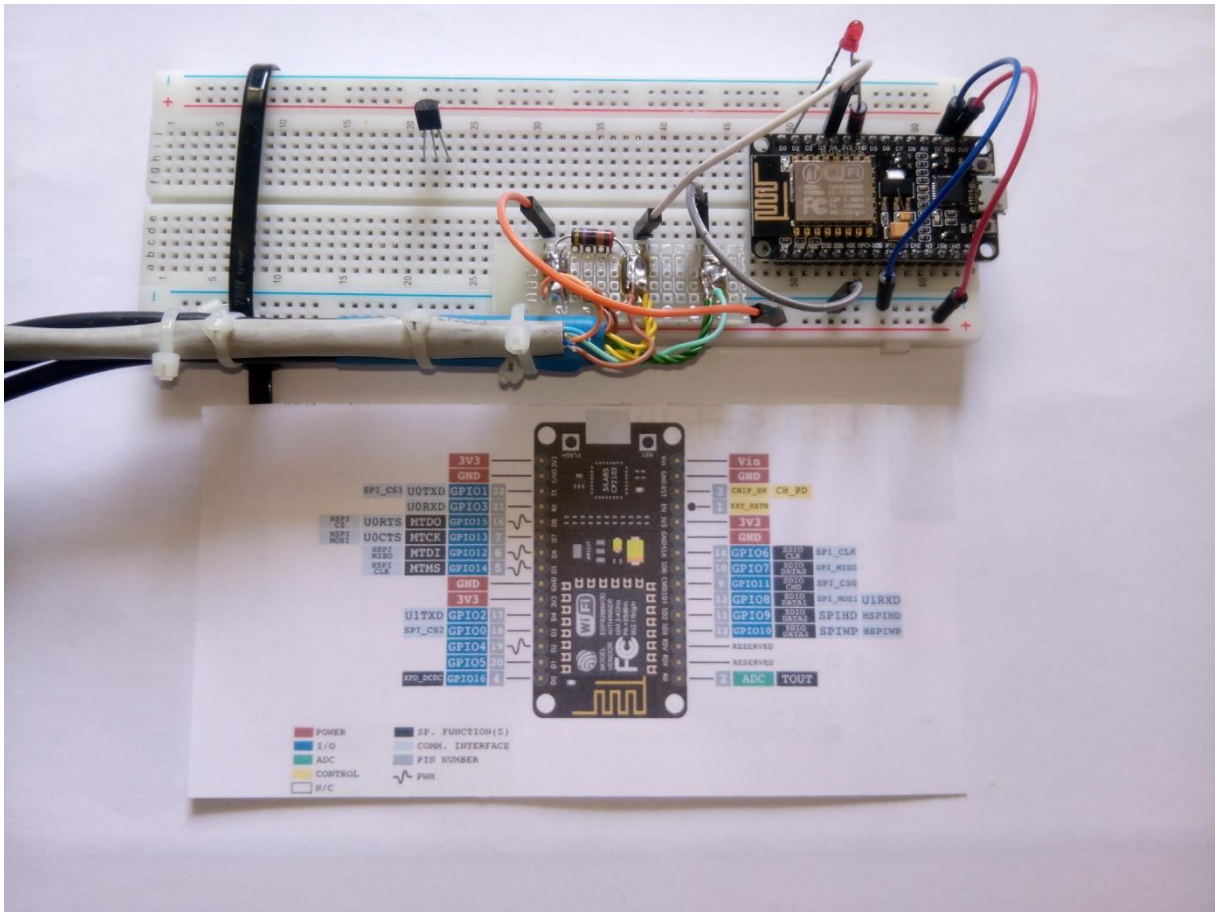
Datensammler

ESP8266**DS18B20**SQL-Datenbank

Temperaturüberwachung einer Zentralheizung
mit Vorlauftemperatur,
Rücklauftemperatur
und Warmwasser
(natürlich auch erweiterbar)



Autor: Waldemar Breiling DJ1DZ Aufgestellt am 20.04.2017

Inhaltsverzeichnis

1. Vorwort

Zur ersten Überlegung Temperaturdaten zu sammeln stellt sich die Frage: was soll gemessen werden – Zimmertemperatur, Raumtemperatur, Wassertemperatur, Kühlschrank oder irgendwelche Gegenstände die temperaturüberwacht werden sollen? In unserem Fall ist es die Zentralheizung, konkret die Vorlauftemperatur und zum Vergleich die Rücklauftemperatur. Und wenn wir schon mal dabei sind, können wir auch noch die Warmwasserleitung mit einbeziehen.

Jetzt will ich aber nicht ständig in den Keller gehen, um irgendwelche Temperaturen aufzuschreiben und diese mühevoll in einer Grafik darzustellen. Das soll dann schon alles automatisch ablaufen.

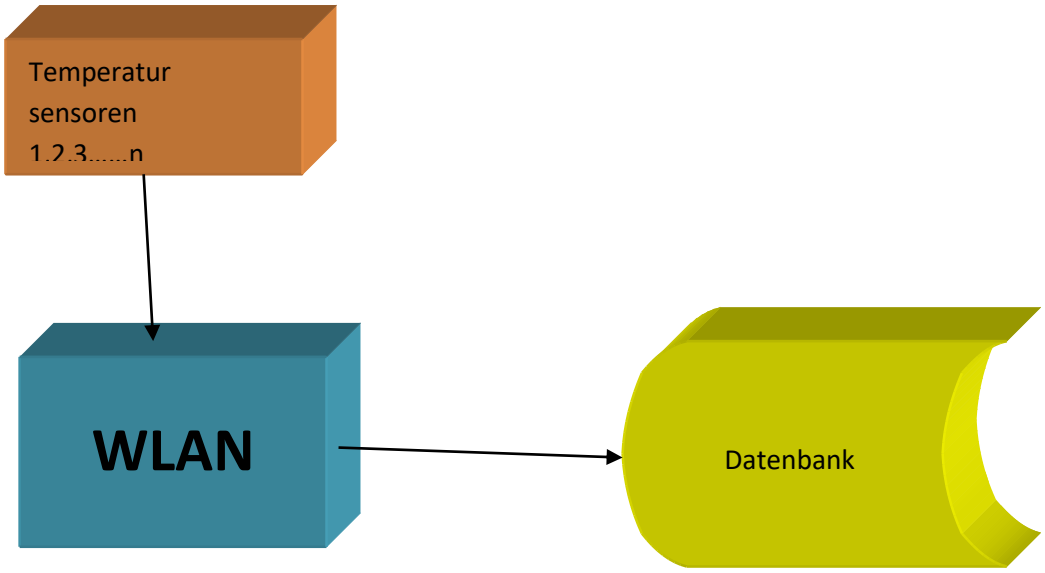
Wie, wird im folgenden Kapitel dargestellt.

2. Zielsetzung

Überwacht werden soll eine Zentralheizung. Temperatursensoren sollen an drei Leitungen angeschlossen werden.

1. Vorlauftemperatur
2. Rücklauftemperatur
3. Warmwasserleitung
4. Wer es genau machen will, kann auch noch die Abgas- und Zulufttemperatur einbeziehen.

Die gemessenen Werte sollen in regelmäßigen Zeitintervallen per Wlan in eine geeignete Datenbank übertragen werden. Diese Datenbank wird die Möglichkeit zur grafischen Darstellung der erfassten Temperaturwerte anbieten können. Und weil wir so energiebewusst sind sollen alle angeschlossene Gerätschaften nur mit minimaler Leistung ihren Dienst verrichten können.

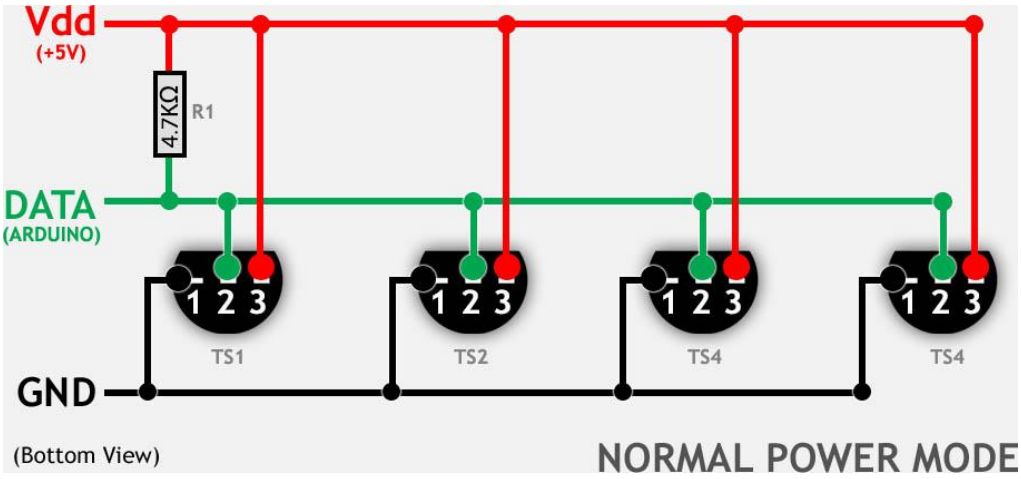


3. Notwendigkeiten.

Listen wir mal erst unsere Hardware auf:

1. Temperatursensoren min 3 Stück DS18B20
2. Widerstand 4,7 kOhm, 330 Ohm
3. LED rot
4. Kabel 3x 0,14mm²
5. NodeMcu Lua ESP8266 mit CP2102 WIFI
6. Laborsteckboard 840 Kontakte
7. Raspberry Pi

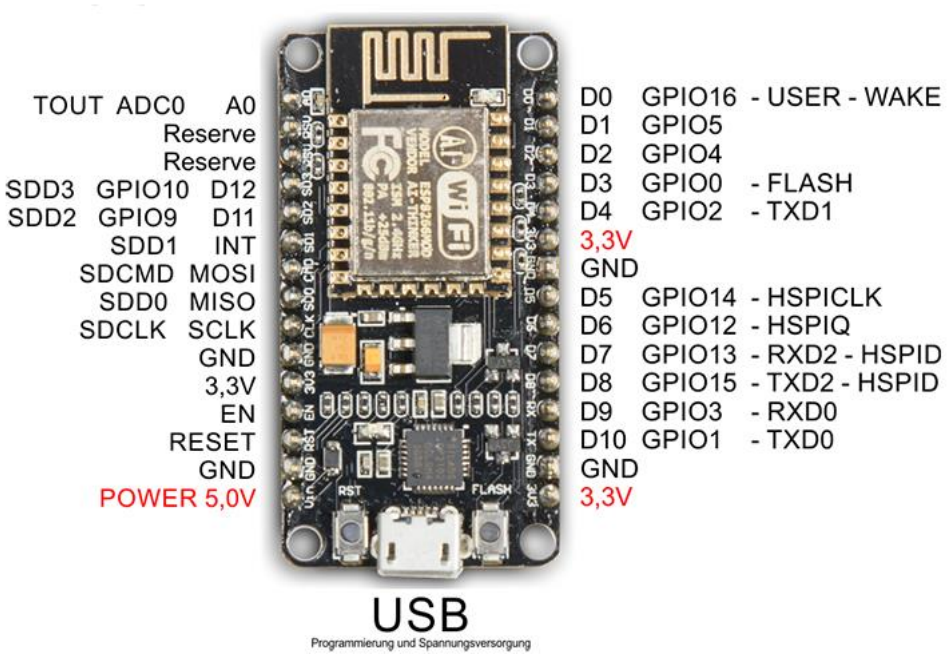
Der DS18B20 ist ein „ONE_WIRE_BUS Device“, dh. die Temperaturdaten und auch die Steuerbefehle werden nur über eine Leitung übertragen. Zusammen mit der Stromversorgung (+3,3V , Masse) sind es also 3 Anschlüsse, die angeschlossen werden müssen. Die Besonderheit ist, dass alle Devices parallel auf dieser Leitung liegen und die Leitungslänge keine große Rolle spielt. Maximale Spannung ist 3-5V. Der ESP8266 bietet 3,3V



Zur eigentlichen Datenerfassung benutzen wir den NodeMcu Lua ESP8266. Auf dieser Arduino Basis befindet sich ein komplettes Wlan-System. Der Vorteil liegt auf der Hand, die ermittelten Daten stehen direkt zur Verfügung und können mit Datum und Uhrzeit in die Datenbank eingepflegt werden.

Aber Achtung! Die Anschlussbelegung stimmt nicht mit den Bezeichnungen des Arduinos überein.

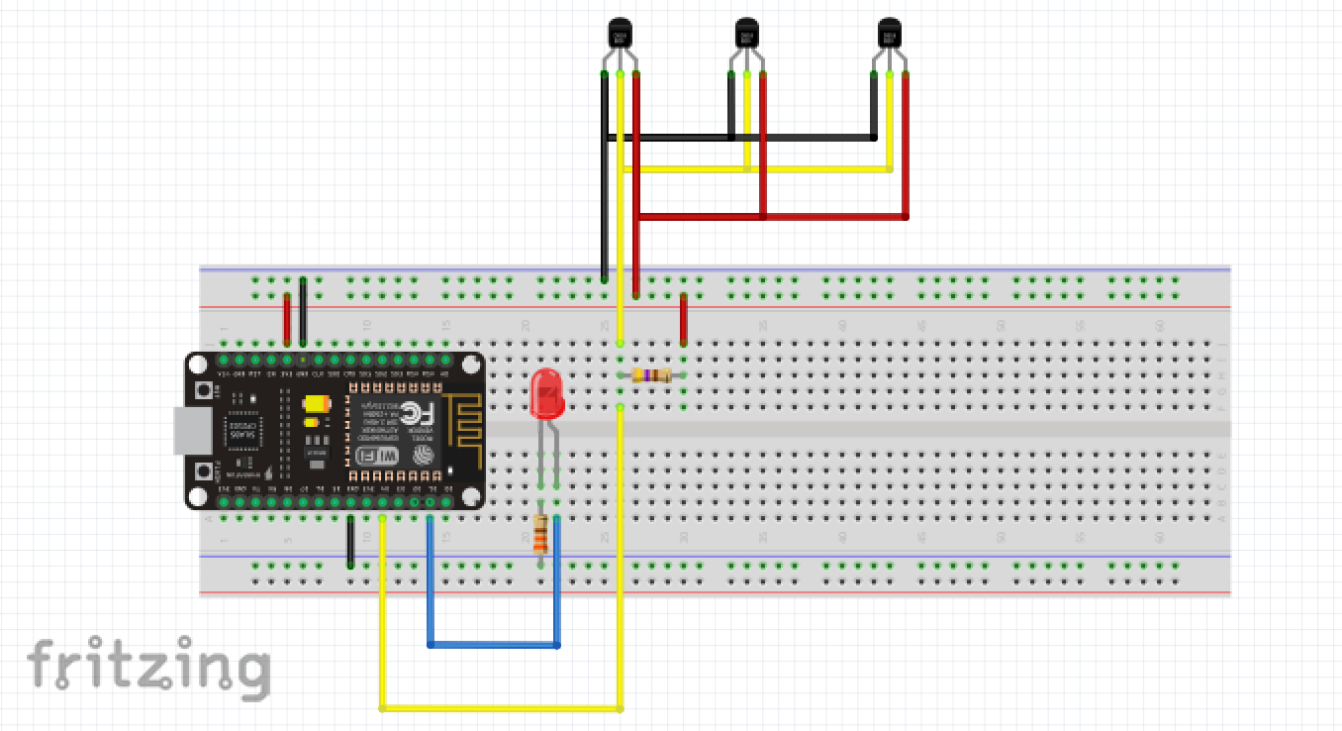
(in der Programmierung werden die GPIO Bezeichnungen benutzt)



Über die USB-Schnittstelle (5V) wird der ESP8266 mit Strom versorgt und gleichzeitig auch programmtechnisch bedient. Die Stromaufnahme in unserem Fall liegt bei maximal 50mA.
Und weil wir im Stromverbrauch so geizig sind (alle Geräte sind ständig in Betrieb), benutzen wir für die Datenbank einen Raspberry Pi Version 3. Hier liegt die Stromaufnahme bei ca. 500mA. Voraussetzung ist natürlich, dass der Wlan-Router nicht ausgeschaltet wird, um den Datenfluss nicht zu unterbrechen.

Für den nötigen Aufbau benutzen wir ein Experimentier-Board mit 840 Kontakten (kleinere Boards z.B. 400 Kontakte sind natürlich auch zu gebrauchen).

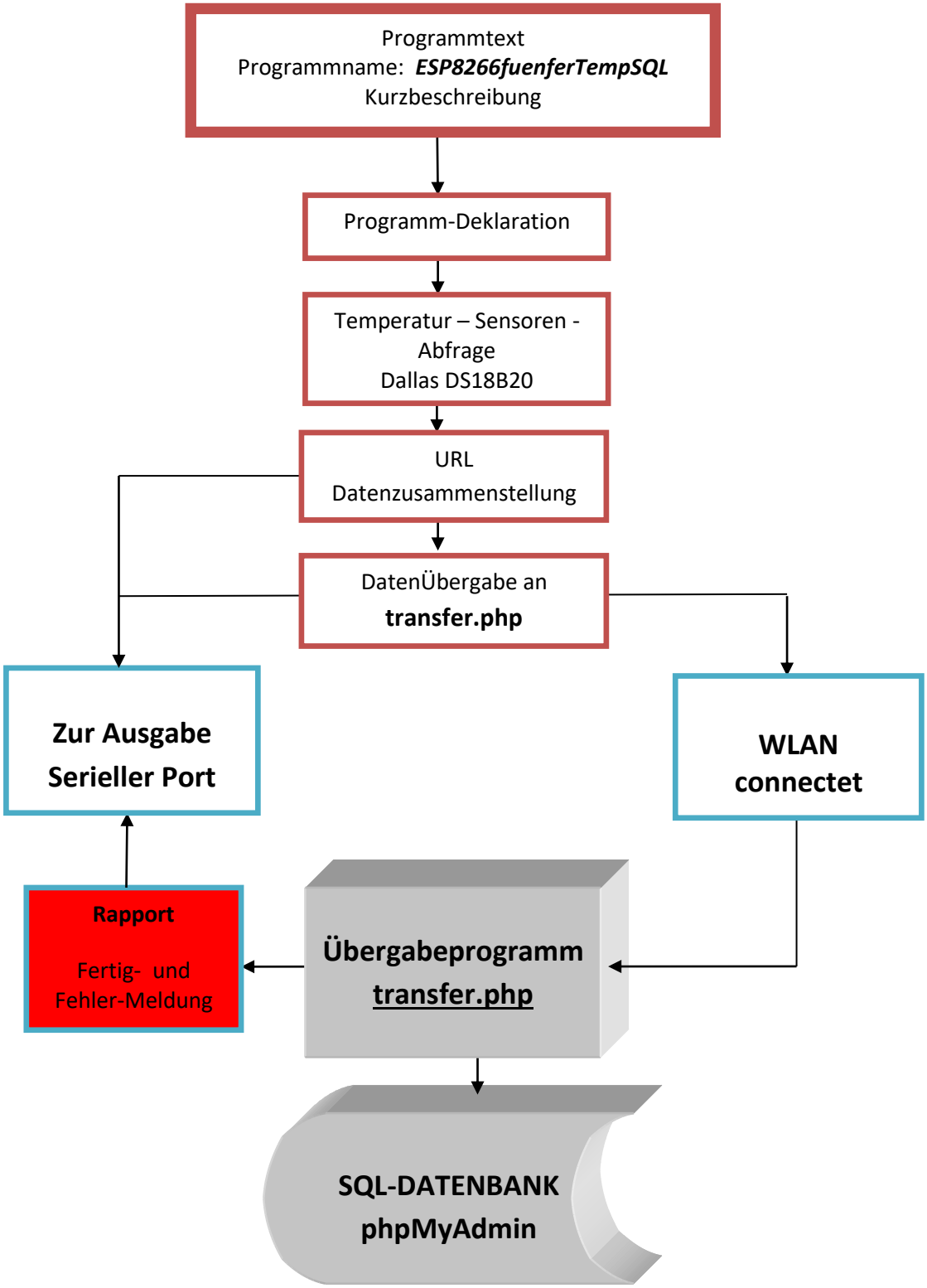
Auf diesem Board gesteckt sieht es dann so aus.



4. Datenfluss

Für die Programmierung des esp8266 benutzen wir, wie auch sonst für den Arduino nano, mini, uno, usw. , das Tool „ARDUINO IDE“- ist kostenfrei bei https://github.com/arduino/Arduino/ runter zu laden und die Installationsanweisung ist auch dabei.

Der Programmablaufplan



5. Software ESP8266fuenfTempSQL.ino

```
/************************************************************************
 * Prg.-Name ESP8266fuenfTempSQL - Zusammengestellt bei DJ1DZ           *
 * Dieses Programm verbindet sich mit dem Heimnetz per Wlan.            *
 * Anschließend werden Sensoren abgefragt.                              *
 * Die Ergebnisse der Sensorenabfragen werden in die Eingabesequenz zur Übernahme in  *
 * die MySQL-Datenbank eingebettet.                                     *
 * Mit Aufruf dieser Sequenz werden die Sensordaten in die Datenbank eingepflegt.     *
 * Beispiel: http://localhost:80/test/transfer.php?T1=-15,25&T2= 26&key=deinpasswort  *
 * Parallelausgabe siehe Monitor.                                       *
 ************************************************************************/
#include <ESP8266WiFi.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#define TEMPERATURE_PRECISION 12
#define ONE_WIRE_BUS 2                  // Digitalport Pin D4 definieren

OneWire oneWire(ONE_WIRE_BUS);          // Ini oneWire instance
DallasTemperature sensors(&oneWire);    // Dallas Temperature Library für Nutzung der
                                        //oneWire Library vorbereiten
DeviceAddress tempDeviceAddress;        // Verzeichniss zum Speichern von Sensor Adressen
const char* ssid     = "WLAN-1DZ";      //eigenen Wlan-Router  einfügen!
const char* password = "421234567850";  //eigenes Router-Passwort einfügen!
const char* host = "192.168.178.56";    //eigene IP-Nr. für raspberry pi 3 eintragen!
int count = 0;
int wlancontrol = 5;                    //Digitalport D1 definieren
float T1, T2, T3, T4, T5;
int numberOfDevices;                    // Anzahl der gefundenen Sensoren
//-----------------------------------------------------------------------------------------
void setup() {
  Serial.begin(9600);
  delay(10);
                                        // Verbindungsaufbau zu einem WiFi-Netzwerk
 pinMode(wlancontrol, OUTPUT);          // Digital Pin 5 - an wenn wifi verbunden.
  Serial.println();
  Serial.println();
  Serial.println(ssid);
  Serial.print("Connecting to ");
```

```
 /* Der ESP8266 wird als WiFi-Client gesetzt, ansonsten würde er standardmäßig
 versuchen, sowohl als Client als auch als Access Point zu fungieren Das könnte dazu führen,
dass  anderen WiFi-Geräten auf dem WiFi-Netzwerk Netzwerk-Probleme verursachen.
 */

 WiFi.mode(WIFI_STA);
 WiFi.begin(ssid, password);

 while (WiFi.status() != WL_CONNECTED) {
  delay(1000);
  Serial.print(".");
 }

 Serial.println("");
 Serial.println("WiFi connected");
 digitalWrite (wlancontrol,HIGH);
 Serial.println("IP address: ");
 Serial.println(WiFi.localIP());
 sensors.begin();
 numberOfDevices = sensors.getDeviceCount();

  Serial.print("Habe ");
  Serial.print(numberOfDevices, DEC);
  Serial.println(" Sensoren gefunden.");
 Genauigkeit();
}
//--------------------------------------------------------------------------------
void loop() {

 Serial.print("connecting to ");
 Serial.println(host);

 // Verwenden der WiFiClient-Klasse, um TCP-Verbindungen zu erstellen

 WiFiClient client;
 const int httpPort = 80;
 if (!client.connect(host,httpPort)) {
   digitalWrite (wlancontrol,LOW);
  Serial.println("connection failed");
 }
```

```
 // Wir erstellen nun einen URL für die Anfrage
   Dallastemp();
   String url = "/transfer.php";   //Übergabe via Raspberry root/var/www/html/transfer.php
   url += "?T1="+(String)T1;
   url += "&T2="+(String)T2;
   url += "&T3="+(String)T3;
   url += "&T4="+(String)T4;
   url += "&T5="+(String)T5;
   url += "&pass=deinpasswort";

 Serial.print("Requesting URL: ");
 Serial.println(url);

 // Damit wird die Anfrage an den Server gesendet

 client.print(String("GET ") + url + " HTTP/1.1\r\n" +
              "Host: " + host + "\r\n" +
              "Connection: close\r\n\r\n");
 unsigned long timeout = millis();
 while (client.available() == 0) {
  if (millis() - timeout > 5000) {
    Serial.println(">>> Client Timeout !");
    client.stop();
//    return;
  }
 }

 // Liest  alle Zeilen der Antwort vom Server und druckt  sie auf Serial

 while(client.available()){
  String line = client.readStringUntil('\r');
  Serial.print(line);
 }

 Serial.println();
 Serial.println("closing connection");
 Serial.println("-----------------------------------------------------");
 delay(120000);     //alle 2 Minuten
}
//---------------------------------------------------------------------
float Dallastemp()
```

```
{
  // Aufruf der Funktion sensors.requestTemperatures()
  // Dadurch werden alle Temperatur-Werte abgefragt.
  Serial.print("Abfrage der Temperatur... ");
  sensors.requestTemperatures();
  Serial.println("DONE");

  // Ausgabe der Daten für jeden Sensor
  for(int i=0 ;i<numberOfDevices; i++) {
    float tempC = sensors.getTempCByIndex(i);
      if (i==0){ T1=tempC;}
      if (i==1){ T2=tempC;}
      if (i==2){ T3=tempC;}
      if (i==3){ T4=tempC;}
      if (i==4){ T5=tempC;}
    Serial.print("Sensor T");
    Serial.print(i+1, DEC);
    Serial.print(" hat grad Celsius: ");
    Serial.println(tempC);
  }
  delay(1000);
}
//--------------------------------------------------------------
float Genauigkeit()
  // Setzen der Genauigkeit
 {  for(int i=0 ;i<numberOfDevices; i++) {
    if(sensors.getAddress(tempDeviceAddress, i)) {
      sensors.setResolution(tempDeviceAddress, TEMPERATURE_PRECISION);
      Serial.print("Sensor T");
      Serial.print(i+1);
      Serial.print(" hat eine genauigkeit von ");
      Serial.println(sensors.getResolution(tempDeviceAddress), DEC);
    }
  }
  Serial.println("");
}
```

Und so könnte die serielle Ausgabe auf dem Monitor aussehen.

```
WLAN-1DZ
Connecting to ..
WiFi connected
IP address:
192.168.178.108
Habe 5 Sensoren gefunden.
Sensor T1 hat eine Genauigkeit von 12
Sensor T2 hat eine Genauigkeit von 12
Sensor T3 hat eine Genauigkeit von 12
Sensor T4 hat eine Genauigkeit von 12
Sensor T5 hat eine Genauigkeit von 12

connecting to 192.168.178.55
Abfrage der Temperatur... DONE
Sensor T1 hat Grad Celsius: 17.25
Sensor T2 hat Grad Celsius: 17.06
Sensor T3 hat Grad Celsius: 16.81
Sensor T4 hat Grad Celsius: 17.06
Sensor T5 hat Grad Celsius: 17.69
Requesting URL:
/transfer.php?T1=17.25&T2=17.06&T3=17.69&T4=16.81&T5=17.06&pass=deinpasswort
HTTP/1.1 200 OK
Date: Wed, 19 Apr 2017 16:29:18 GMT
Server: Apache/2.4.10 (Raspbian)
Vary: Accept-Encoding
Content-Length: 125
Connection: close
Content-Type: text/html; charset=UTF-8

<h1>T1 ist 17.25 und T2 ist 17.06 und T3 ist 17.69 und T4 ist 16.81 und T5 ist 17.06</h1>
Neuer Datensatz mit id 7509 angelegt
closing connection
-------------------------------------------------------
```

Und so sieht die Transfer.php auf dem Raspberry pi aus, mit dem entsprechenden Pfad root/var/www/html/transfer.php für 5 Meßstellen oder auch einfach den einen oder anderen Sensor weglassen.

```
<?php
$T1=$_GET['T1'];
$T2=$_GET['T2'];
$T3=$_GET['T3'];
$T4=$_GET['T4'];
$T5=$_GET['T5'];

$pass=$_GET['pass'];

echo"<h1>T1 ist $T1 und T2 ist $T2 und T3 ist $T3 und T4 ist $T4 und T5 ist $T5$

error_reporting(E_ALL);
$pdo = new PDO('mysql:host=localhost;dbname=Raspi', 'root', $pass);

$statement = $pdo->prepare("INSERT INTO `messstelle1` (vorlauf, ruecklauf,wasse$
//$statement->execute(array('info@php-einfach.de', 'Klaus', 'Neumann'));
$statement->execute();
 echo $statement->errorInfo()[2];
$neue_id = $pdo->lastInsertId();
echo "Neuer Datensatz mit id $neue_id angelegt";
?>
```

6. Datenbank und Auswertung

Zur Installation der Datenbank kann natürlich auch der Desktop-PC oder Laptop etc. benutzt werden. Wir haben uns aber für den Raspberry Pi 3 entschieden weil die Stromaufnahme nur max 500mA beträgt und da diese Geräte immer im Einsatz sein müssen (es werden ja alle 2 Minuten Daten übermittelt) ist der Raspi erste Wahl. Hinzu kommt, dass er WIFI an Bord hat und letztlich kann er ja auch noch andere Aufgaben im häuslichen Wlan-Netz ausführen.

Wem die 500mA noch zu viel sind kann die Datenbank auch in eine Cloud verlegen aber in der Regel sind diese kostenpflichtig (meist Testversionen sind kostenlos).

MYSQL ist eine der beliebtesten Datenbanken. Sie lässt sich gut mit PHP (unsere fransfer.php) ansprechen
Die Installation ist in wenigen Schritten erledigt.
Wir öffnen auf dem Raspian-Desktop die Shell und scheiben den Befehl

sudo apt-get install mysql-server mysql-client php5-mysql

durch diesen Befehl wird MYSQL auf dem Raspberry Pi installiert.
Als nächstes müsst wir noch 2x ein Root Passwort eingeben um es festzulegen.
Anschließend ein restart ausführen um alles abzuschließen:
MYSQL ist nun erfolgreich installiert.
Zum Abschluss noch

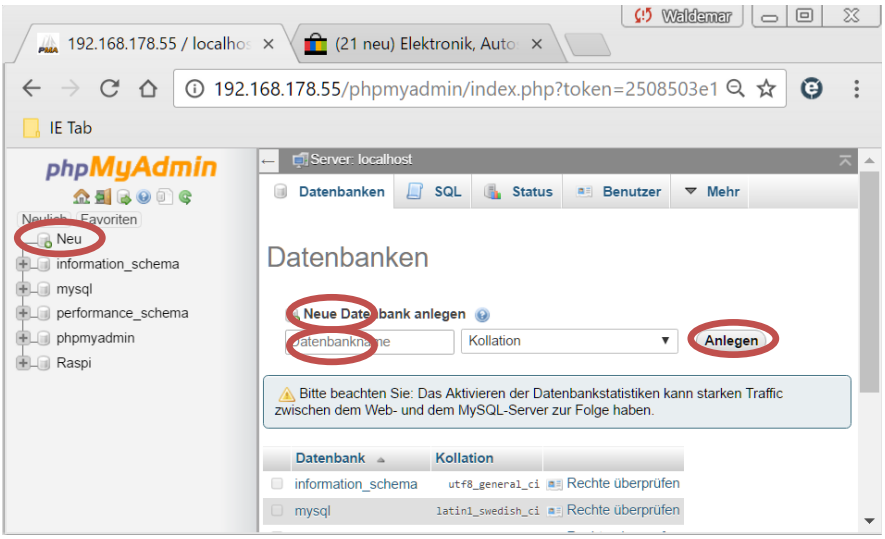
sudo restart

fertig

Wird die Datenbank gestartet erscheint das Anmeldefenster.

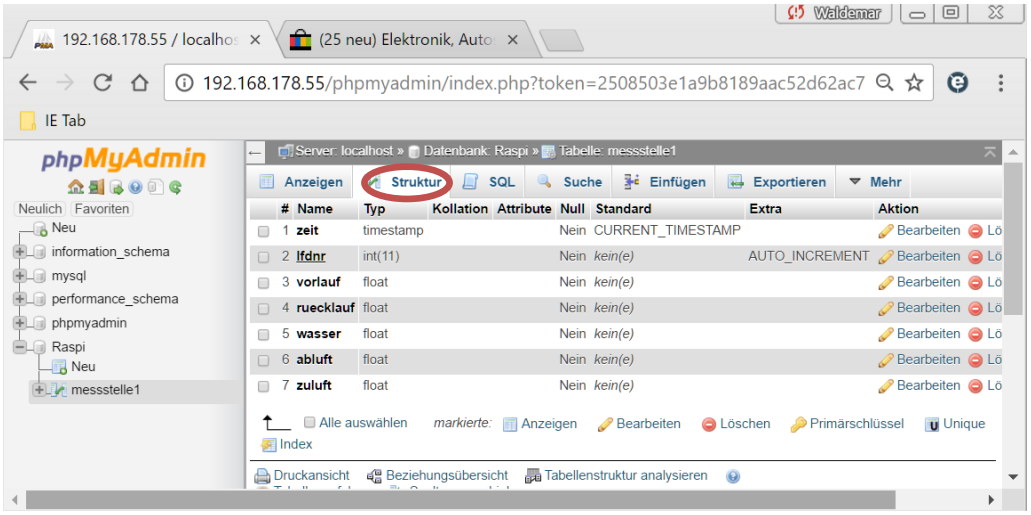
Nach erfolgreicher Anmeldung müssen wir zuerst eine Datenbank erzeugen.
Unter

Neu-Neue Datenbank anlegen-Datenbankname-Anlegen



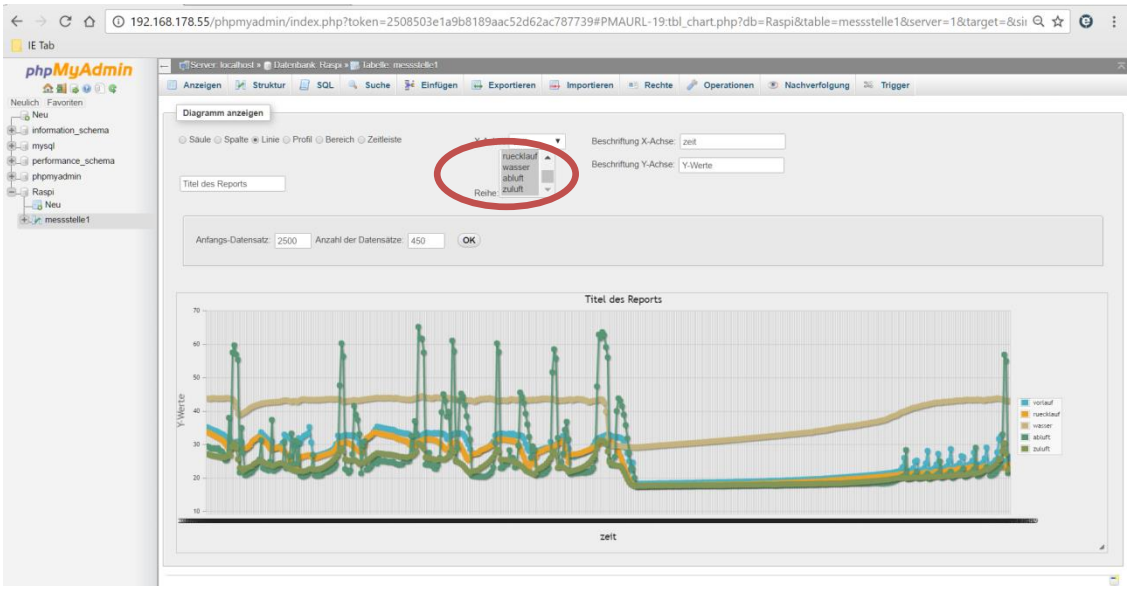
Nennen wir die Datenbank *Raspi*
Unter *Raspi-Neu-Tabellenname* tragen wir *messstelle1* ein

Und unter Name die benötigten Felder
Zeit – lfdnr – vorlauf – rücklauf – wasser – abluft – zuluft
Bei Typ, Null, Standard und Extra entsprechend ausfüllen.
Dann sollte das unter dem Reiter *Struktur* dann so aussehen



Fertig

Sollte alles funktionieren können wir etwa nach einem Tag (hier ca 24h) eine Grafik erstellen die dann so aussehen könnte.



Hier können die Kurven der Messsonden ab- und wieder zugeschaltet werden.