****

**2021届毕业设计说明书**

# 世纪餐厅管理系统的设计与实现

院 、 部： 计算机与信息科学学院

学生姓名： 石玮

指导教师： 徐艳山 职称 讲师

专 业： 信息与计算科学

班 级： 信息1702

完成时间： 2021-5

第一指导教师：徐艳山

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，使用的是分布式开发模式，各模块之间功能是相互独立互不影响的，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](#_Toc15685)

[1 绪论 1](#_Toc22260)

[1.1选题背景 1](#_Toc10734)

[1.2国内外研究现状 1](#_Toc2782)

[1.2.1国内研究现状 1](#_Toc27811)

[1.2.2 国外研究现状 2](#_Toc17161)

[1.3设计内容和意义 2](#_Toc28776)

[1.3.1 设计内容 2](#_Toc11144)

[13.2 意义 2](#_Toc29086)

[1.4本文结构 3](#_Toc13677)

[2开发工具及相关技术简介 4](#_Toc29304)

[2.1 intelliJ IDEA 4](#_Toc482)

[2.2 JDK1.8 4](#_Toc8244)

[2.3 MySQL 4](#_Toc16123)

[2.4 Navcat premium 4](#_Toc5921)

[2.5 Visual Studio Code 4](#_Toc19504)

[2.6 Git 5](#_Toc9134)

[2.7 java 5](#_Toc27558)

[2.8 Spring Boot 5](#_Toc20759)

[2.9 Bootstrap 5](#_Toc11944)

[2.10 SSM架构 5](#_Toc23157)

[3系统分析 7](#_Toc32514)

[3.1 需求分析 7](#_Toc31024)

[3.2组织结构分析 7](#_Toc29723)

[3.3执行流程分析 8](#_Toc9255)

[4 系统设计 9](#_Toc23046)

[4.1功能模块设计 9](#_Toc11541)

[4.2数据库设计 10](#_Toc16856)

[4.3用户界面设计 14](#_Toc21994)

[4.3.1 登录界面 14](#_Toc28176)

[4.3.2 系统主界面 14](#_Toc1597)

[4.3.3 员工管理界面 15](#_Toc17102)

[4.3.4 部门管理界面 16](#_Toc18689)

[4.3.5 商品管理界面 16](#_Toc24873)

[4.3.6客户个人界面 16](#_Toc15644)

[4.4 本章小结 17](#_Toc3568)

[5系统实现 18](#_Toc14237)

[5.1 登录验证功能模块 18](#_Toc18314)

[5.2 查询管理模块 19](#_Toc3223)

[5.3 其他功能模块 24](#_Toc19422)

[5.4本章小结 26](#_Toc23541)

[6 系统测试 27](#_Toc5728)

[6.1测试目的 27](#_Toc27068)

[6.2测试方法 27](#_Toc24513)

[6.3测试用例 27](#_Toc1432)

[7 总结 28](#_Toc13261)

[参考文献 29](#_Toc25520)

# 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各个平台上运行，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

中国虽然是举世闻名的传统美食强国,拥有五千年的传统饮食文化和巨大的餐饮消费市场,随着中国人民的生活物质水平和其生活方式的巨大转变,餐饮行业已经具有巨大的创新性和投资市场,被誉为当代中国的一个重要黄金产业,但同样我们也应该清醒地看到,餐饮行业不仅在发展中面临着巨大的市场机遇,也正在面临着前无法预期的所未来挑战与考验。这些挑战主要来源于以下几方面 :

（1）人才的专业化程度不足导致管理困难:因餐饮业门坎较低，中国的大多数餐饮企业的老板都是白手起家，从小型餐馆做起，习惯了人工管理的居多，以致于当餐厅日益壮大后，出现管理困难的局面，并没有设计一套智能化的人员管理系统，所以我国的餐饮行业的管理方式无论是从观念意识、经营思想还是管理水平都有待专业化、系统化。

（2）信息传递效率差，信息获取滞后,许多中小型餐饮行业没有一套完善的、智能化的信息系统，导致信息获取滞后没有及时应对市场冲击带来的风险。人员的管理也没有一套系统的模式，导致出现事故时缺少应对方案；没有合理安排员工工作，员工之间的工作时间比例不平衡

### 1.2.2 国外研究现状

国外由于科技发展迅速，管理意识超前，加上起步早，已经形成了标准化、系统化的工作流程和人员管理。如KFC.必胜客等，从原材料采购、收银、点菜、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

第二章为本系统所用到的开发工具及相关技术的简单介绍，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：介绍本系统的各个功能模块具体是怎样实现的。

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，是一款java编程语言开发的应用软件。IDEA是业界被公认为最好的java开发工具之一，以其酷炫的ui，智能的代码提示、简洁的系统设置、强大的插件系统、JavaEE支持、各类版本控制工具(github、svn等)、CVS整合、代码自动排版等功能赢得了广大程序员的喜爱，相比于其他工具，IDEA运行代码的速度更快，debug更为方便

## 2.2 JDK1.8

JDK是 Java 语言开发所的软件开发工具包，里面包含有jre-Java Runtime Environment，简称JRE，是java代码运行是所需的环境，里面又包含有JVM-Java Virtual Machine，java虚拟机，因为它的存在，java应用程序才能在各个品台上得以运行

2.3 MySQL

MySQL是一个关系型数据库管理系统， 是目前比较流行的关系型数据库管理系统之一，工作方式为单进程多线程，存储引擎为innoDB，内存中存在缓存池，采用插入缓冲策略使得处理数据的速度很快

## 2.4 Navcat premium

Navicat premium是一款数据库管理工具,其简结的ui设计深得大众喜爱，拥有独特的视觉化sql语句创建，根据表格能自动生成对应的sql语句以及er图，非常便捷。

## 2.5 Visual Studio Code

Visual Studio Code是继notepad++后最为强大的一款跨平台源代码编辑器，不仅本身支持多种语言的代码提示，还能通过自身的插件功能来实习对其他冷门语言的支持，除此之外还能运行node，微信小程序等文件。

## 2.6 Git

Git是目前最为先进的分布式版本控制系统，并且是开源的。他让每台电脑上都有自己的版本库，这样在不联网的情况下也能对代码进行版本控制，多人协作时，只需把各自的修改推送给对方，就可以互相看到对方的修改了

## 2.7 java

Java是一种可以撰写跨平台应用程序的、面向对象的程序设计语言，其发展历史已经有26年了，由于其跨平台的特性使得它在开发web应用方面有着天然的优势。

## 2.8 Spring Boot

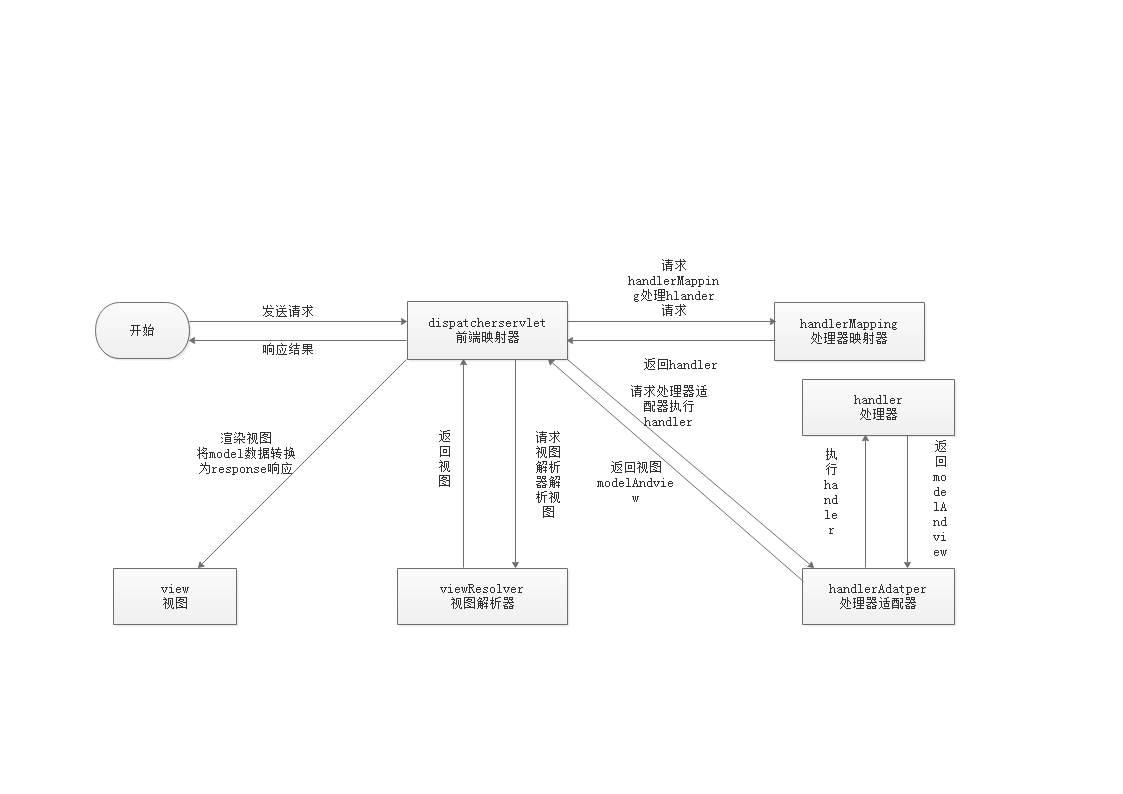
Spring Boot基于Spring4.0设计，省去Spring繁琐的参数配置，做到了到手即用的特点，内置Tomcat服务器，只需启动application应用类就能启动服务，帮助开发人员快速开发、快速整合、配置简化。并提供了一系列大型项目通用的非业务性功能，例如：内嵌服务器、安全管理、运行数据监控、运行状况检查和外部化配置等。

## 2.9 Bootstrap

Bootstrap是一款基于HTML、css、JavaScript的前端开发框架，自定义了许多的css样式和js插件，开发人员能直接使用这些插件，且实现了响应式布局，同一套页面能兼容不同的分辨率设备。

## 2.10 SSM架构

SSM框架整合了spring MVC ，spring和mybatis三大框架，采用标准的Model 、View、Controller三层架构模式，使用spring MVC来完成请求的转发和视图管理，spring实现业务对象管理，mybatis作为数据对象的持久化引擎，具体流程如下：



# 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：

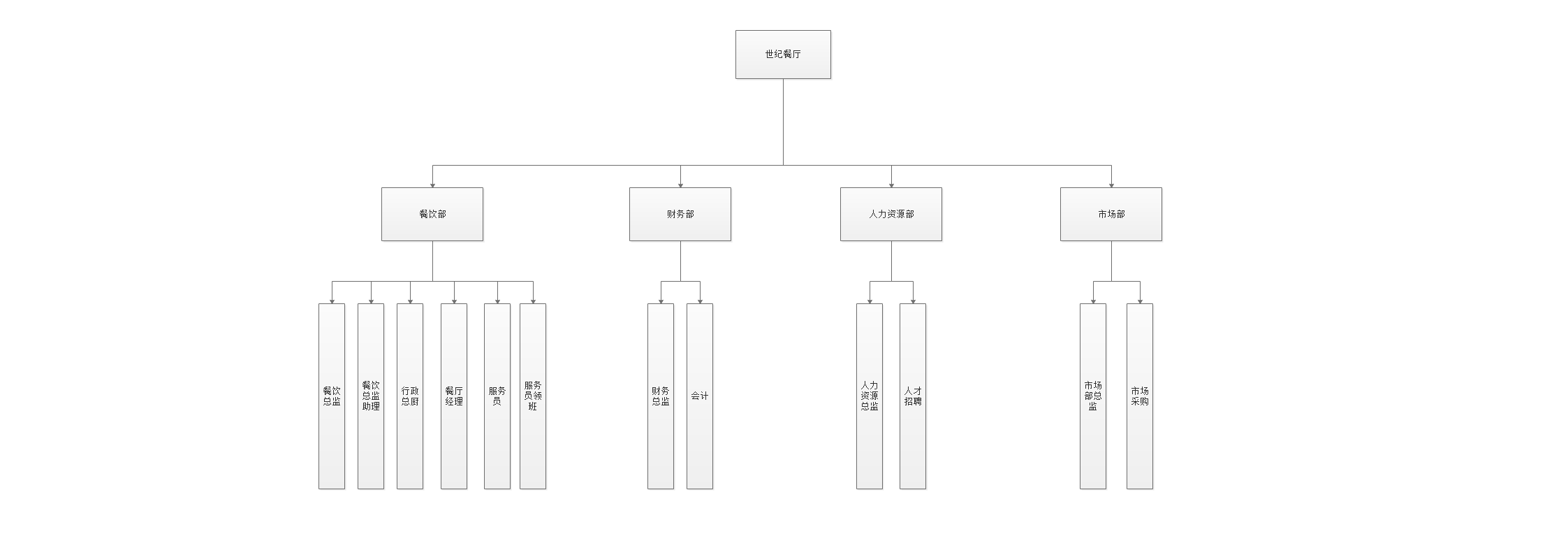


图 3.2-1

## 3.3执行流程分析

针对传统管理系统的流程分析，本人设计了如下的流程图：

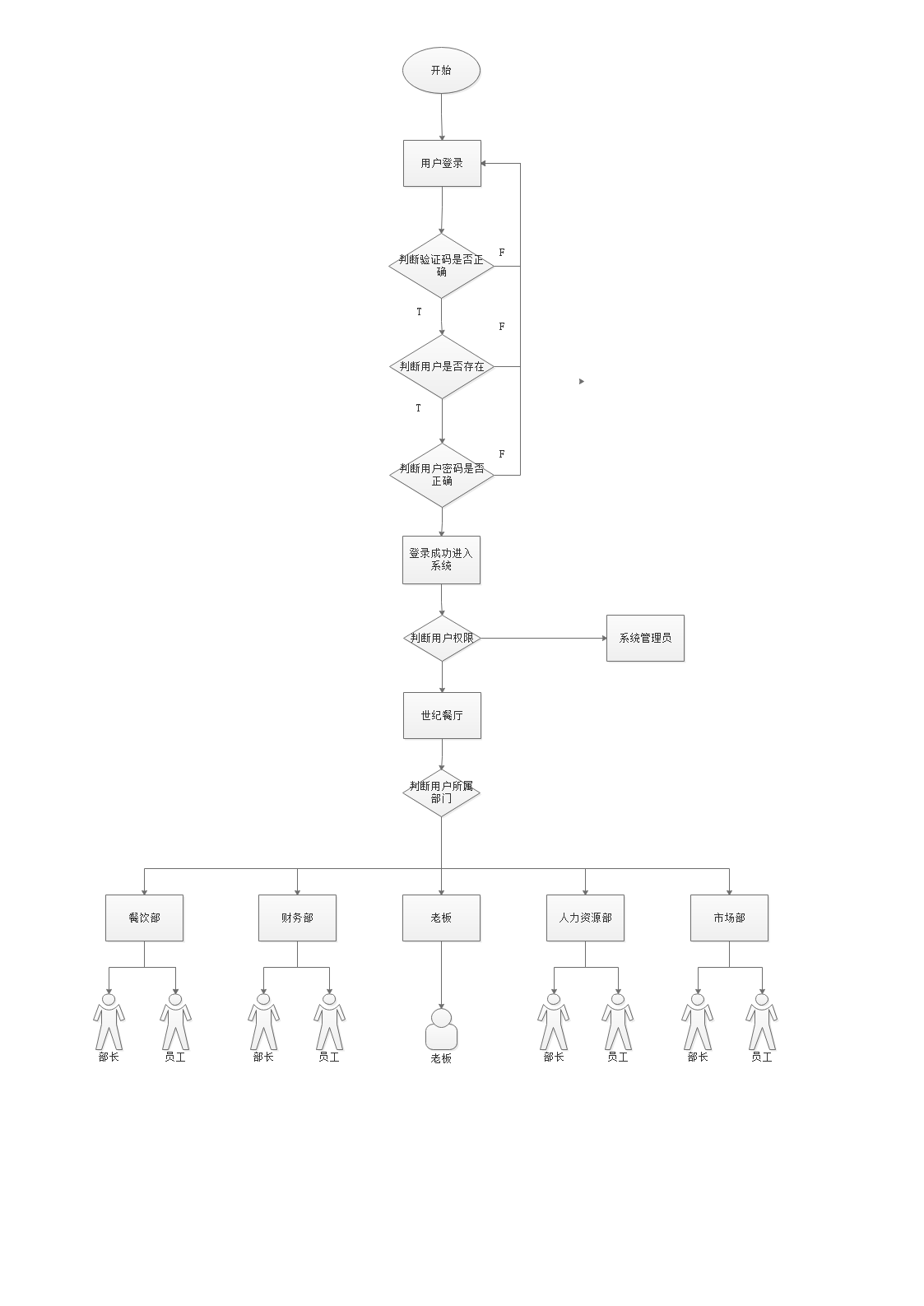


图 3.3-1

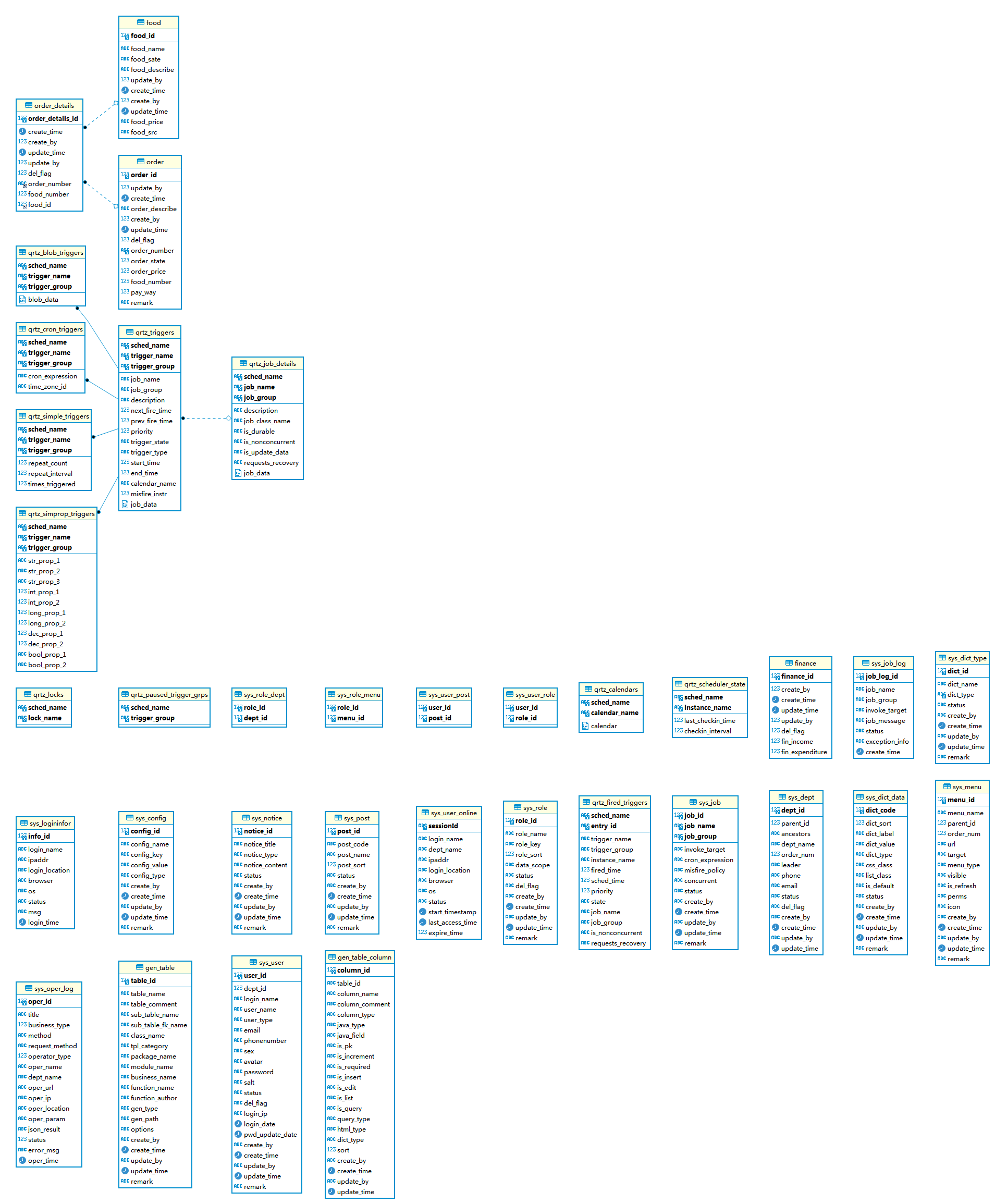
# 4 系统设计

## 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

以下为本系统所使用的主要表的字段设计

表4.2-1各表格公用字段

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 数据类型 | 是否为空 | 默认值 |  |
| Create\_time | datetime | false |  | 创建时间 |
| Create\_by | Interger | false |  | 创建者 |
| update\_time | datetime |  |  | 更新时间 |
| update\_by | Interger |  |  | 更新者 |
| Del\_flag | Interger | false | 0 | 删除标识 |

表4.2-2订单表（order）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 数据类型 | 是否为空 | 默认值 |  |
| Order\_id | Interger | false |  | 订单id |
| order\_describe | String |  |  | 订单描述 |
| Order\_number | String | false |  | 订单编号 |
| Order\_price | double |  |  | 订单总价 |
| Order\_state | Interger | False | 0 | 订单状态 |
| Pay\_way | Interger |  |  | 支付方式 |

表4.2-3订单菜品详细表（order\_details）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 数据类型 | 是否为空 | 默认值 |  |
| Order\_details\_id | Interger | False |  | 主键 |
| Order\_number | String | False |  | 订单编号 |
| Food\_id | Interger | False |  | 菜品id |
| Food\_number | Interger |  |  | 菜品数量 |

表4.2-4菜单表（food）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 数据类型 | 是否为空 | 默认值 |  |
| food\_id | Interger | false |  | Id |
| food\_name | String |  |  | 菜名 |
| food\_state | Interger |  | 0 | 状态，是否售罄 |
| food\_describe | text | False |  | 菜品描述 |
| Food\_price | double |  |  | 价格 |

表 4.2-5用户信息表（sys-user）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 数据类型 | 是否为空 | 默认值 |  |
| user\_id | Interger | false |  | Id |
| Dept\_id | Interger | false |  | 部门id |
| user\_name | String |  |  | 用户名 |
| Login\_name | String |  | 0 | 登录账号 |
| email | String | False |  | 用户邮箱 |
| Phone\_number | String | false |  | 用户手机号 |
| Sex | Char |  |  | 用户性别 |
| Avatar | String |  |  | 头像路径 |
| Password | String |  |  | 密码 |
| Salt | String |  |  | 盐密码 |
| Status | Char |  |  | 账号状态 |
| Login\_ip | String |  |  | 最后登录ip |
| Login\_date | Datetime |  |  | 最后登录时间 |
| Pwd\_update | Datetime |  |  | 密码最后更新 |

表 4.2.6 用户在线记录表（sys\_user\_online

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 数据类型 | 是否为空 | 默认值 |  |
| session\_id | Interger | false |  | Id |
| Login\_name | String |  | 0 | 登录账号 |
| Dept\_name | String |  |  | 部门名称 |
| ipaddr | String | false |  | Ip地址 |
| Ip\_location | String |  |  | Ip登录地点 |
| browser | String |  |  | 浏览器类型 |
| os | String |  |  | 操作系统 |
| Status | String |  |  | 在线状态 |
| Start\_timestamp | Datetime |  |  | Session创建时间 |
| Last\_access\_time | Datetime |  |  | Session最后访问时间 |
| Expire\_time | Datetime |  |  | 超时时间 |

表 4.2.7 部门表（sys\_dept）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 数据类型 | 是否为空 | 默认值 |  |
| dept\_id | Interger | false |  | Id |
| parent\_id | Interger | false |  | 父部门id |
| ancestors | String |  |  | 祖级列表 |
| Dept\_name | String |  |  | 部门名称 |
| Order\_num | Integer |  |  | 显示顺序 |
| Leader | String |  |  | 负责人 |
| leader\_phone | String |  |  | 负责人电话 |
| Leader\_email | String |  |  | 负责人邮箱 |
| Status | Integer |  |  | 部门状态 |

## 4.3用户界面设计

### 4.3.1 登录界面

主打简约风格，图片素材取自网络

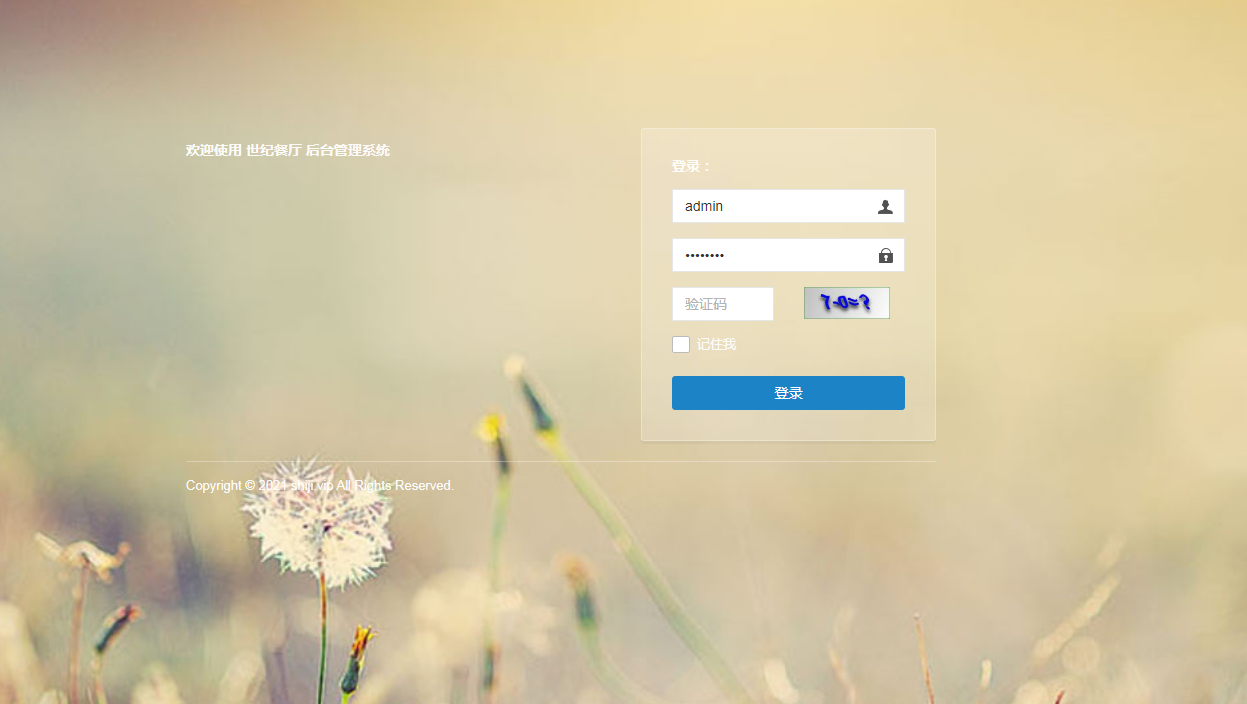


图 4.3.1-1

### 4.3.2 系统主界面

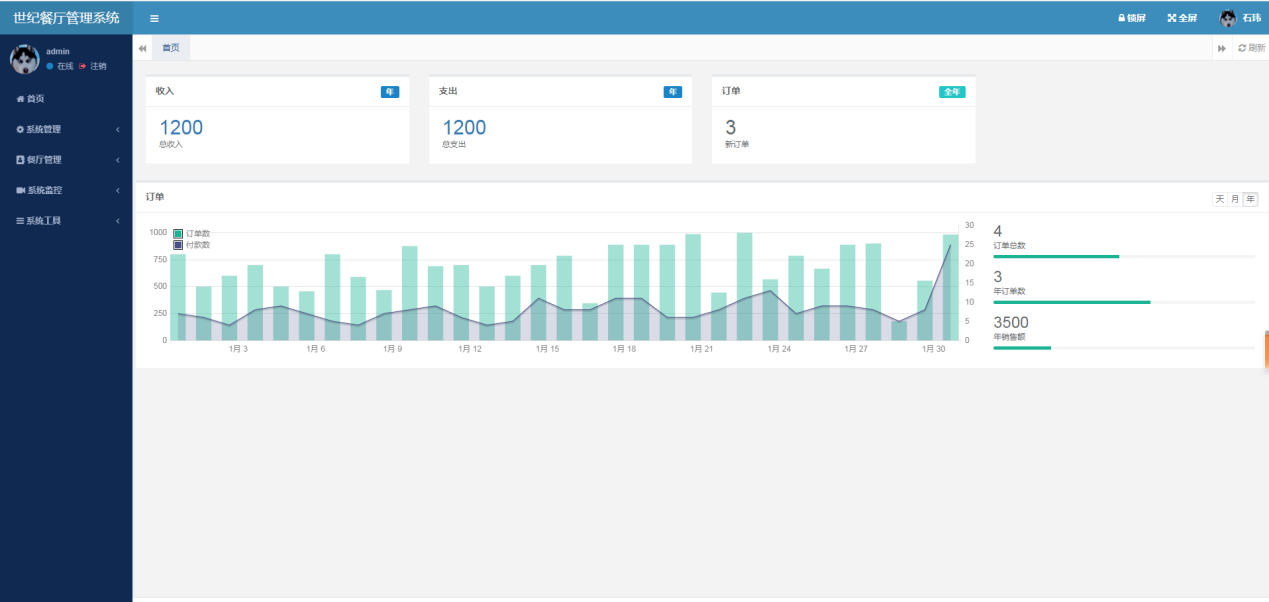


图 4.3.2-1

默认以蓝白为配色，提供有换肤功能，可供客户自己选择

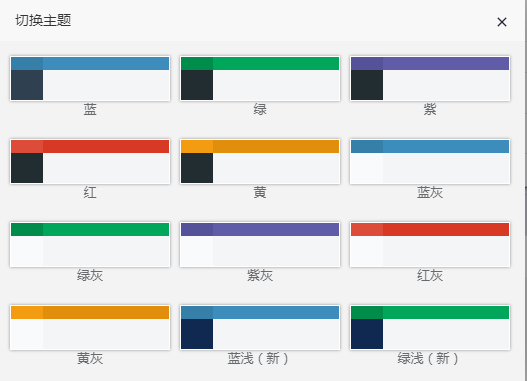


图 4.3.2-2

将主要的功能菜单放到左侧导航栏，右边区域则用来做数据显示，方便客户操作

### 4.3.3 员工管理界面

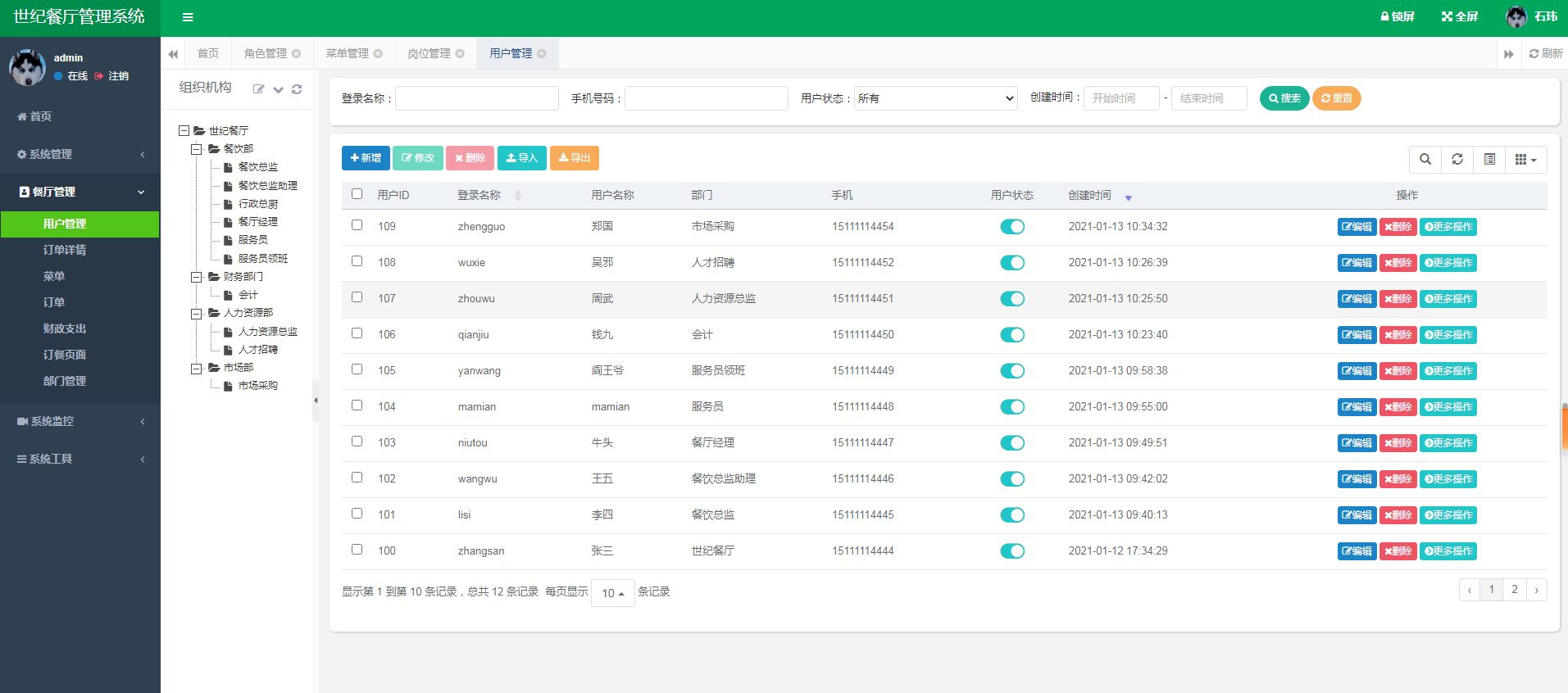


图 4.3.3-1

该界面为老板所能看到的界面，列举了全部门的员工信息，上方工具栏能人工搜索指定员工的个人信息，下方的工具按钮能进行员工信息的增删改查，而右侧的工具按钮则是能对对应的员工信息进行操作。

### 4.3.4 部门管理界面

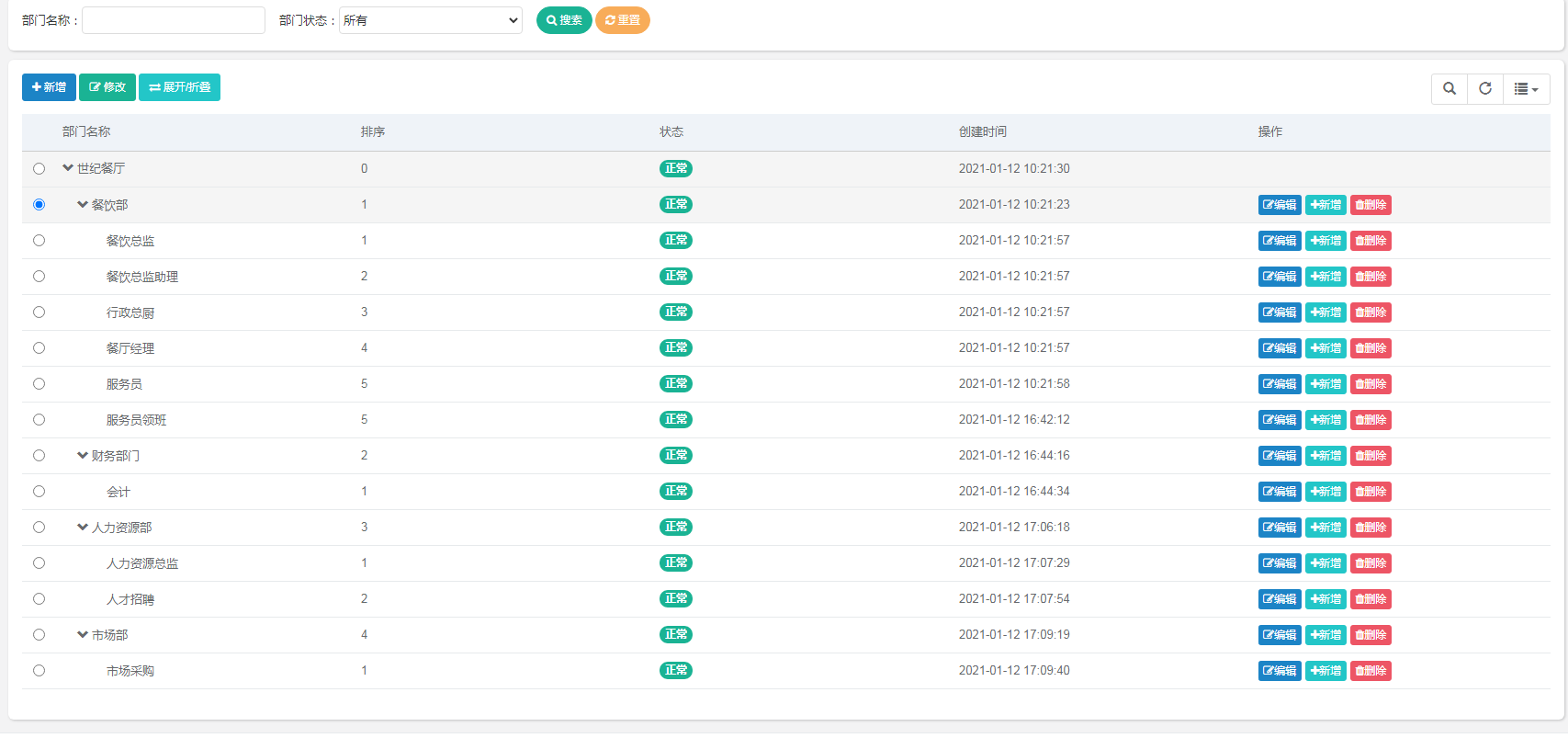


图4.3.4-1

该界面同样为老板所能看到的界面，列举了所有的部门，上方工具栏能人工搜索指定部门，下方的工具按钮能进行对部门的增删改查，而右侧的工具按钮则是能对对应的部门进行操作，同时能将部门下的子部门进行折叠，浏览起来更加简洁，不会因为部门过多而让人应接不暇。

### 4.3.5 商品管理界面



表4.3.5-1

该界面同样为所有员工都能看到的界面，列举了所有的菜品，上方工具栏能人工搜索指定菜品，下方的工具按钮能进行对菜品的增删改查，而右侧的工具按钮则是能对对应的菜品进行操作。

### 4.3.6客户个人界面



表4.3.6-1

客户能查看或修改自己的个人信息，还能上传头像

## 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:

*/\*\*  
 \* 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，就返回json格式的字符串。* **if** (ServletUtils.*isAjaxRequest*(request))  
 {  
 **return** ServletUtils.*renderString*(response, **"{\"code\":\"1\",\"msg\":\"没有登录或登录超时。则提示客户请重新登录\"}"**);  
 }  
  
 **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 = **"用户名或密码错误"**;  
 **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

**public** List<SysUser> selectUserList(SysUser user);  
  
**public** List<SysUser> selectAllocatedList(SysUser user);  
  
**public** List<SysUser> selectUnallocatedList(SysUser user);  
  
**public** SysUser selectUserByLoginName(String userName);  
  
**public** SysUser selectUserByPhoneNumber(String phoneNumber);  
  
**public** SysUser selectUserByEmail(String email);  
  
**public** SysUser selectUserById(Long userId);  
  
**public** List<SysUserRole> selectUserRoleByUserId(Long userId);  
  
**public int** deleteUserById(Long userId);  
  
**public int** deleteUserByIds(String ids);  
**public int** insertUser(SysUser user);  
  
**public boolean** registerUser(SysUser user);  
  
**public int** updateUser(SysUser user);  
  
**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** {  
 String filePath = RuoYiConfig.*getUploadPath*();  
 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，

# 6 系统测试

## 6.1测试目的

为了检测系统是否存在某些漏洞，导致应用程序无法正常运行，需要开发者对各功能模块做一个简单测试，保证项目上线后用户的良好体验，避免造成不必要的财产损失

## 6.2测试方法

使用了swagger工具对各功能模块的控制层接口做了简单测试，通过填入测试用例的数据确保每个接口都能返回正确的message，响应值为200，测试完成之后在登录系统测试各功能之间是否有相互冲突的情况。

## 6.3测试用例

表6.3-1 测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试用户 | 用户身份 | 是否被删除 | 测试情景 | 测试情况 |
| admin | 超级管理员 | false | 登录系统管理员界面 | 成功登录 |
| shiwei | 店长 | false | 登录客户界面 | 页面跳转失败（已解决） |
| zhangsan | 员工 | false | 登录员工界面 | 成功登录 |
| zhengguo | 员工 | true | 测试被删除员工能否登录 | 登陆失败 |
| Admin | 超级管理员 | False | 测试员工的增删改查 | 操作成功 |
| Zhangsan | 员工 | False | 测试头像上传功能 | 操作成功 |
| Ruoyi | 员工 | False | 测试员工的权限，能否修改权限以外的员工信息 | 操作失败 |

# 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。

这是本人第一次完成度比较高的项目，期间遇到许多困难，通过查资料，问前辈都一一解决，这些经历让我意识到经验不足是多么致命的缺点，经验老道的解决问题不拖泥带水，而我自己解决的问题往往会留下许多后遗症。

在分析设计过程中，主要是在分析系统的功能设计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

[1] 《国内餐饮电子管理系统现状浅析》

<https://wenku.baidu.com/view/66e03156f311f18583d049649b6648d7c1c708eb.html>

1. 《餐饮管理系统文献综述》

<https://wenku.baidu.com/view/85018b340875f46527d3240c844769eae109a33c.html>

[3] 《深入理解java虚拟机》

1. 《详细Spring Boot教程之入门》

<https://blog.csdn.net/DBC_121/article/details/104383089>