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

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

# 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)

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

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

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)

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

# 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)

# 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)

#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)

#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()

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


#callback functie
client.on_message = on_message
```