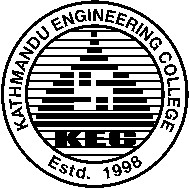
**TRIBHUVAN UNIVERSITY**

**INSTITUTE OF ENGINEERING**

Kathmandu Engineering College

Department of Computer Engineering



Mid-Term Project Report

On

**Stock Market Prediction**

[Code No: CT 654]

By

Abhishek Man Shrestha - 078BCT009

Ayush Acharya - 078BCT022

Bishranta Regmi - 078BCT027

Dhiraj Sah - 078BCT030

Kathmandu, Nepal

2081

# ACKNOWLEDGEMENT

First of all, we would like to acknowledge our deep sense of gratitude to Institute of Engineering for including the minor project in our course. Similarly, we are very thankful to the Department of Computer Engineering, KEC for guiding us throughout this project.

We are also grateful to **Er. Krista Byanju**, our project coordinator for her recommendations regarding the development of our project. Also, we extend a special thanks to **Er. Sudeep Shakya**, Head of Computer Department, **Er. Kunjan Amatya**, Deputy Head of Computer Department as well as **Er. Nabin Neupane**, our year coordinator for all the help support and supervision provided to us so far. We can’t thank the teachers of KEC enough for providing us with the invaluable supervision and necessary information for the successful completion of the project.

We would also like to thank all our friends for the assistance they have provided during this report. We hope all their help and support can result in a project that all of us can be proud of which can help us inherit different set of skills in the field of programming. Finally, we would like to give our sincere gratitude towards anyone who helped us directly or indirectly to finalize this report.

# ABSTRACT

This project focuses on developing a stock market prediction system that integrates historical data analysis and sentiment analysis. We have developed a model for the prediction of stock prices using historical data collected from reliable sources (NEPSE and NepseAlpha). For this, we utilized the Long Short-Term Memory (LSTM) model, a type of recurrent neural network specifically effective in analyzing sequential data. The collected historical data was preprocessed, normalized and divided into an 80% training set and 20% test set. The LSTM model is trained on this dataset, and the output is presented in both tabular and graphical formats. We have created a web application using Flask to visualize these predictions interactively. The next phase of the project involves integrating sentiment analysis into the prediction system. Sentiment data from news articles and social media will be extracted using natural language processing (NLP) techniques. We are exploring algorithms like Convolutional Neural Networks (CNNs) and libraries like Natural Language Toolkit (NLTK) for this purpose. Sentiment scores will be calculated and fused with historical data to refine and enhance the predictive accuracy of the model. The project will culminate in a user-friendly web application where users can view predictions and insights seamlessly. By continuously evaluating and refining the models, we aim to create a robust and efficient system for stock market forecasting.

Keywords: *stock, historical dataset, sentiment, analysis*

# TABLE OF CONTENTS

[ACKNOWLEDGEMENT i](#_Toc184034395)

[ABSTRACT i](#_Toc184034396)

[TABLE OF CONTENTS i](#_Toc184034397)

[LIST OF FIGURES i](#_Toc184034398)

[LIST OF ABBREVIATIONS i](#_Toc184034399)

[CHAPTER 1: INTRODUCTION 1](#_Toc184034400)

[1.1 BACKGROUND THEORY 1](#_Toc184034401)

[1.2 PROBLEM STATEMENT 1](#_Toc184034402)

[1.3 OBJECTIVES 1](#_Toc184034403)

[1.4 SCOPE AND APPLICATIONS 1](#_Toc184034404)

[CHAPTER 2: LITERATURE REVIEW 1](#_Toc184034405)

[2.1 EXISTING SYSTEMS 1](#_Toc184034406)

[2.2 LIMITATIONS OF PREVIOUS SYSTEMS 1](#_Toc184034407)

[2.3 SOLUTIONS PROVIDED BY OUR SYSTEM 1](#_Toc184034408)

[CHAPTER 3: METHODOLOGY 1](#_Toc184034409)

[3.1 PROCESS MODEL 1](#_Toc184034410)

[3.1.1 ITERATIVE MODEL 1](#_Toc184034411)

[3.2 BLOCK DIAGRAM 1](#_Toc184034412)

[3.2.1 Data Collection 1](#_Toc184034413)

[3.2.2 Data Preprocessing 1](#_Toc184034414)

[3.2.3 Feature Extraction 1](#_Toc184034415)

[3.2.4 Model Training 1](#_Toc184034416)

[3.2.5 Model Integration: 1](#_Toc184034417)

[3.2.6. Model Evaluation 1](#_Toc184034418)

[3.2.7 Prediction 1](#_Toc184034419)

[3.2.8 Visualization 1](#_Toc184034420)

[3.3 ALGORITHMS 1](#_Toc184034421)

[3.3.1 Recurrent Neural Network (RNNs) 1](#_Toc184034422)

[3.3.2 Long Short-Term Memory (LSTM) 1](#_Toc184034423)

[3.3.3 Convolutional Neural Networks (CNNs) 1](#_Toc184034424)

[3.4 FLOWCHART 1](#_Toc184034425)

[3.6 Necessary Diagrams 1](#_Toc184034426)

[3.6.1 Data Flow Diagram 1](#_Toc184034427)

[3.6.2 Use Case Diagram 1](#_Toc184034428)

[3.6.3 Sequence Diagram 1](#_Toc184034429)

[3.7 TOOLS USED 1](#_Toc184034430)

[3.7.1 Python 1](#_Toc184034431)

[3.7.2 HTML/CSS/JavaScript 1](#_Toc184034432)

[3.7.3 GitHub 1](#_Toc184034433)

[CHAPTER 4: EPILOGUE 1](#_Toc184034434)

[4.1 TASK COMPLETED 1](#_Toc184034435)

[4.2 TASK REMAINING 1](#_Toc184034436)

[4.2 GANTT CHART 1](#_Toc184034437)

[REFERENCES 1](#_Toc184034438)

[BIBLIOGRAPHY 1](#_Toc184034439)

[SCREENSHOTS 1](#_Toc184034440)

# LIST OF FIGURES

Figure 1: Iterative Process Model 7

Figure 2: System Block Diagram 9

Figure 3: Recurrent Neural Network 11

Figure 4: Convolution Neural Network 12

Figure 5: Long Short-Term Memory 13

Figure 6: Flowchart 14

Figure 7: Data Flow Diagram Level 0 17

Figure 8: Data Flow Diagram Level 1 17

Figure 9: Data Flow Diagram Level 2 18

Figure 10: Use Case Diagram 19

Figure 11: Sequence Diagram 20

Figure 12: Gantt Chart 23

# LIST OF ABBREVIATIONS

**ANN** Artificial Neural Network

**BGRU** Bidirectional Gated Recurrent Unit

**CNN** Convolution Neural Network

**CSS** Cascading Style Sheets

**GRU** Gated Recurrent Unit

**HTML** HyperText Markup Language

**HTTP** HyperText Transfer Protocol

**LSTM** Long Short-Term Memory

**RNN** Recurrent Neural Network

**SVM** Support Vector Machine

# CHAPTER 1: INTRODUCTION

## 1.1 BACKGROUND THEORY

The stock market is a system where people can buy and sell shares of publicly traded companies. Companies issue shares of stock to the public to raise funds for their operations and growth. The stock market operates on the principle of supply and demand. When demand for a company's stock is high, its share price rises, and vice versa. Investors purchase these shares because they believe the company will succeed and grow over time, increasing the value of their investment. If the company performs well, the stock price usually goes up, allowing investors to sell their shares at a higher price and make a profit.

The Nepal Stock Exchange (NEPSE) is the main stock exchange of Nepal. NEPSE opened its trading floor on 13 January 1994. It was established to provide a marketplace for the trading of securities in Nepal. The exchange facilitates transactions for both individual and institutional investors and plays a vital role in the economic development of Nepal by enabling companies to access public funds for expansion and growth.

Stock market prediction involves forecasting the future value of a company's stock, which is crucial for economic growth. By studying patterns in market data, we can predict when to buy stocks and potentially earn profits. This prediction relies on understanding supply and demand. By analyzing large amounts of market data, we can find patterns that give us a good chance of making accurate predictions. Stock market prediction consists of statistical analysis, machine learning, and sentiment analysis to forecast future stock prices using historical data and market sentiment. Integrating these methods and using machine learning helps to develop predictive models offering valuable insights of the market.

## 1.2 PROBLEM STATEMENT

The stock market is vast and unpredictable, with significant fluctuations making it challenging to anticipate whether prices will rise or fall the next day. Investing in stocks can result substantial profits or result in significant losses, making it a scary task for many. Manual methods of predicting future market trends are tedious and often inaccurate, especially when the financial markets are influenced by factors like politics, economics, and investor psychology.

## 1.3 OBJECTIVES

This project is created with the purpose to fulfill the following objectives:

* To analyze trends and patterns of historical trading data (date, closing price, etc.) and sentimental data (news headlines, articles, posts, etc.) to understand market trends.
* To predict future price using the information from analyzed data implementing machine learning algorithms LSTM and CNN.

## 1.4 SCOPE AND APPLICATIONS

The applications of this project are as follows:

* To bridge communication between financial investors and industries as investors can find right companies for investments and companies can find investors for their growth
* To assist the investors and people to make sensible decisions for profitable investments through analysis of historical trading data.

# CHAPTER 2: LITERATURE REVIEW

A study done by Polepally, Vijay & Jakka, Neha & Vishnukanth, Pendly & Raj, Rachakatla & Anish, Gudavelli presents a comprehensive study of machine learning algorithms for stock price prediction. In their 2023 study, they explored the effectiveness of the LSTM-RNN algorithm for predicting stock prices. Presented at the 7th International Conference on Intelligent Computing and Control Systems (ICICCS), their research highlights how LSTM-RNN can capture long-term dependencies in sequential data, making it highly suitable for stock market prediction. This algorithm is adept at handling time series data, which is critical for accurate forecasting in financial markets. However, their approach primarily focuses on historical data without incorporating sentiment analysis, potentially limiting its predictive capabilities in response to market sentiment shifts [1].

Lal and Timalsina's 2022 paper, presented at the 11th IOE Graduate Conference, investigated the use of a CNN-BGRU method for stock price prediction. The combination of Convolutional Neural Networks (CNN) for feature extraction and Bidirectional Gated Recurrent Units (BGRU) for handling sequential data shows promise in forecasting stock prices. Their method leverages CNN’s strength in identifying important patterns in data and BGRU’s ability to process time-dependent sequences effectively. Despite its potential, this method can be complex and resource-intensive, and it does not explicitly include sentiment analysis, which could enhance prediction accuracy [2].

In the 2023 publication in the International Journal of Creative Research Thoughts (IJCRT), Kabir, Sobur, and Amin compared various machine learning models for stock price prediction. Their study evaluates the performance of different algorithms under diverse market conditions, providing insights into their strengths and weaknesses. While their comprehensive approach offers valuable comparisons, it does not integrate additional data sources such as sentiment analysis from news or social media, which can be crucial for improving the accuracy of stock predictions [3].

Choi’s 2018 research discusses a hybrid model combining ARIMA and LSTM for predicting stock price correlations. The hybrid ARIMA-LSTM model leverages the statistical strengths of ARIMA and the deep learning capabilities of LSTM, resulting in improved accuracy for correlation predictions between stocks. This approach demonstrates the benefits of combining traditional statistical methods with modern machine learning techniques. However, the model's complexity may pose challenges for practical implementation, and it focuses on correlation prediction rather than direct price forecasting [4].

Patel, J., Shah, S., Thakkar, P., & Kotecha, K. in their 2015 paper published in Expert Systems with Applications, Patel and colleagues examined various machine learning techniques, including ANN, SVM, random forest, and naive Bayes, for predicting stock prices and index movements. Their study emphasizes the importance of trend deterministic data preparation to enhance the accuracy of machine learning models. By comparing these techniques, the authors provide insights into their effectiveness for financial forecasting. Despite the comprehensive analysis, the study does not incorporate sentiment analysis, which could provide a more holistic view of market dynamics [5].

## 2.1 EXISTING SYSTEMS

**2.1.1 QuantConnect**

An algorithmic trading platform that allows users to create and back test trading strategies which requires a strong understanding of programming and algorithmic trading concepts, making it less accessible to beginners.

**2.1.2 MetaTrader5**

A popular trading platform offering tools for technical analysis and automated trading strategies that primarily focused on technical analysis and doesn't inherently support machine learning models or sentiment analysis.

**2.1.3 AlphaSense**

An AI-powered market intelligence platform that aggregates and analyzes financial data, news, and research reports but doesn't provide specific stock price prediction capabilities.

## 2.2 LIMITATIONS OF PREVIOUS SYSTEMS

* Many platforms like QuantConnect require advanced programming knowledge and have steep learning curves, making them less accessible to beginner users.
* Advanced platforms like AlphaSense have high subscription fees, which can be prohibitive for individual investors or small teams.
* Most tools focus primarily on technical analysis and do not natively support the integration of sentiment analysis from news and social media, limiting the comprehensiveness of predictions.

## 2.3 SOLUTIONS PROVIDED BY OUR SYSTEM

* Use of effective model: This project uses the LSTM algorithm, which excels in stock price forecasting due to its ability to handle time series data and avoid memory issues which is a major problem with RNN.
* Integration of Sentiment Analysis: This system incorporates sentiment analysis from news and social media to further enhance prediction reliability using CNN algorithm. By combining LSTM with CNN, our system aims to deliver superior accuracy in predicting stock values.
* Accessibility: Using widely known programming languages, frameworks, open-source libraries and tools keeps costs low, makes the solution more accessible to broader audience, individual users and small teams.

# CHAPTER 3: METHODOLOGY

## 3.1 PROCESS MODEL

### 3.1.1 ITERATIVE MODEL

*Figure 1: Iterative Process Model*

In the iterative model, the process starts with a simple implementation of the software requirements and iteratively enhances the versions until the complete system is implemented and ready to be deployed.

The various phases of iterative model are as follows:

**1. Requirement analysis:** In the first phase, the identification of the key requirements for the stock prediction system i.e., historical data analysis, sentiment analysis, and the integration of both is done. To develop the software under the incremental model, this phase performs a crucial role.

**2. Design:** In this phase, design of the features to be used in the model, such as moving averages from historical data and sentiment scores from text data are done along with simple LSTM model for time-series prediction using historical data and a basic sentiment analysis module to process textual data.

**3. Implementation:** Implementation phase enables the coding phase of the development system. It involves the final coding that implements a basic LSTM model for predicting stock prices based on historical data and integrate a sentiment analysis module using libraries TextBlob or NLTK.

**4. Testing:** The testing phase checks the performance of each existing function, tests the LSTM model's accuracy in predicting stock prices and verifies that the sentiment scores are correctly influencing the predictions.

**5. Review:** After testing the model's performance, identifying areas for improvement, such as enhancing sentiment analysis or refining the LSTM model is done. Planning to integrate more advanced techniques or additional data sources in the next iteration is done if necessary.

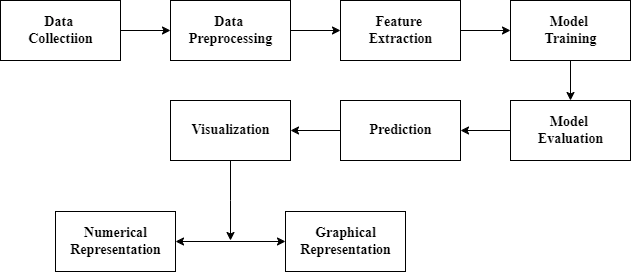
**6. Deployment:** After completing all the phases, software is deployed to its work environment where users can interact with it and provide further feedback if any improvement or correction is required.

**7. Maintenance:** In the maintenance phase, after deployment of the software in the working environment there may be some bugs, some errors or new updates are required. Maintenance involves debugging, enhancement of models and expansion of the application.

Advantages of iterative process model:

* Repeated cycles of development allow for continuous improvement and refinement of the software.
* Early prototypes provide a possible product that user can review and provide feedback on.
* Frequent testing and evaluation throughout each iteration help identify and fix defects early.
* Flexibility in adjusting changes is increased, as each iteration provides an opportunity to adjust requirements based on user feedback and evolving needs.

## 3.2 BLOCK DIAGRAM



*Figure 2: System Block Diagram*

This block diagram outlines the main components of the stock prediction system that are explained below:

### ****3.2.1 Data Collection****

Data collection is the foundational step of our stock market prediction system, ensuring the availability of reliable and accurate data for both historical analysis and real-time updates. The data collection process is divided into two parts: **historical stock price data collection** and **live data acquisition**, which includes sentiment-related information.

**1. Historical Data Collection:** To train our prediction model, we downloaded historical stock price data in **CSV format** from **NepseAlpha**, a reliable source for Nepal Stock Exchange (NEPSE) data. This dataset contains critical information such as opening price, closing price, high, low, and trading volume for individual stocks. This data was thoroughly verified to ensure its integrity and completeness before being used for model training. By focusing on high-quality data from NepseAlpha, we ensured that the model was built on a robust foundation of accurate historical trends.

**2. Live Stock Data Collection:** For real-time updates in the web application, live stock market data is acquired using a **web scraping script**. The script is designed to extract updated stock prices and other relevant information from **ShareSansar**, a trusted financial news and stock market analysis platform. To maintain accuracy and efficiency, the script is scheduled to run automatically **at the end of each trading day**, after the market closes. This new data is appended to the previously downloaded historical dataset, ensuring the model stays up-to-date with the latest market trends. The updated CSV file is then used by the web application to provide real-time predictions to users.

The data collected from **NepseAlpha and ShareSansar contains following parameters:**

* **Date:** The specific trading day to which the stock data corresponds.
* **Open:** The stock's price at the beginning of the trading session on that date.
* **High:** The highest price the stock reached during the trading session.
* **Low:** The lowest price the stock traded at during the session.
* **Close:** The stock's price at the end of the trading session.
* **Percentage Change:** The relative change in the stock's price compared to the previous day, expressed as a percentage.
* **Volume:** The total number of shares traded during the session, indicating the stock's market activity.

**3. Sentiment Data Collection:** To integrate sentiment analysis into the system, we utilize two main data sources:

* **Sentiment Word Dataset:** A dataset containing a list of positive and negative words is fetched from **GitHub** repositories. This dataset serves as a reference for determining the sentiment polarity of textual data, helping the system classify whether a news article or post is positive, negative, or neutral.
* **Live News Articles and Social Media Data:** Live textual data is collected using another **web scraping script**. This script is designed to fetch headlines and articles from various **financial news websites and social media platforms**. These data give insights into the public sentiment around specific companies and market conditions.

**4. Automation and Scalability:** To ensure efficiency, the data collection process is largely automated:

* The web scraping scripts for both stock prices and news headlines are scheduled to run at specific intervals, ensuring the datasets are always up-to-date.
* The entire process is designed to be scalable, allowing easy integration of new stock symbols, additional data sources, or different markets in the future.

By combining high-quality historical data with regularly updated live data, this comprehensive data collection process forms the backbone of our prediction system. It ensures the availability of accurate, timely, and sentiment-enriched data, enabling the model to generate reliable predictions and insights.

### ****3.2.2 Data Preprocessing****

Data preprocessing is a critical phase in the stock market prediction system. It ensures that raw data is cleaned, standardized, and transformed into a format suitable for use in machine learning models like LSTM. The preprocessing steps applied to the historical stock price data are explained below, along with the planned steps for sentiment analysis preprocessing.

**1. Preprocessing Historical Data for LSTM**

**a. Handling Missing Values:** The historical stock price data collected from NepseAlpha was checked for missing values, such as blank entries for certain dates. Missing data points were imputed using interpolation methods or ignored based on their relevance to ensure time-series continuity. This ensured that the model learned effectively from sequential data without any inconsistencies.

**b. Sorting by Date:** The data was sorted chronologically by the "Date" column to maintain the sequential order required for time-series analysis. Chronological order allowed the LSTM model to capture temporal dependencies in the data, which is critical for making future predictions.

**c. Scaling the Data:** The stock prices were normalized using the MinMaxScaler, scaling values between 0 and 1. This step ensured that all numerical features, particularly the "Close" price, were on a similar scale. Just the close price was use for the initial model training and testing which is to be further expanded for other relevant parameters as well. Normalization was necessary to ensure model works for every company’s data as price range for every company varies from each other.

*Normalized value of X= (X- Min value of X)/(MAX-MIN)*

**d. Splitting the Data into Training and Testing Sets:** The dataset was divided into 80% for training and 20% for testing. The training set was used to teach the model patterns in the stock price data, while the testing set evaluated its performance on unseen data. This split ensured that the model’s accuracy was tested on realistic scenarios.

**e. Creating Sequences for LSTM:** The data was structured into sequences of 60-time steps, where each sequence contained stock prices from the previous 60 days, and the target was the stock price on the 61st day. A custom function was implemented to create these sequences, and the resulting data was reshaped into a format suitable for LSTM input. This step allowed the model to learn relationships between past and future stock prices, capturing trends and temporal dependencies.

**2. Future Prediction Data Preparation**

To predict future prices, the last sequence of 60 days from the training data was extracted. This sequence served as the starting point for predicting prices iteratively, day by day, for the desired number of future days. Each prediction was appended to the sequence to generate the next input, simulating how the model would perform in a real-world scenario.

**3. Preprocessing for Sentiment Analysis**

Preprocessing for sentiment analysis will involve the following steps:

**a. Text Cleaning:** Special characters, numbers, and irrelevant text from news articles and social media posts will be removed.

**b. Tokenization:** Text will be split into individual words or phrases to enable easier analysis.

**c. Stop-Word Removal and Lemmatization:** Common words (e.g., "is," "the") will be removed, and words will be converted to their base form (e.g., "running" to "run").

**d. Vectorization:** Processed text will be converted into numerical data using techniques like TF-IDF or word embeddings.

These steps will ensure that textual data is prepared for sentiment analysis, allowing sentiment features to be extracted effectively.

This structured preprocessing ensures that historical stock data is clean, normalized, and ready for training, while also laying the groundwork for sentiment analysis. By focusing on data quality, the system is optimized for accurate and reliable predictions.

### 3.2.3 Feature Extraction

Feature extraction is a crucial step in the stock prediction system that focuses on identifying and isolating meaningful patterns and indicators from the raw data. These features are then used as inputs for the machine learning models, enabling them to better understand trends and make accurate predictions. Below is a detailed breakdown of the feature extraction process in the project:

**Feature Extraction from Historical Data:**

The historical stock price data contains several key parameters such as **Open**, **High**, **Low**, **Close**, and **Volume**. To enhance the predictive capability of the model, additional technical features were generated:

* **Moving Averages:** Calculated the average stock price over a specified time period (e.g., 7-day, 30-day moving averages). This smoothed out short-term fluctuations and highlighted overall trends in the data.
* **Percentage Change:** Measured the relative change in stock prices compared to the previous day. This feature provided insight into daily price volatility and market behavior.
* **Volume:** Included as an independent feature since it reflects market activity and liquidity, which often correlates with price movement.

These features were scaled and normalized along with the primary "Close" price to ensure consistency and compatibility with the LSTM model.

**Planned Feature Extraction from Sentiment Data:**

For sentiment analysis, textual data such as news articles and social media posts will be processed to extract sentiment-based features. These include:

* **Sentiment Polarity: This i**ndicates whether the sentiment in the text is positive, neutral, or negative. Tools like **TextBlob** or **NLTK** will be used to assign polarity scores to each text item.
* **Sentiment Subjectivity: This m**easures whether the sentiment is opinion-based (subjective) or fact-based (objective), helping in identifying text that is likely to influence market behavior.
* **Keyword-Based Features:** This identifies specific financial keywords and their context in the text (e.g., "profit," "loss," "growth"), helping to capture industry-specific sentiment relevant to stock predictions.

The process integrates technical indicators from historical stock data with sentiment-based insights, providing a holistic view of market trends. Through feature extraction, only the most relevant and meaningful data is selected, minimizing noise and enhancing prediction accuracy. The extracted features from both historical and sentiment data are combined into a unified dataset, which serves as input for LSTM and CNN models. These models learn patterns and relationships that influence market movements, bridging the gap between raw data and actionable insights. This process lays a strong foundation for accurate and effective stock price predictions.

### 3.2.4 Model Training

Model training is a vital phase in the stock prediction system where the machine learning models are taught to recognize patterns and relationships in the data. This phase ensures that the models can generalize well to make accurate predictions on unseen data. The process of training models in the project, specifically the LSTM for historical data and the planned CNN for sentiment analysis, is explained in detail below.

**1. Training the LSTM Model for Historical Data**

The Long Short-Term Memory (LSTM) model, a type of recurrent neural network (RNN), was used for analyzing sequential stock price data.

* **Input Data Preparation:** Historical stock data was preprocessed into sequences of 60-time steps, where each sequence represented stock prices over 60 days, and the target was the stock price on the 61st day. The data was divided into 80% training and 20% testing sets to ensure the model could generalize well to new data.
* **Model Architecture:** The LSTM model consisted of multiple layers: Two LSTM layers with 50 units each to capture temporal dependencies and a dense output layer to predict the closing price. The input shape was designed to match the sequence length (timesteps) and number of features.
* **Training Process:** The model was compiled with **'adam' optimizer** to adjust weights efficiently during backpropagation and **mean squared error (MSE)** as the loss function to minimize prediction errors. It was trained over 100 epochs with a batch size of 64, balancing computation time and convergence.
* **Output:** The trained LSTM model was saved in .keras format to be integrated into the web application for real-time predictions.

**2. Planned CNN Model for Sentiment Analysis**

The Convolutional Neural Network (CNN) will be employed to analyze textual data and extract sentiment-related features.

* **Input Data Preparation:** Textual data will be preprocessed (tokenized, vectorized, and converted into embeddings) to represent each word numerically. Sentiment labels (positive, neutral, or negative) will be assigned based on predefined datasets or manually labeled examples.
* **Model Architecture:** The CNN model will include:
  + Convolutional layers to identify sentiment-related patterns in the text.
  + Pooling layers to reduce dimensionality and focus on important features.
  + Dense layers for classification of sentiment polarity and extraction of sentiment scores.
* **Training Process:** The model will be trained using labeled textual data from news articles and social media. Cross-entropy loss will be used as the loss function for classification tasks. Performance will be monitored through metrics like accuracy, precision, and recall.

**3. Continuous Evaluation and Improvement**

During training, evaluation metrics like **Mean Squared Error (**MSE) and Root **Mean Squared Error (RMSE)** were used to measure the LSTM model’s accuracy. The training process also included hyperparameter tuning (e.g., adjusting the learning rate and number of epochs) to optimize the model's performance. After integrating sentiment analysis, the impact of sentiment on predictions will be analyzed to ensure its relevance and accuracy.

By designing and training models specifically for their tasks, the system leverages LSTM's strength in handling sequential data and CNN's ability to analyze textual data. This structured approach ensures the models are robust and capable of generating reliable stock market predictions.

### 3.2.5 Model Integration:

Model integration combines the outputs and features of separate machine learning models into a unified system, enabling more comprehensive stock price predictions. This process integrates the LSTM model, which analyzes historical stock price data, with the CNN model, which processes sentiment data from news articles and social media. By merging quantitative trends and qualitative sentiment insights, the system enhances prediction accuracy.

**Inputs from Each Model:**

* **LSTM Model:** Processes time-series data from historical stock prices to predict future prices. It captures long-term and short-term trends, technical patterns, and market volatility.
* **CNN Model:** Processes textual data to extract sentiment-related features, such as polarity (positive, neutral, or negative) and subjectivity (fact-based or opinion-based). These features reflect market sentiment, which can influence short-term stock price changes.

**Combining the Outputs:**

The integration of both models is carried out using two approaches:

1. **Feature-Level Fusion:** Features extracted from both models, such as LSTM-predicted trends and sentiment polarity from CNN, are combined into a single dataset. This merged feature set is then used as input for another predictive model, such as a neural network, to generate final stock price predictions.
2. **Prediction-Level Fusion:** Each model generates independent predictions: the LSTM forecasts future prices, while the CNN predicts sentiment-driven price changes. These predictions are weighted and combined to produce a unified output:  
   *P Final = α (P LSTM) + β (S sentiment)*

where α and β are tunable weights optimized during model evaluation.

**Implementation Steps:**

1. **Train Models Independently:** The LSTM model is trained on historical data, while the CNN model is trained on sentiment-labeled textual data.
2. **Feature Extraction:** Extract features such as predicted price trends from LSTM and sentiment scores from CNN.
3. **Feature or Prediction Fusion:** Merge features into a combined input dataset or combine predictions using weighted summation.
4. **Evaluate and Refine:** Use metrics like RMSE or MAPE to validate the combined model’s performance and fine-tune weights for optimal accuracy.

**Benefits of Integration:**

* Improves accuracy by combining complementary perspectives: market trends from LSTM and sentiment insights from CNN.
* Captures both long-term trends and short-term sentiment-driven fluctuations.
* Enhances investor confidence by providing predictions that reflect both quantitative and qualitative market dynamics.

By integrating the strengths of LSTM and CNN models, the system provides a more complete view of stock market behavior, offering investors better tools for making informed decisions.

### 3.2.6. Model Evaluation

Model evaluation ensures the accuracy and reliability of the stock prediction system by assessing the performance of the LSTM model for historical data and the planned CNN model for sentiment analysis.

**1. Evaluation Metrics for LSTM Model**

* **Mean Squared Error (MSE):** Measures the average squared difference between predicted and actual prices, with lower values indicating better accuracy.
* **Root Mean Squared Error (RMSE):** Expresses error in the same units as stock prices, offering a clear understanding of prediction accuracy.
* **Mean Absolute Percentage Error (MAPE):** Shows the average percentage error, helping evaluate relative accuracy.
* **Graphical Analysis:** Plots actual vs. predicted prices for visual comparison, highlighting patterns and deviations.

**2. Evaluation Metrics for Sentiment Analysis (Planned for CNN)**

* **Accuracy:** Percentage of correctly classified sentiments (positive, negative, neutral).
* **Precision, Recall, F1 Score:** Key metrics to evaluate sentiment classification performance.
* **Confusion Matrix:** Provides detailed insights into correct and incorrect classifications.

**3. Combined Model Evaluation**

After integrating LSTM and CNN models, the combined system is evaluated to ensure that sentiment data enhances stock predictions. Weighting schemes are optimized, and errors during high volatility or conflicting sentiments are analyzed.

**4. Tools and Techniques**

* **Python Libraries:** scikit-learn for metrics and Matplotlib for visualization.
* **Cross-Validation:** Ensures robustness by splitting data for repeated training and validation.

By rigorously evaluating models, the system ensures reliable predictions, supports informed investment decisions, and identifies areas for refinement.

### 3.2.7 Prediction

Prediction is the final and most critical step of the stock market prediction system, where the trained models are used to forecast future stock prices. This step combines the outputs of historical data analysis and, eventually, sentiment analysis to generate accurate and insightful predictions that assist investors in making informed decisions.

**1. Predictions Using LSTM Model**

* **Historical Data Prediction:**
  + The trained LSTM model takes a sequence of past stock prices (e.g., data from the last 60 days) as input.
  + It predicts the price for the next day by identifying patterns and trends from the historical data.
  + This process is repeated iteratively to forecast prices for multiple future days (e.g., 30 or 60 days).
* **Trend Continuity:**
  + To ensure consistency, the predicted price for the first day in the future is used as the starting point for predicting the next day, maintaining a realistic trend.
  + An offset is applied to align future predictions with the last predicted price from historical data, ensuring smooth continuity.

**2. Planned Integration with Sentiment Analysis**

* **Sentiment Adjustment:**
  + Sentiment scores extracted from news articles or social media will be combined with historical predictions.
  + Positive sentiment may boost predicted prices, while negative sentiment may lead to downward adjustments.
* **Refined Predictions:**
  + The final prediction will incorporate both quantitative market trends and qualitative sentiment insights for greater accuracy and relevance.

**3. Outputs**

* **Graphical Representation:**
  + The predicted prices are plotted alongside actual historical data for easy comparison.
  + A future price curve extends beyond the known data, providing a visual forecast of market trends.
* **Tabular Representation:**
  + Future predictions are presented in a table format, showing day-by-day forecasted prices for a specified period.

**4. Tools and Implementation**

* The trained LSTM model is loaded using TensorFlow/Keras, and predictions are generated programmatically.
* Chart.js and Matplotlib are used to visualize predictions in the web application, while tabular data is dynamically generated for clarity.

By leveraging prediction capabilities, the system provides actionable insights into future stock trends, helping users assess market opportunities and risks with confidence.

### 3.2.8 Visualization

Visualization plays a crucial role in the stock market prediction system, enabling users to interpret the model’s predictions and insights in an intuitive and user-friendly manner. By presenting data through charts, tables, and graphs, the system effectively communicates trends, patterns, and predictions to assist users in making informed investment decisions.

**1. Visualization of Historical and Predicted Data**

* **Actual vs. Predicted Prices:** The model's predictions for historical stock prices (from the test dataset) are plotted alongside the actual stock prices on a **line graph**. This comparison allows users to visually assess the accuracy of the model. This graph is typically color-coded (e.g., blue for actual prices and red for predicted prices) to distinguish between the two data sets clearly.
* **Future Price Predictions:** Predicted prices for the future period (e.g., the next 30 days) are plotted as an extension of the historical graph. The future prediction curve starts from the last predicted value and highlights the projected stock price trend.
* **Tabular Representation:** Predicted prices for future days are also displayed in a **table format**, where each row corresponds to a day, and the columns show the day number and predicted stock price. This tabular data complements the graph by providing precise numerical values for each prediction.

**2. Visualization of Sentiment Analysis Results**

* **Sentiment Trends Over Time:** A **line or bar graph** will visualize sentiment polarity over time. Positive, neutral, and negative sentiment scores will be aggregated daily or weekly to show how public sentiment fluctuates over time.
* **Correlation Between Sentiment and Price Trends:** A combined graph will overlay sentiment scores on historical stock prices, allowing users to analyze how sentiment changes influence market behavior.

**3. Web Application Integration**

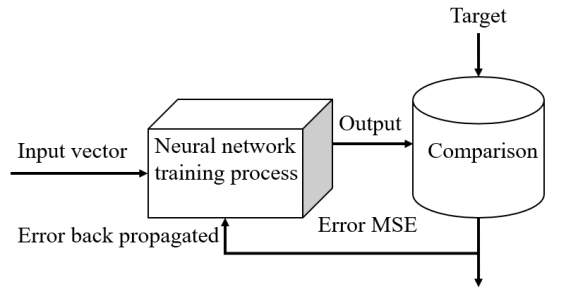
* The visualizations are integrated into the web application, where users can select a stock symbol and specify the prediction duration.
* Once the prediction is processed, the web application dynamically updates the visual elements, displaying:
  + A graph comparing actual vs. predicted prices and future predictions.
  + A table showing detailed numerical values for future predictions.

## 3.3 ALGORITHMS

### 3.3.1 Recurrent Neural Network (RNNs)

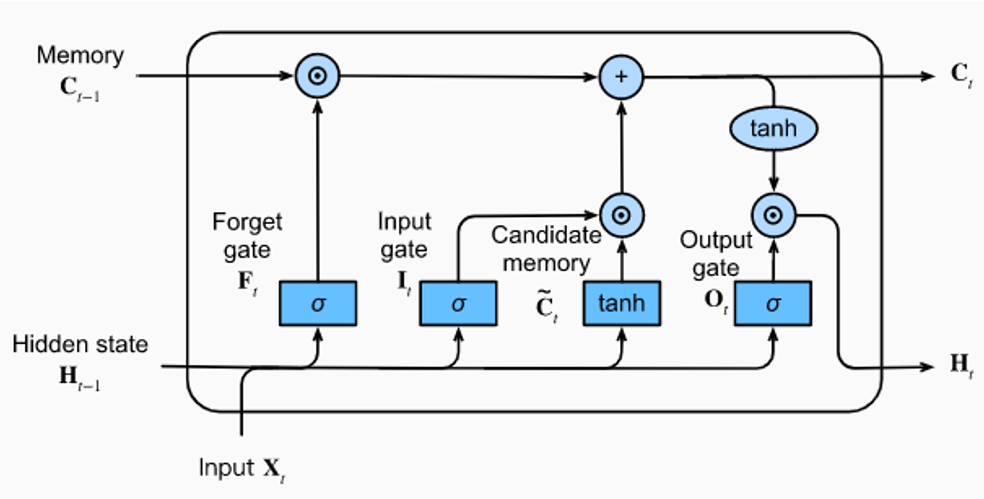
Recurrent Neural Networks (RNNs) are a type of neural network that are designed to process sequential data. They can analyze data with a temporal dimension, such as time series, speech, and text. RNNs can do this by using a hidden state passed from one timestep to the next. The hidden state is updated at each timestep based on the input and the previous hidden state. This structure allows RNNs to retain information about past inputs, effectively enabling them to exhibit a kind of memory over sequences.

However, standard RNNs face challenges when dealing with long sequences due to the problem of vanishing or exploding gradients during backpropagation. This makes it difficult for them to learn long-term dependencies. To address these issues, specialized architectures like Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) were developed. These architectures include mechanisms such as gates to better regulate the flow of information, allowing them to capture both short-term and long-term dependencies more effectively. RNNs are widely used in applications like language modeling, machine translation, video analysis, and stock price prediction. They are particularly well-suited for tasks where the order or context of data significantly impacts the output, as they inherently process data in sequence.



*Figure 3: Recurrent Neural Network*

### 3.3.2 Long Short-Term Memory (LSTM)

LSTMs Long Short-Term Memory is a type of RNNs that can detain long-term dependencies in sequential data. They use memory cell and gates to control the flow of information, allowing them to selectively retain or discard information as needed and thus avoid the vanishing gradient problem that plagues traditional RNNs. LSTMs are widely used in applications such as [natural language processing](https://www.simplilearn.com/tutorials/artificial-intelligence-tutorial/what-is-natural-language-processing-nlp), [speech recognition](https://www.simplilearn.com/tutorials/python-tutorial/speech-recognition-in-python), and time series forecasting.

*Figure 5: Long Short-Term Memory*

There are three types of gates in an LSTM:

**Input gate:** It decides which information to store in the memory cell. It is trained to open when the input is important and close when it is not.

**Forget gate:** It decides which information to discard from the memory cell. It is trained to open when the information is no longer important and close when it is.

**Output gate:** It is responsible for deciding which information to use for the output of LSTM. It is trained to open when the information is important and close when it is not.

The gates in an LSTM are trained to open and close based on the input and the previous hidden state. This allows the LSTM to selectively retain or discard information, making it more effective at capturing long-term dependencies.

Stock prices exhibit temporal dependencies, where current values depend on past trends and patterns. LSTM is highly effective in capturing these dependencies because:

* It processes sequential data and remembers trends over long periods.
* It can model the fluctuations and volatility inherent in stock prices.

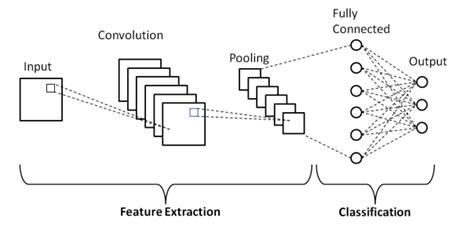
**Application in the Project:**

In this project, the LSTM model is used to predict future stock prices based on historical data.

* **Training Data:** The model is trained on sequences of past stock prices (e.g., the last 60 days) to predict the next day’s price.
* **Handling Time-Series Data:** The model effectively captures patterns such as moving averages, volatility, and trends over time.
* **Output:** After training, the LSTM model generates predictions for future stock prices, which are displayed as graphs and tables in the web application.

### 3.3.3 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are powerful deep learning models primarily used for image and text data analysis. They are particularly effective at processing data with a grid-like structure, such as images, by employing convolutional layers that apply filters to detect local patterns like edges, textures, and shapes. This capability allows CNNs to hierarchically extract features, from low-level details to high-level abstractions, making them ideal for tasks like object recognition, image classification, and natural language processing.



*Figure 4: Convolution Neural Network*

In the context of stock market prediction, CNNs excel at sentiment analysis by extracting meaningful patterns from textual data such as news articles, social media posts, and financial reports. Their ability to recognize contextual nuances in text enables them to classify sentiments as positive, negative or neutral associated with stock-related discussions. By identifying sentiments and trends, CNNs contribute significantly to improving predictive accuracy when integrated with time-series models like LSTM. This combination leverages CNNs for feature extraction and LSTMs for temporal modeling, providing a robust framework for understanding both static and dynamic aspects of market data.

Sentiment analysis involves processing textual data, such as news headlines and social media posts, to determine the sentiment (positive, negative, or neutral). CNN is well-suited for this task because:

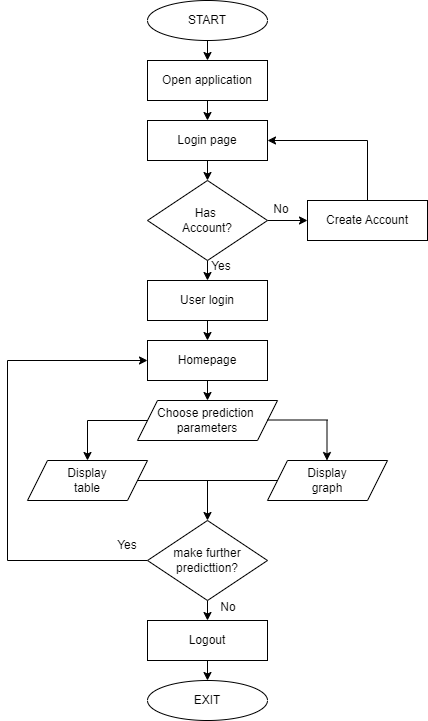
* It can identify patterns and features in text, such as recurring sentiment indicators (e.g., "bullish," "growth").
* It is computationally efficient and can handle large volumes of text data.

**Application in the Project:**

In the context of this project, CNN is used to extract sentiment-related features from textual data:

* **Input Data:** News articles and social media posts are preprocessed into numerical representations (e.g., word embeddings).
* **Feature Extraction:** The CNN model analyzes the text to identify sentiment polarity and subjectivity, which are quantified into scores.
* **Output:** The sentiment scores are integrated with historical data predictions to enhance the accuracy of stock price forecasts.

## 3.4 FLOWCHART



*Figure 6: Flowchart*

## 3.6 Necessary Diagrams

### 3.6.1 Data Flow Diagram

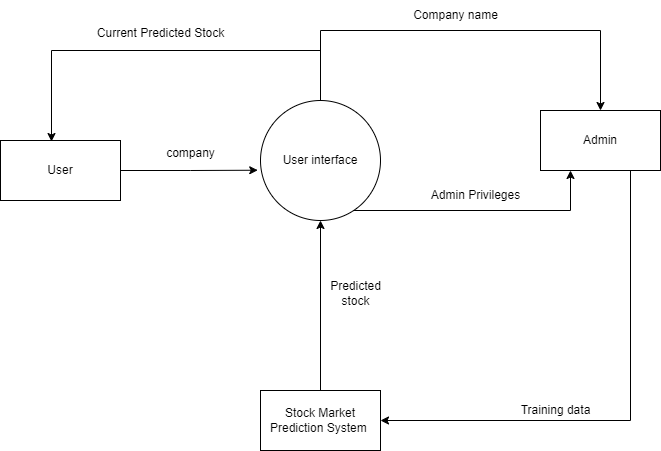
The Data Flow Diagram of the Stock Market Prediction is shown below:

**Data Flow Diagram - 0**



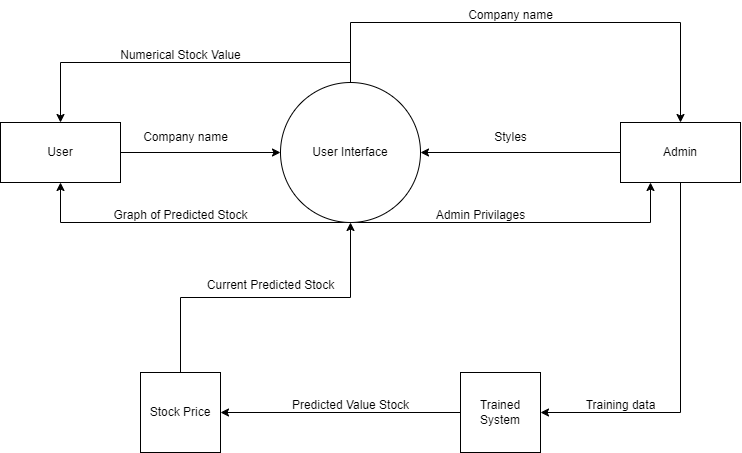
*Figure 7: Data Flow Diagram 0*

**Data Flow Diagram – 1**



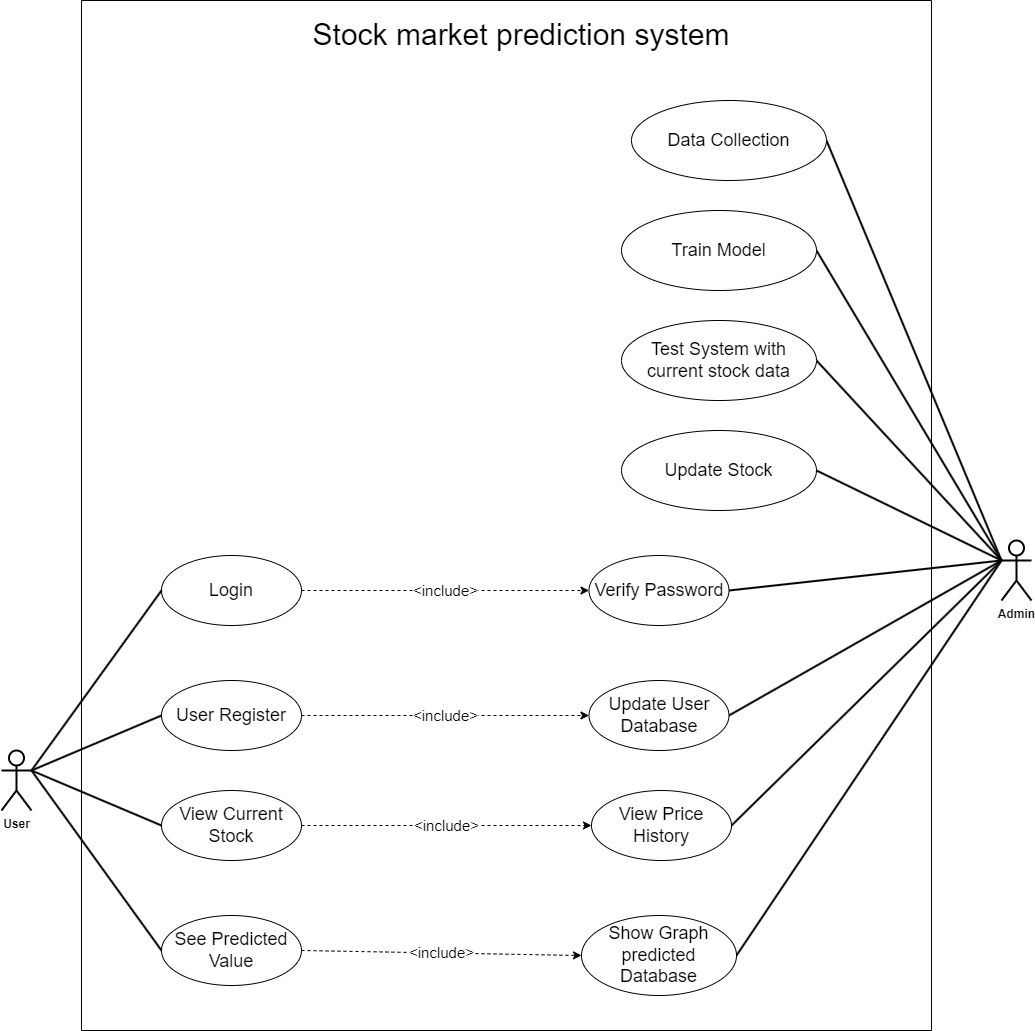
*Figure 8: Data Flow Diagram Level - 1*

**Data Flow Diagram Level – 2**



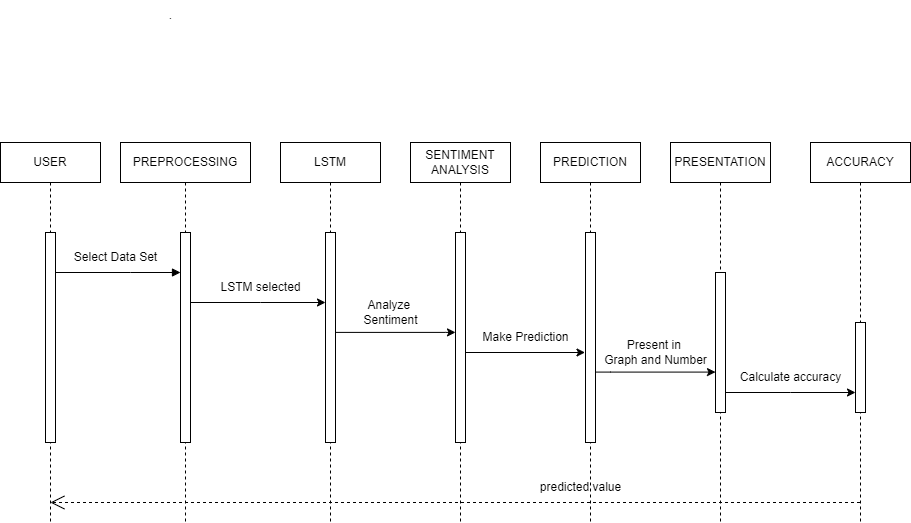
*Figure 9: Data Flow Diagram Level-2*

### 3.6.2 Use Case Diagram



*Figure 10: Use Case Diagram*

### 3.6.3 Sequence Diagram



*Figure 11: Sequence Diagram*

## 3.7 TOOLS USED

### 3.7.1 Python

Python has been used for most of the works in this project which has been explained below:

* Building and visualizing the LSTM model using libraries pandas, numpy, sklearn, tensorflow, matplotlib
* Integrating the model into web application using flask
* Fetching live and updated dataset via web scraping script using libraries beautiful soup and selenium.

### 3.7.2 HTML/CSS/JavaScript

HTML, CSS and JavaScript has been used to create the interactive and easy to use frontend UI of the web application.

* HTML has been used to structure the content of webpages.
* CSS was used to styles and formats their appearance of the webpages.
* JavaScript has been used to fetch the data numerical and graphical from the backend to display in the web interface.

### 3.7.3 GitHub

* GitHub has been sources for certain datasets required for the model training.
* It is used as online repository for the project ensuring proper collaboratory workflow among the team members

# CHAPTER 4: EPILOGUE

## 4.1 TASK COMPLETED

1. **Data Collection**:

* Gathered historical stock market data for analysis and modeling.
* Collected datasets for natural language analysis from news articles.

1. **Web Scraping**:

* Developed a web scraping script to fetch live stock market data.
* Created a script to scrape news headlines and tweets for sentiment analysis.

1. **Model Development**:

* Trained an LSTM model using the historical stock market dataset.
* Integrated the trained LSTM model with the web application for real-time predictions.

1. **User Interface**:

* Designed and implemented an interactive, visually appealing user interface for the web application.

1. **Database Setup**:

* Established a database to manage user information for the web application.

## 4.2 TASK REMAINING

* Train and integrate sentiment analysis for enhanced stock predictions.
* Integrate the database in web application.
* Pipeline live data of company stocks.
* Conduct thorough testing, fix bugs, and optimize performance.

## 4.2 GANTT CHART

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tasks | JUN | JUL | AUG | SEP | OCT | NOV | DEC | JAN | FEB | MAR |
| Documentation |  |  |  |  |  |  |  |  |  |  |
| Planning |  |  |  |  |  |  |  |  |  |  |
| Analysis |  |  |  |  |  |  |  |  |  |  |
| Design |  |  |  |  |  |  |  |  |  |  |
| Implementation |  |  |  |  |  |  |  |  |  |  |
| Testing |  |  |  |  |  |  |  |  |  |  |

*Figure 12: Gantt Chart*

# REFERENCES

1. Polepally, Vijay & Jakka, Neha & Vishnukanth, Pendly & Raj, Rachakatla & Anish, Gudavelli. (2023). An Efficient Way to Predict Stock Price using LSTM-RNN Algorithm. *7th International Conference on Intelligent Computing and Control Systems (ICICCS), 1240-1244,* 2023.
2. Janak Kumar Lal, Arun Kumar Timalsina. (2022). A CNN-BGRU Method for Stock Price Prediction. Proceedings of 11th IOE Graduate Conference, *2350-8914 (Online), 2350-8906 (Print),* 2022.
3. Kabir, Md Humayun & Sobur, Abdus & Amin, Md Ruhul. (2023). Stock Price Prediction Using the Machine Learning. *International Journal of Creative Research Thoughts (IJCRT). f946-f950,* 2023
4. Choi, Hyeong. (2018). Stock Price Correlation Coefficient Prediction with ARIMA-LSTM Hybrid Model, 2018.
5. Patel, J., Shah, S., Thakkar, P., & Kotecha, K. (2015). Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. Expert Systems with Applications, 42(1), 259-268.

# BIBLIOGRAPHY

* https://www.simplilearn.com/tutorials/artificial-intelligence-tutorial/lstm
* https://www ijcrt.org/papers/IJCRT2307700.pdf
* https://ieeexplore.ieee.org/abstract/document/10142902
* https://www.youtube.com/watch?v=b61DPVFX03I
* http://conference.ioe.edu.np/publications/ioegc11/ioegc-11-004-11005.pdf
* <https://www.geeksforgeeks.org/convolutional-neural-network-cnn-in-machine-learning>

# SCREENSHOTS

