Backtesting Framework Development Report

Phase 1: Mean Reversion System - Final Report
Date: July 22, 2025

1. Executive Summary

The primary objective of Phase 1—to develop a robust, reusable, and transparent backtesting framework for mean reversion strategies—has been successfully achieved. The initial implementation faced significant challenges related to data integrity and architectural complexity, leading to unreliable results. Through an iterative debugging process, these challenges were overcome by re-architecting the system into a highly modular, configuration-driven engine.

The final framework is built on a clear separation of concerns, features a high-performance Numba-based core, and includes integrated diagnostics for transparent strategy evaluation. It is now stable, scalable, and ready for the analysis of new strategies in subsequent project phases.

2. Final Framework Architecture

The initial, monolithic approach proved brittle and difficult to debug. The final architecture solves these problems by delegating specific responsibilities to standalone Python modules, orchestrated by a central script and guided by strategy-specific configuration files.

Workflow Diagram:

main_backtester.py -> data_handler.py -> signal_generator.py -> backtest_engine.py -> Results

2.1. Configuration-Driven Design

The framework's flexibility stems from its configuration structure:

- **config.json (Global):** A single file containing global settings such as the database path, date ranges, and transaction costs.
- **strategies/ Directory (Strategy-Specific):** This directory holds a separate JSON file for each trading strategy. This allows all logic—including indicators to calculate, entry rules, and exit parameters—to be defined and version-controlled independently.

2.2. Modular Codebase

- **data_handler.py:** Solely responsible for loading raw price and indicator data from

the SQLite database and merging it onto a clean, 1-minute master index.
- **signal_generator.py:** The "brains" of the strategy logic. It dynamically calculates all required technical indicators in the order specified in the strategy config, handles potential data gaps (NaN values), and generates the final entry signals.
- **backtest_engine.py:** A library of high-performance backtesting functions. Each function is optimized with Numba and tailored to a specific type of exit logic (e.g., stop-loss/take-profit, time-based exits).
- **main_backtester.py:** The orchestrator. It parses user input, loads the correct configurations, and calls the other modules in sequence to execute the backtest.

3. Key Technical Achievements & Solutions

The primary challenge was ensuring data integrity throughout the pipeline. The following solutions were implemented:

- **High-Performance Core:** The main trade simulation loop in backtest_engine.py was JIT-compiled with **Numba**, enabling it to process millions of data points in seconds, eliminating the need for complex and error-prone vectorization for the event loop itself.
- **Robust Data Pipeline:** The final data pipeline follows a strict, logical order:
  1. Load all raw data.
  2. Forward-fill to create a unified 1-minute timeline.
  3. Pass the raw, clean data to the signal_generator.
  4. Calculate all indicators sequentially, ensuring dependencies are met.
  5. Drop any remaining rows with incomplete data before signal evaluation. This process completely resolved the persistent NaN (Not a Number) errors that plagued earlier versions.
- **Integrated Diagnostics:** The signal_generator was enhanced with a crucial diagnostic step. Before combining rules, it evaluates each condition individually and prints the number of times it was met. This tool was instrumental in identifying and resolving the data pipeline issues, providing essential transparency into the "black box" of signal generation.

4. Final Strategy Test Results: Quick Panic ES

With the framework now stable, the "Quick Panic ES" strategy was successfully backtested. The results below are from the final, validated engine.

Parameter: es_decline_pct	Total Trades	Win Rate (%)	Profit Factor	Total Return (%)	Max Drawdown (%)	Avg Duration (hrs)

-1	90	30.00	1.23	13.78	-11.52	78.5
-2	32	25.00	0.92	-2.17	-10.72	95.1
-3	6	0.00	0.00	-6.08	-5.08	120.3
-4	1	0.00	0.00	-1.05	0.00	144.0

Analysis: The diagnostic data and final results confirm the initial hypothesis: the -1% decline threshold provides the most opportunities and is the only profitable variation with the current rules. The negative returns for deeper pullbacks suggest the exit logic or other filters are not well-suited for higher volatility entries.

5. Conclusion

Phase 1 is complete. The project has successfully delivered a stable, modular, and high-performance backtesting framework. The system is no longer a collection of scripts but a cohesive engine capable of rapid and reliable strategy evaluation. The infrastructure is now in place to proceed with confidence into future phases of strategy development and refinement.