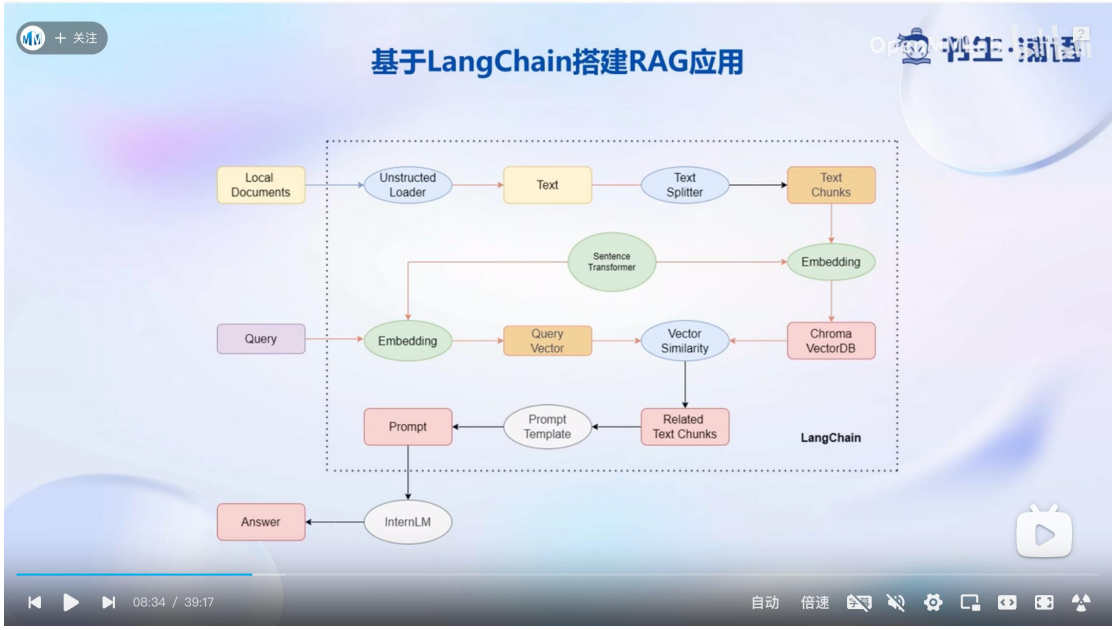
(3)基于 InternLM 和 LangChain 搭建你的知识库

基础作业：

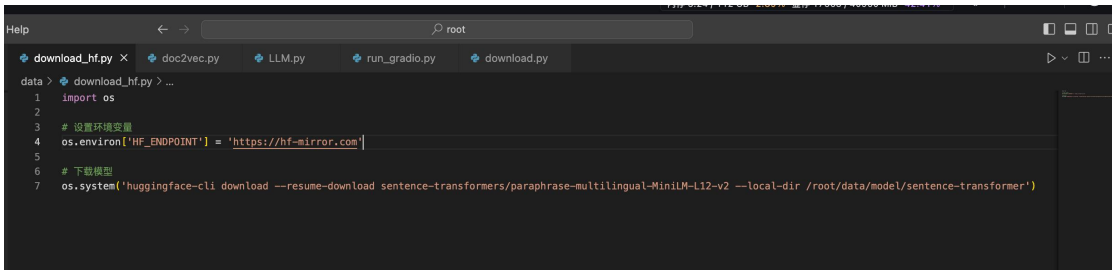
复现课程知识库助手搭建过程（截图）

首先我们回顾一下 RAG 的原理：
RAG 是将本地文档都转化为文本形式后，使用 Embedding 将文本存储进向量数据库。在用户输入 prompt 以后，将用户的 prompt 也转化成 embedding，和向量数据库里的 embedding 进行匹配，然后将用户的 prompt 和文本的 prompt 一起输入大模型，得到模型的回答。RAG 可以低成本地实时更新大模型知识，但是因为需要输入大量文本作为外部知识来源，那么单次提问可以包括的信息来源数量就比较少，并且 RAG 的能力容易受基座模型能力的影响。



1. 环境配置

其中 Internlm 模型部署和模型下载和上节课内容有重合，在这里就不截图了。
这里贴一下 1.3 LangChain 相关环境配置的实现过程。
配置 langchain 用到了开源词向量模型 Sentence Transformer，这里需要安装一下。



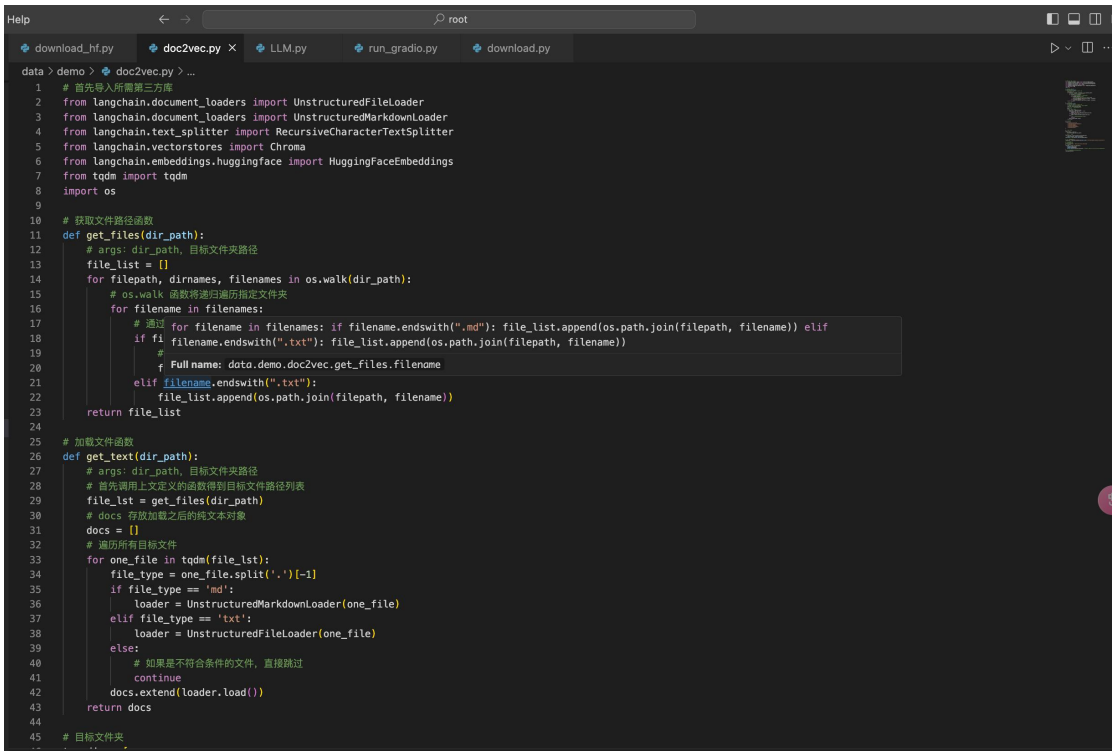
2. 知识库搭建

本小节分为 3 个子步骤：

a. 将文件夹中格式为 txt 或 markdown 格式的文件路径找出来

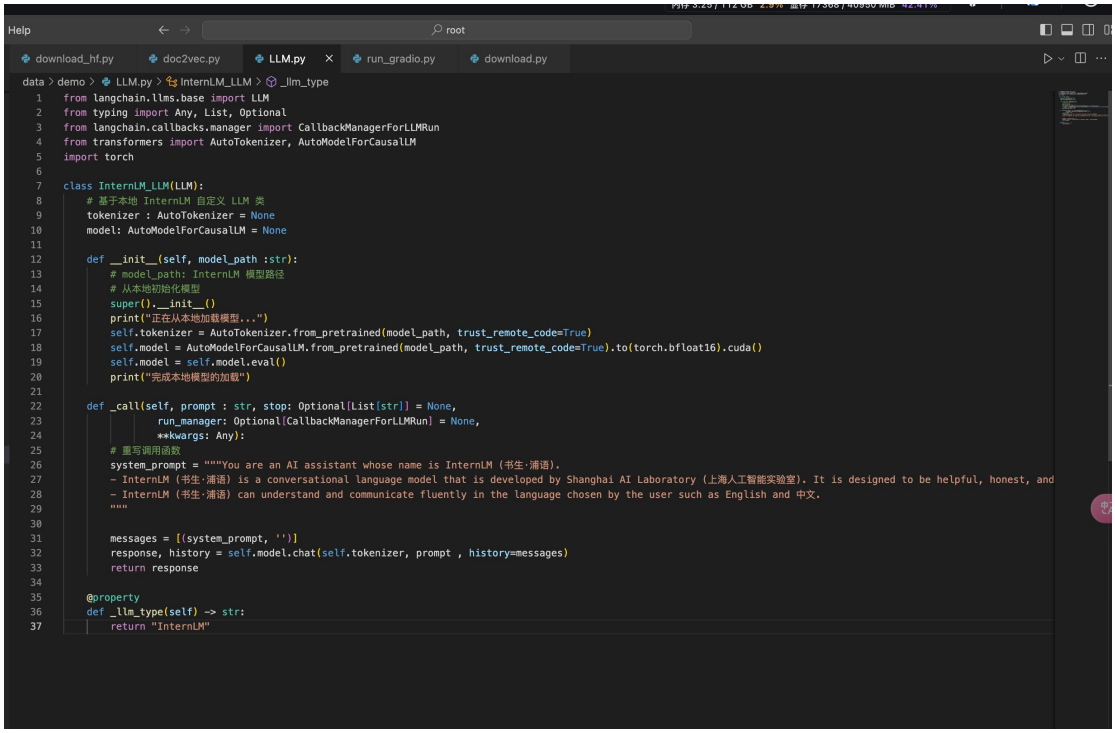
b. 使用 LangChain 提供的 FileLoader 对象来加载目标文件，得到由目标文件解析出的纯文本内容。

c. 由纯文本对象构建向量数据库，先对文本进行分块，接着使用开源词向量模型 Sentence Transformer 对文本块进行向量化。选择 Chroma 作为向量数据库，基于上文分块后的文档以及加载的开源向量化模型，将语料加载到指定路径下的向量数据库



3. InternLM 接入 LangChain

为便捷构建 LLM 应用，我们需要基于本地部署的 InternLM，继承 LangChain 的 LLM 类自定义一个 InternLM LLM 子类，从而实现将 InternLM 接入到 LangChain 框架中。实现的过程从 LangChain.llms.base.LLM 类继承一个子类，并重写构造函数与 _call 函数即可



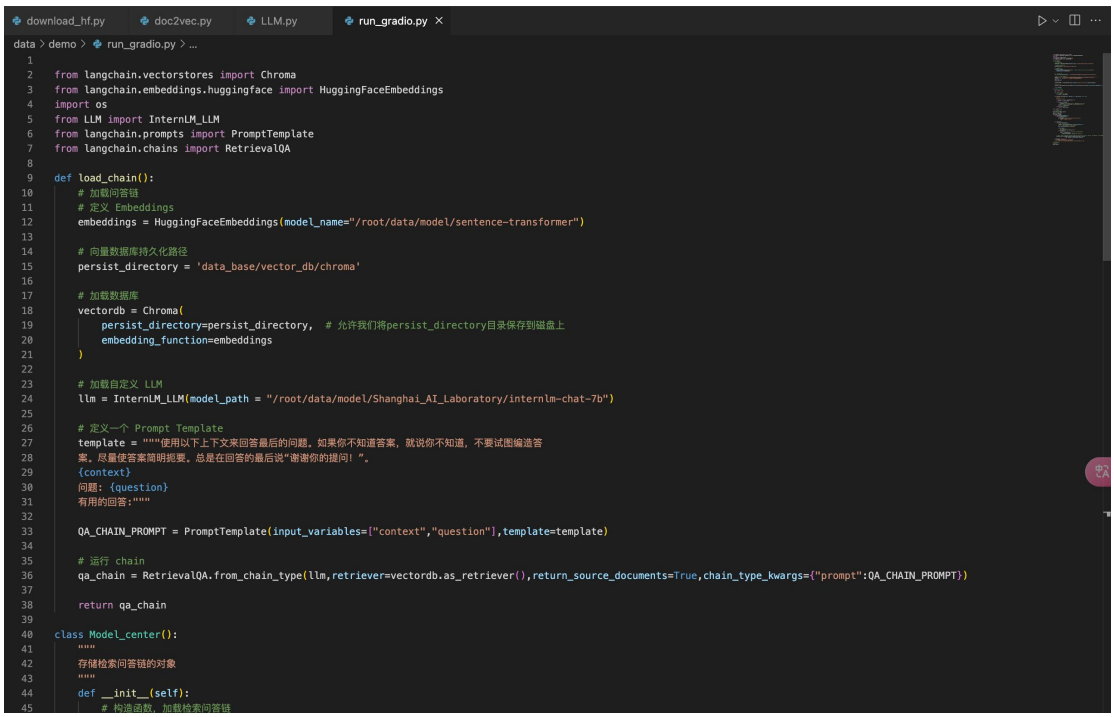
4. 构建检索问答链

本小节分为以下几个步骤:

a.通过 Chroma 以及上文定义的词向量模型将上文构建的向量数据库导入进来

b.构建 Prompt Template，该 Template 其实基于一个带变量的字符串，在检索之后，LangChain 会将检索到的相关文档片段填入到 Template 的变量中，从而实现带知识的 Prompt 构建。

c.调用 LangChain 提供的检索问答链构造函数 RetrievalQA.from_chain_type，基于我们的自定义 LLM、Prompt Template 和向量知识库来构建一个基于 InternLM 的检索问答链



5.demo 结果展示

可以看到，demo 已经部署成功，可以成功运行。

进阶作业：

选择一个垂直领域，收集该领域的专业资料构建专业知识库，并搭建专业问答助手，并在 OpenXLab 上成功部署（截图，并提供应用地址）

我选择了厨房菜谱领域搭建问答助手，代码在
https://github.com/leonfrank/InternLM-HW/tree/main
项目下面
应用地址在
https://openxlab.org.cn/apps/detail/chat-gpt/llm_cuisine
因为没有 GPU 资源，应用没有启动成功。待申请到硬件资源后更新。