梧 州 学 院

毕 业 论 文

论文题目 基于SpringBoot和Vue框架的自动

售货机管理系统的设计与实现

学 院 大数据与软件工程学院

专 业 软件工程

班 级 \*\*软件工程\*班

学 号

学生姓名

指导老师（签名）

完成时间 年 月

梧州学院

学士学位论文原创性声明

本人郑重声明：所呈交的论文是本人在导师的指导下独立进行研究所取得的研究成果。除了文中特别加以标注引用的内容之外，本论文不包含任何其他个人或集体已经发表或撰写的成果作品。对本文的研究所作出重要贡献的个人和集体，均已在文中以明确方式表明。本人完全意识到本声明的后果由本人承担。

论文作者签名：

日期：

摘要

随着科学技术和人们生活质量的提高，互联网行业的高速发展，自动售货机作为一种方便快捷的新零售自动销售服务模式，在大众日常生活圈中，成为一种不可替代的产品。可是部分使用自动售货机售卖商品的企业对自动售货机的管理却跟不上时代的潮流，依旧采用传统的基础管理系统或是人工方式进行管理，此管理模式易导致数据混淆的现象，且大部分用户需要进行购物时无法获取售货机的具体位置，也不能进行线上购物，用户互动性差，使一部分用户群体流失，这显然已不能满足企业的发展需求。因此，针对上述问题开发一套功能齐全、操作方便、界面整洁，用户能获取售货机具体位置，能线上预定商品、线上购物、互动性好的自动售货机管理系统显得尤为重要。

本文以分析自动售货机的管理现状与问题为研究背景，完善现代化售货机管理需求，采用开源轻量级框架Spring Boot和Vue实现自动售货机系统的开发，基于uni-app框架开发移动端微信小程序。从系统目标、业务需求以及功能实现等方面，对自动售货机管理系统的设计和实施进行详细的论述。本系统包含的主要模块有售货机运营管理、商品发布、广告发布、查找设备、商品预定、线上购物、商家入驻等，实现了自动售货机管理系统的功能。

开发本系统是为了提高售货机管理的效率、规范管理售货机运行产生的数据，提高用户发现售货机和使用售货机的频率，提升企业的运营效率，进而带动众多领域相关产业发展，为经济发展添加一份力。

关键词：自动售货机管理系统； 前后端分离； uni-app

**Design and Implementation of Vending Machine Management System Based on Spring Boot and Vue Framework**

Abstract

With the improvement of science and technology and people's quality of life, the rapid development of the Internet industry, vending machine as a convenient and fast new retail automatic sales service mode, in the mass daily life circle, has become an irreplaceable product. However, some enterprises that use vending machines to sell goods cannot keep up with the trend of The Times. They still use traditional basic management system or manual management. This management mode is easy to lead to data confusion, and most users cannot obtain the specific location of the vending machine when they need to shop, nor can they shop online. The loss of part of the user group, which obviously can not meet the needs of the development of enterprises. Therefore, in view of the above problems to develop a complete set of functions, convenient operation, clean interface, users can obtain the specific location of the vending machine, online booking goods, online shopping, good interactive vending machine management system is particularly important.

Based on the analysis of the current situation and problems of vending machine management as the research background, this paper improves the requirements of modern vending machine management, adopts open source lightweight frameworks Spring Boot and Vue to realize the development of vending machine system, and develops wechat mini program on mobile terminal based on uni-app framework. The design and implementation of vending machine management system are discussed in detail from the aspects of system objective, business requirement and function realization. This system includes the main modules of cargo machine operation management, commodity release, advertising release, search equipment, commodity reservation, online shopping, merchants settled in, and so on, to achieve the function of the vending machine management system.

The development of this system is to improve the efficiency of vending machine management, standardize the management of the data generated by the operation of vending machine, improve the frequency of user discovery and use of vending machine, improve the operating efficiency of enterprises, and then drive the development of related industries in many fields, add a force for economic development.

**Key words:** vending machine management system； Anterior and posterior end separation； uni-app

目录

[第一章 绪论 1](#_Toc10612)

[1.1 项目的背景和意义 1](#_Toc3853)

[1.2 国内外研究现状 2](#_Toc21801)

[1.3 论文的主要工作 2](#_Toc29263)

[1.4 论文的结构安排 3](#_Toc4062)

[1.5 本章小结 3](#_Toc19931)

[第二章 相关理论和技术 4](#_Toc11987)

[2.1 Spring Boot框架 4](#_Toc24142)

[2.2 MyBatis-Plus框架 4](#_Toc12070)

[2.3 Redis 4](#_Toc23415)

[2.4 Vue框架 5](#_Toc998)

[2.5 uni-app 5](#_Toc9378)

[2.6 本章小结 6](#_Toc19909)

[第三章 系统需求分析 7](#_Toc25601)

[3.1 可行性分析 7](#_Toc4431)

[3.2 用户需求分析 8](#_Toc2702)

[3.3 系统性能需求 15](#_Toc26570)

[3.4 非功能性需求 15](#_Toc20448)

[3.5 本章小结 16](#_Toc18478)

[第四章 系统总体设计 17](#_Toc32200)

[4.1业务平台架构设计 17](#_Toc12506)

[4.2部署架构设计 18](#_Toc19890)

[4.3系统总体模块设计 19](#_Toc32751)

[4.4数据库设计 21](#_Toc3607)

[4.5 本章小结 30](#_Toc30581)

[第五章 系统详细设计与实现 31](#_Toc14473)

[5.1主体架构细化以及实现 31](#_Toc5046)

[5.2设备部署模块 32](#_Toc9504)

[5.3商品预定模块 37](#_Toc19834)

[5.4设备搜索模块 42](#_Toc11860)

[5.5 本章小结 45](#_Toc28417)

[第六章 系统测试和验证 46](#_Toc28797)

[6.1 系统测试概述 46](#_Toc6638)

[6.2 系统功能测试 46](#_Toc27291)

[6.3 系统性能测试 51](#_Toc19849)

[6.4 本章小结 52](#_Toc12816)

[第七章 总结与展望 53](#_Toc30713)

[7.1 总结 53](#_Toc14704)

[7.2 展望 53](#_Toc12905)

[参考文献 54](#_Toc27287)

[致谢 55](#_Toc1841)

# 第一章 绪论

## 项目的背景和意义

### 1.1.1背景

随着科学技术和人们生活质量的提高，互联网行业的高速发展，自动售货机作为一种方便快捷的新零售自动销售服务模式，在大众日常生活圈中，成为一种不可替代的产品。自动售货机在20世纪70年代于日本和欧美等国家逐渐发展起来，后来陆续推广到其他国家。对比传统的店铺售卖方式来说，自动售货机可算是一种新型的商业零售模式，自动售货机具有无需人工盯守、占地面积小、省成本、省人工、购物方便，能够灵活部署等优点，是一家全天候二十四小时不间断经营的小型超市，使得消费者对购物时间点、便捷性上得到了极大的帮助。同时，它可以美化公共场所和城市环境，提高社会购物功能，也提高商品销售水平，具有巨大的社会效益和经济效益。

随着万物互联的越发成熟，人们能更加方便快捷地获取设备、物品之间的信息，从而使得社会上的电器等设备逐渐智能化、网络化。自动售货机也不例外，设备联网后能够进行远程管理，降低了企业、运营商对自动售货机的运营成本。从而受到了越来越多的国人欢迎和认可，也使得越来越多的人们喜欢使用自动售货机来进行购买、消费。如今在大街小巷、学校、医院、高铁站、写字楼、酒店等场所都能看到自动售货机的身影，由此可见自动售货机在国内已经形成规模，并且面对国内的庞大消费群体，自动售货机市场将有无限可能的发展空间。

面对如此庞大的市场，自动售货机过于分散、容易断货和易出现故障的缺点也给自动售货机经营者带来了业务数据分析难、较高的人工管理成本等问题。所以自动售货机企业、运营商如何能够更加方便快捷地管理设备成为一个棘手的问题。实现更低成本、更高效率地管理和运营自动售货机的目标，是自动售货机管理者要解决的重点问题。因此，设计、研发一套能够方便消费者使用和管理者实时远程监控的自动售货机管理系统是很有必要的，不仅整体提升了自动售货机管理的效率，而且还能不断优化自动售货机的管理工作。

### 1.1.2意义

本论文选题的意义是，结合自动售货机管理系统和移动端微信小程序，实现一个能对自动售货机进行高效率管理且稳定、安全、实用的系统，用低成本，高收益来满足经营者对自动售货机远程监控和管理的需求。基于SpringBoot和Vue框架的自动售货机管理系统，可以辅助运营商实时管理自动售货机运转状态和商品销售情况等信息，能够在自动售货机设备上投放广告，支持商家入驻系统、设置销售策略、统计数据、微信小程序管理等功能，满足企业、运营商对自动售货机进行系统性管理的实际功能需求。同时满足消费者通过移动端微信小程序，以最直观的方式获取运营范围内所有自动售货机的地理位置信息，消费者也能进行线上购物、获取系统相应的优惠活动等功能，极大方便了消费者的购物需求和使用体验。

## 1.2 国内外研究现状

### 1.2.1 国外研究现状

资本主义国家的工业革命后，西方国家的经济一直处于领先水平，近代以来，我国都在持续追赶，自动售货机市场也同样受到了这种经济的折射，相比之下，国外自动售货机普及程度比较高，以美国和日本为例，早在2016年美国就拥有690万台自动售货机、14225家运营商，达到了251亿美元年收入[1]。在日本，自动售货机渠道销售了全国 70% 的罐装饮料[2]。

由此可见，自动售货机在国外的发展情况非常可观，从人均拥有量来比较，美国35人内占一台，日本23人内占一台，成为世界人均占有自动售货机比例名列前茅的国家，欧洲国家的自动售货机也相当普及，根据欧洲自动售货机行业协会的调查数据表明，较为发达的国家安装、运营的自动售货机数量不低于50万台。

### 1.2.2 国内研究现状

由于我国近代的经济发展速度相对于国外较慢，所以自动售货机引进国内的时间也比较晚，我国从1994年引进第一台自动售货机，截至2019年疫情前，我国的自动售货机数量约35万台，人均2200人/台，饮品类占据市场主要份额53%，自动服务机占28%，票 券类占8%[3]，远低于日本、美国等发达国家。随着互联网和万物互联的普及，使得自动售货机在人机交互上得到了重大的提升，使得运营商管理自动售货机变得更加方便，据中国产业调研网发布的《中国自动售货机市场深度调查研究与发展趋势分析报告》显示，灵活、方便、快捷以及自助式服务成为消费的新趋势，这是自动售货机得以发展迅速的主要原因[4]。为此，从我国自动售货机人均拥有量及自动售货机趋势进行分析，由于自动售货机在国内的发展空缺，这对整个自动售货机行业而言，将会有重大发展机遇，发展前景相当广阔。

## 1.3 论文的主要工作

通过研究整体环境对自动售货机市场发展的背景意义，以及在国内和国外的研究现状，分析出当前社会对整个售货机行业存在的问题及发现潜在的新策略，根据系统进行需求的获取和分析，最终设计出符合预期的基于SpringBoot和Vue框架的自动售货机管理系统。在开发目标系统过程中，主要研究的内容有：

1.了解国内商业市场对自动售货机行业的意义。搜索和查阅与课题相关的资料，对课题进行综合研究，完善课题研究计划，拟写论文大纲和软件开发的需求。

2.对本系统的搭建。基于SpringBoot框架、Vue框架和移动端微信小程序等架构技术进行组装，参考国内外现有的自动售货机系统，对SpringBoot、Vue、Redis、uni-app框架等技术和热门框架进行学习掌握。根据实际对技术的了解掌握程度确定系统开发架构。

3.系统需求理论研究。根据国内外自动售货机管理系统的使用体验和实际需求分析，在结合实际市场需求和存在的缺陷的基础上，完成自动售货机系统的软件需求。

4.架构设计和理论研究。在系统研究和理论分析的基础上，可以对系统进行分模块详细设计并完成系统架构设计。

5.针对具体的实现技术及模块进行详细的设计。结合业务需求和功能模块分析，利用UML软件对该系统进行建模与设计。

6.对系统各个功能模块进行数据库设计，以及编码的实现，完成系统测试，依据测试结果，对系统的功能进行修正与改进，增强系统实用性与稳定性。

## 1.4 论文的结构安排

第一章 绪论。根据自动售货机系统的研究背景和意义，阐述了该课题的国内外研究现状，从而讲述设计与实现基于Spring Boot和Vue框架的自动售货机管理系统的必要性。

第二章 介绍相关理论和技术。本章将逐一介绍所选择的系统架构技术，例如Spring Boot、MyBatis-Plus、Redis、Vue、uni-app等系统使用的技术，解释相关理论和技术的选择依据。

第三章 系统需求分析。详细分析该系统的可行性，然后对该系统的用户需求、功能需求及非功能性需求进行分析，改进并明确系统的需求。

第四章 系统总体设计。本章从整体上对基于Spring Boot和Vue框架的自动售货机管理系统进行设计，包括架构设计、功能模块结构划分、数据库设计等，确定系统的总体设计思路和实现架构。

第五章 系统详细设计与实现。通过这一章，阐述自动售货机管理系统中各大功能的具体设计流程，并借助时序图，类图、流程图和其它软件建模技术，用更直观的形式展示系统的设计思路及详细流程。

第六章 系统测试及验证。首先介绍系统测试的软件环境，其次，对自动售货机的主要系统功能和性能进行测试，最后，将预期结果与测试结果进行对比，得出测试结论。

第七章 总结与展望。本章主要总结本课题的研究和设计成果，并在此基础上提出本系统后续的设计、研究方向。

## 1.5 本章小结

在本章中，首先对论文研究涉及的社会状况进行阐述，对自动售货机的发展背景、研究意义进行了说明。随后通过具体的数据分析了我国自动售货机还处于普及阶段，介绍了自动售货机当前环境下存在的一些实际性的问题，从而突出了设计与实现自动售货机管理系统的必要性。通过对自动售货机国内发展状况的调查研究，剖析自动售货机今后在全国范围内大面积普及的发展前景，由此确定该项目系统的大致需求及研究内容，并归纳本文的主要工作和结构安排。

# 第二章 相关理论和技术

本章将对基于Spring Boot和Vue框架的自动售货机管理系统运用到的相关理论和技术进行详细描述。基于Spring Boot和Vue框架的自动售货机管理系统，前后端均采用在web管理系统领域下深受开发者热爱的框架。系统后端使用Java语言进行开发，并基于MVC模式，采用Spring Boot框架进行开发，使用MyBatis-Plus做数据持久层框架，采用MySQL数据库做数据存储，使用Alibaba Druid 做数据库连接池，用Redis做持久化缓存。前端采用的是现如今最火热的Vue框架，前端UI界面使用iViewUI组件库，移动端微信applet使用uni-app开发，移动端控件使用基础UI库uViewui，地图模块使用地图引擎amap。Web端作为自动售货机管理系统，移动端微信小程序主要由消费者进行使用。

## 2.1 Spring Boot框架

Spring Boot是在Spring4.0的基础上设计而成，它既继承Spring框架最初的优良特性，又通过简化配置，使Spring应用构建与开发的全过程得到进一步的简化。另外Spring Boot通过集成大量的框架使得依赖包的版本冲突，以及引用的不稳定性等问题得到了很好的解决[5]。Spring Boot核心理论主要分为三个核心：1、可以帮助开发者迅速整合Spring，MyBatis，第三方框架；2、基本上去掉了xml的配置并以注解的方式出现；3、不需要外部容器，内嵌服务器（Tomcat）容器简化Web项目。帮助所有使用Spring框架的开发者们更快的入门。

该系统使用Spring Boot框架是因为省略了Spring框架中烦杂而庞大的配置文件以及开发团队觉得原始Spring程序在最初构建时可能会有一些麻烦，因此，采用Spring Boot框架让文件配置更加简单、高效，让开发者把更多时间投入到系统的业务逻辑处理中，从而达到快速开发的目的。

## 2.2 MyBatis-Plus框架

MyBatis是一款高级的数据持久层框架，它可以自动化装配JDBC操作，例如手动设置驱动程序、建立连接、获取执行对象和释放连接，可以使用配置或注解形式进行开发。plus是“外加”的意思，顾名思义MyBatis-Plus是MyBatis的加强版，在不改变MyBatis的情况下进行功能增强，简化开发，提高开发效率。

本系统采用MyBatis-Plus框架，原因是基本上去掉了MyBatis大量的xml映射文件，数据层继承MyBatis-Plus框架内的基础类就可以实现简单的增删改查操作，复杂的sql语句也可以通过注解的形式开发，让开发者更加方便高校地进行开发。

## 2.3 Redis

Redis的全称是Remote Dictionary Server(远程数据服务)。软件是用C语言编写的。Redis是一个键值存储系统，支持丰富的数据类型。因其在高并发场景下极高性能和易于配置、部署的优势，获得学术界和产业界广泛认可[6]，由于关系型数据库采用磁盘作为存储介质，在数据读写方面存在速度缓慢的缺点，且大量I/O操作消耗的性能也比较大，因此为了提高性能，以Redis为代表的内存数据库应运而生。Redis是一个非结构化数据库，支持使用非结构化语言(Not-only Structured Query Language,No SQL)查询。同时，Redis通过I/O(Input/Output)多路复用和DRAM(Dynamic Random Access Memory)提供了高吞吐、高并发和低时延的服务，在数据缓冲、消息队列、Key-Value存储等场景都发挥了重要的作用。

## 2.4 Vue框架

### 2.4.1 MVVM模式

MVVM是Model-View-ViewModel的缩写[7]。MVVM模式分为View、ViewModel、Model三层。其中View 层为视图层，是与用户直接交互的部分。Model层对服务器里的数据进行二次封装，以模型的形式表现出来。ViewModel层是MVVM模型的核心层，起着连接的作用，它用于将视图层中的数据同步更新到模型层，并把Model层的数据变化传递给View层。

### 2.4.2 Vue框架

Vue.js 是一个基于MVVM模式的前端框架，也是一个用于构建用户界面的渐进式框架[8]。Vue的设计方式是自底向上逐层开发，开发人员易于使用[9]。总体而言,使用Vue框架有以下优点:

1. 简单容易理解，学习成本低，开发速度快。
2. Vue模块提供了过滤器、指令，开发者可以非常方便的操作DOM。
3. 数据双向绑定，开发者使用Vue规定的语法即可获取数据的更新，完成数据与页面的交互。
4. 路由机制强大。

## 2.5 uni-app

uni-app 被广泛认为是较好的小程序开发框架、App 跨平台框架和方便的 H5 开发框架[10]。uni-app框架是基于Vue.js开发的，通过uni-app，可以开发跨平台产品，因此，如果对Vue有一定的了解，那么可以快速掌握这个框架，当然，如果也掌握微信applet，也能快速掌握它，因为uni-app封装并兼容微信小程序的组件和api。开发者编写一组代码，可以发布到多个平台，如iOS、Android、Web(响应式)，以及各种小程序(微信/支付宝/百度/今日头条/QQ/钉钉/淘宝)、快应用。uni-app功能框架图如图2-1所示。



图2-1 uni-app功能框架图[11]

与其他移动端跨平台框架(如Cordova/ReactNative/NativeJS)相比，uni-app在跨端平整、开发成本、性能体验、周边生态、扩展灵活性等方面具有明显优势。

1. 跨端数量多。
2. 性能体验优秀。
3. 生态开放丰富。
4. 开发成本低。

## 2.6 本章小结

本章主要概述了基于Spring Boot和Vue框架开发自动售货机管理系统需要用到的关键技术和理论内容。介绍了系统前后端分离架构需要用的技术点，以及移动端用的技术内容和开源框架，对选用的框架技术进行研究，叙述系统搭建所需用到的技术，为系统开发做好技术方面的准备工作。

# 第三章 系统需求分析

在软件系统的开发过程中，需求分析是软件开发的重要组成部分，在这一部分，为了得以开展有规划性的软件开发工作，通过分析市场需求，获取系统总体解决方案，再结合方案进行系统开发，避免“盲目摸索、错误的过程和结果”，正确的系统需求分析是软件实现的前提条件。本章主要围绕着系统业务需求、角色分析、系统功能性需求以及系统非功能性需求展开论述、总结。

## 3.1 可行性分析

基于Spring Boot和Vue框架的自动售货机管理系统可针对运营商管理分散在各个区域的自动售货机，以此提供实时、高效的管理能力，同时消费者可利用系统包含的移动端微信小程序，通过地图标记点获取附近区域内的自动售货机地理位置进行购物，也能够在微信小程序里下单商品，选择配送上门或者线下取货两种方式，还可以参与优惠活动，提高自动售货机的使用频率，降低企业和经营者在自动售货机上的运营成本和推广成本，提高企业、运营商的营业效率，带动整个社会的经济流动，解决了消费者四处寻找自动售货机的烦恼，同时支持线上购物，给消费者带来更好的购物体验。下面分不同层面对本系统的可行性进行分析。

### 3.1.1 技术可行性

从技术方面分析，PC端自动售货机管理系统采用前后端分离技术，后端使用Spring Boot技术框架结合Java语言，成熟的Spring Boot技术框架和Java语言使得开发工作变得更加轻松，拥有快速开发等优点，并且Spring Boot内嵌服务器（Tomcat）容器简化Web项目的开发。前端使用Vue框架结合iViewUI组件库使得页面开发变得十分简单，加上Vue框架里的数据双向绑定使得视图和数据之间的联系变得紧密。移动端采用微信小程序作为消费者使用的平台，微信小程序是依托在微信运行的程序，微信小程序的开发无需考虑Android、IOS等系统限制。拥有开发速度快、学习成本低等优点，使用uni-app作为开发框架，实现一套代码，可发布到各种快应用、小程序等平台上，页面组件使用uViewui组件库可以方便开发页面。拥有这些技术作为支撑，前后端框架配合数据库，实现数据的查询、显示等功能，微信小程序负责消费者与系统之间的联系，因此，根据上述技术实现系统的功能是可行的。

### 3.1.2 经济可行性

目前，在移动互联网的高速发展下，上网通讯基本融入了大部分人的生活中，而且现如今手机已经成为一个必不可少的通讯工具，其中微信更是通讯软件中的佼佼者，基本每个人都在使用，这为系统微信小程序部分提供了保障，用户基本无需经济投入，就能够使用系统的微信小程序进行信息浏览、购物，用较少的资金投入到微信小程序的功能模块开发中，产生的收益是可观的。自动售货机管理系统部分主要成本为核心业务功能的开发，总体研发成本并不多，也是企业、运营商所能接受的。

### 3.1.3 操作可行性

本系统移动端基于微信小程序，而微信小程序无需用户安装应用，搜索小程序名称即可进入自动售货机商城，实现信息浏览、商品购买。管理员通过系统管理后台可对自动售货机运行状态、商品数据进行系统性的管理。其操作难度不高，只需根据系统菜单栏的功能模块进行基本操作即可，减少管理工作的负担。系统采用的框架技术，也能对系统维护工作带来便利。

## 3.2 用户需求分析

### 3.2.1 系统需求概述

由于无人零售行业不断地发展，自动售货机的数量也随之增长，管理和运营自动售货机成为了企业、运营商棘手的问题，由于自动售货机过于分散、容易断货和易出现故障的缺点也给自动售货机经营者带来了业务数据分析难、较高的人工管理成本等问题。同时由于自动售货机过于分散，当消费者刚好看到售货机且需要购物的时候才会选择下单，而且部分用户需要购物时会存在找不到自动售货机地理位置等情况，同时也有部分用户由于各种原因不能出门进行购物。因此，针对以上问题，让高速发展的计算机、互联网参与到烦杂的信息管理工作中，拟设计基于Spring Boot和Vue框架的自动售货机管理系统，不仅能给消费者提供便利，还能大大提升企业、运营商的运营效率和降低管理难度，提升自动售货机管理环节的准确性，减少管理工作的复杂度，给消费者更好的购物体验。综上所述，消费者、设备维护员、财务、运维、管理员、系统管理员等用户对自动售货机管理系统都存在信息需求。

本自动售货机管理系统开发主要目的是使自动售货机管理、运营这两项工作更规范、更合理、更系统。设备维护员、财务、运维、管理员、系统管理员是后台管理系统的主要用户，消费者是移动端微信小程序的主要用户。综合以上内容，本系统的主要核心功能如下：

1. 设备管理。系统能够对运营的自动售货机进行设备添加，添加设备信息包括选择设备分组、设备信息，还应具备设备的经纬度及具体地址，设置设备的有效期；能对自动售货机设备配置货道商品，可添加货道商品，设备货道最大库存量，库存警告，是否启用该货道，编辑、删除货道等；能实时查看设备报警信息，比如库存不足、设备故障、广告故障、商品故障、卡货等信息，系统能通过自动售货机货道商品库存量不足进行补货提醒，维修人员可根据设备报警信息进行设备维修申请，管理员能对维修申请进行审核，同时系统能显示维修信息列表，维修人员可以根据维修结果对维修状态进行更改。
2. 商品管理。系统可以将待出售的商品进行增加，增加的商品信息有，商品名称价格、计量单位、商品品牌等，还有商品图片之类的资料，管理人员可审查新增商品，编辑商品信息、下架商品等，还可配置商品分类、品牌、规格、计量单位等。
3. 销售管理。系统能查看用户购买商品订单记录，可对线上下单的订单进行收货地址的修改，打印发货单，也可导出全部订单为XLS工作表，对用户申请退货的订单进行审核，能查看交易投诉，售后原因，能查看收款记录，退款流水。
4. 会员管理。系统能够对微信小程序注册的会员进行管理，同时系统能够注册、查看、编辑、禁用会员，还应对禁用的会员放置回收站管理，可对回收站的会员进行查看、启用、编辑操作。还应对会员评价、积分历史、会员资金、充值记录、提现申请进行操作管理。
5. 商品预定。用户通过微信小程序点击预定选项，进入到商品预定页面，用户可以通过城市名称或自动售货机名称搜索该城市内的自动售货机，选择某个自动售货机可以浏览到该自动售货机内的商品， 商品能显示配图、价格名称等信息，点击商品能够进入商品详情页面，可以选择把商品添加至购物车或立即下单预定，点击预定按钮跳转到填写订单页面，用户可选择开发票、备注，同时可使用优惠券进行购物，点击微信支付按钮付款，付款完成，提示订单支付成功， 在订单信息里面提供了取货码和自动售货机的具体位置信息，供用户到现场进行商品取货。
6. 查找设备。用户通过微信小程序点击查找设备，可按城市查询、按设备名称查询，选择设备项能通过地图查看设备地理位置坐标，也能显示设备地址信息、联系电话、微信号等信息。

### 3.2.2 需求用例分析

1. 设备管理。该功能点主要通过管理员、运维、设备维护员进行使用和操作，管理员可操作设备分组、添加设备、配置货道商品、禁启用设备、查看设备报警、查看补货提醒、查看维修列表、维修审核。运维可操作设备分组、添加设备、配置货道商品、禁启用设备、查看设备报警、查看补货提醒、查看维修列表。设备维护员可查看设备报警、查看补货提醒、填写维修申请、查看维修列表、设置维修状态、配置货道商品数量。设备管理用例图如图3-1所示：

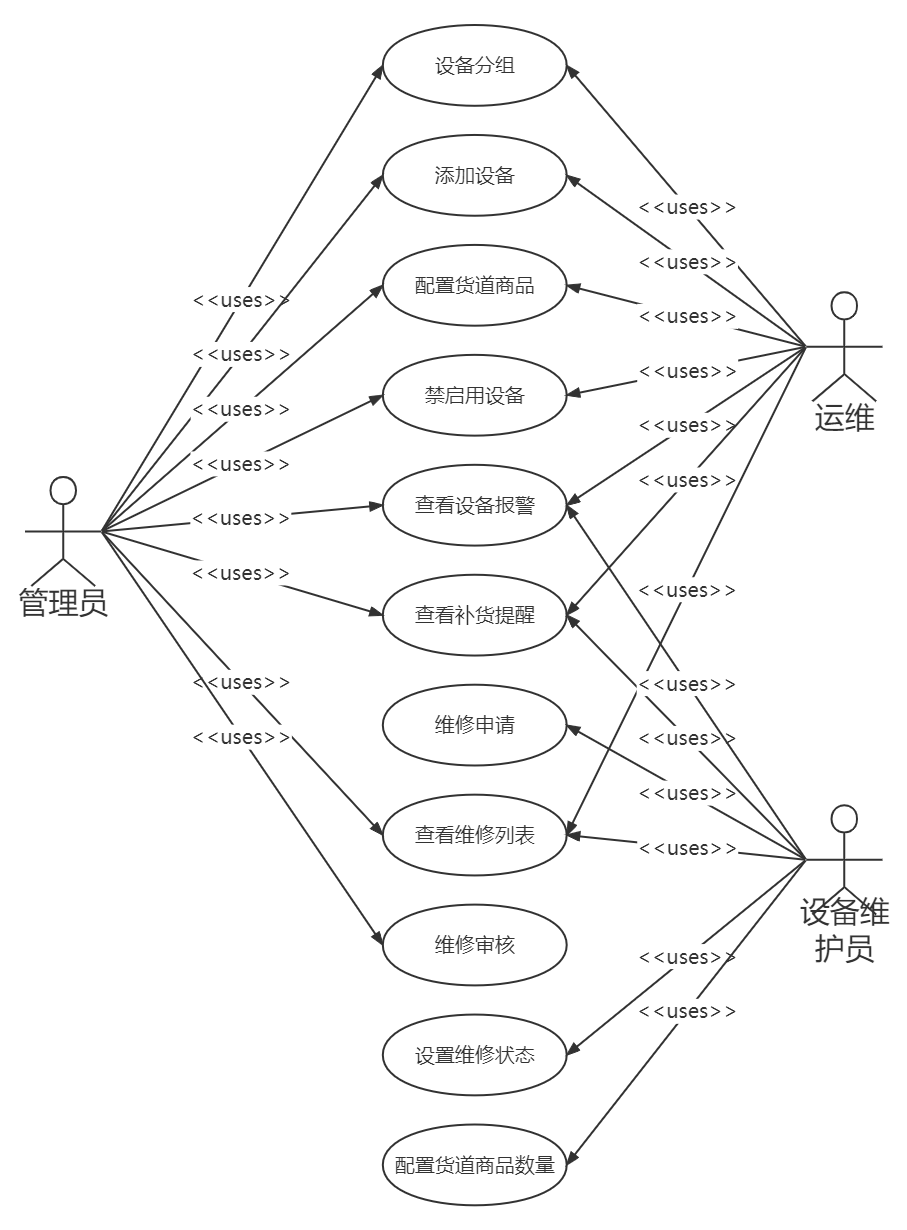


图3-1 设备管理用例图

设备分组。该模块主要对自动售货机分组进行管理，主要功能有添加分组、编辑、禁启用设备分组、分组配置。设备分组的用例说明详见表3-1所示：

表3-1 设备分组的用例说明

|  |  |
| --- | --- |
| 用例名称 | 设备分组 |
| 简要描述 | 对自动售货机进行分组管理 |
| 主要执行者 | 管理员、运维 |
| 前置条件 | 管理员、运维登录系统 |
| 基本事件流 | 1. 管理员、运维进入特征管理的Web页面。 2. 管理员、运维选择导航栏设备分组选项。 3. 管理员、运维选择添加分组、编辑、禁用、启用、分组配置按钮。 4. 管理员、运维填写相关表单，并且点击确定。 5. 前端将表单请求发送至后端系统。 6. 后端业务系统根据业务逻辑对请求进行处理，返回处理后的最终结果。 7. 前端展示处理结果。 |
| 异常事件流 | 1. 无法添加 2. 维护数据 3. 服务器繁忙，请稍后重试 |

货道商品。该模块主要对自动售货机的货道进行商品配置，主要功能有添加、编辑、删除、查看货道商品。货道商品的用例说明详见表3-2所示：

表3-2 货道商品的用例说明

|  |  |
| --- | --- |
| 用例名称 | 货道商品 |
| 简要描述 | 对自动售货机进行货道商品配置 |
| 主要执行者 | 管理员、运维 |
| 前置条件 | 管理员、运维登录系统 |
| 基本事件流 | 1. 管理员、运维进入特征管理的Web页面。 2. 管理员、运维选择导航栏设备列表选项。 3. 管理员、运维点击“货道商品”按钮。 4. 系统显示货道商品页面。 5. 管理员、运维选择添加货道商品、编辑、删除按钮。 6. 管理员、运维填写相关表单，并且点击确定。 7. 前端将表单请求发送至后端系统。 8. 后端业务系统根据业务逻辑对请求进行处理，返回处理后的最终结果。 9. 前端展示处理结果。 |
| 异常事件流 | 通过异常捕捉，获取到后端抛出的异常信息，前端页面进行异常提示。 |

1. 商品管理。该功能点主要通过管理员、运维进行使用和操作，管理员可操作商品列表、商品审核、商品分类、品牌管理、规格管理、计量单位，运维可添加商品、查看商品列表。商品管理用例图如图3-2所示：

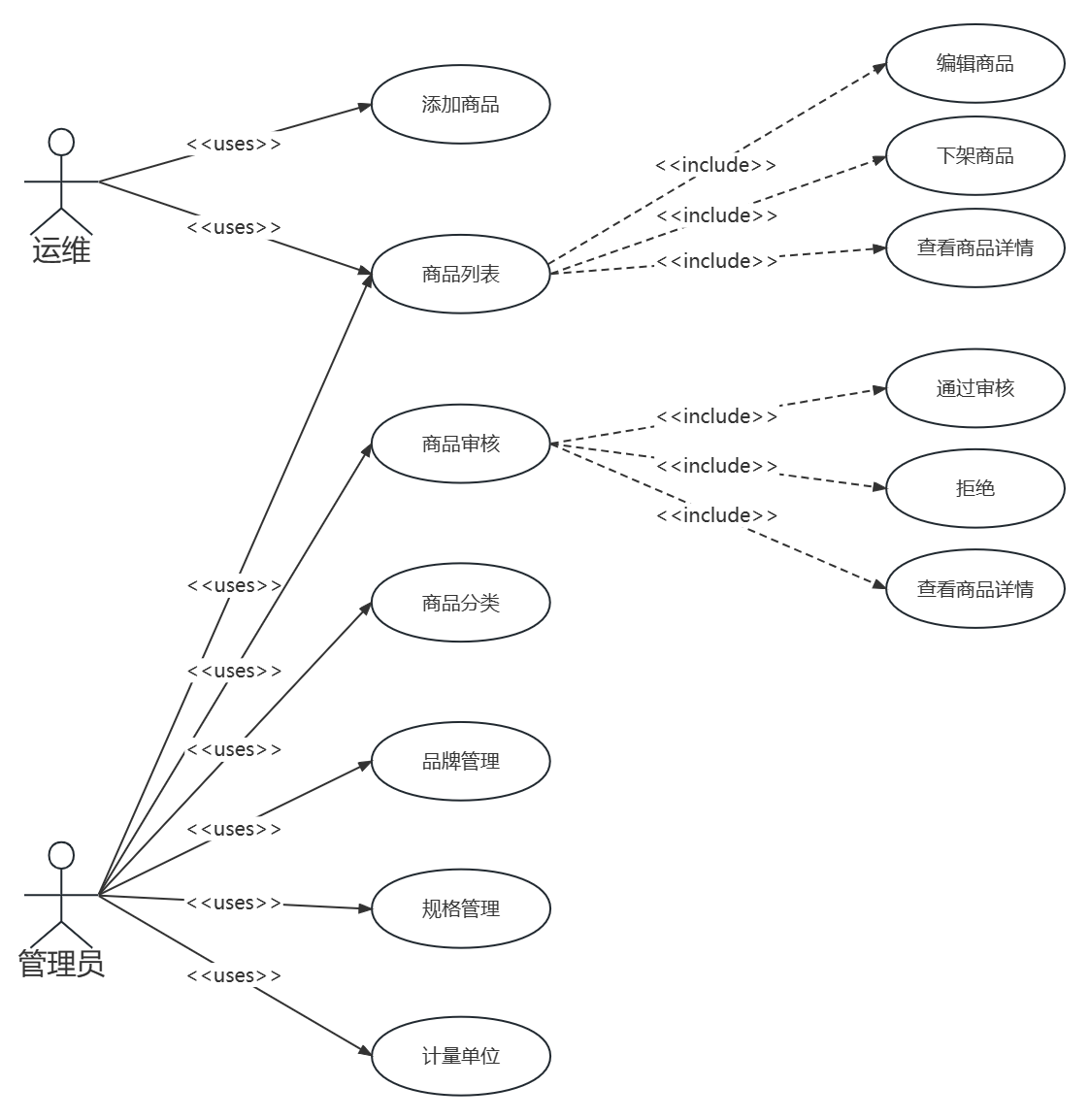


图3-2 商品管理用例图

添加商品。该模块主要对自动售货机售卖的商品进行添加，主要功能选择商品品类、填写商品详情、保存商品、保存为模板、商品发布成功。添加商品的用例说明详见表3-3所示：

表3-3 添加商品的用例说明

|  |  |
| --- | --- |
| 用例名称 | 添加商品 |
| 简要描述 | 对自动售货机售卖的商品发布 |
| 主要执行者 | 运维 |
| 前置条件 | 运维登录系统 |
| 基本事件流 | 1. 运维进入特征管理的Web页面。 2. 运维选择导航栏添加商品选项。 3. 系统显示实物商品、商品模板导入选项。 4. 运维点击实物商品项。 5. 系统显示选择商品品类。 6. 运维选择商品品类。 7. 系统显示填写商品详情页面。 8. 运维填写相关商品信息。 9. 运维点击保存商品。 10. 系统显示商品发布成功。 |
| 异常事件流 | 通过异常捕捉，获取到后端抛出的异常信息，前端页面进行异常提示。 |
| 扩展点 | 3a：若运维点击商品模板导入项。  3a\_01：系统显示模板选项。  3a\_02：运维点击模板项。  3a\_03：系统显示该模板的商品信息  3a\_04：运维选择修改信息、保存商品。  3a\_05：系统显示商品发布成功。  8a：若运维点击保存为模板。  8a\_01：系统显示是否确认保存。  8a\_02：运维点击保存按钮。  8a\_03：系统显示保存成功。 |

1. 销售管理。该功能点主要通过管理员、运维进行使用和操作，管理员可操作商品订单、售货管理、交易投诉、售后原因、收据记录、退款流水，运维可操作商品订单。销售管理用例图如图3-3所示：

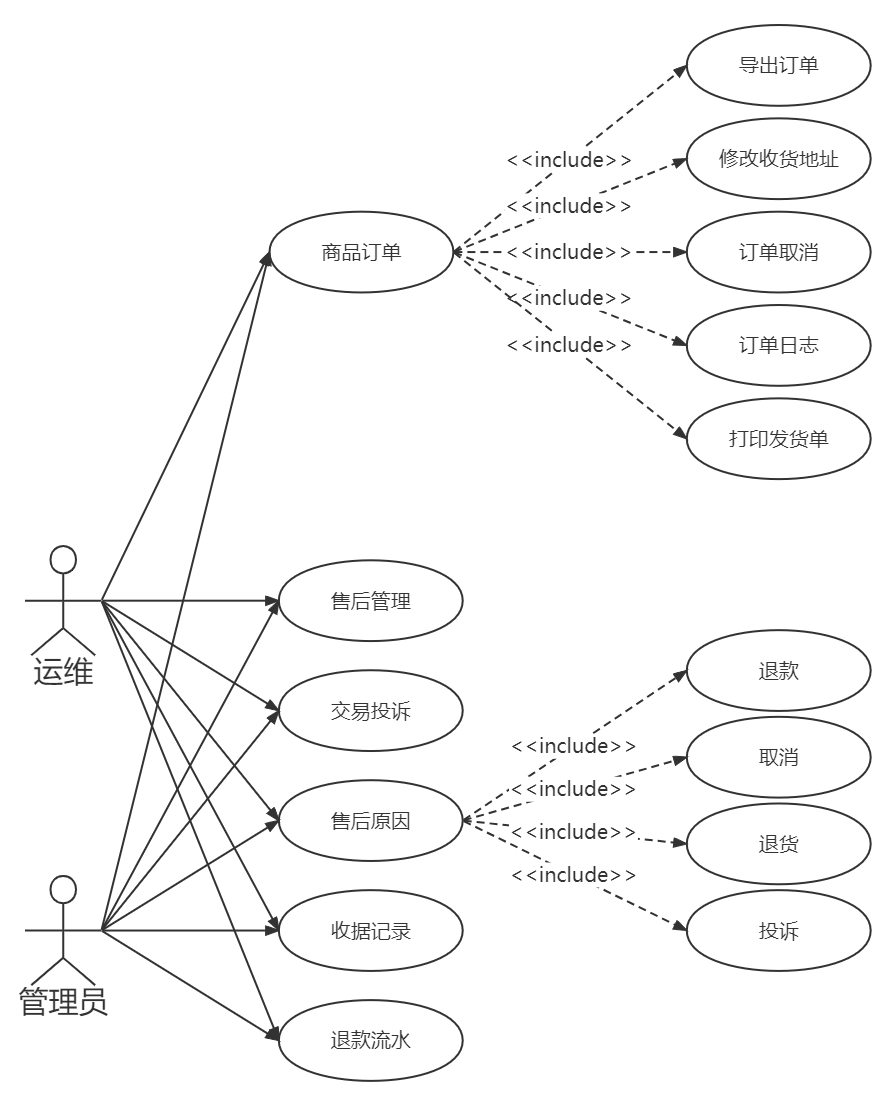


图3-3 销售管理用例图

1. 会员管理。该功能点由管理员进行使用和操作，包括回收站、会员评价、会员管理、积分历史、充值记录、会员资金、提现申请。会员管理用例图如图3-4所示：

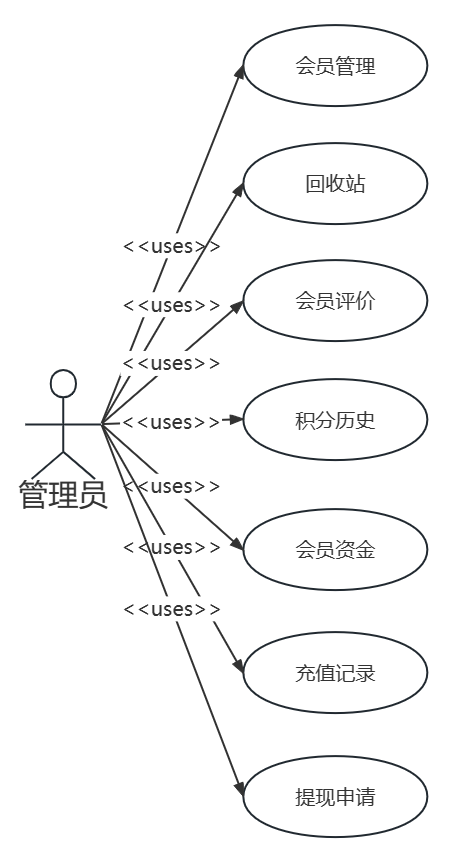


图3-4 会员管理用例图

1. 商品预定（微信小程序）。消费者登录进入主页面，选择预定商品、查询设备、查看商品、查看购物车、查看订单等，使用的功能有：预定商品、查询设备、选择商品、立即预定、加入购物车、查看购物车、填写订单、提交订单、查看订单、取消订单、查看取货码、查看订单关联设备信息。商品预定用例图如图3-5所示：

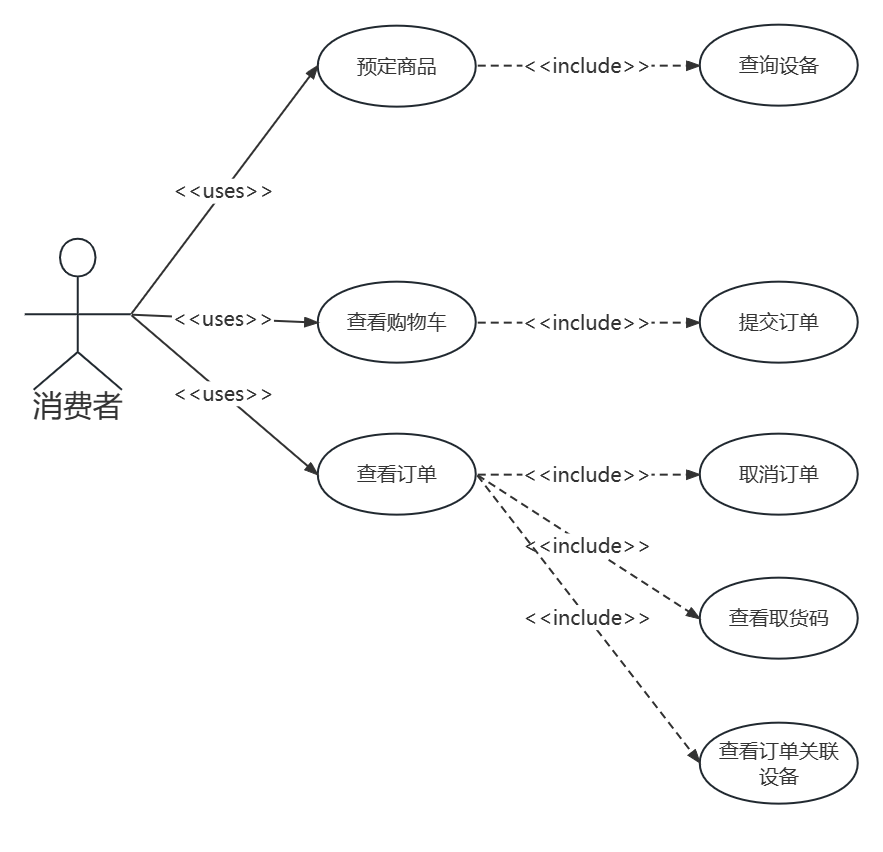


图3-5 商品预定用例图

1. 查找设备（微信小程序）。消费者登录进入主页面，可选择查找设备、选择设备。使用的功能有：查询设备、按城市查询、按设备名称查询、选择设备。查找设备用例图如图3-6所示：

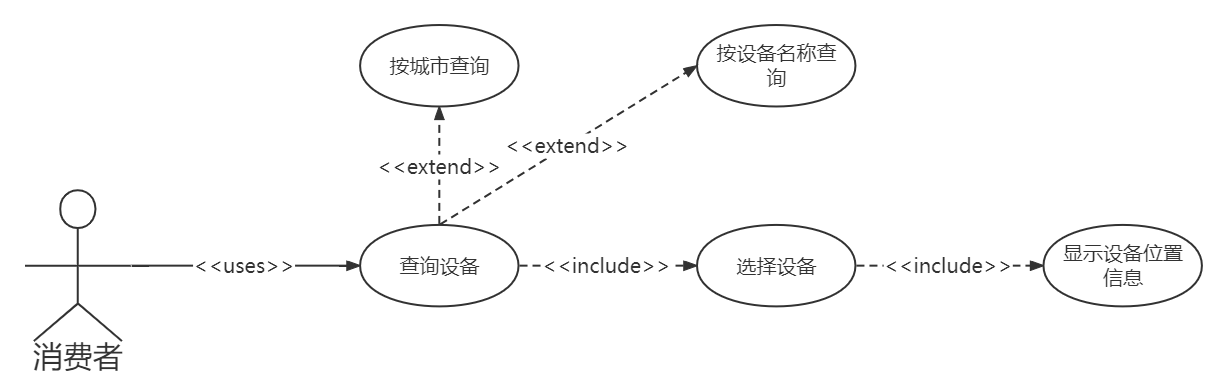


图3-6 查找设备用例图

## 3.3 系统性能需求

1. 并发量：针对系统的并发量设计搭建集群服务器，使用Nginx做负载均衡，按照一定的算法（权重、[轮询](https://so.csdn.net/so/search?q=%E8%BD%AE%E8%AF%A2&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/qq_35031260/article/details/_blank)），将客户端请求转发到不同应用服务器上，减轻单个服务器压力，提高系统并发量。并能承受上万并发量请求。
2. 用户量：结合系统总体需求，用户量最少能够容纳上万人同时在线操作和使用，根据较大的用户并发量，分配更高的性能提升每一个用户每一秒对服务器发送请求的相应时间。

## 3.4 非功能性需求

非功能性需求分析是为了自动售货机管理系统快速、跨界、安全稳定的运行，让用户得到较优的服务体验，以及为自动售货机管理系统管理者提供舒适的操作和提高工作效率。

1. 安全性：系统采用滑块验证码进行安全登录验证，确保系统的登录安全性；使用Spring Security安全框架，保证系统的登录、权限、加密、缓存等保护数据安全功能，并且可以自动防御很多网络攻击，例如CSRF攻击，会话固定攻击等，同时还提供了HTTP防火墙来拦截大量的非法请求。作为一个开放平台，Spring Security还通过引入第三方依赖关系，支持更多的认证方法，同时，也可以自定义认证逻辑，极大提升了系统的安全性。
2. 性能特性：针对当前高并发的请求访问，系统使用Redis对高访问数据进行缓存，可以极大的减少对执行较慢的I/O操作，减少对数据库访问造成资源的消耗。并且通过Elasticsearch搜索引擎的倒排索引能极快检索出数据信息，大大降低了系统的并发压力，保证系统能够稳定、高效运行。
3. 平台易用性：基于Spring Boot和Vue框架的自动售货机管理系统根据用户的舒适体验所设计，各类功能模块界面简洁，并且通过图标、输入提示语和悬停提示等设计功能，基本上符合使用者的（易用）操作。
4. 可维护性：根据本系统的设计需求，设计了菜单管理、系统设置、oss资源、行政地区、信任登录、支付设置、验证码等功能，管理者可通过相应模块直接进行设置，减少了后期代码的维护工作。并且通过MyBatis-Plus持久层框架，使各业务层与数据层解耦，根据各类日志信息保证了系统的可维护性。

## 3.5 本章小结

该章主要分四部分对系统的需求进行了综述，第一部分为可行性分析，第二部分在对系统进行可行性分析基础上入手，深度剖析了用户需求，得出软件需求，第三部分对系统的性能需求进行了简要概括，第四部分对非功能性需求进行了描述，最后得到了系统的总体需求，为后续工作做好准备。

# 第四章 系统总体设计

本章主要根据第三章需求分析对系统进行总体设计，主要讨论内容包括业务平台架构设计、部署架构设计、系统总体模块设计、数据库设计等方面。

## 4.1业务平台架构设计

根据系统需求分析，该系统由管理后台和移动端微信小程序两大部分组成，其中管理后台涉及业务繁杂，微信小程序有支付需求，因此分层设计系统是一个很不错的选项，分层设计可以简化系统设计，让不同的“参与者”专注于某一层面的事物，不仅利于维护，还有良好的可扩展性，根据MVC设计原理将系统架构分为五层架构，各层的具体功能如下：

1. 视图层。通过视图层能够使得消费者和系统管理用户实现交互操作。本系统主要使用简单易开发 且流行的Vue前端框架，再结合iView UI组件库使得页面更加美观简洁，微信小程序使用uni-app作为开发框架，UI组件库使用uViewui使得开发更加高效。
2. 网关。使用Nginx负载均衡可以将多个请求分摊到多台服务器上，以此来提高系统的可靠性与可伸缩性。当用户在视图层触发相应操作后，发送请求时，先向负载均衡服务器发送请求，再由负载均衡服务器按照配置规则转发至不同web服务器。
3. 控制层。接收到负载均衡服务器的转发请求，由控制层映射请求地址，请求参数转移到业务层的处理逻辑中，从业务层获得的数据通过SpringMVC框架返回到视图层中进行数据展示。
4. 业务层。主要承担一切与业务有关的任务，经过业务层逻辑处理后，能够将数据高效的梳理出来，提供用户真正需要的数据，尽可能减少对数据层的调用，降低数据库的消耗，大大提升系统的稳定性和日后维护工作的效率。
5. 数据层。又称数据持久层，按照业务层逻辑对数据库数据进行调用。本系统使用MyBatis-Plus持久层框架。将操作的逻辑与数据访问分开。通过redis、mongoDB做高访问数据缓存，提高数据的访问效率。

根据本系统软件架构分析，分析和设计出平台架构图，能够更加有效地解决系统多端任务，业务繁杂和请求访问量大等问题，结合得出本系统的架构：客户端、前端交互、网关、API层、能力层、服务层、持久层、基础支撑，业务平台架构图如图4-1所示：

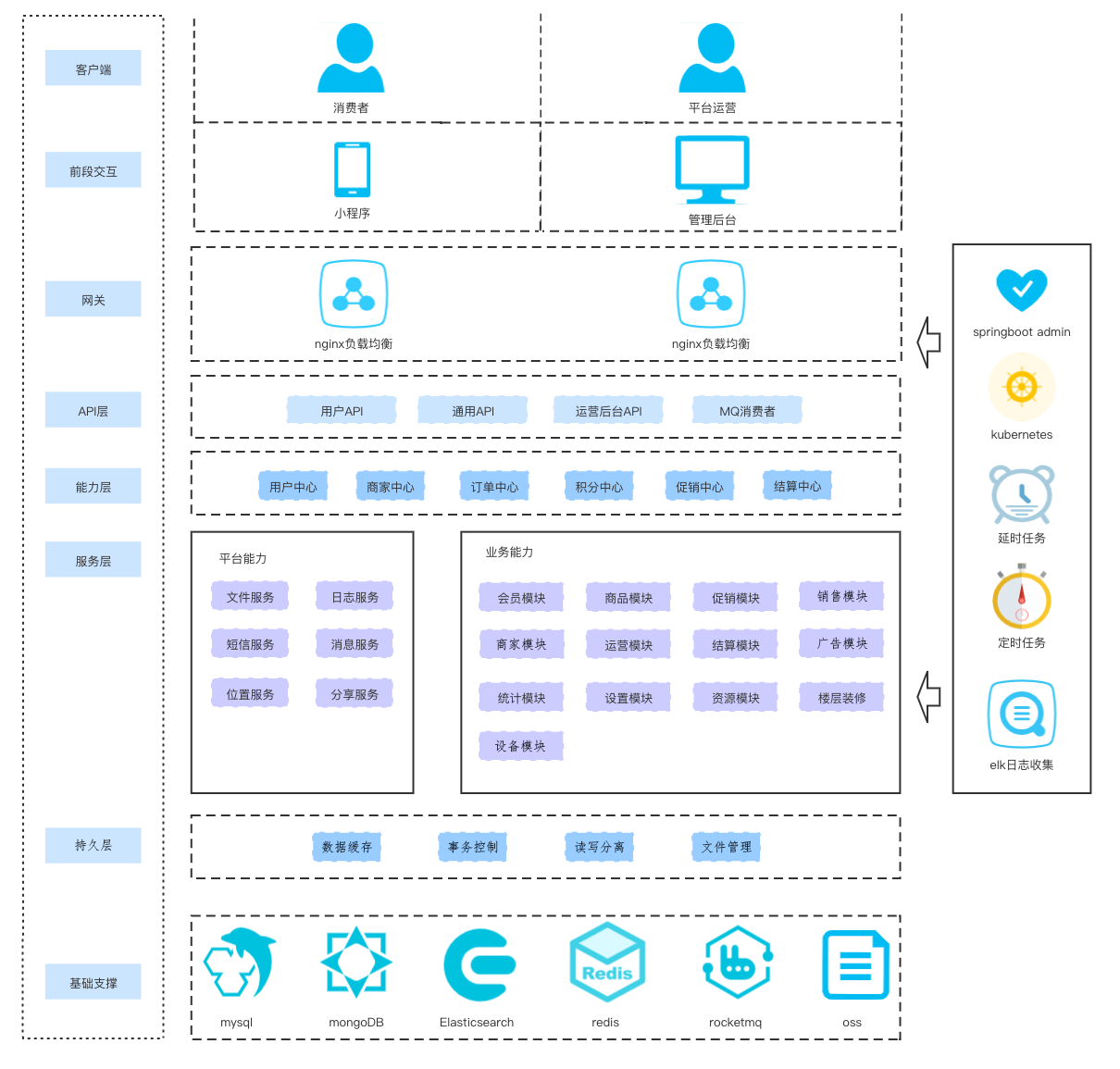


图4-1 业务平台架构图

## 4.2部署架构设计

本系统总体架构采用B/S结构，采用前后端分离开发模式，业务逻辑由服务端进行，而客户端只需要web浏览器，根据业务需求系统还配备移动端微信小程序，便于系统后续运维升级，使用Nginx负载均衡可以将多个请求分摊到多台服务器上，其次对web登录进行认证，应用服务器分多个中心，有助于系统的稳定性。最后部署MySql数据库集群、Redis数据缓存集群等。本系统部署架构图如图4-2所示：

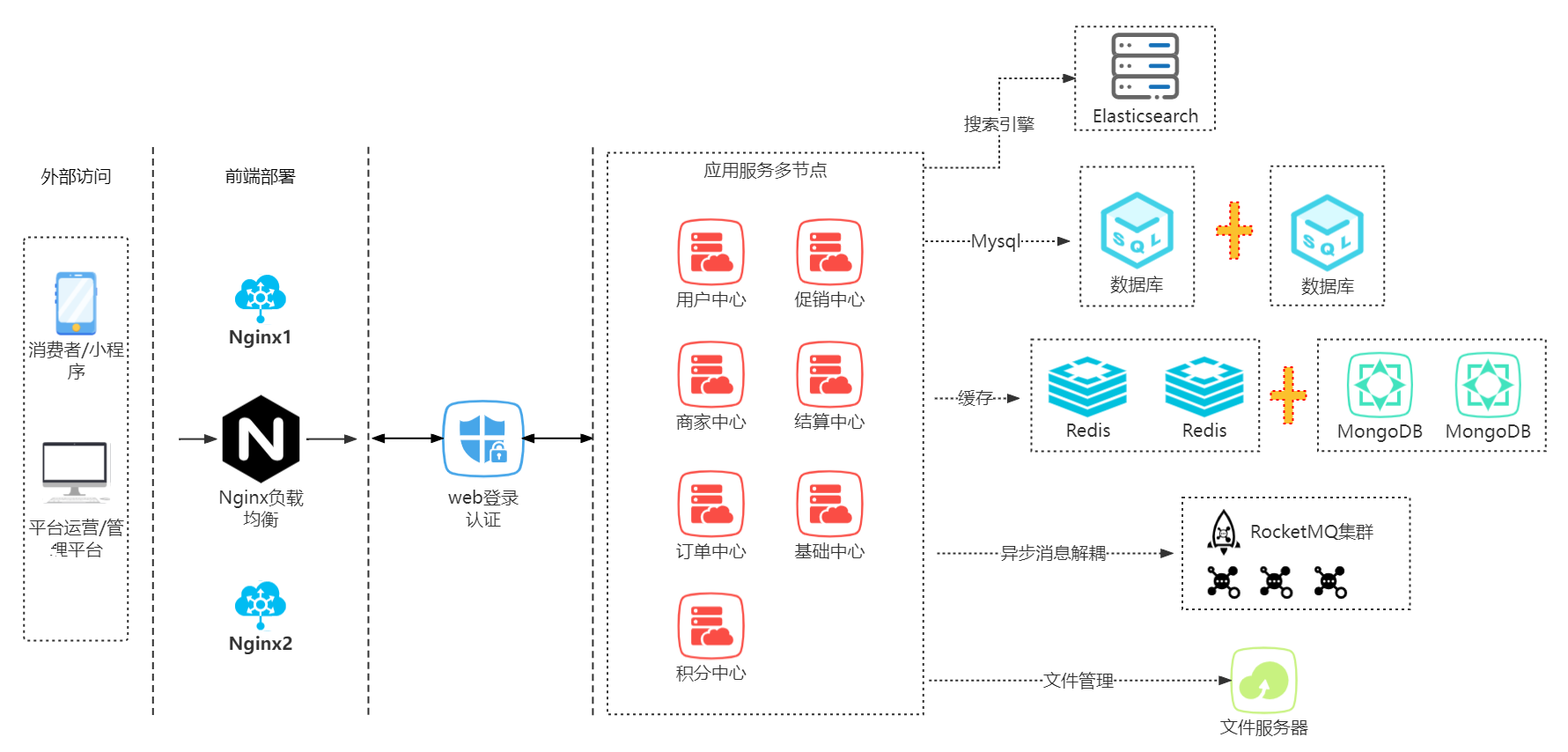


图4-2 部署架构图

## 4.3系统总体模块设计

分工协作是加快工作效率的一个好方法，模块化设计就是采用了“分工协作”、“分而治之”的思想，把一个项目，分成不同的模块，不同模块再进行细分，直至能够更加方便地解决问题为止。

模块化编程拥有分工实现、适合多人协作；便于维护、管理、移植等特点，实现系统的高内聚、低耦合，更加方便系统后续的维护工作。

根据系统的需求分析的研究结果建立模块结构图，以系统需求分析为出发点，对系统的功能结构进行设计，设计出有特色、有风格的自动售货机管理系统。本系统微信小程序前台主要有七大功能模块，管理后台主要有十一大功能模块，如图4-3所示：

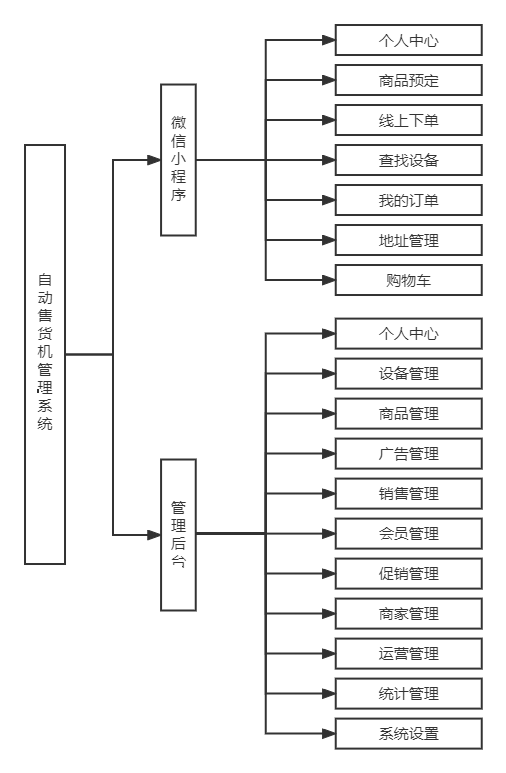


图4-3 系统功能结构图

### 4.3.1微信小程序

1. 个人中心。主要提供用户编辑个人信息，查看售后信息、评价、消息、积分、优惠券、领劵中心、砍价记录、系统信息，同时支持用户进行每日签到、意见反馈等操作。
2. 商品预定。商品预定供用户进行线上预约商品使用，下单成功后可通过订单提供的取货码和设备位置信息进行线下取货。
3. 线上下单。线上下单供用户进行线上下单商品，填写配送地址，由人工或机器人

方式进行商品配送，完成订单。

1. 查找设备。查找设备供用户在平台上查找自动售货机设备的地址位置信息，通过地图实时坐标的方式显示位置，用户可按城市查找或按设备名称查找。
2. 我的订单。我的订单供用户查看订单的信息，可获取订单的状态，查看取货码、设备地理位置信息，取消订单等操作。
3. 地址管理。地址管理供用户添加收货地址使用，支持添加更多收货地址，设置默认地址等。
4. 购物车。供消费者对加入购物车的商品进行操作，可操作商品数据信息对其进行增、删、改、查，也可以对选中的商品进行结算。

### 4.3.2管理后台

1. 个人中心。个人中心主要功能是给系统用户浏览和修改个人信息使用，可修改用户头像、昵称，安全设置可对账户密码进行修改。
2. 设备管理。该功能是对自动售货机设备进行数据的增、删、改、查操作，子模块分设备分组、设备列表、设备报警、补货提醒、维修管理，维修管理又分为维修申请、维修列表、维修审核、维修状态四个模块。
3. 商品管理。商品管理功能主要是对自动售货机所需售卖的商品进行管理，子模块分添加商品、商品列表、商品审核、商品分类、品牌列表、规格列表、计量单位。
4. 广告管理。广告管理功能主要用于管理投放在自动售货机显示屏上的广告，子模块分广告列表、广告分组、设备广告配置、广告投放记录。
5. 销售管理。销售管理主要用于对自动售货机微信小程序平台订单或线下自动售货机产生的订单进行管理。
6. 会员管理。会员管理主要用于对微信小程序用户进行管理，子模块分会员列表、回收站、会员评价、积分历史、会员资金、充值记录、提现记录、提现申请。
7. 促销管理。促销管理功能主要用于对消费者购物给予相应的优惠活动。子模块分优惠券、券活动、砍价活动等等。
8. 商家管理。商家管理功能主要用于第三方企业、运营商加盟本系统使用，子模块分商家列表、商家审核、商家结算。
9. 运营管理。运营管理主要是对微信小程序平台进行管理，子模块分移动端管理，搜索热词，意见反馈。
10. 统计管理。可对系统重要数据进行统计，包括会员统计、订单统计、销量统计、商品统计。
11. 系统设置。主要功能用于对系统部分数据进行设置，分用户管理、菜单管理、部门管理、角色管理、系统设置、oss资源、行政地区、信任登录、支付设置、验证码。

## 4.4数据库设计

数据库是搭建系统的基础，是构建自动售货机管理系统后台数据存储的重要环节。数据库中数据实体及其关系的设计会影响数据的操作效率，如何更加合理的设计数据库关系着自动售货机管理系统开发与运行的工作效率。

为保证系统的工作效率，设计结构清晰合理的数据库是需要开发人员重点考虑的方面。

### 4.4.1 数据库实体联系

在数据库表设计过程中，基类可直接设计成表，通过对系统的需求分析，可以得到管理员类和会员类、设备类等基类。以下给出部分数据实体属性图：

管理员信息在自动售货机管理系统中处于核心实体类地位，管理员实体属性主要有备注、邮箱、管理员登录密码、昵称、管理员头像、手机号、管理员用户名、状态等，如图4-4所示。

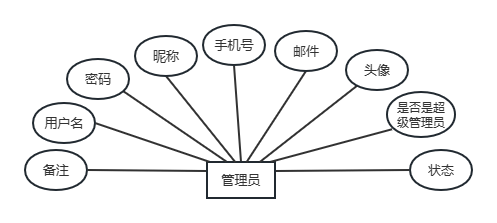


图4-4 管理员实体属性图

会员实体类的主要属性有昵称、会员等级、积分数量、会员密码、积分总数量、会员性别、会员生日、会员地址、手机号码、会员头像、会员状态、会员用户名等，如图4-5所示。

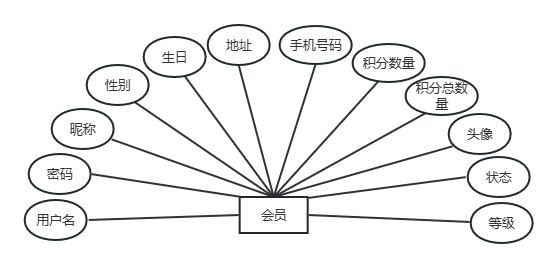


图4-5 会员实体属性图

设备实体类的主要属性有设备组名称、设备编码、设备名称、是否运营、是否故障、开始时间、结束时间、在线状态、运维人员姓名、运维人员电话、地址、经纬度、备注等，设备实体属性图如图4-6所示。

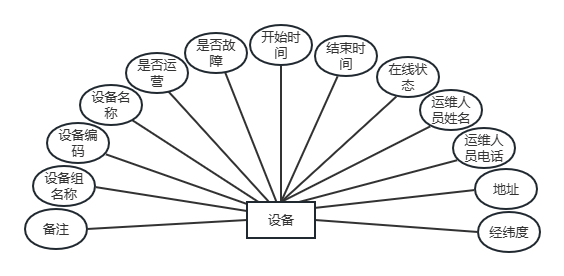


图4-6 设备实体属性图

综上，结合各个实体属性和系统需求分析后，得出系统数据库E-R图，系统数据库E-R图如图4-7所示。

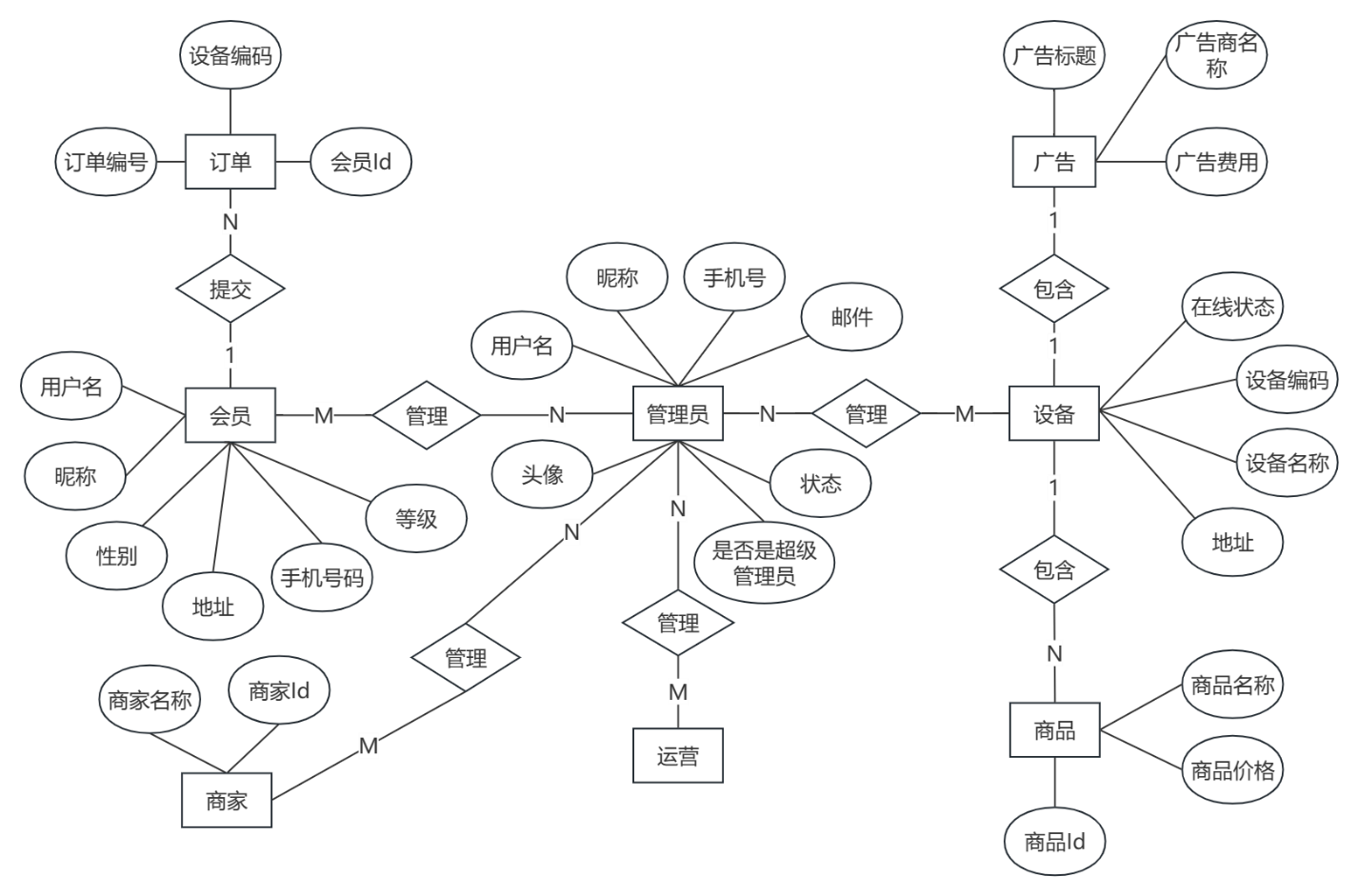


图4-7系统E-R图

### 4.4.2 数据表结构设计

本系统采用MySQL关系型数据库作为数据存储库，再根据数据库概念设计，结合三大范式（1NF、2NF、3NF）的满足关系。通过对实体类之间的细致关系进行分析，以及对实体类的属性名称、属性类型等具体内容的设计，得出系统数据库中8张核心数据库表。

1. 管理员表

管理员表（li\_admin\_user）包括唯一标识、管理员用户名、密码、昵称、手机号、邮箱、管理员头像、是否是超级管理员、状态、备注、所属部门id、角色id集合、创建者、创建时间、更新者、更新时间、删除标识。如表4-1管理员表所示。

表4-1 管理员表（li\_admin\_user）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段说明 | 字段名 | 数据类型（长度） | 是否为空 | 备注 |
| 唯一标识 | id | bigint(19) | 否 | 主键，雪花分布式id |
| 管理员用户名 | username | varchar(200) | 否 |  |
| 密码 | password | varchar(255) | 否 |  |
| 昵称 | nick\_name | varchar(255) | 否 |  |
| 手机号 | mobile | varchar(20) | 是 |  |
| 邮箱 | email | varchar(255) | 是 |  |
| 管理员头像 | avatar | varchar(1000) | 是 |  |
| 是否是超级管理员 | is\_super | bit(1) | 否 | 1-超级管理员  0-普通管理员 |
| 状态 | status | bit(1) | 否 | 1-正常  0-禁用 |
| 备注 | description | varchar(255) | 是 |  |
| 所属部门id | department\_id | bigint(19) | 否 |  |
| 角色id集合 | role\_ids | varchar(255) | 是 |  |
| 创建者 | create\_by | varchar(64) | 是 |  |
| 创建时间 | create\_time | datetime(6) | 是 |  |
| 删除标志 | delete\_flag | bit(1) | 否 | 1-删除  0-未删除 |
| 更新者 | update\_by | varchar(64) | 是 |  |
| 更新时间 | update\_time | datetime(6) | 是 |  |

1. 会员表

会员表（li\_member）包括会员用户名、昵称、会员密码、会员地址、会员性别、手机号码、积分数量、积分总数量、会员头像、会员地址id、会员状态、唯一标识、客户端、最后一次登录时间、会员等级id、经验值数量、会员生日、创建者、创建时间、更新者、更新时间、删除标识。如表4-2会员表所示。

表4-2 会员表（li\_member）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段说明 | 字段名 | 数据类型（长度） | 是否为空 | 备注 |
| 唯一标识 | id | bigint(19) | 否 | 主键，雪花分布式id |
| 会员用户名 | username | varchar(200) | 否 |  |
| 会员密码 | password | varchar(255) | 否 |  |
| 昵称 | nick\_name | varchar(255) | 否 |  |
| 会员性别 | sex | int(1) | 是 | 1-男  0-女 |
| 会员生日 | birthday | datetime(6) | 是 |  |
| 会员地址id | region\_id | varchar(255) | 是 |  |
| 会员地址 | region | varchar(255) | 是 |  |
| 手机号 | mobile | varchar(20) | 是 |  |
| 积分数量 | point | bigint(20) | 是 |  |
| 积分总数量 | total\_point | bigint(20) | 是 |  |
| 会员头像 | face | varchar(255) | 是 |  |
| 会员状态 | disabled | bit(1) | 否 | 1-正常  0-禁用 |
| 客户端 | client\_enum | varchar(255) | 是 |  |
| 最后一次登录时间 | last\_login\_date | datetime(6) | 是 |  |
| 会员等级id | grade\_id | bigint(19) | 是 |  |
| 创建者 | create\_by | varchar(64) | 是 |  |
| 创建时间 | create\_time | datetime(6) | 是 |  |
| 删除标志 | delete\_flag | bit(1) | 否 | 1-删除  0-未删除 |
| 更新者 | update\_by | varchar(64) | 是 |  |
| 更新时间 | update\_time | datetime(6) | 是 |  |

1. 设备表

设备表（li\_machine）包括唯一标识、设备组名称、设备组id、设备编码、设备名称、是否运营、是否故障、开始时间、结束时间、在线状态、运维人员姓名、运维人员电话、地址名称、地址id、详细地址、经纬度、备注、创建者、创建时间、更新者、更新时间、删除标识。如表4-3设备表所示。

表4-3 设备表（li\_machine）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段说明 | 字段名 | 数据类型（长度） | 是否为空 | 备注 |
| 唯一标识 | id | bigint(19) | 否 | 主键，雪花分布式id |
| 设备组名称 | machine\_group\_name | varchar(255) | 是 |  |
| 设备组id | machine\_group\_id | varchar(32) | 是 | 默认为0 |
| 设备编码 | machine\_no | varchar(255) | 否 |  |
| 设备名称 | machine\_name | varchar(255) | 否 |  |
| 是否运营 | operation\_state | bit(1) | 否 | 1-是  0-否 |
| 是否故障 | fault\_state | bit(1) | 否 | 1-是  0-否 |
| 开始时间 | start\_time | datetime(6) | 否 |  |
| 结束时间 | end\_time | datetime(6) | 否 |  |
| 在线状态 | on\_line | bit(1) | 否 | 1-在线  0-离线 |
| 运维人员姓名 | operation\_name | varchar(64) | 是 |  |
| 运维人员电话 | operation\_phone\_number | varchar(11) | 是 |  |
| 地址名称 | machine\_address\_path | varchar(255) | 否 | '，'分割 |
| 地址id | machine\_address\_id\_path | varchar(255) | 否 | '，'分割 |
| 详细地址 | machine\_address\_detail | varchar(255) | 否 |  |
| 经纬度 | machine\_center | varchar(255) | 否 | '，'分割 |
| 备注 | note | varchar(255) | 是 |  |
| 创建者 | create\_by | varchar(64) | 是 |  |
| 创建时间 | create\_time | datetime(6) | 是 |  |
| 删除标志 | delete\_flag | bit(1) | 否 | 1-删除  0-未删除 |
| 更新者 | update\_by | varchar(64) | 是 |  |
| 更新时间 | update\_time | datetime(6) | 是 |  |

1. 商品表

商品表（li\_goods）包括唯一标识、品牌id、分类path、卖点、商品详情、购买数量、更新者、库存、创建者、是否为推荐商品、商品好评率、更新时间、缩略图路径、上架状态、小图路径、原图路径、评论数量、卖家id、商品价格、卖家名称、审核状态、审核信息、下架原因、是否自营、商品移动端详情、商品视频、商品类型、商品名称、创建时间、计量单位、删除标识。如表4-4商品表所示。

表4-4 商品表（li\_goods）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段说明 | 字段名 | 数据类型（长度） | 是否为空 | 备注 |
| 唯一标识 | id | bigint(19) | 否 | 主键，雪花分布式id |
| 商品名称 | goods\_name | varchar(255) | 否 |  |
| 商品价格 | price | double(20) | 否 | 默认为0 |
| 品牌id | brand\_id | bigint(19) | 是 |  |
| 分类path | category\_path | varchar(255) | 是 |  |
| 计量单位 | goods\_unit | varchar(255) | 是 |  |
| 卖点 | selling\_point | varchar(255) | 是 |  |
| 上架状态 | market\_enable | varchar(255) | 否 |  |
| 商品详情 | intro | mediumtext | 是 |  |
| 购买数量 | buy\_count | int(10) | 是 |  |
| 库存 | quantity | int(8) | 否 |  |
| 商品好评率 | grade | double(20) | 是 |  |
| 缩略图路径 | thumbnail | varchar(255) | 是 |  |
| 小图路径 | small | varchar(255) | 是 |  |
| 原图路径 | original | varchar(255) | 是 |  |
| 评论数量 | comment\_num | int(11) | 是 |  |
| 卖家id | store\_id | bigint(19) | 否 |  |
| 卖家名称 | store\_name | varchar(255) | 是 |  |
| 审核状态 | auth\_flag | varchar(255) | 否 |  |
| 审核信息 | auth\_message | varchar(255) | 否 |  |
| 下架原因 | under\_message | varchar(255) | 否 |  |
| 是否自营 | self\_operated | bit(1) | 否 | 1-是  0-否 |
| 商品移动端详情 | mobile\_intro | mediumtext | 是 |  |
| 商品视频 | goods\_video | varchar(255) | 是 |  |
| 是否为推荐商品 | recommend | bit(1) | 是 |  |
| 商品类型 | goods\_type | varchar(255) | 是 |  |
| 创建者 | create\_by | varchar(64) | 是 |  |
| 创建时间 | create\_time | datetime(6) | 是 |  |
| 删除标志 | delete\_flag | bit(1) | 否 | 1-删除  0-未删除 |
| 更新者 | update\_by | varchar(64) | 是 |  |
| 更新时间 | update\_time | datetime(6) | 是 |  |

1. 设备广告表

设备广告表（li\_machine\_adverts）包括唯一标识、显示位置、广告标题、广告费用、广告商名称、开始时间、结束时间、排序、备注、是否开启、创建者、创建时间、更新者、更新时间、删除标识。如表4-5设备广告表所示。

表4-5 设备广告表（li\_machine\_adverts）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段说明 | 字段名 | 数据类型（长度） | 是否为空 | 备注 |
| 唯一标识 | id | bigint(19) | 否 | 主键，雪花分布式id |
| 显示位置 | show\_type | varchar(255) | 否 |  |
| 广告标题 | title | varchar(255) | 否 |  |
| 广告费用 | advert\_price | double(20) | 否 |  |
| 广告商名称 | advert\_shop\_name | varchar(64) | 是 |  |
| 开始时间 | start\_time | datetime(6) | 是 |  |
| 结束时间 | end\_time | datetime(6) | 是 |  |
| 排序 | sort | Varchar(30) | 是 |  |
| 备注 | note | varchar(255) | 是 |  |
| 是否开启 | disabled | bit(1) | 否 | 1-开启  0-关闭 |
| 创建者 | create\_by | varchar(64) | 是 |  |
| 创建时间 | create\_time | datetime(6) | 是 |  |
| 删除标志 | delete\_flag | bit(1) | 否 | 1-删除  0-未删除 |
| 更新者 | update\_by | varchar(64) | 是 |  |
| 更新时间 | update\_time | datetime(6) | 是 |  |

1. 优惠券表

优惠券表（li\_coupon）包括唯一标识、更新者、优惠券类型、折扣、活动名称、优惠券类型（分为免费领取和活动赠送）、创建时间、商家承担比例、活动描述、删除标识、发行数量、领取限制、更新时间、已被使用的数量、已被领取的数量、消费门槛、时间范围类型、有效期、商家名称、优惠券名称、商家id、面额、活动开始时间、活动结束时间、关联范围类型、范围关联的id、创建者。如表4-6优惠券表所示。

表4-6 优惠券表（li\_coupon）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段说明 | 字段名 | 数据类型（长度） | 是否为空 | 备注 |
| 唯一标识 | id | bigint(19) | 否 | 主键，雪花分布式id |
| 优惠券名称 | coupon\_name | varchar(255) | 否 |  |
| 优惠券类型 | coupon\_type | varchar(255) | 否 |  |
| 面额 | price | double(20) | 否 |  |
| 折扣 | coupon\_discount | double(20) | 否 |  |
| 优惠券类型 | get\_type | varchar(255) | 否 | 分为免费领取和活动赠送 |
| 商家承担比例 | store\_commission | double(20) | 否 | 平台发布时可以提供一定返点 |
| 活动描述 | description | varchar(255) | 是 |  |
| 发行数量 | publish\_num | int | 否 | 如果是0则是不限制 |
| 领取限制 | coupon\_limit\_num | int | 否 |  |
| 已被使用的数量 | used\_num | int | 否 |  |
| 已被领取的数量 | received\_num | int | 否 |  |
| 消费门槛 | consume\_threshold | double(20) | 否 |  |
| 时间范围类型 | range\_day\_type | varchar(32) | 是 |  |
| 有效期 | effective\_days | int | 否 |  |
| 商家名称 | store\_name | varchar(255) | 是 | 如果是平台，默认值为：platform |
| 商家id | store\_id | bigint | 是 | 如果是平台，默认值为 ：0 |
| 活动名称 | promotion\_name | varchar(255) | 否 |  |
| 活动开始时间 | start\_time | datetime(6) | 否 |  |
| 活动结束时间 | end\_time | datetime(6) | 否 |  |
| 关联范围类型 | scope\_type | varchar(255) | 否 | 类型分类：全品类、部分商品分类、指定商品。默认为：指定商品 |
| 范围关联的id | scope\_id | mediumtext | 是 |  |
| 创建者 | create\_by | varchar(64) | 是 |  |
| 创建时间 | create\_time | datetime(6) | 是 |  |
| 删除标志 | delete\_flag | bit(1) | 否 | 1-删除  0-未删除 |
| 更新者 | update\_by | varchar(64) | 是 |  |
| 更新时间 | update\_time | datetime(6) | 是 |  |

1. 意见反馈表

意见反馈表（li\_feedback）包括唯一标识、创建时间、会员名称、反馈内容、手机号、图片、类型。如表4-7意见反馈表所示。

表4-7 意见反馈表（li\_feedback）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段说明 | 字段名 | 数据类型（长度） | 是否为空 | 备注 |
| 唯一标识 | id | bigint(19) | 否 | 主键，雪花分布式id |
| 创建时间 | create\_time | datetime(6) | 否 |  |
| 会员名称 | user\_name | varchar(255) | 否 |  |
| 反馈内容 | context | varchar(500) | 否 |  |
| 手机号 | mobile | varchar(11) | 否 |  |
| 图片 | images | varchar(255) | 否 | 多个图片使用：(，)分割 |
| 类型 | type | varchar(255) | 否 | 类型分为：功能相关、优化反馈、其他 |

1. 预存款充值记录表

预存款充值记录表（li\_recharge）包括唯一标识、创建者、创建时间、充值订单编号、会员id、会员名称、充值金额、充值方式、支付状态、支付插件id、第三方流水、支付时间。如表4-8预存款充值记录表所示。

表4-8 预存款充值记录表（li\_recharge）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段说明 | 字段名 | 数据类型（长度） | 是否为空 | 备注 |
| 唯一标识 | id | bigint(19) | 否 | 主键，雪花分布式id |
| 创建者 | create\_by | varchar(64) | 否 |  |
| 创建时间 | create\_time | datetime(6) | 否 |  |
| 充值订单编号 | recharge\_sn | varchar(255) | 否 |  |
| 会员id | member\_id | bigint | 否 |  |
| 会员名称 | member\_name | varchar(255) | 否 |  |
| 充值金额 | recharge\_money | double | 否 |  |
| 充值方式 | recharge\_way | varchar(255) | 否 | 如：支付宝，微信 |
| 支付状态 | pay\_status | varchar(255) | 否 |  |
| 支付插件id | payment\_plugin\_id | bigint | 否 |  |
| 第三方流水 | receivable\_no | varchar(255) | 是 |  |
| 支付时间 | pay\_time | datetime(6) | 否 |  |

## 4.5 本章小结

本章着重介绍了系统总体设计的方向。通过对系统的需求分析，对系统平台架构、部署架构和总体模块进行了设计。最后阐述了系统数据库的设计，给出了系统的数据实体属性图、E-R图和数据表结构，为之后的系统开发做好了充分的准备。

# 第五章 系统详细设计与实现

在开发视图当中，架构设计的核心是前后端分离。本系统采用Spring Boot框架开发后端、前端使用Vue框架加上微信小程序作为主体架构实现。主体架构细分为展示、控制、业务、数据四层架构，在此基础上，进一步继续细化，以系统核心模块实现描述其详细设计，主要核心模块有设备部署、商品预定和设备搜索。本章将一一阐述各核心模块具体实现的人机界面设计、类图、流程图、序列图，最终通过具体代码编程实现前端界面和后端逻辑，将需求变为可运行操作的软件产品。

## 5.1主体架构细化以及实现

本系统的架构核心分为视图架构、逻辑架构、数据架构、开发架构,为了提高系统的性能、安全性、可用性、扩展性、伸展性，使用分层设计是有必要的，分层的设计有利于提高开发人员的开发效率，可使系统开发分工协作明确，用户使用更加稳定。因此本系统主要架构分展示层、控制层、业务层、数据层。如图5-1所示。

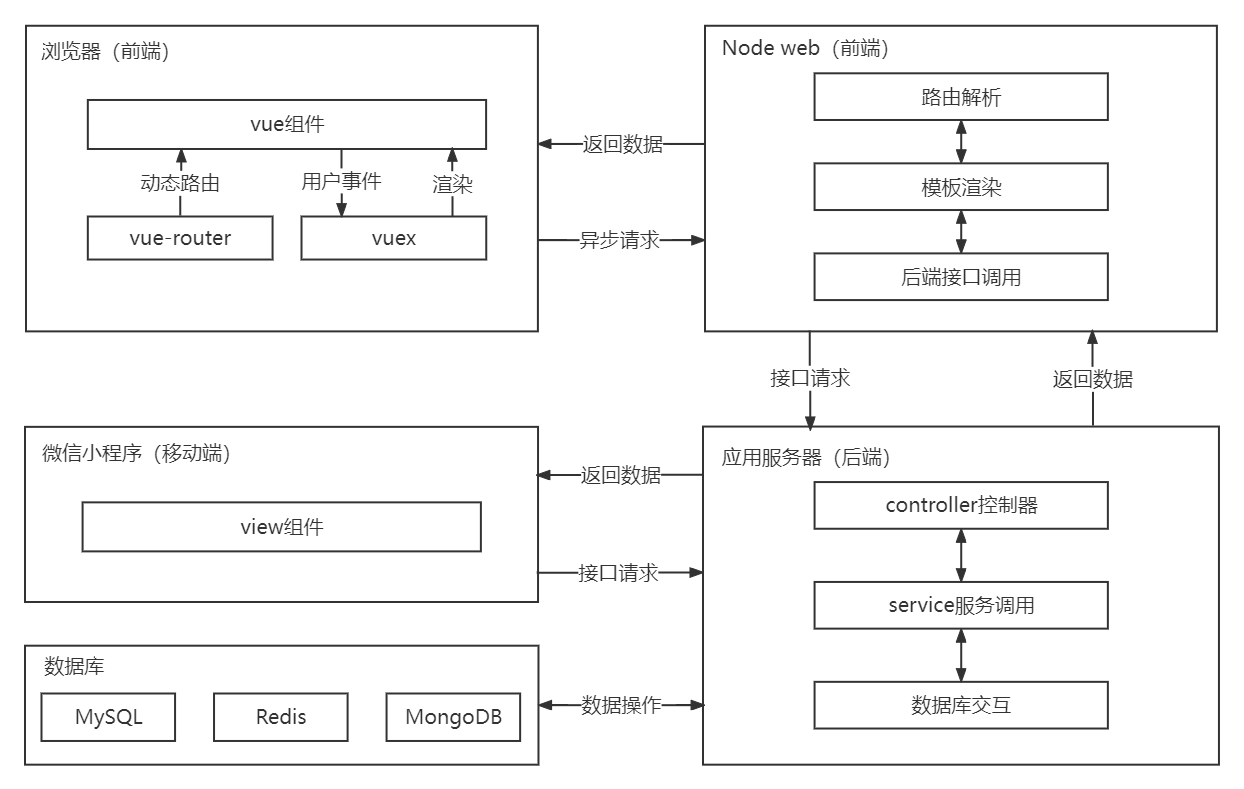


图5-1 数据交互图

### 5.1.1 展示层

展示层可以以视图信息方式展示出来，使用者可以通过触发相应组件获取对应的数据信息，组件触发相应的函数调用后端接口获取数据。展示层包括界面布局、组件、样式、js逻辑代码，为了更加方便快捷地进行系统开发，前端采用Vue开发框架，具有开发简单、学习成本低、代码量少易于理解等优点，基于iView组件库结合开发UI界面，iView作为一个功能非常丰富的组件，它拥有漂亮的UI，丰富的API和超高的代码质量。前端的工作流程主要由使用者触发Vue组件按钮，按钮执行相应的Vue函数，Vue函数进行简单逻辑代码处理后，调用相应的API请求后端数据，后端根据前端的请求数据进行逻辑处理后，将数据响应给前端，前端再进行数据展示。

### 5.1.2 控制层

控制层的职能是负责读取视图展示层的数据，控制数据的请求规范，将不规范的数据进行拦截，返回错误提示给后端，当请求数据符合规范时，控制层映射请求地址，将请求传输到业务层进行逻辑处理，然后将业务层返回的数据传输到前端页面。控制层使用REST风格作为请求规范，REST风格简化了书写和隐藏资源访问，用户不能通过请求地址知道相应资源的操作。解决了传统url请求的复杂度，同时也提高了系统的安全性。本系统也将进行自定义注解的开发，可使控制层无需过多编写烦杂的代码。也定义了前后端交互VO，使用统一的数据返回格式，有助于前端更好地处理数据。

### 5.1.3 业务层

业务层主要负者处理控制器传递的请求，再根据业务需求编码实现，为了方便业务层逻辑代码的处理，统一将业务层代码封装为一个模块，控制层按照业务需求调用不同业务层。业务层划分为业务接口类和业务接口实现类两个层次，其中业务接口类可对特定业务的实现方法进行编写，同时也继承了Mybatis-plus内置的IService省去了简单的业务方法编写。其次使用实现类去实现业务接口类的具体方法，实现类同时也继承了Mybatis-plus内置的ServiceImpl类，可以实现与较为繁杂的数据层交互。

### 5.1.4 数据层

数据层负者和数据库交互，按照业务层逻辑执行数据CRUD，并通过Mybatis-plus执行MySQL数据库操作，数据层则继承了BaseMapper，实现了简便的数据库操作，复杂的操作语句亦可以通过@注解方式进行SQL语句编写，大大提升了数据的操作效率。为了缓解热点数据的高访问，系统还加入了Redis、MongoDB作为数据的缓存容器。

## 5.2设备部署模块

### 5.2.1 模块分析

设备部署模块是管理自动售货机的核心内容之一，系统将严格管控和审核管理员对设备的部署，根据系统需求分析可得出设备部署详细设计步骤，首先管理员通过浏览器访问系统管理后台网址，根据功能模块选择添加设备，根据系统页面提示信息下拉选择设备分组名称，如果系统不存在设备分组返回上一步，存在则配置设备地理位置信息，通过高德地图获取设备定位，获取定位失败返回上一步；成功则管理员提交部署申请，系统判断是否符合部署标准，符合提交成功，管理员选择配置货道商品，系统存在对应商品信息，配置成功，不存在返回上一步，若管理员不提交部署申请则直接结束。设备部署模块的流程图如图5-2所示。

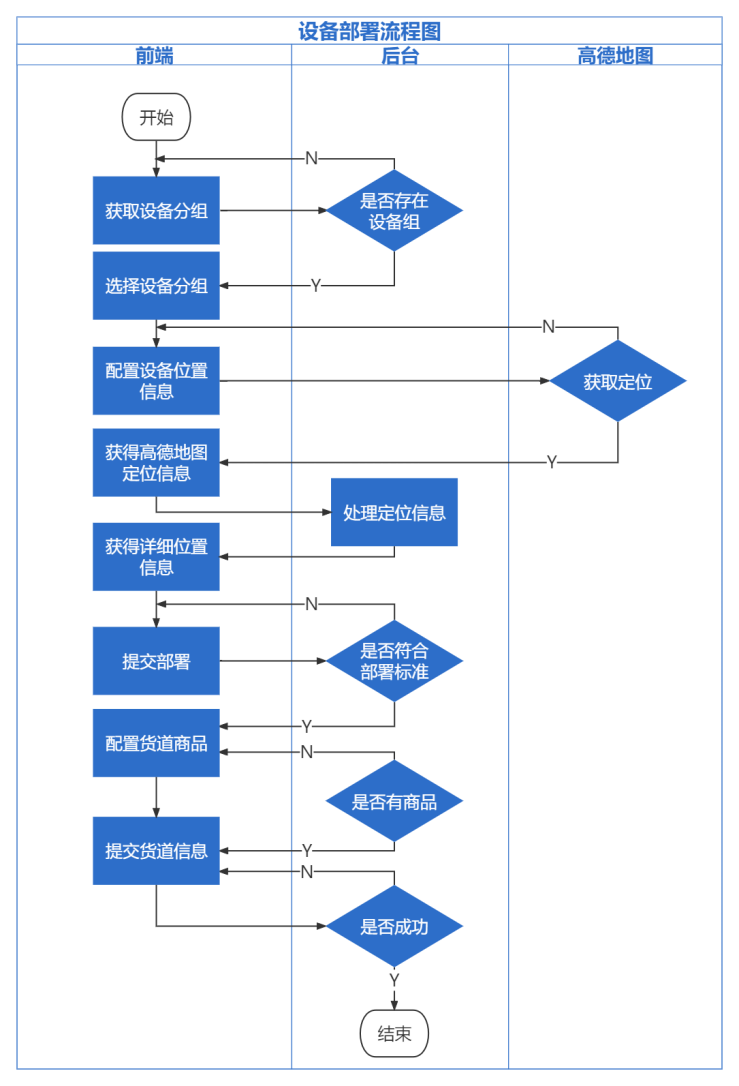


图5-2 设备部署流程图

### 5.2.2 模块设计

设备部署模块中的“选择设备分组”功能由控制器MachineGroupManagerController类的getByPage()方法、业务层MachineGroupService实现类的getMachineGroup()方法、数据层MachineGroupMapper pageByMachineGroup()方法组合完成。如图5-3所示。

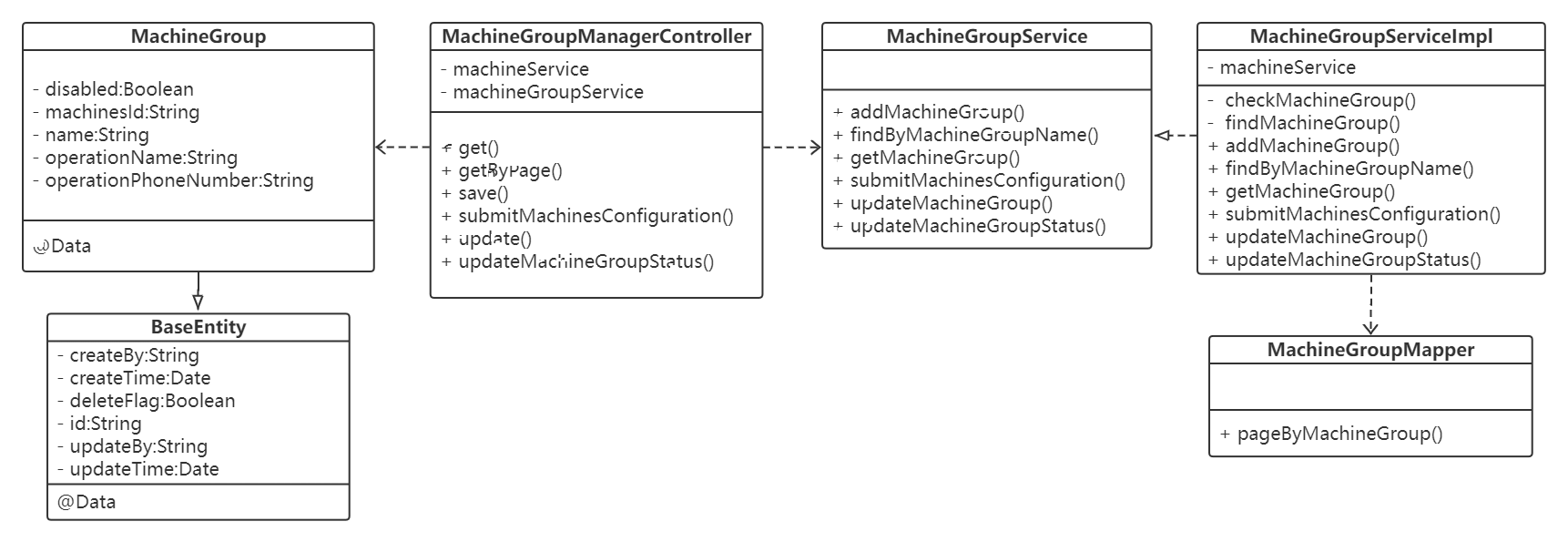


图5-3 设备部署（MachineGroup）类图

设备部署模块中“获取定位”功能通过高德地图amap-jsapi-loader API获取设备部署位置点，传递位置点信息的 城市代码和镇、街道名称由控制器RegionController类的getRegion()方法，触发业务层RegionService实现类的getRegion()方法，该方法根据城市代码和镇、街道名称调用数据层RegionMapper，获取地址id、地址名称并根据地址层级分段封装数据并回传数据。如图5-4所示。



图5-4 设备部署（Region）类图

设备部署模块中“提交部署”功能由控制器MachineManagerController类的save()方法验证是否符合设备部署标准，符合标准调用业务层machineService实现类的getMachine()方法，再传递数据层MachineMapper保存数据信息。如图5-5所示。

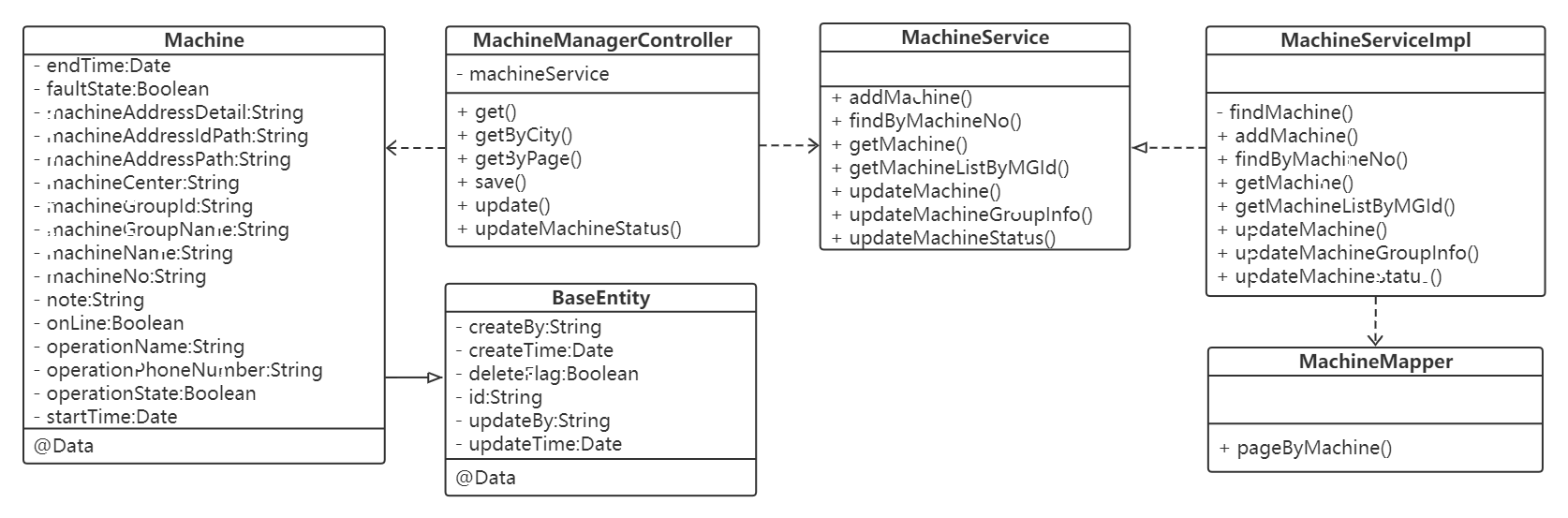


图5-5 设备部署（Machine）类图

设备部署模块中“配置货道商品”功能主要由控制器MachineLineProductController类、业务层MachineLineProductService实现类、数据层MachineLineProductMapper组合完成。如图5-6所示。

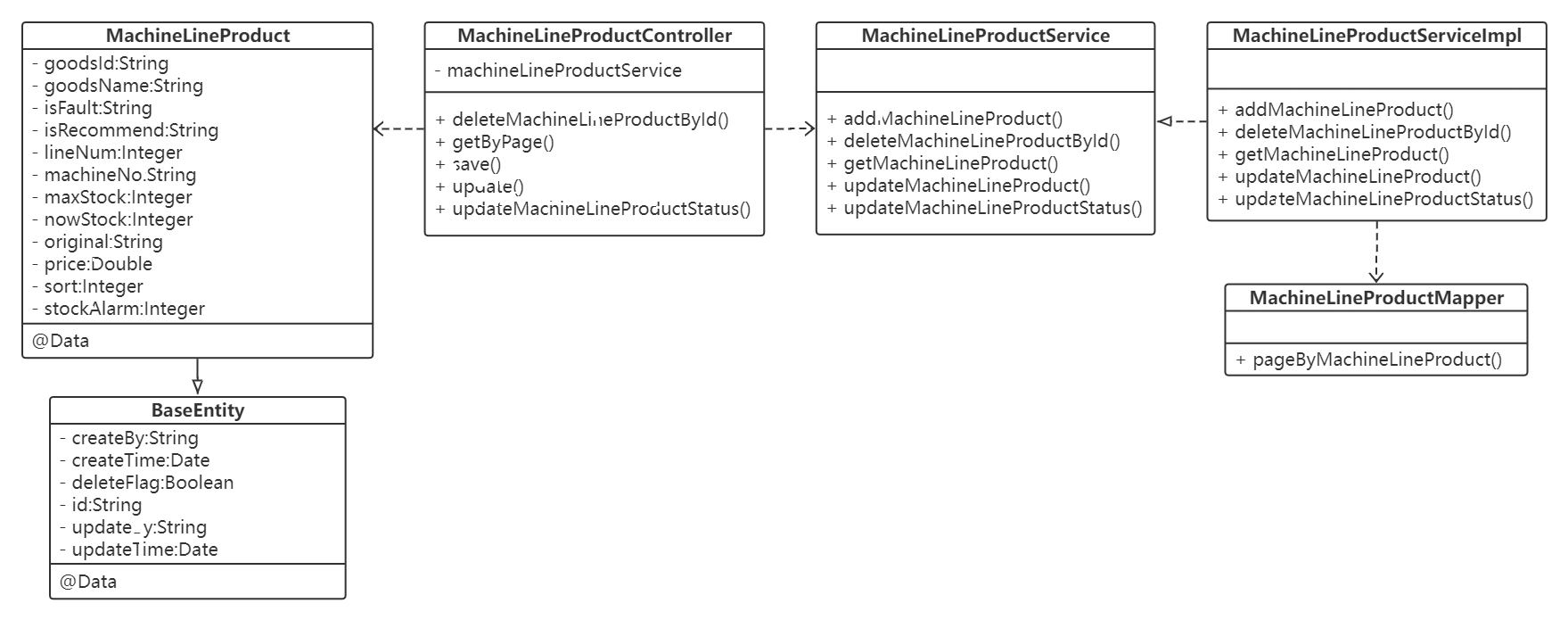


图5-6 设备部署（MachineLineProduct）类图

### 5.2.3 核心实现过程

1．新建名为Machine实体类，该类需继承BaseEntity类，在该类中编写设备名称、设备编码、详细地址等属性，设置为私有访问修饰符，并且在实体类上方使用@Data、@TableName("li\_machine")、@NoArgsConstructor注解；

2．新建名为MachineMapper接口类，继承MyBatisPlus的BaseMapper接口类，实现基本的增删改查功能，编写pageByMachine方法实现操作数据库返回page页面类型数据；

3．新建名为MachineService接口类，继承MyBatisPlus的IService接口类，实现基本的业务逻辑方法，并在此类中编写通过名称获取设备、通过城市获取设备列表、获取所有设备列表等业务方法；

4．新建名为MachineServiceImpl实体类，该类需继承MachineService类，并实现MachineService接口类的所有方法；

5．新建控制类，命名为MachineManagerController。声明MachineService对象，并编写getByPage、save、update方法，使用这些方法调用service层相应业务。

设备部署核心代码如下：

|  |
| --- |
| //提交部署  @Override  public Machine addMachine(MachineAddDTO machineAddDTO) {  //添加设备  Machine machine = new Machine(  machineAddDTO.getMachineGroupName(),  略...  );  machine.setId(SnowFlake.getIdStr());  //保存设备  this.save(machine);  return machine;  }  //配置货道商品  @Override  public MachineLineProductadd MachineLineProduct(MachineLineProduct machineLineProductAddDTO) {  MachineLineProduct machineLineProduct = new MachineLineProduct();  machineLineProduct.setMachineNo(machineLineProductAddDTO.getMachineNo());  略...  );  //保存设备货道商品  this.save(machineLineProduct);  return machineLineProduct;  } |

### 5.2.4 实现结果

管理员选择设备分组效果图如图5-7所示，获取定位效果图如图5-8所示，部署设备信息界面如图5-9所示，配置货道商品如图5-10所示。

图5-7 选择设备分组界面 图5-8 获取设备位置界面

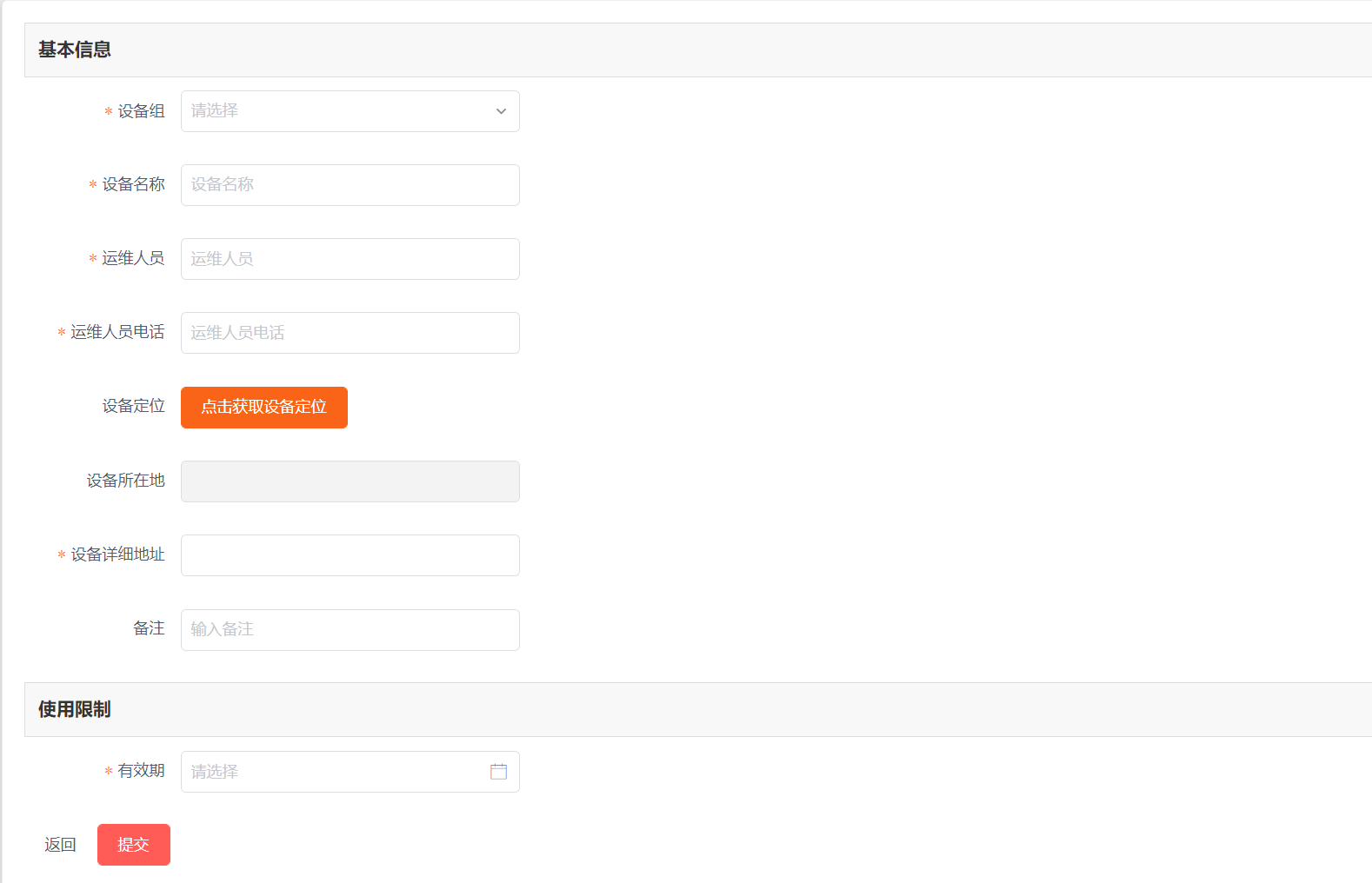


图5-9 部署设备信息界面



图5-10 配置货道商品界面

## 5.3商品预定模块

### 5.3.1 模块分析

商品预定模块是自动售货机线上销售的核心功能之一，亦是消费者购物方式中的重要渠道，根据需求分析可得出商品预定模块详细设计步骤，主要由微信小程序和系统后台完成这项任务。首先消费者登录系统微信小程序，微信小程序首页提供预定功能选项，当消费者选择预定功能，微信小程序显示系统默认选择的设备，并展示该设备的货道商品。当消费者通过搜索选择其它设备时，系统后台判断设备是否存在，存在则提供该设备货道商品信息，不存在返回上一步。消费者可通过加入购物车或者立即预定选项方式选购商品，再进行订单填写，填写后若消费者不支付订单则流程结束，当消费者选择微信支付支付订单，系统后台处理支付业务，再调用微信支付API获取支付信息，消费者不确认支付则流程结束，当消费者输入密码支付，微信支付判断是否支付成功，支付失败流程结束，支付成功由系统生成订单、取货码信息，消费者查看订单、取货码信息流程结束。商品预定模块的流程图如图5-11所示。

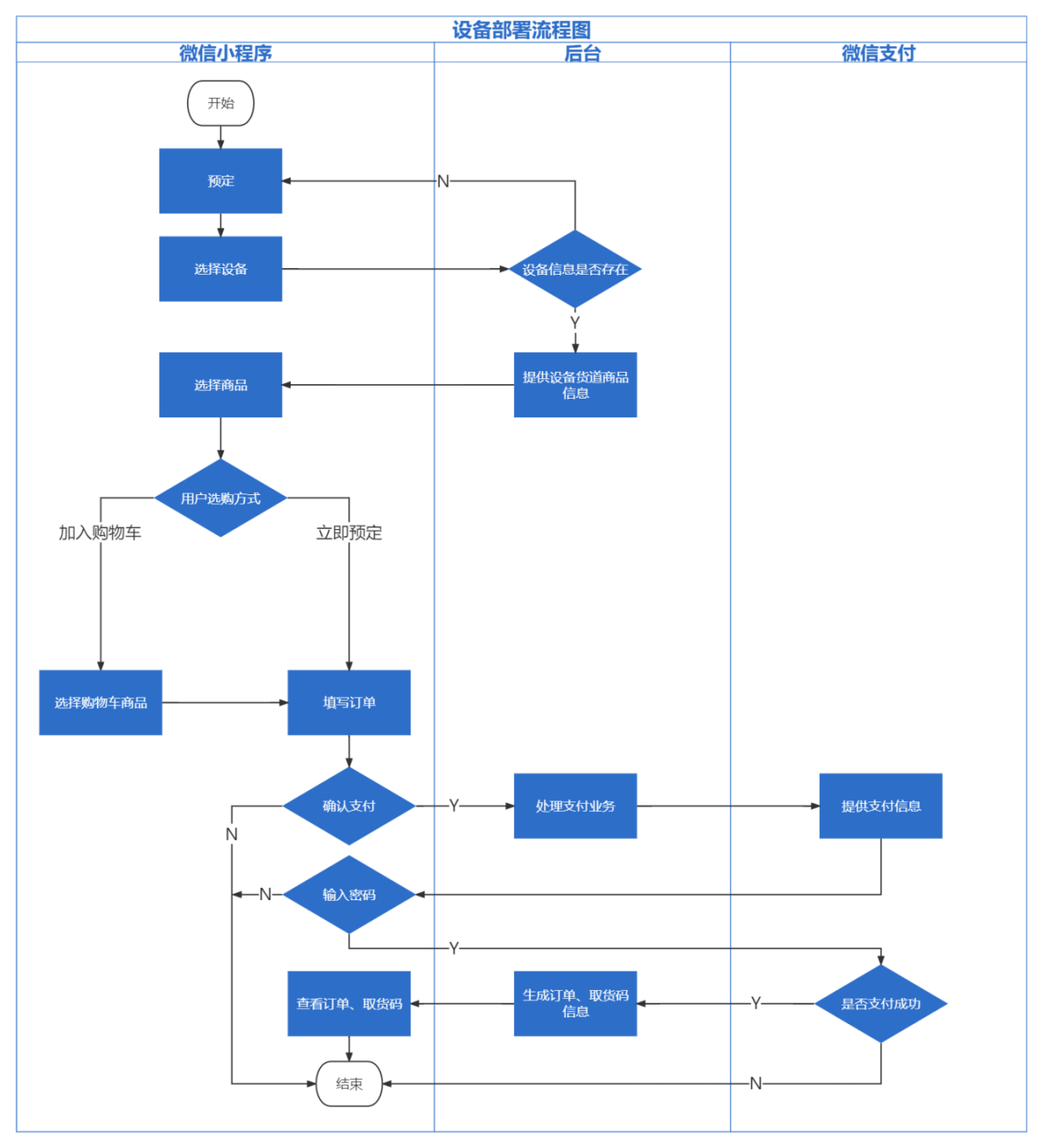


图5-11 商品预定流程图

### 5.3.2 模块设计

商品预定模块通过微信小程序调用后端接口，由控制层调用业务层再调用数据层获取设备默认数据；当用户点击选择设备时通过微信小程序调用后端接口，由控制层调用业务层再调用数据层获取设备信息；用户支付订单通过微信小程序调用后端接口，由控制层调用业务层再调用数据层，往数据库写入数据，返回成功结果；最后用户付款通过微信小程序调用后端接口，由控制层调用业务层，业务层调用微信支付接口并生成取货码，然后调用数据层更新订单信息，返回订单、取货码等信息。商品预定序列图如图5-12所示。

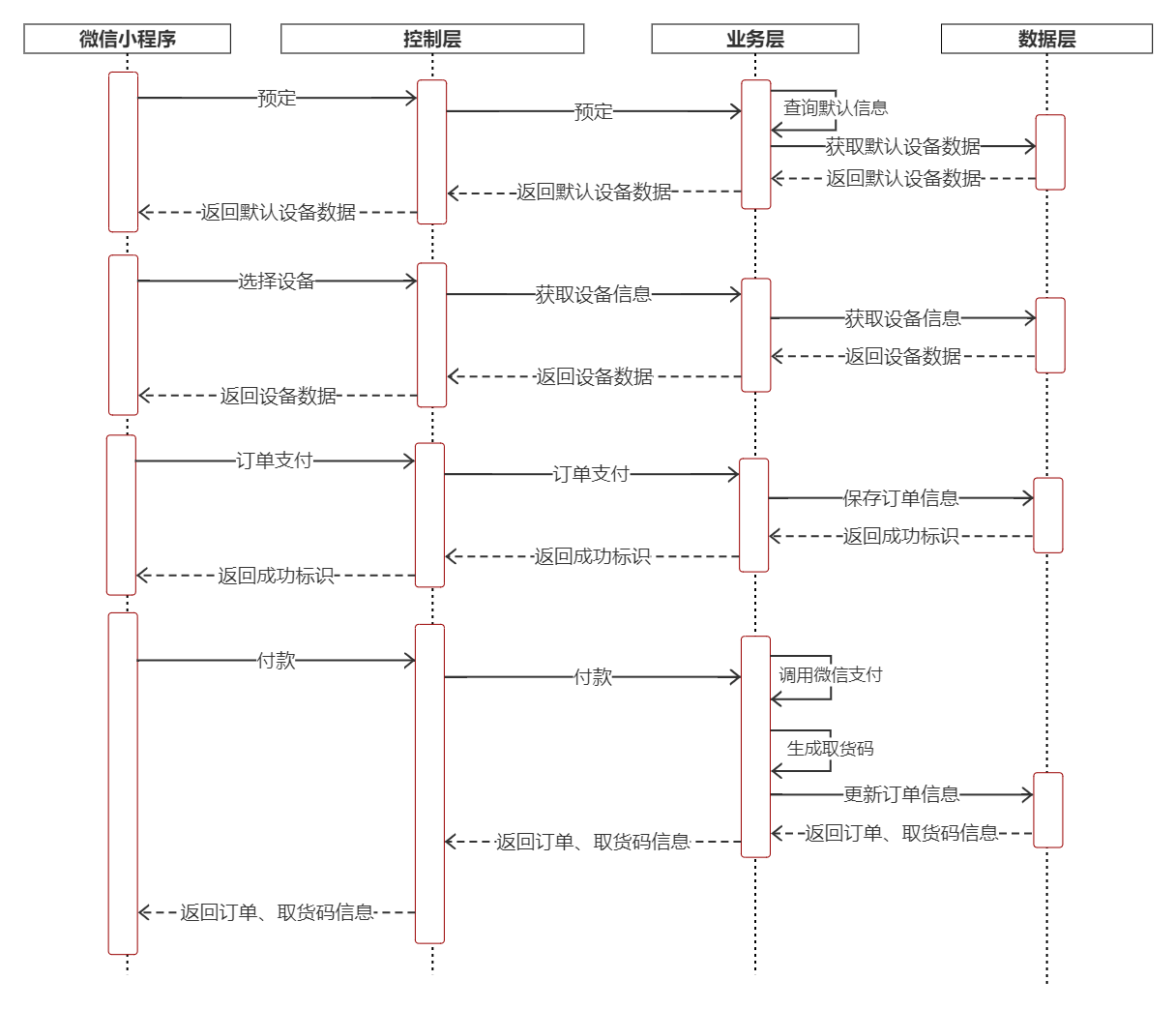


图5-12 商品预定序列图

### 5.3.3 核心实现过程

1．新建名为TradeParams实体类，在该类中编写设备编码、设备货道数、取货码等属性，设置为私有访问修饰符，并且在实体类上方使用@Data注解；

2．新建名为TradeMapper接口类，继承MyBatisPlus的BaseMapper接口类，实现基本的增删改查功能，编写updateTradePrice方法实现操作数据库修改交易金额；

3．新建名为TradeService接口类，继承MyBatisPlus的IService接口类，实现基本的业务逻辑方法，并在此类中编写创建交易、获取交易详情、整笔交易付款等业务方法；

4．新建名为TradeServiceImpl实体类，该类需继承TradeService类，并实现TradeService接口类的所有方法；

5．新建控制类，命名为CartController。声明TradeService对象，并编写add、cartAll、crateTrade等方法，使用这些方法调用service层的相应业务。

商品预定模块业务层核心代码实现如下：

|  |
| --- |
| //提交订单  @Override  public Trade createTrade(TradeParams tradeParams) {  //获取购物车  CartTypeEnum cartTypeEnum = getCartType(tradeParams.getWay());  TradeDTO tradeDTO = this.readDTO(cartTypeEnum);  //设置基础属性  tradeDTO.setClientType(tradeParams.getClient());  tradeDTO.setStoreRemark(tradeParams.getRemark());  tradeDTO.setParentOrderSn(tradeParams.getParentOrderSn());  tradeDTO.setMachineNo(tradeParams.getMachineNo());  tradeDTO.setLineNum(tradeParams.getLineNum());  if (tradeParams.getReservation() != null) {  if (tradeParams.getReservation()) { //直接购买，生成取货码  String snowflake = SnowFlake.getIdStr();  String pickUpCode = snowflake.substring(snowflake.length() - 6);  tradeDTO.setPickUpCode(pickUpCode);  }  }  //订单无收货地址校验  if (tradeDTO.getMemberAddress() == null) {  throw new ServiceException(ResultCode.MEMBER\_ADDRESS\_NOT\_EXIST);  }  //构建交易  Trade trade = tradeBuilder.createTrade(tradeDTO);  this.cleanChecked(this.readDTO(cartTypeEnum));  return trade;  } |

### 5.3.4 实现结果

消费者登录微信小程序首先将显示“预定”选项，如图5-13（a）所示。预定页面，消费者可根据城市名称或设备名称搜索设备，如图5-13（b）所示。选定某个设备将显示该设备商品列表，如图5-13（c）所示。当消费者点击商品时，展示该商品的详情信息，如图5-13（d）所示。当消费者选择“立即预定”或从购物车结算商品后显示填写订单页面，如图5-13（e）所示。用户支付该订单成功页面，如图5-13（f）所示。

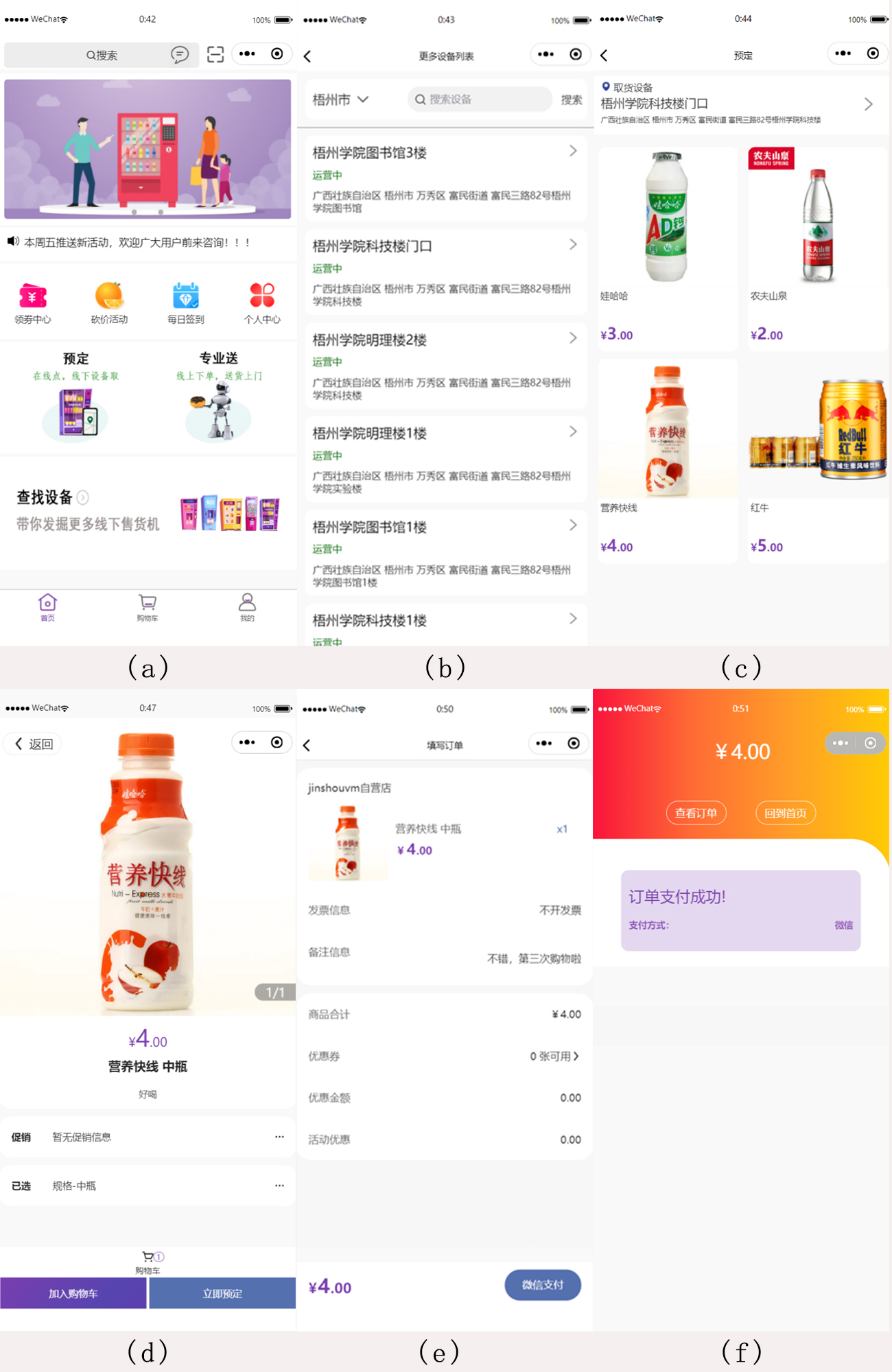


图5-13商品预定界面

## 5.4设备搜索模块

### 5.4.1 模块分析

设备搜索模块是消费者获取购物途径的核心功能之一，消费者可根据微信小程序“查找设备”功能，通过当前城市或设备名称搜索，以地图标点方式显示当前设备具体位置，消费者可根据位置信息进行线下购物。根据需求分析可得设备搜索模块详细设计步骤，主要由微信小程序和系统后台完成这项工作。首先消费者登录系统微信小程序，微信小程序首页提供查找设备功能选项，当消费者选择查找设备功能，微信小程序通过调用后台接口获取当前城市内的设备信息，消费者亦可通过其它城市名称或设备名称搜索设备，当消费者选定设备将通过微信小程序map组件，由腾讯地图显示设备位置信息。设备搜索模块的流程图如图5-14所示。

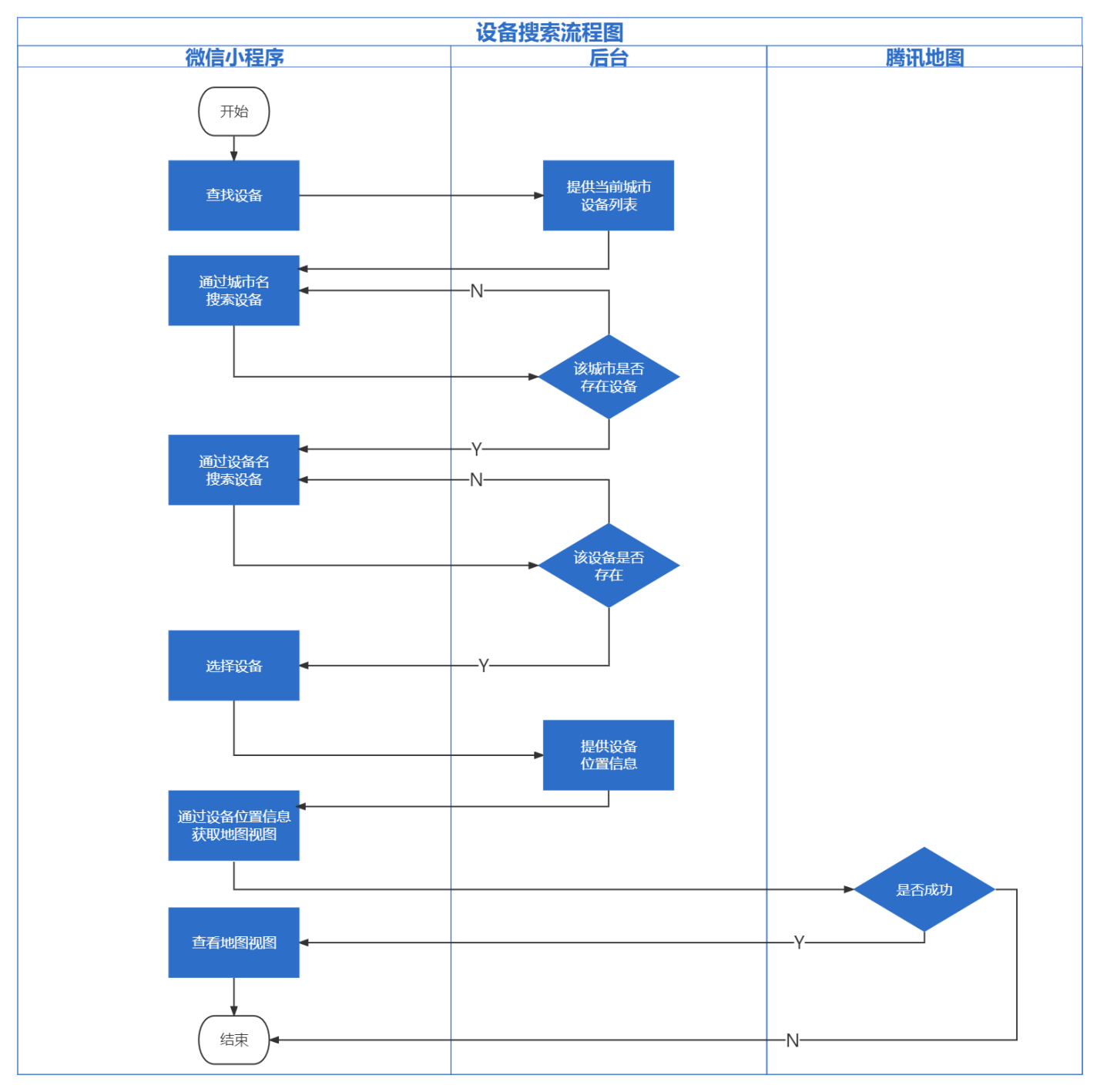


图5-14设备搜索流程图

### 5.4.2 模块设计

设备搜索模块中“通过城市名搜索设备”功能由控制器MachineManagerController类的getByCity()方法，调用业务层machineService实现类的getMachine(String city)方法处理业务按城市名称筛选后，调用数据层MachineMapper selectList()方法获取数据并返回。“通过设备名搜索设备”由控制器MachineManagerController类的getByPage()方法，调用业务层machineService实现类的getMachine(MachineSearchVO machineSearchVO, PageVO page)方法处理设备名称查询条件后，调用数据层MachineMapper PageByMachine()方法获取数据库数据并返回。设备搜索类图如图5-15所示。

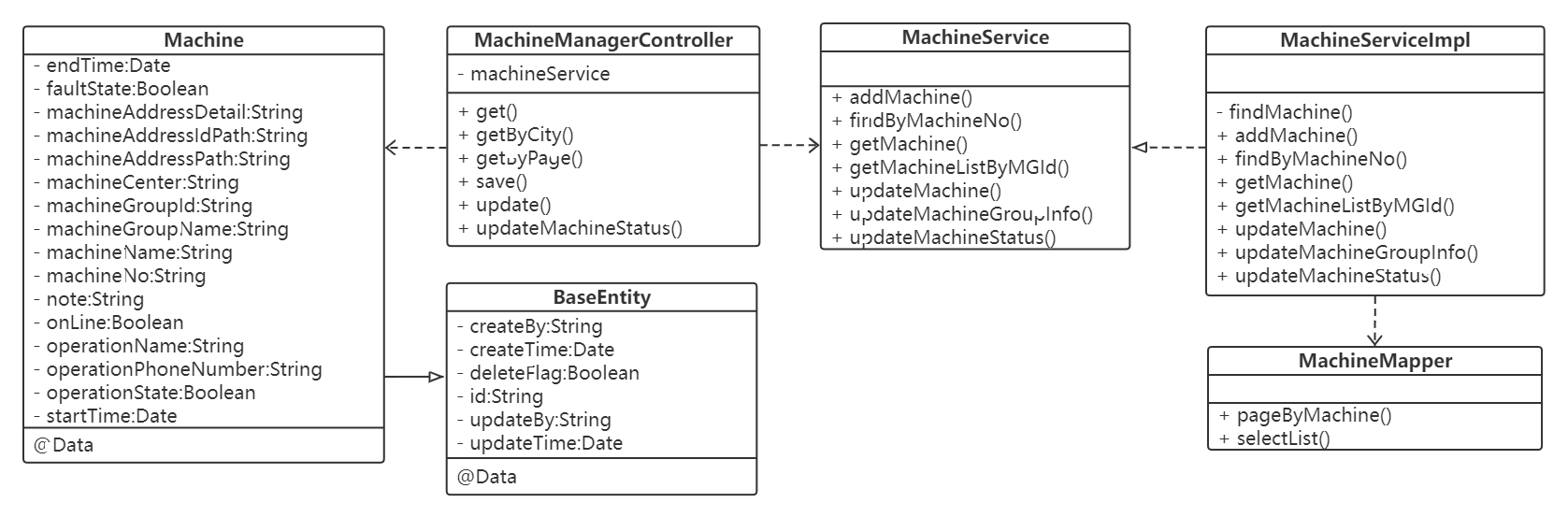


图5-15设备搜索类图

### 5.4.3 核心实现过程

在5.2.3的基础上给MachineService接口类增加findByMachineNo(String machineNo)、getMachine(String city)、getMachineList()等方法，实现设备搜索业务；

给MachineManagerController类增加getByCity(String city)、get(@PathVariable String machineNo)方法，使用这些方法调用service层相应业务。

设备搜索模块业务层核心代码实现如下：

|  |
| --- |
| // 通过城市名搜索设备  @Override  public List<Machine> getMachine(String city) {  QueryWrapper<Machine> queryWrapper = Wrappers.query();  //城市名称查询  queryWrapper.like(CharSequenceUtil.isNotBlank(city), "machine\_address\_path", city);  queryWrapper.orderByDesc("create\_time");  return this.baseMapper.selectList(queryWrapper);  }  // 通过设备名搜索设备  @Override  public IPage<Machine> getMachine(MachineSearchVO machineSearchVO, PageVO page) {  QueryWrapper<Machine> queryWrapper = Wrappers.query();  //设备编号查询  queryWrapper.eq(CharSequenceUtil.isNotBlank(machineSearchVO.getMachineNo()), "machine\_no", machineSearchVO.getMachineNo());  //设备名称查询  queryWrapper.like(CharSequenceUtil.isNotBlank(machineSearchVO.getMachineName()), "machine\_name", machineSearchVO.getMachineName());  略...  queryWrapper.orderByDesc("create\_time");  return this.baseMapper.pageByMachine(PageUtil.initPage(page), queryWrapper);  } |

### 5.4.4 实现结果

消费者登录微信小程序首页将显示“查找设备”选项，如图5-16（a）所示。消费者点击“查找设备”按钮将显示当前城市的设备列表信息，如图5-16（b）所示。同时消费者亦可按不同城市搜索设备或按设备名称搜索设备，如图5-16（c）、5-17（a）所示。当消费者选择某个设备时，将显示该设备地理位置信息，如图5-17（b）所示。

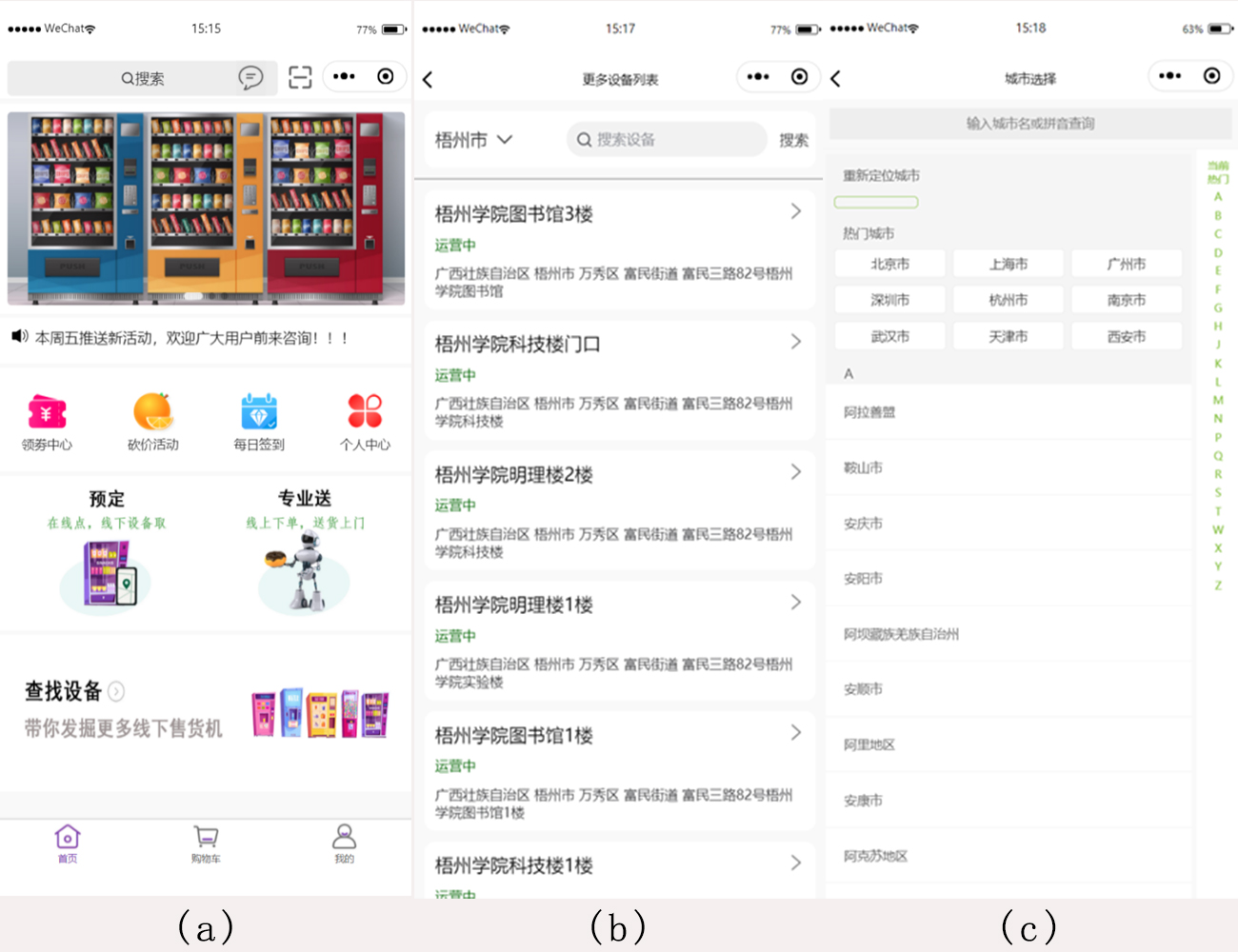


图5-16设备搜索界面（一）

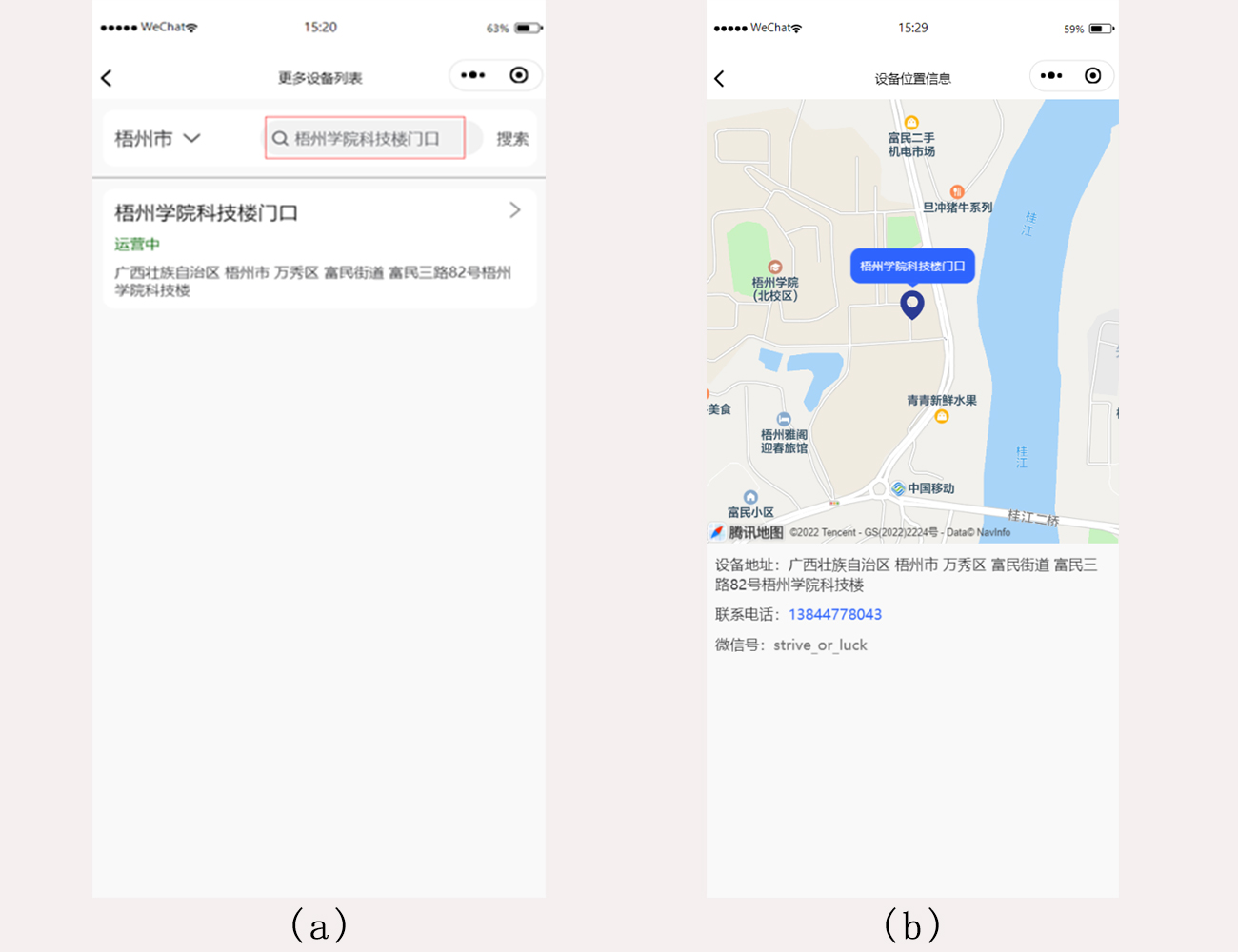


图5-17设备搜索界面（二）

## 5.5 本章小结

本章首先论述了基于SpringBoot和Vue框架的自动售货机管理系统的主体架构细化实现，概述了本系统分层架构设计原理和重要性，以及每个层涉及的技术原理和用处；接着详细论述了系统中三个核心模块设计与实现的思路和方法，对每个模块通过流程图、类图、序列图和效果截图等辅助说明设计与实现的效果，最后展示了实现功能的核心代码。

# 第六章 系统测试和验证

系统测试是整个软件开发周期的最后一个环节，有效的系统测试能够给软件开发人员提供分析特定测试结果和软件的功能，为改进和完善软件提供了基础。系统测试，描述用于促进身份验证软件的真实性，完整性，安全性和质量的过程[12]。系统测试在项目开发过程中十分重要，它可以帮助开发人员尽早的发现并解决被测对象相对于目标对象的缺陷，保证开发质量[13]。接下来，本章阐述系统的核心功能测试和性能测试。

## 6.1 系统测试概述

本次系统测试的环境信息如表6-1所示。

表 6-1 系统测试环境信息表

|  |  |  |
| --- | --- | --- |
| 系统测试环境信息 | 服务端操作系统 | Windows 11 家庭中文版 |
| 服务端硬件信息 | 12th Gen Intel(R) Core(TM) i5-12500H 2.50 GHz、16.0 GB RAM（内存） |
| 服务端软件环境 | JDK11 |
| Tomcat9.0.65 |
| Maven3.6.3 |
| MySQL8.0.30 |
| Redis6.2.6 |
| elasticsearch8.5.0 |
| 客户端浏览器 | 谷歌浏览器Google Chrome、  微软Microsoft Edge |
| 微信小程序测试环境 | 操作系统 | Android |
| 设备名称 | Redmi K30 Pro |
| 系统版本 | MIUI 13.0.5 |
| 安卓版本 | 12 |
| 处理器 | 高通骁龙865 八核 2.84 GHz |
| 微信版本 | 8.0.28 |

## 6.2 系统功能测试

功能测试主要用来验证一个项目的功能需求是否达到用户提出的预期标准[14]，功能测试是保证系统的功能正常运行必须的一步，同时也是很重要的一步[15]。测试前要设置测试用例，测试用例的设置必须严谨完整，然后逐一测试，检查项目是否满足用户的需求[16]。

### 6.2.1 微信小程序功能模块测试用例

微信小程序功能提供用户线上查看售货机商品、位置等信息，同时也可以管理自己的数据信息，获取系统最新的公告、打折活动等，还可以查看个人信息、设备列表、设备地理位置、设备商品、购物车、公告、优惠券等。该功能测试用例如表6-2所示。

表 6-2 微信小程序模块测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 测试模块 | 测试步骤 | 预期结果 | 实际结果 |
| 使用微信授权登录 | 打开微信小程序，点击“使用微信授权”按钮 | 登录成功，进入微信小程序首页 | 测试成功 |
| 进入个人中心 | 登录微信小程序，点击底部导航栏“我的”按钮 | 进入个人信息界面 | 测试成功 |
| 进入购物车 | 登录微信小程序，点击底部导航栏“购物车”按钮 | 进入购物车界面 | 测试成功 |
| 查看设备列表 | 登录微信小程序，在首页点击“查找设备”按钮 | 进入设备列表页面，并显示设备 | 测试成功 |
| 查看设备地理位置 | 登录微信小程序，在首页点击“查找设备”按钮，进入设备列表页面，选择相应设备 | 显示设备地理位置信息 | 测试成功 |
| 查看设备商品 | 登录微信小程序，在首页点击“预定”按钮 | 进入设备预定页面，并显示默认设备商品 | 测试成功 |
| 查看公告 | 登录微信小程序，进入首页 | 首页显示公告信息 | 测试成功 |
| 查看优惠券 | 登录微信小程序，进入首页点击“领劵中心”按钮 | 进入领取中心界面 | 测试成功 |

### 6.2.2 设备部署功能模块测试用例

设备部署功能主要由管理员对设备进行添加以及对设备进行货道商品配置操作，由管理员进入系统后台，对设备进行添加，添加成功后，对设备货道进行配置商品，用户通过微信小程序可查看设备以及对应的商品信息。设备部署功能测试用例如表6-3所示。

表 6-3 设备部署功能模块测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 设备部署 | | |
| 用例说明 | 管理员对设备进行部署及设备货道商品配置内容测试 | | |
| 测试目的 | 测试管理员是否成功部署设备及商品配置 | | |
| 前置条件 | 管理员已经成功登录系统后台 | | |
| 操作步骤 | 1. 管理员点击菜单栏中的【设备列表】 2. 进入设备列表页面 3. 点击【添加设备】按钮 4. 选择设备分组 5. 填写设备名称、运维人员姓名和电话号码 6. 选择设备定位 7. 点击【点击获取设备定位】按钮 8. 添加备注 9. 选择设备有效期 10. 点击【提交】按钮 11. 系统更改部署内容数据，并显示已添加设备内容列表 12. 管理员选择新增的设备点击【货道商品】按钮 13. 进入货道商品配置页面 14. 点击【添加货道商品】按钮 15. 选择商品名称 16. 输入货道编号、最大库存数量、当前库存数量、库存告警数量 17. 点击【确定】按钮 18. 系统提示添加设备商品成功，并显示已添加的商品信息 | | |
| 序号 | 操作 | 预期结果 | 实际结果 |
| 1 | 点击【设备列表】 | 显示设备列表信息 | 显示设备列表信息 |
| 2 | 点击【添加设备】 | 跳转添加设备页面 | 跳转添加设备页面 |
| 3 | 选择设备分组 | 弹出设备分组列表 | 弹出设备分组列表 |
| 4 | 填写设备名称、运维人员姓名和电话号码 | 当输入信息为空提示输入当输入信息有误提示修正 | 当输入信息为空提示输入，当输入信息有误提示修正 |
| 5 | 点击【点击获取设备定位】 | 弹出高德地图位置选择器 | 弹出高德地图位置选择器 |
| 6 | 选择设备有效期 | 弹出日期时间选择器 | 弹出日期时间选择器 |
| 7 | 点击【提交】 | 显示添加设备成功，并在设备列表页面显示 | 显示添加设备成功，并在设备列表页面显示 |
| 8 | 点击【货道商品】 | 跳转设备货道商品页面 | 跳转设备货道商品页面 |
| 9 | 点击【添加货道商品】 | 弹出添加设备货道商品弹窗 | 弹出添加设备货道商品弹窗 |
| 10 | 选择商品名称 | 弹出商品列表 | 弹出商品列表 |
| 11 | 输入货道编号、最大库存数量、当前库存数量、库存告警数量 | 当输入信息为空提示输入当输入信息有误提示修正 | 当输入信息为空提示输入，当输入信息有误提示修正 |
| 12 | 点击【确定】 | 显示添加设备商品成功，并在设备货道商品页面显示 | 显示添加设备商品成功，并在设备货道商品页面显示 |

### 6.2.3 商品预定功能模块测试用例

商品预定功能由用户通过微信小程序根据设备查看商品，选择商品下单预定，填写订单，支付，完成后可以查看订单、取货码等信息。商品预定功能测试用例如表6-4所示。

表 6-4 商品预定功能模块测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 商品预定 | | |
| 用例说明 | 用户在微信小程序对某个设备的商品预定内容测试 | | |
| 测试目的 | 测试用户是否成功预定商品及查看订单 | | |
| 前置条件 | 用户已经成功登录微信小程序 | | |
| 操作步骤 | 1. 用户在首页点击【预定】按钮 2. 进入默认设备商品信息页面 3. 点击取货设备栏目 4. 进入更多设备列表页面 5. 用户通过城市选项选择“梧州市” 6. 系统显示梧州市内的设备列表 7. 用户通过输入框输入“梧州学院科技楼门口” 8. 系统显示该设备信息 9. 点击【梧州学院科技楼门口】设备项 10. 跳转该设备商品信息页面 11. 点击【营养快线】商品 12. 进入营养快线商品详情页面 13. 点击【加入购物车】按钮 14. 系统提示商品已添加到购物车 15. 点击【立即预定】按钮 16. 进入填写订单页面 17. 点击【微信支付】按钮 18. 弹出支付弹窗 19. 输入密码支付 20. 跳转订单支付成功页面 21. 点击【查看订单】按钮 22. 显示订单列表页面 23. 点击【取货码】按钮 24. 弹出取货码信息弹窗 25. 点击订单选项 26. 进入订单详情页面 | | |
| 序号 | 操作 | 预期结果 | 实际结果 |
| 1 | 点击【预定】按钮 | 进入默认设备商品信息页面 | 进入默认设备商品信息页面 |
| 2 | 点击取货设备栏目 | 进入更多设备列表页面 | 进入更多设备列表页面 |
| 3 | 通过城市选项选择“梧州市” | 系统显示梧州市内的设备列表 | 系统显示梧州市内的设备列表 |
| 4 | 通过输入框输入“梧州学院科技楼门口” | 显示设备名称为“梧州学院科技楼门口”的设备信息 | 显示设备名称为“梧州学院科技楼门口”的设备信息 |
| 5 | 点击【梧州学院科技楼门口】设备项 | 进入设备名为“梧州学院科技楼门口”的商品信息页面 | 进入设备名为“梧州学院科技楼门口”的商品信息页面 |
| 6 | 点击【营养快线】商品 | 进入营养快线商品详情页面 | 进入营养快线商品详情页面 |
| 7 | 点击【加入购物车】按钮 | 系统提示商品已添加到购物车 | 系统提示商品已添加到购物车 |
| 8 | 点击【立即预定】按钮 | 进入填写订单页面 | 进入填写订单页面 |
| 9 | 点击【微信支付】按钮 | 弹出支付弹窗 | 弹出支付弹窗 |
| 10 | 输入密码支付 | 跳转订单支付成功页面 | 跳转订单支付成功页面 |
| 11 | 点击【查看订单】按钮 | 显示订单列表页面 | 显示订单列表页面 |
| 12 | 点击【取货码】按钮 | 弹出取货码信息弹窗 | 弹出取货码信息弹窗 |
| 13 | 点击订单选项 | 进入订单详情页面 | 进入订单详情页面 |

### 6.2.4 设备搜索功能模块测试用例

设备搜索功能由用户通过微信小程序查看自动售货机设备的地理位置信息，方便用户寻找附近的自动售货机设备进行线下购物，也方便商品预定和线上下单就近选择自动售货机设备取物。设备搜索功能测试用例如表6-5所示。

表 6-5 设备搜索功能模块测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 设备搜索 | | |
| 用例说明 | 用户在微信小程序对自动售货机设备搜索及查看信息内容测试 | | |
| 测试目的 | 测试用户是否成功搜索自动售货机设备及查看信息 | | |
| 前置条件 | 用户已经成功登录微信小程序 | | |
| 操作步骤 | 1. 用户在首页点击【查找设备】按钮 2. 进入设备列表页面，并显示当前所在城市的设备列表信息 3. 用户通过城市选项选择“广州市” 4. 系统显示广州市内的设备列表 5. 用户通过输入框输入“梧州学院图书馆1楼” 6. 系统显示该设备信息 7. 点击【梧州学院图书馆1楼】设备项 8. 进入该设备位置信息页面 | | |
| 序号 | 操作 | 预期结果 | 实际结果 |
| 1 | 点击【查找设备】按钮 | 进入设备列表页面，并显示当前所在城市的设备列表信息 | 进入设备列表页面，并显示当前所在城市的设备列表信息 |
| 2 | 用户通过城市选项选择“广州市” | 系统显示广州市内的设备列表 | 系统显示广州市内的设备列表 |
| 3 | 用户通过输入框输入“梧州学院图书馆1楼” | 系统显示设备名称为“梧州学院图书馆1楼”的设备信息 | 系统显示设备名称为“梧州学院图书馆1楼”的设备信息 |
| 4 | 点击【梧州学院图书馆1楼】设备项 | 进入设备名称为“梧州学院图书馆1楼”的设备位置信息页面 | 进入设备名称为“梧州学院图书馆1楼”的设备位置信息页面 |

## 6.3 系统性能测试

系统性能测试就是建立性能测试模型并以系统的性能指标为目标，制订性能测试计划。通过测试不同场景下的系统性能，不断探索系统设计与资源消耗之间的平衡。要保证系统在生产环境正常稳定运行，系统应该能够承受一定的高并发量和较好的稳定性。为了模拟真实环境，系统需要进行API自动化测试，模拟系统高并发请求环境。通过工具虚拟化产生多次并发访问过程中记录性能瓶颈，负载容量及系统各指标性能变化。本系统的性能测试采用Apifox接口测试工具进行API自动测试。负载测试目的及测试方案如表6-6所示。

表 6-6 系统性能测试方案

|  |  |
| --- | --- |
| 操作系统 | Windows 11 家庭中文版 |
| 硬件环境 | 12th Gen Intel(R) Core(TM) i5-12500H 2.50 GHz、16.0 GB RAM（内存） |
| 测试工具 | Apifox |
| 测试目的 | 测试服务器并发用户的承载能力 |
| 测试方案 | 1. 100并发量同时访问自动售货机管理系统网站，系统能够正常、平稳运行，且通过率要求为100%，平均访问耗时低于0.1秒。 2. 300并发量同时访问自动售货机管理系统网站，系统能够正常、平稳运行，且通过率要求为100%，平均访问耗时低于0.5秒。 |

使用Apifox 自动化测试模块，设置线程数100和300，循环次数1，间隔停顿0毫秒，分别对系统API进行测试，测试结果如图6-1、图6-2所示。根据测试结果得出，系统达到了性能测试方案的要求。



图6-1 100并发量测试结果图



图6-2 300并发量测试结果图

## 6.4 本章小结

本章首先对系统的测试做了概述，然后对移动端微信小程序功能模块，设备部署功能模块、商品预定功能模块和设备搜索功能模块进行用例测试。最后，还测试了系统的各项性能。基于测试结果，表明该系统达到了预期的效果。

# 第七章 总结与展望

## 7.1 总结

本次毕业设计主要围绕自动售货机企业、运营商管理和运营问题，为自动售货机企业、运营商搭建了一套自动售货机管理系统。文中介绍了系统的研究背景意义、开发技术和开发过程。

系统针对自动售货机在销售运营上的不足，使用移动端微信小程序拓展了用户使用范围，在微信小程序端开发了自动售货机设备搜索功能，通过微信小程序map组件以腾讯地图方式展示设备坐标，解决了用户在公共场所寻找设备难的痛点。为了拓展自动售货机销售运营业务，微信小程序端还开发了商品预定和线上购买商品功能，有效提升自动售货机的运营效率。系统管理后台开发采用了近些年比较流行的SpringBoot和Vue框架进行开发，使用MySQL数据库做数据存储，为了解决微信小程序购物车等高频数据操作，使用Redis作为高频数据缓存点，针对商品使用elasticsearch搜索引擎，对于管理系统的图片、视频使用阿里云OSS做为静态资源库，使用MyBatis-Plus做持久层框架，结合以上技术搭配其他主流开发插件，可使开发效率高，系统可扩展性强，经系统总体测试得出该系统功能完善，符合自动售货机管理和运行的标准。

## 7.2 展望

本课题的实现，目前可以满足自动售货机企业、运营商对设备管理和运营的需求，并且用户可以通过微信小程序对自动售货机设备进行搜索，商品预定、在线购物等，提升了自动售货机运营效率。由于个人能力以及时间限制的原因，本系统的设计依然存在有待完善的方面，比如系统的稳定性和安全性仍待优化，许多功能也没有完美实现，文中介绍的自动售货管理系统与实际应用版还存在很大的差距，在系统设计过程中，发现用户搜索设备功能仅对自动售货机做位置点展示，对于部分用户来说还不够直观。

我希望在以后的学习中能更好的完善此系统，针对该系统的后续发展规划，以及用户量的提升，需要解决系统的稳定性和安全性问题。对于设备搜索部分可使用地图导航形式寻找设备位置会更加高效。设计本系统的初衷是能够提高自动售货机管理的效率以及用户的参与性，针对本系统的未来发展，应结合其他系统进行优化，增加移动端与自动售货机交互性等，并在自动售货机管理系统方面一点点探索，希望能为无人零售行业的发展提供一点帮助。

# 参考文献

1. 王瑞珑.自动售货机的发展现状及趋势研究[J].现代职业教育,2021(36):224-225.
2. 高振,江若尘.消费情景、感知质量对产品溢价的影响——基于自动售货机业态的实证研究[J].北京工商大学学报(社会科学版),2020,35(06):15-27.
3. 张冀洲.基于游戏化理论的饮品自动售货机设计研究[D].东北石油大学,2022.
4. 张慧.数字经济背景下自动售货机发展影响因素及建议[J].时代经贸,2022,19(04):33-36.
5. 邓笑.基于Spring Boot的校园轻博客系统的设计与实现[D].华中科技大学,2018.
6. 张华兵等.基于NoSQL数据库的模型设计方法[J].电子技术与软件工程,2019(23):174-175.
7. 柴青山.基于MVVM模式的Vue.js框架在物流软件自动化测试系统中的应用研究[D].北京邮电大学,2019.
8. 王志任.基于Vue.js的开发平台的设计与实现[D].广东工业大学,2018.
9. 朱二华.基于Vue.js的Web前端应用研究[J].科技与创新,2017(20):119-121.
10. 李英玲等.基于uni-app+SpringBoot的移动智能办公系统设计与实现[J].西南民族大学学报(自然科学版),2022,48(03):313-321.
11. DCloud.功能框架[EB/OL].2022.https://uniapp.dcloud.io/?id=功能框架
12. 曹明昊.基于SpringBoot和Vue框架的邯郸市现代农业园区信息管理系统的研发[D].河北工程大学,2021.
13. 田海晴.基于SpringBoot和Vue框架的共享运营管理平台的设计与实现[D].山东大学,2020.
14. Esnaashari Mehdi,Damia Amir Hossein.Automation of software test data generation using genetic algorithm and reinforcement learning[J].Expert Systems With Applications,2021,183:1-3.
15. 高志平.基于SpringBoot框架与ITIL方法的运维管理系统的设计与实现[D].华东师范大学,2021.
16. FRANCESCO P,GAIA R,MASSIMO D I,et al.A Cooperation Project in Lesotho:Renewable Energy Potential Maps Embedded in a WebGIS Tool[J].Sustainability,2021,13(18):1-3.

# 致谢

时光飞逝，转眼间，二年的学习生涯即将落下帷幕。回顾这两年的点点滴滴，在我求学的道路上，有太多良师益友，他们在我孤独时给予我陪伴，在我遇到挫折时给予我鼓励，在我遇到困难时给予我帮助，与我一同度过了最难忘青春年华。

首先，我要感谢我的导师吴炎桃老师，在论文撰写的整个过程中，从论文选题、到撰写开题报告、最后到正文撰写，吴老师都提出了很多宝贵意见。吴老师指出的每一个问题，指导的每一个思路，都使我受到了很大的启发，给予我及时的帮助，使我能够顺利完成论文写作工作。

在此我要感谢梧州学院大数据与软件工程学院的所有老师，你们爱岗敬业的治学态度和那无私的奉献精神，不仅使我对专业知识上有了更进一步的理解，还促进我在德智体美劳方面均衡发展，将所学的专业知识和工作互相印证，受益匪浅。而且使我能够将所学理论应用于对现实问题的分析和解决，继而提高自己的专业水平。

感谢我的室友和同学，是你们的帮助、陪伴和支持让我感受到校园的温暖，在我的论文写作过程中，许多同学都给我带来了信息支持，在此一并表示感谢。最后感谢梧州学院为我带来宝贵的学习成长机会，使我能够走上新的台阶，开始一段新的人生！