Universe Scientific Publishing
Computer System Network and Telecommunications,计算机系统网络和电信(1) 2019,3
ISSN：2661－3719(Print)　　ISBN:2661－3727(Online)

基于 SpringMVC 框架的电商平台的设计与实现

黄　睿　段华琼
四川大学锦城学院　计算机与软件学院　四川　成都　611731

DOI：10.18686/jsjxt.v1i3.1264

【摘要】网上购物已经成为人们生活中的一种购物习惯,各种购物平台因此应运而生。本论文研究了一个基于 SpringMVC 框架的电子商务平台的设计与实现。该平台采用 Java 作为主要开发语言,使用 Maven 管理工程,以 Eclipse 作为开发工具,Mysql 作为后台数据库存储用户信息和商品信息。

【关键词】电商平台;设计与实现;

1　引言

信息技术产生和发展改变了人们的消费习惯,使人们从以前人们走遍多家商店挑选商品的生活方式改变为足不出户就可以通过电脑和手机浏览并选购自己喜爱的商品,人们也可以随时随地搜索到最合适的合作伙伴,信息化为人们的工作和生活提供了极大的便利[1]。在线购物系统旨在满足电子商务时代人们对网上购买和购物的需求,能够节省传统店面、店员成本[2]。本文将基于 SpringMVC 框架来设计和实现一个电子商务平台。开展网上购物系统的设计与研究至关重要,不仅能够优化网购流程,并且能够为消费者提供更多元的服务,刺激消费,促进社会经济的长远发展[3]。

2　系统功能设计

系统结构包括 Web 前端和后端。前端实现 Web 网页,显示内容;后端实现数据库的调用、读写等。在系统设计中,客户端和服务器端的通信基于 HTTP 协议完成,HTTP 协议是互联网上应用最广泛的一种网络协议标准[4]。

2.1　前台功能设计

前台是展示给用户使用的界面,主要包括以下功能:

1)登录和注册。用户可以自行注册账号然后登陆平台,已有账号的用户可以直接登录平台跳转到主页。

2)账号信息管理。用户登陆后,可以前往用户信息管理页面修改账号的详细信息。

3)商品搜索。在主页上方的搜索栏和导航栏可以检索用户想要的商品分类,并跳转到相应页面。

4)添加到购物车和结算。挑选好商品后,用户可以添加商品至购物车,在购物车页面用户可以结算购物车内的商品。

2.2　后台功能设计

后台是管理员使用的界面,主要包括以下功能:

1)添加商品。管理员登录后进入商品上架页面,填写好商品的相关信息之后,便可以上架商品。

2)管理商品列表和信息。管理员可以修改商品相关信息,例如修改商品价格。

3　相关技术

本系统采用 SpringMVC 框架,以 ioc(控制反转)和 aop(面向切面编程)为内核,使用 JavaBean 来完成系统的设计。

前端使用了 Vue 框架开发页面以及 BooStrap 框架美化界面,BooStrap 框架是简洁、直观、强大的热门开源框架,该框架提高了页面的可观赏性、美观的页面增强了用户的体验感。Vue 框架具有轻量级框架、双向数据绑定、指令、插件化等特性。通过 AJAX 请求向后端请求数据,返回到前端页面渲染出完整页面,同时通过双向数据绑定在前端和后端实时进行数据更新。

后端开发平台为 Eclipse。Eclipse 是一个开放源代码、基于 Java 的可扩展开发平台。就其本身而言,它只是一个框架和一组服务,通过插件组件构建开发环境。Eclipse 附带了一个标准的插件集,包括 Java 开发工具(Java Development Kit,JDK)。

本系统开发语言为 Java。Java 是一门面向对象编程语言,不仅吸收了 C＋＋语言的各种优点,还摒弃了 C＋＋里难以理解的多继承、指针等概念,因此

Java 语言具有功能强大和简单易用两个特征。

4 数据库设计

本系统共设计了以下几个数据库表:

• 用户信息表:用于储存用户信息。

表 1 用户信息表

字段名	类型	可为空	默认	注释
用户 id	Int	False	Null	用户 id,对应一名用户
昵称	Varchar(20)	False	Null	用户昵称
邮箱	Varchar(50)	False	Null	用户邮箱,使用 check 约束
密码	Varchar(10)	False	Null	用户密码
姓名	Varchar(10)	Ture	Null	用户姓名
手机号	Int(11)	Ture	Null	用户手机号

• 商品信息表:用来储存商品信息,由管理员管理。

表 2 商品信息表

字段名	类型	可为空	默认	注释
商品 ID	Int(8)	False	Null	商品默认 ID 序列号
商品名	Varchar(20)	False	Null	商家设定商品名称
价格	Varchar(10)	False	Null	商家设定价格
商品类别	Varchar(10)	False	Null	该商品所属商品类别序号
商品数量	Varchar(10)	Ture	Null	该商品预购的数量
图片路径	Varchar(255)	Ture	Null	商家上传图片的路径
商品描述	Varchar(255)	Ture	Null	商家上传的商品描述

• 管理员信息表:用来储存管理员的账号信息,构建系统时设定。

表 3 管理员信息表

字段名	类型	可为空	默认	注释
管理员 ID	Int(6)	False	Null	构建系统时初始设定
密码	Varchar(12)	False	Null	管理员登录密码

• 购物车信息表:用来储存用户购物车的信息,由用户操作添加数据。

表 4 购物车信息表

字段名	类型	可为空	默认	注释
用户 id	Int(8)	False	Null	该购物车主的身份证号

续表

字段名	类型	可为空	默认	注释
商品 id	Int(8)	False	Null	购物车里的商品的 id
商品数量	Varchar(10)	Ture	Null	该商品添加到购物车的数量
总价	Varchar(20)	False	Null	总商品的价格

5 系统部分功能实现

5.1 商品主页

用户进入平台的第一个页面就是首页,用户可以在首页浏览商品。



图 1 商品主页

5.2 用户注册



图 2 注册页面

点击注册按钮进入注册页面。用户填写的信息必须规范,且 ID 不能和数据库中的重复。电话等信息也会用正则表达式匹配,避免用户填写不规范的信息。

部分代码:

1)前端 Vue 部分

```
add1() { //用户注册操作,将用户信息添加的方法
if(this.VerificationName()== 1 &&  //验证 name 格式是否符合要求
this.VerificationTruename()== 1 &&
this.VerificationId()== 1 &&//验证 id 格式
this.VerificationEmail() == 1 &&//验证
```

ISSN: 2661-3719(Print) ISBN:2661-3727(Online)

```
email 格式
    this.VerificationPhone()== 1 &&//验证手机号的格式
    this.VerificationPassword()== 1 &&
    this.VerificationPassword2()== 1 )//验证两次密码输入是否一致
      {
      var cus = {
      name:this.Z_name,
      truename:this.Z_truename,
      id:this.Z_id,
      email:this.Z_email,
      phone:this.Z_phone,
      password:this.Z_password
        };
      axios.post('/logincustomers',cus)
        .then(function (response){
          alert("用户注册成功!");
         refesh();
        })
       .catch(function (error){
          console.log(error);
        alert("用户注册失败,请重新输入。");
      });
        this.Z_name = this.Z_truename = this.Z_id = this.Z_email = this.Z_phone = this.Z_password = this.Z_Rpassword = '';
      }
  },
  //用户登录的方法
    yzmm1(){
      var t=this.search1(); //从数据库检索对应 ID 的密码
      if(vm.D_id == null){
        alert("请填写用户 id!");
      }else if(vm.D_password == null){
        alert("请填写用户 password!");
      }else if(t == vm.D_password){
        this.sendid(vm.D_id);
        alert("欢迎用户"+vm.D_id+"!");
        newH();
        session.setAttribute("id",vm.D_id);
      }else{
        alert("用户账号或密码错误!");
      }
    },
```

5.3 后端主要方法代码

1)从后端请求数据的方法,此方法在前端通过 AJAX 从数据库中调取相应数据,再传递到前端页面

```
@GetMapping("/customers")
public Object getCus() {
  String  sql = "SELECT * FROM customaccount";
    List<Map<String, Object>> list = jdbcTemplate.queryForList(sql);
    return list;
}
```

2)将前端通过 AJAX 传递的数据,保存到数据库相应的数据表中。@PostMapping("/customers")

```
public Object addCus(@RequestBody Customer cus) {
    String sql = "INSERT INTO customaccount (name, truename, email, password, phone) values (?,?,?,?,?)";
      jdbcTemplate.update(sql,new Object[]{cus.getName(),cus.getTruename(),c      us.getEmail(), cus.getPassword(), cus.getPhone()});
      String sqlQuery = "SELECT * FROM customaccount where name = ?";
      List<Map<String, Object>> list = jdbcTemplate.queryForList(sqlQuery, new Object[]{cus.getName()});
    return list.get(0);
}
```

5.4 购物车页面

用户可以在购物车页面查看添加到购物车的商品,添加到购物车的商品信息都是从数据库中调取的,调取信息到前端后,在前端进行一个数据分析,再渲染到页面上。在 Vue 双向绑定的技术下,可以实现数据库和页面显示的数据同步一致。只有登录账号才能打开购物车页面,才能把商品添加到购物

ISSN: 2661-3719(Print) ISBN:2661-3727(Online)

车,且不能超过商品总数。在购物车页面可以修改购买的数量,也可以删除商品

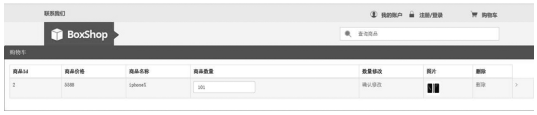


图 3 购物车页面

部分代码(h5+Vue):

```
<tr v-for="item in search(keywords)" :key="item.id">
    <td>{{ item.goodsid}}</td>
    <td>{{ item.price}}</td>
    <td>{{ item.goodname }}</td>
    <td><input type="text" class="form-control" v-model="item.goodnum"></td>
    <td><a href="" @click.prevent="put(item.id,item.goodsid,item.goodnum)">确认修改</a></td>
    <td><img :src="item.picture" width="40px" height="40px" alt="icon"></td>
    <td><a href="" @click.prevent="del(item.goodsid)">删除</a></td>
    <td></td>
</tr>
methods: {
  del(goodsid) {
  let uri = '/cart/' + goodsid;
  axios.delete(uri)
    .then(function(response){
      vm.query();
    })
    .catch(function(error){
      console.log(error);
    });
  put(q,w,e){
  var goods = { id: q, goodsid: w, goodnum: e}
  let uri = '/cart';
  axios.put('/cart',goods)
    .then(function(response){
      vm.query();
    })
    .catch(function(error){
      console.log(error);
    });
  },
}
```

5.5 添加商品页面

管理员可以在这个页面添加商品,录入相关的商品信息然后点击添加,触发 AJAX 请求将商品信息保存到数据库的对应的数据表中

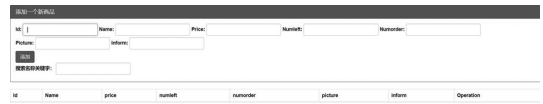


图 4 添加商品页面

部分代码:

```
add() { //添加商品的方法
  var id = parseInt(this.id);
  var price = parseFloat(this.price);
  var numleft = parseInt(this.numleft);
  var numorder = parseInt(this.numorder);
  var go = {
    goodId: id,
    goodName: this.name,
    goodPrice: price,
    goodNumY: numleft,
    goodNumS: numorder,
    picture: this.picture,
    inform: this.inform
  }
  axios.post('/goods', go)
    .then(function(response) {
      vm.query();
      alert("changdushi:"+vm.list.length);
  })
  .catch(function(error) {
    console.log(error);
    });
  this.id = this.name = this.price = this.numleft = this.numorder = this.picture = this.inform = ''
    },
```

5.6 用户管理页面

管理员通过此页面管理用户信息,注册的用户一览无余。可以对用户账号进行检索,然后便可以

ISSN：2661－3719(Print)　　ISBN:2661－3727(Online)

对账号进行删除和修改操作。

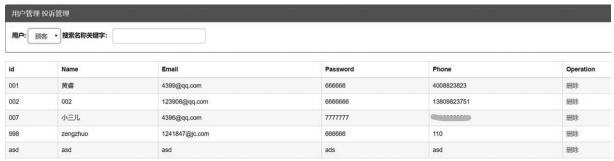


图 5　管理用户信息页面

前端 Vue 向服务器请求数据的代码：

```
//调取所有用户的方法
query1(){
axios.get('/logincustomers')
  .then(function (response){
    console.log(response);
    vm.list1 = response.data;
  })
  .catch(function (error){
    console.log(error);
  });
},
```

6　结束语

信息技术的升级与变革，使计算机网络技术等广泛地应用于诸多领域中，电子商务因此走进了人们的日常生活中，并成为了社会不可或缺的部分[3]。社会经济与科学技术的发展，是网上购物发展的必要前提和基础，同样的网上购系统的设计与开发也要服务和作用于社会经济、科学技术和人们生活[5]。因此，本文基于 SpringMVC 框架设计和实现了一个电子商务平台，该平台设计和实现了用户注册、用户管理、添加商品、商品管理、购物车、选购商品等功能。经过测试，平台运行稳定，功能实现完整，但是由于开发成本较低、开发人员经验不丰富，本平台的界面和功能设计粗略，人机交互体验有待提升。

【参考文献】

[1]王晟宇，叶传奇，刘迪，等. 基于 Java 技术的 B2C 网上购物商城系统的设计与实现_王晟宇[J]. 电脑知识与技术，2019，卷缺失(7)：288－290.

[2]董萍. 基于 J2EE 在线购物系统的设计与实现_董萍[J]. 宁波职业技术学院学报，2018，22(5)：96－99.

[3]李淑娣. 基于 PHP 技术的网上购物系统的设计与实现[J]. 科学咨询(科技・管理)，2019，卷缺失(3)：56.

[4]李雪莲. 基于 Android 混合开发模式的网上购物系统的设计与实现_李雪莲[J]. 电子设计工程，2018，26(5)：23－27.

[5]卢露. 网上购物系统的设计与开发探析[J]. 电脑迷，2016，卷缺失(11)：104.