LOW LEVEL DESIGN AND IMPLEMENTATION DOCUMENT

PAYCRYPTO: A Fintech Application for the Currency of Future

UE20CS461A – Capstone Project Phase – 2

Submitted by:

Aryansh Garg	**PES2UG20CS070**
Harsh Agrawal	**PES2UG20CS908**
Harshit Sharma	**PES2UG20CS909**
Sagarikha M.	**PES2UG20CS292**

Under the guidance of

Prof. Gauri S. Rapate
Associate professor
PES University

August - December 2023

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
Electronic City, Bengaluru – 560 100, Karnataka, India

TABLE OF CONTENTS

Note:

Section 1	**Common for Prototype/Product Based and Research Projects**
Section 2 & 3	**Applicable for Prototype / Product Based Projects.**
Section 4	**Applicable for Research Projects.**
Appendix	**Provide details appropriately**

1.Introduction

This document contains details regarding the low-level design of our project, PayCrypto: A fintech application for the currency of the future. The aim of this document is to provide a detailed technical blueprint and with a better understanding of the way our project is structured and make it easier to understand the working of the project. It will serve as a reference for maintenance, testing and communication with stakeholders.

It will also provide transparency and accountability as it contains a clear record of the design decisions and technical choices made during the development process.

Design Constraints, Assumptions, and Dependencies

- **Assumptions**

- Users will have access to the internet and a compatible device to use the app.
- The app will be deployed on a cloud-based infrastructure.
- Users will have some basic understanding of cryptocurrency and its market trends.
- The app's predictive analysis algorithm will be accurate in predicting cryptocurrency prices.

- **Dependencies**

- The app will depend on external APIs to access data related to cryptocurrency market trends.
- The app will depend on user input to refine the sentimental analysis algorithm.

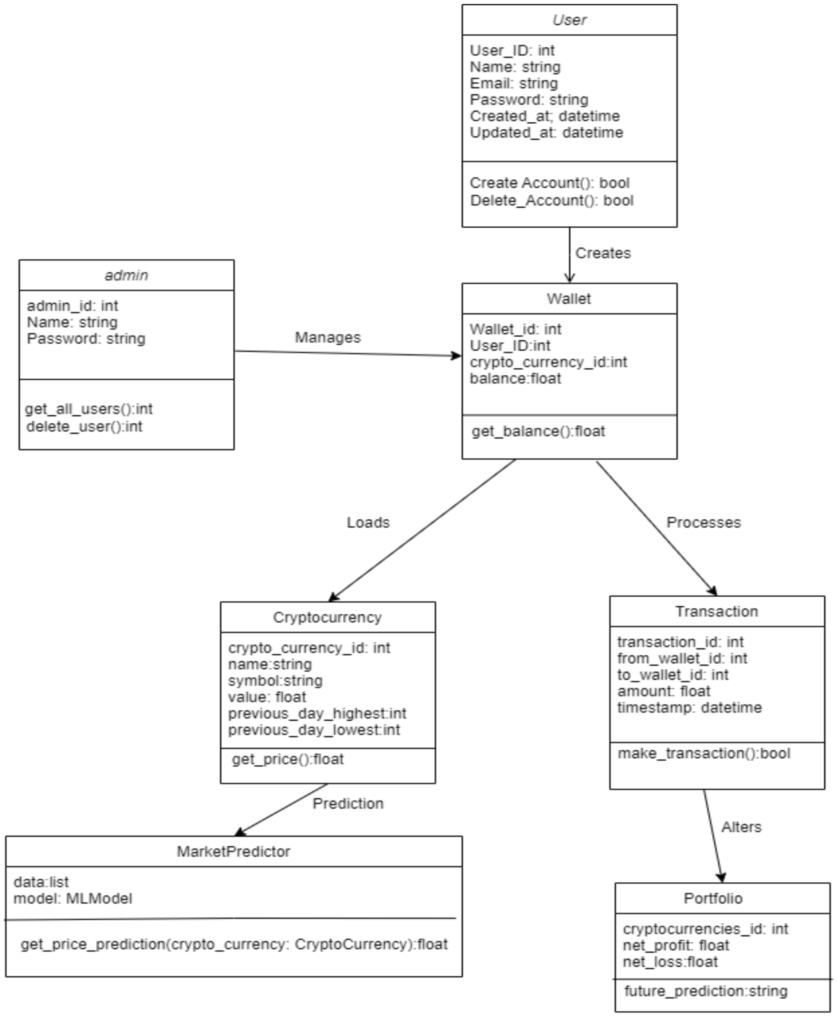
- **Constraints**

- The app's performance will be limited by the speed and accuracy of external APIs used to gather market data.
- The app's performance may also be constrained by the user's hardware and network capabilities.
- The app will need to adhere to security and privacy regulations regarding user data storage and transmission.
- The app will need to adhere to regulations regarding the use of predictive analysis algorithms for financial purposes.

Design Description

- **Data storage module:** It is responsible for storing historical and real time data.
- **Historical data analysis module:** This module is responsible for pre-processing and analyzing historical data for the future predictions of the currencies.
- **User interface module:** This will interact with the end user and receive commands to perform desirable tasks.
- **Wallet transfer module:** This will be responsible for transferring the amount from one user's wallet to another as per the user's need.
- **Security module:** This will work as a firewall between a user and an account and will be responsible for preventing unauthorized users from accessing the account.

1.1. Master Class Diagram



Class Name

- User
- Admin
- Wallet
- Cryptocurrency
- Prediction
- Transaction
- MarketPredictor
- Portfolio

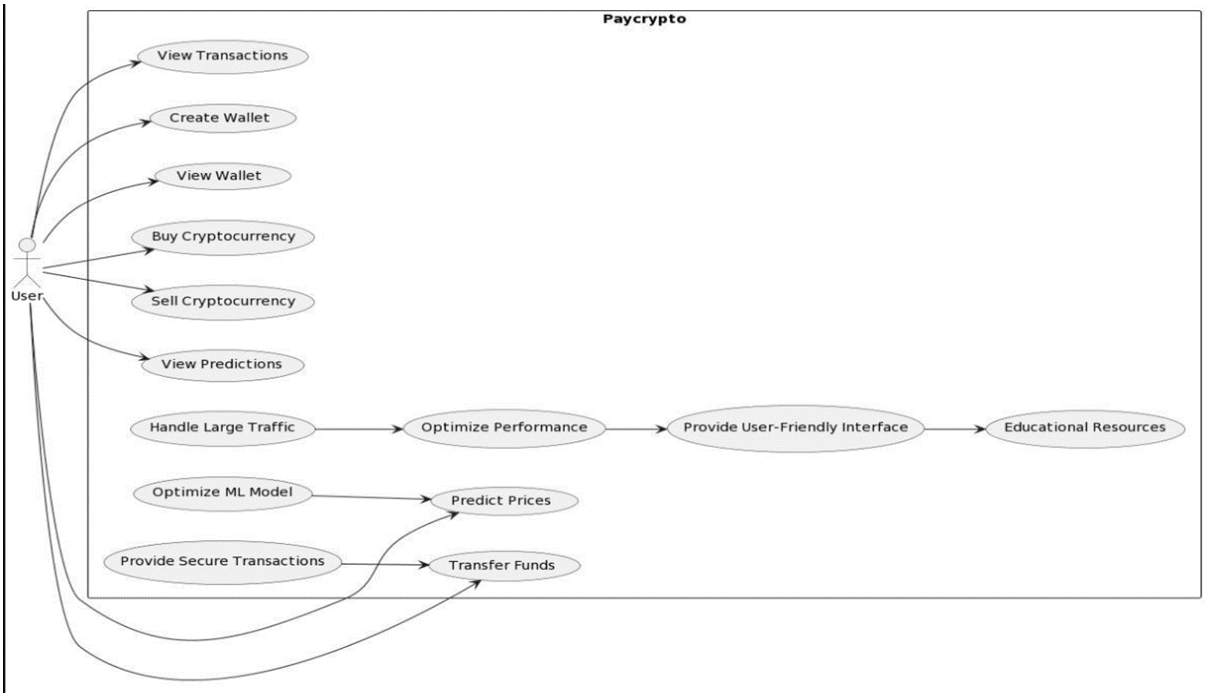
Class Description

- User: Represents a user of the system.
- Admin: Represents an administrator of the system.
- Wallet: Represents a cryptocurrency wallet.
- Cryptocurrency: Represents a cryptocurrency.
- Prediction: Represents a prediction of the future price of a cryptocurrency.
- Transaction: Represents a transaction between two wallets.
- MarketPredictor: A class that can be used to make predictions about the future price of cryptocurrencies.
- Portfolio: A class that represents a portfolio of cryptocurrencies.

Method

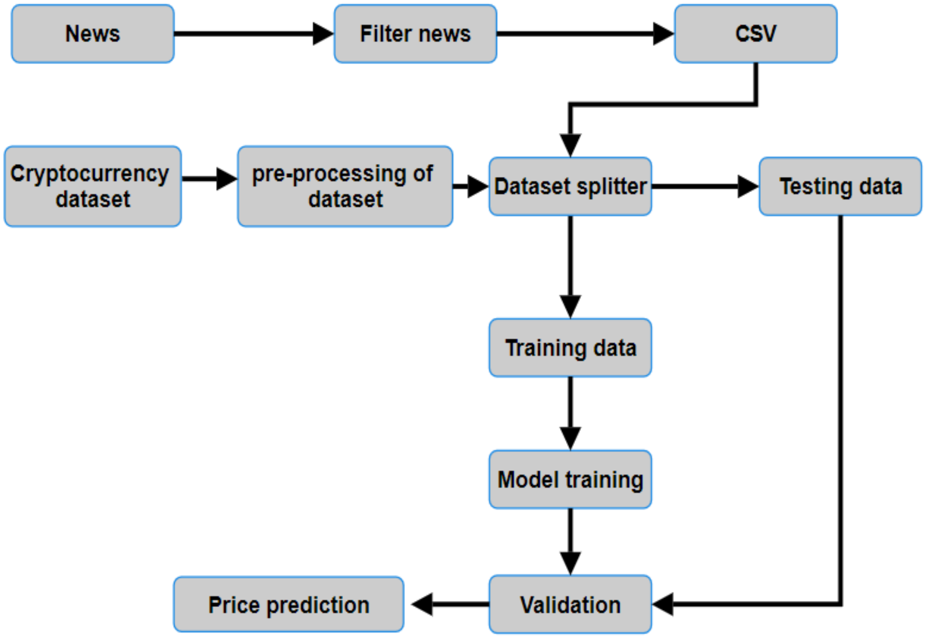
- User.CreateAccount(): Creates a new user account.
- User.Delete_Account(): Deletes a user account.
- Admin.Manages(): Manages users and wallets.
- Wallet.get_all_users(): Gets a list of all users who own the wallet.
- Wallet.delete_user(): Deletes a user from the wallet.
- Wallet.get_balance() : Gets the balance of the wallet.
- Cryptocurrency.get_price(): Gets the current price of the cryptocurrency.
- Prediction.get_price_prediction(): Gets a prediction of the future price of the cryptocurrency.
- Transaction.make_transaction(): Makes a transaction between two wallets.
- MarketPredictor.get_price_prediction(): Gets a prediction of the future price of a cryptocurrency.
- Portfolio.get_cryptocurrencies(): Gets a list of all cryptocurrencies in the portfolio.
- Portfolio.get_net_profit(): Gets the net profit of the portfolio.
- Portfolio.get_net_loss(): Gets the net loss of the portfolio

3.2 Use Case Diagram



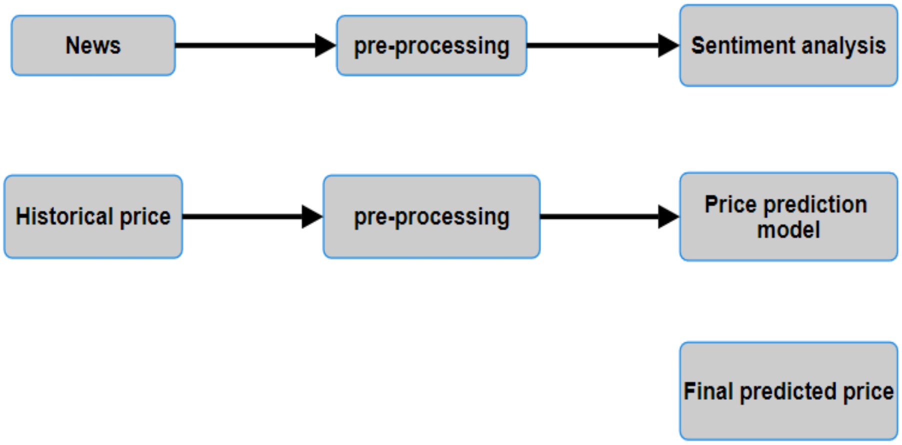
Use Case Item	Description
Create wallet	A user can create wallet to make transactions with other users in cryptocurrency
View transactions	Users can view transactions made by him/her in the past and keep track of the same.
View predictions	Users can view the predictions made by PayCrypto for the future prices of the currencies available on the platform and can make investment decisions according to it.
Buy crypto	Users can buy cryptocurrency available on the platform after looking at the predictions for maximum profit.
Sell crypto	Users can sell cryptocurrency to make profits

3.3 Sequence Diagram



- The sequence diagram gives an overview of the processes running in the backend for the price prediction model.
- Basically, the model consists of two parts, first is historical data analysis and the second is sentiment analysis. The model uses features from historical data as well as sentiment analysis to predict the future prices of the cryptocurrency.
- Historical data is pre-processed and split into training and testing data.
- For sentiment analysis three categories have been created namely positive, negative and neutral.
- The combination of both the metrics will be used to predict the prices of cryptocurrencies.

3.4 Packaging and Deployment Diagrams



The above diagram depicts the structure of the model where the two branches that are sentiment analysis and historical data analysis combine into the model for the price prediction.

3.5 Proposed Methodology / Approach

- **Basic approach**

The project is divided basically into two major parts

- The cryptocurrency trading with price prediction
- Peer to peer cryptocurrency transactions

For the first part, a machine learning model will be developed with historical data and social media sentiment analysis to predict the prices of different currencies.

Sentiment analysis is being used along with historical data to bring in the fact that many influential personalities can affect the prices through their opinions and views towards the crypto market on social media.

- **Constraints**

- The app's performance will be limited by the speed and accuracy of external APIs used to gather market data.
- The app's performance may also be constrained by the user's hardware and network capabilities.
- The app will need to adhere to security and privacy regulations regarding user data storage and transmission.
- The app will need to adhere to regulations regarding the use of predictive analysis algorithms for financial purposes.

- **Assumptions**

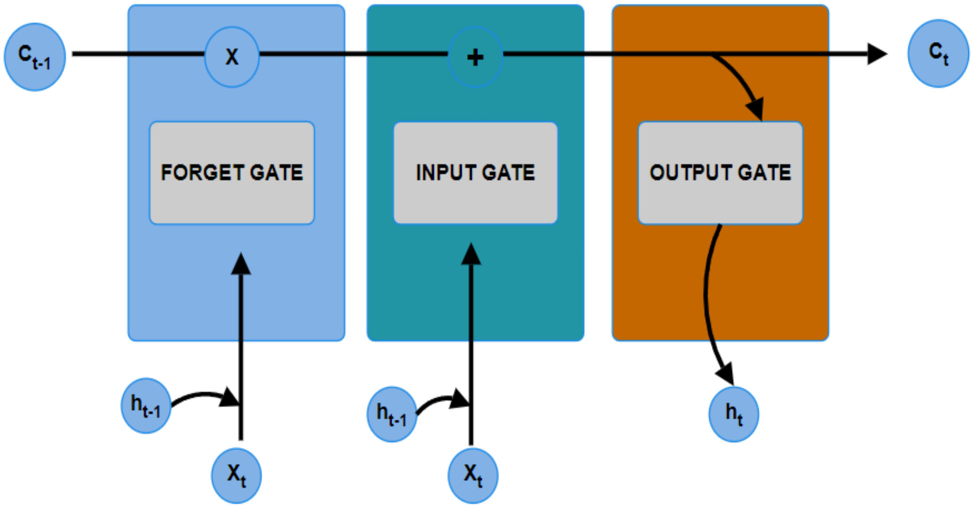
- Users will have access to the internet and a compatible device to use the app.
- The app will be deployed on a cloud-based infrastructure.
- Users will have some basic understanding of cryptocurrency and its market trends.
- The app's predictive analysis algorithm will be accurate in predicting cryptocurrency prices.

- **Dependencies**

- The app will depend on external APIs to access data related to cryptocurrency market trends.
- The app will depend on user input to refine the sentimental analysis algorithm.

4.1 Algorithm and Pseudocode

- For various learning issues involving sequential data, recurrent neural networks with long short-term memory (LSTM) have emerged as an effective and scalable approach. They are useful for capturing long-term temporal dependencies since they are generic and effective. The LSTM is an RNN-style architecture with gates that govern the flow of information between cells. The input and forget gate structures can modify information traveling along the cell state, with the ultimate output being a filtered version of the cell state based on context from the inputs



LSTM memory cell

The above figure depicts the LSTM memory cell which involves input, output and forget gate.

The equations for the gates in LSTM are: -

$$i_t = \sigma(w_i[h_{t-1}, x_t] + b_i) \quad \ldots\ldots\ldots\ldots \textbf{(1)}$$

$$f_t = \sigma(w_f[h_{t-1}, x_t] + b_f) \quad \ldots\ldots\ldots\ldots \textbf{(2)}$$

$$o_t = \sigma(w_o[h_{t-1}, x_t] + b_o) \quad \ldots\ldots\ldots\ldots \textbf{(3)}$$

$$\begin{aligned}
i_t &\rightarrow represents\ input\ gate. \\
f_t &\rightarrow represents\ forget\ gate. \\
o_t &\rightarrow represents\ output\ gate. \\
\sigma &\rightarrow represents\ sigmoid\ function. \\
w_x &\rightarrow weight\ for\ the\ respective\ gate(x)\ neurons. \\
h_{t-1} &\rightarrow output\ of\ the\ previous\ lstm\ block(at\ timestamp\ t-1). \\
x_t &\rightarrow input\ at\ current\ timestamp. \\
b_x &\rightarrow biases\ for\ the\ respective\ gates(x).
\end{aligned}$$

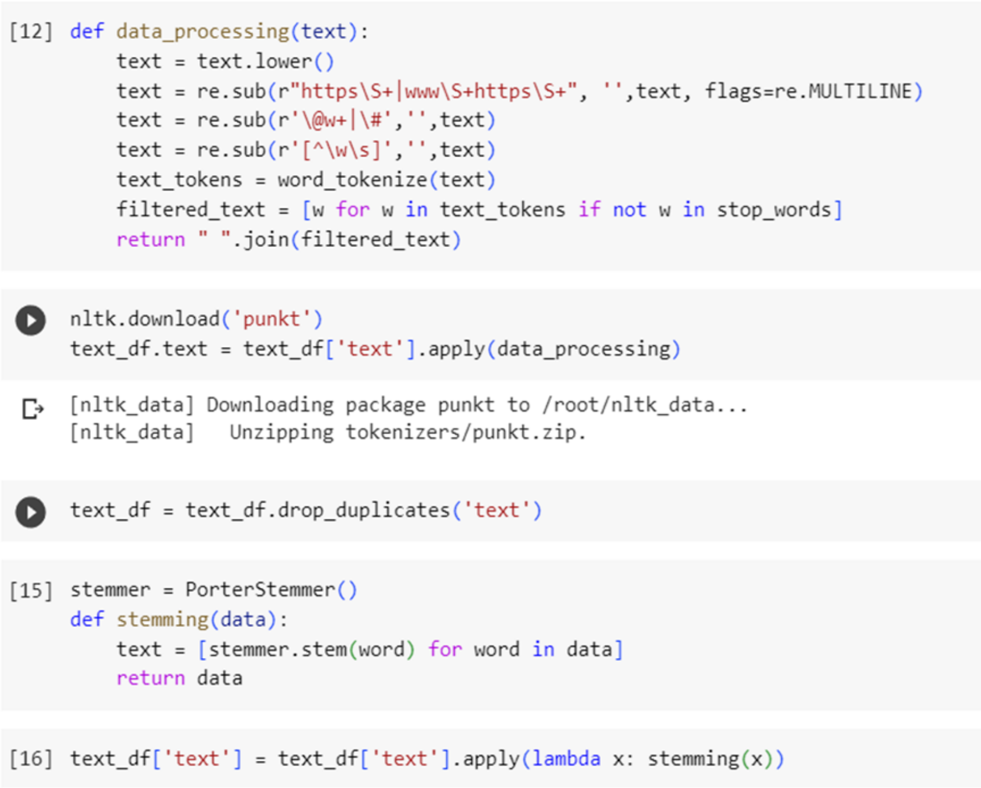
First equation is for *Input Gate* which tells us that what new information we’re going to store in the cell state.

Second is for the forget gate which tells the information to throw away from the cell state.

Third one is for the output gate which is used to provide the activation to the final output of the LSTM block at timestamp ‘t’.

Data pre-processing:

The historical data of the cryptocurrency has to be pre-processed for the desired data points to be used in the model.



```
[12] def data_processing(text):
         text = text.lower()
         text = re.sub(r"https\S+|www\S+https\S+", '',text, flags=re.MULTILINE)
         text = re.sub(r'\@w+|\#','',text)
         text = re.sub(r'[^\w\s]','',text)
         text_tokens = word_tokenize(text)
         filtered_text = [w for w in text_tokens if not w in stop_words]
         return " ".join(filtered_text)
```

```
nltk.download('punkt')
text_df.text = text_df['text'].apply(data_processing)
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

```
text_df = text_df.drop_duplicates('text')
```

```
[15] stemmer = PorterStemmer()
     def stemming(data):
         text = [stemmer.stem(word) for word in data]
         return data
```

```
[16] text_df['text'] = text_df['text'].apply(lambda x: stemming(x))
```

The above commands will pre-process the dataset to produce the desired outcome that will be used as input to the model.

REFERENCES:

- -A. Encean and D. Zinca, "Cryptocurrency Price Prediction Using LSTM and GRU Networks," 2022 International Symposium on Electronics and Telecommunications (ISETC), Timisoara, Romania, 2022, pp. 1-4, doi: 10.1109/ISETC56213.2022.10010329.

- Anh-Dung Vo, Quang-Phuoc Nguyen, and Cheol-Young Ock: "Sentiment Analysis of News for Effective Cryptocurrency Price Prediction", INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH VOLUME 9, ISSUE 04, APRIL 2020 ISSN22778616

.