THOMAS MORE

Project Iot Slimme serre

Vermeulen Mathias
Graduaat Iot

2023 - 2024

Inhoud

1. Opdracht

De opdracht van dit project is om een slimme serre te maken voor de klant. Bij dit product worden planten zonder menselijke interventie van bloemzaad tot bloem/plant ontwikkeld. Sensoren zorgen voor de visuele data. Wanneer bepaalde grenswaarde overschreden worden, zullen er automatische acties ondernomen worden. De gegevens zullen ook nog doorgestuurd worden van de esp32 naar de rasberry pi. Hier wordt de data opgeslagen in een databank. Ook zullen de gegevens nog eens doorgestuurd worden naar grafana, zodat er grafieken geraadpleegd kunnen worden.

2. Iot in de landbouw

Het Internet of Things (IoT) speelt een cruciale rol in de moderne landbouw, waarbij technologieën worden ingezet om de efficiëntie, productiviteit en duurzaamheid van landbouwactiviteiten te verbeteren. Hier zijn enkele voorbeelden

- Besproeiing kan aangepast worden aan de specifieke belangen van de gewassen
- Voedingsstoffen kunnen op de juiste moment en in de juiste hoeveelheid gegven worden
- Groeipatronen kunnen geanalyseerd worden. Ook kan er tijdig ingegrepen worden bij afwijkingen
- Geautomatiseerde systemen zorgen ervoor dat handmatige taken vervangen worden
- Er kunnen besluiten genomen worden dankzij data dat doorgestuurd is geweest.
- …

Het belang van IoT in de landbouw is enorm, aangezien het technologieën biedt die boeren helpen om efficiënter, productiever en milieuvriendelijker te werken. Specifiek in kweeksystemen kunnen IoT-oplossingen de groeiomstandigheden optimaliseren, de productiekosten verlagen en de opbrengsten verhogen. Door gebruik te maken van geavanceerde sensoren, automatisering en datagestuurde besluitvorming kunnen moderne landbouwbedrijven beter inspelen op de uitdagingen van vandaag en de toekomst.

3. Vereisten

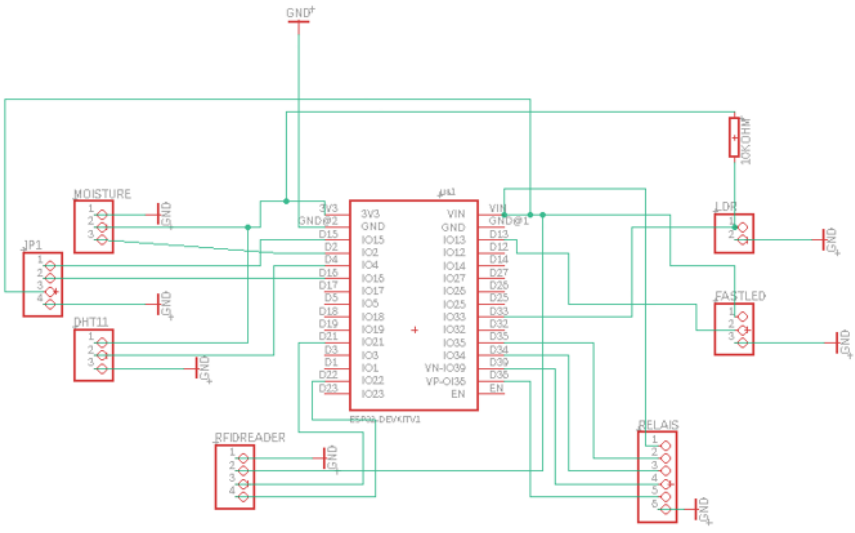
- Draadloze battery powered module
- Modulaire ontwerp
- Uitlezen van de sensoren
- Doorsturen van de data naar de server dmv mqtt
- Opslagen van de data op een server
- Weergeven van de data dmv van grafana
- Acties uitvoeren adhv bepaalde sensorwaarden
- LCD en RFID om de module in te stellen

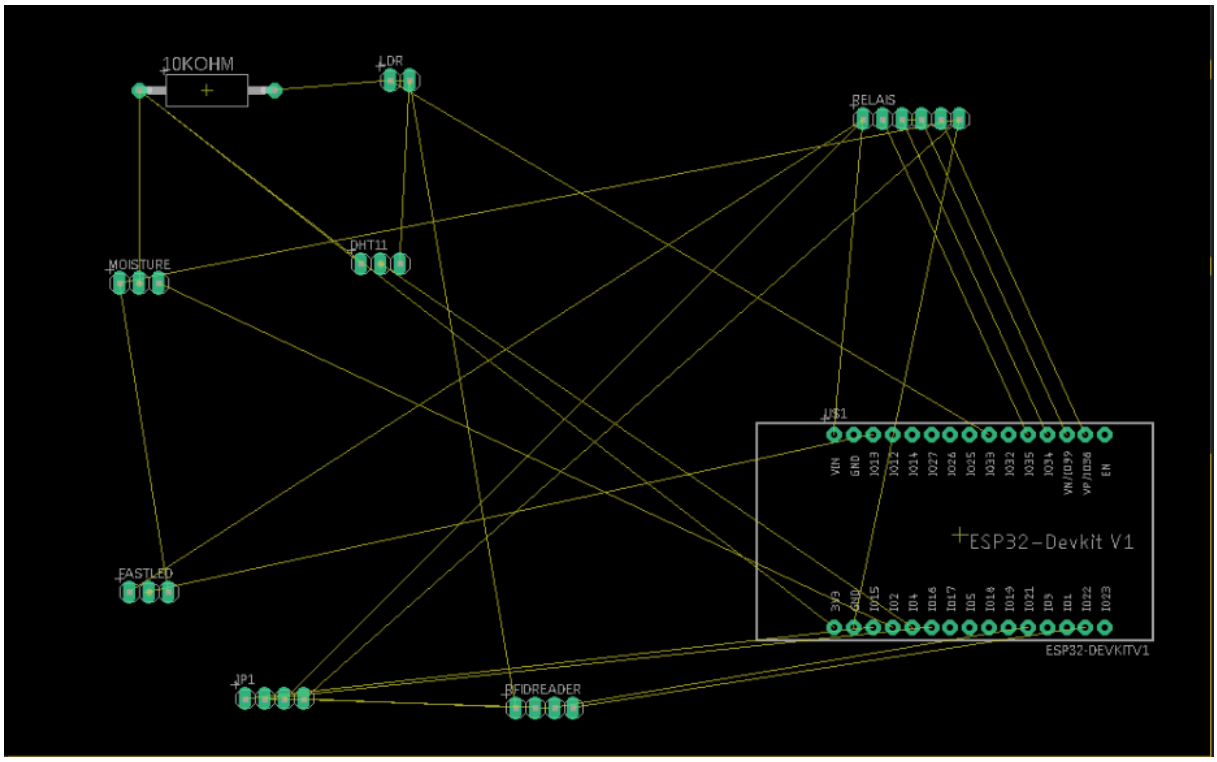
4. Benodigdheden

- Esp32
- LCD
- Rfid tad
- 12 volt adapter
- Dc buckconverter
- Afstandssesnor
- Dht11
- Moisture sensor
- LDR
- Servo motor
- Sensor voor de bodemvochtigheid
- Fastled
- Relais
- Rasberry pi
- Jumper wires
- Weerstandjes
- Waterpomp
-

5. Schema & pcb

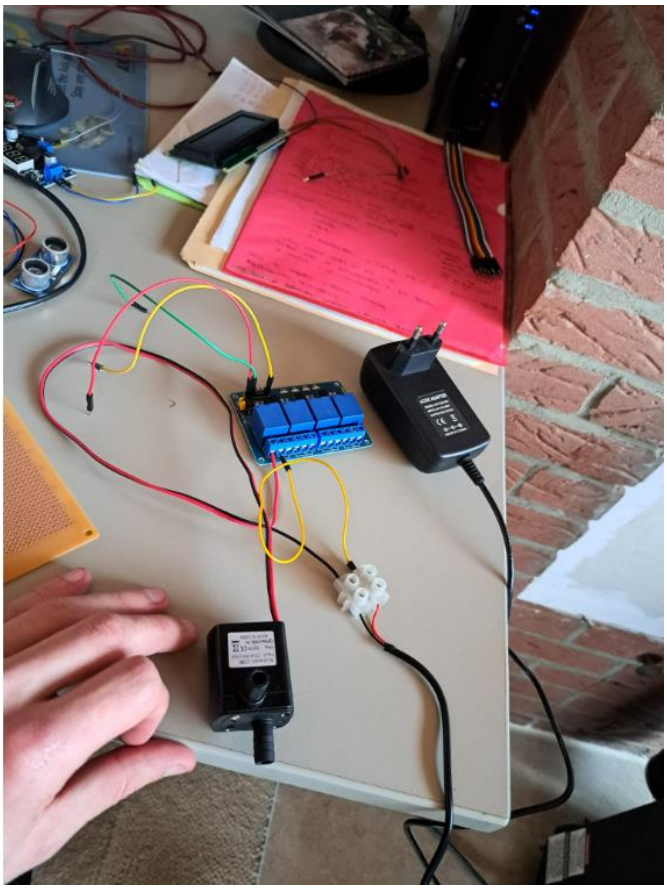
op het schema en het pcb plaatje, zijn er geen componenten terug te vinden. Er zijn pin headers opgezet geweest, zodat je de sensoren en esp32 er alleen maar hoeft in te drukken/plaatsen, zodat deze later nog kan gebruikt worden voor andere projecten.





6. Geschreven Connectie sensoren esp32

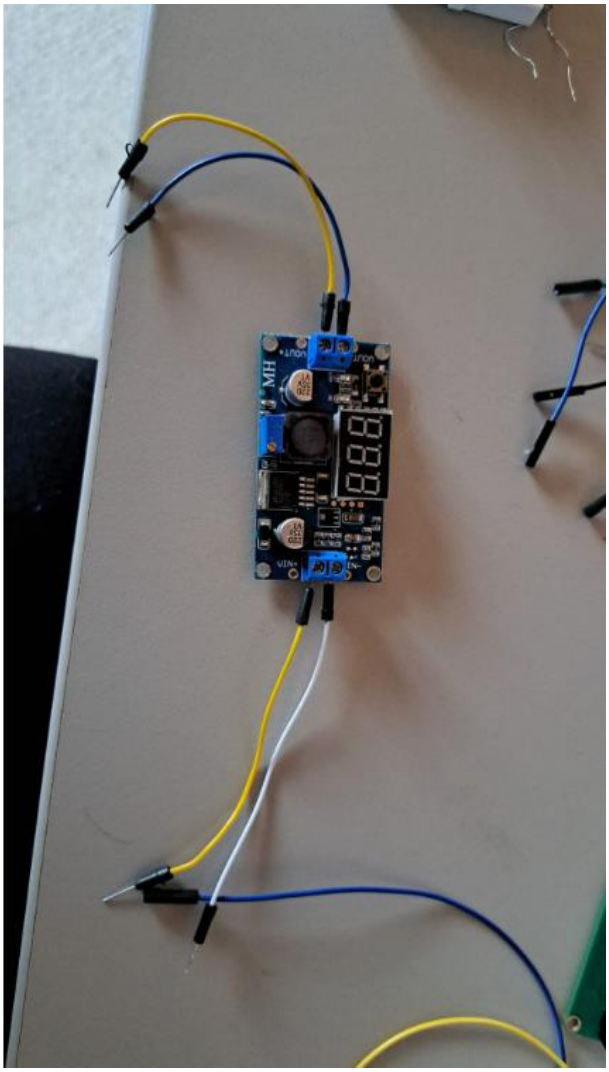
- De dht11 heeft drie connecties. Een op de 3.3v, een op de ground en de middelste (data pin) loopt naar pin nummer 4 van de esp32
- De LDR is verbonden op pin nummer 33 waar ook een weerstand op staat deze weerstand is verbonden met de 3.3v en de andere kant van de LDR is verbonden met de ground.
- De fastked heeft ook 3 pinnen om aan te sluiten. Een op de 3.3v de andere op de ground en de middelste op pin nummer 13 van de esp32
- De afstandssesnor heeft vier pinnen. Een ground en 3.3v pin. De trig pin staat op pin 5 en de echo pin op pin 18.
- De waterpomp is niet rechtstreeks verbonden aan de esp. Hiertussen staat nog een relais. De realis heeft meerdere pinnen, maar er worden er maar drie gebruikt. De vcc pin naar de 3.3v, de ground naar de ground van de esp32 en dan nog een IN4 naar pin nummer 26 op de esp32. De rode draad van de 12v adapter wordt verbonden met de middelste opening (COM) en de de ground wordt verbonden met de ground van de waterpomp. De rode draad van de waterpomp wordt verbonden met de links opening (NO) van de relais.



- De moisture sensor heeft ook drie pinnen. De vcc gaat naar de 3.3v en de ground gaat naar de ground van de esp32. De onderste AOUT gaat naar pin nummer 2 van de esp32.
- Als laatste is er dan nog de servo motor. Deze heeft ook een verbinding met de 3.3v en ground van de esp32. De onderste is verbonden met pin nummer 21 van de esp32.

7. Dc buckconverter

De buckconverter staat tussen de esp32 en de adapter. Op de uin+ komt de 12v binnen. De uin- is vrebonden met de ground van de adapter (zwarte draad). Aan de andere kan is er nog de uout+ dat verbonden is met de vin pin van de esp32 en de uout – dat verbonden is met de ground van de esp32. Om de buckconverter te veranderen naar een output van 5v, verbind je deze eerst alleen met de 12v adapter. En duw je op de knop rechts bovenaan (zie afbeelding) zodat de volt weergegeven wordt wanneer het de converter verlaat. Dan moet er alleen nog aan het knopje gedraaid wordt links in het midden (zie afbeelding) dit kan het best met schroevendraaier gedaan worden. Zet deze dan naar 5v.

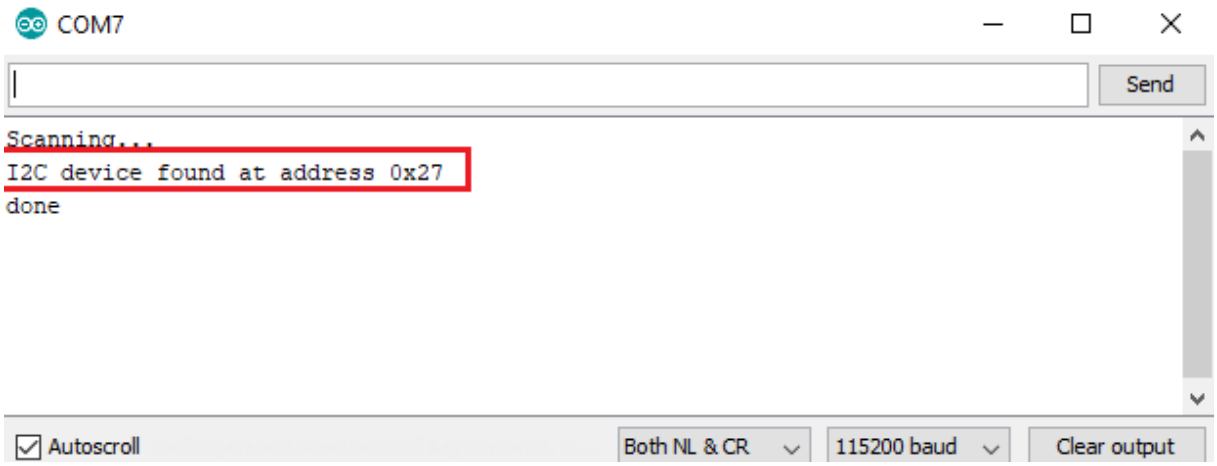


8. LCD

De LCD werkt op i2c en heeft dus maar vier pinnen. Een ground en vcc dat naar de esp32 gaat en dan nog een SDA en SCL. Deze laatste twee zijn geconnecteerd met nummers 21 en 22 van de esp32.
Om de LCD werkende te krijgen verbind je dit apparaat zoals beschreven staat en voer je als eerst volgende code is.

```
#include <Wire.h>

void setup() {
  Wire.begin();
  Serial.begin(115200);
  Serial.println("\nI2C Scanner");
}

void loop() {
  byte error, address;
  int nDevices;
  Serial.println("Scanning...");
  nDevices = 0;
  for(address = 1; address < 127; address++ ) {
    Wire.beginTransmission(address);
    error = Wire.endTransmission();
    if (error == 0) {
      Serial.print("I2C device found at address 0x");
      if (address<16) {
        Serial.print("0");
      }
      Serial.println(address,HEX);
      nDevices++;
    }
    else if (error==4) {
      Serial.print("Unknow error at address 0x");
      if (address<16) {
        Serial.print("0");
      }
      Serial.println(address,HEX);
    }
  }
  if (nDevices == 0) {
    Serial.println("No I2C devices found\n");
  }
  else {
    Serial.println("done\n");
  }
  delay(5000);
}
```

Open de Seriële monitor en je krijgt, en je krijgt dan het volgende te zien.



De code wordt dan weggedaan, en er worden een nieuwe geplaatst. Vul de ‘ 0x27 is bij “ LiquidCrystal_I2C lcd (0x27, lcdColums, cldRows);

```
#include <LiquidCrystal_I2C.h>

// set the LCD number of columns and rows
int lcdColumns = 16;
int lcdRows = 2;

// set LCD address, number of columns and rows
// if you don't know your display address, run an I2C scanner sketch
LiquidCrystal_I2C lcd(0x27, lcdColumns, lcdRows);

void setup(){
  // initialize LCD
  lcd.init();
  // turn on LCD backlight
  lcd.backlight();
}

void loop(){
  // set cursor to first column, first row

  lcd.setCursor(0, 0);
  // print message
  lcd.print("Hello, World!");
  delay(1000);
  // clears the display to print new message
```

```
  lcd.clear();
  // set cursor to first column, second row

}
```

Nu kan je de code aanpassen en de gegevens laten zien op het schermpje.

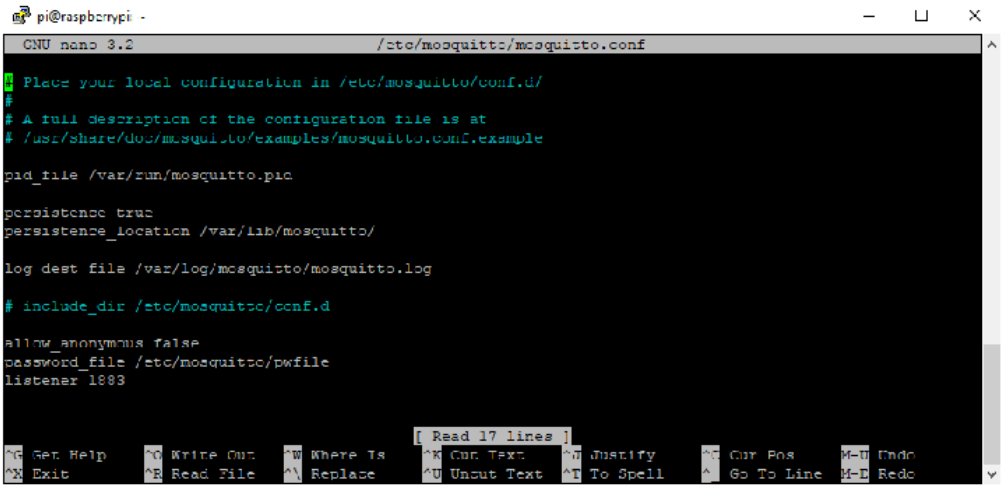
9. Documentatie grafana

Om de gegevens te kunnen visualiseren, gebruiken we grafana. Grafana haalt deze uit een database dat op de rasberry py staat. Voordat ze hierop terechtkomen moeten ze eerst via mqtt kunnen worden verstuurd van de esp32 naar de rasberry. Volg de onderstaande stappen om de rasberry klaar te maken om data te ontvangen, opslaan en door te sturen naar grafana.

Installatie mqtt:

Mqtt zorgt voor de verbinding tussen Iot apparaten. Kopieer volgende stappen een voor een om mqtt te installeren op de rasberry pi

1. sudo apt update
2. sudo apt upgrade
3. sudo apt install mosquitto mosquitto-clients
4. sudo systemctl status mosquitto
5. Dan voer je onderstaand commando uit dat weergegeven is op de afbeelding en zorg je ervoor dat de file die je opent hetzelfde is.
   - Sudo nano /etc/mosquitto/mosquitto.conf



Als laatste stap maak je nog een user name en wachtwoord aan, die doe je op dezelfde manier als onderstaande afbeelding toont.

- Sudo mosquitto_passwd -c /etc/mosquitto/pwfile username
  - Sudo : root user
  - Mosquitto_passwd : make password
  - -c : create new password file (more info on manual page (man mosquitto_passwd)
  - /etc/... : location for the password file
  - Username : free to choose name for the user

Installatie influxdb

Influxdb is een database dat de data (in dit geval) van de esp32 opslaagd. Kopieer elke stap één voor één om influxdb te installeren. Wat er tussen de haakjes staat is niet nodig.

1. Sudo apt update (zeker zijn dat de packages die nu geïnstalleerd zijn, up to date zijn)
2. Sudo apt upgrade (zeker zijn dat de packages die nu geïnstalleerd zijn, up to date zijn)
3. curl https://repos.influxdata.com/influxdata-archive.key | gpg --dearmor | sudo tee /usr/share/keyrings/influxdb-archive-keyring.gpg >/dev/null
4. echo "deb [signed-by=/usr/share/keyrings/influxdb-archive-keyring.gpg] https://repos.influxdata.com/debian stable main" | sudo tee /etc/apt/sources.list.d/influxdb.list
5. sudo apt update
6. sudo apt install influxdb
7. sudo systemctl unmask influxdb
8. sudo systemctl enable influxdb
9. sudo systemctl start influxdb
10. Als laatst doe je hetzelfde als op ondestaande afbeelding getoond wordt om een user name en wachtwoord te maken.

- CREATE USER user WITH PASSWORD ‘userpassword’
- GRANT ALL ON databasename TO user

Installatie grafana

Grafana zal de data die hij krijgt van de database (influxdb) op grafieken kunnen weergeven. Kopieer elke stap één voor één om grafana te installeren op derasberry pi.

1. sudo apt update
2. sudo apt upgrade
3. curl https://apt.grafana.com/gpg.key | gpg --dearmor | sudo tee /usr/share/keyrings/grafana-archive-keyrings.gpg >/dev/null
4. echo "deb [signed-by=/usr/share/keyrings/grafana-archive-keyrings.gpg] https://apt.grafana.com stable main" | sudo tee /etc/apt/sources.list.d/grafana.list
5. sudo apt update
6. sudo apt install grafana
7. sudo systemctl enable grafana-server
8. sudo systemctl start grafana-server

Om naar grafana te gaan geef je het ip-adress in van je rasberry pi gevolgd door een dubbelpunt teken en het getaal 3000. Bijvoorbeeld --) 192.168.5.20:3000
Om het ip adres te vinden van de rasberry pi, geef je het commando ‘ifconfig’ in de terminal.

Je gaat voor een keer kunnen inloggen met ‘ admin ‘ voor zowel de gebruikersnaam als het wachtwoord. Je veranderd deze meteen.

10. Code esp32

```
#include <WiFi.h>
#include <PubSubClient.h>
#include <DHT11.h>
#include <FastLED.h>
#include <LiquidCrystal_I2C.h>
#include <Wire.h>
// hier staan alle libraries die nodig zijn voor dit project
// Pin definities
#define LED_PIN 13
#define NUM_LEDS 30
#define LDR_PIN 33
#define SOUND_SPEED 0.034
#define RELAY_PIN 23
#define TRIG_PIN 5
#define ECHO_PIN 18
#define DHT_PIN 4
#define DHTTYPE DHT11
#define Waterpomp_pin 26
#define LDR_THRESHOLD 2000
#define sensor_pin 2
#define SERVO_PIN 21  // Pin waar de servo is aangesloten  // Soil moisture
sensor O/P pin
```

```
const char* ssid = "";
const char* password = "";
const char* mqttServer = "";
const int mqttPort = 1883;
const char* mqttUser = "Mathias";
const char* mqttPassword = "Wpla";
const char* clientID = "client_livingroom";
// dit zijn de instellingen van de wifi en mqtt

// MQTT topics
const char* moisture_topic = "home/livingroom/moisture";
const char* temperature_topic = "home/livingroom/temperature";
```

```
const char* humidity_topic = "home/livingroom/humidity";
const char* light_topic = "home/livingroom/light";
const char* distance_topic = "home/livingroom/distance";
const int servoMinPulseWidth = 500;   // Minimum pulsbreedte in microseconden
const int servoMaxPulseWidth = 2400;  // Maximum pulsbreedte in microseconden
const int servoFrequency = 50;

int lcdColumns = 16;
int lcdRows = 2;

WiFiClient espClient;
PubSubClient client(espClient);
DHT11 dht11(DHT_PIN);
CRGB leds[NUM_LEDS];
LiquidCrystal_I2C lcd(0x27, 16, 2);


// Variabelen
int _moisture, sensor_analog;
long duration;
float distanceCm;
float distanceInch;

void setup_wifi() {
  delay(10);
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");//hier wordt er geconnecteerd met de wifi
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}// hier wordt er verbinding gemaakt meet de wifi

void reconnect() {
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    if (client.connect(clientID, mqttUser, mqttPassword)) {
      Serial.println("connected");
```

```
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      delay(2000);
    }// dit is bedoeld om te verbinden met de mqtt van de rasberry
  }
}

void setup() {
  Serial.begin(115200);
  setup_wifi();
  client.setServer(mqttServer, mqttPort);

  FastLED.addLeds<NEOPIXEL, LED_PIN>(leds, NUM_LEDS);
  pinMode(TRIG_PIN, OUTPUT);  // Sets the trigPin as an Output
  pinMode(ECHO_PIN, INPUT);   // Sets the echoPin as an Input
  pinMode(RELAY_PIN, OUTPUT);
  pinMode(Waterpomp_pin, OUTPUT);
  pinMode(LDR_PIN, INPUT);
  // hier worden de pinmodes gedeclareerd, wat wordt een in of output.

  lcd.init();
  lcd.backlight();
}

void loop() {




  if (!client.connected()) {
    reconnect();
  }
  client.loop();
  delay(1000);
  waterpomp();

  gegevens();
  gegevens2();

  servo();

  // LDR Sensor
  int LDR_val = analogRead(LDR_PIN);
  Serial.print("LDR = ");
  Serial.println(LDR_val);
  lcd.setCursor(0, 0);
```

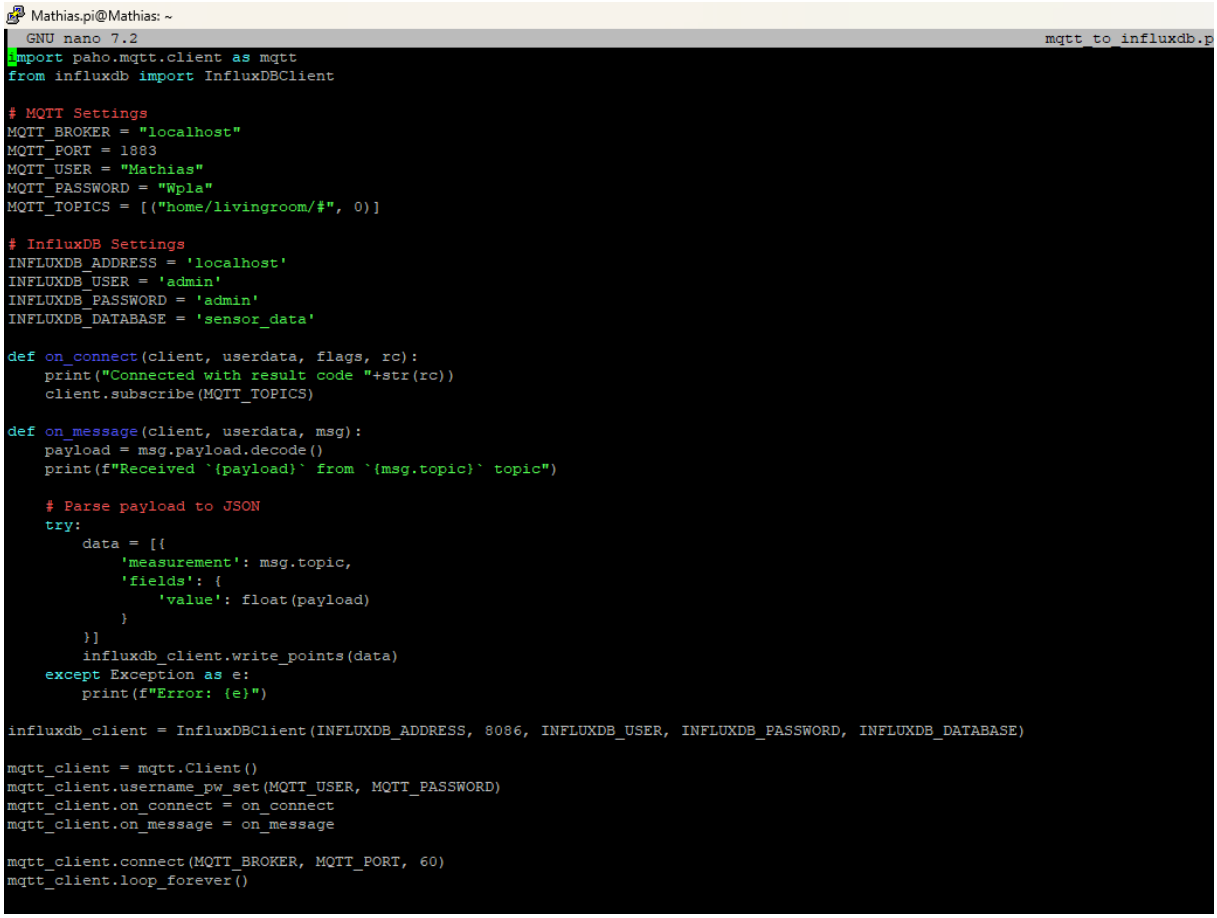
```
  lcd.print("Licht: ");
  lcd.setCursor(0, 1);
  lcd.print(LDR_val);
  delay(1000);
  lcd.clear();
  client.publish(light_topic, String(LDR_val).c_str());

  // Controleer de waarde van de LDR en schakel het licht in of uit
  if (LDR_val > LDR_THRESHOLD) {
    fill_solid(leds, NUM_LEDS, CRGB(255, 0, 0));  // Rood
  } else {
    fill_solid(leds, NUM_LEDS, CRGB::Black);
  }
  FastLED.show();// hier wordt de data van de ldr weergegeven op de lcd,
wanneer de waarde boven de 2000is, zal de led aan gaan.


  // Ultrasonic Sensor
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  duration = pulseIn(ECHO_PIN, HIGH);
  distanceCm = duration * SOUND_SPEED / 2;
  Serial.print("Distance (cm): ");
  Serial.println(distanceCm);
  lcd.setCursor(0, 0);
  lcd.print("Afstand: ");
  lcd.setCursor(0, 1);
  lcd.print(distanceCm);
  lcd.print("cm");
  delay(1000);
  lcd.clear();
  client.publish(distance_topic, String(distanceCm).c_str());
}// hier is de code om de afstandssensor te meten en weer te geven op de lcd,
als ook de data door te sturen naar de rasberry pi

void waterpomp() {


  digitalWrite(Waterpomp_pin, HIGH);
  // Wacht 3 seconden
  delay(3000);

  // Zet het relais uit
  digitalWrite(Waterpomp_pin, LOW);
  // Wacht 3 seconden
```

```
  delay(3000);
}

void gegevens(){
    // Soil Moisture Sensor
  sensor_analog = analogRead(sensor_pin);
  _moisture = (100 - ((sensor_analog / 4095.00) * 100));
  Serial.print("Moisture = ");
  Serial.print(_moisture);
  Serial.println("%");
  lcd.setCursor(0, 0);
  lcd.print("Bodemvocht: ");
  lcd.setCursor(0, 1);
  lcd.print(_moisture);
  delay(1000);
  lcd.clear();
  client.publish(moisture_topic, String(_moisture).c_str());
}//hier wordt bodemvochtigheid berekend en weergegeven op de lcd, de data
wordt ook naar de rasberry gestuurd.


void servo(){
  int temperature = dht11.readTemperature();
  int humidity = 0;
  int result = dht11.readTemperatureHumidity(temperature, humidity);
  int dutyCycle = map(0, 0, 180, servoMinPulseWidth, servoMaxPulseWidth);

  if (isnan(temperature)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }

  Serial.print("Temperature: ");
  Serial.print(temperature);
  Serial.println(" °C");




  if (temperature > 20) {
    dutyCycle = map(180, 0, 180, servoMinPulseWidth, servoMaxPulseWidth);
  } else {
    dutyCycle = map(0, 0, 180, servoMinPulseWidth, servoMaxPulseWidth);
  }

  // Send the duty cycle to the servo
  ledcWrite(0, dutyCycle);
```

```
  delay(3000);  // Wait 3 seconds before reading again
}// hier zal de servo 180 graden draaien en de deur opendoen wanneer het
warmer is dan 20 graden in de kweekbak. anders blijft die in zijn begin
positie staan.



void gegevens2(){
  int temperature = dht11.readTemperature();
  int humidity = 0;
  int result = dht11.readTemperatureHumidity(temperature, humidity);
  int dutyCycle = map(0, 0, 180, servoMinPulseWidth, servoMaxPulseWidth);


  if (result == 0) {
    Serial.print("Temperature: ");
    Serial.print(temperature);
    Serial.print(" °C\tHumidity: ");
    Serial.print(humidity);
    Serial.println(" %");
    lcd.setCursor(0, 0);
    lcd.print("Temp: ");
    lcd.print(temperature);
    lcd.print("C");
    delay(2000);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Vochtigheid: ");
    lcd.print(humidity);
    lcd.print("%");
    delay(1000);
    lcd.clear();
    client.publish(temperature_topic, String(temperature).c_str());
    client.publish(humidity_topic, String(humidity).c_str());
  } else {
    Serial.println(DHT11::getErrorString(result));
  }
}
```

11. Code rasberry pi

Plak onderstaande code in een nano bestand die je een eigen naam geeft. Bij de mqtt settings en influx dB settings geef je jouw gegevens voor een goede connectie. Run dit bestand nadat je ook de arduino code hebt gecompileerd.



```
Mathias.pi@Mathias: ~
  GNU nano 7.2                                                                mqtt_to_influxdb.p
import paho.mqtt.client as mqtt
from influxdb import InfluxDBClient

# MQTT Settings
MQTT_BROKER = "localhost"
MQTT_PORT = 1883
MQTT_USER = "Mathias"
MQTT_PASSWORD = "Wpla"
MQTT_TOPICS = [("home/livingroom/#", 0)]

# InfluxDB Settings
INFLUXDB_ADDRESS = 'localhost'
INFLUXDB_USER = 'admin'
INFLUXDB_PASSWORD = 'admin'
INFLUXDB_DATABASE = 'sensor_data'

def on_connect(client, userdata, flags, rc):
    print("Connected with result code "+str(rc))
    client.subscribe(MQTT_TOPICS)

def on_message(client, userdata, msg):
    payload = msg.payload.decode()
    print(f"Received `{payload}` from `{msg.topic}` topic")

    # Parse payload to JSON
    try:
        data = [{
            'measurement': msg.topic,
            'fields': {
                'value': float(payload)
            }
        }]
        influxdb_client.write_points(data)
    except Exception as e:
        print(f"Error: {e}")

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

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_BROKER, MQTT_PORT, 60)
mqtt_client.loop_forever()
```

CONTACT

Naam | Functie
xxx.xxx@thomasmore.be
Tel. + 32 xx xx xx xx

VOLG ONS

www.thomasmore.be
fb.com/ThomasMoreBE
#WeAreMore

THOMAS MORE