LLM 双模型对话系统（DeepSeek & Kimi）

概述

这是一个基于 Gradio 的用户可视化界面，允许用户选择不同的大型语言模型（LLM）进行对话。目前支持两种模型：DeepSeek 和 Kimi。该系统提供了基础对话功能和多轮对话功能，能够记录用户的对话历史，并将其与新的输入一起发送给大模型以实现连续对话。

特性

- **基础对话功能**：可以通过 API 调用不同的 LLM 模型来回答用户的问题。
- **多轮对话功能**：记录用户的对话历史，实现连续对话。

安装依赖库

确保你已经安装了 gradio 和 requests 库。你可以使用以下命令来安装这些库：

```
pip install gradio requests
```

运行

- 将上述代码保存为一个 Python 文件，例如 llm_chat_system.py。
- 在终端中导航到保存文件的目录。
- 运行脚本：python llm_chat_system.py
- 打开浏览器并访问

  http://localhost:7860

  即可看到并使用对话系统。

使用说明

界面布局

- **Markdown 标题**：显示系统名称和描述。
- **模型选择下拉菜单**：用户可以选择要使用的 LLM 模型（"DeepSeek" 或 "Kimi"），默认值为"DeepSeek"。
- **清空对话按钮**：清空当前对话历史。
- **聊天机器人组件**：显示对话内容，包括用户和助手的消息。
- **文本输入框**：用户可以在此输入消息并提交。

功能操作

- **选择模型**：在下拉菜单中选择要使用的 LLM 模型。
- **输入消息**：在文本输入框中输入你的问题或消息。
- **提交消息**：点击回车键或点击“提交”按钮发送消息。
- **查看回复**：对话机器人的回复将显示在聊天机器人组件中。
- **多轮对话**：对话历史会被记录，新的输入会包含之前的对话内容。

- **清空对话**：点击“清空对话”按钮清除当前对话历史。

代码结构

1. 导入库
导入所需的库，包括 gradio、requests、os 和类型注解。
2. 配置区
定义 DeepSeek 和 Kimi 的 API 密钥和 URL。
3. 模型核心逻辑
- **build_messages**：构建适合 API 的消息格式。
- **call_deepseek**：调用 DeepSeek API。
- **call_kimi**：调用 Kimi API。

4. 主处理函数
- **handle_conversation**：处理用户输入并更新对话历史。
- **clear_chat**：清空对话历史。

5. Gradio 界面
- 创建主题。
- 创建界面组件，包括 Markdown 标题、下拉菜单、清空对话按钮、聊天机器人组件和文本输入框。
- 绑定交互事件。

6. 启动服务器
启动 Gradio 应用程序，监听所有网络接口，使用端口 7860，并且不共享到公网。