计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2022,31(3):85−94 [doi: 10.15888/j.cnki.csa.008341]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于 Spring MVC 的大数据交易集市平台①

滕　飞

(武警工程大学 装备管理与保障学院, 西安 710086)
通信作者: 滕　飞, E-mail: whyifeifei@sina.com

摘　要: 传统的大数据交易集市以促成大数据交易为主, 类似普通商品交易平台, 提供的主要功能偏重于数据目录管理、交易过程的事务管理. 这种方式存在诸多弊端, 除公开数据外, 各数据所有者考虑到数据安全、隐私或数据被滥用, 或分享数据损害自身竞争优势, 仍对数据交换持谨慎态度, 同时各企业对不同行业、多领域存在的数据了解有限, 制约了数据的交换, 仅靠数据目录无法完全达到能力输出的目的, 为有效解决上述弊端, 本文设计了一种大数据交易集市平台, 系统采用 Spring MVC 构建后台整体架构, 集成了 Redis、MyBatis 等技术, 向数据机构提供公用的 Hive 集群保存数据集, 并实现了数据集购买, 查看样本数据, 用户筛选数据集内容等功能.

关键词: 大数据交易集市平台; 数据商品; Spring MVC; Hive

引用格式: 滕飞.基于 Spring MVC 的大数据交易集市平台.计算机系统应用,2022,31(3):85–94. http://www.c-s-a.org.cn/1003-3254/8341.html

Big Data Trading Market Platform Based on Spring MVC

TENG Fei

(College of Equipment Management and Support, Engineering University of PAP, Xi'an 710086, China)

Abstract: Traditional big data trading markets focus on facilitating big data transactions, similar to ordinary commodity trading platforms. Their main functions lie in data catalog management and transaction management in the transaction process. This traditional mode has many drawbacks. Except for public data, considering data security, privacy, data abuse, or damage to own competitive advantages due to data sharing, data owners are still cautious about data exchange. At the same time, companies have limited understandings of data in many fields, which also restricts data exchange. The capacity output cannot be fully achieved by the data catalog alone. To effectively address the above drawbacks, this study designs a big data trading market platform. The system uses Spring MVC to build the overall back-end architecture and integrates Redis, Mybits, and other technologies to provide data institutions with public Hive clusters for saving data sets. Moreover, it realizes the functions of data set purchase, sample data viewing, and user screening of data set content.

Key words: big data trading market platform; data commodity; Spring MVC; Hive

大数据 (big data), 指无法在一定时间范围内用常规软件工具进行捕捉、管理和处理的数据集合, 是需要新处理模式才能具有更强的决策力、洞察发现力和流程优化能力的海量、高增长率和多样化的信息资产. 大数据具有以下特点: (1) 数据体量巨大. 可以达到 TB、PB 级别; (2) 数据类型繁多, 例如网络日志、视频、图片、地理位置信息等; (3) 处理速度快; (4) 商业价值, 只要利用合理的分析手段, 可以挖掘出数据中潜在的商业价值. 在大数据时代, 如何充分挖掘出蕴藏于数据资源中的价值正在成为各国 IT 业界, 学术界和政府共同关

① 收稿时间: 2021-04-28; 修改时间: 2021-05-19; 采用时间: 2021-06-08; csa 在线出版时间: 2022-01-24

注的焦点[1]. 2015 年中国大数据市场规模达到 33.85 亿元, 2020 年大数据交易市场规模达到 540 亿元. 大数据市场不仅规模大, 并且呈现快速增长趋势, 为了更好地运用数据资源, 目前具有的大数据交易平台有日本富士通公司的大数据交易集市 Data plaza、BDEX、贵阳大数据交易所、数据堂等, 开发大数据交易平台具有重大意义. 为解决大数据交易中存在的自身数据不出门 、数据交换受制约等问题, 并为用户提供查看样本数据、根据需求筛选数据集内容等功能, 同时创客可以利用平台购买数据开发相关的应用, 并发布在本平台, 本文设计了一种大数据交易集市平台, 并利用 Spring MVC 构建后台整体架构开发出相应的原型系统.

1 系统设计

1.1 系统总体规划

目前的大数据交易平台主要的模式有:

(1) 提供交易平台, 发布数据目录, 不直接提供数据.

(2) 发布数据目录, 数据提供方上传数据并保存.

(3) 发布数据目录, 发布销售数据.

(4) 提供采集工具, 采集公开网页数据, 收取服务费.

在现有的交易模式中, 私有数据所有者出于数据安全的考虑, 担心隐私泄露、数据被滥用、数据被竞争对手利用, 往往不愿意交换自己拥有的数据. 同时在不同的行业和专业领域之间, 彼此难以相互了解对方的数据, 靠数据目录使得数据难以发挥自身的价值. 本文提出的大数据交易集市平台是基于各方的分布式数据和应用建立和运营一个公信的数据和应用服务交易平台, 本系统可以为数据机构提供数据和应用交易枢纽代理, 如图 1 所示. 主要特点如下.

图 1 数据和应用交易枢纽图

(1) 数据由提供者存储和管理: 数据交易集市不直接管理数据, 谁的数据谁控制, 数据请求发生时由数据代理根据需求从数据所有者获取数据交给需求方. 数据提供者可将数据保存自己公司的服务器上, 或保存到本系统提供的 Hive 集群中, 当发生交易时, 系统拥有权限, 通过相应的 API 接口获取服务器中的数据集.

(2) 提供数据样本数据: 用户可以在数据商城查看样本数据集, 进一步了解数据商品.

(3) 筛选数据集内容功能: 用户可以根据需求选择需要的字段, 过滤掉数据集中无价值的信息.

(4) 数据应用可开发: 提供数据应用开发环境, 满足基于大数据的创新应用需求.

大数据交易集市原型系统的总体规划如图 2 所示.

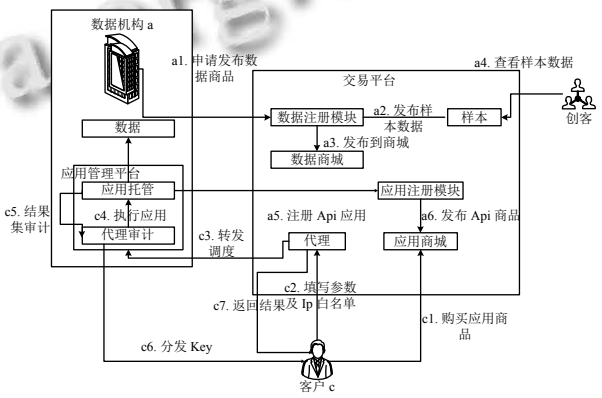

图 2 系统总体规划图

使用大数据交易集市平台的有数据机构 (数据商品的提供方)、创客 (应用程序的开发者)、客户.

数据机构可以将拥有的数据发布到平台的数据商城中进行交易, 同时把数据集中部分数据作为样本数据加入到平台的样本库中, 创客通过查看样本数据, 决定是否购买相应的数据集. 数据机构也可以利用自身拥有的数据的优势开发应用程序, 开发出的应用程序可以发布到平台的应用商城中, 客户通过应用商城购买需要的应用.

交易平台会对数据机构提供的数据集、应用程序进行审核, 审核通过后才会发布到商城中交易, 这样有效避免数据集中存在违法数据等问题.

1.2 软件架构设计

大数据交易集市平台的软件架构设计如图 3 所示. 软件架构从上至下可以分为 3 层, 分别是 SaaS 层[2]、PaaS 层[3] 和 IaaS 层[4], 下面进行详细的介绍.

SaaS 层: 给用户提供的能力是使用在云基础架构上运行的云服务提供商的应用程序. 本系统的 SasS 层中包含前台、后台管理系统、应用沙箱.

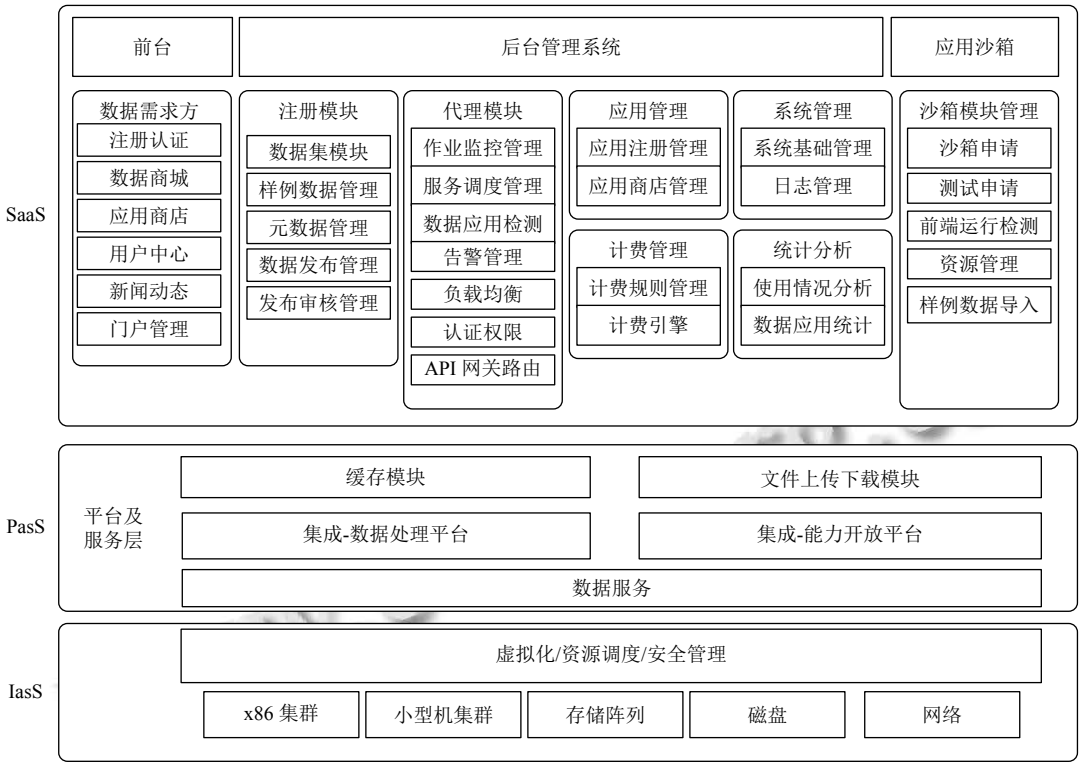

图 3 软件架构图

PaaS 层: 平台即服务作为云计算的一种, 是将基础设施平台作为一种服务呈现给用户的商业模式, 是一种低成本的方案[5]. 可以让用户使用由云服务提供商支持的编程语言、库和开发工具等开发应用程序, 并在相应的基础设施上部署. 本系统的 PaaS 层主要包含缓存模块、文件上传下载模块等.

IaaS 层: 它将硬件资源以服务的方式提供给云平台服务开发商, 这样就对后者屏蔽了基础设施的异构性[6]. 用户可以在云服务提供商提供的基础设施上部署和运行任何软件. 本系统的 IaaS 层主要实现虚拟化、资源调度、安全管理等功能, 硬件上有 X86 集群、小型集群等.

1.3 主要功能设计

下面介绍大数据交易集市主要的两个业务场景.

业务场景一: 数据发布和服务代理请求数据, 如图 4 所示. 本系统实现该功能可以分为以下几步.

(1) 数据交易集市为数据机构 A 和 B 部署交易端, 包括数据应用运行容器及数据审计网关;

(2) 数据机构 A 和 B 开发可交易数据接口;

(3) 数据机构 A 和 B 通过代理注册可交易的数据, 并通过数据采集接口提供样本数据;

(4) 数据请求方通过注册功能进行数据检索、察看样本, 找到需求数据;

(5) 请求方接受数据使用条款;

(6) 代理将数据提取应用部署至数据所有者侧 A 和 B, 启动应用对数据进行处理;

(7) 处理结果经 A 和 B 过滤、审计后再传回代理;

(8) 如果有多个数据源返回结果, 代理对结果进行过滤、融合处理 (过滤和审计操作按交易条件可选);

(9) 代理将结果传回请求方.

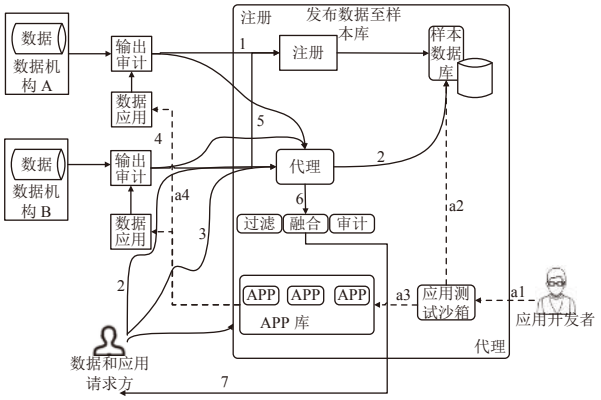

图 4 数据发布和服务代理请求调用数据图

业务场景二: 应用的测试发布、数据和应用服务请求及处理, 如图 5. 本系统实现该功能步骤如下.

(1) 开发者提出开发请求;

(2) 开发者利用样本数据在应用沙盒中进行数据处理功能和安全性测试;

(3) 数据交易集市对提交的应用进行安全和可用性测试后发布至应用商店;

(4) 数据交易代理将该应用部署至数据所有者侧并完成初始化, 等待用户调用.

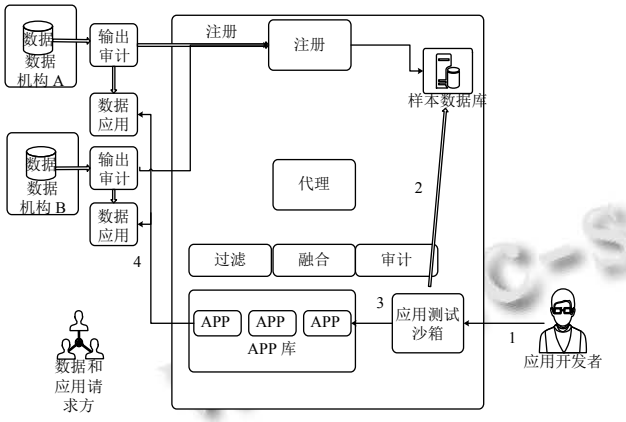

图 5 应用的测试发布、数据和应用服务请求及处理图

2 相关技术

2.1 Spring MVC

Spring MVC 框架实际上是 Spring 框架中的 Spring Web MVC 模块[7]. 其作为 Spring Framework 的后续产品, 融合在 Spring Web Flow 里面. Spring MVC 可以提供 MVC (model view controller) 软件开发模式, 并且支持 IOC 和 AOP.

2.2 Hive

Hive 由 Facebook 开发, 是一种构建于 Hadoop[8] 集群之上的数据仓库应用, 旨在实现 Hadoop 上的数据操作与传统 SQL 的结合[9]. 在 Hadoop 中用来处理结构化数据. 开发人员直接使用 Hadoop 会面临学习成本太高、项目周期要求太短、MapReduce[10] 实现复杂查询逻辑开发难度大等问题. Hive 通过将结构化的数据文件映射为一张数据库表, 并提供类 SQL 查询接口, 本质是将 SQL 转换为 MapReduce 程序, 提供快速开发的能力, 有效减少开发人员的学习成本.

3 系统实现

3.1 总体模块设计

在本文提出的大数据交易模式中, 由数据提供方在大数据交易集市平台上发布数据及相关的应用, 数据需求方根据需要在平台购买相应的数据集或服务, 如图 6 所示.

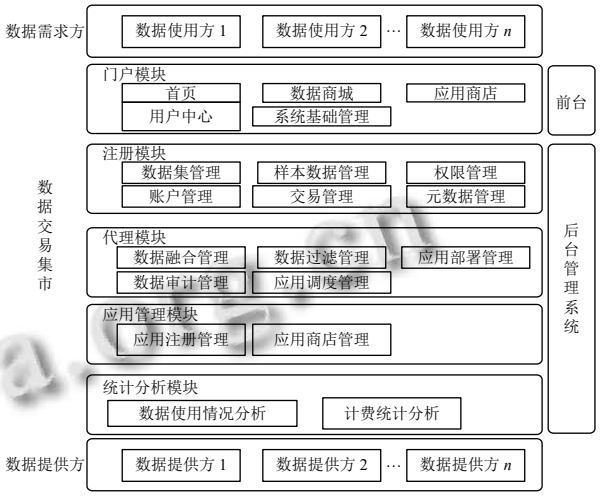

图 6 总体实施方案图

数据交易集市有前台和后台管理系统组成. 前台主要模块是门户模块, 该模块又可以分为多个子模块, 下面详细介绍几个主要的功能模块.

(1) 数据商城模块实现数据商城列表、商品详情. 展示数据机构发布的所有数据商品, 方便有需求的买家挑选购买.

(2) 应用商城模块实现应用商城列表、商品详情. 展示创客发布的所有应用商品, 方便有需求的买家挑选购买.

(3) API 商城模块实现 API 商城列表、商品详情. 展示商家发布的所有 API 商品, 方便有需求的买家挑选购买.

(4) 个人中心模块实现个人信息、用户认证、账户安全、白名单管理、消息通知等功能.

(5) 购物车模块, 用户购物活动中的购物车, 类似在淘宝、京东中的作用.

(6) 买家中心模块实现管理已购数据、已购应用、已购 API.

(7) 卖家中心功能有统计分析、已售数据、已售应用、已售 API、数据发布、应用发布、API 发布、应用鉴权等.

(8) 数据和应用发布, 展现卖家发布的数据和应用商品情况列表, 支持用户发布数据和应用商品, 通过审核以及数据机构应用鉴权后, 可上架销售.

后台管理系统由 3 部分组成, 其中数据及应用注册模块, 主要实现数据集及应用的定义及发布、元数据管理、抽样数据管理、数据交易记录等功能. 数据代理模块, 数据代理是数据交易的核心功能, 主要实现数据应用的部署及清理、需求数据获取及返回结果过滤或合并等.

应用管理模块, 主要为应用开发者提供开发环境及上线应用的管理等功能. 其中应用沙箱主要用于应用开发, 开发者可以在应用沙箱中利用样本数据进行应用开发, 测试完成的应用可通过发布功能在应用商店进行发布.

3.2 开发环境

编程语言: Java; 开发工具: STS (Spring tools suite); 操作系统: Windows 10; 数据库: MySQL 5.7; JDK 版本: JDK 1.8 版本; 浏览器: Google Chrome; 机器: Lenovo ThinkPad T470; 内存: 4 GB 以上; 硬盘: 500 GB 以上; 集群环境: Hadoop/Hive 集群是由 5 台与单机服务器组成. 每台服务器上安装 Ubuntu 系统, 并安装配置 Hadoop-2.7.3 和 Hive1.2.

3.3 项目架构

数据交易集市平台采用前端分离的模式开发, 整个系统有 3 个子项目组成, datamarket (大数据交易集市门户系统)、bjmarketportal (大数据交易集市后台管理系统)、datamarketReact (大数据交易集市前端页面). 如图 7, datamarket 和 bjmarketportal 通过 RESTful API 和 datamarketReact 页面进行交互. 后台使用 Java 语言在 STS 中开发. 前端页面系统 datamarketReact 采用 React 框架[11] 开发, 使用 webpack 前端构建工具, 采用 es2016 新一代 JavaScript 编码模式, 页面中的图表展示使用到 echarts, 前端可以通过后台的 RESTful API 与后台进行交互.

后台部分使用 Spring MVC 框架构建项目整体架构, 并采用了经典的 Web 3 层架构模型, MVC 模式把应用程序拆分成 3 个部分: 模型, 视图和控制器[12]. 即从高到低次是 Controller 层、Service 层和 Dao 层, 下面进行详细的介绍.

本项目在开发中, 通过在 STS 中配置 maven-3.5.0 实现对 Jar 包的管理, 并在 maven[13] 的配置文件 settings.xml 中完成指定本地仓库地址, 设置阿里云镜像, 部分代码如下所示. 通过 maven 避免了开发人员手工添加 Jar 包和 Jar 包版本冲突的问题.

```
<mirror>
<id>alimaven</id>
<name>aliyun maven</name>
<url>http://maven.aliyun.com/nexus/content/groups/public/</url>
<mirrorOf>central</mirrorOf>
</mirror>
```

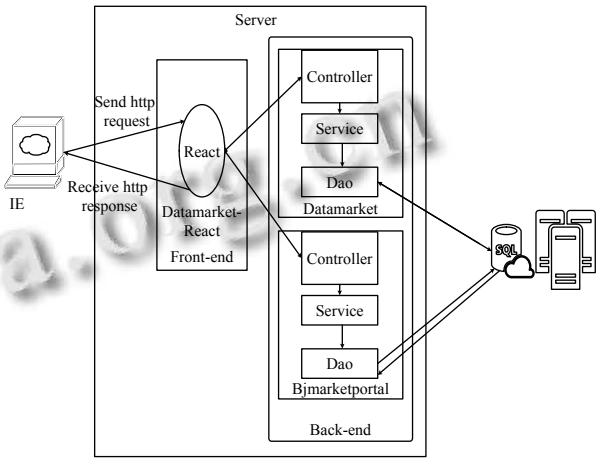

图 7 工程架构图

在传统的单体架构[14] 中, 登录用户的信息往往保存在 session[15] 中, 当登录用户增多不仅会增加服务器的压力, 同时在分布式应用上, 会限制负载均衡器的能力和应用的扩展性, 本项目采用基于 JWT (JSON Web token)[16] 的认证方式, 通过 pom.xml 文件实现在 Spring MVC 框架中集成 JWT.

在数据库的操作方面, MyBatis 的优势在于利用配置文件, 不需要在代码中编写 SQL 语句[17]. MyBatis Generator 插件可以根据表结构生成 JavaBean. 本系统实现了 Spring MVC 和 MyBatis、MyBatis Generator 的集成. pom.xml 配置文件部分代码如下:

```
<dependency>
<groupId>org.mybatis</groupId>
<artifactId>mybatis-spring</artifactId>
<version>${mybatis.spring.version}</version>
</dependency>
```

大数据集往往需求较大的存储空间, 为满足业务需求, 本系统使用 MySQL 保存用户信息、商品详情、样本数据等信息, 使用 Hive 保存存储数据公司提供的数据集.

在接口测试方面, 实现了 Spring MVC 和 Swagger 框架的集成, Swagger 是一个简单又强大的 RESTful 风格的 API 生成文档工具. pom.xml 配置文件部分代码

如下:

```
<dependency>
<groupId>com.mangofactory</groupId>
<artifactId>swagger-springmvc</artifactId>
<version>1.0.2</version>
</dependency>
```

项目在 Spring MVC 框架中还集成了分页插件 pagehelper、lombok、Redis 等技术.

3.4 高访问机制

在大数据交易集市平台中, 用户浏览数据商城, 查看样本数据集, 以及查看个人信息等操作都会涉及到对数据库进行 I/O 操作. 当用户访问量增大, 对数据库会进行频繁的读写操作, 系统在性能上会出现瓶颈, 本文采用 NoSQL 技术解决这一问题. Redis[18] 是现在最受欢迎的 NoSQL 数据库[19] 之一, Redis 包含多种数据结构、支持网络、基于内存、可选持久性的键值对存储数据库. 本文采用 Redis 实现系统的高访问机制, 当用户从系统获取数据时, 系统会首先从 Redis 中查找, 如果没有找到, 再从数据库中读取, 有效提高系统读取数据的性能, 缓解数据库访问压力, 过程如图 8 所示.

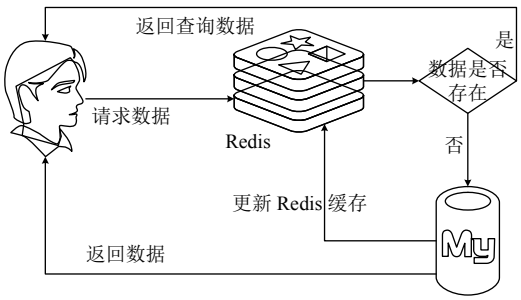

图 8 Redis 缓存机制

3.5 基于 RESTful 的 API 接口设计

3.5.1 RESTful API 介绍

API (application programming interface, 应用程序编程接口)[20] 用来提供应用程序与开发人员基于某软件得以访问的一组例程, 而开发人员无需关心其内部工作机制的细节. 在 REST (representational state transfer) 中每个资源由 URI 标识, RESTful[21] 是一种架构的规范与约束, 通过 POST, GET, PUT, DELETE 操作对资源进行 CRUD[22] 操作.

3.5.2 权限验证流程

系统提供的所有 API 必须通过认证鉴权才可使用. 权限验证流程如图 9 所示. 用户登录系统时, 前端会向后端传入登录验证信息, 如表 1 所示.

验证成功后, 服务器端会生成 Token. 之后将 Token 和该用户的权限保存到 Redis 中, 如表 2 所示.

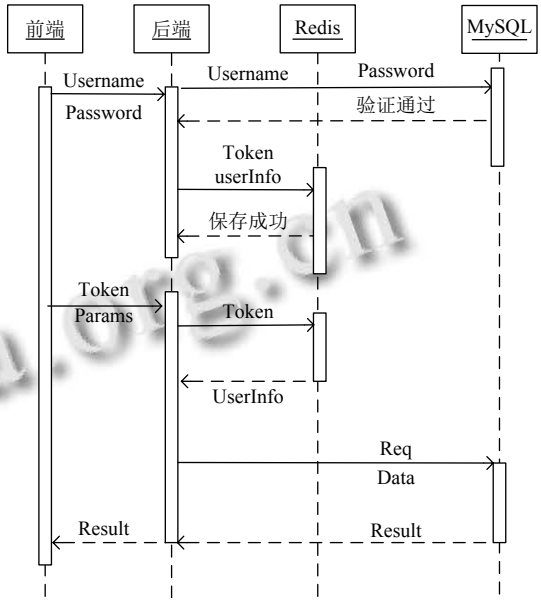

图 9 权限验证流程图

表 1 用户登录信息表

参数名	描述
username	用户名
password	密码

表 2 Token 信息表

键	值属性	描述
Token	userId	用户id
	username	用户登录名称
	roleId	角色id

本系统的 Token 由随机字符串、用户名称、时间戳拼接后 md5 摘要生成, 60 min 过期.

3.5.3 接口设计

接口数据传输采用 HTTP 协议, 提交和返回数据都为 JSON 格式[23], 并统一采用 UTF-8 字符编码. Spring MVC 框架中使用@RequestBody 注解把前端发送的 JSON 字符串转为 Java 对象, 使用@ResponseBody 注解把后台 Java 对象转换 JSON 字符串. 在返回数据方面, 失败时 re_message 字段返回失败原因. 下面通过本系统的后台管理 bjmarketportal 项目中数据商品管理模块详细介绍 API 接口的设计. 数据商品管理模块主要实现数据商品信息维护, 数据商品审核、发布等功能. 数据商品管信息管理模块的 API 接口设计如表 3.

表 3 数据商品管理模块接口

接口链接	应用场景	请求方式
/api/v1/datacommodity	发布数据商品	POST
/api/v1/datacommodity{id}	按id审核数据商品	PUT
/api/v1/ datacommodity	按条件查询数据商品详情	GET
/api/v1/datacommodity{id}	按id查询数据商品信息	GET

下面重点介绍其中的根据 id 查询数据商品详情, 接口链接地址: /api/v1/ datacommodity{id}, id 是用户从前端提交的请求参数, 请求方式是 GET 类型, 后端根据商品的 id 从数据库中查找商品详情信息, 如表 4.

表 4 数据商品表

字段	注释	备注
data_commodity_id	数据商品id	
data_set_commodity_name	数据商品名称	
img	数据商品图标	无文件返回“nofile”
file	数据数据样例文件	无文件返回“nofile”
data_charge_type_code	计费类型	1: 按次; 2: 一口价
data_charge_price	计费单价	
data_desc1	简介	
data_desc2	亮点	
data_desc3	应用场景	
data_desc4	使用方法	
example_is_download	是否允许下载样例数据	1: 允许; 2: 不允许
data_provider_name	数据提供方	
data_type	数据提供类型	
data_single	是否可以交易	1: 是; 2: 否
sale_counts	总销量	

查询结果会以成 JSON 格式发送到前端页面, 当查询成功时返回结果如下所示:

```
{
“data_commodity_id”: 12,
“data_set_commodity_name”: 全国天气数据集,
“data_charge_type_code”: “2”,
“img”: “nofile”,
“file”: “nofile”,
“data_charge_price”: “950”,
“data_desc1”: “全国天气数据集简介”,
“data_desc2”: “可以按照日期和城市查询天气数据”,
“data_desc3”: “天气预报, 应急救灾”,
“data_desc4”: “数据分析, 机器学习”,
“example_is_download”: “1”,
“data_type”: “txt 文本”,
“data_single”: “1”,
“sale_counts”: 150
}
```

后台执行成功, 将数据添加到数据库中, 如果执行失败, 会将失败信息以 JSON 格式发送到前端, 如下:

```
{“re_Message”: “57-2020-11-04-19:37-/datamarket/api/v1
/datacommodity -数据字典不存在****code”,
“returnMessageDisplay”: “服务器不稳定请稍后再试”,
“resultInfo”: null}
```

3.6 数据集购买

数据集购买功能位于 datamarket 项目的数据商城模块中, 数据商品可以按照行业分为政府、医疗、金融、电商、汽车、房地产、教育、旅游等. 大数据具有量大、价值密度低的特点, 用户在使用数据集进行数据分析、开发应用软件时, 往往不会使用到数据集中的所有数据, 需要清除掉无价值的信息, 本系统在购买数据商品功能基础上, 提供查看样本数据、按照字段筛选数据等功能.

数据机构首先上传数据商品的详情信息和样本数据, 后台管理员审核通过后, 会将数据商品发布到数据商城中. 创客根据自身需求, 查看数据商品详情和样本数据, 选择合适的数据商品, 完成支付后, 可以下载相应的数据集, 并且可以使用数据商城提供的数据筛选功能去除数据集中无价值的信息. 下面通过数据机构 A 提供的全国天气数据集 National_weather_data 详细介绍数据商品购买功能.

全国天气数据集 National_weather_data 记录了 2000 年到 2018 年全国各个城市每一天的天气数据, 每条记录字段包含: 城市、日期、天气状况、最高气温、最低气温、风力风向, 如表 5 所示.

表 5 全国天气数据样例表

年份	城市	日期	最高气温 (°C)	最低气温 (°C)	天气	风向	风力
2018	深圳	1205	26	23	多云	东南风	3
2018	深圳	1206	25	16	阴	东北风	1

数据机构 A 首先上传该数据集的详情信息和对应的样本数据. 数据机构首先创建样本数据集对应的表结构, 填写每个字段名称、字段的数据类型等, 如图 10.

图 10 设置样本数据集

数据机构 A 提交信息后, 后台会在 MySQL 数据库中创建对应的表, 使用 MyBatis 框架实现动态创建

表结构, 把样本数据名称作为表名, 字段名称和数据类型作为表的属性和相应的数据类型. 之后数据机构 A 在系统中填写样本数据表记录, 样本数据表的记录行数一般在 10–50 行之间, 部分代码如下:

```
<select id="createTable" parameter Type="com. datamarket. datashop.
exampledata.
CreateSql">
create table `${name}`
( <foreach collection="columnSqls" item="fields" separator=", ">
<include refid="commonSql"></include>
</foreach>
<if test="primaryKey"> , PRIMARY KEY (`${primaryKey}`)
```

系统管理员会对数据商品的介绍信息和样本数据内容进行审核, 检查数据商品是否符合商城的规定, 有无违法信息, 审核通过后授权数据机构向 Hive 集群上传数据商品.

为确保数据提供者的数据安全性, 系统会向数据提供者提供 Hive 集群资源, 数据提供者可以上传、修改、查看、删除数据集, 数据机构和客户达成购买协议, 并完成交易后, 数据交易集市才可以获得相应数据集的获取权限, 数据代理根据需求从数据所有者获取数据交给客户. 下面通过 national_weather_data 数据集进行详细介绍.

数据机构 A 将数据集中的数据保存到 txt 文件中, 命名为 national_weather_data.txt, 文件中每一行代表一条数据记录, 每条数据记录中的一列代表一个属性值, 每个属性之间用英文逗号隔开. 当数据商品详情和样本数据被后台管理员审核通过后, 会通过数据商城的 API 接口将 national_weather_data.txt 中的数据导入到 Hive 集群中管理, 过程如图 11 所示.

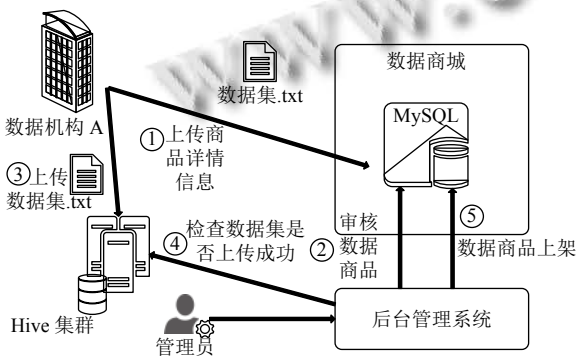

图 11 数据机构上传数据商品图

Hive 集群保存数据集的过程如下:

(1) 表名以及表中的字段名称和样本数据表 national_weather_data 相对应, MySQL 中的字段数据类型, VARCHAR 和 FLOAT 分别对应 Hive 中的 string 和 int 类型.

(2) 为保证数据集内容的充足性, 系统输出审计功能会检查 txt 文件的行数, 当行数小于 100 行时, 停止上传数据, 并反馈数据机构数据量过小, 不符合要求. 同时随机抽取其中 5 行记录供管理员查看, 审查其内容是否符合数据商品的详情描述, 部分代码如下:

```
if [ ${BEFOR_VALUE} -lt 100 ]; then
echo "`date` - FAILURE count(*) = $BEFOR_VALUE"
echo "FAILURE"
```

(3) 数据机构向管理员提供 Hive 表的分区字段, national_weather_data 表的分区字段是 year 和 city.

(4) 后台程序生成建表语句, 如下所示, 在 Hive 集群中建立表结构.

```
create table national_weather_data
(year string,
date string,
city string,
maximum_temperature int,
minimum_temperature int,
weather string,
wind_direction string,
wind_rating int,
)PARTITIONED BY ( `year` string, `city` string)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|' NULL DEFINED AS ''
STORED AS TEXTFILE;
```

(5) 执行导数脚本, 把 txt 文件中保存的数据导入到 Hive 集群中, 执行脚本时需要传递 3 个参数, table_name 表名, year 年份, city 城市.

```
LOAD DATA LOCAL INPATH '${FILE_NAME}' OVERWRITE
INTO TABLE ${table_name} partition (year='${YEAR}', city='${
CITY }');"
```

(6) 数据集上传结束后, 为确保数据集上传成功, 系统审计功能会检查上传到 Hive 中的数据集行数, 如果小于 txt 文件的行数, 则向数据机构反馈, 数据上传失败. 部分代码如下:

```
echo "hive table rows:$AFTER_VALUE"
if [ $AFTER_VALUE -ne $BEFOR_VALUE ]; then
echo "`date` - FAILURE:AFTER_VALUE(*) = $AFTER_VALUE <>
BEFOR_VALUE(*) = $BEFOR_VALUE "
echo "FAILURE"
current_2=`date +%s`
current_dura=`expr $current_2 - $current_1`
```

数据商城提供数据集的信息筛选功能, 客户可以根据数据集的字段选择需要的数据, 去除无用的数据项, 如图 12 所示. 后台系统会根据用户选择的字段生成 HQL[24] 语句, 从 Hive 集群中将相应的数据导出到 txt 文件中, 之后用户可以下载相应的 txt 文件. 在商品支付方面, 创客可以通过支付宝、微信、网银等手段完成支付, 这个过程类似淘宝、京东等网站商城.

图 12 选择需要的字段

3.7 模块企业注册功能

用户注册如果想要成为买家、卖家、创客等, 就需要进一步认证. 认证通过后, 用户就拥有购买、使用商品的权限, 可以发布、管理商品, 甚至成为一名平台创客, 申请创客空间资源, 开发、测试、运行应用. 认证分为个人认证和企业认证.

企业认证需要用户提供公司名称、地址、三证合一号码、营业执照副本, 以及运营者姓名、身份证号、联系方式、持证照正反面等信息, 如图 13 所示, 如确认用户信息真实有效, 提出申请后 7 个工作日内就会得到回复. 如被驳回, 按照驳回原因, 重新修改提交.

图 13 企业认证

下面详细介绍企业认证功能的实现. 用户点击提交按钮后, 前端会调用 bjmarketportal 项目中的企业认证接口, 接口地址是/api/v1/enterprisecertification, 访问方式是 POST, 表单中的数据会以 JSON 格式发送到后台. bjmarketportal 中实现企业认证功能的代码由 Controller 层、Service 层、Dao 层、Pojo (plain ordinary Java object)、Vo (value object) 等几部分组成, 下面详细介绍后台企业认证功能的实现.

Pojo: Pojo 类可方便开发人员使用数据库中的数据表. 通过 MyBatis Generator 插件从 MySQL 数据库中生成企业认证信息表对应的 Pojo 类—CorporateInformation.

Vo: 常用于业务层之间的数据传递, 可以用于封装从页面表单提交的数据. 该模块中 Enterprise CertificationVo 类用于封装企业认证表单数据.

Controller 层: 接收客户端的请求, 然后调用 Service 层业务逻辑, 获取到数据, 传递数据给视图. Corporate Information Controller 类对应该模块 Controller 层代码. 类中使用@Autowired 注解实现 Service 层 Corporate Information Service 对象的自动装配; 并创建 enterprise Certification 方法实现企业认证功能, 通过@Request Mapping 注解, 设置该方法的访问路径/api/v1/user Center/enterprisecertification, 访问方式是 POST.

Service 层: 用于实现模块中的业务逻辑 Corporate-InformationService 类对应该模块的 Service 层代码, 通过@Service 注解声明其是一个 Service 类, 并自动注册到 Spring 容器中; 通过@Autowired 获取 Dao 层 CorporateInformationMapper 对象; certification 方法实现企业认证功能.

Dao 层: 负责模块中数据库操作, CorporateInfor-mationMapper 类对应该模块中的 Dao 层代码, 采用 MyBatis 框架实现, 通过@Mapper 注解其是一个 Dao 层类, 并交給 Spring 容器管理 , 同时省去了 mapper 映射文件的书写. 该模块部分代码如下:

```
@Controller
@RequestMapping(value = "/mail/userCenter")
public class CorporateInformationController extends BaseController
<Object> {
   @Autowired
   private SandboxieService sandboxieService;
   @Autowired
   private CorporateInformationService corporateInformationService;
   @Autowired
   private DataShopService dataShopService;
   @RequestMapping(value = "enterprisecertification",
   method=RequestMethod.POST)
   @ResponseBody
    public String enterpriseCertification(EnterpriseCertificationVo
enterpriseCertificationVo, HttpServletResponse response) throws
Exception {
String enterpriseName=enterpriseCertificationVo.getEnterpriseName();
```

```
if (corporateInformationService.isEnterprise(enterpriseName)) {…}
…
}
```

4 结论与展望

本文设计的大数据交易集市平台, 通过向数据机构提供公用的 Hive 集群资源, 实现数据交易集市不存储数据, 数据由提供者存储和管理, 在商城发布数据商品的详情信息、样例数据供客户查看, 由数据代理获取数据交给需求方, 有效解决了数据交易中私有数据持有者担心数据泄露, 不愿意分享数据的问题. 同时设计出数据发布和服务代理请求数据、应用的测试发布、应用服务请求业务流程. 并在此基础上使用 Spring MVC 框架, 并集成了 MyBatis、Swagger 等技术开发实现了大数据交易集市平台, 利用 Redis 解决系统在高并发下中性能下降问题, 采用基于 RESTful 风格的 API 接口设计, 最后通过数据集购买功能和企业认证功能详细介绍了项目的实现, 并且系统具有前后端分离、低耦合高内聚以及易扩展性.

参考文献

1 徐计, 王国胤, 于洪. 基于粒计算的大数据处理. 计算机学报, 2015,38(8):1497–1517.[doi:10.11897/SP.J.1016.2015.01497]

2 Benlian A, Hess T. Opportunities and risks of software-as-a-service: Findings from a survey of IT executives. Decision Support Systems, 2011, 52(1): 232–246. [doi: 10.1016/j.dss.2011.07.007]

3 Lawton G. Developing software online with platform-as-a-service technology. Computer, 2008, 41(6): 13–15. [doi: 10.1109/MC.2008.185]

4 Bhardwaj S, Jain L, Jain S. Cloud computing: A study of infrastructure as a service (IAAS). International Journal of Information Technology and Web Engineering, 2010, 2(1): 60–63.

5 董振江, 王雪, 胡洁, 等. 一种多层次的云平台安全解决方案. 电子技术应用, 2013, 39(4): 133–136. [doi: 10.3969/j.issn.0258-7998.2013.04.040]

6 张龙昌, 褚庆, 杨艳红. 三网融合下数字图书馆云服务平台架构研究. 计算机与数字工程, 2015, 43(11): 1974–1980, 2005. [doi: 10.3969/j.issn.1672-9722.2015.11.017]

7 吴婉楠. 基于 SpringMVC 和 MyBatis 框架的炒股比赛系统的设计与实现 [硕士学位论文]. 南京: 南京大学, 2016.

8 Taylor RC. An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics. BMC Bioinformatics, 2010, 11 Suppl 12(S12): S1.

9 肖凯. 基于 Hive 架构的电力设备状态信息数据仓库的研究 [硕士学位论文]. 北京: 华北电力大学, 2013.

10 Dean J, Ghemawat S. MapReduce: Simplified data processing on large clusters. Communications of the ACM, 2008, 51(1): 107–113. [doi: 10.1145/1327452.1327492]

11 League of Extraordinary Developers. React: 引领未来的用户界面开发框架. 寸志, 范红春, 杨森, 等译. 电子工业出版社, 2015.

12 宋涛. 基于 MVC 模式的 J2EE 项目架构的可重用性应用及研究 [硕士学位论文]. 武汉: 中国地质大学 (武汉), 2006.

13 Teng F, Wu QW. Design and implementation of the information system of retired veteran cadres bureau based on SpringBoot framework. Proceedings of 2021 IEEE International Conference on Consumer Electronics and Computer Engineering (ICCECE). Guangzhou: IEEE, 2021. 87–92. [doi: 10.1109/ICCECE51280.2021.9342126]

14 王纪军, 张斌, 顾永生, 等. 云环境中 Web 应用的微服务架构评估. 计算机系统应用, 2017, 26(5): 9–15. [doi: 10.15888/j.cnki.csa.005746]

15 周华中, 张少娟, 朱俊杰. Session 在 Web 编程中的应用. 微机发展, 2002, 12(3): 1–4.

16 Solapurkar P. Building secure healthcare services using OAuth 2.0 and JSON Web token in IOT cloud scenario. 2016 2nd International Conference on Contemporary Computing and Informatics (IC3I) IEEE, 2016.

17 杨帆. 基于 Spring 和 MyBatis 框架的物业管理系统的设计与实现 [硕士学位论文]. 武汉: 华中科技大学, 2016.

18 Redis. http://redis.io. (2016-01-28).

19 申德荣, 于戈, 王习特, 等. 支持大数据管理的 NoSQL 系统研究综述. 软件学报, 2013, 24(8): 1786–1803.

20 Miller FP, Vandome AF, Mcbrewster J. Application programming interface. Saarbrücken: Alphascript Publishing, 2010. 105–185.

21 Dhalla HK. Benchmarking the performance of RESTful applications implemented in spring boot Java and MS. Net core. Journal of Computing Sciences in Colleges, 2020, 36(3): 178.

22 陈晴, 朱洲森, 梁静, 等. 基于同构型分布式数据库一体化 CRUD 操控适配解析器的方法. 中国, 110059102A. 2019-07-26.

23 Wang GH. Improving data transmission in web applications via the translation between XML and JSON. 3rd International Conference on Communications & Mobile Computing IEEE Computer Society, 2011.

24 谢恒, 王梅, 乐嘉锦, 等. 基于 Hive 的计算结果特征提取与重用策略. 计算机研究与发展. 2015, 52(9): 2014–2024.