

Stock Price Prediction System

By

**Mr. Ayush Dilip Padvekar (20220804015)**

**Mr. Shubham Manish Lotwala (20220804034)**

**Mr. Shritej Suresh Deshpande (20220804058)**

**Under Guidance Of**

## Mrs. Veena Kiragi

**Submitted to**

## School of Computer Science, Engineering and Application

**In partial fulfilment of the requirements**

**for the award of the degree**

# Master of Computer Applications (MCA)

**2022 - 2023**



**School of Computer Science, Engineering and Applications**

# CERTIFICATE OF COMPLETION

This is to certify that the project report entitled **Stock Price Prediction System** submitted to **School of Computer Science, Engineering and Application, *D Y Patil International University*** in partial fulfilment of the requirements for the ***Project- I course,* SEM II** of the degree of **Master of Computer Applications (MCA),** is an original work carried out by **Mr. Ayush Padvekar (20220804015), Mr. Shubham Lotwala (20220804034)** and **Mr. Shritej Deshpande** (**20220804058**) under my guidance.

The matter embodied in this project is genuine work done by the student and has not been submitted whether to this university or to any other university for the fulfilment of the requirements of any course of study.

**Signature of the student**: **Signature of the Guide**:

Date:

Mr. Ayush Padvekar (20220804015)

Mr.Shubham Lotwala (20220804034)

Mr.Shritej Deshpande (20220804058)

Date: Mrs. Veena Kiragi

School of Computer Science, Engineering and Application

**Signature of the project In-charge**: Date:

Mrs. Sarika Jadhav

School of Computer Science,

Engineering and Application

**Signature of the HOD**: **Signature of the Director**:

Date:

Dr. Maheshwari Biradar

School of Computer Science, Engineering and Application

Date:

Dr. Bahubali Shiragapur

School of Computer Science, Engineering and Application

# ABSTRACT

In this project we attempt to implement machine learning approach to predict stock prices. Machine learning is effectively implemented in forecasting stock prices. The objective is to predict the stock prices in order to make more informed and accurate investment decisions. We propose a stock price prediction system that integrates mathematical functions, machine learning, and other external factors for the purpose of achieving better stock prediction accuracy and issuing profitable trades.

There are two types of stocks. You may know of intraday trading by the commonly used term "day trading". Intraday traders hold securities positions from at least one day to the next and often for several days to weeks or months. LSTMs are very powerful in sequence prediction problems because they’re able to store past information. This is important in our case because the previous price of a stock is crucial in predicting its future price. While predicting the actual price of a stock is an uphill climb, we can build a model that will predict whether the price will go up or down.

# ACKNOWLEDGEMENT

It gives us immense pleasure to express our deepest sense of gratitude and sincere thanks to our highly respected and esteemed guide **Mrs. Veena Kiragi, School of Computer Science, Engineering and Application,** for her valuable guidance, encouragement, and help in completing this work. Her useful suggestions for this whole work and cooperative behaviour are sincerely acknowledged. I would like to express our sincere thanks to the project in-charge **Mrs. Sarika Jadhav,** **School of Computer Science, Engineering and Application** and Head of department **Dr. Maheshwari Biradar, School of Computer Science, Engineering and Application** for giving us this opportunity to undertake this project. Also wish to express our gratitude to the director **Dr. Bahubali Shiragapur, School of Computer Science, Engineering and Application** for his kind-hearted support**.**

We are also grateful to our faculty members for their constant support and guidance. We also wish to express our indebtedness to our parents as well as our family member whose blessings and support always helped us to face the challenges ahead. At the end we would like to express our sincere thanks to all our friends and others who helped us directly or indirectly during this project work.

Place: Students Name:

Mr. Ayush Padvekar (20220804015)

Mr.Shubham Lotwala (20220804034)

Mr.Shritej Deshpande (20220804058)

Date:

**Table of Contents Page No.**

CHAPTER 1: INTRODUCTION1-3

1.1 Background1

1.2 Objectives 1

1.3 Purpose 2

1.4 Scope 2

1.5 Applicability 3

CHAPTER 2: SURVEY OF TECHNOLOGIES4

CHAPTER 3: REQUIREMENTS AND ANALYSIS5-30

3.1 Problem Definition 5

3.2 Requirements Specification6

3.3 Feasibility Study7

3.4 Planning and Scheduling 8

3.5 Software and Hardware Requirements 9

3.6 Preliminary Product Description9

3.7 Conceptual Models / Design Documents10

3.7.1 DFD10

A. 0-Level DFD10

B. 1-Level DFD11

C. 2-Level DFD12

3.7.2 Flow Chart13

3.7.3 UML14

A. Use Case Diagram14

B. Activity Diagram15

C. Sequence Diagram16

D. Pseudocodes18

E. Decision Table29

F. Decision Tree30

CHAPTER 4: SYSTEM DESIGN31-39

4.1 Basic Modules31

CHAPTER 5: IMPLEMENTATION AND TESTING42-43

5.1 Implementation Approaches40

5.2 Coding Details and Code Efficiency41

5.3 Testing Approach 43

5.3.1 Unit Testing 43

5.3.2 Integrated Testing43

CHAPTER 6: RESULTS AND DISCUSSION44-46

6.1 User Documentation44

6.2 Test Cases46

CHAPTER 7: CONCLUSIONS47-48

7.1 Conclusion47

7.2 Limitations of the System48

7.3 Future Scope of the Project48

CHAPTER 8: REFERENCE49

Table of Figures Page No.

Figure 1: Gantt Chart8

Figure 2: 0 - LEVEL DFD10

Figure 3: 1 - LEVEL DFD11

Figure 4: 2 - LEVEL DFD12

Figure 5: Flowcharts13

Figure 6: Use Case Diagram14

Figure 7: Activity Diagram15

Figure 8: Sequence Diagram for Accuracy Prediction16

Figure 9: Sequence Diagram for Stock Comparison17

Figure 10: Sequence Diagram for Dashboard 17

Figure 11: Decision Table29

Figure 12: Decision Tree30

Figure 13: HOME PAGE31

Figure 14: Input Prediction Page32

Figure 15: Output Prediction Page33

Figure 16: Input Data Accuracy Page34

Figure 17: Output Data Accuracy Page35

Figure 18: Input Stock Comparison Page 36

Figure 19: Output Stock Comparison Page 37

Figure 20: Input Dashboard Page 38

Figure 21: Output Dashboard Page 39

Figure 22: Test Case 46

**CHAPTER 1**

**INTRODUCTION**

* 1. **Background**

The stock market is a dynamic and composite system where investors can buy and sell currencies, stocks, equities and derivatives over virtual platforms supported by brokers. Automated trading systems (ATS) are operated by computer programs and can perform better than human orders. To evaluate and control the performance of ATSs, risk strategies and safety measures are required. Time-series prediction is a common technique used in many real-world applications such as weather forecasting and financial market prediction, and is based on Recurrent Neural Networks (RNN) and Long-short Term Memory (LSTM) and Gated Recurrent Unit (GRU). In this project, LSTM model is used to predict the stock price.

* 1. **Objectives**

1. Stock market prediction is an action of attempting to determine the future value of a company stock or other financial investment traded on a stock exchange. The yield significant profit is the resulting price of a stock, helping investors invest wisely to make good profits.
2. To provide investors with an understanding of the stock market and how to make informed decisions.
3. To develop a model that accurately predicts stock prices in the short-term and long-term.
   1. **Purpose**

The purpose of a stock price prediction system is to analyze historical stock price data, as well as current market trends and other relevant data, to forecast future stock prices. This can be valuable for investors who are trying to make informed decisions about buying or selling stocks. By using a stock price prediction system, investors may be able to identify stocks that are likely to increase in value, as well as those that are likely to decline in value. However, it's important to note that stock price prediction systems are not foolproof and cannot guarantee accurate predictions, as the stock market is highly unpredictable and subject to numerous factors that can impact stock prices.

* 1. **Scope**

A stock price prediction system can be used to predict the future price of a stock based on historical data. The system can be used by investors to make informed decisions about when to buy and sell stocks. It can also be used by financial institutions to identify potential investment opportunities. The system can also be used to identify trends in the stock market and to generate trading signals. Finally, the system can be used to analyze the performance of different stocks and to identify potential risks and rewards associated with investing in them.

1. Generating trading signals based on the analysis of historical data.
2. Identifying potential investment opportunities.
3. Predicting future stock prices based on historical data.
4. Analyzing the performance of different stocks.
5. Identifying potential risks and rewards associated with investing in stocks.
6. Improving the accuracy of stock price predictions.
7. Automating the process of stock trading.
8. Providing personalized advice for investors.
   1. **Applicability**

Several sectors of finance and investing can benefit from the use of a stock price prediction system, including:

1. **Investment management:** To make educated decisions regarding purchasing and selling stocks in their portfolios, investment managers can employ stock price prediction algorithms.
2. **Trading:** To find trading opportunities and decide whether to enter or leave deals, traders might use stock price prediction algorithms.
3. **Risk management:** To detect potential dangers and take action to mitigate them, risk managers might employ stock price prediction algorithms.
4. **Research and analysis:** To examine market trends and get insight into how different stocks and sectors behave, researchers and analysts might use stock price prediction algorithms.
5. **Portfolio optimisation:** By choosing stocks that are likely to offer the best returns while minimising risk, portfolio managers can use stock price prediction systems to optimise their holdings.

**CHAPTER 2**

**SURVEY OF TECHNOLOGIES**

Systems for predicting stock prices employ a variety of technologies and methods, including:

1. **Machine learning**: Neural networks and decision trees are two examples of machine learning techniques that are frequently used to analyse historical stock price data and find patterns that may be utilised to generate forecasts.
2. **Natural language processing:** To discover patterns and sentiment that could affect stock prices, natural language processing (NLP) techniques can be used to analyse news articles, social media messages, and other sources of text data.
3. **Data visualisation:** Charts and graphs are examples of data visualisation tools that may be used to assist investors see and comprehend trends in stock prices and other market data.
4. **Cloud computing:** By storing and processing vast volumes of data, cloud computing platforms like Amazon Web Services and Google Cloud Platform make it simpler to analyse and make predictions.
5. **Big data analytics:** Historical stock price data and other market data may be processed and analysed using big data analytics tools like Apache Hadoop and Apache Spark.
6. **Sentiment analysis:** To ascertain the general sentiment towards a specific stock or firm, sentiment analysis techniques can be employed to examine posts on social media and other data sources.
7. **Deep learning:** By analysing time-series data and making predictions about future stock values, deep learning techniques like convolutional neural networks (CNNs) and recurrent neural networks (RNNs) can be used.

**CHAPTER 3**

**REQUIREMENTS AND ANALYSIS**

* 1. **Problem Definition**

The challenge of accurately forecasting future stock prices is the main issue that a stock price prediction system seeks to address. The stock market is extremely intricate and influenced by a variety of variables, such as economic trends, corporate performance, current affairs, and geopolitical developments. As a result, it may be difficult for investors to decide whether to buy or sell stocks with knowledge.

In order to find patterns and trends that can be utilised to anticipate future stock prices, stock price prediction systems analyse historical stock price data as well as other pertinent market data. The system processes and analyses massive volumes of data using cutting-edge algorithms and techniques, including machine learning and deep learning, and then makes predictions based on that analysis.

But creating and putting in place a system for predicting stock prices comes with a number of difficulties. These include the potential for errors or inaccurate predictions, the complexity of the stock market, the availability and quality of data, the need to continuously update and improve algorithms and models. Furthermore, there is no assurance that a stock price prediction system would make correct forecasts because the stock market is incredibly unpredictable and influenced by a variety of outside factors.

* 1. **Requirements Specification**

A stock price prediction system's requirements specification may include the following:

1. **Data sources:** For information on past stock prices, market trends, current events, and other pertinent data, the system requires access to dependable and high-quality data sources.
2. **Algorithm selection:** To analyse the data and generate predictions, the system should make use of cutting-edge algorithms and methodologies including machine learning, deep learning, and natural language processing.
3. **Model training:** To increase the precision of predictions, the system should be able to train models utilising past data.
4. **Real-time processing:** The system must be able to process data in real-time and modify forecasts in response to new information.
5. **User interface:** Users should be able to view predictions, examine data, and come to wise conclusions using a system that has an intuitive and user-friendly interface.
6. **Accuracy:** The system should strive to provide predictions that are accurate and have a small margin of error.
7. **Security:** To safeguard sensitive data and stop unauthorised access, the system should be built with robust security measures.
8. **Scalability:** The system should be scalable, meaning it should be able to accommodate changing user and data volume loads.
9. **Upkeep:** The system needs to be easy to maintain and upgrade, with little downtime.
   1. **Feasibility Study**

A stock price prediction system feasibility study should determine whether it is feasible and doable to construct such system. A feasibility study may take the following variables into account:

**Technical viability:** A system for predicting stock prices needs access to a lot of data and sophisticated algorithms and methodologies, such deep learning and machine learning. The required technology should be evaluated for availability and suitability for integration into the organization's current infrastructure as part of the feasibility assessment.

**Data accessibility**: In order to obtain past stock prices, market trends, current news events, and other pertinent data, the system must have access to reputable and high-quality data sources. The availability, accessibility, and cost of such data should all be evaluated as part of the feasibility study.

**Market demand:** Investors, traders, and other financial professionals may find value in a stock price forecasting system. The prospective market size and demand for such a system should be assessed in the feasibility study.

**Cost:** Creating and implementing a system for predicting stock prices can be pricey. The development, implementation, and maintenance expenses of the system should be evaluated, as well as if the prospective rewards outweigh the expenditure.

Legal and ethical difficulties are raised when data and algorithms are used to predict the future of the financial markets. These issues include data privacy, bias, and transparency. These concerns and the organization's ability to adhere to moral and legal obligations should be taken into account in the feasibility assessment.

* 1. **Planning and Scheduling**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Month  Activity | FEB | | | | MAR | | | | APR | | | |
| Weeks | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| Research |  |  |  |  |  |  |  |  |  |  |  |  |
| Define Specification |  |  |  |  |  |  |  |  |  |  |  |  |
| Project planning |  |  |  |  |  |  |  |  |  |  |  |  |
| Design |  |  |  |  |  |  |  |  |  |  |  |  |
| Development |  |  |  |  |  |  |  |  |  |  |  |  |
| Testing |  |  |  |  |  |  |  |  |  |  |  |  |
| Documentation |  |  |  |  |  |  |  |  |  |  |  |  |

Estimated time:

Actual time:

**Figure 1: Gant Chart**

* 1. **Software and Hardware Requirements**
* **Hardware Requirement:**
  + - Any web enabled device.
    - 64-bit OS.
    - 512 MB RAM

* **Software Requirement:**
  + - Python Language
    - Libraries
    - Pandas, Numpy, Sklearn, Tensorflow, Streamlit. Etc
  1. **Preliminary Product Description**

In order to predict future stock prices, a software programme called the stock price prediction system employs machine learning algorithms to examine past stock price data and other pertinent financial data. To produce precise forecasts, the system will combine data from a range of sources, including financial statements, news items, and market movements.

The system will be created to offer customers accurate and current stock price forecasts for a variety of equities and markets. Users will be able to enter the names of individual equities they want to follow, and the system will produce price predictions for those stocks in the future.

Additionally, the system will be built to send consumers instant notifications of important shifts in stock prices or market patterns. Users will be able to stay up to date on significant stock market changes and make better financial decisions thanks to this functionality.

The stock price prediction system will, in general, be a useful tool for traders and investors trying to make data-driven investment decisions based on precise and trustworthy market projections.

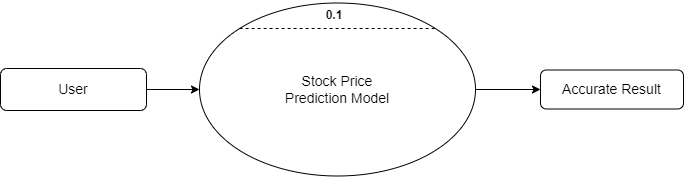
* 1. **Conceptual Models / Design Documents**

**3.7.1 DFD**

In Software engineering DFD (data flow diagram) can be drawn to represent the system of different levels of abstraction. Higher-level DFDs are partitioned into low levels-hacking more information and functional elements. Levels in DFD are numbered 0, 1, 2 or beyond. Here, we will see mainly 3 levels in the data flow diagram, which are: 0-level DFD, 1-level DFD, and 2-level DFD.

1. **0 - LEVEL DFD**

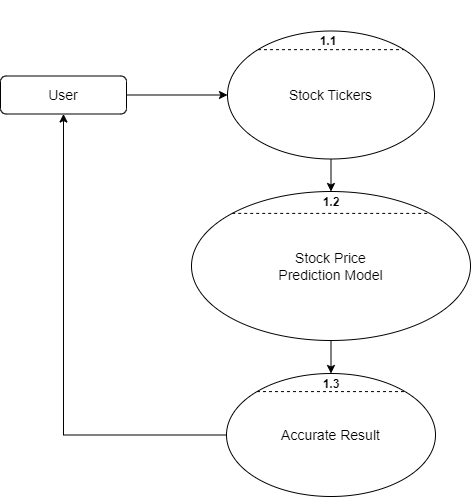
It is also known as a context diagram. It’s designed to be an abstraction view, showing the system as a single process with its relationship to external entities. It represents the entire system as a single bubble with input and output data indicated by incoming/outgoing arrows.



**Figure 2: 0 - LEVEL DFD**

1. **1 - LEVEL DFD**

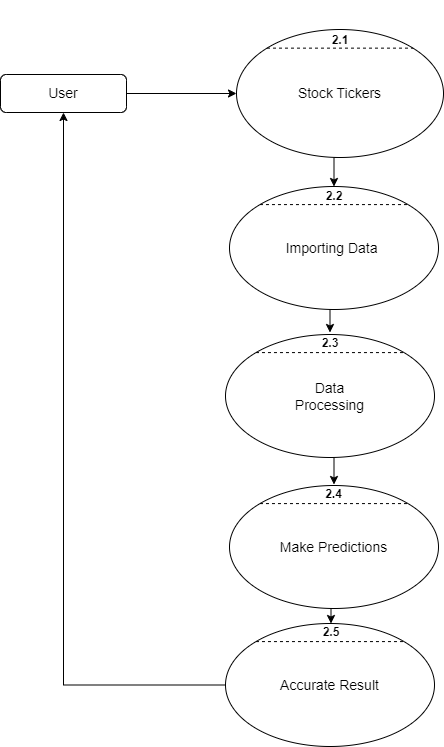
In 1-level DFD, the context diagram is decomposed into multiple bubbles/processes. In this level, we highlight the main functions of the system and breakdown the high-level process of 0-level DFD into subprocesses.



**Figure 3: 1 - LEVEL DFD**

1. **2 - LEVEL DFD**

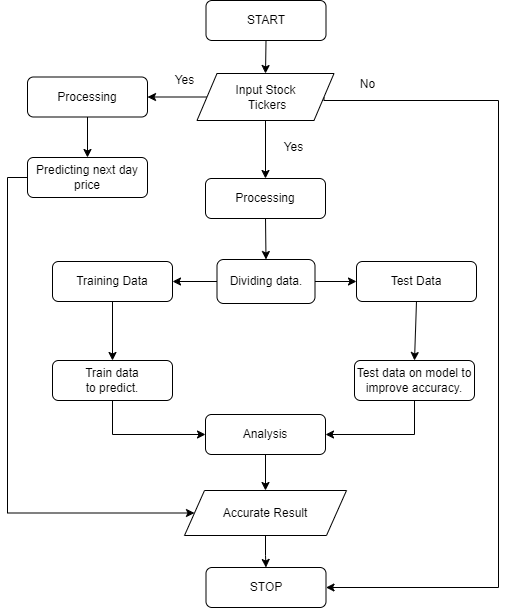
2-level DFD goes one step deeper into parts of 1-level DFD. It can be used to plan or record the specific/necessary detail about the system’s functioning.



**Figure 4: 2 - LEVEL DFD**

**3.7.2 Flowcharts**

A flowchart is a type of diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving a task. The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows.

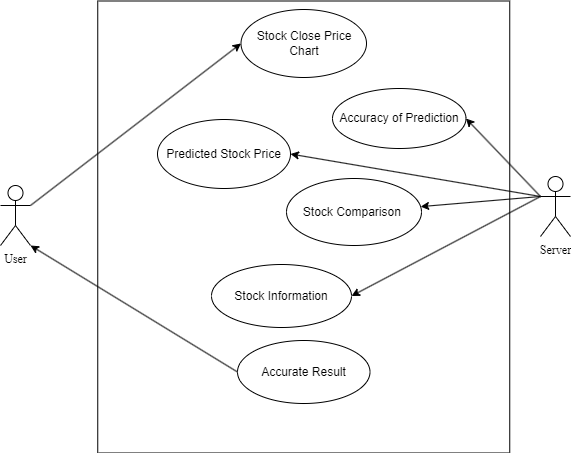


**Figure 5: Flowcharts**

**3.7.3 UML Diagrams**

1. **Use Case Diagram**

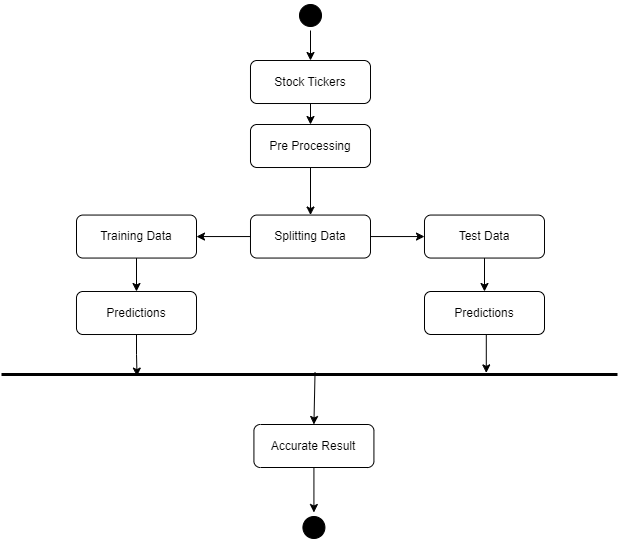
In the Unified Modelling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. To build one, you'll use a set of specialized symbols and connectors. An effective use case diagram can help your team discuss and represent:



**Figure 6: Use Case Diagram**

1. **Activity Diagram**

An activity diagram is a behavioral diagram i.e. it depicts the behavior of a system. An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths that exist while the activity is being executed.



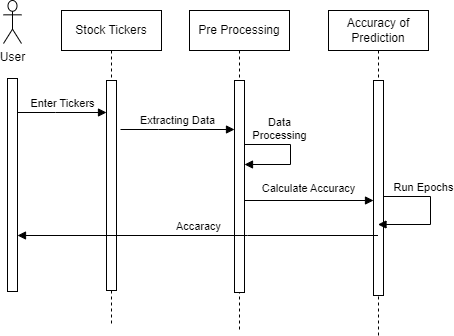
**Figure 7: Activity Diagram**

1. **Sequence Diagram**

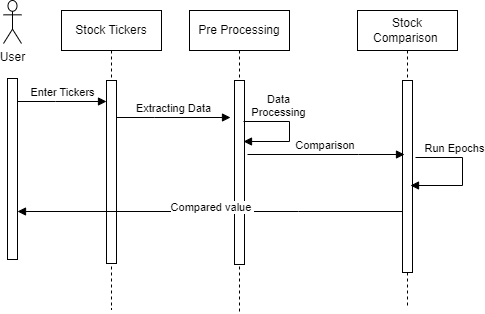
A sequence diagram is a type of interaction diagram because it describes how and in what order a group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process. Sequence diagrams are sometimes known as event diagrams or event scenarios.

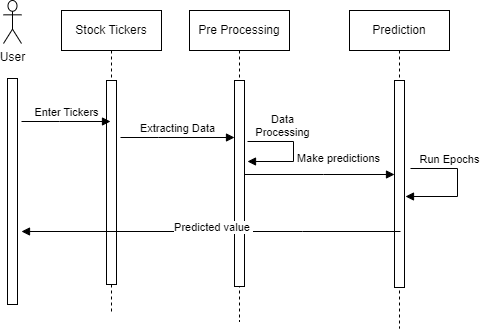
Sequence diagrams can be useful references for businesses and other organizations. Try drawing a sequence diagram to:

* Represent the details of a UML use case.
* Model the logic of a sophisticated procedure, function, or operation.
* See how objects and components interact with each other to complete a process.
* Plan and understand the detailed functionality of an existing or future scenario.



**Figure 8: Sequence Diagram for Accuracy Prediction**





**Figure 10: Sequence Diagram for Stock Comparison**

**Figure 9: Sequence Diagram for Stock Comparison**

1. **Pseudocodes**

**#Prediction**

if selected equals to "Prediction":

{

create a title "Stock Price Prediction"

take an input from the user as a stock ticker symbol

set start\_date and end\_date as per the desired range

create a dictionary named stock\_info

if ticker is not an empty string:

{

try to extract stock information using yfinance library

if successful, add stock information to the stock\_info dictionary

}

else:

{

write "Invalid symbol. Please enter a valid stock symbol."

}

if stock\_info is not empty:

{

display stock information and stock price data in tables

}

create a function named load\_data that takes a stock ticker as an argument and returns historical stock data

create a function named train\_model that takes stock data, splits it into train and test data, fits it to a linear regression model, and returns the model and test data

extract historical data using load\_data function

display a line chart of the historical stock price

extract model and test data using train\_model function

predict the stock price for the next day using the model

display the predicted stock price

}

**#Data Accuracy**

If selected equals "Data Accuracy"

{

then Display the title "Data Accuracy"

Prompt user to enter stock ticker and store in user\_input variable

If user\_input is an empty string then:

{

Display an empty string

}

Else:

{

Set start variable as today's date minus 20 years

Set end variable as today's date

Override yf module's pdr function

Download stock data for user\_input between start and end dates and store in df variable

Display the message "Closing Price History"

Draw a line chart of the 'Close' column of the df variable

Create a new dataframe named data with 'Close' column of the df variable

Convert the data dataframe to a numpy array and store it in dataset variable

Scale the dataset between 0 and 1 and store it in scaled\_data variable

Set training\_data\_len variable as 80% of the length of dataset rounded up

Create x\_train and y\_train arrays by iterating over the training\_data array and appending data for the past 60 days to x\_train and the data for the 61st day to y\_train

Convert x\_train and y\_train arrays to numpy arrays

Reshape the x\_train array to be 3-dimensional

Create an LSTM model with two LSTM layers, one dense layer with 25 units, and one output layer with 1 unit

Compile the model using adam optimizer and mean\_squared\_error loss function

Train the model using x\_train and y\_train with a batch size of 1 and 1 epoch

Create test\_data array with data from the remaining 20% of the dataset after the training data

Create x\_test array by iterating over the test\_data array and appending data for the past 60 days to x\_test

Convert x\_test array to numpy array

Reshape the x\_test array to be 3-dimensional

Make predictions using the model and x\_test, then invert the scaling to get the predicted stock prices and store them in predictions variable

Calculate the root mean squared error (RMSE) of the predictions compared to the actual stock prices

Display the RMSE value

Create a train dataframe with the data up to training\_data\_len

Create a valid dataframe with the data from training\_data\_len to the end of the dataset

Add a 'Predictions' column to the valid dataframe with the predicted stock prices

Display the message "Model"

Draw a line chart of the 'Close' column of the train dataframe

Draw a line chart of the 'Close' and 'Predictions' columns of the valid dataframe

}

}

**#Comparison**

IF selected equals to "Comparison":

{

Display "Stock Comparison" with title

Split the page into two columns using colu1 and colu2

In colu1:

Allow user to enter ticker symbol for the first stock as stock1

In colu2:

Allow user to enter ticker symbol for the second stock as stock2

Set the start\_date to today's date minus 20 years

Set the end\_date to today's date

IF stock1 equals empty string:

{

Display an empty string

}

ELSE IF stock2 equals empty string:

{

Display an empty string

}

ELSE:

{

Define a function named plot\_stock1 with parameter symbol

Download stock1's data for the given period using Yahoo Finance API

Create a subplots object fig1 and an ax object

Plot the Close prices on the ax object with date on x-axis and price on y-axis with label as symbol

Set x-label as "Date" and y-label as "Price"

Display the legend

Return fig1

Define a function named plot\_stock2 with parameter symbol

Download stock2's data for the given period using Yahoo Finance API

Create a subplots object fig2 and an ax object

Plot the Close prices on the ax object with date on x-axis and price on y-axis with label as symbol

Set x-label as "Date" and y-label as "Price"

Display the legend

Return fig2

Display "Comparing stock1 and stock2" where stock1 and stock2 are the entered ticker symbols in uppercase

Split the page into two columns using graph1 and graph2

In graph1:

Call the function plot\_stock1 with parameter stock1 to get fig1

Display the fig1 on the Streamlit page

Create a Ticker object for stock1 and assign it to stock1

IF stock1 is not empty string:

{

TRY :

Get the info for stock1 using Yahoo Finance API

Display the info in the format:

Name: <name>

Symbol: <symbol>

Exchange: <exchange>

Market Cap: ₹<market cap in billions>B if available, else "N/A"

PE Ratio: <trailing PE ratio> if available, else "N/A"

Forward PE Ratio: <forward PE ratio> if available, else "N/A"

Display "Price" with red color

Open Price: ₹<open price> if available, else "N/A"

Close Price: ₹<close price> if available, else "N/A"

High Price: ₹<high price> if available, else "N/A"

Low Price: ₹<low price> if available, else "N/A"

EXCEPT:

Display "Invalid symbol. Please enter a valid stock symbol."

}

In graph2:

Call the function plot\_stock2 with parameter stock2 to get fig2

Display the fig2 on the Streamlit page

Create a Ticker object for stock2 and assign it to stock2

IF stock2 is not empty string :

{

TRY:

Get the info for stock2 using Yahoo Finance API

Display the info in the format:

Name: <name>

Symbol: <symbol>

Exchange: <exchange>

Market Cap: ₹<market cap in billions>B if available, else "N/A"

PE Ratio: <trailing PE ratio> if available, else "N/A"

Forward PE Ratio: <forward PE ratio> if available, else "N/A"

Display "Price" with red color

Open Price: ₹<open price> if available, else "N/A"

Close Price: ₹<close price> if available, else "N/A"

High Price: ₹<high price> if available, else "N/A"

Low Price: ₹<low price> if available, else "N/A"

EXCEPT:

Display "Invalid symbol. Please enter a valid stock symbol."

}

**#Dashbord**

if selected is equal to "Dashboard":

{

then display title as "Stock Data Analysis Dashboard"

get user input for stock ticker symbol

if stock ticker symbol is empty then:

{

display nothing

}

Else:

{

set start date as 20 years ago from current date

set end date as current date

define function to load stock data using Yahoo Finance API

pass in stock ticker symbol as parameter

download data from start date to end date

reset index of data

return data

call the function and pass in stock ticker symbol as argument

store the returned data as "df"

define function to create a stock price chart

add a trace to the chart with x-axis as date and y-axis as closing price from "df"

set chart title as "Stock Prices" and enable x-axis range slider

return the chart

define function to create a daily return chart

add a trace to the chart with x-axis as date and y-axis as percentage change in closing price from "df"

set chart title as "Daily Returns" and enable x-axis range slider

return the chart

define function to create a moving average chart

add a trace to the chart with x-axis as date and y-axis as 50-day rolling mean of closing price from "df"

set chart title as "Moving Average" and enable x-axis range slider

return the chart

define function to create a volume chart

add a trace to the chart with x-axis as date and y-axis as volume from "df"

set chart title as "Volume" and enable x-axis range slider

return the chart

define function to create a candlestick chart

add a trace to the chart with x-axis as date and y-axis as open, high, low, and close prices from "df"

set chart title as "Candlestick Chart" and enable x-axis range slider

return the chart

define function to create the layout of the dashboard

display subheading "Stock Prices" and plot the stock price chart

divide the dashboard into two columns with equal width

in the first column, display subheading "Daily Returns" and plot the daily return chart

display subheading "Moving Average" and plot the moving average chart

in the second column, display subheading "Volume" and plot the volume chart

display subheading "Candlestick Chart" and plot the candlestick chart

end the layout function

call the layout function to display the dashboard

}

}

1. **Decision Table**

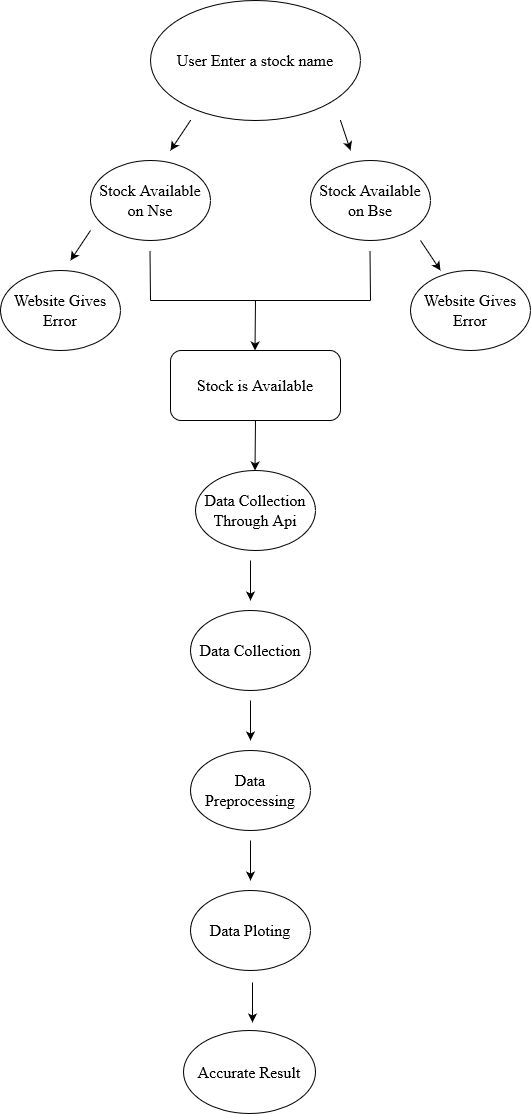
A decision table is a scheduled rule logic entry, in table format, that consists of conditions, represented in the row and column headings, and actions, represented as the intersection points of the conditional cases in the table. Decision tables are best suited for business rules that have multiple conditions. Adding another condition is done by simply adding another row or column.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Condition | Rule 1 | Rule 2 | Rule 3 | Rule 4 | Rule 5 |
| Check if user Enter Stock Tricker | T | T | T | T | F |
| Correct Stock Ticker | T | F | T | F | - |
| Stock Ticker should be available on NSE or BSE | T | T | F | F | - |
| OUTPUT | T | F | F | F | F |

**Figure 11: Decision Tables**

1. **Decision Tree**

Decision Tree is the most powerful and popular tool for classification and prediction. A Decision tree is a flowchart-like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label.



**Figure 12: Decision Tree**

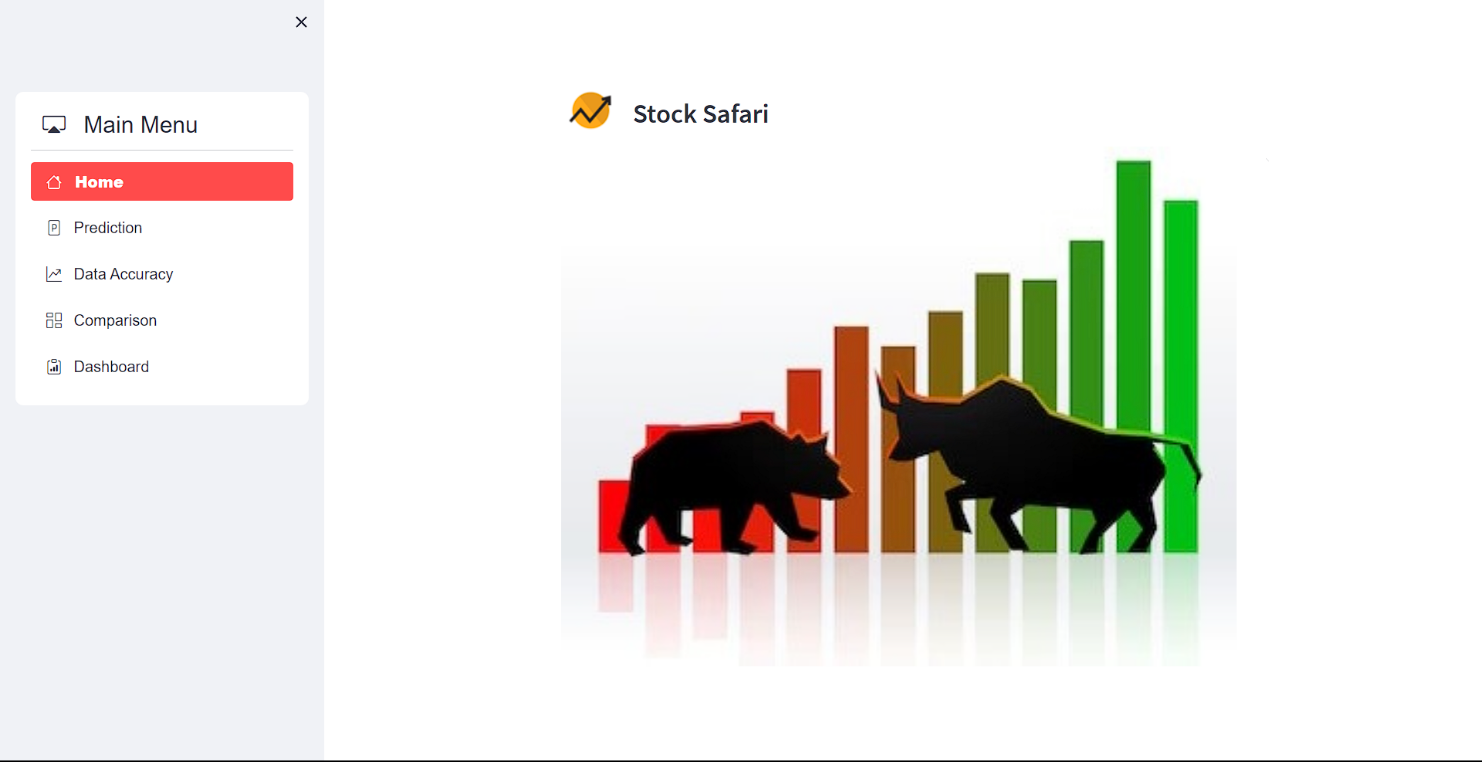
# CHAPTER 4

# SYSTEM DESIGN

**4.1 Basic Modules**

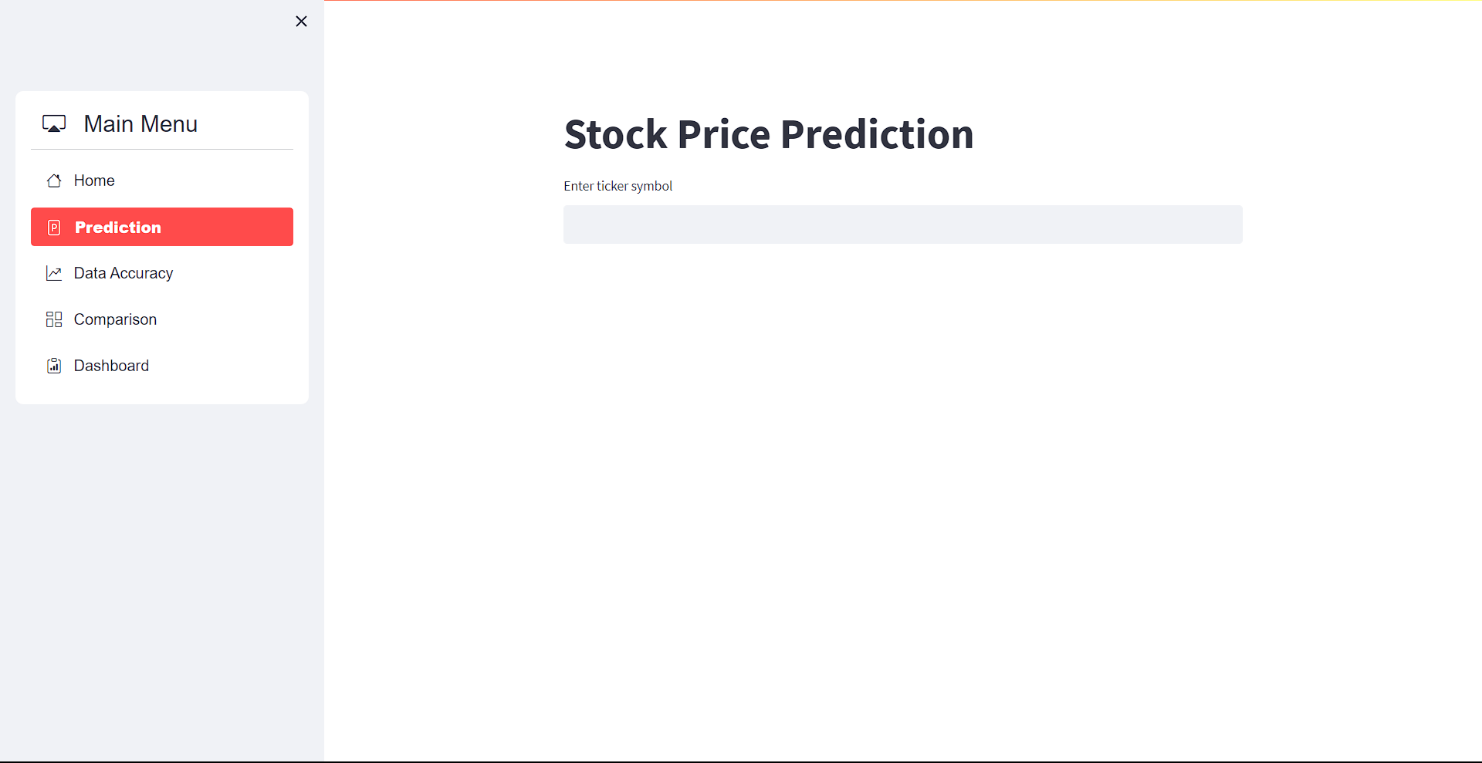
* + 1. **HOME PAGE**





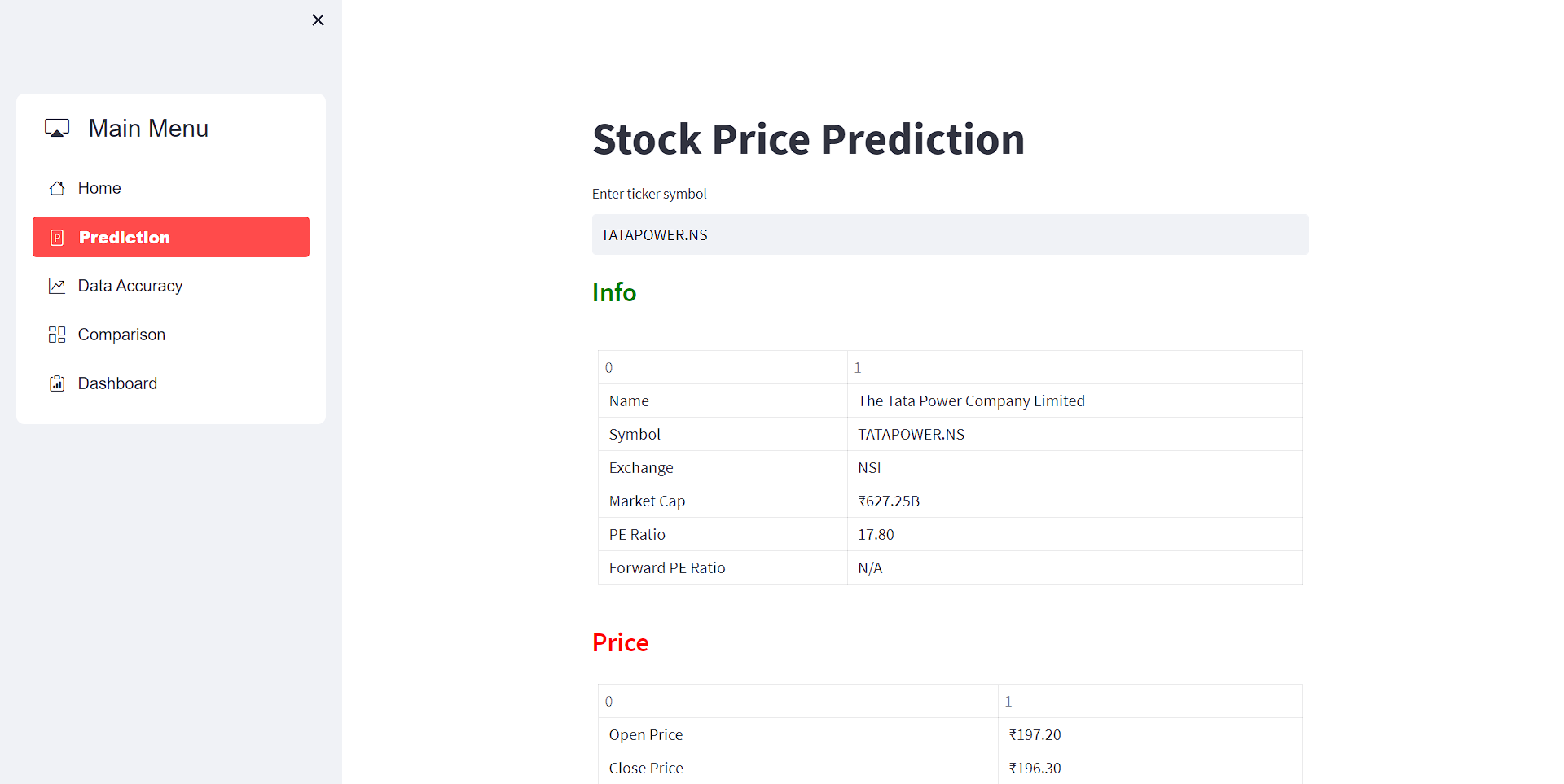
**Figure 13: Home Page**

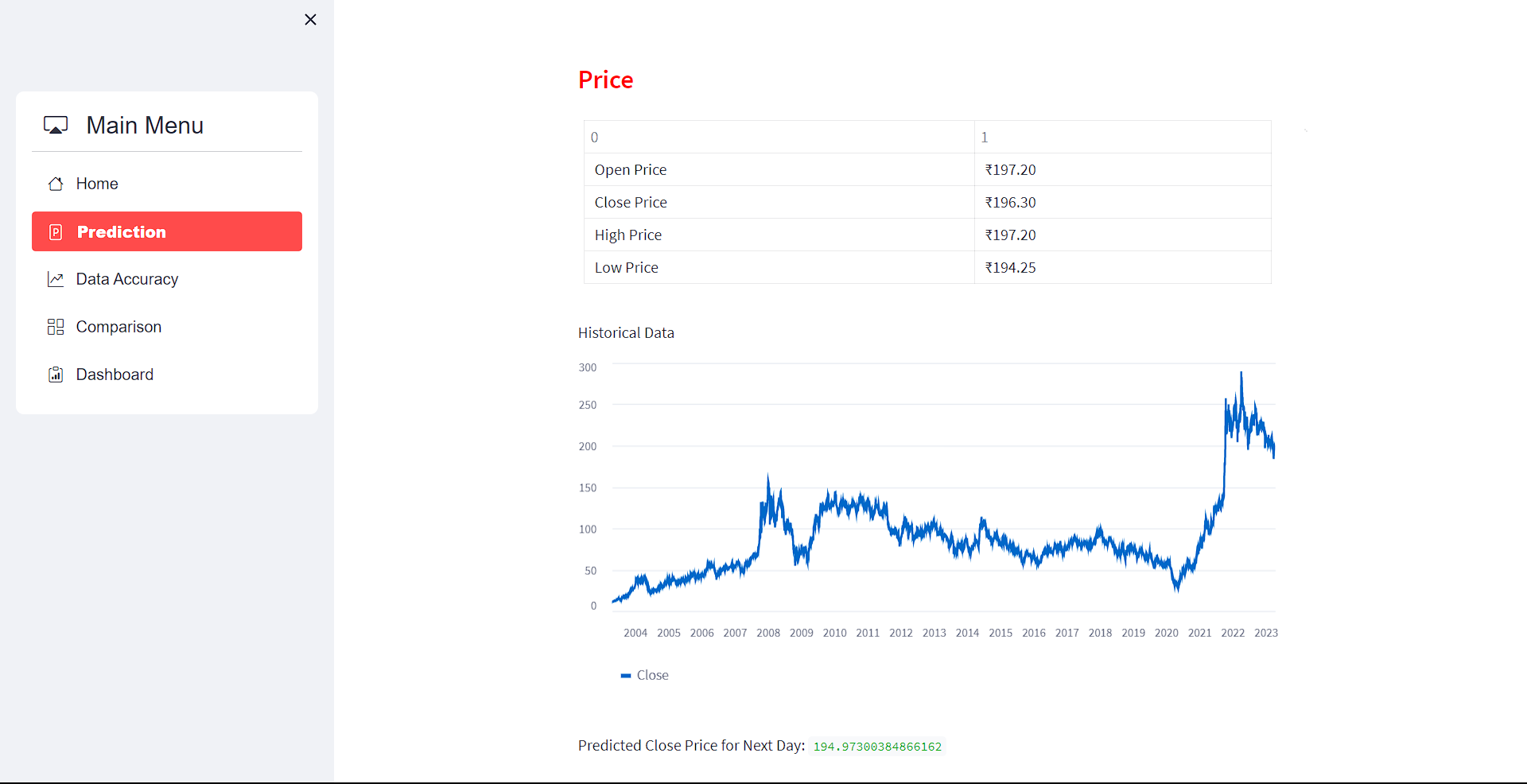
* + 1. **Input Prediction Page**



**Figure 14: Input Prediction Page**

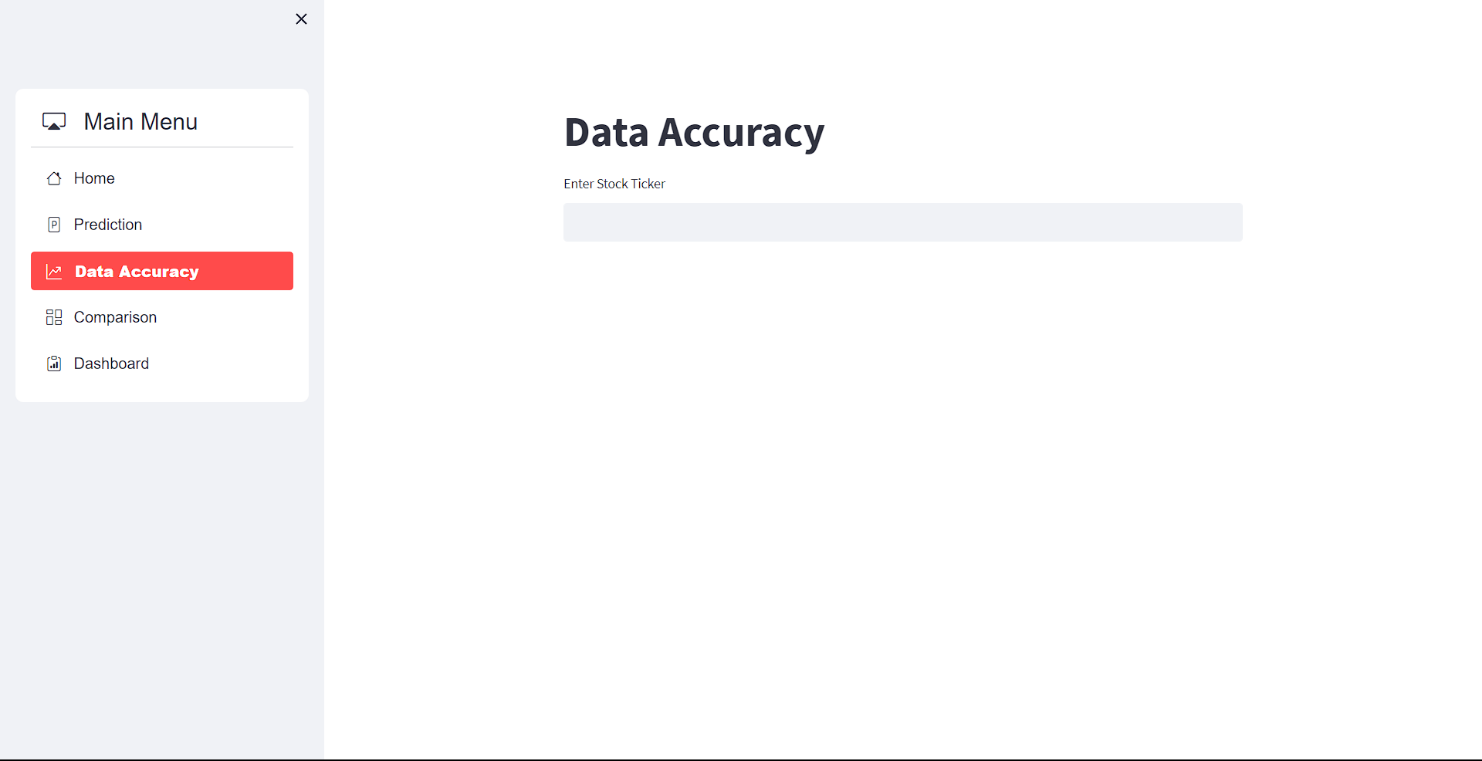
* + 1. **Output Prediction Page**



1. 

**Figure 15: Output Prediction Page**

* + 1. **Input Data Accuracy Page**



**Figure 16: Input Data Accuracy Page**

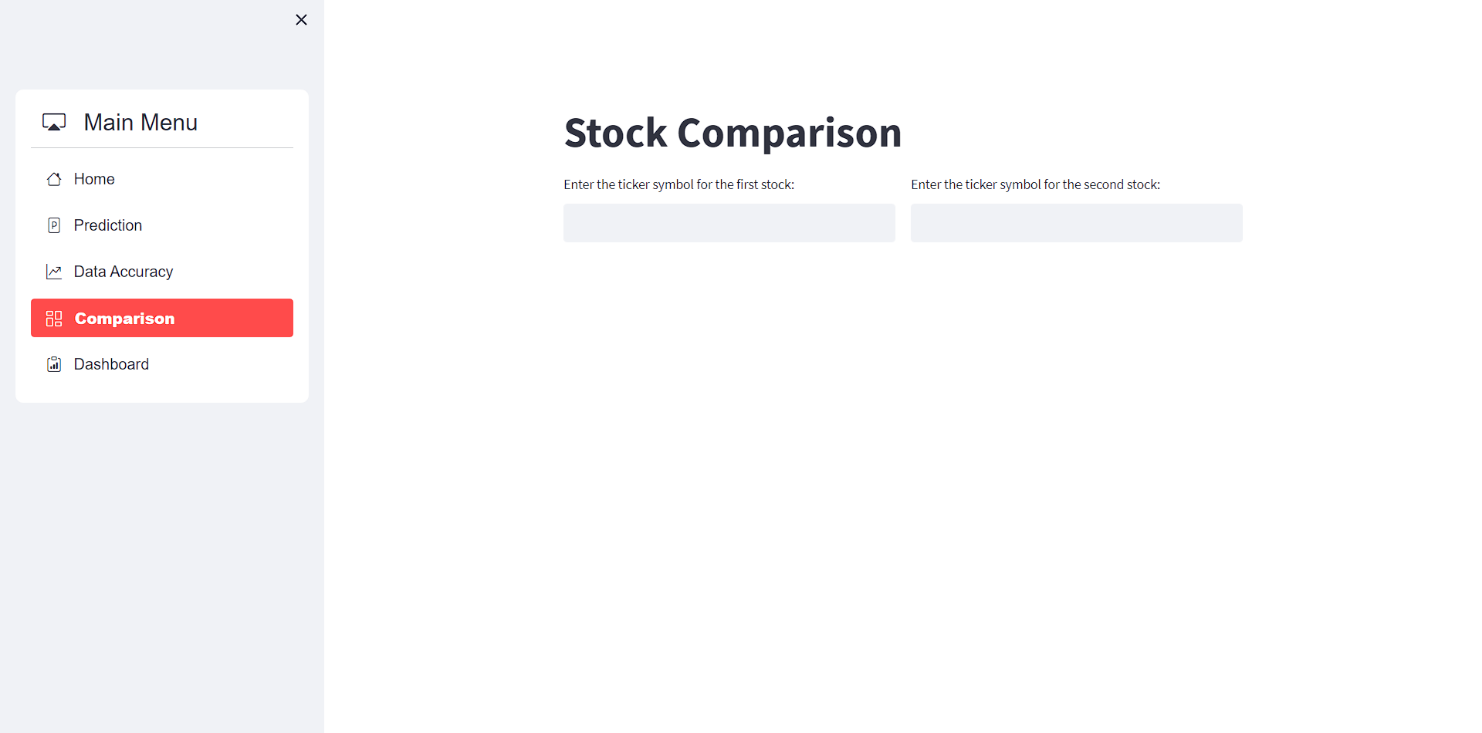
* + 1. **Output Data Accuracy Page**





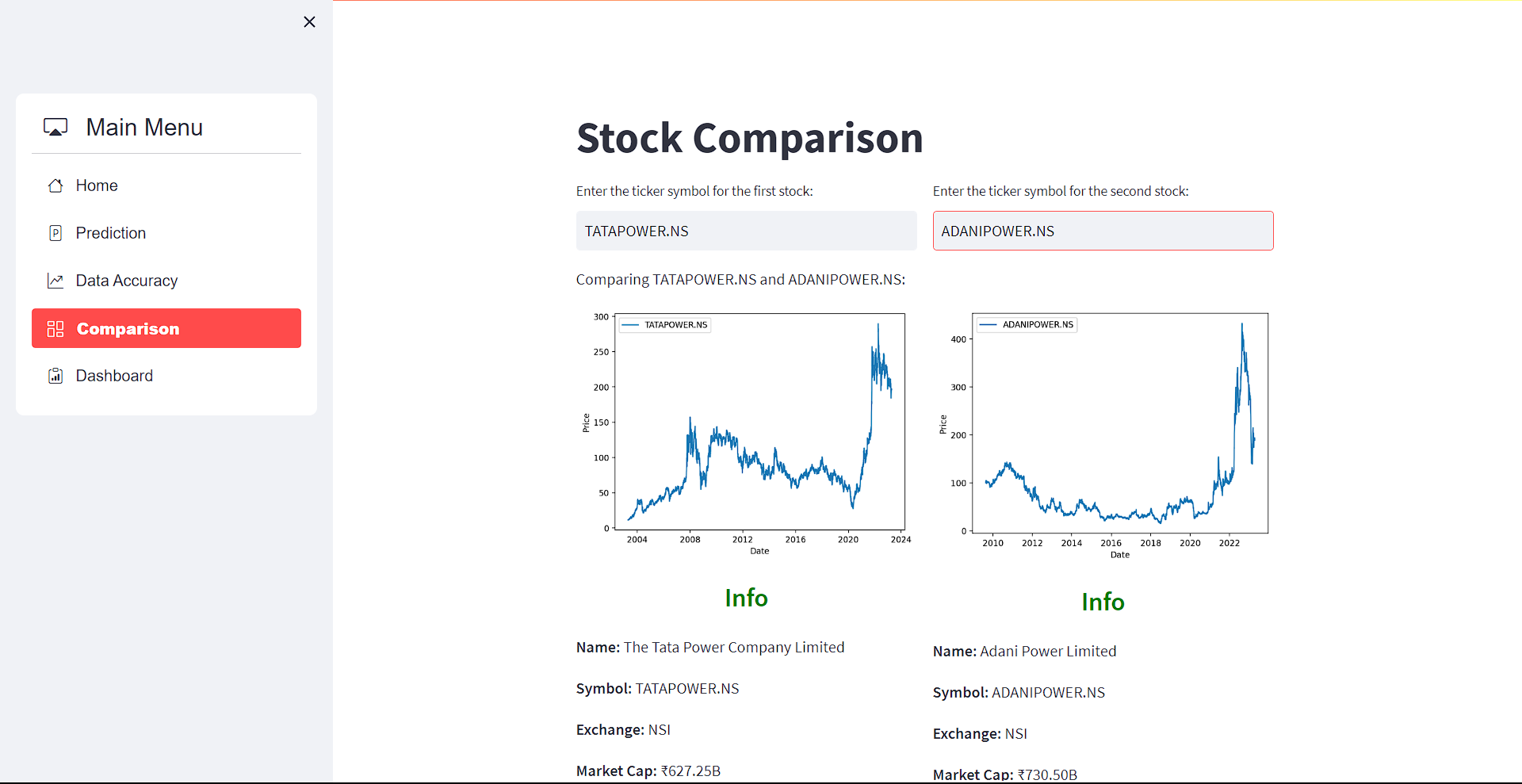
**Figure 17: Output Data Accuracy Page**

* + 1. **Input Stock Comparison Page**



**Figure 18: Input Stock Comparison Page**

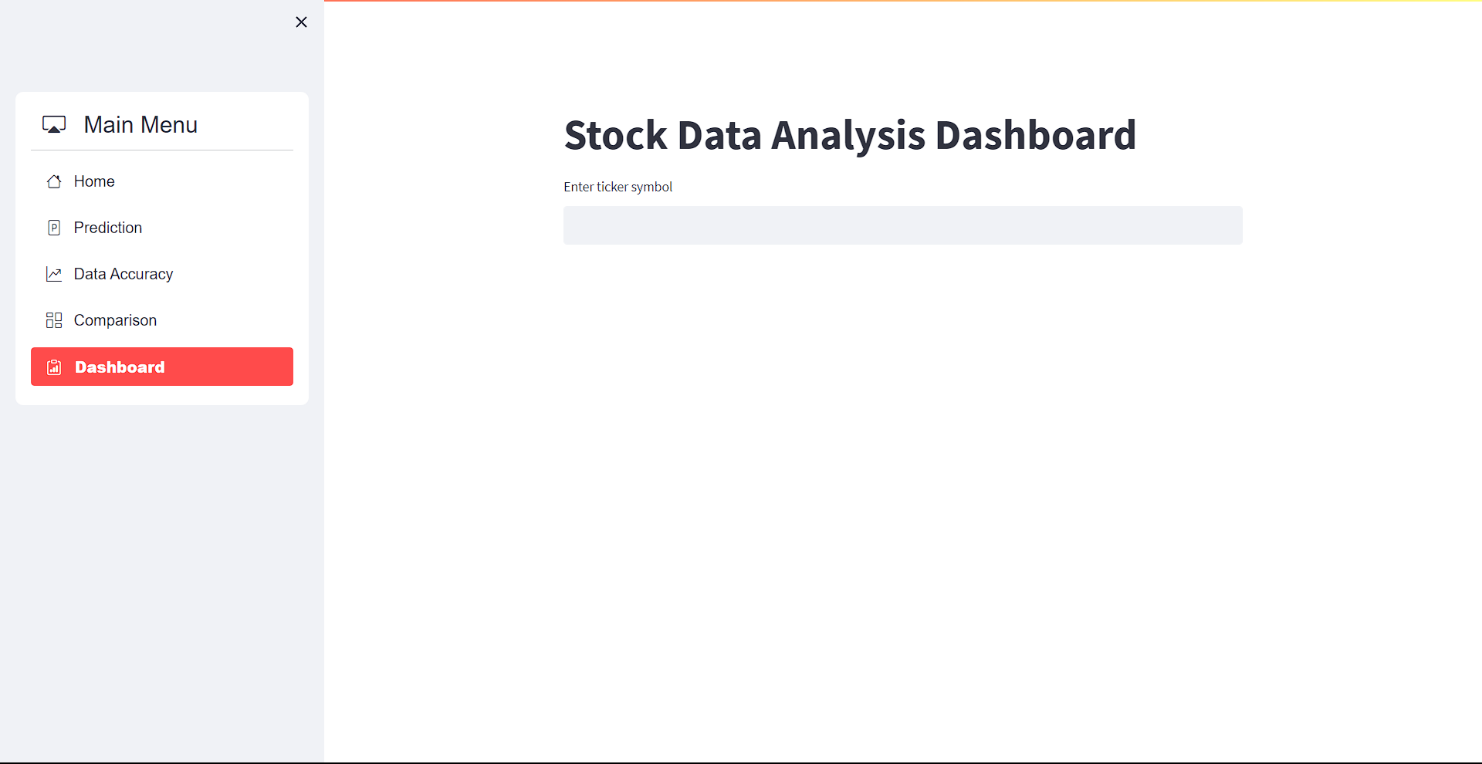
* + 1. **Output Stock Comparison Page**





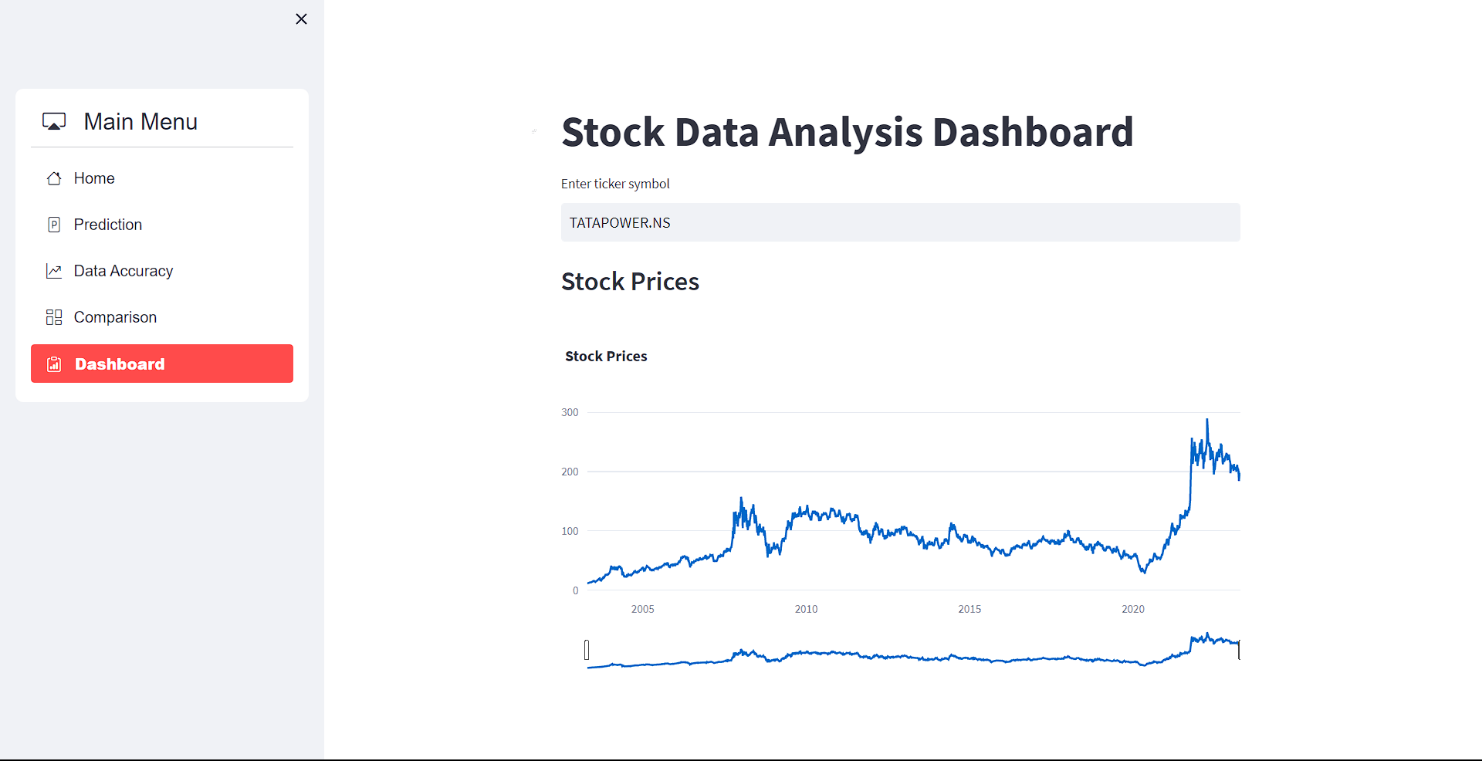
**Figure 19: Output Stock Comparison Page**

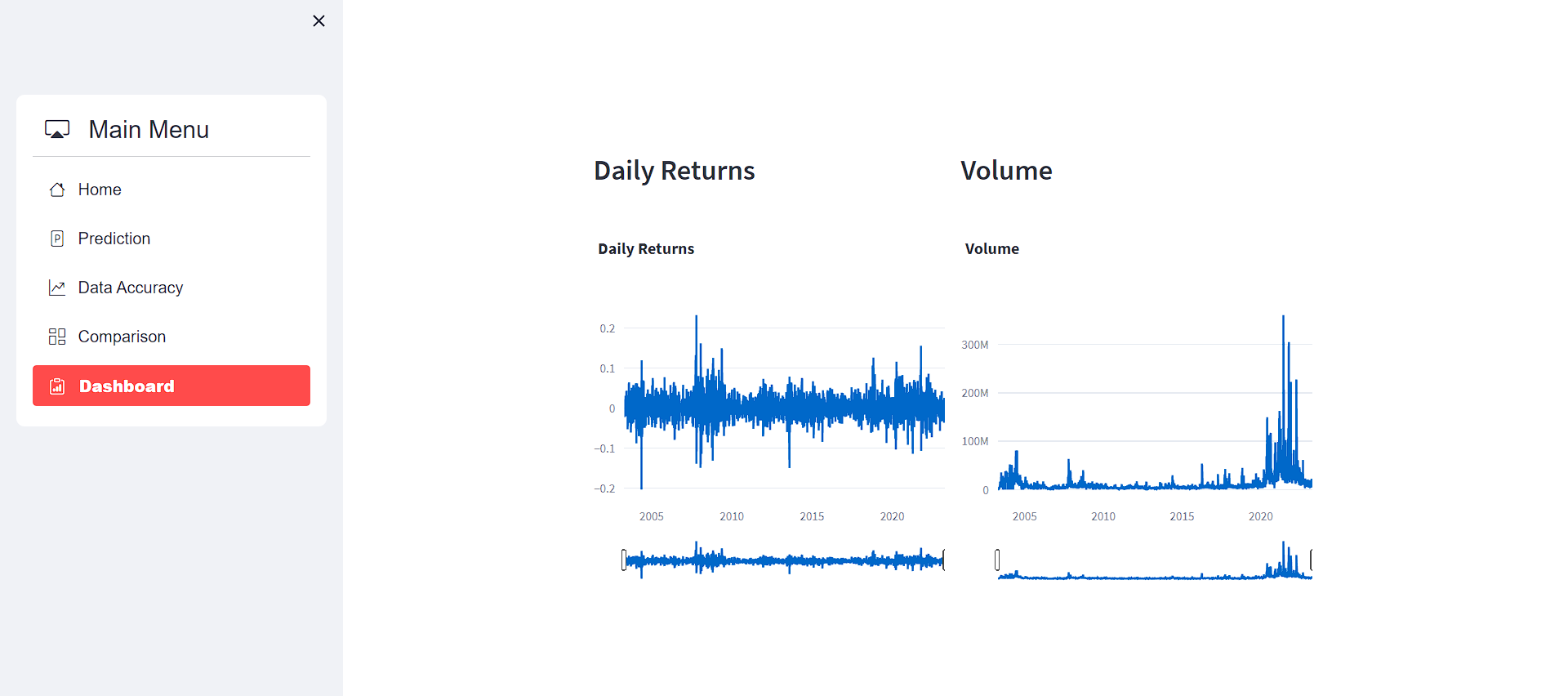
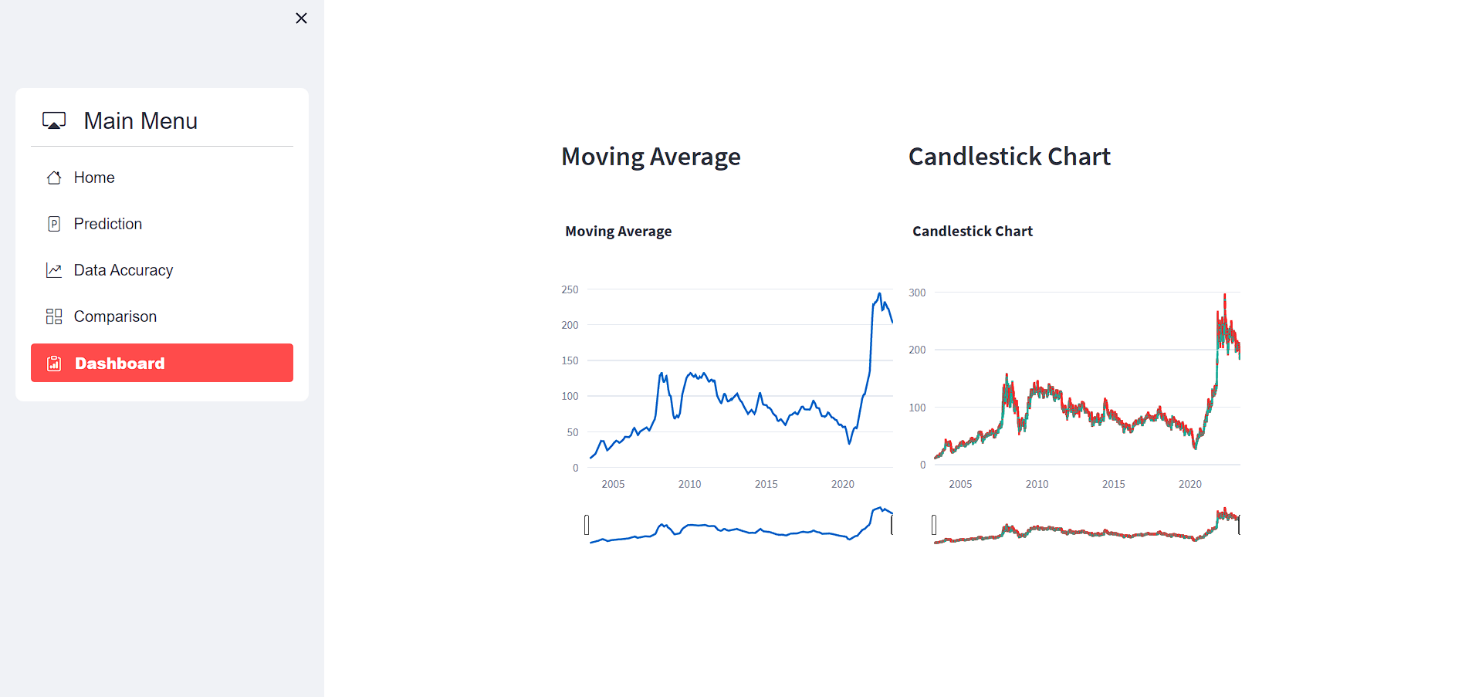
* + 1. **Input Dashboard Page**



**Figure 20: Input Dashboard Page**

* + 1. **Output Dashboard Page**





**Figure 21: Output Dashboard Page**

# CHAPTER 5

# IMPLEMENTATION AND TESTING

* 1. **Implementation Approaches**

A stock price prediction system can be implemented using a variety of techniques. Here are some common approaches:

1. Time-series analysis: This strategy involves evaluating historical stock prices and discovering patterns and trends in the data. To develop predictions based on historical data, methods including moving averages, exponential smoothing, and autoregressive integrated moving average (ARIMA) models can be utilised.

2. Machine learning: To anticipate future stock values, machine learning methods like decision trees, random forests, support vector machines (SVMs), and neural networks can be trained on past market data. To forecast future prices, the algorithms can be fed information such as volume, sentiment in the news, and previous prices.

3. Sentiment analysis: This strategy involves examining news reports, social media posts, and other information sources to determine how people feel about a specific stock. Future stock prices can be predicted using this knowledge.

4. Fundamental analysis: This strategy examines a company's financial records and other information to ascertain its true worth. Future stock prices can be predicted using this knowledge.

5. Hybrid strategy: To build a more reliable stock price prediction system, combine the aforementioned strategies. As an illustration, a system might combine machine learning and time-series analysis to create predictions.

* 1. **Coding Details and Code Efficiency**

**# Prediction Code**

if selected == "Prediction":

st.title('Stock Price Prediction')

ticker = st.text\_input('Enter ticker symbol')

start\_date = (datetime.today() - timedelta(days=365\*20)).strftime('%Y-%m-%d')

end\_date = datetime.today().strftime('%Y-%m-%d')

stock\_info = {}

if ticker != '':

try:

stock = yf.Ticker(ticker)

stock\_info = {

'Name': stock.info['longName'],

'Symbol': stock.info['symbol'],

'Exchange': stock.info['exchange'],

'Market Cap': '₹{:,.2f}B'.format(stock.info['marketCap']/1e9) if 'marketCap' in stock.info else 'N/A',

'PE Ratio': '{:.2f}'.format(stock.info['trailingPE']) if 'trailingPE' in stock.info else 'N/A',

'Forward PE Ratio': '{:.2f}'.format(stock.info['forwardPE']) if 'forwardPE' in stock.info else 'N/A',

'Open Price': '₹{:.2f}'.format(stock.info['regularMarketOpen']) if 'regularMarketOpen' in stock.info else 'N/A',

'Close Price': '₹{:.2f}'.format(stock.info['regularMarketPrice']) if 'regularMarketPrice' in stock.info else 'N/A',

'High Price': '₹{:.2f}'.format(stock.info['regularMarketDayHigh']) if 'regularMarketDayHigh' in stock.info else 'N/A',

'Low Price': '₹{:.2f}'.format(stock.info['regularMarketDayLow']) if 'regularMarketDayLow' in stock.info else 'N/A'

}

except:

st.write('Invalid symbol. Please enter a valid stock symbol.')

# Display the stock summary information in a table format

if stock\_info:

st.write('<p style="font-size:26px; color:green;"><b>Info<b></p>', unsafe\_allow\_html=True)

hide\_table\_row\_index = """

<style>

thead tr th:first-child {display:none}

tbody th {display:none}

</style>

"""

info\_data = pd.DataFrame([[key, stock\_info[key]] for key in ['Name', 'Symbol', 'Exchange', 'Market Cap', 'PE Ratio', 'Forward PE Ratio']])

st.markdown(hide\_table\_row\_index, unsafe\_allow\_html=True)

st.table(info\_data.style.hide\_index().set\_properties(\*\*{'text-align': 'left', 'padding-left': '10px'}))

st.write('<p style="font-size:26px; color:red;"><b>Price<b></p>', unsafe\_allow\_html=True)

price\_data = pd.DataFrame([[key, stock\_info[key]] for key in ['Open Price', 'Close Price', 'High Price', 'Low Price']])

st.table(price\_data.style.hide\_index().set\_properties(\*\*{'text-align': 'left', 'padding-left': '10px'}))

def load\_data(ticker):

data = yf.download(ticker, start\_date, end\_date)

return data

def train\_model(data):

X = data.drop(['Close'], axis=1)

y = data['Close']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0)

model = LinearRegression()

model.fit(X\_train, y\_train)

return model, X\_test, y\_test

if ticker:

data = load\_data(ticker)

st.write('Historical Data')

st.line\_chart(data['Close'])

model, X\_test, y\_test = train\_model(data)

tomorrow = data.iloc[-1].drop(['Close']).values.reshape(1, -1)

prediction = model.predict(tomorrow)

st.write('Predicted Close Price for Next Day:', prediction[0])

* 1. **Testing Approach**

A stock price prediction system testing strategy should include live testing with historical data testing. Testing against historical stock prices would determine whether the system's forecasts were accurate. Live testing compares the system's forecasts to current stock values in real-time. This can be accomplished by modelling trades based on the system's forecasts and reviewing the outcomes. It is crucial to regularly assess the system's performance and make the required modifications to raise its precision.

**5.3.1 Unit Testing**

Unit testing is a crucial procedure in software development that involves examining individual parts or chunks of code to make sure they operate properly when taken apart. Unit testing can assist make sure that the underlying code for a stock price prediction system that performs calculations and data processing is operating properly. This may entail testing certain data manipulation or computation capabilities as well as the system's capacity to handle various forms of data input and generate precise results. Unit tests can also be used to make sure that the system keeps working correctly over time, especially as new data is introduced to it or as the codebase is updated.

**5.3.2**  **Integrated Testing**

A key step in creating a stock price prediction system is integrated testing. It entails putting the complete system, including all parts and modules, through testing to make sure it functions properly and generates reliable results. While non-functional testing assures that the system is stable, dependable, and secure, functional testing guarantees that the system predicts stock values appropriately. Stress testing should be a part of the testing procedure to assess the system's performance under difficult circumstances. Overall, comprehensive testing must be carried out completely prior to deployment in order to guarantee the precision and dependability of a stock price prediction system.

# CHAPTER 6

# RESULTS AND DISCUSSION

* 1. **User Documentation**

Here is a user documentation for a stock price prediction system:

1. **Introduction**

* The stock price prediction system is a tool that helps users predict the future price of a particular stock.
* It uses historical data, technical indicators, and other factors to generate predictions.

1. **System Requirements**

* Internet connection
* Web browser (Chrome, Firefox, Safari, etc.)

1. **How to Use the System**

* **Prediction**
* Select the stock you want to predict and enter its ticker symbol.
* Click on the "Enter" button to generate the prediction results.
* Review the prediction results, including the predicted stock price and Graphs
* **Accuracy**
* Select the stock you want to predict and enter its ticker symbol.
* Click on the "Enter" button to generate the accuracy of the stock.
* Review the accuracy of the project in a graphical view.
* **Comparison**
* Select the stock you want to predict and enter its ticker symbols of two stocks.
* Click on the "Enter" button to generate the comparison results.
* Review the comparison results, including the graphs of two stocks.
* **Dashboard**
* Select the stock you want to predict and enter its ticker symbol.
* Click on the "Enter" button to generate the dashboard results.
* Review the dashboard results, including the graphical representation of stock.
  1. **Test Case**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Scenario** | **Prediction** | **Test Case Id** | **Prediction 1 A** |
| **Test Case Description** | **Positive case** | **Test Priority** | **High** |
|  |
| **Pre - Requisite** | **Valid Stock Ticker** | **Post-** **Requisite** | **NA** |  |

**Figure 22: Test Case**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr.No.** | **Action** | **Input** | **Expected Output** | **Actual Output** | **Result** |
| **1** | **Launch App** | **localhost** | **Home Page** | **Home Page** | **Pass** |
| **2** | **Stock Price Prediction** | **Correct Stock Ticker: TATAPOWER.NS** | **Prediction Success** | **Prediction Success** | **Pass** |
|  |
| **3** | **Stock Price Prediction** | **Input Number** | **Prediction Success** | **Error** | **Fail** |  |
|  |
| **4** | **Stock Price Prediction** | **Input Symbols** | **Prediction Success** | **Error** | **Fail** |  |
|  |
| **5** | **Stock Price Prediction** | **Incorrect Stock Ticker** | **Prediction Success** | **Error** | **Fail** |  |
|  |
| **6** | **Data Accuracy** | **Correct Stock Ticker: BHEL.NS** | **Data Accuracy Graph** | **Data Accuracy Graph** | **Pass** |  |
|  |
| **7** | **Data Accuracy** | **Input Number** | **Data Accuracy Graph** | **Error** | **Fail** |  |
|  |
| **8** | **Data Accuracy** | **Input Symbols** | **Data Accuracy Graph** | **Error** | **Fail** |  |
|  |
| **9** | **Data Accuracy** | **Incorrect Stock Ticker** | **Data Accuracy Graph** | **Error** | **Fail** |  |
|  |
| **10** | **Stock Comparison** | **Correct 2 Stock Ticker: TATAPOWER.NS & ADANIPOWER.NS** | **Compare Success** | **Compare Success** | **Pass** |  |
|  |
|  |
| **11** | **Stock Comparison** | **Input Number** | **Compare Success** | **Error** | **Fail** |  |
|  |
| **12** | **Stock Comparison** | **Input Symbols** | **Compare Success** | **Error** | **Fail** |  |
|  |
| **13** | **Stock Comparison** | **Incorrect Stock Ticker** | **Compare Success** | **Error** | **Fail** |  |
|  |
| **14** | **Dashboard** | **Correct Stock Ticker** | **Dashboard Page** | **Dashboard Page** | **Pass** |  |
|  |
| **15** | **Dashboard** | **Input Number** | **Dashboard Page** | **Error** | **Fail** |  |
|  |
| **16** | **Dashboard** | **Input Symbols** | **Dashboard Page** | **Error** | **Fail** |  |
|  |
| **17** | **Dashboard** | **Incorrect Stock Ticker** | **Dashboard Page** | **Error** | **Fail** |  |
|  |

# CHAPTER 7

# CONCLUSIONS

* 1. **Conclusion**

Although there are several drawbacks that must be considered, linear regression can be a useful method for estimating stock values. On the other hand, the simple and straightforward technique of linear regression can provide crucial information about the connections between numerous variables and stock prices. If investors can identify the variables that correlate most strongly with price fluctuations, they will be better equipped to make decisions about when to purchase or sell equities.

However, using linear regression to forecast stock values has a number of important disadvantages. The first is that there is a linear relationship between the stock price and the variables being considered. This could not always be the case, especially when the economy is uncertain or when the markets are very volatile. Second, linear regression analyses historical data to predict the future. While past trends may be useful indicators, they may not necessarily be accurate price predictions. External factors like changes in governmental policy, unforeseen economic developments, or technological advances can have a big impact on stock values. Last but not least, it is critical to keep in mind that predicting stock prices is a challenging process with many facets, and no one method or model can guarantee

* 1. **Limitations of the System**

Market volatility: The stock market is highly unpredictable because of the many factors that could influence stock values. Among these factors are political turmoil, natural disasters, global events, and shifting economic situations. Consequently, predicting stock values can be quite challenging.

Limited historical data: Stock price forecasts occasionally rely on historical information, which isn't always a reliable indicator of future performance. Additionally, the quality of historical data may be impacted by the accuracy of the data source and the data collection process.

Insider trading: Insider trading can have a significant impact on stock values and is difficult to predict because it involves illegal behaviours that are typically kept a secret.

Psychological factors: Stock prices are influenced by both investor emotions and market psychology. It can be difficult to predict these traits because they are usually influenced by emotions and other intangible variables.

* 1. **Future Scope of the Project**

1. We would like to develop this application further to forecast bitcoin trade.
2. To improve analysis, sentiment analysis will be added.

**CHAPTER 8**

**REFERENCE**

1. **"Stock price prediction using machine learning and deep learning techniques:** A review" by Vinayakumar R, et al. (2019). This paper provides an overview of various machine learning and deep learning techniques that have been used for stock price prediction and compares their performance.
2. **"Stock price prediction using regression analysis" by Li, et al. (2019).** This paper describes how regression analysis can be used to predict stock prices and compares the performance of various regression models.
3. **"Recurrent neural networks for stock price prediction:** A systematic literature review" by Chong, et al. (2019). This paper reviews the literature on using recurrent neural networks (RNNs) for stock price prediction and discusses the advantages and limitations of RNNs.
4. **"Stock price prediction using news articles and sentiment analysis"** by Wu, et al. (2019). This paper describes how news articles and sentiment analysis can be used to predict stock prices and compares the performance of various sentiment analysis techniques.
5. **"Stock price prediction using machine learning and sentiment analysis"** by Zheng, et al. (2020). This paper describes a system that combines machine learning and sentiment analysis to predict stock prices and compares its performance with traditional prediction methods.