SCHOOL OF COMPUTER SCIENCE ENGINEERING AND INFORMATION SYSTEMS (SCORE)

SWE4002 – CLOUD COMPUTING

FALL SEMESTER 2024 – 2025

SLOT : B2

REVIEW – 03

TITLE : REAL TIME STOCK PRICE PREDICTION

Submitted by :

NAME	REGISTER NO
KEERTHANA L.R	21MIS0014
HARSHITHA S	21MIS0036
NANDHITHA H	21MIS0107
RUTHIKA J	21MIS0359

Under the guidance of :

PROF. ASHA M M

1. PROJECT TITLE : REAL TIME STOCK PRICE PREDICTION

2. ABSTRACT :

The Stock Price Prediction App addresses the growing need for accessible, reliable, and accurate tools to analyze and predict stock price movements in a volatile market. While existing platforms provide basic charting tools and historical data, they often lack advanced analytics and machine learning integration, making them less effective for detailed predictions. Users frequently face challenges such as limited access to real-time insights, a lack of user-friendly interfaces, and the inability to interpret complex technical indicators without expertise. Additionally, current solutions may require expensive subscriptions or significant technical knowledge, creating a barrier for many investors. To solve these issues, this app combines technical indicator visualization and machine learning models to deliver actionable insights in an intuitive interface. It offers real-time stock data and predictive analytics using models like Linear Regression, Random Forest, and XGBoost, helping users make informed trading decisions. Features such as Bollinger Bands, MACD, and RSI allow users to identify trends and patterns effectively. Deployed on AWS, the app ensures scalability and accessibility, catering to a wide audience from beginners to seasoned traders. Results demonstrate improved accuracy in short-term price predictions and enhanced user satisfaction due to its simplicity and functionality. By bridging the gap between traditional charting tools and modern predictive analytics, the app empowers users to navigate the stock market confidently and make data-driven decisions.

3. LITERATURE SURVEY:

Shreya Pawaskar (2022) investigated machine learning techniques for stock price prediction, addressing challenges posed by market volatility. The study implemented Multiple Linear Regression, Polynomial Regression, Decision Tree Regressor, and Random Forest Regressor, using RMSE and R^2 to evaluate performance. Decision Tree Regressor outperformed others with an R^2 score of 1.0. The research highlighted machine learning's benefits in financial forecasting, including improved accuracy, decision-making, and reduced human error. Despite its effectiveness, limitations such as dataset sensitivity and external factors like economic conditions were noted. This study emphasizes the role of machine learning in providing practical, data-driven solutions for stock market predictions [1].

Sidra Mehtab, Jaydip Sen, and Abhishek Dutta (2020) proposed a hybrid framework for stock price prediction using machine learning and deep learning models. Using historical NIFTY 50 index data (December 2014–July 2020), they developed eight machine learning models, including Random Forest and XGBoost, and four LSTM-based deep learning models. A novel walk-forward validation approach optimized LSTM models, with univariate models using one-week prior data achieving the highest accuracy for weekly predictions. Deep learning outperformed machine learning, demonstrating superior predictive capabilities. The study found univariate LSTM models more effective than multivariate ones, emphasizing their strength in capturing temporal dependencies [2].

Payal Soni, et al (2022) provided a systematic review of machine learning approaches in stock price prediction. Their study explored traditional methods such as Random Forest and Support Vector Machines, as well as advanced deep learning techniques like LSTM and CNN. The paper highlighted the role of sentiment analysis and graph-based models in improving prediction accuracy. Results showed that deep learning techniques outperformed traditional methods in capturing complex patterns, while graph-based approaches added insights into relationships among stocks. Challenges include model complexity and limited data scalability, with future scope directed towards integrating sentiment and historical data [3].

Yang and Pan (2021) proposed a novel ensemble deep learning model combining stock prices and news data for stock prediction. Utilizing S&P 500 data, their model blended Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) networks into a two-level ensemble architecture. Their experiments demonstrated significant performance gains, with a 57.55% reduction in mean squared error and improvements in precision, recall, and F1-score compared to existing methods. They emphasized the importance of integrating sentiment analysis and temporal data to enhance prediction accuracy, suggesting that ensemble approaches offer robust solutions for predicting complex financial trends effectively [4].

Nti et al. (2020) conducted a comprehensive evaluation of ensemble learning techniques for stock market prediction using datasets from Ghana, South Africa, the U.S., and India. They compared methods like bagging, boosting, stacking, and blending with Decision Trees, Support Vector Machines, and Neural Networks. Results showed that stacking achieved superior accuracy, followed by blending, while bagging and boosting demonstrated computational efficiency. Stacking and blending excelled on large datasets but required careful selection of base and meta-learners. The study emphasized ensemble methods' effectiveness in regression and classification tasks, revealing performance differences based on dataset origin and ensemble strategy [5].

Shrivastav and Kumar (2022) proposed an ensemble model combining Deep Learning, Gradient Boosting Machine (GBM), and Distributed Random Forest (DRF) for stock price prediction. Using high-frequency data from Coca-Cola stock, their model demonstrated superior performance compared to individual methods, achieving an RMSE of 0.18 and an R^2 of 0.99. The ensemble utilized bagging, boosting, and stacking techniques to leverage the strengths of each model, outperforming standalone models in prediction accuracy. This approach highlights the potential of combining machine learning techniques to analyze non-linear, stochastic datasets, opening new dimensions for big data analysis in stock market forecasting [6].

Yang et al. (2020) proposed a hybrid method combining LSTM with Ensemble Empirical Mode Decomposition (EEMD) for stock price prediction, addressing the complexity and nonlinearity of stock market data. The method first decomposes time-series data into smoother subsequences using EEMD, followed by applying LSTM to predict each sequence. The final prediction is obtained by fusing these individual outputs. The study tested the approach on both synthetic and real-world stock data, such as SP500 and DAX indices, demonstrating superior accuracy compared to traditional models like SVR and KNR. The method emphasizes adaptability and precision in financial forecasting [7].

Ampomah et al. (2020) evaluated tree-based ensemble machine learning models for predicting stock price movement direction. Using eight datasets from NYSE, NASDAQ, and NSE, models like Random Forest, XGBoost, and Extra Trees were assessed based on accuracy, precision, recall, and other metrics. AdaBoost excelled in training accuracy, while Extra Trees performed best on test datasets. Preprocessing included normalization and PCA for dimensionality reduction. The study highlighted the robustness of ensemble methods over single models, with Extra Trees achieving the highest mean performance across metrics. These insights underscore the efficiency of ensemble classifiers in handling financial data's nonlinearity and noise [8].

Nabi et al. (2020) introduced a novel approach for stock price prediction leveraging Gradient Boosting Machine with Feature Engineering (GBM-wFE) and Principal Component Analysis (PCA) for feature selection. The study utilized 11 datasets from Nasdaq and S&P 500, spanning 25 years, to validate their method. By incorporating engineered features, such as high-low price differences and open-close averages, the approach significantly improved classification accuracy. GBM-wFE outperformed existing models, achieving an average accuracy of 99.28% and a mean absolute percentage error (MAPE) of 0.19%. This research pioneers feature engineering for multiclass classification in stock prediction, demonstrating robust results against benchmarks [9].

Justice Kwame Appati et al. (2021) developed an ensemble scheme for stock price prediction, integrating bidirectional Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) models. Their approach, applied to the S&P 500 dataset (1950-2019), achieved remarkable accuracy with a Mean Squared Error (MSE). Previous studies, such as Hossain et al. (2018), explored standalone LSTM and GRU techniques but recorded higher errors. The ensemble model exploits the bidirectional architecture for enhanced data utilization, outperforming traditional deep learning models in forecasting volatile financial trends, demonstrating its robustness in capturing complex temporal dependencies [10].

4. EXISTING SYSTEM :

Existing stock analysis systems primarily offer tools for visualizing historical data and basic technical indicators such as moving averages, RSI, or MACD. Platforms like Yahoo Finance, TradingView, and other broker-provided tools are widely used but are often limited to manual charting and static insights. While some advanced platforms incorporate analytics, they frequently come with expensive subscriptions and are designed for professional traders, making them inaccessible for casual investors. Additionally, these systems often lack integration with machine learning models for accurate short-term predictions, leaving users to rely on their judgment or external analysis. This creates a gap for a user-friendly solution that combines real-time data, advanced predictive analytics, and affordability.

5. PROPOSED SYSTEM :

The proposed system is a user-friendly, web-based application designed to bridge the gap between traditional stock analysis tools and modern predictive technologies. Built using Streamlit, it provides a seamless interface for users to visualize historical stock data and explore advanced technical indicators like Bollinger Bands, RSI, MACD, SMA, and EMA. Unlike existing systems, this app integrates machine learning models such as Linear Regression, Random Forest, Extra Trees, KNeighbors, and XGBoost to deliver accurate short-term stock price predictions. By leveraging real-time data from sources like Yahoo Finance, it ensures up-to-date insights for users. The system is designed for scalability and accessibility, being deployed on AWS, which ensures high performance and global reach. It caters to both beginners and experienced traders by simplifying complex analyses and eliminating the need for expensive subscriptions or specialized expertise. The proposed solution empowers users to make data-driven decisions confidently.

6. BRIEF DESCRIPTION OF THE PROJECT :

The Stock Price Prediction App is a simple and easy-to-use tool for analyzing stock market trends and predicting prices. It helps users view historical stock data, check technical indicators like RSI, MACD, and moving averages, and make short-term predictions using machine learning models such as Linear Regression and Random Forest. The app gets real-time stock data from Yahoo Finance and processes it to provide clear insights. Built with Python and Streamlit, it runs smoothly and is accessible to everyone. Deployed on AWS, it is reliable, fast, and scalable, making it a helpful tool for investors to make smarter decisions.

7. HARDWARE AND SOFTWARE REQUIREMENTS :

Hardware Requirements:

- **Processor:**
  - Minimum: Dual-core processor
- **RAM:**
  - Minimum: 4 GB
  - Recommended: 8 GB or more for smoother performance, especially when working with large datasets.
- **Storage:**
  - Minimum: 10 GB of free disk space

- **Graphics:**
  - Minimum: Integrated graphics for basic usage

<u>**Software Requirements:**</u>

- **Operating System:**
  - Windows, macOS, or Linux (Ubuntu recommended).
- **Python:**
  - Python 3.7 or higher.
- **Streamlit:**
  - Web application framework used for building and deploying the app.
- **Libraries:**
  - **pandas:** For data manipulation and analysis.
  - **yfinance:** To fetch historical stock price data.
  - **ta:** For calculating technical indicators like RSI, MACD, and moving averages.
  - **scikit-learn:** For machine learning models like Linear Regression, Random Forest, etc.
  - **XGBoost:** For advanced boosting algorithms in predictions.
- **AWS (Amazon Web Services):**
  - **AWS Account:** For deploying the app on the cloud (EC2 instance, S3 for data storage, etc.).
- **IDE/Text Editor:**
  - Visual Studio Code, PyCharm, or any Python-compatible IDE for development.
- **Web Browser:**
  - Chrome, Firefox, or any modern browser for accessing the Streamlit app.

8. FEATURES :

- **Visualize Technical Indicators:** Explore various technical indicators such as Bollinger Bands, MACD, RSI, SMA, and EMA to gain insights into stock price trends.
- **Recent Data Display:** View the most recent data of the selected stock, including the last 10 data points.
- **Price Prediction:** Predict future stock prices using machine learning models including Linear Regression, Random Forest Regressor, Extra Trees Regressor, KNeighbors Regressor, and XGBoost Regressor.

9. SETUP :

- Navigate to the project directory:
    - **cd stock-price-prediction-app**
- Install the required Python packages using pip:
    - **pip install -r requirements.txt**

10.USAGE :

1. Run the Streamlit app:

   streamlit run app.py

2. The app will open in your default web browser. Use the sidebar to choose options for visualization, recent data display, or making price predictions.
3. Follow the on-screen instructions to input the stock symbol, select a date range, and choose technical indicators or prediction models.

11. TECHNOLOGIES :

- Python
- Streamlit

- pandas
- yfinance
- ta (Technical Analysis Library)
- scikit-learn
- XGBoost

12. IMPLEMENTATION CODE :

app.py

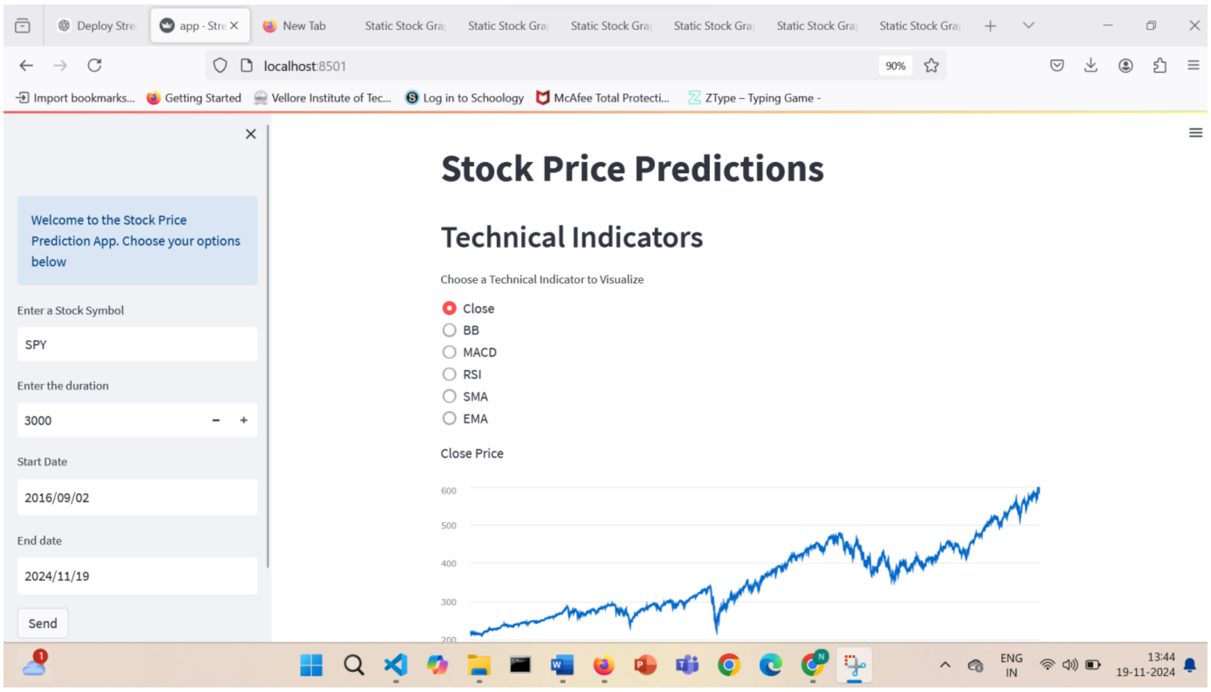
```
import streamlit as st
import pandas as pd
import yfinance as yf
from ta.volatility import BollingerBands
from ta.trend import MACD, EMAIndicator, SMAIndicator
from ta.momentum import RSIIndicator
import datetime
from datetime import date
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsRegressor
from xgboost import XGBRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import ExtraTreesRegressor
from sklearn.metrics import r2_score, mean_absolute_error

st.title('Stock Price Predictions')
st.sidebar.info('Welcome to the Stock Price Prediction App. Choose your options below')
st.sidebar.info("Created and designed by [Vikas Sharma](https://www.linkedin.com/in/vikas-
sharma005/)")

def main():
    option = st.sidebar.selectbox('Make a choice', ['Visualize','Recent Data', 'Predict'])
    if option == 'Visualize':
        tech_indicators()
    elif option == 'Recent Data':
        dataframe()
    else:
        predict()
@st.cache_resource
def download_data(op, start_date, end_date):
    df = yf.download(op, start=start_date, end=end_date, progress=False)
    return
        else:
        st.sidebar.error('Error: End date must fall after start date')
data = download_data(option, start_date, end_date)
scaler = StandardScaler()

def tech_indicators():
    st.header('Technical Indicators')
    option = st.radio('Choose a Technical Indicator to Visualize', ['Close', 'BB', 'MACD', 'RSI',
'SMA', 'EMA'])

    # Bollinger bands
    bb_indicator = BollingerBands(data.Close)
    bb = data
    bb['bb_h'] = bb_indicator.bollinger_hband()
    bb['bb_l'] = bb_indicator.bollinger_lband()
    # Creating a new dataframe
    bb = bb[['Close', 'bb_h', 'bb_l']]
    # MACD
    macd = MACD(data.Close).macd()
    # RSI
    rsi = RSIIndicator(data.Close).rsi()
    # SMA
    sma = SMAIndicator(data.Close, window=14).sma_indicator()
    # EMA
    ema = EMAIndicator(data.Close).ema_indicator()

    if option == 'Close':
        st.write('Close Price')
        st.line_chart(data.Close)
    elif option == 'BB':
        st.write('BollingerBands')
        st.line_chart(bb)
    elif option == 'MACD':
        st.write('Moving Average Convergence Divergence')
        st.line_chart(macd)
    elif option == 'RSI':
        st.write('Relative Strength Indicator')
        st.line_chart(rsi)
    elif option == 'SMA':
        st.write('Simple Moving Average')
        st.line_chart(sma)
    else:
        st.write('Expoenetial Moving Average')
        st.line_chart(ema)

def dataframe():
    st.header('Recent Data')
    st.dataframe(data.tail(10))

def predict():
    model = st.radio('Choose a model', ['LinearRegression', 'RandomForestRegressor',
'ExtraTreesRegressor', 'KNeighborsRegressor', 'XGBoostRegressor'])
    num = st.number_input('How many days forecast?', value=5)
    num = int(num)
    if st.button('Predict'):
        if model == 'LinearRegression':
            engine = LinearRegression()
            model_engine(engine, num)
        elif model == 'RandomForestRegressor':
            engine = RandomForestRegressor()
            model_engine(engine, num)
        elif model == 'ExtraTreesRegressor':
            engine = ExtraTreesRegressor()
            model_engine(engine, num)
        elif model == 'KNeighborsRegressor':
            engine = KNeighborsRegressor()
            model_engine(engine, num)
        else:
            engine = XGBRegressor()
            model_engine(engine, num)
def model_engine(model, num):
    # getting only the closing price
```

stock_market_prediction.ipynb

```
!pip install streamlit pyngrok yfinance scikit-learn ta xgboost

from pyngrok import ngrok
# Replace "YOUR_AUTH_TOKEN" with your actual ngrok auth token
ngrok.set_auth_token("YOUR_AUTH_TOKEN")
# Connect to the existing ngrok tunnel using its ID
public_url = ngrok.connect(8501, "YOUR_AUTH_TOKEN")
print(f"Streamlit app is running on {public_url}")

with open('app.py', 'w') as f:
    f.write("""
import streamlit as st
import pandas as pd
import yfinance as yf
from ta.volatility import BollingerBands
from ta.trend import MACD, EMAIndicator, SMAIndicator
from ta.momentum import RSIIndicator
import datetime
from datetime import date
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsRegressor
from xgboost import XGBRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import ExtraTreesRegressor
from sklearn.metrics import r2_score, mean_absolute_error

st.title('Stock Price Predictions')
st.sidebar.info('Welcome to the Stock Price Prediction App. Choose your options below')
st.sidebar.info("Created and designed by [Vikas Sharma](https://www.linkedin.com/in/vikas-
sharma005/)")
def main():
    option = st.sidebar.selectbox('Make a choice', ['Visualize','Recent Data', 'Predict'])
    if option == 'Visualize':
        tech_indicators()
    elif option == 'Recent Data':
        dataframe()
    else:
        predict()
@st.cache_resource
def download_data(op, start_date, end_date):
    df = yf.download(op, start=start_date, end=end_date, progress=False)
    return df

option = st.sidebar.text_input('Enter a Stock Symbol', value='SPY')
option = option.upper()
today = datetime.date.today()
duration = st.sidebar.number_input('Enter the duration', value=3000)
before = today - datetime.timedelta(days=duration)
```

```
start_date = st.sidebar.date_input('Start Date', value=before)
end_date = st.sidebar.date_input('End date', today)
if st.sidebar.button('Send'):
    if start_date < end_date:
        st.sidebar.success(f'Start date: {start_date}\n\nEnd date: {end_date}')
        download_data(option, start_date, end_date)
    else:
        st.sidebar.error('Error: End date must fall after start date')

data = download_data(option, start_date, end_date)
scaler = StandardScaler()
def tech_indicators():
    st.header('Technical Indicators')
    option = st.radio('Choose a Technical Indicator to Visualize', ['Close', 'BB', 'MACD', 'RSI',
'SMA', 'EMA'])

    # Bollinger bands
    bb_indicator = BollingerBands(data.Close)
    bb = data
    bb['bb_h'] = bb_indicator.bollinger_hband()
    bb['bb_l'] = bb_indicator.bollinger_lband()
    # Creating a new dataframe
    bb = bb[['Close', 'bb_h', 'bb_l']]
    # MACD
    macd = MACD(data.Close).macd()
    # RSI
    rsi = RSIIndicator(data.Close).rsi()
    # SMA
    sma = SMAIndicator(data.Close, window=14).sma_indicator()
    # EMA
    ema = EMAIndicator(data.Close).ema_indicator()

    if option == 'Close':
        st.write('Close Price')
        st.line_chart(data.Close)
    elif option == 'BB':
        st.write('BollingerBands')
        st.line_chart(bb)
    elif option == 'MACD':
        st.write('Moving Average Convergence Divergence')
        st.line_chart(macd)
    elif option == 'RSI':
        st.write('Relative Strength Indicator')
        st.line_chart(rsi)
    elif option == 'SMA':
        st.write('Simple Moving Average')
        st.line_chart(sma)
    else:
        st.write('Exponential Moving Average')
        st.line_chart(ema)
def dataframe():
    st.header('Recent Data')
    st.dataframe(data.tail(10))
def predict():
    model = st.radio('Choose a model', ['LinearRegression', 'RandomForestRegressor',
'ExtraTreesRegressor', 'KNeighborsRegressor', 'XGBoostRegressor'])
    num = st.number_input('How many days forecast?', value=5)
    num = int(num)
    if st.button('Predict'):
        if model == 'LinearRegression':
            engine = LinearRegression()
            model_engine(engine, num)
        elif model == 'RandomForestRegressor':
            engine = RandomForestRegressor()
            model_engine(engine, num)
        elif model == 'ExtraTreesRegressor':
            engine = ExtraTreesRegressor()
            model_engine(engine, num)
        elif model == 'KNeighborsRegressor':
            engine = KNeighborsRegressor()
            model_engine(engine, num)
        else:
            engine = XGBRegressor()
            model_engine(engine, num)

def model_engine(model, num):
    # getting only the closing price
    df = data[['Close']]
    # shifting the closing price based on number of days forecast
    df['preds'] = data.Close.shift(-num)
    # scaling the data
    x = df.drop(['preds'], axis=1).values
    x = scaler.fit_transform(x)
    # storing the last num_days data
    x_forecast = x[-num:]
    # selecting the required values for training
    x = x[:-num]
    # getting the preds column
    y = df.preds.values
    # selecting the required values for training
    y = y[:-num]

    #spliting the data
    x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=.2, random_state=7)
    # training the model
    model.fit(x_train, y_train)
    preds = model.predict(x_test)
    st.text(f'r2_score: {r2_score(y_test, preds)} \nMAE: {mean_absolute_error(y_test, preds)}')
    # predicting stock price based on the number of days
    forecast_pred = model.predict(x_forecast)
    day = 1
    for i in forecast_pred:
        st.text(f'Day {day}: {i}')
        day += 1

if __name__ == '__main__':
    main()
    """)
print("Streamlit app code has been written to app.py")

import subprocess
# Run the Streamlit app in the background
process = subprocess.Popen(['streamlit', 'run', 'app.py', '--server.port', '8501'])

# Display the public URL
public_url = ngrok.connect(8501)
print(f"Streamlit app is running on {public_url}")
```

OUTPUT SCREENSHOTS :



Welcome to the Stock Price Prediction App. Choose your options below

Enter a Stock Symbol: SPY

Enter the duration: 3000

Start Date: 2016/09/02

End date: 2024/11/19

Send

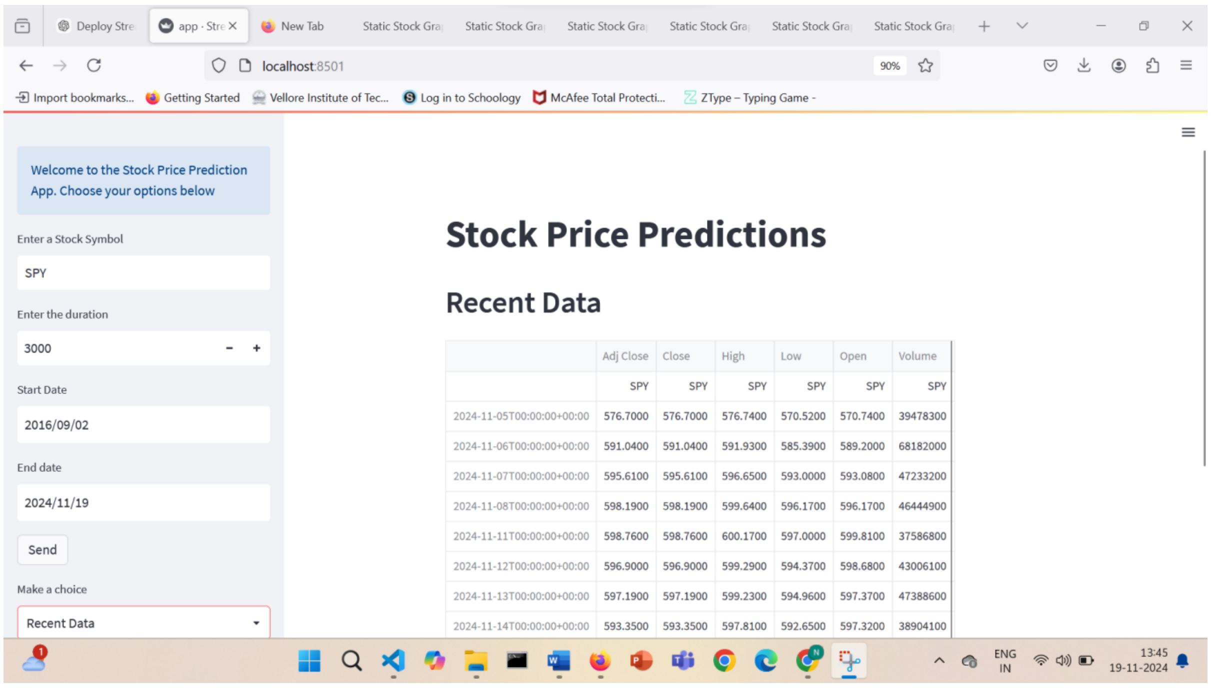
Stock Price Predictions

Technical Indicators

Choose a Technical Indicator to Visualize

[✓ Close]
BB
MACD
RSI
SMA
EMA

Close Price


Welcome to the Stock Price Prediction App. Choose your options below

Enter a Stock Symbol: SPY

Enter the duration: 3000

Start Date: 2016/09/02

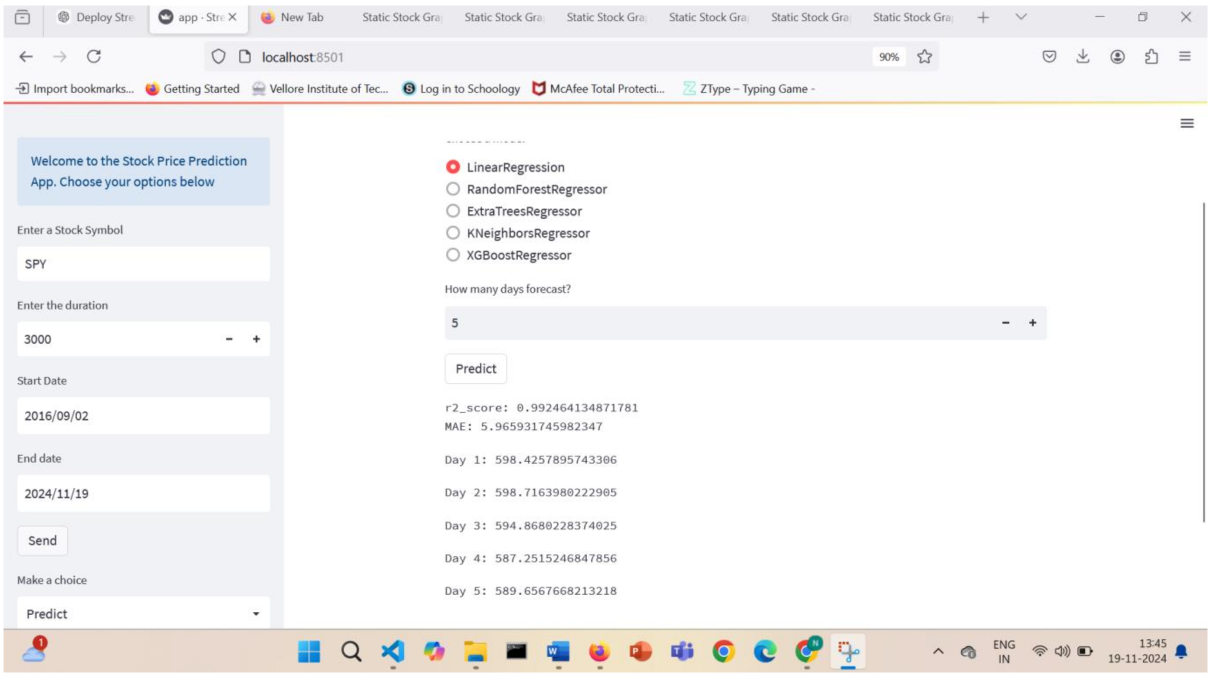
End date: 2024/11/19

Make a choice: Recent Data

Stock Price Predictions

Recent Data

	Adj Close	Close	High	Low	Open	Volume
	SPY	SPY	SPY	SPY	SPY	SPY
2024-11-05T00:00:00+00:00	576.7000	576.7000	576.7400	570.5200	570.7400	39478300
2024-11-06T00:00:00+00:00	591.0400	591.0400	591.9300	585.3900	589.2000	68182000
2024-11-07T00:00:00+00:00	595.6100	595.6100	596.6500	593.0000	593.0800	47233200
2024-11-08T00:00:00+00:00	598.1900	598.1900	599.6400	596.1700	596.1700	46444900
2024-11-11T00:00:00+00:00	598.7600	598.7600	600.1700	597.0000	599.8100	37586800
2024-11-12T00:00:00+00:00	596.9000	596.9000	599.2900	594.3700	598.6800	43006100
2024-11-13T00:00:00+00:00	597.1900	597.1900	599.2300	594.9600	597.3700	47388600
2024-11-14T00:00:00+00:00	593.3500	593.3500	597.8100	592.6500	597.3200	38904100


Welcome to the Stock Price Prediction App. Choose your options below

Enter a Stock Symbol: SPY

Enter the duration: 3000

Start Date: 2016/09/02

End date: 2024/11/19

Make a choice: Predict

[✓ LinearRegression] RandomForestRegressor ExtraTreesRegressor KNeighborsRegressor XGBoostRegressor

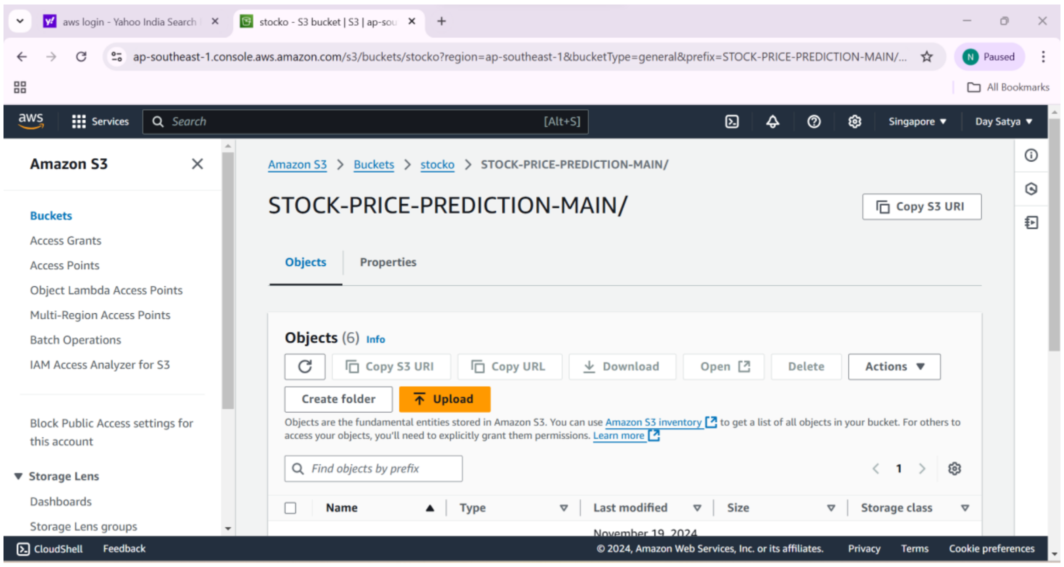
How many days forecast? 5

```
r2_score: 0.992464134871781
MAE: 5.965931745982347

Day 1: 598.4257895743306

Day 2: 598.7163980222905

Day 3: 594.8680228374025

Day 4: 587.2515246847856

Day 5: 589.6567668213218
```

DEPLOYMENT IN AWS :

INSTANCE USED : S3 BUCKET

Amazon S3 (Simple Storage Service) is a scalable, secure, and high-performance object storage service provided by AWS. It allows users to store, retrieve, and manage data as objects in "buckets," which act as containers. S3 is designed for flexibility, supporting a wide range of use cases such as data backups, hosting static websites, big data analytics, and application data storage.



aws login - Yahoo India Search | stocko - S3 bucket | S3 | ap-sou

ap-southeast-1.console.aws.amazon.com/s3/buckets/stocko?region=ap-southeast-1&bucketType=general&prefix=STOCK-PRICE-PREDICTION-MAIN/...

Services | Search [Alt+S] | Singapore | Day Satya

Amazon S3

Buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3

Block Public Access settings for this account

Storage Lens
Dashboards
Storage Lens groups

Amazon S3 > Buckets > stocko > STOCK-PRICE-PREDICTION-MAIN/

STOCK-PRICE-PREDICTION-MAIN/

Copy S3 URI

Objects | Properties

Objects (6) Info

Copy S3 URI | Copy URL | Download | Open | Delete | Actions

Create folder | Upload

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

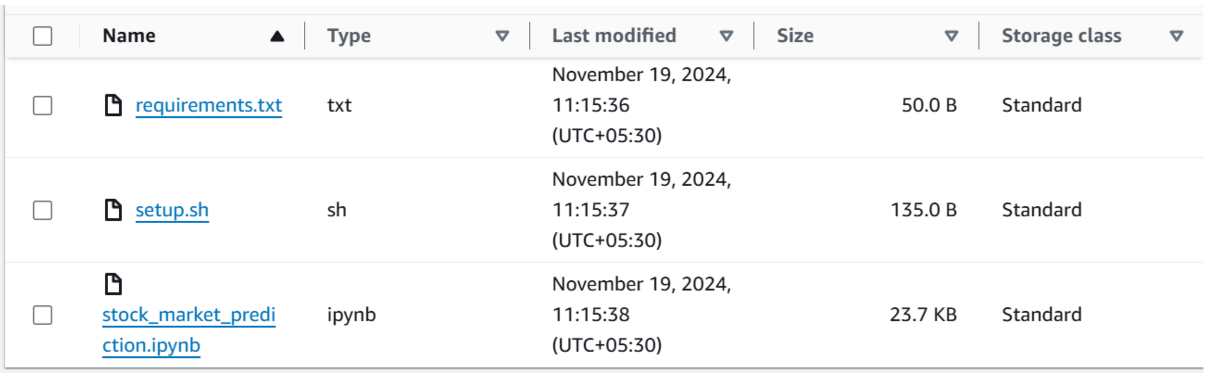
Find objects by prefix

Name | Type | Last modified | Size | Storage class

November 19, 2024

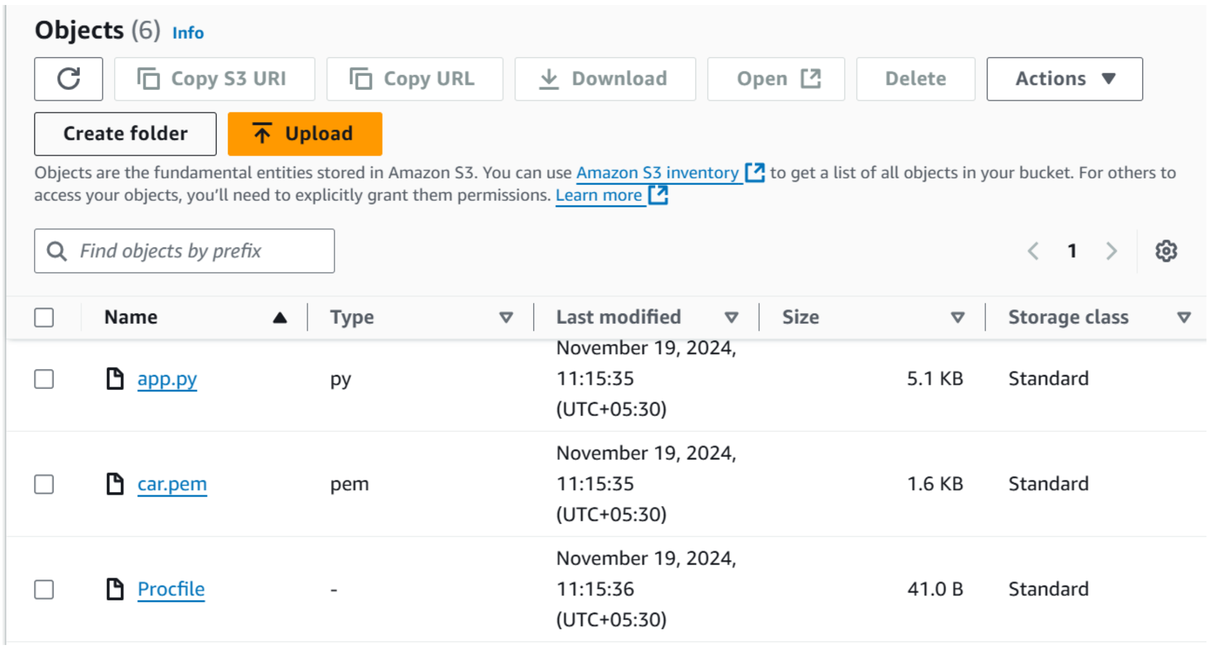
CloudShell | Feedback | © 2024, Amazon Web Services, Inc. or its affiliates. | Privacy | Terms | Cookie preferences

OBJECTS MODULE :



Name	Type	Last modified	Size	Storage class
requirements.txt	txt	November 19, 2024, 11:15:36 (UTC+05:30)	50.0 B	Standard
setup.sh	sh	November 19, 2024, 11:15:37 (UTC+05:30)	135.0 B	Standard
stock_market_prediction.ipynb	ipynb	November 19, 2024, 11:15:38 (UTC+05:30)	23.7 KB	Standard

PEM KEY:

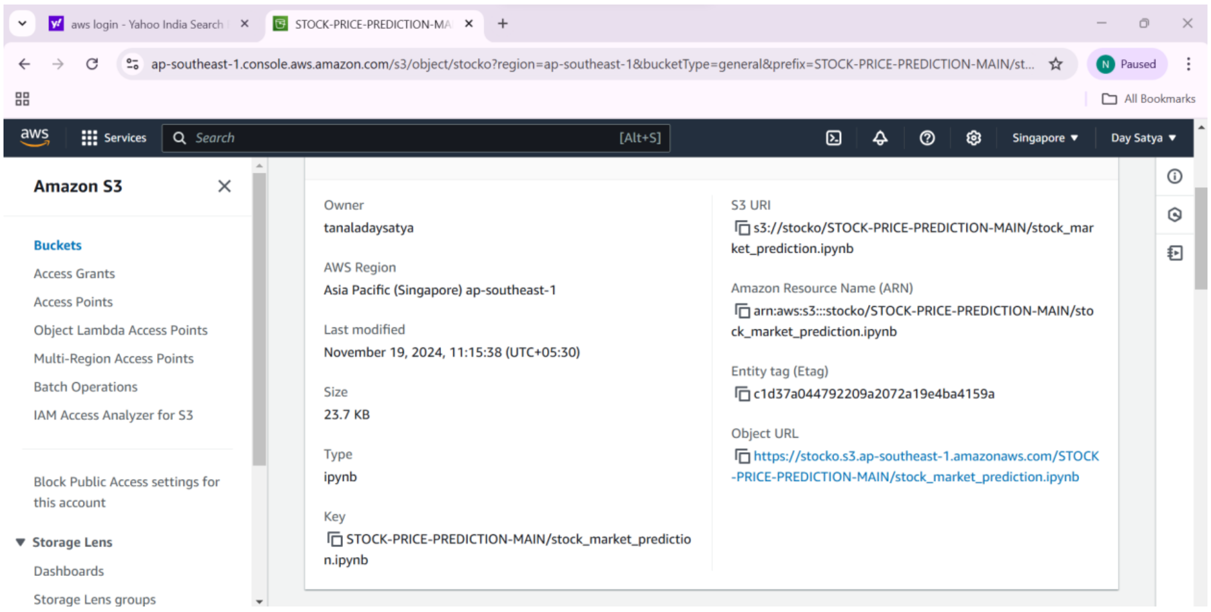


Objects (6) Info

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
app.py	py	November 19, 2024, 11:15:35 (UTC+05:30)	5.1 KB	Standard
car.pem	pem	November 19, 2024, 11:15:35 (UTC+05:30)	1.6 KB	Standard
Procfile	-	November 19, 2024, 11:15:36 (UTC+05:30)	41.0 B	Standard

OBJECTS INFO:



ap-southeast-1.console.aws.amazon.com/s3/object/stocko?region=ap-southeast-1&bucketType=general&prefix=STOCK-PRICE-PREDICTION-MAIN/st...

aws login - Yahoo India Search
STOCK-PRICE-PREDICTION-MA
Paused
All Bookmarks
aws
Services
Search
[Alt+S]
Singapore
Day Satya
Amazon S3
Buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups

Owner
tanaladaysatya

AWS Region
Asia Pacific (Singapore) ap-southeast-1

Last modified
November 19, 2024, 11:15:38 (UTC+05:30)

Size
23.7 KB

Type
ipynb

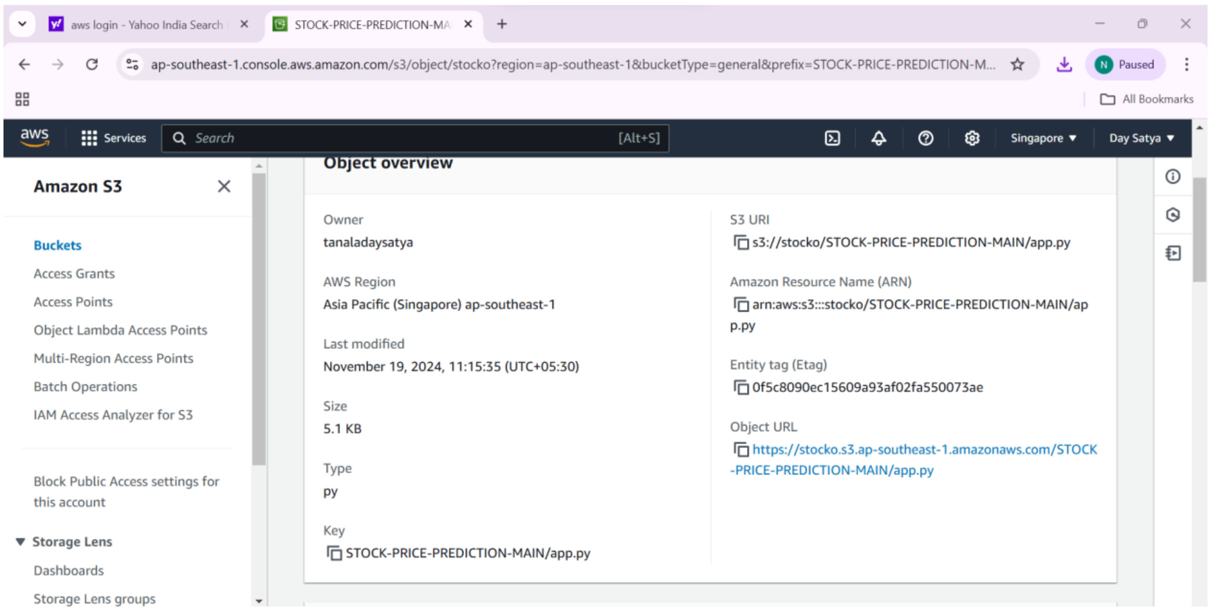
Key
STOCK-PRICE-PREDICTION-MAIN/stock_market_prediction.ipynb

S3 URI
s3://stocko/STOCK-PRICE-PREDICTION-MAIN/stock_market_prediction.ipynb

Amazon Resource Name (ARN)
arn:aws:s3:::stocko/STOCK-PRICE-PREDICTION-MAIN/stock_market_prediction.ipynb

Entity tag (Etag)
c1d37a044792209a2072a19e4ba4159a

Object URL
https://stocko.s3.ap-southeast-1.amazonaws.com/STOCK-PRICE-PREDICTION-MAIN/stock_market_prediction.ipynb


aws login - Yahoo India Search
STOCK-PRICE-PREDICTION-MA
ap-southeast-1.console.aws.amazon.com/s3/object/stocko?region=ap-southeast-1&bucketType=general&prefix=STOCK-PRICE-PREDICTION-M...
Paused
All Bookmarks
Services
Search
[Alt+S]
Singapore
Day Satya
Amazon S3
Buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups

Object overview

Owner
tanaladaysatya

AWS Region
Asia Pacific (Singapore) ap-southeast-1

Last modified
November 19, 2024, 11:15:35 (UTC+05:30)

Size
5.1 KB

Type
py

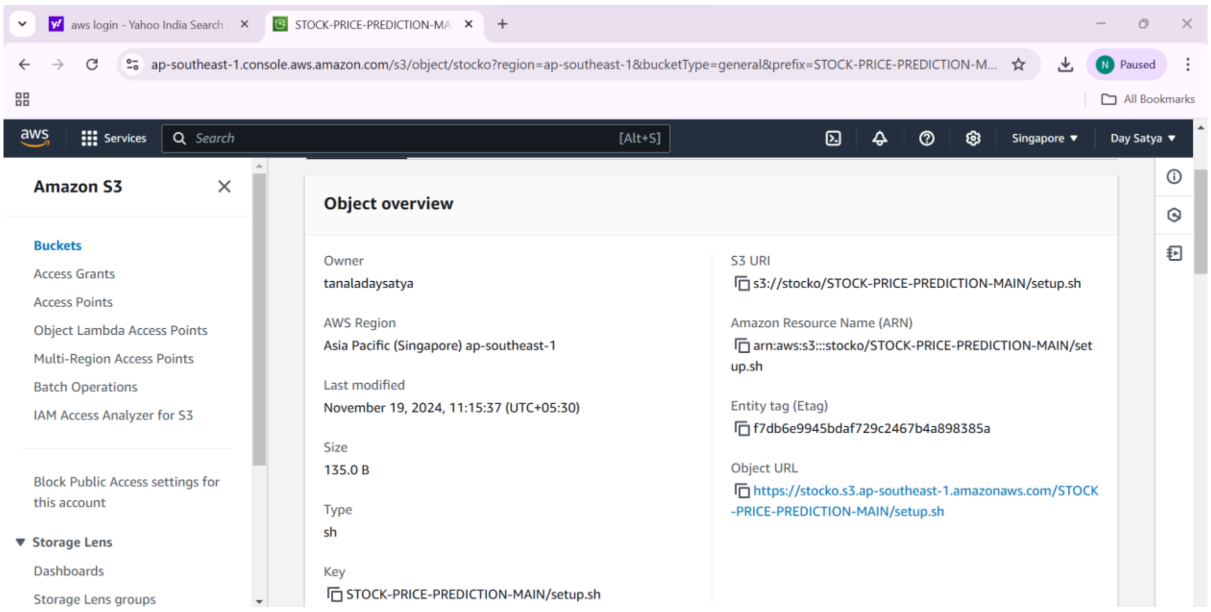
Key
STOCK-PRICE-PREDICTION-MAIN/app.py

S3 URI
s3://stocko/STOCK-PRICE-PREDICTION-MAIN/app.py

Amazon Resource Name (ARN)
arn:aws:s3:::stocko/STOCK-PRICE-PREDICTION-MAIN/app.py

Entity tag (Etag)
0f5c8090ec15609a93af02fa550073ae

Object URL
https://stocko.s3.ap-southeast-1.amazonaws.com/STOCK-PRICE-PREDICTION-MAIN/app.py


aws login - Yahoo India Search
STOCK-PRICE-PREDICTION-MA
ap-southeast-1.console.aws.amazon.com/s3/object/stocko?region=ap-southeast-1&bucketType=general&prefix=STOCK-PRICE-PREDICTION-M...
Paused
All Bookmarks
Services
Search
[Alt+S]
Singapore
Day Satya
Amazon S3
Buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups

Object overview

Owner
tanaladaysatya

AWS Region
Asia Pacific (Singapore) ap-southeast-1

Last modified
November 19, 2024, 11:15:37 (UTC+05:30)

Size
135.0 B

Type
sh

Key
STOCK-PRICE-PREDICTION-MAIN/setup.sh

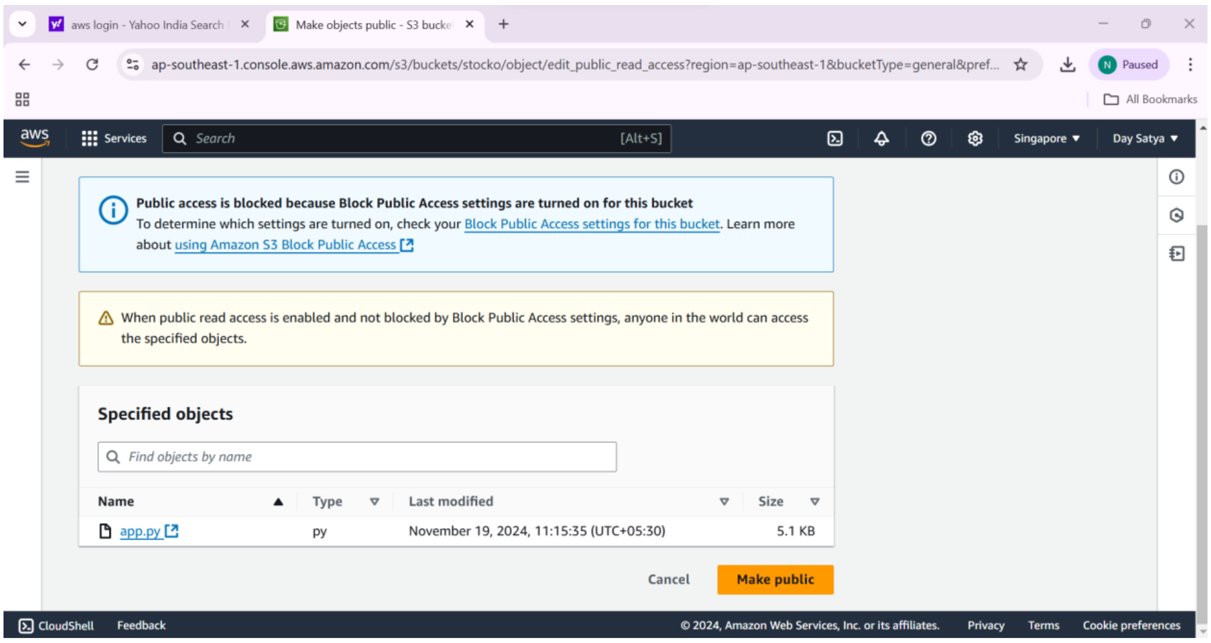
S3 URI
s3://stocko/STOCK-PRICE-PREDICTION-MAIN/setup.sh

Amazon Resource Name (ARN)
arn:aws:s3:::stocko/STOCK-PRICE-PREDICTION-MAIN/setup.sh

Entity tag (Etag)
f7db6e9945bdaf729c2467b4a898385a

Object URL
https://stocko.s3.ap-southeast-1.amazonaws.com/STOCK-PRICE-PREDICTION-MAIN/setup.sh

ACCESS TO PUBLIC:


aws login - Yahoo India Search
Make objects public - S3 bucke
ap-southeast-1.console.aws.amazon.com/s3/buckets/stocko/object/edit_public_read_access?region=ap-southeast-1&bucketType=general&pref...
Paused
All Bookmarks
Services
Search
[Alt+S]
Singapore
Day Satya
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Public access is blocked because Block Public Access settings are turned on for this bucket
To determine which settings are turned on, check your Block Public Access settings for this bucket. Learn more about using Amazon S3 Block Public Access

When public read access is enabled and not blocked by Block Public Access settings, anyone in the world can access the specified objects.

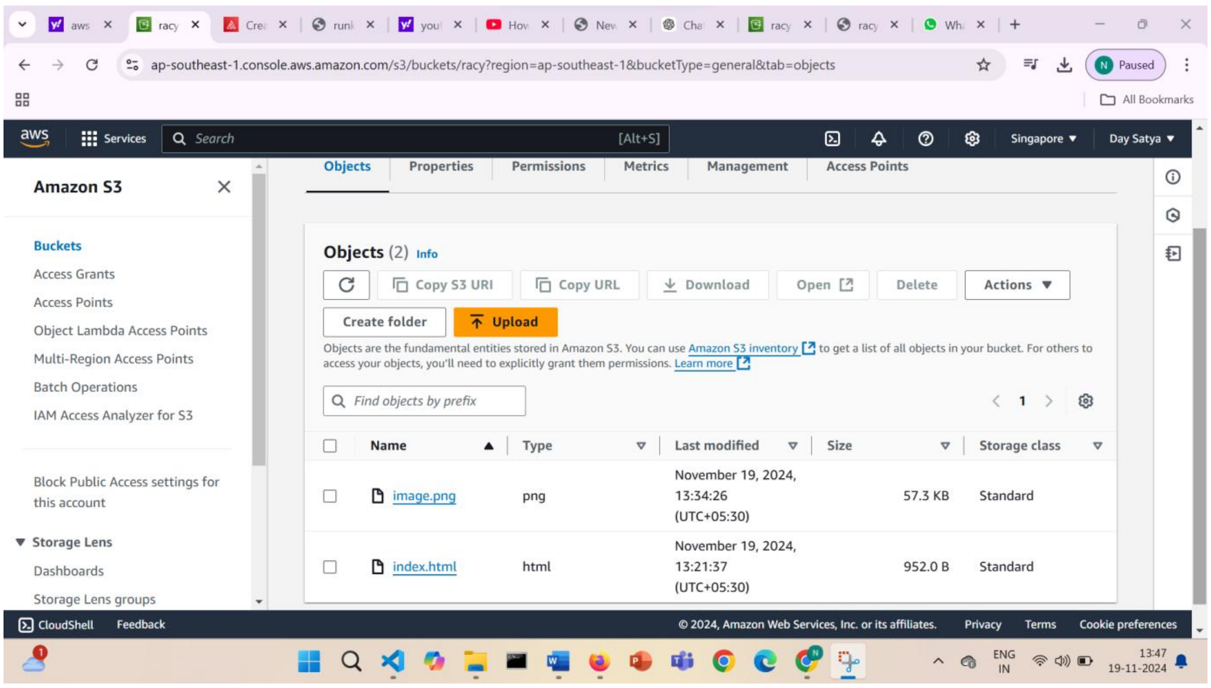
Specified objects

Find objects by name

Name	Type	Last modified	Size
app.py	py	November 19, 2024, 11:15:35 (UTC+05:30)	5.1 KB

Cancel | Make public

CREATED HTML FILE TO HOST :

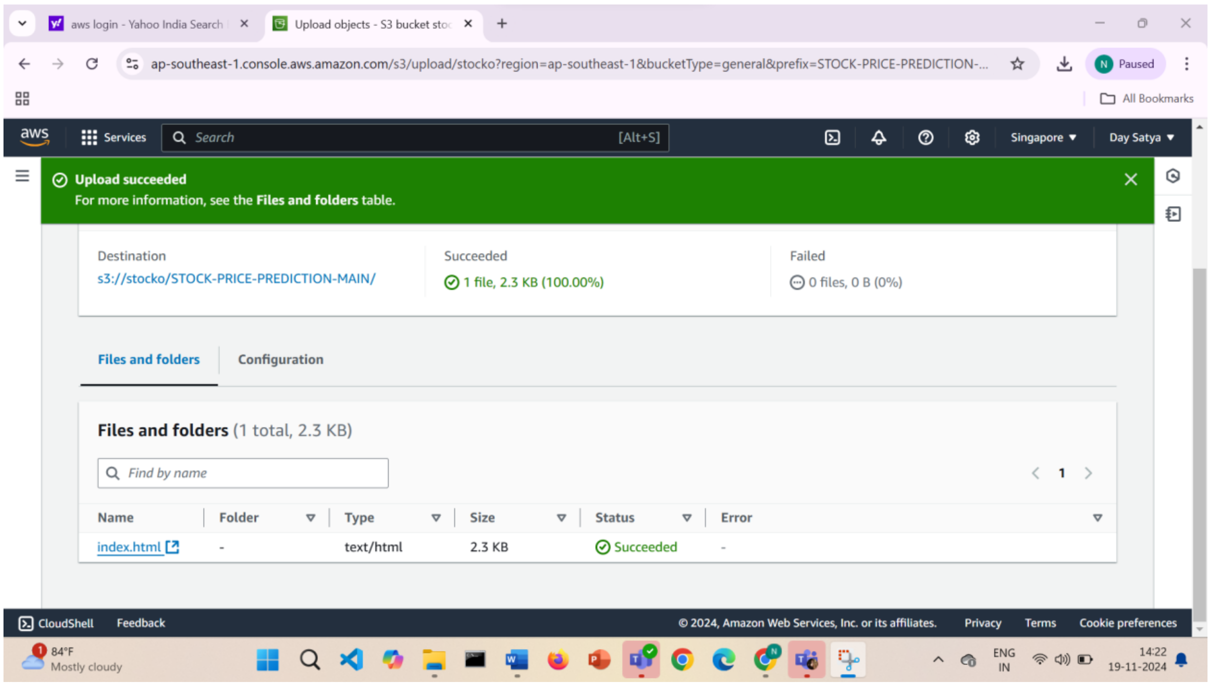


Objects (2) Info

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
image.png	png	November 19, 2024, 13:34:26 (UTC+05:30)	57.3 KB	Standard
index.html	html	November 19, 2024, 13:21:37 (UTC+05:30)	952.0 B	Standard

STATIC HOST:


Upload succeeded
For more information, see the Files and folders table.

Destination
s3://stocko/STOCK-PRICE-PREDICTION-MAIN/

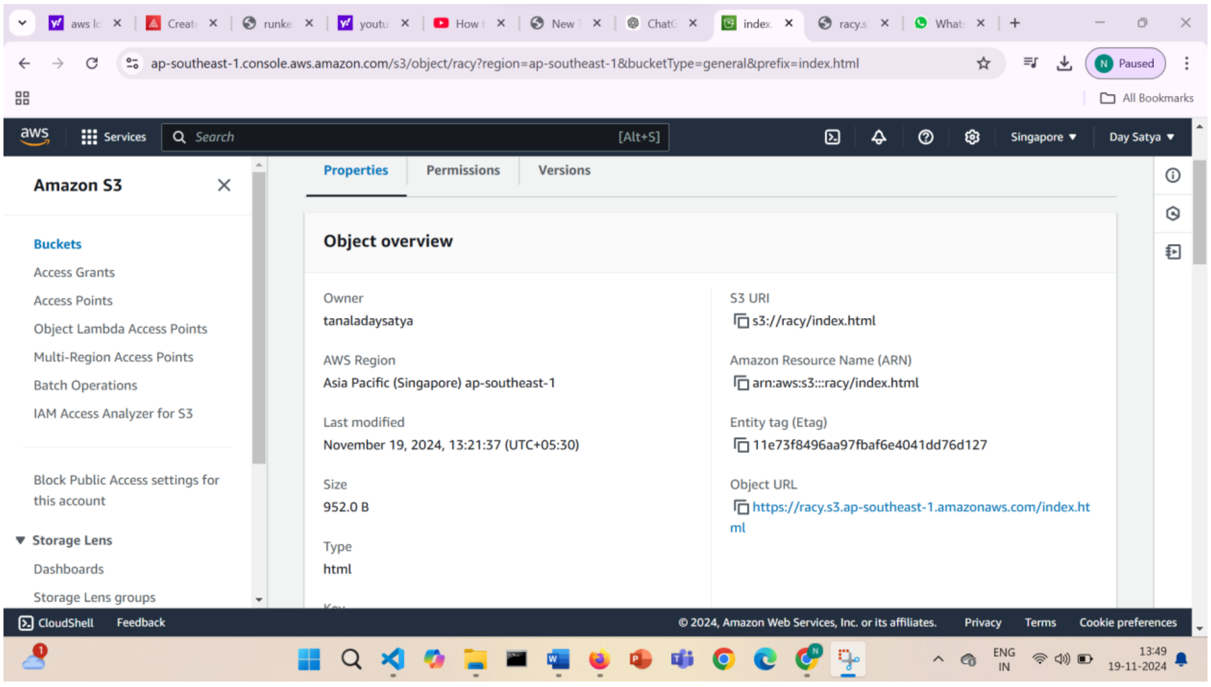
Succeeded
1 file, 2.3 KB (100.00%)

Failed
0 files, 0 B (0%)

Files and folders (1 total, 2.3 KB)

Name	Folder	Type	Size	Status	Error
index.html	-	text/html	2.3 KB	Succeeded	-

DYNAMIC HOST(Public Url)


Object overview

Owner
tanaladaysatya

AWS Region
Asia Pacific (Singapore) ap-southeast-1

Last modified
November 19, 2024, 13:21:37 (UTC+05:30)

Size
952.0 B

Type
html

S3 URI
s3://racy/index.html

Amazon Resource Name (ARN)
arn:aws:s3:::racy/index.html

Entity tag (Etag)
11e73f8496aa97fbaf6e4041dd76d127

Object URL
https://racy.s3.ap-southeast-1.amazonaws.com/index.html

REFERENCES

1. Pawaskar, S. (2022). Stock price prediction using machine learning algorithms. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, *10*.
2. Mehtab, S., Sen, J., & Dutta, A. (2021). Stock price prediction using machine learning and LSTM-based deep learning models. In *Machine Learning and Metaheuristics Algorithms, and Applications: Second Symposium, SoMMA 2020, Chennai, India, October 14–17, 2020, Revised Selected Papers 2* (pp. 88-106). Springer Singapore.
3. Soni, P., Tewari, Y., & Krishnan, D. (2022). Machine learning approaches in stock price prediction: a systematic review. In *Journal of Physics: Conference Series* (Vol. 2161, No. 1, p. 012065). IOP Publishing.
4. Li, Y., & Pan, Y. (2022). A novel ensemble deep learning model for stock prediction based on stock prices and news. *International Journal of Data Science and Analytics*, *13*(2), 139-149.
5. Nti, I. K., Adekoya, A. F., & Weyori, B. A. (2020). A comprehensive evaluation of ensemble learning for stock-market prediction. *Journal of Big Data*, *7*(1), 20.
6. Shrivastav, L. K., & Kumar, R. (2022). An ensemble of random forest gradient boosting machine and deep learning methods for stock price prediction. *Journal of Information Technology Research (JITR)*, *15*(1), 1-19.
7. Yujun, Y., Yimei, Y., & Jianhua, X. (2020). A hybrid prediction method for stock price using LSTM and ensemble EMD. *Complexity*, *2020*(1), 6431712.
8. Ampomah, E. K., Qin, Z., & Nyame, G. (2020). Evaluation of tree-based ensemble machine learning models in predicting stock price direction of movement. *Information*, *11*(6), 332.
9. Nabi, R. M., Soran Ab M, S., & Harron, H. (2020). A novel approach for stock price prediction using gradient boosting machine with feature engineering (gbm-wfe). *Kurdistan Journal of Applied Research*, *5*(1), 28-48.
10. Appati, J. K., Denwar, I. W., Owusu, E., & Soli, M. A. T. (2021). Construction of an ensemble scheme for stock price prediction using deep learning techniques. *International Journal of Intelligent Information Technologies (IJIIT)*, *17*(2), 1-24.