AI Trader – Handover Document

1. Executive Summary (The What & Why)

Project Goal: To build a modular AI-driven trading platform that automates signal generation, portfolio management, and trade execution using probabilistic reasoning and AI agents.
Key Problem: Conventional bots rely on static indicators; this system integrates AI-driven probabilistic reasoning with quantitative backtesting for adaptive intelligence.
Main Features:

1. Strategy Engine – modular support for breakout, momentum, mean reversion.
2. Backtesting Core – unified engine for validation and diagnostics.
3. Probabilistic Agents – LangChain-powered reasoning nodes.
4. Risk Management – ATR-based position sizing and exposure control.
5. Observability – Grafana dashboards and structured logs.

2. Current Status (The Where)

Last Milestone: Fully functional backtesting loop; watchlist ingestion complete; all tests passing.
Current Focus: Implement filter strategies and probabilistic trade agents.
Key Decisions:

1. FastAPI remains orchestration layer.
2. ATR-based risk logic mandatory.
3. Monorepo continues until decomposition justified.
4. Docker-first model for local-cloud parity.
5. Grafana for observability.

3. Technical Context (The How)

Tech Stack:
Python 3.11+, FastAPI, pandas, numpy, LangChain, Alpaca, Telegram API, Azure Blob, Docker, Grafana, GitHub Actions.
Architecture Overview:
Market Data → Strategy Engine → Probabilistic Agent → Risk Controller → Execution → Observability & Storage.
Environment & Infrastructure Setup:

1. Environment parity maintained via Docker containers and Compose for local and CI builds.
2. CI/CD via GitHub Actions deploying to Azure App Service.
3. Environment variables managed through .env (excluded from git).
4. Azure Blob for raw data and Grafana for observability.
5. Multi-stage Docker build ensures lightweight deployable images.

Core File Structure:

```
ai_trader/ ■■■ app/ ■ ■■■ backtest/ ■ ■■■ strategies/ ■ ■■■ sources/ ■ ■■■
adapters/ ■ ■■■ api/ ■ ■■■ main.py ■■■ tests/ ■■■ docs/specs/ ■■■ Dockerfile ■■■
docker-compose.yml ■■■ requirements.txt
```

4. Tools & Resources

1. IDE: VSCode
2. Infra: Azure App Service
3. Storage: Azure Blob

4. Observability: Grafana + Prometheus
5. CI/CD: GitHub Actions

5. Recommended Next Steps

1. P0 – Implement Filter Strategies (volatility, liquidity, range).
2. P1 – Integrate Probabilistic Agents.
3. P2 – Extend Backtest Analytics.
4. P3 – Strategy Orchestration.
5. P4 – Docker Hardening and Reporting.