项目复盘

一、汇报大纲

1. 项目背景简介（1页）： 公司为什么做这个项目，项目目标，团队组成，我的角色。
2. 整体流程概述（1页）： 一幅流程图，展示从“原始文档”->“收集”->“清洗解析”->“切片处理”->“导入Dify”->“API服务”->“前端应用”的全过程。
3. 核心挑战与解决方案（5-8页，重点）：

- 挑战一：数据质量参差不齐
  - 解决方案： 数据清洗标准化（展示你总结的《清洗规则》，附1-2个正则表达式例子）。
- 挑战二：多格式文档解析困难
  - 解决方案： 解析工具选型（列个表对比PyMuPDF, pdfplumber, OCR等），附一段简单的Python代码示例（如用pdfplumber提取文本）。
- 挑战三：文本切片影响检索精度
  - 解决方案： 详述你的切片策略演进过程（固定长度->重叠->按结构分割）。画一个图对比不同chunk_size的效果。强调“内容完整性优先于长度一致性”的原则。

4. 成果与量化价值（1-2页）：
   - 接入了X个产品线，Y份文档，形成了ZGB的知识库。
   - 效果提升： 经过优化后，问答准确率/召回率提升了大概多少（可以说“显著提升”，或者给个范围如30%-50%）。
   - 沉淀了一套《SOP》和自动化脚本，提升了后续接入效率。
5. 反思与未来展望（1页）：
   - 反思：早期对数据质量重视不够，走了弯路；手动处理占比仍较高。
   - 展望：未来希望能引入更智能的解析和切片模型，实现全自动化管道

二、时间线

第1个月：环境搭建与技术选型（摸索与规划阶段）

工作内容：

与研究团队一起调研不同的LLM应用开发平台（如LangChain、Dify、FastGPT等）。最终选择Dify，理由是开源、可私有化部署、图形化界面降低开发门槛。完成Dify平台的本地化或云服务器部署（Docker Compose部署），配置基础环境（网络、存储、访问权限等）。

产出物：

《LLM平台选型报告》、《Dify平台部署与配置文档》。

“数据”相关经验：

这个阶段主要是概念性认知。你会意识到原始知识文档（Word、PDF、PPT）直接上传后，检索效果很差，从而认识到“数据清洗和解析”是影响最终智能体效果的最关键前置环节。

第2-3个月：流程跑通与MVP（最小可行产品）构建（核心开发阶段）

工作内容：

这是你最核心的工作阶段。与产品部门同事紧密合作，选取1-2个核心产品线的手册、文档进行试点。

数据收集：

从Confluence、GitRepo、共享盘等地方收集第一批原始文档。

数据清洗与解析探索：

这是你需要重点总结的部分。你会遇到大量问题，并尝试各种工具和方法去解决（具体方法见第二部分）。

Dify工作流搭建：

在Dify中创建知识库，上传处理后的文本，配置检索策略（相似度/关键字）、测试问答效果。

API封装与联调：

与后端同事合作，将Dify提供的API进行二次封装（例如增加认证、日志、限流），供前端调用。完成一个简单的Web页面进行测试。

产出物：

第一批清洗后的标准格式文档、初步的《知识文档处理规范》、《Dify知识库配置指南》、API接口文档。

“数据”相关经验：

大量实践。你会积累大量针对你们公司特定文档格式的清洗和解析经验，形成一套初步但有效的方法论。

第4个月：瓶颈与优化（问题解决阶段）

工作内容：

MVP上线后，收集用户反馈。主要问题会集中在“搜不到”、“答案不准确”、“段落截断奇怪”。这个阶段你主要进行效果优化。

1. 归因分析： 发现效果不好八成是数据问题。你会回头更深入地研究文本切片（Chunking）策略。调整 chunk size（块大小）和 chunk overlap（块重叠），尝试不同的文本分割器（按段落、按标题、按句子）。
2. 数据清洗加强： 发现某些PDF解析效果极差（特别是扫描版），开始引入OCR工具（如PaddleOCR）进行文本提取。
3. 元数据增强： 开始在切片时给每个文本块添加元数据（如：来源产品、文档标题、章节号），以便检索后能精确定位来源。

产出物：

《项目初期问题总结与优化方案》、更新版的《知识文档处理规范》。

“数据”相关经验：

从“能用”到“好用”的进阶，理解了切片策略对检索精度的影响，掌握了处理复杂格式文档的技巧。

第5-6个月：扩展与沉淀（推广与复盘阶段）

工作内容：

将成功经验复制到其他产品线。流程标准化，可能会编写一些自动化脚本（如用Python写一个自动处理文件夹内所有文档的脚本，集成格式化、清洗、切片步骤）。编写操作手册，培训产品部门的同事如何提交符合规范的文档。

产出物：

多个产品的知识库、自动化处理脚本、最终的《企业知识库文档接入标准操作流程（SOP）》、项目总结报告。

“数据”相关经验：

流程化、自动化。你将零散的经验固化为可重复执行的流程和工具，这是价值的升华。

三、涉及到的文档

3.1 铝加工生产管理平台

1. 产品设计与定义文档

- 产品需求文档（PRD）：描述项目的整体目标、用户画像、功能列表（如点位监测、报警配置等）；
- 功能规格说明书（FSD）：比PRD更技术化，详细定义每个功能的输入、输出、处理逻辑和界面元素；
- 系统架构设计文档：描述技术选型、微服务划分、数据流图等；
- 墨刀原型（不完整，无法录入知识库）；

2. 核心业务规则文档
   - 节能策略算法说明：
     - 《数据机房温度场调控策略》：描述如何根据多个温度点的读数，动态调整空调风量、风速、温度设定值。例如，“当A区温度>26°C，B区温度<22°C时，执行XXXX风道调整”。
     - 《中央空调（VRV）群控策略》：针对办公建筑，如何根据上下班时间、区域人流量、室外温度等因素，自动调节不同分区的空调模式与温度。
     - 《节能效果计算模型文档》：如何计算和评估节能量（如同比、环比算法）。
   - 设备点位表：同样非常重要。
     - 数据机房：每个温度传感器的点位信息、空调设备的控制点位（可读写）。
     - 空调系统：各房间温控器点位、主机运行状态点位等。
3. 行业与咨询文档
   - 能源审计报告：你提到的这个非常关键。通常包含：
   - 企业能源消耗结构分析（电、水、气）。
   - 主要用能设备清单及能耗分析。
   - 存在的问题及节能潜力分析。
   - 节能措施建议。这些报告是制定节能策略的重要输入。
4. 技术与实施文档
   - 机房风道设计图纸（可能为PDF版CAD图）：图纸中的设计说明文字是重要知识。
   - 设备选型手册：所采用的空调、传感器等设备的官方技术手册。
   - API文档、部署手册、测试报告等。
5. 项目与客户文档
   - 招投标文件：技术方案书、投标应答书。
   - 客户交付文档：用户手册、培训材料。
   - 项目案例报告：为吸引客户而制作的成功案例总结，如《XX银行数据中心节能项目案例》，内含具体节能数据。

3.2 节能管理平台

1. 产品设计与定义文档
    - PRD、FSD、架构图、设计稿等。
2. 核心业务规则文档
    - 节能策略算法说明：
    - 《数据机房温度场调控策略》：描述如何根据多个温度点的读数，动态调整空调风量、风速、温度设定值。例如，“当A区温度>26°C，B区温度<22°C时，执行XXXX风道调整”。
    - 《中央空调（VRV）群控策略》：针对办公建筑，如何根据上下班时间、区域人流量、室外温度等因素，自动调节不同分区的空调模式与温度。
    - 《节能效果计算模型文档》：如何计算和评估节能量（如同比、环比算法）。
    - 设备点位表：同样非常重要。
    - 数据机房：每个温度传感器的点位信息、空调设备的控制点位（可读写）。
    - 空调系统：各房间温控器点位、主机运行状态点位等。
3. 行业与咨询文档（这部分很有特色）
    - 能源审计报告：你提到的这个非常关键。通常包含：
    - 企业能源消耗结构分析（电、水、气）。
    - 主要用能设备清单及能耗分析。
    - 存在的问题及节能潜力分析。
    - 节能措施建议。这些报告是制定节能策略的重要输入。
    - 行业技术白皮书/研究报告：公司收集的关于机房PUE值优化、中央空调节能技术路径等的研究文档。
4. 技术与实施文档
    - 机房风道设计图纸（可能为PDF版CAD图）：图纸中的设计说明文字是重要知识。
    - 设备选型手册：所采用的空调、传感器等设备的官方技术手册。
    - API文档、部署手册、测试报告等。
5. 项目与客户文档
    - 招投标文件：技术方案书、投标应答书。
    - 客户交付文档：用户手册、培训材料。
    - 项目案例报告：为吸引客户而制作的成功案例总结，如《XX银行数据中心节能项目案例》，内含具体节能数据。

四、积累和展示的数据清洗、解析、切片经验（核心内容）

重点，问题和解决方案

数据清洗（Data Cleaning）

经验1：格式标准化

问题：

来源文档编码不统一（UTF-8/GBK）、换行符混乱（CR/LF/CRLF）。

方法：

统一转换为UTF-8编码，Unix换行符（\n）。

工具：

Python (codecs库)、dos2unix命令、Notepad++。

经验2：无用信息剔除

问题：

文档包含页眉、页脚、页码、免责声明、网址等无关信息。

方法：

编写正则表达式（Regex） 进行匹配和剔除。这是你最需要展示的具体技能。

举例：

^第\d+页$匹配页码，(版权所有|Copyright|©|confidential)匹配版权信息。

经验3：特殊字符处理

问题：

文档中存在乱码、不可见字符、多余的空格和制表符。

方法

使用字符串处理函数（如Python的str.strip(), str.replace()）清理，或用正则\s+匹配多余空白字符。

数据解析与切片（Data Parsing & Chunking）

经验4：格式解析（Parsing）

问题：

不同格式（PDF/Word/PPT/Excel）需要不同解析器，解析后格式丢失。

方法：

- Word/PPT：
  使用python-docx、pptx库，能较好保留标题、段落结构。
- PDF：这是重灾区。
  - 文本型PDF： 使用PyMuPDF（fitz）、pdfplumber，效果较好。
  - 扫描型PDF/图片： 使用OCR技术，如PaddleOCR、Tesseract，你需要谈谈调参经验（如识别语言包、清晰度预处理）。

工具：

Python各类解析库、PaddleOCR。

经验5：文本切片（Chunking） - 最体现深度的地方

问题：

直接按固定长度（如512字）切割，会把一个完整的操作步骤或概念切到两个块里，导致检索信息不完整。

方法：

- 初级： 使用递归字符文本分割器，尝试不同的chunk_size（e.g., 200-1000）和 chunk_overlap（e.g., 50-100）。这是Dify等平台的默认方法。
- 高级： 使用语义分割器。利用NLP模型识别段落、标题（# ## ###）、句子边界进行切割。你可以说你探索过这种方法，但由于计算资源和小公司场景，最终选择了基于规则的分割，这非常真实。
- 自定义规则： 针对公司文档特点定制规则。例如，你们的文档都用 “步骤1：” 开头，你就可以按这个模式来分割，保证每个操作步骤的完整性。

结论：

强调没有银弹，最终采用的策略是：优先按标题结构分割，其次按段落，最后再按固定长度兜底，并设置了重叠字符以确保上下文连贯。

经验6：元数据（Metadata）附加

方法： 为每个文本块附加来源、产品线、文档标题、更新时间等信息。这极大地帮助了后续的检索排序和答案溯源。

工具： 在Dify或代码中手动配置。