# 项目设计方案：变电站智能数字图书馆

## 1. 项目背景与目标

### 1.1. 项目背景

变电站图纸档案是电网安全运行的核心技术资料。然而，传统的档案管理方式仍面临严峻挑战，集中体现在：

1. **“图-物-档”信息孤岛**：图纸（CAD、扫描件）、设备实物（断路器、变压器）和设备档案（台账、定值单、修改单）三者之间的数据严重脱节，信息不一致。
2. **版本混乱与安全隐患**：图纸“一图-多版”现象普遍，运维人员极易拿到过期图纸进行操作，这是重大安全隐患。修改流程不透明、追溯困难。
3. **检索效率低下**：非结构化图纸和文档内容无法被有效搜索，工程师排查故障或技改时，如同“大海捞针”。

### 1.2. 项目目标

本项目旨在**融合基础文件管理与人工智能技术**，构建一个集“管理、存储、智能分析、主动服务”于一体的**变电站智能数字图书馆**。

* **近期目标（基础 - PDF）**：实现《变电站施工图纸管控工具》中的所有基础功能，包括用户权限、厂站台账、图纸文件的上传、下载、删除、重命名、检索和首页统计。
* **远期目标（智能 - DOCX）**：引入“AI + KG + Agent”架构，实现以下核心价值：
  1. **AI赋能**：自动“读懂”图纸和文档内容。
  2. **KG连接**：构建“图-物-档”数字孪生知识大脑，彻底打通信息孤岛。
  3. **Agent服务**：提供智能查询助手和自动化工作流，变“被动存储”为“主动服务”，从根本上解决版本混乱问题，保障电网安全。

## 2. 总体设计思路：“AI + KG + Agent”三层模型

为实现上述目标，平台将基于“基础功能 + 智能引擎”的思路构建，其核心智能化设计遵循三层模型：

1. **AI内容理解层（赋能“读懂”）**
   * **职责**：将所有非结构化、半结构化的图纸和档案“读懂”，提取为结构化信息。
   * **技术**：
     + **智能OCR/CV（针对图纸）**：自动识别图框信息（名称、编号、版本）、电气符号（断路器、隔离开关等）、设备清单，并（核心能力）**重建电气拓扑连接关系**。
     + **NLP（针对文档）**：自动从修改单、运维记录、说明书中提取设备ID、故障现象、修改内容等关键实体。
2. **知识图谱中枢层（构建“大脑”）**
   * **职责**：解决“图-物-档”脱节的核心。将AI提取的所有零散信息点，连接成一个统一的变电站知识网络。
   * **技术**：构建变电站本体（Ontology），定义实体（如：变电站、设备、图纸、修改单、台账）及它们之间的关系（如：设备-[绘制于]->图纸，修改单-[修改了]->图纸，设备-[关联]->台账）。
   * **价值**：实现关联推理。例如，查询一个设备时，系统能瞬时返回所有相关图纸、台账和历史修改单。
3. **智能体服务交互层（实现“助手”）**
   * **职责**：作为用户与“知识大脑”交互的窗口，提供主动、智能的服务。
   * **技术**：
     + **智能查询助手**：支持自然语言查询（如：“帮我调出3#主变的所有接线图和上次的修改单”）。
     + **智能工作流**：将高风险的“图纸修改”流程自动化、智能化。

## 3. 系统功能架构

融合基础功能和智能功能，系统功能架构设计如下：

变电站智能数字图书馆  
├── 1. 基础支撑平台 (系统管理)  
│ ├── 1.1. 部门管理  
│ ├── 1.2. 用户管理  
│ ├── 1.3. 角色与权限管理  
│ ├── 1.4. 菜单管理  
│ └── 1.5. 系统日志管理  
│  
├── 2. 核心业务模块 (基础功能强化)  
│ ├── 2.1. 首页仪表盘 (强化)  
│ │ ├── 按变电站统计  
│ │ ├── 按文件类型统计  
│ │ ├── 按文件大小统计  
│ │ ├── 近7天上传趋势  
│ │ └── (新增) 智能告警 (如：版本不一致告警、待处理工单)  
│ │  
│ ├── 2.2. 厂站管理  
│ │ ├── 厂站台账维护 (增删改查)  
│ │ ├── 厂站信息导入/导出  
│ │ └── 厂站关联用户 (数据隔离)  
│ │  
│ └── 2.3. 图纸档案管理 (核心强化)  
│ ├── 基础操作：上传、批量下载、重命名、删除  
│ ├── 版本管理：版本历史追溯、新旧版本对比  
│ ├── 在线预览  
│ └── 智能关联视图：预览图纸时，侧边栏显示KG关联的"设备"、"台账"、"修改单"  
│  
└── 3. 智能引擎模块 (智能升级)  
 ├── 3.1. AI内容理解 (后台任务)  
 │ ├── 图纸OCR/CV服务 (含拓扑重建)  
 │ └── 文档NLP实体提取服务  
 │  
 ├── 3.2. 知识图谱中枢  
 │ ├── 变电站知识图谱构建  
 │ └── 关联推理查询接口  
 │  
 └── 3.3. 智能体服务 (Agent)  
 ├── 智能查询助手 (支持自然语言)  
 └── 智能工作流引擎 (如：图纸修改)

## 4. 核心模块详细设计

### 4.1. 图纸档案管理（智能化升级）

本模块是融合设计的核心。

#### 4.1.1. 智能上传处理流程

用户执行“文件上传”时，触发后台智能管线（**FastAPI + Celery**）：

1. **API接收**：FastAPI 接收文件，保存至存储服务器，并立即返回“上传成功，正在处理”。
2. **任务派发**：FastAPI 将文件信息（如路径、ID）发送到 **Celery** 任务队列（如使用Redis作为Broker）。
3. **后台AI处理**：Celery Worker（独立的Python进程）消费任务，执行耗时的AI分析：
   * 如为图纸，调用 **AI内容理解引擎** 执行OCR/CV，提取图框、符号、拓扑。
   * 如为文档，调用引擎执行NLP，提取关键实体。
4. **注入KG**：AI引擎将结构化结果发送至 **知识图谱中枢**，自动创建/更新图谱中的节点和关系。
5. **状态更新**：处理完成后，更新数据库中该文件的状态为“已处理”。

#### 4.1.2. 检索方式（三合一）

提供三种检索方式，满足不同用户需求：

1. **基础检索**：按关键字（厂站名称、文件名称、文件类型、上传时间）进行检索 (SQL查询)。
2. **全文检索**：基于AI提取的图纸和文档内容，进行全文内容搜索 (Elasticsearch 或 数据库全文索引)。
3. **智能查询**：由 **智能体(Agent)** 提供自然语言查询服务。
   * *用户输入*：“查找110kV-A母线的所有相关图纸和设备台账”
   * *Agent处理*：Agent将自然语言转为KG查询语句（Cypher），从知识图谱中精确抓取，并返回结果。

#### 4.1.3. 智能关联视图（解决“图-物-档”脱节）

在用户点击“预览”时，界面升级为“智能关联视图”：

* **左侧（70%）**：显示图纸或文档的预览内容。
* **右侧（30%）**：显示 **知识图谱中枢** 实时返回的关联信息：
  + **关联设备**：[列表] 本图纸上绘制的所有设备。
  + **关联档案**：[列表] 关联的设备台账、说明书。
  + **修改历史**：[列表] 所有影响了本图纸/设备的“历史修改单”。
  + **引用关系**：[列表] 本图纸引用了哪些图纸，或被哪些图纸引用。

### 4.2. 智能工作流：图纸修改（解决“版本混乱”）

这是解决“版本混乱”的核心场景。

1. **发起修改**：工程师在系统中发起对某“权威版本”图纸的修改请求。
2. **智能风控（Agent + KG）**：
   * **Agent** 立即向 **KG** 查询：“这张图纸关联了哪些设备？引用了哪些其他图纸？”
   * **Agent** 自动提醒工程师：“警告：本次修改可能影响 5 台设备和 3 张关联图纸 [点击查看列表]”。
3. **协同审批**：Agent根据KG中定义的规则（如“110kV等级修改需A、B岗审批”），自动推送工单。
4. **自动归档与分发（Agent）**：审批通过后，Agent自动执行：
   * 将旧图纸标记为“历史版本”并归档。
   * 将新图纸上架，标记为“权威版本”。
   * 自动更新KG中所有相关的节点和关系。
   * 主动向相关运维班组推送“图纸变更通知”。

## 5. 技术架构方案 (全Python栈)

采用前后端分离架构，后端和AI引擎统一使用Python技术栈。

* **前端**：**Vue.js 3** ( + Vite + Pinia + Element Plus / Tailwind CSS)
  + 负责所有用户界面和交互。
* **后端**：**FastAPI** (Python)
  + 提供高性能的RESTful API，负责用户认证、权限、CRUD、文件上传等所有同步请求。
  + 使用Pydantic进行数据校验。
  + 使用SQLAlchemy 作为ORM。
* **异步任务与AI管线**：**Celery** (Python)
  + 负责处理所有耗时的后台任务，特别是“智能上传处理流程”（OCR, NLP, KG注入）。
* **任务队列/缓存**：**Redis**
  + 作为Celery的Broker和Backend。
  + 同时用于热点数据缓存和用户Session管理。
* **业务数据库**：**SQLite** (开发) / **MySQL | PostgreSQL** (生产)
  + 存储系统管理、厂站台账、文件索引、用户信息等结构化业务数据。
* **知识图谱数据库**：**Neo4j** (或 JanusGraph)
  + 存储“图-物-档”的知识图谱数据，负责关联推理查询。
* **文件存储**：**本地磁盘** (开发) / **MinIO** (生产)
  + S3兼容的对象存储服务，用于存放所有上传的图纸和档案原文件。

## 6. 部署架构 (全Python栈)

采用内网部署，服务容器化（使用Docker Compose或K8s）。

* **NGINX服务器 (1台)**：
  + 作为反向代理，对外统一暴露端口。
  + 负责前端静态文件服务和后端API的请求转发。
* **应用与AI服务器 (1-N台, 需GPU)**：
  + **FastAPI应用 (容器)**：运行后端API服务。
  + **Celery Worker (容器)**：运行异步AI任务。**此容器必须能访问GPU资源**以执行高效的OCR/CV模型推理。
  + **AI模型**：(如 PaddleOCR, Transformers)
* **数据库服务器 (1台)**：
  + **MySQL (容器)**：业务数据库。
  + **Neo4j (容器)**：图数据库。
* **缓存与存储服务器 (1台)**：
  + **Redis (容器)**：任务队列与缓存。
  + **MinIO (容器)**：文件对象存储。

## 7. 实施建议（分阶段）

建议分三阶段实施，平滑过渡，控制风险。

* **第一阶段：基础平台建设 (对齐PDF)**
  + **目标**：快速上线一个可用的文件管理系统。
  + **内容**：实现PDF方案中的所有基础功能（系统管理、厂站管理、基础的图纸CRUD和统计）。使用FastAPI + Vue + SQLite + 本地存储。
  + **产出**：**(已完成)** 《变电站智能数字图书馆V1.0》。
* **第二阶段：智能引擎构建 (AI+KG后台占位符)**
  + **目标**：构建“读懂”数据的能力和“知识大脑”。
  + **内容**：
    1. **(下一步)** 集成Celery，将文件上传改为异步处理。
    2. **开发占位符**：创建 ocr\_task.py, nlp\_task.py 等Celery任务，初期只打印日志 (print("TODO"))。
    3. 部署Neo4j，定义变电站知识图谱本体。
    4. 开发KG注入任务（kg\_task.py）的占位符。
  + **产出**：平台具备智能数据处理管线，为AI赋能做好准备。
* **第三阶段：智能应用上线 (Agent服务)**
  + **目标**：交付智能化核心价值，解决业务痛点。
  + **内容**：
    1. **填充占位符**：使用真实AI模型（如PaddleOCR）替换 ocr\_task.py 中的 print("TODO")。
    2. **上线应用**：开发智能体服务（Agent）、“智能查询助手”（自然语言搜索）和“智能关联视图”。
    3. **上线工作流**：“图纸修改智能工作流”。
  + **产出**：《智能化变电站智能数字图书馆V2.0》，实现从“档案库”到“知识资产”的飞跃。