任务：

1. ~~智能体定义，完成~~
2. ~~Task定义，完成~~
3. ~~Tool示例定义=>~~
   1. ~~描述设置~~
4. **记忆、多轮对话**

**可以往后排**

1. 多种调用场景（crew配置不同），请求体格式设置
   1. 聊天（最高层对话的请求，不会指定具体的方法，接入聊天Agent可能委派其他Agent）
   2. 规划往返路线（往返层对话的请求，直接指定Agent和Task）
   3. 查询目标地点清单->规划路线（直接指定Agent、Task）
   4. 规划目标路线（直接指定Agent、Task）
   5. 查询地点详情（直接指定Agent、Task）
2. 实际方法包接入

郑贵宇同学的方法已通过单元测试，具体结构还待解析，传入AI可能需要去掉一些多余信息，实现一层封装。传给前端需要实现另外一层封装。

1. 服务器适配，流式输出

前端可以用正则匹配，前端有空可以实现打字机效果

1. 方法得到的其他信息直接传给前端（新建请求、图片、路线信息等）

此时正在流式传输，标记特殊信息然后直接把方法的结果以非流式的结构（完全生成后）于SSE流发送到前端

1. 减少不必要的中间过程 / 分离中间过程的输出

应该是做分离（将思考过程放在一个单独窗口，“显示思考过程✔”）

or前端在一次会话结束之后正则去掉思考过程只保留结果.(匹配上一次Final Answer)

1. 前端直接调用方法

后端为方法提供endpoint，封装方法输出（非流式，但速度不会慢）

* 1. 规划路线
  2. 查询地点详情（手动创建）

1. ~~数据库保存对话~~=>前端定时保存

~~等前端给一下数据结构~~

* 1. ~~新建chat表，字段uid、userid、history、agentMemory、time、lastTime~~
  2. ~~前端打开时查询user的所有chat~~
  3. ~~点击一个chat，前端加载，后端加载~~
  4. ~~前端自动发送保存信息，数据库接收~~
  5. ~~后端给数据库更新对话结果(messages，AgentMemory)~~
  6. ~~History的结构：等lzy~~

1. ~~其他数据库/后端（非智能体部分）的存储过程与触发器~~

~~这个用AI应该特别快~~

* 1. ~~广场~~
     1. ~~无输入返回推荐的帖子~~
     2. ~~有输入，关键词搜索帖子~~
  2. ~~帖子浏览页~~
     1. ~~输入帖子id，获取帖子具体信息(对应前端格式)+评论区信息~~
     2. ~~点赞与取消点赞~~
     3. ~~收藏与取消收藏~~
     4. ~~评论的发送，成功返回评论的uid~~
  3. ~~登录~~
     1. ~~登录验证，登录成功/失败返回信息~~
     2. ~~注册新账号请求处理，成功/失败返回信息~~
     3. ~~重置密码，成功/失败返回信息~~
  4. ~~个人资料~~
     1. ~~返回个人资料~~
     2. ~~上传个人资料~~
     3. ~~加载个人的所有帖子（通过用户搜索）~~
     4. ~~加载个人的收藏（通过收藏搜索）~~
  5. ~~通知~~
     1. ~~加载个人的所有获点赞和获评论~~
  6. ~~规划页~~
     1. ~~新建一个chat~~
     2. ~~获取所有chat~~
     3. ~~加载一个chat~~
     4. ~~向后端保存前端页面chat内容~~
     5. ~~将生成的post保存到个人主页~~