Exp No: 2 Implement programs for visualizing time series data.

**Aim:** To implement programs for visualizing time series data.

**Objectives:**

The primary objective of this experiment is to implement various visualization techniques for time series data to analyze trends and relationships. This experiment focuses on:

1. Understanding time series data and its visualization techniques.
2. Using Python libraries such as Matplotlib and Seaborn for data visualization.
3. Implementing different types of visualizations, including:
   * 3D bar charts for closing price distribution.
   * Box plots for OHLC (Open, High, Low, Close) price analysis.
   * Histograms to observe price distribution patterns.
   * Scatter plots to analyze trading volume versus closing prices.
   * Heatmaps to explore correlations between different financial features.
4. Interpreting the graphical representations to gain insights into stock price behavior.

**Background/Scope:**

Time series data analysis is essential in finance, stock market predictions, and economic forecasting. The visualization of such data helps in understanding historical trends, identifying anomalies, and detecting correlations between features. In this experiment:

* The dataset used contains historical stock price data of Solana cryptocurrency.
* Different visualization techniques are applied to reveal insights into price fluctuations, volume variations, and overall trends.
* By leveraging these visualization techniques, financial analysts can better comprehend market behaviors, make informed trading decisions, and detect potential investment opportunities.

**Code:**

import pandas as pd

import matplotlib.pyplot as plt import seaborn as sns

import numpy as np

from mpl\_toolkits.mplot3d import Axes3D

file\_path = r"C:\Users\Lenovo\Downloads\coin\_Solana.csv"

df = pd.read\_csv(file\_path)

df['Date'] = pd.to\_datetime(df['Date'])

df = df.sort\_values(by='Date')

**Visualization 1: 3D Bar Chart for Closing Price Distribution**

closing\_price\_ranges = ["Low (<25%)", "Mid (25-75%)", "High (>75%)"]

percentiles = np.percentile(df['Close'], [25, 75]) labels =[]

for price in df['Close']:

if price < percentiles[0]:

labels.append(closing\_price\_ranges[0]) elif price > percentiles[1]:

labels.append(closing\_price\_ranges[2]) else:

labels.append(closing\_price\_ranges[1])

unique\_labels, counts = np.unique(labels, return\_counts=True)

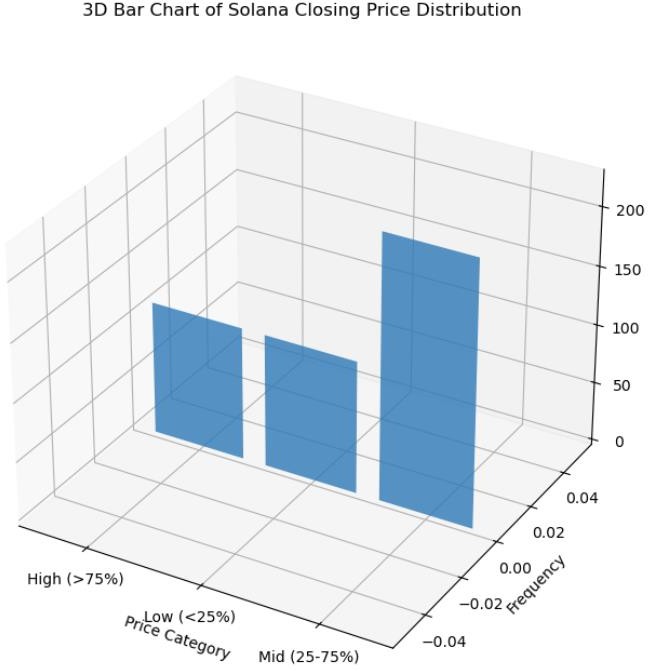
fig = plt.figure(figsize=(8, 8))

ax = fig.add\_subplot(111, projection='3d')

ax.bar(unique\_labels, counts, zs=0, zdir='y', alpha=0.8) ax.set\_xlabel("Price Category") ax.set\_ylabel("Frequency")

ax.set\_zlabel("Count")

ax.set\_title("3D Bar Chart of Solana Closing Price Distribution") plt.show()

**OUTPUT:**

**Visualization 2: Box Plot for OHLC Prices**

plt.figure(figsize=(10, 6))

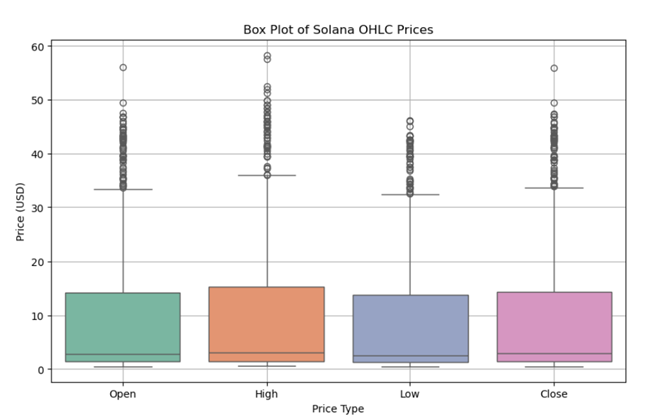
sns.boxplot(data=df[['Open', 'High', 'Low', 'Close']], palette="Set2") plt.xlabel("Price Type")

plt.ylabel("Price (USD)")

plt.title("Box Plot of Solana OHLC Prices") plt.grid()

plt.show()

**OUTPUT:**



**Visualization 3: Histogram of Closing Prices**

plt.figure(figsize=(10, 5)

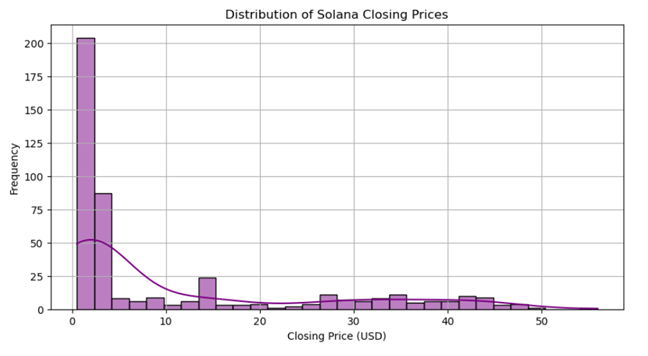
sns.histplot(df['Close'], bins=30, kde=True, color='purple') plt.xlabel("Closing Price (USD)")

plt.ylabel("Frequency")

plt.title("Distribution of Solana Closing Prices") plt.grid()

plt.show()

**OUTPUT:**



**Visualization 4: Scatter Plot of Volume vs. Closing Price** plt.figure(figsize=(10, 5))

plt.scatter(df['Volume'], df['Close'], alpha=0.5, color='red') plt.xlabel("Trading Volume")

plt.ylabel("Closing Price (USD)") plt.title("Trading Volume vs. Closing Price") plt.xscale('log') # Log scale for better visualization plt.grid()

plt.show()

**OUTPUT:**



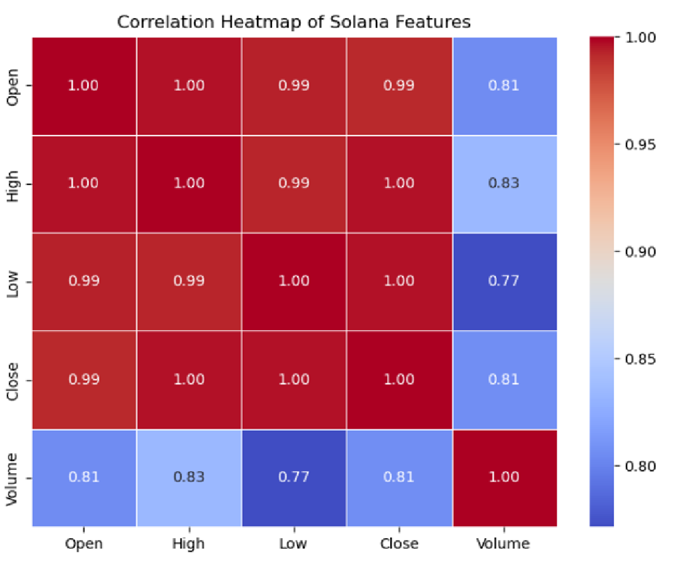
**Visualization 5: Heatmap for Correlation Between Features** plt.figure(figsize=(8, 6))

corr = df[['Open', 'High', 'Low', 'Close', 'Volume']].corr()

sns.heatmap(corr, annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5) plt.title("Correlation Heatmap of Solana Features")

plt.show()

**OUTPUT:**



**RESULT:**

The required output is successfully executed for visualizing the data.