

# 基于 Spring Boot 地质档案信息系统<sup>①</sup>



雷震, 焦学军, 张惠怡

(河南省地质矿产勘查开发局 测绘地理信息院, 郑州 450006)  
通信作者: 焦学军, E-mail: [576723484@qq.com](mailto:576723484@qq.com)

**摘要:** 为了实现对地质档案的信息化管理, 系统结合 Gateway 服务网关和 Consul 注册中心, 构建了一个基于 Spring Boot 微服务架构的地质档案信息系统. 研发过程使用前后端分离的开发模式, 通过 Layui 开发前台页面的主要部分, 并结合 Spring Boot 框架搭建后端微服务实例, 以关系型数据库 MySQL 及非关系型数据库 Redis 共同作为系统数据存储的载体, 建立了用户管理、档案入库、档案借阅归还、OCR 图片识别等功能模块. 系统将地质档案进行电子化存储, 促进了资源共享、统一调用, 减少了人员的维护工作量, 提高了工作效率, 为地质档案信息数据融合提供了支撑.

**关键词:** Spring Boot; Layui; 前后端分离; 地质档案

引用格式: 雷震, 焦学军, 张惠怡. 基于 Spring Boot 地质档案信息系统. 计算机系统应用, 2022, 31(12): 104-111. <http://www.c-s-a.org.cn/1003-3254/8834.html>

## Geological Archives Information System Based on Spring Boot

LEI Zhen, JIAO Xue-Jun, ZHANG Hui-Yi

(Surveying and Mapping Geographic Information Institute, Henan Geological and Mineral Exploration and Development Bureau, Zhengzhou 450006, China)

**Abstract:** To realize the information-based management of geological archives, this study constructs an information system for geological archives, which is based on Spring Boot microservices architecture and is in systematic combination with the service Gateway and Consul registry. In the research and development process, the development mode of front and rear end separation is employed, and the main part of the front page is developed through Layui. By the Spring Boot framework, a back-end microservice example is built. With both the relational database MySQL and non-relational database Redis as the storage carrier of system data, functional modules are established, such as user management, archive warehousing, archive lending and return, and OCR image recognition. In this system, the geological archives are stored electronically, which promotes the sharing and unified transfer of resources, reduces the maintenance workload of personnel, improves work efficiency, and provides a reference for the data fusion of geological archive information.

**Key words:** Spring Boot; Layui; front and rear end separation; geological archives

随着信息技术的不断发展与普及, 企事业单位对纸质档案的电子化建设与管理越来越迫切, 如何科学地保管和有效利用档案是管理者亟需思考的问题. 建立档案信息化系统来实现电子化的档案管理, 是档案工作的发展方向, 同时也是提升档案工作质量, 把档案

信息资源最大限度激活的必然要求<sup>[1]</sup>.

项目以河南省地矿局档案室为依托, 设计建设地质档案信息化系统, 以实现全面化的信息管理、自动化的档案馆业务、精细化的档案馆藏资源管理、知识化的档案信息服务等内容<sup>[2]</sup>, 达到地矿局内各部门同时

<sup>①</sup> 基金项目: 河南省档案局科技项目 (2020-X-25)

收稿时间: 2022-03-29; 修改时间: 2022-04-22; 采用时间: 2022-04-29; csa 在线出版时间: 2022-07-29

组织录入、借用档案的最终目标,及系统操作简单、访问速度快、数据共享、安全存储和免维护的最终效果。

同时,在地质档案信息化系统开发时需要考虑与其他系统的对接以及本系统易扩性,且遵守高内聚低耦合的设计原则,有必要采用扩展性较强、开发便捷的开发技术,因此项目选取采用基于 Spring Boot 微服务框架技术进行设计实现。

## 1 系统关键技术

### 1.1 Spring Boot 微服务

Spring Boot 微服务框架是大量开源库的集合被十分广泛的应用在互联网行业,其可将系统应用拆分成多个核心功能<sup>[3]</sup>。

每项功能都可看作是一个微服务,可以进行独立的构建、部署,因此在底层业务架构上若存在多个微服务,它们之间是不存在相互影响的。所以,微服务的发布只需抽取系统公共逻辑业务,并把这些业务编制成框架组件即可。

这种方式为系统业务提供了便于延展的基础支撑,同时可以大范围减少系统性能损耗及服务调用的频率,符合低耦合的设计原则<sup>[4]</sup>。因此,基于 Spring Boot 微服务框架下开发的档案管理系统,在技术上提供了更灵活的低层逻辑和快速开发能力。

### 1.2 Gateway 服务网关

Gateway 是 API 网关服务,其构建是基于 Spring 生态系统。它的目的是提供一种简单而有效的方式对 API 进行路由,以及提供一些强大的过滤器功能<sup>[5]</sup>。

系统中 Gateway 的主要配置是 cors-configurations 相关项中开启跨域:携带认证信息、跨域的源(网站域名/ip)、跨域请求里的 head 字段、跨域的 method、跨域有效期等; routes 相关项中开启路由转发功能,添加 name 为 system-service-provider-lb 等路由信息,当请求 <http://localhost:8506/system-service-provider-lb/xxx> 时会转发到对应名称的服务处。相关配置代码如下所示。

```
server:
port: 8506 #端口号
spring:
  application:
    name: gateway-provider-payment-lb
  http:
```

```
encoding:
  charset: UTF-8
  force: true
cloud:
  gateway:
    globalcors:
      cors-configurations:
        ['/**']:
          allow-credentials: true #认证信息
          allowed-origins: "*" #跨域的源, *代表全部
          allowed-headers: "*" #跨域请求里的 head
          allowed-methods: #跨域的方式
            - OPTIONS
            - GET
            - POST
          max-age: 3600 #跨域有效期
    locator:
      enabled: false
      lowerCaseServiceId: true
    routes: #路由信息配置
      - id: lb_test_route-lb
        uri: lb://system-service-provider-lb
        predicates:
          - Path=/system-service-provider-lb/**
        filters:
          - StripPrefix=1
```

### 1.3 Consul 注册中心

Consul 用于实现分布式系统的服务发现与配置,特点有键值存储、多数据中心等。其中键值存储是指可以根据用户需要通过 key/value 形式根据存储动态配置:多数据中心是用户若要把业务扩展到多个区域,无需建立额外的逻辑层内容<sup>[6]</sup>。

系统 Consul 采用的是 Windows 下 1.6.0 版本,只需在 bat 类型的文件下添加 Consul 启动配置即可,采用此种形式可以简化部署配置的流程。相关配置代码如下。

```
%1 mshta vbscript:CreateObject("WScript.Shell").Run("%~s0 %*", 0, FALSE)(window.close)&&exit
%consul agent -server%
consul agent -dev
```

## 2 系统设计

### 2.1 系统开发工具

系统采用 IntelliJ IDEA 开发平台,涉及 Java、Html、CSS、JavaScript、Layui、ECharts 等多种开发语言,利用 MySQL 关系型数据库及 Redis 非关系型数据库等

存储工具,构建地质档案信息化系统。

## 2.2 系统总体架构

系统采用前、后台相分离的方式进行开发设计,系统架构如图1所示。

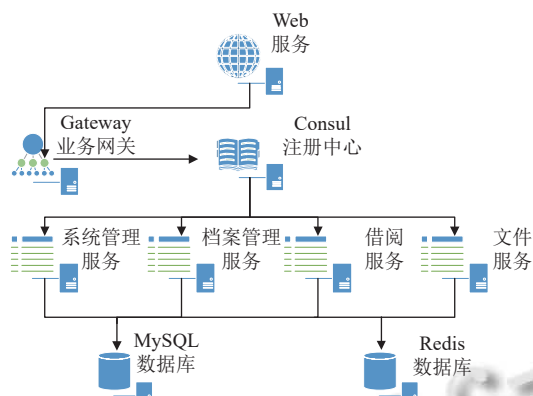


图1 系统架构

Web服务以Layui作为前端框架,使用Layui搭建页面可使得页面美观、内容丰富<sup>[7]</sup>;并结合Echarts插件优秀的绘图能力,可以方便地绘制系统中各种数据图形,如曲线图、柱状图、饼状图、散点图和地图等<sup>[8]</sup>。

后端使用Java作为开发编程语言,采用Spring Boot框架通过搭建Gateway、Consul,把各种服务注册到Consul中,当有请求发生时Gateway网关可以根据设定的方式进行匹配,从而把请求分发到不同的后端服务上。

后端的系统管理服务用来实现用户管理、全局配置等;档案管理服务用来实现档案基础数据录入、统计分析等;借阅服务是把借阅用户数据与档案数据相结合,实现借阅业务;文件服务是把已录入的档案基础数据与电子化的档案资源进行关联结合,提供对文档资源的上传、下载、OCR识别功能等。

数据层采用MySQL关系型数据库和Redis非关系型数据库。其中,开源MySQL核心程序采用完全的多线程编程,可以灵活地为用户提供服务,而不过多的占用系统资源,符合系统的基础支撑需求,同时选用支持事务、外键、行级锁等功能的InnoDB作为MySQL存储引擎来存储数据,保证数据的不丢失、不冗余;开源Redis是一种基于内存的高性能数据库,键值对(key-value)作为其存储方式,存储读写能力可达10w QPS<sup>[9]</sup>,可支持如链表、集合等多种数据形式,可被用于数据缓存、或消息队列等,因此选用Redis来维护当前用户

与系统之间的关系,键值对中的key值由当前用户账号、密码及时间经签名后作为本次登录的唯一标识token,value值以JSON字符串的形式存储当前用户的登录信息。

## 2.3 系统功能模块

地质档案信息化系统分为3个模块,分别是系统管理、档案管理、个人借阅管理。系统模块如图2所示。

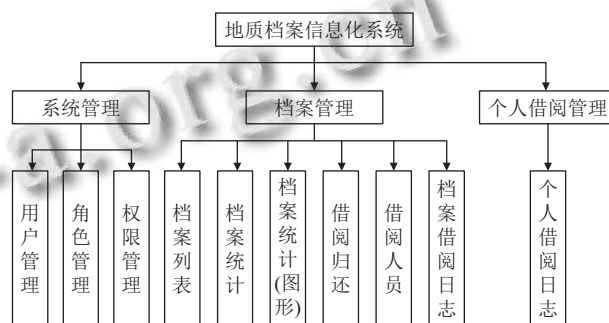


图2 系统模块

系统管理包含3个子功能,分别是用户管理、角色管理、权限管理。用户管理是新增或修改不同类型的用户信息,如档案管理员、借阅人员等;角色管理是赋予用户不同的角色;权限管理则是赋予角色不同的权限。

档案管理包含6个子功能,分别是档案列表、档案统计、档案统计(图形)、借阅归还、借阅人员、档案借阅日志。档案列表是新增或修改档案信息,上传相关的文件材料,并且可以根据需要进行OCR图片识别;档案统计是以文字列表的方式统计显示不同类型档案数量及借阅次数;档案统计(图形)是以图形的方式统计显示不同档案数量占比情况;借阅归还是对用户借还请求的审批操作,即将某档案借阅给某借阅人员,或者某借阅人员归还某档案;借阅人员是新增或修改借阅者的信息;档案借阅日志是根据不同筛选条件进行档案借阅记录的查询。

个人借阅管理包含一个子功能,即个人借阅日志。个人借阅日志是借阅人员登录个人账号,查询个人的借阅历史。

## 2.4 系统权限设计

根据实际的业务要求,系统有3种不同类型权限账号,分别超级管理员、档案管理员、借阅人员。系统权限如图3所示。

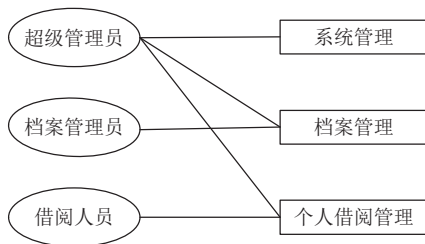


图3 系统权限

图3中, 超级管理员具有最高等级权限, 可以操作系统中的系统管理、档案管理、借阅管理等模块; 档案管理人员, 只能操作系统中的档案管理模块; 借阅人员, 只能操作个人借阅管理模块。

### 2.5 系统数据表设计

通过对系统进行功能分析, 设计出地质档案信息化系统的相关数据表. 数据表的设计分为了两类, 分别是用户关联表、档案关联表, 如图4所示。

主要的数据库表如下。

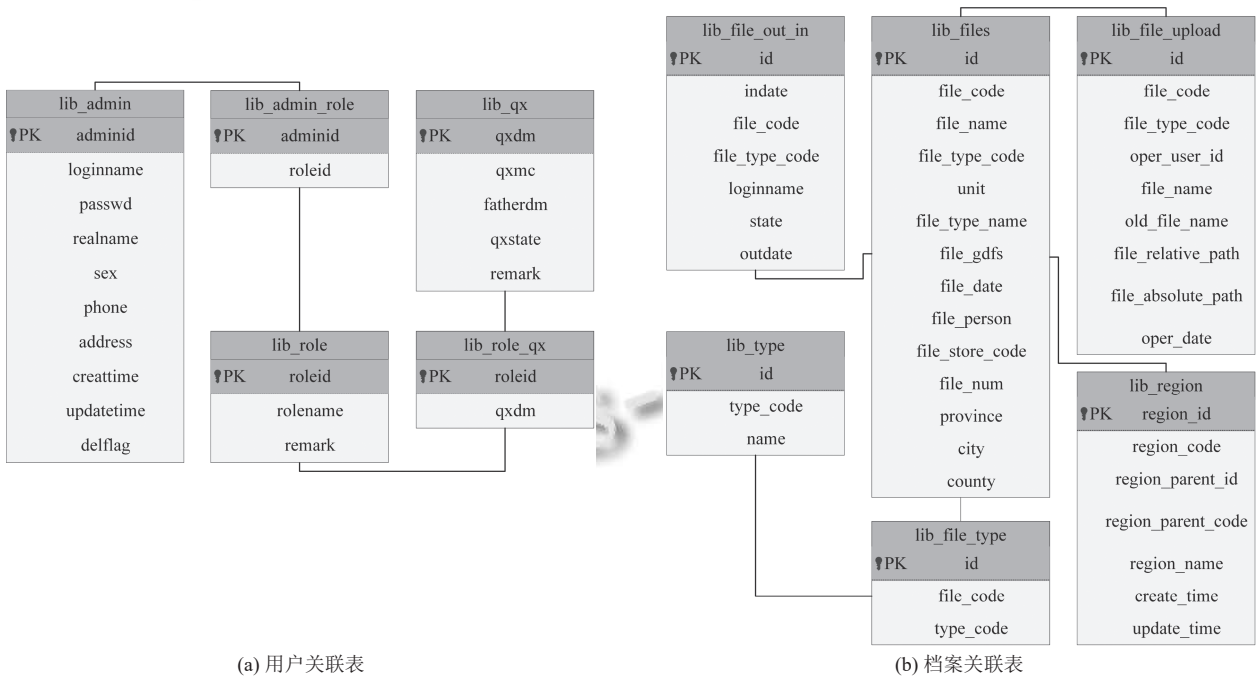


图4 系统数据表设计

## 3 系统实现及算法

### 3.1 虚拟路径映射

外部访问已存在的资源内容 (如已电子化的地质档案图片), 若不设置虚拟路径系统是无法直接获取资源的. 在系统后端进行虚拟路径映射的代码配置, 其中

#### (1) 用户表 lib\_admin

用户表用于存储用户的信息, 包括 ID、登录账号、登录密码、姓名、性别、地址、创建时间、是否删除等字段。

#### (2) 档案表 lib\_files

档案表用于存储档案信息, 包括 ID、档案代号、档案名称、档案单位、归档日期、归档人、存放位置代码、档案归属城市代码等字段。

#### (3) 档案材料上传记录表 lib\_file\_upload

档案材料上传记录表用于记录上传档案电子材料信息, 包括 ID、档案代号、文件名称、电子文件存储位置绝对路径、电子文件存储位置相对路径、上传用户 ID、上传时间等字段。

#### (4) 档案借阅记录表 lib\_file\_out\_in

档案借阅记录表用于记录档案借阅操作的信息, 包括 ID、档案代号、借阅人账号、借阅状态、借阅时间、归还时间等字段。

addResourceHandler 指的是外部访问时展示的路径, 即虚拟路径; addResourceLocations 指系统中文件存放的物理路径, 即实际路径. 相关配置代码如下。

```

@Configuration
public class UploadConfig extends WebMvcConfigurerAdapter {

```



```

@Override
public void addResourceHandlers(ResourceHandlerRegistry registry){
    String virtualPath = "虚拟路径";
    String realPath = "实际路径";
    registry.addResourceHandler("/" + virtualPath + "**").
        addResourceLocations("file:" + realPath);//映射操作
}

```

### 3.2 分页算法

对于访问某些页面,如“档案列表”等,其档案信息数据量是十分巨大的.如果全部查询显示,不仅页面排版显示会出现问题,而且响应时间长,影响用户体验.系统结合前端 Layui 的 Table 组件显示方式,在后端进行分页的代码设计,其思想是利用 HashMap 函数获取每条数据的信息,以<key, value>形式输出至 ArrayList 中,并根据前端 Table 组件接收数据的语法格式,组装数据传回前台页面.相关算法实现如下.

```

ArrayList data = new ArrayList();
int start = (pageIndex-1) * pageSize;
int end = start + pageSize;//计算起始、结束的位置数量
for (int i = 0, l = alist.size(); i < l; i++){
    //取出数据存入 HashMap 中
    HashMap record = (HashMap) alist.get(i);
    if (record == null) continue;
    if (start <= i && i < end){
        data.add(record);//存储符合条件的数据
    }
}
HashMap result = new HashMap();//组成数据传回前台
result.put("data", data);
result.put("total", alist.size());

```

### 3.3 借阅归还算法

系统在处理借阅档案时,首先获取档案信息,然后判断档案的状态及可借数量,如果满足借阅条件,则获取借阅人员信息并建立与档案之间的关联,之后修改档案状态和可借数量;在处理归还档案时,首先获取借阅人员与档案关联信息,然后修改档案状态和可借数量.相关算法实现如下.

```

/*借阅代码*/
FileList fileInfo = new FileList();
//档案状态"0"代表可借阅
if("0".fileInfo.getStatus() && fileInfo.getAmounts() > 0){
    Reader reader = new Reader();
    LendList lendList = new LendList();
    lendList.setReaderId(reader.getReaderId());//借阅者 ID

```

```

lendList.setFileId(fileInfo.getFileId());//档案 ID
lendList.setLendDate(new Date());//借阅时间
LendServer lendServer = new LendServer();
//提交借阅者与档案关联信息,进行借阅操作
lendServer.addLendRecordInfo(lendList);
//档案可借数量减 1
fileInfo.setAmounts(fileInfo.getAmounts() - 1);
if(fileInfo.getAmounts() <= 0){
    //变更档案状态,"1"代表不可借阅
    fileInfo.setStatus("1");
}
}
/*归还代码*/
FileList fileInfo = new FileList();
Reader reader = new Reader();
ReturnList returnList = new ReturnList();
LendServer lendServer = new LendServer();
//根据借阅者 ID、档案 ID 获取借阅信息
returnList.setRecord = lendServer.queryLendRecordInfo(
    reader.getId(), fileInfo.getFileId());
returnList.setReturnDate(new Date());//归还时间
ReturnServer returnServer = new ReturnServer();
returnServer.addRecordInfo(returnList);//归还操作
//档案可借数量加 1
fileInfo.setAmounts(fileInfo.getAmounts() + 1);
if("0".fileInfo.getStatus() && fileInfo.getAmounts() > 0){
    fileInfo.setStatus("0");//变更档案状态,"0"代表可借阅
}

```

### 3.4 OCR 图片识别算法

系统结合 tesseract 光学字符识别引擎,提供图像文字转换成可下载文本的 OCR 图片识别功能.在后端文件服务中进行 OCR 图片识别的代码设计,首先是获取图片,然后进行图片识别操作,最后显示识别的文本内容并提供下载功能,相关算法实现如下.

```

ITesseract instance = iniOCR();
//根据图片路径,实例化文件
File file = new File(path);
String filepath = file.getParent();
String filename = file.getName();
String result = instance.doOCR(file);//进行图片识别操作
//把识别的结果存到 TXT 文件中,提供下载文本
orcToTxt(filepath, filename, result);

```

## 4 压力测试

采用 Apache Jmeter 开源测试工具对系统进行压力测试,测试环境如表 1 所示.

在 Apache Jmeter 设置中,Ramp-up 含义是线程需要多长时间全部启动,如果线程数为 200,准备时长为

10 s, 那么需要每秒启动 20 个线程. 本次测试采用线程量 100, 准备时长 1 s, 循环 100 次, 共计 10000 次请求. 压力测试结果数据如图 5 所示.

通过压力测试结果可看出, 平均响应时间为 653 ms; 异常为 0; 吞吐量为每秒 141.7. 运行过程中未出现任何异常, 系统并发性、接口吞吐量、业务成功率等均符合应用需求, 满足日常多部门同时使用的性能方案.

表 1 测试环境

环境	客户端	服务器
系统	Windows 10	Windows Server 2008 R2 Standard
CPU	11th Gen Intel(R) Core(TM) i5-11400H @ 2.70 GHz 2.69 GHz	Intel (R) Xeon (R) CPU E3-1225 v5 3.30 GHz 3.30 GHz
内存 (GB)	16	16



图 5 压力测试结果

## 5 系统效果

### 5.1 用户管理

超级管理员的用户管理视图如图 6 所示, 此类用户选择用户管理后, 通过该视图可以对其他用户进行操作, 如查看、添加、编辑、删除、密码修改、赋予权限等.

### 5.2 借阅归还

档案管理员的借阅归还视图如图 7 所示, 此类用

户选择借阅归还后, 通过该视图可以对档案及借阅人员关联操作.

### 5.3 个人借阅日志

借阅人员的个人借阅日志视图如图 8 所示, 此类用户选择个人借阅日志后, 通过该视图可以查看、导出档案借阅历史, 及借阅状态, 并可通过 OCR 识别档案电子内容后下载如图 9 所示.



图 6 用户管理视图



图7 借阅归还视图



图8 个人借阅日志视图



图9 OCR 图片识别视图

## 6 结论

系统基于 Spring Boot 微服务框架, 及 Gateway 服务网关和 Consul 注册中心等相关技术, 配合使用 Java、Layui、MySQL 关系型数据库、Redis 非关系型数据库等开发语言及存储工具, 结合实际需求, 设计实现了地质档案信息化系统. 系统做到了档案的全过程管理, 实现了不同类型人员登录的权限验证, 及档案入库、借阅归还、OCR 图片识别等功能, 使得档案管理员与借阅人员之间的交互更方便、更规范、更有效, 提升了工作质量与工作效率.

### 参考文献

- 1 容依媚, 肖秋会. 我国智慧档案馆研究综述. 北京档案, 2019, (8): 15-18. [doi: 10.3969/j.issn.1002-1051.2019.08.005]
- 2 任越, 问宇鹏. 智慧档案馆知识服务的构成要素及服务层次探析. 山西档案, 2021, (3): 15-23. [doi: 10.3969/j.issn.1005-9652.2021.03.003]
- 3 郑若鹞. 应用 Spring Boot 的后勤报修系统的设计与实现. 福建电脑, 2021, 37(10): 66-69.
- 4 黄瑞泉. 基于 Spring Boot 框架的地图监管系统. 计算机系统应用, 2021, 30(8): 89-95. [doi: 10.15888/j.cnki.csa.008020]
- 5 牛昆亮. 基于微服务的移动办公平台的设计与实现 [硕士学位论文]. 石家庄: 河北科技大学, 2020.
- 6 张宁溪, 朱晓民. 基于 Docker、Swarm、Consul 与 Nginx 构建高可用和可扩展 Web 服务框架的方法. 电信技术, 2016, (11): 21-25.
- 7 张佳亮, 普次仁. 基于 Java 的西藏大学图书资源共享系统的实现. 数字技术与应用, 2021, 39(2): 130-132.
- 8 季金亮, 冯建中, 贺日兴, 等. 混合云果园监测系统设计与开发. 中国果树, 2021, (9): 82-86.
- 9 杨旭. Redis 分布式存储在影片推荐中的应用. 现代电影技术, 2021, (9): 31-35. [doi: 10.3969/j.issn.1673-3215.2021.09.005]

(校对责编: 牛欣悦)