AgentForge Pre-Search Document

Domain: Finance

Codebase: Ghostfolio

Phase 1: Define Your Constraints

1. Domain Selection

Which domain?
Finance — portfolio intelligence and compliance-style analysis using Ghostfolio.

What specific use cases will you support?

- Portfolio performance across time ranges
- Asset allocation by class / sector / geography
- Concentration & diversification checks
- Dividend income analysis
- Transaction pattern summaries
- Multi-currency exposure
- FIRE progress tracking
- Market data lookup

What are the verification requirements for this domain?

- All financial figures must be tool-derived (no generated numbers)

- Deterministic recomputation of allocation & performance
- Concentration risk rule enforcement
- Confidence scoring when data is incomplete

What data sources will you need access to?

- Ghostfolio REST API (portfolio, orders, accounts, performance)
- Ghostfolio market data cache
- Seeded test portfolios for eval ground truth

---

2. Scale & Performance

Expected query volume?

- MVP: 100–1,000 users
- ~5 queries per user per day

Acceptable latency for responses?

- <5s single-tool
- <15s multi-step

Concurrent user requirements?

- MVP: 50 concurrent
- Design scalable to 1,000+

Cost constraints for LLM calls?

- Target ≤ $0.01 per query at scale

- Token usage tracked per request

---

3. Reliability Requirements

What's the cost of a wrong answer in your domain?

Medium–High — incorrect financial interpretation leads to user mistrust and bad decisions.

What verification is non-negotiable?

- Tool-backed outputs only
- Deterministic financial validation
- Allocation rule checks

Human-in-the-loop requirements?

Trigger when:

- Confidence < 0.7
- Missing market data
- Extreme concentration risk

Audit/compliance needs?

- Full trace per request
- Tool call logs
- Historical eval results

---

4. Team & Skill Constraints

Familiarity with agent frameworks?
Intermediate — choosing LangChain for fast delivery and strong tooling.

Experience with your chosen domain?
Intermediate finance knowledge; Ghostfolio provides structured domain logic.

Comfort with eval/testing frameworks?
High — CI-driven automated testing and deterministic validation.

Phase 2: Architecture Discovery

---

5. Agent Framework Selection

LangChain vs LangGraph vs CrewAI vs custom?
LangChain — fastest path to reliable tool calling and native observability.

Single agent or multi-agent architecture?
Single agent — domain is analysis, not role delegation.

State management requirements?

- Conversation memory per session
- Portfolio snapshot cache
- Redis-ready for production

Tool integration complexity?
Low — tools map directly to REST endpoints with typed schemas.

---

6. LLM Selection

GPT-5 vs Claude vs open source?
GPT-5 for strong reasoning, structured output, and reliability.

Function calling support requirements?
Strict schema tool calling required.

Context window needs?

- Conversation history
- Portfolio snapshot
- Tool outputs

Cost per query acceptable?
≤ $0.01 at scale.

---

7. Tool Design

What tools does your agent need?

- authenticate
- get_portfolio_holdings
- get_portfolio_performance
- get_portfolio_details
- get_orders
- lookup_symbol
- import_activities

External API dependencies?
Ghostfolio REST API only.

Mock vs real data for development?

- Real Ghostfolio Docker instance
- Seeded deterministic portfolios

Error handling per tool?

- 401 → re-authenticate
- 404 → user feedback
- timeout → retry once
- partial data → confidence drop

---

8. Observability Strategy

LangSmith vs Braintrust vs other?
LangSmith — tight LangChain integration and eval dataset support.

What metrics matter most?

- Tool selection accuracy
- Tool success rate
- Latency breakdown
- Token usage
- Final confidence

Real-time monitoring needs?
Trace inspection for debugging agent decisions.

Cost tracking requirements?
Per-request token and daily aggregate cost.

---

9. Eval Approach

How will you measure correctness?

- Deterministic recomputation of financial metrics
- Comparison with known seeded ground truth

Ground truth data sources?

- Seeded Ghostfolio portfolios
- Precomputed expected allocation & performance

Automated vs human evaluation?
Automated for CI; manual review for edge failures.

CI integration for eval runs?
Eval suite runs on every commit.

---

10. Verification Design

What claims must be verified?

- Portfolio value
- Allocation percentages
- Performance metrics
- Risk flags

Fact-checking data sources?

- Tool outputs only (Ghostfolio API)

Confidence thresholds?

- ≥ 0.7 required for normal response
- < 0.7 → uncertainty surfaced

Escalation triggers?

- Missing market data
- High concentration risk
- Incomplete portfolio state

Phase 3: Post-Stack Refinement

---

11. Failure Mode Analysis

What happens when tools fail?

- Retry once
- Return partial analysis with warning

How to handle ambiguous queries?

- Clarification question before tool execution

Rate limiting and fallback strategies?

- Cache portfolio snapshot per session
- Use last known market data with staleness notice

Graceful degradation approach?

- Structured partial responses instead of hard failure

---

12. Security Considerations

Prompt injection prevention?

- Ignore tool schema overrides from user input
- Tool access controlled by system layer

Data leakage risks?

- No persistent storage of portfolio data
- Session-only memory

API key management?

- Environment variables
- Never exposed to model

Audit logging requirements?

- Compliance rule triggers logged
- Tool access history stored

---

13. Testing Strategy

Unit tests for tools?
Yes — API wrappers and financial calculations.

Integration tests for agent flows?
Docker Ghostfolio + seeded portfolios.

Adversarial testing approach?

- Attempts to bypass verification

- Requests for investment advice

Regression testing setup?
50+ eval dataset executed in CI.

---

14. Open Source Planning

What will you release?
Ghostfolio financial agent evaluation dataset.

Licensing considerations?
MIT (no Ghostfolio source modification).

Documentation requirements?

- Setup guide
- Eval usage instructions
- Architecture overview

Community engagement plan?

- Publish dataset
- Share demo and results
- Enable reuse for other agents

---

15. Deployment & Operations

Hosting approach?

- Agent: FastAPI service
- Ghostfolio: Docker container

CI/CD for agent updates?

- GitHub Actions
- Automated eval runs

Monitoring and alerting?

- LangSmith traces
- Cost threshold alerts

Rollback strategy?

- Container version pinning
- Revert on eval regression

---

16. Iteration Planning

How will you collect user feedback?

- Thumbs up/down per response
- Correction submission

Eval-driven improvement cycle?

1. Detect failure in eval
2. Classify failure type
3. Improve tool / verification / prompt
4. Re-run CI

Feature prioritization approach?

- Eval failure frequency
- User feedback volume

Long-term maintenance plan?

- Version-lock Ghostfolio API
- Expand test dataset with real scenarios
- Add multi-portfolio support

Final Stack Summary

Agent Framework: LangChain
LLM: GPT-5
Observability: LangSmith
Verification: Deterministic + rule-based + confidence scoring
Eval: 50+ automated test cases
Deployment: Docker + FastAPI