In dit document ga ik je uitleggen hoe je data kan versturen van de esp32 naar de raspberry pi

En dit kan laten zien op grafana:

Voor MQTT en grafana werkende te krijgen op mijn raspberry pi heb ik deze turtorial gevolgd:

https://diyi0t.com/visualize-mqtt-data-with-influxdb-and-grafana/

Ik heb hier een project gemakt dat het volgende doet.

Ik lees en Dallas 18b20 temperatuur sensor en een BH1750 in met mijn esp32 (code komt later) en stuur deze code dan op naar de MQTT broker van de raspberry pi waar ik deze inlees en laat zien in de terminal (zie foto hieronder).

```
pi@nicoschepens:~/PythonOefeningen $ python get_MQTT_data.py
MQTT to InfluxDB bridge
Connected with result code 0
home/livingroom/temperature 22
home/livingroom/light 183
home/livingroom/temperature 21
home/livingroom/light 194
home/livingroom/temperature 21
home/livingroom/light 199
home/livingroom/temperature 21
home/livingroom/light 182
home/livingroom/temperature 21
home/livingroom/light 130
```

Ook heb ik een lcd gebruikt om de temperatuur en de gewenste waarde af te drukken. En heb als toepassing op de licht sensor een automatische gordijn bediening gerealiseerd die dat ook op het lcd scherm lat zien of de gordijnen naar boven of naar benden gaan en ook met een indicatie van een led.

De arduino Code:

```
#include <WiFi.h>
#include <PubSubClient.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include <LiquidCrystal.h>
#include <Wire.h>
#include <BH1750.h>
BH1750 lightMeter(0x23);



#define ONE_WIRE_BUS 26
```

Hier zet ik alle bibliotheken dat ik zal nodig hebben en stel ik het adres van de lichtsensor en op welke pin deze zich bevind.

```
const char* ssid = "t[redacted]";// jouw eigen wifi
const char* password = "[redacted]";// u wachtwoord
const char* mqttServer = "192.168.0.121"; //u ip adress van de raspberry pi
const char* light_topic = "home/livingroom/light";
const char* temperature_topic = "home/livingroom/temperature";
const int mqttPort = 1883;
const char* mqttUser = "nico"; // u user vanm MQTT
const char* mqttPassword = "raspberry"; // u wachtwoord van MQTT
const char* clientID = "client_livingroom"; // MQTT client ID


const int RS = 2, EN = 4, d4 = 5, d5 = 18, d6 = 19, d7 = 23;

const int buttonPin1 = 27;      // the number of the pushbutton pin
const int buttonPin2 = 25;      // the number of the pushbutton pin

int red_light_pin= 13;
int green_light_pin = 12;
int blue_light_pin = 14;
int Counter = 18;
int Temp = 0;
int stand = LOW;
int omlaag = LOW;
int omhoog = LOW;
int buttonState1 = 0;
int buttonState2 = 0;
int led1Pin= 32;
int led2Pin= 33;
```

Hier definieer ik alle variabele dat ik ga gebruiken ook de pinnen van het lcd scherm en alle in en uitgangen definieer ik hier.

```
LiquidCrystal lcd(RS, EN, d4, d5, d6, d7);
WiFiClient espClient;
PubSubClient client(espClient);
OneWire oneWire(ONE_WIRE_BUS);
float temp=0.0;
DallasTemperature sensors(&oneWire);
```

Hier word alles gestart.

```
void setup() {
  Serial.begin(115200);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.println("Connecting to WiFi..");
  }
  Serial.println("Connected to the WiFi network");
  client.setServer(mqttServer, mqttPort);
  Wire.begin();
  ;sensors.begin();
  lightMeter.begin();
```

We maken hier verbinding met wifi en starten we ook de temperatuur en de licht sensor.

```
  pinMode(red_light_pin, OUTPUT);
  pinMode(green_light_pin, OUTPUT);
  pinMode(blue_light_pin, OUTPUT);
  pinMode(buttonPin1, INPUT);
  pinMode(buttonPin2, INPUT);
  pinMode(led1Pin, OUTPUT);
  pinMode(led2Pin, OUTPUT);
  lcd.begin(16, 2);
```

We declareren hier de pinmodes dus dat wil zeggen dat we gaan zeggen of het een ingang of uitgang is.

```
void loop() {
  client.loop();

  while (!client.connected()) {
    Serial.println("Connecting to MQTT...");
    if (client.connect(clientID, mqttUser, mqttPassword)){
    {
      Serial.println("connected");
    }}
    else
    {
      Serial.print("failed with state ");
      Serial.print(client.state());
      delay(2000);
    }
  }
```

We maken hier verbinding met de MQTT Broker.

```
sensors.requestTemperatures();
Temp = (sensors.getTempCByIndex(0));
uint16_t lux = lightMeter.readLightLevel();
```

We vragen de waarde op dat de sensor inleest.

```
lcd.setCursor(0, 0);
lcd.print("Temp=");
lcd.setCursor(0, 1);
lcd.print("Set=");
lcd.setCursor(6, 0);
lcd.print(sensors.getTempCByIndex(0));
lcd.setCursor(5, 1);
lcd.print(Counter);
```

We printen de waarde van de temperatuur af en ook de gewenste waarde af op het lcd scherm.

```
buttonState1 = digitalRead(27);
if (buttonState1 == 1){
Counter = Counter + 1;
Serial.println(Counter);
lcd.setCursor(5, 1);
lcd.print(Counter);
}
buttonState2 = digitalRead(25);
if (buttonState2 == 1){
Counter = Counter - 1;
Serial.println(Counter);
lcd.setCursor(5, 1);
lcd.print(Counter);
}
```

We stellen hiermee de gewenste hier in en drukken deze ook af.

```
if (Counter < Temp){
RGB_color(0, 255, 0); // Green
delay(1000);

}
if (Counter > Temp){
RGB_color(255, 0, 0); // Red
delay(1000);
```

We gaan hier de RGB led sturen aan de hand van of de gewenste waarde overschreden is of niet.

```
if (lux < 50 && stand == LOW && omlaag == LOW){
lcd.clear();
lcd.setCursor(0,0);
lcd.print("Gordijn gaat");
lcd.setCursor(0, 2);
lcd.print("Omlaag");
digitalWrite(led1Pin, HIGH);
digitalWrite(led2Pin, LOW);
omlaag = HIGH;
omhoog = LOW;
delay(1000 * 5);
}

if (lux > 50 && stand == LOW && omhoog ==LOW){
lcd.clear();
lcd.setCursor(0,0);
lcd.print("Gordijn gaat");
lcd.setCursor(0, 2);
lcd.print("Omhoog");
digitalWrite(led1Pin, LOW);
digitalWrite(led2Pin, HIGH);
omlaag = LOW;
omhoog = HIGH;
delay(1000 * 5);
}
```

Hiermee gaan we de gordijnen naar boven of naar benden laten gaan.

```
// check if returns are valid and print the sensor data
  if (isnan(Temp) || isnan(lux)) {
  Serial.println("Failed to read");
} else {
  Serial.print("light: ");
  Serial.print(lux);
  Serial.println(" lx");
  Serial.print("Temperature: ");
  Serial.print(Temp);
  Serial.println(" *C");

String hs="Light: "+String((float)lux)+" lx ";
String ts="Temp: "+String((float)Temp)+" C ";


(client.publish(temperature_topic, String(Temp).c_str())); {
  Serial.println("Temperature sent!");
(client.publish(light_topic, String(lux).c_str())); {
  Serial.println("Light sent!");
}

}

client.disconnect();  // disconnect from the MQTT broker
delay(1000*5);
```

We gaan nu de ingelezen waarde van de sensoren naar de MQTT Broker sturen.

```
void RGB_color(int red_light_value, int green_light_value, int blue_light_value)
 {
 digitalWrite(red_light_pin, red_light_value);
 digitalWrite(green_light_pin, green_light_value);
 digitalWrite(blue_light_pin, blue_light_value);
}
```

Hier gaan we de RGB led gaan sturen.

Python code :

```
import paho.mqtt.client as mqtt
```

We importeren de juiste client.

```
MQTT_ADDRESS = '192.168.0.121'
MQTT_USER = 'nico'
MQTT_PASSWORD = 'raspberry'
MQTT_TOPIC = 'home/+/+'
```

We stellen het adres, gebruiker, wachtwoord en onderwerp in

```
def on_connect(client, userdata, flags, rc):
    """ The callback for when the client receives a CONNACK response from the s$
    print('Connected with result code ' + str(rc))
    client.subscribe(MQTT_TOPIC)
```

Deze functie wal geroepen worden wanneer de esp32 connecteert met de raspberry en gaat ook melden of er een fout code is met het connecteren.

```
def on_message(client, userdata, msg):
    """The callback for when a PUBLISH message is received from the server."""
    string_payload = msg.payload.decode("utf-8")
    print(msg.topic+" "+string_payload)
```

Deze functie gaat de ontvangen data printen in de terminal.

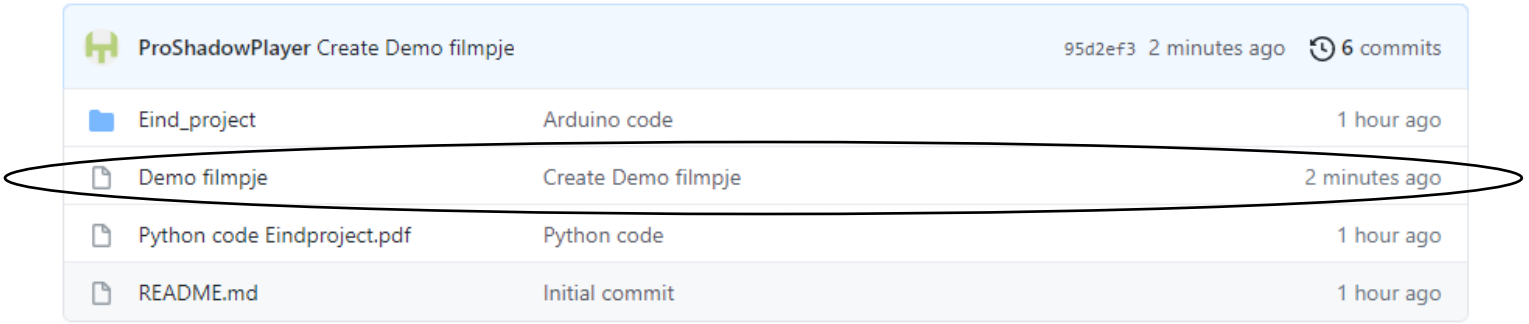
```
def main():
    mqtt_client = mqtt.Client()
    mqtt_client.username_pw_set(MQTT_USER, MQTT_PASSWORD)
    mqtt_client.on_connect = on_connect
    mqtt_client.on_message = on_message

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

Deze functie is de hoofd functie die alles gaat regelen.

```
if __name__ == '__main__':
    print('MQTT to InfluxDB bridge')
    main()
```

En hier gaan we connecteren met de influx bridge.

Zo dat was alle code en voor meer uitleg kunt u altijd naar het demofilmpje bekijken op mijn github pagina.


ProShadowPlayer Create Demo filmpje
95d2ef3 2 minutes ago
6 commits
Eind_project
Arduino code
1 hour ago
Demo filmpje
Create Demo filmpje
2 minutes ago
Python code Eindproject.pdf
Python code
1 hour ago
README.md
Initial commit
1 hour ago