LAPORAN PRAKTIKUM

Junita Mareska Sihombing

11323042

DIII-TEKNOLOGI INFORMASI

INSTITUT TEKNOLOGI DEL

FAKULTAS VOKASI

```python
import json
from topic import Topic
from data_classes import BrokerSettings, ClientSettings

class Simulator:
    def __init__(self, settings_file):
        self.default_client_settings = ClientSettings(
            clean=True,
            retain=False,
            qos=2,
            time_interval=10
        )
        self.topics = self.load_topics(settings_file)

    def read_client_settings(self, settings_dict: dict, default: ClientSettings):
        return ClientSettings(
            clean=settings_dict.get('CLEAN_SESSION', default.clean),
            retain=settings_dict.get('RETAIN', default.retain),
            qos=settings_dict.get('QOS', default.qos),
            time_interval= settings_dict.get('TIME_INTERVAL', default.time_interval)
        )

    def load_topics(self, settings_file):
        topics = []
        with open(settings_file) as json_file:
            config = json.load(json_file)
            broker_settings = BrokerSettings(
                url=config.get('BROKER_URL', 'localhost'),
                port=config.get('BROKER_PORT', 1883),
                protocol=config.get('PROTOCOL_VERSION', 4) # mqtt.MQTTv311
            )
            broker_client_settings = self.read_client_settings(config, default=self.default_client_settings)
            # read each configured topic
            for topic in config['TOPICS']:
                topic_data = topic['DATA']
                topic_payload_root = topic.get('PAYLOAD_ROOT', {})
                topic_client_settings = self.read_client_settings(topic, default=broker_client_settings)
                if topic['TYPE'] == 'single':
                    # create single topic with format: /{PREFIX}
                    topic_url = topic['PREFIX']
                    topics.append(Topic(broker_settings, topic_url, topic_data, topic_payload_root, topic_client_settings))
                elif topic['TYPE'] == 'multiple':
                    # create multiple topics with format: /{PREFIX}/{id}
                    for id in range(topic['RANGE_START'], topic['RANGE_END']+1):
                        topic_url = topic['PREFIX'] + '/' + str(id)
                        topics.append(Topic(broker_settings, topic_url, topic_data, topic_payload_root, topic_client_settings))
                elif topic['TYPE'] == 'list':
                    # create multiple topics with format: /{PREFIX}/{item}
                    # create single topic with format: /{PREFIX}
                    topic_url = topic['PREFIX']
                    topics.append(Topic(broker_settings, topic_url, topic_data, topic_payload_root, topic_client_settings))
                elif topic['TYPE'] == 'multiple':
                    # create multiple topics with format: /{PREFIX}/{id}
                    for id in range(topic['RANGE_START'], topic['RANGE_END']+1):
                        topic_url = topic['PREFIX'] + '/' + str(id)
                        topics.append(Topic(broker_settings, topic_url, topic_data, topic_payload_root, topic_client_settings))
                elif topic['TYPE'] == 'list':
                    # create multiple topics with format: /{PREFIX}/{item}
                    for item in topic['LIST']:
                        topic_url = topic['PREFIX'] + '/' + str(item)
                        topics.append(Topic(broker_settings, topic_url, topic_data, topic_payload_root, topic_client_settings))
        return topics

    def run(self):
        for topic in self.topics:
            print(f'Starting: {topic.topic_url} ...')
            topic.start()
        for topic in self.topics:
            # workaround for Python 3.12
            topic.join()

    def stop(self):
        for topic in self.topics:
            print(f'Stopping: {topic.topic_url} ...')
            topic.stop()

```

Penjelasan :

1. Kelas simulator
    - Membaca konfigurasi dari file JSON ‘
    - Mengatur pengaturan klien MOTT dan topik
    - Menginisalisasi objek-objek topik (topik) berdasarkan konfigurasi.
    - Mengatur mekasnisme untuk memulai dan menghentikan proses yang terkait dengan topik
2. Kelas simulator
   Konstruktor(int)

    - Membuat pengaturan klien MQTT default menggunakan kelas ClientSettings (seperti opsi sesi bersih, QoS, retensi pesan, dan interval waktu).
    - Memuat semua topik dari file JSON menggunakan metode load_topics.

   Metode read_client_settings
    - Membaca pengaturan klien dari konfigurasi JSON.
    - Jika beberapa pengaturan tidak tersedia dalam file JSON, nilai default digunakan.

   Metode load_topics
    a. Membaca file JSON untuk mengambil:
       Pengaturan broker MQTT (seperti URL, port, dan protokol).
    - Pengaturan default klien untuk broker.
    - Daftar topik, yang bisa berupa:
    - Tipe single: Satu topik dengan nama tetap.
    - Tipe multiple: Sekumpulan topik dengan pola nama berbasis rentang angka.
    - Tipe list: Sekumpulan topik dengan pola nama berbasis elemen dari daftar.

   Setiap topik diinisialisasi sebagai objek Topic, dengan informasi seperti:
    - URL broker.
    - URL topik spesifik.
    - Data payload topik.
    - Akar payload tambahan.
    - Pengaturan klien untuk topik tersebut.

   Metode run
    - Menjalankan semua topik dengan memanggil metode start() pada masing-masing objek topik.
    - Menunggu proses setiap topik selesai dengan metode join().

   Metode stop
    - Menghentikan semua topik dengan memanggil metode stop() pada setiap objek topik.

3. File Konfigurasi JSON
   Simulator ini membutuhkan file konfigurasi dalam format JSON, yang mencakup:

    a. Pengaturan Broker:
        - URL broker MQTT.

    - Port broker.
    - Versi protokol MQTT.
b. Pengaturan Klien:
    - Sesi bersih, QoS, interval waktu, dll.
c. Daftar Topik:
    - Tipe topik (single, multiple, atau list).
    - Prefix nama topik.
    - Data payload untuk setiap topik.
    - Rentang angka atau daftar elemen untuk tipe multiple dan list.

4. Proses simulasi
    - File JSON dibaca, dan pengaturan broker, klien, serta topik diekstrak.
    - Simulator membuat objek topik berdasarkan tipe dan konfigurasi masing-masing.
    - Metode run() menjalankan komunikasi pada topik-topik tersebut.
    - Jika diperlukan, semua proses dihentikan dengan stop().