LLMBackendRequest API 文档

请求格式

请求方法：POST
Content-Type: application/json
字符编码：UTF-8
主对象结构

字段名	类型	必填	描述
model	string	是	模型名称
enginePrompt	string	是	提供给引擎的LLM级别提示文本
active	Dict<string,NodeDocument>	是	活动文档映射，key为文档ID，value为NodeDocument对象
reference	LLMReference[]	是	LLMReference对象数组
referenceNodes	NodeDocument[]	是	NodeDocument类型的引用节点，key是节点 ID,value是节点内容, 解析完 JSON 后直接用 yaml 进行序列化
conversation	LLMConversation[]	是	LLMConversation对象数组

数据结构定义

LLMReference

可以是以下两种类型之一：

LLMDocumentReference

字段名	类型	必填	描述
type	string	是	固定值："document"
value	string	是	文档引用字符串

LLMWebsiteReference

字段名	类型	必填	描述
type	string	是	固定值："url"
value	string	是	URL字符串

LLMConversation

字段名	类型	必填	描述
type	string	是	会话类型，可选值：'system'、'assistant'、'user'、'tool'
content	string	是	会话内容
name	string	否	名称
tool_call_id	string	否	工具调用ID

示例

```
{
  "enginePrompt": "现在你在编写一个 xxxxx ，你要.....",
  "active": {
    "doc1": {
      "id": "doc1",
      "name": "示例文档",
      "description": "这是一个示例",
      "engine": "dolphin",
      "effects": [],
      "inputs": {},
      "nodes": [],
      "outputs": {}
    }// 这里直接全部使用 yaml 进行序列化
  },
  "reference": [
    {
      "type": "document",
      "key": ".....",
      "value": "参考文档1 内容:...."
    },{
      "type": "url",
      "key": "https://......",
      "value": "参考网址解析完的内容:...."
    }
  ],
  "referenceNodes": [{
    "id": "ref1",
    "name": "参考节点",
    "description": "参考节点描述",
    "engine": "dolphin",
    "effects": [],
    "inputs": {},
    "nodes": [],
    "outputs": {}
  }], // 这里直接全部使用 yaml 进行序列化
  "conversation": [ // 传入的历史对话列表，最后一个必定为 user（用户输入）
    {
      "type": "user",
      "content": "用户输入"
    }
  ]
```

```
}
```

注意: 所有 Document 类型传入大模型时使用 yaml 进行序列化