```
import re
from typing import NamedTuple
import paho.mqtt.client as mqtt
from influxdb import InfluxDBClient
```

- Re: importeren we voor het verwerken van reguliere expressies die nuttig zijn voor patroonherkenning.
- Namedtuple: Voor het creeren van tuplen met benoemde velden voor een beter leesbare en onderhoudbare code.
- Paho.mqtt.client: Voor het het mqtt protocol om berichten te versturen.
- InfluxDB client: is voor het communiceren met influxDB, om een grote hoeveelheid tijd gestempelde data op te slaan.

```
# WiFi-gegevens
SSID = "******"
PASSWORD = "*******"
```

We stellen onze wifi gegevens in dus naam en wachtwoord.

```
# INFLUXDB-GEGEVENS
INFLUXDB_ADDRESS = '192.168.*.***'
INFLUXDB_USER = '********'
INFLUXDB_PASSWORD = '************'
INFLUXDB_DATABASE = 'serre'

influxdb_client = InfluxDBClient(INFLUXDB_ADDRESS, 8086, INFLUXDB_USER,
INFLUXDB_PASSWORD, INFLUXDB_DATABASE)
```

Dit deel stelt de verbinding met een InfluxDB-server in door de benodigde configuratiegegevens waaronder; ip adres, gebruikersnaam, wachtwoord en de naam van de database. Vervolgens creëren we een clientobject dat we gebruiken om te communiceren met de server.

```
# MQTT-gegevens
MQTT_ADDRESS = "192.168.*.***"
MQTT_PORT = 1883
MQTT_USER = "*****"
MQTT_PASSWORD = "******"
MQTT_CLIENT_ID = "Raspi"
MQTT_TOPIC = "serre/#"
MQTT_REGEX = 'serre/([^/]+)/([^/]+)'
```

We stellen de configuratieparameters in voor het verbinden van een MQTT-client met een MQTT-broker. We geven het ip adres, inloggegevens, de client-ID, het geabonneerde topic en een patroon (MQTT_REGEX) voor het verwerken van berichten.

```
class SensorData(NamedTuple):
    location: str
    measurement: str
    value: float
```

‘Sensordata’ is een benoemde tuple voor het opslaan van sensorinformatie, dit bestaat uit een locatie, een type meting en een waarde.

```
def on_connect(client, userdata, flags, rc):
    """De callback voor wanneer de client een CONNACK-respons van de server ontvangt."""
    print('Verbonden met resultaatcode ' + str(rc))
    client.subscribe(MQTT_TOPIC)
```

De ‘on_connect’ functie is een callback die wordt uitgevoerd wanneer de MQTT-client verbinding maakt met de broker. Het print de resultaat code van de verbinding en abonneert de client op een specifieke topic.

```
def _parse_mqtt_message(topic, payload):
    match = re.match(MQTT_REGEX, topic)
    if match:
        location = match.group(1)
        measurement = match.group(2)
        if measurement == 'status':
            return None
        return SensorData(location, measurement, float(payload))
    else:
        return None
```

De _parse_mqtt_message functie analyseert een MQTT-bericht door het topic te vergelijken met de REGEX. Als het topic overeenkomt, haalt het de locatie en het type meting op. Als de meting geen ‘status’ is, retourneert het een ‘sensorData’ object met de bijbehorende gegevens. Als het topic niet overeenkomt of de meting ‘status’ is, retourneert het ‘none’.

```
# DATA NAAR INFLUXDB
def _send_sensor_data_to_influxdb(sensor_data):
    json_body = [
        {
            'measurement': sensor_data.measurement,
            'tags': {
                'location': sensor_data.location
            },
            'fields': {
                'value': sensor_data.value
            }
        }
    ]
    influxdb_client.write_points(json_body)
```

De _send_sensor _data_to_influxdb functie neem teen ‘Sensordata’ object, en zet de data om in een formaat zodat InfluxDB dat kan begrijpen en schrijft deze data naar de InfluxDB-database met behulp van de ‘influxdb_client.write_points’ methode.

```
# Callback-functie voor het ontvangen van berichten
def on_message(client, userdata, msg):
    """De callback voor wanneer een PUBLISH-bericht wordt ontvangen van de server."""
    print(msg.topic + ' ' + str(msg.payload))
    sensor_data = _parse_mqtt_message(msg.topic, msg.payload.decode('utf-8'))
    if sensor_data is not None:
```

```
        _send_sensor_data_to_influxdb(sensor_data)
```

De ‘on message’ functie is een callback functie die wordt gebruikt om MQTT-berichten te verwerken wanneer ze worden ontvangen van de server. Het decodeert het bericht en analyseert het met behulp van de ‘_parse_mqtt_message’ functie, en stuurt vervolgens de resulterende sensorgegevens door naar de inffluxDB met behulp van de ‘_send_sensor_data_to_influxdb’ functie.

```
#Initialisatie INFLUXDB_database
def _init_influxdb_database():
    databases = influxdb_client.get_list_database()
    if len(list(filter(lambda x: x['name'] == INFLUXDB_DATABASE, databases))) == 0:
        influxdb_client.create_database(INFLUXDB_DATABASE)
    influxdb_client.switch_database(INFLUXDB_DATABASE)
```

De ‘_init_influxdb_database’ functie zorgt ervoor dat de gewenste influxDB-database bestaat door deze aan te maken als deze nog niet bestaat. Vervolgens wordt er naar deze database geschakeld om ervoor te zorgen dat verdere operaties worden uitgevoerd op de juiste database.

```
#Verbinding maken met MQTT-broker
def main():
    _init_influxdb_database()

    mqtt_client = mqtt.Client(MQTT_CLIENT_ID)
    mqtt_client.username_pw_set(MQTT_USER, MQTT_PASSWORD)
    mqtt_client.on_connect = on_connect
    mqtt_client.on_message = on_message

    mqtt_client.connect(MQTT_ADDRESS, 1883)
    mqtt_client.loop_forever()
```

We maken verbinding met de MQTT-broker om berichten te ontvangen. Het initialiseert een influxdb-database en maakt een mqtt-client aan met een specifieke client-ID (gebruikersnaam en wachtwoord) vervolgens definieert het de callback functie voor het verbinden met de broker en het ontvangen van berichten vervolgens maken we verbinding via de opgegeven poort. Tenslotte starten we de loop zodat we blijven luisteren naar berichten.

```
#Startpunt
if __name__ == '__main__':
    print('MQTT naar InfluxDB brug')
    main()
```

We controleren of het script rechtstreeks wordt uitgevoerd . Als dat het geval is, wordt de boodschap "MQTT naar InfluxDB brug" afgedrukt en wordt de ‘main()’ functie uitgevoerd, waarmee de verbinding met de MQTT-broker wordt opgezet en berichten worden ontvangen. Dit zorgt voor een soepele uitvoering van het script als het als standalone wordt gebruikt.