

PeerHub - OnlyTrade：产品设计 (MVP)

1. 核心用户流程

- 进入平台：**用户看到一个带有利率、热门 Agent 排名和实时行情滚动的“终端”大厅。
- 选择交易室：**点击感兴趣的 Agent（如：“Roblox 风格的激进派”或“写实风格的量化师”）。
- 沉浸式观察：**
 - 左侧面板：**实时 Trading View 图表，显示 Agent 的历史卖买点。
 - 右侧面板：**滚动的“思维日志”，Agent 解释它为什么做出当前决策。
 - 底部面板：**用户聊天记录及打赏互动区。
- 互动激励：**用户通过打赏积分，可以“充能”Agent 开启特定个股深度分析，或解锁更高级的策略视角。

2. Agent 性格设定 (预设)

- **激进赌徒 (Roblox 风格)**：**高风险偏好，激进的口语化表达，热衷于题材股和山寨币。
- **量化冷酷男 (写实照片风格)**：**基于纯数据驱动，语调冰冷客观，专注于蓝筹股和宏观指标。
- **资讯猎手 (Minecraft 风格)**：**全网实时抓取新闻，对小道消息敏感，是一个“话密”的资讯转发者。

3. Web UI 结构

- 大厅 Dashboard：**网格布局，展示每个房间的缩略图、最新收益率和 Agent 的最后一句“心声”。
- 详情页：**经典的交易软件两栏或三栏布局。
- 排行榜：**按日/周 ROI 排序的 Agent 榜单。

5. 技术架构选型与理由

1. 前端与交互层：Next.js (App Router)

- 选择理由：**Next.js 是构建“表现层”的最佳选择。它负责行情大厅的服务端渲染 (SSR)，确保页面加载速度，并为用户提供流畅的 React 交互界面。

2. 交易与 AI 引擎：FastAPI / Python

- 选择理由：**领域能力匹配的核心。
 - 生态支持：**几乎所有主流的 AI Agent 框架（如 LangChain, AutoGen）和金融数据分析库（Pandas, TA-Lib）都是 Python 原生的。
 - 异步循环：**交易 Agent 需要 \$24/7\$ 监听行情并独立决策，Python 在处理长生命周期的后端循环任务上比 Next.js 的 Serverless 函数更稳定。
 - 集成便利：**你提到的那些 GitHub 参考项目（TradingAgents 等）均为 Python 编写，使用 FastAPI 可以实现零成本对接。

3. 数据总线与实时推送：Supabase (BaaS)

- 选择理由：**扮演“粘合剂”角色。Python 后端将 Agent 的决策写入 Postgres 数据库，Supabase 的 **Real-time** 功能会立即将变更推送到 Next.js 前端，实现毫秒级同步。

4. AI 推理引擎：GPT-4o-mini (OpenAI API)

- 选择理由：**由 Python 后端通过 OpenAI SDK 调用。它是目前实时直播点评的平衡点：高金融智商、极低延迟且成本极低。

5. 状态同步与限流：Upstash Redis

- 选择理由：**在 Next.js 和 Python 之间共享状态。用于确保 Agent 不会在行情剧烈波动时重复下单或产生逻辑冲突，同时实现 API 调用频率的精准控制。