**How I’m Building an Open‑Source AI Trading System (Inspired by Hedge Funds, Built for Everyone)**

**Introduction: Why Build an AI Trading System?**

Hedge funds spend millions building automated trading pipelines. But what if you could experiment with a modular, AI‑driven trading system on your own laptop?

That’s the idea behind [ai-trading-system](https://github.com/saurabhj9/ai-trading-system) — an **open‑source project that brings agent‑based workflows, backtesting, and state management into one lightweight framework**.

Inspired by [virattt/ai-hedge-fund](https://github.com/virattt/ai-hedge-fund), this system takes a lighter, modular, and faster approach aimed at being both **a learning resource** and **a foundation for collaboration**.

**Why Open Source?**

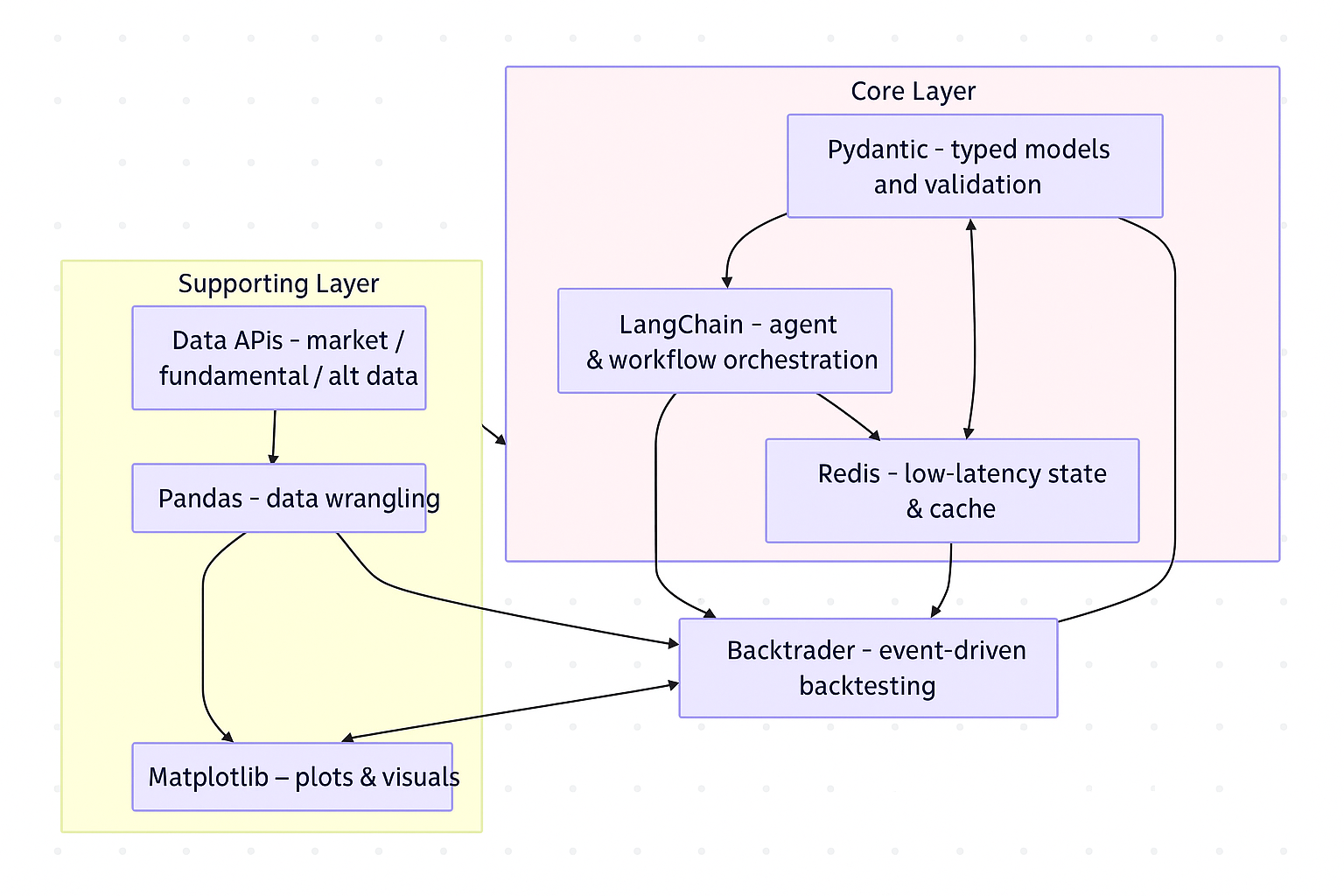
1. **Collaboration** → Trading systems require data engineers, ML researchers, and quants. Open sourcing invites all.
2. **Transparency** → Most frameworks are black boxes. This project is auditable and educational.
3. **Building in Public** → Show progress, recruit contributors, and make algo research accessible.

The goal is simple: **take hedge‑fund‑inspired ideas and build them into an open platform anyone can learn from**.

**Tech Stack: Modern but Minimal**

The curated stack is both modular and lightweight:

* **LangChain** → orchestrating workflows.
* **Pydantic** → strict data validation layer.
* **Redis** → lightning‑fast state manager.
* **Backtrader** → historical backtesting engine.
* *(Supporting)* Pandas/Matplotlib handle data wrangling and plotting.

**FIGURE 1 - AI Trading System Tech Stack" to the image**

**Design Principles: Hedge Fund DNA in an Open Source Project**

Building trading systems isn’t just about picking libraries — it’s about engineering discipline. The architecture is guided by a few hedge‑fund‑grade principles:

* **Modularity** → Every agent (technical, sentiment, risk, portfolio) is independent and replaceable.
* **Auditability** → All trade decisions are logged: inputs, reasoning, outputs. Nothing is a black box.
* **Resilience** → Multi‑level caching and provider failover ensure continuity even during API outages.
* **Scalability** → Orchestration patterns (sequential, parallel, hierarchical) let the workflow grow in complexity as needed.
* **Risk‑First Architecture** → A dedicated risk agent validates every decision before it can reach the portfolio.

These principles explain *why* the stack looks the way it does — speed alone isn’t the goal, it’s hedge‑fund‑grade robustness adapted for open experimentation.

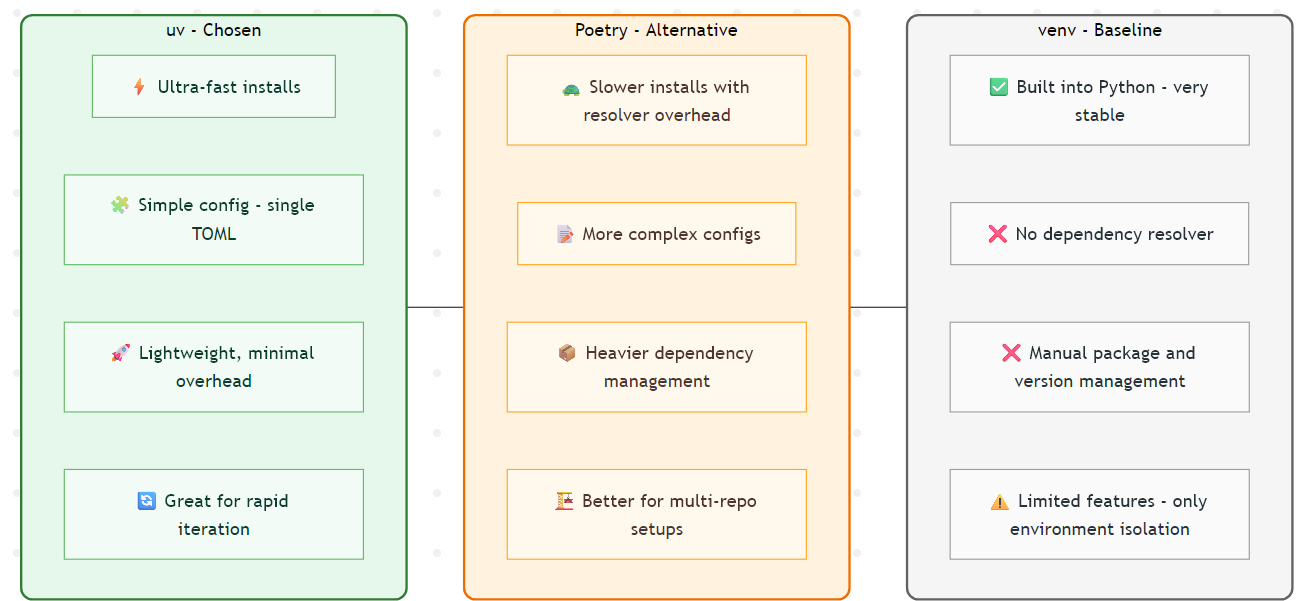
**Why uv Instead of Poetry?**

The inspiration used Poetry. I went with **uv** because:

* 🚀 Faster dependency resolution & installs
* 🧩 Simpler setup, less overhead
* ⚡ Ideal for rapid AI/finance prototyping

In a domain where you might test dozens of agent variations quickly, shaving off seconds per iteration compounds.

**FIGURE 2: uv vs Poetry vs venv comparison**



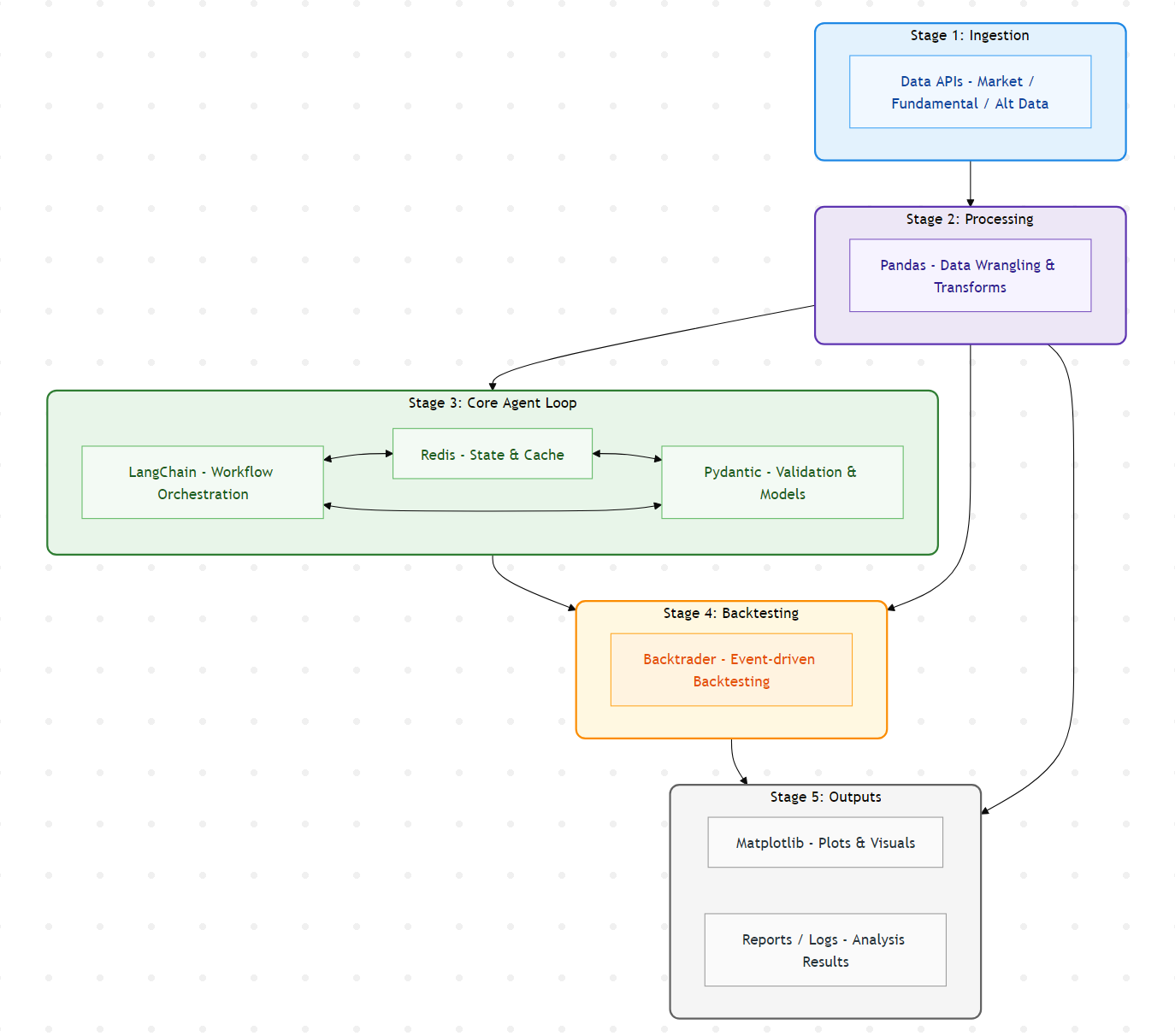
**Architecture: High-Level Overview**

The system reflects professional trading pipelines in five parts:

1. **Pricing / Data Ingestion**
2. **Analysis (Feature Engineering / TA)**
3. **Decision‑Making Agents**
   * Example: today a **Technical Analysis agent**.
   * Tomorrow: configure a **Reinforcement Learning agent** → no code refactoring needed.
4. **Backtesting Engine (Backtrader)**
5. **Execution Layer (Future)**

A **Redis-backed state manager** ties everything together with low‑latency communication.

**FIGURE 3: Modern systems diagram**



**Phase 1: What’s Completed**

Phase 1 was about building the **skeleton**:

* ✅ Tech stack & repo finalized
* ✅ Sequential agentic workflow skeleton
* ✅ Redis state manager implemented
* ✅ First Technical Analysis agent built
* ✅ Unit tests added

You can try Phase 1 today:

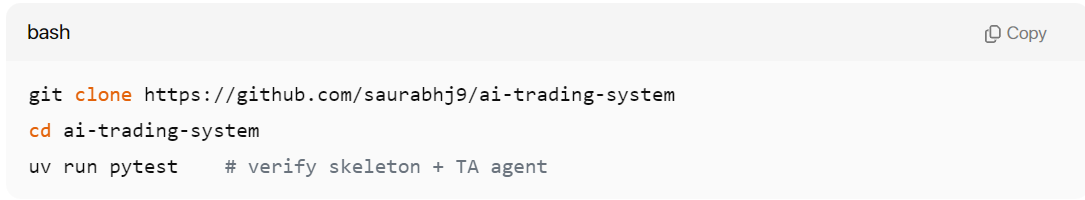
bash

Copy

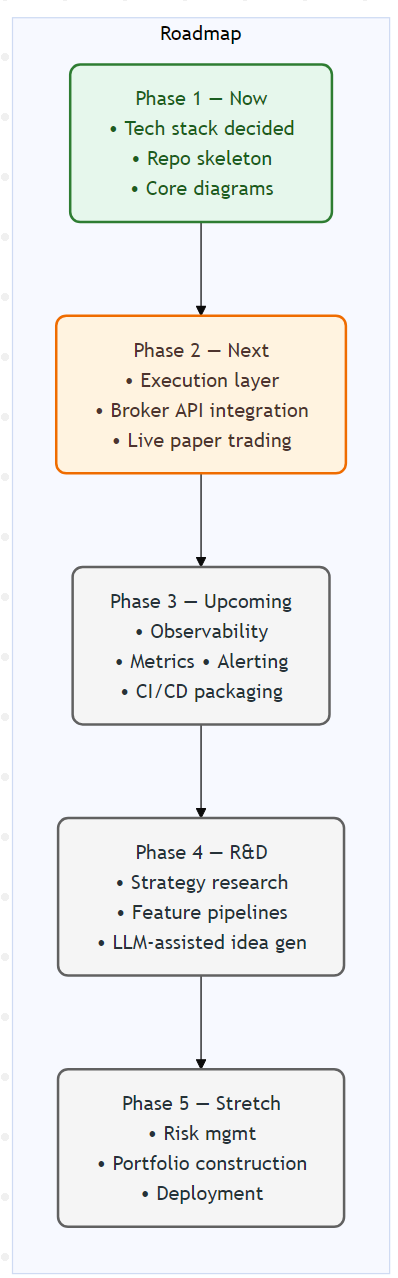
git clone https://github.com/saurabhj9/ai-trading-system

cd ai-trading-system

uv run pytest # verify skeleton + TA agent



**FIGURE 4: Roadmap progress**



**What’s Next?**

Phase 2 will push into:

* Reinforcement learning and smarter ML‑driven agents
* Richer feature pipelines & cross‑asset signals
* Multi‑asset backtesting
* The first **execution bridge** into live/paper trading

**Call to Action**

👉 [ai-trading-system](https://github.com/saurabhj9/ai-trading-system)

Follow updates on **X** and Medium. Contributions welcome from quants, ML engineers, or curious tinkerers. Let’s open‑source the next generation of AI trading research together.

**Disclaimer**

This project is **for research and educational purposes only**.  
All results are experimental and should **not be used for live trading without rigorous testing and professional oversight**.

**About the Author**

I build at the intersection of **AI, trading systems, and open‑source development**, sharing experiments in public to advance quant/AI research.

*Tags*: #AITrading #OpenSource #Quant #Python #Backtesting #Finance