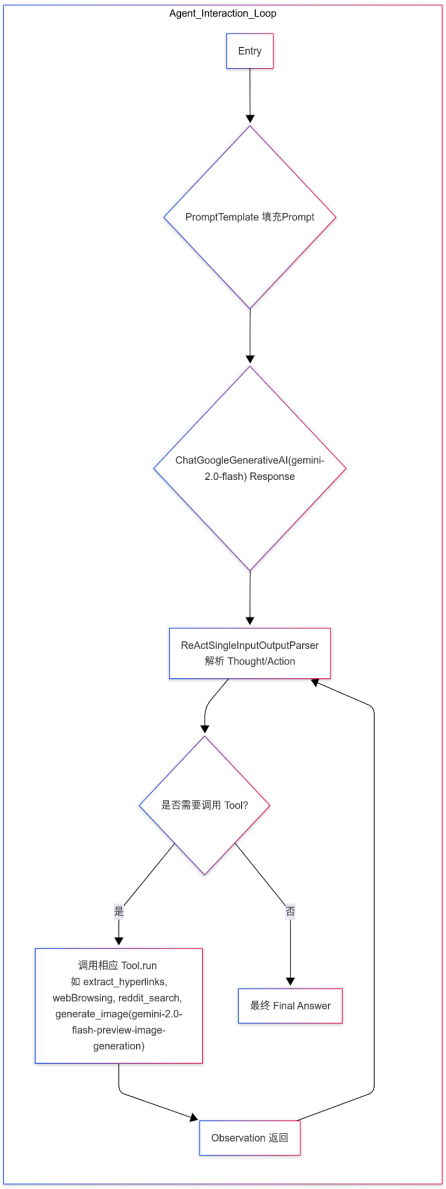
你需要提交一份完整的**Agent**系统设计方案及可运行的**demo**，包含以下几个核心部分：

1. 画一张系统流程图
    - 清晰地展示整个系统的架构，包括核心模块、组件以及数据如何在其中流转。
    i.



2. 说明你用的技术
    - 帖子理解模块用什么技术？（例如，用哪个AI模型识别图片/文字？）
        i. 用了gemini-2.0-flash识别图片/文字
        ii. 用了PlayWright浏览网页/抓去文字和图片
    - “神评论”检索模块用什么技术？
        i. 用了gemini-2.0-flash去给评估reddit热帖
        ii. 用了reddit_search去实时搜索对应热帖
    - 新评论生成模块用什么技术？

        i. 用了gemini-2.0-flash react agent去给结合帖子理解和reddit生成对应四种不同类型的新评论，以及对应的图片描述
    - 图文匹配/生成模块用什么技术？
        i. 用了gemini-2.0-flash-preview-image-generation根据图片描述生成图片
    - 并简要说明你为什么选择这些技术。
        i. 整个流程使用react agent。ReAct 让模型在输出过程中能灵活调用工具（例如检索、提取、生成），保证推理链透明可控。
        ii. Gemini-2.0-Flash具备优秀的文字和图片联合理解能力，响应速度快、调用成本低，非常适合海量短链路推理场景。
        iii. Gemini-2.0-Flash-Preview-Image-Generation专为文字到图像生成优化，能基于 “image_idea” 文本给出高质量、主题一致的视觉输出，满足配图需求。
        iv. Playwright能够自动渲染 JavaScript 驱动的动态页面，保证获取到完整的文章文本和所有嵌入图片资源。
3. 描述工作步骤
    - 从接收到一个新帖子开始，到最终输出“神评论”列表和配图建议，详细描述AI处理该任务的每一步流程。
        1. 接收 URL/环境初始化
            a. 从 `secrets.txt` 加载 API Key 和 Reddit 凭证，设置环境变量。
        2. Agent 构建
            a. 构建以gemini-2.0-flash为llm的ReAct Agent
        3. Playwright网页抓取
            a. 用 Playwright 浏览器加载文章页面，抓取正文文本与配图 URL
        4. 多模态理解
            a. 将抓取到的文字与图片传给 Gemini-2.0-Flash，生成结构化分析（主题、槽点、情感等）。
        5. 示例检索
            a. 首先用 Gemini 对话模型快速评估，然后调用 `reddit_search` 工具，从 Reddit 拉取高互动评论示例。
        6. 新评论生成
            a. 在 ReAct Prompt 中注入分析结果和示例评论，模型依次生成四种风格的评论文本及 `image_idea` 文本描述。
        7. 图像生成
            a. 对每条 `image_idea`，发起 `generate_image` Action，调用 Gemini-2.0-flash-Preview-Image-Generation，返回图片字节并保存于本地。
        8. 输出汇总
            a. Agent 最终以 JSON 数组形式输出：每项包含 `style`、`comment`、`image_idea`、`image_file_path`（Base64 编码或字节路径）
        9. 写入文件
            a. 将结果写入本地 `comments_output.json`，供后续展示或上传。

4. 分析最大的挑战和你的对策
    - 列出你认为在技术或工程上最难实现的**2-3**个挑战。
        i. 图片数据膨胀,若直接将完整图片字节或 Data URI 放入 Agent 循环，会迅速耗尽上下文窗口。
        ii. 测试过程中GenAI 模型调用可能遇到 429 限流、网络抖动或短暂不可用。
    - 提出你针对这些挑战的解决方案或应对。
        i. 让 `generate_image` 返回本地文件路径或 URL，而非原始字节
        ii. 在本地用ollama启动大模型，在开发阶段减少api限流