校园周边商铺系统的设计与开发

数学与信息学院 计算机科学与技术专业
105032016001 曹德梓 指导老师 林岭

【摘 要】本论文以网上商铺为核心，主要论述了如何创建出校园周边商铺系统。整个系统分为三个部分：前端游客展示系统，店铺店主管理系统，以及超级管理员系统。游客可以通过登录前端游客展示系统查询校园周边的商铺信息以及相关的产品，商铺店主可以通过店铺店主管理系统进行商铺信息和商品的相关修改，超级管理员通过超级管理员系统进行审核和修改页面信息等相关操作。该系统的所有部分均前后端分离，前端采用 MUI，通过 Ajax 与后台交互。后台是基于 SpringBoot 的框架开发。

【关键词】SpringBoot; MUI;网上商铺;前后端分离

目 录

1 引言

1.1 系统背景以及目的

随着“互联网+”的发展，越来越多人选择在网上寻找相关的店铺信息以及相关的产品，而作为传统的实体店铺为了提升竞争力，方便顾客了解店铺的相关信息，将店铺信息放在互联网中。尤其对于学生这个群体，他们更倾向于先从网上获取相关的信息，然后再选择去相应的店铺消费。相比以前全部线下交易，线上线下结合的交易模式因为其方便性，扩展性，丰富性越来越成为主流。事实证明，线上线下结合的方式，不仅有利于店主丰富营销手段，提高销售量，增加店铺利润，还有利于学生消费前了解信息，有计划地安排消费行程。

此系统配合时代潮流，结合微信公众号。将微信公众号作为入口，游客和店主都可以通过微信信息直接认证登录，增加了该系统的便利性。

1.2 系统可行性分析

1.2.1 技术可行性

此系统采用 Eclipse 工具进行开发，按照 MVC 设计模式进行软件的架构搭建。前端采用轻量级的 MUI 框架，后台使用 SpringBoot 框架，ORM 框架使用 Mybatis, 数据存储部分使用关系型数据库 MySql 和非关系型数据库 Redis。使用面向对象语言 Java 进行功能逻辑的编写。

系统前端采用的是 MUI 框架，MUI 基于响应式布局设计的框架，能够有效的使用不同的移动设备布局的问题。前端数据的操作使用的是 jQuery 进行处理，极大程度上提高了数据处理的效率，同时通过异步加载技术进行数据传输，减少访问服务器的次数。SpringBoot 是用来简化 Spring 应用初搭建以及开发过程，是一个整合很多可插拔的组件（框架），内嵌许多使用工具如 Tomcat 等，方便开发。使用 Redis 缓存技术，可以有效的减少访问数据库的频率，有效提高了系统的运行速度。使用 Maven 工具进行项目管理，方便添加和管理所需要的 Jar 包。本系统所采用的技术均是成熟稳定的，这使得系统的实现成为了可能。

1.2.2 经济可行性

本系统只需要使用的用户关注的微信公众号，从公众号的入口就登录系统，意味着只需要移动设备就可满足系统的操作。所需要的费用仅为服务器的租赁和域名的费用。极大程度上节省财力物力。依附目前国内主流的即时通讯软件—微信，使得访问该商铺的人流量大大增加。

1.2.3 操作可行性

游客通过关注微信公众号的方式，访问前端展示系统时，系统会自动获取该用户的信息，实现自动登录和注册的功能。游客可以通过商品分类，区域分类找到想要的信息。店铺管理员访问店铺管理系统的时候，系统也会自动获取用户的信息，完成注册和登录。店家可以修改自己的店铺信息，添加店铺，添加商品，绑定本地账号等操作。超级管理员系统则需要对应权限的账号登录，可以有对店铺申请进行审核，增加修改页面信息等操作。操作难度不高，方便可行。

2. 系统开发涉及的技术介绍

2.1 Eclipse

Eclipse 是一个开源的工具，作为 Java 的集成开发环境使用。因为其免费，功能强大，支持多种插件安装等特点，已经成为主流的开发工具之一。本项目使用 Eclipse 开发工具开发。

2.2 SpringBoot

Spring Boot 是由 Pivotal 团队提供的全新框架，其设计目的是用来简化新 Spring 应用的初始搭建以及开发过程。该框架使用了特定的方式来进行配置，从而使开发人员不再需要定义样板化的配置。通过这种方式，Spring Boot 致力于在蓬勃发展的快速应用开发领域(rapid application development)成为领导者。本项目通过 SpringBoot 简化配置，达到快速搭建的目的。

2.3 MUI

MUI 是一套前端框架，是基于 HTML5+功能开发，具有响应式和轻量的特点。本项目在搭建前端页面的时候，使用官网的样式，简约优雅。

2.4 Mybatis

MyBatis 是一款优秀的持久层框架，它支持定制化 SQL、存储过程以及高级映射。MyBatis 避免了几乎所有的 JDBC 代码和手动设置参数以及获取结果集。MyBatis 可以使用简单的 XML 或注解来配置和映射原生信息，将接口和 Java 的 POJOs(Plain Ordinary Java Object, 普通的 Java 对象)映射成数据库中的记录。在项目中每个 Dao 层的接口对应一个 xml 文件，对数据库的增删改查操作的语句写在文件中。

2.5 jQuery

jQuery 是一个快速、简洁的 JavaScript 框架，是继 Prototype 之后又一个优秀的 JavaScript 代码库（或 JavaScript 框架）。jQuery 设计的宗旨是“write Less，Do More”，即倡导写更少的代码，做更多的事情。它封装 JavaScript 常用的功能代码，提供一种简便的 JavaScript 设计模式，优化 HTML 文档操作、事件处理、动画设计和 Ajax 交互。本项目中，通过 jQuery 实现页面的逻辑，封装和传输前端的数据，实现前端和后端的交互。

2.6 Redis

Redis 是一个 key-value 存储系统，支持 string(字符串)、list(链表)、set(集合)、zset(sorted set --有序集合)和 hash（哈希类型）。在项目中通过 jedis 对前端展示系统信息进行缓存，在没有修改数据库信息的情况下，不用再次访问数据库，而是从 redis 中读取信息，增加信息的读取速度。

2.7 Java

Java 是一种面向对象语言，具有简单性、面向对象、分布式、健壮性、安全性、平台独立与可移植性、多线程、动态性等特点 。Java 可以编写桌面应用程序、Web 应用程序、分布式系统和嵌入式系统应用程序等。本项目使用 Java 语言编写。

2.8 MySql

MySQL 是一个关系型数据库管理系统，由瑞典 MySQL AB 公司开发，属于 Oracle 旗下产品。MySQL 是最流行的关系型数据库管理系统之一，在 WEB 应用方面，MySQL 是最好的 RDBMS (Relational Database Management System，关系数据库管理系统) 应用软件之一。本项目中数据存储使用 MySql 5.7 数据库存储。

2.9 微信公众号平台

微信公众平台主要面向名人、政府、媒体、企业等机构推出的合作推广业务。在这里可以通过渠道将品牌推广给线上平台作用。微信公众平台于 2012 年 08 月 23 日正式上线，曾命名为“官号平台”和“媒体平台”，创造更好的用户体验，形成一个不一样的生态循环。本项目中，通过微信公众号，方便获取用户信息，提高该系统的方便性。

3 系统总体设计

3.1 需求分析

3.1.1 功能模块

由于采用微信公众号号，系统在用户登录的时候自动调用微信接口，获取用户的相关信息，自动完成登录注册。游客进入前端展示系统时，可以通过店铺类别，地点信息，标题关键字查找等方式查找相应的信息。当店家用户进入店家管理系统的时候，系统会自动创建其用户信息。用户可以通过本地账号绑定，实现微信账号和本地绑定，店主除了可以使用微信登录以外，还可以通过本地的账号密码绑定。由于涉及安全信息，本地账号密码使用 MD5 加密。店家用户通过添加商铺功能，添加商铺的商铺名，地点，联系电话等基本信息。上传店铺的首页图和详细图。填写验证码，即可完成申请。添加商铺是否成功需要超级管理员进行审核。店家通过商铺信息修改，修改商铺信息。店家用户通过商店信息模块可以添加，下架，编辑商品。通过超级管理员系统可以对页面头条的轮播图进行增加，删除，该改变轮播图顺序。超级管理员可以通过类别管理功能，对店铺的一级类别，二级类别进行添加，修改，删除。超级管理员可以通过区域管理，对店铺地理位置添加，修改，删除区域类别。超级管理员可通过账号管理，查看，修改，删除用户的信息。超级管理员可通过商铺管理，对申请的项目进行审核，并给出建议。同时对店铺有添加，修改，删除的权限。

3.1.2 UML

统一建模语言(Unified Modeling Language，UML)是一种为面向对象系统的产品进行说明、可视化和编制文档的一种标准语言，是非专利的第三代建模和规约语言。UML 使用面向对象设计的建模工具，但独立于任何具体程序设计语言。

游客用户通过微信公众号进图前端展示系统，在系统中通过店铺查询和商品查询功能查询校园周边商铺的信息。详细如图 3-1 游客用例所示。

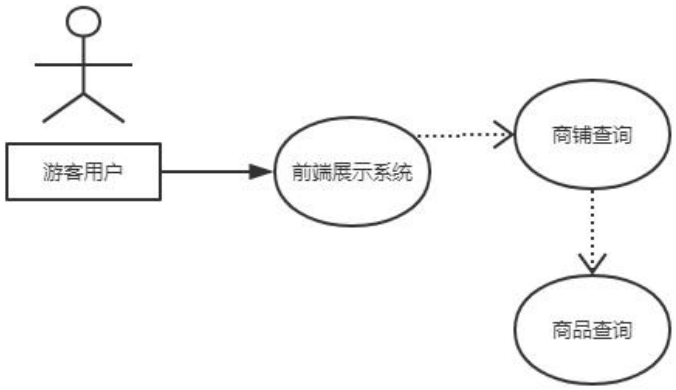

图 3-1 游客用例图

店家用户关注微信公众号，进入商铺管理系统可对商铺进行查询，新增，修改等操作。对每个商品上下架，信息修改和增加操作等，图 3-2 店家用户用例图所示

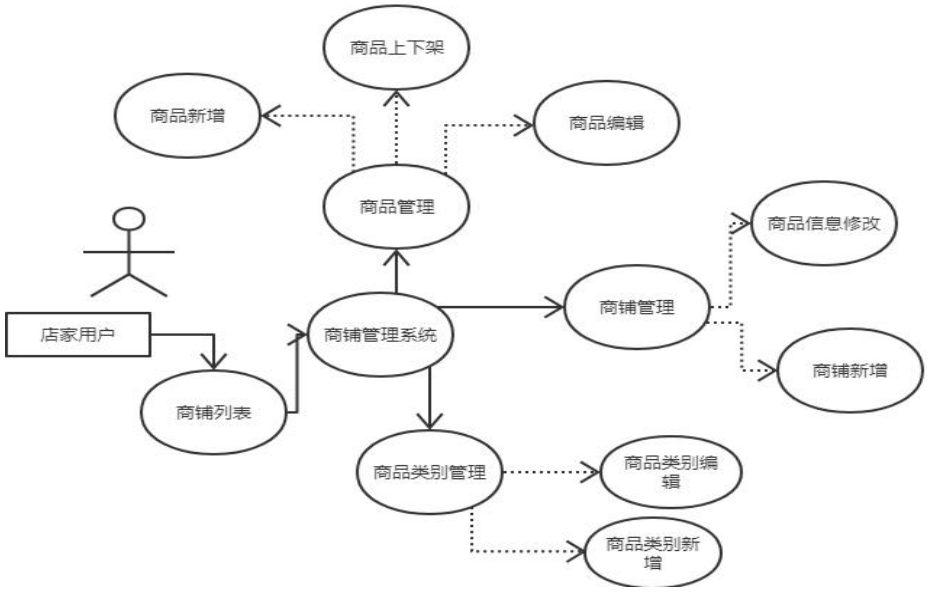

图 3-2 店家用户用例图

超级管理员登录超级管理员系统，在系统中可对头条管理，类别管理，账号管理，商铺管理和管理进行操作。详细如图 3-3 超级管理员用例图

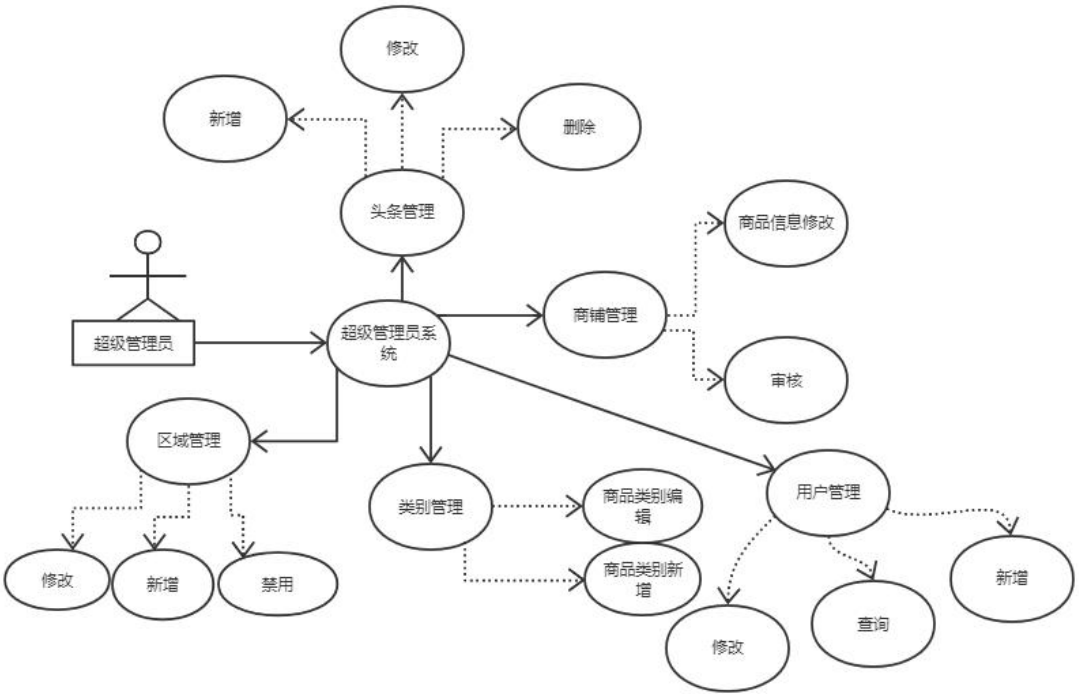

图 3-3 超级管理员用例图

3.1.3 数据字典

系统数据使用 Mysql 数据库存储，总共用 11 张表

1） 区域信息表（tb_area）
2） 头条信息表（tb_headline）
3） 本地账户信息表（tb_local_auth）
4） 用户信息表（tb_person_info）
5） 商品信息表（tb_product）
6） 商品类别信息表（tb_product_category）
7） 商品图片表（tb_product_img）
8） 商铺信息表（tb_shop）
9） 商铺-用户连接表（tb_shop_auth_map）
10） 商铺类别表（tb_shop_category）
11） 用户微信信息表（tb_wechat_map）

详细如 ER 图 3-4 所示：

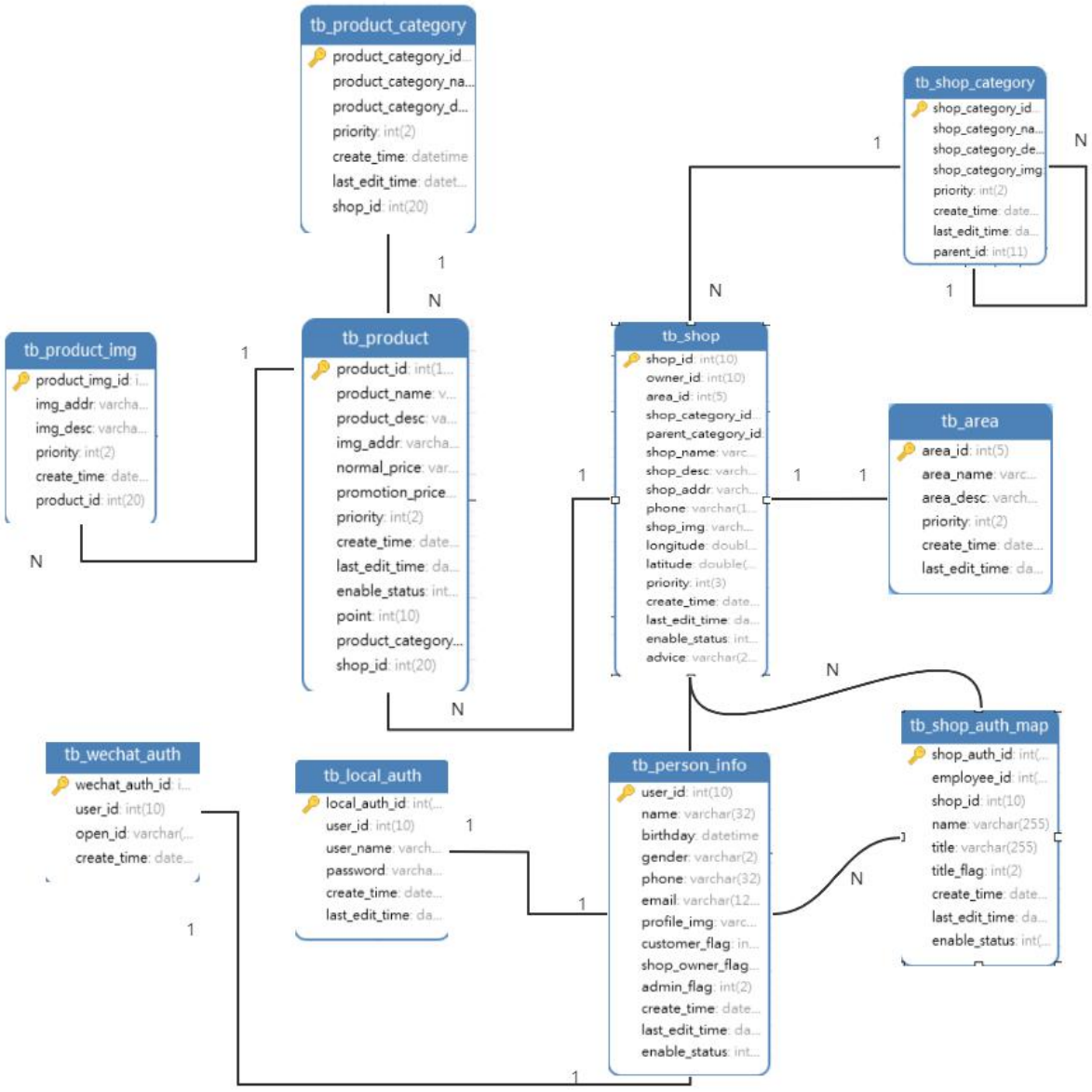

图 3-4 ER 图

数据库各个表的数据详细说明：

1）区域信息表：记录了当前系统区域的类别。包含区域的具体信息。详细字段如下表格 3-1 所示。

表 3-1 区域信息表

属性名	类型	详细说明	备注
area_id	int	区域主键	主键，自增长，不为空
area_name	Varchar(200)	区域名称	不为空
area_desc	Varchar(1000)	区域详细地址	
priority	int	权重	不为空
create_time	datetime	创建时间	
Last_edit_time	datetime	最后修改时间	

2）头条信息管理表：头条信息管理表记录了前段展示系统轮播图的信息。详细字段如表 3-2 所示。

表 3-2 头条信息管理表

属性名	类型	详细说明	备注
line_id	int	头条信息记录主键	主键，自增长，不为空
line_name	varchar(1000)	头条名称	
line_link	Varchar(2000)	头条连接	
line_img	Varchat(2000)	头条图片的展示地址	不为空
Priority	Int	权重	不为空
enable_status	int	是否可用	不为空
create_time	Datetime	创建时间	

3）本地账户信息表：本地账户信息表存储店主用户登录后绑定的本地账户的信息。详细字段如表 3-3 所示。

表 3-3 本地账户信息表

属性名	类型	详细说明	备注
local_auth_id	int	本地账户用户主键	主键，自增长，不为空
user_id	int	用户信息表的主键	外键
user_name	Varchar(128)	用户名	不为空
Password	Varchar(128)	密码	不为空
Create_time	datetime	创建日期	
last_edit_time	datetime	最后修改日期	

4）用户信息表：用户信息表是所有用户的信息的总表，这张表包含用户的所有所有信息。详细字段如表 3-4 所示。

表 3-4 用户信息表

属性名	类型	详细说明	备注
User_id	int	用户信息主键	主键，自增长，不为空
name	Varchar(32)	用户名	
birthday	datetime	出生日期	
gender	varchar	性别	
phone	Varchar(32)	手机号码	
email	Varchar(128)	电子邮件	
Profile_img	Varchar(124)	头像	
Customer_flag	int	客户标识	不为空
Shop_owener_flag	int	店主标识	不为空
Admin_flag	int	超级管理员标识	不为空
Create_time	datetime	创建时间	
last_edit_time	datetime	信息最后修改时间	
Enable_status	int	用户状态	不为空

5）商品信息表：商品信息表包含店铺中商品的信息，详细字段如表 3-5 所示。

表 3-5 商品信息表

属性名	类型	详细说明	备注
Product_id	int	商品主键	主键，自增长，不为空
Product_name	Varchar(100)	商品名称	不为空
Product_desc	Varchar(2000)	商品描述	
Img_addr	Varchar(2000)	商品图片	不为空
Normal_price	Varchar(100)	商品价格	不为空
priority	int	权重	不为空
Create_time	datetime	创建时间	
Last_edit_time	datetime	最后修改时间	
Enable_status	int	商品状态	不为空
Product_category_id	int	商品类别主键	外键，不为空
Promotion_price	int	商品打折后的价格	不为空
Shop_id	Varchar(100)	商品所属商铺的主键	外键，不为空

6）商品类别表：商品类别表包含商品所属类别的信息，详细字段如表 3-6 所示。

表 3-6 商品类别表

属性名	类型	详细说明	备注
Product_category_id	int	商品类别主键	主键，自增长，不为空
Product_category_name	Varchar(100)	商品类别名称	不为空
Product_category_desc	Varchar(500)	商品类别描述	
priority	int	权重	不为空
Create_time	datetime	创建时间	
Last_edit_time	datetime	最后修改时间	
Shop_id	int	商铺的主键	外键，不为空

7）商品图片表：商品图片表包含商品详细图的地址，详细字段如表 3-7 所示。

表 3-7 商品图片表

属性名	类型	详细说明	备注
Product_img_id	Int	商品图片主键	主键，自增长，不为空
Img_addr	Varchar(2000)	图片地址	不为空
Img_desc	Varchar(2000)	商品图片说明	
priority	int	权重	不为空
Create_time	datetime	创建时间	
Product_id	int	商品的主键	外键，不为空

8）商铺信息表：商铺信息表包含商铺的详细信息，详细字段如表 3-8 所示。

表 3-8 商铺信息表

属性名	类型	详细说明	备注
Shop_id	int	商铺信息主键	主键，自增长，不为空
Owner_id	Int	店长主键	外键，不为空
Area_id	Int	区域主键	外键，不为空
Shop_category_id	Int	店铺类别主键	外键，不为空
Parent_category_id	Int	店铺一级类别主键	外键，不为空

Shop_name	Varchar(256)	店铺名称	不为空
Shop_desc	Varchar(1024)	店铺详细	
Shop_addr	Varchar(200)	店铺详细地址	
phone	Varchar(128)	店铺联系号码	
Shop_img	Varchar(1024)	商铺图片地址	
priority	Int	权重	不为空
Create_time	datetime	创建时间	
Last_edit_time	Datetime	最后修改时间	
Enable_status	Int	商铺状态	不为空
advice	Varchar	管理员	

9）商铺-用户连接表：商铺-用户连接表为商铺和用户的中间表信息，因为用户和商铺属于多对多关系，所以产生中间表。详细字段如表 3-9 所示。

表 3-9 商铺-用户连接表

属性名	类型	详细说明	备注
Shop_auth_id	int	商铺-用户记录主键	主键，自增长，不为空
Employee_id	id	用户信息主键	外键，不为空
Shop_id	id	商铺主键	外键，不为空
title	Varchar(255)	店员或者店主	不为空
Create_time	datetime	创建时间	
Last_edit_time	datetime	最后更新时间	
Enable_status	int	状态	不为空

10）商铺类别表：商铺类别表包含商铺的类别信心。详细字段如表 3-10 所示。

表 3-10 商铺类别表

属性名	类型	详细说明	备注
Shop_category_id	id	商铺类别主键	主键，自增长，不为空
Shop_category_name	Varchar(100)	类别名称	不为空
Shop_category_desc	Varchar(1000)	类别说明	
Shop_category_img	Varchar(2000)	类别图片	
Priority	Int	权重	
Create_time	datetime	创建时间	
Last_edit_time	datetime	修改时间	
Parent_id	int	商铺类别父主键	

11）微信用户信息表：微信用户信息表用于存储微信用户的相关信息。它通过用户信息表（tb_person_info）的主键，与本地用户信息表（tb_local_auth）绑定。详细字段如 3-11 表所示。

表 3-11 微信用户信息表

属性名	类型	详细说明	备注
Wechat_auth_id	int	微信用户主键	主键，自增长，不为空
User_id	int	用户信息主键	外键，不为空
Open_id	Varchar(512)	微信用户的唯一标识	不为空

Create_time	datetime	创建时间	

4 系统详细设计

4.1 系统功能界面

本系统界面展示了各个功能模块。其中每个功能模块都是结合 MUI 和 jQuery 进行传统的增删改查操作。通过 Ajax 的异步传输，可以使网页异步刷新，提高用户的体验感。同时，因为 MUI 是响应式布局，所以能够很好地适应各个设备之间展示。游客系统和店家管理系统依照类似系统的 App 布局，尽量还原其布局。超级管理员系统采用传统的侧边栏布局，给管理者清晰，简洁之感。布局简各个功能模块清晰，都配有图片，方便用户更好地识别和使用。

4.1.1 游客和店家入口界面

用户只需要关注微信公招号，从微信公众号的游客入口/商家入口进入，即可进入对应的界面。游客和店家入口界面介绍见表 4-1-1，登录效果见图 4-1。

表 4-1-1 进入游客界面

用例	用户自动登录或注册
简要描述	用户关注微信公众号进入游客入口/商家入口时，自动登录注册
参与者	任何用户
前置条件	关注微信公共号，点击游客入口/商家入口
主事件流	游客点击游客入口/商家入口，活动流程开始触发 UC4.1.1 游客关注微信公众号，点击游客入口
备选流	1）网络延迟，无法获取用户微信信息
后置条件	成功跳转对应的界面首页

页面效果如下：

图 4-1 游客和店家入口界面

实现代码以及详细说明：

点击游客入口/商家入口以后，会自动跳转跳转后台接口。后台接口调用微信提供的接口获取微信用户信息。通过返回的 openId, 到数据库中查找该用户是否注册，若没有注册，则自动注册，然后跳转对应的首页。若已经注册，则直接跳转对应的首页。

```
@RequestMapping(value = "/logincheck", method = { RequestMethod.GET })
public String doGet(HttpServletRequest request, HttpServletResponse response) {
        String code = request.getParameter("code");
        String roleType = request.getParameter("state");
        WechatAuth auth = null;String openId = null;WeiXinUser user = null;
        if (null != code) {
                UserAccessToken token;
                try {
                        token = WeiXinUserUtil.getUserAccessToken(code);
                        log.debug("weixin login token:" + token.toString());
                        String accessToken = token.getAccessToken();
                        openId = token.getOpenId();
                        user = WeiXinUserUtil.getUserInfo(accessToken, openId);
                        log.debug("weixin login user:" + user.toString());
                        request.getSession().setAttribute("openId", openId);
                        auth = WechatAuthService.getWechatAuthByOpenId(openId);
                } catch (IOException e) {
                        log.error("error in getUserAccessToken or getUserInfo or findByOpenId: "
                                        + e.toString());
                        e.printStackTrace();
                }
        }
        if (FRONTEND.equals(roleType)) {
                PersonInfo personInfo = WeiXinUserUtil
                                .getPersonInfoFromRequest(user);
                if (auth == null) {
                        personInfo.setCustomerFlag(1);
                        auth = new WechatAuth();
                        auth.setOpenId(openId);
                        auth.setPersonInfo(personInfo);
                        WechatAuthExecution we = WechatAuthService.register(auth, null);
                        if (we.getState() != WechatAuthStateEnum.SUCCESS.getState()) {
                                return null;
                        }
                }
                personInfo = personInfoService.getPersonInfoById(auth.getUserId());
                request.getSession().setAttribute("user", personInfo);
                return "frontend/index";
        }
```

图 4-2 微信自动登录注册代码

4.1.2 游客展示页面

游客展示页面主要是提供给游客展示的页面。游客可以通过该页面通过类别，关键字，店铺名称查询相关的店铺，然后通过商铺查看相关的产品。游客页面见表 4-1-2。

表 4-1-2 游客页面

用例	浏览游客展示页面

简要描述	游客浏览游客展示页面，查询店铺信息
参与者	任何游客
前置条件	点击游客入口，进入游客展示页面
主事件流	游客进入游客展示页面，活动流程开始触发 UC4.1.2 页面上角是轮播图，下面是主要商铺的大类。点击类别展示对应类别商铺的信息。 UC4.1.2.1 进入店铺列表页面，页面上角可通过搜索关键字查询店铺信息。 UC4.1.2.2 通过店铺的二级类别查询店铺信息。 UC4.1.2.3 选择区域信息筛选店铺。 UC4.1.2.4 点击商铺图标，进入商铺详情。 UC4.1.2.4.1 进入商铺详情，页面上方显示基本店铺信息。 UC4.1.2.4.2 商铺分类栏目，通过商品分类筛选商品。 UC4.1.2.4.3 关键字搜索栏，搜索相关商品。 UC4.1.2.4.4 点击商品列表进入商品详情。
备选流	1. 网络原因导致一直加载
后置条件	继续浏览商铺或者商品信息

页面效果图如下：

图 4-3 游客首页

图 4-4 商铺列表页面

图 4-5 商品详细页面

实现代码以及详细说明：

游客页面代码核心功能就是通过前端选择的数据传送到后台，通过后台的逻辑判断进行

条件筛序，得出用户想要的信息。

后台核心代码如图 4-6：

```
@RequestMapping(value = "/listshops", method = RequestMethod.GET)
@ResponseBody
private Map<String, Object> listShops(HttpServletRequest request) {
        Map<String, Object> modelMap = new HashMap<String, Object>();
        int pageIndex = HttpServletRequestUtil.getInt(request, "pageIndex");
        int pageSize = HttpServletRequestUtil.getInt(request, "pageSize");
        if ((pageIndex > -1) && (pageSize > -1)) {
                long parentId = HttpServletRequestUtil.getLong(request, "parentId");
                long shopCategoryId = HttpServletRequestUtil.getLong(request,
                                "shopCategoryId");
                long areaId = HttpServletRequestUtil.getLong(request, "areaId");
                String shopName = HttpServletRequestUtil.getString(request,
                                "shopName");
                Shop shopCondition = compactShopCondition4Search(parentId,
                                shopCategoryId, areaId, shopName);
                ShopExecution se = shopService.getShopList(shopCondition,
                                pageIndex, pageSize);
                modelMap.put("shopList", se.getShopList());
                modelMap.put("count", se.getCount());
                modelMap.put("success", true);
        } else {
                modelMap.put("success", false);
                modelMap.put("errMsg", "empty pageSize or pageIndex");
        }

        return modelMap;
}
```

图 4-6 根据条件筛选信息代码

商品列表展示中，为了提高每次访问效率，在没有数据库更新的情况下，将数据存储在 Redis 当中。当第二次访问页面的时候，不需要访问 MySql 数据库，直接从 **Redis** 中读取。

部分代码图 4-7 如下：

```
@Override
public List<ShopCategory> getAllSecondLevelShopCategory()
                throws IOException {
        String key = SCLISTKEY + "ALLSECOND";
        List<ShopCategory> shopCategoryList = null;
        ObjectMapper mapper = new ObjectMapper();
        if (!jedisKeys.exists(key)) {
                ShopCategory shopCategoryCondition = new ShopCategory();
                shopCategoryCondition.setShopCategoryDesc("ALLSECOND");
                shopCategoryList = shopCategoryDao
                                .queryShopCategory(shopCategoryCondition);
                String jsonString = mapper.writeValueAsString(shopCategoryList);
                jedisStrings.set(key, jsonString);
        } else {
                String jsonString = jedisStrings.get(key);
                JavaType javaType = mapper.getTypeFactory()
                                .constructParametricType(ArrayList.class,
                                                ShopCategory.class);
                shopCategoryList = mapper.readValue(jsonString, javaType);
        }
        return shopCategoryList;
}
```

图 4-7 后台代码

4.1.3 商铺管理页面

商铺管理页面是店家所使用的。店家用户可以通过该系统添加自己的商铺，添加商品信息，添加商品类别，修改账号密码，绑定本地账号等功能。商铺管理页面见表 4.1.3。

表 4.1.3 商铺管理页面

用例	商铺管理页面
简要描述	店家用户进入商铺管理页面对商铺信息和用户信息进行管理
参与者	店家用户
前置条件	微信公众号点击商家入口，进入商铺管理页面
主事件流	用户进入商铺管理页面，活动流程开始。 UC4.1.3.1 页面上方是用户信息，下方是商铺列表。 UC4.1.3.1.1 点击退出系统按钮，进入登录页面。 UC4.1.3.1.1.1 账号密码输入进入商铺管理页面。 UC4.1.3.1.2 点击绑定本地账号按钮，进入绑定本地账号页面。 UC4.1.3.1.3 点击修改密码按钮，进入账号修改页面。 UC4.1.3.1.4 点击添加商铺按钮，进入添加商铺页面。 UC4.1.3.1.5 点击商铺信息，进入商铺管理页面。 UC4.1.3.1.5.1 点击商铺管理按钮，对该商铺信息进行修改。 UC4.1.3.1.5.2 点击商品管理按钮，对商铺进行添加，修改，上架下架处理。 UC4.1.3.1.5.3 点击商品类别管理，对商铺类别进行添加和修改。
备选流	1）跳转页面时，若服务器没有相关的用户信息，则跳转失败，返回登录页面。 2）登录页面中，账号密码错误，显示失败。 3）验证码错误显示提交失败。 4）上传名称，图片等信息为空，显示图片失败。
后置条件	继续浏览操作

图 4-8 登录页面

图 4-9 修改密码页面

图 4-10 商品管理页面

前端通过用户点击按钮 ，携带相关参数提交后台。后台接收参数，处理请求。

4.1.4 超级管理员系统页面

超级管理员系统给具有最高权限的管理员使用。超级管理员可以审核商铺的申请，修改等操作，可以对区域信息进行更新，头条管理，用户信息管理等。详细如表 4.1.4 所示。

表 4.1.4 超级管理员页面

用例	超级管理员页面
简要描述	超级管理员登录超级管理员页面进行管理
主要参与者	超级管理员
前置条件	账号密码登录
主事件流	用户登录，活动流程开始。 UC4.1.4.1 头条管理，对游客首页的轮播图增删改查。 UC4.1.4.2 类别管理，对商铺的一级类别增删改查。 UC4.1.4.3 区域管理，对商区的区域信息增删改查。 UC4.1.4.4 账号管理，对账号信息进行管理。 UC4.1.4.5 商铺管理，对商铺信息进行增删改操作，以及审核。
备选流	
后置条件	继续浏览操作

图 4-11 超级管理员登录页面

图 4-12 超级管理员页面

4.2 系统测试

本系统实现完成后进行测试。避免出现设计过程中的错误和漏洞。以下对系统进行白盒测试。

4.2.1 避免无身份状态访问页面

由于一些页面的请求没有携带用户信息，后台无法判断用户信息，导致没有登录也可以访问没有权限的页面。

为了防止上述现象的发生，采用拦截器进行拦截，用户登录时，在 session 中添加用户信息，当用户请求其它页面时候，拦截器先判断 session 中是否有该用户的信息，如果有则请求通过，若没有，则返回登录页面。相关的后台代码如图 4-13：

```
@Override
public boolean preHandle(HttpServletRequest request,
                HttpServletResponse response, Object handler) throws Exception {
        Object userObj = request.getSession().getAttribute("user");
        if (userObj != null) {
                PersonInfo user = (PersonInfo) userObj;
                if (user != null && user.getUserId() != null
                                && user.getUserId() > 0 && user.getEnableStatus() == 1
                                && user.getShopOwnerFlag() == 1)
                        return true;
        }
        PrintWriter out = response.getWriter();
        out.println("<html>");
        out.println("<script>");
        out.println("window.open ('" + request.getContextPath()
                        + "/shop/ownerlogin','_self')");
        out.println("</script>");
        out.println("</html>");
        return false;
}
```

图 4-13 判断是否有用户信息

4. 2. 2 商铺注册测试

商铺注册是在店家通过微信公众号入口登录店家管理系统时，添加商铺的操作。要实现商铺注册的功能。为此做了白盒测试。测试用例间用例设计表 4.2.2.1,测试用例图见表 4.2.2.2：

表 4.2.2.1 测试用例设计表

	有效等价类	无效等价类
商铺名称	不为空	为空
商铺电话	不为空	为空
商铺区域	不为空	为空
商铺图片	不为空	为空
商铺类型	不为空	为空
验证码	不为空，验证码正确	验证码错误

表 4.2.2.2 测试用例表

编号	商铺名称	商铺区域	商铺电话	商铺类别	商铺图片	验证码	预期结果	实际结果
1	Test	东门	123	二手书	不为空	正确	成功	成功
2		东门	123	二手书	不为空	正确	失败	失败
3	Test		123	二手书	不为空	正确	失败	失败
4	Test	东门		二手书	不为空	正确	失败	失败
5	Test	东门	123		不为	正确	失败	失败

					空			
6	Test	东门	123	二手书		正确	失败	失败
7	Test	东门	123	二手书	不为空	错误	验证码错误	验证码错误

通过以上结论，注册店铺逻辑严谨，测试通过。

5. 结论

本文针对《校园商铺管理系统设计与开发》的设想，设计，架构与实践。在系统的设计中采用了 MUI 的前端框架，以实现更好的交互界面。MUI 框架能够很好的自适应屏幕大小，不会产生信息混乱的结果。后台使用了 SpringBoot 的开发框架，用于快速地搭建项目，减少了传统的 SSM 项目中繁杂的配置问题。使用了 Mybatis 持久化框架，通过提供 DAO 层，将业务逻辑和数据访问逻辑分离，使系统的设计更清晰，更易维护，更易单元测试。sql 和代码的分离，提高了可维护性。在系统展示过程中，为了提高系统的响应速度，加了 Redis 缓存数据库，用来减少访问数据库的频率。同时对于重要的账号信息采用了 MD5 加密，增强了系统信息的安全性。Maven 的 jar 包管理使得项目更好地管理和部署。该系统与微信公众号紧密联合，方便登录用户注册，同时也增加了该系统的便捷性。

本系统开发完成，并且成功部署在服务器上。用户可关注微信公众号就可以使用其功能。不过，还有许多需要改进的地方，比如在这个系统上其实还有很多功能可以完善，但是难度也大大地增加，今后会考虑逐渐地完善；在代码的规范方面做的不够，代码的不规范也许对自己没有多大的影响，但是对于其他人阅读自己的代码会造成较大的困扰，这也是需要改进的地方；代码的复用性不够，很多代码功能基本逻辑类似，但却写了很多遍，需要考虑把代码进一步抽象，提取可复用的代码，减少代码冗余。

通过这个项目从开始搭建到最后项目的部署，完整的学习到了一个软件系统研发的整套流程。从可行性分析，查看是否有条件和能力去实现这个系统，从需求分析，查看这套系统的意义是什么，可以解决社会中哪些痛点。接着需要整体设计系统，避免开发到最后出现项目结果与预期设想出现偏离的情况。然后对系统功能详细设计，有了功能目标，更快实现系统需要的功能。整体功能实现后，要逐步去测试，发现系统的漏洞和不足。最后，在服务器上搭建项目需要运行的环境，将完成好的项目部署上线。

技术的发展日新月异，通过这次的项目实践，不仅掌握了目前的主流技术，更重要的是掌握了主动学习的能力，只有保持终生学习的观念，才能不被技术的潮流淘汰，成为时代的弄潮儿。

6. 致谢

首先，我要感谢我的指导老师林岭老师，在设计此系统上，给了我足够的选择权，让我可以根据自己的兴趣开发这个系统，在时间的安排，导师给了非常明确的目标和步骤，督促自己完成此系统。在导师的督促和鼓励下，我对这次的项目有巨大的成就感，

其次，非常感谢大学四年陪伴我们的每一位老师，在学校和老师培养下，不仅仅让我掌握了专业的基础知识，更重要的是学会如何学习，提升自我。感谢学校和老师提高大量的资源和知识，让我得到成长和进步。

最后，非常感谢这个互联网时代，每个技术者都愿意分享自己的知识，让我在遇到

很多棘手的问题时候，能够很好得到解答。

参考文献

[1] 李刚. 疯狂 Java 讲义（第 2 版）：电子工业出版社，2014,2
[2] 赵景晖. Java 程序设计：北京机械工业出版社，2005：1-2
[3] Joshua.Effective Java 中文版第二版[M].北京： 机械工业出版社， 2017，156-208
[4] Paul.MySQL 技术内幕[M].北京: 人民邮电出版社， 2011，95-177
[5] Bruce，Java 编程思想第四版[M].北京：人民邮电出版社，2007，54-622
[6] 超高性能 key-value 数据库 Redis， 开源社区网[引用日期 2020-03-08]
[7] Craig Walls. SpringBoot 实战 北京：人民邮电出版社，2016，9
[8] 杨开振. Java EE 互联网轻量级框架整合开发第四版 北京：电子工业出版社，2017，6
[9] 微信公众号开发文档 [引用日期 2020.03.08]
[10] Douglas Crockford. JavaScript 语言精粹修订版 北京：电子工业出版社 2012.9
[11] Baron,Peter,Vadim.高性能 MySQL[M].北京：电子工业出版社,2013,40-50
[12] Steven, William. Java 设计模式第二版[M].北京：电子工业出版社，2012，84-88
[13] Charlie,Binu.Java.性能优化权威指南[M].北京：人民邮电出版社，2014，40-110
[14] Cay，Gray.Java 核心技术卷 II（高级特性）[M].北京机械工业出版社，2008，150-160
[15] 蒋卫祥，朱利华，闾枫.JavaEE 企业级项目开发.高等教育出版社,2018，2-20

Design and Development of Shop System Around Campus

CAO De-Zi 105032016001 Advisor：Lin Ling
Major in Computer Science and Technology College of Mathematics and Informatics

【**Abstract**】This paper mainly discusses how to create the system of shops around the campus. The whole system is divided into three parts: front-end tourist display system, store owner management system, and super administrator system. Visitors can log in the front-end visitor display system to query the information of the shops around the campus and related products. The shop owner can modify the shop information and commodities through the shop owner management system. The super administrator can review and modify the page information and other related operations through the super administrator system. All parts of the system are separated from the front end and the back end. The front end uses MUI and interacts with the back end through Ajax. The background is based on the Springboot framework development.

【**Key words**】**SpringBoot; MUI; Online Shop; Ajax**