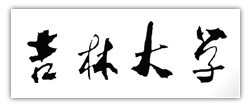


****

**本科生毕业论文（设计）**

**中文题目** 基于微信餐厅服务系统服务器端的设计与实现

**英文题目** Design and Implementation of WeChat Restaurant Service System Server

**学生姓名** 余海涛 **班级**551702 **学号** 55170237

**学 院** 软件学院

**专 业** 软件工程

**指导教师** 王勇 **职称** 讲师

**吉林大学学士学位论文（设计）承诺书**

本人郑重承诺：所呈交的学士学位毕业论文（设计），是本人在指导教师的指导下，独立进行实验、设计、调研等工作基础上取得的成果。除文中已经注明引用的内容外，本论文（设计）不包含任何其他个人或集体已经发表或撰写的作品成果。对本人实验或设计中做出重要贡献的个人或集体，均已在文中以明确的方式注明。本人完全意识到本承诺书的法律结果由本人承担。

学士学位论文（设计）作者签名：

2021年5月20日

**基于微信餐厅服务系统设计与实现**

摘要

自2014年我国开始大规模地部署4G网络以来，互联网便开始从pc互联网朝着移动互联网方向高速发展，O2O模式也随之萌芽且逐渐发展成熟。随着网络普及率的提高，以及移动设备互联网应用的深入，餐饮行业开始与互联网接轨。除了倍受关注的外卖行业，“在线点餐”课题也被频繁提出，这类服务开始在北京、上海等经济发达的一线城市开始试运行，并受到了餐饮行业商家和消费者的一致欢迎。至今，移动互联网、O2O模式、智能移动端管理系统的概念已经发展完善，餐饮市场上各式各样的智能下单服务陆续出现在我国的各大城市，二维码和手机、平板等移动智能设备也出现在了大部分餐厅的点餐桌上，“扫码下单”也成为了餐桌上的标语。为了让更多的餐厅能用上智能的餐厅管理系统，享受其带来的便利与利润，本文采用主流的前后端分离技术，设计与开发一套从餐厅消费者到管理员使用的完整的微信餐厅服务系统。

**关键词：微信小程序，餐饮，Spring Boot，Vue，前后端分离**

Design and Implementation of WeChat Restaurant Service System

Author: Yu Haitao

Tutor: Wang Yong

**Abstract**

Since China began to deploy 4G network on a large scale in 2014, the Internet has started to develop rapidly from PC Internet to mobile Internet, and O2O mode has sprouted and gradually matured. With the improvement of network penetration and the deepening of Internet application of mobile devices, the catering industry began to integrate with the Internet. In addition to the takeout industry, the topic of "online ordering" has been frequently proposed. This kind of service has been put into trial operation in Beijing, Shanghai and other economically developed first tier cities, and has been unanimously welcomed by the catering industry businesses and consumers. So far, the concepts of mobile Internet, O2O mode and intelligent mobile terminal management system have been developed and perfected. All kinds of intelligent ordering services in the catering market have appeared in major cities in China. QR code, mobile phone, tablet and other mobile intelligent devices have also appeared on the ordering table of most restaurants, and "code scanning ordering" has also become a slogan on the table. In order to make more restaurants use the intelligent restaurant management system and enjoy the convenience and profit, this paper uses the mainstream front-end and back-end separation technology to design and develop a complete wechat restaurant service system from restaurant consumers to administrators.

**Keywords: WeChat applet, catering, Spring Boot, Vue, front and rear end separation**

**目 录**

[第1章 绪论 1](#_Toc71900745)

[1.1概述 1](#_Toc71900746)

[1.2背景介绍 1](#_Toc71900747)

[1.3微信点餐的优势 2](#_Toc71900748)

[1.4研究意义 3](#_Toc71900749)

[1.5研究现状 4](#_Toc71900750)

[1.6论文结构 4](#_Toc71900751)

[第2章 相关技术介绍 6](#_Toc71900752)

[2.1 Apache Maven 6](#_Toc71900753)

[2.2 Spring Boot 7](#_Toc71900754)

[2.3 Mybatis 11](#_Toc71900755)

[2.4 开发工具介绍 12](#_Toc71900756)

[2.4.1 IDEA 12](#_Toc71900757)

[2.4.2 VSCODE 13](#_Toc71900758)

[2.4.3 微信web开发者工具 13](#_Toc71900759)

[第3章 需求与可行性分析 15](#_Toc71900760)

[3.1食客用户的需求 15](#_Toc71900761)

[3.2管理员用户的需求 16](#_Toc71900762)

[3.3可行性分析 18](#_Toc71900763)

[第4章 系统分析与设计 19](#_Toc71900764)

[4.1系统概要设计 19](#_Toc71900765)

[4.1.1系统整体架构 19](#_Toc71900766)

[4.1.2微信小程序业务架构 19](#_Toc71900767)

[4.1.3后台管理业务架构 20](#_Toc71900768)

[4.2数据库结构设计 23](#_Toc71900769)

[4.2.1数据库概要 23](#_Toc71900770)

[4.2.1数据库表单结构 24](#_Toc71900771)

[第5章 系统功能开发实现 28](#_Toc71900772)

[5.1环境准备 28](#_Toc71900773)

[5.1.1 开发环境配置 28](#_Toc71900774)

[5.1.2 部署环境配置 30](#_Toc71900775)

[5.2后台管理系统主要功能实现 31](#_Toc71900776)

[5.2.1 管理员登录模块 31](#_Toc71900777)

[5.2.2 菜单管理模块 34](#_Toc71900778)

[5.2.3 餐桌管理模块 39](#_Toc71900779)

[5.2.4 排队管理模块 41](#_Toc71900780)

[5.2.5 订单管理模块 43](#_Toc71900781)

[5.2.6 评论管理模块 45](#_Toc71900782)

[5.2.7 店铺设置模块 47](#_Toc71900783)

[5.2.8 桌台模式模块 50](#_Toc71900784)

[5.3微信点餐小程序主要功能实现 54](#_Toc71900785)

[5.3.1 用户登录模块 54](#_Toc71900786)

[5.3.2 扫码点餐模块 55](#_Toc71900787)

[5.3.3 订单模块 57](#_Toc71900788)

[5.3.4 排队预约模块 59](#_Toc71900789)

[5.3.5 餐厅评价模块 60](#_Toc71900790)

[5.4部署服务器 62](#_Toc71900791)

[第6章 总结 63](#_Toc71900792)

[参考文献 64](#_Toc71900793)

[致 谢 65](#_Toc71900794)

# 第1章 绪论

## 1.1概述

本文将简要介绍我国餐饮行业近年与互联网结合发展情况以及研究课题的现状，阐述智能服务系统解决方案为餐饮行业发展提供强大动力的事实。然后介绍如何搭建一个前后端分离的B/S结构餐厅服务系统，系统包括两个B端一个S端。B端分为餐厅食客使用的微信小程序，和餐厅工作人员使用的后台管理系统网页。S端为运行在Linux服务器上，以spring boot为主体框架搭建的负责数据存储和业务处理的服务器端程序。本文将重点论述该基于微信餐厅服务系统的功能设计思想和实现过程、程序中数据的处理和存储的方式以及多端信息交互的方法。其中会用到目前较新的技术框架和方案，提供最为便捷的开发思路和体系，并会总结我在开发过程中得到的经验，为其他开发者提供有效的参考。

## 1.2背景介绍

餐饮业是重要的服务行业，受到国家的高度重视，因为饮食直接关系到我们人民的生命健康和生活水平[1]。餐饮业是最贴近人民日常生活，最能影响人民生活质量的行业之一。2020年公布数据表明，我国餐饮业零售额度为12352亿元，体现我国餐饮行业规模之大。相比2019年的10345亿元，同比增长19.4%，体现我国人民生活质量提高，更多人选择餐厅就餐，在饮食方面消费的投入增加[2]。餐饮行业发展如此迅速，离不开“在线点餐”对餐饮行业服务和管理高效化的技术支持。若餐厅还是延用传统的营业模式，必然不会有当今餐饮行业的井喷式发展。

目前餐饮服务行业与互联网结合的服务系统产物大体可分为两种。

一种是以美团，饿了么为首的外卖服务软件，这类软件的出现和普及给餐饮行业带来了跨时代意义的改变。它们出现之前，市场上存在的外卖服务十分稀少，就算有也基本都是由商家独立提供，即食客打电话给餐厅订外卖，餐厅派自家员工为食客派送。这种外卖形式受限极大，能有商家订餐电话的食客必然是曾经到店用餐过的食客，这使得食客的订餐成本提高，选择权受限。而外卖软件和商家入驻普及后，食客可以通过外卖软件，以图文并茂的形式，看到多家餐厅的菜品，并且还能看到其他食客对餐厅和菜品的评价，因此大幅扩大了食客的选择范围。从2011年开始，外卖行业在中国兴起，经过几年的发展，大量商家和外卖骑手陆续加入到外卖行业，并频繁成为热议话题进入人们视野中。2020年，我国受到疫情的严重影响，大量行业产业经济大幅受创，传统餐饮门店不得不关门闭店。但却没有给外卖行业带来太多影响，外卖行业仍然处于上升阶段，这年新增外卖相关企业670000多家[3]。虽然外卖行业发展迅猛，但由于我国的餐饮行业监管制度没有跟上变化，外卖行业的负面问题也层出不穷，卫生条件差，料理包加热出餐等问题成为消费者谨慎点外卖的理由，还是有很多消费者选择门店就餐。

企业的创新需要跟上时代的变化，即使是小小的餐饮店也应如此，餐饮行业与互联网结合的第二种产物便是餐厅的在线自助点餐系服务统，对于线下餐饮门店，传统餐饮门店的管理和服务多为纯人工服务管理，对于市井小店，由店员用纸币记录点菜单进行服务。若是较为前沿的餐厅则可能采用柜台机进行餐厅管理，这种系统多为封闭的，不与客户交互，全由店员操作的嵌入式系统。消费者与柜台的信息交互还需要服务员充当媒介，即服务员记录客人点餐，再将订单递到柜台，由收银员添加订单，通知后厨制作，这种模式的效率显然是较低的。在线自助点餐服务系统将成为主流。在提供自助点餐服务的餐厅，消费者进入餐厅后，选择合适座位数的空闲餐桌坐下，拿出移动智能设备，打开摄像头扫描餐桌上的点餐码，即可看到餐厅菜单，相比传统的纸质菜单本，手机点餐可更灵活地编辑图片文字信息，还可让消费者添加评价信息，使菜单信息内容更加丰富，且更具有参考性。

## 1.3微信点餐的优势

相比传统的餐厅服务模式，“扫码点餐”模式有着多方面的优势。首先对食客而言，若是用传统方式点餐，十分依赖服务员在场，而餐厅的服务员数量有限，若没有空闲的服务员，需要等待。即使有服务员在场，点餐步骤也十分繁琐，需要翻阅纸质菜单，还要等服务员记录点餐内容。并且食客就餐完成后，也还要到收银台结账支付，才完成整个用餐流程。而“扫码点餐”模式中，食客只需一部手机即可独自完成从点餐、下单最后支付的所有步骤。这种利用智能餐厅服务系统优化后的点餐流程，让食客在这个生活节奏快速的时代节约了宝贵的时间，也同时提高了食客的用餐体验。对于餐厅店家方面，我调查采访了市里多家使用“扫码点餐”服务方式的餐厅，结果表示：“扫码点餐”模式减少了餐厅工作人员的服务流程，使得同样规模的餐厅需要的服务员人数减少，人力成本上就节省了约30%。此外，优化的点餐流程平均减少了食客20%的占桌时间，进而让餐厅的翻台率提高，使餐厅平均增加约25% 的营业收入，这个数字在节假日餐厅生意火热的时候则会更高。此外，将餐厅服务系统引进餐厅，还为餐厅管理带来了许多便利。例如，使用服务系统可以更高效快速地自定义编辑餐厅的菜单，餐厅管理员在后台管理系统上对菜单内容修改后可以直接反映到食客手机上看到的菜单界面上，而不像传统方式一样，若更改菜单内容则需要在多本菜单本上涂改，甚至重新打印菜单，这也为餐厅节约了成本。

既然是“扫码点餐”模式，那么可选的实现方式有这以下几种：Web页面、移动应用以及小程序。Web应用的好处是开发难度低，但实际中基于网页的方式并不能为用户带来良好的体验。移动客户端虽然可以给用户最优的使用体验，但也提高了了用户的使用门槛，必须下载该移动APP才能使用，这在一些临时就餐场景中是十分麻烦的，且为了适配主流的两种移动设备系统Android和IOS，开发者需要开发两个版本的客户端，这大大增加了开发成本，所以也并不适合使用。综合考虑后决定，选择使用小程序的形式实现客户端的点餐系统，它能带来不错的用户体验，无需下载，点击即用，让食客用户在使用的时候门槛降低。此外小程序可以借用微信平台，来实现跨平台使用，因为不论是安卓设备，还是IOS设备，人们几乎都会安装微信，不需要针对不同的操作系统去开发不同版本的应用，这一点让系统的开发难度降低，也大大减少了工时。而小程序中亦有多种选择：微信小程序、支付宝小程序、快应用都可以做到点击即用。但是快应用不支持IOS首先被淘汰；支付宝虽是应用最广泛的支付平台，有7.11亿活跃用户（2020年数据），但它的业务更倾向于单纯的支付，用户黏度不如微信。微信作为最火热的社交应用，拥有更大的用户基数，更强的社交性质，更大的用户黏度，选择微信平台开发小程序最为合适的做法。

## 1.4研究意义

社会变化至今，不论是工业生产活动还是日常生活活动，都朝着更高效、更高品质的方向发展。人们的生活节奏越来越快，而饮食是每一个人在每一天都需要面对的问题，餐饮行业也必然最求高效化、高品质化的解决方案，传统的餐饮作业模式已经不适合现在这个科技改变生活的年代，让餐厅数字化、网络化是时代发展的趋势。这次课题研究将研发一套为餐厅管理员和消费者使用的餐厅服务系统解决方案，让实体门店餐饮行业也感受到移动互联网发展带来的便利。

## 1.5研究现状

前面已经分析了消费者客户端技术路线的选择策略，选择使用最符合开发和运作需求的微信小程序客户端形式。微信小程序开发中，使用官方小程序原生框架mina，它由微信官方提供，系统功能强大，结构轻便简洁。它提供了视图层元素的描述方法和风格定义方法，也给出了逻辑层处理的结构定义，可简单直接地处理用户的操作，同时微信官方还提供可满足大部分开发和应用场景的系统API供调用。

关于餐厅员工使用的餐厅后台管理系统，选择采用web页面形式进行前端独立开发。如今前后端分离开发已经成为主流，Vue作为简单易学的前端轻量级框架，存在强大的数据双向绑定功能，拥有组件化开发，数据与结构分离的先进思想，同时有性能优异，运行速度快的优点，成为前端开发的热门框架。而Layui拥有丰富的简洁美观的前端组件库，模块化编程，兼容性强，是为开发人员提供的优秀ui框架。于是餐厅后台管理系统前端页面决定采用Vue+Layui的技术路线进行开发。

服务器端的Java web应用开发从最早的Servlet技术开始已经经历了许多变化，先是java Servlet 作为一种改进的CGI（公共网关接口）出现，解决了每次请求都要启动一个系统进程（一个JVM）的问题，大大降低了系统开销。后来Web编程脚本语言（PHP/ASP/JSP）兴起，JSP（java server page）也随之出现，它的主要作用还是动态展示网页内容。随着业务的复杂化，设计模式对开发的效率作用更为突出，各路框架横空出世，MVC、ORM成为了主流，其中Spring+Spring MVC+Mybatis便是MVC、ORM模式的一个优秀经典实例。Spring Boot消除了大量繁琐的XML配置，内嵌入了Tomcat服务器且不需要手动部署，同时利用注解完成了大部分自动配置功能，大大提高了web应用的开发效率，于是本文选择使用spring boot作为服务器后端的主体框架。

## 1.6论文结构

本文通过充分调查需求和学习技术知识，介绍一个前后端分离的微信餐厅服务系统的开发过程。

第一章绪论，介绍了当前O2O模式中餐饮行业的背景和现状，以及本次研究的目的和意义。

第二章相关技术介绍，先介绍后端服务器开发过程中要用到的主要框架。会从基本介绍和如何配置使用这两个角度来介绍。然后会介绍开发工具，并说明选择原因。

第三章需求与可行性分析，首先分别餐厅食客和餐厅管理人员的角度来分析，探究系统需要拥有哪些功能才能完成基本的服务，然后从开发人员角度分析项目开发的可行性。

第四章系统分析与设计，首先对系统的架构进行了概要的分析与设计，然后设计了系统数据库的结构。

第五章系统功能开发与实现，分别介绍了后台管理系统和微信点餐小程序各功能模块的前后端实现过程。

第六章总结，对该系统开发的全流程做出总结，评价系统的能力，分析系统的不足。

# 第2章 相关技术

## 2.1 Apache Maven

Maven是一款在Java领域内被广泛使用的项目构建工具，在项目中充当着“管家”角色。它最大的作用是管理项目的jar包依赖，有了它就不用再向以前一样手动下载jar包加入lib库中再引用，因此避免了繁琐的导入流程。此外它还能配置项目的编译参数和打包方式，为项目部署提供了便利。

## 2.2 Spring Boot

1. Spring系列简介

在Spring框架出现以前，J2EE领域从最初的Servlet开始，经历了许多的变化。传统J2EE项目中，使用Servlet处理http请求，JavaBean和EJB处理业务逻辑，JSP、JSF渲染前端页面，逐渐形成和定义了经典的三层组件结构。但这时候，web应用的开发还是基于Web Form的结构的，其视图和逻辑紧密耦合，页面是基于文件URL渲染的，这使得代码难以维护和扩展，非常不利于模块化业务的编写。正因如此，MVC的概念被人提出，即“模型-视图-控制器”的设计模式。

图2-1 MVC结构

MVC结构将页面视图与业务层分离，只需统一接口，彼此分开编码，互不影响，具有耦合性低，重用性高，易于维护，部署的优点。Spring MVC Framework则是MVC模式的一项优秀实践。

讲Spring MVC之前，先要介绍一下Spring的由来：Spring之前的时代，J2EE以SUN公司研发了EJB，Java后端都依照着Enterprise Java Bean的标准进行开发。但EJB运行在EJB容器服务器上，Servlet以及JSP却运行在Web容器上，为了让Web容器能够调用EJB内的业务功能，每开发一个EJB，都需要创建多个配置文件和接口，这样导致配置的工作量不亚于开发的工作量，这给测试人员也带来了更多繁文缛节的流程。同时Web容器通过网络调用EJB容器，必然十分依赖网络质量，这导致服务器的性能难以稳定。种种原因导致当时EJB的规范饱受开发者诟病。2002年，美籍作者Rod Johnson 出了一本书来总结他发现的许多EJB存在的缺陷，在书中他还提出了一种全新的解决方案，这就是最初的Spring概念。

如今Spring已经发展多年，成为了Java程序员最欢迎的轻量级后端框架，它同时具备卓越的性能和优秀的思想理念，理所当然地坐上了后端开发框架的王座。Spring之所以卓越，必然是因为它的两大核心理念——IOC和AOP。IOC其实就是基于控制反转思想的一个Java Bean的容器，它接管了MVC三层内几乎所有的实例，负责对它们进行创建、调用和销毁。IOC不仅接管了实例的生命周期，还很好地协调了实例之间的依赖关系，完成依赖的自动注入，使得一个实例在创建的时候可以非常简单的注入另一个实例作为其类成员使用。该功能是通过类ApplicationContext实现的，它比基础的IOC容器多了一个步骤，在初始化时会对所有实例的属性进行是否已注入的自检，防止在使用实例时抛出空指针异常。但Spring不论是向IOC容器添加Bean还是为Bean注入依赖，都离不开步骤繁琐XML文件配置。AOP则基于动态代理设计模式的思想，目的是减少代码的冗余量，消除重复代码。Spring 的AOP中定义了“前置、后置、环绕、异常、引介”五种通知类型，将重复的代码置于通知中，把核心的业务逻辑代码独立出来，解决了模块间重复代码过多，耦合性太强的问题。开发人员利用Spring的AOP更高效地实现事务处理、异常处理、日志管理以及权限控制等功能。但AOP也需要编写XML文件来配置切面、切入点等，相对繁琐。

说完Spring，回归到Spring MVC。先前说到，MVC模式用控制层来协调匹配用户界面和业务逻辑之间的映射关系，目的是降低两者之间的耦合性，让业务更加方便拓展，让代码更加容易复用。Spring MVC秉持这种理念，与自家的Spring集成使用，给出如下方案：

后端

IOC容器

前端 视图层

控制层

业务层

数据访问层

NoSQL

数据库

夹带AOP事务管理的模型层

图2-2 Spring MVC结构

Spring MVC利用Spring的IOC容器，将控制层、业务层、数据访问层的Java Bean存储在IOC容器中，规范管理了各个组件，并用控制反转的方式调用。Spring MVC也利用了Spring的AOP特性，为模型层扩展了非常方便实现的事务管理功能。相比之前更为火热的Struts2框架，Spring MVC的执行效率更高，与Spring框架的配合更加简便，于是后来逐渐取代了Structs2。

Spring和Spring MVC的出现，集成了Java web开发技术发展演化出的先进思想，对后端开发人员来说是更称手的工具，但Spring框架中存在大量的XML配置内容，让开发人员不光要用JAVA代码编写业务逻辑，还要经常修改配置文件内容，在适配第三方框架时，更是要新建各种xml、properties文件，这让项目的结构变得非常复杂，代码变得难以维护，当时人们称Spring为“配置地狱”。于是，在Java 5推出“注解（Annotations）”的新特性，Java 6.0允许自定义注解，Java 8支持多重注解等一系列Java版本的更替中，Spring开始将注解大范围应用到框架中，将复杂的配置过程改为Java代码内的声明式注解来实现，极大幅度地降低了项目结构的复杂程度，提高了编码开发的效率。虽然Spring有简化其配置过程，但简化得不够彻底，不够智能，真正彻底简化的是Spring Boot。

Spring Boot集成了先前框架的优点，同时提供了消除XML配置的完整解决方案，但又同时支持使用XML配置以兼容旧的开发模式。在Spring Boot项目里，如果想要配置程序的端口号，数据库源，数据库连接池，静态资源地址，服务响应最大时间……等等内容，都可以十分简洁清晰的分行缩进格式写在application.yml文件中。若想要将MVC三层中的Java Bean加入到IOC容器托管，只需在其类声明前使用注解，注解中可以指定名称，方便后续在其他类中作为成员注入。此外，它还提供了“starters”风格的组件依赖。该风格提倡将各种开发环境场景进行封装形成一个starter，当希望在Spring Boot上使用该扩展框架时，只需要导入该扩展框架的starter。如今，许多优秀的第三方框架都会有一个“spring-boot-starter-xxx”依赖版本，这极大地增强了Spring Boot的可扩展性。此外，它还提供了强大的自动装配能力，在之前Spring中所有需要XML文件配置的模块，例如添加一个拦截器，修改静态文件路径等配置需求，都支持在代码中直接插入注解的形式进行的方便设置。在部署方面上，很长时间以来，人们都是将项目模块打成war包，然后配置在tomcat服务器内运行。而Spring Boot内嵌了一个微型服务器，并提供了三种选择，缺省选择为tomcat。通过内嵌服务器，只需要将项目打包成jar包，直接运行jar包就可以完成部署。

一言以蔽之，Spring Boot凭借其自身轻量，使用简易，starter组件丰富而功能强大且易扩展的特点，是Java社区中最具影响力的技术框架。

1. Spring Boot使用方法

介绍中也有提到，Spring Boot具有十分轻量的特点，只需用Maven导入其依赖即可使用。也多亏了“starters”模式，我们只需要按照约定的格式创建项目结构，引入一个它的starter依赖，组件就会自动配置，无需程序员过多操心。

1. 创建一个空白的maven项目
2. Maven配置中设置parent项目，如图：

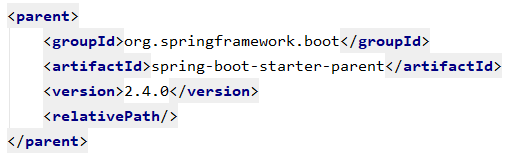


图2-3 spring-boot-parent

1. 对于一个java web项目场景，在Maven中引入Spring Boot网页应用依赖即可，该依赖内部包含了web应用开发所需的必要依赖环境：spring-web、spring-webmvc、spring-boot-starter-tomcat等。

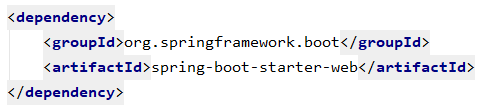


图2-4 spring-boot-web依赖

1. 创建启动类：



图2-5 spring boot 启动类

1. 至此，一个基本的Spring Boot项目已经搭建完成。执行启动类的main方法，即可在默认端口80上开启一个spring boot服务。在此基础上，再根据项目需求导入数据库访问、权限控制等组件，以此完善服务的功能。

## 2.3 Mybatis

1. Mybatis 简介

后端必然要与数据库交互，Mybatis便是一款简化与数据库交互过程的半自动化的Java持久层框架。它的特点是支持动态SQL、高级映射。

与Java最基本的JDBC相比：

1. Mybatis将SQL语句和Java代码进行了分离，Java代码定义了接口，SQL语句放在xxxMapper.xml的配置文件中，利用了参数映射方式，将Java函数的参数映射到SQL语句的变量。同时在查询语句中，返回值会通过属性名和字段名映射直接封装为Java对象。
2. 其动态SQL功能，对SQL语句进行半自动化封装，减少了大量的代码冗余，同时又兼顾了灵活性。
3. Mybatis 使用方法
4. 用Maven导入mybatis-spring-boot-starter依赖，指定适当的版本号
5. 在application.yml文件中配置mybatis映射文件地址：

mybatis:

mapper-locations: classpath:mapper/\*.xml

1. 在项目源码目录和资源目录下均分别新建mapper文件夹
2. 源码目录mapper文件夹内新建Java接口，接口使用@Mapper注解标注，接口内声明SQL方法。
3. 实现SQL方法：
   1. 注解方式实现，在Mapper接口类中的SQL方法前使用@Select、@Insert、@Update、@Delete等注解，在注解内书写SQL语句。
   2. XML文件配置实现，在资源目录的mapper文件夹内新建与Mapper接口类同名的.xml后缀文件，在XML文件内用xml节点的方式配置实现SQL方法。该方式实现还需指定namespace映射到源码目录的Mapper接口下，并且每个<insert>、<select>、<update>、<delete>节点需要指定id值，且其id值必须与Mapper接口的方法名一致，

这两种方式均可以实现SQL方法，但是两者的适用场景不同，如果用注解的方式实现，更适合的是比较简单的SQL语句，这种方式更加简便，不需要创建多余的XML配置文件，只需在原本接口上加一行注解。但是注解方式也有弊端，当面临比较复杂的SQL业务时，注解方式书写SQL语句反而变得更加困难，也难以维护。例如当书写动态查询方法时，需要额外使用<script>节点将其他节点包括在内，出现引号和双引号时，还需要使用转义字符，而在XML文件内书写，格式更加清晰，也更容易维护。

## 2.4 开发工具介绍

### 2.4.1 IDEA

Idea是JetBrains公司开发的用于Java代码编写的集成开发环境，有最好的代码补全、内容格式化、代码高亮显示体验，在编码过程中，还能智能地提供代码改进策略。不仅如此，开发者为Idea开发了许多好用的插件，lombok就是具有代表性的一个插件，它让Java Entity的编写变得非常简单。

不仅是代码编写的体验上，Idea同时拥有开发过程中需要的大部分工具：支持Git智能管理，点击按钮即可自动提交git commit、git push、git pull等操作；支持SSH终端，不需要打开额外窗口即可通过SSH登陆云端的Linux服务器；支持可视化数据库连接和管理，在开发环境下即可直观的查看数据库表单内容，创建、编辑表单；支持SFTP协议连接，可将云端服务器的文件目录映射到本地IDE上，也可将本地主机的文件拖拽上传到云端服务器，大大简化了部署难度。不夸张地说，众多光环于一身IDEA是Java程序员最好用的集成开发环境。

### 2.4.2 VSCODE

vscode作为一款非常轻量和简单的文本编辑器，十分适合用来编写前端代码。本次项目的前端由vscode配合其live server插件进行开发。

### 2.4.3 微信web开发者工具

微信web开发者工具由微信公众平台官方推出，可用于微信小程序和微信公众号网页的前端开发。提供了许多微信平台的API，如获取用户信息、发送Http请求、获取用户会话文件、调用用户摄像头进行二维码扫描等。最好用的是提供了移动端模拟器，可以直观看到项目运行在移动端时的状态。

# 第3章 需求与可行性分析

## 3.1食客用户的需求

对于开发人员来说，开发微信餐厅服务系统，开发人员的用户是餐厅的老板。而餐厅使用该微信餐厅服务系统的目标，是服务餐厅的用户，也就是前来就餐的食客。利用微信餐厅服务系统，餐厅能够为食客提供更便捷高效的服务，让食客更快更方便地点餐，更方便地结账，进而提升食客就餐体验，也同时提高餐厅的翻台效率，提升餐厅的口碑，最终为餐厅带来更高的收益。所以说，该微信餐厅服务系统最重要的环节在于对餐厅食客用户的需求满足，这也是判断该系统成功与否的关键。因此在对食客用户需求分析时，应当想得更加全面，更加具体。我在分析食客用户需求时，从一个开发人员的身份转变到一个食客的角度，考虑到以下食客用户需求：

1. 进入系统

作为一名食客，进入餐厅的目的就是吃饭，如果为了吃饭还要手机下载一个APP，这对食客来说是非常不便且不情愿的。因此进入系统的方式选择用微信扫描小程序二维码，直接无需下载即可进入系统。

1. 用户登录

食客使用系统需要有个身份标识，用来查看指定的就餐记录，那么用户登录就变得有必要了。最方便快捷的登录方式就是使用微信小程序API，用手机号码直接登录。

1. 排队预约

餐厅平常不忙可以直接入座就餐，但若餐厅生意火爆的时候，就需要有一个排队功能，让食客有序就餐。并且排队功能应该区分不同的餐桌类型，让不同用餐人数的食客团体选择不同座位数量的餐桌进行排队，这样能最高效的安排餐厅资源。

1. 浏览菜单

系统应当提供一个清晰规整且易操作的界面，以分类的形式展示餐厅所提供的食物，食物应该提供完整的信息，包括食物图片、名称、价格、份量等。

1. 购物车

为了方便食客点餐，系统应当提供购物车功能，可以将餐品加入购物车中。

1. 生成订单

将全部要点的餐品加入购物车后，生成订单，显示收费内容，如餐品费用，可选服务费、餐位费等其余费用，并提供支付按钮。

1. 历史订单

食客可在小程序内查看自己在餐厅的消费记录，查看历史账单，账单还应该可分类筛选：进行中的订单、待评价的订单、已评价的订单。

1. 餐厅评价

订单完成后，食客可以针对这次就餐体验，进行评价，评价形式为评分+文字+图片。并且食客在小程序中，应当要能看到其他食客的评价。

## 3.2管理员用户的需求

对于餐厅来说，使用该系统的场景是什么？通常是餐厅收银员在柜台，用电脑打开餐厅管理后台的网页，进行餐桌结账、排队管理、订单管理、菜单管理等餐厅管理业务。此外，餐厅管理员还应该能通过系统直观了解餐厅目前各个餐位的就餐情况。

1. 登录系统

管理员用浏览器输入网址进入餐厅后台管理系统的登录页。登录页进行身份验证，以保证系统安全。

1. 店铺信息维护

通过餐厅后台管理系统，可设置餐厅名称、地址、联系电话以及首页轮播图等餐厅广告信息，用于展示在食客使用的小程序页面上，给食客留下餐厅映像，提高回头客的数量。

1. 餐桌管理
2. 录入餐桌：商家管理员可以将餐厅内现实餐桌的信息录入到后台管理系统中，可设置餐桌图片、桌号、座位数量、所在位置、当前状态和备注等信息。
3. 信息维护：当现实中餐桌发生变动，也应当能通过后台管理系统修改餐桌的信息或添加和删除餐桌。
4. 状态切换：支持餐桌的状态切换，最经常出现的业务是修改餐桌的状态，例如当一张餐桌出现故障，应当将其状态修改为“停用”。平常餐桌还“空闲”、“占用”两种状态，用来区分餐桌上是否有食客就餐，若餐桌上有食客正在就餐，应当为“占用”状态，平时则为“空闲”状态。
5. 菜单管理
6. 餐品分类：餐品列表展示到食客的小程序页面上时，应当是分类展示的，这样更方便食客进行点餐，因此后台需要设置餐品分类。餐厅管理员在添加餐品前，应当首先添加餐品类别，只需指定类别名称即可，并且餐品类别应当支持修改和删除。特殊的是，若要删除一个餐品类别，为了后台数据的稳定性，应该不允许去删除一个已被餐品使用的类别，只有将该类别的所有餐品都修改类别或者删除后，才允许删除该餐品类别。
7. 添加餐品：管理员将餐厅提供的餐品添加到数据库内，餐品信息包含：餐品图片、名称、分类、价格、月售数、备注、状态等。填写餐品分类信息时应当从已存在的餐品类别中选择，而不是手动输入。
8. 切换餐品状态：餐厅在日常营业中，可能出现今日餐品原材料不足，无法继续制作某一餐品的情况。对于这种类似情况，为餐品设置一个状态量是有必要的，餐品状态有：“在售”、“售罄”、“下架”。“在售”状态时，餐品正常显示在食客的小程序页面上；“售罄”状态下，则会在小程序页面上标注，已售罄，并且无法加入到购物车中；“下架”状态的餐品则直接不会显示在点餐小程序的页面上。管理员可以根据营业情况在系统后台对餐品的状态进行切换。
9. 删除餐品：删除之前录入到系统中的但已经不需要的餐品。
10. 订单管理
11. 订单查看：订单的生成是食客下单支付后自动生成的，后台无需操作生成订单，可以直接查看订单。订单需包括必要的信息：时间、菜品、费用、客户号码等。
12. 订单筛选：随着餐厅的营业，一个餐厅的订单量是很大的，当需要找到特定订单时，筛选功能尤为重要。系统应当支持通过订单的信息属性对订单进行多条件筛选。
13. 订单统计：对于筛选结果，勾选需要统计的订单，计算总营业额。
14. 排队管理
15. 查看队伍：食客在小程序端选择不同的餐桌类型进行排队，后台管理系统应当能看到排队的情况，并且支持分不同队伍类型查看。
16. 排队作废：对于排队序号为1的食客，可查看其手机号码，联系入座，若联系不上，还应当为管理员提供排队作废功能，避免队伍堵塞。
17. 评论管理
18. 查看评论：每个订单都可对应一条食客的评论，评论中有评分、文字、图片等信息，管理员在后台系统可看到食客的评论
19. 回复评论：商家管理员对评用户评论做出回复。
20. 删除评论图片：同时为了避免文字、图片中存在网络不良信息，为管理员提供删除评论图片的功能。
21. 桌台模式

后台系统对店铺、菜品、订单、评论等信息的查询和维护功能，更多是餐厅还未营业的时候就设置好，在日常营业中的使用频率是较低的，更多时候是进行各个桌台订单的处理。因此后台管理系统应当有一个桌台模式，桌台模式里，可以清晰地看到各个餐桌的使用情况，对应订单情况，以及提供结算订单功能，以便将“占用”状态的餐桌变为“空闲”状态。此外桌台模式还应提供一些快捷入口，如排队列表的管理。

## 3.3可行性分析

产品功能上，对于上述餐厅食客和管理员在使用餐厅服务系统过程中需求的功能，已能基本满足食客用餐体验和餐厅管理员对餐厅的维护需求，并且这些功能对食客和餐厅来说都是需要的功能，一方面方便食客点餐，另一方面也为餐厅管理带来便利，盈利带来提高。

技术上，食客使用的微信小程序，微信官方已有成熟的技术支持，使用MINA框架即可完成前端开发。管理员使用的网页页面使用Vue、Layui技术框架也可快速完成开发。服务器后端使用Spring Boot服务加MYSQL数据库，完成业务处理、数据处理、数据存储的功能。前后端之间则用HTTP请求进行交互。

# 第4章 系统分析与设计

## 4.1系统概要设计

### 4.1.1系统整体架构

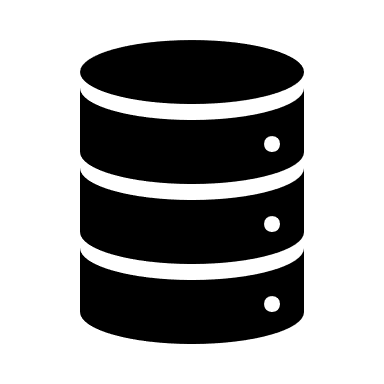
从使用需求上，该微信餐厅服务系统有餐厅食客和餐厅管理员两个用户身份团体。餐厅食客使用便携的移动智能设备：手机、平板等，选用微信平台的微信小程序作为技术支持。餐厅管理人员使用电脑或平板作为系统接入设备，使用浏览器网页作为系统入口。系统整体架构如图4-1所示。

微信餐厅服务系统

微信小程序前端

管理后台

前端



数据库

服务器

后端

http请求

http响应

http请求

http响应

SQL插入、修改等

SQL 查询

图4-1 微信餐厅服务系统架构图

小程序和管理后台发送向服务器后端http请求，服务器后端根据请求路由选择执行不同的业务逻辑。执行业务逻辑过程中，遇到需要保存的数据，则向数据库请求建立数据库连接，向数据库服务器发送SQL插入语句；或者遇到其他类型的业务逻辑则向数据库服务器发送SQL查询、修改、删除等语句。

### 4.1.2微信小程序业务架构

从系统功能上，点餐小程序系统可分为五大模块：用户登录模块、扫码点餐模块、排队预约模块、餐厅评价模块和订单模块。一次普通用餐场景中，至少按顺序经历用户登录模块、扫码点餐模块和订单模块。在此基础上，辅助以排队预约、餐厅评价模块，来提升食客就餐体验。

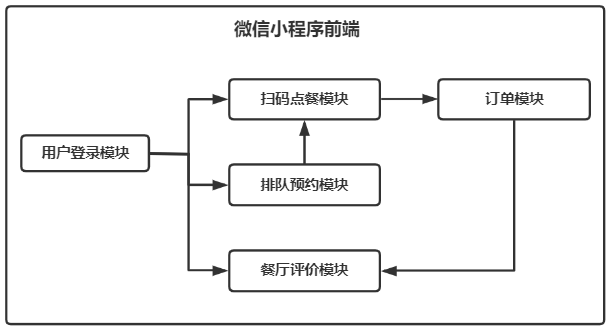


图4-2 微信小程序前端架构图

各模块具体所需功能如下：



图4-3 小程序前端功能模块内容

### 4.1.3后台管理业务架构

一位餐厅的管理员，不必要懂得软件、程序设计的知识。餐厅后台管理的主要作用是为餐厅管理人员提供一个和系统服务器进行数据交互的简易窗口，将一些管理业务封装成简单的图形界面、输入框和按钮，让非开发者可以轻易使用。同时架构需要保证可扩展性、安全性和自定义性，以便适用于不同类型的餐厅。

后台管理业务主要分为：管理员登录模块、桌台模式模块、店铺设置模块、餐桌管理模块、菜单管理模块、订单管理模块、排队管理模块、评论管理模块。

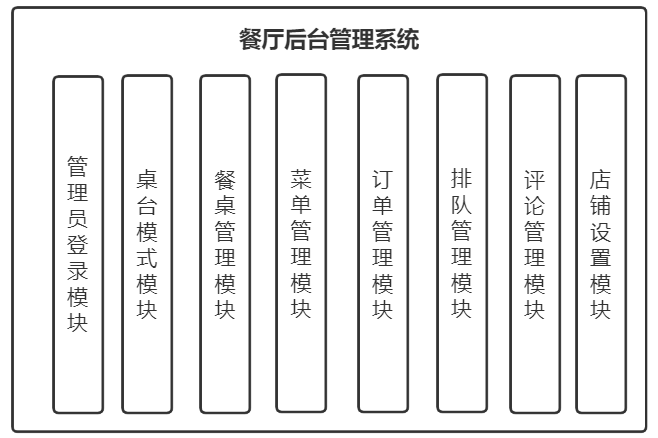


图4-4 餐厅后台管理系统功能模块

这里有提出一个概念：桌台模式模块。餐厅日常营业场景下，餐厅后台管理系统需要一个固定窗口，在这个窗口内，管理员可以直观的看到餐厅目前营业状况下的实时信息，例如管理员需要了解目前餐厅有多少张空的桌子，分别是哪些桌子；有多少张餐桌正有顾客在用餐，分别是哪些；正在用餐的餐桌上，顾客的点餐内容是什么，对应订单是哪个；以及当一位顾客点餐后，管理员希望能在这个窗口上看到一条消息，告诉管理员几号餐桌在某个时间点有一个新的订单；在餐厅忙碌时，使用排队系统，管理员可以通过这个窗口直接看到排队情况，不同类型的餐桌分别有多少顾客排队，已经排队到第一位的顾客是否能够正常就坐点餐，若不能则需要将他的排队手动作废。

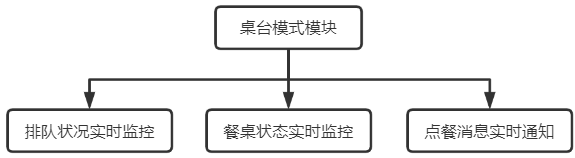


图4-5 后台管理系统桌台模式模块

介绍完桌台模式模块后，下面简要介绍一下餐厅后台管理系统中其余模块的作用和应当具备的功能，如下图所示：

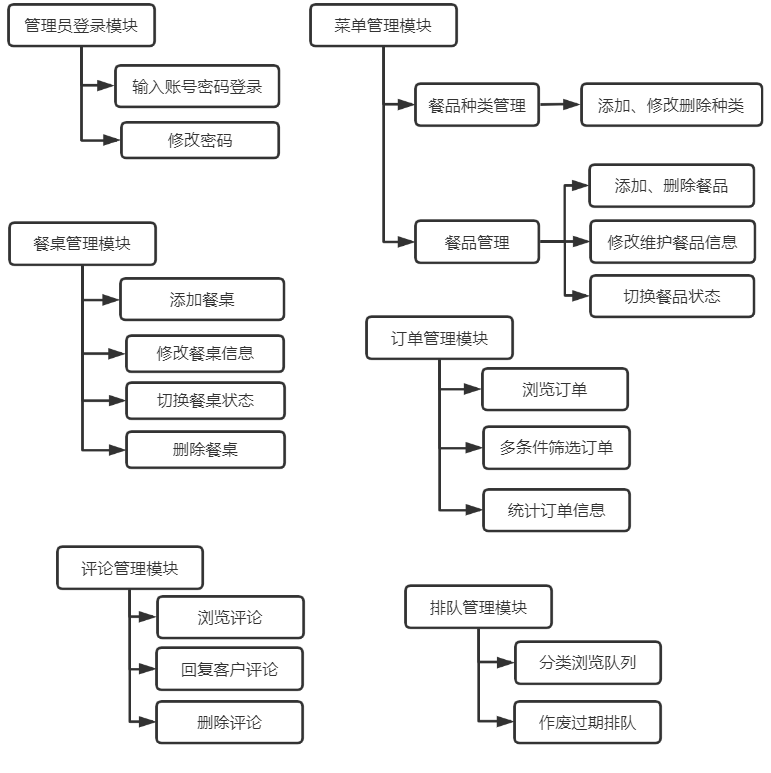


图4-6 后台管理系统其余功能模块

## 4.2数据库结构设计

本系统在数据的保存上选择采用MySQL数据库，并且将MySQL数据库运行在和后端服务器相同的Linux服务器上。关于数据库的版本，选择为5.1.49版本，之所以选择该版本，是因为该版本发布时间够早，使用人群最多，使用上最为稳定，可以省去很多开发部署上的麻烦。

### 4.2.1数据库概要

在MySQL服务器上新建数据库，命名为restaurant。进行如下配置和连接参数设置，并创建所需的数据表单，配置如下：

表4-1 数据库相关配置

|  |  |  |
| --- | --- | --- |
| 配置项 | 值 | 说明 |
| default-character-set | utf8 | Mysql客户端的默认字符集设置 |
| port | 3306 | 端口号设置 |
| driver-class-name | com.mysql.jdbc.Driver | 驱动名称 |
| engine | InnoDB | 数据库引擎模块 |
| map-underscore-to-camel-case | true | 匹配驼峰式命名 |

在数据库连接时，设定以下的参数：

表4-2 数据库连接参数

|  |  |  |
| --- | --- | --- |
| 参数名 | 值 | 说明 |
| useUnicode | true | 设置使用unicode的字符编码格式，配合下一个参数使用 |
| characterEncoding | utf-8 | 设置使用utf-8的格式将数据解析，数据变为字节码格式，最终转化为GBK格式存储在数据库中 |
| useSSL | false | 不建立SSL连接 |
| serverTimezone | Asia/shanghai | 设置数据库的时区为上海时区，避免日期在存取过程中出现偏差 |

数据库全部表单一览：

表4-3 表单注释

|  |  |
| --- | --- |
| 表单名称 | 注释 |
| admin | 管理员账号表 |
| carousel\_figure | 轮播图表 |
| food | 餐品表 |
| food\_type | 餐品种类表 |
| inform | 订单通知表 |
| line | 排队队列表 |
| order | 订单表 |
| remark | 评论表 |
| store\_info | 餐厅配置信息表 |
| table | 餐桌表 |

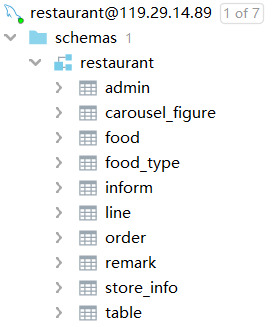


图4-7 数据库全部表单

### 4.2.1数据库表单结构

表4-4 管理员账号表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 长度 | 约束 | 说明 |
| id | varchar | <50 | 主键、非空 | 管理员账号 |
| password | varchar | <50 | 非空 | 管理员密码 |

表4-5 轮播图表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 长度 | 约束 | 说明 |
| id | int | 11 | 主键、非空、自增 | 为轮播图拟定一个id |
| url | varchar | <500 | 非空 | 轮播图片在服务器文件系统中的路径 |

表4-6 餐品表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 长度 | 约束 | 说明 |
| id | int | 11 | 主键、非空、自增 | 餐品的编号 |
| name | varchar | <50 | 非空、默认值为“未命名” | 餐品的名称 |
| price | double | 11 | 非空 | 餐品价格，含两位小数 |
| type\_id | int | 11 | 非空、外键 | 餐品类型ID，为外键，对应餐品类型表中的id属性 |
| image | varchar | <500 | 非空，默认值为空串 | 餐品图片地址 |
| monthlySales | int | 11 | 非空 | 餐品月售数量 |
| remark | varchar | <500 | 非空，默认值为空串 | 商品的备注，用于说明商品份量，口味等信息 |
| state | int | 11 | 非空，默认值为0 | 餐品的状态，有“在售”、“售罄”、“下架”三种状态，对应值分别为0，1，2 |

表4-7 餐品种类表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 长度 | 约束 | 说明 |
| id | int | 11 | 主键、非空、自增 | 餐品种类的编号 |
| name | varchar | <50 | 非空、唯一 | 餐品种类的命名 |

表4-8 订单通知表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 长度 | 约束 | 说明 |
| tableId | varchar | <50 | 组合主键、非空、外键 | 有新订单的桌台号，外键参考餐桌表中的id属性 |
| date | datetime |  | 组合主键、非空 | 通知的时间 |

表4-8 排队预约表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 长度 | 约束 | 说明 |
| phone | varchar | <20 | 主键、非空 | 排队用户的手机号 |
| serialNumber | int | 11 | 非空 | 所排队伍中的位置 |
| mealsNumber | int | 11 | 外键、非空 | 所排队伍类型，即就餐人数，为外键，参考餐桌表的seats属性 |
| date | datetime |  | 非空 | 排队开始的时间 |

表4-9 订单表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 长度 | 约束 | 说明 |
| id | int | 11 | 主键、非空、自增 | 订单编号 |
| date | datetime |  | 非空 | 订单创建时间 |
| state | int | 11 | 非空，默认值为0 | 订单状态：0（进行中,未评价），1（已完成,未评价），2（进行中，已评价），3（已完成，已评价） |
| phone | varchar | <20 | 非空 | 下单用户账号 |
| content | varchar | max | 非空 | 点餐内容，包含餐品被下单时刻的所有信息，为当时餐品信息的快照，以对象转JSON字符串的格式存储 |
| tableId | varchar | <100 | 非空 | 订单就餐的桌号 |
| nOfDiners | int | 11 | 非空 | 就餐人数 |
| mealFee | double | 10 | 非空 | 下单时刻，餐厅规定的每人餐位费。 |
| sum | double | 10 | 非空 | 订单总支付金额 |
| remark | varchar | <500 | 非空 | 食客填写的订单备注 |

表4-10 评论表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 长度 | 约束 | 说明 |
| id | int | 11 | 主键、非空、自增 | 评论编号 |
| order\_id | int | 11 | 非空、外键 | 评论对应的订单号，外键参考订单表的id属性 |
| date | datetime |  | 非空 | 发布评论的时间 |
| score | int | 11 | 非空 | 评分评价 |
| text | varchar | <5000 | 可以为空 | 文字评价内容 |
| images | varchar | <5000 | 可以为空 | 评价中图片的地址列表，为数组转JSON字符串存储。 |
| phone | int | 11 | 非空 | 评论的发布者 |
| reply | varchar | <5000 | 可以为空 | 商家对评论的回复 |

表4-11 餐厅配置信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 长度 | 约束 | 说明 |
| id | int | 11 | 主键、非空、自增 | 配置编号 |
| name | varchar | <500 | 非空 | 配置项名称 |
| value | varchar | <500 | 非空 | 配置项的值 |

表4-12 餐桌表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 长度 | 约束 | 说明 |
| id | varchar | <100 | 主键、非空 | 餐桌号码 |
| seats | int | 11 | 非空 | 该餐桌提供的座位数 |
| position | varchar | <500 | 非空 | 餐桌所在位置，如大厅、包间等 |
| state | int | 11 | 非空 | 餐桌的状态：“空闲”、“就餐”、“维修”分别对应0，1，2 |
| image | varchar | <500 | 非空 | 餐桌的图片地址 |
| remark | varchar | <500 | 可以为空 | 餐桌的备注说明 |
| orderId | int | 11 | 可以为空 | 餐桌对应的订单号，当餐桌为“就餐”状态时，记录该餐桌上的订单ID |

# 第5章 系统功能开发实现

## 5.1环境准备

根据先前概要分析的系统整体架构，项目要开发一个运行在云端linux系统的服务器上的后端服务系统，供小程序前端和后台管理前端与之进行交互。这三个端在本地window操作系统上更方便开发，功能开发完成后再部署到云端服务器上。

表5-1 开发、部署系统所需环境

|  |  |
| --- | --- |
| 开发环境 | win10系统 |
| 部署环境 | CentOS 8 |
| 开发语言 | Java 8 |
| 数据库 | Mysql v5.1.49 |
| 后端IDE | IDEA |
| 后台管理系统前端IDE | VSCODE |
| 微信小程序前端IDE | 微信web开发者工具 |
| 后端开发框架 | Spring Boot |
| 后台管理系统开发框架 | Vue、LayUI |
| 微信小程序开发框架 | Mina |

### 5.1.1 开发环境配置

在本地win10操作系统先后安装JDK8、IDEA、Maven、VSCODE、微信web开发者工具，并在微信公众平台注册小程序账号。

1. 后端Maven导入依赖：

表5-2 Maven依赖

|  |  |  |
| --- | --- | --- |
| 依赖项 | 版本 | 作用 |
| spring-boot-starter-web | 2.4.0 | Spring Boot创建基本web项目的依赖 |
| spring-boot-starter-thymeleaf | 2.4.0 | 使用Spring MVC 中的模板技术，达成路由映射静态资源页面 |
| shiro-spring | 1.4.0 | 引入Spring Shiro框架，实现权限控制，保证系统安全性 |
| lombok | 1.18.12 | 使用lombok插件，简化实例类的编写 |
| mybatis-spring-boot-starter | 2.1.4 | 引入MyBatis框架，实现与数据库的交互 |
| druid-spring-boot-starter | 1.1.17 | 配置druid数据库连接池，对数据库访问进行监控 |
| mysql-connector-java | 5.7 | mysql连接驱动 |
| fastjson | 1.2.47 | 提供Java对象与JSON字符串的相互转化功能 |
| bcprov-jdk16 | 1.46 | 微信用户授权模块 |

1. 后端配置application.yml文件，主要内容如下：

端口号设置：80

静态资源访问路径配置：classpath:/static/

设置文件上传中运行最大文件大小：100MB

设置前后端日期数据格式化：yyyy-MM-dd HH:mm:ss

配置数据源地址、数据库连接账号密码、数据连接池线程数目、mybatis映射文件地址、实体类地址等。

1. VSCODE环境
2. 配置LayUI项目目录结构

表5-3 LayUI目录结构

|  |  |
| --- | --- |
| 目录 | 作用 |
| api | 存放json数据，配置项目结构 |
| css | 存放页面的样式文件 |
| images | 存放页面中的图标、图片 |
| js | layui各模块的js组件库 |
| lib | layui、字体、jquery等库 |
| page | 首页内嵌页面HTML文件 |

1. 导入LayUI的方法：在html文件的script标签内，引入项目路径下layui.js文件。
2. 导入Vue的方法：script标签引入https://cdn.jsdelivr.net/npm/vue/dist/vue.js和https://cdn.jsdelivr.net/npm/vue-resource@1.5.1文件。
3. 小程序环境

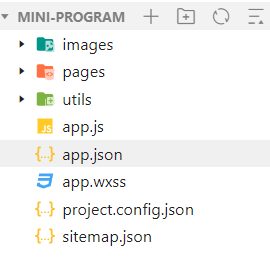


图5-1 小程序目录结构

在app.json中配置小程序名称，标题栏色彩，各页面路径等内容。在pages文件夹内存放各个页面的文件。

### 5.1.2 部署环境配置

1. 购买腾讯云轻量级应用服务器，配置开放3306端口（MySQL连接），80端口（后端服务连接），8888端口（宝塔面板接入）以及22端口（SSH登录连接）。



图5-2 腾讯云轻量级服务器防火墙

2. 安装JDK 8

3. 安装MySQL服务器，并实现第四章4.2节中数据库结构设计中的内容。

4. 购买域名[www.linqiu.ltd](http://www.linqiu.ltd)，配置服务器ip和域名进行绑定，以后就不需要记住ip地址来访问页面，直接输入域名网址即可。

## 5.2后台管理系统主要功能实现

前后端分离实现一个功能模块基本原理的主要步骤如下：

1. 后端创建对应模块主要操作对象的实体类A。
2. 后端编写关于该实体类对应数据库中表单的存取操作，将SQL语句编写在AMapper.xml文件中或者是以注解的形式编写在AMapper的Java类中。
3. 后端编写关于功能模块的业务，写在命名为IAService的Java接口类文件中，并用一个业务类实现该接口，在业务类调用Mapper提供的方法来处理该功能模块的业务逻辑。
4. 后端创建Controller类，在控制器设置http请求路由、请求参数、返回数据格式等内容，并调用业务类为不同的路由执行不同的业务，业务层则调用持久层完成对数据库的存取访问。
5. 前端编写功能模块的页面，并通过向后端发送http请求获取前端页面需要显示的内容。
6. 前端接收用户操作动作、用户输入数据，根据不同的用户操作类型，选择执行对应的业务，向后端发送对应的业务路由，待后端接收处理，处理结构打包发送给前端，前端对处理结果进行解析，将处理结果回显给用户。

### 5.2.1 管理员登录模块

1. 业务流程

管理员输入网址[www.linqiu.ltd](http://www.linqiu.ltd) 进入餐厅的后台管理系统，前端系统首先对浏览器内的cookie进行检查，若存在登录状态的cookie，则读取信息自动发送登录请求，后台处理登录请求，登录成功后成功进入主页，登录不成功则跳转至登录页面；若不存在该cookie，则前端跳转到登陆页面。在登陆页面中，手动输入账号、密码，再发送登录请求，将前端输入的账号密码传入后端控制层，控制层接收数据参数，判断账号和密码的正确性，将结果封装返回给前端，前端对返回体进行解析，若账号密码正确则进入系统主页，并且将登录信息存到浏览器本地cookie中，方便下次自动登录。

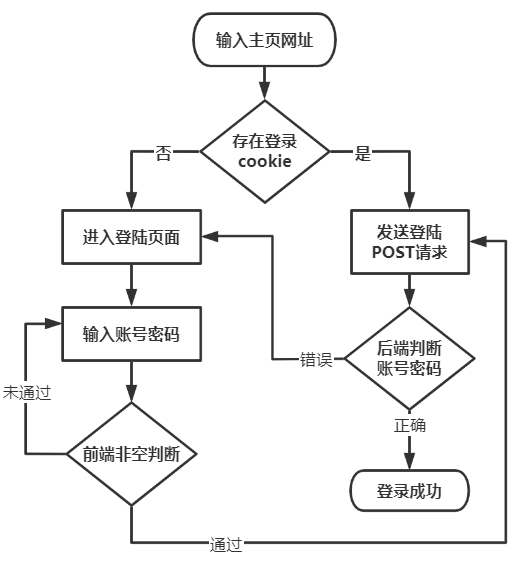


图5-3 管理员登陆流程图

1. 关键函数方法

表5-4 前端部分函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| getCookie | 从浏览器缓存中根据名称获取cookie信息，用于获取缓存的登录状态信息，实现自动登录。 |
| setCookie | 向浏览器缓存中添加cookie，同时设置cookie名称和存活时间。 |
| onClickLogin | 执行登录操作，首先进行前置输入检查，检查账号密码是否为空，检查通过后才向后端发送“/admin/login”请求。获取后端返回结果后，若账号密码正确则进入主页并保存cookie。 |

表5-5 后端部分函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| AdminController.login | 控制层方法，接收前端发送的“/admin/login”请求，获取账号密码数据，并传给业务层的login函数进行处理。 |
| AdminService.login | 业务层方法，接收控制层传来的账号密码参数，调用持久层查询数据库，根据id查询admin表，获取admin实例。这过程中判断账号是否存在、密码是否正确。 |
| AdminMapper.selectById | 持久层方法，用于查询数据库的admin表，根据admin的id字段获取admin对象。 |

1. 实现结果

图5-4 登录失败显示提示



图5-5 登录成功进入主页

1. 该模块其余功能简述
2. 退出登录：

在主页右上角点击管理员账号名称，选择退出登录，即从主页退出到登录页面，并且将浏览器中的“loginInfo”cookie删除。

1. 修改密码：

在主页右上角点击管理员账号名称，选择修改密码，主页内会打开修改密码的tab标签页，输入账号、新旧密码发送到后端，即可执行修改密码的业务逻辑，成功修改后将新的密码更新到数据库，前端自动关闭修改密码界面。

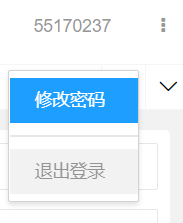
 

图5-6 管理员退出登录和修改密码功能页面

### 5.2.2 菜单管理模块

菜单管理模块主要分为两个部分：餐品种类管理和餐品管理。

管理员成功登录系统进入主页后，点击菜单管理选项卡，展开餐品种类管理和餐品管理两个选项。点击不同选项在主页分别打开餐品种类管理和餐品管理的tab层级页面。餐品种类页面展示所有餐品种类，并且按餐品种类的id顺序排列，由于餐品种类的id为自增字段，即也是按创建时间顺序。餐品页面展示所有餐品的列表，也是按照id顺序排列。由于使用了layui框架，也可以点击列表的表头指定字段，根据指定字段进行排序。



图5-7 餐品种类管理页面

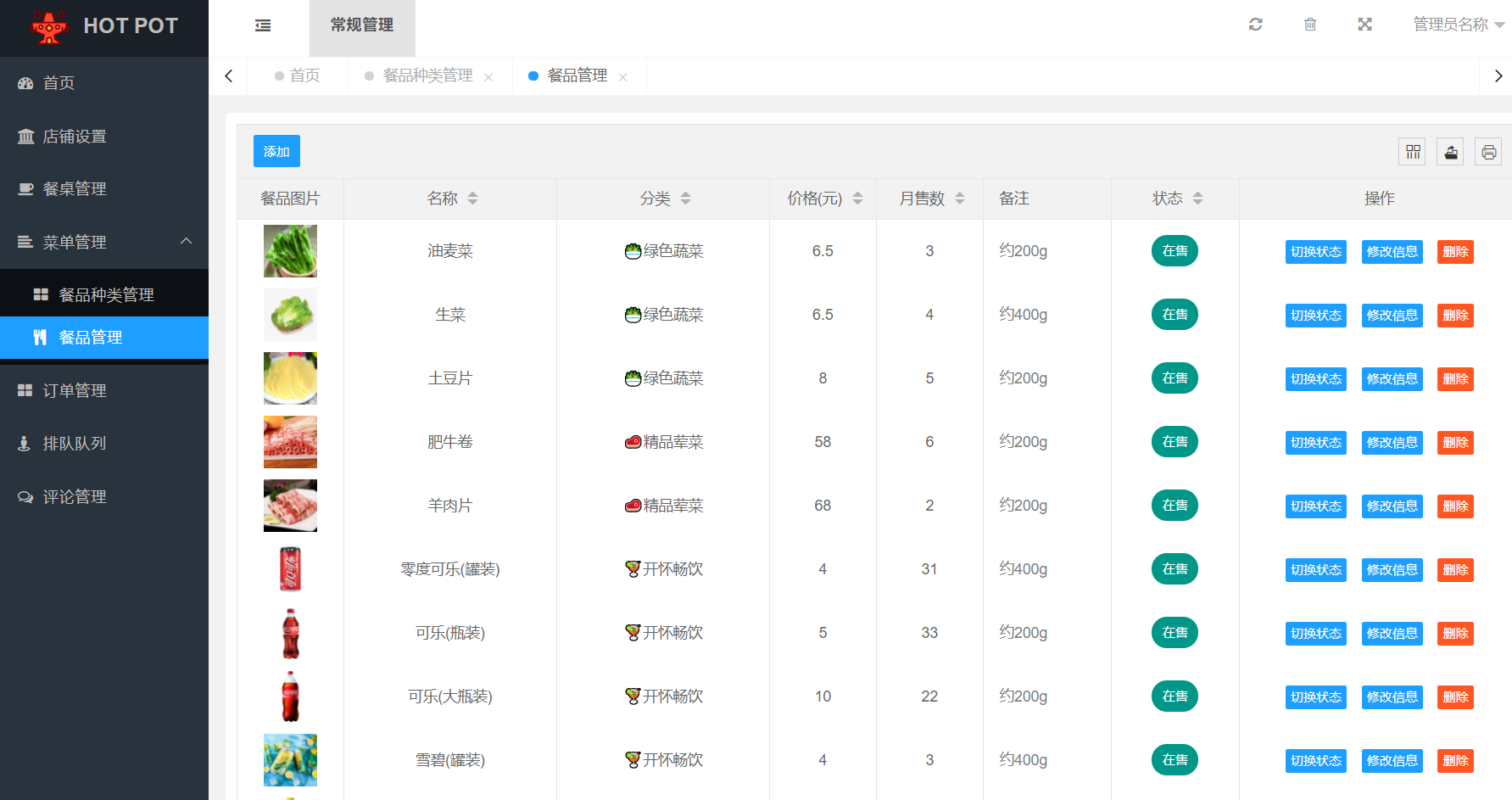


图5-8 餐品管理页面

1. 各功能的业务流程
   1. 餐品种类添加：前端输入餐品种类名称即可确认添加，后端收到数据后判断名称是否满足唯一性约束，若出现重复的名称，则添加失败。
   2. 餐品种类删除：选择要删除的餐品种类那一行的删除按钮，即向后端传递该菜品种类的id，后端查询数据库，判断餐品中是否存在仍使用该餐品种类id的餐品，若不存在，则成功删除；若存在，则拒绝删除，前端显示原因。
   3. 餐品种类修改：点击修改后输入新的名称，后端也要检查唯一性约束。
   4. 餐品添加：输入餐品名称、价格、备注、图片，在已存在的餐品种类中选择餐品种类，点击保存按钮，将数据发送到后端，后端新建一个Food对象，将数据封装到Food中并插入到数据库内。
   5. 餐品修改：点击餐品列表的修改按钮，修改餐品信息，提交新的数据内容到后端，后端执行修改餐品的业务逻辑，并将数据更新到数据库。
   6. 餐品状态切换：点击餐品列表的切换状态按钮，前端弹出选择状态窗口，根据当前状态，显示另外两个状态，点击选择其他状态后，发送餐品id和新状态到后端，后端执行更新餐品状态的业务逻辑。

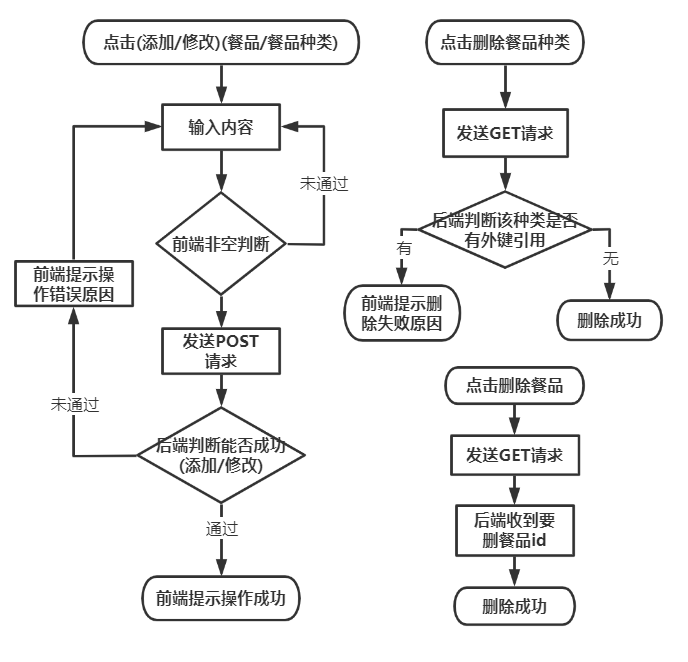


图5-9 菜单管理模块部分功能流程

1. 关键函数方法

表5-6 前端部分函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| table.render | LayUI的库函数，传入一个GET请求路径，例如获取全部食物的路径“/food/all”，即可获得数据库中所有的餐品表单项，并更具字段名显示在页面的列表上。同时可为每列添加操作按钮（修改、删除等），使得点击行级按钮时，同时可获得该行的数据对象。在需要修改、删除指定行的项目的时候，可直接获取其id，作为参数发送对应业务请求。 |
| addFoodType | 函数内调用layer.open函数，打开添加餐品类型的页面。 |
| addFoodType.submit | 添加餐品类型页面的提交函数，前置非空判断通过后，将餐品类型名称作为表单内容，向后端发送“/foodType/add”请求，若返回成功，则提示添加成功，反之显示“名称重复”的提示框。 |
| editFoodType | 调用layer.open函数打开修改餐品种类的页面，并将旧的餐品种类名称传入该页面。 |
| editFoodType.submit | 向后端发送“/foodType/eidit”的POST请求已修改餐品类型信息。 |
| addFood | 打开添加餐品页面。 |
| addFood.submit | 新增的餐品信息提交到“/food/add”POST请求。 |
| editFood | 打开修改餐品页面。 |
| editFood.submit | 将新修改的餐品形象提交到“/food/edit”POST请求。 |
| switchFoodState | 修改餐品的状态。若餐品本身时“在售”状态，那么只提供另外两种状态选项“售罄”、“下架”。售罄的在小程序端无法加入购物车，下架的餐品不会在小程序页面显示。点击选项后发送“/food/switchState”的POST请求。 |

表5-7 后端部分函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| FoodTypeService.add | 处理添加餐品类型业务：收到要添加的FoodType对象作为参数，先更具名称查询数据库，若查询结果不为空，则返回添加失败信息，否则执行插入语句，添加新的餐品类型。 |
| FoodTypeService.edit | 处理修改餐品类型业务。将更新的餐品种类信息写入数据库。 |
| FoodTypeService.delete | 删除指定餐品类型，删除之前根据餐品类型id查询餐品表，若有结果，则拒绝删除该餐品类型，否则删除。 |
| FoodService.add | 处理餐品添加业务，若传来的餐品对象无图片地址，则查询店铺信息表，获取默认图片地址，设置为该餐品图片。 |
| FoodService.edit | 修改指定餐品，若未指定图片地址，同理设置为默认图片地址。 |
| FoodService.switchState | 处理修改餐品状态业务。 |
| FoodService.all | 获取所有餐品对象，存在数组中，传递给前端。 |
| FoodTypeService.all | 获取所有餐品类型对象，传递给前端。 |

1. 实现结果



图5-10 添加餐品种类

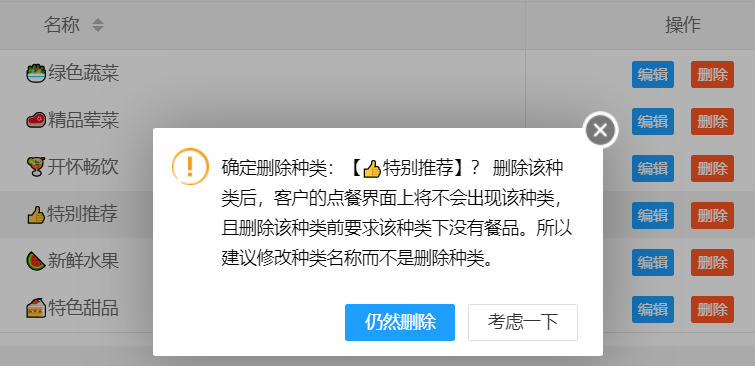


图5-11删除餐品种类

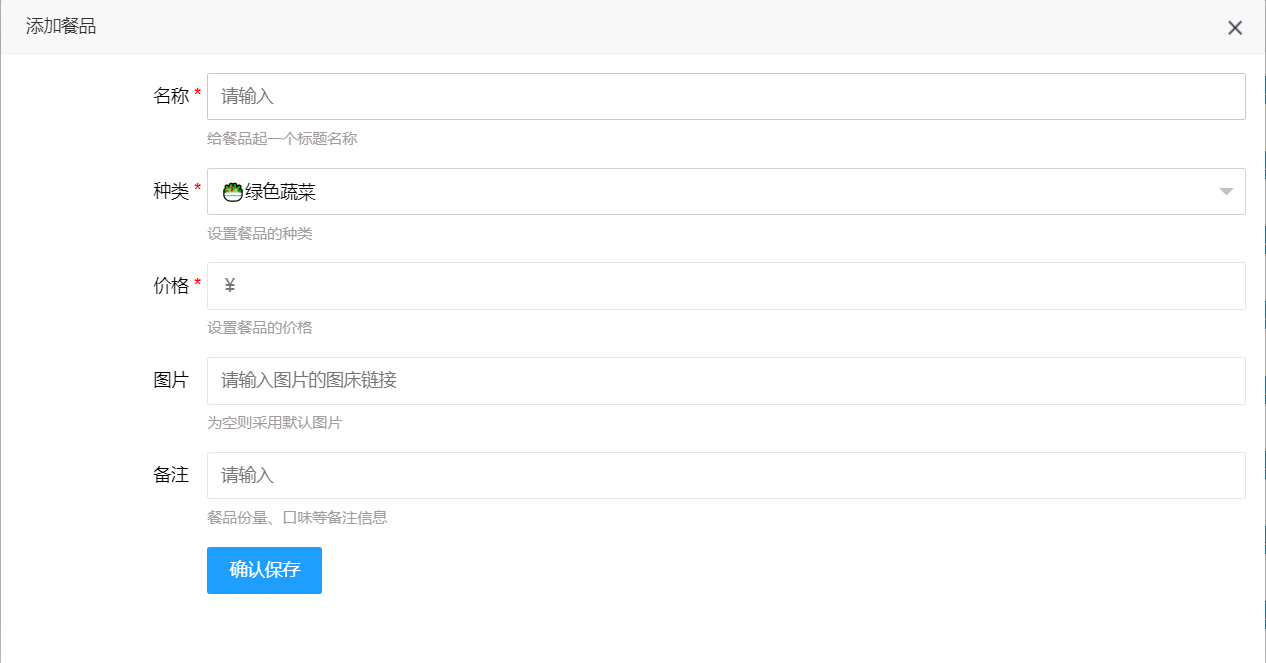


图5-12添加餐品



图5-13切换餐品状态

### 5.2.3 餐桌管理模块

餐桌管理模块主要功能为添加、修改、删除餐桌和切换餐桌状态。界面如下图所示：



图5-14餐桌管理页面

1. 各功能的业务流程
   1. 添加新餐桌：输入餐桌号、座位数、位置，可选输入图片、备注。点击提交按钮。若餐桌号不违反主键约束，即可添加成功，否则添加失败。
   2. 编辑餐桌：输入新的座位数、位置、图片、备注等信息，注意桌号不可修改，点击提交按钮。
   3. 删除餐桌：点击删除按钮，仍存在未完成订单的餐桌不允许删除。
   4. 切换餐桌状态：点击切换状态，选择新状态进行切换，对于正在用餐状态的餐桌，必须将订单结束后才能切换状态。
   5. 全部启用餐桌：点击全部启用按钮，将所有“维修”状态的餐桌切换为“空闲”状态。
2. 模块内关键函数方法

表5-8 前端部分函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| addTable.submit | 添加餐桌，携带餐桌信息发送“/table/add”的POST请求。 |
| editTable.submit | 修改餐桌，携带新餐桌信息发送“/table/edit”POST请求。 |
| switchState.submit | 切换餐桌状态，携带餐桌id和状态发送“/table/switchState”的POST请求。 |
| enableAll | 启用全部餐桌，向服务器发送“/table/enableAll”GET请求。 |

表5-9 后端部分函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| TableService.add | 添加餐桌，若餐桌号重复则添加失败。 |
| TableService.modifyById | 修改餐桌，根据餐桌id来决定修改哪个餐桌信息。 |
| TableService.switchState | 切换餐桌状态，根据餐桌id来决定修改哪个餐桌，换成前端传入的状态值。无法切换原来为“就餐”状态的餐桌。 |
| TableService.enableAll | 查询数据库，将所有状态为“维修”（值为2）的餐桌修改为“空闲”状态（值为1）。 |

1. 实现结果



图5-15 添加餐桌

图5-16 删除餐桌和修改餐桌状态

### 5.2.4 排队管理模块

考虑到餐厅存在生意火热时餐位不足的情况，系统需要提供一个排队的功能，餐厅的食客在微信小程序端选择不同的餐桌类型（例如2人桌、4人桌、6人桌）进行排队（小程序端的排队流程会在下文小程序端业务实现过程部分介绍），则会产生多个排队队列，后台需要了解各个队列的目前状态，必要时进行管理，于是后台管理系统的排队管理模块便被设计出来。页面如下：

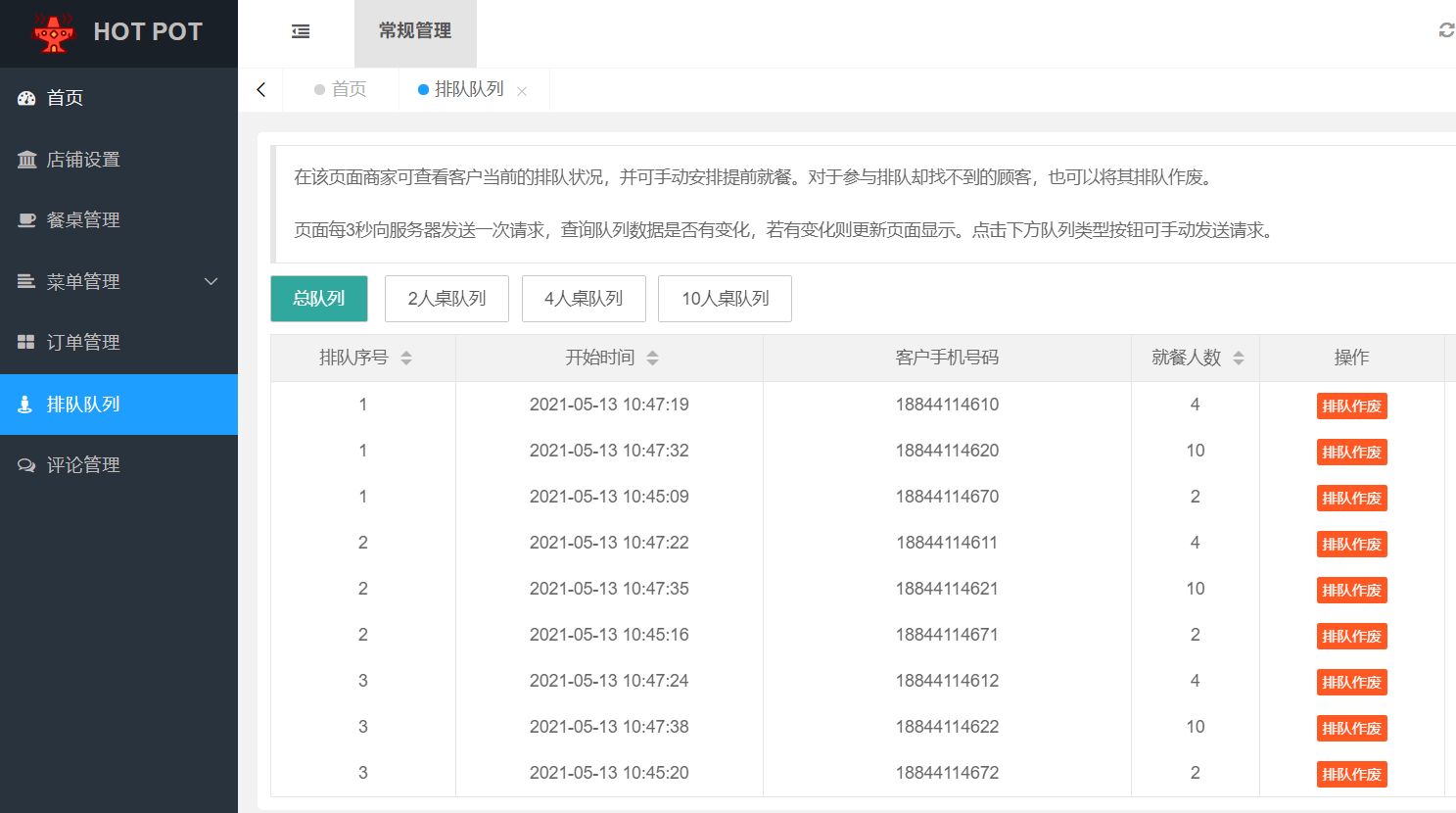


图5-17排队管理页面

1. 各功能的业务流程
   1. 分不同队列查看：进入排队管理页面后，点击具体队列的选项，则列表内容切换为对应队列。
   2. 排队作废：点击排队作废按钮，即可将该用户从队列中删除。
2. 关键函数方法

表5-10 前端部分函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| line.init | 排队页面初始化方法，先向后端发送“/line/all”GET请求，获取所有排队队列，随后发送“/line/getMealsNumberOption”请求，获取所有队列类型，每个队列类型渲染成一个按钮。 |
| line.filter | 发送“line/all/{mealsNumber}”GET请求，获取不同类型的排队队列。 |
| line.delete | 发送“/enqueue/cancel”POST请求，取消指定用户的排队任务。 |
| line.updateData | 排队信息是时刻可能变化的，当有食客排到队伍第一位，扫码就餐后，食客的排队会自动作废，管理后台则需要对排队列表内容进行更新。该函数开启一个定时任务，每3s向后台请求一次排队信息数据，将得到的新数据和旧数据比较，若出现变化，则重新渲染排队列表表格。 |

表5-11 后端部分函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| LineService.getNumberOption | 用distinct关键字查询数据库餐桌表的seats字段，并从小到大排序，即得到了所有不同的餐桌类型，即是排队类型。 |
| LineService.cancelEnqueue | 取消指定用户的排队，实现方式如下：首先进入方法后上线程锁，避免出现脏读、幻读，然后根据用户手机号查询该排队信息，删除该条排队信息，并更新该排队信息对应的排队类型队列，将所有序号大于被删除排队信息的项的序号减少1，即更新整个队伍状态为正确状态。方法最后，释放线程锁。 |

1. 实现结果



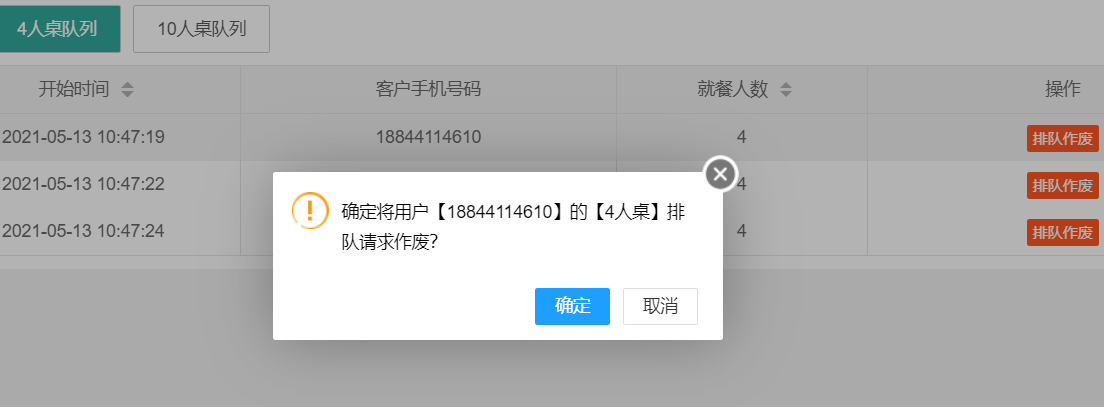


图5-18 管理后台排队管理

### 5.2.5 订单管理模块

订单管理主要用于管理员对历史订单的查阅统计，主要业务为多条件查询筛选订单。在主页点击订单管理选项卡，打开页面如下图：



图5-19 订单管理页面

1. 各功能的业务流程
   1. 查看订单：管理员进入订单管理页面后，页面展示订单列表，列表中展示订单的简要信息：编号、时间、状态、总金额等，每条订单后提供一个“详情”按钮，点击后打开订单的详情信息页面，详情信息页面应当额外包括点餐内容图片、客户备注、额外收费内容等信息。
   2. 条件搜索：点击条件搜索，展开各条件的输入框和选择框，输入筛选条件后，在页面显示多条件综合筛选的结果列表。各条件处理方式如下表：

表5-12 筛选条件处理方式

|  |  |
| --- | --- |
| 字段 | 筛选方式 |
| 订单状态 | 等于选定状态 |
| 订单编号 | 等于输入内容 |
| 下单日期 | 在选择日期的全天内 |
| 客户手机号 | 包含输入内容 |
| 点餐内容 | 包含输入内容 |
| 就餐桌号 | 等于输入内容 |
| 用餐人数 | 等于输入内容 |
| 订单金额 | 等于输入内容 |
| 备注信息 | 包含输入内容 |

* 1. 统计总金额：勾选需要加入统计的订单，点击统计按钮，计算出订单的总金额，弹出通知。

1. 关键函数方法

表5-13 前端部分函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| order.init | 订单管理页面初始化，发送“/order/all”请求，获取所有订单。 |
| order.detail | 打开订单详情页面，传递订单号作为参数。 |
| orderDetail.init | 订单详情页面初始换，发送“/order/{id}”请求，根据id获取指定订单。 |
| order.filter | 收集订单筛选参数，携带参数发送“/order/all”请求，获取多条件查询结果，并重新渲染订单列表。 |
| order.statistics | 点击统计按钮后，检测被勾选的订单项目，将所有被勾选的订单项的总金额相加，得到结果通知到页面。 |

表5-14 后端部分函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| OrderService.all | 业务层方法，获取全部订单内容。 |
| Order(Map) | Order对象的一个新的构造函数，用一个Map对象作为参数构建。原因是前端传来多条件筛选参数的值全部都是字符串类型，保存在Map中，而数据库查询时需要使用正确的数据类型进行查询，因此在构造函数中对数据类型进行转换，获得一个正确的Order对象。 |
| OrderService.getByConditions | 业务层函数，使用多条件构造的Order对象作为参数，传递给持久层进行查询，获取结果集合。 |
| OrderMapper.selectByConditions | 持久层函数，用Mybatis的动态sql语句进行查询，每个字段首先用if节点的test属性进行非空判断，若存在值，则会被作为查询条件加入到where子句当中。对于不同的字段名，采取不同的判断依据，例如“nOfDiners”字段，则采取“=”的判断条件，而“content”字段则采取“like”的判断条件并用两个“%”包围判断值。 |

1. 实现结果

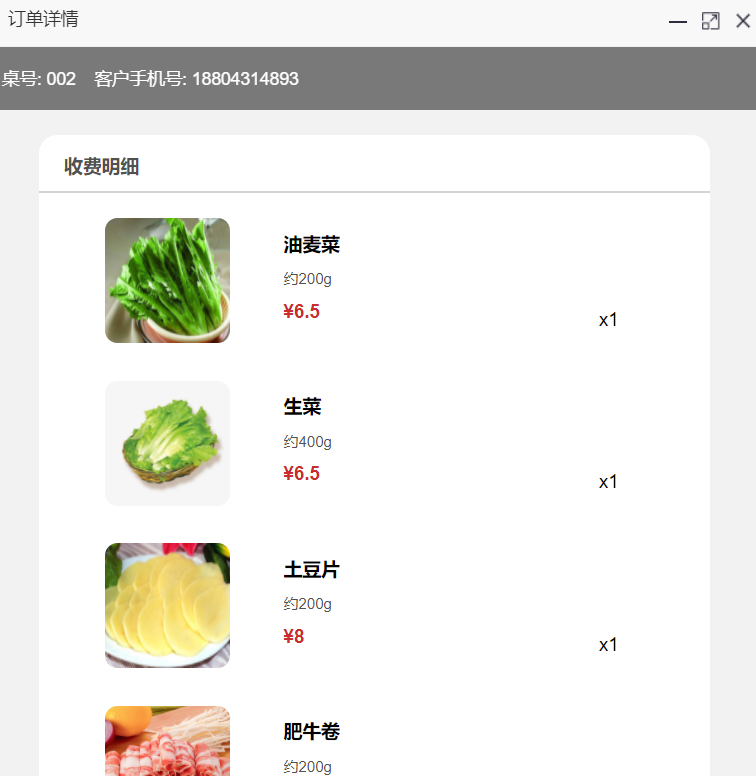
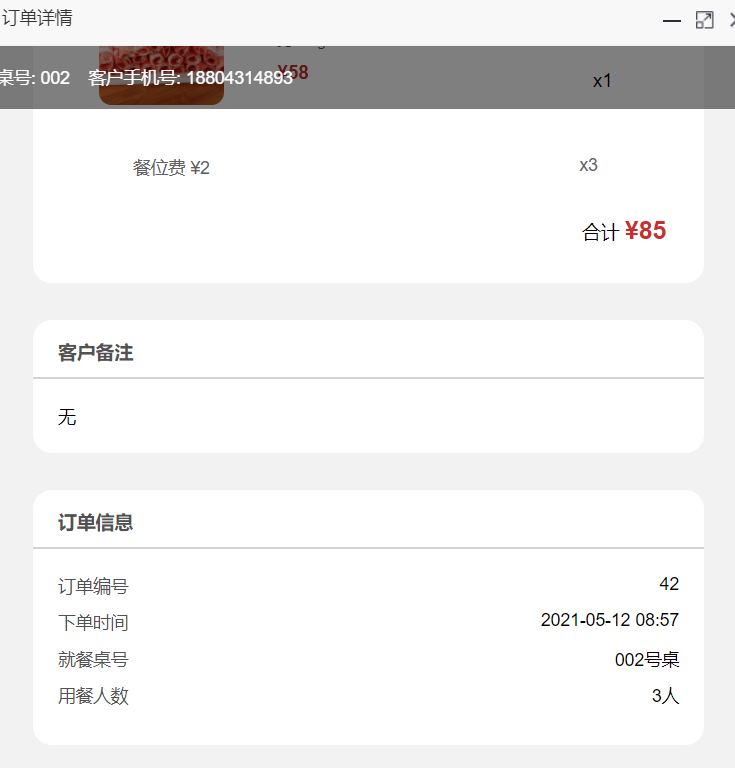
 

图5-20 订单详情页面



图5-21 订单多条件查询结果

### 5.2.6 评论管理模块

评论管理页面如下图所示：

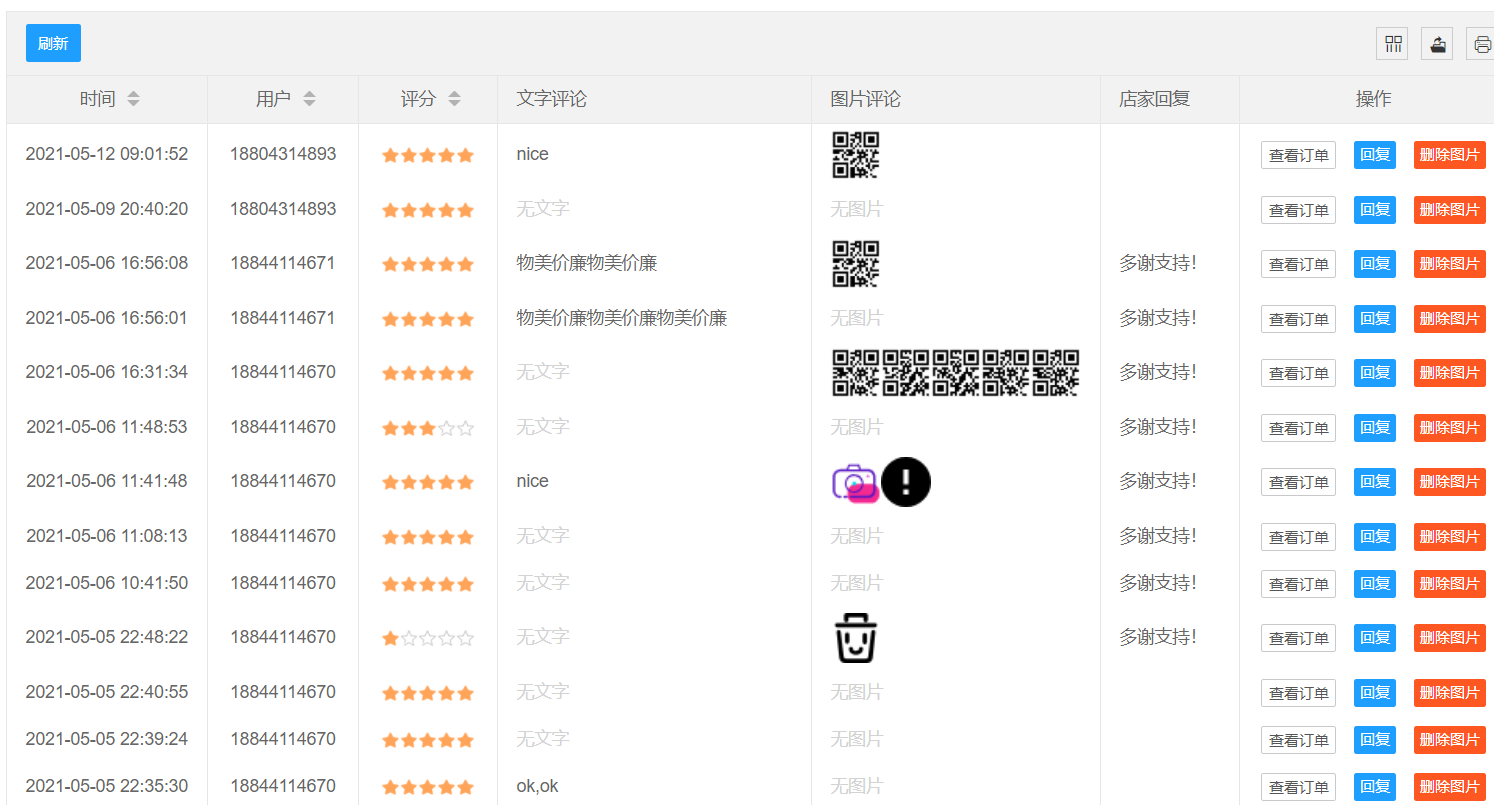


图5-22 评论管理页面

该模块主要用于管理员浏览顾客对就餐体验的评价，并可对评价做出商家回复，此外由于评价中存在图片信息，为了避免有顾客恶意上传暴力、血腥、色情等违规图片，为管理员提供删除评论图片的功能。

1. 各功能的业务流程
   1. 浏览评论：点击评论管理选项，即可打开评论列表页面，列表包含评论时间、用户、评分星数、文字评论内容、评论图片、店家回复等内容。
   2. 查看订单：点击查看订单，即可根据该评论对应的订单号，打开OrderDetail页面，该页面的相关实现已经在订单管理模块介绍。
   3. 回复评论：点击回复按钮，商家输入回复内容并提交即可恢复食客评论。
   4. 删除图片：选择要删除图片的评论，点击删除图片。
2. 关键函数方法

表5-15前端部分关键函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| remark.init | 页面初始化，发送“/remark/all”请求获得所有评价。 |
| remark.openOrderDetail | 打开订单详情页面，传入订单id作为参数，在订单详情页面的初始化中，将订单id作为参数发送“/order/{id}”请求，获取订单详情内容，并渲染到页面上。 |
| remark.reply | 商家回复，打开回复内容输入框，提交后将输入框的内容作为参数发送到“/remark/reply”请求。 |
| remark.delete | 商家删除评论图片，点击后将评论的id作为参数，发送“/remark/delete”请求。 |

表5-16后端部分关键函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| RemarkService.all | 业务层函数，向持久层请求查询数据库，获取评论表单的全部评论。 |
| RemarkService.deleteImage | 处理删除评论图片业务，根据指定id更新评论表单，删除评论图片字段的值。 |
| RemarkService.reply | 商家回复业务，根据指定id更新评论表单的商家回复字段内容。 |

1. 实现结果

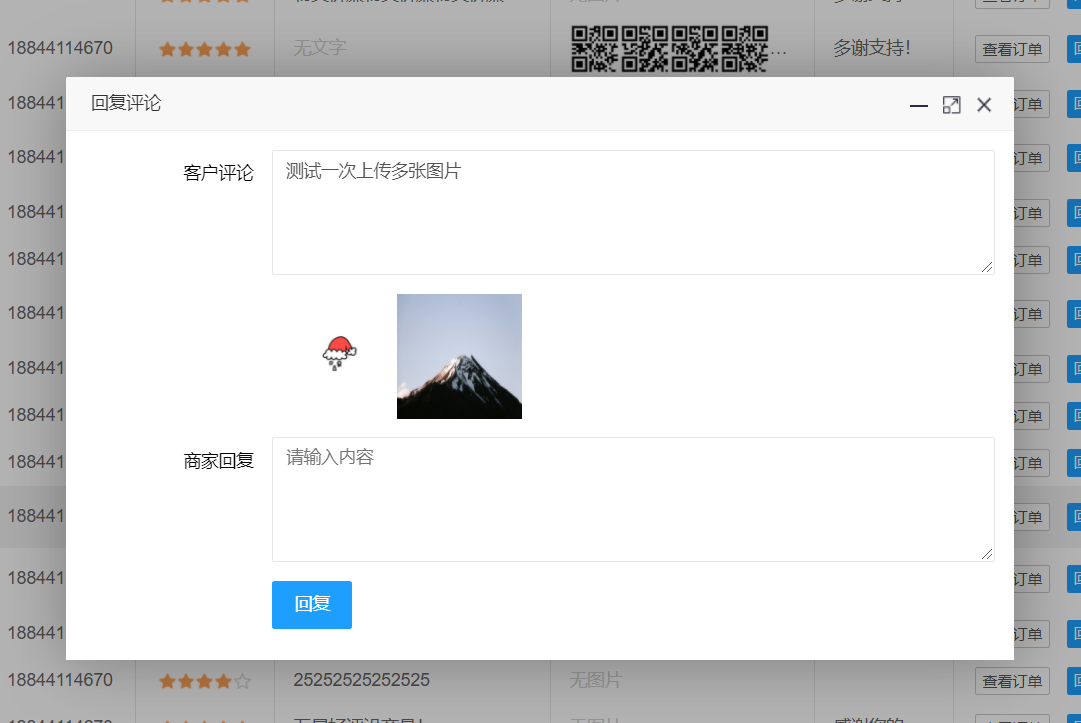


图5-23 商家回复评论

### 店铺设置模块

1. 各功能的业务流程
   1. 基本信息设置：点击进入店铺设置页面，在输入框内修改配置项内容，点击保存按钮，向后端发送更新请求。可设置的配置项有：店铺名称、店铺地址、联系电话、餐位费。
   2. 上传默认餐桌、餐品图片：点击更新图片按钮，在文件系统选择图片上传到服务器端，即可更新默认图片。
   3. 上传、删除小程序首页轮播图片：点击添加按钮，选择图片上传，若已上传图片大于五张，则拒绝上传。点击删除按钮，删除先前上传过的轮播图片。
2. 关键函数方法

表5-17前端部分关键函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| storeInfo.init | 店铺信息页面初始化。向后端发送“/storeInfo/all”请求，获取到店铺信息数组后，根据storeInfo.name展示配置名称，storeInfo.value给输入框、图像地址填入原始值。 |
| carouselFigure.all | 发送“/scarouselFigure/all”请求，获取全部轮播图的地址，渲染轮播列表。 |
| storeInfo.edit | 修改店铺信息，将店铺信息以Map集合数据格式发送到后端的“/storeInfo/edit”请求。 |
| carouselFigure.add | 根据轮播图列表的大小判断图片数量，若小于5张，则选择文件进行上传，并发送“/carouselFigure/add”请求。 |

表5-18后端部分关键函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| StoreInfoService.edit | 店铺信息修改业务，根据配置名称查询数据库，修改对应的配置值。 |
| StoreInfoService.all | 查询数据库中的“storeInfo”表单，获取全部配置项的名称和值。 |
| FileConfig.addResourceHandlers | 在@Configuration注解的类中实现WebMvcConfigurer接口，类中内实现该接口方法，该方法用于配置本地资源映射路径，配置该方法后，可以通过网址获取linux服务器文件系统中的文件，包括图片。可以将该网址+文件路径作为前端img标签src属性的值，即可直接将图片嵌入到网页内。 |
| FileUtil.save | 编写的工具方法，用于保存http请求中带有的MultipartFile类型的文件对象，转化为File对象，并调用File.transferTo方法将文件写入到linux服务器的文件系统内，并返回其所在路径和文件名。 |
| StoreInfoService.default(Food/Table)Image | 业务层方法，接收从控制层传来的餐桌/餐品默认图片文件，调用FileUtil.save函数保存到文件系统，并获取到路径的字符串，将字符串的值更新到数据库中。 |
| CarouselFigureService.add | 用于处理添加轮播图业务，接收控制层传递的轮播图片文件，首先查询数据库，计算轮播图片数量，若小于5张，则调用FileUtil.save函数保存图片，并将路径写入到数据库。 |

1. 实现结果

图5-24 设置店铺基本信息和默认图片



图5-25 设置小程序首页的轮播图

### 桌台模式模块

桌台模式模块是后台管理系统中相对来说最为重要的模块，该模块整合了很多部分的功能，包括餐桌状态的实时监控，餐桌的分类、门店信息的展示、快捷功能、开关店铺按钮、桌台数据的统计、营业额的统计。并且桌台模式作为后台管理系统的主页使用，页面如下：



图5-26 桌台模式

本模块涉及功能较多，选择主要的业务讲解：

1. 各功能的业务流程
   1. 门店信息展示及维护功能：前端显示门店信息卡片，包括门店图片、名称、状态。点击卡片弹出店铺设置的小页面，可以维护店铺信息。
   2. 快捷功能：点击叫号、账单分别弹出排队队列和账单管理的小页面，进行快捷管理。点击开关店按钮可以修改店铺当前状态，若店铺关闭，则前端小程序无法下单。
   3. 桌台数据统计：对当前店铺各餐桌状态的统计，让管理员直观的看到不同的状态餐桌数量。
   4. 营业额统计：统计不同时间段的营业额。
   5. 新订单通知：当客户在小程序下单支付后，后台弹出新订单的通知，并给出桌号信息和时间信息。
   6. 实时监控各餐桌的状态：页面执行定时任务，每3s从后台获取一次桌台状态的最新数据，若数据有变化，则渲染桌台卡片页面为新的数据，并且刷新桌台数据卡片和营业额信息卡片的值。
   7. 筛选显示餐桌：根据餐桌所在位置、餐桌容量、餐桌状态多条件筛选餐桌，显示到页面上。
   8. 结算账单：点击用餐状态的餐桌，弹出订单的详情页面，并且提供一个结账按钮，点击结账按钮，将这台餐桌修改为空闲状态，订单修改为已完成状态，并获取最新的页面数据，包括餐桌状态统计、营业额统计等。
2. 关键函数方法

表5-19前端部分关键函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| getStoreInfo | 发送“/storeInfo/all”请求，获取店铺信息。 |
| getTableStateCount | 发送“/table/stateCount”请求，获取餐桌状态统计信息。 |
| getOrderCount | 发送“/order/count”请求，获取营业额统计信息。 |
| getTableFilter | 发送“/table/classify”请求，获取餐桌的筛选条件信息。 |
| getInform | 发送“/inform/all”请求，获取通知列表。 |
| tableFilter | 传入前端餐桌数组的索引值，根据索引值获取餐桌对象，根据当前的筛选条件，判断该餐桌对象的位置、状态、容量，全都符合筛选条件时，则返回true，将该函数的返回值取反应用到hidden属性上，实现对餐桌的筛选。 |
| updateData | 定时任务，每3s向后端发送一次“/table/all”请求，获取所有餐桌数据，与本地的旧数据进行JSON字符串化的比对，若存在不同，则更新数据，并发送其他获取数据的请求更新餐桌状态统计信息、营业额统计信息、新订单通知信息等。 |
| orderDetail.tally | 发送“/order/tally”请求，传递订单号和餐桌号用于结算订单，结算后订单状态为“已完成”，餐桌状态为“空闲”。 |

表5-20后端部分关键函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| \*Service.all | 某业务的all函数，都代表从数据库获取某表全部内容。例如InformService.all，StoreInfoService.all等。 |
| TableService.stateCount | 餐桌状态统计业务，从数据库获取全部餐桌，计算不同状态的餐桌的数量。 |
| OrderService.count | 订单营业额统计业务，从数据库根据时间段查询订单，并统计时间段内的营业总额。 |
| OrderService.tally | 订单结算业务，根据订单id修改订单状态为“完成”，更具餐桌id修改状态为“空闲”。 |

1. 实现结果



图5-22 结算订单



图5-27 餐桌筛选

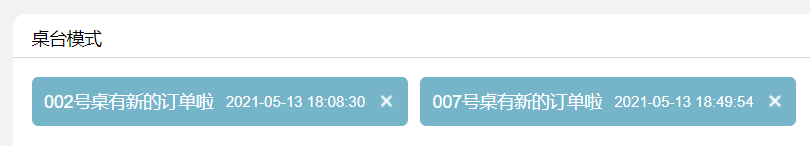


图5-28 新订单提醒

## 5.3微信点餐小程序主要功能实现

微信小程序运行在安装有微信应用的任何移动设备上，拥有极大的跨平台便携性。微信通过http请求与后端服务器进行交互，常用发送请求的函数如下：

表5-21微信与后端交互的API函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| wx.request | 最基本的发送HTTP请求方法，可指定GET、POST请求类型。 |
| wx.uploadFile | 微信发送上传文件请求的方法。 |

根据第四章概要设计中的分析，点餐小程序主要包含用户登录、扫码点餐、排队预约、餐厅评价、订单五大模块。下面简要介绍各模块的前后端实现过程。

### 5.3.1 用户登录模块

使用微信客户端扫描本点餐小程序的二维码，进入小程序首页。首页包含一个轮播图组件和五大功能按钮，只有“餐厅评价”功能允许食客预览，其余三个功能需要执行“授权登录”后才能使用。

1. 各功能的业务流程
2. 轮播图播放：进入首页后，展示商家设置的轮播图，自动播放，最多五张。
3. 授权登录：点击授权登录按钮，通过微信授权获取手机号码，作为客户的账号ID登录小程序系统，若未登录就使用其他功能弹出请先登录的提示。
4. 关键函数方法

表5-22前端部分关键函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| checkLogIf | 判断缓存中是否有用户登录信息 |
| getPhoneNumber | 获取用户授权获取手机号码，向后端发送“/user/oauth”请求，携带encryptedData、iv、codes作为参数，成功获取后执行登录操作，并将获取的手机号写在缓存中。 |

表5-23后端部分关键函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| UserController.wxOauth | 处理“/user/oauth”请求，收到微信客户端发来的三个参数，并配置小程序id、密钥，使用微信提供的bcprov工具对参数进行解码，获取手机号并返回给前端小程序。 |

1. 实现结果

图5-29 小程序用户登录模块

### 5.3.2 扫码点餐模块

该模块包含半个点餐流程：扫码二维码获取桌号、进入餐单页面、将餐品加入购物车、生成订单。

1. 各功能业务流程
2. 进入菜单：点击扫码点餐，扫描餐桌上的二维码，获取餐桌号，若该餐桌空闲，则进入菜单，否则打开进行中的订单或者餐桌已占用的提示。
3. 选择就餐人数：进入菜单页面后，必须先选择用餐人数方可继续点餐。
4. 浏览餐单：左边选择菜品类别，右边显示菜品信息。售罄状态的餐品将提示售罄，并无法加入购物车，下架状态的餐品不会在菜单上显示。
5. 购物车管理：选择餐品点击加号加入购物车1份，点击减号减少1份，并且在类别、购物车按钮上显示已选数量、总金额。点击购物车图片则升起购物车页面，显示已点内容，并提供清空购物车按钮。
6. 生成订单：当购物车中有内容时，点击“选好了”按钮，根据购物车内容跳转至订单页面，并显示点餐餐品和数量以及总金额。
7. 关键函数方法

表5-24前端部分关键函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| wx.scanCode | 扫描二维码，获取桌号，发送“/table/{id}”请求，获取餐桌信息，若该餐桌状态为“空闲”，则进入菜单页面；若为“就餐”状态，则判断就餐用户是否为本用户，若不是，则提示无法点餐，若是则打开该订单详情页面。 |
| menu.onLoad | 发送“/food/front/menu”请求，获取订单内容。 |
| menu.filter | 根据所选餐品种类，显示不同的餐品。 |
| menu.handleCartBtnTap | 处理餐品加减按钮的点击事件，若点击加号，则往购物车数组添加1份该餐品，反之减少。并且该餐品类别的计数器以及购物车的计数器。 |
| menu.handleTapCartImage | 处理购物车点击事件，升起购物车页面，根据购物车数组显示点餐内容。 |
| clearCart | 清空购物车数组 |
| jumpToOrder | 根据购物车内容生成订单并跳转至支付页面。 |

表5-25后端部分关键函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| FoodService.getMenu | 处理获取菜单业务，从数据库获取全部餐品，并根据类别构造不同数组，最后封装为Menu对象传递给小程序。 |
| TableService.getById | 根据餐桌号获取餐桌信息，返回前端。 |

1. 实现结果

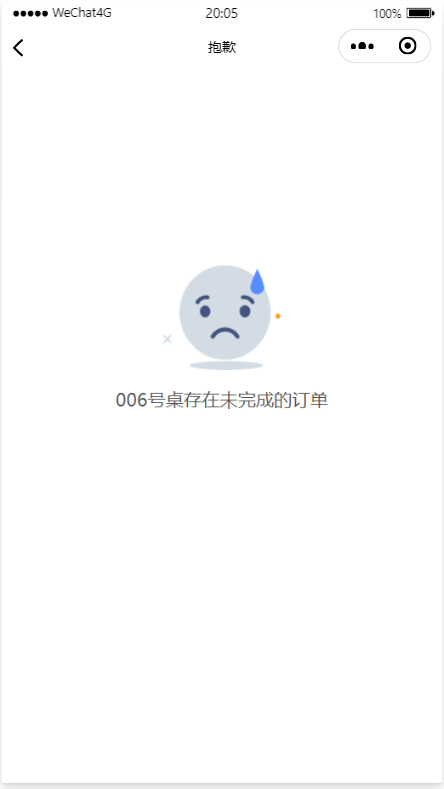
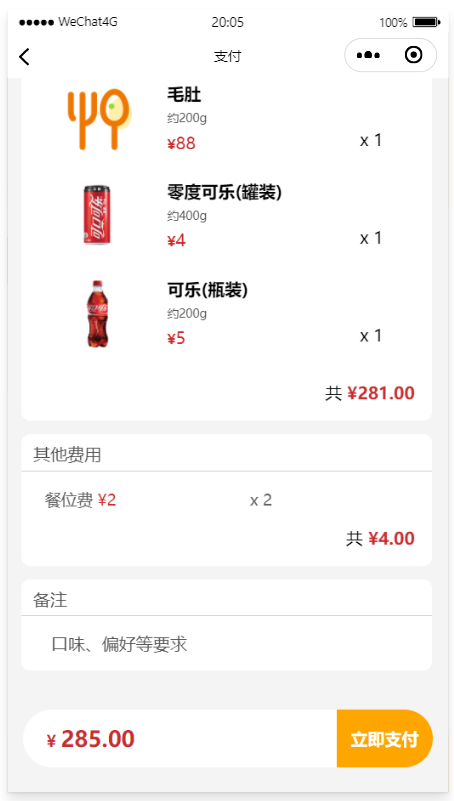
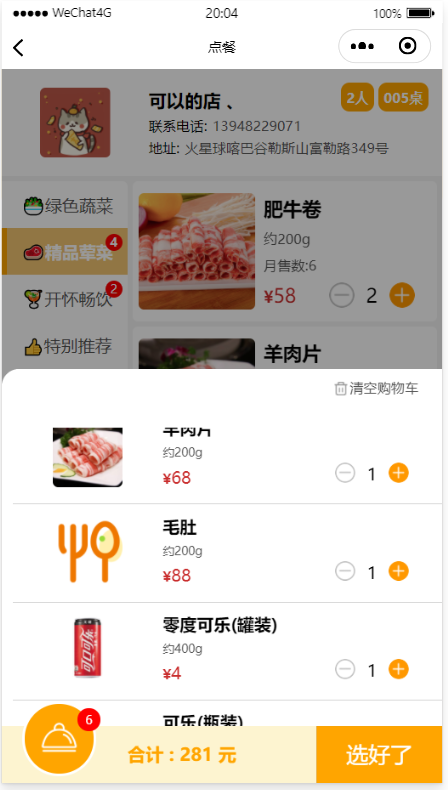
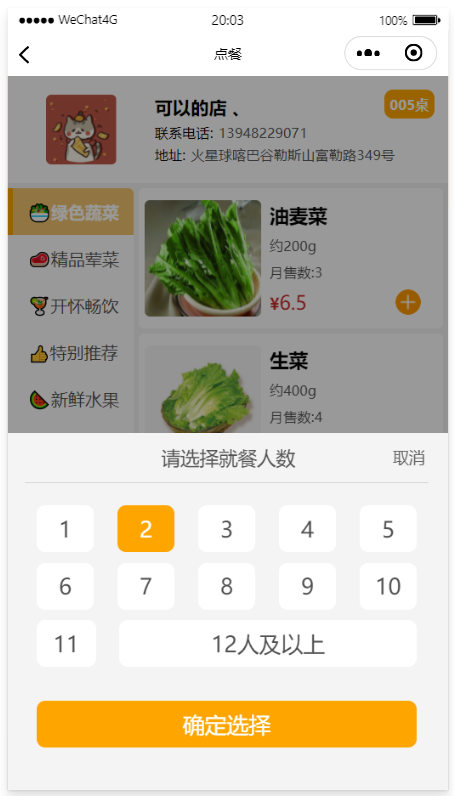


图5-30 小程序点餐模块功能实现

### 订单模块

1. 各功能业务流程
2. 支付订单：在支付页面，确认点餐内容无误并输入备注后，点击立即支付按钮完成支付。
3. 浏览历史订单：在主页点击“我的订单”，即可浏览该用户的全部订单，订单可分类显示：进行中、已完成、已评价。
4. 查看订单详情：在浏览订单页面选择订单点击查看详情，可以看到更多关于该订单的信息。
5. 关键函数方法

表5-26前端部分关键函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| pay.submit | 支付成功后发送“/order/add”请求，将订单信息发送到后端。 |
| order.onLoad | 发送“/order/{phone}”请求，获取指定用户的全部订单。 |
| order.filter | 根据不同状态筛选显示订单。 |

表5-27前端部分关键函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| OrderService.add | 添加订单业务，将新增的订单插入数据库，并切换餐桌状态为“就餐”，并添加该餐桌有新订单的通知，若查到该用户的排队信息，也一并删除。 |
| OrderService.getByPhone | 根据用手机号查询数据库，获得该用户的全部订单，返回给前端。 |

1. 实现结果

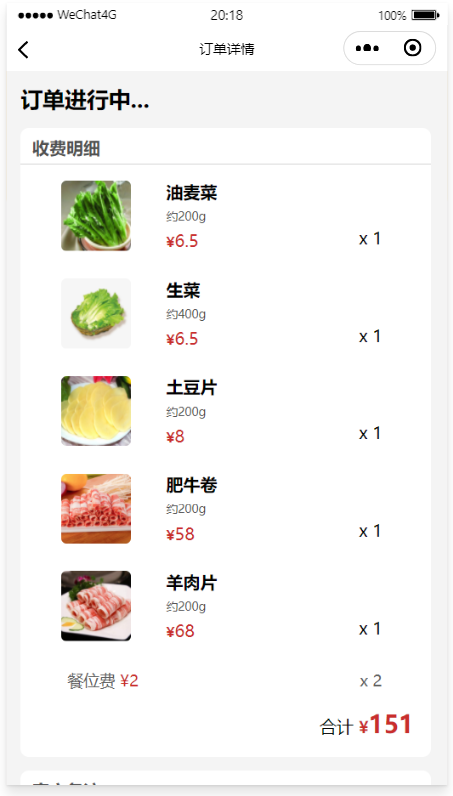
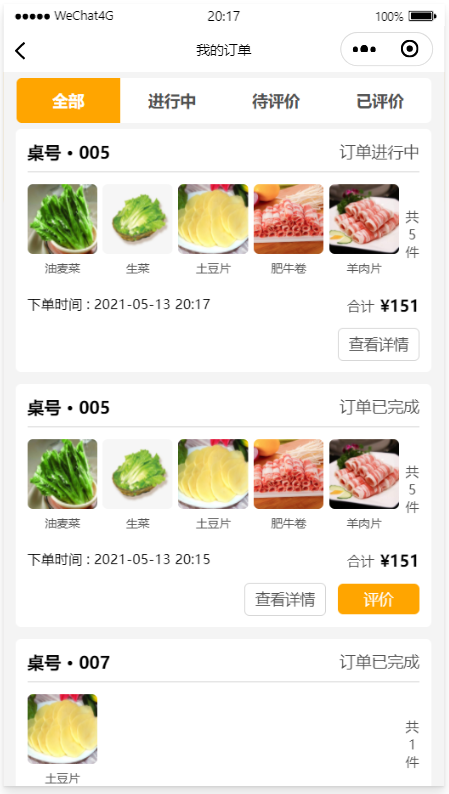


图5-31 小程序订单模块功能实现

### 5.3.4 排队预约模块

1. 各功能业务流程
2. 开始排队：进入排队预约页面后，首先选择要排的队伍，页面显示该类型餐桌的空闲个数、队伍中的人数，然后点击立即排队即可加入指定队伍。之后页面上显示已经等待的时间、自己在队伍中的位置，并实时更新这些数据。
3. 取消排队：点击取消排队按钮。
4. 关键函数方法（省略前端）

表5-28后端部分关键函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| LineService.enqueue | 处理入队业务，首先获取线程锁，访问数据库获得客户要排的指定队列的总人数，设置当前客户位置为总人数加一，然后将该条信息插入数据库。为避免重复排队，插入前先查询该手机号是否已经在某个排队队列中。 |
| LineService.cancel | 取消排队业务，需要线程锁，将该条记录删除并将该用户后面的人位置往前。此外队列中用户点餐后也会自动执行该函数。 |

1. 实现结果

图5-32 小程序订单模块功能实现

### 5.3.5 餐厅评价模块

1. 各功能业务流程
2. 浏览评论：在首页点击“餐厅评价”，即可浏览餐厅的评价。分为全部评价和我的评价两个筛选标签。
3. 添加评价：在“我的订单”页面选择待评价的订单，点击发布评价即可打开编写评价页面，可以打分、输入文字、插入图片，然后提交评价。
4. 关键函数方法

表5-29前端部分关键函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| remarks.onLoad | 发送“/remark/all”请求，获取全部评论。页面显示评论的评分、文字、图片。 |
| remarks.filter | 读取缓存中的手机号，根据手机号判断，点击“我的评价”时只显示自己评论。 |
| remark.add | 发送“/remark/add”请求，将评分、文字、用户信息作为参数发送，但无法同时传递图片文件，所以发送请求成功后获取评论返回的id，后续根据id上传图片。 |
| wx.chooseImage | 微信提供的选择手机图片或拍摄图片添加的接口。 |
| remark.upload | 发送“/remark/ uploadImage”请求，上传图片。 |

表5-30后端部分关键函数

|  |  |
| --- | --- |
| 函数名 | 说明 |
| RemarkService.all | 从数据库获取全部评论列表，传递到前端。 |
| RemarkService.add | 向数据库插入评价，不包括图片信息，使用Mybatis的主键回填，获取自增的评价id，返回给前端。 |
| RemarkService.uploadImage | 获得前端传递的图片文件，保存到服务器本地文件系统，并获得其路径，根据前端传递的评论id参数将图片路径更新到数据库。由于一个评论可以有多个图片，所以数据库保存的是图片路径数组的JSON化字符串。要使用是再使用JSON将字符串转对象即可。 |

1. 实现结果

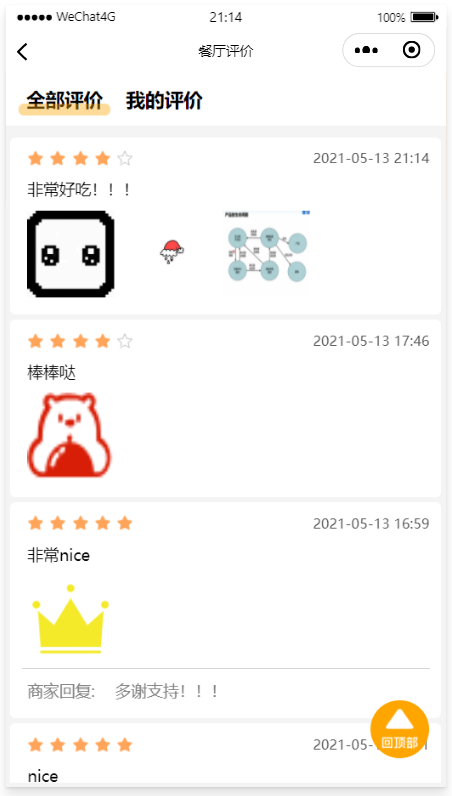
 

图5-33 小程序评价模块功能实现

## 5.4部署服务器

项目开发完成后，需要打包部署到云端服务器上，部署步骤如下：

1. 将后台管理系统的前端项目文件放在后端Spring Boot项目的resource文件夹下的static静态文件夹内。
2. 使用Maven生命周期中的package指令，将后端项目打包成名称为“restaurant-5.12.jar”的jar包。
3. 使用SFTP协议将jar包传输到云端linux服务器上的/root/restaurant目录下。
4. 用终端工具通过SSH登录服务器，在jar包目录下执行该条指令：“nohup java -jar restaurant-5.12.jar > log.log &”。
5. 至此项目已经部署成功，使用域名或ip地址加路由即可访问服务器提供的服务，运行时的日志输出在/root/restaurant/log.log文件内，上传的图片等文件保存在/root/restaurant/upload目录下。

# 第6章 总结

从2019年12月确定该选题至今天，已过去将近五个月的时间。这段时间我过的充实且有意义。前期，我查阅文献、寻找教学视频、读框架的官方指导文档，巩固了以前学过的后端知识，也学到了很多新的前端知识，学习了以前从未接触的LayUI和Vue框架，也对小程序的开发流程更加了解，这部分学习历程让我对前后端分离开发的理解更进一步，也让我后续的编码过程更加流畅。从今年1月中旬开始编码，陆续共写下了3000多行Java代码，5000多行html和js代码。这段过程中，也发现了许多开发过程中好用的工具：Postman、宝塔面板、MobaXterm等。

总体来说，这项“微信餐厅服务系统”的实现是成功的：完成了基本的餐厅管理功能，能灵活的修改餐单，查看餐厅的实时状态，管理订单、评论、店铺信息；也完成了基本的食客点餐功能，提供了点餐、支付、排队服务，支持订单查看、评论功能。但该系统也存在一些不足，后续可考虑一些功能：例如支持食客在用餐过程中进行加菜、退菜；又或是后台管理系统内允许商家指定一些优惠策略，当点餐内容达成优惠条件时，食客减少支付金额。总之，中国的餐饮消费市场是极其庞大的，软件可以满足人们的需求，也可以随着人们的需求进化，为人们的生活带来越来越多的便利。

# 参考文献

1. 全国餐饮业发展规划纲要(2009—2013)[EB/OL].

http://file.mofcom.gov.cn/article/gkml/200904/20090491713218.shtml.

1. 李淑华,余松涛.基于微信的高校点餐管理系统设计与实现[J].齐齐哈尔大学学报(自然科学版),2020,36(05):88-94.
2. 李新锦,鲁志萍,刘苑如.基于微信小程序的校园点餐系统的设计[J].电脑知识与技术,2020,16(07):50-51+56.
3. 王艳辉.基于微信小程序订餐系统的设计与研究[J].信息与电脑(理论版),2019,31(21):97-98.
4. 刘琤,赵桓鑫,王先勇.基于Spring Boot技术的微信平台校园食堂智能点餐系统设计[J].无线互联科技,2019,16(15):37-38.
5. 宋丹丹. 基于协同过滤的美食点餐推荐系统的设计与实现[D].武汉轻工大学,2018.
6. 申燕萍,何梦磊.基于微信小程序的点餐系统[J].电脑知识与技术,2018,14(04):62-63+83.
7. 袁源.基于遗传算法的智能点名系统的设计与实现[J].网络安全技术与应用,2015(04):37+41.
8. Wang J Y, Chang P Y, Liu S. An efficient roll call system for a smart phone[C]//System Science and Engineering (ICSSE), 2016 International Conference on. IEEE, 2016: 1-3.
9. Shannon L J, Ward Y D. A Case Study: From Game Programming to ICTs[M]//ICTs and the Millennium Development Goals. Springer US, 2014: 237-250.
10. 霍婉晖. 基于微信公众平台的智能点餐系统的设计与实现[D].吉林大学,2016.

# 致 谢

首先要感谢指导我完成毕业设计的王勇导师，感谢您在众多选择该题的学生中选择了我，给了我机会来开发这个我感兴趣的项目，也感谢您在我开发程序和撰写论文过程中对我提供的亲切关怀与悉心指导。老师时刻抱着严谨的科学态度和精益求精的工作作风，深深的感染了我并激励我前进！

同时我要感谢培养我四年的母校，感谢吉林大学软件学院，是在这里，我从一个懵懂无知的少年成长为社会所需要的人才。学校不仅教会了我赖以谋生的专业知识，还培养了我应对问题的思维方式、为人处世的道理，使我终生受益。