LAPORAN PRAKTIKUM IOT

(MQTT Server)

Disusun oleh

Chenith Siro Fenisia Sirait

11323008

D3 Teknologi Informasi

INSTITUT TEKNOLOGI DEL

FAKULTAS VOKASI

2024/2025

1. **Definisi Class Simulator**
   a. **Konstruktor**

```
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)
```

 settings_file: File konfigurasi JSON yang berisi detail pengaturan.
 self.default_client_settings: Nilai default untuk pengaturan klien seperti:

- clean: Apakah koneksi bersih setiap kali terhubung.
- retain: Apakah pesan akan disimpan oleh broker.
- qos: Tingkat kualitas layanan (0, 1, atau 2).
- time_interval: Interval waktu antar pesan.

 self.topics: Berisi daftar topik yang dimuat melalui fungsi load_topics.

   b. **Fungsi** read_client_settings

```
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)
    )
```

 Membaca pengaturan klien dari dictionary (settings_dict).
 Jika pengaturan tidak ditemukan, digunakan nilai default (default).

   c. **Fungsi load_topics**

```
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
        )
```

 Metode load_topics: Metode ini memuat konfigurasi topik dari file settings JSON dan membuat instance Topic berdasarkan konfigurasi tersebut.
 topics: Sebuah list untuk menyimpan instance topik.
 open(settings_file): Membuka file settings JSON, yang diharapkan berisi berbagai konfigurasi.

 json.load(json_file): Mengurai konten JSON menjadi dictionary Python.
 broker_settings: Membuat instance dari BrokerSettings dengan nilai dari konfigurasi atau nilai default.

d. Membuat Instance Topik

```
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)
```

 Iterasi melalui topik yang dikonfigurasi: Untuk setiap topik yang didefinisikan dalam konfigurasi, metode ini mengambil data yang relevan.
 topic_data: Data yang terkait dengan topik.
 topic_payload_root: Akar payload opsional, default ke dictionary kosong jika tidak ditentukan.
 topic_client_settings: Membaca pengaturan klien yang spesifik untuk topik, dengan default ke pengaturan broker jika tidak disediakan.

e. Tipe Topik

```
    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}
        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))
rn topics
```

Tipe Topik: Tergantung pada TYPE dari topik, metode ini membuat instance topik yang berbeda:

- **single**: Membuat satu topik dengan format /{PREFIX}.
- **multiple**: Membuat beberapa topik dengan format /{PREFIX}/{id} dalam rentang yang diberikan.
- **list**: Membuat beberapa topik dengan format /{PREFIX}/{item} berdasarkan daftar yang diberikan.

f. Menjalankan Simulator

```python
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()
```

- **Metode run**: Metode ini memulai semua topik yang telah dimuat.
- **topic.start()**: Memanggil metode start untuk masing-masing topik, yang mungkin menjalankan proses atau thread terkait.
    - **topic.join()**: Menggabungkan (join) thread, memastikan bahwa main thread menunggu sampai semua topik selesai.

g. Menghentikan Simulator

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

- **Metode stop**: Metode ini menghentikan semua topik yang sedang berjalan.
    - **topic.stop()**: Memanggil metode stop untuk masing-masing topik untuk menghentikan proses atau thread yang terkait.