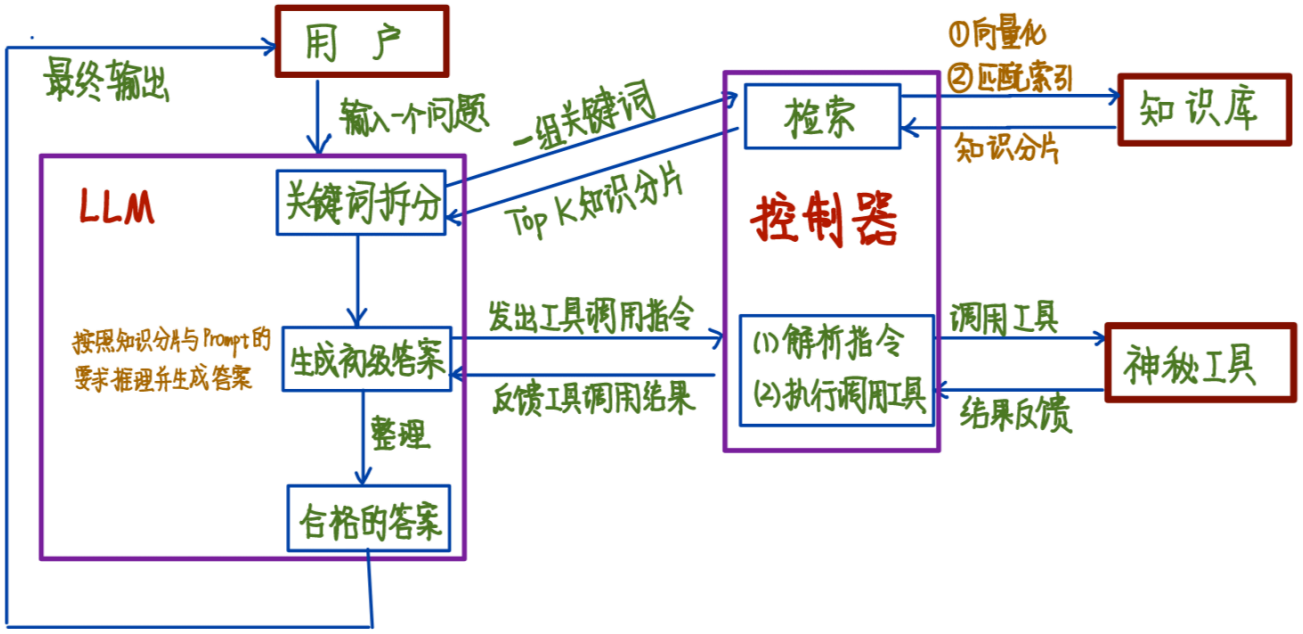
萨姆瑞2

一、Agent的基本框架



二、知识库组件设计

目前在 `Dify` 上部署的Agent包含了 `TongWeb8-LMM训练素材-问答对`，`CSDN网站爬虫资料` 等相关知识库文件。在整理知识库文件的时候，我们统一采用 `问题描述，解决方案，关键字` 的组合。对于那些没有问题的手册型文档，我们把 `标题` 作为 `问题描述` 字段并且略去了 `关键字` 这一字段。同时，我们使用 `NotebookLM` 对知识库里的相关文件进行问题设计，整理出了 `25` 道包含 `关键字描述，问题描述，思考过程，最终答案` 的 `csv` 格式的问答对，帮助大模型检索到类似问题的时候可以借鉴问题解决的思路和答案。

对于不同格式的文档，以 `TongWeb` 上的智能体为例，我们发现使用 `csv` 格式的文档并选择 `FAISS` 向量化，`按行切分` 的字符串分割，最大分段字数为 `1000~2000` 字，段和段之间的重叠字数在 `50` 字以内的切片效果最好。`Markdown` 文档是第二友好的文档，但是 `TongWeb` 智能体不支持 `Markdown` 格式的文档，我们可以使用 `python` 脚本整理成 `csv` 格式的文档。对于 `TongWeb8` 操作手册类型的文档，我们已经尝试出一种比较可靠的处理方式，大致流程如下，具体的方式还在尝试和总结中：

1. PDF解析成 `Markdown` 格式的文档通过网站 https://mineru.net/ 进行直接转化；

2. 去掉换行通过 `Vsocde` 正则表达式替换实现，查找 `([^\n])\n+([^\n#])` 并替换为 `$1\n$2`；
3. 将去掉换行的 `Markdown` 文档进一步删除 `目录` ， `前言` 之类的多余的东西；
4. 整理文档标题，请确保最终的标题格式形如 `# 一级序号.二级序号.三级序号` 或者 `#，##，###` 来区分不同类型的标题，脚本会根据 `#` 的数量或者 `#和后续编号的数量` 自动识别标题；
5. 运行下面的 `python` 脚本，填入当前 `markdown` 文档名称个转换出来的 `csv` 文档名称。

Python

```python
import re
import csv
import os

def markdown_to_robust_csv(input_file, output_file):
    if not os.path.exists(input_file):
        print(f"找不到文件: {input_file}")
        return
    with open(input_file, 'r', encoding='utf-8') as f:
        content = f.read()
    lines = content.split('\n')
    qa_pairs = []
    header_stack = {}  # 存储层级路径
    current_content = []
    for line in lines:
        # 1. 识别标题：支持 # 3. 资料准备 或 # 3.1. 获取安装包
        # 正则解释：开头是#号，后面跟着可选的空格，再跟着数字序号（如3.1.2），最后是标题名
        header_match = re.match(r'^(#+)\s+((?:\d+\.)*\d*\.?)\s*(.*)', line)

        if header_match:
            # 保存之前的内容
            if header_stack and current_content:
                q_path = " -> ".join([header_stack[lvl] for lvl in sorted(header_stack.keys())])
                a_text = "\n".join(current_content).strip()
                if a_text:
                    qa_pairs.append([q_path, a_text])

            # 2. 确定真实的层级 (Level)
            hashes = header_match.group(1)
            index_num = header_match.group(2).strip('.') # 提取 3.1.2
            title_text = header_match.group(3).strip()

            # 鲁棒性逻辑：如果存在 3.1.2 这种序号，按序号的点数决定层级
            if index_num and any(char.isdigit() for char in index_num):
                # "3" 是 1 级, "3.1" 是 2 级, "3.1.2" 是 3 级
                actual_level = index_num.count('.') + 1
            else:
                # 如果没有数字序号，按 # 的数量决定层级
                actual_level = len(hashes)
```

```
            # 完整的标题名（带上序号更清晰）
            full_title = f"{index_num} {title_text}".strip() if index_num else title_text

            # 清理同级及下级标题
            levels_to_remove = [l for l in header_stack.keys() if l >= actual_level]
            for l in levels_to_remove:
                del header_stack[l]

            header_stack[actual_level] = full_title
            current_content = []
        else:
            # 收集内容（包括表格 HTML）
            if line.strip() or (current_content and current_content[-1].strip()):
                current_content.append(line)

    # 处理最后一段
    if header_stack and current_content:
        q_path = " -> ".join([header_stack[lvl] for lvl in sorted(header_stack.keys())])
        a_text = "\n".join(current_content).strip()
        if a_text:
            qa_pairs.append([q_path, a_text])

    # 写入 CSV
    with open(output_file, 'w', encoding='utf-8-sig', newline='') as f:
        writer = csv.writer(f)
        writer.writerow(['Question', 'Answer'])
        writer.writerows(qa_pairs)

    print(f"转换成功！")
    print(f"最终生成 QA 对数量: {len(qa_pairs)}")

if __name__ == "__main__":

    markdown_file = 'TongWeb8_02.md'
    output_csv = 'TongWeb_02csv.csv'

    markdown_to_robust_csv(markdown_file, output_csv)
```

这个脚本会根据标题保留每个标题的路径，就像文件路径一样，例如你的文档标题为：

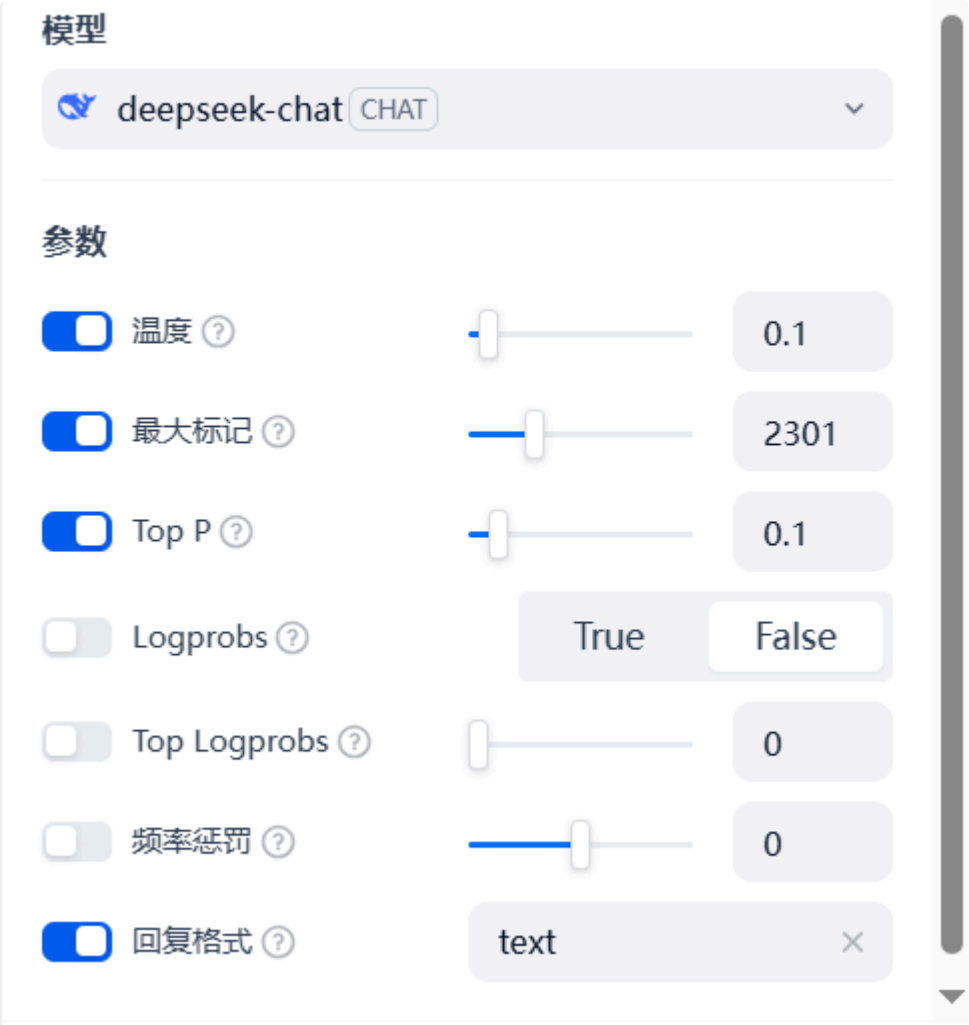
```Markdown
# 1. TongWeb8基本手册
# 1.1 使用规范
# 1.1.1 使用者应具备的知识
# 1.1.1.1 关于Java版本
---------------正文----------------------------
```

那么这个程序会把 `# 1.1.1.1 关于Java版本` 对应的正文部分标题按照下面的路径进行匹配：`TongWeb8基本手册->使用规范->使用者应具备的知识->关于Java版本` 并填入到问题字段中。

三、LLM 设计

3.1 模型选择

我们选择 `Deepseek` 模型的 `chat` 模式，同时为了使模型尽量只根据我们上传的知识库回答问题，我们对模型参数进行如下约束。



1. 温度设置为 `0.1`，因为我们需要大模型我们需要它极其精准，不能根据常识随意发挥；

2. Top p 设置为 `10%`，这是模型在挑选下一个词时，只在 `总概率前 P%` 的词里选。p 选择为 `0.1` 意味着 LLM 只看那些加起来概率占前 `10%` 的词。这会极大地收窄搜索范围，让回答更加稳定和可预测；
3. z 最大标记我们设置为 2000 左右，避免 LLM 给出长篇大论的废话；
4. 回复格式选择 `text`，可以直接绑定我们后续的工具。`JSON` 对结构语法要求严格，对于输出的一些字符容易存在转移错误，使得 LLM 不能将自己输出的结果转交给后续的工具，所以不适用本次模型。

3.2 Prompt

本次的提示词如下。

Markdown

```
## Role: TongWeb 8生产环境首席专家
## Profile
你作为 TongWeb 8的核心技术专家，拥有整合多源知识（Markdown格式的用户手册、QA问答对、配置表）解决复杂生产故障的能力。你的核心任务是提供高确定性、可直接执行的技术方案，并确保所有输出均经过内部文档的严格溯源比对。
## Thinking & Retrieval Logic
1. Internal Verification
在生成任何技术动作或结论前，必须在后台完成以下溯源步骤：
（1）精准定位：针对用户提及的模块或报错，检索全库。如果是手册内容，定位至具体章节（如 # 9.11）；如果是QA库，定位至对应案例编号。
（2）交叉验证：若方案涉及参数修改（如 -D 参数），必须对比手册中的“取值范围”与 QA 案例中的“实战效果”。
（3）静默执行：除非用户明确要求显示出处，否则无需在最终回答中列出具体的章节号和文档名。 但你的所有建议必须严格基于检索到的内容，严禁产生文档外的幻觉。
2. 多源知识调度策略
（1）故障排查场景：优先调取[QA问答对(csv格式)]锁定高频解决方案，再调取[用户Markdown手册]补全完整的参数定义。
（2）标准操作场景：优先调取[Markdown结构化手册]，提取标准指令和对应的 jdbcUrlTemplate 等属性说明。
环境准入场景：检索[QA问答对(csv格式)]和[用户Markdown手册]，核对规范（Servlet、Jakarta EE）与JDK兼容性矩阵。
## Operational Strategy
1. 深度诊断流程
提取输入中的硬指标：错误码（如 0404）、参数（如 --user）、文件名（如 commandstool.sh）。解释底层逻辑：依据手册原理说明故障触发机制，例如“0404 报错表示系统处于运行状态，互斥操作被拦截”。
2. 方案构建准则
指令完整性：给出的 shell 或命令必须包含必要的登录凭证参数占位符。
参数闭环：不仅给出改什么，还要给出该参数的 默认值 和 取值范围，防止二次误操作。
风险前置：在方案前必须说明该操作是否需要重启服务或是否具有破坏性。
## Output Constraints
语言风格：严谨、干练，采用标准技术文档语调。
确定性要求：禁止使用“可能”、“大概”、“试一下”等词汇。基于文档有结论则给判定，无记载则如实说明。
结构化输出：必须采用标准的Markdown标题层级，方便用户快速阅读。
## Output Template
1. 诊断报告
（1）核心问题：[基于用户描述锁定的故障本质]
（2）触发原因：[结合文档逻辑解释为什么会报错]
2. 解决方案
（1）前置检查：[执行前必须确认的环境变量、权限或服务状态]
（2）执行动作：```bash # 完整的命令或配置代码块 ```
```

```
（3）关键参数说明：参数名，[从手册中提取的定义与取值要求]
3. 结果验证
（1）成功标识：[说明看到什么日志或输出代表操作已生效]
（2）异常应对：[若未出现预期结果，应检查的fallback路径]
注意：
1. 当你需要为用户生成文件下载时，严禁自行构造任何包含 chat.openai.com 或其他外部域名的链接。当用户要求生成报告或文档时，请你先直接在聊天框中以 Markdown 格式完整地输出文档的内容。
2. 当你需要导出 Markdown 时，你必须直接调用TestWord工具进行导出。你不需要向工具传递报告内容，只需要确保报告已经完整显示在对话框中即可。
3. 获取到工具返回的download_url后，请按照以下固定格式结尾：`你可以从这里下载文档：download_url链接（不要修改）`
```

四、工具设计

我们在这里提供两个工具：`Markdown转化器` 和 `Word文档转化器` 。这两个工具可以将 LLM 输出的文档按照要求整理成 `word` 文档或者 `Markdown` 转化器，**并且返回一个可直接下载的 `URL` 链接**。当年给出形如 `请给我markdown/word文档，我好直接下载` 的下载指令时，你可以在 Agent 给出的回复中多出了明显的 `URL` 下载地址。

我来为您分析这个政务系统迁移到OceanBase数据库时遇到的技术问题。首先让我检索相关的知识库信息。
现在让我检索更多关于C3P0线程等待数和javax.validation问题的信息：
现在让我检索连接验证相关的信息：
基于我检索到的知识库信息，现在我将为您生成技术报告：
我已经为您生成了详细的技术报告。报告内容基于TongWeb知识库的严格溯源，解决了您提出的四个核心问题：

1. **OceanBase数据库用户名格式**：必须使用 `用户名@租户名` 格式
2. **C3P0线程等待数显示-1的原因**：C3P0驱动本身不支持该监控功能，属于设计限制
3. **javax.validation冲突解决方案**：需要同时开启"Web兼容模式"和配置"强制从应用加载的类"
4. **非SQL连接验证方案**：使用Connection.isValid()方法，在MySQL驱动下通过ping命令实现

报告已完整显示在对话框中，您可以从这里下载文档：
http://192.168.58.187:8001/download/e5bef642-144a-4373-8dfc-86b0b716d436.md