校园公告发布平台“青言速递”架构设计草案

版本号： V1.0

名字: 青言速递

发布日期： 2025-10-10

编写人： 江周霖

一、项目概述与设计目标

“青言速递”是一款基于 Web 的校园公告发布与传播平台，旨在为全校师生提供一个统一、便捷、高效的信息发布与接收渠道。系统的目标是取代传统微信群与线下公告栏，实现校园信息的数字化与智能化传递。

为实现这一目标，系统架构设计遵循以下原则：

- **高可用性**：支持百人级并发访问，页面响应时间控制在 3 秒以内；
- **可扩展性**：采用模块化设计，便于后续接入微信小程序或移动端；
- **安全性**：基于 JWT 实现统一身份认证与角色访问控制；
- **高性能**：通过缓存与索引优化保障系统响应速度；
- **易部署性**：利用 Docker 容器化简化部署与运维。

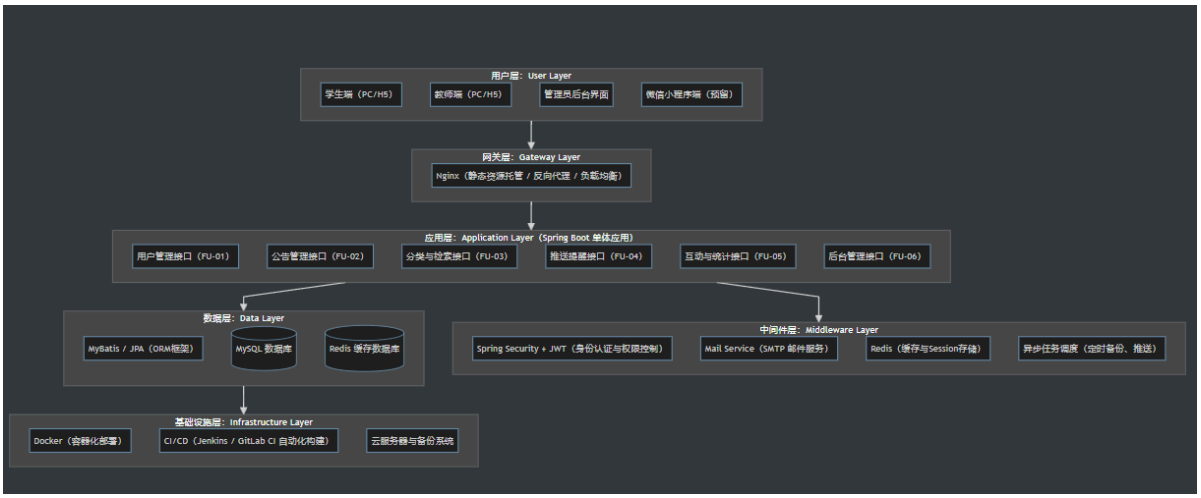
二、总体架构设计

系统采用 **B/S 模式 + 前后端分离架构**，前端与后端通过 RESTful API 进行通信，将页面展示、业务逻辑和数据存储进行分离，确保系统结构清晰、维护便捷。

架构设计理念

- **分层清晰**：前端负责交互与展示，后端处理业务逻辑，数据库专注数据持久化；
- **模块独立**：功能模块独立开发，统一接口规范；
- **通信统一**：前后端通信采用 HTTPS + JSON；
- **扩展预留**：系统结构中预留微信小程序与消息推送接口。

三、系统总体架构图（逻辑分层）



四、架构分层说明

1 用户层（User Layer）

- 主要包含三类角色：学生、教师、管理员；
- 前端采用响应式布局，适配 PC、平板与移动端；
- 管理员可登录后台界面执行管理与审核操作；
- 预留微信小程序接口，便于未来扩展访问方式。

2 网关层（Gateway Layer）

- **Nginx**：负责静态资源托管、反向代理与负载均衡；
- 在请求转发时，执行基础的安全校验和跨域配置；
- 不采用独立网关服务，而是通过 Nginx 配置实现轻量化管理。

3 应用层（Application Layer）

核心业务逻辑集中在一个 **Spring Boot 单体应用** 中，各功能通过独立的接口模块组织。

模块	职责
FU-01 用户管理	注册、登录、个人资料修改、角色分配
FU-02 公告管理	公告创建、编辑、审核、发布与撤回
FU-03 分类与检索	公告分类维护、关键词搜索、分页展示
FU-04 推送提醒	公告发布后即时通知与浏览器提醒
FU-05 互动与统计	评论、点赞、浏览量统计与数据分析
FU-06 后台管理	用户与公告审核、权限配置、图表统计

所有模块均通过 **Service 层** 调用接口逻辑，确保代码复用与可维护性。

4 中间件层（Middleware Layer）

- **Spring Security + JWT**：实现统一登录认证与权限控制，区分角色（student、teacher、admin）；
- **Redis 缓存与会话管理**：缓存公告列表、热门信息、统计数据；
- **邮件服务（JavaMail）**：用于注册验证、密码重置、系统通知；
- **异步任务调度**：执行缓存清理、数据备份与定时推送等任务。

---

5 数据层（Data Layer）

- **MySQL**：存储用户、公告、评论等结构化数据；
  - 对高频字段（如发布时间、分类 ID、作者 ID）建立索引；
  - 支持全文索引以优化搜索性能；
- **Redis**：负责热点数据与访问计数缓存；
- **ORM 框架（MyBatis / JPA）**：提升数据访问安全与开发效率。

---

6 基础设施层（Infrastructure Layer）

- **Docker**：后端、数据库、Redis、Nginx 均以容器形式运行，保证环境一致性；
- **CI/CD**：使用 Jenkins 或 GitLab CI 实现代码提交 → 构建 → 测试 → 部署的自动化流程；
- **云服务器与备份**：MySQL 采用每日全量与定时增量备份，日志集中化管理。

五、系统部署架构图（Docker 化）

六、核心实现说明

1 用户认证与权限控制

- 使用 **JWT（JSON Web Token）** 实现无状态登录；
- 前端请求统一携带 Token，后端通过 Spring Security 校验；
- 根据用户角色动态授权访问接口。

2 公告发布与审核流程

- 教师创建公告后，若分类需审核 → 状态置为 `pending`；
- 管理员审核后可将状态改为 `published` 或 `rejected`；
- 公告发布后自动写入 Redis 缓存，并推送给订阅用户。

3 评论与统计设计

- 评论支持多级嵌套（通过 `parent_id` 实现）；
- 浏览量统计使用 Redis 自增计数，并定期异步同步到 MySQL；
- 数据统计通过 SQL 聚合 + Redis 缓存结合实现，使用 ECharts 展示结果。

4 推送机制

- 通过 WebSocket 实现实时消息推送；
- 邮件用于系统级通知（如审核结果、重要公告）。

七、非功能性需求实现策略

领域	实现措施
性能	Redis 缓存；分页与索引优化；CDN 加速前端资源
安全	JWT + HTTPS；BCrypt 密码加密；防 SQL 注入与 XSS
可靠性	Docker 环境部署；定期备份；日志与监控系统
易用性	简洁的公告发布流程；统一 UI 设计；移动端兼容
扩展性	模块解耦；接口标准化；预留小程序与推送扩展接口

八、技术选型与版本建议

层级	技术选型	说明
前端	Vue 3 + Element Plus	响应式 UI，后台与前端统一框架
后端	Spring Boot 3.x + Spring Security + JWT	结构简洁，开发效率高
数据层	MySQL 8.0 + Redis 6.x	稳定高效的数据存储方案
构建部署	Maven + Docker + Jenkins	支持自动化部署流程
通知推送	JavaMail（SMTP） / WebSocket	实时通信机制

九、总结与展望

该架构方案基于实际校园公告需求，从简洁性与实用性出发，采用单体架构与模块化设计，既保证开发与部署的高效性，也为后续功能扩展保留空间。其主要优势包括：

- **结构清晰，逻辑集中**：单体结构降低运维复杂度；
- **安全体系完善**：JWT + HTTPS 全方位保障；
- **易于扩展**：接口化设计便于未来增加功能模块；
- **部署简单可靠**：Docker 环境一键构建、快速上线。

是否希望我接下来为此版本生成一份 **正式发布版 Markdown 文件（.md）**，以便你直接在 Typora 或 GitHub 项目中使用？

十、后续可以进行扩展的方向增加

1. 增加“点赞”、“收藏”、“订阅”等互动功能；
2. 接入 **微信小程序** 或 **校园 App 端**；
3. 优化全文检索功能，可考虑接入 ElasticSearch；
4. 增强后台统计可视化与日志分析能力。