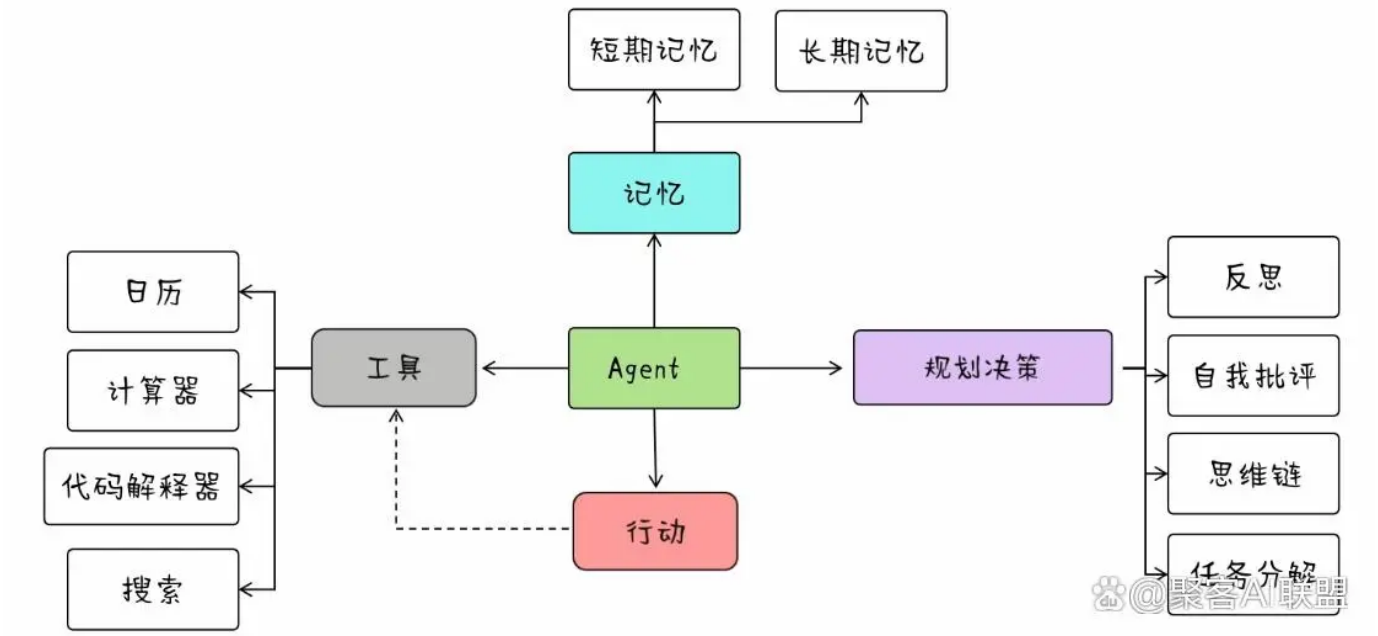
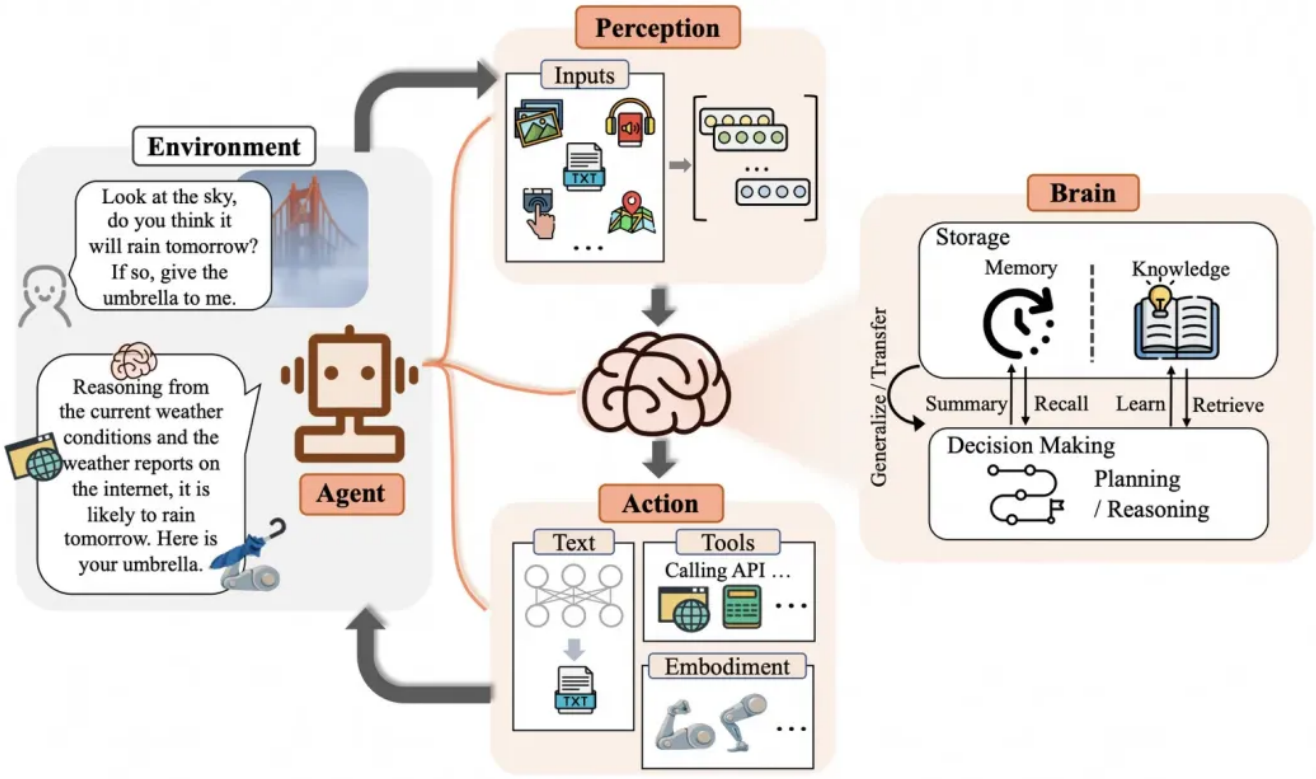
02-Agent智能体

1、什么是 Agent

- 1） 简单来说，AI Agent就是能自主完成任务的智能体，**让大模型“代理/模拟”「人」的行为，使用某些“工具/功能”来完成某些“任务”的能力。**
- 2） 许多大厂、独角兽公司、研究所、高校，也给Agent下过许多定义，比较经典的一个定义是OpenAI的研究主管Lilian Weng给出的定义是：**Agent = 大模型（LLM）+ 规划（Planning）+ 记忆（Memory）+ 工具使用（Tool Use）；**这个定义实际上是从技术实现的角度对Agent进行了定义，它指的是要实现一个Agent，就需要支持这些能力，它需要基于大模型，需要有规划的能力，能思考接下来要做的事情，需要有记忆，能够读取长期记忆和短期记忆，需要能够使用工具，他是将支持这些能力的集合体定义为了Agent。



- 3) 另外的一个定义是复旦大学NLP团队给出来的，他们认为Agent的概念框架包括三个组件：**大脑、感知、行动**[2]。大脑模块作为控制器，承担记忆、思考和决策等基本任务。感知模块从外部环境感知并处理多模态信息，而行动模块则使用工具执行任务并影响周围环境。比如：当人类询问是否会下雨时，感知模块将指令转换为大模型可以理解的表示，然后，大脑会根据当前天气和互联网天气报告开始推理，最后，行动模块作出回应并将雨伞递给人类。通过重复上述过程，Agent可以不断获得反馈并与环境互动。



2、Agent 开发平台

随着2023年生成式AI元年的到来，Agent 被广泛认为是AI落地应用的重要方向。 自2022年年底，国外已经开始探索 Agent 开发框架与技术，而国内的 Agent 开发平台在2023年下半年才开始发布，2024年被定义为Agent元年。

国外的 Agent 开发平台：LangChain、LlamaIndex、AutoGPT、NexusGPT，OpenAI的GPTs和Swarm。

国内的 Agent 开发平台：Dify（主打海外市场）、FastGPT、智谱智能体中心、天工SkyAgents、扣子Coze、星火智能体、腾讯元器，百度的文心智能体AgentBuilder和千帆AppBuilder。

2.1、低代码框架：敏捷开发的入口

- 代表工具：Coze（字节跳动）、Dify（国内开源）、LangFlow（LangChain生态）
- 技术特点：
  - 可视化编排：支持拖拽式工作流设计（如Coze的节点式编辑器），内置预置模板（客服Bot、数据分析助手等），适合无编程基础的用户快速验证想法。
  - 云原生集成：Coze直接对接云存储、预置，开发者无需自建基础设施。
- 局限性：

  自定义工具接入困难，难以实现复杂逻辑，不进行二次开发也难以实现与企业实际需求相匹配。

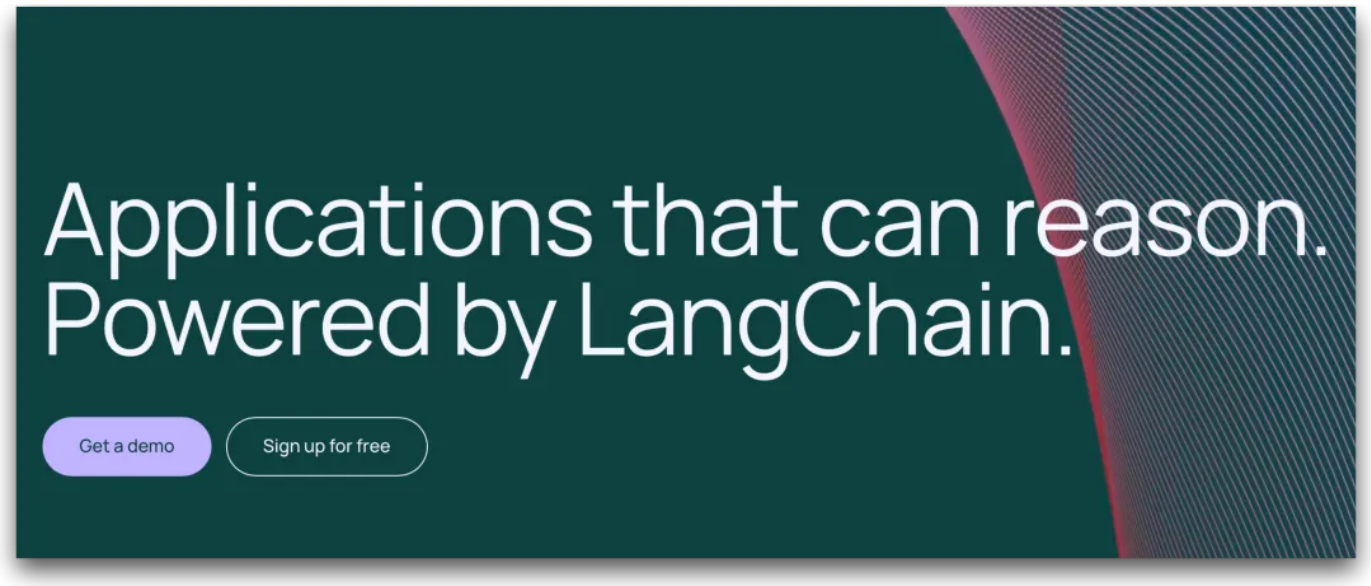
2.2、原生基础框架：轻量级开发的核心

- 核心能力：基于大模型原生function calling（如DeepSeek-V3的Tool Use）
- 技术实现：
  - 直接通过OpenAI兼容API调用工具，无需中间层封装。
  - 示例代码片段（DeepSeek-V3）：

```python
response = client.chat.completions.create(
    model="deepseek-chat",
    messages=[{"role": "user", "content": "查询北京天气并生成图表"}],
    tools=[search_tool, plot_tool]  # 注册工具列表
)
```

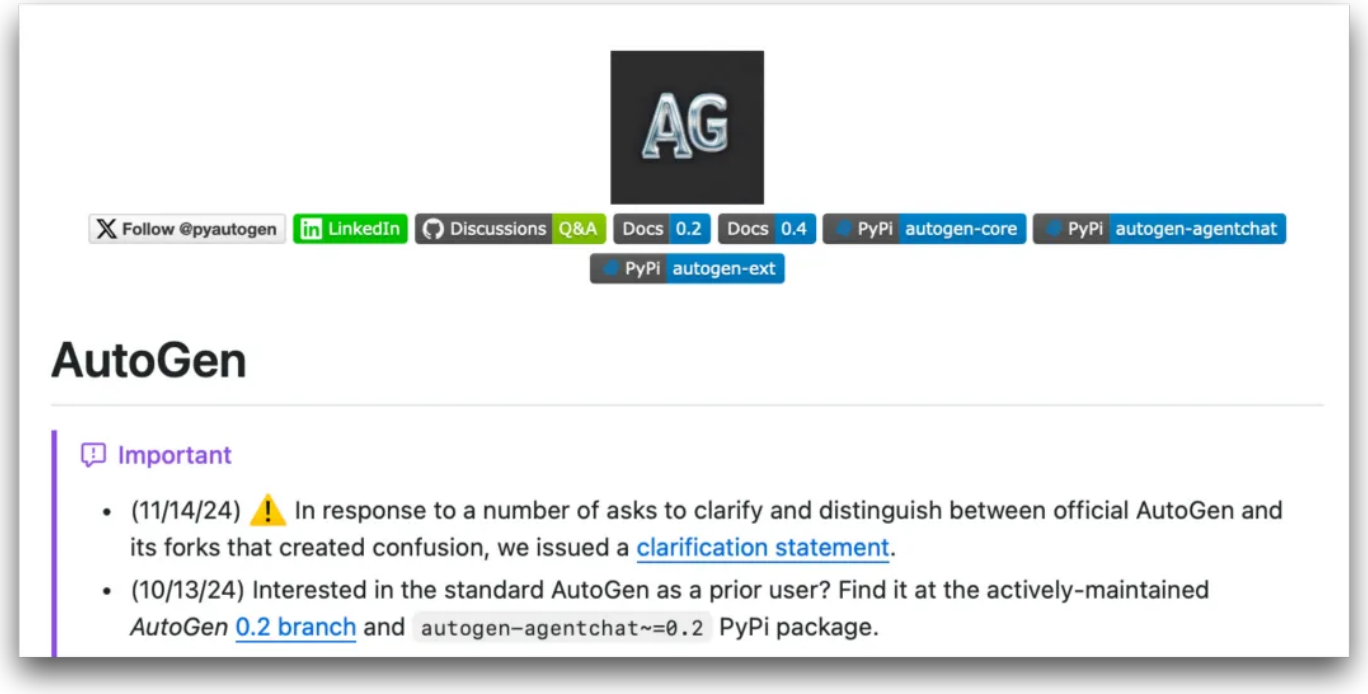
- 适用场景：简单工具链，无需复杂框架即可实现基础Agent功能，适合轻量级需求或学习底层原理。

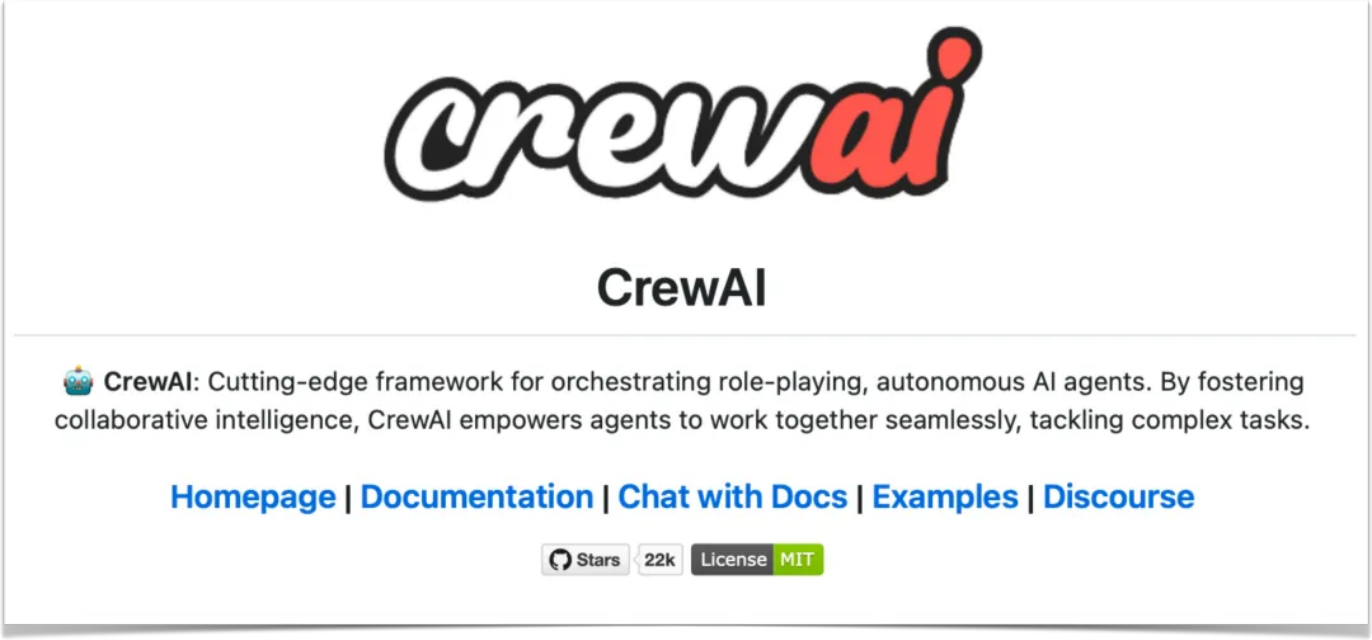
2.3、代码框架：工程项目落地的基石



- LangChain：
  - 模块化设计：通过Chain、Agent、Memory等抽象层实现可复用性
  - 典型应用：构建带历史记忆的问答系统（如ConversationalRetrievalChain）
- LangGraph：
  - 图计算引擎：使用StateGraph定义节点（Agent/Tool）和边（流转逻辑）
  - 优势场景：多阶段审核流程（如用户需求→方案生成→合规检查→结果输出）
- LlamaIndex：
  - 数据增强：支持RAG（检索增强生成）与结构化数据查询（SQL转换）
  - 最新进展：已与LangChain生态深度集成（如LlamaIndexTool）

2.4、Multi-Agent框架：复杂系统的解耦方案



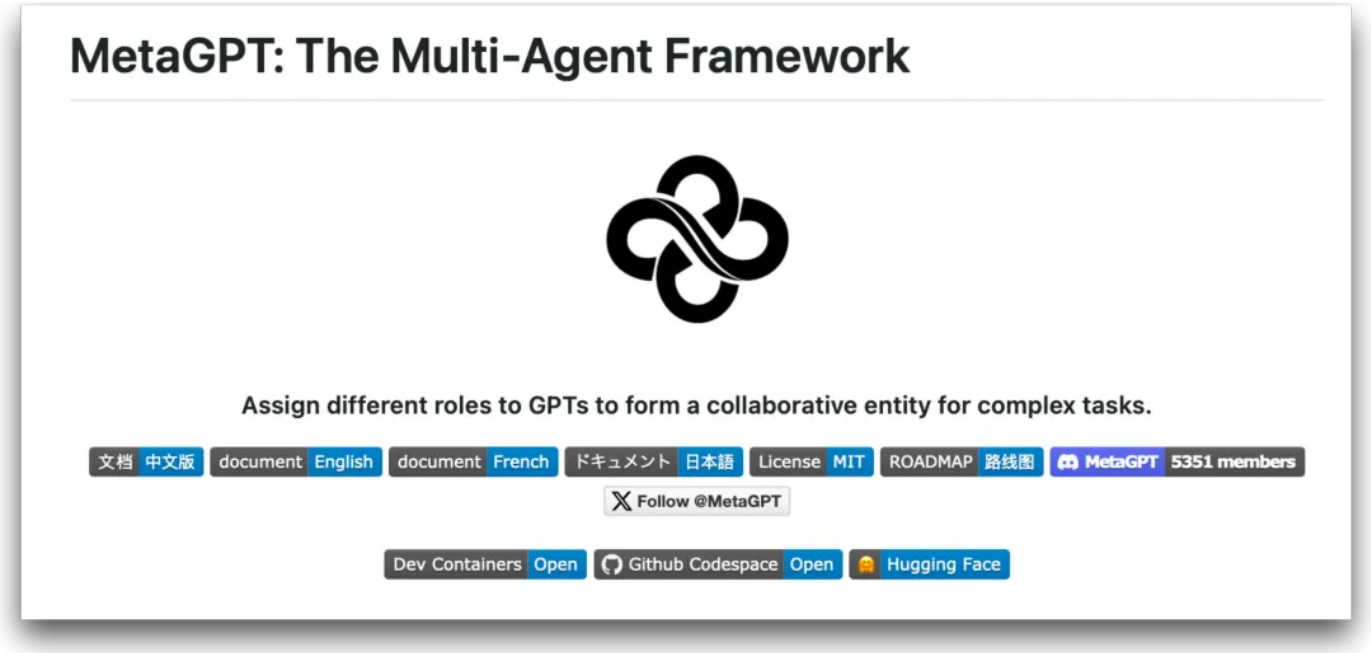


- AutoGen（微软）：
  - 角色化设计：内置UserProxyAgent（用户代理）、AssistantAgent（执行AI）、GroupChat（多Agent会话）
  - 实验数据：在复杂任务（如学术论文分析）中，多Agent协作相比单Agent错误率降低40%（来源：Microsoft Research）
- CrewAI：

- 任务流水线：通过Task定义原子操作，Crew编排任务依赖关系
- 特色功能：支持Tools优先级调度和资源竞争解决（如多个Agent争用GPU）

2.5、热门集成项目：垂直场景的快速落地

- MetaGPT：
  - 标准化流程：模拟软件公司角色分工（产品经理→工程师→测试员）
  - 开源数据：已实现超90%的简单Python脚本生成自动化（GitHub案例）
- ChatDev：
  - 领域聚焦：专为智能体开发优化的全流程框架（需求分析→代码生成→测试部署）



3、 框架选型策略

面对繁多的 Agent 开发平台如何选择？我们可以从模型支持、智能体能力、操作难易度、生态能力等方面进行对比。

- **模型支持**
  - 大模型厂商的 Agent 开发平台通常仅支持自家大模型。DeepSeek出圈后，国内 Agent 开发平台基本都已集成DeepSeek全系模型。
  - 原来的多模型 Agent 开发平台中，Dify和FastGPT支持国外和国内的多款大模型。Dify支持的大模型最多。扣子国内版仅支持国内的大模型。

- **智能体能力**
  - Agent 开发平台基本都能够提供 Agent 功能扩展，包括通过知识库实现检索增强，通过插件及API实现工具调用，通过工作流实现复杂任务执行，通过数据库实现信息读写等。
- **操作难易度**
  - 操作难易度是衡量 Agent 开发平台对非技术开发者友好性的重要指标。一个优秀的 Agent 开发平台能够提供直观、易于理解的操作页面和流程，以及高度模块化、即插即用的工具，以便让非技术开发者也能轻松开发出功能强大的 Agent 。
  - 一般可以从可视化程度、易理解性、即插即用性等方面评估 Agent 开发平台的操作难易度。比较而言，扣子的操作最便捷、最友好。
- **生态能力**
  - Agent 开发平台不仅是一个技术平台，还是一个生态平台。 Agent 开发平台的用户包括大模型供应商、 Agent 的个人开发者、 Agent 的企业开发者、Agent配套的插件/功能模块的专业开发者、API供应商、C端用户、B端用户等。
  - 生态能力决定了平台的吸引力、活跃度、可持续性，反映了平台的活力和潜力。像Dify和Coze等流行的 Agent 开发平台，相关的文章、教程、项目十分丰富。一个拥有丰富生态、开放合作、可持续发展且治理有方的平台，无疑是实现业务智能化转型的最佳伙伴。

总结关键考量因素：

- 模型能力依赖：如你提到的DeepSeek-V3和GPT-4的强Agent能力可减少框架的复杂度，而弱模型需依赖框架的工程化补偿（如ReAct模式）。
- 开发效率 vs 灵活性：低代码工具快但受限，代码框架灵活但学习成本高。

4、技术选型量化评估模型

维度	低代码框架	原生API	LangChain	多Agent框架
开发速度（1-5）	5	4	3	2
灵活性（1-5）	2	3	4	5
适合团队规模	<5人	<10人	5-20人	>20人
典型业务场景	MVP验证	轻量工具	企业应用	复杂系统

5、学习路径

- 初学者：
  - 从低代码平台（如Coze）或原生function calling入手
  - 入门工具：Coze（1天搭建客服Bot）+ DeepSeek-V3原生API（天气查询Demo）。
  - 关键目标：理解Agent的核心逻辑（感知→规划→执行→反馈）。
- 进阶开发：
  - 转向进阶版框架，学习模块化设计
  - 核心框架：LangChain（实现带记忆的问答系统)+ LangGraph（构建电商订单处理流水线）。
- 复杂系统
  - 尝试多Agent框架（如CrewAI）或参考AutoGen的设计理念