

# **LAPORAN PRAKTIKUM**

## **INTERNET OF THINGS**

**Tugas Simulator.py**



**Nama : Jessica Anastasya Purba**

**NIM : 11323045**

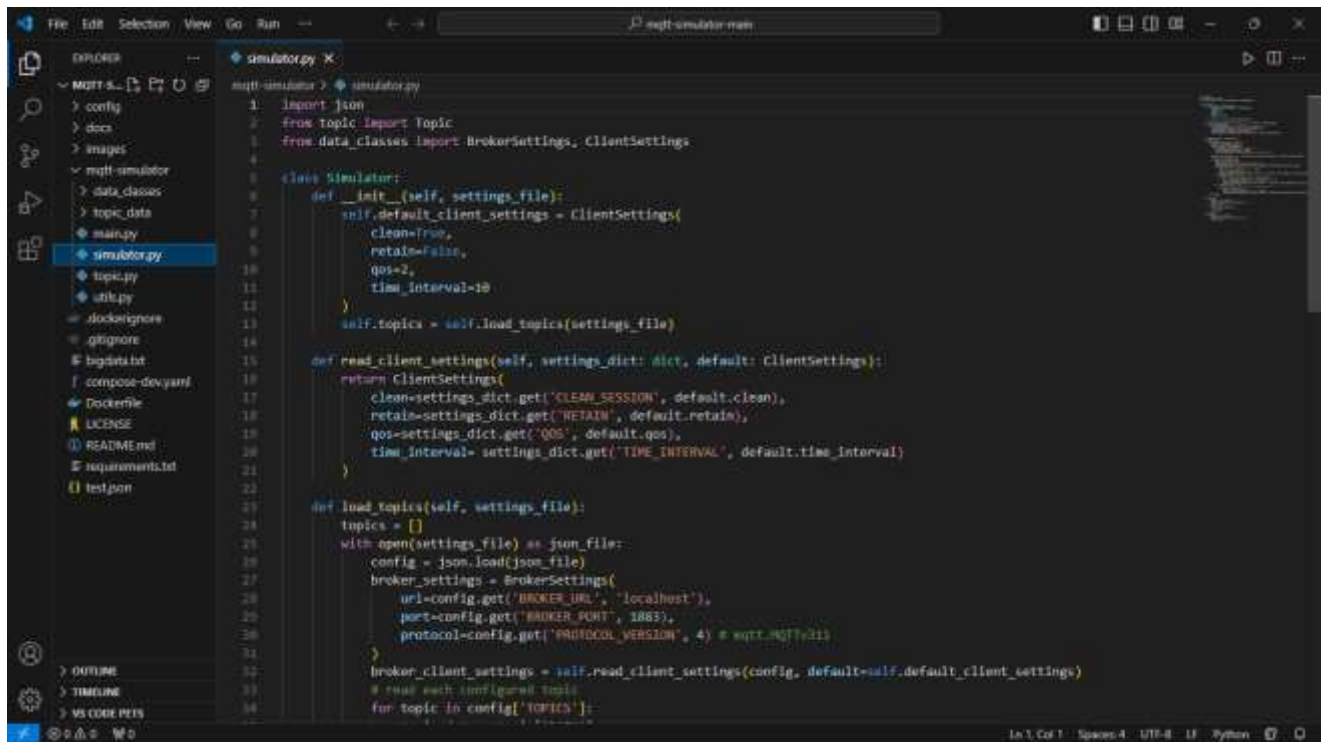
**Prodi : D3 – Teknologi Informasi**

**INSTITUT TEKNOLOGI DEL**

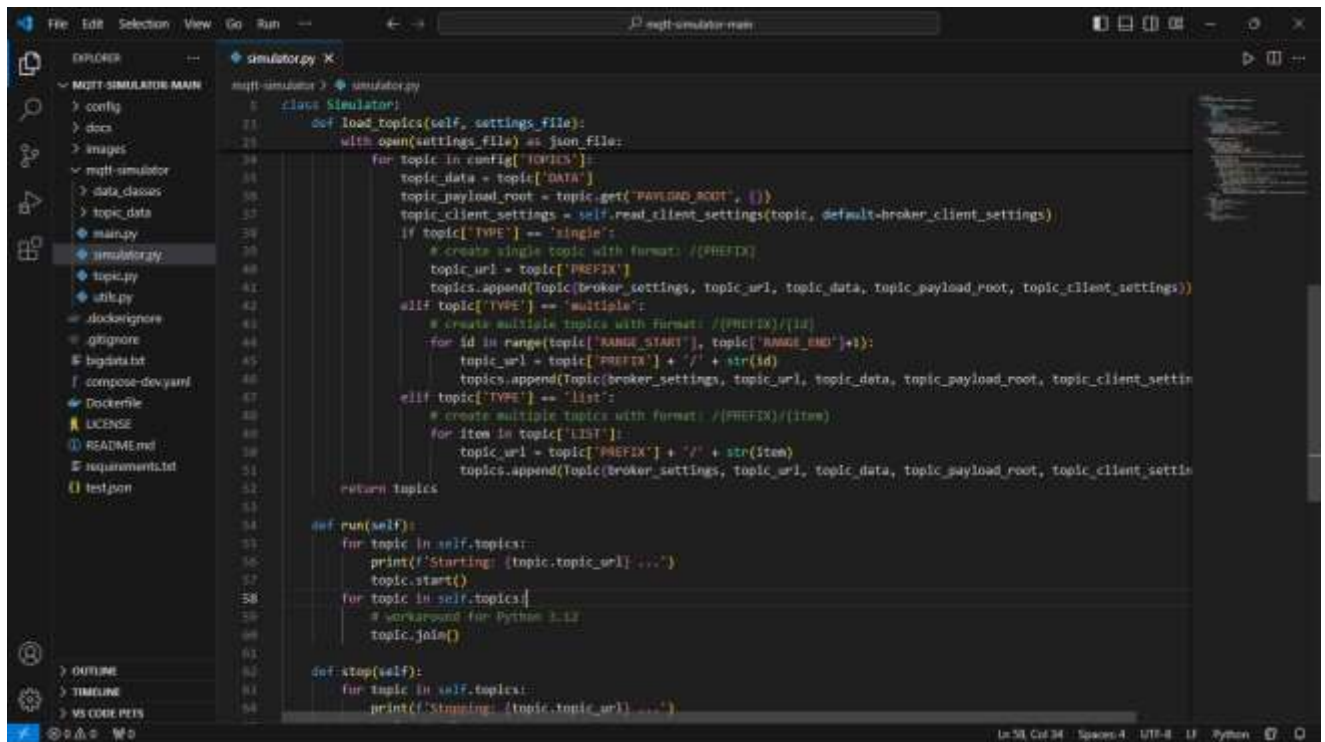
**FAKULTAS VOKASI**

**TAHUN AJARAN 2024/2025**

## Code program:



```
1 import json
2 from topic import Topic
3 from data_classes import BrokerSettings, ClientSettings
4
5 class Simulator:
6     def __init__(self, settings_file):
7         self.default_client_settings = ClientSettings(
8             clean=True,
9             retain=False,
10             qos=2,
11             time_interval=10
12         )
13         self.topics = self.load_topics(settings_file)
14
15     def read_client_settings(self, settings_dict: dict, default: ClientSettings):
16         return ClientSettings(
17             clean=settings_dict.get('CLEAN_SESSION', default.clean),
18             retain=settings_dict.get('RETAIN', default.retain),
19             qos=settings_dict.get('QOS', default.qos),
20             time_interval=settings_dict.get('TIME_INTERVAL', default.time_interval)
21         )
22
23     def load_topics(self, settings_file):
24         topics = []
25         with open(settings_file) as json_file:
26             config = json.load(json_file)
27             broker_settings = BrokerSettings(
28                 url=config.get('BROKER_URL', 'localhost'),
29                 port=config.get('BROKER_PORT', 1883),
30                 protocol=config.get('PROTOCOL_VERSION', 4) * mqtt.MQTT311
31             )
32             broker_client_settings = self.read_client_settings(config, default=self.default_client_settings)
33             # read each configured topic
34             for topic in config['TOPICS']:
```



```
35         def load_topics(self, settings_file):
36             with open(settings_file) as json_file:
37                 for topic in config['TOPICS']:
38                     topic_data = topic['DATA']
39                     topic_payload_root = topic.get('PAYLOAD_ROOT', {})
40                     topic_client_settings = self.read_client_settings(topic, default=broker_client_settings)
41                     if topic['TYPE'] == 'single':
42                         # create single topic with format: /(PREFIX)
43                         topic_url = topic['PREFIX']
44                         topics.append(Topic(broker_settings, topic_url, topic_data, topic_payload_root, topic_client_settings))
45                     elif topic['TYPE'] == 'multiple':
46                         # create multiple topics with format: /(PREFIX)/(id)
47                         for id in range(topic['RANGE_START'], topic['RANGE_END']+1):
48                             topic_url = topic['PREFIX'] + '/' + str(id)
49                             topics.append(Topic(broker_settings, topic_url, topic_data, topic_payload_root, topic_client_settings))
50                     elif topic['TYPE'] == 'list':
51                         # create multiple topics with format: /(PREFIX)/(item)
52                         for item in topic['LIST']:
53                             topic_url = topic['PREFIX'] + '/' + str(item)
54                             topics.append(Topic(broker_settings, topic_url, topic_data, topic_payload_root, topic_client_settings))
55             return topics
56
57     def run(self):
58         for topic in self.topics:
59             print(f'Starting: {topic.topic_url} ...')
60             topic.start()
61         for topic in self.topics:
62             # workaround for Python 3.12
63             topic.join()
64
65     def stop(self):
66         for topic in self.topics:
67             print(f'Stopping: {topic.topic_url} ...')
```

## Penjelasan :

Kode ini merupakan implementasi Python untuk simulator yang berfungsi dengan pengaturan broker MQTT dan daftar topik yang didefinisikan dalam file JSON. Berikut adalah poin-poin utama dari kode tersebut:

### 1. Impor Modul dan Kelas

- Menggunakan modul json untuk membaca file konfigurasi JSON.
- Kelas Topic, BrokerSettings, dan ClientSettings digunakan untuk mengelola topik serta pengaturan broker dan klien.
- 

### 2. Kelas Simulator

- **\_\_init\_\_**: Menginisialisasi simulator dengan pengaturan klien default dan memuat topik dari file JSON.
- **read\_client\_settings**: Membaca pengaturan klien MQTT dari file JSON, dengan nilai default sebagai cadangan jika atribut tertentu tidak ada.
- **load\_topics**: Membaca file JSON, mengatur broker dan klien, lalu membuat daftar topik berdasarkan tipe (single, multiple, atau list) dengan URL yang dihasilkan sesuai pengaturan.
- **run**: Memulai semua topik dalam daftar, menampilkan informasi saat memulai, dan memastikan proses berjalan hingga selesai.
- **stop**: Menghentikan semua topik dan menampilkan pesan saat masing-masing topik dihentikan.

### 3. Konfigurasi JSON

- File JSON berisi informasi seperti URL broker, port, versi protokol, serta daftar topik (TOPICS) dengan berbagai konfigurasi (tipe topik, data, payload, dan lain-lain).

### 4. Format Topik

- **Single:** URL dengan format /PREFIX.
- **Multiple:** URL dalam format /PREFIX/{id} untuk rentang ID tertentu.
- **List:** URL dalam format /PREFIX/{item} untuk elemen yang ada dalam daftar tertentu.

### 5. Fungsionalitas Utama

- Simulator membaca pengaturan, membangun topik, dan menjalankan atau menghentikan proses untuk setiap topik sesuai perintah.

Kode ini berguna untuk mensimulasikan interaksi dengan topik MQTT berdasarkan pengaturan yang fleksibel, mendukung publikasi atau langganan dengan cara yang terstruktur dan otomatis.