**Lab 2 Report**

1. **项目概述**

本项目是一个基于 Gradio 框架开发的 Web 聊天界面，旨在支持多种大语言模型的切换和调用，提供流式响应、多轮对话历史管理等功能，方便用户使用不同模型进行自然语言交互。

本系统支持对 OpenAI、DeepSeek 等模型的调用，并通过统一的接口封装，实现模块化管理和扩展。前端采用 Gradio 构建，组件清晰，交互简洁直观。

1. **项目功能**

|  |  |
| --- | --- |
| **功能类别** | **描述** |
| 多模型支持 | 自定义或预设调用多个不同大语言模型（OpenAI、DeepSeek 等） |
| 流式响应 | 实现 token 级别的实时输出，提高交互体验 |
| 历史对话管理 | 用户可查看、删除和切换不同会话 |
| UI 前端 | 使用 Gradio 实现响应式界面，支持对话区域、模型选择、会话栏等 |

1. **Tasks:**

**3.1不同模型在进行API调用的时候是否有区别，如果有区别，区别在什么地方?  
3.1.1 请求结构差异**

不同平台的模型对请求参数的结构定义并不统一。例如：

OpenAI GPT-4/3.5 使用 messages 字段，包含一组带有 role 的对话信息；

Anthropic Claude 使用 prompt 或 messages，不严格区分 system/user/assistant；

DeepSeek/Moonshot 多采用与 OpenAI 类似的结构，但字段命名可能存在微调（如 stream / streaming）。

这种差异要求开发者在统一封装时对参数做动态适配，以实现 Prompt 的跨模型兼容。

**3.1.2 身份验证机制不同**

各模型提供商在身份验证方面各有要求：

OpenAI 和 DeepSeek 通常使用 Bearer Token；

Claude API 需要 x-api-key 头；

Moonshot 可能还需要额外的请求签名校验。

统一管理 API 密钥并在调用中注入对应 Header，是系统开发中的重要安全环节。

**3.1.3 响应结构格式不同**

返回的结构同样存在不一致性：

OpenAI 通常返回 response['choices'][0]['message']['content']

Claude 返回 response['completion']

DeepSeek 返回 response['data']['content']

因此，系统必须实现一套“统一的响应提取逻辑”，确保后续模块处理的是标准格式。

**3.2自己在设计用户交互页面时的思路，自己为什么要这样子设计自己的页面，页面当中的组件的作用都是什么**