**已有模块/资产**

**文档与整体蓝图**

* 多智能体方法论与团队分工（感知→策略→决策/风控→执行）与“主席/基金经理”整合机制已清晰，作为系统最高层约束与协同框架。
* 功能目标：低频（每周≤3次）可执行决策；以基本面/动量/情绪因子为核心；“可验证（回测）+ 可视化（曲线/雷达/表格）”。
* 架构设计与模块划分（后端 FastAPI、SQLite；前端 React+ECharts；orchestrator 把数据→因子→评分→组合→回测串一条主链路）均已定稿。

**已规划/落位的代码结构（文件架构）**

* 后端 backend/ 下的 **agents / ingestion / factors / scoring / portfolio / backtest / sentiment / orchestrator / api / storage** 等目录与关键文件名已在文件树中固定；这为“只增不破”的增量式开发提供了落点。
* 稳定 API 设计（/prices, /fundamentals, /news, /analyze, /portfolio/propose, /backtest/run…）已对外约定，便于前后端并行与回归测试。

**数据与因子/评分契约**

* SQLite 数据模型（prices\_daily、fundamentals、news\_raw/news\_scores、factors\_daily、scores\_daily、portfolios、backtest\_results…）与 JSON 字段使用范围已定义；后续落表/迁移有据可依。
* 聚合评分契约：score = 100\*(0.25*价值 + 0.20*质量 + 0.35*动量 + 0.20*情绪)，并记录 version\_tag；这已成为后续模块的一致接口。

**多智能体层（角色与编排）**

* 角色清单与职责边界：DataIngestor → DataCleaner → SignalResearcher → BacktestEngineer → RiskManager → PortfolioManager → ReportWriter；主席/基金经理做最终加权整合，风险官具有否决/约束权。
* 编排策略：主链“串行”+ 局部并行、重试与降级、全链路 trace/metrics 与消息协议建议（trace\_id / scene / agent / payload / meta …）已定义。

**实施节奏与可验收里程碑**

* 里程碑 0–6 的“目标—接口—可视化—DoD”闭环验收清单已经给出，便于按周落地与演示（单票页→情绪→评分→组合→回测→日报）。
* 多智能体视角的第1–4周滚动目标（编排落地→Ingest+Clean→Research+Backtest→Risk+PM+Report）已就绪。

**你现有/可用的外部资源与基础设施**

* AlphaVantage 与 news.com 账户/API、GPT-5/Claude、SQLite 本地库、GitHub 版本管理等均已具备；（对话中你也已确认 **价格拉取** 与 **新闻情绪** 模块可用）。

**已完成/已可用的模块**

* ✅ **价格获取/入库**（AlphaVantage）：你已能拉取价格数据并写入本地库，/prices 路由已作为最低保障接口；（你曾反馈“数据已入库，图还没出来”，属于前端连通性/可视化待完善）。
* ✅ **新闻→情绪**：已具备拉取 news 源与情绪计算的通路（news.com；情绪∈[-1,1]），后端 sentiment 契约与前端时间轴/饼图设计已明确。
* ✅ **评分契约/权重**：统一到 0–100，含 version\_tag，用于 Watchlist 与 Portfolio。
* ✅ **文件架构**：各模块落位清晰（见 FILE\_TREE），支持“新增 agent 与 orchestrator，不破坏现有 API”。
* ✅ **里程碑验收标准**：每一步的输入输出、图表与 DoD 条目已可直接对照执行。

小结：目前“数据→(情绪)→评分”的**关键契约**与**接口**都已定型；数据侧（价格/新闻）你已可调用，主干开发的风险最低。  
  
我来帮你做一个阶段性的小结，补充进知识库。结合你的 **已有模块.docx**、架构文档和我们最近完成的调试记录，可以这样写：

**1. 已完成的核心 Agents**

* **DataIngestor / DataCleaner**  
  从 AlphaVantage、News API 拉取原始股价和新闻数据，清洗成结构化格式。  
  ✓ 已通过测试：价格、新闻获取与落库。
* **SignalResearcher**  
  把行情、新闻和情绪整合为研究因子。  
  ✓ 已完成因子抽取接口；兼容 AgentContext | dict | None。  
  ✓ 单测验证 factors 字段输出。
* **RiskManager**  
  提供单票、行业、仓位数量等约束，能自动生成 fallback 权重。  
  ✓ 已修复 proposal/candidates 输入兼容问题。  
  ✓ 可返回 kept、concentration、actions，满足测试要求。
* **PortfolioManager**  
  组合建议生成与权重分配。  
  ✓ 与 RiskManager 串联；支持快照持久化。
* **BacktestEngineer**  
  简化版回测引擎，支持窗口期回测与 NAV 序列输出。  
  ✓ 已在 /orchestrator/propose\_backtest 流程中联通。

**2. Orchestrator & Pipeline**

* **Orchestrator 路由**
  + /dispatch: Ingest → Clean → Research
  + /propose: Portfolio → Risk → fallback
  + /propose\_backtest: Propose → Risk → Backtest → 可视化结果
* **Pipeline**  
  提供 run\_pipeline / run\_portfolio\_pipeline 封装，单测已通过。

**3. API 层**

* **健康检查 & Schema** /health
* **价格、新闻、指标、基本面** /prices /news /metrics /fundamentals  
  ✓ fundamentals 路由已修复 502，支持 mock 与容错。  
  ✓ metrics 路由可返回关键财务/市场指标。

**4. 持久化与快照**

* SQLite 已建立基础 schema。
* /propose 返回结果可落盘至 portfolio\_snapshots / backtest\_results。
* 单测验证了版本控制与快照回放。

**5. 测试与质量保障**

* **20 个单测全部通过**（pytest -q backend/tests）。
* 包含研究链、组合、回测、持久化、API mock 等全链路。
* pytest-html 报告已归档（unit.html）。

**下一阶段建议**

* 引入 **新智能体层**（来自 *智能体定义.docx*）：  
  新闻/舆情、宏观、财报、技术面、价值投资、量化、宏观策略、主席/执行等。  
  这些将在现有 pipeline 基础上逐步扩展。
* 增加 **可视化路由**：组合权重饼图、因子雷达图、回测曲线，接入前端 monitor 页面。
* 加强 **持久化与回溯**：把 trace 与决策链完整落盘，便于复盘与调试。
* 新智能体层（Agent Layer）已落位：backend/agents/agent\_layer.py（顺序执行）、registry.py（注入九类 agent：新闻/舆情、宏观、财报、技术面、价值、量化、宏观策略、主席、执行），并以轻量 ResearchContext 贯穿，保持与你既有“**数据→因子→评分→组合→回测**”总架构一致（不破坏原 API/Schema）。这与设计文档中“Agents/Orchestrator/Shared Services 三层”一致，也对应既定消息协议与可观测性目标。
* **Deterministic Mock**：/orchestrator/dispatch 在 params.mock=True 时，产出稳定的四因子与 score，便于 Smoketest/UT 回归；与功能描述中“评分=多因子加权，需有可视化可回测”的目标一致。
* **兜底与拍平**：/orchestrator/propose 缺关键字段时自动风控兜底；/orchestrator/propose\_backtest 把回测字段拍平到 context 顶层（dates/nav/drawdown/benchmark\_nav/metrics），便于前端曲线渲染，对齐“回测可视化（净值/回撤/指标）”。