Python Project 1:
Detect Stock Buying/Selling Signals In Real Time

Tan Sang Dao
June 17, 2020

In the last 3 months until now (16/3 – 16/6/2020), Netflix (NFLX) price has increased by more than 50%.

If NFLX is in your stock portfolio, how to make buying/selling decision effectively?

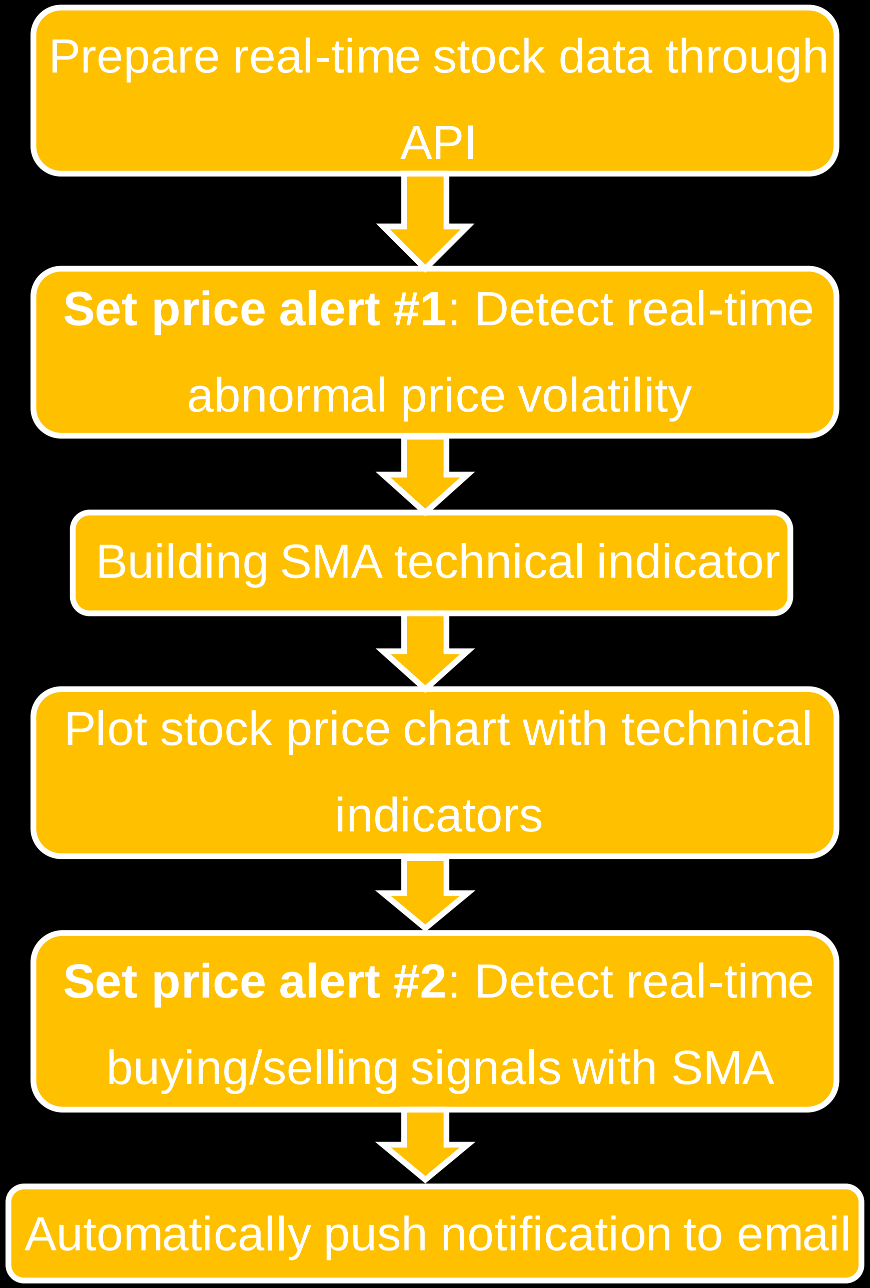
Outline

Requirement:

Design an automatic stock tracking program, which sends alert notification when the price is highly unstable or reaches potential entry/exit points

Data & API:

Program uses stock market data from Yahoo Finance & API of Alpha Vantage



Prepare real-time stock data through API

Set price alert #1: Detect real-time abnormal price volatility

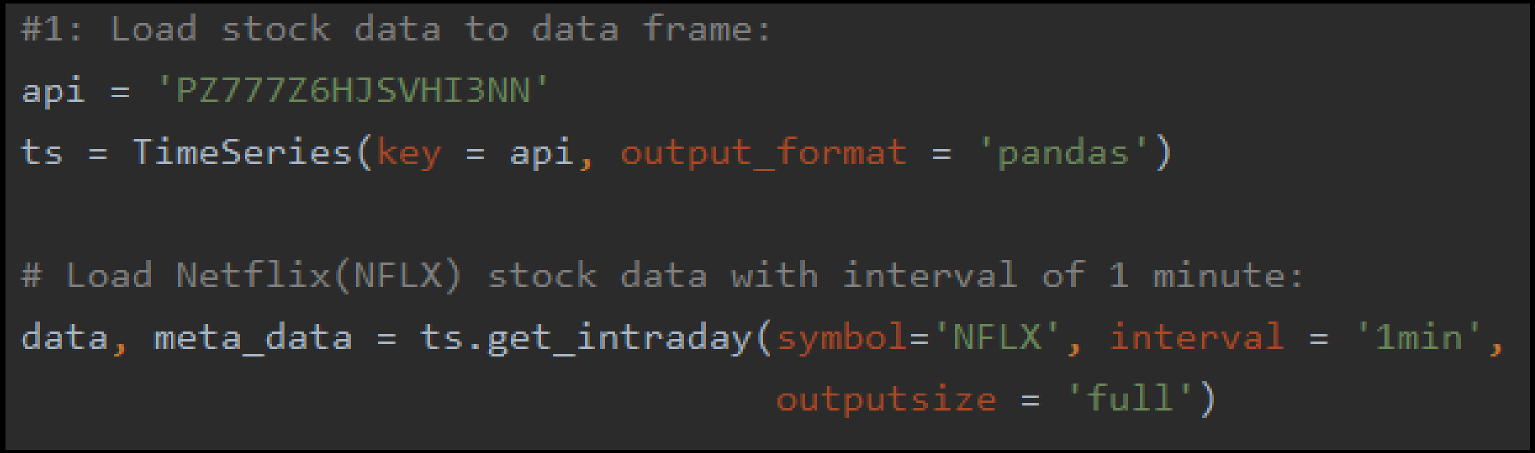
Building SMA technical indicator

Plot stock price chart with technical indicators

Set price alert #2: Detect real-time buying/selling signals with SMA

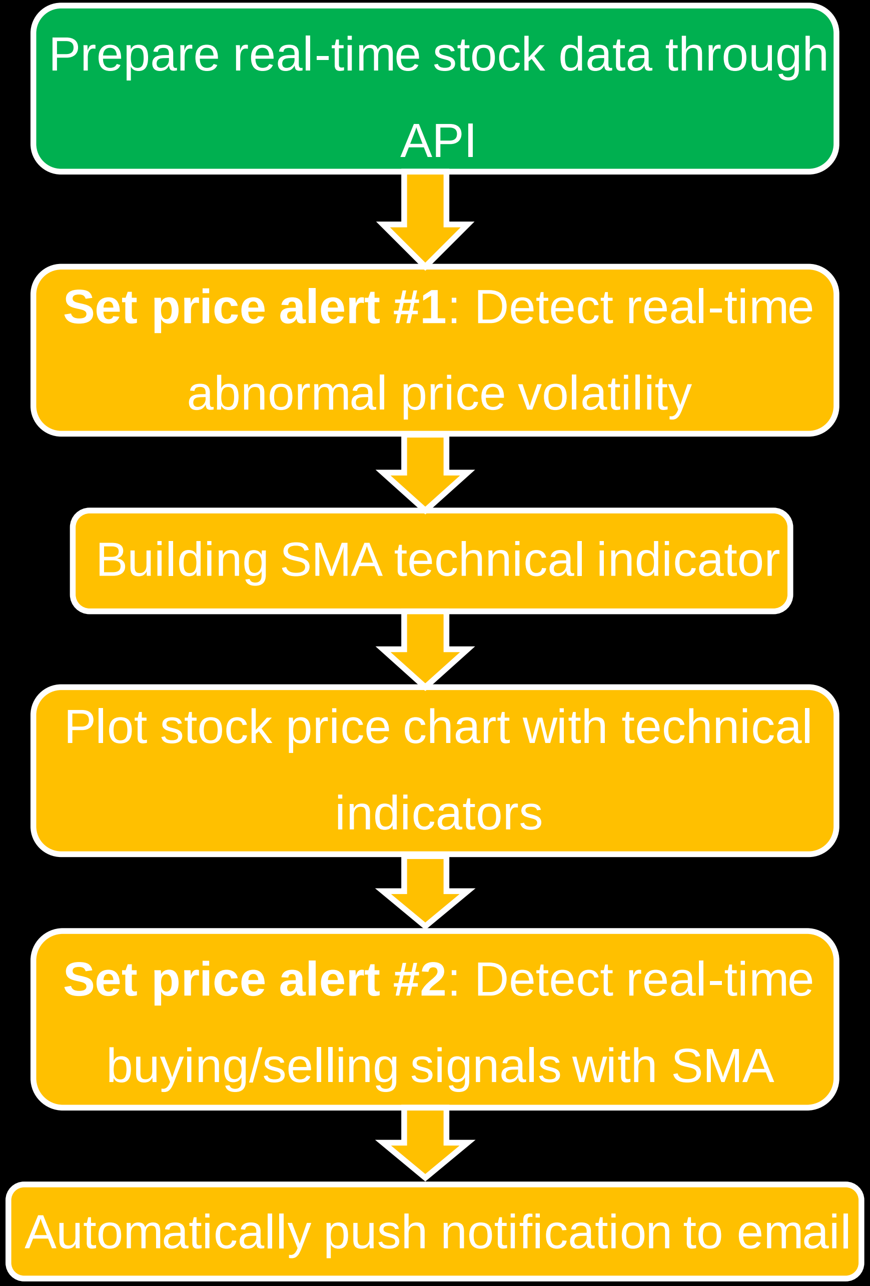
Automatically push notification to email

Prepare real-time stock data



```
#1: Load stock data to data frame:
api = 'PZ777Z6HJSVHI3NN'
ts = TimeSeries(key = api, output_format = 'pandas')

# Load Netflix(NFLX) stock data with interval of 1 minute:
data, meta_data = ts.get_intraday(symbol='NFLX', interval = '1min',
                                  outputsize = 'full')
```



Prepare real-time stock data through API

Set price alert #1: Detect real-time abnormal price volatility

Building SMA technical indicator

Plot stock price chart with technical indicators

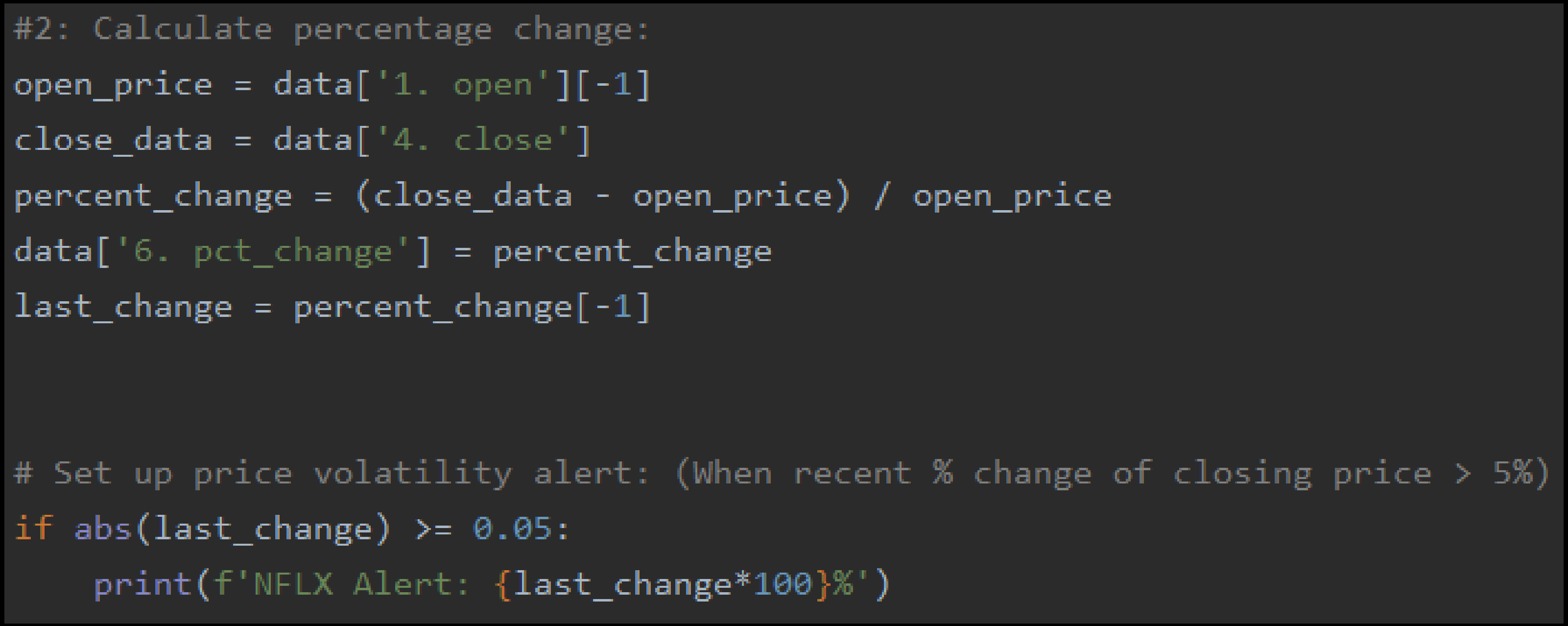
Set price alert #2: Detect real-time buying/selling signals with SMA

Automatically push notification to email

Set price alert #1

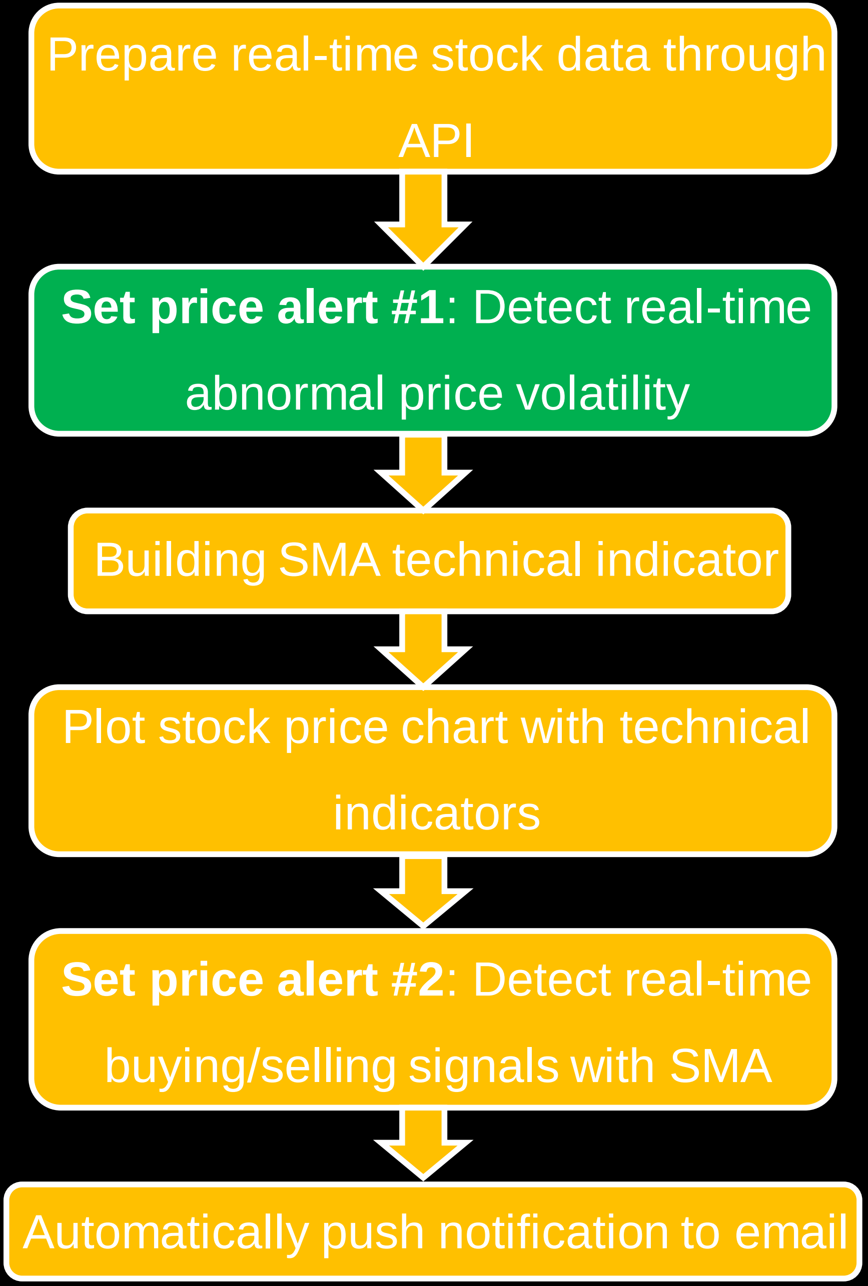
During a trading day, if the closing price differs from opening price more than 5%

==> Send alert



```
#2: Calculate percentage change:
open_price = data['1. open'][-1]
close_data = data['4. close']
percent_change = (close_data - open_price) / open_price
data['6. pct_change'] = percent_change
last_change = percent_change[-1]


# Set up price volatility alert: (When recent % change of closing price > 5%)
if abs(last_change) >= 0.05:
    print(f'NFLX Alert: {last_change*100}%')
```



Prepare real-time stock data through API

Set price alert #1: Detect real-time abnormal price volatility

Building SMA technical indicator

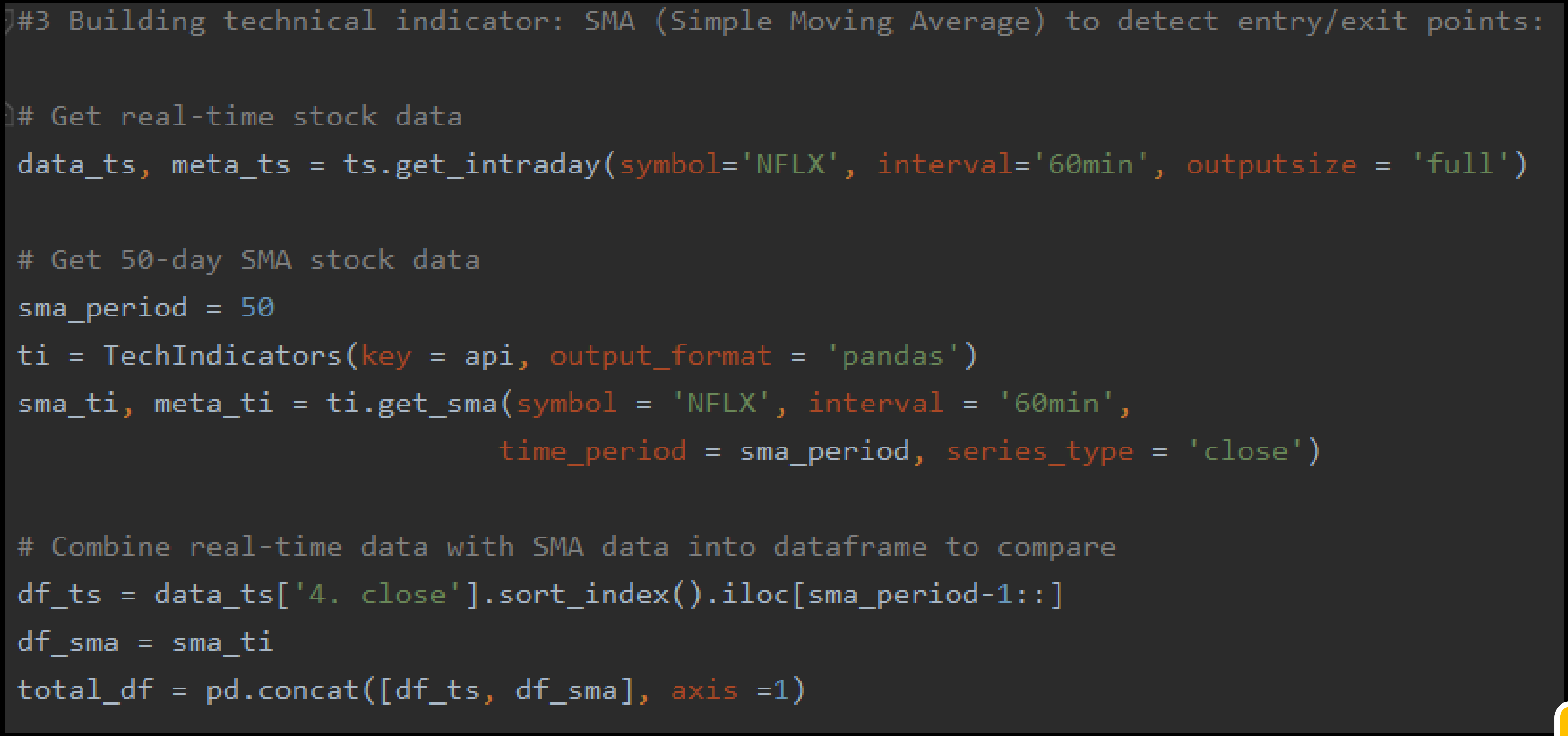
Plot stock price chart with technical indicators

Set price alert #2: Detect real-time buying/selling signals with SMA

Automatically push notification to email

Building SMA technical indicator

Simple Moving Average indicator (SMA) with a period of 50 days

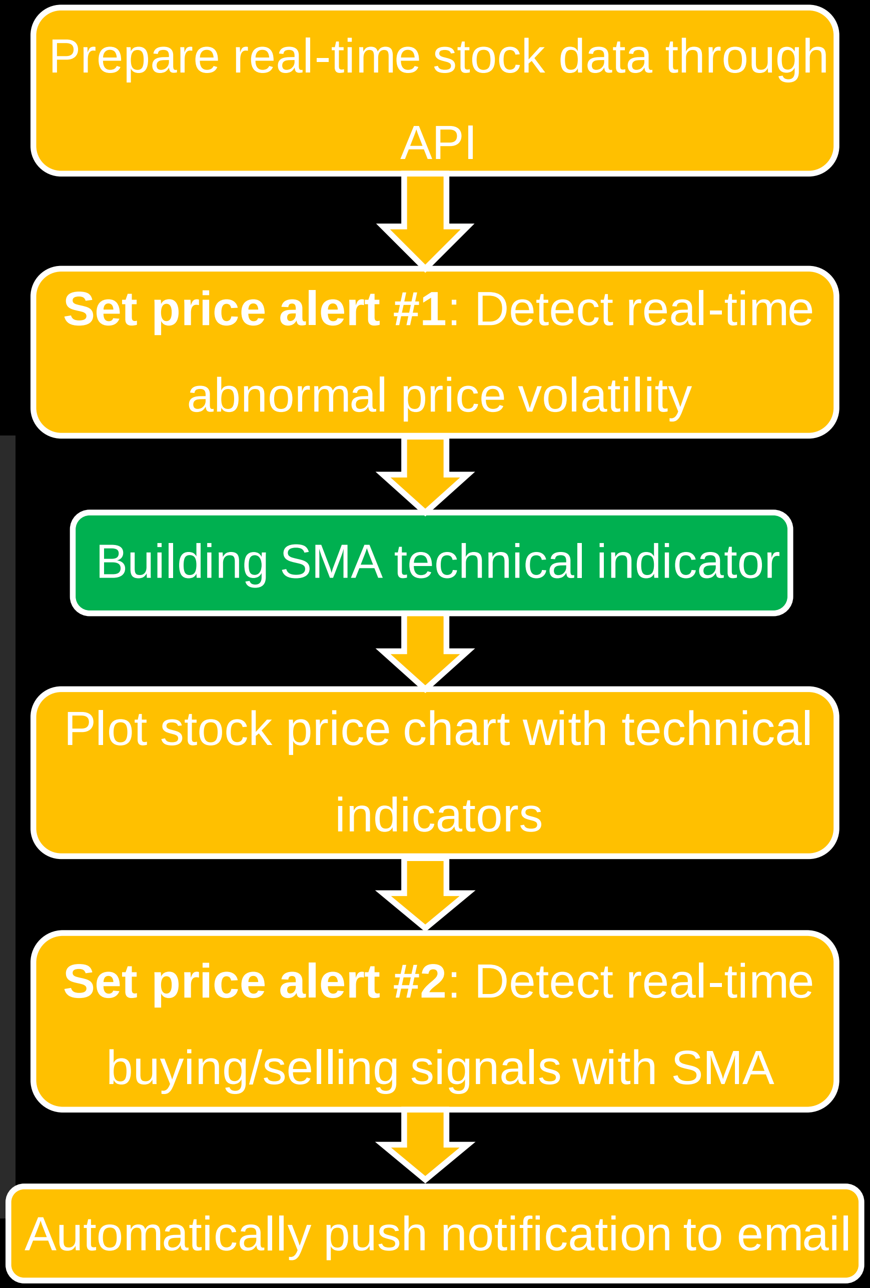


```
#3 Building technical indicator: SMA (Simple Moving Average) to detect entry/exit points:

# Get real-time stock data
data_ts, meta_ts = ts.get_intraday(symbol='NFLX', interval='60min', outputsize = 'full')

# Get 50-day SMA stock data
sma_period = 50
ti = TechIndicators(key = api, output_format = 'pandas')
sma_ti, meta_ti = ti.get_sma(symbol = 'NFLX', interval = '60min',
                             time_period = sma_period, series_type = 'close')

# Combine real-time data with SMA data into dataframe to compare
df_ts = data_ts['4. close'].sort_index().iloc[sma_period-1::]
df_sma = sma_ti
total_df = pd.concat([df_ts, df_sma], axis =1)
```



Prepare real-time stock data through API

Set price alert #1: Detect real-time abnormal price volatility

Building SMA technical indicator

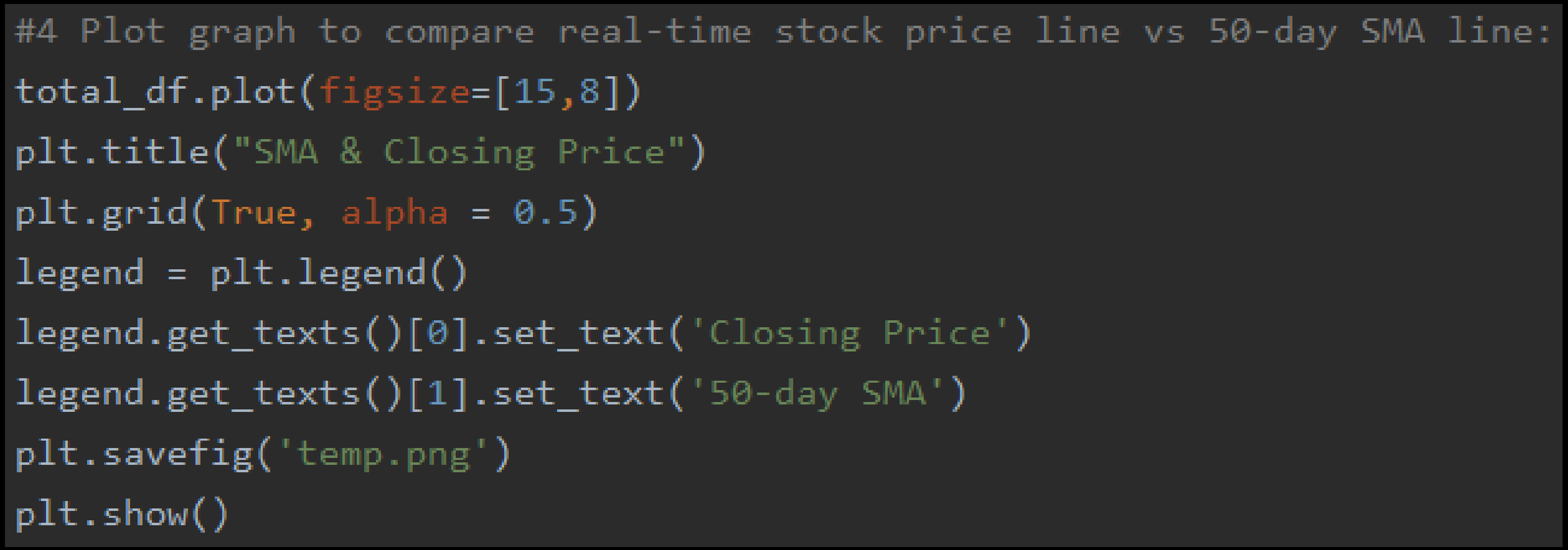
Plot stock price chart with technical indicators

Set price alert #2: Detect real-time buying/selling signals with SMA

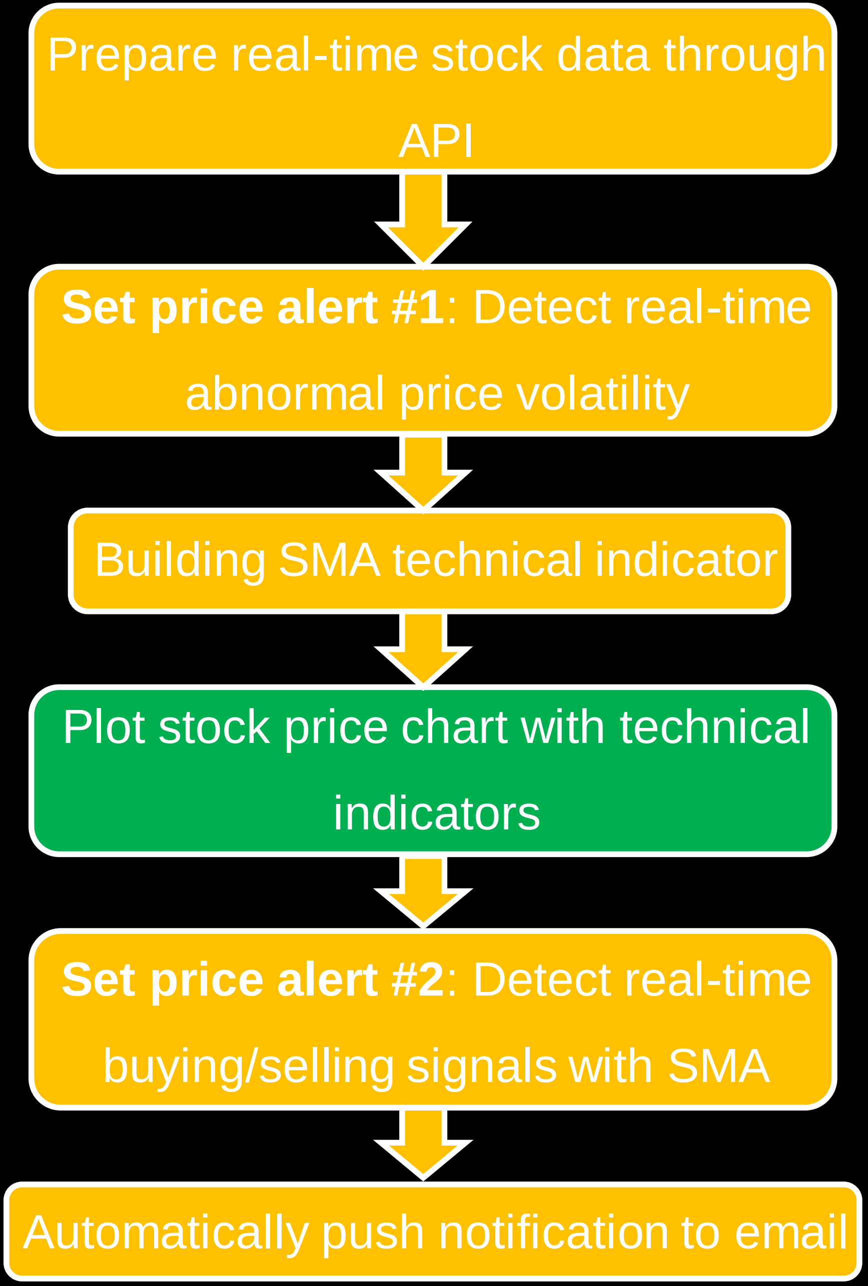
Automatically push notification to email

Plot stock price chart with technical indicator

Build chart to compare Closing Price line vs SMA line



```
#4 Plot graph to compare real-time stock price line vs 50-day SMA line:
total_df.plot(figsize=[15,8])
plt.title("SMA & Closing Price")
plt.grid(True, alpha = 0.5)
legend = plt.legend()
legend.get_texts()[0].set_text('Closing Price')
legend.get_texts()[1].set_text('50-day SMA')
plt.savefig('temp.png')
plt.show()
```



Prepare real-time stock data through API

Set price alert #1: Detect real-time abnormal price volatility

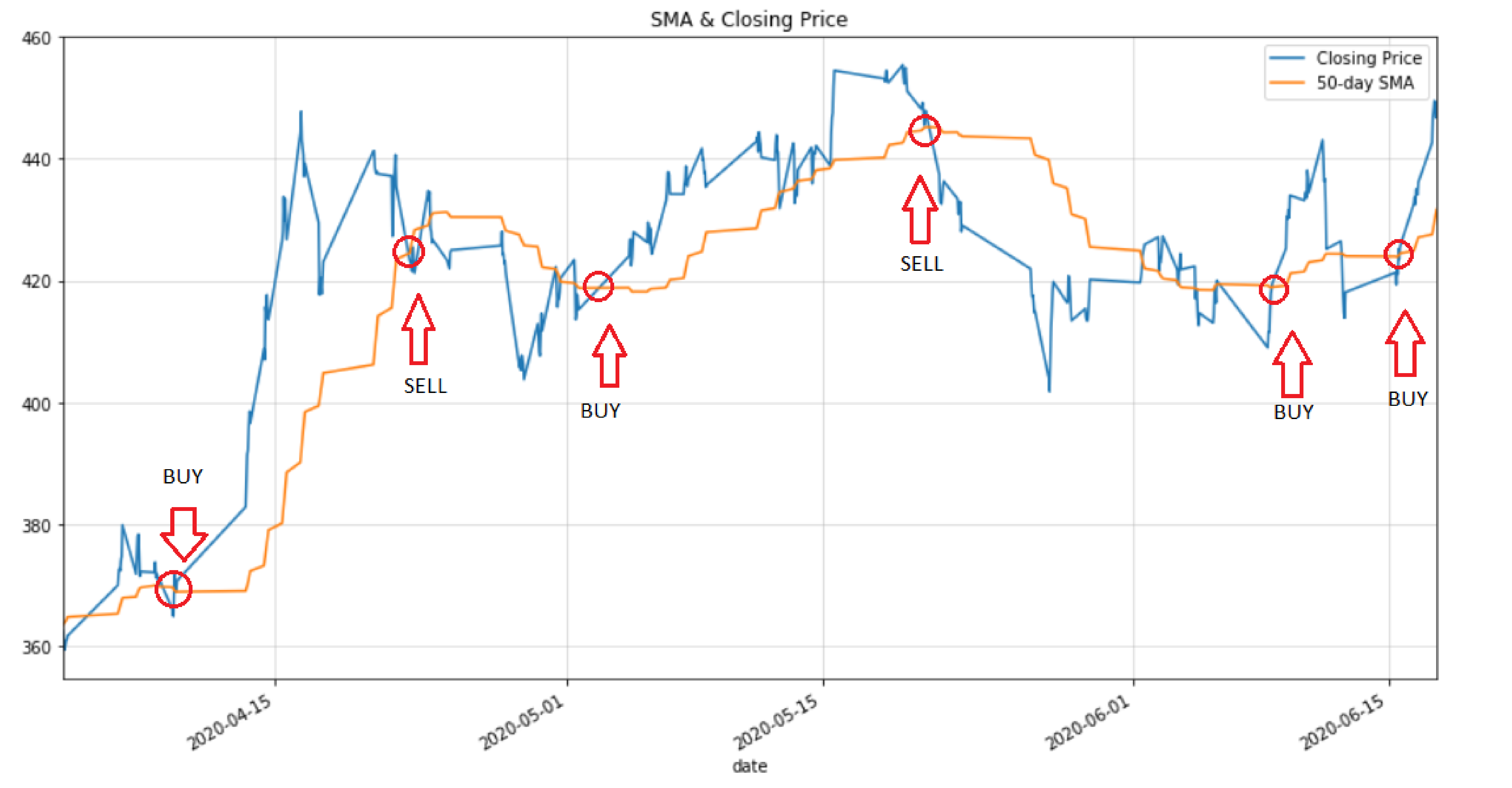
Building SMA technical indicator

Plot stock price chart with technical indicators

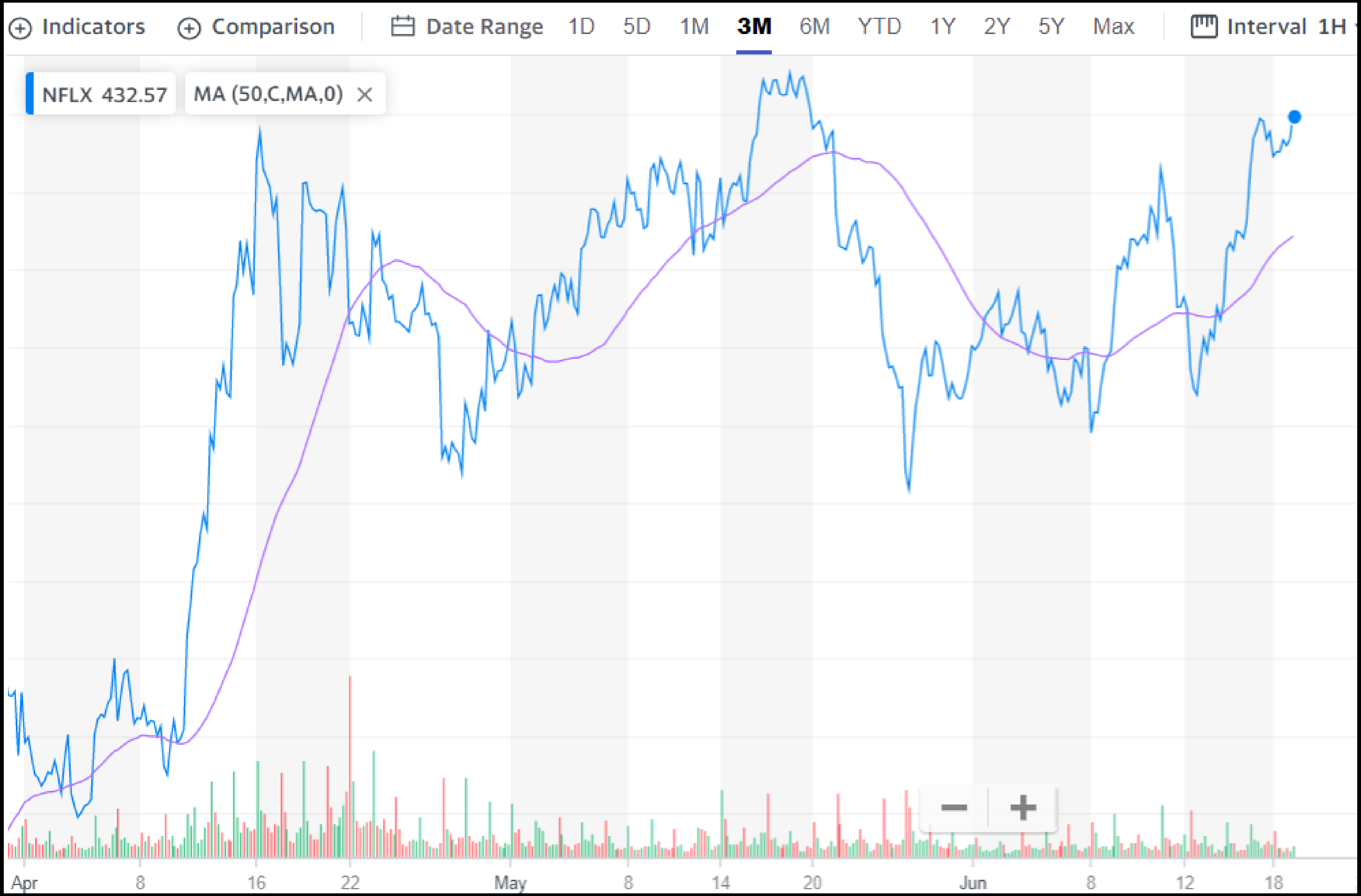
Set price alert #2: Detect real-time buying/selling signals with SMA

Automatically push notification to email

50-day SMA vs Closing Price chart



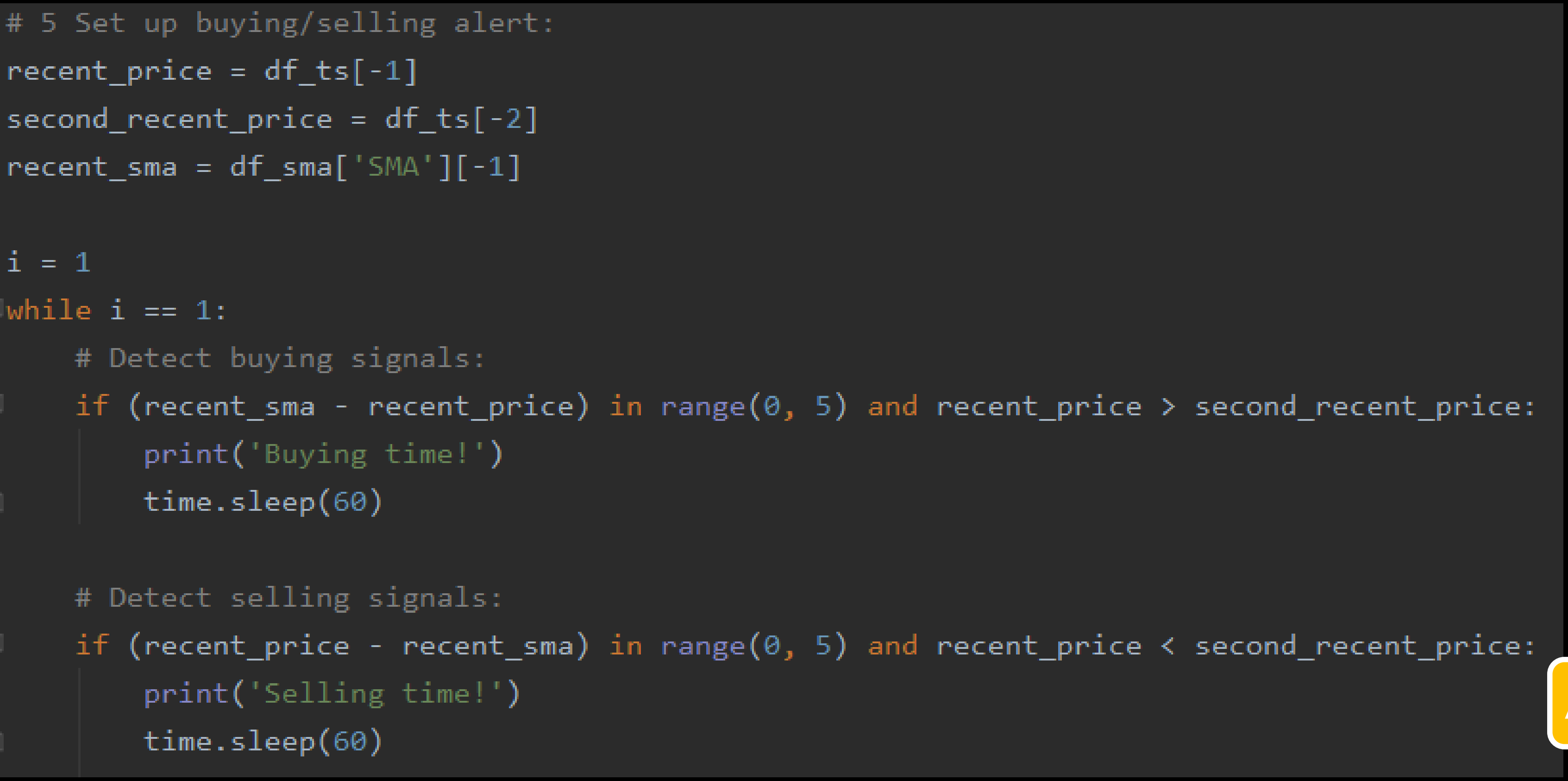
Accuracy Check with Yahoo Finance



Set price alert #2

Buying signals: Buy stock <u>before</u> closing price line cross over SMA line (Price line is increasing & 5 points lower than SMA)

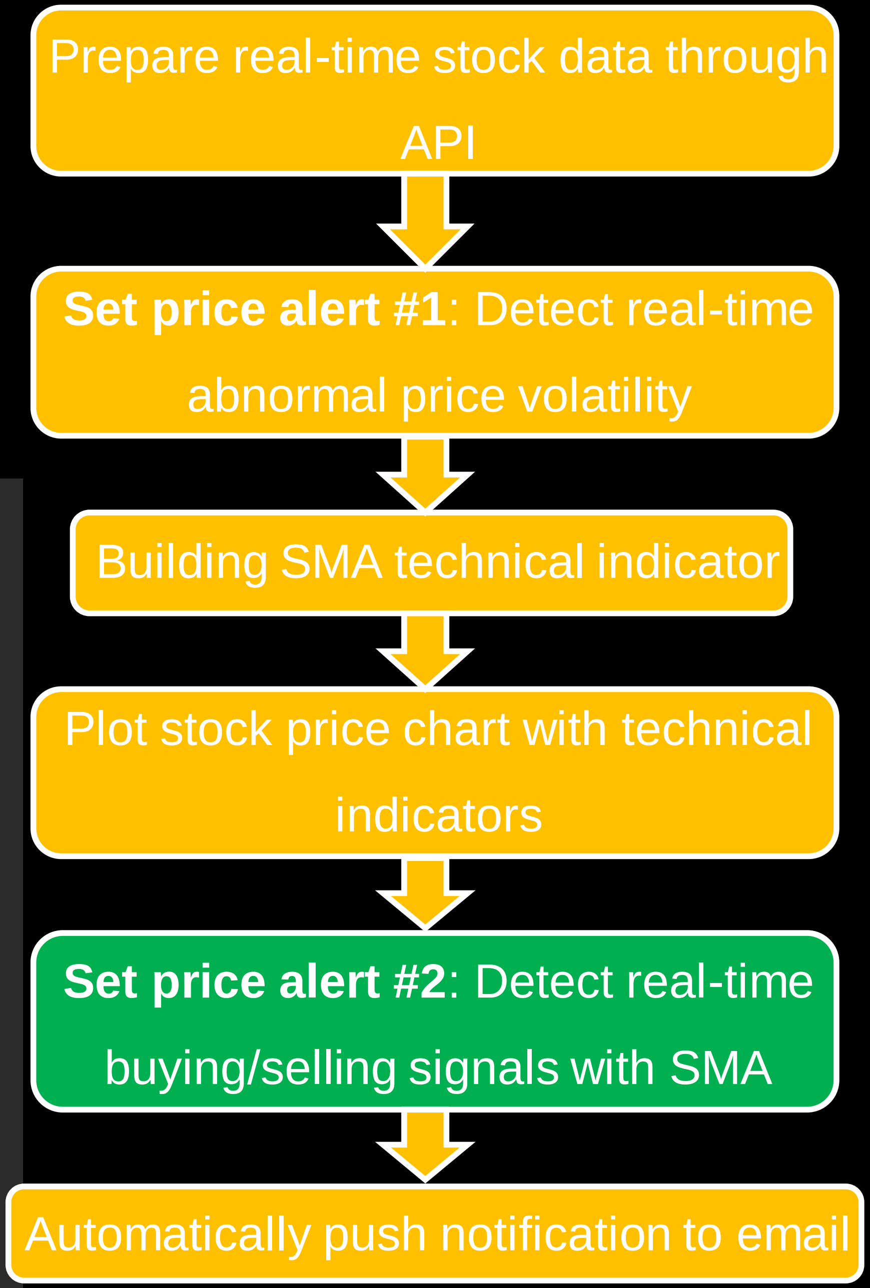
Selling signals: Sell stock <u>before</u> SMA line cross over closing price line (Price line is decreasing & 5 points higher than SMA)



```python
# 5 Set up buying/selling alert:
recent_price = df_ts[-1]
second_recent_price = df_ts[-2]
recent_sma = df_sma['SMA'][-1]

i = 1
while i == 1:
    # Detect buying signals:
    if (recent_sma - recent_price) in range(0, 5) and recent_price > second_recent_price:
        print('Buying time!')
        time.sleep(60)

    # Detect selling signals:
    if (recent_price - recent_sma) in range(0, 5) and recent_price < second_recent_price:
        print('Selling time!')
        time.sleep(60)
```



Prepare real-time stock data through API

Set price alert #1: Detect real-time abnormal price volatility

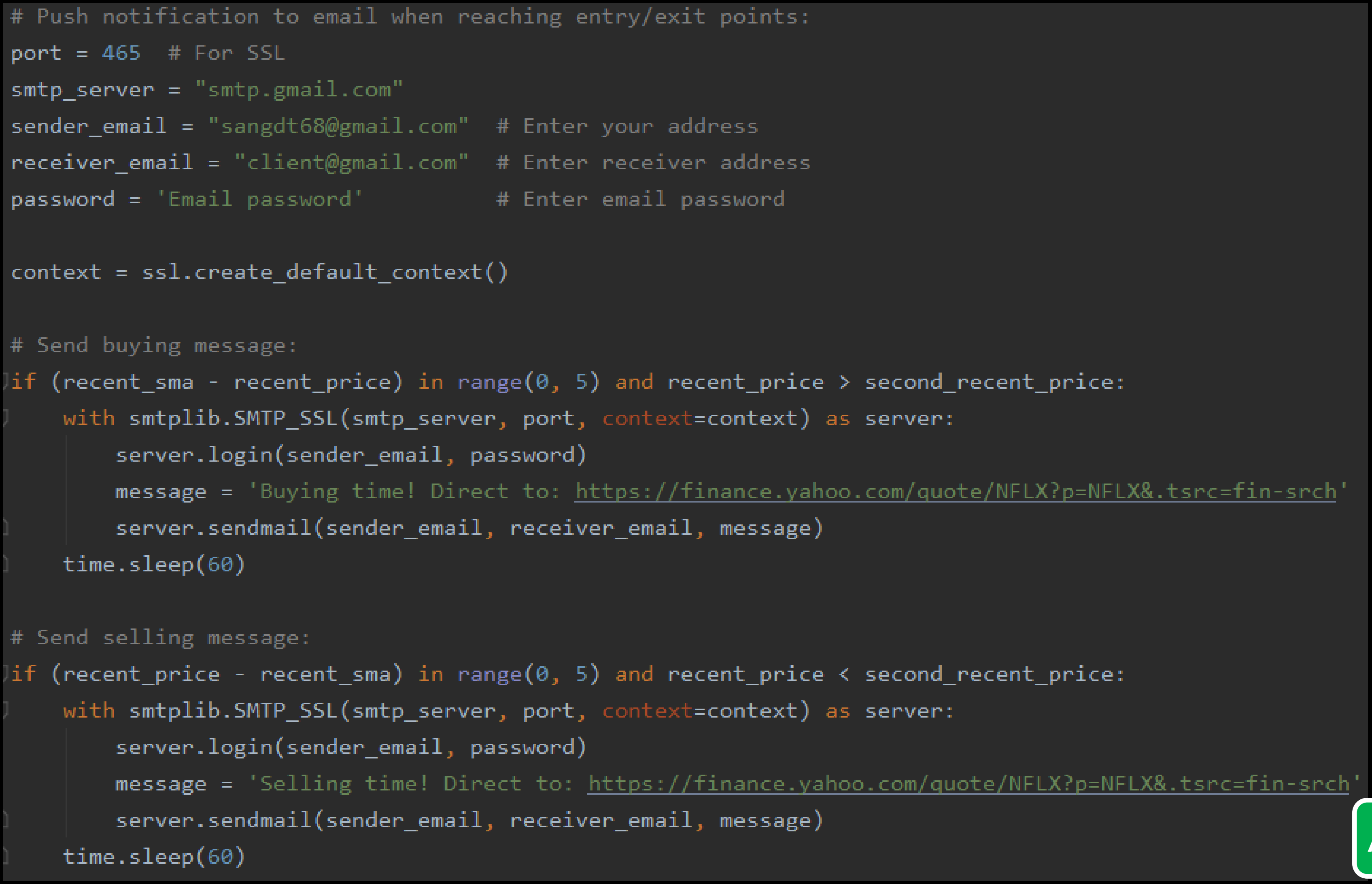
Building SMA technical indicator

Plot stock price chart with technical indicators

Set price alert #2: Detect real-time buying/selling signals with SMA

Automatically push notification to email

Send email notification

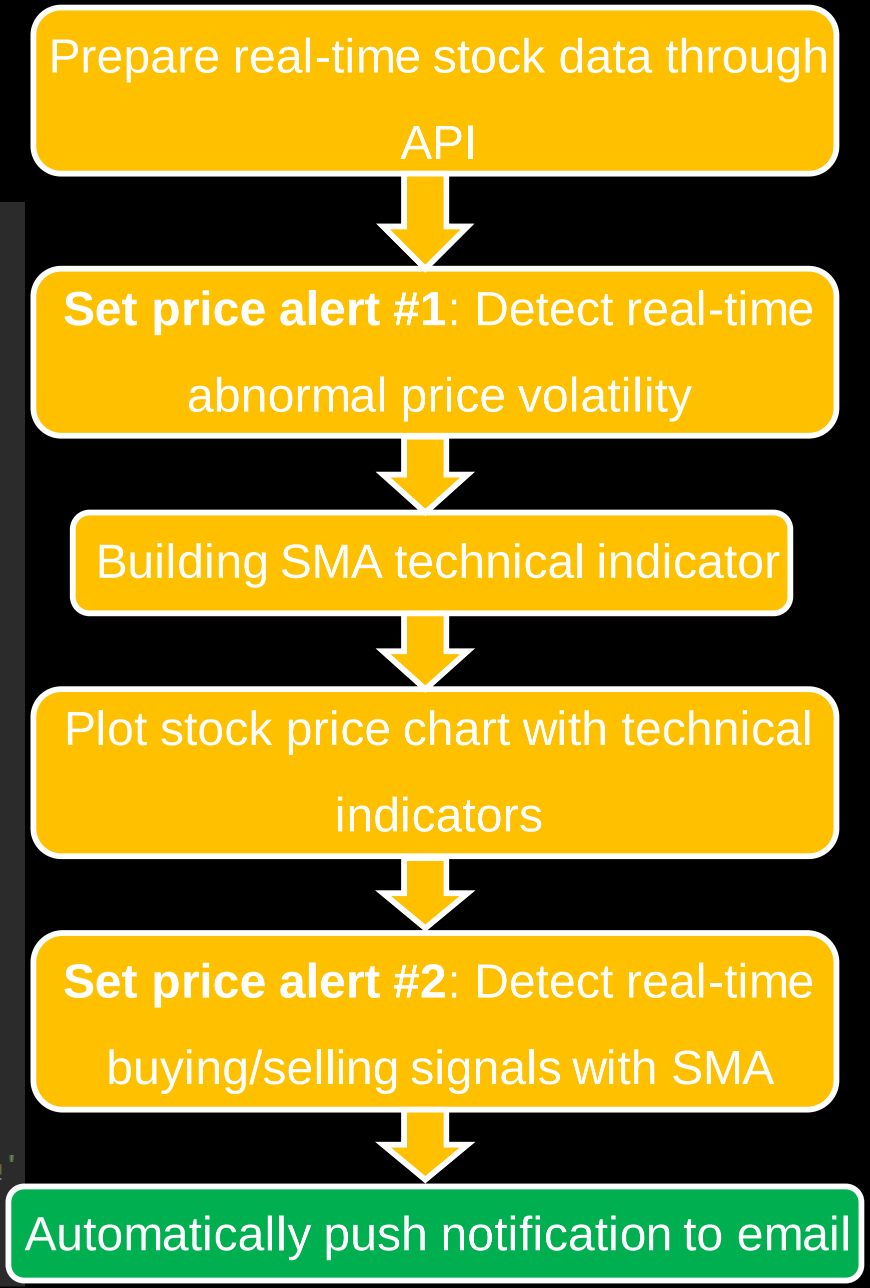


```
# Push notification to email when reaching entry/exit points:
port = 465  # For SSL
smtp_server = "smtp.gmail.com"
sender_email = "sangdt68@gmail.com"  # Enter your address
receiver_email = "client@gmail.com"  # Enter receiver address
password = 'Email password'          # Enter email password

context = ssl.create_default_context()

# Send buying message:
if (recent_sma - recent_price) in range(0, 5) and recent_price > second_recent_price:
    with smtplib.SMTP_SSL(smtp_server, port, context=context) as server:
        server.login(sender_email, password)
        message = 'Buying time! Direct to: https://finance.yahoo.com/quote/NFLX?p=NFLX&.tsrc=fin-srch'
        server.sendmail(sender_email, receiver_email, message)
    time.sleep(60)

# Send selling message:
if (recent_price - recent_sma) in range(0, 5) and recent_price < second_recent_price:
    with smtplib.SMTP_SSL(smtp_server, port, context=context) as server:
        server.login(sender_email, password)
        message = 'Selling time! Direct to: https://finance.yahoo.com/quote/NFLX?p=NFLX&.tsrc=fin-srch'
        server.sendmail(sender_email, receiver_email, message)
    time.sleep(60)
```



Prepare real-time stock data through API

Set price alert #1: Detect real-time abnormal price volatility

Building SMA technical indicator

Plot stock price chart with technical indicators

Set price alert #2: Detect real-time buying/selling signals with SMA

Automatically push notification to email

Thank you!

Full code: https://github.com/zaubers8/Python-Projects.git