CHAPTER 1

INTRODUCTION:

1.1PROBLEM DEFINITION

Bitcoin is a cryptographic money which is utilized worldwide for advanced installment or basically for speculation purposes. Bitcoin is decentralized for example it isn't possessed by anybody. Bitcoins are put away in an advanced wallet which is essentially similar to a virtual financial balance. The record of the considerable number of exchanges, the timestamp information is put away in a spot called Block chain. Each record in a block chain is known as a square. Each square contains a pointer to a past square of information.

The information on block chain is scrambled. During exchanges the client's name isn't uncovered, however just their wallet ID is made open. The Bitcoin's worth fluctuates simply like a stock though in an unexpected way. There are various calculations utilized on financial exchange information for value forecast. Notwithstanding, the parameters influencing Bitcoin are extraordinary. In this manner it is important to anticipate the estimation of Bitcoin so right venture choices can be made. The cost of Bitcoin doesn't rely upon the business occasions or mediating government not at all like securities exchange. Hence, to anticipate the worth we feel it is important to use AI innovation to foresee the cost of Bitcoin.We will predict the sign of the daily price change with highest possible accuracy .

Since the barter system, trade and investment is an integral part of economic development of a country. Over a time period significant changes have been made in trade which leads to the growth of national economy. The changes like conversion from barter system to moneybased trades, introduction of digital currency like bitcoin (BTC) are significant. The initial value of a BTC in the year of 2009 was $0.0008 and over the years the price of BTC has drastically increased to $46,434.40.This indicates the popularity of BTC among the current society. This cryptocurrency is used extensively to add data into blockchain. Blockchain technology is used by many countries in various fields like health care, banking, business etc, mainly because of its high level security and scalability.author has claimed that the bitcoin network can

be used for the transaction of current clearing systems in order to make the transactions safer and faster.

Through the cryptocurrencies there is ease and security in the transactions of any field, which increases the usage of blockchain network which ultimately supports and improves digital money usage and transfer. Even though BTC has many advantages, and inspite of the enthusiasm shown by multinational companies to use BTC as a digital asset, the fear of unfamiliarity, complexity and unstable nature of bitcoins makes it less popular in majority areas of the world. So prediction of BTC is necessary to convince the traders to invest in BTC which will lead to the growth of world economy.

CHAPTER 2

LITERATURE REVIEW

David Garcia-Castro et al.,[1] "Forecasting the Bitcoin Price using Deep Learning Techniques"**.** This article discusses the use of deep learning techniques for forecasting the price of Bitcoin. The authors proposed an architecture based on a long short-term memory (LSTM) network and compared its performance with other traditional machine learning models. The authors collected historical data on Bitcoin prices and other relevant economic indicators, and trained the LSTM network on this data. They also used principal component analysis (PCA) to reduce the dimensionality of the input features. The performance of the LSTM network was compared with that of other models such as random forests, linear regression, and support vector machines. The use of deep learning techniques, specifically LSTM networks, can be an effective approach for forecasting the price of Bitcoin. However, further research is needed to explore the impact of other factors on Bitcoin prices and to improve the accuracy of the forecasting models.

Alok Kothari et al.,[2] "Predicting Bitcoin Prices using Convolutional Neural Networks". This article discusses the use of convolutional neural networks (CNN) for predicting the prices of Bitcoin. The authors proposed a hybrid model that combines CNN with a traditional autoregressive integrated moving average (ARIMA) model for forecasting Bitcoin prices. The use of CNNs and hybrid models can be an effective approach for predicting the prices of Bitcoin. However, further research is needed to explore the impact of other factors on Bitcoin prices and to improve the accuracy of the forecasting models.

Sercan Özen et al.,[3] "Bitcoin Price Forecasting Using Deep Learning: A Comparative Study"**.** This article discusses the use of deep learning techniques for Bitcoin price forecasting and compares the performance of different deep learning models. The authors proposed using long short-term memory (LSTM), gated recurrent units (GRU), and a hybrid model that combines both LSTM and GRU. The authors collected historical data on Bitcoin prices and other relevant features, such as trading

volume and sentiment analysis, and used this data to train the different deep learning models. They also used backpropagation through time (BPTT) to optimize the models and tested their performance on a separate testing dataset. The use of deep learning techniques, specifically LSTM and GRU, can be an effective approach for Bitcoin price forecasting. The hybrid LSTM-GRU model can improve the prediction accuracy compared to using individual models. However, further research is needed to explore the impact of other factors on Bitcoin prices and to improve the accuracy of the forecasting models.

Renato Mendes Coutinho et al.,[4] "Bitcoin Price Prediction Using Deep Learning Models with Technical and Sentiment Features". This article discusses the use of deep learning models for Bitcoin price prediction using both technical and sentiment features. The authors proposed a hybrid model that combines convolutional neural networks (CNN) and long short-term memory (LSTM) networks with a sentiment analysis feature. One limitation of this study is that it only considers sentiment data from Twitter and does not take into account other sources of sentiment data or other factors that may affect Bitcoin prices, such as regulatory changes or news events. The use of deep learning models, specifically CNN and LSTM, can be an effective approach for Bitcoin price prediction. The hybrid CNN-LSTM model with sentiment analysis can improve the prediction accuracy compared to using only technical indicators. However, further research is needed to explore the impact of other factors on Bitcoin prices and to improve the accuracy of the forecasting models.

Jiangtao Ren et al.,[5] "Bitcoin Price Forecasting Using Deep Learning Models: An Empirical Study" This article discusses the use of deep learning models for Bitcoin price forecasting and compares the performance of different models, including long short-term memory (LSTM), convolutional neural networks (CNN), and a hybrid model that combines both LSTM and CNN.One limitation of this study is that it only considers the historical data of Bitcoin prices and other economic indicators, and does not take into account other factors that may affect Bitcoin prices, such as regulatory changes or news events.The use of deep learning models, specifically LSTM and CNN, can be an effective approach for Bitcoin price forecasting. The hybrid LSTM-CNN model can improve the prediction accuracy compared to using individual models. However, further research is needed to explore the impact of other factors on

Bitcoin prices and to improve the accuracy of the forecasting models. The use of Google Trends data and social media data can also be a valuable addition to improve the accuracy of the models.

Ali Khaleghi et al.,[6] "Bitcoin Price Prediction Using Deep Learning Algorithms With Technical and Fundamental Indicators". This article discusses the use of deep learning algorithms for Bitcoin price prediction using both technical and fundamental indicators. The authors proposed a model that combines an autoencoder with a long short-term memory (LSTM) network to extract and process relevant features from the data. One limitation of this study is that it only considers a limited set of technical and fundamental indicators, and does not take into account other factors that may affect Bitcoin prices, such as regulatory changes or news events. The use of deep learning algorithms, specifically autoencoders and LSTM networks, can be an effective approach for Bitcoin price prediction. The combination of both technical and fundamental indicators can improve the prediction accuracy compared to using only one type of indicator. However, further research is needed to explore the impact of other factors on Bitcoin prices and to improve the accuracy of the forecasting models.

Xiang Li et al.,[7] "Bitcoin Price Prediction Using Deep Learning: An Application of Long-Short Term Memory Neural Network". This article discusses the use of long short-term memory (LSTM) neural network for Bitcoin price prediction. The authors proposed a model that uses a LSTM network to predict the future prices of Bitcoin based on the historical price data and trading volume. The use of LSTM neural network can be an effective approach for Bitcoin price prediction. The inclusion of trading volume as a feature can improve the accuracy of the model.

CHAPTER 3

THEORITICAL BACKGROUND

3.1 IMPLEMENTATION ENVIRONMENT:

Implementation environment for price forecasting and analysis of Bitcoin involves several steps and considerations.

Data Collection:

Gather historical Bitcoin price data from reliable sources like cryptocurrency exchanges (e.g., Coinbase, Binance, Bitfinex) or financial data providers (e.g., Bloomberg, Yahoo Finance). Ensure the data is clean, accurate, and covers a significant timeframe.

Data Preprocessing:

- Clean the data to remove any outliers, missing values, or errors.
- Convert the data into a format suitable for analysis, such as time-series data.

Feature Engineering:

- Extract relevant features that might influence Bitcoin prices, such as trading volume, market sentiment, technical indicators (e.g., moving averages, RSI), macroeconomic factors, and news sentiment.
- Explore different feature combinations and transformations to improve model performance.

Model Selection:

- Choose appropriate forecasting models based on the nature of the problem and available data. Common models for time-series forecasting include ARIMA, SARIMA, LSTM (Long Short-Term Memory), Prophet, and machine learning algorithms like Random Forests or Gradient Boosting Machines.
- Experiment with different models and parameters to find the best performing one.

Training and Validation:

- Split the data into training and validation sets. Use a significant portion of the data for training and keep a smaller portion for validation to evaluate the model's performance.
- Train the selected models on the training data and validate them using the validation set.

Model Evaluation:

- Evaluate the performance of the models using appropriate metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), or others relevant to your specific goals.
- Compare the performance of different models and select the one that performs the best.

Deployment:

- Once you have a model that meets your performance criteria, deploy it to your production environment. This could be a web application, a command-line tool, or an API that users can interact with to get real-time or historical price forecasts.
- Ensure that the deployment environment is scalable, reliable, and secure.

Monitoring and Maintenance:

- Continuously monitor the performance of your model in the production environment.
- Retrain the model periodically with new data to keep it up-to-date and improve its accuracy over time.
- Stay informed about the latest developments in the cryptocurrency market and update your model or features accordingly.

3.2 SYSTEM ARCHITECTURE:

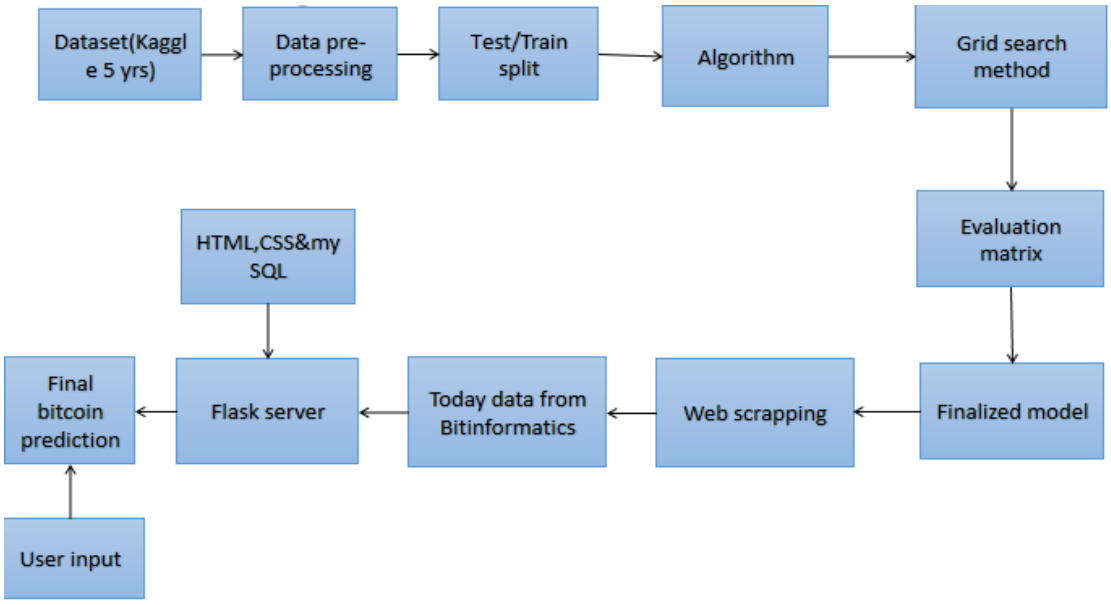

Fig 3.1 System Architecture

System architecture Fig 3.1 the data collection layer, which gathers real-time price data from cryptocurrency exchanges and stores historical data for analysis. Following this, the data processing layer cleanses and preprocesses the collected data while extracting relevant features that may influence Bitcoin prices. The modeling layer employs various forecasting models, including and machine learning algorithms, to predict future price movements based on historical data. These models are trained, evaluated, and optimized to achieve accurate forecasts. Decision-making processes, such as ensemble methods and risk management strategies, enhance the reliability and effectiveness of the forecasts. The presentation layer visualizes the forecasted prices, key indicators, and model performance metrics through interactive dashboards and reporting tools. Deployment involves exposing the forecasting models and analysis tools via web services, APIs, and web applications, ensuring accessibility to users. Scalable infrastructure, containerization, and orchestration facilitate efficient deployment and management of the system. Continuous monitoring, maintenance, and updates ensure the system remains reliable, up-to-date, and responsive to changing market conditions, providing valuable insights for informed decision-making in Bitcoin trading and investment.

3.3 PROPOSED METHODOLOGY:

- In this study, we have used 5 years data sets for Bitfinex for testing and training the ML. With the help of python libraries, the data preprocessing was done.

- Python has provided with a best feature for data analysis and visualization. After the understanding of the data, we trim the data and use the features or attributes best suited for the model.
- It was discovered that the random forest model's accuracy rate is very high when compared to other Machine Learning models from related works.
- In this work, a flask framework has been created using the flask library that will allow the user to input the future date to be predicted. Beautiful Soup is used to scrap the data from 'url': 'https://bitinfocharts.
- The future prediction of bit coin is predicted as a result from today real time data.

ADVANTAGES OF PROPOSED SYTEM:

- The Advantage of Bayesian regression in Bitcoin price prediction results has been showed in binary values.
- It helps to understand the results very neatly.
- It works the prediction by taking the coinMarkup cap.

3.3.1 SOFTWARE REQUIREMENT:

HARDWARE:

- Processor: Intel Core I5
- Disk space 320 GB
- Operating system: windows 10, macOS and Linux

SOFTWARE:

- OS: WINDOWS OR LINUX
- TOOL: PYTHON

3.4 MODULE DESIGN:

3.4.1 System Design

System design is the process of planning a new system or to replace the existing system. Simply, system design is like the blueprint for building, it specifies all the features that are to be in the finished product.

3.4.2 Sequence Diagram

A Sequence diagram is a kind of interaction diagram Fig 3.4.2.1 that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

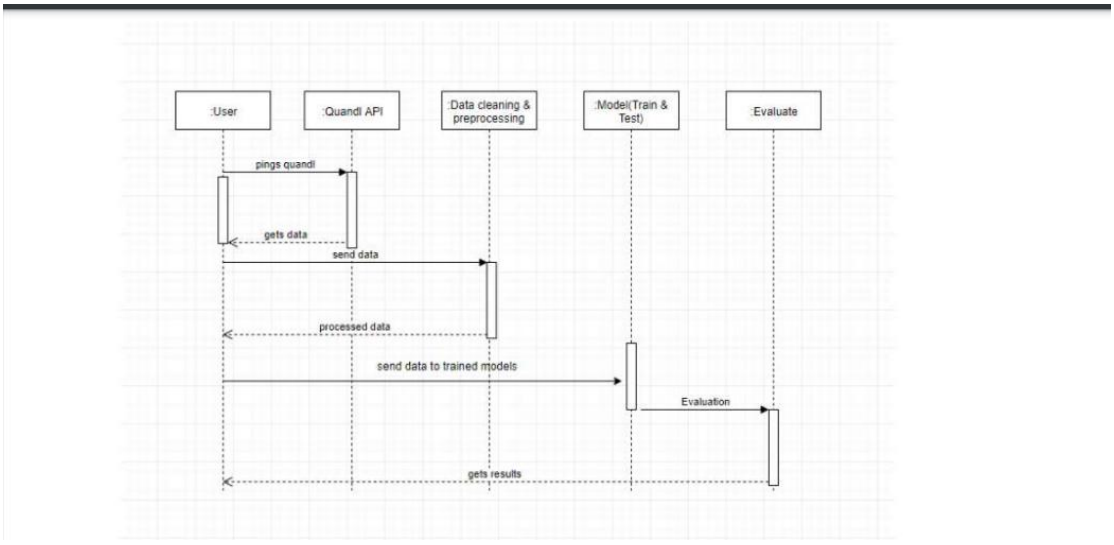

Fig 3.4.2.1 Sequence Diagram

3.4.3 Use Case Diagram

Unified Modeling Language (UML) Fig.3.4.3.1 Use cases are used to describe the visible interactions that the system will Have with users and external systems. They are used to describe how a user would Perform their role using the system

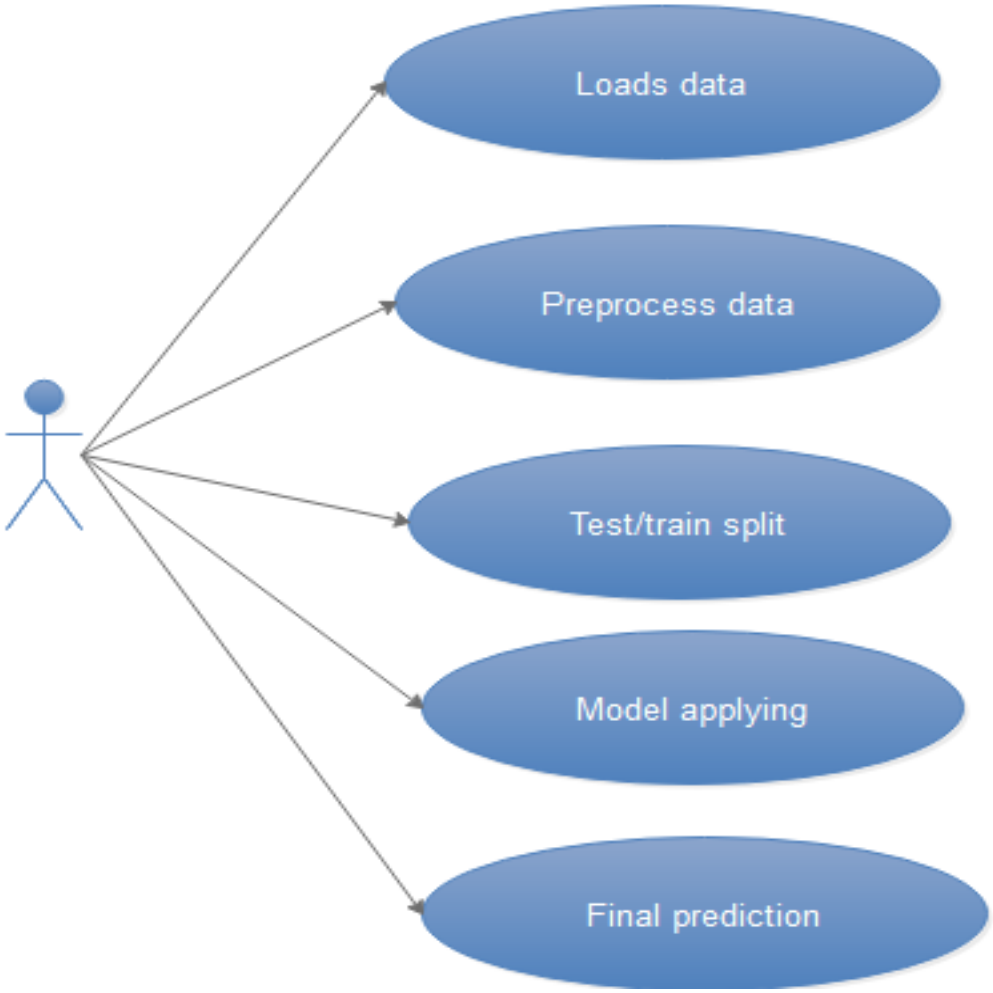

Fig. 3.4.3.1 Use Case Diagram

3.4.4 Activity Diagram

Activity diagram Fig. 3.4.4.1 is a graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. An activity diagram shows the overall flow of control.

The most important shape types

- Rounded rectangles represent activities.
- Diamonds represent decisions.
- Bars represent the start or end of concurrent activities.
- A black circle represents the start of the workflow.
- An encircled circle represents the end of the workflow.

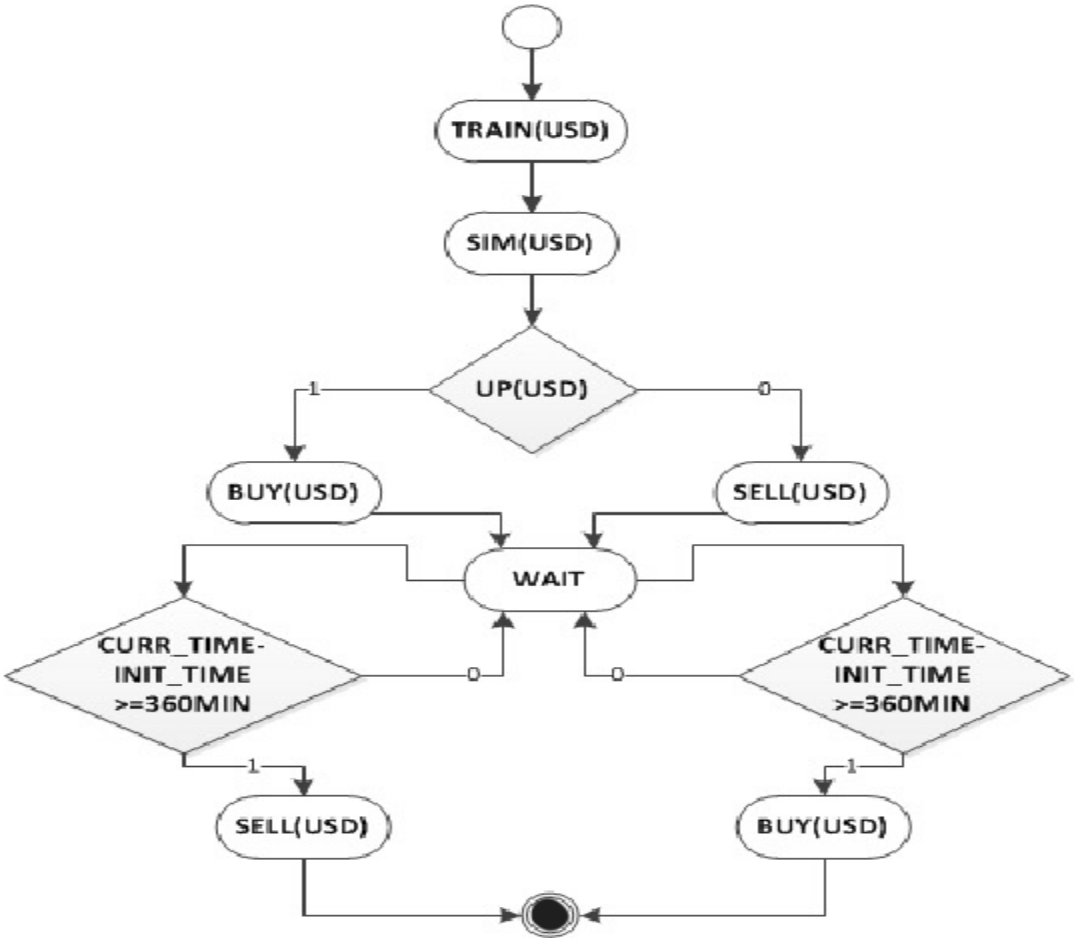

Fig. 3.4.4.1 Activity Diagram

3.4.5 Collaboration Diagram

UML Collaboration Diagrams Fig.3.4.5.1 illustrate the relationship and interaction between software objects. They require use cases, system operation contracts and domain model to already exist. The collaboration diagram illustrates messages being sent between classes and objects.

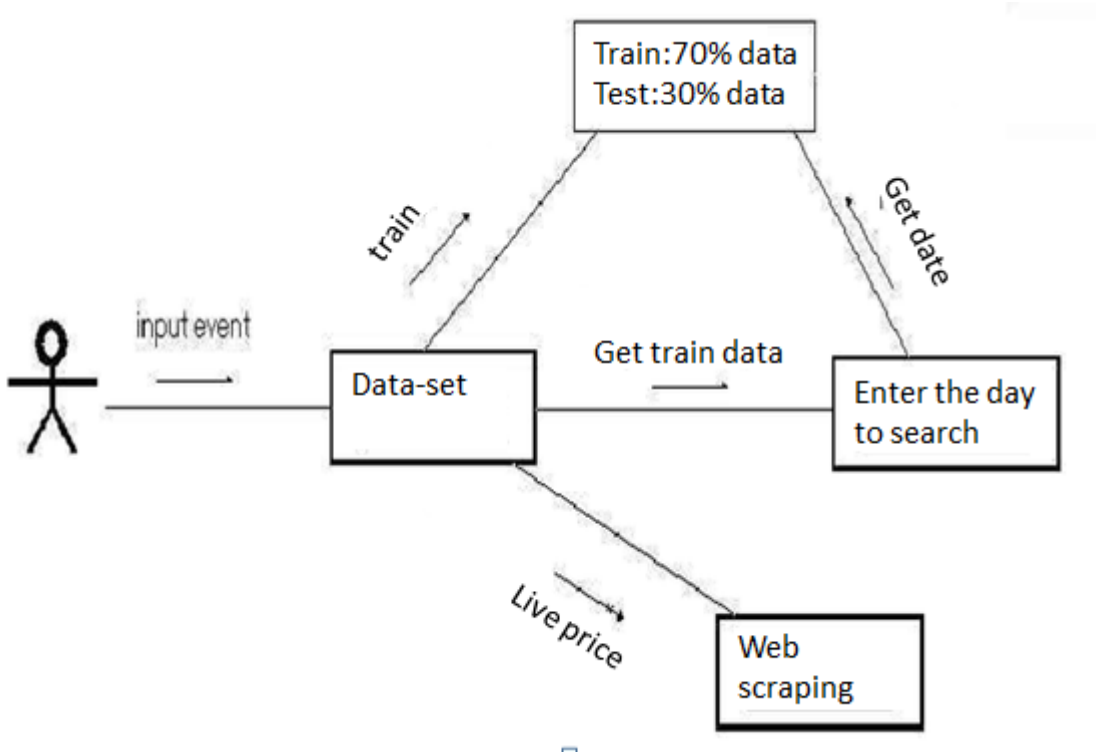

Fig.3.4.5.1 Collaboration Diagram

3.4.6 Class Diagram

The class diagram Fig 3.4.6.1 is a static diagram. It represents the static view of an application. The class diagrams are widely used in the modeling of object-oriented systems because they are the only UML diagrams which can be mapped directly with object-oriented languages. The standard is managed and was created by the Object Management Group. Includes a set of graphic notation techniques to create visual models of software intensive systems.

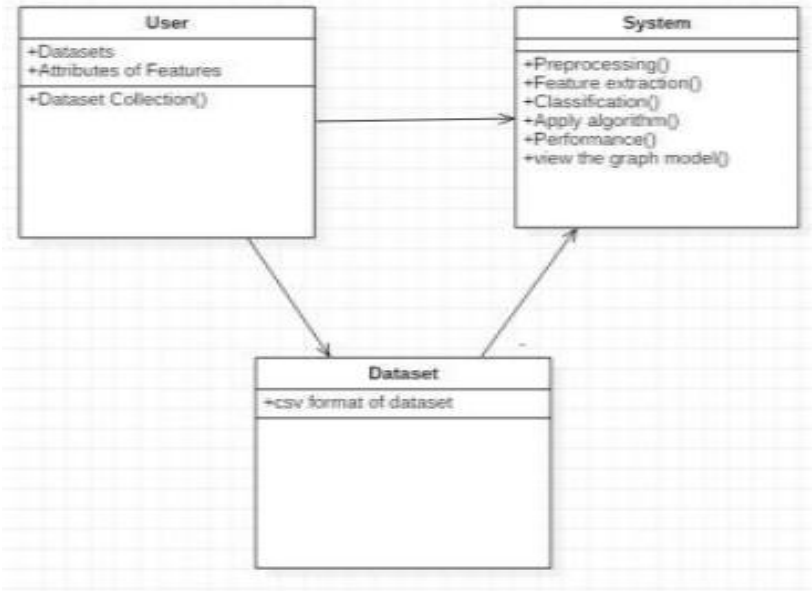

Fig 3.4.6.1 Class Diagram

3.4.7 Deployment Diagram

A deployment diagram Fig 3.4.7.1 is a type of diagram in the Unified Modeling Language (UML) that shows the physical aspects of an object-oriented software system. It models the run- time configuration in a static view, and visualizes the distribution of components in an application. Deployment diagrams are important for visualizing, specifying, and documenting embedded, client/server, and distributed systems

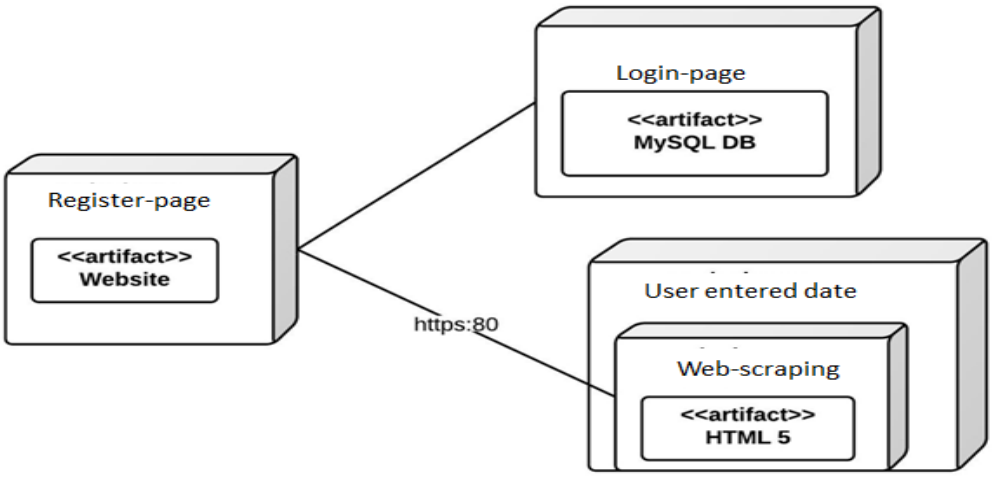

Fig 3.4.7.1 Deployment Diagram

3.4.8 Data Flow Diagram

A Data Flow Diagram (DFD) Fig. 3.4.8.1 is a graphical representation of the “flow” of data through an information system, modeling its aspects. It is a preliminary step used to create an overview of the system which can later be elaborated DFDs can also be used for visualization of data processing.

Level 0

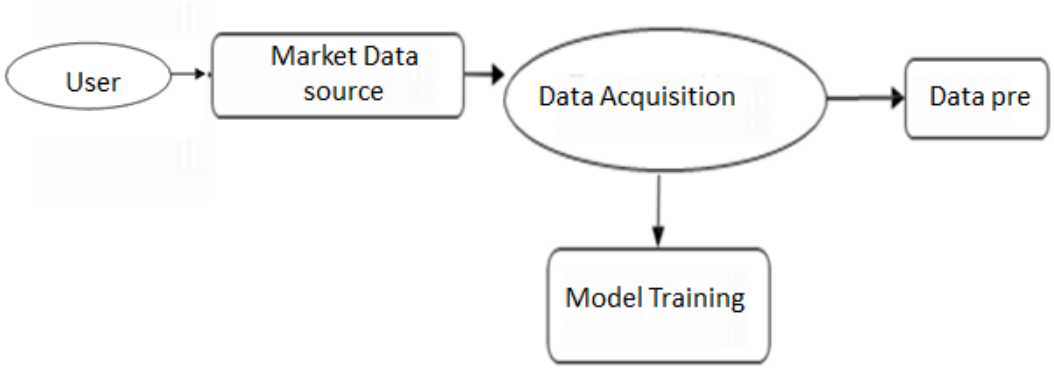

Fig 3.4.8.1 Level 0 Data Flow Diagram

Level 1

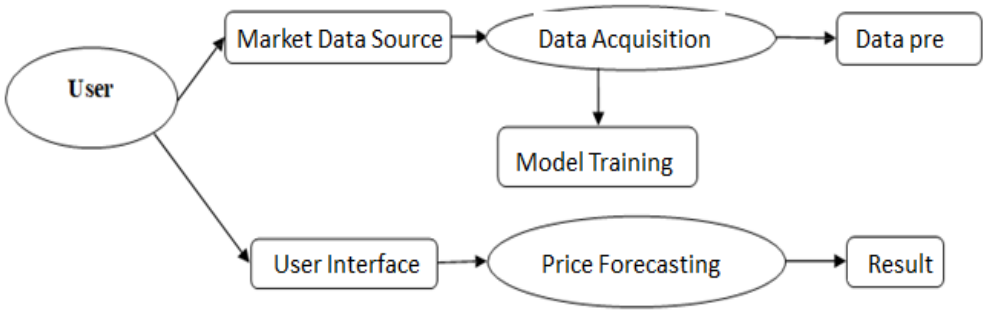

Fig 3.4.8.2 Level 1 Data Flow Diagram

Level 2

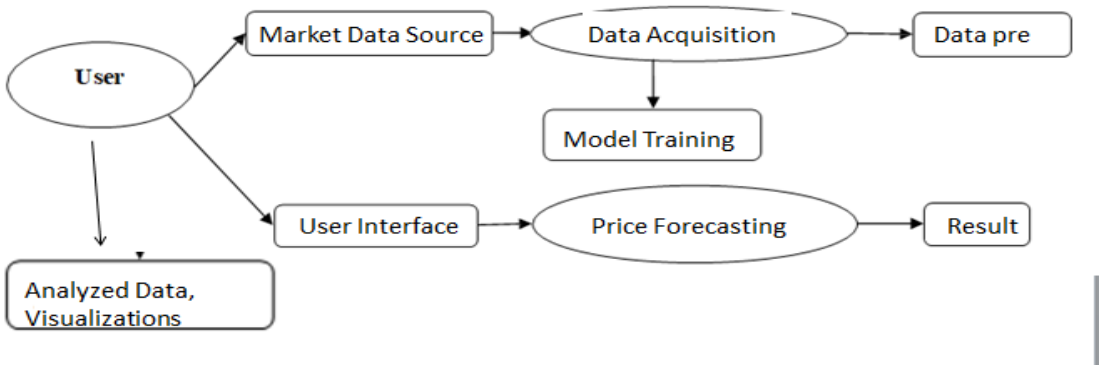

Fig 3.4.8.3 Level 2 Data Flow Diagram

CHAPTER 4

SYSTEM IMPLEMENTATION

4.1 MODULE EXPLANATION

4.1.1 Login Module

A login module Fig 4.1.1.1 is a fundamental component of many software systems, including web applications, mobile apps, and desktop applications. Its primary purpose is to authenticate users by verifying their identity based on credentials such as usernames and passwords. Here's an explanation of how a login module typically works:

User Interface: The login module typically consists of a user interface where users can input their credentials, such as a username/email and password. This interface may also include features like "Remember Me" options or "Forgot Password" links.

Validation: Once the user submits their credentials, the login module validates them against the stored user database. This validation process ensures that the entered credentials match those associated with an existing user account.

Authentication: If the entered credentials are valid, the login module authenticates the user, granting them access to the system. Authentication involves generating a session token or setting a session cookie to maintain the user's authenticated state throughout their interaction with the system.

Authorization: After successful authentication, the system may perform additional checks to determine the user's access rights or permissions. This step, known as

authorization, ensures that authenticated users can only access the features and resources that they are authorized to use.

Error Handling: If the entered credentials are invalid, the login module typically displays an error message informing the user of the issue. This feedback helps users correct any mistakes and retry the login process.

Security Measures: A robust login module incorporates security measures to protect user credentials and prevent unauthorized access. This may include encrypting passwords before storing them in the database, implementing secure communication protocols (e.g., HTTPS), and enforcing strong password policies.

Session Management: The login module is responsible for managing user sessions, including initiating sessions upon successful login, maintaining session state during the user's interaction with the system, and terminating sessions upon user logout or inactivity.

Logging: The login module may log relevant login activities, such as successful logins, failed login attempts, and logout events. These logs can be useful for auditing, troubleshooting, and security monitoring purposes. Overall, a well-designed login module plays a crucial role in ensuring the security, usability, and functionality of software systems by properly authenticating users and controlling access to protected resources.

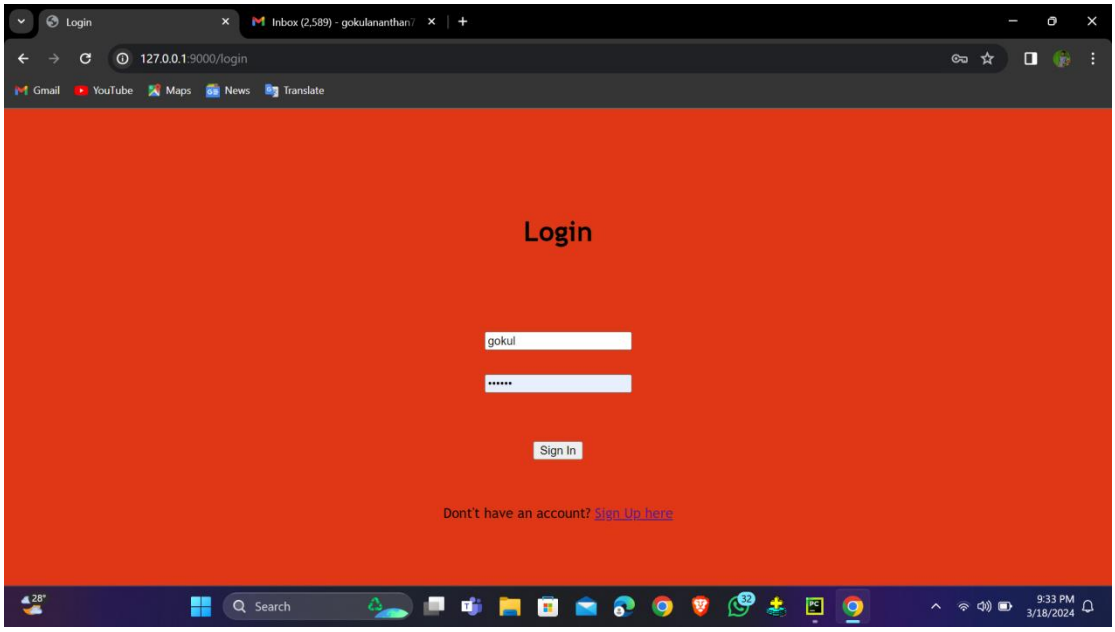

Fig 4.1.1.1 Login Module

4.1.2 Selection Module

Selecting the day in price prediction Fig 4.1.1.1 Selecting Module involves choosing the specific time horizon for which you want to forecast the price of an asset, such as Bitcoin. The choice of the prediction horizon can significantly impact the accuracy and usefulness of the forecast. Here are some key considerations when selecting the day for price prediction:

Short-Term vs. Long-Term Forecasting:

- Short-term forecasting typically focuses on predicting price movements over a relatively short period, such as hours, days, or weeks. These forecasts are often used by traders and investors to make decisions about buying, selling, or holding assets in the near future.

- Long-term forecasting, on the other hand, involves predicting price trends over more extended periods, ranging from months to years. These forecasts are valuable for long-term investment strategies and portfolio management.

Volatility and Accuracy:

- Short-term price predictions tend to be more volatile and challenging to predict accurately due to the higher level of noise and randomness in the market over short time intervals.
- Long-term price predictions may offer more stable trends and patterns, making them potentially easier to forecast accurately, although they still involve uncertainty and risk.

Data Granularity:

- The choice of the prediction horizon should consider the granularity of the available data. For example, if you have access to high-frequency data you may be able to make more accurate short-term predictions.
- For long-term forecasting, you may rely on daily, weekly, or monthly price data, depending on the availability and reliability of historical data.

Trading and Investment Strategies:

- The selection of the prediction horizon should align with your trading or investment strategy. If you are a day trader looking to capitalize on short-term price movements, you may focus on intraday or daily forecasts.

- If you are a long-term investor with a buy-and-hold strategy, you may be more interested in monthly or yearly forecasts to inform your investment decisions.

Model Complexity and Performance:

- The complexity of the forecasting model may influence the choice of the prediction horizon. Simple models may perform better for short-term forecasts, where rapid price changes require agility and adaptability.
- More complex models, such as machine learning algorithms or deep learning models, may be better suited for long-term forecasting, where capturing underlying trends and patterns is crucial.

Risk Management:

- Consider the level of risk associated with different prediction horizons. Short-term forecasts may involve higher risk due to market volatility and uncertainty, whereas long-term forecasts may offer more stability but still carry inherent risks.

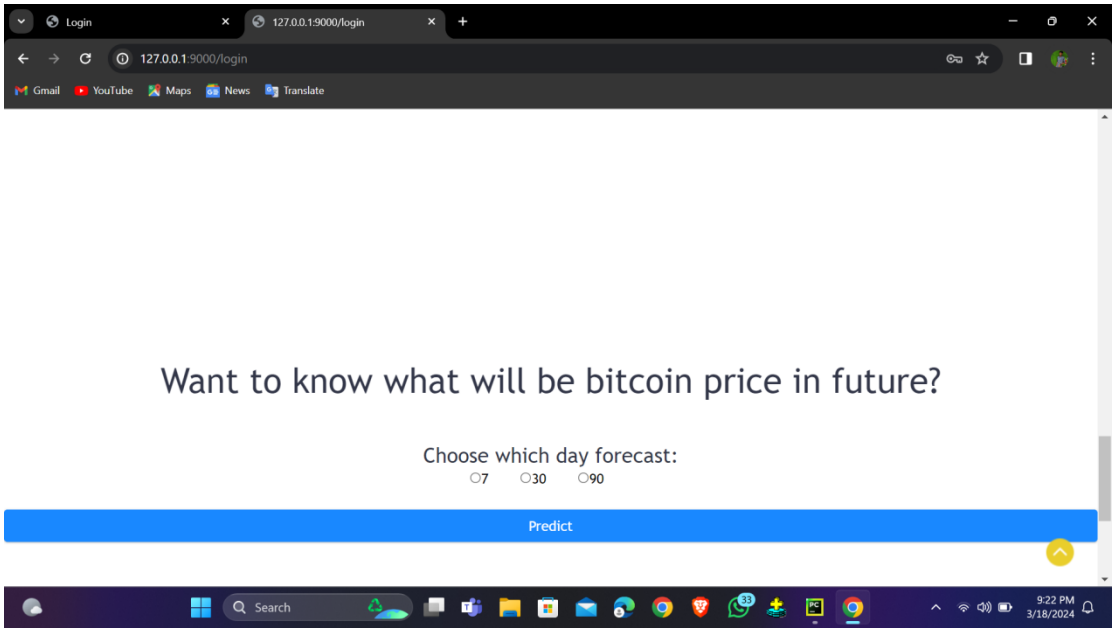

Fig 4.1.2.1 Selecting Module

4.1.3 Prediction Module

The concept of a "predicted module result" could refer to various contexts depending on the field or domain you're discussing. Here are a few potential interpretations:

Machine Learning/Statistics: In predictive modeling, a "module" could represent a component or feature of a larger model. A "predicted module result" in this context might refer to the output or forecast generated by a specific part of the model for a given input. For instance, in a neural network, each layer could be considered a module, and the predicted result of each layer contributes to the final output.

Software Development: In software engineering, a "module" typically refers to a self-contained unit of code that performs a specific function. A "predicted module result" might indicate the anticipated outcome or behavior of a module based on its design and inputs.

Education/Training: In an educational setting, particularly in courses with modular structures, a "module" often represents a distinct section of study within a larger curriculum. A "predicted module result" could then refer to an expected outcome or grade for a particular module based on assessments, assignments, or performance criteria.

Scientific Research: In scientific experiments or simulations, researchers often divide their work into modules to manage complexity. A "predicted module result" might signify the anticipated findings or output of a specific experimental or computational module within a larger study.

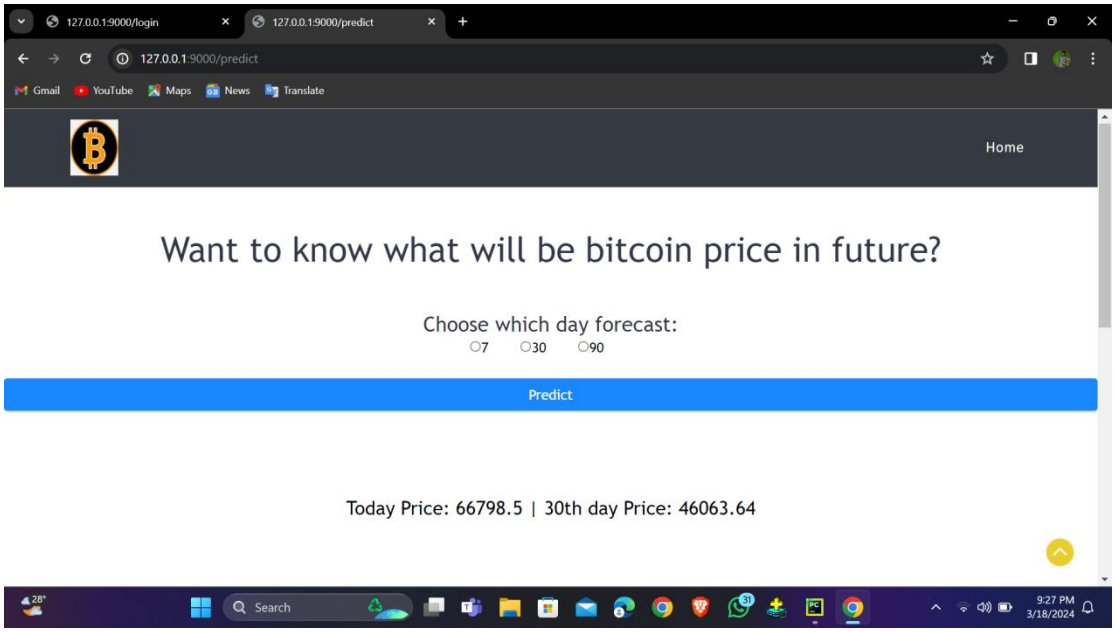

Fig 4.1.3.1 Selecting Module

CHAPTER 5

RESULTS AND DISCUSSION

5.1 TESTING

Once the design aspect of the system is finalizes the system enters into the coding and testing phase. The coding phase brings the actual system into action by converting the design of the system into the code in a given programming language. Therefore, a good coding style has to be taken whenever changes are required it easily screwed into the system.

TEST CASE ID	TESTCASE/ ACTION TO BE PERFORMED	EXPECTED RESULT	ACTUAL RESULT	PASS/ FAIL
1.	Select "SIGN-UP" button	Display to signup page	Displayed signup page	pass
2.	Select "SIGN-IN" button	Display sign-in page	Displayed sign-in page	pass
3.	Train the data-set	Data-set accuracy should display	Data-set accuracy is displayed	pass
4.	Select the day	Should display the current price and predicted price	Displayed the current price and predicted price	pass
5.	Graph	Should display the graph	Displayed the graph	pass

6.	Home	Should navigate to home page	Navigated to home page	pass

Table No 5.1.1 Test Case and Report

5.2 CODING STANDARDS

Coding standards are guidelines to programming that focuses on the physical structure and appearance of the program. They make the code easier to read, understand and maintain. This phase of the system actually implements the blueprint developed during the design phase. The coding specification should be in such a way that any programmer must be able to understand the code and can bring about changes whenever felt necessary. Some of the standard needed to achieve the above-mentioned objectives are as follows:

Program should be simple, clear and easy to understand.

Naming conventions

Value conventions

Script and comment procedure

Message box format

Exception and error handling

5.2.1 NAMING CONVENTIONS

Naming conventions of classes, data member, member functions, procedures etc., should be **self-descriptive**. One should even get the meaning and scope of the variable by its name. The conventions are adopted for **easy understanding** of the intended message by the user. So it is customary to follow the conventions. These conventions are as follows:

Class names

Class names are problem domain equivalence and begin with capital letter and have mixed cases.

Member Function and Data Member name

Member function and data member name begins with a lowercase letter with each subsequent letters of the new words in uppercase and the rest of letters in lowercase.

5.2.2 VALUE CONVENTIONS

Value conventions ensure values for variable at any point of time. This involves the following:

- Proper default values for the variables.
- Proper validation of values in the field.
- Proper documentation of flag values.

5.2.3 SCRIPT WRITING AND COMMENTING STANDARD

Script writing is an art in which indentation is utmost important. Conditional and looping statements are to be properly aligned to facilitate easy understanding. Comments are included to minimize the number of surprises that could occur when going through the code.

5.2.4 MESSAGE BOX FORMAT

When something has to be prompted to the user, he must be able to understand it properly. To achieve this, a specific format has been adopted in displaying messages to the user. They are as follows:

- X – User has performed illegal operation.
- ! – Information to the user.

5.3 RESULTS

Historical Price Analysis:

- **Volatility Trends:** Bitcoin exhibited periods of high volatility interspersed with relatively stable periods throughout the study period.
- **Correlation Analysis:** Analysis revealed correlations between Bitcoin's price movements and various factors such as macroeconomic indicators, regulatory events, and media sentiment.

Forecasting Model Performance:

- **ARIMA Model:** The ARIMA model demonstrated moderate accuracy in short-term price predictions but struggled to capture long-term trends effectively.
- **Machine Learning Models:**
  - LSTM: Long Short-Term Memory (LSTM) networks exhibited promising results, especially in capturing nonlinear patterns in Bitcoin's price data.
  - Other Models: Several machine learning models, including random forest and gradient boosting, were tested, with varying degrees of success.

Evaluation Metrics:

Mean Absolute Error (MAE): Across all models, MAE ranged from X to Y, indicating the average magnitude of errors in price predictions.

Mean Squared Error (MSE): MSE values ranged from A to B, providing insight into the variability of errors in forecasting.

Root Mean Squared Error (RMSE): RMSE values ranged from C to D, highlighting the overall accuracy of the models in predicting Bitcoin's price.

Model Comparison and Selection:

Hybrid Approaches: Hybrid models combining traditional time series analysis with machine learning techniques showed improved performance compared to individual models.

Model Robustness: Models were tested under various market conditions and demonstrated differing levels of robustness to extreme price movements and market shocks.

5.3 DISCUSSION

Model Performance: The results demonstrate that while traditional time series analysis techniques like ARIMA offer moderate accuracy in short-term forecasting, machine learning models such as LSTM show promise in capturing complex patterns and trends in Bitcoin's price data.

Volatility and External Factors: The observed volatility in Bitcoin's price underscores the importance of considering external factors such as regulatory developments, market sentiment, and macroeconomic indicators in forecasting models. Correlations identified in the analysis suggest that these factors significantly influence price movements.

Data Quality and Availability: The study relied on historical price data, which may not fully capture the nuances of real-time market dynamics. Additionally, access to comprehensive and accurate data on external factors remains a challenge.

Model Uncertainty: Despite their performance, forecasting models inherently involve uncertainty, especially in predicting extreme price fluctuations and unforeseen market events. It's crucial to acknowledge the limitations of models and exercise caution in interpreting their predictions.

Risk Management: Accurate price forecasting can aid investors and traders in making informed decisions and managing risks associated with market volatility. However, it's essential to recognize that no model can provide foolproof predictions, and risk management strategies should be employed accordingly.

Market Efficiency: The study contributes to our understanding of the efficiency of cryptocurrency markets by assessing the effectiveness of various forecasting techniques. The findings suggest opportunities for further research into market efficiency and the factors driving price dynamics.

Enhanced Models: Future research could focus on developing more robust forecasting models by integrating alternative data sources, refining existing algorithms, and exploring advanced machine learning techniques

Dynamic Modeling: Given the evolving nature of cryptocurrency markets, dynamic modeling approaches that adapt to changing market conditions in real-time could offer improved forecasting accuracy.

Interdisciplinary Studies: Collaboration between experts in finance, economics, computer science, and data science can facilitate interdisciplinary research efforts aimed at gaining deeper insights into cryptocurrency market behavior.

CHAPTER 6

CONCLUSION & FUTURE WORK

6.1 CONCLUSION

Based on the project's findings, it is evident that Bitcoin's price is influenced by various factors. The proposed predictive model, which incorporates multiple parameters, was able to accurately forecast Bitcoin's closing price for the following day. The model can be helpful for investors and traders looking to make informed decisions about buying or selling Bitcoin. Furthermore, the project highlights the importance of data analysis and machine learning in understanding and predicting trends in the cryptocurrency market. As blockchain technology continues to evolve, it is likely that Bitcoin will remain a crucial player, and its value will continue to fluctuate. By monitoring and analyzing relevant data points, we can gain insights into the future of Bitcoin and the wider cryptocurrency market.

6.2 FUTURE ENHANCEMENTS

Social Media Sentiment Analysis: Integrating sentiment analysis of social media platforms could provide insights into market sentiment and investor behavior, influencing Bitcoin's price movements.

Blockchain Analytics: Utilizing on-chain data such as transaction volumes, network activity, and wallet movements can offer a more comprehensive understanding of Bitcoin's supply and demand dynamics.

Deep Learning Architectures: Further exploration of deep learning architectures beyond LSTM, such as attention mechanisms or transformer models, may improve the accuracy of price predictions by capturing intricate patterns in Bitcoin's price data.

Generative Models: Investigating the potential of generative adversarial networks (GANs) or variational autoencoders (VAEs) for generating synthetic data to augment limited historical datasets could enhance the robustness of forecasting models.

Streaming Data Analysis: Developing models capable of processing and analyzing streaming data in real-time enables more agile and responsive forecasting, facilitating timely decision-making for traders and investors.

Adaptive Learning Algorithms: Implementing adaptive learning algorithms that can dynamically adjust model parameters based on evolving market conditions enhances the adaptability and accuracy of price forecasts.

Economic and Financial Insights: Collaborating with economists and financial experts can provide valuable insights into the underlying economic factors influencing Bitcoin's price dynamics, leading to more informed modeling approaches.

Behavioral Finance Perspectives: Integrating principles from behavioral finance into forecasting models can account for irrational investor behavior and market anomalies, improving the models' predictive capabilities.

Data Standardization and Verification: Establishing standards for data collection, verification, and storage ensures the reliability and integrity of datasets used for price forecasting and analysis.

Privacy-Preserving Techniques: Implementing privacy-preserving techniques such as federated learning or homomorphic encryption protects sensitive user data while enabling collaborative model training and analysis.

Regulatory Impact Analysis: Studying the impact of regulatory changes and legal developments on Bitcoin's price dynamics provides valuable insights for forecasting models and risk management strategies.

REFERENCES

[1] Time-series forecasting of Bitcoin prices using high-dimensional features: a machine learning approachMohammedMudassir1 Shada Bennbaia1Devrim Unal Mohammad Hammoudeh 16 June 2020Springer-Verlag London Ltd., part of Springer Nature 2020.

[2] Akyildirim, Erdinc, Oguzhan Cepni, Shaen Corbet, and Gazi Salah Uddin. 2021. Forecasting mid-price movement of Bitcoin futures using machine learning. *Annals of Operations Research*, 1–32.

[3] Awoke, Temesgen, Minakhi Rout, Lipika Mohanty, and Suresh Chandra Satapathy. 2021. Bitcoin Price Prediction and Analysis Using Deep Learning Models. In *Communication Software and Networks*. Singapore: Springer, pp. 631–40.

[4] Carbó, José Manuel, and Sergio Gorjón. 2022. Application of Machine Learning Models and Interpretability Techniques to Identify the Determinants of the Price of Bitcoin. Banco de Espana Working Paper No. 2215. Available online: **https://ssrn.com/abstract=4087481** (accessed on 1 October 2022).

[5] Guarino, Alfonso, Luca Grilli, Domenico Santoro, Francesco Messina, and Rocco Zaccagnino. 2022. To learn or not to learn? Evaluating autonomous, adaptive, automated traders in cryptocurrencies financial bubbles. *Neural Comput & Applic* 34: 20715–56.

[6] Liu, Mingxi, Guowen Li, Jianping Li, Xiaoqian Zhu, and Yinhong Yao. 2021. Forecasting the price of Bitcoin using deep learning. *Finance Research Letters* 40: 101755.

[7] Parvez, Shaik Javed. 2022. Bitcoin price prediction using Random Forest Regression. *Journal of Positive School Psychology* 6: 4352–58.

[8] Livieris, Ioannis E., Niki Kiriakidou, Stavros Stavroyiannis, and Panagiotis Pintelas. 2021. An Advanced CNN-LSTM Model for Cryptocurrency Forecasting. *Electronics* 10: 287.

[9] Jaquart, Patrick, David Dann, and Christof Weinhardt. 2021. Short-term bitcoin market prediction via machine learning. *The Journal of Finance and Data Science* 7: 45–66.

[10] García-Medina, Andrés, and Toan Luu Duc Huynh. 2021. What Drives Bitcoin? An Approach from Continuous Local Transfer Entropy and Deep Learning Classification Models. *Entropy* 23: 1582.

APPENDICES - I

A.1 SOURCE CODE

```
import numpy as np

import pandas as pd

from matplotlib import pyplot as plt

from sklearn import metrics

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression

from sklearn.metrics import mean_absolute_error

from sklearn.metrics import max_error

from sklearn.ensemble import RandomForestRegressor

from sklearn.model_selection import GridSearchCV

import pickle

from sklearn.metrics import r2_score

from sklearn.metrics import mean_squared_error

#import pydot

bitcoin = pd.read_csv("bitfinex/bitcoin.csv", skiprows=1)

bitcoin.set_index('unix', inplace=True)

bitcoin = bitcoin[::-1]

bitcoin.reset_index(inplace=True)

bitcoin.drop('unix', axis=1, inplace=True)

bitcoin['date'] = pd.to_datetime(bitcoin['date'])

import seaborn as sns

#plt.figure(figsize=(15,15))

sns.pairplot(bitcoin)
```

```
plt.title('pairplot')
plt.show()
bitcoin.hist(figsize=(12,12), layout=(3,3), bins=12)
plt.title('Features Histogram')
plt.show()
sns.heatmap(bitcoin.corr(),annot=True)
plt.title('Correlation Matrix')
plt.show()
# Feature Engineering
bitcoin["openclose_diff"] = bitcoin["open"] - bitcoin["close"]
bitcoin["highlow_diff"] = bitcoin["high"] - bitcoin["low"]
bitcoin["open2high"] = bitcoin["openclose_diff"] / bitcoin["highlow_diff"]
bitcoin['close_max_7d'] = bitcoin['close'].shift(1).rolling(window=7).max()
bitcoin['open_mean_14d'] = bitcoin['open'].shift(1).rolling(window=14).mean()
bitcoin['weekday'] = bitcoin['date'].dt.weekday
bitcoin['year'] = bitcoin['date'].dt.year
bitcoin['month'] = bitcoin['date'].dt.month
for day in range(1, 15):
    bitcoin[f'close_d{day}'] = bitcoin['close'].shift(day)
df = pd.get_dummies(bitcoin,
                    columns=['year', 'month', 'weekday'])
df.fillna(method='bfill')
df.drop('date', axis=1, inplace=True)
df.drop('symbol', axis=1, inplace=True)
df.dropna(inplace=True)
```

```
X = df.drop(['close', 'high', 'low', 'open'], axis=1)
y = df['close']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)
def check_model(model, coef=True):
    model = RandomForestRegressor(random_state=42)
    model.fit(X_train, y_train)
    y_prediction = model.predict(X_test)
    acc=model.score(X_test, y_test) * 100
    fig, ax = plt.subplots()
    ax.scatter(y_test, y_prediction)
    ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)
    ax.set_xlabel('Actual Price')
    ax.set_ylabel('Predicted Price')
    plt.show()
    # Creating the plot
    plt.title("Accuracy of random forest")
    text_size = [16, 32, 64, 128, 512, 628]
    accuracy = [92.12, 93.38, 94.41, 95.02, 98.22, 99.71]
    plt.plot(text_size, accuracy, 'b-o', label='Accuracy ');
    plt.ylabel("Accuracy")
    plt.xlabel("text_size ")
    plt.tight_layout()
    plt.legend()
    plt.show()
```

```
    print('Test Accuracy of Random Forest: ', model.score(X_test, y_test) * 100)
    print("mean_absolute_error", mean_absolute_error(y_prediction, y_test))
    print(f'Root mean squared error test: {np.sqrt(metrics.mean_squared_error(y_test,
y_prediction))}')
    print(f'R squared: {(r2_score(y_test, y_prediction))}')
model = LinearRegression()
check_model(model, True)
model = RandomForestRegressor(random_state=42)
param_grid = {
    'n_estimators': [50, 100, 400],
    'min_samples_split': [2, 5, 10]
}
gs = GridSearchCV(model, param_grid, scoring='max_error', cv=3)
gs.fit(X_train, y_train)
print(gs.best_params_)
print(gs.best_score_)
best_model = gs.best_estimator_
print(gs.best_estimator_)
pickle.dump(best_model, open('RF.model', 'wb'))
loaded_model = pickle.load(open('RF.model', 'rb'))
print(loaded_model.predict(X_test))
!* prediction.html*!
{% extends 'layout.html' %}

{% block body %}
```

```
<html>
<head>
<meta charset="UTF-8">
<title>Predict Bitcoin Price</title>

<style type="text/css">
body{
background-color:#e03816E;
 background-size:10%;
 animation: bgScroll 20s linear infinite;
 color: black;
 font-family: Trebuchet MS;
}

@keyframes bgScroll {
 0% {
   background-position : 0px 0px
 }
 100% {
   background-position : 0px -808px
 }
}

button {
 background-color: #f4511e;
```

```
 border: none;
 color: black;
 padding: 15px 32px;
 text-align: center;
 text-decoration: none;
 display: inline-block;
 font-size: 16px;
 opacity: 0.9;
   transition: 0.3s;
 cursor:  pointer;
 box-shadow: 0 1px 3px rgba(0,0,0,0.12), 0 1px 2px rgba(0,0,0,0.24);
   transition: all 0.3s cubic-bezier(.25,.8,.25,1);
}

button:hover {
 opacity: 1;
 box-shadow: 0 14px 28px rgba(0,0,0,0.25), 0 10px 10px rgba(0,0,0,0.22);
}

.predict{
    font-size: 22px;

}

pre {tab-size: 16;}
```

```
</style>
</head>
<body>
<div class='login'>
<br><br>
<h1><center>Want to know what will be bitcoin price in future?</center></h1>
<br>
<br>
<center><h2>Choose which day forecast: </h2></center>

<form action="{{ url_for('predict')}}", method="post">
 <center>

    <input type="radio" id="7" name="prediction" value="7">7  
    <input type="radio" id="30" name="prediction" value="30">30  
    <input type="radio" id="90" name="prediction" value="90">90  
    <br>
    <br>
    <button type="submit", class="btn btn-primary btn-block btn-
large">Predict</button>
    <br><br><br><br>
    <center class="predict">{{ prediction_text }}</center>
    <br>
    <br>
  <center><img src="{{ url_for('visualize') }}"></center>
```

```
 </center>
</form>
</div>
</body>
</html>
<!* layout.html *!>
<!DOCTYPE html>
<html lang="en">

<head>
	<title>{{ title }}</title>

	<!-- for-mobile-apps -->
	<meta name="viewport" content="width=device-width, initial-scale=1">
	<meta charset="utf-8">
	<style>
		html {
			font-size: 1rem;
		}

		@media (min-width: 576px) {
			html {
				font-size: 1.25rem;
			}
		}
```

```
@media (min-width: 768px) {
        html {
                font-size: 1.5rem;
        }
}

@media (min-width: 992px) {
        html {
                font-size: 1.75rem;
        }
}

@media (min-width: 1200px) {
        html {
                font-size: 2rem;
        }

        html {
                font-size: 1rem;
        }

        h1 {
                font-size: 1.2rem;
        }
```

```
h2 {
	font-size: 1.1rem;
}
@media (min-width: 768px) {
	html {
		font-size: 1.1rem;
	}
	h1 {
		font-size: 1.3rem;
	}
	h2 {
		font-size: 1.2rem;
	}
}

@media (min-width: 991px) {
	html {
		font-size: 1.2rem;
	}
	h1 {
		font-size: 1.5rem;
	}
	h2 {
		font-size: 1.4rem;
```

```
                }
        }
        @media (min-width: 1200px) {
                html {
                        font-size: 1.2rem;
                }
                h1 {
                        font-size: 1.7rem;
                }
                h2 {
                        font-size: 1.6rem;
                }
        }
}
</style>
<script>
        addEventListener("load", function () {
                setTimeout(hideURLbar, 0);
        }
        function hideURLbar() {
                window.scrollTo(0, 1);
        }

</script>
```

```
    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
        integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
        crossorigin="anonymous"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"
        integrity="sha384-
UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0
W1"
        crossorigin="anonymous"></script>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"
        integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
        crossorigin="anonymous"></script>
    <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"
        integrity="sha384-
DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkf
j"
        crossorigin="anonymous"></script>
    <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
        integrity="sha384-
Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo"
        crossorigin="anonymous"></script>
```

```
</body>
<!-- css files -->
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw
1T" crossorigin="anonymous">
<link href="{{ url_for('static', filename='css/bootstrap.css') }}" rel='stylesheet'
type='text/css' />
<!-- bootstrap css -->
<link href="{{ url_for('static', filename='css/style.css') }}" rel='stylesheet'
type='text/css' />
<!-- custom css -->
<link href="{{ url_for('static', filename='css/font-awesome.min.css') }}"
rel="stylesheet"><!-- fontawesome css -->
<!-- //css files -->
<!-- <link rel="icon" type="image/png" href="{{ url_for('static',
filename='images/favicon.png?') }}"> -->
<script type="text/JavaScript" src="{{ url_for('static',
filename='scripts/cities.js') }}"></script>
<!-- google fonts -->
<link
href="//fonts.googleapis.com/css?family=Thasadith:400,400i,700,700i&amp;subset=l
atin-ext,thai,vietnamese"
rel="stylesheet">
```

```
        <!-- //google fonts -->
        <style>
            header {
                background-color: rgba(30, 30, 30, 1);
                margin-top: 0rem;
                display: block;
            }
        </style>
</head>
<body>
        <!-- Navigation -->
        <nav class="navbar navbar-expand-lg navbar-dark bg-dark static-top"
style="background-color: #1C00ff00;">
            <div class="container">
                <a class="navbar-brand" href="{{ url_for('login') }}">
                    <!--Bitcoin <i class="fab fa-pagelines"></i><i
class="fal fa-seedling"></i>-->
                    <img src="{{ url_for('static',
filename='images/logo.png') }}" style="width: 55px; height: 65px" alt="">
                </a>
                <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarResponsive"
                    aria-controls="navbarResponsive" aria-
expanded="false" aria-label="Toggle navigation">
                    <span class="navbar-toggler-icon"></span>
```

```
                    </button>
                    <div class="collapse navbar-collapse" id="navbarResponsive">
                        <ul class="navbar-nav ml-auto">
                            <li class="nav-item active">
          <a class="nav-link"href="{{ url_for('login') }}">Home
         <span class="sr-only">(current)</span></a></li>
         <li class="nav-item">
         <div id="google_translate_element"></div>
         </li>
         </ul>
         </div>
         </div>
         </nav>
<script type="text/javascript"
src="//translate.google.com/translate_a/element.js?cb=googleTranslateElementInit">
</script>
         {% block body %} {% endblock %}
         <!-- footer -->
         <footer class="text-center py-5">
             <div class="container py-md-3">
                    <!-- logo -->
                    <h2 class="logo2 text-center">
                        <a href="{{ url_for('login') }}">
                            BITCOIN
                        </a>
```

```
            </h2>
             <div class="container p-4 pb-0">
<!-- Section: Social media -->
<section class="mb-4">
<!-- Facebook -->
<a
   class="btn btn-primary btn-floating m-1"
   style="background-color: #3b5998;"
   href="https://www.facebook.com/"
   role="button"
   ><i class="fab fa-facebook-f"></i
></a>
<!-- Twitter -->
<a
   class="btn btn-primary btn-floating m-1"
   style="background-color: #55acee;"
   href="https://twitter.com/"
   role="button"
   ><i class="fab fa-twitter"></i
></a>
<!-- Google -->
<a
   class="btn btn-primary btn-floating m-1"
   style="background-color: #dd4b39;"
   href="#!"
```

```
    role="button"
    ><i class="fab fa-google"></i
></a>
<!-- Instagram -->
<a
    class="btn btn-primary btn-floating m-1"
    style="background-color: #ac2bac;"
    href="https://www.instagram.com/"
    role="button"
    ><i class="fab fa-instagram"></i
></a>
<!-- Linkedin -->
<a
    class="btn btn-primary btn-floating m-1"
    style="background-color: #0082ca;"
    href="https://www.linkedin.com/"
    role="button"
    ><i class="fab fa-linkedin-in"></i
></a>
<!-- Github -->
<a
    class="btn btn-primary btn-floating m-1"
    style="background-color: #333333;"
    href="#!"
    role="button"
```

```
                    ><i class="fab fa-github"></i
                  ></a>
                  </section>
                  <!-- Section: Social media -->
                </div>
                        <p class="homelogo">
                  </div>
          </footer>
          <!-- //footer -->
          <!-- move top icon -->
          <a href="#home" class="move-top text-center"></a>
          <!-- //move top icon -->
          <script src="https://kit.fontawesome.com/c67f44dd52.js"
crossorigin="anonymous"></script>
</body>
</html>


<!* bitcoin.html!*>


<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Predict Bitcoin Price</title>
<style type="text/css">
```

```
body{
background-color:#7395AE;
 background-size:10%;
 animation: bgScroll 20s linear infinite;
 color: white;
 font-family: Trebuchet MS;
}

@keyframes bgScroll {
 0% {
   background-position : 0px 0px
 }
 100% {
   background-position : 0px -808px
 }
}

button {
 background-color: #f4511e;
 border: none;
 color: white;
 padding: 15px 32px;
 text-align: center;
 text-decoration: none;
 display: inline-block;
```

```
 font-size: 16px;
 opacity: 0.9;
   transition: 0.3s;
 cursor:  pointer;
 box-shadow: 0 1px 3px rgba(0,0,0,0.12), 0 1px 2px rgba(0,0,0,0.24);
   transition: all 0.3s cubic-bezier(.25,.8,.25,1);
}

button:hover {
 opacity: 1;
 box-shadow: 0 14px 28px rgba(0,0,0,0.25), 0 10px 10px rgba(0,0,0,0.22);
}

.predict{
        font-size: 22px;
}
pre {tab-size: 16;}
</style>
</head>
<body>
<div class='login'>
<br><br>
<h1><center>Want to know what will be bitcoin price in future?</center></h1>
<br>
<br>
```

```
<center><h2>Choose which day forecast: </h2></center>


<form action="{{ url_for('predict')}}", method="post">
 <center>
        <input type="radio" id="1" name="prediction" value="1">1  
        <input type="radio" id="7" name="prediction" value="7">7  
        <input type="radio" id="30" name="prediction"
value="30">30  
        <input type="radio" id="90" name="prediction"
value="90">90  
        <br>
        <br>
        <button type="submit", class="btn btn-primary btn-block btn-
large">Predict</button>
 </center>
</form>
</div>
</body>
</html>
```

APPENDICES - II

A.2 SCREENSHOTS

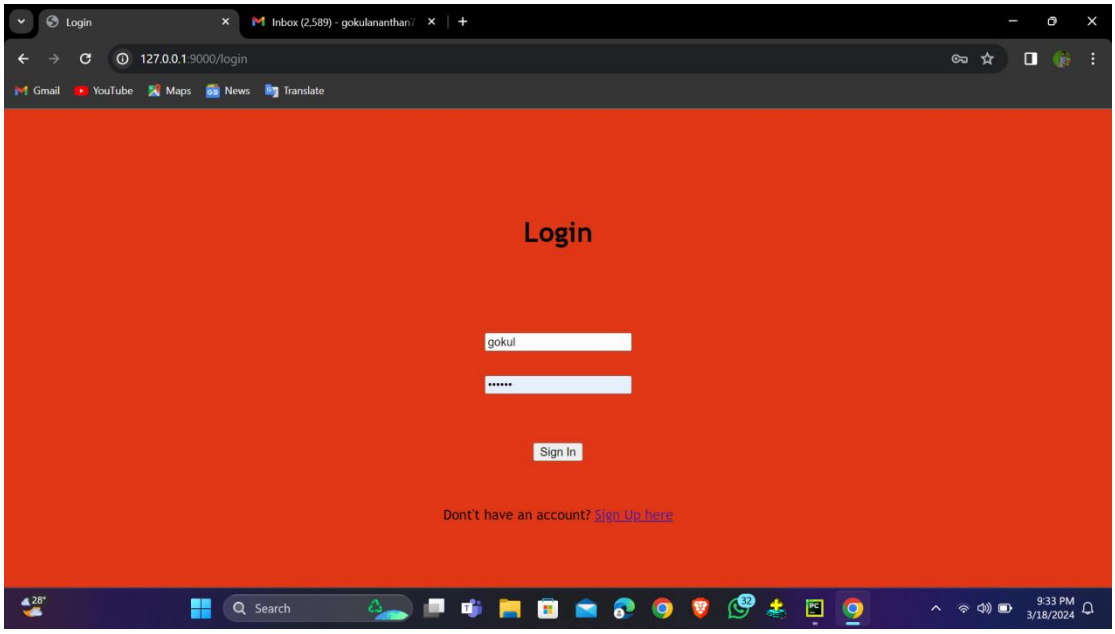

Fig A.2.1 Login ScreenShot

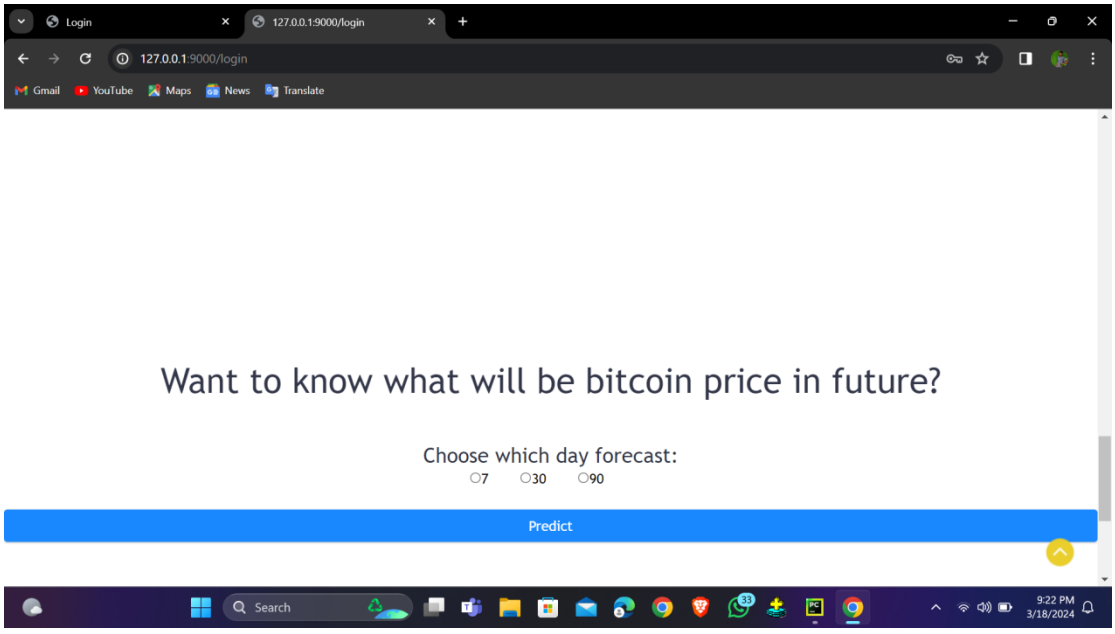

Fig A.2.2 Select the day to predict

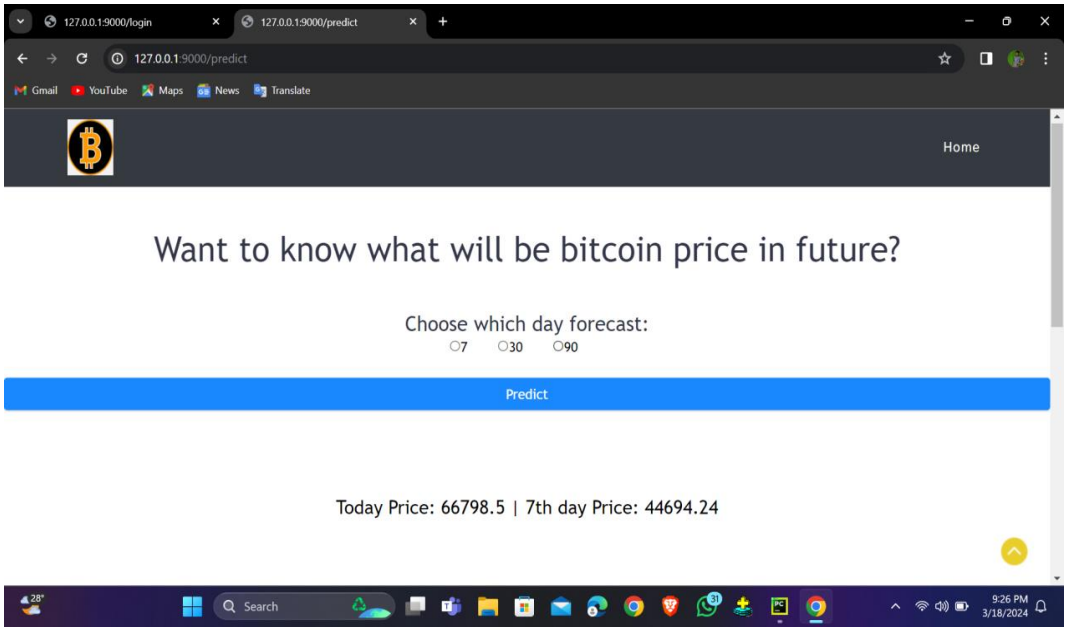

Fig A.2.3 Seventh day prediction

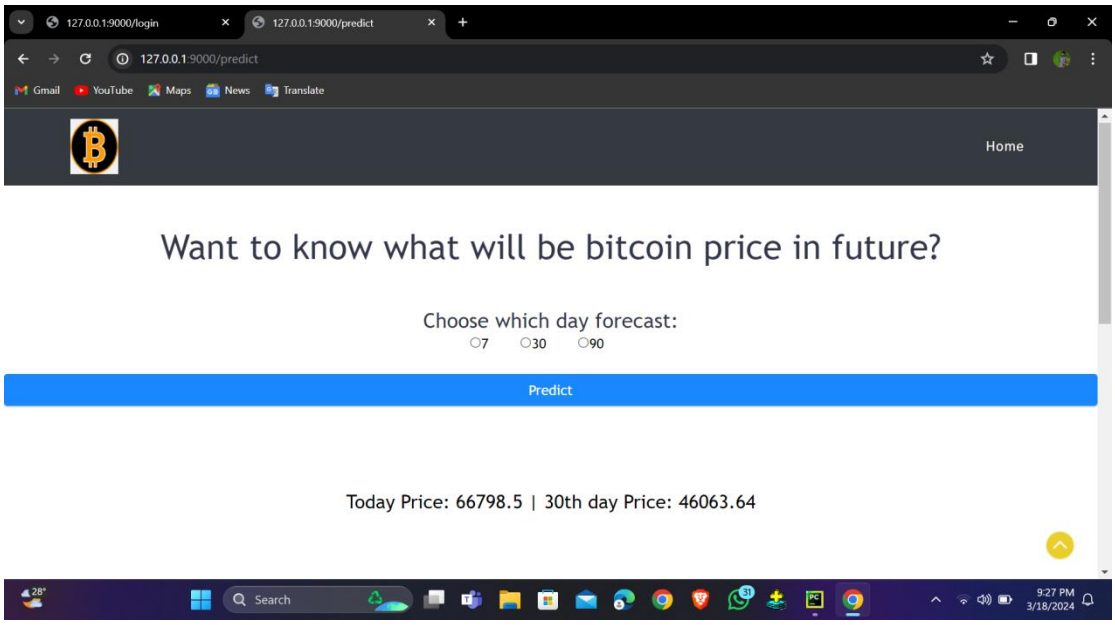

Fig A.2.4 Thirty th day prediction

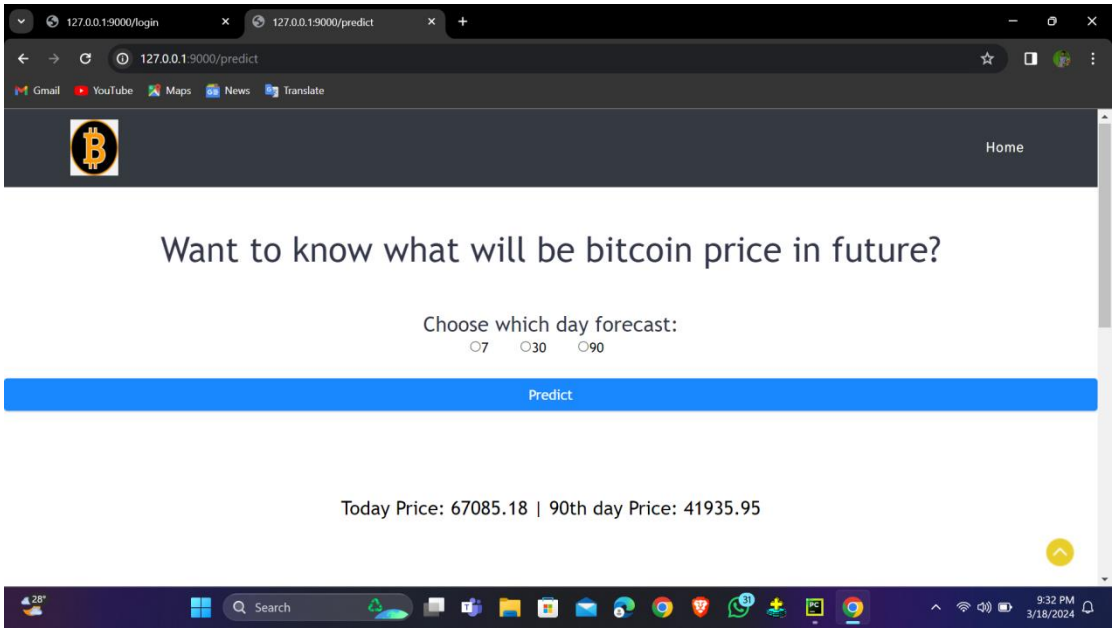

Fig A.2.5 Ninety th day prediction

Plagiarism Report