LAPORAN PRAKTIKUM

Internet of Things

MQTT

Anjelika A. Simamora 11323057

D-III Teknologi Informasi

INSTITUT TEKNOLOGI DEL

FAKULTAS VOKASI

- Bukti berhasil dijalankan

```
PS D:\Sems3\IoT\mqtt-simulator-master> py mqtt-simulator/main.py
Starting: lamp/1 ...
Starting: lamp/2 ...
Starting: air_quality ...
Starting: temperature/roof ...
Starting: temperature/basement ...
Starting: freezer ...
Starting: location ...
[11:34:39] Data published on: lamp/1
[11:34:41] Data published on: temperature/roof
[11:34:42] Data published on: freezer
[11:34:43] Data published on: temperature/basement
[11:34:43] Data published on: air_quality
[11:34:45] Data published on: lamp/2
[11:34:45] Data published on: location
[11:34:47] Data published on: lamp/1
[11:34:48] Data published on: lamp/1
[11:34:49] Data published on: location
[11:34:49] Data published on: freezer
[11:34:49] Data published on: temperature/roof
[11:34:51] Data published on: air_quality
[11:34:51] Data published on: temperature/basement
[11:34:51] Data published on: air_quality
[11:34:55] Data published on: lamp/1
[11:34:56] Data published on: lamp/1
[11:34:57] Data published on: air_quality
[11:34:57] Data published on: lamp/2
[11:34:57] Data published on: location
[11:34:57] Data published on: temperature/roof
[11:34:57] Data published on: freezer
[11:34:57] Data published on: freezer
[11:34:59] Data published on: temperature/basement
[11:35:03] Data published on: lamp/1
[11:35:04] Data published on: lamp/1
```

Step by Step

- Dimulai dengan mengimport kelas yang didefenisikan untuk mengelola pengaturan serta interaksi broker MQTT dalam aplikasi yang sedang dibangun.

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

  - Json : digunakan untuk memuat dan memparsing file konfigurasi JSON
  - Topic : kelas yang menangani setiap topik dalam broker MQTT
  - BrokerSettings dan ClientSettings mendefenisikan pengaturan untuk broker MQTT dan pengaturan klien MQTT.
- Kelas simulator bertanggung jawab untuk mensimulasikan interaksi dengan broker MQTT dan mengelola topik yang terdaftar pada file pengaturan.Metode _init merupakan konstruktor dari kelas simulator.
  - Settings_file merupakan parameter untuk file konfigurasi dalam format JSON yang berisi pengaturan broker dan topik
  - self.default_client_settings: Menyimpan pengaturan default untuk klien MQTT menggunakan kelas ClientSettings, yang mencakup

        - clean: menentukan apakah sesi bersih saat digunakan
        - retain: menentukan apakah pesan yang diterbitkan akan dipertahankan oleh broker
        - qos : menentukan kualitas layanan untuk pengiriman pesan
        - time_interval: menentukan interval wajtu dalam pengaturan
    - self.topics: Menyimpan daftar topik yang dimuat dari file konfigurasi dengan memanggil metode load_topics().

```
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)
```

- Kemudian membaca pengaturan klien menggunakan read_client_settings yang berfungsi untuk membaca pengaturan klien dari dictionary dan mengoversinya menjadi objek clientSettings.Selain itu juga menggunakan settings_dict_get() untuk mengambil niali dari dictionary.

```
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)
    )
```

- Memuat topik dari file konfigurasi JSON dengan fungsi load_topics untuk membuka file konfigurasi JSON, membaca pengaturan broker serta mengonfigurasi pengaturan klien, yang kemudian fungsi ini akan membaca daftar topik yang ada dalam file konfigurasi dan membuat objek topik sesuai dengan tipe topiknya.

```python
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}
                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
```

Membaca dan membuat topik berdasarkan tipe, pada bagian ini bertanggung jawab untuk menentukan tipe topik yang ada dalam file konfigurasi dan membuat topik sesuai dengan tipenya.

- Kemudian diakhiri dengan menjalankan dan menghentikan topik dengan run dan stop.

```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()

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