N. Indraneel Reddy, 2026, 14:2
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

AI Driven Cryptocurrency Trading System

N. Indraneel Reddy[1], N. Manohar Guptha[2], P. Jyothi Prakash[3], M. Teja Pavan[4], Mrs.R. Mano Ranjani[5]

[1,2,3,4]UG Student, Department of Computer Science and Engineering, School of Engineering and Technology, Dhanalakshmi Srinivasan University, Trichy-621112-Tamilnadu.

[5]Assistant Professor, Department of Computer Science and Engineering , Dhanalakshmi Srinivasan Institute of Technology, Trichy-621112- Tamil Nadu.

Abstract- Cryptocurrency markets are highly dynamic and operate continuously, requiring traders to analyze price movements and make decisions in real time. Manual trading methods often fail to respond efficiently to rapid market fluctuations, leading to delayed actions and inconsistent results. This paper presents the design and implementation of a real-time cryptocurrency trading bot that performs market analysis, visualizes price movements, and executes simulated trades using technical indicators. The proposed system utilizes live market data obtained from the Binance public API, candlestick chart visualization, portfolio tracking, and an automated trading strategy based on Exponential Moving Average (EMA) crossover logic. The system is implemented as a web application and focuses on transparency, simplicity, and real-time responsiveness. The results demonstrate that automated rule-based trading can improve decision consistency and provide a practical platform for understanding algorithmic trading concepts.

Keywords: Cryptocurrency Trading, Algorithmic Trading, EMA Strategy, Real-Time Data, Trading Bot.

I. INTRODUCTION

Cryptocurrencies have emerged as a major financial asset class due to their decentralized nature and global accessibility.

Unlike traditional stock markets, cryptocurrency markets operate twenty- four hours a day and are known for rapid price fluctuations. These characteristics make continuous monitoring and fast decision-making essential for successful trading. However, human traders are limited by attention span, emotional bias, and delayed reactions to market changes.

Automated trading systems aim to address these limitations by executing trades based on predefined rules and real- time market data. Such systems reduce emotional influence, improve execution speed, and maintain consistency in trading decisions. Technical indicators play a crucial role in algorithmic trading by identifying market trends and potential entry or exit points. Among these indicators, Exponential Moving Averages (EMAs) are widely used due to their simplicity and effectiveness.

This project focuses on developing a real-time cryptocurrency trading bot that integrates live market data, technical analysis, and automated trading logic within a web-based interface. The system is designed for educational and simulation purposes and does not involve real financial transactions.

Technical analysis forms the foundation of many automated trading strategies. Indicators such as moving averages, relative strength index, and momentum oscillators are commonly used to identify trends and potential reversal points in the market. Among these indicators, the Exponential Moving Average (EMA) is particularly effective because it assigns greater importance to recent price data, making it more responsive to current market conditions. EMA-based crossover strategies are widely used due to their simplicity, clarity, and practical effectiveness in trend-following systems.

II. SYSTEM OVERVIEW

Fig.1 EHR Development vs project phrases

The proposed cryptocurrency trading bot is developed as a real-time, web-based application that integrates live market data, technical analysis, and automated trading logic into a single platform. The system is primarily intended for educational and simulation purposes, allowing users to observe market behavior and understand algorithmic trading mechanisms without involving real financial risk.

The system continuously collects live price data and historical candlestick information from a public cryptocurrency exchange API.

This data forms the foundation for all subsequent analysis and decision-making processes. By separating data acquisition, processing, analysis, and execution into distinct components, the system achieves modularity, clarity, and ease of maintenance.

At a high level, the system workflow begins with fetching real-time market prices and historical data at regular intervals. The retrieved data is then cleaned, formatted, and synchronized to ensure consistency across charts and indicators. Technical indicators, specifically Exponential Moving Averages (EMA), are computed using the processed data and overlaid on candlestick charts to visually represent market trends.

The user interface serves as the interaction point between the system and the user. It provides real-time candlestick charts, EMA overlays, portfolio balances, and a detailed order history. Users can manually place trades or enable automated trading mode, where the system executes trades automatically based on indicator signals. All actions are logged and displayed transparently to help users understand how each trading decision is made.

III. SYSTEM ARCHITECTURE

The system architecture of the proposed cryptocurrency trading bot is designed using a layered and modular approach to ensure clarity, scalability, and ease of understanding. Each layer performs a specific function and communicates with adjacent layers through well-defined data flows. This design helps isolate responsibilities, reduce system complexity, and simplify future enhancements.
The architecture is structured to support real-time data handling, technical analysis, and automated trading simulation while maintaining transparency in system operations. Figure 1 illustrates the interaction among the major architectural components.

A. Market Data Source

The primary source of data for the system is a public cryptocurrency exchange API, which provides live market prices and historical candlestick data. The API delivers essential trading information such as open, high, low, close prices and timestamps for selected trading pairs. Since the data source is external and dynamic, the system is designed to handle frequent updates and minor network delays without interrupting operation.

B. Data Acquisition Layer

The data acquisition layer is responsible for establishing communication with the market data source and retrieving real-time and historical data at fixed intervals. This layer sends HTTP requests to the exchange API and receives responses in structured format. To ensure uninterrupted operation, the system includes basic error handling mechanisms such as request retries and response validation.

This layer acts as a bridge between the external market environment and the internal system components, ensuring that only relevant and valid data is forwarded for further processing.

C. Data Processing Layer

The data processing layer converts raw market data into a usable format suitable for chart visualization and indicator calculation. This includes sorting data by time, aligning candlestick intervals, normalizing price values, and removing incomplete or duplicate entries.

Time synchronization is also handled at this stage to ensure consistency between historical data and real-time price updates. By maintaining clean and well-structured data, this layer improves the accuracy of technical analysis and trading decisions.

D. Technical Analysis Layer

The technical analysis layer computes indicators that assist in identifying market trends and potential trading opportunities. In the current implementation, Exponential Moving Averages EMA(9) and EMA(21) are calculated using closing prices derived from historical candlestick data.

These indicators are updated continuously as new price data becomes available. The calculated values are used both for graphical display and as inputs to the trading logic. The separation of indicator computation from trade execution allows additional indicators to be added in the future without modifying core trading rules.

E. Trading Logic Layer

The trading logic layer evaluates technical indicator values to determine trading signals. Buy and sell decisions are generated based on predefined EMA crossover conditions. The system checks current market state, indicator alignment, and available virtual balance before executing any trade.

Instead of placing real trades on an exchange, this layer interacts with a virtual portfolio module. This design choice ensures risk-free experimentation and allows users to observe trading behavior without financial consequences.

F. Virtual Portfolio and Order Management

The virtual portfolio module maintains simulated account balances, open positions, and completed trades. Each transaction updates the portfolio state and is recorded in the order history with details such as price, time, trade type, and quantity.

This module provides transparency and traceability, enabling users to analyze trading outcomes and understand the effect of each trading decision.

G. Presentation Layer

The presentation layer represents the user-facing component of the system. It displays real-time candlestick charts, EMA indicator overlays, portfolio balances, and order history. Interactive controls allow users to switch between manual and automated trading modes, enable or disable indicators, and monitor system activity.

The presentation layer ensures that complex backend operations are visualized in a clear and user-friendly manner, improving usability and learning effectiveness.

H. System Interaction Flow

The overall system operation follows a continuous cycle: data acquisition, processing, analysis, decision-making, and visualization. Each cycle updates the system state in real time, ensuring that users receive accurate and timely information. The modular interaction among layers ensures stable performance and supports future scalability.

Architectural Advantages

The proposed architecture offers several advantages:

- Modular and easy to extend
- Clear separation of responsibilities
- Real-time responsiveness
- Safe simulation-based trading
- Suitable for academic and research purposes

The system architecture of the proposed cryptocurrency trading bot follows a layered design that enables efficient handling of real-time data, technical analysis, and trading simulation.

The architecture ensures that each component performs a specific task while maintaining clear interaction with other components.

The architecture begins with the Market Data Source, which provides live and historical cryptocurrency price data through a public exchange API. This data is collected by the Data Acquisition Layer, which periodically fetches price updates and candlestick information.

The fetched data is then forwarded to the Data Processing Layer, where it is cleaned, formatted, and synchronized to ensure accuracy and consistency. This processed data is used by the Technical Analysis Layer to calculate indicators such as Exponential Moving Averages (EMA-9 and EMA-21).

Based on the indicator values, the Trading Logic Layer evaluates predefined rules to generate buy and sell signals. Instead of executing real trades, the system updates a Virtual Portfolio and Order Management Module, which simulates trading behavior and records transaction history.

Finally, the Presentation Layer displays real-time charts, indicator overlays, portfolio balances, and order history to the user. This layered architecture improves modularity, simplifies debugging, and allows easy future enhancements.

The system architecture of the proposed cryptocurrency trading bot is designed using a layered and modular approach to ensure clarity, scalability, and ease of understanding. Each layer performs a specific function and communicates with adjacent layers through well-defined data flows. This design helps isolate responsibilities, reduce system complexity, and simplify future enhancements.

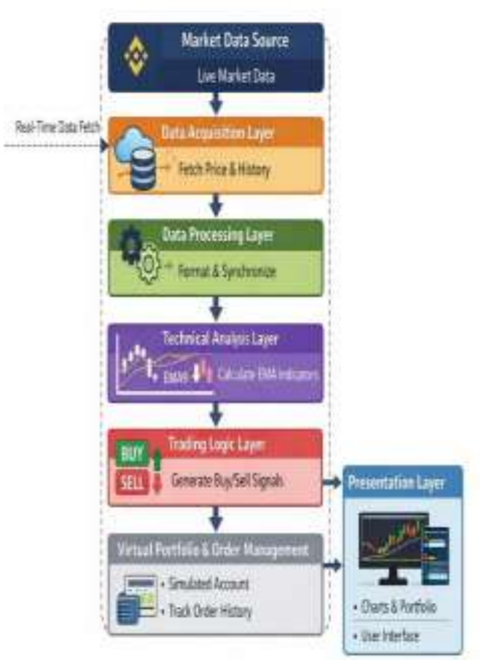

FIGURE 1: System Architecture of the Crypto Trading Bot

The system architecture is designed to support continuous operation in a highly dynamic market environment. Since cryptocurrency prices change rapidly, the architecture emphasizes timely data flow and minimal processing delay between data acquisition and decision execution. Each layer processes only the required information and immediately forwards it to the next layer, reducing computational overhead.

The separation between the Technical Analysis Layer and the Trading Logic Layer plays a critical role in maintaining system flexibility. Indicator calculations are kept independent from trading decisions, ensuring that changes in strategy rules do not affect indicator computation. This approach allows the system to support multiple trading strategies using the same indicator data.

Based on the indicator values, the Trading Logic Layer evaluates predefined rules to generate buy and sell signals. Instead of executing real trades, the system updates a Virtual Portfolio and Order Management Module, which simulates trading behavior and records transaction history.

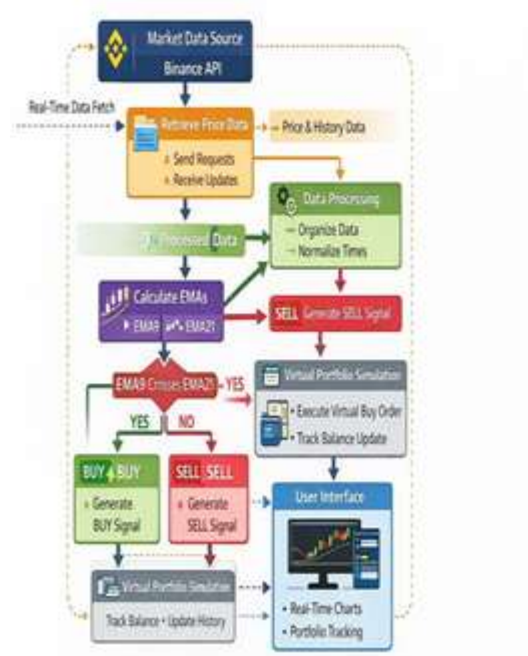

FIGURE 2: Detailed flow of the System Architecture for the Crypto Trading Bot

The separation between the Technical Analysis Layer and the Trading Logic Layer plays a critical role in maintaining system flexibility. Indicator calculations are kept independent from trading decisions, ensuring that changes in strategy rules do not affect indicator computation. This approach allows the system to support multiple trading strategies using the same indicator data.

The Virtual Portfolio and Order Management module acts as a controlled environment that closely mimics real trading conditions. It enforces balance constraints, prevents duplicate or invalid trades, and records each transaction with accurate timestamps. This module is essential for performance evaluation, as it allows users to analyze profit, loss, and trading frequency over time.

From a usability perspective, the Presentation Layer is tightly synchronized with backend operations.

Any change in market data, indicator values, or portfolio state is immediately reflected in the user interface. This real- time feedback helps users understand how technical indicators influence trading decisions and how automated logic responds to market movements.

Data Flow and Control Flow

The architecture supports both data flow and control flow mechanisms. Market data flows sequentially from the data source to analysis and trading layers, while control flow is influenced by user actions such as enabling automated trading or placing manual trades. This dual-flow design ensures that user interventions do not disrupt real-time data processing.

Scalability and Extensibility

The modular nature of the architecture allows easy extension of system functionality. New indicators, alternative trading strategies, or additional visualization components can be integrated by adding new modules without restructuring the entire system. This makes the architecture suitable for future research and experimentation.

Reliability and Safety

To ensure reliability, the architecture avoids direct interaction with live trading accounts. All trading actions are executed within a simulated environment, eliminating financial risk. This safety-focused design makes the system appropriate for academic use, demonstrations, and student projects.

IV. TRADING STRATEGY

The trading strategy used in the proposed system is based on a rule- driven technical analysis approach. The objective of this strategy is to identify short-term market trends and generate clear buy and sell signals using historical price behavior. A simple and widely accepted indicator- based method is chosen to ensure transparency, reliability, and ease of understanding.

The system implements an Exponential Moving Average (EMA) crossover strategy, which is commonly used in algorithmic trading due to its effectiveness in trend detection. EMAs assign greater weight to recent prices, allowing the strategy to react faster to market changes compared to simple moving averages.

A. Selection of EMA Indicators

Two EMAs with different time periods are used:

- EMA-9 (Short-Term EMA): Represents recent price movements and responds quickly to market changes.
- EMA-21 (Long-Term EMA): Represents broader market trends and provides a smoother price direction.

The combination of short-term and long-term EMAs helps in identifying potential trend reversals and momentum shifts.

B. Buy Signal Generation

A buy signal is generated when the short-term EMA (EMA-9) crosses above the long-term EMA (EMA-21). This crossover indicates increasing bullish momentum and suggests that the market price may continue to rise.

Before executing a buy trade, the system verifies:

- Availability of sufficient virtual balance
- Absence of an existing open buy position
- Validity of the crossover condition over consecutive data points

Once these conditions are satisfied, a virtual buy order is placed at the current market price.

Signal Confirmation and Noise Reduction

To avoid false signals caused by short- term price noise, the system applies basic confirmation rules such as:

- Evaluating crossover persistence over multiple intervals
- Ignoring repeated signals within a short time frame
- Preventing consecutive buy or sell actions without trend reversal

These safeguards improve the reliability of trading decisions.

Manual vs Automated Strategy Execution

The trading strategy can be executed in two modes:

- **Manual Mode:**

Users manually trigger buy or sell actions while using EMA indicators for visual guidance.

- **Automated Mode:**

The system continuously monitors EMA crossover conditions and executes trades automatically without user intervention.

This dual-mode approach allows users to compare manual decision- making with automated trading behavior.

Strategy Limitations

While the EMA crossover strategy performs well in trending markets, it may generate frequent signals during sideways or highly volatile market conditions. Since the strategy does not predict price movements but reacts to them, some trades may be delayed. These limitations highlight the need for additional risk-management techniques, which are discussed as future enhancements.

Justification of Strategy Choice

The EMA crossover strategy is selected because:

- It is simple and widely accepted
- Easy to implement and explain
- Suitable for educational applications
- Provides clear visual interpretation on charts
- Aligns well with real-time trading simulation

V. SYSTEM OPERATION

The system operation describes how the proposed cryptocurrency trading bot functions in real time, from data retrieval to trade execution and user interaction. The system operates continuously while the application is active and responds dynamically to both market changes and user inputs. Its operation is designed to ensure accuracy, transparency, and smooth coordination among all architectural components.

A. Initialization Phase

When the system is launched, the application initializes essential components such as API connectivity, chart configuration, and virtual portfolio setup. Historical candlestick data for the selected trading pair is first loaded to provide sufficient data for indicator calculation. Initial EMA values are computed based on this historical data to avoid incorrect or incomplete signals during startup.

B. Real-Time Data Update Cycle

After initialization, the system enters a continuous real-time update cycle. In this cycle, live market prices are fetched at regular intervals. Each new price update contributes to the formation of the current candlestick and triggers recalculation of EMA values.

This continuous update ensures that charts, indicators, and trading logic remain synchronized with the latest market conditions.

C. Indicator Evaluation Process

At each update interval, the system evaluates the updated EMA values to detect potential crossover events.

The comparison between short-term and long-term EMAs determines whether a trading signal should be generated. The evaluation process is optimized to avoid repeated or conflicting signals within short time periods.

D. Trade Execution Workflow

When a valid trading signal is detected, the system executes the corresponding trade action within the virtual trading environment. Before execution, the system performs necessary validations such as checking available balance and current portfolio state.

Each executed trade updates:

- Virtual account balance
- Asset holdings
- Order history records

This workflow ensures consistent and realistic trade simulation.

Manual Trading Interaction

In manual trading mode, the system allows users to place buy or sell orders directly through the user interface. Manual trades follow the same validation and execution workflow as automated trades, ensuring consistency across both modes of operation.

Manual mode enables users to test personal trading decisions while observing indicator behavior.

Automated Trading Operation

In automated mode, the system independently monitors indicator conditions and executes trades without user intervention. Users can enable or disable automated trading at any time. When disabled, the system immediately stops executing trades while continuing to update market data and indicators.

This mode demonstrates the effectiveness of rule-based trading and allows comparison with manual trading performance.

User Interface Synchronization

All system activities are reflected in the user interface in real time. Candlestick charts, EMA overlays, portfolio balances, and order history are updated immediately after each data update or trade execution. This synchronization ensures that users can clearly track system behavior and trading outcomes.

Error Handling and Stability

Basic error-handling mechanisms are incorporated to handle temporary data unavailability, network delays, or invalid user actions. In such cases, the system maintains its previous stable state and resumes normal operation once valid data is restored. This improves system reliability and user experience.

System Shutdown and Session End

When the application is closed, the system safely terminates ongoing processes. Trading activity is stopped, and the session ends without affecting stored data. This ensures controlled and predictable system behavior.

VI. CONCLUSION

This paper presented the design and implementation of a real-time cryptocurrency trading bot developed for educational and simulation purposes. The system integrates live market data, technical analysis, and automated trading logic within a structured and modular architecture. By using Exponential Moving Average (EMA) crossover strategy, the system demonstrates how rule-based trading decisions can be executed consistently without human emotional bias.

The proposed system successfully performs real-time market monitoring, indicator calculation, and trade execution in a virtual trading environment. The use of candlestick charts and EMA overlays provides clear visualization of market trends, while the virtual portfolio and order history enable effective analysis of trading outcomes. Both manual and automated trading modes allow users to understand the differences between human-driven and algorithm-driven decision-making.

The modular system architecture ensures flexibility, reliability, and ease of future expansion. Since the system avoids real financial transactions, it provides a safe platform for experimentation and learning. Overall, the project serves as a practical introduction to algorithmic trading concepts and highlights the importance of structured strategies in dynamic financial markets.

REFERENCES

1. J. J. Murphy, Technical Analysis of the Financial Markets, New York, NY, USA: New York Institute of Finance, 1999.
2. S. Mohan, S. Mullapudi, S. Sammeta, P. Vijayvergia, and D. C. Anastasiu, "Stock price prediction using machine learning and deep

learning frameworks," in Proc. IEEE Int. Conf. Big Data, Los Angeles, CA, USA, 2019, pp. 381–389.

3. J. Patel, S. Shah, P. Thakkar, and K. Kotecha, "Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques," Expert Systems with Applications, vol. 42, no. 1, pp. 259–268, 2015.
4. A. N. Refenes and Y. Abu-Mostafa, "Neural networks in financial trading," IEEE Transactions on Neural Networks, vol. 4, no. 6, pp. 995–1003, Nov. 1993.
5. H. Wang, Z. Lei, X. Zhang, B. Zhou, and J. Peng, "A review of deep learning for renewable energy forecasting," IEEE Access, vol. 7, pp. 131394