**基于SSM的网上商场**

**摘要**

随着计算机技术和网络的发展，网络购物平台因为其成本小，没有时间，空间上的限制，慢慢成为新的零售模式和理念，仅仅是信息的浏览和发布已经不能满足人们的需求，人们想尽最大可能的享受互联网所带来的便捷，网上购物恰好符合人们的要求，使人们能够点击屏幕便可以选购自己心仪的商品。

本系统分为前台和后台部分，前台针对用户，主要包括购物车管理，订单管理，修改账户密码，添加收货地址信息,用户注册账号，登录系统等。后台是针对管理员，主要包括商品的修改删除添加理，处理订单的状态，用户个人的信息管理等。在最后的系统开发实现环节，使用面向对象来进行程序编写，使程序可重用性大大提高，让系统设计的更合理也更具有通用性，从而让系统实用性大大提高。建立后的网站系统是一个动态、交互式、具有用户账号注册登录，商品浏览、商品购买、系统管理、等功能的电子商务网站。

本系统以Java语言为基础，Ajax和Bootstrap为开发页面的框架，通过Servlet来实现功能，JDBC连接MYSQL数据库。在不断的不懂、思考、错误、网上查询、尝试、学习中，我也感觉到了自己的成长，也让我对系统开发有更深的了解。

关键字：JSP;Mysql;Java;网上购物系统

**abstract**

With the development of computer technology and network, online shopping platform has gradually become a new retail mode and concept because of its small cost, no time and space restrictions. Only the browsing and publishing of information can not meet people's needs. People want to enjoy the convenience brought by the Internet as much as possible. Online shopping just meets people's requirements, so that people can Click on the screen to choose your favorite products.

This system is divided into foreground and background parts. The foreground is aimed at users, mainly including shopping cart management, order management, account password modification, receiving address information addition, user account registration, login system, etc. The background is for the administrator, mainly including the modification, deletion and addition of goods, the status of processing orders, the user's personal information management, etc. In the last part of the system development, the object-oriented programming is used to improve the reusability of the program, make the system design more reasonable and more general, so as to improve the practicability of the system. The established website system is a dynamic, interactive, e-commerce website with the functions of user account registration and login, commodity browsing, commodity purchase, system management, etc.

The system is based on Java language, Ajax and bootstrap are the framework of developing page, and functions are realized through servlet. JDBC connects MySQL database. In the continuous incomprehension, thinking, error, online query, trial and learning, I also feel my growth, and let me have a deeper understanding of system development.

Keywords: JSP; MySQL; Java; online shopping system

**目录**

[引言 - 1 -](#_Toc13601)

[第1章 绪论 - 2 -](#_Toc11246)

[1.1研究背景 - 2 -](#_Toc20295)

[1.2研究意义 - 2 -](#_Toc24622)

[1.3发展状况 - 2 -](#_Toc17935)

[1.4研究的主要内容 - 2 -](#_Toc28254)

[第2章 系统开发环境及项目开发工具 - 4 -](#_Toc15503)

[1.1 开发技术简介 - 4 -](#_Toc15825)

[1.1.1 JSP简介 - 4 -](#_Toc29438)

[1.1.2 Servlet相关简介 - 4 -](#_Toc29518)

[1.1.3 MySql相关简介 - 4 -](#_Toc29385)

[1.1.4 IDEA简介 - 5 -](#_Toc4936)

[1.1.5 J2EE - 5 -](#_Toc27361)

[1.2 开发环境配置 - 5 -](#_Toc7677)

[1.2.1 JDK的安装与配置 - 5 -](#_Toc24546)

[第3章 需求分析 - 6 -](#_Toc27371)

[3.1 可行性分析 - 6 -](#_Toc7572)

[3.1.1经济可行性分析 - 6 -](#_Toc30841)

[3.1.2技术可行性分析 - 6 -](#_Toc1182)

[3.1.3社会可行性分析 - 6 -](#_Toc26978)

[3.3 结构化分析 - 6 -](#_Toc16493)

[3.3.1 数据流图 - 6 -](#_Toc2170)

[3.3.2 数据字典 - 9 -](#_Toc15119)

[3.2 系统需求分析 - 15 -](#_Toc15698)

[第4章 系统设计 - 16 -](#_Toc4934)

[4.1 系统模块设计 - 16 -](#_Toc18083)

[4.1.1前台面向用户的功能模块设计 - 16 -](#_Toc27273)

[4.1.2 后台模块的功能模块设计 - 17 -](#_Toc27601)

[4.2 系统流程设计 - 17 -](#_Toc27712)

[4.3 数据库设计 - 20 -](#_Toc15507)

[4.3.1数据库概要设计 - 20 -](#_Toc31588)

[4.3.2数据库表 - 25 -](#_Toc383)

[4.4本章小结 - 28 -](#_Toc29113)

[第5章 系统实现 - 29 -](#_Toc31454)

[5.1前台设计 - 29 -](#_Toc28070)

[5.1.1用户注册 - 29 -](#_Toc22426)

[5.1.2 用户登录 - 29 -](#_Toc19464)

[5.1.3 商品浏览 - 30 -](#_Toc11342)

[5.1.4添加商品到购物车 - 34 -](#_Toc5498)

[5.1.5订单相关页面 - 38 -](#_Toc13293)

[5.2 后台实现部分 - 41 -](#_Toc8925)

[5.2.1商品类型管理 - 41 -](#_Toc11943)

[5.2.2商品管理 - 42 -](#_Toc6953)

[5.2.3订单管理 - 45 -](#_Toc6388)

[5.2.4公告管理 - 47 -](#_Toc15570)

[5.2.5用户管理 - 48 -](#_Toc1000)

[第6章 系统测试 - 52 -](#_Toc14149)

[6.1软件测试 - 52 -](#_Toc24897)

[6.1.1注册登录测试 - 52 -](#_Toc15683)

[6.1.2购物车功能测试 - 54 -](#_Toc26830)

[6.1.3订单功能测试 - 55 -](#_Toc32552)

[6.1.4商品评价测试 - 55 -](#_Toc10782)

[6.1.5收货地址测试 - 56 -](#_Toc29792)

[6.1.6商品分类管理测试 - 57 -](#_Toc13466)

[6.1.7商品的添加，修改测试 - 57 -](#_Toc20852)

[6.1.8订单管理测试 - 58 -](#_Toc30080)

[6.1.9用户管理测试 - 58 -](#_Toc26132)

[6.1.10商品评论回复管理测试 - 58 -](#_Toc21468)

[6.2测试计划执行情况 - 59 -](#_Toc2795)

[6.3测试效果 - 60 -](#_Toc26553)

[6.4小结 - 60 -](#_Toc464)

[总结与展望 - 61 -](#_Toc16060)

[致 谢 - 62 -](#_Toc20563)

[参考文献 - 63 -](#_Toc17581)

[附录 - 64 -](#_Toc9648)

[附录A 外文文献及其译文 - 64 -](#_Toc23588)

[附录B 主要参考文献的题录及摘要或参考文献原文 - 73 -](#_Toc12611)

[附录C 部分源代码 - 76 -](#_Toc31142)

插图清单

[图2-1 原理图 8](#_Toc28406)

[图2-2 安装成功图 10](#_Toc2078)

[图3-1 顶层图数据流图 12](#_Toc31030)

[图3-2 一层数据流图 12](#_Toc28258)

[图3-3 用户注册及信息管理二层数据流图 13](#_Toc13467)

[图3-4 订购商品二层数据流图 13](#_Toc15944)

[图3-5 订单管理二层数据流图 14](#_Toc17667)

[图3-6 商品管理二层数据流图 14](#_Toc17082)

[图4-1 系统功能图 17](#_Toc20436)

[图4-2 前台购物流程图 18](#_Toc16139)

[图4-3 后台管理流程图 19](#_Toc15661)

[图4-4 注册功能流程图 19](#_Toc15829)

[图4-5 登录流程图 19](#_Toc19778)

[图4-6 商品搜索购买流程图 20](#_Toc9930)

[图4-7 商品管理流程图 21](#_Toc27088)

[图4-8 商品实体属性图 22](#_Toc9372)

[图4-9 顾客实体属性图 22](#_Toc11918)

[图4-10 管理员实体属性图 22](#_Toc18293)

[图4-11 订单实体属性图 23](#_Toc26292)

[图4-12 收货地址实体属性图 23](#_Toc24690)

[图4-13 购物车实体属性图 23](#_Toc31081)

[图4-14 商品类别实体属性图 24](#_Toc1553)

[图4-15 系统公告实体属性图 24](#_Toc22583)

[图4-16 商品评价实体关系图 24](#_Toc5699)

[图4-17 ER图 25](#_Toc12732)

[图5-1 用户登录 31](#_Toc9933)

[图5-2 首页 32](#_Toc27461)

[图5-3 商品信息图 34](#_Toc3831)

[图5-4 注册 34](#_Toc3605)

[图5-5 购物车 36](#_Toc3430)

[图5-6 订单 40](#_Toc30467)

[图5-7 编辑分类 43](#_Toc15907)

[图5-8 添加商品 45](#_Toc2567)

[图5-9 订单管理 47](#_Toc18822)

[图5-10 公告管理 49](#_Toc31696)

[图5-11 用户管理 50](#_Toc8279)

[图6-1 注册测试 55](#_Toc28075)

[图6-2 登录测试 56](#_Toc1663)

[图6-3 购物车测试 57](#_Toc8211)

[图6-4 订单测试 57](#_Toc20406)

[图6-5 评论测试 58](#_Toc17413)

[图6-6 商品信息测试 58](#_Toc9565)

[图6-7 收货信息测试一 59](#_Toc11015)

[图6-8 收货信息测试二 59](#_Toc4805)

[图6-9 分类测试 59](#_Toc21913)

[图6-10 添加商品测试 60](#_Toc3075)

[图6-11 订单管理测试 60](#_Toc26431)

[图6-12 用户管理测试 61](#_Toc21220)

[图6-13 评论回复测试 61](#_Toc10466)

表格清单

[表4-1 表名user 28](#_Toc4986)

[表4-2 表名product 29](#_Toc4351)

[表4-3 表名：orders 29](#_Toc25425)

[表4-4 表名：orderitem 30](#_Toc16694)

[表4-5 表名：car 30](#_Toc1781)

[表4-6 表名：admin 30](#_Toc3174)

[表4-7 表名：notice 31](#_Toc27639)

[表4-8 表名：category 31](#_Toc15315)

[表4-9 表名：goodscomments 31](#_Toc28961)

[表6-1 测试计划表 61](#_Toc9229)

[表6-2 测试结果表 62](#_Toc31421)

**引言**

目前，互联网在以一种令人难以想象的冲击力在改变着人们的生活方式，不仅与人们的生活息息相关，还改变了生产方式。互联网的诞生和迅速发展不仅改变了传统信息传递方式，还打破了传统交流方式中时间和空间的各种限制，极大地改变了人们从形式到内容，从生产到生活，从物质到精神的各种活动，并且不断给人们带来了新的机会和挑战。

自21世纪以来，中国互联网高速发展，涉及到互联网基础建设，互联网政务，互联网学术以及互联网学术等方面。根据有关部门的相关报告，截至2020年，中国网民人数达到9亿，互联网普及率到达64%。巨大的网民规模造就了中国这个世界上最大的消费市场。同时，基于计算机科学信息技术的高速发展，数字经济也有了相当夯实的技术基础和用户基础。

本次毕业设计的课题是利用SSM+bootstrap+jsp技术搭建一个网上商场的购物平台。近年世界各国都出现了电子商务热，越来越多的企业都在建设自己的网站。网上购物这个逐渐流行于21世纪的购物方式已经被越来越多的人所接受我国政府也正全面，积极，稳妥的推进电子商务的发展。网上商店在增加移动支付功能，完善各项功能后以更大的势头发展。电子商务代表着未来世界贸易的发展方向和趋势，其应用和推广将给社会和经济带来巨大的效益。大力发展电子商务在国民经济各个领域的应用，力争在越来越激烈的竞争中占据有利位置。因此对于网上商城的研究和实现，以及搭建一个实用的网上商店是很有必要的。

本系统为消费者提供了网上浏览商品，登陆，注册，搜索商品，加入购物车，购买商品。满足消费者的网上购物的基本活动。为商家提供了，管理注册用户，管理商品信息，增加、删除商品等功能从商家的方面考虑，为商家节约了成本，方便了管理。

**第1章 绪论**

**1.1研究背景**

21世纪以来，全球各国进入一个崭新的经济时代，[信息](http://www.99inf.net/" \t "_blank)通信[技术](http://www.99inf.net/" \t "_blank)和网络的迅猛发展创造了一个无边界的数字世界，产品和金融服务几乎可以在人们感受不到的时间跨度中在世界各地进行交易，以网络和电子商务为主要特征的新经济以不可争辩的事实席卷了全球每一个行业。

自从中国加入WTO这个国际贸易组织后，传统企业发展的必然趋势是网络化、全球化。通过建设电子商务基础平台，使得企业不必专注于复杂多变的[信息技术](http://www.99inf.net/" \t "_blank)，而用最小的资金投入、最简单的接入方式去联系最广泛的客户和供应商，进行最先进、最有效的电子商务应用服务,相关企业实行电子商务则成为企业发展与国际接轨的必要条件，所以电子购物平台的实现是非常迫切的[3]。

当今的零售企业，如果依旧使用传统的方式例如：发传单，实体店内打广告宣传自己的产品、本公司的文化则显得十分落后，与现代企业的发展不相符，并且难以适应激烈的市场竞争，最终被淘汰。在这个流量为王的时代，互联网明显更适合宣传产品，随意点击一个网站都可以看到铺天盖地的产品广告。因此，当代零售企业纷纷搭建具有本公司文化的网站，利用在互联网技术快速的发展让自己的公司跟上时代的潮流。

**1.2研究意义**

随着电子商务的发展，无店铺零售悄然兴起，它是商业经济发展到一定阶段的产物，也是顺应新的消费需求和技术进步的必然结果。无店铺零售则大多不存在地段方面的制约。二是经营成本。借助不断发展创新的计算机技术，无实体店零售往往能够节省从生产到销售过程中的许多过程，进而有效降低经营成本与交易成本，带来消费者、商家双赢的局面[2]。

鉴于互联网的特点以及对国内相关网上购物网站现状的研究和分析，本次毕业设计是以java Web 开发为基础的网上购物网站开发作为本人的毕业设计主要课题。立足于设计一个在网络平台上运行的商城，以小小的力量推动中国互联网的发展。

**1.3发展状况**

随着现代计算机信息相关技术的深入，广泛地应用到现代社会的许多领域并且发挥着不可替代的作用。企业信息管理的必然趋势一定会涉及到计算机信息技术。计算机软件作为计算机信息技术应用的一部分，充分发挥计算机的相关优势，将海量复杂的数据交给计算机系统来处理，相对于人工管理来说，计算机系统有其所无法比拟的优点。例如：查找迅速、可靠性相当高、存储能力极强，不受空间时间约束、保密性好，由于加密技术的存在，重要信息泄露的概率小、寿命长、成本不高等[6]。以上这些特点不仅能够长时间极大地提升企业信息管理的效率，节约成本，是真正的合理利用社会有限的资源，也是企业信息管理数字化、正规化,科学化与全球数字经济接轨的重要条件。

**1.4研究的主要内容**

作为一个网上商城系统,就应该做到能提供强大的业务支持功能，系统能实现顾客的注册账户功能、登录系统功能、商品的查询，订购等功能。该系统基本上具备一个商品销售网站应该具备的功能，该项目基本上体现了构建一个动态商务网站所需要的技术。

主要功能：

（1）用户的功能模块主要是实现了用户对系统账号注册，登录系统，系统公告、商品的浏览以及查询订购商品操作，在注册登录后，可以实现将物品加入购物车并提交订单等功能。

（2）后端信息管理模块主要是实现管理员的登录，以及登陆之后对公告的修改，商品的添加删除修改，用户的订单状态管理，以及删除相关订单，管理前台用户信息等功能

（3）注册登录模块：进入系统主页首先默认是游客角色。可以浏览搜索商品，在注册登陆之后成为会员才能加入商品到购物车并提交订单

（4）购买商品模块：成功登陆之后，才可以将商品加入到购物车，选择完商品之后可以继续浏览商品加入购物车或者直接提交购物车里的商品到订单

（5）管理员模块：登录管理员账户之后可以对商品进行各种操作，以及对公告的修改，对订单状态的处理，对用户信息的维护等。

**第2章 系统开发环境及项目开发工具**

1.1 开发技术简介

1.1.1 JSP简介

JSP就是Java 服务器页面的简称。JSP是一种HTML扩展对象，在其上我们可以通过特定的方式，让我们直接使用Java代码来帮助我们实现页面功能。其中有几个使我们在JSP常用的编码方式，我们习惯于用TL表达式来帮助实现功能，而不直接使用Java代码来实现功能。

JSP在JSP代码执行前创建了变量，因此在JSP代码内可以直接使用这些变量，此变量又名为隐含变量，创建这些变量的作用是为了使我们使用这些对象方便。下图可以我们帮助地理解这两者的关系：

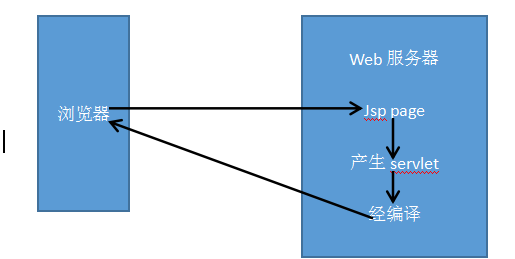


图2- 1原理图

1.1.2 Servlet相关简介

Servlet全称是java servlet,Servlet是一种服务器端的java程序，具有独立性，它可以处理用户传过来的请求，也可以通过响应向用户输送数据。Servlet就是处理用户的HTTP请求，然后回送HTTP响应，若是没有Servlet技术出现的话，我们还要手动的使用一些基本、底层的类去解析HTTP。我们只需要在自己编写的类上重写对应的请求方法就可以处理用户通过给HTTP方法传送过来的请求了。与传统的java程序不同，它由web服务器进行加载，该web服务器必须包含可以启动servlet的jvm。

1.1.3 MySql相关简介

MySQL数据库是一种小型数据库，它一般用于小型软件开发或程序员初学者进行学习的，不过我们都是使用5.0版本的，因为6.0之后版本的MySQL都是要钱的。MySQL数据库本质是一种文件系统，我们一般都是用SQL语句来对它进行操作，从而对数据进行。MySQL数据库是一种关系型数据库的管理系统，关系型数据库是指在不同的表中保存数据的数据库。它保存关系数据模型,即保存数据也保存数据的关系，文件与文件也有关系。因此关系型数据库是最常用的，它的特性是数据操作慢和安全性高。

MySQL也是学校所教的数据库语言之一，我们常用Navicat Lite对MySQL进行操作，因为Navicat Lite可以简化数据库操作，它还可形象生动的展示数据的总体内容，并且还可以直接通过他操作数据库，例如：添加表和字段值等、修改表、表名和表中内容等。因此身为初学者的我们，通常都会使用它来进行数据库实现，这是一个很方便的工具。

1.1.4 IDEA简介

IDEA 全称IntelliJ IDEA，是用于java开发的集成环境（也可用于其他语言），它支持代码的提示、JUnit单元测试。IDEA的开发效率不是一般的快，尤其是对于java语言，它容纳了开发者在开发中需要的绝大部分实用功能，可以让开发者脱离鼠标，做他们想做的事，尽最大的能力加快开发效率，。Idea的优点如下：

(1)对于JUnit单元测试的完美支持

(2)xml语法提示支持：对于市面上几乎所有的流行框架的xml文件都支持语法提示，语法高亮。

(3)不需要任何的插件，完全支持JSP。

（4）使用者众多，意味着社区它的开发者社区活跃，遇到与之相关的问题可以很快解决

(5)Idea 对项目版本控制的支持是绝无仅有的，无论是SVN 还是Git，亦或是GitHub都支持的非常好。尤其是对Git的支持，图形化的操作页面，会让你省却很多命令的烦恼。让普通的程序开发者在编程过程中直接在IDEA中就能完成代码的提交、拉取，查看版本控制服务器内容等等。

（6）Idea 内嵌了Maven插件，而且也内嵌了Maven，当然也可以使用外部的Maven。

（7）Idea 会把整个工程的文件都做Index，所以搜索文件什么，都是毫秒级别就出来的。

（8）我们还可以有选择地重新编译和加载部分字节码文件，再加上 IDEA 强大的 DEBUG 功能，可以很方便地帮助我们进行代码的调试工作

1.1.5 J2EE

J2EE是一种中间件的基础架构，由于它的存在，开发者只用花费精力去编写业务逻辑的代码，至于其他的系统深层问题，比如内存管理，gc，线程管理等都由系统本身完成。

J2EE：它将应用开发划分为多个不同的层次结构，并在每一个层上定义组件，非常典型的J2EE包括客户层，网页显示层，业务逻辑层，信息管理层。

**1.2 开发环境配置**

1.2.1 JDK的安装与配置

（1）从官网下载JDK1.8并安装。

（2）进行JDK1.8的环境变量的设置：

JAVA\_HOME的值改为JDK1.8的安装根路径Path设为：%JAVA\_HOME%\bin;

CLASSPATH属性设为：.;%JAVA\_HOME%\lib; 设置完成后在cmd中输入java出现java的版本信息

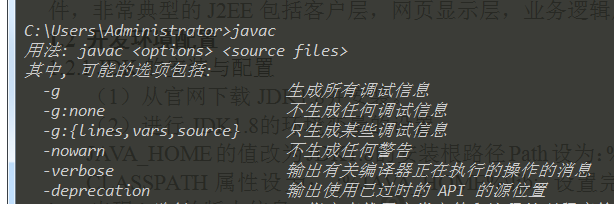


图2-2 安装成功图

**第3章 需求分析**

**3.1 可行性分析**

3.1.1经济可行性分析

缩短了供货周期，压缩了库存，裁减了人员，就可以依据缩短时间的天数和库存物资的减少，以及裁减人员的工资使原材料和劳动力资源得到更合理的利用，从而降低了成本，避免了过多的消耗，减少管理费用的支出等等。所以此项目在经济上是可行的。

3.1.2技术可行性分析

Java是一种优秀的高级编程语言，无论是在windows系统还是linux系统，他都可以进行代码的开发，它最大的优点就是与平台无关。Jsp将前端页面逻辑和前端的网页显示相互分离开来，支持组件的复用，使基于 Java Web的项目的开发变得迅速和容易。目前J2EE技术已经非常成熟，结合jsp，servlet及相关的关系数据库技术，完全可以达到并实现相关功能。

3.1.3社会可行性分析

政府鼓励扶持小型企业发展，网店成为许多小公司的重要选择，加快农村流通体系建设扩大网上消费群体。无论是国家还是当地政府都支持传统企业进军电子商务，出台了一系列优惠政策，希望以网络销售带动门店销售，并对网购平台诚信体系，支付手段等方面予以支持和政策指导，以推进实体市场和网上市场的结合。对于消费用户来说，会根据消费用户的调查报告设计系统会考虑到人性化界面和操作的方便性。所以社会可行性方面没有问题。

**3.3 结构化分析**

3.3.1 数据流图

数据流图是一种描述系统逻辑模型的工具，用少数符号反映信息在系统中的流动，处理存储情况。本系统的数据流图如下所示：

首先是顶层图，它主要说明了系统运行的边界。

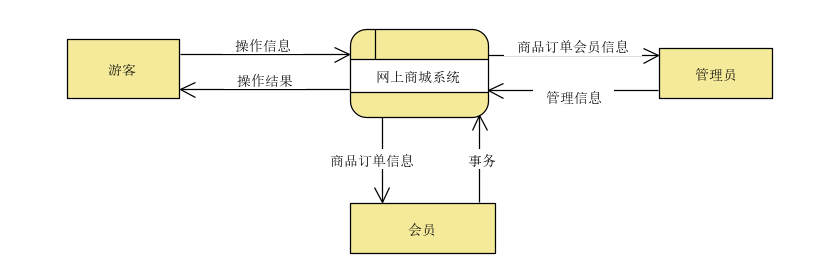


图3-1顶层图数据流图

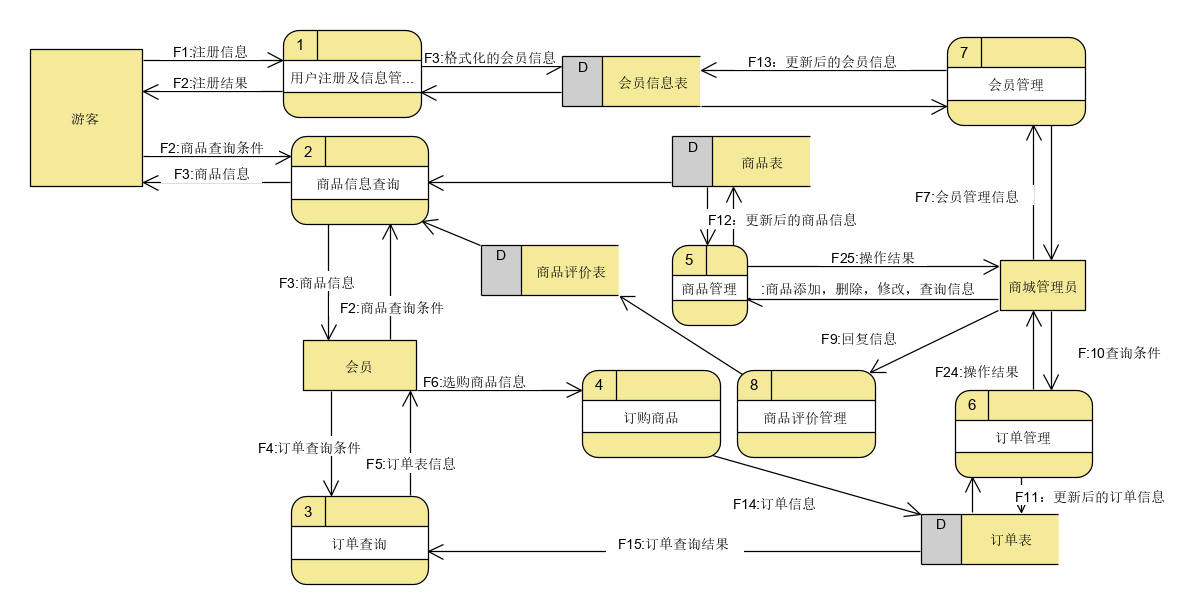


图3-2一层数据流图

第2层数据流图是对上层的详细说明和拆分，更清楚的描述了系统数据的处理情况。如下所示：

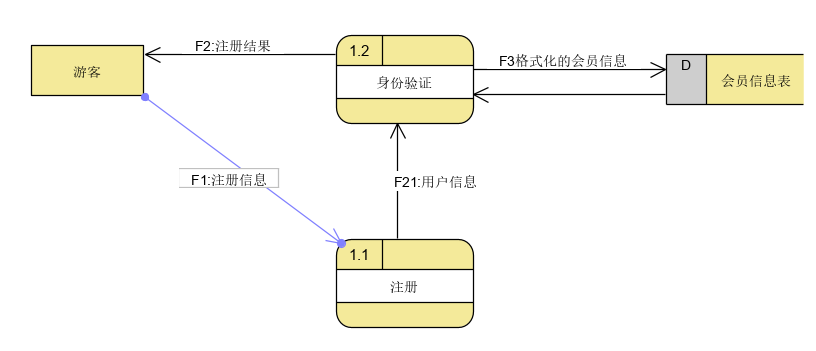


图3- 3用户注册及信息管理二层数据流图

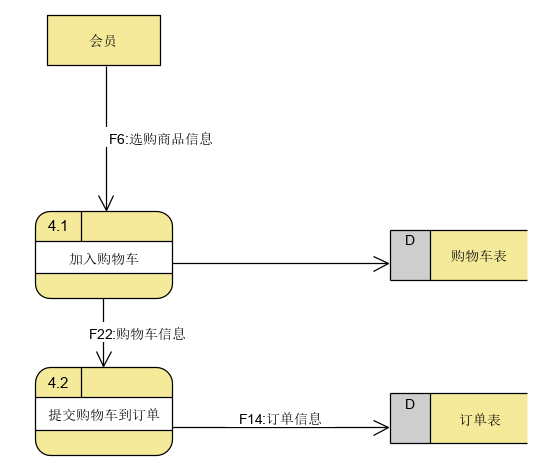


图3-4订购商品二层数据流图

图3-4

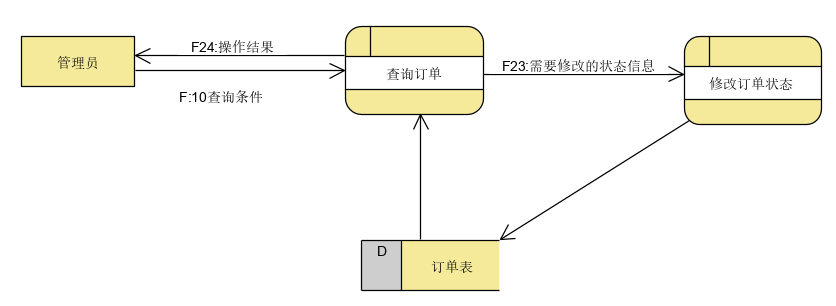


图3- 5订单管理二层数据流图

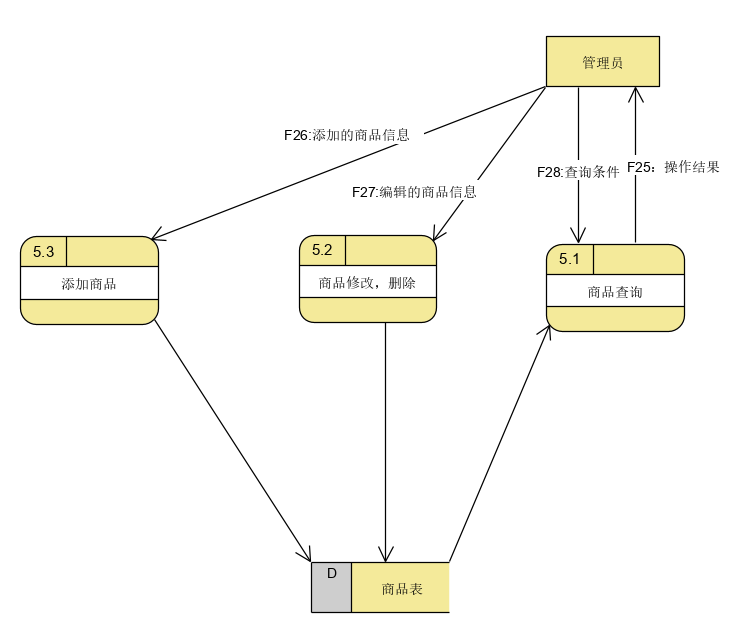


图3- 6商品管理二层数据流图

3.3.2 数据字典

以下列出本系统的主要数据字典条目。

（1）数据结构

数据结构名：用户

含义：系统的的主体数据结构，定义了一个用户的有关信息。

组成：用户id、用户名、密码、姓名、邮箱、电话号码、性别、生日。

数据结构名：管理员

含义：系统的的主体数据结构，定义了一个管理员的用户名与密码

组成：管理员用户名、密码

（2）数据项

1)用户id

数据项名：用户id

数据项含义：唯一标识每一个用户

别名：用户编号

数据类型：字符型

长度：255

取值范围：随机

2)用户名

数据项名：用户名

数据项含义：用户的账号名称

别名：用户名称

数据类型：字符型

长度：11

取值含义：手机号

3）密码

数据项名：密码

数据项含义：账户密码

别名：口令

数据类型：字符型

长度：15

4）姓名

数据项名：姓名

数据项含义：用户的真实名字

别名：名字

数据类型：字符型

长度：15

取值含义：用户名字

5）邮箱

数据项名：邮箱

数据项含义：用户的邮箱

别名：电子邮箱

数据类型：字符型

长度：20

6）电话号码

数据项名：电话号码

数据项含义：用户的手机号

别名：手机号

数据类型：字符型

长度：11

7）性别

数据项名：性别

数据项含义：用户的性别

数据类型：字符型

长度：2

8）生日

数据项名：生日

数据项含义：用户的出生日期

别名：出生日期

数据类型：字符型

长度：255

9）管理员用户名

数据项名：管理员用户名

数据项含义：管理员的账号

别名：管理员的账号

数据类型：字符型

长度：15

（3）数据流定义：

1）名称：注册信息

编号：F1

简述：未注册的用户提供自己的信息注册账号

数据来源：顾客

数据去向：会员信息表

数据组成：用户名+密码+姓名+电话+邮箱+性别+生日

2）名称：商品查询条件

编号：F3

简述：用户查询商品输入的查询条件

数据来源：用户

数据去向：商品查询

数据组成：商品名

4）名称：订单表信息

编号：F5

简述：所属这个用户的所有订单信息

数据来源：订单表

数据去向：用户

数据组成：用户名+总金额+收货信息

4）名称：订购信息

编号：F6

简述：用户选择的商品的信息

数据来源：用户

数据去向：购物车表

数据组成：用户名+商品id+数量

5）名称：会员管理信息

编号：F7

简述：管理员对会员信息的修改或删除

数据来源：管理员

数据去向：会员管理

数据组成：会员信息表除用户名之外的各项信息

6）名称：查询条件

编号：F10

简述：管理员搜索订单输入的条件

数据来源：管理员

数据去向：订单表

数据组成：订单号

7）名称：订单信息

编号：F14

简述：用户提交的订单信息

数据来源：用户

数据去向：订单表

数据组成：购物车信息+收货信息+用户名

8）名称：购物车信息

编号：F22

简述：用户提交购物车到订单

数据来源：用户

数据去向：订单表

数据组成：用户名+数量+总金额+商品id

9）名称：添加的商品信息

编号：F26

简述：管理员添加商品

数据来源：管理员

数据去向：商品表

数据组成：一个商品的完整信息

10）名称：查询条件

编号：F28

简述：管理员搜索商品输入的条件

数据来源：管理员

数据去向：商品表

数据组成：商品的名字

（4）数据存储

1）数据存储文件：会员信息表

文件组成：用户id,用户名，密码，姓名，电话，邮箱，性别，生日

说明：记录会员所有信息

流入的数据流：会员信息

流出的数据流：会员信息

2）数据存储文件：商品表

文件组成：商品id,商品名，商品图片，商品描述，商品价格，商品上架 日期，商品库存，商品销量，商品类别

说明：记录商品信息

流入的数据流：商品信息

流出的数据流：商品信息

3）数据存储文件：订单表

文件组成：订单id,用户名，收货人，收货地址，收货人电话，下单时间， 订单状态。

说明：记录所有的订单信息

流入的数据流：订单信息

流出的数据流：订单信息

4）订单详细记录表

数据存储名：订单详细记录

说明：记录每个订单的详细信息

流入的数据流：订单信息

流出的数据流：订单信息

组成：订单子项编号、数量、总计、所属订单号、商品id

5）数据存储文件：购物车表

文件组成：购物车id,用户名，商品id，数量

说明：记录个人的购物车信息

流入的数据流：购物车信息

流出的数据流：购物车信息

6）数据存储文件：收货信息表

文件组成： 用户名，收货人，收货地址，收货人电话

说明：记录个人的收货信息

（5）处理过程

1）用户注册

处理过程名：注册

说明：无账号的游客根据个人信息注册账号

输入数据流：F1

输出数据流：F21

处理：游客根据自己的信息注册账号，密码成为会员

2）商品信息查询

处理过程名：商品信息查询

说明：用户根据商品名查询商品

输入数据流：F2

输出数据流：F3

处理：系统根据输入的信息查询对应的商品给用户浏览

3）加入购物车

处理过程名：加入购物车

说明：用户挑选商品加入购物车

输入数据流：F6

输出数据流：F22

处理：会员将心仪的商品添加到自己的购物车里

4）订单查询

处理过程名：订单查询

说明：根据用户名查询该用户所有的订单信息

输入数据流：F4

输出数据流：F5

处理：系统通过登录的用户名查询该用户的订单信息提供给用户

5）添加商品

处理过程名：添加商品

说明：系统管理员通过后台添加新商品

输入数据流：F26

处理：系统通过后台添加新商品到数据库中

6）编辑商品

处理过程名：编辑商品

说明：系统管理员通过后台编辑已存在的商品信息

输入数据流：F27

处理：系统管理员通过后台修改商品的各种信息

7）修改订单状态

处理过程名：修改订单状态

说明：系统管理员根据订单状态处理订单

输入数据流：F23

处理：系统管理员通过后台修改订单的状态（已付款的订单去发货）

**3.2 系统需求分析**

通过对软件工程的理论的学习，系统的相关分析和设计是一个项目工程开发能否成功的重点，正确获取系统用户的需求为项目开发提供了基础。此网上购物系统的重点是提供产品展示，以及商品订购，是宣传自己商店的商品给用户，让客户能够通过自己的需要来选择要购买的商品：

因为传统购物存在一定的局限性和问题，所以经过总结，有下面几个需求：

顾客需求：

（1）普通用户可以通过系统浏览商品；

（2）普通用户可以通过系统搜索框按商品名称搜索商品

（3）普通用户可以注册成为会员；用户可以注册成为会员，并可以登陆进入网站拥有更多功能。

（4）会员可以浏览和检索商品；

（5）会员可以加入购物车；购物车必须会员才可以操作，

（6）会员可以浏览并且管理自己的购物车，根据自己的需求来添加商品或删除商品；

（8）会员可以浏览并且管理自己的订单；

（9）会员可以管理自己的账户信息：修改密码；

商场管理员的需求：

商场管理员可以对系统公告进行修改，发布。

商场管理员可以对顾客，商品，订单进行管理和维护。

购物车的需求：

查看购物车的内容。

清空所有商品的功能。

全选和选部分商品提交。

系统优势，方便用户，用户只用短短的几步便可以买到心仪的商品，极大限度的提高了网上商城的运营效率，。搜索功能使用户在极短的时间便可以查询到自己想要的商品，节省了选购时间，在提高用户体验的同时，也降低了服务器压力。

**第4章 系统设计**

**4.1 系统模块设计**

在软件系统设计中，模块一般情况下都是按对应的功能划分的通常称为功能模块。因为这样做可以保证系统的数据稳定性，以及降低模块之间的耦合度。合理的功能模块划分不但能够提高开发者的开发效率，并且对于一些相似的模块组件可以复用，从而达到了减少开发系统时间的目的。而且这样做也有利于系统后期的维护。本系统分为两各部分，即前台部分主要面向顾客和后台部分面向管理员。

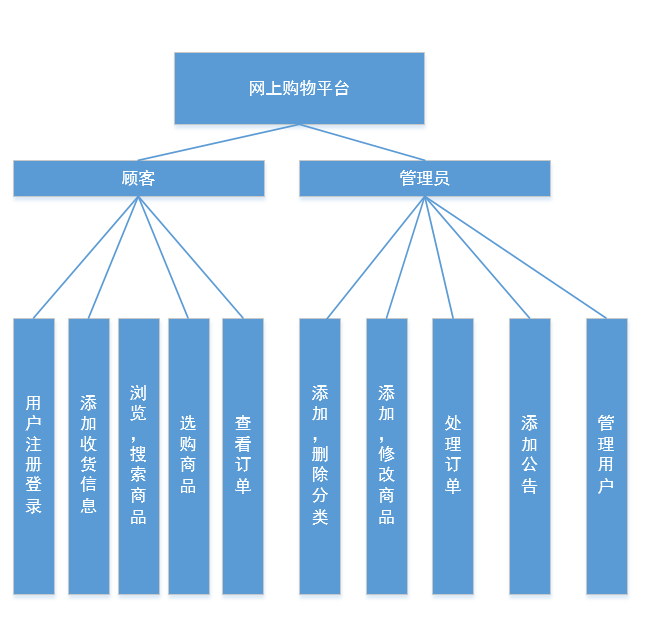


图4-1

4.1.1前台面向用户的功能模块设计

前台模块主要为用户提供系统账号的注册，登录，浏览商品信息，购买商品，填写并提交订单的服务。这样，将前台部分在细分为用户管理模块，商品查询模块，购物车模块，添加收货信息模块，订单模块。顾名思义，这些模块的名称基本解释了他们的功能。下面详细说明一下：

用户管理模块：

为了方便网站的管理，必须由一套完整的用户管理体系，该网站用户管理模块主要实现用户的注册，登录，修改密码等功能。

购物车模块

在超市购物，可以根据自己的需要将很多商品挑选至购物车或购物篮中，然后到收银台结账。而在网上商城中，通常都会采用一种被称为购物车的技术来模拟显示生活，购物车应该一直和用户绑定的，无论什么时候用户在登陆之后都可以看到他原来添加的商品，在网上商城中所选商品必须通过购物车进行暂存，然后生成订单。可以随时添加查看修改清空购物车中的内容，他都会持久化数据到数据库中

商品搜索模块

用户根据商品的名称可以搜索到所需要的商品信息。

添加收货信息模块

用户在登录之后添加个人的收货地址的相关信息，也可以删除收货地址。

订单模块：

用户在购物车中选择自己需要商品提交订单，不需要的可以留在购物车里，可以查看订单状态，确认收货等等。

4.1.2 后台模块的功能模块设计

后台部分的功能是对系统里的资源进行管理和维护，以后台的业务流程和前台部分的设计为基础，基于用户的需求分析来开发后台系统管理模块应具有的相关功能。后台管理模块的各个功能如下：

管理员身份验证模块：

为系统管理员提供一个后台系统的接口，该接口的功能是验证管理员的身份。输入登录账号和密码，如果通过验证则跳转后台主页，否则提示错误。

商品管理模块：

向商品表插入商品信息，也可以修改信息，也可以删除，向商品类别表添加商品的类别，删除商品的类别

处理订单模块

管理员对用户订单的执行，修改订单状态，查看及其对已完成的订单的删除。

用户信息管理模块

管理顾客用户的信息

公告管理模块

管理系统公告。

**4.2 系统流程设计**

本系统分为前台购物流程和后台订单处理流程。

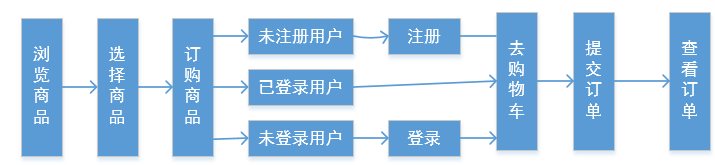


图4-2前台购物流程图

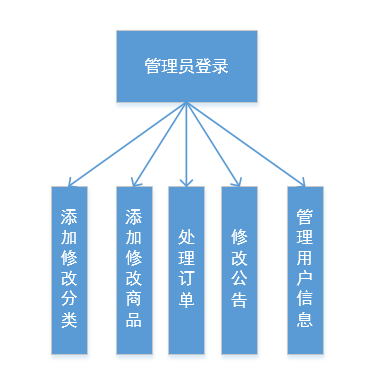


图4-3后台管理流程图

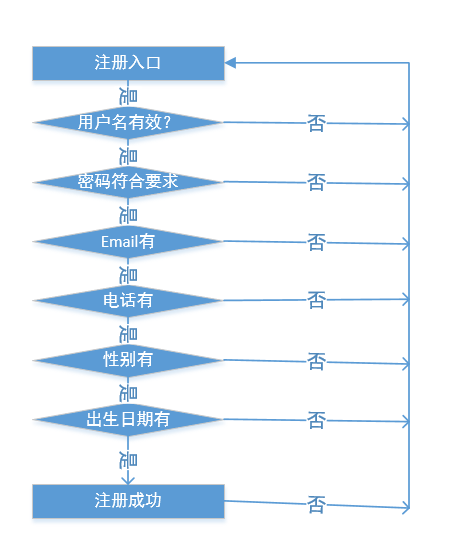


图4-4注册功能流程图

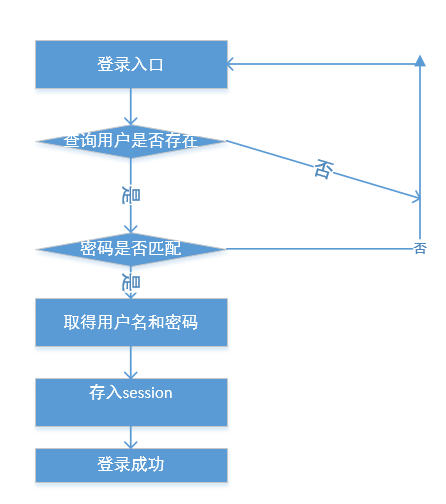


图4-5登录流程图

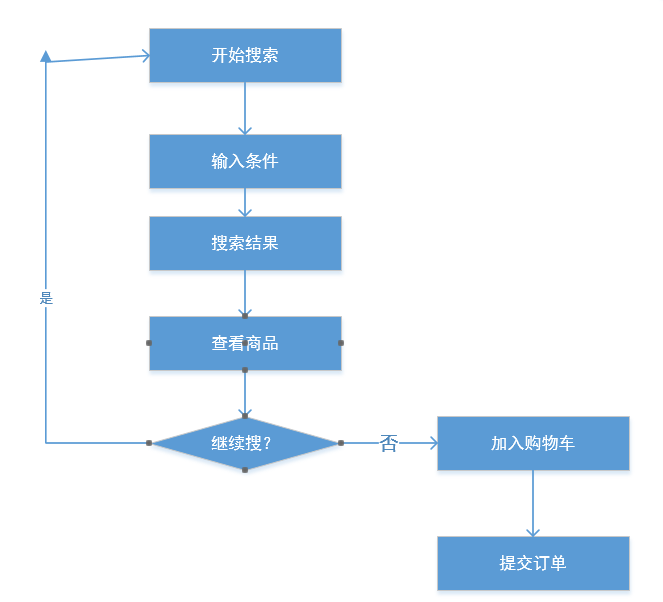


图4-6商品搜索购买流程图

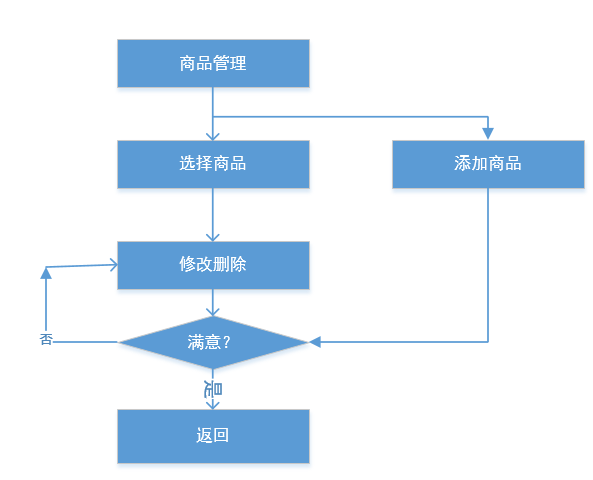


图4-7商品管理流程图

**4.3 数据库设计**

4.3.1数据库概要设计

概念结构是许多模型的基础，相对于其他模型，它更独立与机器，更抽象，更稳定。概念结构极其重要，在实际中经常使用ER图来描述概念模型。

实体、属性、联系这三个是E-R模型的组成元素。E-R模型用E-R图来表示。实体是用户工作环境中所涉及的事务，属性是对实体特征的描述。

2.属性是实体的性质。用椭圆框表示，与实体之间用一条线相连表的主码是关键属性，关键属性项加下划线。

3.各子系统模块中主键相同的字段之间存在着相互关联的关系。

4.在程序中实现对他们的完整性和一致性控制。

从数据需求分析中得出系统的实体属性图，遵循三范式原则，对实体之间的依赖关系进行了整合，得出系统E-R图：

说明：图表中用矩形表示实体。实体与实体的作用关系用菱形表示，用无向边把菱形与有关实体连接，在边上标明联系的类型。实体的属性用椭圆表示，并用无向边把实体与属性联系起来。

实体分析：

根据网上商城的总体设计，我们定义了网上商城整个系统分为三个模块。由此我们可以首先确定该系统中的实体：管理员、会员、商品、订单、地址信息，公告共6个实体。

管理员可以管理会员、管理商品、管理订单，一个管理员可以管理多个会员，可以管理多个商品，可以管理多个订单信息。

一个订单包括多个商品信息。一个商品可以在多个订单中。

一个会员可以有多个订单，一个订单只能被一个会员所有。

一个商品可以被多个用户购买，一个用户可以买多个商品，

所以其实体属性图，和ER图如下：

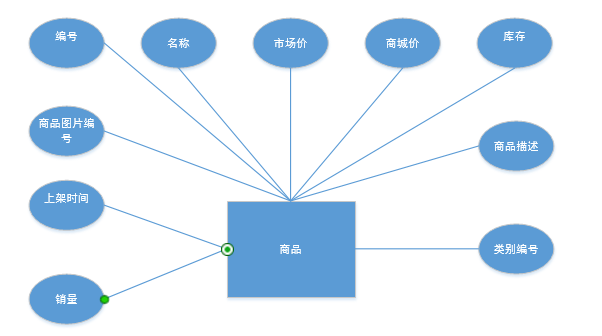


图4-8商品实体属性图

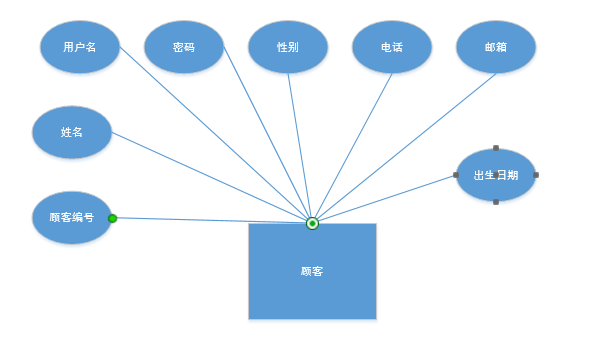


图4-9顾客实体属性图

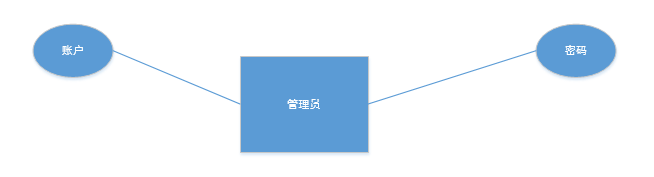


图4-10管理员实体属性图

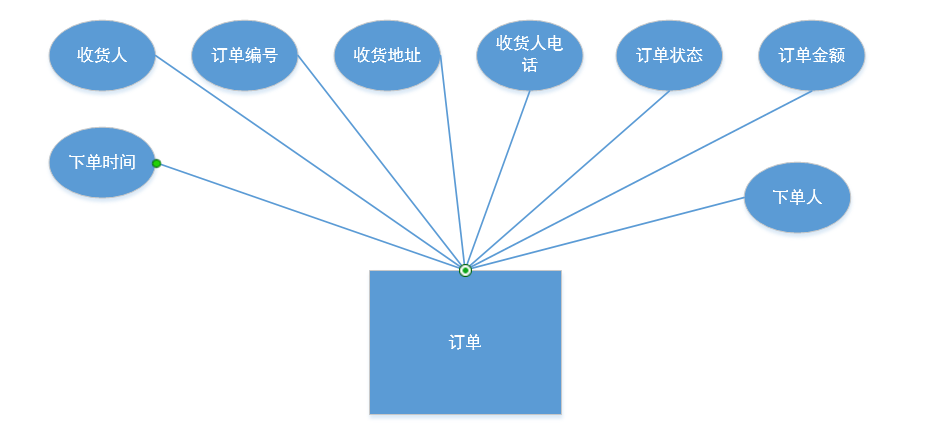


图4-11订单实体属性图

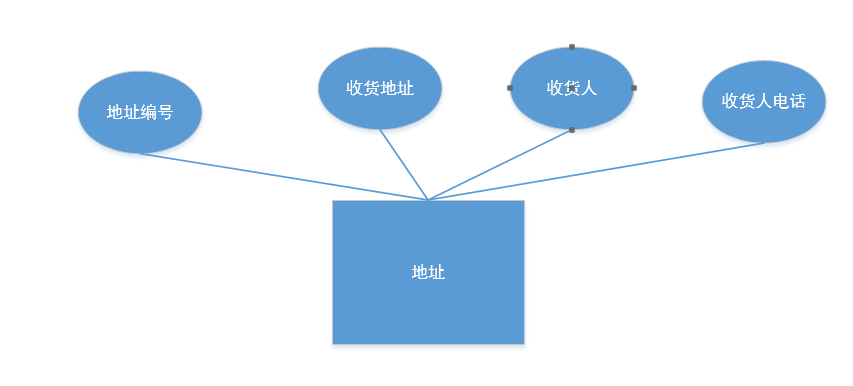


图4-12收货地址实体属性图

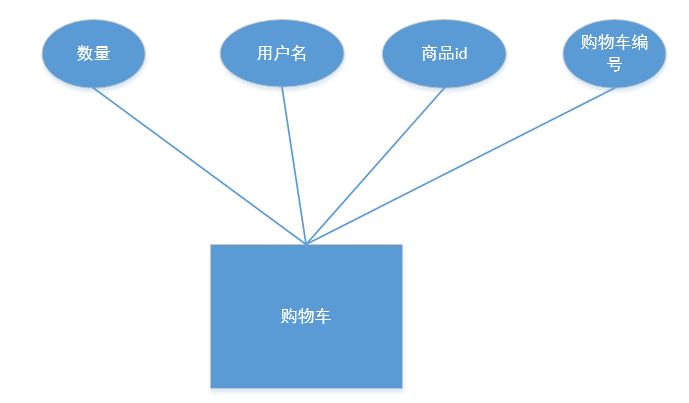


图4-13购物车实体属性图

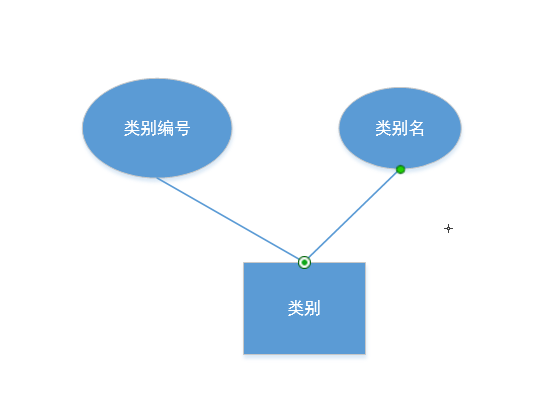


图4-14商品类别实体属性图

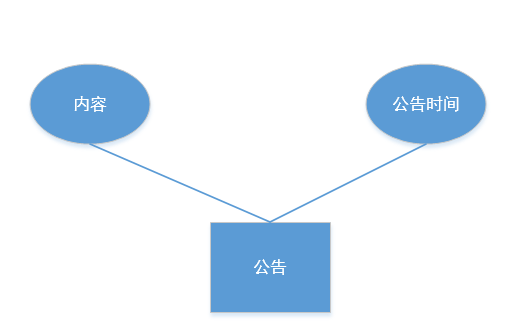


图4-15系统公告实体属性图

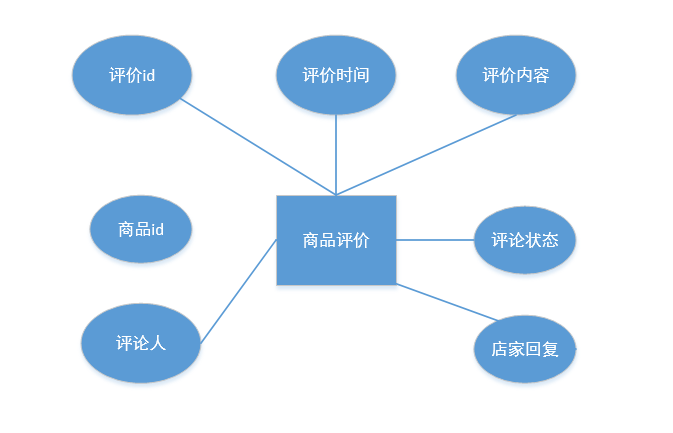


图4-16商品评价实体关系图

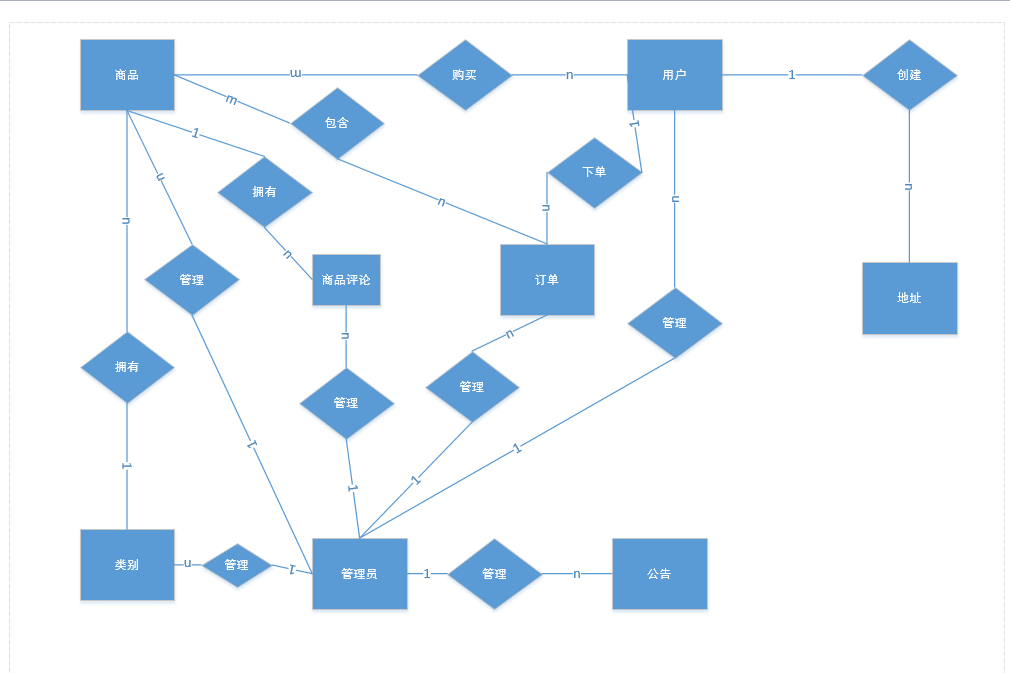


图4-17 ER图

其中，各个实体的属性如下：

用户信息实体的主要属性如下：

用户id ，用户名 ，密码 ，姓名 ，电话 ，邮箱， 性别， 生日；

商品分类实体的主要属性如下：

分类id ，分类名

商品实体的主要属性如下：

商品id ，商品名， 商品图片 ，商品价格， 商品描述 ，商品库存， 商品销量 商品分类 ，商品上架时间

订单实体的主要属性如下：

订单号 ，订单时间 ，订单价格 ，订单收货人， 订单收货地址，订单收货人电话，订单状态，订单拥有人

购物车实体的主要属性如下：

购物车id，数量，商品id，用户名

商品评论实体的主要属性

评论id,评论人，评论内容，评论时间，回复内容，评论状态,商品id

4.3.2数据库表

在本系统的数据库中，根据需求建立了如下基本表，具体如下

（1）用户表

User(uid,username,password,name.email,telephone,sex,birthday)

（2）收货信息表

Adress(aid,address,phone,person)

（3）商品表

Product（pid,pimage,pname,pdesc,pdate,pmarket\_price,pcount,is\_latest,cid）

（4）订单表

Orders（oid,address,phone,person,ordertime,state,total,username）

（5）管理员表

admin（acount,pwd）

（6）分类表

Category（cid,cname）

（7）购物车表

Car（cid,count,pid,uid）

（8）订单条目项表

Orderitem（itemid,oid,pid,subtotal,count）

（9）评论表

Goodscomments（gid,comment,username,pid,state,anser）

根据以上分析数据库表结构如下：

数据库名：store

表4-1 表名user

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 列名 | 数据类型 | 长度 | 主键 | 允许空 | 默认值 | 说明 |
| 1 | Uid | varchar | 255 | 是 | 不 |  | 用户的唯一id |
| 2 | username | varchar | 255 |  | 不 |  | 登录名字 |
| 3 | password | varchar | 255 |  | 不 |  | 用户密码 |
| 4 | name | varchar | 255 |  |  |  | 用户姓名 |
| 5 | email | varchar | 255 |  |  |  | 邮箱 |
| 6 | telephone | varchar | 255 |  |  |  | 电话 |
| 7 | sex | varchar | 255 |  |  |  | 性别 |
| 8 | birthday | varchar | 255 |  |  |  | 生日 |

表4-2 表名product

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 列名 | 数据类型 | 长度 | 主键 | 允许空 | 默认值 | 说明 |
| 1 | pid | varchar | 255 | 是 | 不 |  | 商品唯一id |
| 2 | pname | varchar | 255 |  | 不 |  | 商品名称 |
| 3 | pimage | varchar | 255 |  | 不 |  | 商品图片 |
| 4 | pdate | varchar | 255 |  | 不 |  | 上架日期 |
| 5 | pdesc | varchar | 255 |  |  |  | 商品描述 |
| 6 | Shop\_price | double | 0 |  | 不 |  | 商品价格 |
| 7 | Market\_price | double | 0 |  | 不 |  | 超市价格 |
| 8 | pcount | int | 11 |  | 不 |  | 库存 |
| 9 | Is\_latest | int | 11 |  | 不 |  | 销量 |
| 10 | cid | varchar | 255 |  | 不 |  | 外键，受类别表id约束 |

表4-3 表名：orders

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 列名 | 数据类型 | 长度 | 主键 | 允许空 | 默认值 | 说明 |
| 1 | Oid | varchar | 255 | 是 | 不 |  | 订单唯一id |
| 2 | address | varchar | 255 |  | 不 |  | 该订单地址 |
| 3 | name | varchar | 255 |  | 不 |  | 收件人 |
| 4 | telephone | varchar | 255 |  | 不 |  | 电话 |
| 5 | ordertime | data | 0 |  | 不 |  | 订单时间 |
| 6 | state | int | 11 |  | 不 |  | 订单状态 |
| 7 | total | double | 0 |  | 不 |  | 总金额 |
| 8 | username | varchar | 255 |  | 不 |  | 外键受user表用户名约束 |

表4- 4 表名：orderitem

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 列名 | 数据类型 | 长度 | 主键 | 允许空 | 默认值 | 说明 |
| 1 | Itemid | varchar | 255 | 是 | 不 |  | 某个订单中的一个购物项 |
| 2 | count | int | 11 |  | 不 |  | 数量 |
| 3 | subtotal | double | 0 |  | 不 |  | 小计 |
| 4 | oid | varchar | 255 |  | 不 |  | 外键受订单表oid约束 |
| 5 | pid | varchar | 255 |  | 不 |  | 外键受商品表oid约束 |

表4- 5 表名：car

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 列名 | 数据类型 | 长度 | 主键 | 允许空 | 默认值 | 说明 |
| 1 | cid | varchar | 255 | 是 | 不 |  | 购物车唯一id |
| 2 | uid | varchar | 255 |  | 不 |  | 外键 |
| 3 | Pid | varchar | 255 |  | 不 |  | 外键商品id |
| 4 | count | int | 11 |  | 不 |  | 数量 |

表4- 6 表名：admin

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 列名 | 数据类型 | 长度 | 主键 | 允许空 | 默认值 | 说明 |
| 1 | Acount | varchar | 255 | 是 | 不 |  | 管理员账户 |
| 2 | pwd | varchar | 255 |  | 不 |  | 管理员密码 |

表4- 7 表名：notice

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 列名 | 数据类型 | 长度 | 主键 | 允许空 | 默认值 | 说明 |
| 1 | nid | varchar | 255 | 是 | 不 |  | 公告id |
| 2 | notice | varchar | 255 |  | 不 |  | 公告内容 |
| 3 | time | date | 0 |  | 不 |  | 增加公告的时间 |

表4- 8 表名：category

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 列名 | 数据类型 | 长度 | 主键 | 允许空 | 默认值 | 说明 |
| 1 | cid | varchar | 255 | 是 | 不 |  | 分类的id |
| 2 | cname | varchar | 255 |  | 不 |  | 分类名称 |

表4- 9 表名：goodscomments

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 列名 | 数据类型 | 长度 | 主键 | 允许空 | 默认值 | 说明 |
| 1 | gid | int |  | 是 | 否 |  | 主键 |
| 2 | username | vachar | 255 |  | 否 |  | 外键用户名 |
| 3 | comment | vachar | 255 |  | 否 |  |  |
| 4 | time | date |  |  | 否 |  |  |
| 5 | pid | vachar | 255 |  | 否 |  | 外键商品id |
| 6 | anser | vachar | 255 |  |  |  |  |
| 7 | state | int |  |  | 否 | 0 |  |

**4.4本章小结**

本章完成了网上商城系统所需数据库的设计与实现。按照数据库开发的流程，从需求出发到数据库的概念设计，再到数据的逻辑设计，最后在mysql数据库中完成表的基本创建。

**第5章 系统实现**

**5.1前台设计**

网站前台部分提供给用户使用，是系统对外的窗口。主要包括：

网站首页、查看公告，产品展示，用户登录注册，购买商品，查看订单

由于篇幅有限，本文只对部分模块的实现进行描述。

5.1.1用户注册

游客通过注册页面填写个人信息进行注册，每项必要信息都必须符合要求才能注册成功，注册成功即成为会员。否则注册失败。

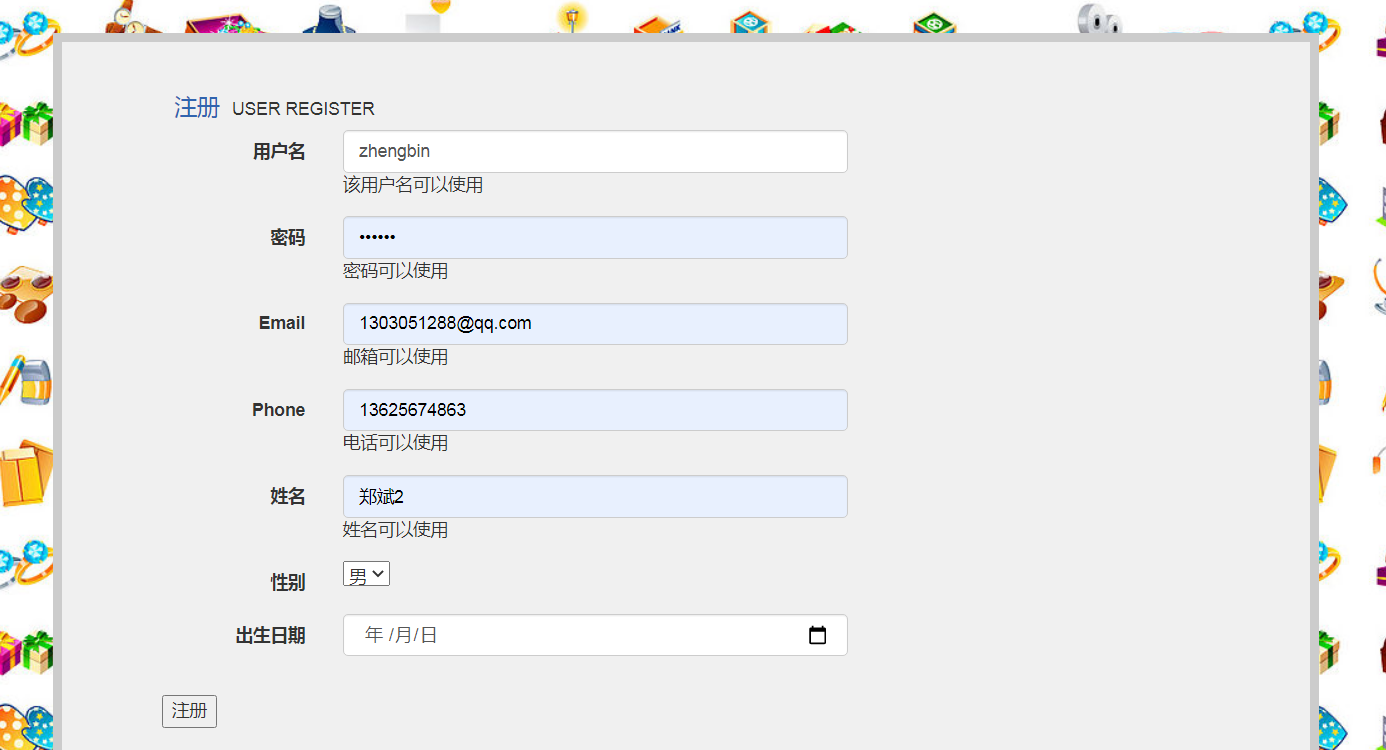


图5-1用户注册

### 5.1.2 用户登录

用户登录页面主要用来接收用户输入的用户名和密码，并更新用户在网站中的状态信息。只有当用户名和密码都正确才能登陆成功，如果输入不正确，系统会显示登录失败。会员登录窗口的运行结果如图所示。

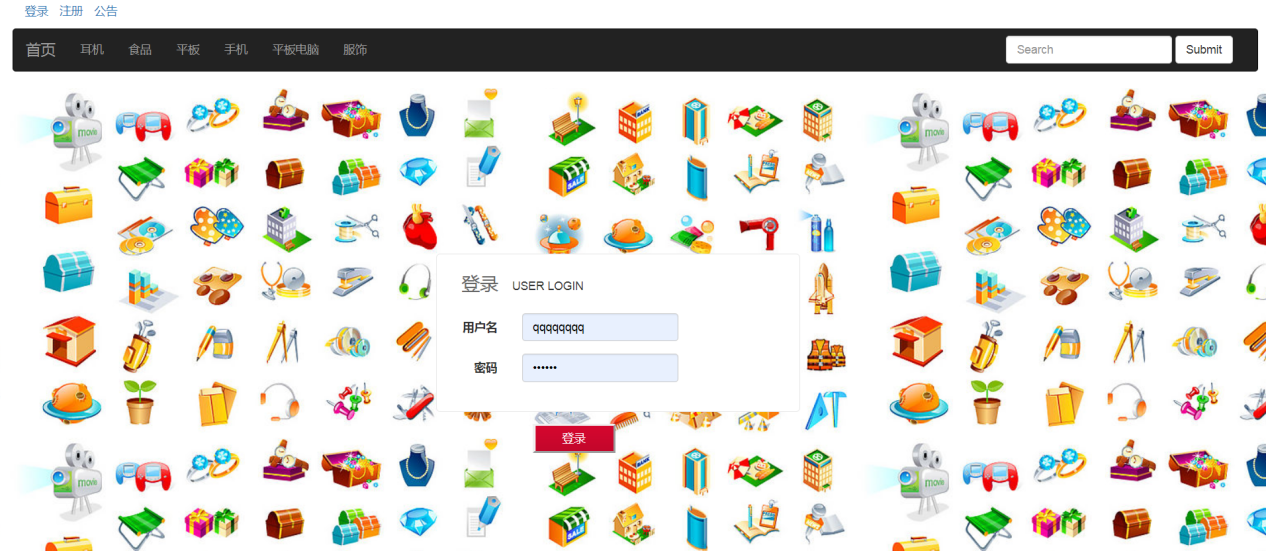


图5-2用户登录

*//登陆验证*

**boolean** isSuccess = **service**.findUserByUsernameAndPassword(user);

ModelAndView mav = **new** ModelAndView();

**if**(isSuccess) {

*// 成功*

mav.addObject(**"msg"**, **"登录成功！，请前往首页选购"**);

*// 添加到session*

session.setAttribute(**"user"**, user);

mav.setViewName(**"index"**);

**return** mav;

}**else** {

*// 失败*

mav.addObject(**"msg"**,**"登录失败"**);

mav.setViewName(**"msg"**);

**return** mav;

}

}

Java代码部分

@RequestMapping(**"/doLogin"**)*//登陆验证***public** ModelAndView doLogin(User user,HttpSession session) {  
  
 **boolean** isSuccess = **service**.findUserByUsernameAndPassword(user);  
 ModelAndView mav = **new** ModelAndView();  
 **if**(isSuccess) {  
 *// 成功* mav.addObject(**"msg"**, **"登录成功！，请前往首页选购"**);  
 *// 添加到session* session.setAttribute(**"user"**, user);  
 mav.setViewName(**"index"**);  
 **return** mav;  
 }**else** {  
 *// 失败* mav.addObject(**"msg"**,**"登录失败"**);  
 mav.setViewName(**"msg"**);  
 **return** mav;  
 }  
}

### 5.1.3 商品浏览

用户在商品显示页面对具体商品进行详细了解，可以查看商品的评论等，查看已买过此商品的顾客的评价。并在商品显示页面对商品进行订购，已注册登录的会员可以加入购物车并提交订单。显示主页的热门和最新商品，热门商品是根据销量的数量来排列的，最新商品是按照上架时间来排列显示的。

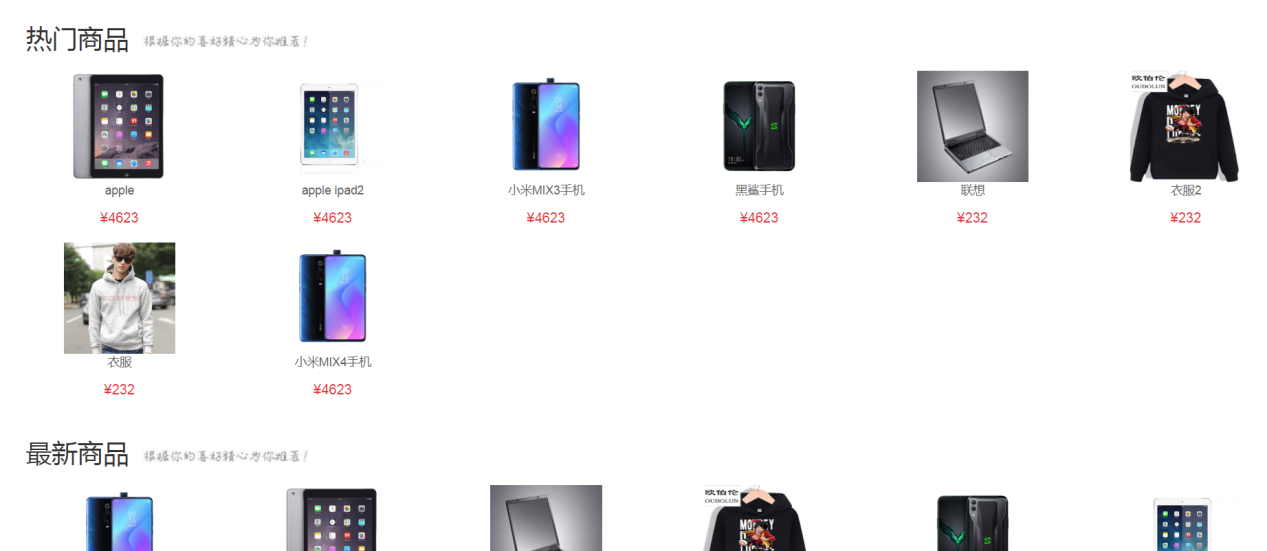


图5-3首页

*//发送请求，获取最新商品信息*

**$**.getJSON(**"${**pageContext.request.contextPath**}/latestProduct.action"**,**function**(result){

**$**(result).each(**function**(index,element) {

**$**(**"#latest"**).append(**"<div class='col-md-2' ><a href='${**pageContext.request.contextPath**}/infoProduct.action?pid="**+element.**pid**+**"'><imgsrc='${**pageContext.request.contextPath**}/pic/"**+element.pimage+**"' ></a><p><ahref='${**pageContext.request.contextPath**}/infoProduct.action?pid="**+element.**pid**+**">"**+element.pname+**"</a></p>**+element.shop\_price+**"</font></p></div>"**)

})

})

})

单个商品的详细信息页面：



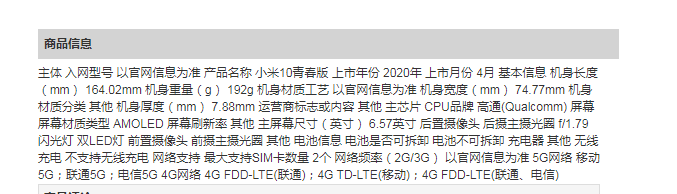


图5-4 商品信息图

未注册的用户想要登录系统必须要注册账户，只有所有的数据项符合要求才能注册成功，只要有一项不符合，便不能注册成功。



图5-5注册

用户注册模块

主要实现代码：

<**script**>

**$**(**"#inputPassword"**).blur(**function** () {

**var** reg=/^[0-9]{6}$/;

**if**(!reg.test(**$**(**"#inputPassword"**).val())){

**$**(**"#ptext"**).text(**"密码需要6个数字"**);

}**else** {

**$**(**"#ptext"**).text(**"密码可以使用"**);

}

})

**$**(**"#inputEmail3"**).blur(**function** () {

**if**(!reg.test(**$**(**"#inputEmail3"**).val())){

**$**(**"#pemail"**).text(**"邮箱格式不对"**);

}**else** {

**$**(**"#pemail"**).text(**"邮箱可以使用"**);

}

})

**$**(**"#inputphone"**).blur(**function** () {

**var** reg= /^1[3456789]\d{9}$/;

**if**(!reg.test(**$**(**"#inputphone"**).val())){

**$**(**"#pphone"**).text(**"电话格式不对"**);

}**else** {

**$**(**"#pphone"**).text(**"电话可以使用"**);

}

})

**$**(**"#name"**).blur(**function** () {

**if**(**$**(**"#name"**).val()!=**""**&&**$**(**"#name"**).val().trim()!=**""**){

**$**(**"#pname"**).text(**"姓名可以使用"**);

}**else** {

**$**(**"#pname"**).text(**"姓名不能为空"**);

}

})

**$**(**"#username"**).blur(**function** () {

**var** reg=/^[A-z]{8}$/;

**if**(reg.test(**$**(**"#username"**).val())){

**$**.**ajax**({

**type**:**"POST"**,

**url**:**"${**pageContext.request.contextPath**}/have.action"**,

**data**:{**"username"**:**$**(**"#username"**).val()},

success:**function**(msg){

**if**(msg == **"yes"**){

**$**(**"#username"**).text(**"该用户名已存在"**);

}**else if**(msg == **"no"**){

**$**(**"#username"**).text(**"该用户名可以使用"**);

}

}

})

}**else**{

**$**(**"#username"**).text(**"用户名需要8个字母"**);

}

})

**function** *m*() {

**if**(**$**(**"#username"**).text()==**"该用户名可以使用"**&&**$**(**"#ptext"**).text()==**"密码可以使用"**&&**$**(**"#pemail"**).text()==**"邮箱可以使用"**&&**$**(**"#pphone"**).text()==**"电话可以使用"**&&**$**(**"#pname"**).text()==**"姓名可以使用"**){

**$**(**".form-horizontal"**).submit();

}**else** {

*alert*(**"请检查所有注册项符合要求！"**)

}

}

</**script**>

5.1.4添加商品到购物车

如果商品库存足够，用户就可以看见加入购物车的按钮，如果库存不足，就会看见库存不足。在商品加入购物车后，用户可以修改数量，另外，用户可以选择部分购物车内的商品提交到订单，而没被选中的商品会留在购物车内，最后也有删除所有商品的按钮，和删除单个商品的按钮。



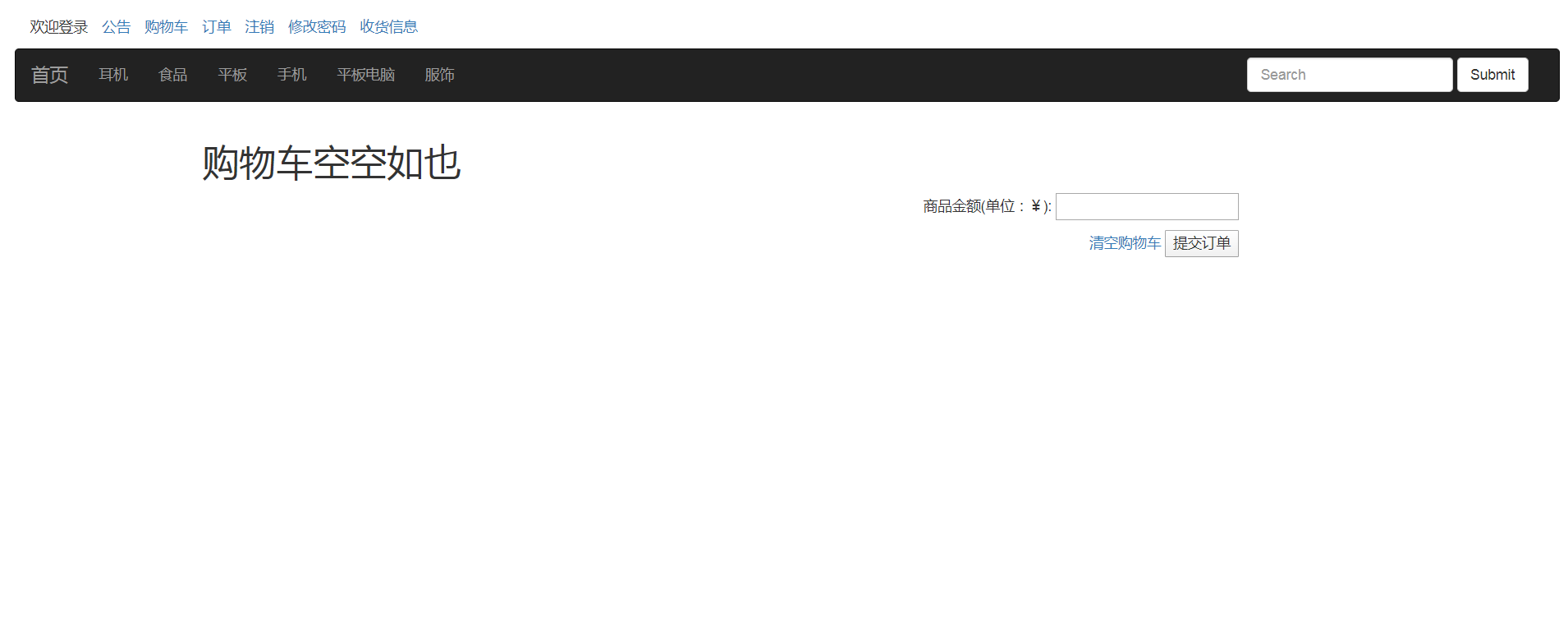


图5-6购物车

实现代码：

**public** ModelAndView addCar1(String pid, **int** count,HttpSession session) {

ModelAndView mav=**new** ModelAndView();

User user=(User)session.getAttribute(**"user"**);

*//通过pid找到商品内容*

Product p = **service**.findProductByPid(pid);

*//将count加入到购物车中*

Cart cart=**new** Cart();

cart.setCid(UUIDUtil.*getUUId*());

cart.setCount(count);

cart.setUid(user.getUsername());

cart.setPid(pid);

cart.setProduct(p);

**if**(**service1**.isIncart(cart)){

**service1**.updateCart(cart);

}**else** {**service1**.insetcart(cart);

}

List<Cart> carts=**service1**.findCart(user);

System.***out***.println(carts);

mav.addObject(**"carts"**,carts);

mav.setViewName(**"cart"**);

**return** mav;

}

**public** ModelAndView showCar(HttpSession session) {

User user=(User)session.getAttribute(**"user"**);

List<Cart> carts=**service1**.findCart(user);

ModelAndView mav=**new** ModelAndView();

mav.addObject(**"carts"**,carts);

System.***out***.println(carts);

mav.setViewName(**"cart"**);

**return** mav;

}

**public** ModelAndView delCar(String cid,HttpSession session) {

**service1**.deleteproduct(cid);

User user=(User)session.getAttribute(**"user"**);

List<Cart> carts=**service1**.findCart(user);

ModelAndView mav=**new** ModelAndView();

mav.addObject(**"carts"**,carts);

mav.setViewName(**"cart"**);

**return** mav;

}

**public** String clearCar(HttpSession session) {

User user=(User)session.getAttribute(**"user"**);

**service1**.deletecar(user);

**return "cart"**;

}

}

Js部分：

<**script**>

**function** *my*() {

***window***.**location**.**href**=**"${**pageContext.request.contextPath **}/order/addOrder.action?totalRow="**+totalRow;

}

**function** *up*(obj) {

**var** $td = **$**(obj).parents(**'tr'**).**children**(**'td'**);

**var** name= $td.eq(8).find(**"a"**).text();

**var** count= $td.eq(4).find(**"input"**).val();

***console***.log(name+11);

***console***.log(name)

**$**.**ajax**({

**type**: **"post"**,

**url**: **"${**pageContext.request.contextPath **}/car/updatecount.action"**,

**data**: {**"name"**:name,

**"count"**:count},

success: **function**(msg){

**if**(msg==**"yes"**){

***console***.log(1);

*// history.go(0);*

}**else** {

***console***.log(0)

}

}

});

}

***window***.**onload**=**function** (ev) {

*Xiaoji*();

**var** all=***document***.getElementById(**"all"**);

**var** inputs=***document***.getElementsByClassName(**"select"**);

all.addEventListener(**'change'**,**function** () {

**for**(i=0;i<inputs.**length**;i++){

**if**(all.**checked**){

inputs[i].**checked**=all.**checked**;

*getMoney*();

}**else** {

inputs[i].**checked**=**false**;

**$**(**"#in"**).val(0)

}

}

})

**for**(**var** i=0;i<inputs.**length**;i++){

inputs[i].addEventListener(**'change'**,**function** () {

*qx*();

});

}

**var** *qx*=**function** () {

**var** count=0;

**for**(**var** j=0;j<inputs.**length**;j++){

**if**(inputs[j].**checked**){

count++;

}**if**(count==inputs.**length**){

all.**checked**=**true**;

}**else** {

all.**checked**=**false**;

}

}

}

}

**function** *getMoney*() {

**var** totalRow=0;

**$**(**":checkbox:checked"**).parents(**"tr"**).find(**'td:eq(5)'**).each(**function**(){

totalRow += *parseFloat*(**$**(**this**).text());

***console***.log(totalRow);

});

**$**(**"#in"**).val(totalRow)

}

**function** *Xiaoji*() {

**var** tb=***document***.getElementById(**"my"**)

**var** rows=tb.**rows**;

**for**(**var** i=1;i<rows.**length**;i++){

rows[i].**cells**[5].**innerHTML**= *parseFloat*(rows[i].**cells**[3].**innerHTML**)\**parseFloat*(rows[i].**cells**[4].**getElementsByTagName**(**"INPUT"**)[0].**value**)

*getMoney*();

}

}

**function** *Getcid*() {

**var** arr=**new *Array***();

**$**(**":checkbox:checked"**).parents(**"tr"**).find(**'td:eq(7)'**).each(**function**(){

arr.push(**$**(**this**).find(**"span"**).text());

})

**var** t=**$**(**"#in"**).val();

***console***.log(arr);

**var** inputs=***document***.getElementsByClassName(**"select"**);

**var** count=0;

**for**(**var** j=0;j<inputs.**length**;j++){

**if**(inputs[j].**checked**){

count++;

}}

**if**(count==0){

**$**(**"#submi"**).attr(**"disabled"**,**false**);

*alert*(**"已选商品为空，请至少选择一件商品"**)

}**else** { ***window***.**location**.**href**=**"${**pageContext.request.contextPath **}/order/addOrder.action?"**+**"totalRow="**+t+**"&arr="**+arr;

}

}

</**script**>

5.1.5订单相关页面

当用户从购物车选中商品提交到订单时，订单开始生成，此时订单处于未付款状态，这时候用户可以退出，在订单页面可以看到订单状态，只有当用户确认订单之后，订单的状态才是已付款等待发货的状态





图5-7 订单

代码实现：

Js部分：

<**script**>

**function** *cl*(obj) {

**var** $th = **$**(obj).parents(**'tr'**).**children**(**'th'**);

**var** oid=$th.eq(0).text();

***console***.log(oid);

**$**.**ajax**({

**type**:**"post"**,

**url**:**"${**pageContext.request.contextPath **}/order/utilOrder1.action?"**,

**data**:{**"oid"**:oid},

success:**function** (msg) {

*alert*(**"操作成功"**)

***history***.go(0);

}

})

}

</**script**>

Java部分：

**public** ModelAndView addOrder(HttpSession session , **double** totalRow, @RequestParam(value=**"arr"**) String[] arr) {

ModelAndView mav = **new** ModelAndView();

System.***out***.println(1);

List<Cart> carts=**new** ArrayList<Cart>();

**for**(String a:arr){

**service**.findcid(a);

carts.add(**service**.findcid(a));

}

User loginUser = (User)session.getAttribute(**"user"**);

**if**(loginUser == **null**) {

mav.addObject(**"msg"**, **"请先登录，再继续购买！"**);

mav.setViewName(**"login"**);

**return** mav;

}

Order order = **new** Order();

order.setOid(UUIDUtil.*getUUId*());

order.setState(0);

order.setOrdertime(**new** Date(System.*currentTimeMillis*()));

order.setUser(loginUser);

**service**.addOrder(order);

**for**(Cart c : carts){

OrderItem orderItem = **new** OrderItem();

orderItem.setItemid(UUIDUtil.*getUUId*());

Product p=**service2**.findProductByPid(c.getPid());

c.setProduct(p);

orderItem.setCount(c.getCount());

System.***out***.println(p.getShop\_price());

System.***out***.println(c.getCount());

orderItem.setSubtotal(p.getShop\_price()\*c.getCount());

orderItem.setProduct(c.getProduct());

orderItem.setOrder(order);

order.getItems().add(orderItem);

**service**.addOrderItem(orderItem);

}

**for**(String a :arr){

**service1**.deletebycid(a);

}

List<Address> list=**new** ArrayList<>();

list=**service1**.getlist(loginUser);

mav.addObject(**"list"**,list);

mav.addObject(**"order"** , order);

mav.setViewName(**"order\_info"**);

**return** mav;

}

**5.2 后台实现部分**

5.2.1商品类型管理

管理员在登录管理系统之后，可以修改已经存在的商品分类，添加商品分类，删除已存在的商品分类。



图5-8编辑分类

代码实现：

Java部分：

**public class** AdminCategroyController {

@Autowired

CategoryService **service**;

@RequestMapping(**"/addCategory"**)

**public** String addCategory() {

**return "admin/category/add"**;

}

@RequestMapping(**"/editCategory"**)

**public** ModelAndView editCategory(String cid) {

ModelAndView mav = **new** ModelAndView();

Category c = **service**.queryCategoryByCid(cid);

System.***out***.println(c);

mav.setViewName(**"admin/category/edit"**);

mav.addObject(**"category"**,c);

**return** mav;

}

@RequestMapping(**"/listCategory"**)

**public** ModelAndView listCategory() {

ModelAndView mav = **new** ModelAndView();

List<Category> list = **service**.listCategory();

mav.setViewName(**"admin/category/list"**); *// 要显示的视图*

mav.addObject(**"list"**, list); *// 交给视图显示的数据*

**return** mav;

}

@RequestMapping(**"/saveCategory"**)

**public** ModelAndView saveCategroy(Category c) {

ModelAndView mav = **new** ModelAndView();

c.setCid(UUIDUtil.*getUUId*());

System.***out***.println(c);

**service**.addCategory(c);

mav.addObject(**"msg"**, **"添加成功"**);

mav.setViewName(**"admin/welcome"**);

**return** mav;

}

@RequestMapping(**"/updateCategory"**)

**public** ModelAndView updateCategory(Category c) {

ModelAndView mav = **new** ModelAndView();

**service**.updateCategory(c);

mav.addObject(**"msg"**, **"修改成功"**);

mav.setViewName(**"admin/welcome"**);

**return** mav;

}

@RequestMapping(**"/deleteCategory"**)

**public** ModelAndView deleteCategory(Category c) {

ModelAndView mav = **new** ModelAndView();

**service**.deleteCategory(c);

mav.addObject(**"msg"**, **"删除成功"**);

mav.setViewName(**"admin/welcome"**);

**return** mav;

}

}

5.2.2商品管理

管理员在登录系统后可以查看所有的商品，通过搜索栏搜索商品，然后对想要更改的商品进行信息的更改，也可以通过添加商品按钮来加入新的商品。如下图所示：





图5- 9添加商品

代码实现：

Java部分：

**public class** AdminProductController {

@Autowired

ProductService **service**;

@RequestMapping(**"/showAddProduct"**)

**public** String showAddProduct() {

**return "admin/product/add"**;

}

@RequestMapping(**"/listProduct"**)

**public** ModelAndView listProduct(**int** curPage) {

ModelAndView mav = **new** ModelAndView();

PageBean page = **service**.listProduct(curPage);

mav.setViewName(**"admin/product/list"**);

mav.addObject(**"page"**,page);

**return** mav;

}

@RequestMapping(**"/addProducts"**)

**public** ModelAndView addProduct(Product p,String cid,MultipartFile pictrueFile) **throws** IllegalStateException, IOException {

*// 设置图片名称，不能重复，使用 uuid*

String picName = UUIDUtil.*getUUId*();

*// 获取提交上来的文件名*

String oriName = pictrueFile.getOriginalFilename();

System.***out***.println(oriName);

System.***out***.println(23);

*// 通过文件名获取后缀名*

String extName = oriName.substring(oriName.lastIndexOf(**"."**));

*// 上传图片*

pictrueFile.transferTo(**new** File(**"C:\\Users\\Administrator\\Desktop\\C\\src\\main\\webapp\\pic\\"** + picName + extName));

System.***out***.print(picName);

p.setPid(UUIDUtil.*getUUId*());

p.setPimage(picName + extName);

Category c = **new** Category();

c.setCid(cid);

p.setCategory(c);

p.setPdate(**new** SimpleDateFormat(**"yyyy年MM月dd日 HH:mm:ss"**).format(**new** Date()));

ModelAndView mav = **new** ModelAndView();

**service**.addProduct(p);

mav.addObject(**"msg"**,**"添加商品成功"**);

mav.setViewName(**"admin/welcome"**);

**return** mav;

}

@RequestMapping(**"/updateProduct"**)

**public** ModelAndView updateProduct(MultipartFile pictrueFile,Product p)**throws** IllegalStateException ,IOException{

String picName = UUIDUtil.*getUUId*();

*// 获取提交上来的文件名*

String oriName = pictrueFile.getOriginalFilename();

*// 通过文件名获取后缀名*

String extName = oriName.substring(oriName.lastIndexOf(**"."**));

*// 上传图片*

pictrueFile.transferTo(**new** File(**"C:\\Users\\Administrator\\Desktop\\C\\src\\main\\webapp\\pic\\"** + picName + extName));

p.setPimage(picName + extName);

ModelAndView mav = **new** ModelAndView();

**service**.updateProduct(p);

mav.addObject(**"msg"**, **"修改成功"**);

mav.setViewName(**"admin/welcome"**);

**return** mav;

}

@RequestMapping(**"/editProduct"**)

**public** ModelAndView editProduct(String pid) {

ModelAndView mav = **new** ModelAndView();

Product p = **service**.queryProductByPid(pid);

mav.setViewName(**"admin/product/edit"**);

mav.addObject(**"product"**,p);

**return** mav;

}

@RequestMapping(**"/deleteProduct"**)

**public** ModelAndView deleteProduct(Product p) {

ModelAndView mav = **new** ModelAndView();

**service**.deleteProduct(p);

mav.addObject(**"msg"**, **"删除成功"**);

mav.setViewName(**"admin/welcome"**);

**return** mav;

}

}

5.2.3订单管理

管理员在登录系统后可以查看所有的订单，通过搜索栏搜索订单，然后对想要更改的订单进行订单状态的更改，对已经付款的订单进行发货处理，也可以将已经完成的订单进行删除，可以通过订单详情按钮来查看订单的具体商品内容。如下图所示：



图5-10订单管理

代码实现：

Java部分：

**public class** AdminOrderController {

@Autowired

OrderService **service**;

@Autowired

ProductService **service1**;

@RequestMapping(**"/listOrder"**)

**public** ModelAndView listOrder(**int** curPage) {

ModelAndView mav = **new** ModelAndView();

PageBean page = **service**.listOrder(curPage);

mav.setViewName(**"admin/order/list"**);

mav.addObject(**"page"**,page);

**return** mav;

}

@RequestMapping(**"/adminOrder"**)

@ResponseBody

**public** List<OrderItem> adminOrder(String oid){

**return service**.selectOrderItemByOid(oid);

}

@RequestMapping(**"/listOrderQita"**)

**public** ModelAndView listOrderyfk(**int** curPage,**int** state) {

ModelAndView mav = **new** ModelAndView();

PageBean page = **service**.listOrderQita(curPage , state);

System.***out***.println(page);

mav.setViewName(**"admin/order/list2"**);

mav.addObject(**"page"**,page);

**return** mav;

}

@RequestMapping(**"/utilOrder"**)

**public** ModelAndView utilOrder(String oid) {

ModelAndView mav = **new** ModelAndView();

Order order = **service**.selectOrderByOid(oid);

System.***out***.println(1);

List<OrderItem> orderItems=**service**.selectOrderItemByOid(oid);

System.***out***.println(orderItems);

**for**(OrderItem o:orderItems){

System.***out***.println(o.getCount());

**int** s=o.getProduct().getPcount()-o.getCount();

System.***out***.println(s);

o.getProduct().setPcount(s);

o.getProduct().setIs\_latest(o.getCount()+o.getProduct().getIs\_latest());

**service1**.updateProduct(o.getProduct());

}

**int** state = order.getState();

**if**(state != 3) {

state = state+1;

order.setState(state);

**service**.updateOrder(order);

}**else** {

order.setState(3);

**service**.updateOrder(order);

}

mav.addObject(**"msg"**,**"success"**);

mav.setViewName(**"admin/msg"**);

**return** mav;

}

@RequestMapping(**"/deleteOrder"**)

**public** ModelAndView deleteProduct(Order o) {

ModelAndView mav = **new** ModelAndView();

**service**.deleteOrder(o.getOid());

System.***out***.println(o);

mav.addObject(**"msg"**, **"删除成功"**);

mav.setViewName(**"admin/welcome"**);

**return** mav;

5.2.4公告管理

管理员在登录系统后可以查看所有的公告信息，可以添加新的公告，可以删除已存在的公告信息。



图5-11公告管理

代码实现：

Java部分：

**public** ModelAndView notice(){

List<notice> list= **service**.findnotice();

ModelAndView mav=**new** ModelAndView();

mav.setViewName(**"/admin/notice"**);

mav.addObject(**"list"** ,list);

**return** mav;

}

**public** ModelAndView deletenotice(String nid){

**service**.deletenotice(nid);

ModelAndView mav=**new** ModelAndView();

mav.setViewName(**"/admin/welcome"**);

mav.addObject(**"msg"** ,**"删除成功"**);

**return** mav;

}

5.2.5用户管理

管理员在登录系统后可以查看所有的用户信息，通过搜索栏搜索用户，然后对想要更改的用户进行用户信息的更改，也可以将用户进行删除。如下图所示：





图5-12用户管理

代码实现：

<**script**>

**$**(**"#password"**).blur(**function** () {

**var** reg=/^[0-9]{6}$/;

**if**(!reg.test(**$**(**"#password"**).val())){

**$**(**"#ptext"**).text(**"密码需要6个数字"**);

}**else** {

**$**(**"#ptext"**).text(**"密码可以使用"**);

}

})

**$**(**"#email"**).blur(**function** () {

**var** reg= /^([[A-Za-z0-9\_\-\.])+\@([A-Za-z0-9\_\-\.])+\.([A-Za-z]{2,4})$/;

**if**(!reg.test(**$**(**"#email"**).val())){

**$**(**"#pemail"**).text(**"邮箱格式不对"**);

}**else** {

**$**(**"#pemail"**).text(**"邮箱可以使用"**);

}

})

**$**(**"#phone"**).blur(**function** () {

**var** reg= /^1[3456789]\d{9}$/;

**if**(!reg.test(**$**(**"#phone"**).val())){

**$**(**"#pphone"**).text(**"电话格式不对"**);

}**else** {

**$**(**"#pphone"**).text(**"电话可以使用"**);

}

})

**$**(**"#name"**).blur(**function** () {

**if**(**$**(**"#name"**).val()!=**""**&&**$**(**"#name"**).val().trim()!=**""**){

**$**(**"#pname"**).text(**"姓名可以使用"**);

}**else** {

**$**(**"#pname"**).text(**"姓名不能为空"**);

}

})

**$**(**"#sex"**).blur(**function** () {

**if**(**$**(**"#sex"**).val()==**"男"**||**$**(**"#sex"**).val()==**"女"**){

**$**(**"#psex"**).text(**"性别可以使用"**);

}**else** {

**$**(**"#psex"**).text(**"性别只能为男或女"**);

}

})

**function** *m*() {

**if**(**$**(**"#ptext"**).text()==**"密码可以使用"**&&**$**(**"#pemail"**).text()==**"邮箱可以使用"**&&**$**(**"#pphone"**).text()==**"电话可以使用"**&&**$**(**"#pname"**).text()==**"姓名可以使用"**&&(**$**(**"#sex"**).val()==**"男"**||**$**(**"#sex"**).val()==**"女"**)){

**$**(**"#userAction\_save\_do"**).submit();

}**else** {

*alert*(**"请检查所有项符合要求且不为空！"**)

}

}

***window***.**onload**=**function** () {

**function** *x*() {

**var** reg=/^[0-9]{6}$/;

**if**(!reg.test(**$**(**"#password"**).val())){

**$**(**"#ptext"**).text(**"密码需要6个数字"**);

}**else** {

**$**(**"#ptext"**).text(**"密码可以使用"**);

}

}

**function** *x1*() {

**var** reg= /^([[A-Za-z0-9\_\-\.])+\@([A-Za-z0-9\_\-\.])+\.([A-Za-z]{2,4})$/;

**if**(!reg.test(**$**(**"#email"**).val())){

**$**(**"#pemail"**).text(**"邮箱格式不对"**);

}**else** {

**$**(**"#pemail"**).text(**"邮箱可以使用"**);

}

}

**function** *x2*() {

**var** reg= /^1[3456789]\d{9}$/;

**if**(!reg.test(**$**(**"#phone"**).val())){

**$**(**"#pphone"**).text(**"电话格式不对"**);

}**else** {

**$**(**"#pphone"**).text(**"电话可以使用"**);

}

}

**function** *x3*() {

**if**(**$**(**"#name"**).val()!=**""**&&**$**(**"#name"**).val().trim()!=**""**){

**$**(**"#pname"**).text(**"姓名可以使用"**);

}**else** {

**$**(**"#pname"**).text(**"姓名不能为空"**);

}

}

**function** *x4* () {

**if**(**$**(**"#sex"**).val()==**"男"**||**$**(**"#sex"**).val()==**"女"**){

**$**(**"#psex"**).text(**"性别可以使用"**);

}**else** {

**$**(**"#psex"**).text(**"性别只能为男或女"**);

}

}

*x*();

*x1*();

*x2*();

*x3*();

*x4*();

}

</**script**>

Java部分：

**public** ModelAndView getuser(**int** curPage){

ModelAndView mav=**new** ModelAndView();

PageBean page =**service**.listUser( curPage);

mav.addObject(**"page"**,page);

mav.setViewName(**"admin/usermanage/messageuser"**);

**return** mav;

}

**public** ModelAndView deleteuser(String uid){

ModelAndView mav=**new** ModelAndView();

**service**.deleteuser(uid);

mav.addObject(**"msg"**, **"删除成功"**);

mav.setViewName(**"admin/welcome"**);

**return** mav;

}

**public** ModelAndView updateuser(String uid){

ModelAndView mav=**new** ModelAndView();

User user =**service**.finduser(uid);

mav.addObject(**"user"**, user);

mav.setViewName(**"admin/usermanage/edit"**);

**return** mav;

**第6章 系统测试**

6.1软件测试

为了在正式上线之前，尽量发现并解决清除软件中不易见的错误或者bug，以此来提高软件的可用性，需要进行软件测试。此部分将对系统进行单元测试和综合测试，使用的测试方法是黑盒测试法。合适的选取测试用例力求最大程度覆盖此系统功能的方方面面，降低系统出现错误的可能性。

测试用例包括被测试的功能，因输入的数据和预期的输出结果，测试所用的数据选用少量，有效的测试数据进行测试，尽可能覆盖所有方面。基本目标是：设计一组数据，比较与预期结果是否相同，此测试用例应包含的方面：

数据的正确性测试：用户输入的数据是否能被系统正确处理。预期结果是否和实际结果一致

（1）数据的容错性测试：当用户输入非法数据（非法类型，不符合要求的数据）时，系统应能给出提示并进行相应的处理。当用户输入正确的数据，系统给出的结果应该与预期结果相同。

（2）完整性测试：对没有经过系统验证的人使用软件系统的企图，程序的数据处理能够保持数据库文件的完整性。

（3）接口与接口的测试，测试各个模块之间的数据传输，保证数据输入输出的一致性和正确性。

（4）压力测试：输入少量数据运行各个功能，输入大量数据运行测试。

（5）边界值分析法：确定边界情况（刚好等于，稍小于，大于），针对系统在测试过程中主要输入一些合法或者非法的数据，主要在边界值附件取

错误推测：根据测试的经验和感觉，参照以往的系统出现错误之处设计测试用例

效率：完成预定的功能，系统运行的时间

（6）回归测试，按照测试用例将所有的测试点测试完毕，并且解决发现的问题，再测试

6.1.1注册登录测试







图6-1注册测试

只要当所有项符合要求时，才会注册成功，当鼠标焦点失去的时候会自动判断输入是否符合。



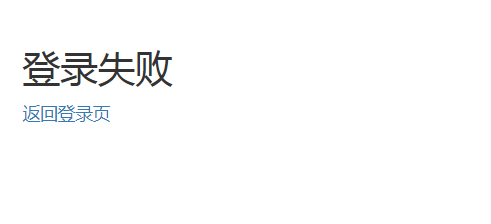


图6-2登录测试

如果账号密码匹配的话，就会登陆成功并进入首页。不匹配的话会显示登录失败。

6.1.2购物车功能测试



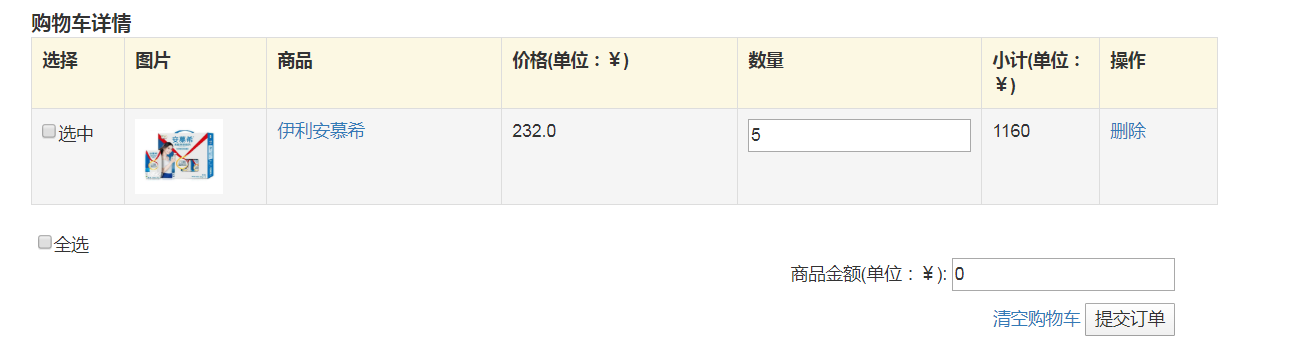


图6-3购物车测试

刚加入购物车的商品是没有被选中的，所以购物车的总金额为0，只会计算被选中商品的价格，可实现单选，全选，多选，删除商品等功能。

6.1.3订单功能测试

用户只有在登陆之后才能看见订单选项，点击可以看到相关信息





图6-4订单测试

在此页面可以确认收货，确认收货之后可以评价相关的产品。

6.1.4商品评价测试

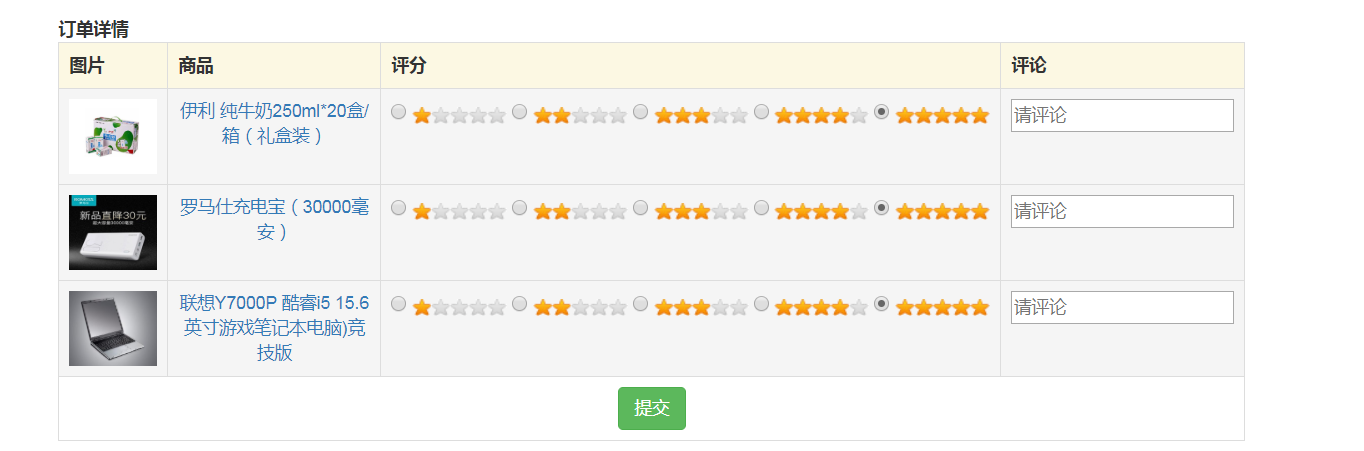


图6-5评论测试

用户在完成订单确定收货之后，可以在订单页面进入评价页面对购买的商品进行评价，打分等操作。



图6-6商品信息测试

在商品信息页面可以看到顾客的评价以及店家的回复等等。

6.1.5收货地址测试



图6-7收货信息测试一

用户在登陆之后可以点击收货信息进入此页面。完成对收货信息的添加



图6-8收货信息测试二

在确认订单的页面可以选择地址也可以新增收货信息。

6.1.6商品分类管理测试



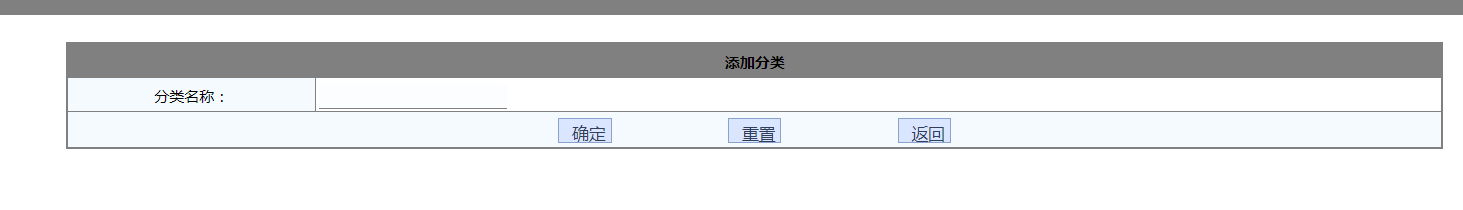


图6-9分类测试

管理员登录之后，可以添加分类，修改已经存在的分类。经测试，该功能无异常，可以使用

6.1.7商品的添加，修改测试

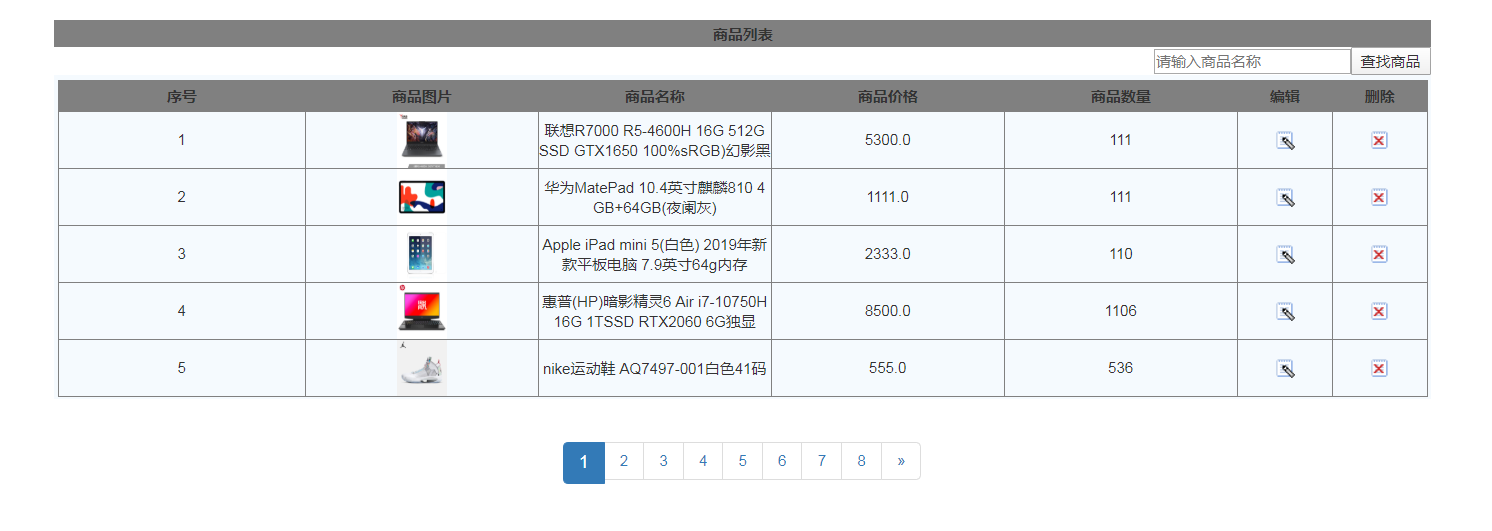
 

图6-10添加商品测试

管理登录后台后，可以添加商品，只有当所有信息都被填写之后，才能添加商品，也可以对已存在的商品信息进行修改，或者删除商品，以上功能都没有问题，可正常使用。

6.1.8订单管理测试



图6-11订单管理测试

订单有4种状态，管理员可根据不同的状态进行查询，并对相关订单的状态进行操作，也可以查看订单的内容，对于不需要的订单也可删除。经过测试，订单状态修改流程可以正常使用。

6.1.9用户管理测试

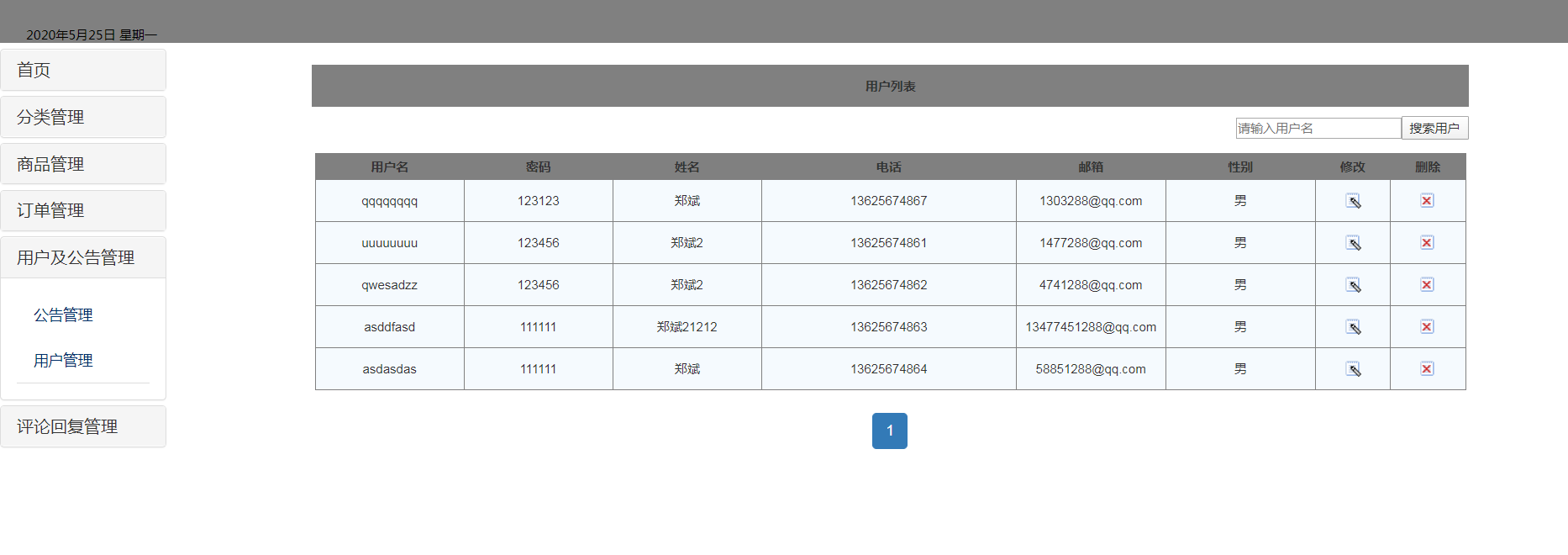


图6-12用户管理测试

管理员登录管理系统之后，可以搜索，查看用户的信息资料，并且可以修改除用户名之外的信息，也可以删除用户账号，经测试，此功能无问题。

6.1.10商品评论回复管理测试





图6-13评论回复测试

用户在确认收货之后，可以对商品进行评价，并且管理员可以对其进行回复操作，如上图。经测试，此功能可以正常运行。

6.2测试计划执行情况

表6-1测试计划表

|  |  |  |  |
| --- | --- | --- | --- |
| 测试类型 | 测试内容 | 测试目的 | 测试方法 |
| 功能测试 | 前台用户注册登录，后台管理员登录 | 确定所有相关功能正确与否 | 黑盒测试法 |
| 功能测试 | 商品的浏览，查询，加入购物车，以及购物车操作，订单功能 | 确定功能是否正常，数据是否正确 | 边界值法，等价类法 |
| 功能测试 | 后台管理 | 增删改查是否正确，是否能正确走预期的过程 | 手动测试 |
| 用户界面测试 | 导航，连接，颜色等 | 页面风格是否合理，一致 | 手工测试 |
| 安全测试 | 密码，超时 | 系统是否安全 | 黑盒法，手动测试 |

6.3测试效果

测试计划是对系统的功能进行的测试，测试范围包括了系统所有的功能，并且对相关功能的按钮做了多次测试，但是，测试计划不包含没有被发现的问题，所以当系统通过测试用例，并进行修改之后，再进行多次测试，系统无错误就认为通过测试。根据实际情况，此软件对顾客购物模块，订单模块，以及管理员管理模块都进行了安全性限制，需要验证才能进行操作，因此前后都有对应的验证功能。需要改进的方面支付功能不了解，后续应该可以加上支付宝，微信等支付功能模块。测试结果：基本完成网上购物平台系统的功能模块，基本达到了用户的需求。

表6-2测试结果表

|  |  |
| --- | --- |
| 测试内容 | 结论 |
| 界面 | 主界面比较友好，测试效果良好 |
| 后台 | 后台账号安全性有待提高，密码较为简单 |
| 注册，登录 | 功能基本通过测试，后续可以考虑短信验证 |
| 功能（订单，商品，购物车） | 基本完成需求，无大碍 |

6.4小结

这一章利用黑盒测试对系统进行了测试，严格挑选测试用例，尽最大的可能使测试用例覆盖范围，边界等方面覆盖系统所有功能。系统从可行性研究到编码，测试阶段都经过了严谨的开发过程，使得每一个测试点都顺利通过测试。对发现的问题通过总结，参考别人的成功经验得出一套方案。系统通过了单元测试和综合测试，在试运行中满足基本流程，可以正常使用。

**总结与展望**

将近一个学期的毕业设计，使我对Java开发有了更深的理解和体会，也让我对于Java开发的知识有了更系统、更全面的体会。想要开发一个Java Web系统，而且这个系统要完善且符合实际情况的话，则需要首先了解怎样才能开发一个完整的Java Web系统？开发一个完整且符合实际的Java系统，你该怎么做？它的捕步骤是什么呢？等等一系列的问题的答案，你才拥有了开发一个完整的Java Web系统的基础。当然，这都是在你编码能力合格的基础上的。当你这些问题都弄懂了，编码能力也合格，那你就可以开始设计你的系统了。也就是你可以开始需求分析，你要根据指导老师给的要求，去想要实现的功能有哪些？这些功能具体要求是什么？分析出来的结果是否符合要求符合实际？一个完整的业务流程是怎样的？业务流程是否符合实际情况？怎么实现它？你该用什么工具来完成呢？等等。这也只是开发一个完整的Java Web系统的第一步，后面还有数据库设计、系统实现与测试、系统调试和运行。其中系统实现和测试还可以细分为：数据库实现、数据访问层DAO的实现、前端页面的设计与实现、功能模块的实现。本文首先论述了此系统的开发背景、研究意义，随后在分析了此系统的必要性，得出此系统的设计应满足用户在使用时的日常需求，在整个系统运行过程中的流程尽量简短，人性化。

从顾客的角度看，此系统完成了用户账号注册登录功能，商品浏览功能，加入商品到购物车的功能，查看订单的功能，添加收货地址，评论商品的功能。以上功能能够满足用户网上购物的必要流程，保障系统正常运作。从管理员的角度看，此系统实现了添加商品，查看订单，处理订单状态，添加系统公告，管理用户账号，回复商品评价等功能，可以保障系统管理员对此系统的正常维护。

通过对本课题的研究，我对java web开发有了更深刻的认知，激发了我对编程的兴趣，编码能力也因此得到了很大的提高。在后台开发中，加深理解了SSM框架的原理，并且我还学习了MySQL数据库方面的相关知识，能实现对单表，多表进行增删改查。

虽然系统总体的功能都实现了，但是我还是感觉到了一些系统功能设计的不合理，但是由于自己个人技术水平的不够、经验的不足和时间的仓促，让我无法对系统进行大规模的修改和完善，因此我还是非常遗憾的，为此为列下来以下几点还可进行改进的地方：

（1）在订单模块模块中，没有实现具体的付款功能，应当调用支付宝或者微信接口来实现付款。

（2）注册登录时应当使用手机验证码来注册登录更加安全。

（3）可以根据用户习惯来推荐商品，每个用户推荐的商品应当不同。

（4）一些系统页面的美化不足，论文一些内容的表述存在不当之处。

在将来的学习工作当中，我会继续学习，逐步完善系统。在安徽工程大学短暂的四年求学生涯中，除了课堂上的知识，我们学会的还有为人处世的态度。有了这些，我们才能走得更远，这也是今后自己要努力学习、奋斗的方向。最后，我谨对那些在毕设各阶段中给予了我襄助、鼓励的所有人表达由衷的祝福和感谢。

# **致 谢**

白驹过隙，忽然而已。转眼间丰富多彩的校园生活已成过往。值此本科毕业论文完结之际，谨对多年来予以我襄助、激励的良师诤友致以最恳挚的谢忱。

首先我要向我的毕设指导老师邹珊老师表示由衷的谢意。感谢邹老师在系统设计开发以及论文编写的各个阶段予以的中肯见解，从最初的选题、文献搜寻到系统的需求分析、实现最后到论文编写、纠错等各阶段都给予我莫大的帮助。在老师的指点中看到了她一毫不苟、专注的学术作风。让我学会了对待任何真实的事物都应保持这种严谨的态度，这也是我们前进的目标更是成功的关键。其次，我还要小组同学在在系统开发、调试阶段对我的指点，帮助我解决诸多了bug才能顺利运行系统。

感谢所有传授我知识的老师，他们含辛茹苦的教导，让我拥有了足够的知识，让我对系统的实现和论文编写充满了动力和信心。同时也感谢和我同组的同学，因为大家的和睦相处、互相帮助，让我们结下了深深的友谊，也让我们彼此都拥有一个更好的开发环境和论文编写环境。在完成毕业设计的过程中，我深深的感觉到，作为一个合格的大学生，我们应该不仅仅只会程序编写技术，更重要的是对系统整体结构的布局和设计。在今后的工作和学习当中，我会不断完善自己，让自己变得更加强大，为我的母校争光，也为自己获得更美好的未来。

作者：

日期： 年 月 日

**参考文献**

[1]陈海汝,何青,潘轩平,张大宇,胡铁平.基于云服务器的SSM框架后台搭建与实现[J].信息系统工程,2019(11):114-115+117.

[2]蔡呈杰,王贵鑫.基于SSM框架的学子商城系统的设计与实现[J].科学技术创新,2019(32):69-71.

[3]侯金奎,鹿旭东,陈春雷,王磊.基于模型驱动的Web应用服务系统开发理论框架[J].小型微型计算机系统,2018,39(10):2345-2352.

[4]鞠宏军,佘春燕.基于SSM的高并发慕课网的设计与实现[J].电脑知识与技术,2020,16(02):54-56+90.

[5]戚琦,廖建新,王纯,武家春.基于敏捷方法的轻量级J2EE架构的应用[J].计算机系统应用,2007(02):53-56.

[6]王钱,王蓉,张利.基于iBatis的通用数据持久层的研究与设计[J].微计算机信息,2007(12):172-173+128.

[7]王艳清,陈红.基于SSM框架的智能web系统研发设计[J].计算机工程与设计,2012,33(12):4751-4757.

[8]徐刚,翟梦娇.基于SSM的美容资讯商务网站的设计与实现[J].商丘职业技术学院学报,2019,18(06):65-71.

[9]叶惠仙,沈文杰.基于Bootstrap+spring boot框架的在线考试系统开发[J].网络安全技术与应用,2019(12):54-57.

[10]张文龙,吴林辉,杨晨耀,蒋卫祥.基于SSM框架+vue的Web网盘系统的设计与实现[J].电脑知识与技术,2019,15(34):62-63+65.

[11]张俊萍,朱小冬,侯娜,张鲁,梁欣.基于SSM的软件体系结构开发过程研究[J].计算机测量与控制,2011,19(08):2029-2032.

[12]谌湘倩,狄文辉,孙冬.基于SSH框架与AJAX技术的JavaWeb应用开发[J].计算机工程与设计,2009,30(10):2590-2592+2596.

[13]朱运乔.基于SpringBoot+SSM框架的Web应用系统搭建与实现[J].电脑编程技巧与维护,2019(10):23-25.

[14]陈峰. 基于SSM框架的B2C网上商城系统的设计与实现[D].湖南大学,2018.

[15]唐文静,张志勤,王庆军.基于SSH2的辅助选择的网上商城的设计与实现[J].计算机与数字工程,2019,47(06):1550-1554.

[16]Abdellatif, T. and F. Boyer. A node allocation system for deploying JavaEE systems on Grids. 2009. Hammemet, Tunisia.

[17]Li, Z. and Z. Weixi. Design of tourism e-business system based on  JavaEE multi-pattern. 2012. Sanya, China.

[18]Andrea Capiluppi,Nemitari Ajienka,Steve Counsell. The Effect of Multiple Developers on Structural Attributes: A Study Based on Java Software[J]. Elsevier Inc.,2020.

**附录**

附录A 外文文献及其译文

Core Java™ Volume II–Advanced Features

When Java technology first appeared on the scene, the excitement was not about a well-crafted programming language but about the possibility of safely executing applets that are delivered over the Internet (see Volume I, Chapter 10 for more information about applets). Obviously, delivering executable applets is practical only when the recipients are sure that the code can't wreak havoc on their machines. For this reason, security was and is a major concern of both the designers and the users of Java technology. This means that unlike other languages and systems, where security was implemented as an afterthought or a reaction to break-ins, security mechanisms are an integral part of Java technology.

Three mechanisms help ensure safety:

• Language design features (bounds checking on arrays, no unchecked type conversions, no pointer arithmetic, and so on).

• An access control mechanism that controls what the code can do (such as file access, network access, and so on).

• Code signing, whereby code authors can use standard cryptographic algorithms to authenticate Java code. Then, the users of the code can determine exactly who created the code and whether the code has been altered after it was signed.

Below, you'll see the cryptographic algorithms supplied in the java.security package, which allow for code signing and user authentication.

As we said earlier, applets were what started the craze over the Java platform. In practice, people discovered that although they could write animated applets like the famous "nervous text" applet, applets could not do a whole lot of useful stuff in the JDK 1.0 security model. For example, because applets under JDK 1.0 were so closely supervised, they couldn't do much good on a corporate intranet, even though relatively little risk attaches to executing an applet from your company's secure intranet. It quickly became clear to Sun that for applets to become truly useful, it was important for users to be able to assign different levels of security, depending on where the applet originated. If an applet comes from a trusted supplier and it has not been tampered with, the user of that applet can then decide whether to give the applet more privileges.

To give more trust to an applet, we need to know two things:

• Where did the applet come from?

• Was the code corrupted in transit?

In the past 50 years, mathematicians and computer scientists have developed sophisticated algorithms for ensuring the integrity of data and for electronic signatures. The java.security package contains implementations of many of these algorithms. Fortunately, you don't need to understand the underlying mathematics to use the algorithms in the java.security package. In the next sections, we show you how message digests can detect changes in data files and how digital signatures can prove the identity of the signer.

A message digest is a digital fingerprint of a block of data. For example, the so-called SHA1 (secure hash algorithm #1) condenses any data block, no matter how long, into a sequence of 160 bits (20 bytes). As with real fingerprints, one hopes that no two messages have the same SHA1 fingerprint. Of course, that cannot be true—there are only 2160 SHA1 fingerprints, so there must be some messages with the same fingerprint. But 2160 is so large that the probability of duplication occurring is negligible. How negligible? According to James Walsh in True Odds: How Risks Affect Your Everyday Life (Merritt Publishing 1996), the chance that you will die from being struck by lightning is about one in 30,000. Now, think of nine other people, for example, your nine least favorite managers or professors. The chance that you and all of them will die from lightning strikes is higher than that of a forged message having the same SHA1 fingerprint as the original. (Of course, more than ten people, none of whom you are likely to know, will die from lightning strikes. However, we are talking about the far slimmer chance that your particular choice of people will be wiped out.)

A message digest has two essential properties:

• If one bit or several bits of the data are changed, then the message digest also changes.

• A forger who is in possession of a given message cannot construct a fake message that has the same message digest as the original.

The second property is again a matter of probabilities, of course. Consider the following message by the billionaire father:"Upon my death, my property shall be divided equally among my children; however, my son George shall receive nothing."

That message has an SHA1 fingerprint of

2D 8B 35 F3 BF 49 CD B1 94 04 E0 66 21 2B 5E 57 70 49 E1 7E

The distrustful father has deposited the message with one attorney and the fingerprint with another. Now, suppose George can bribe the lawyer holding the message. He wants to change the message so that Bill gets nothing. Of course, that changes the fingerprint to a completely different bit pattern:

2A 33 0B 4B B3 FE CC 1C 9D 5C 01 A7 09 51 0B 49 AC 8F 98 92

Can George find some other wording that matches the fingerprint? If he had been the proud owner of a billion computers from the time the Earth was formed, each computing a million messages a second, he would not yet have found a message he could substitute.

A number of algorithms have been designed to compute these message digests. The two best-known are SHA1, the secure hash algorithm developed by the National Institute of Standards and Technology, and MD5, an algorithm invented by Ronald Rivest of MIT. Both algorithms scramble the bits of a message in ingenious ways. For details about these algorithms, see, for example, Cryptography and Network Security, 4th ed., by William Stallings (Prentice Hall 2005). Note that recently, subtle regularities have been discovered in both algorithms. At this point, most cryptographers recommend avoiding MD5 and using SHA1 until a stronger alternative becomes available. (See http://www.rsa.com/rsalabs/node.asp?id=2834 for more information.)

The Java programming language implements both SHA1 and MD5. The MessageDigest class is a factory for creating objects that encapsulate the fingerprinting algorithms. It has a static method, called getInstance, that returns an object of a class that extends the MessageDigest class. This means the MessageDigest class serves double duty:

• As a factory class

• As the superclass for all message digest algorithms

For example, here is how you obtain an object that can compute SHA fingerprints:

MessageDigest alg = MessageDigest.getInstance("SHA-1");

(To get an object that can compute MD5, use the string "MD5" as the argument to getInstance.)

After you have obtained a MessageDigest object, you feed it all the bytes in the message by repeatedly calling the update method. For example, the following code passes all bytes in a file to the alg object just created to do the fingerprinting：

InputStream in = . . .

int ch;

while ((ch = in.read()) != -1)

alg.update((byte) ch);

Alternatively, if you have the bytes in an array, you can update the entire array at once:

byte[] bytes = . . .;

alg.update(bytes);

When you are done, call the digest method. This method pads the input—as required by the fingerprinting algorithm—does the computation, and returns the digest as an array of bytes.

byte[] hash = alg.digest();

The program in Listing 9-15 computes a message digest, using either SHA or MD5. You can load the data to be digested from a file, or you can type a message in the text area.

Message Signing

In the last section, you saw how to compute a message digest, a fingerprint for the original message. If the message is altered, then the fingerprint of the altered message will not match the fingerprint of the original. If the message and its fingerprint are delivered separately, then the recipient can check whether the message has been tampered with. However, if both the message and the fingerprint were intercepted, it is an easy matter to modify the message and then recompute the fingerprint. After all, the message digest algorithms are publicly known, and they don't require secret keys. In that case, the recipient of the forged message and the recomputed fingerprint would never know that the message has been altered. Digital signatures solve this problem.

To help you understand how digital signatures work, we explain a few concepts from the field called public key cryptography. Public key cryptography is based on the notion of a public key and private key. The idea is that you tell everyone in the world your public key. However, only you hold the private key, and it is important that you safeguard it and don't release it to anyone else. The keys are matched by mathematical relationships, but the exact nature of these relationships is not important for us. (If you are interested, you can look it up in The Handbook of Applied Cryptography at http://www.cacr.math.uwaterloo.ca/hac/.)

The keys are quite long and complex. For example, here is a matching pair of public and private Digital Signature Algorithm (DSA) keys.

Public key:

Code View:

p:

fca682ce8e12caba26efccf7110e526db078b05edecbcd1eb4a208f3ae1617ae01f35b91a47e6df63413c5e12ed0899bcd132acd50d99151bdc43ee737592e17

q: 962eddcc369cba8ebb260ee6b6a126d9346e38c5

g:678471b27a9cf44ee91a49c5147db1a9aaf244f05a434d6486931d2d14271b9e35030b71fd73da179069b32e2935630e1c2062354d0da20a6c416e50be794ca4

y:

c0b6e67b4ac098eb1a32c5f8c4c1f0e7e6fb9d832532e27d0bdab9ca2d2a8123ce5a8018b8161a760480fadd040b927281ddb22cb9bc4df596d7de4d1b977d50

Private key:

Code View:

p:

fca682ce8e12caba26efccf7110e526db078b05edecbcd1eb4a208f3ae1617ae01f35b91a47e6df63413c5e12ed0899bcd132acd50d99151bdc43ee737592e17

q: 962eddcc369cba8ebb260ee6b6a126d9346e38c5

g:

678471b27a9cf44ee91a49c5147db1a9aaf244f05a434d6486931d2d14271b9e35030b71fd73da179069b32e2935630e1c2062354d0da20a6c416e50be794ca4

x: 146c09f881656cc6c51f27ea6c3a91b85ed1d70a

It is believed to be practically impossible to compute one key from the other. That is, even though everyone knows your public key, they can't compute your private key in your lifetime, no matter how many computing resources they have available.

It might seem difficult to believe that nobody can compute the private key from the public keys, but nobody has ever found an algorithm to do this for the encryption algorithms that are in common use today. If the keys are sufficiently long, brute force—simply trying all possible keys—would require more computers than can be built from all the atoms in the solar system, crunching away for thousands of years. Of course, it is possible that someone could come up with algorithms for computing keys that are much more clever than brute force. For example, the RSA algorithm (the encryption algorithm invented by Rivest, Shamir, and Adleman) depends on the difficulty of factoring large numbers. For the last 20 years, many of the best mathematicians have tried to come up with good factoring algorithms, but so far with no success. For that reason, most cryptographers believe that keys with a "modulus" of 2,000 bits or more are currently completely safe from any attack. DSA is believed to be similarly secure.

Figure 9-12 illustrates how the process works in practice.

Suppose Alice wants to send Bob a message, and Bob wants to know this message came from Alice and not an impostor. Alice writes the message and then signs the message digest with her private key. Bob gets a copy of her public key. Bob then applies the public key to verify the signature. If the verification passes, then Bob can be assured of two facts:

• The original message has not been altered.

• The message was signed by Alice, the holder of the private key that matches the public key that Bob used for verification.

You can see why security for private keys is all-important. If someone steals Alice's private key or if a government can require her to turn it over, then she is in trouble. The thief or a government agent can impersonate her by sending messages, money transfer instructions, and so on, that others will believe came from Alice.

The X.509 Certificate Format

To take advantage of public key cryptography, the public keys must be distributed. One of the most common distribution formats is called X.509. Certificates in the X.509 format are widely used by VeriSign, Microsoft, Netscape, and many other companies, for signing e-mail messages, authenticating program code, and certifying many other kinds of data. The X.509 standard is part of the X.500 series of recommendations for a directory service by the international telephone standards body, the CCITT.

The precise structure of X.509 certificates is described in a formal notation, called "abstract syntax notation #1" or ASN.1. Figure 9-13 shows the ASN.1 definition of version 3 of the X.509 format. The exact syntax is not important for us, but, as you can see, ASN.1 gives a precise definition of the structure of a certificate file. The basic encoding rules, or BER, and a variation, called distinguished encoding rules (DER) describe precisely how to save this structure in a binary file. That is, BER and DER describe how to encode integers, character strings, bit strings, and constructs such as SEQUENCE, CHOICE, and OPTIONAL.

中文译文：

Java核心技术 卷Ⅱ高级特性

当Java技术刚刚问世时，令人激动的并不是因为它是一个设计完美的编程语言，而是因为它能够安全地运行通过因特网传播的各种applet。很显然，只有当用户确信applet的代码不会破坏他的计算机时，用户才会接受在网上传播的可执行的applet。正因为如此，无论过去还是现在，安全都是设计人员和Java技术使用者所关心的一个重大问题。这就意味着，Java技术与其他的语言和系统有所不同，在那些语言和系统中安全是事后才想到要去实现的，或者仅仅是对破坏的一种应对措施，而对Java技术来说，安全机制是一个不可分割的组成部分。

Java技术提供了以下三种确保安全的机制：

(1)语言设计特性（对数组的边界进行检查，无不检查类型的转换，无指针算法等）。

(2)访问控制机制，用于控制代码能够执行的功能（比如文件访问，网络访问等）。

(3) 代码签名，利用该特性，代码的作者就能够用标准的加密算法来表明Java代码的身份。这样，该代码的使用者就能够准确地知道谁创建了该代码，以及代码被标识后是否被修改过。

下面，我们要介绍java.security包提供的加密算法，用来进行代码的标识和用户身份认证。

正如我们前面所说，applet 是在Java平台上开始流行起来的。实际上，人们发现尽管他们可以编写像著名的“nervous text”那样栩栩如生的applet，但是在JDK1.0安全模式下无法发挥其一整套非常有用的作用。例如，由于JDK1.0下的applet要受到严密的监督，因此，即使applet在公司安全内部网上运行时的风险相对较小，applet也无法在企业内部网上发挥很大的作用。Sun公司很快就认识到，要使applet真正变得非常有用，用户必须可以根据applet的来源为其分配不同的安全级别。如果applet来自值得信赖的提供商，并且没有被篡改过，那么applet的用户就可以决定是否给applet授予更多的运行特权。

如果要给予applet更多的信赖，你必须知道下面两件事：

(1)applet来自哪里？

(2)在传输过程中代码是否被破坏？

在过去的50年里，数学家和技术机科学家已经开发出各种各样成熟的算法，用于确保数据和电子签名的完整性，在java.security包中包含了许多这些算法的实现。在下面几节，我们将要介绍消息摘要是如何检测数据文件中的变化的，以及数字签名是如何证明签名者的身份的。

消息摘要是数据块的数字指纹。例如，所谓的SHA1（安全散列算法#1）可将任何数据块，无论其数据有多长，都压缩为160位（20字节）的序列。与真实的指纹一样，人们希望任何两条消息都不会有相同的SHA1指纹。当然这是不可能的—因为只存在2160 个SHA1指纹，所有肯定会有某些消息具有相同的指纹。因为2160 是一个很大的数字，所以存在重复指纹的可能性微乎其微，那么这种重复的可能性到底小到什么程度呢？根据James Walsh在他的《True Odds:How Risks Affect Your Everyday Life》,Merritt Publishing出版社1996年出版，一书中所阐述的，你和他们所有的人都死于雷击的概率，比伪造的消息与原来消息具有相同的SHA1指纹的概率还要高。（当然，可能有你不认识的其他10个以上的人会死于雷击，但这里我们讨论的是你选择的特定的人的死亡概率）。

消息摘要具有两个基本属性：

(1)如果数据的1位或者几位改变了，那么消息摘要也将改变。

(2）拥有给定消息的伪造者不能创建与原消息具有相同摘要的假消息。

当然，第二个属性又是一个概率问题。让我们来看看下面这位亿万富翁下的遗嘱：“我死了之后，我的财产将由我的孩子平分，但是，我的儿子George应该拿不到一个子。”

这份遗嘱的SHA1指纹为：

2D 8B 35 F3 BF 49 CD B1 94 04 E0 66 21 2B 5E 57 70 49 E1 7E

这位有疑心病的父亲将这份遗嘱交给一位律师保存，而将指纹交给另一位律师保存。现在，假设George能够贿赂那位保存遗嘱的律师，他想修改这份遗嘱，使得Bill一无所得。当然，这需要将原指纹改为下面这样完全不同的位模式：

2A 33 0B 4B B3 FE CC 1C 9D 5C 01 A7 09 51 0B 49 AC 8F 98 92

那么George能够找到与该指纹相匹配的其他文字吗？如果从地球形成之时，他就很自豪地拥有10亿台计算机，每台计算机每秒钟处理一百万条信息，他依然无法找到一个能够替换的遗嘱。

人们已经设计出大量的算法，用于计算这些消息摘要，其中最著名的两种算法是SHAI和MD5。SHAI是由美国国家标准和技术学会开发的加密散列算法，MD5是由麻省理工学院的Ronald Rivest发明的算法。这两种算法都使用了独特巧妙的方法对消息中的各个位进行扰乱。如果要了解这些方法的详细信息，请参阅William Stallings撰写的《Cryptography and Network Security》一书，该书由Prentice Hall出版社于2005年出版口值得注意的是，最近人们在这两种算法中发现了某些微妙的规律性，因此许多密码人员建议最好避免使用MD5，而应该使用SHA1算法，直到有更强的加密算法出现。（查看http://www.rsa.com/rsalabs/node.asp?id=2834以了解更多的信息）。

Java编程语言已经实现了SHA1和MD5。MessageDigest类是用于创建封装了指纹算法的对象的“工厂”，它的静态方法getInstance返回继承了MessageDigest类的某个类的对象。这意味着MessageDigest类能够承担下面的双重职责：

（1）作为一个工厂类。

（2）作为所有消息摘要算法的超类。

例如，下面是如何获取一个能够计算SHA指纹的对象的方法：

MessageDigest alg = MessageDigest.getInstance(“SHA-1”)；

（如果要获取计算MD5的对象，请使用字符串“MD5”作为getInstance的参数。）

当你已经获取MessageDigest对象之后，通过反复调用update方法，将信息中的所有字节提供给该对象。例如，下面的代码将文件中的所有字节传给上面建立的alg对象，以执行指纹算法：

InputStream in=….

int ch；

while((ch=in.read())!=-1)

alg.updat((byte) ch)；

另外，如果这些字节存放在一个数组中，那就可以一次完成整个数组的更新：

byte[] bytes =．．．；

alg.update(bytes)；

当完成上述操作后，调用digest方法。该方法填充输入信息—指纹算法需要的—并且进行相应的计算，然后以字节数组的形式返回消息摘要。

byte[] hash=alg.digest()；

程序清单9-15中的程序计算了一个消息摘要，既可以用SHA，也可以使用MD5来计算。可以从文件加载需要计算摘要的数据，也可以直接将信息输入文本区域。图9-11显示了该应用程序的画面。

消息签名

在上一节中，我们介绍了如何计算原始消息的消息摘要和指纹的方法。如果消息改变了，那么改变后的消息的指纹与原消息的指纹将不匹配。如果消息和它的指纹是分开传送的，那么接收者就可以检查消息是否被篡改过。但是，如果消息和指纹同时被截获了，对消息进行修改，再重新计算指纹，这是一件很容易的事情。毕竟，消息摘要算法是公开的，不需要使用任何密钥。在这种情况下，假消息和新指纹的接收者永远不会知道消息已经被篡改。数字签名解决了这个问题。

为了了解数字签名的工作原理，我们需要解释关于公共密钥加密技术领域中的几个概念。公共密钥加密技术是基于公共密钥和私有密钥这个两个基本概念的。它的设计思想是你可以将公共密钥告诉世界上的任何人，但是，只有自己才拥有私有密钥，重要的是你要保护你的私有密钥，不将它泄漏给其他任何入。这些密钥之间存在一定的数学关系，但是这种关系的具体性质对于实际的编程来说并不重要(如果你有兴趣，可以参阅http://www.cacr.math.uwaterloo.ca/hac/站点上的《The Handbook of Applied Cryptography》 一书)。

密钥非常长，而且很复杂。例如，下面是一对匹配的数字签名算法(DSA)公共密钥和私有密钥。

公共密钥：

p: fca682ce8e12caba26efccf7ll0e526db078b05e6ecbcdleb4a208f3ae1617ae0lf35b9la47e6df63413c5e12ed0899bcd132acd50d9915lbdc43ee737592el7

q: 962eddcc369cba8ebb260ee6b6a126d9346e38c5

g:67847lb27a9cf44ee9la49c5147dbla9aaf244f05a434d648693ld2d1427lb9e35030b7lfd73da179069b32e2935630elc2062354d0da20a6c416e50be794ca4

y：

c0b6e67b4ac098ebla32c5f8c4clfee7e6fb9d832532e27d0bdab9ca2d2a8123ce5a8018b816la6048efadd040b927281ddb22cb9bc4df596d7de4dlb977dS0

私有密钥：

p：

fca682ce8e12caba26efccf7ll0e526db078b05edecbcdleb4a208f3ae1617ae0lf35b9la47e6df63413c5e12ed0899bcd132acd50d9915lbdc43ee737592e17

q: 962eddcc369cba8ebb260ee6b6a126d9346e38c5

g：

67847lb27a9cf44ee9la49c5147dbla9aaf244f05a434d648693ld2d1427lb9e35030b7lfd73da179069b32e2935630elc2062354d0da20a6c416e50be794ca4

x: 146c09f881656cc6c5lf27ea6c3a9lb85edld70a

在现实中，几乎不可能用一个密钥去推算出另一个密钥。也就是说，即使每个人都知道你的公共密钥，不管他们拥有多少计算资源，他们一辈子也无法计算出你的私有密钥。

任何人都无法根据公共密钥来推算私有密钥，这似乎让人难以置信。但是时至今日，还没有人能够找到一种算法，来为现在常用的加密算法进行这种推算。如果密钥足够长，那么要是使用穷举法—也就是直按试验所有可能的密钥—所需要的计算机将比用太阳系中的所有原子来制造的计算机还要多，而且还得花费数千年的时间。当然，可能会有人提出比穷举更灵活的计算密钥的算法。例如，RSA算法（该加密算法由Rivest, Shamir和Adleman发明）就利用了对数值巨大的数字进行因子分解的困难性。在最近20年里，许多优秀的数学家都在尝试提出好的因子分解算法，但是迄今为止都没有成功。据此，大多数密码学者认为，拥有2000位或者更多位“模数”的密钥目前是完全安全的，可以抵御任何攻击。DSA被认为具有类似的安全性。

图9-12展示了这项工作的处理过程。

假设Alice想要给Bob发送一个消息，Bob想知道该消息是否来自Alice，而不是冒名顶替者。Alice写好了消息，并且用她的私有密钥对该消息摘要签名。 Bob得到了她的公共密钥的拷贝，然后Bob用公共密钥对该签名进行校验。如果通过了校验，则Bob可以确认以下两个事实：

（1)原始消息没有被篡改过。

（2)该消息是由Alice签名的，她是私有密钥的持有者，该私有密钥就是Bob

与她用于校验的公共密钥相匹配的密钥。

你可以看到私有密钥的安全性为什么是最重要的。如果某个人偷了Alice的私有密钥，或者政府要求她交出私有密钥，那么她就麻烦了。小偷或者政府代表就可以假扮她的身份来发送消息和资金转账指令等等，而其他人则会相信这些消息确实来自于Alice。

X.509证书格式

为了利用公共密钥这种密码系统，必须将公共密钥分发出去。最通用的一种签名证书格式称为X.509格式。X.509格式的证书被VeriSign、微软、网景和其他许多公司广泛应用于对电子邮件消息进行签名，对程序代码进行认证，以及对许多其他类型的数据进行认证等等。 X.509标准是由国际电话标准机构，即国际电报电话咨询委员会(CCITT)提出的用于目录服务的X.500系列建议的组成部分。

X.509证书的具体结构是用一种形式化表示来描述的，称为“抽象语法表示法#1”(abstract syntax notation)即ASN.1。图9-13显示了第3版X.509格式的ASN.1定义。虽然具体的语法对我们并不重要，但是你可以看到，ASN.1为证书文件的结构给出了精确的定义。“基本编码规则”(basic encoding rules)，即BER，精确地描述了如何将该结构保存为二迸制文件。也就是说，BER描述了如何对整数、字符串、位串以及诸如SEQUENCE、CHOICE和OPTIONAL的结构进行编码的方法。

附录B 主要参考文献的题录及摘要或参考文献原文

【1】【篇名】基于云服务器的SSM框架后台搭建与实现

【作者】陈海汝,何青,潘轩平,张大宇,胡铁平.

【刊名】信息系统工程

【中文关键词】云服务器; SSM框架; 微信小程序; MSQL数据库;

【中文摘要】摘要：云服务器已经成为了中小型企业和个人最适用的服务器平台,便利和维护费用低等优点使其成为了小型项目首选服务器选择。SSM框架是商业中最常用的后台框架,框架具有维护便利、拓展性高和移植性强等优点,将SSM后台框架部署在云服务器可以在很短的时间搭建后台并上线项目。

【2】【篇名】基于SSM框架的学子商城系统的设计与实现

【作者】蔡呈杰,王贵鑫

【刊名】科学技术创新

【中文关键词】学子商城; 系统; 软件;

【中文摘要】摘要：从20世纪90年代开始,随着互联网,大数据的来临,人类进入快节奏的信息时代,即互联网时代,其中网上浏览商品,购买商品已经是人们都会做的事情,已经成为人们生活不可或缺的一部份,因此开发出一款用户体验好,用户数据的安全性高,可维护性强的商城平台是多么重要。虽然目前已经存在很多商城平台,但其中还是有很多不足和需要改进的地方,这次设计是基于SSM框架开发的一款学子商城项目,该项目继承了已经存在项目的一部分优点,并在此基础上有了创新和改进,比如使用分层架构的思想,用户登录需要输入验证码等,不但提高了安全性,而且后端代码易于管理,方便后期功能的添加与修改,满足更多用户的需求。

【3】【篇名】基于模型驱动的Web应用服务系统开发理论框架

【作者】侯金奎,鹿旭东,陈春雷,王磊.

【刊名】小型微型计算机系统

【中文关键词】Web应用服务系统;模型驱动开发;模型转换;模型精化;

【中文摘要】摘要：从软件设计实践出发,针对基于Web应用服务和设计模式的软件设计方法,提出了一种新的模型驱动开发理论框架.该框架以类型范畴理论为基础,以服务构件标识为范畴对象,以服务构件之间的交互和依赖关系为范畴态射,从而用范畴图表给出了软件体系结构模型的框架描述.设计模式看成是构件规范的模板,模板的实例化和规范的集成过程就是软件设计和实现的过程,从而建立起了模型转换和精化中特性描述和一致性研究的统一框架.应用研究表明,该框架符合模型驱动软件设计的本质要求,为模型转换和模型驱动的Web应用服务开发提供了新的认知、分析和研究的指导架构.

【4】【篇名】基于SSM的高并发慕课网的设计与实现

【作者】鞠宏军,佘春燕.

【刊名】电脑知识与技术

【中文关键词】负载均衡;慕课;SSM;Nginx;Redis;

【中文摘要】摘要：为了帮助IT学子能够更加便利的获取学习资源,该系统采用B/S架构的方式,将大量学习资源通过互联网整合在一起,提供给用户学习使用。系统主要实现视频学习、发布讨论、撰写博客、站内搜索、程序在线测评等功能,用户也可以通过评论的方式实现问题解答、交流互动。

【5】【篇名】基于敏捷方法的轻量级J2EE架构的应用

【作者】戚琦,廖建新,王纯,武家春

【刊名】计算机系统应用

【中文关键词】敏捷建模;J2EE;Struts;Spring;Hibernate;

【中文摘要】摘要：本文提出Sfruls+Spring+Hibernate三个开源框架的松耦合层次结构,加之Tomcat开源服务器组成一个轻量级的J2EE架构。文章结合敏捷软件开发方法说明了轻量级J2EE架构设计实现的过程,并针对目标系统彩信短信管理平台设计了Struts+Spring+Hibernate的系统概念模型。本文希望通过目标系统的开发探索敏捷开发的思想在轻量级J2EE系统架构中的应用。

【6】【篇名】基于iBatis的通用数据持久层的研究与设计

【作者】王钱,王蓉,张利

【刊名】微计算机信息

【中文关键词】持久层对象; iBatis; SQL;

【中文摘要】摘要：目前持久层开发仍然是持久层对象和数据库表格一一对应,严重影响了开发和维护效率,针对目前传统的持久层开发思想,本文提出了一种基于iBatis框架的通用数据持久层设计方案,避免了重复劳动。

【7】【篇名】基于SSM框架的智能web系统研发设计

【作者】王艳清,陈红

【刊名】计算机工程与设计

【中文关键词】智能; web系统; 布局; 效率; SSM框架;

【中文摘要】摘要：为了解决传统CMS系统不能对web布局进行修改的局限性,降低维护web系统成本及风险,提高管理效率,提出了集建站和维护系统于一体智能web系统设计。设计中使用了SSM框架,很大程度上提高了开发效率及系统的可靠性和易用性;Ajax、jQuery、jqGrid等技术的采用实现了拖动页面布局修改web布局的效果。此外还对传统的菜单管理和内容管理等功能进行了改进,从而提高了web系统维护效率,降低维护风险,更方便管理人员。

【8】【篇名】基于SSM的美容资讯商务网站的设计与实现

【作者】徐刚,翟梦娇

【刊名】商丘职业技术学院学报

【中文关键词】SSM; 购物; 美妆信息;

【中文摘要】摘要：随着人们对于美好生活的向往,人们越来越注重个人外在美.虽然网上购物的网站较多,但集购物与美容资讯于一体的网站较少.因此,以Spring+Spring MVC+My Batis作为基本架构,设计并实现了一个网站.该网站不仅具有基本的用户购物功能,还提供了各种美妆信息供用户参考,以便用户选择合适的商品,同时后台设计简洁直接,便于管理.与一般网站相比,该网站不仅可以帮助用户选择合适的商品,还具有小巧、直接、高效、自动化、延迟较低等特点,进而可以更好地提高用户体验.

【9】【篇名】基于Bootstrap+spring boot框架的在线考试系统开发

【作者】叶惠仙,沈文杰

【刊名】网络安全技术与应用

【中文关键词】在线考试系统;Bootstrap;spring boot框架;信息化;

【中文摘要】摘要：Internet的开放分布性特点使各种考试突破了时间和空间限制,该在线考试系统设计使用Bootstrap+spring boot框架,实现用户管理,题库管理、成绩管理、考试自动组合出卷与评卷管理等功能。本框架能够减少底层开发工作量,系统模块之间具有高内聚、低耦合的特点,进而提升系统的可用性和可维护性。本系统能满足各科课程教学考试考核需求,简化并改善传统纸质考试模式,提高考试效率与学院信息化建设。

【10】【篇名】基于SSM框架+vue的Web网盘系统的设计与实现

【作者】张文龙,吴林辉,杨晨耀,蒋卫祥.

【刊名】电脑知识与技术

【中文关键词】文件上传; SSM; vue,Web网盘系统; 系统设计;

【中文摘要】摘要：针对当前Web网盘系统存在的问题,提出了利用SSM+vue的技术架构,进行技术方案设计。该文首先通过UML的用例图分析了Web网盘系统的需求;其次应用SSM框架设计了系统的架构;前端使用vue实现前后端分离,最后介绍了系统的实现。SSM+vue框架能简化Web网盘系统的开发,可以有效地降低各层之间的耦合度,提高系统的可维护性。

【11】【篇名】The Effect of Multiple Developers on Structural Attributes: A Study Based on Java Software

【作者】Andrea Capiluppi; Nemitari Ajienka; Steve Counsell;

【刊名】The Journal of Systems & Software

【外文关键词】Object oriented; Metrics; Collaborative development; Open source; Software structure;

【外文摘要】Context : Long-term software projects employ different software developers who collaborate on shared artifacts. The accumulation of changes pushed by different developers leave traces on the underlying code, that have an effect on its future maintainability, and even reuse.  
Objective : This study focuses on the how the changes by different developers might have an impact on the code: we investigate whether the work of multiple developers, and their experience, have a visible effect on the structural metrics of the underlying code.  
Method : We consider nine object-oriented (OO) attributes and we measure them in a GitHub sample containing the top 200 ‘forked’ projects. For each of their classes, we evaluated the number of distinct developers contributing to its source code, and their experience in the project.  
Results : We show that the presence of multiple developers working on the same class has a visible effect on the chosen OO metrics, and often in the opposite direction to what the guidelines for each attribute suggest. We also show how the relative experience of developers in a project plays an important role in the distribution of those metrics, and the future maintenance of the Java classes.  
Conclusions : Our results show how distributed development has an effect on the structural attributes of a software system and how the experience of developers plays a fundamental role in that effect. We also discover workarounds and best practices in 4 applied case studies.

附录C 部分源代码

主要代码：

**package** com.ahpu.ssm.controller.admin;  
@Controller  
@RequestMapping(**"/admin"**)  
**public class** AdminController {  
 @Autowired  
 UserService **service**;  
 @Autowired  
 ProductService **service1**;  
 @Autowired  
 OrderService **service2**;  
 @RequestMapping(**"/getuser"**)  
 **public** ModelAndView getuser(**int** curPage){  
 ModelAndView mav=**new** ModelAndView();  
 PageBean page =**service**.listUser( curPage);  
  
 mav.addObject(**"page"**,page);  
 mav.setViewName(**"admin/usermanage/messageuser"**);  
 **return** mav;  
 }  
 @RequestMapping(**"/finduserbyname"**)  
 **public** ModelAndView finduserbyname(String uname){  
 ModelAndView mav=**new** ModelAndView();  
 User u=**service**.finduser(uname);  
  
 mav.addObject(**"u"**,u);  
 mav.setViewName(**"admin/usermanage/finduser"**);  
 **return** mav;  
 }  
 @RequestMapping(**"/deleteuser"**)  
 **public** ModelAndView deleteuser(String uid){  
 ModelAndView mav=**new** ModelAndView();  
 **service**.deleteuser(uid);  
 mav.addObject(**"msg"**, **"删除成功"**);  
 mav.setViewName(**"admin/welcome"**);  
 **return** mav;  
  
 }  
 @RequestMapping(**"/updateuser"**)  
 **public** ModelAndView updateuser(String username){  
 ModelAndView mav=**new** ModelAndView();  
 User user =**service**.finduser(username);  
  
 mav.addObject(**"user"**, user);  
 mav.setViewName(**"admin/usermanage/edit"**);  
 **return** mav;  
  
 }  
 @RequestMapping(**"/updatemessage"**)  
 **public** ModelAndView updatemessage(User user){  
 ModelAndView mav=**new** ModelAndView();  
 **service**.updatemessage(user);  
  
 mav.addObject(**"msg"**, **"修改成功"**);  
 mav.setViewName(**"admin/welcome"**);  
 **return** mav;  
  
 }  
 @RequestMapping(**"/adminLogin"**)  
 **public** String login(){  
 **return "admin/adminLogin"**;  
 }  
 @RequestMapping(**"doLogin"**)  
 @ResponseBody  
 **public** String dologin(@Param(**"admin"**) admin admin,HttpSession session){  
*// admin a=new admin();  
// a.setAcount(acount);  
// a.setPwd(pwd);* **if**(**service**.adminLogin(admin)){  
 session.setAttribute(**"admin"**, admin);  
 String msg=**"yes"**;  
 **return** msg;  
 }**else** {  
 String msg=**"no"**;  
 **return** msg;  
  
 }  
  
 }  
  
   
   
   
 @RequestMapping(**"/addProduct"**)  
 **public** String addProduct() {  
 **return "admin/product/add"**;  
 }  
   
 @RequestMapping(**"/index"**)  
 **public** String index() {  
 **return "admin/home"**;  
 }  
   
 @RequestMapping(**"/top"**)  
 **public** String top() {  
 **return "admin/top"**;  
 }  
 @RequestMapping(**"/bottom"**)  
 **public** String bottom() {  
 **return "admin/bottom"**;  
 }  
 @RequestMapping(**"/left"**)  
 **public** String left() {  
 **return "admin/left"**;  
 }  
 @RequestMapping(**"/welcome"**)  
 **public** String welcome() {  
 **return "admin/welcome"**;  
 }  
 @RequestMapping(**"/msg"**)  
 **public** ModelAndView msg() {  
 **int** s0=**service2**.findcount(0);  
 **int** s1=**service2**.findcount(1);  
 **int** s2=**service2**.findcount(2);  
 **int** s3=**service2**.findcount(3);  
 **int** s4=**service2**.findcount(4);  
 List list= **new** ArrayList();  
  
 list.add(s0);  
 list.add(s1);  
 list.add(s2);  
 list.add(s3);  
 list.add(s4);  
  
 List list1=**new** ArrayList();  
 list1.add(**service**.selectcount(0));  
 System.***out***.println(**service**.selectcount(0));  
 list1.add(**service**.selectcount(1));  
  
 ModelAndView mav=**new** ModelAndView();  
 mav.addObject(**"list"**,list);  
 mav.addObject(**"list1"**,list1);  
 mav.setViewName(**"admin/msg"**);  
 **return** mav;  
 }  
 @RequestMapping(**"/notice"**)  
 **public** ModelAndView notice(){  
 List<notice> list= **service**.findnotice();  
 ModelAndView mav=**new** ModelAndView();  
 mav.setViewName(**"/admin/notice"**);  
 mav.addObject(**"list"** ,list);  
  
  
  
 **return** mav;  
 }  
 @RequestMapping(**"/deletenotice"**)  
 **public** ModelAndView deletenotice(String nid){  
 **service**.deletenotice(nid);  
 ModelAndView mav=**new** ModelAndView();  
 mav.setViewName(**"/admin/welcome"**);  
 mav.addObject(**"msg"** ,**"删除成功"**);  
 **return** mav;  
 }  
 @RequestMapping(**"/updatenotice"**)  
 @ResponseBody  
 **public** String updatenotice(String notice){  
  
 notice n=**new** notice();  
 n.setNotice(notice);  
 n.setTime(**new** Date(System.*currentTimeMillis*()));  
 **service**.addnotice(n);  
  
  
 **return "success"**;  
 }  
  
 @RequestMapping(**"/anser"**)  
 **public** ModelAndView anser(**int** curPage ){  
 ModelAndView mav=**new** ModelAndView();  
 PageBean page=**service1**.findallcomments(curPage);  
 mav.addObject(**"page"**,page);  
 mav.setViewName(**"admin/product/comments"**);  
  
  
 **return** mav;  
 }  
  
 @RequestMapping(**"/yesanser"**)  
 **public** ModelAndView yesanser(**int** curPage ){  
 ModelAndView mav=**new** ModelAndView();  
 PageBean page=**service1**.findallcommentsyes(curPage);  
 mav.addObject(**"page"**,page);  
 mav.setViewName(**"admin/product/commentsyes"**);  
  
  
 **return** mav;  
 }  
 @RequestMapping(**"/updateanser"**)  
 **public** ModelAndView updateanser(String anser,String gid ){  
 ModelAndView mav=**new** ModelAndView();  
 Comments comments=**service1**.findcommentsbygid(gid);  
 System.***out***.println( comments.getComment()  
);  
 comments.setAnser(anser);  
 comments.setState(1);  
 **service1**.updatecomments(comments);  
 mav.addObject(**"msg"**,**"回复成功"**);  
 mav.setViewName(**"admin/welcome"**);  
  
  
 **return** mav;  
 }  
  
}