## Dify工作流 - 长篇文章生成

创建一个长篇文章生成工作流应用去学习更多的Dify工作流节点，包括LLM、代码执行、迭代节点。

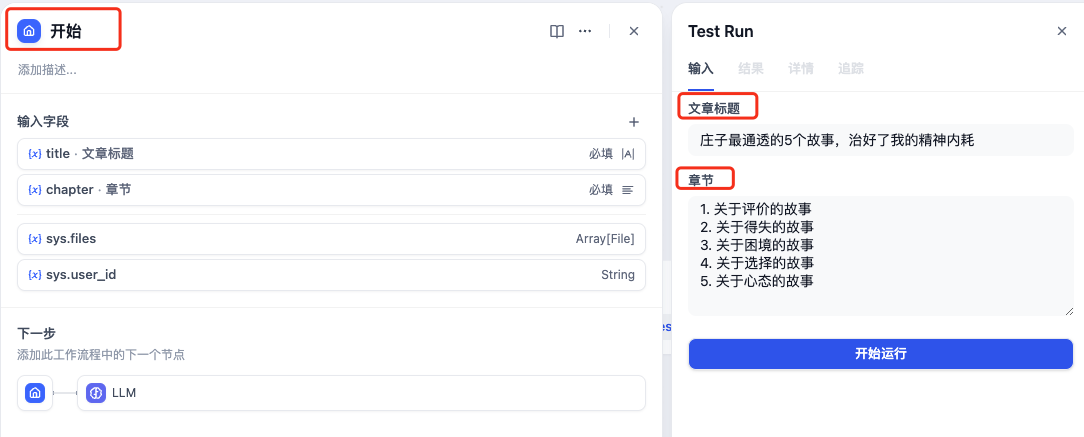
需求：用户输入文章标题和文章各个一级章节，让长篇文章生成工作流生成更多的子章节，并最终输出一篇具有吸引力的长文。该原理可应用于撰写标书大纲。

### 1.创建Workflow



### 2.开始

开始节点定义了title（文章标题）、chapter（文章一级章节）输入变量。



### 3.LLM



### 4.LLM提示词

提示词中定义了角色、技能、目标、限制和输出示例，让LLM严格按照意图来输出文本。

这个LLM节点的作用是根据用户输入的文章标题和各章节名称，生成各个章节下的子章节，丰富文章章节结构，并且以json格式输出，供下一个代码执行节点使用。

注意

这里注意使用GLM-4模型，将提示词设置在USER中，如果设置在SYSTEM中，运行该LLM节点会报错，这和Dify调用GLM大模型的方式有关。如果换成GPT-4o，设置在SYSTEM中即可。

### 5.提示词Prompt

## 角色：文章撰写专家

## 技能：

你根据用户输入的文章标题/title和各章节名称/chapter，生成各个章节下的子章节。

## 目标：

- 确保生成的每个子章节和父章节紧密相关。

- 纵观整体章节，必须保证各章节过渡连贯流畅。

- 最终输出json字符串，详细请看以下输出示例。

## 限制：

- 输出的json字符串必须是标准json字符串，不要包含任何XML标签和转义字符，如`\n`。

- 严重按照输出示例中的例子输出json字符串，不要输出其他任何文本。

## 输出示例：

[ { "chapter": "引言", "subchapter": "1. 气候变化对沿海城市影响的概述 2. 理解这些影响的重要性" }, { "chapter": "海平面上升", "subchapter": "1. 海平面上升的原因 2. 对沿海基础设施和社区的影响 3. 受影响城市的例子" }, { "chapter": "风暴频率增加", "subchapter": "1. 气候变化与风暴频率的关联 2. 更频繁和严重的风暴对沿海地区的影响 3. 最近风暴的案例研究" }, { "chapter": "结论", "subchapter": "1. 关键点总结 2. 应对气候变化的紧迫性 3. 对政策制定者和社区的行动呼吁" } ]

### 6.代码执行

代码节点支持运行 Python / NodeJS 代码以在工作流程中执行数据转换。它可以简化工作流程，适用于Arithmetic、JSON transform、文本处理等情景。

该节点极大地增强了开发人员的灵活性，使他们能够在工作流程中嵌入自定义的 Python 或 Javascript 脚本，并以预设节点无法达到的方式操作变量。通过配置选项，你可以指明所需的输入和输出变量，并撰写相应的执行代码。

### 7.json解析

本小节中使用Python3代码，代码执行节点解析输入的json字符串，输出Array[Object]变量，供下一个迭代节点使用。

def main(arg1: str) -> dict:

import json

data = json.loads(arg1)

# Create an array of objects

result = [{'chapter': item["chapter"], 'subchapter': item["subchapter"]} for item in data]

return {

'result': result

}

输入json字符串：

[{"chapter": "1. 关于评价的故事","subchapter": "1. 认识自我与外界评价的差异 2. 庄子与惠子对话的启示 3. 从评价中解脱，找回内心的平和"},{"chapter": "2. 关于得失的故事","subchapter": "1. 得失之间的辩证关系 2. 庄子丧妻的从容态度 3. 从得失观照人生，学会释然"},{"chapter": "3. 关于困境的故事","subchapter": "1. 面对困境的心态调整 2. 庄子与监河侯的对话寓意 3. 突破困境，实现自我成长"},{"chapter": "4. 关于选择的故事","subchapter": "1. 选择与命运的关系 2. 庄子拒绝做官的智慧 3. 做出明智选择，活出精彩人生"},{"chapter": "5. 关于心态的故事","subchapter": "1. 心态决定人生高度 2. 庄子与蝴蝶的故事启示 3. 调整心态，战胜精神内耗"}]

输出Array[Object]变量：

{

"result": [

{

"chapter": "1. 关于评价的故事",

"subchapter": "1. 认识自我与外界评价的差异 2. 庄子与惠子对话的启示 3. 从评价中解脱，找回内心的平和"

},

{

"chapter": "2. 关于得失的故事",

"subchapter": "1. 得失之间的辩证关系 2. 庄子丧妻的从容态度 3. 从得失观照人生，学会释然"

},

{

"chapter": "3. 关于困境的故事",

"subchapter": "1. 面对困境的心态调整 2. 庄子与监河侯的对话寓意 3. 突破困境，实现自我成长"

},

{

"chapter": "4. 关于选择的故事",

"subchapter": "1. 选择与命运的关系 2. 庄子拒绝做官的智慧 3. 做出明智选择，活出精彩人生"

},

{

"chapter": "5. 关于心态的故事",

"subchapter": "1. 心态决定人生高度 2. 庄子与蝴蝶的故事启示 3. 调整心态，战胜精神内耗"

}

]

}



重要节点

迭代

上一篇中，代码执行节点输入Array[Object]变量。迭代节点循环Array[Object]，取出其中每一个Object，Object示例如下：

{

"chapter": "1. 关于评价的故事",

"subchapter": "1. 认识自我与外界评价的差异 2. 庄子与惠子对话的启示 3. 从评价中解脱，找回内心的平和"

}

一个Object代表了一个章节（包含父章节和各个子章节）。因为包含5个章节，所以迭代节点循环迭代5次，根据每一个Object生成章节的详细内容，如何生成？在迭代节点中嵌套LLM节点，使用LLM节点编写每一个章节的具体内容。

嵌套的LLM节点中，在SYSTEM和USER中都分别编写了提示词Prompt。

迭代节点

每一个LLM节点输出的都是String类型的Text文本。所以最终迭代节点输出的就是Array[String]数组，这是一个包含每个章节详细内容的数组。

迭代节点

迭代节点输出内容：

根据迭代节点的输出结果，下一个代码节点，仅仅需要将Array[String]中每一个String取出并用换行符\n拼接就能形成最终的长篇文章Text String。

代码执行

## 技术标书生成思路

已有条件：

1.招标文件

2.投标标书目录大纲-拆分-章节-迭代-合并-word

3.公司已有的技术或方案（从以往的技术标书中使用Workflow提取）-为每个章节创建Prompt模板

