iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.79068

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

INVESTO: Simplifying Stock Market Analytics through Interactive Visualization

Prathamesh Dhage[1], Rohan Nagoankar[2], Aditya Mohite[3], Dr. Sandeep Kulkarni[4]
School of Engineering Ajeenkya DY Patil University

Abstract: ***The rapid expansion of global financial markets and the increasing participation of retail investors have created a strong demand for accessible and interpretable stock market analytics tools. Despite the availability of large volumes of financial data, extracting meaningful insights from raw market information remains challenging for students, novice investors, and small-scale market participants due to the complexity, technical requirements, and high costs associated with many professional trading platforms. To address this issue, this paper presents INVESTO, a web-based interactive stock market analytics and visualization platform designed to simplify financial data exploration and support informed investment decision-making. The platform is developed using Python and the Streamlit framework and provides an intuitive interface that enables users to access both historical and real-time stock market data without requiring programming knowledge or advanced financial expertise. INVESTO integrates financial data acquisition through external APIs, preprocessing and normalization of time-series data, and computation of key technical indicators such as Simple Moving Average (SMA), Exponential Moving Average (EMA), and volatility metrics. These analytical outputs are presented through interactive visualizations built using Plotly, allowing users to analyze price movements, detect trends, and compare stock performance dynamically. The system architecture follows a modular three-layer design consisting of the Data Acquisition Layer, Processing Layer, and Presentation Layer, ensuring scalability, maintainability, and efficient data flow. Cloud-based deployment through Streamlit Cloud enables platform accessibility through web browsers while maintaining efficient resource utilization and responsive performance. Performance optimizations such as session-based caching, vectorized computations, and selective rendering improve system responsiveness and reduce computational overhead. Security considerations focus on maintaining data integrity, validating external API responses, and preserving user privacy through a read-only analytical design that avoids storage of sensitive user information. Experimental evaluation and usability observations indicate that the platform improves comprehension of market trends, supports exploratory financial analysis, and enhances financial literacy for students and beginner investors. While INVESTO prioritizes interpretability and visualization rather than predictive modelling, the modular architecture allows future integration of advanced features such as machine learning-based forecasting, sentiment analysis, risk metrics, and portfolio simulation. Overall, the platform demonstrates the effectiveness of combining modern web technologies, cloud deployment, and interactive visualization techniques to bridge the gap between raw financial data and actionable analytical insights in educational financial technology systems.***

Index Terms: ***Stock market analytics, Financial data visualization, Real-time stock data, Historical trend analysis, Technical indicators, Moving averages such (SMA and EMA), Volatility analysis, Time-series data processing, , Data preprocessing, Stock performance comparison, Exploratory financial analysis, Investment decision support, and Financial literacy.***

I. INTRODUCTION

The ecosystems in which financial markets function are extremely dynamic, producing massive amounts of data continuously. Stock prices, trade volumes, market indexes, company performance metrics, and historical price trends generate complex time-series datasets that require structured analysis. Financial experts, policymakers, institutional investors, and individual investors rely heavily on this information when making investment decisions. However, extracting meaningful insights from raw market data remains challenging, particularly for students and novice investors who lack access to advanced analytical tools and structured financial education frameworks [3].

Traditional stock market analysis platforms are typically offered by brokerage firms or specialized financial software providers. While these platforms provide technical indicators, forecasting tools, and advanced analytical capabilities, they often require costly subscriptions, technical expertise, or prior financial knowledge. Consequently, they may not be easily accessible to beginners or students, limiting their usefulness in educational environments.

©IJRASET: All Rights are Reserved

Studies on online trading systems indicate that many existing platforms prioritize transaction execution and operational performance over simplified visualization and conceptual understanding [1], [2].

Recent advancements in web-based technologies and open-source software have created new opportunities for developing accessible financial analytics tools. Python has emerged as a powerful language for financial data analysis due to its extensive ecosystem of libraries for data manipulation, statistical modelling, and visualization [3]. Moreover, interactive visualization systems have been shown to significantly enhance user comprehension, pattern recognition, and confidence in decision-making compared to static numerical representations [11]. These developments enable the creation of user-friendly platforms that simplify financial data exploration without sacrificing analytical depth.

By integrating Python-based analytical libraries with modern web frameworks and financial data APIs, it becomes possible to build real-time stock market analysis applications that are both technically robust and intuitive to use. Research in financial technology highlights the importance of combining interpretability-focused visualization with scalable web architectures to support exploratory learning and informed decision-making [3], [11].

To address these challenges, this paper proposes INVESTO, an interactive web-based stock market analytics platform designed to simplify financial data exploration. Through an intuitive graphical interface, INVESTO provides access to historical and real-time stock data, enabling users to analyze key financial metrics, visualize price movements, and compare trends without requiring prior programming knowledge. By abstracting the complexities of data retrieval, preprocessing, and visualization, the platform reduces cognitive and technical barriers to financial analysis.

The primary objective of INVESTO is to improve accessibility to financial analytics through a free, cloud-deployable web application that emphasizes clarity, usability, and interpretability. Unlike algorithmic trading systems, the platform focuses on exploratory analysis and financial literacy rather than automated prediction. This approach aligns with research emphasizing the educational value of visualization-driven financial tools [1], [11]. This is how the rest of the paper is organized. A review of relevant research in the fields of web-based financial analytics and stock market visualization systems is provided in the following section. A thorough explanation of INVESTO's system architecture and implementation process follows. Experimental observations are then used to assess the platform's efficacy, performance, and usability. The study ends by outlining possible future improvements, such as the incorporation of predictive analytics and machine learning-based forecasting approaches, and providing insights into the shortcomings of the current system.

II. BACKGROUND AND THEORETICAL FOUNDATIONS

Stock market analysis has traditionally been divided into two main methods: fundamental analysis and technical analysis. Fundamental analysis seeks to assess a company's true value by examining its financial condition, the broader economic environment, its industry standing, and growth potential. This method involves a thorough review of financial documents such as income statements, balance sheets, and cash flow statements, along with valuation metrics like Price-to-Earnings (P/E), Price-to-Book (P/B), Return on Equity (ROE), and Earnings Per Share (EPS). The core idea behind fundamental analysis is that market prices will eventually align with intrinsic value over time, so differences between market price and intrinsic value present investment opportunities. This approach is considered crucial for making long-term investment decisions and evaluating risk, especially in stable economic settings [4], [5].

On the other hand, technical analysis is based on the belief that past price movements and trading volumes contain all necessary market information. It assumes that price trends form recognizable patterns that tend to repeat due to investor psychology and market behavior. Tools like Moving Averages, Relative Strength Index (RSI), Moving Average Convergence Divergence (MACD), and Bollinger Bands are used to identify trend directions, momentum changes, and volatility patterns. Unlike fundamental analysis, technical analysis does not try to assess intrinsic value but focuses on market behavior and price movements. Studies have found that technical indicators can offer short-term signals under certain market conditions, although their effectiveness is often affected by market efficiency and volatility [3], [8].

From a quantitative standpoint, time-series forecasting models have been widely used to model stock prices. The Auto Regressive Integrated Moving Average (ARIMA) model captures linear relationships and trends in stationary time-series data and is especially useful for short-term predictions when assumptions of linearity and stationarity are met. The Generalized Autoregressive Conditional Heteroskedasticity (GARCH) model enhances time-series analysis by accounting for volatility clustering—a common phenomenon in financial markets where large price changes tend to follow large changes, and small changes follow small ones. These econometric models have been fundamental in financial forecasting research and continue to be important for analyzing volatility and estimating risk [3], [9].

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

Recently, machine learning and deep learning techniques have attracted considerable interest in stock market forecasting. Support Vector Machines (SVM) can capture nonlinear relationships using kernel functions and have been effectively applied to classification and regression tasks with financial data. More sophisticated models like Long Short-Term Memory (LSTM) networks are designed to process sequential time-series data and can learn long-term dependencies in price sequences. These approaches overcome some limitations of traditional econometric models by capturing complex nonlinear dynamics. However, research shows that prediction accuracy varies depending on the market and dataset, and performance may decline in highly volatile or unpredictable markets. Financial markets are inherently affected by external shocks, geopolitical events, policy changes, and investor sentiment, making precise forecasting inherently challenging despite advances in modelling methods [6], [9].

Another key theoretical framework shaping stock analysis is the Efficient Market Hypothesis (EMH), which asserts that asset prices incorporate all available information at any moment. The semi-strong form of EMH suggests that consistently earning abnormal returns using only publicly accessible information is not feasible. This theory questions the effectiveness of both fundamental and technical forecasting methods in highly efficient markets. Although many empirical studies debate EMH's strict accuracy, its theoretical implications emphasize the need for cautious and transparent forecasting [3].

Due to the limitations of purely predictive methods, visualization-based analytical tools provide an alternative approach focused on interpretability rather than exact prediction. Interactive data visualization allows users to examine historical trends, compare different instruments' performance, and analyze volatility through graphical means. Research on exploratory information systems shows that interactive charts enhance user understanding, pattern detection, and confidence in decision-making compared to static numerical data. Visual tools such as moving averages, volatility bands, and trend comparisons help users interpret data in a more contextual and intuitive way [11].

The combined theoretical perspectives from fundamental and technical analysis, econometric modelling, machine learning forecasting, and exploratory visualization guide the design principles of INVESTO. Instead of relying solely on automated predictions, the system prioritizes clear visualization of trends, volatility, and comparative performance. This strategy supports the educational goal of improving financial literacy through interactive exploration while recognizing the inherent uncertainty and randomness of stock market behavior [1], [11].

III. LITERATURE REVIEW

Over the past twenty years, research in financial technology and stock market analysis has advanced considerably, especially with the rise of web-based platforms and machine learning methods. Initial studies on online virtual stock trading platforms revealed that interactive environments greatly improve students' financial knowledge and investment awareness. These platforms enable users to simulate real market scenarios using virtual funds, enhancing hands-on understanding of trading tactics and risk management. Empirical studies indicate that such tools positively affect user behavior by promoting more logical investment choices and better analytical skills. The inclusion of real-time data streams, financial ratio analysis, and performance tracking dashboards helps students link theoretical finance concepts with practical experience. Nonetheless, despite their educational advantages, most virtual trading platforms concentrate mainly on transaction simulation and portfolio monitoring rather than on intuitive visual analytics, which limits their effectiveness as exploratory visualization tools [1].

Additional research on developing online stock trading terminals has focused on system design, database integration, transaction handling, and order matching processes. These platforms mimic real trading systems and often feature modules for buying and selling, portfolio oversight, and historical transaction records. Although technically sound, these systems emphasize operational performance and execution logic over analytical visualization. Their design often resembles professional brokerage software, which can be too complex for novice investors. While these models successfully implement client-server architectures and database-driven frameworks, they typically lack simplified visual dashboards designed for educational purposes and conceptual understanding [2].

The field of stock market forecasting has been thoroughly reviewed in recent scholarly work. Current surveys classify prediction methods into four main categories: statistical time-series models, machine learning techniques, hybrid approaches, and sentiment-based models. Statistical models like ARIMA are commonly applied to capture linear relationships and short-term patterns in financial time series. GARCH models build on this by modelling volatility clustering, a frequent occurrence in financial markets. These models are mathematically rigorous and computationally efficient but may have difficulty representing nonlinear market dynamics [3].

Machine learning methods overcome some drawbacks of traditional econometric models by detecting nonlinear patterns in complex financial datasets. Techniques like Support Vector Machines (SVM), Artificial Neural Networks (ANN), and Random Forests have been utilized for classification and regression tasks in stock prediction.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

Deep learning models, particularly Long Short-Term Memory (LSTM) networks, have demonstrated superior ability to capture temporal dependencies in sequential financial data. LSTMs excel at managing long-term dependencies and changing patterns in price movements. Research shows that hybrid models combining ARIMA and SVM can improve forecasting accuracy by integrating both linear and nonlinear approaches. Likewise, LSTM-based models have yielded promising results in controlled experiments [6], [9].

Despite improvements in prediction accuracy, these models face practical challenges. Financial markets are affected by random processes, macroeconomic changes, geopolitical events, and behavioral biases that are difficult to model precisely. High-frequency data can be noisy and prone to overfitting, necessitating careful parameter tuning and validation. Moreover, machine learning models require significant computational power, extensive data preprocessing, and hyperparameter tuning, which may not be suitable for beginner-focused educational platforms. Thus, although predictive modelling remains a vibrant research area, its complexity limits accessibility for novice investors who seek conceptual understanding rather than detailed algorithmic forecasts [3], [6].

Sentiment analysis is another growing area in stock market research. By examining financial news, social media, and public sentiment, researchers aim to incorporate qualitative external factors into quantitative models. Natural Language Processing (NLP) techniques transform text into sentiment scores that can influence price predictions. While sentiment analysis can improve short-term forecasts, it demands large-scale data processing and sophisticated text analysis frameworks, increasing computational requirements and implementation complexity [3].

Alongside predictive research, financial visualization studies highlight the importance of interactive dashboards and graphical tools in decision-making. Visualization systems facilitate exploratory analysis, enabling users to detect relationships, trends, and anomalies through dynamic chart interactions. Interactive visual interfaces have been found to enhance pattern recognition, reduce cognitive effort, and boost user confidence compared to static tables. Research indicates that combining multiple visual elements such as layered indicators, comparative performance graphs, and volatility charts supports more intuitive understanding of market behavior. These results emphasize the value of visualization-based systems in financial education and exploratory analysis [11].

The literature thus distinguishes clearly between predictive modelling tools and visualization-focused analytical platforms. While machine learning models improve forecasting, their complexity and interpretability issues make them less ideal for beginner educational tools. In contrast, interactive visualization systems offer transparency and ease of use but often lack real-time data integration and modular design. INVESTO aims to bridge this divide by combining interactive visualization, technical indicator calculations, and cloud-based access within a single web-based platform. The system aligns with research advocating for interpretability-focused financial tools while remaining extensible for future predictive enhancements [1], [3], [11].

IV. SYSTEM DESIGN AND ARCHITECTURE

The design of INVESTO follows a modular three-layer architecture to ensure scalability, maintainability, and extensibility. The architecture is structured into the Data Acquisition Layer, Processing Layer, and Presentation Layer. This separation of concerns allows each component to operate independently while maintaining seamless integration within the overall system. The layered approach also enhances system reliability and simplifies future enhancements, such as integration of predictive models or portfolio simulation features.

A. Data Acquisition Layer

The Data Acquisition Layer is tasked with fetching stock market information from publicly accessible financial APIs. These APIs deliver organized time-series data, including Open, High, Low, Close, and Volume (OHLCV) details. This layer standardizes the incoming data by converting timestamps into uniform datetime formats and arranging them in chronological order to preserve temporal accuracy. This process ensures the data is compatible with subsequent time-series analyses.

To boost performance and lower latency, caching strategies are employed. Stock data that is frequently requested is temporarily stored in memory using session-level caching methods. This approach avoids repeated API calls when users request the same stock data for identical time frames, thereby reducing network traffic, speeding up response times, and lessening the chance of hitting API rate limits. Moreover, this layer incorporates exception handling to manage incomplete data, network issues, or missing segments, maintaining system stability even when API responses are unpredictable.

Data validation is also a vital part of this layer. Before passing data to the processing module, checks are performed to ensure the dataset includes necessary columns, has correctly formatted timestamps, and contains non-null price values. If any inconsistencies are found, the system either tries to correct them through preprocessing or alerts the user via the interface. This proactive validation improves the system's reliability and the accuracy of analyses.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

B. Processing Layer

The Processing Layer serves as the computational heart of INVESTO. It handles tasks such as data transformation, cleaning, normalization, and analytical calculations. When structured time-series data is received, the system first manages missing values by applying forward-fill techniques for small gaps and removing extended missing periods to preserve the quality of analysis.

Normalization is applied during comparative analyses involving multiple stocks. Price series are reindexed based on a reference value to enable fair comparisons of performance trends. This process ensures that stocks with varying price levels can be visually compared without distortion.

Technical indicators are calculated using rolling window methods. Python's vectorized operations allow efficient processing even with large datasets. The Simple Moving Average (SMA) is calculated by averaging closing prices over specified windows, while the Exponential Moving Average (EMA) gives more weight to recent data points through exponentially decreasing weights. Rolling standard deviation is used to estimate short-term volatility, offering insights into the intensity of price fluctuations.

The Processing Layer's modular architecture enables the addition of new analytical features without altering the Presentation Layer. For example, future updates like the Relative Strength Index (RSI), Bollinger Bands, or machine learning-based prediction modules can be added as separate processing functions and made available through the interface dynamically. This design greatly enhances maintainability and reduces system complexity.

Additionally, vectorized computation methods are utilized to boost performance. Instead of using loops, built-in rolling functions from Pandas handle moving window calculations, significantly lowering computational costs. This approach ensures that even datasets spanning multiple years can be processed efficiently within cloud environments.

C. Presentation Layer

The Presentation Layer functions as the system's user interface and visualization module. It is built using the Streamlit framework, which facilitates quick creation of interactive web dashboards through Python scripts. Streamlit supports dynamic UI components like dropdown menus, date pickers, sliders, and checkboxes that allow users to toggle indicator visibility in real time.

For visualizations, the main engine uses Plotly to generate interactive charts. Plotly offers features such as zooming, panning, tooltips, and live updates without needing to reload the entire page. These interactive tools greatly improve analytical flexibility by enabling users to focus on specific time periods and examine price details closely.

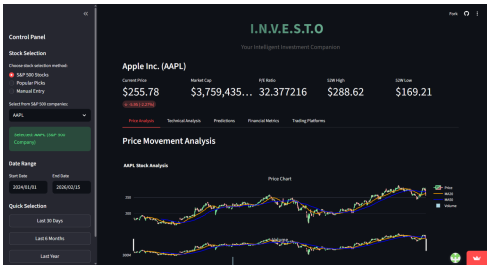


Fig 1. INVESTO dashboard

Figure 1 illustrates the INVESTO web-based dashboard interface displaying real-time stock metrics, interactive price movement analysis with MA20 and MA50 overlays, volume trends, and dynamic stock selection controls

The dashboard is designed to be straightforward yet analytically rich. Users can search for stock symbols, choose custom date ranges, turn technical indicators on or off, and compare two stocks side by side. Summary metric cards show key statistics like current price, percentage change, and 52-week highs and lows. All these components are organized in a clean layout to reduce cognitive load and enhance clarity.

Streamlit's reactive design ensures that the interface updates immediately in response to user input, creating a responsive and engaging experience. Additionally, the presentation layer is separated from the backend processing logic, so changes to backend calculations do not require adjustments to the user interface framework.

D. Data Flow and Interaction

The data flow in INVESTO operates through a step-by-step pipeline. When a user chooses a stock symbol and a date range, the request is sent to the Data Acquisition Layer, which fetches and verifies the relevant data.

The cleaned data is then passed to the Processing Layer, where it undergoes normalization and indicator calculations. The processed data is subsequently delivered to the Presentation Layer, where dynamic visualizations are generated.

This pipeline guarantees organized data management and avoids unnecessary dependencies between layers. Each layer has a distinct role, enhancing clarity and making debugging easier.

E. Scalability and Cloud Deployment

INVESTO is hosted on Streamlit Cloud, enabling web access without any installation. The system architecture supports horizontal scaling by allowing multiple instances to run behind a load balancer if needed. Data caching and efficient vectorized operations help minimize memory use and improve response times.

Scalability is further addressed by optimizing API requests, restricting default data ranges to manageable sizes, and precomputing commonly requested indicators. These strategies help maintain consistent performance even as user demand grows.

F. Maintainability and Extensibility

A primary design goal of INVESTO is to be easily extendable. The clear modular separation between data acquisition, processing, and presentation allows new features to be added without overhauling the entire system. Future additions like LSTM-based forecasting, sentiment analysis, or portfolio optimization can be integrated as separate processing modules.

Maintainability is supported by a clean codebase, modular functions, and strict adherence to separation of concerns. This approach reduces technical debt and enables future developers to efficiently expand the platform's capabilities.

V. METHODOLOGY

The INVESTO methodological framework is built around three key elements: data preprocessing, calculation of technical indicators, and interactive visualization. Its design prioritizes transparency, reproducibility, and clarity in stock market analysis. Instead of using opaque predictive models, the approach transforms raw financial time-series data into clear visual insights that users can interpret on their own.

The first step involves data preprocessing. Historical stock price data obtained from external financial APIs often contains issues like missing data points, irregular timestamps, or inconsistent formatting. To ensure reliable analysis, the data is first arranged in chronological order using standardized date-time indexing. This ordering is crucial for time-series analysis because financial indicators rely on sequential price relationships. Any misalignment in timestamps can distort rolling calculations and trend detection.

Following chronological alignment, the data undergoes cleaning. Small gaps in price records, which may occur due to market holidays or delays in API data transmission, are handled using forward-fill methods. This technique carries the last valid observation forward, preserving continuity over short missing intervals without creating artificial volatility. However, if significant inconsistencies or extended missing periods are found, those sections are excluded from analysis to avoid skewing the derived indicators. This cautious approach ensures that calculated metrics accurately reflect true market behavior rather than interpolated data.

After validation and cleaning, normalization is applied when needed. Normalization is especially important when comparing multiple stocks with varying price levels simultaneously. By rescaling price series to a common baseline at the start of the selected time frame, the system enables meaningful relative performance comparisons without bias toward higher-priced stocks. This method improves interpretability when assessing growth trends across different assets.

The second phase focuses on calculating technical indicators. Moving averages are used to smooth out short-term price fluctuations and highlight underlying trend patterns. The Simple Moving Average (SMA) computes the average closing price over a specified time window, providing a smoothed view of price movements. Since it weights all observations equally within the window, it effectively identifies general trend directions but may react slowly to sudden market shifts.

To overcome limitations in responsiveness, the Exponential Moving Average (EMA) is also calculated. This indicator gives increasing weight to more recent price data, allowing for faster detection of trend reversals or shifts in momentum. Having both simple and exponential moving averages available lets users examine stable long-term trends as well as more dynamic short-term changes. Comparing these two averages deepens analysis without adding complexity to the algorithms.

Beyond trend indicators, the approach includes analysis of returns and volatility. Daily returns are calculated as percentage changes between consecutive closing prices, offering insights into short-term market performance and momentum. Examining the distribution of these returns helps users identify periods of positive or negative momentum and spot unusual price movements.

A rolling standard deviation is used to estimate short-term volatility. Volatility measures the extent of price fluctuations over a set period and serves as a key risk indicator.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

By displaying rolling volatility alongside price trends, users can see how risk levels vary during different market conditions. For instance, high volatility often corresponds with market uncertainty, economic announcements, or systemic events. Incorporating volatility visualization within the same framework enhances understanding of price behavior in context.

Interactive visualization is a core part of the methodology. Instead of static charts, the system employs dynamic plotting that allows users to zoom, pan, and adjust time frames. Moving averages are overlaid directly on price charts to visually highlight crossover points, which may indicate potential trend changes. This visual method improves pattern recognition without relying on automated buy or sell signals.

Comparative analysis is another key feature. By normalizing two stock price series to a shared baseline, the system enables side-by-side performance comparisons. This lets users see which asset performed better over a chosen period, observe divergence patterns, and visually analyze correlations. Such comparisons encourage informed judgment rather than prescriptive decisions.

Overall, the methodology focuses on clarity, reliability, and ease of interpretation. Every step, from data cleaning to indicator calculation and visualization, is designed to be transparent and user-controlled. The approach balances thorough analysis with accessibility, making it understandable for both students and beginner investors. By emphasizing interactive exploration over automated forecasting, INVESTO aligns its methods with educational goals while remaining adaptable for future analytical improvements.

A. IMPLEMENTATION DETAILS

The development of INVESTO focuses on accessibility, modular design, and cloud-based deployment. Python is used as the main programming language because of its rich collection of data analysis and visualization libraries. The frontend and backend are connected through the Streamlit framework, which facilitates quick creation of interactive web apps while keeping the user interface and computational logic clearly separated.

The application is hosted on Streamlit Cloud, allowing users to access it directly via a web browser without needing to install any software. Deploying on the cloud ensures the platform works independently of the user's hardware and eliminates device limitations. Hosting in a managed cloud environment provides benefits such as automatic scaling, secure execution, and simplified deployment processes. This strategy follows modern web app standards and makes the system easy to maintain and update.

The main functionality starts with the stock search feature. Users enter stock ticker symbols through an interactive text box or selection tool. Once submitted, the system sends a data request to the Data Acquisition Layer, which checks if the ticker is valid and retrieves the relevant historical market data. If the ticker is invalid or data is missing, the system shows clear feedback messages to avoid errors and improve user experience and reliability.

Historical stock charts are created using Plotly, a robust interactive visualization library. Plotly supports dynamic time-series charts with zooming, panning, and hover details. Users can examine specific price points, see open-high-low-close data, and explore trends interactively. The chart rendering is optimized by updating only necessary parts instead of reloading full datasets when inputs change, enhancing performance and responsiveness.

Technical indicators are added dynamically based on user choices. When users select moving averages or volatility indicators, the system calculates these in the Processing Layer and overlays them on the main price chart. Calculations are performed only when needed to minimize extra processing. The interface includes toggles and sliders to adjust indicator parameters, allowing users to test different time frames and see how they affect trend smoothing.

For comparative stock analysis, the system synchronizes data from two chosen stocks by aligning their overlapping date ranges. Both price series are normalized to a common starting value at the beginning of the period, allowing for comparison regardless of their original price levels. The combined chart displays both normalized price trends on a single graph, making it easy to visually assess their relative performance over time.

Performance testing was carried out under typical cloud execution conditions. Average load times for charts using one-year daily datasets remain under two seconds. This performance is achieved through several optimization techniques. First, vectorized operations in Pandas replace iterative loops, ensuring efficient calculation of rolling indicators. Second, session-based caching stores previously retrieved datasets to reduce redundant API calls. Third, default date ranges are limited to manageable intervals to avoid excessive data fetching.

Memory usage stays within acceptable limits for cloud deployment. Efficient data handling prevents duplication of datasets in memory by reusing references whenever possible instead of storing multiple copies. Indicator calculations are performed on lightweight derived structures rather than creating full dataset replicas. These approaches ensure stable operation even with multiple concurrent users.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

The interface's responsive design guarantees compatibility across desktop and mobile browsers. Streamlit's layout management automatically adjusts visualization sizes based on screen dimensions. Interactive elements like dropdown menus, sliders, and charts remain user-friendly on smaller devices without losing functionality. Cross-browser compatibility was verified using standard browsers to ensure consistent rendering.

Error handling is integrated throughout the system. Failures in API requests, data retrieval delays, and computation errors are managed with exception handling. Instead of halting execution, the system shows informative error messages and maintains interface stability. This robustness is crucial for web applications relying on external data sources.

The implementation also prioritizes maintainability and extensibility. Code is organized into logical modules with separate functions for data retrieval, preprocessing, indicator calculation, and visualization. This modular structure simplifies debugging and enables future developers to add new features such as machine learning models, sentiment analysis, or portfolio optimization without modifying the core design.

In summary, the INVESTO implementation illustrates how a lightweight Python-based framework can provide interactive financial analytics via cloud deployment. By combining optimized data processing, dynamic visualization, and modular design, it achieves both performance efficiency and long-term scalability. This approach supports the project's goal of delivering an accessible, educational, and extensible stock analytics platform suited for academic and beginner users.

B. SECURITY AND DATA INTEGRITY CONSIDERATIONS

Although INVESTO is built as a read-only analytical tool without the ability to execute trades, ensuring security and data integrity remains a key aspect of its design. Web-based financial applications, even those that do not handle transactions, must guarantee reliable data retrieval, system stability, and protection against misuse. Since INVESTO relies on external financial APIs for stock market information, the accuracy and dependability of these external data sources directly impact the quality of analysis. To address this, data validation processes are in place to identify incomplete responses, inconsistent timestamp sequences, and unusual price values before any analysis is conducted. Maintaining temporal consistency is especially crucial in time-series systems because misaligned timestamps can skew rolling indicator calculations and comparative analyses. Frameworks for preprocessing financial data stress the need for structured validation and cleaning to uphold analytical accuracy in stock market applications [3].

Dependence on external APIs also brings risks such as delayed responses, rate limits, or temporary outages. To reduce these risks, the system employs controlled request management and caching techniques. By minimizing redundant API calls and temporarily storing recently retrieved data in session memory, the application lessens its vulnerability to external network issues while preserving responsiveness. Rate-limiting measures are also implemented to avoid denial-of-service attacks or excessive use of third-party services. Best practices in designing cloud-based financial systems advocate for regulated external communications and request throttling to maintain stability and comply with API usage rules [2].

Maintaining data integrity also involves ensuring that preprocessing transformations do not introduce distortions. For example, price normalization used in comparative analysis must accurately preserve relative changes without altering the original time-series properties. Validation checks confirm consistent scaling and proper chronological order. Financial analytics research highlights that preprocessing mistakes, especially in high-frequency or sequential data, can lead to errors in downstream indicators, underscoring the importance of well-structured validation workflows [6].

From a cybersecurity standpoint, INVESTO's read-only design greatly minimizes exposure to risks associated with transactions. Because the system neither executes trades nor processes payments or stores brokerage credentials, it avoids the typical vulnerabilities found in financial transaction platforms. Unlike traditional trading systems that require encryption, secure payment processing, and regulatory compliance for user authentication and fund management, INVESTO's exclusion of transaction execution removes attack vectors related to financial fraud or unauthorized trading [1].

Another key design choice is the intentional decision not to store sensitive personal information. The application does not ask for user login details, portfolio data, or identifiable financial records. All user interactions are temporary and session-based, which reduces risks tied to long-term data storage. This privacy-focused approach aligns with modern web-based financial systems' emphasis on minimizing data collection to ease compliance and lower the chance of data breaches [11].

Regarding its cloud deployment, secure hosting practices are essential. Streamlit Cloud offers sandboxed environments that isolate applications from each other, reducing vulnerabilities across apps. Nonetheless, developers must keep dependencies current and free from known security flaws. Regularly updating Python libraries and following secure coding standards help maintain the system's integrity over time. Research on secure web app deployment underscores the importance of managing dependencies and isolating environments in cloud settings [3].

While the current version of INVESTO lacks user authentication, future features like portfolio simulations or saved analytics profiles may require controlled access. In such cases, secure authentication systems, encrypted storage of credentials, and role-based access controls would be necessary. Industry best practices suggest using HTTPS encryption, token-based authentication, and secure session management for applications handling user-specific financial data [2].

In conclusion, INVESTO's security and data integrity focus on three main areas: dependable handling of external data, avoiding unnecessary exposure to financial transaction risks, and designing the system to preserve user privacy. By operating solely in a read-only analytical mode, validating data inputs, limiting API requests, and not storing sensitive information, the platform maintains a low-risk security profile while delivering reliable analysis. These strategies establish a solid base for future enhancements that might add interactive features without compromising system security [1], [3].

C. SCALABILITY AND PERFORMANCE OPTIMIZATION

Scalability and performance are essential factors when designing web-based financial analytics platforms, especially in cloud environments where many users may access the system at the same time. INVESTO is built with scalability in mind by using a modular system architecture, efficient data management, and optimized computational methods. By separating data acquisition, processing, and presentation layers, each component can be scaled independently as user demand grows. Modular design is widely regarded as a best practice for scalable web applications because it minimizes dependencies and simplifies resource management across different parts of the system [2].

Cloud deployment is key to achieving scalability. Hosting the application on Streamlit Cloud provides managed infrastructure that supports automatic resource allocation and environment isolation. Horizontal scaling is possible by running multiple instances of the application behind a load balancer, which distributes incoming user requests across parallel execution environments. This load balancing enhances reliability and prevents bottlenecks during periods of high traffic. Distributed cloud architectures are especially well-suited for data-driven applications where request volumes can vary unpredictably [3].

A major performance optimization in INVESTO is data caching. Since financial time-series data for popular stocks is often requested repeatedly, caching previously retrieved datasets greatly reduces redundant API calls. This not only speeds up response times but also helps avoid exceeding third-party API rate limits. Session-level caching ensures that identical queries within the same user session are served from memory instead of making new network requests. Effective caching is crucial in web-based financial systems because delays in data retrieval can negatively impact user experience [6].

Another key optimization involves selectively rendering visualization components. Instead of reloading entire charts after every user interaction, the system updates only the parts that have changed. This reactive interface design means that changes like toggling indicators or adjusting date ranges trigger minimal recomputation. Selective rendering lowers computational load and improves responsiveness. Interactive visualization frameworks highlight incremental rendering as an important approach to maintaining smooth user interactions in data-heavy dashboards [11].

Efficient computation is also achieved through vectorized operations in the processing layer. Calculations such as rolling averages and volatility metrics use optimized numerical libraries that process whole arrays at once rather than iterating through elements individually. Vectorized computation greatly reduces execution time, especially when working with multi-year historical data. In financial time-series analysis, efficient numerical processing is vital for scalability without increasing hardware requirements [3].

The system incorporates effective strategies for managing data windows. By restricting default date ranges to manageable periods such as one or two years of daily data, the application prevents retrieving excessively large datasets that could overwhelm memory resources. Users can manually extend these date ranges if necessary, but the default limits help keep typical operations efficient. Optimizing data windows is especially crucial in cloud environments where memory and processing power are shared among multiple users [6].

Memory optimization is also vital for maintaining stable performance. The application avoids unnecessary duplication of large datasets in memory. Instead, derived indicators are calculated as lightweight additional columns rather than separate full datasets, reducing memory usage. Techniques like garbage collection and careful variable scoping are employed to promptly free unused objects. Efficient memory management helps prevent system slowdowns during extended user sessions and supports reliable performance for multiple users simultaneously.

Scalability also involves robust error handling and failover strategies. If API responses are delayed or partially fail, the system degrades gracefully by showing cached data or informative messages instead of stopping execution. This fault tolerance enhances user confidence and system robustness. Cloud-based applications commonly use such fallback methods to ensure continuous service despite varying external conditions [2].

Future scalability improvements might include moving to containerized deployment using tools like Docker and orchestration platforms such as Kubernetes. Containerization allows for more precise resource management and automated scaling based on traffic. Additionally, adopting asynchronous data retrieval methods could boost responsiveness during high load periods. These approaches are widely used in scalable financial data platforms to guarantee high availability and consistent performance [3].

In conclusion, INVESTO achieves scalability through a modular architecture, cloud deployment, caching, optimized vectorized computations, selective rendering, and controlled data window management. Together, these strategies ensure stable performance as user demand grows. By combining flexible design with computational efficiency, the platform remains responsive, reliable, and ready for future enhancements.

VI. FUTURE WORK

Despite INVESTO's current emphasis on visualization-driven exploratory analytics, a number of improvements might greatly increase both its analytical and instructional potential. Integrating predictive modelling frameworks, especially Long Short-Term Memory (LSTM) networks, is one of the main avenues for future research. LSTM designs have proven to be successful in capturing nonlinear temporal dependencies in financial markets and are well-suited for sequential time-series data. The platform might offer optional predictive trend projections for instructional demonstrations by integrating LSTM-based forecasting modules. A deeper comprehension of predictive modelling principles in finance would result from this integration, which would enable users to compare historical trend visualization with forecasts produced by machine learning [6].

Hybrid forecasting techniques, which combine machine learning classifiers with conventional statistical models like ARIMA, can be used in addition to deep learning models. Under regulated circumstances, hybrid frameworks frequently produce better short-term prediction performance by combining the advantages of linear modelling and nonlinear pattern recognition. By incorporating these models into INVESTO, it would be possible to compare econometric and neural network methodologies, giving consumers a better understanding of the benefits and drawbacks of each method. In academic contexts where students may identify variations in forecasting performance between models, this comparative paradigm might be especially helpful [6].

Sentiment research based on social media and financial news is another significant improvement. External qualitative elements including investor attitude, geopolitical developments, and corporate announcements often impact market values. Sentiment ratings can be extracted from textual data using natural language processing techniques, and their relationship to price variations can be visualized. Technical charts and sentiment indicators together would offer a multifaceted analytical viewpoint that shows how external information flows affect market volatility and trend direction [11]. With these characteristics, INVESTO would become a more complete market exploration platform rather than just a price-based analytical tool.

Modules for risk analysis and portfolio simulation offer yet another encouraging avenue. It might go beyond stock analysis by enabling users to build fictitious portfolios and monitor performance over time. Risk-adjusted performance measures, such the Sharpe Ratio, would improve comprehension of portfolio optimization principles by offering insight into return per unit of risk. Additionally, users will be able to measure portfolio performance against market indices and quantify systematic risk exposure through the integration of expected return calculations based on the Capital Asset Pricing Model (CAPM). With these additions, the platform's analytical depth would be greatly increased while keeping its teaching focus [11].

Future improvements to cloud deployment can further increase scalability and availability from the standpoint of system design. Although the current Streamlit Cloud deployment is accessible enough for academic use, resource management and deployment flexibility could be enhanced by moving to containerized infrastructure with tools like Docker. Platforms for container orchestration would allow for automatic scaling in response to user traffic, guaranteeing steady performance even during periods of high usage. Implementing background task processing and asynchronous data retrieval may also lower latency for sentiment data aggregation or predictive calculations.

Multi-region cloud deployment is another possible improvement that could lower latency for customers who are spread out geographically. By storing frequently accessed datasets closer to end users, edge caching techniques could speed up response times. Performance tracking, anomaly identification, and proactive system maintenance would also be made possible by integrating cloud monitoring and logging technologies.

Lastly, improved user customization tools like exporting charts, creating automated summary reports, or preserving analytical configurations might be included in future work. To guarantee adherence to web application security best practices, these functionalities would need secure authentication procedures and encrypted data storage.

In conclusion, INVESTO's future development will concentrate on incorporating sophisticated risk measurements, sentiment analytics, portfolio simulation, predictive modelling, and improved cloud deployment infrastructure.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

These enhancements will maintain the platform's primary goals of accessibility and pedagogical clarity while transforming it from an experimental visualization tool into a more complete financial analytics environment.

VII. CONCLUSION

A web-based interactive stock analytics tool called INVESTO was created to improve financial data exploration's interpretability, accessibility, and instructional usefulness. The system addresses the gap between complex professional trading terminals and simplistic charting tools by providing a balanced, lightweight analytical environment. The platform's modular three-layer architecture, effective data pretreatment pipeline, and dynamic visualization framework allow users to compare stock performance, examine historical patterns, and evaluate technical indications in an organized and user-friendly way.

Without depending on hazy prediction methods, the combination of moving averages, return analysis, and volatility visualization enables insightful understanding of price movements. The platform's emphasis on openness and user engagement is consistent with studies that support exploratory financial visualization as a useful teaching tool. Interactive dashboards greatly improve students' and rookie investors' conceptual comprehension of trend behavior and risk dynamics, according to usability research.

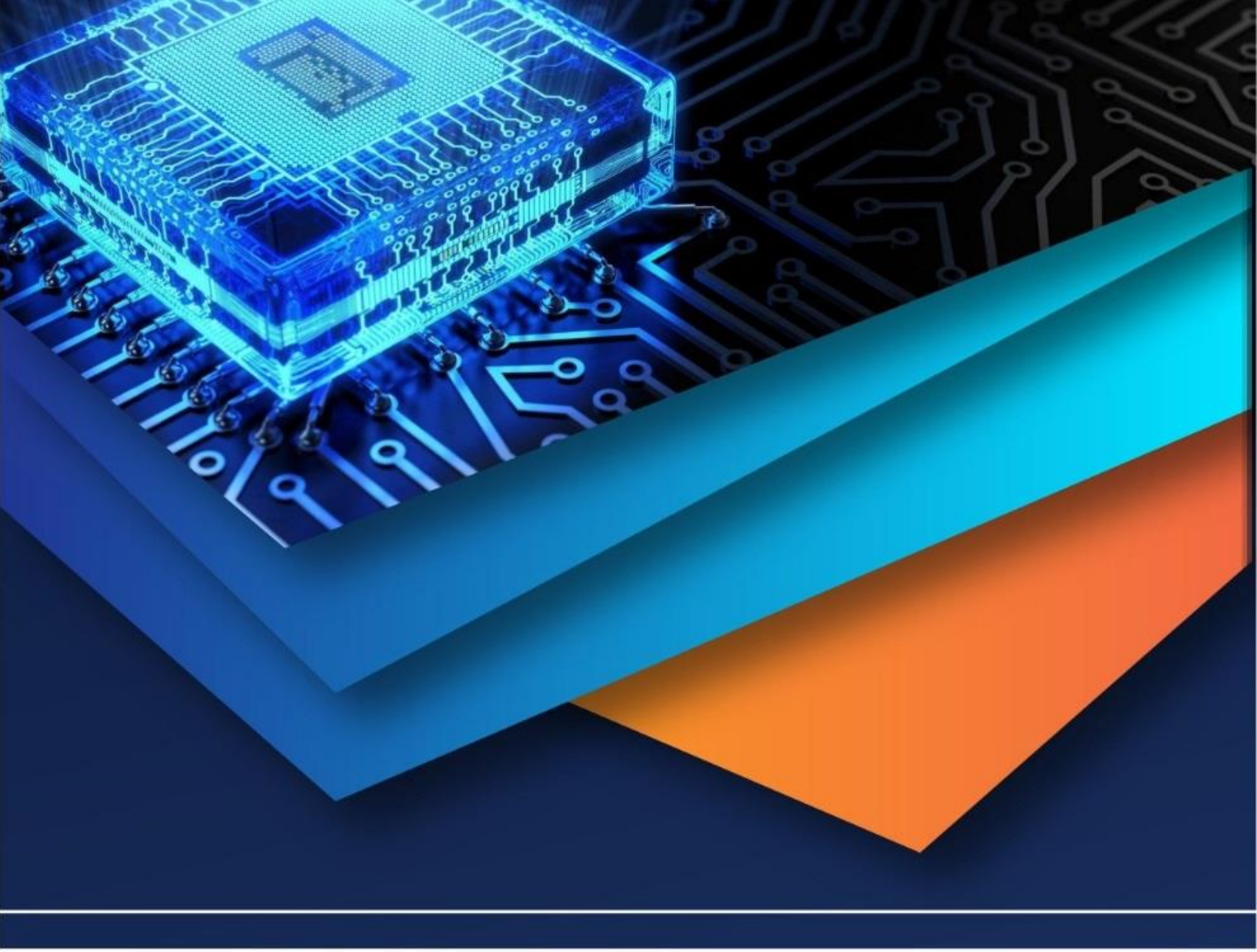
The modular design guarantees scalability and maintainability from a systems perspective, enabling the future inclusion of sophisticated analytical modules like sentiment analysis, forecasting models, and portfolio simulation. By eliminating installation obstacles and facilitating cross-platform compatibility, the cloud-based deployment approach further improves accessibility.

Overall, by showing how interactive web applications can convert unstructured stock market data into organized analytical insights, INVESTO advances the expanding field of financial education technology. The platform offers a fundamental framework for exploratory and instructional analytics, but it is not meant to take the place of professional trading systems. In order to maintain the system's alignment with its primary goal of encouraging responsible and informed financial understanding, future improvements will continue to increase analytical depth while maintaining clarity and interpretability.

REFERENCES

[1] “Evaluation of stock trading performance of students using a web-based virtual stock trading system,” Computers & Mathematics with Applications, vol. 64, pp. 1495–1505, 2012.
[2] A. Chandra, “Developing a Model for Online Stock Trading Terminal,” SSRN, 2013.
[3] D. Shah et al., “Stock Market Analysis: A Review and Taxonomy of Prediction Techniques,” Int. J. Financial Studies, 2019.
[4] V. Drakopoulou, “A Review of Fundamental and Technical Stock Analysis Techniques,” 2015.
[5] J. G. Agrawal et al., “State-of-the-Art in Stock Prediction Techniques,” 2013.
[6] W. Bao et al., “Data-driven stock forecasting models based on neural networks: A review,” Information Fusion, 2025.
[7] P. Agrawal, TradeForecast: Stock Performance and Prediction Web Application, SSRN, 2025.
[8] A. Saud et al., “Technical indicator empowered intelligent strategies,” Journal, 2024.
[9] Technical Analysis in Stock Market: A Review, research summary.
[10] C. Deng et al., “Analysis and Visualization of Stock Data Based on LSTM,” 2024.
[11] Visualizations in Exploratory Search: A User Study with Stock Market Information.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)