

## 第 2 章 Dify 的应用

### 2.1 构建应用

在 Dify 中，一个“应用”是指基于 GPT 等大语言模型构建的实际场景应用。通过创建应用，你可以将智能 AI 技术应用于特定的需求。它既包含了开发 AI 应用的工程范式，也包含了具体的交付物。

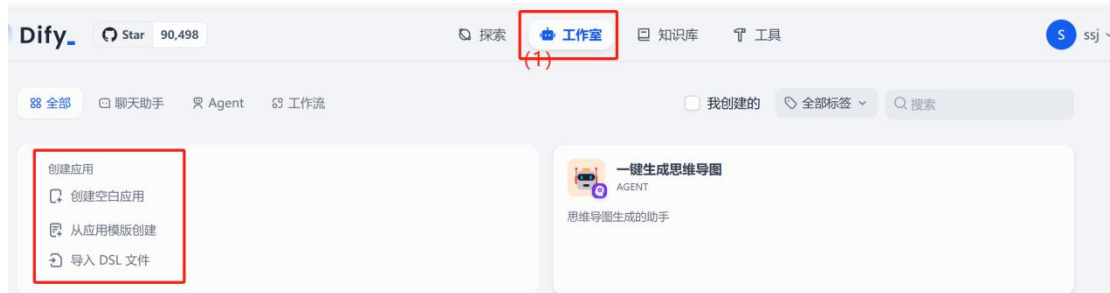


图 2.1 进入 Dify 页面后，点击工作室，左下角有创建应用。

#### 2.1.1 创建方式

你可以通过 3 种方式在 Dify 的工作室内创建应用：

- (1) 基于应用模板创建（新手推荐）
- (2) 创建一个空白应用
- (3) 通过 DSL 文件（本地/在线）创建应用

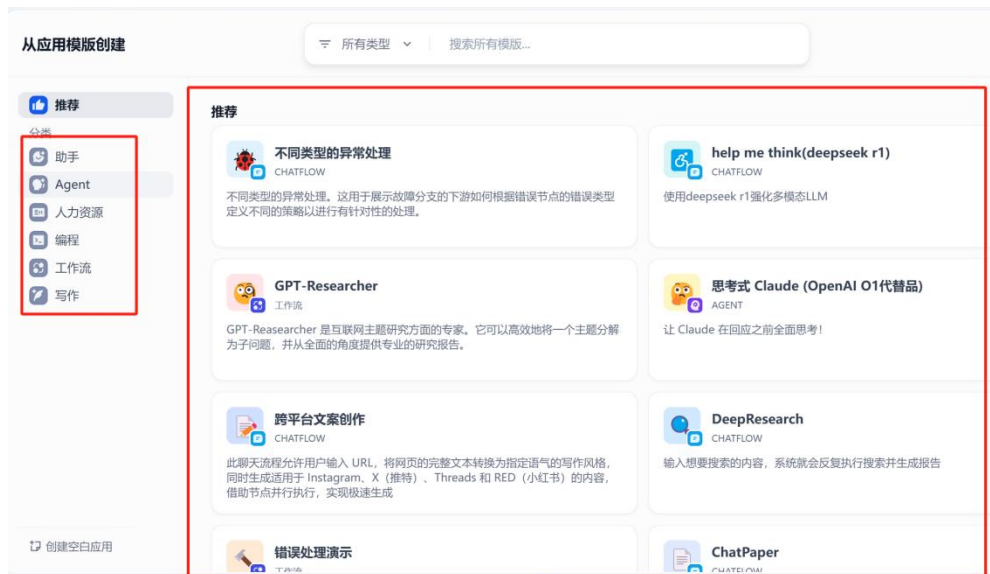
#### 2.1.2 基于应用模板创建

初次使用 Dify 时，你可能对于应用创建比较陌生。为了帮助新手用户快速了解在 Dify 上能够构建哪些类型的应用，Dify 团队内的提示词工程师已经创建好了多场景、高质量的应用模板。

你可以从导航选择【工作室】，在应用列表内选择【从模板创建】。



点击之后，会有很多模板，你可以根据自己的任务进行选择



### 2.1.3 创建空白应用

如果你需要在 Dify 上创建一个空白应用，你可以从导航选择【工作室】，在应用列表内选择【从空白创建】。




点击之后，有【五种应用类型】可供选择。Dify 上可以创建 5 种不同的应用类型，分别是聊天助手、文本生成应用、Agent 和 Chartflow 和工作流。


创建应用时，你需要给应用起一个【名字】、选择合适的【图标】，或者上传喜爱的图片用作图标、使用一段清晰的文字描述此应用的【用途】，以便后续应用在团队内的使用。

### 创建空白应用

#### 选择应用类型


新手适用


**聊天助手**  
简单配置即可构建基于 LLM 的对话机器人

**Agent**  
具备推理与自主工具调用的智能助手


**文本生成应用**  
用于文本生成任务的 AI 助手

进阶用户适用

**Chatflow**  
支持记忆的复杂多轮对话工作流

**工作流**  
面向单轮自动化任务的编排工作流

#### 应用名称 & 图标



#### 描述 (可选)

#### 2.1.4 导入 DSL 文件

Dify DSL 是由 Dify.AI 所定义的 AI 应用工程文件标准，文件格式为 YML。该标准涵盖应用在 Dify 内的基本描述、模型参数、编排配置等信息。

你可以从导航选择【工作室】，在应用列表内选择【导入 DSL 文件】。

当您想用别人已经写好的流程时，对方给您 DSL 文件或者 URL，你就可以进行导入，从而直接运用对方定义好的工作流。



##### (1) 文件导入

导入 DSL

×

文件

URL




↑

拖拽文件至此，或者 [选择文件](#)

取消

创建

如果你从社区或其它人那里获得了一个应用模板（DSL 文件），可以从工作室选择 **【 导入 DSL 文件 】**。DSL 文件导入后将直接加载原应用的所有配置信息。例如在 Github 上有人分享的 DSL 文件，如下图所示，你就可以下载下来，通过文件导入。

 思维导图生成助手.yml	Mindmap-generate-assistant
 思维导图生成助手mindmap_generator.yml	Mindmap-generate-assistant
 解读Github项目智能机器人.yml	Add files via upload

## (2) URL 导入

你也可以通过 URL 导入 DSL 文件，参考的链接格式：

[https://example.com/your\\_dsl.yml](https://example.com/your_dsl.yml)

导入 DSL

×

文件

URL

DSL URL

输入 DSL 文件的 URL

取消

创建

## 2.2 应用类型

Dify 中提供了五种应用类型：

- ◆ **聊天助手**：基于 LLM 构建对话式交互的助手
- ◆ **文本生成应用**：面向文本生成类任务的助手，例如撰写故事、文本分类、翻译等
- ◆ **Agent**：能够分解任务、推理思考、调用工具的对话式智能助手
- ◆ **对话流**：适用于定义等复杂流程的多轮对话场景，具有记忆功能的应用编排方式
- ◆ **工作流**：适用于自动化、批处理等单轮生成类任务的场景的应用编排方式

### 2.2.1 聊天助手

对话型应用采用一问一答模式与用户持续对话。

#### (1) 适用场景

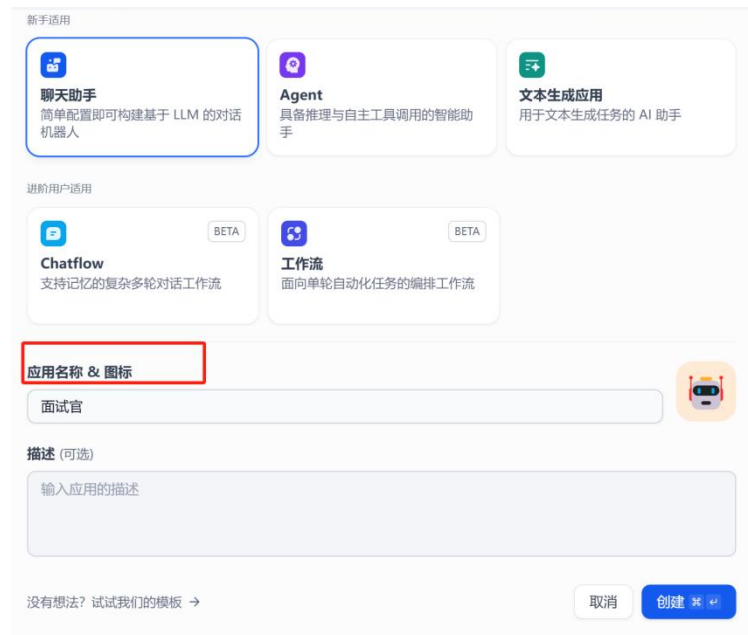
对话型应用可以用在客户服务、在线教育、医疗保健、金融服务等领域。这些应用可以帮助组织提高工作效率、减少人工成本和提供更好的用户体验。

#### (2) 如何编排

对话型应用的编排支持：对话前提示词，变量，上下文，开场白和下一步问题建议。

#### (3) 创建应用

在首页点击“创建应用”按钮创建应用。填上应用名称，应用类型选择聊天助手。



我们现在将创建【面试官】这个聊天助手

创建应用后会自动跳转到应用概览页。点击左侧菜单【编排】来编排应用。

#### (4) 填写提示词

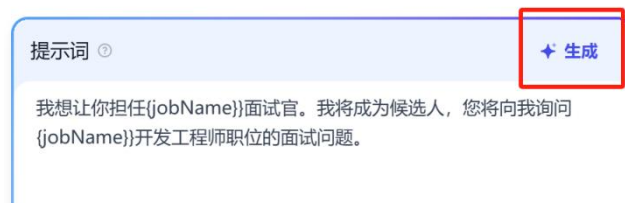
提示词用于约束 AI 给出专业的回复，让回应更加精确。你可以借助内置的提示生成器，编写合适的提示词。提示词内支持插入表单变量，例如 `{{input}}`。提示词中的变量的值会替换成用户填写的值。

##### (1) 输入提示指令。

例如：我想让你担任`{{jobName}}`面试官。我将成为候选人，您将向我询问`{{jobName}}`开发工程师职位的面试问题。

##### (2) 右侧内容框将自动生成提示词。

#### 编排



点击【提示词】右上方的【生成】，进入到【提示词生成器】。

### 提示词生成器

提示词生成器使用配置的模型来优化提示词，以获得更高的质量和更好的结构。请写出清晰详细的说明。

/root/Qwen2.5-72B-Instruct-AWQ/ CHAT

试一试

Python 代码助手 翻译机器人 总结会议纪要 润色文章  
职业分析师 Excel 公式专家 旅行规划助手 SQL 生成 Git 大师

指令

我想让你担任(jobName)面试官。我将成为候选人，您将向我询问(jobName)开发工程师职位的面试问题。

生成

### 生成的提示词

提示词

```
```xml
<instruction>
请根据以下提供的信息，担任(jobName)开发工程师职位的面试官。您需要向我提出一系列面试问题，这些问题应该涵盖(jobName)开发工程师所需的关键技能和知识。请确保问题具有挑战性，同时也能评估候选人的实际工作经验和解决问题的能力。请不要在输出中包含任何XML标签。

1. 首先，根据(jobName)确定面试的领域和技术栈。
2. 然后，准备5-10个面试问题，这些问题应该包括但不限于以下方面：
- 技术基础知识
- 项目经验
```
```

变量

| 变量 KEY  | 字段名称    |
|---------|---------|
| jobName | jobName |

聊天增强

对话开场白

欢迎！我是本次Python开发工程师职位的面试官。为了更好地评估您的技术能力和实际工

取消

应用

输入【指令】点击【生成】，右端就会出现生成的提示词，下面还会自动生成对话的开场白。

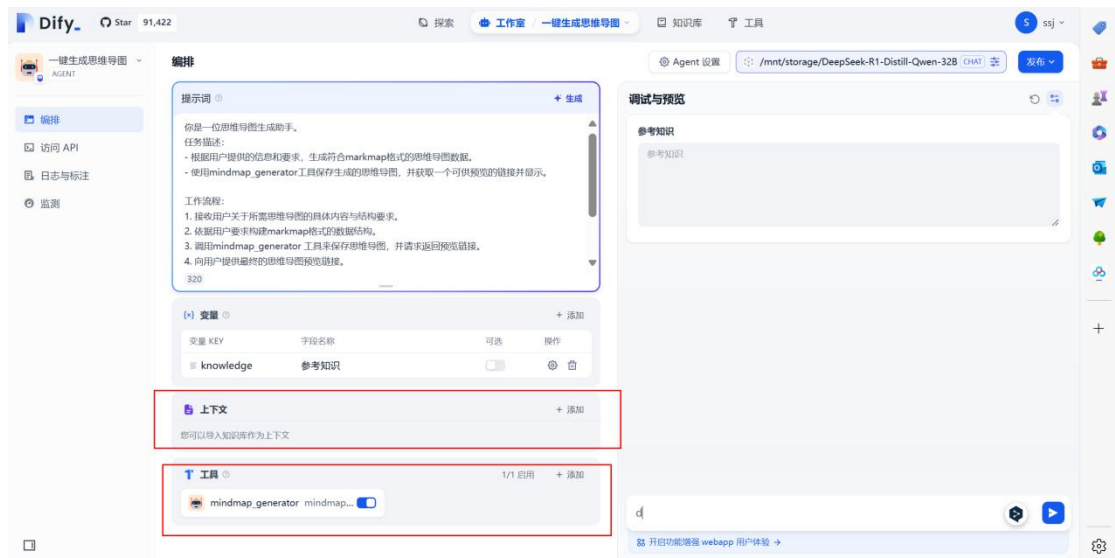


例如我输入 jobName 为【大模型】之后他会像面试官一样发题目

## 2.2.2 Agent

智能助手（Agent Assistant），利用大语言模型的推理能力，能够自主对复杂的人类任务进行目标规划、任务拆解、工具调用、过程迭代，并在没有人类干预的情况下完成任务。

在【上下文】中，你可以添加智能助手可以用于查询的知识库工具，这将帮助它获取外部背景知识。



在【工具】中，你可以添加需要使用的工具。工具可以扩展 LLM 的能力，比如联网搜索、科学计算或绘制图片，赋予并增强了 LLM 连接外部世界的的能力。Dify 提供了两种工具类型：第一方工具和自定义工具。

你可以直接使用 Dify 生态提供的第一方内置工具，或者轻松导入自定义的 API 工具（目前支持 OpenAPI / Swagger 和 OpenAI Plugin 规范）。



### 2.2.3 文本生成应用

文本生成类应用是一种根据用户提供的提示，自动生成高质量文本的应用。它可以生成各种类型的文本，例如文章摘要、翻译等。





| 功能名称  | 功能含义  |
|-------|---|
| 前缀提示词 | 提示词用于对 AI 的回复做出一系列指令和约束。可插入表单变量例如{input}。这段提示词不会被最终用户所看到。 |
| 变量    | 变量将以表单形式让用户在对话前填写，用户填写的表单内容将自动替换提示词中的变量                   |
| 上下文   | 支持添加知识库做为上下文信息，且支持两种召回方式。 <b>N</b> 选 <b>1</b> 召回和多路召回。    |



|       |   |
|-------|---|
| 更多类似的 | 一次生成多条文本，可在此基础上编辑并继续生成  |
| 文字转语音 | 启用后，文本可以转换成语音   |
| 内容审查  | 您可以调用审查 <b>API</b> 或者维护敏感词库来使模型更安全地输出。支持 <b>OpenAI Moderation</b> ，关键词， <b>API</b> 拓展三种方式 |

#### 2.2.4 Chatflow

面向对话类情景，包括客户服务、语义搜索、以及其他需要在构建响应时进行【多步逻辑】的对话式应用程序。该类型应用的特点在于支持对生成的结果进行多轮对话交互，调整生成的结果。

**常见的交互路径：**给出指令 → 生成内容 → 就内容进行多次讨论 → 重新生成结果 → 结束

#### 2.2.5 工作流（Workflow）

非对话式自动化流程，专注批量处理与后台任务

工作流可以发布【工具】，该类型应用无法对生成的结果进行多轮对话交互。

**常见的交互路径：**给出指令 → 生成内容 → 结束

#### 2.2.6 如何选择应用类型

##### 交互复杂度

需简单问答 → 聊天助手

需分步骤引导 → Chatflow

需零交互执行 → Workflow

##### 任务类型

生成固定格式内容 → 文本生成应用

调用外部工具 → Agent

处理结构化数据 → Workflow

##### 技术能力

新手优先选择模板化类型（聊天助手/文本生成）

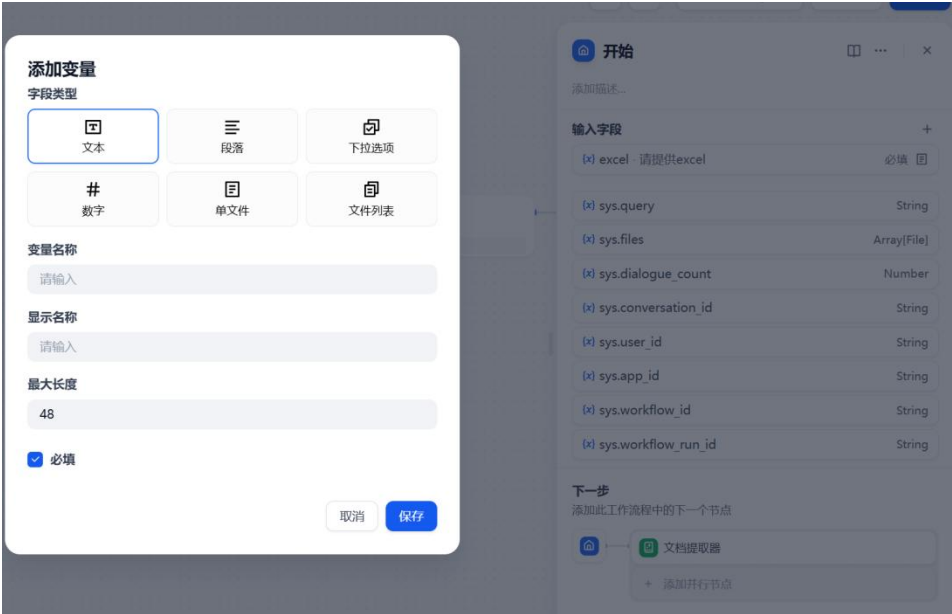
开发者推荐使用可编程类型（Agent/Workflow）

### 第 3 章 Dify 的组件

节点是工作流中的关键构成，通过连接不同功能的节点，执行工作流的一系列操作。

#### 3.1 开始节点

应用场景：作为所有工作流的入口，定义输入参数格式。  
在开始节点的设置页，你可以看到两部分设置，分别是【输入字段】和预设的【系统变量】。  
允许应用使用者单独上传文件，支持文档类型文件、图片、音频、视频和其它文件类型。支持通过本地上传文件或粘贴文件 URL。详细用法请参考文件上传。



|      |   |
|------|---|
| 文本   | 短文本，由应用使用者自行填写内容，最大长度 256 字符。                               |
| 段落   | 长文本，允许应用使用者输入较长字符。  |
| 下拉选项 | 由应用开发者固定选项，应用使用者仅能选择预设选项，无法自行填写内容。                          |
| 数字   | 仅允许用户输入数字。  |
| 单文件  | 允许应用使用者单独上传文件，支持文档类型文件、图片、音频、视频和其它文件类型。支持通过本地上传文件或粘贴文件 URL。 |
| 文件列表 | 允许应用使用者批量上传文件，支持文档类型文件、图片、音频、视频和其它文件类型。支持通过本地上传文件或粘贴文件 URL。 |

【系统变量】

Chatflow 和 Workflow 是不同的。

Chatflow 类型应用系统变量

| 变量名称                             | 数据类型        | 说明  | 备注   |
|----------------------------------|-------------|---|--|
| <code>sys.query</code>           | String      | 用户在对话框中初始输入的内容  |  |
| <code>sys.files</code>           | Array[File] | 用户在对话框内上传的图片  | 图片上传功能需在应用编排页右上角的“功能”处开启                   |
| <code>sys.dialogue_count</code>  | Number      | 用户在与 Chatflow 类型应用交互时的对话轮数。每轮对话后自动计数增加 1，可以和 if-else 节点搭配出丰富的分支逻辑。<br><br>例如到第 X 轮对话时，回顾历史对话并给出分析 |  |
| <code>sys.conversation_id</code> | String      | 对话框交互会话的唯一标识符，将所有相关的消息分组到同一个对话中，确保 LLM 针对同一个主题和上下文持续对话  |  |
| <code>sys.user_id</code>         | String      | 分配给每个应用用户的唯一标识符，用以区分不同的对话用户   |  |
| <code>sys.app_id</code>          | String      | 应用 ID，系统会向每个 Workflow 应用分配一个唯一的标识符，用以区分不同的应用，并通过此参数记录当前应用的基本信息                                    | 面向具备开发能力的用户，通过此参数区分并定位不同的 Workflow 应用      |
| <code>sys.workflow_id</code>     | String      | Workflow ID，用于记录当前 Workflow 应用内所包含的所有节点信息   | 面向具备开发能力的用户，可以通过此参数追踪并记录 Workflow 内的包含节点信息 |
| <code>sys.workflow_run_id</code> | String      | Workflow 应用运行 ID，用于记录 Workflow 应用中的运行情况   | 面向具备开发能力的用户，可以通过此参数追踪应用的历次运行情况             |

Workflow 类型应用系统变量

| 变量名称  | 数据类型        | 说明   | 备注   |
|---|-------------|--|--|
| <code>sys.files</code><br><code>[LEGACY]</code> | Array[File] | 文件参数，存储用户初始使用应用时上传的图片  | 图片上传功能需在应用编排页右上角的“功能”处开启                   |
| <code>sys.user_id</code>                        | String      | 用户 ID，每个用户在使用工作流应用时，系统会自动向用户分配唯一标识符，用以区分不同的对话用户                |  |
| <code>sys.app_id</code>                         | String      | 应用 ID，系统会向每个 Workflow 应用分配一个唯一的标识符，用以区分不同的应用，并通过此参数记录当前应用的基本信息 | 面向具备开发能力的用户，通过此参数区分并定位不同的 Workflow 应用      |
| <code>sys.workflow_id</code>                    | String      | Workflow ID，用于记录当前 Workflow 应用内所包含的所有节点信息                      | 面向具备开发能力的用户，可以通过此参数追踪并记录 Workflow 内的包含节点信息 |
| <code>sys.workflow_run_id</code>                | String      | Workflow 应用运行 ID，用于记录 Workflow 应用中的运行情况                        | 面向具备开发能力的用户，可以通过此参数追踪应用的历次运行情况             |

Workflow 类型应用系统变量

### 3.2 结束节点

定义一个 **workflow** 流程结束的最终输出内容。

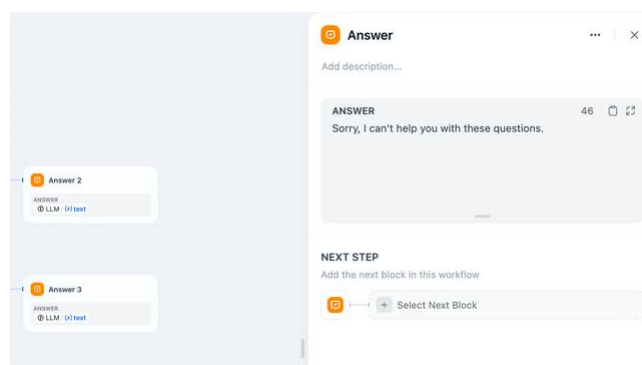
一般都是输出一个变量/值结束



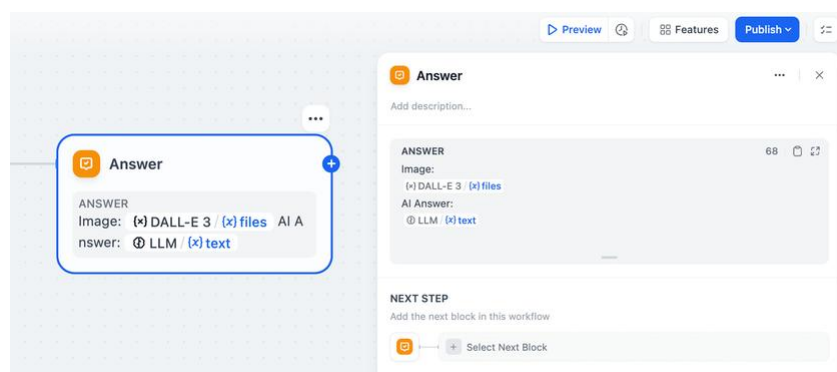
### 3.3 直接回复节点

定义一个 **Chatflow** 流程中的回复内容。可以回复文本，输出图片，LLM 输出等，看你需要什么样子的返回过程。

#### 【回复纯文本】



#### 【输出图片/LLM 的输出】

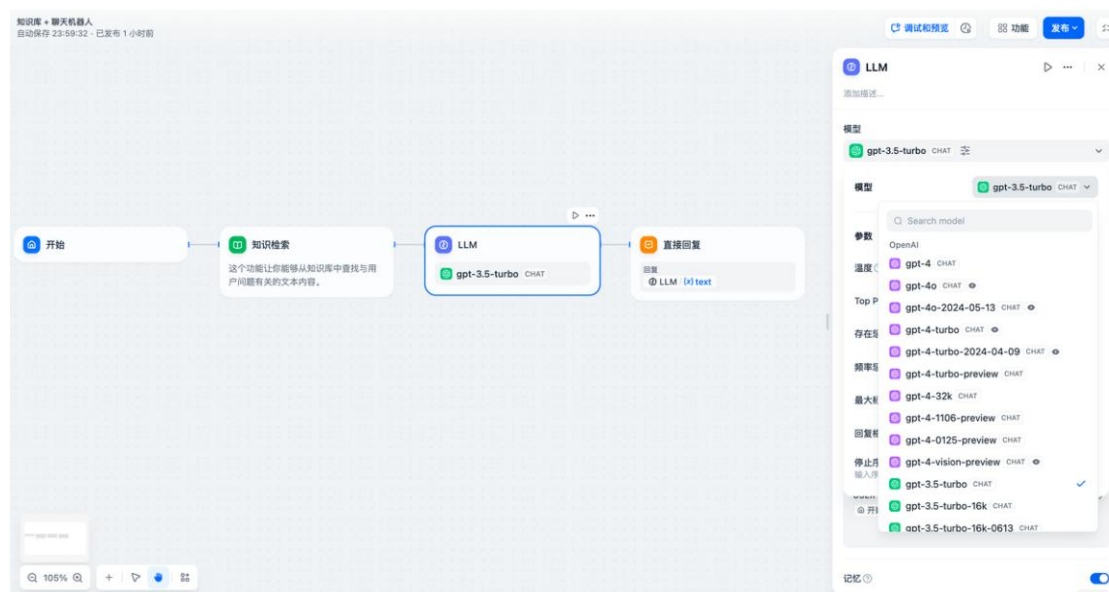


当然这些回复

### 3.4 LLM 节点

【定义】：调用大语言模型的能力，处理用户在“开始”节点中输入的信息（自然语言、上传的文件或图片），给出有效的回应信息。

【如何添加】：在应用编辑页中，点击鼠标右键或轻点上一节点末尾的 + 号，添加节点并选择 LLM。



【填写上下文】（可选），上下文可以理解为向 LLM 提供的背景信息，常用于填写知识检索的输出变量。

【System】：定义助手的底层行为逻辑，相当于给模型一个「隐形人格」或「初始规则」。通常对用户不可见，但会持续影响整个对话的走向。优先级最高，会覆盖模型的默认行为。修改 **SYSTEM** 可彻底改变助手性格（如从「严谨律师」切换到「幽默朋友」）。

【USER】：模拟真实用户的输入，用于触发模型响应或提供上下文。

【ASSISTANT】：定义助手历史回复内容，用于维持对话一致性或纠正错误。可用于「伪造」对话历史，强制模型延续特定风格。通过修改 **ASSISTANT** 历史可改变模型后续回答（如从「肯定语气」切换到「谨慎语气」）。

| 角色        | 是否用户可见 | 影响范围    | 示例场景      |
|-----------|--------|---------|-----------|
| SYSTEM    | ✗ 不可见  | 全局隐性控制  | 设定助手的专业领域 |
| USER      | ✓ 可见   | 单次输入触发  | 用户提问或指令   |
| ASSISTANT | ✓ 可见   | 后续回答一致性 | 助手的历史回复记录 |

SYSTEM ⓘ

55 Jinja (x) 🗑️ 🔍

你是一名SEO专家和主题领域专家。你的任务是根据用户提供的关键词以及谷歌搜索的上下文，生成一个SEO文章标题。

USER ⓘ

92 Jinja (x) 🗑️ 🔍

为了了解我的文章应该关于什么，这些是  
📄 开始 / (x) keyword 的最高排名结果: 📄 开始 / (x) title  
是什么原则使这些排名靠前?

ASSISTANT ⓘ

358 Jinja (x) 🗑️ 🔍

### 翻译为中文  
为了制作一个对关键词" 📄 开始 / (x) keyword "友好的SEO文章标题，并且与你分享的排名靠前的结果中观察到的原则相符，理解这些标题之所以有效的原因很重要。以下是可能有助于它们高排名的原则：  
1. \*\*关键词的放置和清晰度\*\*：每个标题通过包含精确关键词或非常接近的变体直接回应了查询。这种清晰度确保搜索引擎可以轻松理解内容的相关性。  
2. \*\*简洁和直接性\*\*：标题简洁，使其易于快速阅读和理解。它们避免了不必要的词语，直奔主题。  
3. \*\*包含定义或解释\*\*：标题暗示文章将定义或解释概念，这正是搜索"{{keyword}}"的人所寻找的。  
4. \*\*呈现的多样性\*\*：尽管覆盖了类似的内容，每个标题都从略微不同的角度接近主题。这种多样性可以吸引更广泛的受众兴趣。

USER ⓘ

146 Jinja (x) 🗑️ 🔍

鉴于这些原则，请帮我生成一个标题，该标题将以模仿排名靠前的标题的语法为基础，针对关键词"  
📄 开始 / (x) keyword "进行优化。请不要复制，而是给出更好的选项，并避免使用“掌握”、“全面”、“发现”或“揭示”等语言。不要使用动名词，只用主动态、现在时。只需返回标题。

+ 添加消息

【模型参数】：模型的参数会影响模型的输出效果。不同模型的参数会有所区别。下图为 gpt-4 的参数列表。



参数 加载预设

温度 ☒ 0.7

Top P ☐ 1

存在惩罚 ☐ 0

频率惩罚 ☐ 0

最大标记 ☐ 512

种子 ☐ 0

回复格式 ☐ 请选择

停止序列 ☐ 输入序列并按 Tab 键

|       |  |
|-------|--|
| 温度    | 通常是 0-1 的一个值，控制随机性。温度越接近 0，结果越确定和重复，温度越接近 1，结果越随机。   |
| Top P | 控制结果的多样性。模型根据概率从候选词中选择，确保累积概率不超过预设的阈值 P。   |
| 存在惩罚  | 用于减少重复生成同一实体或信息，通过对已经生成的内容施加惩罚，使模型倾向于生成新的或不同的内容。参数值增加时，对于已经生成过的内容，模型在后续生成中被施加更大的惩罚，生成重复内容的可能性越低。 |
| 频率惩罚  | 对过于频繁出现的词或短语施加惩罚，通过降低这些词的生成概率。随着参数值的增加，对频繁出现的词或短语施加更大的惩罚。较高的参数值会减少这些词的出现频率，从而增加文本的词汇多样性。         |

如果你不理解这些参数是什么，可以选择加载预设，从创意、平衡、精确三种预设中选择。

参数 加载预设

温度 ☒ 创意

Top P ☐ 平衡

精确

|      |   |
|------|---|
| 记忆   | 开启记忆后问题分类器的每次输入将包含对话中的聊天历史，以帮助 LLM 理解上文，提高对话交互中的问题理解能力。 |
| 记忆窗口 | 记忆窗口关闭时，系统会根据模型上下文窗口动态过滤聊天历史的传递数量；打开时用户可以精确控制聊          |



|            |  |
|------------|--|
|            | 天历史的传递数量（对数）。                                      |
| 对话角色名设置    | 由于模型在训练阶段的差异，不同模型对于角色名的指令遵循程度不同。                   |
| Jinja-2 模板 | LLM 的提示词编辑器内支持 Jinja-2 模板语言，实现轻量级数据转换和逻辑处理，参考官方文档。 |
| 错误重试       | 针对节点发生的部分异常情况，通常情况下再次重试运行节点即可解决。                   |

### 3.5 问题分类器

**【定义】**: 通过定义分类描述，问题分类器能够根据用户输入，使用 LLM 推理与之相匹配的分类并输出**【分类结果】**，向下游节点提供更加精确的信息。

#### **【示例展示】**

当用户输入不同的问题时，问题分类器会根据已设置的分类标签 / 描述自动完成分类：

“iPhone 14 如何设置通讯录联系人？” —> “与产品操作使用相关的问题”

“保修期限是多久？” —> “与售后相关的问题”

“今天天气怎么样？” —> “其他问题”

#### **【配置步骤】**：

1. 选择输入变量，指用于分类的输入内容，支持输入文件变量。客服问答场景下一般为用户输入的问题 `sys.query`;
2. 选择推理模型，问题分类器基于大语言模型的自然语言分类和推理能力，选择合适的模型将有助于提升分类效果
3. 编写分类标签/描述，你可以手动添加多个分类，通过编写分类的关键词或者描述语句，让大语言模型更好的理解分类依据。
4. 选择分类对应的下游节点，问题分类节点完成分类之后，可以根据分类与下游节点的关系选择后续的流程路径。

### 3.6 知识检索

**【定义】**：从知识库中检索与用户问题相关的文本内容，可作为下游 LLM 节点的上下文来使用。

**【配置流程】**：

1. 选择查询变量。查询变量通常代表用户输入的问题，该变量可以作为输入项并检索知识库中的相关文本分段。在常见的对话类应用中一般将开始节点的 `sys.query` 作为查询变量，知识库所能接受的最大查询内容为 200 字符；
2. 选择需要查询的知识库，可选知识库需要在 Dify 知识库内预先创建；
3. 在元数据筛选板块中配置元数据的筛选条件，使用元数据功能筛选知识库内的文档。
4. 指定召回模式。自 9 月 1 日后，知识库的召回模式将自动切换为多路召回，不再建议使用 N 选 1 召回模式；
5. 连接并配置下游节点，一般为 LLM 节点；

### 3.7 条件分支

**【定义】**：根据 If/else/elif 条件将 Chatflow / Workflow 流程拆分成多个分支。

**【节点功能】**：条件分支的运行机制包含以下六个路径：

IF 条件：选择变量，设置条件和满足条件的值；

IF 条件判断为 True，执行 IF 路径；

IF 条件判断为 False，执行 ELSE 路径；

ELIF 条件判断为 True，执行 ELIF 路径；

ELIF 条件判断为 False，继续判断下一个 ELIF 路径或执行最后的 ELSE 路径；

**【条件类型】**：支持设置以下条件类型：

包含 (Contains)

不包含 (Not contains)

开始是 (Start with)

结束是 (End with)  
是 (Is)  
不是 (Is not)  
为空 (Is empty)  
不为空 (Is not empty)

### 3.8 模板转换

【定义】允许借助 Jinja2 的 Python 模板语言灵活地进行数据转换、文本处理等。

【什么是 Jinja2】：Jinja 是一个快速、表达力强、可扩展的模板引擎。

—— <https://jinja.palletsprojects.com/en/3.1.x/>

【怎么用】：可以输入任何文本，当输入变量的时候就需要在变量的左右加上双括号。



### 3.9 列表操作

【变量】：列表操作节点仅接受以下数据结构变量：

Array[string]  
Array[number]  
Array[file]

【例子】：

['章','张','沈']：这种字符串的组合一起  
[1,2,3,4,5,6,7]：这种多个数字存在一起  
['a.txt','b.txt','c.txt']：这种多个文件组合在一起



【过滤条件】：处理输入变量中的数组，添加过滤条件。

【取第 N 项】：如名所示

【取前 N 项】：如名所示

【排序】：提供对于输入变量中数组的排序能力，支持根据文件属性进行排序。就是升序、降序

【输出变量】：可以是输出结果、可以是首个变量、可以是最后一个变量，这是可以进行选择的。

### 3.10 变量赋值

【定义】：变量赋值节点用于向可写入变量进行变量赋值，已支持以下可写入变量。

【用法】：通过变量赋值节点，你可以将 workflow 内的变量赋值到会话变量中用于临时存储，并可以在后续对话中持续引用。

其实就是有一个变量，之后就进行赋值。



### 3.11 变量聚合

【定义】：将多路分支的变量聚合为一个变量，以实现下游节点统一配置。

【用法】：其实就是有很多变量，变成只能聚合同一种数据类型的变量。若第一个添加至变量聚合节点内的变量数据格式为 `String`，后续连线时会自动过滤可添加变量为 `String` 类型。

### 3.12 HTTP 请求

【定义】：允许通过 HTTP 协议发送服务器请求，适用于获取外部数据、webhook、生成图片、下载文件等情景。

该节点支持常见的 HTTP 请求方法：

【GET】，用于请求服务器发送某个资源。

【POST】，用于向服务器提交数据，通常用于提交表单或上传文件。

【HEAD】，类似于 GET 请求，但服务器不返回请求的资源主体，只返回响应头。

【PATCH】，用于在请求-响应链上的每个节点获取传输路径。

【PUT】，用于向服务器上传资源，通常用于更新已存在的资源或创建新的资源。

【DELETE】，用于请求服务器删除指定的资源。

一般用的是 GET 和 POST。

这个具体的用法在案例【】【】中进行学习。

### 3.13 Agent

Agent 节点是 Dify Chatflow/Workflow 中用于实现自主工具调用的组件。它通过集成不同的 Agent 推理策略，使大语言模型能够在运行时动态选择并执行工具，从而实现多步推理。

从下拉菜单选择所需的 Agent 推理策略。Dify 内置了 Function Calling 和 ReAct 两种策略，可在 Marketplace → Agent 策略分类中安装使用。

#### 1. Function Calling

通过将用户指令映射到预定义函数或工具，LLM 先识别用户意图，再决定调用哪个函数并提取所需参数。它的核心是调用外部函数或工具，属于一种明确的工具调用机制。

**优点:**(1)精确：对于明确的任务，可以直接调用相应的工具，无需复杂的推理过程。(2)易于集成外部功能：可以将各种外部 API 或工具封装成函数供模型调用。结构化输出：模型输出的是结构化的函数调用信息，方便下游节点处理。

#### 2. ReAct (Reason + Act)

ReAct 策略使 Agent 交替进行思考和行动：LLM 首先思考当前状态和目标，然后选择并调用合适的工具，工具的输出结果又将引导 LLM 进行下一步的思考和行动，如此循环，直到问题解决。

**优点：**（1）有效利用外部信息：能够有效地利用外部工具获取信息，解决仅靠模型自身无法完成的任务。（2）可解释性较好：思考和行动的过程是交织的，可以一定程度上追踪 Agent 的推理路径。（3）适用范围广：适用于需要外部知识或需要执行特定操作的场景，例如问答、信息检索、任务执行等。

### 3.13 代码执行

注意点：（1）`def main` 一定是 `main`

（2）输入变量要注意定义

（3）输出变量一定记得是个字典类似的，虽然下面的输出变量也是一个 `String`

## 代码执行

▶ □ ... ×

添加描述...

输入变量

+

arg1

⊕ LLM1 / textString

🗑

PYTHON3

✦ 📄 ↗

```
1 import re
2
3 def main(arg1: str) -> dict:
4     pattern = r'!\[image\].*?
5     enhance=true&private=true\'
6     match = re.search(pattern, arg1,
7         re.DOTALL)
8
9     result = match.group()
```

输出变量 \*

+

result

String

▼

🗑

## 代码执行

▶ □ ... ×

添加描述...

输入变量

+

arg1

⊕ LLM1 / textString

🗑

PYTHON3

✦ 📄 ↗

```
6
7     result = match.group()
8
9     return {
10         "result": result,
11     }
12
```

输出变量 \*

+

result

String

▼

🗑