智慧电子商城设计与实

关键词：微信平台；智能推荐的前台客户系统，数据可视化后台管理系统；辅助决策。

1. ：绪论

1.1 课题背景

移动电商、微信平台、社交电商

......

在国家大力推进互联网+的大背景下，随着科技变革、产业革命、消费升级，我国电商的发展步入了规模持续增长、业态不断创新结构日趋优化的新阶段。

1.2 研究现状和发展趋势

APP、WebApp、微信推广、社交电商

......

1.3 研究内容

系统整体架构、以模块为单位的微服务设计

系统与微信平台的信息交互

商品推荐、数据分析统计、可视化展现

......

1.4 论文组织结构

第二章：相关理论与技术。

本章节针对电商项目的特点，对项目的架构设计和技术选型进行详细阐述。后端项目的开发语言以kotlin为主、java8为辅；采用社区活跃度很高的SpringCloud微服务架构；数据库采用互联网行业最常用的mysql；前端框架采用当前比较流行的React框架；前端商城借助微信公众平台作为入口。

2.1 Spring Cloud微服务架构

在当下互联网大环境下，互联网产品的需求变化快，必然要求产品迭代更新也要快，然而现实情况是很多的产品都面临着迭代速度慢的问题。分析其中原因，就是随着产品功能的不断累积、变更，应用实现越来越复杂，代码规模也越来越大，开发团队工作在一个逻辑复杂、模块耦合度高的单块架构应用上，从而导致应用难于维护和更新。

2.1.1传统的单体架构

所谓单体应用就是项目所有的功能归档在一个项目中的应用。单体架构在小微企业比较常见，典型代表就是一个应用、一个数据库、一个web容器就可以运行起来。单体应用是最早的应用形态，不需要太关注整体性能，项目规模中小型时，开发和部署都相对方便。



  随着业务的不断扩大，需求功能的持续增加，单体架构已经很难满足业务快速变化的需要，容易造成如下问题，当项目规模变大时，代码容易产生冲突，造成版本管理困难；局部功能模块出现问题，可能会影响整体，因此稳定性也较差；在系统维护性和扩展性方面，随着项目规模的扩大，需求的不断增加，系统复杂性直线上升，容易造成系统不易理解、扩展需求的开发困难等问题；从技术选型和兼容性上来看，单一构架应用兼容的技术相对单一，初始的技术选型严重限制了其将来采用不同语言或者框架的能力，在一定程度上限制了项目技术的革新。

2.1.2微服务架构

从业界的讨论来看，微服务本身并没有一个严格的定义，这里借用ThoughtWorks的首席科学家——马丁·福勒（Martin Fowler）先生，对微服务的这段描述：

微服务架构是一种架构模式，它是提倡将单一应用程序划分成一组小的服务，服务之间互相协调、互相配合，为用户提供最终价值。每个服务运行在其独立的进程中，服务与服务件采用轻量级的通信机制进行沟通（通常采用HTTP的RESTful API）。每个服务都围绕着具体业务进行构建，并且能够被独立的部署到生产环境、类生产环境等。另外，应尽量避免统一的、集中式的服务管理机制，对具体的一个服务而言，应根据业务上下文，选择合适的语言、工具对其进行构建。



总结下来，微服务就是由多个以单一应用构成的小服务，自己拥有自己的进程与轻量化处理，服务依业务功能设计，以全自动的方式部署，并通过各自暴露的API接口相互通讯。同时服务会使用最小的规模的集中管理能力，服务可以用不同的编程语言与数据库等技术实现。

相比于传统的单体架构，微服务架构有很多优点。首先，通过将巨大单体应用分解为多个服务，解决了复杂性问题。在功能不变的情况下，应用被分解为多个可管理的分支或服务。每个服务都有一个用RPC-或者消息驱动API定义清楚的边界。微服务架构模式给采用单体式编码方式很难实现的功能提供了模块化的解决方案，由此，单个服务很容易开发、理解和维护。  
 第二，这种架构使得每个服务都可以有专门开发团队来开发。开发者可以自由选择开发技术，提供API服务。这就意味着开发者不需要被迫使用某项目开始时采用的过时技术，而是可以随着技术的更新换代而选择最新的技术。甚至于，因为服务都是相对简单，即使用最新技术重写以前代码也不是很困难的事情。  
 第三，微服务架构是每个微服务独立部署。开发者不再需要协调其它服务部署对本服务的影响。这种改变可以加快部署速度。微服务架构使得持续化部署成为可能。  
 最后，微服务架构模式使得每个服务独立扩展。开发者可以根据每个服务的规模来部署满足需求的规模。甚至可以使用更适合于服务资源需求的硬件。

微服务设计原则

。。。。。。。。。

2.1.3 Spring Cloud微服务架构

Spring Cloud是一个基于SpringBoot实现的微服务架构开发工具。它为微服务架构中涉及的配置管理、服务治理、断路器、智能路由、微代理、控制总线、全局锁、决策竞选、分布式会话和集群状态管理等操作提供了一种简单的开发方式。



2.2 SpringBoot框架

Spring Boot是由Pivotal团队提供的全新框架，其设计目的是用来简化新Spring应用的初始搭建以及开发过程。该框架使用了特定的方式来进行配置，从而使开发人员不再需要定义样板化的配置。其实spring boot其实不是什么新的框架，它默认配置了很多框架的使用方式。Spring Boot的出现大大简化了Spring应用程序的开发，其中最重要的是以下四个核心：

（1）自动配置：针对很多Spring应用程序常见的应用功能，Spring Boot能自动提供相关配置。

（2）起步依赖：告诉Spring Boot需要什么功能，它就能引入需要的库。

（3）命令行界面：这是Spring Boot的可选特性，借此你只需写代码就能完成完整的应用程序，无需传统项目构建。

（4）Actuator：让你能够深入运行中的Spring Boot应用程序，一探究竟。

2.3 Kotlin语言（可选）+JAVA8

Kotlin是由 [JetBrains](https://baike.baidu.com/item/JetBrains" \t "https://baike.baidu.com/item/Kotlin/_blank) 开发的一个用于现代多平台应用的静态编程语言，它可以编译成Java字节码，也可以编译成JavaScript，方便在没有JVM的设备上运行。

2.4 MySQL数据库

[MySQL](https://baike.baidu.com/item/MySQL/471251" \t "https://baike.baidu.com/item/MySQL%E6%95%B0%E6%8D%AE%E5%BA%93/_blank)是一种[开放源代码](https://baike.baidu.com/item/%E5%BC%80%E6%94%BE%E6%BA%90%E4%BB%A3%E7%A0%81" \t "https://baike.baidu.com/item/MySQL%E6%95%B0%E6%8D%AE%E5%BA%93/_blank)的关系型[数据库管理](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%AE%A1%E7%90%86" \t "https://baike.baidu.com/item/MySQL%E6%95%B0%E6%8D%AE%E5%BA%93/_blank)系统（RDBMS），使用最常用的数据库管理语言--[结构化查询语言](https://baike.baidu.com/item/%E7%BB%93%E6%9E%84%E5%8C%96%E6%9F%A5%E8%AF%A2%E8%AF%AD%E8%A8%80" \t "https://baike.baidu.com/item/MySQL%E6%95%B0%E6%8D%AE%E5%BA%93/_blank)（SQL）进行数据库管理。

MySQL因为其速度、可靠性和适应性而备受关注。它提供 [TCP/IP](https://baike.baidu.com/item/TCP/IP" \t "https://baike.baidu.com/item/mySQL/_blank)、ODBC 和 [JDBC](https://baike.baidu.com/item/JDBC" \t "https://baike.baidu.com/item/mySQL/_blank)等多种[数据库](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93" \t "https://baike.baidu.com/item/mySQL/_blank)连接途径，同时通过优化的 [SQL](https://baike.baidu.com/item/SQL" \t "https://baike.baidu.com/item/mySQL/_blank)查询算法，能有效地提高查询速度；而且mysql支持多线程，能充分利用CPU资源；MySQL还提供用于管理、检查、优化数据库操作的各种[管理工具](https://baike.baidu.com/item/%E7%AE%A1%E7%90%86%E5%B7%A5%E5%85%B7" \t "https://baike.baidu.com/item/mySQL/_blank)，这里我们采用轻量级的可视化管理工具SQLyog。本项目选择使用mysql作为数据库，一方面是由于其代码开源免费。

2.5 React 前端框架

2.6微信公众平台

1. ：需求分析

3.1 系统功能性需求

（使用UML图例）

3.2 系统非功能性需求

3.3 可行性分析

1. ：系统设计

4.1 系统总体设计

数据库设计（ER图）

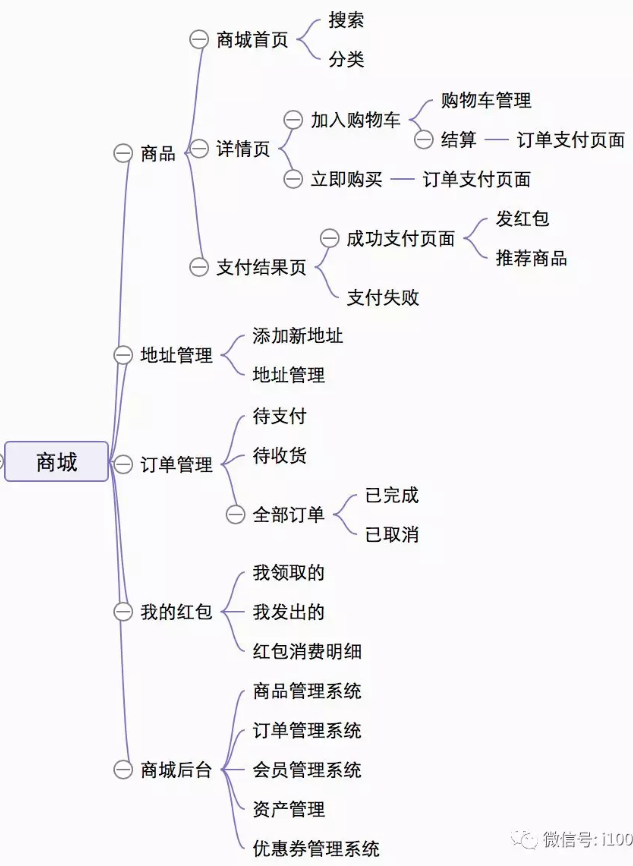
4.2 功能模块设计

（模块图为主）

会员模块、店铺模块、商品中心、订单模块、购物车模块、评价模块、物流中心

4.3 基于微型平台前台客户系统

商品展示、商品推荐系统、店铺推荐系统

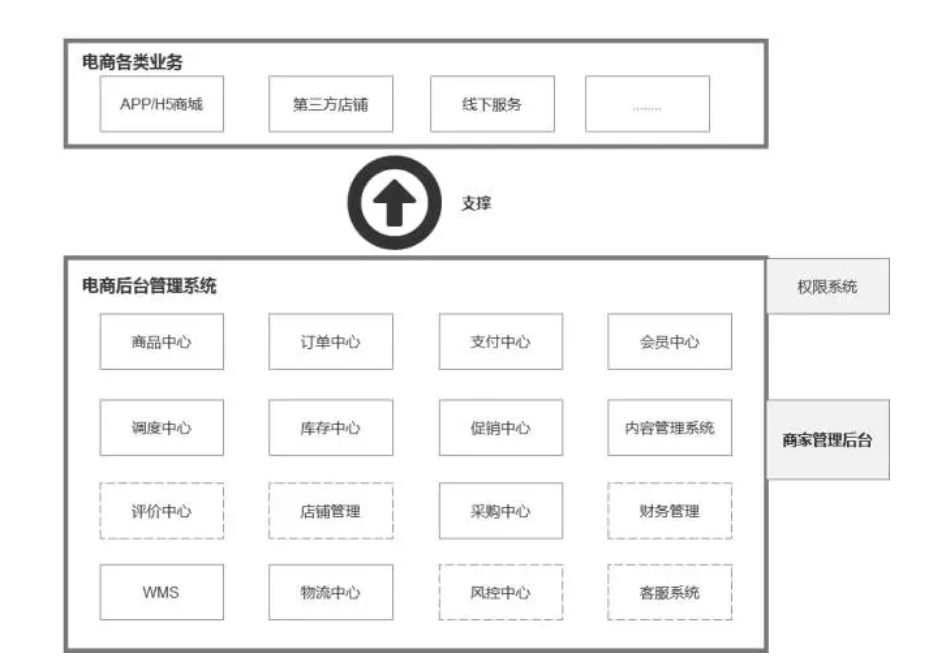


4.4 数据可视化的后台管理系统

（UML图为主）

会员管理、店铺管理、商品管理、订单管理、购物车管理、评价管理、物流中心

数据分析统计、可视化、货源地域分析





1. ：系统的实现与测试(实验设计与结果分析)。

5.1

5.2

5.3

5.4

1. ：总结与展望。

6.1 工作总结

6.2 未来展望

# 参考文献

# 致谢