# 世纪餐厅管理系统的设计与实现

摘 要

当一家餐厅的人员规模足够大时，人员的管理就很难靠人工完成，而世纪餐厅管理系统是基于Spring Boot框架进行开发的一款B/S架构的web应用程序，采用分布式开发，各模块之间功能相互独立，其主要功能侧重于后台的人员，菜品管理。数据存储方面采用流行的MySQL数据库，简单易上手，能帮助管理者更好的管理餐厅。

关键词 ：餐厅管理系统;Spring Boot;B/S架构;MySQL

ABSTRACT

When the staff scale of a restaurant is large enough, it is difficult to manage the staff manually. Century restaurant management system is a B / S Architecture Web Application Based on spring boot framework. It adopts distributed development, and the functions of each module are independent of each other. Its main functions focus on backstage staff and food management. Data storage using popular MySQL database, easy to use, can help managers better manage the restaurant.

Key words: restaurant management system; spring boot; B / S architecture; MySQL

目录

[世纪餐厅管理系统的设计与实现 1](#_Toc70191200)

[1 绪论 1](#_Toc70191201)

[1.1选题背景 1](#_Toc70191202)

[1.2国内外研究现状 1](#_Toc70191203)

[1.2.1国内研究现状 1](#_Toc70191204)

[1.2.2 国外研究现状 2](#_Toc70191205)

[1.3设计内容和意义 2](#_Toc70191206)

[1.3.1 设计内容 2](#_Toc70191207)

[13.2 意义 2](#_Toc70191208)

[1.4本文结构 2](#_Toc70191209)

[2开发工具及相关技术简介 4](#_Toc70191210)

[2.1 intelliJ IDEA 4](#_Toc70191211)

[2.2 JDK1.8 4](#_Toc70191212)

[2.3 MySQL 4](#_Toc70191213)

[2.4 Navcat premium 4](#_Toc70191214)

[2.5 Visual Studio Code 4](#_Toc70191215)

[2.6 Git 5](#_Toc70191216)

[2.7 java 5](#_Toc70191217)

[2.8 Spring Boot 5](#_Toc70191218)

[2.9 Bootstrap 5](#_Toc70191219)

[3系统分析 6](#_Toc70191220)

[3.1 需求分析 6](#_Toc70191221)

[3.2组织结构分析 6](#_Toc70191222)

[4 系统设计 7](#_Toc70191223)

[4.1 功能模块设计 7](#_Toc70191224)

[4.2数据库设计 8](#_Toc70191225)

[4.3用户界面设计 9](#_Toc70191226)

[4.3.1 登录界面 9](#_Toc70191227)

[4.3.2 系统主界面 9](#_Toc70191228)

[4.3.3 员工管理界面 9](#_Toc70191229)

[4.3.4 部门管理界面 9](#_Toc70191230)

[4.3.5 商品管理界面 9](#_Toc70191231)

[4.4 本章小结 9](#_Toc70191232)

[5系统实现 10](#_Toc70191233)

[5.1 登录功能模块 10](#_Toc70191234)

[5.2 查询管理模块 11](#_Toc70191235)

[5.3 其他功能模块 12](#_Toc70191236)

[6 系统测试 13](#_Toc70191237)

[7 总结 14](#_Toc70191238)

[参考文献 15](#_Toc70191239)

# 1 绪论

## 1.1选题背景

帮助管理者更加轻松的掌控大局，提高员工的工作效率，为此我开发了这款名为世纪餐厅的管理系统。

本系统基于Spring Boot框架开发，java语言编写，使得该应用程序能在个平台上运行，且Spring Boot框架已集成Tomcat服务器，能做到安装即可用，且操作简单，大大降低管理者的学习成本。

Spring Boot框架是目前java开发的主流框架，其优点在于省去了繁杂的配置文件，内置servlet容器，只需java环境就能运行，便于客户使用的同时也使代码维护变得简单。、

Web页面采用了HTML5技术，使用了thymeleaf来进行前端的数据展示，ajax、jQuery来进行与后台的交互,动态的展示数据库内容。

数据库方面采用流行的MySQL数据库，InnoDB存储引擎，对数据进行一个较为安全的管理，同时MySQL具有体积小用量大等优势。

## 1.2国内外研究现状

### 1.2.1国内研究现状

中国是举世闻名的美食大国，拥有五千年的饮食文化和巨大的餐饮市场，随着人民生活水平和生活方式的转变，餐饮业具有巨大的投资市场，被称为中国的黄金产业，但同样也应看到，餐饮业不仅面临着巨大的发展机遇，也面临着前所未有的挑战和考验。这些挑战主要来源于以下几方面 :

（1）人才的专业化程度不够导致内功不足:因餐饮业门坎较低，中国的大多数餐饮企业的老板是从小店发展起来的，家族式管理的居多，还没有发展到采用智能化管理系统，许多还是"人治”，并没有一套智能化的人员管理系统，所以从观念意识、经营思想和管理水平还有待专业化。

（2）应变能力差，缺乏先进的信息工具:现在的餐饮市场火爆，许多以前做电子、房地产等其他行业的老板都凭借雄厚的资金实力挤进餐饮市场，争先恐后的上规模、上档次、 比菜品、比服务、拼价格，使餐饮市场竞争激烈，但是许多餐饮企业缺乏对市场的应变能力和灵敏的信息工具，在现今网络经济的时代，许多餐饮企业还处在手工及半手工状态，即使有计算机也只当个点菜器和计算器用，并没有真正通过计算机系统来实现改造流程、强化管理、降低成本、堵漏节流等作用。缺乏科学和标准的管理体系:国外著名的快餐连锁经过上百年的探索都形成了标准化的工作流程和方法。中餐因其菜品的多样化和特色化的服务很难实现标准化管理，这使中餐企业的成本控制很难实现，但近两年也出现了引进快餐式经营特点的中餐企业，例如全聚德集团和宁波的向阳渔港集团，从流程、服务、出品都开了中餐标准化的先河。

### 1.2.2 国外研究现状

国外著名的快餐连锁经过上百年的探索都形成了标准化的工作流程和方法。如KFC.必胜客等，从原材料采购、点菜、收银、财务、成本都已形成了一定的标准化模式，信息化。高科技程度高，能对菜品的选择进行科学的调控，剔除销量不好的产品。对热销产品进行总结，继续开发新的产品，给决策机构提供了很多素材，对制定相应的政策提供了帮助。高度系统化的人员管理，每个人的分工明确，使管理人员能及时下打任务指令到每个岗位，极大提高管理效率，减少经营成本。

## 1.3设计内容和意义

### 1.3.1 设计内容

本系统主要是为了帮助餐饮行业更好的进行人员管理而设计的一套管理系统，其主要功能侧重于人员管理，包括但不限于，员工信息管理（新增、修改、删除），餐厅的营收记录，菜品信息管理（新增、修改、删除）等。

### 13.2 意义

本系统致力于帮助餐饮行业进行更好的人员管理，减少管理成本的同时，能更加有效率的经营餐厅，消息也能及时传递到每个员工的手上，让管理更加轻松。

## 1.4本文结构

世纪餐厅管理系统设计说明书，针对本管理系统的设计思维及实现方法作出了一个系统的介绍，整体结构如下：

第一章为绪论，主要介绍本人设计本系统的前因后果，以及用到的技术

第二章为开发工具及相关技术简介：介绍完成本系统所用到的技术以及开发工具。

第三章为系统分析：分析本系统的可行性以及模块功能的确定。

第四章为系统设计：设计本系统的流程图及数据库。

第五章为系统实现：介绍本系统的功能模块。

第六章为总结：总结本人在这次项目中遇到的困难及收获。

# 2开发工具及相关技术简介

本系统所用到的开发工具有intelliJ IDEA-2018, jdk-1.8, Mysql-5.6, navcat premium, Visual Studio Code, Git，使用到的技术有java开发语言，Spring Boot框架，bootstrap框架。

## 2.1 intelliJ IDEA

IDEA 全称 IntelliJ IDEA，是java编程语言开发的集成环境。IntelliJ在业界被公认为最好的java开发工具，尤其在智能代码助手、代码自动提示、重构、JavaEE支持、各类版本工具(git、svn等)、JUnit、CVS整合、代码分析、 创新的GUI设计等方面的功能可以说是超常的

## 2.2 JDK1.8

JDK是 Java 语言的软件开发工具包，主要用于移动设备、嵌入式设备上的java应用程序。JDK是整个java开发的核心，它包含了JAVA的运行环境（JVM+Java系统类库）和JAVA工具，而JDK1.8则是目前使用较为广泛的一个版本。

## 2.3 MySQL

MySQL是一个关系型数据库管理系统， 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。

## 2.4 Navcat premium

Navicat premium是一款数据库管理工具,是一个可多重连线资料库的管理工具，它可以让你以单一程式同时连线到 MySQL、SQLite、Oracle、MariaDB、Mssql、及PostgreSQL 资料库，让管理不同类型的资料库更加的方便

## 2.5 Visual Studio Code

Visual Studio Code是Microsoft在2015年4月30日Build开发者大会上正式宣布一个运行于 Mac OS X、Windows和 Linux 之上的，针对于编写现代Web和云应用的跨平台源代码编辑器，可在桌面上运行，并且可用于Windows，macOS和Linux。它具有对JavaScript，TypeScript和Node.js的内置支持，并具有丰富的其他语言（例如C++，C＃，Java，Python，PHP，Go）和运行时（例如.NET和Unity）扩展的生态系统。

## 2.6 Git

Git是一个开源的分布式版本控制系统，可以有效、高速地处理从很小到非常大的项目版本管理。

Git 是 Linus Torvalds 为了帮助管理 Linux 内核开发而开发的一个开放源码的版本控制软件。

## 2.7 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。Java语言作为静态面向对象编程语言的代表，极好地实现了面向对象理论，允许程序员以优雅的思维方式进行复杂的编程。

Java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

## 2.8 Spring Boot

Spring Boot基于Spring4.0设计，不仅继承了Spring框架原有的优秀特性，而且还通过简化配置来进一步简化了Spring应用的整个搭建和开发过程,而Spring框架是Java平台上的一种开源应用框架，提供具有控制反转特性的容器。尽管Spring框架自身对编程模型没有限制，但其在Java应用中的频繁使用让它备受青睐，以至于后来让它作为EJB（EnterpriseJavaBeans）模型的补充，甚至是替补。

## 2.9 Bootstrap

Bootstrap是美国Twitter公司的设计师Mark Otto和Jacob Thornton合作基于HTML、CSS、JavaScript 开发的简洁、直观、强悍的前端开发框架，使得 Web 开发更加快捷。Bootstrap提供了优雅的HTML和CSS规范，它即是由动态CSS语言Less写成。Bootstrap一经推出后颇受欢迎，一直是GitHub上的热门开源项目，包括NASA的MSNBC（微软全国广播公司）的Breaking News都使用了该项目。国内一些移动开发者较为熟悉的框架，如WeX5前端开源框架等，也是基于Bootstrap源码进行性能优化而来

# 3系统分析

## 3.1 需求分析

（1）做为一个人员管理系统，最主要的功能则是员工信息的增删改查，要求实现对员工的基本信息如：性别、年龄、姓名、岗位、电话号码等做些简单的操作。

（2）要求显示餐厅的年、月、日收入与支出。

（3）要求能显示部门信息，能新增、删除、修改部门。

（4）要求能具有商品信息管理、商品信息查询、库存查询，一遍采购部门及时补充商品。

（5）要求有登录界面，但不开放注册功能，员工账号由上司开通，老板账号由开发者开通。

（6）要求能实时监控在线人数。

（7）要求各级领导的权限各不相同，权限与级别挂钩。

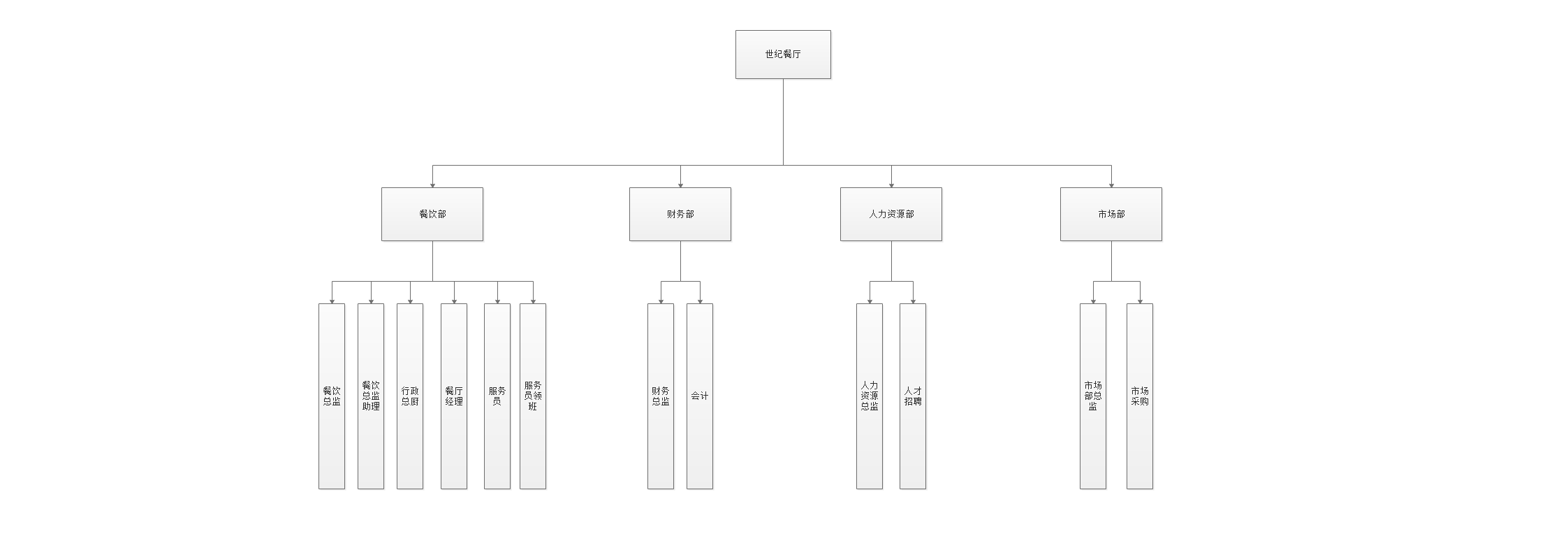
（8）要求上级能对下级的权限进行修改。

## 3.2组织结构分析

注：目前系统自定义的有一套组织结构模板，客户也可自主定义

本餐厅组织结构为：餐饮部、财务部、人力资源部、市场部

其结构图为：



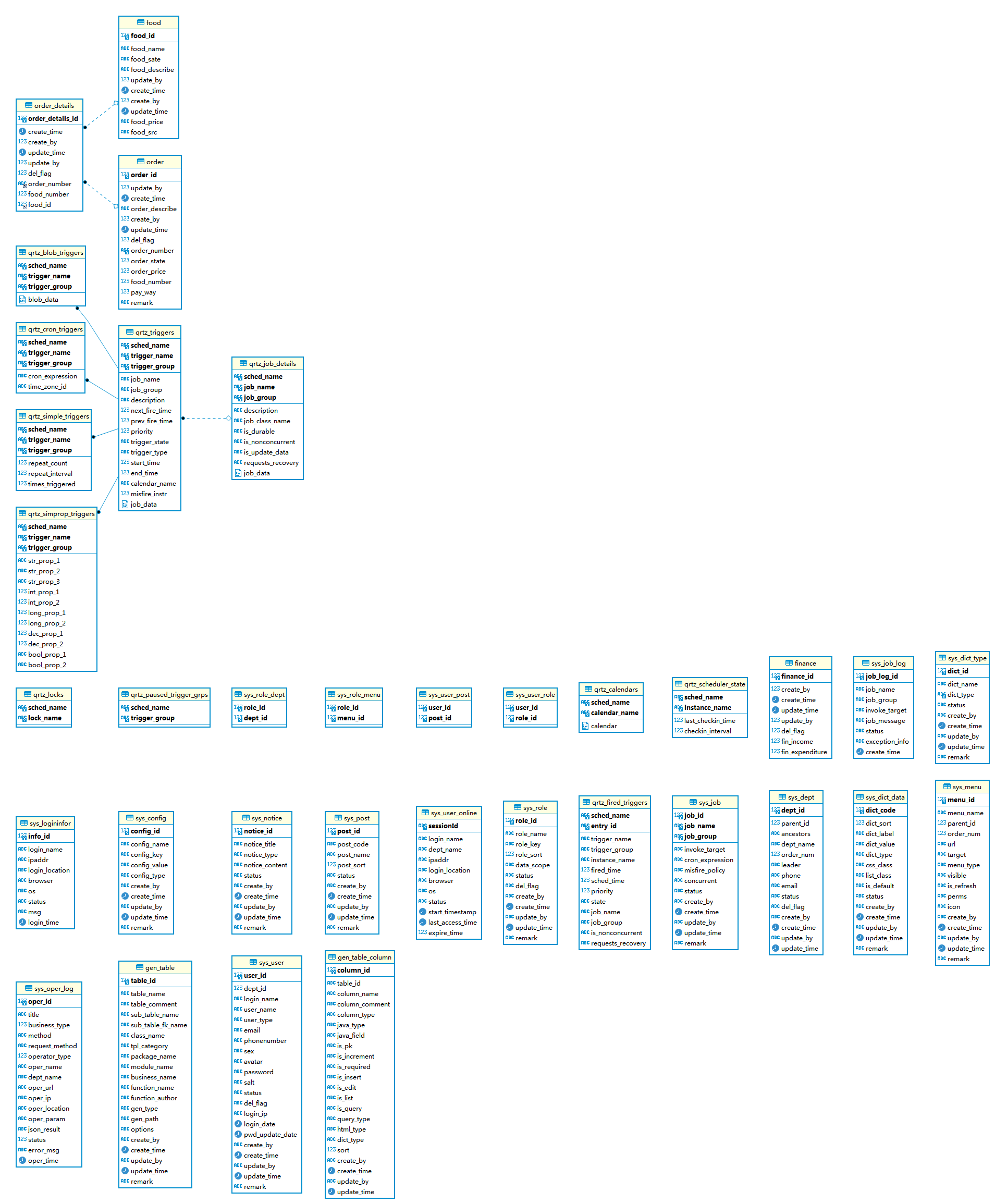
# 4 系统设计

## 4.1功能模块设计

根据需求分析，本系统设计了如下几个功能模块：

1. 登录功能模块，能让客户登录系统界面，出来输入账号密码以外还要求客户输入验证码做人工判断，并且记录用户的登陆时间，长时间不进行操作会提示客户重新登录
2. 查询管理模块，对数据库的数据进行操作的功能模块，对餐厅信息进行增删改查等
3. 其他功能模块，为方便以后添加功能而存在的模块，目前包含对年月日流水的记录、记录在线人数等

## 4.2数据库设计



4.2-1数据库er图

图4.2-1为本系统所用的数据库er图

## 4.3用户界面设计

### 4.3.1 登录界面

### 4.3.2 系统主界面

### 4.3.3 员工管理界面

### 4.3.4 部门管理界面

### 4.3.5 商品管理界面

### 4.3.6客户个人界面

## 4.4 本章小结

本系统主要着重设计了对人员管理方面的功能，也注重页面的美观每个页面的样式都力求简洁漂亮，同时也让各功能之间的切换比较流畅，视图采用标签页的形式展现，可以一键关闭或者关闭部分标签页，而且标签之间的切换并不会刷新整个页面，对各个浏览器也进行了兼容，并且能全屏显示

# 5系统实现

## 5.1 登录验证功能模块

当客户在网页上输入账号、密码、验证码后，前端通过from表单提交将数据传输给后端，后端通过springboot框架将自动识别前端传来的Json数据中的账号、密码与验证码，首先先检验验证码是否正确,使用shiro实现验证码的生成与检验，错误则直接返回错误信息，若正确，则将账号、密码与数据库的员工表进行比对，如果账号密码有不存在、账号或密码错误、账号已被删除、则返回提示信息给前端，前端接收到后展示给客户,密码错误5次锁定账号10分钟，若账号与密码正确则根据用户的权限等级跳转到对应的页面

页面跳转方面使用的则是thymeleaf框架，通过Springboot解析前端传来的url，得到页面的名称，thymeleaf将自动拼接为xx.html，并在系统静态资源目录下找到该页面展示出来

实现该功能所用到的配置文件与代码：

SysLoginController.java:

**package** com.ruoyi.web.controller.system;  
  
**import** javax.servlet.http.HttpServletRequest;  
**import** javax.servlet.http.HttpServletResponse;  
**import** org.apache.shiro.SecurityUtils;  
**import** org.apache.shiro.authc.AuthenticationException;  
**import** org.apache.shiro.authc.UsernamePasswordToken;  
**import** org.apache.shiro.subject.Subject;  
**import** org.springframework.stereotype.Controller;  
**import** org.springframework.web.bind.annotation.GetMapping;  
**import** org.springframework.web.bind.annotation.PostMapping;  
**import** org.springframework.web.bind.annotation.ResponseBody;  
**import** com.ruoyi.common.core.controller.BaseController;  
**import** com.ruoyi.common.core.domain.AjaxResult;  
**import** com.ruoyi.common.utils.ServletUtils;  
**import** com.ruoyi.common.utils.StringUtils;  
  
*/\*\*  
 \* 登录验证  
 \*   
 \** ***@author*** *ruoyi  
 \*/*@Controller  
**public class** SysLoginController **extends** BaseController  
{  
 @GetMapping(**"/login"**)  
 **public** String login(HttpServletRequest request, HttpServletResponse response)  
 {  
 *// 如果是Ajax请求，返回Json字符串。* **if** (ServletUtils.*isAjaxRequest*(request))  
 {  
 **return** ServletUtils.*renderString*(response, **"{\"code\":\"1\",\"msg\":\"未登录或登录超时。请重新登录\"}"**);  
 }  
  
 **return "login"**;  
 }  
  
 @PostMapping(**"/login"**)  
 @ResponseBody  
 **public** AjaxResult ajaxLogin(String username, String password, Boolean rememberMe)  
 {  
 UsernamePasswordToken token = **new** UsernamePasswordToken(username, password, rememberMe);  
 Subject subject = SecurityUtils.*getSubject*();  
 **try** {  
 subject.login(token);  
 **return** success();  
 }  
 **catch** (AuthenticationException e)  
 {  
 String msg = **"用户或密码错误"**;  
 **if** (StringUtils.*isNotEmpty*(e.getMessage()))  
 {  
 msg = e.getMessage();  
 }  
 **return** error(msg);  
 }  
 }  
  
 @GetMapping(**"/unauth"**)  
 **public** String unauth()  
 {  
 **return "error/unauth"**;  
 }  
}

## 5.2 查询管理模块

该模块功能主要为客户对数据库的操作，后台连结数据库方面采用JDBC驱动，搭配Mybaits的逆向工程插件使用能自动根据数据的表格生存对应的实体类、mapper接口和mapper.xml文件。前端发送请求后，后台的controller层接收请求，根据请求的类型调用service层的方法，service处理好请求数据后，调用对应的mapper接口，通过mybatis辅助进对数据的增删改查。

实现该功能所需要的部分代码：

SysUserController.java

@Log(title = **"用户管理"**, businessType = BusinessType.***EXPORT***)  
@RequiresPermissions(**"system:user:export"**)  
@PostMapping(**"/export"**)  
@ResponseBody  
**public** AjaxResult export(SysUser user)  
{  
 List<SysUser> list = **userService**.selectUserList(user);  
 ExcelUtil<SysUser> util = **new** ExcelUtil<SysUser>(SysUser.**class**);  
 **return** util.exportExcel(list, **"用户数据"**);  
}  
  
@Log(title = **"用户管理"**, businessType = BusinessType.***IMPORT***)  
@RequiresPermissions(**"system:user:import"**)  
@PostMapping(**"/importData"**)  
@ResponseBody  
**public** AjaxResult importData(MultipartFile file, **boolean** updateSupport) **throws** Exception  
{  
 ExcelUtil<SysUser> util = **new** ExcelUtil<SysUser>(SysUser.**class**);  
 List<SysUser> userList = util.importExcel(file.getInputStream());  
 String operName = ShiroUtils.*getSysUser*().getLoginName();  
 String message = **userService**.importUser(userList, updateSupport, operName);  
 **return** AjaxResult.*success*(message);  
}  
  
@RequiresPermissions(**"system:user:view"**)  
@GetMapping(**"/importTemplate"**)  
@ResponseBody  
**public** AjaxResult importTemplate()  
{  
 ExcelUtil<SysUser> util = **new** ExcelUtil<SysUser>(SysUser.**class**);  
 **return** util.importTemplateExcel(**"用户数据"**);  
}  
  
*/\*\*  
 \* 新增用户  
 \*/*@GetMapping(**"/add"**)  
**public** String add(ModelMap mmap)  
{  
 mmap.put(**"roles"**, **roleService**.selectRoleAll().stream().filter(r -> !r.isAdmin()).collect(Collectors.*toList*()));  
 mmap.put(**"posts"**, **postService**.selectPostAll());  
 **return prefix** + **"/add"**;  
}  
  
*/\*\*  
 \* 新增保存用户  
 \*/*@RequiresPermissions(**"system:user:add"**)  
@Log(title = **"用户管理"**, businessType = BusinessType.***INSERT***)  
@PostMapping(**"/add"**)  
@ResponseBody  
**public** AjaxResult addSave(@Validated SysUser user)  
{  
 **if** (UserConstants.***USER\_NAME\_NOT\_UNIQUE***.equals(**userService**.checkLoginNameUnique(user.getLoginName())))  
 {  
 **return** error(**"新增用户'"** + user.getLoginName() + **"'失败，登录账号已存在"**);  
 }  
 **else if** (StringUtils.*isNotEmpty*(user.getPhonenumber())  
 && UserConstants.***USER\_PHONE\_NOT\_UNIQUE***.equals(**userService**.checkPhoneUnique(user)))  
 {  
 **return** error(**"新增用户'"** + user.getLoginName() + **"'失败，手机号码已存在"**);  
 }  
 **else if** (StringUtils.*isNotEmpty*(user.getEmail())  
 && UserConstants.***USER\_EMAIL\_NOT\_UNIQUE***.equals(**userService**.checkEmailUnique(user)))  
 {  
 **return** error(**"新增用户'"** + user.getLoginName() + **"'失败，邮箱账号已存在"**);  
 }  
 user.setSalt(ShiroUtils.*randomSalt*());  
 user.setPassword(**passwordService**.encryptPassword(user.getLoginName(), user.getPassword(), user.getSalt()));  
 user.setCreateBy(ShiroUtils.*getLoginName*());  
 **return** toAjax(**userService**.insertUser(user));  
}  
  
*/\*\*  
 \* 修改用户  
 \*/*@GetMapping(**"/edit/{userId}"**)  
**public** String edit(@PathVariable(**"userId"**) Long userId, ModelMap mmap)  
{  
 List<SysRole> roles = **roleService**.selectRolesByUserId(userId);  
 mmap.put(**"user"**, **userService**.selectUserById(userId));  
 mmap.put(**"roles"**, SysUser.*isAdmin*(userId) ? roles : roles.stream().filter(r -> !r.isAdmin()).collect(Collectors.*toList*()));  
 mmap.put(**"posts"**, **postService**.selectPostsByUserId(userId));  
 **return prefix** + **"/edit"**;  
}  
  
*/\*\*  
 \* 修改保存用户  
 \*/*@RequiresPermissions(**"system:user:edit"**)  
@Log(title = **"用户管理"**, businessType = BusinessType.***UPDATE***)  
@PostMapping(**"/edit"**)  
@ResponseBody  
**public** AjaxResult editSave(@Validated SysUser user)  
{  
 **userService**.checkUserAllowed(user);  
 **if** (StringUtils.*isNotEmpty*(user.getPhonenumber())  
 && UserConstants.***USER\_PHONE\_NOT\_UNIQUE***.equals(**userService**.checkPhoneUnique(user)))  
 {  
 **return** error(**"修改用户'"** + user.getLoginName() + **"'失败，手机号码已存在"**);  
 }  
 **else if** (StringUtils.*isNotEmpty*(user.getEmail())  
 && UserConstants.***USER\_EMAIL\_NOT\_UNIQUE***.equals(**userService**.checkEmailUnique(user)))  
 {  
 **return** error(**"修改用户'"** + user.getLoginName() + **"'失败，邮箱账号已存在"**);  
 }  
 user.setUpdateBy(ShiroUtils.*getLoginName*());  
 **return** toAjax(**userService**.updateUser(user));  
}

Mybaits-config.xml

*<?***xml version="1.0" encoding="UTF-8"** *?>***<!DOCTYPE configuration  
PUBLIC "-//mybatis.org//DTD Config 3.0//EN"  
"http://mybatis.org/dtd/mybatis-3-config.dtd"*>***<**configuration**>  
   
 <**settings**>  
 <**setting name="cacheEnabled" value="true"** /> <**setting name="useGeneratedKeys" value="true"** /> <**setting name="defaultExecutorType" value="REUSE"** /> <**setting name="logImpl" value="SLF4J"** /> </**settings**>  
   
</**configuration**>

ISysUserService,java

*/\*\*  
 \* 根据条件分页查询用户列表  
 \*   
 \** ***@param user*** *用户信息  
 \** ***@return*** *用户信息集合信息  
 \*/***public** List<SysUser> selectUserList(SysUser user);  
  
*/\*\*  
 \* 根据条件分页查询已分配用户角色列表  
 \*   
 \** ***@param user*** *用户信息  
 \** ***@return*** *用户信息集合信息  
 \*/***public** List<SysUser> selectAllocatedList(SysUser user);  
  
*/\*\*  
 \* 根据条件分页查询未分配用户角色列表  
 \*   
 \** ***@param user*** *用户信息  
 \** ***@return*** *用户信息集合信息  
 \*/***public** List<SysUser> selectUnallocatedList(SysUser user);  
  
*/\*\*  
 \* 通过用户名查询用户  
 \*   
 \** ***@param userName*** *用户名  
 \** ***@return*** *用户对象信息  
 \*/***public** SysUser selectUserByLoginName(String userName);  
  
*/\*\*  
 \* 通过手机号码查询用户  
 \*   
 \** ***@param phoneNumber*** *手机号码  
 \** ***@return*** *用户对象信息  
 \*/***public** SysUser selectUserByPhoneNumber(String phoneNumber);  
  
*/\*\*  
 \* 通过邮箱查询用户  
 \*   
 \** ***@param email*** *邮箱  
 \** ***@return*** *用户对象信息  
 \*/***public** SysUser selectUserByEmail(String email);  
  
*/\*\*  
 \* 通过用户ID查询用户  
 \*   
 \** ***@param userId*** *用户ID  
 \** ***@return*** *用户对象信息  
 \*/***public** SysUser selectUserById(Long userId);  
  
*/\*\*  
 \* 通过用户ID查询用户和角色关联  
 \*   
 \** ***@param userId*** *用户ID  
 \** ***@return*** *用户和角色关联列表  
 \*/***public** List<SysUserRole> selectUserRoleByUserId(Long userId);  
  
*/\*\*  
 \* 通过用户ID删除用户  
 \*   
 \** ***@param userId*** *用户ID  
 \** ***@return*** *结果  
 \*/***public int** deleteUserById(Long userId);  
  
*/\*\*  
 \* 批量删除用户信息  
 \*   
 \** ***@param ids*** *需要删除的数据ID  
 \** ***@return*** *结果  
 \** ***@throws*** *Exception 异常  
 \*/***public int** deleteUserByIds(String ids);  
  
*/\*\*  
 \* 保存用户信息  
 \*   
 \** ***@param user*** *用户信息  
 \** ***@return*** *结果  
 \*/***public int** insertUser(SysUser user);  
  
*/\*\*  
 \* 注册用户信息  
 \*   
 \** ***@param user*** *用户信息  
 \** ***@return*** *结果  
 \*/***public boolean** registerUser(SysUser user);  
  
*/\*\*  
 \* 保存用户信息  
 \*   
 \** ***@param user*** *用户信息  
 \** ***@return*** *结果  
 \*/***public int** updateUser(SysUser user);  
  
*/\*\*  
 \* 修改用户详细信息  
 \*   
 \** ***@param user*** *用户信息  
 \** ***@return*** *结果  
 \*/***public int** updateUserInfo(SysUser user);

## 5.3 其他功能模块

针对用户个人界面本系统实现了用户自传头像的功能，代码如下：

*/\*\*  
 \* 通用文件上传请求  
 \*/*@PostMapping(**"/common/upload"**)  
@ResponseBody  
**public** AjaxResult uploadFile(MultipartFile file) **throws** Exception  
{  
 **try** {  
 *// 上传文件路径* String filePath = RuoYiConfig.*getUploadPath*();  
 *// 上传并返回新文件名称* String fileName = FileUploadUtils.*upload*(filePath, file);  
 String url = **serverConfig**.getUrl() + fileName;  
 AjaxResult ajax = AjaxResult.*success*();  
 ajax.put(**"fileName"**, fileName);  
 ajax.put(**"url"**, url);  
 **return** ajax;  
 }  
 **catch** (Exception e)  
 {  
 **return** AjaxResult.*error*(e.getMessage());  
 }  
}

查看流水方面最主要的是对前端页面的操作，根据点击年月日来显示对应的流水，也进行了简单的操作显示，代码如下：

**$**(**"#btnDay"**).click(**function** () {  
 **$**(**"#btnMonth"**).removeClass();  
 **$**(**"#btnMonth"**).addClass(**"btn btn-xs btn-white"**);  
 **$**(**"#btnYear"**).removeClass();  
 **$**(**"#btnYear"**).addClass(**"btn btn-xs btn-white"**);  
  
 **$**(**"#btnDay"**).removeClass();  
 **$**(**"#btnDay"**).addClass(**"btn btn-xs btn-white active"**);  
  
 **$**.ajax({  
 **url**: **"order/OrderStatistics?flag=0"**,  
 **type**: **"post"**,  
 success: **function** (result) {  
 **$**(**"#orders"**).text(**"日订单数"**);  
 **$**(**"#sale"**).text(**"日销售额"**);  
 **$**(**"#orderStatistics"**).text(result.orderPrices == **null** ? 0 : result.**orderPrices** )  
 **$**(**"#ordersOfDate"**).text(result.orders == **""** ? 0 : result.orders )  
  
 }  
  
 }  
 )  
  
})  
  
**$**(**"#btnMonth"**).click(**function** () {  
 **$**(**"#btnDay"**).removeClass();  
 **$**(**"#btnDay"**).addClass(**"btn btn-xs btn-white"**);  
 **$**(**"#btnYear"**).removeClass();  
 **$**(**"#btnYear"**).addClass(**"btn btn-xs btn-white"**);  
  
 **$**(**"#btnMonth"**).removeClass();  
 **$**(**"#btnMonth"**).addClass(**"btn btn-xs btn-white active"**);  
  
 **$**.ajax({  
 **url**: **"order/OrderStatistics?flag=1"**,  
 **type**: **"post"**,  
 success: **function** (result) {  
 **$**(**"#orders"**).text(**"月订单数"**);  
 **$**(**"#sale"**).text(**"月销售额"**);  
 **$**(**"#orderStatistics"**).text(result.orderPrices == **null** ? 0 : result.**orderPrices** )  
 **$**(**"#ordersOfDate"**).text(result.orders == **""** ? 0 : result.orders )  
  
 }  
  
 }  
 )  
  
})  
  
**$**(**"#btnYear"**).click(**function** () {  
 **$**(**"#btnMonth"**).removeClass();  
 **$**(**"#btnMonth"**).addClass(**"btn btn-xs btn-white"**);  
 **$**(**"#btnDay"**).removeClass();  
 **$**(**"#btnDay"**).addClass(**"btn btn-xs btn-white"**);  
  
 **$**(**"#btnYear"**).removeClass();  
 **$**(**"#btnYear"**).addClass(**"btn btn-xs btn-white active"**);  
  
 **$**.ajax({  
 **url**: **"order/OrderStatistics?flag=2"**,  
 **type**: **"post"**,  
 success: **function** (result) {  
 **$**(**"#orders"**).text(**"年订单数"**);  
 **$**(**"#sale"**).text(**"年销售额"**);  
 **$**(**"#orderStatistics"**).text(result.orderPrices == **null** ? 0 : result.**orderPrices** )  
 **$**(**"#ordersOfDate"**).text(result.orders == **""** ? 0 : result.orders )  
  
 }  
  
 }  
 )  
  
})

## 5.4本章小结

登录功能模块实现了用户的登录与权限验证，查询管理模块实现了对数据库的增删改查操作，其他功能模块实现的有头像上传，流水记录，

# 7 总结

本系统是针对现在社会中小型餐饮行业落后人工管理现象设计的，该系统主要完成了对商品信息，员工信息，会员信息还有财务信息的数字化，让管理者更加得心应手的管理自己的企业。

由于本人第一次独立分析与设计一个完整的信息系统，在经验和技术方面都存在着很多不足之处，另外，由于时间比较仓促，对于这次的分析和设计也很不完善。

在分析设计过程中，主要是在分析系统上耗费了大量的时间，对很多组织结构和业务流程分析得不够彻底，也忽略了很多细节性问题。

同时,我也在生活中很关注这种软件的使用情况,它的发展可以更加人性化，以目前的科技，完全可以省去在前台点菜的步骤，客人进入餐厅，完全可以通过平板电脑进行在线点菜，厨房也可以在第一时间为客人准备，这样的处理方式现在很多大型的餐饮企业已经实现，但是我个人认为可以推广到中小型企业，一旦这个得到实现，加上这样的一个系统，完全可以在餐饮行业的竞争中取得很大的优势。

# 参考文献

务部、行政部、餐饮部、