

CRYPTONEX

PROJECT SUPERVISOR

Mr. Shoaib Raza

PROJECT CO-SUPERVISOR

Mr. Supervisor Name

PROJECT TEAM

Mohammad Sharjeel Ghauri K180398

Ali Aman Charolia K181091

Muhammad Sualeh Alam K180216

Submitted in partial fulfilment of the requirement for the degree of Bachelors of Science in Computer Science

Submitted on: June 7th 2022

FAST SCHOOL OF COMPUTING

NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

KARACHI CAMPUS

JUNE 2022

|  |  |
| --- | --- |
| Project Supervisor | Mr. Shoaib Raza |
| Project Team | Mohammad Sharjeel Ghauri K180398  Ali Aman Charolia K181091  Muhammad Sualeh Alam K180216 |
| Submission Date | June 7th ,2022 |

Supervisor

**Mr. Shoaib Raza ------------------------------------**

Head Of Department

**Dr. Zulfiqar Ali Memon ------------------------------------**

FAST SCHOOL OF COMPUTING

NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

KARACHI CAMPUS

# 

# **ACKNOWLEDGEMENT**

We would like to mention that our project would not have reached its maximum potential without our supervisor Mr. Shoaib Raza who has been a source of constant support nudging us to improve our work. Moreover, guiding us and correcting our path whenever we deviated from the goal encouraged us to work hard.

We are also thankful to our parents for supporting us financially as well as emotionally in this tough journey at FAST NUCES. Without their moral support, this journey would have been impossible.

Furthermore, we would like to express our deepest appreciation to our mentor Dr. Farrukh Hassan for guiding us and giving us useful tips to mould our project perfectly.

# **ABSTRACT**

We all have a responsibility to be efficient stewards of resources. If you are adding value and helping other people in some way, then money should be flowing into your life with ease. Cryptocurrencies are known for being highly turbulent and volatile, with prices fluctuating dramatically even in the space of seconds that might not be noticed by any trader. A human trader may not be able to monitor the prices and carry out his trades 24/7. Most non-technical people are not aware of when to enter the market thus our project makes this process simplified. People might get profit or loss both by surprise via emotional trading. Combined, these factors limit the effectiveness of human traders in several ways. A Crypto trading bot is software that helps you automate your trading strategies. Cryptonex has been proposed where it will be available to people in the form of web and android apps. The basic working of our crypto trading bots is the automated execution of orders based on your desired trading strategy at your crypto exchange without your manual intervention using Binance API. This reduces risks and gives you an extra edge over manual trading ensuring user satisfaction.

**Table Of Contents**

[ACKNOWLEDGEMENT 3](#_Toc104768295)

[ABSTRACT 4](#_Toc104768296)

[Document Information 9](#_Toc104768297)

[INTRODUCTION 10](#_Toc104768298)

[PROBLEM STATEMENT 10](#_Toc104768299)

[NEED FOR THE PRODUCT 11](#_Toc104768300)

[Requirement Analysis 12](#_Toc104768301)

[Project Scope 12](#_Toc104768302)

[Not In Scope 13](#_Toc104768303)

[Project Objectives 13](#_Toc104768304)

[1. Stakeholders 14](#_Toc104768305)

[1.1. Not technical stakeholders 14](#_Toc104768306)

[1.2. Operating Environment 14](#_Toc104768307)

[1.3. System Constraints 14](#_Toc104768308)

[1.4. Assumptions and Dependencies 14](#_Toc104768309)

[2. External Interface Requirements 15](#_Toc104768310)

[2.1. Hardware Interfaces 15](#_Toc104768311)

[2.2. Software Interfaces 15](#_Toc104768312)

[2.3. Communications Interfaces 15](#_Toc104768313)

[2.3.1. System Feature 1 15](#_Toc104768319)

[2.3.1.1. Stimulus/Response Sequences 15](#_Toc104768320)

[2.3.1.2. Functional Requirements 15](#_Toc104768321)

[2.3.2. System Feature 2 16](#_Toc104768322)

[2.3.3. System Feature 3 16](#_Toc104768323)

[2.3.4. System Feature 4 17](#_Toc104768324)

[3. Use Cases 18](#_Toc104768325)

[4. Non-functional Requirements 22](#_Toc104768329)

[4.1. Performance Requirements 22](#_Toc104768330)

[4.2. Safety Requirements 22](#_Toc104768331)

[4.3. Security Requirements 22](#_Toc104768332)

[4.4. User Documentation 22](#_Toc104768333)

[5. Design Details: 23](#_Toc104768334)

[5.2. Design Considerations 23](#_Toc104768335)

[5.2.1. Assumptions and Dependencies 23](#_Toc104768336)

[5.2.2. Risks and Volatile Areas 23](#_Toc104768337)

[7. System Architecture 24](#_Toc104768338)

[7.1. System Level Architecture 24](#_Toc104768339)

[7.2. Software Architecture 25](#_Toc104768340)

[7.3. Design Strategy 26](#_Toc104768341)

[7.4. Detailed System Design 27](#_Toc104768342)

[8. Database Design 28](#_Toc104768343)

[8.1. ER Diagram 28](#_Toc104768344)

[9. Data Dictionary 29](#_Toc104768345)

[9.1. User Credentials 29](#_Toc104768346)

[9.2. Binance 31](#_Toc104768347)

[9.3. Coin 32](#_Toc104768348)

[9.4. Trading Bot 33](#_Toc104768349)

[9.5. Order 34](#_Toc104768350)

[10. Application Design 35](#_Toc104768351)

[10.1. Sequence Diagram 35](#_Toc104768352)

[10.2. State Diagram 36](#_Toc104768353)

[11. Implementation Details 37](#_Toc104768354)

[11.1. System architecture 37](#_Toc104768355)

[11.2. Development Tools/Software Used 38](#_Toc104768356)

[11.3. User Interface 39](#_Toc104768357)

[11.4. Web Application 43](#_Toc104768358)

[12. Test Plans 47](#_Toc104768359)

[12.1. Purpose of Test Plan 47](#_Toc104768360)

[12.2. Environment Needs 47](#_Toc104768361)

[12.3. Validation Testing 47](#_Toc104768362)

[12.4. Test Approach(s) 47](#_Toc104768363)

[12.5. Test Pass/Fail Criteria 48](#_Toc104768364)

[12.6. Non-Functional Testing 48](#_Toc104768365)

[12.7. Functional Testing 49](#_Toc104768366)

[13. Project Code 52](#_Toc104768367)

[13.1. Frontend Code: 52](#_Toc104768368)

[13.2. Backend Code: 54](#_Toc104768369)

[14. Limitation of System 56](#_Toc104768370)

[15. Future Work 56](#_Toc104768371)

[16. Conclusion 57](#_Toc104768372)

[17. References 58](#_Toc104768373)

**LIST OF FIGURES**

[Figure 1 Registration 18](#_Toc104763142)

[Figure 2:Login 19](#_Toc104763143)

[Figure 3:Order 20](#_Toc104763144)

[Figure 4:Order History 21](#_Toc104763145)

[Figure 5:System Architecture Diagram 24](#_Toc104763146)

[Figure 6 Software Architecture Diagram 25](#_Toc104763147)

[Figure 7:Class diagram 27](#_Toc104763148)

[Figure 8:ER Diagram 28](#_Toc104763149)

[Figure 9:Sequence Diagram 35](#_Toc104763150)

[Figure 10:Trading Bot State Diagram 36](#_Toc104763151)

[Figure 11:System Architecture (Implementation) 37](#_Toc104763152)

[Figure 12:Left (Dark Mode) Right (Light Mode): Main Dashboard of Cryptonex App 39](#_Toc104763153)

[Figure 13:Left (Dark Mode) Right (Light Mode): Recent Coins Statistics 40](#_Toc104763154)

[Figure 14:Left (Dark Mode) Right (Light Mode): Selected Coin Analytics with Crypto News 41](#_Toc104763155)

[Figure 15:Left (Dark Mode) Right (Light Mode): Coin Rates with Real Time Price Fluctuation 42](#_Toc104763156)

[Figure 16: Login Page 43](#_Toc104763157)

[Figure 17:Registration Page 43](#_Toc104763158)

[Figure 18:(Dark & Light Mode): Main Dashboard on web. 44](#_Toc104763159)

[Figure 19:(Dark & Light Mode): Bot settings 45](#_Toc104763160)

[Figure 20:(Dark & Light Mode): Trending coins and their statistics 46](#_Toc104763161)

[Figure 21:Frontend Dashboard 52](#_Toc104763162)

[Figure 22:Frontend Main 52](#_Toc104763163)

[Figure 23:Frontend Routes 53](#_Toc104763164)

[Figure 24:Frontend Splash 53](#_Toc104763165)

[Figure 25:Firebase API 54](#_Toc104763166)

[Figure 26:Main 54](#_Toc104763167)

[Figure 27:Trader Model 55](#_Toc104763168)

[Figure 28:User Model 55](#_Toc104763169)

**List Of Tables**

[Table 1:Registeration 18](#_Toc104763352)

[Table 2:Login 19](#_Toc104763353)

[Table 3:Order 20](#_Toc104763354)

[Table 4:Order History 21](#_Toc104763355)

[Table 5:Data Dictionary: User Credential 30](#_Toc104763356)

[Table 6:Data Dictionary: Binance 31](#_Toc104763357)

[Table 7:Data Dictionary: Coin 32](#_Toc104763358)

[Table 8:Data Dictionary: Trading Bot 33](#_Toc104763359)

[Table 9:Data Dictionary: Order 34](#_Toc104763360)

[Table 10:Items to be Tested/Not Tested in Non-Functional Testing 47](#_Toc104763361)

[Table 11:Test Case 1 48](#_Toc104763362)

[Table 12:Test Case 2 48](#_Toc104763363)

[Table 13:Item to be Tested/Not Tested in Functional Testing 49](#_Toc104763364)

[Table 14:Test Case 1 49](#_Toc104763365)

[Table 15:Test Case 2 49](#_Toc104763366)

[Table 16:Test Case 3 50](#_Toc104763367)

[Table 17:Test Case 4 50](#_Toc104763368)

[Table 18:Test Case 5 50](#_Toc104763369)

[Table 19:Test Case 6 50](#_Toc104763370)

[Table 20:Test Case 7 51](#_Toc104763371)

# **Document Information**

|  |  |
| --- | --- |
| **Category** | **Information** |
| Customer | FAST-NUECS KHI |
| Project Title | CRYPTONEX |
| Document Version | 1.0 |
| Author(s) | M.Sharjeel Ghauri,Ali Aman,Sualeh Alam |
| Approves(s) | Mr.Shoaib Raza |

**Definition of term, Acronyms and Abbreviations**

|  |  |
| --- | --- |
| **Terms** | **Description** |
| API | Application Programming Interface |
| RSI | Relative Strength Index |
| CS | Computer Science |
| APP | Application Portability Profile |
| MAC | Macintosh computer |
| SDS | Software Design Specification |
| SRS | Software Requirement Specification |

# **INTRODUCTION**

The rise of cryptocurrency trading had been growing for the last few years. People use simple methods of a signal which are from other traders on different social media platforms or used emotional trading to earn with was not a legitimate method. It could lead to loss and severe loss could lead to loss of life due to immense depression caused by the loss of all the money during trading. Thus, we decided to make a method for smart trading and that is to automate the trading process. This leads to less time consumption and efficient smart trading with fewer chances of loss. Using signals from another person can be risky and less reliable so this automation process will generate signals based on mathematical calculation with the usage of real-time data from the Binance API. Binance is the largest crypto trading platform used by traders to trade cryptocurrency. This project has a significant academic value as we will be touching on financial accounting and algorithmic trading using AI which is a deep pool that needs to be understood to carry out the process of creating this project. If this project is successful and is taken to an extraordinary level of accuracy then we could earn millions of dollars from this project. We could also sell this project to venture capitalists and could get a handsome amount of investment for this project. The technical indicators used for this project are extremely difficult to understand. Professional traders use these indicators for their trading process but they do not know the backend of these indicators. For us to automate this trading process we need to have a deep knowledge of these technical indicators. Moreover, every currency has a different indicator and works very differently form one another so handling different currencies during this process can be extremely difficult for us.

# **PROBLEM STATEMENT**

The Crypto market has been volatile since the beginning of the crypto era and people have been facing a huge number of losses financially. Humans may not be able to monitor the prices and carry out their trades all day as there is resting time as well at night. Usually, crypto market flips overnight. Traders are unable to monitor the change which took place overnight. They might get profit or loss both by surprise. Investors also have the opportunity to take part in cryptocurrency trading around the world and at any hour of the day. Combined, these factors limit the effectiveness of human cryptocurrency trading in several ways. Furthermore, emotional trading is also a great factor that causes losses to people. According to sources recently in 2022 Luna coin market crashed from 120 US dollars to 0.010 US dollars. Due to this eight people suicide. Thus, high fluctuation in the crypto market has also resulted in the loss of life for many years.

# **NEED FOR THE PRODUCT**

The process of trading is entering the market at a low price and exiting at a high price thus generating the maximum amount of profit. Most of the non-technical people are not aware of when to enter the market thus our project makes this process simplified. We will be using AI-powered algorithmic trading and technical indicators data from the Binance application to enhance the process of entering and exiting from the market vice versa buying and selling leading to enter the market at the profitable point. More problems that are being catered to by this process is that people use emotions to trade which can be disastrous. Our project is based on emotionless trading and eliminates any emotional trading errors and works more on numbers and data which is used by the algorithms. Hence, this is an efficient trading approach. Manual trading is automated saving time and effort. We will be achieving our objective by developing an automated trading bot in Python that will process real-time data and will make predictions based on different machine learning, technical indicators, and financial algorithms. For processing real-time data, we will be using a **Binance** API. The Binance API is a method that allows you to connect to the Binance servers via Python or several other programming languages. With it, you can automate your trading. Our main goal in this project is to make a decent profit by running a set of algorithms, with which we can automatically buy, sell or hold assets in a timely, efficient, and automated manner day or night from anywhere in the world.

# 

# **Requirement Analysis**

## **Project Scope**

The core idea of this project is to develop an automated system for trading crypto currency which a non-technical person can use easily and can generate profit. The process of the trading is entering the market at a low price and exiting at a high price thus generating maximum amount of profit. People use simple methods of signal which are from other trader on different social media platforms or used emotional trading to earn with was not a legitimate method. It could lead to loss and severe loss could lead to loss of life due immense depression caused by the loss of all the money during trading. Thus, we decided to make a method for smart trading and that is to automate the trading process. This leads to less time consumption and efficient smart trading with less chances of loss. Using signals from other person can be risky and less reliable so this automation process will generate signals on the basis of mathematical calculation with usage of real time data from the Binance API. Binance is the largest crypto trading platform used by the traders to trade crypto currency. This project has a significant academic value as we will be touching financial accounting and algorithmic trading using AI which is a deep pool that needs to be understood to carry out the process of creating this project. The technical indicators (such as RSI) used for this project are extremely difficult to understand. Professional traders use these indicators for their trading process but they do not know the backend of these indicators. For us to automate this trading process we need to have a deep knowledge of these technical indicators. More over every currency has a different indicator and work very differently form one another so handling different currencies during this process can be extremely difficult for us. Some of the features of our product are market Scouting/Tracking for updates in prices, user notification via Email (Gmail), user notification via WhatsApp message and placing of order to buy and sell you owned currency as well as setting the limit of profit the user needs.

## 

## **Not In Scope**

Our product is based on spot trading but future trading is also possible. Bitcoin futures track the price movements of the world’s largest digital asset, Bitcoin. It allows investors to gain exposure to Bitcoin without having to hold the underlying cryptocurrency. Like traditional futures contracts, Bitcoin futures are legal contracts to buy or sell Bitcoin at a future date. Bitcoin futures offer protection against volatility and adverse price movements. Also, it is a proxy tool for traders to speculate on the future prices of Bitcoin. With a Bitcoin futures contract, you can take a long position if you expect the price of Bitcoin to rise. Conversely, you take a short position to reduce the impact of losses when the price of Bitcoin falls. When investing for the long-term, Bitcoin may experience occasional bear markets as sentiments change. In these uncertain periods, your Bitcoin portfolio is unlikely to gain very much, if any at all. In such scenarios, Bitcoin futures contracts can be useful to protect your Bitcoin investments against downside risk.

## **Project Objectives**

The process of trading is entering the market at a low price and exiting at a high price thus generating the maximum amount of profit. Most the non-technical people are not aware of when to enter the market thus our project makes this process simplified. We will be using AI-powered algorithmic trading and technical indicators data from the Binance application to enhance the process of entering and exiting from market vice versa buying and selling leading to entering the market at the profitable point. More problems that are being catered to by this process is that people use emotions to trade which can be disastrous. Our project is based on emotionless trading and eliminates any emotional trading errors and works more on numbers and data which is used by the algorithms. Hence, this is an efficient trading approach. Manual trading is automated which benefits users by saving their time and efforts. We will be achieving our objective by developing an automated trading bot in Python that will process real-time data and will make predictions based on different machine learning, technical indicators, and financial algorithms. For processing real-time data, we will be using a Binance API. The Binance API is a method that allows you to connect to the Binance servers via Python or several other programming languages. With it, you can automate your trading. Our main goal in this project is to make a decent profit by running a set of algorithms, with which we can automatically buy, sell or hold assets in a timely, efficient, and automated manner day or night from anywhere in the world.

# **Stakeholders**

## **Not technical stakeholders**

Naïve users

Users with a limited knowledge of crypto currency

Investors

Technical Stakeholders

Developers

**Payment / Exchange Services (Marketplace)**

## **Operating Environment**

The software can be operated on windows via web application and android smartphone via android application. Hardware requirement would be a stable connection of internet and a windows pc and android smartphone with adequate and latest hardware components such as fast internet, 3 GB ram (Android 6.0 Marshmallow or higher, a 64-bit processor (any new-generation processor from Qualcomm, MediaTek, Exynos, or Unisoc), and at least 2GB of ram) for smartphones application, not outdated components.

## **System Constraints**

* Software constraints

Internet connectivity 24/7

* Hardware constraints

Android Smartphone.

* Cultural constraints (includes language etc.)

English

Urdu

(Both languages will be supported in the smartphone application)

* Legal constraints

Few legal constraints such as keeping the data protected from unauthorized access and not selling it to third parties. All the data should be kept confidential.

## **Assumptions and Dependencies**

Our project is very diverse and it can be implemented on cloud service as well that will be running our server 24/7 but due to lack of resources such as financial assistance as cloud services is expensive, we will be running our project on local server and demonstrating it in final evaluation.

# **External Interface Requirements**

## **Hardware Interfaces**

Specifically, there are no devices except laptops, mobiles and (PCs with at least 8 GB ram for our server) and a PC which one can access our website and mobile application.

## **Software Interfaces**

We will be using databases such as firebase. Our product has different components such as main server running on python, web and mobile application built on dart language. Our final product will be a flutter application with web and mobile interfaces.

## **Communications Interfaces**

Our customer can communicate with us via E- Mail and customer care helpline. Feature includes login, searching coin, payment of the coin by the user and additional feature of the golden customer which will have predictions of the coin etc.



## **System Feature 1**

**LOGIN:**

#### Description and Priority

It is our most important priority for customers to login or either signup because they won’t be able to trade without filling details including their location and NIC.

## **Stimulus/Response Sequences**

First customer will visit our application with domain (name) for web and app name for smartphones which will be provided by us then he will choose the coin he wants to buy or sell by another search feature after selection of coin when he clicks on “buy or sell” button he will have popup of login and then he will manage to buy or sell the coin.

## **Functional Requirements**

When customer want to login on our website his details will be verified by our data base, then he will receive a verification code via email or contact number and then he can manage to login.

REQ-1: Verification of customer details on our database

## **System Feature 2**

**SEARCHING:**

#### Description and Priority

It is not important priority because it depends upon customer choice customer might be able to find his desire coin on our homepage of application.

#### Stimulus/Response Sequences

First customer will visit our application with domain (name) for web and app name for smartphones which will be provided by us then if customer is not able to find his desired coin on our homepage customer needs to use our search option by entering name of coin he wants to buy and if it is available on our application then he can buy that coin or else not.

#### Functional Requirements

When customer use our search option our API will be called to retrieve name of coin that customer is looking for.

REQ-1: Name of the crypto coin in database table.

### System Feature 3

**PAYMENT:**

#### Description and Priority

It is our top priority because without coin payment customer won’t be able to buy the coin and neither can sell it.

#### Stimulus/Response Sequences

After customer login to his/her account then he can pay to buy his desired crypto coin by first filling detail and then customer can pay amount of money by means of credit card or cash by depositing it in his binance wallet. When the customer wills to sell the coin, he will again place an order and as soon as the coin is sold, he will receive the money in his binance wallet.

#### Functional Requirements

When customer orders to buy a crypto currency, his amount needed for order completion will be checked from his binance wallet through binance API.If the amount is sufficient enough to complete his order, then only, he can buy the crypto currency or else not.

REQ-1: Binance Wallet details.

### System Feature 4

**Prediction of Future Price for Golden Customer and much more:**

#### Description and Priority

It is important to reward the customers who are loyal to our application and the customers who have subscribed to our trading bot application.

#### Stimulus/Response Sequences

If a customers want to obtain the perks of a golden customer, he/she can subscribe our service by paying a monthly amount. Details of the subscription will be available online via option provided to them on our application.

#### Functional Requirements

Customer who has subscription to our service can only view our cryptocurrency predictor as well as other features used by professional traders to carryout safe trades with minimized losses.

REQ-1: Customer should subscribe our service to get those additional perks.

# **Use Cases**

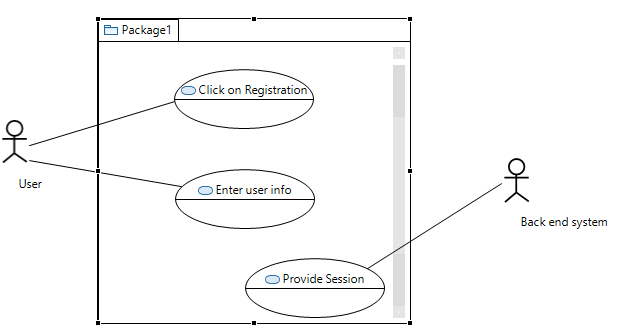


Figure 1 Registration

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| <Use case 1: Registration> | | | | | |
| Use case Id: | | 1.1 | | | |
| Primary Actors: Customer | | | | Primary Actors: System | |
| Feature: Administrator | | | | | |
| Pre-condition: | | The user must have the Application opened. User credential must be verified such as E-mail and verification code | | | |
| Scenarios | | | | | |
| Step# | Action | | | | Software Reaction |
| 1. | Enter binance API KEY | | | | Verify API key |
| 2. | Enter Email | | | | Verify email |
| 3. | Enter verification code | | | | Dual authentication gives access the user to proceed with his registration |
| Alternate Scenarios | | | | | |
| 1a: If an information field is not entered the user is prompted to enter it. | | | | | |
| Post Conditions | | | | | |
| Step# | Description | | | | |
| 1 | The user will be registered and his information and credentials will be saved in the database. Now the user will be able to log. | | | | |
| Use Case Cross referenced | | | Login | | |

Table 1:Registeration

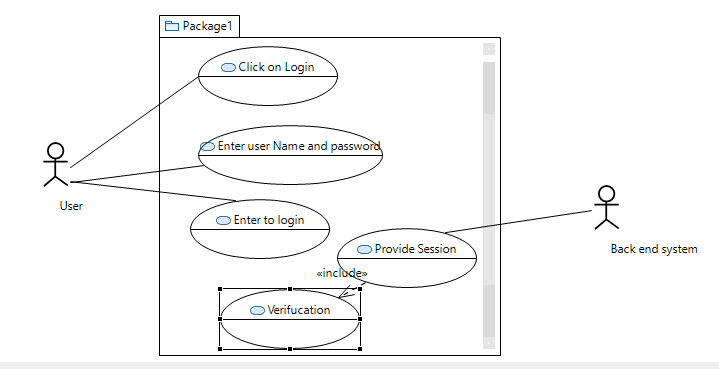




Figure 2:Login

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| <Use case 2: Login> | | | | | |
| Use case Id: | | 1.2 | | | |
| Primary Actor: Customer | | | | Other Actor: Backend System | |
| Feature: Administrator | | | | | |
| Pre-condition: | | User must be registered | | | |
| Scenarios | | | | | |
| Step# | Action | | | | Software Reaction |
| 1. | The user presses the login option. | | | | System takes input |
| 2. | The user is prompted to enter username. | | | | System takes input |
| 3. | The user is prompted to enter password | | | | System takes input |
| 4. | The user is prompted to enter verification code received via email/contact number | | | | System sends verification code to user via email/Contact |
| 5. | Press the login button. | | | | System verifies user inputs |
| 6. | Authentication is done from the database. | | | | If the inputs by user are verified then login successfully otherwise system asks user enter valid inputs. |
| Alternate Scenarios | | | | | |
| 1a: If the id password is not verified the user is prompted about the wrong id password. | | | | | |
| Post Conditions The user will be logged in. | | | | | |
| Step# | Description | | | | |
| 1. | Now user can place orders to buy and sell and view information of every coin available | | | | |
| Use Case Cross referenced | | | Place orders to buy and sell his assets | | |

Table 2:Login

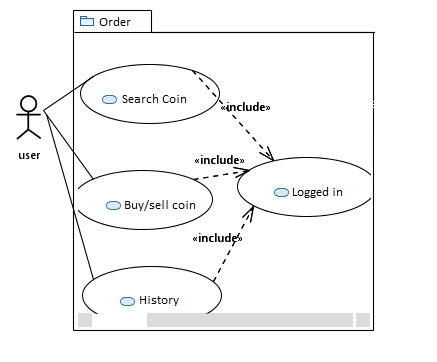


Figure 3:Order

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| <Use case 3: Order> | | | | | |
| Use case Id: | | 1.3 | | | |
| Primary Actor: Customer | | | | Other Actor: Backend System | |
| Feature: Administrator | | | | | |
| Pre-condition: | | The user must be registered, logged in, then only the order can be placed | | | |
| Scenarios | | | | | |
| Step# | Action | | | | Software Reaction |
| 1. | User view available coins | | | | System will list down the available coins |
| 2. | User can check coin status (bearish/bullish | | | | System will show Coin trends |
| 3. | User place order to buy/sell coin | | | | System will buy/sell coin if user wants on the basis of limits set |
| 4. | User can check his order history | | | | System will show user order history |
| Alternate Scenarios | | | | | |
| 1a: No Alternatives | | | | | |
| Post Conditions | | | | | |
| Step# | Description | | | | |
|  | User order history will be saved in database and now user owns the coin in case of buy or the amount in dollars if he sells the coin. | | | | |
| Use Case Cross referenced | | | Order History/logs | | |

Table 3:Order

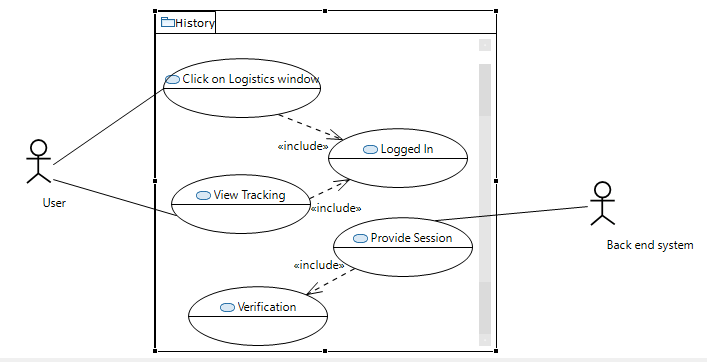


Figure 4:Order History

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| <Use case 4: Order Track> | | | | | |
| Use case Id: | | 1.4 | | | |
| Primary Actor: Customer | | | | Other Actor: Backend System | |
| Feature: Order | | | | | |
| Pre-condition: | | The user must be registered, logged in and must have placed an order | | | |
| Scenarios | | | | | |
| Step# | Action | | | | Software Reaction |
| 1. | User presses order history button | | | | User navigates to order history/logistics page |
| 2. | Check orders | | | | System will show user orders |
| Alternate Scenarios: | | | | | |
| 1a: No Alternatives | | | | | |
| Post Conditions | | | | | |
| Step# | Description | | | | |
|  | User can view his logs for all the order, his purchases as well as his sells with date and amount of purchase and sell. | | | | |
| Use Case Cross referenced | | | The user will be updated about their orders. | | |

Table 4:Order History

# **Non-functional Requirements**

## **Performance Requirements**

Our application will work perfectly in normal circumstances and the customer will be able to view our main onboarding screen as soon as they open the application with all the coin information. Although it will be a rare case that the application will crash due to an abundant number of users, Users will have to wait till the site is back on its usual flow.

## **Safety Requirements**

The safe order placement is guaranteed by our application that the trades will not be giving any sort of loss of resources to the user. The customer should not use our application if he is a patient with hypertension. Also, naïve users should trade with the recommended profit margin told by the application by default to protect themselves from losses.

## **Security Requirements**

We will be using user authentication to make our transactions safe and secure for users as well as we will be keeping all the secret keys secure and safe.

## **User Documentation**

Guided Assist will be turned on by default

User Manual

* Helpline
* Online help
* Password security
* Screenshot protection
* API Generation Instructions

# **Design Details:**

# **Design Considerations**

## **Assumptions and Dependencies**

We are assuming that the users on the CRYPTONEX platform already have a cryptocurrency Binance account with sufficient balance for trading. Secondly, we are currently assuming that Firebase database is capable of handling multiple users, but we are unsure about it and might change the Firebase database with Oracle SQL database for better handling of multiple users in future.

## **Risks and Volatile Areas**

There could be several known and unknown risks involved in this project. First, as the number of users will gradually increase, the load of the server needs to be managed efficiently to cater multiple users and to keep stability of the server. Secondly, a possible risk associated with this project could be the banning of cryptocurrency in the country of Pakistan can pose as a serious risk to our project. Lastly, Binance (universal platform for cryptocurrency trading) can also restrict Pakistan users in the future due to constant violation of Binance policies.

# **System Architecture**

## **System Level Architecture**

The following diagram provides the user with an overview of our project and describes the architecture of CRYPTONEX in more detail.

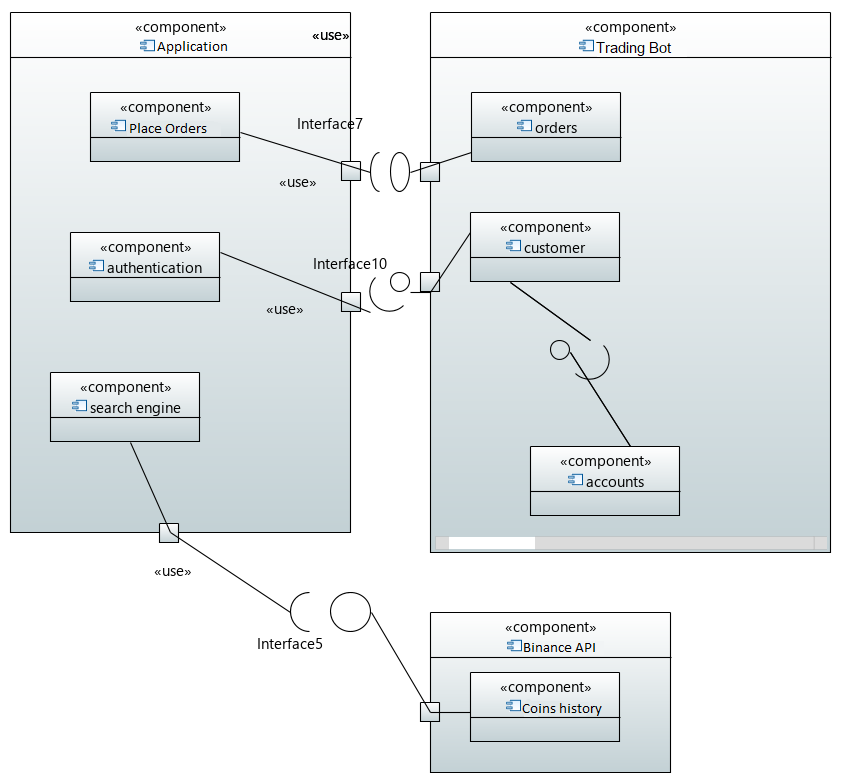


Figure 5:System Architecture Diagram

## **Software Architecture**

Graphical user interface

Description automatically generated

Figure 6 Software Architecture Diagram

## **Design Strategy**

First of all, we have designed our front-end using Flutter for mobile application and web application. After that, we will create our database using Firebase. After that we will work on the back-end server in which we will be using purely Python which includes the entire logic of our CRYPTONEX Trading Bot including buying, selling, real-time cryptocurrency price fetching, and much more. Finally, we have linked the front-end and back-end together to make our application fully functional.

This system is not only for one-time use, but it can also be used 24/7 continuously to buy latest available coins which are listed on Binance exchange platform. This application is dynamic and updates frequently after each interval. People who have a smartphone, a laptop, and a stable internet connection, can access, and take full advantage of this application. Data in our project is managed by Firebase database and this will also deal with concurrency of many users and synchronization.

## **Detailed System Design**

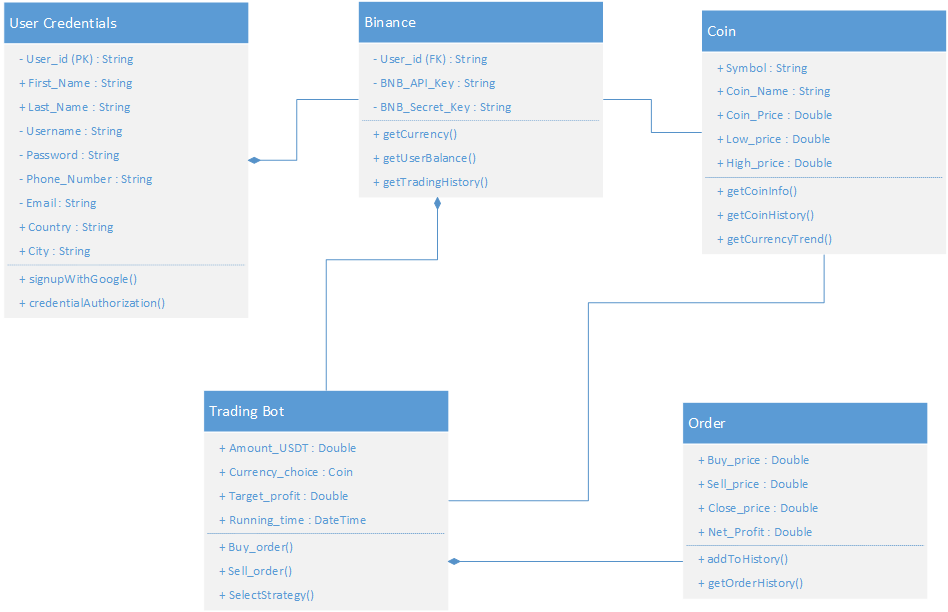


Figure 7:Class diagram

# **Database Design**

## **ER Diagram**

Diagram

Description automatically generated

Figure 8:ER Diagram

# **Data Dictionary**

## **User Credentials**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| <User Credentials> | | | | | | | |
| Name | User Credentials | | | | | | |
| Alias | - | | | | | | |
| Where-used/how-used | User Credentials class is used for storing user details. | | | | | | |
| Column  Name | | Description | Type | Length | Null able | Default Value | Key Type |
| User\_ID | | A unique user id to uniquely identify different users. | String | Varies depending on the users | No | None | PK |
| First\_Name | | User’s first name | String | 155 | No | None |  |
| Last\_Name | | User’s last name | String | 155 | Yes | None |  |
| Phone\_Number | | User’s phone number for sending messages | String | 155 | No | None |  |
| City | | User’s city when registration | String | 155 | Yes | None |  |
| Country | | User’s country at the time of registration | String | 155 | Yes | None |  |
| Username | | User’s username for registration | String | 155 | No | None |  |
| Email | | User’s email address | String | 155 | No | None |  |
| Password | | User’s password for registration | String | 155 | No | None |  |

Table 5:Data Dictionary: User Credential

## **Binance**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| <Binance> | | | | | | | |
| Name | Binance | | | | | | |
| Alias | - | | | | | | |
| Where-used/how-used | Binance class is used for fetching online data of cryptocurrencies and orders. | | | | | | |
| Column  Name | | Description | Type | Length | Null able | Default Value | Key Type |
| User\_ID | | A unique user id to uniquely identify different users. | String | 155 | No | None | FK |
| API\_KEY | | User’s generated Binance API Key for trading | String | 155 | No | None |  |
| API Secret\_KEY | | User’s generated Binance API Secret Key | String | 155 | No | None |  |

Table 6:Data Dictionary: Binance

## **Coin**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| <Coin> | | | | | | | |
| Name | Coin | | | | | | |
| Alias | - | | | | | | |
| Where-used/how-used | Coin class is used to store coin related information. | | | | | | |
| Column  Name | | Description | Type | Length | Null able | Default Value | Key Type |
| Symbol | | Symbol of the coin which user will select for trading | String | 155 | No | None |  |
| Coin\_Price | | Current price of coin | Double | 10 | No | None |  |
| High\_Price | | Highest price of the coin for a candle | Double | 10 | No | None |  |
| Low\_Price | | Lowest price of the coin for a candle | Double | 10 | No | None |  |
| Coin\_Name | | Full name of the coin for trading | String | 155 | No | None |  |

Table 7:Data Dictionary: Coin

## **Trading Bot**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| <Trading Bot> | | | | | | | |
| Name | Trading Bot | | | | | | |
| Alias | - | | | | | | |
| Where-used/how-used | Trading Bot class is used for automated buying, selling of cryptocurrency assets. | | | | | | |
| Column  Name | | Description | Type | Length | Null able | Default Value | Key Type |
| Amount\_USDT | | Total amount of money for which trading will be done. | Double | 10 | No | None |  |
| Currency\_Choice | | Selecting the crypto currency for trading. | Double | 10 | No | None |  |
| Target\_Profit | | This is the desired profit selected by the user for trading. | Double | 10 | Yes | None |  |
| Running\_Time | | Setting up the running time for the bot. | Integer | 10 | No | None |  |

Table 8:Data Dictionary: Trading Bot

## **Order**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| <Order> | | | | | | | |
| Name | Order | | | | | | |
| Alias | - | | | | | | |
| Where-used/how-used | Order class will be used to store data related to order information and will keep a track for all orders history. | | | | | | |
| Column  Name | | Description | Type | Length | Null able | Default Value | Key Type |
| Buy\_Price | | This will be the buying price at which the order is placed. | Double | 10 | No | None |  |
| Sell\_Price | | This will be the selling price at which the order is sold. | Double | 10 | No | None |  |
| Net\_Profit | | Profit generated for each order | Double | 10 | Yes | None |  |
| Low\_Price | | Lowest price of the coin for a candle | Double | 10 | Yes | None |  |

Table 9:Data Dictionary: Order

# **Application Design**

## **Sequence Diagram**

* + 1. User Sequence Diagram

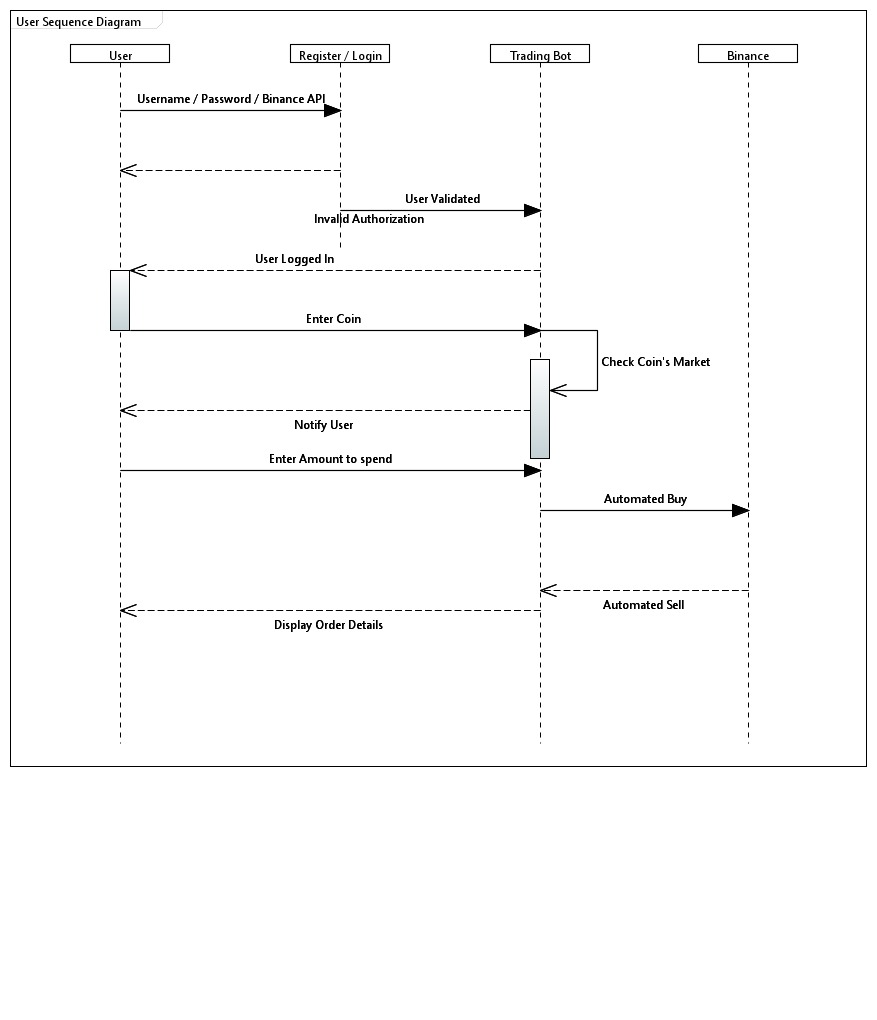


Figure 9:Sequence Diagram

## **State Diagram**

* + 1. Trading Bot State Diagram

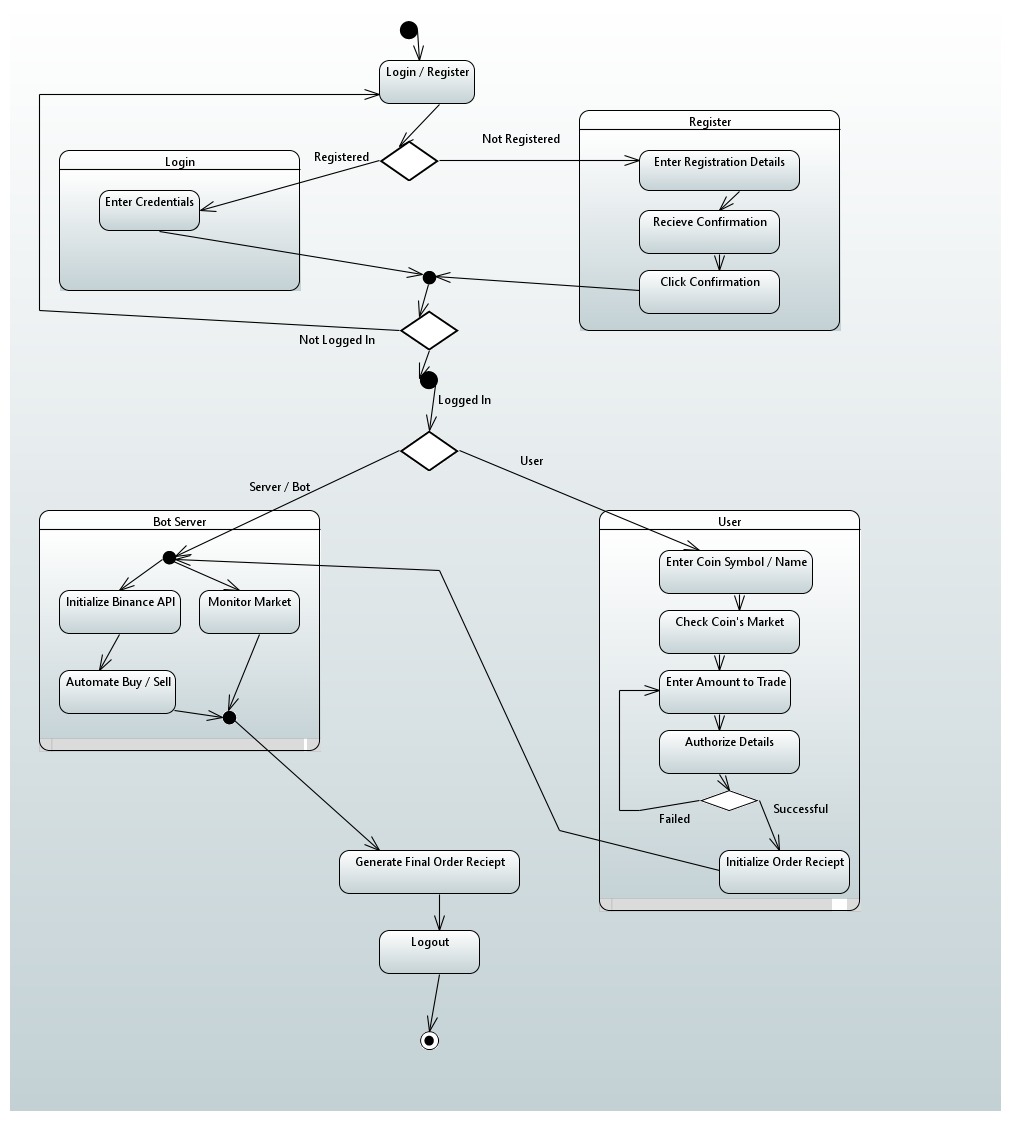


Figure 10:Trading Bot State Diagram

# **Implementation Details**

## **System architecture**

The figure shown below illustrates the architecture of the project "CRYPTONEX" android application and website build using flutter. Both applications depend on python server as the backend, paired with firebase as database.

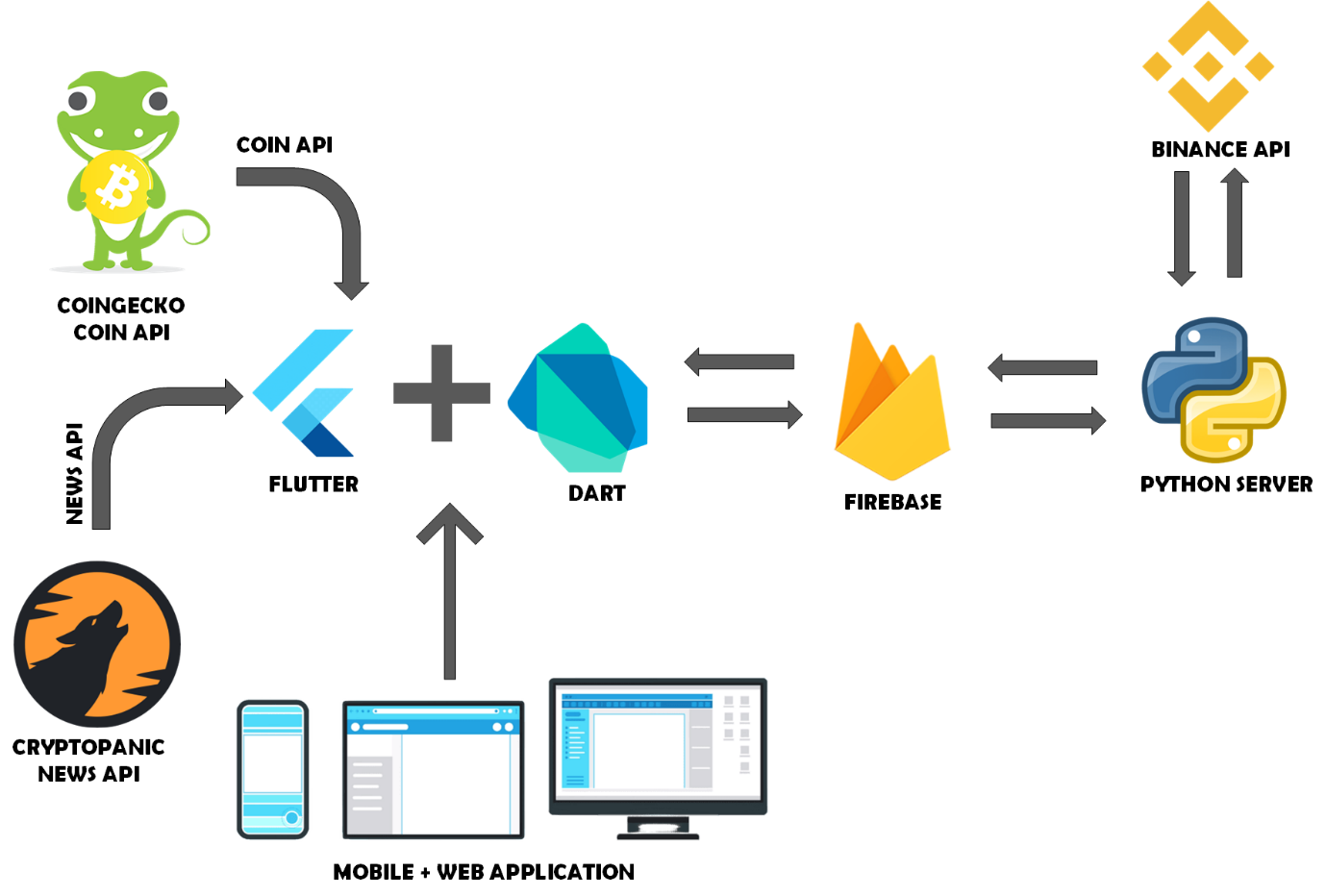


Figure 11:System Architecture (Implementation)

## **Development Tools/Software Used**

* VS Code
* Android studio
* PyCharm
* Firebase
* Python
* Dart
* Flutter
* Twilio

## **User Interface**

The team made sure to use a consistent theme in both the website and the android

application. Some of the examples are shown below.

* + 1. Android Application

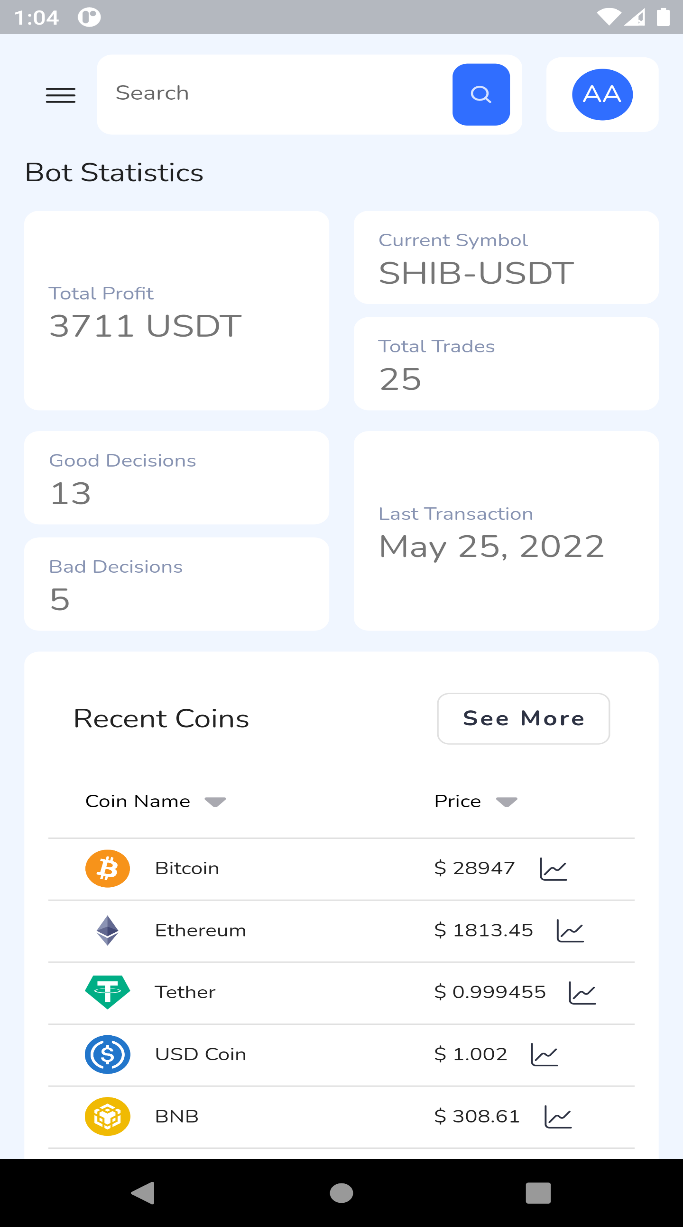
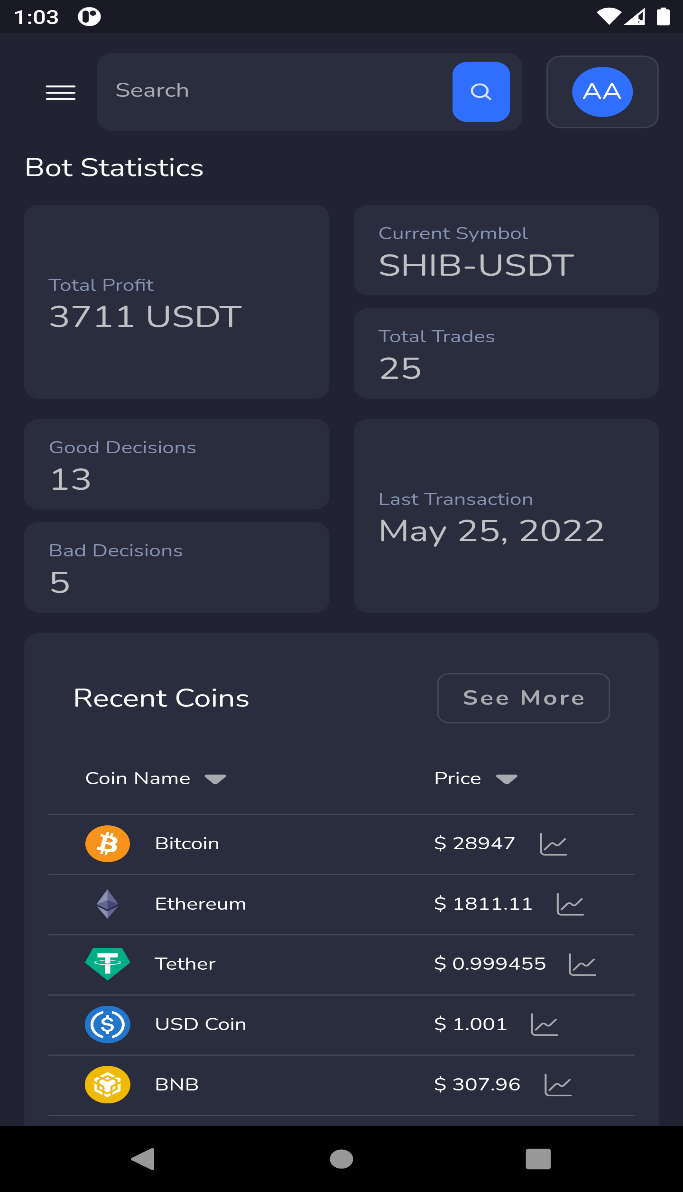


Figure 12:Left (Dark Mode) Right (Light Mode): Main Dashboard of Cryptonex App

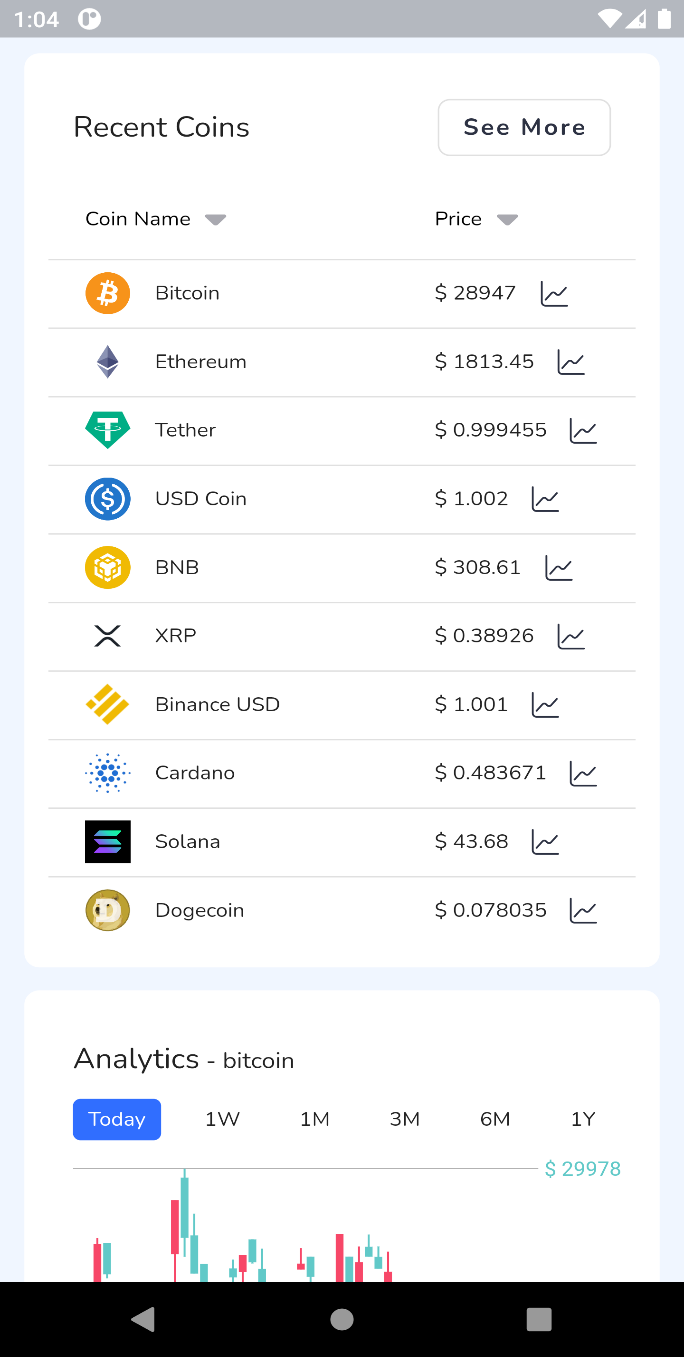
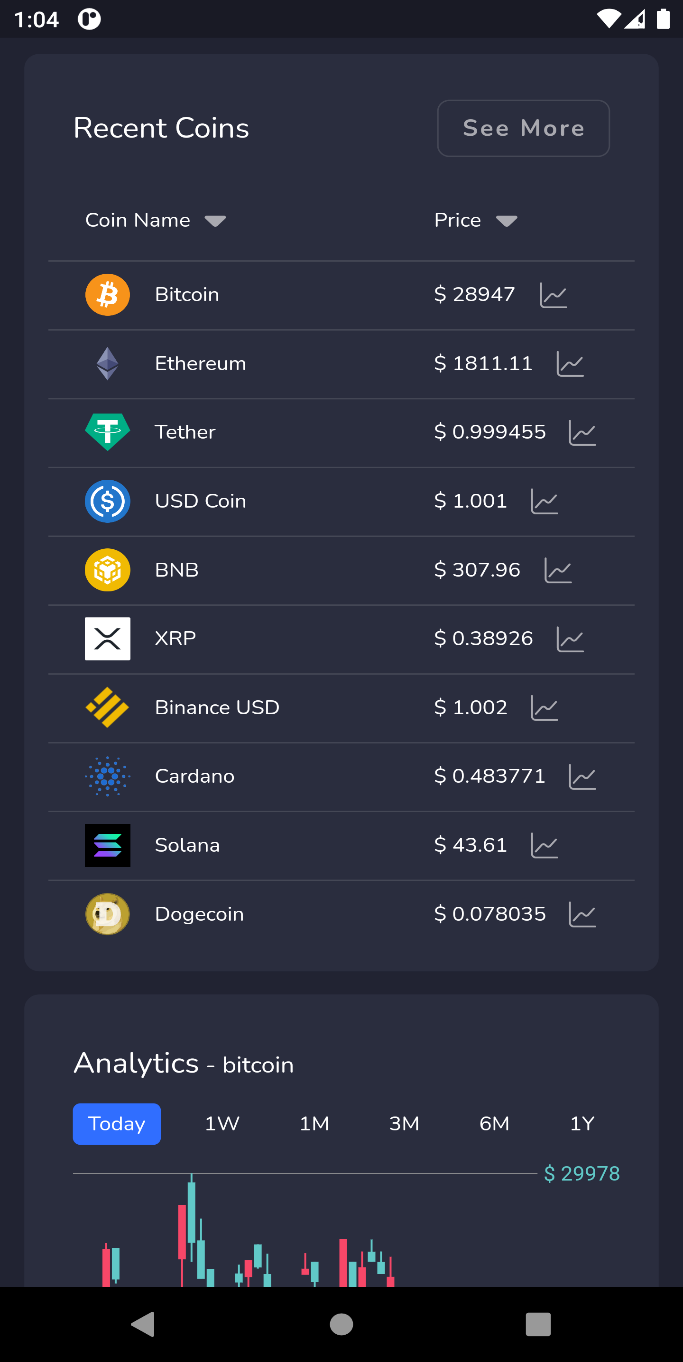


Figure 13:Left (Dark Mode) Right (Light Mode): Recent Coins Statistics

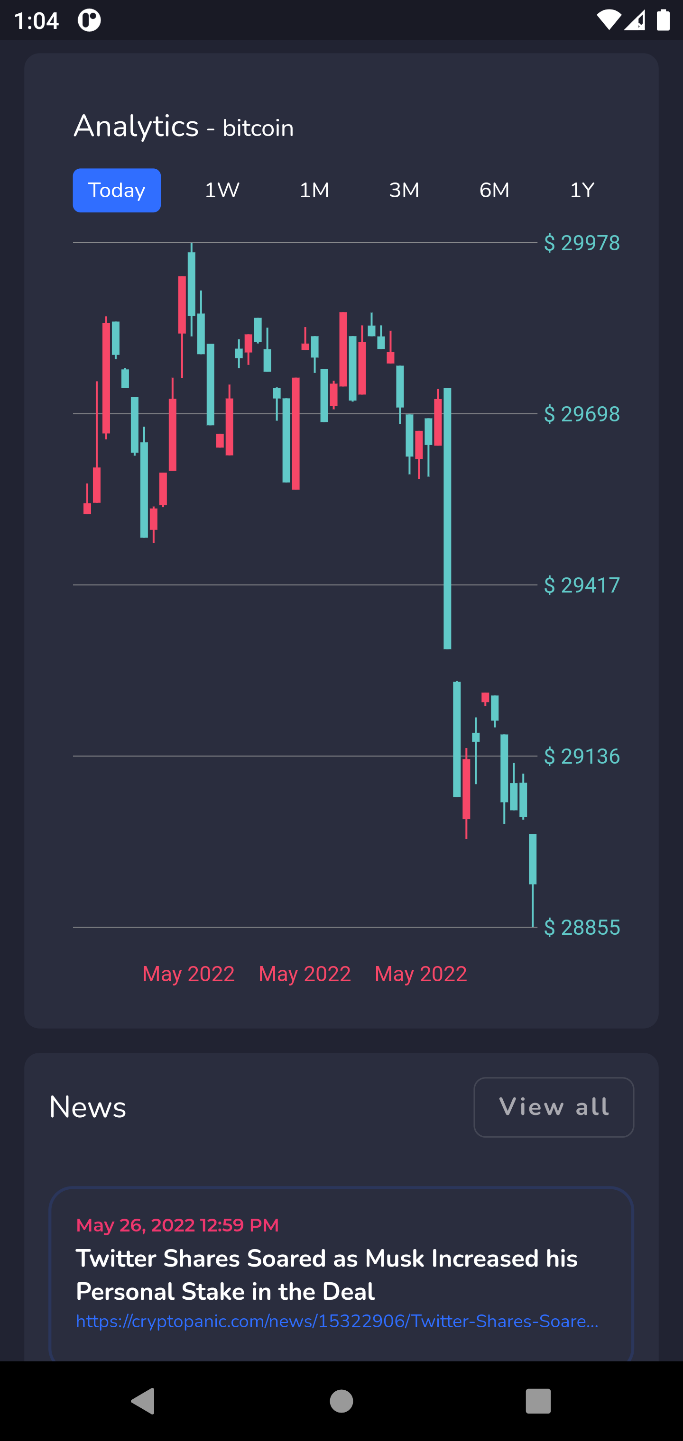


Figure 14:Left (Dark Mode) Right (Light Mode): Selected Coin Analytics with Crypto News

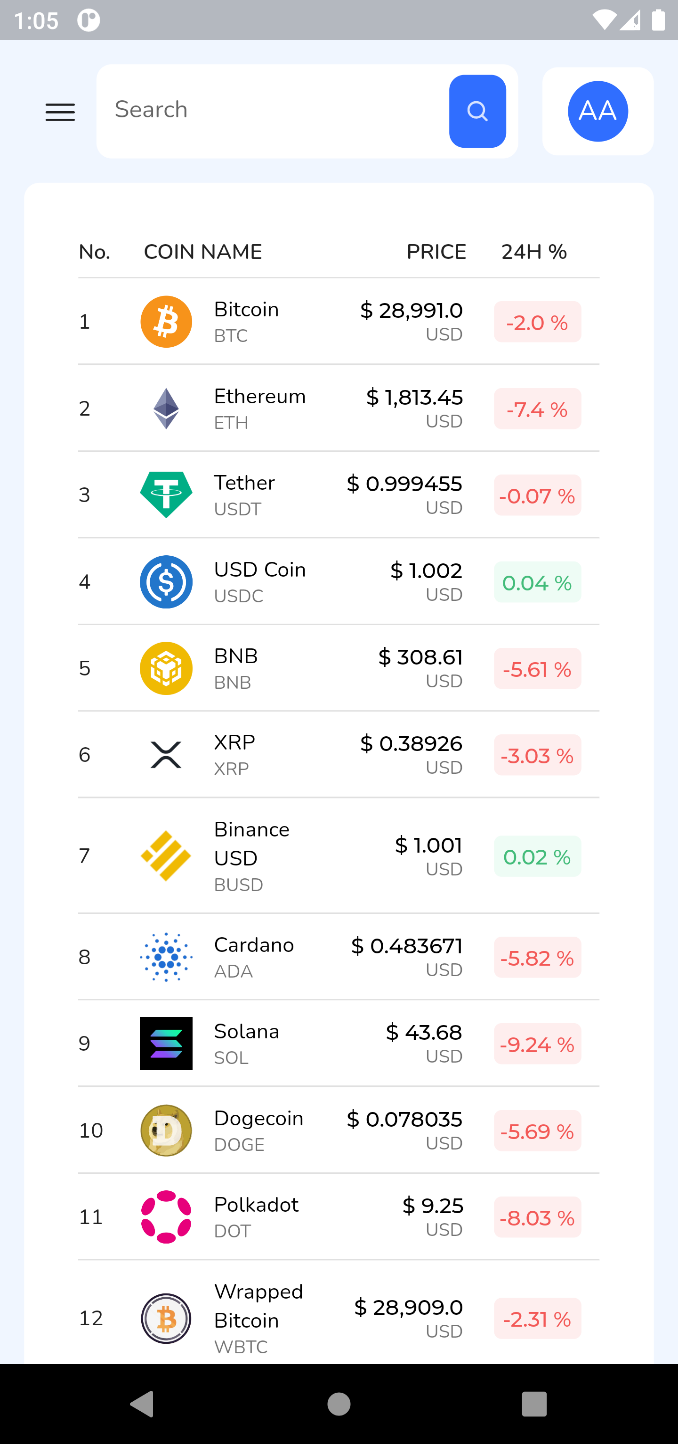
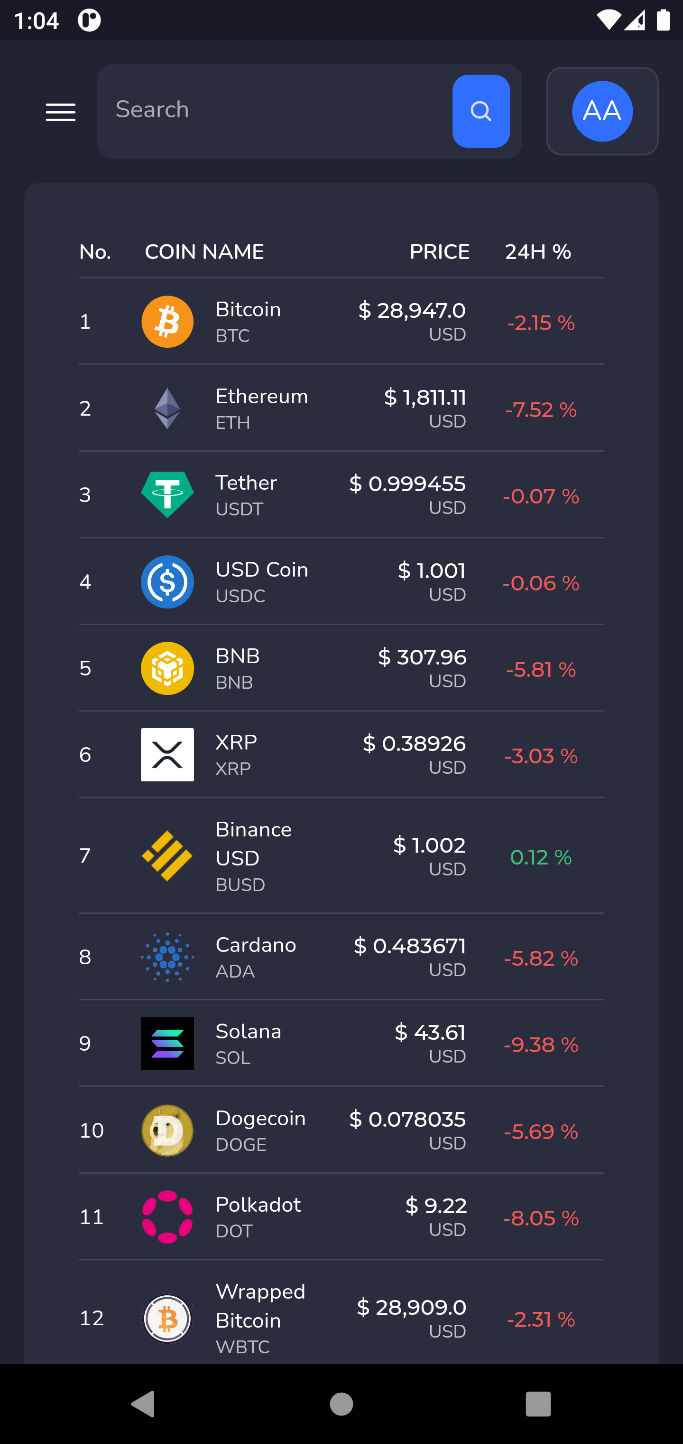


Figure 15:Left (Dark Mode) Right (Light Mode): Coin Rates with Real Time Price Fluctuation

## **Web Application**

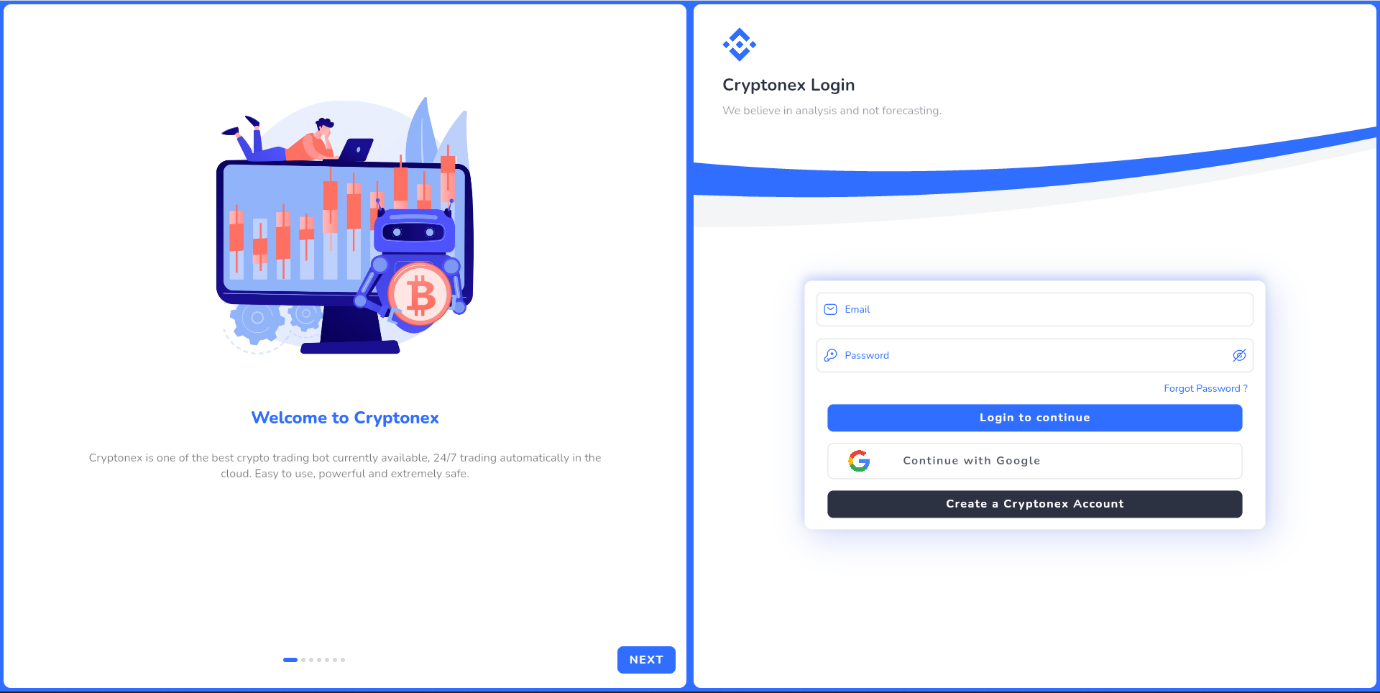


Figure 16: Login Page

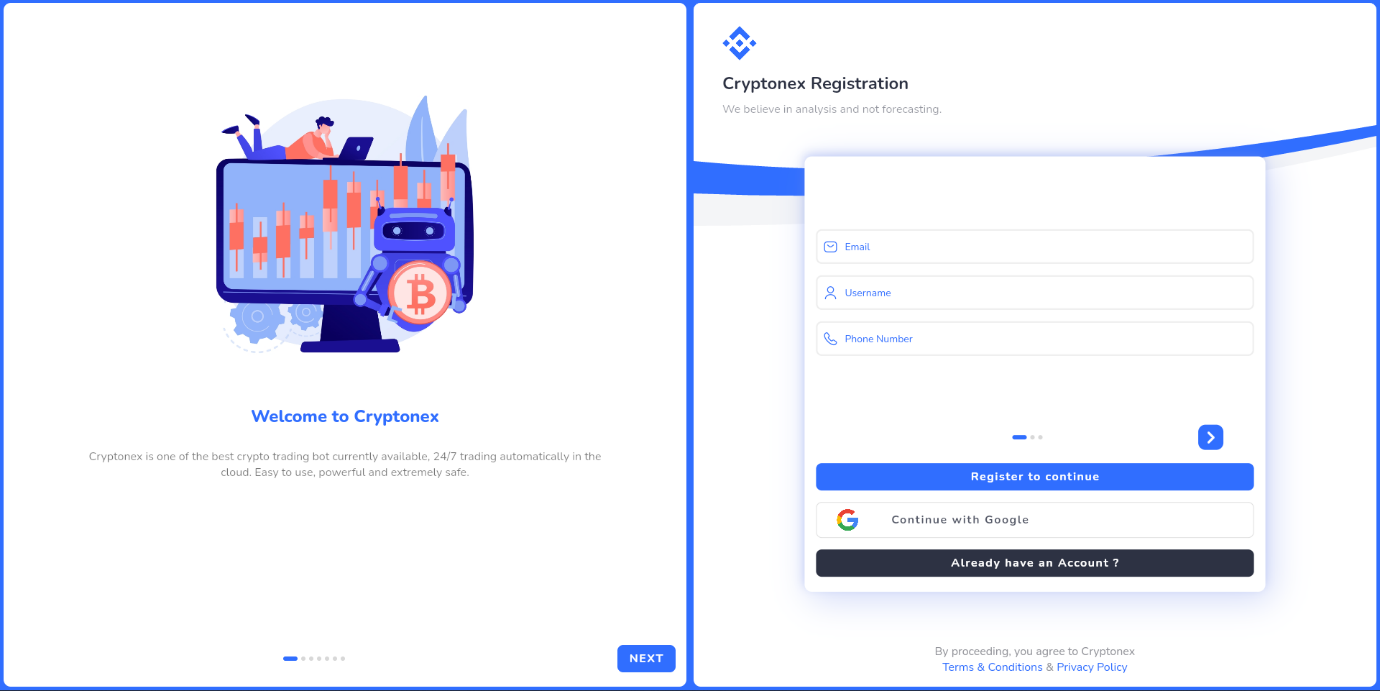
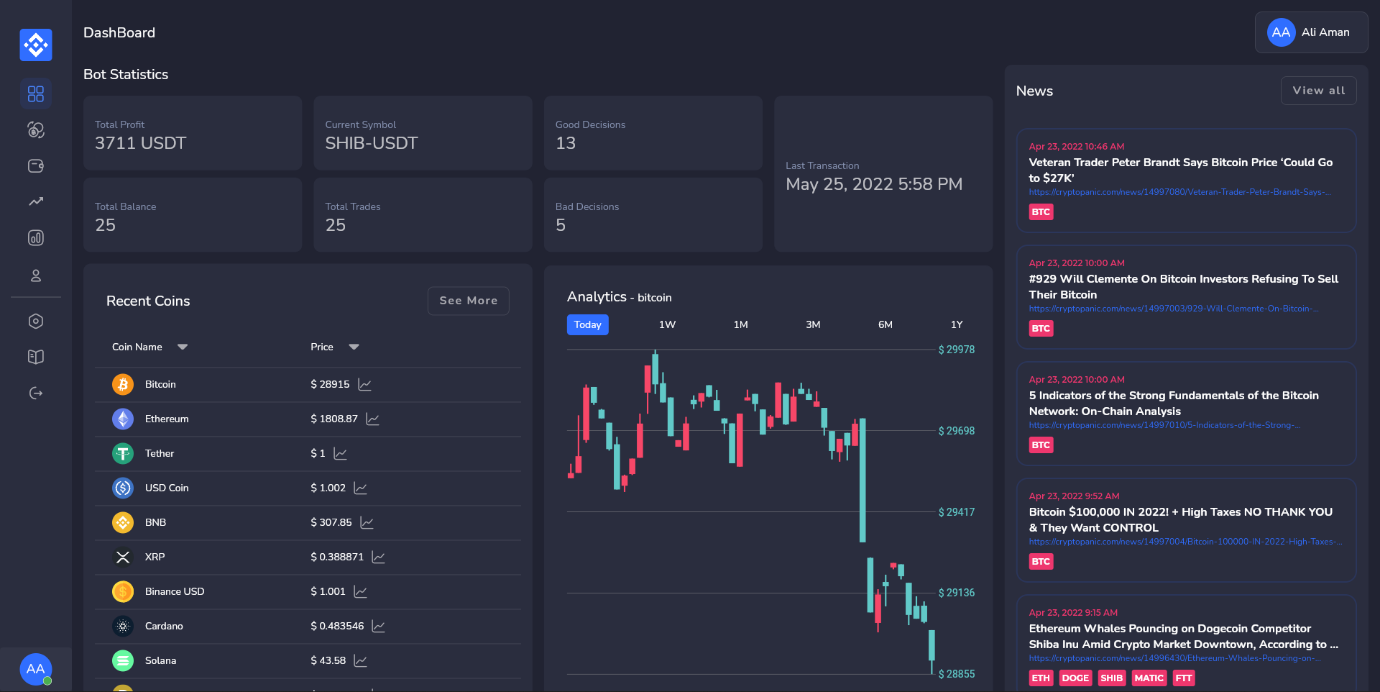


Figure 17:Registration Page



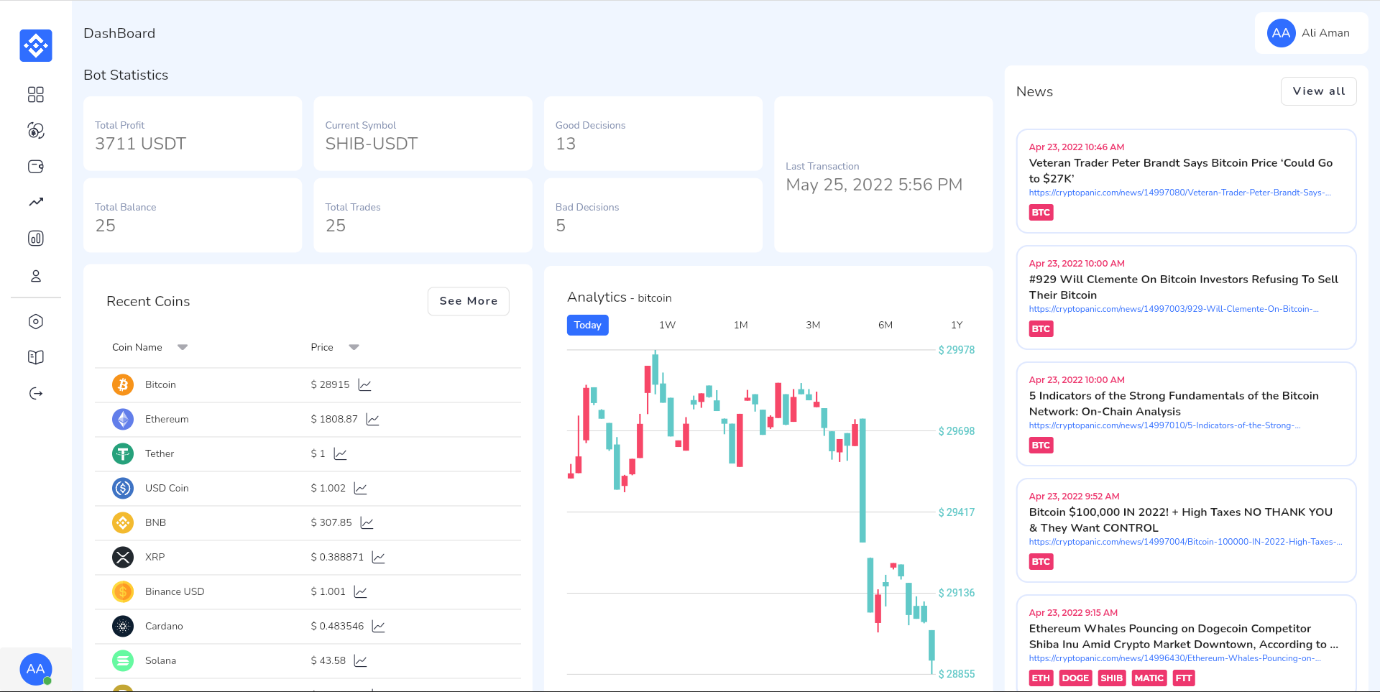
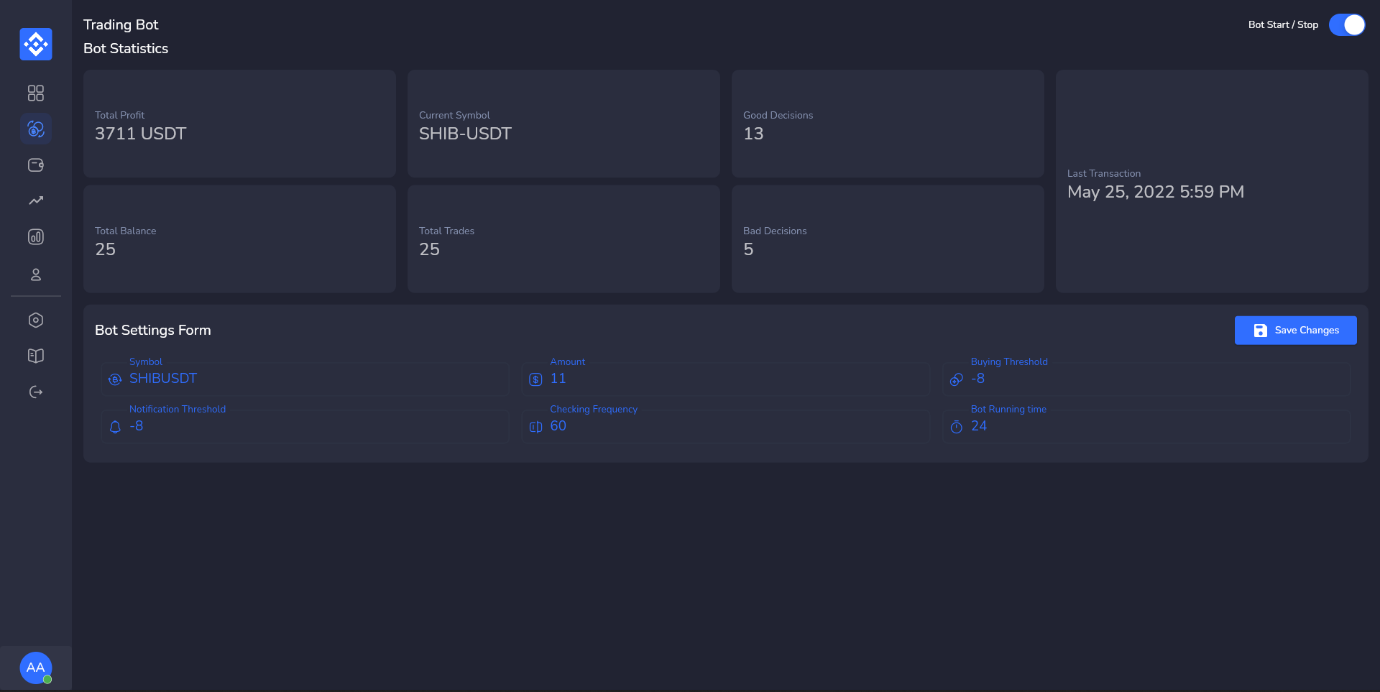


Figure 18:(Dark & Light Mode): Main Dashboard on web.



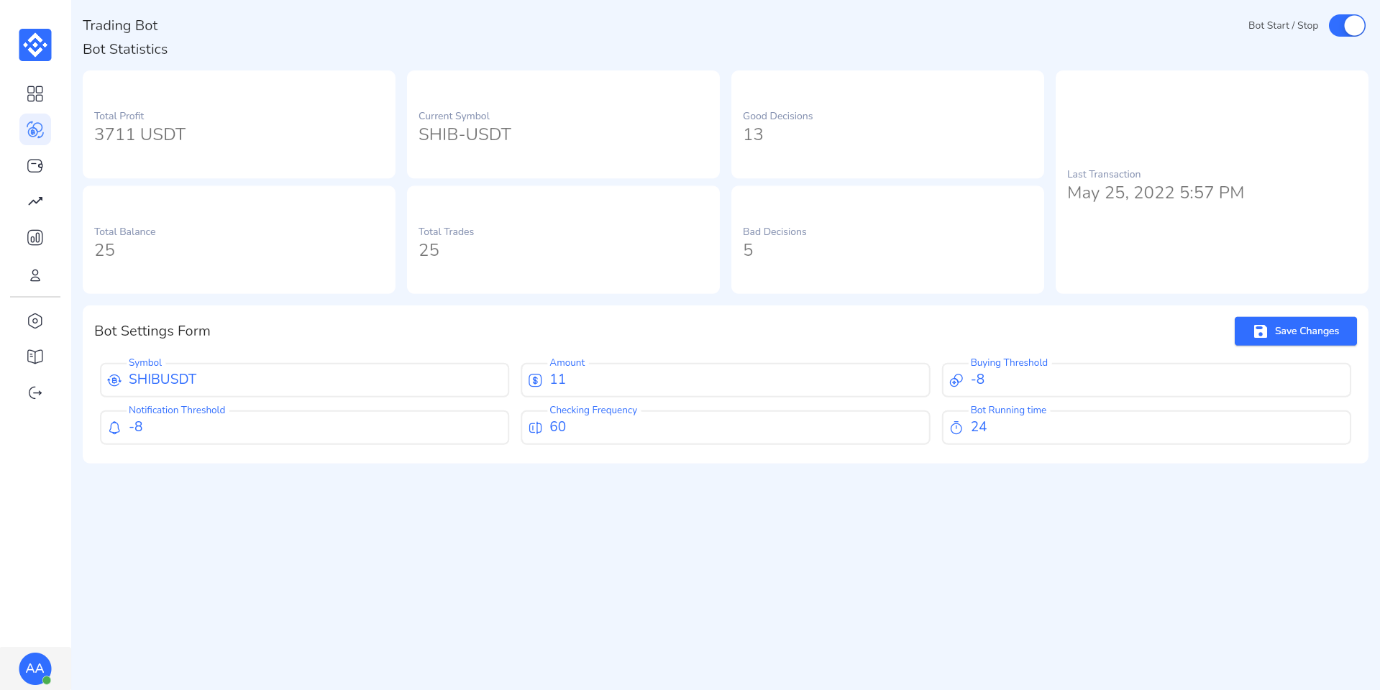


Figure 19:(Dark & Light Mode): Bot settings



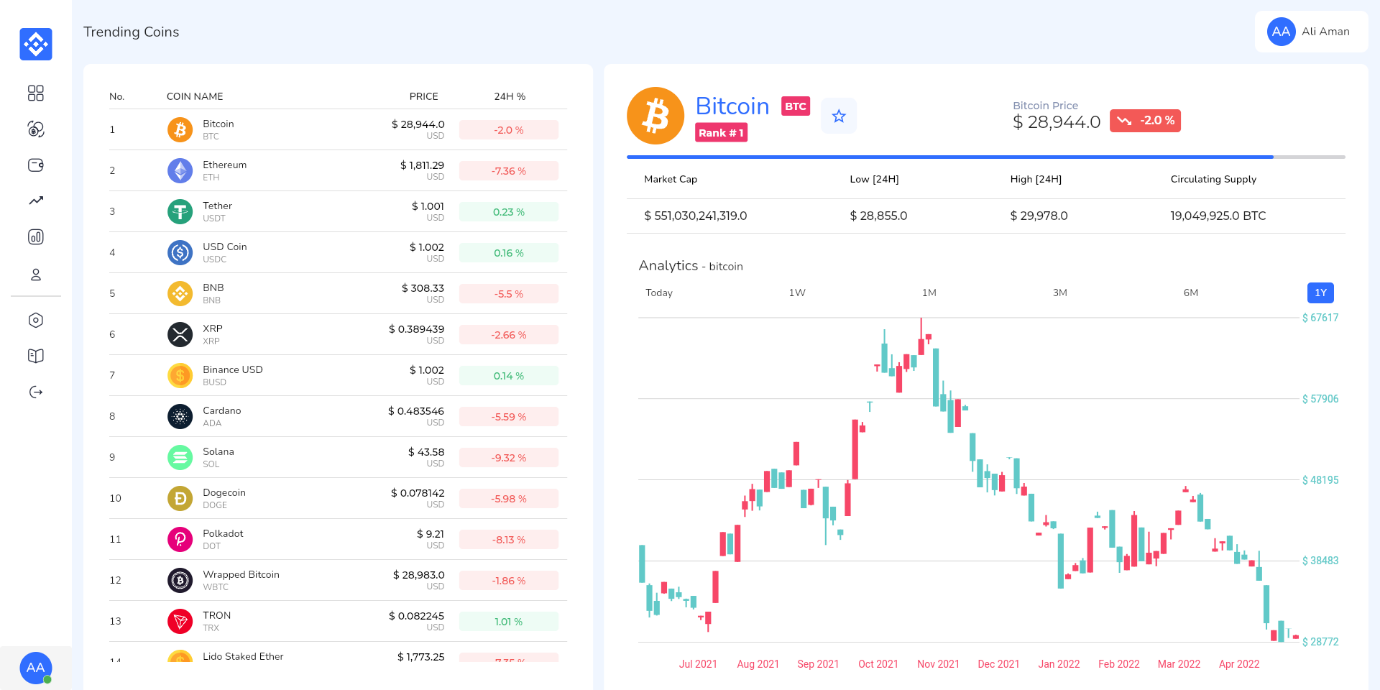


Figure 20:(Dark & Light Mode): Trending coins and their statistics

## **Test Plans**

## **Purpose of Test Plan**

This Test Plan document, documents and tracks the necessary information required to effectively validate and test the functionalities of the CRYPTONEX Project. The main objective is to test if the project meets its requirements. Three types of testing will be carried out, unit testing, performance testing and system testing to ensure the entire system is functional and working properly.

## **Environment Needs**

* The project works on Web and Android platform only and requires an Android version greater than 5.0
* A stable internet connection is required and user must be connected on same network.
* Age Restriction 18+.
* Only English (UK) language supported.

## **Validation Testing**

* + 1. Items to be Tested/Not Tested

|  |  |  |
| --- | --- | --- |
| **Items To Test** | **Test Description** | **Test Date** |
| Text Colour | Text colour is visible on screen and understandable | 01/05/2022 |
| Crypto News | Cryptocurrency related news is visible | 03/05/2022 |
| Menu Bar | Menu bar is visible and responsive | 03/05/2022 |
| Buttons | Buttons are clickable and working | 03/05/2022 |
| Crypto Analytics | Candle stick graphs are visible | 07/05/2022 |

Table 10:Items to be Tested/Not Tested in Non-Functional Testing

## **Test Approach(s)**

Unit Testing: The main objective is to test the individual’s components of the system in order to check if they are functioning properly.

1. **Integration Testing:** Testing of all integrated modules to verify the combined functionality after integration is in workable state.
2. **System Testing:** The entire system is tested as per the requirements. Overall requirement specifications and all the combined parts of a system are tested.
3. **Interface Testing:** The objective of this testing is to validate the interface as per the business requirement. This testing includes the size of the buttons and input field present on the screen, alignment of all text, tables, and content in the tables. It also validates the menu of the application, after selecting different menu and menu items, it validates that the page does not fluctuate and the alignment remains same after hovering the mouse on the menu or sub-menu.
4. **Performance Testing**: The objective of this testing is to ensure the system works under heavy traffic and also on low latency.
5. **Compatibility Testing:** This testing will ensure that the application works on different browsers and operating system and different android phones. This type of testing also validates whether web application runs on all versions of all browsers or not.

## **Test Pass/Fail Criteria**

If the application crashes at any stage of the testing, then it is a failure else it is Pass.

**Test Deliverable(s)**

Test Plans, Test Specification, and Test Cases

## **Non-Functional Testing**

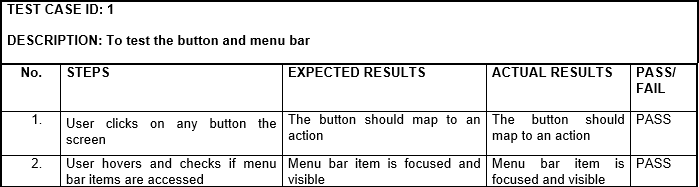
**Test Cases**

Table 11:Test Case 1

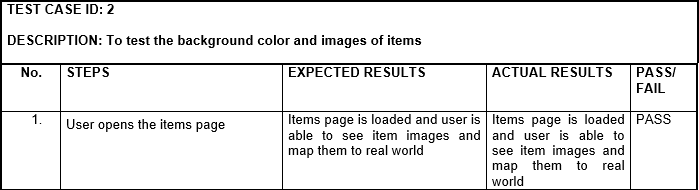


Table 12:Test Case 2

## **Functional Testing**

**Items to be Tested/Not Tested in Functional Testing**

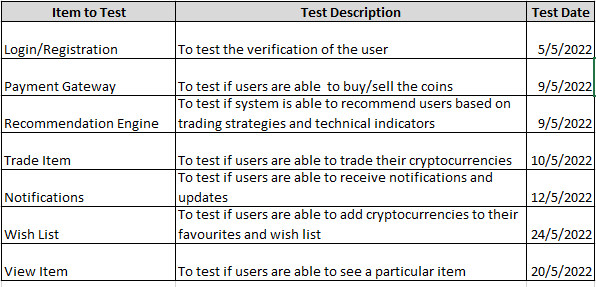


Table 13:Item to be Tested/Not Tested in Functional Testing

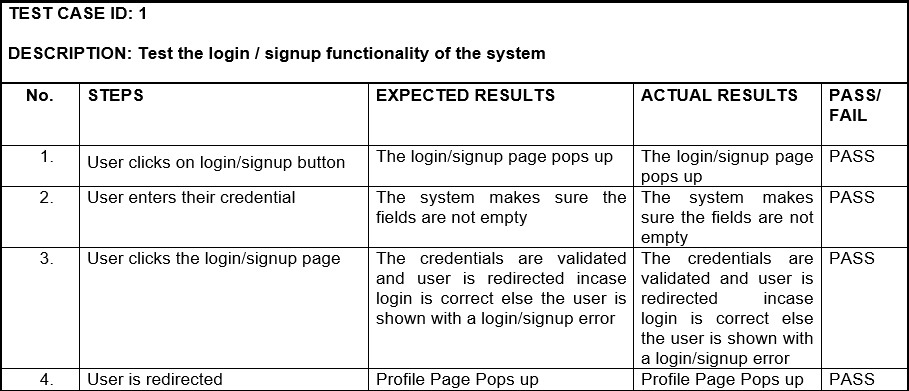


Table 14:Test Case 1

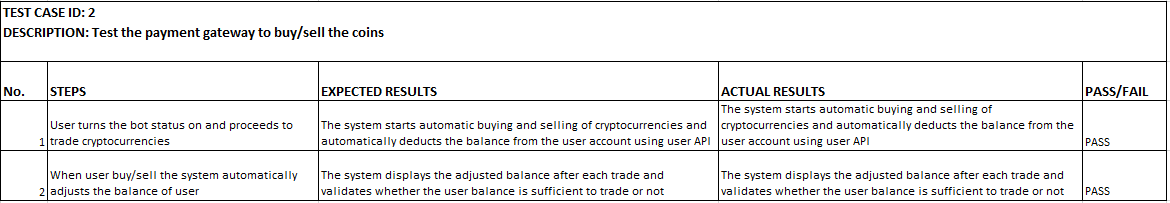


Table 15:Test Case 2

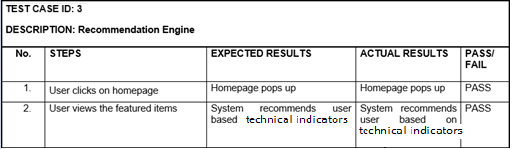


Table 16:Test Case 3

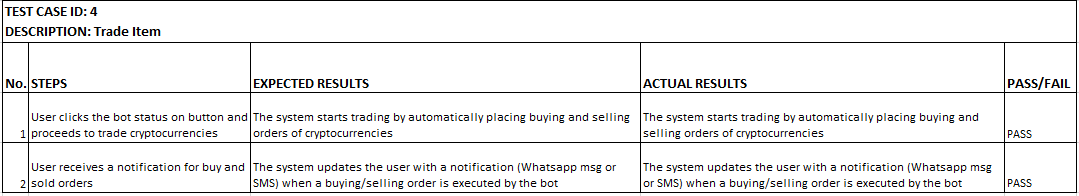


Table 17:Test Case 4

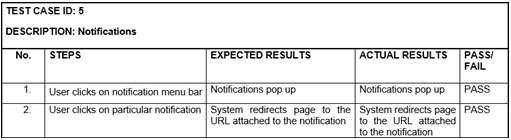


Table 18:Test Case 5

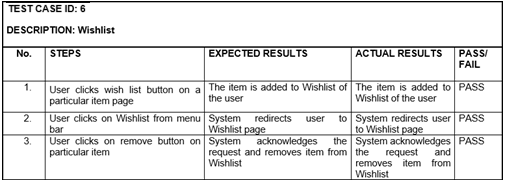


Table 19:Test Case 6

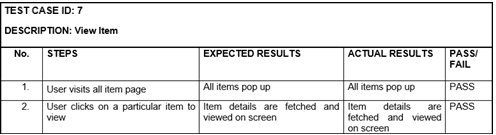


Table 20:Test Case 7

# **Project Code**

## **Frontend Code:**

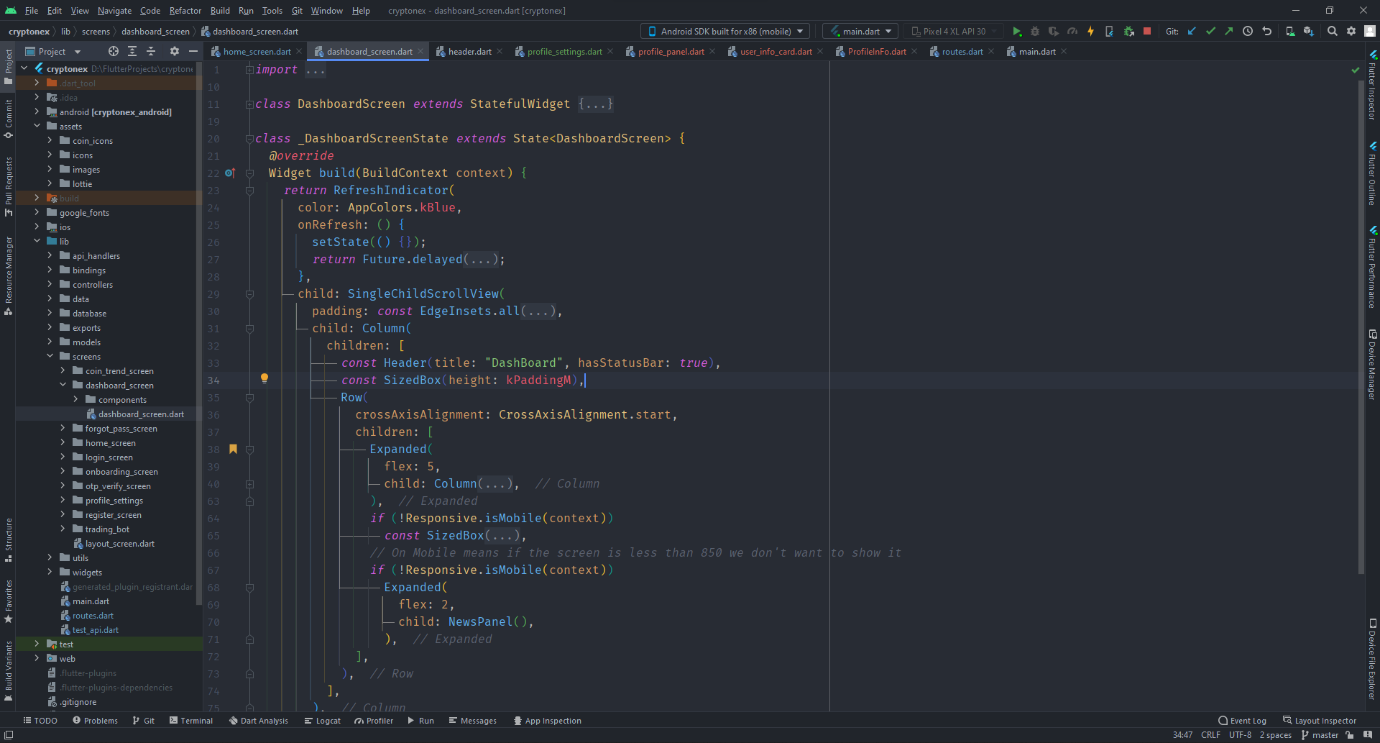


Figure 21:Frontend Dashboard

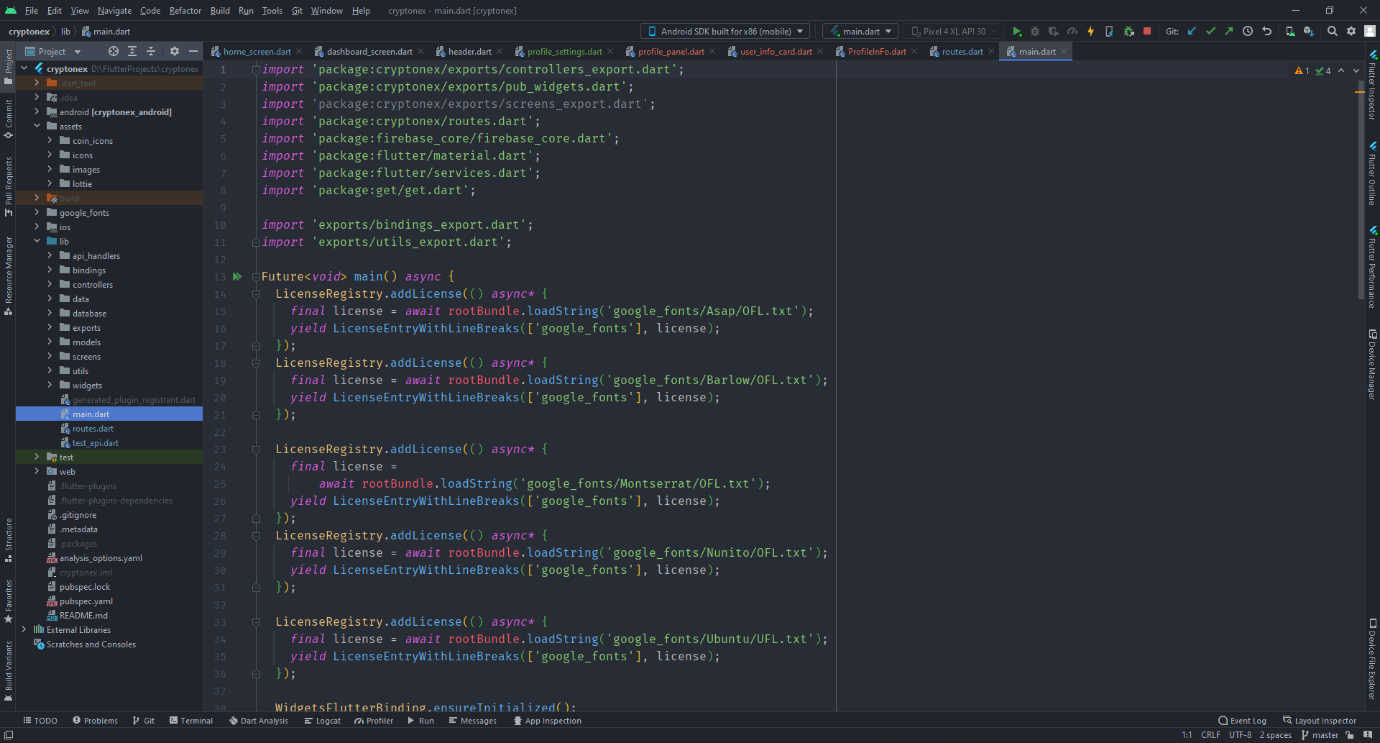


Figure 22:Frontend Main

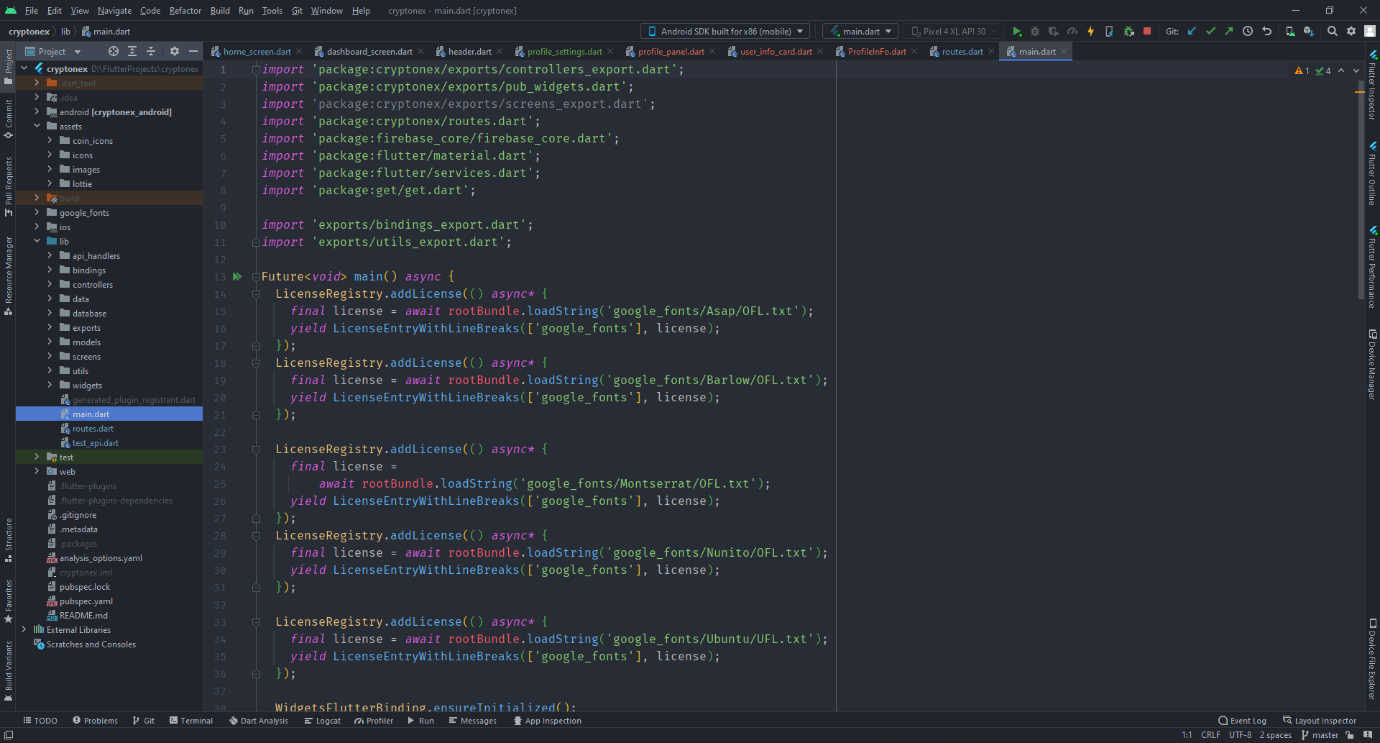


Figure 23:Frontend Routes

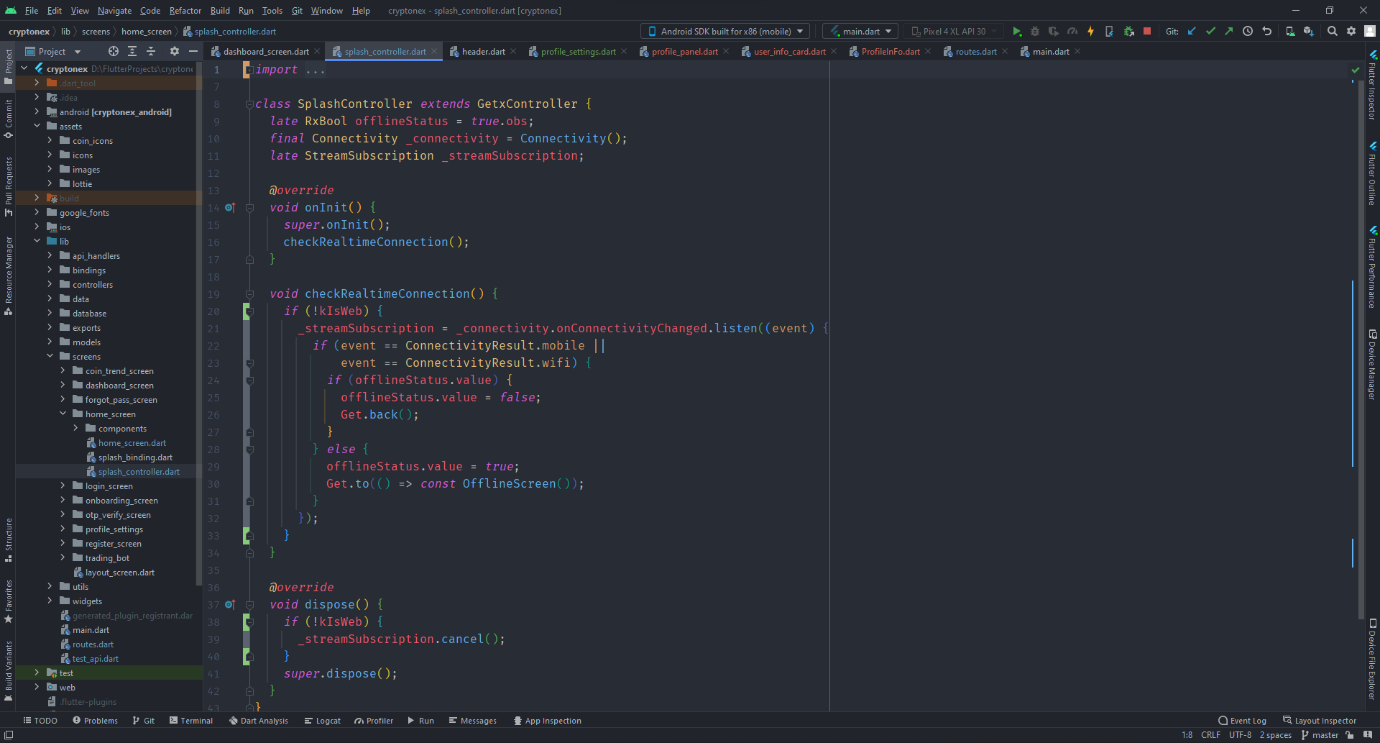


Figure 24:Frontend Splash

## **Backend Code:**

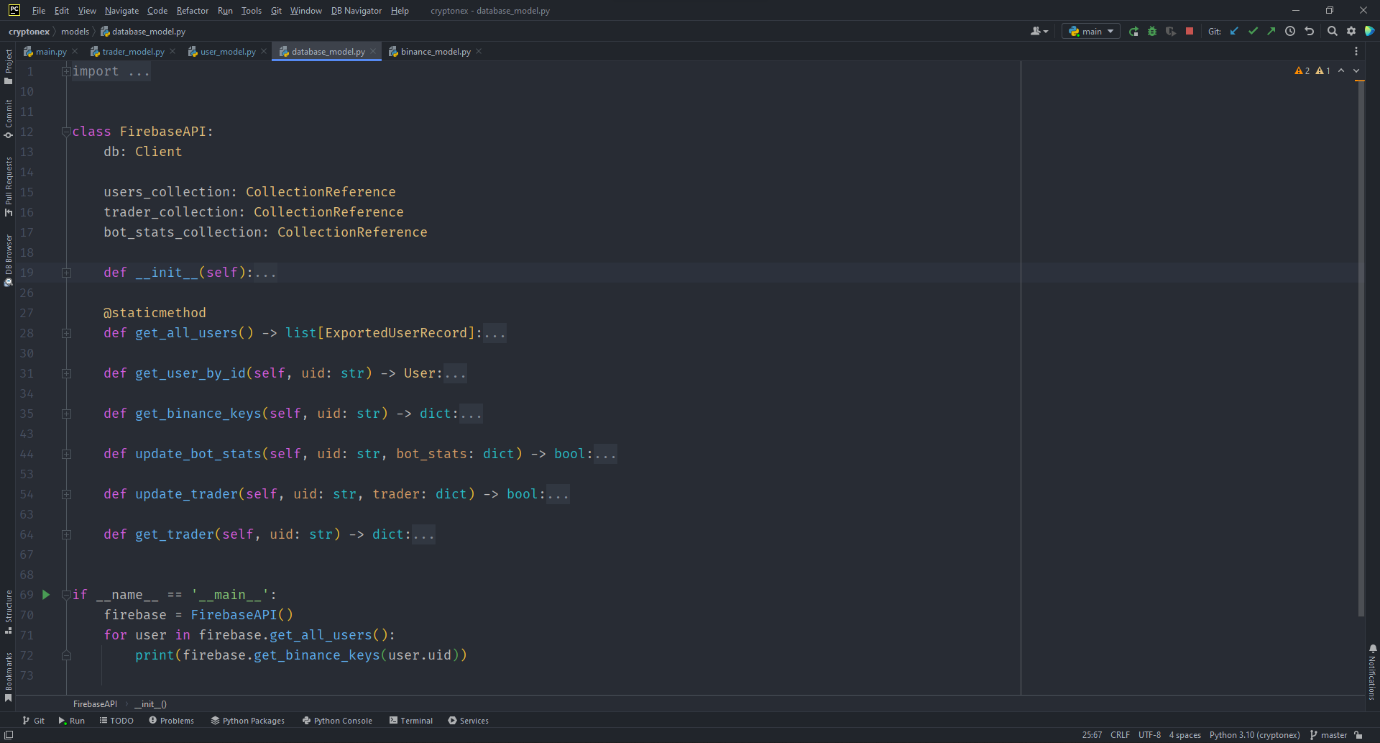


Figure 25:Firebase API

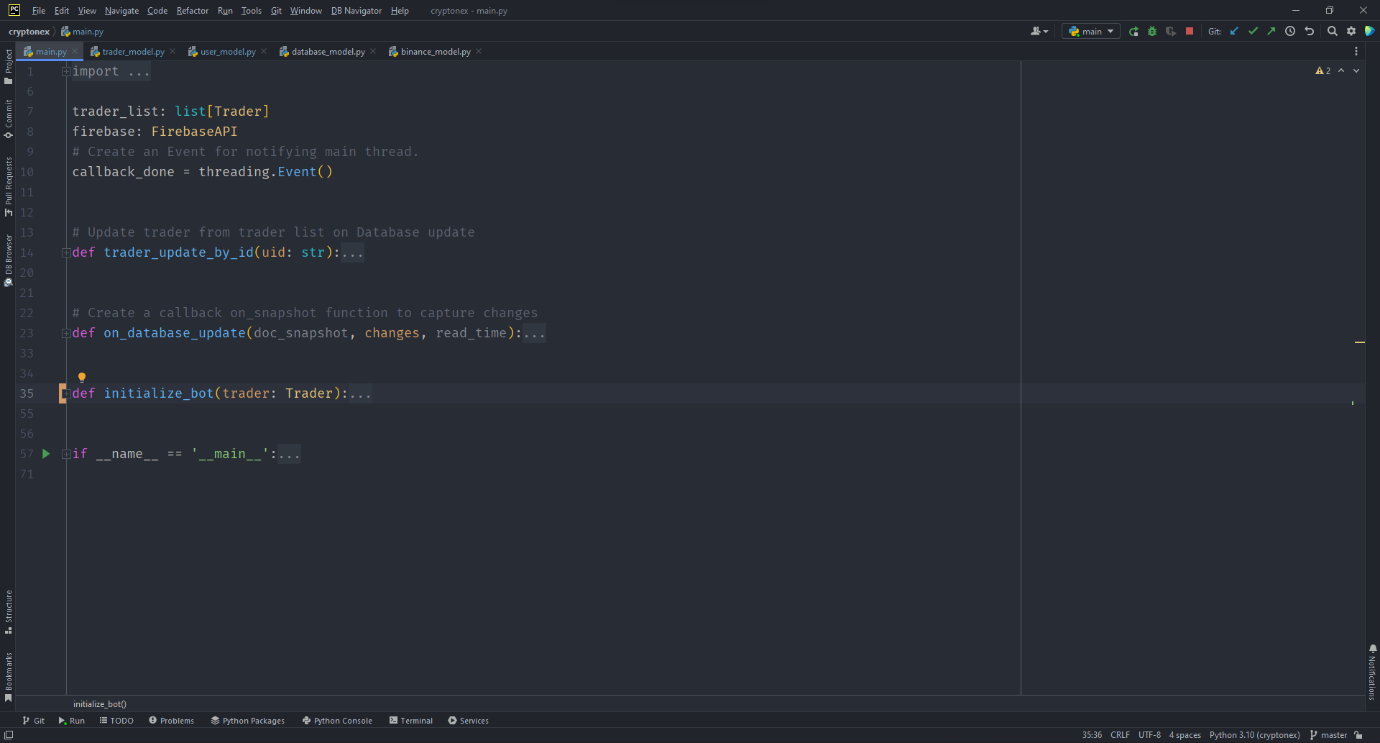


Figure 26:Main

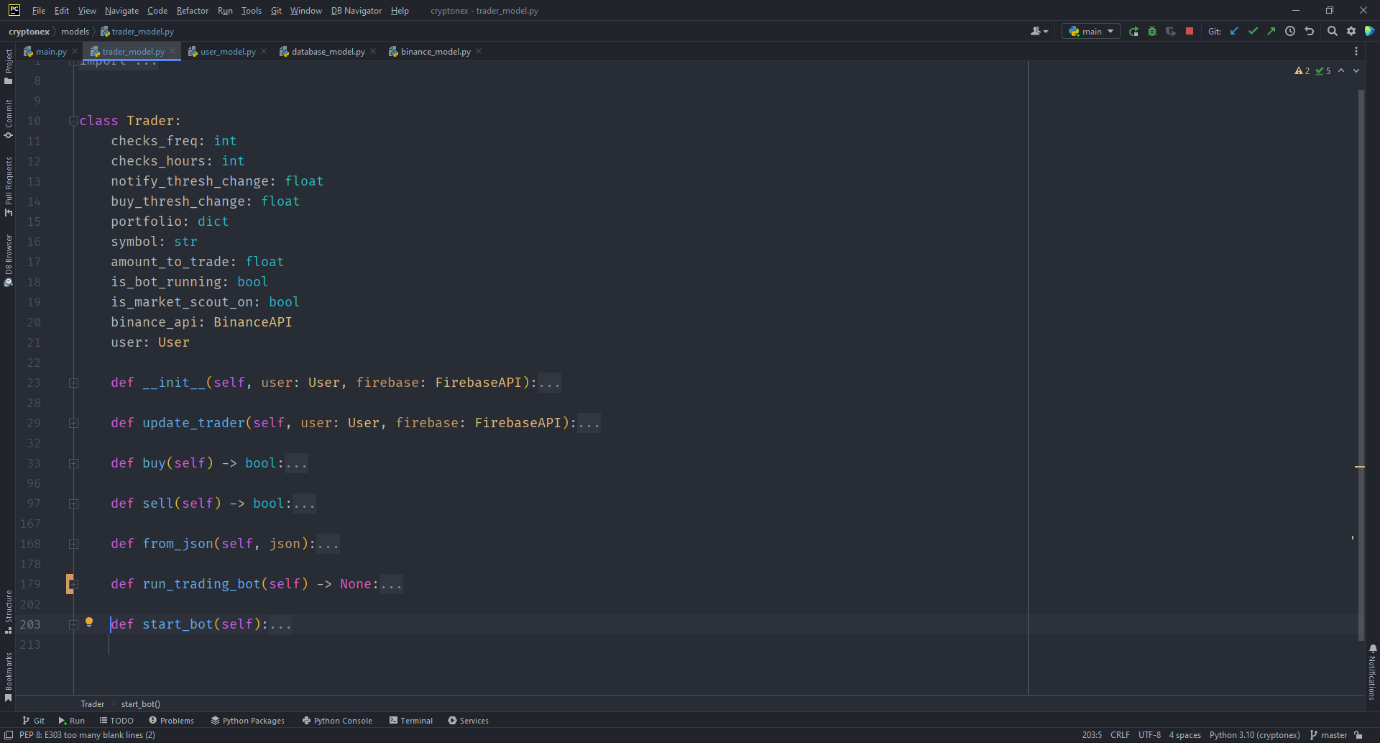


Figure 27:Trader Model

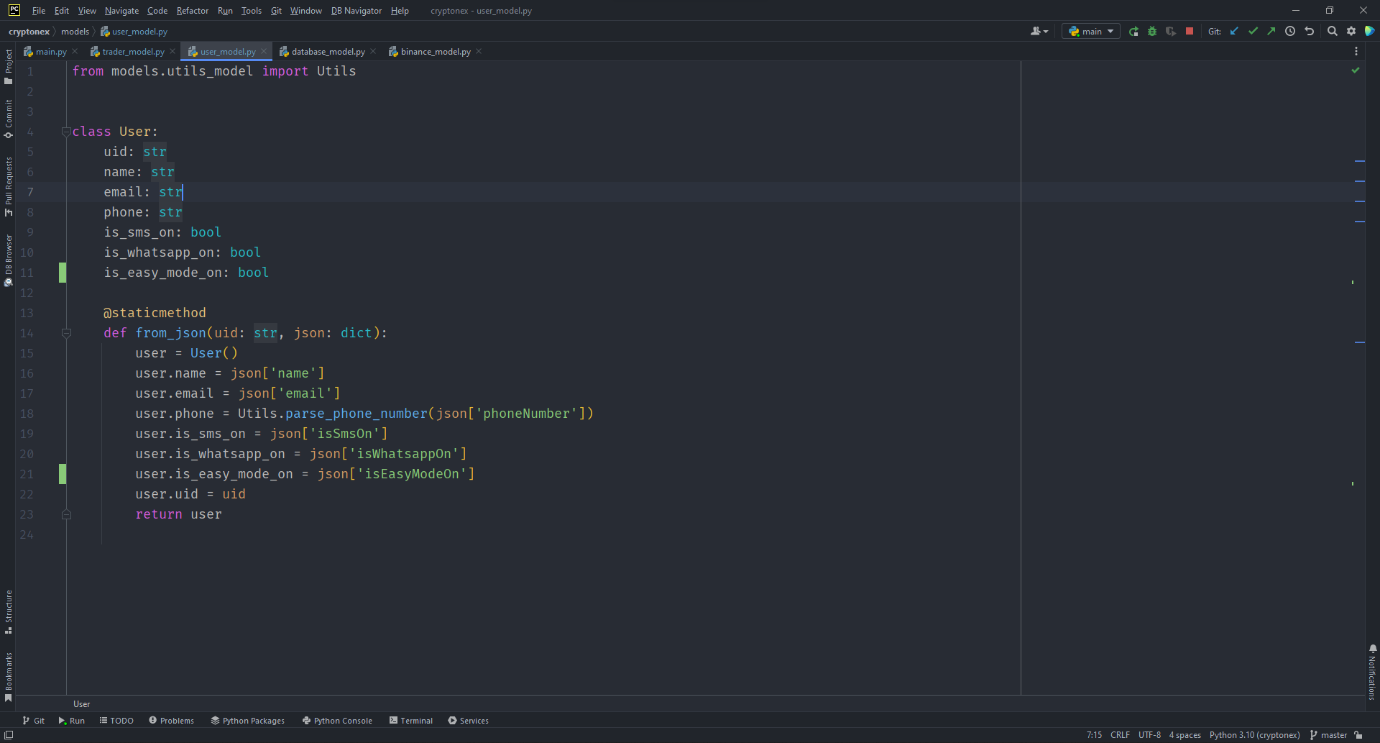


Figure 28:User Model

# **Limitation of System**

* Cryptonex will be functioning only in Pakistan but can be extended to other countries as well.
* Cryptonex app need a highly stable internet connection for proper functionality.
* Only expert users can modify trading bot defined parameters on which the trading will take place.
* Users can not use Cryptonex trading bot until they have provided their binance API key and secret key to access Binance exchange using Cryptonex trading bot application.

# **Future Work**

The team behind Cryptonex realizes that the project has potential to be a successful application in near future for its stop loss functionality. Some of the planned future work includes:

* Deploying the application server on AWS so that the Cryptonex Bot runs 24/7 and uses elastic load balancing internet stability services of AWS for seamless trading.
* Expanding the domain for Cryptonex outside of Pakistan, on an international level.
* Publishing the application in Urdu and/or other languages to make easy to use for locals.
* This project can be further extended to incorporate other different currencies such as Ethereum, Bitcoin, and many more.
* With further development, the CRYPTONEX trading bot can be extended to be used for MAC OS and Apple devices.
* Conclusively, our trading bot can also be developed for Binance Future Trading.

# **Conclusion**

The Final Year Project was truly a learning experience that led all the team members to learn the importance of communication, collaboration and dealing with last minute errors. Moreover, strong and stable team with a diverse strength point of every individual made this project successful. This gives us the lesson that a diverse team is always on the right path towards success. Our team was steadfast throughout the year, striving to make the best version of our application by revising and editing the work. Trade It also enabled the team members to look in to different domains, for example the android and web developers got to learn a lot from each other and learn about each other ‘s domains.

# **References**

1. Jignesh Davda, Binance Python API – A Step-by-Step Guide, August 30, 2021 [https://algotrading101.com/learn/binance-python-api-guide/?fbclid=IwAR3N870zexP4ZCXGDVndyE2rZ9D-3gdSCYuJc67LDaySRI2hU4Uu\_y5QIlo]
2. Sam McHardy, Python-binance Documentation. September 27, 2021 [https://readthedocs.org/projects/python-binance/downloads/pdf/stable/]
3. Binance Academy ,5 Essential Indicators Used in Technical Analysis Mar 2, 2021 [https://academy.binance.com/en/articles/5-essential-indicators-used-in-technical-analysis]
4. Murphy, John J. (1999). Technical analysis of the financial markets. October 1, 2021 [https://en.wikipedia.org/wiki/Technical\_indicator]
5. Mayank Vadsola Building a cryptocurrency dashboard using Plotly and Binance API,

Feb 23, 2021

[https://towardsdatascience.com/building-a-cryptocurrency-dashboard-using-plotly-and-binance-api-352e7f6f62c9]