Diagram

Description automatically generatedCairo University

Faculty of Computers and Artificial intelligence Department of Computer Sciences

# Crypto Manager

Supervised by

*Dr. Samar Hesham*

*TA. Hussein Khaled*

Implemented by

|  |  |
| --- | --- |
| *20186004* | *Adham Alaa Eldin* |
| *20186006* | *Andrew Emad* |
| *20186027* | *Marwan Ayman* |
| *20186018* | *Omar Tarek* |
| *20186047* | *Abdulrahman Ayman* |

Graduation Project

Academic Year 2021-2022

Final project documentation

Graphical user interface, text, application

Description automatically generated

## Table of Contents

Contents

[Crypto Manager 1](#_Toc108789850)

[Table of Contents 3](#_Toc108789851)

[1.Introduction 7](#_Toc108789852)

[1.1 Motivation 7](#_Toc108789853)

[1.2 Problem definition 8](#_Toc108789854)

[1.3 Project objective 8](#_Toc108789855)

[1.4 Gantt chart of project time plan 9](#_Toc108789856)

[1.5 Project development methodology 11](#_Toc108789857)

[1.6 The used tools in this project 18](#_Toc108789858)

[1.7 Report Organization 19](#_Toc108789859)

[2. Related Work 20](#_Toc108789860)

[3. System analysis 22](#_Toc108789861)

[3.1 project specification 22](#_Toc108789862)

[3.1.1 Functional requirements 22](#_Toc108789863)

[3.1.2 Non-functional requirements 23](#_Toc108789864)

[3.2 Use Case Diagrams 24](#_Toc108789865)

[4. System Design 35](#_Toc108789866)

[4.1 System component diagram 35](#_Toc108789867)

[4.2 System class diagrams 36](#_Toc108789868)

[4.3 Sequence diagrams 38](#_Toc108789869)

[4.4 Project ERD 44](#_Toc108789870)

[4.5 System GUI Design 45](#_Toc108789871)

[5.Implementation and Testing 49](#_Toc108789872)

[5.1 implementation 49](#_Toc108789873)

[5.2 Testing 55](#_Toc108789874)

[References 61](#_Toc108789875)

List of tables

Table(1) 10

Table(2) 26

Table(3) 27

Table(4) 27

Table(5) 28

Table(6) 29

Table(7) 30

Table(8) 31

Table(9) 32

Table(10) 33

Table(11) 34

Table(12) 56

Table(13 56

Table(14) 56

Table(15) 57

Table(16) 57

Table(17) 58

Table(18) 58

Table(19) 58

Table(20) 58

Table(21) 59

Table(22) 59

Table(23) 60

Table(24) 60

Table(25) 60

List of figures

Figure(1)………………………………………………………………………………………………………………………………………….9

Figure(2)………………………………………………………………………………………………………………………………………….9 Figure(3)………………………………………………………………………………………………………………………………………….11 Figure(4)………………………………………………………………………………………………………………………………………….13 Figure(5)………………………………………………………………………………………………………………………………………….13Figure(6)………………………………………………………………………………………………………………………………………….14Figure(7)………………………………………………………………………………………………………………………………………….14Figure(8)………………………………………………………………………………………………………………………………………….15Figure(9)………………………………………………………………………………………………………………………………………….16 Figure(10) 16

Figure(11) 17

Figure(12) 17

Figure(13) 24

Figure(14) 25

Figure(15) 25

Figure(16) 35

Figure(17) 36

Figure(18) 36

Figure(19) 37

Figure(20) 38

Figure(21) 38

Figure(22) 39

Figure(23) 39

Figure(24) 40

Figure(25) 41

Figure(26) 41

Figure(27) 42

Figure(28) 42

Figure(29) 43

Figure(30) 44

Figure(31) 45

Figure(32) 45

Figure(33) 46

Figure(34) 46

Figure(35) 47

Figure(36) 47

Figure(37) 48

Figure(38) 49

Figure(39) 49

Figure(40) 50

Figure(41) 50

Figure(42) 51

Figure(43) 51

Figure(44) 52

Figure(45) 52

Figure(46) 53

Figure(47) 53

Figure(48) 54

List of abbreviations

TC\_ID = Test case number

LSTM = Long Short-Term Memory

API = Application Programming Interface

SVM = Support Vector Machine

XGBoost = Extreme Gradient Boosting

SQL = Structured Query Language

BTC =Bitcoin

ETH = Ethereum

SARIMA = Seasonal Autoregressive Integrated Moving Average

REST = Representational State Transfer

RMSE= Root Mean Square Error

LR = Linear Regression

GUI = Graphical User Interface

ERD = Entity Relationship Diagram

OCHLV = Open, High, Closed and Volume values

# 1.Introduction

## 1.1 Motivation

Since cryptocurrencies have the potential to be extremely detrimental to society, it is essential to understand the primary motives of the project that we are doing. The blockchain technology underlying bitcoin and other cryptocurrencies has been hailed as a potential gamechanger for a large number of industries, from shipping and supply chains to banking and healthcare.

This makes investment in crypto currencies more attractive for its decentralized nature, protection form inflation and High Security as it is way more secure than the normal transactions which include more people than technology. For people who believe in that promise, investing in cryptocurrency represents a way to earn high returns while supporting the future of a rising technology that is booming nowadays.

The motivation for idea of this project is the oblivious rise of cryptocurrency trading field which is happening more rapidly than it ever was and with increase of its popularity it becomes more important for investors to enter this field the application would help those new investors in making decisions about buying selling or holding the cryptocurrencies and also to spread awareness about how important the cryptocurrencies have become in our daily life and to shine a light on that.

The main application idea is to provide the investor with all the possible tools to help him make a sound decision in investment, we will be displaying live data for the cryptocurrencies, providing the user with the latest news, using machine learning to predict the rise and fall of cryptocurrencies within the next days and training him through a live simulation.

## 1.2 Problem definition

Crypto currency investment: These days, crypto currency is becoming more and more important in our lives. It is the new way of trading and will soon replace normal currencies for its transparency and ease to trade. The main problem is the lack of knowledge when investing or trading in crypto that makes people lose a lot of money or be afraid of even trying to invest in it although it can be more profitable than normal investments and gives high revenues. Our job is to help and support the user with decision making tools to guide him through the crypto world to decrease the risks involved as much as possible with:

1. Live data on crypto currencies
2. giving news and updates of how a certain currency is faring
3. prediction tools that predict if a currency will go up or down
4. A live simulation to familiarize the user with the real trading environment

## 1.3 Project objective

Our main objective is to deliver a reliable predictor by applying the development skills that we learnt using machine learning techniques and help attract more users into crypto investments and erase the fear of new investors and providing more knowledge to general people about what crypto investment is and how is works and keeping the users updated through daily live news about cryptocurrencies and providing accurate predictions to the users because not only will it help our project but it also will help enlighten a greater number of people who might have certain fears when thinking about investing in cryptocurrencies as the more people invest in reliable secure crypto currencies the less banking problems that will happen as the transactions and currencies will be more technology oriented avoiding the human error that normally happens in normal banking transactions and this is our objective

Suggested solutions for crypto predictions: we first tried Traditional Machine learning techniques in time series Forecasting to predict the crypto prices, in order to choose the perfect algorithm for each coin we calculated the root mean square error for each one for each coin

## 1.4 Gantt chart of project time plan



Figure(1): gantt chart of time plan



Figure(2): progress milestones

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Task Title | Description | Task status |
| 1 | Abstract | The existing problem is that the determination of investing in cryptocurrency and we are trying to solve it by providing a reliable software that has a decent accuracy in decision making whether to invest or not. | (Due 21/11/2021) Completed |
| 2 | Background | The main area is the crypto field, and the motivation is to help people not waste money by providing an accurate decision maker for them. Techniques are machine learning related techniques. | (Due 24/11/2021) Completed |
| 3 | Problem definition | Our app will be tackling two problems:  **1-Crypto currency investment:**  The problem is the lack of knowledge when investing or trading in crypto that makes people lose a lot of money or be afraid of trying to.  **2-stock market investment:**  Stock market is a popular way to invest money though it comes with a lot of risks. Lack of knowledge on the subject can be detrimental. | (Due 03/12/2021) Completed |
| 4 | Related work | Similar researches that faced the same problems we are trying to solve. | (Due 06/12/2021) Completed |
| 5 | Stakeholders | Determining Person who can affect or be affected by the system. | (Due 10/12/2021) Completed |
| 6 | System architecture | Defines the structure of a software system. | (Due 12/12/2021) Completed |
| 7 | Use-case Diagram | Shows the use cases for the system (the actors and the relationships between the actors and use cases). | (Due 20/12/2021) Completed |
| 8 | Functional Requirements | Defines the functionalities of the system. | (Due 24/12/2021) Completed |
| 9 | Non-functional Requirements | Define system attributes such as security, reliability, performance, maintainability and usability. | (Due 27/12/2021) Completed |
| 10 | Entity Relationship Diagram | Shows the relationships of entity sets stored in the database. | (Due 07/02/2022) Completed |
| 11 | Class Diagram | Describes the structure of the system by showing the system's classes, their attributes, operations and the relationships among objects. | (Due 11/02/2022) Completed |
| 12 | Sequence Diagram | Describe the interactions among classes in terms of an exchange of messages over time. | (Due 15/02/2022) Completed |
| 13 | Component Diagram | Shows how the components are wired together to form the system. | (Due 17/02/2022) Completed |
| 14 | Implementation | In this phase we implement our system functionalities. at the beginning we create our DL model (like LSTM, Linear regression.) then perform training and testing on it until a we reach a satisfying result. | (Due 29/06/2022)  Completed |
| 15 | Testing | **The testing will be done in two phases:**   1. Designing the test cases 2. Applying system testing | (Due 15/07/2022)  Completed |
| 16 | Technologies we need to learn | **What we have learned:**  Python, Java Script, HTML, NN  **What we are planning to learn:**  Machine learning topics. | (Due 30/03/2022)  Completed |

Table(1): The project plan

## 1.5 Project development methodology

We used the waterfall model development methodology

Diagram

Description automatically generated

Figure(3): waterfall model

We used the waterfall development are that it allows for easier control as a schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one.

We also chose the waterfall model as the requirements were clear and easy to understand

We first started with defining what we want our system to do then we clearly wrote down the requirements based on the actual needs to solve a real-world problem

We then moved to the design phase of the project, and we chose the three-layered architecture as we found it the most suitable architecture for the project

We then moved to the implementation phase in which we developed each set of functionalities within a certain schedule and deadline

We then moved to the testing phase which we used the equivalence partitioning testing approach

Then finally we moved to the deployment phase, and we did some maintenance and tweaks for our project to be as stable as possible

Also, the waterfall model is simple and easy to understand and use

Easy to manage as for each phase we had specific deliveries and a review process.

Phases are processed and completed one at a time so that gave us more time to optimize each of the functionalities of each phase.

The waterfall model has clearly defined stages so in each stage we knew what need to be completed and when it was needed to be

We also had well understood milestones that we wanted to achieve so we put our best in delivering a complete and stable product

Also, we had little issues with documenting the results of the process and the process itself to keep track of how far we have come through the project and if there are any fixes needed to be done

In this project we will use some of the machine learning techniques like sentiment analysis and LSTM (Long Short-Term Memory) aiming to predict the price movement of the cryptocurrency.

The following graphs are comparisons of traditional machine learning algorithms on different coins:

Bitcoin

Figure(4): Algorithms comparison for bitcoin

Ethereum

Figure(5): Algorithms comparison for Ethereum

Tether

Figure(6): Algorithms comparison for Tether

Binance

Figure(7): Algorithms comparison for Binance

XRP

Figure(8): Algorithms comparison for XRP

although we got good results in training and testing, the actual performance was poor as shown in the figures below

The following graphs are bitcoin actual vs predicted prices:

Chart, line chart

Description automatically generated

Figure(9): Linear regression results

Chart, line chart

Description automatically generated

Figure(10): XGBoost results

Chart, line chart

Description automatically generated

Figure(11):SVM results

We then tried to use the LSTM algorithm to predict crypto currency prices but we still did not get good results as the model overfitted since our data was small but eventually we tried using LSTM with sentiment analysis and we got better results

Chart, line chart

Description automatically generated

Figure(12):LSTM with & without sentiment analysis results

## 1.6 The used tools in this project

Binance API which is a method that allows you to connect to the Binance servers via Python or several other programming languages. Which with we can automate your trading. More specifically, Binance has a RESTful API that uses HTTP requests to send and receive data we used Binance API to retrieve live real time prices in order to be used in our live simulation

flask-sql alchemy is the Python SQL toolkit and Object Relational Mapper that gives application the flexibility of SQL we used it in the backend database implementation to be able to deal with it using python functions

Nomics API: A cryptocurrency index that provide wide range of cryptocurrency data and meta data

CryptoCompare API: is a platform providing data about the crypto like the data for cryptocurrencies to comparisons of the various crypto exchanges, to recommendations for where to spend your crypto assets we used it for returning the current price of any cryptocurrency and all the trading info for the requested currencies to be compared

## 1.7 Report Organization

So far, we have talked about the motivation for our project, problem definition and the project main objective  
We have also displayed the Gantt chart of the project time plan and we talked about the project development methodology and the used tools in this project

**The next chapters will cover:**   
The work that’s been done with relation to our project and examples will be stated

The system analysis which will include the functional and non-functional requirements and the use case diagrams

The system design which will include:

• System Component Diagram

• System Class Diagrams

• Sequence Diagrams

• Project ERD

• System GUI Design

The implementation and testing of our project and the results

Finally, the references

# 2. Related Work

A research by the author Vatsal H. Shah: It revolves around applying machine learning techniques to stock market predictions Algorithm used: Decision Stump, Linear Regression, Support Vector Machines, Boosting Conclusion: Of all the Algorithms we applied, we saw that only Support Vector Machine combined with Boosting gave us satisfactory results. Linear Regression gave lower mean squared errors while predicting the EMA pattern.

A research whose authors are JG Agrawal, V Chourasia, A Mittra: State-of-the-Art in Stock Prediction Techniques Algorithm used Multinomial Logistic Regression (MLR)isotonic regression function Multi-Layered Feed Forward, Adaptive Neuro-Fuzzy Inference System, Multilayer perceptron network Conclusion: the existing techniques are not suitable for prediction of stock market trends as well as price of different socks. There exists a gap between technologies and user requirement for a safe and accurate stock prediction system.

A research whose authors are Xiao Ding, Yue Zhang ,Ting Liu and Junwen Duan :deep learning method for event driven stock market prediction Algorithm used : bags-of-words, Support Vector Machines ,feed forward neural network, convolutional neural network prediction model Conclusion: deep learning is useful for event-driven stock price movement prediction by proposing a novel neural tensor network for learning event embeddings, and using a deep convolutional neural network to model the combined influence of long-term events and short-term events on stock price movements.

Pintelas et al conducted interesting research, evaluating sophisticated deep learning models for predicting cryptocurrency prices and movements. Their research revealed the significant limitations of deep learning models for exhibiting reliable forecasts.

Based on their experimental analysis, the authors highlighted the need for adopting more advanced algorithmic approaches for the development of efficient and reliable cryptocurrency models.

Patel et al proposed a hybrid cryptocurrency prediction approach, which focuses on Litecoin and Monero cryptocurrencies. The proposed model is based on a recurrent neural network architecture which utilizes LSTM and GRU layers. The data in their study

contained daily Litecoin data from 24 August 2016 to 23 February 2020 and Monero data from 30 January 2015 to 23 February 2020 concerning average price, open price, close price, high and low prices, as well as the volume of trades. The reported experiments demonstrated that the proposed hybrid model outperforms traditional LSTM networks exhibiting some promising results

Along this line, Livieris et al considered improving the forecasting

performance and reliability of deep learning models utilizing three widely utilized ensemble strategies, i.e., averaging, bagging, and stacking. The authors utilized hourly prices of Bitcoin, Ethereum, and Ripple from 1 January 2018 to 31 August 2019. Additionally, they conducted an exhaustive performance evaluation of various ensemble models using several Conv-based and LSTM-based learners as base models. Their analysis highlighted that deep learning and ensemble learning may efficiently be adapted to develop strong, and reliable cryptocurrency prediction models, but with significant computational cost.

The main thing the differentiates our project that we will try to predict the price movement of cryptocurrencies accurately also we will take user related information to recommend for the user best investment by providing reliable decision support mechanism. unlike other cryptocurrency websites, which only gives the users screeners and charts, we will provide our investors with accurate price predictions to aid them in their investment decisions and we will train the newcomers in our live trading simulation

# 3. System analysis

## 3.1 project specification

### 3.1.1 Functional requirements

1. **Sign-up**: The system shall allow users to create a new account and be added to the system database.
2. **Log in**: The system shall allow users to log into his account the system will check the validity of his credentials.
3. **Log out**: The system shall allow users to log out of his account the system will redirect the user to the login screen.
4. **Show live market**: The system shall display the live cryptocurrencies data.
5. **Predict** **prices**: The system shall display prediction for price movements of the cryptocurrencies within next week.
6. **Give overview on coin**: The system shall display a visualization for price movements of the cryptocurrencies within last year as well as an overview about the coin.
7. **Display news**: The system shall display news about the cryptocurrencies and provide direct link to these articles.
8. **Display historical data**: The system shall display a table of history data for cryptocurrencies the table will contain the OCHLV values
9. **Display market exchanges**: The system will display data on the some of the available exchanges as well as provide a link to them
10. **Run trading simulation**: the system shall allow the user to train on a simulation that mimic how real market would work perform actions like buy and sell
11. **Search**: the system will allow user to search for a coin by name or by its symbol
12. **Subscribe to newsletter**: system will allow the user to subscribe to newsletter and receive emails about coins prices
13. **Add coins to watchlist**: system shall allow user to put specific coins to watchlist to show more interest in them for which emails and home page will be customed to.
14. **Convert cryptocurrencies:** The system shall allow users to convert cryptocurrencies into different currencies to know it values.

### 3.1.2 Non-functional requirements

**Usability**: An English literate should be able to use system within few seconds by reading the command and hints displayed on buttons and panels the visual representation of price movement will be at least 50\*50 pixels and explanation will be written under it.

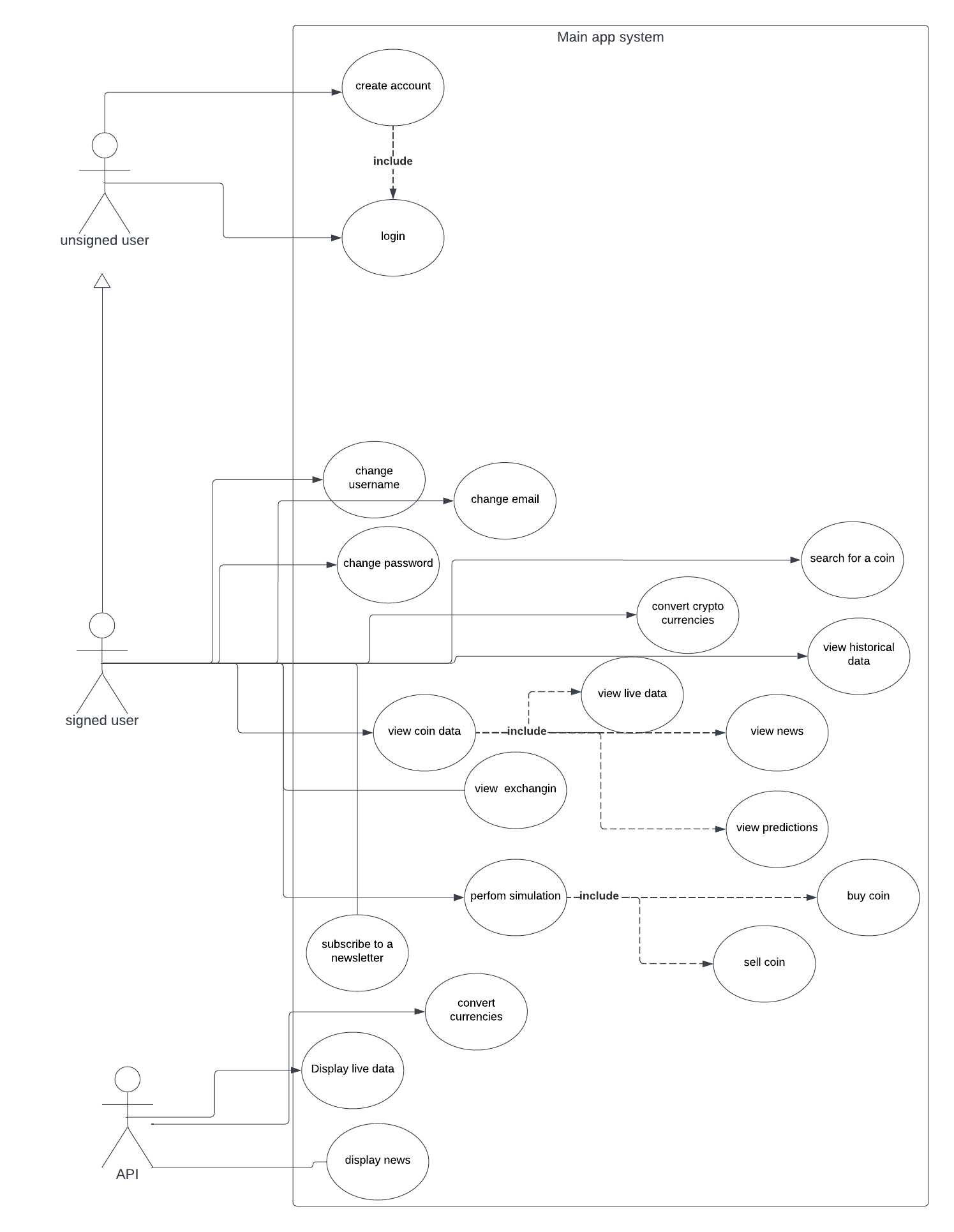
**Performance**: this software should take no longer than 5 seconds to display the pages and screens the predictions model should be executed and displayed in less than 10 seconds

**Maintainability**: the system must have the prediction execution model separated from the rest of the code and communicate with use of an API so if a better or a faster prediction model was found it can be replaced with it.

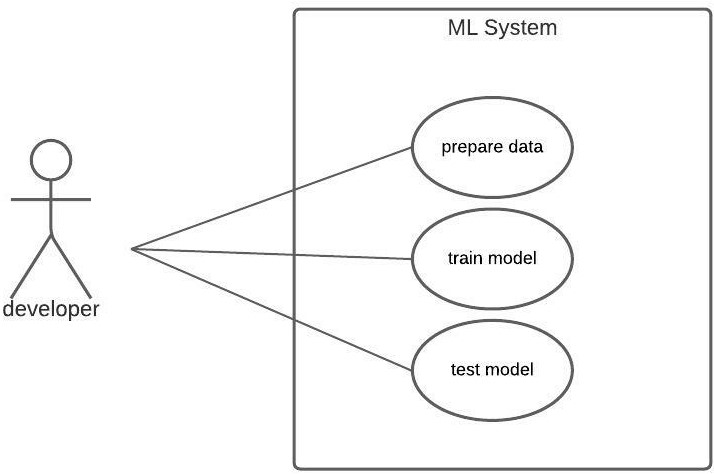
**Security**: the system shall encrypt the user passwords and will be stored in the database encrypted

**Accuracy**: This software will predict the right movement for a cryptocurrency with accuracy of at least 50%

## 3.2 Use Case Diagrams



Figure(13): Main app use cases



Figure(14): Developer with ML system use case

Diagram

Description automatically generated

Figure(15): Admin use case

|  |  |  |
| --- | --- | --- |
| Use Case ID: | UC1 | |
| Use Case Name: | Register | |
| Actors: | unsigned User | |
| Pre-conditions: | - | |
| Post-conditions: | A new user will be added to the system. | |
| Flow of events: | **Actor Action** | **System Action** |
| 1- User click in on register button. |  |
|  | 2-System ask user to enter his credentials (email, username, password) |
| 3-user enter his credentials |  |
|  | 4- System check if user already exists. |
| . | 5- System check password validity. |
|  | 6-System add user to the system account. |
|  | 7-- System sign into the user account. |
| Exceptional Flow of events: | **Actor Action** | **System Action** |
| At step 3- User enters an existing email. |  |
|  | 4- System notify the user that the user email already exists. |
| Exceptional Flow of events: | At step 3- User enters an existing email. |  |
|  | 4- System notify the user that password doesn’t match the required format. |

Table(2): Register use case

|  |  |  |
| --- | --- | --- |
| Use Case ID: | UC2 | |
| Use Case Name: | Sign in | |
| Actors: | unsigned User | |
| Pre-conditions: | should have an active account | |
| Post-conditions: | User will be signed in his account. | |
| Flow of events: | **Actor Action** | **System Action** |
| 1- User click on sign in button. |  |
|  | 2-System ask user to enter his login credentials |
| 3-user enter his credentials |  |
|  | 4- System validate user login info. |
|  | 5- System load user data from database |
|  | 6- System sign into the user account. |
| Exceptional Flow of events: | **Actor Action** | **System Action** |
| At step 3- User enters wrong credentials (email or password). |  |
|  | 4- System notify the user that username or password is wrong and should re-enter |

Table(3): Sign in use case

|  |  |  |
| --- | --- | --- |
| Use Case ID: | UC3 | |
| Use Case Name: | Log out | |
| Actors: | signed User | |
| Pre-conditions: | should have an active account and signed in | |
| Post-conditions: | User will log out of his account | |
| Flow of events: | **Actor Action** | **System Action** |
| 1- User click on log out button. |  |
|  | 2-System log out the user from his account |

Table(4): Log out use case

|  |  |  |
| --- | --- | --- |
| Use Case ID: | UC4 | |
| Use Case Name: | Update user data | |
| Actors: | signed User | |
| Pre-conditions: | should have an active account and signed in | |
| Post-conditions: | User account info will be updated by the new values | |
| Flow of events: | **Actor Action** | **System Action** |
| 1- User click on update button. |  |
|  | 2-System show the user options for info to change |
| 3-user choose change budget |  |
|  | 4- System ask user to enter the value. |
| 5- user enter the new value |  |
|  | 6- System update user’s info in database |
| Alternate Flow of events: | **Actor Action** | **System Action** |
| At step 3- user choose update ownership |  |
|  | 4- System show the user the available cryptocurrencies |
| 5- user choose a cryptocurrency |  |
|  | 6- System ask user to enter the value. |
| 7- user enter the new value |  |
|  | 8- System update user’s info in database |
| Exceptional Flow of events: | **Actor Action** | **System Action** |
| At step 5 - user enter an invalid value |  |
|  | 6- System notify user that value is invalid |
| Extends | Update budget, update owned cryptocurrencies | |

Table(5): update user data use case

|  |  |  |
| --- | --- | --- |
| Use Case ID: | UC5 | |
| Use Case Name: | Display cryptocurrency live information | |
| Actors: | signed User | |
| Pre-conditions: | should have an active account and signed in | |
| Post-conditions: | User will view a live info for the cryptocurrency as well as the prediction for its future values | |
| Flow of events: | **Actor Action** | **System Action** |
| 1- User click on home button. |  |
|  | 2-System calls the live cryptocurrencies data API |
|  | 3-System filter needed data from the retrieved data put it in the right format |
|  | 4- System display the new data for the user |
|  | 5-System retrieve old data from the database for each available cryptocurrency |
|  | 6- System calls the Prediction algorithm API |
|  | 7-System retrieve the predicted price movements from the API |
|  | 8- System display the predicted price movements |
| Includes: | Retrieve live data, predict cryptocurrency price movement | |

Table(6): Display crypto live information use case

|  |  |  |
| --- | --- | --- |
| Use Case ID: | UC6 | |
| Use Case Name: | Send new coins data to database | |
| Actors: | Admin | |
| Pre-conditions: | should have an active account and signed in | |
| Post-conditions: | New coins’ data will be added to database | |
| Flow of events: | **Actor Action** | **System Action** |
| 1- User click on update database button. |  |
|  | 2-System prepare the coins models |
|  | 3-System check database data if it has same date |
|  | 4- System loop over the coins and insert each of them |
| Alternate Flow of events: | **Actor Action** | **System Action** |
|  | At step 3- System check database data if it has same date and find data with same date |
|  | 4- System display a warning to the user that data in database has same date and ask if he would like to overwrite it. |
| 5- user confirm the update. |  |
|  | 6- System overwrite the old data. |

Table(7): Send coins to database use case

|  |  |  |
| --- | --- | --- |
| Use Case ID: | UC7 | |
| Use Case Name: | Overview coin data | |
| Actors: | signed User | |
| Pre-conditions: | should have an active account and signed in | |
| Post-conditions: | The cryptocurrency data will be visualized | |
| Flow of events: | **Actor Action** | **System Action** |
| 1- User click on overview button for a coin. |  |
|  | 2-system retrieve coin data for the previous months from database. |
|  | 3-System build the graph for the coin data and show general info about the coin. |
|  | 4-system display the graph for the user. |

Table(8): Visualize data use case

|  |  |  |
| --- | --- | --- |
| Use Case ID: | UC8 | |
| Use Case Name: | view recommended investments | |
| Actors: | signed User | |
| Pre-conditions: | should have an active account and signed in | |
| Post-conditions: | User will log out of his account | |
| Flow of events: | **Actor Action** | **System Action** |
| 1- User click on recommendations button. |  |
|  | 2-System retrieve previous cryptocurrency data from data base |
|  | 3-System calls the predictor API and send to it the data and user info |
|  | 4- System run the recommendation Algorithm. |
|  | 5- System predictor API returns the recommendations |
|  | 6- System display recommendation to the user |
| Includes | Recommend investments | |

Table(9): View recommended investments use case

|  |  |  |
| --- | --- | --- |
| Use Case ID: | UC9 | |
| Use Case Name: | Buy coin in live simulation | |
| Actors: | signed User | |
| Pre-conditions: | should have an active account, signed in and have enough amount of money | |
| Post-conditions: | A pop message saying transaction successful will appear | |
| Flow of events: | **Actor Action** | **System Action** |
| 1- User click on buy button for a coin |  |
| 2-User will determine the Quantity he wishes to buy |  |
|  | 3-System will check if he has money |
|  | 4- System will update the database for the money and the owned coins |
|  | 5-System will send a “Transaction successful” to the user |
| Alternate Flow of events: | **Actor Action** | **System Action** |
|  |  | 3-System checks if he has money |
|  |  | 4- System will send “Not enough money” to the user |

Table(10): Buy coin in live simulation use case

|  |  |  |
| --- | --- | --- |
| Use Case ID: | UC10 | |
| Use Case Name: | sell coin in live simulation | |
| Actors: | signed User | |
| Pre-conditions: | should have an active account, signed in and have enough Quantity of crypto coin | |
| Post-conditions: | A pop message saying transaction successful will appear | |
| Flow of events: | **Actor Action** | **System Action** |
| 1- User click on sell button for a coin |  |
| 2-User will determine the Quantity he wishes to sell |  |
|  | 3-System will check if he the Quantity  he wishes to sell |
|  | 4- System will update the database for the money and the owned coins |
|  | 5-System will send a “Transaction successful” to the user |
| Alternate Flow of events: | **Actor Action** | **System Action** |
|  |  | 3-System will check if he the Quantity  he wishes to sell |
|  |  | 4- System will send “Not enough money” to the user |

Table(11): Sell coin in live simulation use case

# 4. System Design

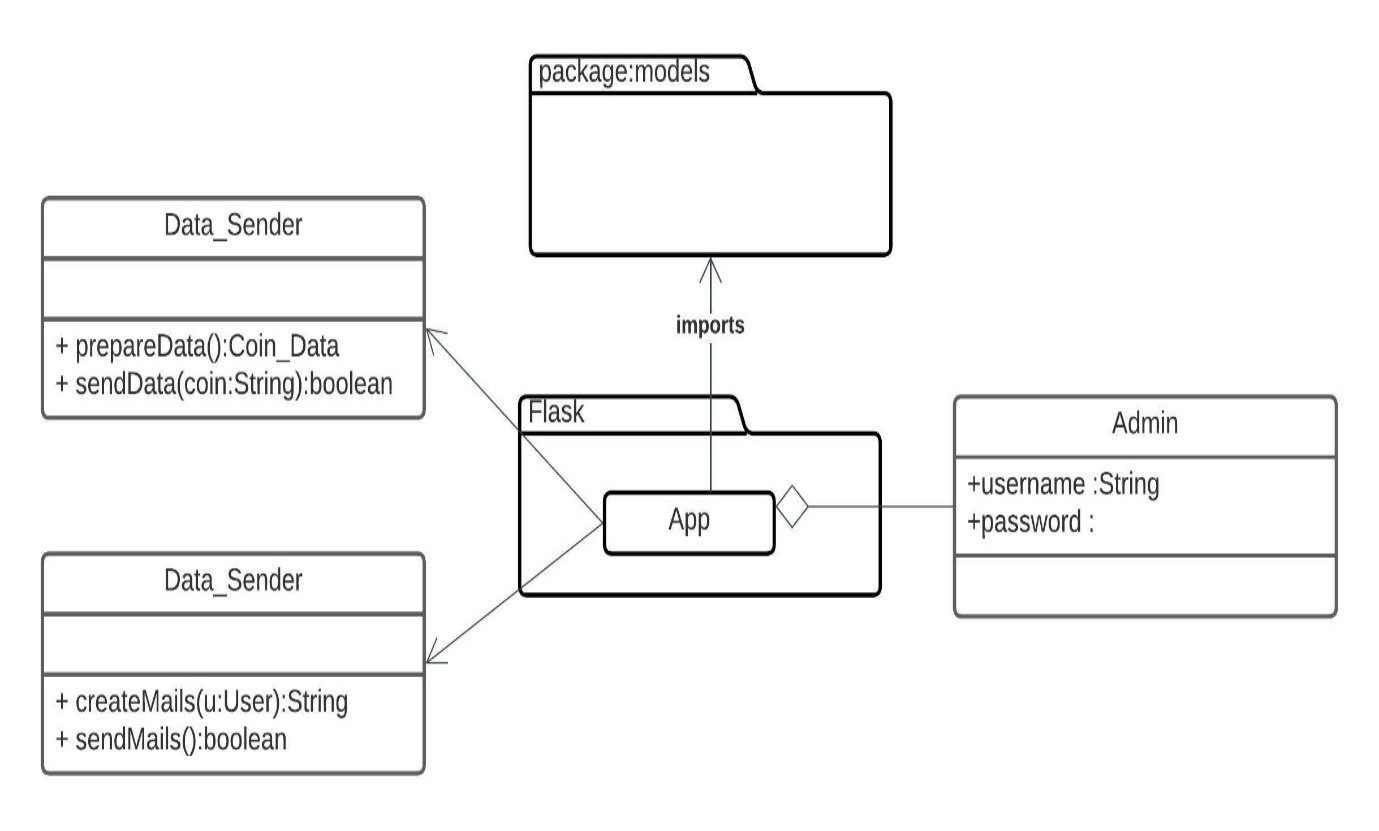
## 4.1 System component diagram

Diagram, schematic

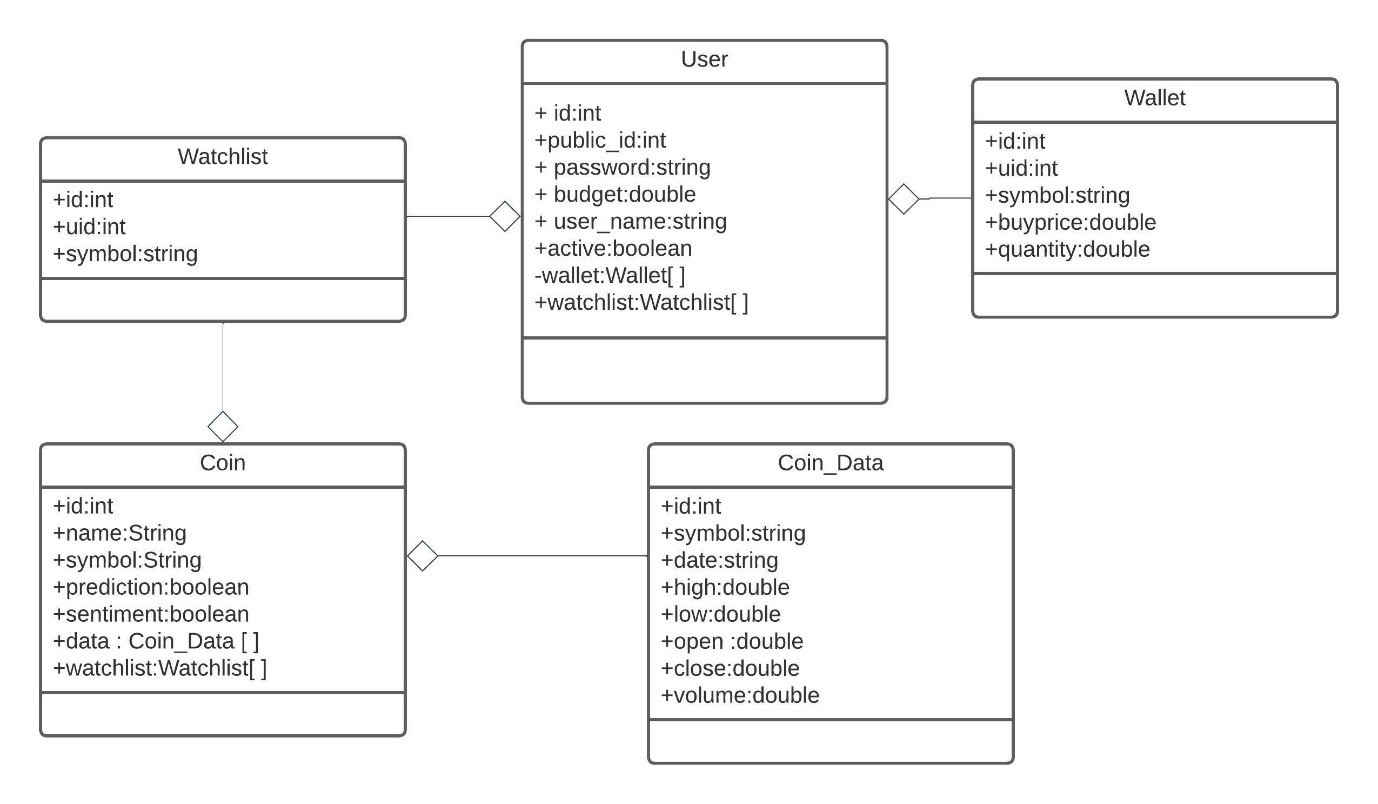
Description automatically generated

Figure(16): System component diagram

## 4.2 System class diagrams



Figure(17): class diagram



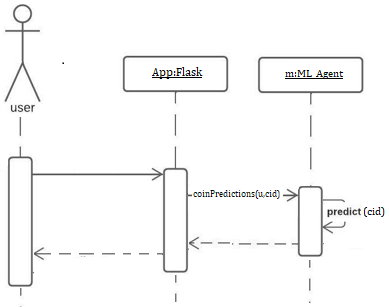
Figure(18): class diagram

Diagram

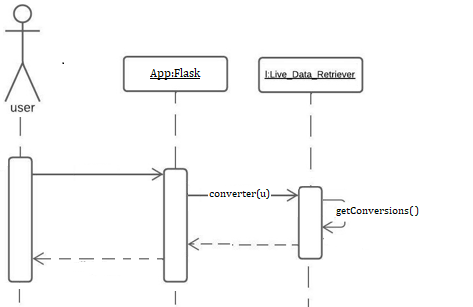
Description automatically generated

Figure(19):class diagram

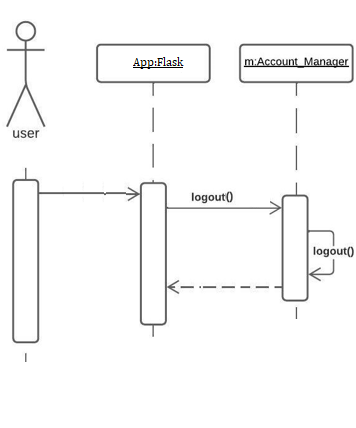
## 4.3 Sequence diagrams



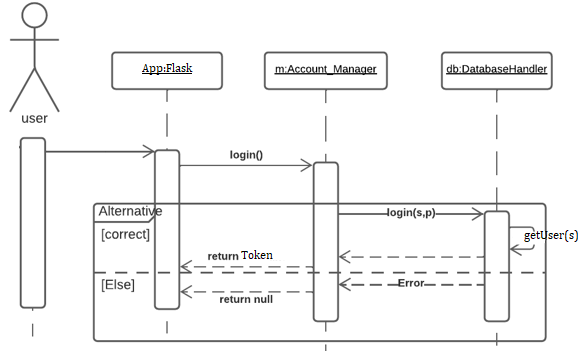
Figure(20)



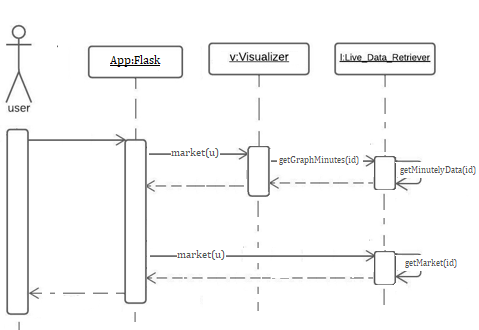
Figure(21)



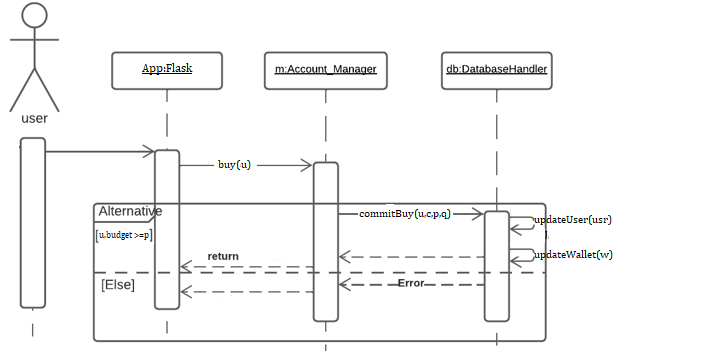
Figure(22)



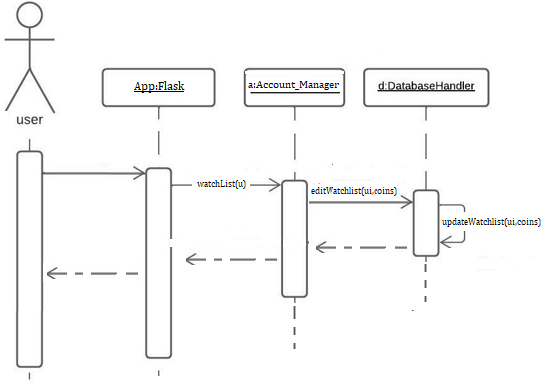
Figure(23)



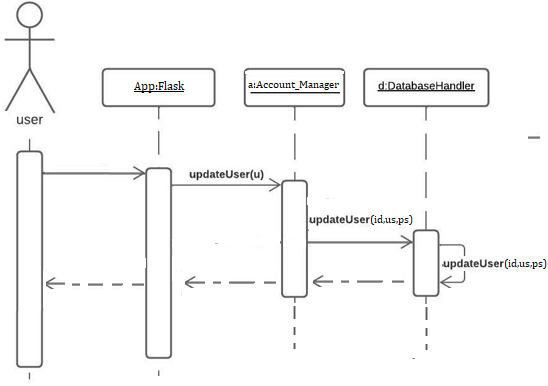
Figure(24)



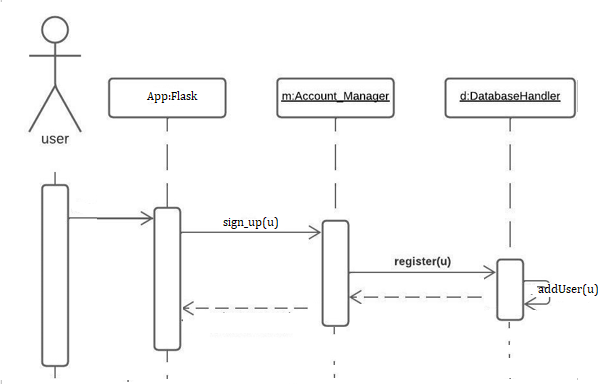
Figure(25)



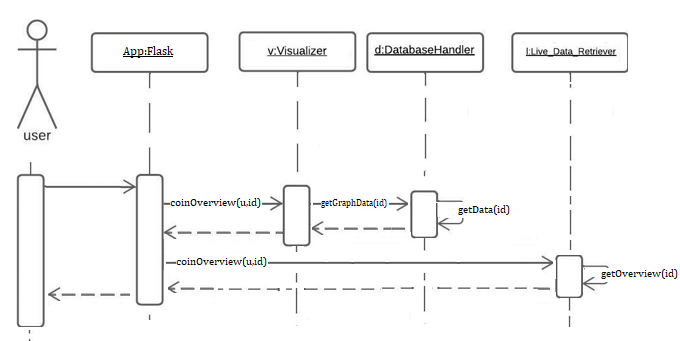
Figure(26)



Figure(27)



Figure(28)



Figure(29)

## 4.4 Project ERD

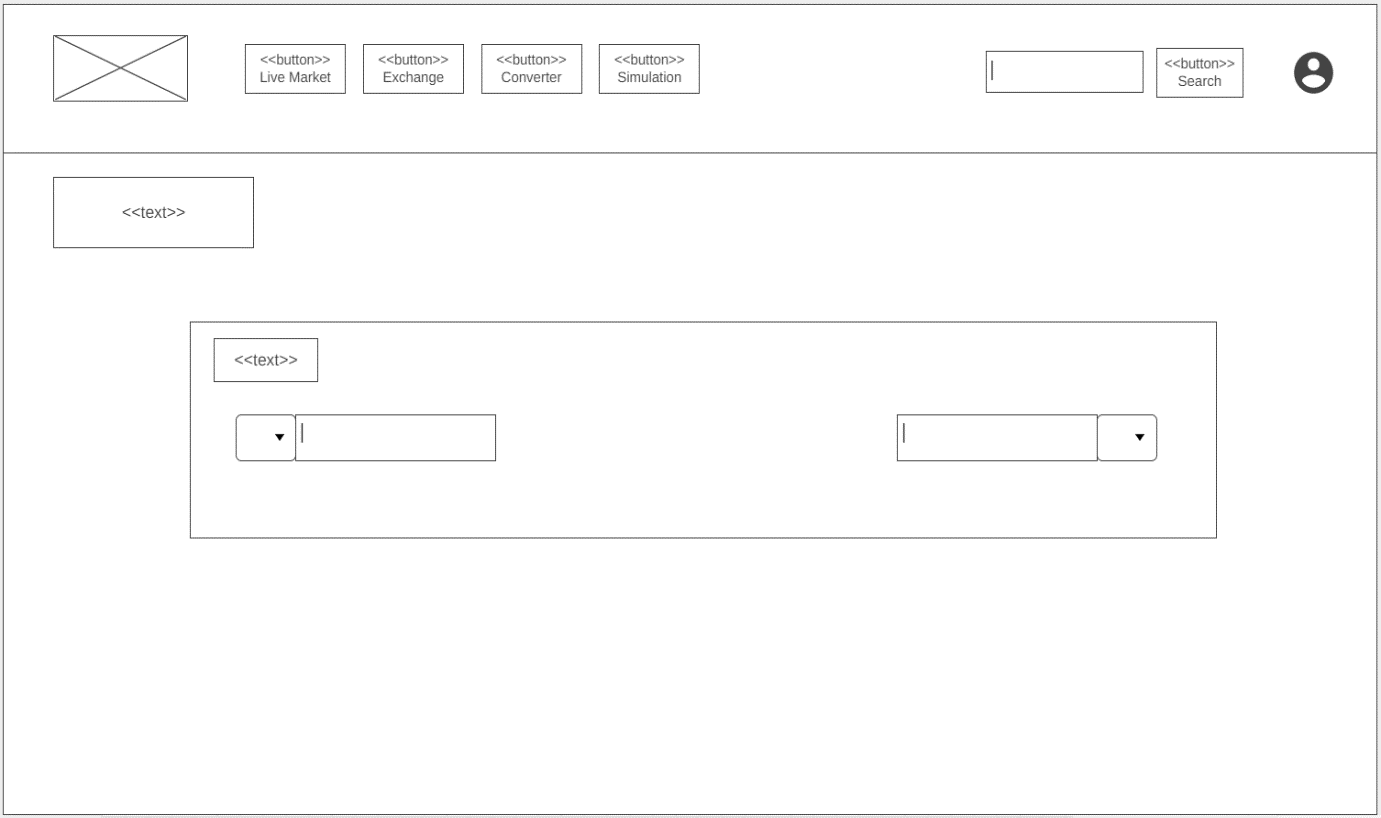
Diagram

Description automatically generated

Figure(30): ERD diagram

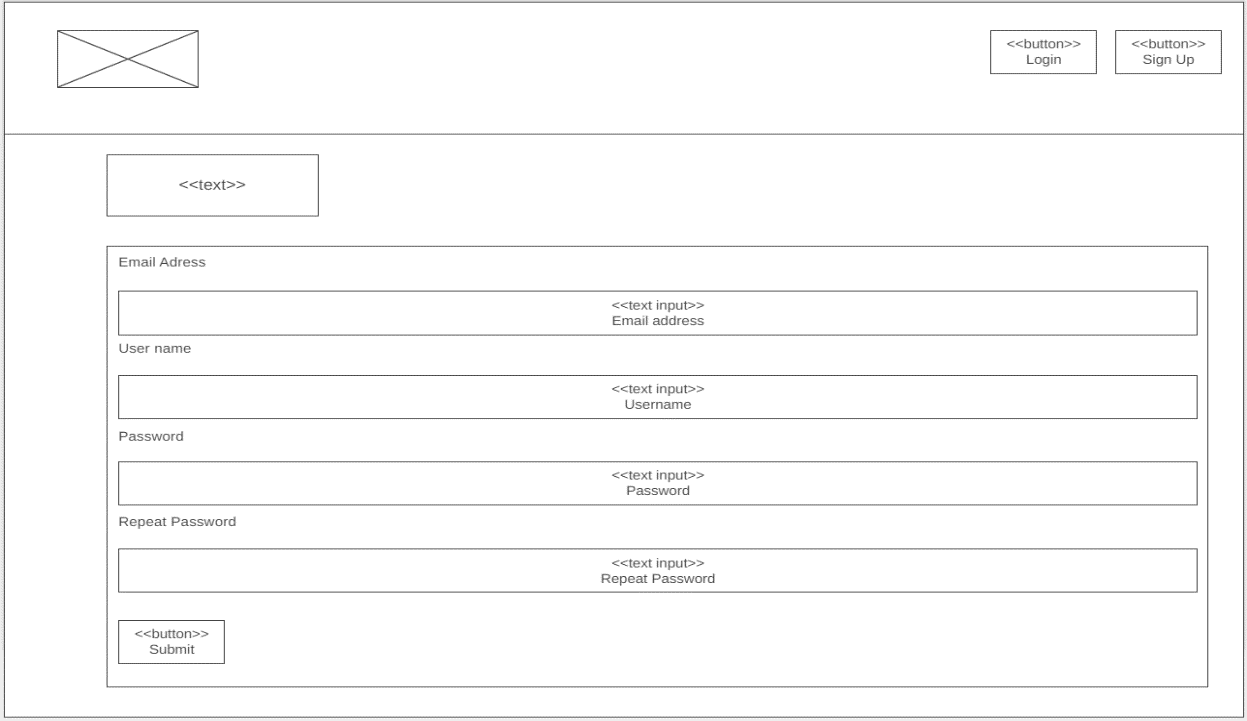
## 4.5 System GUI Design

Converter



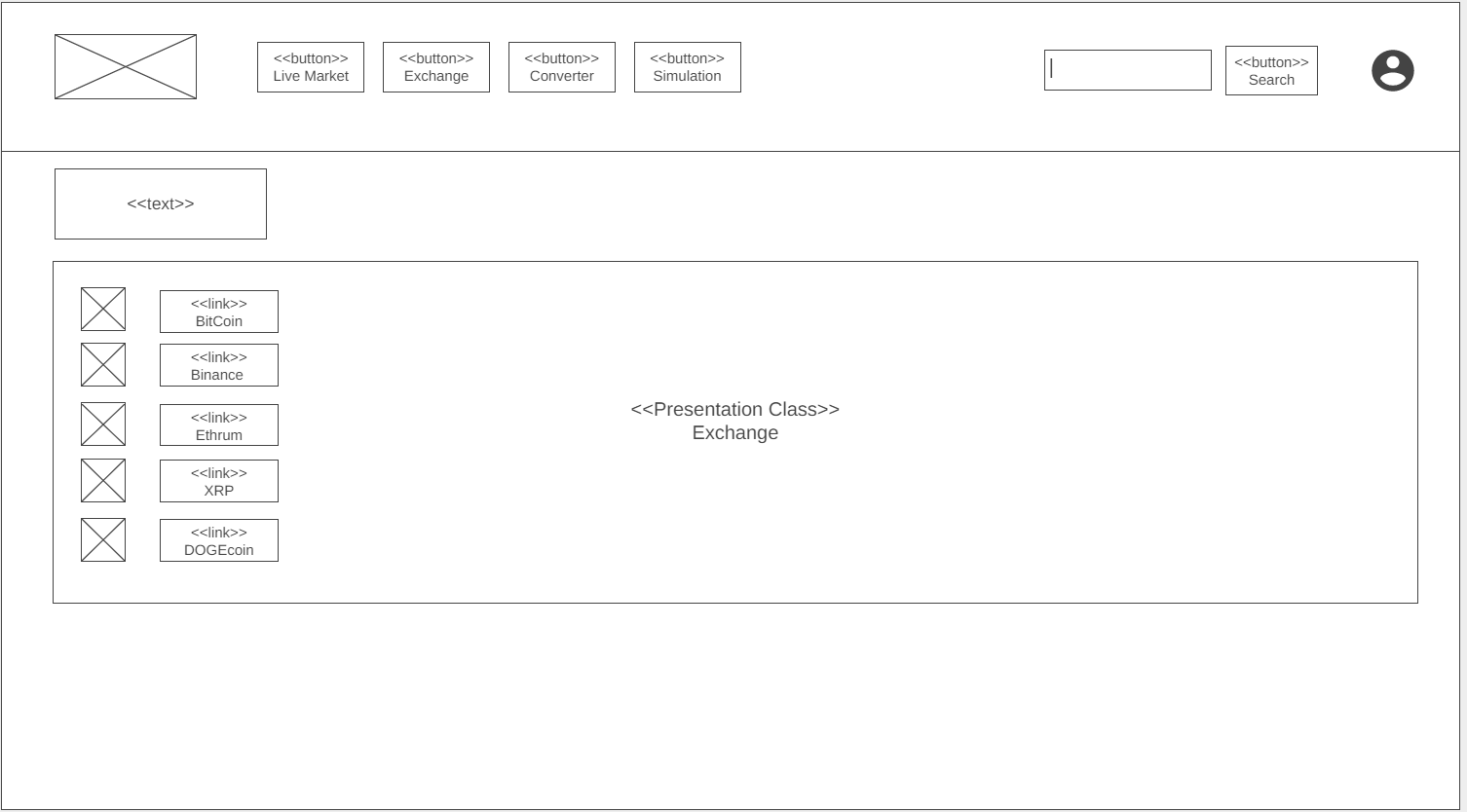
Figure(31): The UI design of converter page

SignUp



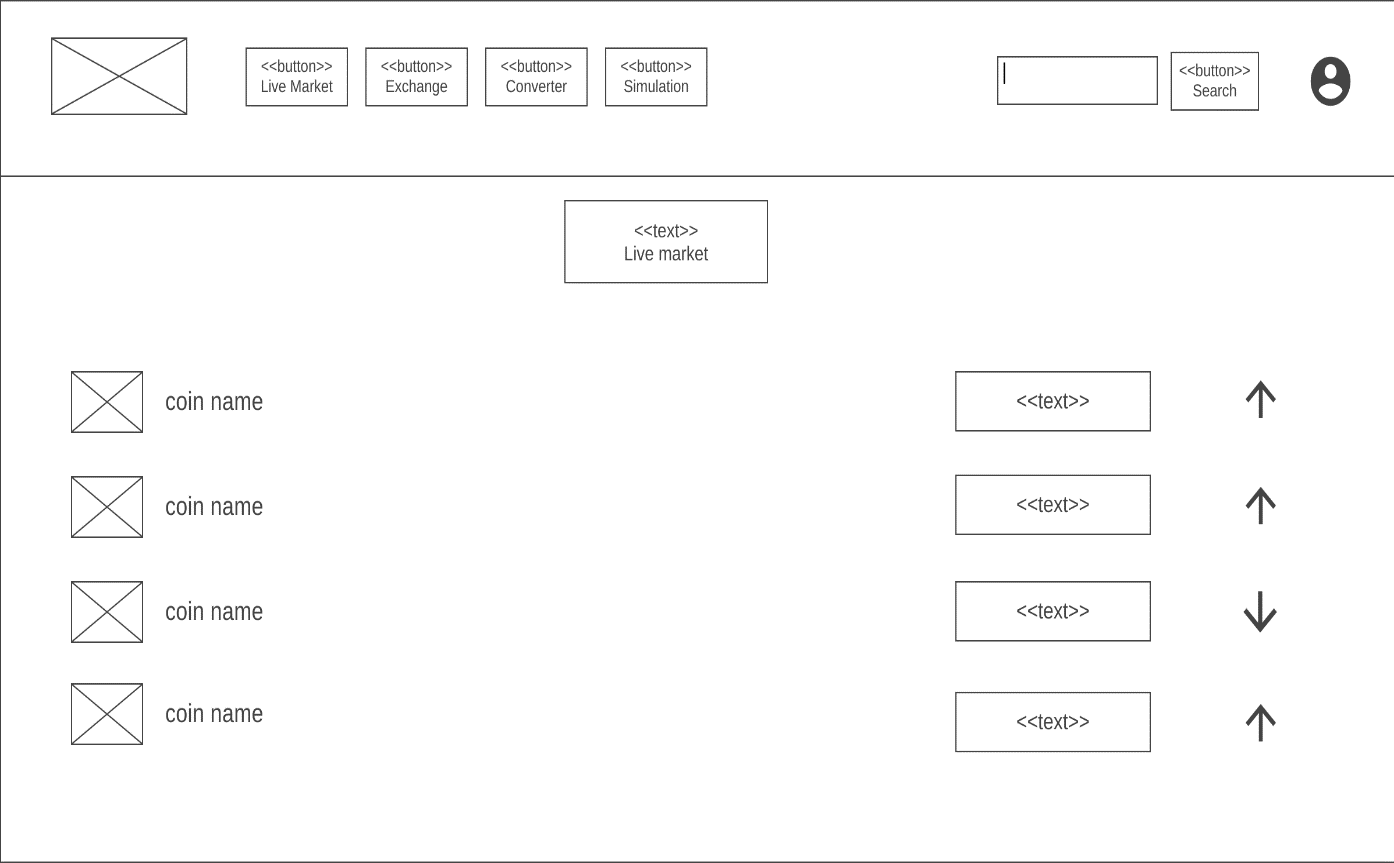
Figure(32): The UI design of signup page

Exchange



Figure(33): The UI design of exchange page

Live market



Figure(34): The UI design of live market page

Simulation

Graphical user interface, table

Description automatically generated

Figure(35): The UI design of simulation page

Overview

A picture containing chart

Description automatically generated

Figure(36): The UI design of overview page

News

Graphical user interface

Description automatically generated

Figure(37): The UI design of news page

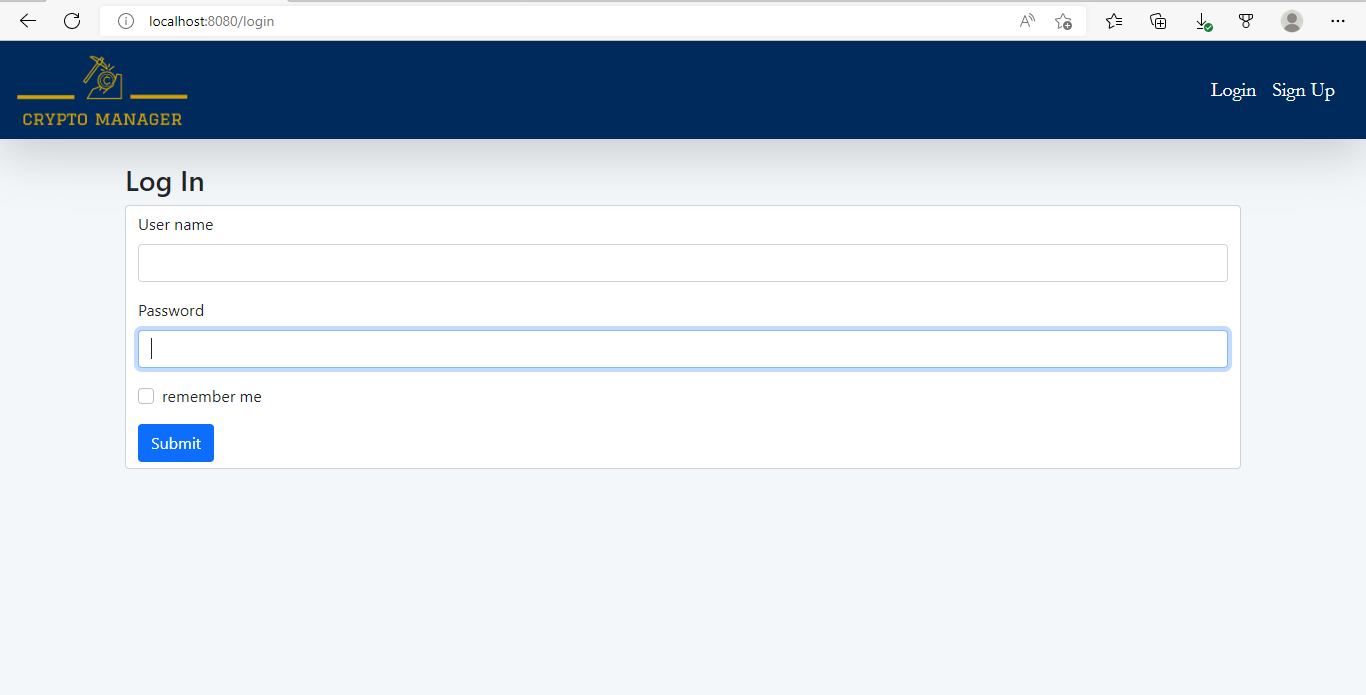
# 5.Implementation and Testing

## 5.1 implementation

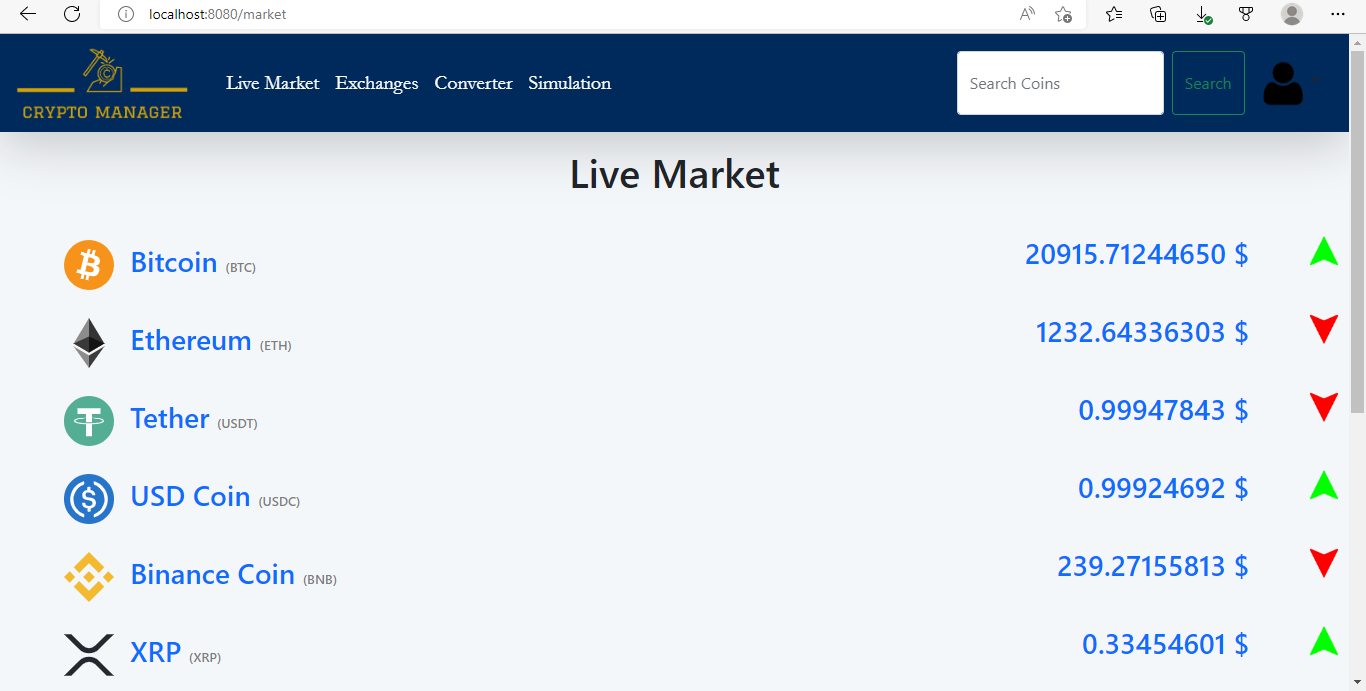
Graphical user interface, text, application

Description automatically generated

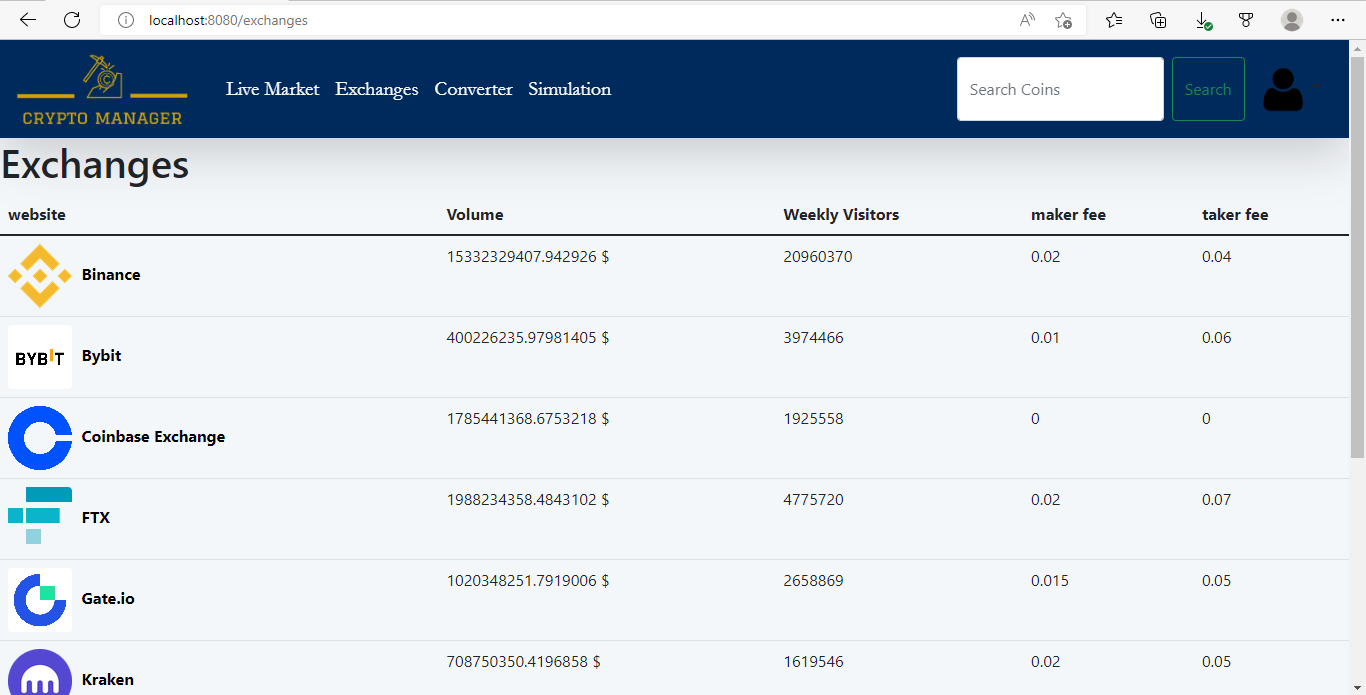
Figure(38): The sign up page



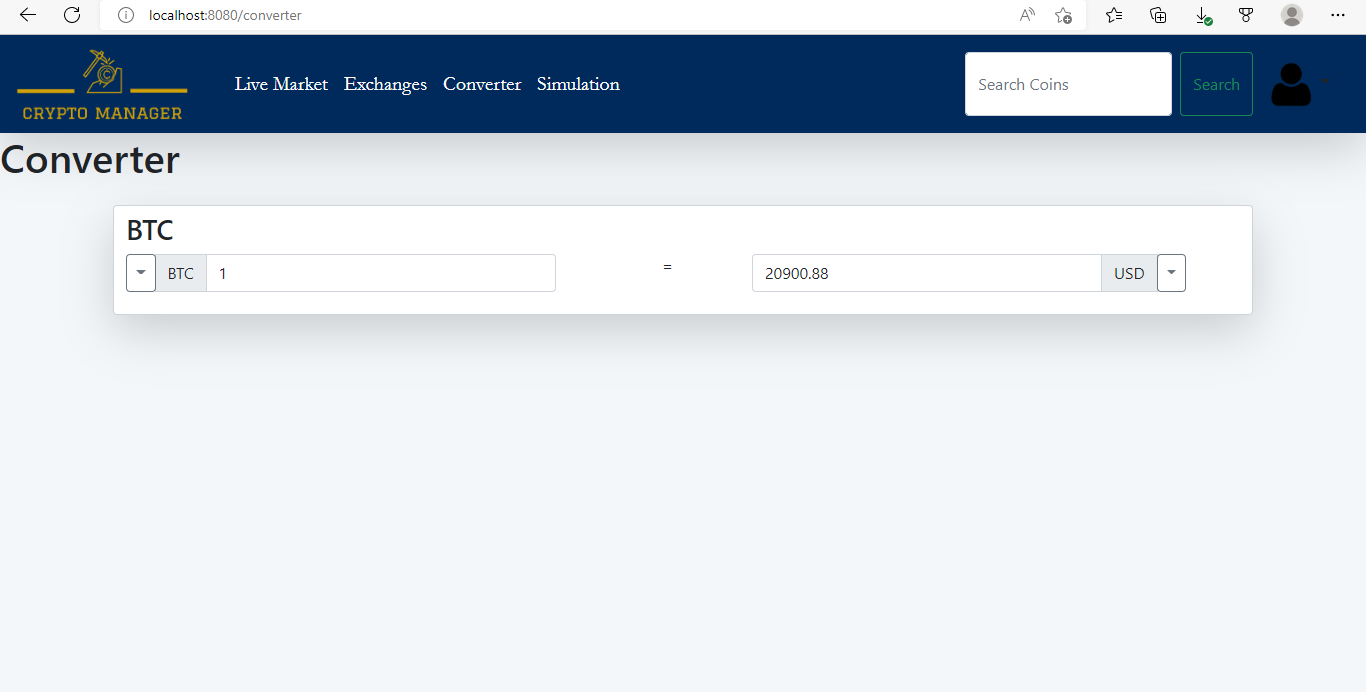
Figure(39): The login page



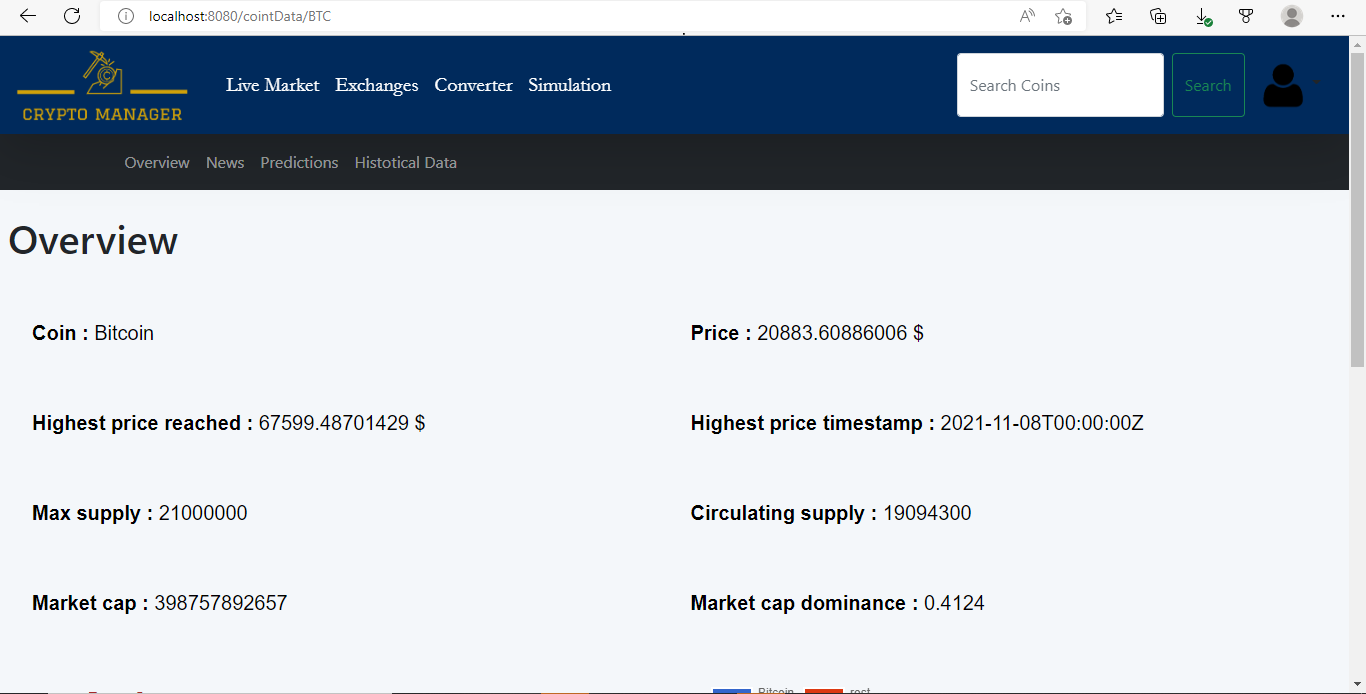
Figure(40): The live market page



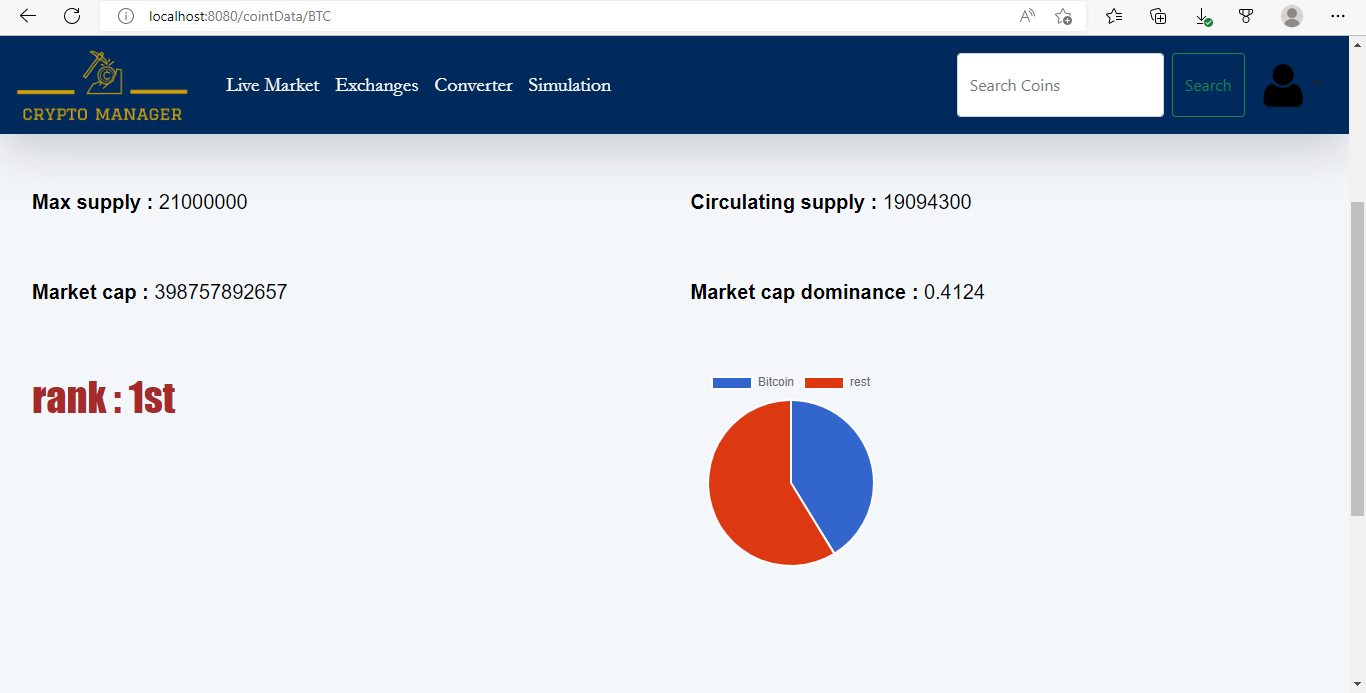
Figure(41): The exchanges page



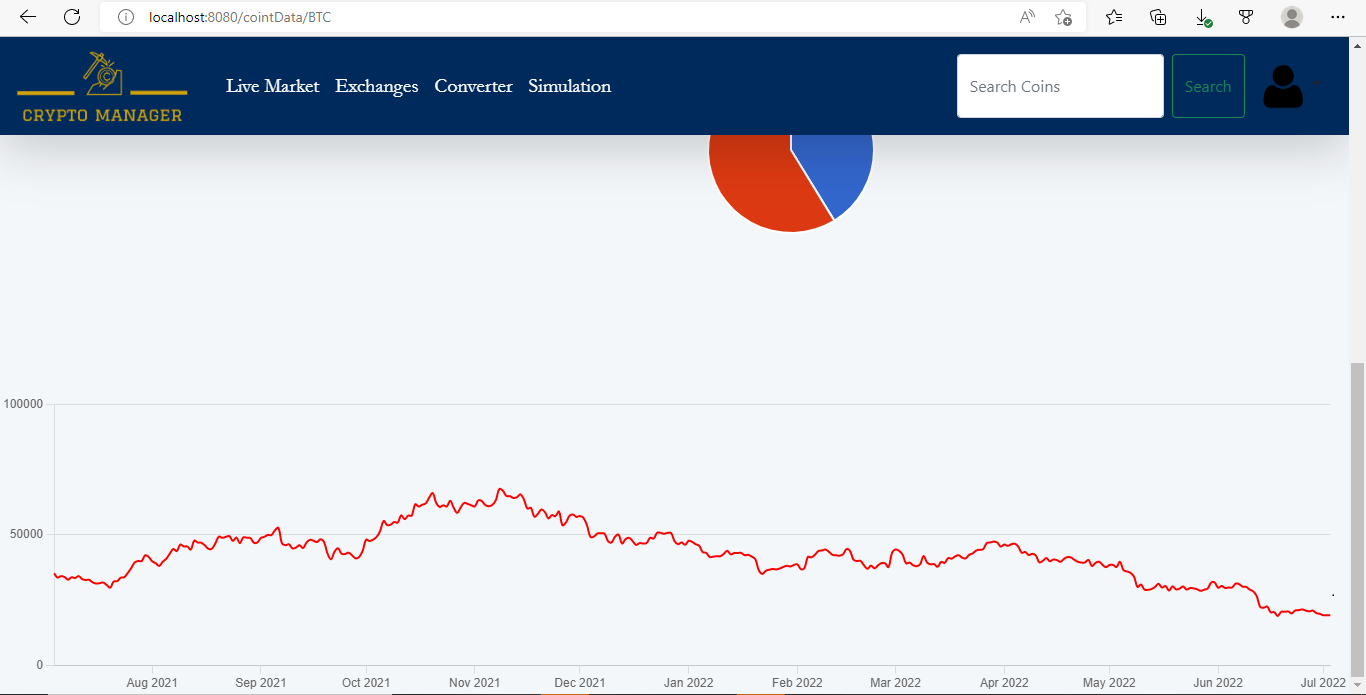
Figure(42): The coin converter page



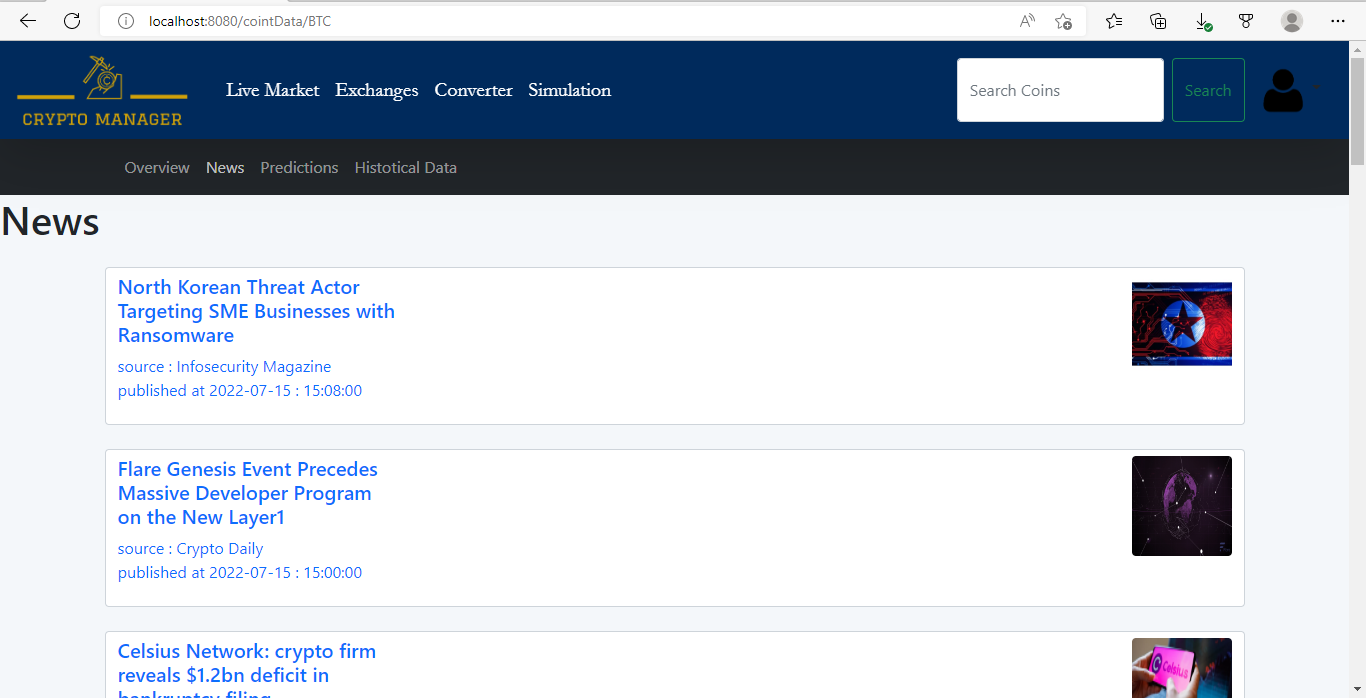
Figure(43): The coin overview page



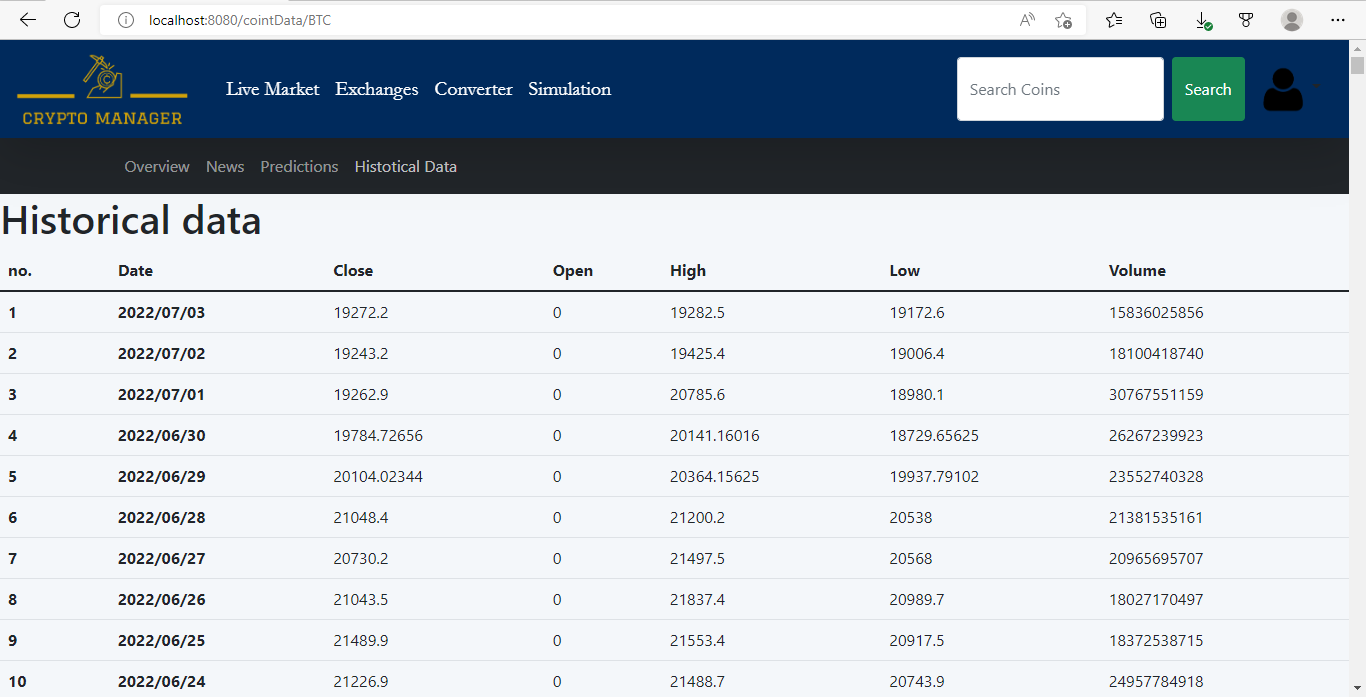
Figure(44): The coin overview page(con)



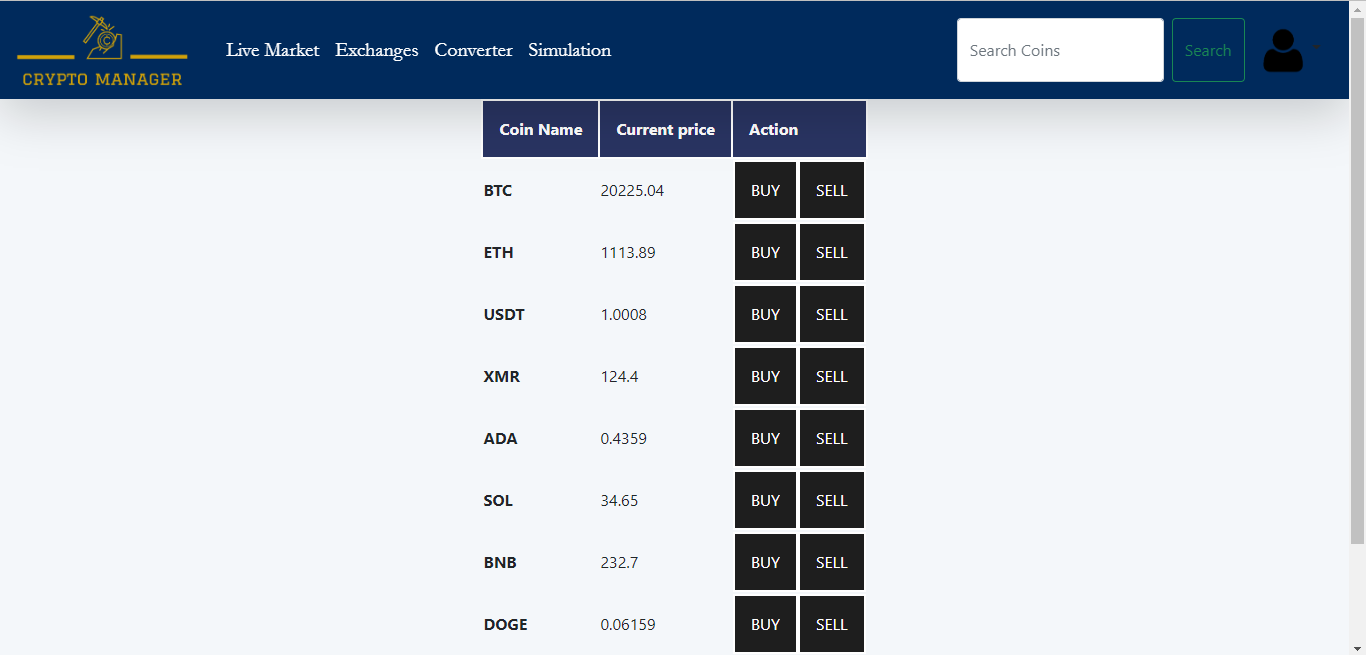
Figure(45): The coin overview page(con)



Figure(46): The news page



Figure(47) The coin historical data page



Figure(48): The simulation page

## 5.2 Testing

System Testing

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Name:** | | | | **Crypto Manager** | | | | | |
| **Module Name:** | | | | **Signup** | | | | | |
| **Test Case\_ID** | **Test Case Description** | **Test steps** | **Pre-Conditions** | | **Test Data** | **Post-Condition** | **Expected Output** | **Actual Output** | **State** |
| TC\_Signup\_001 | Valid email  Valid username  Valid password1  Valid password2 | User will click on the signup button | - | | Email: [andrew12@gmail.com](mailto:andrew12@gmail.com)  Username: Andrew  Password1:12345678  Password2:12345678 | A user account will be created | User created | User created | Passed |
| TC\_Signup \_002 | Valid email  Valid username  Valid password1  Valid password2 | User will click on the signup button | Email already exists | | Email: [andrew12@gmail.com](mailto:andrew12@gmail.com)  Username: Andrew  Password1:12345678  Password2:12345678 | Error message will appear to the user | Email already exists | Email already exists | Passed |
| TC\_Signup \_003 | Valid email  Valid username  Valid password1  Valid password2 | User will click on the signup button | Username already exists | | Email: [omar189@gmail.com](mailto:omar189@gmail.com)  Username: Omar  Password1: omar1234  Password2: omar1234 | Error message will appear to the user | Username already exists | Username already exists | Passed |
| TC\_Signup \_004 | Invalid email  Valid username  Valid password1  Valid password2 | User will click on the signup button | - | | Email: [ahmed@gmail.com](mailto:ahmed@gmail.com)  Username: Ahmed  Password1:12345678  Password2:12345678 | Error message will appear to the user | Email must be greater than 7 characters | Email must be greater than 7 characters | Passed |
| TC\_Signup \_005 | Valid email  Invalid username  Valid password1  Valid password2 | User will click on the signup button | - | | Email: [andrew12@gmail.com](mailto:andrew12@gmail.com)  Username: Ali  Password1:12345678  Password2:12345678 | Error message will appear to the user | Username must be greater than 3 characters | Username must be greater than 3 characters | Passed |
| TC\_Signup\_006 | Valid email  Valid username  Valid password1  Invalid password2 | User will click on the signup button | - | | Email: [mohamed@gmail.com](mailto:mohamed@gmail.com)  Username: Mohamed  Password1:12345678  Password2:12345678 | Error message will appear to the user | Password don’t match | Password don’t match | Passed |
| TC\_Signup\_007 | Valid email  Valid username  Invalid password1  Valid password2 | User will click on the signup button | - | | Email: [mohamed@gmail.com](mailto:mohamed@gmail.com)  Username: Mohamed  Password1:123456  Password2:123456 | Error message will appear to the user | Password must be at least 7 characters | Password must be at least 7 characters | passed |

Table(12): signup test cases

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Name:** | | | | **Crypto Manager** | | | | | |
| **Module Name:** | | | | **Login** | | | | | |
| **Test Case\_ID** | **Test Case Description** | **Test Steps** | **Pre-Conditions** | | **Test Data** | **Post-Conditions** | **Expected Output** | **Actual Output** | **State** |
| TC\_Login\_001 | Valid username  Valid password | User will click on the login button | User must have a username and password | | Username: Andrew  Password:12345678 | User can navigate through the website | Logged in successfully | Logged in successfully | Passed |
| TC\_Login\_002 | Valid username  Valid password | User will click on the login button | User must have a username and password | | Username: Ahmed  Password: ahmed123 | Error message will appear to the user | Incorrect password | Incorrect password | Passed |
| TC\_Login\_003 | Valid username  Valid password | User will click on the login button | Username already exists | | Username: Omar  Password: omar1234 | Error message will appear to the user | Username does not exist | Username does not exist | Passed |

Table(13): login test cases

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Name:** | | | **Crypto Manager** | | | | | | |
| **Module Name:** | | | **Logout** | | | | | | |
| **Test Case\_ID** | **Test Case Description** | **Test Steps** | | **Pre-Conditions** | **Test Data** | **Post-Conditions** | **Expected Output** | **Actual Output** | **State** |
| TC\_Logout\_001 | Valid username  Valid password | User will click on the logout button | | User must  be logged in |  | User will be redirected to the login page |  |  | Passed |

Table(14): logout test cases

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Name:** | | | | **Crypto Manager** | | | | | |
| **Module Name:** | | | | **buyCoin** | | | | | |
| **Test Case\_ID** | **Test Case Description** | **Test Steps** | **Pre-Conditions** | | **Test Data** | **Post-Conditions** | **Expected Output** | **Actual Output** | **State** |
| TC\_buyCoin\_001 | Invalid coinQuantity | User will click on the buy button | User must be logged in | | coinQuantity<=0 |  | Invalid input | Invalid input | Passed |
| TC\_buyCoin \_002 | Valid coinQuantity | User will click on the buy button | User must be logged in | | 0<coinQuantity<=maximum\_quantity |  | Transaction done Successfully | Transaction done Successfully | Passed |
| TC\_buyCoin \_003 | Invalid coinQuantity | User will click on the buy button | User must be logged in | | coinQuantity>maximum\_quantity |  | Not enough Money | Not enough Money | Passed |

Table(15): buy test cases

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Name:** | | | **Crypto Manager** | | | | | | |
| **Module Name:** | | | **sellCoin** | | | | | | |
| **Test Case\_ID** | **Test Case Description** | **Test Steps** | | **Pre-Conditions** | **Test Data** | **Post-Conditions** | **Expected Output** | **Actual Output** | **State** |
| TC\_sellCoin\_001 | Invalid coinQuantity | User will click on the sell button | | User must be logged in | coinQuantity<=0 |  | Invalid input | Invalid input | Passed |
| TC\_sellCoin \_002 | Valid coinQuantity | User will click on the sell button | | User must be logged in | 0<coinQuantity<= Wallet\_quantity |  | Transaction done Successfully | Transaction done Successfully | Passed |
| TC\_sellCoin \_003 | Invalid coinQuantity | User will click on the sell button | | User must be logged in | coinQuantity>Wallet\_quantity |  | Not enough Coin | Not enough Money | Passed |

Table(16): sell test cases

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Name:** | | | **Crypto Manager** | | | | | | |
| **Module Name:** | | | **market** | | | | | | |
| **Test Case\_ID** | **Test Case Description** | **Test Steps** | | **Pre-Conditions** | **Test Data** | **Post-Conditions** | **Expected Output** | **Actual Output** | **State** |
| TC\_market\_001 |  | User will click on the market button | | User must be logged in |  | User will observe live crypto data |  |  | Passed |

Table(17): market test case

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Name:** | | | **Crypto Manager** | | | | | | |
| **Module Name:** | | | **coinNews** | | | | | | |
| **Test Case\_ID** | **Test Case Description** | **Test Steps** | | **Pre-Conditions** | **Test Data** | **Post-Conditions** | **Expected Output** | **Actual Output** | **State** |
| TC\_coinNews\_001 |  | User will click on the news button in the market page | | User must be logged in and in the live market page |  | User will be able to see news about the cryptocurrencies |  |  | Passed |

Table(18): news test case

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Name:** | | | **Crypto Manager** | | | | | | |
| **Module Name:** | | | **coinData** | | | | | | |
| **Test Case\_ID** | **Test Case Description** | **Test Steps** | | **Pre-Conditions** | **Test Data** | **Post-Conditions** | **Expected Output** | **Actual Output** | **State** |
| TC\_coinData\_001 |  | User will click on the Historical data button | | User must be logged in and in the live market page |  | User will observe the historical crypto data (high, low, etc.) |  |  | Passed |

Table(19): coindata test case

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Name:** | | | **Crypto Manager** | | | | | | |
| **Module Name:** | | | **coinPredection** | | | | | | |
| **Test Case\_ID** | **Test Case Description** | **Test Steps** | | **Pre-Conditions** | **Test Data** | **Post-Conditions** | **Expected Output** | **Actual Output** | **State** |
| TC\_coinPredection \_001 |  | User will click on the Predictions button | | User must be logged in and in the live market page |  | User will be able to see the predicted values for different crypto coins |  |  | Passed |

Table(20): coinPrediction test case

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Name:** | | | **Crypto Manager** | | | | | | |
| **Module Name:** | | | **coinOverview** | | | | | | |
| **Test Case\_ID** | **Test Case Description** | **Test Steps** | | **Pre-Conditions** | **Test Data** | **Post-Conditions** | **Expected Output** | **Actual Output** | **State** |
| TC\_coinOverview\_001 |  | User will click on the overview button so that he can navigate through it | | User must be logged in and in the market page |  | User will be able to see information about the crypto (price, highest price reached, Market cap, max supply, graph about the coin, etc.) |  |  | Passed |

Table(21): coinoverview test case

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Name:** | | | **Crypto Manager** | | | | | | |
| **Module Name:** | | | **profile** | | | | | | |
| **Test Case\_ID** | **Test Case Description** | **Test Steps** | | **Pre-Conditions** | **Test Data** | **Post-Conditions** | **Expected Output** | **Actual Output** | **State** |
| TC\_profile\_001 |  | User will click on the Profile button | | User must be logged in |  | User will be able to see his profile (budget, Quantity, etc.) |  |  | Passed |

Table(22): profile test case

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Name:** | | | **Crypto Manager** | | | | | | |
| **Module Name:** | | | **Converter** | | | | | | |
| **Test Case\_ID** | **Test Case Description** | **Test Steps** | | **Pre-Conditions** | **Test Data** | **Post-Condition** | **Expected Output** | **Actual Output** | **State** |
| TC\_Converter\_001 | Valid input | User enter a quantity of crypto currency | | User must be logged in | Quantity=-1  coin Name=BTC | User will observe a change in currency | USD: 19649.88 | USD: 19649.88 | Passed |
| TC\_Converter\_002 | Valid input | User enter a quantity of crypto currency | | User must be logged in | Quantity=0  coin Name=ETH | User will not  observe a change in currency | USD: 0 | USD: 0 | Passed |
| TC\_Converter\_003 | Valid input | User enter a quantity of crypto currency | | User must be logged in | Quantity=1  coin Name=BNB | User will  observe a change in currency | USD: 224.82 | USD: 224.82 | Passed |
| TC\_Converter\_004 | Valid input | User enter a quantity of crypto currency | | User must be logged in | Quantity=BTC  coin  name=BTC | Error pop up message | Nan | Nan | Passed |

Table(23): converter test cases

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Name:** | | | **Crypto Manager** | | | | | | |
| **Module Name:** | | | **Exchange** | | | | | | |
| **Test Case\_ID** | **Test Case Description** | **Test Steps** | | **Pre-Conditions** | **Test Data** | **Post-Conditions** | **Expected Output** | **Actual Output** | **State** |
| TC\_exchange\_001 |  | User will click on the exchange button | | User must be logged in |  | User will be able to see the (volume, weekly visitors, maker fees, taker fees) for each coin |  |  | Passed |

Table(24): exchange test case

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Name:** | | | **Crypto Manager** | | | | | | |
| **Module Name:** | | | **newsletter** | | | | | | |
| **Test Case\_ID** | **Test Case Description** | **Test Steps** | | **Pre-Conditions** | **Test Data** | **Post-Conditions** | **Expected Output** | **Actual Output** | **State** |
| TC\_newsLetter\_001 |  | The user will click the checkbox | | User must be logged in |  | An email will be sent to the user |  |  | Passed |

Table(25): news letter test case

# References

1. Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems Book by Geron Aurelien
2. Deep Learning-Based Approaches for Sentiment Analysis Book by Dr. Basant Agarwa
3. [Cryptocurrency Prediction with LSTM | by Soner Yıldırım | Towards Data Science](https://towardsdatascience.com/cryptocurrency-prediction-with-lstm-4cc369c43d1b)
4. 2021. DEEP LEARNING-BASED APPROACHES FOR SENTIMENT ANALYSIS. [S.l.]: SPRINGER VERLAG, SINGAPOR.
5. [(PDF) Comparative Performance of Machine Learning Algorithms for Cryptocurrency Forecasting (researchgate.net)](https://www.researchgate.net/publication/326837070_Comparative_Performance_of_Machine_Learning_Algorithms_for_Cryptocurrency_Forecasting)
6. Burniske, C. and Tatar, J., n.d. Cryptoassets.
7. Géron, A., n.d. Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow.
8. Shah, V.H., 2007. Machine learning techniques for stock prediction. Foundations of Machine Learning| Spring, 1(1), pp.6-12.