PortfolioPilot: An Agentic Platform for Financial Portfolio Management Algorithm Development and Evaluation

Jared Chan Xu Yang[1], **Haokai Ma**[2], **Yunshan Ma**[1]

[1]Singapore Management University
[2]National University of Singapore
jared.chan.2024@computing.smu.edu.sg, haokai.ma@nus.edu.sg, ysma@smu.edu.sg

Abstract

Developing new portfolio-management algorithms typically demands substantial programming effort, limiting rapid experimentation and excluding finance professionals without coding skills. Current robo-advisory tools offer pre-built but rigid strategies, restricting customization and experimentation. We introduce **PortfolioPilot**, an open-source, agentic platform that enables users to generate bespoke portfolio through natural-language descriptions. Leveraging the Anthropic Claude API, PortfolioPilot dynamically synthesizes executable TypeScript algorithms that run in the frontend with security validation. The system integrates real-time backtesting with historical market data, classical optimization algorithms (Markowitz, LSTM, ARIMA), and interactive performance visualizations.

Introduction

Algorithmic portfolio management is essential for navigating complex financial markets, leveraging methods from classical mean-variance optimization to modern deep learning and reinforcement learning approaches (Lee et al. 2024). However, developing such strategies typically requires extensive programming, creating barriers for researchers and financial practitioners without coding expertise. Existing platforms partially address this through pre-built algorithms, yet their rigid templates limit innovation and prevent the integration of advanced forecasting methods or custom logic. Conversely, flexible research environments assume strong programming skills, exhibiting a persistent accessibility gap.

To bridge these limitations, we introduce **PortfolioPilot**, an interactive web platform that combines user-specified natural-language strategies with built-in classical optimization models (*i.e.,* Markowitz, GMVP, and reinforcement learning (Lee et al. 2024)) and advanced forecasting techniques (*i.e.,* ARIMA, LSTM, and Autoformer (Wu et al. 2021)). Leveraging the Anthropic Claude API, PortfolioPilot dynamically synthesizes secure and executable TypeScript algorithms with multi-layer security validation, empowering users to generate bespoke strategies in the automatic manner. This platform facilitates real-time forecasting and backtesting against historical market data, providing customizable performance analytics. A user-friendly frontend and robust backend architecture collectively ensure its usability for both researchers and practitioners.

System Design

PortfolioPilot comprises four tightly coupled layers: the parameter-driven dashboard, an AI-assisted code-generation service, a curated library of built-in models, and a unified forecasting and backtesting engine. Figure 1 illustrates the end-to-end workflow: dashboard options and appropriate natural language prompts are standardized into a unified JSON format; when users request customized portfolio strategies, the Anthropic Claude API returns TypeScript code, which firstly undergoes security validation and then executed on the frontend end alongside canonical models that run in the Python backend. All artifacts are persisted to Supabase and can be revisited through the History interface. Source code and demos are publicly available at: https://github.com/JarudeC/portfoliopilot.

Dashboard and Parameter Capture. User can select up to eight DOW30 tickers, choose forecast and portfolio construction models from dropdown lists, and specify numeric quantities such as lookback window, evaluation horizon, and transaction cost. Selecting *Custom AI* in either model list triggers a textbox, where users can specify strategies in natural language. Default values guarantee a viable configuration, and a reset link restores baseline settings. The complete configuration is standardized into a JSON format and forwarded to the server.

AI-Assisted Strategy Generation. When the JSON format contains a request for customizing strategy, a Next.js API route composes a domain-specific prompt that embeds the user description alongside the dashboard parameters and submits it to the Anthropic Claude API. The response is a TypeScript function to compute weight or generate prediction, subjected to multi-layered safety checks that sanitize unsafe patterns, enforce execution constraints, and validate function signatures prior to execution. If no customized strategy is requested, this stage is bypassed and the relevant built-in model is instantiated directly in the Python backend.

Built-in Model Library. The library implements a range of optimization methods, including classical Markowitz mean–variance optimization (Markowitz 1952), global minimum-variance portfolio (GMVP), clustering-enhanced

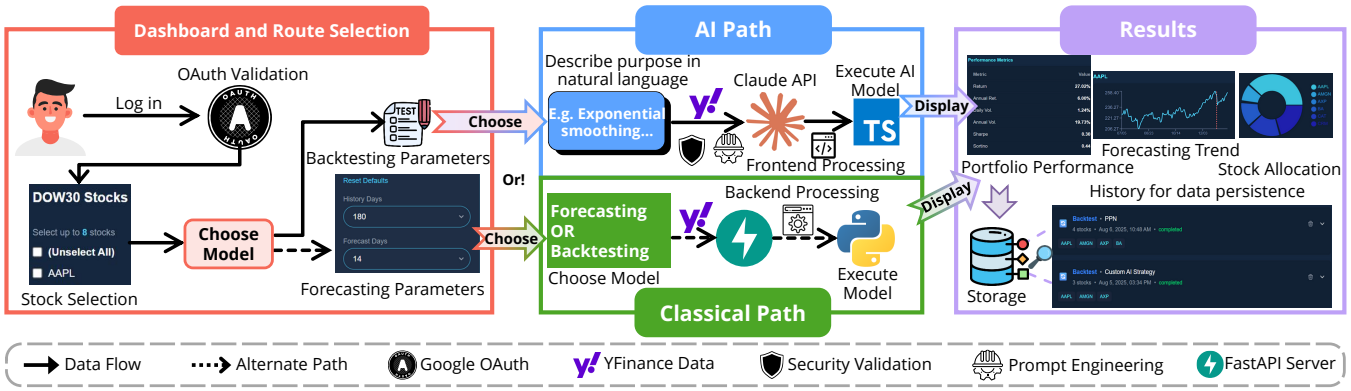

Figure 1: Illustration of the overall system architecture of our PortfolioPilot. Here, a user can select up to eight DOW30 tickers, choose the forecasting model and the portfolio construction model and then obtain the results through interactive equity curves, allocation heatmaps, and tabulated metrics (return, Sharpe, maximum drawdown, turnover, MAE, and MAPE where relevant).

GMVP variant (Park 2020), and reinforcement learning-based approaches such as MarginTrader (Gu et al. 2023), an A2C agents, and portfolio policy network (Zhang et al. 2020). Forecasting models, including ARIMA, LSTM, and Autoformer(Wu et al. 2021), adhere to a unified interface, whose results can be routed to any optimizer via a plugin signal handler, ensuring interoperability and fair evaluation.

Forecasting and Backtesting Engine. The Python backend retrieves daily prices from external data sources, executes rolling-window simulations under the user-specified cost structure, and computes the comprehensive metric suite. In addition to return, Sharpe ratio, maximum drawdown, and turnover, the system also reports mean squared error and mean absolute percentage error. Equity curves, allocation paths, and tabulated metrics are streamed to the frontend, where they are rendered with Recharts.

History and Session Management. Each completed run, whether generated by AI or drawn from the built-in library, is recorded in Supabase along with its metadata (*i.e.,* model type, asset list, parameter set, and execution timestamp). The History page employs lazy retrieval and infinite scrolling, permits filtering by model category, ticker symbol, or date range, and supports on-demand deletion. This persistent log facilitates longitudinal comparison of alternative strategies under identical market conditions.

System Implementation and Demonstration

The frontend of PortfolioPilot is built with Next.js, TypeScript, and React hooks, styled with Tailwind CSS, and relies on Recharts for client-side visualization. The backend uses FastAPI, integrating NumPy, Pandas, and PyTorch to support data ingestion, forecasting, and portfolio construction. User authentication is handled by Supabase OAuth; the metadata are stored in a PostgreSQL instance secured by JWT tokens. End-to-end latency remains below one second for strategy generation and under three seconds for rolling-window evaluation on a four-core CPU server.

When a user issues a customized query as "*create a momentum strategy on large-cap technology stocks with monthly rebalancing*", the dashboard serializes the parameters and sends them to the Next.js route. The service embeds the query into a finance-related template, and obtain the TypeScript implementing the required weight computation function from the Anthropic Claude API. Multi-layer security validation then executes the code in the frontend alongside the built-in models under the identical data, transaction cost, and horizon settings. We also conduct the unit test to verify that each generated strategy produces valid weight vectors and respects the imposed risk constraints.

Conclusion and Future Work

PortfolioPilot bridges the gap between investment intuition and executable portfolio logic through natural language strategy generation and unified evaluation. This system demonstrates that domain-specific prompt engineering, combined with multi-layer security validation, enables rapid prototyping of portfolio algorithms without traditional coding barriers. Current AI-generated strategies successfully implement foundational approaches including momentum, mean reversion, and risk parity variants, providing a solid foundation for portfolio construction.

Future development will focus on three key areas. First, advancing prompt engineering capabilities to support more sophisticated multi-factor models, options-based strategies, and complex derivatives positioning through enhanced natural language understanding. Second, expanding model diversity through integration with open-source and closed-source LLMs (*i.e.,* ChatGPT, LLaMA, and Qwen) and domain-specific financial LLMs to generate increasingly nuanced algorithmic approaches. Third, implementing collaborative research features including strategy sharing and comparative analysis tools to foster community-driven innovation in quantitative finance. The system's open-source foundation positions it as a research playground for algorithm experimentation and a stepping stone toward democratized quantitative finance tools. It further enables practitioners to focus on strategy logic rather than implementation complexity while continuously expanding the sophistication of AI-generated approaches.

Acknowledgments

This research was supported by the Singapore Ministry of Education (MOE) Academic Research Fund (AcRF) Tier 1 grant.

References

Gu, J.; Du, W.; Rahman, A. M.; and Wang, G. 2023. Margin trader: a reinforcement learning framework for portfolio management with margin and constraints. In *Proceedings of the Fourth ACM International Conference on AI in Finance*, 610–618.

Lee, Y.; Kim, J. H.; Kim, W. C.; and Fabozzi, F. J. 2024. An Overview of Machine Learning for Portfolio Optimization. *Journal of Portfolio Management*, 51(2).

Markowitz, H. M. 1952. Portfolio selection. *The Journal of Finance*, 7: 77–91.

Park, J. 2020. Clustering Approaches for Global Minimum Variance Portfolio. *arXiv preprint arXiv:2001.02966*.

Wu, H.; Xu, J.; Wang, J.; and Long, M. 2021. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34: 22419–22430.

Zhang, Y.; Zhao, P.; Wu, Q.; Li, B.; Huang, J.; and Tan, M. 2020. Cost-sensitive portfolio selection via deep reinforcement learning. *IEEE Transactions on knowledge and data engineering*, 34(1): 236–248.