# 关于内网引入开发环境组件的建议说明

# 一、背景与目的

当前正在筹备开发一套面向内部的数据报表生产管理系统。系统预期将具备现代化的数据报表的前端可视化功能、模块化服务架构，以及良好的扩展性与维护性。

在进行前期技术选型时，已在 GitHub 上评估并选定一个成熟开源框架，具备良好的结构设计、社区支持，并能较好支持当前系统建设目标。然而，在

实际落地过程中，由于现有内网开发环境受限（仅支持 JDK 8，缺乏 Redis、Vue3、Nginx 等依赖环境），导致开发工作难以推进。

**本提案旨在：**在数据安全原则的前提下，申请在开发环境中部署必要的现代开发组件，以支撑项目顺利实施。

二、组件清单与用途说明

|  |  |  |  |
| --- | --- | --- | --- |
| 组件名称 | 当前状态 | 应用场景/用途 | 安全说明 |
| JDK 21 | 未安装 | 支持开源框架核心运行（Spring Boot 3.x） | 可通过官网下载后离线部署，无需联网 |
| Redis | 未安装 | 用于处理系统缓存、状态保持、实时交互功能 | 仅本地运行，端口受控 |
| Nginx | 未安装 | 用于静态资源托管、前后端转发 | 本地代理，网络隔离 |
| Git | 未安装 | 用于同步代码更新、克隆代码分支 | 离线部署，无需联网 |

为便于清晰理解，以下列出所需核心组件及其用途：上述组件为当前主流开源技术，广泛用于各行业系统开发，具备稳定性与长期可维护性。

三、目前开发的难点

目前在现有内网环境下进行系统开发，面临如下主要困难：

### 1.缺失代码管理工具

开发时如果缺少 Git等代码管理工具，项目无法进行有效的版本控制。会导致每次修改代码都要手动备份，文件管理混乱，容易覆盖或丢失重要代码。并且如果多人同时开发，代码合并只能靠复制粘贴。难以追踪谁改了什么、什么时候改的、为什么改，冲突处理完全靠人工非常容易引入bug。

### 2. 缺乏前端构建工具，无法实现图表和可视化页面

系统的图形化的功能（如 ECharts 图表、业务看板等），但目前内网环境缺少 Vue3 + Node.js 构建支持，目前的数据图表仅靠传统 HTML、JSP难以满足交互体验要求，手工实现耗时高且维护成本大。

### 3. 组件无法安装，严重拖慢开发进度

当前内网仅支持 JDK 8，而选定的开源框架基于 JDK 21 开发。如果不用框架开发，如“权限鉴权”、“日志管理”等功能可能会缺少jar包，导致功能无法正常开发亦或是效率较慢（需要重新手写这部分功能代码）。

## 四、引入后的技术收益

部署以上组件将带来如下优势：

1. 满足系统功能需求：支持图表、交互式界面等现代业务功能，提升用户体验。
2. 提高开发效率：现有框架结构清晰、功能完整，可快速开发上线，避免重复造轮子。
3. 提升系统质量与扩展性：新版本 JDK 对性能、内存管理等优化显著，有助于后期稳定运行。
4. 利于后期维护与交接：符合当前开发主流，便于新成员快速上手，维护成本低。
5. 利于项目的打包部署：降低系统运行维护的成本，提高系统部署的复杂性。

## 五、风险控制措施

在确保不对内网安全造成影响前提下，本方案拟采用如下方式保障开发环境安全：

**全离线部署：**所有组件和依赖包均由开发人员在外网下载后，经安全扫描导入内网，不涉及外部访问或更新。

**隔离环境测试：**如有顾虑，可先部署在测试/沙箱环境中进行功能验证，确认安全可控后再正式应用。

|  |  |  |  |
| --- | --- | --- | --- |
| 环境类型 | 使用场景 | 核心组件 | 推荐配置说明 |
| 开发环境 | 开发人员本地编程调试 | IntelliJ IDEA、JDK 21、Node.JS、Vue CLI | CPU：16 核 |
| 内存：32GB |
| 硬盘：500GB SSD |
| 测试环境 | 联调、测试验证 | Nginx、Redis、GitSpringBoot、Vue3 编译产物 | 服务器 x2（前端 + 后端） |
| 每台 CPU：16 核 / 内存：32GB / 硬盘：500GB SSD |
| 生产环境 | 正式部署使用 | 同测试环境，去除测试工具，部署完整配置的业务模块 | 服务器 x2（前端 + 后端） |

## 六、环境与组件配置