SEEDLNG

Verslag

Project IOT

- **Ben Janssen**
- **Lian Aarts**
- **Niels Cuyvers**
- **Vincent Somers**

Academiejaar 2020-2021
Campus Geel, Kleinhoefstraat 4, BE-2440 Geel

Inhoud

Description:
RF System on a Chip - SoC (Engineering Samples Only) SMD Wi-Fi IC Single-Core MCU, 4 MB flash and 2 MB PSRAM inside, QFN 56-pin, 7*7 mm
Length: 7 mm
Height: 7 mm
Power consumption:

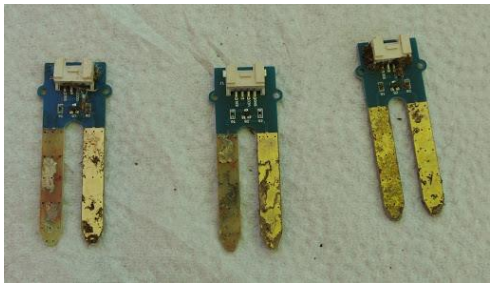
Table 13: Current Consumption Depending on Work Modes

Work mode	Description		Current consumption (Typ)
Modem-sleep	The CPU is powered on	240 MHz	19 mA
		160 MHz	16 mA
		Normal speed: 80 MHz	12 mA
Light-sleep	—		450 μA
Deep-sleep	The ULP co-processor is powered on	ULP-FSM	170 μA
		ULP-RISC-V	190 μA
	ULP sensor-monitored pattern		22 μA @1% duty
	RTC timer + RTC memory		25 μA
	RTC timer only		20 μA
Power off	CHIP_PU is set to low level, the chip is powered off		1 μA

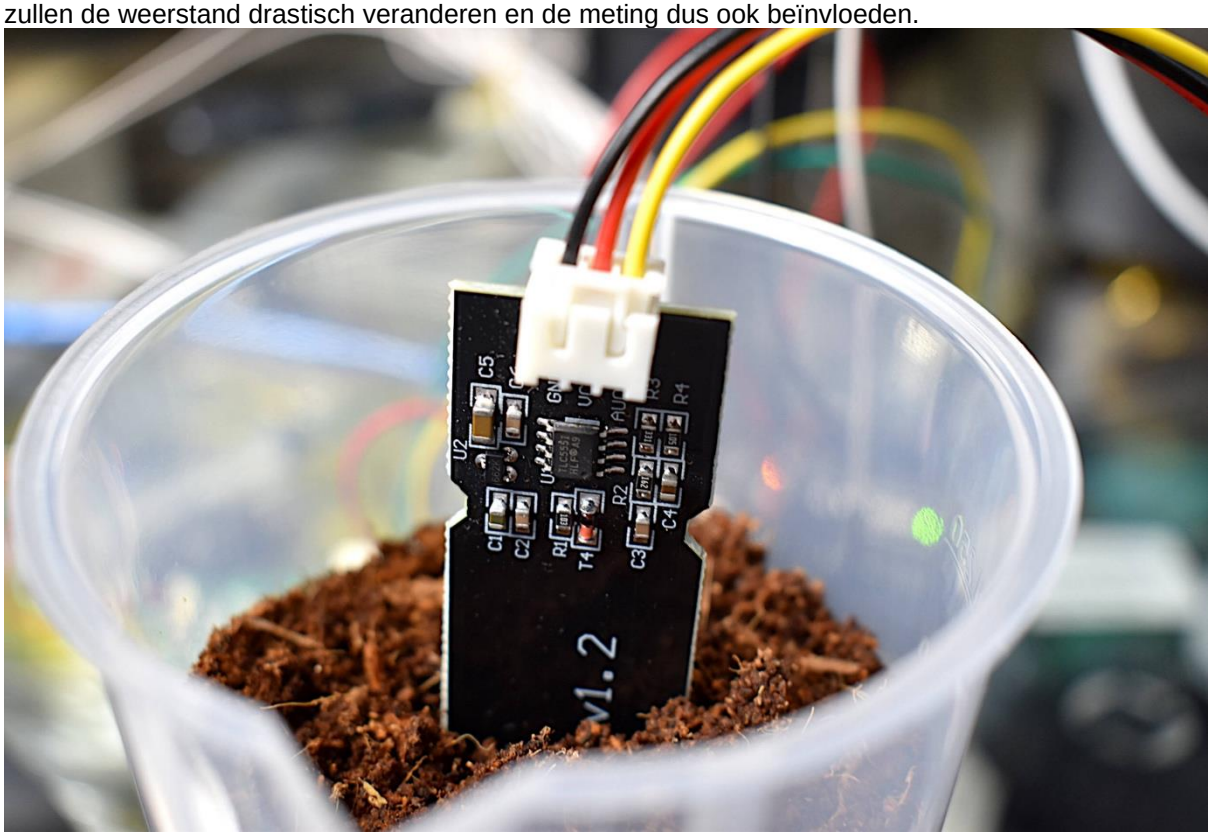
SENSOREN

Vochtigheid sensor

Wanneer we de vochtigheid van aarde willen meten zijn er twee soorten sensoren die we hiervoor kunnen gebruiken. Resistieve en capacitieve vochtigheidssensoren hebben elk hun voor-en nadelen.



En resistieve vochtigheidssensor gaat de weerstand van de aarde meten met behulp van twee geleiders. Wanneer we de vochtigheid willen meten leggen we een spanning aan beide geleiders. De weerstand zal dalen wanneer er meer vocht aanwezig is in de aarde. Een spanning in combinatie met water en mineralen leidt echter tot elektrolyse. Kopermoleculen zullen zich van de anode naar de kathode verplaatsen. De verplaatsing is meestal verwaarloosbaar en niet op te merken. Maar omdat de geleiders een groot oppervlak hebben en bestaan uit dunne pcb-paden zal de sensor snel verteren. We kunnen dit tegenwerken door periodische metingen te doen zodat er geen constante elektrische lading aanwezig is.
Capacitieve sensoren meten een verandering in capaciteit. Vochtigheid in de aarde zal het gedrag van het diëlektricum wijzigen en daardoor ook de capacitieve lading veranderen. Water is echter een slechte geleider, het zijn de ionen(mineralen) in het water die de lading beïnvloeden. De ionen zijn niet enkel aanwezig in water, maar ook andere stoffen zoals mest en voedingstoffen. Meststoffen zullen de weerstand drastisch veranderen en de meting dus ook beïnvloeden.

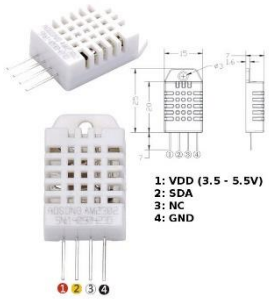


LDR

Een LDR is een lichtsensor, deze meet de waarde van de hoeveelheid licht er op de sensor valt. Het grote voordeel van deze sensoren is dat ze heel goedkoop zijn en heel grote spanningen kunnen verdragen.
We kunnen deze sensor gebruiken om zo de lichtinval op de plant te meten. Als er te veel of te weinig lichtinval is op de plant dan kan dit worden weergegeven op het scherm en of app.

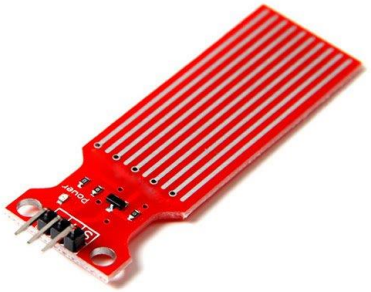
DHT-22

https://www.conrad.be/p/joy-it-sen-dht22-temperatuursensor-temperatuurvochtigheidssensor-geschikt-voor-arduino-asus-asus-tinker-board-banan-2159178
De DHT-22 is een temperatuur en vochtigheidssensor. Deze sensor werkt op 3-5V en kan temperaturen van -40°C tot 80°C meten en is op 0.5°C accuraat.
Deze sensor gebruiken we om zo de temperatuur te kunnen monitoren. Ook kunnen we hiermee del luchtvochtigheid weergeven wat ook zeer belangrijk is voor een plant.



Waterniveau sensor

De waterniveau sensor werkt als een grote weerstand. Wanneer het water contact maakt met de lijntjes zal de weerstand verlagen, op basis hiervan weet de sensor hoe hoog het water staat. Meer water betekent meer geleiding, en dus een kleinere weerstand.
Specificaties:
Spanning: 3-5V (20mA)
Detectiegebied: 40mm x 16mm
Bedrijfstemperatuur: 10℃ - 30℃
Afmeting: 62x20x8mm



ACTOREN

Waterpomp

Door gebruik te maken van een waterpomp kan de bloempot zelfstandig de plant water geven wanneer nodig. Deze waterpomp bevindt zich in een reservoir waar water inzit. Dit water wordt gegeven door de gebruiker en kan tot enkele weken genoeg zijn om de plant tot leven te houden.

VISUALISATIE

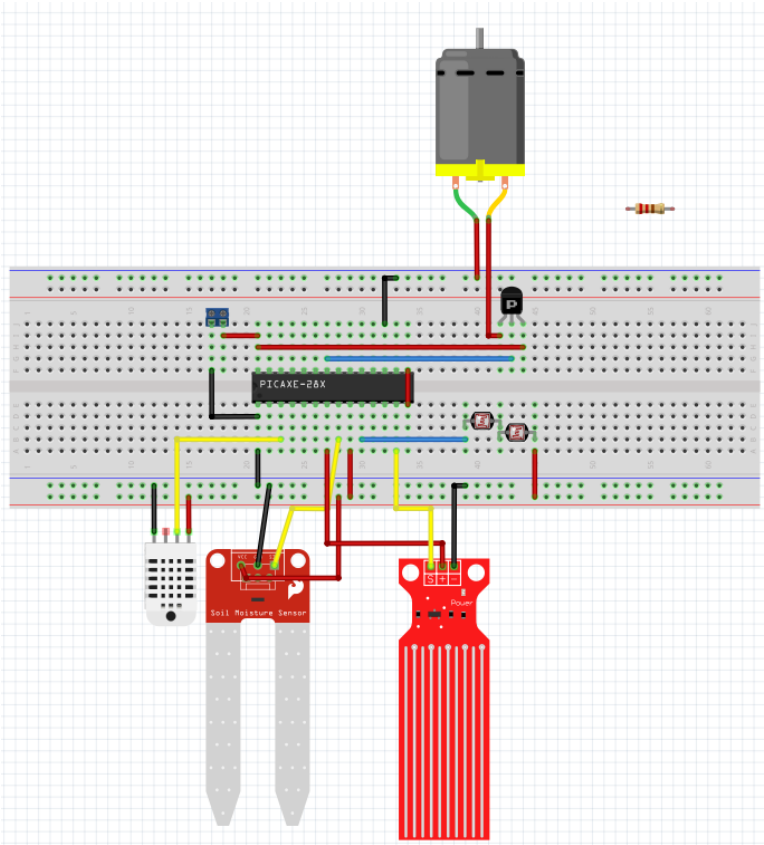
E-paper

Een E-paper display is een display dat werkt met zwarte en witte kleuren. Deze kleuren worden getoond met behulp van een magnetisch veld dat zich in de display bevindt. De witte kleuren worden getoond door de witte deeltjes in de display aan te trekken, en dit is omgekeerd voor de zwarte deeltjes. Het verbruik van deze display maakt deze display het beste om te gebruiken voor dit project omdat het niet altijd moet opstaan. Alleen als de display gerefresht moet worden staat hij aan. Het gedrukte op het scherm blijft staan voor een onbepaalde duur of tot de volgende refresh.
Standby power: <0.017mW
2.66inch E-Paper E-Ink Display Module, 296×152 Pixels, Black / White Dual-Color, SPI Interface (waveshare.com)

Home Assistant

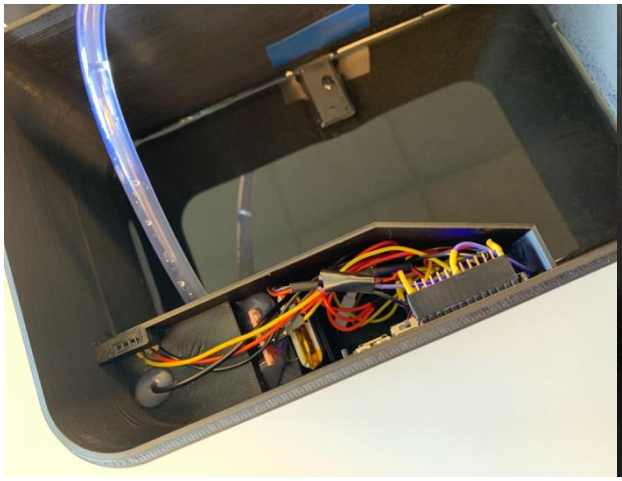
Doordat we een integratie maken met Home Assistant is het mogelijk om de data van de sensoren weer te geven op een dashboard dat je via een smartphone of PC kan monitoren. Ook is het mogelijk om hier de plant in te stellen zodat het systeem weet wat de benodigdheden zijn van de plant in de bloempot.
Als je wilt dat de display donker of licht is kan je dat hier ook aanpassen.

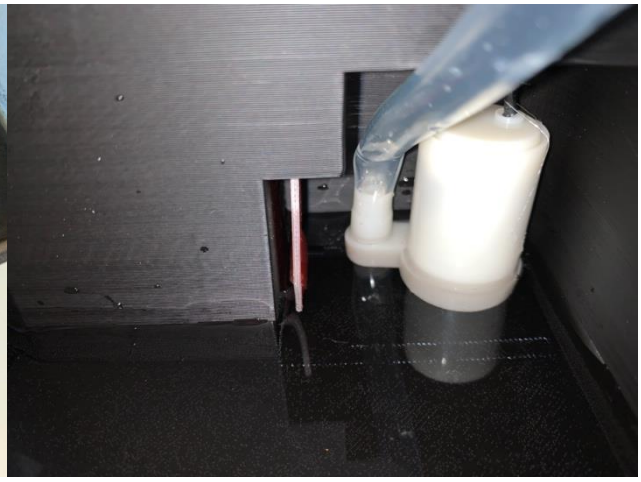
Schematische voorstelling



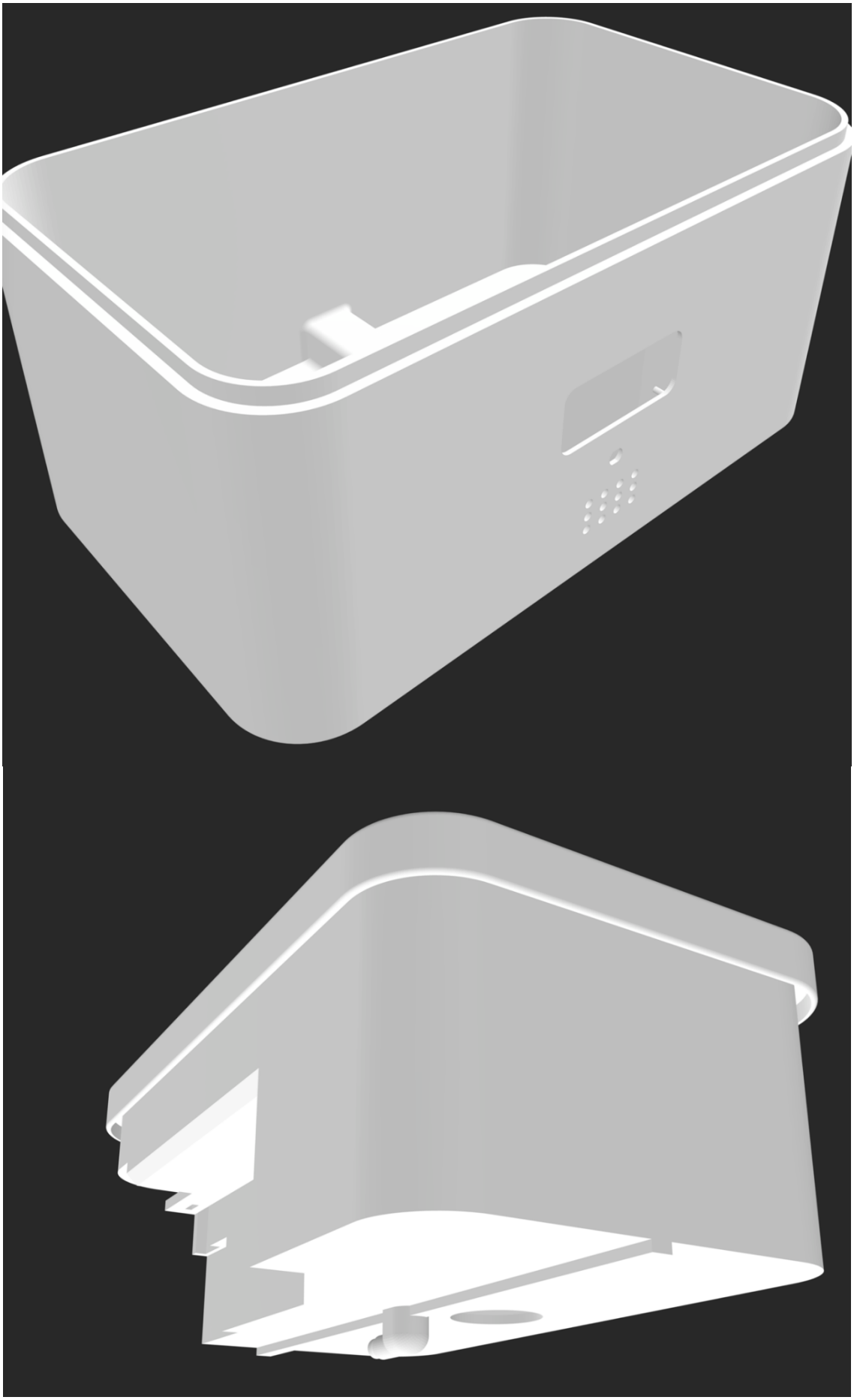
BEHUIZING

De derde iteratie van het design voorziet plaats voor alle sensoren actuatoren. Er is ook plaats voorzien voor een usb-poort die men kan gebruiken voor het opladen van de bloempot. Het waterreservoir heeft een inhoud van ongeveer 700ml.



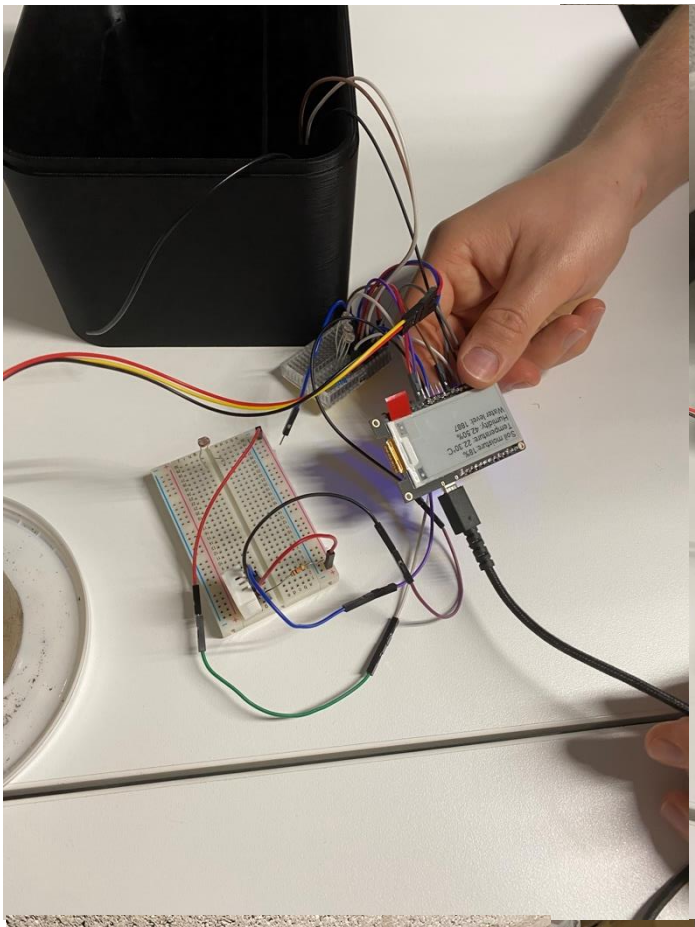


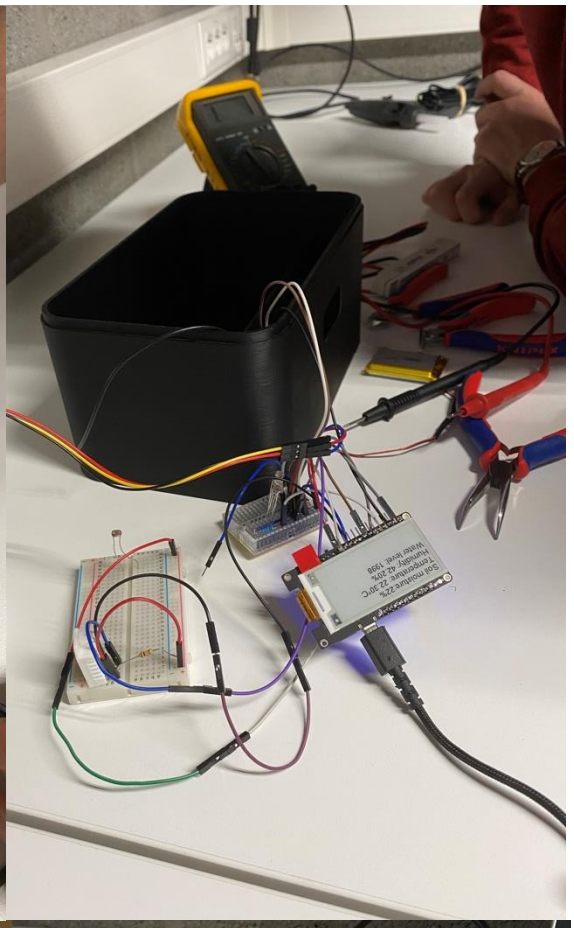
Het tweede deel van de bloempot voorziet plaat voor de plant en de bodemvochtigheid sensor. Interne waterleidingen voorzien de plant van irrigatie wanneer de bodemvochtigheid te laag is.

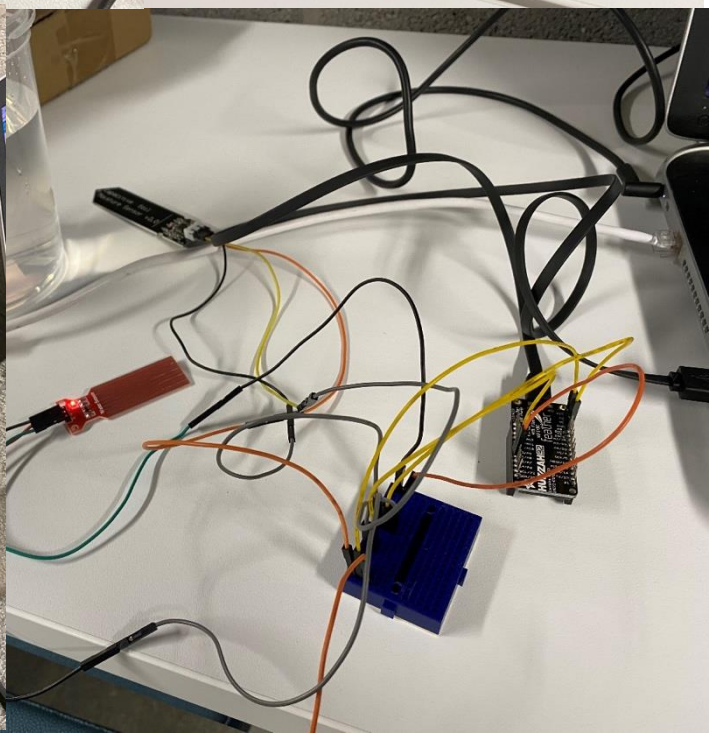


De 3D-tekening is gemaakt met Autodesk Inventor en geprint op een Prusa i3 MK3s.

REALISATIE VAN HET PROJECT:





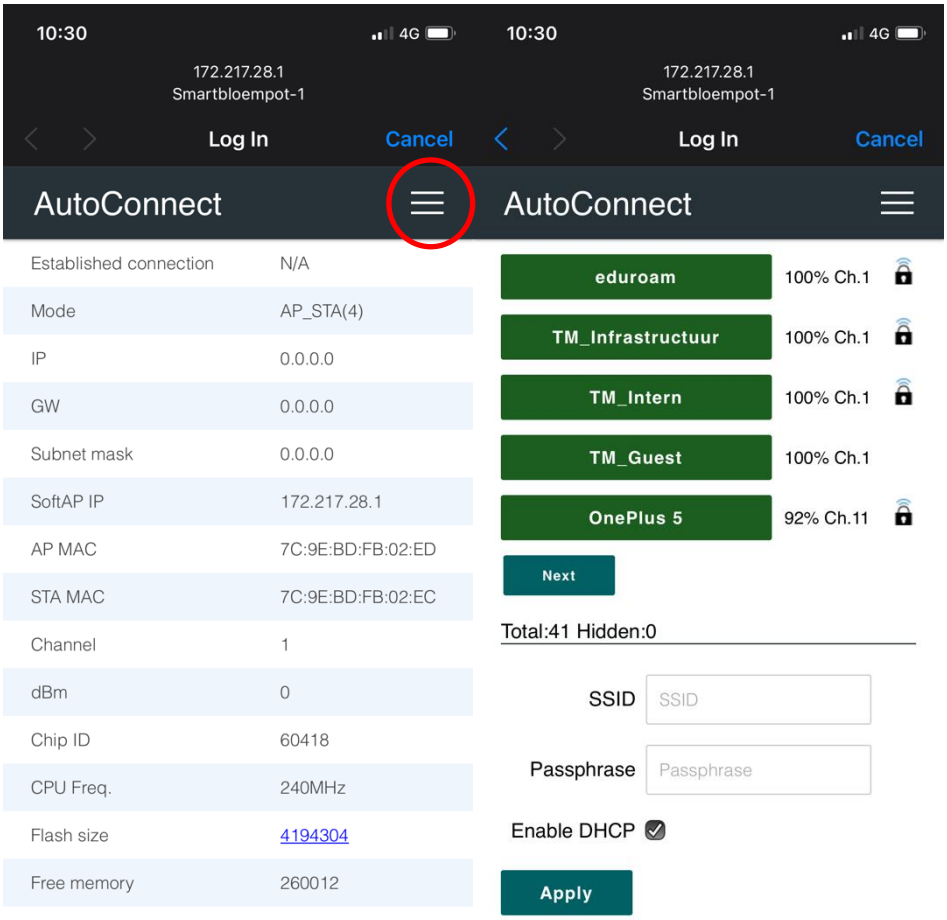



0
0.00%
0.00°C
0%
NIKS GEKREGEN

HANDLEIDING:

Simple Setup

1. Laadt de Seedlng voldoende op via de usb-poort die te vinden is aan de achterzijde van de bloempot
2. Wanneer “WiFi Setup” verschijnt op het scherm kan u verbinding maken via WiFi.
3. Verbind met het netwerk “bloempot-1” WW: passpass



4. Gebruik het hamburger icon om te navigeren naar “Configure new AP”
5. Kies hier het gewenste netwerk en vul het wachtwoord in.
6. De bloempot is nu verbonden met het gekozen netwerk.
7. De meetwaarden zijn zichtbaar op het scherm, de bloempot zal u plant verzorgen

Advanced Setup:

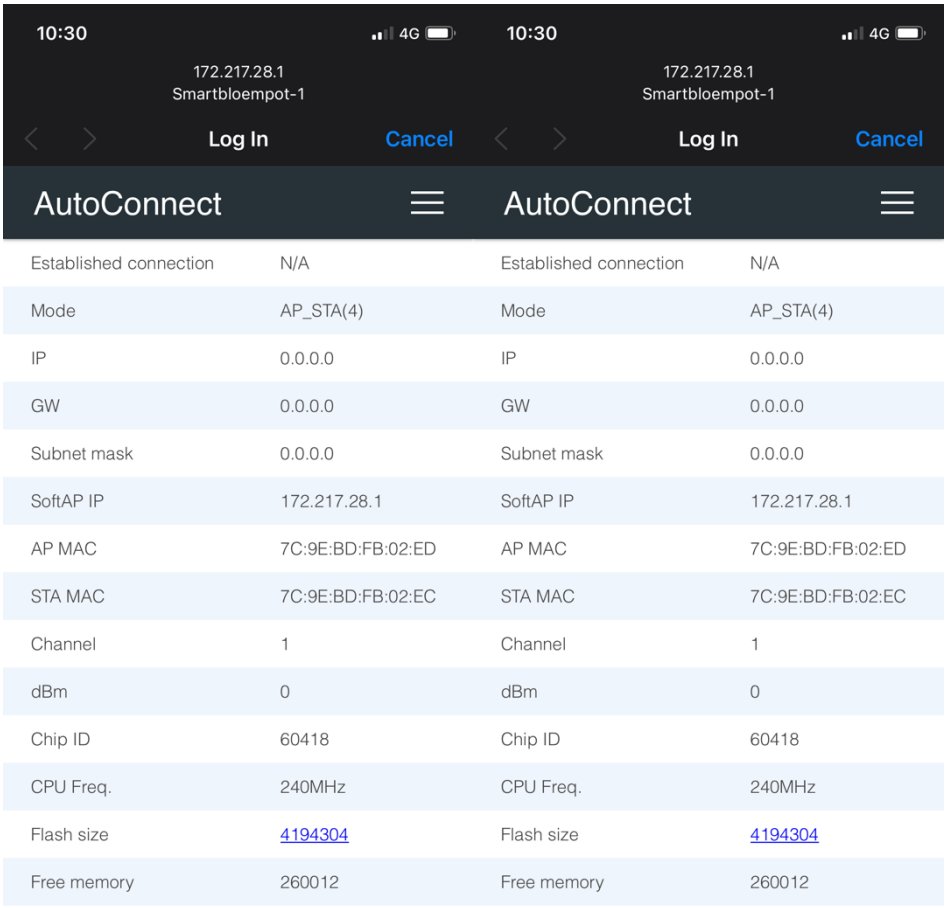
1. Laadt de Seedlng voldoende op via de usb-poort die te vinden is aan de achterzijde van de bloempot
2. Verbindt de Seedling met je computer via de usb-poort aan de achterzijde.

```
#define mqtt_port 1883
#define MQTT_USER "YOUR_MQTT_USER"
#define MQTT_PASSWORD "YOUR_MQTT_USER_PASSWORD"

const char* mqtt_server{"www.YOUR_MQTT_SERVER.com"};
```

3. Upload de code met de juiste MQTT instellingen.
4. Voeg de juist MQTT topics in op Home Assistant

5. Wanneer “WiFi Setup” verschijnt op het scherm kan u verbinding maken via WiFi.
6. Verbind met het netwerk “bloempot-1” WW: passpass



Established connection	N/A	Established connection	N/A
Mode	AP_STA(4)	Mode	AP_STA(4)
IP	0.0.0.0	IP	0.0.0.0
GW	0.0.0.0	GW	0.0.0.0
Subnet mask	0.0.0.0	Subnet mask	0.0.0.0
SoftAP IP	172.217.28.1	SoftAP IP	172.217.28.1
AP MAC	7C:9E:BD:FB:02:ED	AP MAC	7C:9E:BD:FB:02:ED
STA MAC	7C:9E:BD:FB:02:EC	STA MAC	7C:9E:BD:FB:02:EC
Channel	1	Channel	1
dBm	0	dBm	0
Chip ID	60418	Chip ID	60418
CPU Freq.	240MHz	CPU Freq.	240MHz
Flash size	4194304	Flash size	4194304
Free memory	260012	Free memory	260012

7. Gebruik het hamburger icon om te navigeren naar “Configure new AP”
8. Kies hier het gewenste netwerk en vul het wachtwoord in.
9. De bloempot is nu verbonden met het gekozen netwerk.
10. Kies de juiste plantensoort in het Home Assistant dashboard.
11. De meetwaarden zijn zichtbaar op het scherm en op het dashboard van Home Assistant, de bloempot zal u plant verzorgen

Conclusie:

Tijdens het realiseren van het project zal er rekening gehouden worden met alle eisen die gesteld werden. Deze eisen kunnen natuurlijk veranderen doorheen de ontwikkeling van het product.
Het eindresultaat zal een volwaardig product zijn dat klaar is voor gebruik. Het product zal mensen helpen met het onderhouden van hun plant. Ook mensen zonder groene vingers zullen hiermee hun kennis kunnen bijschaven. Het plan van aanpak helpt ons een mooi overzicht te schetsen.
Na enkele weken werken aan het project, hebben we ons concept grondig uitgewerkt.
Onze belangrijkste user stories zijn vervuld; namelijk:

- Omgeving van de plant meten
- Gebruiksvriendelijk
- Parameters van de plant meten
- Compatibel met verschillende planten
- Cloud storage (database op internet)

CODE

Uitleg

```
#define LILYGO_T5_V213

#pragma region Includes
#include <Arduino.h>
```

WiFi Built-In door Arduino

```
#include <WiFi.h>
#include <WebServer.h>
```

AutoConnect door Hieromon Ikasamo

```
#include <AutoConnect.h>
```

PubSubClient door Nick O'Leary

```
#include <PubSubClient.h>
#include <HTTPClient.h>
```

GxEPD door Jean-Marc Zingg

```
#include <GxEPD.h>
#include <GxGDEH0213B73/GxGDEH0213B73.h>
#include <GxIO/GxIO_SPI/GxIO_SPI.h>
#include <GxIO/GxIO.h>

#include <boards.h>
#include <SPI.h>
#include <Wire.h>
```

Adafruit Unified Sensor door Adafruit

```
#include <DHT.h>
```

Adafruit GFX Library door Adafruit

```
#include <gfxfont.h>
#pragma endregion Includes

#pragma region Define
```

De PHP server waar de opgemeten waarde worden opgeslagen

```
const char* serverName = "https://vincentsomers.sinners.be/post-esp-
data.php";
String apiKeyValue = "tPmAT5Ab3j7F9";
```

Bodemvochtigheid sensor
De pin waarde de bodemvochtigheid sensor aanhangt

```
#define soilSensorPIN A4
```

Water sensor
De pin waar de water niveau sensor wordt

```
#define waterLvlPIN A0
```

De pin waar de water niveau sensor wordt geactiveerd

```
#define enableWaterlvl 33
```

De pin waarde de LDR sensor aanhangt

```
#define ldrPin 34
```

De pin waar de mosfet van de water pomp aahangt

```
#define mosfetPin 19
```

MQTT
De poort om verbinding te maken met de MQTT broker.

```
#define mqtt_port 1883
```

De gebruiker die toegang heeft met de MQTT broker.

```
#define MQTT_USER "mqtt-user"
```

Het wachtwoord van de gebruiker die toegang heeft met de MQTT broker.

```
#define MQTT_PASSWORD "mqqt-user-wrong-user"
```

Pas het nummer aan bij elke topic als je de bloempot anders wilt noemen via MQTT

```
#define MQTT_MESSAGE_PLANTTYPE "SmartBloempot/1/Info/plantType"
#define MQTT_MESSAGE_THEME "SmartBloempot/1/Info/thema"
#define MQTT_MESSAGE_SOILMOISTURE "SmartBloempot/1/Sensors/grondVochtigheidPer
centage"
#define MQTT_MESSAGE_AIRMOISTURE "SmartBloempot/1/Sensors/luchtVochtigheid"
#define MQTT_MESSAGE_TEMPERATURE "SmartBloempot/1/Sensors/temperatuur"
#define MQTT_MESSAGE_LDR "SmartBloempot/1/Sensors/lichtWaarde"
#define MQTT_MESSAGE_WATERLEVEL "SmartBloempot/1/Sensors/waterNiveau"

#define MQTT_NOTIFICATION_SOILMOISTURE "SmartBloempot/1/Info/Notificaties/gron
dVochtigheid"
#define MQTT_NOTIFICATION_AIRMOISTURE "SmartBloempot/1/Info/Notificaties/lucht
Vochtigheid"
#define MQTT_NOTIFICATION_TEMPERATURE "SmartBloempot/1/Info/Notificaties/tempe
ratuur"
#define MQTT_NOTIFICATION_LDR "SmartBloempot/1/Info/Notificaties/lichtWaarde"
#define MQTT_NOTIFICATION_WATERLEVEL "SmartBloempot/1/Info/Notificaties/waterN
iveau"

#pragma endregion Define

#pragma region GlobaleVariabelen
```

Het aantal seconden tussen sensor lezingen

```
const int secondsBetweenSensorUpdates {
  2
};

String plantTypeStart{"Niks gekregen"}; // De begin waarde van het planttype
String plantType{plantTypeStart}; // Het ingegeven plant type

bool themaDayNight{false}; // Het kleur schema van de display, als dit false i
s is het scherm wit op zwart, als dit true is is het scherm zwart op wit.

DHT dht(12, DHT22);
float humidity{}, temperature{}; // De DHT gelezen waarde worden opgeslagen in
 deze twee variabele
```

```
int waterLvl{}, eenLiter{2240}, ldrValue{}; // Het waterniveau en de waarde va
n 1 liter worden hier opgeslagen, plus de gelezen waarde van de LDR.

const int airValue{3500}, waterValue {1300};

int soilMoistureValue{}, soilMoisturePercentage{100}; // De waarde van de bode
mvochtigheid sensor en het percentage daarvan.

int neededSoilMoisturePercentage{60}, neededAirMoistureValue{}, neededTemperat
ureValue{21}, neededLdrValue{};

// De waarde die worden vergeleken om een notificatie te krijgen als er een wa
arde te hoog of te laag is.
const int neededSoilMoisturePercentageNotificationGrayZone{10};
const int neededAirMoistureValueNotificationGrayZone{};
const int neededTemperatureValueNotificationGrayZone {2};
const int neededLdrValueNotificationGrayZone{};

const int neededSoilMoisturePercentageTemplateLow {5};
const int neededSoilMoisturePercentageTemplateMedium{20};
const int neededSoilMoisturePercentageTemplateHigh{35};

const int neededAirMoisturePercentageTemplateLow{5};
const int neededAirMoisturePercentageTemplateMedium{20};
const int neededAirMoisturePercentageTemplateHigh{35};

const int neededLdrValueTemplateLow{};
const int neededLdrValueTemplateMedium{};
const int neededLdrValueTemplateHigh{};

const int waterLvlEmpty{1550};
const int waterLvlHalf{1750};
const int waterLvlFull{2000};

// WiFi
// De start waarde voor WiFi, heir wordt ook een AP aangemaakt om daardoor een
 WiFi SSID mee te geven.
WiFiClient wifiClient;
WebServer Server;
AutoConnect portal(Server);
AutoConnectConfig configAC("", "passpass");
String apName{"Smartbloempot-2"};

// MQTT
// De MQTT broker waar de gelezen waarde van de plant worden meegegeven om de
daarna deze te tonen op Home Assistant.
PubSubClient client(wifiClient);
const char* mqtt_server{"cuythi.duckdns.org"};
```

```
const int secondsBetweenUploads{10}; // De aantal seconden tussen MQTT publish
es.

// De start variabelen op de Epaper display te doen werken.
GxIO_Class io(SPI,  EPD_CS, EPD_DC,  EPD_RSET);
GxEPD_Class display(io, EPD_RSET, EPD_BUSY);
U8G2_FOR_ADAFRUIT_GFX u8g2Fonts;
const int screenWidth{};
const int screenHeight{};

#pragma endregion GlobaleVariabelen

#pragma region FunctieDeclaraties
void setupWiFi(); // Deze functie zorgt ervoor dat de ESP een AP wordt en dat
men via deze AP een WiFI SSID kunnen ingeven om dan de ESP32 op hety internet
te hangen.
void rootPage();

void updateSensors(void *parameters); // In deze functie worden de sensoren va
riabelen geupdate met de nieuwe gelezen waarde
void giveWater(void* parameters); // Deze functie laat de waterpomp werken wan
neer de bodemvochtigheid te laag is.

// MQTT
void reconnect(); // Als de verbinden met de MQTT broker weg valt zal deze fun
ctie terug connectie maken
void callback(char *topic, byte *payload, unsigned int length); // Als er een
subscribe gebeurd via MQTT zal deze functie de juiste payload geven aan de jui
ste variabelen.
void publishDataViaMqtt(void *parameters); // Deze functie published de juiste
 waarde in de juiste topic.

// Display
void updateDisplay(void *parameters); // De update van de display gebeurd in d
eze functie met de juiste waarde.
#pragma endregion FunctieDeclaraties

void setup()
{
  Serial.begin(115200);
  pinMode(waterLvlPIN, INPUT);
  pinMode(enableWaterlvl, OUTPUT);
  dht.begin();

  display.init();
  u8g2Fonts.begin(display);
  u8g2Fonts.setFont(u8g2_font_helvR14_tf);
  display.setRotation(3);
```

```
  display.eraseDisplay();
  display.fillScreen(GxEPD_BLACK);

  u8g2Fonts.setForegroundColor(GxEPD_BLACK); // apply Adafruit GFX color
  u8g2Fonts.setBackgroundColor(GxEPD_WHITE); // apply Adafruit GFX color

  display.fillScreen(GxEPD_WHITE);

  setupWiFi();
  client.setServer(mqtt_server, mqtt_port);
  client.setCallback(callback);
  reconnect();

  pinMode(mosfetPin, OUTPUT);
  pinMode(ldrPin, INPUT);

  xTaskCreatePinnedToCore(
    publishDataViaMqtt,
    "publishDataViaMqtt",
    10240,
    NULL,
    3,
    NULL,
    0);

  xTaskCreatePinnedToCore(
    updateDisplay,
    "updateDisplay",
    1024,
    NULL,
    4,
    NULL,
    0);

  xTaskCreatePinnedToCore(
    giveWater,
    "giveWater",
    1024,
    NULL,
    1,
    NULL,
    0);
}

void loop()
{
  xTaskCreatePinnedToCore(
    updateSensors,
```

```
    "updateSensors",
    10240,
    NULL,
    2,
    NULL,
    1);

  vTaskDelay((secondsBetweenSensorUpdates * 1000) / portTICK_PERIOD_MS);

  if (WiFi.status() != WL_CONNECTED) {
    portal.handleClient();
  }
}

void setupWiFi()
{
  u8g2Fonts.setCursor(20, 30);
  u8g2Fonts.println("WIFI SETUP");

  u8g2Fonts.setCursor(20, 50);
  u8g2Fonts.println("Verbind met WiFi:");
  u8g2Fonts.setCursor(20, 70);
  u8g2Fonts.println("'" + apName + "'");

  u8g2Fonts.setCursor(20, 90);
  u8g2Fonts.setFont(u8g2_font_helvR10_tf);
  u8g2Fonts.println("Ga naar de hotspot website.");
  u8g2Fonts.setCursor(20, 110);
  u8g2Fonts.setFont(u8g2_font_helvR10_tf);
  u8g2Fonts.print("Druk op ");
  u8g2Fonts.setFont(u8g2_font_open_iconic_all_2x_t);
  u8g2Fonts.print("\u00db");
  u8g2Fonts.setFont(u8g2_font_helvR10_tf);
  u8g2Fonts.print(" > Configure new AP");

  display.update();

  Serial.println("setupWiFi");
  vTaskDelay(10 / portTICK_PERIOD_MS);

  configAC.apid = apName;
  portal.config(configAC);
  portal.begin();

  Server.on("/", rootPage);
  if (portal.begin()) {
    Serial.println("HTTP server:" + WiFi.localIP().toString());
  }
```

```
  // We start by connecting to a WiFi network
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.print(WiFi.localIP());
  Serial.println();
}

void rootPage() {
  char content[] = "hello, world";
  Server.send(200, "text/plain", content);
}

void updateSensors(void *parameters)
{
  //Serial.println("updateSensors enter");

  digitalWrite(enableWaterlvl, HIGH);
  vTaskDelay(200 / portTICK_PERIOD_MS);
  waterLvl = analogRead(waterLvlPIN);
  //Serial.println(waterLvl);
  digitalWrite(enableWaterlvl, LOW);

  ldrValue = analogRead(ldrPin);
  Serial.print("LDR: ");
  Serial.println(ldrValue);

  if (waterLvl <= waterLvlEmpty) {
    Serial.println("Low Water Level");
  }

  else if (waterLvl <= 1750) {
    Serial.println("Medium Water Level");
  }

  else if (waterLvl > 1750) {
    Serial.println("High Water Level");
  }

  vTaskDelay(1000 / portTICK_PERIOD_MS);

  soilMoistureValue = analogRead(soilSensorPIN);
  soilMoisturePercentage = map(soilMoistureValue, airValue, waterValue, 0, 100
);
  Serial.print("Moistervalue: ");
  Serial.print(soilMoistureValue);
  Serial.print(", ");
  Serial.print(soilMoisturePercentage);
  Serial.println("%");
```

```
  humidity = dht.readHumidity();
  temperature = dht.readTemperature();

  Serial.print("Humidity: ");
  Serial.println(humidity);
  Serial.print("Temperature: ");
  Serial.println(temperature);

  Serial.println(plantType);

  vTaskDelete(NULL);
}

void giveWater(void* parameters) {
  for (;;) {
    Serial.println("Pompen loop");
    if (soilMoisturePercentage < neededSoilMoisturePercentage) {
      Serial.println("Pompen!!!!!");
      digitalWrite(mosfetPin, HIGH);
      vTaskDelay(2000 / portTICK_PERIOD_MS);
      digitalWrite(mosfetPin, LOW);
    }
    vTaskDelay(10000 / portTICK_PERIOD_MS);
  }
  vTaskDelete(NULL);
}

void reconnect()
{
  Serial.println("reconnect");
  // Loop until we're reconnected
  while (!client.connected())
  {
    Serial.print("Attempting MQTT connection...");
    // Create a random client ID
    String clientId = "ESP32Client-";
    clientId += String(random(0xffff), HEX);
    // Attempt to connect
    if (client.connect(clientId.c_str(), MQTT_USER, MQTT_PASSWORD))
    {
      Serial.println("connected");

      client.subscribe(MQTT_MESSAGE_PLANTTYPE);
      client.subscribe(MQTT_MESSAGE_THEME);
    }
    else
    {
```

```
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      vTaskDelay(5000 / portTICK_PERIOD_MS);
    }
  }
}

// MQTT callback functie
void callback(char* p_topic, byte* p_payload, unsigned int p_length)
{
  // concat the payload into a string
  String payload;
  for (uint8_t i = 0; i < p_length; i++) {
    payload.concat((char)p_payload[i]);
  }
  // handle message topic
  if (String(MQTT_MESSAGE_PLANTTYPE).equals(p_topic)) {
    Serial.println(payload);
    plantType = payload;
    // test if the payload is equal to "ON" or "OFF"
    if (payload.equals(String("-")) || plantType == plantTypeStart) {
      neededSoilMoisturePercentage = NULL;
      neededLdrValue = NULL;
      neededTemperatureValue = NULL;
      neededAirMoistureValue = NULL;
    } else if (payload.equals(String("Caladium"))) {
      neededSoilMoisturePercentage = neededSoilMoisturePercentageTemplateLow;
      neededLdrValue = neededLdrValueTemplateHigh;
      neededTemperatureValue = 21;
      neededAirMoistureValue = neededAirMoisturePercentageTemplateHigh;
    } else if (payload.equals(String("Strelitzia"))) {
      neededSoilMoisturePercentage = neededSoilMoisturePercentageTemplateHigh;
      neededLdrValue = neededLdrValueTemplateHigh;
      neededTemperatureValue = 0;
      neededAirMoistureValue = neededAirMoisturePercentageTemplateMedium;
    } else if (payload.equals(String("Calathea"))) {
      neededSoilMoisturePercentage = neededSoilMoisturePercentageTemplateHigh;
      neededLdrValue = NULL;
      neededTemperatureValue = 10;
      neededAirMoistureValue = neededAirMoisturePercentageTemplateHigh;
    } else if (payload.equals(String("Bonsai Ficus Ginseng"))) {
      neededSoilMoisturePercentage = neededSoilMoisturePercentageTemplateHigh;
      neededLdrValue = neededLdrValueTemplateHigh;
      neededTemperatureValue = 15;
      neededAirMoistureValue = neededAirMoisturePercentageTemplateHigh;
    } else if (payload.equals(String("Alocasia Portadora"))) {
```

```
      neededSoilMoisturePercentage = neededSoilMoisturePercentageTemplateLow;
      neededLdrValue = NULL;
      neededTemperatureValue = 10;
      neededAirMoistureValue = neededAirMoisturePercentageTemplateHigh;
    } else if (payload.equals(String("Opuntia bleu ale"))) {
      neededSoilMoisturePercentage = neededSoilMoisturePercentageTemplateLow;
      neededLdrValue = neededLdrValueTemplateMedium;
      neededTemperatureValue = -10;
      neededAirMoistureValue = neededAirMoisturePercentageTemplateLow;
    } else if (payload.equals(String("Echeveria Paarse Parel"))) {
      neededSoilMoisturePercentage = neededSoilMoisturePercentageTemplateMediu
m;
      neededLdrValue = neededLdrValueTemplateMedium;
      neededTemperatureValue = 5;
      neededAirMoistureValue = neededAirMoisturePercentageTemplateMedium;
    } else if (payload.equals(String("Anthurium Clarinervium S"))) {
      neededSoilMoisturePercentage = neededSoilMoisturePercentageTemplateLow;
      neededLdrValue = neededLdrValueTemplateMedium;
      neededTemperatureValue = 5;
      neededAirMoistureValue = neededAirMoisturePercentageTemplateMedium;
    }
  }

  else if (String(MQTT_MESSAGE_THEME).equals(p_topic)) {
    if (payload.equals(String("on"))) {
      themaDayNight = true;
    } else {
      themaDayNight = false;
    }
  }
}

void publishDataViaMqtt(void *parameters)
{
  //Serial.println("publishDataViaMqtt enter");
  for (;;)
  {
    //Serial.println("publishDataViaMqtt loop");
    if (!client.connected())
    {
      reconnect();
    }
    client.loop();

    String tempStrM{String(soilMoisturePercentage)};
    char tempCharArrM[tempStrM.length() + 1];
    tempStrM.toCharArray(tempCharArrM, tempStrM.length() + 1);
    client.publish(MQTT_MESSAGE_SOILMOISTURE, tempCharArrM);
```

```
    String tempStrH{String(humidity)};
    char tempCharArrH[tempStrH.length() + 1];
    tempStrH.toCharArray(tempCharArrH, tempStrH.length() + 1);
    client.publish(MQTT_MESSAGE_AIRMOISTURE, tempCharArrH);

    String tempStrT{String(temperature)};
    char tempCharArrT[tempStrT.length() + 1];
    tempStrT.toCharArray(tempCharArrT, tempStrT.length() + 1);
    client.publish(MQTT_MESSAGE_TEMPERATURE, tempCharArrT);

    String tempStrLdr{String(ldrValue)};
    char tempCharArrLdr[tempStrLdr.length() + 1];
    tempStrLdr.toCharArray(tempCharArrLdr, tempStrLdr.length() + 1);
    client.publish(MQTT_MESSAGE_LDR, tempCharArrLdr);

    String tempStrWaterLvl{String(waterLvl)};
    char tempCharArrWaterLvl[tempStrWaterLvl.length() + 1];
    tempStrWaterLvl.toCharArray(tempCharArrWaterLvl, tempStrWaterLvl.length()
+ 1);
    client.publish(MQTT_MESSAGE_WATERLEVEL, tempCharArrWaterLvl);

    // Notificaties via MQTT
    if (neededTemperatureValue != NULL && (temperature > neededTemperatureValu
e + neededTemperatureValueNotificationGrayZone || temperature < neededTemperat
ureValue - neededTemperatureValueNotificationGrayZone )) {
      if (temperature > neededTemperatureValue + 2)
        client.publish(MQTT_NOTIFICATION_TEMPERATURE, "De temperatuur moet lag
er!");
      else if (temperature < neededTemperatureValue - 2)
        client.publish(MQTT_NOTIFICATION_TEMPERATURE, "Warmer AUB!");
    }

    if (neededSoilMoisturePercentage != NULL && (soilMoisturePercentage > need
edSoilMoisturePercentage + neededSoilMoisturePercentageNotificationGrayZone ||
 soilMoisturePercentage < neededSoilMoisturePercentage -
 neededSoilMoisturePercentageNotificationGrayZone )) {
      if (soilMoisturePercentage > neededSoilMoisturePercentage + neededSoilMo
isturePercentageNotificationGrayZone)
        client.publish(MQTT_NOTIFICATION_SOILMOISTURE, "HELP! Ik verdrink!");
      else if (soilMoisturePercentage < neededSoilMoisturePercentage -
 neededSoilMoisturePercentageNotificationGrayZone )
        client.publish(MQTT_NOTIFICATION_SOILMOISTURE, "Geef me water!");
    }

    if (neededAirMoistureValue != NULL && (airValue > neededAirMoistureValue +
 neededAirMoistureValueNotificationGrayZone || airValue < neededAirMoistureVal
ue - neededAirMoistureValueNotificationGrayZone )) {
```

```
      if (airValue > neededAirMoistureValue + neededAirMoistureValueNotificati
onGrayZone)
        client.publish(MQTT_NOTIFICATION_AIRMOISTURE, "Heel vochtig hier! Zet
me droger.");
      else if (airValue < neededAirMoistureValue -
 neededAirMoistureValueNotificationGrayZone )
        client.publish(MQTT_NOTIFICATION_AIRMOISTURE, "Het is hier precies een
 woestijn! Plaats me vochtiger.");
    }

    if (neededLdrValue != NULL && (ldrValue > neededLdrValue + neededLdrValueN
otificationGrayZone || ldrValue < neededLdrValue -
 neededLdrValueNotificationGrayZone )) {
      if (ldrValue > neededLdrValue + neededLdrValueNotificationGrayZone )
        client.publish(MQTT_NOTIFICATION_LDR, "Ik zie het licht niet! Plaats m
e in de zon.");
      else if (ldrValue < neededLdrValue - neededLdrValueNotificationGrayZone)
        client.publish(MQTT_NOTIFICATION_LDR, "Aaaah mijn blaadjes! Plaats me
in de schaduw.");
    }

    if (waterLvl != NULL && waterLvl < waterLvlEmpty) {
      client.publish(MQTT_NOTIFICATION_WATERLEVEL, "Water niveau laag! Vul me
met water.");
    }

    const int neededLdrValueNotificationGrayZone {}; // ??????????????????????
????????????????????????????????????????????????????

    if (WiFi.status() == WL_CONNECTED) {
      HTTPClient http;

      // Your Domain name with URL path or IP address with path
      http.begin(serverName);

      // Specify content-type header
      http.addHeader("Content-Type", "application/x-www-form-urlencoded");

      // Prepare your HTTP POST request data
      String httpRequestData = "api_key=" + apiKeyValue + "&bodemvochtigheid="
 + String(soilMoisturePercentage)
                             + "&waterlvl=" + String(waterLvl) + "&humidity=
" + String(humidity) + "&temperature=" + String(temperature) + "&ldr=" + Strin
g(temperature) + "";
      //Serial.print("httpRequestData: ");
      //Serial.println(httpRequestData);

      // Send HTTP POST request
```

```
      int httpResponseCode = http.POST(httpRequestData);

      // If you need an HTTP request with a content type: text/plain
      //http.addHeader("Content-Type", "text/plain");
      //int httpResponseCode = http.POST("Hello, World!");

      // If you need an HTTP request with a content type: application/json, us
e the following:
      //http.addHeader("Content-Type", "application/json");
      //int httpResponseCode = http.POST("{\"value1\":\"19\",\"value2\":\"67\"
,\"value3\":\"78\"}");

      //      if (httpResponseCode > 0) {
      //        Serial.print("HTTP Response code: ");
      //        Serial.println(httpResponseCode);
      //      }
      //      else {
      //        Serial.print("Error code: ");
      //        Serial.println(httpResponseCode);
      //      }
      // Free resources
      http.end();

      vTaskDelay((secondsBetweenUploads * 1000) / portTICK_PERIOD_MS);
    }
  }
  vTaskDelete(NULL);
}
void updateDisplay(void *parameters)
{
  //Serial.println("updateDisplay enter");
  for (;;)
  {
    //Serial.println("updateDisplay loop");
    u8g2Fonts.setFontMode(1);                  // use u8g2 transparent mode (t
his is default)
    //u8g2Fonts.setFontDirection(1);             // left to right (this is def
ault)
    if (themaDayNight) {
      u8g2Fonts.setForegroundColor(GxEPD_BLACK); // apply Adafruit GFX color
      u8g2Fonts.setBackgroundColor(GxEPD_WHITE); // apply Adafruit GFX color

      display.fillScreen(GxEPD_WHITE);
    } else {
      u8g2Fonts.setForegroundColor(GxEPD_WHITE); // apply Adafruit GFX color
      u8g2Fonts.setBackgroundColor(GxEPD_BLACK); // apply Adafruit GFX color

      display.fillScreen(GxEPD_BLACK);
```

```
    }

    u8g2Fonts.setFont(u8g2_font_open_iconic_all_2x_t);
    u8g2Fonts.setCursor(GxEPD_HEIGHT / 2 - 25, GxEPD_WIDTH - 90);
    u8g2Fonts.println("\u00af ");
    u8g2Fonts.setFont(u8g2_font_helvR14_tf);
    u8g2Fonts.setCursor(GxEPD_HEIGHT / 2, GxEPD_WIDTH - 90);
    u8g2Fonts.print(soilMoisturePercentage);
    u8g2Fonts.print("%");

    u8g2Fonts.setFont(u8g2_font_open_iconic_all_2x_t);
    u8g2Fonts.setCursor(GxEPD_HEIGHT / 2 - 25, GxEPD_WIDTH - 70);
    u8g2Fonts.println("\u00a8 ");
    u8g2Fonts.setFont(u8g2_font_helvR14_tf);
    u8g2Fonts.setCursor(GxEPD_HEIGHT / 2, GxEPD_WIDTH - 70);
    u8g2Fonts.print(temperature);
    u8g2Fonts.print("°C");

    u8g2Fonts.setFont(u8g2_font_open_iconic_all_2x_t);
    u8g2Fonts.setCursor(GxEPD_HEIGHT / 2 - 25, GxEPD_WIDTH - 50);
    u8g2Fonts.println("\u0098 ");
    u8g2Fonts.setFont(u8g2_font_helvR14_tf);
    u8g2Fonts.setCursor(GxEPD_HEIGHT / 2, GxEPD_WIDTH - 50);
    u8g2Fonts.print(humidity);
    u8g2Fonts.println("%");

    u8g2Fonts.setFont(u8g2_font_open_iconic_all_2x_t);
    u8g2Fonts.setCursor(GxEPD_HEIGHT / 2 - 25, GxEPD_WIDTH - 30);
    u8g2Fonts.println("\u008f ");
    u8g2Fonts.setFont(u8g2_font_helvR14_tf);
    u8g2Fonts.setCursor(GxEPD_HEIGHT / 2, GxEPD_WIDTH - 30);
    u8g2Fonts.print(waterLvl);
    u8g2Fonts.print("ml");

    String tempPlantType{plantType};
    tempPlantType.toUpperCase();
    u8g2Fonts.setCursor(GxEPD_HEIGHT / 2 -
((tempPlantType.length() / 2) * 12), GxEPD_WIDTH - 10);
    u8g2Fonts.print(tempPlantType);

    display.update();

    //    u8g2Fonts.setCursor(220, 15);
    //    u8g2Fonts.print(neededSoilMoisturePercentage);
    //
    //    u8g2Fonts.setCursor(220, 35);
    //    u8g2Fonts.print(neededTemperatureValue);
    //
```

```
    //    u8g2Fonts.setCursor(220, 55);
    //    u8g2Fonts.print(neededAirMoistureValue);

    //Serial.println("updateDisplay end");
    vTaskDelay(10000 / portTICK_PERIOD_MS);
  }
  vTaskDelete(NULL);
}
```