

# 茂业百货无线应用系统服务器 设计与实现

**摘要：**茂业百货无线应用软件为绵阳茂业百货公司定制开发，茂业百货无线应用系统服务器为客户端无线应用软件提供数据和管理服务。该无线应用服务器主要包括商家管理、商品管理、促销管理、时尚元素、消息管理等功能。同时为了便于信息的分享与传播，客户端利用了腾讯、新浪微博等平台的开放 API，实现了将信息实时分享到这些平台的功能。

系统采用 MySQL5.1 做为数据库管理系统，GlassFish3 作为应用服务器。采用目前大量使用的基于 JavaEE 的 Strust2+Spring3+JPA 等框架进行构建，采用 Ajax 技术实现异步传输，JSON 作为信息传递格式。

论文主要围绕茂业百货无线应用系统服务器设计与实现过程进行阐述，内容包含了需求分析、概要设计和详细设计，并对系统进行了详细测试，测试结果表明实现内容满足设计需求。

**关键词：**JavaEE； 无线应用； 应用服务器； 茂业百货

# Design and Implementation of Maoye Department Store Wireless Application Server

**Abstract:** Maoye Department Store Wireless Application software is developed for the Maoye Department Store MianYang company. Maoye Department Store Wireless Application Server Provide data and management services to the Wireless Client Application of MaoYe Department Store MianYang company. This Application Server mainly includes company management, commodity management, sales promotion management, fashion management, message management, etc. At the same time in order to facilitate the information sharing and communication, the Client Application used the open API of Tencent and micro-blog of Sina implements the function of real-time sharing of information to the platform.

System development using MYSQL5.1 database, and using Glassfish3 as the web server. Adoption of the extensive use of Strut2 + Spring3 + JPA framework which based on JavaEE for building, etc. Using Ajax technology to realize asynchronous transmission and JSON as message format.

This paper describes the design and realization of Maoye Department Store Wireless Application Server, Content includes requirement analysis, general design and detailed design, and make a detailed test to the system, the test results show that the system meets the design requirements.

**Key words:** JavaEE, Wireless Applications, Application Server, Maoye Department Store

# 目 录

第 1 章 绪 论 .....	1
1.1 概述 .....	1
1.2 选题背景及目的 .....	1
1.3 相关技术 .....	2
1.3.1 Struts2 .....	2
1.3.2 Spring3 .....	2
1.3.3 JPA .....	3
1.3.4 Spring Security .....	3
1.3.5 Ajax .....	3
1.3.6 jQuery EasyUI .....	3
1.3.7 JSON .....	3
1.4 系统开发环境介绍 .....	3
第 2 章 茂业百货无线应用需求分析 .....	5
2.1 需求概述 .....	5
2.2 系统功能性需求 .....	5
2.3 系统功能模块划分 .....	6
2.4 用例文档 .....	7
2.4.1 普通用户用例图 .....	8
2.4.2 系统管理员用例图 .....	9
2.5 系统非功能性需求 .....	9
2.5.1 系统可靠性 .....	9
2.5.2 系统可扩展性 .....	9
2.5.3 系统可维护性 .....	9
2.5.4 系统安全性 .....	10
2.5.5 系统可交互性 .....	10
第 3 章 茂业百货无线应用概要设计 .....	11
3.1 系统架构设计 .....	11

3.1.1 系统表现层的分析与设计 .....	12
3.1.2 系统控制层的分析与设计 .....	12
3.1.3 系统业务逻辑层的分析与设计 .....	12
3.1.4 系统持久层的分析与设计 .....	13
3.2 数据库设计 .....	13
3.2.1 数据库需求分析 .....	13
3.2.2 数据库逻辑设计 .....	14
3.2.3 数据库物理设计 .....	20
第 4 章 茂业百货无线应用服务器系统详细设计与实现 .....	30
4.1 Struts2+Spring3+JPA 架构搭建 .....	30
4.1.1 Struts2 的基本配置 .....	30
4.1.2 Spring3 的基本配置 .....	30
4.1.3 JPA 的基本配置 .....	31
4.1.4 Spring Security 的基本配置 .....	33
4.2 用户登录服务器的设计与实现 .....	36
4.2.1 用户登录页面设计 .....	36
4.2.2 验证码的生成和检验 .....	38
4.2.3 控制层的实现 .....	38
4.2.4 业务逻辑层的实现 .....	38
4.2.5 持久层的实现 .....	38
4.3 海报管理的设计与实现 .....	39
4.3.1 海报管理页面设计 .....	39
4.3.2 控制层的实现 .....	40
4.3.3 业务逻辑层的实现 .....	41
4.3.4 持久层的实现 .....	41
4.4 专柜分类管理的设计与实现 .....	42
4.4.1 专柜分类管理的页面设计 .....	42
4.4.2 控制层的实现 .....	43
4.4.3 业务逻辑层的实现 .....	44
4.4.4 持久层的实现 .....	45

4.5 专柜管理的设计与实现 .....	45
4.5.1 专柜管理的页面设计 .....	45
4.5.2 控制层的实现 .....	46
4.5.3 业务逻辑层的实现 .....	48
4.5.4 持久层的实现 .....	49
4.6 商品管理的设计与实现 .....	49
4.6.1 商品管理的页面设计 .....	49
4.6.2 控制层的实现 .....	51
4.6.3 业务逻辑层的实现 .....	52
4.6.4 持久层的实现 .....	52
4.7 促销管理的设计与实现 .....	53
4.7.1 促销管理的页面设计 .....	53
4.7.2 控制层的实现 .....	54
4.7.3 业务逻辑层的实现 .....	55
4.7.4 持久层的实现 .....	55
4.8 时尚讯息管理的设计与实现 .....	55
4.9 投诉建议管理的设计与实现 .....	55
<b>第 5 章 系统运行测试 .....</b>	<b>56</b>
5.1 系统测试环境 .....	56
5.1.1 系统测试硬件环境 .....	56
5.1.2 系统测试软件环境 .....	56
5.1.3 GlassFish 服务器的安装和配置 .....	56
5.2 服务器系统测试 .....	56
5.2.1 测试需求概述 .....	56
5.2.2 测试用例及结果 .....	57
5.2.3 测试总结 .....	61
<b>结 论 .....</b>	<b>62</b>
<b>致 谢 .....</b>	<b>63</b>
<b>参考文献 .....</b>	<b>64</b>

# 第 1 章 绪 论

## 1.1 概述

茂业百货无线应用是展示绵阳茂业百货商城的商家、最新动态信息的移动客户端软件，为该客户端提供服务的茂业百货无线应用服务器系统实现了商家管理、商品管理、促销管理、时尚元素、消息管理等功能。

茂业百货无线应用的开发是为了让用户对绵阳茂业百货有更加深入的了解，方便用户实时的随地的关注绵阳茂业百货的动态。而且，绵阳茂业百货还可通过该无线应用获取用户的反馈信息对自己的服务进行改进，以更好的姿态去服务消费者，另一方面，通过用无线应用的分享功能，可以巧妙借用户之手宣传自己，降低自己的宣传成本。

茂业百货无线应用服务器端系统，具有稳定性、可拓展性、可维护性等特点。

## 1.2 选题背景及目的

当今，移动互联网发展速度迅猛，3G、Wi-Fi 以及最近的 4G 技术在生活中随处可见，智能手机、平板电脑等移动终端已经得到普及，用户数量的增长呈爆发式<sup>[1]</sup>，可见移动互联时代已经来临。

伴随着移动互联网的迅速发展，越来越多的企业和开发者注意到了移动互联所带来的商机，移动电子商务已经成为现实。由于移动通讯设备可以随时随地的接入互联网，便于携带，越来越受到大众的青睐<sup>[15]</sup>，移动商务成为下一个商业竞争的主要战场。

茂业百货是一家知名专业的百货零售企业，成立于 1996 年，经过十余年的长足发展，在全国华南、西南、华北、华东区域的 18 个城市共拥有 38 家门店。系统为绵阳茂业百货定制，通过移动客户端，茂业百货可以为客户提供方便快捷的信息服务，同时，绵阳茂业百货也可以通过该渠道更好的去宣传自己的最新动态。用户通过客户端的分享功能将信息分享到腾讯 QQ、新浪微博等，可帮助绵阳茂业百货得到更好的宣传。

茂业百货无线应用系统服务器提供了对无线客户端所需信息的服务功能，主要的功能有商家管理、商品管理、促销管理、时尚元素、消息管理等，通过这些功能，可以使用户随时随地的了解到绵阳茂业百货最新的动态信息，为用户带来前所未有的体

验。

## 1.3 相关技术

### 1.3.1 Struts2

Struts2 是一经典的 MVC 框架<sup>[6]</sup>，以 WebWork 为核心，采用拦截器的机制来处理用户的请求，使得业务逻辑控制器与 ServletAPI 完全分离，提供极好的可扩展性，并且提供了良好的输入校验功能，以及 Ajax 的支持，Struts2 的工作流程如下：

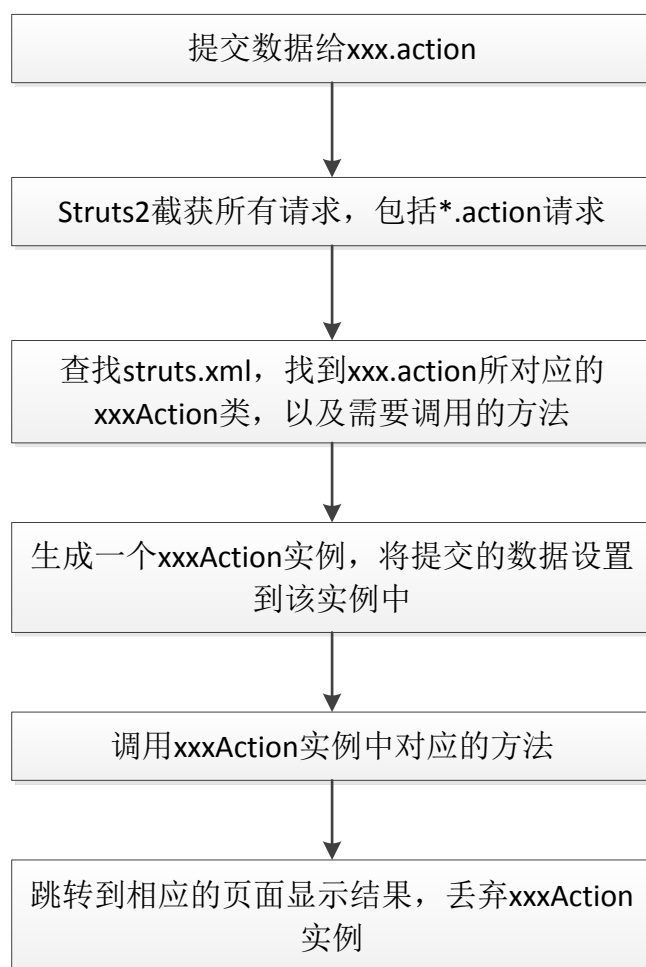


图 1-1 Struts2 工作流程

### 1.3.2 Spring3

Spring 是一个轻量级的框架，不需要特殊容器的支持，Spring 不提供某种功能，它只是将所有的组件部署到它之中，管理，维护，执行他们。Spring 最主要的思想是 IoC(Inversion of Control, 控制反转、反向控制)<sup>[10]</sup>，或者叫做 DI(Dependency Injection,

依赖注入)。Spring 通过反向控制实现了松耦合,一个对象依赖的其他对象会通过被动的方式传递进来,而不是这个对象自己创建或查找已有对象, Spring 提供了面向切面编程的支持,通过分离业务逻辑和系统级服务进行内聚性的开发。使用这些特性可以使代码更干净、更易扩展、和更易调试。

### 1.3.3 JPA

JPA 全称 Java Persistence API,是 Java 官方提出的 Java 持久化标准。JPA 只是一种 ORM 规范,并没有具体的实现代码<sup>[12]</sup>。Hibernate、TopLink 等 ORM 框架都实现了 JPA 规范<sup>[5]</sup>。JPA 支持 XML 和 JDK5.0 注解两种元数据的形式,元数据描述元素和表之间的映射关系,框架据此将实体对象持久化到数据库表中。

### 1.3.4 Spring Security

Spring Security 是一个能够为基于 Spring 的企业应用系统提供声明式的安全访问控制解决方案的安全框架,它提供了一组可以在 Spring 应用上下文中配置的 Bean,充分利用了 Spring 的控制反转核心思想和面向切面编程的功能。为系统提供了声明式的安全控制功能,减少了重复大量重复编码的工作。

### 1.3.5 Ajax

Ajax (Asynchronous Javascript + XML) 是一种创建交互式网页应用的网页开发技术,可以使网页在不重新全部加载的情况下,对网页的某部分进行更新。

### 1.3.6 jQuery EasyUI

jQuery EasyUI 是一组基于 jQuery 的 UI 插件集合,开发者只需要了解一些简单的 HTML 标签,并不需要编写复杂的 Javascript 和 CSS,就可实现美观的 UI 界面,大大的加快了系统的开发速度和开发效率。

### 1.3.7 JSON

JSON (Javascript Object Notation) 是一种轻量级的数据交换格式<sup>[1]</sup>,JSON 采用完全独立于语言的格式,使其成为理想的数据交换格式,JSON 基于 Javascript 的一个子集,这使得 Javascript 解析 JSON 非常方便。

## 1.4 系统开发环境介绍

数据库服务器: MySql5.1



开发 IDE: MyEclipse8.6

应用服务器: GlassFish3

开发框架: Struts2、Spring3、JPA

## 第 2 章 茂业百货无线应用需求分析

### 2.1 需求概述

茂业百货无线客户端提供绵阳茂业百货最新的商家信息、活动信息、时尚信息、品牌信息等。用户可以通过自己的手机客户端随时随地的把商家信息和茂业百货无线应用客户端分享到腾讯 QQ，新浪微博，微信等平台。

茂业百货无线应用服务器端需要提供对绵阳茂业百货的商家管理、商品管理、促销管理、时尚讯息管理、消息管理和反馈信息管理。

### 2.2 系统功能性需求

茂业百货无线应用客户端部分提供以下主要功能：

1. 海报展示：通过首页的海报展示，用户可以浏览绵阳茂业百货的海报
2. 商家展示：通过首页的链接，用户可以浏览到绵阳茂业百货中的商家的介绍
3. 流行风格：展示绵阳茂业百货向用户推荐的潮流时尚情况
4. 促销列表：显示当前绵阳茂业百货的促销活动信息
5. 商家分层展示：通过绵阳茂业百货中的商家位置，进行商家的展示
6. 品牌展示：通过商家的分类，对商家进行分类显示，方便用户查看
7. 搜索商品：通过搜索关键字，对商家的商品进行搜索
8. 分享商家：把绵阳茂业百货中的商家分享到微信、QQ、新浪微博等平台
9. 分享活动：把促销信息分享到微信、QQ、新浪微博等平台
10. 分享商品：可把商品信息分享到微信、QQ、新浪微博等平台
11. 软件分享：将绵阳茂业百货客户端软件分享到微信、QQ、新浪微博等平台
12. 意见反馈：用户可提交对绵阳茂业百货的意见
13. 用户注册和登录

茂业百货无线应用服务器端部分提供以下主要功能：

1. 登录：管理员登录系统后，可进入对系统信息管理的页面

2. 商家管理：可修改绵阳茂业百货中的所有商家专柜的信息
3. 商品管理：可对商品信息进行添加、修改、删除等操作
4. 促销管理：可对促销折扣等信息进行添加、修改、删除等操作
5. 时尚元素：可对最新的时尚信息进行管理
6. 消息管理：可添加最新的消息
7. 反馈管理：可查看用户反馈的意见信息

## 2.3 系统功能模块划分

根据系统功能的分析可以画出服务器端功能模块图：

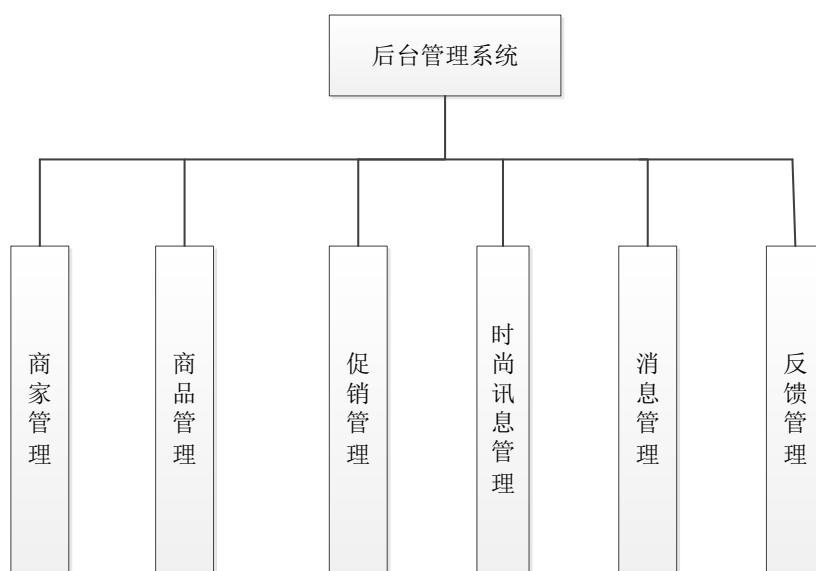


图 2-1 系统功能模块图

1. 商家管理：商家管理的内容包括海报管理（客户端首页的海报）、专柜管理（专柜即商家），专柜分类管理。系统管理员可对绵阳茂业百货中的所有商家设置分类、海报、介绍、专柜位置、联系方式等一系列的信息；
2. 商品管理：系统管理员可以添加商品，其中商品的信息包括商品名称、所属专柜、商品缩略图、商品图片；
3. 促销管理：促销管理可对最新的促销活动信息进行管理；
4. 时尚信息管理：时尚信息管理主要是系统管理员对一些最新的时尚信息进行

管理，向用户推荐最新的时尚潮流信息；

5. 消息管理： 消息管理主要是向用户推送最新的关于茂业百货最新的消息，系统管理员可对消息进行添加、修改、删除操作；
6. 反馈管理： 反馈管理主要是查看手机客户端软件用户对绵阳茂业百货的意见等信息。

## 2.4 用例文档

用例用于描述系统中所有用户的功能。

用例文档详细的描述了系统用例参与的业务，该系统主要的参与者有：

1. 普通用户：浏览绵阳茂业百货的信息，登录后可对商家专柜信息、促销信息、商品信息等进行分享等；
2. 系统管理员：管理绵阳茂业百货中的所有商家专柜的信息，商品信息，促销活动信息等。

系统用户角色用例图，展示了用户之间的泛化关系，泛化关系就是通常理解的继承关系，如图 2-2 所示：

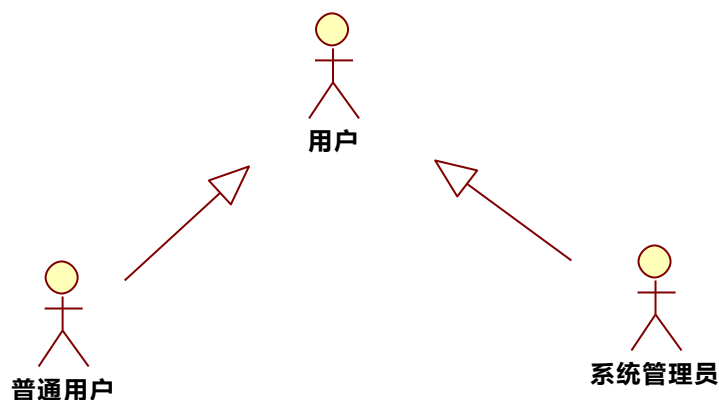


图 2-2 系统用户角色关系图

用例图描述了用户、需求、系统功能单元之间的关系，可以帮助开发者用可视化的方式理解系统的功能需求，系统主要包括以下用例。

### 2.4.1 普通用户用例图

普通用户可以通过客户端软件对系统的信息进行浏览,还可对商家和商品信息进行分享,其用例图如图 2-3 所示:

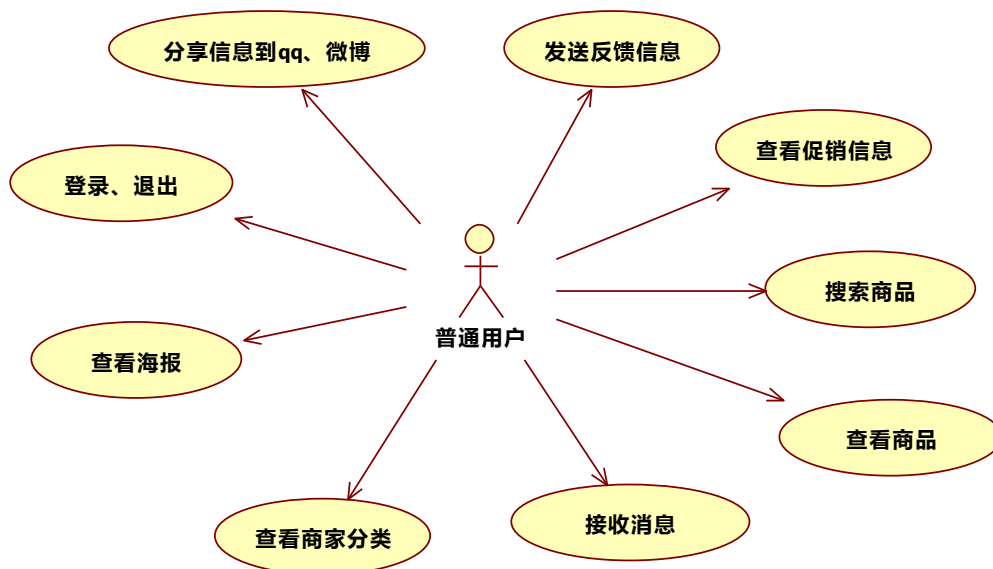


图 2-3 普通用户用例图

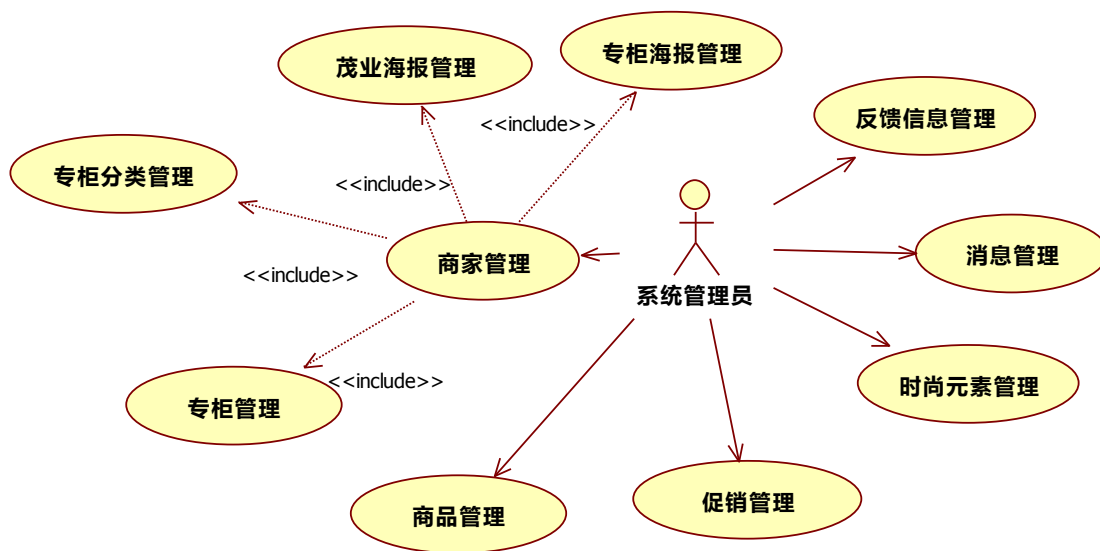


图 2-4 系统管理员用例图

### 2.4.2 系统管理员用例图

系统管理员可对系统中的信息进行管理，主要是对商家信息、商品信息、促销、反馈等信息进行添加、修改以及删除操作，保证系统数据一直处于最新状态，系统管理员用例图如图 2-4 所示。

## 2.5 系统非功能性需求

### 2.5.1 系统可靠性

系统能否正常运行取决于系统的可靠性，茂业百货无线应用系统使用了强健的商业兼容应用服务器 GlassFish，该应用服务器达到产品级质量，可保证系统的正常运行环境。数据库系统采用 MySQL 数据库，MySQL 数据库是一款支持 ACID 即：事务的原子性、一致性、独立性、持久性的 InnoDB（数据库引擎）的事务性数据库，采用该数据库保证了对数据进行操作的正确安全等要求。

### 2.5.2 系统可扩展性

在一个系统开发完成之后，客户可能会需要新的需求，在这种情况下，系统拥有良好的可扩展性十分重要。茂业百货无线应用系统使用 Ajax+Struts2+Spring3+JPA 架构，结构清晰，分层明确，以控制层为例，控制层采用了 Struts2 框架实现，一方面封装通用的 BaseAction，所有的 action 全部继承它，保证了实现 action 的简单性，另一方面 action 的实现全部采用模型驱动的方式，使客户端与服务器端的参数传递相当明确，并且结合 Spring3 开源框架的依赖注入重要特性，实例化 action 需要的 service 层对象，大大的降低了类与类之间的耦合度，而且系统采用 Spring 的接口编程，也提高了系统的可扩展性。

### 2.5.3 系统可维护性

茂业百货无线应用系统采用 Ajax+Struts2+Spring3+JPA 架构实现，系统分层清晰明确，在对系统进行维护中，如果要修改代码，可很快的定位到要修改的文件，系统中的类名、方法名、属性名全部按照所要实现的功能进行命名<sup>[9]</sup>，并且对方法所要实现的功能以及参数，返回值进行了明确的注释说明，方便了维护人员阅读理解，系统实现的代码中使用的面向对象的封装、继承和多态思想<sup>[14]</sup>，大大提高了系统的可维护性。

#### 2.5.4 系统安全性

茂业百货无线应用系统使用 Spring Security, Spring Security 是一个能够为基于 Spring 的企业应用系统提供声明式的安全访问控制解决方案的安全框架, 通过使用 Spring Security 对用户登录系统时的权限进行控制, 保证了系统安全, 并且用户的密码全部进行 MD5 不可逆的加密, 使用户的密码安全性得到了保障。

系统登录验证码的使用, 能有效的防止恶意攻击者通过机器注册灌水, 机器登录来猜测系统用户的登录名和密码。

系统采用 Log4j 方便细致的控制日志的生成过程, 有效的将系统操作人员对系统数据的添加、删除和修改操作详细的记录下来, 保证了系统操作的安全性。

在使用第三方软件进行分享的上, 采用了 OAuth2.0 协议进行了认证, 并采用了 HTTPS 请求数据的方式来保护用户密码等信息的安全。

#### 2.5.5 系统可交互性

茂业百货无线应用服务器设计与实现, 在手机客户端使用了原生的 HTML5 技术, 以及 jQuery 前端框架, 数据请求全部采用异步的 Ajax, 保证的页面的美观和请求响应的迅速, 服务器端采用 jQuery EasyUI 保证了数据管理页面的美观以及实用性。

## 第3章 茂业百货无线应用概要设计

### 3.1 系统架构设计

茂业百货无线应用系统服务器设计与实现采用 Struts2+Spring3+JPA 架构，系统分层实现，条理清晰，其中 Struts2 充当控制层，Spring3 充当业务逻辑层，JPA 为持久层。系统整体架构如图所示：

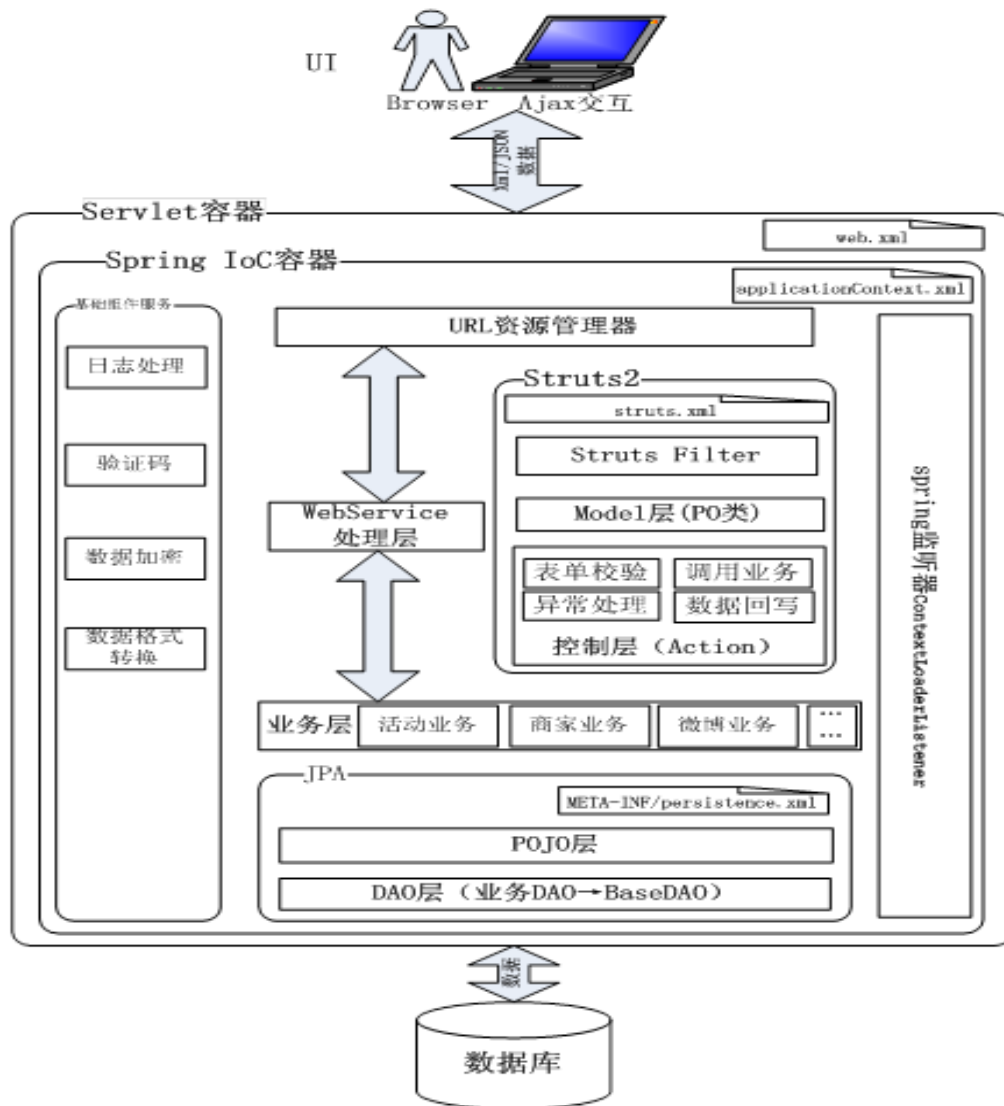


图 3-1 系统分层结构示意图



### 3.1.1 系统表现层的分析与设计

系统表现层为用户看到的页面，用于用户发出请求并显示处理的结果。茂业百货无线应用系统服务器的表现层主要采用 JSP 技术来编写数据的管理页面。使用 JSP 技术编写页面显示数据需要大量的使用 Struts2 标签和 EL 表达式等，使页面的代码相当臃肿，不方便之后的维护和扩展。为了解决这个问题，系统引进了 jQuery-EasyUI。

jQuery-EasyUI 是一组基于 jQuery 的 UI 插件集合，可帮助 web 开发者开发出功能丰富并且美观的 UI 界面。采用 JSP 页面插入 Javascript 脚本的方式，动态的加载 EasyUI 组件，用 EasyUI 组件显示请求的数据结果，使数据的显示变的清晰，使 JSP 页面的结构更加明了，方便了以后进行扩展。

### 3.1.2 系统控制层的分析与设计

系统的控制层使用 Struts2 框架，负责接收前台的请求，调用业务逻辑层组件处理请求并将处理后的数据进行格式转换，以及表单验证，异常处理等<sup>[7]</sup>。Struts2 是经典的 MVC 框架，采用拦截器的机制来处理用户的请求，这使得业务逻辑控制器可以与 Servlet API 完全分离<sup>[3]</sup>，Struts2 实现的控制层，处理 Model 层（数据模型）和 View 层（表现层）之间的交互，实现 Action 类，并在配置文件中配置请求的逻辑地址，供前台进行请求使用。Struts2 实现的控制层主要有以下的功能：

1. 通过拦截器拦截所有的请求并查询配置文件找到处理该请求的 Action 类，以及需要调用的方法；
2. 通过 Model 数据模型获取请求的参数；
3. 调用相应的业务逻辑方法，处理请求并获取结果；
4. 将处理结果数据进行封装转换，返回处理的结果。

### 3.1.3 系统业务逻辑层的分析与设计

系统的业务逻辑层采用 Spring3 框架实现，该层主要调用持久层的方法，并提供给控制层调用。该层实现各种 Service 接口，并通过使用 Spring 的 IoC（控制反转）技术托管业务逻辑层的所有对象<sup>[4]</sup>，在系统启动时，根据 Spring 配置文件，动态的生成相关类的对象，并将这些对象提供给需要它们的控制层对象（Action）使用，即注入到 Action 对象之中，通过使用 Spring 的控制反转技术，将对象的生成和销毁托给

Spring 进行管理<sup>[8]</sup>，程序的开发者不必关心对象自身的生命周期和其他的各种关系，这样使程序结构层次更加清晰明了，降低了层次之间的耦合度，大大的提高了程序的开发速率。

Spring 的另外一个重要特性是 AOP（Aspect Oriented Programming，面向切面编程）技术<sup>[10]</sup>，AOP 是 Spring 实现的拦截器技术，通过使用 Spring 的 AOP 技术可对业务逻辑层的非查询方法进行事务管理，以及将与业务逻辑层无关但和系统完整性相关的部分分离出来，使系统更加完整的同时，层次清晰。

### 3.1.4 系统持久层的分析与设计

系统持久层使用 JPA 框架实现，该层主要实现 POJO（Plain Old Java Objects，简单 Java 对象）和 DAO（Data Access Objects，数据访问对象）。该层负责访问数据库，对数据库进行操作，供业务逻辑层调用。

POJO 类通过 Entity 注解使之成为与数据库中的关系表一一对应的持久化实体类。POJO 类中的每一个属性都对应于相应关系表中的某一字段，通过为每一属性提供 getter 和 setter 方法对属性进行取值和设值操作。通过使用 POJO 对象的方式对数据进行持久化，使程序代码的可重用性增强，当数据库的结构发生改变的时候，只需修改相应的 POJO 对象，对程序代码结构不产生影响，增强了系统的可扩展性。

DAO 层通过使用 Java 的泛型技术实现了 BaseDAO 类，该类封装了系统会广泛用到的数据持久化方法（增删改查），其他的 DAO 类采用不同的 POJO 实体类，并继承 BaseDAO 类，使之成为与 POJO 类对应的实体持久类，可实现与该实体相关的一些特殊要求操作方法，所有的 DAO 类对象都使用 Spring 控制反转技术，通过配置文件，将业务逻辑层需要的相应对象注入到其中。

## 3.2 数据库设计

### 3.2.1 数据库需求分析

茂业百货无线应用系统面向的用户有两种：客户端用户和系统管理员。数据库需求就从这两种用户着手分析。

1. 客户端用户：通过手机客户端可以查看绵阳茂业百货中的所有专柜信息，促销信息，商品信息，搜索商品等；

2. 系统管理员：可以对绵阳茂业百货中的商家、商品、促销、消息等进行管理，其中商家管理包括对海报、专柜、专柜分类、专柜海报的管理，另外系统管理员可通过系统的后台添加时尚潮流信息，供客户端用户了解。

根据系统用户实体的需求，经过分析可得出系统主要有以下几部分构成：用户模块、商家模块、商品模块、促销模块、消息反馈模块、消息推送模块、海报模块。

用户模块涉及到用户的权限的控制，结合 Spring Security 由此派生出以下的实体：权限、资源，为了对用户所拥有的权限进行方便的管理，考虑到对用户进行分组，由此派生出：用户组，菜单组，用户组权限等实体。

商家模块的设计根据需求需要对商家进行分类，由此需要商家、商家分类、以及商家和商家分类之间的关系等实体。

商品模块的设计需要对商品进行分类，其中需要商品、商品分类、商品和商品分类之间的关系、商品图片等实体。

促销模块的设计关系到商家和商家的活动，以及活动中需要的海报。

消息反馈模块的设计涉及到用户以及消息实体和商家之间的关系。

消息推送模块的设计关系到用户以及商家和发送的消息。

海报模块的设计比较简单主要关系到商家和海报以及商家和海报之间的关系等。

### 3.2.2 数据库逻辑设计

根据系统的需求，结合对系统实体的分析以及实体之间的关系，画出 E-R 图，对 E-R 图进行分析化简，可得出系统的逻辑模型<sup>[2]</sup>，以下介绍系统涉及到的五个逻辑模块的逻辑模型。

#### 1. 用户模块

用户模块涉及到用户、商家、用户权限，根据用户权限管理的解决方案，权限管理又抽象出权限、资源、资源与权限之间的关系，由于各个用户所管理的菜单不同，需要对用户进行分组，需要用户组、菜单、用户组和菜单之间关系，最后需要用户和用户之间的关系，形成一个整体。其中菜单和用户组之间是多对多的关系，增加了菜单与用户组之间的关系表，权限与资源之间也是多对多的关系，增加了权限与资源之间的关系表，如图 3-2 所示是用户模块设计的逻辑模型图。

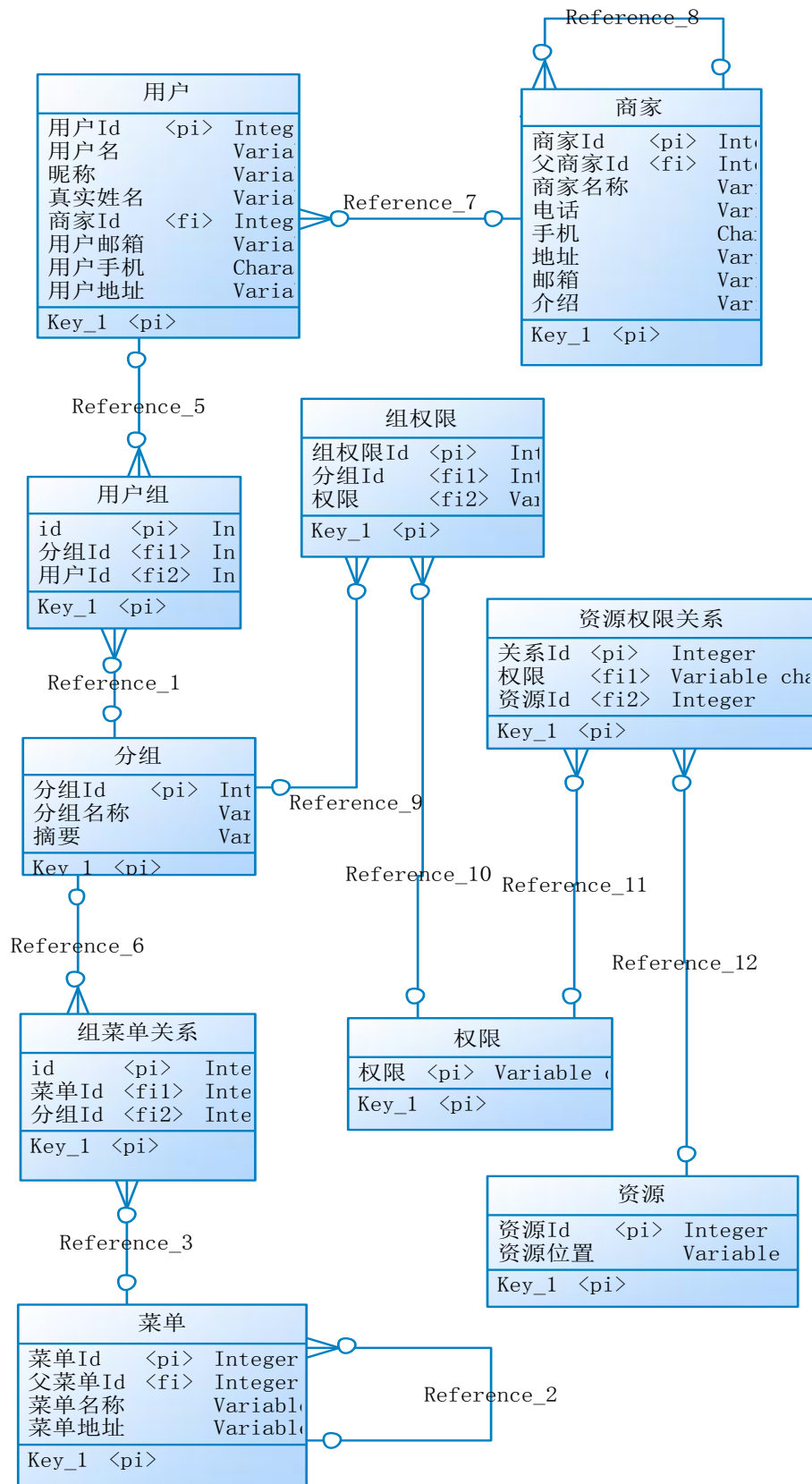


图 3-2 用户模块逻辑图

## 2. 商家模块

商家管理主要是商家的分类的管理，包括的实体有商家、商家分类以及商家与商家分类之间的关系。商家模块的逻辑图如图 3-3 所示：

## 3. 商品模块

商品模块包括商品、商家、商品分类、商品和商品分类关系、商品图片、海报，由于商品和商品分类之间的关系是多对多的关系，增加了两者之间的关系表，商品模块对应的逻辑图如图 3-4 所示：

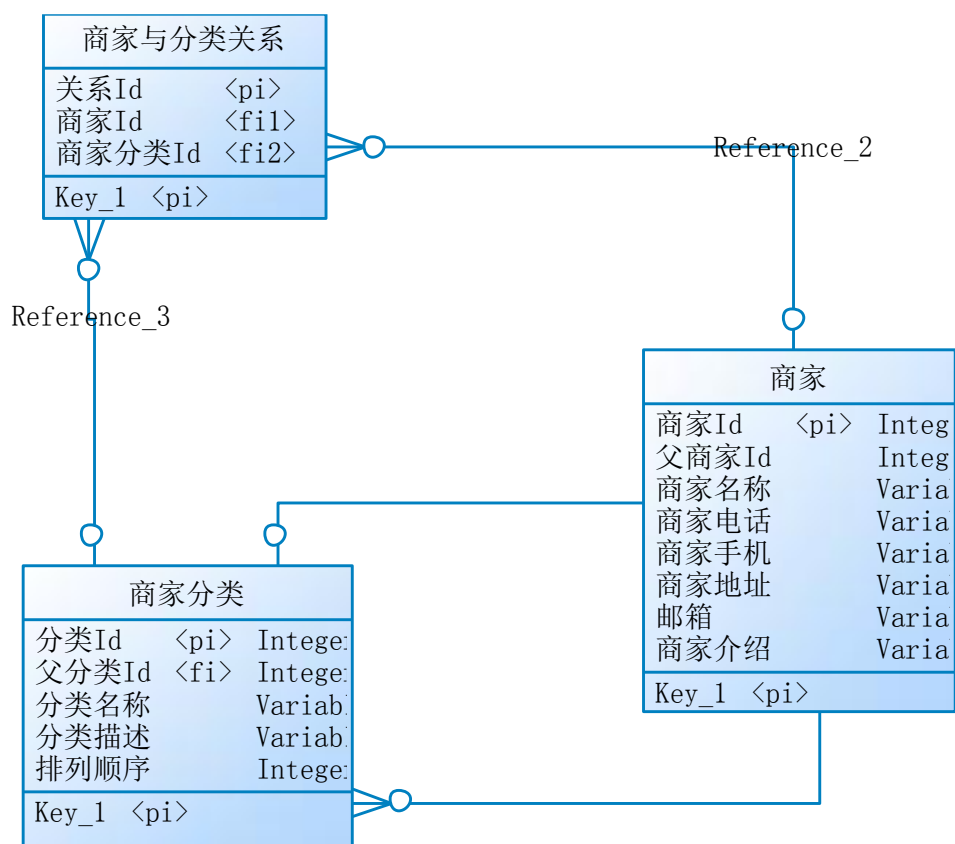


图 3-3 商家模块逻辑图

## 4. 促销模块

促销模块关系到促销活动表、商家以及活动所需要的图片。该模块的逻辑图如图 3-5 所示：

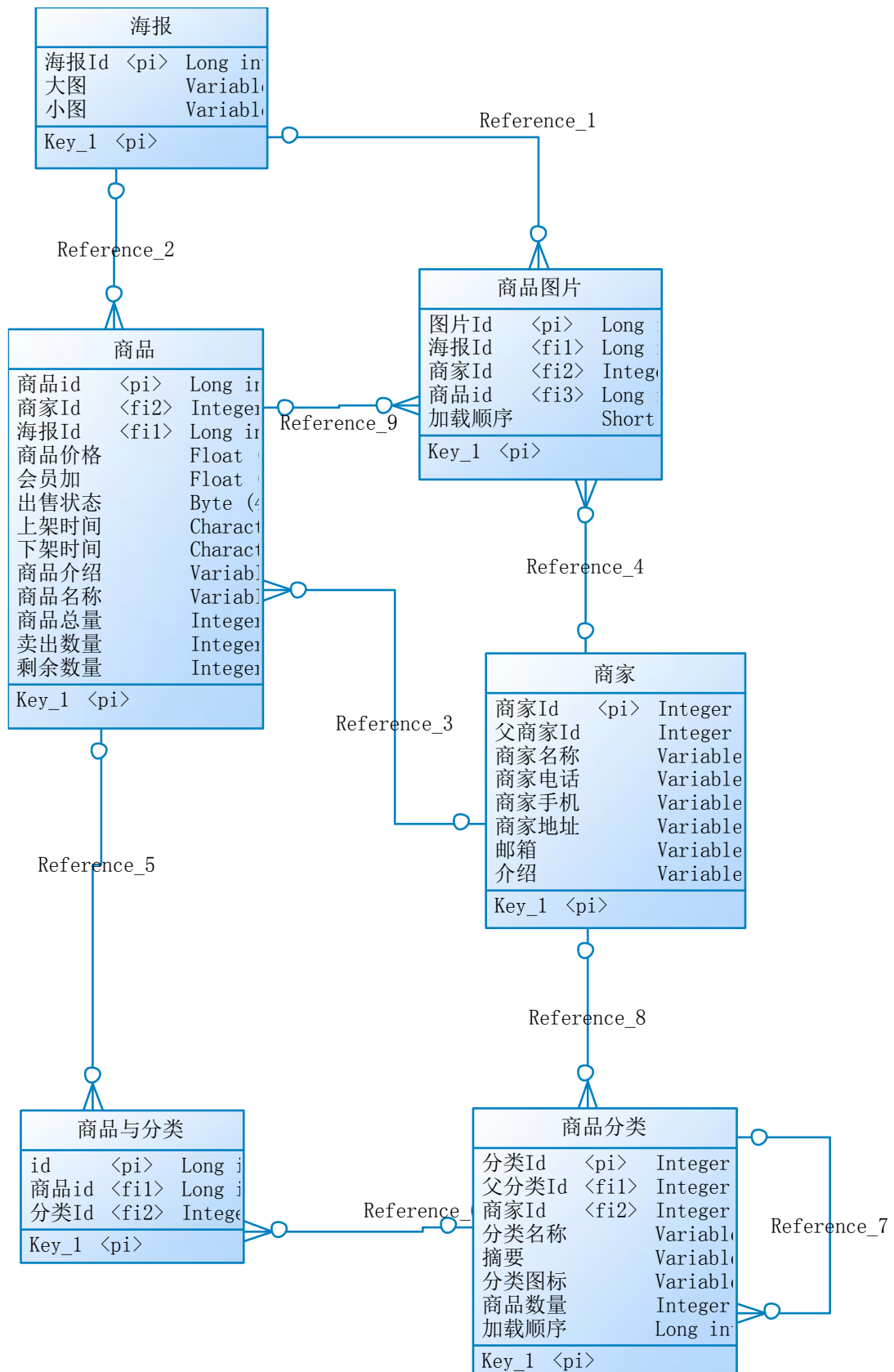


图 3-4 商品模块逻辑图

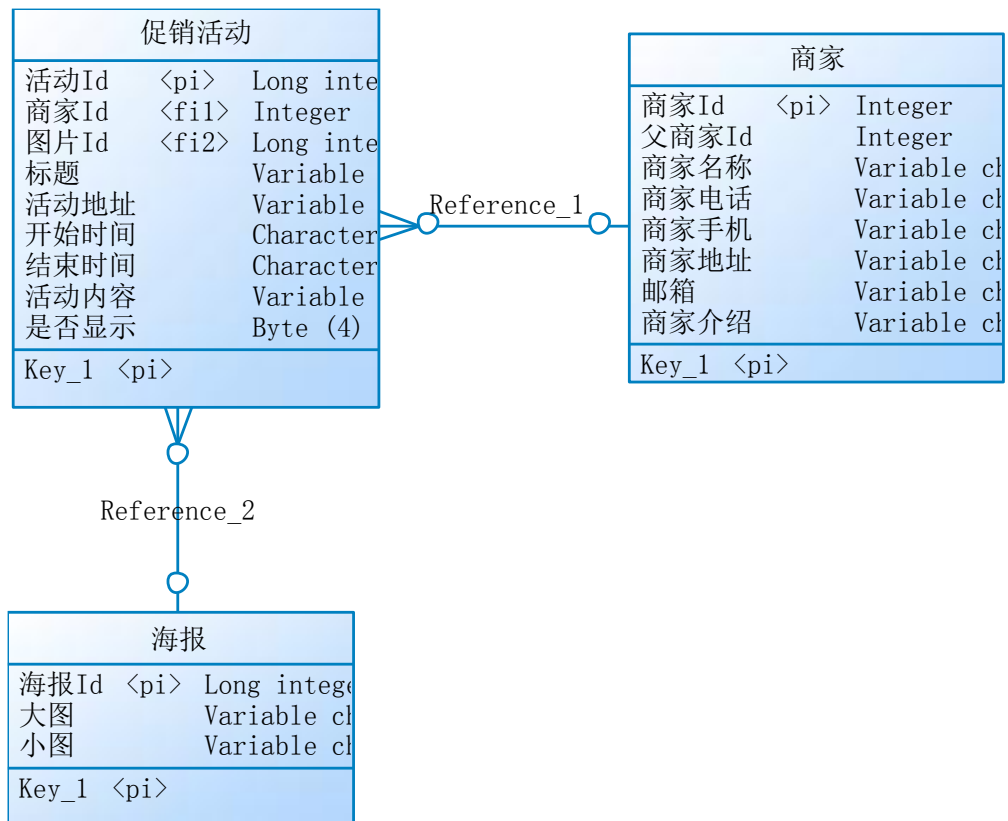


图 3-5 促销模块逻辑图

## 5. 消息反馈模块

消息反馈模块包括用户、消息和商家，对应的逻辑图如图 3-6 所示：

## 6. 消息推送模块

消息推送模块包括商家、推送消息以及发送该消息的用户，为了方便进行管理又增加了消息和商家之间的关系的表。相应的逻辑图如图 3-7 所示：

## 7. 海报模块

海报模块有海报、商家，为方便的进行查询管理增加了海报和商家之间的关系表。该模块的逻辑图如图 3-8 所示：

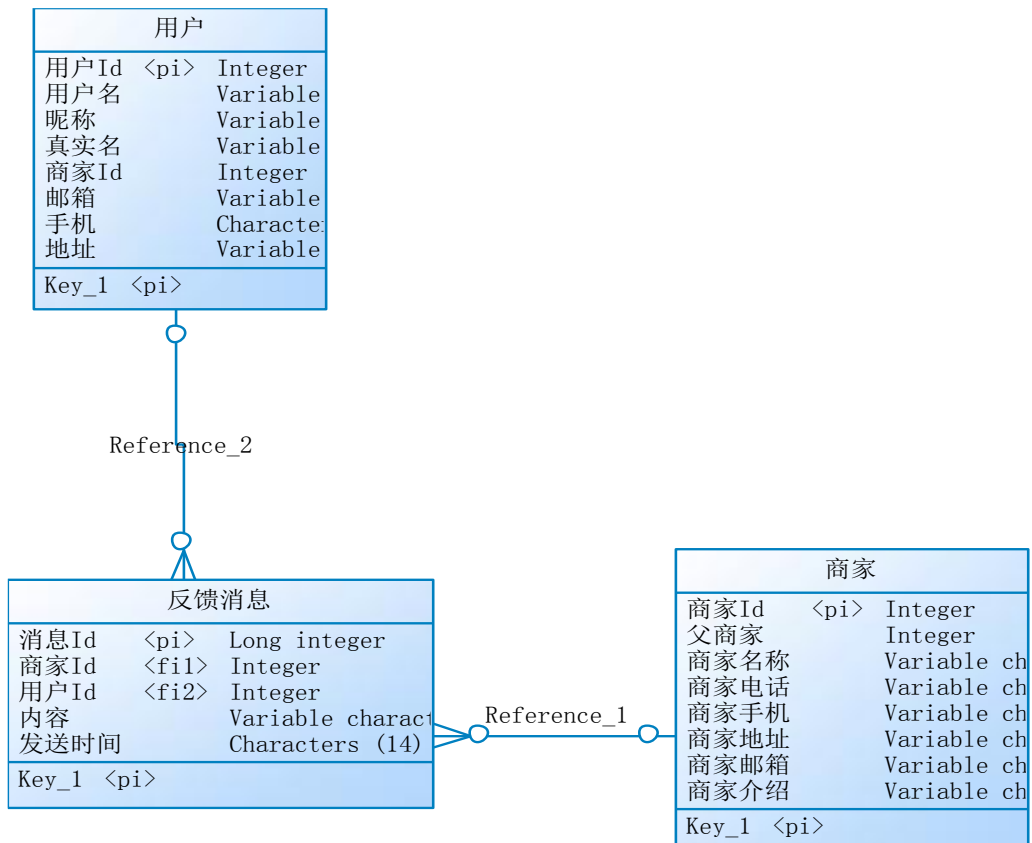


图 3-6 消息反馈模块逻辑图

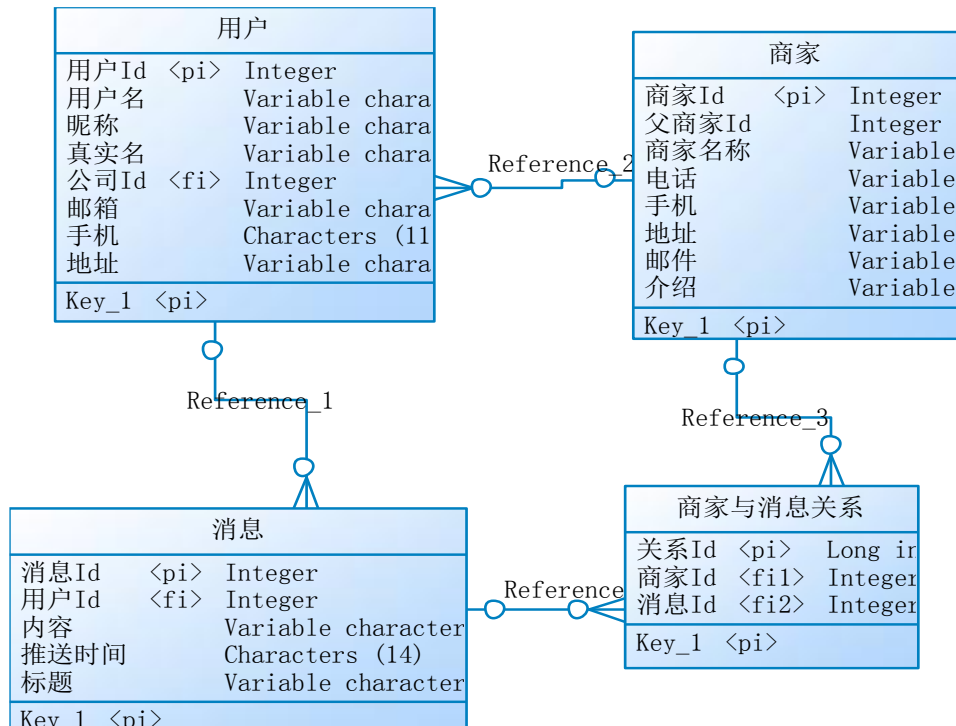


图 3-7 消息推送模块逻辑图



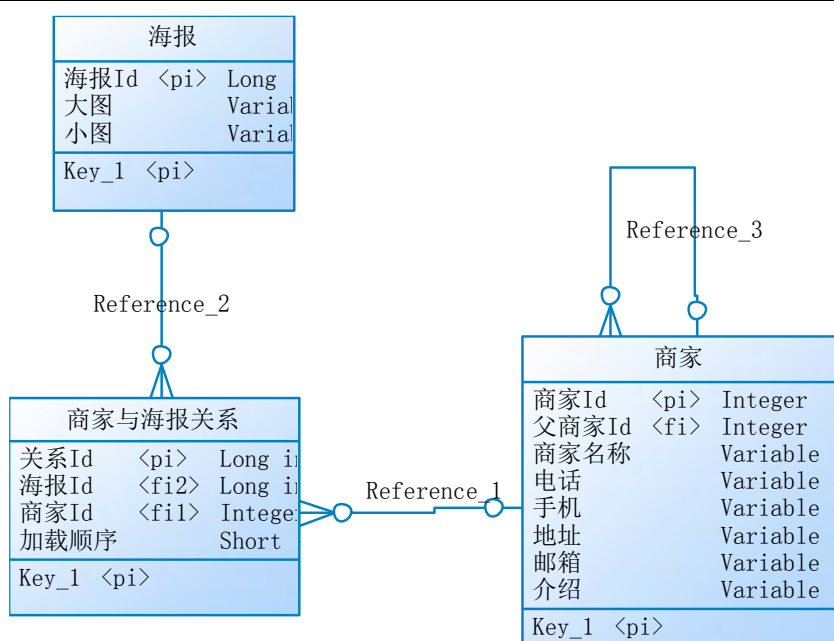


图 3-8 海报模块逻辑图

以上各个模块之间通过公共的实体，形成一个整体，构成了整个数据库。

### 3.2.3 数据库物理设计

根据以上各个模块，结合各个模块之间的关系，构建出了整个数据库，下面列出了数据库中每张表的字段、描述、字段类型、长度、主键、外键等信息。

#### 1. 用户信息表

用户信息表 **member** 包括用户 Id、用户名、昵称、所属商家、邮箱、电话等信息，完整的物理设计如 3-1 表所示。

表 3-1 用户信息 **member** 表

代码	描述	类型	长度	允许空	主键	外键
userId	用户 id	int	11	N	Y	N
userName	用户名	varchar	50	N	N	N
nickName	昵称	varchar	50	Y	N	N
realName	真实名	varchar	50	Y	N	N
companyId	商家 id	int	11	N	N	Y

续表：

email	用户邮箱	varchar	100	Y	N	N
mobile	用户手机	char	11	Y	N	N
address	用户地址	varchar	200	Y	N	N

## 2. 商家信息表

商家信息表 **company** 包括，商家的基本信息情况，包括商家 id、父商家 id、商家名称、电话、手机、地址、邮箱、商家介绍，其中的父商家，表示本商家属于另外一个商家，例如，A 商家（专柜）在绵阳茂业百货之中，那么绵阳茂业百货就是 A 商家的父商家，物理设计如 3-2 表所示。

表 3-2 商家信息 company 表

代码	描述	类型	长度	允许空	主键	外键
companyId	商家 id	int	11	N	Y	N
parentId	父商家 id	int	11	Y	N	Y
companyName	商家名称	varchar	100	N	N	N
tel	商家电话	varchar	13	Y	N	N
mobile	商家手机	char	11	Y	N	N
address	商家地址	varchar	100	Y	N	N
email	商家邮箱	varchar	100	Y	N	N
introduction	商家介绍	varchar	2000	Y	N	N

## 3. 权限表

用户例如管理员有权限对系统进行管理，所以需要权限表 **authority**，权限表的设计只有权限一个字段，具体的物理设计如下表所示。

表 3-3 用户权限 authority 表

代码	描述	类型	长度	允许空	主键	外键
authority	权限	varchar	50	N	Y	N

#### 4. 资源表

资源表 resource 保存着管理员能够访问的资源的地址信息，物理设计如 3-4 表所示。

表 3-4 资源 resource 表

代码	描述	类型	长度	允许空	主键	外键
resourceId	资源 id	int	11	N	Y	N
url	资源位置	varchar	200	N	N	N

#### 5. 资源权限表

资源权限表 resourceauthority 记录了使用该资源需要拥有的权限，物理设计如 3-5 表所示：

表 3-5 资源权限 resourceauthority 表

代码	描述	类型	长度	允许空	主键	外键
id	资源权限关系 id	int	11	N	Y	N
authority	权限	varchar	50	N	N	Y
resourceId	资源 id	int	11	N	N	Y

#### 6. 菜单表

菜单表 menus 记录了整个系统中对数据进行管理的选项的信息，管理员通过这些菜单，对数据库中的数据进行管理，物理设计如 3-6 表所示：

表 3-6 菜单 menus 表

代码	描述	类型	长度	允许空	主键	外键
menuId	菜单 id	int	11	N	Y	N
parentId	父菜单 id	int	11	Y	N	Y
title	菜单名称	varchar	30	N	N	N
url	菜单地址	varchar	200	N	N	N

## 7. 组表

组表 groups 记录了对用户进行分组的所有组的信息,组表的物理设计如 3-7 所示:

表 3-7 组 groups 表

代码	描述	类型	长度	允许空	主键	外键
groupId	分组 id	int	11	N	Y	N
groupName	分组名称	varchar	50	N	N	N
remarks	摘要	varchar	100	Y	N	N

## 8. 组菜单表

组菜单表 groupmenu 记录着每个组可以使用的菜单的信息,物理设计如 3-8 表所示:

表 3-8 组菜单 groupmenu 表

代码	描述	类型	长度	允许空	主键	外键
id	组菜单 id	int	11	N	Y	N
menuId	菜单 id	int	11	N	N	Y
groupId	组 id	int	11	N	N	Y

## 9. 组权限表

组权限 groupauthority 表记录了组与其的权限之间的关系,物理设计如下所示:

表 3-9 组权限 groupauthority 表

代码	描述	类型	长度	允许空	主键	外键
id	组权限 id	int	11	N	Y	N
groupId	组 id	int	11	N	Y	Y
authority	权限	varchar	50	N	Y	Y

## 10. 用户组表

用户组表 groupmembers 记录了用户属于某个组的信息，物理设计如下所示：

表 3-10 用户组 groupmembers 表

代码	描述	类型	长度	允许空	主键	外键
id	用户组 id	int	11	N	Y	N
groupId	组 id	int	11	Y	N	Y
userId	用户 id	int	11	Y	N	Y

## 11. 商品表

商品表 commodity 记录了商品的信息，主要包括商品 id、所属商家、海报、介绍、数量等一系列信息，商品表的物理设计如表 3-11 所示：

表 3-11 商品 commodity 表

代码	描述	类型	长度	允许空	主键	外键
commodityId	商品 id	bigint	20	N	Y	N
companyId	商家 id	int	11	N	N	Y
assetId	海报 id	bigint	20	Y	N	Y
price	商品价格	float	15	N	N	N
memberPrice	会员价	float	15	Y	N	N

续表:

saleState	出售状态	tinyint	4	N	N	N
onShelfTime	上架时间	char	14	Y	N	N
downShelfTime	下架时间	char	14	Y	N	N
introduction	商品介绍	varchar	1500	Y	N	N
commodityName	商品名称	varchar	200	N	N	N
total	商品总量	int	11	Y	N	N
saleNum	卖出数量	int	11	Y	N	N
buyNum	剩余数量	int	11	Y	N	N

## 12. 商品类别表

商品类别 category 表记录了商品的所有分类的信息, 其中的父分类表示的是该分类的上一级分类, 例如裤子是衣服的子分类, 该表的物理设计如表 3-12 所示:

表 3-12 商品类别 category 表

代码	描述	类型	长度	允许空	主键	外键
categoryId	分类 id	int	11	N	Y	N
pCategoryId	父分类 id	int	11	Y	N	Y
companyId	商家 id	int	11	N	N	Y
categoryName	分类名称	varchar	50	N	N	N
remarks	摘要	varchar	200	Y	N	N
pic	分类图标	varchar	200	Y	N	N
commodityCount	商品数量	int	11	Y	N	N
sortOrder	加载顺序	bigint	20	Y	N	N

## 13. 商品与分类表

商品与分类 commoditycategoryrelation 表记录了每件商品与商品类别的对应的关系，商品分类表的物理设计如图表 3-13 所示：

表 3-13 商品与分类 commoditycategoryrelation 表

代码	描述	类型	长度	允许空	主键	外键
id	商品分类 id	bigint	20	N	Y	N
commodityId	商品 id	bigint	20	Y	N	Y
categoryId	商品类别 id	int	11	Y	N	Y

## 14. 商品图片表

商品图片 commodityPic 表主要记录了每件商品所对应的图片的信息，其中的外键 AssetId 对应的是表 asset 中的数据，asset 表中记录了系统中所有图片的路径，商品图片表的物理设计如下表 3-14 所示：

表 3-14 商品图片 commodityPic 表

代码	描述	类型	长度	允许空	主键	外键
id	商品图片 id	bigint	20	N	Y	N
assetId	海报 id	bigint	20	N	N	Y
companyId	商家 id	int	11	Y	N	Y
commodityId	商品 id	bigint	20	N	N	Y
loadIndex	加载顺序	smallint	6	Y	N	N

## 15. 海报表

海报表 asset 中记录了商品海报、商家海报等一切的图片的地址，供其他表引用，海报表的物理设计如下表 3-15 所示：

表 3-15 海报 asset 表

代码	描述	类型	长度	允许空	主键	外键
assetId	海报 id	bigint	20	N	Y	N
thumbPic	大图路径	varchar	500	Y	N	N
smallPic	小图路径	varchar	500	Y	N	N

## 16. 促销活动表

促销活动表 promotionactivity 记录了各个商家促销活动的信息，其中的是否显示字段表示该活动是否在客户端中可见，具体的物理设计如下表 3-16 所示：

表 3-16 促销活动 promotionactivity 表

代码	描述	类型	长度	允许空	主键	外键
id	活动 id	bigint	20	N	Y	N
companyId	商家 id	int	11	N	N	Y
assetId	图片 id	bigint	20	Y	N	Y
title	标题	varchar	50	N	N	N
address	活动地址	varchar	200	N	N	N
starTime	开始时间	char	14	N	N	N
endTime	结束时间	char	14	N	N	N
content	活动内容	varchar	2000	N	N	N
isShow	是否显示	tinyint	4	N	N	N

## 17. 推送消息表

推送消息表 message 记录了管理员向用户推送的消息的信息，推送消息表的物理设计如下表 3-17 所示：



表 3-17 推送消息 message 表

代码	描述	类型	长度	允许空	主键	外键
messageId	消息 id	int	11	N	Y	N
userId	用户 id	int	11	N	N	Y
content	消息内容	varchar	2000	N	N	N
pubTime	发送时间	char	14	N	N	N
title	标题	varchar	200	N	N	N

## 18. 商家推送消息表

商家推送消息表 companymessage 中记录了商家与其推送的消息的对应关系，建立此表可以快速的对商家推送消息进行查询，大大的提高查询的效率，商家推送消息表的物理设计如下表 3-18 所示：

表 3-18 商家推送消息 companymessage 表

代码	描述	类型	长度	允许空	主键	外键
id	商家推送消息 id	bigint	20	N	Y	N
companyId	商家 id	int	11	N	N	Y
messageId	消息 id	int	11	N	N	Y

## 19. 用户反馈消息表

用户反馈消息表 suggmsg 中记录了客户端用户对绵阳茂业百货的反馈信息，用户反馈消息表的物理设计如下表 3-19 所示：

表 3-19 用户反馈信息 suggmsg 表

代码	描述	类型	长度	允许空	主键	外键
id	反馈消息 id	bigint	20	N	Y	N
companyId	商家 id	int	11	N	N	Y

续表：

userId	用户 id	int	11	N	N	Y
content	反馈内容	varchar	2000	N	N	N
sendTime	反馈时间	char	14	N	N	N

## 20. 商家与海报表

商家与海报表 companyposter 中记录了商家与其海报的对应关系，物理设计如下表 3-20 所示：

表 3-20 商家与海报 companyposter 表

代码	描述	类型	长度	允许空	主键	外键
id	商家与海报 id	bigint	20	N	Y	N
assetId	海报 id	bigint	20	N	N	Y
companyId	商家 id	int	11	N	N	Y
loadIndex	加载顺序	smallint	6	Y	N	N

## 第 4 章 茂业百货无线应用服务器系统详细设计与实现

### 4.1 Struts2+Spring3+JPA 架构搭建

#### 4.1.1 Struts2 的基本配置

为系统增加 Struts2 支持需要在应用的 web.xml 配置文件中配置 Struts2 的核心 Filter, 并将 Struts2 的 JAR 包添加到 web 工程的 WEB-INF/lib 下, 下面是配置在 web.xml 中的 Struts2 核心 Filter 配置片段。

```
<!-- Struts2 配置 -->
<filter>
    <filter-name>struts2</filter-name>
    <filter-class>org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter</filter-class>
</filter>
<filter-mapping>
    <filter-name>struts2</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
<filter-mapping>
    <filter-name>struts2</filter-name>
    <url-pattern>*.jsp</url-pattern>
</filter-mapping>
```

#### 4.1.2 Spring3 的基本配置

为系统增加 Spring3 的支持需要将 Spring3 的相关包复制到工程的 WEB-INF/lib 中, 并将 Spring 的监听器配置进 web.xml 中, 使系统启动的时候就自动的创建 Spring 容器。配置代码如下:

```
<!-- Spring 配置 -->
<listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
</listener>
```

在配置了 Spring 之后, 使用 Spring 的依赖注入特性, 整合 Spring 与 Struts2, 使 Spring 托管 Struts2 中的 Action, 通过 struts.objectFactory.spring.autoWire 设置为 type, 使用 Spring 的自动装配策略, 为 Action 注入所需要的业务逻辑组件, 配置文件部分代码示例如下所示:

```
<!-- service -->
```

```

<bean id="AreaImpl" class="service.impl.AreaImpl"/>
<bean id="AssetImpl" class="service.impl.AssetImpl"/>
<bean id="AuthorityImpl" class="service.impl.AuthorityImpl" />
<bean id="CategoryImpl" class="service.impl.CategoryImpl"/>
<bean id="CommodityImpl" class="service.impl.CommodityImpl"/>
<bean id="CompanyBlogImpl" class="service.impl.CompanyBlogImpl"/>
<bean id="CompanyCategoryImpl" class="service.impl.CompanyCategoryImpl"/>
<bean id="CompanyCommentImpl" class="service.impl.CompanyCommentImpl"/>

```

#### 4.1.3 JPA 的基本配置

使用 JPA，需要先将 JPA 规范的实现的 JAR 包，放到 GlassFish 安装目录的 lib 目录下，然后在 web.xml 中配置服务器的数据库连接信息，以便在系统启动的时候获取到数据库连接信息，相应的配置代码如下：

```

<persistence-unit-ref>
  <description>
    Persistence unit for the mall application.
  </description>
<persistence-unit-ref-name>persistence/mallPU</persistence-unit-ref-name>
  <persistence-unit-name>mallPU</persistence-unit-name>
</persistence-unit-ref>
<resource-ref>
  <description>DB Connection</description>
  <res-ref-name>jdbc/mallDataSource</res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
  <res-auth>Container</res-auth>
</resource-ref>

```

之后需要的 persistence.xml 中配置持久化单元，持久化单元使用 GlassFish 中配置的 DataSource，来获取数据库连接信息，配置代码如下所示：

```

<persistence-unit name="mallPU" transaction-type="JTA">
  <provider>
    org.eclipse.persistence.jpa.PersistenceProvider
  </provider>
  <jta-data-source>jdbc/mallDataSource</jta-data-source>
  <!--<mapping-file>META-INF/orm.xml</mapping-file-->
  <class>dao.pojo.TDiscountcoupon</class>
  <class>dao.pojo.TSinglediscountcoupon</class>
  <properties>
    <property name="eclipselink.cache.shared.default"

```

```

        value="true" />
        <property name="eclipselink.target-server" value="SunAS9" />
        <property name="eclipselink.logging.level" value="info" />
        <property name="eclipselink.logging.logger"
            value="ServerLogger" />
    </properties>
</persistence-unit>

```

Spring 可通过容器管理 JPA 的 EntityManagerFactory, 使用 Spring 容器管理 JPA 可以避免在程序中手工的创建 EntityManagerFactory, 并且可以方便的将 EntityManagerFactory 注入到 DAO 组件之中, 以下是使用 LocalContainerEntityManagerFactoryBean 对 EntityManagerFactory 进行控制, 其中注入了持久化单元, 数据源采用 JNDI 的方式获取:

```

<bean id="dataSource"
    class="org.springframework.jndi.JndiObjectFactoryBean">
    <property name="jndiName">
        <value>java:comp/env/jdbc/mallDataSource</value>
    </property>
</bean>
<bean id="entityManagerFactory"
    class="org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean">
    <property name="persistenceUnitName" value="mallPU" />
</bean>

```

Spring 整合 JPA 之后, 所有的持久化操作都应该在事务的环境下进行, 否则对数据库的多个操作都不会提交, 因此需要增加声明式事务<sup>[13]</sup>, 以下是配置事务管理器的代码, 其中使用了 JtaTransactionManager 作为事务管理器的实现类:

```

<bean id="transactionManager"
    class="org.springframework.transaction.jta.JtaTransactionManager">
</bean>

```

最后需要在代码中的 EntityManager 类型的字段上添加 @PersistenceContext 注解, 即持久化上下文注解, 持久化上下文中的 Entity 实体, 可能被 EntityManager 处理, 通过将 @PersistenceContext 注解标注在 EntityManager 字段上, 可以得到由 Spring 容器管理的 EntityManager, 这样由容器管理, 就不需要关注 EntityManager 实例注入后何时进行关闭。

#### 4.1.4 Spring Security 的基本配置

首先在 web.xml 配置拦截器即声明 DelegatingFilterProxy，配置的代码如下所示：

```
<filter>
  <filter-name>springSecurityFilterChain</filter-name>
  <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
</filter>
<filter-mapping>
  <filter-name>springSecurityFilterChain</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

以上配置表示对项目中的所有资源的请求都要经过 Spring Security，之后在 applicationContext.xml 中配置 Spring Security 中重写的拦截器和拦截器的位置，以及退出系统时的跳转链接，配置代码如下所示：

```
<http entry-point-ref="authenticationEntryPoint">
  <session-management>
    <concurrency-control max-sessions="1" />
  </session-management>
  <custom-filter position="FORM_LOGIN_FILTER" ref="loginFilter" />
  <custom-filter after="EXCEPTION_TRANSLATION_FILTER"
ref="exceptionTranslationFilter" />
  <custom-filter before="FILTER_SECURITY_INTERCEPTOR" ref="securityFilter" />
  <logout logout-success-url="/jsp/web/login.jsp"/>
</http>
```

##### 1. authenticationEntryPoint

其中 authenticationEntryPoint 为认证用户身份的切入点，切入点的配置代码如下所示：

```
<b:bean id="authenticationEntryPoint"
  class="org.springframework.security.web.authentication.LoginUrlAuthenticationEntryPoint">
  <b:property name="loginFormUrl" value="/jsp/web/login.jsp"/>
</b:bean>
```

##### 2. loginFilter

重写的用户登录过滤器的配置如下所示：

```
<b:bean id="loginFilter" class="security.UserPwdCodeAuthenticationFilter">
  <b:property name="authenticationManager" ref="authenticationManager" />
```

```

    <b:property name="filterProcessesUrl" value="/j_spring_security_check" />
    <b:property name="authenticationFailureHandler">
    <b:bean
class="org.springframework.security.web.authentication.SimpleUrlAuthenticationFailureHandler">
        <b:property name="defaultFailureUrl" value="/jsp/common/loginfail.jsp" />
        </b:bean>
    </b:property>
    <b:property name="authenticationSuccessHandler">
    <b:bean
class="org.springframework.security.web.authentication.SimpleUrlAuthenticationSuccessHandler">
        <b:property name="defaultTargetUrl" value="/login/decision.action"/>
        </b:bean>
    </b:property>
</b:bean>

```

用户登录过滤器中的 `authenticationManager` 为认证管理器，`authenticationFailureHandler` 为认证失败后的处理器，页面跳转到 `/jsp/common/loginfail.jsp`，`authenticationSuccessHandler` 为认证成功的处理器，请求 `/login/decision.action`，其中的 `authenticationManager` 配置如下：

```

<authentication-manager alias="authenticationManager">
    <authentication-provider user-service-ref="userService">
        <password-encoder hash="md5" />
    </authentication-provider>
</authentication-manager>
<b:bean id="userService" class="security.MyUserDetailsService" autowire="byType" />

```

### 3. exceptionTranslationFilter

`exceptionTranslationFilter` 为异常事务管理器，当事务运行发生异常时，进行处理，配置如下：

```

<b:bean id="exceptionTranslationFilter"
class="org.springframework.security.web.access.ExceptionTranslationFilter">
    <b:property name="authenticationEntryPoint" ref="authenticationEntryPoint"/>
    <b:property name="accessDeniedHandler" ref="accessDeniedHandler"/>
</b:bean>
<b:bean id="authenticationEntryPoint"
class="org.springframework.security.web.authentication.LoginUrlAuthenticationEntryPoint">
    <b:property name="loginFormUrl" value="/jsp/web/login.jsp"/>
</b:bean>

```

```

<b:bean id="accessDeniedHandler"
      class="org.springframework.security.web.access.AccessDeniedHandlerImpl">
    <b:property name="errorPage" value="/jsp/common/noright.html"/>
</b:bean>

```

其中 authenticationEntryPoint 为之前配置的认证切入点，accessDeniedHandler 为认证失败的处理器。

#### 4. securityFilter

securityFilter 为权限拦截器，用来判断用户对其访问的资源是否有权限进行访问，并返回判断结果，配置代码如下所示：

```

<b:bean id="securityFilter" class="security.MySecurityInterceptor">
    <b:property name="authenticationManager" ref="authenticationManager" />
    <b:property name="accessDecisionManager" ref="accessDecisionManager" />
    <b:property name="securityMetadataSource" ref="securityMetadataSource" />
</b:bean>
<b:bean id="accessDecisionManager"
      class="org.springframework.security.access.vote.AffirmativeBased">
    <b:property name="decisionVoters">
        <b:ref local="roleVote" />
    </b:property>
</b:bean>
<b:bean id="roleVote" class="org.springframework.security.access.vote.RoleVoter" />
<b:bean id="securityMetadataSource" class="security.MyInvocationSecurityMetadataSource" >
    <b:constructor-arg type="service.interf.IResource">
        <b:ref bean="ResourceImpl"/>
    </b:constructor-arg>
</b:bean>

```

其中的 authenticationManager 是权限验证的管理器，accessDecisionManager 为验证之后的决策管理器，securityMetadataSource 为认证需要使用的数据源，它们的配置如下所示：

```

<authentication-manager alias="authenticationManager">
    <authentication-provider user-service-ref="userService">
        <password-encoder hash="md5" />
    </authentication-provider>
</authentication-manager>
<b:bean id="userService" class="security.MyUserDetailsService" autowire="byType" />
<b:bean id="accessDecisionManager"

```



```

class="org.springframework.security.access.vote.AffirmativeBased">
  <b:property name="decisionVoters">
    <b:ref local="roleVote" />
  </b:property>
</b:bean>

<b:bean id="roleVote" class="org.springframework.security.access.vote.RoleVoter" />

  <b:bean id="securityMetadataSource"
    class="security.MyInvocationSecurityMetadataSource" >
    <b:constructor-arg type="service.interf.IResource">
      <b:ref bean="ResourceImpl"/>
    </b:constructor-arg>
  </b:bean>

```

以上的配置中用到的数据源和数据库连接信息需要在 GlassFish 应用服务器的 web 管理中配置，启动 GlassFish 服务器，在浏览器的地址栏中输入 <http://localhost:4848>，进入管理窗口，首先点击左边的菜单数据连接池，新建数据连接池，命名为 moPool，选择资源类型为 java.sql.Driver，数据库供应商选择 MySQL 数据库，之后添加连接池需要的属性，其中包括数据库连接时需要的用户名，密码，数据库 URL 链接，保存这些配置，之后在左侧的菜单中选择数据源新建数据源，并将数据源的名字命名为 jdbc/mallDataSource，名字与之上配置文件 persistence.xml 中引用的数据源的名称应该保持一致，最后选择刚刚配置好的数据库连接池 moPool，之后指定数据源名为标识 jdbc 资源的 JNDI（java 命名与目录接口）名称。

## 4.2 用户登录服务器的设计与实现

### 4.2.1 用户登录页面设计

用户需要输入用户名、密码以及验证码进行登录。在登录页面采用提交 form 表单的方式，post 方法进行提交用户数据，form 标签的 action 属性值即提交数据的链接为 Spring securithy 配置时配置的登录链接名称 j\_spring\_security\_check，用户填写表单之后，会触发对填写的表单的进行检验，例如用户没有输入验证码等，验证通过则向服务器端提交请求，否则提醒用户继续提交表单，页面效果如图 4-1 所示：



图 4-1 用户登录

用户登录后，将进入到系统管理页面，该页面显示了系统的管理菜单，用户可根据需要点击管理菜单查看数据，并对数据进行管理，页面效果图如下图 4-2 所示：



图 4-2 管理菜单列表



图 4-3 信息列表

#### 4.2.2 验证码的生成和检验

在用户点击验证码的输入框时，异步的请求服务器，服务器端随机生成四位字符串，将此字符串保存在 session 中，并根据该随机字符串生成验证码图片，最后将生成的图片的路径返回给客户端，客户端通过 img 标签的 src 属性接收该图片的路径，浏览器发出对该图片的请求，获取到该图片，当用户提交填写的表单的时候，会先调用验证码验证函数，通过将验证码发送到服务器，对提交的验证码与 session 中的字符串进行比较，若相同就通过验证，否则验证失败。

#### 4.2.3 控制层的实现

系统对服务器端用户的登录采用了 Spring Security 权限验证的机制，用到了 Spring AOP（面向切面编程）和 servlet 过滤器等相关技术，没使用到 Struts2，因此并没有逻辑意义上的控制层。

#### 4.2.4 业务逻辑层的实现

应用 Spring Security 时在 web.xml 中配置的过滤器 loginFilter 来控制用户登录后的页面跳转，它依次调用 Spring Security 框架中方法，最后委托给了 UserDetailsService 类，我们要做的就是实现 UserDetailsService 类，并且实现其中的 loadUserByUsername 方法，该方法要返回 org.springframework.security.userdetails.User 类型的对象，该对象中保存着用户的用户名，密码以及拥有的权限，系统中实现该类的是自定义的类 MyUserDetailsService 类，该类中实现了具体的访问数据库的操作，并将访问到的数据封装到 userdetail 对象中，该对象是 org.springframework.security.userdetails.User 类型的对象，而决定用户是否具有权限访问特定资源的是配置文件中配置的决策管理器，决策管理器使用的是 org.springframework.security.access.vote.RoleVoter，该类会根据在配置文件中的配置来决定用户是否在当前的权限下能够访问特定的资源。

#### 4.2.5 持久层的实现

业务逻辑层 MyUserDetailsService 类中涉及到的对数据库访问操作的方法封装在持久层的 UserDAO 和 AutoDAO 中，其中 UserDAO 中封装了对用户数据进行查找的方法，AutoDAO 中封装了对权限进行查询的方法，MyUserDetailsService 类中的 loadUserByUsername 方法，正是通过调用这两个类的对象的方法对用户数据进行访问，并将数据进行封装的。

## 4.3 海报管理的设计与实现

### 4.3.1 海报管理页面设计

海报管理的设计有海报的添加、修改删除功能，用户点击海报管理之后的效果如下图 4-4 所示：




海报管理	
添加 修改 删除 上移 下移	
海报	加载次序
1 	1
2 	2
3 	3

图 4-4 海报列表

点击添加按钮之后，可跳转到对海报的添加页面，可对海报进行添加，效果图如下 4-5 所示：

海报管理	
填写海报信息 (带 * 为必填项)	
海报 *	<div>浏览</div>
	<div></div>
加载次序	<div>1</div>
	<div>确定</div>

图 4-5 海报添加

添加图片后的效果图 4-6 如下：



图 4-6 添加海报效果

点击确定按钮，添加成功后将跳转到海报管理页面，显示添加后的列表，并提示用户添加成功：



图 4-7 海报添加成功效果

选择某一海报行，点击修改可对海报信息进行修改，效果图和添加海报后相同。选择某一海报行后，点击删除按钮将弹出对话框，提醒用户是否确认删除，点击确认后删除该行。

### 4.3.2 控制层的实现

#### 1. 海报添加

海报添加向服务器请求/company/companyPoster\_add.action，根据 Struts2 的配置文件，该请求应该调用 CompanyPosterAction 类的对象的 add 方法进行处理，add 方法进行处理时调用了 service 层的函数进行处理，其中当点击添加之后图片的上传请求的是/upload/imageUpload.action，配置文件中设定请求该链接调用的方法是 save 方法，该方法调用 service 层方法对文件进行保存。

## 2. 海报修改

海报修改请求的 action 为/company/companyPoster\_edit.action, 调用该类中的 edit 方法进行海报修改的处理, 当跳转到修改页面的时候, 会首先请求/company/companyPoster\_getbyid.action 根据海报的 id 值取出修改前的数据, 提交的数据中包括要修改的海报的 id, 以及修改后的图片和加载顺序。

## 3. 海报删除

海报删除的请求为/company/companyPoster\_dellogon.action, 调用 dellogon 方法对海报的数据进行删除。

### 4.3.3 业务逻辑层的实现

#### 1. 海报添加

海报添加 action 调用 addAsset 方法, 该方法接收 action 中 add 方法传进的封装了提交信息的 model 对象, 首先查找添加海报的商家 id, 之后创建 asset 海报对象, 将海报信息以及商家添加进去, 接着创建 CompanyPoster 对象, 添加进 asset 对象, 保持他们之间的关系, 最后调用 DAO 层方法, 将对象持久化到数据库。

#### 2. 海报修改

海报修改调用 service 层的 edit 方法, 该方法的实现基本与海报添加方法相同, 不同的是在对 DAO 层调用是进行判断了海报的 id 是否存在, 若不存在海报的 id 则进行保存操作, 若存在海报 id, 则调用 DAO 层的 update 方法, 对 Asset 和 CompanyPoster 进行更改。

#### 3. 海报删除

海报的删除调用了 service 层的 del 方法, 参数为海报的 id、商家 id 以及海报的路径, 对海报进行删除。

### 4.3.4 持久层的实现

#### 1. 海报添加

海报添加 DAO 层继承了 BaseDAO, 该类中封装了一系列的用于增删改查的可重用的方法, 这些方法利用 java 泛型的特性实现了代码的重用, 海报添加中使用了 BaseDAO 的 save 方法。

## 2. 海报修改

海报修改持久层也同样继承了 BaseDAO，该类中的 update 方法会被调用，方法的参数为 service 层创建的与数据库关系表一一对应的 POJO 对象。

## 3. 海报删除

海报删除调用的是 delete 方法，对数据库相关表进行了删除操作，BaseDAO 中的方法首先会调用 openTransactionEntityManager 方法，打开实体事务管理器，然后进行事务的操作，结束后关闭管理器。

## 4.4 专柜分类管理的设计与实现

### 4.4.1 专柜分类管理的页面设计

用户点击专柜分类管理菜单可打开专柜管理的页面，可浏览所有的分类以及他们之间的父子关系，该页面的效果图 4-8 如下：

专柜分类管理					
添加 修改 删除 上移 下移					
	专柜名称	显示序号	父分类	专柜分类说明	
1	1楼璀璨时尚馆	1		1楼璀璨时尚馆	
2	2楼青春少淑馆	2		2楼青春少淑馆	
3	3楼仕女名媛馆	3		3楼仕女名媛馆	
4	4楼绅士名品馆	4		4楼绅士名品馆	
5	5楼儿童家居馆	5		5楼儿童家居馆	
6	珠宝	6	1楼璀璨时尚馆	珠宝	
7	名品	7	1楼璀璨时尚馆	名品	

图 4-8 专柜分类列表

专柜的分类可进行添加、修改、删除操作，点击添加按钮后的效果图如图 4-9 所示：

选中分类的某一行，点击修改按钮后的效果图和添加的效果图类似，不同之处在于表单中有数据可供更改，点击删除的效果图如图 4-10 所示，点击确认后删除该行，点击取消则取消操作。

专柜分类管理

填写商家分类信息(带 \* 为必填项)

父分类

清除

专柜分类名称\*

排列序号 \*

1

专柜分类说明

确定

图 4-9 专柜分类添加



图 4-10 删除专柜分类

#### 4.4.2 控制层的实现

##### 1. 专柜分类添加

控制层的 Struts2 的 action 实现的时候采用的是模型驱动的方式, 即 action 类实现是继承自 ModelDriven 类的, 这样 Struts2 在拦截到请求信息之后, 就会将请求中的数据信息自动的封装到 model 中, 专柜分类的 model 如下表 4-1 所示:

表 4-1 专柜分类 model

字段名称 (与请求信息中的参数名称一致)	说明
private Integer companyCategoryId;	新建分类的父分类 id 值
private String categoryName;	分类名称
private Integer sortOrder;	排序号



续表:

private String description;	分类描述
-----------------------------	------

分类添加的 action 链接为/company/category\_add.action, 根据 Struts2 的配置文件 CategoryAction 类中的 add 方法会被调用, 在处理时, 直接将 Struts2 创建的封装有请求参数的 model 对象传递给业务逻辑层进行处理, 处理后将返回的数据接收到, 并发送给浏览器。其中在数据返回时, 会将数据封装为 JSON 数据格式。

## 2. 专柜分类修改

专柜分类的修改基本上和专柜分类的添加的基本操作流程是一样的, 只是在进行修改之前需要将要修改的分类信息给查询出来, 查询的 action 链接为/company/category\_getById.action, 该链接对应的处理方法是 CategoryAction 类中的 getById 方法, 参数为要查询的分类的 id, 该方法调用业务逻辑层的方法查询分类的信息, 在查出来之后, 由于浏览器端采用的是异步请求的方式, 请求处理返回的是 JSON 格式的数据, 因此在 Struts2 配置文件中不需要设置请求成功后的跳转页面, 返回的数据直接被原来发出请求的页面所接收, 在 action 中采用的是直接向 response 输出流的方式响应请求, 即直接将响应的内容作为输出流的内容, 输入到发出请求端, 由发出请求端接收, 对数据进行处理。在把响应的数据写进输出流之前, 还设置了 http 响应的头参数, 它们是 ContentType: text/html;charset=utf-8 和 Cache-Control: no-cache, 用来控制浏览器对响应内容解析时采用正确的编码和保持最新的响应信息。

## 3. 专柜分类删除

删除专柜分类的链接为/company/category\_del.action, 处理请求的是 CategoryAction 类中的 del 方法, 该方法接收的是要删除的分类的 id, 通过调用 service 层的方法进行处理。

### 4.4.3 业务逻辑层的实现

#### 1. 专柜分类添加

Service 层接收到控制层传递进来的 model 对象后, 首先创建与专柜分类对应的对象, 并将 model 对象中封装的数据取出设置到专柜分类对象之中, 在把新添加的分类设置父分类时, 需要先创建新的分类对象, 并通过调用 DAO 层的方法传递父分类

id 值获取到父分类对象，对父分类进行赋值，之后调用 DAO 层的方法，将添加的分类信息添加到数据库中。

## 2. 专柜分类修改

专柜分类的修改的流程与专柜添加的差别不大，在业务逻辑层进行处理时，首先，应该查找修改后的父分类是否存在，若不存在该父分类，则抛出异常，在进行修改时，显示修改前的信息的 service 的方法，首先以要修改的专柜分类的 id 作为参数调用 DAO 层的方法，查询到封装了该分类的 POJO 对象，由于采用的是 JSON 数据进行前后台的交互，所以在在查询出该对象后需要把其封装的信息提取出来重新组合为 JSONObject，并调用 JSONObject 的 toString 方法将该对象的字符串返回到控制层。

## 3. 专柜分类删除

专柜的删除调用的是 DAO 层的方法，传递的是要删除的分类的 id。

### 4.4.4 持久层的实现

#### 1. 专柜分类添加

持久层类继承自 BaseDAO，调用的是其中的 save 方法，save 方法使用 JPA 规范的 EntityManager 类的 persist 方法，将分类对象持久化到数据库中。

#### 2. 专柜分类修改

专柜分类的修改调用 JPA 的 EntityManager 类的 merge 方法对分类进行修改。

#### 3. 专柜分类删除

专柜删除时，传递的参数是要删除分类的 id，在 DAO 层时 deleteById 首先会调用 findById 查询到要删除的对象，然后调用 delete 方法使用 JPA 中 EntityManager 类的 remove 方法，将该分类从数据库中删除，删除分类时，在对数据库表与系统中的 POJO 类映射配置时，配置的为级联删除。

## 4.5 专柜管理的设计与实现

### 4.5.1 专柜管理的页面设计

专柜管理同样也是包括添加、修改和删除操作，当点击专柜管理菜单时，显示当前的专柜信息列表，如下图 4-11 所示：

专柜管理

添加

修改

删除

	商家名称	商家图标	商家Logo	实体图片	广告图片	所属分类	显示序号	固定电话	移动电话	地址
1	恩曼琳					女装	1	0816-2789123	15881612345	四川省绵阳市
2	洛诗琳					女装	2	0816-2789123	15881412345	绵阳茂业百货

图 4-11 专柜信息列表

点击上方的添加按钮会跳转到添加的页面，添加的页面，效果如 4-12 下所示：

专柜管理

填写专柜信息 (带 \* 为必填项)

专柜名称 \*

洛诗琳

专柜图标 \*

浏览



专柜Logo

浏览



实体图片 \*

浏览



广告图片

浏览

图 4-12 添加专柜

4.5.2 控制层的实现

1. 专柜添加

专柜添加的链接为/company/companyE\_addSon.action，调用 CompanyEditAction 类的 addSon 方法来处理请求，类似之前的专柜分类的添加，action 中用到了封装了请求参数的 model 对象，专柜的 model 对象主要包括的信息如表 4-2 所示：

表 4-2 专柜 model

字段名称（与请求信息中的参数一致）	说明
private String companyName	专柜名称
private String Icon	专柜图标
private String logoPic	专柜 LOGO
private String FacePic	专柜实体图片
private String AdPic	广告图片
private Integer categoryId	所属分类
private Integer sortOrder	显示顺序
private String tel	固定电话
private String mobile	移动电话
private String address	地址
private String introduction	介绍

其中涉及到了图片的上传，图片上传使用的是 Uploadify，它是 jQuery 的一个上传插件，该插件可以实现批量上传文件，并且可以显示上传的进度，通过使用该插件，把属性 auto 设置为 true，uploader 设置为 toContextPath('/upload/imageUpload.action')，即可在选中文件后自动的向链接/upload/imageUpload.action 提交图片，根据 Struts2 的配置，该请求会调用 ImageAction 类中的 save 方法进行处理，save 方法调用业务逻辑层的 saveUploadFile 方法将上传的文件保存在服务器中的文件夹中，处理之后将会把保存图片的路径返回给浏览器端，前端接收到此图片的路径后，就将此路径作为隐藏表单的 value 值，这样在用户点添加按钮的时候，上传的只是保存在服务器端的文件的路径，图片在点击添加按钮之前其实已经被提交到服务器中了，这样做可以方便的对用户提交的图片进行及时处理。

## 2. 专柜修改

专柜修改的 action 链接为/company/companyE\_editSon.action，同样在修改之前会

把将要修改的专柜的信息查询出来，查询专柜信息的链接为 `/company/companyF_getById.action`，修改后处理的流程与专柜的添加十分类似，专柜修改的总体处理流程与之前介绍的专柜分类修改的处理流程一致。

### 3. 专柜删除

专柜删除的链接为 `/company/companyE_delBatchSon.action`，处理删除请求的方法是 `CompanyEditAction` 类中的 `delBatchSon`，该方法实现的是批量删除专柜，在前端实现的时候，首先会选出被选中的专柜的集合，之后对集合进行遍历，遍历时将每一个专柜的 `id` 取出，将它们组合为以逗号 “,” 相隔的字符串，并将该字符串作为参数传递给该 `action`，`delBatchSon` 方法将该参数的值从 `model` 中取出，作为参数调用 `service` 层的方法进行处理，最后将 `service` 层处理后返回的数据作为处理结果返回给浏览器端。

## 4.5.3 业务逻辑层的实现

### 1. 专柜添加

控制层调用的是业务逻辑层的 `addCompany` 方法进行专柜的添加，该方法接收封装了前台提交的数据的 `model` 对象作为参数，在该方法中创建专柜的实例，并将 `model` 中封装的数据取出，添加到专柜的实例之中，最后调用 `DAO` 层的方法持久化该实例到数据库。在对提交的图片进行保存的时候，实际上保存的是提交的图片在服务器端的路径，从之上控制层的实现中可知，在点击添加按钮之前，其实图片已经上传到服务器端，点击添加按钮之后，提交的其实只是保存图片的服务器端的路径，在对图片进行保存时，控制层调用的是 `FileHandleImpl` 类中的 `saveUploadFile` 方法把图片文件保存到服务器端的，在该方法中首先会创建一个唯一的保存图片的文件夹，之后采用工具类 `UUID`（采用一定的算法，结合当前机器的网卡、时间、随机数等生成不重复的十六进制的数字）的 `randomUUID` 方法生成唯一的字符串作为文件的文件名，之后创建文件的输入流和输出流将图片文件保存到设定到的目录之中，成功之后将文件名返回，在控制层将文件的保存路径和文件名称组合为完整的图片访问路径，返回到浏览器端，浏览器之后提交的数据就是该文件路径。

### 2. 专柜修改

专柜修改中的 `service` 方法是 `editCompany` 方法，调用 `DAO` 层的方法将修改后的

对象持久化到数据库之中。

### 3. 专柜删除

专柜的删除与之前删除专柜分类,海报等基本相同,注意一点的是,批量删除时,在业务逻辑控制层,根据将要删除的专柜 id,查出各个专柜对象,并经其添加到一个集合之中,该集合将作为参数提供给 DAO 层。

#### 4.5.4 持久层的实现

##### 1. 专柜添加

专柜的添加的 DAO 层依然是采用的 BaseDAO 中的公用的方法 save 方法进行保存。

##### 2. 专柜修改

专柜修改的持久层实现也是使用了基本的数据库操作,使用 BaseDAO 中的方法。

##### 3. 专柜删除

专柜删除使用的是批量的删除,在接收到业务逻辑层传递进来的专柜集合之后,遍历集合,将集合中的专柜一一进行删除。

## 4.6 商品管理的设计与实现

### 4.6.1 商品管理的页面设计

商品管理同其他的管理一样,包括商品的添加、修改、删除操作,当点击商品管理菜单时显示的是商品的列表,效果如图 4-13 所示。

当点击添加按钮的时候会将页面跳转到商品添加的页面,页面的效果如图 4-14 所示。

当选中其中的一个商品后点击修改时,会跳转到修改商品信息的页面,页面的效果如图 4-15 所示。

商品管理					
<div>添加 修改 删除</div>					
	<input type="checkbox"/>	商品名称	所属专柜	商品缩略图	商品图片
1	<input type="checkbox"/>	韩版连衣裙	洛诗琳		
2	<input type="checkbox"/>	夏季连衣裙	洛诗琳		
3	<input type="checkbox"/>	连衣裙	恩曼琳		
<div>10 第1页 共1页</div>					

图 4-13 商品信息列表

商品管理

填写商品信息 (带 \* 为必填项)

商品名称 \*

专柜 \*  若无则请先添加专柜!

商品缩略图 \* 

浏览

商品详细图片 \* 

浏览

确定

图 4-14 商品添加

商品管理

填写商品信息 (带 \* 为必填项)

商品名称 \*

专柜 \* 

洛诗琳

 若无则请先添加专柜!

商品缩略图 \* 

浏览

商品详细图片 \* 

浏览

删除

删除

图 4-15 商品添加效果

#### 4.6.2 控制层的实现

##### 1. 商品添加

商品添加的链接为/commodity/commodityE\_add.action，在添加商品的时候同样涉及到商品图片的添加，添加的策略依然是在用户确定好自己要上传的图片后，就触发类型为 file 的 input 的 onchange 事件，事件发生后的处理函数为将图片提交到服务器，在服务器端成功的保存好图片之后，就将图片的路径作为返回值返回到请求端，并回调了客户端的回调函数，回调函数的作用是提示用户图片已经上传成功，并将返回的图片的路径作为 img 标签的 src 属性，这样就实现了用户对所上传图片的预览，最后创建一个隐藏的 input，将返回的那个图片路径作为 value 值设置给该 input，这样在填写完商品的详细信息之后，点击添加按钮就可以将其他信息连同保存图片的路径信息一并的交给 CommodityEditAction 中的 add 方法进行处理，add 方法将封装好的商品的 model 作为参数调用业务逻辑层的 addCommodity 方法，之后接收该方法返回的数据，并将其返回给请求端，其中商品的 model 中的字段信息如下表 4-3 所示：

表 4-3 商品 model

字段名称（与请求中的参数名称一致）	说明
private Long commodityId	商品 id
private Integer companyId	商品所属专柜 id
private String commodityName;	商品名称
private String picUrl;	商品缩略图 url
private String[] assets;	商品详情图片的所有 url

其中的商品详情的图片的 url 为一个字符串的数组，这是由于，商品详情图片可能不只有一张，可能有多张，在浏览器端通过多个 name 属性相同的隐藏的 input 即可发送 http 请求时方法多个同名不同值的参数，在 Struts2 拦截后，会将同名的参数封装为一个 String 数组。

##### 2. 商品修改

商品的修改在修改时需要将要修改的信息查出，查找商品信息的链接为



/commodity/commodityF\_getbyid.action,通过商品的 id 将商品的信息呈现给用户查看,用户修改之后提交数据的流程和商品的添加流程基本相同,不同的是/commodity/commodityE\_edit.action 链接的处理方法为 CommodityEditAction 中的 edit 方法,调用的业务逻辑层的方法为 editCommodity 方法。

### 3. 商品删除

在选中要删除的商品之后,点击删除按钮,会将该商品的 id 作为参数请求 /commodity/commodityE\_delbatch.action,处理方法为 delbatch 方法,该方法调用业务逻辑层的方法,并将返回值作为结果返回给客户端。

## 4.6.3 业务逻辑层的实现

### 1. 商品添加

在业务逻辑层创建商品的对象,将 model 中的数据取出,封装到商品对象中调用 DAO 层的方法,将数据持久化到数据库中,在对商品的图片进行保存时,对数组进行遍历,创建一个 asset 对象之后,调用 DAO 层保存,并将创建商品图片与商品之间的关系对象,调用 DAO 层进行保存。

### 2. 商品修改

先将 model 中的商品的 id 取出,通过 id 调用 DAO 层方法将商品取出,之后根据 model 中的信息,重新修改该商品对象,最后调用 DAO 层方法将商品对象持久化到数据库中。

### 3. 商品删除

商品删除的业务逻辑层,先根据要删除的商品的 id 数组,将要删除的对象查出,添加到集合之中,再对查出的集合进行遍历,在遍历时将商品图片的路径保存到一个字符串数组之后,之后将商品对象删除,最后调用 DAO 层方法,把商品图片路径数组作为参数,将图片删除。

## 4.6.4 持久层的实现

### 1. 商品添加

使用继承的方法 BaseDAO 中提供的基本的方法 save 对数据进行保存,BaseDAO 是所有实体 DAO 层都继承的类,该类实现了多种基本的数据库操作的方法,BaseDAO

类是基于 Java 的泛型机制的，其他 DAO 类继承以自己实体为泛型参数的 BaseDAO 类，在服务器进行创建具体的 DAO 类时，会调用器父类的构造方法，采用 Java 的反射机制从而获取到继承的 BaseDAO 中传入的泛型值的类型，通过该类型的名字，便可知道该类型对应的 POJO 类的类名，从而在采用 sql 语句进行操作时可方便的与数据库中的表相对应。

## 2. 商品修改

商品的修改用的是 BaseDAO 中的方法，直接将对象持久化数据库中即可。

## 3. 商品删除

商品删除采用的依然是 BaseDAO 中基本的删除实体的方法实现的。

# 4.7 促销管理的设计与实现

## 4.7.1 促销管理的页面设计

促销管理包括促销消息的添加，修改和删除，界面效果如图 4-16、4-17 所示：

折扣管理									
<div>添加 修改 删除</div>									
	<input type="checkbox"/>	专柜名称	专柜图标	标题	详情	地址	开始时间	结束时间	海报
1	<input type="checkbox"/>	洛诗琳		端午节全场六折	端午节将至，在洛诗琳购物端午节当天可享6折优惠	绵阳茂业百货女装区洛诗琳专柜	2014-05-30	2014-05-30	
2	<input type="checkbox"/>	恩曼琳		毕业季全场6折	又是一年毕业季，在这个难忘的季节里，恩曼琳全场6折，欢迎各位毕业季同学选购。	绵阳茂业百货女装区恩曼琳专柜	2014-05-22	2014-05-30	
<div>10 第 1 共 1 页</div>									

图 4-16 促销管理列表

折扣管理

填写折扣信息(带 \* 为必填项)

专柜

女装

恩曼琳

折扣标题 \*

毕业季全场6折

折扣详情 \*

又是一年毕业季,在这个难忘的季节里,恩曼琳全场6折,欢迎各位毕业季同学选购。

折扣地址 \*

绵阳茂业百货女装区恩曼琳专柜

折扣海报

浏览



折扣海报

浏览



开始日期 \*

2014-05-22

结束日期 \*

2014-05-30

确定

图 4-17 促销添加效果

#### 4.7.2 控制层的实现

##### 1. 促销添加

促销的添加,在浏览器端提交的字段为对应服务器端的 model 中对应的字段,之后将该 model 对象作为参数传递给业务逻辑层的方法。

##### 2. 促销修改

和之前的修改信息的操作类似,先将要修改的信息查出,之后调用业务逻辑层的方法。

### 3. 促销删除

在确认删除促销信息后，会将该促销信息的 id 作为参数传递给服务器，之后进行相关操作。

#### 4.7.3 业务逻辑层的实现

##### 1. 促销添加

将控制层传递进来的 model 对象中的封装的信息取出，封装到对应的 POJO 对象，调用 DAO 层方法。

##### 2. 促销修改

根据控制层中的 model 中的促销的 id，调用 DAO 层方法查出要修改的对象，将该对象中的信息进行修改，之后持久化到数据库。

##### 3. 促销删除

先查找出该对象，之后调用 DAO 层方法将该对象删除。

#### 4.7.4 持久层的实现

持久层的实现都是使用继承自 BaseDAO 中的方法实现的。

### 4.8 时尚讯息管理的设计与实现

时尚讯息的管理有添加、修改、删除，该功能的实现流程和之上的促销管理相同。

### 4.9 投诉建议管理的设计与实现

管理员对和手机客户端消息主要包括对消息回复。具体的控制层和业务逻辑层的实现和之前的各个功能的实现类似，即查找出用户提交的消息内容，进行回复时传递消息的 id，并将回复的内容添加到相应的表中，DAO 层的实现是 BaseDAO 中的基本的操作。

## 第 5 章 系统运行测试

### 5.1 系统测试环境

#### 5.1.1 系统测试硬件环境

茂业百货无线应用系统服务器设计与开发需要 PC 机一台和服务器一台，PC 机和服务器的配置要求为：

PC 机配置：CPU1.8GHz，内存 2G

服务器配置：CPU2.1GHz，内存 3G

#### 5.1.2 系统测试软件环境

操作系统：Windows xp 和 Windows 7

浏览器：Mozilla FireFox 和 Google Chrome

Web 应用服务器：GlassFish3

数据库：MySql5.1

#### 5.1.3 GlassFish 服务器的安装和配置

首先安装 JDK，GlassFish3 依赖的 JDK 版本要求是 JDK1.6 以上，本系统中使用的是 JDK1.6 版本，之后安装 GlassFish 服务器，安装完成之后，需要将系统中使用到的某些相关的 jar 文件添加到 GlassFish 的安装目录之中，将 eclipselink.jar，mysql-connector-java-5.1.7-bin.jar 两个 jar 包添加到 glassfish3/glassfish/lib 目录下，将 btrace-agent.jar，btrace-boot.jar，flashlight-agent.jar 添加到 glassfish/lib/monitor 目录下，完成服务器的配置。

### 5.2 服务器系统测试

#### 5.2.1 测试需求概述

系统测试主要针对的是系统中对信息进行操作正确性，以及一些兼容性问题，具体的测试需求有：

1. 系统管理页面测试：对系统管理界面中的显示正确性，对部分文本域的检查，页面跳转等测试；

2. 登录功能测试：对用户登录系统时，对用户名，密码，验证码，以及向用户传递管理菜单的正确性测试；
3. 商家管理模块各个菜单功能的测试：对专柜的海报、专柜分类、专柜等的添加、修改、删除等的测试，操作后与数据库中的一致；
4. 商品管理的测试：包含对商品添加、修改、删除的测试；
5. 消息管理的测试：对消息查看的测试；
6. 时尚潮流测试：包括对时尚潮流讯息的添加、修改、删除的测试。

### 5.2.2 测试用例及结果

依据系统实现的各项功能，对系统进行测试，以下为测试时的用例数据表。

表 5-1 登录模块测试用例表

测试编号	输入	预期结果	实际结果
Login-01	用户名: mymy 密码: 123456	登录成功, 获取到管理员所拥有的菜单	进入主界面, 显示管理菜单列表
Login-02	用户名: abc 密码: 123456	跳转到提示信息错误的页面	跳转到错误页面, 显示“用户名或密码错误”
Login-03	用户名: mymy 密码: asdfg	跳转到提示信息错误的页面	跳转到错误页面显示“用户名或密码错误”

表 5-2 商家管理模块测试用例表（部分）

测试编号	输入	预期结果	实际结果
Company-01	点击商家海报菜单	显示现有海报列表	与预期结果一致

续表：

Company-02	填写海报信息： 海报：选择图片 加载次序：3	显示提示框添加成功	跳转到商家海报列表页面，并提示添加成功，海报列表页面自动刷新，有新海报信息
Company-03	修改海报信息：修改加载次序为3的海报图片，点击确定	显示提示框修改成功	跳转到商家海报列表页面，提示修改成功，修改的图片得到更新
Company-04	选中加载次序为4的海报，点击删除按钮	显示提示框删除成功	显示删除成功，并刷新了信息列表
Company-05	填写分类信息：父分类：1楼璀璨时尚馆 专柜分类名称：家具 排列序号：21 专柜分类说明：家具	显示提示框分类添加成功	跳转到分类列表界面，提示添加成功，可在列表中看到添加的信息
Company-06	修改商家信息：选择洛诗琳商家，将其地址改为四川省绵阳市	显示提示框修改成功	跳转到商家列表界面，提示修改成功，并自动刷新信息列表，可看到地址已经更新
Company -07	修改专柜海报信息：选择加载次序为1的商家，将加载次序更改为2	显示提示框修改成功	跳转到专柜海报列表界面，提示修改成功，信息得到刷新

表 5-3 商品信息管理测试用例表（部分）

测试编号	输入	预期结果	实际结果
Commodity-01	点击商品管理菜单	显示所有商品的信息列表	与预期结果一致
Commodity-02	点击添加按钮	跳转到填写商品信息界面	跳转到填写商品信息界面，信息填写处为空白
Commodity-03	填写商品信息：商品名称：夏季连衣裙 专柜：洛诗琳 商品缩略图：图片一张 商品详细图：图品三张	提示框显示添加成功	跳转到商品列表界面，提示添加成功，在列表中可以看到添加的商品信息。
Commodity-04	选中夏季连衣裙商品，点击修改按钮	跳转到商品信息修改页面，该商品信息被一一列出	与预期结果一致
Commodity-05	修改夏季连衣裙信息：将商品详细图删除一张	提示框显示修改信息成功	跳转到商品信息列表的页面，弹出修改成功提示框，夏季连衣裙详细图片减少一张
Commodity-06	选中韩国连衣裙，点击删除按钮	提示商品删除成功	弹出商品删除成功提示框，商品列表刷新，删除的商品不在列表之中

表 5-4 促销管理测试用例表

测试编号	输入	预期结果	实际结果
Activity-01	点击折扣管理菜单	显示全促销信息列表	与预期结果一致



续表:

Activity-02	点击添加按钮	跳转到活动信息的添加页面	跳转到活动信息的添加页面，可以输入活动的信息
Activity-03	修改活动信息：选择端午节全场 6 折活动，将活动结束时间修改为 5 月 31 日	提示修改成功	跳转到活动信息显示页面，弹出提示框，修改成功，可看到活动结束时间已经是 5 月 31 日
Activity-04	选中毕业季全场六折活动信息，点击删除按钮	弹出确认删除提示框，点击确定，提示删除成功	弹出删除成功提示框，自动刷新活动列表，被删除的活动记录消失

表 5-5 时尚潮流讯息测试用例表

测试编号	输入	预期结果	实际结果
Fashion-01	填写时尚信息：标题：牛仔裤时尚 主要内容：牛仔裤很时尚 主图：图片一张 时尚图片&说明：图片一张 说明：牛仔裤很时尚	弹出提示框添加成功	弹出添加成功提示框，跳转到时尚信息列表页面，可看到添加的时尚信息
Fashion-02	修改时尚信息：选择牛仔裤很时尚，将主要内容更改为：牛仔裤真的很好看	弹出提示框修改成功	弹出修改成功提示框，跳转到时尚信息列表，可看到主要内容已经变化变为牛仔裤真的很好看

续表：

Fashion-03	选中牛仔裤很时尚时尚信息	弹出确认对话框，点击确定，提示删除成功	弹出确认对话框，时尚信息列表刷新后该信息消失
------------	--------------	---------------------	------------------------

表 5-6 消息管理测试用例表

测试编号	输入	预期结果	实际结果
Message-01	添加消息： 消息标题：小米手机降价了 消息内容：小米手机真的降价啦 点击确定按钮	提示消息添加成功	提示消息添加成功，消息列表中出现添加的内容
Message-02	选中之前添加的消息点击删除按钮	提示消息删除成功	提示删除成功，该消息在消息列表中消失

表 5-7 意见反馈测试用例表

测试编号	输入	预期结果	实际结果
Suggest-01	点击投诉建议按钮	显示出所有建议内容的列表	与预期结果一致

### 5.2.3 测试总结

经过多次反复测试，上述测试用例在谷歌 Chrome 和 FireFox 浏览器中均运行正常，期间遇到了一些问题，经过仔细测试修改都得到解决，并对某些地方的页面进行了修改，目前系统运行正常，有些问题可能没有测试出来，需要后期继续完善。

## 结 论

茂业百货无线应用系统服务器设计与实现，完成了绵阳茂业百货中商家的管理、商品管理、促销管理、时尚元素、消息管理和建议反馈等功能。

整个系统的设计与实现建立在 JavaEE 企业级应用框架之上，功能分层实现，代码结构清晰，开发过程中涉及到的知识点较多，在设计与实现系统中如何尽可能的使代码得到复用，并且保证实现层次的清晰成为一个难点，经过此次设计开发，总结出系统具有以下优点：

1. 系统开发效率高：系统中应用到了几个 JavaEE 中流行的框架，Struts2、Spring3 和 JPA，这些框架使服务器端的开发变得容易并且高效；
2. 系统可扩展性好：系统功能实现过程中，对功能进行分层实现，层与层之间高度分离，降低了他们之间的耦合度，方便进行扩展；
3. 可维护性好：系统中用到的文件、类名以及参数、变量名等都采用与实际的功能相关的英文单词作为标识，并在一些必要的地方写有注释，方便了以后进行维护；
4. 安全性好：系统中使用的 Spring Security 框架，对用户登录以及对资源的访问进行拦截判断，保证了资源的安全性，对用户的密码进行不可逆的 MD5 加密，使用户信息得到了安全性的保障。

系统还有一些需要进行完善和改进的地方：

1. 系统功能不全面，系统中某些功能实现上还可以进一步改进，还可以添加一些功能来进一步完善系统。
2. 测试不完整，由于系统目前的功能较少，肯定有些方面还存在一些问题，细节上应该还有问题可以改进。

通过这几月对该系统的开发，加深了自己对 JavaEE 框架的理解，在开发过程中对使用的框架一边进行学习，一边进行实践，使自己的动手能力有了很大的提高，同时通过这次开发提高了我对软件开发的信心、学习探索热情，为今后在计算机方面的发展做了一个很好的开端。

## 致 谢

茂业百货无线应用系统服务器设计与实现以及学位论文的完成是在导师杨雷老师的精心指导下完成的。杨老师严谨的治学态度，严肃的科学态度，深深的激励着自己要好好努力。从开始选择课题到系统的完成，杨老师在整个过程中都给了我很多指导，使我学到了很多知识。自进入杨老师的实验室以来，不仅在学业上得到了杨老师的精心指导，同时在思想上和生活上也得到了很大的帮助，在此谨向辛勤教学的杨老师致以诚挚的谢意和崇高的敬意。

同时还要感谢实验室的学姐学长，在他们的帮助下，当遇到问题时才一步一步的解决，在进入实验室后正是在他们的指导帮助中渐渐的学会了基本的开发知识，还要感谢在实验室中的学弟学妹们，他们在实验中的学习的认真态度和积极向上的生活态度时刻的在促使我积极向上。

最后还要感谢同我一起完成毕业设计的同学们和父母，有他们的鼓励我才完成了系统的设计与实现。

## 参考文献

- [1] 高金勇, 陈晓建等.JavaScript+jQuery 从入门到精通.北京:化学工业出版社,2012.
- [2] 王珊, 萨师煊.数据库系统概论.北京:高等教育出版社,2012.
- [3] 李刚, 轻量级 JavaEE 企业应用实战(第 3 版) .北京:电子工业出版社,2011.
- [4] 刘京华等.Java Web 整合开发.北京:清华大学出版社,2010.
- [5] 刘伟, 张利国.Hibernate 开发与实战.北京:电子工业出版社,2009.
- [6] (加)库尼亚瓦(Kurniawan,B.).深入浅出 Struts 2[M].上海: 人民邮电出版社, 2009: 26-58.
- [7] 杨少波等.J2EE 项目实训:Struts 框架技术.北京:清华大学出版社,2008.12.
- [8] 杨少波, 顾益军等.J2EE 项目实训:Spring 框架技术.北京:清华大学出版社,2008.
- [9] Allan Vermeulen .Java 编码规范.北京:人民邮电出版社,2003.7.
- [10] Rod Johnson, Juergen Hoeller, Keith Donald.Spring Framework Reference Documentation[CP].  
<http://docs.spring.io/spring/docs/3.2.9.RELEASE/spring-framework-reference/htmlsingle/>,2012
- [11] Peter Tarasewich,RobertC. Nickerson,Merrill Warkentin.Wireless/Mobileecommerce:technologies, applications,and issues.Seventh AmericasConference on Information Systems 2011.
- [12] Christian Bauer, Gavin King.Java persistence with Hibernate .Beijing : Posts&Telecom Press, 2007.
- [13] Mark Richards.Java Transaction Design Strategies[M].Prentice Hall PTR,2006.
- [14] Bruce Eckel.Thinking in Java (4th Edition) [M].Prentice Hall PTR,2006:6-78.
- [15] James A. Senn .The Emergence of M-Commerce. Computer 2000.12