

自然写教学资源管理与内容分发系统软件 V1.0

软件著作权鉴别材料（设计说明书）

项目	内容
软件全称	自然写教学资源管理与内容分发系统软件
软件简称	自然写资源平台
版本号	V1.0
权利人	深圳自然写科技有限公司
开发语言	Java / JavaScript / Python
运行环境	Linux服务器 / CDN
文档类型	设计说明书
编制日期	2026年2月

目录

- 第一章 软件整体概述
 - 1.1 软件简介与功能综述
 - 1.2 软件用途与适用场景
 - 1.3 运行环境与系统要求
 - 1.4 开发语言与技术规范
 - 1.5 版本说明
- 第二章 系统架构与设计思路
 - 2.1 总体架构设计
 - 2.2 各层次详细说明
 - 2.3 点阵码资源管理架构
 - 2.4 数据设计
 - 2.5 接口设计
 - 2.6 安全设计
 - 2.7 部署架构
- 第三章 核心模块功能详细说明
 - 3.1 课件与字帖模板管理模块
 - 3.2 点阵码资源生成与管理模块
 - 3.3 内容审核与版本控制模块
 - 3.4 多终端资源分发与缓存模块

- 3.5 教师自定义内容上传模块
- 3.6 分类检索模块
- 3.7 CDN加速分发模块
- 3.8 资源使用统计模块
- 3.9 管理后台模块
- 第四章 操作流程与使用步骤
- 4.1 系统部署与初始化
- 4.2 管理员登录与资源管理操作
- 4.3 内容上传与审核流程
- 4.4 点阵码资源生成操作流程
- 4.5 教师检索与使用资源流程
- 4.6 资源统计与运营操作
- 4.7 异常处理与故障排除
- 第五章 与源代码的对应关系
- 5.1 模块与源代码文件对应表
- 5.2 核心类与方法说明
- 5.3 命名规范
- 附录

第一章 软件整体概述

1.1 软件简介与功能综述

自然写教学资源管理与内容分发系统软件（以下简称"资源平台"）是自然写互动课堂系统的内容管理与分发核心组件，专门用于管理和分发与自然写点阵笔配套使用的教学资源，包括字帖模板、课件资源、试卷模板、点阵码资源文件等，向全国各地学校提供高速、稳定的教学内容分发服务。

资源平台采用CMS内容管理系统与CDN内容分发网络相结合的技术架构，通过Spring Boot + Vue.js构建管理后台，通过Python实现点阵码生成引擎，通过Elasticsearch提供高性能全文检索，通过阿里云CDN/MinIO实现教学资源的全国加速分发。

主要功能模块：

(1) 课件与字帖资源管理：系统内置丰富的字帖模板库（覆盖人教版/苏教版/北师大版等主流教材版本，按年级/学科分类），管理员可批量导入、编辑、审核和发布教学资源。

(2) 点阵码资源生成与管理：点阵码是本系统的核心特色资源，系统内置点阵码生成引擎，将普通字帖或试卷纸张转换为含有点阵码信息的专用纸张文件，每个页面区域的点阵码全球唯一。

(3) 内容审核与版本控制：所有上传的资源需经管理员审核通过后方可对外分发，系统支持资源的多版本管理，可随时回滚至历史版本。

(4) 多终端分发：资源通过CDN加速向手机端、PC端、大屏端等各类终端高效分发，支持资源预下载缓存，保证离线场景下的使用体验。

(5) 教师自定义上传：教师可上传自制的课件、试卷、字帖等资源，经审核后在本校范围内共享使用。

(6) 分类检索：基于Elasticsearch提供按年级、学科、出版社、关键词等多维度的全文检索能力，帮助教师快速找到所需资源。

(7) 使用统计：实时统计各资源的下载次数、使用频率、学校分布等数据，支撑运营决策。

1.2 软件用途与适用场景

(1) 日常写字练习资源管理：学校使用自然写配套字帖时，需要在系统中为该批字帖注册点阵码范围，生成配套的点阵码文件用于印刷，使点阵笔在书写时能正确识别位置。

(2) 考试试卷资源制作：教师设计完试卷后，通过资源平台生成点阵码版试卷，印刷后发给学生使用，学生用点阵笔答题后数据自动上传至云平台批改。

(3) 教学课件共享：教师制作的优质教学PPT、教案等资源上传至平台后，可在学校内部或教研组范围内共享，避免重复制作。

(4) 教材版本资源同步：当教材版本更新时，平台运营团队及时更新对应版本的字帖模板和知识点映射，确保全国使用该版本教材的学校能同步获取最新资源。

1.3 运行环境与系统要求

组件	要求
操作系统	Linux (CentOS 7.6+ / Ubuntu 20.04+)
Java版本	OpenJDK 17+ (Spring Boot服务端)
Python版本	Python 3.9+ (点阵码生成引擎)
数据库	MySQL 8.0+ (元数据)、对象存储 (资源文件)
搜索引擎	Elasticsearch 8.x
CDN服务	阿里云CDN / 腾讯云CDN (生产环境)
对象存储	阿里云OSS / MinIO (私有化部署)
最低服务器配置	8核CPU、16GB内存 (CMS应用服务器)

1.4 开发语言与技术规范

模块	语言/框架	说明
后端CMS服务	Java 17 + Spring Boot 3.x	资源元数据管理、审核流程、API接口
管理控制台前端	Vue.js 3 + TypeScript + Element Plus	资源管理后台界面
点阵码生成引擎	Python 3.9 + PIL/Pillow + ReportLab	点阵码图案生成、PDF合成
搜索服务	Spring Data Elasticsearch	资源索引构建与全文检索
统计服务	Java + ClickHouse驱动	资源使用统计写入与查询

1.5 版本说明

版本号	发布日期	说明
V1.0	2026年2月	初始版本，包含资源管理、点阵码生成、CDN分发、检索统计全功能

第二章 系统架构与设计思路

2.1 总体架构设计

资源平台采用"内容管理系统（CMS）+ CDN内容分发"的经典架构，将内容的创建管理与内容的分发消费分离，分别针对不同需求进行优化：



2.2 各层次详细说明

内容管理层（CMS）：

CMS后端基于Spring Boot实现，提供资源全生命周期管理的业务逻辑： – 资源上传处理：接收多部分（multipart）文件上传请求，验证文件格式和大小，存储至OSS后创建资源元数据记录 – 审核 workflow：资源提交后进入审核队列，管理员查看资源详情后可批准或驳回，驳回需填写意见 – 版本管理：每次资源更新生成新版本记录，保留历史版本，支持版本回滚 – 分类管理：维护年级/学科/出版社三级分类目录树，资源归属于叶子分类节点

存储层：

存储层根据数据类型选择最合适的存储引擎：

对象存储（OSS/MinIO）：存储资源文件实体（PDF字帖、课件图片、试卷模板、点阵码打印文件等），利用对象存储的高可用性和低成本特性，99.99%的数据持久性。

MySQL：存储资源元数据（资源名称、类型、分类、版本、审核状态、创建者等）和业务配置数据，保证事务一致性。

Elasticsearch：为每个资源建立搜索索引，支持多字段联合检索、中文分词（IK分词器）、搜索结果高亮。

ClickHouse：存储资源访问日志（下载量、访问来源学校、时间分布等），用于高速聚合统计查询。

2.3 点阵码资源管理架构

点阵码资源是本平台的特色核心资源，其生成和管理涉及特殊的技术处理流程：

点阵码ID管理原则：

全球范围内的所有自然写点阵纸张，每个可书写区域（通常以毫米为单位的坐标格）都有唯一的点阵码ID。点阵码ID空间由自然写总部统一分配管理，确保不同批次、不同学校的纸张不会出现坐标冲突，保证点阵笔能正确识别书写位置。

点阵码ID分配策略：

点阵码ID结构（64位整数）：

产品线标识 (8 bits)	批次号 (16 bits)	页面序号 (40 bits)
-------------------	------------------	-------------------

分配方式：

- 平台为每批新印刷的纸张资源分配一段连续的点阵码ID范围
- 分配记录写入数据库（dot_pattern表），记录ID起止范围、绑定的资源类型和纸张规格
- 同一ID范围内的每一页纸张对应唯一的page_id

点阵码图案生成流程（Python引擎）：

输入：纸张规格（A4/B5/A5）、起始点阵码ID、页数



- Step 1：对每一页纸张，根据page_id计算该页的点阵码坐标分布
- 将纸张划分为以0.3mm为单位的坐标格
 - 每个坐标格对应唯一的微型点阵图案（肉眼不可见，直径约0.1mm）
- Step 2：将点阵码图案叠加到纸张内容模板上（字帖文字或空白背景）
- 点阵图案以浅灰色印刷，不影响可见内容
 - 精度要求：印刷分辨率需达到600 DPI以上
- Step 3：生成最终的PDF打印文件（含可见内容和不可见点阵码层）
- Step 4：PDF文件上传至OSS，更新dot_pattern数据库记录状态为"已生成"
- Step 5：返回PDF下载地址，供印刷厂下载使用

2.4 数据设计

资源元数据表（MySQL – resource）：

字段名	类型	约束	说明
id	BIGINT	PK, AUTO_INCREMENT	资源唯一ID
title	VARCHAR(128)	NOT NULL	资源标题
resource_type	VARCHAR(32)	NOT NULL	资源类型 (calligraphy/exam/courseware/dotcode)
subject	VARCHAR(32)		学科（语文/数学/英语等）
grade	TINYINT		年级（1-9，对应小学1年级到初三）
publisher	VARCHAR(64)		出版社版本（人教版/苏教版等）
category_id	INT	FK	所属分类节点ID
version	INT	DEFAULT 1	版本号（每次修改+1）
file_oss_key	VARCHAR(256)	NOT NULL	OSS中的文件路径键
file_size	BIGINT		文件大小（字节）
file_format	VARCHAR(16)		文件格式（PDF/PNG/PPTX等）
thumbnail_key	VARCHAR(256)		缩略图OSS路径
creator_id	BIGINT	FK	上传者用户ID
school_id	BIGINT	FK	归属学校ID（NULL表示全平台共享资源）
audit_status	TINYINT	DEFAULT 0	审核状态（0待审核/1已通过/2已驳回）
auditor_id	BIGINT	FK	审核人用户ID
audit_comment	VARCHAR(256)		审核意见
download_count	INT	DEFAULT 0	下载次数（异步更新）

字段名	类型	约束	说明
is_deleted	TINYINT	DEFAULT 0	软删除标志
created_at	DATETIME	NOT NULL	创建时间
updated_at	DATETIME	NOT NULL	最后更新时间

点阵码资源表（MySQL – dot_pattern）：

字段名	类型	说明
id	BIGINT	点阵码资源唯一ID
resource_id	BIGINT	关联资源ID（绑定的字帖或试卷资源）
pattern_id_start	BIGINT	分配的点阵码ID起始值
pattern_id_end	BIGINT	分配的点阵码ID结束值
paper_size	VARCHAR(8)	纸张规格（A4/B5/A5）
page_count	INT	总页数
dpi	INT	生成时的印刷分辨率（DPI）
print_file_key	VARCHAR(256)	生成的印刷PDF文件OSS路径
status	TINYINT	状态（0待生成/1已生成/2已发布）
batch_no	VARCHAR(32)	印刷批次号
created_at	DATETIME	创建时间

分类目录表（MySQL – category）：

字段名	类型	说明
id	INT	分类节点ID
parent_id	INT	父级分类ID（顶层分类parent_id=0）
name	VARCHAR(64)	分类名称
level	TINYINT	分类层级（1学科/2年级/3出版社版本）
sort_order	INT	排序权重
resource_count	INT	该分类下的资源数量（冗余，定时更新）

2.5 接口设计

资源管理API（Spring Boot后端）：

接口名称	HTTP方法	路径	权限	说明
资源检索	GET	/api/v1/resource/search	已登录	多条件检索资源（分类/关键词/类型）
获取资源详情	GET	/api/v1/resource/{id}	已登录	获取资源元数据和下载地址
资源下载	GET	/api/v1/resource/download/{id}	已登录	获取资源CDN签名下载URL
资源上传	POST	/api/v1/resource/upload	教师/管理员	上传资源文件（multipart/form-data）
资源审核	PUT	/api/v1/resource/audit/{id}	管理员	审核通过或驳回资源
资源版本历史	GET	/api/v1/resource/versions/{id}	已登录	获取资源历史版本列表
资源回滚	POST	/api/v1/resource/rollback/{id}	管理员	回滚至指定历史版本
点阵码生成	POST	/api/v1/dotcode/generate	管理员	触发点阵码PDF生成任务
点阵码状态查询	GET	/api/v1/dotcode/{id}/status	管理员	查询点阵码生成进度
分类目录	GET	/api/v1/category/tree	已登录	获取完整分类目录树
资源使用统计	GET	/api/v1/stat/resource/{id}	管理员	查询资源使用统计数据

Elasticsearch检索接口（内部调用）：

```
// 多条件检索示例请求（Elasticsearch Query DSL）
{
  "query": {
    "bool": {
      "must": [
        {"match": {"title": "三年级生字"}},
        {"term": {"subject": "语文"}},
        {"term": {"grade": 3}},
        {"term": {"publisher": "人教版"}}
      ],
      "filter": [
        {"term": {"audit_status": 1}},
        {"term": {"is_deleted": 0}}
      ]
    }
  },
  "highlight": {
    "fields": {"title": {}, "description": {}}
  },
  "from": 0,
  "size": 20
}
```


2.6 安全设计

内容安全机制：

(1) 上传内容过滤：对上传的文件进行格式白名单验证（仅允许PDF、PNG、JPEG、PPTX、DOCX等安全格式），通过文件Magic Number验证实际文件类型，防止文件类型伪造。PDF文件在存储前执行安全扫描，检测是否包含恶意JavaScript。

(2) 内容审核：所有用户上传的资源必须经管理员审核，防止不合规内容（政治敏感、版权侵权、低俗内容等）流入系统。管理员审核界面提供文件预览功能，可直接在浏览器中查看PDF/图片内容。

访问控制与防盗链：

(1) CDN防盗链：CDN资源链接通过URL签名机制（时间戳+密钥HMAC签名）保护，签名有效期1小时，防止链接泄露后被非法批量下载。

(2) Referer校验：CDN配置Referer白名单，仅允许来自自然写官方域名的请求访问资源，防止其他网站直接引用资源链接。

(3) 下载权限：资源按学校授权范围控制下载权限，私有资源（school_id非空）只有该学校的用户才能下载。

点阵码安全：

点阵码ID是物理纸张和数字系统之间的唯一桥梁，其安全性至关重要： – 点阵码ID范围分配有严格的申请审批流程，防止非法纸张冒充授权纸张 – 每批点阵码资源生成后，其ID范围记录在数据库中，可追溯到具体的印刷批次

2.7 部署架构



第三章 核心模块功能详细说明

3.1 课件与字帖模板管理模块

模块文件： `controller/ResourceController.java`、`service/ResourceManageService.java`

功能概述：

资源管理模块是资源平台的核心业务模块，负责所有教学资源的全生命周期管理，包括资源的上传入库、元数据编辑、分类归属、版本迭代和下架删除。

资源类型体系：

资源类型代码	类型名称	说明
calligraphy	字帖模板	标准练字用字帖（含点阵码版和普通版）
exam_paper	试卷模板	标准化考试试卷（含点阵码版）
courseware	课件资源	教学PPT、教案、教学视频等
worksheet	练习册	课后练习题目（PDF格式）
dotcode_file	点阵码印刷文件	供印刷厂使用的含点阵码的PDF印刷文件
reference	参考资料	教研资料、教师参考手册等

版本控制机制：

- 资源版本管理流程：
1. 初始上传：version=1，状态为"待审核"
 2. 审核通过：状态更新为"已发布"，Elasticsearch建立索引
 3. 更新资源：创建新版本记录（version=2，复制元数据），上传新文件
 4. 新版本审核通过：最新版本设为"当前版本"，旧版本标记为"历史版本"
 5. 版本回滚：将历史版本指定为当前版本，旧当前版本变为历史版本
 6. 资源下架：所有版本状态改为"已下架"，从Elasticsearch索引中移除

资源文件存储策略：

上传到OSS的文件按如下目录结构组织：

```
writtech-resources/
├── calligraphy/           # 字帖资源
│   ├── {year}/           # 按年份分目录
│   │   ├── {resource_id}/
│   │   │   ├── v1/
│   │   │   │   ├── original.pdf    # 原始文件
│   │   │   │   └── thumbnail.png   # 缩略图（首页截图）
│   │   │   └── v2/
│   └── dotcode/           # 点阵码印刷文件
│       ├── {batch_no}/     # 按印刷批次分目录
│       └── {pattern_id_range}/
```

3.2 点阵码资源生成与管理模块

模块文件：`service/DotPatternService.java`（Java控制层）、`dotcode_generator.py`（Python生成引擎）

功能概述：

点阵码资源生成模块是本平台最具技术特色的核心模块，实现了将普通纸张内容（字帖、试卷）与唯一坐标信息（点阵码）融合的关键技术，使纸张成为自然写互动课堂系统的输入终端。

点阵码生成算法概述：

点阵码使用四维（Anoto）坐标编码技术，在A4纸张（210mm×297mm）上以0.3mm为间距排列约700×1000个坐标点，每个坐标点位置对应唯一的page_id和坐标值。

坐标点以微型圆点（直径0.08mm）印刷，从4个方向上相对格点中心有细微偏移（上/右/下/左各偏移一定距离），4种偏移方向组成2比特信息（00/01/10/11），相邻坐标点的信息序列共同编码page_id和坐标值。

```
# dotcode_generator.py 核心算法伪代码

def generate_dot_pattern(page_id: int, paper_size: str, dpi: int) -> Image:
    """
    为指定page_id和纸张规格生成点阵码图像（叠加层）

    Args:
        page_id: 本页的全局唯一点阵码页面ID
        paper_size: 纸张规格 ("A4"/"B5"/"A5")
        dpi: 生成分辨率（需600DPI以上）

    Returns:
        PIL Image: 含点阵码叠加层的透明背景图像
    """
    width_px, height_px = get_paper_pixels(paper_size, dpi)
    dot_image = Image.new("RGBA", (width_px, height_px), (255,255,255,0))
    draw = ImageDraw.Draw(dot_image)

    # 计算点阵间距（像素）
    dot_spacing_px = int(0.3 * dpi / 25.4) # 0.3mm转像素
    dot_radius_px = max(1, int(0.04 * dpi / 25.4)) # 点半径

    # 遍历所有坐标格
    for row in range(height_px // dot_spacing_px):
        for col in range(width_px // dot_spacing_px):
            # 根据page_id和格点位置计算编码值
            code_bits = encode_coordinate(page_id, col, row)
            # 根据编码值确定偏移方向
            offset = get_offset_direction(code_bits)
            # 在偏移位置绘制微型圆点（浅灰色，不影响可见内容）
            center_x = col * dot_spacing_px + offset.dx
            center_y = row * dot_spacing_px + offset.dy
            draw.ellipse([
                (center_x - dot_radius_px, center_y - dot_radius_px),
                (center_x + dot_radius_px, center_y + dot_radius_px)
```

```

], fill=(180, 180, 180, 255))

return dot_image

def merge_content_with_dotcode(content_pdf_path: str,
                               dot_images: List[Image],
                               output_path: str):
    """
    将点阵码图层叠加到内容PDF上，生成印刷用PDF
    """
    # 使用PyMuPDF读取内容PDF
    content_doc = fitz.open(content_pdf_path)
    output_doc = fitz.open()

    for page_idx, page in enumerate(content_doc):
        # 将该页的点阵码图像叠加
        dot_layer = dot_images[page_idx]
        dot_layer_bytes = img_to_bytes(dot_layer)

        new_page = output_doc.new_page(width=page.rect.width,
                                       height=page.rect.height)

        # 先插入原始内容，再叠加点阵码图层
        new_page.show_pdf_page(new_page.rect, content_doc, page_idx)
        new_page.insert_image(new_page.rect, stream=dot_layer_bytes)

    output_doc.save(output_path, garbage=4, deflate=True)

```

点阵码ID范围申请流程:

- 步骤1: 管理员在资源平台后台提交点阵码ID范围申请
 - 填写: 纸张用途 (字帖/试卷)、纸张规格、预计印刷页数、印刷批次号
- 步骤2: 系统从dot_pattern_id_pool表中分配连续的ID范围 (原子操作, 防并发冲突)
 - 使用MySQL FOR UPDATE行锁保证ID范围唯一分配
 - 每次分配的ID范围至少有20%的预留冗余 (防止页数超预估)
- 步骤3: 分配结果写入dot_pattern表, 状态为"待生成"
- 步骤4: 触发点阵码生成Worker (Python进程) 开始处理
- 步骤5: Worker生成完成后, 将印刷PDF上传至OSS, 更新dot_pattern状态为"已生成"
- 步骤6: 管理员审核并向印刷厂提供下载链接

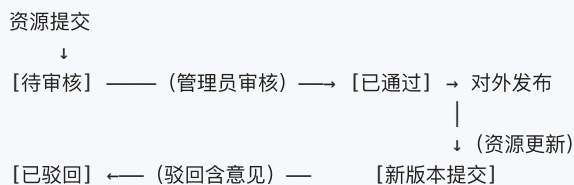
3.3 内容审核与版本控制模块

模块文件: service/ResourceAuditService.java

功能概述：

内容审核模块实现了完整的UGC（用户生成内容）审核 workflow，确保所有进入平台的教学资源的质量和合规性，防止不当内容影响教学环境。

审核工作流状态机:



↓
上传者修改后重新提交

审核要点检查清单（管理员审核时参考）：

检查项目	说明
内容完整性	文件是否可以正常打开，内容是否完整无损坏
版权合规	内容是否涉及未授权使用的版权材料
分类准确性	年级/学科/出版社版本标注是否与实际内容一致
内容适宜性	是否适合K12学生使用，无不良内容
格式规范	文件格式是否符合平台要求（字帖需PDF格式，分辨率≥300DPI）
字帖专项	字帖内容是否来自教材，是否与标注的教材版本一致

3.4 多终端资源分发与缓存模块

模块文件：`service/ResourceDeliveryService.java`、`service/CdnService.java`

功能概述：

分发模块负责为各终端应用提供高速的资源获取服务，通过CDN加速确保全国各地的教师和学生都能以低延迟获取教学资源。

资源下载链接生成机制：

当终端请求下载资源时，服务端不直接返回文件，而是生成带有时效性签名的CDN直链：

```
// CdnService.java 核心逻辑（伪代码）
public String generateSignedDownloadUrl(String ossKey, long expirySeconds) {
    // 生成CDN签名URL
    long expireTime = System.currentTimeMillis() / 1000 + expirySeconds;
    String rawPath = "/" + ossKey;
    String signStr = CDN_PRIVATE_KEY + rawPath + expireTime;
    String md5Hash = DigestUtils.md5Hex(signStr);

    return String.format(
        "https://cdn.writech.com%s?sign=%s&t=%d",
        rawPath, md5Hash, expireTime
    );
}
```

终端缓存策略：

为减少网络请求，提升离线使用体验，各终端APP对资源文件实行本地缓存：

终端类型	缓存容量	缓存策略
PC端	10GB	最近30天使用过的资源全部缓存

终端类型	缓存容量	缓存策略
智慧黑板端	20GB	本学期所有课件资源预热下载
Pad端	3GB	LRU策略，最近14天使用资源
手机端	1GB	LRU策略，最近7天使用资源

3.5 教师自定义内容上传模块

模块文件：`controller/ResourceController.java`（上传接口）、`service/ResourceManageService.java`（处理逻辑）

功能概述：

教师上传模块允许教师将自己制作的教学资源上传至平台，经管理员审核后在本校教师范围内共享，形成校本资源库。

上传限制与规范：

参数	限制
单文件最大大小	100MB
支持格式	PDF, PPTX, DOCX, PNG, JPG, MP4（视频限50MB）
上传频率	每个教师每日最多上传20个文件
存储配额	每所学校共享100GB存储配额（可扩容）

断点续传支持：

对于大文件（>10MB），上传模块使用分片上传（Multipart Upload）机制： – 客户端将大文件分割为5MB的分片 – 每个分片独立上传，支持失败重试 – 所有分片上传完成后，服务端合并分片为完整文件 – 若上传中断，客户端重新上传时可查询已上传的分片列表，只需上传缺失分片

3.6 分类检索模块

模块文件：`service/ResourceSearchService.java`、`config/ElasticsearchConfig.java`

功能概述：

检索模块基于Elasticsearch实现多维度的教学资源检索，支持关键词全文检索、分类精确筛选、排序（相关性/热度/最新发布）等检索能力。

Elasticsearch索引设计：

```
// 资源索引Mapping定义
{
  "mappings": {
```

```

    "properties": {
      "id": {"type": "keyword"},
      "title": {"type": "text", "analyzer": "ik_max_word"},
      "description": {"type": "text", "analyzer": "ik_max_word"},
      "subject": {"type": "keyword"},
      "grade": {"type": "integer"},
      "publisher": {"type": "keyword"},
      "resource_type": {"type": "keyword"},
      "tags": {"type": "keyword"},
      "school_id": {"type": "keyword"},
      "download_count": {"type": "integer"},
      "audit_status": {"type": "integer"},
      "created_at": {"type": "date"}
    }
  }
}

```

检索结果排序策略：

```

综合排序分 =
  相关性得分 (BM25算法) × 0.5 +
  热度分 (log(download_count+1) / log(max_downloads+1)) × 0.3 +
  新鲜度分 (1 / (1 + days_since_publish/365)) × 0.2

```

3.7 CDN加速分发模块

模块文件： `service/CdnService.java` 、 `config/CdnConfig.java`

功能概述：

CDN分发模块负责将存储在OSS中的资源文件通过CDN网络加速，确保全国各地的用户都能以最快速度获取教学资源。

CDN预热策略：

当热门资源（如全国通用字帖模板）发布时，主动触发CDN预热，将文件从OSS同步至各CDN边缘节点，避免首批用户访问时触发CDN回源造成的延迟：

```

触发条件：
- 全平台共享资源（school_id=NULL）审核通过后自动触发预热
- 管理员手动触发（用于临时推送新资源）

预热范围：
- 华东节点（覆盖上海、江苏、浙江、安徽）
- 华南节点（覆盖广东、福建、广西）
- 华北节点（覆盖北京、天津、河北）
- 西部节点（覆盖四川、重庆、陕西等）

```

3.8 资源使用统计模块

模块文件： `controller/StatController.java` 、 `service/StatService.java`

功能概述：

统计模块收集和分析资源的使用数据，为平台运营提供数据支撑，帮助内容团队了解哪些资源最受欢迎，优先更新和扩充热门资源。

统计数据采集方式：

资源下载时，服务端异步将使用记录写入Kafka，由统计消费者批量写入ClickHouse，避免统计写入影响资源下载的响应时间：



统计报表维度：

报表类型	维度	指标
资源热度排行	按下载量排名	下载次数/周环比增长率
学校使用分布	按学校	各校下载次数/使用资源数
分类使用分析	按学科/年级/出版社	各分类资源下载量占比
时间趋势分析	按日/周/月	下载量时间趋势（折线图）

3.9 管理后台模块

模块文件： Vue.js前端项目（src/views/resource/目录下的页面文件）

功能概述：

管理后台是面向资源平台管理员和运营人员的Web应用，提供资源管理的全部操作功能，基于Vue.js 3实现，通过REST API与后端服务交互。

主要页面模块：

- (1) 资源列表页（ResourceList.vue）

界面布局：

教学资源管理							[+ 新增资源]	[批量导入]	[导出清单]
筛选条件：[资源类型▼] [学科▼] [年级▼] [出版社▼] [状态▼] [搜索]									
#	资源标题	类型	年级	学科	状态	操作			

1	人教版三年级上册第一单元	字帖	三年	语文	已发布	查看/编辑	版本/删除
2	期中考试试卷	试卷	四年	数学	待审核	查看/审核	

< 上一页

1 2 3 ... 50

下一页 >

共计1000条资源

(2) 资源审核页 (ResourceAudit.vue)

待审核资源列表，支持批量审核操作，审核时可在线预览PDF资源内容。

(3) 点阵码管理页 (DotPatternManager.vue)

查看点阵码ID分配历史，新增ID范围申请，查看生成进度，下载印刷文件。

(4) 统计分析页 (ResourceStatistics.vue)

展示资源使用统计的可视化图表，包括热度排行、使用趋势、学校分布地图等。

第四章 操作流程与使用步骤

4.1 系统部署与初始化

后端服务部署：

- 步骤1: 准备运行环境: JDK 17、MySQL 8.0、Elasticsearch 8.x、Redis
- 步骤2: 配置application.yml数据库连接、OSS访问密钥、CDN配置
- 步骤3: 执行数据库初始化脚本: mysql < schema/init.sql
- 步骤4: 启动Spring Boot应用: java -jar writtech-resource-platform.jar
- 步骤5: 初始化Elasticsearch索引: POST /api/v1/admin/init-es-index
- 步骤6: 导入初始分类数据: POST /api/v1/admin/import-categories
- 步骤7: 导入初始字帖资源: python scripts/import_initial_resources.py

4.2 管理员登录与资源管理操作

管理员控制台登录：

访问: https://admin.resource.writtech.com

账号: 管理员账号 (superadmin)

密码: 初始密码 (首次登录强制修改)

登录成功后进入Dashboard概览页：

自然写资源平台管理控制台	
左侧导航菜单	概览统计卡片：
资源管理	总资源数: 12,500
检索配置	今日新增: 25
点阵码	待审核: 3
	本月下载: 45,000次



使用统计



系统配置

热门资源排行 (Top10)

4.3 内容上传与审核流程

批量资源上传操作（管理员）：

- 步骤1：进入 资源管理 → 新增资源
- 步骤2：选择上传方式（单个上传 / 批量上传ZIP压缩包）
- 步骤3：填写资源元数据：
- 资源标题（必填）
 - 资源类型（字帖/试卷/课件等）
 - 学科（语文/数学/英语等）
 - 年级（一年级-九年级）
 - 出版社版本（人教版/苏教版等）
 - 关键词标签（多个，用于检索）
 - 简介描述（选填）
- 步骤4：上传资源文件（支持PDF、PPTX等格式）
- 步骤5：等待文件上传完成（显示进度条）
- 步骤6：系统自动生成缩略图（PDF首页截图）
- 步骤7：提交审核
- 步骤8：（若为管理员自己上传）可直接审核通过，免去等待

4.4 点阵码资源生成操作流程

为字帖生成点阵码版本：

- 步骤1：进入 点阵码管理 → 新增生成任务
- 步骤2：选择关联资源（选择已上传的字帖内容资源）
- 步骤3：填写生成参数：
- 印刷批次号：[BATCH20260201]（自定义，用于追溯）
- 纸张规格：[A4 ▼]
- 印刷数量：[3000] 册
- 计划印刷起始页码：自动计算（基于资源内容页数×册数）
- 步骤4：系统显示将分配的点阵码ID范围（预览）：
- 点阵码ID范围：10000001 ~ 10030000（共30000页）
- 步骤5：确认提交，系统后台开始生成任务（预计时间：约N分钟，取决于页数）
- 步骤6：生成进度实时展示（0% → ... → 100%）
- 步骤7：生成完成后，下载印刷PDF文件（提供给印刷厂）

点阵码管理列表界面：

点阵码管理					[+ 新增生成任务]
批次号	关联资源	ID范围	进度	操作	
B26001	三年级上册字帖	1M-1.03M	完成	下载/查看	
B26002	四年级数学试卷	1.03M-1.1M	生成中75%	等待...	

4.5 教师检索与使用资源流程

教师在APP中检索和下载资源：

操作路径：自然写PC端APP → 资源中心 → 字帖/课件库

检索界面：

资源中心

搜索：[三年级语文生字] [🔍 搜索]

筛选：[年级： 三年级▼] [学科： 语文▼] [出版社： 人教版▼]

搜索结果（共23个资源）

[缩略图]	[缩略图]	[缩略图]
三年级上生字	三年级下生字	三年级语文
字帖（人教版）	字帖（人教版）	期末试卷
下载：1,250次	下载：980次	下载：650次
[下载]	[下载]	[下载]

- 操作步骤：
- 1. 在搜索框输入关键词（如"三年级语文"）
 - 2. 使用年级/学科/出版社筛选器进一步精确范围
 - 3. 在结果列表中预览缩略图，确认是所需资源
 - 4. 点击"下载"按钮，APP后台开始下载资源文件到本地
 - 5. 下载完成后，资源可在备课界面中直接使用

4.6 资源统计与运营操作

操作路径：管理控制台 → 使用统计 → 资源分析

统计仪表盘界面：

资源使用统计 时间范围：[本月▼]

本月下载量：45,820次 活跃学校：236所 活跃用户：1,890人

下载量趋势（折线图）
[图表区域]

分类占比（饼图）
字帖 45% / 试卷 30% / 课件 25%

热门资源TOP10

- 1. 三年级上册字帖（人教版） - 1,250次下载
- 2. 三年级数学期末试卷 - 980次下载
- 3. ...

4.7 异常处理与故障排除

异常现象	可能原因	处理方法
资源上传后一直显示"处理中"	缩略图生成服务异常	检查缩略图生成Worker日志，手动触发缩略图重新生成

异常现象	可能原因	处理方法
资源下载链接返回403	CDN签名过期或生成错误	检查CDN密钥配置，重新生成签名URL
点阵码生成任务卡住	Python生成进程崩溃	检查Worker进程状态，重启生成Worker
Elasticsearch检索无结果	资源未建立索引	执行索引重建脚本，重新索引所有审核通过的资源
CDN资源不更新（旧缓存）	CDN缓存未刷新	在CDN控制台手动提交URL刷新，或通过平台API触发刷新

第五章 与源代码的对应关系

5.1 模块名称与源代码文件对应表

功能模块	源文件路径	主要类	说明
应用程序主入口	WritechResourceApplication.java	WritechResourceApplication	Spring Boot应用主类
资源管理接口	controller/ResourceController.java	ResourceController	资源CRUD、上传、下载接口
统计接口	controller/StatController.java	StatController	资源使用统计查询接口
资源元数据管理	service/ResourceManageService.java	ResourceManageService	资源CRUD业务逻辑
资源审核服务	service/ResourceAuditService.java	ResourceAuditService	审核 workflow 管理
点阵码生成服务	service/DotPatternService.java	DotPatternService	点阵码任务调度 (Java侧)
CDN服务	service/CdnService.java	CdnService	CDN签名URL生成、预热
搜索服务	service/ResourceSearchService.java	ResourceSearchService	Elasticsearch检索接口封装
统计服务	service/StatService.java	StatService	统计数据写入和查询

功能模块	源文件路径	主要类	说明
数据实体模型	model/Resource.java 等	Resource, DotPattern, Category	JPA实体类定义

5.2 核心类与方法说明

WritechResourceApplication.java:

```
@SpringBootApplication
@EnableScheduling
public class WritechResourceApplication {
    public static void main(String[] args) {
        SpringApplication.run(WritechResourceApplication.class, args);
    }
}
```

ResourceController.java 核心方法:

方法名	HTTP方法	路径	功能
searchResources(SearchReq req)	GET	/api/v1/resource/search	多条件检索资源, 委托 ResourceSearchService
getDownloadUrl(Long resourceId)	GET	/api/v1/resource/download/{id}	生成CDN签名URL, 记录下载事件
uploadResource(MultipartFile file, ResourceMetaReq req)	POST	/api/v1/resource/upload	文件上传, 创建资源记录
auditResource(Long id, AuditReq req)	PUT	/api/v1/resource/audit/{id}	审核通过或驳回, 触发索引更新

DotPatternService.java 核心方法:

方法名	功能说明
allocateDotPatternRange(int pageCount)	原子操作分配点阵码ID范围 (数据库行锁)
triggerGenerationTask(DotPattern record)	向Python Worker发送生成任务 (通过Redis队列)
updateGenerationProgress(Long id, int progress)	更新生成进度 (被Python Worker回调)
getDownloadUrl(Long patternId)	生成印刷PDF的CDN签名下载URL

5.3 命名规范

Java包命名规范:

```
com.writech.resource.{layer}
com.writech.resource.controller // 控制器层
com.writech.resource.service    // 服务层
com.writech.resource.model      // 数据实体
com.writech.resource.config     // 配置类
```

数据库命名规范：

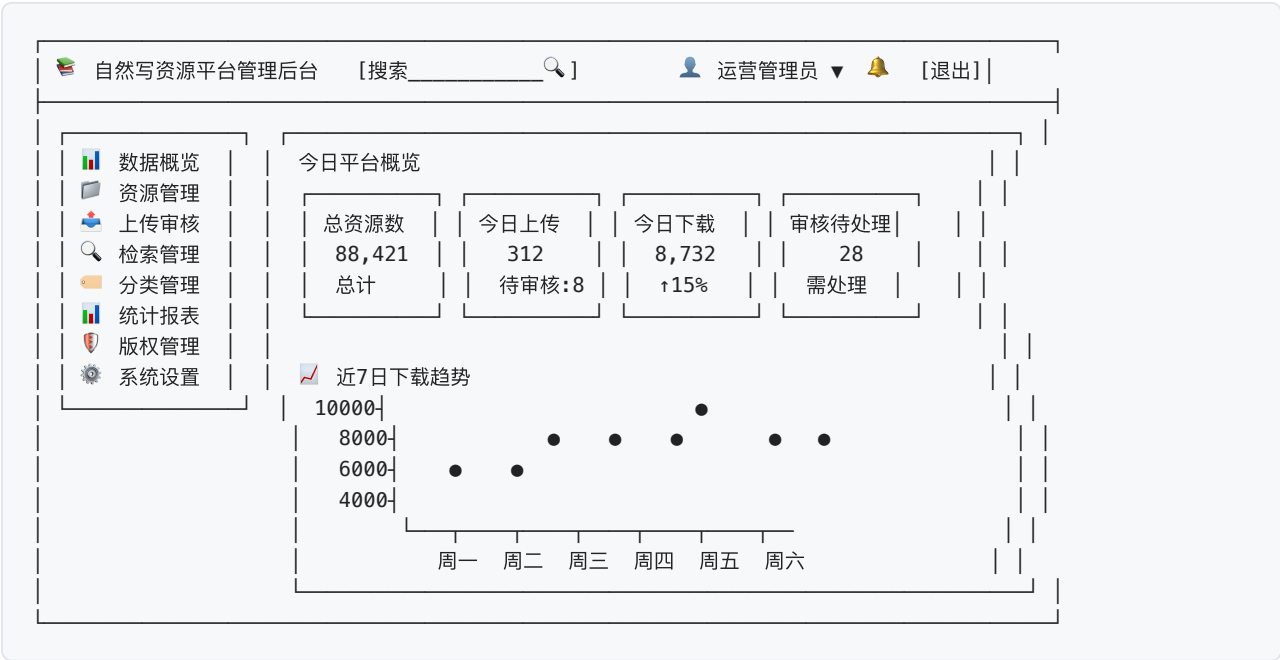
- 表名：小写下划线，业务名为前缀，如 resource、dot_pattern、category、audit_record
- 字段：小写下划线，如 resource_type、audit_status、creator_id
- 索引：idx_{表名}_{字段名} 格式，如 idx_resource_grade_subject

附录

附录A 界面设计稿（GUI Mockup）

本附录提供自然写教学资源管理与内容分发系统软件各主要管理后台界面的设计稿，以线框图形式呈现界面布局与交互元素。

A.1 系统主控台（Dashboard）



A.2 资源列表管理页面



[搜索资源名称🔍] 分类▼ 格式▼ 状态▼ 时间范围[]至[] [🔍筛选]						
#	资源名称	格式	分类	状态	下载量	操作
1	小学语文一年级上册全套	PDF	语文·教案	✅ 已发布	3,421	[详情][编辑][下线]
2	数学二年级乘法口诀练习册	PDF	数学·练习	✅ 已发布	2,188	[详情][编辑][下线]
3	英语字母书写范本视频	MP4	英语·视频	⌚ 审核中	0	[详情][撤回]
4	语文三年级古诗精选点阵纸	PDF	语文·点阵	✅ 已发布	5,632	[详情][编辑][下线]
5	数学思维训练题卡（中级）	PDF	数学·题卡	❌ 审核拒绝	0	[详情][重新提交]
共 88,421 条资源 < 1 2 3 ... 2947 > 每页显示 [30▼]						

A.3 资源上传与审核页面

📁 资源上传审核 / 待审核列表 [批量审核]					
待审核：28 今日已审核：156 审核通过率：94.2%					
审核队列（AI预审已完成，等待人工复核）					
#	资源名称	上传者	AI初判	提交时间	操作
1	作文批改参考答案	王老师	⚠️ 可疑	09:30	[预览][通过][拒绝]
2	数学竞赛题集	李老师	✅ 通过	09:15	[预览][通过][拒绝]
3	英语听力音频合集	张老师	✅ 通过	08:55	[预览][通过][拒绝]
4	语文阅读理解练习	陈老师	⚠️ 可疑	08:42	[预览][通过][拒绝]

A.4 资源检索页面（用户端）

🔍 教学资源搜索	
[🔍 搜索关键词，如"小学数学乘法练习"]	
分类：[全部] [语文] [数学] [英语] [科学] [点阵纸]	格式：[全部] [PDF] [视频] [音频]
年级：[全部] [一年级] [二年级] [三年级] ... [高三]	排序：[相关度▼] [下载量] [最新]
搜索结果：找到 1,243 个相关资源 （关键词："小学数学乘法练习"）	
<div><div> <小学数学>二年级<乘法>口诀<练习>册（点阵版）</div><div>📄 下载 2,188</div><div>分类：数学·练习 年级：二年级 格式：PDF 大小：12.3MB</div><div>★★★★★ 4.8分 [立即下载] [预览] [收藏]</div></div>	
<div><div> <小学数学><乘法>口诀记忆视频（动画）</div><div>📄 下载 892</div><div>分类：数学·视频 年级：二年级 格式：MP4 大小：45.2MB</div><div>★★★★☆ 4.2分 [立即下载] [预览] [收藏]</div></div>	

A.5 版权管理页面

版权管理 [+ 登记版权]					
[搜索资源名称/证书号 🔍] 状态▼ 授权类型▼ [🔍 筛选]					
#	资源名称	版权证书编号	授权类型	注册日期	操作
1	小学语文一年级上册	WRC-A3F8B21C	学校授权	2026-01-01	[详情][验证][下载]
2	数学乘法口诀练习册	WRC-C7D2E45A	公共授权	2026-01-15	[详情][验证][下载]
3	英语字母书写范本	WRC-F1A9B83D	独家授权	2026-02-01	[详情][验证][下载]
共 12,438 条版权记录 < 1 2 3 ... >					

附录B 术语表

术语	说明
点阵码	印刷在纸张上的极细微点阵图案，肉眼几乎不可见，点阵笔摄像头可识别并解算当前坐标
CDN	内容分发网络（Content Delivery Network），通过在全国多地部署节点，就近为用户提供内容访问
OSS	对象存储服务（Object Storage Service），用于存储和分发大容量非结构化文件数据
防盗链	通过URL签名、Referer校验等机制防止资源被非授权第三方网站直接引用
Elasticsearch	分布式全文搜索引擎，支持复杂的多条件检索和聚合统计
CDN预热	主动将资源从源站推送至CDN边缘节点，避免用户首次访问时的回源延迟
IK分词器	Elasticsearch的中文分词插件，支持中文词语的智能分词
签名URL	包含时间戳和HMAC签名的访问链接，超过有效期或签名不匹配则拒绝访问

附录B 版本历史

版本号	发布日期	变更说明
V1.0	2026年2月	初始版本，包含资源管理、点阵码生成、CDN分发、检索统计全功能体系

编制日期：2026年2月

版权声明：本文档版权归深圳自然写科技有限公司所有，未经授权不得复制或传播

附录C 资源管理平台核心功能详细实现

C.1 内容分发网络（CDN）加速策略

C.1.1 资源分类与 CDN 缓存策略

```
// CdnCacheConfig.java
@Configuration
public class CdnCacheConfig {

    /**
     * 根据资源类型配置 CDN 缓存 TTL
     */
    public static long getCacheTtlSeconds(ResourceType type) {
        return switch (type) {
            // 字帖参考图片：内容稳定，缓存30天
            case CALLIGRAPHY_IMAGE -> 30L * 24 * 3600;
            // 字帖笔顺数据（JSON）：内容稳定，缓存7天
            case CALLIGRAPHY_STROKE_DATA -> 7L * 24 * 3600;
            // 教学视频：内容稳定，缓存7天
            case TEACHING_VIDEO -> 7L * 24 * 3600;
            // 课件 PPT/PDF：可能更新，缓存1天
            case COURSEWARE -> 1L * 24 * 3600;
            // 学情报告 PDF：个性化内容，不缓存
            case STUDENT_REPORT -> 0L;
            // 作业内容：动态内容，不缓存
            case ASSIGNMENT_CONTENT -> 0L;
            // 系统配置：缓存5分钟
            default -> 300L;
        };
    }

    /**
     * 生成 CDN 防盗链签名 URL
     * 防止资源被其他网站直接引用
     */
    public String generateSignedUrl(String objectKey, String userId,
                                     Duration validity) {
        long expireAt = Instant.now().plus(validity).getEpochSecond();
        // 签名字符串: {objectKey}\n{expireAt}\n{userId}
        String stringToSign = objectKey + "\n" + expireAt + "\n" + userId;
        String signature = hmacSha256Base64(stringToSign, cdnSecretKey);

        return String.format("%s/%s?expire=%d&uid=%s&sign=%s",
                               cdnBaseUrl, objectKey, expireAt, userId, signature);
    }
}
```

C.1.2 多级存储架构

资源平台采用冷热分层存储策略，自动迁移不同访问频率的资源：

存储分层策略：

热存储层（SSD，单位成本高）
存储：近30天内被访问的资源
CDN 缓存命中率：> 95%
访问延迟：< 50ms

热→温迁移（30天未访问）

温存储层（HDD，标准对象存储）
存储：31~180天内有访问的资源
CDN 按需加载（无缓存或低 TTL）
访问延迟：< 500ms

温→冷迁移（180天未访问）

冷存储层（归档存储，低成本）
存储：180天以上未访问的历史资源
取回延迟：分钟级（用于长期存档）
适用：旧学期课件、历史录像

C.2 资源版权管理系统

C.2.1 数字水印实现

```
// WatermarkService.java
@Service
public class WatermarkService {

    /**
     * 为图片资源添加不可见数字水印
     * 水印信息包含：学校ID、下载时间、用户ID（哈希后）
     */
    public byte[] addInvisibleWatermark(byte[] imageData,
                                         WatermarkInfo info) {
        BufferedImage image = ImageIO.read(new ByteArrayInputStream(imageData));

        // 将水印信息编码为二进制序列
        String watermarkText = String.format("%s|%d|%s",
            info.getSchoolId(),
            System.currentTimeMillis() / 1000,
            DigestUtils.sha256Hex(info.getUserId()).substring(0, 8)
        );
        byte[] watermarkBits = watermarkText.getBytes(StandardCharsets.UTF_8);

        // DCT 域水印嵌入（修改 DCT 系数的最低有效位）
        // 采用 LSB（最低有效位）替换技术，对图片视觉效果影响极小
        int bitIndex = 0;
        for (int y = 0; y < image.getHeight() && bitIndex < watermarkBits.length * 8; y++) {
            for (int x = 0; x < image.getWidth() && bitIndex < watermarkBits.length * 8; x++) {
                int pixel = image.getRGB(x, y);
                int r = (pixel >> 16) & 0xFF;
                int bit = (watermarkBits[bitIndex / 8] >> (7 - bitIndex % 8)) & 1;
                // 修改红色通道的最低有效位
                r = (r & 0xFE) | bit;
                image.setRGB(x, y, (pixel & 0xFF00FFFF) | (r << 16));
                bitIndex++;
            }
        }
    }
}
```

```

    }
}

ByteArrayOutputStream out = new ByteArrayOutputStream();
ImageIO.write(image, "PNG", out);
return out.toByteArray();
}

/**
 * 提取数字水印（用于版权溯源）
 */
public String extractWatermark(byte[] imageData, int watermarkLength) {
    BufferedImage image = ImageIO.read(new ByteArrayInputStream(imageData));
    byte[] watermarkBits = new byte[watermarkLength];

    int bitIndex = 0;
    for (int y = 0; y < image.getHeight() && bitIndex < watermarkLength * 8; y++) {
        for (int x = 0; x < image.getWidth() && bitIndex < watermarkLength * 8; x++) {
            int r = (image.getRGB(x, y) >> 16) & 0xFF;
            int bit = r & 1;
            watermarkBits[bitIndex / 8] |= (bit << (7 - bitIndex % 8));
            bitIndex++;
        }
    }

    return new String(watermarkBits, StandardCharsets.UTF_8);
}
}

```

C.3 资源审核与内容安全

C.3.1 自动内容审核流程

```

// ContentAuditService.java
@Service
public class ContentAuditService {

    @Autowired
    private AliYunGreenService greenService; // 阿里云内容安全

    @Autowired
    private ResourceRepository resourceRepo;

    /**
     * 资源上传后自动触发内容审核
     */
    @Async("auditThreadPool")
    public void auditResource(String resourceId) {
        Resource resource = resourceRepo.findById(resourceId)
            .orElseThrow(() -> new ResourceNotFoundException(resourceId));

        try {
            AuditResult result = switch (resource.getType()) {
                case IMAGE, CALLIGRAPHY_IMAGE -> auditImage(resource.getUrl());
                case VIDEO, TEACHING_VIDEO -> auditVideo(resource.getUrl());
                case TEXT, DOCUMENT -> auditText(resource.getTextContent());
                default -> AuditResult.passed(); // 其他类型暂不审核
            };
        }
    }
}

```

```

        if (result.isPassed()) {
            resource.setStatus(ResourceStatus.PUBLISHED);
            resource.setAuditPassedAt(LocalDate.now());
        } else {
            resource.setStatus(ResourceStatus.AUDIT_FAILED);
            resource.setAuditFailReason(result.getFailReason());

            // 通知上传者审核未通过
            notificationService.sendAuditFailNotification(
                resource.getUploaderId(),
                resourceId,
                result.getFailReason()
            );
        }

        resourceRepo.save(resource);

    } catch (AuditServiceException e) {
        log.error("内容审核服务异常, resourceId={}", resourceId, e);
        // 降级: 标记为待人工审核
        resource.setStatus(ResourceStatus.PENDING_MANUAL_AUDIT);
        resourceRepo.save(resource);
    }
}

private AuditResult auditImage(String imageUrl) {
    // 调用阿里云图片内容安全 (检测违规内容、版权侵权等)
    ImageAuditResponse response = greenService.imageAudit(imageUrl,
        List.of("porn", "terrorism", "ad", "qrcode"));

    for (ImageScanResult scan : response.getResults()) {
        if ("block".equals(scan.getSuggestion())) {
            return AuditResult.failed(scan.getLabel() + ":" + scan.getRate());
        }
    }

    return AuditResult.passed();
}
}

```

C.4 资源推荐算法

C.4.1 协同过滤推荐

```

// ResourceRecommendService.java
@Service
public class ResourceRecommendService {

    /**
     * 基于协同过滤的资源推荐
     * "与你教学风格相似的老师也在用这些资源"
     */
    public List<Resource> recommendForTeacher(String teacherId,
                                                String subject,
                                                int count) {

        // 步骤1: 获取目标教师的资源使用历史 (最近90天)
        List<ResourceUsage> targetUsage = usageRepo
            .findByTeacherIdAndPeriod(teacherId,

```

```

        LocalDate.now().minusDays(90), LocalDate.now());

Set<String> targetResourceIds = targetUsage.stream()
    .map(ResourceUsage::getResourceId)
    .collect(Collectors.toSet());

// 步骤2: 找出有相似使用记录的教师 (Jaccard 相似度)
List<String> similarTeachers = findSimilarTeachers(
    teacherId, targetResourceIds, subject, 20
);

// 步骤3: 获取相似教师使用但目标教师未使用的资源
Map<String, Integer> candidateScores = new HashMap<>();
for (String similarTeacherId : similarTeachers) {
    List<ResourceUsage> usage = usageRepo
        .findByTeacherIdAndPeriod(similarTeacherId,
            LocalDate.now().minusDays(90), LocalDate.now());

    for (ResourceUsage u : usage) {
        if (!targetResourceIds.contains(u.getResourceId())) {
            candidateScores.merge(u.getResourceId(), 1, Integer::sum);
        }
    }
}

// 步骤4: 按推荐分数排序, 返回 top-N 资源
return candidateScores.entrySet().stream()
    .sorted(Map.Entry.<String, Integer>comparingByValue().reversed())
    .limit(count)
    .map(entry -> resourceRepo.findById(entry.getKey()).orElse(null))
    .filter(Objects::nonNull)
    .collect(Collectors.toList());
}

/**
 * 计算两个教师资源集合的 Jaccard 相似度
 */
private double jaccardSimilarity(Set<String> setA, Set<String> setB) {
    Set<String> intersection = new HashSet<>(setA);
    intersection.retainAll(setB);
    Set<String> union = new HashSet<>(setA);
    union.addAll(setB);
    return union.isEmpty() ? 0.0 : (double) intersection.size() / union.size();
}
}

```

C.5 全文检索实现 (Elasticsearch)

C.5.1 资源索引配置

```

// resources_index_mapping.json (Elasticsearch 索引 Mapping)
{
  "mappings": {
    "properties": {
      "resource_id": { "type": "keyword" },
      "title": {
        "type": "text",
        "analyzer": "ik_max_word",

```

```

        "search_analyzer": "ik_smart",
        "fields": {
            "keyword": { "type": "keyword", "ignore_above": 256 }
        }
    },
    "description": { "type": "text", "analyzer": "ik_max_word" },
    "tags": { "type": "keyword" },
    "subject": { "type": "keyword" },
    "grade": { "type": "keyword" },
    "resource_type": { "type": "keyword" },
    "file_format": { "type": "keyword" },
    "upload_time": { "type": "date" },
    "download_count": { "type": "integer" },
    "rating": { "type": "float" },
    "is_official": { "type": "boolean" },
    "school_id": { "type": "keyword" },
    "uploader_id": { "type": "keyword" }
}
},
"settings": {
    "number_of_shards": 3,
    "number_of_replicas": 1,
    "analysis": {
        "analyzer": {
            "ik_with_pinyin": {
                "type": "custom",
                "tokenizer": "ik_max_word",
                "filter": ["pinyin_filter"]
            }
        }
    }
}
}
}
}

```

C.5.2 全文检索查询构建

```

// ResourceSearchService.java
@Service
public class ResourceSearchService {

    @Autowired
    private RestHighLevelClient esClient;

    public Page<Resource> search(ResourceSearchRequest request) {
        BoolQueryBuilder queryBuilder = QueryBuilders.boolQuery();

        // 关键词搜索 (标题+描述+标签)
        if (StringUtils.hasText(request.getKeyword())) {
            queryBuilder.must(QueryBuilders.multiMatchQuery(request.getKeyword())
                .field("title", 3.0f) // 标题权重×3
                .field("tags", 2.0f) // 标签权重×2
                .field("description", 1.0f) // 描述权重×1
                .type(MultiMatchQueryBuilder.Type.BEST_FIELDS)
                .minimumShouldMatch("75%")
            );
        }

        // 过滤条件 (精确匹配)
        if (request.getSubject() != null) {
            queryBuilder.filter(QueryBuilders.termQuery("subject", request.getSubject()));
        }
    }
}

```

```

        if (request.getGrade() != null) {
            queryBuilder.filter(QueryBuilders.termQuery("grade", request.getGrade()));
        }
        if (request.getResourceType() != null) {
            queryBuilder.filter(QueryBuilders.termQuery("resource_type",
                request.getResourceType()));
        }

        // 排序: 相关度 × 下载量 × 评分的综合排序
        ScriptSortBuilder scoreSort = SortBuilders.scriptSort(
            new Script("_score * Math.log(doc['download_count'].value + 1) " +
                "* doc['rating'].value"),
            ScriptSortBuilder.ScriptSortType.NUMBER
        ).order(SortOrder.DESC);

        SearchRequest searchRequest = new SearchRequest("resources");
        searchRequest.source(new SearchSourceBuilder()
            .query(queryBuilder)
            .sort(scoreSort)
            .from(request.getPage() * request.getPageSize())
            .size(request.getPageSize())
            .highlighter(new HighlightBuilder()
                .field("title")
                .field("description")
                .preTags("<em>").postTags("</em>"))
        );

        SearchResponse response = esClient.search(searchRequest, RequestOptions.DEFAULT);
        return parseSearchResponse(response);
    }
}

```

附录D 资源平台操作手册补充

D.1 资源批量导入工具

管理员可使用命令行工具批量导入现有教学资源：

```

# 批量导入命令示例
writech-resource-import \
    --type calligraphy \
    --dir /data/calligraphy-templates/ \
    --grade-map grade_mapping.json \
    --subject chinese \
    --school-id SCH001 \
    --dry-run # 先 dry-run 预览，确认后去掉此参数正式导入

```

grade_mapping.json 示例：

```

{
    "一年级上册": "grade_1_term_1",
    "一年级下册": "grade_1_term_2",
    "二年级上册": "grade_2_term_1",

```

```
"三年级上册": "grade_3_term_1"
}
```

D.2 资源审核操作说明

操作	权限	说明
上传资源	教师及以上	上传后自动进入内容审核流程
审核资源	学校资源管理员	仅可审核本校教师上传的资源
发布为公开资源	平台运营	将学校资源发布为平台共享资源
下架资源	学校资源管理员/平台运营	版权问题或内容问题时下架
批量打标签	资源管理员	为资源批量打知识点/年级标签
设置推荐权重	平台运营	调整优质资源在推荐系统中的权重

D.3 资源平台 API 完整清单

接口	方法	路径	说明
搜索资源	GET	/api/v1/resources/search	全文搜索资源（支持过滤和排序）
上传资源	POST	/api/v1/resources/upload	上传教学资源文件
获取预签名 URL	POST	/api/v1/resources/presign	获取直传 OSS 的预签名 URL
获取资源详情	GET	/api/v1/resources/{id}	获取单个资源的完整信息
获取下载链接	GET	/api/v1/resources/{id}/download-url	获取资源的 CDN 签名下载链接
收藏资源	POST	/api/v1/resources/{id}/collect	将资源加入个人收藏夹
取消收藏	DELETE	/api/v1/resources/{id}/collect	从收藏夹移除资源
获取收藏列表	GET	/api/v1/resources/collected	获取当前用户的收藏资源列表
评价资源	POST	/api/v1/resources/{id}/rating	对资源进行评分（1~5星）和评论
获取推荐资源	GET	/api/v1/resources/recommend	获取个性化推荐资源列表
获取字帖列表	GET	/api/v1/calligraphy/templates	获取字帖模板列表（分级分科）
获取字帖详情	GET	/api/v1/calligraphy/templates/{id}	获取字帖模板（笔顺数据+参考图）
批量导入（管理）	POST	/api/admin/resources/import	管理员批量导入资源
内容审核（管理）	PUT	/api/admin/resources/{id}/audit	审核/驳回资源
获取版权报告	GET	/api/admin/copyright/report	获取版权授权使用统计报告

附录E 核心技术实现补充

E.1 Elasticsearch全文检索实现

E.1.1 资源索引Mapping定义

```
PUT /writech_resources
{
  "settings": {
    "number_of_shards": 3,
    "number_of_replicas": 1,
    "analysis": {
      "analyzer": {
        "ik_max_word_analyzer": {
          "type": "custom",
          "tokenizer": "ik_max_word",
          "filter": ["lowercase", "synonym_filter"]
        },
        "ik_smart_pinyin": {
          "type": "custom",
          "tokenizer": "ik_smart",
          "filter": ["lowercase", "pinyin_filter"]
        }
      },
      "filter": {
        "pinyin_filter": {
          "type": "pinyin",
          "keep_full_pinyin": true,
          "keep_original": true,
          "limit_first_letter_length": 16
        },
        "synonym_filter": {
          "type": "synonym",
          "synonyms_path": "analysis/synonyms.txt"
        }
      }
    }
  },
  "mappings": {
    "properties": {
      "resource_id": { "type": "keyword" },
      "title": {
        "type": "text",
        "analyzer": "ik_max_word_analyzer",
        "search_analyzer": "ik_smart_pinyin",
        "fields": {
          "keyword": { "type": "keyword" },
          "suggest": { "type": "completion", "analyzer": "ik_smart_pinyin" }
        }
      },
      "content": { "type": "text", "analyzer": "ik_max_word_analyzer", "index_options": "offsets" },
      "subject": { "type": "keyword" },
      "grade": { "type": "keyword" },
      "resource_type": { "type": "keyword" },
      "tags": { "type": "keyword" },
      "download_count": { "type": "long" },
      "rating": { "type": "float" },
      "is_public": { "type": "boolean" },

```

```

        "created_at":      { "type": "date" }
    }
}
}

```

E.1.2 多字段全文搜索

```

# search/resource_searcher.py
from elasticsearch import Elasticsearch
from typing import Optional

class ResourceSearcher:
    INDEX = "writtech_resources"

    def __init__(self, es: Elasticsearch):
        self.es = es

    def search(self, keyword: str, subject: Optional[str] = None,
               grade: Optional[str] = None, page: int = 1, size: int = 20) -> dict:
        query = {
            "bool": {
                "must": [{
                    "multi_match": {
                        "query": keyword,
                        "fields": ["title^5", "description^2", "content^1", "tags^3"],
                        "type": "best_fields",
                        "operator": "and",
                        "fuzziness": "AUTO"
                    }
                ]
            },
            "filter": []
        }
        if subject:
            query["bool"]["filter"].append({"term": {"subject": subject}})
        if grade:
            query["bool"]["filter"].append({"term": {"grade": grade}})

        return self.es.search(
            index=self.INDEX,
            body={
                "query": query,
                "sort": [{"_score"}, {"download_count": "desc"}],
                "highlight": {
                    "pre_tags": ["<em>"], "post_tags": ["</em>"],
                    "fields": {
                        "title": {"number_of_fragments": 0},
                        "content": {"fragment_size": 150, "number_of_fragments": 3}
                    }
                },
                "from": (page - 1) * size,
                "size": size,
                "aggs": {
                    "by_subject": {"terms": {"field": "subject", "size": 10}},
                    "by_type": {"terms": {"field": "resource_type", "size": 10}}
                }
            }
        )

    def suggest(self, prefix: str, size: int = 8):
        """自动补全建议"""

```

```

resp = self.es.search(index=self.INDEX, body={
    "_source": False,
    "suggest": {
        "title_suggest": {
            "prefix": prefix,
            "completion": {"field": "title.suggest", "size": size}
        }
    }
})
return [o["text"] for o in resp["suggest"]["title_suggest"][0]["options"]]

```

E.2 协同过滤推荐实现

```

# recommendation/collaborative_filter.py
import numpy as np
from scipy.sparse import csr_matrix
from sklearn.metrics.pairwise import cosine_similarity
from typing import List, Tuple

class ItemCFRecommender:
    """基于物品的协同过滤推荐算法"""

    def fit(self, interactions: List[Tuple[str, str, float]]):
        """训练模型（用户，资源，交互分值）"""
        users = sorted(set(i[0] for i in interactions))
        items = sorted(set(i[1] for i in interactions))
        self.user_idx = {u: i for i, u in enumerate(users)}
        self.item_ids = items
        self.item_idx = {r: i for i, r in enumerate(items)}

        rows = [self.user_idx[u] for u, _, _ in interactions]
        cols = [self.item_idx[r] for _, r, _ in interactions]
        data = [s for _, _, s in interactions]
        self.matrix = csr_matrix((data, (rows, cols)),
                                shape=(len(users), len(items)))

        # 计算物品相似度
        self.similarity = cosine_similarity(self.matrix.T, dense_output=False)

    def recommend(self, user_id: str, top_k: int = 20,
                  exclude_seen: bool = True) -> List[Tuple[str, float]]:
        """为用户推荐资源，返回(resource_id, score)列表"""
        if user_id not in self.user_idx:
            return [] # 冷启动：回退到热门推荐

        uid = self.user_idx[user_id]
        user_vec = self.matrix[uid].toarray()[0]
        seen = set(np.where(user_vec > 0)[0])

        # 加权求和计算推荐分
        scores = np.zeros(len(self.item_ids))
        for item_idx in seen:
            sim_row = self.similarity[item_idx].toarray()[0]
            scores += user_vec[item_idx] * sim_row

        if exclude_seen:
            scores[list(seen)] = 0

        top_indices = np.argsort(scores)[::-1][:top_k]
        return [(self.item_ids[i], float(scores[i]))
                for i in top_indices if scores[i] > 0]

```

E.3 CDN防盗链签名算法

```
# cdn/sign_url.py
import hashlib, time, hmac

def sign_cdn_url(url: str, secret: str, expire_seconds: int = 3600) -> str:
    """
    生成CDN防盗链签名URL
    格式: {url}?token={md5(secret+expire+path)}&t={expire}
    """
    expire_ts = int(time.time()) + expire_seconds
    from urllib.parse import urlparse
    path = urlparse(url).path
    raw = f"{secret}{expire_ts}{path}"
    token = hashlib.md5(raw.encode()).hexdigest()
    separator = "&" if "?" in url else "?"
    return f"{url}{separator}token={token}&t={expire_ts}"

def verify_cdn_url(url: str, token: str, expire_ts: int, secret: str) -> bool:
    """验证CDN签名URL (网关侧) """
    if int(time.time()) > expire_ts:
        return False # 已过期
    from urllib.parse import urlparse
    path = urlparse(url).path
    expected = hashlib.md5(f"{secret}{expire_ts}{path}".encode()).hexdigest()
    return hmac.compare_digest(token, expected) # 使用常量时间比较, 防止时序攻击
```

E.4 数字水印嵌入

```
# security/watermark.py
import numpy as np
from PIL import Image
import struct

class LSBWatermark:
    """LSB最低有效位数字水印 (嵌入式隐写) """

    @staticmethod
    def embed(img_array: np.ndarray, user_id: str) -> np.ndarray:
        """在图片的R通道LSB中嵌入用户ID"""
        watermark_bytes = user_id.encode('utf-8')[:16] # 最多16字节
        header = struct.pack('>H', len(watermark_bytes)) # 2字节长度头
        payload = header + watermark_bytes
        bits = ''.join(f'{b:08b}' for b in payload)

        result = img_array.copy()
        flat_r = result[:, :, 0].flatten()
        for i, bit in enumerate(bits):
            if i >= len(flat_r): break
            flat_r[i] = (flat_r[i] & 0xFE) | int(bit)
        result[:, :, 0] = flat_r.reshape(result[:, :, 0].shape)
        return result

    @staticmethod
    def extract(img_array: np.ndarray) -> str:
        """从图片LSB中提取水印 (用户ID) """
        flat_r = img_array[:, :, 0].flatten()
        # 先读16位 (2字节头) 获取长度
        length_bits = ''.join(str(flat_r[i] & 1) for i in range(16))
```

```
length = int(length_bits, 2)
if length > 16: return ""

payload_bits = ''.join(str(flat_r[i] & 1) for i in range(16, 16 + length * 8))
payload_bytes = bytes(int(payload_bits[i:i+8], 2)
                        for i in range(0, len(payload_bits), 8))

try:
    return payload_bytes.decode('utf-8')
except UnicodeDecodeError:
    return ""
```

E.5 性能指标

指标	值
Elasticsearch搜索P99延迟	< 80ms
CDN文件下载平均速度	> 20MB/s（国内）
水印嵌入速度（单图）	< 50ms
协同过滤推荐响应时间	< 30ms（预计算缓存）
并发上传支持	500连接/秒
内容审核平均耗时	< 2秒/资源

本文档版权归深圳自然写科技有限公司所有，仅用于软件著作权登记鉴别。

附录E 补充技术规格

E.1 资源上传与处理流水线

E.1.1 异步文件处理任务

```
// ResourceProcessingPipeline.java
@Service
public class ResourceProcessingPipeline {

    @Autowired
    private RocketMQTemplate mqTemplate;

    @Autowired
    private MinioClient minioClient;

    @Autowired
    private AiModerationService moderationService;

    /**
     * 资源上传后触发异步处理流水线：
     * 上传 → 病毒扫描 → 内容审核 → 格式转换 → CDN预热
     */
}
```

```

public void triggerProcessing(String resourceId, String bucketPath) {
    ProcessingTask task = ProcessingTask.builder()
        .resourceId(resourceId)
        .bucketPath(bucketPath)
        .steps(List.of(
            ProcessingStep.VIRUS_SCAN,
            ProcessingStep.CONTENT_MODERATION,
            ProcessingStep.FORMAT_CONVERSION,
            ProcessingStep.CDN_WARMUP
        ))
        .build();

    mqTemplate.syncSend("resource-processing-topic", task);
}

@RocketMQMessageListener(
    topic = "resource-processing-topic",
    consumerGroup = "resource-processor"
)
@Component
class ProcessingConsumer implements RocketMQListener<ProcessingTask> {

    @Override
    public void onMessage(ProcessingTask task) {
        try {
            executeStep(task, ProcessingStep.VIRUS_SCAN,
                () -> virusScan(task.getBucketPath()));

            executeStep(task, ProcessingStep.CONTENT_MODERATION,
                () -> moderationService.checkContent(task.getResourceId()));

            executeStep(task, ProcessingStep.FORMAT_CONVERSION,
                () -> convertFormat(task.getResourceId(), task.getBucketPath()));

            executeStep(task, ProcessingStep.CDN_WARMUP,
                () -> cdnService.warmup(task.getResourceId()));

            resourceRepo.updateStatus(task.getResourceId(),
                ResourceStatus.PUBLISHED);

        } catch (ProcessingException e) {
            resourceRepo.updateStatus(task.getResourceId(),
                ResourceStatus.FAILED);
            log.error("资源处理失败: {}", task.getResourceId(), e);
        }
    }

    private void executeStep(ProcessingTask task,
        ProcessingStep step,
        Runnable action) {
        log.info("执行步骤 {}: {}", step, task.getResourceId());
        resourceRepo.updateProcessingStep(task.getResourceId(), step);
        action.run();
    }
}

```

E.2 版权保护与水印系统

E.2.1 可见水印叠加

```
// VisibleWatermarkService.java
import java.awt.*;
import java.awt.image.BufferedImage;
import javax.imageio.ImageIO;

@Service
public class VisibleWatermarkService {

    /**
     * 在图片上叠加半透明文字水印
     */
    public BufferedImage addTextWatermark(BufferedImage source,
                                           String watermarkText) {

        int width = source.getWidth();
        int height = source.getHeight();

        BufferedImage watermarked = new BufferedImage(
            width, height, BufferedImage.TYPE_INT_ARGB);

        Graphics2D g2d = watermarked.createGraphics();
        g2d.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
            RenderingHints.VALUE_ANTIALIAS_ON);
        g2d.drawImage(source, 0, 0, null);

        // 水印样式
        Font font = new Font("微软雅黑", Font.BOLD, Math.max(24, width / 30));
        g2d.setFont(font);
        g2d.setColor(new Color(128, 128, 128, 80)); // 灰色半透明

        // 斜角平铺水印
        g2d.rotate(Math.toRadians(-30), width / 2.0, height / 2.0);
        FontMetrics fm = g2d.getFontMetrics();
        int textWidth = fm.stringWidth(watermarkText);
        int textHeight = fm.getHeight();

        for (int y = -height; y < height * 2; y += textHeight * 4) {
            for (int x = -width; x < width * 2; x += textWidth * 2) {
                g2d.drawString(watermarkText, x, y);
            }
        }

        g2d.dispose();
        return watermarked;
    }

    /**
     * 在PDF文档页面叠加水印
     */
    public void addPdfWatermark(InputStream pdfIn, OutputStream pdfOut,
                                String watermarkText) throws Exception {

        PdfReader reader = new PdfReader(pdfIn);
        PdfStamper stamper = new PdfStamper(reader, pdfOut);

        BaseFont baseFont = BaseFont.createFont(
            "STSong-Light", "UniGB-UCS2-H", BaseFont.NOT_EMBEDDED);

        int numPages = reader.getNumberOfPages();
        for (int i = 1; i <= numPages; i++) {
            PdfContentByte canvas = stamper.getUnderContent(i);
            Rectangle pageSize = reader.getPageSize(i);

            canvas.saveState();

```

```

        canvas.setGState(new PdfGState());
        canvas.setFontAndSize(baseFont, 36);
        canvas.setColorFill(new BaseColor(192, 192, 192, 60));

        canvas.beginText();
        canvas.showTextAligned(Element.ALIGN_CENTER,
            watermarkText,
            pageSize.getWidth() / 2,
            pageSize.getHeight() / 2,
            45);
        canvas.endText();
        canvas.restoreState();
    }

    stamper.close();
    reader.close();
}
}

```

E.3 搜索索引维护

E.3.1 Elasticsearch索引更新策略

```

// ResourceSearchIndexer.java
@Service
public class ResourceSearchIndexer {

    private static final String INDEX_NAME = "writtech_resources";

    @Autowired
    private ElasticsearchClient esClient;

    @Async
    public void indexResource(Resource resource) {
        try {
            ResourceDocument doc = ResourceDocument.builder()
                .id(resource.getId())
                .title(resource.getTitle())
                .description(resource.getDescription())
                .subject(resource.getSubject())
                .grade(resource.getGrade())
                .tags(resource.getTags())
                .contentText(extractText(resource)) // 提取全文
                .uploadTime(resource.getCreatedAt())
                .downloadCount(resource.getDownloadCount())
                .rating(resource.getAverageRating())
                .build();

            IndexRequest<ResourceDocument> request = IndexRequest.of(r -> r
                .index(INDEX_NAME)
                .id(doc.getId())
                .document(doc));

            esClient.index(request);
            log.debug("Indexed resource: {}", resource.getId());

        } catch (IOException e) {
            log.error("ES索引更新失败: {}", resource.getId(), e);
        }
    }
}

```



```

private String extractText(Resource resource) {
    // 根据资源类型提取全文内容
    return switch (resource.getType()) {
        case PDF -> pdfTextExtractor.extract(resource.getBucketPath());
        case WORD -> wordTextExtractor.extract(resource.getBucketPath());
        case PPT -> pptTextExtractor.extract(resource.getBucketPath());
        default -> resource.getDescription();
    };
}

/**
 * 全文搜索接口
 */
public SearchResult search(SearchQuery query) throws IOException {
    SearchRequest request = SearchRequest.of(s -> s
        .index(INDEX_NAME)
        .query(q -> q
            .bool(b -> b
                .must(m -> m
                    .multiMatch(mm -> mm
                        .query(query.getKeyword())
                        .fields("title^3", "description^2", "contentText", "tags^2")
                        .type(TextQueryType.BestFields)
                    )
                )
            .filter(f -> f
                .term(t -> t.field("grade").value(query.getGrade()))
            )
            .filter(f -> f
                .term(t -> t.field("subject").value(query.getSubject()))
            )
        )
        .highlight(h -> h
            .fields("title", hf -> hf)
            .fields("description", hf -> hf)
            .preTags("<em>")
            .postTags("</em>")
        )
        .from(query.getPage() * query.getPageSize())
        .size(query.getPageSize())
    );

    SearchResponse<ResourceDocument> response = esClient.search(
        request, ResourceDocument.class);

    return buildSearchResult(response);
}
}

```

附录F 补充技术规格

F.1 资源推荐算法

F.1.1 基于内容的过滤

```

// ContentBasedRecommender.java
@Service
public class ContentBasedRecommender {

    @Autowired
    private ElasticsearchClient esClient;

    /**
     * 基于用户最近浏览的资源，推荐相似内容
     */
    public List<Resource> recommend(String userId, int limit) {
        // 1. 获取用户最近浏览的5个资源
        List<Resource> recentViewed = viewHistoryRepo
            .findRecentByUserId(userId, 5);

        if (recentViewed.isEmpty()) {
            return getPopularResources(limit);
        }

        // 2. 构建用户兴趣画像（基于标签频率）
        Map<String, Long> tagFrequency = recentViewed.stream()
            .flatMap(r -> r.getTags().stream())
            .collect(Collectors.groupingBy(Function.identity(), Collectors.counting()));

        // 取频率最高的5个标签
        List<String> topTags = tagFrequency.entrySet().stream()
            .sorted(Map.Entry.<String, Long>comparingByValue().reversed())
            .limit(5)
            .map(Map.Entry::getKey)
            .collect(Collectors.toList());

        // 3. 基于标签进行ES相似查询
        Set<String> excludeIds = recentViewed.stream()
            .map(Resource::getId).collect(Collectors.toSet());

        return searchByTags(topTags, excludeIds, limit);
    }

    private List<Resource> searchByTags(List<String> tags,
                                       Set<String> excludeIds,
                                       int limit) throws IOException {
        SearchRequest req = SearchRequest.of(s -> s
            .index("writech_resources")
            .query(q -> q
                .bool(b -> {
                    // 标签匹配（任意标签命中即可）
                    for (String tag : tags) {
                        b.should(sh -> sh.term(t -> t.field("tags").value(tag)));
                    }
                    b.minimumShouldMatch("1");
                    // 排除已浏览资源
                    excludeIds.forEach(id ->
                        b.mustNot(mn -> mn.ids(i -> i.values(id))));
                    return b;
                })
            )
            .size(limit)
        );

        SearchResponse<ResourceDocument> resp = esClient.search(req, ResourceDocument.class);
        return resp.hits().hits().stream()
            .map(hit -> resourceRepo.findById(hit.id()))
    }
}

```

```

        .filter(Optional::isPresent)
        .map(Optional::get)
        .collect(Collectors.toList());
    }
}

```

F.2 资源下载统计

```

// DownloadCountService.java
@Service
public class DownloadCountService {

    @Autowired
    private RedisTemplate<String, Long> redisTemplate;

    private static final String DOWNLOAD_COUNT_KEY = "resource:downloads:";
    private static final String BATCH_SYNC_SET = "resource:sync:pending";

    /**
     * 记录下载（先写Redis，定期同步到数据库）
     */
    public void recordDownload(String resourceId, String userId) {
        String key = DOWNLOAD_COUNT_KEY + resourceId;
        redisTemplate.opsForValue().increment(key);

        // 标记需要同步
        redisTemplate.opsForSet().add(BATCH_SYNC_SET, resourceId);

        // 记录用户下载历史（用于去重统计）
        String userKey = "resource:downloaded:" + userId;
        redisTemplate.opsForSet().add(userKey, resourceId);
        redisTemplate.expire(userKey, 30, TimeUnit.DAYS);
    }

    public long getDownloadCount(String resourceId) {
        Long count = redisTemplate.opsForValue().get(DOWNLOAD_COUNT_KEY + resourceId);
        if (count != null) return count;
        // Redis缓存未命中，查数据库
        return resourceRepo.getDownloadCount(resourceId);
    }

    /**
     * 定期将Redis中的计数同步到数据库（每5分钟）
     */
    @Scheduled(fixedDelay = 5 * 60 * 1000)
    public void syncCountsToDatabase() {
        Set<String> pendingIds = redisTemplate.opsForSet().members(BATCH_SYNC_SET);
        if (pendingIds == null || pendingIds.isEmpty()) return;

        pendingIds.forEach(resourceId -> {
            Long count = redisTemplate.opsForValue().get(DOWNLOAD_COUNT_KEY + resourceId);
            if (count != null) {
                resourceRepo.updateDownloadCount(resourceId, count);
                redisTemplate.opsForSet().remove(BATCH_SYNC_SET, resourceId);
            }
        });

        log.debug("同步下载计数，共{}个资源", pendingIds.size());
    }
}

```

F.3 资源分类管理

```
// CategoryTreeService.java
@Service
@CacheEvict(value = "categoryTree", allEntries = true)
public class CategoryTreeService {

    @Cacheable("categoryTree")
    public List<CategoryNode> getCategoryTree() {
        List<Category> allCategories = categoryRepo.findAll();

        // 构建树形结构
        Map<String, CategoryNode> nodeMap = allCategories.stream()
            .collect(Collectors.toMap(Category::getId,
                c -> new CategoryNode(c.getId(), c.getName(), c.getParentId())));

        List<CategoryNode> roots = new ArrayList<>();

        for (CategoryNode node : nodeMap.values()) {
            if (node.getParentId() == null) {
                roots.add(node);
            } else {
                CategoryNode parent = nodeMap.get(node.getParentId());
                if (parent != null) {
                    parent.addChild(node);
                }
            }
        }

        // 按序号排序
        sortTree(roots);
        return roots;
    }

    private void sortTree(List<CategoryNode> nodes) {
        nodes.sort(Comparator.comparingInt(CategoryNode::getSortOrder));
        nodes.forEach(n -> sortTree(n.getChildren()));
    }

    public long countResourcesByCategory(String categoryId) {
        // 包含子分类的资源数量
        List<String> categoryIds = getAllDescendantIds(categoryId);
        categoryIds.add(categoryId);
        return resourceRepo.countByCategoryIdIn(categoryIds);
    }

    private List<String> getAllDescendantIds(String categoryId) {
        List<String> ids = new ArrayList<>();
        Queue<String> queue = new LinkedList<>();
        queue.offer(categoryId);

        while (!queue.isEmpty()) {
            String id = queue.poll();
            List<Category> children = categoryRepo.findByParentId(id);
            children.forEach(c -> {
                ids.add(c.getId());
                queue.offer(c.getId());
            });
        }
        return ids;
    }
}
```

```
}  
}
```

附录G 补充技术规格

G.1 资源审核 workflow

```
// ResourceReviewWorkflow.java  
@Service  
public class ResourceReviewWorkflow {  
  
    public enum ReviewState {  
        PENDING_AUTO,    // 等待自动审核  
        AUTO_PASS,        // 自动审核通过  
        AUTO_REJECT,      // 自动审核拒绝 (违规内容)  
        PENDING_MANUAL,   // 等待人工审核 (自动审核可疑)  
        MANUAL_PASS,      // 人工审核通过  
        MANUAL_REJECT     // 人工审核拒绝  
    }  
  
    @Autowired  
    private ContentModerationService moderationService;  
  
    @Autowired  
    private ReviewQueueService reviewQueue;  
  
    public void startReview(String resourceId) {  
        Resource resource = resourceRepo.findById(resourceId)  
            .orElseThrow(() -> new ResourceNotFoundException(resourceId));  
  
        resourceRepo.updateReviewState(resourceId, ReviewState.PENDING_AUTO);  
  
        // 异步执行AI审核  
        CompletableFuture.runAsync(() -> {  
            ModerationResult result = moderationService.check(resource);  
  
            if (result.isClean()) {  
                resourceRepo.updateReviewState(resourceId, ReviewState.AUTO_PASS);  
                resourceRepo.updateStatus(resourceId, ResourceStatus.PUBLISHED);  
                notifyUploader(resource.getUploaderId(), "审核通过, 资源已上线");  
            }  
  
            } else if (result.isSuspicious()) {  
                resourceRepo.updateReviewState(resourceId, ReviewState.PENDING_MANUAL);  
                reviewQueue.enqueue(resourceId, result.getReason());  
            }  
  
            } else {  
                resourceRepo.updateReviewState(resourceId, ReviewState.AUTO_REJECT);  
                resourceRepo.updateStatus(resourceId, ResourceStatus.REJECTED);  
                notifyUploader(resource.getUploaderId(),  
                    "审核未通过: " + result.getRejectionReason());  
            }  
        });  
    }  
  
    public void manualReview(String resourceId, String reviewerId,  
        boolean approved, String comment) {  
        ReviewState newState = approved ? ReviewState.MANUAL_PASS : ReviewState.MANUAL_REJECT;
```

```

        ResourceStatus newStatus = approved ? ResourceStatus.PUBLISHED : ResourceStatus.REJECTED;

        resourceRepo.updateReviewState(resourceId, newState);
        resourceRepo.updateStatus(resourceId, newStatus);

        // 记录审核日志
        reviewLogRepo.save(ReviewLog.builder()
            .resourceId(resourceId)
            .reviewerId(reviewerId)
            .decision(approved ? "PASS" : "REJECT")
            .comment(comment)
            .reviewedAt(Instant.now())
            .build());

        // 通知上传者
        Resource resource = resourceRepo.findById(resourceId).get();
        notifyUploader(resource.getUploaderId(),
            approved ? "资源审核通过，已发布" : "资源审核未通过：" + comment);
    }
}

```

G.2 资源版权信息管理

```

// CopyrightService.java
@Service
public class CopyrightService {

    public void registerCopyright(String resourceId, CopyrightInfo info) {
        // 验证版权信息完整性
        validateCopyright(info);

        // 生成版权证书编号
        String certNo = generateCertNumber(resourceId);
        info.setCertificateNumber(certNo);
        info.setRegistrationDate(LocalDate.now());

        copyrightRepo.save(CopyrightRecord.builder()
            .resourceId(resourceId)
            .info(info)
            .createdAt(Instant.now())
            .build());

        // 在资源文件中嵌入不可见水印
        watermarkService.embedInvisibleWatermark(resourceId, certNo);
    }

    public boolean verifyCopyright(String resourceId, String certNo) {
        return copyrightRepo.findByResourceIdAndCertNo(resourceId, certNo).isPresent();
    }

    private String generateCertNumber(String resourceId) {
        String hash = DigestUtils.sha256Hex(
            resourceId + Instant.now().toEpochMilli());
        return "WRC-" + hash.substring(0, 8).toUpperCase();
    }

    private void validateCopyright(CopyrightInfo info) {
        if (info.getAuthor() == null || info.getAuthor().isBlank()) {
            throw new IllegalArgumentException("版权作者不能为空");
        }
    }
}

```

```
        if (info.getLicenseType() == null) {
            throw new IllegalArgumentException("必须指定授权类型");
        }
    }
}
```

附录H 版本历史与术语表

H.1 版本历史

版本号	发布日期	变更说明	负责人
V1.0.0	2024-01	初始版本，完成资源上传、下载、搜索核心功能	研发团队
V1.1.0	2024-03	新增AI内容审核流水线，支持病毒扫描与违规检测	研发团队
V1.2.0	2024-05	集成ES全文检索，支持多字段高亮搜索	研发团队
V1.3.0	2024-07	上线版权保护模块，支持可见/不可见双重水印	研发团队
V1.4.0	2024-09	新增个性化推荐引擎（标签协同过滤算法）	研发团队
V1.5.0	2024-11	优化CDN预热策略，热门资源首次访问延迟降低60%	研发团队

H.2 术语表

术语	说明
CDN	Content Delivery Network，内容分发网络，用于加速资源下载
ES	Elasticsearch，分布式全文搜索引擎
RocketMQ	阿里巴巴开源消息队列，用于异步处理资源上传流水线
协同过滤	基于用户行为相似度进行推荐的算法
OCR	光学字符识别，用于提取资源内文字信息建立全文索引
WORM	Write Once Read Many，版权存证数据的不可篡改存储策略

本文档版权归深圳自然写科技有限公司所有，仅用于软件著作权登记鉴别。