****

本科毕业设计（论文）

**B2C电商系统设计与实现**

|  |  |
| --- | --- |
| **学院**  **专业**  **学生姓名**  **学生学号**  **指导教师**  **提交日期** | **计算机科学与工程** |
| **计算机科学与技术** |
| **池慧洋** |
| **201830570071** |
| **林伟伟** |
| **2022 年 5 月 14 日** |

摘 要

随着互联网的发展，越来越多的人使用电子商城购物，因此如何稳定高效的设计并实现一个B2C电子商务系统是如今需要研究的一重要问题。

从功能上来说，本项目分别设计并实现了商品模块、库存模块、搜索模块三个模块，另外实现网关模块实现服务之间的交互。其中功能最多的模块的是商品模块，其中包括属性分类、属性分组、品牌、平台属性、商品本身信息的管理。

从技术上来说，本项目在前端使用了VUE 和 Thymeleaf 框架，后端使用面向服务的分布式架构，服务开发中使用了 Spring 框架 以及 Java 语言 ，服务存储上分别使用了MySQL结合MybatisPlus存储数据。 技术的先进性主要体现在：本项目在服务治理方面使用了Spring Cloud 组件和 Nginx ，充分展现了微服务架构的优势，包括减少单点错误、错误隔离、允许在不同的模块中使用不同的技术、互相隔离持续的开发等。在商品模块使用Redis优化三级分类访问速度，在搜索模块使用ElasticSearch作为搜索引擎。

最后对此系统进行了各模块的功能测试以及性能测试与优化，测试结果为功能上顺利通过，所做的优化对系统性能有所提升。

关键词：电子商务系统；微服务；Spring 框架

Abstract

With the development of the Internet, more and more people use e-malls to shop. So, how to design and implement a B2C e-commerce system stably and efficiently is a significant issue that needs to be solved today.

Functionally speaking, this project designs and implements three modules: product module, ware module and search module, and realizes the interaction between services with the gateway module. Among them, the module with the most abundent functionalities is the product module, which is related with management of attribute classification, attribute group, brand, platform attributes, and product information.

Technically speaking, the VUE and Thymeleaf framework are used on the front end. The architecture on the backend is SOA, and the Spring framework and the Java language are chosen for implementation. In addition, MySQL and MybatisPlus are used on the service store to store data. Here are some highlighted technologies: the usage of Spring Cloud components in SOA governance combined with Nginx, which fully demonstrates the advantages of a microservices architecture, including no Single Points of failure, fault isolation, enable the leverage of the heterogeneous technologies, independent and continuous deployments, etc. the usage of Redis to optimize the access speed of three-level classification in the product module, and the usage of ElasticSearch in the search module as search engine.

Finally, The functionality of each module is tested, and the performance is optimized. The test result is that the function are all implemented successfully, and the optimization improves the system performance.

**Keywords:** e-commerce; Microservice; Spring Framework

目 录

[摘 要 I](#_Toc103442584)

[Abstract II](#_Toc103442585)

[目 录 III](#_Toc103442586)

[第一章 绪论 1](#_Toc103442587)

[1.1 选题背景及意义 1](#_Toc103442588)

[1.2 国内外研究现状 2](#_Toc103442589)

[1.3 技术方案调研 2](#_Toc103442590)

[1.3.1 前端技术综述 2](#_Toc103442591)

[1.3.2 后端技术综述 3](#_Toc103442592)

[1.3.3 总结本文技术方案 7](#_Toc103442593)

[1.4 本文主要内容与结构 8](#_Toc103442594)

[第二章 商城系统需求分析 9](#_Toc103442595)

[2.1 会员前台消费功能需求 9](#_Toc103442596)

[2.1.1 首页展示 9](#_Toc103442597)

[2.1.2 商品搜索 9](#_Toc103442598)

[2.1.3 商品详情展示 10](#_Toc103442599)

[2.2 商家后台管理功能需求 10](#_Toc103442600)

[2.2.1 商品管理 10](#_Toc103442601)

[2.2.2 库存管理 10](#_Toc103442602)

[第三章 商城系统设计 11](#_Toc103442603)

[3.1 微服务系统设计 11](#_Toc103442604)

[3.2 业务模块流程设计 12](#_Toc103442605)

[3.2.1 商品模块 12](#_Toc103442606)

[3.2.2 库存模块 13](#_Toc103442607)

[3.2.3 搜索模块 13](#_Toc103442608)

[3.3 程序设计原则 13](#_Toc103442609)

[3.3.1 数据结构设计原则 13](#_Toc103442610)

[3.3.2 后端程序设计原则 15](#_Toc103442611)

[3.3.3 前端程序设计原则 15](#_Toc103442612)

[3.3.4 服务治理设计原则 15](#_Toc103442613)

[第四章 商城系统程序实现 16](#_Toc103442614)

[4.1 商品模块业务实现 16](#_Toc103442615)

[4.1.1 数据结构 16](#_Toc103442616)

[4.1.2 后端程序 18](#_Toc103442617)

[4.1.3 前端程序（后管） 25](#_Toc103442618)

[4.1.4 前端程序（客户端） 26](#_Toc103442619)

[4.2 库存模块业务实现 27](#_Toc103442620)

[4.2.1 数据结构 27](#_Toc103442621)

[4.2.2 后端程序 28](#_Toc103442622)

[4.3 搜索模块业务实现 28](#_Toc103442623)

[4.3.1 数据结构 28](#_Toc103442624)

[4.3.2 后端程序 29](#_Toc103442625)

[4.3.3 前端程序 31](#_Toc103442626)

[4.4 服务治理功能实现 32](#_Toc103442627)

[4.4.1 使用Spring Cloud组件 33](#_Toc103442628)

[4.4.2 使用Nginx 34](#_Toc103442629)

[第五章 商城系统测试 36](#_Toc103442630)

[5.1 测试环境 36](#_Toc103442631)

[5.2 商品模块 36](#_Toc103442632)

[5.3 库存模块 44](#_Toc103442633)

[5.4 搜索模块 46](#_Toc103442634)

[5.5 性能测试分析与优化 47](#_Toc103442635)

[5.5.1 测试工具与性能指标 48](#_Toc103442636)

[5.5.2 对中间件性能测试与分析 48](#_Toc103442637)

[5.5.4 业务的性能优化 49](#_Toc103442638)

[第六章 总结与展望 51](#_Toc103442639)

[6.1 总结 51](#_Toc103442640)

[6.2 展望 51](#_Toc103442641)

[参考文献 52](#_Toc103442642)

[致谢 53](#_Toc103442643)

第一章 绪论

## 1.1 选题背景及意义

[我们的时代为电商系统的火爆提供了人口基础和物质基础。越来越多的人使用互联网，截至2021年1月，由WeAreSocial和Hootsuite提供的数据可知，全球已经有46.6亿人使用互联网，为互联网电商项目带来了广大的人群基础。而且，随着人们生活质量的提高，服务因为能够提升生活品质得到广泛的重视 ，电子商务由于可以加速工作效率，提高生活质量也在人们生活中占据着越来越重要的地位。因此，如何快速高效地构建一个稳定高效的电商系统是如今需要解决的一大问题。B2C电商系统可以使用互联网从事商务行为从而简化商业买卖活动，具体来说通过虚拟购物车和订单的方式，完成商品在商家和客户之间的流通。](https://www.remnote.com/doc/QgJS7g4XZP227vMp8)

[放眼全球，欧美地区的电子商务会比起步最早，互联网在1994年走进了全球人的视野，电子贸易也一时成为了发展的热点。1995年，Ebay以及Amazon公司在美国成立，欧洲也成立了许多新兴的网站。然而，2001年左右，是.COM的衰退期，导致了很多的纯电子商务公司消失，但更多的商业人士开始认识到电子商务系统的发展空间，用电商系统来拓展自己的业务。如英国第四大连锁超市Safeway就使用电子商务系统经营自己的副食业务。到了2016年，已经有80%的美国制造商拥有了自己的网站，全球电子商务市场规模超过25万亿美元，在世界经济上大放异彩。](https://www.remnote.com/doc/kE2oKxcTMhTckPjjc)

[虽然电子商务起源于欧美，但亚洲人口密度大，人力资源充裕，为电子商务带来了丰厚的土壤。亚洲地区的零售交易额在2017年全球市场已经占了46%，根据国家统计局的有关数据，2021年上半年中国网络零售市场交易规模为6.11万亿元 ，中国网络零售额也连续九年成为全球第一大网络零售市场。中国人的生活已经离不开各种电子商务平台。阿里巴巴提出的双十一，双十二等购物狂欢节深入人心，全球排名前十的电商企业，中国占有四席之地。具体来说，阿里巴巴拥有的淘宝等系统全球份额排名第一，京东商城排名第四。因此，能够支撑起如此庞大的业务量的这些公司使用的技术和架构也值得我们学习。](https://www.remnote.com/doc/gsMjMBoWJmBBFTwf7)

[按照交易形式的不同，目前市场电子商务系统上的交易形式可以分为五种 ，分别是B2B, B2C, C2B, C2C以及O2O。本文将实现的是B2C的商业模式，即商家自己经营商品给客户。](https://www.remnote.com/doc/pH37FPGHhYzvJpszJ)

## 1.2 国内外研究现状

[本节将别从设计，实现和优势方面总结了截止当前关于基于微服务架构的电子商务系统的研究，介绍当前的研究现状。](https://www.remnote.com/doc/GXiwP9tD7CThv6HEw)

首先从设计上，孙惟斯[1]结合服务设计需求提出B2C电子商务网站服务流程的体验设计原则。姚远[2]也基于服务设计理念，对B2C电子商务网站进行了设计分析。本文将会利用这些理念进行设计。

实现方面，Akhan Akbulut[3]通过三个案例学习，分别提供API 网关设计模式，责任链模式以及异步消息队列模式的优势和实现方式。鹏向峰[4]在2013年将业务功能按照四个层次，采用MVC分层架构，设计并实现了B2C电商系统。2014年，Xiaodi Huang[5]分析并测试了新奇的基于第三方支付系统的商务系统Bulapay。 徐士川[6]在2018年设计并实现了电子商城系统中的订单与秒杀模块之后，熊益益[7]和钟俊林[8]都分别在2019年设计并实现了基于微服务架构的自助微商城。另外，阿里巴巴和京东的后台也都使用了分布式的架构。

最终，我从前人的研究中验证了我们最终选择并实现的微服务架构的优势。Wilhelm Hasselbring[9]的Microservice Architectures for Scalability, Agility and Reliability in E-Commerce一文中通过讨论了耦合，集成，扩展和监控在otto.de中的实现，展示了微服务系统灵活，可靠与扩展性的优势。J Abdul Rasheedh[10]探索了响应式的微服务可以很低的开发需求达到高度的灵活和可拓展。最后Armin Balalaie and Abbas Heydarnoori[11]解释了微服务系统在运维方面的优势。

## 1.3 技术方案调研

为了形成一个完整的方案，本章将对市场中应用的常用的技术进行调研来确定本项目的技术路线。

### 1.3.1 前端技术综述

本项目在后台管理系统的前端使用了[1.3.1.1 VUE](https://www.remnote.com/doc/Ysqvpdu4Gu8adEbHJ)，在前台的前端使用了[1.3.1.2 Thymeleaf](https://www.remnote.com/doc/zNHwGSTGWGXCKkgwq)作为前端开发框架。

1.3.1.1 VUE

[VUE的核心思想 是MVVM思想 ， 在此之前，开发人员需要从后端获得需要的数据模型，之后需要手动操作DOM完成Model和View的同步。 而使用VUE之后，就可以将DOM 操作封装在指令中 ，完成Model和View的自动同步 。用此概念构建出来的系统也被称为响应式系统。](https://www.remnote.com/doc/dgJGyLptPz5TPeocn)

[在本项目中主要使用的功能为 VUE声明式渲染 与 5）、双向绑定。 要使用声明式渲染，首先要首先创建Vue实例 ，之后要 构造函数接收一个对象 ，通过设置对象里的属性分别绑定视图 ，封装方法 ，封装数据 。](https://www.remnote.com/doc/beJrodxNXF8gMQp4Q)

[VUE为我们提供了很丰富的指令 ，表达形式常为v-xx ，其预期值为 单个JavaScript表达式 ， 表达式的值改变时，将其产生的连带影响，响应式地作用于DOM 。 在本项目中还大量的使用指令进行绑定，事件处理 ，循环和判断。](https://www.remnote.com/doc/2H4RuDxNFtf5u8sz6)

1.3.1.2 Thymeleaf

在前台选择使用Thymeleaf的原因有两个，首先是避免前后端分离的性能损失，其次最重要的是[前后端分离这种模式不利于SEO](https://www.remnote.com/doc/9EbzaZJrKZivBR5nd)，具体来说， [在前后端分离的项目中，要获取页面的全量数据需要执行js代码。而例如百度蜘蛛是执行不了js的，因此，如果使用前后端分离，会导致即使百度收入了我们的商品页面，也不会有真正的数据显示。](https://www.remnote.com/doc/rTbQgPKpYe3qsCWcA) 因此，本项目选择使用Thymeleaf作为模板引擎。

Thymeleaf中[比较常用的就是表达式 和th属性 。表达式可以完成变量取值，选择变量 等功能，而th属性 可以完成设置属性值 ，迭代 和 条件运算 的功能。](https://www.remnote.com/doc/9r5tjEZRPTAA4zhcD)

为了形成一个完整的方案，本章将对市场中应用的常用的技术进行调研来确定本项目的技术路线。

### 1.3.2 后端技术综述

1.3.2.1 分布式架构

首先要选择的是软件的架构，最简单的架构是[Single-Tier applications](https://www.remnote.com/doc/FpxrpN2DbQX5TfLqJ)，在这种架构里，[所有功能都部署到一个功能里](https://www.remnote.com/doc/hN5LF2JGRn72Peut6) ，这种架构的优势是[简单易用](https://www.remnote.com/doc/nd6PhELGT2uWLrreR) ，但缺点也很明显，比如[性能扩展比较难](https://www.remnote.com/doc/8DSTzRvf6jWsvSHXR)， [不利于协同开发](https://www.remnote.com/doc/aZGWiNJpDxdJh6jme?aliasId=H67SeF7dSZwaYtjr6) ，因此这种系统[很难升级维护](https://www.remnote.com/doc/qk7M5wqbxedW7RpL8?aliasId=xNWSW9qMKi3PjED2p) 。之后是[垂直应用架构](https://www.remnote.com/doc/wdXzzNFXqGbL9vwNP) ，[通过切分业务来实现各个模块独立部署](https://www.remnote.com/doc/miXKHMbAxBSP3Xefr) ，但是坏处是[公用模块无法重复利用](https://www.remnote.com/doc/MQnasaehYGgbaqSTs) 。 因此通过[将各个应用通过分层独立出来](https://www.remnote.com/doc/aMay4ZqqCeiakoNHa) 可以实现各个应用之间的相互调用，这种架构称为[分布式应用架构](https://www.remnote.com/doc/z677sx82xSe8XSw3T) ，但是[模块之间关系的变化会造成配置需要修改。](https://www.remnote.com/doc/m6X7RCSCtD6fm5WSZ?aliasId=4cwu3QXeuHp6PdMru) 对于电商系统[数据量大](https://www.remnote.com/doc/zL6Q64N8XGQXuwYfX) 的特点，我们可以选择分布式的存储数据，减少每个数据库的数据量。为了解决[并发度高](https://www.remnote.com/doc/NMGkdXiagEPvFBpz6) 的问题，我们可以通过分布式需要提高系统的可拓展性，来增加峰值流量。结合以上考虑，我选择使用分布式架构。最后，为了使这个大而复杂的系统更加容易维护和管理，此项目会使用[面向服务的分布式架构](https://www.remnote.com/doc/E6YNZJsgFkjjJ9iKZ) ，通过配置中心来协调各个微服务之间的关系。

1.3.2.2 服务开发技术

（1）Java 语言

在语言的选择上考虑社区的活跃数量以及我本身的熟悉程度，我考虑将C++, Java, Python作为候选对象，以[执行效率](https://www.remnote.com/doc/ttcCLWP3g82CfbMNB)和[开发效率](https://www.remnote.com/doc/wpGgon5GYJqtiNpXh)为标准，对后端语言作出选择。对于[Python](https://www.remnote.com/doc/fsnS5gfdSvaqNo5v9) ，[Python语言短小精悍 在Web开发上也有其对应的框架比如Django, Flask, 因此我们花比较少的时间就可以开发一个完整的项目](https://www.remnote.com/doc/daL5jNKYknZeQJGpL)。但由于 [其为解释型语言，执行效率较慢](https://www.remnote.com/doc/F9oZdZyTmG8QFuZMv?aliasId=HfXCe4ZXuJHNreTC5) ，[其次由于 GIL 的存在，不能实现多个核心的并行，执行效率较慢，因此排除](https://www.remnote.com/doc/M6kLSnRdqiC4MN6ii) 。对于C++，[由于可以直接面对硬件编程，效率很好](https://www.remnote.com/doc/zYmzfFHAWj6cdRxHq) ，而且其通过兼容的C语言的调用操作系统的系统调用[支持高效并发](https://www.remnote.com/doc/dhuaxAKcZHt2iPHQW)，[但开发成本较高，体现在市面上存在的库不规范，不成体系，故排除](https://www.remnote.com/doc/hqYRqKaodWiWzHWxu) 。最后JAVA[因为JVM的存在既是编译型语言，又是解释型语言，因此效率介于C++ 和 JAVA之间](https://www.remnote.com/doc/jFwdnsPd2cKC4Lith) ，可以满足电商系统的需求，而且[有众多Web开发框架，其中最著名的就是Spring家族的框架，而其优势就是可以利用Spring Cloud组件方便的实现上一段所说的微服务架构。](https://www.remnote.com/doc/GYjQQBAgTb7rvtmQz?aliasId=q2aZ2jpp5qppkydBv) 因此，最后结合执行效率和开发效率，我选择java作为毕业设计的实现语言，Spring框架作为开发的主要框架。

（2） Spring 框架

对于框架的选择，最具有影响力JAVAWEB框架就是Spring全家桶。Spring框架的核心概念为[@Bean](https://www.remnote.com/doc/hzD3MAWRp65o8hswG)以及与其相关的[AOP](https://www.remnote.com/doc/NWhKqGGuAKAgq5qAC)和[IoC](https://www.remnote.com/doc/k4kfGipKJZvCMpPDe)，之后，Spring MVC为Spring提供了MVC框架，通过Dispatcher Servlet, ModelAndView 和 View Resolver，让应用开发变得很容易。 但随着Spring的配置文件越来越多，其配置过程也变得越来越繁琐，比如，我们需要手动添加 Maven 配置、各种 XML 配置，Pivotal为了解决这个问题，推出了Springboot，通过提供自动配置功能，降低构建一个项目的复杂度。因此，在Springboot的帮助下，我们只需要简短的几行代码就可以实现一个接口服务。

随着容器化的普及，微服务Spring社区也为我们提供了Spring Cloud开发和治理框架，让我们更好的管理微服务。由于本项目后端选择使用微服务架构实现，因此在实现上选择使用Spring Cloud + Spring Boot 的结合

1.3.2.3 数据存储技术

本节将介绍本项目使用到的技术[MySQL和MybatisPlus](https://www.remnote.com/doc/uFhoPC8wBweYb9NfW)， [Redis](https://www.remnote.com/doc/dPgncxtSeEy2PmJab)，以及 [Elasticsearch](https://www.remnote.com/doc/8Rop9fh7bybDEcY9o)。

（1）MySQL和MybatisPlus

常见的可以用于SQL的持久层框架有Mybatis和Hibernate。[使用Mybatis我们可以手动编写SQL语句](https://www.remnote.com/doc/5WyWynN8Ew8gnCNyf?aliasId=QgtdYhNpWomL9LoNQ) ，因此[Mybatis和Hibernate相比的优势之一是灵活度比较高](https://www.remnote.com/doc/P4HMdFiaaeWGrNzzf?aliasId=msxHQgeNm8YujYTFK)。其次，灵活度比较高带来的开发过程比较繁杂的问题，我们还可以用已经成熟的无代码开发项目[MyBatisPlus](https://www.remnote.com/doc/SJRinamF9uS3CNb7L) 来解决。因此，综合开发灵活度，开发复杂度考虑，本文选择使用MybatisPlus系列框架。

[此框架主要使用了ORM的概念](https://www.remnote.com/doc/QLGZwYSNkYG3NpedX?aliasId=JDfsmQXeFZF6HYx52) [封装了JDBC](https://www.remnote.com/doc/dFSfoKkQt9SEmHEFw?aliasId=WtFRdPiPZhpjTucCi) [因此我们可以只关注如何编写SQL，而不关注如何连接，关闭数据库。](https://www.remnote.com/doc/Ths9nE9Nfnn5eLYqY?aliasId=EoiQ49WKtdQiqqkdn)

[MyBatis还有以下的设计特点，其提供接口简单而又相关性弱，由于其接口和配置的简单性，其可以支持极其快速的开发。 而且，其还可以支持多语言和平台，比如除了Java，它还支持Ruby以及C#等主流语言。 不仅如此，它还免费以及开源，能够吸引到更多的人加入到此社区。 最后，它还支持很多的高级特性，因此其有很高的学习价值。](https://www.remnote.com/doc/7YoExHBcvDTwdAzF8)

（2）Redis

[Redis是一种常用的缓存中间件](https://www.remnote.com/doc/9ZkaLSvL5bvsoiN5k)，SpringBoot对其进行了整合。其中[此整合是通过“RedisAutoConfiguration”来完成的，它会将所有的配置信息，都放置到了“RedisProperties”中。 容器的自动配置还会为容器中添加RedisTemplate 和 StringRedisTemplate 。他们都用Map操作 Redis ，只不过由于Redis的String操作较多，StringRedisTemplate规定了key和value都要是String，进行针对性的优化](https://www.remnote.com/doc/pogyJJKEvBdvikPth)

[另外，Lettuce 和 Jedis 是 redis 的客户端，RedisTemplate 是springboot Lettuce 和 Jedis 的再一层封装。 因此RedisAutoConfiguration自动配置类，会导入 Lettuce 和 Jedis 的配置类](https://www.remnote.com/doc/ZmHFj5TWwYaMuC55z)

（3） Elasticsearch

ElasticSearch是常用的全文检索引擎。本项目中用到的最核心的概念为[全文检索](https://www.remnote.com/doc/p28iBE5qdHHRWE4Fv)，意味着[对一篇文章进行全部字段索引](https://www.remnote.com/doc/p28iBE5qdHHRWE4Fv?aliasId=ubkHqJWqjT94eCzgm)。

与传统关系型和非关系型数据库不同的是，其建立了 [倒排索引](https://www.remnote.com/doc/2sRWtsEmQ6J3en2QY?aliasId=NRXNkqXWnFqPFvTX8) ，建立一个[Trie](https://www.remnote.com/doc/QSH8jd24A7zeYasjw) 树，[再加上一些压缩技术](https://www.cnblogs.com/dh-dh/p/12851655.html" \l ":~:text=%E6%9F%A5%E6%89%BEterm%20dictionary)%E3%80%82-,%E5%86%8D%E5%8A%A0%E4%B8%8A%E4%B8%80%E4%BA%9B%E5%8E%8B%E7%BC%A9%E6%8A%80%E6%9C%AF,-%EF%BC%88%E6%90%9C%E7%B4%A2%20Lucene%20Finite)，极大的提高了内存存储的使用率。

但是由于其与Mysql均为存储数据的工具，因此在概念层面还是有很大的相似之处。下面通过与Mysql类比来介绍其核心概念：

[其数据架构的主要概念](https://www.remnote.com/doc/SmqLzHyjh3ph4K5as?aliasId=auxpjfmzcx2MM872p)有[索引](https://www.remnote.com/doc/AWyaiu45BGPyQToqz?aliasId=ep9HBMbM82RmiEd3S) ，[类型](https://www.remnote.com/doc/XD6cQnBBrRXtLY2b5?aliasId=giQqsEiTsrFJDfJSk) ，[文档](https://www.remnote.com/doc/7MAeFx3e2FrDk2Yyh?aliasId=XXJbr6B3TBYaxoE7S) ，和[属性](https://www.remnote.com/doc/QRbgsDJJbss92FLut?aliasId=DAmjBMdgKGtm77cuy) 。分别类似于Mysql的数据库，表，数据行和数据列。

[ES中的增PUT/POST、删Delete、改\_update、查GET 操作对应于Mysql中的在数据库中的增insert、删delete、改update、查search](https://www.remnote.com/doc/ewr5DvFfCMggEsHB5)

其[Mapping](https://www.remnote.com/doc/z3dAqM52qAuWBBxEc) 对应了[schema定义表、每个表的字段，还有表和字段之间的关系。](https://www.remnote.com/doc/amERhFzvL72obyGxL?aliasId=fXEsRLGG9YxBuPfwm)

关于数据类型中的[复合类型](https://www.remnote.com/doc/Wq7sXvmc4czWSLm5j) ，本项目较多的使用[nested](https://www.remnote.com/doc/vzwasx4KJ6C6JENwd) 来 [存储对象数据类型](https://www.remnote.com/doc/YhMraRxShPnhXmfWu)

1.3.2.4 服务治理技术

（1）Spring Cloud 组件

Spring Cloud 为我们提供了众多的组件来管理微服务，我们将按照我们需要的功能来选择组件。比如，我们需要[中心化的发现，配置微服务](https://www.remnote.com/doc/yqks79z24oF6RBQXo?aliasId=Mmop7eRR4ZbPtWhJv)， [服务之间的通信](https://www.remnote.com/doc/9QztindR6xY9yyb3m?aliasId=ZcZjnadZLgePji2G3)。 以及 [追踪服务之间的通信](https://www.remnote.com/doc/gZNFmroAYMT93NWYM?aliasId=tJE2z5dZKnq39Ksvp)，在意外情况发生时，我们还需要 [使用断路器熔断](https://www.remnote.com/doc/HLtSqsoh47D8iRdwr?aliasId=2yMtyHuwDL6sLcFNx)，以及其他辅助功能。

在中心化的管理上，所需要的功能分别是服务发现，服务配置管理以及服务之间的链接。在原生的组件中，[Eureka](https://www.remnote.com/doc/thmaCw6b6wj5tPHKC) 和 [Consul](https://www.remnote.com/doc/ozQA5Nzinka7mB8sM) 可以实现服务发现，我们使用 Spring Cloud Config Server 完成服务配置的中心化，为了能够在应用配置更新的时候统一更新，我们还需要运用[Spring Cloud Bus](https://www.remnote.com/doc/jhZZuvZNnDcBmXKAN) 把所有的实例注册到总线上。之后，阿里巴巴推出了Nacos组件，此组件结合了上述三者的功能，使我们只使用Nacos即可完成服务的注册和更改的配置与全局刷新，大大提高了生产效率。因此，本文中为了解决中心化管理所选择的组件是Nacos。

为了实现[服务之间的通信](https://www.remnote.com/doc/9QztindR6xY9yyb3m?aliasId=ZcZjnadZLgePji2G3)，我们需要实现的功能是传递信息和路由，为了实现这两个功能需要用到的技术是网关，可供选择的有Netflix的Zuul以及Spring Cloud 原生的API Gateway，由于Zuul在2020.0.0版本发布的时候，作为Netflix组件之一被移除，[并且官方推荐使用Spring Cloud Gateway 作为替代](https://www.remnote.com/doc/ghDqZJQeJdp8YyfiG?aliasId=K7tHH8KTF6wEo6qju) ，因此本文在技术选择的时候也选择使用了Spring Cloud Gateway。另外，为了简化服务之间的调用，我们还用了Open Feign组件封装HTTP请求。

在服务间可以通信之后，由于数据分析或者调试异常的需要，我们还需要[追踪服务之间的通信](https://www.remnote.com/doc/gZNFmroAYMT93NWYM?aliasId=tJE2z5dZKnq39Ksvp) ，在这里选择的技术是[Spring Cloud Sleuth](https://www.remnote.com/doc/vvnBiQMSzsjnPC3GK)和[Zipkin](https://www.remnote.com/doc/HQsJt7skZ3mdw2TDe)。最后，当负载过大时，我们可以使用断路器[Sentinel](https://www.remnote.com/doc/pW43YBfbEERLsCS7q) 限制流量。

（2）Nginx

本项目选择使用Nginx来完成反向代理，从而实现动静分离和负载均衡功能。在首页中，静态资源的获取由Nginx提供。

具体来说，首先，Nginx可以[隐藏真实的服务器资源](https://www.remnote.com/doc/LysjY4fg5ESadEz6M) ，提高服务器的安全性。使用Nginx的第二个好处是，通过Nginx的代理，前端的请求可以根据目前节点的状态，被分配到合适的服务器上，从而减少单个服务器的负载。最后，动静分离可以通过其提供的[代理缓存](https://www.remnote.com/doc/azZsgp52oCwGL9NKf)，和[对任意的请求访问都通过gzip压缩处理](https://www.remnote.com/doc/XL8heghXYztcFku5g)来加快访问速度，[降低系统调用I/O产生的系统瓶颈](https://www.remnote.com/doc/aNF8TAjDMxNf4bQ5u)。

使用Nginx之后[反向代理的数据流](https://www.remnote.com/doc/krf2e4ALn2iEKKqmG)是：[在本机上，浏览器请求的域名，如hannahmall.com，在本机被解析为Nginx所在的设备。](https://www.remnote.com/doc/iTLX7BScB4yaB7NZE) [在nginx上，nginx在其端口（本项目中是80）接收到请求后，解析请求头求得host，然后使用在“hannahmall.conf”中配置的规则，将请求转到hannahmall-product服务部署的位置的对应端口。](https://www.remnote.com/doc/9uotsRv9w3ZSFznuL?aliasId=CzSDZAQLzSYkRWW2n)[在服务器上，该服务响应请求，返回响应结果。](https://www.remnote.com/doc/7wWoPcucW6AvXqCrP)

### 1.3.3 总结本文技术方案

1.开发工具

后端开发工具：IDEA

前端开发工具：Vscode, [vue-tools](https://www.remnote.com/doc/s5zYAw72QJQn7fPSD?aliasId=a6ZN2KBjHz73DfpPS)

数据库设计工具：PowerDesigner

网页调试工具: Chrome DevTools

API接口调试工具：Postman

性能压测工具：Jmeter

远程连接工具：MobaXterm

2.开发环境

开发语言以及版本：Java, JDK1.8

数据库

关系型数据库：Mysql

非关系型数据库：Redis

依赖管理工具：Maven

3.开发框架

前端：Vue

后端：Spring Boot, Spring Cloud

4.组件的选择

Nacos

Feign

Gateway

[Sentinel](https://www.remnote.com/doc/tzZqiQ42jkcnHFztD?aliasId=wLKGo5dYoLHdtKuAq)

## 1.4 本文主要内容与结构

[本文的研究主要内容是如何使用微服务实现一个B2C网上商城，以达到配置简介化，开发敏捷化，业务灵活化的目的。在研究学习的过程上，本论文将遵循软件开发的通用步骤，在第二章中，我们将从需求分析入手，分别进行2.1 会员前台消费功能需求分析和2.2 商家后台管理功能需求 分析，之后从用户需求中提炼出不同的功能模块，在第三节中，我们将根据第二节的需求分析，在 3.1 微服务系统设计 中将系统拆分成微服务模块，之后在3.2 业务模块流程设计 中介绍我们的系统是如何满足上述需求的 。之后本章将依据3.3 程序设计原则 分别进行 4.2.1 商品模块业务实现 ，4.2.2 库存模块业务实现 ，4.2.3 搜索模块业务实现 （包括 数据结构以及后端程序 ）以及4.2.4 服务治理功能实现，来获得一个一体化的微服务系统。在第五节中进行程序的测试来验证系统的可靠性，首先介绍5.2.1 测试环境，之后对照第四章程序实现介绍各个功能模块的功能测试，最后对模块的某些功能进行了5.2.5 性能测试分析与优化。](https://www.remnote.com/doc/S334RipdNxYxK2WLT)逻辑顺序上，每个章节都依赖于前一个章节的工作，每一个章节也都是上一个章节工作的延伸。

[我通过校内实习，掌握了Java, Spring Boot, Spring Cloud组件的使用，学习了正在搭建的卓燃后台系统的架构和代码逻辑，自己搭建了一个货币兑换系统的演示。因此，本论文将在此基础之上，进一步添加更多功能，解决技术难点，构建一个完备的微服务系统。](https://www.remnote.com/doc/jkPdhXv4RtSB4kg8L)

第二章 商城系统需求分析

在第二章中，我们将从[需求分析](https://www.remnote.com/doc/mxcc7mmgvZuzGEmhG)入手，分别进行[2.1 会员前台消费功能需求](https://www.remnote.com/doc/BogPNcboveTnqgYsK)分析和[2.2 商家后台管理功能需求](https://www.remnote.com/doc/iMqrPbru2Sf5vPfKn) 分析

## 2.1 会员前台消费功能需求

用户使用我们的商城，通过[首页展示](https://www.remnote.com/doc/vLNb3MmWBQqR5teft?aliasId=kAQGwP4sBJj4kBmMT)和[商品搜索](https://www.remnote.com/doc/6YcA6QyRCNdKgPn5J?aliasId=NMi8eotTyyXWR6yZt)获得商品信息，之后可以跳转到[商品详情页](https://www.remnote.com/doc/TWHuC43SCXSemBzRk?aliasId=fAKkFQ62LZf5crDPY)获取商品详情。

### 2.1.1 首页展示

商城首页是用户登录后首先开启的页面，根据[首因效应](https://www.remnote.com/doc/3c9hTNoTCFsoosX57?aliasId=kmshsNAbmDCCp3uRb)，也就是用户形成的第一印象会很大程度决定用户之后的体验，B2C商城的首页设置必须足够有吸引力。 此页面需要包涵用户所需要的基本信息，以及轮播图。其中最重要的是通从首页进入搜索，从而抵达自己需要的商品详情页面。

下面是用户在首页使用搜索功能的两种方式：

用户可以[通过在搜索框输入检索关键字展示检索页面。](https://www.remnote.com/doc/3mnFYouGLorYmqA2e?aliasId=pHXTsSY78JRzEZQ8z)

[用户可以通过侧边的三级分类进入商品检索](https://www.remnote.com/doc/wHjruq3KL2eHit7sK?aliasId=DFvnGjZEFdqhv4HXR)。[由于访问首页时就要在页面的侧边查询，加载一级目录,所以我们需要在加载首页时获取该数据](https://www.remnote.com/doc/YhSZW855PEHAiGMQs) 之后先获取一级分类数据，用户选中后在查询二级分类数据

### 2.1.2 商品搜索

为了完成商品搜索，首先[商家要上架商品](https://www.remnote.com/doc/K59iPNWsKrTdrfqnM?aliasId=Dn6hrz7focz4yyaRB) ，在商品上架了之后，[会员就可以进行数据检索](https://www.remnote.com/doc/oN3s9pAskTPF3CLhn?aliasId=m45XkTozDuaooiBFy) 。

商家上架商品 ：

并不是商家录入的数据都需要被用户检索，商品在上架之后才可以被检索 ，商家点击上架后，商品的状态会由新建变成上架，会员才可以检索到。

会员检索商品 ：

刚刚在[2.1.1 首页展示](https://www.remnote.com/doc/vLNb3MmWBQqR5teft) 中提到了用户可以[用两种方式从首页进入搜索页面](https://www.remnote.com/doc/KtRqb9S64vT7qz3AN?aliasId=FLpHiGpu5XYTWapMG)，接下来我们要介绍搜索页面的需求

[进入搜索页面之后，用户需要使用不同的检索手段获得自己需要的信息，常用的手段有，全文检索 （对应关键字高亮 ） 过滤检索 ，排序检索 ，具体来说，全文检索指的是用户输入商品名称来进行检索 。过滤检索指的是，后台管理系统按照品牌 ，分类 以及其对应的基本属性聚合以提供筛选条件 ，之后用户通过筛选条件过滤出自己所需要的信息。排序检索指的是用户根据指定字段排序搜索结果。另外我们还提供了分页展示的功能。](https://www.remnote.com/doc/fg8AnmNeegpLLFeHw)

### 2.1.3 商品详情展示

商品详情页的主要功能为让用户了解商品的细节，具体来说，商品的细节有：[标题和副标题](https://www.remnote.com/doc/yXWji7DuqmHa5ZfMi) ，[商品图片](https://www.remnote.com/doc/8cQuyjekSQGCjmntE)，[商品价格](https://www.remnote.com/doc/4s5zPMa6CQiyGieZa)，[是否有货](https://www.remnote.com/doc/ZnvJYrRbsPHwEXGM7)。最后还有 [商品介绍](https://www.remnote.com/doc/6EtHAdhWSZqSgmqPm) 和 [规格包装](https://www.remnote.com/doc/WAo37Wn24if4L5YTG)。

## 2.2 商家后台管理功能需求

后台承担管理的职责，这里主要实现[商品管理](https://www.remnote.com/doc/sedSoY2rujC5YFzGo?aliasId=MxrLSfCpZ2WipZZAg)，和[库存管理](https://www.remnote.com/doc/wGyMJ7RCpZz3MDzBc?aliasId=ux2t4R8DA5ihjPm9n)的内容。

### 2.2.1 商品管理

在添加一个商品时，我们可以借用[类与对象的概念](https://www.remnote.com/doc/rRnJvoHvBWeNXyb5A?aliasId=5ixgJtCsTfgRcJGh8) ，把平台商品的存储分为两个步骤，首先我们存储[SPU](https://www.remnote.com/doc/DtfTr986gvPrBtFok)，即标准化产品单元，来增强代码的可复用性。为了区分不同的库存单元，我们引入了[SKU](https://www.remnote.com/doc/yD8Dhd9Rj9QHFNi4P) 的概念。

商品在平台的属性分为两种，一种是[基本属性](https://www.remnote.com/doc/2BBxWApGC84yBzbmq)，对应前文所说的SPU，之后考虑此商品有不同的库存，因此我在基本属性的基础上又添加了[销售属性](https://www.remnote.com/doc/RKZsJFb7jMYeekvzw)。我们还要考虑到底如何存储一个商品，什么样的信息如何在前端进行展示。

为了解决商品属性多且杂乱的问题，我们还要用一个结构来整理商城的信息，因此从商城的结构划分来说商城应该有不同的[属性分类](https://www.remnote.com/doc/EK2KyMfQe7TRPpYGA)，在不同的属性分类下还可以设置不同的[属性分组](https://www.remnote.com/doc/adtzBB4oPPx6g8E87)，也就是说，[属性分组](https://www.remnote.com/doc/adtzBB4oPPx6g8E87)是以 [属性分类](https://www.remnote.com/doc/EK2KyMfQe7TRPpYGA) 组织起来的

因此，商家通过操作商品后台系统完成商品的管理。具体来说，商家管理的内容有[商品本身的元信息](https://www.remnote.com/doc/cvgxWKnXMCs8EzKaR)，[平台属性](https://www.remnote.com/doc/PqHkXn9AHEwyM3Yjq)，以及[属性分类和属性分组](https://www.remnote.com/doc/zG7CMYPx7zPadjZng?aliasId=ELxm9PbfvzjBCG3cu)。

### 2.2.2 库存管理

库存管理的基础功能为管理存储在仓库里的商品。商家需要库存管理的功能来管理自己的仓库与所存的商品，因此首先商家会有查询和创建仓库和库存的需求。之后为了更新库存，还需要创建采购需求，如果要批量处理采购需求还需使用采购单需求。

第三章 商城系统设计

在本节，我们将根据第二节的[需求分析](https://www.remnote.com/doc/mxcc7mmgvZuzGEmhG)，在 [3.1 微服务系统设计](https://www.remnote.com/doc/yBrNNhwLs39dGoM88) 中将系统拆分成微服务模块，之后在[3.2 业务模块流程设计](https://www.remnote.com/doc/8fJQcKXtnWe9qjetz) 中介绍各个模块之间流程的组合。之后在[3.3 程序设计原则](https://www.remnote.com/doc/4PB7rfzm6cMRYZDMY) 介绍程序设计的原则。为了保持文章的连贯性，依照原则设计的具体实现将在第四章描述。

## 3.1 微服务系统设计

在[1.3.2.1 分布式架构](https://www.remnote.com/doc/Y8rugraMCFxKd9Pdi) 中已经说明，本项目选择使用基于SOA的分布式模型来实现本电商系统，在[第二章需求分析](https://www.remnote.com/doc/mxcc7mmgvZuzGEmhG?aliasId=8QuDgZTZafbmB2LT4)之后我们还需要[根据业务需求划分微服务模块](https://www.remnote.com/doc/5accPSZCazmiBEZYL) ，之后根据需求新增非业务模块。

在业务边界的划分上，搜索服务主要对用户提供[2.1.2 商品搜索](https://www.remnote.com/doc/6YcA6QyRCNdKgPn5J) 功能支持，库存服务主要对商家提供[2.2.2 库存管理](https://www.remnote.com/doc/wGyMJ7RCpZz3MDzBc)支持，商品服务对商家提供[2.2.1 商品管理](https://www.remnote.com/doc/sedSoY2rujC5YFzGo) 支持和对用户提供 [2.1.1 首页展示](https://www.remnote.com/doc/vLNb3MmWBQqR5teft) 和 [2.1.3 商品详情展示](https://www.remnote.com/doc/TWHuC43SCXSemBzRk) 功能。此处选择合并用户和商家需求在同一个模块实现的原因是：用户和商家实际上是使用的同一张数据表，[如果割分为两个模块，会导致出现一库多服务的反范式](https://www.remnote.com/doc/JYNhooFYzg8dp8Byn)。

关于非业务模块，为了代码的复用与可读性，本项目还需要拆分出[通用代码](https://www.remnote.com/doc/cRYbBhqwvBwJSEjsg?aliasId=qCNduGdQJEueEiTA2) 模块存放微服务之间的共同内容。 比如[定义常量](https://www.remnote.com/doc/BiAR6JurxyQP78ueM?aliasId=QBr9X6RcSekP7wAPh) ，[自定义异常](https://www.remnote.com/doc/TBe6YHB2DKgiB8GBf?aliasId=tsyWeRYFJQoWqTFBq) ， [通用工具代码 ，包括分页工具类 ， 查询工具类 ，前后端统一返回结构 等](https://www.remnote.com/doc/Hk9bLpdmNLfRMfd3G) 。

本项目选择设计的另一个非业务模块是网关，此模块实现了两个功能，第一个功能[使用网关统一配置跨域](https://www.remnote.com/doc/yqJSgGmvbtbD7zEDA?aliasId=x3rc94WHgZTEW5D38) 来处理跨域请求，第二个功能是[配置网关路由与路径重写](https://www.remnote.com/doc/oFpTizxypH8EsnDFg?aliasId=amikZ93eJG8X37568)，进一步转发请求以实现负载均衡。

[跨域指的是浏览器不能执行其他网站的脚本 ，是浏览器对javascript施加的安全限制。 当访问不同协议， 域名， 端口 中的内容时需要进行跨域。](https://www.remnote.com/doc/LjAMhYZQHjuoCaBor) [为了解决跨域，有两种常用的方法，第一种使用nginx部署为同一域 ， 第二种是配置当次请求，允许其跨域。 具体来说，需要给请求添加相应头](https://www.remnote.com/doc/dxkqBg9p5SoXdjLuv) 。在本文中选择第二种， [使用网关统一配置跨域](https://www.remnote.com/doc/yqJSgGmvbtbD7zEDA?aliasId=x3rc94WHgZTEW5D38) 。

## 3.2 业务模块流程设计

Diagram

Description automatically generated

图3-1：B2C商城流程图

图3-1为三个模块的流程图，绿色为库存模块负责的内容，黄色为商品模块，红色为检索模块。每一个模块基本都会使用到[3.1 微服务系统设计](https://www.remnote.com/doc/yBrNNhwLs39dGoM88) 中分析的非业务模块的功能。

### 3.2.1 [商品模块](https://www.remnote.com/doc/TWHuC43SCXSemBzRk?aliasId=PwHBJtG3oAyBYHXnK)

在[2.2.1 商品管理](https://www.remnote.com/doc/sedSoY2rujC5YFzGo) ，[2.1.1 首页展示](https://www.remnote.com/doc/vLNb3MmWBQqR5teft) 和 [2.1.3 商品详情展示](https://www.remnote.com/doc/TWHuC43SCXSemBzRk)中，从需求方面列出了后台系统的组成部分和前台元素。

本节将以上架一个手机为例，介绍商家添加一个商品之后用户查看的全过程：

商家首先添加商品三级分类，比如手机的一级分类是电子设备，二级分类是移动设备，三级分类是手机。之后添加品牌以及和其业务相关的属性，比如华为，三星等，之后添加属性分组，比如手机的主体，基本信息，存储参数等。接着关联品牌和分组的属性，比如华为，三星需要被关联到手机分类下，主体，基本信息，存储参数等参数也要被关联到手机分类下。之后就可以添加属性了，分别添加基本属性，比如入网型号，上市年份和销售属性，比如内存，颜色，同时关联这些属性到属性分组中，比如入网型号，上市年份等需要被关联到基本信息中，而销售属性没有对应的属性分组。在添加完属性之后，就可以开始添加商品了，每个商品都会有对应分类，品牌，分组，属性等信息，之后就可以在发布商品供用户选择了。

用户点开的是商品首页，通过搜索栏或者点击三级分类可以跳转到搜索页面，之后可以选择了解商品详情。

### 3.2.2 库存模块

[在2.2.2 库存管理 中，已经分析过商家对于库存管理的需求，此处将介绍商家具体的操作流程：商家分别通过搜索仓库信息和库存信息来了解商品的库存情况，如果需要查询的部分没有库存则创建采购需求，到一定时间手动合并采购需求为采购单，之商家分配采购单给用户，用户通过领取采购单 来完成采购需求。](https://www.remnote.com/doc/kswvMwRHAstmeQict?aliasId=uymeWi5b5g8FP2Jb5)

### 3.2.3 搜索模块

搜索模块的流程比较简单，[为了完成商品搜索，首先商家要上架商品 ，在商品上架了之后，会员就可以进行数据检索](https://www.remnote.com/doc/o7yyYRFXig9c9ro7S) 。

Diagram

Description automatically generated

图3-2 功能模块和需求分析的对应

## 3.3 程序设计原则

在完成流程设计之后，我们要完成对程序实现的设计。由于程序等于算法加数据结构，我们首先要设计[数据结构](https://www.remnote.com/doc/guPJjqcBmzN56CgPq)，商品服务和库存服务使用的数据存储工具是数据库，包括Mysql和Redis，而搜索服务使用的数据存储工具是ElasticSearch。而在设计[后端程序](https://www.remnote.com/doc/yfPvPdqoNxsnzChs9)的时候，我们应该[将系统拆分成多个层次、每个层次有独立的职责，多个层次协同提供完整的功能。](https://www.remnote.com/doc/j7Nvth2yD5e8iW2cW?aliasId=7g5x5e7PwKS8AfZta) 从而[简化系统设计](https://www.remnote.com/doc/3D2cJSNz9d2qtiEMp?aliasId=7xMXhFEXzmnmcKB2h)，[提高程序的复用性](https://www.remnote.com/doc/rqEBqvcmPwj8v3t6v?aliasId=ziR3GJdwRBucMqXSL) ， 这些层次之间通过[不同的对象划分](https://www.remnote.com/doc/WxHehaZKZRrf9yeeo?aliasId=8heggu8xwh4TuTzoM)相互交互最后作用于数据结构。

### 3.3.1 [数据结构](https://www.remnote.com/doc/guPJjqcBmzN56CgPq)设计原则

在[1.3.2.3 数据存储技术](https://www.remnote.com/doc/CQX5nGfdM3dEMn8FC)中已经介绍了本文所使用的具体技术。[在计算机体系结构中，为了保持访问时间和费用的平衡，以金字塔结构存储数据，具体来说将数据依次在寄存器，缓存，内存 以及辅助存储器中。](https://www.remnote.com/doc/e4qQd5eiM6QYK6hDX?aliasId=fFfBPawNTKxYggeSK)类比此概念，本项目在设计上也使用了缓存的思想，使用Redis存储访问更加频繁的数据。最后，结合商城系统对[全文商品检索和文本分析](https://www.remnote.com/doc/X9HN3gf3cd8PYdAr9)的特殊需求，我们还使用了另一种缓存ElasticSearch来弥补Mysql和Redis [全文检索不够强大](https://www.remnote.com/doc/5kyxckGcJwr9WvC8A) 以及[数据分析支持不足](https://www.remnote.com/doc/BFNZqP4gumGTJGE7R)的特点。

关系型数据库(Mysql)

[在关系型数据库的设计上，首先是对于数据库中表和表中字段的设计。](https://www.remnote.com/doc/pfzG3qt3aZmBAefMb)

[对于数据库表的设计，我主要设计了两种表，分别是业务表和关联表，每一个业务表对应一种核心功能，围绕着主键和业务展开，当有多对多的关系的时候，我们可以使用分表，即建立关联表分别存储两个表的主键来扩展数据库。](https://www.remnote.com/doc/nAKramnPALX5WQw98)

[对于数据库中表的字段，从功能上分类，可以分为三个部分，最重要的是本表业务字段， 和本表业务紧密相连 ，接下来是外表关联字段，和其他表通过id关联主键信息，最后是外表业务字段，提供冗余字段从而减少查询的次数](https://www.remnote.com/doc/pmfcLiAeTmpZdLQ6r)。[诚然，我们可以参考数据库设计三大范式减少数据库的冗余信息。数据库的第三范式 要求字段没有冗余，即不存在传递依赖 。这样做虽然节省了存储空间，但是会导致多次的分布查询或者连表查询。然而现在的存储空间越来越便宜，电商系统对速度的要求却很高，因此没有冗余的数据库未必是最好的数据库。因此，在字段的设计上，反范式化操作是有必要的，因此为了解决电商系统对速度的高要求，我们设计了外表业务字段 。但是编程的时候需要维护，注意此字段的更新。此外，在查询时，根据电商系统数据量大的特点，我们使用分步查询而不是连表查询。因为其带来的笛卡尔积很危险。](https://www.remnote.com/doc/mk8xCuD5yLDxEfBXH)

[在表中数据类型的选择上，本项目主要使用了三种数据类型，分别是number， char ，以及datetime。 number 可以细分为Bigint，主要使用在id的存储，tinyint，主要使用在控制位布尔值的存储，和普通的 int，存放排序信息等其他数字。接下来是char 类型，本项目 主要使用varchar ，根据不同的业务场景设置不同的大小，在长度够用的情况下，越短越好，比如名字设置为varchar(200) ，地址设置为 varchar(500)，最后 datetime 常用来存储一些时间字段，比如付款表中的update\_time 和create\_time 字段](https://www.remnote.com/doc/FmZFmJg4EW9RdnzeK)

非关系性数据库(Redis)

[对于复杂的业务，已经不能够通过代码层面的优化和数据库层面的优化，我们一般都会将部分数据放入缓存中， 加速访问，从而增加吞吐量，这时就需要我们使用缓存。](https://www.remnote.com/doc/ss46x3YEP4P2qWyFr)[缓存设计关注的内容首先就是对放入缓存的数据的选择](https://www.remnote.com/doc/Mqvpjxidi94b6xFnL?aliasId=byTcv8cgLARdfFmtG)，本项目将[访问量大且更新频率不高的数据](https://www.remnote.com/doc/uWSCni4qwiXyd9mh2) 放入缓存，比如[商品分类](https://www.remnote.com/doc/g7ATFbFgxmN6Pyis7)，每个用户进入首页之后都会访问，但是一旦确定之后就几乎很少更新。

搜索引擎（ElasticSearch)

在ES的设计上，由于ES不擅长处理关联关系，分部查询的时间消耗太大，我选择使用了[宽表设计](https://www.remnote.com/doc/yAGLjfzfkCCBCYrEX) ，使用嵌套对象处理关联关系，从而提高读取性能，这也是数据库中结合现实情况[反范式化设计思想的充分体现](https://www.remnote.com/doc/H8XhDCHfLmdepFcXY) 。

### 3.3.2 [后端程序](https://www.remnote.com/doc/yfPvPdqoNxsnzChs9)设计原则

在[1.3.2.2 服务开发技术](https://www.remnote.com/doc/Mpy5nEWBvLhyfKXrH)中已经介绍了本文所使用的具体技术。

在每个独立的微服务中，我选择使用分层架构。即[将系统拆分成不同的层次、每个层次有独立的职责，多个层次协同提供完整的功能。](https://www.remnote.com/doc/j7Nvth2yD5e8iW2cW)

在确定使用分层架构后，我们可以定义不同层次之间的数据传输对象。比如我们使用[VO](https://www.remnote.com/doc/XXAPgRiRMTPKkutaQ)与前端进行交互，使用[DAO](https://www.remnote.com/doc/T8SJgbPjGWAtXcqdj)访问数据库 ，他们都可以称为[DTO](https://www.remnote.com/doc/daTzJrzazwrDayWxK) 。

程序层次的具体实现上，通过VO从前端获取所需要的数据，之后在[界面层](https://www.remnote.com/doc/ApQRC6BSHy5HYGQcq)由[Controller](https://www.remnote.com/doc/FbWEoi8vTnuDusKPd) 来[处理请求](https://www.remnote.com/doc/i3KSP6KJ2HXj7rkYM) ，[接受和校验数据](https://www.remnote.com/doc/D9RnPRPt6FC3zSARs) 。之后[业务逻辑层](https://www.remnote.com/doc/9cnQFnYP9BwS9yodS) 由[Service](https://www.remnote.com/doc/9LoY2CXiABz2GCYFd) [接收界面层的数据，进行业务处理](https://www.remnote.com/doc/BiuChkiiAW8zEoZ5Q)，之后[调用数据访问层 的方法](https://www.remnote.com/doc/itFcZCwBEfZ9YPT3e)。接下来在[数据访问层](https://www.remnote.com/doc/ivG6D2y2hsKfdrCrz)，[Dao完成对数据的增删改查](https://www.remnote.com/doc/YkB47eufYf8MtDWTn)。按照原路径传回，最后[Controller](https://www.remnote.com/doc/STEBovYcCC8eeTt6P) [接收Service处理完的数据，封装页面指定的VO](https://www.remnote.com/doc/4dzkDzj9Mg2fBvshB) 。

为简化开发，本项目还使用了[人人代码生成器](https://gitee.com/renrenio/renren-generator?aliasId=ruNBmCJsCxoQCLxGt)生成了简单的增删改查代码。

### 3.3.3 前端程序设计原则

在[1.3.1 前端技术综述](https://www.remnote.com/doc/zSm93sw3YEzLo89dK) 中已经介绍过了项目前端所使用的技术。

总的来说，前端程序实现不作为本项目的重点，因此UI部分借鉴了已有的代码，本项目重点完成数据展示和重要功能。

后台管理系统的前端系统我使用了[人人UI](https://gitee.com/renrenio/renren-ui?aliasId=6rqyL2qLhMret2TfG) 作为项目的脚手架，在其基础之上利用ElementUI的组件完成了三级分类树的实现。

客户端系统的前端的UI设计我借鉴了[京东商城首页](https://www.jd.com/?country=USA?aliasId=6NLzm2M8QbKNBGJFs)的UI设计，之后根据需求分析修改了部分代码以完成额外的功能与呈现数据。

### 3.3.4 服务治理设计原则

在实现完单个微服务后，还需要使用Nginx结合SpringCloud组件进行服务的治理。在[1.3.2.4 服务治理技术](https://www.remnote.com/doc/kaNYjGKnLxpbQFjYb)中具体介绍了本文使用的服务治理技术。从流程上来说，在实现上数据都注册进入Nacos，并配置好Sentinal和GateWay，就可以完成本项目的治理功能。

第四章 商城系统程序实现

本章将依据[3.3 程序设计原则](https://www.remnote.com/doc/4PB7rfzm6cMRYZDMY) 分别进行 [4.2.1 商品模块业务实现](https://www.remnote.com/doc/kPbTpNCN48zsMQp5y) 、[4.2.2 库存模块业务实现](https://www.remnote.com/doc/5KLCtgY87ZfmXaWrd) 、[4.2.3 搜索模块业务实现](https://www.remnote.com/doc/bCFgJgyQmFEqAiTMm) （包括 [数据结构](https://www.remnote.com/doc/guPJjqcBmzN56CgPq)以及[后端程序](https://www.remnote.com/doc/yfPvPdqoNxsnzChs9) ）以及[4.2.4 服务治理功能实现](https://www.remnote.com/doc/RHRGoKSjri9FoWKJS) ，来获得一个一体化的微服务系统。

## [4.1 商品模块业务实现](https://www.remnote.com/doc/nhjjJ3PacL74BZbn4?aliasId=2Wu8vYLBFA2dD7M3Q)

在本节，我首先介绍商品模块的[数据结构](https://www.remnote.com/doc/GDg9pyuvjLmtDu4s9) ，之后介绍其[后端程序](https://www.remnote.com/doc/gvtWtL3NzXQ9XrAmD) 。 以及[前端程序（后管）](https://www.remnote.com/doc/sC8Bu3Z3hJ3hWeTng) 和[前端程序（客户端）](https://www.remnote.com/doc/JaqWFvQqfmdLjCuiJ)。

### 4.1.1 [数据结构](https://www.remnote.com/doc/guPJjqcBmzN56CgPq)

**MySQL设计与实现**

[数据库中需要存储哪些字段是根据2.2.1 商品管理 中的分析决定的， 为了存储属性分类与属性分组 ，我设计了pms\_category 和pms\_attr\_group两个属性表 ，具体来说，包括 pms\_category 来存放分类，pms\_attr\_group 来存放分组。为了存储品牌信息，我设计了pms\_brand 属性表。 pms\_sku\_info 表与pms\_spu\_info 分别存放商品本身的元信息，即SPU与SKU。所有的平台属性都存储在pms\_attr 表中，通过一个字段来区分其是基础属性还是销售属性。针对多对多关系设计的关联表 ，以防止巨大的笛卡尔积 与增加可维护性。平台属性 通过 属性关联表 pms\_product\_attr\_value 和pms\_sku\_sale\_attr\_value 来与商品本身的元信息对应。另外通过pms\_attr\_attrgroup\_relation 与属性分组关联。pms\_attr\_attrgroup\_relation 使属性与属性分组关联， pms\_category\_brand 使属性与品牌关联。关于表中字段类型的选择在3.3.1 数据结构设计原则 中的关系型数据库(Mysql)部分有描述，下面会详细描述6张主要业务表和4张关联表的作用及各个字段的含义，最后结合ER图4-1描述各表之间的表间关系。](https://www.remnote.com/doc/9WfCMzQuEK5dMj4gE)

A screenshot of a computer

Description automatically generated with medium confidence

图4-1：商品模块ER图

[对于pms\_category 分类业务表，保存分类信息，只保存本表业务字段，包括主键cat\_id (primary) ，名称name，父分类ID parent\_cid，当前层次 cat\_level，是否显示状态 show\_status，当前排序sort，当前图标icon ，计量单位product\_unit，和商品数量product\_count](https://www.remnote.com/doc/7Y7xNLZ36KkHi37kT)

[对于pms\_attr\_group 分组业务表，保存分组信息，本表业务字段有主键attr\_group\_id，分组id，组名attr\_group\_name，排序字段sort，描述descript，组图标 icon 。](https://www.remnote.com/doc/97gJYpGAqpjq2oKoW)

[对于pms\_brand 品牌业务表，只保存 本表业务字段， 包括主键 brand\_id，属性id，属性名 name ，品牌logo logo，介绍descript， 是否显示状态show\_status，显示状态[0-不显示；1-显示]，检索首字母 first\_letter， 排序字段sort。](https://www.remnote.com/doc/2aqfBCjo6PdGhc3sb)

[对于 pms\_attr 平台属性业务表，保存所有的属性信息，本表业务字段有主键 attr\_id，分类名称attr\_name，检索位search\_type，是否需要检索[0-不需要，1-需要]， 属性类型位attr\_type， 属性类型 [0-销售属性，1-基本属性，2-既是销售属性又是基本属性] ， 属性图标icon，可选值列表value\_select，启用状态enable，快速展示show\_desc 。外表关联字段](https://www.remnote.com/doc/fHvAMmmbF4kJyHpZL)

[对于pms\_sku\_info sku信息业务表，本表业务字段有主键 sku\_id，sku名称sku\_name，sku描述 spu\_description ，sku图片 sku\_default\_img，sku标题 sku\_subtitle， 价格price，销售量sale\_count。外表关联字段 有spu\_id ，catalog\_id ，以及 brand\_id 。](https://www.remnote.com/doc/GWNfaZCzkzazhQehS)

[对于 pms\_spu\_info spu业务表，本表业务字段 有 商品id spu\_id ，商品名称 spu\_name ，商品描述spu\_description，重量 weight，上架状态 publish\_status，创建日期 create\_time，更新日期 update\_time 。外表关联字段有catalog\_id 和 brand\_id。](https://www.remnote.com/doc/3nH35BgbioYK3NPzk)

[对于pms\_sku\_sale\_attr\_value关联表 ，关联pms\_attr 和 pms\_sku\_info 表，本表业务字段 只有 主键id，外表关联字段 有sku\_id ， attr\_id ，，外表业务字段 有 attr\_name ，attr\_value ，attr\_sort。](https://www.remnote.com/doc/DHjiKuvM4TtTGq9e2)

[对于pms\_product\_attr\_value关联表 ，关联pms\_attr 和 pms\_spu\_info 表，本表业务字段只有 主键 id，外表关联字段 有spu\_id ， attr\_id ， attr\_name ，外表业务字段 有 attr\_value ，attr\_sort ，外表业务字段 有 attr\_name ，attr\_value，attr\_sort 和 quick\_show](https://www.remnote.com/doc/sDBn8anGThomgZm8X)

[对于 基本属性和属性分组 关联表 pms\_attr\_attrgroup\_relation ，本表业务字段 有主键 id，外表关联字段 有 attr\_id ，attr\_group\_id 。](https://www.remnote.com/doc/euuLHvGzCKN7dhRTa)

[对于品牌和分类关联表，本表业务字段 有主键 id，外表关联字段 brand\_id ，cat\_id (primary)](https://www.remnote.com/doc/PtrxsbsoNksGzmZd9)

**使用 MyBatisPlus**

使用Mybatis包括搭建环境，与使用MyBatisPlus增删查改两个部分。

搭建环境的步骤是，[在.pom中添加依赖](https://www.remnote.com/doc/2ZCD4SqQmnj2E353m?aliasId=oFrZG4ahxPLJ66Pwm) ，即从Maven中央仓库中引入Starter之后，再编写[主配置文件](https://www.remnote.com/doc/xjN9H9JLStAh3xzie?aliasId=yoiwq5oEP9Yewhipw)。主配置文件选择使用了xml的文件格式，其中比较重要的标签有 帮助我们配置数据库的环境，[为了配置环境，我们分别通过 标签来选择支持哪种类型的事物 ，以及 标签来配置我们想连接的数据源。](https://www.remnote.com/doc/DEiyqTkFCwnc4Lou4?aliasId=PsATKyfAyQppwzaF7)帮助我们使我们的文件更加简洁，以及 来配置配置映射文件的位置。

要使用Mybatis进行增删查改，我们就要使用[映射配置文件](https://www.remnote.com/doc/pRqHgadmmou86Rpu3?aliasId=K9eB2PguKWCpyE5uw)。为了使用此文件，首先我们要 [在数据访问层接口中创建一个抽象方法](https://www.remnote.com/doc/5Kcx4HraxPJfQGSuv)，其次我们通过[id](https://www.remnote.com/doc/b7btwhPWmbEc5dk53)来指定DAO对应的抽象方法，接着[在主配置文件的 标签中注册上](https://www.remnote.com/doc/CcBX84HJq3hDyrLt2)我们写好的方法，最后就可以[在测试文件中写测试方法来验证我们的SQL语句](https://www.remnote.com/doc/mHE9HB25asackeWpj?aliasId=Sx8kKadHZywZrmYQA)了。

**使用 Redis**

使用Redis包括安装Redis，配置Redis和访问Redis三个部分。

本项目在虚拟机的docker中安装Redis， 第一步是在 Docker Hub下载 Redis 的镜像，之后设置其映射端口和挂载目录，即可通过Socket来访问所安装的Redis

[要整合Redis，首先pom 文件引入 data-redis-starter，之后 yml 文件添加 redis 配置 。](https://www.remnote.com/doc/AK4vz3pzAwNkhiB7L) 具体来说，要[找到 SpringBoot中整合的 Redis Starter](https://www.remnote.com/doc/oEKeNa7GKFCNdtndW)，之后在[pom文件中引入依赖](https://www.remnote.com/doc/whPqN78XZpk3yRPF3) 。之后[配置 redis 属性，比如host和port](https://www.remnote.com/doc/qM8GTvfGrNNHYD4fH)。

之后就可以使用Springboot自动配置好的[RedisTemplate 和 StringRedisTemplate](https://www.remnote.com/doc/EJ3jcWeA3nZnmzuNJ) 进行数据库的访问操作了。

### 4.1.2 [后端程序](https://www.remnote.com/doc/yfPvPdqoNxsnzChs9)

在[2.2.1 商品管理](https://www.remnote.com/doc/sedSoY2rujC5YFzGo) 中我们已经分析出了商品模块所需要的功能，在[3.3.2 后端程序设计原则](https://www.remnote.com/doc/4zdhMn8BrR7EdGXwh)中，我已经阐述了本文后端代码层次的划分。在本节，由于篇幅所限，我将结合商品模块所需要的重点功能和本文后端代码层次划分阐述后端程序的实现。

[**属性分类和属性分组**](https://www.remnote.com/doc/f3csFz7xSvB93WAkp)

[**属性分类**](https://www.remnote.com/doc/pcccQGTGeptueMRQo)

[对三级分类来说，首先要完成 三级分类基础增删查改，之后为了方便管理，我还完成了 三级分类拖拽效果](https://www.remnote.com/doc/bT3fzefoqhn3r953d)

对于三级分类增删查改来说，后端较为重要的功能有查询时 [树形结构的获取](https://www.remnote.com/doc/Fbx76m5LqPv7BbvQS?aliasId=7FBtL9v9bfSFXNkzK) ，删除时的 [逻辑删除](https://www.remnote.com/doc/SYbzfhSdDxhXg5HGF?aliasId=9BehpY8cd3brqiaon)

在树形结构获取中，[在Controller层，CategoryController.java 中编辑list方法 。在Service层，CategoryService中添加listWithTree 之后在CategoryServiceImpl.java 添加listWithTree以及其辅助getChildren方法实现 。listWithTree方法获取所有分类以及子分类，并返回json树形结构。具体来说， 通过分类的parent\_cid来进行判断，如果是一级分类，其值为0。之后 使用递归的方法获得组装成树形结构的三级分类 。其实现逻辑为 首先查询所有分类，之后找到所有的一级分类，再通过getChildren 递归查找当前分类的所有子分类，找到之后将其设置当前分类的子分类。在getChildren 中， 递归的找传入分类的子分类，之后给子菜单排序 。最后，在CategoryEntity.java 添加字段children 存放所获得的数据](https://www.remnote.com/doc/myPY9XMgAcxDxCewo)

Text

Description automatically generated with medium confidence

图4-2：ListWithTree代码

在逻辑删除中，[在Controller层，在CategoryController添加removeMenuByIds 方法 ，之后在Service层，在CategoryServiceImpl添加removeMenuByIds 方法。但是一般为了保证安全性我们都使用逻辑删除。为了实现逻辑删除，首先我们要 在application.yml中配置全局逻辑删除规则 ，之后我们还需要在CategoryEntity的showStatus字段上加@TableLogic注解](https://www.remnote.com/doc/3DHNTryrRkku2fzo4)

对于拖拽功能来说，后端较为重要的功能有[批量拖拽之后批量保存](https://www.remnote.com/doc/evFGEExeGPZGMARiM?aliasId=b4XnwDXBZ88CrCMGh) 。实现上是 [Controller层，在CategoryController中添加 updateSort 方法 调用 categoryService的updateBatchById 方法](https://www.remnote.com/doc/QdpYhN37v5mQntvri)

最后，为了配合客户端实现分类的渲染，还需要后台接口分别[提供一级分类数据](https://www.remnote.com/doc/PkTgT8iTnJxi5KJdq?aliasId=pDGvnCpPbAQm3k38h) ，和[提供二级三级分类数据](https://www.remnote.com/doc/yx9v5CpDGNFbjFscr?aliasId=6tGHpjw8hKmNXdiPe) 。

要提供一级分类数据，[Controller层， 新建web包存放前端页面的Controller ，Service层，CategoryService中添加getTopCategorys方法之后CategoryServiceImpl中添加getTopCategorys方法实现。 此方法查询所有1级分类](https://www.remnote.com/doc/vFMFoSpw2XcoPjCZr) 要提供二级和三级分类，[首先VO方面 定义Catelog2Vo ，Controller层，IndexController中添加getCatalogJson方法 ，Service层，CategoryService中添加getCatalogJson方法 ，之后 CategoryServiceImpl中添加getCatalogJson方法实现 ，此方法先从数据库中查询所有的一级分类，根据一级分类的ID到数据库中找到对应的二级分类，之后根据二级分类的ID，到数据库中寻找到对应的三级分类](https://www.remnote.com/doc/4DDfJj2LNweyHtSi5)

[**属性分组**](https://www.remnote.com/doc/6py4efDkuHSKzXucr)

对于属性分类组来说，后端较为重要的功能有[提供修改时回显所属分类的id](https://www.remnote.com/doc/QXzHXnK4JtqYkFmXn?aliasId=BFKovnEBDyXkiiuB7) 。 [为了存储数据，首先要在AttrGroupEntity中新增属性 ，在Controller层， 修改AttrGroupController的info方法 ，添加接口 info，在Service层，CategoryServiceImpl中添加findCatelogPath的实现。递归逻辑类似listWithTree，区别是这次是从子分类找父分类，而之前是从父分类找子分类](https://www.remnote.com/doc/2HgAT9vgt6DqQFxz3)

关于[属性关联](https://www.remnote.com/doc/rFvcdyoGDGxmACruk)，[我们还要实现 平台属性 与属性分组的关联，其中包括查询分组已关联属性来回显，查询分组未关联的属性来新增分组与属性关联 。](https://www.remnote.com/doc/ykyxFBJvdbxmn2QME)

[**品牌**](https://www.remnote.com/doc/faBNjDyvrESKmNbwJ)

关于品牌，重点在处理其与分类的关系上，因此功能有[获取当前品牌关联的所有分类](https://www.remnote.com/doc/bhxEkQ4ahqEBjBBES?aliasId=KDGtLjMJ7uPKwnjqY) ，以及[新增品牌与分类关联关系](https://www.remnote.com/doc/mBw4n4J7imYmRxKHc?aliasId=5mTx2aCmduDzQZxrW) 。另外我们还需要注意[4.2.1 设计原则综述，关系型数据库小结中提到的冗余字段的更新](https://www.remnote.com/doc/qKeQSZLHMPdbpXSZc?aliasId=Dhch2QhXrZur37Gyb) 。

关于[获取当前品牌关联的所有分类](https://www.remnote.com/doc/bhxEkQ4ahqEBjBBES?aliasId=KDGtLjMJ7uPKwnjqY) ，[Controller层，在Category Brand Relation Controller中新增cateloglist方法 ，之后调用category Brand Relation Service 的 list 方法即可](https://www.remnote.com/doc/Hfk64WriTzaPouTK8)关于[新增品牌与分类关联关系](https://www.remnote.com/doc/mBw4n4J7imYmRxKHc?aliasId=5mTx2aCmduDzQZxrW) ， [在Controller层， 修改Category Brand Relation Controller.java 原生的save方法 。在Service层，Category Brand Relation Service新增saveDetail 之后 Category Brand Relation Service Impl中save Detail方法的实现。此方法根据传入的 CategoryBrandRelationEntity 存储信息到数据库。首先获取品牌和分类的id，之后查询品牌分类的详细名字，最后设置品牌分类的详细名字。](https://www.remnote.com/doc/SxJkGFccJApsYP48x)

关于冗余字段的更新，在实现[品牌修改](https://www.remnote.com/doc/dtmFyn3uTRKa9krnm) 或者[分类修改](https://www.remnote.com/doc/mhPS8bmEkjTnoE5fn) 的时候都需要进行冗余字段的更新。

为了实现品牌的级联更新，[在Controller层， 修改Brand Controller的update方法 。在Service层，在Brand Service 添加update Detail方法, 在Brand Service Impl添加updateDetail方法实现 。为了实现级联更新，还需要在Category Brand Relation Service添加updateBrand 方法，之后 Category Brand Relation ServiceImpl 实现updateBrand 方法 ，此方法根据传入的 brandId和name更新对应的Category Brand RelationEntity 。](https://www.remnote.com/doc/9M7nKQ66zpTarZFNf)

为了实现分类的品牌更新，[和品牌修改相似，在Controller层修改Category Controller中的update 方法。在在Service层，在CategoryService新增updateCascade 方法之后在CategoryServiceImpl添加updateCascade的实现。 Category Brand RelationService中添加updateCategory 之后CategoryBrandRelationServiceImpl中实现update Category 方法。 在DAO层，Category Br](https://www.remnote.com/doc/cTdnTcbmNBquaZoAk)[and Relation Dao中新增update Category 方法之后在](https://www.remnote.com/doc/cTdnTcbmNBquaZoAk)[CategoryBrandRelationDao.xml中完成SQL语句](https://www.remnote.com/doc/cTdnTcbmNBquaZoAk)

Text

Description automatically generated with low confidence

图4-3：CategoryBrandRelationDao.xml 中的SQL语句

[**平台属性**](https://www.remnote.com/doc/EDQnvky2joj6p4SiJ)

[关于属性，分别有基本属性 和 销售属性 的功能需要实现，但其拥有太多相似之处，因此有很多共用的接口。对于属性，也是要完成其增删查改。这里我会介绍规格参数的新增 ，规格参数列表查询，以及规格修改 。另外，我们还要实现 平台属性 与属性分组的关联，其中包括查询分组已关联属性 来回显，查询分组未关联的属性 来新增分组与属性关联 。](https://www.remnote.com/doc/rZhy33F3M4wTXxXxd)

[要完成 规格参数的新增，VO层面需要 新建vo包，封装AttrVo ，Controller层面，需要修改AttrController的save方法 ， 使其接收VO。在Service层，需要在AttrService中新增saveAttr ，之后AttrServiceImpl中方法实现 saveAttr 。这个方法帮助我们保存一个属性，在保存属性的时候，由于我们使用了分部查询，因此在保存商品的时候，不仅要保存(商品属性)基本数据，而且要保存(商品属性与属性分组)关联关系](https://www.remnote.com/doc/bdafofoXgNZM7C9tZ)。

[要实现规格修改，为了回显分类信息与分组信息，我们 还需要查询属性详情 之后再修改属性](https://www.remnote.com/doc/574AwXMDYcYG5CqXX) 。对于查询属性详情，[Controller层，修改AttrController.java 的info方法 ，Service层，需要完成AttrService的getAttrInfo方法 ，并且在AttrServiceImpl中完成getAttrInfo方法实现 。此方法是为了通过attrId获得AttrRespVo，首先通过attrId获得对应的attrEntity，之后还需要使用Attr Attrgroup RelationDao ，AttrGroupDao 和Category Dao查询属性对应的分组和分类。和 query Base AttrPage 不同的是 ，getAttrInfo 不需要通过首先查询条件获得page ，但getAttrInfo 需要回显所有的三级分类，需要额外使用categoryService的findCatelogPath 方法查询。](https://www.remnote.com/doc/fk5fxwyvSXsXdpyph) 对于修改属性，[Controller层，修改AttrController的update方法 。Service层，增加AttrService的updateAttr方法 ，以及完成AttrServiceImpl的updateAttr方法实现 。updateAttr方法接收一个 attrVo，根据此更新属性。步骤为首先修改自身属性，之后使用attrAttrgroupRelationDao修改分组关联关系。](https://www.remnote.com/doc/PQW4hc8QsvCugGJks)

[规格参数列表](https://www.remnote.com/doc/Tj7bgGdrumKXFtSAN?aliasId=ZXE5DsDcbCZ5mkkpN)查询，[Controller层，首先修改AttrController的baseAttrList方法 从而合并基本属性和销售属性的controller。在Service层，修改AttrService的queryBaseAttrPage方法 之后又修改queryBaseAttrPage方法实现 。此方法是为了查询平台属性分页， 需要返回AttrRespVo 类型的数据，因此该函数首先判断需要查询基本属性还是销售属性，之后根据限制构造分页条件，之后还需要使用AttrAttrgroupRelationDao ，AttrGroupDao 和CategoryDao查询属性对应的分组和分类。为了代码的整洁，我实现了属性类型枚举 ，再 修改AttrServiceImpl 使其使用 AttrEnum](https://www.remnote.com/doc/r7Keg6Rf3Tye82q2J)

最后，为了配合客户端完成[sku组合切换](https://www.remnote.com/doc/ZZi4BqCMhxCw3dfe8?aliasId=KWJ9dgaZ66Pc9bDi4)，后端还需要[提供搜索属性对应的商品](https://www.remnote.com/doc/eAaCDvM8jB7spKwnP)。[VO层，修改SkuItemSaleAttrVo，添加AttrValueWithSkuIdVo类型的属性。DAO层，修改获得分组属性的mapper文件](https://www.remnote.com/doc/ogML2YqZsFbjLXCXS)

**[商品本身的元信息](https://www.remnote.com/doc/9f8zA9b6LEySuFWTd)**

[关于商品，我们首先要通过新增商品 获得商品，之后我们可以进行商品管理](https://www.remnote.com/doc/zzfBb9t8F75DiGffh) 。[对于新增商品，还有获取分类关联的信息 和商品本身新增两个步骤](https://www.remnote.com/doc/WRWEENoaFATH74PEA)

[获取分类关联的信息 包括 获取分类关联的品牌 和 获取分类下所有分组以及属性 。](https://www.remnote.com/doc/gx39EGrvraDTvtKWT) [为了 获取分类关联的品牌 VO方面要在vo包中添加BrandVo ，Controller层Category Brand Relation Controller添加relation BrandsList方法 ，Service层，Category Brand Relation Service添加get Brands ByCatId方法 ，最后Category Brand Relation ServiceImpl中实现getBrands ByCatId方法](https://www.remnote.com/doc/sukh8DvLwqHXNysaP) [为了 获取分类下所有分组以及属性 ，VO方面要新建AttrGroup WithAttrsVo ，Controller层，Attr Group Controller中添加getAttr Group WithAttrs ，Service层，Attr GroupService中添加getAttr Group With AttrsBy CatelogId，最后Attr Group ServiceImpl中getAttr Group With ByCatelogId方法实现](https://www.remnote.com/doc/aiGksriKdhLnrDsX3)

为了完成商品本身新增 [VO方面，抽取与被新增商品相关的VO。其中最重要的是 SpuSaveVo.java ，其中包含Skus.java ，以及一些其他抽取出来的属性，比如 BaseAttrs ，Images 等 。之后就是分别保存商品的SPU和SKU信息。为了 保存SPU基本信息 ，在Controller层，修改SpuInfoController的save方法 ，之后 在SpuInfoService中添加saveSpuInfo方法 ，对于saveSpuInfo方法，首先使用SpuInfoServiceImpl中的saveBaseSpuInfo方法实现保存spu基本信息 ，之后分别使用SpuInfoDescServiceImpl中的saveSpuInfoDesc方法 和 SpuImagesServiceImpl中的saveImages方法 保存spu的描述图片 和 保存spu的图片集 ，最后使用ProductAttrValueServiceImpl中的saveProductAttr方法 保存spu的规格参数 。至此，SPU的基本信息就已经保存完成，接下来需要保存当前spu对应的所有sku信息 为了保存sku的基本信息 ，我在SkuInfoService添加saveSkuInfo方法 ，之后 在SkuInfoServiceImpl添加了saveSkuInfo方法实现 ，和保存SPU相似的逻辑分别保存图片信息以及销售属性等。](https://www.remnote.com/doc/RGxBD55hepkaJvbYe)

对于商品管理，[分别需要实现 SPU检索 ， SKU检索和SPU规格维护 。](https://www.remnote.com/doc/sTv3o5FxD4kjtDJTC)检索功能不再赘述，[对于SPU规格维护 分为两个步骤，首先要获取SPU规格，之后修改商品规格。要获取SPU规格，Controller层AttrController中添加baseAttrListForSPU方法 ，之后ProductAttrValueService中添加baseAttrListForSPU方法，最后 4.ProductAttrValueServiceImpl中添加baseAttrListForSPU方法实现 。 要修改商品规格，Controller层，AttrController中添加updateSpuAttr方法，Service层在ProductAttrValueServiceImpl中添加updateSpuAttr方法实现 。 此方法接受需要更新的spuId和ProductAttrValueEntity List来更新数据库里的数据。首先要删除这个spuId之前对应的所有属性，之后设置要更新的属性对应的spuId，最后批量保存所获得的productAttrValueEntityList 。](https://www.remnote.com/doc/pSXiYiufzTbMFxQkn)

最后我们还需要为客户端提供商品详情接口。 [为了获得销售属性的组合，Controller层，添加skuItem 方法 。 在Service层，添加SkuSaleAttrValueService ， SkuSaleAttrValueServiceImpl 获得spu的所有销售属性组合 ，最后使用 SkuInfoServiceImpl 中的 queryItemById 获得属性详情。DAO层里面有 SkuSaleAttrValueDao.java 来支持Service 。](https://www.remnote.com/doc/W8EwapmTwu6Wx4MXh)[为了加速查询一个商品详情的速度，我们可以使用异步编排优化queryItemById 方法。页面的信息需要从各个微服务里面查询，如果使用单线程查询，会导致页面加载的很慢。因此这里选择使用多线程Future，每访问一个微服务开一个线程，提高访问速率。为了使用异步编排， 首先要配置线程池 ，包括使用MyThreadConfig 加入线程池，再用ThreadPoolConfigProperties 配置其属性。之后在SkuInfoServiceImpl中修改queryItemById方法。 queryItemById 根据需求来异步编排，这里我的第一步sku基本信息,第二部sku的图片信息互不影响，然后3，4，5步必须要获取第一步的返回值后才能执行，然后方法需要在这5个任务都完成之后才返回值。](https://www.remnote.com/doc/wEFBzkPXiajpyoCBE)



图4-4：异步编排代码

### 4.1.3 前端程序（后管）

[**属性分类 和 属性分组**](https://www.remnote.com/doc/f3csFz7xSvB93WAkp)

**属性分类**

前端程序中，后管部分最主要的是三级分类的展示，新增和删除

[要树形展示三级分类数据， 在Script中需要修改category.vue中的getMenus 方法，向后台发送请求获得数据，把解构出来的数据赋值给menus 。修改defaultProps的属性，添加children 属性 ，并绑定template中的props属性](https://www.remnote.com/doc/wWDaxrxxD5rZugcNa)

[要新增一个三级分类，在Template中，首先 模板中添加分类对话框，在Script中， 在data中加入对话框所需要的数据 ，之后添加append 方法对应更多添加按钮的细节， 之后使用addCategory 发送post请求 。还要注意在确定之后自动关闭对话框，刷新出新的菜单以及默认展开添加后的菜单](https://www.remnote.com/doc/eHLvfmnjdQnQN39Aa)

[修改分类与新增分类 分类共用同一对话框，只是需要在添加按钮后添加修改按钮 以及 在remove方法后加上修改方法 。为了进行对话框类型的划分 ，submitData方法 根据data中新增 调用 addCategory 代码 还是 editCategory 代码 。之后 在data中新增 对话框类型 dialogType 和话框提示信息 title区分对话框所需要的信息，之后submitData方法 根据data中新增 调用 addCategory 代码 还是 editCategory 代码 来发送请求修改数据](https://www.remnote.com/doc/LCou9EvNqa2BHBFkn)

[关于删除的页面效果，我实现了 节点后面添加按钮 ，取消展开合并 ，依据分类层级显示对应按钮 ，以及为节点添加选择框 。之后，完善了 删除后刷新页面 ，删除之前消息确认弹框 ，删除之后结果消息提示 ，以及删除成功后展示之前的父节点 等细节问题 。最后，添加了 批量删除按钮 和 批量删除方法](https://www.remnote.com/doc/y5HYkb9tJHWdG9cGz)

最后，是三级分类的拖拽效果。[为了完成基础的拖拽效果，在Template 中，需要 在el-tree中加入draggable属性开启拖拽功能，之后在Script中，data 中加入this.maxLevel属性 代表当前拖拽节点的层级， 编写allowDrop()函数，以及其辅助函数countNodeLevel 限制可拖拽范围 ，防止四级菜单的出现，之后 在中绑定allow-drop属性绑定编写好的allowDrop()函数 。为了进行 拖拽节点数据收集 ，Template 中需要 在el-tree 的 node-drop 属性绑定handleDrop 方法 再 增加开关控制拖拽data中新增的draggable。 Script 中， data中加入要修改的节点和要展开的父节点 之后完成 handleDrop 方法。 最后，我还通过开关按钮控制拖拽功能的开启与批量保存](https://www.remnote.com/doc/WdpMCAfXesNyS8Mfh)

**属性分组**

属性分组中，[在前端 通过公共组件抽取复用三级分类组件 ，之后完成了查询， 新增， 修改 等功能。在新增 的时候，我使用了ElementUI 的Cascader 组件 ，并实现getCategorys 函数完成 级联选择器数据的获取](https://www.remnote.com/doc/Nu2pTgrgBCjqw7cnu) 。

[**品牌**](https://www.remnote.com/doc/faBNjDyvrESKmNbwJ)

[在完成品牌的后管时， Template 层面我进行了一些页面优化 ，比如 修改brand-add-or-update.vue 中form的显示状态为switch开关，并给开关绑定change事件。以及 修改编辑页面的宽度 。添加updateBrandStatus方法](https://www.remnote.com/doc/KeKFWRW3tzpTNMw77)

### 4.1.4 前端程序（客户端）

商品模块的客户端，包括商品主页和详情页两部分。

[商城主页的重点内容是 渲染一级分类数据 和渲染二级三级分类数据](https://www.remnote.com/doc/J2FG4xBPcc7eLM34G) 。[在4.2.2后端程序，属性分类的最后一小节介绍了后台接口的实现，这里介绍前台数据的渲染](https://www.remnote.com/doc/BAmiXiTP6gRwkjYQJ?aliasId=GsuSXRNoefAan7dbw) 。要渲染一级分类数据，我们要在前台使用[th:each 遍历后台数据获取一级分类](https://www.remnote.com/doc/NkrRapYf9nTRGST7c)。 要渲染二级分类和三级分类的数据，我实现了后台的[getCatalogJson](https://www.remnote.com/doc/BnBMvtA6sT3emisrt) 方法返回json类型的数据。

[商城详情页的内容包括了详情页渲染 ， 销售属性渲染 之后点击 sku组合切换 。](https://www.remnote.com/doc/fKWaZyx33R78h8krj) [4.2.2 后端程序，商品本身的元信息的最后一小节，描述了此处的后台接口](https://www.remnote.com/doc/B7XeAf9pccWjTuLiM?aliasId=TkqEfzeGEgane3HWs)。 [详情页要渲染的数据有 设置商品细节， 商品介绍， 和规格包装。商品细节又包括 标题和副标题，商品图片，商品价格，和是否有货](https://www.remnote.com/doc/3QKFuHchZtzs4RWhM) 。基本都是通过thymeleaf与后端相应数据绑定实现，此处不再赘述

在[4.2.2后端程序，平台属性的最后一小节](https://www.remnote.com/doc/zXJE8jmbjvqSE9MHH?aliasId=FQHKtx3NCQbMnAY8s) 已经描述了此处后端需要额外完成的内容。[在销售属性渲染中前端要完成的功能是 获取到所有加了 checked 的属性（即获取当前的skus组合，再获取到最后一个选中的属性的组合），取出他们的交集，得到skuId。](https://www.remnote.com/doc/pXGvJFnTwBoLeYZSD)

## [4.2 库存模块业务实现](https://www.remnote.com/doc/4fWRaL8qnLdP6ehWu?aliasId=yXHRirxwnqSeQFptn)

### 4.2.1 [数据结构](https://www.remnote.com/doc/guPJjqcBmzN56CgPq)

**MySQL设计与实现**

[仓储系统数据库的设计，包括vms\_ware\_info 存储仓库信息， vms\_ware\_sku 存储仓库的商品信息， vms\_purchase 和 vms\_purchase\_detail 存储采购需求和其生成的采购单信息。关于表中字段类型的选择在3.3.1 数据结构设计原则 中的关系型数据库(Mysql)部分有描述，下面会详细描述各表的作用及各个字段的含义，最后结合ER图描述各表之间的表间关系。](https://www.remnote.com/doc/fPgCkrfWL9NTNuRkT)

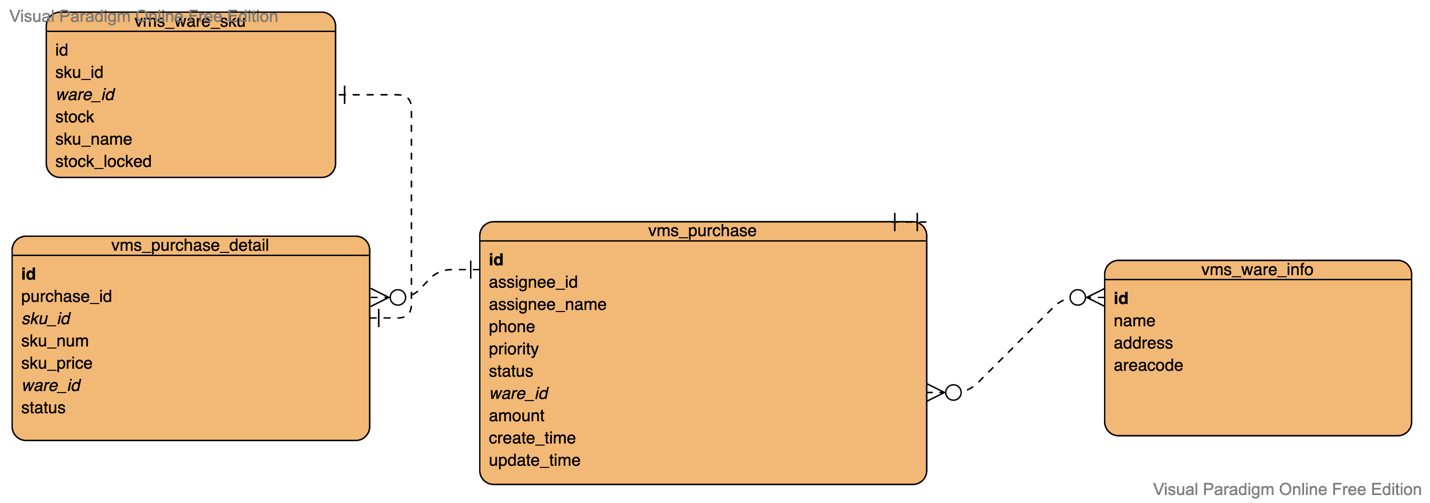


图4-5：仓储模块ER图

[对于vms\_ware\_info 仓库信息业务表，只保存 本表业务字段 主键id ，仓库名称 name](https://www.remnote.com/doc/tXMaz7e6Tdw5YHmk2) [对于 vms\_ware\_sku 关联表 ，保存每一个仓库的商品保存了几件，库存锁定了几件的信息，保存了本表业务字段 主键id， 外表关联字段 sku\_id 和 ware\_id ，以及 外表业务字段 sku\_name](https://www.remnote.com/doc/KwuDNwMSRF6sMPjqq) [对于 vms\_purchase 采购单 业务表，存放采购需求， 本表业务字段 有主键 id ，采购人id assignee\_id ，采购人名 assignee\_name， 联系方式 phone ，优先级信息 priority ，付款状态status， 总金额amount， 创建时间create\_time ，更新时间update\_time。外表关联字段 有 ware\_id](https://www.remnote.com/doc/x54QSJQ3QuRjhm3Bi) [对于采购项业务表vms\_purchase\_detail ，本表业务字段 有主键 id， 采购数量sku\_num， 采购金额 sku\_price。 外表关联字段 有 purchase\_id ， sku\_id ，ware\_id 。](https://www.remnote.com/doc/qWjBt7YREreDKR7ZZ)

在[3.3.1 数据结构设计原则](https://www.remnote.com/doc/qKeQSZLHMPdbpXSZc)中具体介绍了本文使用的数据存储技术MySQL，Redis和ES的设计，由于ES的使用在搜索模块已经详细介绍过，本章将详细描述MyBatisPlus与Redis的使用过程。

### 4.2.2 [后端程序](https://www.remnote.com/doc/yfPvPdqoNxsnzChs9)

在[3.2.2 库存模块](https://www.remnote.com/doc/4fWRaL8qnLdP6ehWu) 中我们已经分析过库存模块需要的功能，在[3.3.2 后端程序设计原则](https://www.remnote.com/doc/4zdhMn8BrR7EdGXwh) 中，我已经阐述了本文层次的划分。在本节，由于篇幅所限，我将结合库存模块所需要的重点功能和本文层次划分阐述后端程序的实现。

库存模块的重点功能有[查询库存](https://www.remnote.com/doc/L7HHmuZMYzQBGPnGY?aliasId=KGiAayfjFxj6f4ZRw)， [合并采购需求](https://www.remnote.com/doc/MaXS6Zp8psgZgh87d?aliasId=Wq8PGy46y35i3wtmw) 。

对于**查询库存**，[总的来说，我们一共要进行三种查询，分别是分1.库存的查询 ， 2.采购需求的查询 和 3.未领取的采购单的查询 。](https://www.remnote.com/doc/zAYgsAQKddtLfaGR2) 由于其逻辑较为相似，都是通过前端传来的参数，构造Query Wrapper传入分页工具。我选择对[未领取的采购单的查询](https://www.remnote.com/doc/aMFucHQXRKLYSdhmH?aliasId=gkLq3WyKWyMJfL3Ye) 进行介绍。[要查询未领取的采购单，在Controller层中的 PurchaseController 中添加 unreceiveList 方法， Service层中PurchaseService 添加 queryPageUnreceivePurchase 方法 ，最后Purchase ServiceImpl 中 queryPage Unreceive Purchase 方法实现](https://www.remnote.com/doc/eoL9Sz6dKAyAKFPE7) 。在[queryPage UnreceivePurchase](https://www.remnote.com/doc/Lvjnqor6mumdeFWx6) 方法中，[接受分页参数采购单状态，之后查询新建的和已分配的购物单](https://www.remnote.com/doc/Bbgi974vWW9rSCeXM)。

对于**合并采购需求**，[在VO方面我抽取了数据结构MergeVo，在Controller层 Purchase Controller 中添加 merge 接口，在Service层，Purchase Service 中添加 mergePurchase 方法，最后Purchase ServiceImpl 中 mergePurchase 方法实现 ，为了代码的整洁，我还规定了库存系统常量 。](https://www.remnote.com/doc/Nsqvznk3MdPKGzh5J) 对于[mergePurchase](https://www.remnote.com/doc/N72X8arRobRK8FFeu) 方法，[此方法首先 新建一个Purchase Entity，之后为此purchaseEntity设置默认值，按照要求合并之后使用purchase DetailService更新](https://www.remnote.com/doc/NtpCaGCKTqEuQizMb)

## [4.3 搜索模块业务实现](https://www.remnote.com/doc/pH2E9uihbNwo93967?aliasId=DsmmGYm4fikEMz4b9)

### 4.3.1 [数据结构](https://www.remnote.com/doc/guPJjqcBmzN56CgPq?aliasId=ZQ3PTkFF8L5MW88Ey)

**ES 设计与实现**

[要设计搜索模块的数据结构，我们要考虑的内容是存储商品的什么信息到 es 中 以及如何存储商品的信息 。](https://www.remnote.com/doc/9ChXRxcYfXEjGY7Km)

[一般来说， 由于ES存储在内存中，容量有限，只保存需要使用的数据，比如 只保存sku\_id ，如果需要全部图片或者完整介绍，去数据库检索即可。因此，在实现的时候，我选取了以下3.1.2节会员检索商品部分 中分析过的检索需求所必须的字段： 为了过滤检索 ，我选择存储品牌，分类，以及基本属性 ，为了全文检索 ，我选择存储sku 的标题 sku\_title ，为了排序检索 ，我选择存储 sale\_count和price 。 最后，由于很多sku拥有相同的基本属性 ，我们可以通过链接spu\_id 检索共有这些规格的商品 。另外，我们还需要存储一些冗余字段来方便前端页面的展示，比如品牌的名称和图片等。](https://www.remnote.com/doc/NFChH2M3HxuonoMfb)

[在设计ES中存储的数据模型的时候，我们要做好时间和空间的平衡，如果我们时间换空间。只保存sku的有用信息，通过连表查询其spu的 基本属性 。假设我们检索关键字“手机”，对应的spu有一万条，我们需要将这一万条 spu\_id 都发出去，从而获得这些spu的所有关联属性进行分析，发送spu的spu\_id数据传输一次就10000\*4=4MB，并发情况下如果有 1000 条检索请求，就会发送 4GB 的数据。传输阻塞时间会导致业务无法继续。 如果我们选择使用空间换时间，即 将 sku 的全量信息保存到 es 中 ，虽然会由于对于相同对应相同spu的sku, attrs 属性字段会重复产生很多冗余字段 ，假设我们有100万个sku数据，每个数据对应20个冗余字段，一共对应2KB的数据，最终会产生2G的额外开销。在内存越来越便宜的背景下，其实是可以接受的。](https://www.remnote.com/doc/RGkDEZhrESRQm6oWk)

在具体实现上，我们首先通过ES 中的[Mapping](https://www.remnote.com/doc/z3dAqM52qAuWBBxEc) 在ES中规定数据类型，之后在后台系统创建出[SkuEsModel](https://www.remnote.com/doc/ssQtZEwEPvChtEb6Z)

在设计Mapping时，对于[存储内容分析最后提到的冗余存储用来展示的字段](https://www.remnote.com/doc/XWH5TAZ9R3ZHiGfAa?aliasId=2HuHSeYpXzxA6oaRy)，我们要设置[index](https://www.remnote.com/doc/xqtEMNLaH2QjQ8gJK) 为[false](https://www.remnote.com/doc/QxmFqTNoatEH8yQqX)从而使[属性不能用来被全文检索](https://www.remnote.com/doc/gBA4jpMdDbgxJk8DT)，设置 [doc\_values](https://www.remnote.com/doc/WY3YGhQLza5F9bjxp) 为[false](https://www.remnote.com/doc/QEZyTBsmwQYofavvu) 来 [避免排序，数据分析等操作，从而更节省磁盘空间](https://www.remnote.com/doc/wdrAyGkh5gvw4bbBM) 。最后我们还对需要[精确搜索](https://www.remnote.com/doc/S6xekggoTfypL8g3a) 的字段设置为类型 [keyword](https://www.remnote.com/doc/4PEQ4MT24ywABZjaA) ，比如[skuPrice](https://www.remnote.com/doc/2RAKWASdbBXRnnyTD)，从而 [解决精度问题](https://www.remnote.com/doc/DGuDQjoMi5JWzLxG3) 。因此设计好的mapping如下：

### 4.3.2 [后端程序](https://www.remnote.com/doc/yfPvPdqoNxsnzChs9)

在[3.2.3 搜索模块](https://www.remnote.com/doc/pH2E9uihbNwo93967)中我们已经分析过库存模块需要的功能在[3.3.2 后端程序设计原则](https://www.remnote.com/doc/4zdhMn8BrR7EdGXwh) 中，我已经阐述了本文层次的划分。在本节，由于篇幅所限，我将结合搜索模块所需要的重点功能和本文层次划分阐述后端程序的实现。

搜索部分分为商品上架和上架后的搜索两个部分。

**商品上架**

商品上架需要首先设计其存储对象，[首先考虑复用之前商品服务的SpuInfoEntity.java ，但它只能表示一个Spu的信息，与索引的数据模型（刚刚构建好的Mapping）并不对应，所以我们要建立专门的vo进行数据传输。](https://www.remnote.com/doc/cQsJE7kFbGmmiWYpf) ­

[要在后台完成商品的上架，VO层面， 首先要构造基本数据 ， sku在ES中模型对应的vo 即SkuEsModel。 Controller层，SpuInfoController中添加spuUp方法 ， Service层，SpuInfoService中添加up方法 之后在SpuInfoServiceImpl.java 中实现up方法 。具体来说， 前端点击上架后，发送请求并带上参数 spuid, 根据商品id去属性值表查询此商品所有属性值，并取出id集合再 根据id集合去属性表查询，这些属性可以检索的id的集合, 根据 pms\_attr 查询到检索属性后，用检索属性和 原先根据 spuid 查询 pms\_sku\_info 表得到商品相关属性进行比较，pms\_sku\_info 包含 从 pms\_attr 字段attr\_id 则数据保存否则过滤, 根据 skuIds 去查询远程仓库中是否有库存 , 之后远程调用search 服务 productStatusUp函数 , 组装数据，设置SkuEsModel后将数据发给es进行保存 ，修改当前spu的上架状态 。](https://www.remnote.com/doc/4L4xTMYqxhT52h8QX)

**商品搜索**

[后端要抽取构建DSL所要的VO ，之后使用elasticsearch包动态构建 SearchRequest](https://www.remnote.com/doc/S9hX3var2cpHMZKfD)

[关于第一步，明确查询和返回模型。新建VO类，保存所需参数。 查询参数中需要根据3.1.2 商品搜索中会员检索商品 的分析，封装页面所有可能传递过来的查询条件，因此我在VO包下添加 SearchParam。 在VO包添加SearchResult 返回模型，返回模型中比较重要的功能是聚合，即后台管理系统按照品牌 ，分类 以及其对应的基本属性 聚合 以提供筛选条件 。](https://www.remnote.com/doc/pkJPncSgiqaHgyarC)

Text

Description automatically generated

图4-6: SearchRequest 方法

在构建Search Request的时候，[Controller层，实现SearchController.java。 Service层，实现MallSearchService.java 和在MallSearchServiceImpl.java 中添加search 方法 完成Search动作，包括使用buildSearchRequest，准备检索请求动态构建检索需要的DSL语句，执行检索请求，使用buildSearchResponse分析响应数据，封装成需要的格式](https://www.remnote.com/doc/e3fGd8WJD9ZyexA53) 。

[buildSearchRequest函数负责创建构建DSL检索条件对象。之后利用建造者模式向QueryBuilders创建各种条件，比如用must语句分词匹配skuTitle，用filter按照分类，品牌，属性过滤，操作属性的时候需要使用nested语句。最后加上排序，分页，高亮 ，以及聚合分析](https://www.remnote.com/doc/iN4wRsBC8wjE5sZQ9) [聚合分析包括聚合品牌、分类、以及规格聚合，规格聚合内部又包含了按照规格名字和id分组的子聚合。至此buildSearchRequest已经完成。](https://www.remnote.com/doc/LcdpQgAoKXyqKRodY)

[buildSearchResult 函数负责封装ES response返回的信息为SearchResult。步骤为返回所有查询到的商品，提取当前所有商品涉及到的所有属性信息，包括ID，名字，图片。提取当前所有商品涉及到的所有品牌信息与分类信息，最后提取分页信息。](https://www.remnote.com/doc/o2cLwfhaA5BiE5M4X)

### 4.3.3 前端程序

在上一节测试好检索与返回商品数据之后还需要在前端进行数据的展示。[搜索服务的前端程序包括三个部分， 页面基本功能，页面排序功能，以及 面包屑导航](https://www.remnote.com/doc/dfF6P6WiSffFcarYb)

**页面基本功能显示**

[包括 页面基本数据渲染， 页面筛选条件渲染，以及 页面分页数据渲染](https://www.remnote.com/doc/ij7Hy3iFg7yDiwRdw)

页面基本属性渲染分为[右侧商品数据渲染](https://www.remnote.com/doc/GCW5Yi8SiFqzcPg6E) 和[品牌及分类及属性的渲染](https://www.remnote.com/doc/fftSin6nBLuAPnMhP) 。主要是视图部分的实现。在渲染商品的时候 [要用th:utext防止之前的高亮不能渲染](https://www.remnote.com/doc/DputDePfDuRhbw4D7) ，在渲染品牌的时候[th:each遍历之后从brand对象中分别取出img, name](https://www.remnote.com/doc/GxWxpy9Kib4v6yejh?aliasId=Pn2Zo7sH8eNwvnMiJ) 。

页面筛选条件需要实现的功能是[当我点击某分类时能筛选出该分类的产品,属性也是同理](https://www.remnote.com/doc/ceitkwuJTmR8xteqi)。 因此在方法部分，[修改list.html ，添加searchProducts 方法](https://www.remnote.com/doc/E4BKzgW4wPQRakXqH) ，[searchProducts 是js方法，传入参数名和参数的值，在原来的页面上添加参数，要注意如果前面一个参数都没有，就要以?开始](https://www.remnote.com/doc/QmtsJw7i76QMQ6TxY) 。在视图部分，[在点击位置调用searchProducts](https://www.remnote.com/doc/AdgmnGnYkvkNo7oNd) ，包括[修改分类超链接](https://www.remnote.com/doc/4sbZPy2K9fLCvvQRw)， [修改品牌超链接](https://www.remnote.com/doc/zM6EKjNmxR2HCRiBH) ，和 [修改属性超链接](https://www.remnote.com/doc/2u8jcocn7AP7bpFir) 。

页面分页搜索，在方法部分需要添加[js方法，给page\_a绑定事件](https://www.remnote.com/doc/mBt7KFpro88aoso5R)，并调用[replaceAndAddParamVal](https://www.remnote.com/doc/BzPeypenTnDhezvEa) 修改URL。在视图部分，实现的功能有[上一页在不是第一页的时候显示，下一页在不是最后一页的时候显示](https://www.remnote.com/doc/fuRka9t2fWu2oA5jA)，[给上一页和下一页添加自定义属性，跳转的时候使用](https://www.remnote.com/doc/9Z9hGNMrQJGGiscvb) ，并[遍历修改后端代码 ，将各个页码遍历显示](https://www.remnote.com/doc/KbPZAcc72QDuPQ3xg)

**页面排序功能**

[为了实现页面排序功能，首先要 sort\_a 标签的单击事件，使得点击切换样式并且跳转指定位置，具体来说，使用changeStyle 函数， 没被点击的恢复默认样式，被点击的修改样式。使用 replaceAndAddParamVal 函数，跳转到指定位置，具体来说，此函数 添加的逻辑类似searchProducts ，如果URL中原来没有参数就添加，有就替换](https://www.remnote.com/doc/Rr3bzGjFvFqq7bG8Y)

**面包屑导航功能**

在此处我们使用[属性型](https://www.remnote.com/doc/32DzBkjAzyaXjPkdQ) 面包屑导航，目的是为了[体现网站的架构层级，能够帮助用户快速学习和了解网站内容和组织方式](https://www.remnote.com/doc/DQMDyN8sTcno3Ztti?aliasId=WutQBA8nWWMymSZ7A) 。后端将数据准备好之后前端负责属性和品牌面包屑导航的渲染。[在视图层面 使用th:each 遍历后端传来的面包屑，在方法层面 在选属性的时候，使用replaceAndAddParamVal方法， 注意这里对于属性，我们不能进行replace操作，而要进行add操作，因此添加了forceAdd字段](https://www.remnote.com/doc/bKZt66vQpRAkJYn8u)。另外在[选中内容以后就从筛选条件中去除](https://www.remnote.com/doc/EW5eaMjkRGpMvyykX)。在视图层实现的时候[如果路径中已经存在就不再显示了。具体来说先用th:with声明变量，再用th:if进行判断是否已经存在](https://www.remnote.com/doc/TJaZa4JNFvqnpo2pa)

## 4.4 服务治理功能实现

在[3.3.4 服务治理设计原则](https://www.remnote.com/doc/AqrMzNeibunssZoQr) 中介绍了本文服务治理的设计，本章将详细描述其实现，包括[4.2.4.1 使用Spring Cloud组件](https://www.remnote.com/doc/52Jz3Gj5nrSsfEND3) 和[4.2.4.2 使用Nginx](https://www.remnote.com/doc/r6mGtHAf5AfALZspL)。

### 4.4.1 使用Spring Cloud组件

**使用Nacos**

使用Nacos包括启动[Nacos Server](https://www.remnote.com/doc/xQu2xjbM9HvoJ52iL) 和 [将微服务注册到nacos中](https://www.remnote.com/doc/kMxmGWQNjQXXa6suJ) 两个部分。

本项目在虚拟机的docker中启动Nacos Server， 第一步是在 Docker Hub下载 Nacos 的镜像，之后[访问连接：http://192.168.56.10:8848/nacos/](https://www.remnote.com/doc/tuRs7KzeYCZCaebBk?aliasId=cSAGm2F9XxbcLAqWx) （此处假设192.168.56.10位虚拟机的地址），在浏览器[输入用户名和密码](https://www.remnote.com/doc/Zs8JjPYiJvuLC9YCF) 即可访问Nacos。

为了[将微服务注册到nacos中](https://www.remnote.com/doc/kMxmGWQNjQXXa6suJ)，第一步为[引入依赖](https://www.remnote.com/doc/aK87LYCPxoJCafh3v)，需要分别[引入Spring Cloud Alibaba全局依赖](https://www.remnote.com/doc/BJxD7NMAqgdYPjR8q)和[引入 Nacos Discovery Starter依赖](https://www.remnote.com/doc/GCE3ZCbTDMHxmrodH)。第二步位[写配置](https://www.remnote.com/doc/58ssRveA58prxeYTZ)，即[在 bootstrap.properties 配置文件中配置 Nacos Server 地址](https://www.remnote.com/doc/AxYcAxNeoyYBu9PYS)，从而使用[Nacos作为注册中心](https://www.remnote.com/doc/hP9WezNzi4uMLFMbo?aliasId=5A5LEvPu676cyzvYZ) 。为了使用[Nacos作为配置中心](https://www.remnote.com/doc/Z45PGpRYKgSPvMKJY?aliasId=Dwko7S677gnXrBGJ4)，我们还需要配置[命名空间](https://www.remnote.com/doc/h6qGvZP6Wxi5MWyfr)，[配置集](https://www.remnote.com/doc/YZt65CifBrEyGLJXB)，[配置分组](https://www.remnote.com/doc/35cxFmzh2Zee7qrgZ)等信息。

第三部为[使用 @EnableDiscoveryClient 注解开启服务注册与发现功能](https://www.remnote.com/doc/aeFFPLEwp46wstYxC)。

另外要注意的是，[数据库配置更新需要重启服务Nacos](https://www.remnote.com/doc/PFS6B85h5dCSCxiYr?aliasId=yoomA9CxHXZoFiqva) 。

**使用Gateway**

在[3.1 微服务系统设计](https://www.remnote.com/doc/yBrNNhwLs39dGoM88) 中非业务模块的设计已经分析了Gateway的需要实现两个功能是处理跨域请求和配置路由，本节将介绍其实现：

在跨域功能的实现上，[新建跨域配置类](https://www.remnote.com/doc/LwvWS84KNiAfnR7ob)，即[在hannahmall-gateway中新建config包，包下新建配置类GulimallCORSConfiguration](https://www.remnote.com/doc/Nev2bKFAcFLzdjRhi)，[利用springframework中的CorsConfiguration配置设置Access-Control-Allow-Origin 和 Access-Control-Allow-Methods 为true从而允许所有方法，来自所有来源的请求跨域，另外设置Access-Control-Allow-Credentials为true使得跨域请求包含cookie](https://www.remnote.com/doc/Rkt85Sc7NRR3m3PQ2)

要实现使用Gateway进行路径重写，首先要理解一个 [路由](https://www.remnote.com/doc/F2QkSanbv3k29Go7T) 包含了 [ID](https://www.remnote.com/doc/tb9rMq83D68Fmtvv5)， [URL](https://www.remnote.com/doc/ryKoJriKBKy5hwzHN)，[断言](https://www.remnote.com/doc/6Y8vnwSEmjsNmyHBa) 以及[过滤器](https://www.remnote.com/doc/bZChkPZCbwYkptgax) 。[满足某些断言（predicates）就使用指定的过滤器（filter）路由到指定的地址（uri）](https://www.remnote.com/doc/m2jXZayWwkBdDwmuh?aliasId=Fv6LHrMQ8az9KJGNc)。每添加一个服务，最重要的就是要为其配置 [断言](https://www.remnote.com/doc/CqiP5Np3DR8N7LEaP) 和 [过滤器](https://www.remnote.com/doc/2dtBk7vg6zXi56WR4)。 要额外注意的是，[当一个请求满足多个路由的谓词条件时，请求只会被首个成功匹配的路由转发](https://www.remnote.com/doc/erzLvkWcimtwskZMe) 。在本项目中每个服务都要经过网关进行路径重写，如下分别为商品服务，库存服务和首页访问的路径重写：



图4-7:GateWay代码

**使用Sentinal**

使用Sentinal包括安装Sentinal启动台，和SpringBoot 整合 Sentinel两个部分。

本项目在虚拟机的docker中安装Sentinal启动台，包括下载最新镜像，运行Sentinal并设置映射端口，开启开机自启。

要整合Sentinel，首先要引入.pom 文件，接下来修改服务的配置文件。包括制定控制台的安装位置，与sentinel单独连接的端口，本机的ip地址三个部分。

由于网关是本项目请求最多的部分，因此本项目选择在网关配置 Sentinal。

### 4.4.2 使用Nginx

[nginx在本项目完成了反向代理配置 和 负载均衡到网关 两个功能](https://www.remnote.com/doc/DdgHkBFnc2Jxz3gfh)

先明确两个文件的含义，在这里使用[nginx.conf](https://www.remnote.com/doc/hGWSpucEo6Xz84ZZj) 指 [nginx的根配置文件](https://www.remnote.com/doc/hGWSpucEo6Xz84ZZj?aliasId=JLvcab6Du6EZgd2CN) ，用[hannahmall.conf](https://www.remnote.com/doc/nK4p4vJeqpSGRtyjg) 指 [nginx的server块配置文件](https://www.remnote.com/doc/nK4p4vJeqpSGRtyjg?aliasId=JAA2Z6M85f3EghFKE)

[为了实现反向代理的数据流的操作过程](https://www.remnote.com/doc/c25MGahjXdkD64p9s)，[首先，在本地机器上，修改windows hosts文件，指定本地域名地址 ，之后在Nginx所在的设备上, 创建 nginx.conf 和 hannahmall.conf ， 不要忘记修改配置文件完毕后，重启nginx](https://www.remnote.com/doc/oeXnEsyoZ93LHqQJj)

[另外，为了实现负载均衡，还需要修改nginx.conf ， 配置上游服务器为网关地址，即，将upstream映射到我们的网关服务。 之后修改hannahmall.conf ， 配置代理地址为上面配置的上游服务器名 。 最后配置网关服务，在“hannahmall-gateway”添加路由规则, 将域名为\*\*.hannahmall.com转发至商品服务 。](https://www.remnote.com/doc/drf8AGYAzF6MT33Ms)

第五章 商城系统测试

在第五节中进行程序的测试来验证系统的可靠性，首先介绍[5.2.1 测试环境](https://www.remnote.com/doc/FJ39WcmAcFH6LgQ4A)，之后对照[第四章程序实现](https://www.remnote.com/doc/AzBjJxTyuoe7Gg5ct?aliasId=psa9G8hKG2G3M8mEG)介绍各个功能模块的功能测试，最后对模块的某些功能进行了[5.2.5 性能测试分析与优化](https://www.remnote.com/doc/FRfDxi4qJfTZFusXe)。

## 5.1 测试环境



图5-1：测试环境

## [5.2 商品模块](https://www.remnote.com/doc/nhjjJ3PacL74BZbn4?aliasId=KfnuS6mKYJRXQXP2N)

本节将对 [4.2.1 商品模块业务实现](https://www.remnote.com/doc/kPbTpNCN48zsMQp5y) 进行功能测试。

**[平台属性](https://www.remnote.com/doc/EDQnvky2joj6p4SiJ)**

平台属性基础功能

[测试目的](https://www.remnote.com/doc/epbb4ThRnQumsauKs)

测试平台属性的增删查改功能

[测试步骤](https://www.remnote.com/doc/TcS84h5kb8d5Aapyi)

在后台管理系统点击品牌维护进入平台属性维护界面

点击新增按钮，在输入框输入数据，点击确定

点击修改按钮，在输入框更新数据，点击确定

点击删除按钮，点击确定

[预期结果](https://www.remnote.com/doc/MHYerWh2MQfQTPyMx)

成功新增，修改，和删除

[测试结果](https://www.remnote.com/doc/wJhQ8cv9wWbsq83Jg)

测试通过

SKU组合切换

[测试目的](https://www.remnote.com/doc/epbb4ThRnQumsauKs)

测试是否可以点击不同的SKU组合进行切换

[测试步骤](https://www.remnote.com/doc/TcS84h5kb8d5Aapyi)

进入商品详情页面

选择不同的销售属性组合，检查是否可以跳转

[预期结果](https://www.remnote.com/doc/MHYerWh2MQfQTPyMx)

选择不同的销售属性组合，可以跳转到不同的SKU

[测试结果](https://www.remnote.com/doc/wJhQ8cv9wWbsq83Jg)

测试通过

A picture containing graphical user interface

Description automatically generated

图5-2：SKU组合切换功能

**[品牌](https://www.remnote.com/doc/faBNjDyvrESKmNbwJ)**

品牌关联功能

[测试目的](https://www.remnote.com/doc/epbb4ThRnQumsauKs)

测试品牌是否能够成功与分类关联

[测试步骤](https://www.remnote.com/doc/TcS84h5kb8d5Aapyi)

点击关联，选择分类，点击确定

[预期结果](https://www.remnote.com/doc/MHYerWh2MQfQTPyMx)

品牌和分类能够成功关联

[测试结果](https://www.remnote.com/doc/wJhQ8cv9wWbsq83Jg)

测试通过

Graphical user interface, text, application

Description automatically generated

图5-3：品牌关联功能

[**属性分类 和 属性分组**](https://www.remnote.com/doc/f3csFz7xSvB93WAkp)

**属性分类**

属性分类基础功能

[测试目的](https://www.remnote.com/doc/epbb4ThRnQumsauKs)

测试三级分类是否能够正常展示，新增和删除

[测试步骤](https://www.remnote.com/doc/TcS84h5kb8d5Aapyi)

在后台管理系统点击分类维护进入分类维护界面

点击Append，输入所提示的信息，点击确认

点击Delete，并点击确认

[预期结果](https://www.remnote.com/doc/MHYerWh2MQfQTPyMx)

显示的时候三级分类不再显示Append

添加和删除之后效果可以即刻体现在分类上

[测试结果](https://www.remnote.com/doc/wJhQ8cv9wWbsq83Jg)

测试通过

Graphical user interface, application

Description automatically generated

图5-4：属性分类基础功能

属性分类修改功能

[测试目的](https://www.remnote.com/doc/epbb4ThRnQumsauKs)

测试是否能正常修改某个三级分类

[测试步骤](https://www.remnote.com/doc/TcS84h5kb8d5Aapyi)

在后台管理系统点击分类维护进入分类维护界面

点击某三级分类后的Edit按钮

在输入框修改之后点击确认按钮

[预期结果](https://www.remnote.com/doc/MHYerWh2MQfQTPyMx)

[点击某三级分类后的Edit按钮](https://www.remnote.com/doc/binxJReFnH5sDsC2t)，弹出的对话框可以回显三级分类此时的信息

[在输入框修改之后点击确认按钮](https://www.remnote.com/doc/rX8iji8JxmbMYaHBe)之后可以保存

[测试结果](https://www.remnote.com/doc/wJhQ8cv9wWbsq83Jg)

测试通过

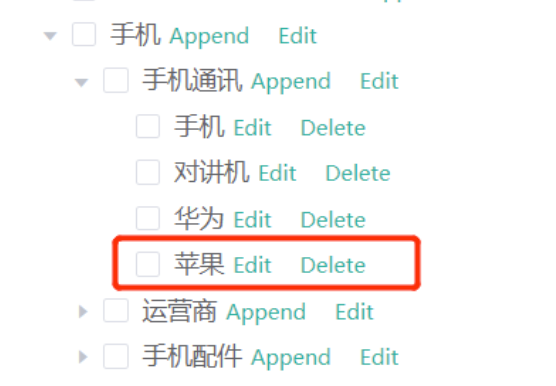


图5-5：属性分类修改功能

属性分类拖拽功能

[测试目的](https://www.remnote.com/doc/epbb4ThRnQumsauKs)

测试是否能正常拖拽三级分类，并批量保存

[测试步骤](https://www.remnote.com/doc/TcS84h5kb8d5Aapyi)

在后台管理系统点击分类维护进入分类维护界面

拖拽修改节点的顺序

点击批量保存按钮

[预期结果](https://www.remnote.com/doc/MHYerWh2MQfQTPyMx)

[点击批量保存按钮](https://www.remnote.com/doc/o5SSsAP5SE365c6D8) 后拖拽修改的顺序能够成功保存

[测试结果](https://www.remnote.com/doc/wJhQ8cv9wWbsq83Jg)

测试通过

**属性分组**

属性分组基础功能

[测试目的](https://www.remnote.com/doc/epbb4ThRnQumsauKs)

测试属性分组的 [查询](https://www.remnote.com/doc/mxbT2aBtvPq7fa6nW)， [新增](https://www.remnote.com/doc/hLx53bu6xqiK9oFQx)， [修改](https://www.remnote.com/doc/ibzatMqf8eqK2WjE8) ，删除 功能

[测试步骤](https://www.remnote.com/doc/TcS84h5kb8d5Aapyi)

在后台管理系统点击分组维护进入分类维护界面

点击三级分类

点击新增按钮，在输入框输入信息之后点击确定

点击修改按钮，输入框更新信息之后点击确定

点击删除按钮，点击确定

[预期结果](https://www.remnote.com/doc/MHYerWh2MQfQTPyMx)

[点击三级分类](https://www.remnote.com/doc/RcPCpoA6GjGJ5eYMW)能够查询到此分类下的分组

新增，修改，删除成功

[测试结果](https://www.remnote.com/doc/wJhQ8cv9wWbsq83Jg)

测试通过

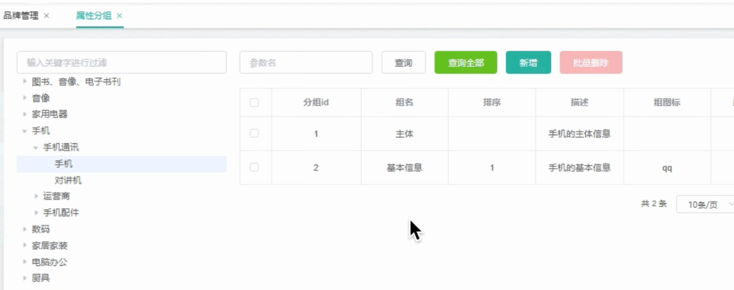


图5-6：属性分组基础功能

属性分组关联功能

[测试目的](https://www.remnote.com/doc/epbb4ThRnQumsauKs)

测试属性分组是否能正常关联

[测试步骤](https://www.remnote.com/doc/TcS84h5kb8d5Aapyi)

在后台管理系统点击分组维护进入分类维护界面

点击页面上的关联按钮

点击新建关联

选择需要关联的属性，点击确定

[预期结果](https://www.remnote.com/doc/MHYerWh2MQfQTPyMx)

[点击页面上的关联按钮](https://www.remnote.com/doc/yefwrcBA46CKYZ42D)，弹出与当前分组关联的属性

[点击新建关联](https://www.remnote.com/doc/apiBNKQxqJpe4L4S7)，界面会显示还没有关联的属性

[选择需要关联的属性，点击确定](https://www.remnote.com/doc/AwgJHxujdELDasw3H) 之后关联保存成功

[测试结果](https://www.remnote.com/doc/wJhQ8cv9wWbsq83Jg)

测试通过



图5-7：属性分组关联功能

[**商品本身的元信息**](https://www.remnote.com/doc/9f8zA9b6LEySuFWTd)

新增商品

[测试目的](https://www.remnote.com/doc/epbb4ThRnQumsauKs)

测试新增一个商品

[测试步骤](https://www.remnote.com/doc/TcS84h5kb8d5Aapyi)

分别输入商品的基本信息，规格参数，销售属性，SKU信息，点击保存

[预期结果](https://www.remnote.com/doc/MHYerWh2MQfQTPyMx)

添加之后可以分别在SPU管理和商品管理中找到自己的商品

[测试结果](https://www.remnote.com/doc/wJhQ8cv9wWbsq83Jg)

测试通过

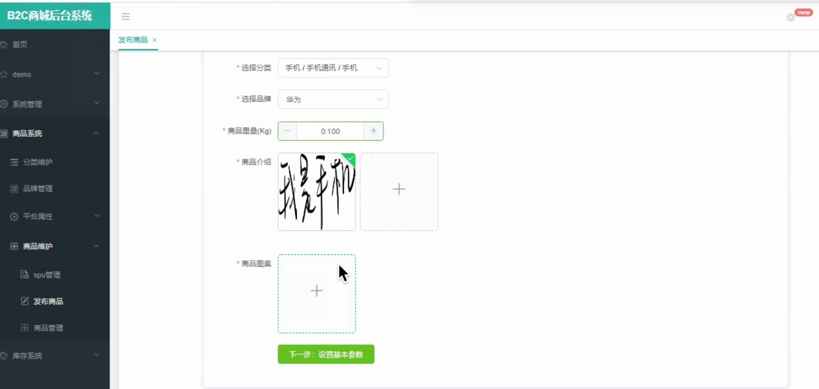


图5-8：新增商品

商品详情

[测试目的](https://www.remnote.com/doc/epbb4ThRnQumsauKs)

测试商品详情页是否可以展示

[测试步骤](https://www.remnote.com/doc/TcS84h5kb8d5Aapyi)

进入商品首页

搜索商品进入商品详情页

[预期结果](https://www.remnote.com/doc/MHYerWh2MQfQTPyMx)

商品详情能够正常显示

[测试结果](https://www.remnote.com/doc/wJhQ8cv9wWbsq83Jg)

测试通过

Graphical user interface, application

Description automatically generated

图5-9：图片展示



图5-10：规格与包装展示

## [5.3 库存模块](https://www.remnote.com/doc/4fWRaL8qnLdP6ehWu?aliasId=koFLkDrksxkDA79PD)

本节将对[4.2.2 库存模块业务实现](https://www.remnote.com/doc/5KLCtgY87ZfmXaWrd) 进行功能测试。

[查询库存](https://www.remnote.com/doc/L7HHmuZMYzQBGPnGY?aliasId=KGiAayfjFxj6f4ZRw)

[测试目的](https://www.remnote.com/doc/epbb4ThRnQumsauKs)

测试查询功能

[测试步骤](https://www.remnote.com/doc/TcS84h5kb8d5Aapyi)

查询库存

查询采购需求

查询未领取的采购单

[预期结果](https://www.remnote.com/doc/MHYerWh2MQfQTPyMx)

[查询库存](https://www.remnote.com/doc/3ahw2AJqTRKmRkZXh) 之后根据查询条件显示库存

[查询采购需求](https://www.remnote.com/doc/Rs6tSYJ6CRSjiBq4C) 之后根据查询条件显示库存

[查询未领取的采购单](https://www.remnote.com/doc/LMeDfNebg6MhnxfbL) 之后根据查询条件显示库存

[测试结果](https://www.remnote.com/doc/wJhQ8cv9wWbsq83Jg)

测试通过



图5-11：查询库存

[合并采购需求](https://www.remnote.com/doc/MaXS6Zp8psgZgh87d?aliasId=Wq8PGy46y35i3wtmw)

[测试目的](https://www.remnote.com/doc/epbb4ThRnQumsauKs)

测试能否成功合并采购需求

[测试步骤](https://www.remnote.com/doc/TcS84h5kb8d5Aapyi)

选择要合并的商品

在批量操作下选择合并整单

选择负责人

点击确定

[预期结果](https://www.remnote.com/doc/MHYerWh2MQfQTPyMx)

采购单合并成功

[测试结果](https://www.remnote.com/doc/wJhQ8cv9wWbsq83Jg)

测试通过

Graphical user interface, application

Description automatically generated

图5-12：合并采购需求

[分配采购单](https://www.remnote.com/doc/wN4PgWtyJDpd35QPG?aliasId=c5PB2t3Gr3F2xX3DZ)

[测试目的](https://www.remnote.com/doc/epbb4ThRnQumsauKs)

测试能否成功分配采购单

[测试步骤](https://www.remnote.com/doc/TcS84h5kb8d5Aapyi)

进入采购单页面

点击分配按钮

选择被分配的采购人员

[预期结果](https://www.remnote.com/doc/MHYerWh2MQfQTPyMx)

采购单人名变为被分配的采购人员

[测试结果](https://www.remnote.com/doc/wJhQ8cv9wWbsq83Jg)

测试通过



图5-13：分配采购单

## [5.4 搜索模块](https://www.remnote.com/doc/pH2E9uihbNwo93967?aliasId=3EFgjXZoD4PTCkfBB)

本节将对[4.2.3 搜索模块业务实现](https://www.remnote.com/doc/bCFgJgyQmFEqAiTMm) 进行功能测试。

[商品上架](https://www.remnote.com/doc/Mon5ihwvKhtrK6KQc)

[测试目的](https://www.remnote.com/doc/epbb4ThRnQumsauKs)

测试商品能否成功上架

[测试步骤](https://www.remnote.com/doc/TcS84h5kb8d5Aapyi)

在后台管理系统点击上架

[预期结果](https://www.remnote.com/doc/MHYerWh2MQfQTPyMx)

在前台可以搜索到已上架的商品

[测试结果](https://www.remnote.com/doc/wJhQ8cv9wWbsq83Jg)

测试通过

[页面排序功能](https://www.remnote.com/doc/DxXfm4g2N6LLpQnco)

[测试目的](https://www.remnote.com/doc/epbb4ThRnQumsauKs)

测试页面能否根据选择的条件进行排序

[测试步骤](https://www.remnote.com/doc/TcS84h5kb8d5Aapyi)

搜索关键字时候进入搜索页面

点击按价格排序

点击按销量排序

[预期结果](https://www.remnote.com/doc/MHYerWh2MQfQTPyMx)

[点击按价格排序](https://www.remnote.com/doc/b673F2EqaBH6WWsvk) 之后，结果按照价格排序

[点击按销量排序](https://www.remnote.com/doc/YykZPS7HaffieTsue) 之后，结果按照销量排序

[测试结果](https://www.remnote.com/doc/wJhQ8cv9wWbsq83Jg)

测试通过



图5-14：页面排序功能

[面包屑导航功能](https://www.remnote.com/doc/9NBGJjXeiqknFoDNx)

[测试目的](https://www.remnote.com/doc/epbb4ThRnQumsauKs)

测试面包屑导航是否可以正常使用

[测试步骤](https://www.remnote.com/doc/TcS84h5kb8d5Aapyi)

进入商品搜索页面

在筛选条件中选择相应分类

在筛选条件中选择相应品牌

在筛选条件中选择相应属性

[预期结果](https://www.remnote.com/doc/MHYerWh2MQfQTPyMx)

所选择的分类，品牌，属性会进入面包屑，点击面包屑中的元素会取消此筛选条件的选择。

[测试结果](https://www.remnote.com/doc/wJhQ8cv9wWbsq83Jg)

测试通过

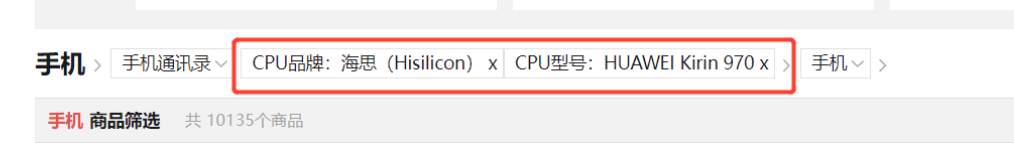


图5-15：面包屑导航功能

## 5.5 性能测试分析与优化

长的响应时间会严重影响用户体验，因此在前台我们还需要考虑用户的性能需求。在这一章，我们将对项目性能进行测试。在本节首先介绍[5.2.5.1 测试工具与性能指标](https://www.remnote.com/doc/uLNMQ4B5bWQkFqmnX) ，之后对[中间件性能测试与分析](https://www.remnote.com/doc/z9p8PrkGQKhkEgdN9?aliasId=yYEGacXX5inC4wc6H)，再结合[中间件性能测试与分析](https://www.remnote.com/doc/z9p8PrkGQKhkEgdN9?aliasId=yYEGacXX5inC4wc6H)进行商品首页的性能测试，最后[进行业务的性能优化](https://www.remnote.com/doc/xrbEb9osm5PcMT9cK?aliasId=3kQdYpaKZjMPhcgBj)。

### 5.5.1 测试工具与性能指标

[本项目使用的测试工具为JMeter，模拟多个用户访问一个接口，来测试这个接口的性能如何。测试步骤为第一创建测试计划，添加线程组 ， 第二添加http请求 ， 第三添加监听器 ，第四点击绿色小三角启动压测，并等待数据稳定之后查看分析结果 。](https://www.remnote.com/doc/Fkn2aEcgJ9ETq5CZC)

[测试程序性能使用的指标有响应时间，吞吐量和错误率。 测试时间指的是响应时间指用户从客户端发起一个请求开始，到客户端接收到从服务器端返回的响应结束，整个过程所耗费的时间。 此时间需要测量最大响应时间， 平均响应时间 和 最少响应时间，以及不同分位 ，比如TP90，TP95 和TP99。 接下来是吞吐量指标，常见的指标有HPS，TPS 和QPS。JMeter中使用的是TPS。 在实际项目中，我们要兼顾吞吐量和响应时间。](https://www.remnote.com/doc/FLqqsnpptQ7ooqZPk)

### 5.5.2 对中间件性能测试与分析

在本节，分别进行了Nginx和Gateway的测试与分析

Nginx的测试与分析

压测结果如下：

A screenshot of a computer

Description automatically generated with medium confidence

图5-16：Nginx压测结果

[启动jmeter, 大概50秒左右，停止jmeter](https://www.remnote.com/doc/ZZAi9StweHA7ez4ny)，之后[使用docker stats动态查看doker各个容器的状态](https://www.remnote.com/doc/zPMQPPCPkHNgH7mGB)：

A screenshot of a computer

Description automatically generated with medium confidence

图5-17：使用docker status查看状态

通过CPU和MEM占用情况，[可以得出结论 nginx 是 CPU 密集型](https://www.remnote.com/doc/K5gQqj6X7gz8dTv2J)

GateWay的测试与分析

压测结果如下：

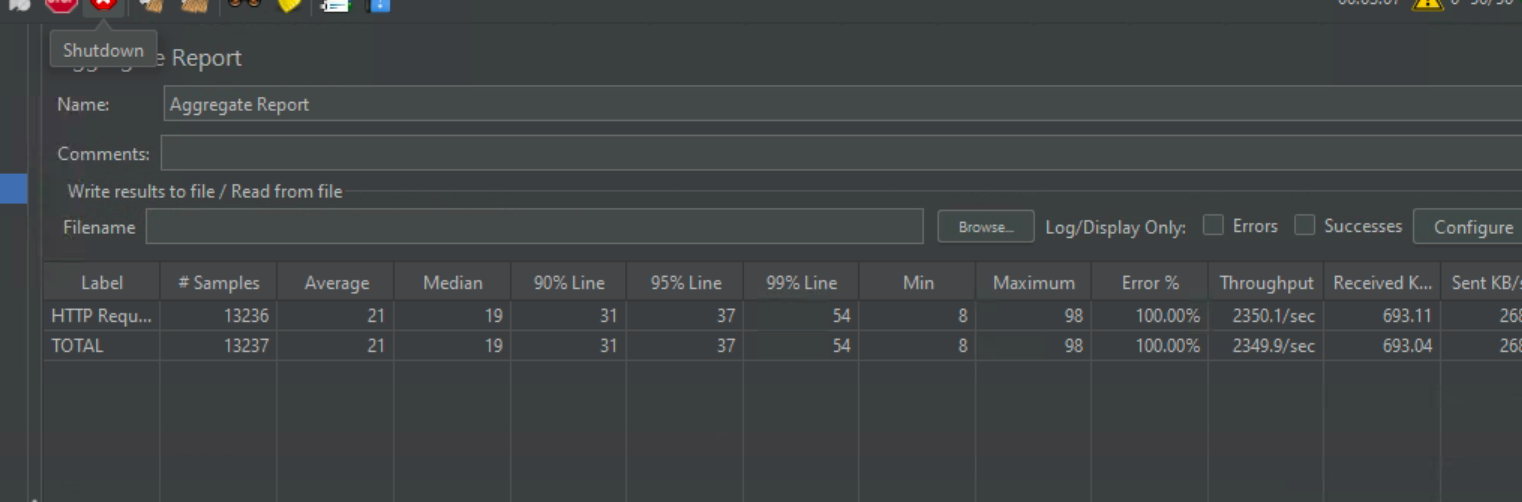


图5-18：GateWay压测结果

使用 [jvisualvm](https://www.remnote.com/doc/JZGzKdBosixmMdaXt) 进行内存分析

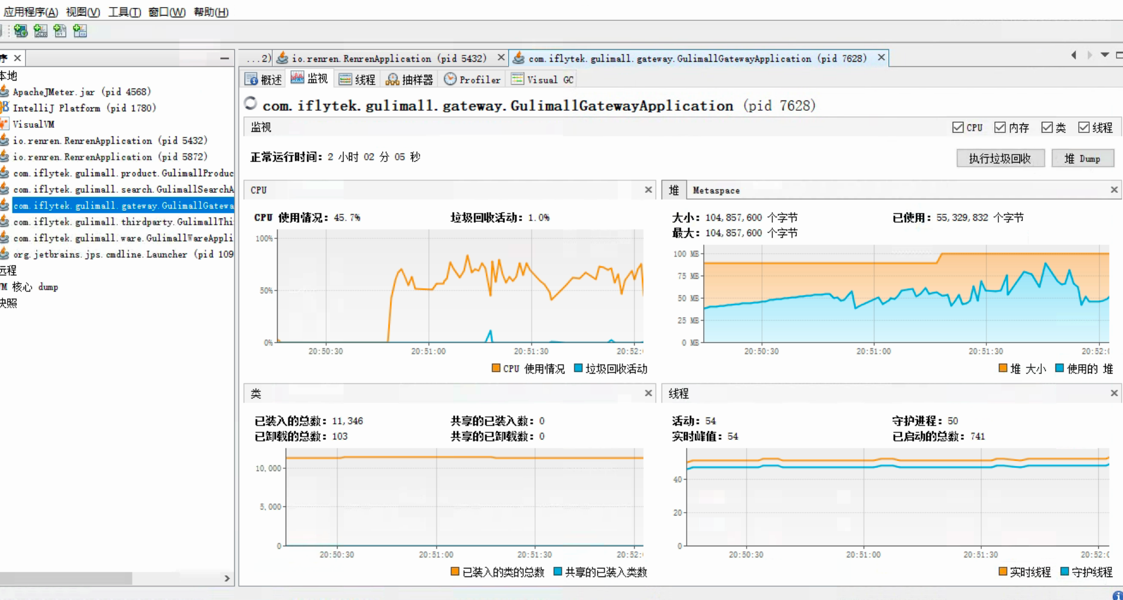


图5-19：jvisualvm内存分析

通过CPU占用情况，[可以得出结论网关是 cup 密集型](https://www.remnote.com/doc/pWRmvZHjR7jD2G4SK)

### 5.5.4 业务的性能优化

获取三级分类属于 [访问量大且更新频率不高的数据](https://www.remnote.com/doc/uWSCni4qwiXyd9mh2)，对此[除了进行代码层面的优化和数据库层面的优化](https://www.remnote.com/doc/ss46x3YEP4P2qWyFr)，我们还可以为其添加缓存。但是，在添加缓存的时候要注意避免[缓存穿透](https://www.remnote.com/doc/BoyRd4RZrX4D9cvYj)，[缓存雪崩](https://www.remnote.com/doc/LrCz6PeYEYF7EGmdQ)和[缓存击穿](https://www.remnote.com/doc/j8nHqBRkLAupN9Hs2)问题。[使用缓存之后的数据流](https://www.remnote.com/doc/JJjYhLmaQoWrGSMfD)为，[当请求一个函数时，首先判断缓存中是否存在。如果存在，程序将直接从缓存中读取数据。没有则到数据库中查询，并将查询结果放入到 Redis 缓存中](https://www.remnote.com/doc/Lgd8ck2a3S5ymDKDy) 。本节对三级分类的测试包括未添加缓存的接口测试和已添加缓存的接口测试。

未加缓存的性能

Graphical user interface, application

Description automatically generated

图5-20：未加缓存压测

添加缓存的性能

Graphical user interface

Description automatically generated

图5-21：添加缓存压测

可见添加缓存会对性能有较大的影响。

第六章 总结与展望

## 6.1 总结

本论文通过实践的方法完成了一个B2C电子商城的设计与实现，从[技术方案调研](https://www.remnote.com/doc/czqinaAatGNAt53Ki?aliasId=EFNhXSwNJyG4hrNEi) ，[需求分析](https://www.remnote.com/doc/mxcc7mmgvZuzGEmhG)，[程序设计](https://www.remnote.com/doc/av23uZY2w7As3NNrN) ，[程序实现](https://www.remnote.com/doc/AzBjJxTyuoe7Gg5ct) ，到最后的[程序测试](https://www.remnote.com/doc/uBor5g3B4FiKMKeTR)，主要工作可归纳为如下：

对技术方案进行了调研，包括[前端技术](https://www.remnote.com/doc/zSm93sw3YEzLo89dK?aliasId=2Eq6uYTwfGxTg4LbE) 和[后端技术](https://www.remnote.com/doc/tLCwYhfiNxXqehrRx?aliasId=gdQYbmz3H7mJWHnTt)，其中重点为后端技术，包括使用SpringBoot程序的开发和Spring Cloud组件的治理。

对需求进行了细致的分析，包括[会员前台消费功能需求](https://www.remnote.com/doc/BogPNcboveTnqgYsK?aliasId=mq7f5WBRegQ38ymwH) 和 [商家后台管理功能需求](https://www.remnote.com/doc/iMqrPbru2Sf5vPfKn?aliasId=Np53njXPcHhjkhmNA)

根据微服务设计原则对程序进行了设计，包括[微服务系统设计](https://www.remnote.com/doc/yBrNNhwLs39dGoM88?aliasId=Z75pS3RTyaEnWCfur) 和 [3.2 业务模块流程设计](https://www.remnote.com/doc/8fJQcKXtnWe9qjetz) 以及 [3.3 程序设计原则](https://www.remnote.com/doc/4PB7rfzm6cMRYZDMY)。其中的重点是[3.3 程序设计原则](https://www.remnote.com/doc/4PB7rfzm6cMRYZDMY)，主要包括 [3.3.1 数据结构设计原则](https://www.remnote.com/doc/qKeQSZLHMPdbpXSZc) 和 [3.3.2 后端程序设计原则](https://www.remnote.com/doc/4zdhMn8BrR7EdGXwh)。

对设计出的程序完成了实现，包括[4.2.1 商品模块业务实现](https://www.remnote.com/doc/kPbTpNCN48zsMQp5y) ，[4.2.2 库存模块业务实现](https://www.remnote.com/doc/5KLCtgY87ZfmXaWrd) ，[4.2.3 搜索模块业务实现](https://www.remnote.com/doc/bCFgJgyQmFEqAiTMm) 和[4.2.4 服务治理功能实现](https://www.remnote.com/doc/RHRGoKSjri9FoWKJS)。

最后完成了程序测试，分别包括各模块的功能测试与性能测试，并对测试结果进行了分析。

## 6.2 展望

由于本系统主要以学习技术为目的，个人时间和能力所限，模块的丰富度不是很高。因此在接下来的工作中，还可以添加更多功能，来更真实的还原电商系统，比如添加[购物车](https://www.remnote.com/doc/t7yDKzjtbXf4mgXHk)，[订单服务](https://www.remnote.com/doc/ThR3YQ2rs2JNwHD5J)，[秒杀服务](https://www.remnote.com/doc/aMYWyvB7wmtmsXjeP)等。

参考文献

1. 孙惟斯. 基于服务设计理念的B2C电子商务网站设计研究[D]. 华东理工大学,2014.
2. 姚远. 基于服务设计理念的B2C电子商务网站设计研究[J]. 数字技术与应用,2016, (02): 153.
3. Akbulut A, Perros H G. Performance analysis of microservice design patterns[J]. IEEE Internet Computing, 2019, 23(6): 19-27.
4. 彭向峰. 电子商务平台—购物网站的设计与实现[D]. 山东大学, 2012.
5. Huang X, Dai X, Liang W. BulaPay: a novel web service based third-party payment system for e-commerce[J]. Electronic Commerce Research, 2014, 14(4): 611-633.
6. 徐士川. 电子商城系统中订单模块与秒杀模块的设计与实现[D]. 南京大学, 2018.
7. 熊益益. 基于微服务架构的电商平台的研究与实现[D]. 北京邮电大学, 2019.
8. 钟俊林. 基于微服务架构的自助微商城的研究与实现[D]. 北京邮电大学,2019.
9. Hasselbring W , Steinacker G . Microservice Architectures for Scalability, Agility and Reliability in E-Commerce[C], IEEE International Conference on Software Architecture Workshops. IEEE, 2017: 243-246.
10. Rasheedh J A, Saradha S. Reactive Microservices Architecture Using a Framework of Fault Tolerance Mechanisms[C], 2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC). IEEE, 2021: 146-150.
11. Armin, Balalaie, Abbas, et al. Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture[J]. IEEE Software, 2016, 33(3):42-52..

致 谢

首先要感谢各位朋友，让我在大学充分的探索了生活的可能性。学姐在交换过程中的分享，让我对自己毕业之后的规划有了明确的认知。学长在科研方向的指导，让我对计算机领域的多个方向有了更深刻的了解。身边一起分享和进步的同学，也使我的大学生活更加充实与饱满。

其次要感谢学校里的各位老师，本论文的指导老师林伟伟老师，提供给我了给我自由选择题目的机会，使得本题目能够和我的规划的职业规划发展方向契合。具体来说，在本次毕业项目中，我深入的了解了微服务系统的设计与实现，为以后的职业发展打下了基础。并提供建设性的建议，使得本论文的内容和结构更加的合理。还要感谢本科期间其他老师给予的课程，科研方面的指导，开拓了我在计算机科学与技术领域的眼界。

最后还要感谢自己。只有做到自尊，自信，自爱，享受每一个当下，充分体会在自己规划的轨道上前进的快乐，才能批判性的面对质疑的声音，包容和爱他人，用自己的独特性为社会共同体作出正面影响。