摘　要

本文主要从当前日益增加的用户订餐需求和商家需要降低管理成本的需求出发，以“轻便快捷”的微信小程序为依托，设计并实现了一款基于微信小程序的，满足用户和商家需要的订餐管理系统。

本文设计并实现的订餐管理系统主要包括小程序前端和管理后台两个部分，其中前端使用微信小程序里面的 WXML，WXSS 的可视化层技术，实现了添加购物车和下单美食，可以评论以及分享，可以管理个人中心的功能，管理后台主要使用 Flask MVC 技术框架，实现了账号管理、美食管理、会员管理以及财务管理等功能，对于平台数据本文使用 MySQL 数据库来进行数据管理，并使用微信开发平台提供的 API 接口来实现前后端数据互连，进而保证用户、商家和平台管理人员方便快捷的使用该应用。

关键词：Flask；MVC；订餐系统；微信小程序

Abstract

This paper starts from the increasing demand of customers' ordering and the demand of businessmen to reduce the cost of management. Based on the "light and fast" wechat small program, we design and implement a small program based on wechat to meet the needs of customers and businesses.

The ordering management system designed and implemented in this paper mainly includes two parts: the front-end of the small program and the back-end of the management. The front-end uses the wxml in the wechat small program, and the visualization layer technology of wxss, which realizes the addition of shopping cart and order operation, can comment and share, can manage the function of personal center, and the management back-end mainly uses flask MVC technology framework realizes account management, food management, member management, financial management and other functions. For the platform data, this paper uses MySQL database for data management, and uses the API interface provided by wechat development platform to realize the front and back data interconnection, so as to ensure that users, businesses and platform managers can use the application conveniently and quickly.

Key words: Flask；MVC；Ordering system；Applet of WeChat

目　录

第一章 绪论.....1

1.1 项目背景与意义.....1

1.2 国内外发展现状.....2

1.2.1 外卖点餐发展现状.....2

1.2.2 微信小程序发展现状.....2

第二章 开发综述.....4

2.1 Python 编程语言.....4

2.2 Flask 框架.....4

2.3 MySQL 数据库.....5

2.4 微信小程序.....5

2.4.1 小程序简介.....5

2.4.2 小程序框架.....6

2.4.3 小程序 API.....6

2.5 本章小结.....7

第三章 需求分析与设计.....8

3.1 需求分析.....8

3.1.1 外卖市场需求分析.....8

3.1.2 小程序市场需求分析.....9

3.1.3 系统功能需求.....9

3.2 系统总体设计.....10
3.2.1 系统开发结构设计.....10
3.2.2 系统架构设计.....11
3.2.3 系统功能架构.....12
3.3 本章小结.....12
第四章 系统运行所需环境及数据库设计.....13
4.1 系统环境配置.....13
4.2 数据库表设计.....14
4.2.1 数据库表分析.....14
4.2.2 数据表 E-R 图设计.....15
4.2.3 数据表结构设计.....19
4.3 本章小结.....23
第五章 系统功能模块的设计与实现.....24
5.1 后端的设计与实现.....24
5.1.1 管理员登陆模块的设计与实现.....24
5.1.2 管理员信息编辑模块的设计与实现.....27
5.1.3 美食编辑模块的设计与实现.....28
5.1.4 财务管理模块的设计与实现.....32
5.2 小程序端的设计与实现.....33
5.2.1 登陆模块的设计与实现.....33
5.2.2 美食展示模块的设计与实现.....37

5.2.3 购物车模块的设计与实现 39

5.2.4 下单模块的设计与实现 40

第六章 总结与展望 44

参 考 文 献 45

致 谢 46

第一章 绪论

1.1 项目背景与意义

随着互联网的迅猛发展，互联网已然成为巨大信息的交换和流通的平台，人们的生活质量也因此不断提高。其中外卖 app 的发展就是一个很好的例子。互联网餐饮外卖以其食品种类齐全，方便快捷，成为现代大部分人生活中重要的组成部分，而且激发了餐饮外卖行业的创业与就业[1]。外卖平台虽然流量大，但是增加了餐厅经营难度，平台抽佣较多，使得店家成本高，用户数据基本绑定在第三方，使用户留存成难题。据了解某个大型外卖平台目前的佣金约为 26%。如此高的外卖佣金意味着他们的收入将大大减少。但微信小程序没有佣金，所以当它们被用于外卖时，商家自然可以降低成本，提高利润。这使得企业和商家不得不向小程序跳槽。

腾讯于 2017 年推出的微信小程序是一款依托于微信运行的手机应用。据调查显示，截止到 2016 年 12 月微信全球共计 8.89 亿月活用户，正是这巨大的用户信息量带动了信息消费 1742.5 亿元，同时也带动了微信小程序的发展[2]。自小程序诞生以来，各个行业都有不少企业和商家选择并且发展了小程序，其中还包括很多外卖行业的企业和商家。微信用户只需要在微信的搜索功能里直接搜索或扫描二维码即可。微信小程序具备四大特性，分别是无需下载、触手可及、用完即走和无需卸载，这让它成为了免去客户端的轻型应用程序。这种不需要下载安装即可使用的应用，它真正实现了互联网中“触手可及”的梦想。

毫无疑问的是，“外卖点餐+小程序”符合当前我国发展经济模式，这种结合无论是学生，还是成年人甚至是老年人都带来极大的便利。传统的人工点餐降低了人们的就餐体验以及出错率比较高，而利用微信小程序实现的二维码点餐系统，更加能给客户带来更优质的体验感[3]。因为它拥有轻服务的小程序技术，用户无需下载应用，点开微信即可使用，也不用耗费时间人力去到店点餐，这对所有用户来说是非常方便的。

因此，将个外卖点餐与小程序结合是完全适合当下社会的需求，本系统的开发具有以下意义：从商家层面来说，本设计能够帮助商家结合微信庞大的流量群体，整合线下销售服务和互联网，拓展出视野更加宽阔的线上下线一体化市场，通过微信线上的用户群体，线下服务客户，可以拉近自己与用户的距离，达成交易。同时，它可以为企业节省人力和运营费用；从开发者角度来看，小

程序开发门槛相对较低，难度小于 APP，既能够满足简单的基础应用，又适合生活服务类线下商铺以及非刚需低频应用的转换；从用户层面来看，本设计响应小程序发展趋势，可以节约使用时间成本和手机内存空间，改善现代人生活。

1.2 国内外发展现状

1.2.1 外卖点餐发展现状

长期以来，网上订餐的方式在国外得到了广泛的应用，不仅节省了时间，而且为餐饮管理提供了一种更加简单有效的方式。美国率先提出网上订购模式。对于餐饮业来说，发达国家的发展速度和需求远远高于不发达国家。毕竟，发达国家的人时间观念很强，对服务的需求也很高。更重要的是，他们可以不断找到更先进的管理方法和手段，并将其发挥到极致，尤其是信息技术。顾客的消费水平在逐渐提高，餐饮行业的竞争相当激烈[4]。据调查，国外网上订购大多是通过互联网开办类似超市的网上餐饮，再与专业物流配送公司合作，因此他们的订购体系广为传播，得到了群众的广泛认可和高度赞赏。

近年来，中国餐饮业发展迅猛，据调查统计 2017 年我国的外卖市场规模突破 2000 亿元，我国网上外卖用户规模达到 3.43 亿人，这数据信息足以表明我国的餐饮市场潜力巨大[5]。当今大部分人选择以订餐的方式来解决自己的用餐问题，这从而诞生了 O2O 外卖平台，它不仅能够满足消费者的需求而且也顺应了国家与社会经济发展趋势[6]。商店可以根据自己的条件和环境要求选择适合他们的营销策略。只有这样，他们才能从激烈的市场竞争中崭露头角。在当今的大数据时代，了解如何使用当今快速增长的网络以及掌握如何使用微信可以为您带来巨大的商机。在中国，在线食品订购已基本遍布整个中国外卖市场。例如，美团和饿了么这两个阵营占领了中国的大部分市场，并且已经出现了无数具有在线食品订购系统的公司。

1.2.2 微信小程序发展现状

时代是不断向前发展的，同时科学技术也越来越成熟。轻型应用程序在 2017 年开始变得炙手可热。原因是推出了拥有近 10 亿用户的微信小程序。根本原因是技术的发展，应用程序市场模型的缺陷以及用户探索应用程序的意愿下降。

随着小程序的出现，市场迫切需要开发人员以较低的成本开发应用程序并分发应用程序，以便用户可以更有效地发现应用程序并使用新的应用程序载体。

微信小程序不但给用户带来了极大的方便，无需考虑适应性和分发，还免

去了传统 app 需要下载的烦琐，即用即来，关闭即走，不会占用用户设备过多的内存[7]。它的功能恰好满足了市场的需求。于是，微信小程序成为 2018 年最热销的网点，无数的企业家，开发商和投资机构进入，共同创造了小程序的早期生态。移动互联网时代也加入了逐渐兴起的“轻服务”。经济的高速发展，让人们不得不加快生活节奏，时间效益成为了现代人考虑事情的要点之一。那么“轻服务”就是一种便捷高效的方式，为他人节约时间的服务。当下最稀有的莫过于时间，最好的服务不仅仅是有温度的，而且也必须是“轻”的[8]。微信小程序无需下载即可使用。它们也与“轻服务”的概念相吻合。

目前，微信的开放环境已经创建了大量的第三方公司。微信订餐从在线订餐中脱颖而出，于是微信订餐的发展直突飞猛进，必将成为未来外卖订购的主流趋势。通过微信点餐系统，群众更加方便快捷，商店与顾客之间的交流更加畅通，同时商店可以更好地为顾客服务。大学生是美团、饿了么等外卖平台的主要消费群体，但由于需下载以及内存的占用，学生尤其高度支持并赞赏微信订餐[9]。这是一种全新的外卖订购模型，它突破了传统模型。

综上所述，外卖点餐与微信小程序的联合是历史之必然。根据日常需求搜索或查找附近的小程序，始终可以找到满足需要的小程序。由于小程序的出现，传统的电子商务和零售将重生。目前小程序所展现出来的势头只是星星之火，未来可以燎原，潜力无限。本系统开发的点餐微信小程序，能给外卖行业注入新血液，减少了高峰期排队点餐和就餐的等待时间，节约了用户时间，能给消费者客户带来更高效更快捷订餐服务[10]。只需要拥有一个微信就能使用，快捷高效。

第二章 开发综述

2.1 Python 编程语言

Python 的创始人是荷兰的吉多·范罗苏姆，是一种大范围使用的计算机编程语言，是一种解释型脚本、面向对象的动态类型语言，吉多·范罗苏姆一开始是用于编写自动化脚本的。但是日新月异升级的版本以及新的语言语法功能的加入，Python 逐渐完善并且它越来越多地被用于独立的大型项目开发中。因为 Python 语言具有三大特性，分别是简洁性、易读性和可扩展性，国外越来越多的研究机构使用 Python 进行科学计算，甚至一些著名的大学使用 Python 来教授编程课程。Python 在设计上坚持着有条有理，一目了然的风格，使 Python 成为一种易于阅读和易于维护的编程语言，所以被大量用户认可。在编程语言排行榜内位列第三，也是时下最热门的编程语言之一。在计算机语言中 Python 发展速度不可小觑，可以说是最适合零基础人士入门学习的计算机语言之一[11]，广泛应用于企业级 Web 应用开发和移动应用开发。

Python 一开始就被设计为有可扩展的特性。并不是全部的特性和功能都集中到语言核心中。程序员可以方便地使用 C、C++、Cython 来编写扩充模块，因为它提供了丰富的 API 和工具。其次，Python 还具有可嵌入性，如果其他程序需要脚本语言，那么 Python 也可以被集成到它们之内。因此，越来越多人还把 Python 作为一种“胶水语言”（glue language）使用。

2.2 Flask 框架

Python 中四大主流 MVC 框架目前有 Flask、Django、Tornado 以及 Twisted。MVC 分层有助于管理复杂的应用程序，可以在不依赖业务逻辑的情况下专注于视图设计[12]。其中 Flask 是一种非常强大的框架，它问世于上面提到的三个框架之后，这让它具有了其他框架的一些特性。以 Werkzeug 为 WSGI 工具集，简单来说就是 request 和 response。模板引擎则采用 Jinja2。Flask，可以用一个字来形容，那就是“轻”。并且它是一个可以定制的框架，相比其他框架更加灵活，使用方便，它微小的特性更适用于我们开发一些小项目，因此被称之为“微框架”。同时它具有可扩展性，Flask 在操作上并不会束缚我们，它可以让开发者有更多的自由选择权利，例如可以选择用什么数据库插件去存储他们

的数据。

总结 Flask 有以下几点特点：

1）整个系统采用统一语言开发，以 Werkzeug 为 WSGI 工具集，jinja2 为模板引擎；开发人员可以完全专注于业务逻辑，前端和后端的分离便于开发和后期维护。

2）自由灵活、可扩展性极强，基本所有功能都需要第三方插件框架。

3）因为是由框架处理操作，所以稳定性和安全性极高。

总之，Flask 就是提供一个平台给开发者，开发者可以不断地安装插件来完善自己的项目。

2.3 MySQL 数据库

MySQL 是被开发者用的最多，也是被大众所认同的关系型数据库。由于 MySQL 是开源的，因此我们可以免费使用。MySQL 的优点在于：可以创建不同的数据表，并将它们联系起来。在有秩序的表中查找数据，可以大大提高我们查找数据的效率，而并不是在查找一堆杂乱的数据。目前大部分的中小型企业及许多的互联网公司都选用 MySQL 来作为数据库，MySQL 具有极高的性能，对于应付百万级的数据，还是相当轻松的。因为它能在成千上万条的数据中支持同时处理，就连电商巨头阿里巴巴也是选择 MySQL 作为数据库，足以见到其强大之处。MySQL 使用的也是标准的 SQL 数据语言形式，只有少部分地方会 SQL 语法有不一样的地方[13]。大部分的编程语言都能被 MySQL 支持，例如 C、C++、Python、Java，同时也支持应用于许多系统上，如微软的 Windows 系统和开源的 Linux 系统以及苹果的 Mac 系统。

2.4 微信小程序

2.4.1 小程序简介

小程序的问世，让世界的人为之惊叹。它相当于一座桥梁，能出色地将用户与服务连接起来，能在微信内快速被使用和传播。一般网页使用 HTML 和 CSS 来编写，但是小程序主要使用 WXML 与 WXSS。它主要的开发语言是 JavaScript。

1、微信小程序中的 JavaScript 跟网页里的 JavaScript 是非常类似的，唯一不同的是微信小程序具备了一些内置的 API 方法，这些封装好的方法我们在开发过程中是可以直接拿来调用的。

2、WXML 是基于 XML 语法开发的。结合组件和事件系统建立页面，可以改动相应的标签和处理事件逻辑。

3、WXSS 继承了 CSS 大部分的特点，它就是一个样式语言，可以用来修饰 WXML 里面的组件。

2.4.2 小程序框架

小程序框架系统的开发可以分为两部分，分别是视图层（View）和逻辑层（AppService），而 JavaScript 则是在两者之间提供数据传输和实践系统。技术框架图如下图 2-1 所示。

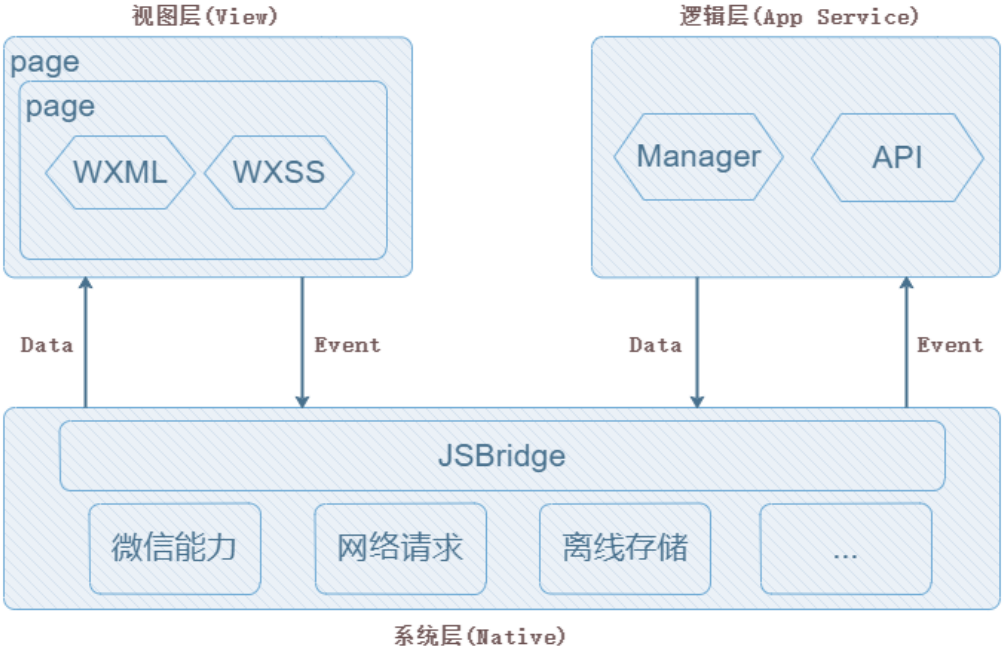

图 2-1 技术框架图

逻辑层专门用来处理逻辑，接收视图层的请求以及返回参数。例如请求数据和接口调用等。视图层则是对页面进行渲染的。

App 是小程序的核心部分，由 app.js、app.json、app.wxss 三种类型文件构成。分别起到处理小程序全局逻辑、小程序全局页面设置、小程序全局页面公共视图层样式的作用。而 Page 一般存在四种基础的文件类型，分别是 JS 当前页面逻辑文件、.WXML 页面描述文件、.WXSS 样式表文件、.JSON 配置文件构成。

2.4.3 小程序 API

API 就是开放的一些应用程序编程接口。现在很多网站为了拥有更多的访问量以及用户，都会将自己的资源开放给开发者调用。微信研发团队也不例外，

供给了开发者大量丰富的原生 API。简单来说就是微信开发者封装了很多接口函数，开发者能够很方便的去调用这些函数并且开发者并不需要去关心这些代码具体是怎么实现的。例如获取用户基本信息、数据存储以及转发分享等功能。这极大的减少了我们开发者的时间与精力。

2.5 本章小结

本章主要介绍了开发此订餐小程序之前所涉及到的技术以及工具，从而希望读者能通过文字更加深入了解到这些技术和工具的来源、拥有哪些主要功能，为接下来的开发做好充足的准备

第三章 需求分析与设计

3.1 需求分析

3.1.1 外卖市场需求分析

餐饮行业作为中国服务性第三支柱产业，它在人们的生活中一直扮演着无可替代的角色。改革开放以来，我国的经济发展突飞猛进以及百姓的人均 GDP 有显著的提高，所以餐饮行业的发展得以呈现出朝气蓬勃的发展趋势。但是在这激烈的餐饮行业竞争内我们必须要有新鲜血液的注入，需要有一种模式来带领餐饮行业的持续发展，并且满足消费者日益变化的需求，是餐饮业面临的紧迫问题。

国家经济的快速发展给餐饮行业带来了无限潜力，人们外出就餐的可能性大大地提高，消费者自己做饭的机会越来越少，选择外出就餐的人也越来越多。但是随着互联网的发展和智能手机的出现，消费者不仅可以通过手机订餐就能吃到自己想要的食物，而且不用以到店点餐的方式去花太多的时间和精力。在餐饮业中外卖已经普遍得到消费者们的认可，因此越来越多的餐饮企业和小企业除了吸引到店消费的顾客以外，还将推出送货服务，作为消费者更方便的辅助营销手段[14]。从以前的到店点餐可以附带一些小卡片，到至今的互联网订购平台，外卖行业势如破竹，外卖正以意想不到的速度逐渐渗透到我们生活中的方方面面。据统计，有 60%以上的外卖营收占据着大部分餐馆的日常主营业务中。甚至有的商家干脆取消到店点餐的服务，在家里或工厂里做饭，专门从事配送服务，这样不仅节省了成本，而且扩大了市场。致使这个行业逐渐发展成为今天的 O2O 平台。

2020 年毫无疑问是一个不平凡的年，一场疫情打破了我们正常生活的节奏。在此国家号召人们，少外出，少聚餐，也同时号召人们尽量不要在餐饮店就餐，可以实行打包带走或者点外卖的方式来避免人流量过多的频繁交流。众志成城，抗击疫情，是我们每一个人的职责所在，不外出就是对国家作出最大了最大的贡献。在此情况下，不出门买菜以及不想自己动手做饭的人，外卖订餐便是作为消费者们一个不二的选择。基于生活消费的本质，在线外卖服务已经与生活相关服务相结合。一方面拓展服务入口，另一方面与生活消费服务相结合，构建生活服务生态系统。从经济方面来讲，拉动了餐饮消费的提高；从人民群众方面来讲，一是让商家能与消费者有更好的交流，二是更方便消费者

节省不必要花费的时间和经精力，尤其是对上班族。

3.1.2 小程序市场需求分析

2017 年 1 月 9 日前，APP 曾主宰着手机程序市场，但在此之后，小程序问世，因登录之后不用安装，所以 App 被压制，相反小程序成为了手机程序界的新领头羊。微信小程序自出现以来就在社会各个行业碰撞出火花，小程序登录在逐渐的成长和完善，就为了给大家提供更轻松自在的服务。虽然它们很小，但对大众媒体，特别是应用程序，影响深远。

在全国人大第十二次会议上，李克强总理在政府工作报告首次提出了“互联网+”行动计划，即互联网将与社会各部门相结合的经济发展的新方向，来带动国民经济的发展[15]。此后，大部分餐饮店以“互联网+外卖”为盈利目标，手机里也因此出现很多外卖 App，例如美团、饿了么、百度糯米等在移动互联网市场上极速发展。但由于创业者、开发者甚至是用户都将局限于“以应用为中心”的思想，所以 App 面临遭遇瓶颈期。就在此时微信拥有庞大的流量群，并且微信小程序相对于移动 APP 更加方便快捷，无需下载，更加符合现今社会追求“轻服务”的理念。小程序的使用日益丰富，甚至成为消费者们必备的软件，在互联网发展势头强劲的背景下，凭借“社交化”和“去中心化”方面的创新优势，外卖小程序将会对传统 App 引发新一轮竞争博弈[16]，他的发展必将为这类应用的停滞不前的发展注入新的活力。

3.1.3 系统功能需求

外卖系统需要有一个合理的后端处理对策，一是为了能给商家带来用户管理、财务管理和菜品的正常录入，库存的及时更新，以及把评论区展现给用户，给消费者带来更好的体验感。二是防止用户账号密码以及地址信息泄露，造成不必要的麻烦。

“外卖管理系统”后端主要功能有“账号管理”、“美餐管理”、“会员管理”以及“财务管理”。小程序前端主要功能有“个人信息管理功能”、“添加购物车”、“分享和评论功能”。

本微信小程序订餐系统包含一下几个功能模块需求

服务器端：

1、用户模块

1）管理员注册、登录、退出登录。

2）管理员间可以互相查看访问记录、编辑信息以及增删账号操作。

2、美食模块

1）美食的添加删除、分类查找。

2）美食的信息编辑以及库存更新。

3、会员模块

1）用户信息查看。

2）用户订单和评论查看。

4、财务模块

1）订单列表查看。

2）财务流水统计。

小程序前端：

1、首页

1）菜品展示以及搜索功能。

2）分享和添加购物车功能。

2、购物车

1）用户购物车菜品的增删操作。

2）用户结算功能。

3、我的信息

1）我的订单查看。

2）地址管理。

3）评价查看。

3.2 系统总体设计

3.2.1 系统开发结构设计

MVC 模式是一种软件架构模式，也是一种分层架构模式，一般把软件模式分为三部分，模型(Model)+视图(View)+控制器(Controller)。

1、视图层（View）是最外面的一层，视图呈现给用户，代表着用户交互页面。它可以将用户的请求传递到后端，也就是控制器层（Controller），同时也可以获得模型层(Model)传递过来的参数，并实时更新页面，当然也包括一些 DOM 事件或者 AJAX 请求操作(发布事件)，都是放在视图层来完成。

2、模型层(Model)是最里面的一层，它封装着数据并且用于和数据库交互。模型层可以直接访问数据。模型层并不依靠“视图层”和“控制器层”，所以说模型层是和它们“分离”的，页面的展示和操作完全与它无关。

3、控制器(Controller)是中间的一层，它负责处理视图层发送过来的请求，然后根据需要去调用数据层 Model 中的方法来获取数据并且传递给视图层。

MVC 三层各司其职，简单来说，当有一层作出改变时，另外两层都是接

连变化的，是相辅相成的，但又互不干扰。所以维护性极高，当我们修改数据的时候或许只需要修改某一层就足够了，为开发者维护和调试代码节省了很多时间。

而在本订餐系统中，View 层则是前端，前端用的是 Python Flask 中的 jinja2 模板引擎。Model 层则是一个个对应数据表字段的实体类，以及一些 property 装饰器。Controller 层则是用 Flask 中的路由注册，注册路由的过程就是建立 URL 和 Python 视图函数或函数映射的过程并且建立一一对应的映射关系，并且这些路由函数是可以设置映射地址的，客户端可以通过地址去访问每一个 Controller 类。

3.2.2 系统架构设计

本微信小程序订餐系统的体系结构是由客户端（前端）和服务端（后端）构成的，客户端是为使用本订餐系统的用户提供的。本订餐系统的全部功能几都是由客户端、服务端和数据库交互的结果。本微信小程序订餐系统的总体架构如下图 3-1。

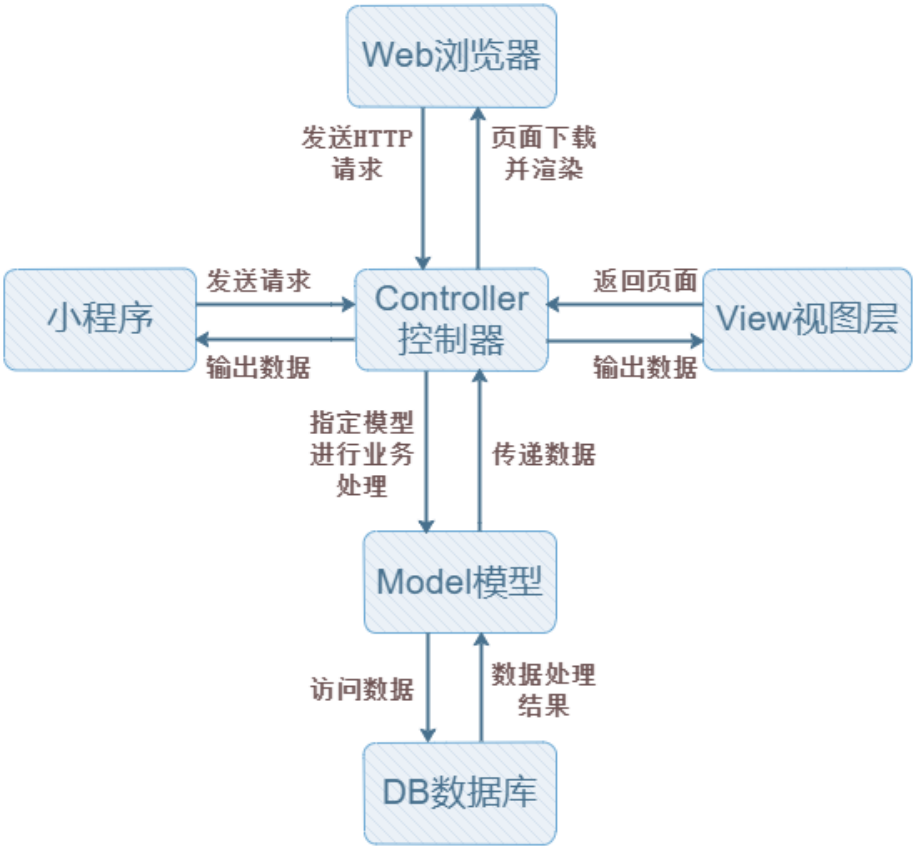

图 3-1 总体架构图

3.2.3 系统功能架构

根据 3.1.3 系统的功能需求分析，作出微信小程序订餐系统功能模块如下图 3-2 所示：

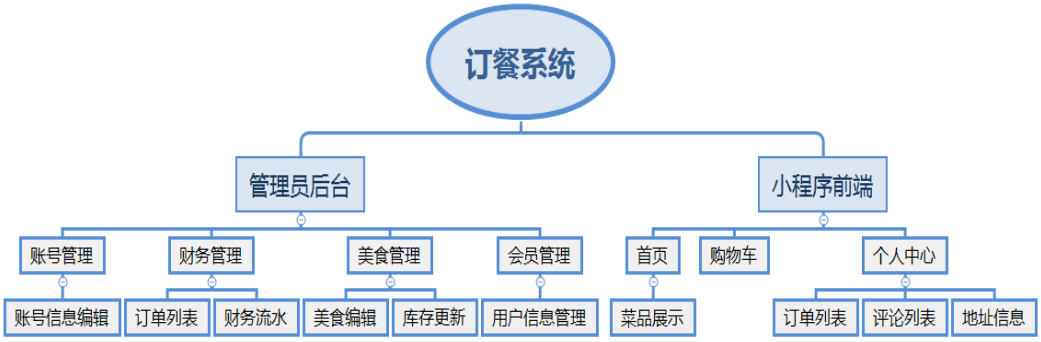

图 3-2 "微信小程序订餐系统"功能

3.3 本章小结

本章主要介绍了订餐小程序系统的总体设计，通过本章希望能让读者了解到订餐小程序系统是如何设计的、拥有哪些主要功能，以及页面所对应的功能与逻辑，从而让读者能通过文字，初步了解订餐小程序系统是如何使用以及使用逻辑。

第四章 系统运行所需环境及数据库设计

4.1 系统环境配置

Python 环境的安装及其配置

在进行 Python 应用开发之前需要安装 Python 环境，Python 目前在很多平台上都能使用。只要下载适合你所选择平台的二进制代码（win32/64 位），然后安装 Python 即可。本系统使用的 Python 版本为 Python3.5.1，相应的版本可以在 https://www.python.org/downloads/官网进行下载，理论上只要是 Python3 以上版本都可以满足本系统的运行时环境，下载安装完成之后，在计算机属性里对系统进行环境变量配置，在 Path 里添加 Python 安装的路径即可。最后在 cmd 里面输入 python，就会看到 Python 版本，代表安装成功了。另外需要注意 Python 中虚拟环境工具 Virtualenv，因为在开发 Python 应用程序时，默认的 Python 版本或许不适合此时开发的这个应用程序，一旦开发多个程序，就极有可能用到多个版本的 Python 环境。Virtualenv 就是用来搭建虚拟且独立的 Python 环境，可以使每个项目环境各自拥有一套独立的 Python 环境。Virtualenv 并不需要手动安装，因为 Pycharm 开发工具非常强大，本身就自带 Virtualenv 工具，只需在 setting 里面中的 Project Interpreter 添加对应的扩展包即可。

MySQL 数据库

在开发的时候，不管你是大的还是小的项目，数据库是必不可少的。本系统采用 MySQL 数据库，版本为 5.7，安装 MySQL 数据库你可以到 https://www.mysql.com/官网进行下载，安装的过程中需要特别注意账号密码都需要记住，否则将无法连接到刚安装完成的 MySQL 数据库。通常学习时我们都将账号设置为 root，密码设置为 123456，这里仅提供参考。

Pycharm

本系统使用的开发工具为 Pycharm，这款工具是由位于捷克的布拉格的 JetBrains 公司所开发，Pycharm 是 Python 程序员公认的最好开发工具之一，它自备一整套工具，可以帮助开发者在使用 Python 语言时提高效率。例如 Debug 调试、代码跳转、智能提示等等。安装 Pycharm 可以到官网 https://www.jetbrains.com/pycharm/下载。在下载的时候，特别需要注意勾选 pip 模块的安装，pip 是包管理工具，它具有安装和卸载功能，因为本系统后面需要通过 pip 来下载一些插件，例如 Flask-SQLAlchemy：操作数据库、Flask-script：

支持命令行选项等。本设计采用 Pycharm2018.1.4 破解版，专业版是需要有一

个授权码的，授权码百度一下你就可以获得，当然如果你经济宽裕的话也可以去下载专业版的。

4.2 数据库表设计

4. 2. 1 数据库表分析

为了满足订餐系统的开发，本系统需要建立与用户模块、菜品模块、会员模块和财务模块等的表来支持研发需求。

本设计对每个模块都提出数据表需求：

1、用户信息记录表(user)：用于存放管理员的登陆的基本资料，包括管理员 uid、用户名、手机号码、登陆用户名和密码、登陆密码的随机加密秘钥、邮件等信息。

2、会员用户记录表(member)：用于存放会员用户的登陆记录，如会员用户名、会员手机号码、性别、头像、更新时间等信息。

3、美食记录表(food)：用于存放美食列表记录，如美食名称、id、售卖金额、描述、库存、标签等信息。

4、美食分类记录表(food_cat)：用于存储美食的分类表，包括类别名称、权重、更新时间等。

5、美食销售记录表(food_sale_change_log)：用于存储美食销售情况，需要记录商品 id、售卖数量、售卖金额、会员 id、售卖时间等。

6、美食库存变更表(food_stock_change_log)：用于存放美食数据库存变更表，需要商品 id、变更数量、变更之后总量、插入时间等。

7、图片记录表(images)：用于存放美食图片的表，需要 id、文件名、插入时间等。

8、会员地址记录表(member_address)：用于存放会员用户地址的表，需要会员 id、变更数量、变更之后总量、插入时间等。

9、购物车记录表(member_cart)：用于存放会员用户心仪的商品，需要会员 id、商品 id、总量、插入时间等。

10、分享记录表(wx_share_history)：用于存放会员用户心仪的商品，需要会员 id、商品 id、总量、插入时间等。

11、会员评论记录表(member_comments)：用于存放会员用户的评价，需要会员 id、商品 id、订单 id、评分、评论内容、插入时间等。

12、在线购买订单记录表(pay_order)：用于存放会员用户的订单，需要会员 id、订单应付金额、运费、订单状态、插入时间等。

13、订单详情记录表(pay_order_item)：用于存放订单详情的表，需要会员 id、订单 id、售价、数量、订单状态、插入时间等。

14、事件队列记录表(queue_list)：用于处理订单队列的表，需要队列数据、状态值、插入时间。

4.2.2 数据表 E-R 图设计

根据上面的数据表的需求分析，我们可以清晰得到各个实体的设计思路以及它们之间的关系，并且使用实体关系（E-R）表现出来。各 E-R 图如下图 4-1 至 4-11.所示：

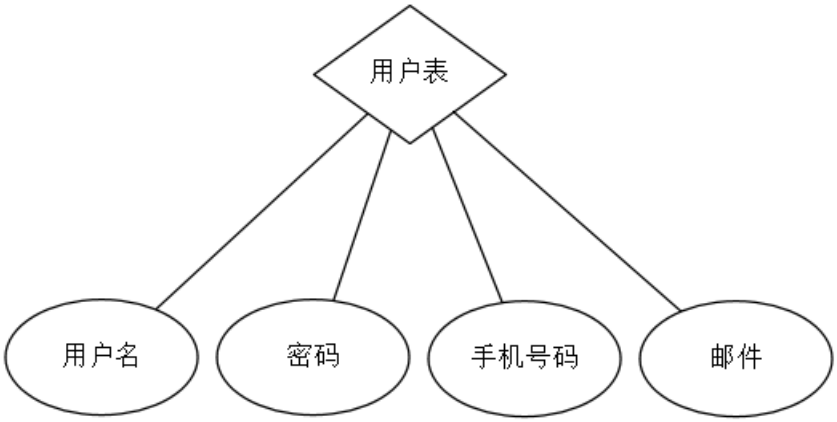

图 4-1 用户表(user)实体 E-R 图

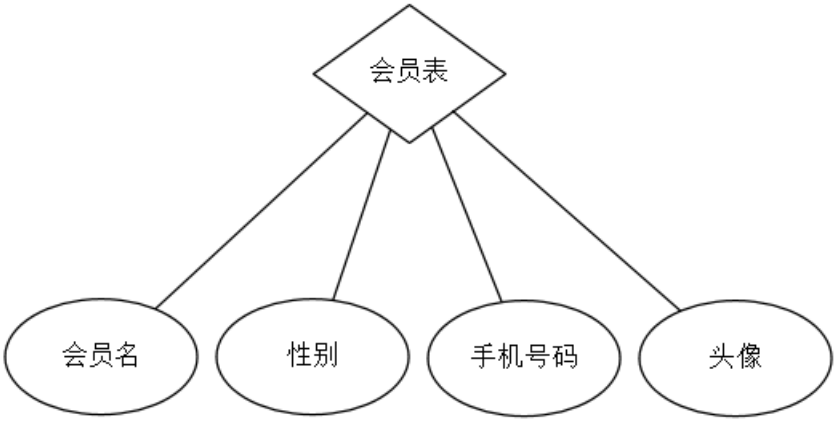

图 4-2 会员表(member)实体 E-R 图

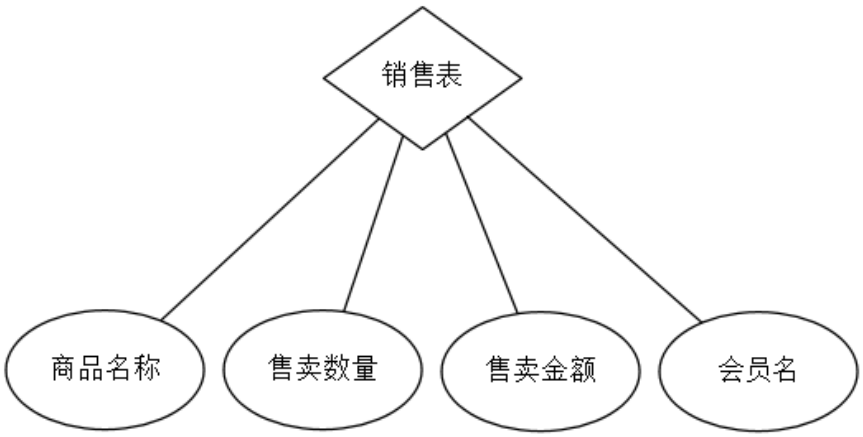

图 4-3 销售表(food_sale_change_log)实体 E-R 图

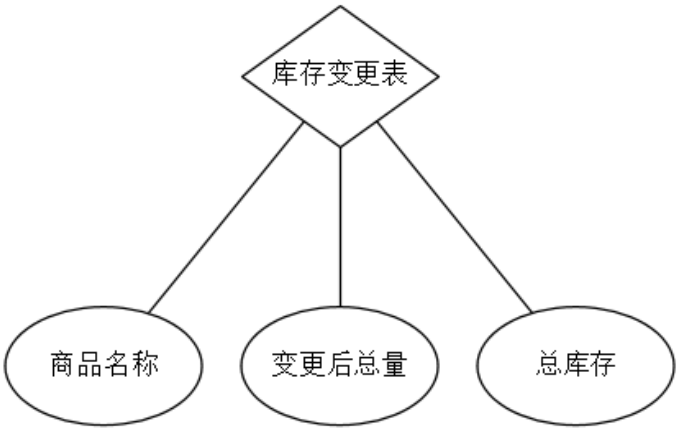

图 4-4 库存变更表(food_stock_change_log)实体 E-R 图

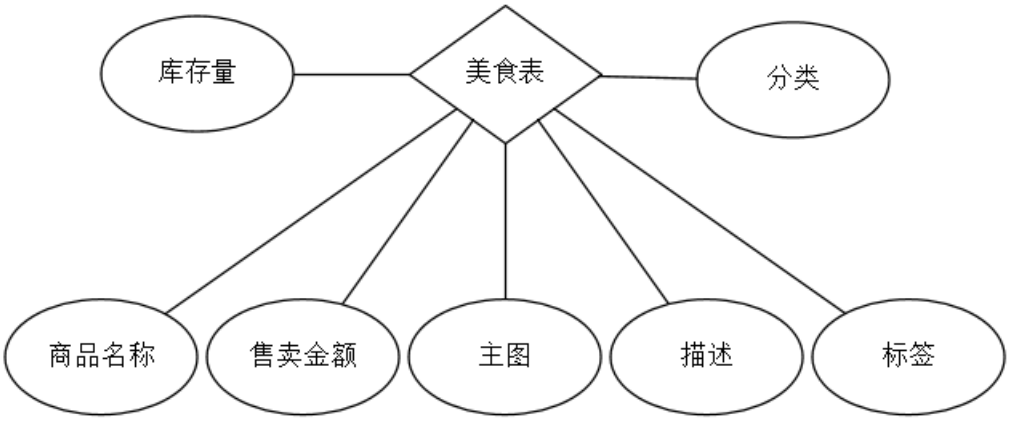

图 4-5 美食表(food)实体 E-R 图

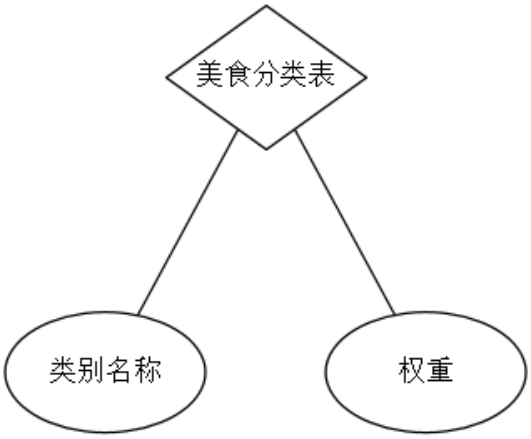

图 4-6 美食分类表(food_cat)实体 E-R 图

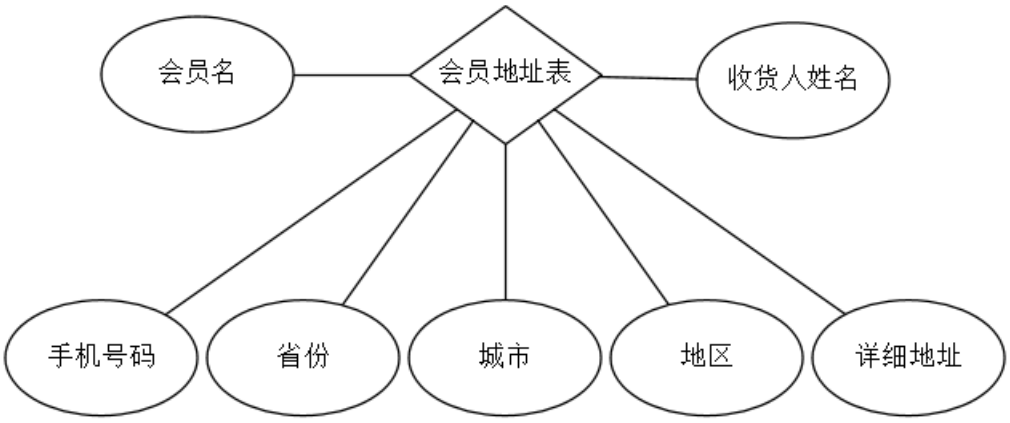

图 4-7 会员地址表(member_address)实体 E-R 图

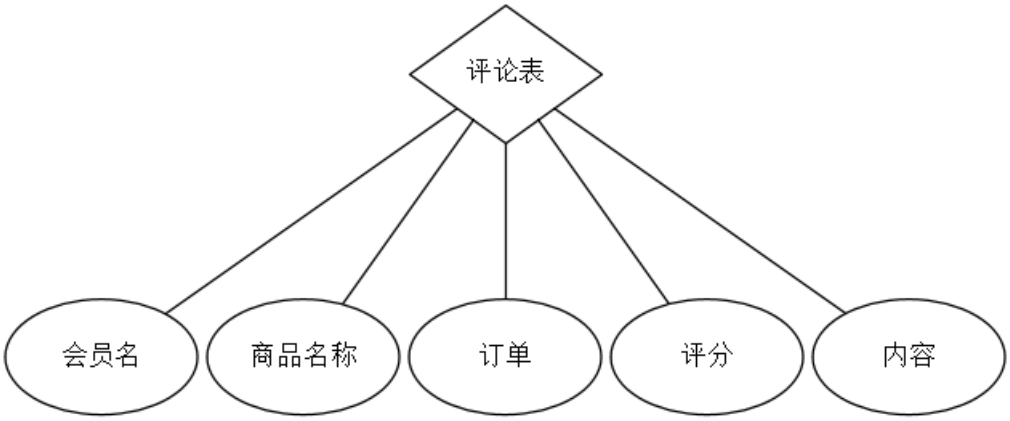

图 4-8 评论表(member_comments)实体 E-R 图

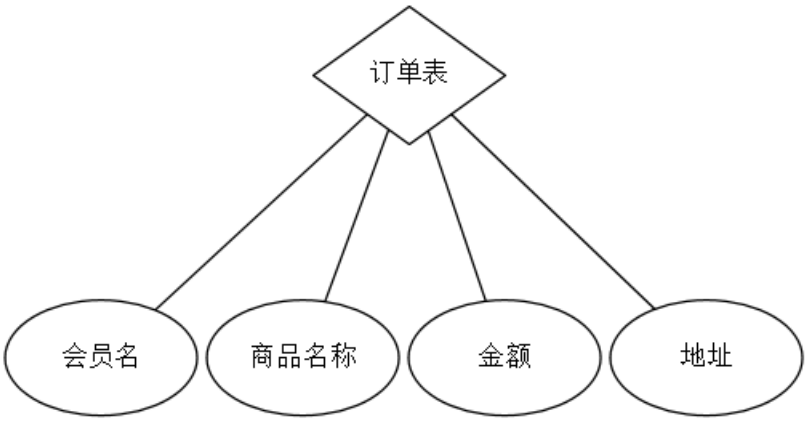

图 4-9 订单表(pay_order)实体 E-R 图

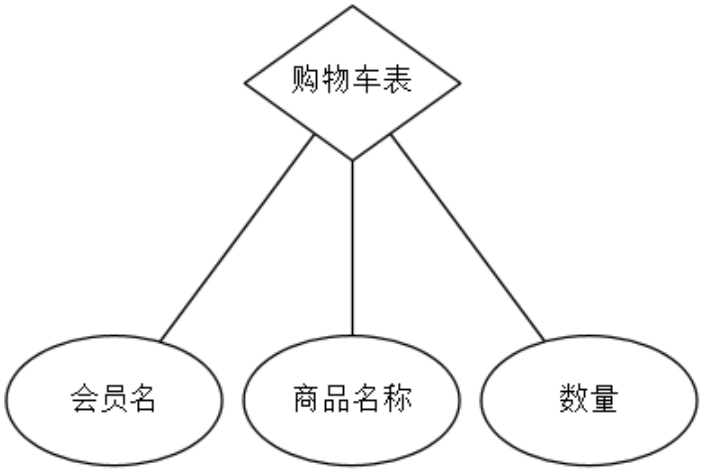

图 4-10 购物车表(member_cart)实体 E-R 图

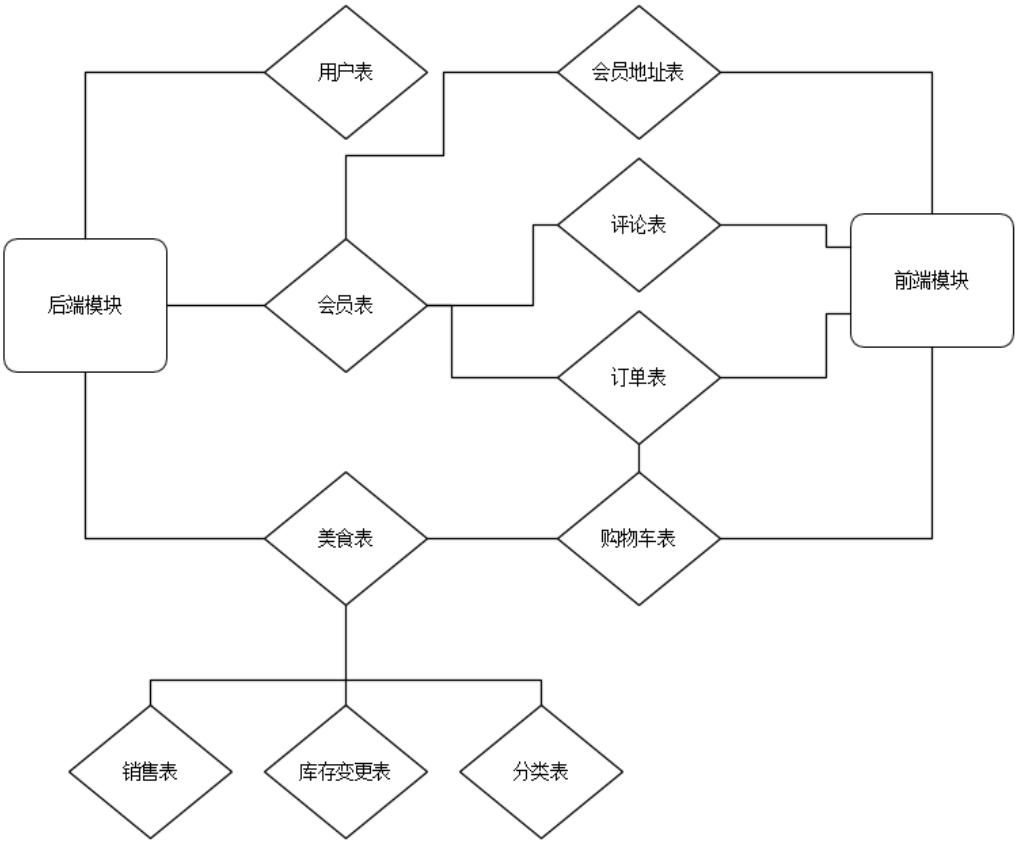

图 4-11 数据库总表关系 E-R 图

4.2.3 数据表结构设计

每个表的表结构设计如表 4-12 至 4-19 所示：

表 4-12 会员评论记录(member_comments)表结构

名	类型	长度	字段含义
id	int	11	主键
member_id	int	11	会员 id
food_ids	varchar	200	商品 ids
pay_order_id	int	11	订单 id
score	tinyint	4	评分
content	varchar	200	评论的内容
created_time	timestamp	无	插入时间

表 4-13 用户信息记录(user)表结构

名	类型	长度	字段含义
uid	bigint	20	主键
nickname	varchar	100	管理员用户名
mobile	varchar	20	手机号码
email	varchar	100	邮箱地址
login_name	varchar	20	登陆用户名
login_pwd	varchar	32	用户登录密码
login_salt	varchar	32	登陆密码的随机加密密钥
status	tinyint	1	有效值，1 是有效，0 是无效
update_time	timestamp	无	最后一次更新时间
created_time	timestamp	无	插入时间

表 4-14 会员用户记录(member)表结构

名	类型	长度	字段含义
id	int	11	主键
nickname	varchar	100	会员名
mobile	varchar	11	手机号码
sex	tinyint	1	性别，1 男 2 女
avatar	varchar	200	头像
salt	varchar	32	随机 salt
status	tinyint	1	有效值，1 是有效，0 是无效
update_time	timestamp	无	最后一次更新时间
created_time	timestamp	无	插入时间

表 4-15 美食记录(food)表结构

名	类型	长度	字段含义
id	int	11	主键
cat_id	int	11	分类 id
name	varchar	100	食品名称
price	decimal	10	售卖金额
main_image	varchar	100	主图
summary	varchar	10000	描述
stock	Int	11	库存量
tags	varchar	200	标签
status	tinyint	1	有效值，1 是有效，0 是无效
update_time	timestamp	无	最后一次更新时间
created_time	timestamp	无	插入时间

表 4-16 美食库存变更(food_stock_change_log)表结构

名	类型	长度	字段含义
id	int	11	主键
food_id	int	11	商品 id
unit	int	11	变更之后总量
total_stock	int	11	总库存
created_time	timestamp	无	插入时间

表 4-17 图片记录(images)表结构

名	类型	长度	字段含义
id	int	11	主键
file_key	varchar	60	文件名
created_time	timestamp	无	插入时间

表 4-18 在线购买订单记录(pay_order)表结构

名	类型	长度	字段含义
id	int	11	主键
order_sn	varchar	40	随机订单号
member_id	bigint	11	会员 id
total_price	decimal	10	订单应付金额
yun_price	decimal	10	运费金额
pay_price	decimal	10	订单实付金额
prepay_id	varchar	120	第三方预付 id
status	tinyint	4	订单状态。1 是支付完成、0 是无效、-1 是申请退款、-2 是退款中、-9 是退款成功、-8 是待支付、-7 是完成支付待确认
express_status	tinyint	4	快递状态。-8 是待支付、-7 是已付款待发货、1 是确认收货、0 是失败
express_address_id	int	11	快递地址 id
express_info	varchar	1000	快递信息
update_time	timestamp	无	最后一次更新时间
created_time	timestamp	无	插入时间

表 4-19 订单详情记录(pay_order_item)表结构

名	类型	长度	字段含义
id	int	11	主键
pay_order_id	varchar	40	订单 id
member_id	bigint	11	会员 id
quantity	decimal	10	购买数量。默认为 1
price	decimal	10	商品总价
food_id	decimal	10	美食表 id
status	tinyint	4	状态值，1 是成功，0 是失败
update_time	timestamp	无	最近一次更新时间
created_time	timestamp	无	插入时间

4.3 本章小结

本章主要介绍了订餐小程序系统运行所需要的环境以及表的总体设计，通过本章希望能让读者了解到订餐小程序系统的基本环境和工具的安装，需要建立哪些对应的主表，从而让读者能通过文字，初步了解订餐小程序系统的初始工作如何操作的。

第五章 系统功能模块的设计与实现

5.1 后端的设计与实现

5.1.1 管理员登陆模块的设计与实现

1.登录流程

登录流程图，如图 5-1 所示，用户若执行登录操作，则服务端会拿着用户输入的账号密码到数据库去查询此账户信息是否正确，如果正确则登录成功，如果错误则需要用户重新校验信息重新登录。登陆界面如图 5-2 所示。

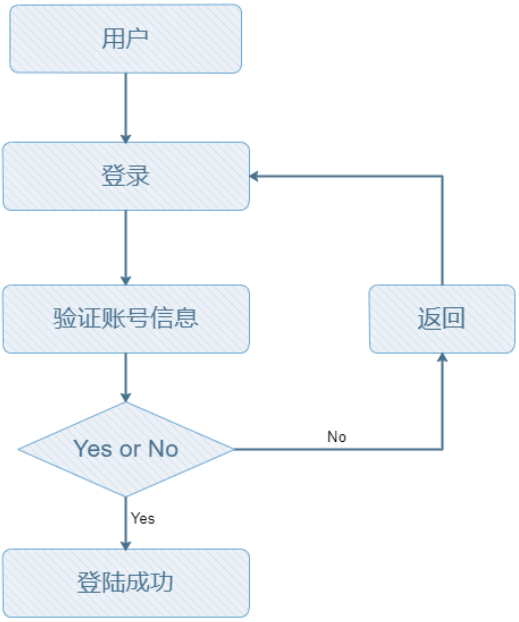

图 5-1 登录流程图

图 5-2 登录界面

2.登陆实现细节

登陆模块主要是通过管理员的用户名和登陆密码来判断这个用户是否存在，首先第一个判断登陆用户名，因为我在 user 表中将 login_name 设置成唯一主键，是全局唯一的。所以不管是否有成千上万的用户，用户名都是唯一的。如果输入的用户名在数据库是已经有的，那么将会登陆失败，反之登陆成功。

第二是校验用户输入的密码，登陆密码是拿用户输入的密码和数据库存储的随机加密密钥进行某种加密的密码得到 login_pwd，再拿去数据库进行对比，如果正确则登录成功，如果错误则需要用户重新校验信息重新登录。加密算法使用哈希 MD5 和 base64 来实现。具体加密算法如下图 5-3。

```
@staticmethod
def genePwd( pwd, salt ):
    m = hashlib.md5()
    str = "%s-%s"%( base64.encodebytes( pwd.encode("utf-8") ) , salt)
    m.update(str.encode("utf-8"))
    return m.hexdigest()
```

图 5-3 密码加密算法

为了能让管理员有更好的交互体验感，本系统使用 AJAX 进行异步提交，AJAX 是一种创建快速动态 web 页面的技术。AJAX 通过在后台与服务器交换少量数据，使 web 页面能够异步更新。这意味着可以在不重新加载整个页面的情况下更新页面的一部分。如果是传统的网页需要更新内容，必需重载整个网页面。具体逻辑实现代码如下图 5-4。

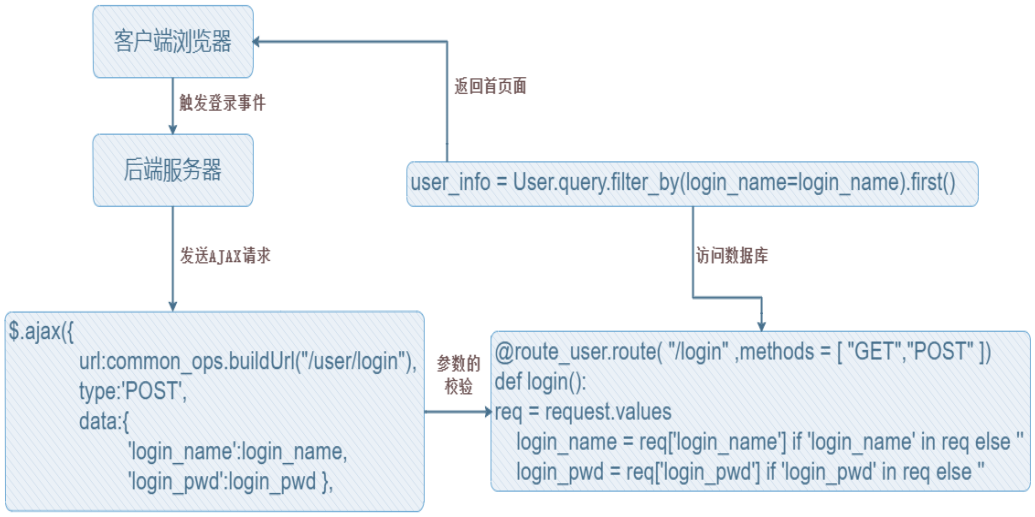

图 5-4 AJAX 异步提交逻辑图

3.拦截器

在登陆成功之后的首页面是需要判断它是否已经登陆的，没有登陆就需要跳转到登陆页面，所以这里需要将用户的登陆类型给存储起来去判断它是否已

经登陆。本系统选择用 cookies 作为响应头来存储它们，而不选用 session 的原因是当有多台服务器进行访问的时候会出现供不应求的情况。Cookies 的原理是，将一些信息保存到客户端，它与每个请求一起带来，服务器通过这些信息确定客户机的身份。由于 cookies 是可以模拟的，所以本系统将 cookies 和 uid 进行字符串拼接并且加密。加密算法是跟登陆密码加密一样的使用哈希 MD5 进行加密，通过用户 uid、用户名、用户密码以及随机加密密钥进行加密。具体代码如下图 5-5。

```
response = make_response(json.dumps({'code': 200, 'msg': '登录成功~~'}))
response.set_cookie( app.config['AUTH_COOKIE_NAME'], '%s#%s' % (
    UserService.geneAuthCode(user_info), user_info.uid),  60 * 60 * 24 * 120)
return response
```

图 5-5 核心代码

但是，此时如果将浏览器里面的 cookies 进行清除，页面是还停留在首页面的。理论上来说是应该跳转到登陆页面的，这迫使我们在每个页面都需要写入方法去判断它是否已经登陆过。由于太过于麻烦，本系统引入拦截器的思想。具体方法和效果如下图 5-6 和图 5-7。

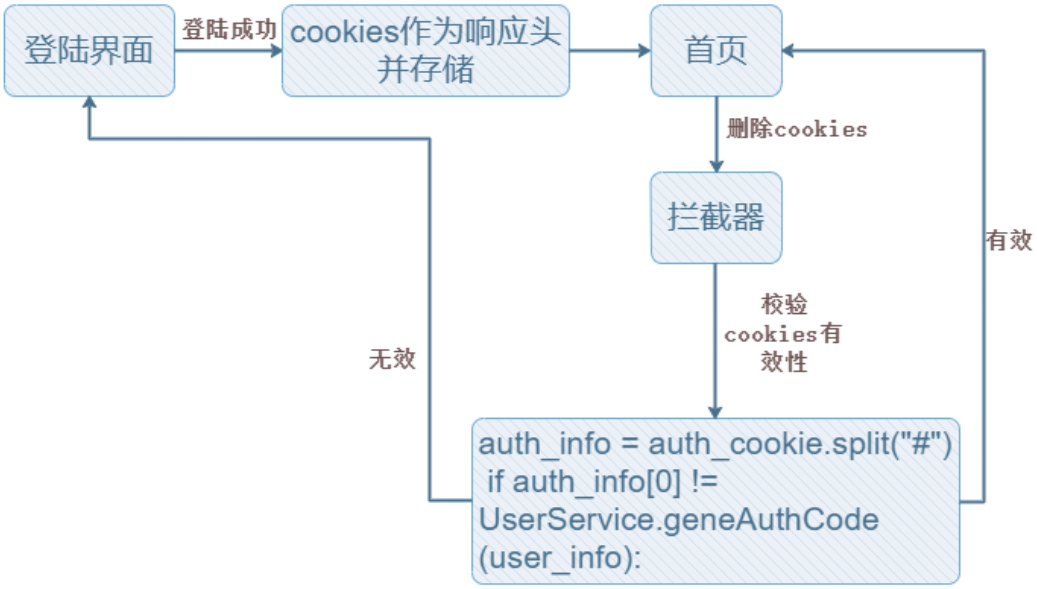

图 5-6 判断是否登陆逻辑图

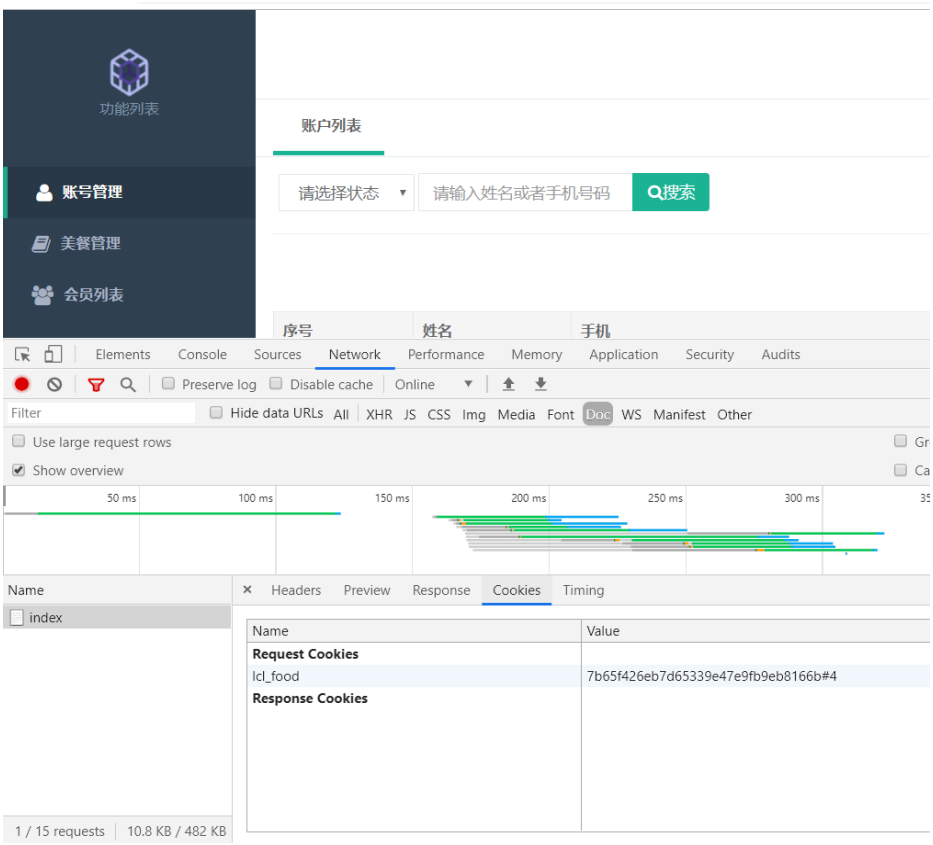

图 5-7 cookies 效果图

这是一个请求之前的方法，在请求所有 controller 里面的方法之前，会通过拦截器拦截下来。原理是在请求页面之前通过判断页面是是否存在 cookies，假设不存在，则跳转到登陆页面，如果存在就不管它。核心就是会从 cookies 里面取出 uid，通过 uid 去数据库查出他的个人信息并且生成一个授权码，最后将这个生成的授权码和 cookies 里面的授权码进行对比，如果一致，那么就代表它是对的，如果不一致就说明这个 cookies 是经过修改的。但是在这个过程中，有一些页面是不需要去判断它是否已经登陆的，例如 static 静态文件、favicon.ico 图标，切记，在登陆页面也是不需要去判断登陆的，不然会发生多次重定向。

5.1.2 管理员信息编辑模块的设计与实现

1.账号信息编辑

首先在 edit.html 里最外层包裹一个 user_edit_wrap 的 JS 绑定事件，通过 click 点击事件来获取传递的变量，例如昵称和手机号码。同样的都需要先进行 JS 的参数有效性判断。参数有效性判断成功通过后就可以发送 AJKX 请求到后端。另外需要记得将后端的 methods 方法改为 get 和 post，因为它默认是 get 方法。当请求的方法是 get 的时候就执行页面展示功能，如果是 post 请求就需要进行语法处理。效果如下图 5-8。