

使用教程



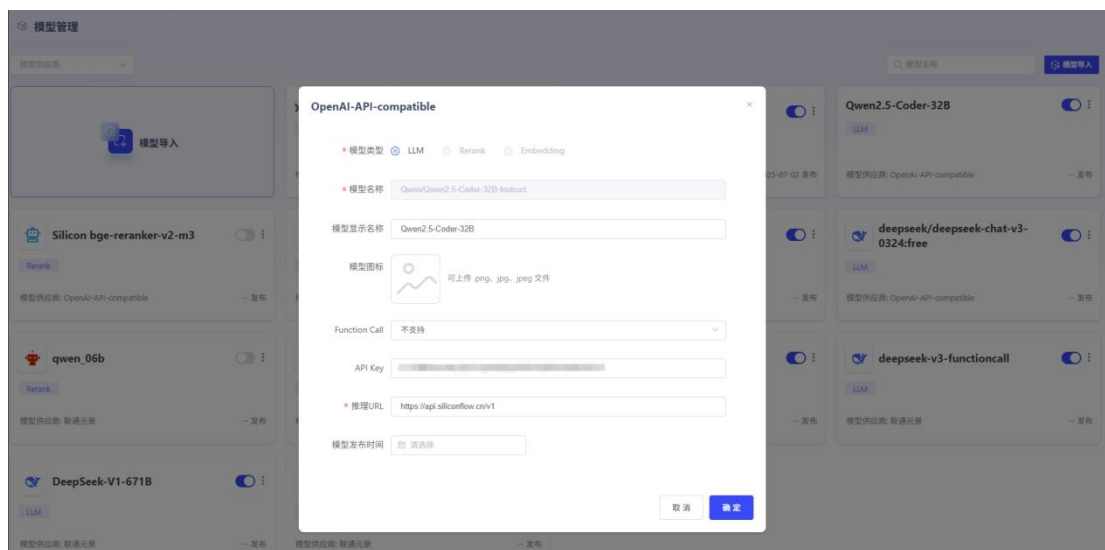
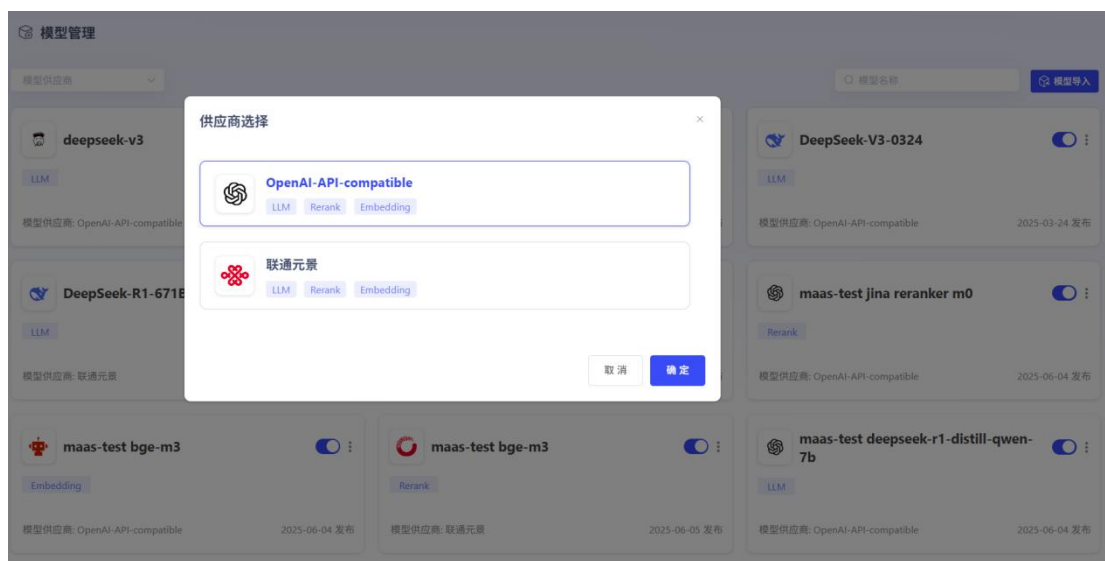
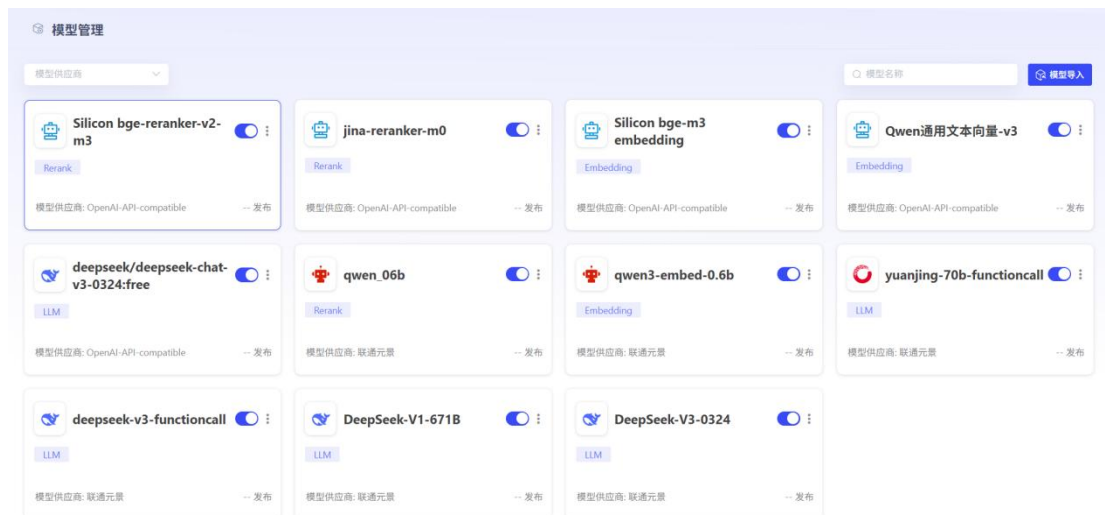
平台产品介绍

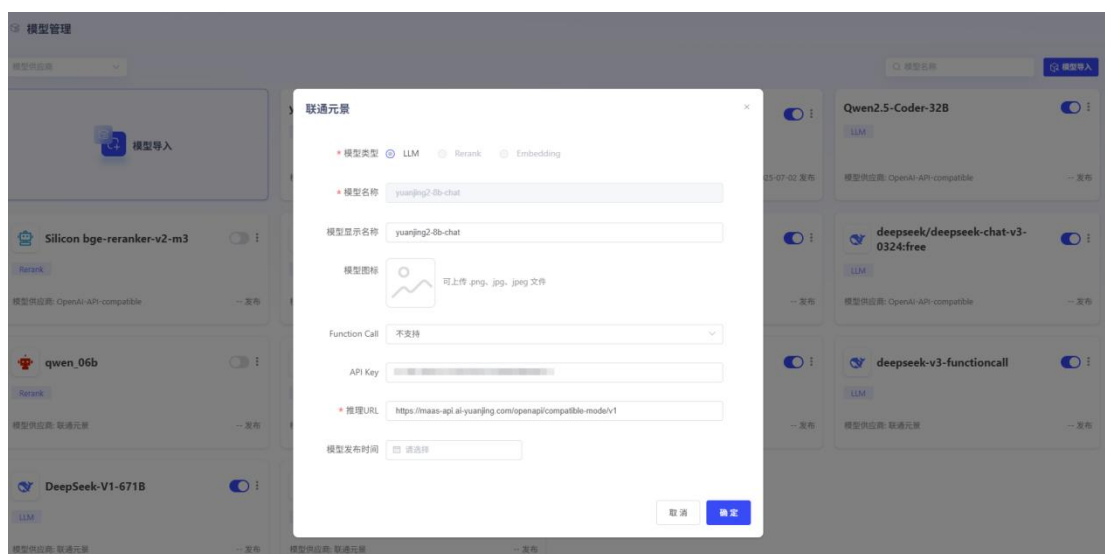
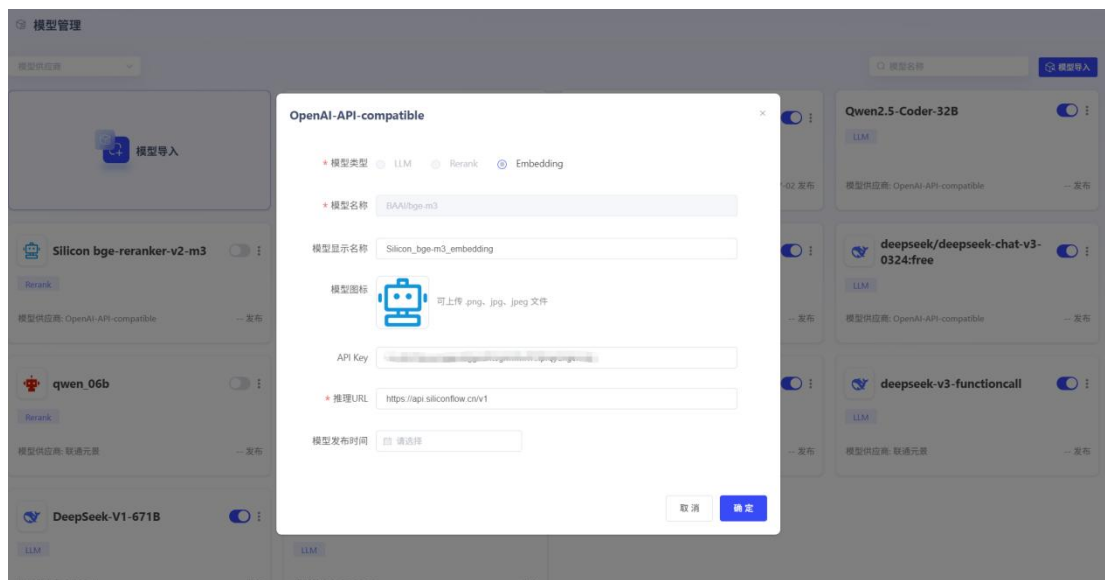
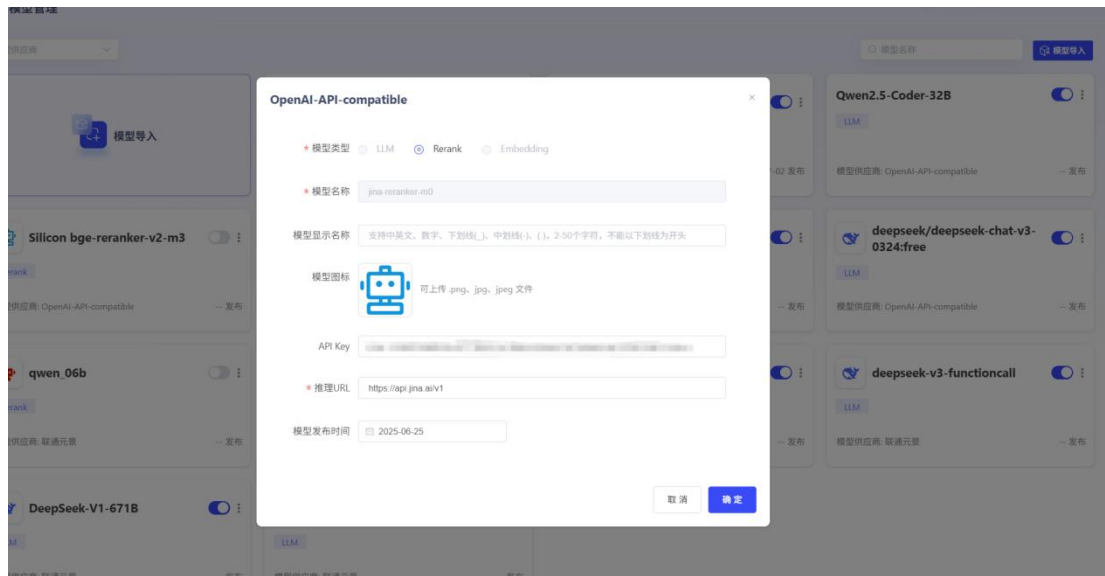
本平台是一款面向企业级场景的一站式、商用 license 友好的智能体开发平台，致力于为企业提供安全、高效、合规的一站式 AI 解决方案。我们以"技术开放、生态共建"为核心理念，通过整合大语言模型、业务流程自动化等前沿技术，构建了覆盖模型全生命周期管理、MCP、联网检索、智能体快速开发、企业知识库建设、复杂 workflow 编排等完整功能体系的 AI 工程化平台。平台采用模块化架构设计，支持灵活的功能扩展和二次开发，在确保企业数据安全和隐私保护的同时，大幅降低了 AI 技术的应用门槛。无论是中小型企业快速构建智能化应用，还是大型企业实现复杂业务场景的智能化改造，都能提供强有力的技术支撑，助力企业加速数字化转型进程，实现降本增效和业务创新。

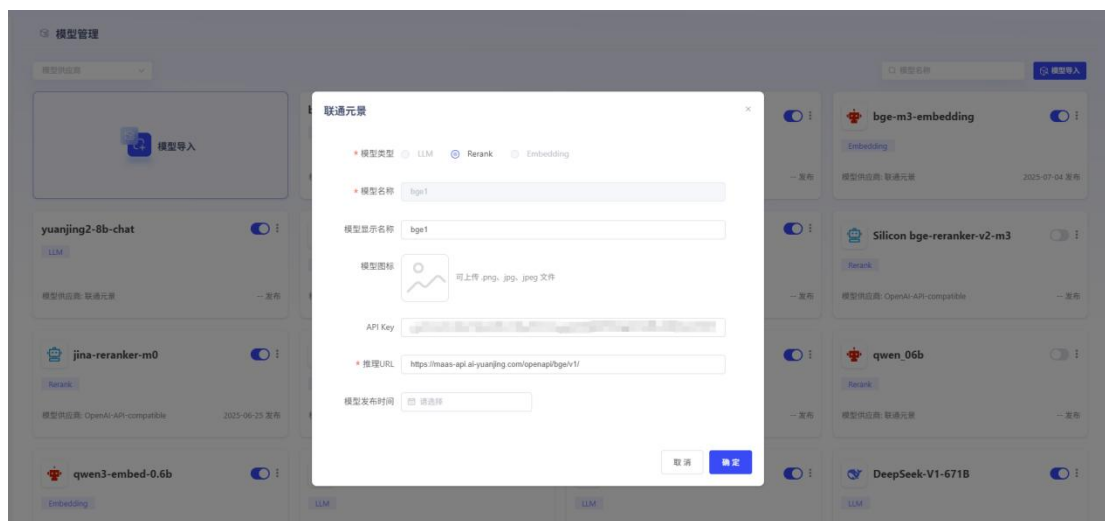
平台操作说明

一、模型管理

用户可通过配置，在平台中接入外部模型。根据模型供应商规范，关联推理 url。支持导入的模型包括 LLM、Embedding 和 Rerank，开启服务后，可用于模型推理。







模型导入样例：

```
curl--location 'https://maas.ai-yuanjing.com/openapi/compatible-mode/v1/chat/completions' \
--header 'Content-Type: application/json' \
--header 'Accept: application/json' \
--header 'Authorization: Bearer sk-abc*****xyz' \
--data '{
  "model": "yuanjing-70b-chat",
  "messages": [{
    "role": "user",
    "content": "你好"
  }]
}'
```

【模型名称】必须为 curl 中可以正确请求的 model；例如

yuanjing-70b-chat

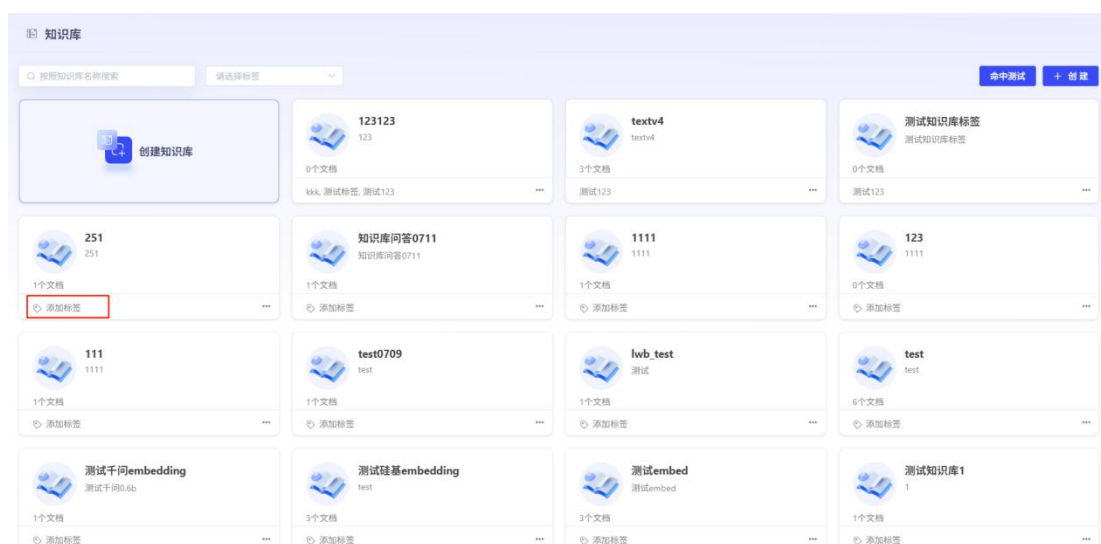
【API Key】必须为上述 curl 中可以正确请求的 key；例如
sk-abc*****xyz（注意不填 Bearer 前缀）

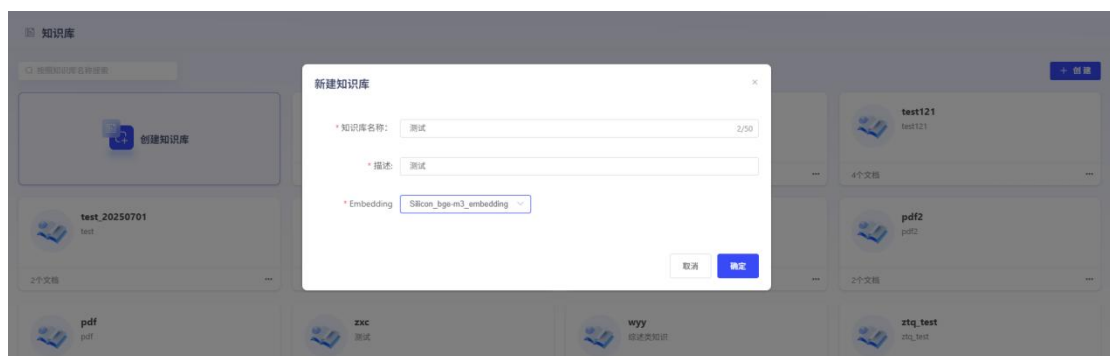
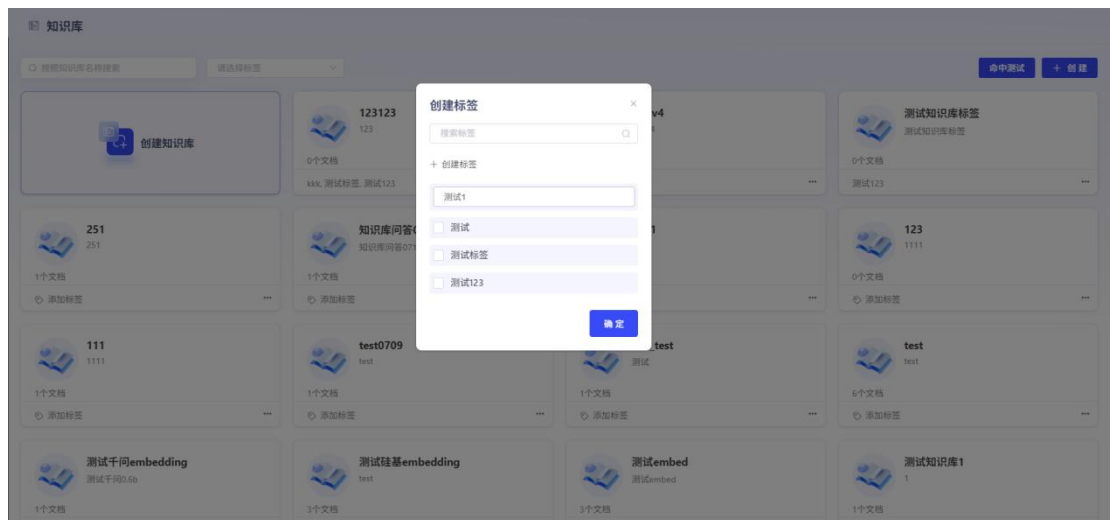
【推理 URL】必须为上述 curl 中可以正确请求的 url；例如
https://maas.ai-yuanjing.com/openapi/compatible-mode/v1（注意不带
/chat/completions 或 /embeddings 或 /rerank 后缀）

二、知识库

1、创建知识库

若用户要使用 RAG 功能，需要先创建知识库，关联 Embedding 模型。其中 Embedding 模型，需提前在模型管理模块上传。用户也可进行标签设置，方便进行知识库筛选分类。（已创建但未填写内容的标签，可通过 Backspace 键快捷删除）





创建知识库：

- 1) 文件上传：平台支持用户上传本地文件或者从 url 上传。



新增文件

1文件上传

2参数设置

从文件上传

url文件上传

url单条上传

将文件拖到此处，或点击上传

*批量上传支持.xlsx格式，仅可上传1个。文档最多可添加100条url，文件不超过15mb

[模板下载](#)

*当前内容不自动更新

urlImport_template.xlsx

11.20KB

✓

✕

下一步

序号	地址
1	https://baike.baidu.com/item/%E5%B7%B4%E8%A5%BF/54222?fromModule=lemma_search-box
2	https://baike.baidu.com/item/%E9%95%BF%E5%BE%81%E5%9B%9B%E5%8F%B7%E4%B9%99/S3701976?fromModule=home_hotspot

新增文件

1文件上传

2参数设置

从文件上传

url文件上传

url单条上传

本次最多可添加10条

* url地址

https://baike.baidu.com/item/%E9%95%BF%E5%BE%81%E5%9B%9B%E5%8F%B7%E4%B9%99/S3701976?fromModule=home_hotspot

✓

+ 追加1条url

开始解析

下一步

2) 参数配置：平台支持自动分段与自定义分段 2 种模式。自定义分段，用户可设置分段时的标点符号、可分割最大值、可重叠值。

新增文件

1文件上传

2参数设置

分段方式配置：☐ 自动分段 ☒ 自定义分段

* 标点符号：

中文双号

6

按照所选的标识符切分文本。切分后，按设置的切片最大长度对切分后的文本组合成单一切片内容

* 可分割最大值：

200

可填写范围最小为200，最大为500

* 可重叠值：

0.2

解析方式：

☒ 文本提取

上一步

确定

文档处理状态：

1) 处理状态：上传完毕的文档可在文档列表中查看。文档内容的解析切分状态可在当前状态列中查看。当状态为“处理完成”，该文档知识即可在后续 RAG

使用知识库中生效。



2) 查看分段结果：点击操作列的“查看”，即可查看状态为处理完成的文档分段结果。

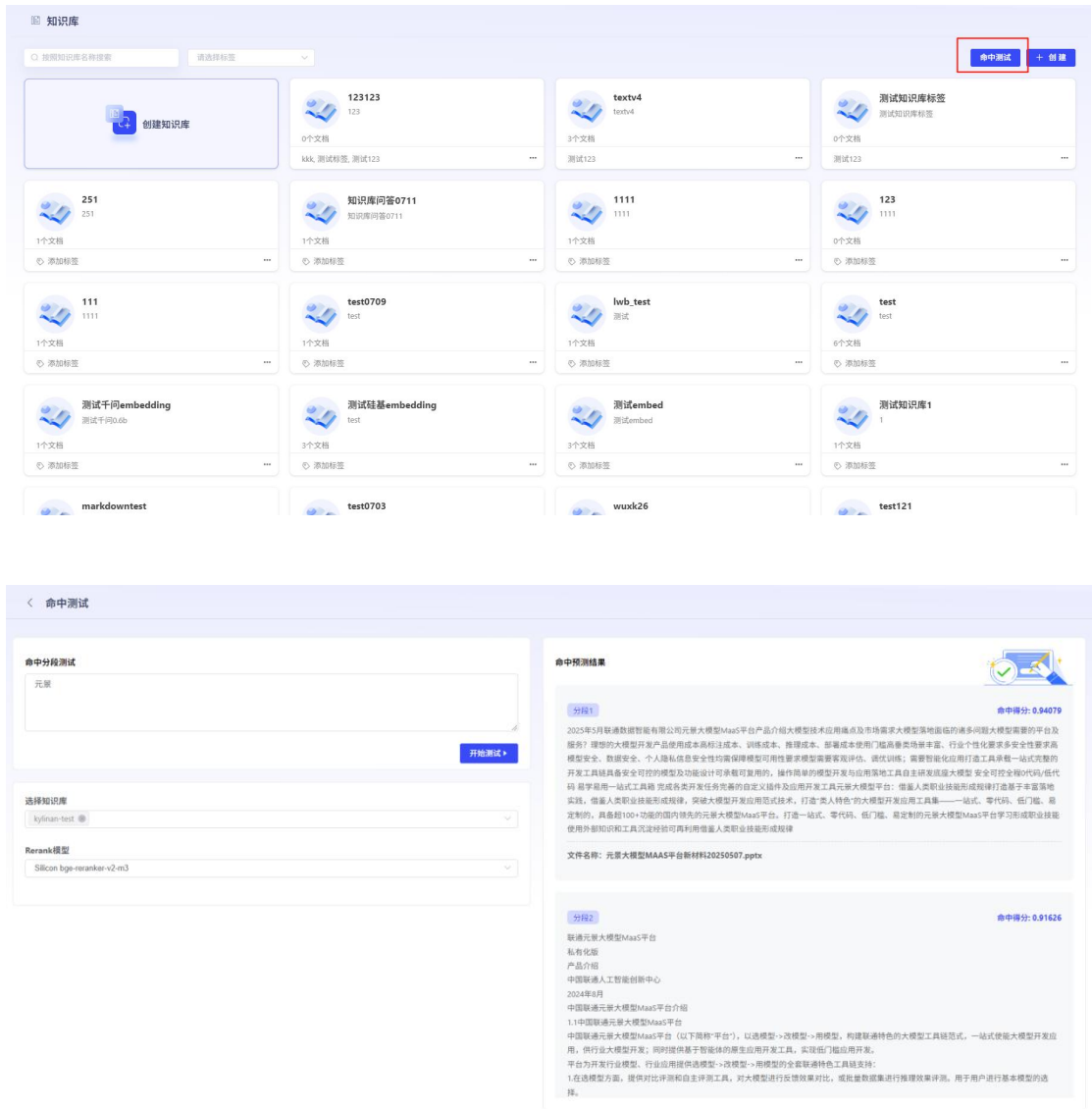


在分段结果查看中，可以查看文档分段策略，及每一个分段的内容结果。可根据用户需求，点击每个分段卡片上的启停开关，对单独分段进行启动和停止。也可点击一键全启动或一键全停用，对整个文档的分段结果进行启动和停止。

注：停止的分段内容在使用 RAG 能力时不会生效。如需查看切段完整信息，点击分段卡片，可展示该分段的详情。

2、命中测试

当用户需要测试知识库是否生效时，可使用“命中测试”功能快速测试。点击“命中测试”按钮，进入命中测试界面。在“命中测试分段”中输入关键词，勾选想要测试的知识库，关联 Rerank 模型。如命中结果，即可获得相关命中得分，及分段内容，确认该知识分段已生效。

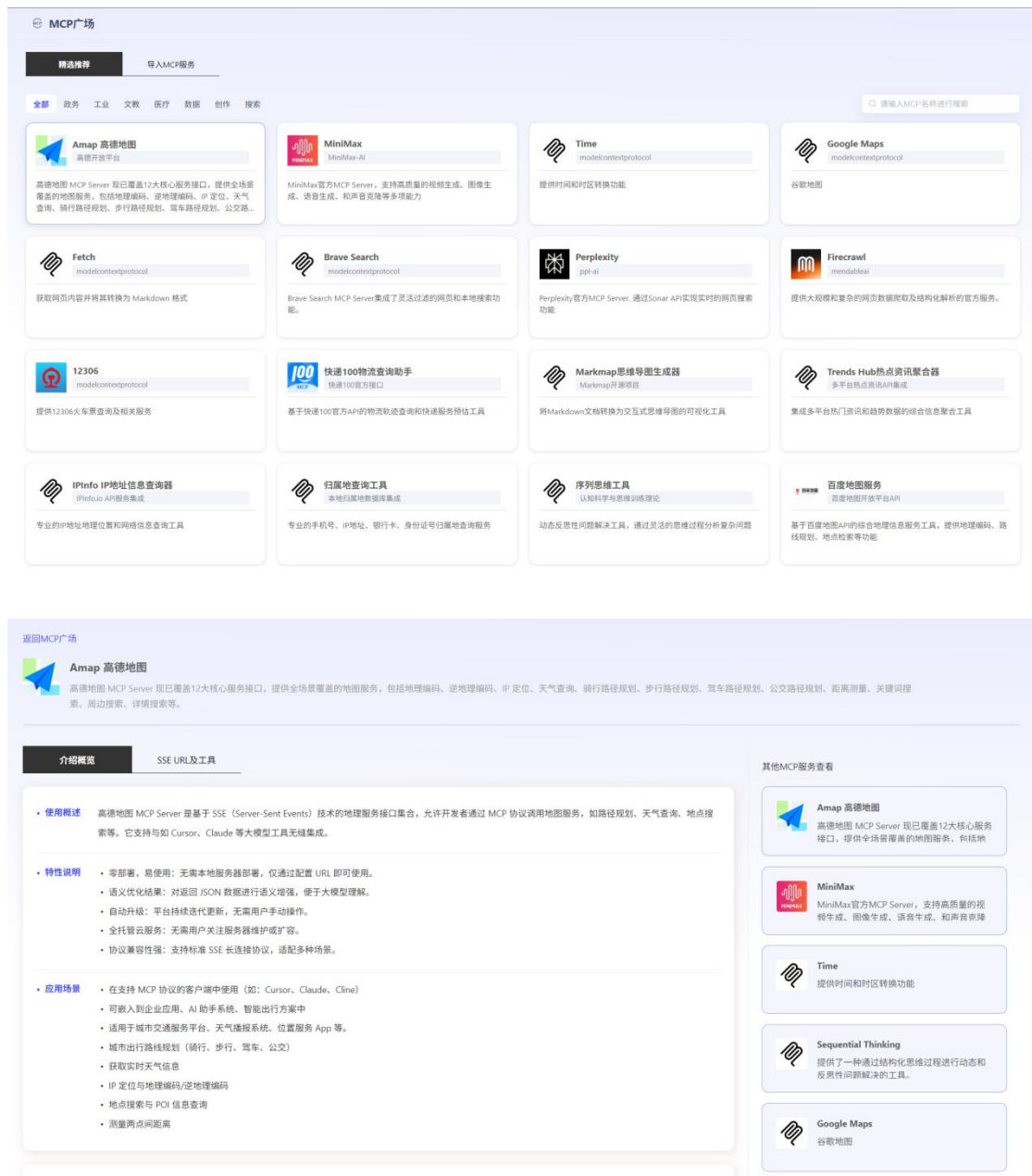


三、MCP 广场

MCP 广场集合纳管 MCP server，并提供精选推荐供用户选择，即选即用。

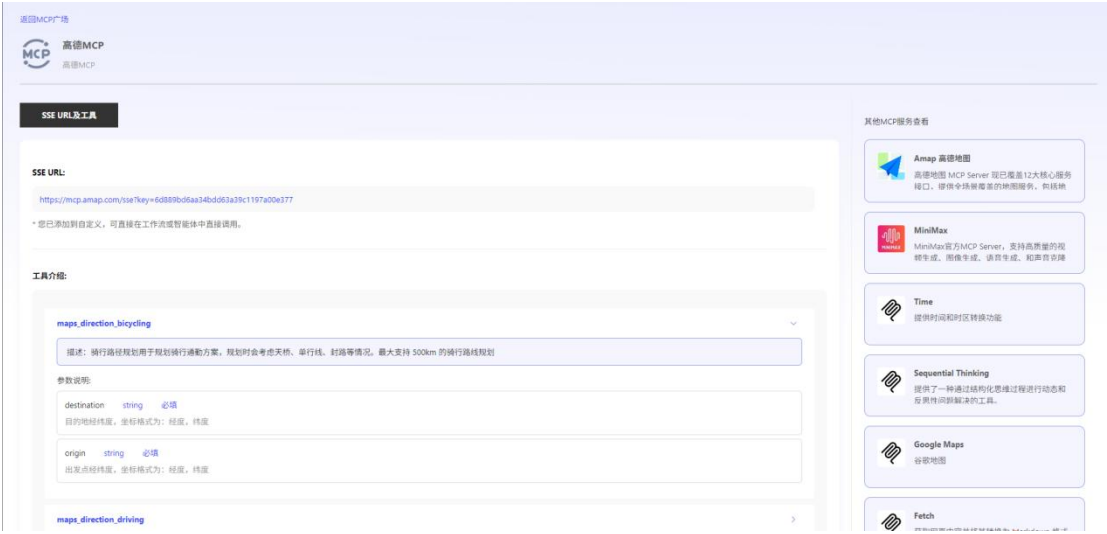
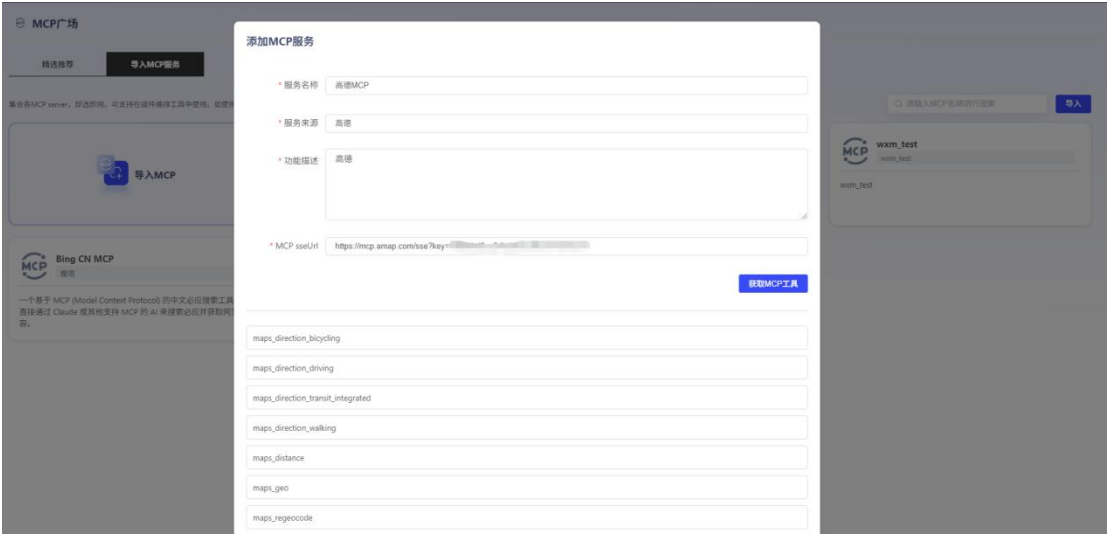
1、精选推荐

平台提供优选 MCP server，用户点击“导入 MCP 服务”，在编辑界面将 url 替换为自己的链接后使用。从精选推荐导入的 MCP server，可在“导入 MCP 服务”模块统一查看。



2、导入 MCP 服务

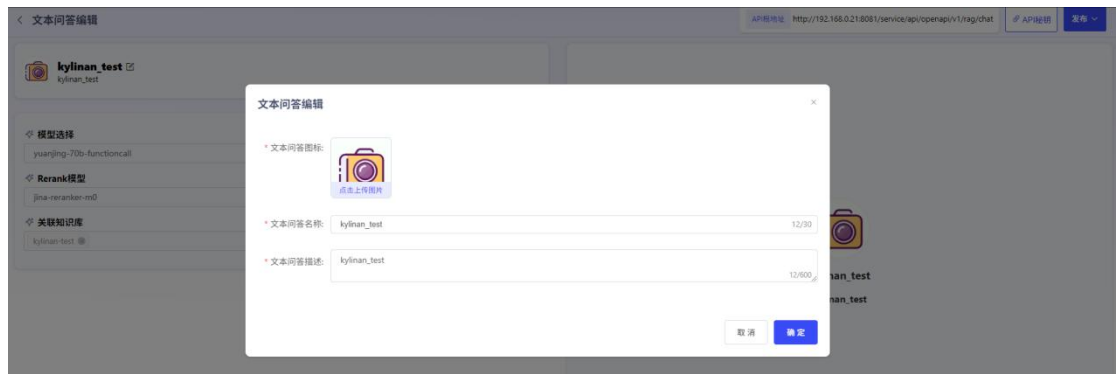
点击“导入”，关联自己的 MCP，可支持在工作流中使用。用户需填写服务名称、服务来源、功能描述、MCP ServerURL。点击“获取 MCP 工具”，获取成功后，将显示可用接口。点击“确认发布”，即可在平台上查看每个工具的详细内容。



四、文本问答

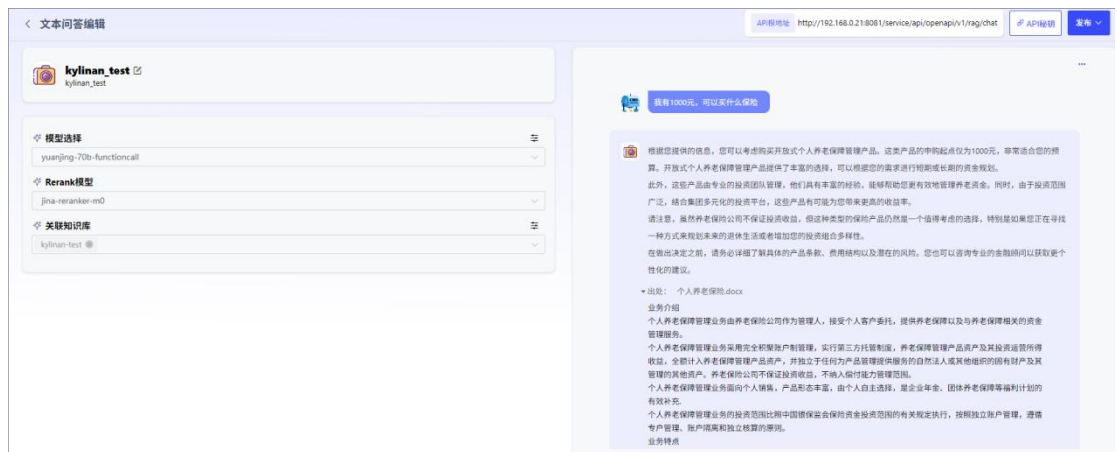
1、文本问答创建

点击“创建文本问答”即可创建文本问答应用。用户可自行设定文本问答图标、名称、描述。



2、文本问答编辑

可通过选择大模型、rerank 模型、知识库，进行文本问答。问答将局限于知识库内，并给出相应出处。



3、文本问答发布

编辑完毕的文本问答应用，点击“发布”可进行发布方式选择，用户可进行私密发布，也可进行公开发布。

私密发布：发布后仅对自己可见，可在“探索广场” - “私密发布的”查看。

公开发布：发布后可对全部用户进行共享，所有用户可在“探索广场” - “全部”查看。

API根地址

http://192.168.0.21:8081/service/api/openapi/v1/rag/chat

API密钥

发布

☐

私密发布：仅自己可见

☒

公开发布：组织内可见

保存

已发布的文本问答也可取消发布后，重新进行编辑。



kylinan_test

kylinan_test

删除

取消发布

2025-06-23 16:43:44

私密

...

4、生成 API

平台已针对应用封装了 API，可点击“API 密钥”生成应用的专属 API-Key 授权，供用户进行调用。

API密钥

×

密钥	创建时间	操作
ww-d9ffc36ccfa2427f829c81cace62f674	2025-06-19 15:58:52	<div>复制</div> <div>删除</div>

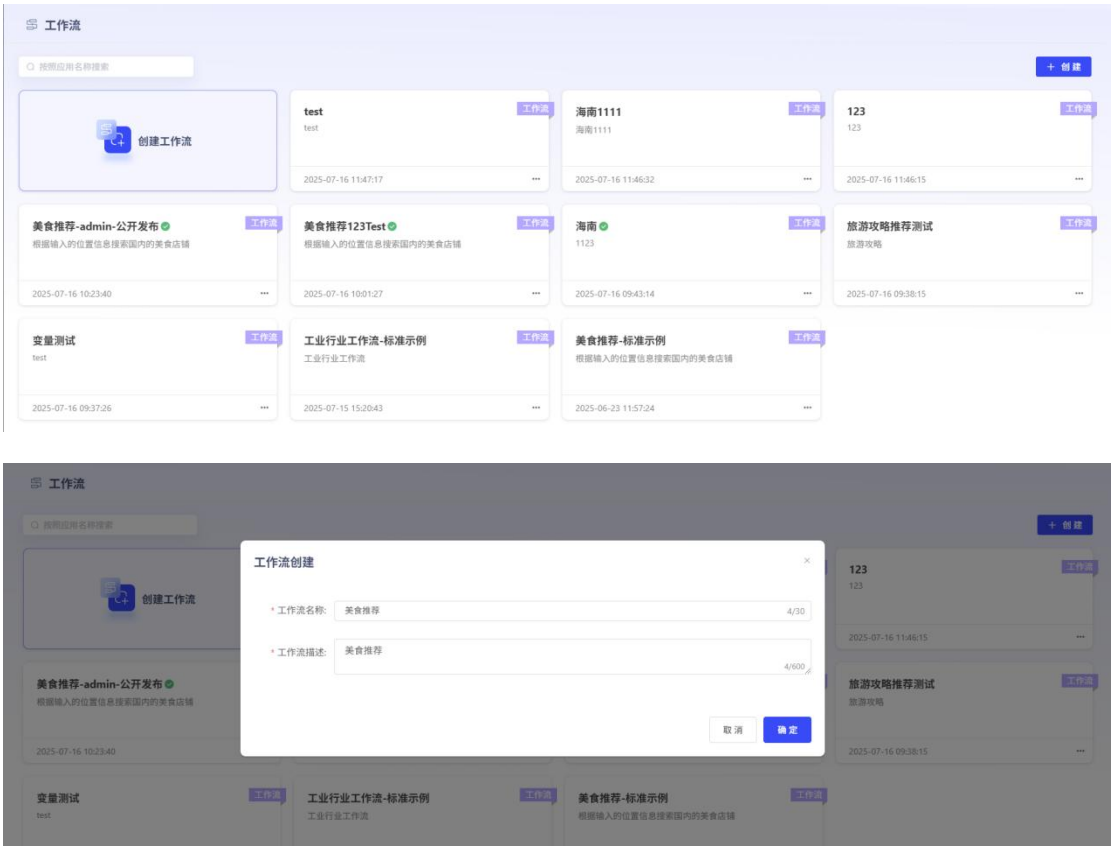
取消

创建

五、 workflows

1、 workflows 创建

点击“创建工作流”即可进入 workflows 创建界面。用户可自行设定 workflows 名称、 workflows 描述描述。平台内置了标准 workflows 示例，用户也可直接复制使用。



2、 workflows 编辑

平台提供 MCP、意图识别、API、代码、大模型、分支器、知识库、文档生成、文档解析节点。

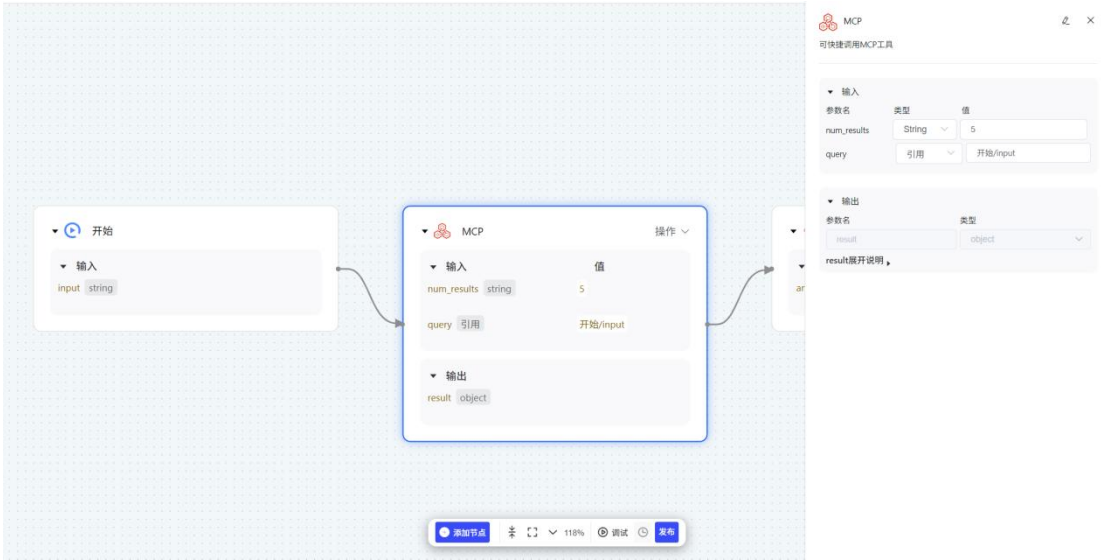
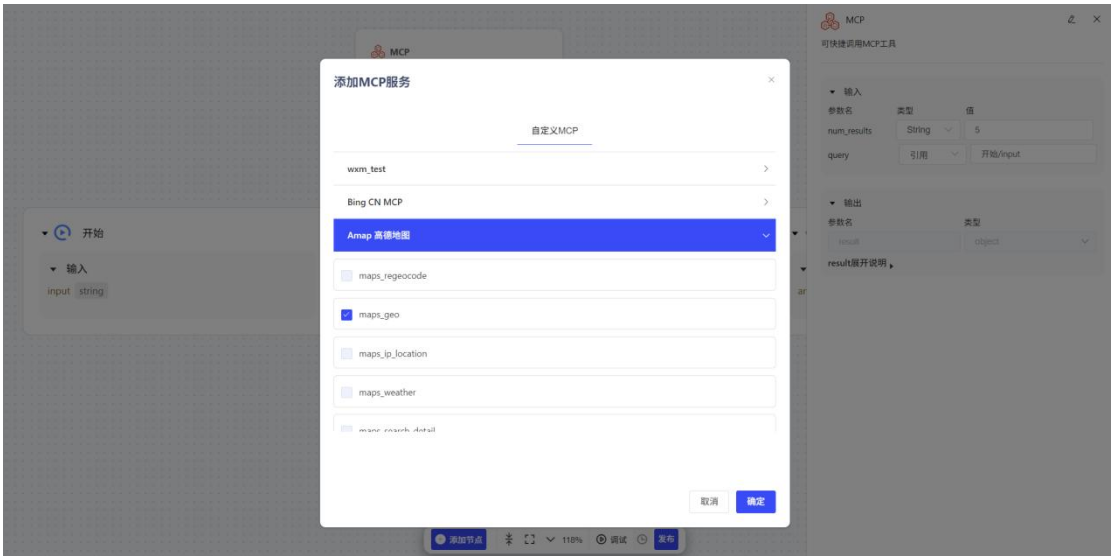
开始

“开始” 节点不可以被复制或删除。定义所需的输入参数，会在插件被应用调

用时，由思考模型根据参数描述从用户输入的原始内容中抽取并传入。

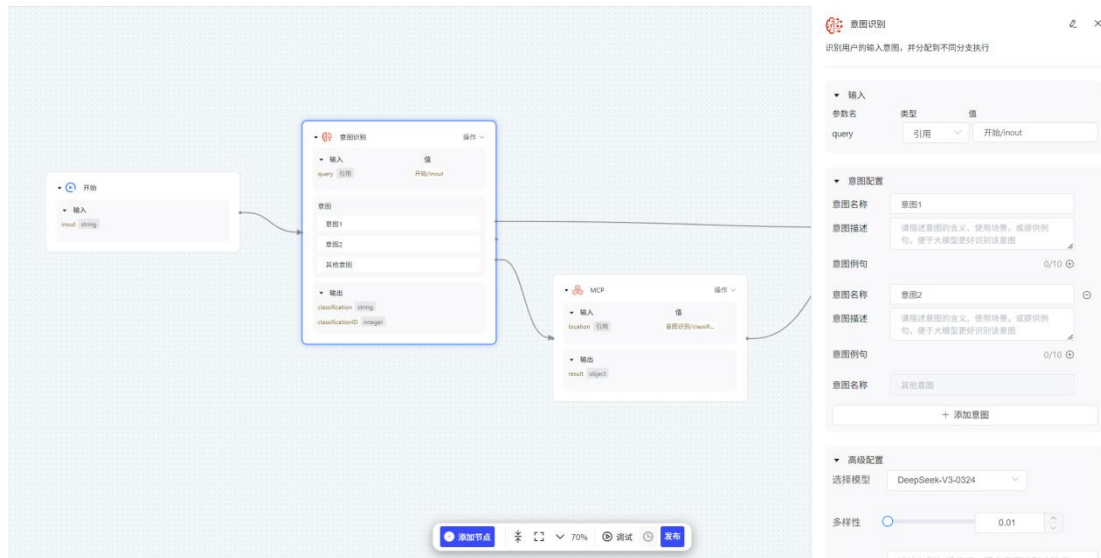
MCP

调用 MCP server 的能力。可添加已发布在 MCP 广场中的 MCP server，并关联对应接口生成结果。



意图识别

识别用户的输入意图，并分配到不同分支执行。



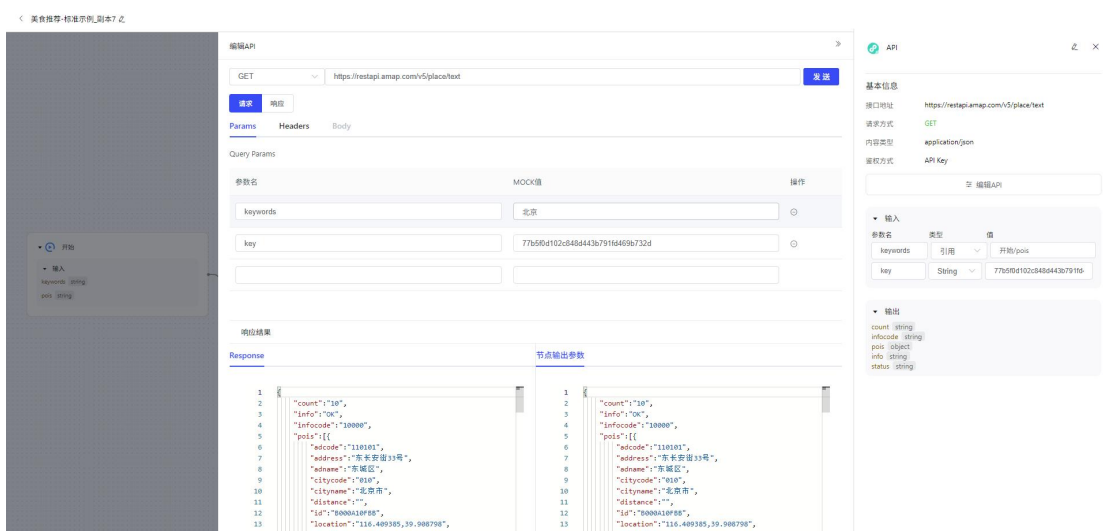
API

支持接入遵守 RESTful 架构规范并采用标准认证机制的 API。API 接入前需要已经完成服务化，确保所有接入的 API 均遵循 RESTful 架构规范，确保使用无状态的请求模型并通过 HTTP 标准方法 (如 GET、POST)进行资源操作，同时接口使用标准的认证机制。此外，接口设计符合 OpenAPI 规范，接入的 API 已经过性能测试、安全审查和兼容性检测，以确保 API 的稳定性和可靠性。

使用说明：

I.添加 API 节点，可以手动逐步配置 API 基本信息。配置信息包括：请求参数和响应参数。您可不配置响应参数，直接根据 API 请求结果的返回信息自动解析输出参数，平台解析规则说明：当前支持接口返回内容类型有：JSON、XML、HTML、Plain Text、YAML、CSV。其中 JSON、XML、YAML 可被解析为 JSON 输出，便于您在后续节点引用某个参数或属性，其余类型支持以整个 string 类型输出。

ii.配置完成请求参数和响应参数后，点击【发送】，进行接口请求，下方将展示对应接口的原始 API 返回信息，和按照响应参数配置获得的参数返回信息，您可以检查返回信息是否符合预期。

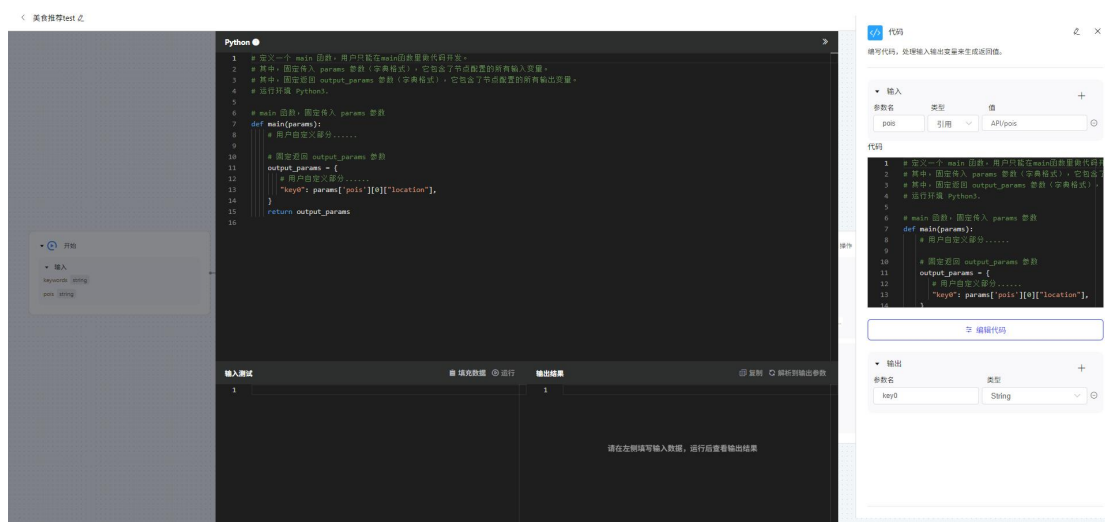


iii.在完成 API 原始信息编辑后，您需要继续在右侧面板中对输入参数进行值配置，可以引用前序节点输出，或手动输入对应参数值。需要确保引用类型与设置类型一致。

API 节点的 http 请求发送成功即为接口调试通过，在整体流程调试前，API 节点需要为调试通过状态。

代码

在代码节点中，可通过编写代码实现自定义的处理功能。引入代码节点到画布中，可在编辑器（IDE）内看到默认的示例代码。



编辑器的使用：

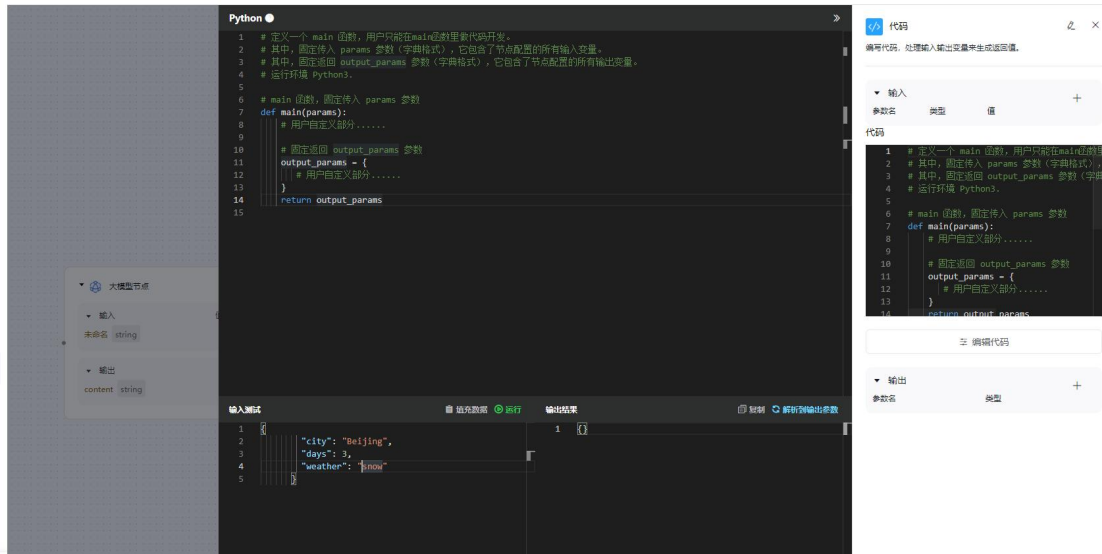
引用输入：配置了输入参数名和参数值后，可以在编辑器中引用输入参数。编辑器引入输入参数时，需要通过字典变量 params 引入代码节点的输入参数。代码节点支持不设置输入参数，但是设置的输出参数需要配置对应的默认值或参数引用关系。

返回输出：需要在编辑器中定义一个字典变量，作为编辑器中函数的输出。代码节点的输出参数是该字典变量的键（key）。

运行时环境：编辑器支持的运行时环境为 Python 3 。运行环境预置了 NumPy、bs4 包。当前运行环境暂不支持进行 request 请求或获取文件。

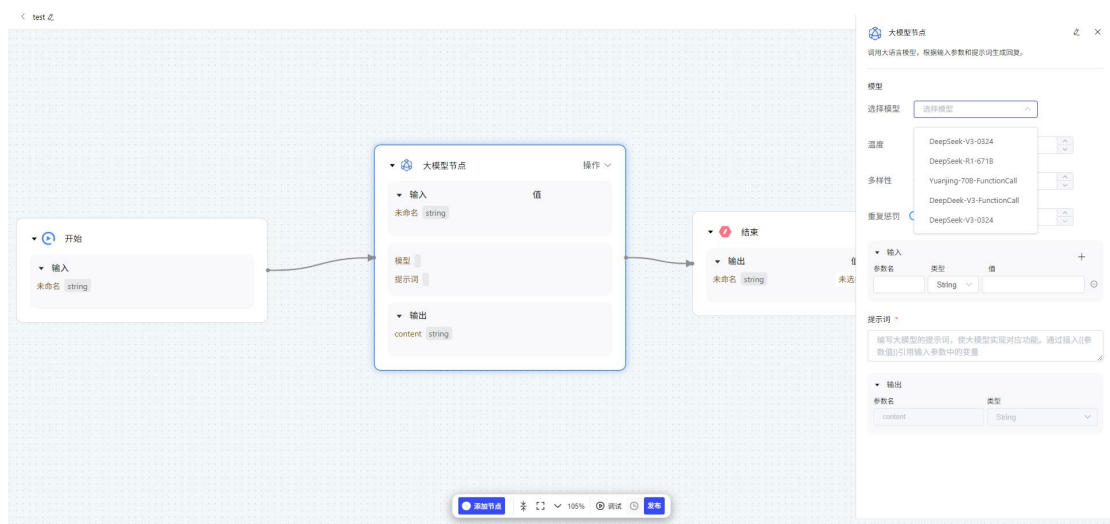
编辑器测试功能：

输入测试：在输入测试区域可以输入测试数据，并进行调试运行。“填充数据”功能可以根据当前输入参数类型，生成输入数据。



大模型

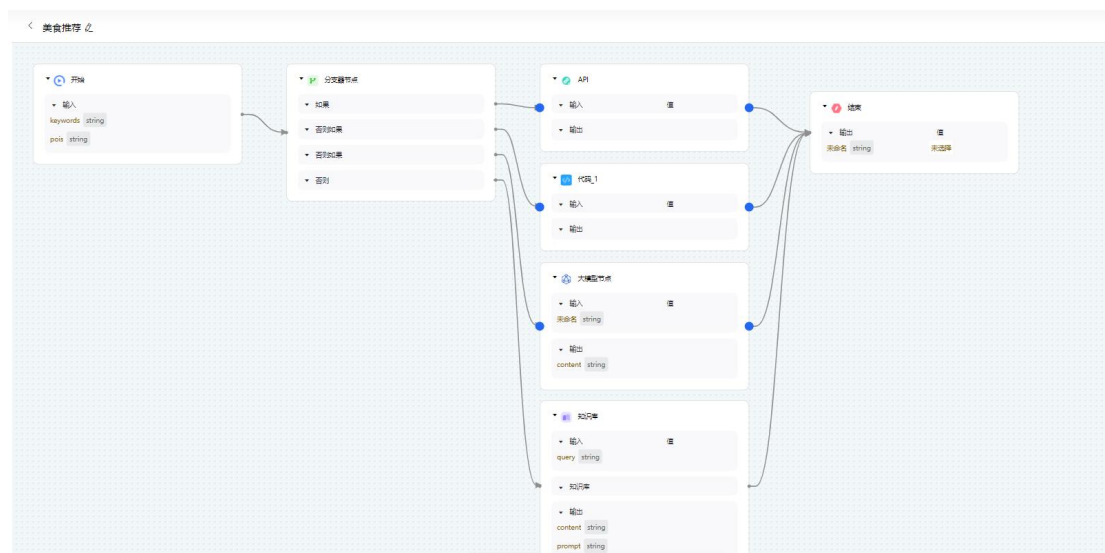
大模型节点可调用大语言模型，根据输入参数和提示词生成回复。



分支器

分支器节点用于设计 workflow 内的分支流程，可以连接多个下游节点。当向该节点输入参数时，节点会从上到下依次判断是否符合条件，若设定条件成立则运行对

应的条件分支，若均不成立则运行“否则”分支。可通过拖拽分支条件配置面板来设定分支条件的优先级。在每个分支条件中，支持选择判断关系（且/或），以及同时添加多个条件。



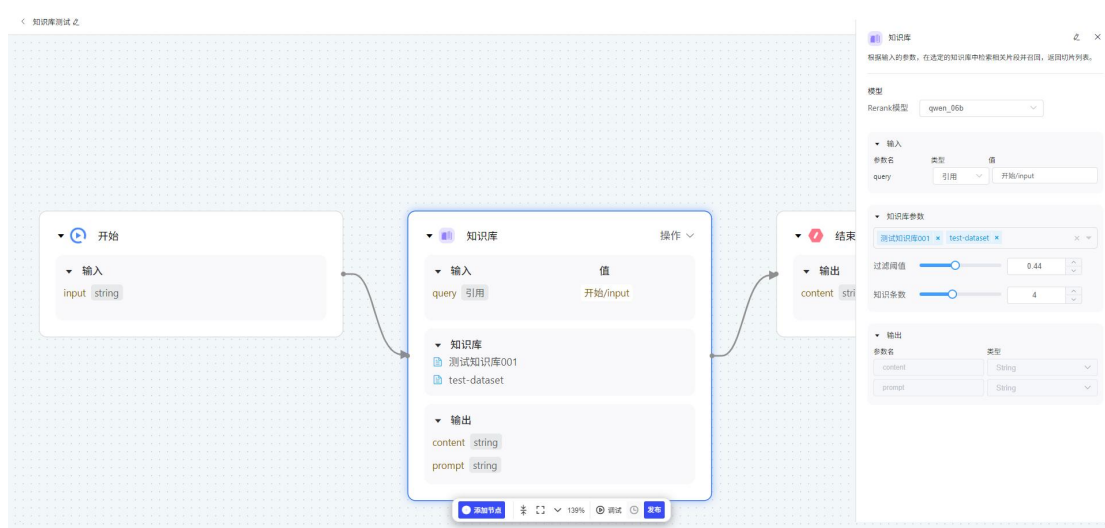
知识库

知识库节点支持根据输入的 query，在选定的知识库中检索相关片段并召回，返回切片列表。你可以上传文件并建立知识库，在知识库节点中勾选想要使用的知识库进行检索。知识库节点需选择一个 Rerank 模型后，才能生效。

输入参数：参数名不可修改，参数类型为 string，上级节点的输出参数会强制转换为 string 类型作为知识库节点的输入，输入参数有两种类型：1) 引用类型为引用上一个节点的输出变量，2) 常量类型，可以输入一个 string 类型的入参。

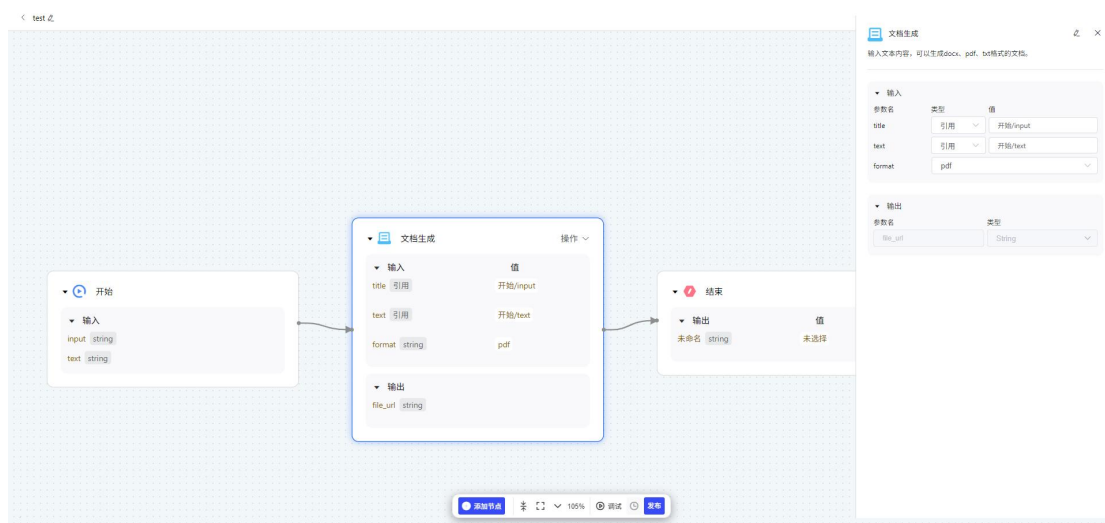
选择知识库：选择需要检索的知识库，支持选择多个知识库。

输出参数：在知识库中检索输出的变量信息及变量类型。



文档生成

输入文本内容，可以生成 docx、pdf、txt 格式的文档。



文档解析节点

输入 txt、pdf、docx、xlsx、csv、pptx 等格式文档的 URL，可以解析提取出文档的文本内容。



结束

结束节点是组件的最终节点，输出组件运行后的最终结果。该结果将输出给调用此工具的应用或下游组件。可以配置输出参数：定义组件需要输出的参数。

3、工作流调试与发布

编辑完毕的工作流，点击“调试”，运行成功后，即可进行发布。点击“发布”可进行发布方式选择，用户可进行私密发布，也可进行公开发布。发布完成的工作流可作为工具，被智能体调用。

私密发布：发布后仅对自己可见，可在“探索广场” - “私密发布的”查看。

公开发布：发布后可对全部用户进行共享，所有用户可在“探索广场” - “全部”查看。



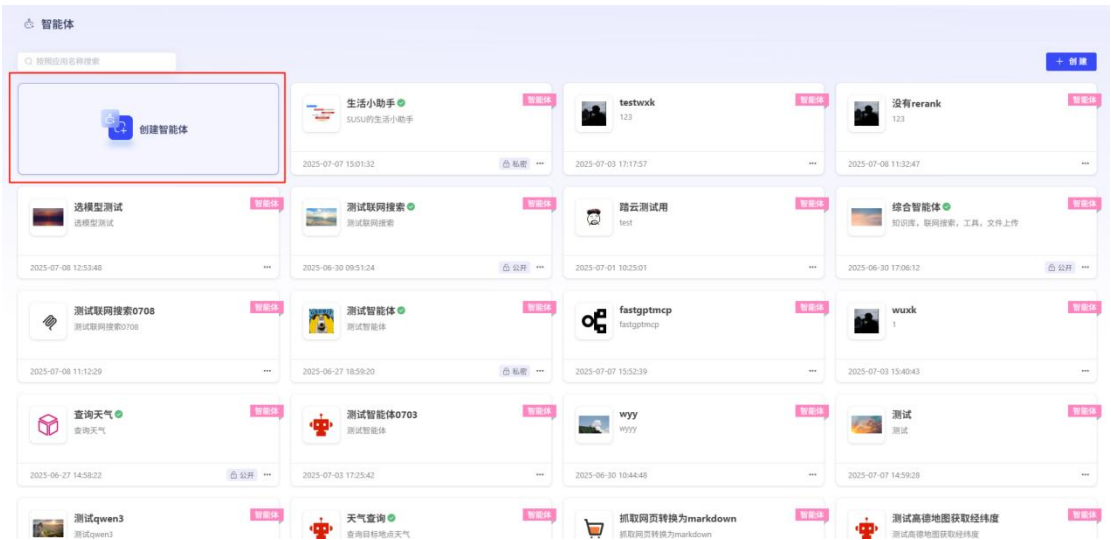
已发布的工作流也可取消发布后，重新进行编辑。



六、智能体

1、智能体创建

点击“创建智能体”即可创建智能体。用户可自行设定智能体图标、智能体名称、智能体描述。



2、智能体编辑

智能体可使用以下几类功能进行应用功能 0 代码开发：

选择模型服务：用户可选择平台中已经纳管的模型，创建智能体。

开场白：用于编辑开场问候语

系统提示词：填写应用功能描述、应用处理流程描述、以及对生成结果的要求。

推荐问题：可设置引导问题

知识库：用户通过上传文档为大模型进行知识库外挂。外挂知识库后，可与大模型交互文档中的内容。知识库需在“工作室” - “知识库”中提前添加。

联网检索：通过配置联网检索的 url 和 key，可启用“博查”网络搜索辅助问答。

工具：用户可添加关联已发布的工作流。用户通过点击“添加”，选择“工作流”，即可添加已发布的工作流。

智能体编辑

API地址: <http://192.168.0.21:8081/service/api/openapi/v1/agent/chat> API密钥 发布

test0624 test0624

智能体配置

模型选择: DeepSeek-V3-0324

开场白: 我说test (6/100)

系统提示词: 描述你想创建的应用, 详细描述应用的详细功能及作用, 以及对该应用生成结果的要求 (0/500)

推荐问题: 你好 (+ 添加)

关联知识库: 请选择关联知识库

Rerank模型: 请选择模型

联网检索

test0624 我说test

你好

请输入内容,用Ctrl+Enter可换行 发送

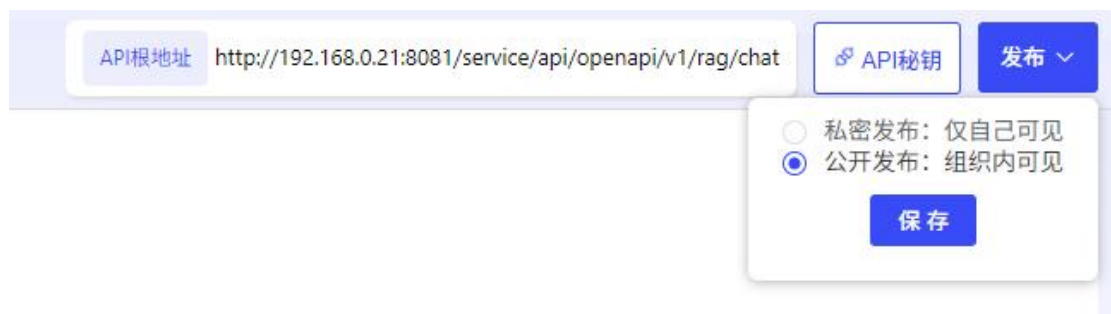
3、智能体发布

编辑完毕的智能体，点击“发布”可进行发布方式选择，用户可进行私密发

布，也可进行公开发布。

私密发布：发布后仅对自己可见，可在“探索广场” - “私密发布的”查看。

公开发布：发布后可对全部用户进行共享，所有用户可在“探索广场” - “全部”查看。



The image shows a web interface for configuring an API. It includes a text field for the 'API根地址' (API Root Address) with the value 'http://192.168.0.21:8081/service/api/openapi/v1/rag/chat'. To the right is a button labeled 'API密钥' (API Key). Further right is a dropdown menu labeled '发布' (Publish). Below the dropdown, there are two radio button options: '私密发布：仅自己可见' (Private publish: only visible to self) and '公开发布：组织内可见' (Public publish: visible within the organization). The 'Public publish' option is selected. A '保存' (Save) button is located below these options.

已发布的智能体也可取消发布后，重新进行编辑。



The image displays a card for an AI agent named '高考填报专家' (Gaokao Zhanbao Zhuanjia), which has a green checkmark indicating it is verified. The card features a blue icon with a white feather and the text '高考志愿填报' (Gaokao Zhanbao Zhuanjia). Below the name, it says '高考填报助手依据高考成绩、位次、兴趣及' (Gaokao Zhanbao Zhuanjia assistant based on Gaokao scores, ranks, interests, and...). A context menu is open on the right side of the card, showing two options: '删除' (Delete) and '取消发布' (Cancel publish). At the bottom of the card, the timestamp '2025-07-09 22:23:22' is visible, along with a lock icon and the word '私密' (Private), followed by a three-dot menu icon.


4、生成 API

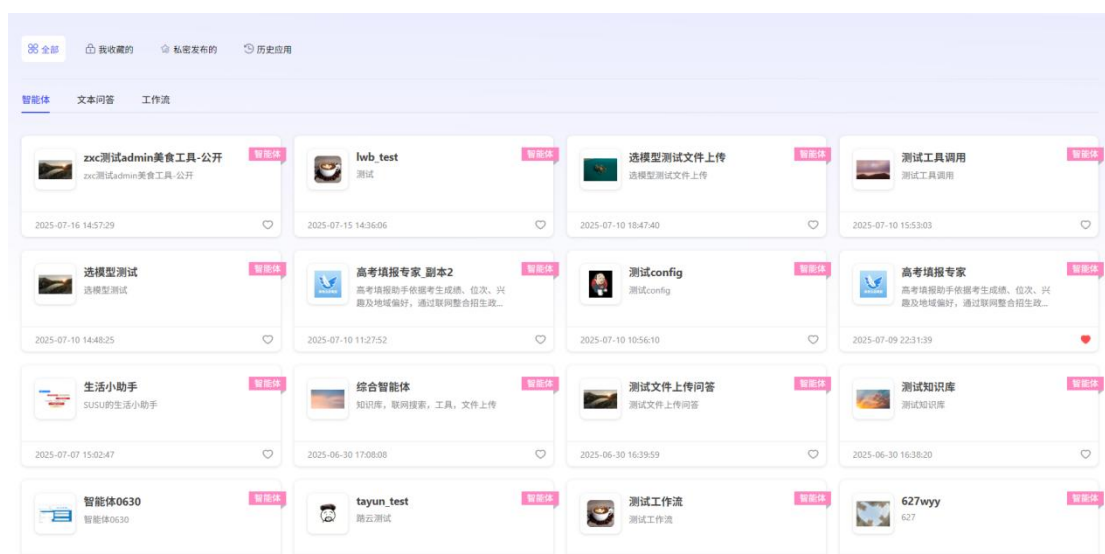
平台已针对应用封装了 API，可点击“API 密钥”生成应用的专属 API-Key 授权，供用户进行调用。



七、应用广场

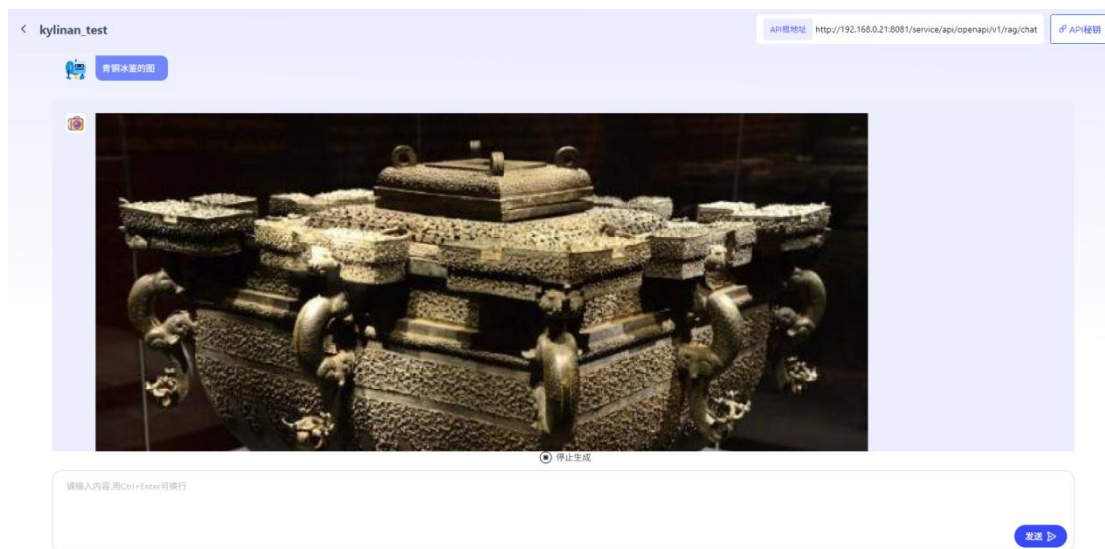
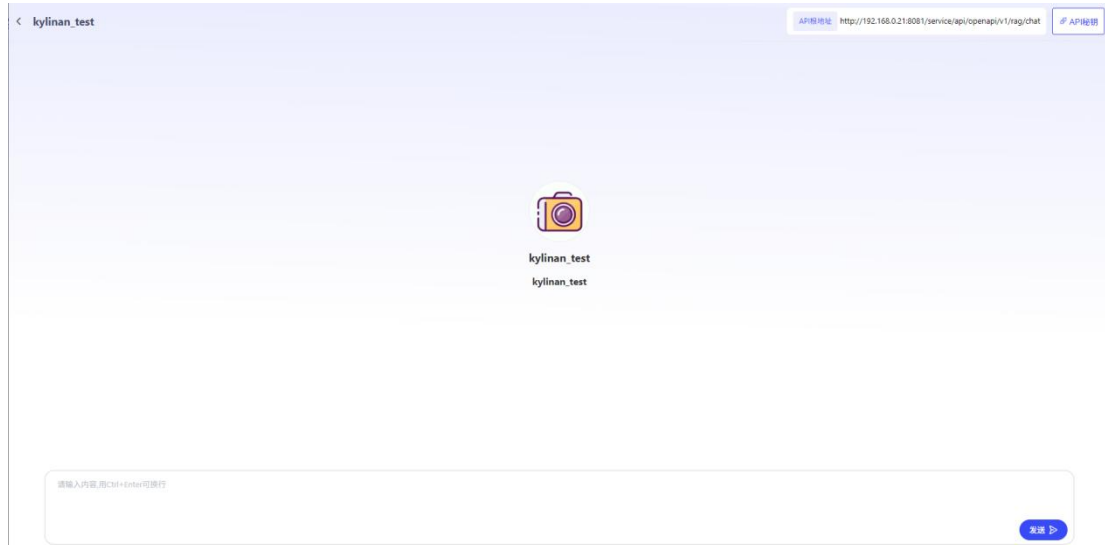
1、应用体验

支持用户在探索广场上使用已发布的应用。包括其他用户公开发布的应用以及自己私密发布的应用。对于常用应用，可以点击  进行收藏，方便下次使用。



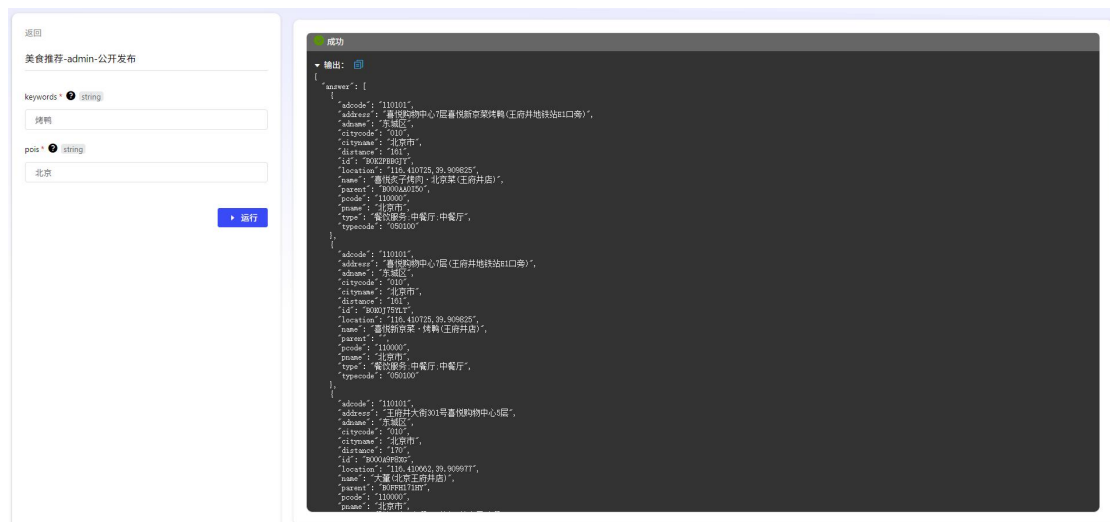
文本问答体验

用户点击文本问答卡片,可在线使用文本问答,并可根据 API 接口进行调用。
具体调用方式详见接口文档。



工作流体验

用户点击工作流卡片，可在线体验工作流。



智能体体验

用户点击智能体卡片，可在线使用智能体，并可根据 API 接口进行调用。具体调用方式详见接口文档。





2、历史应用

可通过历史应用查看历史使用过的全部应用。智能体可查看历史对话内容，可在历史对话窗口下继续交互提问，也可以新建对话再次使用。对于文本问答和工作流应用，不记录历史对话内容，仅支持在线使用。

八、账号

1、个人信息

用户可在个人信息界面，修改登录密码。



2、组织管理

此模块可进行组织、用户、角色的统一管理。

1) 组织管理

组织查看

管理员可对该组织内所有子组织情况进行查看，并可编辑、删除、停用和新增子组织。

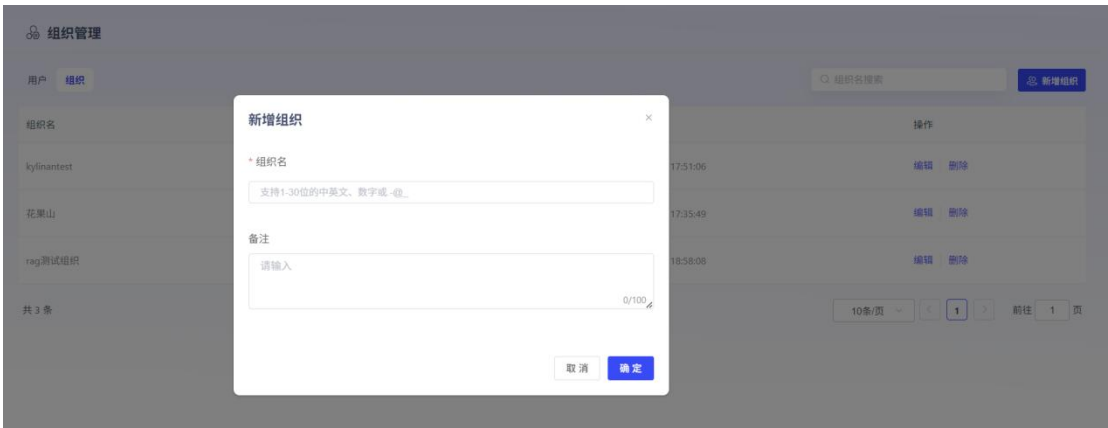
点击“编辑”，即更改子组织名称。



新增组织

点击“新增组织”，可对组织名进行设置。点击“确认”，即可完成新增子组织。用户创建下级组织时，系统默认在新组织中创建一个例如"组织管理员"角色，默认拥有新组织的所有权限；同时该用户默认加入新组织，并对应该

角色。



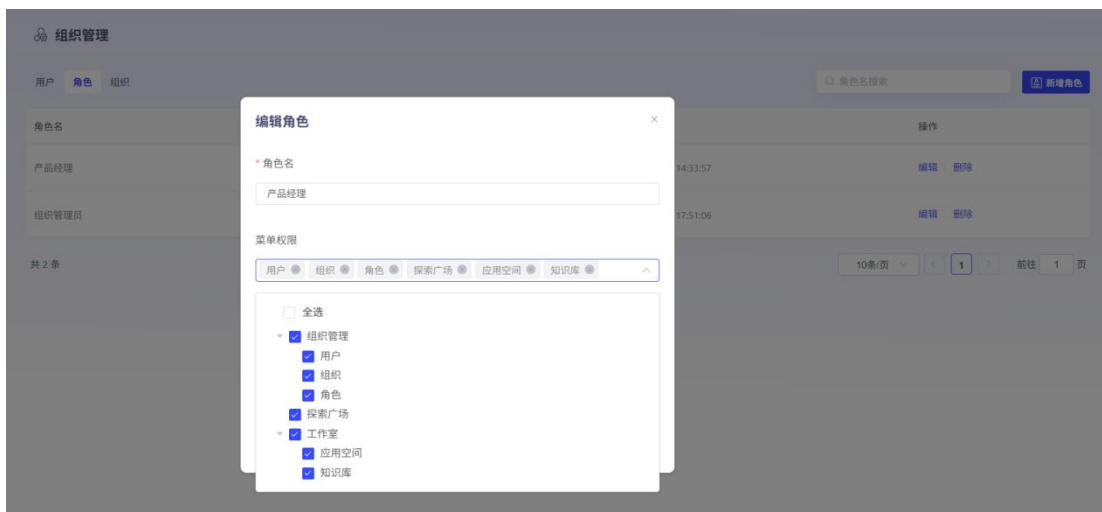
2) 角色管理

角色查看

管理员可对该组织内所有角色情况进行查看，并可编辑、删除、停用和新增角色。其中“组织管理员”角色不可被删除和停用。同一个用户可以在多个组织中。

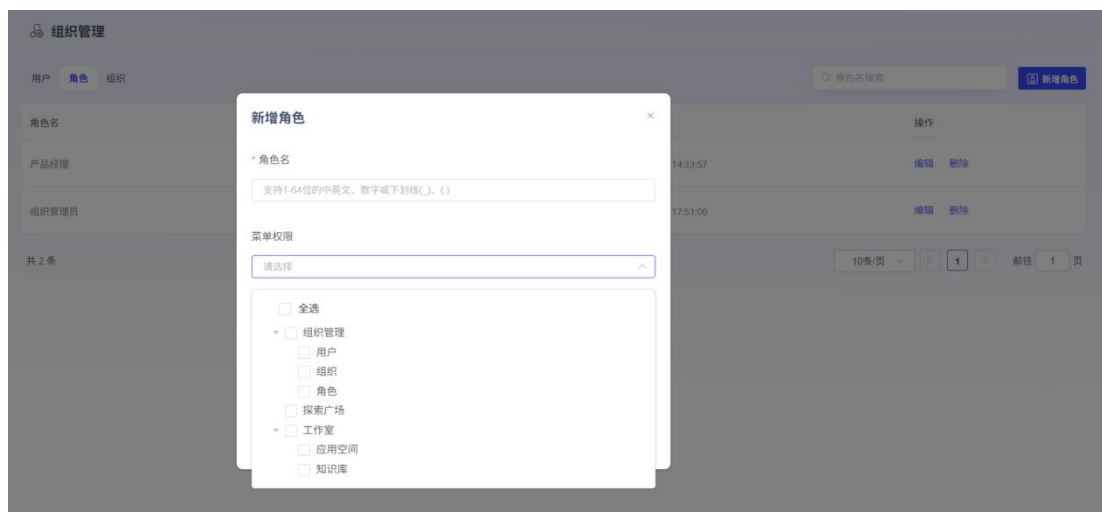
点击“编辑”，即更改角色名称以及菜单权限。





新增角色

点击“新增角色”，可对角色名和菜单权限进行设置。点击“确认”，即可完成新增角色。用户创建的新角色，权限集合不超过用户当前对应角色的权限集合。

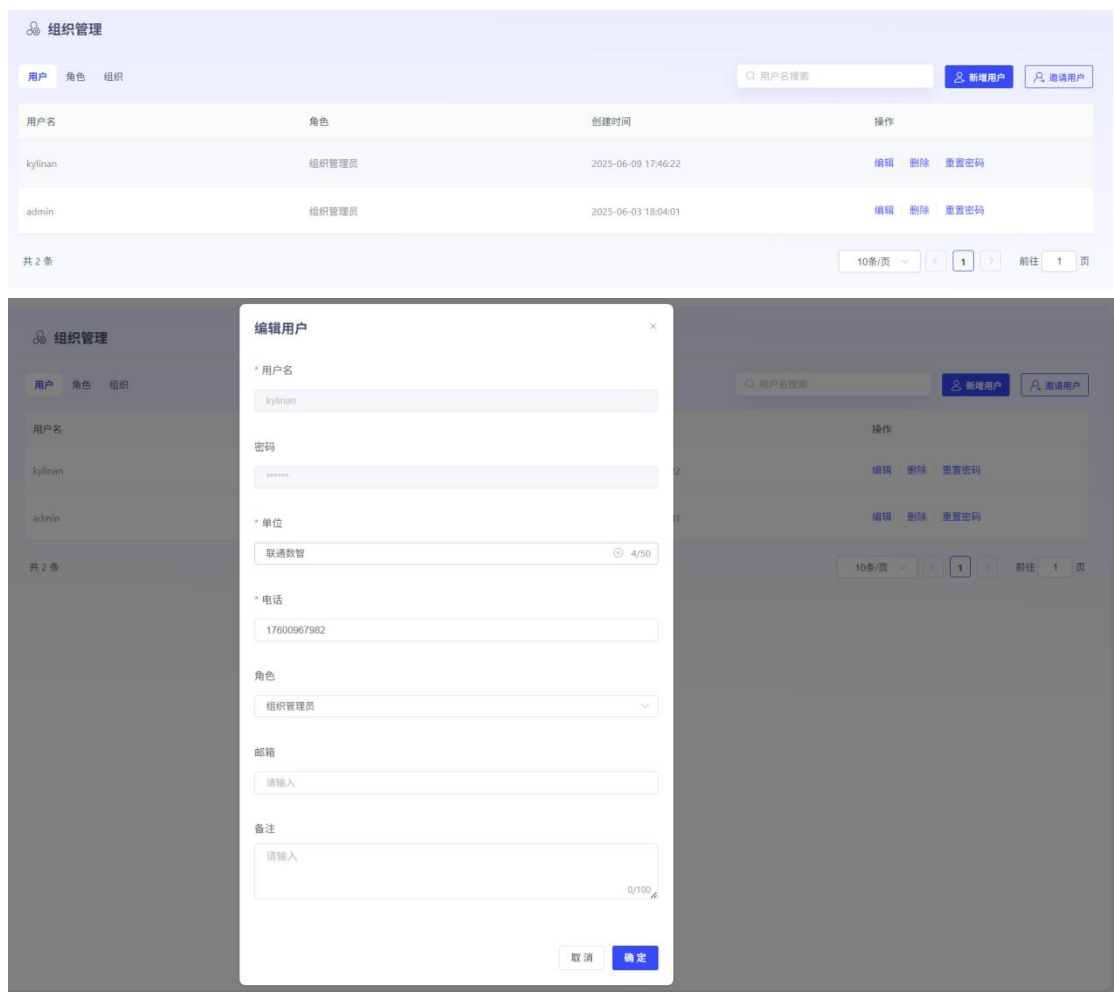


3) 用户管理

用户查看

管理员可对该组织内所有用户情况进行查看，并可编辑、删除、重置密码、新增账户和邀请用户。

点击“编辑”，即更改用户单位、电话、角色、邮箱。

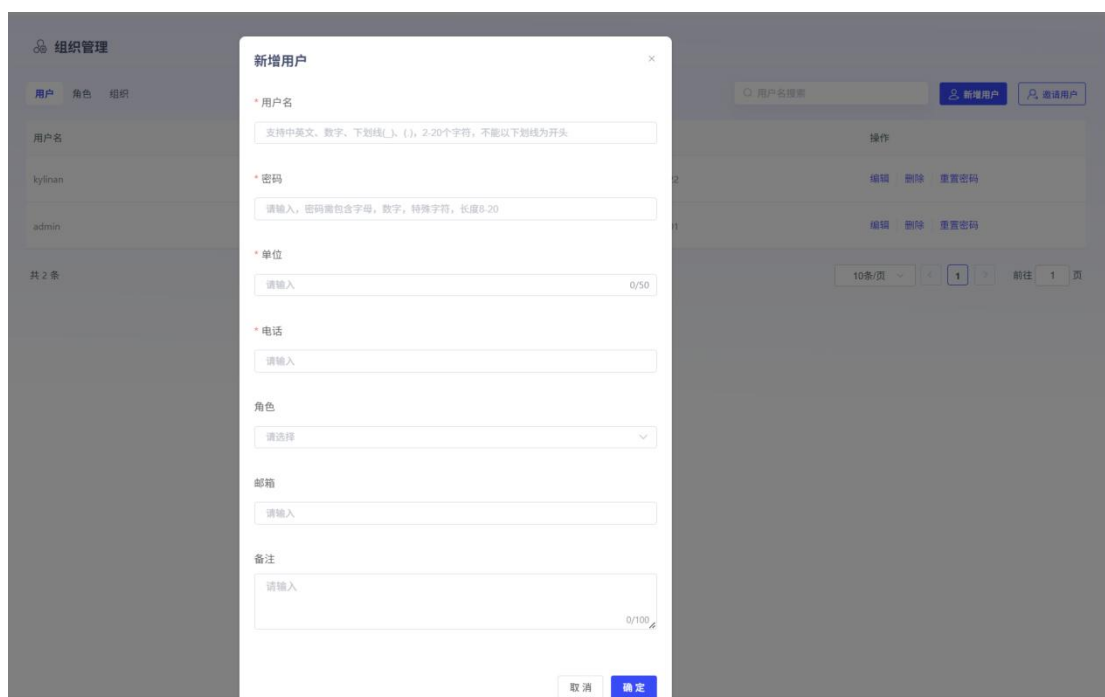


新增账户

点击“新增账户”，可对用户名、密码、单位、电话、角色、邮箱进行设置。

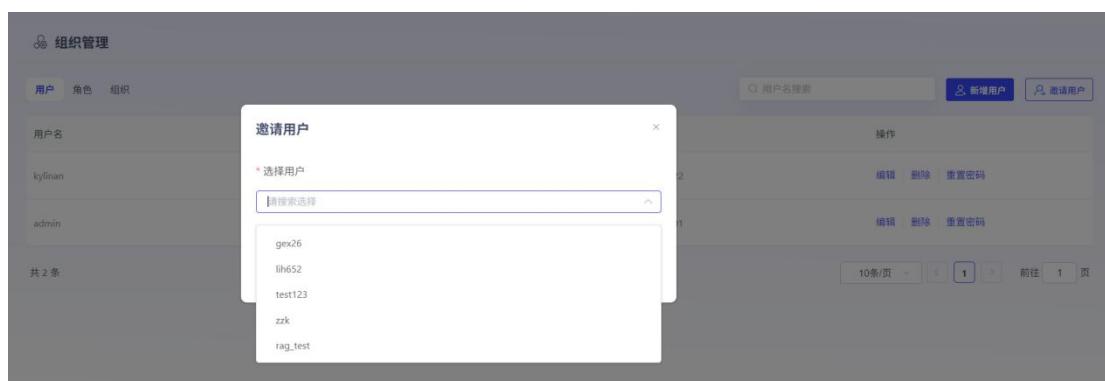
点击“确认”，即可完成新增用户。用户被分配到某个组织，可不对应任何角色。

用户所在组织中，对应的角色有创建用户的权限，则该用户创建的新用户必须默认属于该组织。



邀请用户

点击“邀请用户”，可对特定用户进行邀请。管理员可邀请该系统下任意不在该组织的用户，被邀请的用户自动加入该组织。



3、组织切换

平台支持多租户，用户可进行组织切换，查看不同组织下创建的应用，并进行编辑和使用。



|-- 系统 --



--- 系统 ---

测试组

测试组 - yy

开发组

运营组

Open API

文本问答API

文本生成应用无会话支持，适合用于翻译/文章写作/总结 AI 等等。

请求接口

Type	Instructions
方法	Http
请求URL	按照应用实际API根地址，例如： http://localhost:8081/service/api/openapi/v1/rag/chat
字符编码	UTF-8
请求类型	POST
鉴权方式(header参数)	Authorization: Bearer {API Key}
请求格式	Content-Type: application/json
响应格式	非流式: Content-Type: application/json 流 式: Content-Type: text/event-stream

请求参数

Parameter	Required	Type	Instructions
stream	否	bool	是否以流式接口的形式返回数据，默认为非流式。true为流式，false为非流式。
query	是	string	用户提出的问题或提示语

响应参数

Parameter	Type	Instructions
code	int	状态码，用于表示请求成功或具体的错误类型
message	string	提示信息，通常用于提供关于code的详细解释或请求成功的确认信息
msg_id	string	提示信息ID
data	array	当前响应文本，包含了根据用户输入和知识库搜索得到的答案
history	array	包含之前对话历史的数组，用于上下文管理
finish	int	仅在流式返回中有该字段。表示流是否结束，0：未结束，1：正常结束，2：生成长度导致结束，3：异常结束，4：命中安全护栏结束

data

Parameter	Type	Instructions
output	string	当前响应文本内容片段
searchList	array	知识增强搜索结果

searchList

Parameter	Type	Instructions
kb_name	string	知识库名字
snippet	string	知识内容片段
title	string	文件标题

history

Parameter	Type	Instructions
query	string	请求文本
response	string	模型响应文本

调用示例

流式

```
curl -k --location 'http://localhost:8081/service/openapi/v1/rag/chat' \
--header 'Content-Type: application/json' \
--header 'Accept: application/json' \
--header 'Authorization: Bearer <API Key>' \
--data '{
    "stream": true,
    "query": "请一句话介绍DeepSeek"
}'
```

非流式

```
curl -k --location 'http://localhost:8081/service/api/openapi/v1/rag/chat' \
--header 'Content-Type: application/json' \
--header 'Accept: application/json' \
--header 'Authorization: Bearer <API Key>' \
--data '{
    "stream": false,
    "query": "请一句话介绍DeepSeek"
}'
```

响应示例

流式

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "DeepSeek", "searchList":  
[]}, "history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "是", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "深度求索", "searchList":  
[]}, "history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "推出的", "searchList":  
[]}, "history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "AI", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "工程化", "searchList":  
[]}, "history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "平台", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "提供", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "模型", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "纳", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "管", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "、", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "工作", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "流", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "编排", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "、", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "知识", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "库", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "管理等", "searchList":  
[], "history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "全套", "searchList": []},  
"history": [], "finish": 0}
```



```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "功能", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "支持", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "企业", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "高效", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "构建", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "智能化", "searchList":  
[]}, "history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "应用", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "并", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "降低", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":  
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "AI", "searchList": []},  
"history": [], "finish": 0}
```

```
data: {"code": 0, "message": "success", "msg_id":
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "技术", "searchList": []},
"history": [], "finish": 0}

data: {"code": 0, "message": "success", "msg_id":
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "门槛", "searchList": []},
"history": [], "finish": 0}

data: {"code": 0, "message": "success", "msg_id":
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "。", "searchList": []},
"history": [], "finish": 0}

data: {"code": 0, "message": "success", "msg_id":
"bf10ca0882a09ec6a4e1e26190a841b2", "data": {"output": "", "searchList": []},
"history": [], "finish": 1}
```

非流式

```

{
  "code": 0,
  "message": "success",
  "msg_id": "f89327a67dc05fec500f009b2c605401",
  "data": {
    "output": "DeepSeek是深度求索推出的模块化AI工程化平台，提供从模型纳管到应用落地的完整工具链，支持多租户架构和企业级知识库等功能，帮助企业降低AI应用门槛并加速数字化转型。",
    "searchList": [
      {
        "kb_name": "DeepSeek",
        "title": "README.pdf",
        "snippet": "库建设、复杂工作流编排等完整功能体系的AI工程化平台。平台采用模块化架构设计，支持灵活的功能扩展和二次开发，在确保企业数据安全和隐私保护的同时，大幅降低了AI技术的应用门槛。无论是中小型企业快速构建智能化应用，还是大型企业实现复杂业务场景的智能化改造，DeepSeek都能提供强有力的技术支撑，助力企业加速数字化转型进程，实现降本增效和业务创新。 🔥 平台核心优势 ✓ 企业级工程化：提供从模型纳管到应用落地的完整工具链，解决LLM技术落地“最后一公里”问题 ✓ 开放开源生态：采用宽松友好的 Apache 2.0 License，支持开发者自由扩展与二次开发 ✓ 全栈技术支持：配备专业团队为生态伙伴提供 架构咨询、性能优化 全周期赋能 ✓ 多租户架构：提供多租户账号体系，满足用户成本控制、数据安全隔离、业务弹性扩展、行业定制\n化、快速上线及生态协同等核心需求 ▶ 核心功能模块 1. 模型纳管（Model Hub） ▶ 支持 数百种专有/开源大模型（包括GPT、Claude、Llama等系列）的统一接入与生命周期管理 ▶ 深度适配 OpenAI API 标准 及 DeepSeek 生态模型，实现异构模型的无缝切换 ▶ 提供 多推理后端支持（vLLM、TGI等）与 自托管解决方案，满足不同规模企业的算力需求 2. 可视化工作流（Workflow Studio） ▶ 通过 低代码拖拽画布 快速构建复杂AI业务流程 ▶ 内置 条件分支、API、大模型、知识库、代码 等多种节点，支持端到端流程调试与性能分析 3. 企业级知识库、RAG Pipeline ▶ 提供 知识库创建→文档解析→向量化→检索→精排 的全流程知识管理能力，支持\npdf/docx/txt/xlsx/csv/pptx等 多种格式 文档，还支持网页资源的抓取和接入 ▶ 集成 多模态检索、级联切分 与 自适应切分，显著提升问答准确率 4. 智能体开发框架（Agent Framework） ▶ 可基于 函数调用（Function Calling） 的Agent构建范式，支持工具扩展、私域知识库关联与多轮对\n话 ▶ 支持在线调试 5. 后端即服务（BaaS） ▶ 提供 RESTful API ，支持与企业现有系统（OA/CRM/ERP等）深度集成 ▶ 提供 细粒度权限控制，保障生产环境稳定运行 "
      }
    ],
    {
      "kb_name": "DeepSeek",
      "title": "README.pdf",

```

```
      "snippet": "API + 应用程序导向 API + 应用程序导向 编程方法 支持的  
LLMs RAG引擎 Agent 工作流 可观测性 本地部署  
license友好 多租户 快速开始 Docker安装 从源码安装 典型应用场景 智能客服：基于  
RAG+Agent实现高准确率的业务咨询与工单处理知识管理：构建企业专属知识库，支持语义搜索与智能摘要生  
成 流程自动化：通过 workflow引擎实现合同审核、报销审批等业务的AI辅助决策平台已成功应用于 金融、工  
业、政务 等多个行业，助力企业将LLM技术的理论价值转化为实际业务收益。我们诚邀开发者加入开源社区，  
共同推动AI技术的民主化进程。 许可证 DeepSeek根据Apache License 2.0发布。"  
    },  
  ],  
  "history": [  
    {  
      "query": "请一句话介绍DeepSeek",  
      "response": "DeepSeek是深度求索推出的模块化AI工程化平台，提供从模型纳管到应用落  
地的完整工具链，支持多租户架构和企业级知识库等功能，帮助企业降低AI应用门槛并加速数字化转型。"  
    },  
  ],  
  "finish": 1  
}
```

智能体创建对话API

智能体应用支持会话持久化，可将之前的聊天记录作为上下文进行回答，可适用于聊天/客服 AI 等。

请求接口

Type	Instructions
方法	Http
请求URL	按照应用实际API根地址，例如： http://localhost:8081/service/api/openapi/v1/agent/conversation
字符编码	UTF-8
请求类型	POST
鉴权方式(header参数)	Authorization: Bearer {API Key}
请求格式	Content-Type: application/json
响应格式	非流式: Content-Type: application/json 流 式: Content-Type: text/event-stream

请求参数

Parameter	Required	Type	Instructions
title	是	string	对话标题

响应参数

Parameter	Type	Instructions
code	int	状态码，用于表示请求成功或具体的错误类型
msg	string	提示信息，通常用于提供关于code的详细解释或请求成功的确认信息
data	array	当前响应文本，包含了根据用户输入和知识库搜索得到的答案

data

Parameter	Type	Instructions
conversation_id	string	当前响应文本片段ID

调用示例

```
curl -k --location
'http://localhost:8081/service/api/openapi/v1/agent/conversation' \
--header 'Content-Type: application/json' \
--header 'Accept: application/json' \
--header 'Authorization: Bearer <API Key>' \
--data '{
    "title": "你好，DeepSeek"
}'
```

响应示例

```
{
  "code": 0,
  "data": {
    "conversation_id": "56"
  },
  "msg": ""
}
```

智能体对话API

智能体应用支持会话持久化，可将之前的聊天记录作为上下文进行回答，可适用于聊天/客服 AI 等。

请求接口

Type	Instructions
方法	Http
请求URL	按照应用实际API根地址，例如： <code>http://localhost:8081/service/api/openapi/v1/agent/chat</code>
字符编码	UTF-8

Type	Instructions
请求类型	POST
鉴权方式(header参数)	Authorization: Bearer {API Key}
请求格式	Content-Type: application/json
响应格式	非流式: Content-Type: application/json 流 式: Content-Type: text/event-stream

请求参数

Parameter	Required	Type	Instructions
conversation_id	是	string	历史响应文本片段ID
stream	否	bool	是否以流式接口的形式返回数据，默认为非流式。true为流式，false为非流式
query	是	string	用户提出的问题或提示语

响应参数

Parameter	Type	Instructions
code	int	状态码，用于表示请求成功或具体的错误类型
message	string	提示信息，通常用于提供关于code的详细解释或请求成功的确认信息
gen_file_url_list	array	模型生成输出的文件url列表
response	string	当前响应文本，包含了根据用户输入和知识库搜索得到的答案
search_list	array	知识增强搜索结果
history	array	包含之前对话历史的数组，用于上下文管理
usage	array	token使用量
finish	int	仅在流式返回中有该字段。表示流是否结束，0：未结束，1：正常结束，2：生成长度导致结束，3：异常结束，4：命中安全护栏结束

gen_file_url_list

Parameter	Type	Instructions
output_file_url	string	输出文件url

history

Parameter	Type	Instructions
query	string	请求文本
response	string	模型响应文本

search_list

Parameter	Type	Instructions
kb_name	string	知识库名字
snippet	string	知识内容片段
title	string	文件标题

usage

Parameter	Type	Instructions
prompt_tokens	int	输入提示词token数
completion_tokens	int	输出文本token数
total_tokens	int	输入加输出总的token数

调用示例

流式

```
curl -k --location 'http://localhost:8081/service/api/openapi/v1/agent/chat' \
--header 'Content-Type: application/json' \
--header 'Accept: application/json' \
--header 'Authorization: Bearer <API Key>' \
--data '{
    "stream": true,
    "conversation_id": "56",
    "query": "请一句话介绍DeepSeek"
}'
```

非流式

```
curl -k --location 'http://localhost:8081/service/api/openapi/v1/agent/chat' \
--header 'Content-Type: application/json' \
--header 'Accept: application/json' \
--header 'Authorization: Bearer <API Key>' \
--data '{
    "stream": false,
    "conversation_id": "56",
    "query": "请一句话介绍DeepSeek"
}'
```

响应示例

流式

```
data: {"code": 0, "message": "success", "response": "DeepSeek",  
"gen_file_url_list": [], "history": [], "finish": 0, "usage": {"prompt_tokens":  
0, "completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "是", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "深度求索",  
"gen_file_url_list": [], "history": [], "finish": 0, "usage": {"prompt_tokens":  
0, "completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "推出", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "的", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "AI", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "工程", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "化", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "平台", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "提供", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "从", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "模型", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "纳", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "管", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "到", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "应用", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "落", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "地的", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "完整", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "工具", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```



```
data: {"code": 0, "message": "success", "response": "链", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": ", ", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "支持", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "企业", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "级", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "AI", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "应用的", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "快速", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "构建", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "与", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "智能化", "gen_file_url_list":  
[], "history": [], "finish": 0, "usage": {"prompt_tokens": 0,  
"completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "改造", "gen_file_url_list": [], "history": [], "finish": 0, "usage": {"prompt_tokens": 0, "completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "【", "gen_file_url_list": [], "history": [], "finish": 0, "usage": {"prompt_tokens": 0, "completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "1", "gen_file_url_list": [], "history": [], "finish": 0, "usage": {"prompt_tokens": 0, "completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "^", "gen_file_url_list": [], "history": [], "finish": 0, "usage": {"prompt_tokens": 0, "completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "】", "gen_file_url_list": [], "history": [], "finish": 0, "usage": {"prompt_tokens": 0, "completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "。", "gen_file_url_list": [], "history": [], "finish": 0, "usage": {"prompt_tokens": 0, "completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

```
data: {"code": 0, "message": "success", "response": "", "gen_file_url_list": [], "history": [], "finish": 1, "usage": {"prompt_tokens": 0, "completion_tokens": 0, "total_tokens": 0}, "search_list": [], "qa_type": [1]}
```

非流式

```
{
  "code": 0,
  "message": "success",
  "response": "DeepSeek是深度求索推出的AI工程化平台，提供从模型纳管到应用落地的完整工具链，支持企业级知识库、RAG Pipeline、智能体开发等功能，助力企业数字化转型【1^】【2^】。",
  "gen_file_url_list": [

  ],
  "search_list": [
    {
      "kb_name": "DeepSeek",
      "title": "README.pdf",
    }
  ]
}
```

```
"snippet": "库建设、复杂工作流编排等完整功能体系的AI工程化平台。平台采用模块化架构设计，支持灵活的功能扩展和二次开发，在确保企业数据安全和隐私保护的同时，大幅降低了AI技术的应用门槛。无论是中小型企业快速构建智能化应用，还是大型企业实现复杂业务场景的智能化改造，DeepSeek都能提供强有力的技术支撑，助力企业加速数字化转型进程，实现降本增效和业务创新。🔥 平台核心优势 ✓  
企业级工程化：提供从模型纳管到应用落地的完整工具链，解决LLM技术落地\“最后一公里\“问题 ✓ 开放开源生态：采用宽松友好的 Apache 2.0 License，支持开发者自由扩展与二次开发 ✓ 全栈技术支持：配备专业团队为生态伙伴提供 架构咨询、性能优化 全周期赋能 ✓ 多租户架构：提供多租户账号体系，满足用户成本控制、数据安全隔离、业务弹性扩展、行业定制\ n化、快速上线及生态协同等核心需求 ▶ 核心功能模块  
1. 模型纳管（Model Hub） ▶ 支持 数百种专有/开源大模型（包括GPT、Claude、Llama等系列）的统一接入与生命周期管理 ▶ 深度适配 OpenAI API 标准 及 DeepSeek 生态模型，实现异构模型的无缝切换 ▶ 提供 多推理后端支持（vLLM、TGI等）与 自托管解决方案，满足不同规模企业的算力需求  
2. 可视化工作流（Workflow Studio） ▶ 通过 低代码拖拽画布 快速构建复杂AI业务流程 ▶ 内置条件分支、API、大模型、知识库、代码 等多种节点，支持端到端流程调试与性能分析  
3. 企业级知识库、RAG Pipeline ▶ 提供 知识库创建→文档解析→向量化→检索→精排 的全流程知识管理能力，支持 \npdf/docx/txt/xlsx/csv/pptx等 多种格式 文档，还支持网页资源的抓取和接入 ▶ 集成 多模态检索、级联切分 与 自适应切分，显著提升问答准确率  
4. 智能体开发框架（Agent Framework） ▶ 可基于 函数调用（Function Calling） 的Agent构建范式，支持工具扩展、私域知识库关联与多轮对\ n话 ▶ 支持在线调试  
5. 后端即服务（BaaS） ▶ 提供 RESTful API ，支持与企业现有系统（OA/CRM/ERP等）深度集成 ▶ 提供 细粒度权限控制，保障生产环境稳定运行 "
```

```
},  
{  
    "kb_name": "DeepSeek",  
    "title": "README.pdf",  
    "snippet": "API + 应用程序导向 API + 应用程序导向 编程方法 ✓ ✓ 支持的LLMs  
✓ ✓ RAG引擎 ✓ ✓ Agent ✓ ✓ 工作流 ✓ ✓ 可观测性 ✓ ✓ 本地部署 ✓ ✗ license  
友好 ✓ ✗ 多租户 🚀 快速开始 Docker安装 从源码安装 🎯 典型应用场景 智能客服：基于  
RAG+Agent实现高准确率的业务咨询与工单处理 知识管理：构建企业专属知识库，支持语义搜索与智能摘要生成  
流程自动化：通过工作流引擎实现合同审核、报销审批等业务的AI辅助决策 平台已成功应用于金融、工业、政务 等多个行业，助力企业将LLM技术的理论价值转化为实际业务收益。我们诚邀开发者加入开源社区，共同推动AI技术的民主化进程。 📄 许可证 DeepSeek根据Apache License 2.0发布。"  
}  
],  
"history": [  
    {  
        "query": "请一句话介绍DeepSeek",  
        "response": "DeepSeek是深度求索推出的AI工程化平台，提供从模型纳管到应用落地的完整工具链，支持企业级知识库、RAG Pipeline、智能体开发等功能，助力企业数字化转型【1^】【2^】。"  
    }  
],  
"usage": {  
    "completion_tokens": 0,  
    "prompt_tokens": 0,  
    "total_tokens": 0  
},  
"finish": 1  
}
```