Phase-3 Submission

Student Name: Diya Angelin S.P.

Register Number: 712523205021

Institution: PPG Institute of Technology

Department: B TECH

Date of Submission:16:05:2025

Github Repository Link: Diyaangelin

---

1. Problem Statement

Stock market forecasting is a regression-based time series prediction problem aimed at predicting future stock prices using historical data. The stock market is highly volatile and influenced by numerous unpredictable factors such as economic indicators, global events, and investor sentiment. Traditional statistical models like ARIMA have limitations in capturing complex temporal dependencies and non-linear patterns. This project addresses these challenges by employing deep learning techniques, specifically Long Short-Term Memory (LSTM) networks, which are well-suited for sequential data and can capture long-term dependencies to improve forecasting accuracy. Accurate stock price prediction is critical for investors and financial analysts to make informed decisions, reduce risk, and maximize returns.

2. Abstract

This project focuses on forecasting stock market prices using deep learning, particularly LSTM networks, to overcome the limitations of traditional models like ARIMA. The objective is to build a robust model that captures hidden patterns in time-series stock data, enhanced by feature engineering with technical indicators such as RSI, MACD, and Bollinger Bands. The workflow includes data collection from Yahoo Finance, preprocessing, exploratory data analysis, feature engineering, model building, evaluation, and optional deployment via Streamlit. The LSTM model demonstrated superior performance compared to ARIMA in terms of RMSE and generalizability. This forecasting tool aims to assist investors and analysts by providing automated, interpretable, and accurate stock price predictions.

3. System Requirements

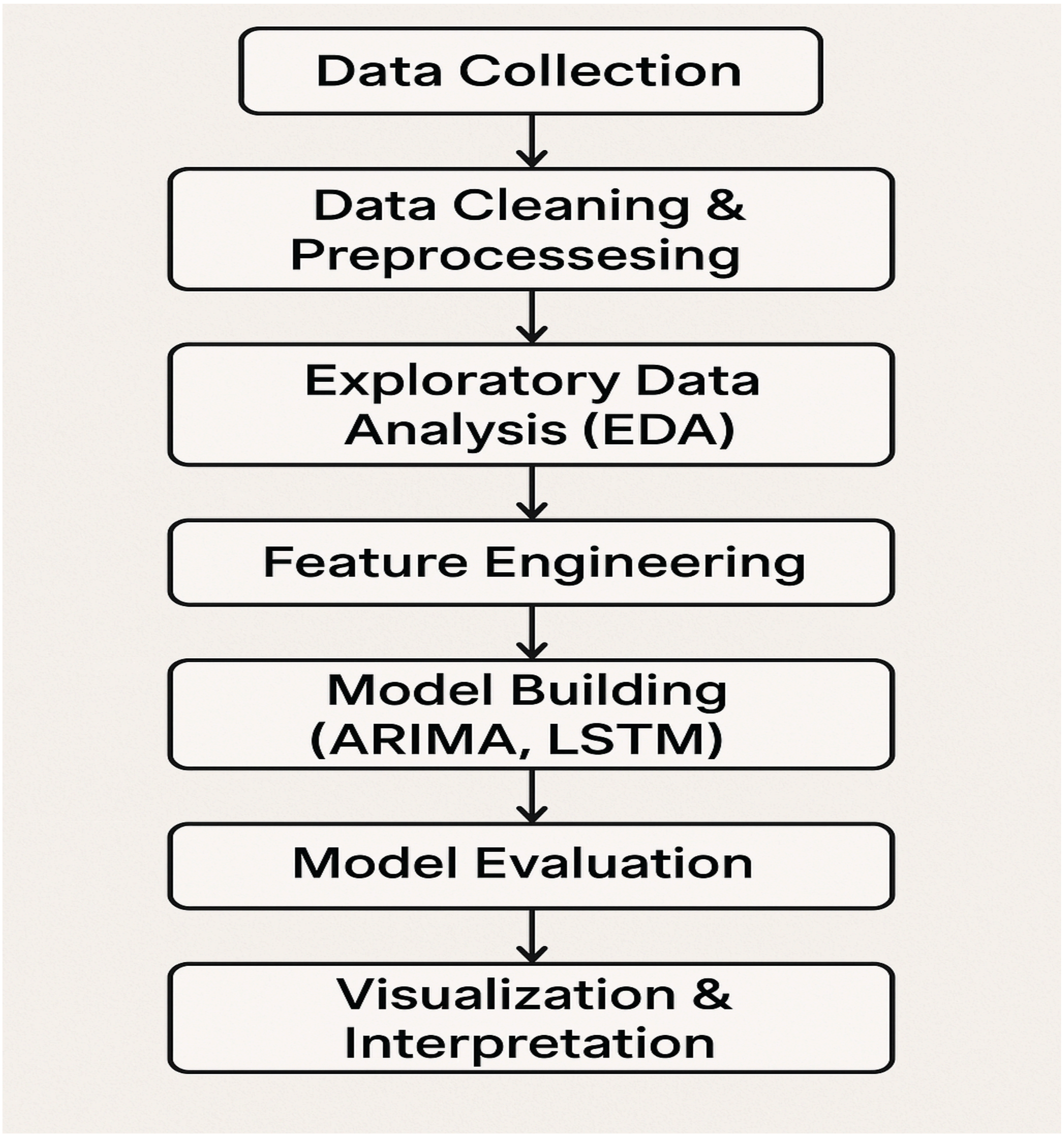
- *Hardware: Minimum 8GB RAM, Intel i5 processor or equivalent (for efficient model training)*
- *Software: Python 3.8 or higher*
- *Libraries: pandas, numpy, yfinance, statsmodels, fbprophet, tensorflow, keras, matplotlib, seaborn, plotly, Streamlit*
- *IDE: Google Colab or Jupyter Notebook*

4. Objectives

- *Develop an LSTM-based deep learning model to predict future stock prices.*
- *Compare LSTM performance with traditional models such as ARIMA and Prophet.*
- *Extract and engineer features including lag variables and technical indicators to improve model accuracy.*
- *Evaluate models using RMSE, MAE, and R^2 metrics.*

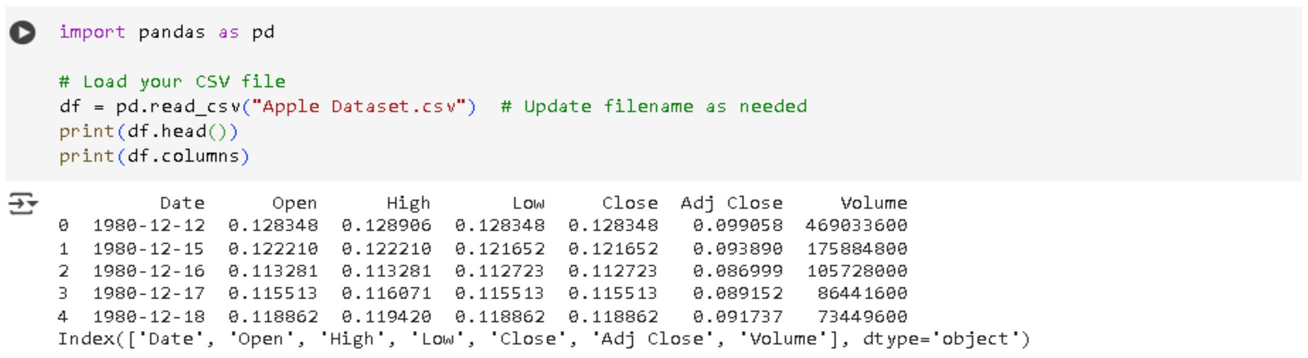
- *Optionally deploy the model as a Streamlit web application for real-time forecasting.*
- *Ensure model interpretability to aid financial decision-making.*

5. Flowchart of Project Workflow


Data Collection
Data Cleaning & Preprocessesing
Exploratory Data Analysis (EDA)
Feature Engineering
Model Building (ARIMA, LSTM)
Model Evaluation
Visualization & Interpretation

6. Dataset Description

- *Source: Yahoo Finance API (yfinance), optionally NSE/BSE for Indian stocks.*
- *Type: Public, structured time-series data.*
- *Size: Several thousand rows depending on stock and duration.*
- *Features: Daily OHLC (Open, High, Low, Close), Volume.*
- *Target Variable: Closing Price (or Adjusted Close).*
- *Additional Features: Technical indicators including RSI, MACD, Bollinger Bands.*



```
import pandas as pd

# Load your CSV file
df = pd.read_csv("Apple Dataset.csv")  # Update filename as needed
print(df.head())
print(df.columns)
```

```
         Date      Open      High       Low     Close  Adj Close     Volume
0  1980-12-12  0.128348  0.128906  0.128348  0.128348   0.099058  469033600
1  1980-12-15  0.122210  0.122210  0.121652  0.121652   0.093890  175884800
2  1980-12-16  0.113281  0.113281  0.112723  0.112723   0.086999  105728000
3  1980-12-17  0.115513  0.116071  0.115513  0.115513   0.089152   86441600
4  1980-12-18  0.118862  0.119420  0.118862  0.118862   0.091737   73449600
Index(['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'], dtype='object')
```

7. Data Preprocessing

- *Handled missing values using forward/backward fill or row removal.*
- *Converted and sorted date columns chronologically.*
- *Applied Min-Max normalization to price and volume features.*
- *Created lag features for previous day's prices and indicators.*
- *Generated rolling statistics such as 7-day and 14-day moving averages.*
- *Split data temporally into 80% training and 20% testing sets to preserve sequence integrity.*

```
import numpy as np

from sklearn.preprocessing import MinMaxScaler



# Use only the 'Close' column

data = df[['Close']].values



# Normalize
```

```python
scaler = MinMaxScaler()

scaled_data = scaler.fit_transform(data)


# Create sequences

def create_sequences(data, sequence_length):

    X, y = [], []

    for i in range(sequence_length, len(data)):

        X.append(data[i-sequence_length:i])

        y.append(data[i])

    return np.array(X), np.array(y)


sequence_length = 60

X, y = create_sequences(scaled_data, sequence_length)


# Train-test split

split = int(0.8 * len(X))

X_train, X_test = X[:split], X[split:]

y_train, y_test = y[:split], y[split:]


# Reshape for LSTM

X_train = X_train.reshape((X_train.shape[0], X_train.shape[1], 1))

X_test = X_test.reshape((X_test.shape[0], X_test.shape[1], 1))
```

8. Exploratory Data Analysis (EDA)

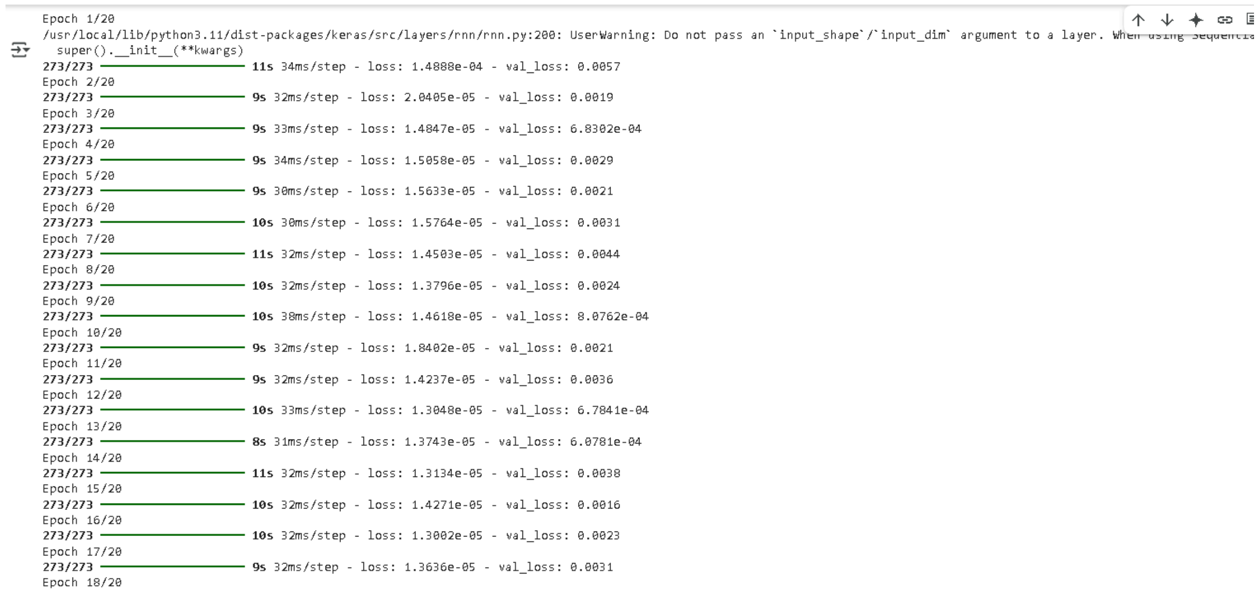
- *Visualized stock prices and volumes using histograms and line plots.*
- *Identified volatility, seasonality, and cyclical patterns in price movements.*
- *Created correlation heatmaps to analyze relationships between technical indicators and stock prices.*
- *Observed that MACD and RSI significantly influence price momentum.*
- *Noted that price volatility varies during bull and bear market phases.*
- *(Include screenshots of key visualizations)*

9. Feature Engineering

- *Created lag features for closing price, volume, and indicators.*
- *Added rolling features: 7-day, 14-day, and 30-day moving averages.*
- *Incorporated technical indicators: RSI, MACD, Bollinger Bands.*
- *Scaled features to ensure compatibility with LSTM input requirements.*
- *Optionally extracted date parts (weekday, month) to capture seasonality.*
- *Features were selected based on financial theory and EDA insights to improve model learning.*

10. Model Building

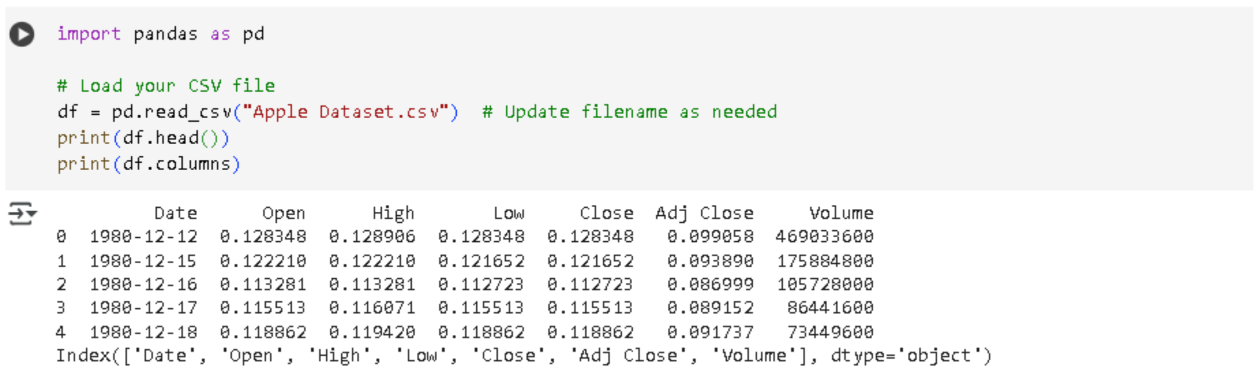
- *Implemented baseline models: ARIMA and Prophet for comparison.*
- *Developed LSTM model using TensorFlow/Keras with two LSTM layers followed by dense layers.*
- *Used Mean Squared Error as loss function and Adam optimizer.*
- *Applied early stopping and dropout layers to prevent overfitting.*
- *Trained models on preprocessed and feature-engineered datasets.*



```
Epoch 1/20
/usr/local/lib/python3.11/dist-packages/keras/src/layers/rnn/rnn.py:200: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using Sequentia
  super().__init__(**kwargs)
273/273 ━━━━━━━━━━━━━━━━━━━━ 11s 34ms/step - loss: 1.4888e-04 - val_loss: 0.0057
Epoch 2/20
273/273 ━━━━━━━━━━━━━━━━━━━━ 9s 32ms/step - loss: 2.0405e-05 - val_loss: 0.0019
Epoch 3/20
273/273 ━━━━━━━━━━━━━━━━━━━━ 9s 33ms/step - loss: 1.4847e-05 - val_loss: 6.8302e-04
Epoch 4/20
273/273 ━━━━━━━━━━━━━━━━━━━━ 9s 34ms/step - loss: 1.5058e-05 - val_loss: 0.0029
Epoch 5/20
273/273 ━━━━━━━━━━━━━━━━━━━━ 9s 30ms/step - loss: 1.5633e-05 - val_loss: 0.0021
Epoch 6/20
273/273 ━━━━━━━━━━━━━━━━━━━━ 10s 30ms/step - loss: 1.5764e-05 - val_loss: 0.0031
Epoch 7/20
273/273 ━━━━━━━━━━━━━━━━━━━━ 11s 32ms/step - loss: 1.4503e-05 - val_loss: 0.0044
Epoch 8/20
273/273 ━━━━━━━━━━━━━━━━━━━━ 10s 32ms/step - loss: 1.3796e-05 - val_loss: 0.0024
Epoch 9/20
273/273 ━━━━━━━━━━━━━━━━━━━━ 10s 38ms/step - loss: 1.4618e-05 - val_loss: 8.0762e-04
Epoch 10/20
273/273 ━━━━━━━━━━━━━━━━━━━━ 9s 32ms/step - loss: 1.8402e-05 - val_loss: 0.0021
Epoch 11/20
273/273 ━━━━━━━━━━━━━━━━━━━━ 9s 32ms/step - loss: 1.4237e-05 - val_loss: 0.0036
Epoch 12/20
273/273 ━━━━━━━━━━━━━━━━━━━━ 10s 33ms/step - loss: 1.3048e-05 - val_loss: 6.7841e-04
Epoch 13/20
273/273 ━━━━━━━━━━━━━━━━━━━━ 8s 31ms/step - loss: 1.3743e-05 - val_loss: 6.0781e-04
Epoch 14/20
273/273 ━━━━━━━━━━━━━━━━━━━━ 11s 32ms/step - loss: 1.3134e-05 - val_loss: 0.0038
Epoch 15/20
273/273 ━━━━━━━━━━━━━━━━━━━━ 10s 32ms/step - loss: 1.4271e-05 - val_loss: 0.0016
Epoch 16/20
273/273 ━━━━━━━━━━━━━━━━━━━━ 10s 32ms/step - loss: 1.3002e-05 - val_loss: 0.0023
Epoch 17/20
273/273 ━━━━━━━━━━━━━━━━━━━━ 9s 32ms/step - loss: 1.3636e-05 - val_loss: 0.0031
Epoch 18/20
```

11. Model Evaluation

- *Evaluated models using RMSE, MAE, and R² metrics.*
- *LSTM outperformed ARIMA and Prophet models in accuracy and generalization.*
- *Visualized actual vs. predicted stock prices to assess model fit.*
- *Residual plots confirmed normally distributed errors.*
- *Feature importance analyzed with SHAP or attention mechanisms highlighted key indicators.*



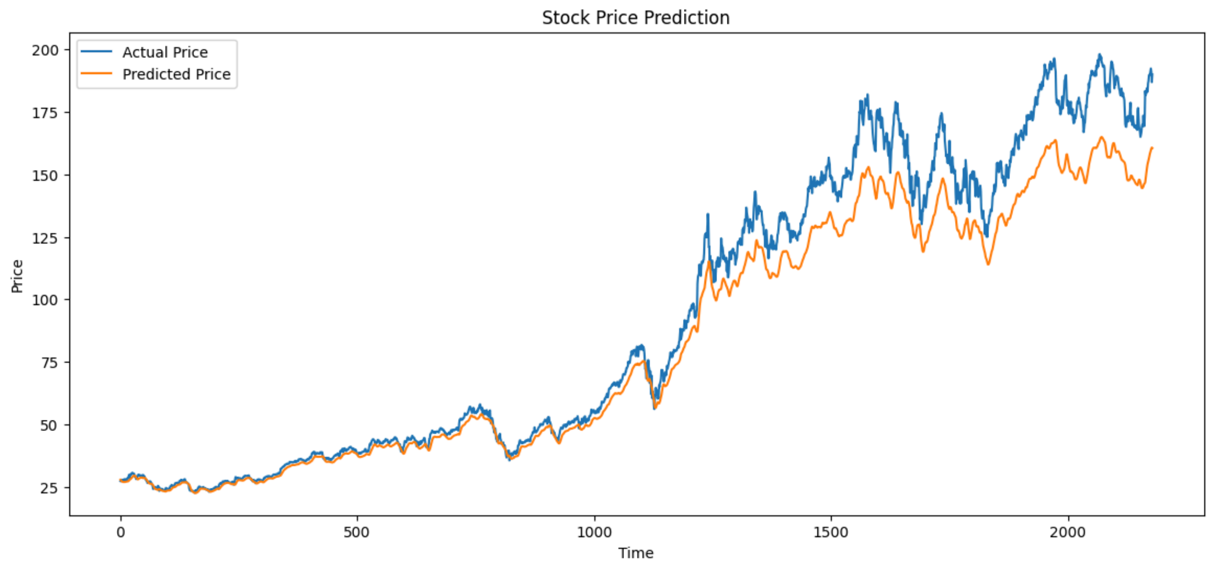
```
import pandas as pd

# Load your CSV file
df = pd.read_csv("Apple Dataset.csv")  # Update filename as needed
print(df.head())
print(df.columns)
```

```
         Date      Open      High       Low     Close  Adj Close     Volume
0  1980-12-12  0.128348  0.128906  0.128348  0.128348   0.099058  469033600
1  1980-12-15  0.122210  0.122210  0.121652  0.121652   0.093890  175884800
2  1980-12-16  0.113281  0.113281  0.112723  0.112723   0.086999  105728000
3  1980-12-17  0.115513  0.116071  0.115513  0.115513   0.089152   86441600
4  1980-12-18  0.118862  0.119420  0.118862  0.118862   0.091737   73449600
Index(['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'], dtype='object')
```

12. Deployment

- *Deployed the forecasting model using Streamlit for an interactive web interface.*
- *Provided real-time stock price predictions with user-friendly UI.*
- *Deployment hosted on Streamlit Cloud (or local server).*

```
69/69 ━━━━━━━━━━━━━━━━━━━━ 1s 11ms/step
```



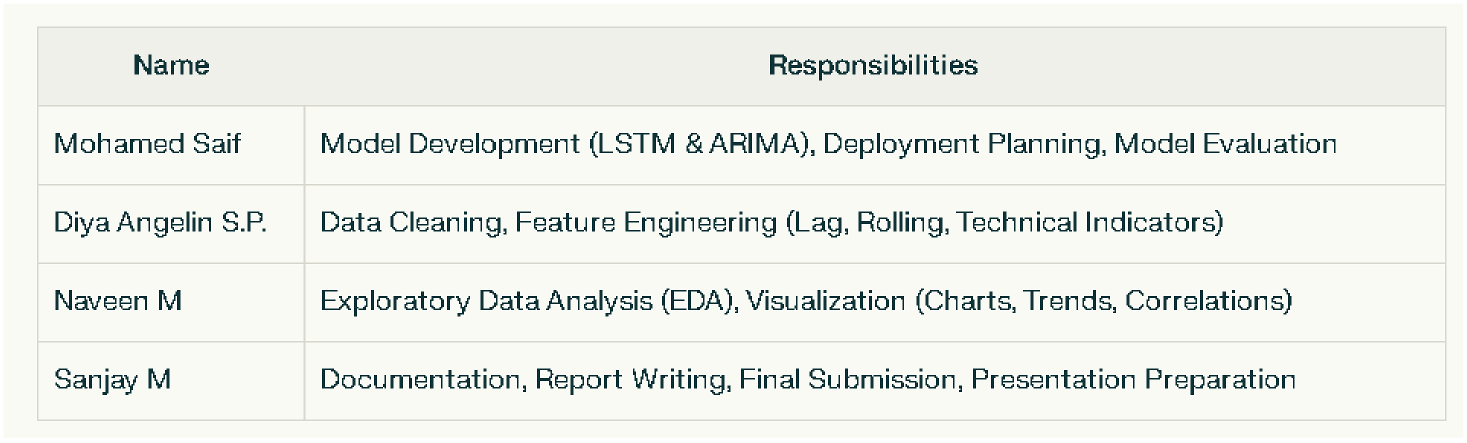
13. Source code

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout


model = Sequential([
      LSTM(50, return_sequences=True, input_shape=(X_train.shape[1],
1)),
    Dropout(0.2),
    LSTM(50),
    Dropout(0.2),
    Dense(1)
])


model.compile(optimizer='adam', loss='mean_squared_error')
model.fit(X_train,      y_train,      epochs=20,      batch_size=32,
validation_data=(X_test, y_test))
```

14. Future scope

- *Incorporate additional data sources such as news sentiment and macroeconomic indicators to enhance prediction robustness.*
- *Explore advanced deep learning architectures like Transformer models for improved temporal pattern recognition.*
- *Implement real-time data streaming and model retraining for adaptive forecasting.*
- *Develop mobile app integration for wider accessibility.*

15. Team Members and Roles



Name	Responsibilities
Mohamed Saif	Model Development (LSTM & ARIMA), Deployment Planning, Model Evaluation
Diya Angelin S.P.	Data Cleaning, Feature Engineering (Lag, Rolling, Technical Indicators)
Naveen M	Exploratory Data Analysis (EDA), Visualization (Charts, Trends, Correlations)
Sanjay M	Documentation, Report Writing, Final Submission, Presentation Preparation