A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

import json

import http.client

import urllib.parse

from datetime import datetime

import psycopg2

import os

from utils.schemas import NEWS\_TABLE\_SCHEMA

from utils.functions import validated\_data

def lambda\_handler(event, context):

META\_DATA\_INFO = ['author', 'title', 'description', 'url', 'source', 'category', 'published\_at', 'country']

# Database connection

try:

db\_conn = psycopg2.connect(

dbname=os.environ['DB\_NAME'],

user=os.environ['DB\_USER'],

password=os.environ['DB\_PASSWORD'],

host=os.environ['DB\_HOST'],

port=os.environ['DB\_PORT']

)

cursor = db\_conn.cursor()

except Exception as e:

error\_msg = f"Database connection failed due to: {e}"

print(error\_msg)

return {

'statusCode': 500,

'body': json.dumps(error\_msg)

}

# Prepare API request

date\_today = datetime.today().strftime('%Y-%m-%d')

params = urllib.parse.urlencode({

'access\_key': os.environ['API\_ACCESS\_KEY'],

'sources': '-cnn',

'categories': 'health, science, technology, -general,-sports',

'languages': 'en',

'date': date\_today,

'sort': 'published\_desc',

'limit': 100,

})

# API connection and data retrieval

try:

api\_conn = http.client.HTTPSConnection('api.mediastack.com', timeout=10)

api\_conn.request('GET', f'/v1/news?{params}')

res = api\_conn.getresponse()

data = res.read()

news\_data = json.loads(data.decode('utf-8'))

api\_conn.close()

if not news\_data['data']:

return {

'statusCode': 204,

'body': json.dumps('No news data to process.')

}

# Data insertion

for item in news\_data['data']:

validated\_data\_list = validated\_data(item, NEWS\_TABLE\_SCHEMA)

if not validated\_data\_list:

print("Data validation failed.")

continue # Skip insertion if data is not valid

placeholders = ', '.join(['%s'] \* len(META\_DATA\_INFO))

columns = ', '.join(META\_DATA\_INFO)

sql = f"INSERT INTO news\_articles ({columns}) VALUES ({placeholders})"

cursor.execute(sql, tuple(validated\_data\_list))

db\_conn.commit()

return {

'statusCode': 200,

'body': json.dumps('Data retrieved and stored successfully')

}

except (ValueError, TypeError) as e:

error\_msg = f"Data validation error: {e}"

print(error\_msg)

db\_conn.rollback()

return {'statusCode': 400, 'body': json.dumps(error\_msg)}

except Exception as e:

error\_msg = f"Failed to insert data due to error: {e}"

print(error\_msg)

db\_conn.rollback()

return {'statusCode': 500, 'body': json.dumps(error\_msg)}

finally:

cursor.close()

db\_conn.close()

0-0----------------------------------

import json

import http.client

import urllib.parse

from datetime import datetime

import psycopg2

import os

from utils.schemas import NEWS\_TABLE\_SCHEMA

from utils.functions import validated\_data

def lambda\_handler(event, context):

META\_DATA\_INFO = ['author', 'title', 'description', 'url', 'source', 'category', 'published\_at', 'country']

# Database connection

try:

db\_conn = psycopg2.connect(

dbname=os.environ['DB\_NAME'],

user=os.environ['DB\_USER'],

password=os.environ['DB\_PASSWORD'],

host=os.environ['DB\_HOST'],

port=os.environ['DB\_PORT']

)

cursor = db\_conn.cursor()

except Exception as e:

error\_msg = f"Database connection failed due to: {e}"

print(error\_msg)

return {

'statusCode': 500,

'body': json.dumps(error\_msg)

}

# Prepare API request using the modified connection setup

date\_today = datetime.today().strftime('%Y-%m-%d')

params = urllib.parse.urlencode({

'access\_key': os.environ['API\_ACCESS\_KEY'],

'sources': '-cnn',

'categories': 'health, science, technology, -general,-sports',

'languages': 'en',

'date': date\_today,

'sort': 'published\_desc',

'limit': 100,

})

# Revised API connection and data retrieval

conn = http.client.HTTPSConnection('api.mediastack.com')

conn.request('GET', '/v1/news?' + params)

res = conn.getresponse()

data = res.read()

news\_data = json.loads(data.decode('utf-8'))

conn.close()

# Check for data presence

if 'data' not in news\_data or not news\_data['data']:

return {

'statusCode': 204,

'body': json.dumps('No news data to process.')

}

# Data insertion into the database

try:

for item in news\_data['data']:

validated\_data\_list = validated\_data(item, NEWS\_TABLE\_SCHEMA)

if not validated\_data\_list:

print("Data validation failed.")

continue # Skip insertion if data is not valid

placeholders = ', '.join(['%s'] \* len(META\_DATA\_INFO))

columns = ', '.join(META\_DATA\_INFO)

sql = f"INSERT INTO news\_articles ({columns}) VALUES ({placeholders})"

cursor.execute(sql, tuple(validated\_data\_list))

db\_conn.commit()

return {

'statusCode': 200,

'body': json.dumps('Data retrieved and stored successfully')

}

except (ValueError, TypeError) as e:

error\_msg = f"Data validation error: {e}"

print(error\_msg)

db\_conn.rollback()

return {'statusCode': 400, 'body': json.dumps(error\_msg)}

except Exception as e:

error\_msg = f"Failed to insert data due to error: {e}"

print(error\_msg)

db\_conn.rollback()

return {'statusCode': 500, 'body': json.dumps(error\_msg)}

finally:

cursor.close()

db\_conn.close()