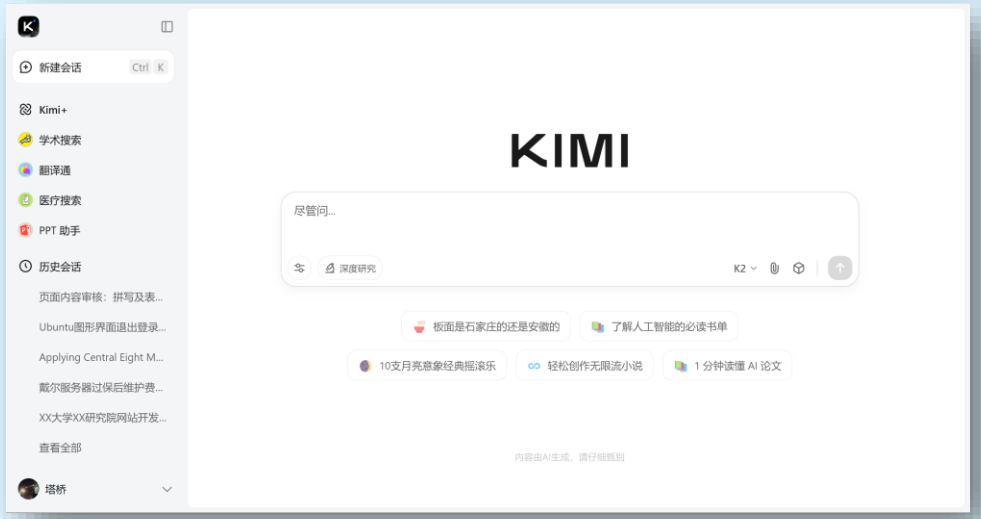

API

使用大模型：网页端


新建会话
Ctrl K
Kimi+
学术搜索
翻译通
医疗搜索
PPT 助手
历史会话
KIMI
尽管问...
深度研究
K2
板面是石家庄的还是安徽的
了解人工智能的必读书单
10支月亮意象经典摇滚乐
轻松创作无限流小说
1 分钟读懂 AI 论文
内容由AI生成，请仔细甄别
塔桥

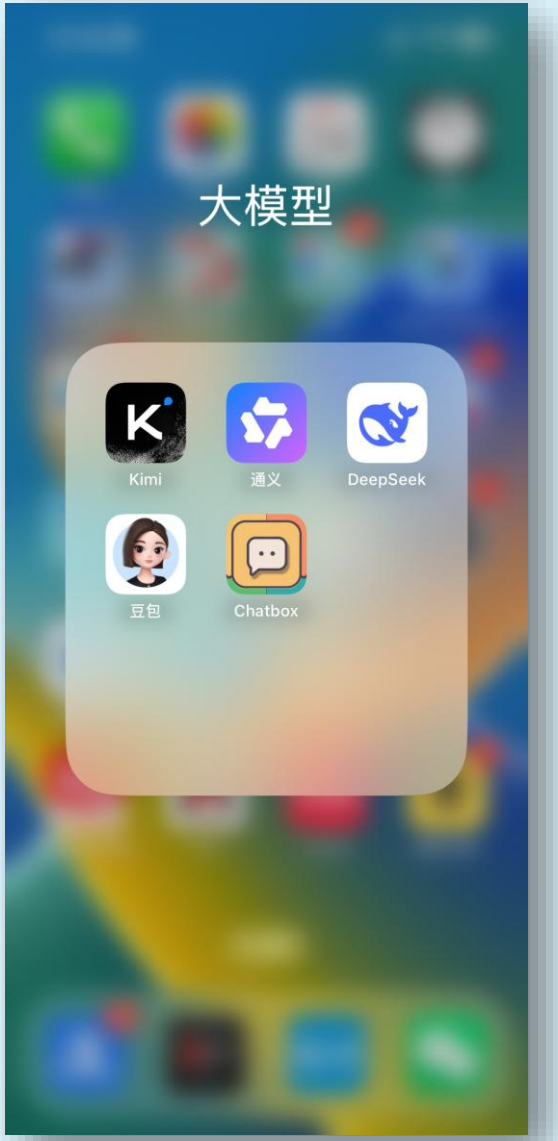

deepseek
开启新对话
今天
创建设置菜单流程图Mermaid代码
7 天内
GMM经典论文与应用指南
Markdown及其扩展工具关系解析
分娩前婴儿衣物准备清单解析
请对照原文，逐句翻译：《寝公...
非负实数求最大值问题
Excel整理名字组号方法总结
深度学习模型分类及应用概述
30 天内
基于RAG的问答系统流程图
塔桥
今天有什么可以帮到你?
给 DeepSeek 发送消息
深度思考
联网搜索


腾讯混元
API 接入
登录
对话
视觉生成
模型接入
开源模型
创作工具
最新动态
历史记录
Hunyuan-TurboS
混元 AI Studio
有问题，尽管问
联网搜索
AI分身
New
AI 播客
拍照搜题
深度搜索
AI翻译
探索更多使用案例

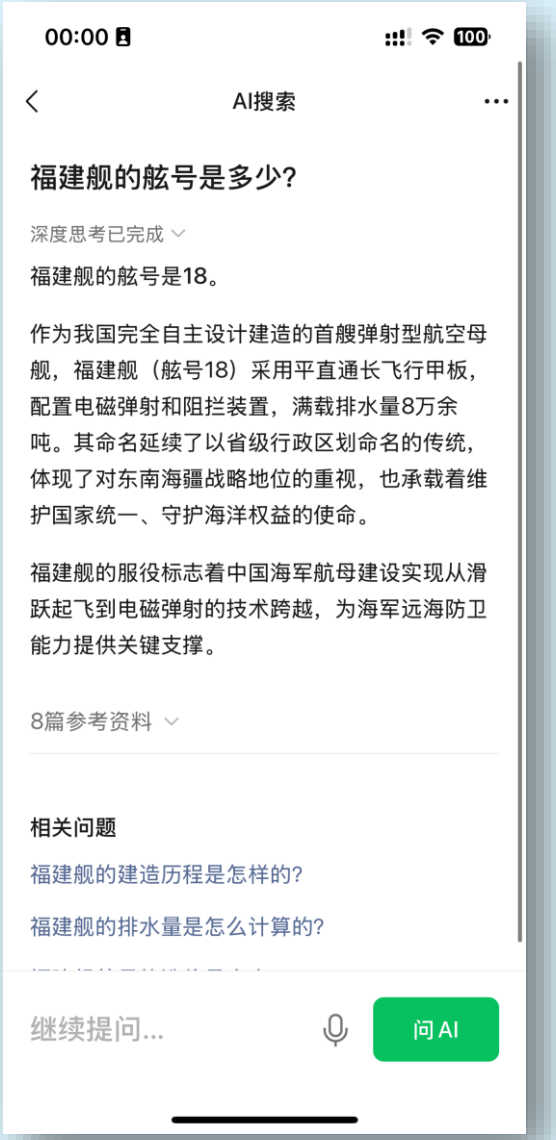

ChatGPT 5
新聊天
搜索聊天
库
Codex
Sora
GPT
DALL·E
PDF to Markdown
image generator
新项目
VAR-Infer-Revising
CRO
多重检验
BLLMFS
Carlton Chen
Plus
有什么可以帮忙的?
询问任何问题

使用大模型：手机、应用程序

淘宝AI搜索

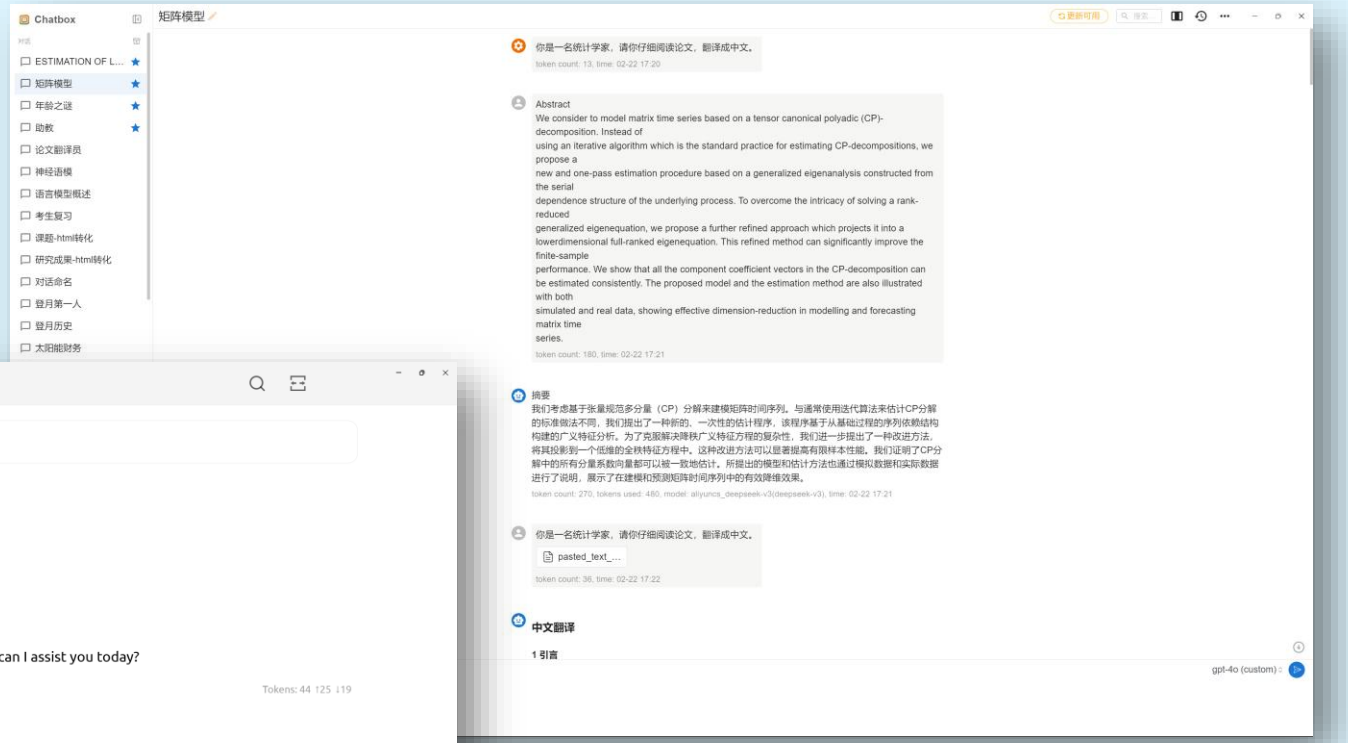


手机app

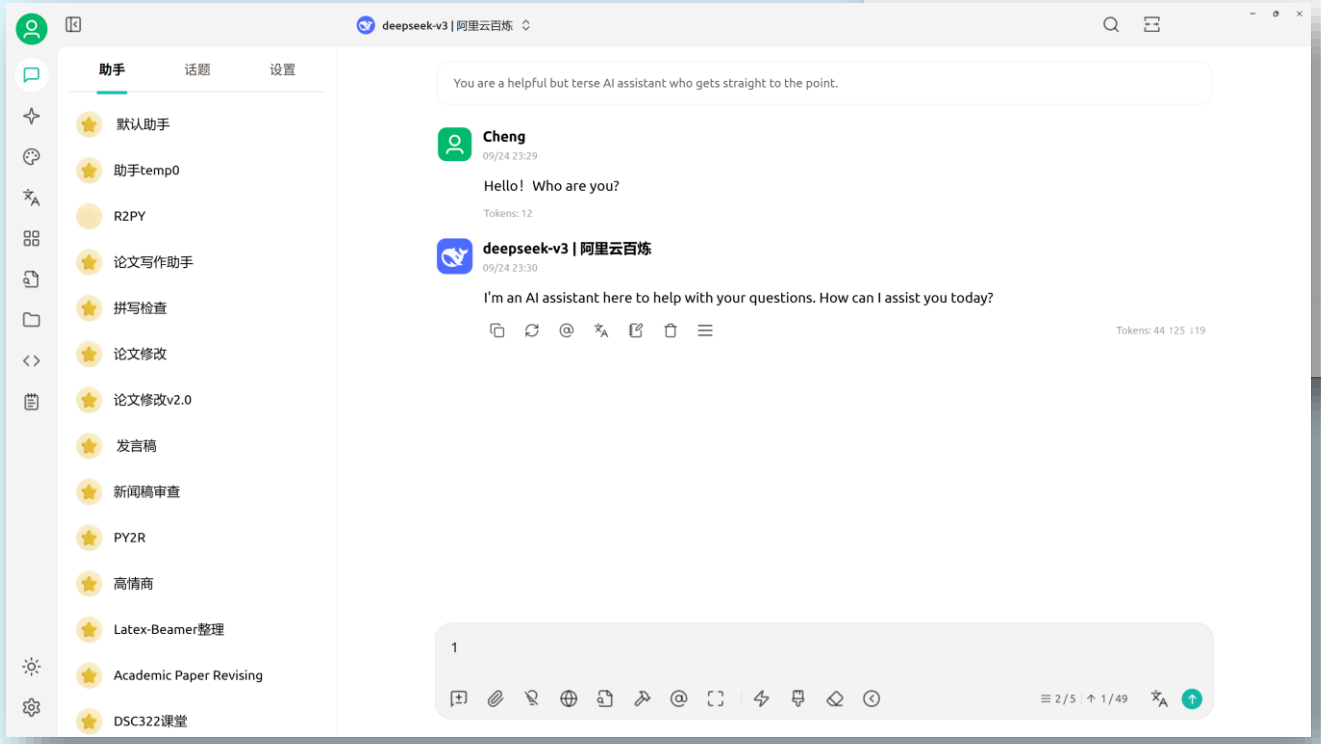


微信AI搜索

使用大模型：工具平台



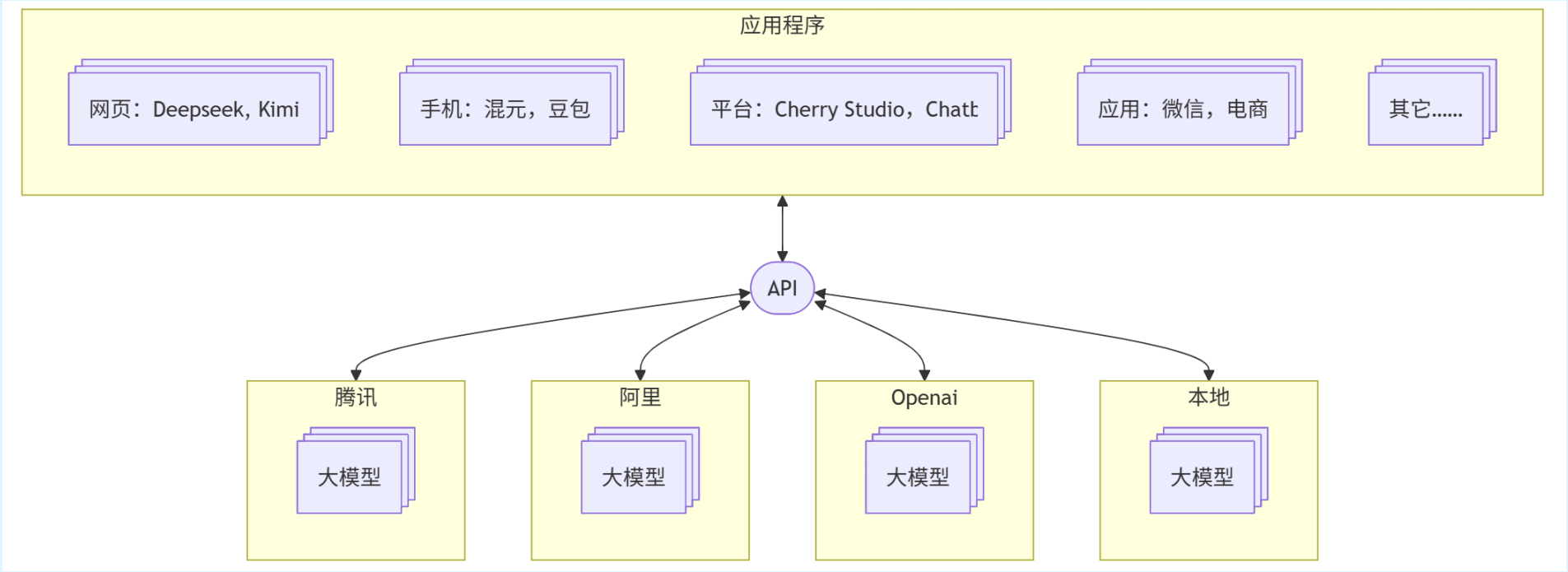
Chatbox



Cherry Studio

什么是API

- **API** 的全称是 **应用程序编程接口**（**A**pplication **P**rogram **I**nterface，API）。
  - **应用程序：** 指任何软件程序，比如你的手机App、网站、电脑软件。
  - **编程：** 指开发者通过写代码来与这个接口互动。
  - **接口：** 指两个不同系统之间相互通信和交互的“契约”或“连接点”。
- **API 是一套预先定义好的规则、协议和工具，允许不同的软件应用程序之间相互通信、交换数据和服务，而无需知道对方内部的实现细节。**

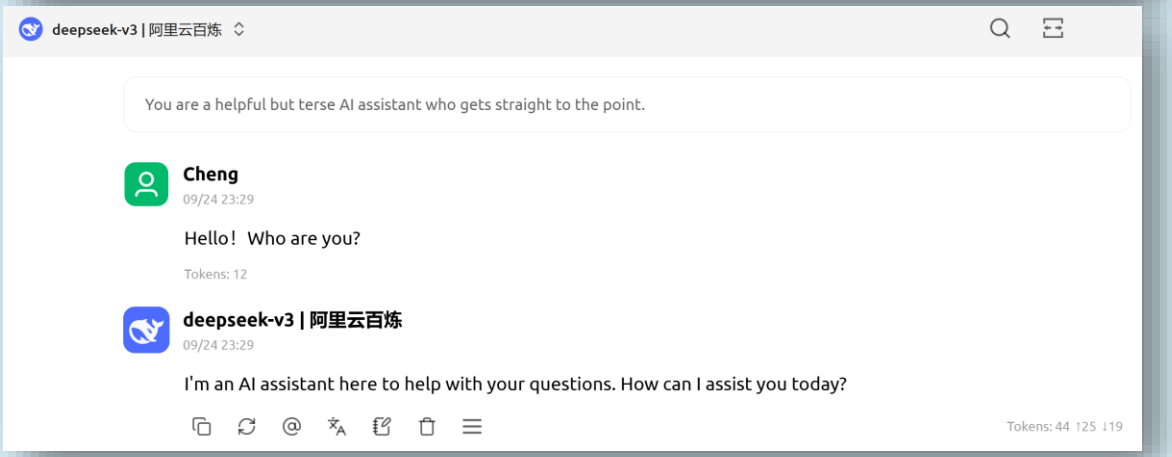


什么是API

- 想象一下你去一家餐厅吃饭：
  - **你（顾客）** 相当于一个**软件应用**（比如一个手机App或网站）。
  - **厨房** 相当于另一个**软件系统或数据库**（比如微信的服务器、银行的数据库、天气数据的来源）。厨房里有你需要的所有食材和资源，但你无法直接进入厨房。
  - **服务员** 就是 **API**。
  - **菜单** 就是 **API的文档**。
- 过程是这样的：
  1. 你看菜单（**查阅API文档**），决定点什么菜（**发起什么请求**）。
  2. 你告诉服务员：“我要一份牛排，七分熟”（**发出请求**）。
  3. 服务员把你的要求准确地传达给厨房（**API接收请求并传递给系统**）。
  4. 厨房做好牛排后，交给服务员（**系统处理完请求，返回数据**）。
  5. 服务员把牛排端到你桌上（**API将数据返回给你的应用**）。
- 在这个过程中，你不需要知道厨房是如何运作的，厨师是谁，牛排是从哪个冰箱里拿出来的。你只需要通过服务员（API）这个标准接口，就能获得你想要的食物（数据或功能）。

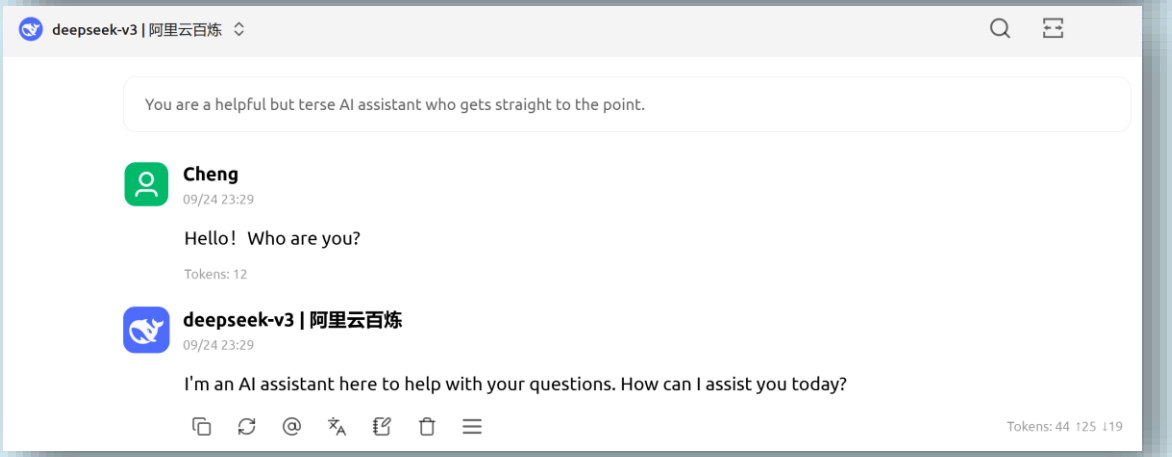
Python代码实现

```python
import OpenAI

with open('API-key.txt', 'r', encoding='utf-8') as file:
    key = file.read()

client = OpenAI(
    api_key = key,
    base_url = "https://dashscope.aliyuncs.com/compatible-mode/v1/
    ",
)

def get_llm_response(prompt):
    """This function takes as input a prompt, which must be a
    string enclosed in quotation marks,
    and passes it to deepseek-v3 model. The function then saves
    the response of the model as
    a string.
    """
    completion = client.chat.completions.create(
        model="deepseek-v3",
        messages=[
            {
                "role": "system",
                "content": "You are a helpful but terse AI
                assistant who gets straight to the point.",
            },
            {"role": "user", "content": prompt},
        ],
        temperature=0.0,
    )
    response = completion.choices[0].message.content
    return response
```



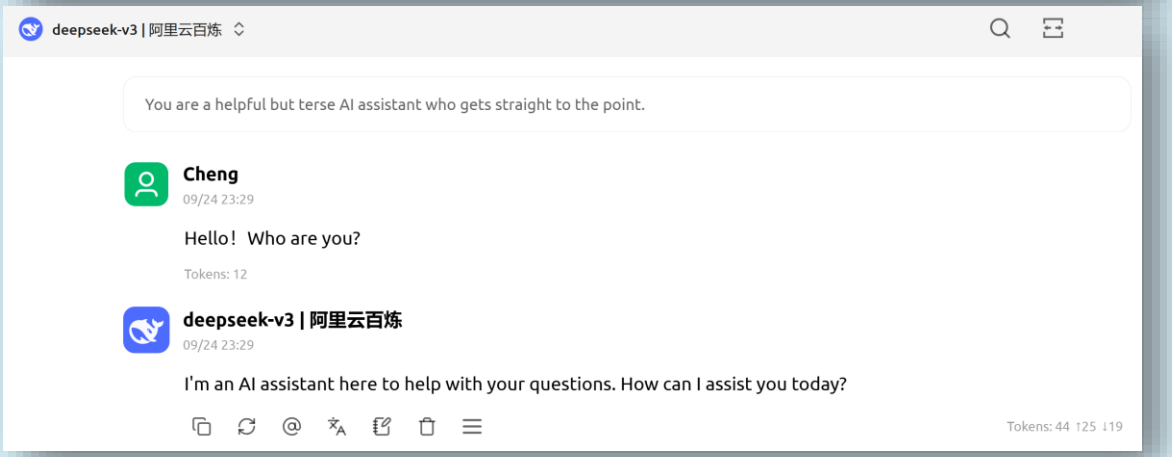
Python代码实现

```python
import OpenAI

with open('API-key.txt', 'r', encoding='utf-8') as file:
    key = file.read()

client = OpenAI(
    api_key = key,
    base_url = "https://dashscope.aliyuncs.com/compatible-mode/v1/
    ",
)

def get_llm_response(prompt):
    """This function takes as input a prompt, which must be a
    string enclosed in quotation marks,
    and passes it to deepseek-v3 model. The function then saves
    the response of the model as
    a string.
    """
    completion = client.chat.completions.create(
        model="deepseek-v3",
        messages=[
            {
                "role": "system",
                "content": "You are a helpful but terse AI
                assistant who gets straight to the point.",
            },
            {"role": "user", "content": prompt},
        ],
        temperature=0.0,
    )
    response = completion.choices[0].message.content
    return response
```



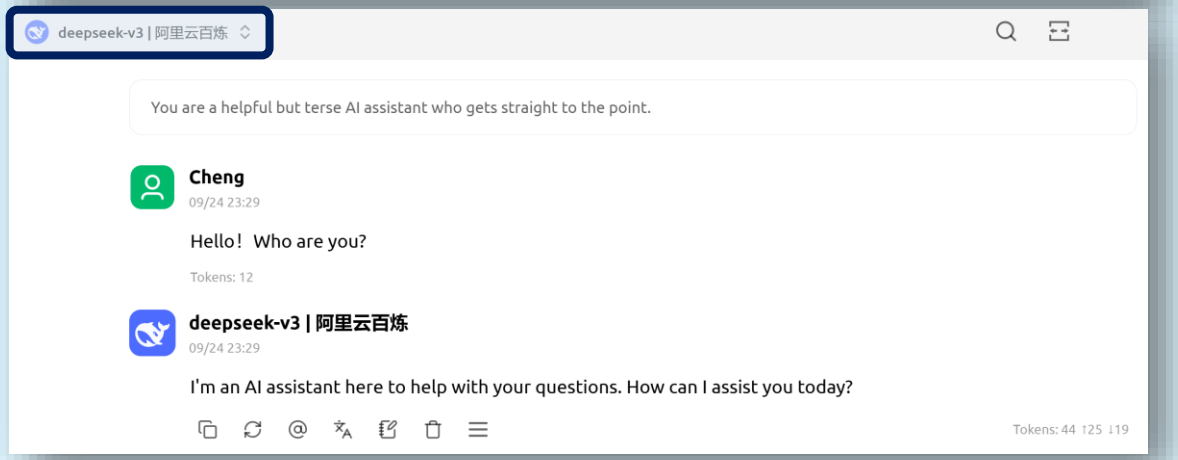
Python代码实现

```python
import OpenAI

with open('API-key.txt', 'r', encoding='utf-8') as file:
    key = file.read()

client = OpenAI(
    api_key = key,
    base_url = "https://dashscope.aliyuncs.com/compatible-mode/v1/
    ",
)

def get_llm_response(prompt):
    """This function takes as input a prompt, which must be a
    string enclosed in quotation marks,
    and passes it to deepseek-v3 model. The function then saves
    the response of the model as
    a string.
    """
    completion = client.chat.completions.create(
        model="deepseek-v3",
        messages=[
            {
                "role": "system",
                "content": "You are a helpful but terse AI
                assistant who gets straight to the point.",
            },
            {"role": "user", "content": prompt},
        ],
        temperature=0.0,
    )
    response = completion.choices[0].message.content
    return response
```



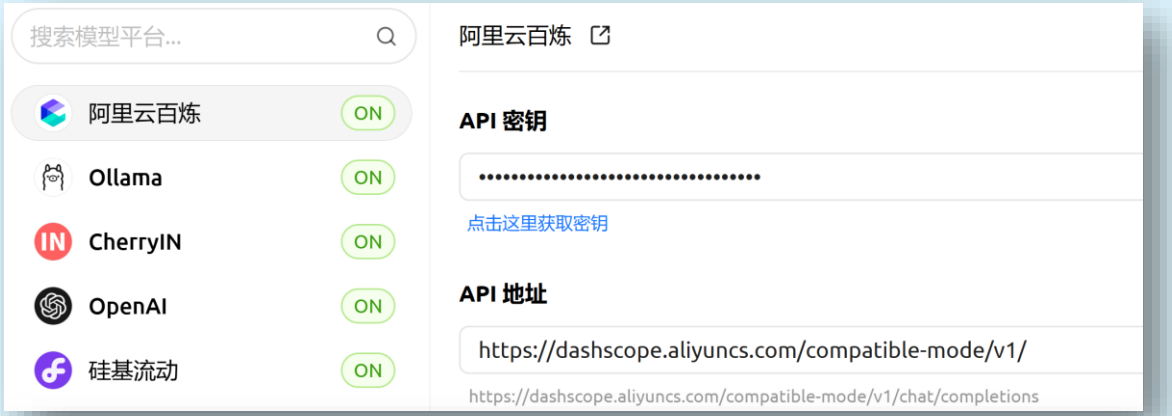
Python代码实现

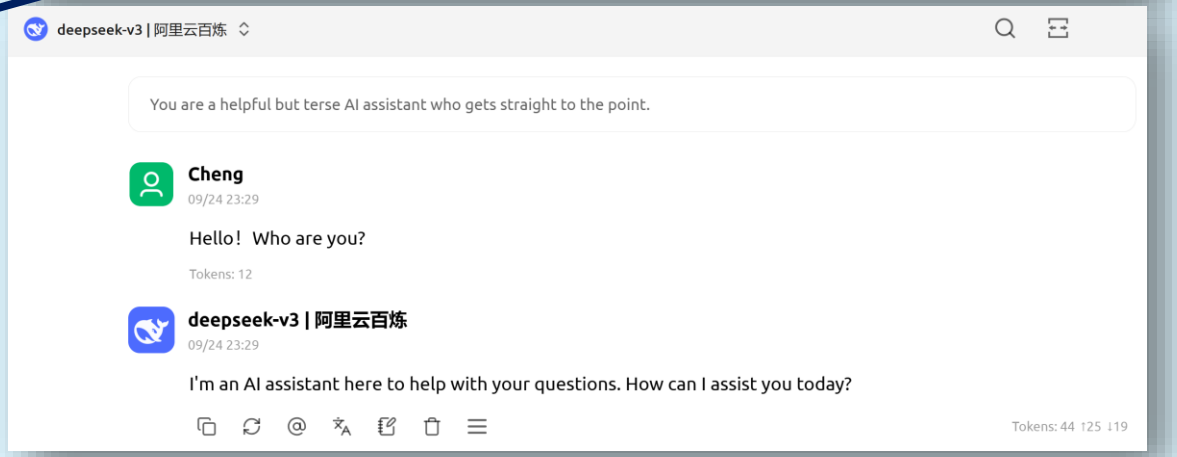
```python
import OpenAI

with open('API-key.txt', 'r', encoding='utf-8') as file:
    key = file.read()

client = OpenAI(
    api_key = key,
    base_url = "https://dashscope.aliyuncs.com/compatible-mode/v1/
    ",
)

def get_llm_response(prompt):
    """This function takes as input a prompt, which must be a
    string enclosed in quotation marks,
    and passes it to deepseek-v3 model. The function then saves
    the response of the model as
    a string.
    """
    completion = client.chat.completions.create(
        model="deepseek-v3",
        messages=[
            {
                "role": "system",
                "content": "You are a helpful but terse AI
                assistant who gets straight to the point.",
            },
            {"role": "user", "content": prompt},
        ],
        temperature=0.0,
    )
    response = completion.choices[0].message.content
    return response
```



Python代码实现

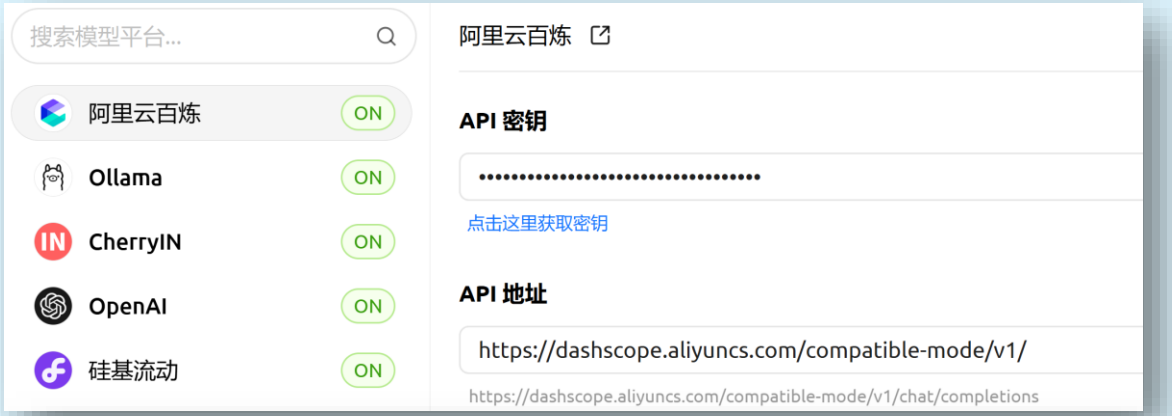
```python
import OpenAI

with open('API-key.txt', 'r', encoding='utf-8') as file:
    key = file.read()

client = OpenAI(
    api_key = key,
    base_url = "https://dashscope.aliyuncs.com/compatible-mode/v1/
    ",
)

def get_llm_response(prompt):
    """This function takes as input a prompt, which must be a
    string enclosed in quotation marks,
    and passes it to deepseek-v3 model. The function then saves
    the response of the model as
    a string.
    """
    completion = client.chat.completions.create(
        model="deepseek-v3",
        messages=[
            {
                "role": "system",
                "content": "You are a helpful but terse AI
                assistant who gets straight to the point.",
            },
            {"role": "user", "content": prompt},
        ],
        temperature=0.0,
    )
    response = completion.choices[0].message.content
    return response
```

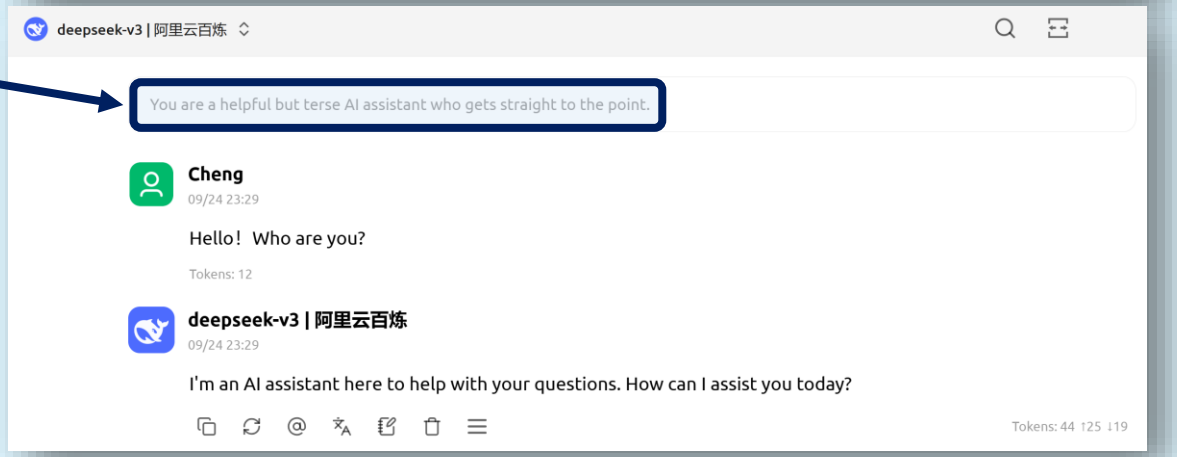




Python代码实现

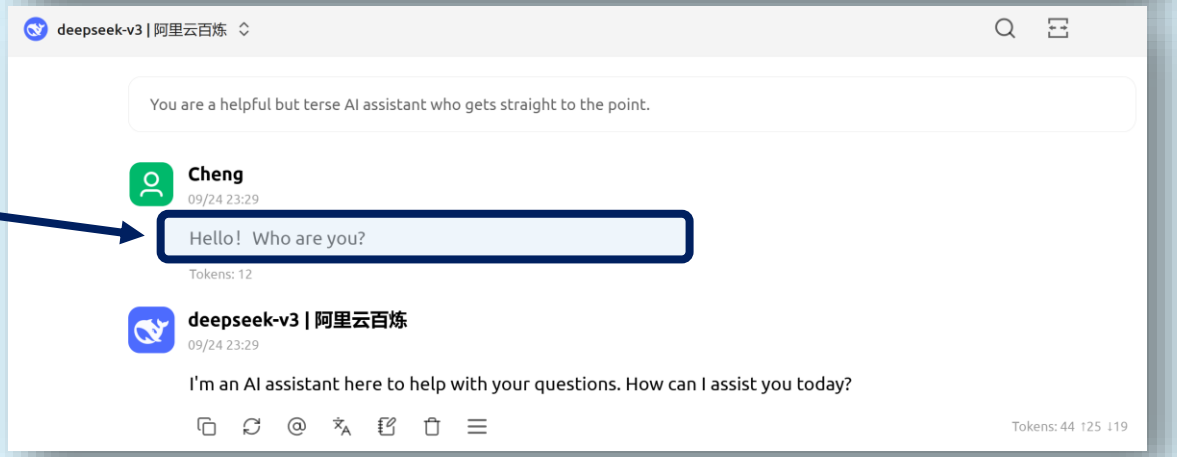
```python
import OpenAI

with open('API-key.txt', 'r', encoding='utf-8') as file:
    key = file.read()

client = OpenAI(
    api_key = key,
    base_url = "https://dashscope.aliyuncs.com/compatible-mode/v1/
    ",
)

def get_llm_response(prompt):
    """This function takes as input a prompt, which must be a
    string enclosed in quotation marks,
    and passes it to deepseek-v3 model. The function then saves
    the response of the model as
    a string.
    """
    completion = client.chat.completions.create(
        model="deepseek-v3",
        messages=[
            {
                "role": "system",
                "content": "You are a helpful but terse AI
                assistant who gets straight to the point.",
            },
            {"role": "user", "content": prompt},
        ],
        temperature=0.0,
    )
    response = completion.choices[0].message.content
    return response
```





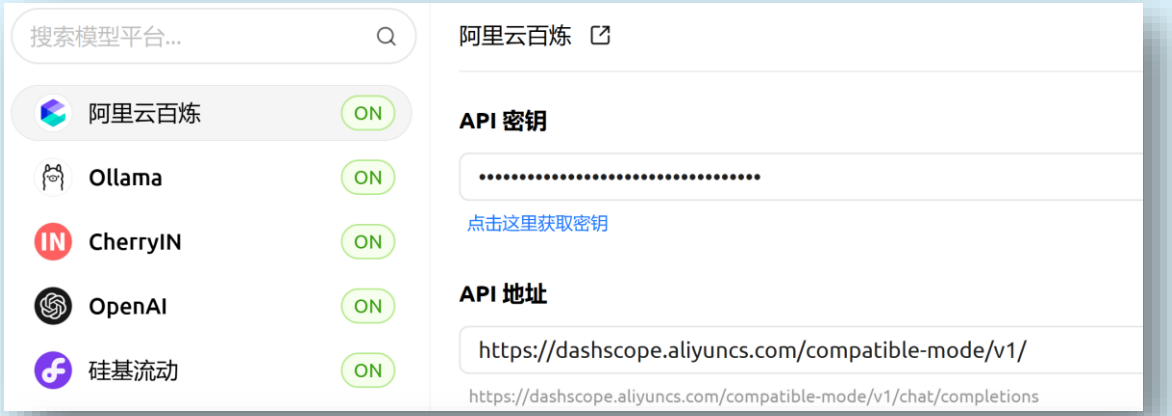
Python代码实现

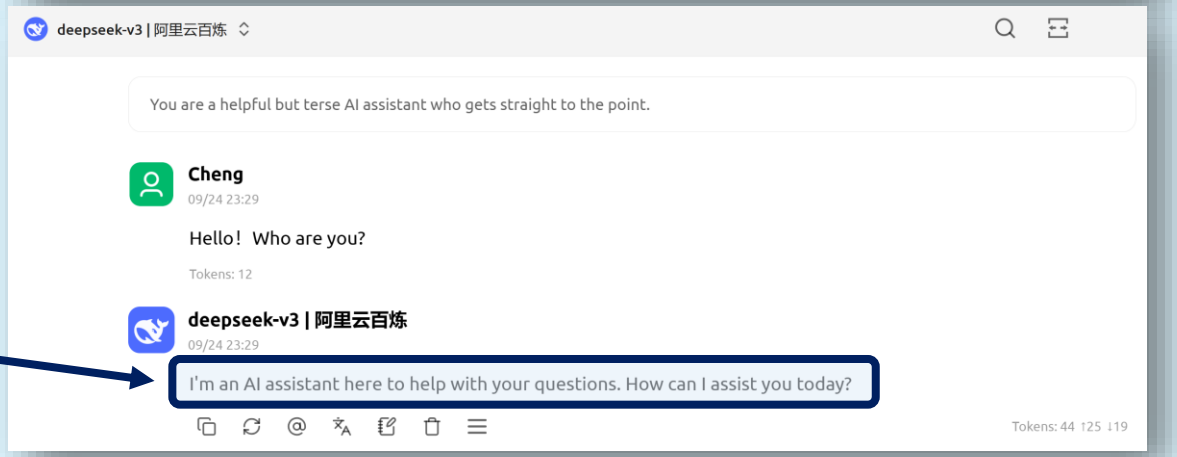
```python
import OpenAI

with open('API-key.txt', 'r', encoding='utf-8') as file:
    key = file.read()

client = OpenAI(
    api_key = key,
    base_url = "https://dashscope.aliyuncs.com/compatible-mode/v1/
    ",
)

def get_llm_response(prompt):
    """This function takes as input a prompt, which must be a
    string enclosed in quotation marks,
    and passes it to deepseek-v3 model. The function then saves
    the response of the model as
    a string.
    """
    completion = client.chat.completions.create(
        model="deepseek-v3",
        messages=[
            {
                "role": "system",
                "content": "You are a helpful but terse AI
                assistant who gets straight to the point.",
            },
            {"role": "user", "content": prompt},
        ],
        temperature=0.0,
    )
    response = completion.choices[0].message.content
    return response
```



Python代码实现

```python
import OpenAI

with open('API-key.txt', 'r', encoding='utf-8') as file:
    key = file.read()

client = OpenAI(
    api_key = key,
    base_url = "https://dashscope.aliyuncs.com/compatible-mode/v1/
    ",
)

def get_llm_response(prompt):
    """This function takes as input a prompt, which must be a
    string enclosed in quotation marks,
    and passes it to deepseek-v3 model. The function then saves
    the response of the model as
    a string.
    """
    completion = client.chat.completions.create(
        model="deepseek-v3",
        messages=[
            {
                "role": "system",
                "content": "You are a helpful but terse AI
                assistant who gets straight to the point.",
            },
            {"role": "user", "content": prompt},
        ],
        temperature=0.0,
    )
    response = completion.choices[0].message.content
    return response
```



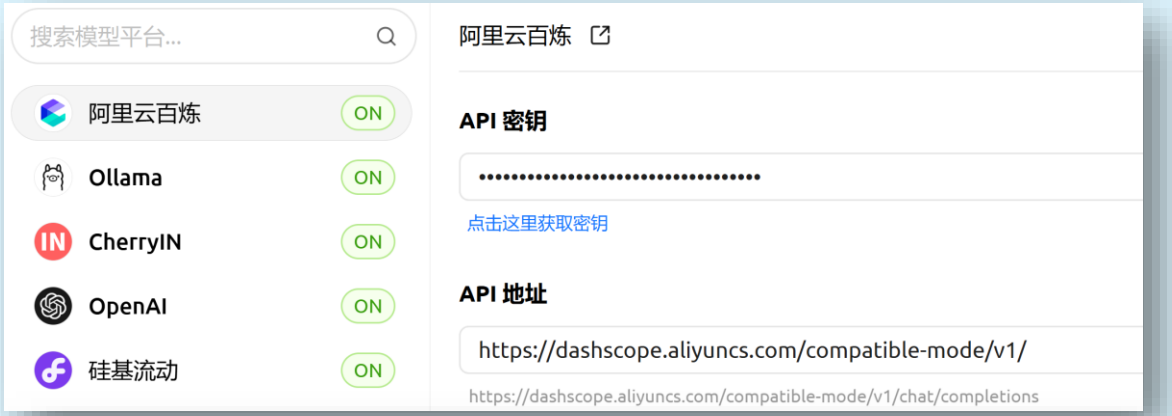


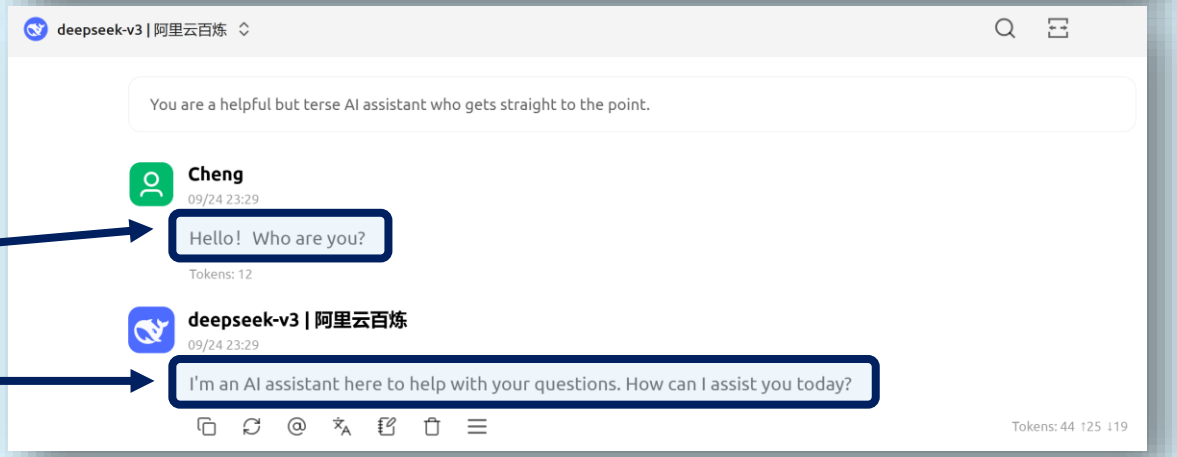
Python代码实现

```python
import OpenAI

with open('API-key.txt', 'r', encoding='utf-8') as file:
    key = file.read()

client = OpenAI(
    api_key = key,
    base_url = "https://dashscope.aliyuncs.com/compatible-mode/v1/
    ",
)

def get_llm_response(prompt):
    """This function takes as input a prompt, which must be a
    string enclosed in quotation marks,
    and passes it to deepseek-v3 model. The function then saves
    the response of the model as
    a string.
    """
    completion = client.chat.completions.create(
        model="deepseek-v3",
        messages=[
            {
                "role": "system",
                "content": "You are a helpful but terse AI
                assistant who gets straight to the point.",
            },
            {"role": "user", "content": prompt},
        ],
        temperature=0.0,
    )
    response = completion.choices[0].message.content
    return response
```

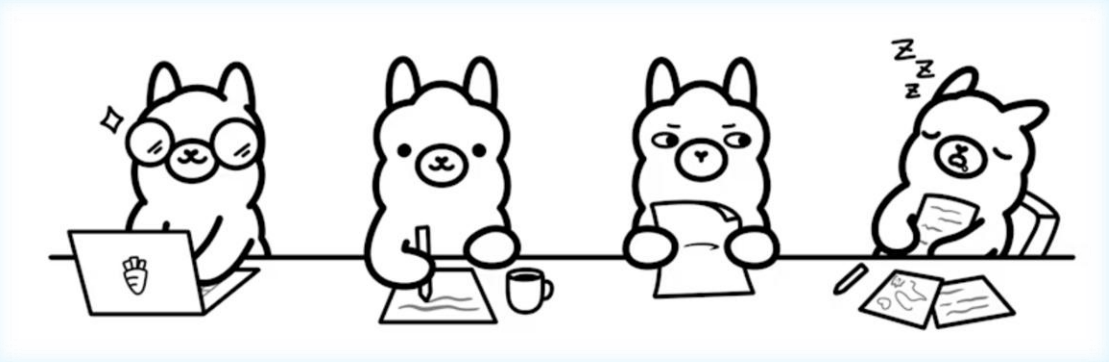
```
prompt = "Hello! Who are you?"
get_llm_response(prompt)
```

```
"I'm an AI assistant here to help with your questions. What do you need?"
```





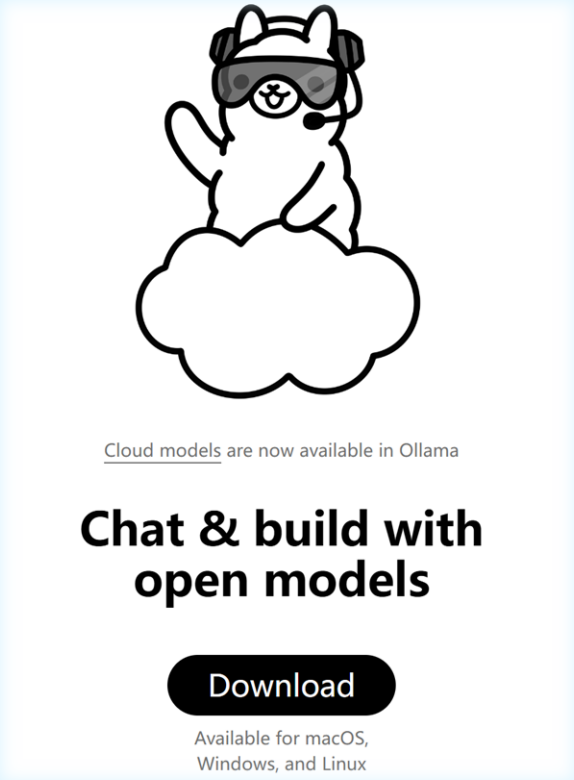
本地部署大模型：Ollama

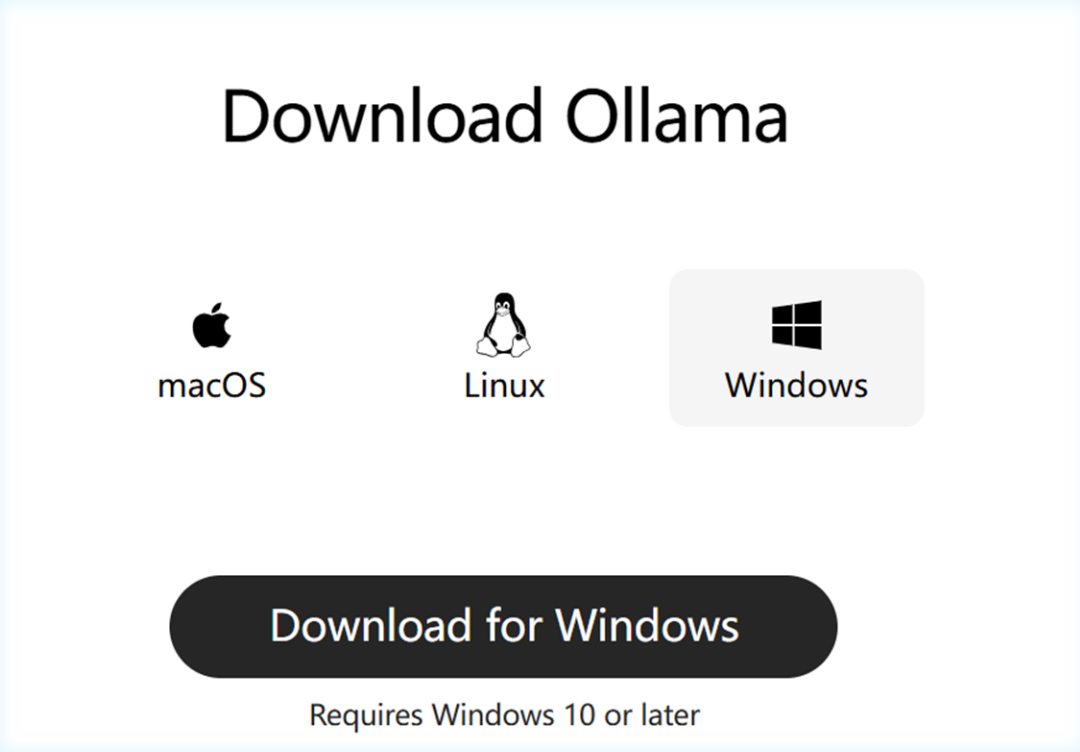


- Ollama 是一个本地大模型运行与管理工具，主要用于在个人电脑上快速部署和调用各种开源大语言模型。它的特点包括：
  - ✓ **本地运行：**Ollama 可以让你直接在本地电脑上运行大模型，不需要依赖云端。这样既能提高隐私和安全性，也可以在离线环境中使用。
  - ✓ **简单调用：**它提供了命令行工具和 API 接口，让用户可以像调用 ChatGPT 一样来运行模型。
  - ✓ **模型管理：**支持下载和管理多个开源模型（如 Llama 2、Mistral、Deepseek 等）。**可扩展与定制：**可以加载本地自定义模型或微调模型。与其他开发框架（如 Python、Node.js、前端应用）结合，用于构建 AI 应用。
  - ✓ **跨平台支持：**Ollama 目前支持 **macOS**、**Linux** 和 **Windows**（预览版），安装方式可直接下载安装包。

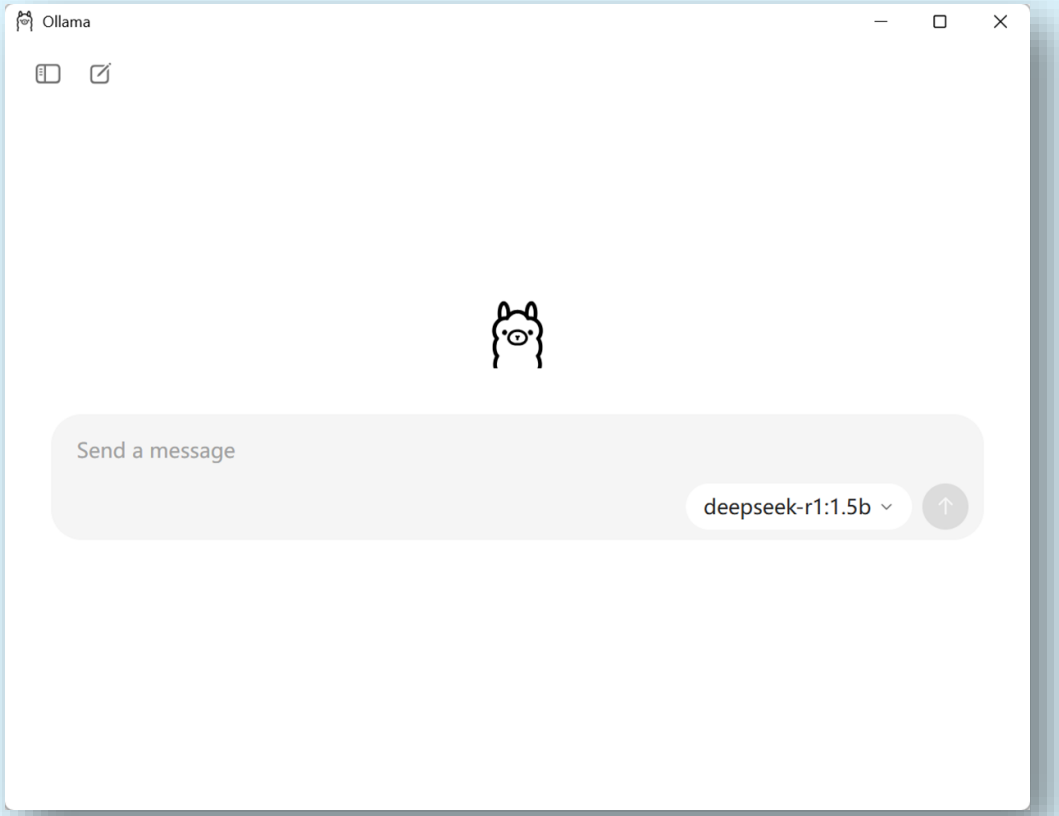
本地部署大模型：Ollama

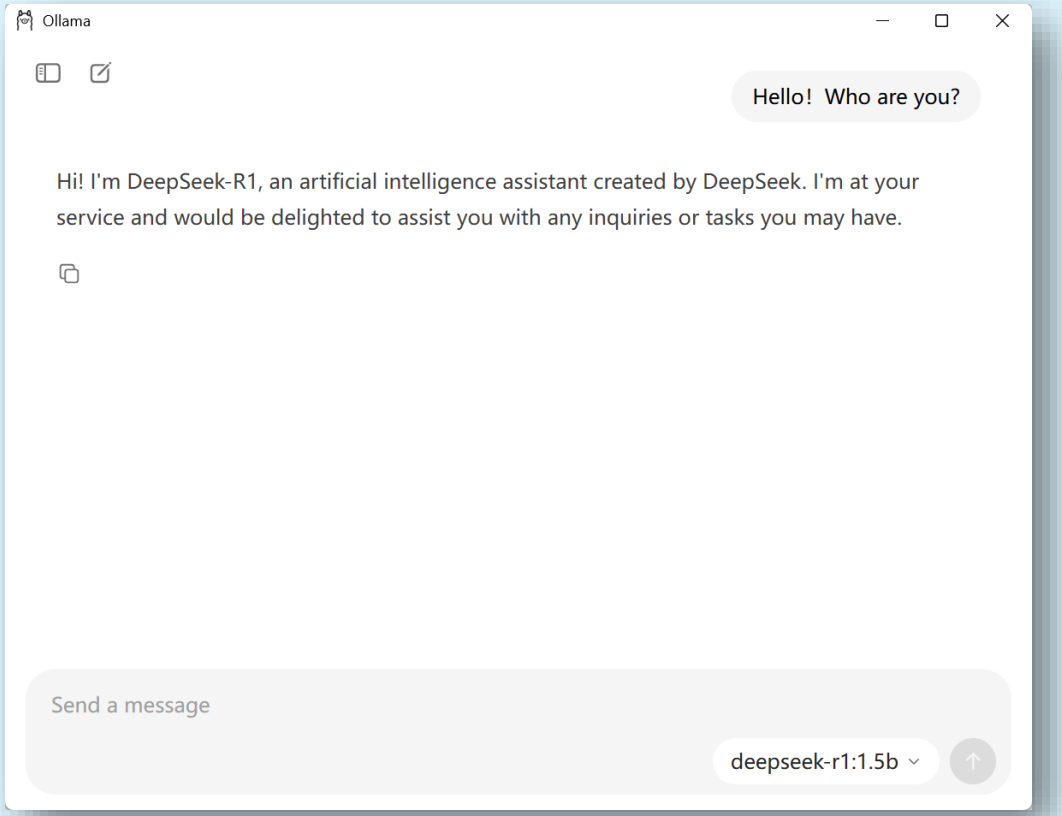
下载地址：https://ollama.com/download



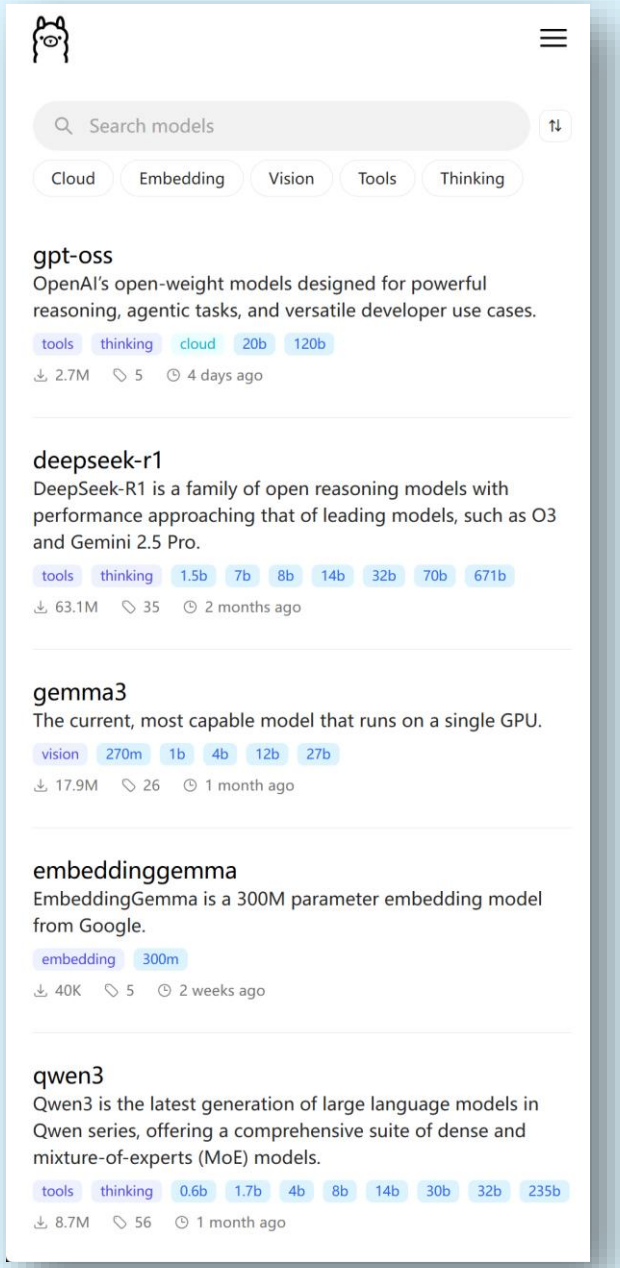


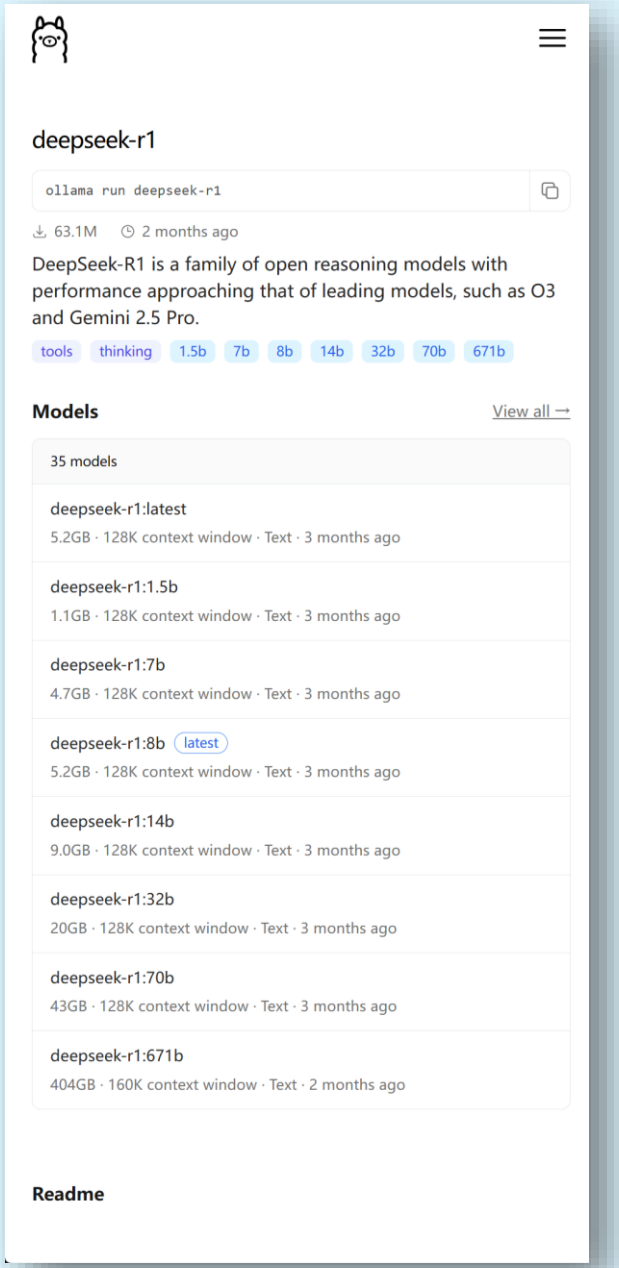
本地部署大模型：Ollama

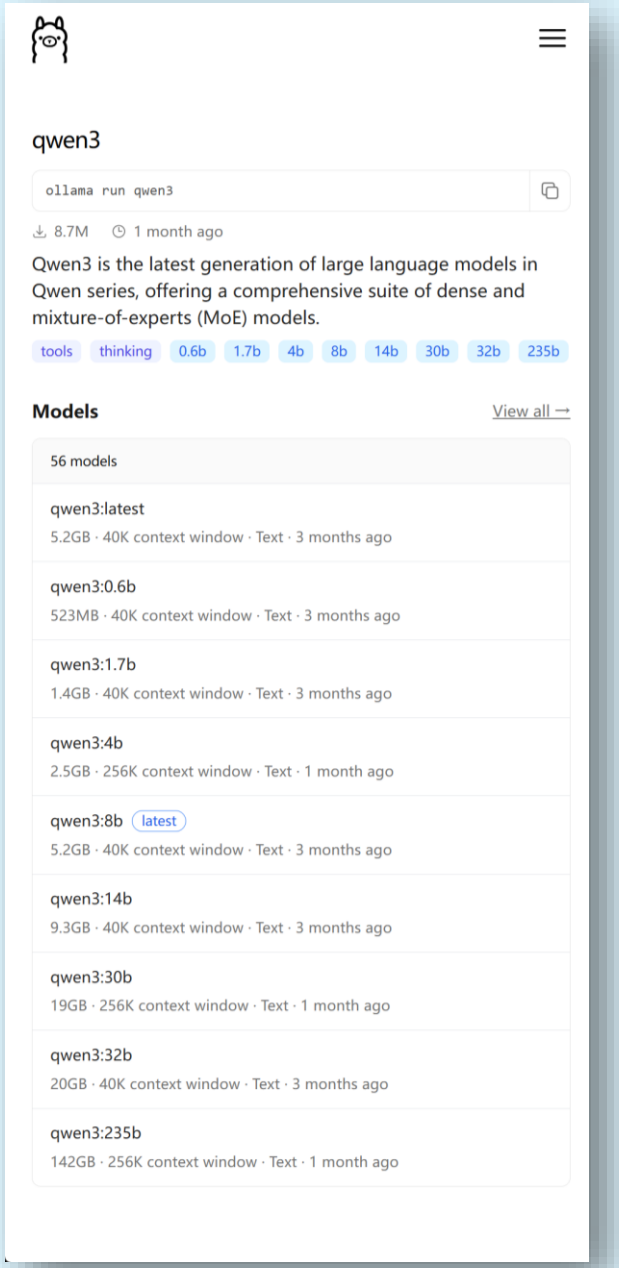




本地部署大模型：Ollama



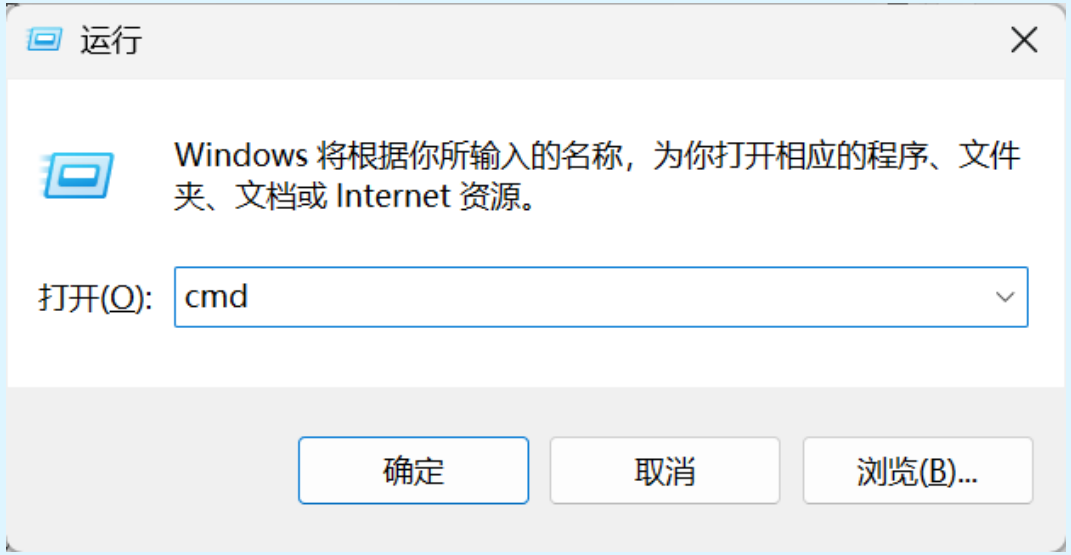




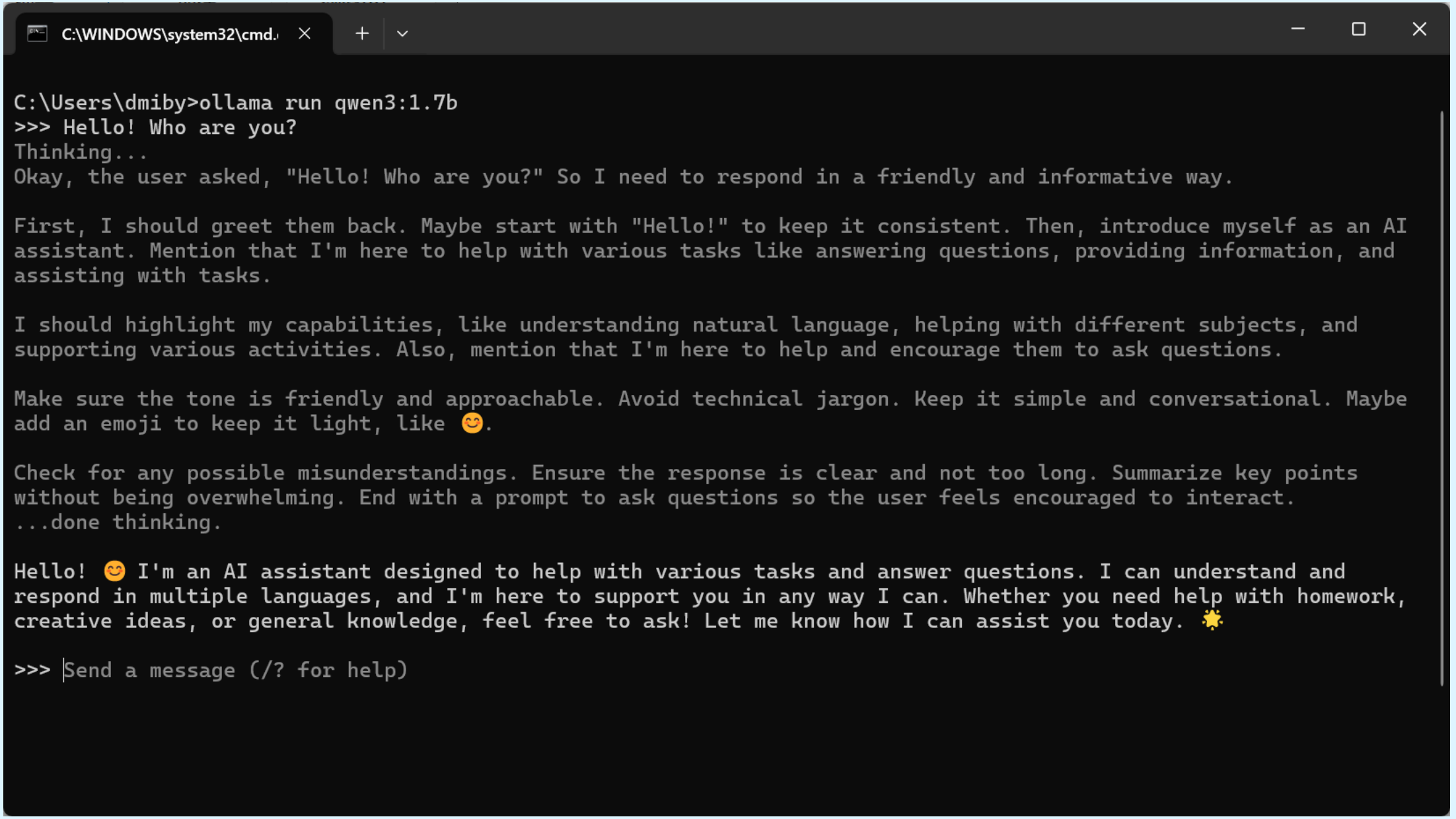
本地部署大模型：Ollama

以Windows系统为例：

- **Win + R**：打开“运行”对话框
- 输入 **cmd 回车**，打开命令行窗口
- 输入 **Ollama run qwen3:1.7b**
  - ✓ 首次运行将下载qwen3:1.7b模型
  - ✓ 以后运行，将开启对话



本地部署大模型：Ollama



本地部署大模型：Ollama

本地部署大模型：Ollama