import json

import logging

import socketio

import threading

import pandas as pd

from flask import Flask

from flask\_socketio import SocketIO

from eef\_sdk.api.consumer import EefConsumer

from eef\_sdk.consumer.strategy\_types import (

RetryStrategy,

DiscardStrategy,

DlqStrategy,

ReprocessStrategy,

CompleteStrategy,

)

# Initialize Flask app and SocketIO for real-time updates

fl = Flask(\_\_name\_\_)

socketio = SocketIO(fl)

df = pd.DataFrame()

payload\_list = []

# Logger setup

logger = logging.getLogger(\_\_name\_\_)

logging.basicConfig(level=logging.INFO)

# Function to handle incoming Kafka messages

def listen\_handler\_function(payload, headers, key):

"""

Handles Kafka messages and maintains the global payload list.

Emits the updated payload list to WebSocket clients.

"""

print(f'Listened to the event: {payload}')

print('---- Event ID:', headers.get('eefEventId'))

print('---- Event Key:', key)

print('---- Event Correlation ID:', headers.get('eefEventCorrelationId', ''))

global payload\_list

payload\_list.append(json.loads(payload))

temp\_df = pd.DataFrame(payload\_list)

socketio.emit('new\_data', {'data': temp\_df.to\_dict('records')}) # Emit data to WebSocket clients

# Simplified on\_commit\_handler\_function

def on\_commit\_handler\_function(\*args):

"""

Called when Kafka event consumption is committed successfully.

"""

logger.info('On commit handler run')

logger.info(json.dumps(args[0], indent=4))

logger.info('Transaction of event consumption finished successfully')

# Return CompleteStrategy to indicate successful handling

return CompleteStrategy()

# Function for handling rollback of Kafka event consumption

def on\_rollback\_handler\_function(\*args):

"""

Handles rollback of Kafka event consumption.

"""

logger.info('On rollback handler run')

logger.info(json.dumps(args[0], indent=4))

logger.info('Transaction of event consumption rolled back')

# Function to read messages from Confluent Kafka

def read\_confluent\_kafka():

"""

Initializes the Kafka consumer and sets up listeners.

"""

path\_to\_config\_file = 'configuration/config.yaml' # Specify your Kafka configuration file

consumer = EefConsumer(path=path\_to\_config\_file)

consumer.on\_commit(on\_commit\_handler\_function)

consumer.on\_rollback(on\_rollback\_handler\_function)

consumer.listen(listen\_handler\_function) # Start listening to Kafka events

# Start the Kafka consumer in a separate thread

threading.Thread(target=read\_confluent\_kafka, daemon=True).start()

# Run the Flask-SocketIO server

if \_\_name\_\_ == '\_\_main\_\_':

socketio.run(fl, port=8060) # WebSocket server on port 8060