**学号：2021013214**



**2025 届本科生毕业论文**

在线订餐平台智能推荐系统

设计与实现

|  |  |
| --- | --- |
| 学院： | 信息工程学院 |
| 专业： | 软件工程 |
| 年级班级： | 2021级03班 |
| 学生姓名： | 李佐彤 |
| 指导教师： | 代媛 |
| 协助指导教师： |  |
| 完成日期： | 2025年06月 |

**本科生毕业论文（设计）的独创性声明**

本人声明：所呈交的本科毕业论文（设计）是我个人在导师指导下独立进行的研究工作及取得的研究结果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究结果，也不包含其他人和自己本人已获得西北农林科技大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同事对本研究所做的任何贡献均已在论文的致谢中作了明确的说明并表示了谢意。如违反此声明，一切后果与法律责任均由本人承担。

本科生签名： 时间： 年 月 日

**关于本科生毕业论文（设计）知识产权的说明**

本毕业论文（设计）的知识产权归属西北农林科技大学。本人同意西北农林科技大学保存或向国家有关部门或机构送交论文的纸质版和电子版，允许论文被查阅和借阅。

本人保证，在毕业离开西北农林科技大学后，发表或者使用本毕业论文（设计）及其相关的工作成果时，将以西北农林科技大学为第一署名单位，否则，愿意按《中华人民共和国著作权法》等有关规定接受处理并承担法律责任。

任何收存和保管本论文各种版本的其他单位和个人（包括作者本人）未经本论文作者的导师同意，不得有对本论文进行复制、修改、发行、出租、改编等侵犯著作权的行为，否则，按违背《中华人民共和国著作权法》等有关规定处理并追究法律责任。

本科生签名：　　　　　　 时间：　　　年　　　月　　　日

指导教师签名：　　 　　 　 时间：　　　年　　　月　　　日

在线订餐平台智能推荐系统设计与实现

摘 要：当前我国已全面迈入互联网时代，海量的网络信息既错综复杂又相互关联。传统关系型数据库在表示实体间关联关系时存在明显的冗余问题，这使得其在处理复杂关联时显得力不从心。相较之下，图数据库能够有效解决这一难题。知识图谱作为一种典型的图数据库，在表示实体间关联关系方面具有独特优势。为提升订餐推荐效率，本研究基于菜品信息、套餐组合、用户历史行为等多维度属性及其关联关系，构建了专业的外卖订餐推荐知识图谱，并实现了配套的在线智能推荐系统。

在系统构建过程中，首先对订餐业务流程进行了全面梳理，重点聚焦管理员与用户两大角色，其中推荐结果主要面向用户端展示。同时结合推荐内容的具体需求，对餐品类型进行了系统性的调研与分类整理。

在技术实现层面，创新性地采用自顶向下与自底向上相结合的设计方法，构建了订餐智能推荐知识图谱。该方法明确了模式层中实体及其关系的分类与定义规范，并利用Neo4j图数据库高效存储和管理知识图谱数据，实现了图谱信息的可视化展示功能。

系统架构方面，采用B/S三层架构模式设计智能推荐系统：前端基于VUE框架和Element UI组件库开发表示层；后端采用SpringBoot框架实现业务逻辑层。最终实现的系统能够完整支持基于知识图谱的智能推荐流程，并能根据用户行为实时更新推荐结果。

关键词：订餐智能推荐系统；知识图谱；Vue；Spring Boot；Neo4j

Online food ordering platform intelligent recommendation system Design and implementation

**Abstract:** At present, China has fully entered the Internet age where information exhibits complex interconnections. Traditional relational databases prove inefficient in representing entity relationships due to their inherent redundancy, making them less suitable for handling complex associations. In this context, graph databases emerge as an effective solution, with knowledge graphs serving as a specialized form that excels at representing intricate relationships between entities.

To optimize food ordering recommendations, this research establishes a dedicated knowledge graph for takeout recommendations by analyzing dish attributes, meal combinations, and user behavior patterns. The study simultaneously develops an intelligent online recommendation system to complement this knowledge graph. The development process begins with a thorough examination of ordering workflows, paying particular attention to the distinct roles of administrators and end-users, where recommendation results are primarily presented to users. This analysis is combined with an in-depth investigation of food categories based on specific recommendation requirements.

The technical implementation adopts an innovative dual approach that integrates both top-down and bottom-up design methodologies. This hybrid approach clearly defines entity classifications and relationship specifications within the schema layer. The Neo4j graph database serves as the foundation for efficient data storage and management, while also enabling effective visualization of the knowledge graph's information network.

The system architecture follows a three-tier B/S model, featuring a presentation layer constructed with the VUE framework and Element UI components, coupled with a business logic layer implemented through the SpringBoot framework. The completed system demonstrates robust capabilities in executing comprehensive recommendation processes based on the knowledge graph, featuring real-time updates that respond to evolving user behavior patterns, while effectively presenting the complex network of relationships between food items and users.

**Keywords:** Intelligent recommendation system for ordering; knowledge graph; Vue; Spring Boot; Neo4j

目 录

[第1章 绪论 - 1 -](#_Toc9780)

[1.1 研究背景与意义 - 1 -](#_Toc11855)

[1.2 研究（应用）现状 - 1 -](#_Toc15422)

[1.3 主要研究内容 - 2 -](#_Toc24160)

[1.3.1 b端和c端业务功能实现 - 3 -](#_Toc6169)

[1.3.2 智能推荐算法的设计与实现 - 4 -](#_Toc16064)

[1.4 论文组织结构 - 5 -](#_Toc29382)

[第2章 相关理论与方法 - 6 -](#_Toc5388)

[2.1 知识图谱基本理论 - 6 -](#_Toc1413)

[2.1.1 知识图谱相关概念 - 6 -](#_Toc13275)

[2.1.2 知识图谱的结构 - 6 -](#_Toc17993)

[2.2 知识图谱在SpringBoot框架的应用 - 7 -](#_Toc8348)

[2.2.1 Neo4j数据库的相关概念 - 7 -](#_Toc24322)

[2.2.2 将Neo4j数据库引入Spring Boot框架 - 8 -](#_Toc29985)

[第3章 知识图谱的设计与智能推荐方法 - 9 -](#_Toc21119)

[3.1 知识图谱模式层设计 - 9 -](#_Toc24607)

[3.2 数据的采集及存储 - 9 -](#_Toc30528)

[3.2.1 数据的采集 - 9 -](#_Toc7757)

[3.2.2 数据存储 - 10 -](#_Toc7817)

[3.2.3实体入库存储 - 10 -](#_Toc19868)

[3.3 智能推荐方法 - 14 -](#_Toc16704)

[3.3.1 推荐方法的基本概念 - 14 -](#_Toc29484)

[3.3.2 推荐算法的实现 - 14 -](#_Toc18574)

[第4章 在线订餐智能推荐系统的设计与实现 - 17 -](#_Toc4584)

[4.1 需求分析 - 17 -](#_Toc30863)

[4.2 系统设计 - 17 -](#_Toc24295)

[4.2.1 系统架构设计 - 17 -](#_Toc30116)

[4.2.2 用户小程序模块设计 - 19 -](#_Toc18958)

[4.2.3 智能推荐模块 - 19 -](#_Toc24859)

[4.2.4 管理端模块设计 - 20 -](#_Toc1157)

[4.2.5 数据库设计 - 20 -](#_Toc12358)

[4.3 系统实现 - 23 -](#_Toc15348)

[4.3.1 表示层实现 - 23 -](#_Toc14661)

[4.3.2 业务层实现 - 27 -](#_Toc14513)

[4.3.5 数据层实现 - 28 -](#_Toc2819)

[第5章 总结与展望 - 29 -](#_Toc15932)

[5.1 总结 - 29 -](#_Toc4663)

[5.2 展望 - 29 -](#_Toc18622)

[参考文献 - 31 -](#_Toc23411)

[致 谢 - 33 -](#_Toc22377)

# 第1章 绪论

## 1.1 研究背景与意义

随着现代互联网技术的迅速发展,全球进入一个高速发展的信息化时代。人们的生活节奏也在不断加快,大家希望拥有较为快捷的生活服务，因此出现了许多的订餐系统，极大提高了人们的生活质量。相对于传统的线下点餐，订餐系统更加符合现代人们生活的需求，用户只需通过手机就能够实现点餐操作，同时也可以有效降低高峰期排队点餐和等餐时间，提高顾客就餐的满意度。不仅如此，点餐系统可以融入许多个性化的东西，使其贴合每个用户的饮食喜好，变得“智能”。

近年来，知识图谱(Knowledge Graph, KG)技术不断发展，作为推动人工智能的核心驱动力，具有高效的语义处理功能，为人工智能处理各领域中的问题都提供了可能。知识图谱以实体概念为节点，以关系为边，能够以一种直观和可视化方式揭示知识及其之间的复杂关联。

将知识图谱利用在推荐领域，可以利用用户历史行为，产品特征等进行有效推荐。本研究主要利用知识图谱对已有的套餐，菜品之间的关系进行梳理，再将用户的历史行为，用户喜好作为条件进行最终的推荐。基于知识图谱设计订餐推荐也是将知识图谱作用于餐饮行业的一个新的尝试，有利于知识图谱本身的发展，也有利于用户订餐效率的提升。

## 1.2 研究（应用）现状

就餐饮业的发展速度而言，各地区经济发展程度不同，发展速度及需求就不一样。经济发达的地区远比经济落后的地区要强很多，这是因为经济发达地区的人们时间观念较强,同时对服务要求也较高（吕倩等2021）。正是在这种需求下，推进了“互联网+餐饮”的发展，通过运用先进的管理方法，孕育出移动点餐平台，这给消费者和商家带来了很多的便利，促进了餐饮业的发展。在国外，将信息技术应用到餐饮业中的技术已经非常成熟，通过点餐系统，消费者有充裕的时间进行浏览餐馆菜单，进行比价，选择性价比较高的美食进行购买。目前，大多数餐馆通过网络订餐销售额已大于到店消费销售额。在国内，餐饮业的信息化管理发展相对于国外比较晚，也陆续出现了网上订餐平台，这些餐饮服务平台的出现满足了消费者的个性化需求，对于餐馆提高质量的食品和服务起到至关重要的作用（Omer Gibreel et al. 2018）。与此同时，订餐系统智能化也是近些年来十分热门的话题。

智能订餐推荐系统主要根据用户的历史行为、偏好以及餐品特征，为用户提供个性化的餐饮推荐。随着移动互联网和外卖行业的快速发展，越来越多的餐饮平台（如美团外卖、饿了么等）利用这些技术优化用户体验，提升平台的活跃度和订单量。现有的推荐主要依赖两种基本方法：协同过滤和内容推荐（赵敬宇等 2020）。协同过滤通过分析不同用户的行为相似性，进行餐品推荐；内容推荐则根据餐品的属性（如菜系、口味、热量等）以及用户的偏好进行匹配（Wang, X. et al.2020）。

然而，尽管这些方法在一定程度上提升了推荐的效果，但仍然面临一些挑战。例如，协同过滤方法在面对用户数据稀缺（如冷启动问题）时，推荐效果较差；而内容推荐则往往过于依赖固定的规则，缺乏足够的灵活性来适应用户的动态需求。此外，现有系统推荐的餐品种类有限，缺乏足够的多样性，用户的推荐体验可能会受到局限。

为了提升推荐效果和解决上述问题，基于知识图谱的推荐系统逐渐成为研究的重点。知识图谱通过构建餐品、用户和其他相关实体（如商家、时间、地点等）之间的关系网络，帮助系统更全面地理解餐饮数据和用户需求（Wang, X. et al.2020）。它通过整合多种数据源（如用户评价、社交媒体反馈、位置数据等），不仅能提供更准确的推荐，还能够提升推荐结果的可解释性，帮助用户理解推荐背后的逻辑。与协同推荐相比，知识图谱能够有效缓解冷启动问题，因为它允许系统在缺乏大量用户行为数据的情况下，仍能基于餐品的多维属性和外部信息进行推荐（Almudena Sánchez et al. 2018）。

近年来，知识图谱在多个领域的应用得到了显著发展，尤其是在智能推荐系统中的应用取得了显著进展。将知识图谱应用于推荐，本质使一种知识推理任务，依据知识图谱已有的知识，经过知识图谱里面的一些推理动作来满足向用户推荐的需求。Lao和 Cohen在 2010年（Lao, N. and William, W. 2010)首次提出基于路径排序的推理方法，该方法本质上是一种随机游走推理方法，算法核心是将连接两个实体之间的路径作为特征来预测实体之间潜在的关系。知识图谱通过将领域知识和用户行为数据进行整合，能够提供更加精准和个性化的推荐。例如，在电商领域，知识图谱可以将产品、品牌、类别、用户兴趣等多维度信息进行关联，从而生成个性化的推荐列表，提高推荐的相关性和精准度（Xia, F. et al. 2019）。引入深度学习技术，结合知识图谱进行推荐，能够进一步提升系统的智能性和预测准确度（Zhao, Y. et al. 2023）。除了餐饮，商品类的推荐，知识图谱在电影、音乐、新闻等领域的推荐系统中也表现出了很好的效果，能够超越传统的推荐算法，为用户提供更具多样性和个性化的选择。

随着深度学习和实时数据分析技术的进步，未来的智能订餐推荐系统将会更加智能和精准。深度学习能够从复杂的用户行为和餐品特征中提取更高层次的模式，而实时数据分析能够根据用户的实时需求和环境变化做出即时调整。通过结合这些先进的技术，智能订餐系统将能够实现真正的个性化推荐，从而极大地提高用户体验，推动外卖行业的进一步发展和创新（Xie, X. et al. 2021）。

## 1.3 主要研究内容

本文的研究主要包括订餐平台开发和订餐菜品推荐知识图谱的建立。整体的技术路线如图1-1所示。

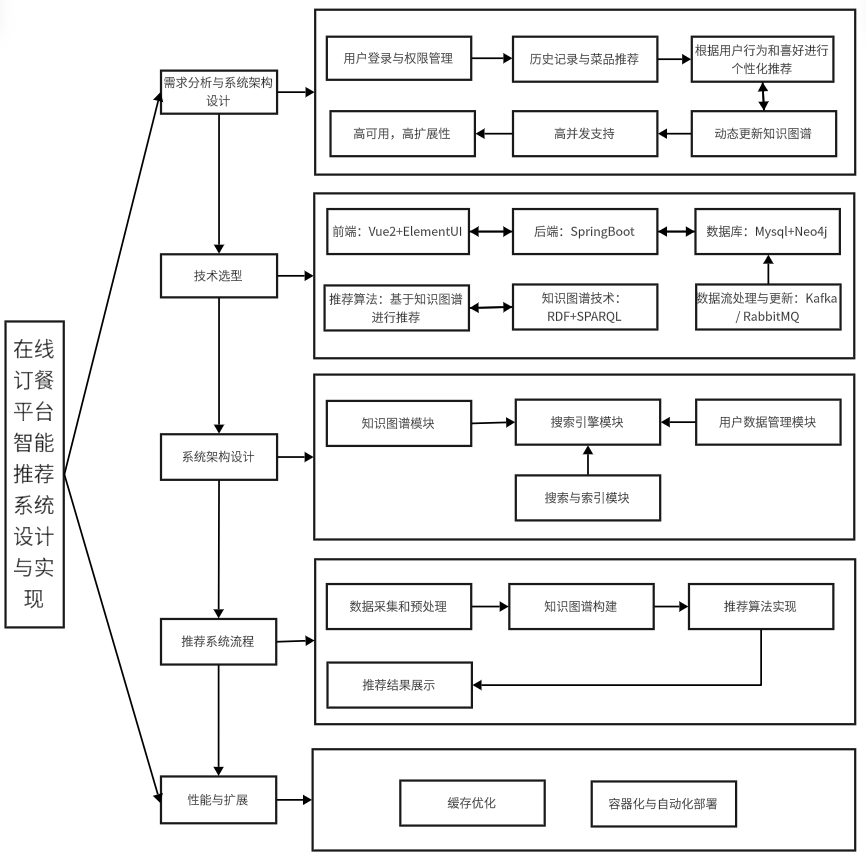


图1-1 技术路线图

首先需要保证数据的完整性和准确性，在进行本次研究之前，需要对附近的商家进行调研，并对得到的数据进行预处理。本次研究不仅要实现传统的外卖点餐逻辑，实现智能推荐，还要兼顾软件的性能。传统的软件分为b端和c端，分别用于处理企业管理和用户使用，b端主要实现管理员的功能，c端主要实现用户前台功能。软件的性能主要基于中间件Redis和推荐算法的优化进行实现。智能推荐主要通过知识图谱进行实现，我们打算构建菜品，用户行为，套餐，历史订单等多种实体的知识图谱。通过知识图谱中的实体关系，能更好地进行冷启动问题地处理，提升用户体验。

### 1.3.1 b端和c端业务功能实现

后台（b端）业务功能如图1-2所示。

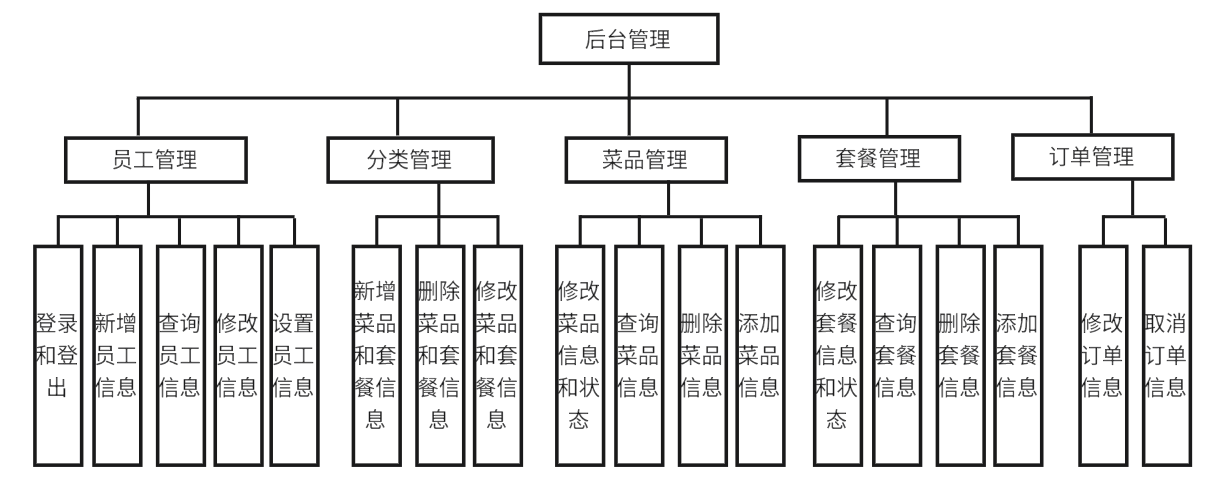


图1-2 后台功能结构图

前台（c端）业务功能如图1-3所示。

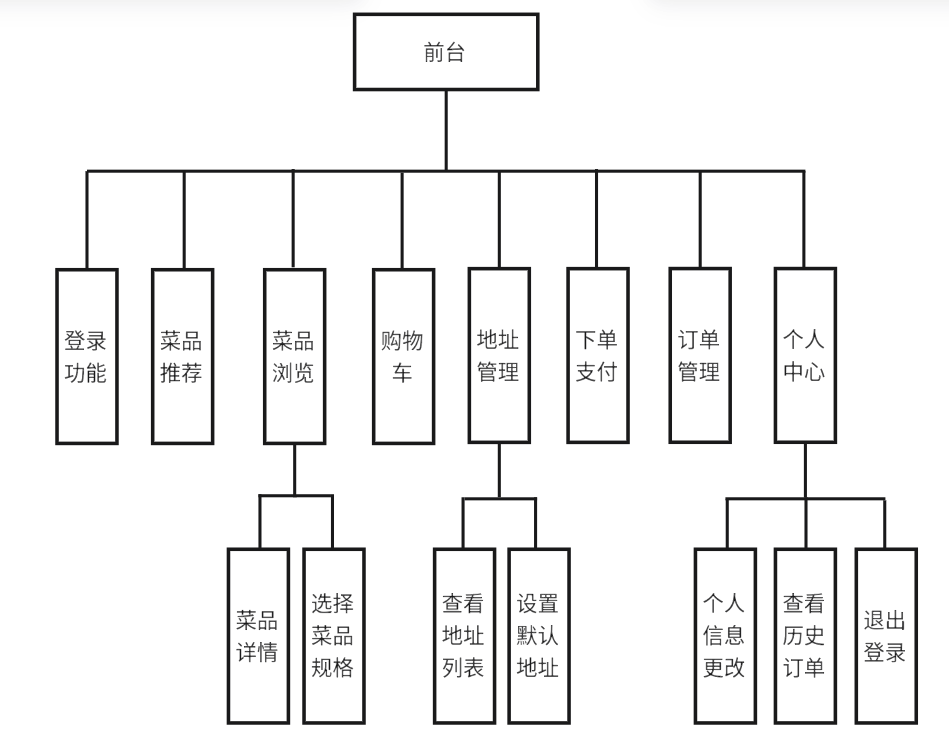
****

图1-3 前台功能结构图

### 1.3.2 智能推荐算法的设计与实现

智能推荐算法我们打算基于知识图谱进行实现。传统的推荐主要使用协同过滤，这种方法可以进行有效推荐，但是缺点也比较明显，冷启动的速度较慢。为了兼顾性能，并且提高推荐的准确性和多样性，我们打算构建菜品，用户行为，套餐，历史订单等多种实体的知识图谱。通过知识图谱中的实体关系，能更好地进行冷启动问题地处理（Zhang, L. et al. 2019）。

知识图谱的实体主要用Neo4j数据库进行存储，打算将与智能推荐相关的实体存到图数据库中，Mysql中存放其他业务的数据。由于推荐同时涉及到两个数据库的查询和修改，所以需要用Redis暂存当前的推荐结果以确保系统本身的稳定性和时效性。

推荐的实时性也是很重要的一方面，即当实体变化时，推荐的结果需要动态调整。例如：当顾客的口味，喜好因某种原因发生变化时，我们需要根据该顾客新的喜好以及近期的订单即使做出调整。

## 1.4 论文组织结构

本文共有五章，各章节内容如下：

第一章为绪论部分，阐述了基于知识图谱的外卖订餐系统的研究背景以及研究意义，介绍了知识图谱发展以及目前在订餐智能推荐方面应用到的程度，概括了研究的主要内容以及应用到的技术和方法，最后介绍了论文的组织结构。

第二章介绍了本文相关的基本理论和方法，主要包括知识图谱的基础理论以及自然语言处理中分词和依存句法分析的基础理论。

第三章阐述了梨果病虫害知识图谱的构建过程，主要包含梨果病虫害知识的采集、模式层设计、半结构化数据预处理和数据入库等。

第四章介绍了图形化诊断软件的设计思路以及采用的技术框架和技术路线，并展示主要功能界面。

第五章对本文的工作进行了总结，并对未来工作进行了展望。

# 第2章 相关理论与方法

## 2.1 知识图谱基本理论

知识图谱的概念是由Google在2012年首次提出的，其目的主要为智能搜索推荐服务提供数据资源。随着互联网的迅猛发展和计算机性能的不断提升，大数据、云计算和区块链等技术促使全球进入数据时代。面对大量杂乱无章的数据，如何有效的提取相关信息，使数据发挥作用，并有效管理和联想相关的内容，是目前研究的热点，知识图谱的提出为解决这些问题提供了新思路。

### 2.1.1 知识图谱相关概念

知识图谱是知识工程的一个分支，属于一种结构化的语义知识库。它的基本组成单元是“实体-关系-实体”三元组，以及与实体相关的属性键值对。实体通过关系相互连接，形成一个网络状的图谱结构。要理解知识图谱，首先需要掌握其中一些关键术语的含义，以下对这些名词的介绍有助于更好地理解知识图谱。

（1）实体：实体是知识图谱中的基本单元。代表现实世界存在的个体，例如用户点餐时下单的菜品和套餐等就是实体。实体在知识图谱中以节点的形式存在。

（2）属性：在知识图谱中，属性用于描述实体的性质或特征。每个实体都有一个唯一的标识符以及若干属性，通过这些属性可以将实体划分为不同的类别，例如在“菜品”节点加上“类别”属性，表明其为某个类别的菜品。

（3）关系：关系用于连接两个不同的实体。在现实世界中，实体与实体之间并非完全独立，它们通常存在某种关系。因此，知识图谱通过关系将两个实体连接起来。在知识图谱中，关系作为图中的边，并且是有向的。

（4）知识：由“实体-关系-实体”这样的一个三元组构成的一条数据就是一条知识。它解释了某两个实体之间的关系或者表明了某两个实体具有关联性。

（5）知识库：知识库是由多条知识构成的集合。在知识图谱中，通常不会只有单一的“实体-关系-实体”这样的知识。由于一个实体可能与多个实体存在关系，许多实体通过关系相互连接，最终形成一个知识库。从数据结构的角度来看，知识库可以视为一个包含大量节点的图结构。

### 2.1.2 知识图谱的结构

知识图谱的结构主要指逻辑结构和体系架构（徐增林等2016）。从知识图谱的逻辑层面来看，可以将其划分为两个层次：模式层和数据层。模式层负责定义知识图谱中存在的实体及其之间的关系类型。例如订餐智能推荐知识图谱的实体主要有订餐菜品，套餐，菜品种类，套餐种类等。数据层由一系列事实组成，遵照模式层规定的规范约束存储。例如菜品名称实体有“巴西牛排”和“黄焖鸡”等。在数据层，将与订餐智能推荐相关的信息作为独立的实体，抽象出它们之间的关系，并使用知识图谱工具进行存储。通过可视化工具查看时，显示的内容就是数据层的具体信息。数据层的数据可以通过图数据库进行存储，本文采用Neo4j数据库进行存储和查看。在完成基础数据存储后，可以根据实体的属性对知识库中的数据进行进一步结构化和规范化，形成数据的概念模板。经过规范化处理后，符合模板的数据即构成模式层数据。

知识图谱的体系架构指的是构建知识图谱的方式和过程。构建知识图谱的方法有多种，Douglas（Douglas BLenat 1995）通过程序对这些方法进行组织，Freebase（Kurt Bollacker et al. 2008）和Wikidata（Denny Vrandeci and Markus Krötzsch 2014）则通过人工标注来构建，DBpedia和YAGO（Fabian M.Suchanek et al. 2007）则从大规模半结构化的Web知识库中提取知识进行构建。总体而言，构建知识图谱有两种方式：自顶向下和自底向上。自顶向下的方式是在构建知识图谱之前，先抽象出实体的概念和属性，然后根据这些概念来创建实体并将其加入数据库。这种方式通常需要领域专家来定义实体的模板。自底向上的方式则是在没有明确实体概念的情况下使用，首先获取数据以形成数据层的内容，然后通过一定的归纳和管理去除冗余数据和置信度低的数据，逐步构建上层的知识图谱。

订餐智能推荐的知识图谱适合采用自底向上的构建方式，因为该场景的数据通常来自于用户行为、餐馆信息、菜品信息、用户评价等多个来源，这些数据大多是半结构化或非结构化的，适合通过自底向上的方式进行采集和处理。自底向上的方法可以灵活地从大量数据中提取知识，分析用户的点餐历史、口味偏好等信息，并逐步发现潜在的关系和模式。此方式还能够实时适应需求变化，并在知识图谱构建过程中不断优化推荐算法。

## 2.2 知识图谱在SpringBoot框架的应用

在Spring Boot中应用知识图谱，主要是利用Spring Boot的灵活性和强大的生态系统来构建和集成知识图谱服务。本文主要将Neo4j集成在Spring Boot中，利用图数据库和推荐算法产生针对不同用户的推荐结果。

### 2.2.1 Neo4j数据库的相关概念

Neo4j 是一个开源的图数据库，专门用于处理和查询复杂的关系数据。与传统的关系型数据库不同，Neo4j 采用图数据模型，通过节点、边和属性来存储数据，特别适合表示和分析事物之间的关系（陈韶健 2017）。它使用 Cypher 查询语言，这种语言的语法类似 SQL，但专门为图数据设计，使得查询复杂的图形数据结构更加高效。Neo4j 在社交网络分析、推荐系统、知识图谱、欺诈检测等多个领域都有广泛应用，尤其是在需要处理复杂的连接查询时表现出色。由于其灵活的数据建模方式和高效的查询引擎，Neo4j 可以应对大规模图数据存储和查询的需求，并支持分布式部署以满足企业级需求，它适合需要动态调整数据模型和处理复杂关系的应用。

### 2.2.2 将Neo4j数据库引入Spring Boot框架

在Spring Boot中接入Neo4j数据库可以通过引入spring-boot-starter-data-neo4j依赖并配置Neo4j连接信息实现。

首先，创建一个用@NodeE注解的实体类表示图数据库中的节点，然后定义继承Neo4jRepository的Repository接口来进行CRUD操作。

接着，可以在服务层封装数据访问逻辑，并在控制器中处理HTTP请求。

最后，确保Neo4j数据库正在运行，使用Cypher查询增强功能来执行复杂查询。这样便能在Spring Boot应用中顺利使用Neo4j进行数据存取。

# 第3章 知识图谱的设计与智能推荐方法

本章主要介绍在线智能订餐知识图谱的构建方式与过程，采用自顶向下和自底向上相结合的构建方法，构建过程主要包含初步确定知识图谱模式层，数据的采集和稳定存储，数据的预处理，构建实体间关系，更新知识图谱模式层，数据入库存储等步骤。

## 3.1 协同过滤模式层设计

本文面向图书馆图书推荐场景，结合协同过滤的核心理论与业务需求，参考张亮（2023）在电商推荐中提出的用户-商品交互建模方法，并融合王芳等（2022）的混合协同过滤框架，构建面向图书馆的协同过滤模式层。模式层定义了协同过滤模型的实体类型、交互关系与属性约束，如表3-1和表3-2所示。

表3-1 用户图书协同过滤模式层实体

|  |  |  |
| --- | --- | --- |
| 实体类别 | 含义描述 | 属性示例 |
| 用户 | 菜品实体 | 用户ID，活跃度，兴趣标签，借阅频次 |
| 图书 | 菜品所在种类 | 名称，分类，关键词，概述 |
| 交互 | 由若干菜品组成的套餐 | 借阅次数，借阅时长，评分，时间戳 |
| 隐向量 | 潜在特征向量，通过矩阵分解生成 | 更新周期，相似度 |

表3-2 关系类型

|  |  |  |
| --- | --- | --- |
| 关系类型 | 含义描述 | 描述 |
| 借阅 | 用户—图书 | 用户对图书的借阅行为 |
| 相似 | 用户之间—图书之间 | 基于行为和特征向量的相似性关系 |
| 预测 | 隐向量—图书 | 借阅次数，借阅时长，评分，时间戳 |
| 关联 | 图书—关键词 | 更新周期，相似度 |
| 影响 | 交互—隐向量 | 用户借阅行为触发隐向量更新 |

## 3.2 数据的采集及存储

### 3.2.1 数据的采集

本文的数据（主要是商家的菜品和套餐数据）采集主要使用线上调研和实地调研两种形式进行。

线上调研主要通过互联网工具收集数据，通过选择一些常见的电商平台，如美团，饿了么等，这些平台上的商家菜单信息较为完整，并且更新较快。在平台搜索商家的店铺，查看其菜单，套餐内容，价格等方面的信息。除此之外，还会浏览小红书，微博，抖音等社交平台，查看顾客分享的菜品照片，视频和评价。

实地调研主要通过访问商家店铺，与电源沟通等方式获取第一手数据。进入餐厅或者外卖店铺，可以通过查看菜单，并记录菜品和套餐的内容，价格，促销信息等。也可以向店员询问热门菜品，套餐组合及价格，也可以询问它们对当前菜单和套餐设置的思路。

通过以上手段拿到数据之后，通过Excel进行分类整理。随后将这些数据通过Api接口放入Neo4j和Mysql数据库。

### 3.2.2 数据存储

拿到这些数据之后，首先要整理和清洗数据，确保其格式统一并去除无效项，并将源数据转化为 CSV 格式，确保所有字段完整并符合预期。之后使用 MySQL 创建数据库和表结构，导入商家、菜品、套餐等数据。通过 LOAD DATA INFILE 命令，将 CSV 数据导入 MySQL 中并进行验证，在 Spring Boot 项目中集成 Neo4j，将菜品、套餐、商家等数据存储到 Neo4j 图数据库中，使用 Spring Data Neo4j 定义实体类和 Repository，并通过 API 接口导入数据。然后使用 Docker 启动 Neo4j 实例，验证数据是否成功导入，通过 Neo4j Browser 执行 Cypher 查询检查数据和关系的准确性。通过这一系列流程，能够高效地管理和分析商家、菜品、套餐等之间的关系。

### 3.2.3实体入库存储

在数据处理的过程中，抽取实体的同时，自底向上的更改了模式层的初步设计。最终的模式层实体定义和关系定义如表3-3和表3-4所示。模式层共定义3个实体，主要包括菜品和套餐相关的信息。

表3-3 模式层的实体定义

|  |  |  |
| --- | --- | --- |
| 实体类别 | 含义描述 | 实体定义 |
| 菜品 | 菜品实体 | DishNeo4j |
| 菜品种类 | 菜品所在种类 | CategoryNeo4j |
| 套餐 | 由若干菜品组成的套餐 | SetmealsNeo4j |

表3-4 模式层的关系定义

|  |  |  |  |
| --- | --- | --- | --- |
| 关系类别 | 关系定义 | 头实体 | 尾实体 |
| 菜品所属种类 | InCatrgory | 菜品 | 菜品种类 |
| 菜品所在套餐 | InSetmeals | 菜品 | 套餐 |

这里为了降低知识图谱数据库的负担，所有的实体均使用相应的实体id，最后的推荐结果将根据知识图谱所返回的id列表进行查询生成，实体的具体定义存放在MYSQL中，菜品信息表，菜品分类表，套餐信息表的定义如表3-5，表3-6和表3-7所示。

表3-5菜品信息表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 字段描述 | 数据类型 | 长度 | 是否主键 | 约束 | 备注 |
| id | 主键 | bigint | - | 是 | not null | 唯一约束 |
| name | 菜品名称 | varchar | 64 | 否 | - | - |
| category\_id | 菜品分类id | bigint | - | 否 | not null | - |
| price | 菜品价格 | decimal | 10.2 | 否 | - | - |
| code | 商品码 | varchar | 64 | 否 | not null | - |
| image | 图片url | varchar | 200 | 否 | not null | - |
| description | 菜品描述 | varchar | 400 | 否 | - | - |
| status | 状态 | int | - | 否 | not null | 1. 停售 2. 起售 |
| sort | 顺序 | int | - | 否 | not null | - |
| create\_time | 创建时间 | datetime | - | 否 | not null | - |
| update\_time | 修改时间 | datetime | - | 否 | not null | - |
| create\_user | 创建人 | bigint | - | 否 | not null | - |
| update\_user | 修改人 | bigint | - | 否 | not null | - |

表3-6菜品分类信息表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 字段描述 | 数据类型 | 长度 | 是否主键 | 约束 | 备注 |
| id | 主键 | bigint | - | 是 | not null | 唯一约束 |
| type | 类型 | int | - | 否 | - | 1. 菜品分类 2. 套餐分类 |
| name | 分类名称 | varchar | 64 | 否 | not null | - |
| sort | 顺序 | int | - | 否 | not null | - |
| create\_time | 创建时间 | datetime | - | 否 | not null | - |
| update\_time | 修改时间 | datetime | - | 否 | not null | - |
| create\_user | 创建人 | bigint | - | 否 | not null | - |
| update\_user | 修改人 | bigint | - | 否 | not null | - |

表3-7套餐信息表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 字段描述 | 数据类型 | 长度 | 是否主键 | 约束 | 备注 |
| id | 主键 | bigint | - | 是 | not null | 唯一约束 |
| name | 菜品名称 | varchar | 64 | 否 | - | - |
| category\_id | 菜品分类id | bigint | - | 否 | not null | - |
| price | 菜品价格 | decimal | 10.2 | 否 | - | - |
| code | 商品码 | varchar | 64 | 否 | not null | - |
| image | 图片url | varchar | 200 | 否 | not null | - |
| description | 菜品描述 | varchar | 400 | 否 | - | - |
| status | 状态 | int | - | 否 | not null | 1. 停用 2. 启用 |
| create\_time | 创建时间 | datetime | - | 否 | not null | - |
| update\_time | 修改时间 | datetime | - | 否 | not null | - |
| create\_user | 创建人 | bigint | - | 否 | not null | - |
| update\_user | 修改人 | bigint | - | 否 | not null | - |

本文使用Neo4j数据库来存储和管理知识图谱，以实现订餐智能推荐系统的构建。首先，在模式层定义了知识图谱的基本结构，即“头实体-关系-尾实体”三元组，确保数据存储符合图谱的基本规则。构建知识图谱的程序使用Java语言实现，并包含多个关键操作，如连接数据库、预定义实体列表、创建节点、构建关系以及关闭数据库连接等。

在创建节点的过程中，我们遍历实体列表，将其中的实体逐一加入到Neo4j数据库中，形成相应的实体节点。这些节点可以是用户、菜品、套餐、菜品种类等实体。在创建关系时，除了频繁出现的月份和关键词等特殊关系外，大多数关系都是一对一关系，确保图谱结构清晰且符合逻辑。例如，套餐与菜品之间的关系是多对多的关系，一个套餐实体可以与多个菜品实体相关联，同时，一个菜品也可能属于多个套餐。因此，在图谱中，一个套餐实体可以作为多个菜品实体的尾节点。而菜品种类与菜品之间则是典型的一对多关系，即一个菜品种类可以包含多个菜品，但每个菜品只属于一个菜品种类。

通过以上步骤，系统逐步建立了完整的订餐智能推荐知识图谱。数据关系的创建完成后，整个图谱得以构建，并可以为推荐系统提供丰富的实体和关系信息，进一步支撑智能推荐的功能。最后，所有数据最终存入数据库并经过统计整理，具体的入库数据和统计结果如表3-5所示，体现了系统在知识图谱构建过程中所处理的数据量与结构化效果。通过这种方式，推荐系统可以更好地理解用户需求，实现个性化推荐，从而提升用户的订餐体验和满意度。

表3-5 知识图谱入库数据统计

|  |  |  |  |
| --- | --- | --- | --- |
| 实体 | 实体数量 | 关系 | 关系数量 |
| 菜品 | 42 | 菜品种类 | - |
| 菜品种类 | 42 | 菜品 | 42 |
| 套餐 | 105 | 菜品 | 236 |

在构建知识图谱的过程中，节点的创建和关系的建立是核心步骤之一，Java通过与Neo4j数据库的结合，利用Cypher查询语言来执行这些操作。在创建节点时，我们通常使用类似“CREATE (n:Category {category: '%s'})”这样的Cypher语句，这里的%s占位符代表待插入的属性值。例如，在导入菜品类别数据时，'%s'将被替换成实际的类别名称，从而构成完整的Cypher语句并在数据库中创建节点。与此类似，关系的建立则通过“MATCH (n:Category {category: '%s'}), (d:Setmeal {setmeal: '%s'}) CREATE (n)-[:INSETMEAL]->(d)”等语句进行，在这类语句中，MATCH用于查找已存在的节点，CREATE则用来在节点间建立关系。关系类型INSETMEAL表示类别与套餐之间的关联关系，在实际执行时，%s占位符会被相应的类别名称和套餐名称替换，以构成合法的Cypher查询语句。

在整个过程中，占位符的使用使得Java代码更加灵活和通用，能够根据不同的数据动态构建查询语句，避免了硬编码。通过这种方式，我们可以高效地将大量的业务数据导入到Neo4j图数据库中，且不必担心硬编码的查询语句需要频繁修改。虽然Neo4j会对不合法的语句抛出错误，导致程序终止，但由于数据在导入前已经经过严格的验证，确保符合数据库预设的模式，导致所有的节点和关系数据都能够成功导入数据库。

数据的成功导入不仅意味着节点和关系的创建完成，还代表着构建的知识图谱可以支持后续的智能推荐和数据分析功能。随着数据的不断增加和图谱的不断扩展，系统能够根据图谱中的节点和关系动态推断出更多的业务逻辑，提供个性化、智能化的服务，从而提升用户体验并加速业务的发展。

菜品套餐推荐知识图谱如图3-1所示。

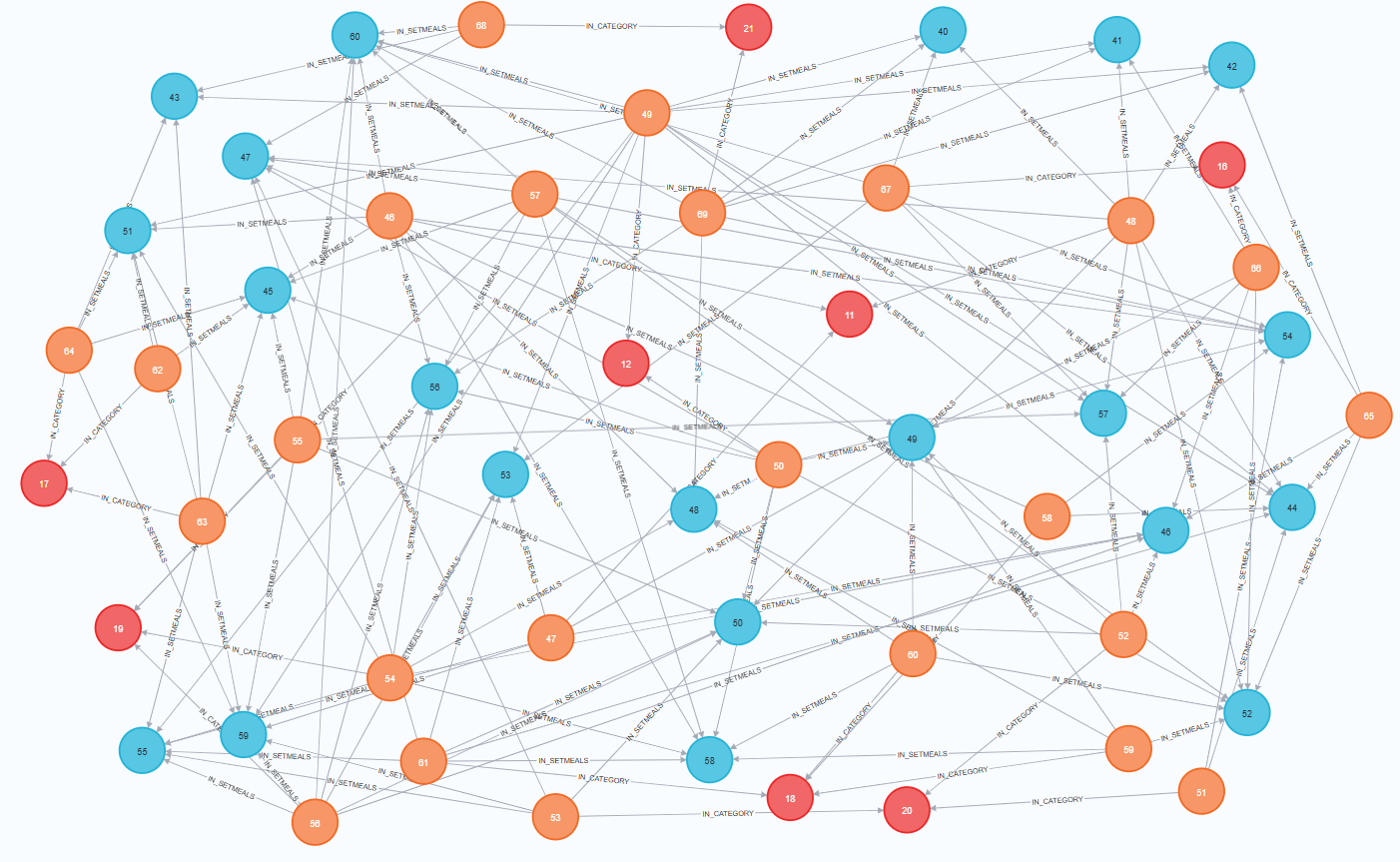


图3-7 菜品套餐推荐知识图谱

## 3.3 智能推荐方法

### 3.3.1 推荐方法的基本概念

常见的推荐方法有以下几种：

1. 协同推荐：基于用户或物品之间的相似度进行推荐，主要分为用户协同过滤和物品协同过滤。
2. 内容推荐：根据物品的属性信息（如标题，类别，描述等）与用户的兴趣偏好进行推荐。
3. 混合推荐：结合了协同过滤和内容推荐的优点，综合多种算法进行推荐。
4. 基于模型的推荐：使用机器学习算法（如矩阵分解，神经网络等）训练模型，从而进行预测和推荐。
5. 基于知识的推荐：通过整合实体和关系信息，利用图谱推理和语义分析，为用户提供更精准、个性化且可解释的推荐，解决冷启动问题并提升推荐的多样性和准确性。

本文主要采用基于知识图谱的方式进行推荐。

基于知识图谱的推荐方法利用图谱中丰富的实体（如用户、物品、标签等）和它们之间的关系信息，构建用户与物品之间的多维度关联。这种方法通过将用户的行为、偏好、历史数据以及物品的属性和上下文信息等整合到知识图谱中，借助图谱推理和语义分析，为用户提供更加精准、个性化的推荐。知识图谱通过建立实体之间的语义关系（如相似性、关联性、上下游关系等），能够捕捉到比传统推荐方法更深层次的语义信息，从而提高推荐的准确性和可解释性。例如，在菜品推荐中，知识图谱能够结合用户点餐历史、菜品的类别、套餐等信息，推断出更符合用户喜好的菜品。

与传统的基于协同过滤或内容推荐的方法相比，基于知识图谱的推荐不仅能够解决冷启动问题（即对于新用户或新物品的推荐困境），还能够通过图谱中的结构化信息更好地发掘用户潜在的兴趣。例如，知识图谱能够通过推理发现一个用户可能感兴趣的领域或物品，即使这些物品在用户历史行为中并没有直接的体现。此外，基于知识图谱的推荐方法能够提高推荐的多样性，避免推荐内容的单一性和过度集中，进一步优化用户的体验。知识图谱的可解释性也使得推荐系统能够提供更为透明和易于理解的推荐结果，增强用户的信任感和接受度。总体而言，基于知识图谱的推荐方法通过全面整合多维信息、进行语义推理，能够显著提升推荐系统的整体效果与性能。

### 3.3.2 推荐算法的实现

本文使用的关键符号，并将研究任务形式化，在这个标准的推荐系统场景中，定义了用户集和项目集，单个用户和项目分为表示为 和（沈学利等2025）。

知识图谱模块：用模块 G = <h, r, t> 表示知识图谱，描述头实体 (h) 与尾部实体 (t) 之间的语义关联，其中 r 是它们之间的交互关系，通过不同类型的实体和它们之间的关系来组织外部项目属性。知识图谱可以提供一个关于菜品、套餐、菜品类别等的详细关联信息，帮助构建菜品与套餐之间的关系结构，并进一步支持个性化推荐。知识图谱中的关系为推荐系统提供了丰富的上下文信息，能够有效提高推荐结果的准确性与多样性。

推荐模块：N 为推荐依赖的订单数目，D 为菜品（ 表示第 i 种菜品），C 为菜品种类（ 表示第 i 个菜品种类），S 为套餐（ 表示第 i 个套餐），H 为某个订单中的套餐（ 表示第 i 个订单的第 j 个套餐），K 为某个订单中的菜品（ 表示第 i 个订单的第 j 个菜品，不包含订单中的套餐中的菜品），菜品种类，菜品和套餐的推荐系数依次为：G, F, L。该模块根据用户的历史订单、菜品种类、套餐等信息，结合知识图谱，推导出用户可能感兴趣的餐品。通过深度学习算法，结合传统推荐系统和知识图谱的优势，实现更加精准的个性化推荐，提升推荐系统的解释能力。

任务制定：需要根据用户的订单历史行为，知识图谱中存储的菜品、套餐等信息，对用户将来可能要订的餐品进行推理，通过公式化设定可以整合传统推荐系统和知识图谱的优势，提升推荐的准确性和系统的解释能力。具体地，通过分析用户历史订单数据，可以得到用户的饮食偏好，这些偏好通过知识图谱得到进一步的增强，最终生成一个个性化的推荐列表。

针对推荐结果的需求建模：在整个推荐系统中，如果将用户的历史行为（用户 id 以及历史订单）直接作为推荐系统的输入，推荐结果会倾向于用户已经点过的菜品和套餐，这样会导致推荐的多样性不足。为了解决这个问题，推荐系统需要引入知识图谱所具有的过图结构增强聚合邻居节点的特征，提取用户近期历史订单里的餐品数据，通过这种方式可以很好地对用户的喜好进行分析，并根据知识图谱中的关系进一步生成多样化的推荐内容。对于新用户，系统能够利用已有图谱数据的节点和边进行有效推荐，解决项目冷启动的问题，从而快速为新用户生成有效的推荐，避免了传统方法中因为缺乏用户历史数据而无法做出准确推荐的难题。

推荐的内容主要有三个部分：需要推荐的菜品类别，菜品以及套餐。

首先是菜品类别的推荐：每个菜品类别为 ，第 j 天的套餐数量为 ，菜品数量为 ，根据以上信息，会得到每个节点的推荐系数 ，推荐的过程如下：

通过公式计算得到每个菜品类别的推荐系数，可以进一步通过每个菜品类别的推荐系数来评估菜品类别的受欢迎程度以及与用户偏好的匹配度。这些推荐系数可以根据历史数据中的订单频次、用户的消费习惯以及知识图谱中的关系等因素进行加权，从而确定菜品类别的推荐优先级。根据推荐系数，就可以得到每个节点的权重，而菜品的推荐会依据所在菜品类别的权重进行调整，使得推荐结果更加精准和符合用户需求。

接下来是菜品的推荐：每个菜品为 ，第 j 天的套餐数量为 ，菜品数量为 ，根据上述信息，就能得到每个菜品节点的推荐系数 ，推荐的过程如下：

通过类似的公式，计算得到每个菜品的推荐系数，从而评估每个菜品的推荐优先级。菜品推荐将更加依赖于所在菜品类别的权重以及该菜品在知识图谱中的相邻节点的相关性。这样，菜品推荐不仅能依赖用户的历史行为，还能通过知识图谱对菜品的相似性与关系进行建模，从而提升推荐的准确性和多样性。

最后是套餐的推荐：每个套餐为 ，第 j 天的套餐数量为 ，菜品数量为，根据上述信息，可以得到每个套餐节点的推荐系数 ，推荐的过程如下：

根据计算得到的每个套餐的推荐系数 ，可以判断哪些套餐更符合用户的口味偏好和历史行为。套餐推荐的权重则会由菜品节点的推荐系数与套餐的具体内容共同决定。通过知识图谱中的关系推理，推荐系统能够识别出哪些套餐与用户的需求相匹配，从而提供个性化且合理的套餐推荐。

这些推荐过程中的相关查询主要依赖于知识图谱中的节点之间的关系，例如某个菜品是否属于某个套餐中，这在这张关系图中很容易得到。通过知识图谱提供的强大结构化数据，推荐系统能够快速识别出与用户需求高度匹配的菜品、菜品类别和套餐，提高推荐的准确性和多样性。通过这种方式，推荐系统不仅能处理已知用户的数据，还能够在面对新用户时，依托知识图谱进行冷启动推荐，快速为用户生成有效的个性化推荐列表。

# 第4章 在线订餐智能推荐系统的设计与实现

本章从功能性和非功能性的实际系统需求分析出发，层次化设计系统功能并划分模块，以列表的方式向用户展示了知识图谱中收录的订餐菜品和套餐信息。

## 4.1 需求分析

该系统的需求分析涵盖功能性和非功能性两个维度。在功能性需求方面，采用模块化设计理念，将系统解耦为多个功能独立的模块，各模块既保持功能自治性又实现信息互通。非功能性需求则从性能指标、系统可靠性和用户体验等多元视角，为诊断系统的开发设定了明确的边界条件。系统需求分析框架如图4-1所示。

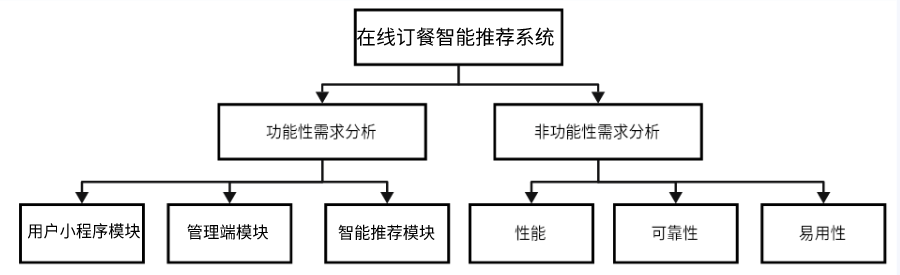


图4-1 需求分析

在功能需求分析中，将系统分为用户小程序模块、管理端模块和智能推荐模块。用户小程序可以向用户展示商家的所有菜品，套餐以及根据用户的实际情况所生成的推荐结果，用户可以在小程序下单完成在线订餐。管理端模块主要负责管理商家的菜品，套餐的添加和禁用，启用情况，以及订单状态。智能推荐模块主要负责对用户的历史行为（如历史订单）进行分析，对涉及到的菜品，套餐进行分析，并给出适合该用户的结果。

非功能性需求分析重点关注系统的性能、可靠性和易用性三个核心方面。在性能方面，系统需确保快速响应，要求页面渲染时间尽可能短；可靠性方面则强调系统运行的高稳定性，包括数据字段的准确呈现、图片资源的正常加载以及用户操作流程的无缺陷执行；易用性方面要求系统提供直观友好的交互体验，采用响应式设计适配多终端设备，并保持界面简洁美观，从而有效降低用户使用门槛并提升操作效率。

## 4.2 系统设计

### 4.2.1 系统架构设计

本文主要使用B/S三层架构模式对系统组成与结构进行设计。三层架构包含表示层、业务层和数据层，如图4-2所示。使用三层架构模式的主要优势是各层结构比较清晰，目的明确，开发时仅需关注当前开发的层级。但这种架构也有其缺点，若希望在表示层加入功能，就需要相对应的在业务层和数据层也加入相应的代码。由于本系统的需求设计已经对系统的功能需求进行了详尽的分析，因此可以避免中途加入功能的情况（李芝 2018）。

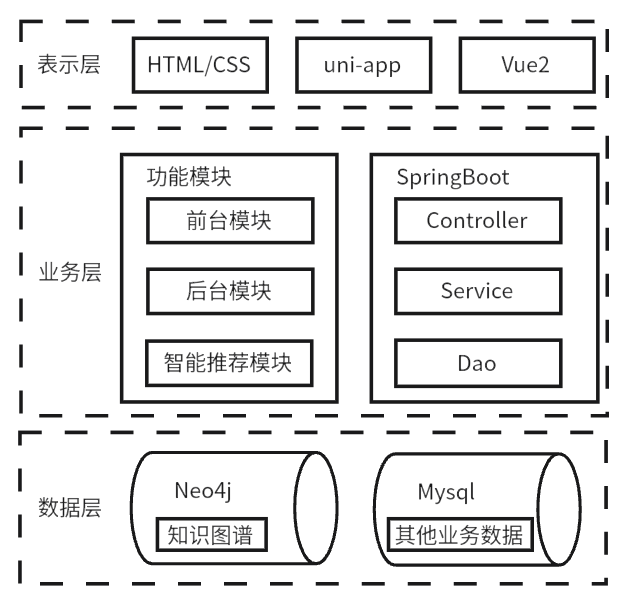


图4-2 本系统三层架构及使用的技术

针对梨果病虫害诊断系统而言，三个层次的具体作用分别如下：

（1）表示层主要负责用户和在线订餐智能推荐系统之间的交互。用户可以通过小程序端进行订餐，期间产生的数据会通过后端服务器进行处理。管理员可以通过浏览器登录管理页面，对用户产生的订单进行处理。

（2）业务层主要处理表示层产生的请求并给出响应。业务层主要对用户的一些操作进行处理（例如数据查询，订单提交等），这些功能按照需求从数据层获取数据并按照设计格式包装后对表示层做出响应。

（3）数据层主要负责的是存储在线订餐智能推荐系统中所包含的订单数据和用户信息数据。在订餐系统中，与菜品，套餐相关的数据存储在Neo4j数据库中的订餐智能推荐知识图谱，用户信息数据存储在MySQL数据库中。

### 4.2.2 用户小程序模块设计

用户小程序模块主要负责向用户展示商家的菜品，套餐信息以及支持用户的一系列订单操作。当用户进入小程序时，会提示登录信息，登录之后可以看到订餐页面。用户的登录流程如图4-3所示。用户可以选择自己喜欢的菜品，套餐，也可以根据系统推荐的内容进行选择。用户也可以查看自己的订单记录，每次订单开始之前需要选择送餐地址。订餐之后，由商家安排订单的派送。

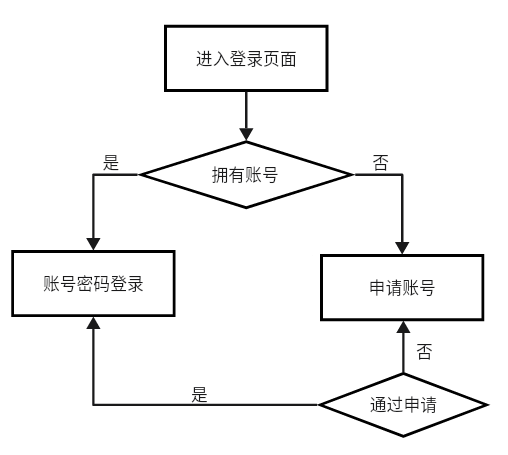


图4-3 用户登录流程

### 4.2.3 智能推荐模块

智能推荐模块是在线订餐智能推荐系统的核心模块，该模块的核心功能是将用户的历史行为进行分析并将推荐结果返回到服务端。用户的历史行为将作为智能推荐的主要依据，生成的推荐结果。智能推荐模块的功能完整流程包括：（1）后端将用户的历史订单数据传给知识图谱作为输入；（2）输入的数据是以对象的形式，数据图谱负责将每条订单数据拆分；（3）提取实体，在输入的数据中提取涉及到的菜品和套餐的实体；（4）知识图谱查询，通过分析用户的历史行为，在知识图谱中寻找相关的实体数据，并进行整合；（5）生成结果，将知识图谱中寻找的答案以菜品和套餐列表的形式在用户小程序端展示给用户（赵同明 2024）。

在推荐功能中，最核心的挑战在于利用输入的数据生成推荐的算法。知识图谱模块需要将用户的历史行为进行拆分，这个过程类似于知识问答系统。以高锐涛在水稻种植智能问答系统（高锐涛 2024）为例，它可以根据用户的提问内容给出匹配的结果。服务端用整理的用户历史行为向知识图谱模块进行提问，期望系统能以用户可以识别和使用的方式给出答案，也就是最终的推荐结果。

### 4.2.4 管理端模块设计

管理端主要是给商家使用，主要包括以下内容：

数据看板为商家提供全面的经营数据分析，实时展示订单量、营业额、订单完成率等关键指标，并支持按时间维度查看菜品和套餐的销量趋势。商家可以通过直观的图表了解热门商品和销售高峰时段，同时支持数据导出，方便财务对账和长期经营策略制定。

菜品与套餐管理模块涵盖分类管理、商品上下架及信息维护功能。商家可以自定义菜品分类（如主食、饮品、小吃等），并灵活调整菜单结构。每个菜品和套餐均可设置详细属性，包括名称、价格、描述、图片等，并支持一键启用或停售，以应对库存变化或促销需求。

订单管理负责全流程跟踪订单状态，从“待接单”“制作中”到“配送中”“已完成”各环节均可实时更新。系统会自动推送新订单提醒，确保商家及时处理，并支持手动调整异常订单（如取消或退款），保障运营流畅性。

员工账号管理用于维护后台操作权限，管理员可添加或删除员工账号，分配不同的角色（如店长、客服、后厨），并控制其访问权限（如仅查看订单或管理菜单）。员工通过账号密码登录系统，操作记录可追溯，确保管理安全合规。

整体系统围绕提升运营效率设计，覆盖从数据分析、商品管理到订单处理和人员权限的全链条需求，帮助商家高效管理外卖业务。

### 4.2.5 数据库设计

本系统主要用到两个数据库。与订餐菜品，套餐相关的功能使用Neo4j数据库中存储的在线外卖订餐智能推荐知识图谱，其他相关的功能使用MySQL关系型数据库。在线外卖订餐智能推荐图谱和推荐算法在第3章中已经明确了设计与实现，本小节主要介绍MySQL数据库中表格的设计。MySQL中的数据主要包括：订单数据，管理员数据，用户数据，地址表，以及购物车数据。

表4-1 订单表设计

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 字段描述 | 数据类型 | 长度 | 是否主键 | 约束 | 备注 |
| id | 主键 | bigint | - | 是 | not null | 唯一约束 |
| number | 订单号 | varchar | 50 | 否 | - | - |
| status | 订单状态 | int | - | 否 | not null | 1. 待付款 2. 待派送 3. 已派送 4. 已完成 5. 已取消 |
| user\_id | 下单用户 | varchar | - | 否 | not null | - |
| address\_book\_id | 地址id | varchar | - | 否 | not null | - |
| order\_time | 下单时间 | datetime | - | 否 | not null | - |
| checkout\_time | 结账时间 | datetime | - | 否 | not null | - |
| pay\_method | 支付方式 | int | - | 否 | not null | - |
| amount | 实收金额 | decimal | 10.2 | 否 | not null | - |
| remark | 备注 | varchar | 100 | 否 | - | - |
| phone | 手机号 | varchar | 255 | 否 | - | - |
| address | 收货地址 | varchar | 255 | 否 | - | - |
| user\_name | 收获姓名 | varchar | 255 | 否 | - | - |
| consignee | 昵称 | varchar | 255 | 否 | - | - |

表4-2 管理员工表设计

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 字段描述 | 数据类型 | 长度 | 是否主键 | 约束 | 备注 |
| id | 主键 | bigint | - | 是 | not null | 唯一约束 |
| name | 姓名 | varchar | 32 | 否 | not null | - |
| username | 用户名 | varchar | 32 | 否 | not null | - |
| password | 密码 | varchar | 64 | 否 | not null | - |
| phone | 手机号 | varchar | 11 | 否 | not null | - |
| sex | 性别 | varchar | 2 | 否 | not null | - |
| id\_number | 身份证 | varchar | 18 | 否 | not null | - |
| status | 状态 | int | - | 否 | not null | 1. 禁用 2. 正常 |
| create\_time | 创建时间 | datetime | - | 否 | not null | - |
| update\_time | 更新时间 | datetime | - | 否 | not null | - |
| create\_user | 创建人 | bigint | - | 否 | not null | - |
| update\_user | 修改人 | bigint | - | 否 | not null | - |

表4-3 用户表设计

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 字段描述 | 数据类型 | 长度 | 是否主键 | 约束 | 备注 |
| id | 主键 | bigint | - | 是 | not null | 唯一约束 |
| name | 名字 | varchar | 50 | 否 | - | - |
| phone | 手机号 | varchar | 100 | 否 | not null | - |
| sex | 性别 | varchar | 2 | 否 | - | - |
| id\_number | 身份证号 | varchar | 18 | 否 | - | - |
| avatar | 头像 | varchar | 500 | 否 | - | - |
| status | 状态 | Int | - | 否 | - | 1. 禁用 2. 正常 |

表4-4 地址表设计

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 字段描述 | 数据类型 | 长度 | 是否主键 | 约束 | 备注 |
| id | 主键 | bigint | - | 是 | not null | 唯一约束 |
| user\_id | 用户id | bigint | - | 否 | not null | - |
| consignee | 收货人 | varchar | 50 | 否 | not null | - |
| sex | 性别 | tinyint | - | 否 | - | 1. 女 2. 男 |
| phone | 手机号 | varchar | 11 | 否 | - | - |
| province\_code | 省级区划编号 | varchar | 12 | 否 | - | - |
| province\_name | 省级名称 | varchar | 32 | 否 | - | - |
| city\_code | 市级区划编号 | varchar | 12 | 否 | - | - |
| city\_name | 市级名称 | varchar | 32 | 否 | - | - |
| detail | 详细地址 | varchar | 200 | 否 | - | - |
| label | 标签 | varchar | 100 | 否 | - | - |
| is\_default | 是否默认 | tinyint | - | 否 | not null | 1. 否 2. 是 |
| create\_time | 创建时间 | datetime | - | 否 | not null | - |
| update\_time | 修改时间 | datetime | - | 否 | not null | - |
| create\_user | 创建人 | bigint | - | 否 | not null | - |
| update\_user | 修改人 | bigint | - | 否 | not null | - |

表4-5 购物车数据表设计

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名 | 字段描述 | 数据类型 | 长度 | 是否主键 | 约束 | 备注 |
| id | 主键 | bigint | - | 是 | not null | 唯一约束 |
| name | 名字 | varchar | 50 | 否 | - | - |
| image | 图片 | varchar | 100 | 否 | - | - |
| user\_id | 用户id | bigint | - | 否 | not null | - |
| dish\_id | 菜品id | bigint | - | 否 | - | - |
| setmeal\_id | 套餐id | bigint | - | 否 | - | - |
| dish\_flavor | 口味 | varchar | - | 否 | - | - |
| number | 数量 | int | - | 否 | not null | - |
| amount | 金额 | Decimal | 10.2 | 否 | not null | - |
| create\_time | 创建时间 | datetime | - | 否 | not null | - |

## 4.3 系统实现

### 4.3.1 表示层实现

在技术实现上，该外卖订餐管理端的前端表示层采用 Vue 2 作为核心框架，并结合 uni-app 进行跨平台开发，以构建高效、一致的用户交互界面。

UI 组件库 选用了 Element-UI，这是一个基于 Vue 2 的成熟开源组件库，提供了丰富的预制组件（如表格、表单、弹窗等），能够显著提升开发效率。Element-UI 的组件不仅视觉风格统一、美观，而且具备良好的可定制性，相比原生 HTML 组件，其交互体验和开发便捷性更优。开发者可通过 npm 安装该库，并在项目的 package.json 中配置依赖，即可在 Vue 组件中直接调用。

数据请求层 使用 Axios 作为 HTTP 客户端，它是一个基于 Promise 的异步请求库，支持浏览器端发送 XMLHttpRequest，并能够对后端返回的数据进行标准化处理。Axios 的拦截器机制便于统一管理请求（如添加 Token 鉴权）和响应（如错误处理），确保前后端数据交互的可靠性和可维护性。

这一技术方案兼顾了开发效率与用户体验，通过 Vue 2 的数据驱动视图特性、Element-UI 的标准化组件以及 Axios 的稳定网络请求，为管理端提供了流畅、易维护的前端架构。

用户端模块主要包括菜品套餐的展示，推荐功能以及个人信息展示。表示层的图文信息页面包含系统logo、左部导航栏、菜品信息，套餐信息，用户地址信息，历史订单信息。这样的布局参考了传统意义的系统布局。菜品和套餐信息主要使用列表的形式，图片下方展示图片对应的名称，效果如图4-4所示。点击图片可以看到菜品的详细信息，详细信息以弹窗的形式展示，弹窗展示如图4-5所示。

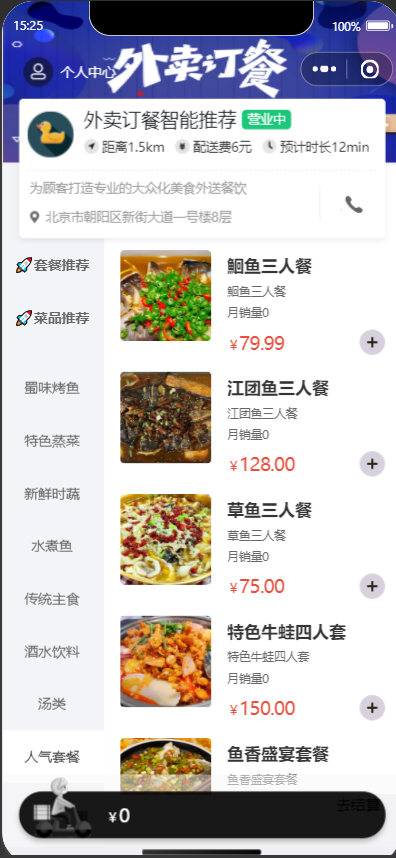


图4-4 菜品列表



图4-5 菜品详细信息

用户端模块中包含的操作有查看菜单、上下滑动菜品列表、点击列表中的其中一项产生详细信息弹窗，以及关闭详细信息弹窗等。其中点击菜品产生详细信息弹窗这一操作与业务层和数据层存在信息的交互，其他操作均只涉及表示层实现。菜品详细信息弹窗使用Element-ui库的弹窗组件实现，点击图片时向业务层发送数据请求，业务层返回对应数据请求结果，表示层响应请求并以预定义格式渲染数据。用户可以点击个人中心查看历史订单和地址信息，切换至个人中心的操作通过部署VUE2的路由实现，切换个人中心实际上是切换系统路由至系统的个人中心。历史订单列表通过VUE2的列表组件实现，列表设置屏幕占用比例，采用了图片动态懒加载设计，增强了易用性。

个人中心模块主要展示了用户近期的历史订单和地址详情入口。用户可以查看不同状态的历史订单并重复下单，并设置一些送餐的地址。个人中心表示层效果如图4-6所示。



图4-6 个人中心

管理模块包含登录页和管理员视图下的用户管理功能，以及订单数据的查看，菜品，套餐数据的处理。登录页分为员工账号添加和登录两个部分。登录页如图4-7所示，管理员添加员工账号页面如4-8所示。员工管理功能在管理员视图下可见。由管理员在平台创建账号供商家员工使用。



图4-7 登录页面

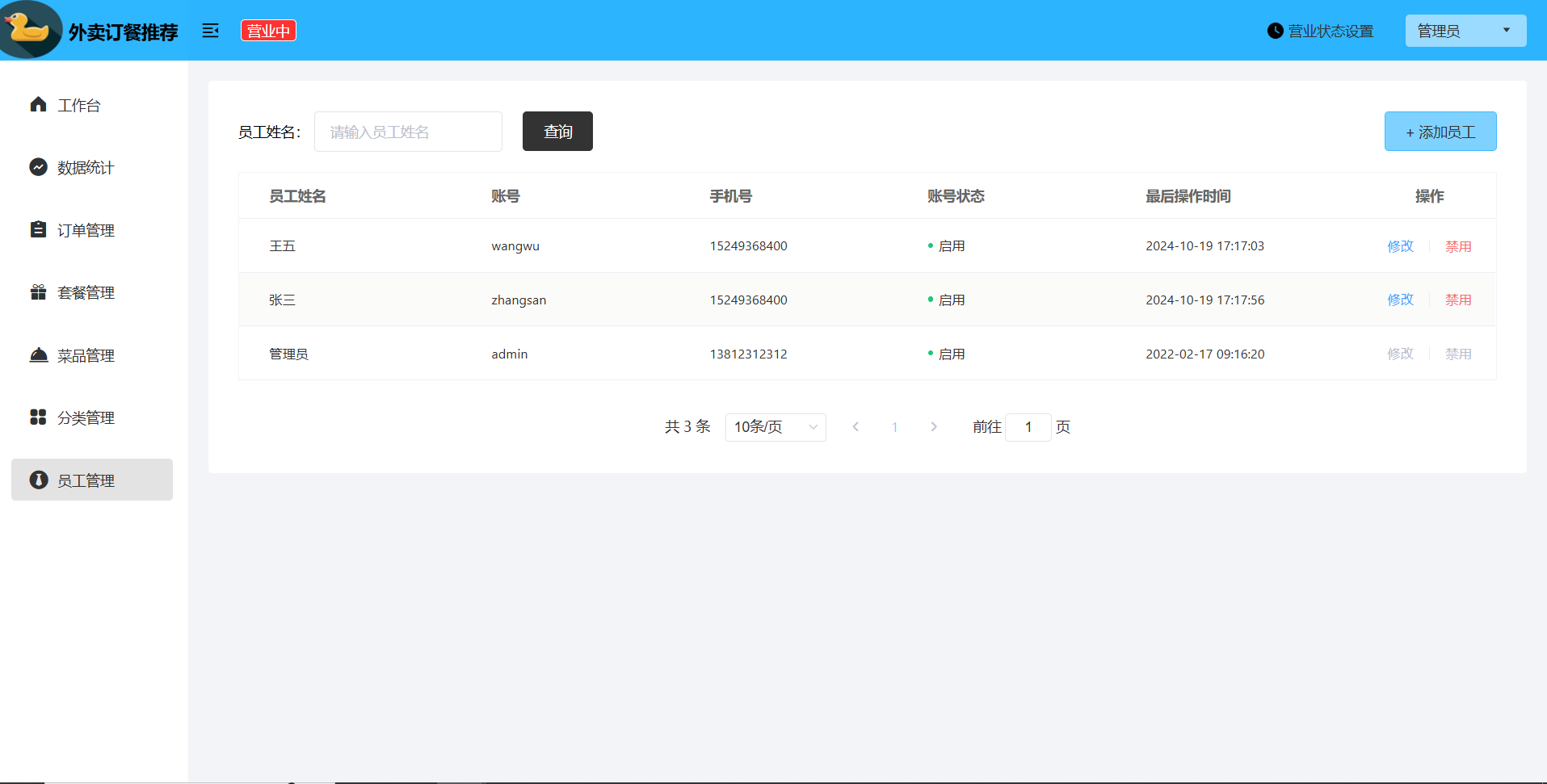


图4-8 管理员添加账号页面

管理员模块的主要功能使提供订单数据的看板，统计订单情况，各个菜品的营销情况以及各种状态订单的处理，菜品管理页面和数据看板页面如图4-9和图4-10所示。



图4-9 菜品管理页面

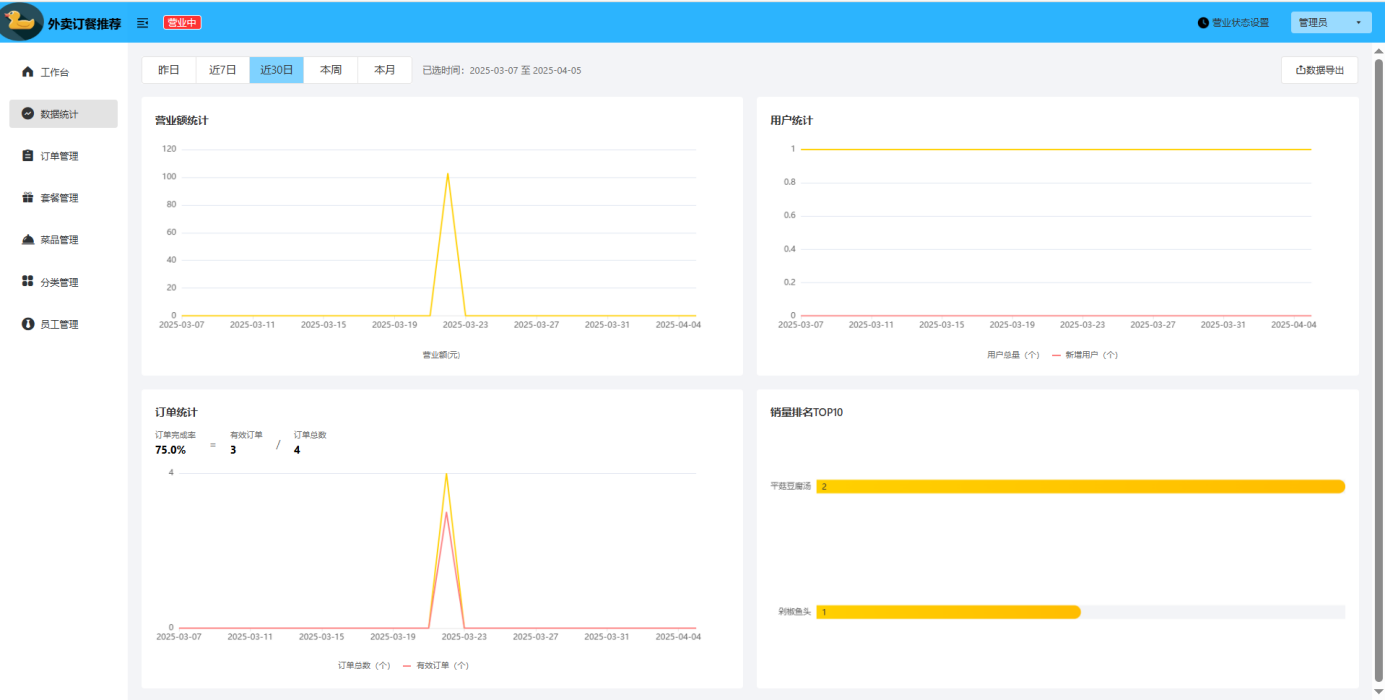


图4-10 数据看板页面

表示层发送至业务层的请求包括：（1）管理端模块中订单，菜品，套餐相关的全部信息；（2）管理端的登录信息；（3）用户端模块的菜品，推荐和订单信息。部分请求接口如图4-11所示。



图4-11 数据看板页面

### 4.3.2 业务层实现

业务层使用SpringBoot框架实现业务逻辑。SpringBoot将整个业务层划分为三个部分，分别是Controller层、Service层和Dao层（罗光武 2024）。多层次将表示层请求的任务分解，使业务层代码更加解耦合。

1. Controller层负责接收表示层发出的请求，并将请求下发给请求处理方法。在Controller层中，构建请求接收类DTO，请求接受类中包含请求接收方法。请求接收方法调用Service层的请求处理类中对应的请求处理方法，将请求中包含的参数传递至请求处理方法，并将请求处理方法返回的结果包装为浏览器的响应返还至表示层。 （2）Service层负责处理Controller层分配的请求。在Service层中，构建请求处理类VO，请求处理类中请求处理方法。接口响应的数据类型如图4-11所示。



图4-11 接口响应的数据类型

（3）Dao层包含处理数据库的全部操作。对Service层提供一些可供调用的接口。按照上层的需求，Dao层包含查询实体相联系的全部实体、查询实体相联系的某个实体、按ID增加用户信息、按ID修改用户信息、按ID删除用户信息和按ID查询用户信息等方法，这些数据库操作方法封装在Dao层以确保查询结果的完整性和安全性。

### 4.3.5 数据层实现

本系统的数据层由在线订餐智能推荐知识图谱和MySQL数据库两部分组成，其中知识图谱部分已在前文详述，本节重点介绍MySQL数据库的用户表实现。

根据数据库设计规范，我们使用数据定义语言(DDL)创建了用户表，该表包含用户ID、用户名等核心字段，并设置了相应的约束条件以确保数据完整性。考虑到用户表将存储大量用户信息，我们特别在用户ID字段上建立了索引，这一优化措施能显著提高查询效率，降低系统响应时间，从而有效维护整体性能。表结构设计充分考虑了实际业务需求和数据操作特点，在保证功能完整性的同时优化了数据库访问性能。

# 第5章 总结与展望

## 5.1 总结

我国外卖餐饮订餐方式经历了从电话订餐到互联网订餐的演进过程，目前行业体系已日趋成熟。通过市场调研发现，传统餐饮行业与互联网技术的融合程度仍有提升空间，为此我们提出了区域性餐饮网络营销的创新方案。该方案以城区中小型餐饮企业为核心服务对象，充分结合周边住宅区和校园区的地理特征，构建开放型餐饮服务平台。

在技术实现层面，系统采用Redis作为高性能缓存数据库，其基于内存操作的单线程架构避免了多线程切换开销，配合底层Hash算法和C语言编写优势，展现出卓越的读写性能。我们充分利用这些特性，将商品数据及分类信息存储在Redis中，显著提升了系统响应速度，优化了用户体验。

本系统的核心创新点在于智能推荐功能，通过分析用户历史行为数据生成个性化推荐结果，有效降低用户搜索成本，提升订餐效率。经过完整的开发测试周期，平台各项功能均得到充分验证和持续优化，现已具备稳定的运行表现。系统设计兼顾多类型用户需求，支持商家自主管理菜品信息，并具备云端部署能力，展现出良好的适用性和扩展性。

最终实现的订餐系统包含完整的前后台功能：前端支持用户浏览菜品、购物车管理、个人信息维护及在线支付等全流程操作；后台为管理员提供菜品管理、订单处理等运营功能，共同构成了一个高效便捷的在线订餐解决方案。

## 5.2 展望

本系统相对于其他各大外卖订餐系统，仍需在以下三个方面进行完善提升：

在系统功能方面，当前流程设计较为简单，缺乏创新性的营销功能模块。我们计划在后续版本中陆续推出在线秒杀抢购、节日专属折扣、多人拼团等特色营销活动，以增强平台的吸引力和竞争力。

在系统安全方面，目前可能存在未知的安全隐患。考虑到一旦遭受黑客攻击可能导致用户隐私泄露或系统瘫痪等严重后果，我们将采取以下改进措施：开展全面的安全审计工作，实施严格的数据加密方案，建立规范的日志记录机制，并定期进行漏洞扫描和系统更新。

在并发处理方面，当用户访问量激增时系统可能出现性能问题。为此我们将从多个层面进行优化：优化核心算法和代码结构，提升数据库查询效率，并引入分布式技术架构，确保系统能够稳定应对高峰时段的访问压力。

经过完整的开发和测试流程，平台各项功能已得到充分验证和持续优化，运行稳定性显著提升。该系统适用于各类用户群体和餐饮商家，支持商家自主管理菜品信息，同时具备云端部署能力，展现出良好的实用性和扩展性。

参考文献

谭淞,向欢,张瑞娜,等.烟叶烘烤工艺知识图谱构建与应用研究[J].现代农业科技,2024(06):166-172.

赵赛,杨婉霞,王巧珍等.基于马铃薯病虫害知识图谱的问答系统[J].农业工程,2023,13(08):29-37.DOI:10.19998/j.cnki.2095-1795.2023.08.006.

刘迪,汤胡欣,周亚男等.基于知识图谱的杀虫剂研究现状可视化分析[J].生物灾害科学,2023,46(03):306-313.

余宏杰,高星.水稻病害防控模型与知识图谱构建研究[J].山东农业工程学院学报,2023,40(08):29-34.DOI:10.15948/j.cnki.37-1500/s.2023.08.016. 10

胡韬.软件架构的演进与发展趋势分析[J].集成电路应用,2024,41(02):72-73.DOI:10.19339/j.issn.1674-2583.2024.02.025.

邹洪.基于人工智能的自然语言处理技术在软件测试中的应用研究[J].网络安全和信息化,2024(04):59-61.

徐增林,盛泳潘,贺丽荣,等.知识图谱技术综述[J].电子科技大学学报,2016,45(04):589-606.

薛慧君.基于JSP的电子商务网站设计及应用[J].电脑知识与技术,2016,12(34):287-289.DOI:10.14004/j.cnki.ckt.2016.4779.

李佳欣. 基于知识图谱的苹果病虫害信息推荐技术研究[D].西北农林科技大学,2023.DOI:10.27409/d.cnki.gxbnu.2023.002691.

苏芝,李茹.基于HTML5的船舶智能监控系统软件界面自适应实现[J].上海船舶运输科学研究所学报,2018,41(04):34-38.

罗光武,陈典灿,吴荷,等.应用Springboot+Vue框架的时间管理软件的设计与实现[J].工业控制计算机,2024,37(04):64-66.

赵同明,钱佳琛,王翔,等.健康饮食知识图谱和问答系统构建研究[J].现代信息科技,2024,8(06):7-10.DOI:10.19850/j.cnki.2096-4706.2024.06.002.

曾德晶,张军,管党根,等.长江流域取水许可知识图谱问答系统[J/OL].人民长江:1-8[2024-05-04].http://kns.cnki.net/kcms/detail/42.1202.TV.20240312.1556.002.html.

高锐涛,林达伟,郭亮,金鸿,王红.基于知识图谱的水稻种植智能问答系统设计与实现[J].计算机工程:1-15.

陈韶健. 2017. Neo4j全栈开发. 北京: 电子工业出版社: 17.

沈学利,王嘉慧.基于知识图谱的用户兴趣推荐[J/OL].计算机系统应用,1-11[2025-03-09].https://doi.org/10.15888/j.cnki.csa.009850.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. DBpedia– A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. Semantic Web Journal, 6(2), 2013.

Lao, N. and William, W. Relational retrieval using a combination of path-constrained random walks. Machine Learning, 81(1):53–67, 2010.

Bordes, A. et al. Translating embeddings for modeling multi-relational data. In: Proc. of the Advances in Neural Information Processing Systems. Red Hook .pp. 2787-2795. 2013.

Tom Heath,Talis Christian Bizer. Linked Data:Evolving the Web into a Global Data Space.Molecular Ecology,11(2):670-684. 2011.

Douglas B Lenat. CYC: A large-scale investment in knowledge infrastructure. Communications of the ACM, 38(11):33–38, 1995.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: A Collaboratively Created Graph Database For Structuring Human Knowledge. In Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pages 1247–1250, New York, 2008.ACM.

Denny Vrandeci ˇ c and Markus Krötzsch. Wikidata: a Free Collaborative Knowledge Base. Communications of the ACM, 57(10):78–85, 2014.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. YAGO: A Core of Semantic Knowledge Unifying WordNet and Wikipedia. In 16th international conference on World Wide Web, pages 697–706, NewYork, 2007.ACM.

致 谢

在完成本篇论文的过程中，我想要向许多人表达我的真诚感激之情。

首先，我要感谢我的指导老师。在整个研究过程中，老师给予了我悉心的指导和无私的支持。老师的专业知识和丰富经验为我的研究提供了宝贵的指导和建议，使我能够不断完善论文的内容和结构。老师的耐心和鼓励让我始终充满信心，克服困难，不断前行。在此，我要向老师表示最诚挚的谢意。

其次，我要感谢我的舍友们。他们与我分享了彼此的思想和经验，在讨论和交流中，我获得了许多启发和帮助。他们的支持和鼓励使我在论文的写作过程中不再感到孤单，每一次交流都是一次宝贵的学习经历。没有他们的帮助和支持，我无法完成这篇论文。

最后，我要感谢我的猫。在我写论文的漫长过程中，它始终陪伴在我身边，给我带来无尽的快乐和安慰。

总之，我要感谢所有在我研究生涯中给予我帮助和支持的人，正是因为有了你们的支持，我才能够顺利完成这篇论文。再次向你们致以最诚挚的谢意！