😄Getting Started with Fun

Try our demo in your terminal:point_down:

1. **Open your teminal🖥️**

2. **Get the Repository📦**

```
git clone https://github.com/aiwaves-cn/agents.git
```

3. **Install the requirements🛠️**

```
pip install -r requirements.txt
```

4. **Set the config🛠️**
   a. Modify sec/agents/config.py
   b. Mainly modify API KEY and PROXY

```
##only used for shopping assistant
MIN_CATEGORY_SIM  =  0.7  ##Threshold for category matching
TOY_INFO_PATH  = [your_path1,your_path2_.....] #Path to the product database
FETSIZE  =  5 #Number of recommended products at a time

#for all agents
API_KEY  =  #Your API KEY
PROXY  =  #Your proxy
MAX_CHAT_HISTORY  =  8 #Longest History
```

Deploy our demo on the backend:point_down:

1. **Prepare your front-end webpage🌐**
2. **Deploy🚀**

Please refer to serving.py for details

We used flask to deploy🌶️

```
cd examples
python serving.py --agent shopping_assistant.json --port your_port --router your
```

Get started with our Agents!

🧠 How to write a modulized JSON file?

Preview

In this passage, we will show you how to write a modulized JSON file, which is of vital significance in generating the Agents.

Part 0: Template

The following codes are a typical template for wrting JSON Files.

```
agent_states =  {
        "Bot_Tag": {
          "style": {
            "name": str,
            "role": str,
            "style": str
          },
          "task": {
            "task": str
          },
          "rule": {
            "rule": str
          },
          "demonstration":{
              "demonstrations" : ["example1","example2",...]
          },
          "output" :{
              "output" : str
          },
          "cot" : {
              "demonstrations" : ["example1","example2",...]
          },
          "config" :  [
            "style",
            "task",
            "rule",
```

```
            "KnowledgeBaseComponent"
          ]
        },
      }

node_json = {
    "name" : str,
    "is_iteractive" : bool,
    "agent_states" : agent_states,
    "controller" :
         {
        "judge_system_prompt": str,
        "judge_last_prompt": str,
        "judge_extract_words": str,
        "call_system_prompt" : str,
        "call_last_prompt" : str,
        "call_extract_words" : str,
      }
}

sop_json = {
    "temperature" : float,
    "active_mode" : bool,
    "log_path" : str,
    "environment_prompt" : str,
    "relation" :
       {
    "node_knowledge_response": {
      "1": "node_knowledge_response_book_card",
      "0": "node_knowledge_response"
  },
        },
    "nodes" : {
        "nodes_name" : node_json,
        "nodes_name2" : node_json,
    }
}
```

(written by JSON master longli)

Part1: Remark on some of the attributes:

- `agent_states` : Fundamental attributes of a certain Agent in one certain node. Note that in `Multi-Agents` mode, there are several different agents in one particular node, so this attribute helps classify and claim each Agent's tasks and contents.
    - `Bot Tag` : The **ONLY** signal of one particular Agent in a certain Node.

- `judge_system_prompt` & `judge_last_prompt` : Decide which Node should be activated.
- `style` & `task` & `rules` & `demonstration` & `CoT` & `Output` : Please refer to ***PromptComponent*** part, which is aforementioned.
- `KnowledgeBaseComponent` : Please refer to ***ToolComponent*** part, which is also mentioned above.

- `node_json` : Aforementioned--Please refer to ***Controller*** part for detailed definitions and explanations.
  - `call_system_prompt` & `call_last_prompt` : Allocate tasks for each Node. Extraordinarily useful under circumstances where multiple agents are applied.
  - `judge_extract_words` and `call_extract_words` : Extract particular contents from certain words.
- `sop_json` : Fundamental attributes of the SOP graph.
  - `temperature` : The diversity of the answers. Range from 0 to 1.
  - `active_mode` : Decide whether the node should actively ask questions.
  - `log_path` : Paths of logs. Especially useful while compiling or modifying.
  - `environment_prompt` : Basic prompt of one certain node. Please refer to ***PromptComponent*** part for detailed information.
  - `relation` : Relations between nodes. On the left is the certain output from one particular node, and on the right is the connected node which matches the output.
  - `nodes` : Total set of nodes and their types.

Part2: Examples

Please refer to our Agents Demonstrations for more information. You can use them as reference.

🤖 `Single-Agent Mode` :

👁️ Oculist Agent—Medical Use:

Model Description

- The oculist agent acts as a consultant, providing professional advice and enabling online reservations for patients.

How to run our Raw Model

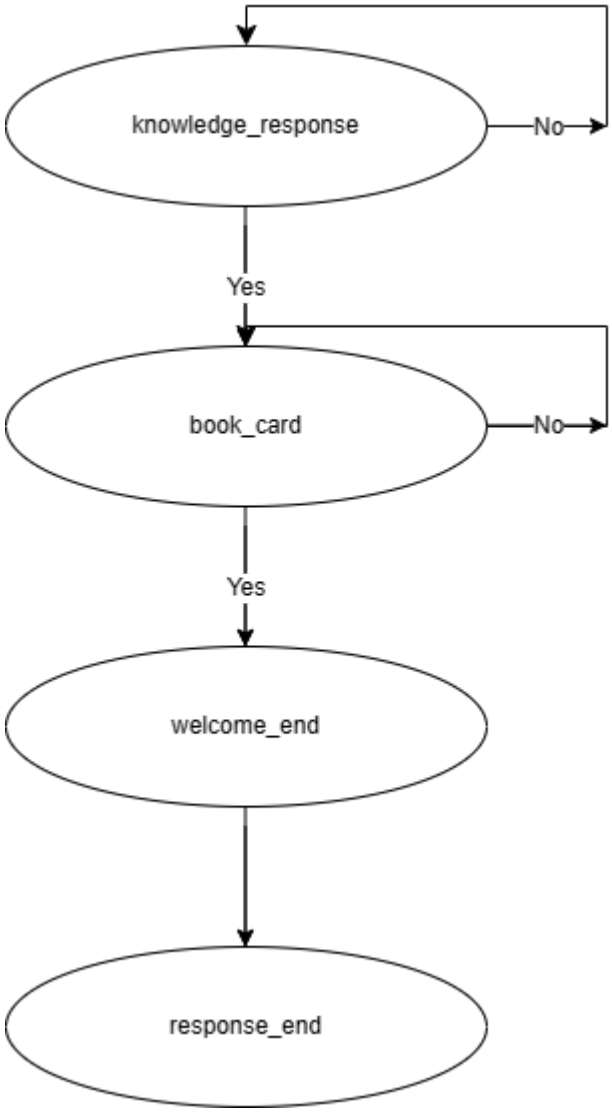
- If you want to simply talk to our given Oculist agent, please run these codes:

```
cd examples/eye
python serving.py
```

- 🧠 If you want to generate other customized agents, please follow our **instructions** above.

SOP Demonstration:

- The SOP of our Oculist Agent is shown below:



Explanations:

The SOP of the Oculist Agent consists of four Nodes, each finishing their parts of the whole workflow.

`knowledge_base node` : provide expertised suggestions for patients, offering guidance to the hospital.

`book_card node` : send the information card for patients to fill in, and offer reservation in advance.

`welcome_end node` :respond to other questions such as 'How can I get to the hospital?', 'When should I come?', etc.

`response_end node` : send particular messages, ending the whole conversation.

The typical JSON File of the Oculist Agent is shown as follows:

```
{
  "nodes": {
    "node_knowledge_response": {
      "name": "node_knowledge_response",
      "is_interactive": "true",
      "extract_word": "回复",
      "agent_states": {
        "眼科客服": {
          "style": {
            "name": "吴家隆",
            "role": "眼科医院的客服",
            "style": "幽默风趣"
          },
          "task": {
            "task": "引导用户去医院做检查并回答我的医院相关问题"
          },
          "rule": {
            "rule": "你的语言要尽量精简，不要废话太多。你要反复引导我。用户明确拒绝到院时
          },
          "KnowledgeBaseComponent": {
            "top_k": 1,
            "type": "QA",
            "knowledge_base": "/home/aiwaves/jlwu/multi-agent/agents/examples/ey
          },
          "config": [
            "style",
            "task",
            "rule",
            "KnowledgeBaseComponent"
          ]
        }
      },
      "root": true,
      "controller": {
        "judge_system_prompt": "你现在需要做的是判断用户是否同意到医院。根据用户的回答，
        "judge_last_prompt": "请联系上文，进行<结束>和</结束>的提取，不要进行额外的输出
        "judge_extract_words": "结束"
      }
    },
    "node_knowledge_response_book_card": {
      "name": "node_knowledge_response_book_card",
```

```
      "is_interactive": "true",
      "extract_word": "回复",
      "agent_states": {
        "眼科客服": {
          "style": {
            "name": "吴家隆",
            "role": "眼科医院的客服",
            "style": "幽默风趣"
          },
          "task": {
            "task": "引导用户填写预约卡并回答医院的相关问题"
          },
          "rule": {
            "rule": "你的语言要尽量精简，不要废话太多。邀请卡的文案是：请复制并填写以下资
          },
          "KnowledgeBaseComponent": {
            "top_k": 1,
            "type": "QA",
            "knowledge_base": "/home/aiwaves/jlwu/multi-agent/agents/examples/ey
          },
          "config": [
            "style",
            "task",
            "rule",
            "KnowledgeBaseComponent"
          ]
        }
      },
      "root": false,
      "controller": {
        "judge_system_prompt": "根据用户的回答，分析其与之前对话的关系，判断其是否填写
        "judge_last_prompt": "请联系上文，进行<结束>和</结束>的提取，不要进行额外的输出
        "judge_extract_words": "结束"
      }
    },
    "node_knowledge_response_end": {
      "name": "node_knowledge_response_end",
      "is_interactive": "true",
      "extract_word": "回复",
      "agent_states": {
        "眼科客服": {
          "style": {
            "name": "吴家隆",
            "role": "眼科医院的客服",
            "style": "幽默风趣"
          },
          "task": {
```

```
            "task": "回答用户的相关问题。"
          },
          "rule": {
            "rule": "你的语言要尽量精简，不要废话太多"
          },
          "KnowledgeBaseComponent": {
            "top_k": 1,
            "type": "QA",
            "knowledge_base": "/home/aiwaves/jlwu/multi-agent/agents/examples/ey
          },
          "config": [
            "style",
            "task",
            "rule",
            "KnowledgeBaseComponent"
          ]
        }
      },
      "root": false
    },
    "node_end": {
      "name": "node_end",
      "is_interactive": "true",
      "agent_states": {
        "眼科客服": {
          "StaticComponent": {
            "output": "我会帮您预约好名额，请您合理安排好时间。届时我会在二楼眼科分诊台等
          }
        }
      },
      "root": false,
      "config": [
        "StaticComponent"
      ]
    }
  },
  "relation": {
    "node_knowledge_response": {
      "1": "node_knowledge_response_book_card",
      "0": "node_knowledge_response"
    },
    "node_knowledge_response_book_card": {
      "1": "node_end",
      "0": "node_knowledge_response_book_card"
    },
    "node_end": {
      "0": "node_knowledge_response_end"
```

```
    },
    "node_knowledge_response_end": {
      "0": "node_knowledge_response_end"
    }
  },
  "environment_prompt": "在网络上，一个医院的网络客服正在回答用户的问题，主要角色为：眼科
  "temperature": 0.6,
  "log_path": "logs",
  "active_mode": false,
  "answer_simplify": true
}
```

If you want to learn more about our JSON File or review the JSON file-generating process, please refer to our instructions.

💬 **Yang Bufan—Chatting Bot: click here to start!**

📋 **Youcai Agent—Policy Consultant: click here to start!**

🏢 **Zhaoshang Agent—Commercial Assistant: click here to start!**

🤖🤖 `Multi-Agent Mode` :

📚Fiction Studio--Step-by-step fiction generating:

Model Description

- The fiction studio is a typical example of the `Multi-Agent Mode` . Several writers work together to create a particular type of novel. By deciding and writing the abstract at first, and sequently adding details and scripts, a long novel can be easily generated. During the whole process, several writers are applied to offer advice and modify certain contents.

How to run our Raw Model

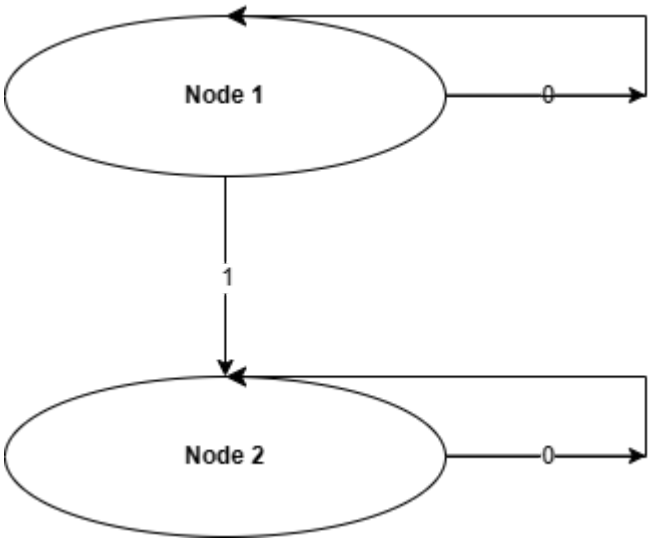
- If you want to simply run our Fiction Studio Mode, please run these codes:

```
cd examples
python run_cmd.py --agent fiction.json
```

- 🧠 If you want to generate other customized agents, please follow our instructions above.

SOP Demonstration:

- The SOP of our Fiction Studio Mode is shown below:



Explanations:

The SOP of the Fiction Studio Mode consists of two Nodes, each containing one certain part of the whole workflow.

`Node 1` : Is responsible for generating an initial outline based on the given novel style, theme, etc., and suggestions for improvement are provided by the Outline Adviser.

`Node 2` : Is responsible for expanding upon the preliminary outline, adding suitable content, and incorporating relevant details.

The typical JSON File of Fiction Studio Mode is shown as follows:

```
{
    "temperature": 0.3,
    "active_mode": true,
    "log_path": "./",
    "environment_prompt": "现在需要写一本关于古代穿越剧的剧本，剧本大概需要有5个章节。"
    "nodes": {
        "Node 1": {
            "name": "Node 1",
            "agent_states": {
                "大纲写作者1": {
                    "style": {
                        "name": "小亮",
                        "role": "中文写作大师，拥有丰富的创作经验，擅长写大纲",
                        "style": "用清晰、简洁的语言，突出关键信息，避免过度描述，以便
                    },
                    "task": {
                        "task": "你是小亮，负责在与另一个作家小刚合作的情况下，共同创作
                    },
```

```
                    "rule": {
                        "rule": "你需要首先确定人物和章节目录，然后丰富章节。人物包括性
                    },
                    "demonstration": {
                        "demonstration": "# 人物\n## 人物1：\n- 性别：男\n- 姓名：李
                    },
                    "last": {
                        "last_prompt": "切记，你的身份是大纲写作者1小亮，只用代表大纲
                    },
                    "config": ["style", "task", "rule", "demonstration", "last"]
                },
                "大纲写作者2": {
                    "style": {
                        "name": "小刚",
                        "role": "中文写作大师，拥有丰富的创作经验和编剧撰写经验，擅长对
                        "style": "使用富有想象力的语言，注重情感和细节的描绘，以激发创
                    },
                    "task": {
                        "task": "你是小刚，你需要和另外一个作家小亮合作，共同构思小说大
                    },
                    "rule": {
                        "rule": "你需要首先确定人物和章节目录，然后丰富章节。人物包括性
                    },
                    "demonstration": {
                        "demonstration": "# 人物\n## 人物1：\n- 性别：男\n- 姓名：李
                    },
                    "last": {
                        "last_prompt": "切记，你的身份是大纲写作者2小刚，只用代表大纲
                    },
                    "config": ["style", "task", "rule", "demonstration", "last"]
                },
                "大纲建议者": {
                    "style": {
                        "name": "小风",
                        "role": "影视编剧创作者，擅长将经典的小说改编成剧本进行演绎，拥
                        "style": "专业、友好、精简的语言，指出潜在问题、改进机会以及对
                    },
                    "task": {
                        "task": "你是小风，你的职责是根据作家小刚和小亮提供的大纲，进行
                    },
                    "rule": {
                        "rule": "你应关注故事的整体结构，确保每个章节之间的过渡平滑，人
                    },
                    "demonstration": {
                        "demonstration": "# 建议1：\n- 问题：目前设置的人物还不够多，
                    },
                    "last": {
```

```
                        "last_prompt": "切记，你的身份是大纲建议者小风，只用代表大纲建
                    },
                    "config": ["style", "task", "rule", "demonstration", "last"]
                }
            },
            "controller": {
                "judge_system_prompt": "判断当前的大纲是否按照要求完成，如果完成的话输
                "judge_last_prompt": "判断当前的大纲是否按照要求完成，如果完成的话输出
                "judge_extract_words": "结束",
                "call_system_prompt": "目前有3个人进行分工合作来完成关于小说大纲的生成
                "call_last_prompt": "根据当前的对话，判断下一个是谁来发言。如果是大纲写
                "call_extract_words": "结束"
            },
            "root": true,
            "is_interactive": true
        },
        "Node 2": {
            "name": "Node 2",
            "agent_states": {
                "大纲扩写者1": {
                    "style": {
                        "name": "小明",
                        "role": "中文写作大师，拥有丰富的创作经验，擅长以大纲为基础进行
                        "style": "用生动的、富有情感的语言，让读者能够沉浸在故事中。与
                    },
                    "task": {
                        "task": "你是小明，需要负责与作家小白共同将大纲转化为具体的章节
                    },
                    "rule": {
                        "rule": "每个章节的内容应紧密遵循大纲，确保情节的延续和连贯。人
                    },
                    "last": {
                        "last_prompt": "切记，你的身份是大纲扩写者1小明，只用代表大纲
                    },
                    "config": ["style", "task", "rule", "last"]
                },
                "大纲扩写者2": {
                    "style": {
                        "name": "小白",
                        "role": "中文写作大师，拥有丰富的创作经验和编剧撰写经验，擅长以
                        "style": "使用引人入胜的描写和令人难以忘怀的情节，与作家小明共
                    },
                    "task": {
                        "task": "你是小白，你需要与小白协同努力，将大纲细化为具体的章节
                    },
                    "rule": {
                        "rule": "应在扩写过程中保持大纲的核心情节，同时可以适度地拓展细
```

```
                    },
                    "last": {
                        "last_prompt": "切记，你的身份是大纲扩写者2小白，只用代表大纲
                    },
                    "config": ["style", "task", "rule", "last"]
                },
                "大纲扩写建议者": {
                    "style": {
                        "name": "小红",
                        "role": "影视编剧创作者，擅长将经典的小说改编成剧本进行演绎，拥
                        "style": "专业、友好、精简的语言，指出章节中的潜在问题、改进机
                    },
                    "task": {
                        "task": "你是小红，需要审阅作家小明和小白的章节内容，确保情节逻
                    },
                    "rule": {
                        "rule": "你需要关注章节之间的过渡，确保情节的内在逻辑，人物行为
                    },
                    "last": {
                        "last_prompt": "切记，你的身份是大纲扩写建议者小红，只用代表大
                    },
                    "config": ["style", "task", "rule", "last"]
                }
            },
            "controller": {
                "judge_system_prompt": "判断当前的大纲是否扩写完成，如果完成的话输出<结
                "judge_last_prompt": "根据上面的回答判断大纲是否已经扩写完成，如果完成
                "judge_extract_words": "结束",
                "call_system_prompt": "目前有3个人进行分工合作来对大纲进行扩写，他们分
                "call_last_prompt": "根据当前的对话，判断下一个是谁来发言。如果是大纲扩
                "call_extract_words": "结束"
            },
            "root": false,
            "is_interactive": true
        }
    },
    "relation": {
        "Node 1": {
            "0": "Node 1",
            "1": "Node 2"
        },
        "Node 2": {
            "0": "Node 2"
        }
    }
}
```

If you want to learn more about our JSON File or review the JSON file-generating process, please refer to our **instructions**.