

下面是基于你给出的 FastAPI 代码整理生成的 《智能体管理接口文档（API Doc）》，采用 Markdown 标准接口文档风格，可直接用于：

- 前端联调
- 接口测试（Postman / Apifox）
- 项目 README / 接口说明文档



智能体管理接口文档（Agents API）

基础信息

- 接口前缀： `/agents`
- 数据格式： `application/json`
- 流式接口协议： `Server-Sent Events (SSE)`
- 统一响应模型： `APIResponse`

一、通用响应结构

```
{
  "success": true,
  "data": {},
  "error": null
}
```

字段	类型	说明
success	boolean	是否成功
data	object	返回数据
error	string	错误信息（失败时）

二、智能体模板相关接口

1 获取智能体模板列表

接口地址

```
GET /agents/templates/list
```

返回示例

json

```
[
  {
    "template_id": "template_001",
    "template_name": "量子销售经理",
    "description": "负责销售咨询的智能体"
  }
]
```

字段说明

字段	说明
template_id	模板ID
template_name	模板名称
description	模板描述

三、智能体实例管理

2 获取智能体实例列表

接口地址

```
GET /agents/instance/list
```

返回示例

json

```
[
  {
    "id": "agent_xxx",
    "name": "量子销售经理",
    "type": "sales",
    "description": "量子销售经理智能体",
    "is_active": true,
    "is_speaking": false,
    "run_time": "RUNNING",
    "cognitive_state": "IDLE"
  }
]
```

3 创建智能体实例 🔗

接口地址

POST /agents/create

请求参数

json

```
{
  "agent_config_id": 1
}
```

参数	类型	必填	说明
agent_config_id	int	是	智能体配置ID（数据库）

返回示例

```
json

{
  "success": true,
  "data": {
    "id": "agent_xxx",
    "name": "量子销售经理",
    "type": "sales",
    "description": "销售智能体",
    "is_active": false,
    "is_speaking": false
  }
}
```

4 获取所有智能体 ID 列表 ::

接口地址

```
GET /agents/list
```

返回示例

```
json

{
  "success": true,
  "data": {
    "agents": ["agent_1", "agent_2"]
  }
}
```

四、消息交互接口

5 发送消息（非流式） ::

接口地址

```
POST /agents/{agent_id}/message
```

请求参数

json

```
{
  "message": "你好",
  "session_id": "session_001"
}
```

参数	类型	必填	说明
message	string	是	用户输入
session_id	string	否	会话ID

返回示例

json

```
{
  "success": true,
  "data": {
    "response": "你好，我是量子销售经理",
    "session_id": "session_001",
    "processing_time": 0.82
  }
}
```

6 流式聊天接口（SSE）

接口地址

```
POST /agents/{agent_id}/stream
```

请求参数

json

```
{
  "message": "帮我分析客户情况",
  "session_id": "session_001"
}
```

事件类型说明

type	说明
stream_start	流开始
chunk	内容片段
stream_end	流结束
error	错误事件

SSE 示例

text

```
data: {"type":"stream_start","session_id":"session_001"}

data: {"type":"chunk","content":"客户目前"}

data: {"type":"chunk","content":"处于跟进阶段"}

data: {"type":"stream_end","session_id":"session_001"}
```

五、智能体状态控制

7 获取智能体信息 / 健康状态

接口地址

```
GET /agents/{agent_id}/info
```

返回示例

json

```
{
  "success": true,
  "data": {
    "status": "healthy",
    "memory_usage": "normal",
    "latency": 120
  }
}
```

8 切换当前活跃智能体

接口地址



```
POST /agents/activate
```

请求参数



json

```
{  
  "agent_id": "agent_001"  
}
```

返回示例



json

```
{  
  "success": true,  
  "data": {  
    "agents": ["agent_001", "agent_002"],  
    "current_activate": "agent_001"  
  }  
}
```

9 关闭指定智能体

接口地址



```
POST /agents/{agent_id}/close
```

返回示例



json

```
{  
  "success": true  
}
```

六、补充说明

- 流式接口适用于 **对话框逐字渲染**
 - `session_id` 用于多轮对话上下文管理
 - 支持同时存在多个智能体实例
 - 激活态智能体用于默认交互路由
-

推荐用途：

- 前端聊天窗口
 - Agent 管理控制台
 - 多智能体协作系统
-

你可以告诉我：

- 是否要拆成 OpenAPI / Apifox / Swagger 风格？
- 是否需要我帮你生成前端 Axios / fetch 调用示例？
- 是否要补一张“智能体生命周期 & 状态流转图”？