

Research on Electricity Based on Java Software Development and Big Data

Zisheng Qi

Zhuhai Haosheng Network Technology Co., Ltd., Zhuhai, Guangdong, 519000, China

Abstract

In recent years, with the development and popularization of internet technology and intelligent mobile devices, numerous e-commerce apps such as Tmall, JD.com, Pinduoduo, and Dewu have emerged, cultivating consumers' online shopping habits, adapting to the life of mobile internet access, and providing a basic basis for the establishment of big data operations and management platforms. E-commerce platforms, as a representative platform, have generated a large amount of data during their development process. However, how to utilize and analyze these data has become a hot topic in current research. The application of big data technology has become the key to collecting e-commerce data and implementing data analysis. Therefore, the paper uses Java as the development language to develop and design the overall architecture of e-commerce software, and conducts research and analysis on big data. At the same time, a brief discussion was also conducted on mobile clients and big data, hoping to provide corresponding support for e-commerce software development and consumer experience improvement.

Keywords

electric field; Java technology; software development; big data

基于 Java 的电商软件开发与大数据研究

齐自盛

珠海浩盛网络科技有限公司, 中国·广东 珠海 519000

摘要

近几年随着互联网技术与智能移动设备的发展普及, 催生出了天猫、京东以及拼多多、得物等众多电商APP, 培养了消费者的网购习惯, 适应了移动互联接入的生活, 也为大数据运营以及管理平台的建立提供了基础依据。以电商平台为代表平台在发展过程中产生了大量的数据量, 但如何实现对这些数据的利用以及分析成为当下研究的热点。而大数据技术的应用, 成为采集电商数据和实现数据分析的关键。因此, 论文以Java为开发语言, 对电商软件总体架构进行了开发设计, 并对大数据展开研究分析。同时, 还对移动客户端与大数据进行了简单论述, 希望能够为电商软件开发以及消费者的体验提升提供相应的支持。

关键词

电商领域; Java技术; 软件开发; 大数据

1 引言

随着互联网以及电商领域的发展, 大数据时代背景下电商领域的发展迎来了新的高峰。而基于大数据的精准营销, 成为电商领域备受瞩目的发展方向, 这不仅促进电商企业的营销效率得到提升, 还让企业在追求电商软件开发创新之外, 借助大数据技术实现了对已有企业的数据进行采集、提取、分析和处理, 这样一来既提高了企业的利润, 提升了电商平台的品质还优化了消费者的购物体验^[1]。

2 系统、总体开发思路及结构设计

一般传统的软件开发设计过程中, 为了能够表达出直

观的功能, 均通过将数据库代码、业务逻辑处理以及前端界面等相关代码融合在一起进行软件编写。但这种, 方法不仅会导致软件代码存在高耦合度和差可读性的问题, 还会对系统中的每一个功能模块的后期维护造成一定程度的影响。因此, 为了避免上述问题, 论文选择利用Java开发语言和分层架构的方式实现对电商系统的开发设计。电商平台中, 高质量、高品质、高性价比的商品是, 广大消费者所注重的基本问题与需求。这对电商软件的开发人员来说, 只有及时地掌握与了解消费者的实际需求, 才能够为消费者提供更好的服务, 因此电商软件的开发就显得极为重要。所以, 针对整个电商软件开发和大数据的研究, 论文主要功能模块设计和安卓客户端与大数据清洗等来实现^[2]。其中, 采用B/S架构实现对电商软件的后台设计, 利用MapReduce与HDFS等开源编程模型, 实现对大数据的清洗作业。而电商

【作者简介】齐自盛(1986-), 男, 中国广东珠海人, 硕士, 工程师, 从事软件开发研究。

APP 的开发与大数据研究，框图见图 1 所示。

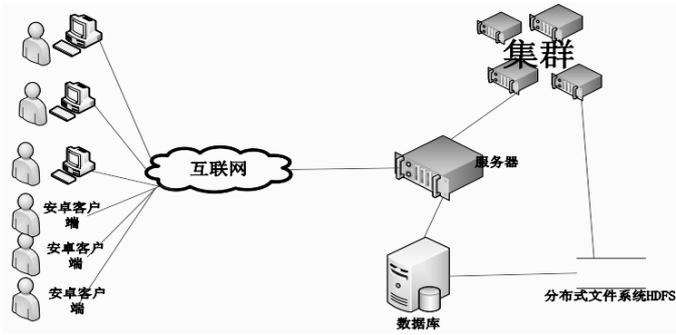


图 1 电商系统网络总体架构框图

从图 1 可以发现，该系统总体结构分为 Web 服务器、数据库、数据清洗等部分组成。电商后台则是整个电商 APP 开发研究的核心部分，数据清洗则是提高了电商后台的服务质量，并实现了对电商平台中消费者的行为以及习惯进行采集与分析，从而为消费者推荐心仪的商品，满足消费者的需求。

3 电商后端平台设计

3.1 基于 B/S 架构的电商后台

电商后台的设计主要是为电商企业的运营人员提供的操作系统，让运营人员通过后台就能够实现对平台优惠券与用户标签等工作的配置操作。而在完成配置前运营人员还可以通过借助数据清洗工具，实现对最后清洗的结果进行查询与分析，从而得到最优的配置方案，并完成对应的配置工作。因此，针对电商后台的设计选择利用 B/S 架构来实现，卷起分成后端接数据库以及 web 管理界面等三个部分，具体见图 2 所示。

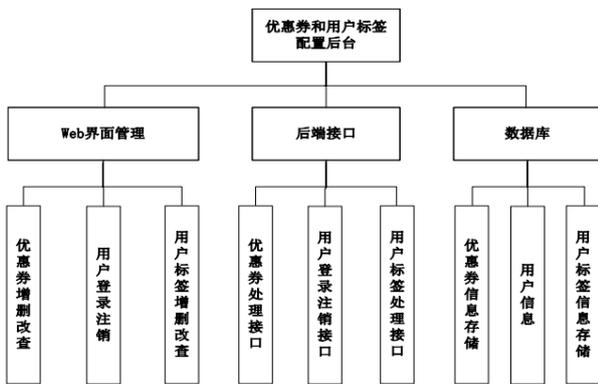


图 2 电商后台主要功能架构框图

其中，从数据库设计方面来看，其作用主要用于对用户标签信息以及优惠券信息的存储；从后端 API 接口设计方面来分析，电商后端设计主要为移动客户端的服务提供相应的接口，而运营人员只需要通过客户端调用 API 接口就能够实现对所提交的优惠券信息以及用户标签信息的调用与处理。此外，web 界面的设计主要实现了对优惠券、用户

标签等信息的管理、增加、删除，以及对生效 / 失效时间和对应编码等参数的管理。

3.2 后台主要功能模块设计

3.2.1 用户登录模块设计

用户登录模块的设计，保障了用户信息的安全以及对不同运营人员的操作进行了查看与配置。此外，当电商系统出现异常问题时，运营人员可以通过对系统日志查询，追踪到异常问题。而用户在登录后台前，需要线上人事或者电商软件开发人员申请一个超级账号后才能够通过登录界面进入平台后端系统。

3.2.2 增 / 删改查功能模块设计

此模块的设计，让运营人员在配置不同优惠券以及用户标签时，出现错误时可以进行修改或者删除等操作。运营人员只需要通过用户登录模块进行后台系统，系统就会自动跳转到增删改查页面，方便运营人员对用户标签信息以及优惠券信息的查询。若是，在增删改时出现操作失败的情况，运营人员就可以通过后端代码与数据库中的代码信息进行比较，然后对提示错误的原因进行修改。

3.2.3 数据传输加密校验功能模块设计

在电商系统开发设计时，考虑到消费者的个人隐私的安全性，以及提高数据通信传输的安全。选择利用数据加密算法实现对原始数据进行加密处理，这样一来在进行数据传输时，就能够保证数据的安全性。此外，在对接口编写时，数据加密算法的应用还能够起到校验的作用，让特定的人员实现对特定的接口应用。

3.2.4 内存缓存模块设计

在对电商优惠券与用户标签等接口开发设计时，为了提高接口的响应速度，只有利用数据清洗工具，定期对数据库当中用户标签信息和优惠券信息等进行清洗，并将其载入到内存当中做缓存。因此，在对内存缓存模块设计时，可以利用线程池技术并对其进行二次封装处理，搭配 Spring 框架当中的初始化 Bean 实现内存缓存功能模块设计。

3.2.5 文件自动载入模块设计

为了能够提高电商软件的开发效率，以及提高代码的清晰度与后期对电商系统的维护效率。需要对大量 properties 配置文件进行更改，并且每一次的修改都需要重新启动项目，这样一来不仅导致开发人员的工作效率降低，还可能会导致配置文件的丢失。因此，论文选择利用 commons-configurationjar 包中的 Properties Configuration 核心类，实现对该类型的文件进行再处理，而该文件自动载入模块开发，实现了自动到同类路径下自动加载指定的配置文件、获取相应的数据信息，从而有效提高开发人员的工作效率^[3]。

3.3 数据库设计

数据库作为电商软件的核心部分，在对电商软件开发设计时，数据库结构的优良直接对电商软件的性能以及效率有

着重要的影响。因此,在对数据库设计时,需要结合电商软件的主要功能以及所需要的统计数据类型方法,才能够实现对数据库当中的每一个表进行设计,并完成数据表和每个数据之间的关系关联。所以,完成数据表设计工作后,还要进行字段属性创建,并且这些属性都各自具备对应的要求,比如说在数据表当中的主键属性不可以是 NULL,且需要设计成自动增长才行;此外,在对应相关联的数据表当中,需要设计对应的外键关联,只有这样才能够提高电商系统的效率。

3.4 数据库 HBase 设计

HBase 数据库 HBase 是一个基于 HDFS 开发的面向列的分布式数据库,擅长处理海量数据的实时查询分析,例如实时分析消息等。HBase 的部署需要 HDFS 和 Zookeeper 的支持。在分布式环境中,ZooKeeper 充当指挥官的角色,提供服务如同步锁服务和配置服务等。HBase 不直接支持 SQL 查询,只提供 Java API 接口。HBase 是构建在 HDFS 之上的,也可以写入本地文件系统。默认情况下,HBase 写入本地文件系统,但这样的 HBase 只能运行在单机上,对于分布式大数据环境来说,单机模式是无意义的。在 HBase 集群中,主要由 Master 和 Region Server 组成,同时也依赖于 Zookeeper。

4 安卓端与大数据研究

4.1 大数据研究

针对电商软件当中的数据,选择利用 Hadoop 实现对数据的分析处理,但想要充分发挥处理 Hadoop 的并行处理优势,还要借助 MapReduce 作业才可以。当作业在被部署到集群上运行时,能够有效提高并行计算处理的速度以及性能。不过,在处理开始前还要生成一个输入分片,并对其切换做成记录。MapReduce 的计算框架主要分为 Map 和 Reduce 等两个处理阶段,且输入以及输出的类型全部由软件开发人员来决定。同时,在 Map 阶段向 Reduce 阶段进行转换时,存在 Shuffle 过程,简单来说就是排序问题与分组问题,实现乐都 Map 的输出键值以及对按键的排序与分组,通过 Shuffle 处理,从而转变成 Reduce 阶段输入。此外,即使面对面海量数据处理计算时,依然具备强大的容错机制。因此,在对电商软件的应用程序开发时,开发人员需要对 Map 和 Reduce 等基本函数进行重新的设计和编写,然后再将其定义在不同类型的接口上。

4.2 数据清洗

以电商软件中次数统计为例,从实际应用开发角度分

析,在电商平台当中存在较多次数统计的需求。例如,消费者的 IP 访问次数以及对某项商品的购买点击次数等,电商后端就会产生大量的日志文件。如电商优惠券的兑换程序后端接口的调用时,就会产生出大量的数据日志文件。这时就需要对这些数据日志文件进行清洗处理,这样一来才能够实现对优惠券兑换按键的点击次数进行统计分析,针对此问题的处理开发一个 Mapreduce 程序,就能够实现对上述问题的处理和解决。并且,对于电商平台当中的次数统计人物,主要由 Map 和 Reduce 等两个部分组成,并在 HDFS 集群当中完成对应阶段核心代码的设计。

根据这两个阶段的统计作业流程来看,系统通过利用日志解析工具,返回一个关于 Map 的集合,这时若是存在需要,则可以利用统计规则实现对日志的过滤处理,并对日志数据进行清洗转换之后,给予 Reduce 进行整合处理,并将整合的结果写入到 HDFS 集群上。

当完成电商软件开发工作后,为了能够保证软件的正常稳定运行,选择利用 Debug 模式,通过直接调试的方式对软件的可行性进行测试。论文选择利用白盒测试方法对电商后台系统进行测试,如测试了增删改用户标签与优惠券以及利用 Email 查询消费者的用户标签等,进行验证,发现其功能达到了设计预期目标^[4]。

5 结语

综上所述,论文通过对基于 Java 的电商软件开发以及大数据研究,发现在电商软件开发设计过程中,主要通过优化配置电商企业的优惠券以及消费者标签为主,才得以实现了软件的各项功能以及后端平台的设计。同时,后端平台的设计还保留了其可扩展性,能够为后续支付接口端对接提供支持。而大数据的应用,实现了对电商平台中的海量数据的收集和分析、处理,充分保障了软件及后端平台的稳定运行。

参考文献

- [1] 谷光明,龙安全.基于Java技术的校园二手商城网站开发[J].安顺学院学报,2022,24(3):117-120+132.
- [2] 王亚婷.基于大数据的精准营销管理系统的设计与实现[D].南京:南京邮电大学,2022.
- [3] 朱建平.基于微服务架构品牌官方电商平台设计与实现[D].南昌:南昌大学,2022.
- [4] 范路桥,高洁,段班祥,等.基于Python+ECharts的手机销售数据可视化系统[J].电脑编程技巧与维护,2022(6):78-81.