

**alamSYS: DEVELOPMENT OF STOCK MARKET PRICE
FORECASTING SYSTEM USING DYNAMIC MODE
DECOMPOSITION, LONG SHORT-TERM MEMORY
WITH ARNAUD LEGOUX MOVING AVERAGE
CONVERGENCE-DIVERGENCE INTEGRATION**

A Special Problem
Presented to
the Faculty of the Division of Physical Sciences and Mathematics
College of Arts and Sciences
University of the Philippines Visayas
Miag-ao, Iloilo

In Partial Fulfillment
of the Requirements for the Degree of
Bachelor of Science in Computer Science

By
OLARTE, John Markton M.

Nilo C. Araneta, M. Sc.
Adviser

June 2023

CERTIFICATE OF APPROVAL

This special problem, entitled **alamSYS: Development of Stock Market Price Forecasting System Using Dynamic Mode Decomposition, Long Short-Term Memory with Arnaud Legoux Moving Average Convergence-Divergence Integration** prepared and submitted by **John Markton M. Olarte** in partial fulfillment of the requirements of the degree of **Bachelor of Science in Computer Science**, is hereby recommended for acceptance and approval.

Insert Name Here

Member, Special Problem Committee

Date Signed : _____

Insert Name Here

Member, Special Problem Committee

Date Signed: _____

NILO C. ARANETA, M. Sc.

Adviser, Special Problem Committee

Date Signed: _____

Accepted in partial fulfillment of the requirements for the degree of **Bachelor of Science in Computer Science**.

DR. ARNEL L. TAMPOS

Chairperson, Division of Physical Sciences and Mathematics

Date Signed: _____

Abstract

Investing in the stock market can provide significant benefits to both individuals and the economy as a whole. Many people, however, are hesitant to participate due to the high risk involved. This problem is worsened by a lack of access to accurate and reliable stock market information, especially in the Philippines. As such, the objective of this study was to develop alamSYS, a system that could collect and process stock market data in order to provide comprehensive stock position suggestions with the utilization of DMD-LSTM model and ALMACD. In addition, alamAPP, which showcases the main functionalities of the alamSYS, was also developed. Wherein, the results show that alamSYS, with its efficient and robust components, together with DMD-LSTM predictions along with ALMACD as a trading algorithm, has the potential to provide credible stock trading advice that yields high returns while minimizing potential risks. Moreover, the mobile-based test application's successful integration with external devices demonstrates the system's adaptability and versatility. Ultimately, the socioeconomic impact of alamSYS provide essential advantages to the stock market, individual investors or traders, and the Philippine economy as a whole by bringing about economic growth, increasing investor trust and participation, and limiting potential risks and losses.

Keywords: Systems Architecture, Computing Methodologies, Applied Computing, Human-Centered Computing

Contents

1	Introduction	1
1.1	Background and Rationale	1
1.2	Statement of the Problem	2
1.3	Significance of the Study	3
1.4	Objectives	4
1.5	Scope and Limitations	4
2	Review of Related Works and Literature	6
3	Materials and Methods	10
3.1	Development Tools and Software Requirements	10
3.1.1	Development Tools	10
3.1.2	Software Requirements	11
3.2	System Diagrams	11
3.2.1	Top-Level Overview Diagram of the alamSYS and Its Interactions to External Systems	11

3.2.2	Process Flow Diagram	13
3.2.3	Data-Flow Diagram (DFD)	20
3.2.4	Object Document Mapper (ODM) Diagram	25
3.2.5	Deep Learning Model Diagram	29
3.2.6	ALMACD Development Diagram	34
3.2.7	Docker-Compose Layer Diagram	35
3.3	Hardware Specifications	37
3.4	Methodology	37
3.4.1	Software Development Process	37
3.4.2	Procedures	45
4	Results and Discussions	49
4.1	alamSYS Documentation	49
4.1.1	Documentation for alamPREPROCESSOR	50
4.1.2	Documentation for alamAPI and alamDB	54
4.1.3	Documentation for alamAPP	66
4.1.4	Build and Deployment Guide	70
4.2	alamSYS System Tests Results and Discussions	73
4.3	DMD-LSTM Model Results and Discussions	81
4.4	ALMACD Results and Discussions	100
4.5	Results and Discussions for the Real World Application of alamSYS	103

5 Conclusions and Future Work	106
5.1 Summary of Findings	106
5.2 The Implications of alamSYS to the Society	109
5.3 Recommendations on Future Works	110
References	113
A Source Code Repository	120
B Raw Data Figures	121
B.1 Exploratory Stocks Data Graphs	121
B.2 Raw Model Testing and Cross-Validation Results	131
B.3 Model Testing Raw Test Results for DMD-LSTM	135
B.4 Daily Return Distribution of the Different Stocks	138
B.5 Raw alamSYS Test Data	152
B.5.1 Raw System Logs	152
B.5.2 PSEI Trading Baseline Data	158
B.5.3 Raw Real-world alamSYS Application	162
B.6 Gantt Chart	163
C Acknowledgements	166
D Author's Contact Information	168

List of Figures

3.1	Top-Level Overview of the alamSYS and Interactions with External Applications/Systems	12
3.2	Full Overview of the Process Flow Diagram for the alamSYS	14
3.3	Overview of the Process Flow Diagram for the Scheduler	15
3.4	Overview of the Process Flow Diagram for the Data Collector	16
3.5	Overview of the Process Flow Diagram for Data Processor	17
3.6	Overview of the Process Flow Diagram for the Deep Learning Model Applicator	18
3.7	Overview of the Process Flow Diagram for the Trading Algorithm Applicator	19
3.8	Overview of the Process Flow Diagram for the Database Updater	19
3.9	Context Diagram of the alamSYS	20
3.10	DFD of Diagram 0	22
3.11	DFD of Diagram 1	23
3.12	DFD of Diagram 1.2	24
3.13	DFD 2: Data-Flow Diagram for the alamSYS	25
3.14	Object Document Mapper for the alamSYS Database	26

3.15	Sample Buy Collection from the alamSYS Database	27
3.16	Sample Sell Collection from the alamSYS Database	27
3.17	Sample Info Collection from the alamSYS Database	28
3.18	Sample ML Models Info Collection from the alamSYS Database .	28
3.19	Sample Stock Risks Profile Collection from the alamSYS Database	29
3.20	DMD-LSTM Model Development Methodology for alamSYS . . .	30
3.21	ALMACD Development Methodology for alamSYS	34
3.22	Docker-Compose Layer Diagram for the alamSYS	36
3.23	Modified Agile Development Methodology	38
4.1	alamPREPROCESSOR Docker Container Directory Tree	53
4.2	alamAPI Web-based API Documentation Using Swagger	55
4.3	alamAPI Web-based API Documentation Using Redocs	55
4.4	Sample API Endpoint: Home Request	56
4.5	Sample API Endpoint: Buy (all) Request	57
4.6	Sample API Endpoint: Sell (all) Request	58
4.7	Sample API Endpoint: Stocks Info (all) Request	59
4.8	Sample API Endpoint: ML Model Info (all) Request	60
4.9	Sample API Endpoint: Stocks Risks Profile (all) Request	61
4.10	alamAPI End User UML Use Case Diagram	63
4.11	alamAPI Docker Container Directory Tree	64
4.12	MongoDB Compass Database Management of alamDB	66

4.13 alamAPP "Buy" Screen	67
4.14 alamAPP "Sell" Screen	68
4.15 alamAPP Prediction Graph Card	69
4.16 alamSYS Required Folder Structure	72
4.17 Deployment Load CPU Utilization of alamAPI and alamDB Over Time	80
4.18 Deployment Load Memory Utilization of alamAPI and alamDB Over Time	80
4.19 Comparison of MAPE Scores for DMD-LSTM Model Training Across Different Window Sizes	82
4.20 Actual vs Predicted Prices on AC for 100 days	85
4.21 Actual vs Predicted Prices for ALI over 100 days	85
4.22 Actual vs Predicted Prices for AP over 100 days	86
4.23 Actual vs Predicted Prices for BDO over 100 days	87
4.24 Actual vs Predicted Prices for BLOOM over 100 days	87
4.25 Actual vs Predicted Prices for FGEN over 100 days	88
4.26 Actual vs Predicted Prices for GLO over 100 days	89
4.27 Actual vs Predicted Prices for ICT over 100 days	89
4.28 Actual vs Predicted Prices on JGS for 100 days	90
4.29 Actual vs Predicted Prices on LTG for 100 days	91
4.30 Actual vs Predicted Prices on MEG for 100 days	91
4.31 Actual vs Predicted Prices on MER for 100 days	92

4.32 Actual vs Predicted Prices on MPI for 100 days	93
4.33 Actual vs Predicted Prices on PGOLD for 100 days	93
4.34 Actual vs Predicted Prices on PSEI for 100 days	94
4.35 Actual vs Predicted Prices on RLC for 100 days	95
4.36 Actual vs Predicted Prices on RRHI for 100 days	95
4.37 Actual vs Predicted Prices on SMC for 100 days	96
4.38 Actual vs Predicted Prices on TEL for 100 days	97
4.39 Actual vs Predicted Prices on URC for 100 days	97
4.40 MAPE Scores for 1 to 10 (Days) Successive Predictions	100
4.41 PSEI Stock Market Price Trend from March 24 to April 12, 2023	104
4.42 Comparison Between the Day-to-day Gains of alamSYS and PSEI	105
B.1 Opening, High, Low, and Closing Prices on AC	121
B.2 Opening, High, Low, and Closing Prices for ALI	122
B.3 Opening, High, Low, and Closing Prices for AP	122
B.4 Opening, High, Low, and Closing Prices for BDO	123
B.5 Opening, High, Low, and Closing Prices for BLOOM	123
B.6 Opening, High, Low, and Closing Prices for FGEN	124
B.7 Opening, High, Low, and Closing Prices for GLO	124
B.8 Opening, High, Low, and Closing Prices for ICT	125
B.9 Opening, High, Low, and Closing Prices on JGS	125
B.10 Opening, High, Low, and Closing Prices on LTG	126

B.11 Opening, High, Low, and Closing Prices on MEG	126
B.12 Opening, High, Low, and Closing Prices on MER	127
B.13 Opening, High, Low, and Closing Prices on MPI	127
B.14 Opening, High, Low, and Closing Prices on PGOLD	128
B.15 Opening, High, Low, and Closing Prices on PSEI	128
B.16 Opening, High, Low, and Closing Prices on RLC	129
B.17 Opening, High, Low, and Closing Prices on RRHI	129
B.18 Opening, High, Low, and Closing Prices on SMC	130
B.19 Opening, High, Low, and Closing Prices on TEL	130
B.20 Opening, High, Low, and Closing Prices on URC	131
B.21 Raw Model Scores for Baseline 5	131
B.22 Raw Model Scores for Baseline 10	132
B.23 Raw Model Scores for Baseline 15	132
B.24 Raw Model Scores for Baseline 20	133
B.25 Raw Model Scores for DMD-LSTM 5	133
B.26 Raw Model Scores for DMD-LSTM 10	134
B.27 Raw Model Scores for DMD-LSTM 15	134
B.28 Raw Model Scores for DMD-LSTM 20	135
B.29 Actual vs Predicted Closing Prices for DMD-LSTM 5 (Using Train Data from PSEI)	136
B.30 Actual vs Predicted Closing Prices for DMD-LSTM 10 (Using Train Data from PSEI)	136

B.31 Actual vs Predicted Closing Prices for DMD-LSTM 15 (Using Train Data from PSEI)	137
B.32 Actual vs Predicted Closing Prices for DMD-LSTM 20 (Using Train Data from PSEI)	138
B.33 Daily Return Distribution of AC	139
B.34 Opening, High, Low, and Closing Prices for ALI	139
B.35 Opening, High, Low, and Closing Prices for AP	140
B.36 Opening, High, Low, and Closing Prices for BDO	141
B.37 Opening, High, Low, and Closing Prices for BLOOM	141
B.38 Opening, High, Low, and Closing Prices for FGEN	142
B.39 Opening, High, Low, and Closing Prices for GLO	143
B.40 Opening, High, Low, and Closing Prices for ICT	143
B.41 Daily Return Distribution of JGS	144
B.42 Daily Return Distribution of LTG	145
B.43 Daily Return Distribution of MEG	145
B.44 Daily Return Distribution of MER	146
B.45 Daily Return Distribution of MPI	147
B.46 Daily Return Distribution of PGOLD	147
B.47 Daily Return Distribution of PSEI	148
B.48 Daily Return Distribution of RLC	149
B.49 Daily Return Distribution of RRHI	149
B.50 Daily Return Distribution of SMC	150

B.51 Daily Return Distribution of TEL	151
B.52 Daily Return Distribution of URC	151
B.53 Raw Risk Profile Scores	152
B.54 Raw Logs of Idle System Statistics	153
B.55 Raw Logs of Deployment System Statistics	154
B.56 Raw Logs of Data Collector Module (DCM) System Statistics . . .	155
B.57 Raw Logs of Data Processor Module (DPM) System Statistics . . .	156
B.58 Raw Logs of alamPREPROCESSOR System Statistics	157
B.59 Day 1 PSEI Trading Raw Data	158
B.60 Day 2 PSEI Trading Raw Data	158
B.61 Day 3 PSEI Trading Raw Data	159
B.62 Day 4 PSEI Trading Raw Data	159
B.63 Day 5 PSEI Trading Raw Data	160
B.64 Day 6 PSEI Trading Raw Data	160
B.65 Day 7 PSEI Trading Raw Data	161
B.66 Day 8 PSEI Trading Raw Data	161
B.67 Day 9 PSEI Trading Raw Data	162
B.68 Day 10 PSEI Trading Raw Data	162
B.69 Real World Application Raw Data Logs	163
B.70 Full Gantt Chart	163
B.71 Gantt Chart for Sprint 1	163

B.72 Gantt Chart for Sprint 2	164
B.73 Gantt Chart for Sprint 3	164
B.74 Gantt Chart for Sprint 4	164
B.75 Gantt Chart for Sprint 5	165
B.76 Gantt Chart for Sprint 6	165

List of Tables

3.1	Collected Market Data Details	30
3.2	ALMA Grid Search Parameters	35
3.3	Summary of Sprints and Activities	39
4.1	Summary of Risk Profile for Each Stock in the alamSYS	62
4.2	Idle System Average Resource Usage Statistics	74
4.3	Internal Load Average Resource Usage Statistics	76
4.4	Deployment Load Test Results (Buy Requests)	78
4.5	Deployment Load Test Results (Sell Requests)	78
4.6	CPU and Memory Utilization Statistics of alamAPI and alamDB Under Deployment Load Testing	79
4.7	DMD-LSTM Training Error Metrics Scores for Different Window Sizes	81
4.8	Baseline LSTM Training Error Metrics Scores for Different Window Sizes	83
4.9	DMD-LSTM Cross-Validation Error Metrics Scores	84
4.10	DMD-LSTM Successive Predictions	99

4.11 Optimal Alma Parameters Validation Results	101
4.12 Return Performance Comparison Between alamSYS and PSEI . .	103

Chapter 1

Introduction

1.1 Background and Rationale

Investing in the stock market has numerous benefits for both the person and the economy. However, people are hesitant to begin because of its complexity. Furthermore, the stock market instruments available to the public in the Philippines do not cater to new investors: (a) features are frequently hidden behind a pay wall; and (b) provided stock market data is overwhelming and confusing.

As a result of these problems, the researcher created a system to provide a publicly available, easy-to-use, data-driven, and accurate stock market tool designed primarily for the Philippine Stock Exchange.

The system, referred to as 'alamSYS' throughout this study, differs from the few current stock market instruments in the Philippines. In comparison to these other tools, alamSYS provides a simplified way to analyze stocks by simply looking at its predictions of whether a stock should be bought or sold, and it is up to the users to further examine the stock using different analysis tools to make a more informed decision.

Aside from that, alamSYS allows users to automate the process of selecting stocks on the Philippine Stock Exchange by utilizing DMD-LSTM and ALMACD, which are publically accessible via the system's web API component. Moreover, users can also implement their own machine learning models and trading algorithms, as the source code of the alamSYS is open-sourced. Further details regarding the alamSYS and its components are discussed in Chapters 3 and 4.

1.2 Statement of the Problem

Despite the clear and functional benefits of investing in the stock market, many Filipinos remain hesitant to do so for the following reasons: (a) the difficulties that come with learning the fundamentals of effective stock investing; (b) the time-consuming nature of technical and fundamental analysis, especially for students and working people on a tight schedule; and (c) the increased financial risk associated with stock market volatility, as well as the potential for emotional decision-making to jeopardize investments.

Moreover, the lack of free and publicly available stock market predictive systems or tools currently creates a gap in the information available to the public when compared to large private individuals or institutions. Where, the public is disadvantaged by this information gap (Kim, 2022).

These factors (*along with other external and internal factors not listed above*) contribute to a lack of confidence and understanding among potential investors, making it difficult for them to take advantage of the opportunities offered by the stock market.

In response to these problems, the author conducted this special problem to explore and improve the utilization of machine learning models in the Philippine Stock Market, particularly deep learning models such as LSTMs through the development of the alamSYS.

1.3 Significance of the Study

The significance of this developed system lies in its ability to greatly benefit the stock market, individual investors, and the economy as a whole. In particular, the system addresses the following technological gaps: (a) The lack of free and publicly available stock market prediction systems or tools; (b) The time and resources required to study complex traditional market analysis tools, such as fundamental and technical analysis; (c) The potential for inaccurate market decisions leading to significant investment losses; and (d) The hesitancy of the Filipino public to begin investing in the Philippine stock market.

Furthermore, contributions of the system to data-driven investing, financial protection and management, and economic development could provide a valuable resource for investors while also promoting financial stability and growth. Wherein, the creation of publicly accessible data-driven investing tools or systems may encourage more Filipinos to participate in the market and take control of their own financial future. Overall, the completion of this special problem has the potential to have a significant impact on the Philippine stock market and economy.

Specifically, this study is significant for the following reasons:

- (a) The development of the alamSYS aims to provide the following benefits to the Filipino people: (1) Provides Filipino investors with fast, accurate, and relevant information for effective decision making in the stock market; (2) Provide an intuitive and user-friendly application interface to facilitate data-driven investment decisions; and (3) Provide a platform for all investors to participate in the stock market, regardless of their investment knowledge, educational attainment, and societal status.
- (b) The development of the alamSYS, aims to provide the following benefits to the future developers or researchers: (1) Provide a developmental framework for innovative stock market tools; (2) Provide a system, where they can apply and test new machine learning models; and (3) Provide an avenue to make stock market analysis and investing more accessible to the public.

- (c) The development of alamSYS could stimulate economic recovery and development in the Philippines by increasing the number of local investors. The benefits of the system could encourage more people to invest in the stock market, leading to a multiplier effect that could benefit the economy in several ways, such as job creation, lowering of unemployment rates, fast developments and innovations in various industries, and increased consumer spending - as indicated in a study by Qamruzzaman and Wei (2019).

1.4 Objectives

This special problem aims to develop a system that makes investing easier, more publicly available, data-driven, and more approachable by minimizing both the time required for stock price trend analysis, and potential financial risk by using Dynamic Mode - Long Short-Term Memory (DMD-LSTM) and integrate Arnaud Legoux Moving Average Convergence-Divergence (ALMACD) as a trading algorithm. More specifically, it aims to do following:

- (a) Develop a system called alamSYS that can collect and process stock market data in order to provide comprehensive stock position suggestions.
- (b) Develop a Stock Market Price Trend Forecasting Deep Learning Model and integrate the forecasting with ALMACD as a trading algorithm and basis for entry and exit positions.
- (c) Finally, develop a mobile-based test application, which from hereon maybe referred to as: alamAPP, to showcase the main functionalities of the developed RESTful API.

1.5 Scope and Limitations

This study was limited only within the companies listed in the Philippine Stock Exchange. Specifically, 20 high volume trade stocks from the year 2021 to 2022 were selected, which are as follows: (1) MEG, (2) JGS, (3) BDO, (4) FGEN, (5)

ICT, (6) ALI, (7) SMC, (8) TEL, (9) GLO, (10) BLOOM, (11) RLC, (12) MER, (13) AC, (14) PGOLD, (15) LTG, (16) MPI, (17) AP, (18) RRHI, (19) URC, and (20) PSE Index will be included in the system, instead of the total 286 listed under the Philippine Stock Exchange.

Chapter 2

Review of Related Works and Literature

Many online stock market analysis tools have recently been available, including: (a) Investa Screener; (b) TradingView; (c) Stock Rover; and (d) Investagrams. These tools can assist investors in performing fundamental and technical analysis on various equities (Benignos & Navarro, 2020; Carey, 2023; DayTradingz, 2023; Investagrams, 2023).

The use of technical analyses, in particular, provides the following benefits: (a) the ability to generate profitable buy and sell signals; (b) the performance of the technical analysis can be utilized and even show similar performance on different markets; (c) can be used to test the significance predictive power of trading rules; and (d) provides an easy-to-understand data analysis and visualization tool in conjunction with fundamental stock analysis (Abbad, Fardousi, & Abbad, 2014; Priatmojo & Sari, 2022; Ausloos & Ivanova, 2002; Tangsripairoj, Sutanaphanit, Treenitikul, Wongaod, & Maneeintr, 2020).

However, the studies cited above show the following limitations to using traditional stock market analysis techniques through the use of online tools: (a) the amount of data becomes overwhelming, especially for new investors/traders; (b)

relying on fundamental and technical analysis alone may lead to missed opportunities or incorrect market decisions; and (c) these tools and techniques do not eliminate emotional trading.

In recent years, 'artificial intelligence' (AI) has grown in popularity as a tool for nearly every endeavor. Particularly in recent years, machine learning techniques are increasingly being utilized in the stock market, specifically for: (a) stock price predictions; (b) stock movement/trend prediction; and (c) stock portfolio management (Kumbure, Lohrmann, Luukka, & Porras, 2022; Strader, Rozycski, Root, & Huang, 2020; Soni, Tewari, & Krishnan, 2022; Rea, 2020; Y. Guo, 2022; Budiharto, 2021).

Moreover, there are currently no machine learning-based technologies accessible in the Philippines, which according to studies aid to: (a) improve decision accuracy and efficiency; (b) increase risk management; and (c) improve trader performance in the short term (Lu, Tao, & Zhao, 2022; Shaheen & Arshad, 2020; and, 2022; Prasad & Seetharaman, 2021; Zhang, 2022; Obthong, Tantisantiwong, Jeamwatthanachai, & Wills, 2020). In comparison, a number of studies in other nations have produced machine learning models for stock market forecasting, some of which are as follows:

The study of Mann and Kutz (2015) in the utilization of DMD for financial stock market prediction has become the foundation of the study by Savaş (2017) on the algorithmic trading strategies with Dynamic Mode Decomposition for the Turkish Stock Market. Wherein, based on their results they found out that the timing of DMD analysis was not significantly accurate, as such they have used a simple moving average with genetic algorithm to improve the market timing of DMD, which prevents 80% of the false trade signals.

In the study by Cui and Long (2016), they have found that DMD was able to capture the dynamic patterns of the Chinese Stock Market, especially in a sideway trending market. Fraz, Fatima, and Uddin (2022), researchers from Pakistan and China, was able to create a model for the KSE-100 and SSE-100 utilizing machine

learning techniques, the GARCH family, and non-linear regime-switching models. Based on Akaike's Information Criteria (AIC) and Bayesian Information Criteria (BIC), their findings revealed that the standard GARCH model is the best-fitted model, and that the LSTM model outperforms the other models in predicting the SSE-100.

More recently, Sarangi, Mohanty, Mohapatra, and Sahu (2023), researchers from India were able to perform and analyze the activity of the Bombay Stock Exchange (BSE) using three machine learning models: linear regression, polynomial regression, and support vector regression.

These studies demonstrate how widely machine learning models are used in the stock market. However, it also demonstrates some limitations of the models, such as: (a) these studies are focused outside of the Philippines, so the model developed may not apply directly or well to the Philippine Stock Exchange; (b) as the study of Shaheen and Arshad (2020) suggests, the results of these models may not necessarily apply to systems in the real world, where the models may be unable to adapt to other situations and conditions; (c) the utilization of these tools has its limitations. First, these tools are only based on the past market data, and may not be able to take account for other events and technical aspects, that can affect the stock market (Concodia, 2020); and (d) most of these tools are closed off to the public, or often behind a payment or subscription walls, limiting the resources available for them. Lastly, there are no existing integrations of these developed machine learning models with other traditional stock market analysis tools, making them somewhat ineffective when utilized in a real-world scenario (Chhajer, Shah, & Kshirsagar, 2022; Zou et al., 2023).

These points are essential to take note, since investing in the stock market brings significant financial benefits to individuals such as: (a) it helps alleviate the effects of inflation over time. Whereas, it was noted that the Philippines' inflation rate as of April 2023 was 6.6% (National Economic and Development Authority, 2023), while savings account deposit interest rates are only 1-3% annually (Bureau of the Treasury Bangko Sentral ng Pilipinas, n.d.). Meaning, savings

in deposit banks may not keep pace with inflation, potentially reducing an individual's purchasing power over time (Royal Bank of Canada Direct Investing Inc., n.d.; EdwardJones, n.d.); and (b) investing in the stock market can provide individuals with the opportunity for significant capital growth without the need for direct investment involvement in business operations (U.S. Securities and Exchange Commission, n.d.).

Also, the stock market is widely acknowledged to play an important role in economic growth because it allocates and provides capital to businesses, which drives economic activity and growth. This is evident from the fact that stock market performance is frequently correlated with the gross domestic product (GDP) of the country (Trade Brains, 2022; Hall, 2022; Bae & Kang, 2017). Furthermore, historical stock price trends can provide insight into broader economic movements (Campbell, 2021).

In a study conducted by Balaba (2017), they discovered that the stock market has a positive impact on the Philippines' economy. The study's findings showed that as the stock market rose, the unemployment rate fell. This is because the performance of the stock market leads to job creation, which in turn leads to economic growth. This, in turn, drives economic growth. This relationship was observed in the Philippines from 2007 to 2017.

Overall, this review of the related literature and studies, shows that one of the issues confronting investors in the Philippine Stock Exchange is the scarcity of resources and instruments for making market decisions. Other countries, on the other hand, have begun to adopt machine learning techniques for stock market prediction and analysis, allowing for more accurate decision-making and lowering the chance of poor investment outcomes. However, using these machine learning models does not guarantee profitable market decisions in the actual world. As a result, alamSYS was created, with the goal of implementing a deep learning model for the Philippine Stock Exchange and deploying this model for use in the real world, accessible through a web-based API.

Chapter 3

Materials and Methods

This chapter discusses the materials and methods used for the design and development of the system: alamSYS. Specifically, the following are discussed in this chapter: (a) Development Tools and Software Requirements; (b) System Diagrams; (c) Hardware Requirements; and (d) Methodology.

3.1 Development Tools and Software Requirements

The development of the alamSYS utilized the following development tools and software requirements:

3.1.1 Development Tools

To develop the alamSYS, the following development tools were utilized: (a) Visual Studio (VS) code, as the project's integrated development environment; (b) MongoDB Compass, used for managing the alamDB; and (c) Github, as the version control system and project's source code repository.

3.1.2 Software Requirements

In the development of the alamSYS and its components the following software and their respective versions were utilized: (a) Python *versions 3.9.10 and 3.11.1*; (b) MongoDB; (c) Jupyter Notebook; (d) Docker; (e) Docker-Compose; (f) Dart and Flutter; (g) Git; and (h) Github.

Moreover, the following Python libraries and modules are also utilized: (a) FastAPI *version 0.85.0*; (b) mongoengine *version 0.24.2*; (c) json; (d) datetime; (e) os; (f) schedule *version 1.1.0*; (g) requests *version 2.28.1*; (h) numpy *version 1.23.5*; (i) tensorflow *version 2.11.0*; (j) matplotlib *version 3.7.0*; (k) pyDMD *version 0.4.0post2301*; and (l) pandas *version 1.5.3*.

Additionally, the following flutter packages are utilized for the development of the alamAPP: (a) http *version 0.13.5*; (b) path_provider *version 2.0.13*; (c) syncfusion_flutter_charts *version 20.4.52*; and (d) lottie *version 2.2.0*

3.2 System Diagrams

In this section, the appropriate system diagrams is shown and discussed. This shall help in the understanding of the system's features, data flow, and processes.

3.2.1 Top-Level Overview Diagram of the alamSYS and Its Interactions to External Systems

Figure 3.1 shows the top-level overview of the alamSYS and its interactions to any third-party or external applications.

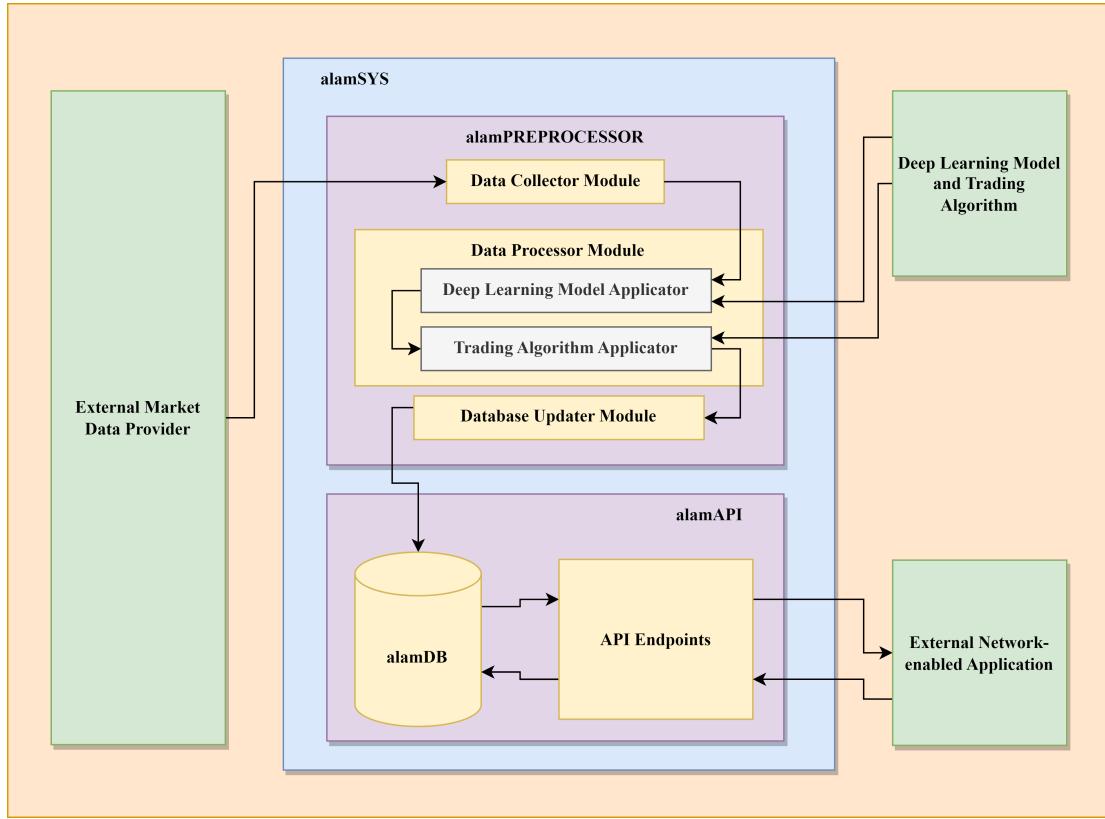


Figure 3.1: Top-Level Overview of the alamSYS and Interactions with External Applications/Systems

As shown from the figure above, the alamSYS is connected to three external entities: (1) External Market Data Provider, which provides the system with the needed historical market data; (2) Deep Learning Model and Trading Algorithm, in the case of this special problem, a deep learning model was developed and utilized by the system, however as previously discussed the system is created to accept any other machine learning models and or proprietary trading algorithms that other developers may or want to develop in the future; and (3) External Application, which can be a web-based or mobile-based application, that utilizes the functionalities provided by the alamSYS, through the API endpoints.

On the middle of the diagram the alamSYS is observed to have three main components, namely, (1) alamPREPROCESSOR, which is further divided into sub-components: (a) Data Collector Module (DCM), which collects the data from

the external market data provider; (b) Data Processor Module (DPM), which processes the historical market data collected by applying the developed machine learning model and sending it to the (c) Database Updater Module (DUM); (2) alamDB, which is based on MongoDB, which is a document-based and non-relational database; finally, the database is connected to the (3) alamAPI, which contains the API endpoints which processes the request and responses of the system to any external application connected to the API via a network.

3.2.2 Process Flow Diagram

The diagram shown in Figure 3.2 the different processes that the system undergoes, once it has been deployed in the server.

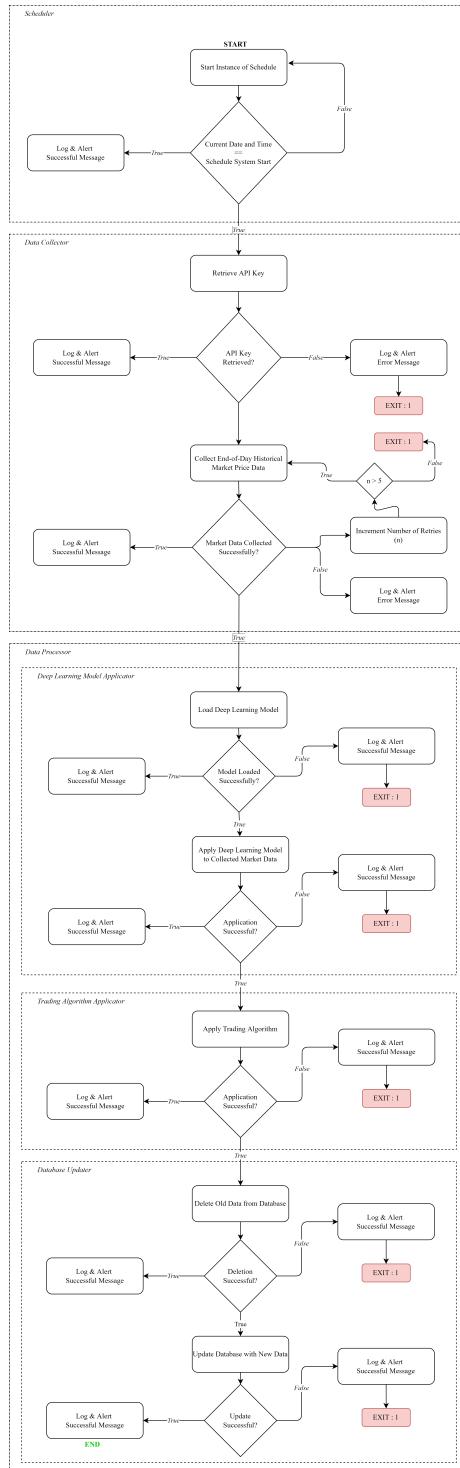


Figure 3.2: Full Overview of the Process Flow Diagram for the alamSYS

To better view and understand the flow, the processes were divided and presented in the succeeding parts.

Scheduler

Using Python's Schedule Library, an instance of a scheduled task is initialized upon the startup of the alamSYS. The scheduled task shall execute every Mondays to Fridays at exactly 6 P.M. Where the whole scheduling process is illustrated in Figure 3.3.

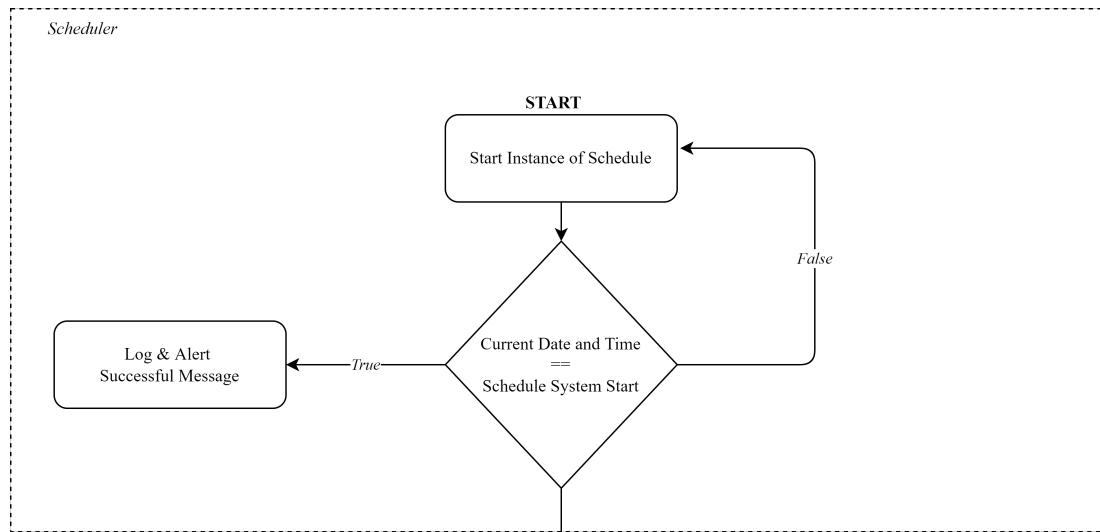


Figure 3.3: Overview of the Process Flow Diagram for the Scheduler

Data Collector

The collection of end-of-day (EOD) market data is the first process in the scheduled task. To collect EOD market data, the system first looks for the EODHD API key, which is stored in system variables or provided by the user in the tools directory. If the system is unable to locate an API key, it logs the error and notifies the user before exiting the program.

Once the API key is obtained, the system connects to the EOD market data provider and attempts to collect all market data five times. If it fails to collect data for the fifth time due to errors (i.e. incomplete payments, unstable network, and no established internet connection), the system logs the error, sends an alert to the user, and exits the program.

All successful processes are also logged and sent to the command line interface to notify the user. Figure 3.4 depicts this, as well as the entire data collection process. The flow of processes discussed above can be observed in Figure 3.4.

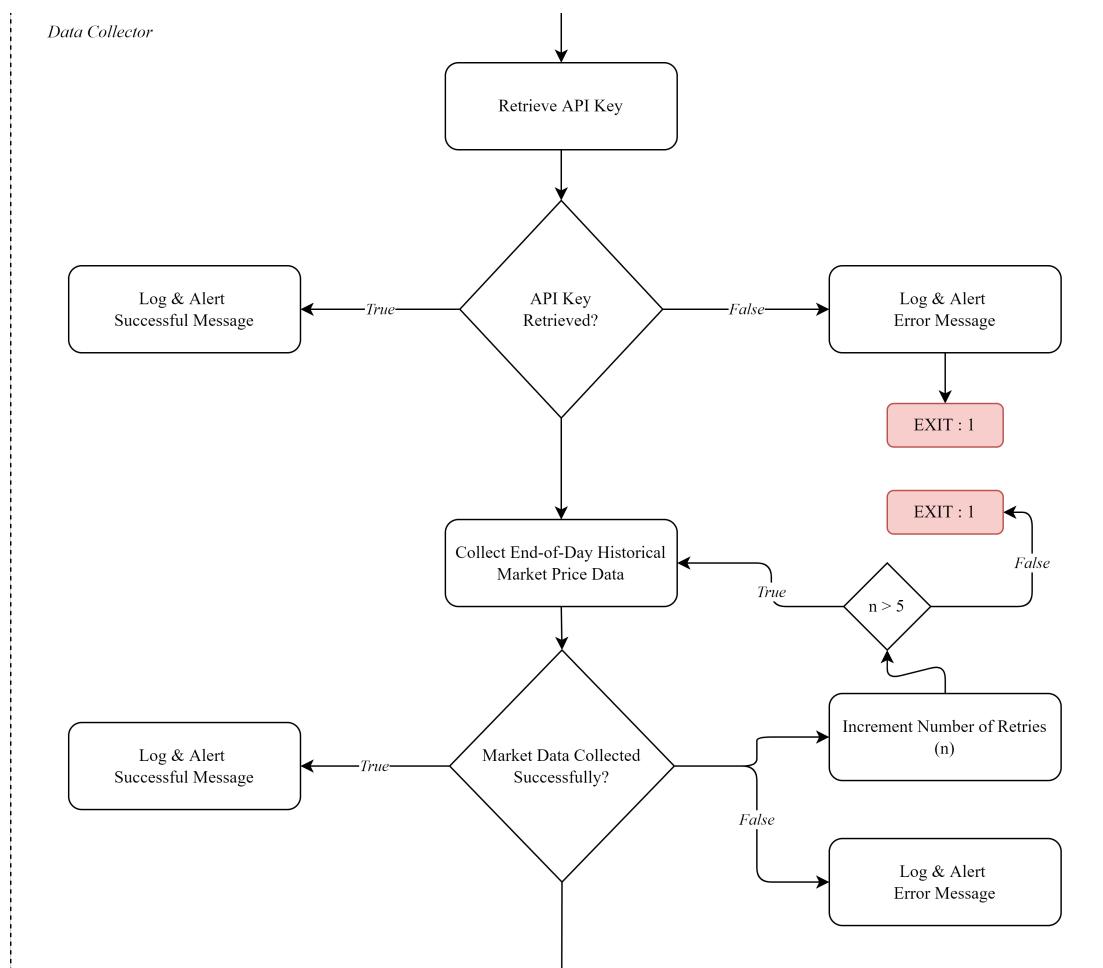


Figure 3.4: Overview of the Process Flow Diagram for the Data Collector

Data Processor

Data Processor is a process that is divided into three subprocesses, as shown in Figure 3.5.

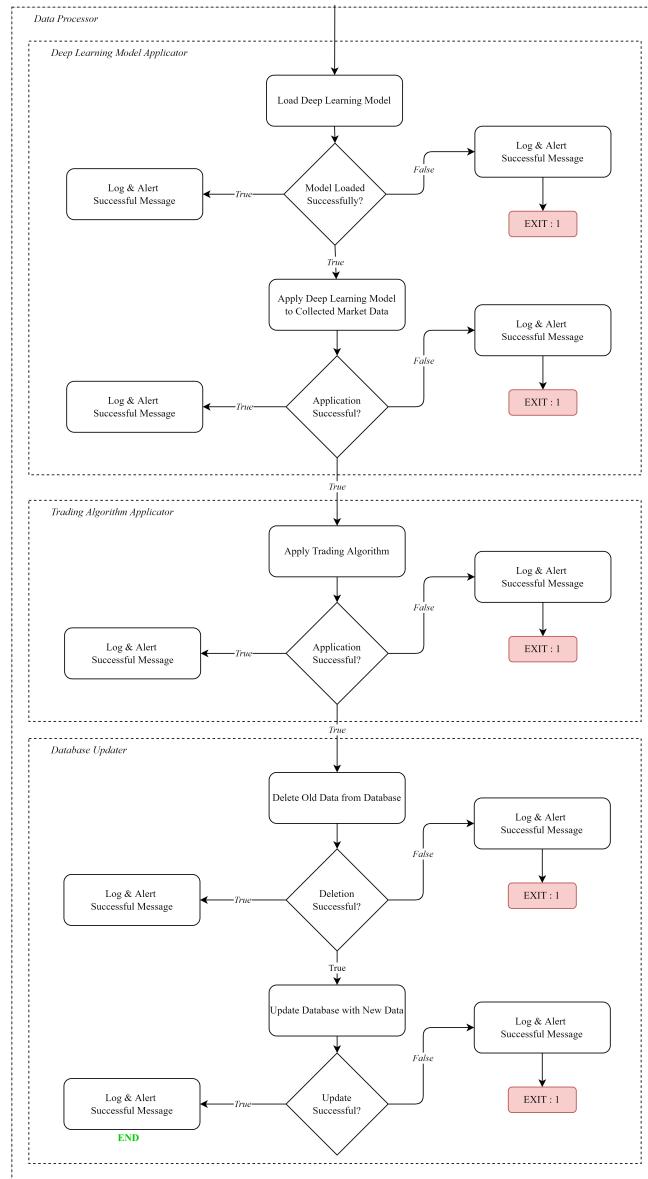


Figure 3.5: Overview of the Process Flow Diagram for Data Processor

Where the three subprocesses are as follows:

- (a) Deep Learning Model Applicator - This subprocess applies the deep learning model to the collected EOD market data for each stock. Specifically, alamSYS applies the DMD-LSTM model developed as part of this special problem. This is done as illustrated in Figure 3.6.

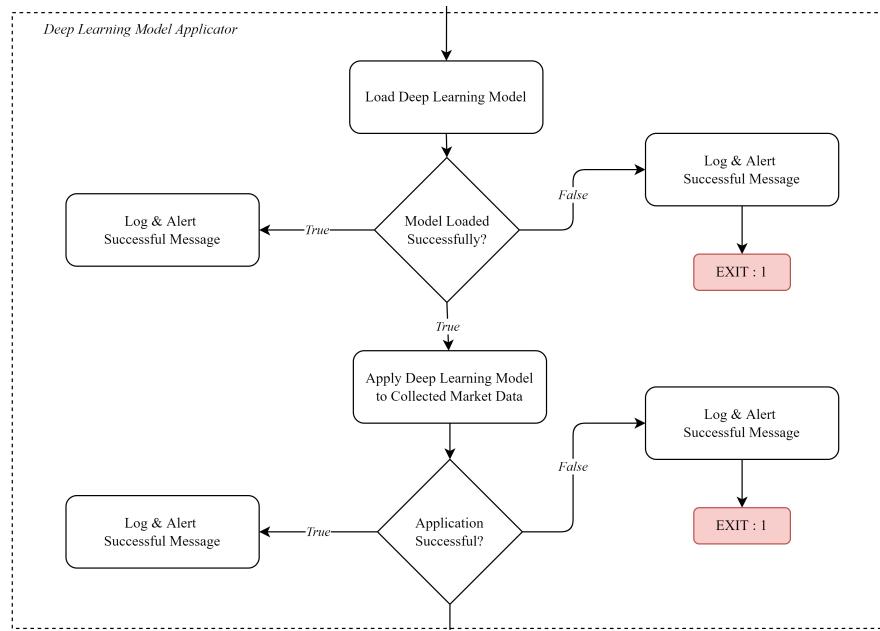


Figure 3.6: Overview of the Process Flow Diagram for the Deep Learning Model Applicator

- (b) Trading Algorithm Applicator - This subprocess applies the trading algorithm to the output data from the deep learning model applicator. Specifically, alamSYS applies ALMACD algorithm developed as part of this special problem. This is done as illustrated in Figure 3.7.

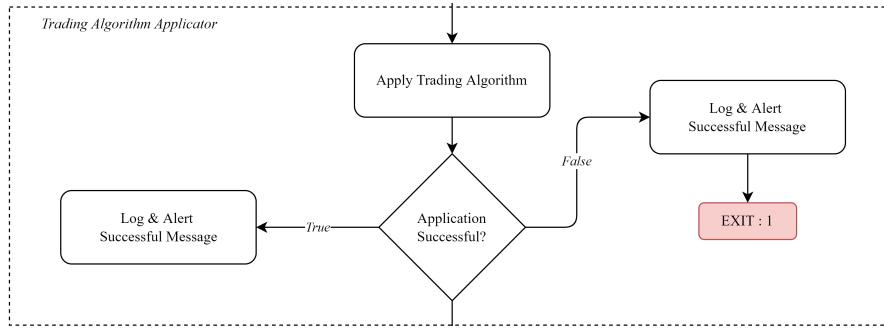


Figure 3.7: Overview of the Process Flow Diagram for the Trading Algorithm Applicator

- (c) Database Updater - This subprocess updates the database with the output data from the trading algorithm applicator. This is done as illustrated in Figure 3.8.

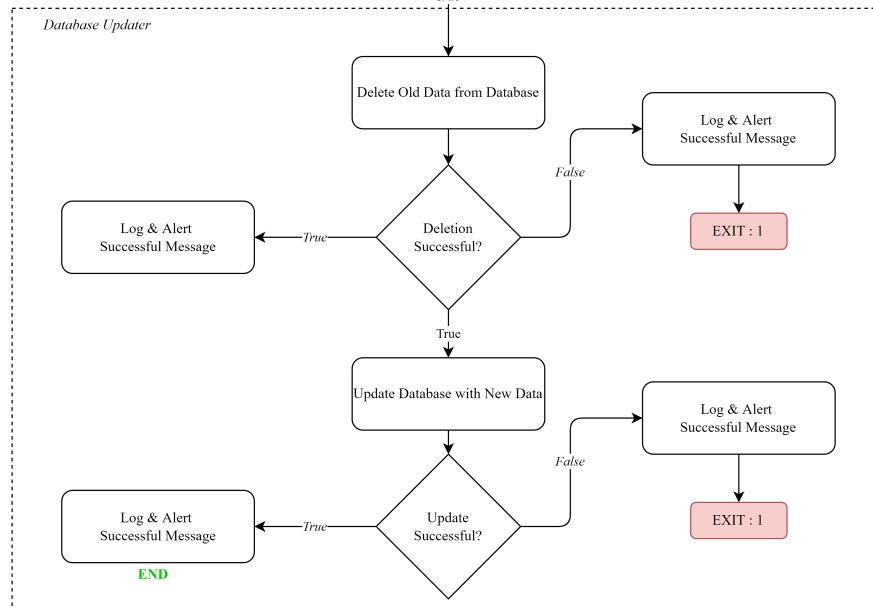


Figure 3.8: Overview of the Process Flow Diagram for the Database Updater

3.2.3 Data-Flow Diagram (DFD)

A data-flow diagram (DFD) helps to understand how processes work and how data flows from one process to the next. This is especially important because it provides an overview of the data's security by demonstrating how it can be accessed. In the case of alamSYS, the only publicly accessible data is the listed stock to buy and sell, as well as other functions as provided in its database and as permitted by the API endpoints.

Furthermore, the DFD paradigm used in the diagrams in this section adheres to the Gane-Sarson DFD symbols, which employ four basic symbols: (1) Entity / External Entity; (2) Data Flow; (3) Process; and (4) Data Store (VisualParadigm, n.d.)

Context Diagram

The overview of the entire process is depicted in a context diagram of the system, labeled process 0, as shown in Figure 3.9.

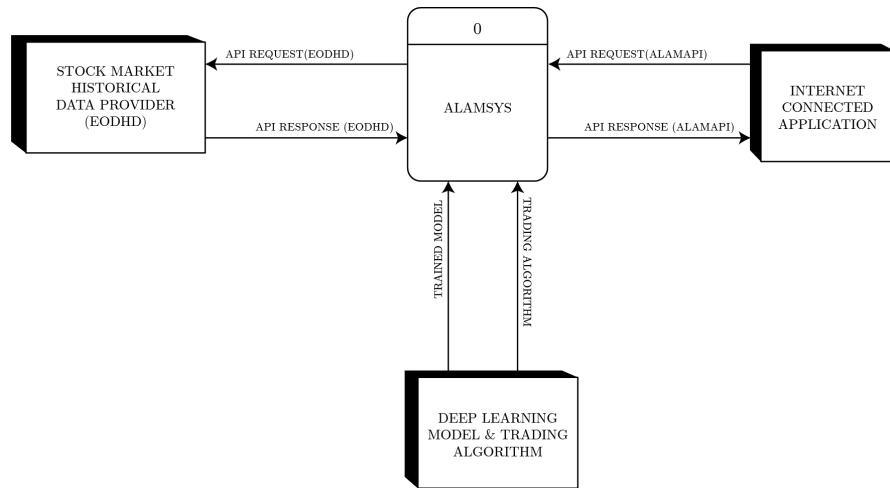


Figure 3.9: Context Diagram of the alamSYS

The diagram above depicts the root process (0), which is the alamSYS it-

self, and is linked to three external entities: (a) Stock Market Historical Data Provider, which was provided by EODHD; (b) Deep Learning Model & Trading Algorithm, these were developed alongside the alamSYS, specifically the DMD-LSTM Model and ALMACD, respectively; and (c) Internet Connected Application, these are any type of applications with internet access. Other applications may include a web-based application, a smart home speaker, and so on.

The data flow lines shown Figure 3.9 include the following: (a) API Request(EODHD), this is the request sent to the EODHD API to collect the stock market historical data; (b) API Response(EODHD), this is the response received from the EODHD API, which contains the end of day stock market data; (c) API Request(alamAPI), this are the requests sent by any internet connected application to the alamSYS via the alamAPI; (d) API Response(alamAPI), this are the responses sent by the alamSYS to any internet connected application via the alamAPI; (e) Trained Model, this is the trained model, referring to the DMD-LSTM, that was used to predict the price movement of the stocks in the Philippine Stock Market; and (f) Trading Algorithm, this is the deployed trading algorithm, referring to the ALMACD, that was used to determine the entry and exit signals of the stocks in the Philippine Stock Market.

In connection, the following are the API requests that can be utilized: (a) Home, this API endpoint returns a greeting message. Which should notify the user that they have connected to the alamAPI successfully; (b) Stocks to Buy, these API endpoints return a json data of recommended stocks to buy based on the current market price, the predicted price uptrend, and the entry signal of the trading algorithm in use; (c) Stocks to Sell, these API endpoints return a json data of recommended stocks to sell based on the current market price, the predicted price downtrend, and the exit signal of the trading algorithm in use; (d) ML Model Info, these API endpoints return a json data of the Machine Learning Models used in the alamSYS, as well as their associated information; and (e) Stock Risks Profile, these API endpoints return a json data of stocks in the alamSYS as well as their risk values based on value at risk (%), volatility (%), and drawdown (%).

DFD of Diagram 0

To better understand how each data stream entering and exiting the root process is processed, it is essential to look inside the inner workings of the root process, which is illustrated in the DFD of Diagram 0, as shown in Figure 3.10.

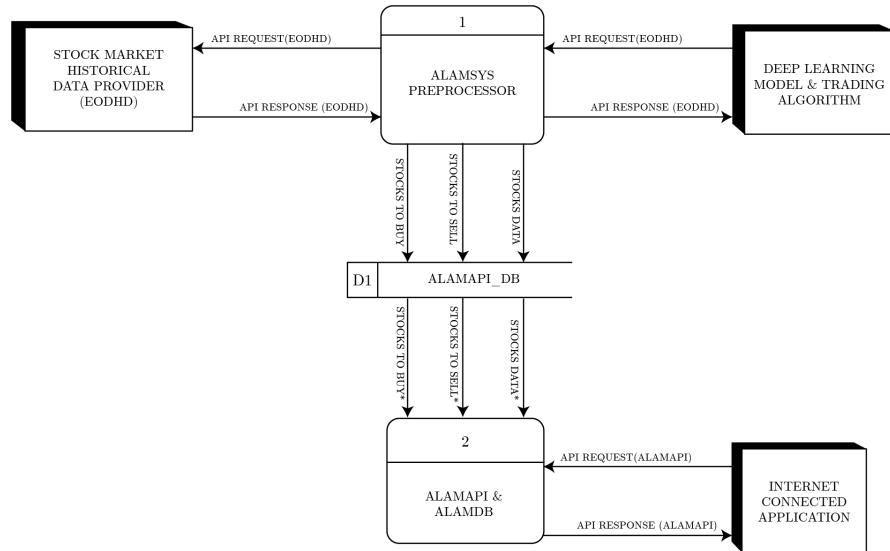


Figure 3.10: DFD of Diagram 0

From the figure above, the root process, has two main processes: (a) alam-SYS Preprocessor, which is the system's stock market data processing unit, which deploys the deep learning model (DMD-LSTM), and the trading algorithm (ALAMACD) to predict the price movement of the stocks in the Philippine Stock Market; and (b) alamAPI & alamDB, which is the system's API and database unit, which is responsible for processing the API requests and responses, as well as storing the data of the system, respectively.

DFD of Diagram 1

To better understand the internal workings of the Process 1, it is useful to check the DFD of that process, which is illustrated in Figure 3.11

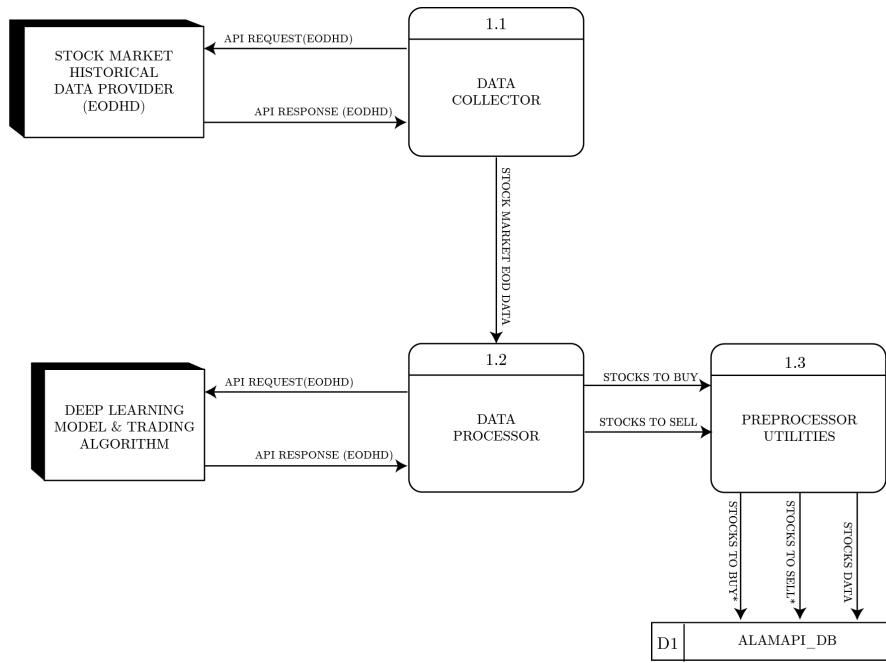


Figure 3.11: DFD of Diagram 1

From the figure shown above, it can be observed that Process 1 is composed of three internal processes, which are as follows: (a) Data Collector, which is the main process responsible for collecting the historical market data using EODHD End of Day Market Data API; (b) Data Processor, which is the main process responsible for processing the collected stock market data using the DMD-LSTM Model and the ALMACD Trading Algorithm; and (c) Preprocessor Utilities, these are a set of tools that contains the utilities used by the data processor, such as the initialization of the database, database related actions, database models, stock symbols, and logs and alerts module.

DFD of Diagram 1.2

This shows the processes inside the process 1.2, which is the data processor.

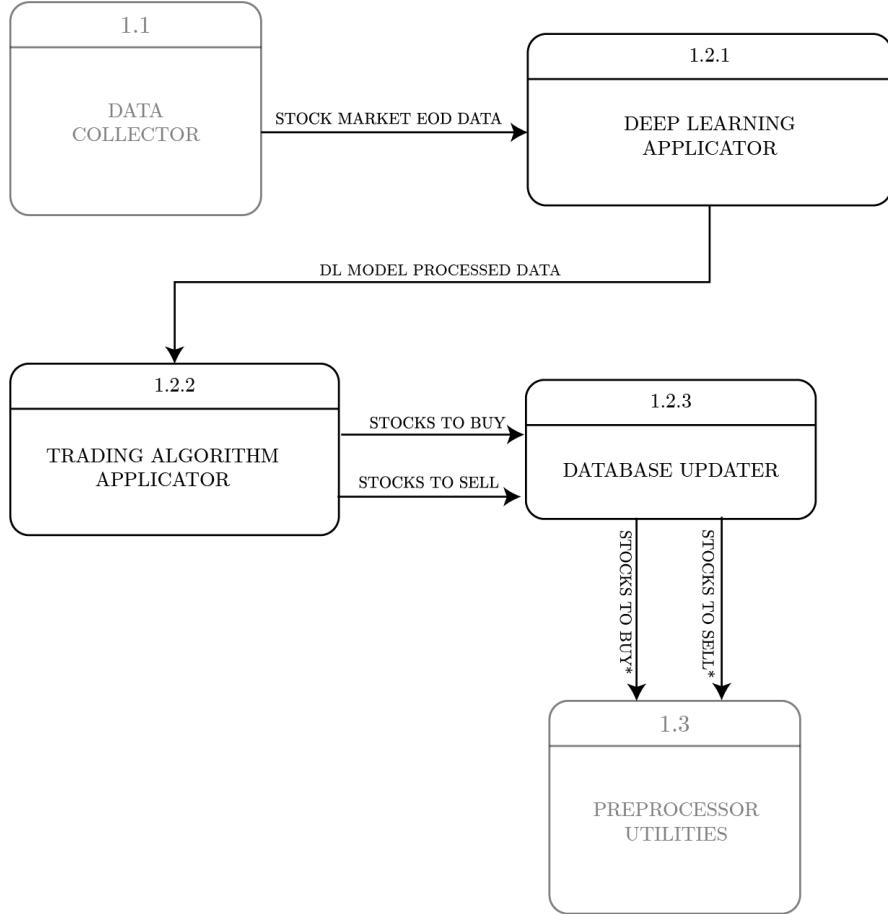


Figure 3.12: DFD of Diagram 1.2

The data processor is further composed of three processes, which are as follows:

- Deep Learning Applicator, this subprocess applies the DMD-LSTM model to the collected stock market end-of-day (eod) data;
- Trading Algorithm Applicator, the eod data alongside the list of predicted stock prices composes the "DL Model Processed Data", which is sent to this subprocess. This subprocess applies the ALMACD to better determine the entry (buy or hold), and exit (sell) signals for each stocks;
- Database Updater, a subprocess responsible for updating the contents of the database, based on the data processed by the Deep Learning Applicator and Trading Algorithm Applicator.

DFD of Diagram 2

Figure 3.13 shows the inner processes of the Process 2 (alamAPI & alamDB).

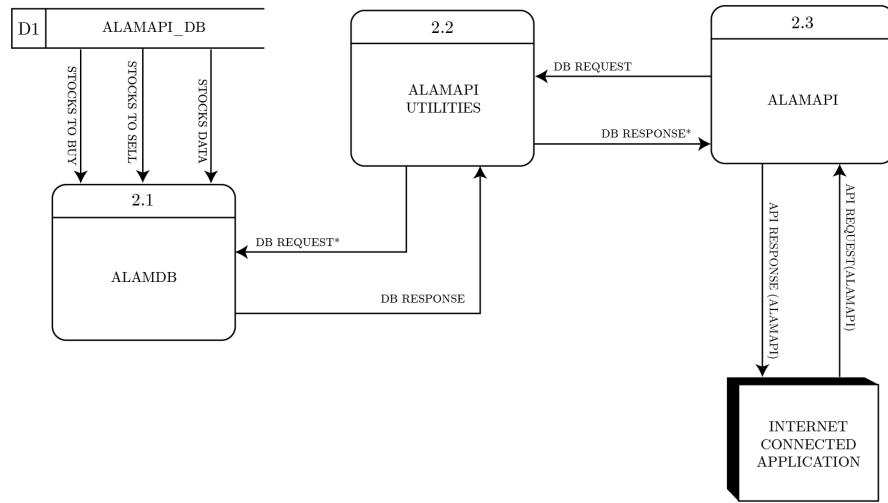


Figure 3.13: DFD 2: Data-Flow Diagram for the alamSYS

The figure above shows three internal processes of the Process 2, namely: (a) alamDB, this process, processes the database request sent by the alamAPI as requested by the Internet Connected Application through the utilization of alamAPI utilities. It also sends the database response to the alamAPI via the same utilities module; (b) alamAPI Utilities, this process serves as a mediator of database request and responses between the alamDB and alamAPI; and (c) alamAPI, this process contains all the API endpoints for the alamAPI, which is responsible for processing the requests and API responses from and to the connected users.

3.2.4 Object Document Mapper (ODM) Diagram

Because the system's database is non-relational, an Object Document Mapper (ODM) diagram rather than an Entity Relationship Diagram (ERD) is shown in this section. The ODM diagram is shown in Figure 3.14:

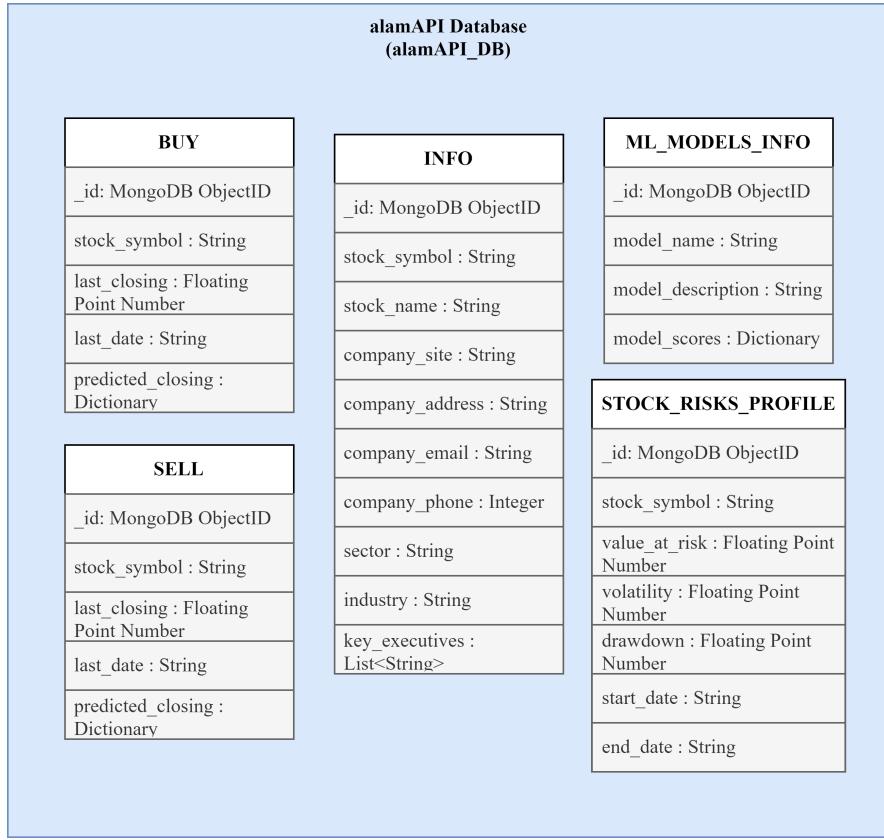


Figure 3.14: Object Document Mapper for the alamSYS Database

As shown from the Figure 3.14, the "alamAPI_DB" is the database name of the system. Where it is composed of five collections namely:

- Buy – this collection contains all the stocks that the data processor predicted and classified as a stock to buy. A sample of which is presented in Figure 3.15.

```

_id: ObjectId("64368d0413193f92da07ba45")
stock_symbol: "MEG"
last_closing: 2.02
last_date: "2023-04-12"
predicted_closing: Object

_id: ObjectId("64368d0413193f92da07ba46")
stock_symbol: "BDO"
last_closing: 138.2
last_date: "2023-04-12"
predicted_closing: Object

_id: ObjectId("64368d0413193f92da07ba47")
stock_symbol: "GLO"
last_closing: 1798
predicted_closing: Object

```

Figure 3.15: Sample Buy Collection from the alamSYS Database

- (b) Sell – this collection contains all the stocks that the data processor predicted and classified as a stock to sell. a sample of which is presented in Figure 3.16.

```

_id: ObjectId("64368d0413193f92da07ba50")
stock_symbol: "JGS"
last_closing: 49.35
last_date: "2023-04-12"
predicted_closing: Object

_id: ObjectId("64368d0413193f92da07ba51")
stock_symbol: "FGEN"
last_closing: 16.54
last_date: "2023-04-12"
predicted_closing: Object

_id: ObjectId("64368d0413193f92da07ba52")
stock_symbol: "ICT"
last_closing: 211
predicted_closing: Object

```

Figure 3.16: Sample Sell Collection from the alamSYS Database

- (c) Info – this collection contains the general and relevant information about a stock, or the general company information. Such as the stock symbol, stock name, company site, company address, company email, company phone number, sector, industry, and the company's key executives. Where all of this information are gathered from their official listing accessed in the database of the Philippine Stock Exchange (PSE). A sample of which is

presented in Figure 3.17.

The screenshot shows the MongoDB interface for the `alamAPI_DB.info` collection. At the top right, it displays "20 DOCUMENTS" and "1 INDEXES". Below the header, there are tabs for "Documents", "Aggregations", "Schema", "Explain Plan", "Indexes", and "Validation". A search bar at the top says "Type a query: { field: 'value' }". Below the search bar are buttons for "Reset", "Find", and "More Options". A "Filter" dropdown is also present. A "ADD DATA" button and an "EXPORT COLLECTION" button are located below the search bar. The main area shows two documents listed:

```

_id: ObjectId("642ad1c3823687269fffb43a9")
stock_symbol: "MEG"
stock_name: "Megaworld Corporation"
company_site: "https://www.megaworldcorp.com"
company_address: "30th Floor, Alliance Global Tower, 36th Street cor. 11th Avenue, Uptown"
company_email: "investorrelations@megaworldcorp.com"
company_phone: 63288886342
sector: "Property"
Industry: "Real Estate Development"
key_executives: Array

_id: ObjectId("642ad1c3823687269fffb43aa")
stock_symbol: "JGS"
stock_name: "JG Summit Holdings, Inc."
company_site: "https://www.jgsummit.com.ph"
company_address: "43/F Robinsons Equitable Tower, ADB Avenue corner Poveda St., Ortigas"
company_email: "TBO@jgsummit.com.ph"

```

Figure 3.17: Sample Info Collection from the alamSYS Database

- (d) Machine Learning (ML) Models Info – this collection contains the details about the Machine Learning Model/s deployed in the system. For the current alamSYS, only one model is deployed, which is the DMD-LSTM model. A sample of which is presented in Figure 3.18.

The screenshot shows the MongoDB interface for the `alamAPI_DB.ml_models_info` collection. At the top right, it displays "1 DOCUMENTS" and "1 INDEXES". Below the header, there are tabs for "Documents", "Aggregations", "Schema", "Explain Plan", "Indexes", and "Validation". A search bar at the top says "Type a query: { field: 'value' }". Below the search bar are buttons for "Reset", "Find", and "More Options". A "Filter" dropdown is also present. A "ADD DATA" button and an "EXPORT COLLECTION" button are located below the search bar. The main area shows one document listed:

```

_id: ObjectId("642ad1c3823687269fffb43bd")
model_name: "DMD-LSTM"
model_description: "Implemented dynamic modes from Dynamic Mode Decomposition (DMD) to the..."
model_scores: Object
  mae: "0.00328"
  nape: "0.00191"
  mse: "0.00002"
  rmse: "0.00419"

```

Figure 3.18: Sample ML Models Info Collection from the alamSYS Database

- (e) Stock Risks Profile - this collection contains the details about the risk profiles for each stock. A sample of which is presented in Figure 3.19.

```

_id: ObjectId('642ad1c3823687269fffb43bf')
stock_symbol: "MEG"
value_at_risk: -5.365715132
volatility: 3.9496962973
drawdown: 57.2481396806
start_date: "2000-01-03"
end_date: "2023-02-10"

_id: ObjectId('642ad1c3823687269fffb43bf')
stock_symbol: "JGS"
value_at_risk: -4.7623573876
volatility: 3.3610904816
drawdown: 43.1840442168
start_date: "2000-01-03"
end_date: "2023-02-10"

```

Figure 3.19: Sample Stock Risks Profile Collection from the alamSYS Database

Note that each collection are their own separate entities, hence the database is called non-relational, as the documents are not in any way related to each other.

3.2.5 Deep Learning Model Diagram

In this section, the process on how the deep learning model was developed is shown in Figure 3.20. Wherein, the process overview is based on the Fine-Tuned Support Vector Regression Model for Stock Predictions by Dash and Dash (2016).

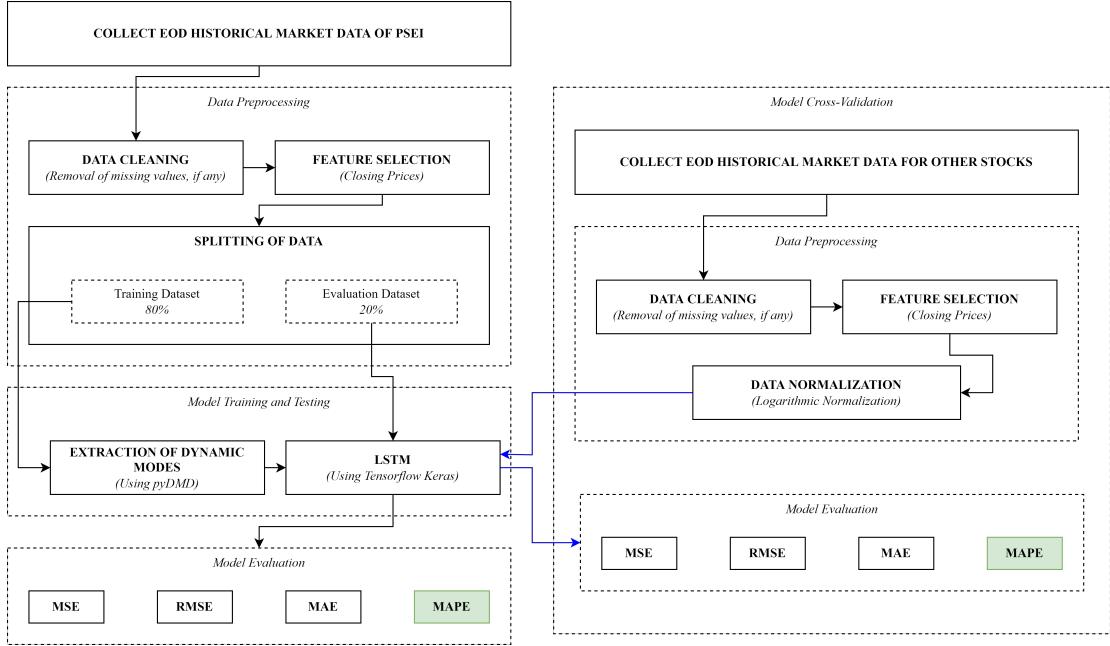


Figure 3.20: DMD-LSTM Model Development Methodology for alamSYS

Data Collection

The market data used to develop the DMD-LSTM model was obtained through EODHD's end-of-day market data API. While PSEI market data was used for model training and testing, market data from other stocks was used for cross-validation of the DMD-LSTM model. The following are the specifics of the stocks gathered:

Table 3.1: Collected Market Data Details

Stock	Data Count	Start Date	End Date
AC	6809	June 27, 1994	February 10, 2023
ALI	6789	June 27. 1994	February 10, 2023
AP	3795	July 16, 2007	February 10, 2023
BDO	5041	May 22, 2002	February 10, 2023
BLOOM	3033	October 30, 2000	February 10, 2023
FGEN	4142	February 02, 2006	February 10, 2023
GLO	6707	January 03, 1995	February 10, 2023

Table 3.1 continued from previous page

Stock	Data Count	Start Date	End Date
ICT	6805	January 03, 1995	February 10, 2023
JGS	6525	June 27, 1994	February 10, 2023
LTG	3774	February 06, 1995	February 10, 2023
MEG	6751	January 03, 1995	February 10, 2023
MER	6799	June 27, 1994	February 10, 2023
MPI	3888	December 18, 2006	February 10, 2023
PGOLD	2762	October 06, 2011	February 10, 2023
PSEI	5675	January 03, 2000	February 10, 2023
RLC	5879	June 27, 1994	February 10, 2023
RRHI	2253	November 11, 2013	February 10, 2023
SMC	6799	June 27. 1994	February 10, 2023
TEL	6814	June 27, 1994	February 10, 2023
URC	6135	June 03, 1995	February 10, 2023

Data Preprocessing

Data preprocessing before model training and testing is composed of three main processes which are as follows: (a) Data Cleaning, this was done to clean any missing values from the data; (b) Feature Selection, closing prices was selected as the main feature of the model; and (c) Splitting of Data, data was split in the ratio of 80:20 for testing and training data, respectively.

Meanwhile for the data preprocessing for cross-validation, the following processes were done: (a) Data Cleaning, this was done to clean any missing values from the data; (b) Feature Selection, closing prices was selected as the main feature of the model; and (c) Data Normalization, using logarithmic normalization method, the data was normalized Logarithmic normalization was utilized to help solved the problem with the data having extreme ranges, which affects the evaluation of the cross-validation. In essence it was used to enable data stability, and increase interpretability (Baeldung, 2022; Tuychiev, 2021; Andrew, 2019).

Model Training and Testing

Using pyDMD the dynamic modes was extracted from the training and testing data, these extracted values alongside the actual closing price data were utilized for the training of an LSTM model using Tensorflow Keras Library.

There are a total of eight model variations trained and tested, which are as follows: (1) Baseline LSTM with window size of 5; (2) Baseline LSTM with window size of 10; (3) Baseline LSTM with window size of 15; (4) Baseline LSTM with window size of 20; (5) DMD-LSTM with window size of 5; (6) DMD-LSTM with window size of 10; (7) DMD-LSTM with window size of 15; and (8) DMD-LSTM with window size of 20. Where the best performing model was used for the cross-validation and was deployed to the system.

Model Evaluation

The DMD-LSTM Model was evaluated using the following error metrics:

- (a) Mean Squared Error (MSE) - a well-known metric for assessing regression models. It calculates the average of the squared differences between predicted and true values. MSE is useful because it penalizes large errors more severely than small errors, which is important in some applications. A lower MSE indicates that the model performed better (Glen, n.d.-a).
- (b) Root Mean Squared Error (RMSE) - another popular metric for evaluating regression models is the RMSE, which is the square root of the MSE. It, like MSE, computes the average of the differences between predicted and true values. Because it is in the same unit as the target variable, RMSE is easier to interpret. A lower RMSE indicates that the model is performing better (Glen, n.d.-b).
- (c) Mean Absolute Error (MAE) - it calculates the average of the absolute differences between predicted and true values. MAE is advantageous because it is more resistant to outliers than MSE and RMSE. A lower MAE indicates that the model is performing better (Secret Data Scientist, 2023).

- (d) Mean Absolute Percentage Error (MAPE) - it computes the average percentage difference between predicted and true values. MAPE is useful because it provides a relative measure of error, which makes comparing model performance across different target variable scales easier. A lower MAPE indicates that the model is performing better (Allwright, 2022). Furthermore, this is the primary error metric used to select the final model deployed to alamSYS.

Model Cross-Validation

The model selected for the cross-validation was the DMD-LSTM model with a window size of 5, due to it being the highest performing model based on having the lowest MAPE scores compared to the other models, this is further discussed on Chapter 4 of this paper.

Model cross-validation was conducted using the stock market data from the other stocks aside from the training data from PSEI. Where cross-validation of an LSTM model must be done before deployment to assess the model's generalization performance. LSTM models are well-known for their ability to capture long-term dependencies in sequential data, but their performance varies greatly depending on the dataset and model hyperparameters used. And a correctly performed cross-validation helps to provide a more accurate estimate of the model's performance on unseen data, which is critical for ensuring that the model performs well in real-world scenarios (Mellema, 2020; Scherzinger, Roennau, & Dillmann, 2019).

Model Deployment

After determining that the DMD-LSTM model works well with non-training stock market data, it was deployed to the alamSYS as a '.keras' file.

3.2.6 ALMACD Development Diagram

This section discusses the development and integration of the Arnaud Legox Moving Average Convergence and Divergence (ALMACD) indicator as a trading algorithm. The ALMACD indicator tells traders when to buy and sell securities based on the convergence and divergence of two ALMA (fast and slow). Specifically, when the fast ALMA crosses above the slow ALMA, the indicator suggests buying the stock. Conversely, when the fast ALMA crosses below the slow ALMA, the indicator suggests selling the stock.

The process of the development and integration of the ALMACD is shown in Figure 3.21.

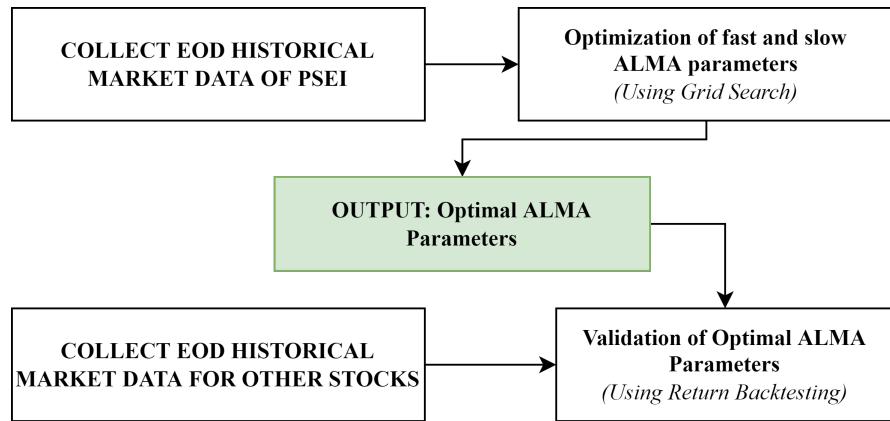


Figure 3.21: ALMACD Development Methodology for alamSYS

Optimization of ALMA Parameters using Grid Search Approach

The ALMACD indicator utilizes two ALMA indicators, one with a fast period and one with a slow period. Where each of them have their own set of optimal parameters, such as their window size, offset, and sigma. In order to find this optimal parameters, a grid search approach was used.

Grid search is a machine learning technique used to fine-tune the hyperparameters of a model. It is a brute force approach that tries every combination of

hyperparameters and evaluates the model performance for each combination. The combination with the best performance is then selected as the optimal hyperparameters (Joseph, 2018). However in the case of the optimization of ALMACD, the grid search approach was utilized to find the optimal parameters for the fast and slow alma, such that it yields a favorable performance by computing the expected return of the trading algorithm. Shown in Table 3.2 are the parameters used for the grid search:

Table 3.2: ALMA Grid Search Parameters

	WINDOW SIZE	SIGMA	OFFSET
FAST ALMA	1 to 10		
SLOW ALMA	10 to 20	1 to 20	0.85, 0.90, and 0.95

Validation of ALMACD Parameters Using Return Backtesting Approach

Before integrating the optimal ALMACD parameters to the alamSYS, it was first validated using the data from other stocks through backtesting, and calculating the potential returns. Once the returns on the validation stocks were computed to be positive, the ALMACD indicator was then integrated to the alamSYS.

Integration of ALMACD to the alamSYS

The ALMACD indicator was integrated to the alamSYS as a trading algorithm. It was implemented to run after the deep learning model to better identify which stocks to buy and sell.

3.2.7 Docker-Compose Layer Diagram

This section illustrates the different layers of the docker-compose containers based on the way it was used in the deployment of the system.

Figure 3.22 shows a diagram based on Docker documentation, to better understand the containers and layers of the alamSYS. *Note that in the diagram the lowest level is the "Server Infrastructure" and the highest level is the "Docker-Compose" layer.*

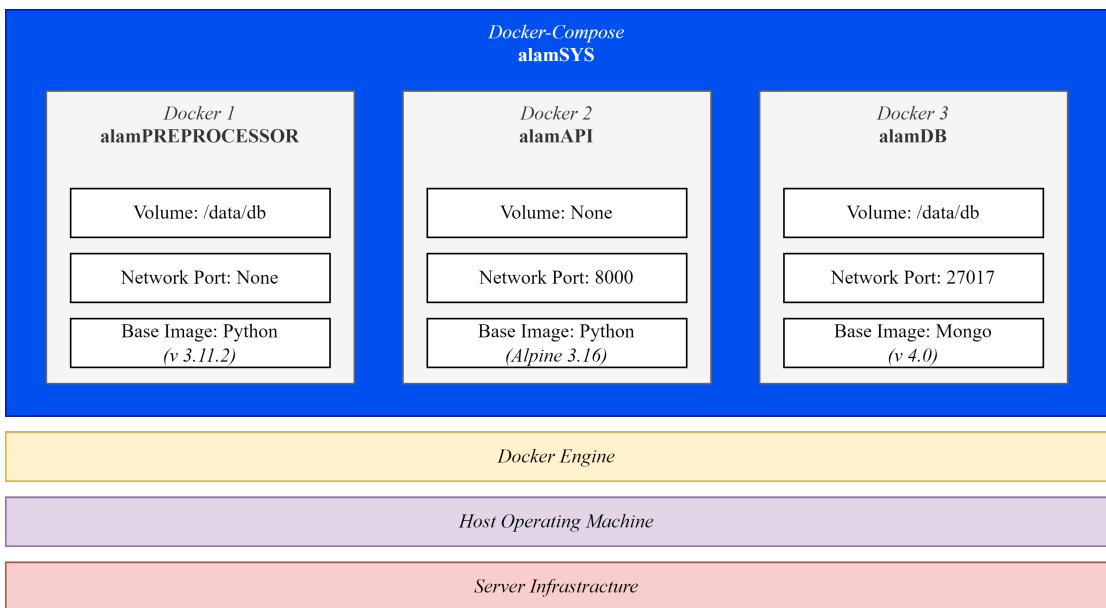


Figure 3.22: Docker-Compose Layer Diagram for the alamSYS

From the figure above, it can be observed that the docker-compose layer is composed of three docker containers, which are as follows: (a) Docker 1 : alamPREPROCESSOR, this is the docker container that contains the alamPREPROCESSOR application. It uses Python 3.11.2 as its based image. Moreover, this container is connected to the data/db volume, and there is no network port exposed; (b) Docker 2 : alamAPI, this is the docker container that contains the alamAPI application. It uses a Python image as well, but is running on top of an Alpine 3.16 image. This was done as Alpine images runs on minimal resources than other Linux based images. Furthermore, it is not attached to any volume, and its network port 8000 is exposed; and (c) Docker 3 : alamDB, this is the docker container that contains the alamDB application which runs on top of the Mongo version 4.0 image. Similarly to alamPREPROCESSOR, it is also connected to the data/db volume, while its network port 27017 is exposed.

3.3 Hardware Specifications

The development of the alamSYS and its components, as well as for the development and testing of the deep learning model and alamAPP, a laptop device with the following hardware specifications was utilized: (a)CPU, Intel(R) Core(TM) i5-8300H CPU @ 2.30GHz to 3.10 GHz; *Note that only the CPU was utilized for the development, however the device has a dedicated GPU: NVIDIA GeForce GTX 1050;* and (b)Memory, 16GB DDR4 @2666Mhz.

In order to test the deployment capacity of the alamSYS, other computers were used to connect to the server. In this case the aforementioned laptop device was used as a deployment server. *Note that a total of 10 desktop devices were utilized.* And the specifications of the devices are: (a) CPU, Intel Core Pentium; and (b) Memory, 4GB DDR4 RAM.

Lastly, the mobile device used to showcase the main features of the alamSYS has the following specifications: (a) CPU, 8-core 3.2GHz (Snapdragon 870 5G); and (b) Memory, 8GB RAM.

3.4 Methodology

3.4.1 Software Development Process

Because of the expected tight time constraints during system development, the author of this paper chose an Agile Software Development Methodology. Agile Sprints were primarily used to manage time efficiently during the software development process (JavaTPoint, n.d.; Milne, 2021). Moreover, a modified Agile Development Methodology as shown in Figure 3.23 was followed in the system development procedures for this special problem.

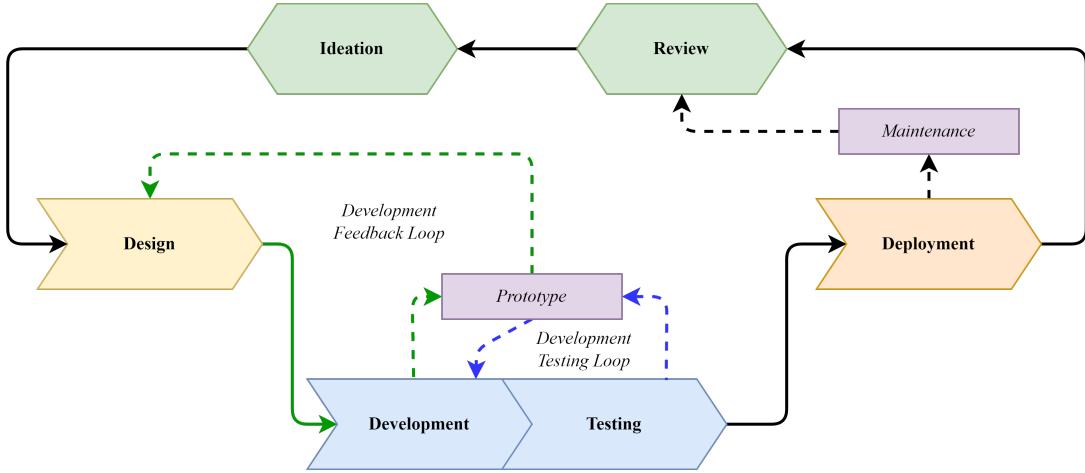


Figure 3.23: Modified Agile Development Methodology

The modified agile development methodology shown above includes additional and specific "feedback loops" such as a development feedback loop that focuses on the continuous design and development process; and a development testing loop that ensures that the software being developed works as intended or better than expected.

The Table 3.3 shows the list of sprints and sub-activities that were followed:

Table 3.3: Summary of Sprints and Activities

Sprint Number	Activities	Completion Date
1	<p>Main Activity: Requirement Analysis, System Planning, and Evaluation</p> <p>Sub-Activities:</p> <ul style="list-style-type: none"> • Topic Proposal • Drafted Chapters 1 to 3 for the Special Problem Proposal • System Architecture and User Requirement Analysis 	September 15, 2022 to December 9, 2022

Table 3.3 continued from previous page

Sprint Number	Activities	Completion Date
2	<p>Main Activity: Development of Initial System Prototype</p> <p>Sub-Activities:</p> <ul style="list-style-type: none"> • Build the different component of the alamSYS as indicated in the top-level overview diagram of the system, the following prototype were initially developed: <ul style="list-style-type: none"> [1.] API endpoints [2.] Database [3.] Preprocessor • Tested the build prototype. 	<p>12 Weeks September 30, 2022 to December 23, 2023</p>

Table 3.3 continued from previous page

Sprint Number	Activities	Completion Date
3	<p>Main Activity: Development and Testing of DMD-LSTM Model</p> <p>Sub-Activities:</p> <ul style="list-style-type: none"> • Stock Market Data until February 10, 2023 was collected. • Development of the Deep Learning Model, following the procedures as previously discussed on section 3.2.5. • Development of Baseline LSTM model for comparison against DMD-LSTM Model. • Deep Learning model training, testing, cross-validation, and evaluation. • Revised of Chapters 1 to 3. 	January 1, 2023 to February 21, 2023

Table 3.3 continued from previous page

Sprint Number	Activities	Completion Date
4	<p>Main Activity: Integration of DMD-LSTM Model to the alamSYS and System Testing</p> <p>Sub-Activities:</p> <ul style="list-style-type: none"> • Integrated DMD-LSTM to alamSYS. • Initial tests of the system. • Created System Tests and Loggers. • System Testing, Results Logging and Summarization. 	January 15, 2023 to March 7, 2023

Table 3.3 continued from previous page

Sprint Number	Activities	Completion Date
5	<p>Main Activity: System Testing, Analysis, Refactoring, and Deployment</p> <p>Sub-Activities:</p> <ul style="list-style-type: none"> • Development test loop was conducted. • System Refactoring. • System Update and inclusion of additional features. • Results Gathering and Summarization. • Started the development of the test application (for showcasing of the system features). 	March 7, 2023 to April 13, 2023

Table 3.3 continued from previous page

Sprint Number	Activities	Completion Date
6	<p>Main Activity: Finalization of Paper, System Defense, and Presentation</p> <p>Sub-Activities:</p> <ul style="list-style-type: none"> • Revised Chapter 1 to 3. • Drafted Chapter 4 to 5. • Finalization of the mobile-based test application. • Reviewed the Content of this paper. • Revised and Finalized the special problem paper. • Created presentation slide deck for the presentation of the special problem. 	March 14, 2023 to May 12, 2023

From Table 3.3, it was shown that it took a total of 34 weeks (from September 15, 2022, to May 12, 2023) to develop the alamSYS and all its components, as well as to write the contents of this paper. This was 5 weeks better than the expected total weeks allotment for the completion of this whole project, as projected in the Gantt Charts shown in Appendix B.

3.4.2 Procedures

This section provides a general overview of the steps conducted in the development of the different components developed in the whole duration of this special problem.

Procedures for the Development of the alamSYS and its Components

The following step-by-step procedures were conducted in the development of the alamSYS and its components (alamAPI, alamDB, and alamPREPROCESSOR):

- (a) User Requirement and Analysis
- (b) System Design and Architecture using DFD
- (c) System Prototyping and Development.
- (d) System Testing and Continuous Development and Integration. Each of the components of the alamSYS were developed into smaller chunks, and each chunks were tested and integrated with the other smaller chunks until a component was functionally developed. Afterwards the same process for the other components were conducted, until a whole working system was developed.
- (e) System Deployment, Review, and Maintenance
- (f) Steps (a) to (e) were repeated until the system was deemed to be stable and ready for deployment within the scopes of this special problem. Moreover, the system shall be maintained even after the submission of this special problem such as the addition of new features.

Procedures for the Development of the DMD-LSTM Model

The following step-by-step procedures were conducted in the development of the DMD-LSTM model:

- (a) Collection of end-of-day Philippine Stock Market Data from the 20 selected stocks from the Philippine Stock Exchange (PSE) for the periods until February 10, 2023.
- (b) Data Preprocessing, Model Training, Testing, Cross-validation, and Evaluation was conducted based on the DMD-LSTM methodology shown in Figure 3.20.
- (c) The best performing DMD-LSTM model was deployed and integrated as part of the alamSYS, specifically used in the alamPREPROCESSOR component.

Procedures for the Development of the ALMACD Trading Algorithm

The following step-by-step procedures were conducted in the development of the ALMACD trading algorithm:

- (a) Collection of end-of-day Philippine Stock Market Data from the 20 selected stocks from the Philippine Stock Exchange (PSE) for the periods until February 10, 2023.
- (b) Using a Grid Search approach, the best ALMA parameters (*i.e.* window size, sigma, and offset) for both slow and fast were determined.
- (c) Using the best ALMA parameters, a trading algorithm was developed which takes in the convergence and divergence of the slow and fast ALMA as basis for entry and exit signals. When the slow ALMA crosses above the fast ALMA, a buy signal is generated. When the slow ALMA crosses below the fast ALMA, a sell signal is generated.
- (d) The trading algorithm was deployed and integrated as part of the alamSYS, specifically used in the alamPREPROCESSOR component.

Procedures for the Development of Mobile-based Test Application

The following step-by-step procedures were conducted in the development of the mobile-based test application:

- (a) User Interface Design and Development using Figma.
- (b) Creation of Initial Prototype using Flutter.
- (c) Continuation of Feature Development and Testing.
- (d) Integration of HTTP methods with the alamAPI.

Note that the purpose of this application is to showcase the main feature of the alamSYS, as such it is not fit for user deployment. However it is further discussed in the recommendations on how it can be potentially deployed by future developers.

Procedures for the System Testing

The following step-by-step procedures were conducted in the system testing:

- (a) Development of tester application, which are as follows:
 1. System Statistics Logger - using the docker stats stream command, the system CPU, memory, and network utilization stats were recorded. This logger was used to determine the idle and on load performance of the alamSYS.
 2. Deployment Tester - this tester application was used to test the deployment reliability of the alamSYS. The test was conducted by deploying the alamSYS in a server while then other computers requests for different functions to the alamSYS over the internet using the HTTP methods. Further the tester also logs the overall response time to process 10, 100, and 1000 requests in a row, as well as the success rate of the requests.
 3. Internal System Stress Tester - this tester application was used to test the internal system reliability of the alamSYS. The test was conducted to run the processes of the alamSYS in a loop for 100 consecutive times, and logs the overall response time as well as success rate for each processes.
- (b) Initial testing of the tester applications, to see if they are working as intended

- (c) Actual testing was done using these tester applications, and all results and system logs were recorded.
- (d) The results of the testing were then analyzed, compared, and summarized. Further details regarding this are discussed on Chapter 4.

Chapter 4

Results and Discussions

This chapter presents results and discussions from this special problem. Its goal is to provide a comprehensive analysis and interpretation of the data collected for alamSYS's internal and external components. Hence to comply, this chapter is divided into the following sections:

- (a) Documentation for alamSYS
- (b) DMD-LSTM Results and Discussions
- (c) ALMACD Results and Discussions
- (d) alamSYS System Tests Results and Discussions
- (e) Results and Discussions for the Real World Application of alamSYS

4.1 alamSYS Documentation

The goal of this section is to thoroughly document the current state of the alamSYS in order to facilitate meaningful discussions.

4.1.1 Documentation for alamPREPROCESSOR

The documentation section for the alamPREPROCESSOR discusses the currently implemented features, as well as a guide to the file directories to manage these files as effectively as possible for any future updates, maintenance work, and testing required.

alamPREPROCESSOR Features

The current features implemented in the alamPREPROCESSOR are as follows:

- (a) The Data Collector Module (DCM) was integrated into the alamPREPROCESSOR to allow the alamSYS to collect end-of-day historical data from the various stocks on the Philippine Stock Exchange (including the PSEI). EODHD data is collected using the web API.
- (b) The Data Processor Module (DPM) was built into the alamPREPROCESSOR to enable the alamSYS to perform the following tasks:
 1. Apply DMD-LSTM to each stock dataset.
 2. Create a list of stocks to buy and sell by applying the ALMACD to the last 200 days of actual stock data and the five predicted data points from the DMD-LSTM model; and
 3. Update the database with new information about which stocks to buy or sell.
- (c) The alamPREPROCESSOR handles items (a) and (b) automatically. This means that they are set to run at a specific time, which is 6 p.m. every Monday through Friday. Some errors are also handled automatically if they are internally related, which means they are not affected by external factors such as a slow and unstable internet connection, a downtime in the EODHD APIs, and so on.
- (d) In relation to item (c), all system logs are saved in the system's shared docker volume at '/data/db/', as discussed further in the previous chapter.

- (e) The alamPREPROCESSOR is also in-charge of adding the following data to the alamDB collection if it does not already exist when the alamSYS starts up:
1. Stock Info Collection
 2. Model Info Collection; and
 3. Stock Risks Profile Collection
- (f) In connection with item (b.3), the alamPREPROCESSOR also saves all historical stock suggestions in the old directories with the current date when they were saved. Furthermore, these data are no longer reflected in the database and are only stored internally in the system.
- (g) Other minor features connected to the utilities directory:
1. Database model definitions
 2. Database actions such as:
 - i. Connecting to the alamDB.
 - ii. Disconnecting to the alamDB.
 - iii. Purging the Buy Collections.
 - iv. Purging the Sell Collections.
 - v. Saving the Updated Stocks to Buy JSON file.
 - vi. Saving the Updated Stocks to Sell JSON file.
 - vii. Saving the Stocks Info Collection from the JSON file.
 - viii. Saving the Model Info Collection from the JSON file; and
 - ix. Saving the Stock Risks Profile Collection from the JSON file.
- (h) Finally, in relation to item (c), system maintainers or administrators can easily restart system critical processes using the system's command line interface provided by docker in cases where errors are not automatically handled by the system. Additionally, they can use the Python program to debug the cause of the error, and they can use vim to change some parts of the source code to fix the error, if ever needed.

alamPREPROCESSOR Docker Management

The management and maintenance of the alamPREPROCESSOR is made possible through the utilization of Docker Container Management tools such as the Docker Desktop, which is a GUI Windows-based software used to maintain and manage docker containers.

Figure 4.1 depicts the tree of files and directories accessible from '/preprocessor/' of the alamPROCESSOR docker container.

```
    └── data_collector
        ├── __init__.py
        └── __pycache__
            ├── __init__.cpython-311.pyc
            ├── collector.cpython-311.pyc
            └── test.cpython-38.pyc
    └── collector.py
    └── data_processor
        ├── __init__.py___.py
        └── __pycache__
            ├── alma_processor.cpython-311.pyc
            ├── ml_processor.cpython-311.pyc
            ├── ml_processor.cpython-38.pyc
            └── process.cpython-311.pyc
    └── alma_processor.py
    └── ml_model
        ├── README.md
        ├── __init__.py
        └── dmd_lstm.keras
    └── ml_processor.py
    └── process.py
    └── main.py
    └── manual_run.py
    └── requirements.txt
    └── utils
        ├── __init__.py
        └── __pycache__
            ├── __init__.cpython-311.pyc
            ├── db_actions.cpython-311.pyc
            ├── init_db.cpython-311.pyc
            ├── logs_and_alerts.cpython-311.pyc
            ├── logs_and_alerts.cpython-38.pyc
            ├── models.cpython-311.pyc
            └── stock_symbols.cpython-311.pyc
        └── db_actions.py
        └── init_db.py
    └── json_data
        ├── *
        ├── model_info.json
        ├── stock_info.json
        └── stock_risks.json
    └── logs_and_alerts.py
    └── models.py
    └── stock_symbols.py
```

8 directories, 36 files

Figure 4.1: alamPREPROCESSOR Docker Container Directory Tree

On the other hand, the alamPREPROCESSOR shares the following directory paths related to error logging and json directories with the other components of the alamSYS via the '/data/db' directory path:

- (a) /data_collector_logs - this is the directory for all the successful operations ('/success_log.csv') , and all the unsuccessful operations ('/error_log.csv') conducted by the data collector.
- (b) /data_processor_logs - this is the directory for all the successful operations ('/success_log.csv') , and all the unsuccessful operations ('/error_log.csv') conducted by the data processor.
- (c) /manual_run_logs - this is the directory for all the successful operations ('/success_log.csv') , and all the unsuccessful operations ('/error_log.csv') conducted by manually running the operations of the alamPREPROCESSOR using the command 'python3 manual_run.py'.
- (d) /preprocessor_utils_logs - this is the directory for all the logged actions conducted by utilities of alamPREPROCESSOR. This is further divided into two directories, which are as follows:
 - 1. db_actions - stores logs of successful operations ('/success_log.csv'), and all the unsuccessful operations ('/error_log.csv') from the utilities related to database actions.
 - 2. init_db - stores logs of successful operations ('/success_log.csv') , and all the unsuccessful operations ('/error_log.csv') from the utilities related to the initialization of the database.
- (f) /scheduled_task_logs - this is the directory for all the successful operations ('/success_log.csv') , and all the unsuccessful operations ('/error_log.csv') conducted by the scheduled operation of the alamPREPROCESSOR.

4.1.2 Documentation for alamAPI and alamDB

The alamAPI and alamDB documentation section discusses the currently implemented features for the API and database, respectively. This should be used as a guide for any future updates, maintenance, and future testing.

alamAPI API Endpoints

This section contains the alamAPI endpoints whose Swagger-based web documentation is accessible via this local link: `localhost:8000/alamAPI/v1/docs`, as shown in Figure 4.2. Alternatively, Redocly's alamAPI API documentation can be accessed via this link: `localhost:8000/alamAPI/v1 docs`, as shown in Figure 4.2.

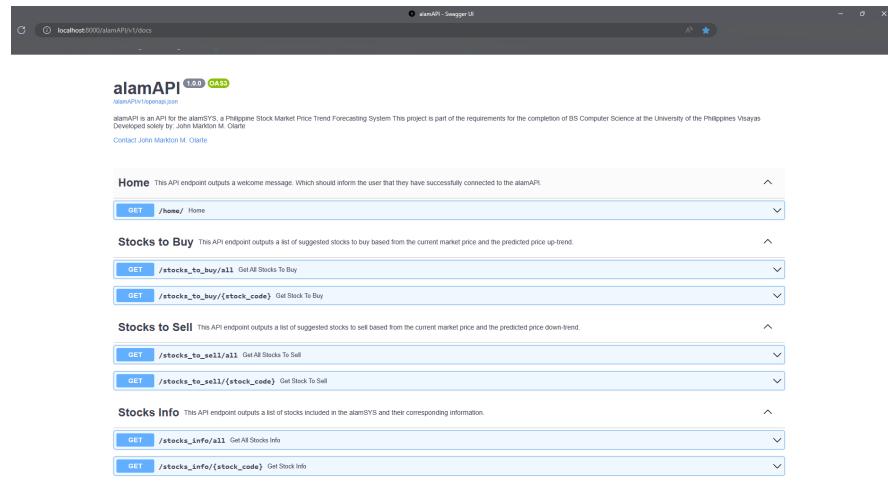


Figure 4.2: alamAPI Web-based API Documentation Using Swagger

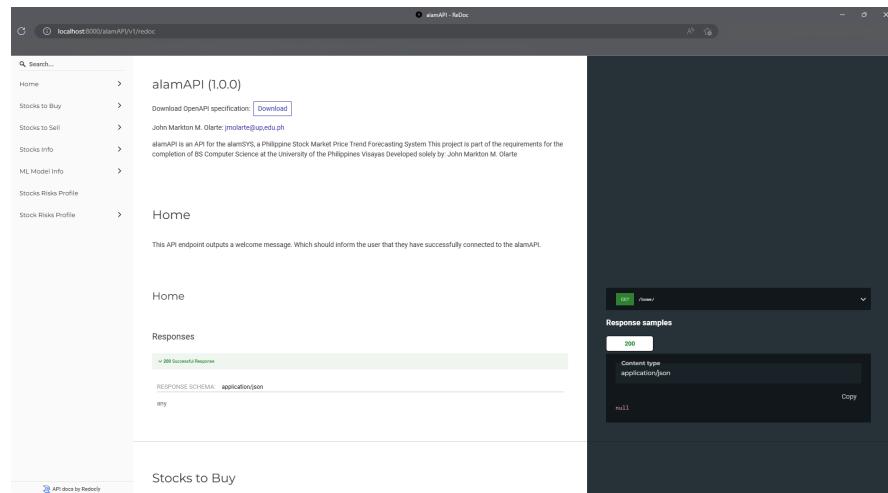


Figure 4.3: alamAPI Web-based API Documentation Using Redocs

Moving on, the Swagger-based documentation was used to show the API Endpoints. To begin, the 'Home' refers to the API endpoint that returns a welcome

message informing the API user that they have successfully connected to the alamAPI and that everything is functioning properly. Figure 4.4 depicts an example request executed using this API endpoint.

The screenshot shows a detailed view of a REST API endpoint for the 'Home' resource. At the top, there's a header bar with 'GET /home/ Home'. Below it, a 'Parameters' section indicates 'No parameters'. A prominent 'Execute' button is centered above a 'Responses' section. The 'Responses' section is expanded to show details for a 200 status code. Under 'Response body', a JSON object is displayed:

```
{
  "message": "Welcome to alamAPI an API for the alamSYS a Philippine Stock Market Price Trend Forecasting System, developed by John Markton M. Olarte as a partial requirement for the degree of Bachelor of Science in Computer Science at the University of the Philippines - Visayas."
}
```

Under 'Response headers', the following information is shown:

```
content-length: 281
content-type: application/json
date: Wed, 18 Apr 2023 04:53:28 GMT
server: uvicorn
```

Below the main responses, there's a 'Responses' section for a 200 status code, labeled 'Successful Response'. It includes a dropdown for 'Media type' set to 'application/json' and a 'Example Value - Schema' field containing the string 'string'.

Figure 4.4: Sample API Endpoint: Home Request

Next, API endpoints related to the 'Stock to Buy', which output a list of suggested stocks to buy based on the current market price and the predicted price up-trend. This API endpoint has two uses: users can use the /all path to see all of the suggested stocks to buy, or they can use /stock_code, which requires a specific stock code and returns the details for that stock if it exists in the stocks to buy list, and if it does not, it instead returns a 'Stock not found' alert. A sample execution, specifically for /all is shown is shown in Figure 4.5

The screenshot shows a detailed view of a REST API endpoint for 'Stocks to Buy'. At the top, a header bar indicates the endpoint is '/stocks_to_buy/all' and its description is 'Get All Stocks To Buy'. Below this, there are sections for 'Parameters' (none listed), 'Responses' (including 'Curl' command, 'Server response' (status 200), and 'Response headers'), and 'Responses' (Code 200, Description 'Successful Response'). The 'Server response' section displays a JSON object representing two stocks:

```

{
  "stocks": [
    {
      "id": "6448b14f576dd851cff34460",
      "stock_symbol": "MOO",
      "last_update": "2023-04-25",
      "predicted_change": [
        {
          "date": "2023-04-26",
          "low": 129.00000000000002,
          "high": 131.00000000000001,
          "mid": 130.50173404116404,
          "open": 130.50000000000002,
          "close": 129.5654473876993
        }
      ],
      "id": "6448b14f576dd851cff34461",
      "stock_symbol": "FISH",
      "last_update": "2023-04-25"
    }
  ]
}

```

The 'Response headers' section shows standard HTTP headers like Content-Type: application/json, Date: Wed, 26 Apr 2023 05:06:35 GMT, and Server: uvicorn.

Figure 4.5: Sample API Endpoint: Buy (all) Request

The third set of API endpoints is related to the 'Stock to Sell,' which returns a list of recommended stocks to sell based on the current market price and the predicted price downtrend. Users can use the /all path to see all of the suggested stocks to sell, or they can use /stock_code, which requires a specific stock code and returns the details for that stock if it exists in the stocks to sell list, and a 'Stock not found' alert if it does not. Figure 4.6 shows a sample execution, specifically for /all.

The screenshot shows a REST API endpoint titled "Stocks to Sell". The endpoint description states: "This API endpoint outputs a list of suggested stocks to sell based from the current market price and the predicted price down-trend." Below the description, there is a configuration panel for a "GET /stocks_to_sell/all" request.

- Parameters:** No parameters.
- Responses:**
 - Curl:** curl -X 'GET' \ http://localhost:8000/stocks_to_sell/all \ -H 'Accept: application/json'
 - Request URL:** http://localhost:8000/stocks_to_sell/all
 - Server response:**
 - Code:** 200
 - Response body:**

```
{
  "stocks": [
    {
      "_id": {
        "field": "644bb54f576dd851cff1446a"
      },
      "stock_symbol": "WMT",
      "last_changeing": "-1.98",
      "last_change": "-25",
      "predicted_change": {
        "end_value": [
          1.937136038052079,
          1.937137042207362,
          1.937134699224129,
          1.935530310942084
        ]
      }
    },
    {
      "_id": {
        "field": "644bb54f576dd851cff1446b"
      },
      "stock_symbol": "WMT",
      "last_change": "-0.5"
    }
  ]
}
```

 - Response headers:**
 - content-length: 3428
 - content-type: application/json
 - date: Wed, 26 Apr 2023 05:12:14 GMT
 - server: uvicorn
 - Responses:**
 - Code:** 200
 - Description:** Successful Response
 - Links:** No links
 - Media type:** application/json
 - Content-Security-Policy:**
 - Example Value | Schema:** "string"

At the bottom, there is a link for another endpoint: "GET /stocks_to_sell/(stock_code) Get Stock To Sell".

Figure 4.6: Sample API Endpoint: Sell (all) Request

The next set of API endpoints is related to the 'Stocks Info', which returns a list of stocks that are part of the alamSYS. This API endpoint has two uses: users can use the /all path to see information from all 20 stocks included in this iteration of the alamSYS, or they can use /stock_code, which requires a specific stock code and returns information from that specific stock if it exists in the alamSYS, and a 'Stock not found' alert if it does not. Figure 4.7 shows a sample execution, specifically for /all.

The screenshot shows a sample API endpoint for 'Stocks Info' with the path '/stocks_info/all'. The interface includes:

- Curl:** A command-line tool for transferring data with URLs. The command shown is:


```
curl -X GET \
http://localhost:8000/stocks_info/all \
-H "Content-Type: application/json"
```
- Request URL:** http://localhost:8000/stocks_info/all
- Server response:**
 - Code:** 200
 - Response body:** A JSON object representing stock information. The JSON starts with:


```
{
  "All Stocks Info": [
    {
      "id": "642edc382358f720fffb43ed",
      "stock_symbol": "RGW",
      "stock_name": "Regent's Corporation",
      "company_email": "info@regentscorp.com",
      "company_address": "20th Floor, Alliance Global Tower, 36th Street cor. 11th Avenue, Uptown Bonifacio, Taguig City 1634, Metro Manila, Philippines",
      "company_website": "investor.regentscorp.com",
      "sector": "Real Estate Sector",
      "industry": "Real Estate Development",
      "key_executive": [
        {
          "name": "Andrew L. Tan",
          "position": "Chairman"
        },
        {
          "name": "Katherine L. Tan",
          "position": "Director"
        }
      ]
    }
  ]
}
```
 - Response Headers:**
 - content-length: 18455
 - content-type: application/json
 - date: Wed, 26 Apr 2023 05:38:41 GMT
 - server: uvicorn
- Response:**
 - Code:** 200
 - Description:** Successful Response
 - Links:** No links
 - Media type:** application/json
 - Example Value | Schema:** string

Figure 4.7: Sample API Endpoint: Stocks Info (all) Request

The fifth set of API endpoints is related to the 'ML Model Info', which returns a list of machine learning models that have been deployed in the alamSYS. In particular, in this iteration of the alamSYS, the current deployed deep learning model is DMD-LSTS, as discussed in Chapter 3. This API endpoint has two uses: users can use the /all path to see information for all models, or they can use /model_name, which requires a model name and returns information for that specific model if it exists in the alamSYS, and a 'Model not found' alert if it does not. Figure 4.8 shows a sample execution, specifically for /all.

ML Model Info This API endpoint outputs a list of the Machine Learning Models used in the alamSYS and their corresponding information.

GET /ml_model_info/all Get All Models Info

Parameters

No parameters

Responses

Curl

```
curl -X GET \
  http://localhost:8080/ml_model_info/all \
  -H 'accept: application/json'
```

Request URL

http://localhost:8080/ml_model_info/all

Server response

Code	Details
200	Response body <pre>{ "All Models Info": [{ "id": "1422efc3e23687269fffa3bd", "model_type": "DNN-LSTM", "model_desc": "Implemented dynamic nodes from Dynamic Mode Decomposition (DMD) to the training input layer of LSTM", "model_accuracy": [{ "test": "0.0032", "repeat": "0.0031", "mean": "0.0032", "rse": "0.0045" }], "Date and Time": "2023-05-23T21:19:38.88762" }] }</pre> <p>Response headers</p> <pre>content-length: 339 content-type: application/json date: Wed, 23 May 2023 23:39:38 GMT server: uvicorn</pre>

Responses

Code	Description	Links
200	Successful Response	No links

Content-Type

application/json

Example Value | Schema

string

GET /ml_model_info/{model_name} Get Model Info

Figure 4.8: Sample API Endpoint: ML Model Info (all) Request

Finally, the final API endpoints related to the 'Stocks Risks Profile', which outputs a list of stocks in alamSYS and their corresponding risk values based on value at risk (%), volatility (%), and drawdown (%). This API endpoint has two uses: users can use the /all path to see the risks profile information for all stocks, or they can use /stock_name, which requires a stock name and returns the risks information for that specific stock if it exists in the alamSYS, and a 'Stock not found' alert if it does not. Figure 4.9 depicts a sample execution, specifically for /all.

The screenshot shows a REST API endpoint for 'Stock Risks Profile'. The endpoint is `/risk_info/all`. The response body is as follows:

```

{
  "All Stock Risks Profile": [
    {
      "_id": {
        "$oid": "642edc3833687309ff943be"
      },
      "stock_symbol": "PSEI",
      "value_at_risk": -5.35715512,
      "volatility": 3.18183686,
      "drawdown": 41.1840442109,
      "start_date": "2000-01-03",
      "end_date": "2023-02-18"
    },
    {
      "_id": {
        "$oid": "642edc3833687309ff943be"
      },
      "stock_symbol": "GME",
      "value_at_risk": -5.35715512,
      "volatility": 3.18183686,
      "drawdown": 41.1840442109,
      "start_date": "2000-01-03",
      "end_date": "2023-02-18"
    }
  ]
}

```

Figure 4.9: Sample API Endpoint: Stocks Risks Profile (all) Request

To go over the following risk values in greater detail. Furthermore, Table 4.1 summarizes the risk values for all stocks included in the alamSYS:

- (a) Value at Risk (VaR) is a statistic that quantifies the extent of potential financial losses within a firm, portfolio, or position over a specific time period, and risk managers use VaR to measure and control the level of risk exposure (Kenton, 2023). The alamSYS calculated the value at risk based on the daily return percentage of each stock from January 03, 2000 to February 02, 2023, that is below 5%. In Table 4.1 it shows a negative VaR, which means that percentile amount is the expected returns below 5% is computed for that values. For example, PSEI shows -1.88780% VaR which means that it has a 1.88780% chance to loss 5% of its value for the next day, while having

99.11220% chance of gaining above 5% on a daily basis. Take note that the negative symbol in VaR indicates the likelihood of losing in the position.

- (b) Volatility is a statistical indicator of the spread of returns for a specific security or market index. The higher the volatility, in most cases, the riskier the security (Hayes, 2023). It was measured in the alamSYS in terms of the volatility percentage based on standard deviation of daily returns of each stock from January 3, 2000 to February 2, 2023. The data presented in Table 4.1 shows that the LTG stock is the most volatile of the stocks included in the alamSYS, implying that investors or traders should keep a closer eye on LTG than other stocks with volatility ranging from around 1.3% to 7%.
- (c) Drawdown is useful for calculating the historical risk of various investments, comparing fund performance, and monitoring personal trading performance (Mitchell, 2023). It was measured in the alamSYS in terms of the drawdown percentage based on the maximum and minimum values of each stock from data beginning January 03, 2000 until February 02, 2023, essentially to measure the stock's performance during that time period. The overall drawdown values from the alamSYS stocks, as shown in Table 4.1, show a positive upside, but this also indicates the possibility of reversed risk, where a drawdown of 50%, for example, can also mean a potential loss of 50% of an investor's total capital.

Table 4.1: Summary of Risk Profile for Each Stock in the alamSYS

STOCK	VALUE AT RISK (%)	VOLATILITY (%)	DRAWDOWN (%)
MEG	-5.36572	3.94997	57.24814
JGS	-4.76236	3.36110	43.18404
BDO	-3.30249	2.36405	39.67065
FGEN	-3.81908	2.55925	30.60426
ICT	-4.82705	3.52084	54.64501
ALI	-4.39048	3.07049	56.02617
SMC	-3.40367	2.38596	45.29878

Table 4.1 continued from previous page

STOCK	VALUE AT RISK (%)	VOLATILITY (%)	DRAWDOWN (%)
TEL	-3.69376	2.46002	44.65912
GLO	-4.12004	3.09260	54.75806
BLOOM	-5.98500	7.06155	100.00000
RLC	-4.52936	3.41699	65.98291
MER	-4.49859	3.25474	76.00003
AC	-4.29065	2.79617	46.68883
PGOLD	-3.11492	2.13482	25.79979
LTG	-6.15322	31.22332	1667.80691
MPI	-4.05584	3.49968	81.13252
AP	-3.19764	2.18729	26.54088
RRHI	-3.03206	2.05393	24.50967
URC	-4.53239	3.19972	42.54262
PSEI	-1.88780	1.31882	30.90366

alamAPI Use Case Diagram

Based on the API endpoints discussed in the previous section, the author has illustrated the use case of the alamAPI in this section. Furthermore, the use case diagram in Figure 4.10 adheres to the Unified Modeling Language (UML) definition of a use case diagram as published in Visual Paradigm's (n.d.) guides. *Note that this only pertains to the interaction of potential users in the alamAPI, and not to the whole alamSYS.*

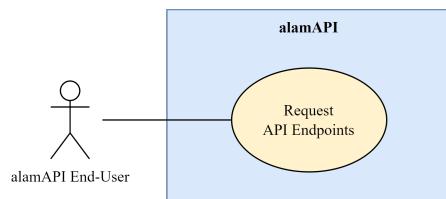


Figure 4.10: alamAPI End User UML Use Case Diagram

The use case diagram above demonstrates how easy and simple it is to use the alamAPI, as end users are only required to create requests from the listed API endpoints.

alamAPI Docker Management

The management and maintenance of the alamAPI container is also made possible through the utilization of Docker Desktop. Figure 4.11 depicts the tree of files and directories accessible from '/alamAPI/' of the alamAPI docker container.

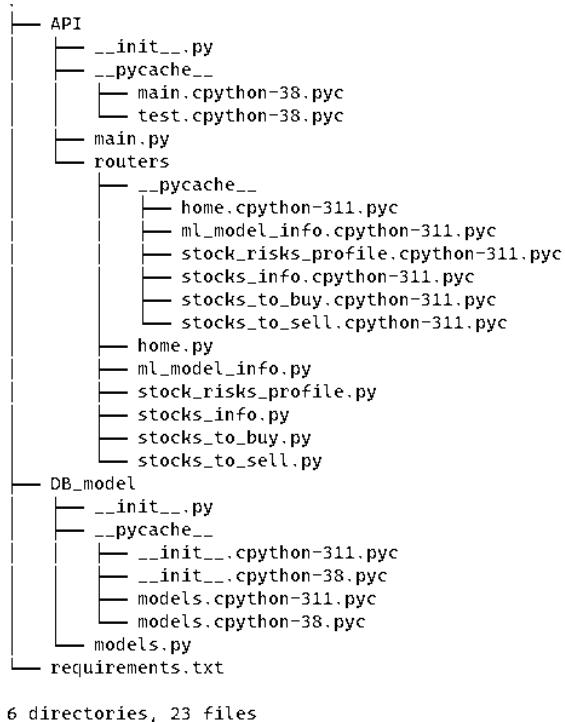


Figure 4.11: alamAPI Docker Container Directory Tree

On the other hand, alamDB's docker container management only displays alamDB status updates, whereas direct management employs known mongoDB command line interface commands. However, utilization of CLI commands through docker desktop can be time-consuming for system administrators or system managers, so it is best to use a GUI-based mongoDB management tool, as discussed

in the succeeding section.

alamDB MongoDB Compass Management

The use of a GUI-based database management tool for alamDB is critical in maintaining the system's ease of maintenance. As a result, the author recommends that future system administrators or maintainers use software such as MongoDB Compass, which is available for Windows users.

Utilization of the MongoDB Compass is straightforward and is only composed of these few steps:

- (a) First, indicate a new connection using the default URI: `mongodb://localhost:27017`.
- (b) Once the URI is initiated, press or tap the 'Connect' button to establish connection to the alamDB.
- (c) Finally, you have access to the alamDB. As shown in Figure 4.12 system users has the access to the following databases:
 1. admin - which currently have an empty collection of data
 2. alamAPI_DB - which corresponds to the database of the alamSYS
 3. config - which is also currently empty; and
 4. local - which contains the `startup_log` collection, and whose data is autogenerated by mongoDB.

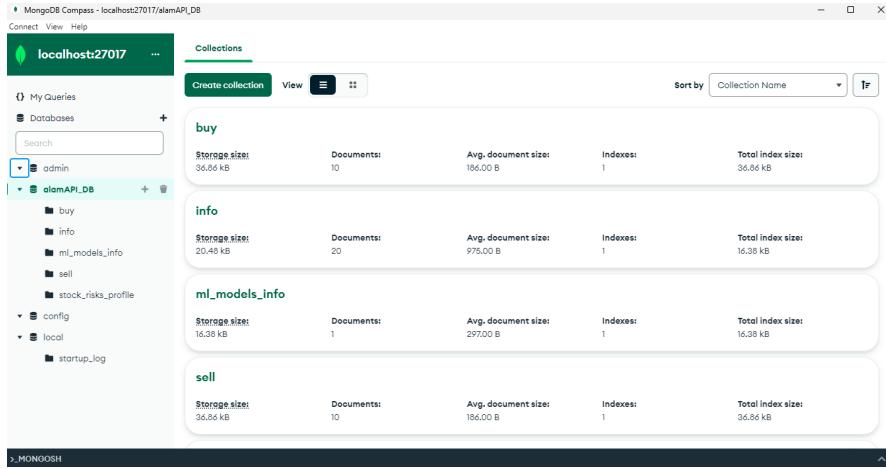


Figure 4.12: MongoDB Compass Database Management of alamDB

4.1.3 Documentation for alamAPP

The alamAPP is a mobile-based test application developed to demonstrate the use of the alamSYS to external internet-connected devices, such as a smartphone application. It was created with the Dart-based Flutter SDK. The user interface and code snippets of how the alamSYS was used in the alamAPP are shown in this section. Furthermore, the full source code for the alamAPP is available in the GitHub repository listed in Appendix A of this paper.

User Interface Showcase

This section displays the alamAPP’s developed user interfaces. It should be noted that this application only highlights the alamSYS’s two most important features, which are its ability to recommend which stocks to buy and which to sell. Additional enhancements to the application are discussed in Chapter 5 of this paper.

The ability of the alamSYS to recommend stocks that are good to buy for the day is one of the key features of the alamSYS that is highlighted in the alamAPP. Users are able to see the list of stocks to buy as well as the necessary information about those stocks, including: (1) The last price as indicated below the stock

symbol; and (2) The predicted price in 5 days as indicated in the green colored number at the rightmost part of the card listing. This is demonstrated in the alamAPP through the "Buy" Screen, presented in Figure 4.13. These details enable users to quickly assess a stock's potential return and choose from the list of stocks more wisely.

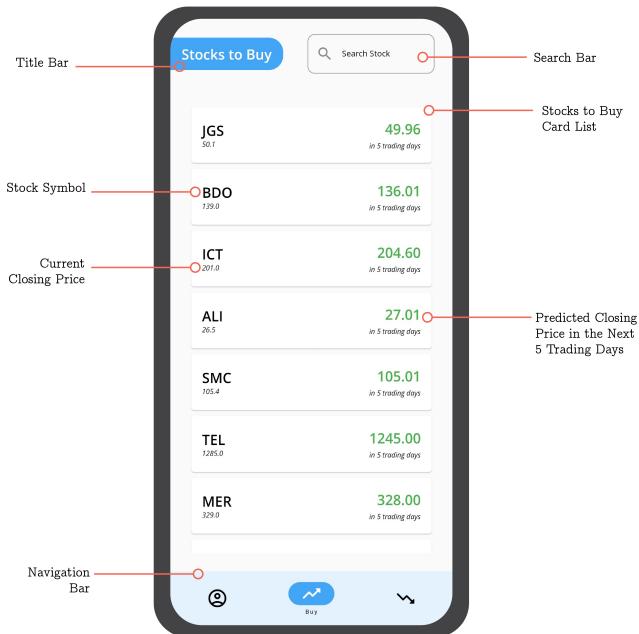


Figure 4.13: alamAPP "Buy" Screen

Another key feature of the alamSYS that is highlighted in the alamAPP is its ability to suggest stocks that must be sold for the day. This is demonstrated in the alamAPP by the "Sell" Screen, as shown in Figure 4.14, where users can see a list of which stocks to sell as well as the necessary information about those stocks, such as: (1) the last price as indicated below the stock symbol, and (2) the predicted price in 5 days as indicated in the red colored number at the rightmost part of the card listing. This information allows users to see the potential loss of the stock at a glance and better mitigate their risk.

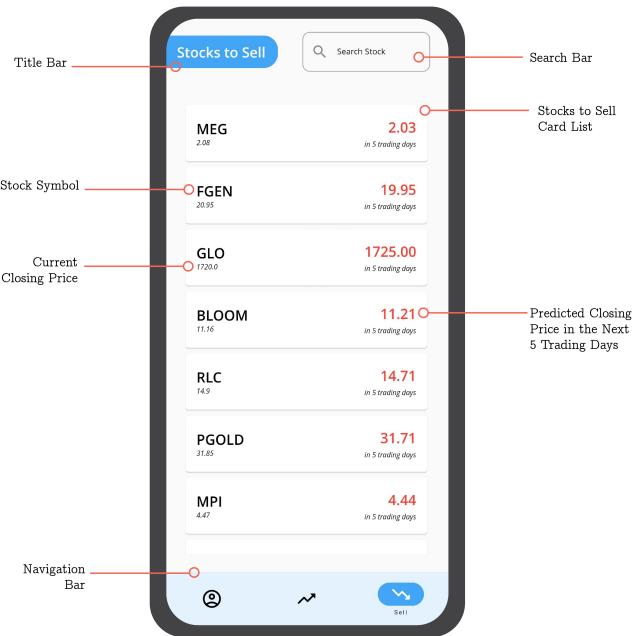


Figure 4.14: alamAPP "Sell" Screen

Users have the option of tapping or selecting the listed stocks in the "Buy" and "Sell" screens to display additional information about the specific stock, including the predicted price graph of the stock (presented in Figure 4.15), which will help users make decisions based on information and further reduce potential risk.

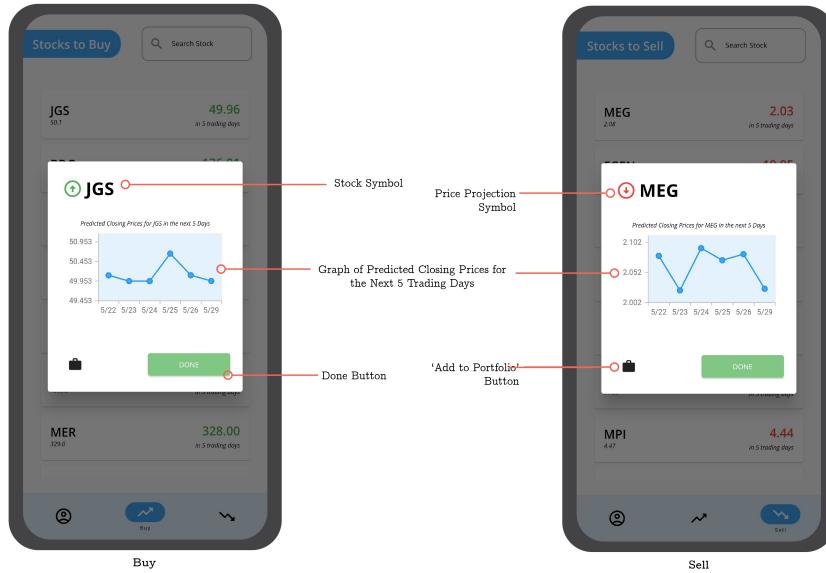


Figure 4.15: alamAPP Prediction Graph Card

Overall, the figures depicting the alamAPP's user interface demonstrate that the alamAPI can be used by external applications by connecting via the internet.

Utilization of alamSYS in the alamAPP

The following code snippet written in Dart, was used in the alamAPP's source code to get the details of the stocks to buy from alamAPI. Note that the [reserved.domain] should be changed to the domain address used on the alamAPI deployment.

```
Future<List<Stocks>> stocksToBuy() async {
    var url = 'https://[reserved.domain]/stocks_to_buy/all';
    var response = await http.get(Uri.parse(url));
    var data = jsonDecode(response.body);
    List<Stocks> stocksToBuy = [];
    if (data['Stocks'] != null) {
        data['Stocks'].forEach((v) {
            stocksToBuy.add(new Stocks.fromJson(v));
        });
    }
}
```

```
    }
    return stocksToBuy;
}
```

Meanwhile for the stocks to sell, the following code snippet was used:

```
Future<List<Stocks>> stocksToSell() async {
    var url = 'https://[reserved.domain]/stocks_to_sell/all';
    var response = await http.get(Uri.parse(url));
    var data = jsonDecode(response.body);
    List<Stocks> stocksToSell = [];
    if (data['Stocks'] != null) {
        data['Stocks'].forEach((v) {
            stocksToSell.add(new Stocks.fromJson(v));
        });
    }
    return stocksToSell;
}
```

4.1.4 Build and Deployment Guide

The build and deployment of the alamSYS docker compose are covered in this section. The discussion is divided into two sections. The first section discusses how to build and run the alamSYS docker compose. The second section discusses the current implementation's deployment using a low-cost tunneling service.

Building and Running Docker Compose of alamSYS

The alamSYS is deployed using docker compose containers, which is a docker tool that allows multiple containers to run concurrently. The docker compose file, on the other hand, is specific to the alamSYS and runs three containers: alamPREPROCESSOR, alamAPI, and alamDB.

Assuming the server device already has docker installed and a copy of the alamSYS source code, which can be accessed via the link in Appendix A, the

following '.env' files should be created and added to the alamSYS root folder.

The first '.env' file is the 'db.env' file, which contains the alamDB's environment variables. Wherein, the file contains the following details as indicated in the code snippet below.

```
# MongoDB
MONGO_INITDB_DATABASE="alamAPI_DB"
MONGO_HOST="mongo_db"
MONGO_PORT=27017

# Timezone
TZ="Asia/Hong_Kong"
```

And the other '.env' file is the 'preprocessor.env', which contains the alamPREPROCESSOR's environment variables. Wherein, the file contains the following details as indicated in the code snippet below. Note that the EOD_API_KEY indicated should be changed to the actual API KEY provided in the account of the user and should not be shared to anyone or publicly. Moreover, other than the API KEY, there is nothing to be changed.

```
# Change API KEY to actual API KEY value
EOD_API_KEY=000000000000

# MongoDB
MONGO_INITDB_DATABASE="alamAPI_DB"
MONGO_HOST="mongo_db"
MONGO_PORT=27017

# Timezone
TZ="Asia/Hong_Kong"

# ml_processor
MODEL_PATH="model"
```

At the end the folder structure of the alamSYS, should look like the one indicated in Figure 4.16.



Figure 4.16: alamSYS Required Folder Structure

After creating and adding the necessary '.env' files to the alamSYS root folder. Using the following command, we can now build and run the docker compose command from the terminal:

```
docker-compose build && docker-compose up
```

The aforementioned command creates the alamSYS containers and then runs them all. Furthermore, in subsequent runs, simply run the second part of the command 'docker-compose up' to initialize the alamSYS containers. Only build the containers on the first initialization or if the alamSYS source code has changed.

Deployment using Network Tunnelling Services

In order to minimize the cost of alamSYS deployment, the developer decided to use the LocalXpose service, which is a network tunneling service. It is a reverse proxy that allows you to expose your localhost services to the internet and can be found at <https://localxpose.io/docs/>

LocalXpose's official documentation contains device-specific instructions. Whereas, the deployment instructions in this section are based on the Windows 11 operating system.

To begin, we must create a reserved domain in the LocalXpose Dashboard, which can be accessed via this link: <https://localxpose.io/dashboard/domain>. In the case of alamSYS, the reserved domain was set to the Asia Pacific (AP) region

in order to maintain relatively strong data transfers. Following that, the user must download the binary program for their device’s operating system; in the author’s case, he downloaded the Windows 64-bit binary file.

The user should launch the command prompt (cmd) from the directory containing the downloaded file. It should be noted that, as of the writing of this paper, the binary file only works with cmd, and using powershell or windows terminal causes errors. To run the binary application loclx, enter the following command into cmd:

```
loclx.exe account login
```

The LocalXPose application would then prompt the user for an access token, which can be obtained at <https://localxpose.io/dashboard/access>. Following that, the user should enter the following command:

```
loclx tunnel http --reserved-domain [reserved.domain] --to  
8000
```

The last command launches the localx tunnel command in http mode with a reserved domain. And the value 8000 at the end indicates that the reserved domain will expose port 8000. This corresponds to the alamAPI’s network port, making the alamAPI accessible via any network with an internet connection. It should also be noted that the user should replace '[reserved.domain]' with the actual reserved domain they created in the LocalXpose Dashboard. Finally, the application will now show the status of the connection.

4.2 alamSYS System Tests Results and Discussions

With the development of alamSYS, we must ensure that all of its components are functioning properly. This section focuses on the system’s performance while

idle and under load, as well as the API’s and database’s ability to handle multiple requests at a time.

The Table 4.2 shows the CPU and memory utilization of each of the alamSYS components whenever it is not processing any information. Wherein, the data presented below is gathered by logging the system utilization of alamSYS within an hour.

Table 4.2: Idle System Average Resource Usage Statistics

	alamAPI	alamDB	alamPREPROCESSOR
CPU Utilization (%)	0.168125	0.254313	0.009769
Memory Utilization (MiB)	45.718311	166.775377	312.798300

From the table above, it is shown that the alamSYS as a whole only utilizes 0.432207% of the total CPU power on average when it is on idle. Wherein, the bulk of the CPU power is being used by the database at 0.254313% which is 58.84% of the total average CPU utilization of the alamSYS.

This result actually shows promising idle performance, as an Ubuntu system’s normal idle CPU utilization is less than 10%. However, it should be noted that the alamSYS is not completely idle because background tasks such as scheduling, mongoDB processes, and the API must remain active in order to respond to any API queries.

Furthermore, the lower average CPU utilization for alamAPI and alamPREPROCESSOR could be attributed to the linux distribution used as their base image, as previously discussed on ‘Docker-Compose Layer Diagram’ of Section 3.2.

The CPU utilization for the alamPREPROCESSOR, in particular, was lower than we would have expected given that it is running a schedule checker every second. This indicate that the schedule library utilizes an efficient way to check for schedules. However, it may not be the case for its memory utilization, which is discussed further below.

Moving on, the alamSYS's memory utilization shows that it uses 525.291988 Mebibytes (MiB) or 550.808572 Megabytes (MB) on average. The memory utilization of alamPREPROCESSOR accounts for the majority (59.55%). This demonstrates that, despite having the lowest CPU utilization, the alamPREPROCESSOR consumes more memory than the combination of the alamAPI and alamDB.

Again, as previously discussed, the alamPREPROCESSOR's high average memory utilization is due to the background schedule checking, which in this case is programmed to use more memory than CPU power.

This average utilization result implies that the alamSYS may be able to be deployed on devices with lower specifications. A Raspberry Pi 4, which has a quad core CPU and at least 2GB RAM, is one example (Zwetsloot, 2019). However, idle performance only shows the minimum CPU and memory utilization of the alamSYS components and does not provide a complete picture of its utilization, particularly under load. As a result, we must investigate the system's CPU and memory utilization while under load.

The internal load averages system utilization of the alamSYS' preprocessor is shown in the following table.

Table 4.3: Internal Load Average Resource Usage Statistics

	Data Collector	Data Processor	PREPROCESSOR (Data Collector & Data Processor)	alamSYS
Failure Rate (%)	0	0		0
Success Rate (%)	100	100		100
Average Runtime (s)	41.72398	8.38061		48.30466
Average CPU Utilization (%)	11.40659	92.71117		20.03138
Average Memory Utilization (MiB)	3.64200	57.09545		794.29436
Average Network Utilization (Mb)	232.73640	154		77.27655

Before exploring into the results shown in Table 4.3, it is critical to first establish a context for how the data was gathered. The data in the above table was collected while the alamSYS, specifically the alamPREPROCESSOR, was subjected to a stress test load of 100 consecutive data collection and processing. Also, the data of average utilization as indicated for each column are independently gathered from each other.

Based on the table above, each component was capable of processing 100 consecutive processes without failure. This means that the alamSYS, specifically the alamPREPROCESSOR, can run at least 100 times in a row without failing. Also, keep in mind that the alamPREPROCESSOR only collects and processes data once per day, and the developer has included an option for system users or maintainers to manually rerun these processes in cases where the alamPREPROCESSOR fails - for example, due to a lost or slow internet connection, a power

outage, and so on.

Meanwhile, the data processor's average runtime is faster than the data collector's, and it runs on an average of 48.30466 seconds. This was already expected because the data collector needed to connect to the internet and was thus constrained by internet speed. Whereas the average speed during the course of this test was 52.98Mbps, this could imply that the data collector's runtime may be slower or faster depending on the internet speed.

Furthermore, the data processor's average runtime was surprisingly fast given that it needed to apply DMD-LSTM to a total of 20 stocks and calculate the position using the ALMACD given a total of 205 data points. Furthermore, looking at its CPU and memory utilization, it can be seen that it uses more than the data collector, with approximately 87.68% more average CPU power used and approximately 93.63% more average memory used.

In line with the CPU and memory utilization, the average CPU utilization of the alamPREPROCESSOR on load is 99.95% higher than when it is idle. And it uses 60.62% more memory on average.

Lastly, looking at the network utilization of the alamPREPROCESSOR, we can see that the data collector used the most network bandwidth, as expected. However, it may be surprising to see that the data processor uses the network when it should not because it does not need to process or collect data from the internet, but this is still within expectations because network bandwidth utilization also accounts for local network utilization. The data processor of alamPREPROCESSOR, in particular, uses the local network to connect to and update the new data in the alamDB.

The following tables show the system's deployment load results. Specifically, to access the buy and sell collections, as shown in Tables 4.4 and 4.5. Additionally, to have a background on the test that results in the following outcomes. It should

be noted that the test consisted of three instances of the same tester application running on ten different computers. Where each application requests 10, 100, 1000 buy and sell data from the alamDB via the alamAPI, which is tunneled over the internet for server access outside the university's local network using LocalXpose services.

Table 4.4: Deployment Load Test Results (Buy Requests)

	Number or Requests		
	10	100	1000
Success Rate (%)	100	100	100
Average Processing Time (s)	11.905222	139.618550	1159.773569

Table 4.5: Deployment Load Test Results (Sell Requests)

	Number or Requests		
	10	100	1000
Success Rate (%)	100	100	100
Average Processing Time (s)	13.384126	130.119867	1642.995011

As per the tables above, the alamSYS was able to handle all consecutive and simultaneous requests from all external devices. It was able to handle 71,000 requests in approximately one hour and 20 minutes.

Furthermore, the data shows that a request is processed in 1.34 seconds on average. Wherein, the fastest processing time from the data collected was 0.93 seconds. This result is within the expected response time of APIs, where good is defined as 0.1 to one second, and any time between one and two seconds is acceptable, as long as it does not exceed two seconds, which users may perceive

as an interruption in the process (Juviler, 2022).

It is also worth noting that in a separate internal test for the alamSYS, the alamAPI only takes on average 0.0094 seconds to send its response. And that the additional delay in response time on deployment is caused by network-related factors, such as the time it takes the tunneling service to accept the query and send back the system's response. In fact the network related delay contributes 99.30% of the average deployment response time. Hence, it might be useful to deploy the system in a network that further minimizes this additional delay in the response time.

Meanwhile Table 4.6 shows the average CPU and memory utilization of alamAPI and alamDB as the load testing as stated above, was being processed.

Table 4.6: CPU and Memory Utilization Statistics of alamAPI and alamDB Under Deployment Load Testing

	alamAPI	alamDB
CPU Utilization (%)	17.725949	1.070133
Memory Utilization (MiB)	44.620837	129.273305

To further visualize the system utilization over time, Figure 4.17 shows the CPU utilization of alamAPI and alamDB over time. And Figure 4.18 shows the memory utilization of alamAPI and alamDB.

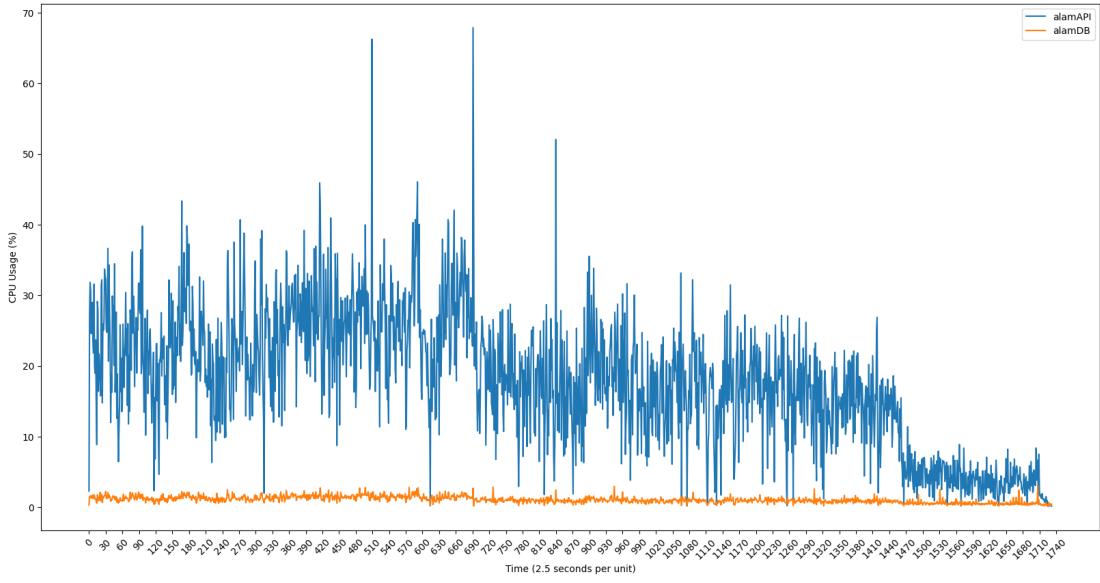


Figure 4.17: Deployment Load CPU Utilization of alamAPI and alamDB Over Time

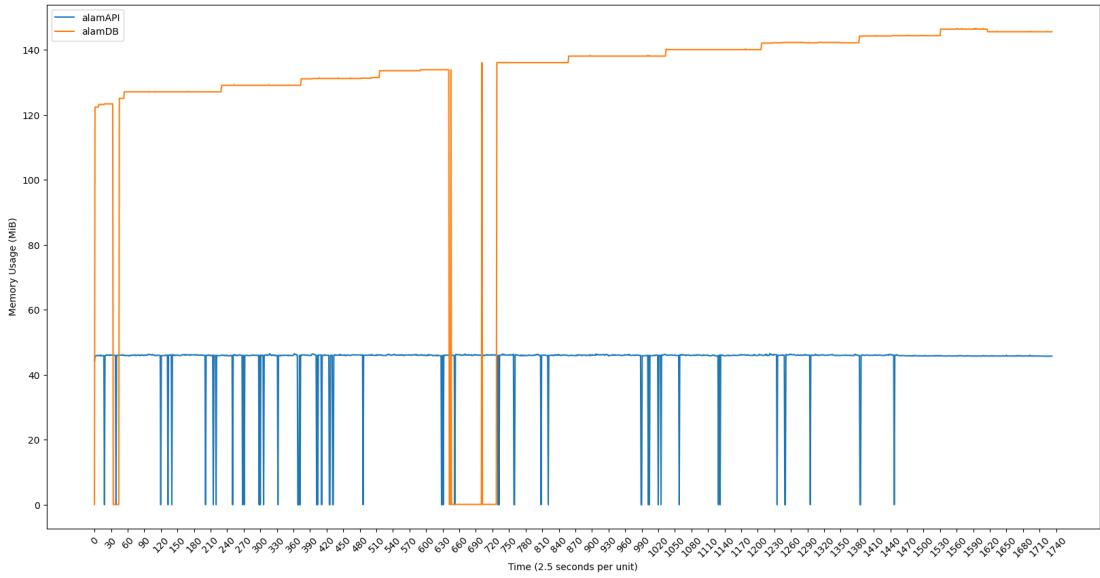


Figure 4.18: Deployment Load Memory Utilization of alamAPI and alamDB Over Time

Based on the table and figures above, processing all requests consumes only 17.73% and 1.07% of the CPU for alamAPI and alamDB, respectively. In terms of memory usage, they are 44.62 MiB and 129.27 MiB, respectively.

Furthermore, at around 1740 time units, the CPU utilization for alamAPI was shown to be reduced. This is due to the fact that most tester applications have already completed processing all of the requests that they are programmed to run, reducing the load on the server by half.

4.3 DMD-LSTM Model Results and Discussions

This section presents and discusses the Deep Learning Model's training, testing, and cross-validation results.

In Table 4.7 the training error metrics are shown for each of the window sizes tested.

Table 4.7: DMD-LSTM Training Error Metrics Scores
for Different Window Sizes

Error Metrics	Window Sizes			
	5	10	15	20
MSE	0.000037	0.787877	0.006917	0.057851
RMSE	0.006106	0.887624	0.083166	0.240522
MAE	0.004175	0.755407	0.067645	0.202746
MAPE	0.000001	0.000194	0.000017	0.000053

Where it is observed that the best performing model based on having the lowest MAPE score is the DMD-LSTM with a window size of 5. Moreover, we can see the differences from each MAPE score for each window size in the Figure 4.19 shown below.

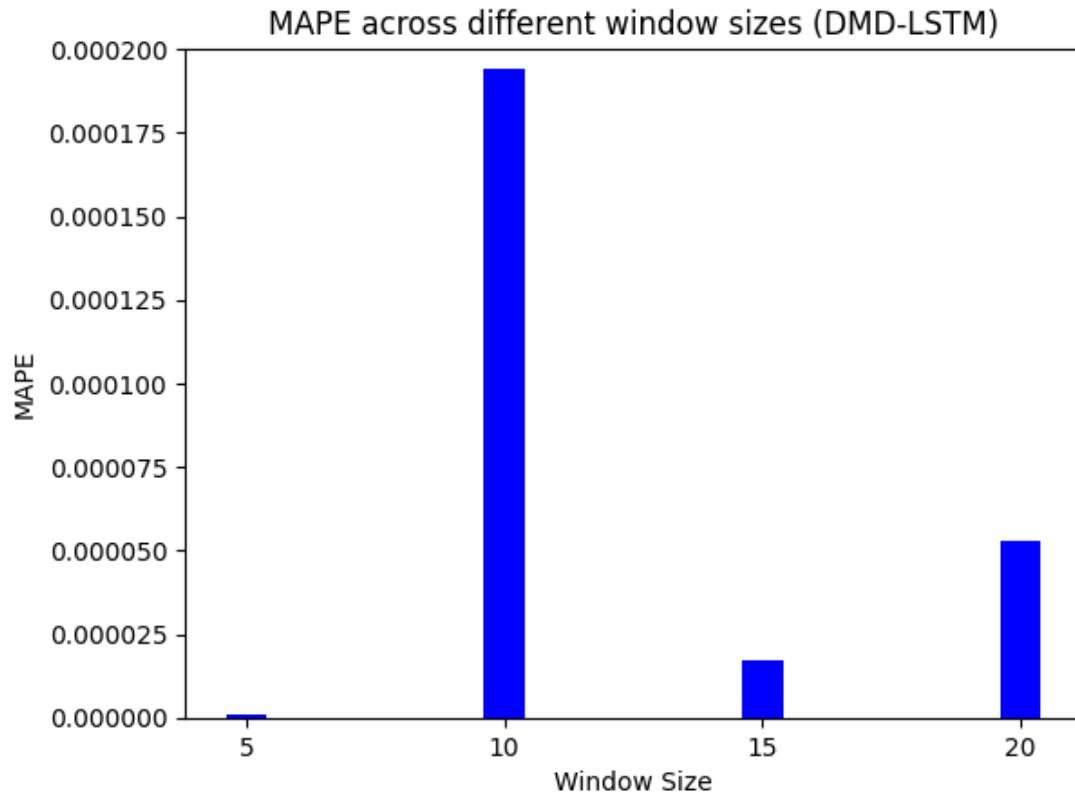


Figure 4.19: Comparison of MAPE Scores for DMD-LSTM Model Training Across Different Window Sizes

The figure above also shows that the MAPE score for window sizes 15 and 20 is higher than the MAPE score for window size 10. MAPE score increases from window size 15 to size 20, indicating that increasing window size may result in a lower performing model.

Furthermore, as previously stated, the window size of 5 results in the best MAPE score being the lowest. Where it outperforms the worst performing model (DMD-LSTM with window size 10) by 0.000193 units. As illustrated clearly in Figure 4.19.

Knowing that the DMD-LSTM model performs as expected based on the training data scores, it is critical that we also examine the training data results from

a baseline LSTM. The baseline LSTM is, as the name implies, a simple LSTM model lacking the DMD component. The table below shows the results of the baseline LSTM training.

Table 4.8: Baseline LSTM Training Error Metrics Scores
for Different Window Sizes

Error Metrics	Window Sizes			
	5	10	15	20
MSE	2912.840703	191.935882	1118.183283	706.136814
RMSE	53.970739	13.854093	33.439248	26.573235
MAE	35.301888	9.480864	22.099720	18.285352
MAPE	0.009618	0.002527	0.006024	0.005004

According to the table above, the baseline LSTM with window size 10 performs the best, with the lowest MAPE score of 0.002527 when compared to the other baseline LSTM models.

However, the DMD-LSTM model with window size 5 outperforms it by 0.002526. As a result, the alamSYS makes use of the DMD-LSTM model, specifically the one with a window size of 5. Where from now on, the DMD-LSTM model refers to the DMD-LSTM model with a window size of 5.

Nonetheless, the DMD-LSTM model's performance is limited to the training dataset from PSEI, and it must be cross-validated using data from other stocks, which includes the PSEI validation dataset. The results of this cross-validation is presented in Table 4.9. It should also be noted that cross-validation uses logarithmic normalization as a data preprocessing technique to make the dataset more normal, which aids in analyzing the model's performance with the given dataset. Normalization techniques, in particular, allow for closer variation within the forecasted data. (S.Gopal Krishna Patro, 2015).

Table 4.9: DMD-LSTM Cross-Validation Error Metrics Scores

Stocks	MSE	RMSE	MAE	MAPE
PSEI	0.00002	0.00419	0.00328	1.510000e-03
AC	0.00236	0.04856	0.03414	6.110000e-03
ALI	0.00255	0.05054	0.03645	1.597000e-02
AP	0.00129	0.03596	0.02515	9.220000e-03
BDO	0.00160	0.03999	0.02799	7.250000e-03
BLOOM	0.01883	0.13721	0.06901	1.052898e+12
FGEN	0.00224	0.04733	0.03265	1.197000e-02
GLO	0.00211	0.04595	0.03149	4.680000e-03
ICT	0.00335	0.05785	0.03731	3.005818e+11
JGS	0.00331	0.05752	0.03992	2.009923e+11
LTG	0.01567	0.12518	0.05858	3.583335e+12
MEG	0.00431	0.06565	0.04422	1.393042e+11
MER	0.00326	0.05708	0.03770	9.170000e-03
MPI	0.00273	0.05230	0.03390	2.497000e-02
PGOLD	0.00149	0.03865	0.02818	7.880000e-03
RLC	0.00338	0.05817	0.03978	6.922000e-02
RRHI	0.00131	0.03618	0.02699	6.390000e-03
SMC	0.00137	0.03702	0.02317	5.690000e-03
TEL	0.00178	0.04214	0.03002	4.240000e-03
URC	0.00297	0.05447	0.03742	1.798000e-02

As shown in the table above, the chosen DMD-LSTM model performs well across all other stocks, demonstrating that the model is not overfitted to the training dataset. This score additionally suggests that the model works with non-training data.

The figures below show a 100-day worth of predicted prices versus actual prices to better visualize the performance of the DMD-LSTM model for each stock.

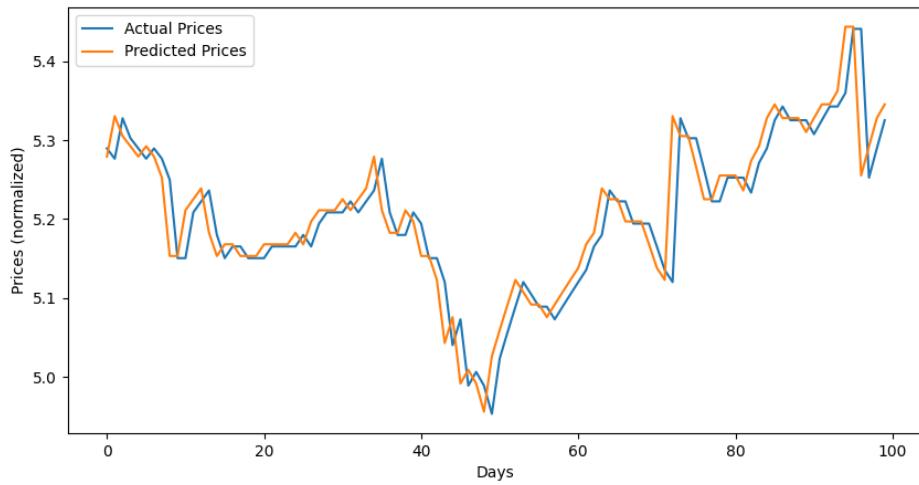


Figure 4.20: Actual vs Predicted Prices on AC for 100 days

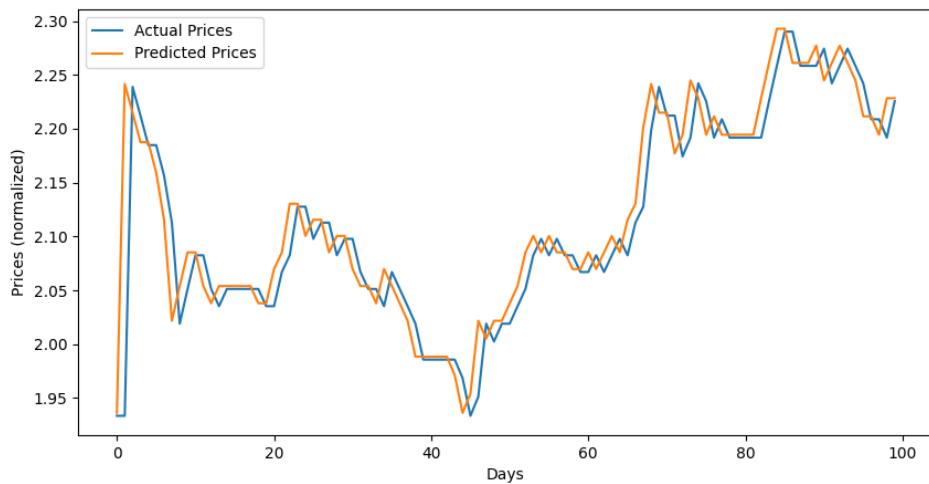


Figure 4.21: Actual vs Predicted Prices for ALI over 100 days

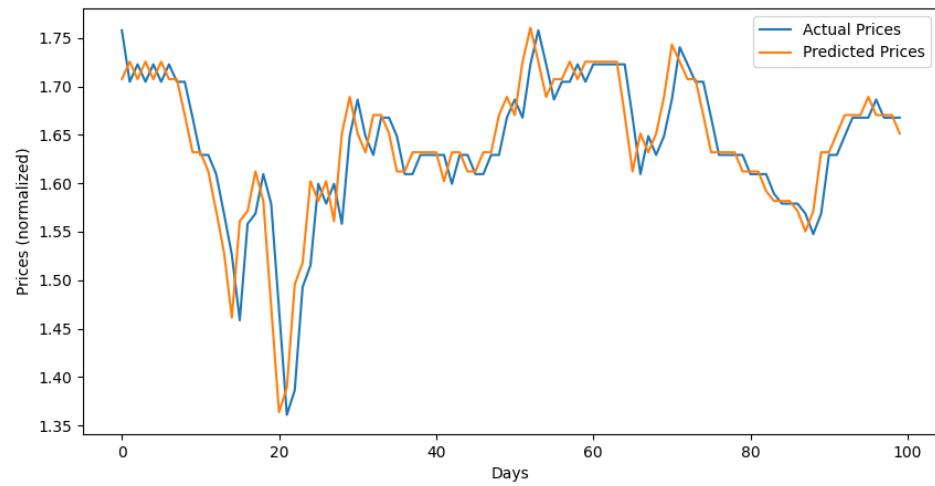


Figure 4.22: Actual vs Predicted Prices for AP over 100 days

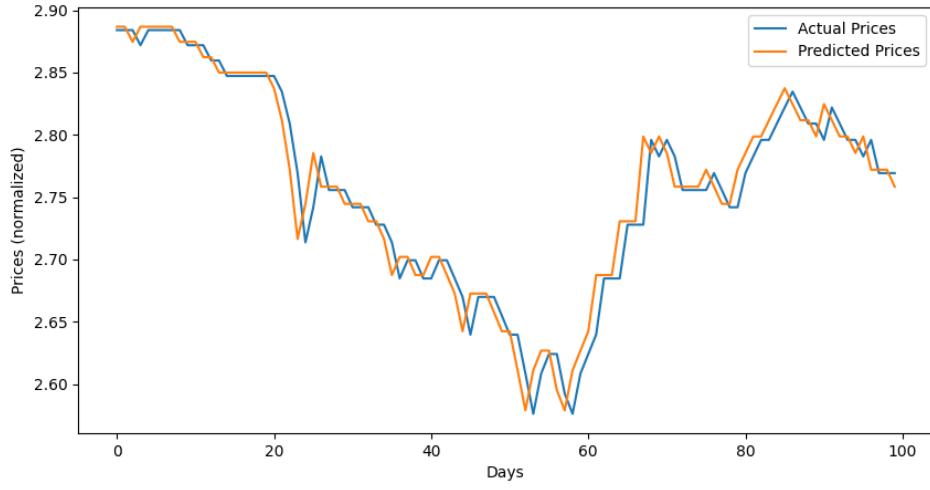


Figure 4.23: Actual vs Predicted Prices for BDO over 100 days

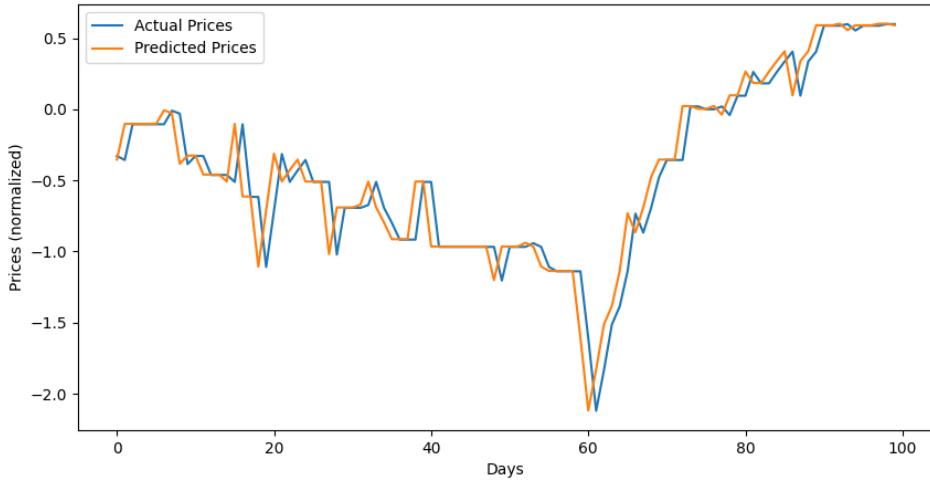


Figure 4.24: Actual vs Predicted Prices for BLOOM over 100 days

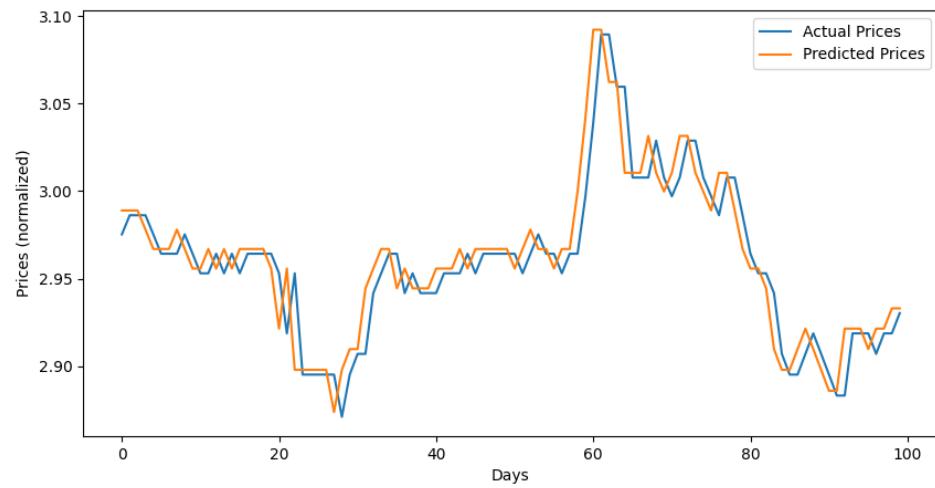


Figure 4.25: Actual vs Predicted Prices for FGEN over 100 days

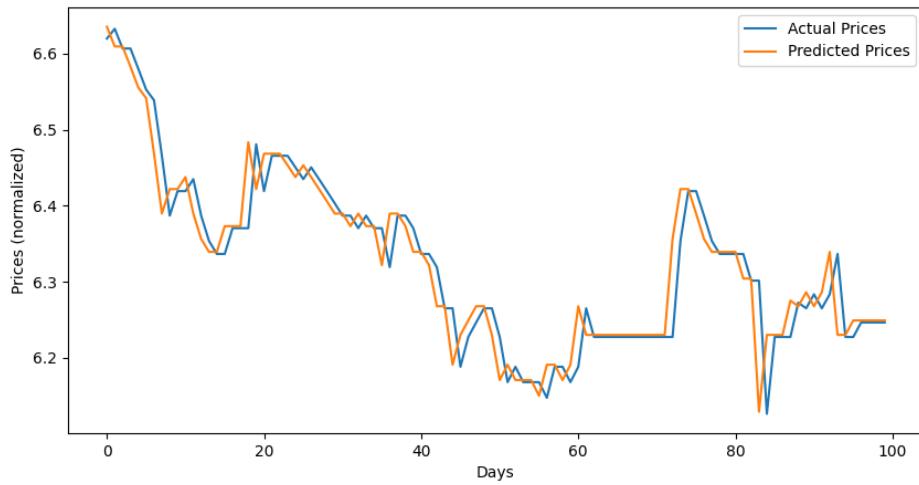


Figure 4.26: Actual vs Predicted Prices for GLO over 100 days

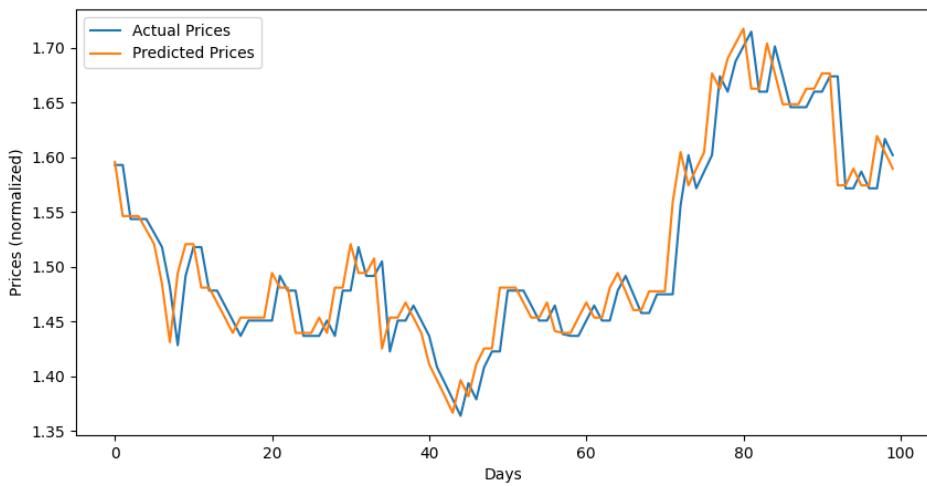


Figure 4.27: Actual vs Predicted Prices for ICT over 100 days

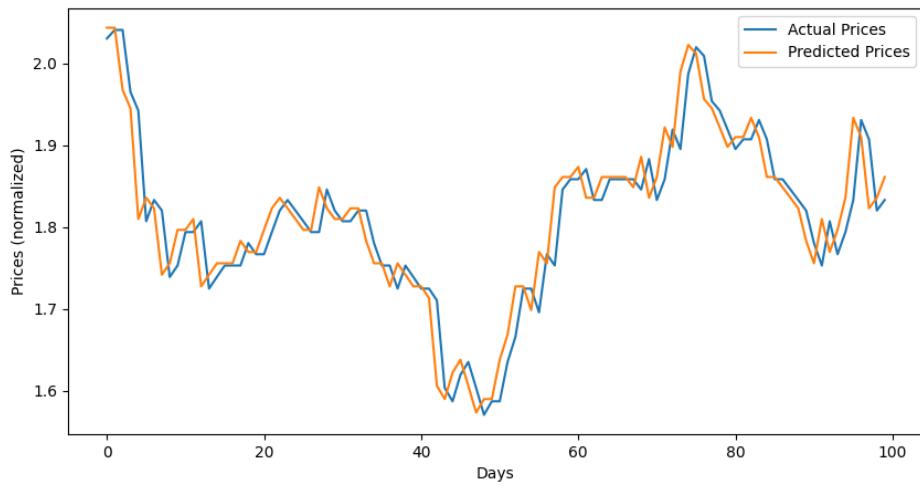


Figure 4.28: Actual vs Predicted Prices on JGS for 100 days

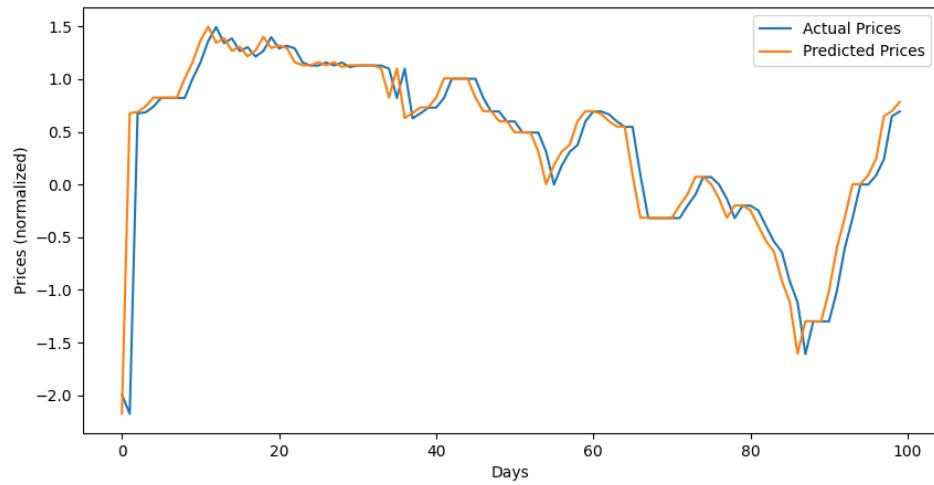


Figure 4.29: Actual vs Predicted Prices on LTG for 100 days

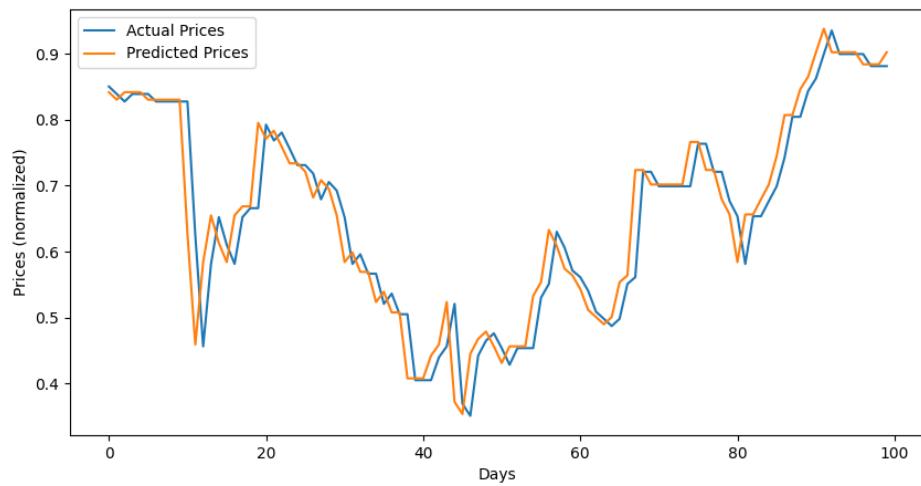


Figure 4.30: Actual vs Predicted Prices on MEG for 100 days

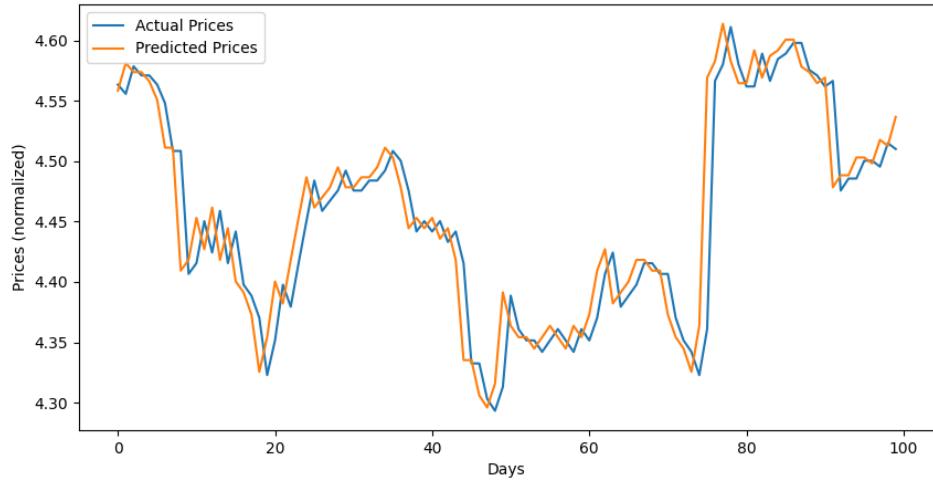


Figure 4.31: Actual vs Predicted Prices on MER for 100 days

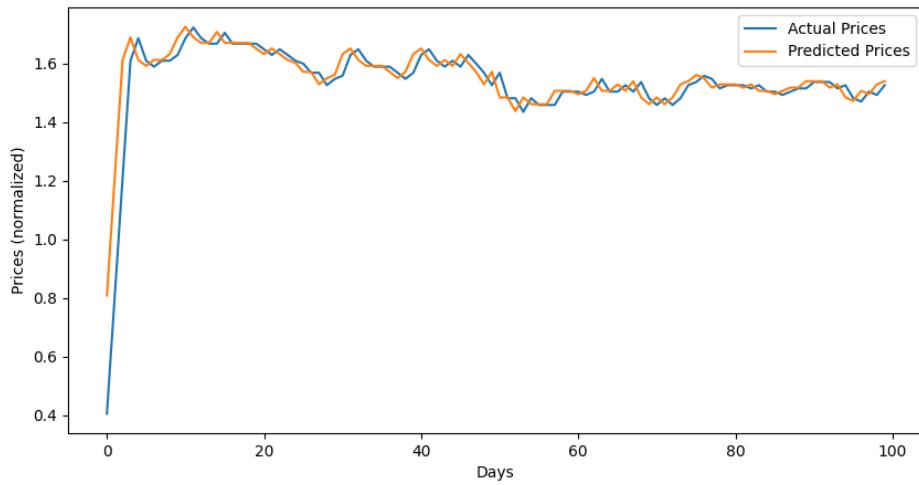


Figure 4.32: Actual vs Predicted Prices on MPI for 100 days

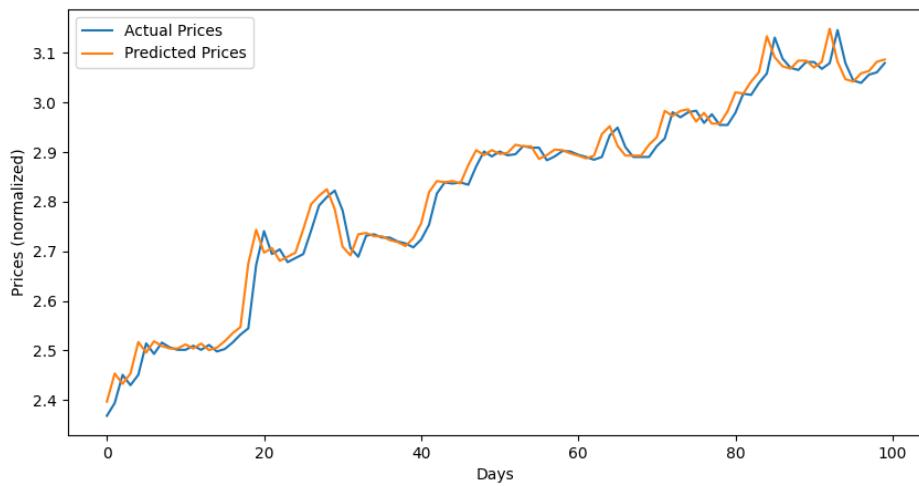


Figure 4.33: Actual vs Predicted Prices on PGOLD for 100 days

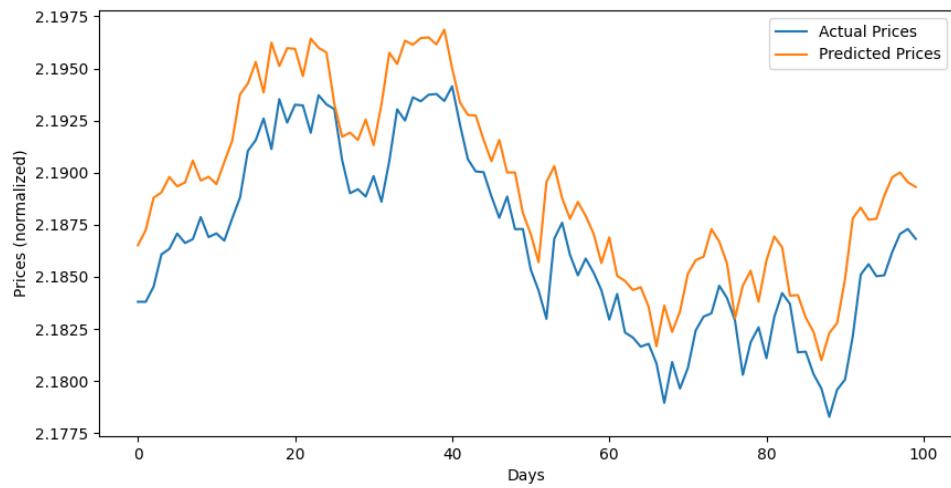


Figure 4.34: Actual vs Predicted Prices on PSEI for 100 days

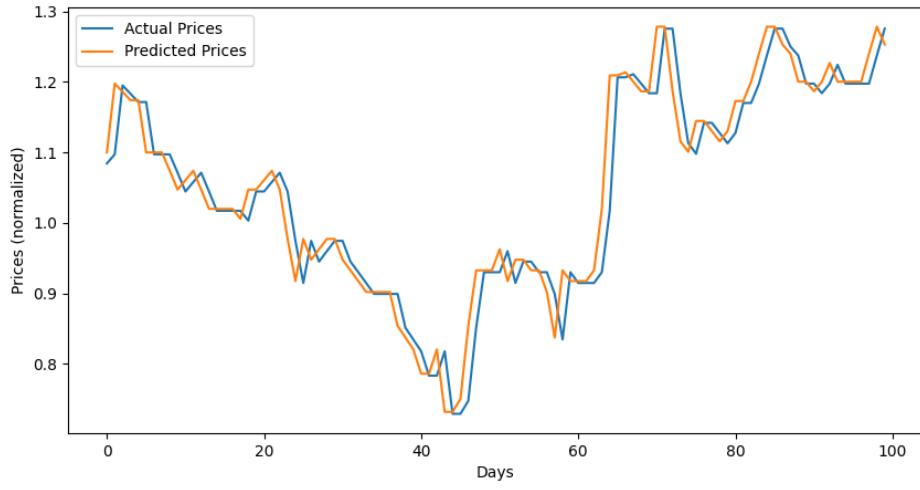


Figure 4.35: Actual vs Predicted Prices on RLC for 100 days

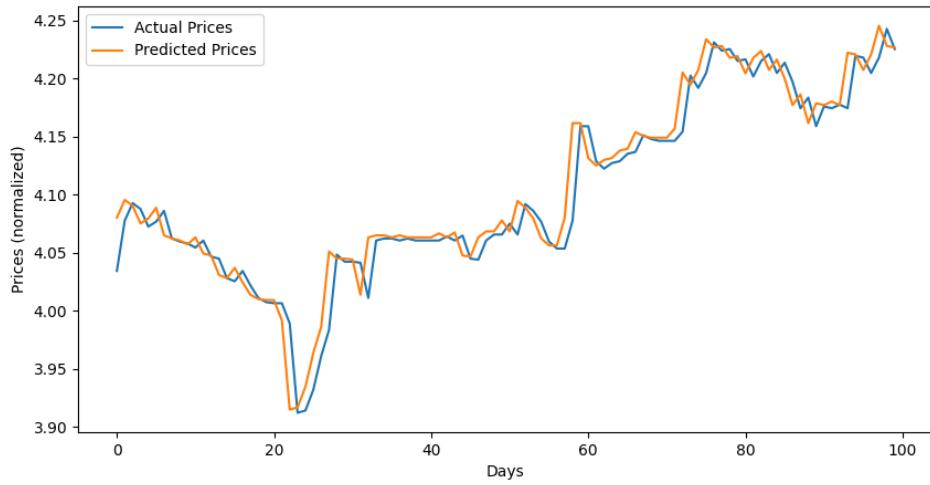


Figure 4.36: Actual vs Predicted Prices on RRHI for 100 days

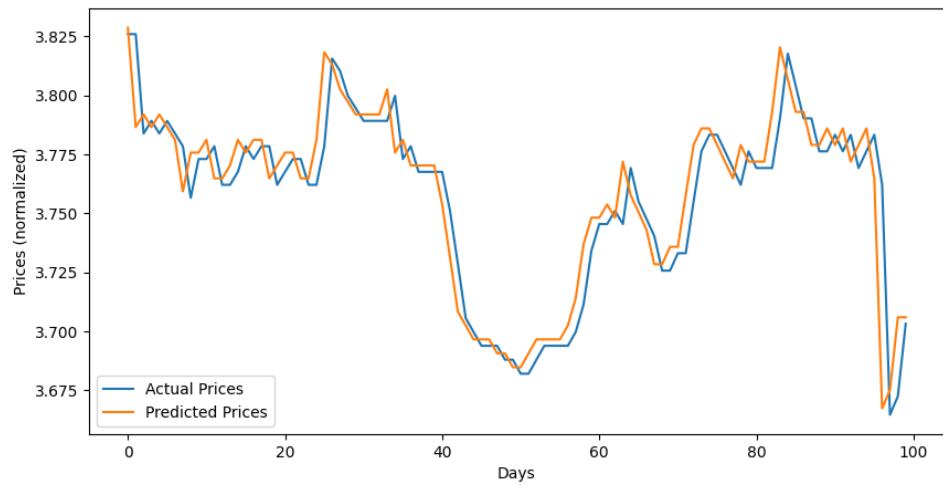


Figure 4.37: Actual vs Predicted Prices on SMC for 100 days

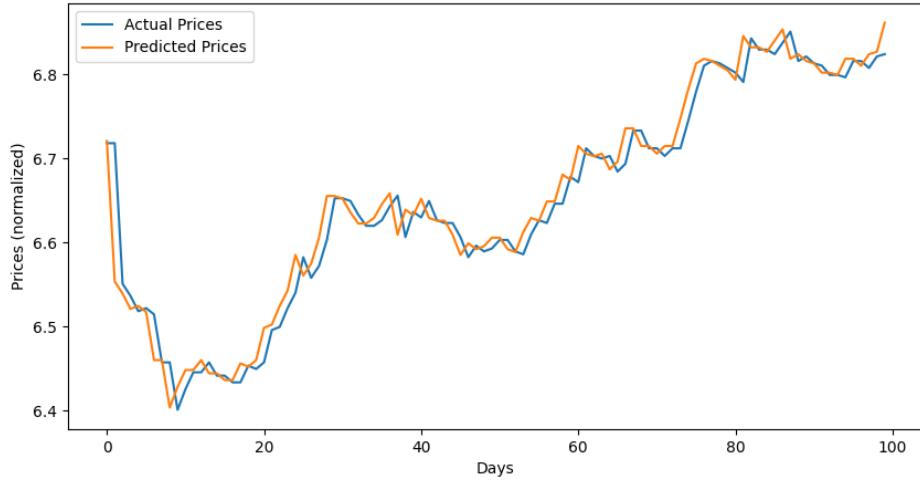


Figure 4.38: Actual vs Predicted Prices on TEL for 100 days

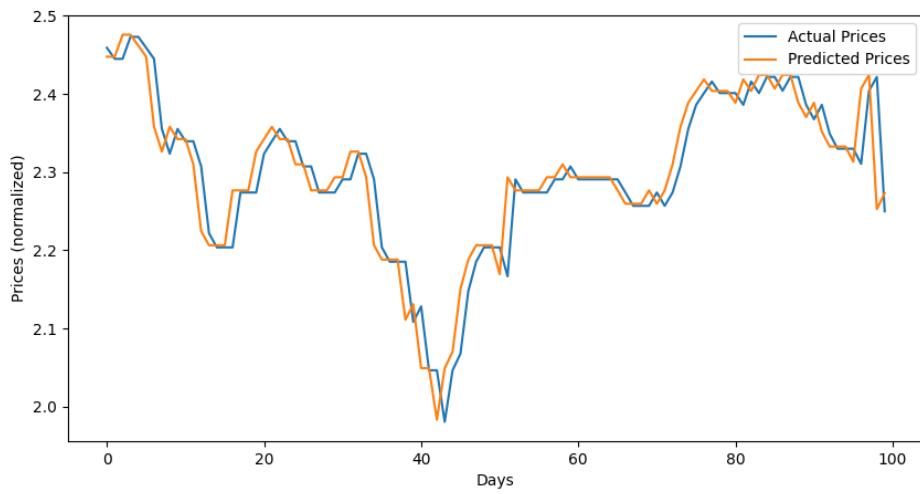


Figure 4.39: Actual vs Predicted Prices on URC for 100 days

The figures above show that the predicted prices follow the actual price trend. In addition, the discrepancy between predicted and actual prices is relatively small, as evidenced by the error metrics scores shown in Table 4.9.

However, the MAPE scores for BLOOM, ICT, JGS, LTG, and MEG range from ten billion to hundred billion. This outlier in the data is, fortunately, just the result of the applied logarithmic normalization, where some of the data in the datasets of the aforementioned stocks are in the negative range, that influence the calculation of the MAPE scores using the scikit-learn library. Because this library handles the calculation of the MAPE scores, there is no way to fix this bug. Moreover, if we take a look at the graphs of the 100 days prediction versus the actual for the aforementioned stocks in Figures 4.24, 4.27, 4.28, 4.29, and 4.30, respectively, it can still be observed that the model performs well on these stocks.

Not to mention that the other error metrics used show the same performance levels across the different stocks when the DMD-LSTM model is utilized. Meanwhile when the data normalization is removed, the MAPE scores for BLOOM, ICT, JGS, LTG, and MEG become 0.068108, 0.037207, 0.039754, 0.057332, and 0.044411 units, respectively.

Another observation from the graphs comparing actual and predicted prices over 100 days is that the predicted values appear to be higher than the actual prices. This indicates the possibility of loss because the model overestimates its prediction.

The successive predictions for the following day and up to ten days were tested using the price data from PSEI in order to make the system's predictions more useful for actual utilization. Table 4.10 shows the MAPE scores for the successive predictions of the DMD-LSTM for each days.

Table 4.10: DMD-LSTM Successive Predictions

Successive Days Predicted	Actual and Predicted Data Ratio	MAPE Score
1	100%	0.00973
2	80%	0.13403
3	60%	0.15782
4	40%	0.15646
5	20%	0.13910
6	0%	0.12494
7	-20%	0.11283
8	-40%	0.10014
9	-60%	0.08914
10	-100%	0.08976

From the table above it must be noted that the ratio values highlighted in red is to demonstrate that, despite the fact that negative ratio values shouldn't exist, doing so simply indicates that the data used to forecast the subsequent price data was overlapping by 2 to 5 times, depending on the ratio, and no longer used any actual data.

Moreover, in the integration of the DMD-LSTM model to the alamSYS, the 5 days successive predictions was utilized. Where it is shown from the Table 4.10 that it still performs well, even if the actual and predicted data ratio is only at 20%. This is also to limit the effect of stock market volatility that might affect the accuracy of the successive predictions of the model.

However, it can also be observed that the MAPE scores for successive days with zero to negative actual and predicted data ratio outperforms the MAPE scores from successive days 2 to 5 as illustrated in Figure 4.40, shown below.

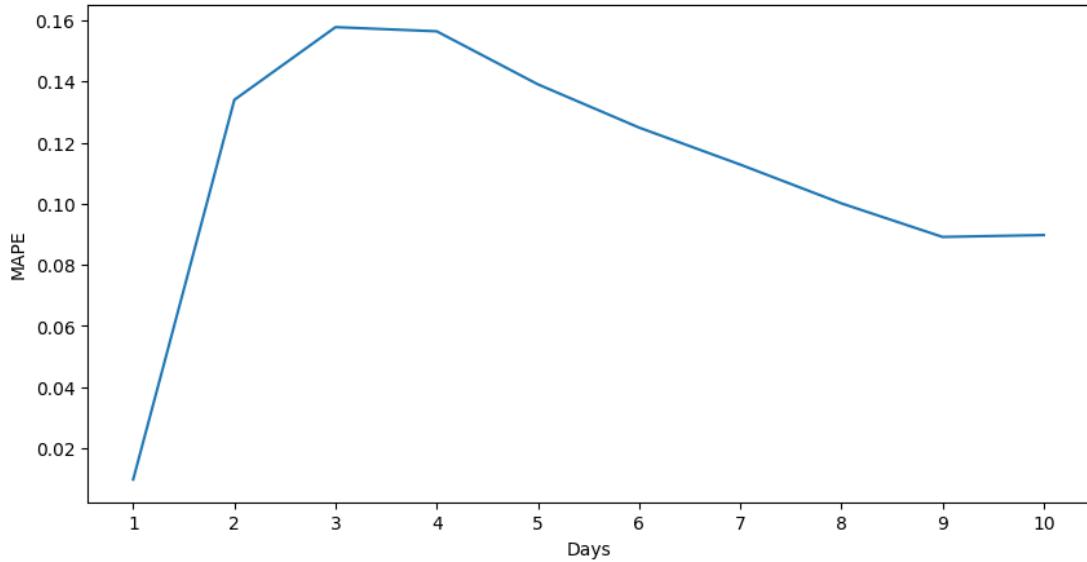


Figure 4.40: MAPE Scores for 1 to 10 (Days) Successive Predictions

Yet, since doing so might result in a poor generalization of data, they were not chosen to be the maximum consecutive days of predictions to be integrated in the alamSYS. As a matter of fact, it could be argued that these data's MAPE scores are overfitted, rendering them unreliable. On the contrary, it might also imply that the model maintains its accuracy for a longer time, even if the majority of the data used are those produced by the model itself. This could be a good thing, and may be attributed to the use of the dynamic modes, as first suggested in the study of Mann and Kutz (2015). In light of these considerations, additional testing is required to establish which of the two claims is true.

Overall, the results from the model training, evaluation, and cross-validation shows that the DMD-LSTM model developed in this special problem performs on par with the other studies that utilizes dynamic modes, as mentioned in Chapter 2 of this paper.

4.4 ALMACD Results and Discussions

The ability to predict consecutive days in the stock market is useless without a trading strategy - which allows risk mitigation and increases the probability of

positive returns over time. Trading strategies, in particular, are based on a pre-defined set of rules and criteria that are used to determine when to buy and sell stocks. (Hayes, 2022).

A variety of algorithmic trading, on the other hand, refers to the use of mathematical and computational techniques to determine the best position to take for a specific set of stocks. Additionally, the possibility of loss due to the influence of human emotion is eliminated. (WallStreetMojo, n.d.).

Whereas, the author used the Arnaud Legoux Moving Average Convergence and Divergence (ALMACD) trading strategy in this special problem and integrated it into the alamSYS as the system's internal trading algorithm. ALAMCD uses predicted prices for the next 5 days, as well as 200 days of actual stock price data, to track the signals and output a simple flag indicating whether to buy or sell that stock at that time.

The compounded expected return after return backtesting is provided for each stock in Table 4.11 using the optimized parameters for the fast and slow ALMA. This was done to validate the potential returns for all stocks, not just the PSEI, from which the best ALMA parameters were derived.

Table 4.11: Optimal Alma Parameters Validation Results

Stock	Compounded Expected Return
PSEI	113966.8500
AC	20893.1914
ALI	1072.1418
AP	690.7100
BDO	2541.9970
BLOOM	495.4600
FGEN	581.0804
GLO	60538.0035
ICT	2815.6103

Table 4.11 continued from previous page

Stock	Compounded Expected Return
JGS	1569.8650
LTG	397.2854
MEG	149.2233
MER	8586.0306
MPI	146.0200
PGOLD	721.2700
RLC	649.4767
RRHI	1050.7000
SMC	2557.0770
TEL	72070.5000
URC	3207.5394

Based on the table of expected returns above, all stocks are expected to return a positive yield over time when these optimal ALMA parameters are used. It is also worth noting that the expected return is calculated for each unit of stock, which means that if we use the expected compounded return value of MPI at PHP 146.02, which appears to be the lowest - the actual return could be at least PHP 146,020, assuming the minimum board lot required for the stocks is 1000 shares (Pesobility, n.d.).

However, despite the high potential returns, investors should proceed with caution for two reasons. First, the expected return is based on historical price data, which may not follow the trend of future price data, potentially rendering the trading algorithm obsolete (Quantified Strategies, 2023). Second, the return calculation does not account for and compensate for the additional fees associated with buying and selling the stock, which can affect the overall actual returns. Moreover, the author investigated the potential for returns by following the alam-SYS predictions, as discussed further in the succeeding section.

4.5 Results and Discussions for the Real World Application of alamSYS

Following the earlier discussions of the system's theoretical performance in terms of DMD-LSTM model performance and optimal ALMACD's expected returns for each stock. It is critical to test the alamSYS's actual performance in recommending which stocks to buy and sell in a simulated real-world application. It should be noted that this is referred to as 'simulated' because no actual money exchange or utilization occurred, but all calculations are based on real-world fees and data.

The results in Table 4.12, in particular, are based on data collected from March 24, 2023 to April 12, 2023 - a total of 10 trading days. A simple set of rules was followed to standardize the results and minimize the bias that could be introduced:

- (a) When the alamSYS indicated that the stock was to be purchased for that day, buy five times the required boardlot.
- (b) Similarly, if the alamSYS indicates that the stock must be sold, it must be sold at five times the required boardlot, regardless of the calculated return.
- (c) The PSEI will be bought and sold on a daily basis for the baseline comparison; and
- (d) All calculations are done with First Metro Securities' trading calculator.

Table 4.12: Return Performance Comparison Between alamSYS and PSEI

	Realized Profit (PHP)	Realized Gain (%)
alamSYS	7,839.75	1.51
PSEI	-22,788.90	-13.810

The above table displays the cumulative profit and gains over a 10-day trading period, demonstrating that the realized profit from following the stock recommendations of the alamSYS yields PHP 30,628.65 more than the baseline cumulative

returns of the PSEI. This is because trading the PSEI resulted in a loss of PHP 22,788.90 (a gain of -13.810%), whereas following the trading suggestions of the alamSYS resulted in a profit of PHP 7,839.75 (a gain of 1.51%).

Given that the PSEI was down by around 2% during this period, as shown in Figure 4.41, the positive performance of the alamSYS affirms the findings from Sections 4.3 and 4.4.



Figure 4.41: PSEI Stock Market Price Trend from March 24 to April 12, 2023

Note: The figure above was generated using the chart tool from Investagrams.com

However, the loss in PSEI may be attributed to the fact that no specific trading strategy was followed, but this is still a fair battle between the two systems - as the use of the alamSYS was based solely on its recommendations. Whereas, additional technical analysis on the stock market may result in higher alamSYS gains.

Figure 4.42 depicts the day-to-day gains of both systems to further visualize the performance of alamSYS's suggestions in comparison to the performance of PSEI.

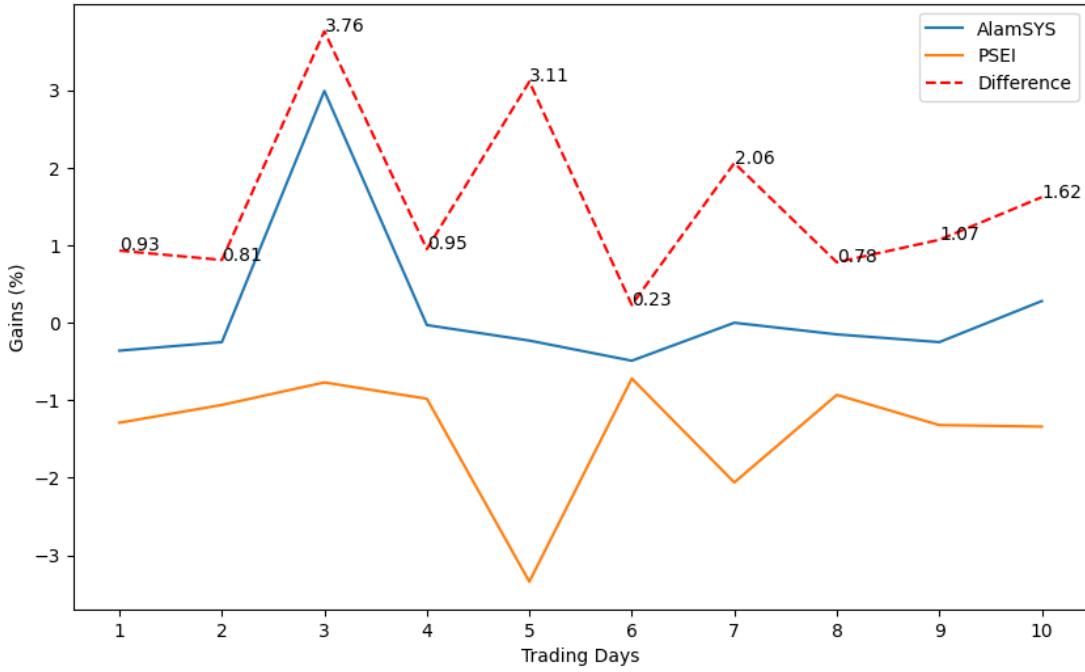


Figure 4.42: Comparison Between the Day-to-day Gains of alamSYS and PSEI

The above figure additionally demonstrates how the alamSYS performs better on a daily basis compared to the PSEI, highlighted by the 'Difference' line. Whereas the alamSYS performed better than the PSEI by 0.93%, 0.81%, 3.76%, 0.95%, 3.11%, 0.23%, 2.06%, 0.78%, 1.07%, and 1.62% over a 10-day period. Also, even though the cumulative return was positive, the alamSYS's day-to-day performance still yielded negative returns. It is worth noting, however, that it was able to reduce the risk of further losses, as can be observed in Figure 4.42.

Chapter 5

Conclusions and Future Work

Chapter 5 presents the findings from this unique problem, as well as recommendations for future research. This chapter specifically covered the following topics:

- (a) Summary of the main findings discussed in Chapter 4 that are relevant to the objectives of this special problem.
- (b) Contributions of the special problem were identified, emphasizing its strengths and limitations.
- (c) Finally, the author provides a roadmap for future works that outlines what needs to be investigated further, additional features that can be added to the system, other external applications or systems that may connect to or be used by the current system, and so on.

5.1 Summary of Findings

The following is a summary of the key findings, in accordance with the objectives outlined in Chapter 2 of this special problem and building on the discussions of the results in Chapter 4.

(a) In connection to the development of the alamSYS:

1. When idle, the alamSYS's average CPU and memory utilization is 0.43% and 525.29 MiB, respectively. Whereas the alamPREPROCESSOR uses the least amount of CPU power, it also uses the most memory due to the scheduling program being active.
2. Both alamAPI and alamPREPROCESSOR use a small amount of CPU power when idle, which is due to the low system requirements of their linux-based base images.
3. The alamPREPROCESSOR was able to process 100 consecutive data collection and processing, indicating a 100 times higher working capacity than expected.
4. The alamPREPROCESSOR's average runtime is 48.30 seconds. Which have been affected by the internet connection speed of 52.98Mbps on average.
5. It also uses an average of 20.03% of the CPU power and 794.29 MiB of memory on load. This means that the alamPREPROCESSOR can be used with low-end devices.
6. When alamPREPROCESSOR is loaded, its CPU and memory utilization are 99.95% and 60.62% higher, respectively, than when it is idle.
7. The average deployed API response time is 1.34 seconds, which is within the acceptable range for avoiding user-perceived interruptions.
8. The delay from the tunneling service accounts for 99.30% of the response time, and the actual response time of the alamAPI is only 0.009 seconds on average; and
9. The average CPU power utilization of alamAPI and alamDB on load is 99.05% and 76.24% higher, respectively, than their idle CPU power utilizations. This allows them to process 71,000 requests in about an hour and 20 minutes. On a different note, onload memory utilization was found to be lower than idle values, but this could simply be due to dips in zero MiB memory utilization over time, as shown in Figure 4.18.

Overall, the findings show that the alamSYS and its three major components (alamPREPROCESSOR, alamAPI, and alamDB) were successfully devel-

oped and integrated. Its idle and on-load performance were guaranteed to be dependable, stable, and efficient.

(b) In connection to the development of DMD-LSTM Model, and ALMACD Trading Algorithm:

1. The DMD-LSTM model with a window size of 5 outperformed the other eight LSTM models that were trained and tested. Where the MSE, RMSE, MAE, and MAPE values are 0.000037, 0.006106, 0.004175, and 0.000001, respectively.
2. Cross-validation of the selected DMD-LSTM model reveals acceptable performance outside of its training data.
3. Up to ten days of consecutive predictions show that the DMD-LSTM model still produces a range of acceptable MAPE scores. However, to avoid the possibility of extrapolation problems affecting the predictions, the alamSYS only integrated up to 5 days of successive predictions. The extrapolation problem on LSTM models were also demonstrated in the studies by Jing, Li, and Peng (2019); and S. Guo, Sun, Pei, and Li (2023).
4. The integration of optimized ALMACD on top of the DMD-LSTM model ensures that the alamSYS recommendations return positive yields for all stocks.
5. The DMD-LSTM and ALMACD applications in alamSYS were tested in real-world trading for a total of 10 trading days, outperforming the PSEI on cumulative return. Furthermore, the alamSYS is capable of mitigating risk and reducing potential losses in its position due to its ability to react quickly on potential price changes attributed to its 5 days in advance predictions and reactive ALMA parameters.

Overall, the combination of DMD-LSTM predictions and ALMACD as a trading algorithm ensures that the alamSYS suggests stock positions that, in theory, yield the highest positive returns while mitigating potential risks associated with market price volatility.

- (c) In connection to the development of the mobile-based test application. The mobile-based application demonstrated the potential of the alamSYS with other external internet-connected devices, such as a smartphone device.

5.2 The Implications of alamSYS to the Society

Based on the findings summary presented in the previous section, the combination of several key components, including the alamSYS, DMD-LSTM model with the optimized ALMACD, and mobile-based test application (alamAPP), has the potential to revolutionize the Philippine financial stock market.

The alamSYS was discovered to be dependable, stable, and efficient, with quick response times and the ability to handle a huge number of queries. While, through cross-validation, the DMD-LSTM model performed within an acceptable error range and proved acceptable performance outside of its training data.

Furthermore, the incorporation of optimized ALMACD confirms that all alamSYS suggestions generate positive returns, limiting potential risks and lowering potential losses. Also, the alamAPP illustrated how the alamSYS might be used with other internet-connected devices. Overall, it positions itself as an accessible stock market investment tool for all Filipinos, with the goal of supporting and empowering investors to make informed decisions and navigate the stock market confidently, even amid difficult market situations.

To put it simply, the sociological implications of a system such as alamSYS have a substantial impact on the potential benefits it could offer to the stock market, individual investors or traders, and the Philippine economy as a whole. The system has the potential to drive economic growth by providing jobs and pushing market advances. The system could boost investor trust and involvement in the stock market by offering credible stock trading advice and limiting potential risks and losses, resulting in a more healthy and prosperous economy.

5.3 Recommendations on Future Works

This section presents a road map for future research based on the insights gained in the preceding sections of this chapter. The roadmap outlines key areas for additional research, potential system enhancements, opportunities for integration with external applications or systems, and other applicable considerations.

(a) In connection to the development of the alamSYS:

1. Automated Model and Trading Algorithm Evaluator and Trainer - Given the possibility of changes in market price trend data over time, the patterns on which the current DMD-LSTM is based may become obsolete. As a result, it may be advantageous to incorporate an additional component that would revalidate the deep learning model as well as the trading algorithm, and if it performs worse than 10% of the current performance, it would automatically rerun the training with the new data until it finds a model and optimal parameters that performs inline or even better than the current one.
2. Alternative System Deployment Methods can be Explored - The tunneling service was found to be responsible for the majority of the delay in the alamSYS response time after deployment. As a result, it may be beneficial to investigate better alternatives, such as deploying the system on a server with a fast and reliable internet connection, though this may necessitate a larger budget. Another low-cost option is to deploy the system on cloud computing platforms such as AWS and Google Cloud.
3. Integration of other Models and Trading Algorithm - Initially, it was planned to integrate multiple combinations of machine learning models and trading algorithms into alamSYS, but due to time constraints, this was not pursued. However, the system was designed with this in mind, and the source code already allows for this feature to be integrated. Future developers, for example, can investigate the integration of ARIMA-based models in alamSYS. This provides users with multiple sources of data-driven information on the market's stocks. Where their decision is not limited to a single piece of information.

4. A Closer Examine on the Stock Suggestions of the System - Because the alamSYS currently only logs its daily predictions, it may be useful for future research to examine the trend of these predictions and try to see the accuracy of this suggestion, as well as potentially add improvements to the system; and
 5. System Maintenance and Additional Features or Improvements to the Current System - The completion of this paper does not indicate the end of the alamSYS's development lifetime; it is expected that the developer, and other developers in the future, will continue to work on improving the system from its initial deployment, using the continuous integration, continuous delivery, and deployment - CI/CD pipeline (RedHat, 2022).
- (b) In connection to the development of DMD-LSTM Model, and ALMACD Trading Algorithm:
1. Solving for the Observed Prediction Offset - The figures comparing predicted and actual price data for each stock revealed a distinct offset pattern. With this in mind, it may be worthwhile to investigate further and add potential additional steps to the predictions to compensate for the offset and improve the model's performance. This could be done during training or as a post-processing technique.
 2. Exploration of Other Machine Learning Models - There are numerous machine learning models that can be investigated for their potential to predict stock market movement. As a result, it is recommended that future developers or researchers be able to devise novel approaches to creating and integrating other models based on the findings and limitations of this special problem.
 3. Redefine the Optimal Fast, and Slow ALMA Parameters - The current optimal ALMA parameters, which do not account for the additional fees for buying and selling stock positions, could possibly not yield to the highest return potentials. As a result, future developers must investigate the possibility of including these fees from the Philippine Stock Exchange in order to better compute the optimal ALMA parameters, and then compare this results on the results of the current optimized ALMA parameters deployed in the alamSYS; and

4. Exploration of Other Trading Strategies and Algorithms - The stock market strategies is not confined on the utilization of moving averages such as Arnaud Legoux Moving Average in creating simple yet effective trading strategies. Hence, it would be beneficial if other trading strategies were explored being integrated alongside the machine learning models to be developed as well.
 - (c) In connection to the development of the mobile-based test application, future developers are encouraged to use other internet-connected or Internet-of-Things (IoT) devices such as smart speakers, smartwatches, smart glasses, and so on. To create an application that makes effective use of the alamSYS. Making certain that it is widely used.

References

- Abbad, J. M., Fardousi, B., & Abbad, M. M. (2014). Advantages of using technical analysis to predict future prices on the amman stock exchange. *International Journal of Biometrics*, 9, 1.
- Allwright, S. (2022). *What is a good mape score?* Retrieved April 15, 2023, from <https://stephenallwright.com/good-mape-score/>
- and, D. G. S. (2022, jan). Machine learning models in stock market prediction. *International Journal of Innovative Technology and Exploring Engineering*, 11(3), 18–28. Retrieved from <https://doi.org/10.35940%2Fijitee.c9733.0111322> doi: 10.35940/ijitee.c9733.0111322
- Andrew. (2019). *You should (usually) log transform your positive data.* Retrieved April 15, 2023, from <https://statmodeling.stat.columbia.edu/2019/08/21/you-should-usually-log-transform-your-positive-data/>
- Ausloos, M., & Ivanova, K. (2002, may). Mechanistic approach to generalized technical analysis of share prices and stock market indices. *The European Physical Journal B - Condensed Matter*, 27(2), 177–187. Retrieved from <https://doi.org/10.1140/epjb/e20020144> doi: 10.1140/epjb/e20020144
- Bae, K. H., & Kang, J. (2017). *Does the stock market benefit the economy?* Retrieved from <https://www.efmaefm.org/OEFMSYMPOSIUM/2017/papers/Does%20the%20Stock%20Market%20Benefit%20the%20Economy%20-%20updated.pdf>
- Baeldung. (2022). *Normalization inputs for an artificial neural network.* Retrieved April 15, 2023, from <https://www.baeldung.com/cs/normalizing-inputs-artificial-neural-network>
- Balaba, J. M. (2017). *Does the stock market drive the philippine economy?* Retrieved from <https://www.dlsu.edu.ph/wp-content/uploads/>

- <pdf/conferences/research-congress-proceedings/2017/RVREBM/RVREBM-I-003.pdf>
- Benignos, M. G., & Navarro, M. M. (2020). Evaluation of online stock trading platforms for filipino investors/traders in the philippine market amidst the covid-19 pandemic. In (p. 2988-2994). Retrieved from <https://www.ieomsociety.org/detroit2020/papers/612.pdf>
- Budiharto, W. (2021). Data science approach to stock prices forecasting in indonesia during covid-19 using long short-term memory (LSTM). *Journal of Big Data*, 8(1). Retrieved from <https://doi.org/10.1186/s40537-021-00430-0> doi: 10.1186/s40537-021-00430-0
- Bureau of the Treasury Bangko Sentral ng Pilipinas. (n.d.). *19 selected domestic interest rates weighted average in percent per annum for periods indicated*. Retrieved October 06, 2022, from www.bsp.gov.ph/Statistics/Financial%20System%20Accounts/tab19_dir.aspx
- Campbell, G. (2021, 5). *Does the stock market reflect the economy?* Retrieved October 07, 2022, from <https://www.economicsobservatory.com/does-the-stock-market-reflect-the-economy>
- Carey, T. W. (2023). *Top technical analysis tools for traders*. Retrieved May 22, 2023, from <https://www.investopedia.com/articles/active-trading/121014/best-technical-analysis-trading-software.asp>
- Chhajer, P., Shah, M., & Kshirsagar, A. (2022). The applications of artificial neural networks, support vector machines, and long-short term memory for stock market prediction. *Decision Analytics Journal*, 2, 100015. Retrieved from <https://www.sciencedirect.com/science/article/pii/S2772662221000102> doi: <https://doi.org/10.1016/j.dajour.2021.100015>
- Concoda. (2020). *Why artificial intelligence will never beat the stock market*. Retrieved May 21, 2023, from <https://www.tradersmagazine.com/departments/equities/why-artificial-intelligence-will-never-beat-the-stock-market/>
- Cui, L. X., & Long, W. (2016, 11). Trading strategy based on dynamic mode decomposition: Tested in chinese stock market. *Physica A: Statistical Mechanics and its Applications*, 461, 498-508. doi: 10.1016/j.physa.2016.06.046
- Dash, R., & Dash, P. K. (2016, 3). A hybrid stock trading framework integrating technical analysis with machine learning techniques. *The Journal of Finance and Data Science*, 2, 42-57. doi: 10.1016/j.jfds.2016.03.002

- DayTradingz. (2023). *10 best stock analysis software apps and tools of 2023*. Retrieved May 22, 2023, from <https://daytradingz.com/best-stock-analysis-software/>
- EdwardJones. (n.d.). *Why invest in stocks?* Retrieved September 19, 2022, from <https://www.edwardjones.com/us-en/market-news-insights/guidance-perspective/benefits-investing-stock>
- Fraz, T. R., Fatima, S., & Uddin, M. (2022). Modeling and forecasting stock market volatility of cpec founding countries: Using nonlinear time series and machine learning models. *Journal of Independent Studies and Research-Management, Social Sciences and Economics*.
- Glen, S. (n.d.-a). *Mean squared error: Definition and example*. Retrieved April 15, 2023, from <https://www.statisticshowto.com/probability-and-statistics/statistics-definitions/mean-squared-error/>
- Glen, S. (n.d.-b). *Rmse: Root mean square error*. Retrieved April 15, 2023, from <https://www.statisticshowto.com/probability-and-statistics/regression-analysis/rmse-root-mean-square-error/>
- Guo, S., Sun, N., Pei, Y., & Li, Q. (2023). 3d-unet-lstm: A deep learning-based radar echo extrapolation model for convective nowcasting. *Remote Sensing*, 15(6). Retrieved from <https://www.mdpi.com/2072-4292/15/6/1529> doi: 10.3390/rs15061529
- Guo, Y. (2022). *Stock price prediction using machine learning*. Retrieved from <https://www.diva-portal.org/smash/get/diva2:1672304/FULLTEXT01.pdf>
- Hall, M. (2022, 9). *How the stock market affects gdp*. Retrieved October 07, 2022, from <https://www.investopedia.com/ask/answers/033015/how-does-stock-market-affect-gross-domestic-product-gdp.asp>
- Hayes, A. (2022). *What is a trading strategy? how to develop one*. Retrieved April 22, 2023, from <https://www.investopedia.com/terms/t/trading-strategy.asp>
- Hayes, A. (2023). *Volatility: Meaning in finance and how it works with stocks*. Retrieved April 26, 2023, from <https://www.investopedia.com/terms/v/volatility.asp>
- Investagrams. (2023). *Your guide to the financial markets*. Retrieved May 22, 2023, from <https://www.investagrams.com/>
- JavaTPoint. (n.d.). *Agile model*. Retrieved April 16, 2023, from <https://www>

- .javatpoint.com/software-engineering-agile-model
- Jing, J., Li, Q., & Peng, X. (2019). Mlc-lstm: Exploiting the spatiotemporal correlation between multi-level weather radar echoes for echo sequence extrapolation. *Sensors*, 19(18). Retrieved from <https://www.mdpi.com/1424-8220/19/18/3988> doi: 10.3390/s19183988
- Joseph, R. (2018). *Grid search for model tuning*. Retrieved April 17, 2023, from <https://towardsdatascience.com/grid-search-for-model-tuning-3319b259367e>
- Juviler, J. (2022). *Api response time, explained in 1000 words or less*. Retrieved April 22, 2023, from <https://blog.hubspot.com/website/api-response-time>
- Kenton, W. (2023). *Understanding value at risk (var) and how it's computed*. Retrieved April 26, 2023, from <https://www.investopedia.com/terms/v/var.asp>
- Kim, T. (2022, 7). *Alternative data for hedge funds: The competitive edge to investing*. Retrieved November 04, 2022, from <https://www.similarweb.com/corp/blog/investor/asset-research/hedge-funds-use-alternative-data>
- Kumbure, M. M., Lohrmann, C., Luukka, P., & Porras, J. (2022, 7). Machine learning techniques and data for stock market forecasting: A literature review. *Expert Systems with Applications*, 197, 116659. doi: 10.1016/j.eswa.2022.116659
- Lu, Z., Tao, R., & Zhao, Y. (2022). Machine learning in stock market movement intelligent forecast system. *2022 International Conference on Data Analytics, Computing and Artificial Intelligence (ICDACAII)*, 402-406.
- Mann, J., & Kutz, J. N. (2015, 8). Dynamic mode decomposition for financial trading strategies.
- Mellema, G. R. (2020). Improved active sonar tracking in clutter using integrated feature data. *IEEE Journal of Oceanic Engineering*, 45(1), 304-318. doi: 10.1109/JOE.2018.2870234
- Milne, A. (2021). *The agile development methodology explained*. Retrieved April 16, 2023, from <https://www.netsolutions.com/insights/agile-development-methodology/>
- Mitchell, C. (2023). *Drawdown: What it is, risks and examples*. Retrieved April 26, 2023, from <https://www.investopedia.com/terms/d/drawdown.asp>

- National Economic and Development Authority. (2023). *Gov't to continue timely measures to protect purchasing power as inflation eases further in april - neda*. Retrieved May 11, 2023, from <https://neda.gov.ph/govt-to-continue-timely-measures-to-protect-purchasing-power-as-inflation-eases-further-in-april-neda>
- Obthong, M., Tantisantiwong, N., Jeamwatthanachai, W., & Wills, G. (2020). A survey on machine learning for stock price prediction: Algorithms and techniques. In (p. 63-71). SCITEPRESS - Science and Technology Publications. doi: 10.5220/0009340700630071
- Pesobility. (n.d.). *Board lot of philippine stock exchange*. Retrieved April 22, 2023, from <https://www.pesobility.com/ref/boardlot>
- Prasad, A., & Seetharaman, A. (2021). Importance of machine learning in making investment decision in stock market. *Vikalpa: The Journal for Decision Makers*, 46, 209 - 222.
- Priatmojo, Y. S., & Sari, S. P. (2022). Technical analysis to determine signal lines in sharia stock trading: Case in indonesia sharia stock market. *International Journal of Scientific and Research Publications (IJSRP)*.
- Qamruzzaman, M., & Wei, J. (2019). Do financial inclusion, stock market development attract foreign capital flows in developing economy: a panel data investigation. *Quantitative Finance and Economics*, 3(1), 88-108. Retrieved from <https://www.aimspress.com/article/doi/10.3934/QFE.2019.1.88> doi: 10.3934/QFE.2019.1.88
- Quantified Strategies. (2023). *Why trading strategies are not working (how to know when trading systems stopped performing?)*. Retrieved April 22, 2023, from <https://www.quantifiedstrategies.com/why-trading-strategies-are-not-working/>
- Rea, L. (2020, 8). *Predicting future stock market trends with python and machine learning*. Retrieved October 05, 2022, from <https://towardsdatascience.com/predicting-future-stock-market-trends-with-python-machine-learning-2bf3f1633b3c>
- RedHat. (2022). *What is ci/cd?* Retrieved April 23, 2023, from <https://www.redhat.com/en/topics/devops/what-is-ci-cd>
- Royal Bank of Canada Direct Investing Inc. (n.d.). Key benefits of investing in stocks. *Royal Bank of Canada Direct Investing Inc.* Retrieved from www6.royalbank.com/en/di/hubs/investing-academy/

- chapter/key-benefits-of-investing-in-stocks/jv7atg13/jv7atg1j
- Sarangi, P. K., Mohanty, J., Mohapatra, S. K., & Sahu, P. (2023). Computing stock market price behavior using machine learning approach. *2023 6th International Conference on Information Systems and Computer Networks (ISCON)*, 1-5.
- Savaş, M. C. (2017). *Algorithmic trading strategies using dynamic mode decomposition: Applied to turkish stock market*. Retrieved from <https://etd.lib.metu.edu.tr/upload/12621107/index.pdf>
- Scherzinger, S., Roennau, A., & Dillmann, R. (2019). Contact skill imitation learning for robot-independent assembly programming. *CoRR, abs/1908.06272*. Retrieved from <http://arxiv.org/abs/1908.06272>
- Secret Data Scientist. (2023). *What is mae (mean absolute error)?* Retrieved April 15, 2023, from <https://secretdatascientist.com/mae-mean-absolute-error/>
- S.Gopal Krishna Patro, K. K. s. (2015). Normalization: A preprocessing stage. *International Journal of Advanced Research in Science, Engineering and Technology, 205*. Retrieved from 10.17148/IARJSET.2015.2305
- Shaheen, M., & Arshad, M. (2020). Use of machine learning in stock market prediction. *European Journal of Technology*.
- Soni, P., Tewari, Y., & Krishnan, D. (2022, 1). Machine learning approaches in stock price prediction: A systematic review. *Journal of Physics: Conference Series, 2161*, 012065. doi: 10.1088/1742-6596/2161/1/012065
- Strader, T. J., Rozycski, J. J., Root, T. H., & Huang, Y.-H. (2020). *Machine learning stock market prediction studies: Review and research directions* (Vol. 28). Retrieved from <https://scholarworks.lib.csusb.edu/jitim/vol28/iss4/3>
- Tangsripairoj, S., Sutanaphanit, P., Treenitikul, S., Wongaod, W., & Maneeintr, K. (2020). Stockify: A web application of fundamental stock analysis for new investors. *2020 - 5th International Conference on Information Technology (InCIT)*, 76-81.
- Trade Brains. (2022, 7). *How does the stock market affect the economy?* Retrieved October 07, 2022, from <https://tradebrains.in/how-stock-market-affect-the-economy/>
- Tuychiev, B. (2021). *How to differentiate between scaling, normalization, and log transformations.* Retrieved April 15, 2023, from

- <https://towardsdatascience.com/how-to-differentiate-between-scaling-normalization-and-log-transformations-69873d365a94>
- U.S. Securities and Exchange Commission. (n.d.). *What are stocks?* Retrieved October 07, 2022, from www.investor.gov/introduction-investing/investing-basics/investment-products/stocks
- Visual Paradigm. (n.d.). *What is use case diagram?* Retrieved April 26, 2023, from <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/>
- VisualParadigm. (n.d.). *Gane-sarson data flow diagram tutorial.* Retrieved November 15, 2022, from <https://online.visual-paradigm.com/knowledge/software-design/gane-sarson-dfd-tutorial/>
- WallStreetMojo. (n.d.). *Algorithmic trading.* Retrieved April 22, 2023, from <https://www.wallstreetmojo.com/algorithmic-trading/>
- Zhang, X. (2022). Application of business analysis in stock market forecasting -machine learning in stock market practice. *BCP Business & Management.*
- Zou, J., Zhao, Q., Jiao, Y., Cao, H., Liu, Y., Yan, Q., ... Shi, J. Q. (2023). *Stock market prediction via deep learning techniques: A survey.*
- Zwetsloot, R. (2019). *Raspberry pi 4 specs and benchmarks.* Retrieved April 22, 2023, from <https://magpi.raspberrypi.com/articles/raspberry-pi-4-specs-benchmarks>

Appendix A

Source Code Repository

Follow this link to access the code repository for this special problem: https://github.com/GravitonXD/OLARTE_SP. It should be noted that this repository also contains all of the miscellaneous files, such as the.tex files used to create this paper, test codes, and so on. The following are the primary directories for this project:

- (a) alamSYS - contains the source code for alamSYS as well as the docker and docker-compose files.
- (b) DeepLearningModel - contains all of the Python notebooks used in the development of this special problem for training, testing, and cross-validation of the models, trading algorithm, and other calculations.
- (c) alamAPP - This directory contains the source code for the mobile-based test application.

Future developers who want to expand the functionality or dig deeper into this special problem can use the following repository links:

- (a) alamSYS - <https://github.com/GravitonXD/alamSYS>
- (b) DMD-LSTM - https://github.com/GravitonXD/alamSYS_DMD-LSTM
- (c) alamAPP - <https://github.com/GravitonXD/alamAPP>

Appendix B

Raw Data Figures

B.1 Exploratory Stocks Data Graphs

The figures below depict combined line graphs of the opening, high, low, and closing prices of each stock in the alamSYS. These figures also demonstrate why closing prices were chosen as the primary training feature of the models developed for this particular problem. Aside from being the most important price target in most trading and investing strategies, closing prices do not differ significantly from the other price metrics.

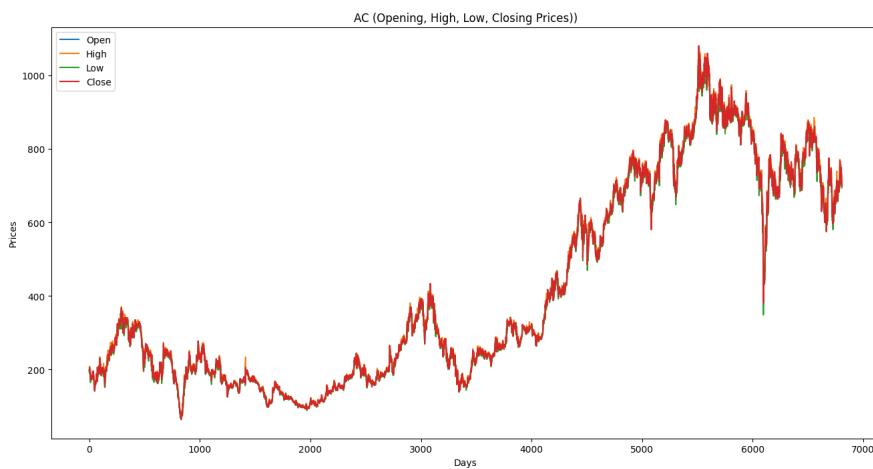


Figure B.1: Opening, High, Low, and Closing Prices on AC

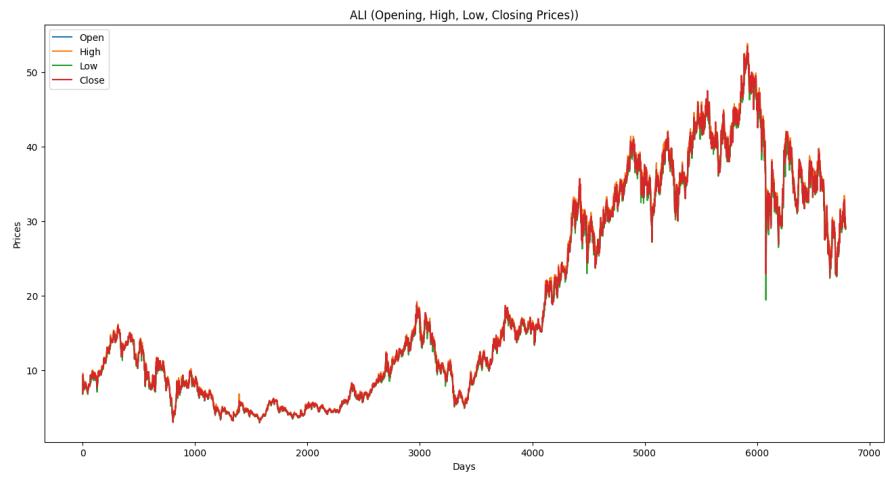


Figure B.2: Opening, High, Low, and Closing Prices for ALI

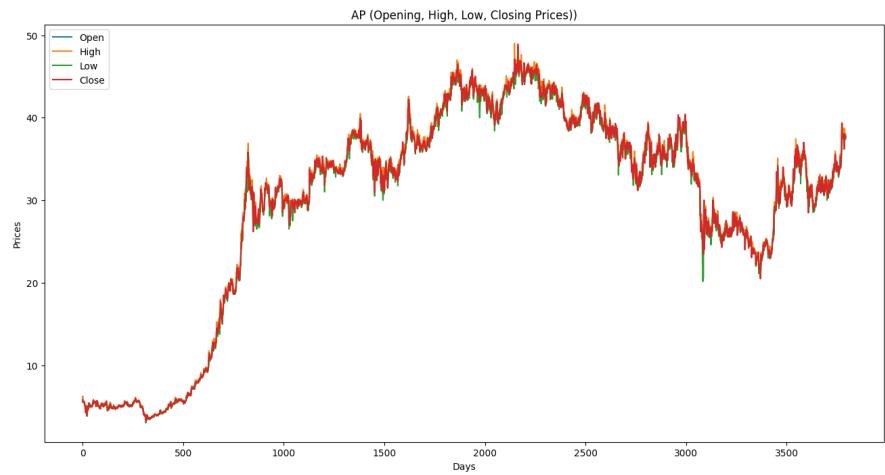


Figure B.3: Opening, High, Low, and Closing Prices for AP

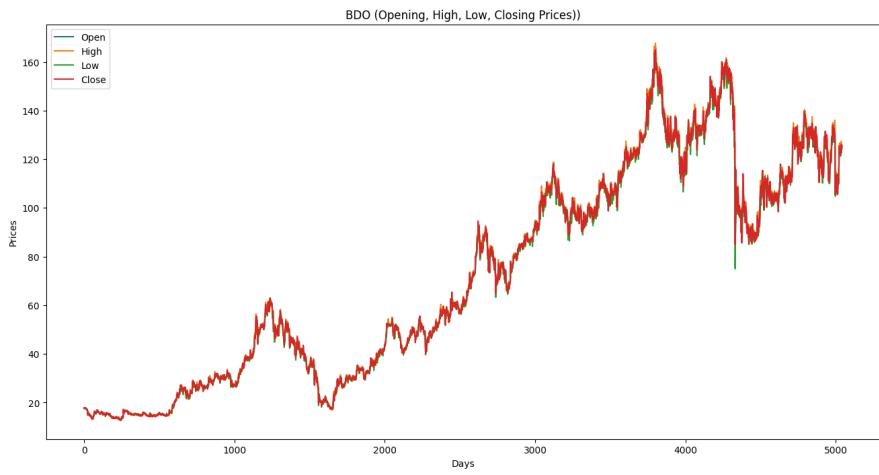


Figure B.4: Opening, High, Low, and Closing Prices for BDO

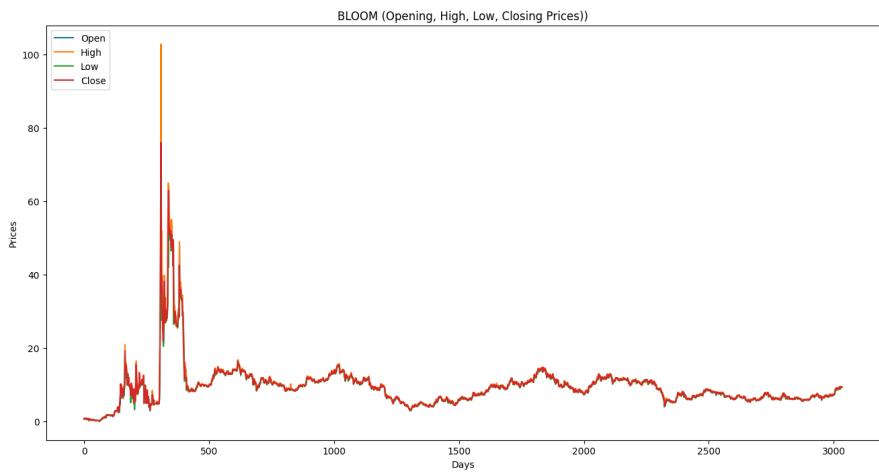


Figure B.5: Opening, High, Low, and Closing Prices for BLOOM



Figure B.6: Opening, High, Low, and Closing Prices for FGEN

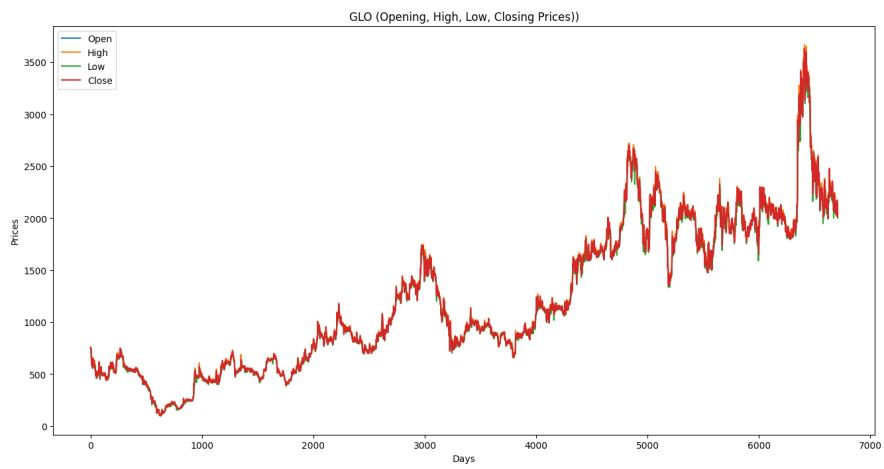


Figure B.7: Opening, High, Low, and Closing Prices for GLO

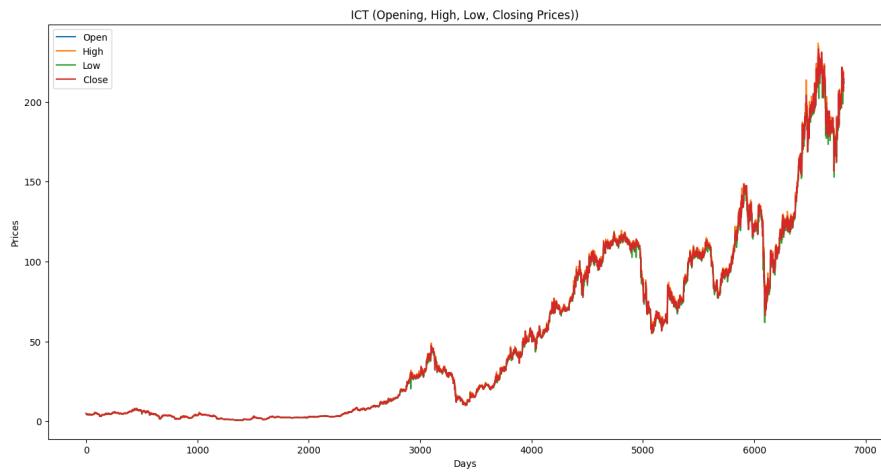


Figure B.8: Opening, High, Low, and Closing Prices for ICT

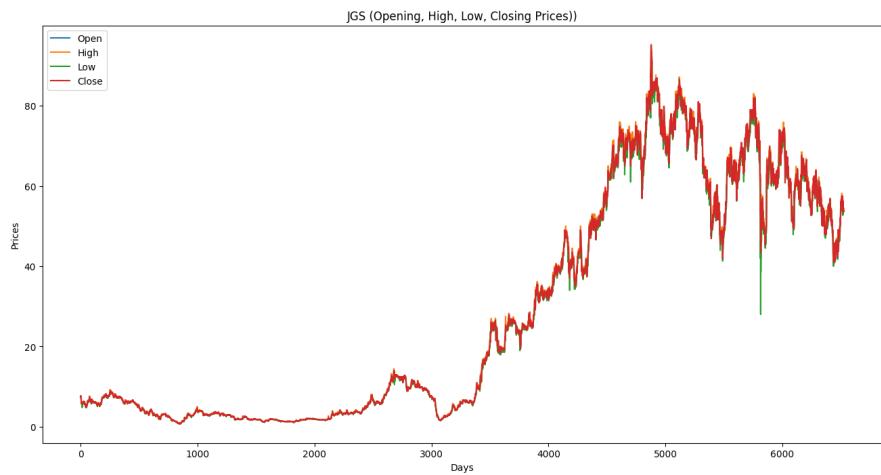


Figure B.9: Opening, High, Low, and Closing Prices on JGS

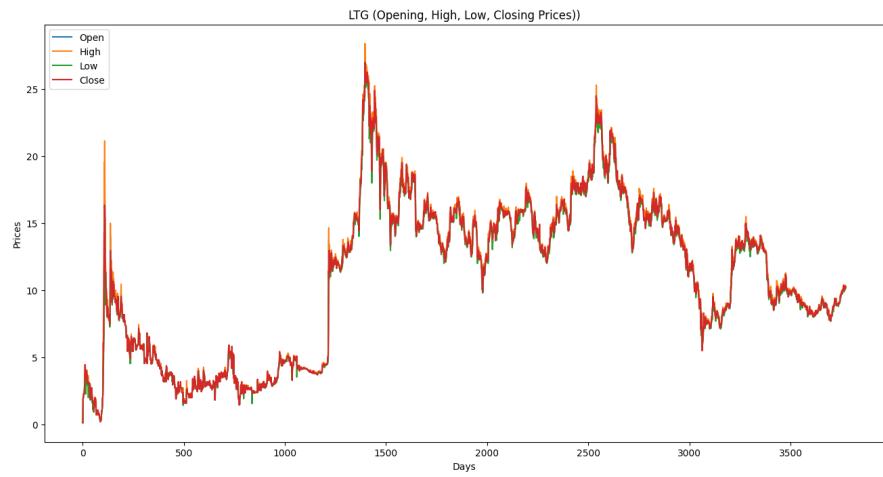


Figure B.10: Opening, High, Low, and Closing Prices on LTG

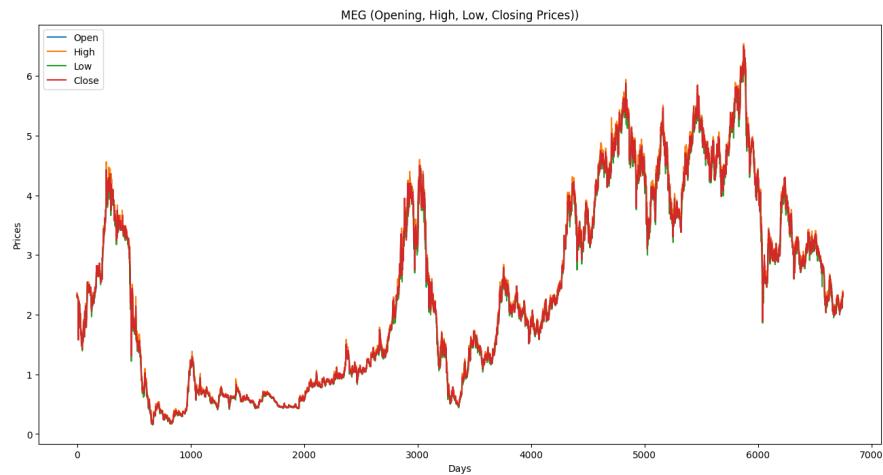


Figure B.11: Opening, High, Low, and Closing Prices on MEG

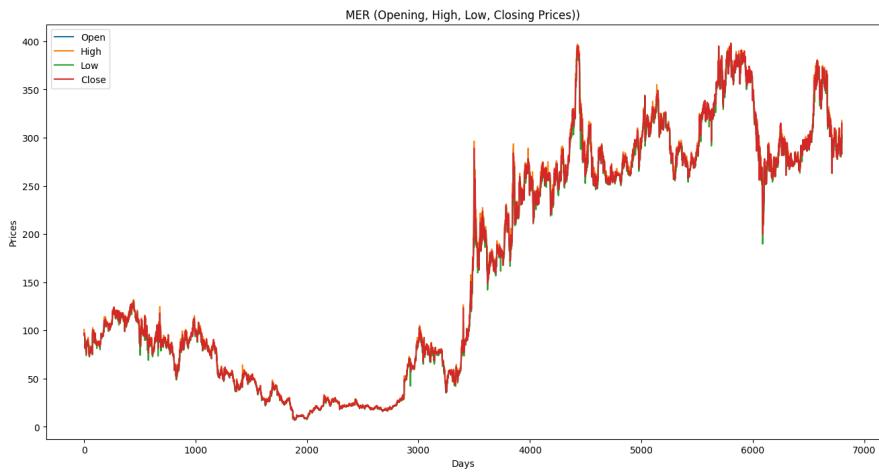


Figure B.12: Opening, High, Low, and Closing Prices on MER

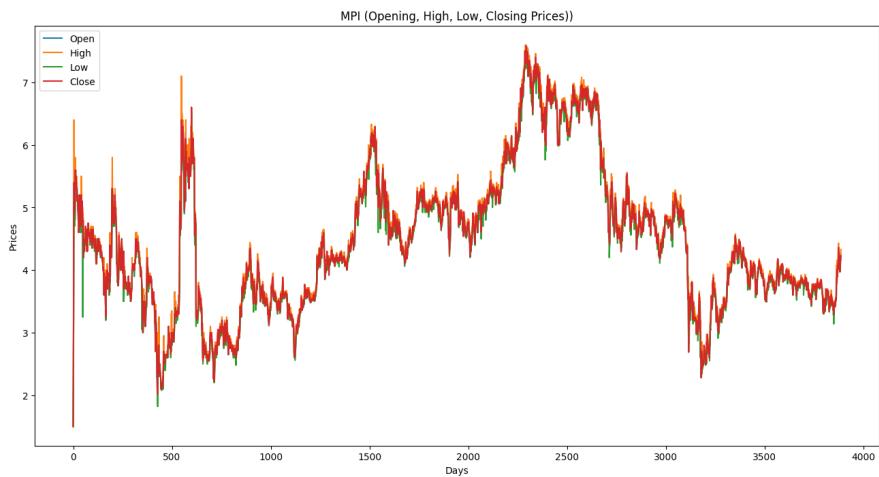


Figure B.13: Opening, High, Low, and Closing Prices on MPI



Figure B.14: Opening, High, Low, and Closing Prices on PGOLD

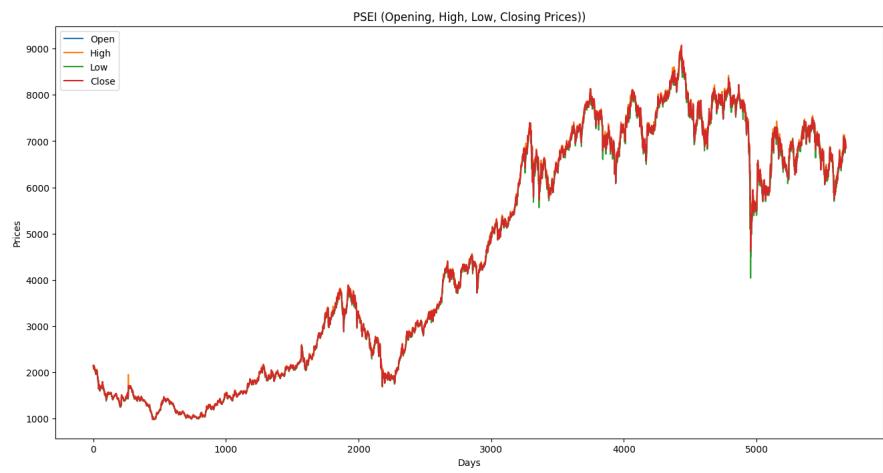


Figure B.15: Opening, High, Low, and Closing Prices on PSEI

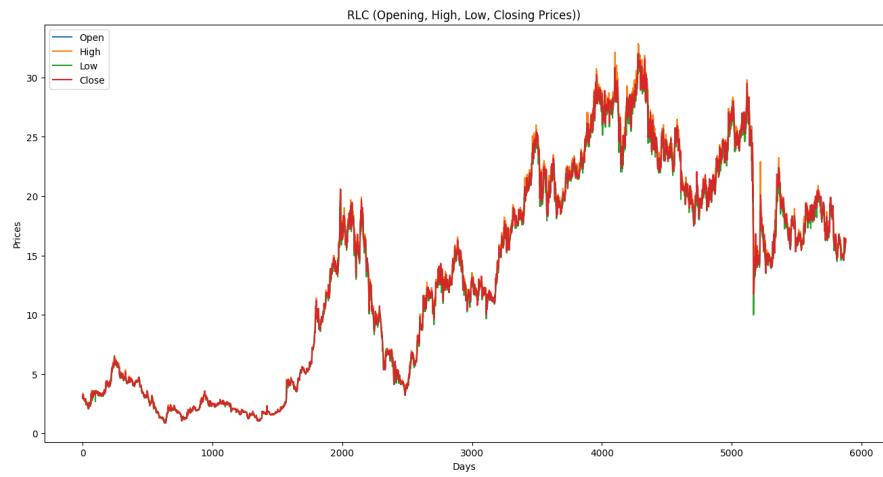


Figure B.16: Opening, High, Low, and Closing Prices on RLC

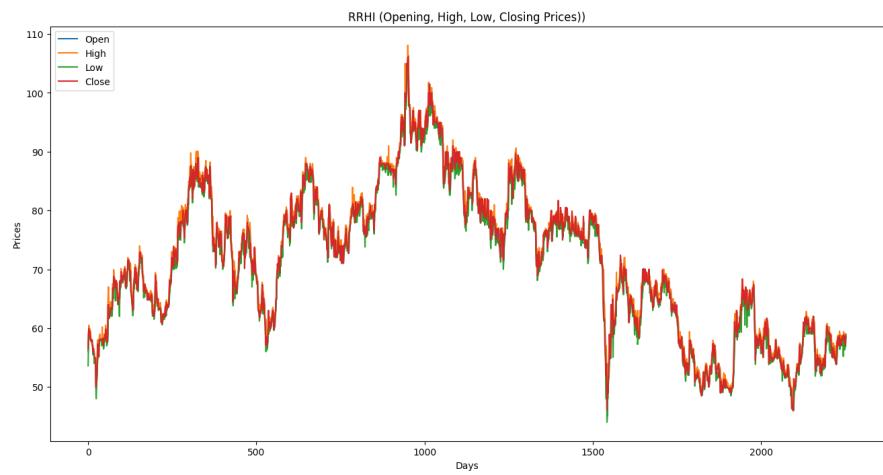


Figure B.17: Opening, High, Low, and Closing Prices on RRHI

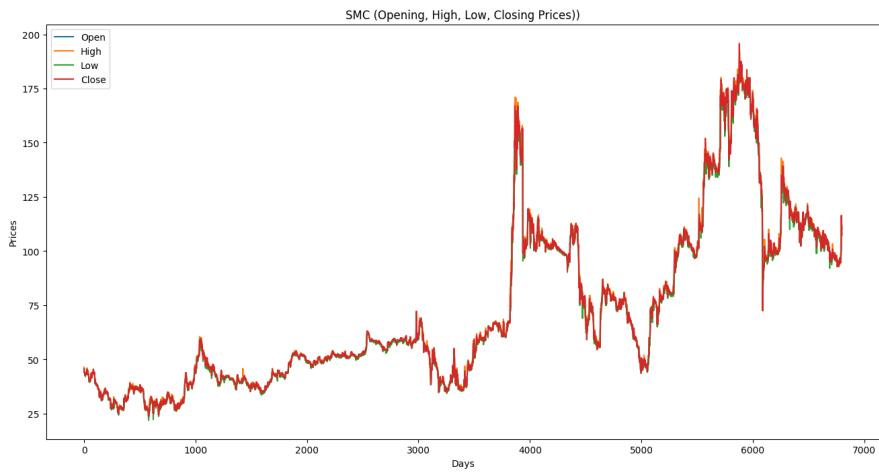


Figure B.18: Opening, High, Low, and Closing Prices on SMC

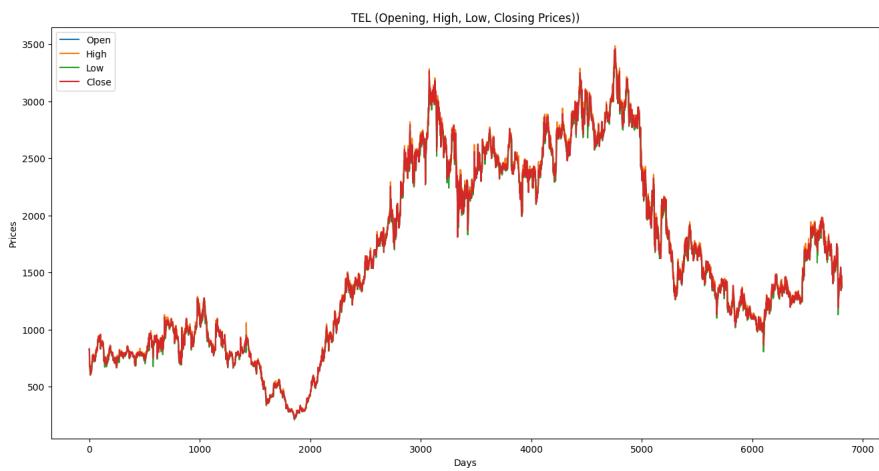


Figure B.19: Opening, High, Low, and Closing Prices on TEL

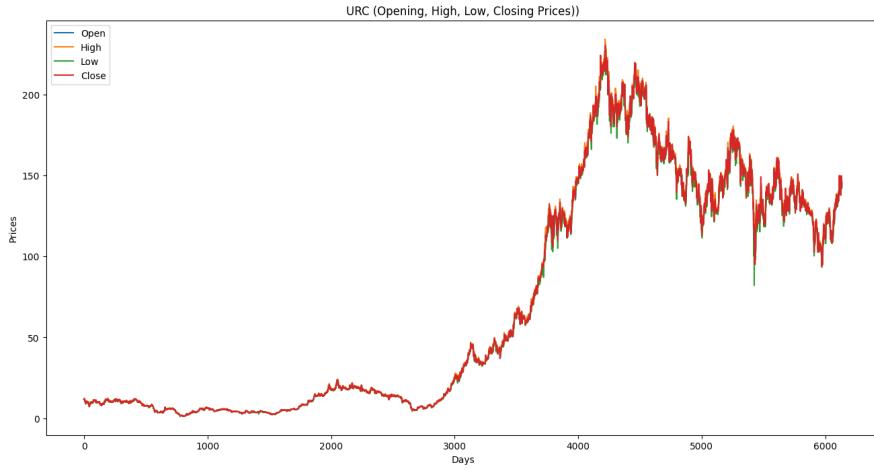


Figure B.20: Opening, High, Low, and Closing Prices on URC

B.2 Raw Model Testing and Cross-Validation Results

The loss metric scores for all eight models trained in this special problem are shown in the figures below.

```
model_baseline5.csv ×
,MSE,RMSE,MAE,MAPE
MEG,0.08072020880069887,0.2841130211741427,0.27684353215352125,1174375793447.243
JGS,0.07874435548854763,0.2806142467668875,0.27468463899133194,38927690524.79277
BDO,0.076570968585490822,0.27671459770837575,0.2738142088271028,0.07038134913330009
FGEN,0.07827957682896561,0.2797848759832554,0.2758278796079067,0.09700174437968641
ICT,0.0781394551452297,0.27953435414136435,0.2736751085473064,1438507573768.6584
ALI,0.0782312069917634,0.2796984215038823,0.27580877757070465,0.1147421270770546
SMC,0.07691227333252271,0.2773306209788647,0.27482445944890904,0.06707341533873201
TEL,0.07692274391391254,0.27734949777115614,0.2741700883069239,0.038119574269088366
GLO,0.07723200155169901,0.2779064618746729,0.2740521994194121,0.040133964505169824
BLOOM,0.09321696719529625,0.30531453813288395,0.2859447234165783,499188752957.71985
RLC,0.07891466401483171,0.2809175395286519,0.27499496364466064,0.2790370163868907
MER,0.07860945075825411,0.28937376974006345,0.2748269458519563,0.06180059505843166
AC,0.0775170120978667,0.2784187710946708,0.2741992480954101,0.04794242756170382
PGOLD,0.07650623749204255,0.2765976093389864,0.2738838287946534,0.07546795486985015
LTG,0.09027857607286971,0.30046393472906147,0.2845669124314392,2960468439858.6323
HPI,0.07896762492375467,0.2810117878733109,0.2764535402803122,0.19535308715023528
AP,0.07613355995038901,0.27592310514052465,0.2735462968120787,0.09278031794532755
RRHT,0.07715208334861212,0.277762638503835,0.2753872091002812,0.06488774812116955
URC,0.07801107752908334,0.27930463212965756,0.2739888283847051,0.10613413435989405
PSEI,0.07615895487932436,0.27596911943064273,0.275949626259796,0.12659175356184194
```

Figure B.21: Raw Model Scores for Baseline 5

```

model_baseline10.csv ×
,MSE,RMSE,MAE,MAPE
MEG,0.00989911754936293,0.0994480354458362,0.06758199064508962,434720971982.4056
JGS,0.007846138726297082,0.08857843262497414,0.06152327438608147,85566335564.84262
BDO,0.0032789824378326032,0.05726239986092622,0.04134258195038248,0.010787188658697832
FGEN,0.00470003599831854,0.06855657225847755,0.049436469316649294,0.018101504501868163
ICT,0.00782900848597167,0.08848209337824896,0.06833878085820871,357524334780.4415
ALI,0.005405470132370961,0.07352190239901958,0.05348223646735878,0.023718671100299755
SMC,0.0032675047380823336,0.05716209179239624,0.03835913806689466,0.009415067971236714
TEL,0.003890710198736975,0.06237555770281317,0.046544974902766156,0.00657076444334346
GLO,0.0047083932277881038,0.06858521909187895,0.04996579018796145,0.0074512575673198805
BLOOM,0.042053914090353764,0.20587051004557864,0.10369141607734932,2475475634567.023
RLC,0.00759224342571167,0.0870587487591631,0.05984216628062193,0.09443111972264119
MER,0.006914484697945786,0.08315338055632968,0.056430766701572516,0.013803759059196301
AC,0.004979648327052059,0.07056662332187973,0.0566851267808547,0.009095293073218335
PGOLD,0.0031896313274378794,0.0564768211520554,0.042048119990808236,0.011758645358492095
LTG,0.03750821267854841,0.193670371194327,0.08933899463198181,3758594209186.506
NPI,0.0055622165729049605,0.07458026932711466,0.04957733886385026,0.03681429790598718
AP,0.0025557353305988954,0.05055428103137157,0.0366241701515080946,0.013458472431218822
RRHI,0.0028975891955775315,0.05382925966031422,0.040626489078535964,0.009641290779521416
URC,0.0068894743382426505,0.08300285741010757,0.057194705397388856,0.028217183434511548
PSEI,0.232829705334228e-05,0.009608761473433623,0.008494095263759951,0.0038998382619416603

```

Figure B.22: Raw Model Scores for Baseline 10

```

model_baseline15.csv ×
,MSE,RMSE,MAE,MAPE
MEG,0.034015743948385566,0.18443357597895663,0.15164354569027236,633909811283.2087
JGS,0.029592741877537167,0.17202541055767653,0.144534965493625,8993505239.85993
BDO,0.02056771950246394,0.14341450241333314,0.12778323238785438,0.03305043108059346
FGEN,0.024561083742794965,0.15671976181322816,0.13684348348281122,0.04888985126466148
ICT,0.028606631454426018,0.16913495042251325,0.139251839245286,961503494279.2739
ALI,0.02492924006341544,0.15788996188300078,0.13549120794438474,0.05804993752172819
SMC,0.02179394927348297,0.14762773883098088,0.134740084267869616,0.0329682898503888
TEL,0.0214783746770632,0.1465550226947654,0.1297835904754104,0.01812192661107948
GLO,0.0229663087789239,0.15154639150743213,0.13391374593363138,0.01978636490688916
BLOOM,0.07631830893786716,0.2762576857534776,0.18855400548274345,5347063331582.995
RLC,0.028379678384273657,0.16846266762779716,0.1432828414731735,0.19419272464274054
MER,0.0272509810000031,0.1658787152878284,0.1409521383013787,0.032682665435146876
AC,0.023753398655402828,0.15412137637395673,0.13366109786697314,0.023605071006055328
PGOLD,0.020648336647877637,0.14369529097321748,0.129340835195131813,0.035568815339653496
LTG,0.07658990441865482,0.27674881105192634,0.1774756863473615,2504326275497.088
NPI,0.026182659841909636,0.161810567769567,0.14017747870431393,0.10104140736854278
AP,0.019192122314937974,0.13853563554168283,0.12615347917006875,0.042758918225015656
RRHI,0.021215358356749883,0.14565492905065,0.13197476027457852,0.03120428832678116
URC,0.02678486002453064,0.16366808784516077,0.13675722964040252,0.05989435898919455
PSEI,0.017520964135483975,0.1323667782113765,0.13221706472713188,0.06066009164259066

```

Figure B.23: Raw Model Scores for Baseline 15

```

model_baseline20.csv ×
,MSE,RMSE,MAE,MAPE
NEG,0.026776145694400132,0.16363418253653522,0.1200530791025837,351366074670.9161
JGS,0.023218979019200704,0.15237775106360082,0.11203671688465545,37079236101.84903
BDO,0.010908683929902947,0.10444464529071343,0.08434523255132731,0.021781428827316717
FGEN,0.01328484000198452,0.1152564098086719,0.09013206705640547,0.03222585823277696
ICT,0.023345216418908073,0.1527914174215124,0.1113250141499374,329061136398.89325
ALI,0.01528648733345764,0.12363821712296633,0.09754795116104639,0.04187354853357004
SMC,0.010628930207385257,0.10309670318388099,0.07783085131763114,0.018916425305997443
TEL,0.009939364863936338,0.0996963633435861,0.0788610299993594,0.01109755333457707
GLO,0.012562185108249477,0.1128811541172265,0.08280467054196113,0.012332156586357538
BLOOM,0.08783425054231923,0.29636843715604944,0.16887164674288263,3169069166729.4937
RLC,0.020691999516582075,0.14384713941449393,0.107188974059568002,0.1199250772426462
HER,0.01812308145308288,0.13462199468542604,0.09868743232128989,0.023716153812441354
AC,0.01394238486958509,0.11807787629181468,0.089344392564019,0.01585107064231005
PGOLD,0.018904295680437671,0.18442363564077661,0.0817661727419002,0.022751911781967227
LTG,0.08067359183535965,0.28403096985251386,0.1329894362117306,1705928819526.0623
NPI,0.014631546535808101,0.12096892979860677,0.09220156709982739,0.08615668537253527
AP,0.01002913628735902,0.10014557547569947,0.08141626445268184,0.029241478007478686
RRHI,0.008797360758064962,0.09379424693479319,0.07504927916672673,0.01764922385349229
URC,0.01978034871975187,0.14035793073336422,0.10510373384399553,0.04846541830892664
PSEI,0.004074235411641981,0.06382973767486422,0.06342811204197056,0.029093785646223835

```

Figure B.24: Raw Model Scores for Baseline 20

```

model_s5.csv ×
,MSE,RMSE,MAE,MAPE
NEG,0.004309424873421442,0.06564620989380454,0.04421975007142992,139304189558.7179
JGS,0.00330889006950838,0.05752295200136062,0.039920870940521255,200992323983.5872
BDO,0.0015994420966082207,0.039993025599574496,0.02798883139579814,0.007247638327583592
FGEN,0.00224052899740447,0.04733422651465266,0.03265112428473698,0.011974381478464668
ICT,0.0033466528508440136,0.05785026232303544,0.03731193820350593,300581835474.86523
ALI,0.0025544953299240854,0.050542015491312625,0.03645126031585048,0.01596931133193516
SMC,0.0013703086979771,0.03701768088329832,0.023174377850353648,0.005691306162396273
TEL,0.0017757163226041189,0.0421392491936451,0.030019849241503773,0.004239817355480039
GLO,0.0021113102967080873,0.0459489668872948,0.03149099407019385,0.004676968507849213
BLOOM,0.018825601880785605,0.13720642069810585,0.069008741213238494,1052898138850.6415
RLC,0.0033839823305168376,0.058172006416461496,0.03977738120052245,0.06922003880110895
HER,0.002586824275804447,0.057884870391202996,0.037700469138267696,0.009169601650490272
AC,0.0023584838369610636,0.0485642383772918,0.0341376208936506,0.006109170178242572
PGOLD,0.0014934950745029397,0.038645763991709874,0.028179754468478487,0.007879349189599448
LTG,0.015669529634599597,0.1251779918140549,0.05857964135132494,3583334717407.9917
NPI,0.00273483507602531,0.052295650641571614,0.033993850342583584,0.024971153007954654
AP,0.0012930670339860782,0.03595924127868679,0.02515073464898651,0.009217250854869657
RRHI,0.0013093195909138609,0.036184521427166355,0.026990284358758658,0.00638869156609276
URC,0.002966671813785288,0.05446716271098843,0.037421547517402926,0.017977619586139627
PSEI,1.7547831110700054e-05,0.004189013142817775,0.003282715775307006,0.001507895820047589

```

Figure B.25: Raw Model Scores for DMD-LSTM 5

```

model_s10.csv ×
,MSE,RMSE,MAE,MAPE
MEG,0.009865752600343098,0.09932649495649737,0.06737299027636771,4357340646705.6676
JGS,0.007845334501853743,0.08857389289092889,0.06135788509350167,103893946584.96439
BDO,0.00327995409232174,0.05727088346028669,0.0413452891693641,0.010743648101452627
FGEN,0.004617395242386965,0.06795141825147556,0.048889136842270545,0.017866975592835
ICT,0.007884621330006128,0.08879539925208649,0.057237276691394216,320809756127.44825
ALI,0.005390492967401505,0.07341997662354234,0.05328993535240997,0.023548819706954743
SMC,0.003187718406540048,0.05645988316087847,0.03586861396100476,0.00880857357236526
TEL,0.0036866145213750832,0.060717497654095426,0.04474277319282044,0.006319036216914404
GLO,0.004552884186453377,0.06747586344164925,0.047282846382667394,0.0070468257691598275
BLOOM,0.04206409341784911,0.20509532763534402,0.10301736984181184,2396590596797.983
RLC,0.007585752882284707,0.0870962277155831,0.059771887078320866,0.09172759091241212
MER,0.006828068233221027,0.08263212591493109,0.055823760181086673,0.013659473951391473
AC,0.004817427887048709,0.06979593746808412,0.04979303078429352,0.008922829587183788
PGOLD,0.003183147916330252,0.056419393087220035,0.04133830254008775,0.011574066171746585
LTG,0.03746440033346864,0.1935573009458997,0.08800307126114834,3804223137493.905
MPI,0.005520841190178309,0.07430236328797564,0.0492521638149232,0.03649038768492956
AP,0.0026026827963171323,0.05101649533550038,0.0367471692891345,0.01358330929825775
RRHI,0.00277849936982922,0.05270531222735449,0.03922518794653118,0.009295106895914377
URC,0.006898006661978345,0.08305423927216686,0.057343117943704225,0.02816949652429363
PSEI,4.085433653339971e-05,0.00639173971105987,0.004984381407170562,0.002288420978937786

```

Figure B.26: Raw Model Scores for DMD-LSTM 10

```

model_s15.csv ×
,MSE,RMSE,MAE,MAPE
MEG,0.015632947601103515,0.1250317863629226,0.08599008102247829,282985182400.4937
JGS,0.012964389929998767,0.11386127493576895,0.07835242363030513,109315071771.05272
BDO,0.004860908219981323,0.06972021385495976,0.05080170327959182,0.013273483429938518
FGEN,0.0069629478504800255,0.08344427991468334,0.061532899353523559,0.022449511828279718
ICT,0.013007479593210832,0.1140503379793801,0.07292148376731492,396061318234.26843
ALI,0.00810466519619875,0.09002591402590007,0.066367103228538,0.029520448994738713
SMC,0.005101124231403319,0.07142215504591919,0.04582754439638881,0.011229880466157778
TEL,0.005439188193806359,0.07375085215647585,0.05519736046986012,0.007819104873858874
GLO,0.007018476178400337,0.08377634617480245,0.05858766515961793,0.008779822537746144
BLOOM,0.06054300374145961,0.24605487953190364,0.1309550411508476,5900492566389.731
RLC,0.011676685314347363,0.10805871234818303,0.07548320907116821,0.11945476091327935
MER,0.010658164462570354,0.10338357926948726,0.07023073059770062,0.01728297609360001
AC,0.007595399918192069,0.0871515399645472,0.00145688930225037,0.011039297095713095
PGOLD,0.004859201309319683,0.06970797163395076,0.0512276244385843,0.014339564237239416
LTG,0.05808674831062965,0.24101192566059806,0.10807781590206503,2316692766061.9355
MPI,0.008080401796946723,0.0898910551553753,0.06020282472442268,0.04473013446882734
AP,0.0038632369071855297,0.06215494274139049,0.04434230453119878,0.016505958531928256
RRHI,0.004132716283832857,0.06428628601523205,0.04803456414328379,0.011401074357359176
URC,0.010867588278427435,0.10424772553119534,0.07207244516575859,0.036246928116491314
PSEI,3.912591469048572e-05,0.006255071117939885,0.004142081727067567,0.0019046016464762977

```

Figure B.27: Raw Model Scores for DMD-LSTM 15

```

model_s20.csv ×
,MSE,RMSE,MAE,MAPE
MEG,0.022323896742406167,0.14941183601845662,0.10290009425682627,266729753264.34708
JGS,0.018713341878839475,0.13679671735493484,0.09445716160359285,33462393302.143425
BDO,0.006756430654995097,0.08219751002916753,0.060170842223956744,0.015760545579428267
FGEN,0.009597141213378582,0.09796499994068587,0.07234479676026784,0.026376477197220943
ICT,0.01852442554536271,0.1361044655996136,0.0884936204475373,501915336431.2399
ALI,0.01079138682075038,0.10525748753449816,0.07817689171626877,0.034775151593376956
SMC,0.007048757669965641,0.08395687982509617,0.0546836838155754,0.013374177511788293
TEL,0.007329064462734207,0.08560995539500185,0.06517997383340782,0.009244621847414265
GLO,0.009758692882070095,0.09878609660306502,0.06921019624724692,0.010395214033356167
BLOOM,0.08238165998516955,0.28702205487587457,0.1577616055528068,3419715539589.91
RLC,0.0161519633593326,0.1270903747706667,0.08988687939937239,0.12828506623288913
HER,0.01468617694758008,0.12118653781497382,0.08247620417598935,0.020392816829558198
AC,0.010767353439365156,0.10376585873678181,0.07244617014525238,0.013014783427266284
PGOLD,0.006606336237122399,0.08127937153498666,0.06071069833794954,0.016982336902546864
LTG,0.07756279210367704,0.2785009732544521,0.12605638618291093,1762772789831.8223
NPI,0.019612065609680215,0.1030148805254863,0.06982793923155896,0.05185919561554108
AP,0.0052532780558852705,0.07247950093568023,0.05175236778740295,0.019295214734668182
RRHI,0.0054869934800239875,0.07407424302700627,0.0564675805428636,0.013389493057254155
URC,0.015310641590353787,0.12373617737086348,0.0868352222276058,0.04379474785289333
PSEI,7.322582229059844e-05,0.00855720879087325,0.006455499639920296,0.0029648261136105067

```

Figure B.28: Raw Model Scores for DMD-LSTM 20

B.3 Model Testing Raw Test Results for DMD-LSTM

The graphs below show the performance of the different DMD-LSTM models trained in the conduct of this special problem based on the test data split of PSEI.

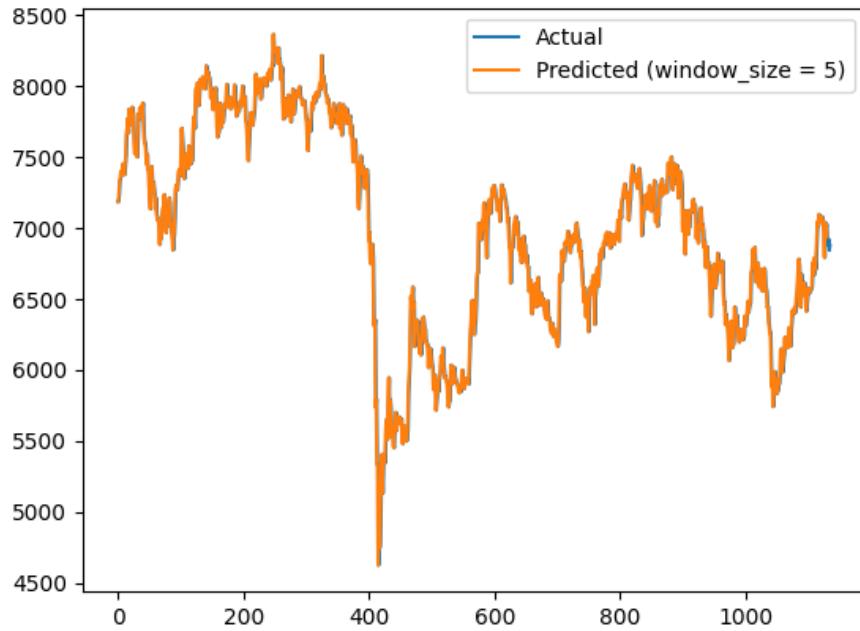


Figure B.29: Actual vs Predicted Closing Prices for DMD-LSTM 5 (Using Train Data from PSEI)

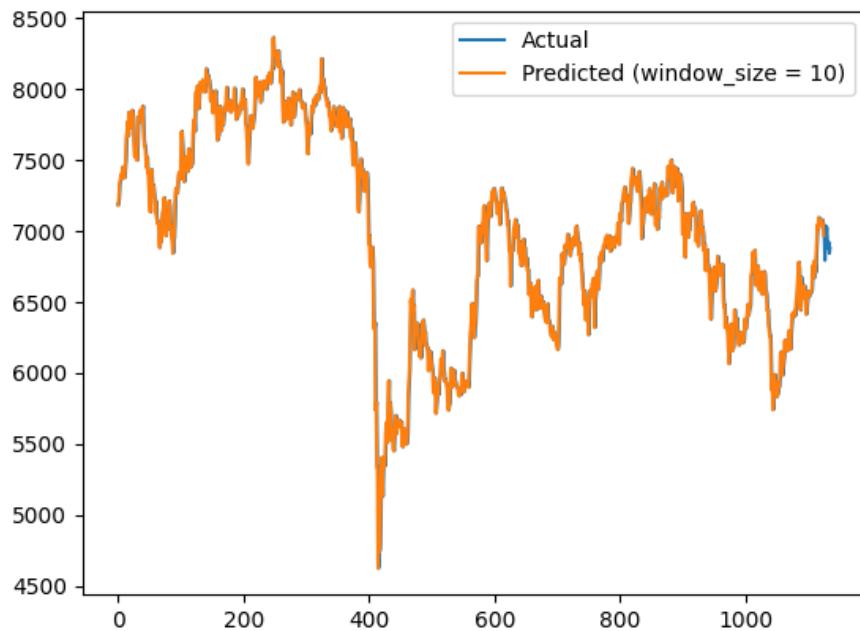


Figure B.30: Actual vs Predicted Closing Prices for DMD-LSTM 10 (Using Train Data from PSEI)

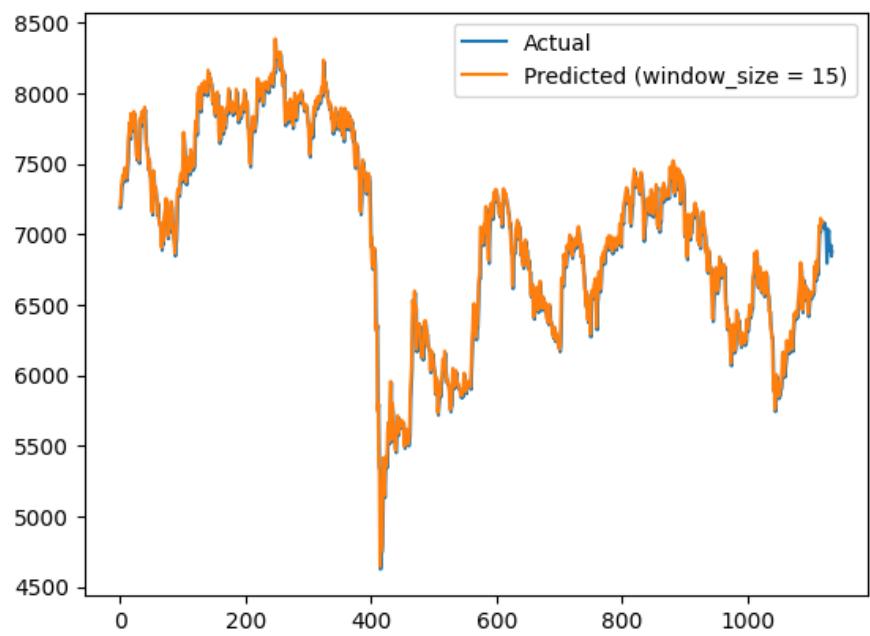


Figure B.31: Actual vs Predicted Closing Prices for DMD-LSTM 15 (Using Train Data from PSEI)

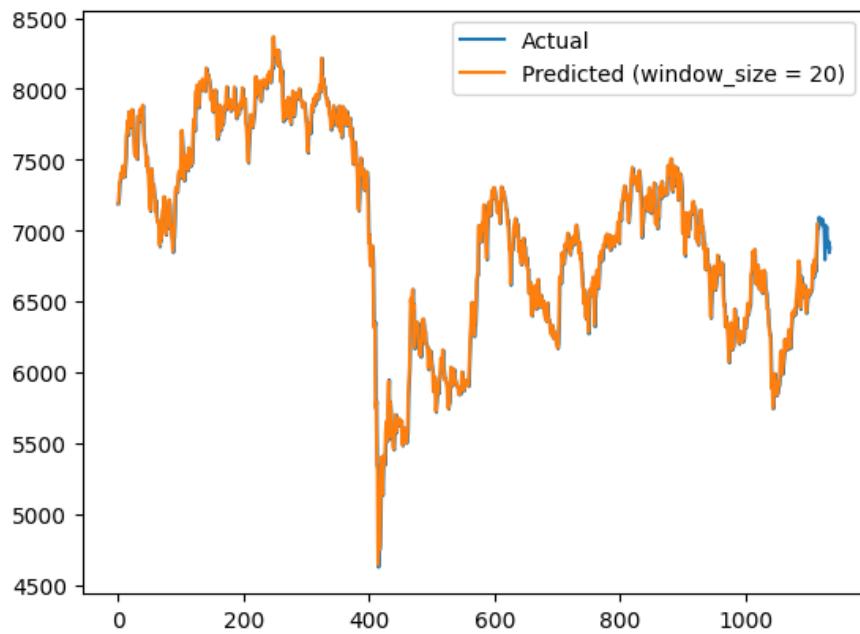


Figure B.32: Actual vs Predicted Closing Prices for DMD-LSTM 20 (Using Train Data from PSEI)

B.4 Daily Return Distribution of the Different Stocks

Figures below show the daily return distribution of each stock, which was used to calculate each stock's risk profile.

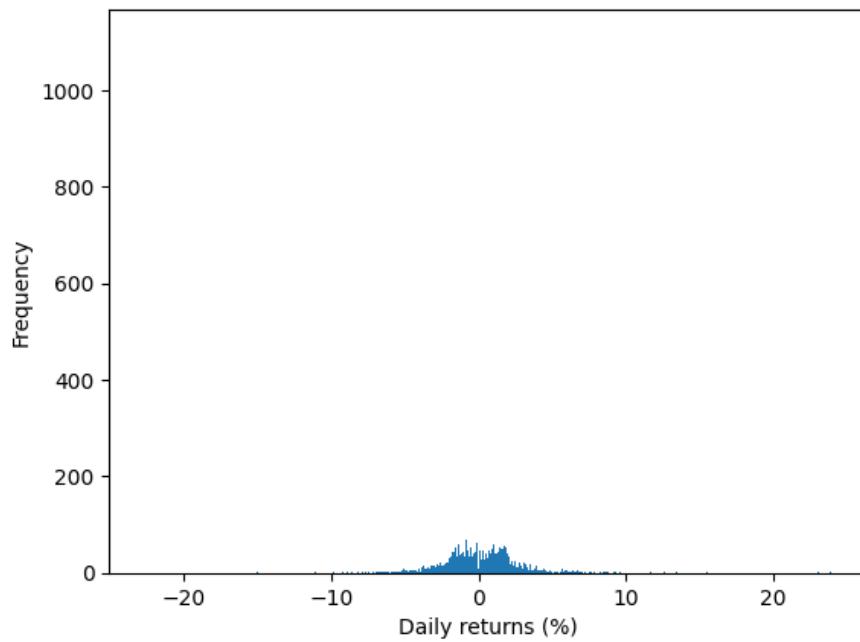


Figure B.33: Daily Return Distribution of AC

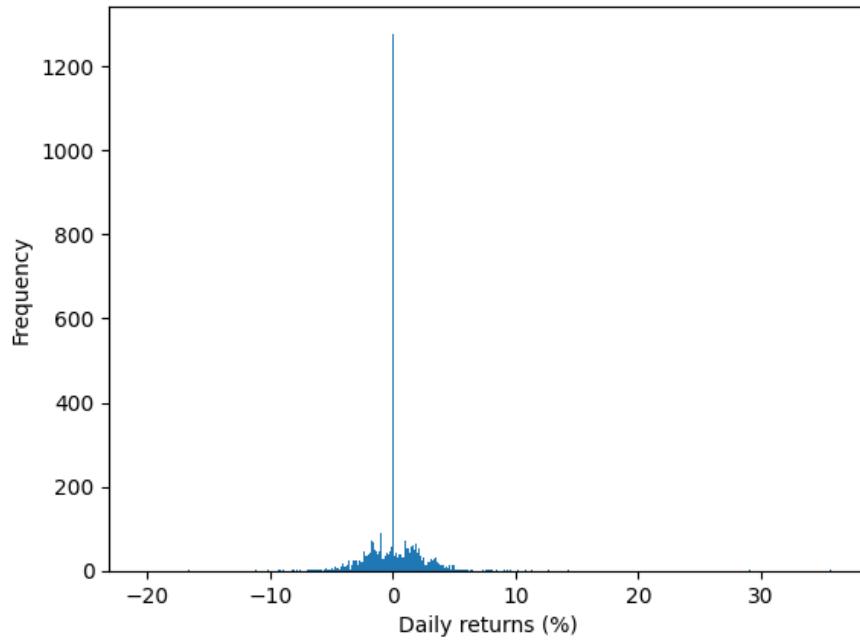


Figure B.34: Opening, High, Low, and Closing Prices for ALI

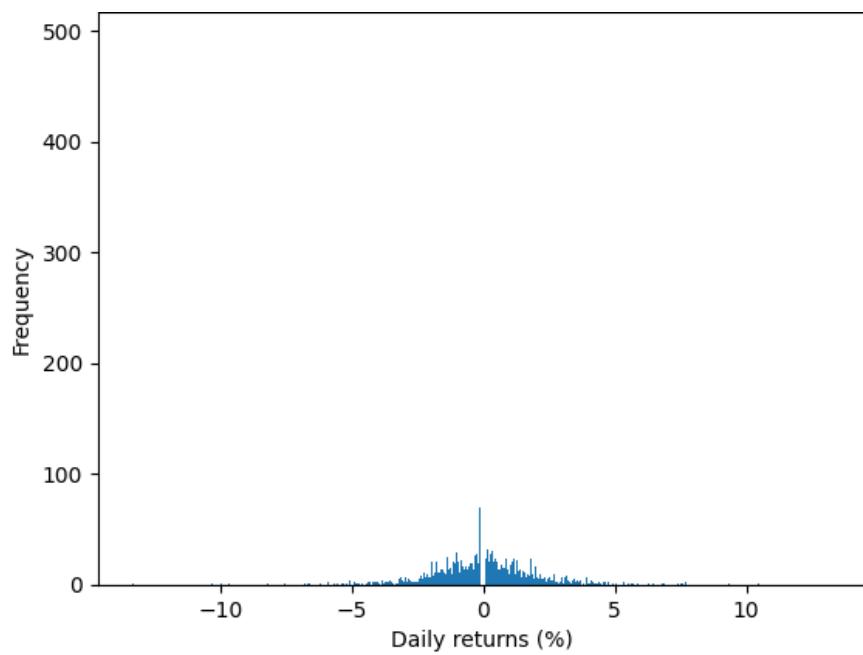


Figure B.35: Opening, High, Low, and Closing Prices for AP

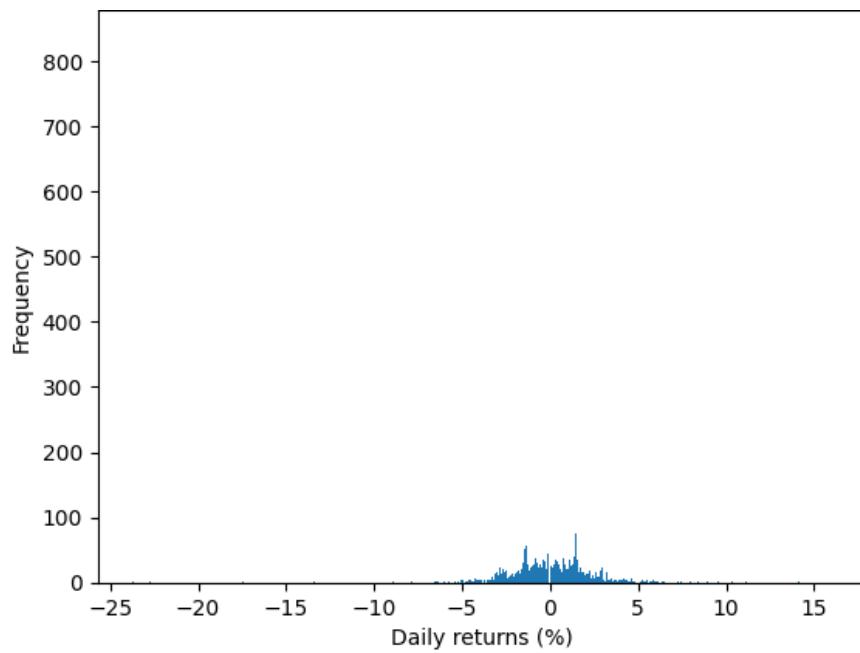


Figure B.36: Opening, High, Low, and Closing Prices for BDO

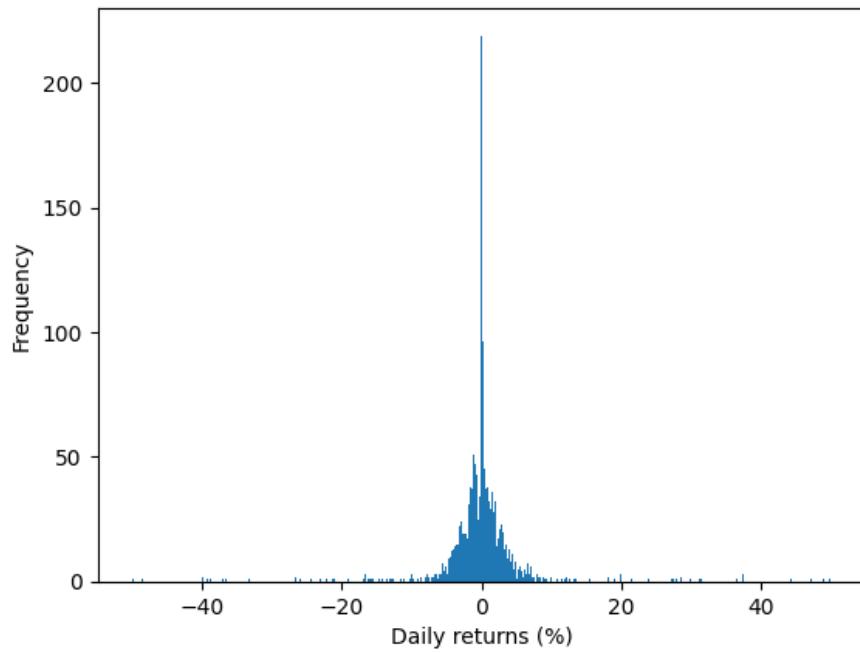


Figure B.37: Opening, High, Low, and Closing Prices for BLOOM

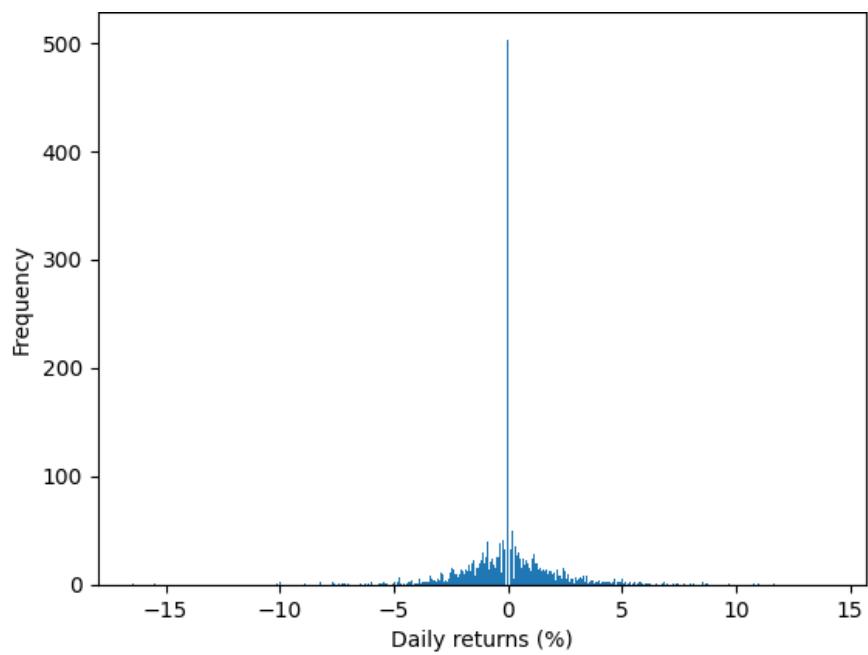


Figure B.38: Opening, High, Low, and Closing Prices for FGEN

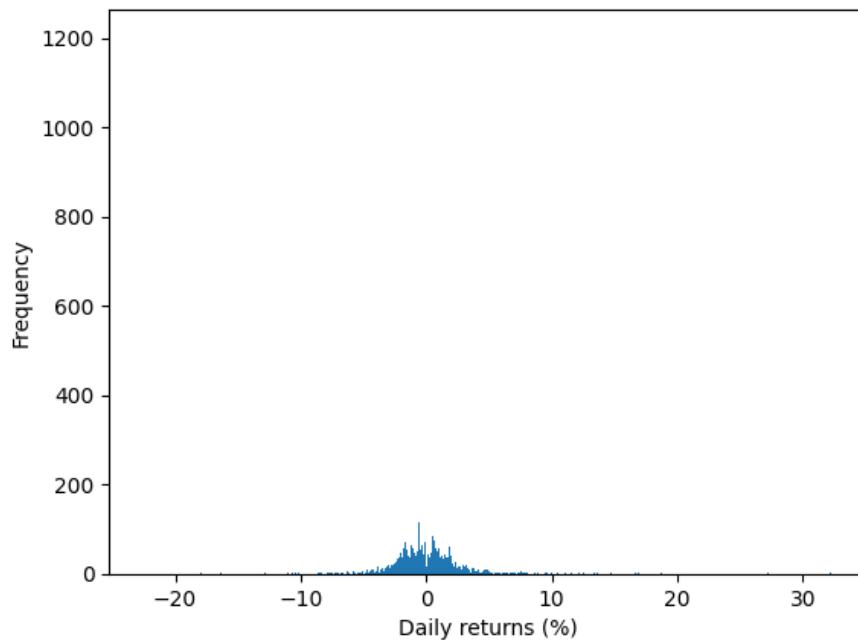


Figure B.39: Opening, High, Low, and Closing Prices for GLO

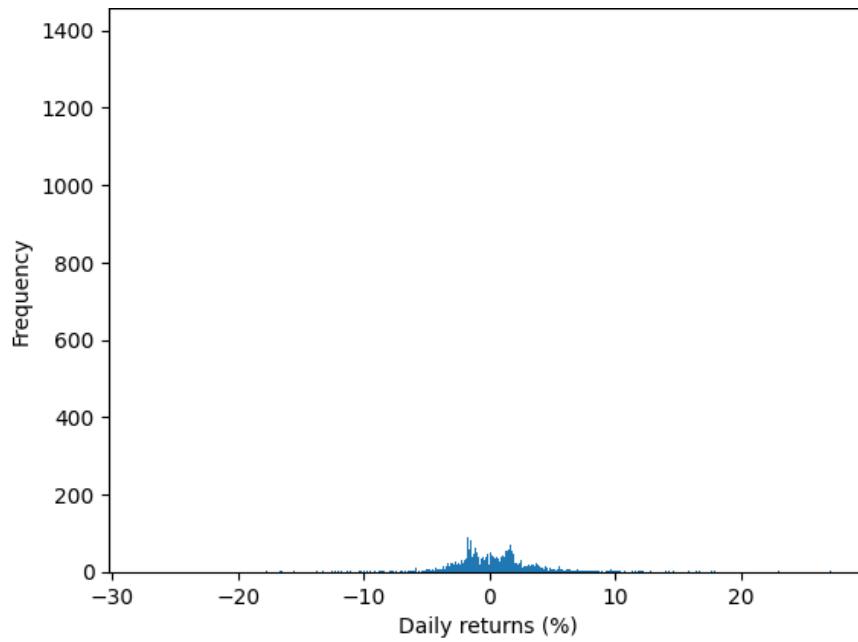


Figure B.40: Opening, High, Low, and Closing Prices for ICT

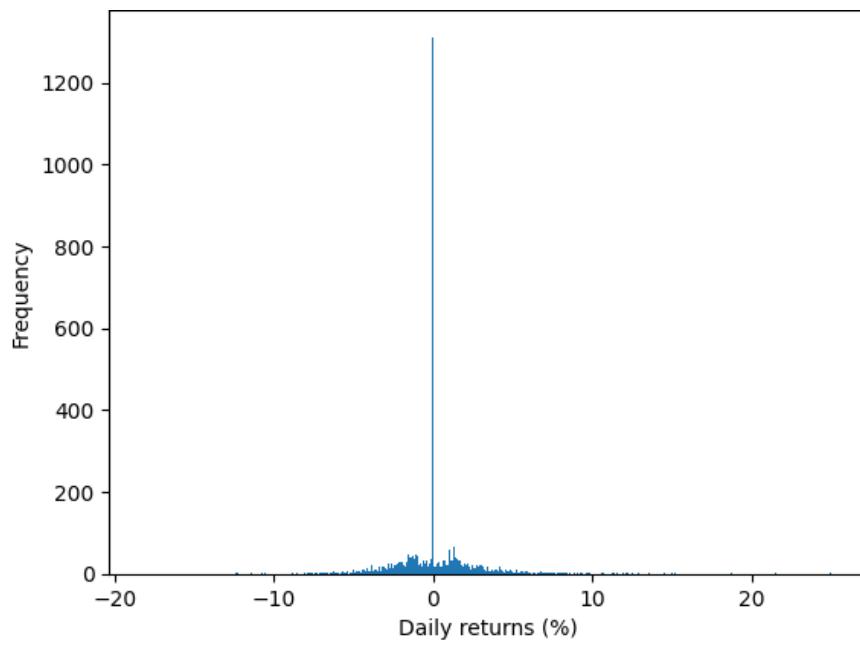


Figure B.41: Daily Return Distribution of JGS

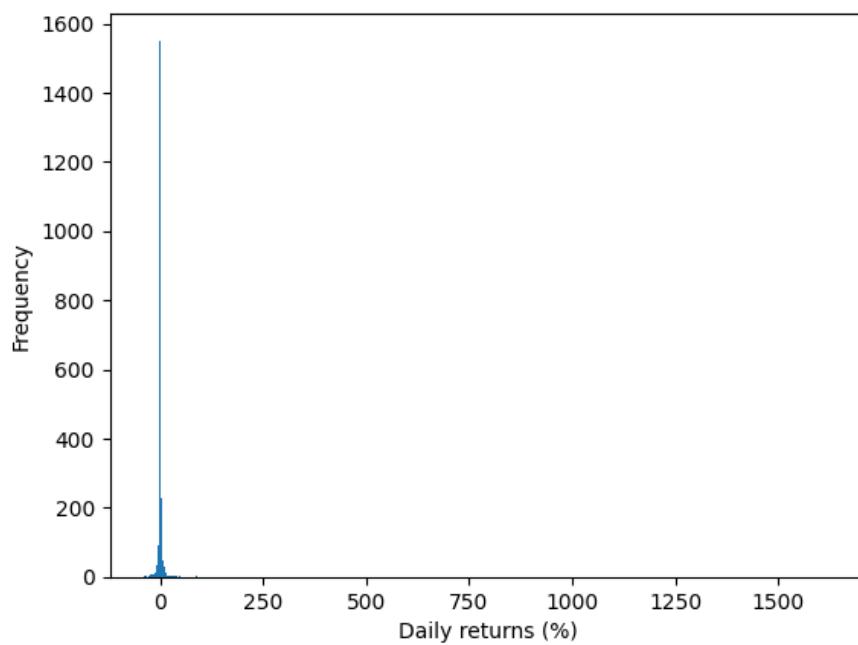


Figure B.42: Daily Return Distribution of LTG

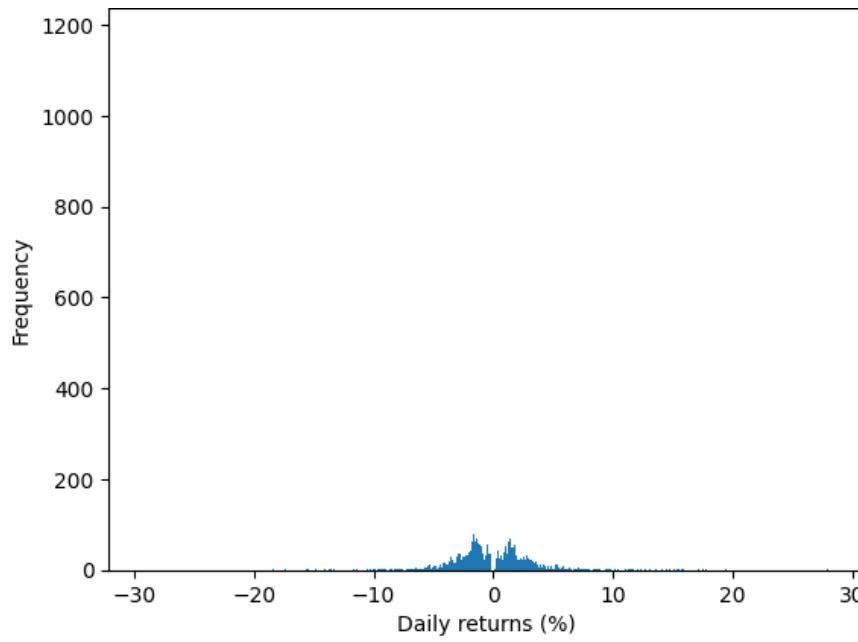


Figure B.43: Daily Return Distribution of MEG

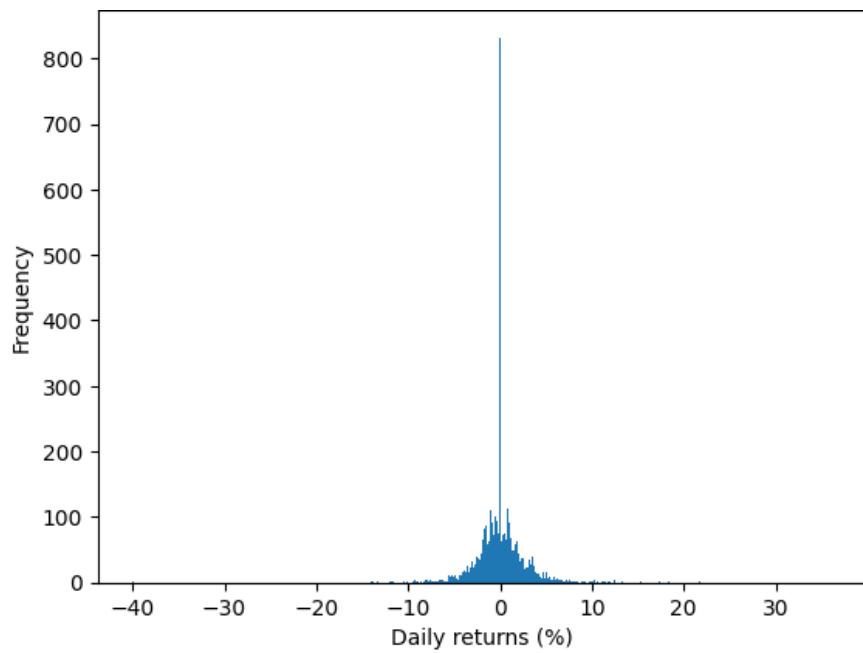


Figure B.44: Daily Return Distribution of MER

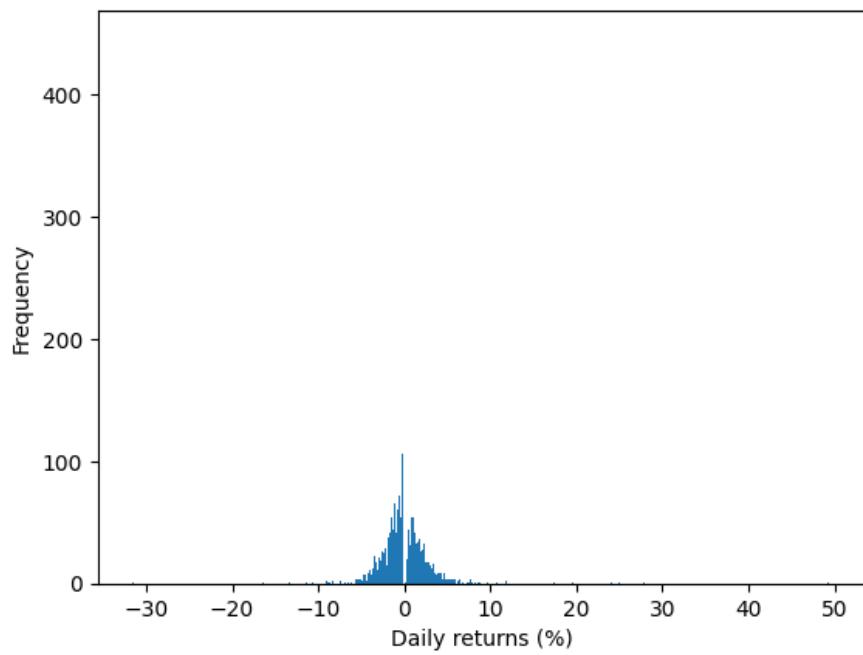


Figure B.45: Daily Return Distribution of MPI

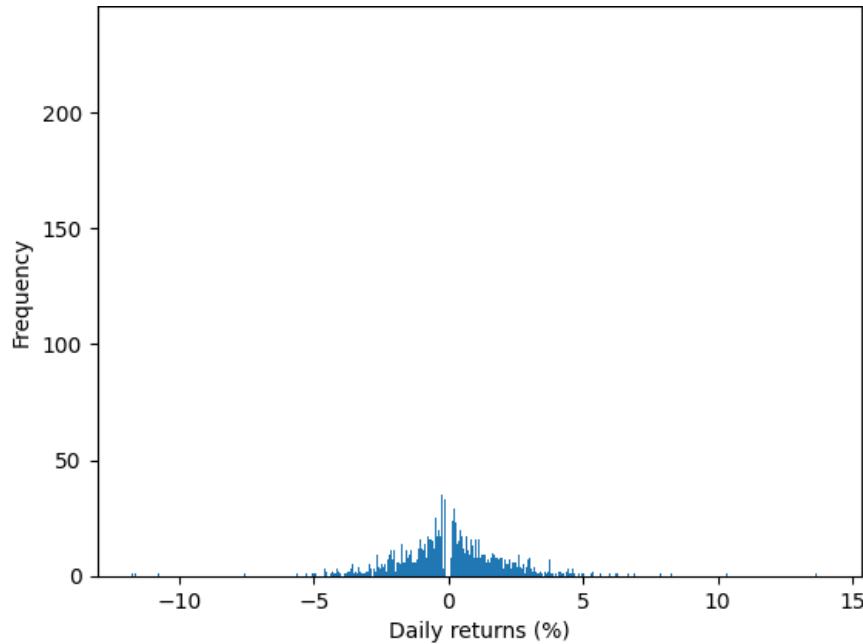


Figure B.46: Daily Return Distribution of PGOLD

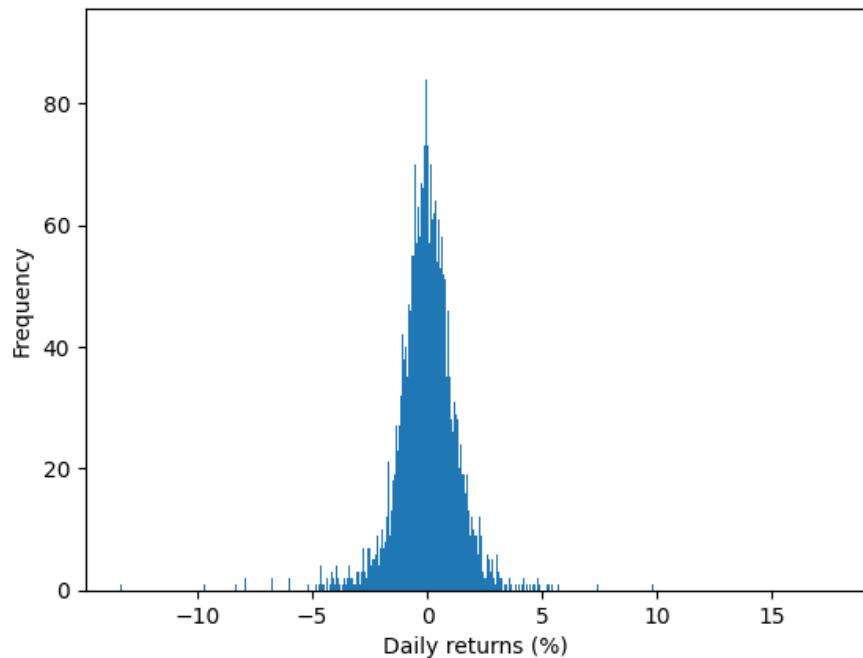


Figure B.47: Daily Return Distribution of PSEI

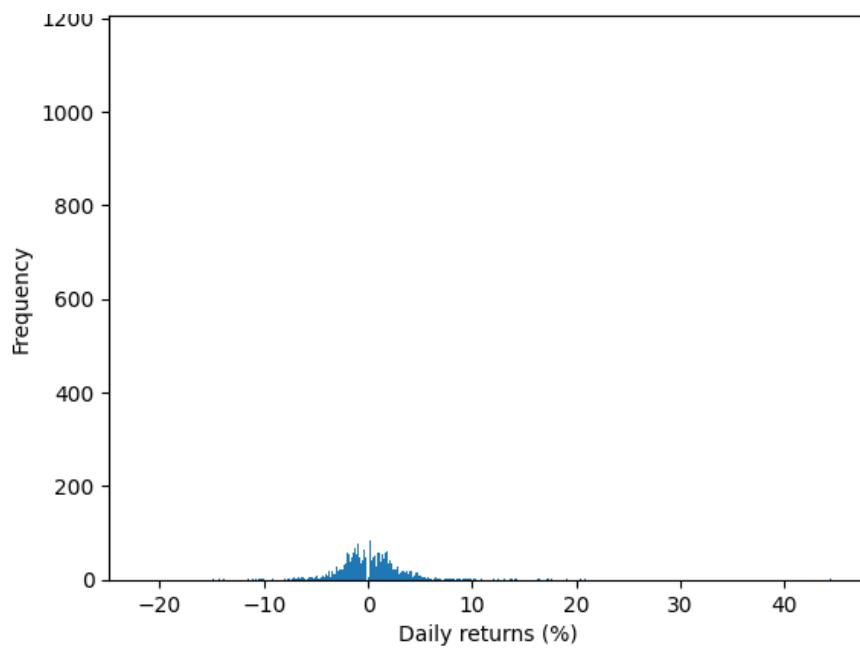


Figure B.48: Daily Return Distribution of RLC

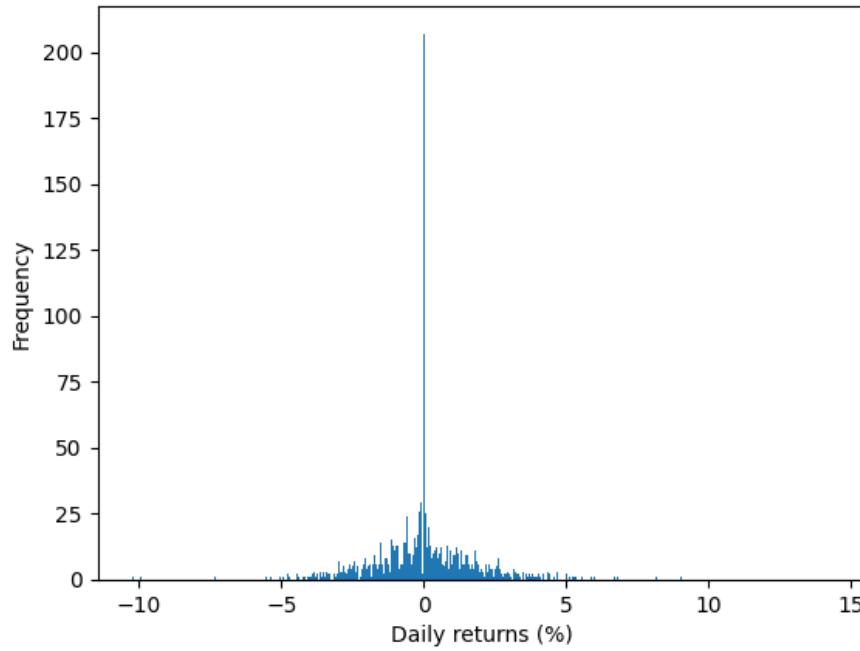


Figure B.49: Daily Return Distribution of RRHI

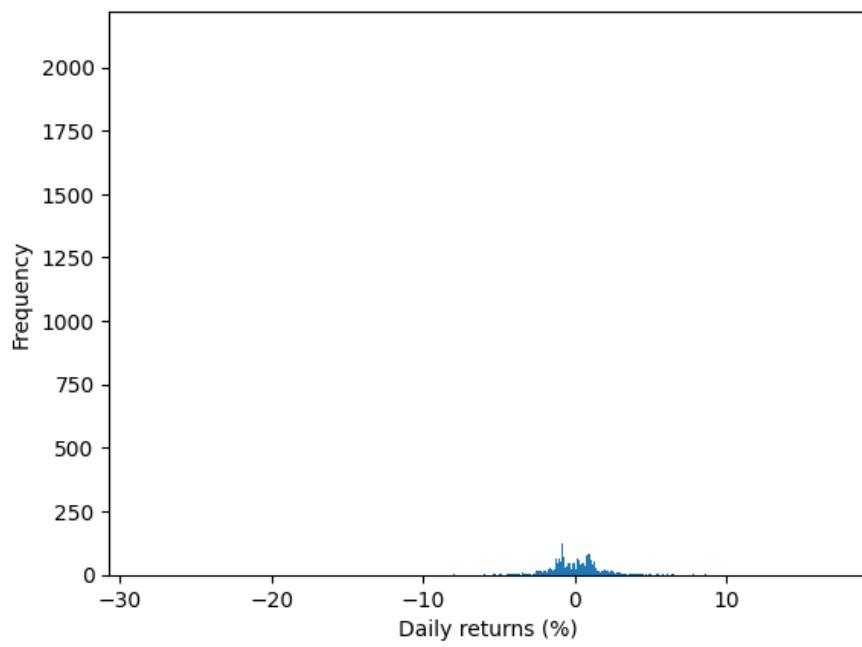


Figure B.50: Daily Return Distribution of SMC

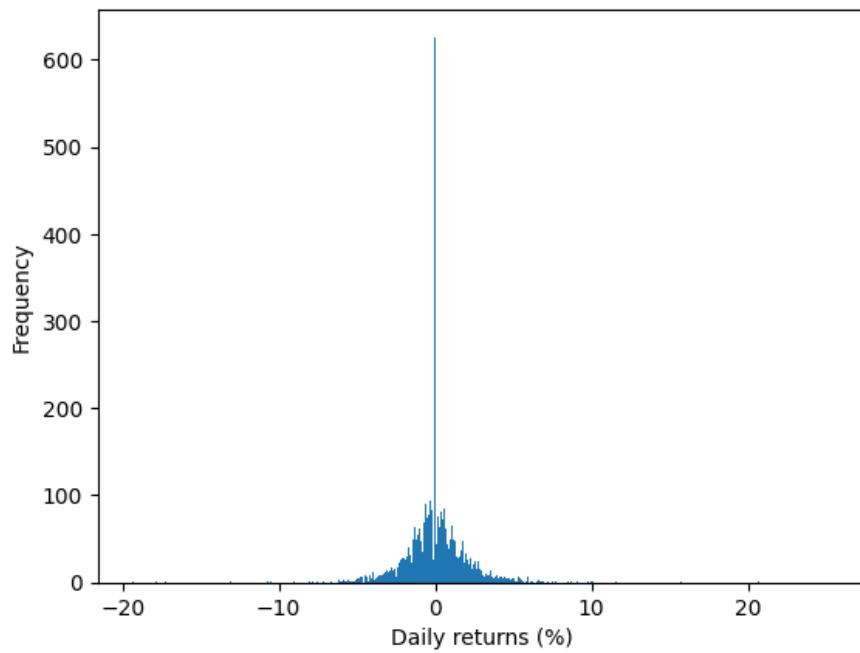


Figure B.51: Daily Return Distribution of TEL

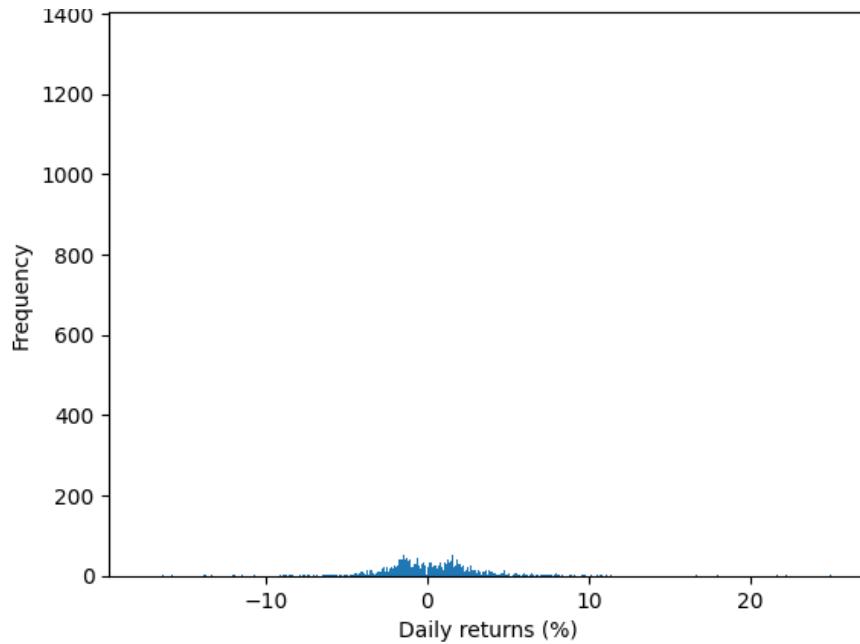


Figure B.52: Daily Return Distribution of URC

stock	value_at_risk%	volatility%	drawdown%	start_date	end_date
MEG	-5.365715	3.949969	57.248140	2000-01-03	2023-02-10
JGS	-4.762357	3.361099	43.184044	2000-01-03	2023-02-10
BDO	-3.302488	2.364049	39.670648	2000-01-03	2023-02-10
FGEN	-3.819079	2.559254	30.604259	2000-01-03	2023-02-10
ICT	-4.827050	3.520840	54.645013	2000-01-03	2023-02-10
ALI	-4.390485	3.070487	56.026166	2000-01-03	2023-02-10
SMC	-3.403674	2.385956	45.298783	2000-01-03	2023-02-10
TEL	-3.693763	2.460023	44.659116	2000-01-03	2023-02-10
GLO	-4.120044	3.092601	54.758065	2000-01-03	2023-02-10
BLOOM	-5.984996	7.061554	100.000000	2000-01-03	2023-02-10
RLC	-4.529364	3.416989	65.982906	2000-01-03	2023-02-10
MER	-4.498595	3.254736	76.000028	2000-01-03	2023-02-10
AC	-4.290654	2.795171	46.688827	2000-01-03	2023-02-10
PGOLD	-3.114919	2.134819	25.799794	2000-01-03	2023-02-10
LTG	-6.153221	31.223317	1667.806911	2000-01-03	2023-02-10
MPI	-4.055836	3.499676	81.132519	2000-01-03	2023-02-10
AP	-3.197641	2.187289	26.540881	2000-01-03	2023-02-10
RRHI	-3.032063	2.053932	24.509668	2000-01-03	2023-02-10
URC	-4.532392	3.199716	42.542617	2000-01-03	2023-02-10
PSEI	-1.887800	1.318818	30.903657	2000-01-03	2023-02-10

Figure B.53: Raw Risk Profile Scores

B.5 Raw alamSYS Test Data

This section is divided into three sections: raw system logs, PSEI trading baseline data, and raw real-world alamSYS application.

B.5.1 Raw System Logs

stats.txt									
1	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
2	d878d6975a9c	alamDB	0.23%	159MiB / 7.684GiB	2.02%	9.55kB / 9.65kB	55.59B / 680kB	■	
3	00b3dc1eb4ce	alamAPI	0.17%	45.7MiB / 7.684GiB	0.58%	9.07kB / 5.64kB	28.29B / 0B	3	
4	47dba53c6237	alamPREPROCESSOR	0.01%	350.7MiB / 7.684GiB	4.53%	3.1kB / 2.65kB	283MB / 12.3kB	15	
5	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
6	d878d6975a9c	alamDB	0.28%	159MiB / 7.684GiB	2.02%	9.55kB / 9.65kB	55.59B / 680kB	■	
7	00b3dc1eb4ce	alamAPI	0.15%	45.7MiB / 7.684GiB	0.58%	9.07kB / 5.64kB	28.29B / 0B	3	
8	47dba53c6237	alamPREPROCESSOR	0.01%	350.7MiB / 7.684GiB	4.53%	3.1kB / 2.65kB	283MB / 12.3kB	15	
9	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
10	d878d6975a9c	alamDB	0.19%	159MiB / 7.684GiB	2.02%	9.55kB / 9.65kB	55.59B / 680kB	■	
11	00b3dc1eb4ce	alamAPI	0.17%	45.7MiB / 7.684GiB	0.58%	9.07kB / 5.64kB	28.29B / 0B	3	
12	47dba53c6237	alamPREPROCESSOR	0.01%	350.7MiB / 7.684GiB	4.53%	3.1kB / 2.65kB	283MB / 12.3kB	15	
13	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
14	d878d6975a9c	alamDB	0.21%	159MiB / 7.684GiB	2.02%	9.55kB / 9.65kB	55.59B / 680kB	■	
15	00b3dc1eb4ce	alamAPI	0.16%	45.7MiB / 7.684GiB	0.58%	9.07kB / 5.64kB	28.29B / 0B	3	
16	47dba53c6237	alamPREPROCESSOR	0.01%	350.7MiB / 7.684GiB	4.53%	3.1kB / 2.65kB	283MB / 12.3kB	15	
17	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
18	d878d6975a9c	alamDB	0.26%	159MiB / 7.684GiB	2.02%	9.75kB / 9.94kB	55.59B / 680kB	■	
19	00b3dc1eb4ce	alamAPI	0.23%	45.7MiB / 7.684GiB	0.58%	9.36kB / 5.83kB	28.29B / 0B	3	
20	47dba53c6237	alamPREPROCESSOR	0.01%	350.7MiB / 7.684GiB	4.53%	3.1kB / 2.65kB	283MB / 12.3kB	15	
21	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
22	d878d6975a9c	alamDB	0.22%	159MiB / 7.684GiB	2.02%	9.75kB / 9.94kB	55.59B / 696kB	■	
23	00b3dc1eb4ce	alamAPI	0.17%	45.7MiB / 7.684GiB	0.58%	9.36kB / 5.83kB	28.29B / 0B	3	
24	47dba53c6237	alamPREPROCESSOR	0.01%	350.7MiB / 7.684GiB	4.53%	3.1kB / 2.65kB	283MB / 12.3kB	15	
25	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
26	d878d6975a9c	alamDB	0.16%	159MiB / 7.684GiB	2.02%	9.75kB / 9.94kB	55.59B / 696kB	■	
27	00b3dc1eb4ce	alamAPI	0.19%	45.7MiB / 7.684GiB	0.58%	9.36kB / 5.83kB	28.29B / 0B	3	
28	47dba53c6237	alamPREPROCESSOR	0.01%	350.7MiB / 7.684GiB	4.53%	3.1kB / 2.65kB	283MB / 12.3kB	15	
29	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
30	d878d6975a9c	alamDB	0.21%	159MiB / 7.684GiB	2.02%	9.75kB / 9.94kB	55.59B / 696kB	■	
31	00b3dc1eb4ce	alamAPI	0.16%	45.7MiB / 7.684GiB	0.58%	9.36kB / 5.83kB	28.29B / 0B	3	
32	47dba53c6237	alamPREPROCESSOR	0.01%	350.7MiB / 7.684GiB	4.53%	3.1kB / 2.65kB	283MB / 12.3kB	15	
33	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
34	d878d6975a9c	alamDB	0.24%	159MiB / 7.684GiB	2.02%	9.75kB / 9.94kB	55.59B / 696kB	■	
35	00b3dc1eb4ce	alamAPI	0.17%	45.7MiB / 7.684GiB	0.58%	9.36kB / 5.83kB	28.29B / 0B	3	
36	47dba53c6237	alamPREPROCESSOR	0.01%	350.7MiB / 7.684GiB	4.53%	3.1kB / 2.65kB	283MB / 12.3kB	15	
37	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
38	d878d6975a9c	alamDB	0.25%	159MiB / 7.684GiB	2.02%	9.94kB / 10.2kB	55.59B / 696kB	■	
39	00b3dc1eb4ce	alamAPI	0.21%	45.7MiB / 7.684GiB	0.58%	9.65kB / 6.03kB	28.29B / 0B	3	
40	47dba53c6237	alamPREPROCESSOR	0.01%	350.7MiB / 7.684GiB	4.53%	3.1kB / 2.65kB	283MB / 12.3kB	15	
41	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
42	d878d6975a9c	alamDB	0.22%	159MiB / 7.684GiB	2.02%	9.94kB / 10.2kB	55.59B / 733kB	■	
43	00b3dc1eb4ce	alamAPI	0.16%	45.7MiB / 7.684GiB	0.58%	9.65kB / 6.03kB	28.29B / 0B	3	
44	47dba53c6237	alamPREPROCESSOR	0.01%	350.7MiB / 7.684GiB	4.53%	3.1kB / 2.65kB	283MB / 12.3kB	15	
45	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
7077	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
7078	d878d6975a9c	alamDB	0.17%	179.5MiB / 7.684GiB	2.28%	88.5kB / 114kB	55.59B / 6.55MB	29	
7079	00b3dc1eb4ce	alamAPI	0.16%	45.73MiB / 7.684GiB	0.58%	113kB / 76.4kB	28.29B / 0B	3	
7080	47dba53c6237	alamPREPROCESSOR	0.01%	350MiB / 7.684GiB	4.56%	3.38kB / 2.65kB	283MB / 12.3kB	15	
7081	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
7082	d878d6975a9c	alamDB	0.28%	179.5MiB / 7.684GiB	2.28%	88.5kB / 114kB	55.59B / 6.55MB	29	
7083	00b3dc1eb4ce	alamAPI	0.16%	45.73MiB / 7.684GiB	0.58%	113kB / 76.4kB	28.29B / 0B	3	
7084	47dba53c6237	alamPREPROCESSOR	0.01%	350MiB / 7.684GiB	4.56%	3.38kB / 2.65kB	283MB / 12.3kB	15	
7085	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
7086	d878d6975a9c	alamDB	0.22%	179.5MiB / 7.684GiB	2.28%	88.5kB / 114kB	55.59B / 6.55MB	29	
7087	00b3dc1eb4ce	alamAPI	0.14%	45.73MiB / 7.684GiB	0.58%	113kB / 76.4kB	28.29B / 0B	3	
7088	47dba53c6237	alamPREPROCESSOR	0.01%	350MiB / 7.684GiB	4.56%	3.38kB / 2.65kB	283MB / 12.3kB	15	
7089	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
7090	d878d6975a9c	alamDB	0.23%	179.5MiB / 7.684GiB	2.28%	88.5kB / 114kB	55.59B / 6.55MB	29	
7091	00b3dc1eb4ce	alamAPI	0.18%	45.73MiB / 7.684GiB	0.58%	113kB / 76.4kB	28.29B / 0B	3	
7092	47dba53c6237	alamPREPROCESSOR	0.01%	350MiB / 7.684GiB	4.56%	3.38kB / 2.65kB	283MB / 12.3kB	15	
7093	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
7094	d878d6975a9c	alamDB	0.24%	179.5MiB / 7.684GiB	2.28%	88.7kB / 114kB	55.59B / 6.57MB	29	
7095	00b3dc1eb4ce	alamAPI	0.23%	45.73MiB / 7.684GiB	0.58%	114kB / 76.6kB	28.29B / 0B	3	
7096	47dba53c6237	alamPREPROCESSOR	0.01%	350MiB / 7.684GiB	4.56%	3.38kB / 2.65kB	283MB / 12.3kB	15	
7097	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
7098	d878d6975a9c	alamDB	0.18%	179.5MiB / 7.684GiB	2.28%	88.7kB / 114kB	55.59B / 6.57MB	29	
7099	00b3dc1eb4ce	alamAPI	0.15%	45.73MiB / 7.684GiB	0.58%	114kB / 76.6kB	28.29B / 0B	3	
7100	47dba53c6237	alamPREPROCESSOR	0.01%	350MiB / 7.684GiB	4.56%	3.38kB / 2.65kB	283MB / 12.3kB	15	
7101	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS	
7102	d878d6975a9c	alamDB	0.23%	179.5MiB / 7.684GiB	2.28%	88.7kB / 114kB	55.59B / 6.57MB	29	
7103	00b3dc1eb4ce	alamAPI	0.15%	45.73MiB / 7.684GiB	0.58%	114kB / 76.6kB	28.29B / 0B	3	
7104	47dba53c6237	alamPREPROCESSOR	0.01%	350MiB / 7.684GiB	4.56%	3.38kB / 2.65kB	283MB / 12.3kB	15	

Figure B.54: Raw Logs of Idle System Statistics

1	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
2	e6af858e1766	alamPREPROCESSOR2	0.01%	294.6MiB / 7.684GiB	3.74%	3.45kB / 2.65kB	0B / 0B	8
3	f73c98947a50	alamAPI2	2.36%	44.24MiB / 7.684GiB	0.56%	60.1kB / 39.2kB	0B / 0B	3
4	daf25cdfc1f2	alamDB2	0.32%	122MiB / 7.684GiB	1.55%	42.5kB / 57.8kB	0B / 0B	29
5	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
6	e6af858e1766	alamPREPROCESSOR2	0.01%	294.6MiB / 7.684GiB	3.74%	3.45kB / 2.65kB	0B / 0B	8
7	f73c98947a50	alamAPI2	27.35%	45.62MiB / 7.684GiB	0.58%	286kB / 273kB	0B / 0B	12
8	daf25cdfc1f2	alamDB2	1.54%	122.4MiB / 7.684GiB	1.56%	64.8kB / 187kB	0B / 0B	36
9	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
10	e6af858e1766	alamPREPROCESSOR2	0.01%	294.6MiB / 7.684GiB	3.74%	3.45kB / 2.65kB	0B / 0B	8
11	f73c98947a50	alamAPI2	31.85%	45.65MiB / 7.684GiB	0.58%	531kB / 526kB	0B / 0B	12
12	daf25cdfc1f2	alamDB2	1.61%	122.4MiB / 7.684GiB	1.56%	85.1kB / 329kB	0B / 0B	36
13	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
14	e6af858e1766	alamPREPROCESSOR2	0.01%	294.6MiB / 7.684GiB	3.74%	3.45kB / 2.65kB	0B / 0B	8
15	f73c98947a50	alamAPI2	30.66%	45.96MiB / 7.684GiB	0.58%	800kB / 808kB	0B / 0B	12
16	daf25cdfc1f2	alamDB2	1.61%	122.4MiB / 7.684GiB	1.56%	188kB / 491kB	0B / 0B	36
17	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
18	e6af858e1766	alamPREPROCESSOR2	0.01%	294.6MiB / 7.684GiB	3.74%	3.45kB / 2.65kB	0B / 0B	8
19	f73c98947a50	alamAPI2	24.59%	45.96MiB / 7.684GiB	0.58%	1MB / 1.82MB	0B / 0B	12
20	daf25cdfc1f2	alamDB2	1.32%	122.4MiB / 7.684GiB	1.56%	125kB / 610kB	0B / 0B	36
21	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
22	e6af858e1766	alamPREPROCESSOR2	0.01%	294.6MiB / 7.684GiB	3.74%	3.45kB / 2.65kB	0B / 0B	8
23	f73c98947a50	alamAPI2	28.89%	45.96MiB / 7.684GiB	0.58%	1.23MB / 1.25MB	0B / 0B	12
24	daf25cdfc1f2	alamDB2	1.51%	122.4MiB / 7.684GiB	1.56%	145kB / 744kB	0B / 0B	36
25	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
26	e6af858e1766	alamPREPROCESSOR2	0.01%	294.6MiB / 7.684GiB	3.74%	3.45kB / 2.65kB	0B / 0B	8
27	f73c98947a50	alamAPI2	29.81%	45.98MiB / 7.684GiB	0.58%	1.5MB / 1.54MB	0B / 0B	11
28	daf25cdfc1f2	alamDB2	1.72%	122.4MiB / 7.684GiB	1.56%	169kB / 914kB	0B / 0B	36
29	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
30	e6af858e1766	alamPREPROCESSOR2	0.02%	294.6MiB / 7.684GiB	3.74%	3.45kB / 2.65kB	0B / 0B	8
31	f73c98947a50	alamAPI2	23.69%	45.93MiB / 7.684GiB	0.58%	1.68MB / 1.73MB	0B / 0B	11
32	daf25cdfc1f2	alamDB2	1.23%	122.4MiB / 7.684GiB	1.56%	184kB / 1.02MB	0B / 0B	36

• • •

6988	daf25cdfc1f2	alamDB2	0.36%	145.6MiB / 7.684GiB	1.85%	21.2MB / 178MB	0B / 0B	41
6989	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
6910	e6af858e1766	alamPREPROCESSOR2	0.01%	296.6MiB / 7.684GiB	3.77%	3.59kB / 2.65kB	0B / 0B	8
6911	f73c98947a50	alamAPI2	0.26%	45.76MiB / 7.684GiB	0.58%	273kB / 293kB	0B / 0B	4
6912	daf25cdfc1f2	alamDB2	0.71%	145.6MiB / 7.684GiB	1.85%	21.2MB / 178MB	0B / 0B	41
6913	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
6914	e6af858e1766	alamPREPROCESSOR2	0.01%	296.6MiB / 7.684GiB	3.77%	3.59kB / 2.65kB	0B / 0B	8
6915	f73c98947a50	alamAPI2	0.18%	45.76MiB / 7.684GiB	0.58%	273kB / 293kB	0B / 0B	4
6916	daf25cdfc1f2	alamDB2	0.19%	145.6MiB / 7.684GiB	1.85%	21.2MB / 178MB	0B / 0B	41
6917	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
6918	e6af858e1766	alamPREPROCESSOR2	0.01%	296.6MiB / 7.684GiB	3.77%	3.59kB / 2.65kB	0B / 0B	8
6919	f73c98947a50	alamAPI2	0.24%	45.76MiB / 7.684GiB	0.58%	273kB / 293kB	0B / 0B	4
6920	daf25cdfc1f2	alamDB2	0.49%	145.6MiB / 7.684GiB	1.85%	21.2MB / 178MB	0B / 0B	41
6921	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
6922	e6af858e1766	alamPREPROCESSOR2	0.02%	296.6MiB / 7.684GiB	3.77%	3.59kB / 2.65kB	0B / 0B	8
6923	f73c98947a50	alamAPI2	0.28%	45.76MiB / 7.684GiB	0.58%	273kB / 293kB	0B / 0B	4
6924	daf25cdfc1f2	alamDB2	0.31%	145.6MiB / 7.684GiB	1.85%	21.2MB / 178MB	0B / 0B	41
6925	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
6926	e6af858e1766	alamPREPROCESSOR2	0.01%	296.6MiB / 7.684GiB	3.77%	3.59kB / 2.65kB	0B / 0B	8
6927	f73c98947a50	alamAPI2	0.25%	45.76MiB / 7.684GiB	0.58%	273kB / 293kB	0B / 0B	4
6928	daf25cdfc1f2	alamDB2	0.28%	145.6MiB / 7.684GiB	1.85%	21.2MB / 178MB	0B / 0B	41
6929	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
6930	e6af858e1766	alamPREPROCESSOR2	0.01%	296.6MiB / 7.684GiB	3.77%	3.59kB / 2.65kB	0B / 0B	8
6931	f73c98947a50	alamAPI2	0.18%	45.76MiB / 7.684GiB	0.58%	273kB / 293kB	0B / 0B	4
6932	daf25cdfc1f2	alamDB2	0.46%	145.6MiB / 7.684GiB	1.85%	21.2MB / 178MB	0B / 0B	41

Figure B.55: Raw Logs of Deployment System Statistics

1	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
2	f10bc4b28a7e	alamDB	0.21%	160.1MiB / 7.684GiB	2.04%	20.4kB / 25kB	54.4MB / 1.99MB	29
3	650af2d6431c	alamPREPROCESSOR	1.63%	611.4MiB / 7.684GiB	7.77%	96.5kB / 7.92kB	352MB / 49.2kB	33
4	8302c7cefe70	alamAPI	0.21%	45.71MiB / 7.684GiB	0.58%	25kB / 16.1kB	28.5MB / 0B	3
5	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
6	f10bc4b28a7e	alamDB	0.26%	160.1MiB / 7.684GiB	2.04%	20.4kB / 25kB	54.4MB / 1.99MB	29
7	650af2d6431c	alamPREPROCESSOR	0.16%	611.4MiB / 7.684GiB	7.77%	124kB / 8.39kB	352MB / 49.2kB	33
8	8302c7cefe70	alamAPI	0.26%	45.71MiB / 7.684GiB	0.58%	25kB / 16.1kB	28.5MB / 0B	3
9	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
10	f10bc4b28a7e	alamDB	0.28%	160.1MiB / 7.684GiB	2.04%	20.4kB / 25kB	54.4MB / 1.99MB	29
11	650af2d6431c	alamPREPROCESSOR	0.75%	611.4MiB / 7.684GiB	7.77%	201kB / 11.7kB	352MB / 385kB	33
12	8302c7cefe70	alamAPI	0.19%	45.71MiB / 7.684GiB	0.58%	25kB / 16.1kB	28.5MB / 0B	3
13	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
14	f10bc4b28a7e	alamDB	0.41%	160.1MiB / 7.684GiB	2.04%	20.4kB / 25kB	54.4MB / 1.99MB	29
15	650af2d6431c	alamPREPROCESSOR	2.86%	611.5MiB / 7.684GiB	7.77%	266kB / 14.3kB	352MB / 385kB	33
16	8302c7cefe70	alamAPI	0.19%	45.71MiB / 7.684GiB	0.58%	25kB / 16.1kB	28.5MB / 0B	3
17	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
18	f10bc4b28a7e	alamDB	0.69%	160.2MiB / 7.684GiB	2.04%	20.6kB / 25.3kB	54.4MB / 2.01MB	29
19	650af2d6431c	alamPREPROCESSOR	0.01%	611.5MiB / 7.684GiB	7.77%	313kB / 15.4kB	352MB / 385kB	33
20	8302c7cefe70	alamAPI	0.22%	45.71MiB / 7.684GiB	0.58%	25.3kB / 16.3kB	28.5MB / 0B	3
21	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
22	f10bc4b28a7e	alamDB	0.18%	160.2MiB / 7.684GiB	2.04%	20.6kB / 25.3kB	54.4MB / 2.01MB	29
23	650af2d6431c	alamPREPROCESSOR	0.47%	611.5MiB / 7.684GiB	7.77%	359kB / 17.8kB	352MB / 385kB	33
24	8302c7cefe70	alamAPI	0.19%	45.71MiB / 7.684GiB	0.58%	25.3kB / 16.3kB	28.5MB / 0B	3
25	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
26	f10bc4b28a7e	alamDB	0.36%	160.1MiB / 7.684GiB	2.04%	20.6kB / 25.3kB	54.4MB / 2.01MB	29
27	650af2d6431c	alamPREPROCESSOR	1.13%	611.5MiB / 7.684GiB	7.77%	442kB / 21.5kB	352MB / 385kB	33
28	8302c7cefe70	alamAPI	0.28%	45.71MiB / 7.684GiB	0.58%	25.3kB / 16.3kB	28.5MB / 0B	3
29	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
30	f10bc4b28a7e	alamDB	0.30%	160.1MiB / 7.684GiB	2.03%	20.6kB / 25.3kB	54.4MB / 2.01MB	29
31	650af2d6431c	alamPREPROCESSOR	0.91%	611.6MiB / 7.684GiB	7.77%	534kB / 25.3kB	352MB / 385kB	33
32	8302c7cefe70	alamAPI	0.19%	45.71MiB / 7.684GiB	0.58%	25.3kB / 16.3kB	28.5MB / 0B	3
33	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
34	f10bc4b28a7e	alamDB	0.26%	160.1MiB / 7.684GiB	2.03%	20.6kB / 25.3kB	54.4MB / 2.01MB	29

• • •

9456	8302c7cefe70	alamAPI	0.18%	45.72MiB / 7.684GiB	0.58%	165kB / 111kB	28.5MB / 0B	3
9457	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
9458	f10bc4b28a7e	alamDB	0.28%	192MiB / 7.684GiB	2.44%	1.13MB / 587kB	54.5MB / 24.9MB	29
9459	650af2d6431c	alamPREPROCESSOR	1.56%	1.017GiB / 7.684GiB	13.24%	154MB / 7.51MB	354MB / 527MB	33
9460	8302c7cefe70	alamAPI	0.20%	45.72MiB / 7.684GiB	0.58%	165kB / 111kB	28.5MB / 0B	3
9461	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
9462	f10bc4b28a7e	alamDB	0.31%	192MiB / 7.684GiB	2.44%	1.13MB / 587kB	54.5MB / 24.9MB	29
9463	650af2d6431c	alamPREPROCESSOR	2.17%	1.017GiB / 7.684GiB	13.24%	154MB / 7.51MB	354MB / 527MB	33
9464	8302c7cefe70	alamAPI	0.21%	45.72MiB / 7.684GiB	0.58%	165kB / 111kB	28.5MB / 0B	3
9465	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
9466	f10bc4b28a7e	alamDB	0.27%	192MiB / 7.684GiB	2.44%	1.13MB / 588kB	54.5MB / 24.9MB	29
9467	650af2d6431c	alamPREPROCESSOR	0.48%	1.017GiB / 7.684GiB	13.24%	154MB / 7.52MB	354MB / 527MB	33
9468	8302c7cefe70	alamAPI	0.26%	45.73MiB / 7.684GiB	0.58%	165kB / 111kB	28.5MB / 0B	3
9469	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
9470	f10bc4b28a7e	alamDB	0.14%	192MiB / 7.684GiB	2.44%	1.13MB / 588kB	54.5MB / 24.9MB	29
9471	650af2d6431c	alamPREPROCESSOR	98.62%	1.041GiB / 7.684GiB	13.54%	154MB / 7.52MB	354MB / 528MB	51
9472	8302c7cefe70	alamAPI	0.18%	45.72MiB / 7.684GiB	0.58%	165kB / 111kB	28.5MB / 0B	3
9473	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
9474	f10bc4b28a7e	alamDB	0.15%	192MiB / 7.684GiB	2.44%	1.13MB / 588kB	54.5MB / 24.9MB	29
9475	650af2d6431c	alamPREPROCESSOR	98.68%	1.054GiB / 7.684GiB	13.71%	154MB / 7.52MB	354MB / 528MB	51
9476	8302c7cefe70	alamAPI	0.18%	45.72MiB / 7.684GiB	0.58%	165kB / 111kB	28.5MB / 0B	3
9477	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
9478	f10bc4b28a7e	alamDB	0.19%	192MiB / 7.684GiB	2.44%	1.13MB / 588kB	54.5MB / 24.9MB	29
9479	650af2d6431c	alamPREPROCESSOR	98.17%	1.065GiB / 7.684GiB	13.86%	154MB / 7.52MB	354MB / 528MB	51
9480	8302c7cefe70	alamAPI	0.14%	45.72MiB / 7.684GiB	0.58%	165kB / 111kB	28.5MB / 0B	3

Figure B.56: Raw Logs of Data Collector Module (DCM) System Statistics

1	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
2	f10bc4b28a7e	alamDB	0.15%	192MiB / 7.684GiB	2.44%	1.23MB / 623kB	54.5MB / 33.2MB	29
3	658af2d6431c	alamPREPROCESSOR	98.18%	1.055GiB / 7.684GiB	13.73%	154MB / 7.54MB	354MB / 528MB	51
4	8302c7cefe70	alamAPI	0.10%	45.91MiB / 7.684GiB	0.58%	275kB / 192kB	28.5MB / 0B	3
5	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
6	f10bc4b28a7e	alamDB	0.14%	192MiB / 7.684GiB	2.44%	1.23MB / 623kB	54.5MB / 33.2MB	29
7	658af2d6431c	alamPREPROCESSOR	97.82%	1.063GiB / 7.684GiB	13.84%	154MB / 7.54MB	354MB / 528MB	51
8	8302c7cefe70	alamAPI	0.10%	45.91MiB / 7.684GiB	0.58%	275kB / 192kB	28.5MB / 0B	3
9	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
10	f10bc4b28a7e	alamDB	0.16%	192MiB / 7.684GiB	2.44%	1.23MB / 624kB	54.5MB / 33.2MB	29
11	658af2d6431c	alamPREPROCESSOR	98.16%	1.075GiB / 7.684GiB	14.00%	154MB / 7.54MB	354MB / 528MB	59
12	8302c7cefe70	alamAPI	0.14%	45.91MiB / 7.684GiB	0.58%	275kB / 192kB	28.5MB / 0B	3
13	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
14	f10bc4b28a7e	alamDB	0.21%	192MiB / 7.684GiB	2.44%	1.24MB / 627kB	54.5MB / 33.2MB	29
15	658af2d6431c	alamPREPROCESSOR	168.69%	930.9MiB / 7.684GiB	11.83%	154MB / 7.55MB	354MB / 528MB	26
16	8302c7cefe70	alamAPI	0.12%	45.91MiB / 7.684GiB	0.58%	275kB / 192kB	28.5MB / 0B	3
17	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
18	f10bc4b28a7e	alamDB	0.16%	192MiB / 7.684GiB	2.44%	1.24MB / 627kB	54.5MB / 33.2MB	29
19	658af2d6431c	alamPREPROCESSOR	122.66%	1.058GiB / 7.684GiB	13.77%	154MB / 7.55MB	354MB / 528MB	51
20	8302c7cefe70	alamAPI	0.11%	45.91MiB / 7.684GiB	0.58%	275kB / 192kB	28.5MB / 0B	3
21	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
22	f10bc4b28a7e	alamDB	0.16%	192MiB / 7.684GiB	2.44%	1.24MB / 627kB	54.5MB / 33.2MB	29
23	658af2d6431c	alamPREPROCESSOR	98.37%	1.071GiB / 7.684GiB	13.94%	154MB / 7.55MB	354MB / 528MB	62
24	8302c7cefe70	alamAPI	0.10%	45.91MiB / 7.684GiB	0.58%	275kB / 192kB	28.5MB / 0B	3
25	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
26	f10bc4b28a7e	alamDB	0.14%	192MiB / 7.684GiB	2.44%	1.24MB / 627kB	54.5MB / 33.2MB	29
27	658af2d6431c	alamPREPROCESSOR	96.23%	1.083GiB / 7.684GiB	14.09%	154MB / 7.55MB	354MB / 528MB	59
28	8302c7cefe70	alamAPI	0.13%	45.91MiB / 7.684GiB	0.58%	275kB / 192kB	28.5MB / 0B	3
29	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
30	f10bc4b28a7e	alamDB	0.38%	192MiB / 7.684GiB	2.44%	1.25MB / 631kB	54.5MB / 33.3MB	29
31	658af2d6431c	alamPREPROCESSOR	162.19%	921.1MiB / 7.684GiB	11.71%	154MB / 7.56MB	354MB / 528MB	26
32	8302c7cefe70	alamAPI	0.21%	45.91MiB / 7.684GiB	0.58%	275kB / 192kB	28.5MB / 0B	3

• • •

1632	8302c7cefe70	alamAPI	0.13%	45.7MiB / 7.684GiB	0.58%	299kB / 288kB	28.5MB / 0B	3
1633	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
1634	f10bc4b28a7e	alamDB	0.12%	189.7MiB / 7.684GiB	2.41%	2.25MB / 987kB	54.5MB / 38.1MB	29
1635	658af2d6431c	alamPREPROCESSOR	96.58%	1.361B / 7.684GiB	16.92%	154MB / 8.59MB	356MB / 538MB	51
1636	8302c7cefe70	alamAPI	0.10%	45.7MiB / 7.684GiB	0.58%	299kB / 288kB	28.5MB / 0B	3
1637	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
1638	f10bc4b28a7e	alamDB	0.16%	189.7MiB / 7.684GiB	2.41%	2.26MB / 990kB	54.5MB / 38.1MB	29
1639	658af2d6431c	alamPREPROCESSOR	177.58%	1.168GiB / 7.684GiB	15.28%	154MB / 8.56MB	356MB / 538MB	26
1640	8302c7cefe70	alamAPI	0.12%	45.7MiB / 7.684GiB	0.58%	299kB / 288kB	28.5MB / 0B	3
1641	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
1642	f10bc4b28a7e	alamDB	0.37%	189.7MiB / 7.684GiB	2.41%	2.26MB / 990kB	54.5MB / 38.1MB	29
1643	658af2d6431c	alamPREPROCESSOR	99.58%	1.284GiB / 7.684GiB	16.71%	154MB / 8.56MB	356MB / 538MB	51
1644	8302c7cefe70	alamAPI	0.15%	45.7MiB / 7.684GiB	0.58%	299kB / 288kB	28.5MB / 0B	3
1645	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
1646	f10bc4b28a7e	alamDB	0.17%	189.7MiB / 7.684GiB	2.41%	2.26MB / 990kB	54.5MB / 38.2MB	29
1647	658af2d6431c	alamPREPROCESSOR	99.86%	1.294GiB / 7.684GiB	16.84%	154MB / 8.56MB	356MB / 538MB	51
1648	8302c7cefe70	alamAPI	0.10%	45.7MiB / 7.684GiB	0.58%	299kB / 288kB	28.5MB / 0B	3
1649	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
1650	f10bc4b28a7e	alamDB	0.18%	189.5MiB / 7.684GiB	2.41%	2.26MB / 991kB	54.5MB / 38.3MB	29
1651	658af2d6431c	alamPREPROCESSOR	100.66%	1.301GiB / 7.684GiB	16.93%	154MB / 8.56MB	356MB / 538MB	51
1652	8302c7cefe70	alamAPI	0.15%	45.69MiB / 7.684GiB	0.58%	299kB / 288kB	28.5MB / 0B	3
1653	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
1654	f10bc4b28a7e	alamDB	0.46%	189.5MiB / 7.684GiB	2.41%	2.27MB / 994kB	54.5MB / 38.4MB	29
1655	658af2d6431c	alamPREPROCESSOR	40.26%	1.064GiB / 7.684GiB	13.85%	154MB / 8.57MB	356MB / 538MB	17
1656	8302c7cefe70	alamAPI	0.12%	45.68MiB / 7.684GiB	0.58%	299kB / 288kB	28.5MB / 0B	3

Figure B.57: Raw Logs of Data Processor Module (DPM) System Statistics

1	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
2	f10bc4b28a7e	alamDB	0.22%	189.4MiB / 7.684GiB	2.41%	2.3MB / 1.039B	54.5MB / 49.6MB	31
3	658af2d6431c	alamPREPROCESSOR	2.60%	1.383GiB / 7.684GiB	16.96%	156MB / 8.659B	359MB / 539MB	53
4	8302c7cefe70	alamAPI	0.17%	45.67MiB / 7.684GiB	0.58%	322kB / 224kB	28.5MB / 0B	3
5	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
6	f10bc4b28a7e	alamDB	0.22%	189.4MiB / 7.684GiB	2.41%	2.3MB / 1.039B	54.5MB / 49.6MB	31
7	658af2d6431c	alamPREPROCESSOR	3.44%	1.383GiB / 7.684GiB	16.96%	156MB / 8.659B	359MB / 539MB	53
8	8302c7cefe70	alamAPI	0.18%	45.67MiB / 7.684GiB	0.58%	322kB / 224kB	28.5MB / 0B	3
9	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
10	f10bc4b28a7e	alamDB	0.30%	189.4MiB / 7.684GiB	2.41%	2.3MB / 1.039B	54.5MB / 49.6MB	31
11	658af2d6431c	alamPREPROCESSOR	2.87%	1.383GiB / 7.684GiB	16.95%	156MB / 8.669B	359MB / 537MB	53
12	8302c7cefe70	alamAPI	0.21%	45.67MiB / 7.684GiB	0.58%	322kB / 224kB	28.5MB / 0B	3
13	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
14	f10bc4b28a7e	alamDB	0.26%	189.4MiB / 7.684GiB	2.41%	2.3MB / 1.039B	54.5MB / 49.6MB	31
15	658af2d6431c	alamPREPROCESSOR	1.33%	1.383GiB / 7.684GiB	16.95%	156MB / 8.669B	359MB / 537MB	53
16	8302c7cefe70	alamAPI	0.18%	45.67MiB / 7.684GiB	0.58%	322kB / 224kB	28.5MB / 0B	3
17	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
18	f10bc4b28a7e	alamDB	0.30%	189.4MiB / 7.684GiB	2.41%	2.3MB / 1.039B	54.5MB / 49.6MB	31
19	658af2d6431c	alamPREPROCESSOR	2.47%	1.382GiB / 7.684GiB	16.94%	156MB / 8.679B	359MB / 537MB	53
20	8302c7cefe70	alamAPI	0.19%	45.67MiB / 7.684GiB	0.58%	322kB / 224kB	28.5MB / 0B	3
21	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
22	f10bc4b28a7e	alamDB	0.21%	189.4MiB / 7.684GiB	2.41%	2.3MB / 1.039B	54.5MB / 49.6MB	31
23	658af2d6431c	alamPREPROCESSOR	1.58%	1.382GiB / 7.684GiB	16.94%	156MB / 8.679B	359MB / 538MB	53
24	8302c7cefe70	alamAPI	0.13%	45.67MiB / 7.684GiB	0.58%	322kB / 224kB	28.5MB / 0B	3
25	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
26	f10bc4b28a7e	alamDB	0.51%	189.4MiB / 7.684GiB	2.41%	2.3MB / 1.039B	54.5MB / 49.6MB	31
27	658af2d6431c	alamPREPROCESSOR	1.88%	1.361B / 7.684GiB	16.92%	156MB / 8.679B	359MB / 538MB	53
28	8302c7cefe70	alamAPI	0.26%	45.67MiB / 7.684GiB	0.58%	322kB / 224kB	28.5MB / 0B	3
29	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
30	f10bc4b28a7e	alamDB	0.45%	189.4MiB / 7.684GiB	2.41%	2.3MB / 1.039B	54.5MB / 49.6MB	31
31	658af2d6431c	alamPREPROCESSOR	4.15%	1.361B / 7.684GiB	16.92%	157MB / 8.689B	359MB / 538MB	53
32	8302c7cefe70	alamAPI	0.22%	45.67MiB / 7.684GiB	0.58%	322kB / 224kB	28.5MB / 0B	3
33	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS

• • •

8139	658af2d6431c	alamPREPROCESSOR	2.46%	1.135GiB / 7.684GiB	14.77%	309MB / 15.2MB	359MB / 1.06GB	53
8140	8302c7cefe70	alamAPI	0.26%	45.51MiB / 7.684GiB	0.58%	442kB / 305kB	28.5MB / 0B	3
8141	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
8142	f10bc4b28a7e	alamDB	0.38%	186.9MiB / 7.684GiB	2.38%	2.49MB / 1.27MB	54.5MB / 49.7MB	31
8143	658af2d6431c	alamPREPROCESSOR	0.28%	1.135GiB / 7.684GiB	14.77%	308MB / 15.2MB	359MB / 1.06GB	53
8144	8302c7cefe70	alamAPI	0.27%	45.51MiB / 7.684GiB	0.58%	442kB / 305kB	28.5MB / 0B	3
8145	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
8146	f10bc4b28a7e	alamDB	0.55%	186.9MiB / 7.684GiB	2.38%	2.49MB / 1.27MB	54.5MB / 49.7MB	31
8147	658af2d6431c	alamPREPROCESSOR	0.50%	1.135GiB / 7.684GiB	14.77%	309MB / 15.2MB	359MB / 1.06GB	53
8148	8302c7cefe70	alamAPI	0.15%	45.51MiB / 7.684GiB	0.58%	442kB / 305kB	28.5MB / 0B	3
8149	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
8150	f10bc4b28a7e	alamDB	0.28%	186.9MiB / 7.684GiB	2.38%	2.49MB / 1.27MB	54.5MB / 49.7MB	31
8151	658af2d6431c	alamPREPROCESSOR	2.51%	1.135GiB / 7.684GiB	14.77%	309MB / 15.2MB	359MB / 1.06GB	53
8152	8302c7cefe70	alamAPI	0.19%	45.51MiB / 7.684GiB	0.58%	442kB / 305kB	28.5MB / 0B	3
8153	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
8154	f10bc4b28a7e	alamDB	0.32%	186.9MiB / 7.684GiB	2.38%	2.49MB / 1.27MB	54.5MB / 49.7MB	31
8155	658af2d6431c	alamPREPROCESSOR	2.78%	1.135GiB / 7.684GiB	14.77%	309MB / 15.2MB	359MB / 1.06GB	53
8156	8302c7cefe70	alamAPI	0.16%	45.51MiB / 7.684GiB	0.58%	442kB / 305kB	28.5MB / 0B	3
8157	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
8158	f10bc4b28a7e	alamDB	0.31%	186.9MiB / 7.684GiB	2.38%	2.49MB / 1.27MB	54.5MB / 49.7MB	31
8159	658af2d6431c	alamPREPROCESSOR	3.63%	1.135GiB / 7.684GiB	14.77%	309MB / 15.2MB	359MB / 1.06GB	53
8160	8302c7cefe70	alamAPI	0.23%	45.51MiB / 7.684GiB	0.58%	443kB / 305kB	28.5MB / 0B	3
8161	CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
8162	f10bc4b28a7e	alamDB	0.72%	187MiB / 7.684GiB	2.38%	2.49MB / 1.27MB	54.5MB / 49.7MB	31
8163	658af2d6431c	alamPREPROCESSOR	4.19%	1.135GiB / 7.684GiB	14.77%	309MB / 15.2MB	359MB / 1.06GB	53
8164	8302c7cefe70	alamAPI	0.21%	45.51MiB / 7.684GiB	0.58%	443kB / 305kB	28.5MB / 0B	3

Figure B.58: Raw Logs of alamPREPROCESSOR System Statistics

B.5.2 PSEI Trading Baseline Data

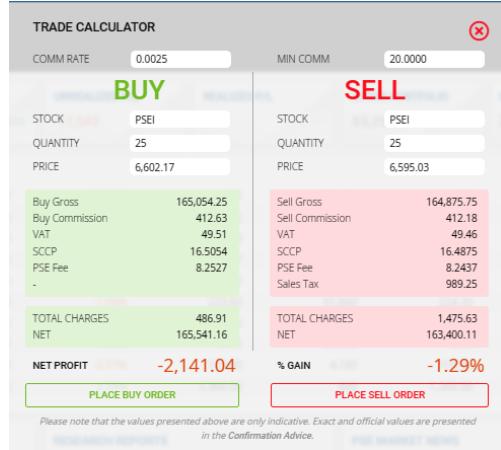


Figure B.59: Day 1 PSEI Trading Raw Data

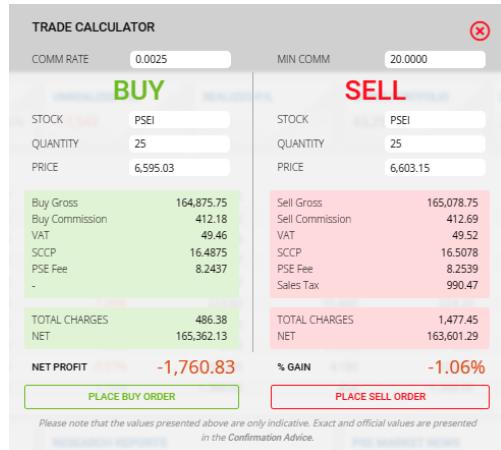


Figure B.60: Day 2 PSEI Trading Raw Data

TRADE CALCULATOR

BUY		SELL	
COMM RATE	0.0025	MIN COMM	20.0000
STOCK	PSEI	STOCK	PSEI
QUANTITY	25	QUANTITY	25
PRICE	6,603.15	PRICE	6,630.97
Buy Gross	165,078.75	Sell Gross	165,774.25
Buy Commission	412.69	Sell Commission	414.43
VAT	49.52	VAT	49.73
SCCP	16,5078	SCCP	16,5774
PSE Fee	8,2539	PSE Fee	8,2887
-		Sales Tax	994.64
TOTAL CHARGES	486.98	TOTAL CHARGES	1,483.67
NET	165,565.73	NET	164,290.57
NET PROFIT	-1,275.16	% GAIN	-0.77%
PLACE BUY ORDER		PLACE SELL ORDER	

Please note that the values presented above are only indicative. Exact and official values are presented in the Confirmation Advice.

Figure B.61: Day 3 PSEI Trading Raw Data

TRADE CALCULATOR

BUY		SELL	
COMM RATE	0.0025	MIN COMM	20.0000
STOCK	PSEI	STOCK	PSEI
QUANTITY	25	QUANTITY	25
PRICE	6,630.97	PRICE	6,644.75
Buy Gross	165,774.25	Sell Gross	166,118.75
Buy Commission	414.43	Sell Commission	415.29
VAT	49.73	VAT	49.83
SCCP	16,5774	SCCP	16,6118
PSE Fee	8,2887	PSE Fee	8,3059
-		Sales Tax	996.71
TOTAL CHARGES	489.03	TOTAL CHARGES	1,486.76
NET	166,263.28	NET	164,631.98
NET PROFIT	-1,631.29	% GAIN	-0.98%
PLACE BUY ORDER		PLACE SELL ORDER	

Please note that the values presented above are only indicative. Exact and official values are presented in the Confirmation Advice.

Figure B.62: Day 4 PSEI Trading Raw Data

TRADE CALCULATOR

COMM RATE	0.0025	MIN COMM	20.0000
BUY		SELL	
STOCK	PSEI	STOCK	PSEI
QUANTITY	25	QUANTITY	25
PRICE	6,644.75	PRICE	6,499.68
Buy Gross	166,118.75	Sell Gross	162,492.00
Buy Commission	415.29	Sell Commission	406.23
VAT	49.83	VAT	48.74
SCCP	16,611.8	SCCP	16,249.2
PSE Fee	8.3059	PSE Fee	8.1246
-		Sales Tax	974.95
TOTAL CHARGES	490.05	TOTAL CHARGES	1,454.30
NET	166,608.80	NET	161,037.69
NET PROFIT	-5,571.10	% GAIN	-3.34%
PLACE BUY ORDER		PLACE SELL ORDER	

Please note that the values presented above are only indicative. Exact and official values are presented in the Confirmation Advice.

Figure B.63: Day 5 PSEI Trading Raw Data

TRADE CALCULATOR

COMM RATE	0.0025	MIN COMM	20.0000
BUY		SELL	
STOCK	PSEI	STOCK	PSEI
QUANTITY	25	QUANTITY	25
PRICE	6,499.68	PRICE	6,529.99
Buy Gross	162,492.00	Sell Gross	163,249.75
Buy Commission	406.23	Sell Commission	408.12
VAT	48.74	VAT	48.97
SCCP	16,249.2	SCCP	16,324.9
PSE Fee	8.1246	PSE Fee	8.1624
-		Sales Tax	979.49
TOTAL CHARGES	479.35	TOTAL CHARGES	1,461.08
NET	162,971.35	NET	161,788.66
NET PROFIT	-1,182.68	% GAIN	-0.72%
PLACE BUY ORDER		PLACE SELL ORDER	

Please note that the values presented above are only indicative. Exact and official values are presented in the Confirmation Advice.

Figure B.64: Day 6 PSEI Trading Raw Data

TRADE CALCULATOR

BUY		SELL	
STOCK	PSEI	STOCK	PSEI
QUANTITY	25	QUANTITY	25
PRICE	6,529.99	PRICE	6,472.04
Buy Gross	163,249.75	Sell Gross	161,801.00
Buy Commission	408.12	Sell Commission	404.50
VAT	48.97	VAT	48.54
SCCP	16.3249	SCCP	16.1801
PSE Fee	8.1624	PSE Fee	8.0900
-		Sales Tax	970.80
TOTAL CHARGES	481.58	TOTAL CHARGES	1,448.11
NET	163,731.33	NET	160,352.88
NET PROFIT	-3,378.45	% GAIN	-2.06%
PLACE BUY ORDER		PLACE SELL ORDER	

Please note that the values presented above are only indicative. Exact and official values are presented in the Confirmation Advice.

Figure B.65: Day 7 PSEI Trading Raw Data

TRADE CALCULATOR

BUY		SELL	
STOCK	PSEI	STOCK	PSEI
QUANTITY	25	QUANTITY	25
PRICE	6,472.04	PRICE	6,488.51
Buy Gross	161,801.00	Sell Gross	162,212.75
Buy Commission	404.50	Sell Commission	405.53
VAT	48.54	VAT	48.66
SCCP	16.1801	SCCP	16.2212
PSE Fee	8.0900	PSE Fee	8.1106
-		Sales Tax	973.27
TOTAL CHARGES	477.31	TOTAL CHARGES	1,451.80
NET	162,278.31	NET	160,760.94
NET PROFIT	-1,517.36	% GAIN	-0.93%
PLACE BUY ORDER		PLACE SELL ORDER	

Please note that the values presented above are only indicative. Exact and official values are presented in the Confirmation Advice.

Figure B.66: Day 8 PSEI Trading Raw Data

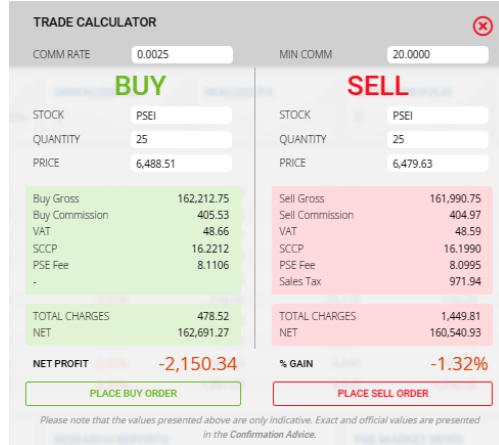


Figure B.67: Day 9 PSEI Trading Raw Data

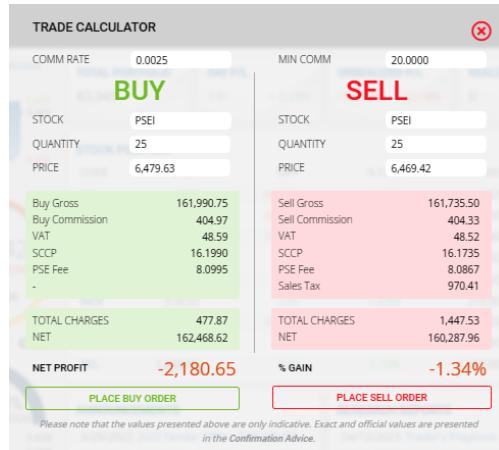


Figure B.68: Day 10 PSEI Trading Raw Data

B.5.3 Raw Real-world alamSYS Application

Figure B.69: Real World Application Raw Data Logs

B.6 Gantt Chart

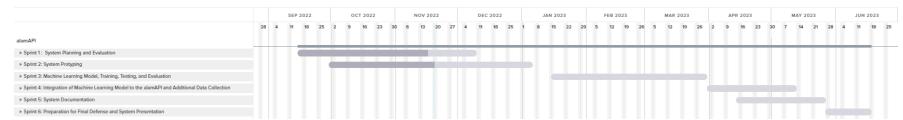


Figure B.70: Full Gantt Chart

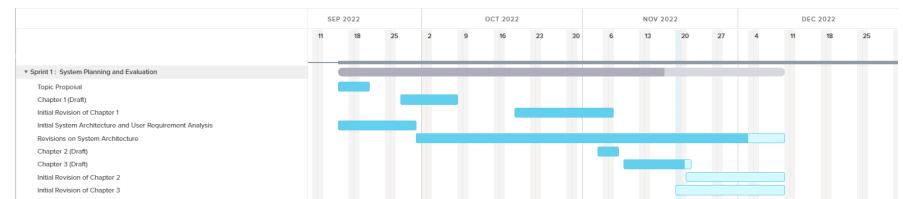


Figure B.71: Gantt Chart for Sprint 1

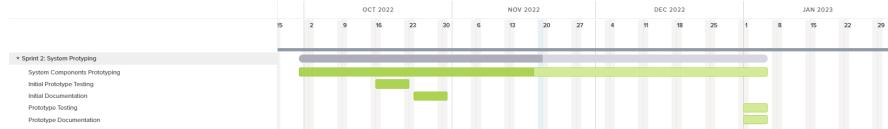


Figure B.72: Gantt Chart for Sprint 2

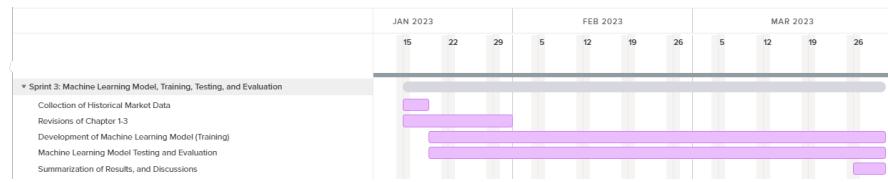


Figure B.73: Gantt Chart for Sprint 3

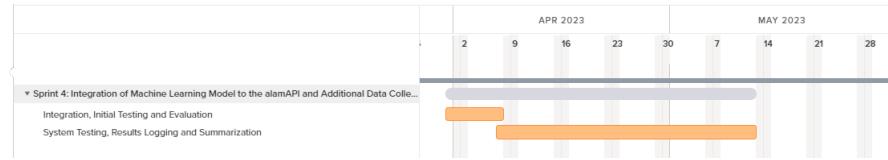


Figure B.74: Gantt Chart for Sprint 4

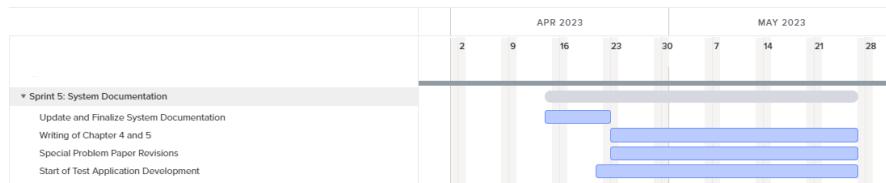


Figure B.75: Gantt Chart for Sprint 5

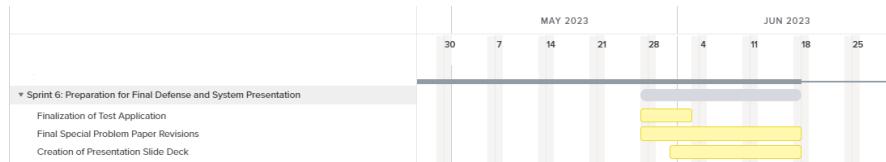


Figure B.76: Gantt Chart for Sprint 6

Appendix C

Acknowledgements

The author is grateful for the chance to express his sincere gratitude to the following individuals for their assistance in making this Special Problem a success:

First of all, the author would like to express their sincere gratitude to oneself for their perseverance and dedication throughout the journey.

Also, his sincere appreciation goes out to Sir Nilo C. Araneta, the Special Problem Adviser, for his invaluable advice, support, and insights. His guidance and knowledge were very helpful in getting this Special Problem finished.

In addition, the author would also like to thank Ma'am Ara Abigail E. Ambita for sharing her knowledge on machine learning. The author is appreciative that she gave him the chance to learn from her, and his work has become much better as a result of her advice and insights.

The author is sincerely grateful to his family for their financial assistance. The author could not have done this without their support, which has been a constant source of inspiration and motivation throughout the journey.

Lastly, the author concludes by thanking God for his grace and blessings, without which this accomplishment would not have been possible. The author has found strength and inspiration in the ever-present presence and direction of God.

Appendix D

Author's Contact Information

The author is open for collaborations and additional conversation in regards to the topics discussed in the development of this Special Problem. Where, the following contact information is given to make it easier for anyone to communicate with the author in the future.

Name:	John Markton M. Olarte
Emails:	jmolarte@up.edu.ph; markton.operation@gmail.com
Contact Number:	+639 2180 10551

If you have any questions about this Special Problem or would like to collaborate on it, please feel free to contact the author using any of the above contact details. The author is eager to discuss ideas for more in-depth study and advancement in this area with interested parties.