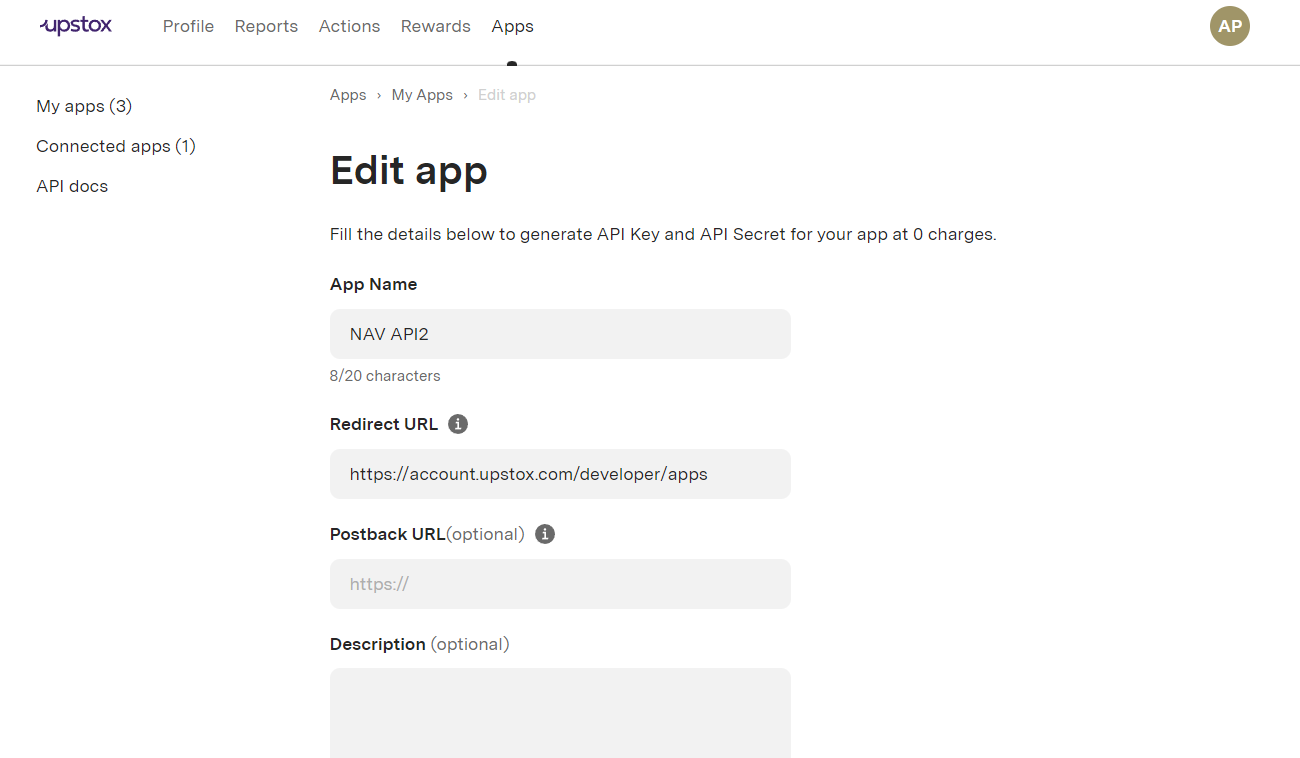
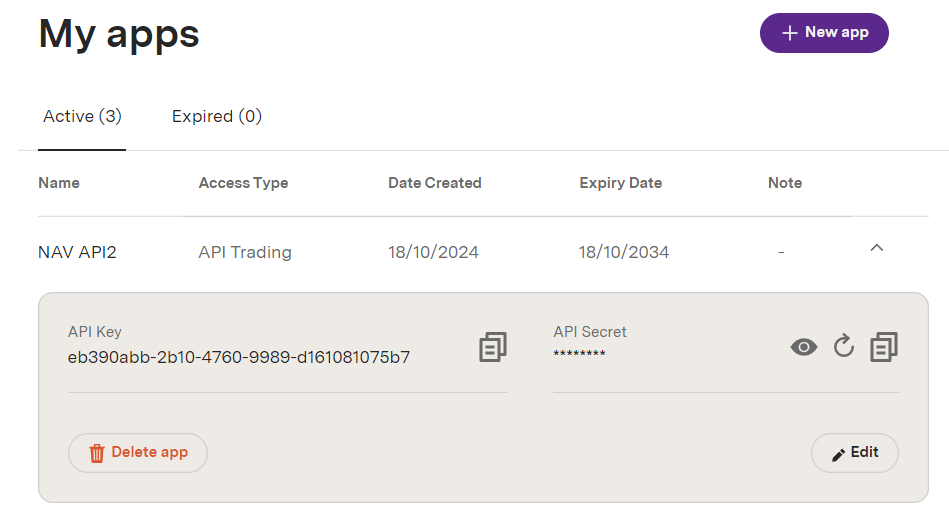
## Create API app in upstox developer portal:

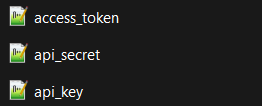
https://account.upstox.com/developer/apps



Copy key & secret to text files:



## Create text files for storing api details:



## Run following commands in cmd:

pip install requests

pip install upstox-python-sdk

pip install pandas

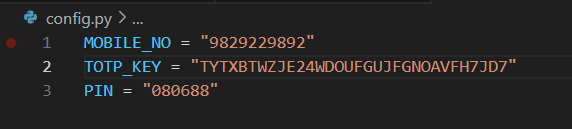
pip install openxyl

pip install pyotp

pip install playwright

python –m playwright install

## Enable TOTP in upstox for auto login and create config.py file for storing details:



## 1\_generate\_code.py: Generate ‘code’ value using below code:

import requests

api\_key=open('api\_key.txt','r').read().strip()

rurl = 'https://127.0.0.1/' # or ‘https://account.upstox.com/developer/apps’

uri = f'https://api.upstox.com/v2/login/authorization/dialog?response\_type=code&client\_id={ api\_key }&redirect\_uri={rurl}'

print(uri)

* Now grab Code from uri:

uri = <https://api.upstox.com/v2/login/authorization/dialog?response_type=code&client_id=5b40e389-d2f7-4000-a57d-453d7d94d856&redirect_uri=https://127.0.0.1/>

redirected to -> <https://127.0.0.1/?code=ok4YyM>

## 2\_generate\_token.py: Now generate ‘access\_token’ using below code:

import requests

api\_key=open('api\_key.txt','r').read().strip()

api\_secret=open('api\_secret.txt','r').read().strip()

uri='https://account.upstox.com/developer/apps'

code='ok4YyM'

url='https://api-v2.upstox.com/login/authorization/token'

headers={

'accept': 'application/json',

'Api-Version': '2.0',

'Content-Type': 'application/x-www-form-urlencoded'

}

data={

'code':code,

'client\_id':api\_key,

'client\_secret':api\_secret,

'redirect\_uri':uri,

'grant\_type':'authorization\_code'

}

response=requests.post(url,headers=headers,data=data)

access\_token=response.json()['access\_token']

print(access\_token,file=open('access\_token.txt','w')) #write access-token to txt file

#print("token: "+access\_token)

## 3\_get\_live\_quotes.py: Now create request:

import upstox\_client

import time

import pandas as pd

instrument\_keys = ["NSE\_EQ|INE00QS24019", "NSE\_EQ|INE00QS24027", "NSE\_EQ|INE00QS24035", "NSE\_EQ|INE00QS24043", "NSE\_EQ|INE00XE07119

access\_token = open("access\_token.txt").read().strip()

# Creating a list to hold the dictionary

out = []

# Function to dynamically create and add a dictionary to the list

def add\_dictionary(ISIN, AskRate, AskQty):

# Creating a dictionary

new\_dict = {"ISIN": ISIN, "AskRate": AskRate, "AskQty": AskQty}

# Append the new dictionary to the list

out.append(new\_dict)

return out

def on\_message(message):

# print(message)

# print(type(message))

message\_dict = message # change to 'message' when getting real time data else 'temp'

# Now you can extract 'bidAskQuote' from message\_dict if it exists

feeds = message\_dict.get("feeds", {})

for feed\_key, feed\_value in feeds.items():

bid\_ask\_quotes = feed\_value["ff"]["marketFF"]["marketLevel"].get(

"bidAskQuote", []

)

# print(f"BidAskQuotes for {feed\_key}: {bid\_ask\_quotes} \n")

for quote in bid\_ask\_quotes:

# Extract 'askQ' from each quote (if it exists)

aq = quote.get("aq", "0.0") # Default to 'N/A' if 'aq' is not present

# print(f"askQty: {aq}")

ap = quote.get("ap", "0.0")

# print(f"askRate: {ap}")

add\_dictionary(feed\_key, ap, aq)

print(f"final op:\n {out} \n")

# Convert the data into a pandas DataFrame for writing into Excel file

df = pd.DataFrame(out)

# Specify the path where you want to save the Excel file

output\_file = "bid\_ask\_quotes.xlsx"

# Write the DataFrame to an Excel file

df.to\_excel(output\_file, index=False)

# df.to\_excel(output\_file, sheet\_name='BidAskData', index=False)

def main():

config = upstox\_client.Configuration()

config.access\_token = access\_token

streamer = upstox\_client.MarketDataStreamer(

upstox\_client.ApiClient(config), instrument\_keys, "full"

)

streamer.on("message", on\_message)

streamer.connect()

try:

time.sleep(15)

finally:

streamer.disconnect()

print("Streamer disconnected.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

## 1.1\_login\_upstox.py: auto-login into opstox using TOTP:

from urllib.parse import parse\_qs, urlparse

import requests

from playwright.sync\_api import Playwright, sync\_playwright, expect

import pyotp

from config import MOBILE\_NO, PIN, TOTP\_KEY

api\_key = open("api\_key.txt", "r").read().strip()

rurl = "https://account.upstox.com/developer/apps"

uri = f"https://api.upstox.com/v2/login/authorization/dialog?response\_type=code&client\_id={ api\_key }&redirect\_uri={rurl}"

print(uri)

code=""

def run(playwright: Playwright) -> str:

browser = playwright.chromium.launch(headless=False)

context = browser.new\_context()

page = context.new\_page()

with page.expect\_request(f"\*{rurl}?code\*") as request:

page.goto(uri)

page.locator("#mobileNum").click()

page.locator("#mobileNum").fill(MOBILE\_NO)

page.get\_by\_role("button", name="Get OTP").click()

page.locator("#otpNum").click()

otp = pyotp.TOTP(TOTP\_KEY).now()

page.locator("#otpNum").fill(otp)

page.get\_by\_role("button", name="Continue").click()

page.get\_by\_label("Enter 6-digit PIN").click()

page.get\_by\_label("Enter 6-digit PIN").fill(PIN)

res = page.get\_by\_role("button", name="Continue").click()

page.wait\_for\_load\_state()

url = request.value.url

print("Redirect Url with code{url}")

parsed = urlparse(url)

code = parse\_qs(parsed.query)["code"][0]

context.close()

browser.close()

return code

with sync\_playwright() as playwright:

code = run(playwright)

api\_key = open("api\_key.txt", "r").read().strip()

api\_secret = open("api\_secret.txt", "r").read().strip()

uri = "https://account.upstox.com/developer/apps"

url = "https://api-v2.upstox.com/login/authorization/token"

headers = {

"accept": "application/json",

"Api-Version": "2.0",

"Content-Type": "application/x-www-form-urlencoded",

}

data = {

"code": code,

"client\_id": api\_key,

"client\_secret": api\_secret,

"redirect\_uri": uri,

"grant\_type": "authorization\_code",

}

response = requests.post(url, headers=headers, data=data)

access\_token = response.json()["access\_token"]

print(

access\_token, file=open("access\_token.txt", "w")

) # write 'access-token' to txt file

print("token: " + access\_token)

## 3.1\_get\_live\_quotes copy.py: for reading top 20 ISIN & writing ask/rate into excel

import upstox\_client

import time

import pandas as pd

input\_file = 'ISIN.xlsx'

df = pd.read\_excel(input\_file)

# Fetch the top 10 rows from the 'ISIN' column

top\_ISIN = df['ISIN2'].head(20)

instrument\_keys = top\_ISIN.tolist()

print(instrument\_keys)

# instrument\_keys = ['NSE\_EQ|INE342T07445']#["NSE\_EQ|INE00QS24019", "NSE\_EQ|INE00QS24027", "NSE\_EQ|INE00QS24035", "NSE\_EQ|INE00QS24043", "NSE\_EQ|INE00XE07119

access\_token = open("access\_token.txt").read().strip()

# Creating a list to hold the dictionary

out2 = []

new\_dict2 = {}

def on\_message(message):

# print(message)

# print(type(message))

message\_dict = message # change to 'message' when getting real time data else 'temp'

# Now you can extract 'bidAskQuote' from message\_dict if it exists

feeds = message\_dict.get("feeds", {})

for feed\_key, feed\_value in feeds.items():

bid\_ask\_quotes = feed\_value["ff"]["marketFF"]["marketLevel"].get(

"bidAskQuote", []

)

print(f"BidAskQuotes for {feed\_key}: {bid\_ask\_quotes} \n")

new\_dict2.update({"ISIN" : feed\_key})

if(bid\_ask\_quotes == []):

print("no data from API\n")

for i in range(5):

new\_dict2.update({f"AskRate{i+1}" : "no data"})

new\_dict2.update({f"AskQty{i+1}" : "no data"})

else:

for i in range(5):

new\_dict2.update({f"AskRate{i+1}" : bid\_ask\_quotes[i].get('ap','0.0')})

new\_dict2.update({f"AskQty{i+1}" : bid\_ask\_quotes[i].get('aq','0.0')})

out2.append(new\_dict2.copy()) # \*\*\*\*\*\*\*\*\* new\_dict2.copy() creates a shallow copy of new\_dict2, so out contains an independent copy of the dictionary.

new\_dict2.clear()

print(f"final op:\n {out2}")

# Convert the data into a pandas DataFrame for writing into Excel file

df = pd.DataFrame(out2)

# Specify the path where you want to save the Excel file

output\_file = "bid\_ask\_quotes.xlsx"

# Write the DataFrame to an Excel file

df.to\_excel(output\_file, index=False)

# df.to\_excel(output\_file, sheet\_name='BidAskData', index=False)

def main():

config = upstox\_client.Configuration()

config.access\_token = access\_token

streamer = upstox\_client.MarketDataStreamer(

upstox\_client.ApiClient(config), instrument\_keys, "full"

)

streamer.on("message", on\_message)

streamer.connect()

try:

time.sleep(15)

finally:

streamer.disconnect()

print("Streamer disconnected.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

## OHLC.py: get ohlc data for 1month period

import requests

import pandas as pd

input\_file = r"C:\Users\07nav\Downloads\upstoxbaji2\ohlc-input.xlsx"

df = pd.read\_excel(input\_file)

# Fetch the top 20 rows from the 'ISIN' column

top\_ISIN = df["InstrumentKeyName"].head(20)

top\_name = df["Name"].head(20)

instrument\_keys = top\_ISIN.tolist()

instrument\_name = top\_name.tolist()

print(instrument\_keys)

access\_token = open("access\_token.txt", "r").read().strip()

# quote = 'https://api.upstox.com/v2/historical-candle/NSE\_EQ|INE040A01034/month/2023-12-31/2023-01-01' #to-date/from-date

# quote='https://api-v2.upstox.com/market-quote/ohlc'

headers = {

"accept": "application/json",

"Api-Version": "2.0",

"Authorization": f"Bearer {access\_token}",

}

out = []

toDateFromDate = "2024-11-30/2022-01-01"

# Calling API for each instrument and appending to out[]

for i in instrument\_keys:

quote = (

"https://api.upstox.com/v2/historical-candle/NSE\_EQ|"

+ i

+ "/month/"+toDateFromDate

) # to-date/from-date

print("instrument: ", i)

response = requests.get(quote, headers=headers)

# Parse the JSON response

response\_data = response.json()

candles = response\_data.get("data", {}).get("candles", [])

print(f"candles for {i}: {candles}")

# Append each candle data with the instrument key as a new row

for candle in candles:

out.append([i] + candle)

print("out: ", out)

# Create a DataFrame

columns = ["instrument", "timestamp", "open", "high", "low", "close", "volume", "unknown"]

df\_out = pd.DataFrame(out, columns=columns)

# Select the relevant columns

df\_out = df\_out[["instrument", "timestamp", "open", "high", "low", "close"]]

# Write to an Excel file

output\_file = r"C:\Users\07nav\Downloads\upstoxbaji2\ohlc-output.xlsx"

df\_out.to\_excel(output\_file, index=False)

print(f"Data written to {output\_file}")