ÇANKAYA UNIVERSITY

FACULTY OF ENGINEERING

COMPUTER ENGINEERING DEPARTMENT

CENG 407

Innovative System Design and Development I

Project Report

Team ID: 202301

Gridy: AI based Grid Trading Strategy Builder

Ali Kerem Demir

202011404

Aydolu Konca

202011402

Erdem Demirtaş

202011406

Ezgi Keten

201911204

Advisor: Dr. Faris Serdar Taşel

Table of Contents

Table Of Figures

Introduction

The project, titled "Gridy: AI-based Grid Trading Strategy Builder," focuses on developing an AI-based system for grid trading strategies in financial markets, with an emphasis on cryptocurrencies and futures markets. The Software Requirement Specification (SRS) section outlines the project's objectives, scope, and detailed functional and non-functional requirements. The Software Design Description (SDD) part offers an exhaustive blueprint for the system's implementation, detailing the architecture, design principles, user interface, and technical aspects to ensure alignment with the specified requirements.

Literature Review

Abstract

This paper presents a comprehensive exploration of AI-based grid trading strategies within the context of financial markets, with a particular focus on cryptocurrency and futures markets. The study delves into the historical evolution of spot markets, the intricacies of order types, the emergence of crypto exchanges, and the transformative impact of decentralized finance (DeFi). Furthermore, it examines the dynamics of derivatives markets, emphasizing the interaction between cryptocurrencies and futures contracts. The paper elucidates the significance of perpetual futures in risk management and their pivotal role in cryptocurrency markets. Additionally, it investigates the historical development, types, applications, and limitations of trading indicators, shedding light on their effectiveness in algorithmic trading. The exploration culminates in an analysis of grid trading, delineating its key concepts, advantages, and disadvantages.

Özet

Bu makale, finansal piyasalardaki yapay zeka tabanlı grid ticaret stratejilerini kapsamlı bir şekilde inceleyerek, özellikle kripto para ve vadeli işlemler piyasalarına odaklanmaktadır. Çalışma, spot piyasaların tarihsel evrimine, sipariş türlerinin karmaşıklıklarına, kripto borsalarının ortaya çıkışına ve merkezi olmayan finansın (DeFi) dönüştürücü etkisine derinlemesine inmektedir. Ayrıca, türev piyasaların dinamiklerini inceleyerek özellikle kripto paralar ile vadeli işlemler arasındaki etkileşimi vurgular. Makale, süresiz vadeli işlemlerin risk yönetimindeki önemini ve kripto para piyasalarındaki kilit rolünü açıklar. Ayrıca, ticaret göstergelerinin tarihsel gelişimini, türlerini, uygulamalarını ve sınırlamalarını inceleyerek, algoritmik ticarette etkinliklerine ışık tutar. Keşif, grid ticaretinin ana konseptlerini, avantajlarını ve dezavantajlarını belirleyerek sona ermektedir.

Introduction

In recent years, the financial landscape has witnessed a paradigm shift with the advent of AI-based grid trading strategies. This paper embarks on an in-depth exploration, commencing with an examination of spot markets and their historical evolution. The discussion extends to technical terms, order types, and the evolution of crypto exchanges, distinguishing between decentralized and centralized platforms. A dedicated section unfolds the intricate relationship between cryptocurrencies and futures contracts, unraveling the creation of crypto futures and their impact on price discovery and market integration. Practical applications of grid trading in crypto futures are illustrated, emphasizing its role in setting price levels and enabling automatic execution. The paper also explores option markets, perpetual futures, and the historical development of trading indicators, providing a foundational understanding for subsequent discussions on algorithmic trading and grid trading.

Spot Markets

Spot markets are markets where the buying or selling of financial assets takes place instantly at the latest price. Also known as the cash market.

Historical Evolution of Spot Markets

Spot markets have a history dating back to ancient civilizations where people traded goods for immediate delivery. Commodities such as precious metals such as gold and silver, agricultural products, energy products and industrial metals are traded in spot markets. The foreign exchange market is also an important part of spot markets.

The first spot market for trading shares was established in Amsterdam in the 17th century. The Amsterdam Stock Exchange was the world's first stock exchange to allow trading in stocks and bonds. Later, spot markets began to emerge in other financial centers.

Technical Terms in the Spot Market

- **Spot Price:** It is the instant buying and selling price of an asset.
- **Bid Price:** It is the price that an investor who wants to buy an asset agrees to pay.
- **Ask Price:** It is the price that an investor who wants to sell an asset agrees to buy.
- **Spread:** It is the difference between the buying price and the selling price.
- **Marginal Buying/Selling:** It is a buying-selling transaction carried out at the current price of an asset.
- **Liquidity:** The degree to which an asset can be easily bought and sold in the spot market.
- **T+1, T+2, T+3:** Refers to post-transaction clearing periods. T+1 means that the settlement will take place one day after the day the transaction takes place, and T+2 means that the settlement will take place two days after the day the transaction takes place.
- **Settlement Date:** It refers to the date on which a transaction will be financially settled.[1]

Derivatives Market

Definition and Functioning of Future Markets

Futures markets, also known as futures exchanges, are financial markets where standardized contracts for the future delivery of assets, such as commodities, financial instruments, or securities, are bought and sold. These contracts are binding agreements that obligate the parties involved to exchange the underlying asset at a predetermined price and date. Futures markets serve as a platform for hedgers and speculators to manage risk and make bets on future price movements. [2] Futures markets typically consist of a central marketplace where contracts are traded. These markets have a standardized contract size, expiration date, and delivery method. Traders can enter into long (buy) or short (sell) positions in these contracts. The central marketplace provides transparency and liquidity by matching buyers and sellers. Various regulatory bodies oversee the operations to ensure fair and orderly trading. [3]

Types of Futures Markets

There are various types of futures markets, categorized based on the underlying assets they involve. Common types include agricultural futures, energy futures, financial futures, and interest rate futures. Each type has its own unique set of contracts and characteristics, tailored to the specific asset class.

Interaction between Cryptocurrencies and Futures Contracts

The interaction between cryptocurrencies and futures contracts represents the intersection of the digital asset space with traditional financial markets. Cryptocurrency futures allow traders to speculate on the future price of cryptocurrencies without owning the underlying assets, bringing the dynamics of cryptocurrencies into the established world of futures trading. [4]

Creation of Crypto Futures and Why this Interaction is Important

The creation of cryptocurrency futures is significant for various reasons. It provides market participants, including institutional investors, with a regulated and standardized way to gain exposure to the crypto market. [5]

Definition of Option Markets

Option markets are financial markets where participants trade financial instruments known as options. Options are derivatives that give the holder the right, but not the obligation, to buy (call option) or sell (put option) an underlying asset at a predetermined price (strike price) within a specified time frame (expiration date). These markets provide investors with opportunities to hedge risk, speculate on future price movements, and enhance portfolio management strategies. [6]

Functioning of Option Markets

The functioning of option markets involves the buying and selling of options. Market participants, including option buyers and sellers, create a market for these contracts. Option prices, known as premiums, are determined by factors such as the underlying asset's price, the strike price, time to expiration, implied volatility, and interest rates. Call options are typically used for bullish strategies, speculation on price increases, and hedging against short positions, while put options are employed for bearish strategies, speculation on price decreases, and hedging against long positions.

Perpetual Futures

Fundamentals of Perpetual Futures

Perpetual futures, often referred to as perpetual swaps, are a type of derivative financial instrument commonly used in cryptocurrency markets. These contracts are designed to track the price of an underlying asset, such as Bitcoin or Ethereum, and they allow traders to speculate on the future price movements of these assets without owning them. Perpetual futures contracts

have distinct characteristics that set them apart from traditional futures contracts, making them a popular choice among cryptocurrency traders. [7]

How Perpetual Futures Work

1. **Opening a Position:**

Perpetual futures contracts enable traders to take long (buy) or short (sell) positions on an underlying asset without an expiration date. To open a position, a trader typically deposits an initial margin, which acts as collateral for the trade. The leverage available in these contracts allows traders to control a larger position size than their initial margin, increasing both potential gains and losses. The absence of an expiration date means that traders can hold their positions indefinitely, allowing for greater flexibility in managing their trades.

2. **Funding Mechanism:**

One distinctive feature of perpetual futures is the funding mechanism. To prevent the contract's price from deviating significantly from the underlying asset's spot price, a funding rate is periodically paid by one side of the contract to the other. This rate is determined by the market's supply and demand dynamics. If the perpetual contract's price is above the spot price, long positions pay short positions, and vice versa. The funding mechanism ensures that the contract remains closely aligned with the spot market.

3. **Continuous Trading:**

Unlike traditional futures contracts with fixed expiration dates, perpetual futures contracts trade continuously, 24/7. This continuous trading is enabled by the funding mechanism and the absence of a maturity date, allowing traders to enter and exit positions at any time. The liquidity and availability of trading opportunities in perpetual futures markets make them attractive for both day traders and long-term investors.

4. **Risk Management:**

Risk management is a crucial aspect of trading perpetual futures. Given the high leverage that these contracts offer, traders can face substantial gains or losses in a short period. Effective risk management includes setting stop-loss orders, monitoring funding rates, and managing position sizes to avoid liquidation. Additionally, understanding the risks associated with market volatility and funding rate fluctuations is essential for traders to protect their capital effectively.

Orders

Order Types

Market Order

This order type determines the market price at which a share is currently traded (its current price). Market orders provide immediate buying and selling at the current price, but require liquidity to be executed. Investors generally prefer to place market orders if they want their transactions to be executed very quickly. Share prices are highly variable, positively or negatively, within seconds. Since the market price may be very variable in case of sudden price movements, your order may be executed at a price you do not want. There may be risks such as buying at a higher price or selling at a cheaper price.

Limit Order

This order type allows you to place an order to buy or sell an asset at the price and amount you want. We cannot place a buy order more expensive than the market price or a sell order cheaper than the market price. When placing a limit buy order, this order is placed at the lowest price the stock is expected to reach. Thus, they aim to buy shares at the cheapest price. When placing a limit sell order, this order is placed at the highest price the stock is expected to reach. This time, it aims to make the highest profit by selling its shares at the highest price. However, in order for an order to be executed, a buyer and a seller are needed. If there are not enough stocks in the market to qualify for a limit order, it may be executed partially or not at all.

Stop Order

This order type aims to minimize the risk of loss by adhering to trading strategies. We can buy at a higher price or sell at a lower price. The investor's expectation is that the stock price will continue its movement in the same direction after passing the stop level. Investors set a "stop price" condition for the execution of buy-sell orders at the market price. Thus, for orders to be executed, the stock must reach or exceed the stop price level. When the share price reaches the stop price level, a buy or sell order at the market price becomes active. It is used to place a sell order to reduce the loss if the price falls, or to place a buy order to avoid missing the opportunity if the price rises.

Stop Limit Order

A stop limit order is similar to a stop order with a small difference. When the stop price condition is met, our buy-sell order becomes active not at the market price, but at the limit price we determine. Investors do not aim for the stock to continue uninterrupted when it reaches the stop price level, but for it to continue its previous movement after changing direction. As with a limit order, if the price of the stock does not reach the limit price level or cannot find enough buyers and sellers at that level, your limit order may not be executed.

Following Order

In this order type, the investor can sell the stock at the highest price or buy it at the lowest price. Your sell order follows the rising stock, and your buy order follows the falling stock. Thus, it aims to sell at the highest and buy at the lowest. If the price of the stock reverses the direction of earnings, the order remains fixed in place. Traders do not need to manually move the stop point. The point to be careful is not to set it too far or too close to the market price. Because if it is too far, there may be unnecessary losses; if it is too close, the transaction may be closed before making a profit. [8]

Order Books

An order book is an electronic list of orders to buy and sell assets. It lists the prices buyers and sellers are willing to pay and how many orders were placed for a particular price. In other words, it is used to understand the market demand and price movements of a financial asset. It also increases market transparency. It is constantly updated simultaneously. It also provides a record of past transactions. Some participants may choose to remain anonymous. Order books are also known as “Continuous Book”.[9]

- **Buy (Bid) Orders:** Buy orders are placed at a price below the current price of the asset.
- **Sell (Ask) Orders:** Sell orders are generally placed at a price above the current price of the asset. [8]
- **Order History:** This is the section that shows all transactions that took place in the past.

Crypto Exchanges

Cryptocurrency exchanges are online platforms where digital assets can be bought and sold. These exchanges allow users to exchange different cryptocurrencies, buy cryptocurrencies in exchange for traditional currencies, or sell their cryptocurrencies to other users.

History of Crypto Exchanges

After Bitcoin emerged as the first cryptocurrency, platforms were required to trade them. In March 2010, the first cryptocurrency exchange, bitcoinmarket.com, appeared (it no longer exists). Following this, Mt.Gox was also released in July of the same year. Later, exchanges that were secure, user-friendly and included various cryptocurrencies were developed. Nowadays, there are many crypto exchanges and millions of investors trade on these platforms.

Spot Markets in Crypto Exchange

In spot markets, investors can exchange cryptocurrencies (Bitcoin, Ethereum, etc.) for other cryptocurrencies or traditional currencies. These transactions are carried out on cryptocurrency exchanges. These markets operate 24/7. They are known for their high volatility and liquidity. [10]

Decentralized Exchanges (DEX)

Dex markets are similar to traditional exchanges but operate without a central intermediary. It allows investors to exchange crypto assets directly with each other. It does not store user funds, does not collect personal data about them and does not have servers. There is no authentication and no asset monitoring.

Centralized Exchanges (CEX)

CEX markets have a central intermediary. Represents traditional cryptocurrency exchanges. It allows investors to buy and sell their assets through a platform. Investors store their assets in the exchange's wallets. Popular CEX exchanges include Binance. [11]

Decentralized Finance (DeFI)

It stands for decentralized finance. Instead of traditional finance, it is based on decentralized blockchain. DEFI includes elements such as cryptocurrencies, smart contracts and Decentralized Exchange (Dex) platforms. Investors use their assets in lending, borrowing, swapping, staking and other financial transactions through DeFI platforms. [12]

Trading Indicators

Trading in financial markets is a complex and dynamic endeavor, requiring a plethora of tools and strategies to navigate its intricacies successfully. One such set of tools that has gained prominence over the years is trading indicators. These technical tools offer traders, investors, and analysts an array of data-driven insights that help inform trading decisions.

Historical Development of Trading Indicators

The use of trading indicators can be traced back to the early days of financial markets. In the 18th century, Japanese rice trader Munehisa Homma developed candlestick charts, which laid the foundation for modern technical analysis. Since then, various trading indicators have been created, refined, and adapted to different asset classes and trading styles. Notable milestones in the development of trading indicators include the creation of moving averages, relative strength index (RSI), and the stochastic oscillator. [13]

Types of Trading Indicators

Trading indicators can be broadly categorized into several types, each with its unique characteristics and applications. Some common types include:

1. Trend-following Indicators: These indicators help traders identify the prevailing market trend, allowing them to buy in an uptrend and sell in a downtrend. Examples include moving averages and the Moving Average Convergence Divergence (MACD). [14]
2. Momentum Indicators: These indicators measure the speed and strength of price movements, assisting traders in identifying potential trend reversals. Examples include the Relative Strength Index (RSI) and the Stochastic Oscillator. [15]
3. Volatility Indicators: Volatility indicators provide insights into market volatility, helping traders assess the risk associated with a particular asset. The Bollinger Bands and Average True Range (ATR) are examples of such indicators. [16]

4. Volume Indicators: Volume indicators focus on trading volume, helping traders understand the strength of price movements. On-Balance Volume (OBV) and the Money Flow Index (MFI) are well-known volume indicators. [17]
5. Oscillators: Oscillators, like the Commodity Channel Index (CCI) and the Relative Strength Index (RSI), are used to identify overbought and oversold conditions in the market. [18]

Applications of Trading Indicators

Trading indicators are essential tools for traders and investors. They offer numerous applications, including:

1. Signal Generation: Trading indicators generate buy and sell signals, helping traders make timely and informed decisions.
2. Risk Management: By providing insights into market conditions, indicators assist in setting stop-loss orders, determining position sizes, and managing risk effectively.
3. Confirmation of Analysis: Trading indicators can validate or invalidate conclusions drawn from other forms of analysis, such as fundamental analysis or chart patterns.
4. Strategy Development: Traders use indicators to create and refine trading strategies that align with their goals and risk tolerance.

Effectiveness and Limitations of Trading Indicators

While trading indicators are invaluable tools, they are not without limitations. Their effectiveness varies depending on market conditions and the asset being traded. Common challenges include:

1. Lagging Nature: Many indicators are lagging in nature, meaning they provide information about past price movements. Traders must be aware of this delay when making decisions.
2. False Signals: Indicators can produce false signals during ranging or sideways markets, leading to potential losses if not used judiciously.
3. Over-Reliance: Traders may become overly reliant on indicators, neglecting other essential aspects of trading, such as risk management and psychological discipline.

4. Interpretation Complexity: Understanding and interpreting trading indicators require a degree of expertise, which can be a barrier for novice traders.

Commonly Used Trading Indicators

Relative Strength Index (RSI):

The Relative Strength Index, commonly known as RSI, is a momentum oscillator that measures the speed and change of price movements. RSI oscillates between 0 and 100 and is typically used to identify overbought and oversold conditions in the market. A reading above 70 suggests overbought conditions, indicating a potential reversal, while a reading below 30 suggests oversold conditions, indicating a potential upward reversal. [19]

The Relative Strength Index (RSI) is calculated using the following formula:

RSI = 100 - [100 / (1 + RS)]

Where RS (Relative Strength) is the average of 'n' days' up closes divided by the average of 'n' days' down closes. 'n' is typically set to 14.

Moving Average Convergence Divergence (MACD):

The MACD is a trend-following momentum indicator that displays the relationship between two moving averages of a security's price. It consists of a MACD line (the difference between the 12-period and 26-period exponential moving averages) and a signal line (a 9-period exponential moving average of the MACD line). Traders use MACD crossovers as buy or sell signals, with bullish crossovers (MACD line crossing above the signal line) signaling potential long positions and bearish crossovers (MACD line crossing below the signal line) suggesting potential short positions. [20]

The MACD is calculated as follows:

MACD Line = 12-period EMA - 26-period EMA Signal Line = 9-period EMA of MACD Line MACD Histogram = MACD Line - Signal Line

Simple Moving Average (SMA) and Exponential Moving Average (EMA):

Moving averages, both simple and exponential, are trend-following indicators that smooth out price data over a specified time period. The SMA gives equal weight to each data point in the calculation, whereas the EMA assigns more weight to recent prices. Traders use moving averages to identify trends, support and resistance levels, and crossovers for potential trading signals. [21]
The Simple Moving Average (SMA) is calculated by adding up the closing prices over a specific period ('n') and then dividing by 'n'. The Exponential Moving Average (EMA) gives more weight to recent prices, and it is calculated using a formula like this:
EMA = (Closing Price - EMA[previous day]) * (2 / (n + 1)) + EMA[previous day]
'n' is the number of periods, typically 20 or 50, depending on the trader's preference.

Bollinger Bands:

Bollinger Bands consist of a simple moving average (typically 20 periods) and two standard deviation bands (typically set at 2 standard deviations above and below the moving average). Bollinger Bands help traders assess price volatility and identify potential overbought or oversold conditions. When the price touches or crosses the upper band, it may indicate overbought conditions, while touching or crossing the lower band may suggest oversold conditions. [22]
Bollinger Bands consist of three lines: the middle band (SMA), the upper band, and the lower band. The upper band is calculated as the sum of the middle band and twice the standard deviation of the price over a specified period ('n'). The lower band is calculated as the middle band minus twice the standard deviation. The standard deviation is calculated from the closing prices over 'n' periods.

Average True Range (ATR):

The Average True Range is a volatility indicator that measures the average range between the daily high and low prices over a specific time period. ATR provides traders with insights into the level of price volatility and helps them determine appropriate stop-loss and take-profit levels. A

higher ATR suggests higher volatility and, consequently, larger position size or wider stops may be warranted. [23]

ATR is calculated as the average of the true ranges over a specified period ('n'). The true range is the largest of the following three values: • Current high minus current low. • Absolute value of the current high minus the previous close. • Absolute value of the current low minus the previous close. A higher ATR value indicates greater price volatility.

Stochastic Oscillator:

The Stochastic Oscillator is a momentum oscillator that compares the closing price of an asset to its price range over a specified period. It generates values between 0 and 100. Traders use the Stochastic Oscillator to identify overbought and oversold conditions. Readings above 80 suggest overbought conditions, while readings below 20 suggest oversold conditions. [24]

The Stochastic Oscillator is calculated using the following formula:

%K = [(Closing Price - Lowest Low in 'n' periods) / (Highest High in 'n' periods - Lowest Low in 'n' periods)] * 100

%D is a smoothed version of %K and is calculated as a simple moving average of %K over 'm' periods. Common values for 'n' and 'm' are 14 and 3, respectively. The Stochastic Oscillator provides values between 0 and 100, with readings above 80 indicating overbought conditions and readings below 20 indicating oversold conditions.

Algorithmic Trading

Algorithmic trading, also known as algo-trading, refers to the process of using computer algorithms to automate and execute trading strategies. These algorithms are designed to analyze market data, identify profitable opportunities, and execute trades at high speeds, which are often impossible for human traders to achieve manually. Algorithmic trading utilizes various quantitative models, statistical analysis, and mathematical computations to make informed decisions based on predetermined parameters and market conditions. The key advantage of

algorithmic trading lies in its ability to swiftly process large volumes of data and execute trades without human intervention, thereby minimizing the impact of emotional and impulsive decision-making. This approach also enables traders to capitalize on even the slightest market inefficiencies and capture profitable opportunities in a timely manner.

In the AI-based grid trading strategy builder project, algorithmic trading plays a pivotal role in automating the execution of grid trading strategies based on predetermined rules and parameters. By integrating AI techniques, such as machine learning algorithms and deep learning models, the system can adapt to dynamic market conditions and continuously optimize the grid trading strategy for maximum profitability. Through the utilization of historical and real-time market data, the AI-based system can identify trends, patterns, and anomalies, enabling it to make data-driven decisions and adjust the grid trading parameters accordingly. Furthermore, the application of algorithmic trading in this project enhances the overall efficiency, accuracy, and speed of trade executions, while simultaneously mitigating risks and ensuring a more systematic and disciplined approach to trading in the cryptocurrency market.

Grid Trading

Grid trading is a well-established trading strategy that has gained popularity among traders seeking to profit from market volatility. This strategy involves placing buy and sell orders at predetermined price levels, creating a grid-like pattern.

Grid trading has a rich history dating back to manual trading strategies. It evolved from early attempts to exploit market fluctuations by traders who sought a systematic approach to trading. The progression from manual methods to automated grid trading systems represents a significant development in the strategy's evolution.

Key Concepts

Grid trading is based on a set of fundamental concepts that are essential to understanding how this trading strategy operates effectively. These key concepts form the building blocks of grid trading and guide the decision-making process of traders and trading bots:

- Grid Spacing: Grid spacing, often referred to as grid interval or step size, is a critical concept in grid trading. It defines the price levels at which buy and sell orders are placed. Traders determine the spacing based on market conditions, volatility, and their risk tolerance. A smaller grid spacing means more frequent trades, while a larger spacing leads to fewer trades but potentially larger price ranges for profit.
- Order Placement: Grid trading involves the systematic placement of buy and sell orders at predetermined price levels. Orders are typically evenly spaced above and below the current market price. This systematic approach allows traders to take advantage of price fluctuations while maintaining a structured trading plan.
- Grid Size: The grid size, often referred to as the number of grid levels, determines the range of prices covered by the grid. It influences the total exposure of the trading strategy. A larger grid size covers a wider price range but may require more capital and can lead to larger drawdowns. Smaller grid sizes offer tighter control but may limit potential profits.
- Grid Direction: Grid trading can be implemented in different directions, depending on market expectations. The most common approaches include bidirectional grids (buy and sell orders on both sides of the market price) and unidirectional grids (buy or sell orders on only one side). The choice of grid direction depends on a trader's outlook on market direction and risk tolerance.
- Lot Sizing: Lot sizing, also known as position sizing, is the determination of the number of lots or contracts traded at each grid level. Proper lot sizing is crucial for managing risk and capital efficiently. Traders must strike a balance between maximizing profit potential and minimizing risk exposure.
- Grid Origination: Grid origination refers to the initial placement of buy and sell orders when starting a grid trading strategy. It can be based on current market conditions or specific technical indicators. Traders often adapt their grid origination to align with their market analysis and risk management strategy.
- Grid Depth: The depth of a grid refers to the number of price levels covered by the grid. Deeper grids involve more buy and sell orders and provide greater exposure to price

movements. Traders must consider their risk tolerance and market conditions when deciding on the depth of their grid.

Advantages of Grid Trading

Grid trading has gained popularity among traders due to several notable advantages:

- Consistent Returns in Ranging Markets
- Reduced Emotional Involvement
- Ability to Trade Multiple Assets Concurrently
- Enhanced Risk Control
- Potential for Compounding Returns
- Versatility in Market Conditions
- Reduced Market Timing Requirements
- Potential for Passive Income

Disadvantages of Grid Trading

While grid trading has its merits and can be a profitable strategy in certain market conditions, it is not without its disadvantages. Traders need to be aware of these drawbacks to make informed decisions and effectively manage the risks associated with grid trading:

- Drawdown Risks
- Market Conditions
- Continuous Monitoring
- Capital Requirements
- Psychological Stress
- Over-Optimization
- Slippage and Latency
- Market Gaps
- Unpredictable Events

Tools

In this section of the literature review, we will present the tools that can be utilized or are planned to be used in the implementation of the project. The pathway to design this project is as follows: • Programming Language Selection • Algorithm Development • Data Analysis and Management • Testing and Optimization • Security and Risk Management • APIs and SDKs

Programming Language Selection

The selection of an appropriate programming language holds paramount importance when embarking on an AI-based grid trading bot project. This decision profoundly impacts various facets of the project, including seamless integration with AI libraries, efficient speed and performance, code readability and ease of maintenance, compatibility with exchange APIs, effective processing of market data, robust security measures, and the availability of a rich set of development tools. Making an informed choice in programming language selection thus becomes pivotal for the successful and streamlined execution of the AI-based grid trading bot project. An AI-based grid trading bot project can be implemented using a variety of programming languages. Some of the notable options include: [25]

Python

1) Rich Library and Workspace Support: Python offers a rich collection of libraries that facilitate the analysis, processing, and visualization of data used in crypto trading. Considering the scope and objectives of the project, a variety of Python libraries can significantly enhance the development of tools and applications aligned with the grid trading strategy in the cryptocurrency market:

• **ccxt:** This comprehensive library facilitates the integration of various cryptocurrency exchange APIs, including Binance, Bitfinex, and others. Its extensive features enable efficient order management, real-time market data retrieval, and analysis, crucial for making informed trading decisions.

• **Backtrader:** With its extensive backtesting capabilities, Backtrader empowers the system to simulate trading strategies using historical data, optimize parameters, and evaluate the potential profitability of the grid trading approach. Its flexibility allows for the customization of trading strategies, catering to the complexities of the cryptocurrency market.

• **TA-Lib (Technical Analysis Library):** TA-Lib provides an array of technical indicators such as Moving Averages, RSI (Relative Strength Index), and MACD (Moving Average Convergence Divergence), essential for market trend analysis and identification of potential buying or selling opportunities. Its integration enhances the system's ability to adapt to dynamic market conditions and adjust the grid trading strategy accordingly.

• **PyAlgoTrade:** With its algorithmic trading capabilities, PyAlgoTrade facilitates the automatic execution of buy/sell orders, position tracking, and portfolio management. Its event-driven architecture allows for the efficient handling of market events, enabling the system to respond swiftly to changes in market conditions and execute trades based on predefined parameters.

• **Dash and Flask:** These libraries enable the creation of user-friendly and interactive interfaces, such as command-line tools and web-based dashboards. Leveraging these libraries, the team can develop intuitive monitoring and management tools, providing real-time insights into market trends, order executions, and portfolio performance, thereby facilitating informed decision-making.

• **NumPy and Pandas:** These fundamental data manipulation libraries offer extensive support for data analysis, manipulation, and preprocessing. Their efficient handling of large datasets and array operations enables the system to process market data effectively, identify patterns, and derive meaningful insights crucial for optimizing the grid trading strategy.

• **Scikit-learn:** Integration of Scikit-learn can empower the system to leverage machine learning models for market trend prediction, price forecasting, and risk assessment. By implementing various supervised and unsupervised learning algorithms, the system can adapt to dynamic market conditions, improve decision-making accuracy, and optimize the grid trading parameters in response to changing market dynamics. [26]

2) Binance API Integration: Python provides libraries compatible with the APIs of Binance and other crypto exchanges. This integration facilitates the automatic implementation of crypto

trading strategies and efficient processing of data. It simplifies functions such as placing buy/sell orders, tracking transactions, and analyzing market movements.

3) Simplicity and Readability: Python's simple and understandable syntax facilitates comprehension for team members working on different stages of the project. It strengthens collaboration within the team and enhances the project's overall efficiency.

4) Rapid Prototyping: Python is well-suited for rapid prototyping, enabling quick testing of new trading strategies and swift implementation of ideas. This capability can shorten the project timeline and offer a more effective development process.

5) Compatibility and Flexibility: Python seamlessly integrates with different systems and works compatibly with other languages. This aids in the seamless integration of various project components and enables flexible adjustments to meet project requirements. Moreover, Python's ability to operate on various platforms and its support from a large user base can help the project team reach a broader audience.

Alongside all the advantages that can contribute to this project, choosing Python as the programming language also entails some notable disadvantages. Here are some significant drawbacks:

1) Performance Limitations: Python can be slower compared to lower-level languages, particularly in tasks involving intensive computation or large-scale data processing.

2) Mobile App Development Challenges: Python may not be the most preferred language for mobile app development, as it may not offer the same performance optimization as some other languages specifically designed for mobile platforms.

3) Debugging and Maintenance Complexities: Python's dynamically typed nature and extensive flexibility might introduce challenges in debugging and maintaining large-scale projects, potentially impacting the overall project robustness.

4) Security Concerns: Python might introduce security challenges, especially in web application development, as certain coding practices can lead to vulnerabilities and risks.

5) Resource Intensiveness in Mobile Devices: Python's resource-intensive nature on mobile devices can lead to performance issues, especially on devices with limited resources, which might affect the user experience.

JavaScript

1) Abundant Web Development Resources: JavaScript is widely recognized for its extensive use in web development, offering a diverse range of libraries and frameworks for creating interactive and user-friendly web interfaces. For this crypto trading bot project, JavaScript can leverage its robust capabilities to develop responsive and dynamic web-based tools that facilitate efficient monitoring and management of trading activities.

2) Node.js for Backend Support: Leveraging Node.js, a JavaScript runtime environment, can provide the project with a robust backend support system. Its event-driven architecture and non-blocking I/O operations enable efficient data processing, real-time updates, and seamless integration with various data sources, essential for effective market analysis and automated trading execution.

3) Asynchronous Programming: JavaScript's asynchronous programming model allows for the execution of multiple operations simultaneously, enhancing the system's responsiveness and performance. This feature is particularly advantageous for handling real-time market data updates and ensuring swift execution of trading orders based on dynamic market conditions.

4) Frontend Development Flexibility: JavaScript, combined with popular frontend frameworks like React.js or Angular, facilitates the development of dynamic and responsive user interfaces, essential for providing traders with intuitive dashboards, customizable trading tools, and real-time market insights.

While JavaScript offers numerous advantages for web-based applications and frontend development, it also presents certain limitations and considerations:

1) Performance Constraints: JavaScript's performance may be comparatively slower than lower-level languages, especially in tasks that involve complex computations or large-scale data processing, potentially affecting the system's real-time data analysis and trade execution capabilities.

2) Security Vulnerabilities: JavaScript's client-side execution model can expose applications to security vulnerabilities if not implemented with appropriate security measures. Cross-site scripting (XSS) attacks and other security threats are potential risks that require thorough consideration and diligent coding practices.

3) Limited Mobile Development Scope: While JavaScript is compatible with mobile development frameworks such as React Native and Ionic, it may face constraints in terms of performance optimization and native functionality integration, which can impact the overall user experience and responsiveness of the trading application.

4) Debugging Complexity: JavaScript's loosely typed nature can introduce challenges in debugging and maintaining large-scale applications, necessitating meticulous testing and debugging practices to ensure the reliability and stability of the trading system.

Java

1) Robust Ecosystem and Cross-Platform Support: Java boasts a robust ecosystem with a wide array of libraries, frameworks, and tools that facilitate the development of scalable and cross-platform applications. In the context of this crypto trading bot project, Java can harness its rich set of libraries for data processing, algorithm implementation, and system integration, ensuring the efficient execution of trading strategies and comprehensive market analysis.

2) High Performance and Scalability: Java's efficient virtual machine (JVM) and robust multi-threading capabilities enable the system to handle complex computations, large datasets, and high-frequency trading operations seamlessly. Its scalability ensures the system's adaptability to evolving market dynamics and the management of multiple trading activities concurrently.

3) Enterprise-Grade Security Features: Java's built-in security features, including its robust authentication mechanisms, access controls, and encryption libraries, provide a secure environment for developing and deploying mission-critical financial applications. Java's emphasis on security compliance and its ability to mitigate potential vulnerabilities make it a reliable choice for implementing secure and compliant trading solutions.

4) Versatile Integration Capabilities: Java's seamless integration with various APIs, databases, and external systems facilitates the efficient exchange of data, real-time market updates, and streamlined order execution. Its compatibility with popular messaging protocols and financial data feeds enables the system to access real-time market data and execute trades with minimal latency and data loss.

While Java offers numerous advantages for building robust and secure trading applications, certain considerations should be taken into account:

1) Development Complexity: Java, as a statically typed language, may involve a more verbose syntax and a steeper learning curve compared to dynamically typed languages. Ensuring proficient Java programming skills within the project team is essential for maintaining code quality and optimizing the system's performance.

2) Memory Management Overhead: Java's automatic memory management through garbage collection can introduce occasional performance overhead, particularly in latency-sensitive applications. Implementing efficient memory management strategies and optimizing resource utilization are crucial for ensuring the system's responsiveness and stability.

3) Longer Development Cycle: Java's strict adherence to coding standards and comprehensive testing practices may contribute to a longer development cycle compared to other languages. Prioritizing thorough testing, code reviews, and quality assurance processes is imperative for delivering a reliable and high-performance trading system.

C++

1) High-Performance Computing Capabilities: C++ excels in performance-critical applications, leveraging its efficient memory management and low-level hardware control. With its robust computational capabilities and minimal runtime overhead, C++ enables the development of high-frequency trading systems that can process large datasets and execute complex trading algorithms with low latency.

2) Extensive Standard Library and Frameworks: C++ offers an extensive standard library and a variety of specialized frameworks for financial analysis and algorithmic trading. Libraries such as Boost and QuantLib provide a rich set of functionalities for mathematical computations, statistical analysis, and the implementation of advanced trading strategies, enhancing the system's analytical capabilities and trade execution efficiency.

Despite its numerous advantages, C++ presents certain challenges that should be considered:

1) Complexity and Development Time: C++'s complex syntax and memory management require meticulous attention to detail and advanced programming expertise. Developing and maintaining

C++ applications may involve a longer development cycle compared to higher-level languages, emphasizing the importance of rigorous testing and code optimization practices.

2) Platform Compatibility and Portability: While C++ offers excellent performance and system-level control, ensuring cross-platform compatibility and portability may require additional effort and considerations. Implementing robust cross-platform development practices and thorough testing across different environments are essential for delivering a seamless and reliable trading system across various platforms and operating systems.

Algorithm Development

The algorithm development phase is a critical component of the project, focusing on the design, implementation, and refinement of the grid trading strategy to ensure effective and profitable trading operations in the cryptocurrency market. This phase involves several intricate steps aimed at creating a robust and adaptive algorithm tailored to the specific objectives outlined in the project description. Strategy Formulation initiates the process with a comprehensive analysis of the grid trading strategy's core principles and objectives, defining specific parameters, risk management protocols, and market indicators. Code Architecture Design emphasizes creating a well-structured and modular code architecture for scalability and maintainability, enabling seamless integration with external APIs and data sources. Backtesting and Validation entail rigorous evaluation under historical market conditions to optimize parameter configurations and benchmark performance. Real-Time Market Integration facilitates live trading operations and dynamic decision-making based on up-to-date market trends and price fluctuations, ensuring timely order execution and position management. Adaptive Strategy Refinement includes continuous monitoring and dynamic adjustments to adapt to evolving market dynamics, optimizing trading strategies and response mechanisms. Risk Management Protocols are integrated within the algorithm to mitigate potential financial risks, minimizing exposure to market volatility and safeguarding the trading portfolio against adverse market conditions. [27]

Data Analysis and Management

The Data Analysis and Management phase involves the compilation, analysis, processing, and management of market data used within the project, crucial for supporting data-driven decision-making processes and optimizing trading strategies based on data. Identification of Data Sources entails determining market data sources, often utilizing exchange APIs such as Binance and custom data providers. Data Processing and Cleaning involve utilizing Python's Pandas library for tasks like data transformation and cleaning, while Data Analytics Applications involve using visualization libraries like Matplotlib and Seaborn for graphical representation and statistical packages for various analyses. Data Storage and Backup operations can utilize SQL-based database management systems (DBMS) such as MySQL, PostgreSQL, and NoSQL databases, with regular backup procedures enhancing data security. Data Security and Privacy measures can be implemented using encryption algorithms and security protocols to ensure data protection within the project. [28]

Testing and Optimization

The Testing and Optimization phase is pivotal for evaluating and enhancing the grid trading strategy's performance. Backtesting Strategies use historical data, employing Python libraries like Backtrader and Zipline. Performance Metrics and Analysis, including metrics like Sharpe ratio and maximum drawdown, guide decision-making. Parameter Optimization fine-tunes strategy parameters using algorithms like genetic algorithms or particle swarm optimization. Stress Testing evaluates system robustness under extreme conditions, informing risk management. Iterative Refinement continuously improves the algorithm by addressing weaknesses for enhanced adaptability and profitability. [29]

Security and Risk Management

Security and Risk Management in cryptocurrency projects involves robust measures such as data encryption, privacy protocols, access control, and authorization. Vulnerability assessments and penetration testing identify weaknesses, while disaster recovery and contingency planning ensure business continuity. Compliance with regulatory standards like GDPR and AML/KYC enhances credibility and mitigates legal risks. [30]

Software Requirement Specification

3. Introduction

3.1 Purpose of this Document

The primary objective of this document is to comprehensively articulate and elucidate the essential requirements, functionalities, and parameters that constitute the foundation for the development of the “AI-Based Grid Trading Strategy Builder” software. It serves as a definitive guide and reference for all stakeholders involved in the project, including developers, designers, testers, and project managers. By outlining the overarching goals, specifications, and constraints of the software, this document establishes a shared understanding among team members, ensuring alignment with the project’s objectives throughout its lifecycle. Furthermore, it facilitates effective communication and collaboration, contributing to the successful realization of the software in accordance with the identified needs and expectations.

1.2 Scope of the Project

The scope of the “AI-Based Grid Trading Strategy Builder” project encompasses the development of a comprehensive platform designed to cater to the diverse needs of individuals interested in cryptocurrency trading. This software aims to provide users with a multifaceted experience, incorporating various features that range from basic account registration to advanced algorithmic trading strategies.

The project's scope extends to the following key components:

1.2.1 Account Management

The software will facilitate user registration, allowing individuals to create accounts within the system. This includes the provision of essential details for the establishment of a personalized

user profile. Furthermore, the system will implement a robust authentication mechanism, ensuring the security and integrity of user accounts through email confirmation.

1.2.2 Account Upgrading Mechanism

To enhance the user experience, a fee-based account upgrading system will be implemented. This system will enable users to access premium services and features beyond the basic functionalities. The upgrade process will be seamlessly integrated into the platform, providing users with a clear pathway to unlock advanced capabilities.

1.2.3 Strategy Generation

The software will empower users to actively participate in algorithmic trading by offering a sophisticated strategy generation module. Users can input specific parameters such as currency pairs and grid sizes to tailor trading strategies according to their preferences and risk appetite. This feature aims to cater to both novice users seeking simplicity and experienced traders requiring customization.

1.2.4 Binance API Integration

To facilitate automated trading, the system will establish integration with the Binance cryptocurrency exchange through the utilization of API keys. Users will be able to securely input their Binance API key and secret, enabling seamless communication between the software and the exchange. This integration is a pivotal element in automating the execution of user-generated trading strategies.

1.2.5 Automated Trading

The software will feature an automated trading mechanism that executes trades based on the strategies generated by users. This functionality is designed to streamline the trading process,

providing efficiency and precision in the execution of predefined strategies. Users will have the ability to monitor and manage their automated trading activities through an intuitive dashboard.

2. General Description

2.1 Glossary

AI: Artificial Intelligence

API: Application Programming Interface

Binance: Cryptocurrency exchange platform

2.2 User Characteristics

The user base for the “AI-Based Grid Trading Strategy Builder” software encompasses a diverse range of individuals engaged in cryptocurrency trading. The system is designed to cater to users at various skill levels, accommodating both novice traders seeking intuitive functionalities and experienced traders looking for advanced features. The platform is crafted to provide a seamless and adaptable experience, ensuring accessibility for users with varying degrees of expertise in the dynamic field of cryptocurrency trading. As the software caters to a broad spectrum of users, its user interface and functionalities are thoughtfully designed to be inclusive and accommodating, fostering an environment conducive to effective engagement for all levels of traders.

2.3 Overview of Functional Requirements

The software will allow users to:

- Register and create accounts
- Upgrade accounts through a fee-based system
- Input parameters to generate trading strategies
- Connect to Binance using API keys
- Execute trading strategies automatically

2.4 General Constraints and Assumptions

- Users are responsible for their trading decisions and financial outcomes.
- Binance API key and secret information will be kept confidential and securely stored.

3. Specific Requirements

3.1 Interface Requirements

3.1.1 User Interface

The user interface for the “AI-Based Grid Trading Strategy Builder” will be designed as a web application, providing users with a seamless and intuitive experience. The following features will be incorporated into the web interface:

3.1.1.1 Registration and Login

Users will be able to register by providing necessary details, including username, email, and password. A secure authentication process will be implemented, requiring users to confirm their registration via email.

3.1.1.2 Account Upgrade

Within the web interface, users can easily navigate to upgrade their accounts by choosing from various premium service options. Clear and concise information about the benefits of each premium tier will be presented, facilitating an informed decision-making process.

3.1.1.3 Strategy Input Forms

The platform will feature user-friendly forms where users can input specific parameters to generate personalized trading strategies. These forms will include fields for selecting the desired currency pairs, defining grid sizes, and customizing other relevant settings. Clear instructions and tooltips will guide users through the process.

3.1.1.4 Dashboard

Upon login, users will be directed to a comprehensive dashboard displaying relevant information. The dashboard will showcase generated trading strategies, ongoing trades, and account status. Interactive charts and visual representations will be incorporated to enhance the user's understanding of their trading activities.

3.1.1.5 Account Management

The web interface will provide users with a dedicated section for managing their accounts. This includes options for updating personal information. Account-related notifications and alerts will also be accessible through this section.

3.1.1.6 Help and Support

A user-friendly help and support section will be integrated into the interface, offering documentation, FAQs, and contact options for customer support. This ensures that users can easily find assistance and information when needed.

3.1.2 Hardware Interface

The web application will be compatible with standard hardware configurations, ensuring accessibility for a broad range of users. It will be optimized for various devices, including desktops, laptops, and tablets, to provide a consistent and responsive experience.

3.1.3 Software Interface

The web application will interact seamlessly with the Binance cryptocurrency exchange through the Binance API. The integration will facilitate real-time data exchange, enabling accurate strategy generation and automated trade execution.

3.1.4 Communication Interfaces

Communication between the users and the system will occur through the web-based interface. Additionally, the system will communicate with the Binance exchange using the Binance API, ensuring secure and reliable data transmission for trading activities.

3.2 Detailed Description of Functional Requirements

- User Registration and Authentication
    - Users can register by providing necessary details.
    - Authentication through email confirmation.
- Account Upgrade
    - Users can upgrade their accounts by paying a fee.
    - Premium accounts provide access to advanced features.
- Strategy Generation
    - Users can input parameters (e.g., currency pair, grid size) to generate trading strategies.
- Binance API Integration
    - Users can input Binance API key and secret for account integration.
- Automated Trading
    - The system executes trades automatically based on user-generated strategies.

3.3 Non-Functional Requirements

3.3.1 Security Requirements

3.3.1.1 Data Encryption

User data, including personal details and API keys, must be encrypted using industry-standard cryptographic algorithms. The encryption protocols should ensure the confidentiality and integrity of sensitive information stored within the system.

3.3.1.2 Secure Storage

The system must employ secure storage mechanisms to safeguard user data and API keys. Access controls and encryption should be applied to prevent unauthorized access or data breaches.

3.3.1.3 Authentication Mechanisms

Robust authentication processes, such as two-factor authentication (2FA), should be implemented to ensure that only authorized users can access the system. This is crucial for preventing unauthorized activities and protecting user accounts.

3.3.2 Performance Requirements

3.3.2.1 Concurrent User Handling

The system should be designed to efficiently handle a large number of concurrent users without compromising performance. This includes optimizing server resources, minimizing latency, and ensuring a seamless user experience even during peak usage periods.

3.3.2.2 Response Time Optimization

The response times for critical functions, such as strategy generation and trade execution, should meet predefined benchmarks. The system must be responsive to user inputs, providing real-time feedback and maintaining an optimal user experience.

3.3.3 Reliability Requirements

3.3.3.1 Minimization of Downtime

The system must strive to minimize downtime for critical functions. Regular maintenance and updates should be scheduled during non-peak hours to reduce the impact on users. In the event of planned downtime, users should be notified in advance.

3.3.3.2 Backup and Recovery

A robust backup and recovery mechanism should be in place to safeguard against data loss or system failures. Regular backups of user data, configurations, and system states should be performed, and efficient recovery procedures should be documented and tested.

3.3.4 Scalability Requirements

3.3.4.1 System Architecture

The system architecture must be designed for scalability to accommodate the anticipated growth in user demand. Scalability should be achieved through modular and extensible components, allowing for seamless integration of additional resources as needed.

3.3.5 Usability Requirements

3.3.5.1 User Interface Design

The user interface should prioritize usability, ensuring that users can easily navigate the platform and access its features. Intuitive design principles, clear instructions, and helpful tooltips should be implemented to enhance the overall user experience.

4. Analysis-UML

4.1 Use Cases

4.1.1 Use Case Diagrams

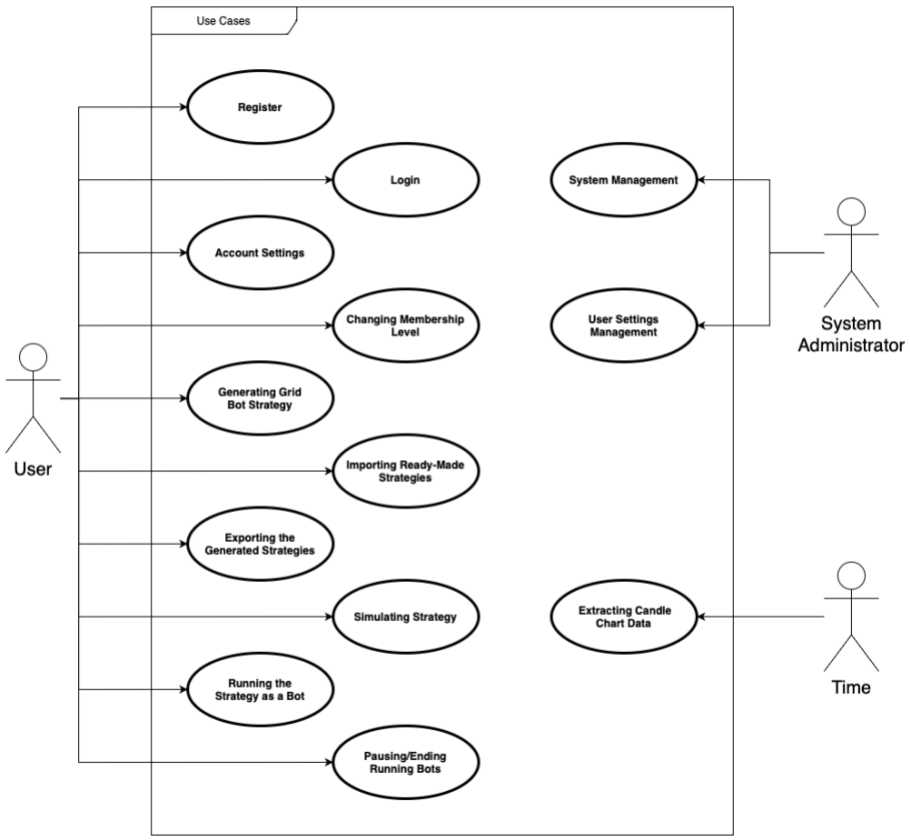

Figure 1 Use Case Diagrams

4.1.2 Description of Use Cases

Use Case 1: Register - User

- Use Case ID: UC01
- Use Case Name: Register
- Description: Allows a user to register by providing necessary details such as username, email, and password.
- Precondition: User is not registered.
- Related Use Cases: Login the System (UC02)
- Postcondition: User is registered, and an account is created.
- Main Flow:
    i. User navigates to the registration page.
    ii. User provides required information (username, email, password).
    iii. User submits the registration form.
    iv. System validates the information.
    v. If valid, the system creates a new user account.
    vi. User receives a confirmation and can now log in.

Use Case 2: Login the System - User

- Use Case ID: UC02
- Use Case Name: Login the System
- Description: Allows a registered user to log in by providing valid credentials.
- Precondition: User is registered.
- Related Use Cases: Register (UC01)
- Postcondition: User is authenticated and gains access to the system.
- Main Flow:
    i. User navigates to the login page.
    ii. User provides valid credentials (username, password).
    iii. System validates the credentials.
    iv. If valid, the system logs the user in.
    v. User gains access to the system.

Use Case 3: Account Settings - User

- Use Case ID: UC03
- Use Case Name: Account Settings
- Description: Allows a logged-in user to modify account details and preferences.
- Precondition: User is logged in.
- Related Use Cases: None
- Postcondition: Account settings are updated.
- Main Flow:
    i. User navigates to the account settings page.
    ii. User modifies desired account details.
    iii. User saves the changes.
    iv. System validates and updates the account settings.

Use Case 4: Changing Membership Level - User

- Use Case ID: UC04
- Use Case Name: Changing Membership Level
- Description: Allows a logged-in user to upgrade or downgrade their membership level by paying a fee.
- Precondition: User is logged in.
- Related Use Cases: None
- Postcondition: Membership level is changed.
- Main Flow:
    i. User navigates to the membership level page.
    ii. User selects a new membership level and initiates payment.
    iii. System processes the payment.
    iv. If successful, the system updates the user's membership level.

Use Case 5: Generating Grid Bot Strategy - User

- Use Case ID: UC05
- Use Case Name: Generating Grid Bot Strategy

- Description: Allows a logged-in user to input parameters and generate a grid trading strategy.
- Precondition: User is logged in.
- Related Use Cases: Simulating Strategy (UC08)
- Postcondition: Trading strategy is generated.
- Main Flow:
    i. User navigates to the strategy generation page.
    ii. User inputs parameters (currency pair, grid size).
    iii. User initiates the strategy generation.
    iv. System generates the trading strategy based on user inputs.

Use Case 6: Importing Ready-Made Strategies - User

- Use Case ID: UC06
- Use Case Name: Importing Ready-Made Strategies
- Description: Allows a logged-in user to import predefined trading strategies.
- Precondition: User is logged in.
- Related Use Cases: None
- Postcondition: Imported strategy is added to the user's profile.
- Main Flow:
    i. User navigates to the strategy import page.
    ii. User selects a ready-made strategy file for import.
    iii. System validates and adds the strategy to the user's profile.

Use Case 7: Exporting the Generated Strategies - User

- Use Case ID: UC07
- Use Case Name: Exporting the Generated Strategies
- Description: Allows a logged-in user to export their generated trading strategies.
- Precondition: User is logged in.
- Related Use Cases: None
- Postcondition: Trading strategy is exported.
- Main Flow:

i. User navigates to the strategy export page.
ii. User selects a generated strategy for export.
iii. System exports the selected strategy in the desired format.

Use Case 8: Simulating Strategy - User

- Use Case ID: UC08
- Use Case Name: Simulating Strategy
- Description: Allows a user to simulate the performance of their trading strategy.
- Precondition: User has a generated trading strategy.
- Related Use Cases: Running the Strategy as a Bot (UC09)
- Postcondition: Simulation results are displayed.
- Main Flow:
    i. User navigates to the strategy simulation page.
    ii. User selects a generated strategy for simulation.
    iii. System simulates the performance based on historical data.
    iv. Simulation results are presented to the user.

Use Case 9: Running the Strategy as a Bot - User

- Use Case ID: UC09
- Use Case Name: Running the Strategy as a Bot
- Description: Allows a user to activate their trading strategy to run automatically on the Binance exchange.
- Precondition: User has a generated and simulated trading strategy.
- Related Use Cases: Pausing/Ending Running Bots (UC10)
- Postcondition: Trading bot is running on the Binance exchange.
- Main Flow:
    i. User navigates to the bot activation page.
    ii. User selects a simulated strategy for activation.
    iii. User provides Binance API key and secret.
    iv. System verifies the API key and secret.
    v. If valid, the system activates the trading bot.

Use Case 10: Pausing/Ending Running Bots - User

- Use Case ID: UC10
- Use Case Name: Pausing/Ending Running Bots
- Description: Allows a user to pause or end the execution of their running trading bots.
- Precondition: User has a running trading bot.
- Related Use Cases: Running the Strategy as a Bot (UC09)
- Postcondition: Trading bot is paused or ended.
- Main Flow:
    i. User navigates to the running bots management page.
    ii. User selects a running bot.
    iii. User chooses to pause or end the selected bot.
    iv. System processes the user's choice.

Use Case 11: System Management - System Administrator

- Use Case ID: UC11
- Use Case Name: System Management
- Description: This use case allows system administrators to manage overall system functionality, ensuring efficient control over critical parameters.
- Precondition: Administrator is authenticated.
- Related Use Cases: User Settings Management (UC12)
- Postcondition: The administrator is authenticated with the required privileges.
- Main Flow:
    i. Administrator logs into the system using valid credentials.
    ii. Once authenticated, the administrator navigates to the system management page.
    iii. The administrator reviews and modifies system settings based on predefined parameters.
    iv. Upon completing the modifications, the system validates and updates the settings.
    v. The changes are applied, and the system notifies the administrator of the successful update.

Use Case 12: User Settings Management - System Administrator

- Use Case ID: UC12
- Use Case Name: User Settings Management
- Description: This use case enables system administrators to manage individual user settings, providing granular control over user-specific configurations.
- Precondition: Administrator is authenticated.
- Related Use Cases: System Management (UC11)
- Postcondition: The administrator is authenticated with the necessary privileges.
- Main Flow:
    i. Administrator logs into the system using valid credentials.
    ii. Administrator navigates to the user settings management page.
    iii. Administrator selects a user and modifies settings.
    iv. System updates and applies the changes.

Use Case 13: Extracting Candle Chart Data - Time

- Use Case ID: UC13
- Use Case Name: Extracting Candle Chart Data
- Description: A scheduled batch job regularly extracts candle chart data from Binance and updates it in the system to ensure the availability of the latest market information for strategy generation and simulation.
- Precondition: None
- Related Use Cases: Generating Grid Bot Strategy (UC05), Simulating Strategy (UC08)
- Postcondition: Candle chart data is updated in the system.
- Main Flow:
    i. The batch job is scheduled to run at predefined intervals.
    ii. The system initiates the batch job to extract candle chart data from Binance.
    iii. The batch job retrieves the latest market data for specified currency pairs.
    iv. Extracted data is processed and updated in the system's database.
    v. The system logs the completion of the batch job.

4.2 Functional Modeling (DFD)

4.2.1 Context Diagram (DFD Level - 0)

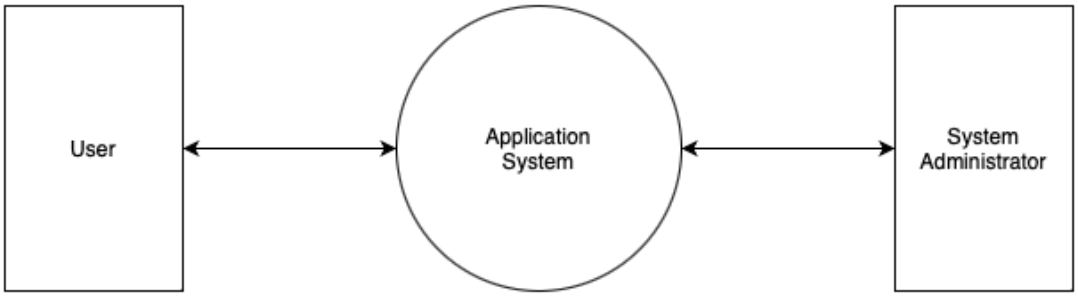

Figure 2 Context Diagram (DFD Level - 0)

4.2.2 DFD Level - 1

Figure 3 DFD Level - 1

4.2.3 DFD Level - 2

4.2.3.1 User DFD Level- 2

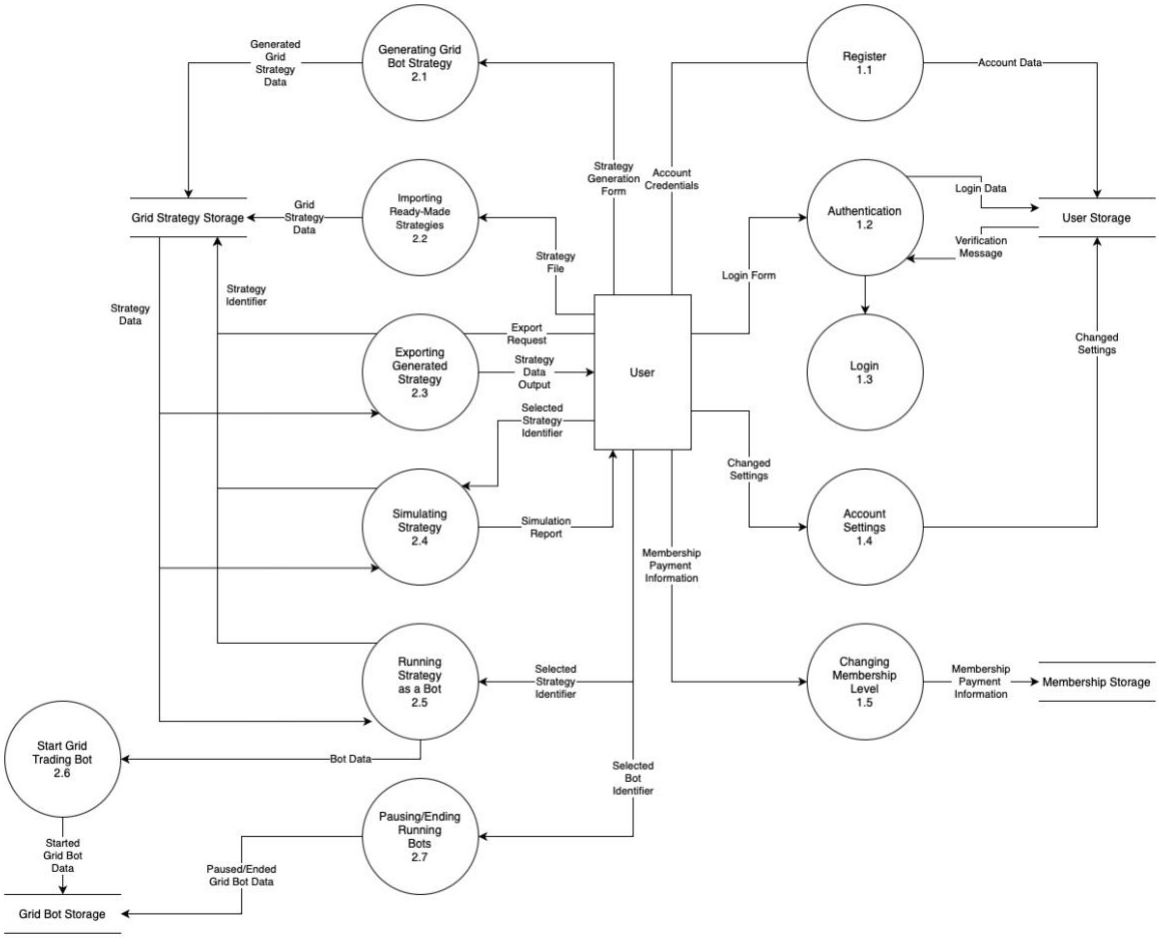

Figure 4 User DFD Level - 2

4.2.3.2 System Administrator DFD Level- 2

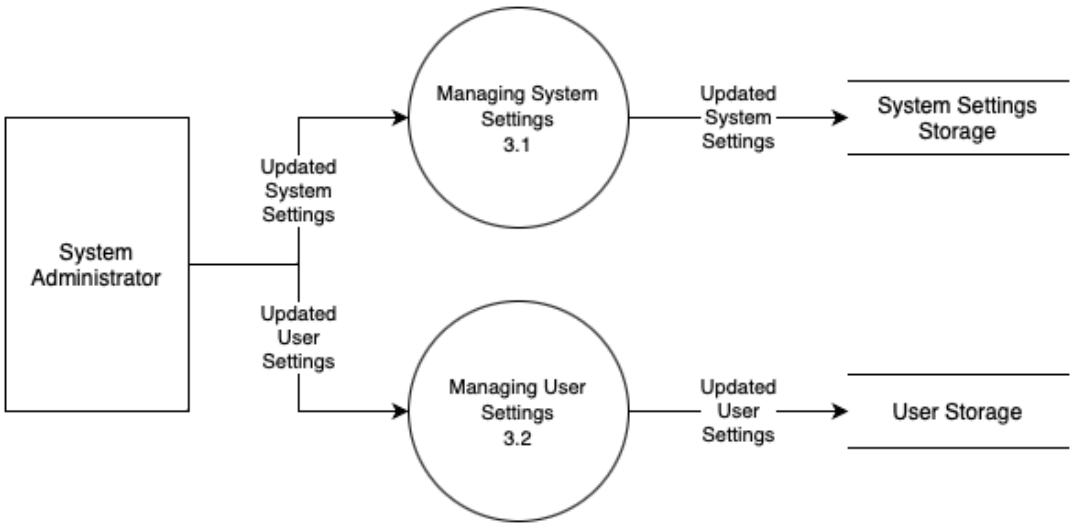

Figure 5 System Administrator DFD Level - 2

Software Design Description

1. Introduction

1.1. Purpose

The purpose of this document is to provide a comprehensive blueprint for the development and implementation of the "AI-Based Grid Trading Strategy Builder" software. This document aims to guide the project team, including developers, architects, and testers, through a detailed design process, ensuring that the final product aligns with the specified requirements and objectives. It serves as a technical roadmap, outlining the system's architecture, components, and interactions, thereby facilitating efficient and effective development. Additionally, this document assists in maintaining a clear understanding of the project's design principles for future reference and updates, ensuring scalability and adaptability of the software to evolving market needs and technological advancements. The SDD plays a crucial role in aligning the development process with the strategic goals of the project, emphasizing reliability, user experience, and advanced algorithmic solutions for optimizing cryptocurrency trading strategies.

1.2. Definitions and Acronyms

- AI (Artificial Intelligence): A branch of computer science dealing with the simulation of intelligent behavior in computers.
- API (Application Programming Interface): A set of routines, protocols, and tools for building software applications, specifying how software components should interact.
- Binance: A global cryptocurrency exchange platform.

- Grid Trading: A quantitative trading strategy that automatically places buy and sell orders at predetermined intervals around a set price.
- Genetic Algorithm: An optimization technique based on the principles of genetics and natural selection, used to solve complex problems by iteratively selecting, combining, and mutating candidate solutions.
- UI (User Interface): The space where interactions between humans and machines occur.
- UX (User Experience): The overall experience of a person using a product, especially in terms of how easy or pleasing it is to use.

2. System Overview

2.1. System Context and Design

The "AI-Based Grid Trading Strategy Builder" is conceptualized as a web-based platform, leveraging advanced AI algorithms for optimizing cryptocurrency trading strategies. At its core, the system is structured to facilitate seamless interaction between users and the cryptocurrency market via the Binance API.

2.1.1. Technical Framework

The system operates on a multi-tier architecture:

1. **Front-End Tier:** The user interface, built for web access, is designed for intuitive navigation, allowing users to effortlessly interact with the platform's features.
2. **Application Tier:** This layer handles the core functionalities - strategy creation, optimization using genetic algorithms, and management of user accounts.
3. **Data Tier:** A robust database system stores user information, trading strategies, and historical market data.

2.1.2. Integration with Binance API

A critical component of the system is its integration with Binance's trading API. This allows for real-time market data retrieval, execution of trades, and monitoring of trading activities based on user-defined strategies. The system ensures security in API communication, particularly in handling sensitive data like API keys.

2.1.3. AI-Driven Strategy Optimization

At the heart of the system is the AI module, employing genetic algorithms to analyze historical market data. This module is designed to identify and optimize the most profitable trading strategies, tailored to user preferences and market conditions.

Pseudocode for a genetic algorithm to optimize grid trading strategies:

1. Initialize:
    - Input: candle_chart_data, user_defined_time_period, parameter_ranges
    - Split candle_chart_data into segments of length user_defined_time_period
    - Generate initial population of grid strategies within parameter_ranges
2. Evaluate:
    - For each strategy in population:
        - Calculate performance over each chart segment
        - Average these performances to get strategy's score
3. Evolution Loop:
    - While improvement is observed over last N generations:
        - Select parent strategies based on their scores (consider diversity)
        - Perform crossover and mutation to create new strategies
        - Evaluate new population using step 2
4. Termination:

- If no improvement over last N generations, exit loop

5. Output:
    - Return the highest scoring strategy from the final population

2.1.4. Scalability and Performance

The design accommodates scalability, capable of handling an increasing number of users and data volume without compromising performance. Cloud-based infrastructure and efficient algorithmic implementations are key considerations to achieve this.

2.2 Background to the Project

The "AI-Based Grid Trading Strategy Builder" is conceived in response to the growing interest and demand in sophisticated cryptocurrency trading tools. With the rise of digital currency trading, particularly on platforms like Binance, there has been an increased need for advanced, automated trading strategies that can adapt to the volatile crypto markets. This project aims to cater to experienced traders familiar with grid bots on Binance, offering them a powerful tool to enhance their trading effectiveness.

The core concept is to leverage AI, specifically genetic algorithms, to analyze historical market data and optimize trading strategies. This approach not only promises to improve the profitability of trades but also introduces a level of customization and adaptability previously unavailable in standard trading bots. The project's uniqueness lies in its AI-driven approach to strategy development, a significant advancement over the traditional, manually-set grid trading methods. By integrating directly with the Binance API, the software ensures seamless execution of trades based on the strategies developed by its AI algorithms. This direct integration is crucial for real-time trading responsiveness, a key factor in the fast-paced crypto trading environment.

The project's inception is grounded in a thorough understanding of the current market needs and technological possibilities, aiming to set a new standard in AI-assisted trading strategies within the cryptocurrency domain.

3. System Design

3.1. Architectural Design

The architectural design of the "AI-Based Grid Trading Strategy Builder" is a multi-layered structure, strategically organized to ensure robust performance, scalability, and security.

- **Front-End Layer:** This layer includes the user interface components, developed using modern web technologies to provide a responsive and intuitive user experience. It is responsible for presenting information to the user and collecting user inputs.
- **Back-End Layer:** This core layer handles business logic, AI algorithm processing, and interaction with the Binance API. It is developed with a focus on high performance and reliability to manage intensive data processing and real-time trading operations.
- **Database Layer:** A secure and scalable database is used for storing user data, trading strategies, historical market data, and transaction logs. It ensures data integrity and quick access to necessary information.
- **API Layer:** The system integrates with the Binance API for executing trades and retrieving market data. This layer handles API requests and responses, ensuring efficient and secure communication with the cryptocurrency exchange.
- **Security and Compliance:** Throughout all layers, security measures are implemented, including data encryption, secure API key management, and compliance with financial regulations.
- **Cloud Infrastructure:** The system is hosted on a cloud platform, offering scalability and high availability. This infrastructure supports handling high user loads and extensive data processing without compromising performance.

3.2. Decomposition Description

3.2.1. System Decomposition

In the AI-Based Grid Trading Strategy Builder project, system decomposition plays a crucial role in breaking down the complex software into smaller, manageable components, enhancing clarity and maintainability. This section will detail each significant component of the system and its function.

1. **User Management Module:** Handles account creation, authentication, and account upgrades. Integrates security measures for user data protection.
2. **Strategy Builder Engine:** The core component where users input parameters to create trading strategies. This engine uses a genetic algorithm to analyze historical market data and optimize strategies for profitability.
3. **Market Data Analyzer:** Responsible for fetching and processing historical market data from Binance and other sources. This data is crucial for the Strategy Builder Engine.
4. **Trading Bot Executor:** Manages the execution of trading strategies on Binance. It communicates with the Binance API, executing trades based on user-defined strategies.
5. **Database Management:** Stores user information, trading strategies, historical data, and transaction logs. Ensures data integrity and efficient retrieval.
6. **User Interface (UI) Layer:** Provides an interactive interface for users to register, create strategies, and manage their accounts.
7. **API Integration Layer:** Facilitates communication between the software and external APIs like Binance for trading operations and data retrieval.
8. **Security Layer:** Ensures secure communication, data encryption, and secure storage of sensitive information like API keys.
9. **Logging and Monitoring:** Tracks system performance, user activities, and potential errors for maintenance and optimization.

3.2.2. Class Diagram

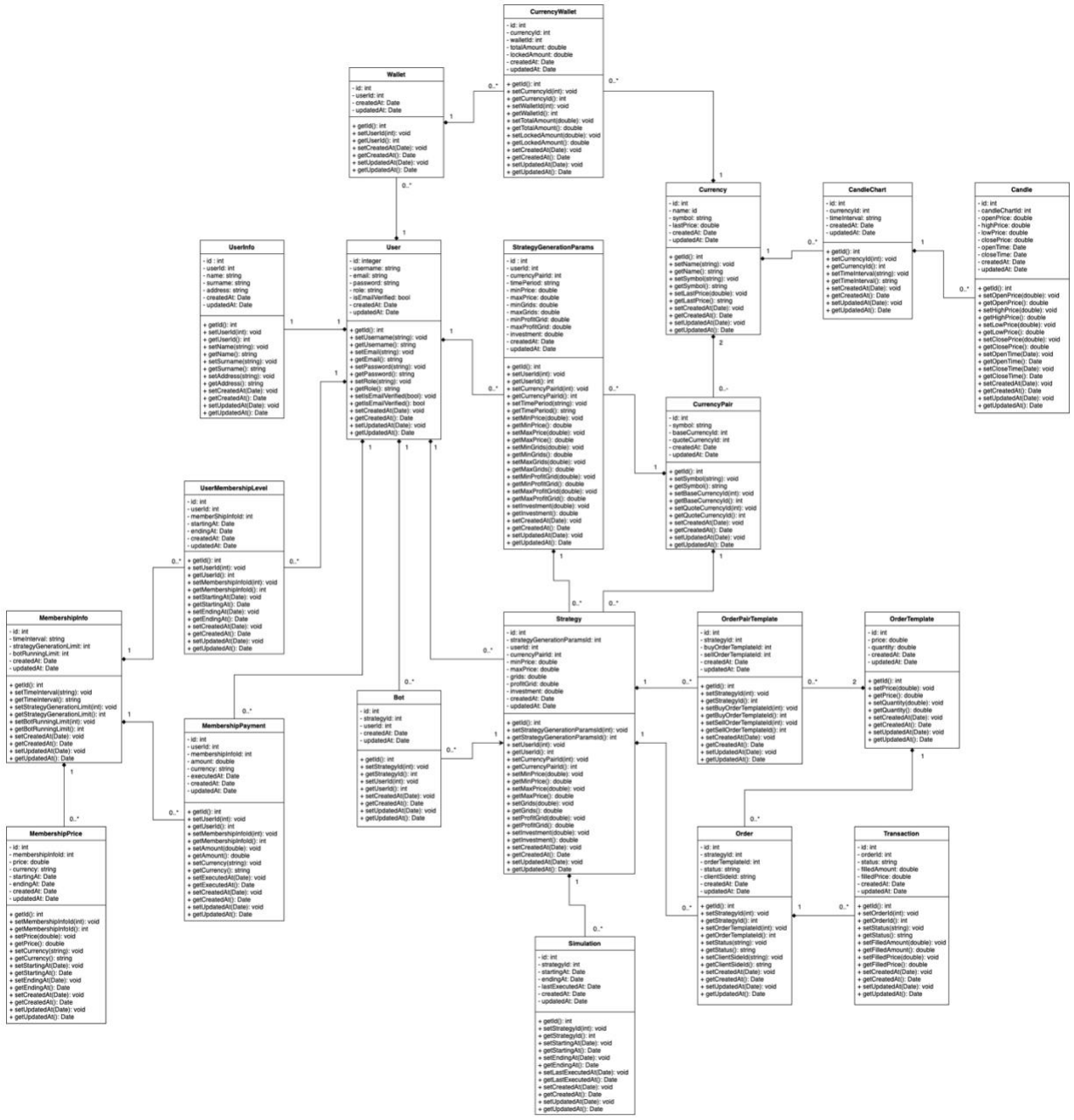

Figure 6 Class Diagram

3.3. System Modeling

3.3.1. Activity Diagrams

3.3.1.1. Register Activity Diagram

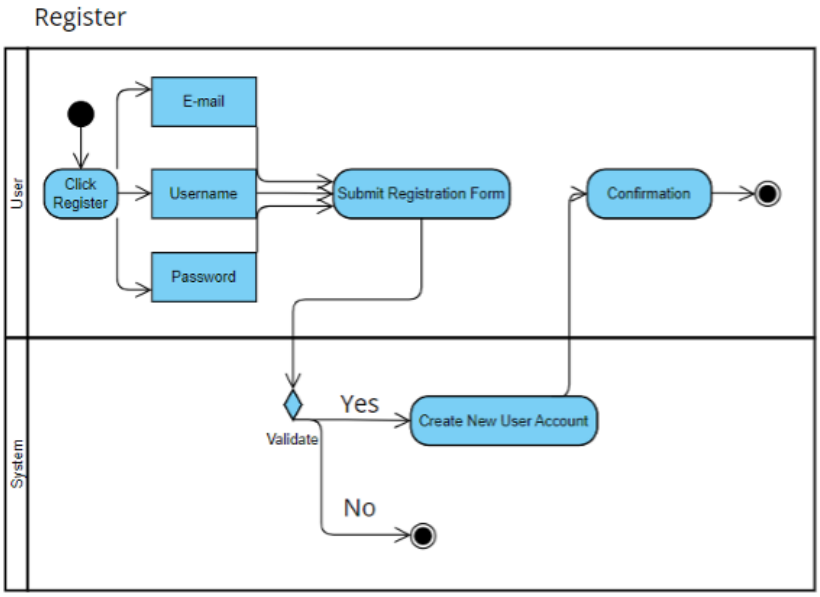

Figure 7 Register Activity Diagram

3.3.1.2. Login the System Activity Diagram

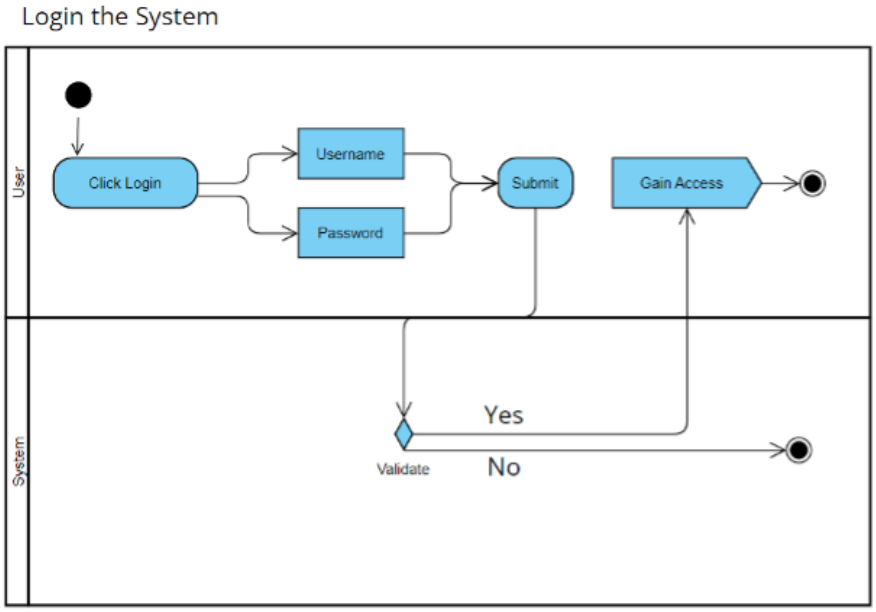

Figure 8 Login the System Activity Diagram

3.3.1.3. Account Settings Activity Diagram

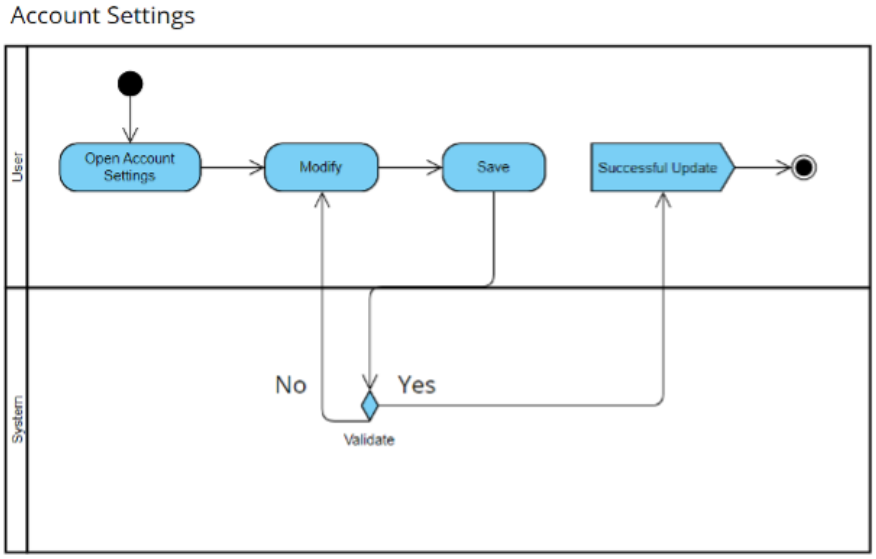

Figure 9 Account Settings Activity Diagram

3.3.1.4. Changing Membership Level Activity Diagram

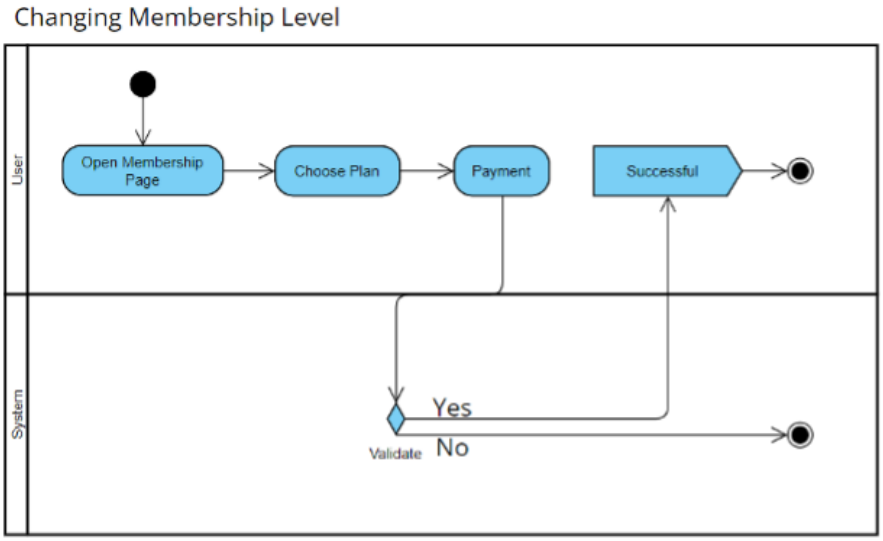

Figure 10 Changing Membership Level Activity Diagram

3.3.1.5. Generating Grid Bot Strategy Activity Diagram

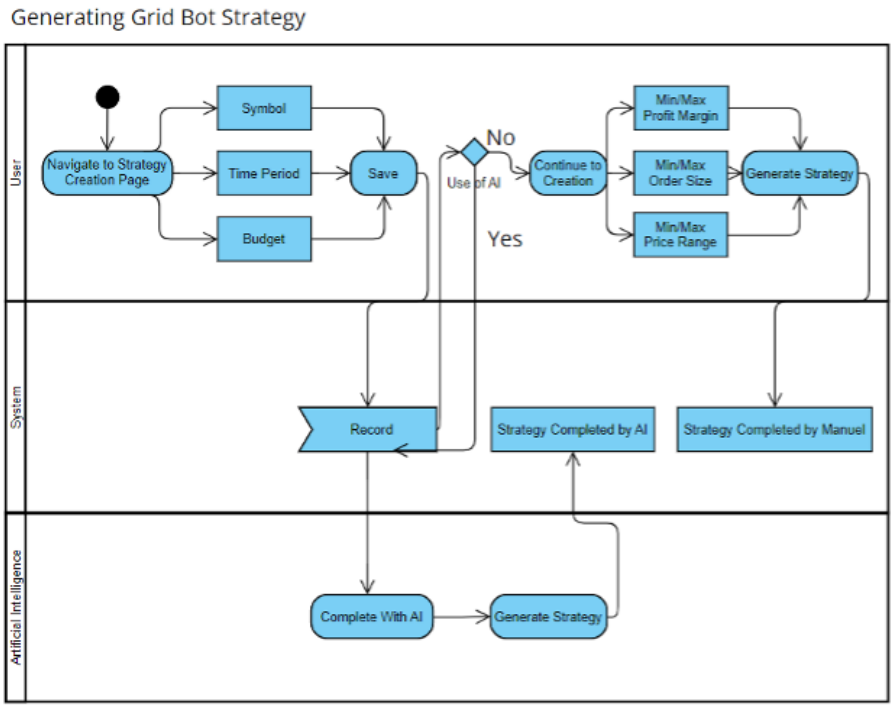

Figure 11 Generating Grid Bot Strategy Activity Diagram

3.3.1.6. Importing Ready-Made / Exporting Generated Strategy Activity Diagram

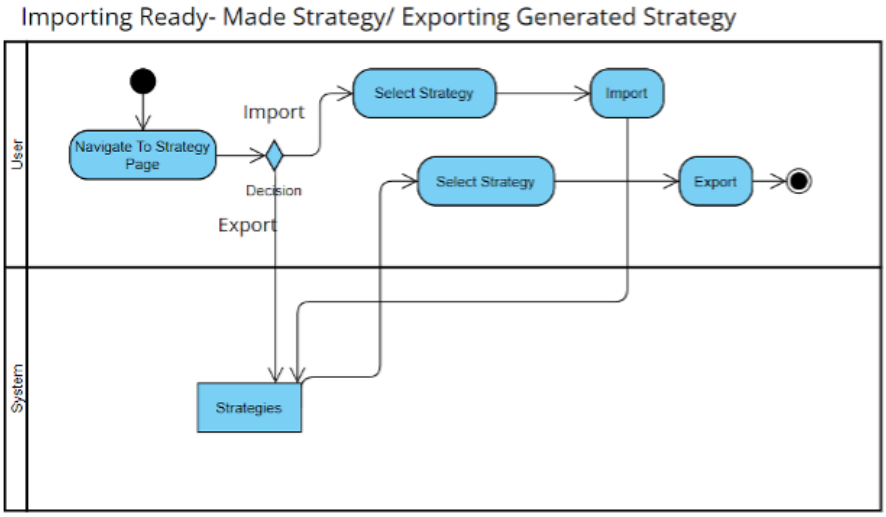

Figure 12 Importing Ready-Made / Exporting Generated Strategy Activity Diagram

3.3.1.7. Simulating Strategy Activity Diagram

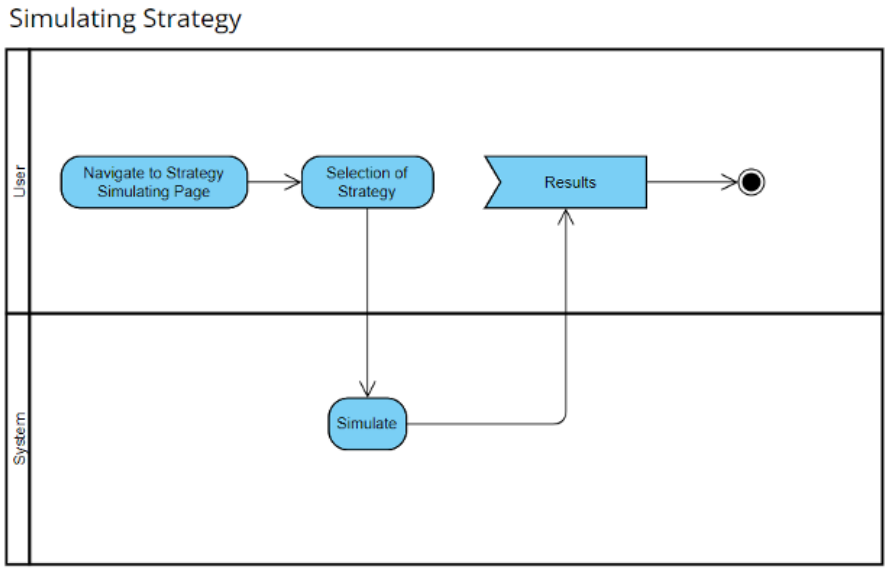

Figure 13 Simulating Strategy Activity Diagram

3.3.1.8. Running the Strategy as a Bot Activity Diagram

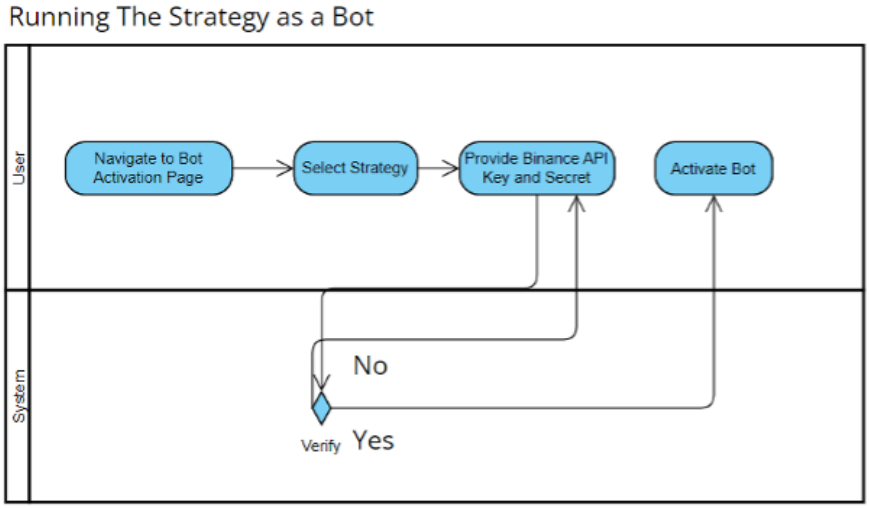

Figure 14 Running the Strategy as a Bot Activity Diagram

3.3.1.9. Pausing/Ending Running Bot Activity Diagram

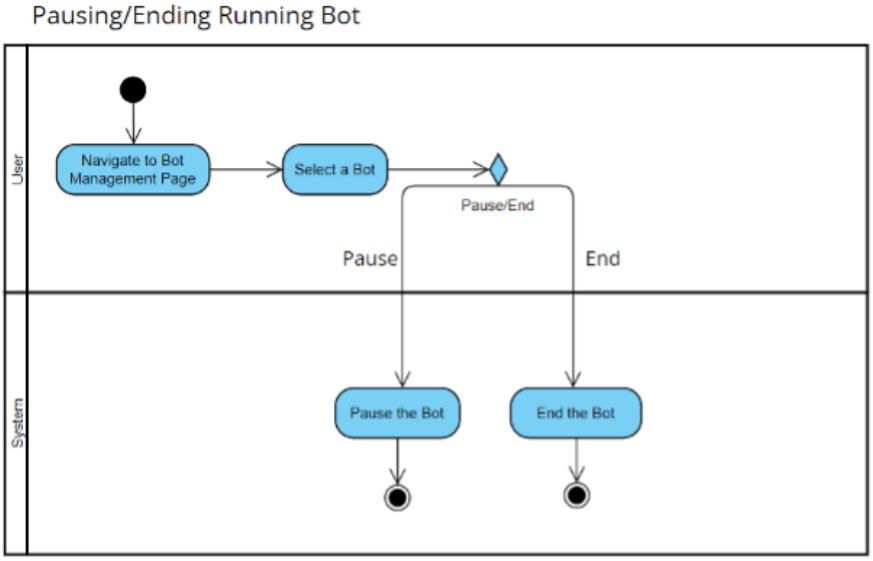

Figure 15 Pausing/Ending Running Bot Activity Diagram

3.3.1.10. System/User Settings Management Activity Diagram

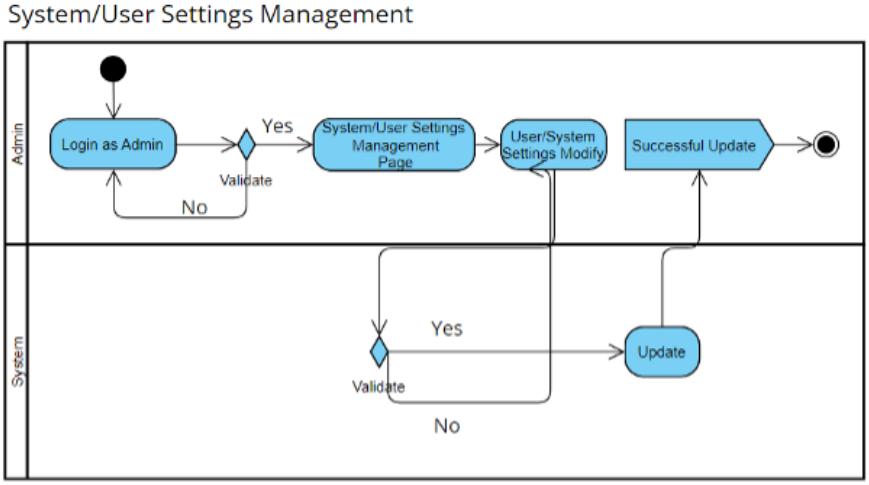

Figure 16 System/User Settings Management Activity Diagram

3.3.1.11. Extracting Candle Chart Data Activity Diagram

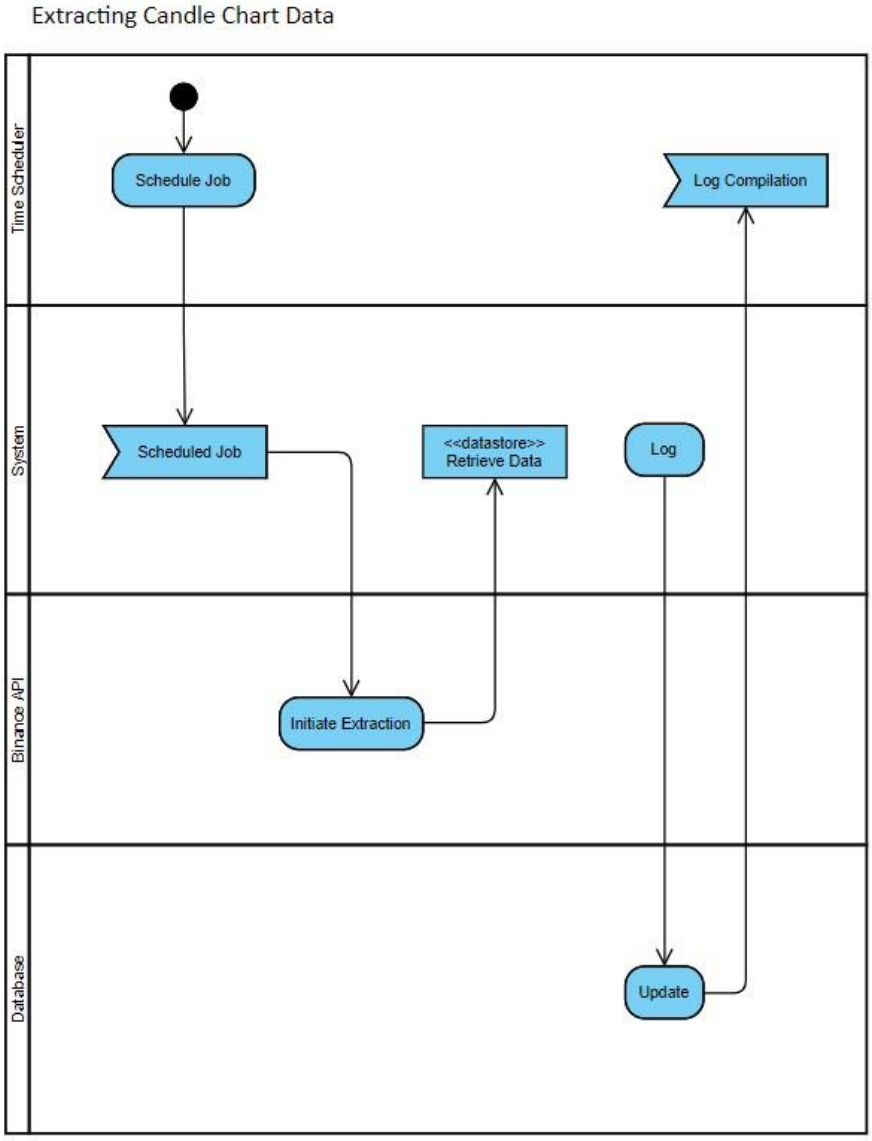

Figure 17 Extracting Candle Chart Data Activity Diagram

3.3.2. Sequence Diagrams

3.3.2.1. Register – User Sequence Diagram

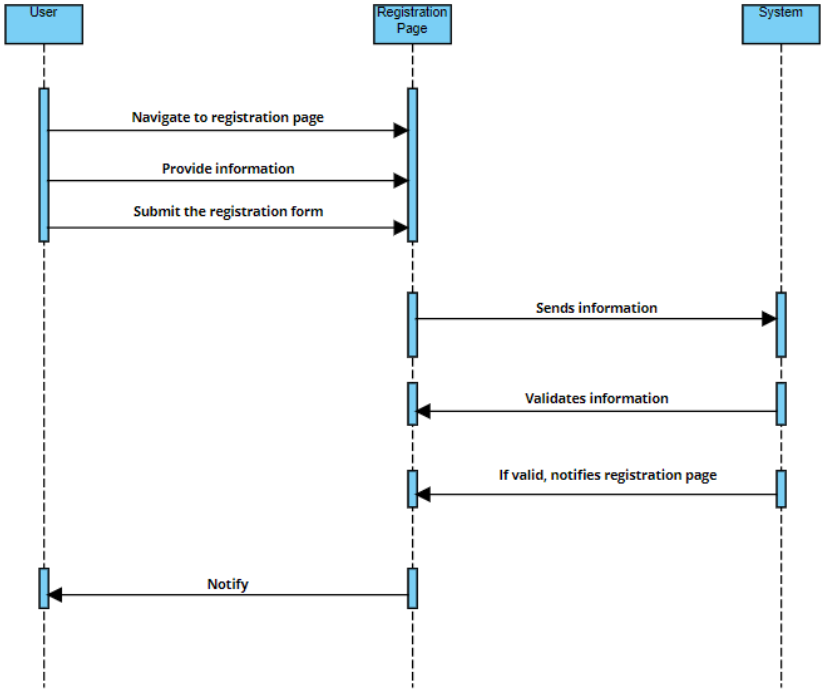

Figure 18 Register – User Sequence Diagram

3.3.2.2. Login the System- User Sequence Diagram

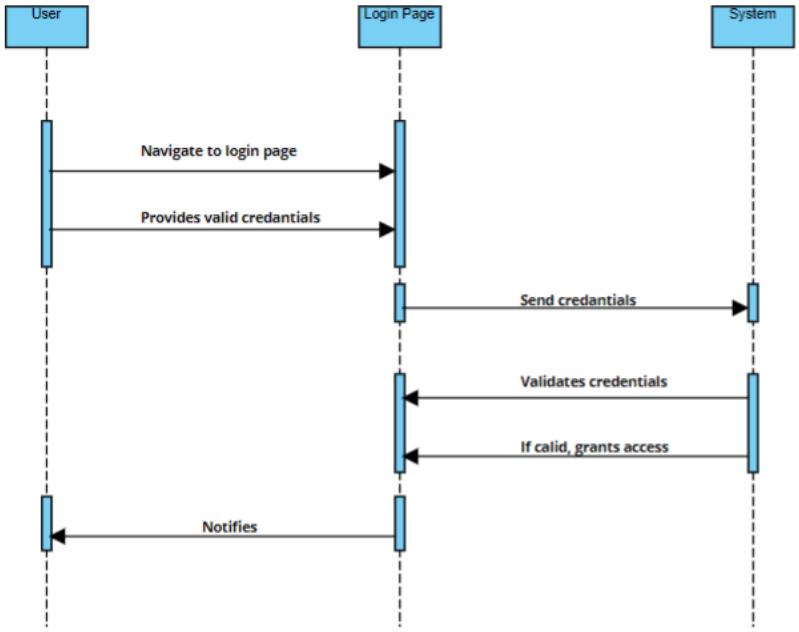

Figure 19 Login the System - User Sequence Diagram

3.3.2.3. Account Settings- User Sequence Diagram

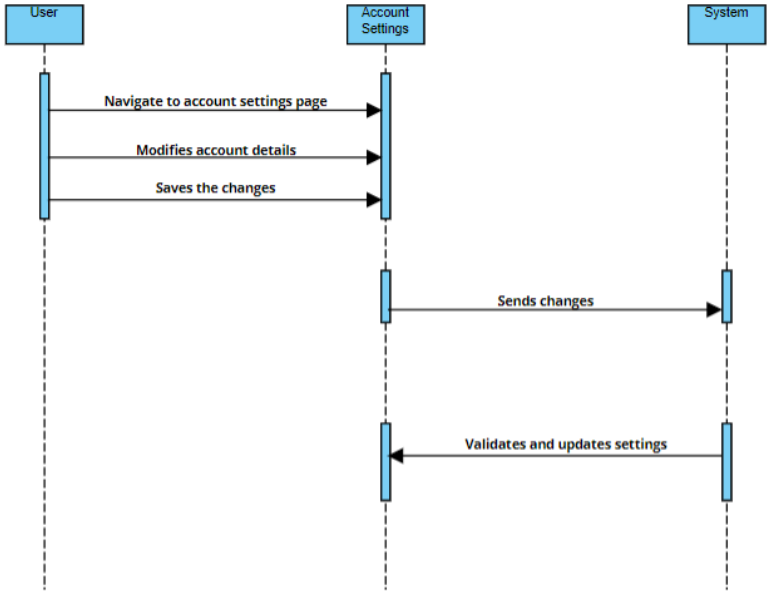

Figure 20 Account Settings - User Sequence Diagram

3.3.2.4. Changing Membership Level- User Sequence Diagram

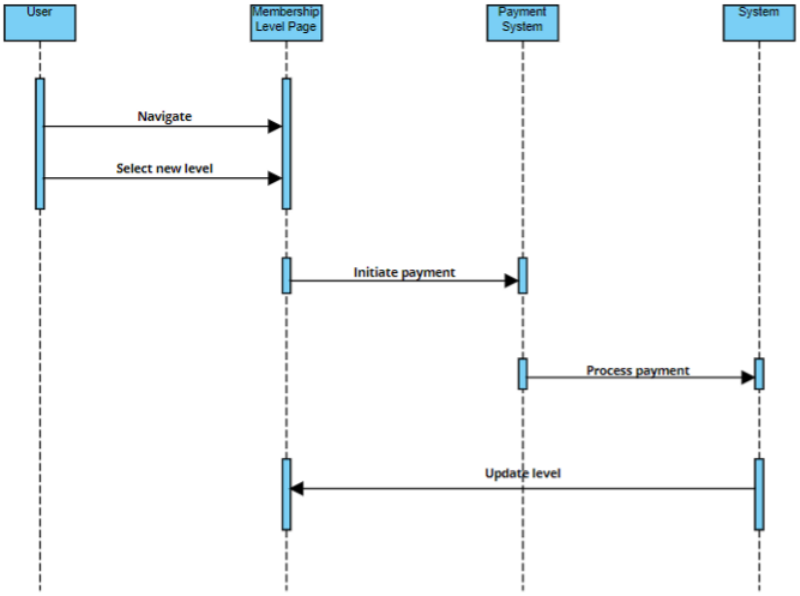

Figure 21 Changing Membership Level - User Sequence Diagram

3.3.2.5. Generating Grid Bot Strategy- User Sequence Diagram

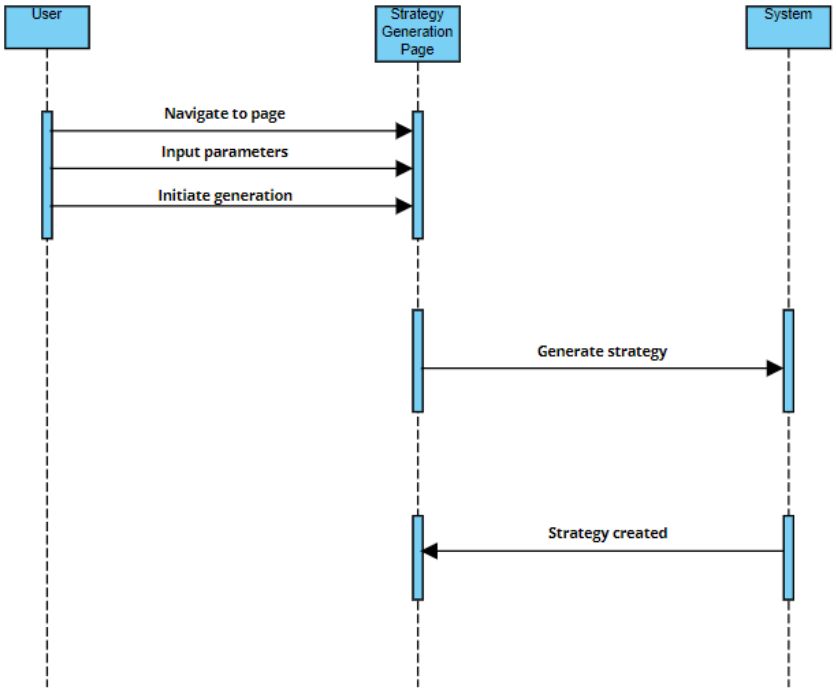

Figure 22 Generating Grid Bot Strategy - User Sequence Diagram

3.3.2.6. Importing Ready-Made Strategies- User Sequence Diagram

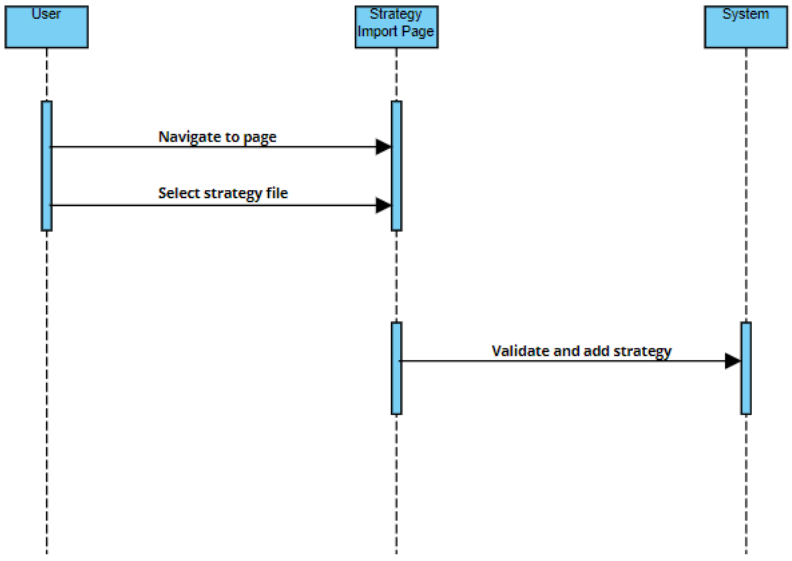

Figure 23 Importing Ready-Made Strategies - User Sequence Diagram

3.3.2.7. Exporting Generated Strategies- User Sequence Diagram

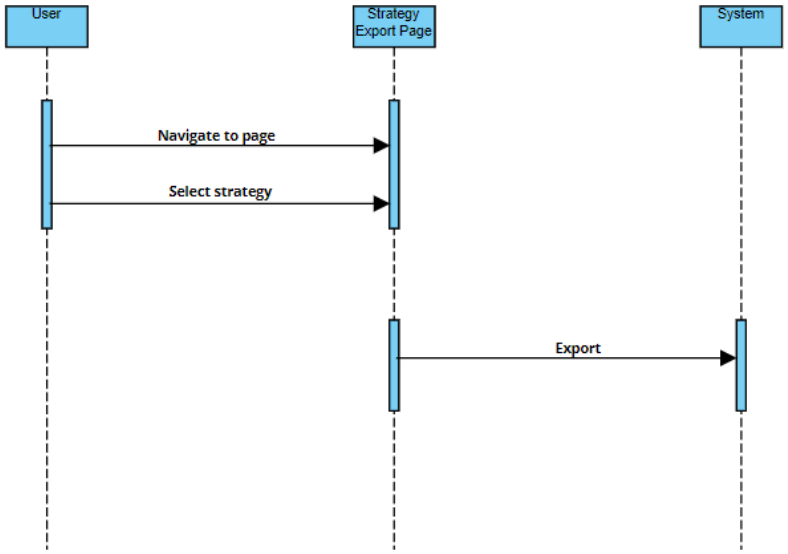

Figure 24 Exporting Generated Strategies - User Sequence Diagram

3.3.2.8. Simulating Strategy- User Sequence Diagram

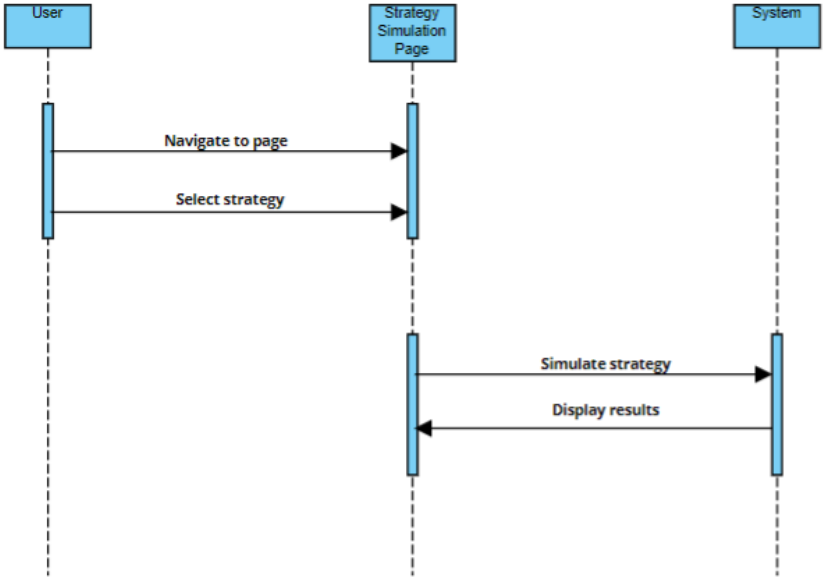

Figure 25 Simulating Strategy - User Sequence Diagram

3.3.2.9. Running the Strategy as a Bot- User Sequence Diagram

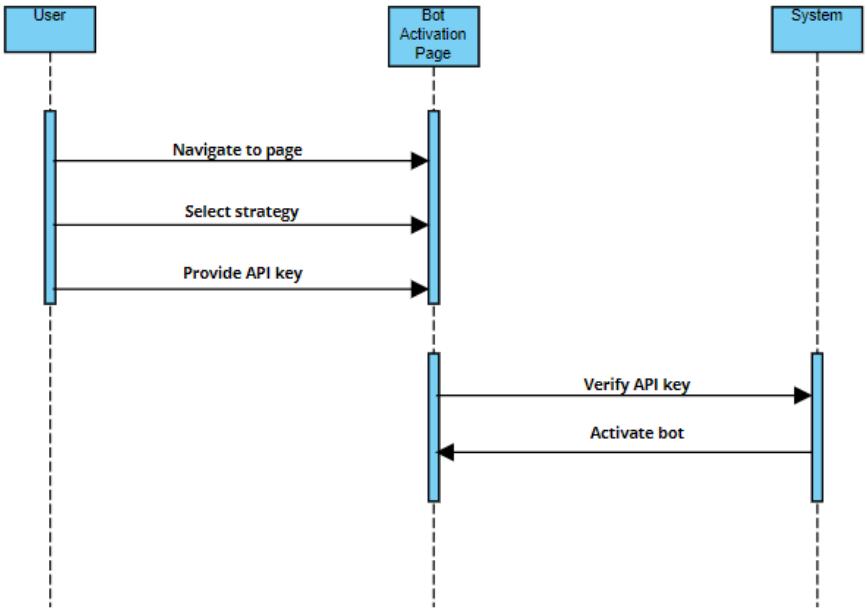

Figure 26 Running the Strategy as a Bot - User Sequence Diagram

3.3.2.10. Pausing/Ending Running Bots- User Sequence Diagram

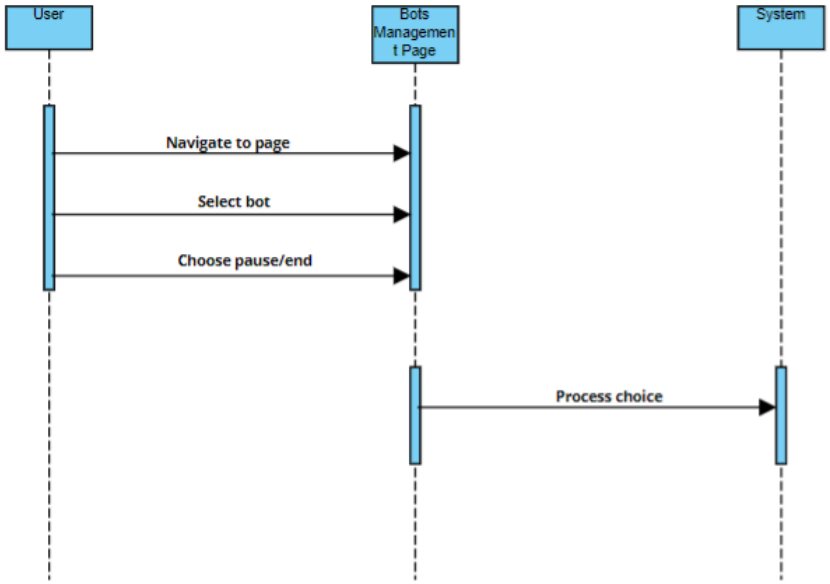

Figure 27 Pausing/Ending Running Bots - User Sequence Diagram

3.3.2.11. System Management- System Administrator Sequence Diagram

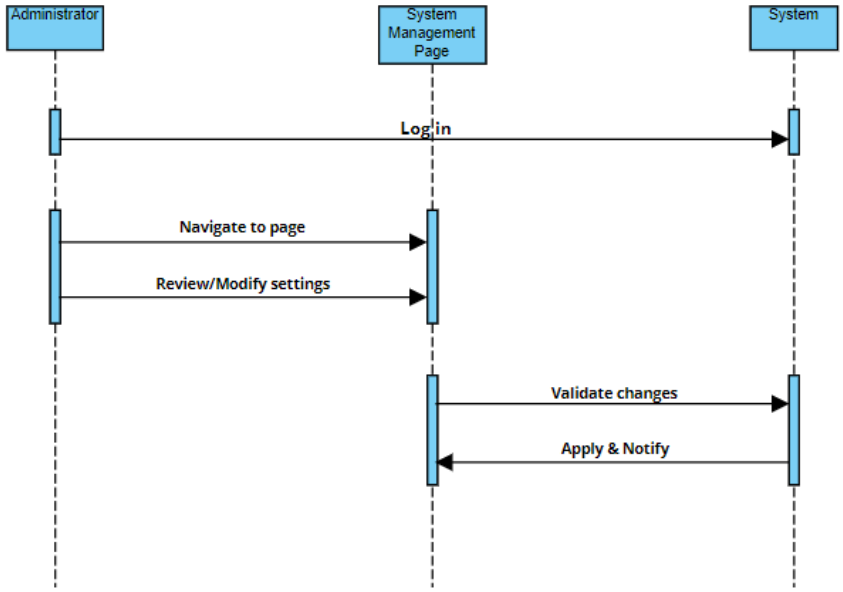

Figure 28 System Management - System Administrator Sequence Diagram

3.3.2.12. User Settings Management- System Administrator Sequence Diagram

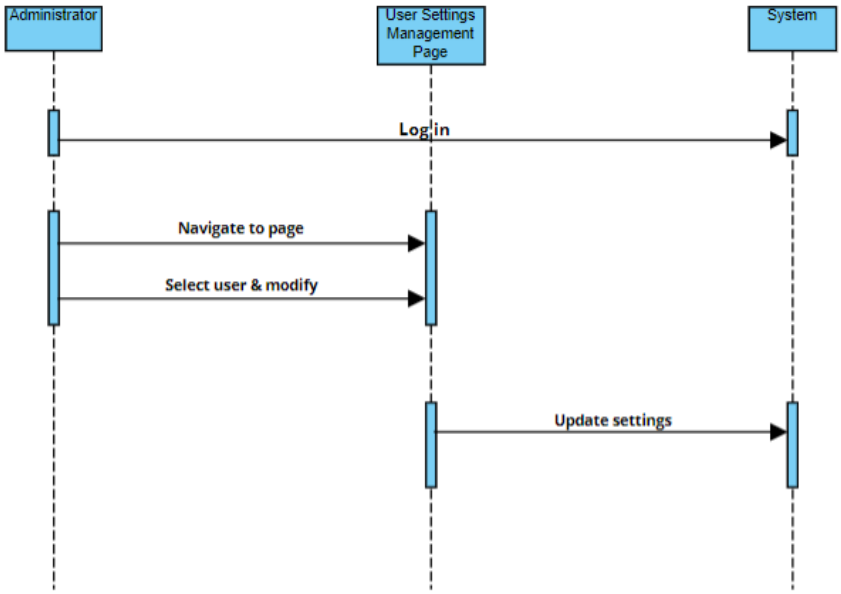

Figure 29 User Settings Management - System Administrator Sequence Diagram

3.3.2.13. Extracting Candle Chart Data- Time Sequence Diagram

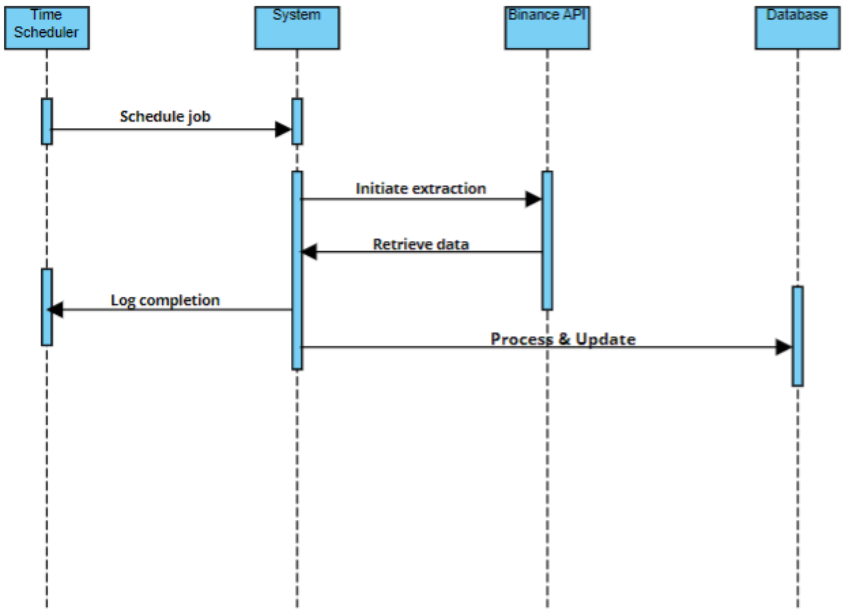

Figure 30 Extracting Candle Chart Data - Time Sequence Diagram

4. User Interface Design

4.1 Register Page

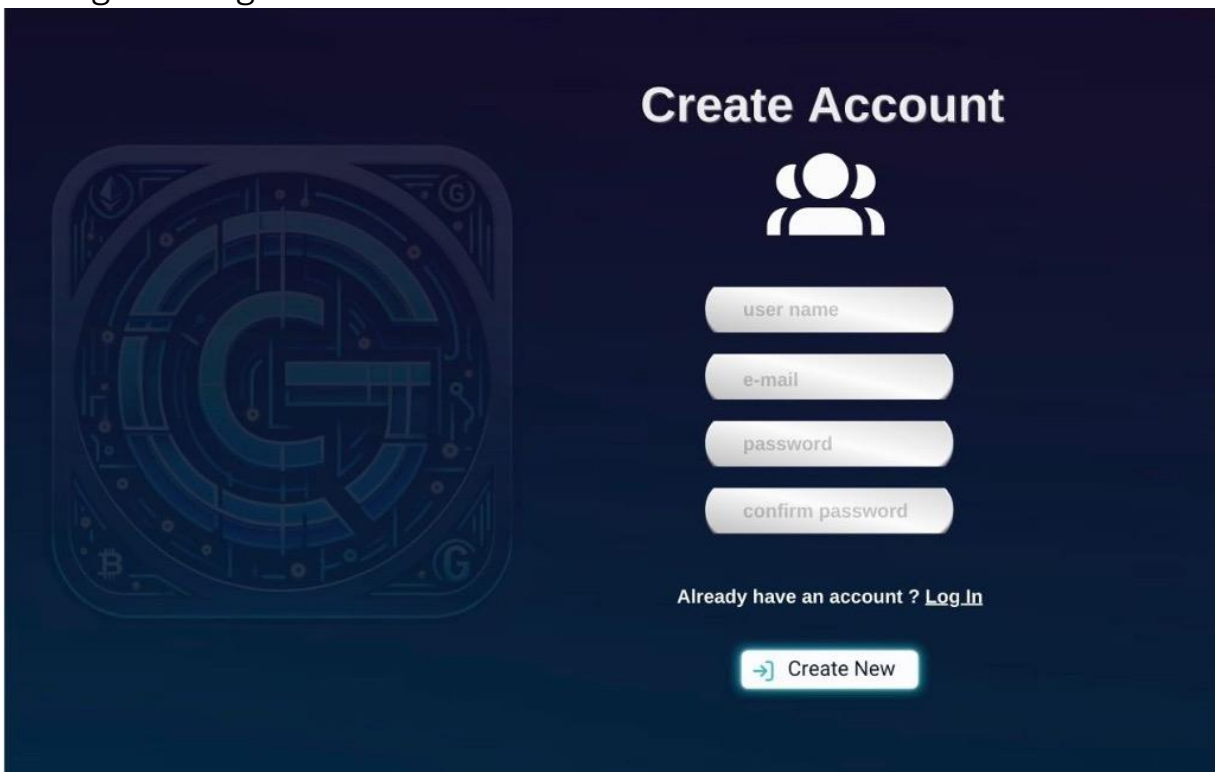

Figure 31 Register Page

4.2 Login Page

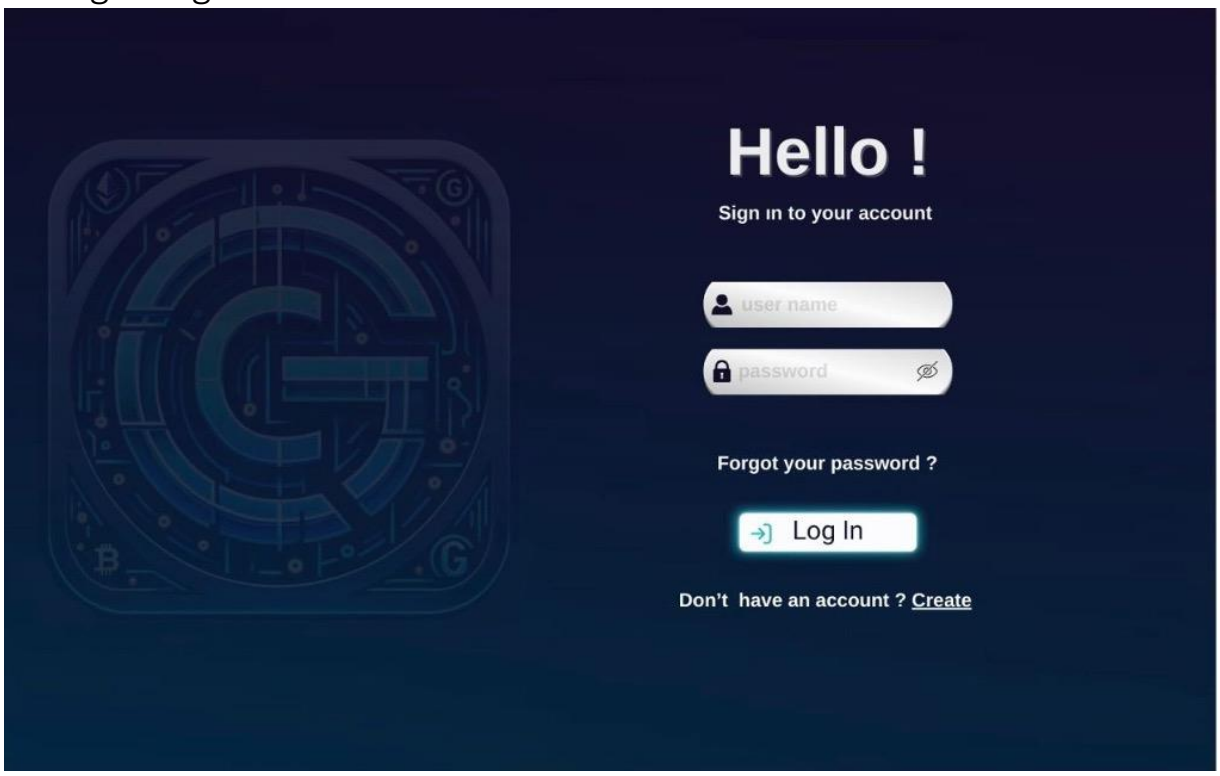

Figure 32 Login Page

4.3 Changing Membership Level Page

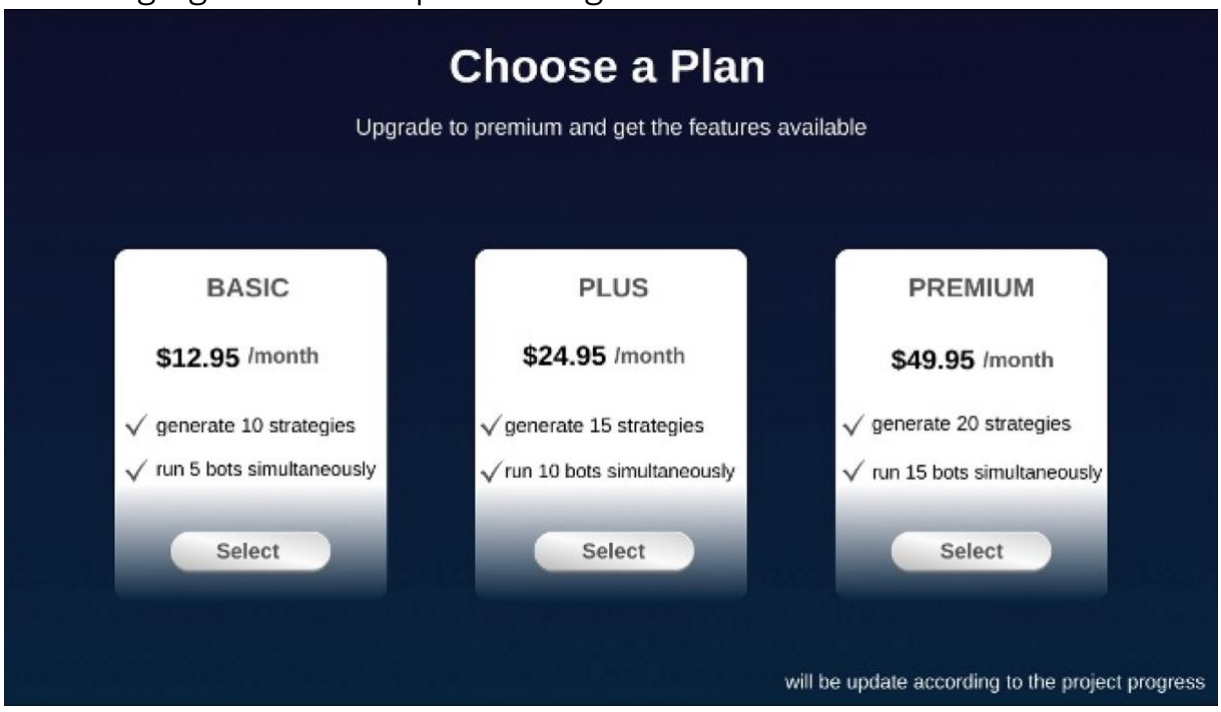

Figure 33 Changing Membership Level Page

4.4 Generating Grid Bot Strategy Page

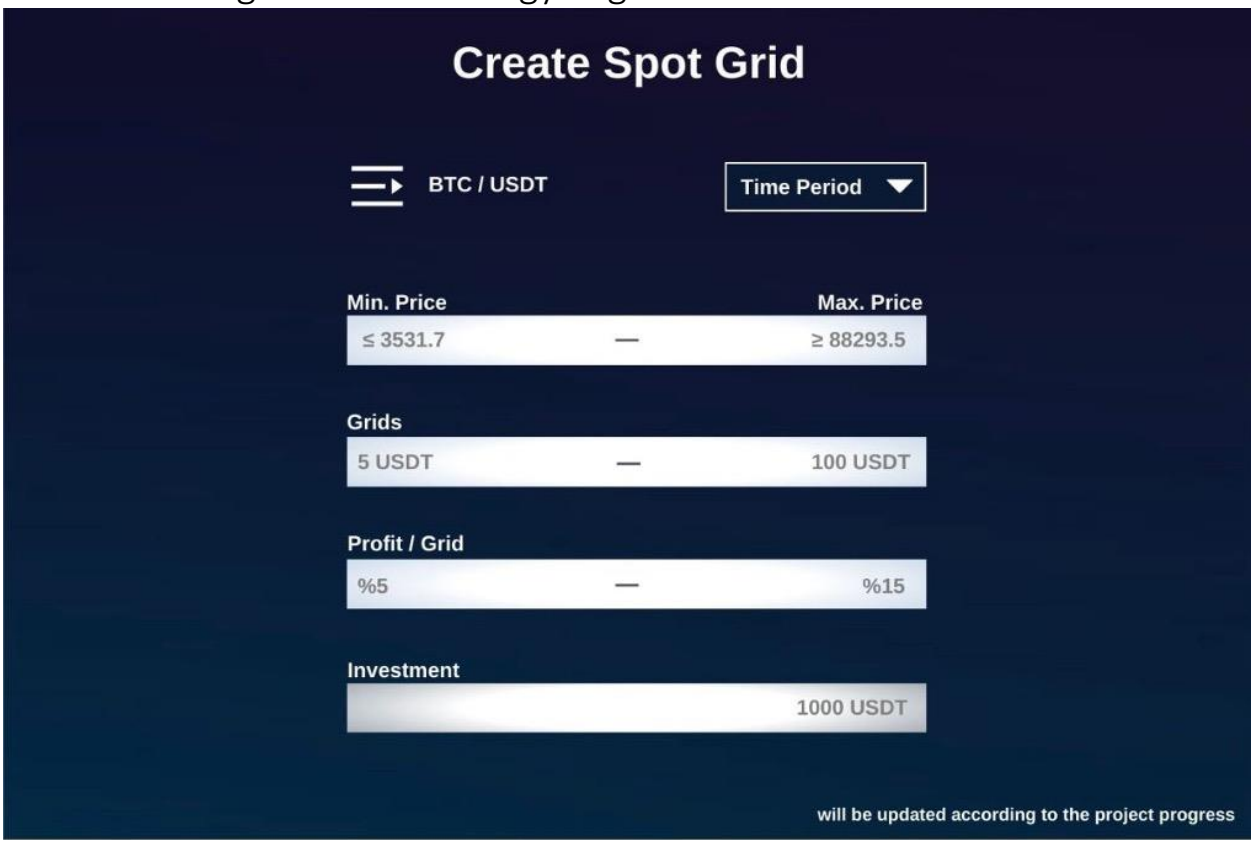

Figure 34 Generating Grid Bot Strategy Page

Conclusion

In conclusion, this comprehensive project report details the development of "Gridy: AI-based Grid Trading Strategy Builder," a sophisticated tool designed for the cryptocurrency market. The report successfully combines an in-depth exploration of financial markets, particularly focusing on the intricacies of cryptocurrency and futures markets, with the practical application of AI in trading strategies. The team has meticulously covered various aspects, including historical market analysis, algorithmic trading, and risk management. This endeavor not only showcases the team's technical prowess and innovative thinking but also contributes significantly to the field of cryptocurrency trading. The project's success in integrating advanced AI techniques and algorithmic trading strategies into a user-friendly platform represents a significant achievement in this domain, promising to enhance trading efficiency and profitability for its users.

Project Plan

Figure 35 Project Plan

References

[1] C. Team, "Spot Market," [Online]. Available: https://corporatefinanceinstitute.com/resources/career-map/sell-side/capital-markets/spot-market/. [Accessed November 2023].

[2] J. C. Hull, "Options, Futures, and Other Derivatives," 10th ed., Pearson, 2017.

[3] R. W. Kolb, "Futures, Options, and Swaps," 5th ed., Wiley, 2015.

[4] R. W. Kolb,Futures, Options, and Swaps, 5th ed.Wiley, 2015.

[5] L. G. McMillan, "Options as a strategic investment," Penguin, 2002.

[6] H. Song ve S. Yam, "An Empirical Analysis of Cryptocurrency Perpetual Swaps," http://arno.uvt.nl/show.cgi?fid=161465

[7] H. Ayes, "Perpetual Futures: What They Are and How They Work," Medium, https://www.investopedia.com/what-are-perpetual-futures-7494870

[8] Wikipedia, "Order (exchange)," [Online]. Available: https://en.wikipedia.org/wiki/Order_(exchange). [Accessed November 2023].

[9] Wikipedia, "Order Book," [Online]. Available: https://en.wikipedia.org/wiki/Order_book. [Accessed November 2023].

[10] Wikipedia, "Cryptocurrency exchange," [Online]. Available: https://en.wikipedia.org/wiki/Cryptocurrency_exchange. [Accessed November 2023].

[11] X. Wu, "Dynamics of Centralized and Decentralized Cryptocurrency Exchanges," 1 August 2023. [Online]. Available: https://medium.com/sciecon-innovate/dynamics-of-centralized-and-decentralized-cryptocurrency-exchanges-83f777023609. [Accessed November 2023].

[12] P. Sandner, "Decentralized Finance — A Systematic Literature Review and Research Directions," 24 February 2022. [Online]. Available: https://philippsandner.medium.com/decentralized-finance-a-systematic-literature-review-and-research-directions-fb3ce59bebda. [Accessed November 2023].

[13] Samuelsson, "The History of Technical Analysis," 11 September 2023. [Online]. Available: https://therobusttrader.com/the-history-of-technical-analysis/. [Accessed October 2023].

[14] C. Mitchell, "Trend Trading: The 4 Most Common Indicators," 15 March 2022. [Online]. Available: https://www.investopedia.com/articles/active-trading/041814/four-most-commonlyused-indicators-trend-trading.asp. [Accessed October 2023].
[15] Elearnmarkets, "Top 5 Momentum Indicators that Analyses Trend Strength," 7 September 2023. [Online]. Available: https://blog.elearnmarkets.com/top-5-momentum-indicators/. [Accessed October 2023].
[16] Elearnmarkets, "5 Important Volatility Indicators that Traders should know," 19 September 2023. [Online]. Available: https://blog.elearnmarkets.com/know-5-important-volatility-indicators/. [Accessed October 2023].
[17] Elearnmarkets, "9 Types of Volume Indicators a trader should know," 7 September 2023. [Online]. Available: https://blog.elearnmarkets.com/volume-indicator/#4-money-flow-index. [Accessed October 2023].
[18] tradeciety.com, "5 BEST TRADING OSCILLATOR INDICATORS TO FIND MARKET ENTRIES," 15 March 2021. [Online]. Available: https://tradeciety.com/5-best-trading-oscillator-indicators-to-find-market-entries. [Accessed October 2023].
[19] "Relative strength index," [Online]. Available: https://en.wikipedia.org/wiki/Relative_strength_index. [Accessed October 2023].
[20] "MACD," [Online]. Available: https://en.wikipedia.org/wiki/MACD. [Accessed October 2023].
[21] "Moving average," [Online]. Available: https://en.wikipedia.org/wiki/Moving_average. [Accessed October 2023].
[22] "Bollinger Bands," [Online]. Available: https://en.wikipedia.org/wiki/Bollinger_Bands. [Accessed October 2023].
[23] "Average true range," [Online]. Available: https://en.wikipedia.org/wiki/Average_true_range. [Accessed October 2023].
[24] "Stochastic oscillator," [Online]. Available: https://en.wikipedia.org/wiki/Stochastic_oscillator. [Accessed October 2023].
[25] DevTeam. "Build a Crypto Trading Bot." DevTeam.Space. Available: https://www.devteam.space/blog/build-a-crypto-trading-bot/

[26] ActiveState. "How to Build an Algorithmic Trading Bot." ActiveState Blog. Available: https://www.activestate.com/blog/how-to-build-an-algorithmic-trading-bot
[27] Financial Modelling & Algorithmic Trading. "How to Build a Trading Bot." YouTube. Available: https://www.youtube.com/watch?v=WcfKaZL4vpA
[28] Yellow Systems. "How to Build a Trading Bot." Yellow Systems Blog. Available: https://yellow.systems/blog/how-to-build-a-trading-bot
[29] Procoders. "How to Create a Crypto Trading Bot." Procoders Blog. Available: https://procoders.tech/blog/how-to-create-a-crypto-trading-bot
[30] Shankar, Rakesh. "How to Code Your Own Algo Trading Robot." Investopedia. Available: https://www.investopedia.com/articles/active-trading/081315/how-code-your-own-algo-trading-robot.asp