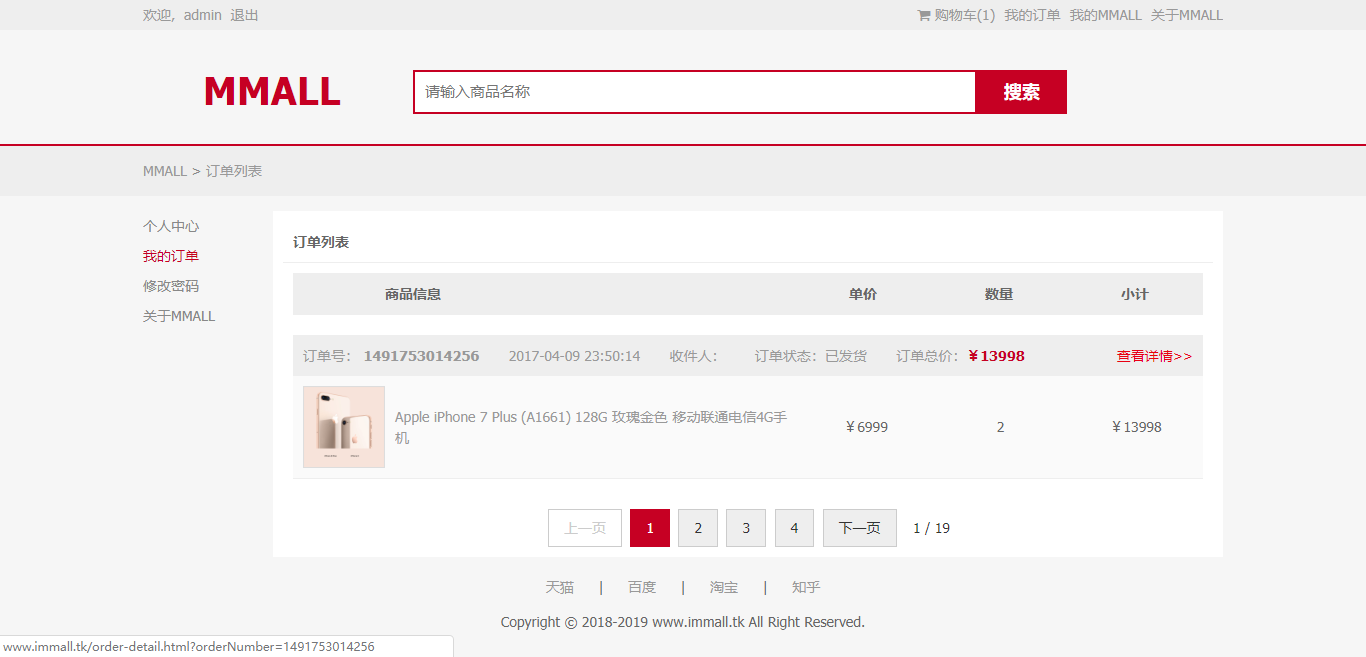


图4.31 点击“查看详情”按钮，查看订单的详情内容

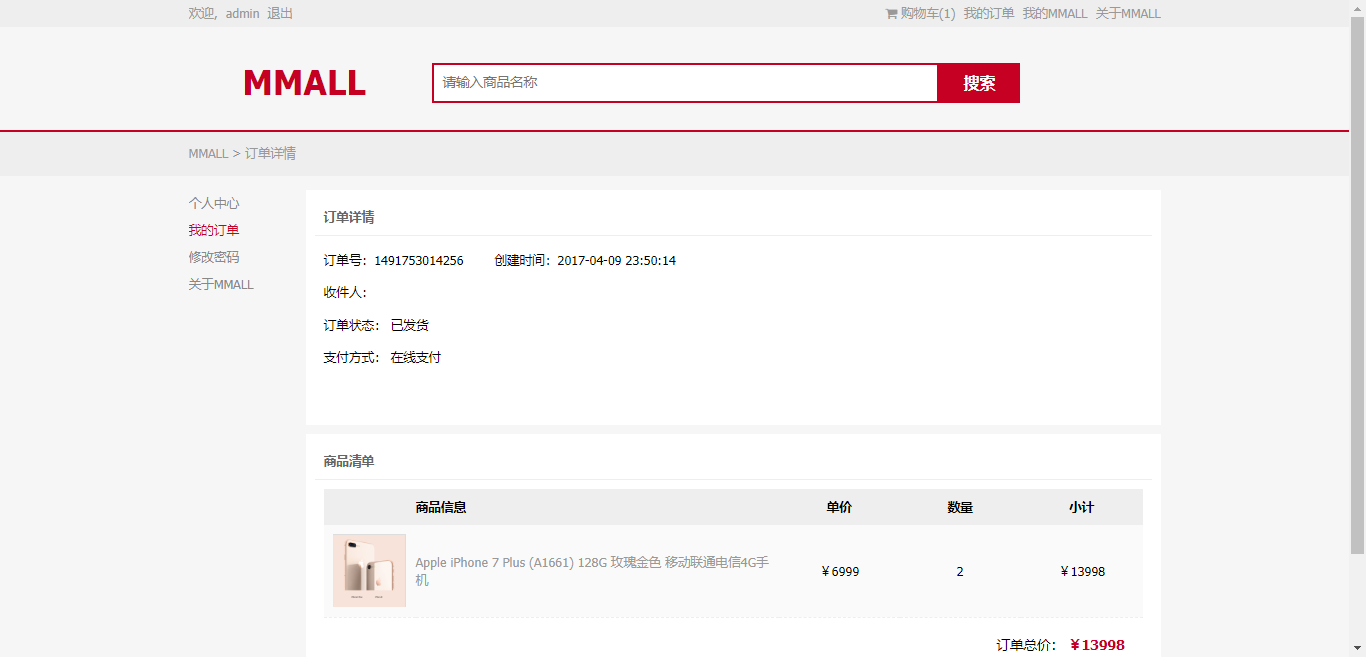


图4.32 查看某笔订单的详情内容

4.1.5.4 取消订单

在创建出来一张订单后，用户可以无理由选择取消支付该笔订单。首先进入订单列表页面，然后选择仍未支付的订单，查看该笔订单的详情（如图4.33所示），然后在订单详情中，点击“取消订单”按钮（如图4.34所示），这时页面会弹出一个信息提示框，询问用户是否确认取消支付该笔订单，点击“确认”按钮（如图4.35所示），便可完成取消订单操作。

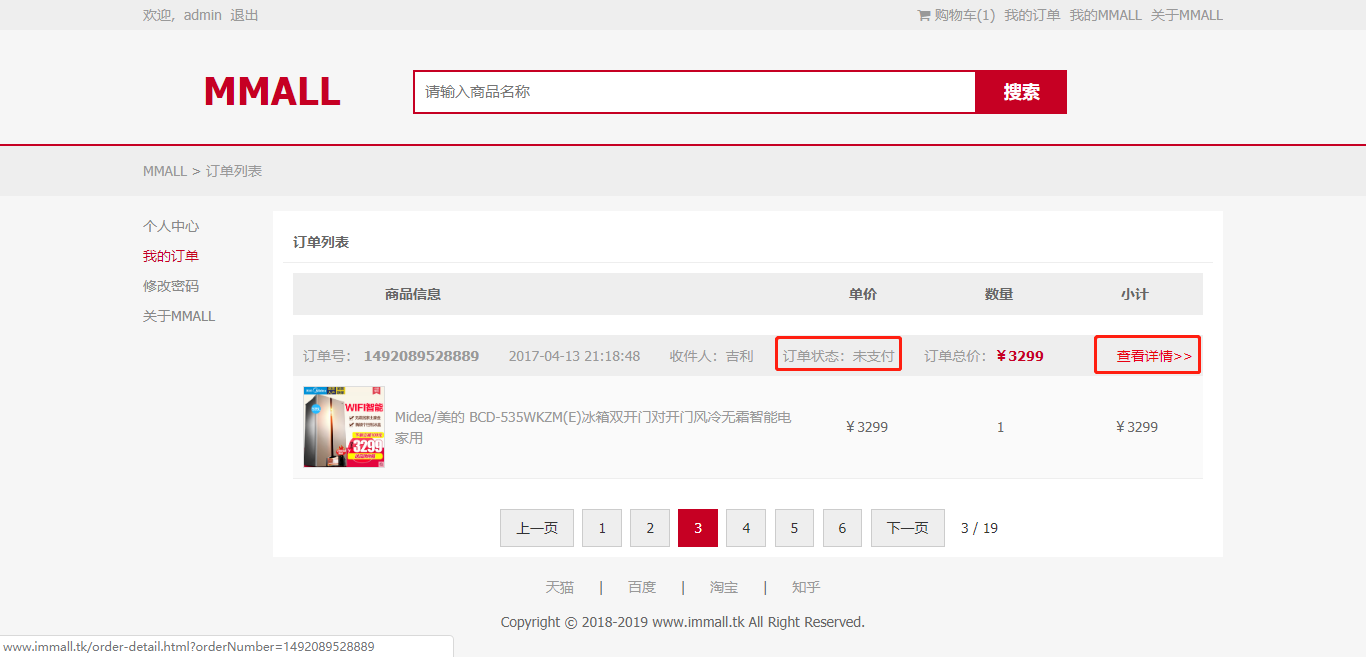


图4.33 进入未支付订单的详情页面



图4.34 点击取消支付此次订单

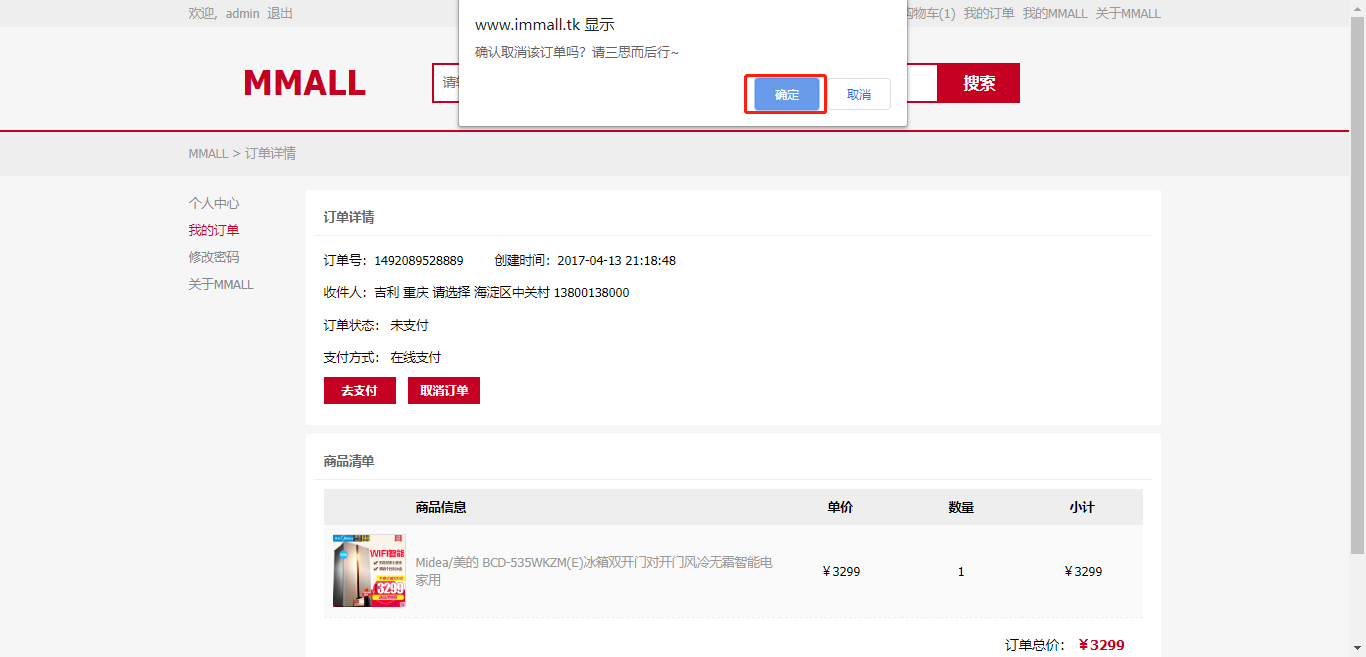


图4.35 确认取消该笔订单

4.1.6 支付模块

4.1.6.1 支付订单

在订单确认页面，确认收货地址和所购商品信息无误后，点击“提交订单”按钮（如图4.36所示），便进入订单支付页面。在订单支付页面，会有一个支付二维码显示出来（如图4.37所示），使用支付宝手机扫码，便可完成这一次的订单支付操作（如图4.38所示）。如果中途用户退出，那么如若该笔订单超过30分钟仍未支付的话，系统会自动进行关单处理。

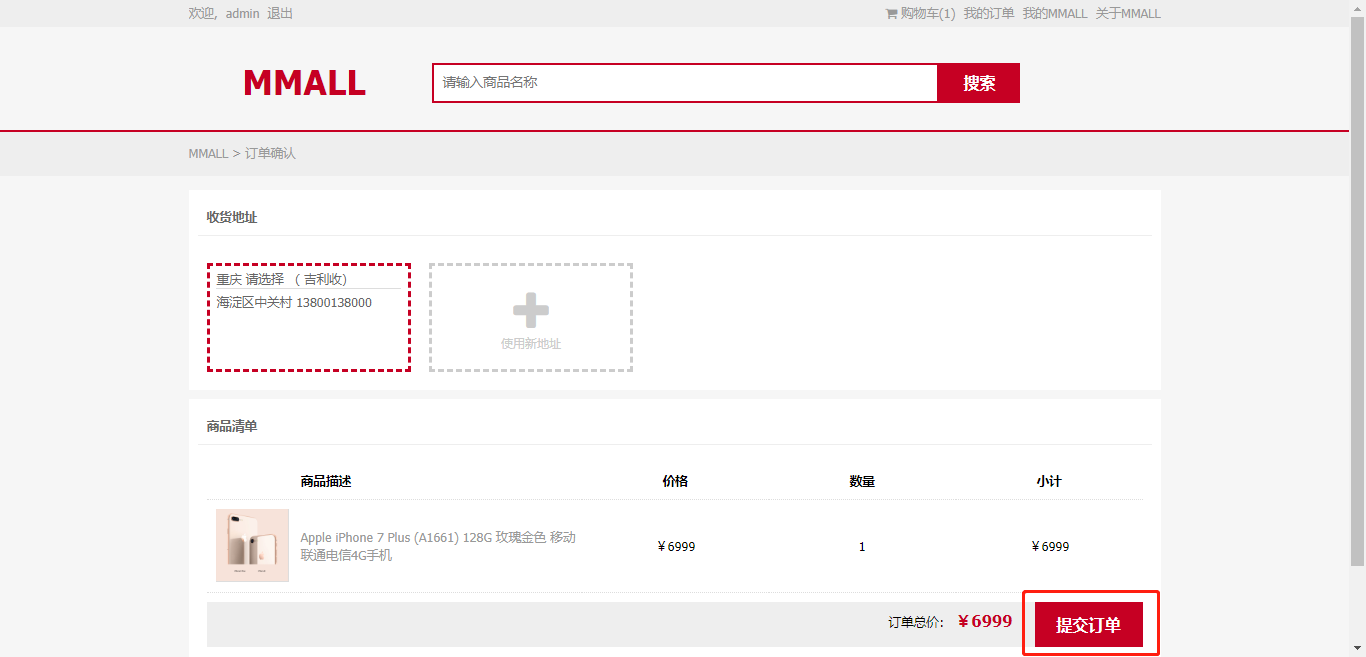


图4.36 生成订单，准备支付订单

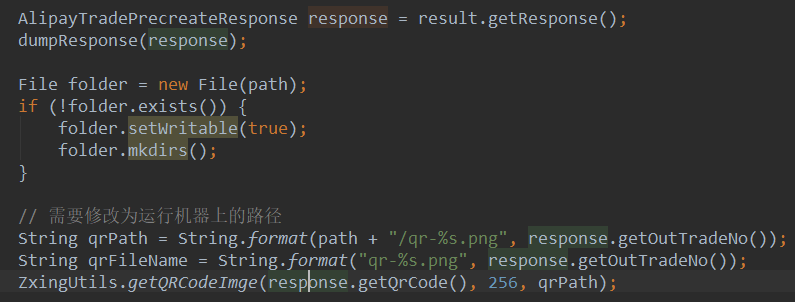


图4.37 支付宝扫码，支付订单

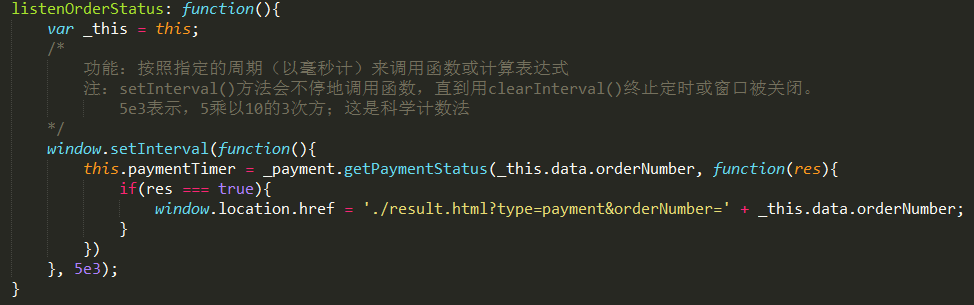


图4.38 订单支付成功

本系统中的订单支付，对接了支付宝-当面付的服务。在创建订单之后，系统后端会请求支付宝那边声明好的接口，然后获取支付宝那边生成的二维码，关键代码如下所示：

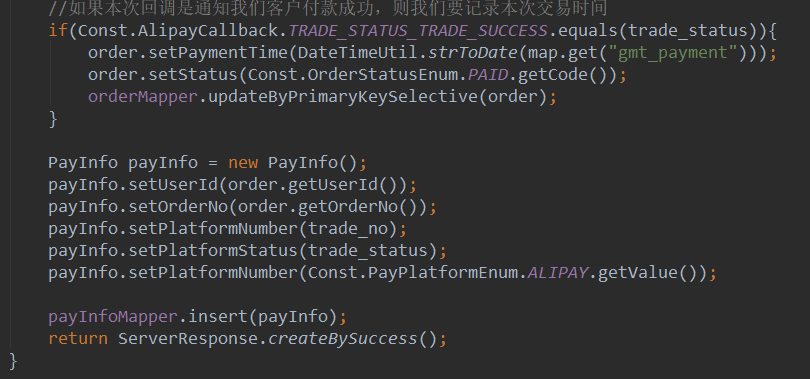


接着前端把该二维码显示出来，然后前端的JS代码每隔5s查询该笔订单是否已经支付成功，如果支付成功则进行页面跳转，关键代码如下所示：

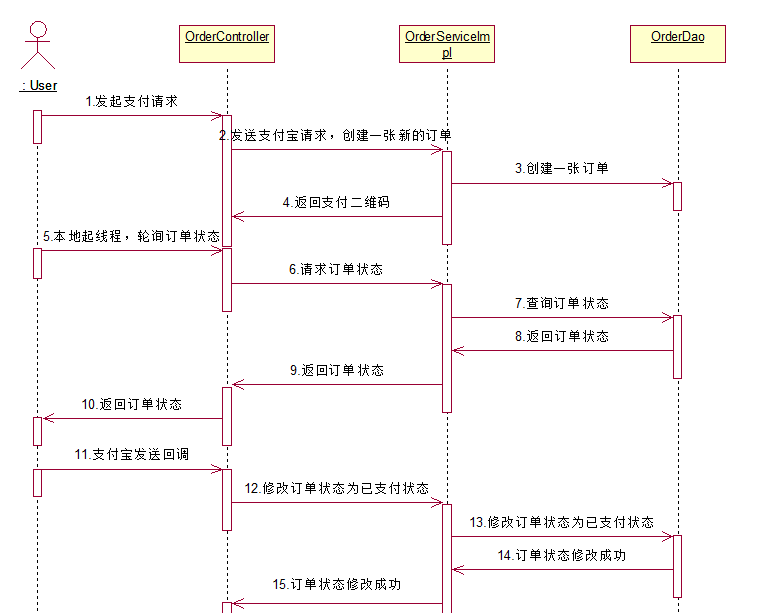


用户在扫码成功付款后，支付宝那边会有一个回调，用来告知我们系统用户已经支付成功，然后我们系统需要修改订单状态为支付成功状态，关键代码如下：





该用例时序图如下图所示：



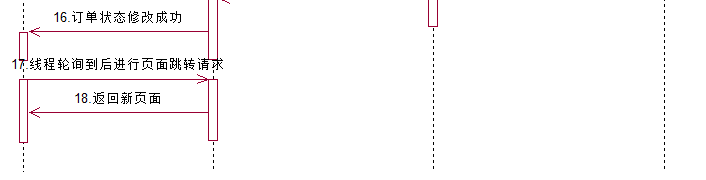


图4.39 支付订单时序图

4.2 管理员端

4.2.1 用户模块

4.2.1.1 浏览用户列表

管理员登录进电商系统管理员端后，在管理端首页，由于前端使用的是React框架，而首页作为home，对于React来说我们把它视为一个组件，React的组件都有一个生命周期，我们在该组件构造出来的时候，调用loadCount()方法，自动异步请求后端的base\_count.do接口，关键代码如下：



该接口会通过MyBatis这一ORM框架，从数据库里面查找用户表、商品表和订单表这三张表的记录总数，然后将这三张表的记录总数返回给前端，由于这个时候是home组件中state的属性发生了改变，因此会触发React组件中的Render()方法，然后对页面上的用户总数、商品总数和订单总数的数值进行渲染。

点击用户总数（如图4.39所示），这时前端会通过Ajax异步请求的方式，调用后端用户模块的list.do接口，该接口通过ORM框架在数据库中查找出用户表的所有记录，然后将其返回给前端。这样我们便可查看到当前电商系统中已注册的所有用户的信息（如图4.40所示）。



图4.40 管理员端首页

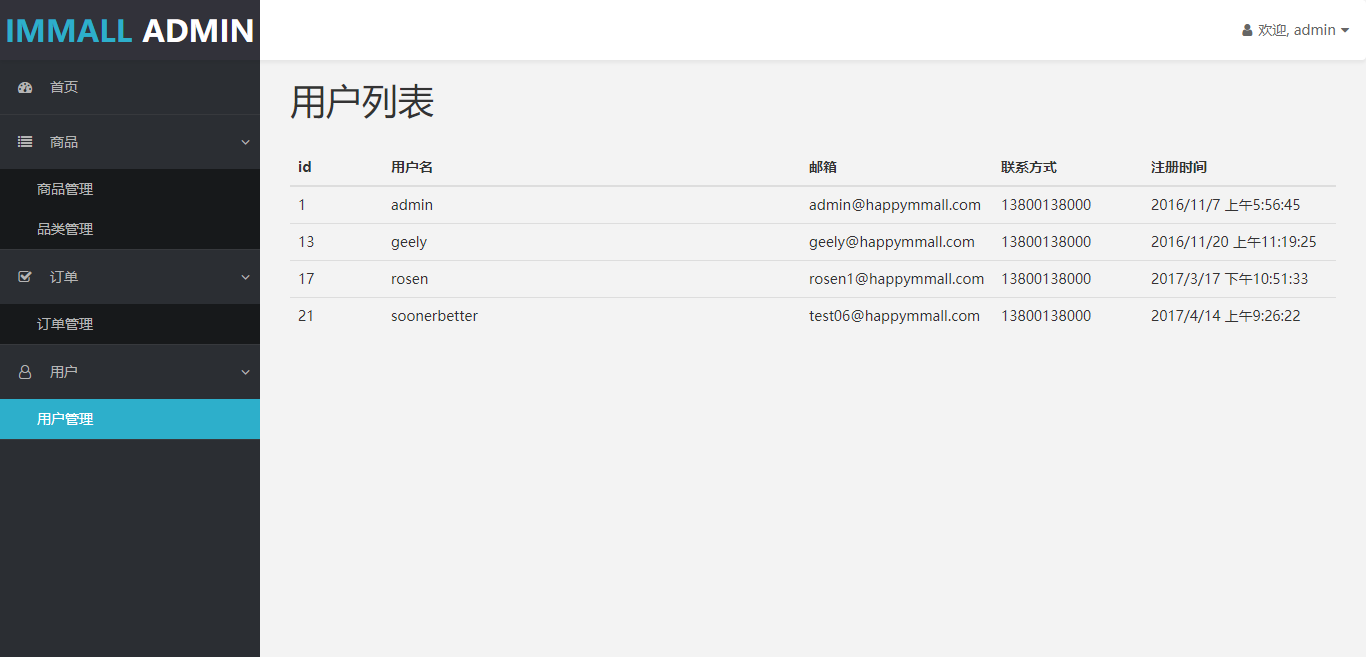
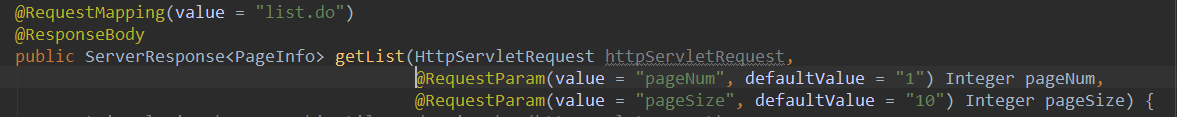


图4.41 查看用户列表

4.2.2 商品模块

4.2.2.1 浏览商品列表

在管理员端首页，左侧导航栏中，点击“商品管理”菜单，这时页面会发生Ajax请求，请求后端商品模块的list.do接口，由于该接口涉及到分页，前端在默认的情况下，会将pageNum=1传递给后端，后端接收到pageNum参数，发现没有传pageSize参数，因此会默认将pageSize赋值为10，关键代码如下：



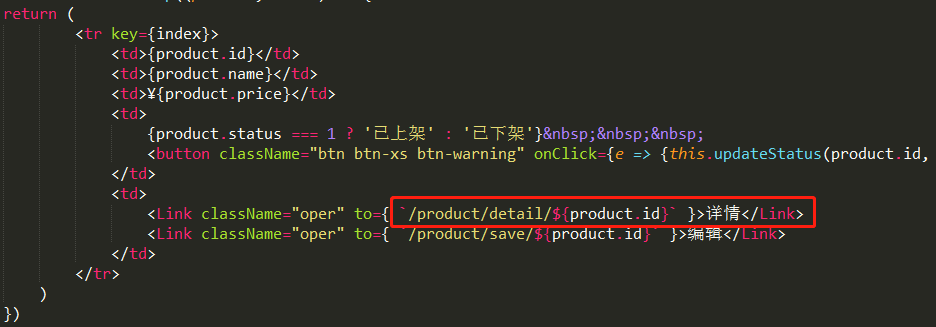
然后再通过框架MyBatis的pageHelper，对从数据库里查询出来的所有商品记录按批次返回回去，接着我们便可在前端查看到迄今为止商城中存在过的所有商品（包括已上架和已下架）（如图4.41所示）。



图4.42 浏览商品列表

4.2.2.2 查看商品详情

在商品管理列表页面，管理员可以查看到迄今为止商城中出现过的所有商品，包括上架和已下架。除此之外，管理员还可以查看这些商品的详情信息。选择某一件商品，然后点击“详情”按钮，这时页面会通过Ajax发起请求，请求后端商品模块的detail.do接口。由于在商品列表渲染的时候，是通过之前请求商品模块的list.do接口而获得的每一件商品信息，因此我们在“详情”按钮这一url上，拼接了每一个商品的productId，关键代码如下：



作为参数传递给后端，后端依照productId在数据库中查询该Id对应的商品记录，然后将其返回。接着前端便是把返回结果回填到商品详情中的表单里面，于是我们便可以看到如图4.42所示的商品详情。



图4.43 查看商品详情

4.2.2.3 编辑商品信息

在商品管理列表页面，管理员可以对商品进行编辑操作。选择一款商品，然后点击“编辑”按钮，与查看详情的处理逻辑类似，先是根据商品的productId请求后端商品模块的detail.do接口，获取到该件商品的详情信息，然后回填到商品编辑页面。在商品编辑页面，可对商品名称、商品描述等信息进行修改，同时还可重新上传商品的照片。上传商品照片，前端通过一个名为react-fileupload的插件完成；而上传富文本则是通过一个名为simditor的插件完成。信息填写好后，点击“提交”按钮（如图4.43所示），这时前端请求后端商品模块的save.do接口，后端通过ORM框架MyBatis对商品信息进行更改。



图4.44 编辑商品信息

4.2.2.4 添加商品

如果在商城上线期间，有新的商品需要上架销售，这时管理员可以执行添加商品操作。在商品管理列表页面，在右上角点击“添加商品”操作（如图4.44所示），即可进入商品添加页面。



图4.45 进入商品添加页面

在商品添加页面，输入商品名称、商品描述，为商品选择所属分类，然后添加图片等富文本内容，这些操作做完后，点击“提交”按钮（如图4.45所示），



图4.46 商品添加

这时前端首先会对填写的商品信息进行校验，如果有非法的，则页面会弹出信息提示框，提示相关信息，校验的关键代码如下：



如果商品信息校验成功，这时前端会发送Ajax请求，调用系统后端product模块的save.do接口，后端代码通过前端传递的product对象是否携带有Id，来判断本次save请求是新增操作还是修改指定的商品信息操作，然后再通过ORM框架，执行商品表的create操作或者update操作。

4.2.2.5 修改商品上下架状态

商城中的在售商品有时候会因为库存等原因需要下架，管理员可以在商品管理列表页面，修改商品的销售状态，达到商品下架的目的。如图4.46所示：



图4.47 下架商品

在商品栏中，点击“下架”按钮，这时页面会弹出询问窗口，询问管理员是否确认下架该款商品。当点击“确定”的时候，前端便会发送Ajax请求，请求到后端商品模块的set\_sale\_status.do接口，后端拿着前端传递过来的productId与status，对指定的商品的销售状态修改为已下架状态。这时，如果刷新商城列表页面，会发现无法查询到先前已下架的商品，因为商城列表页面请求的是那些在售状态的商品。另外，对已下架商品进行上架处理也是同样的操作与实现过程，这里不再过多赘述。

4.2.2.6 搜索商品

在百花缭乱的商品列表记录中，管理员可以根据商品名称或者商品Id，查找指定的商品记录出来。在商品列表页面，选择“按商品ID查询”，输入商品ID，然后点击“搜索”按钮，系统后端接收到前端传来的参数，通过ORM框架MyBatis，对数据库使用精确匹配的方式，查找出相关的商品记录出来，这样我们便可得到系统响应返回回来的商品记录（如图4.47所示）；

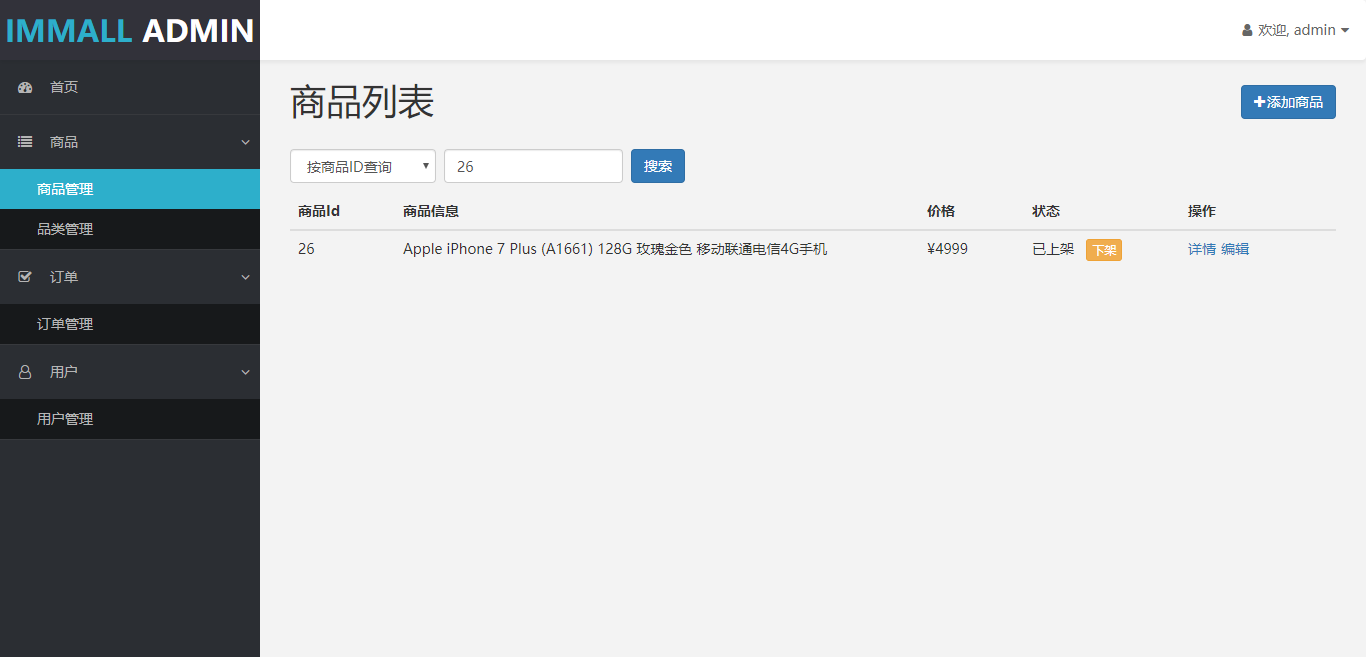
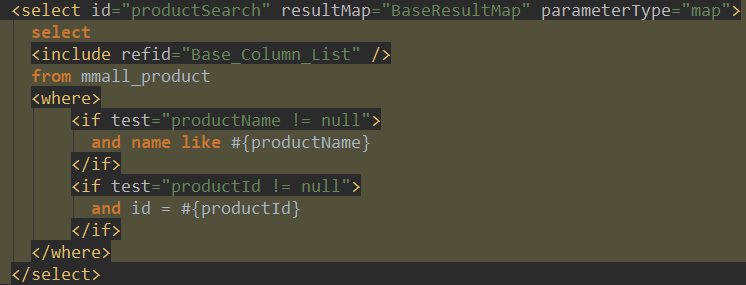


图4.48 按照商品ID查找指定的商品

如果是按商品名称进行查询，那么后端是通过模糊匹配的方式进行查找。另外，如果管理员在没有输入任何内容的情况下点击了“搜索”按钮，那么系统默认是对商品表进行全表查询。关键代码如下：



4.2.3 订单模块

4.2.3.1 浏览订单列表

管理员可以查看商城迄今为止所创建过的所有订单，包括已付款和未付款的。在管理员端首页左侧的导航栏里，点击“订单管理”按钮，这时页面右侧会显示出订单列表（如图4.48所示）。

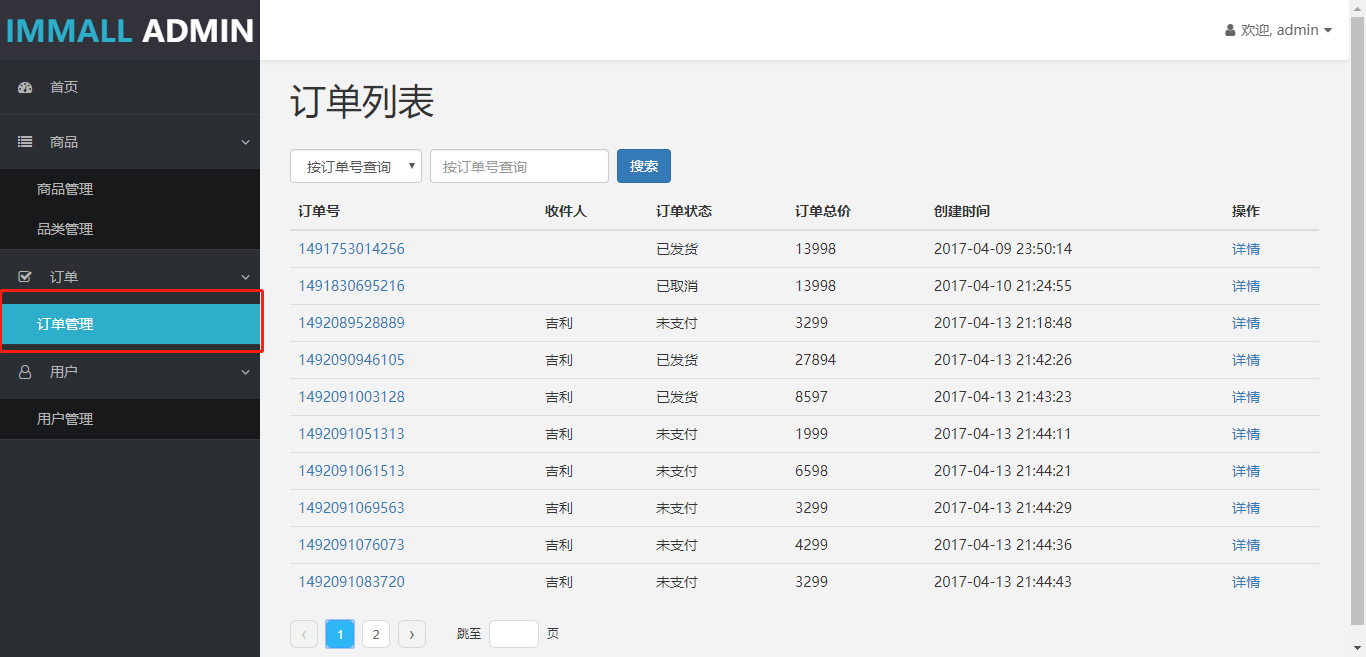
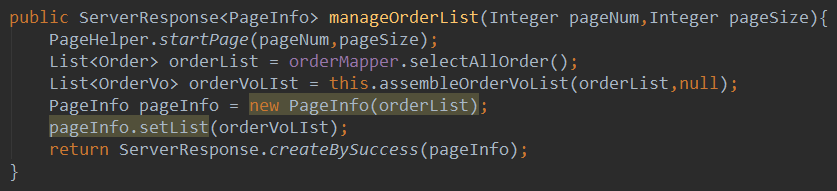


图4.49 订单列表

在点击“订单管理”按钮的时候，前端便会发起Ajax请求，同时携带参数pageNum=1，传递给后端订单模块的list.do接口，该接口会接收pageNum与pageSize两个参数。此时后端代码逻辑发现前端没有传递pageSize参数，因此会默认把pageSize参数的值赋值为10，然后通过ORM框架MyBatis的pageHelper分页插件，把从数据库里面查询出来的所有订单记录按照批次返回给前端。关键代码如下：



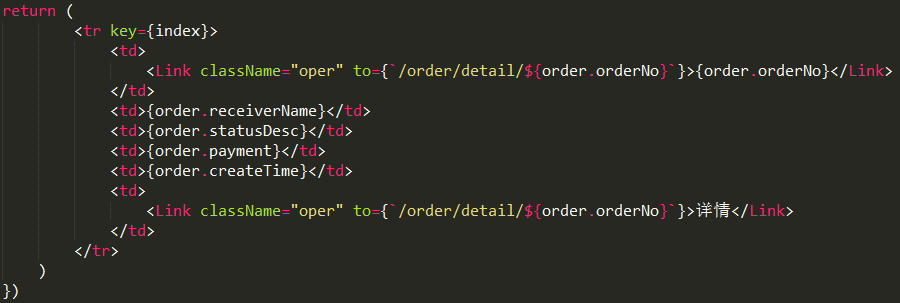
4.2.3.2 查看订单详情

在订单管理列表页面，管理员可以查看某笔订单的详情信息。选择需要查看的订单，点击“详情”按钮，便能跳转到订单详情页面（如图4.49所示）。



图4.50 查看订单详情

在点击“详情”按钮的时候，前端将发起Ajax请求，将指定的orderNo作为请求参数，传递给后端订单模块的detail接口。前端在渲染订单列表的时候，便根据查询出来的每笔订单，将每笔订单号作为参数拼接到请求详情内容时的url字符串后面，关键代码如下：



后端获取到orderNo参数后，便根据orderNo在数据库中查找出指定的订单记录出来，然后将其返回给前端。前端获取到返回回来的订单记录后，回填到页面上。

4.2.3.3 修改订单状态

对于那些已经成功付款的订单，管理员可以选择将订单状态从已付款修改为已发货。选择一笔已经支付过的订单（如图4.50所示），进入该笔订单的详情页面，然后点击“立即发货”按钮，这时页面会弹出一个信息提示框，询问管理员是否确认该笔订单物流已经发货。点击“确定”按钮，将该订单状态从已付款修改为已发货（如图4.51所示）。

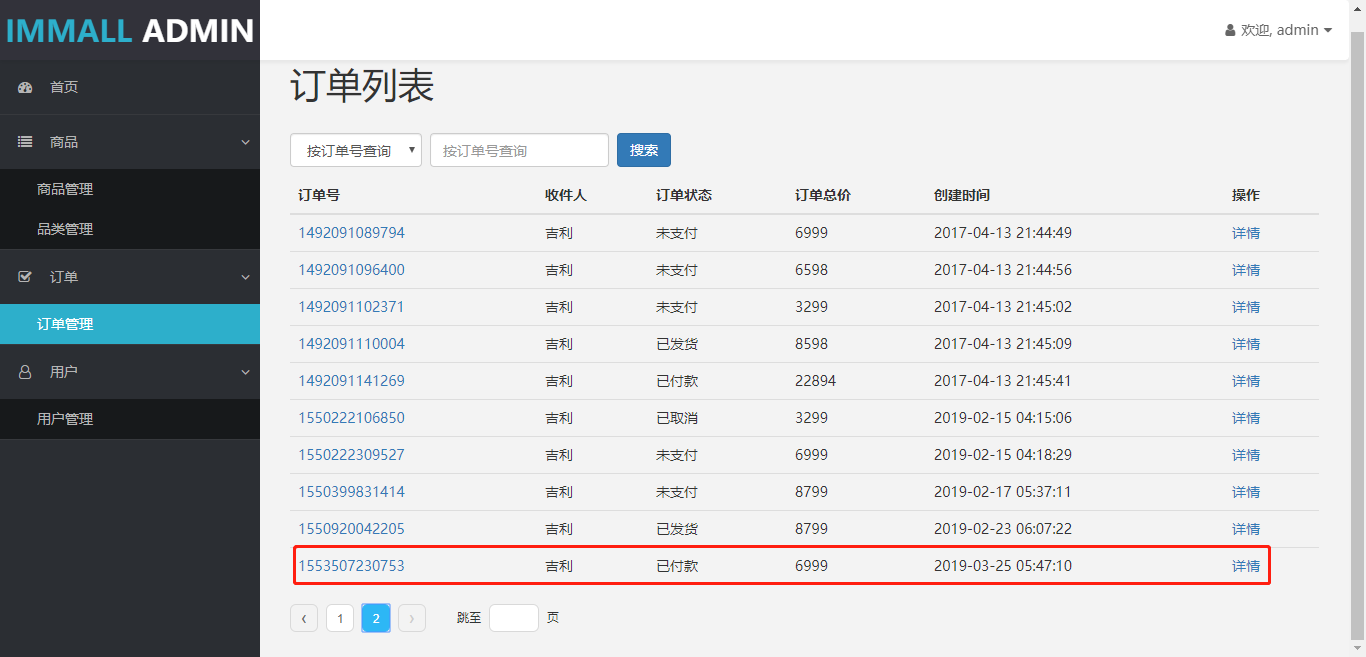
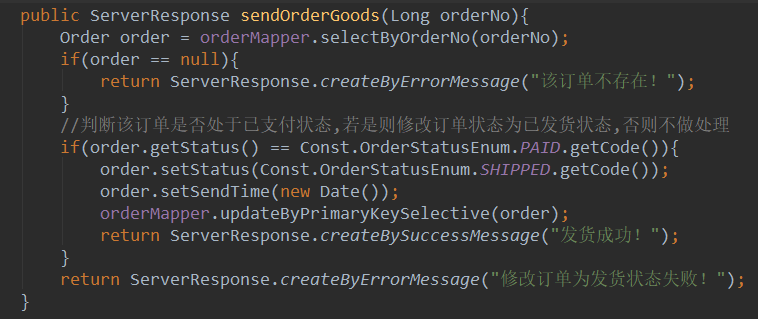


图4.51 修改已付款订单的状态



图4.52将订单状态从已付款修改为已发货

当点击页面弹出框上面的“确定”按钮时，前端即会发送Ajax请求，请求后端订单模块的send\_goods.do接口，该接口接收一个orderNo参数，该orderNo参数由前端传递过来。后端根据传递过来的orderNo参数，通过MyBatis框架在数据库中查询出指定的订单记录出来，先对其判断是否存在该订单，如果存在，那么该笔订单是否处于支付成功状态，是的话才执行真正的修改订单状态操作，否则返回相关的错误信息，关键代码如下：



4.2.3.4 搜索订单

在众多订单记录里面，管理员可以通过订单号，查找指定的订单记录。在订单列表页面，往输入框中输入订单号，然后点击“搜索”按钮，这时前端会发起Ajax请求，携带管理员输入的订单号作为参数，传递给系统后端。系统后端接收到前端传递过来的订单号参数后，通过ORM框架MyBatis，在数据库中进行精确匹配，然后将其查询结果返回给前端（如图4.52所示）。

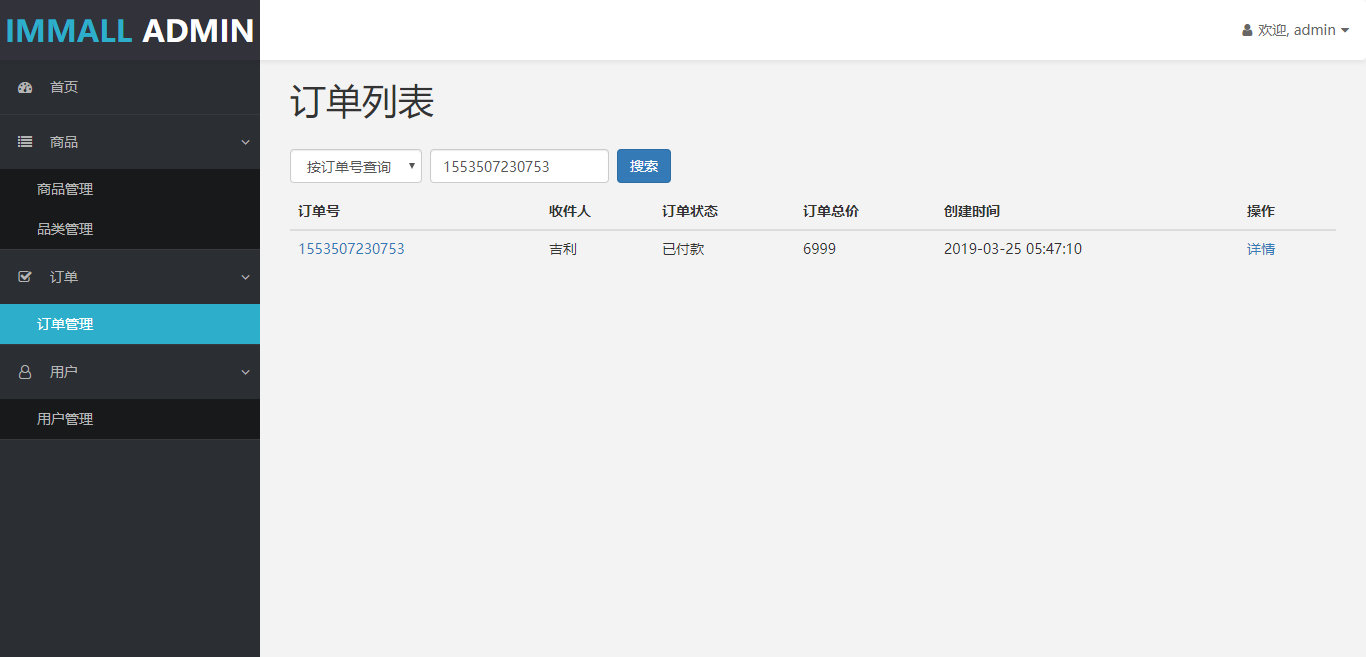
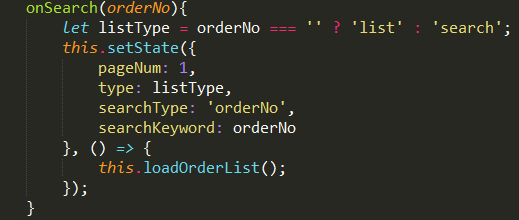
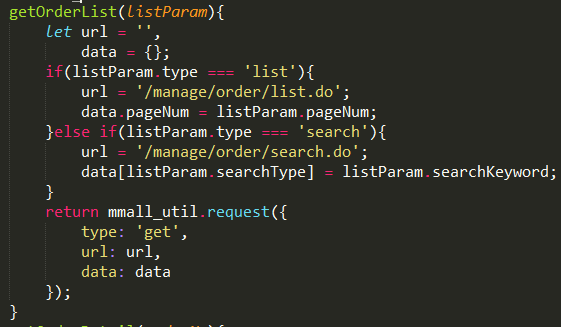


图4.53 按订单号查询订单

另外，如果管理员在没有输入任何订单号信息的情况下点击了“搜索”按钮，那么前端检测到订单号没有内容的话，会发起后端list.do接口的请求；否则发起后端search.do接口的请求。关键代码如下：





4.2.4 品类模块

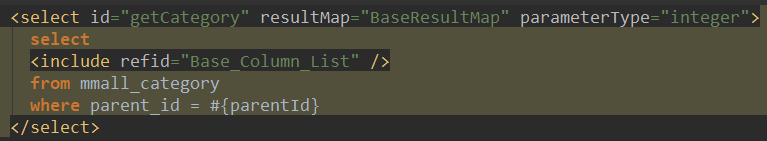
4.2.4.1 浏览品类列表

商城中所有商品都是有品类标识的。管理员可以查看目前商城中存有的所有品类信息。在管理员端首页的左侧导航栏里，点击“品类管理”按钮，即可查看到品类列表（如图4.53所示）。



图4.54浏览品类列表

点击“品类管理”按钮的时候，页面会自动通过Ajax请求后端品类模块的get\_category.do接口，与此同时，由于我们规定查询品类列表的时候默认是查询根品类下面的一级品类信息，因此前端默认将categoryId=0作为参数传递给后端，categoryId为0表明该category是根品类。后端获取到这个参数的时候，通过ORM框架MyBatis，去数据库中查找那些category记录中，parent\_id为前端传来的categoryId的值的记录，然后将这些记录返回。关键代码如下：



4.2.4.2 添加品类信息

如果商城此时有新的商品需要上架，但该商品的所属品类目前商城中并不存在，这时管理员就可以执行添加品类信息的操作。在品类管理列表页面，点击页面右上角的“添加品类”按钮，然后页面会跳转到品类添加页面（如图4.54所示），

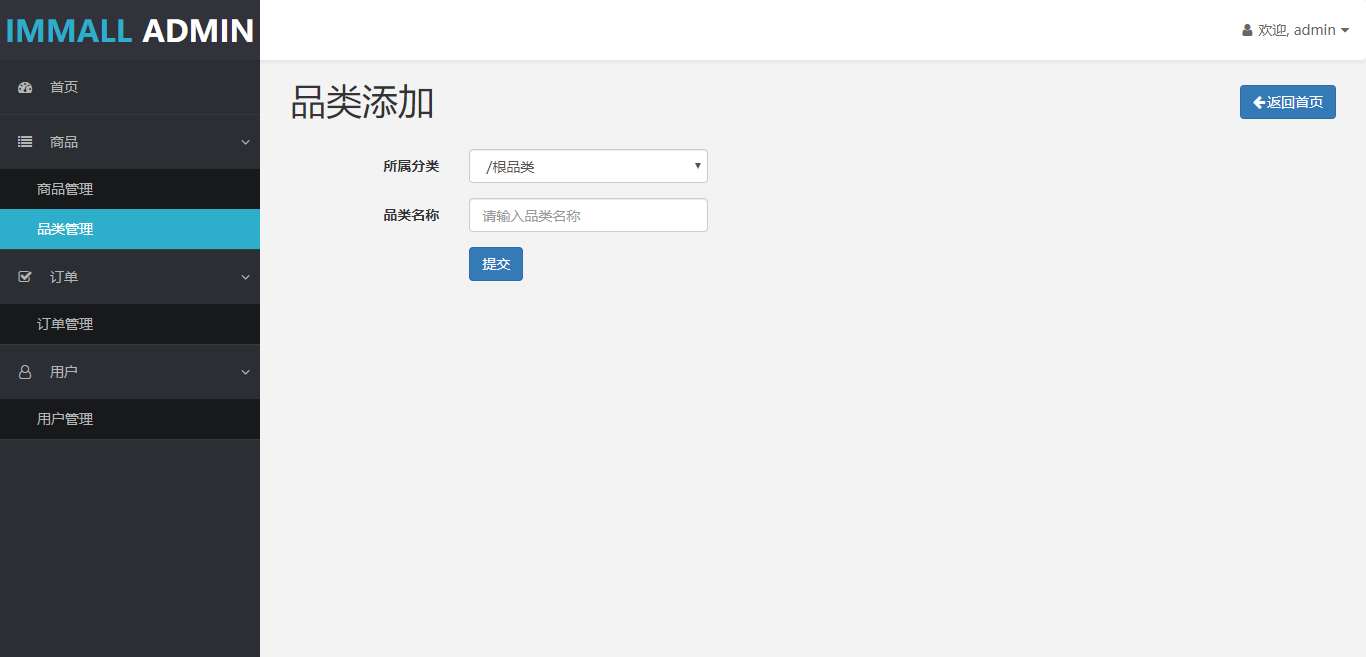
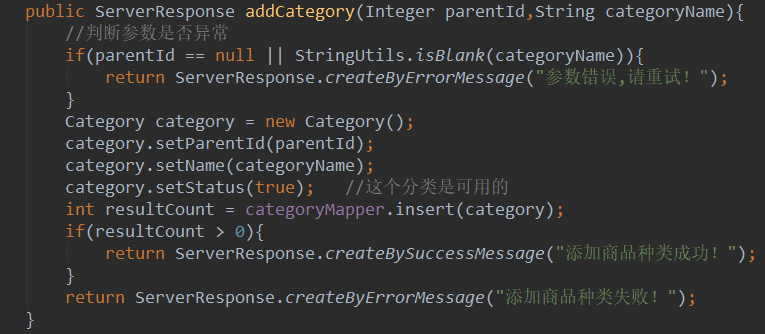


图4.55 新增品类

在品类添加页面，选择新品类的所属分类，然后填上新品类的名称，最后点击“提交”按钮，等待系统后端进行参数校验，如果没有填写新品类的名称，页面会弹出一个信息提示框，提示相关错误信息；否则本次品类新增操作成功。

提交新品类信息的时候，页面会通过Ajax请求，向后端品类模块调用add\_category.do接口，同时会把该新品类的父品类parentId和该新品类的名称categoryName作为参数传递给后端，后端获取到这两个参数后，会对参数进行合法性校验，校验通过便执行新增品类记录的操作。关键代码如下：



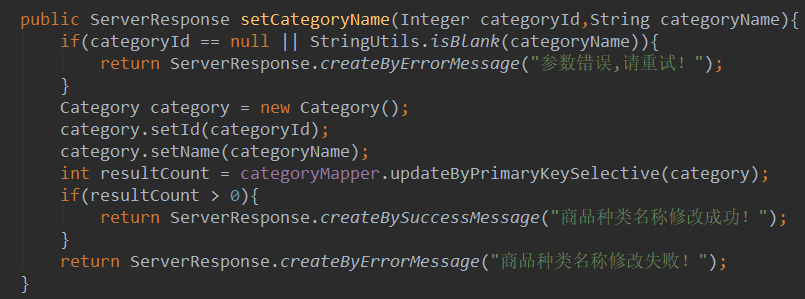
4.2.4.3 修改品类名称

管理员可以修改品类的名称。在品类管理页面，选择需要修改名称的品类，点击“修改名称”按钮，这时页面会弹出一个信息输入框，提示管理员输入新的品类名称，管理员输入完新的品类名称后，点击“确定”按钮，便完成品类名称的修改操作（如图4.55所示）。



图4.56 修改品类名称

在点击“确定”按钮后，前端JS代码获取对话框中的输入框的内容，如果输入框中的内容为空，则不执行任何操作；否则将通过Ajax请求，调用后端品类模块的set\_category\_name.do接口，后端接收到前端传来的categoryId与categoryName两个参数，然后对这两个参数进行合法性校验，如果校验通过，那么便执行为指定品类更新名称的操作；否则返回错误信息。关键代码如下：



4.2.4.4 查看子品类列表

在点击进入品类管理页面时，由于显示的品类默认是根品类下面的一级品类，因此如果想要查看某个一级品类下面的子品类列表信息，可以点击一级品类左侧的“查看子品类”按钮（如图4.56所示），



图4.57 点击“查看子品类”链接

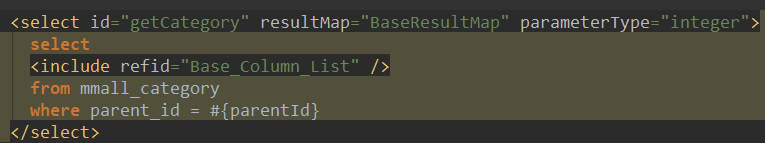
这时前端会携带当前一级品类的Id，发送Ajax请求调用后端品类模块的get\_category.do接口，后端接口接收到前端传递过来的品类Id参数后，通过MyBatis这个ORM框架，前往数据库中查找那些父品类Id为前端传递过来的品类Id的品类记录，然后将查找到的品类列表返回给前端（如图4.57所示）。



图4.58 查看一级品类的子品类列表信息

关键代码如下：





5 系统测试

5.1 测试环境

本系统测试分为服务端测试和客户端测试，下面是服务端与客户端的测试环境：

表5.1 服务端测试环境

|  |  |
| --- | --- |
| 服务端 | |
| 硬件 | CPU：E5 – Xeon  内存：1GB  硬盘大小：2TB  是否有SSD：无 |
| 软件 | OS类型及版本：CentOS 6.7 2.6.32-431.1.2.0.1 el6.x86\_64  Java版本：jdk1.8  Web服务器：Nginx 1.10.2  Web容器：Tomcat 7.2  MySQL数据库：MySQL 5.7.17-log Community Server  Redis数据库：Redis 2.8 |
| 带宽 | 20MB |

表5.2 客户端测试环境

|  |  |
| --- | --- |
| 客户端 | |
| 硬件 | CPU：Intel(R)Core(TM)i5-5200U @ 2.20GHz  内存：4GB  硬盘大小：460GB  是否有SSD：无 |
| 软件 | OS类型及版本：Windows 10 家庭中文版 17134.648  访问工具：浏览器 Google Chrome 版本 73.0.3683.86（正式版本） 64位 |
| 带宽 | 20MB |

5.2 测试范围

没有经过测试的软件或者系统，是不完整、不规范、不合乎工业标准的。软件测试，作为一个系统真正上线前的最后一个阶段，除了能确保开发出来的系统是否符合需求规格说明之外，它还能帮助我们发现系统目前现存的不足，好让我们知晓在往后的迭代周期里，我们应该怎么样去完善系统的功能，做到把一个系统打造成真正的产品，这样才能更好的服务于众。

本次系统测试，我们设计了一些测试用例，这些测试用例覆盖的范围有：

1. 功能性：系统提供的功能是否符合功能需求规格说明书的要求；
2. 准确性：系统提供的功能是否正确执行了业务逻辑；
3. 易用性：系统的操作界面是否简洁美观大方，对用户来说是否清晰明了；
4. 容错性：当系统或者用户操作出现异常时，是否能友好地提示相应的错误信息。

5.3 测试用例

由于本电商系统实际是分为两个子系统进行开发的，一个是电商系统门户端，一个是电商系统管理员端。因此，我们针对这两个子系统，分别设计了一些测试用例，去覆盖那些重要的应用场景。以下是测试用例详情：

5.3.1 门户端

5.3.1.1 匿名用户注册

表5.3 匿名用户注册测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例ID | T001 | 名称 | 匿名用户注册 | |
| 用例描述 | 通过输入合法的用户名、密码、联系电话、邮箱、密码提示问题和密码提示答案，匿名用户能够成功注册成为本电商系统的普通用户。 | | | |
|  | | | | |
| 步骤 | 场景 | 操作细节 | 预期结果 | 实际结果 |
| 1 | 进入注册界面 | 访问电商系统门户端的首页，然后在页面左上角点击“注册”按钮 | 进入注册界面 | 与预期一致 |
| 2 | 匿名用户注册 | 输入合法的用户名、密码、联系电话、邮箱、密码提示问题和密码提示答案，然后点击“注册”按钮，等待系统校验信息 | 系统校验通过，本次注册操作成功 | 与预期一致 |

5.3.1.2 用户登录

表5.4 用户登录测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例ID | T002 | 名称 | 用户登录 | |
| 用例描述 | 普通用户在成功访问到电商系统的情况下，输入合法的用户名和密码，成功登录进电商系统 | | | |
|  | | | | |
| 步骤 | 场景 | 操作细节 | 预期结果 | 实际结果 |
| 1 | 进入登录界面 | 用户访问电商系统门户端的登录页面 | 进入登录界面 | 与预期一致 |
| 2 | 用户登录 | 在登录页面上输入合法的用户名和密码，然后点击“登录按钮”，等待系统校验登录信息 | 系统校验成功，页面自动跳转到系统首页 | 与预期一致 |

5.3.1.3 商品搜索

表5.5 商品搜索测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例ID | T003 | 名称 | 搜索商品 | |
| 用例描述 | 不论是匿名用户还是已登录的普通用户，都能在商城首页通过关键字执行搜索商品的操作。 | | | |
|  | | | | |
| 步骤 | 场景 | 操作细节 | 预期结果 | 实际结果 |
| 1 | 进入商城首页 | 用户访问电商系统首页 | 进入电商系统首页 | 与预期一致 |
| 2 | 商品搜索 | 在首页搜索框里面输入关键字，搜索想要查看的商品，然后点击搜索按钮进行搜索 | 系统将返回携带搜索关键字的商品列表给用户 | 与预期一致 |
| 3 | 用户登录 | 在登录页面上输入合法的用户名和密码，然后点击“登录按钮”，等待系统校验登录信息 | 系统校验成功，页面自动跳转到系统首页 | 与预期一致 |
| 4 | 商品搜索 | 执行步骤2 | 系统将返回携带搜索关键字的商品列表给用户 | 与预期一致 |

5.3.1.4 浏览商品详情

表5.6 查看商品详情测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例ID | T004 | 名称 | 浏览商品详情 | |
| 用例描述 | 不论是匿名用户亦或是普通用户，在商品列表中点击某个商品，能够进入到该件商品的详情页面内，浏览该件商品的详情内容 | | | |
|  | | | | |
| 步骤 | 场景 | 操作细节 | 预期结果 | 实际结果 |
| 1 | 进入商城首页 | 用户访问电商系统首页 | 进入电商系统首页 | 与预期一致 |
| 2 | 浏览商品详情 | 随意点击商品列表中的某个商品，然后进入该件商品的详情内容页面 | 新进入的页面能够正确显示所点击商品的详情内容 | 与预期一致 |
| 3 | 用户登录 | 在登录页面上输入合法的用户名和密码，然后点击“登录按钮”，等待系统校验登录信息 | 系统校验成功，页面自动跳转到系统首页 | 与预期一致 |
| 4 | 浏览商品详情 | 执行步骤2 | 新进入的页面能够正确显示所点击商品的详情内容 | 与预期一致 |

5.3.1.5 购物车添加

表5.7 购物车添加测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例ID | T005 | 名称 | 添加购物车 | |
| 用例描述 | 在登录状态下，对心仪的商品点击“加入购物车”按钮，系统能正确跳转到成功加入购物车的页面 | | | |
|  | | | | |
| 步骤 | 场景 | 操作细节 | 预期结果 | 实际结果 |
| 1 | 进入商城首页 | 用户访问电商系统首页 | 进入电商系统首页 | 与预期一致 |
| 2 | 用户登录 | 在登录页面上输入合法的用户名和密码，然后点击“登录按钮”，等待系统校验登录信息 | 系统校验成功，页面自动跳转到系统首页 | 与预期一致 |
| 3 | 浏览商品详情 | 随意点击商品列表中的某个商品，然后进入该件商品的详情内容页面 | 新进入的页面能够正确显示所点击商品的详情内容 | 与预期一致 |
| 4 | 添加购物车 | 在内容详情页点击“加入购物车”按钮 | 系统进行页面跳转，跳转到商品成功添加进购物车页面 | 与预期一致 |
| 5 | 查看购物车 | 在成功添加进购物车的页面，点击“查看购物车”按钮 | 系统能够跳转到购物车页面并显示出先前已添加进来的商品信息 | 与预期一致 |

5.3.1.6 订单提交

表5.8 订单提交测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例ID | T006 | 名称 | 提交订单 | |
| 用例描述 | 已登录的用户，在添加商品进购物车后，可以对购物车中的商品进行订单提交以及订单支付 | | | |
|  | | | | |
| 步骤 | 场景 | 操作细节 | 预期结果 | 实际结果 |
| 1 | 进入商城首页 | 用户访问电商系统首页 | 进入电商系统首页 | 与预期一致 |
| 2 | 用户登录 | 在登录页面上输入合法的用户名和密码，然后点击“登录按钮”，等待系统校验登录信息 | 系统校验成功，页面自动跳转到系统首页 | 与预期一致 |
| 3 | 浏览商品详情 | 随意点击商品列表中的某个商品，然后进入该件商品的详情内容页面 | 新进入的页面能够正确显示所点击商品的详情内容 | 与预期一致 |
| 4 | 添加购物车 | 在内容详情页点击“加入购物车”按钮 | 系统进行页面跳转，跳转到商品成功添加进购物车页面 | 与预期一致 |
| 5 | 查看购物车 | 在成功添加进购物车的页面，点击“查看购物车”按钮 | 系统能够跳转到购物车页面并显示出先前已添加进来的商品信息 | 与预期一致 |
| 6 | 订单确认 | 在当前的购物车页面，点击“去结算”按钮 | 系统进入订单确认页面 | 与预期一致 |
| 7 | 提交订单 | 在当前的订单确认页面，确认收货地址和商品信息无误后，点击“提交订单”按钮 | 系统进入订单支付页面 | 与预期一致 |
| 8 | 支付订单 | 在当前的订单支付页面，使用手机支付宝对页面中的二维码进行扫码支付 | 系统显示支付成功 | 与预期一致 |

5.3.2 管理员端

5.3.2.1 管理员登录

表5.9 管理员登录测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例ID | T007 | 名称 | 管理员登录 | |
| 用例描述 | 管理员使用合法的账号密码能够成功登录电商系统的管理员端 | | | |
|  | | | | |
| 步骤 | 场景 | 操作细节 | 预期结果 | 实际结果 |
| 1 | 进入登录界面 | 用户访问电商系统管理员端的登录页面 | 进入登录界面 | 与预期一致 |
| 2 | 管理员登录 | 在登录页面上输入合法的用户名和密码，然后点击“登录按钮”，等待系统校验登录信息 | 系统校验成功，页面自动跳转到管理员端首页 | 与预期一致 |

5.3.2.2 商品新增

表5.10 商品新增测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例ID | T008 | 名称 | 商品新增 | |
| 用例描述 | 管理员成功登陆进电商管理系统后，能够执行新增商品操作，并且操作能够成功 | | | |
|  | | | | |
| 步骤 | 场景 | 操作细节 | 预期结果 | 实际结果 |
| 1 | 进入登录界面 | 用户访问电商系统管理员端的登录页面 | 进入登录界面 | 与预期一致 |
| 2 | 管理员登录 | 在登录页面上输入合法的用户名和密码，然后点击“登录按钮”，等待系统校验登录信息 | 系统校验成功，页面自动跳转到管理员端首页 | 与预期一致 |
| 3 | 商品列表浏览 | 进入管理员端首页后，点击“商品管理”导航选项 | 导航栏右侧页面会跳转到商品列表页面 | 与预期一致 |
| 4 | 商品添加 | 在商品列表页面的右上角，点击“添加商品”按钮，进入商品添加页面后，填写新商品的信息，然后完成信息提交 | 在商品添加页面填写完新商品的信息后，点击“提交”按钮，系统显示操作成功 | 与预期一致 |

5.3.2.3 品类新增

表5.11 品类新增测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例ID | T009 | 名称 | 品类新增 | |
| 用例描述 | 管理员在品类管理列表页面中，能够执行新增商品种类的操作 | | | |
|  | | | | |
| 步骤 | 场景 | 操作细节 | 预期结果 | 实际结果 |
| 1 | 进入登录界面 | 用户访问电商系统管理员端的登录页面 | 进入登录界面 | 与预期一致 |
| 2 | 管理员登录 | 在登录页面上输入合法的用户名和密码，然后点击“登录按钮”，等待系统校验登录信息 | 系统校验成功，页面自动跳转到管理员端首页 | 与预期一致 |
| 3 | 品类列表浏览 | 进入管理员端首页后，点击“品类管理”导航选项 | 导航栏右侧页面会跳转到品类列表页面 | 与预期一致 |
| 4 | 品类添加 | 在品类列表页面的右上角，点击“添加品类”按钮，进入品类添加页面后，填写新品类的信息，然后完成信息提交 | 在品类添加页面填写完新品类的信息后，点击“提交”按钮，系统显示操作成功 | 与预期一致 |

5.3.2.4 订单状态修改

表5.12 订单状态修改测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例ID | T010 | 名称 | 订单状态修改 | |
| 用例描述 | 对于已经支付成功的订单，管理员能够修改该笔订单的状态，从已支付的状态修改为已发货状态 | | | |
|  | | | | |
| 步骤 | 场景 | 操作细节 | 预期结果 | 实际结果 |
| 1 | 进入登录界面 | 用户访问电商系统管理员端的登录页面 | 进入登录界面 | 与预期一致 |
| 2 | 管理员登录 | 在登录页面上输入合法的用户名和密码，然后点击“登录按钮”，等待系统校验登录信息 | 系统校验成功，页面自动跳转到管理员端首页 | 与预期一致 |
| 3 | 订单列表浏览 | 进入管理员端首页后，点击“订单管理”导航选项 | 导航栏右侧页面会跳转到订单列表页面 | 与预期一致 |
| 4 | 订单详情查看 | 在订单列表中，选择订单状态为已付款的订单，点击“详情”按钮，进入其订单详情页面 | 导航栏右侧跳转到该笔订单的详情页面 | 与预期一致 |
| 5 | 修改订单状态 | 在订单详情页面，确认订单状态为已付款后，点击按钮“立即发货” | 页面弹出信息提示框请求管理员二次确认，点击确认按钮后系统提示订单发货成功 | 与预期一致 |

5.4 测试总结

经过一系列的测试，我们已经确定系统功能能够基本正常运作，需求规格说明书上的用例需求也已全面覆盖，各个业务模块的执行流程也都符合要求。与此同时，在测试过程中发现的缺陷和不足也都及时得到修复，系统可以作为第一版正式上线。

6 总结与展望

写到这里时，就表明本次毕业论文的撰写即将完成了。本次毕业设计主题围绕企业级电商系统进行研发，使用了Spring+SpringMVC+MyBatis作为项目的基础架构，与此同时，为了做成一个高可复用与较高性能的电商系统，我们还采用了Web容器集群与Redis分布式缓存的架构，用来应对突然爆炸式增长的QPS。换句话说，我们也是想在基本的框架支持下，尝试对架构进行一次升级，探索架构在逐渐演变的过程中，早先可复用的代码是否依旧需要进行重构，以及由于架构的升级，对项目的整体组织影响又有多大。当然，我们从原本的ALL IN ONE架构直接升级为Web容器集群与Redis分布式缓存架构，从发展的角度来看，集群是通过把系统进行横向扩展来达到性能的提升，而Redis分布式缓存则是通过对用户请求产生的会话进行分布式存储（这里采用了一致性哈希算法，解决了哈希倾斜性的问题），解决了服务器集群环境下对Session的共享问题，这样也就意味着，在高并发、高流量的情况下，我们的系统能够更好地实现资源请求的负载；但从资源使用的角度看，其实我们没有做到单台服务器的资源使用最大化，这在商业应用场景中，是决不允许出现的。

通过这次系统开发，我深深感受到，一个优秀的系统，不是一蹴而就的，而是一步步、逐渐演变出来的。软件工程，是浩大、严谨的，在系统开发中，我们遵循了典型的MVC分层架构思想，把“低耦合、高内聚”的编程范式贯穿了整个研发过程，同时还明白了：一个良好的需求与设计开端，是后期编码与测试能否成功交付的重要依据。我相信，这在我即将到来的社会工作之旅中，是意义非常重大的。

最后，本次论文撰写，由于作者水平有限，难免会有一些错误与不足之处，届时恳请各位老师批评斧正。

参考文献

[1] 鸟哥 著. 鸟哥的Linux私房菜基础学习篇[M]. 人民邮电出版社. 2010.

[2] Craig,Walls 著. Spring实战 第4版[M].人民邮电出版社. 2016年4月.

[3] 杨开振 著. Java EE互联网轻量级框架整合开发[M]. 电子工业出版社.

[4] 陶辉 著. 深入理解Nginx-模块开发与架构解析[M]. 机型工业出版社. 2016年2月.

[5] 卡尔森 著. Redis实战[M]. 人民邮电出版社. 2015年11月01日.

[6] (美) Craig Larman 著 李洋 郑龑 等译. UML和模式应用[M]. 机械工业出版社. 2014年10月.

[7] Baron Scbwartz, Peter Zaitsev, Vadim Tkacbenko 著, 高性能MySQL[M]. 电子工业出版社.

[8] 罗伯特·C.马丁 著. 架构 整洁之道[M]. 电子工业出版社.

[9] 赵卓君等. Java 程序设计高级教程 [M]. 北京：北京交通大学出版社、清华大学出版社，2011.

[10] 韩万江，姜立新. 软件项目管理案例教程 [M]. 北京：机械工业出版社，2009.

[11] Andrew S. Tanenbaum, Herbert Bos著，陈向群 译，现代操作系统（原书第4版）. 机械工业出版社. 2017-07-01.

谢 辞

又是一年毕业季，学校教学楼周围的梧桐树又茂盛了许多。回想当初自己大一时候刚刚踏入校园的样子，那个时候多天真、多稚嫩，那个时候的自己，下决心要在大学四年里，创造出属于自己的价值。然后从很早就确立了自己的职业发展路径，接着便是日复一日的学习与坚持，不断学习，不断探索新的知识领域，不断总结自己一路过来的得与失。

然后就到了最后一个骄阳盛夏，这个春夏，经过数月的努力，自己的毕业设计作品与论文终于完成了。在此期间，我非常感谢自己的毕设指导老师——魏志军老师，如果没有他一开始对我论文选题提出具有建设性的意义，我想我的毕设不会进行得这么顺利。所以在这里，我想表达自己对魏老师的感谢，感谢他在我完成毕业作品与论文撰写的过程中所给予的指导与肯定。

与此同时，我还要感谢自己大学四年来遇到的每一位老师，如果没有他们在三寸讲台上倾囊相授与谆谆教导，我想我不会成长得这么快。

然后，我还得感谢大学来认识的好朋友、好伙伴。他们在与我前进的道路上，相互激励、彼此互助，没有他们，也没有今天的自己。

最后，感谢我的家人，他们在生活、学习上给予了我莫大的支持，他们既是我温暖的避风港，也是我前进道路的指明灯。

论文撰写到这里，也意味着自己的大学旅途，终于可以划上一个句号了。对于自己来说，这既是终点，也是起点。大学是一座象牙塔，我们在这里面，经过不断自我怀疑与自我重塑。最终在离开之前，获得了自我意识与自我价值的肯定，这将让我们在未来的工作道路上，能够走得更远，飞得更高。

最后的最后，感谢在百忙之中抽空出来参加论文评审与答辩的老师们，谢谢你们，老师们辛苦了！

附 录