STOCK PRICE PREDICTION

PROJECT REPORT

Group Members:

NAME	ROLL NO
N Sai Sandeep	AM.EN.U4CSE20049
P Laxmi Praneeth	AM.EN.U4CSE20052
Sidharth P V	AM.EN.U4CSE20066

STOCK PRICE PREDICTION

PROBLEM STATEMENT:

We will predict a signal that indicates whether buying a particular stock will be helpful or not by using ML. Stock price prediction is a challenging problem in finance, and machine learning has shown promise in this field. The **goal** of this project is to develop a machine learning model for predicting stock prices using historical data and financial features.

MOTIVATION:

The motive for a software engineering project aimed at stock price prediction could be to provide investors and traders with a reliable tool to make informed decisions about buying stocks. By developing a stock price prediction software, engineers can leverage machine learning models to analyse historical data and identify trends that could impact future stock prices. This can provide investors with valuable insights into which stocks to buy, hold or sell, based on the software's prediction.

SOLUTION:

The process involves collecting and preprocessing financial data, selecting, and training a machine learning model, and evaluating its performance. The project will explore different techniques for feature engineering and model selection and aim to achieve high accuracy and robustness in stock price prediction. The results of this project will contribute to the understanding of the use of machine learning in finance and have practical applications in investment portfolio management.

OBJECTIVE:

We will predict a signal that indicates whether buying a particular stock will be helpful or not by using ML. Stock price prediction is a challenging problem in finance, and machine learning has shown promise in this field. The goal of this project is to develop a machine learning model for predicting stock prices using historical data and financial features.

DATASETS:

https://www.kaggle.com/aaron7sun/stocknews

https://www.kaggle.com/code/ankiii07/nse-tataglobal-stock-price-prediction

DESIGN:

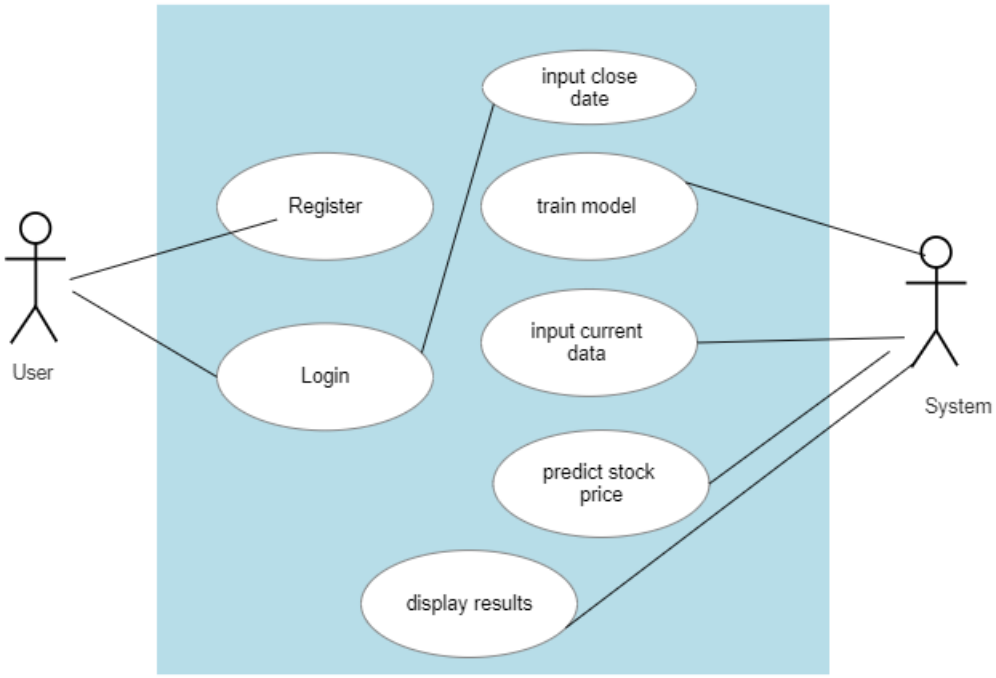
1. **Import Libraries**
   Importing libraries such as Numpy,Pandas, Scikit-Learn which helps us to handle data manipulation, preprocessing, modelling tasks.

2. **Collecting Data**
   In machine learning, algorithms are trained on data, and the quality of the data directly affects the quality of the model's predictions. The model learns from this data and uses it to make predictions on new, unseen data. Collecting a diverse range of data is critical to avoid bias in the model's predictions
3. **Data Cleaning and Preprocessing**
   Perform data cleaning tasks such as removing duplicates, filling in missing data, and handling outliers. Also, pre-process the data to make it suitable for machine learning algorithms by scaling, normalizing, and transforming features as needed.
4. **Feature engineering**
   Feature Engineering helps to derive some valuable features from the existing ones. These extra features sometimes help in increasing the performance of the model significantly and certainly help to gain deeper insights into the data.
5. **Selecting Machine Learning Model**
   Select an appropriate machine learning algorithm for the problem at hand. Some popular algorithms for stock price prediction include Linear Regression, Decision Trees, Random Forests, Gradient Boosting, and Deep Learning algorithms such as LSTM and CNN.
6. **Model Training**
   Train the selected model using the preprocessed data. This involves splitting the data into training and testing sets, training the model on the training set, and evaluating its performance on the testing set.
7. **Monitoring and Maintenance:**
   Continuously monitor the performance of the deployed model and maintain it by updating it with new data and retraining it as needed.

REQUIREMENTS:

Flask,
PyMySQL,
Keras,
Tensorflow,
Libraries:numpy,pandas,scikit-learn

USE CASE:



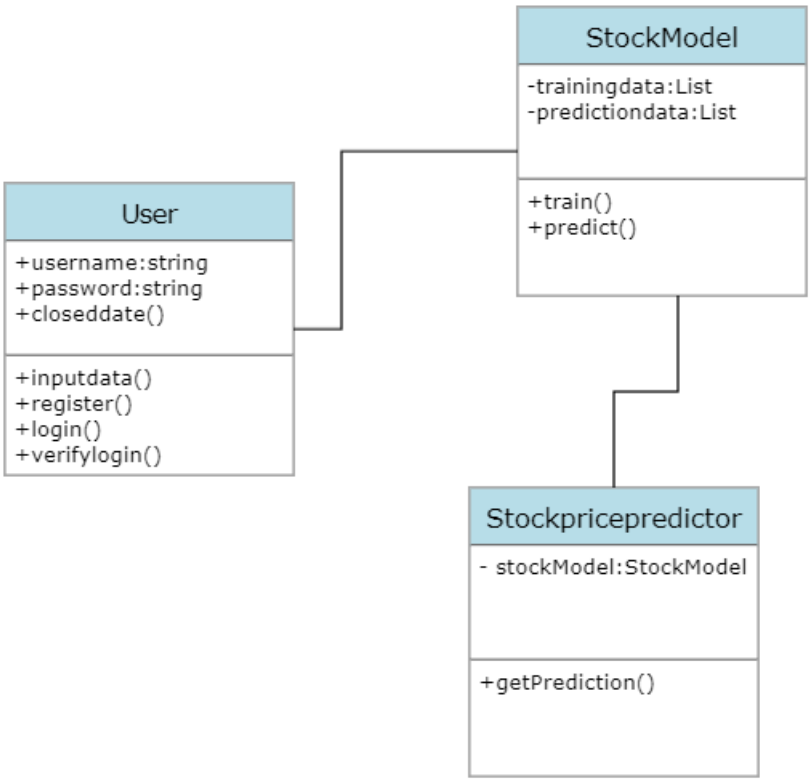
Actors:

- User: the person or entity who is interested in predicting stock prices.
- System: the software or machine learning model that provides stock price predictions.

Use cases:

- Input historical stock data: the user inputs historical stock data, which includes information such as the stock symbol, the date, the opening price, the closing price, and other relevant data.
- Train model: the system uses the historical stock data to train a machine learning model that can predict future stock prices.
- Input current stock data: the user inputs current stock data, which includes information such as the stock symbol, the date, and any other relevant data.
- Predict stock price: the system uses the trained machine learning model to predict the future stock price based on the current stock data.
- Display results: the system displays the predicted stock price to the user.

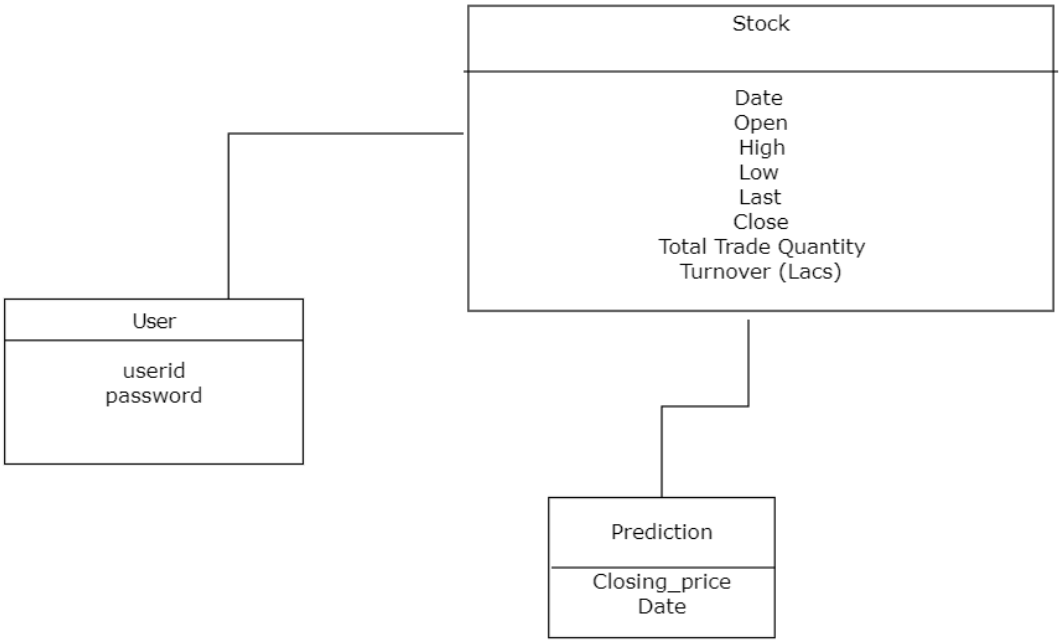
CLASS DIAGRAM:



In this diagram, we have three classes: User, StockModel, and StockPricePredictor.

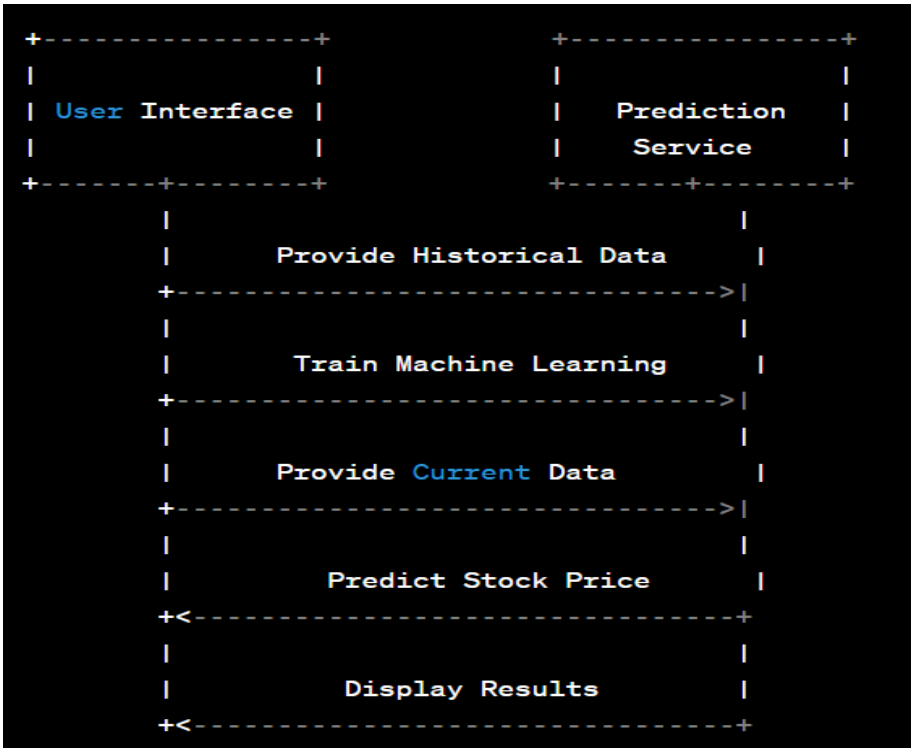
- The **User class** represents the person or entity who is interested in predicting stock prices. It has a method called inputData() which allows the user to input historical and current stock data.
- The **StockModel class** represents the machine learning model that is used to predict future stock prices. It has three attributes: stockSymbol, trainingData, and predictionData. The stockSymbol attribute represents the symbol for the stock that the model is predicting. The trainingData attribute is a list of historical stock data that is used to train the model. The predictionData attribute is a list of current stock data that is used to make predictions. The StockModel class also has two methods: train() which trains the model using the historical stock data, and predict() which uses the trained model to make predictions based on the current stock data.
- The **StockPricePredictor** class represents the system that provides stock price predictions. It has one attribute: stockModel, which is an instance of the StockModel class. The StockPricePredictor class also has a method called getPrediction() which uses the stockModel to predict the future stock price based on the current stock data.

ENTITY RELATION DIAGRAM:



- A User can have many Predictions.
- A Prediction can be associated with many Stocks.
- A Stock can be associated with many Predictions.
- A Stock has one StockPricePredictionModel.
- A StockPricePredictionModel can have many TrainingData and many PredictionData.

Data Flow Diagram:



The flow of information is as follows:

1. The User Interface prompts the user to provide historical stock data.

2. The User Interface sends the historical data to the Prediction Service.
3. The Prediction Service uses the historical data to train the machine learning model.
4. The User Interface prompts the user to provide current stock data.
5. The User Interface sends the current data to the Prediction Service.
6. The Prediction Service uses the trained machine learning model to predict the stock price based on the current data.
7. The Prediction Service sends the predicted stock price to the User Interface.
8. The User Interface displays the predicted stock price to the user.

Learning Algorithms used for prediction are linear regression, support vector machine, knn, random forest regressor, xgboost classifier.

COMPARING MODELS:

	Model	RMSE	MSE	MAE	r2 score
0	Linear Regression	0.385662	0.148736	0.355591	0.397863
1	SVM	0.288503	0.083234	0.203474	0.663038
2	Random Forest Regressor	0.342149	0.117066	0.117066	0.526074

Comparing rmse values svm has less compared to linear regression and rfr so it best fits among them. Comparing training and validation accuracy we got xgboost with less error.

Generating LSTM Model

For making the prediction live we used LSTM model # Generating LSTM Model. Basically LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) that is designed to address the problem of vanishing and exploding gradients that can occur in traditional RNNs. LSTM is particularly useful for tasks that involve sequential data, such as speech recognition, natural language processing, and time series prediction, where it is important to maintain information over longer time periods.

Backend for LSTM model

First, we will import dataset to work on and convert date into required format splitting data into train and test data generate lstm model and save it into saved_lstm_model.h5 this is all about backend part

Frontend for LSTM model

Coming to the frontend part By using FLASK a link will be generated where users will register if they are new users and login. They will enter the closing date and the saved model delivers the predicted price.