*Crypto Trading Platform*

*Baba Badrinath Bellamkonda*

[CRYPTO TRADING 4](#_Toc2141561051)

[ABSTRACT 4](#_Toc89094926)

[INTRODUCTION: 5](#_Toc1517335271)

[OBJECTIVES 6](#_Toc1408341661)

[DATA: 7](#_Toc1574839874)

[1. Structured Data 7](#_Toc2110325491)

[1.1 Data Source 7](#_Toc523249255)

[1.2 Data Attributes 8](#_Toc761669691)

[2. Unstructured Data 8](#_Toc279181203)

[2.1 Data Source 8](#_Toc149862273)

[2.2 News Data 9](#_Toc425107167)

[2.2 Data Attributes 9](#_Toc511764562)

[PROJECT FEATURES 9](#_Toc1812403665)

[1. Cryptocurrency Market Data 10](#_Toc1865351364)

[2. News Data 11](#_Toc1994316631)

[3. Sentiment Data 11](#_Toc2003912652)

[4. User and Transaction Data 12](#_Toc490515181)

[5. RSI(Relative Strength Index) datapoints: 13](#_Toc1143578480)

[MODULES DESCRIPTION 14](#_Toc1052543015)

[1. Administrator Module 14](#_Toc1019643126)

[2. User Module 14](#_Toc17856823)

[3. News Scraping Module 15](#_Toc1107577619)

[4. Sentiment Analysis Module 15](#_Toc535483585)

[5. Wallet and Trading Module 16](#_Toc2114062632)

[6. Peer-to-Peer (P2P) Trading Module 16](#_Toc919647456)

[7. Relative Strength Index (RSI) chart: 17](#_Toc1970147945)

[CALCULATIONS AND METRICS 18](#_Toc1419107441)

[1. Percentage change: 19](#_Toc1252430362)

[2.Difference: 19](#_Toc1852027616)

[3.Average gain: 20](#_Toc964922138)

[4. Average loss 20](#_Toc373167573)

[5. Exponential weighted moving average 20](#_Toc1777848662)

[Relative Strength (RS): 21](#_Toc443064393)

[RSI: 21](#_Toc1192904101)

[Buy Cost Calculation 21](#_Toc1348362926)

[Sell Cost Calculation 22](#_Toc1475224191)

[Buy Transaction – Cost per Coin 22](#_Toc847534746)

[Sell Transaction – Amount Earned 23](#_Toc1239584143)

[Send to Balance (Deduction) 23](#_Toc200777510)

[Calculate Old Value (USD worth of selected coin before swap) 24](#_Toc393277742)

[Calculate New Quantity (how many targets coins user will get) 24](#_Toc1984721463)

[Swap Validation (before calculation) 25](#_Toc1546702241)

[USE CASES 25](#_Toc1368634453)

[Use Case 1: Administrator Login and Management 25](#_Toc1040821210)

[Use Case 2: User Registration and Authentication 27](#_Toc547151538)

[Use Case 3: Wallet Management 29](#_Toc1501057027)

[Use Case 4: Cryptocurrency Buy/Sell Operation 31](#_Toc53517655)

[Use Case 5: Peer-to-Peer (P2P) Trading 32](#_Toc1697224941)

[Use Case 6: News Scraping and Data Storage 34](#_Toc309237061)

[Use Case 7: Sentiment Analysis and Score Computation 35](#_Toc833059144)

[Use Case 8: Dashboard Visualization and User Analytics 37](#_Toc384311211)

[ARCHITECTURE DIAGRAM 39](#_Toc1636242758)

[ER DIAGRAM 41](#_Toc40842590)

[FLOW DIAGRAM 41](#_Toc1096568428)

[UML DIAGRAM 42](#_Toc496219575)

[Admin 42](#_Toc958429826)

[User 43](#_Toc809960285)

[Sequence Diagram: 44](#_Toc898548227)

[CONCLUSION 45](#_Toc201391149)

[FUTURE SCOPE 45](#_Toc1817019158)

[USER INTERFACE 46](#_Toc1794288556)

[Index: 46](#_Toc1761049017)

[Admin Login 46](#_Toc758052574)

[User Management 47](#_Toc353697399)

[User Login 47](#_Toc1291624664)

[User Registration 47](#_Toc1644548398)

[User home page 48](#_Toc1639817816)

[Crypto Prices 48](#_Toc1789927429)

[Buy Crypto 49](#_Toc27493187)

[Sell Crypto 50](#_Toc724126063)

[User Wallet 51](#_Toc2069657102)

[Add Amount 52](#_Toc1981042152)

[Send to Bank 52](#_Toc954644020)

[View Transactions 52](#_Toc797311609)

[Swap 53](#_Toc929588572)

[User Profile 53](#_Toc1459057037)

[Sell on P2P: 54](#_Toc347531865)

[P2P Orders 54](#_Toc66204894)

[Crypto News 55](#_Toc899510647)

[RSI Chart 55](#_Toc1338760530)

[REFERENCES 56](#_Toc1472100974)

# **CRYPTO TRADING TRADING**

## **ABSTRACT**

The Crypto Trading and Sentiment Analysis System is an integrated platform combining automated data collection, sentiment analysis, and simulated crypto trading. It gathers real-time market data using yFinance and extracts news from TradingView via Selenium for sentiment evaluation using a pre-trained crypto model.Technical analysis is performed using the Relative Strength Index (RSI), calculated from 30 days of OHLCV data to identify overbought and oversold conditions through an RSI scatter plot.A Flask-based web app enables users to analyze trends, manage a virtual wallet, and perform simulated trading, with all data securely stored in MySQL. This project showcases the integration of AI and financial analytics to forecast market trends and enhance trading decisions.

## **INTRODUCTION:**

The Crypto Trading and Sentiment Analysis System is a comprehensive platform that integrates automated data acquisition, sentiment evaluation, and a simulated trading environment for cryptocurrencies. The primary objective of this project is to develop a unified system that demonstrates how financial analytics and natural language processing (NLP) can be applied to understand and forecast trends in the digital asset market.

The system periodically collects real-time market data, including prices, volumes, and trading patterns of various cryptocurrencies using Yahoo Finance (yFinance). In parallel, it extracts crypto-related news articles from TradingView through an automated Selenium-based web scraping module. The collected textual data is analyzed using a pre-trained crypto sentiment model, which classifies each news headline as bearish, neutral, or bullish. This sentiment information provides valuable insights into market psychology and potential price movements.

In addition to sentiment analysis, the project incorporates technical indicators to assess market momentum and trend strength. Specifically, the Relative Strength Index (RSI) is used as a momentum indicator to identify potential overbought and oversold conditions. Using 30 days of OHLCV (Open, High, Low, Close, Volume) data and a 14-day RSI period, an RSI scatter plot was generated to visualize each coin’s momentum and detect potential trend reversals or trading opportunities.

A web-based application, developed using Flask, offers users an interactive environment to analyze cryptocurrency trends, manage a virtual wallet, and perform simulated trading activities. Users can execute buy and sell operations, participate in peer-to-peer (P2P) exchanges, and view sentiment-driven analytical insights. All user activities, financial data, and sentiment results are securely stored in a MySQL database for efficient data management and retrieval.

## **OBJECTIVES**

The primary objective of this system is to design and implement an integrated solution that bridges cryptocurrency trading, market data analytics, and natural language processing. The specific aims of the project are outlined below:

1. To automate the collection of cryptocurrency market data, including daily open, high, low, close, and volume metrics using the Yahoo Finance API.
2. To gather current news and media content related to major cryptocurrencies through automated web scraping using Selenium.
3. To perform sentiment analysis on news headlines employing the CryptoBERT transformer model to classify market sentiment as bullish, neutral, or bearish.
4. To simulate cryptocurrency trading operations, enabling users to virtually buy, sell, or transfer assets through a user-friendly web interface.
5. To manage and monitor digital wallets, ensuring real-time balance updates during every transaction.
6. To provide administrative oversight, allowing the system administrator to verify users, track activity logs, and maintain overall data consistency.
7. To store and manage all data securely using MySQL, ensuring efficient retrieval and long-term reliability.
8. To plot the RSI scatter chart and visualize overbought and oversold conditions, enabling traders to quickly identify market momentum, potential trend reversals, and trading opportunities.

## **DATA:**

The dataset used in the Crypto Trading and Sentiment Analysis System integrates both structured and unstructured data sources. This combination enables a comprehensive understanding of cryptocurrency market dynamics by analyzing both numerical trading data and qualitative sentiment information extracted from news sources.

### **1. Structured Data**

The structured dataset refers to well-organized, numerical data collected from financial APIs, primarily using yFinance (Yahoo Finance API). This data is stored in tabular form with fixed fields, making it ideal for quantitative analysis and model-based forecasting.

#### 1.1 Data Source

**API Used:** yFinance

**Asset Type:** Cryptocurrency pairs

**Timeframe:** Last 30 days (daily intervals)

**Storage:** MySQL database (in normalized format

#### 1.2 Data Attributes

The structured data primarily consists of OHLCV (Open, High, Low, Close, Volume) values, which form the foundation for technical analysis and RSI calculation.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Date | Trading day timestamp |
| Open | Opening price of the cryptocurrency on that day |
| High | Highest price recorded during the day |
| Low | Lowest price recorded during the day |
| Close | Closing price of the cryptocurrency |
| Volume | Total trading volume during the day |

### 2. Unstructured Data

The **unstructured dataset** consists of qualitative text data extracted from financial news sources, primarily gathered through **web scraping**.

#### 2.1 Data Source

* **Website:** TradingView (News section)
* **Tool Used:** Selenium WebDriver (Python)
* **Data Type:** Raw text articles or short news sxummaries related to cryptocurrencies.

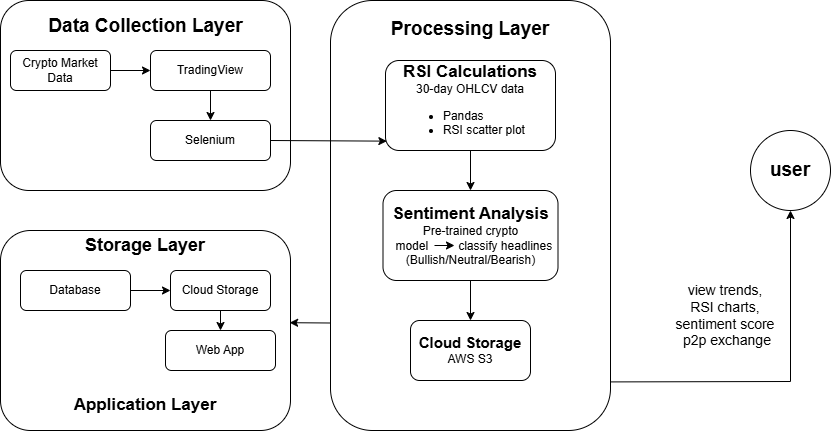
### 2.2 News Data

The script uses Selenium WebDriver (Chrome) to scrape cryptocurrency news from TradingView for popular coins like BTC, ETH, ADA, SOL, AVAX, XRP, BNB, USDT, DOGE, and DOT. It connects to a MySQL database. It loads the news page for each ticker, takes the title, source, and publication date, then scrolls dynamically to gather up to 20 distinct stories. Once the dates have been cleaned up and duplicates eliminated, they can be optionally added to MySQL. Lastly, by figuring out the earliest and latest article dates as well as their combined time span, it examines the news coverage of each currency.

#### 2.2 Data Attributes

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| **Headline** | Title of the news article |
| **Content** | Summary or body of the article |
| **Published Date** | Date and time of publication |
| **Source URL** | Link to the news article |
| **Ticker Reference** | Related cryptocurrency (e.g., BTC, ETH) |

# PROJECT FEATURES



## 1. Cryptocurrency Market Data

Cryptocurrency price and trading data form the foundation of the system’s analytical component. The data is extracted using the yFinance API, covering well-known digital assets such as Bitcoin "AVAXUSD", "SOLUSD", "ADAUSD", "BTCUSD", "XRPUSD", "ETHUSD", "BNBUSD", "USDTUSD", "DOGEUSD" and "DOTUSD".

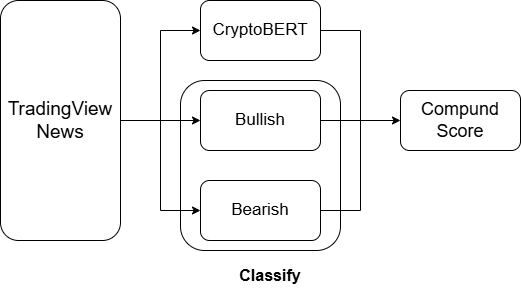
For each asset, the dataset contains key attributes including Date, Open, High, Low, Close, and Volume. An additional computed feature, Percentage Change, represents daily price fluctuations, helping users track performance over time.

The data is retrieved at a one-day interval and stored within the crypto\_data table, enabling efficient querying and analysis for both trading and sentiment correlation.

## 2. News Data

The news data component captures current cryptocurrency-related articles from online financial platforms, primarily TradingView. Using Selenium WebDriver, the system automatically navigates through web pages, extracts relevant article titles, timestamps, and publishers, and filters duplicates to ensure data accuracy.

The resulting dataset is saved as a structured CSV file (news\_data.csv) and imported into MySQL for downstream processing. This real-time collection of news provides the contextual foundation for sentiment evaluation, reflecting how external information and public discourse affect crypto market dynamics.



## 3. Sentiment Data

The sentiment dataset is generated from the textual analysis of news headlines using the ElKulako/CryptoBERT model, a transformer-based neural architecture fine-tuned for cryptocurrency-related contexts.

CryptoBERT is a pre-trained NLP model to analyse the language and sentiments of cryptocurrency-related social media posts and messages. It was built by further training the vinai's bertweet-base language model on the cryptocurrency domain, using a corpus of over 3.2M unique cryptocurrency-related social media posts. (A research paper with more details will follow soon.). The model was trained on the following labels: "Bearish" : 0, "Neutral": 1, "Bullish": 2

Each headline is tokenized, processed through the model, and assigned a sentiment probability across three categories: Bearish, Neutral, and Bullish. The system then computes a weighted sentiment score in the range of -1 to +1, representing the overall emotional tone toward each cryptocurrency.

These scores are aggregated and stored in the sentiment\_scores table, allowing users to interpret market moods and evaluate potential market movements based on sentiment strength.

## 4. User and Transaction Data

All user-related and transactional information is managed through a relational database structure implemented in MySQL. The users table stores authentication details and user-specific metadata.

The wallet table maintains individual account balances, while the transactions and p2p\_orders tables record detailed trading histories.

Each transaction entry includes information such as the cryptocurrency ticker, quantity, transaction value, and timestamp. Automatic updates ensure that every trading action, including P2P exchanges, reflects immediately on the user’s balance. This comprehensive storage model ensures data transparency and supports reliable audit trails for user activities.

## 5. RSI(Relative Strength Index) datapoints:

Technical indicators are mathematical calculations or pattern-based signals derived from historical market data such as price, volume, and open interest.They are used by traders and investors to analyze market trends, predict future price movements, and identify potential trading opportunities.

Among these, momentum indicators focus on measuring the speed and strength of price changes over a specific period.

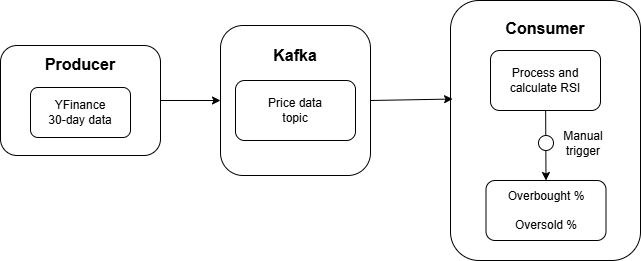
They help assess whether an asset is gaining or losing momentum, which can indicate if a current trend is likely to continue, slow down, or reverse.

Momentum indicators also help identify overbought and oversold market conditions.

The Relative Strength Index (RSI) is one of the most widely used momentum indicators.

In this project, RSI was chosen because it effectively highlights market momentum shifts and provides clear signals about potential trend reversals.

The RSI is calculated using OHLCV data streamed from Yahoo Finance (yFinance) through Kafka with the Finance API and Confluent Kafka Python client. Kafka and Zookeeper are running locally, and the data streaming process is manually triggered, as there is no automatic scheduling on the consumer side. Once the OHLCV data is received, the RSI values are computed using Pandas, and the results are used to identify overbought and oversold conditions for each cryptocurrency. All processed data, including OHLCV and RSI results, is stored in an AWS S3 bucket for easy access.



# **MODULES DESCRIPTION**

## **1. Administrator Module**

The Administrator Module provides privileged control over the entire system. Admins can log in securely, verify new user accounts, and manage system-wide data integrity.

Administrative operations include reviewing user profiles, approving or deactivating accounts, and monitoring transaction logs. Additionally, administrators have access to real-time sentiment and market datasets, enabling oversight of system performance and user engagement.

This module ensures that the platform remains structured, organized, and secure.

## **2. User Module**

The User Module serves as the core interactive layer of the system. Each registered user can access a personalized dashboard to monitor crypto prices, analyze sentiment data, and perform simulated trading.

Users can manage their wallets, add funds, and execute buy or sell operations. Every transaction is validated and logged in the database.

Through this interface, users can also explore peer-to-peer (P2P) trading, where they can directly negotiate and execute coin exchanges with other users. The module promotes a realistic trading experience while remaining entirely within a controlled, educational simulation environment.

## **3. News Scraping Module**

The News Scraping Module is designed to automatically gather cryptocurrency-related news using Selenium WebDriver. It opens the TradingView news feed for chosen tickers, scrolls through the page to reveal more articles, and extracts key attributes such as headlines, publication timestamps, and source information.

To improve accuracy, the system removes duplicate records and ignores non-relevant content. After processing, the curated dataset is saved as a CSV file, which can later be uploaded to cloud storage or integrated into the database. This functionality guarantees that the sentiment analysis engine always works with the latest and most meaningful market updates.

## **4. Sentiment Analysis Module**

The Sentiment Analysis Module is responsible for transforming unstructured textual news data into numerical sentiment metrics. Using CryptoBERT, each headline is processed to determine whether it reflects positive, neutral, or negative investor sentiment.

The system assigns weighted scores (-1, 0, +1) to represent these categories. The aggregated average sentiment for each cryptocurrency provides an indicator of market confidence.

This analytical approach helps correlate media tone with price movement, allowing users to infer possible trends based on public emotion.

## **5. Wallet and Trading Module**

This module manages user funds and virtual trading operations. Each user is assigned a wallet that maintains their available balance. Through the Flask interface, users can initiate buy or sell orders using live market prices retrieved from Yahoo Finance.

When a transaction is completed, the wallet balance is automatically updated, and the record is inserted into the transactions table.

The module ensures financial consistency and supports instant updates, mimicking real-world crypto exchange mechanisms.

## **6. Peer-to-Peer (P2P) Trading Module**

The P2P trading module allows users to engage in direct exchanges with one another. Sellers can list available coins along with their prices and accepted payment methods, while buyers can browse these listings and make purchases.

Upon completion of a transaction, both users’ wallets are updated, and transaction logs are stored for transparency. This feature closely resembles decentralized trading systems and adds an element of interactivity and flexibility to the platform.

## **7. Relative Strength Index (RSI) chart:**

The **Relative Strength Index (RSI)** is a momentum oscillator that measures the speed and change of price movements. It helps identify overbought or oversold market conditions.

For this project, 30 days of OHLCV (Open, High, Low, Close, Volume) data were used to calculate the RSI. The RSI was computed using a **14-day period**, which is the standard setting according to the RSI formula defined by J. Welles Wilder Jr.

The RSI value ranges from **0 to 100**, and it is interpreted as follows:

* **Above 70** → *Overbought* condition (potential price correction or reversal)
* **Below 30** → *Oversold* condition (potential upward reversal)

In this project, an RSI scatter plot was created to visualize the momentum of different coins over a 30-day period. Each data point in the chart represents a coin’s RSI value calculated using 14-day OHLCV data. These threshold lines(Overbought line at RSI = 70 and Oversold line at RSI = 30) help identify when a coin might be overvalued (RSI > 70) or undervalued (RSI < 30).

By plotting all coins together, the scatter plot provides a clear comparative view of which assets are in potential overbought or oversold zones.

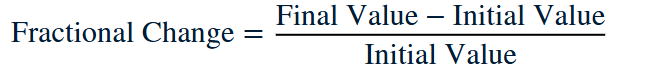
**TOOLS AND TECHNOLOGIES USED**

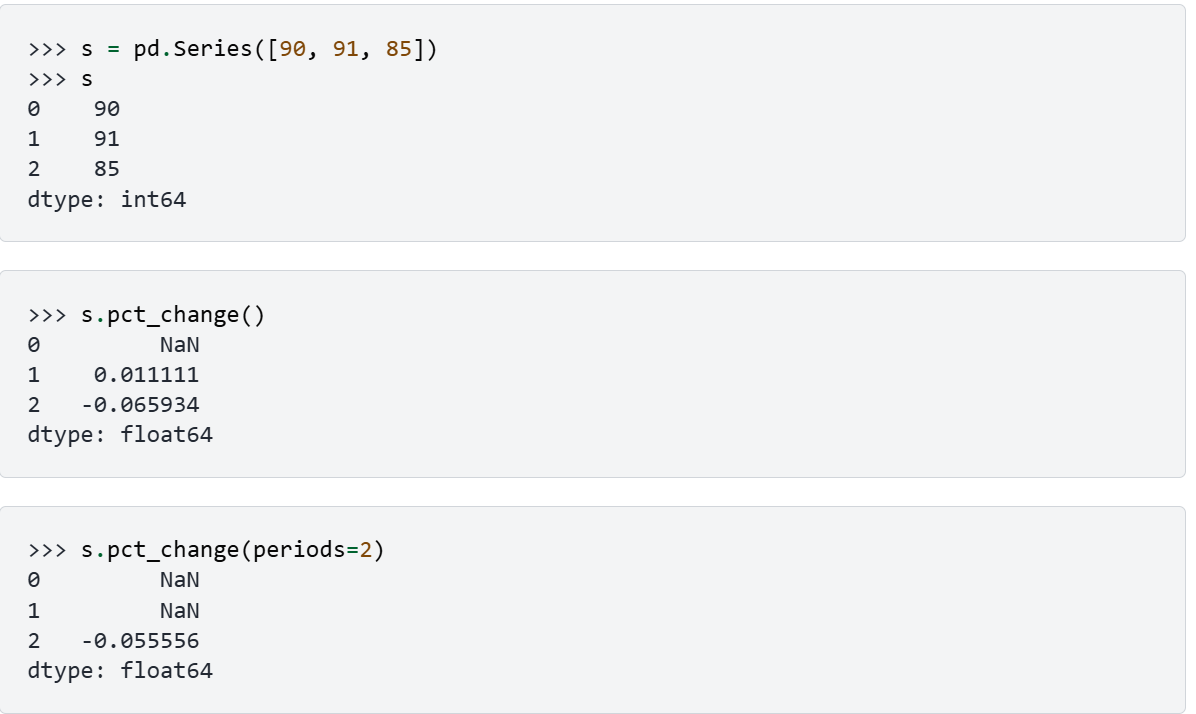
1. Backend & Web Framework
   1. Flask (Python): Handles routing, sessions, backend logic, and serves the web interface.
2. Database & Data Storage
   1. MySQL: Primary relational database for structured data storage and retrieval.
   2. AWS S3 (boto3): Optional cloud storage for backup and large dataset storage.
3. Data Collection & APIs
   1. Selenium: Automates web browsing and scrapes news or market sentiment from TradingView.
   2. yFinance API: Provides cryptocurrency and financial market data.
4. Data Processing & Analysis
   1. Pandas: For data manipulation, preprocessing, and aggregation.
   2. Matplotlib: For visualizing market trends, sentiment analysis results, and other analytics.
   3. Transformers Library (Crypto): Performs sentiment analysis using pre-trained deep learning models.
5. Messaging & Streaming
   1. Apache Kafka: Handles high-throughput, real-time messaging between different microservices or components (e.g., market data streaming, sentiment pipeline).
   2. Confluent Kafka Python (confluent\_kafka): Python client for producing and consuming messages efficiently.
   3. ZooKeeper: Manages Kafka cluster coordination, leader election, and metadata storage.
6. Frontend & Templating
   1. HTML, CSS, Jinja2: Frontend templates for dynamic rendering of user dashboards and analytics

# **CALCULATIONS AND METRICS**

## **1. Percentage change:**

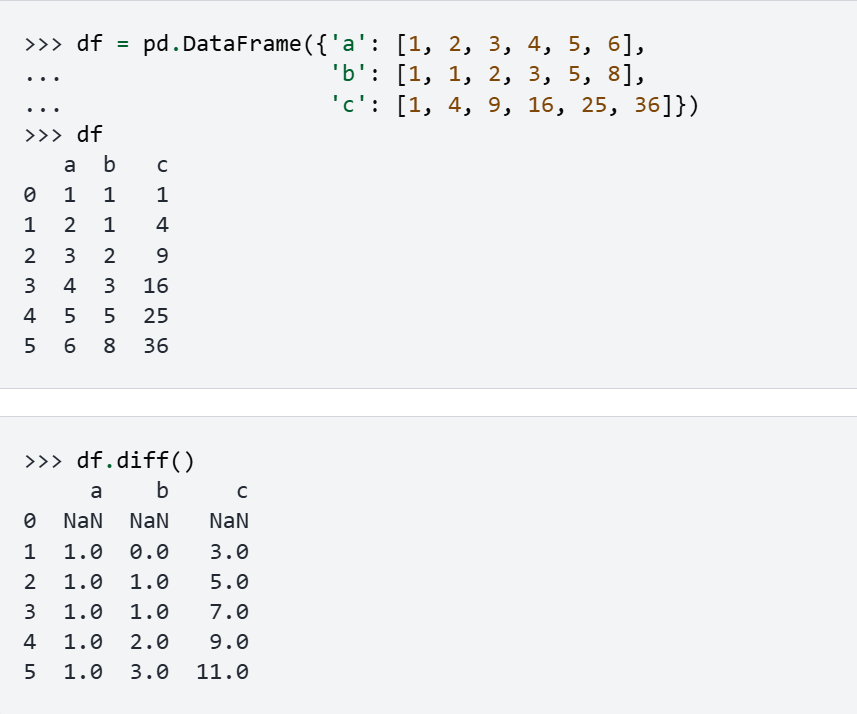
Fractional change between the current and a prior element. Computes the fractional change from the immediately previous row by default. This is useful in comparing the fraction of change in a time series of element





## **2.Difference:**

Calculates the difference of a DataFrame element compared with another element in the DataFrame (default is element in previous row). It takes parameter periods to shift for calculating differences, and accepts negative values.



## **3.Average gain:**

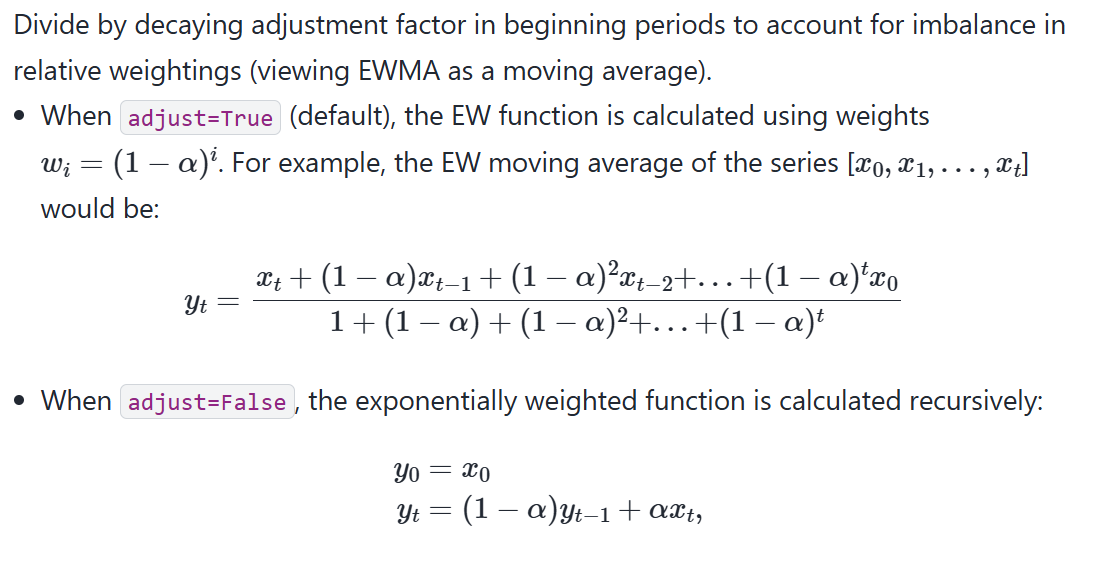
The average gain for the first 14 days is the arithmetic means of the gains

## **4. Average loss**

The average loss for the first 14 days is the arithmetic means of the loss.

## **5. Exponential weighted moving average**

Provide exponentially weighted (EW) calculations. We have calculated ewm with alpha as parameter. The alpha value specifies the smoothing factor; the value range is 0-1.



For example we will calculate the average gain as [(1.72\*13)+0.00]/14 = 1.6 if the previous gain is 1.72 and present gain is 0.00 considering the 14 day window.

## **Relative Strength (RS):**

RS measures the ratio of average gains to average losses over a specific period. A higher RS means that the asset has experienced more gains than losses, indicating bullish momentum, while a lower RS indicates more losses, signaling bearish momentum.

*RS = (Avg Gain)/(Avg Loss)*

## **RSI:**

RSI converts the RS into a standardized index ranging from 0 to 100. It is used to identify overbought or oversold conditions: values above 70 typically indicate overbought (potential price drop), while values below 30 indicate oversold (potential price rise).

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

## **Buy Cost Calculation**

The Buy Cost determines the total amount a user needs to spend when purchasing cryptocurrency. It is calculated by multiplying the price per coin by the number of coins the user wants to buy.A higher price or greater quantity of coins will result in a higher total cost. This calculation helps users understand the total expenditure before confirming a purchase.

* Price per Coin = Closed Price

Buy Cost (Total)=Price per Coin×Number of Coins

## **Sell Cost Calculation**

The Sell Cost (or total sale value) represents the total amount a user receives when selling cryptocurrency. It is calculated by multiplying the current price per coin by the number of coins being sold.A higher coin price or larger quantity results in a greater total return. This calculation helps users estimate their total proceeds before completing the sale transaction.

* Price per Coin = Closed Price

Sell Return (Total)=Price per Coin×Number of Coins

## **Buy Transaction – Cost per Coin**

The Cost per Coin represents the effective purchase price of each individual cryptocurrency coin in a buy transaction. It is calculated by dividing the total amount spent by the number of coins purchased.

This value helps determine how much was paid for each coin, which is essential for tracking average purchase prices, evaluating profitability, and calculating gains or losses when selling later.

Cost per Coin=Total Amount \ Number of Coins

## **Sell Transaction – Amount Earned**

The Sell Amount (Total) represents the total value received from selling a certain number of cryptocurrency coins. It is calculated by multiplying the number of coins sold by the price (or cost) per coin at the time of the sale.

This calculation helps determine the total proceeds a user earns from a sell transaction and serves as the basis for computing overall profit or loss when compared to the original purchase cost.

* Price per Coin = Closed Price

Sell Amount (Total)=Number of Coins×Price per Coin

## **Send to Balance (Deduction)**

The Send to Balance (Deduction) operation updates the user’s wallet by reducing the current balance when an amount is transferred or spent. It ensures that the wallet reflects the correct remaining funds after the transaction.

The deduction is performed by subtracting the entered amount from the existing wallet balance. This process helps maintain accurate records of available funds and prevents overspending.

Wallet Balance (new)=Wallet Balance (old)−Entered Amount

## **Calculate Old Value (USD worth of selected coin before swap)**

The Old Value represents the total USD worth of the cryptocurrency being swapped before the transaction takes place. It is determined by multiplying the existing quantity of the coin by its current market price.

This calculation helps assess how much value the user is exchanging from their existing holdings, serving as the “Existing” side of the swap operation.

Old Value (USD)=Existing Coin Quantity× Existing Coin Price

## **Calculate New Quantity** (how many targets coins user will get)

The New Quantity represents the number of target cryptocurrency coins a user will receive after performing a swap. It is calculated by dividing the USD worth of the existing coin (Old Value) by the current price of the target coin.

This calculation ensures that the user receives the correct number of coins based on the value they are exchanging, maintaining fair and accurate swaps.

New Quantity (Target Coin)=Old Value (USD) / Target Coin Price

## **Swap Validation** (before calculation)

The Swap Validation step ensures that the swap request is legitimate before performing any calculations. It checks that the quantity of the coin being swapped, the price of the coin being swapped from, and the price of the target coin are all positive values.

This prevents invalid or nonsensical transactions, such as swapping zero coins or using invalid prices, which could lead to errors or financial inconsistencies.

Swap is valid only if (From Quantity>0)∧(From Price>0)∧(To Price>0)

# **USE CASES**

## **Use Case 1: Administrator Login and Management**

|  |  |
| --- | --- |
| **Use Case ID** | UC - 1 |
| **UseCase Name** | Administrator Login and Management |
| **Brief Description** | This use case enables the system administrator to log in securely, verify users, and manage data within the platform. The admin can approve or deactivate accounts, review transaction histories, and oversee sentiment analytics to ensure smooth system operation. |
| **Frequency of Use** | Regularly whenever administrative monitoring or user verification is required. |
| **Assumptions** | • The administrator has valid credentials.  • The server and database are active and accessible. |
| **Preconditions** | • The administrator must have an active admin account.  • Internet connection must be stable for dashboard access. |
| **PostConditions** | • If successful – Admin dashboard is displayed with full control access.  • If unsuccessful – Error message “Invalid credentials” is shown. |

|  |  |
| --- | --- |
| **Trigger** | Administrator clicks on “Admin Login” and submits login details. |
| **Normal Flow:** | • Administrator selects “Admin Login.”  • The system displays the admin login form.  • Admin enters username and password.  • System validates credentials.  • On success, the admin dashboard is displayed with user and transaction data.  • Administrator manages user accounts, views logs, or updates records. |
| **Exceptions:** | • Invalid credentials – System prompts: “Incorrect username or password.”  • Database connection failure – System displays: “Unable to connect to database.” |
| **Includes:** | • Admin login form.  • MySQL validation for authentication.  • Admin dashboard interface. |
| **Special Requirements:** | • Multi-level authentication should be implemented for enhanced security. |
| **Alternative Flow:** | • If the admin cancels login – System returns to the main page. |

|  |  |
| --- | --- |
|  |  |
| **Notes and Issues** | • Future improvement: Role-based permissions for multiple admin types. |

## **Use Case 2: User Registration and Authentication**

|  |  |
| --- | --- |
| **Use Case ID** | UC - 2 |
| **UseCase Name** | User Registration and Authentication |
| **Brief Description** | This use case allows new users to register on the platform and existing users to log in securely. It ensures only verified users can access the trading and sentiment analysis modules. |
| **Frequency of Use** | Frequently whenever new users join or existing users log in. |
| **Assumptions** | • The user has access to a valid email address.  • The server and database are running. |
| **Preconditions** | • Registration form and database connection must be active. |
| **PostConditions** | • Successful login – Redirects to user dashboard.  • Failed login – Displays relevant error message. |

|  |  |
| --- | --- |
| **Trigger** | The user clicks on “Register” or “Login” and submits details. |
| **Normal Flow:** | • The user navigates to the registration page.  • System prompts for personal details and password.  • The user submits the form.  • The system stores user data in the users table.  • For login, the user provides credentials.  • The system validates credentials and grants access. |
| **Exceptions:** | • Duplicate email – System displays: “Email already exists.”  • Invalid credentials – System displays: “Incorrect password.” |
| **Includes:** | • Registration and login forms.  • Validation via MySQL database. |
| **Special Requirements:** | • Passwords must be securely hashed before storage. |
| **Alternative Flow:** | • If the user forgets password – Redirect to password recovery. |

|  |  |
| --- | --- |
|  |  |
| **Notes and Issues** | • Future scope: Integrate two-factor authentication (2FA) for better security. |

## **Use Case 3: Wallet Management**

|  |  |
| --- | --- |
| **Use Case ID** | UC - 3 |
| **UseCase Name** | Wallet Management |
| **Brief Description** | This use case enables users to manage their digital wallet by adding funds, viewing balances, and tracking transaction summaries. |
| **Frequency of Use** | Frequently before performing trades or withdrawals. |
| **Assumptions** | • The user is logged in.  • The database is operational. |
| **Preconditions** | • User accounts and wallets must exist in the system. |
| **PostConditions** | • Wallet balance is updated and stored in the database. |

|  |  |
| --- | --- |
| **Trigger** | The user clicks on “Add Balance” or “View Wallet.” |
| **Normal Flow:** | • The user navigates to the wallet section.  • The system displays current balance.  • The user selects “Add Funds.”  • The system updates wallet balance in real time.  • Users can view updated transaction history. |
| **Exceptions:** | • Invalid amount – System displays: “Please enter a valid amount.”  • Database failure – “Wallet update failed.” |
| **Includes:** | • Wallet form and balance view.  • MySQL update query for wallet table. |
| **Special Requirements:** | • Balance must update instantly after each transaction. |
| **Alternative Flow:** | • If the user cancels – System returns to dashboard without change. |

|  |  |
| --- | --- |
|  |  |
| **Notes and Issues** | • Future enhancement: Integrate real-time currency conversion rates. |

## **Use Case 4: Cryptocurrency Buy/Sell Operation**

|  |  |
| --- | --- |
| **Use Case ID** | UC - 4 |
| **UseCase Name** | Cryptocurrency Buy/Sell Operation |
| **Brief Description** | This use case allows users to buy or sell cryptocurrencies using their wallet balance. Each transaction is logged in the system with timestamp and coin details. |
| **Frequency of Use** | Frequently – whenever users perform trading activities. |
| **Assumptions** | • The user has sufficient wallet balance.  • Market data is available. |
| **Preconditions** | • Users must be logged in.  • Wallet balance > 0. |
| **PostConditions** | • Wallet balance and portfolio are updated in the database. |

|  |  |
| --- | --- |
| **Trigger** | User clicks on “Buy” or “Sell” from the dashboard. |
| **Normal Flow:** | • The user selects cryptocurrency to trade.  • The system displays the current market price.  • The user specifies the quantity and confirms the trade.  • The system validates the transaction.  • Database updates wallet and transaction tables.  • The system displays success confirmation. |
| **Exceptions:** | • Insufficient balance – “Not enough funds.”  • Invalid quantity – “Invalid coin quantity.” |
| **Includes:** | • Market data from yFinance.  • Wallet and transaction tables. |
| **Special Requirements:** | • Transaction timestamp and status must be recorded. |
| **Alternative Flow:** | • If the user cancels – System returns to dashboard without changes. |

|  |  |
| --- | --- |
|  |  |
| **Notes and Issues** | • Future scope: Real-time integration with crypto exchange APIs. |

## **Use Case 5: Peer-to-Peer (P2P) Trading**

|  |  |
| --- | --- |
| **Use Case ID** | UC - 5 |
| **UseCase Name** | Peer-to-Peer (P2P) Trading |
| **Brief Description** | This use case enables users to directly sell and purchase cryptocurrencies from one another. Sellers list offers with price and quantity, and buyers select suitable listings to complete trades. |
| **Frequency of Use** | Occasionally – when users prefer direct trading instead of system-based transactions. |
| **Assumptions** | • Both buyer and seller have verified accounts.  • The listed coin is available in the seller's wallet. |
| **Preconditions** | • Users must be logged in.  • The seller must have sufficient coin balance. |
| **PostConditions** | • Successful transaction updates both buyer’s and seller’s wallet balances. |

|  |  |
| --- | --- |
| **Trigger** | The user clicks on the “P2P Trade” option and selects a listing. |
| **Normal Flow:** | • Seller lists a coin for sale with price and quantity.  • Buyer browses listings and selects one.  • The system validates trade availability.  • Wallet balances are updated for both parties.  • Transactions are recorded in the p2p\_orders table. |
| **Exceptions:** | • Listing expired – “Offer no longer available.”  • Insufficient funds – “Not enough balance to purchase.” |
| **Includes:** | • P2P trading interface.  • Database update for both users. |
| **Special Requirements:** | • The system must prevent duplicate or partial transactions. |
| **Alternative Flow:** | • Buyer cancels transaction – System reverts to listing page. |

|  |  |
| --- | --- |
|  |  |
| **Notes and Issues** | • Enhancement: Implement an escrow system for secured P2P exchanges. |

## **Use Case 6: News Scraping and Data Storage**

|  |  |
| --- | --- |
| **Use Case ID** | UC - 6 |
| **UseCase Name** | News Scraping and Data Storage |
| **Brief Description** | This use case involves the automated extraction of cryptocurrency-related news headlines from TradingView using Selenium WebDriver. The system stores the scraped information for further sentiment analysis. |
| **Frequency of Use** | Automatically scheduled at defined intervals. |
| **Assumptions** | • Internet connectivity is stable.  • TradingView website structure remains unchanged. |
| **Preconditions** | • Selenium WebDriver and browser driver are configured. |
| **PostConditions** | • News headlines are saved into news\_data.csv and MySQL database. |

|  |  |
| --- | --- |
| **Trigger** | Cron job or manual initiation of scraping script. |
| **Normal Flow:** | • System launches Selenium WebDriver.  • Navigates to TradingView’s news page.  • Extracts article title, source, and date.  • Filters duplicates.  • Saves data into CSV and MySQL. |
| **Exceptions:** | • No network access – “Unable to fetch data.”  • Structural change – “Web elements not found.” |
| **Includes:** | • Selenium automation.  • CSV and database integration. |
| **Special Requirements:** | • The logging mechanism must record each scraping cycle. |
| **Alternative Flow:** | • Manual rerun by admin in case of failure. |

|  |  |
| --- | --- |
|  |  |
| **Notes and Issues** | • May require periodic XPath updates if site layout changes. |

## **Use Case 7: Sentiment Analysis and Score Computation**

|  |  |
| --- | --- |
| **Use Case ID** | UC - 7 |
| **UseCase Name** | Sentiment Analysis and Score Computation |
| **Brief Description** | This use case allows the system to process news headlines through the CryptoBERT model to determine market sentiment for each cryptocurrency. |
| **Frequency of Use** | Automatically executed after news scraping or on admin triggers. |
| **Assumptions** | • The CryptoBERT model is preloaded and available.  • News data is available in the database. |
| **Preconditions** | • Model dependencies and tokenizer must be installed.  • Database connection must be active. |
| **PostConditions** | • Sentiment scores are generated and inserted into the sentiment\_scores table. |

|  |  |
| --- | --- |
| **Trigger** | Admin or system initiates the sentiment analysis process. |
| **Normal Flow:** | • System loads CryptoBERT and tokenizer.  • Fetches news headlines from database.  • Tokenizes and predicts sentiment class.  • Calculates weighted score.  • Averages score per ticker.  • Stores results in MySQL. |
| **Exceptions:** | • Model not found – “Sentiment model unavailable.”  • Empty dataset – “No headlines to analyze.” |
| **Includes:** | • PyTorch model inference.  • Database update module. |
| **Special Requirements:** | • Each record must include a timestamp for traceability. |
| **Alternative Flow:** | • Manual re-run if previous execution failed. |

|  |  |
| --- | --- |
|  |  |
| **Notes and Issues** | • The model may require periodic retraining to maintain accuracy. |

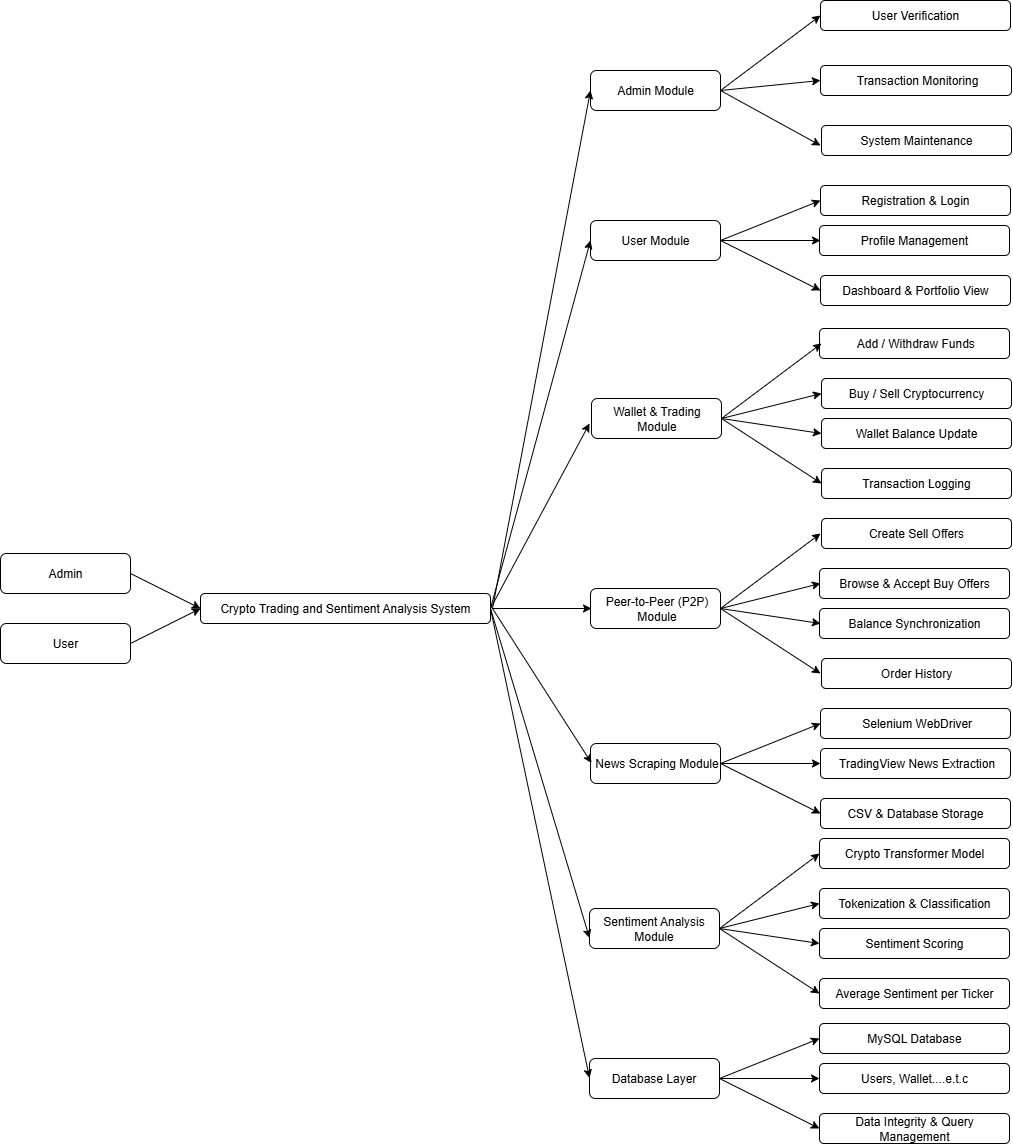
## **Use Case 8: Dashboard Visualization and User Analytics**

|  |  |
| --- | --- |
| **Use Case ID** | UC - 8 |
| **UseCase Name** | Dashboard Visualization and User Analytics |
| **Brief Description** | This use case provides users with an interactive dashboard where they can view cryptocurrency price movements, percentage changes, and sentiment scores. The dashboard integrates both analytical and graphical information, enabling users to understand current market trends and their own trading performance. |
| **Frequency of Use** | Frequently – each time a user logs in or after completing a trade. |
| **Assumptions** | • The user is logged in.  • The database contains updated crypto and sentiment data.  • Flask server is running. |
| **Preconditions** | • Market data and sentiment tables must be synchronized.  • HTML and Jinja2 templates must be loaded correctly. |
| **PostConditions** | • Dashboard displays refreshed values for crypto metrics and user portfolio. |

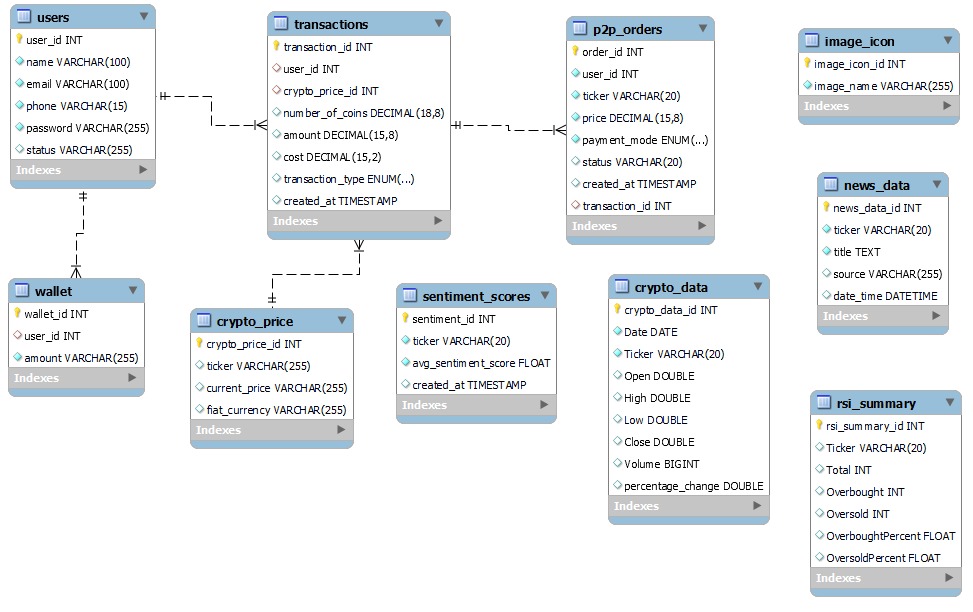
|  |  |
| --- | --- |
| **Trigger** | The user navigates to the “Dashboard” section after login. |
| **Normal Flow:** | • The user logs into the system.  • The system retrieves real-time price and sentiment data from MySQL.  • Flask dynamically renders the dashboard.  • Charts and tables display coin trends, average sentiment, and balance updates.  • Users interact with filters or sort functions for data exploration. |
| **Exceptions:** | • Data retrieval failure – System displays: “Unable to fetch data.”  • Session timeout – “Please log in again.” |
| **Includes:** | • Market visualization templates.  • Database read operations.  • Chart components for analytics. |
| **Special Requirements:** | • Dashboard must auto-refresh at fixed intervals.  • Data updates should not interrupt user interactions. |
| **Alternative Flow:** | • If the user logs out – System redirects to login screen.  • If the network fails – System shows last cached data. |

|  |  |
| --- | --- |
|  |  |
| **Notes and Issues** | • Future enhancement: Integrate AI-based market forecasting graphs |

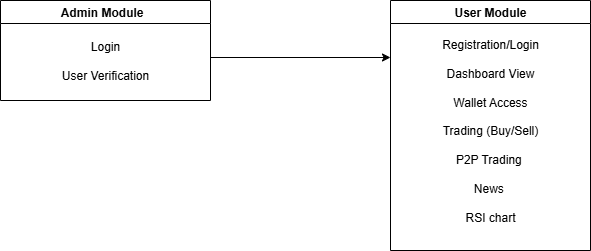
# **ARCHITECTURE DIAGRAM**



# **ER DIAGRAM**

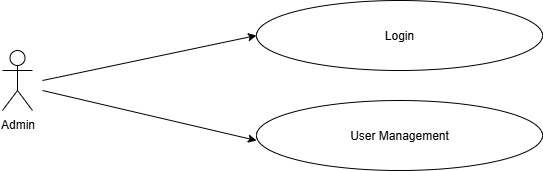


# **FLOW DIAGRAM**

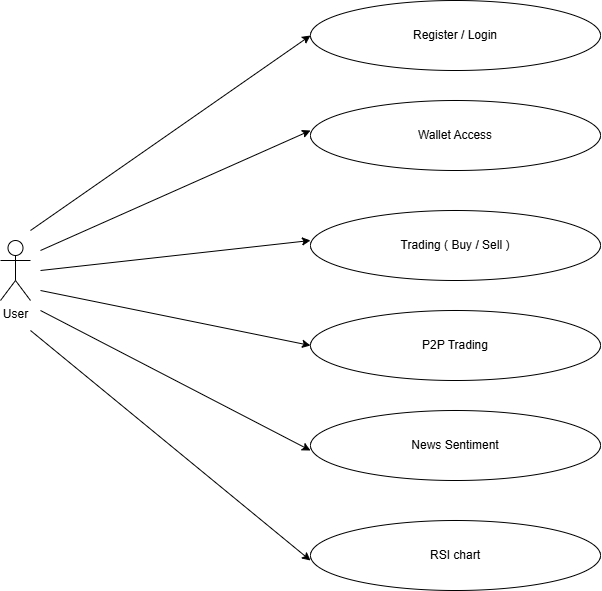


# **UML DIAGRAM**

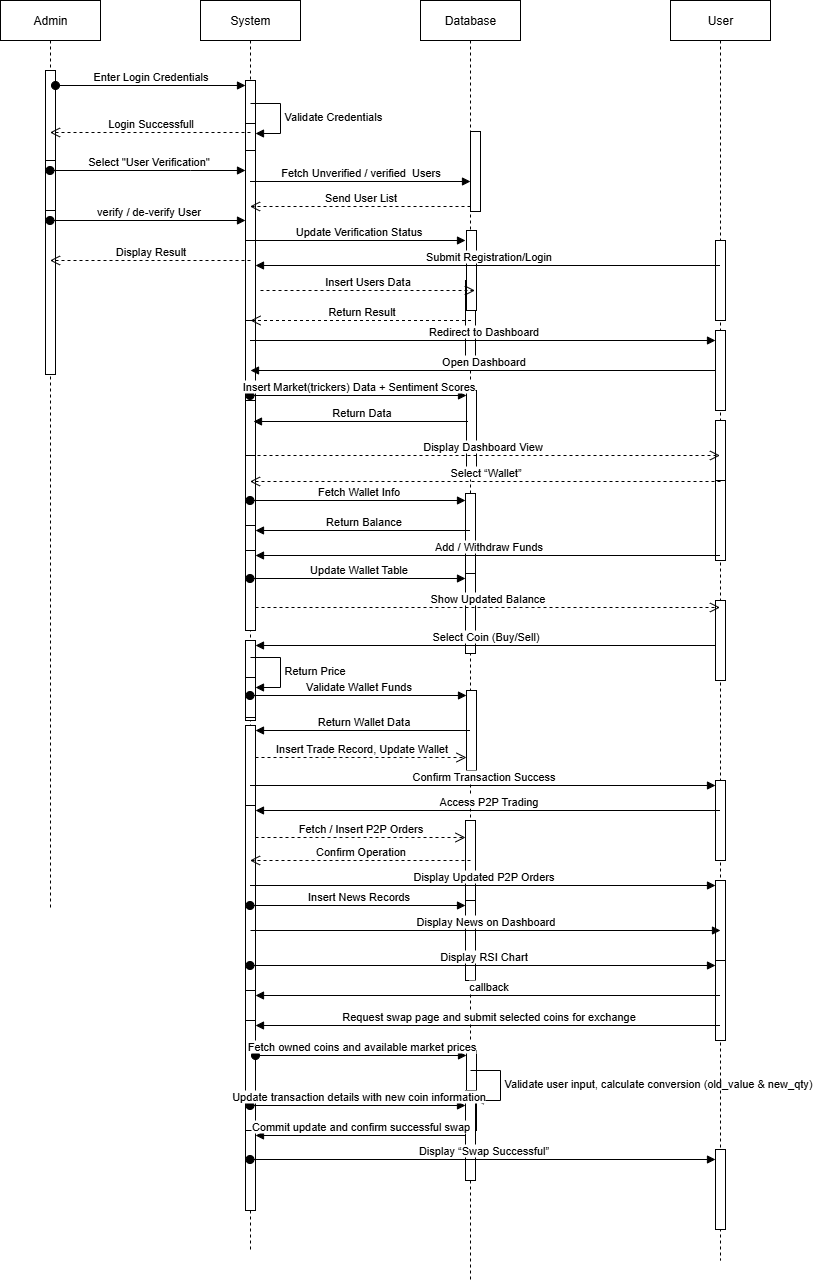
## **Admin**



## **User**



## **Sequence Diagram:**



# **CONCLUSION**

The Crypto Trading and Sentiment Analysis System successfully demonstrates how artificial intelligence and data analytics can be combined to enhance decision-making in the cryptocurrency sector. By automating the acquisition of financial data, analyzing sentiment through deep learning models, and providing a user-driven trading interface, the system serves as a complete prototype for intelligent crypto market analysis.

The project emphasizes the real-world applicability of modern technologies such as NLP, Flask web frameworks, and cloud data storage. It provides an educational yet practical tool for users and researchers to study the influence of sentiment on market behavior.

# **FUTURE SCOPE**

The system can be extended in numerous directions to enhance its analytical and operational capabilities. Integration of real blockchain APIs to enable live trading on decentralized exchanges. Implementation of predictive models such as LSTM or Transformer-based architectures for price forecasting. Deployment of interactive dashboards with real-time graph visualizations. Addition of advanced authentication methods such as multi-factor verification for greater security. Expansion to include Non-Fungible Token (NFT) analytics and cross-chain transaction tracking. Hosting the complete application on cloud platforms (AWS, Azure) to ensure scalability and availability. With these future enhancements, the platform can evolve into a sophisticated intelligent trading assistant, capable of real-time decision-making and predictive financial insights.

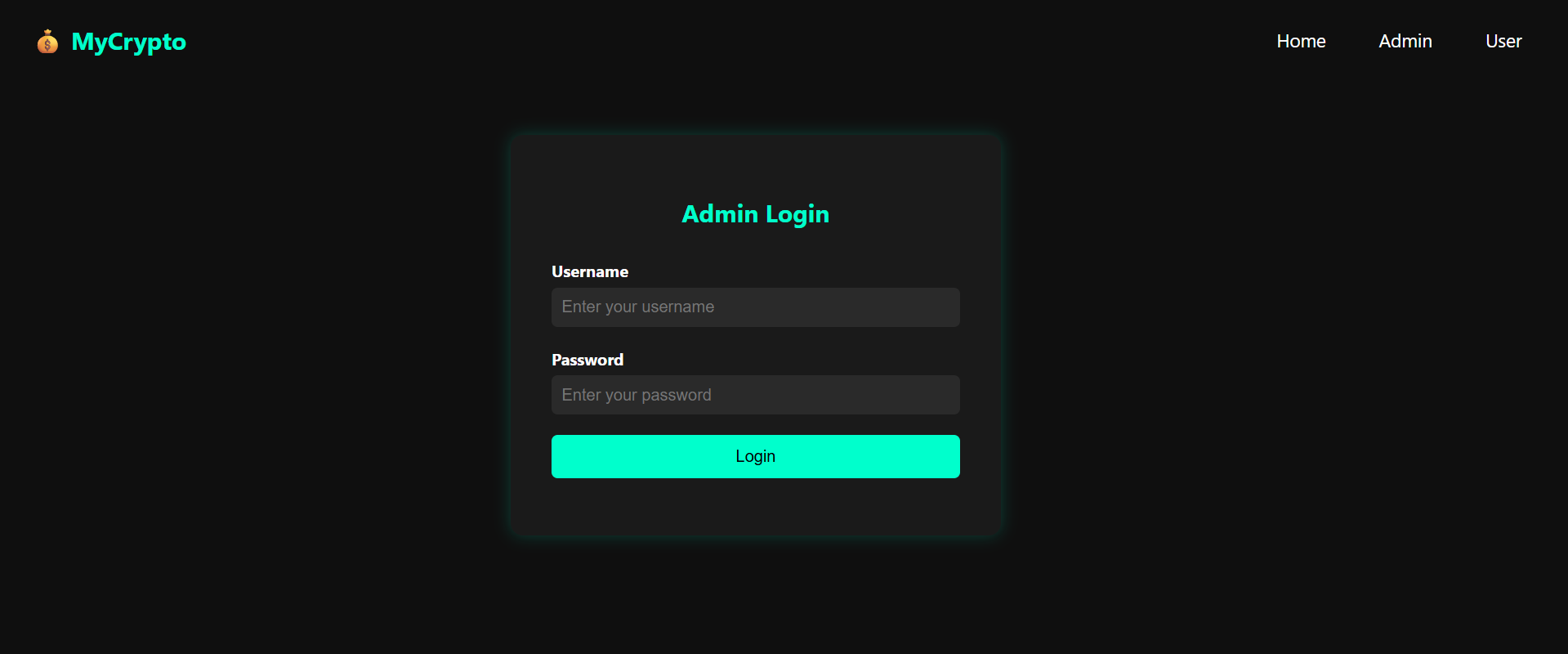
# **USER INTERFACE**

## **Index:**



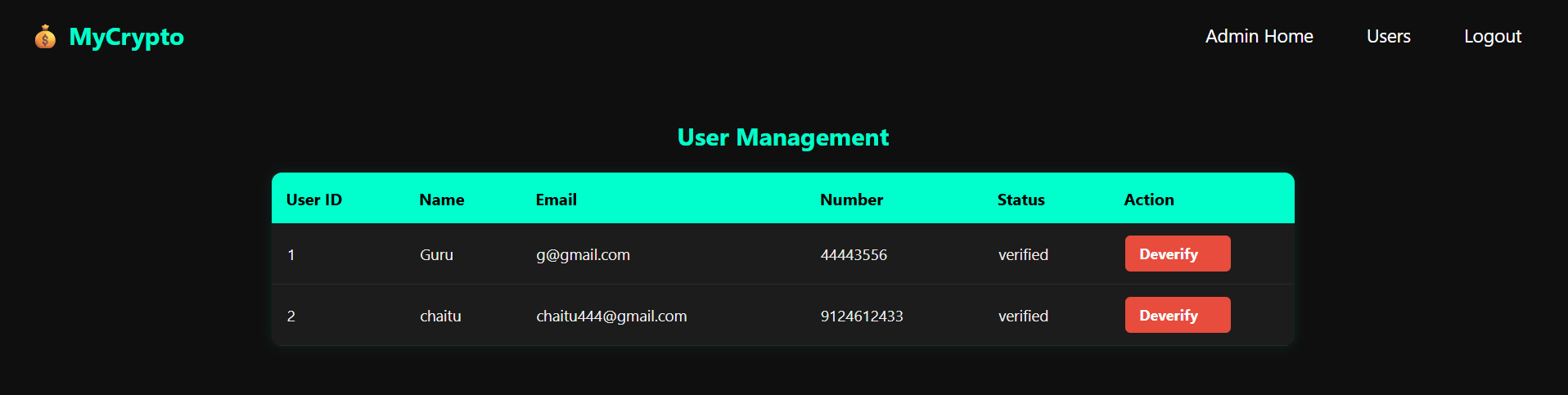
Index page here we can see Trickers Data

## **Admin Login**



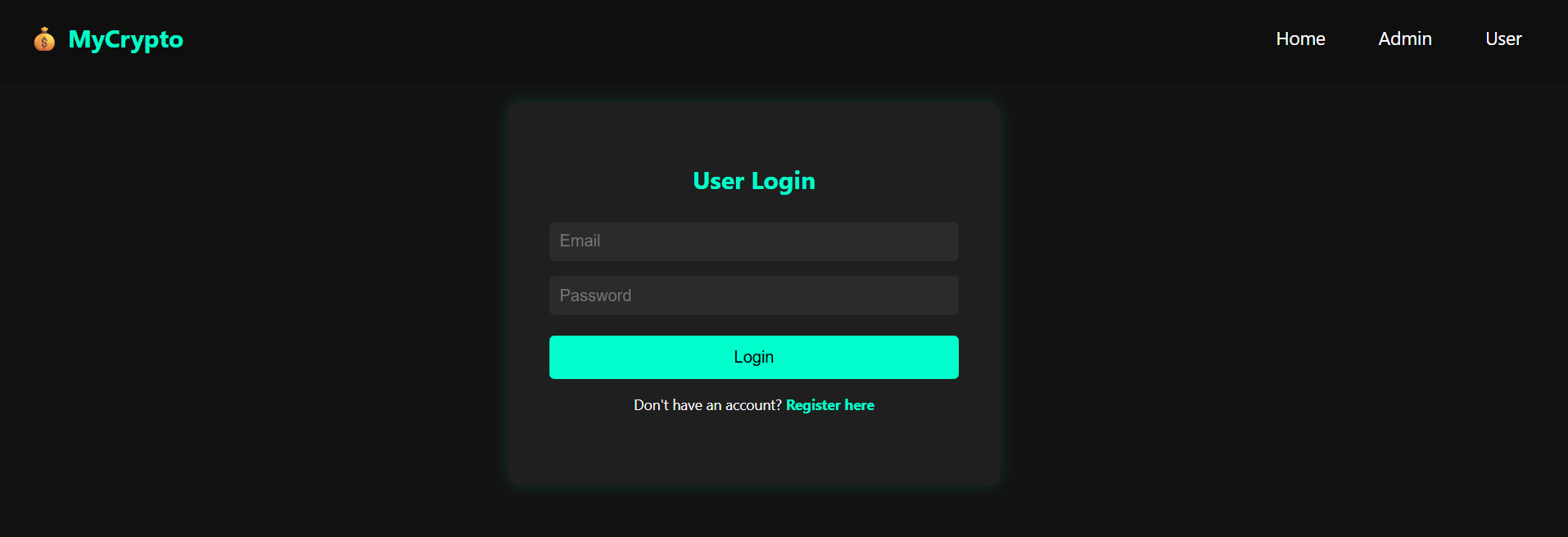
Admin login page here the admin login with default credentials like username and password.

## **User Management**



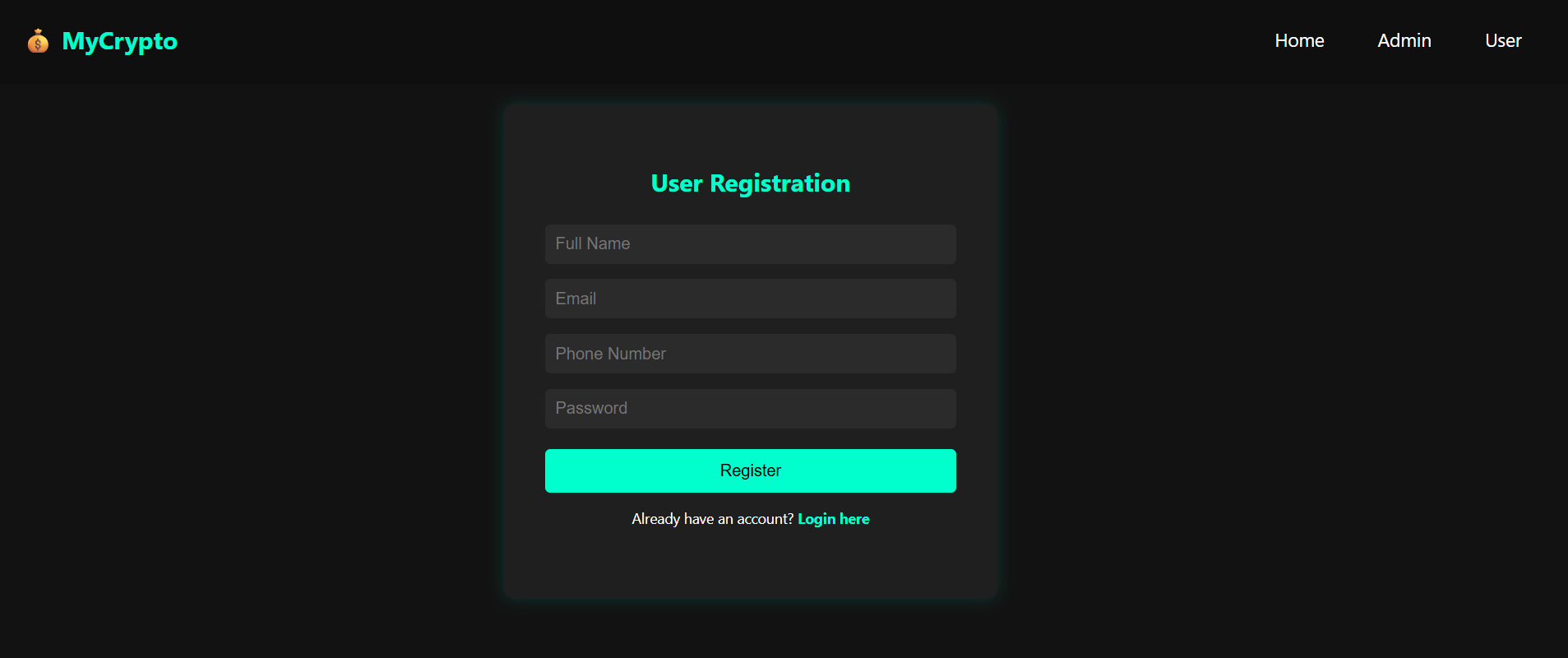
The admin can view the users details and verify / De-verify their accounts

## **User Login**



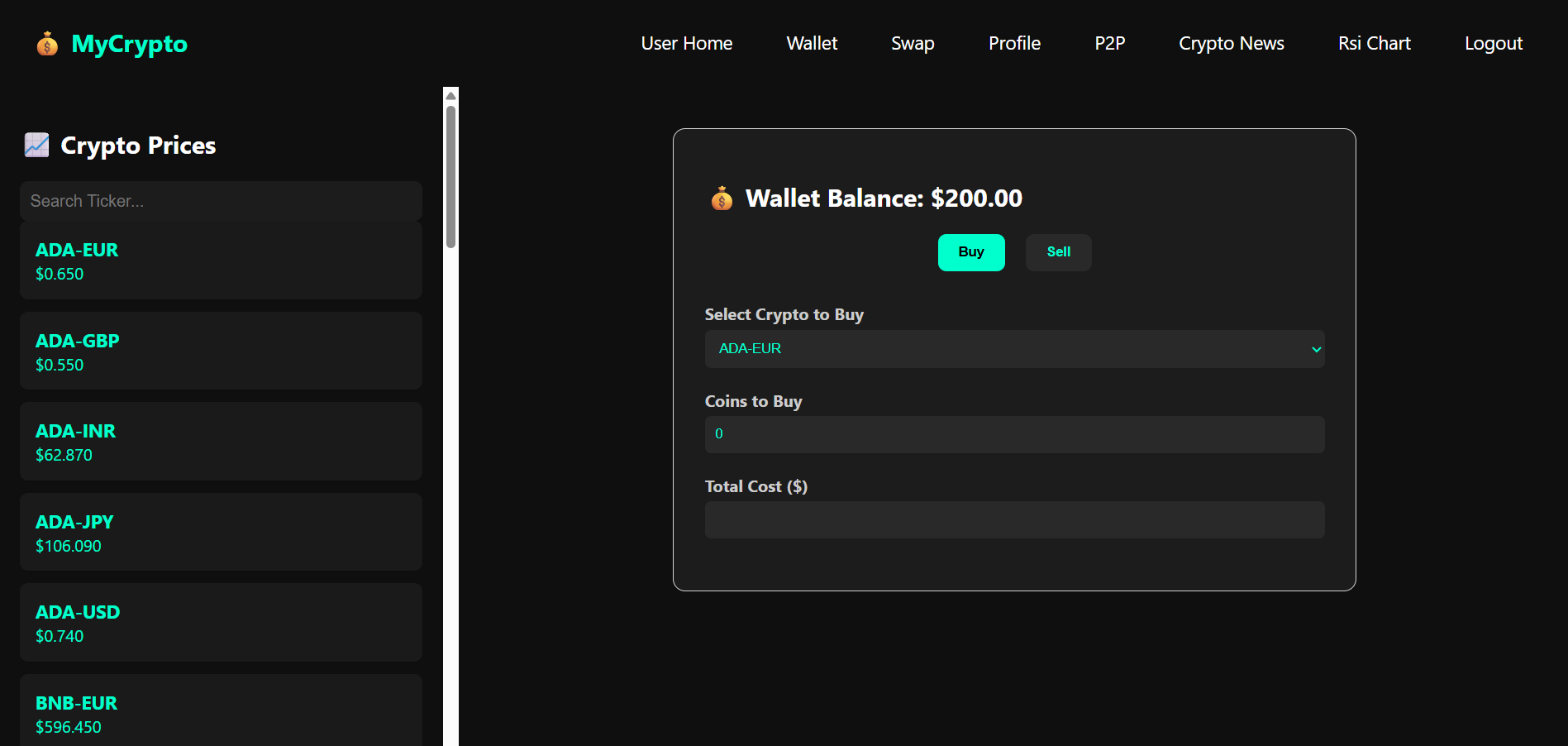
User login with registered details like email and password

## **User Registration**



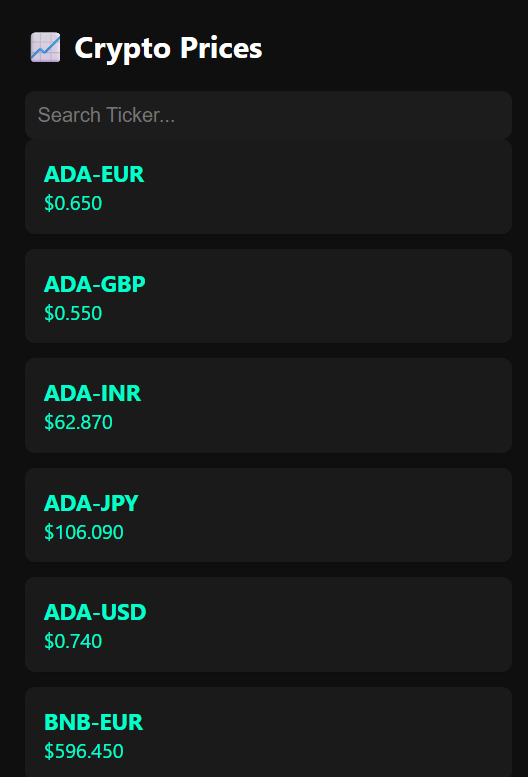
Users register with their details like name, email, phone and password.

## **User home page**



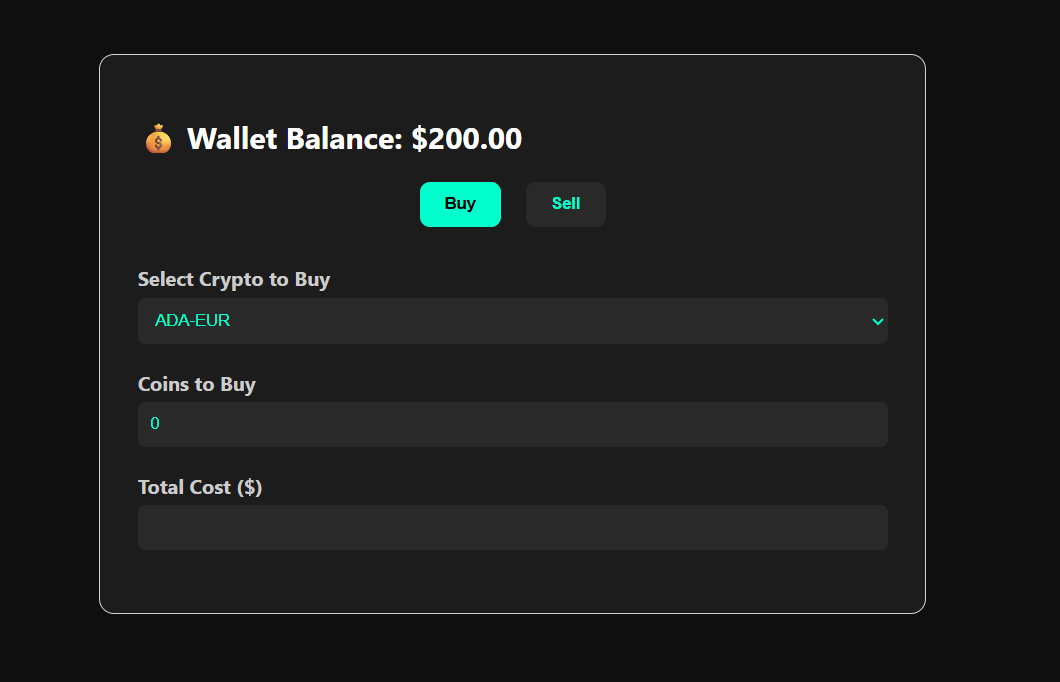
User home page

## **Crypto Prices**



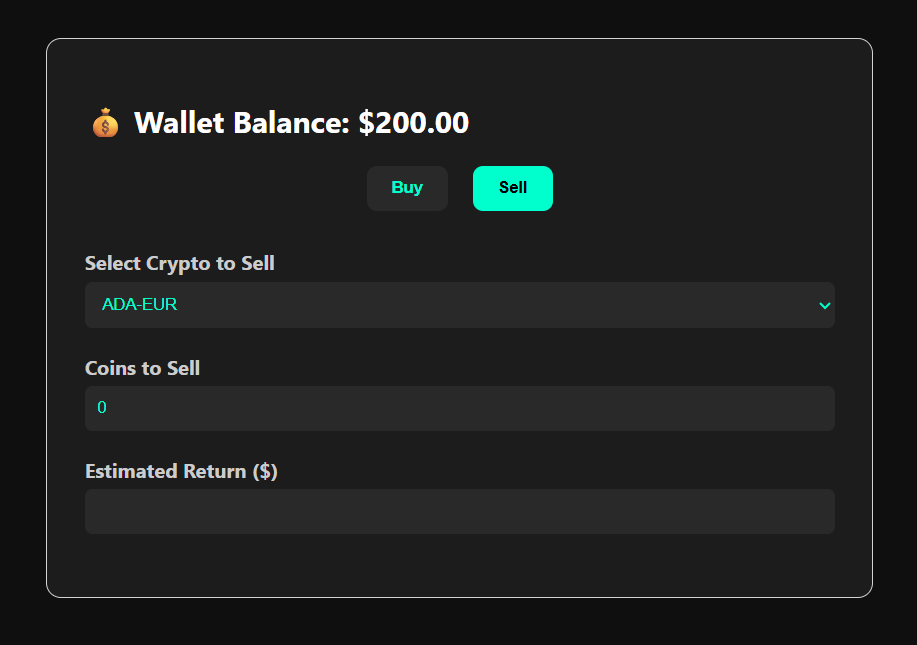
Here the users can view tricker wise prices

## **Buy Crypto**



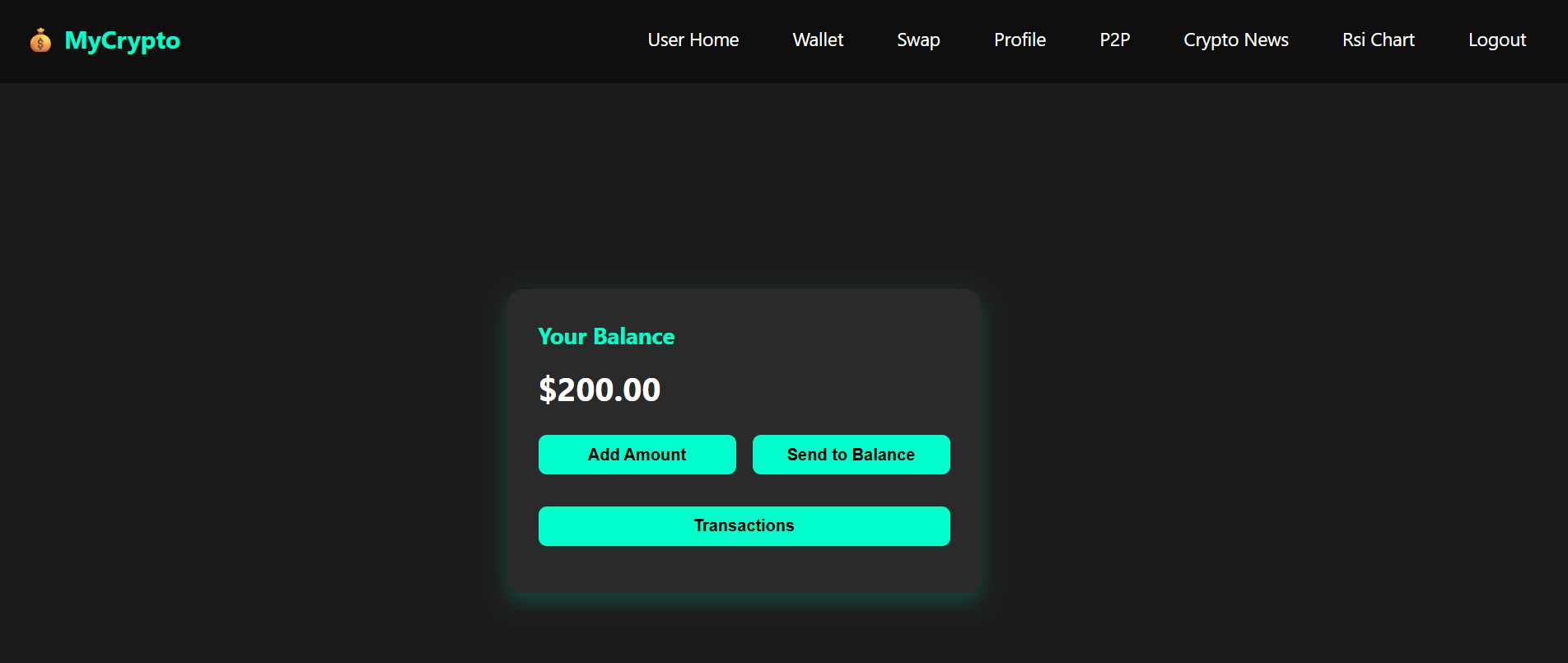
The user can buy tricker the cost Total will be >= wallet balance, then only the user can buy the trickers.

## **Sell Crypto**



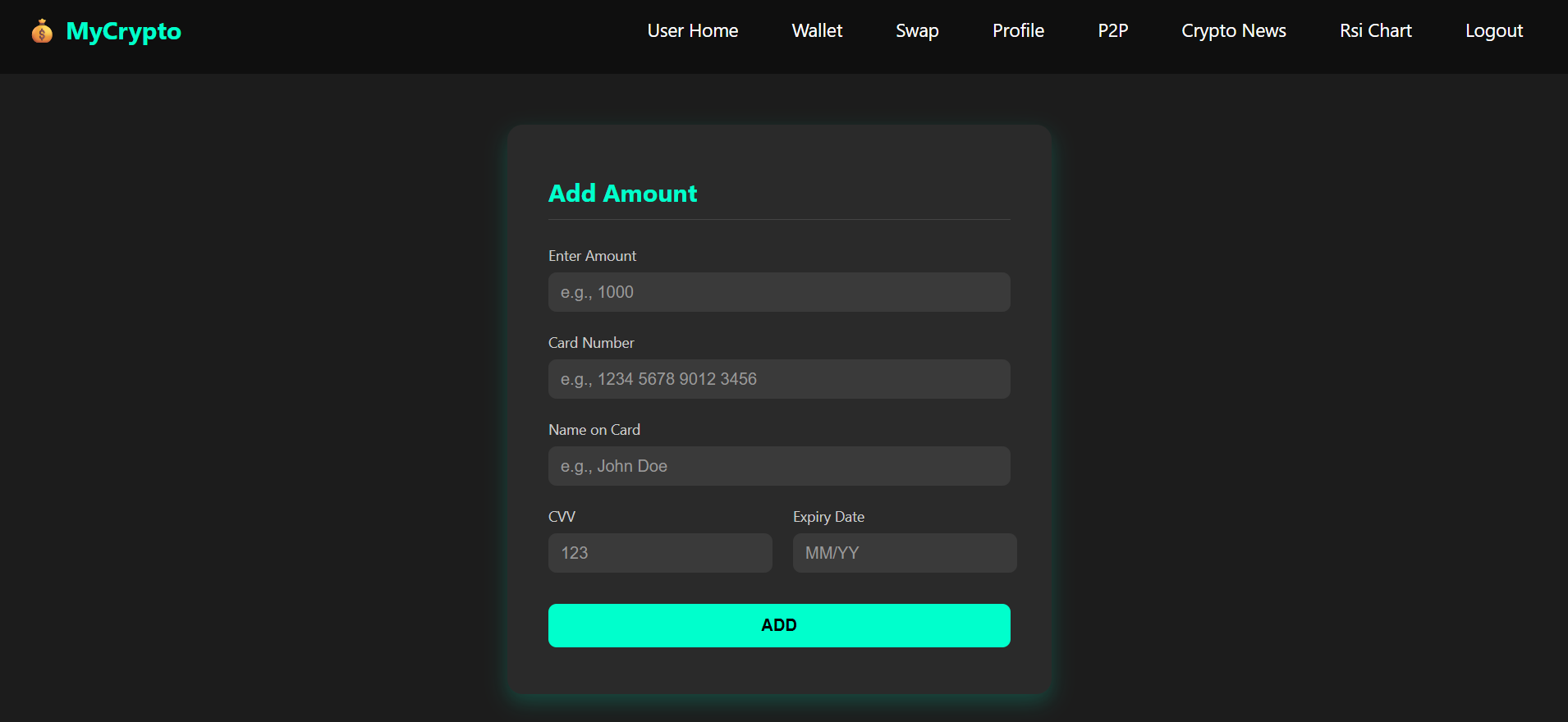
Sell crypto, the users owned trickers only sell here.

## **User Wallet**



Users wallet

## **Add Amount**



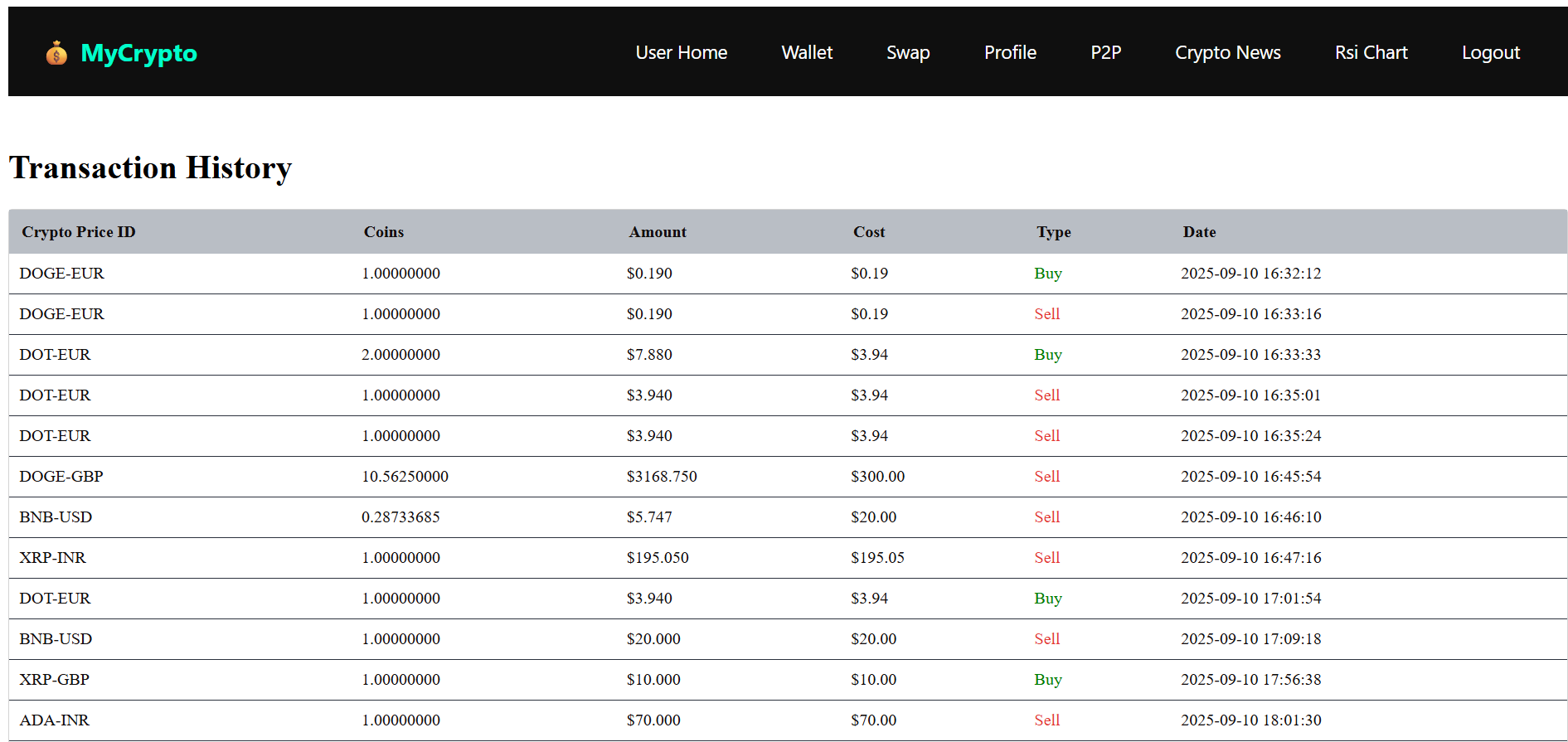
In the wallet the user can add an amount to the Wallet,through details like amount,card number, name on the card, cvv and expiry date.

## **Send to Bank**



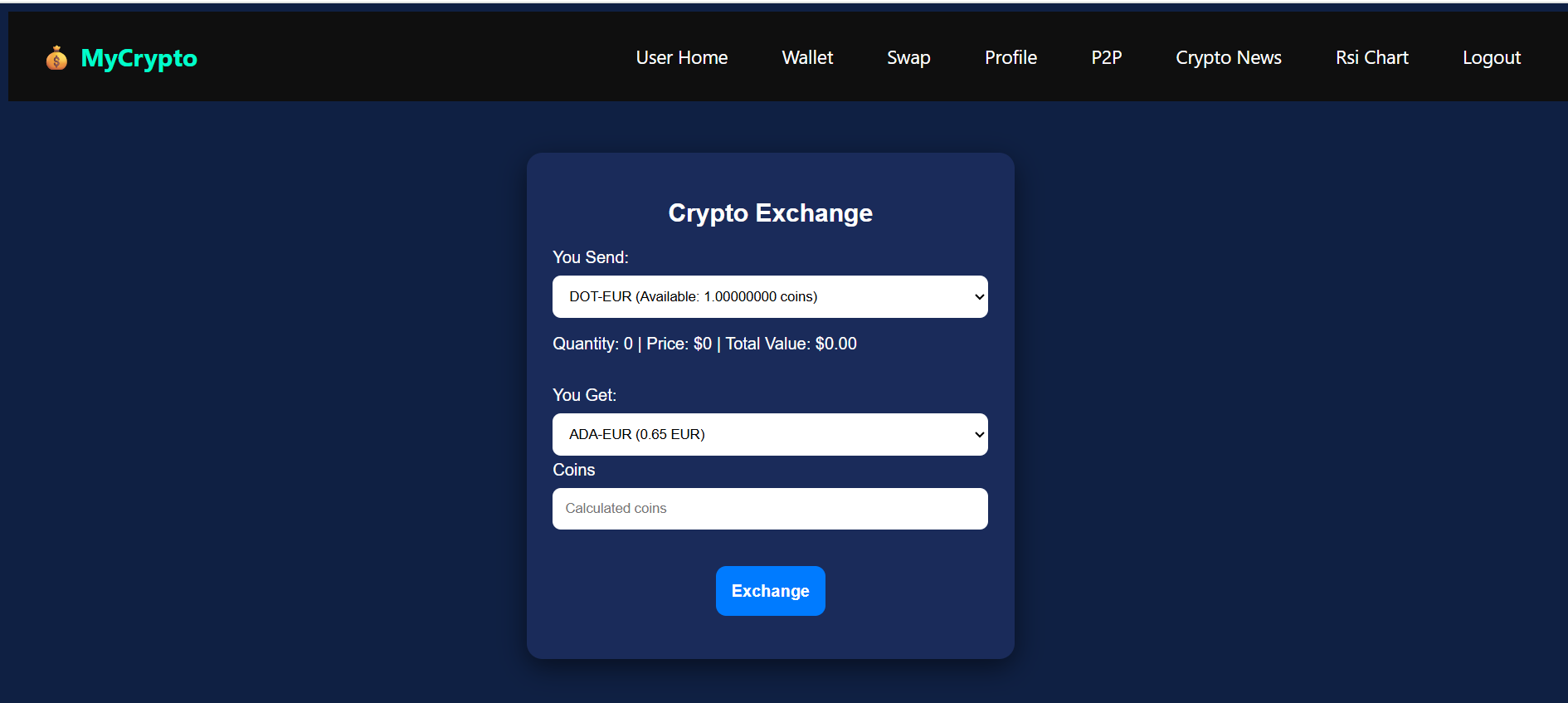
Here the users can self transaction crypto wallet amount to self bank account

## **View Transactions**



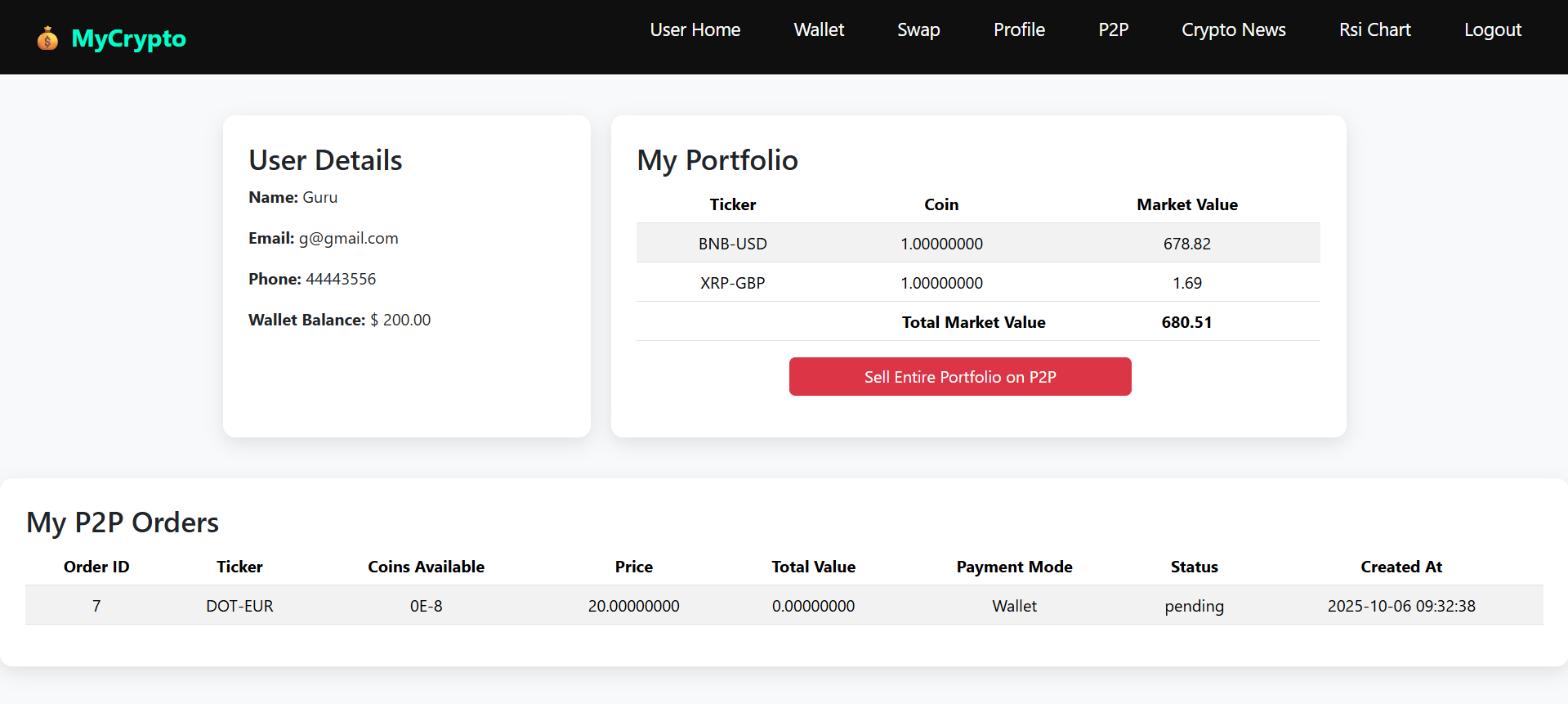
Here the user can we his crypto transactions

## **Swap**



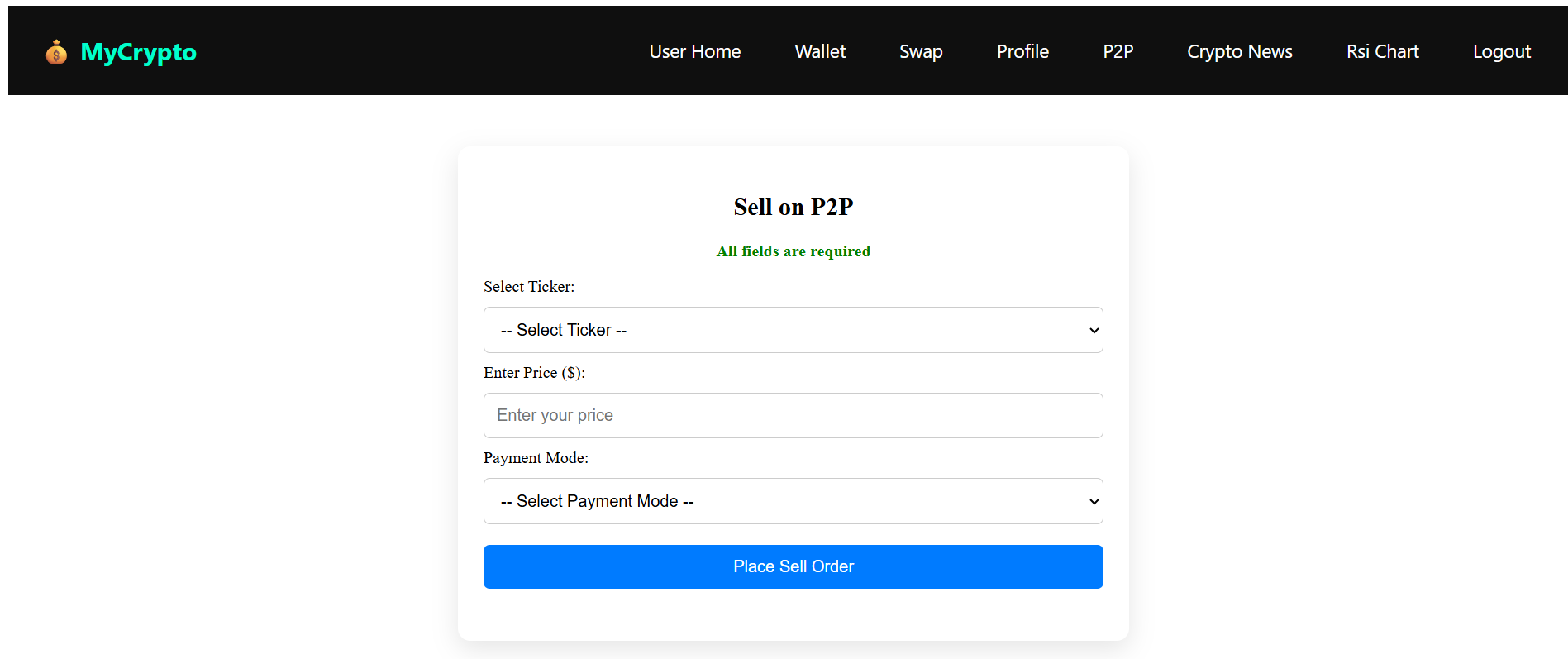
Here the user can swap his tricker with another tricker through his owned coins.

## **User Profile**



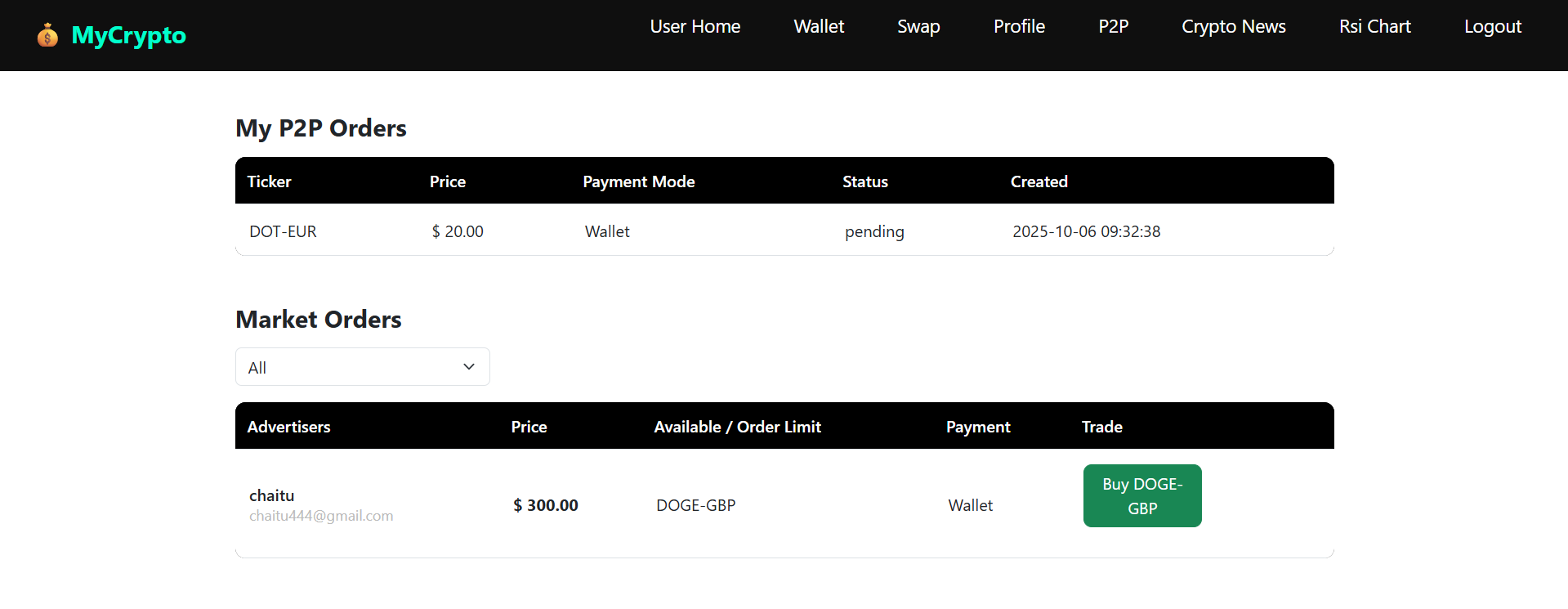
In the user profile the users can view his owned coins and see his p2p orders and sell his p2p orders also.

## **Sell on P2P:**



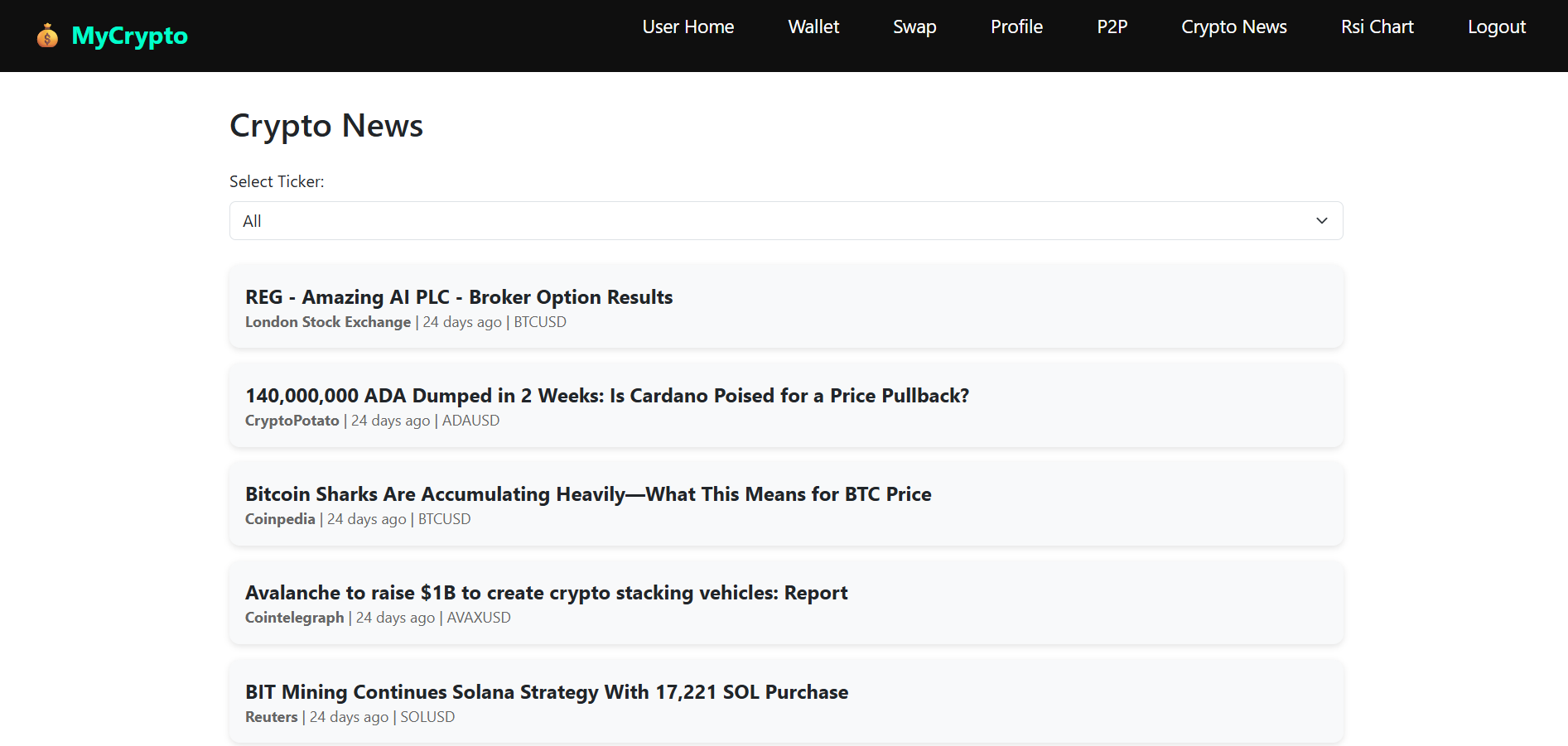
Sell p2p here the user can choose his owned coins and enter his own price for sell that tricker.

## **P2P Orders**



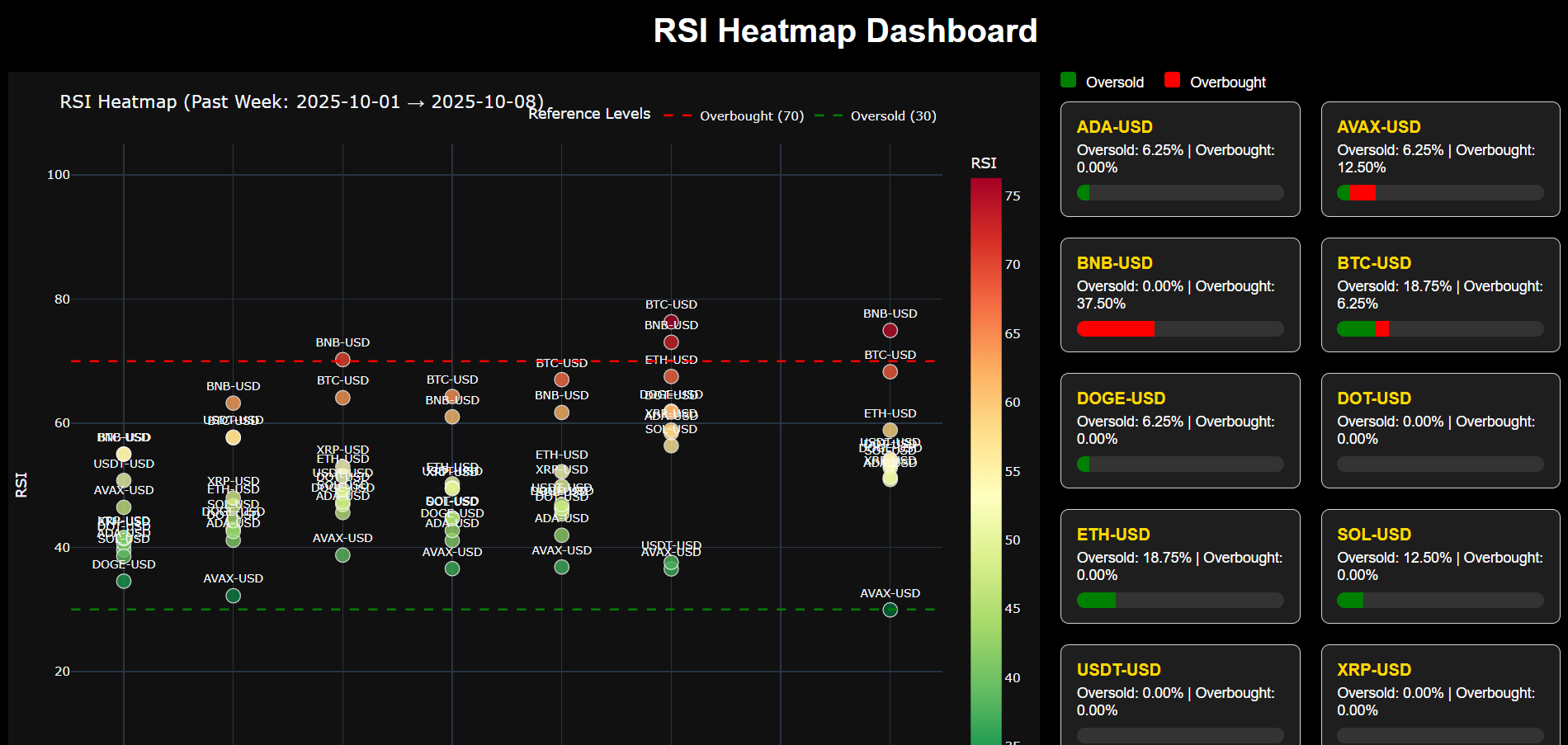
Here the users can view his p2p orders

## **Crypto News**



news data

## **RSI Chart**



RSI Heatmap

# **REFERENCES**

[1]. <https://ranaroussi.github.io/yfinance/>

[2].<https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.pct_change.html>

[4].<https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.diff.html>

[5].<https://corporatefinanceinstitute.com/resources/career-map/sell-side/capital-markets/exponentially-weighted-moving-average-ewma/>

[6].<https://www.investopedia.com/terms/r/rsi.asp>

[7].<https://docs.confluent.io/kafka-clients/python/current/overview.html>

[8].<https://coinmarketcap.com/>

[9].<https://simpleswap.io/>

[10.<https://huggingface.co/ElKulako/cryptobert>

[3].<https://blog.quantinsti.com/indicators-build-trend-following-strategy/?utm_source=google&utm_medium=cpc&utm_campaign=epat_reach_tofu_dsa_india&utm_keyword=&utm_content=&gad_source=1&gad_campaignid=22129587094&gbraid=0AAAAAD2T4TBSKuW7PNzGoNwuH-Oi7m9b1&gclid=Cj0KCQjw9JLHBhC-ARIsAK4PhcqZ2L30UHttpXogPf7WPRM4PO0aEZWWQoUrZkUBLGMhaSZOTDbhUW8aAs-1EALw_wcB>