

Glasbox

Thema: ÜK 216

Dokumentinformationen

Dateiname: ProjektDokumentation_Glasbox
Speicherdatum: 22.12.202322.12.2023

Autoreninformationen

Autor: Yara Cordero Ortiz, Giulia Villiger, Anik Stadelmann

E-Mail: yara.cordero@noseryoung.ch,
giulia.villiger@noseryoung.ch,
anik.stadelmann@noseryoung.ch

Inhaltsverzeichnis

1	Einleitung.....	3
2	Informieren.....	3
2.1	Tools.....	3
2.2	Ressourcen.....	3
3	Planen	4
3.1	Arbeitsaufteilung	4
3.2	Task List	4
3.3	Vorstellung.....	4
4	Realisieren.....	5
4.1	Zusammenbau der Komponente	5
4.2	Code	6
4.3	Node-RED	8
4.4	MQTT Explorer (optional)	10
4.5	Probleme	10
5	Kontrollieren.....	12
5.1	Test-Protokoll.....	12
	12

1 Einleitung

Bei unserem Projekt handelt es sich um einen Sensor, der in einer Glasbox die Luftfeuchtigkeit und Temperatur misst. Falls die Temperatur in einem vorgegeben Bereich, den der Benutzer aber ändern kann, ist leuchtet das Lämpchen grün. Wenn es ein bisschen drüber oder drunter ist leuchtet das Lämpchen orange und wenn die Temperatur höher als die Differenz des gewünschten Maximum oder Minimums ist leuchtet es rot.

2 Informieren

2.1 Tools

- GitHub
- Microsoft Word
- MQTT Explorer (optional)
- Arduino IDE
- Node-RED

- ESP32
- Jumper Cables (4 M/F & 4 M/M)
- Temperatur- und Luftfeuchtigkeitssensor
- RGB-Ampel/RGB-Lämpchen
- Breadboard/Steckplatine (optional)
- microUSB Kabel

2.2 Ressourcen

- [Unser GitHub Repository](#)
- [Node-RED UI](#)
- [Website zu verschiedenen Sensoren](#)
- [Website zur Anpassung von Node-RED Dashboard](#)

3 Planen

3.1 Arbeitsaufteilung

Name	Hauptaufgabe
Yara	Node-RED
Giulia	Code in Arduino
Anik	Dokumentation

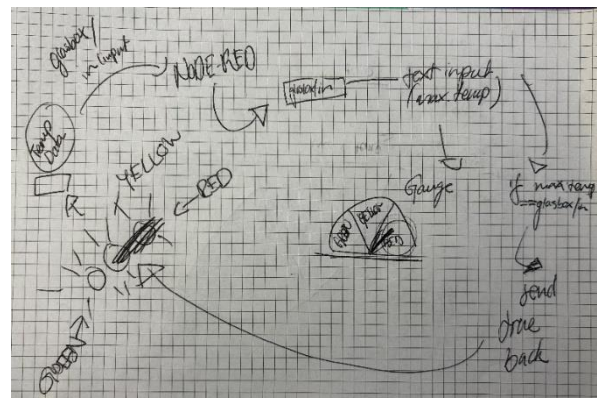
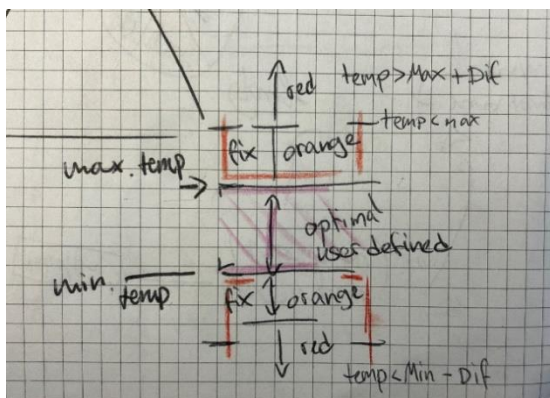
3.2 Task List

Was	Wer	Fertiggestellt am
Dokumentationsfile erstellen und strukturieren	Yara & Giulia	20.12.2023 9:30
Über Anforderungen Informieren	Yara & Giulia	20.12.2023 9:45
Struktur der Datenaufnahme und Ausgabe theoretisch erstellen	Yara & Giulia	20.12.2023 10:20
Hardware Komponente zusammenbauen	Yara & Giulia	20.12.2023 11:00
Min und Max publishen über Node-RED	Yara	20.12.2023 14:00
Node-RED UI designen	Yara	21.12.2023 15:00
Code schreiben	Giulia	20.12.2023 16:00
Code testen, Fehler suchen und debuggen	Giulia	21.12.2023 14:50
Dokumentation schreiben	Yara & Giulia & Anik	22.12.2023 16:15

3.3 Vorstellung

Wir hatten die Vorstellung, dass auf dem Node-RED Dashboard die gewünschten maximalen und minimalen Temperaturwerte eingestellt werden können. Diese Werte sollen dann vom ESP32 verwendet werden, um zu berechnen, ob die gemessene Temperatur diese Grenze überschreitet.

Wenn die Temperatur ausserhalb dieses Bereichs liegt, soll das gelbe LED aufleuchten. Liegt die Temperatur jedoch innerhalb des definierten Minimums und Maximums, soll das grüne LED aufleuchten. Wenn jedoch die gemessene Temperatur mehr als die Differenz des Maximum und das Minimums ist, leuchtet das rote LED auf.



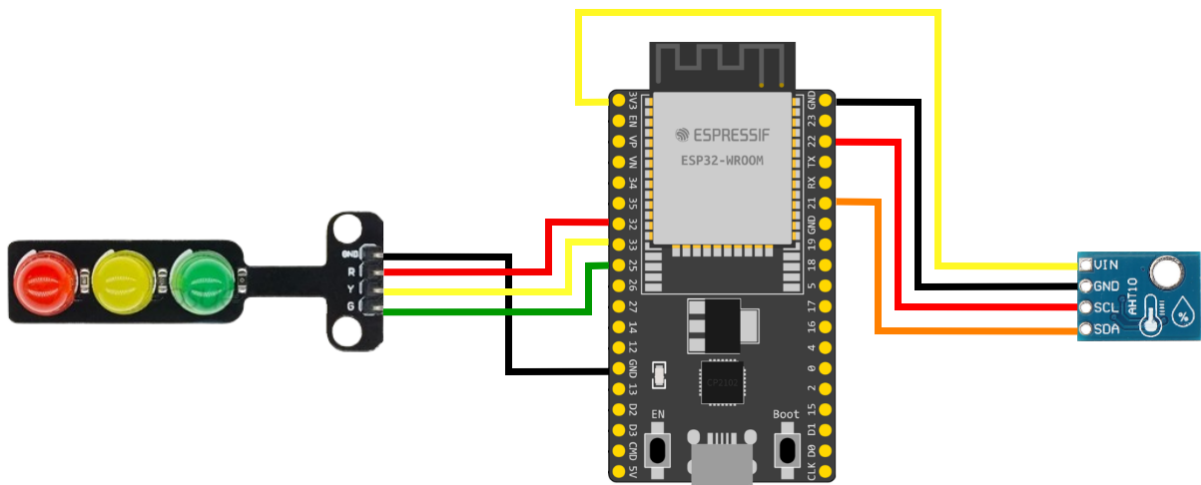
4 Realisieren

4.1 Zusammenbau der Komponente

Benötigte Hardware:

Bezeichnung	Funktion	Anzahl
ESP32	Mikrokontroller	1x
microUSB Kabel	Stromzufuhr und Datentransfer	1x
Breadboard / Steckplatine	Board zur elektronischen Prototypenentwicklung	1x
AHT10	Temperatur- und Luftfeuchtigkeitssensoren	1x
Ampel LED	LED-Licht	1x
GPIO Female-Male Kabel	Verbindung zwischen ESP32 und den Sensor-Pins	5x
GPIO Male-Male Kabel	Bei Verwendung einer Steckplatine	optional

Verbinden sie die Kabel so wie in dieser Abbildung.



Source: draw.io

RGB Ampel: <https://encrypted-tbn3.gstatic.com/images?q=tbn:ANd9GcR4BMu2xd4LXNAMJ5tVLgoW4vKB6vNDCmNt6ie1x8C9pDGPL7I>

ESP32: https://cdn10.bigcommerce.com/s-f2gg5h/product_images/uploaded_images/esp32-wroom-32.jpg

AHT10: <https://encrypted-tbn1.gstatic.com/images?q=tbn:ANd9GcTf602VQk4OuQWCE666RSdWma6YIHfJhIVQIUZZuSeD-68pYW>

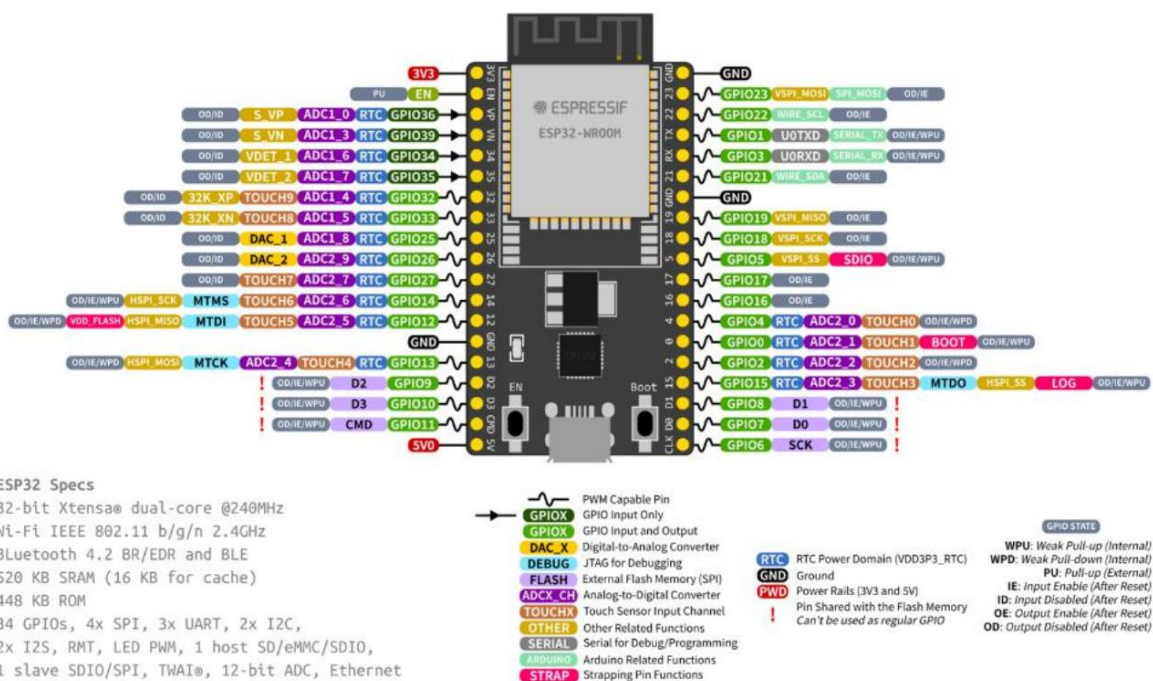
4.2 Code

Der gesamte Code ist in unserem Github Repository zu finden unter `team2_glasbox_Code.ino`. Um diesen Code zu verwenden müssen je nach dem ein paar Sachen geändert werden. Wir verwendeten Arduino D1E.

Wenn Sie andere Pins verwenden für die RGB Lämpchen müssen diese hier angepasst werden. Wichtig ist dass die Pins GPIO Pins sind. Um herauszufinden welche Pins alles GPIO haben

```
const int yellowLED = 33;
const int greenLED = 25;
const int redLED = 32;
```

Hier sehen Sie welche Pins alles GPIO Pins sind.



Source: Dokument Grundlagen01

Hier müssen Sie ihr WLAN angeben und den MQTT Server/Broker den Sie verwenden, je nach dem auch ihre Device ID, dies ist aber kein muss.

```
const char* device_id = "DEFINE_DEVICE_ID_HERE";
const char* ssid = "GuestWLANPortal";
const char* mqtt_server = "noseryoung.ddns.net";
```

Die Topics können Sie auch umbenennen wenn sie wollen, ist aber auch keine pflicht. Solange Sie in Node-RED die Topics richtig angeben.

```
const char* topicTemp = "zuerich/glasbox/temperature/celsius";
const char* topicHum = "zuerich/glasbox/temperature/humidity";
const char* topicMax = "zuerich/glasbox/temperature/max";
const char* topicMin = "zuerich/glasbox/temperature/min";
```

Je nachdem welche Default Range Sie für die optimale Maximum und Minimum Temperatur wollen können Sie diese auch noch hier anpassen. Dies ist aber auch kein muss da diese später auch noch in Node-RED geändert werden können. Diese Werte werden nach dem Start des Programms verwendet bis man sie in Node-RED überschreibt..

```
float Maxtemp = 27;  
float Mintemp = 21;
```

In der void setup Funktion müssen Sie ihren Port anpassen für die MQTT Verbindung und eventuell noch die Baud Rate, diese ist zuständig für die Geschwindigkeit der Datenübertragung. Da diese Dokumentation für den ESP32 dient muss dies aber nicht gemacht werden.

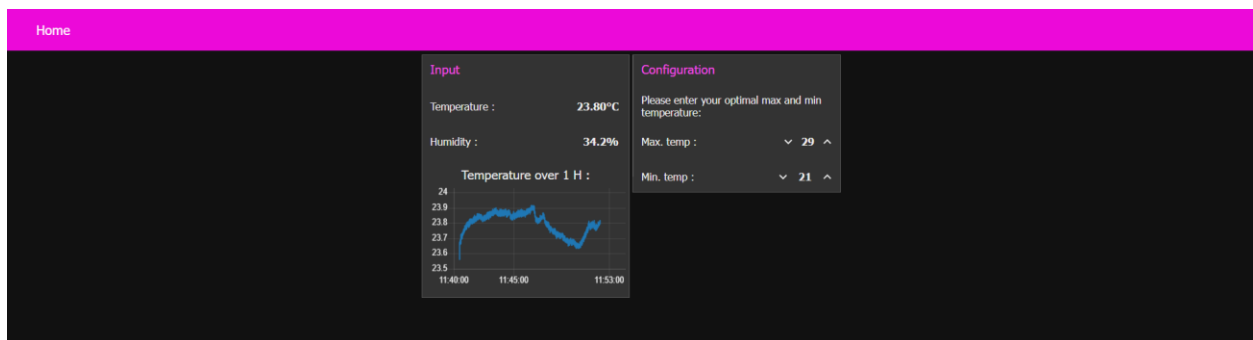
```
Serial.begin(115200); //Baud Rate in Klammer  
setup_aht();  
setup_wifi();  
client.setServer(mqtt_server, 1983); //1983 = MQTT Port  
client.setCallback(callback);
```

Um die Daten zu überprüfen können Sie die Zeichen «/*» und «*/» entfernen und den Serial Monitor in Arduino öffnen. Dort sollte dann die Temperatur, die Luftfeuchtigkeit, Minimal und Maximale Temperatur und deren Differenz ausgegeben werden. Dies ist sehr hilfreich fürs Debuggen.

```
/*  
  Serial.println(temp.temperature);  
  Serial.println(humidity.relative_humidity);  
  Serial.println(Maxtemp);  
  Serial.println(Mintemp);  
  Serial.print("Max Min difference: ");  
  Serial.println(MaxMinDif);  
*/
```

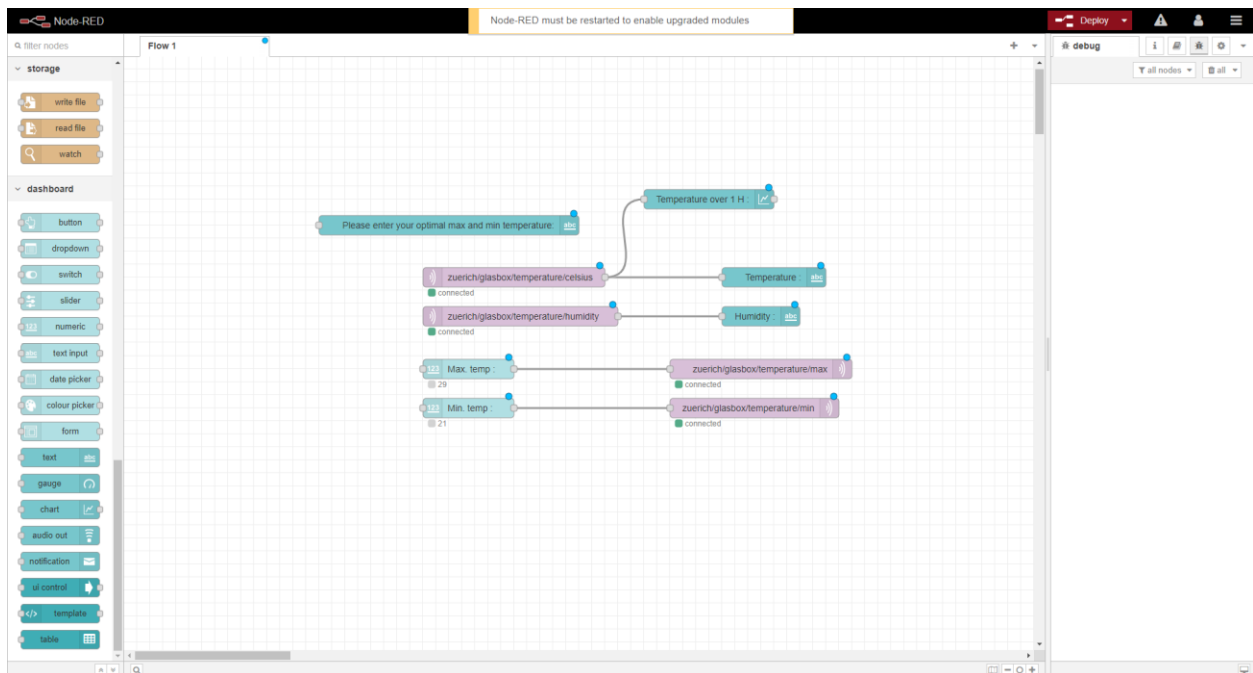
4.3 Node-RED

Wir benutzen Node-RED, um die Daten zu empfangen und sie visuell zu veranschaulichen, aber auch um Daten aufzunehmen, um sie dann an unser ESP32 zu senden. Auf dem Dashboard kann man die aktuelle Temperatur und Luftfeuchtigkeit sehen. Dazu gibt es auch ein Diagramm, die die Temperatur innerhalb einer Stunde, auf einem Liniendiagramm zeigt. Auf der linken Seite ist die Konfiguration zu finden, dort kann man die gewünschte maximal und minimal Temperaturwerte angeben. Diese Werte werden dann auch an das ESP32 gesendet, und dienen zur Bestimmung der Farbe des Lichtes.



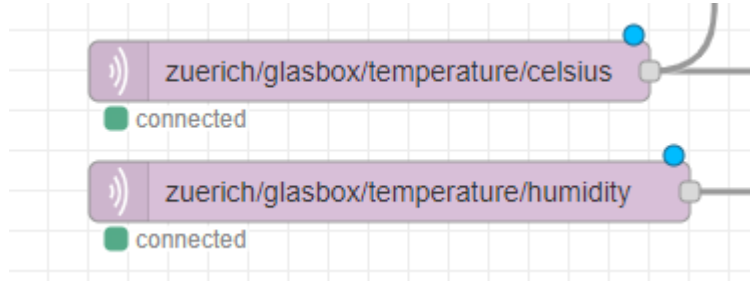
Source: Screenshot Node-RED ui

Um diese Funktionen auf dem Dashboard zu haben, muss man auch im Flow, Nodes hinzufügen und sie so konfigurieren das sie auch miteinander funktionieren.



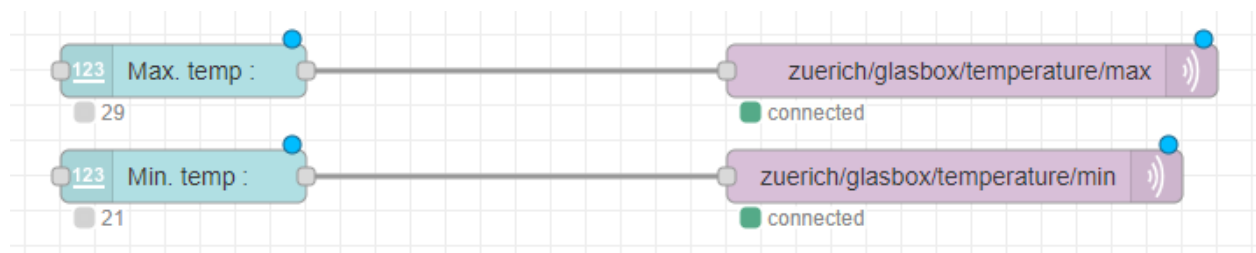
Source: Screenshot Node-RED

In der «MQTT In» Node, abonniert du dich an ein Topic, dies muss aber identisch zu dem Topic sein der du im Code definiert hast. Von da erhältst du die Daten, die dein Sensor aufgenommen hat.



Source: Screenshot Node-RED

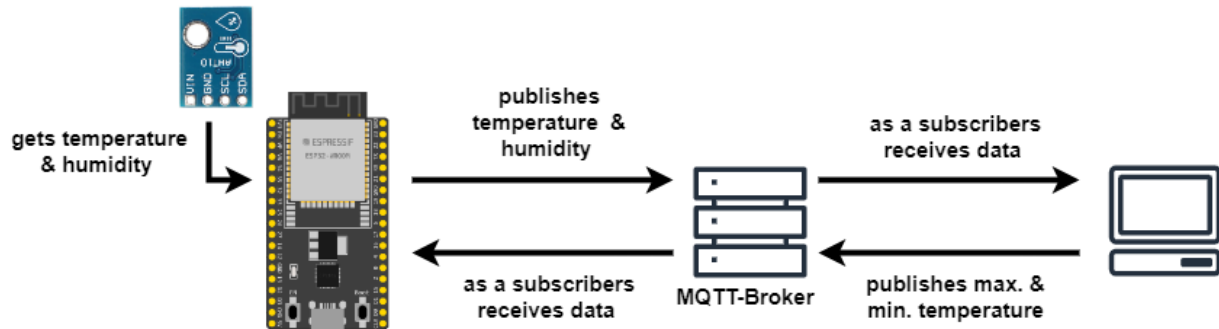
Mit der «numeric» Node kann man Input, vom Dashboard erhalten. Dieser haben wir für unsere max. und min. Temperaturwerte benutzt. Auf der rechten Seite ist der «MQTT out» Node, dieser nimmt die gehaltenen Daten und publiziert sie auf den Server. Man muss darauf achten, dass du den Daten einen neuen Topic Name gibst, der auch Sinn ergibt. Dieser Topic Name wird dann auch im Code referenziert, wenn du die Daten benutzen willst.



Source: Screenshot Node-RED

4.4 MQTT Explorer (optional)

Der MQTT Explorer dient, um zu überprüfen welche Daten veröffentlicht wurden unter welchem Topic. Dies ist ein optionaler Schritt, jedoch sehr hilfreich fürs Debuggen falls sie Probleme haben. Sie sehen also welche Daten an den MQTT-Broker veröffentlicht wurden.

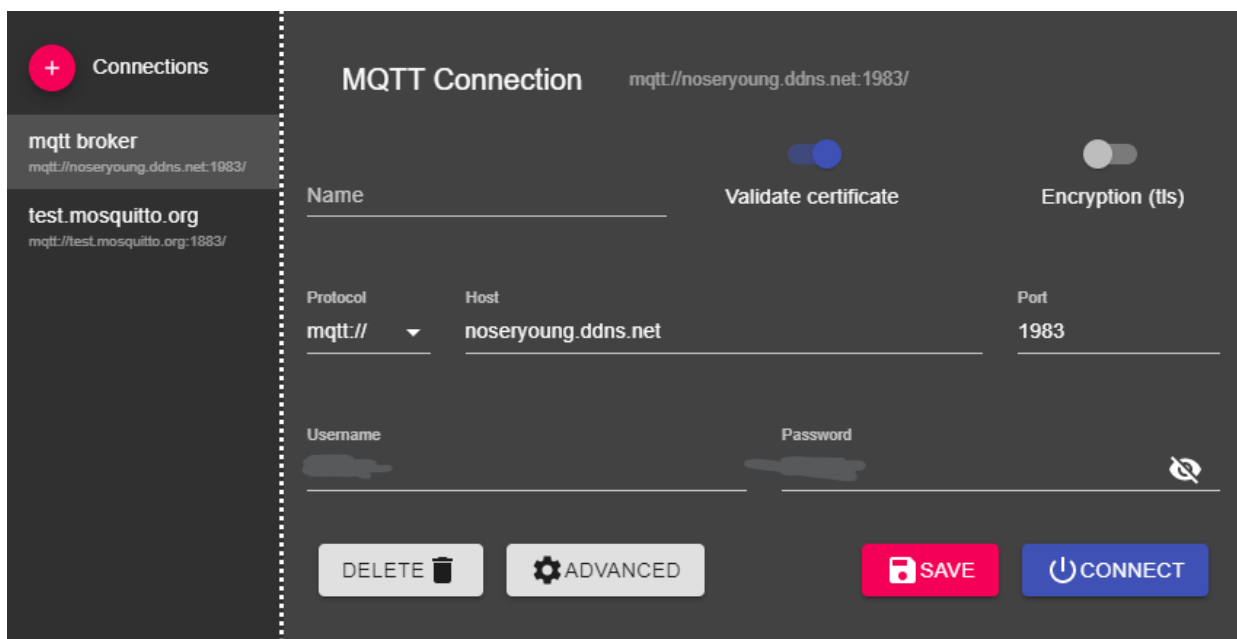


Source: draw.io (icons für MQTT Broker und Computer, Pfeile und Schrift)

ESP32: https://cdn10.bigcommerce.com/s-7f2gg5h/product_images/uploaded_images/esp32-wroom-32.jpg

AHT10: <https://encrypted-tbn1.gstatic.com/images?q=tbn:ANd9GcTlf602VQk4OuQWCE666RSdWma6YIHFJhVQIUZZuSeD - 68pYW>

Passen sie ihren Host und den Port an und geben sie ihr Passwort und Benutzernamen für ihren MQTT-Server ein.



Source: Screenshot MQTT Explorer

4.5 Probleme

Falls sie keine Daten im MQTT Explorer sehen, überprüfen sie ob Sie überall alles richtig geschrieben haben. Im MQTT Explorer sollten sie alle Topics und Daten direkt sehen, jedoch kann es sein, dass es länger dauert bis die Daten vom ESP32 empfangen werden, dies können Sie im Serial Monitor sehen.

5 Kontrollieren

5.1 Test-Protokoll

Testcase	Erwartetes Ergebnis	Erhaltenes Ergebnis	Status
Temperatur ist über dem angegebenen Maximum	Lampe leuchtet gelb oder rot	Lampe leuchtet gelb oder rot	working
Temperatur ist unter dem angegebenen Minimum	Lampe leuchtet gelb oder rot	Lampe leuchtet gelb oder rot	working
Maximum/Minimum wird geändert	Lampe leuchtet gelb oder rot wenn die Temperatur über/unter dem neuen Maximum/Minimum ist, Differenz zwischen Max und Min wird angepasst	Lampe leuchtet gelb oder rot wenn die Temperatur über/unter dem neuen Maximum/Minimum ist, Differenz zwischen Max und Min wird angepasst mit variierender Wartezeit	working
Maximum/Minimum wird nicht geändert wenn ESP32 neu am Strom angeschlossen wird	Default Werte vom Code werden verwendet	Default Werte vom Code werden verwendet	working

