**Date:24.10.25**

**TASK:11**

**Implementation of Stock Market Prediction.**

**CO1, CO2, CO3 S3**

**PROBLEM STATEMENT**

The stock market is highly dynamic and influenced by various factors, making it difficult for investors to predict future prices accurately. The goal is to develop a simple web-based application that uses historical stock data to predict future prices using machine learning techniques.

**AIM :**

To design and implement a **web application for stock market price prediction** using Python and machine learning.

**OBJECTIVE:**

* To fetch real-time stock data using the yfinance library.
* To train a Linear Regression model on historical data to understand stock price trends.
* To predict the next day’s closing price and display it to the user.
* To visualize stock price trends through interactive graphs and charts using Streamlit.
* To provide an easy-to-use interface for users to analyze and predict stock prices.

**DESCRIPTION :**

This project is a Python-based web application developed using Streamlit that enables users to input any stock symbol (such as AAPL, TCS.NS, or INFY.NS) and view comprehensive stock analysis. It fetches historical stock data through the Yahoo Finance API and presents it visually using interactive graphs that display stock trends and predicted values. The application employs a Linear Regression model from scikit-learn to predict the next trading day’s closing price based on past performance. By learning from one year of historical data, the regression model extrapolates future price trends. The system is built using Python libraries such as pandas, numpy, and matplotlib for efficient data processing, numerical computation, and visualization, providing users with an intuitive and insightful platform for stock price prediction.

**ALGORITHM:**

1. Import required libraries (yfinance, pandas, numpy, streamlit, matplotlib, scikit-learn).
2. Take stock symbol input from the user.
3. Fetch historical stock data using the yfinance.download() function.
4. Preprocess data by adding a “Days” column representing each trading day.
5. Split data into independent variable X (Days) and dependent variable y (Closing Price).
6. Train a Linear Regression model using X and y.
7. Predict the next day’s closing price using the trained model.
8. Display

**PROGRAM :**

import streamlit as st

import yfinance as yf

import pandas as pd

import numpy as np

from sklearn.linear\_model import LinearRegression

import matplotlib.pyplot as plt

st.set\_page\_config(page\_title="Stock Market Predictor", page\_icon="📈", layout="centered")

st.title("📈 Simple Stock Market Prediction Web App")

stock\_symbol = st.text\_input("Enter Stock Symbol (e.g., AAPL, TCS.NS, INFY.NS):", "AAPL")

data = yf.download(stock\_symbol, period="1y")

if isinstance(data.columns, pd.MultiIndex):

    data.columns = data.columns.get\_level\_values(0)

if data.empty:

    st.error("No data found for this stock symbol. Try another one.")

    st.stop()

st.subheader("Recent Stock Data")

st.write(data.tail())

data['Days'] = np.arange(len(data))

X = data[['Days']]

y = data['Close']

model = LinearRegression()

model.fit(X, y)

future\_days = np.arange(len(data), len(data) + 7)

future\_predictions = model.predict(future\_days.reshape(-1, 1))

data['Predicted'] = model.predict(X)

predicted\_next\_day = float(model.predict([[len(data)]])[0])

st.subheader("📊 Prediction Result")

st.write(f"Predicted closing price for \*\*next trading day\*\*: \*\*${predicted\_next\_day:.2f}\*\*")

st.subheader("📉 Historical Closing Price")

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

st.subheader("🔮 Actual vs Predicted Trend")

fig, ax = plt.subplots()

ax.plot(data['Days'], data['Close'], label='Actual Price', color='blue')

ax.plot(data['Days'], data['Predicted'], label='Predicted Trend', color='orange', linestyle='--')

ax.set\_xlabel("Days")

ax.set\_ylabel("Price (USD)")

ax.legend()

st.pyplot(fig)

st.subheader("📆 Next 7 Days Predicted Prices")

future\_df = pd.DataFrame({

    'Day': np.arange(1, 8),

    'Predicted Price': future\_predictions

})

st.write(future\_df)

fig2, ax2 = plt.subplots()

ax2.plot(future\_df['Day'], future\_df['Predicted Price'], marker='o', color='green')

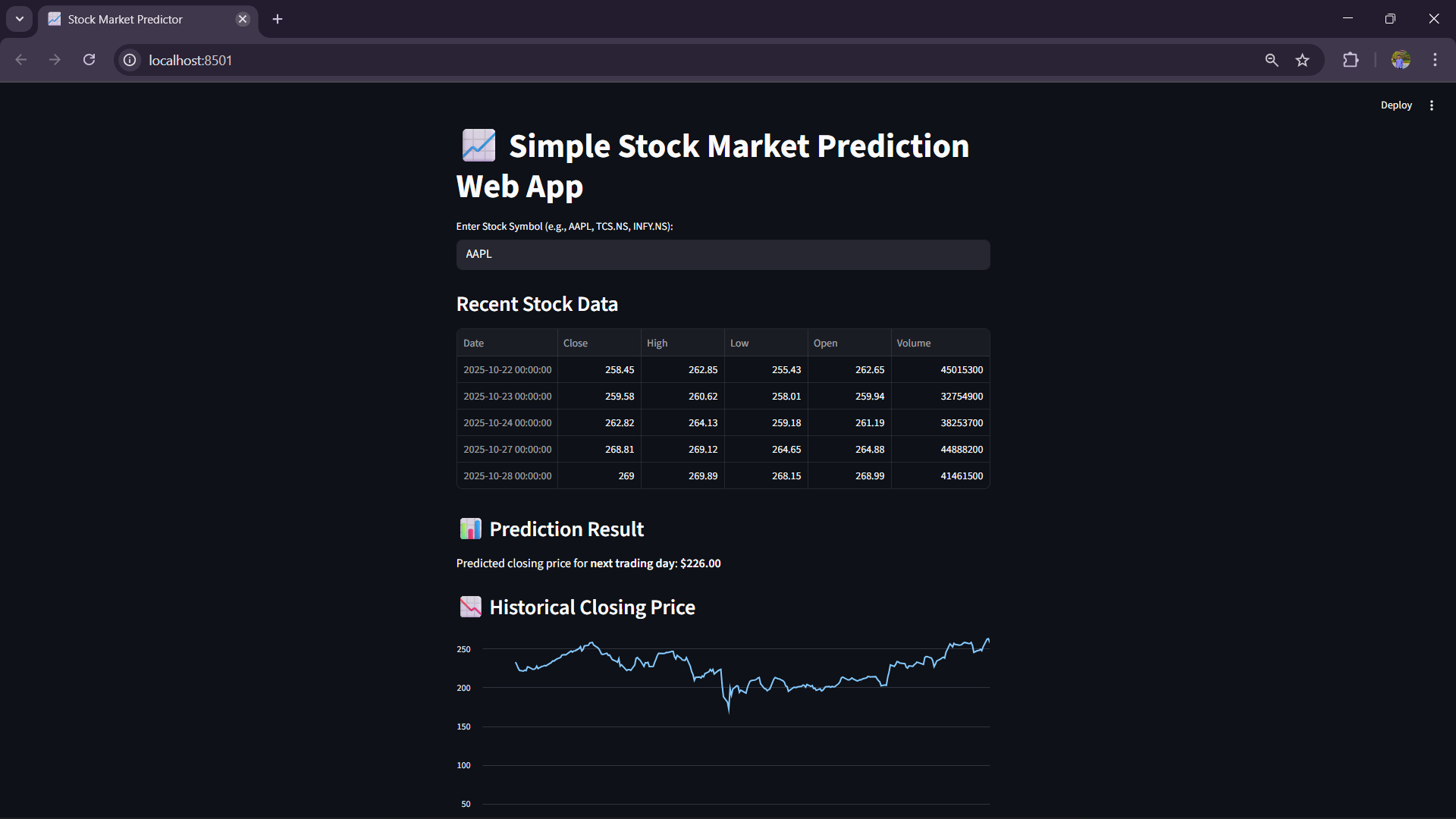
ax2.set\_title("Next 7 Days Predicted Closing Prices")

ax2.set\_xlabel("Days Ahead")

ax2.set\_ylabel("Price (USD)")

st.pyplot(fig2)

**OUTPUT :**



**CONCLUSION:**

Therefore the python program to show the stock markset prediction has been successfully executed.