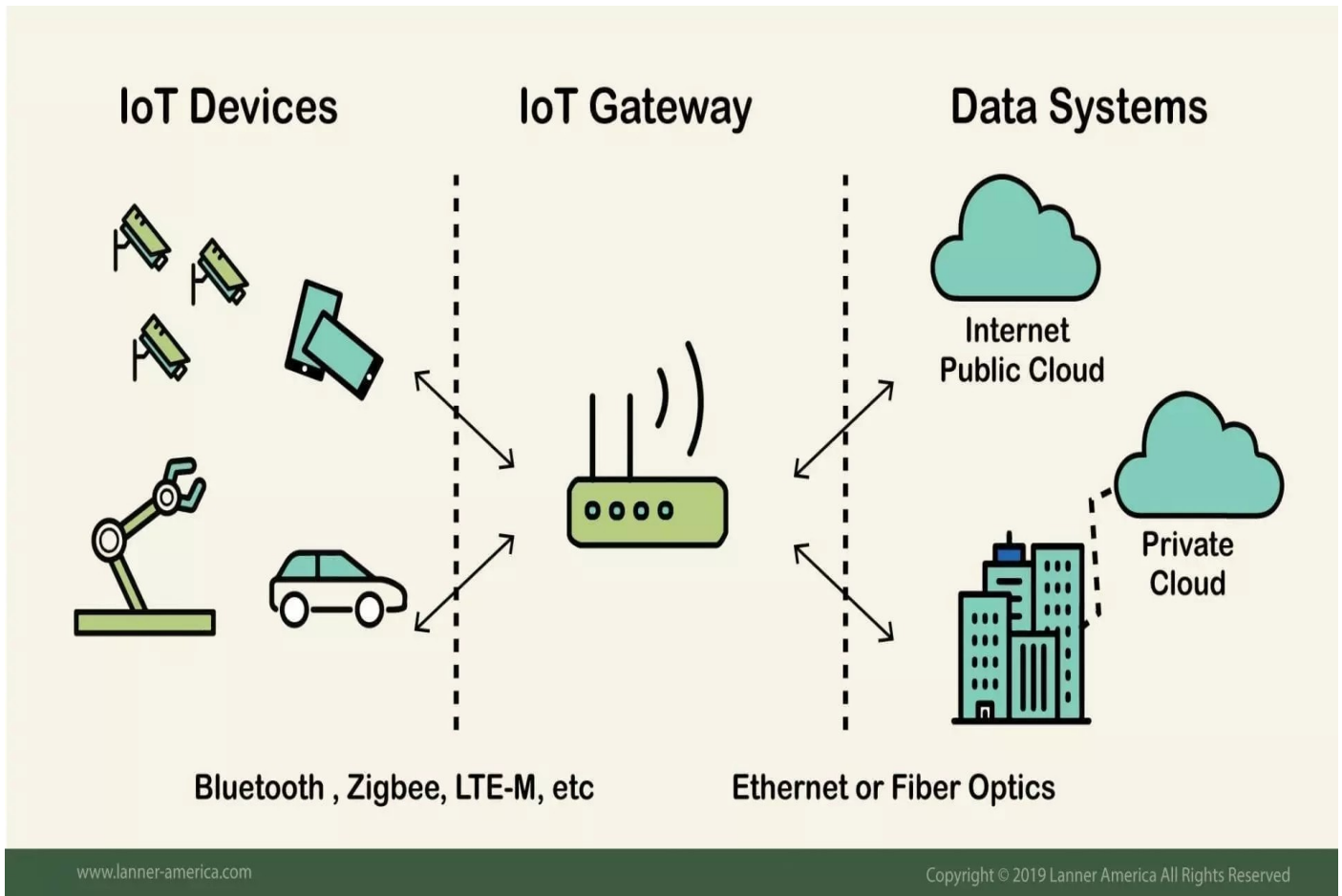


Projektu finala



Egileak: Jeyson Rueda eta Martin Malaxetxebarria

Ikasturtea: 2020-2021



1. orrialdea

lan hau **IURRETA LHII**k sortu du eta [Creative Commons lizentzia](#) baten mende dago.



Ziklo bukaerako proiektua: aurkibidea

1. Erronkaren Planteamendua.....	4
2. Hausnarketa.....	5
3. Bloke diagrama.....	7
3.1 Softwarearen bloke-diagrama.....	7
3.2 Hardwarearen bloke-diagrama.....	7
4. TTGO Lora.....	8
4.1. Sarrera.....	8
4.2. Pinout.....	8
4.3 Kontrolagailuaren plaka arduinon zelan instalatu.....	9
4.4. OLED liburutegiak instalatzea.....	10
4.5. LoRa liburutegia instalatzea.....	12
4.6. Estekak.....	12
5. Temperatura eta hezetasun sentsorea.....	13
5.1. Sarrera.....	13
5.2. Eskema elektronikoak.....	13
5.3. Sentsorearen edo shieldaren pinout.....	14
5.4. Oinarrizko programa.....	14
5.5. Oinarrizko programa funtzioak bihurtuta.....	16
5.6. Material zerrenda.....	17
5.7. Gorabeherak.....	17
5.8. Estekak.....	17
6. CO2 sentsorea.....	18
6.1. Sarrera.....	18
6.2. Eskema elektronikoak.....	18
6.3. Sentsorearen edo shieldaren pinout.....	19
6.4. Oinarrizko programa.....	19
6.5. Oinarrizko programa funtzioak bihurtuta.....	20
6.6. Material zerrenda.....	21
6.7. Gorabeherak.....	21
6.8. Estekak.....	21
7. Led Neopixel.....	22
7.1. Sarrera.....	22
7.2. Eskema elektronikoak.....	22
7.3. Sentsorearen edo shieldaren pinout.....	23
7.4. Oinarrizko programa.....	23
7.5. Oinarrizko programa funtzioak bihurtuta.....	24
7.6. Material zerrenda.....	25
7.7. Gorabeherak.....	26
7.8. Estekak.....	26
8. HLK-PM03 Elikatze-iturria.....	27
8.1. Sarrera.....	27
8.2. Zehastapen teknikoak.....	27
8.3. Pinout.....	28
8.4. Eskema elektrikoak.....	28
8.5. Estekak.....	28
9. TTGO-ESP32 LORA PINOUT.....	29
10. Eskema elektrikoak (KiCad).....	34
11. GateWay konfigurazioa.....	36

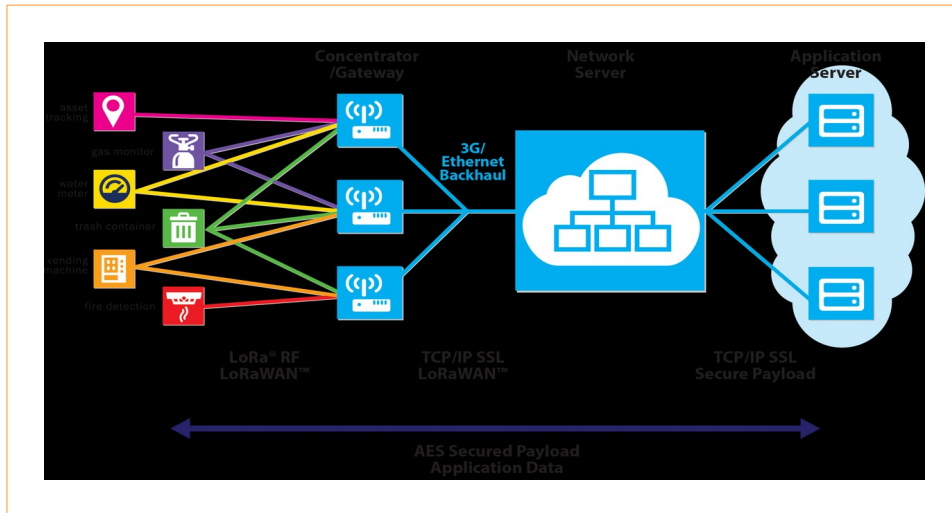


12.Programa Nagusia.....	38
12.1 Programa nagusia Jeyson eta Martin.....	38
12.2 CCS881 sentsorea.....	38
12.3.DTH11 sentsorea.....	39
12.4.Komunikazioa TTN.....	40
12.5.Led Neopixel.....	41
12.6.Lora Gateway Header.....	42
12.8.Oled pantaila.....	44
12.9.The Things Network.....	45
13.KiCad eskemak.....	50
13.1.Kicad sch.....	50
13.2.KiCad pcb.....	50
14.Sistemaren Argazkiak.....	51
15.Gorabeherak.....	52
16.Hobetzeko proposamenak.....	53



1. Erronkaren Planteamendua

CO2, hezetasuna eta temperatura neurketa egin eta balio horiek led edota panataila bidez erakutsi, honekin batera balioa horiek The Things Network web orrialdean erakutsi Lore Gateway batekin. Raspberry batekin datu horiek ere eskolako zerbitzarian batean erakutsi.



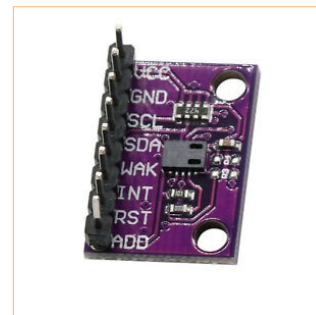
2. Hausnarketa

DHT11: DHT11 tenperatura eta hezetasuna neurtzen dituen sentsore simple bat da; nahiko merkea, baina modulu batean muntatuta ere aurki dezakezu (PCB batean muntatuta, errazago erabiltzeko), Arduinorentzat horrelako osagai elektronikoetan ohikoa den bezala. Plakaren kasuan, 5 kilo omioko pull-up erresistentzia eta funtzionamenduaren berri ematen digun Led bat ditu. Fidagarritasun eta egonkortasun handia du, kalibratutako seinale digitalaren ondorioz.

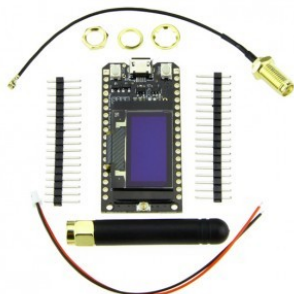


CCS811: CCS811 kontsumo baxuko aire-kalitatearen sentsore bat da. Oxide metalezko sentsore bat (MOX) erabiltzen du osagai lurrunkor organiko (TVOC) eta CO2 eduki baliokideen guztizko baliokideak neurtzeko. Moduluak mikro kontrolagailu bat (MCU) barne hartzen du, gasaren irakurketak zuzentzeko, tenperaturaren eta giro-hezetasunaren arabera.

- Energia hornidura: 3.3V gomendatuak
- Energia kontsumoa: 60mW-tik gora
- ETVOC detekzio-eremua: 0 # 32768ppb
- ECO2 detektatzeko eremua: 400 # 32768ppm
- Baselinazko zuzenketa automatikoa geruza oxido sentikor metalikoaren denbora: 24H
- Konfigurazioa baimendu eta sentsore bat irakurri boterearen ondoren: 20 minutu gutxienez.



TTGO Lora oled: Wifi modulu ezagun hau bateragarria da Arduinorekin, eta hari gabeko konexioa eskaintzen du, modu guztiz errazean. ESP32 OLEDe bi gauzak plaka bakar eta simple batean konbinatzen ditu: ESP32 modulua eta plaka berean dagoen OLED pantaila bat. Gainera, mikroUSB konektore bat dauka elikadurarako eta Arduinoko IDEtik zuzenean programatzeko. Plakak LoRa modulu bat ere badu, datuak distantzia handietara bi noranzkotan transmititzeko aukera ematen duena.



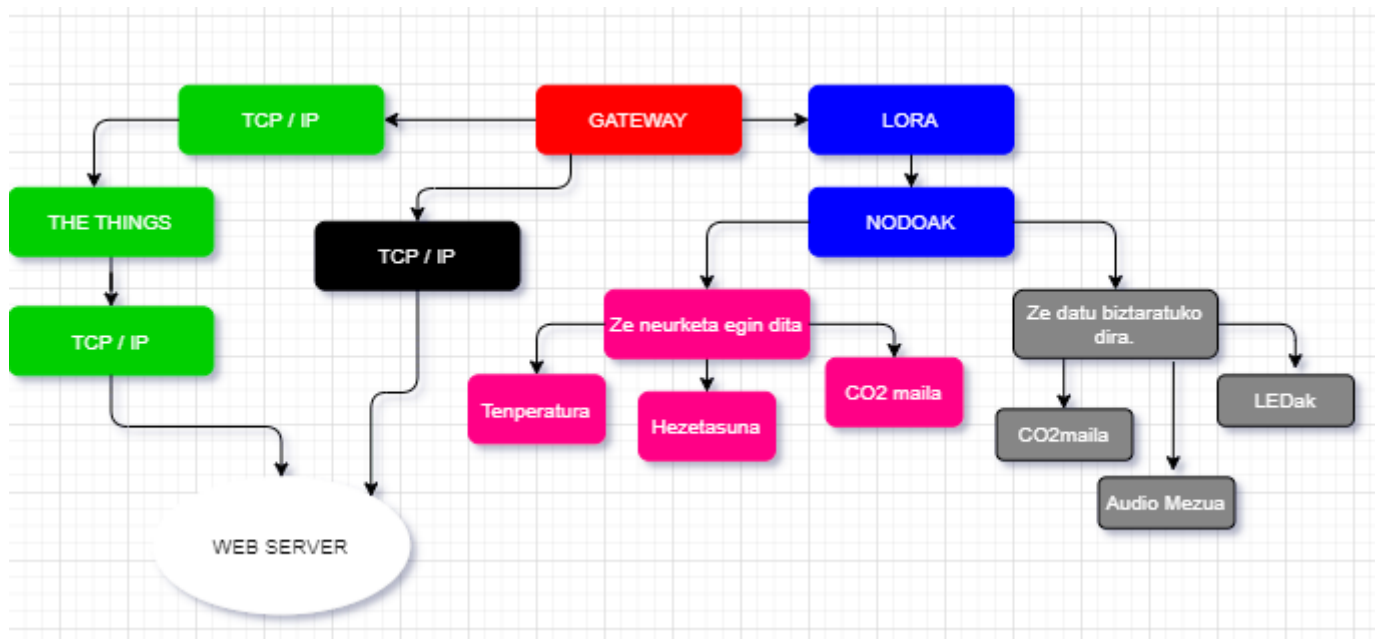
HLK-PM03 : DC-DC doigarria Step-down elikadura-modulua.

- Sarrera tentsioa: 60 V-220 V
- Irteera tentsioa: 3,3 V- 5V
- Bihurketaren eraginkortasuna:% 90

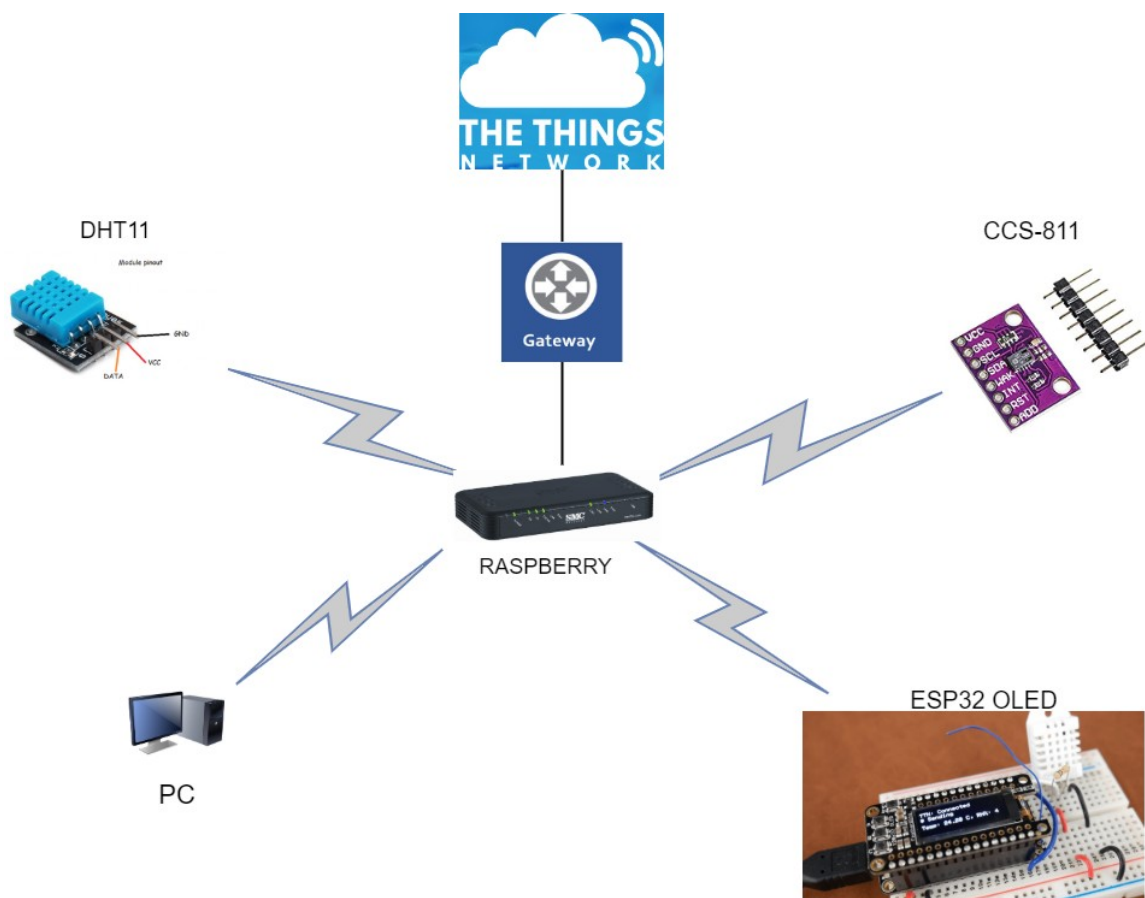


3. Bloke diagrama

3.1 Softwarearen bloke-diagrama



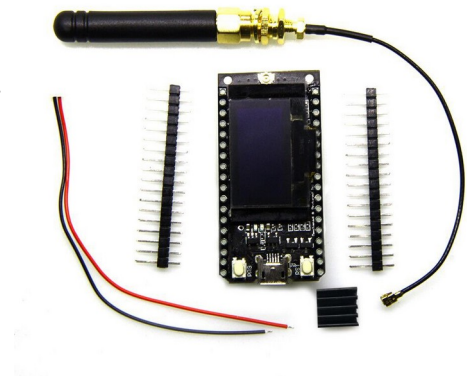
3.2 Hardwarearen bloke-diagrama



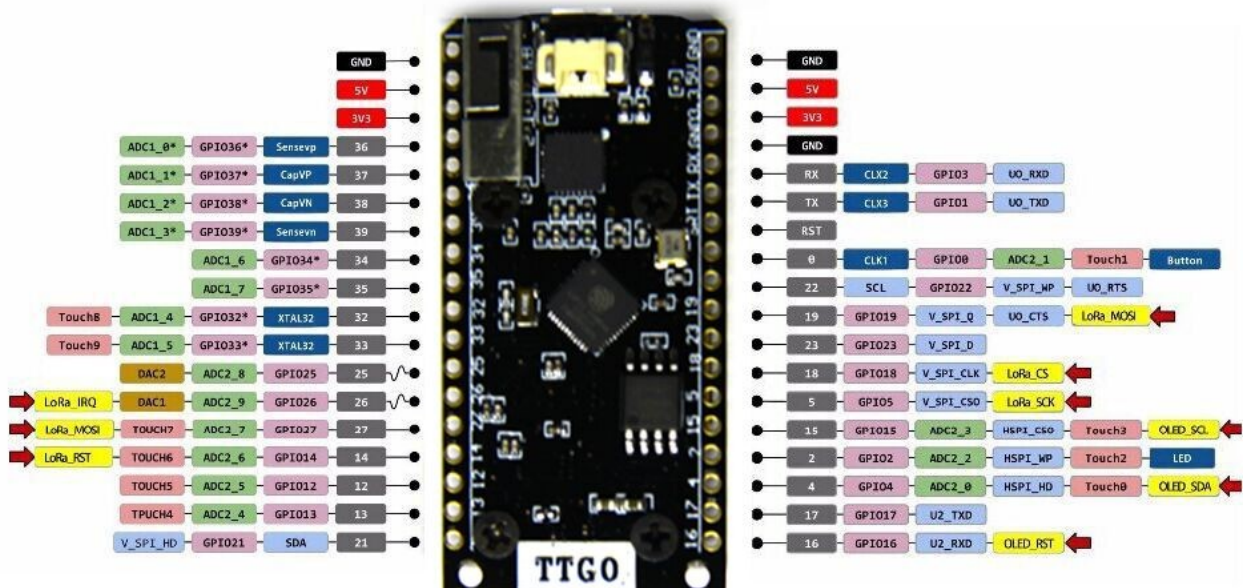
4.TTGO Lora

4.1.Sarrera

ESP32 OLEdek ESP32 modulua eta plaka berean OLED pantaila konbinatzen ditu, wifi eta bluetooth 4.0 konektibitatea eskainiz. Gainera, mikroUSB konektorea dauka elikatze eta JST konektorea plaka bera elikatze. Arduinoko IDEtik zuzenean programatu daiteke. Plakak LoRa modulu bat ere badu, datuak distantzia handietara bi noranzkotan transmititzea ahalbidetzen duena, The Things Network sarera konektatzeko aukerarekin.



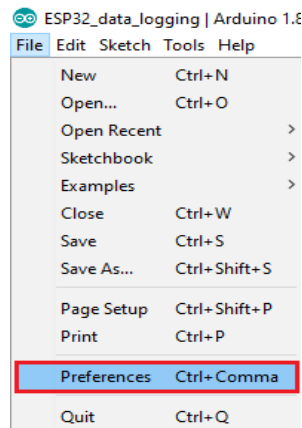
4.2.Pinout



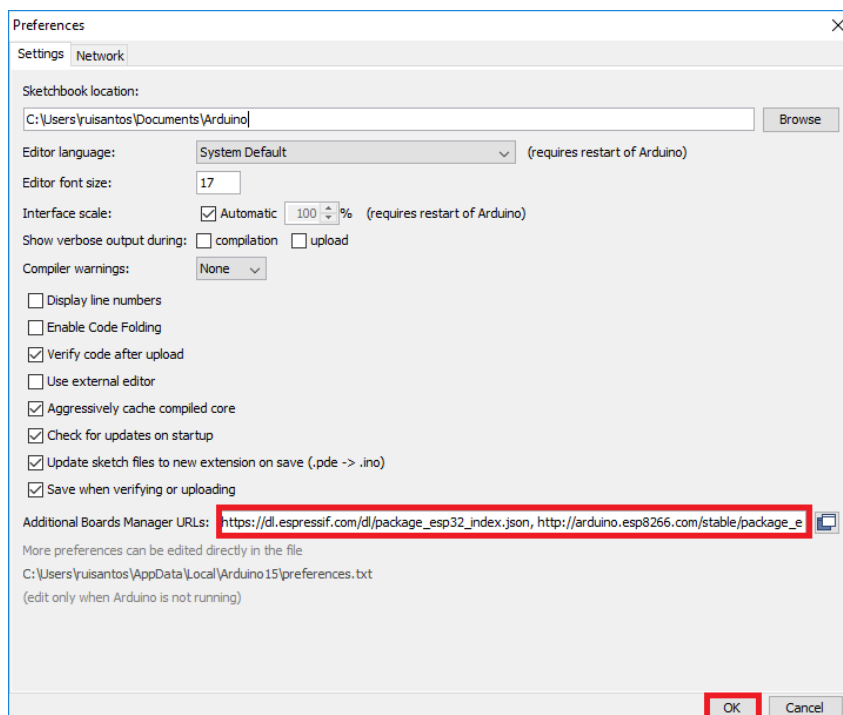
4.3 Kontrolagailuaren plaka arduinon zelan instalatu

ESP32 plaka zure Arduino IDEn instalatzeko, jarraitu jarraibide hauek:

1. Arduinoko zure IDEan, joan Archivo > Preferencias

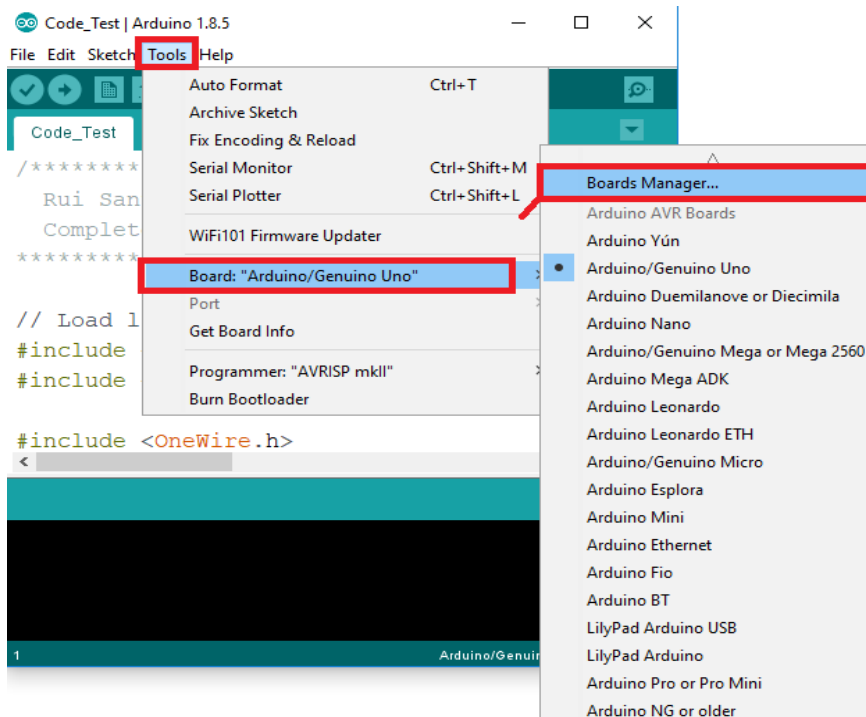


2. Sartu https://dl.espressif.com/dl/package_esp32_index.json "taula-administratzailearen URL gehigarriak" eremuan, hurrengo irudian agertzen den bezala. Gero, egin klik "onartu" botoian:

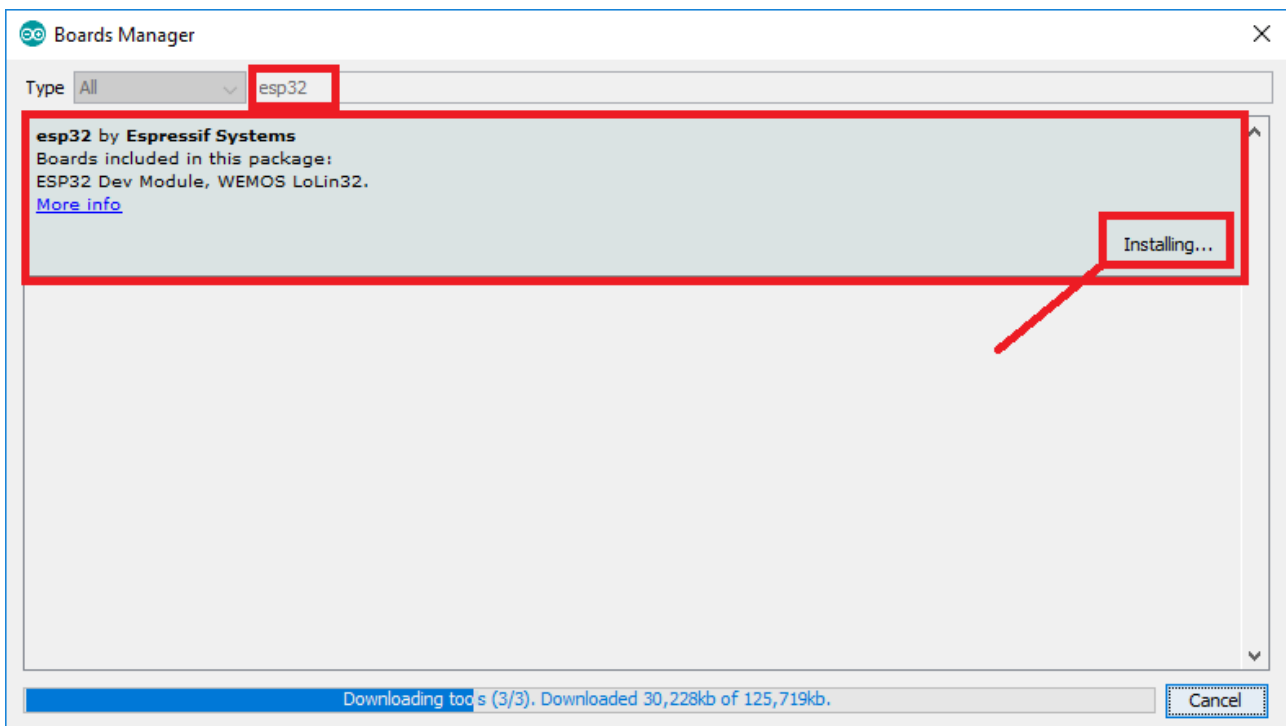


3. Ireki taula administratzailea. Joan Herramientas > Tablero > Administrador de tableros...





4. Bilatu ESP32 eta sakatu "ESP32 by Espressif Systems" instalazio-botoia:



5. Instalatu eta ondoren, plaka berri hori instalatuta egongo lirake.

Bertan utziko dizuet nik jarraitutako pusuak plaka hau instalatzeko :

<https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/>

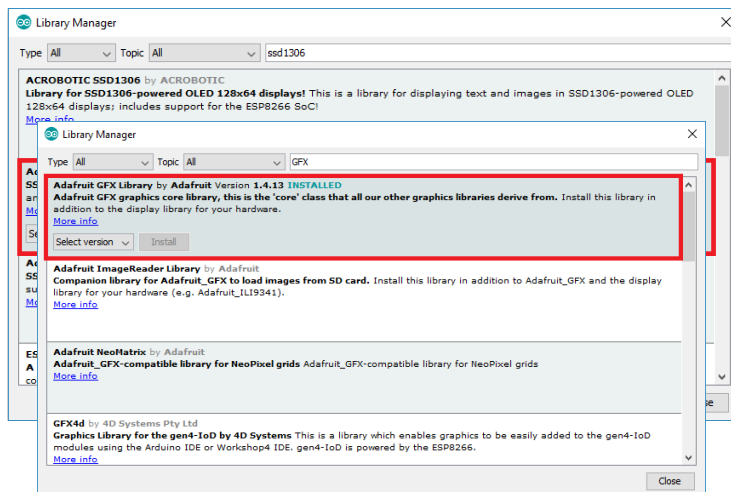
4.4.OLED liburutegiak instalatzea



Liburutegi batzuk daude eskuragarri OLED pantaila ESP32 sistemarekin kontrolatzeko. Tutorial honetan Adafruit-eko bi liburutegi erabiliko ditugu: Adafruit_SSD1306 liburutegia eta Adafruit_GFX liburutegia.

Jarraitu hurrengo urratsak liburutegi horiek instalatzeko.

1. Ireki ezazu Arduinoren IDE eta joan Sketch > Incluir biblioteca > Administrar bibliotecas. Liburutegiko administratzailea ireki beharko litzateke.
2. Idatzi "SSD1306" bilaketa-koadroan eta instalatu Adafruit-eko SSD1306 liburutegia.

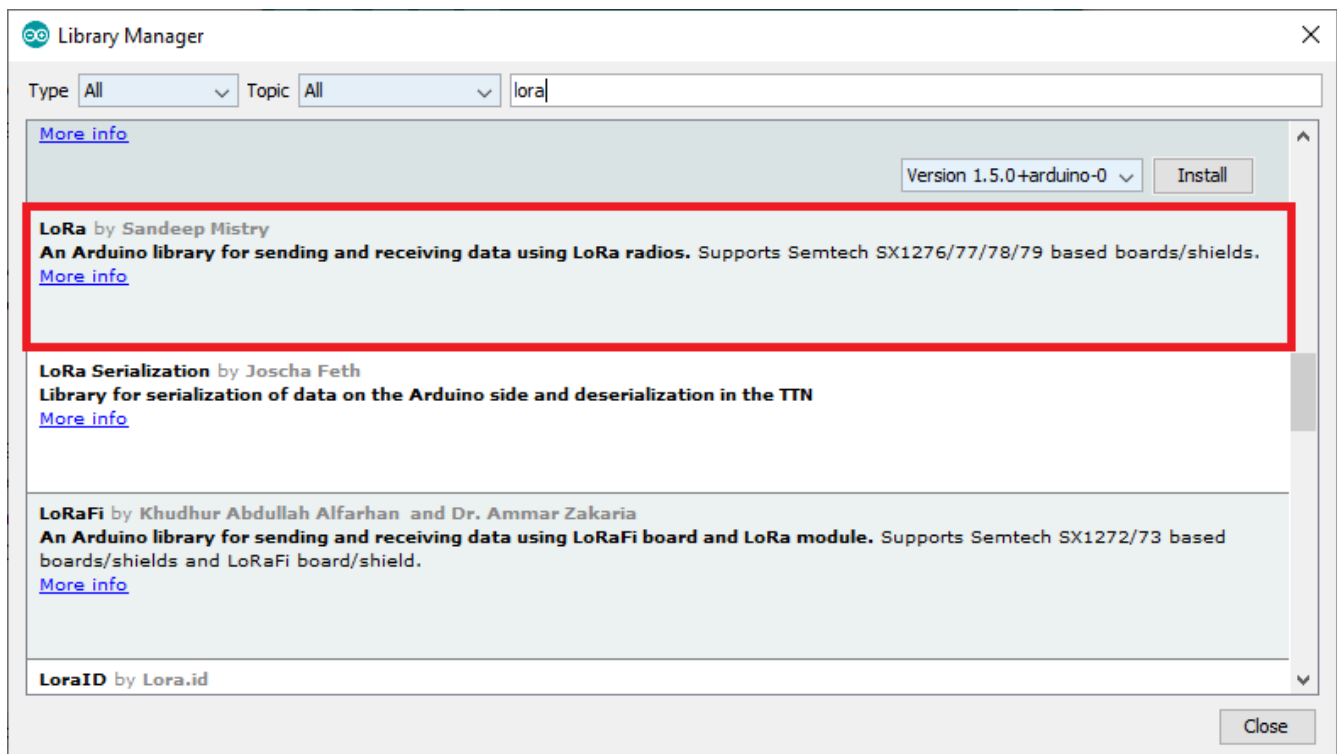


3. Adafruit-eko SSD1306 liburutegia instalatu ondoren, idatzi "GFX" bilaketa-koadroan eta instalatu liburutegia.



4.5. LoRa liburutegia instalatzea

Ireki zure Arduinoko IDE eta joan Sketch > Incluir biblioteca > Administrar bibliotecas eta bilatu "lora". Hautatu hurrengo irudian nabarmendutako LoRa liburutegia eta instalatu.



Liburutegiak instalatu ondoren, Arduino itxi eta zabaldu berriro.

4.6. Estekak:

<https://randomnerdtutorials.com/ttgo-lora32-sx1276-arduino-ide/>

<https://akirasan.net/nodo-lorawan-con-esp32/>

<https://www.sensora-e.com/inicio/2363-modulo-ttgo-lora-v20-esp32.html#:~:text=TTGO%20LORA%20OLED%20V2,-0%20ESP32%20868&text=La%20ESP32%20OLED%20combina%20el,desde%20el%20IDE%20de%20Arduino.>

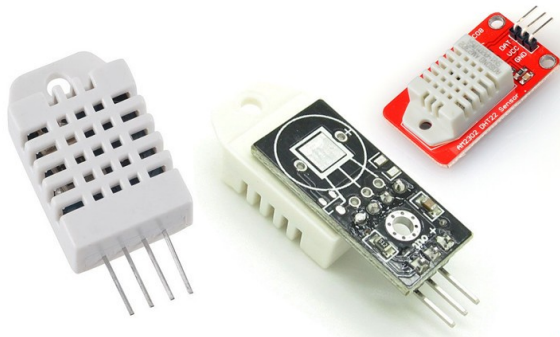


5. Temperatura eta hezetasun sentsoarea

5.1. Sarrera

DHT11 tenperatura eta hezetasun sentsoare digital bat da, errendimendu onekoa eta kostu txikikoa.

Hezetasun sentsoare kapazitibo bat eta inguruko airea neurtzeko termistor bat integratzen ditu, eta datuak seinale digital baten bidez erakusten ditu.



Sensor solo

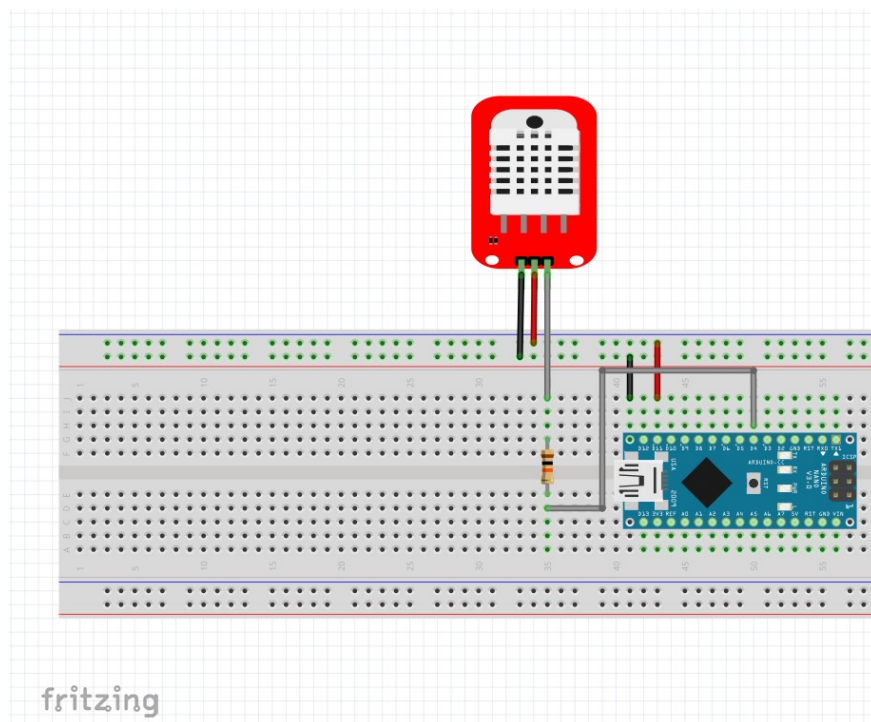
Sensor + R_{pull-up} + C_{filtrado}

Datasheet: <https://www.alldatasheet.com/datasheet-pdf/pdf/1132459/ETC2/DHT22.html>

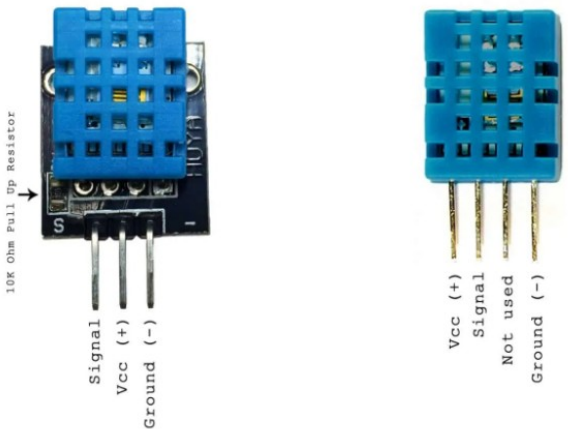
Zehaztapen teknikoak:

DHT11		DHT22
0 - 50°C / ± 2°C	Temperature Range	-40 - 125 °C / ± 0.5 °C
20 - 80% / ± 5%	Humidity Range	0 - 100 % / ± 2-5%
1Hz (one reading every second)	Sampling Rate	0.5 Hz (one reading every two seconds)
15.5mm x 12mm x 5.5mm	Body Size	15.1mm x 25mm x 7.7mm
3 - 5V	Operating Voltage	3 - 5V
2.5mA	Max Current During Measuring	2.5mA

5.2. Eskema elektronikoak



5.3.Sentsorearen edo shieldaren pinout

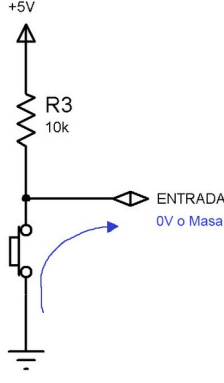


10K Ohm Pull Up Resistor

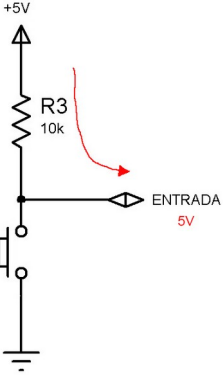
Signal
Vcc (+)
Ground (-)

Vcc (+)
Signal
Not used
Ground (-)

Gu ezkerrekoa daukagu eta horrek 10 K Ω Pull up erresistentzia bat dauka.



ENTRADA
0V o Masa



ENTRADA
5V

Resistencia de Pull-Up

1 2

5.4.Oinarrizko programa

```

/*
Programaren izena: DHTtesterJeysonetaMartin.ino
Egilea: Elektronika    Data: 2021/02/22
Zeregina: Dht sentsorea balioak irakurri eta portu seriean erakutsi
Adibide hau domeinu publikokoa da.
Programaren egoera: Egiaztatze/egiaztatuta
Egiaztatutako plakak: UNO ?? Nano Bai MEGA ???
*/

// Connect pin 1 (ezkerrean) of the sensor to +5V
//3.3V logika duen taula bat Arduino Due batek pin 1 konektatzen badu
//5V ordez, 3.3V!
//Zure DHTPIN DOKTOREARI
//X/Connect pin 4 (eskuinaldean)
///Connect a 10K resistor from pin 2 (datuak) to pin 1 (boterea)

//DHT sentsorea.
//Liburutegi honen bertsio zaharragoek hirugarren parametro bat eraman zuten.
//Parametro hau ez da beharrezkoa prozesadore azkarragoentzat.
//DHT irakurgailuaren algoritmoa prozesu azkarragoetan lan egitera egokitzen da.

/*-----( LIBURUTEGIAK )-----*/
#include "DHT.h"
// - DHT Sensor Library: https://github.com/adafruit/DHT-sensor-library
// - Adafruit Unified Sensor Lib: https://github.com/adafruit/Adafruit\_Sensor

/*-----( KONSTANTEAK )-----*/
#define DHTPIN 4      // Pin digitala DTH Kkonektatua
#define DHTTYPE DHT11 // DHT 11
#define BAUDRATE 9600 //Komunikaziorako datuen abiadura
#define ITXARON 2000 //Itxaroteko denbora

/*-----( OBJETUAK )-----*/
DHT dht(DHTPIN, DHTTYPE);

```



```

void setup() {
  Serial.begin(BAUDRATE);
  Serial.println(F("DHTxx test!"));

  dht.begin();
}

void loop() {
  // Wait a few seconds between measurements.
  delay(1000);

  // Reading temperature or humidity takes about 250 milliseconds!
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();
  // Read temperature as Celsius (the default)
  float t = dht.readTemperature();
  // Read temperature as Fahrenheit (isFahrenheit = true)
  float f = dht.readTemperature(true);

  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }

  // Compute heat index in Fahrenheit (the default)
  float hif = dht.computeHeatIndex(f, h);
  // Compute heat index in Celsius (isFahreheit = false)
  float hic = dht.computeHeatIndex(t, h, false);

  Serial.print(F("Humidity: "));
  Serial.print(h);
  Serial.print(F("%   Temperature: "));
  Serial.print(t);
  Serial.print(F("°C "));
  Serial.print(f);
  Serial.print(F("°F   Heat index: "));
  Serial.print(hic);
  Serial.print(F("°C "));
  Serial.print(hif);
  Serial.println(F("°F"));
}

```

```

DHTxx test!
Humidity: 56.00%   Temperature: 27.50°C 81.50°F   Heat index: 28.39°C 83.10°F
Humidity: 34.00%   Temperature: 27.80°C 82.04°F   Heat index: 27.14°C 80.86°F
Humidity: 34.00%   Temperature: 27.80°C 82.04°F   Heat index: 27.14°C 80.86°F
Humidity: 33.00%   Temperature: 27.90°C 82.22°F   Heat index: 27.17°C 80.90°F
Humidity: 33.00%   Temperature: 27.90°C 82.22°F   Heat index: 27.17°C 80.90°F
Humidity: 32.00%   Temperature: 27.70°C 81.86°F   Heat index: 26.97°C 80.55°F
Humidity: 32.00%   Temperature: 27.80°C 82.04°F   Heat index: 27.04°C 80.68°F
Humidity: 32.00%   Temperature: 27.80°C 82.04°F   Heat index: 27.04°C 80.68°F
Humidity: 31.00%   Temperature: 27.70°C 81.86°F   Heat index: 26.93°C 80.47°F

```



5.5.Oinarritzko programa funtzioak bihurtuta

```
/* Funtzioa: int DTH11SentsoreaSetup (int zeinPin)
 * Zeregina: DTH11 Sentsorea irakurri
 * Erabilitako liburutegiak: DHT Sensor Library
 *                               Adafruit Unified Sensor Lib
 * Liburutegia nondik hartuta: https://github.com/adafruit/DHT-sensor-library (DHT
Sensor Library)
 *                               https://github.com/adafruit/Adafruit\_Sensor
(Adafruit Unified Sensor Lib)
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 * */
void DTH11SentsoreaSetup (void){
    Serial.println(F("DHTxx test!"));
    dht.begin();
}

/* Funtzioa: int DTH11SentsoreaLoop (int zeinPin)
 * Zeregina: DTH11 Sentsorea irakurri
 * Erabilitako liburutegiak: DHT Sensor Library
 *                               Adafruit Unified Sensor Lib
 * Liburutegia nondik hartuta: https://github.com/adafruit/DHT-sensor-library (DHT
Sensor Library)
 *                               https://github.com/adafruit/Adafruit\_Sensor
(Adafruit Unified Sensor Lib)
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 * */

void DTH11SentsoreaLoop (void){

    // Wait a few seconds between measurements.
    delay(2000);

    // Reading temperature or humidity takes about 250 milliseconds!
    // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
    float h = dht.readHumidity();
    // Read temperature as Celsius (the default)
    float t = dht.readTemperature();
    // Read temperature as Fahrenheit (isFahrenheit = true)
    float f = dht.readTemperature(true);

    // Check if any reads failed and exit early (to try again).
    if (isnan(h) || isnan(t) || isnan(f)) {
        Serial.println(F("Failed to read from DHT sensor!"));
        return;
    }

    // Compute heat index in Fahrenheit (the default)
    float hif = dht.computeHeatIndex(f, h);
    // Compute heat index in Celsius (isFahreheit = false)
    float hic = dht.computeHeatIndex(t, h, false);

    Serial.print(F("Humidity: "));
    Serial.print(h);
    Serial.print(F("%  Temperature: "));
    Serial.print(t);
    Serial.print(F("°C "));
    Serial.print(f);
    Serial.print(F("°F  Heat index: "));
```



```

Serial.print(hic);
Serial.print(F("°C "));
Serial.print(hif);
Serial.println(F("°F"));
}

```

```

DHTxx test!
CCS811 test
Humidity: 49.00%  Temperature: 22.90°C 73.22°F  Heat index: 22.52°C 72.55°F
CO2: 400ppm
TVOC: 0
Humidity: 49.00%  Temperature: 22.90°C 73.22°F  Heat index: 22.52°C 72.55°F
CO2: 422ppm
TVOC: 3

```

5.6. Material zerrenda

Izena	Kopurua	Oharrak edota ezaugarri bereziak
DTH 11 (Sentsorea)	1	Pull up resistentzia dauka barnean
Arduino Nano	1	Hau bakarrik probatarako

5.7. Gorabeherak

Lehenengo sentsorea apurtuta egon da eta beste bat eskatu dugu, azken hau bai zihoala.

5.8. Estekak

<https://howtomechatronics.com/tutorials/arduino/dht11-dht22-sensors-temperature-and-humidity-tutorial-using-arduino/>

<https://naylorlamechatronics.com/sensores-temperatura-y-humedad/58-sensor-de-temperatura-y-humedad-relativa-dht22-am2302.html>

<https://www.instructables.com/TTGO-ESP32-LoRa-Board-With-DHT22-Temperature-and-H/>

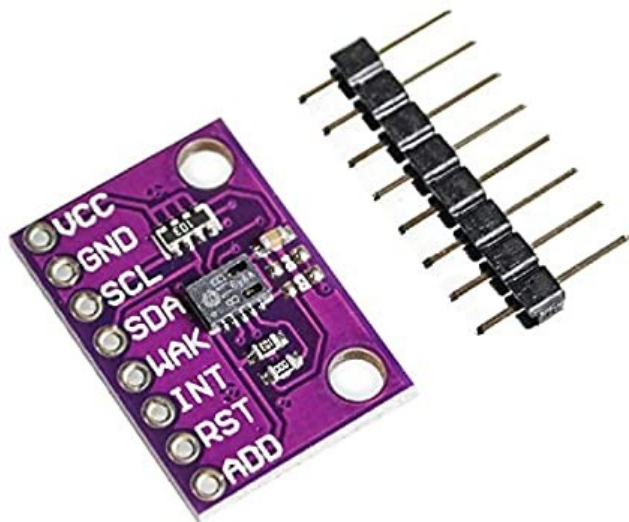
<https://aprendiendoarduino.wordpress.com/tag/dht/>



6.CO2 sentsoarea

6.1.Sarrera

CCS811 barneko airearen kalitatea neurtzeko sentsoire bat da, eta erraz erabil dezakegu Arduino bezalako prozesadore batekin batera. Barneko airearen kalitatea zehazteko, CCS811 MOX (Metal-Oxide) sentsoire multigas bat da, karbono monoxidoa (Co) eta konposatu lurrunkorak (VOCs) neurtzea barne hartzen duena, hala nola etanola, aminak edo hidrokarburo aromatikoak. Horien bidez, CCS811k karbono dioxido baliokidearen (eCO2) kantitatea zehaztu dezake. Neurketa-tartea 400 eta 8192 ppm artekoa da eCO2-n eta 0 eta 1187 ppb artekoa TVOCen.



Datasheet:<https://cdn-learn.adafruit.com/downloads/pdf/adafruit-ccs811-air-quality-sensor.pdf>

Zehaztapen teknikoak:

Airearen kalitatearen sentsoarea

MCU integratua

Oinarri programagarria

Bateria hedatuko bizitza baterako korrante-kontsumo txikia

Irteera: I2C bidea

Tentsio-hornidura: 1.8 V a 3.6 V

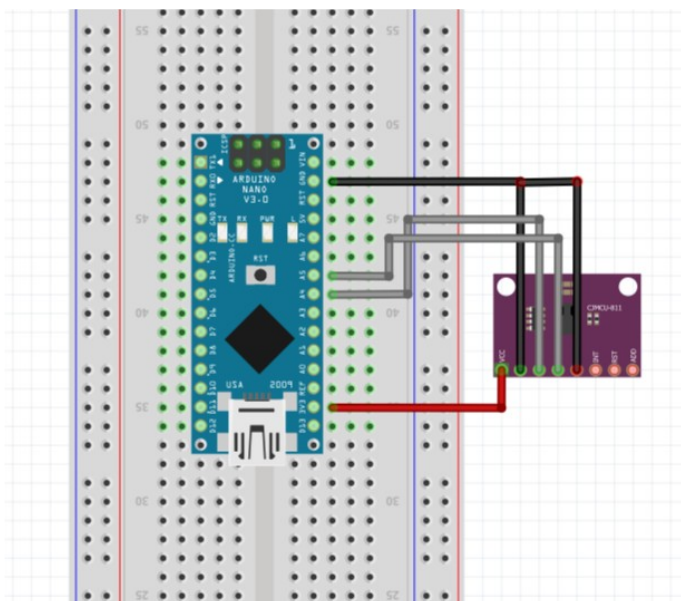
Energia-kontsumoa: 1.2 MW a 46 mW

Konposatu organiko lurrunkorak detektatzea, guztira (TVOC), 0 PPB eta 1187 PPB artean

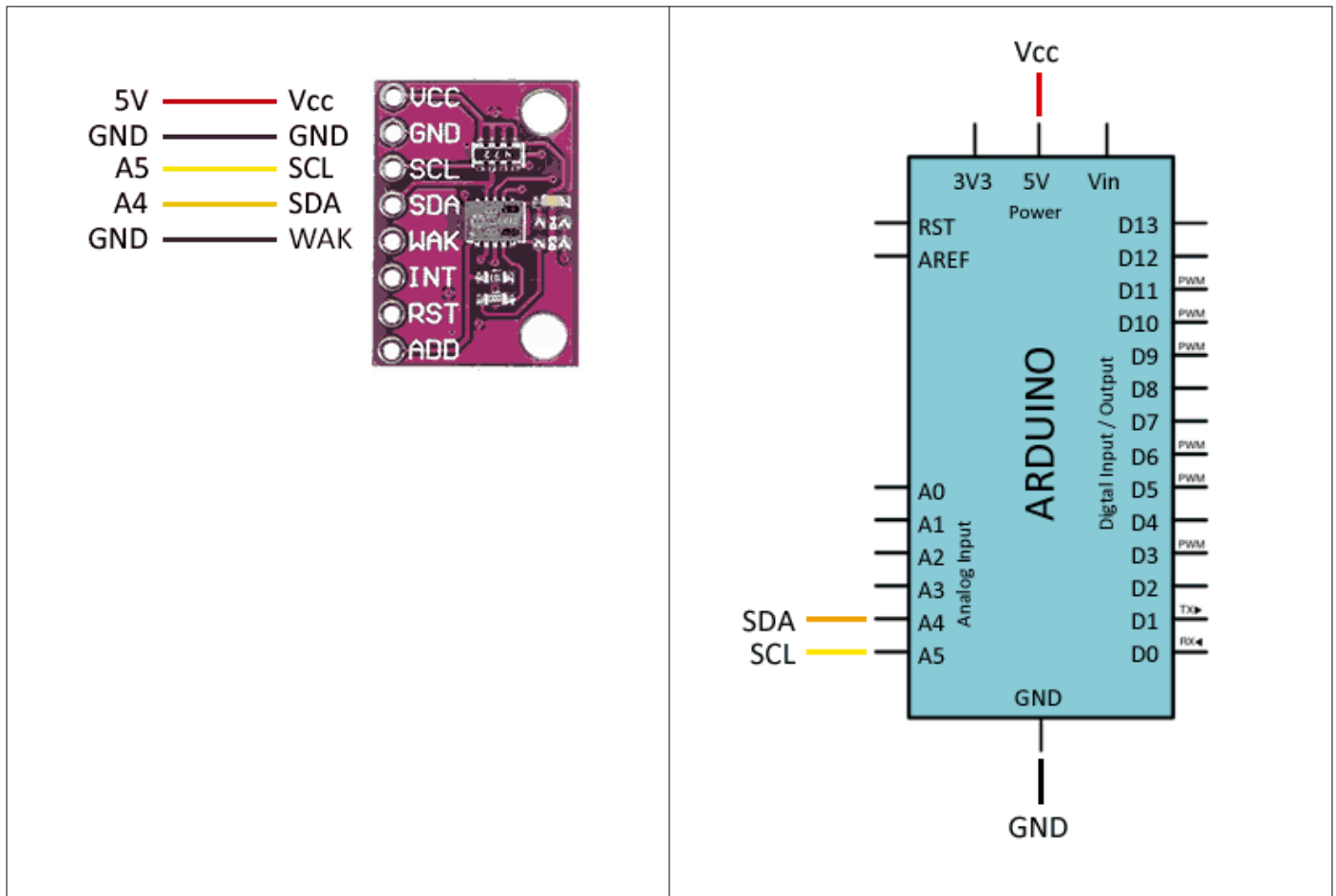
400 ppm-tik 8192 ppm-ra arteko eCO2 detektatzea

Bost funtzionamendu-modu

6.2.Eskema elektronikoak



6.3.Sentsorearen edo shieldaren pinout



6.4.Oinarrizko programa

```
/*  
Programaren izena: CCS881SentsoreaJeysonetaMartin.ino  
Egilea: Elektronika    Data: 2021/02/22  
Zeregina: CCS881 sentsorea balioak irakurri eta portu seriean erakutsi  
Adibide hau domeinu publikokoa da.  
Programaren egoera: Egiaztatzeko/Egiaztatuta  
Egiaztatutako plakak: UNO ?? Nano Bai MEGA ???  
*/  
  
/*-----( LIBURUTEGIAK )-----*/  
#include "Adafruit_CCS811.h"  
// - CCS881 Sensor Library: https://github.com/adafruit/Adafruit\_CCS811  
  
/*-----( KONSTANTEAK )-----*/  
#define BAUDRATE 9600 //Komunikaziorako datuen abiadura  
#define ITXARON 500 //Itxaroteko denbora  
  
/*-----( OBJETUAK )-----*/  
Adafruit_CCS811 ccs;  
  
void setup() {  
    Serial.begin(BAUDRATE);  
  
    Serial.println("CCS811 test");  
}
```



```

if(!ccs.begin()){
    Serial.println("Failed to start sensor! Please check your wiring.");
    while(1);
}

// Wait for the sensor to be ready
while(!ccs.available());
}

void loop() {
    if(ccs.available()){
        if(!ccs.readData()){
            //Serial.print(ccs.calculateTemperature());
            Serial.print("CO2: ");
            Serial.print(ccs.getCO2());
            Serial.println("ppm");
            Serial.print("TVOC: ");
            Serial.println(ccs.getTVOC());
        }
        else{
            Serial.println("ERROR!");
            while(1);
        }
    }
    delay(ITXARON);
}

```

```

CCS811 test
CO2: 400ppm
TVOC: 0
CO2: 400ppm
TVOC: 0

```

6.5.Oinarrizko programa funtzioak bihurtuta

```

/* Funtzioa: int CCS811SentsoreaSetup (int zeinPin)
 * Zeregina: DTH11 Sentsorea irakurri
 * Erabilitako liburutegiak: CCS811 Sensor Library
 * Liburutegia nondik hartuta: https://github.com/adafruit/Adafruit\_CCS811 (CCS811
Sensor Library)
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 * */
void CCS811SentsoreaSetup (void){

    Serial.println("CCS811 test");

    if(!ccs.begin()){
        Serial.println("Failed to start sensor! Please check your wiring.");
        while(1);
    }

    // Wait for the sensor to be ready
    while(!ccs.available());
}

/* Funtzioa: int CCS811SentsoreaLoop (int zeinPin)
 * Zeregina: DTH11 Sentsorea irakurri
 * Erabilitako liburutegiak: CCS811 Sensor Library

```



* Liburutegia nondik hartuta: https://github.com/adafruit/Adafruit_CCS811 (CCS881 Sensor Library)
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 * */

```
void CCS881SentsoreaLoop (void){
  if(ccs.available()){
    if(!ccs.readData()){
      //Serial.print(ccs.calculateTemperature());
      Serial.print("CO2: ");
      Serial.print(ccs.geteCO2());
      Serial.println("ppm");
      Serial.print("TVOC: ");
      Serial.println(ccs.getTVOC());
    }
    else{
      Serial.println("ERROR!");
      while(1);
    }
  }
  delay(500);
}
```

```
DHTxx test!
CCS811 test
Humidity: 49.00%  Temperature: 22.90°C 73.22°F  Heat index: 22.52°C 72.55°F
CO2: 400ppm
TVOC: 0
Humidity: 49.00%  Temperature: 22.90°C 73.22°F  Heat index: 22.52°C 72.55°F
CO2: 422ppm
TVOC: 3
```

6.6. Material zerrenda

Izena	Kopurua	Oharrak edota ezaugarri bereziak
CCS 881 (Sentsorea)	1	
Arduino Nano	1	Hau bakarrik probatarako

6.7. Gorabeherak

Ez dugu izan.

6.8. Estekak

<https://www.luisllamas.es/medir-calidad-del-aire-y-co2-con-ccs811-y-arduino/>

<https://github.com/gcoulby/FritzingParts/blob/master/CJMCU-811.fzpz>

<https://www.digikey.es/es/product-highlight/s/sparkfun/ccs811-indoor-air-quality-sensor-breakout-board>

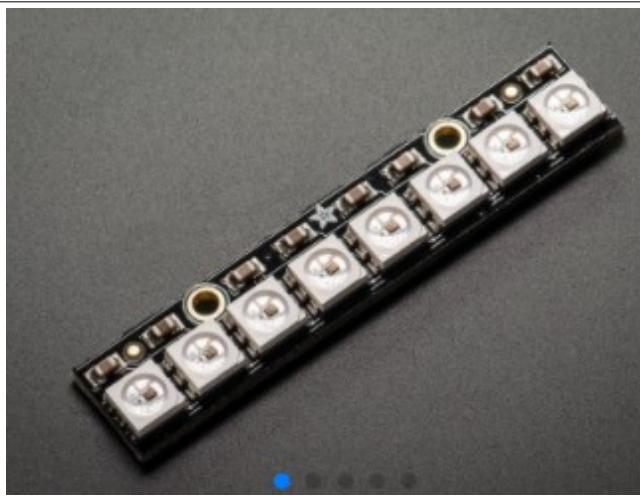


7.Led Neopixel

7.1.Sarrera

NeoPixelak Adafruit-ekoak dira eta 5050 motako Led diodoak dira, WS2812 kontrolagailu integratu batekin. Hainbat neurri eta formatan daude, eta dokumentazio zabala dute, Arduinorekin erabiltzeko prest daudenak. Oso erraza da minutu gutxitan martxan jartzea, pin bakarra behar baitute eta kolore distiratsuak eskaintzen baitituzte.

WS2812 argi-iturri integratua eskalagarri eta eskuragarriko Led baten bilaketan egindako azken aurrerapena da. Led gorri, berde eta urdinak kontrolagailu-txip batekin batera integratuta daude kable bakar baten bidez kontrolatutako gainazalean muntatzeko pakete txiki batean. Banaka erabil daitezke, kate luzeagoetan kateatu edo faktore interesgarriagoetan mihiztatu.



Zehaztapen teknikoak:

Forma: laukizuzena

Pixel kopurua: 8

Neurriak: 51 x 10 x 3mm

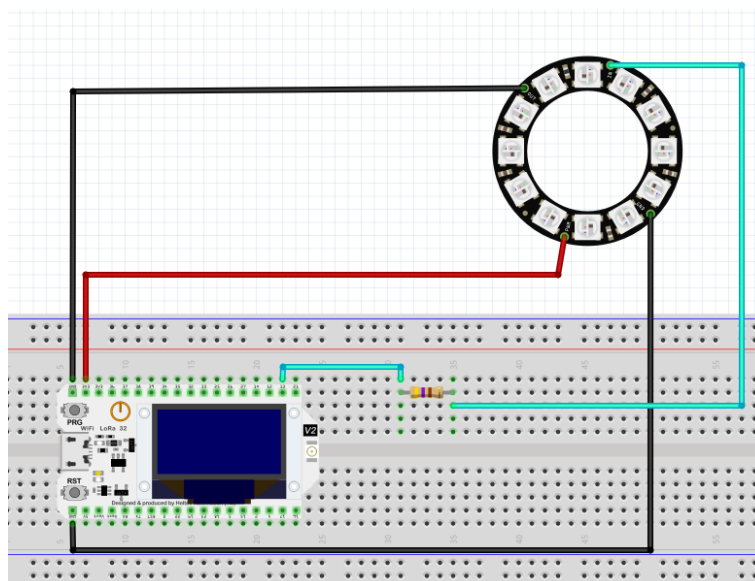
Pisua: 2.57 gramo

Elikadura: 5V

Kontsumoa: 18 mA pixel bakoitzeko

Datasheet: <https://cdn-shop.adafruit.com/datasheets/WS2812.pdf>

7.2.Eskema elektronikoak



7.3.Sentsorearen edo shieldaren pinout



7.4.Oinarritzko programa

```
// NeoPixel Ring simple sketch (c) 2013 Shae Erisson
// Released under the GPLv3 license to match the rest of the
// Adafruit NeoPixel library

#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
  #include <avr/power.h> // Required for 16 MHz Adafruit Trinket
#endif

// Which pin on the Arduino is connected to the NeoPixels?
#define PIN          6 // On Trinket or Gemma, suggest changing this to 1

// How many NeoPixels are attached to the Arduino?
#define NUMPIXELS 16 // Popular NeoPixel ring size

// When setting up the NeoPixel library, we tell it how many pixels,
// and which pin to use to send signals. Note that for older NeoPixel
// strips you might need to change the third parameter -- see the
// strandtest example for more information on possible values.
Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);

#define DELAYVAL 500 // Time (in milliseconds) to pause between pixels

void setup() {
  // These lines are specifically to support the Adafruit Trinket 5V 16 MHz.
  // Any other board, you can remove this part (but no harm leaving it):
  #if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
    clock_prescale_set(clock_div_1);
  #endif
  // END of Trinket-specific code.

  pixels.begin(); // INITIALIZE NeoPixel strip object (REQUIRED)
}

void loop() {
  pixels.clear(); // Set all pixel colors to 'off'

  // The first NeoPixel in a strand is #0, second is 1, all the way up
  // to the count of pixels minus one.
```



```

for(int i=0; i<NUMPIXELS; i++) { // For each pixel...

    // pixels.Color() takes RGB values, from 0,0,0 up to 255,255,255
    // Here we're using a moderately bright green color:
    pixels.setPixelColor(i, pixels.Color(0, 150, 0));

    pixels.show();    // Send the updated pixel colors to the hardware.

    delay(DELAYVAL); // Pause before next pass through loop
}
}

```



7.5.Oinarrizko programa funtzioak bihurtuta

```

/* Funtzioa: void LedNeoPixelSetup void()
 * Zeregina:
 * Erabilitako liburutegiak:
 * Liburutegia nondik hartuta:
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 * */
void NeoPixelSetup (void)
{
    #if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
    clock_prescale_set(clock_div_1);
    #endif
    // END of Trinket-specific code.

    pixels.begin(); // INITIALIZE NeoPixel strip object (REQUIRED)
}

/* Funtzioa: void LedNeoPixelLoop void()

```



```

* Zeregina: DTH11 Sentsorea irakurri
* Erabilitako liburutegiak:
* Liburutegia nondik hartuta:
* Bueltatzen duena: ezer
* Sartutako parametroak: ezer
* */

```

```

void NeoPixelLoop (void)
{
  CCS881SentsoreaLoop ();
  pixels.clear(); // Set all pixel colors to 'off'
  //for(int i=0; i<NUMPIXELS; i++) { // For each pixel...
  // pixels.Color() takes RGB values, from 0,0,0 up to 255,255,255
  // Here we're using a moderately bright green color:
  /*pixels.setPixelColor(0, pixels.Color(125, 255, 0));
  pixels.setPixelColor(1, pixels.Color(255, 0, 0));
  pixels.setPixelColor(2, pixels.Color(0, 255, 0));
  pixels.setPixelColor(3, pixels.Color(0, 0, 255));
  pixels.setPixelColor(4, pixels.Color(255, 125, 0));
  pixels.setPixelColor(5, pixels.Color(0, 255, 125));
  pixels.setPixelColor(6, pixels.Color(125, 0, 255));
  pixels.setPixelColor(7, pixels.Color(255, 0, 125));*/

  if (ccs.geteCO2()> 400 && ccs.geteCO2()< 750){
    pixels.setPixelColor(0, pixels.Color(0, 255, 0));
    pixels.setPixelColor(1, pixels.Color(0, 255, 0));
    pixels.setPixelColor(2, pixels.Color(0, 255, 0));
  }
  else
  {
    if (ccs.geteCO2()>= 750 && ccs.geteCO2()< 1200){
      pixels.setPixelColor(0, pixels.Color(0, 255, 0));
      pixels.setPixelColor(1, pixels.Color(0, 255, 0));
      pixels.setPixelColor(2, pixels.Color(0, 255, 0));
      pixels.setPixelColor(3, pixels.Color(255, 255, 0));
      pixels.setPixelColor(4, pixels.Color(255, 255, 0));
      pixels.setPixelColor(5, pixels.Color(255, 255, 0));
    }
    else
    {
      pixels.setPixelColor(0, pixels.Color(0, 255, 0));
      pixels.setPixelColor(1, pixels.Color(0, 255, 0));
      pixels.setPixelColor(2, pixels.Color(0, 255, 0));
      pixels.setPixelColor(3, pixels.Color(255, 255, 0));
      pixels.setPixelColor(4, pixels.Color(255, 255, 0));
      pixels.setPixelColor(5, pixels.Color(255, 255, 0));
      pixels.setPixelColor(6, pixels.Color(255, 0, 0));
      pixels.setPixelColor(7, pixels.Color(255, 0, 0));
    }
  }
  pixels.show(); // Send the updated pixel colors to the hardware.

  delay(100); // Pause before next pass through loop
}

```

7.6.Material zerrenda

Izena	Kopurua	Oharrak edota ezaugarri bereziak
Led Neopixel	1	
TTGO Lora	1	



7.7.Gorabeherak

Ez dugu izan.

7.8.Estekak

<https://learn.adafruit.com/adafruit-neopixel-uberguide/>

[https://tienda.bricogeek.com/led-neopixel/660-barra-neopixel-8-x-ws2812.html?](https://tienda.bricogeek.com/led-neopixel/660-barra-neopixel-8-x-ws2812.html?gclid=Cj0KCQiA4feBBhC9ARIsABp_nbVA6xkaQWV2733y1gMET1h4rGiRIGw8Mn2Gw73FIECJkmrDtVslwYkaAuBmEALw_wcB)

[gclid=Cj0KCQiA4feBBhC9ARIsABp_nbVA6xkaQWV2733y1gMET1h4rGiRIGw8Mn2Gw73FIECJkmrDtVslwYkaAuBmEALw_wcB](https://tienda.bricogeek.com/led-neopixel/660-barra-neopixel-8-x-ws2812.html?gclid=Cj0KCQiA4feBBhC9ARIsABp_nbVA6xkaQWV2733y1gMET1h4rGiRIGw8Mn2Gw73FIECJkmrDtVslwYkaAuBmEALw_wcB)



8.HLK-PM03 Elikatze-iturria

8.1.Sarrera

HLK-PM03 AC DC bihurgailua elikadura-iturri, botere-iturri edo iturri konmutatu deitzen den gailu elektronikoa bat da. Elektronikan, korrante alternoa irteera batean edo gehiagotan korrante zuzen bihurtzen duen tresna da.

HLK-PM03 AC DC bihurgailua espazio trinkoetan inplementatu daiteke, edo proiektuak ahalik eta txikiena izan behar badu, iturri hori adierazitakoa da.

Energia elektrikoa VCA izatetik VCD izatera aldatzean, hainbat gailu elektronikoa elikatu ditzake, hala nola CD motorra, led-ak, txartelak edo Arduino moduluak, wifi modulua, sentsoreak, eragingailuak, anplifikadoreak, zirkuitu integratuak, etab.



Datasheet: <https://www.mikrocontroller.net/attachment/349613/HLK-PM03.pdf>

8.2.Zehastapen teknikoak

Tentsioa sartzea (AC: 90 ~ 264 V).

Zarata-maila baxua eta kizkurra

Gainkargaren eta zirkuitulaburren aurkako babesa.

Eraginkortasun handia, potentzia-dentsitate handia

Produktua EMCren eta segurtasun-probaren baldintzak betetzeko diseinatuta dago.

Energia-kontsumo txikia, ingurumenaren babesa, karga-galerarik gabe < 0,1 W.

Segurtasun-ezaugarriak

PCB plaka, kobrezko aurpegi bikoitzarekin, 94-V0 suteen sailkapen-mailarako materiala.

Segurtasun-arauak: UL1012, EN60950, UL60950 betetzen ditu.

Gidatzea eta erradiazioa: Espainiako Konstituzioaren segurtasun-araudiaren araua eta 22. artikulua betetzen ditu.

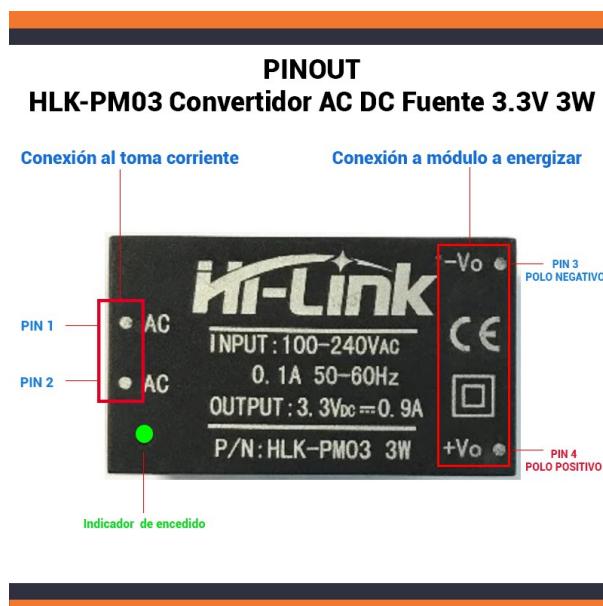
Temperaturaren segurtasunaren diseinua.

Giro-temperaturaren, potentzia horren kondentsadoreek bihurgailu nagusiaren barne-azalera, gehieneko tenperaturakoa, ez da 90 ° C-tik gorakoa.

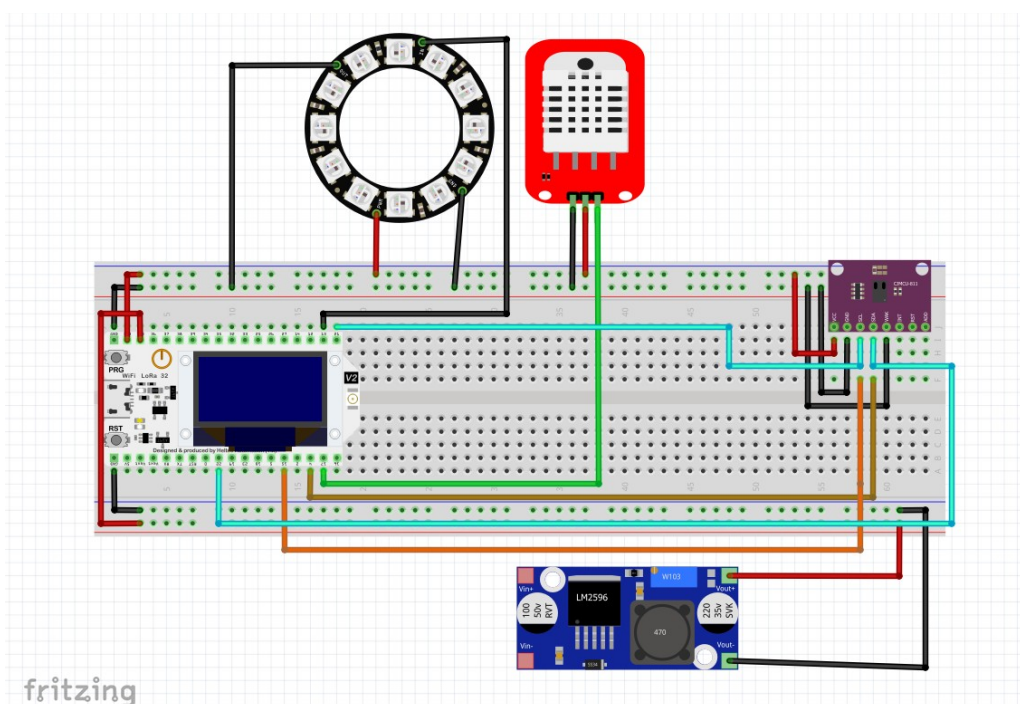
Karkasaren gainazaleko tenperatura maximoa ez da 60 ° C-tik gorakoa.



8.3.Pinout



8.4.Eskema elektrikoa



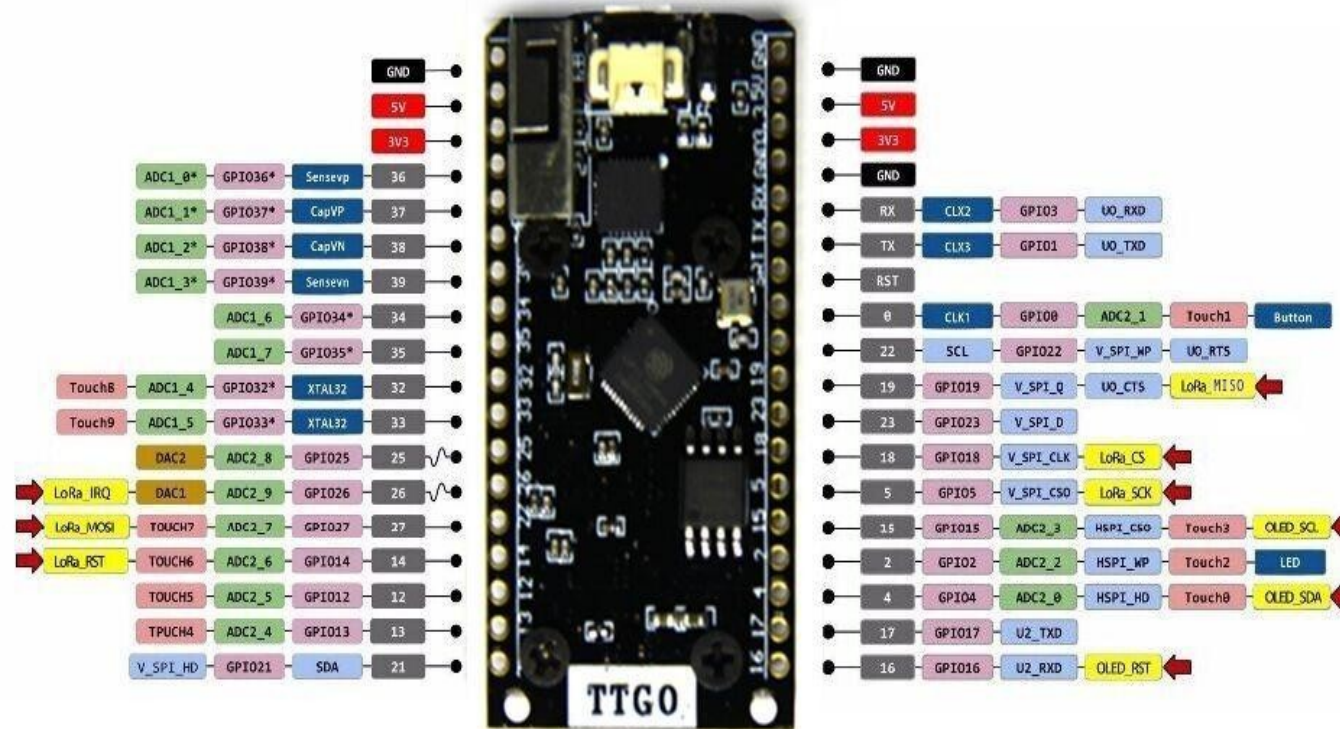
8.5.Estekak

<https://www.amazon.es/Hi-Link-HLK-PM03-Aislado-alimentaci%C3%B3n-Inteligente/dp/B01E15D4ZW>

<https://uelectronics.com/producto/convertidor-ac-dc-fuente-3-3v-tsp-03-reemplazo-hlk-pm03/#:~:text=HLK%2DPM03%20Convertidor%20AC%20DC%20es%20un%20dispositivo%20electr%C3%B3nico%20com%C3%BAnmente,en%20una%20o%20varias%20salidas.>



9. TTGO-ESP32 LORA PINOUT



PLAKAKO ELIKADURA					KANPOKO ELIKADURA		ERABILERA
VIN	5V	3V3	GND	RESET	V	mA	
					3,3	20	TTGO Lora
		X				2,5	DHT11 (Tenperatura eta Hezetasuna)
		X				1	CJMCU-811(CO2_Sentsorea)
		X				480	LED NEOPIXEL
					220 V	900	HLK-PM03 AC-DC 220 V a 3,3V

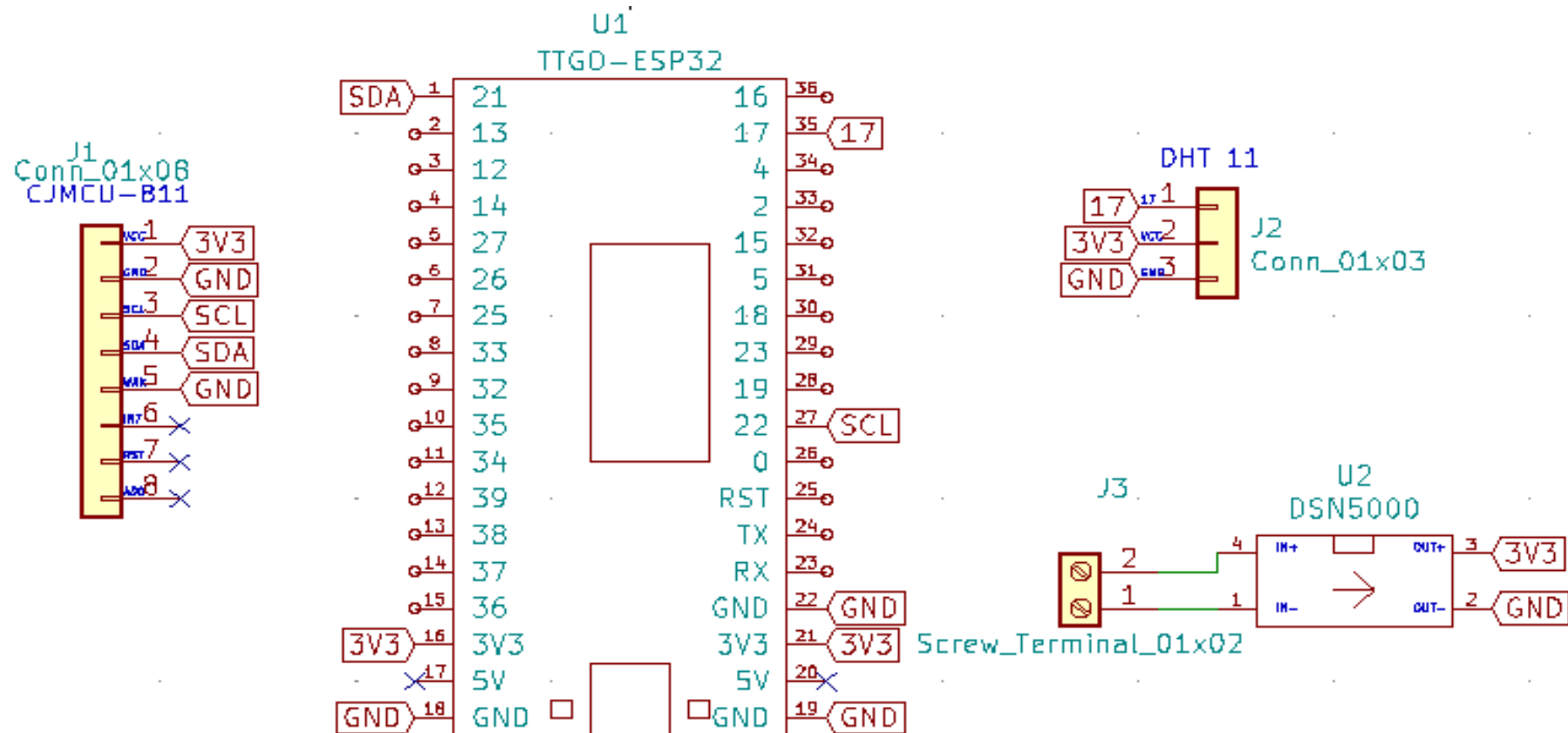
MEMORIAREN ERABILERA			
	FLASH (programa)	RAM (aldagaiak)	EEPROM (konfigurazioa/iraunkorra)
ERABILITAKOA			
HUTSIK			

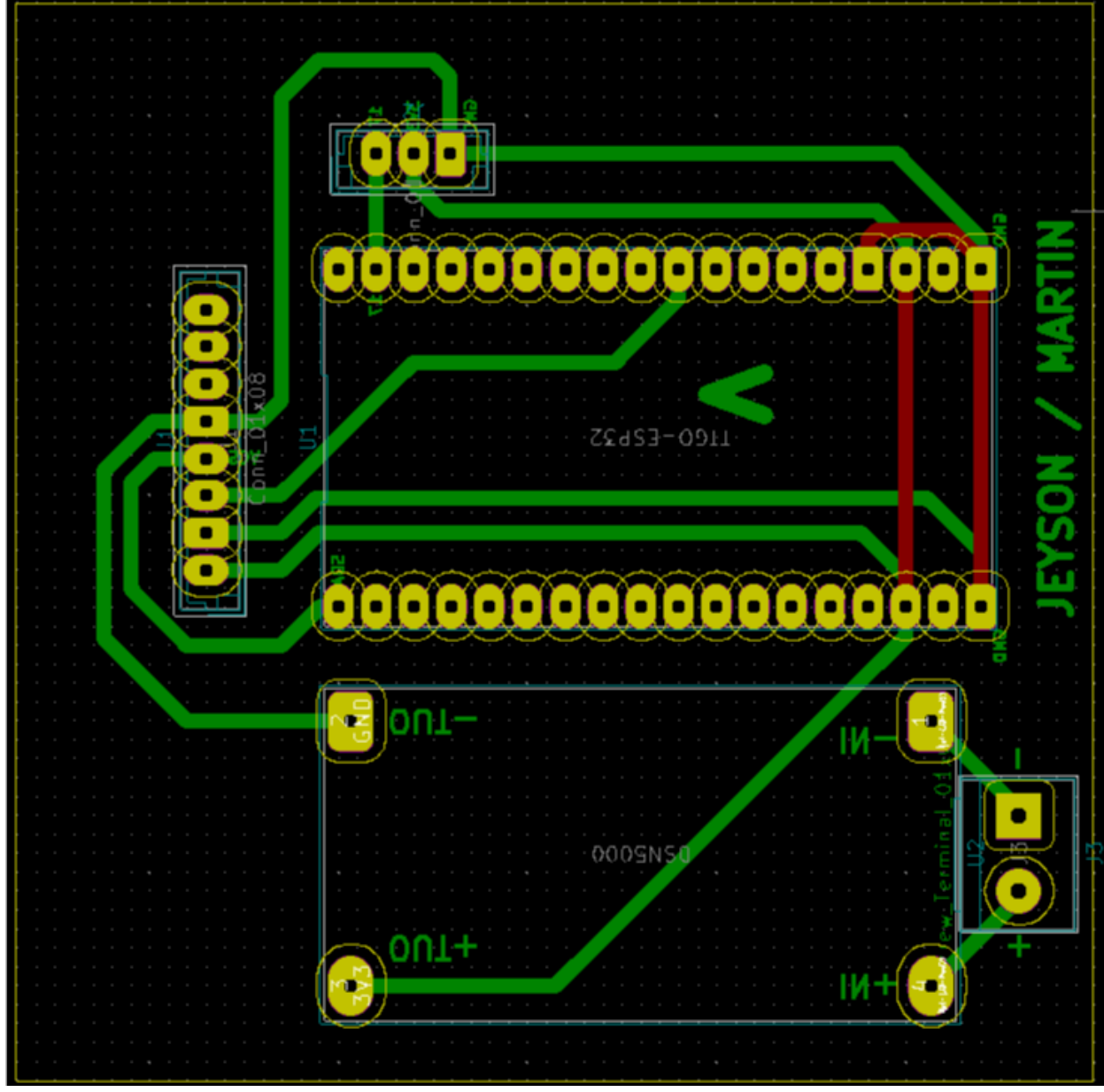
DIGITALA	KOMUNIKAZIOAK	GAINERAKOAK	ERABILERA
----------	---------------	-------------	-----------

36	ADC1_0* / Sensevp		
37	ADC1_1* / CapVP		
38	ADC1_2* / CapVN		
39	ADC1_3* / Sensevn		
34	ADC1_6		
35	ADC1_7		
32	ADC1_4 / XTAL32		
33	ADC1_5 / XTAL32		
25	DAC2 / ADC2_8		
26	DAC1 / ADC2_9 / LoRa_ IRQ		
27	TPUCH7 / ADC2_7 / LoRa_ MOSI		
14	TPUCH6 / ADC2_6 / LoRa_ RST		
12	TPUCH5 / ADC2_5		
13	TPUCH4 / ADC2_4		NeoPixel Komunikazio pina
21	V_SPI_HD / SDA		CCS811 SDA pina
RX	CLX2 / UO_RDX		
TX	CLX3 / UO_TXD		
RST			
0	CLK1 / ADC2_1 / Button		
22	SCL / V_SPI_WP / UP_RTS		CCS811 SCL pina
19	V_SPI_Q / UO_CTS / LoRa_MISO		
23	V_SPI_D		
18	V_SPI_CLK / LoRa CS		
5	V_SPI_CSO / LoRa SCK		
15	HSPI_CSO / OLED_SCL		CCS811 SCL pina
2	HSPI_WP / LED		
DIGITALA	KOMUNIKAZIOAK	GAINERAKOAK	ERABILERA

4	HSPI_HD / OLED_SDA		CCS811 SDA pina
17	U2_TXD		DHT11 komunikazio pina
16	U2_RXD / OLED_RST		

10.Eskema elektrikoak (KiCad)





11. GateWay konfigurazioa

GateWay bat LoRa transmisio modulua duen gailua da eta informazioa Internet eta berarekin komunikatzen diren nodoen artean birbidaltzen du eta alderantziz.

GateWay-a konfiguratzeko tutorial edo manual bat aurkitu dugu interneten.

Hainbat pasu segi ditugu, honako hauek:

- Lehenengo pausua, gure GateWay-a elikatu ondoren Reset botoia 5 segunduz pultsatzea da, goiko aldean duen LEDa azkar parpadeatzen(berde-gorri) jarri arte
- Ondoren Setup botoia 10 segunduz pultsatu mantendu LEDa gorri parpadeatzen jarri harte.
- Hau lortu eta gero GateWay-a orain WiFi AP bat erakusten du, SSID MINIHUB-xxxxxx dena, non xxxxxx GateWay-aren IDA da, 6 digito.
- GateWay-aren pasahitza gailuaren atzeko aldean ikus dezakegu.
- Azkenik, 192.168.4.1 sarrera sartu eta hor Wifia aukeratu dugu



192.168.4.1

MiniHub Setup

Setup network closes in 08:57 Minutes

Configured Networks (1 / 8 max) - Click to remove

MH_CONFIG

—

Scanned Networks (00:55 Minutes ago) - Click to add

+

+

DIRECT-1b-HP M477 LaserJet

+

HP-Print-B6-Officejet Pro 8620

+

VFNL-F49DE8

+

Add Network

Your Network

ADD

CANCEL

SAVE & REBOOT



- Konfigurazioa zuzena bada, Gateway-a berdez kliskatuko du segundo batzuetan, sare horretara konektatzen den bitartean.
- Konfigurazioa zuzena bada, pasabideak BERDE <->GORRIAN kliskatuko du segundo batzuetan, CUPS azken puntura konektatzen den bitartean, eta LNS trafikoko azken puntura konektatzeko beharrezkoa den informazioa lortzen du.
- Konfigurazioa zuzena izan bada, Led berde finkoan egongo da, eta horrek esan nahi du GateWay-a konektatuta dagoela.



12. Programa Nagusia

12.1 Programa nagusia Jeyson eta Martin

```
/*
Programaren izena: ProgramaNagusiaJeysonetaMartinV2.ino
Egilea: Jeyson Rueda eta Martin Malaxetxebarria Data: 2021/02/24
Zeregina: Lora bidezko komunikazioa
Adibide hau domeinu publikokoa da.
Programaren egoera: Egiaztatze/egiaztatuta
Egiaztatutako plakak: UNO ?? Nano Bai MEGA ???
*/
#include "LoraGatewayHeader.h"

void setup() {
  Serial.begin(115200);
  DTH11SentsoreaSetup ();
  OledPantailaSetup ();
  CCS881SentsoreaSetup ();
  TheThingsNetworkSetup ();
  NeoPixelSetup ();
}

void loop() {
  DTH11SentsoreaLoop ();
  OledPantailaLoop ();
  CCS881SentsoreaLoop ();
  TheThingsNetworkLoop ();
  NeoPixelLoop ();
}
```

12.2 CCS881 sentsorea

```
/* Funtzioa: void CCS881SentsoreaSetup (void)
* Zeregina: DTH11 Sentsorea irakurri
* Erabilitako liburutegiak: CCS881 Sensor Library
* Liburutegia nondik hartuta: https://github.com/adafruit/Adafruit\_CCS811 (CCS881
Sensor Library)
* Bueltatzen duena: ezer
* Sartutako parametroak: ezer
* */
void CCS881SentsoreaSetup (void){

  Serial.println("CCS811 test");

  if(!ccs.begin()){
    Serial.println("Failed to start sensor! Please check your wiring.");
    while(1);
  }

  // Wait for the sensor to be ready
  while(!ccs.available());
}

/* Funtzioa: void CCS881SentsoreaLoop (void)
* Zeregina: DTH11 Sentsorea irakurri
* Erabilitako liburutegiak: CCS881 Sensor Library
```



```

* Liburutegia nondik hartuta: https://github.com/adafruit/Adafruit\_CCS811 (CCS881
Sensor Library)
* Bueltatzen duena: ezer
* Sartutako parametroak: ezer
* */

```

```

void CCS881SentsoreaLoop (void){
  if(ccs.available()){
    if(!ccs.readData()){
      //Serial.print(ccs.calculateTemperature());
      Serial.print("CO2: ");
      Serial.print(ccs.getCO2());
      Serial.println("ppm");
      Serial.print("TVOC: ");
      Serial.println(ccs.getTVOC());
    }
    else{
      Serial.println("ERROR!");
      while(1);
    }
  }
  delay(1000);
}

```

12.3.DTH11 sentsorea

```

/* Funtzioa: void DTH11SentsoreaSetup (void)
* Zeregina: DTH11 Sentsorea irakurri
* Erabilitako liburutegiak: DHT Sensor Library
*                               Adafruit Unified Sensor Lib
* Liburutegia nondik hartuta: https://github.com/adafruit/DHT-sensor-library (DHT
Sensor Library)
*                               https://github.com/adafruit/Adafruit\_Sensor
(Adafruit Unified Sensor Lib)
* Bueltatzen duena: ezer
* Sartutako parametroak: ezer
* */

```

```

void DTH11SentsoreaSetup (void){
  Serial.println(F("DHTxx test!"));
  dht.begin();
}

```

```

/* Funtzioa: void DTH11SentsoreaLoop (void)
* Zeregina: DTH11 Sentsorea irakurri
* Erabilitako liburutegiak: DHT Sensor Library
*                               Adafruit Unified Sensor Lib
* Liburutegia nondik hartuta: https://github.com/adafruit/DHT-sensor-library (DHT
Sensor Library)
*                               https://github.com/adafruit/Adafruit\_Sensor
(Adafruit Unified Sensor Lib)
* Bueltatzen duena: ezer
* Sartutako parametroak: ezer
* */

```

```

void DTH11SentsoreaLoop (void){

  // Wait a few seconds between measurements.
  delay(1000);

  // Reading temperature or humidity takes about 250 milliseconds!
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();

```



```

// Read temperature as Celsius (the default)
float t = dht.readTemperature();
// Read temperature as Fahrenheit (isFahrenheit = true)
float f = dht.readTemperature(true);

// Check if any reads failed and exit early (to try again).
if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
}

// Compute heat index in Fahrenheit (the default)
float hif = dht.computeHeatIndex(f, h);
// Compute heat index in Celsius (isFahreheit = false)
float hic = dht.computeHeatIndex(t, h, false);

Serial.print(F("Hezetasuna: "));
Serial.print(h);
Serial.print(F("%  Temperatura: "));
Serial.print(t);
Serial.print(F("°C "));
Serial.print(f);
Serial.print(F("°F  Heat index: "));
Serial.print(hic);
Serial.print(F("°C "));
Serial.print(hif);
Serial.println(F("°F"));
}

```

12.4. Komunikazioa TTN

```

/* Funtzioa: void komunikazioaTTNLoop (void)
 * Zeregina:
 * Erabilitako liburutegiak:
 * Liburutegia nondik hartuta:
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 * */

void komunikazioaTTNLoop (void){
    float h = dht.readHumidity();
    float t = dht.readTemperature();

    unsigned long balioa = ((t*100000000)+(h*10000)+(ccs.geteCO2())); //bidaliko dugun
    zenbakizko txurroaren simulazioa, balioa adibide bat da TTTHHCCCC //txurroa string
    String payload = String(balioa); //karaktereen
    batera pasatu //stringa char
    static uint8_t payloadChar[10]; //stringa char
    arraya sortu //stringa char
    payload.toCharArray((char *)payloadChar, sizeof(payloadChar)); //stringa char
    arraya bihurtu

    // Prepare upstream data transmission at the next possible time. Bidalketa egiteko
    lekua, payloadChar aldagaia bidaltzen du
    LMIC_setTxData2(1, payloadChar, sizeof(payloadChar)-1, 0);
    Serial.println(F("Packet queued"));
}

```



12.5.Led Neopixel

```
/* Funtzioa: void LedNeoPixelSetup void()
 * Zeregina:
 * Erabilitako liburutegiak:
 * Liburutegia nondik hartuta:
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 * */
void NeoPixelSetup (void)
{
    #if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
    clock_prescale_set(clock_div_1);
    #endif
    // END of Trinket-specific code.

    pixels.begin(); // INITIALIZE NeoPixel strip object (REQUIRED)
}

/* Funtzioa: void LedNeoPixelLoop void()
 * Zeregina: DTH11 Sentsorea irakurri
 * Erabilitako liburutegiak:
 * Liburutegia nondik hartuta:
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 * */
void NeoPixelLoop (void)
{
    CCS881SentsoreaLoop ();
    pixels.clear(); // Set all pixel colors to 'off'
    //for(int i=0; i<NUMPIXELS; i++) { // For each pixel...
    // pixels.Color() takes RGB values, from 0,0,0 up to 255,255,255
    // Here we're using a moderately bright green color:
    /*pixels.setPixelColor(0, pixels.Color(125, 255, 0));
    pixels.setPixelColor(1, pixels.Color(255, 0, 0));
    pixels.setPixelColor(2, pixels.Color(0, 255, 0));
    pixels.setPixelColor(3, pixels.Color(0, 0, 255));
    pixels.setPixelColor(4, pixels.Color(255, 125, 0));
    pixels.setPixelColor(5, pixels.Color(0, 255, 125));
    pixels.setPixelColor(6, pixels.Color(125, 0, 255));
    pixels.setPixelColor(7, pixels.Color(255, 0, 125));*/

    if (ccs.getCO2()> 400 && ccs.getCO2()< 750){
        pixels.setPixelColor(0, pixels.Color(0, 255, 0));
        pixels.setPixelColor(1, pixels.Color(0, 255, 0));
        pixels.setPixelColor(2, pixels.Color(0, 255, 0));
    }
    else
    {
        if (ccs.getCO2()>= 750 && ccs.getCO2()< 1200){
            pixels.setPixelColor(0, pixels.Color(0, 255, 0));
            pixels.setPixelColor(1, pixels.Color(0, 255, 0));
            pixels.setPixelColor(2, pixels.Color(0, 255, 0));
            pixels.setPixelColor(3, pixels.Color(255, 255, 0));
            pixels.setPixelColor(4, pixels.Color(255, 255, 0));
            pixels.setPixelColor(5, pixels.Color(255, 255, 0));
        }
        else
        {
            pixels.setPixelColor(0, pixels.Color(0, 255, 0));
            pixels.setPixelColor(1, pixels.Color(0, 255, 0));
        }
    }
}
```



```

    pixels.setPixelColor(2, pixels.Color(0, 255, 0));
    pixels.setPixelColor(3, pixels.Color(255, 255, 0));
    pixels.setPixelColor(4, pixels.Color(255, 255, 0));
    pixels.setPixelColor(5, pixels.Color(255, 255, 0));
    pixels.setPixelColor(6, pixels.Color(255, 0, 0));
    pixels.setPixelColor(7, pixels.Color(255, 0, 0));
  }
}

pixels.show(); // Send the updated pixel colors to the hardware.

delay(100); // Pause before next pass through loop
}

```

12.6.Lora Gateway Header

```

/*-----( LIBURUTEGIAK )-----*/
//DHT11
#include "DHT.h"
// - DHT Sensor Library: https://github.com/adafruit/DHT-sensor-library
// - Adafruit Unified Sensor Lib: https://github.com/adafruit/Adafruit\_Sensor

//CCS811
#include "Adafruit_CCS811.h"
// - CCS811 Sensor Library: https://github.com/adafruit/Adafruit\_CCS811

//The Things Network
#include <lmic.h>
#include <hal/hal.h>
#include <SPI.h>

//Oled Pantaila
#include <LoRa.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

//LED NEOPIXEL
#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
#include <avr/power.h> // Required for 16 MHz Adafruit Trinket
#endif

/*-----( KONSTANTEAK )-----*/
//DHT11
#define DHTPIN 17 // Pin digitala DTH Kkonektatua
#define DHTTYPE DHT11 // DHT 11

//Oled Pantaila
#define SCK 5
#define MISO 19
#define MOSI 27
#define SS 18
#define RST 14
#define DI00 26
//433E6 for Asia
//866E6 for Europe
//915E6 for North America
#define BAND 866E6

//OLED pins

```



```

#define OLED_SDA 4
#define OLED_SCL 15
#define OLED_RST 16
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

//LED NEOPIXEL
#define PIN 13 // On Trinket or Gemma, suggest changing this to 1
#define NUMPIXELS 8 // Popular NeoPixel ring size

//The Things Network
// LoRaWAN NwkSKey, network session key
// This is the default Semtech key, which is used by the early prototype TTN
// network.
static const PROGMEM u1_t NWKSKEY[16] = { 0x4A, 0xB1, 0x46, 0x4A, 0xD2, 0x37, 0x7F,
0xB3, 0x93, 0xA2, 0xC9, 0x48, 0xAA, 0x79, 0xC7, 0x77 };

// LoRaWAN AppSKey, application session key
// This is the default Semtech key, which is used by the early prototype TTN
// network.
static const u1_t PROGMEM APPSKEY[16] = { 0x6D, 0x23, 0xF9, 0xF3, 0xE1, 0xDD, 0xF0,
0x37, 0x59, 0x91, 0xA7, 0xA8, 0xC8, 0xE4, 0x9D, 0x66 };

// LoRaWAN end-device address (DevAddr)
static const u4_t DEVADDR = 0x26011483 ; // <-- Change this address for every node!

const unsigned TX_INTERVAL = 180000;

// Pin mapping
const lmic_pinmap lmic_pins = {
    .nss = 18,
    .rxtx = LMIC_UNUSED_PIN,
    .rst = 14,
    .dio = {26, 33, 32},
};

#define BAUDRATE 115200 //Komunikaziorako datuen abiadura

/*-----( OBJETUAK )-----*/
//DHT11
DHT dht(DHTPIN, DHTTYPE);

//CCS811
Adafruit_CCS811 ccs;

//Oled Pantaila
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RST);

//LED NEOPIXEL
Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);

/*-----( ALDAGIAK )-----*/
//The Things Network
static uint8_t mydata[] = "Hello, world!";
static osjob_t sendjob;

/*-----( FUNTZIOAK )-----*/
//DHT11
void DHT11SentsoreaLoop (void);
void DHT11SentsoreaSetup (void);

//CCS811
void CCS811SentsoreaLoop (void);

```




```

void CCS881SentsoreaSetup (void);

//The Things Network
void os_getArtEui (u1_t* buf) { }
void os_getDevEui (u1_t* buf) { }
void os_getDevKey (u1_t* buf) { }
void TheThingsNetworkSetup (void);
void TheThingsNetworkLoop (void);

//Oled Pantaila
void OledPantailaSetup (void);
void OledPantailaLoop (void);

//LED NEOPIXEL
void NeoPixelLoop (void);
void NeoPixelSetup (void);

//Komunikazioa TTN
void komunikazioaTTNLoop (void)

```

12.8.Oled pantaila

```

/* Funtzioa: void OledPantailaSetup (void)
 * Zeregina:
 * Erabilitako liburutegiak:
 * Liburutegia nondik hartuta:
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 * */
void OledPantailaSetup (void) {
    //reset OLED display via software
    pinMode(OLED_RST, OUTPUT);
    digitalWrite(OLED_RST, LOW);
    delay(20);
    digitalWrite(OLED_RST, HIGH);

    //initialize OLED
    Wire.begin(OLED_SDA, OLED_SCL);
    if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3c, false, false)) { // Address 0x3C for
128x32
        Serial.println(F("SSD1306 allocation failed"));
        for(;;); // Don't proceed, loop forever
    }

    display.clearDisplay();
    display.setTextColor(WHITE);
    display.setTextSize(1);
    display.setCursor(0,0);
    display.print("Kargatzen... ");
    display.display();

    //initialize Serial Monitor
    Serial.begin(115200);

    Serial.println("LoRa Sender Test");

    //SPI LoRa pins
    SPI.begin(SCK, MISO, MOSI, SS);
    //setup LoRa transceiver module
    LoRa.setPins(SS, RST, DIO0);

```



```

if (!LoRa.begin(BAND)) {
    Serial.println("Starting LoRa failed!");
    while (1);
}
Serial.println("LoRa Initializing OK!");
delay(2000);
}

/* Funtzioa: void OledPantailaLoop (void)
 * Zeregina:
 * Erabilitako liburutegiak:
 * Liburutegia nondik hartuta:
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 * */
void OledPantailaLoop (void){
    float h = dht.readHumidity();
    float t = dht.readTemperature();

    if(ccs.available()){
        if(!ccs.readData()){
            //Serial.print(ccs.calculateTemperature());
            Serial.print("CO2: ");
            Serial.print(ccs.geteCO2());
            Serial.println("ppm");
            Serial.print("TVOC: ");
            Serial.println(ccs.getTVOC());
        }
        else{
            Serial.println("ERROR!");
            while(1);
        }
    }

    display.clearDisplay();
    display.setCursor(0,0);
    display.println("Neurketak Jeyson eta Martin");
    display.setCursor(0,10);
    display.print("Tenperatura + hezetasuna");
    display.setCursor(0,20);
    display.print(t);
    display.setCursor(30,20);
    display.print(h);
    display.setCursor(0,40);
    display.print("CO2 + TVCO");
    display.setCursor(0,40);
    display.print(ccs.geteCO2());
    display.setCursor(30,40);
    display.print(ccs.getTVOC());
    display.display();

    delay(1000);
}

```

12.9.The Things Network

```

/* Funtzioa: void onEvent (ev_t ev)
 * Zeregina:
 * Erabilitako liburutegiak:
 * Liburutegia nondik hartuta:
 * Bueltatzen duena: ezer

```



```
* Sartutako parametroak: ezer
* */
```

```
void onEvent (ev_t ev) {
    Serial.print(os_getTime());
    Serial.print(": ");
    switch(ev) {
        case EV_SCAN_TIMEOUT:
            Serial.println(F("EV_SCAN_TIMEOUT"));
            break;
        case EV_BEACON_FOUND:
            Serial.println(F("EV_BEACON_FOUND"));
            break;
        case EV_BEACON_MISSED:
            Serial.println(F("EV_BEACON_MISSED"));
            break;
        case EV_BEACON_TRACKED:
            Serial.println(F("EV_BEACON_TRACKED"));
            break;
        case EV_JOINING:
            Serial.println(F("EV_JOINING"));
            break;
        case EV_JOINED:
            Serial.println(F("EV_JOINED"));
            break;
        case EV_RFU1:
            Serial.println(F("EV_RFU1"));
            break;
        case EV_JOIN_FAILED:
            Serial.println(F("EV_JOIN_FAILED"));
            break;
        case EV_REJOIN_FAILED:
            Serial.println(F("EV_REJOIN_FAILED"));
            break;
        case EV_TXCOMPLETE:
            Serial.println(F("EV_TXCOMPLETE (includes waiting for RX windows)"));
            if (LMIC.txrxFlags & TXRX_ACK)
                Serial.println(F("Received ack"));
            if (LMIC.dataLen) {
                Serial.println(F("Received "));
                Serial.println(LMIC.dataLen);
                Serial.println(F(" bytes of payload"));
            }
            // Schedule next transmission
            os_setTimedCallback(&sendjob, os_getTime()+sec2osticks(TX_INTERVAL),
do_send);
            break;
        case EV_LOST_TSYNC:
            Serial.println(F("EV_LOST_TSYNC"));
            break;
        case EV_RESET:
            Serial.println(F("EV_RESET"));
            break;
        case EV_RXCOMPLETE:
            // data received in ping slot
            Serial.println(F("EV_RXCOMPLETE"));
            break;
        case EV_LINK_DEAD:
            Serial.println(F("EV_LINK_DEAD"));
            break;
        case EV_LINK_ALIVE:
            Serial.println(F("EV_LINK_ALIVE"));
            break;
        default:
    }
```



```

        Serial.println(F("Unknown event"));
        break;
    }
}

/* Funtzioa: void do_send(osjob_t* j)
 * Zeregina:
 * Erabilitako liburutegiak: DHT Sensor Library
 *                               Adafruit Unified Sensor Lib
 * Liburutegia nondik hartuta: https://github.com/adafruit/DHT-sensor-library (DHT
Sensor Library)
 *                               https://github.com/adafruit/Adafruit\_Sensor
(Adafruit Unified Sensor Lib)
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 * */

void do_send(osjob_t* j){
    // Check if there is not a current TX/RX job running
    if (LMIC.opmode & OP_TXRXPEND) {
        Serial.println(F("OP_TXRXPEND, not sending"));
    } else {
        // Prepare upstream data transmission at the next possible time.
        //LMIC_setTxData2(1, mydata, sizeof(mydata)-1, 0);
        komunikazioaTTNLoop ();
        Serial.println(F("Packet queued"));
    }
    // Next TX is scheduled after TX_COMPLETE event.
}

/* Funtzioa: void TheThingsNetworkSetup (void)
 * Zeregina:
 * Erabilitako liburutegiak: DHT Sensor Library
 *                               Adafruit Unified Sensor Lib
 * Liburutegia nondik hartuta: https://github.com/adafruit/DHT-sensor-library (DHT
Sensor Library)
 *                               https://github.com/adafruit/Adafruit\_Sensor
(Adafruit Unified Sensor Lib)
 * Bueltatzen duena: ezer
 * Sartutako parametroak: ezer
 * */

void TheThingsNetworkSetup (void){
    Serial1.println(F("Starting"));

    #ifdef VCC_ENABLE
    // For Pinoccio Scout boards
    pinMode(VCC_ENABLE, OUTPUT);
    digitalWrite(VCC_ENABLE, HIGH);
    delay(180000);
    #endif

    // LMIC init
    os_init();
    // Reset the MAC state. Session and pending data transfers will be discarded.
    LMIC_reset();

    // Set static session parameters. Instead of dynamically establishing a session
    // by joining the network, precomputed session parameters are be provided.
    #ifdef PROGMEM
    // On AVR, these values are stored in flash and only copied to RAM
    // once. Copy them to a temporary buffer here, LMIC_setSession will
    // copy them into a buffer of its own again.
    uint8_t appskey[sizeof(APPSKEY)];
    uint8_t nwkskey[sizeof(NWKSKEY)];

```



```

memcpy_P(appskey, APPSKEY, sizeof(APPSKEY));
memcpy_P(nwkskey, NWKSKEY, sizeof(NWKSKEY));
LMIC_setSession (0x1, DEVADDR, nwkskey, appskey);
#else
// If not running an AVR with PROGMEM, just use the arrays directly
LMIC_setSession (0x1, DEVADDR, NWKSKEY, APPSKEY);
#endif

#if defined(CFG_eu868)
// Set up the channels used by the Things Network, which corresponds
// to the defaults of most gateways. Without this, only three base
// channels from the LoRaWAN specification are used, which certainly
// works, so it is good for debugging, but can overload those
// frequencies, so be sure to configure the full frequency range of
// your network here (unless your network autoconfigures them).
// Setting up channels should happen after LMIC_setSession, as that
// configures the minimal channel set.
// NA-US channels 0-71 are configured automatically
LMIC_setupChannel(0, 868100000, DR_RANGE_MAP(DR_SF12, DR_SF7), BAND_CENTI);
// g-band
LMIC_setupChannel(1, 868300000, DR_RANGE_MAP(DR_SF12, DR_SF7B), BAND_CENTI);
// g-band
LMIC_setupChannel(2, 868500000, DR_RANGE_MAP(DR_SF12, DR_SF7), BAND_CENTI);
// g-band
LMIC_setupChannel(3, 867100000, DR_RANGE_MAP(DR_SF12, DR_SF7), BAND_CENTI);
// g-band
LMIC_setupChannel(4, 867300000, DR_RANGE_MAP(DR_SF12, DR_SF7), BAND_CENTI);
// g-band
LMIC_setupChannel(5, 867500000, DR_RANGE_MAP(DR_SF12, DR_SF7), BAND_CENTI);
// g-band
LMIC_setupChannel(6, 867700000, DR_RANGE_MAP(DR_SF12, DR_SF7), BAND_CENTI);
// g-band
LMIC_setupChannel(7, 867900000, DR_RANGE_MAP(DR_SF12, DR_SF7), BAND_CENTI);
// g-band
LMIC_setupChannel(8, 868800000, DR_RANGE_MAP(DR_FSK, DR_FSK), BAND_MILLI);
// g2-band
// TTN defines an additional channel at 869.525Mhz using SF9 for class B
// devices' ping slots. LMIC does not have an easy way to define set this
// frequency and support for class B is spotty and untested, so this
// frequency is not configured here.
#elif defined(CFG_us915)
// NA-US channels 0-71 are configured automatically
// but only one group of 8 should (a subband) should be active
// TTN recommends the second sub band, 1 in a zero based count.
// https://github.com/TheThingsNetwork/gateway-conf/blob/master/US-global.conf.json
LMIC_selectSubBand(1);
#endif

// Disable link check validation
LMIC_setLinkCheckMode(0);

// TTN uses SF9 for its RX2 window.
LMIC.dn2Dr = DR_SF7;

// Set data rate and transmit power for uplink (note: txpow seems to be ignored by
the library)
LMIC_setDrTxpow(DR_SF7,14);

// Start job
do_send(&sendjob);
}

/* Funtzioa: void TheThingsNetworkLoop (void)
 * Zeregina:

```



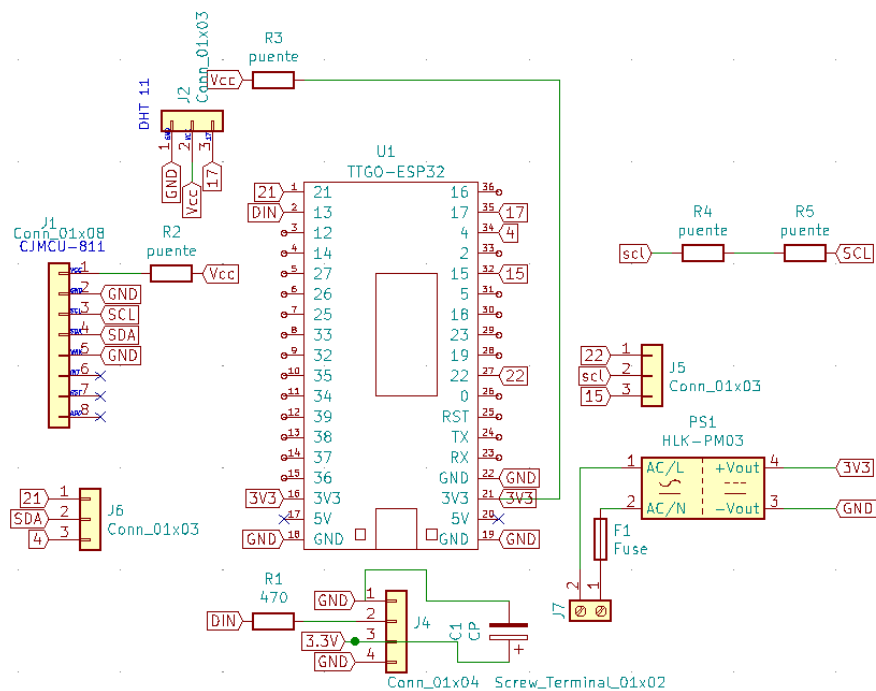
```

* Erabilitako liburutegiak: DHT Sensor Library
*                               Adafruit Unified Sensor Lib
* Liburutegia nondik hartuta: https://github.com/adafruit/DHT-sensor-library (DHT
Sensor Library)
*                               https://github.com/adafruit/Adafruit\_Sensor
(Adafruit Unified Sensor Lib)
* Bueltatzen duena: ezer
* Sartutako parametroak: ezer
* */
void TheThingsNetworkLoop (void){
  os_runloop_once();
}

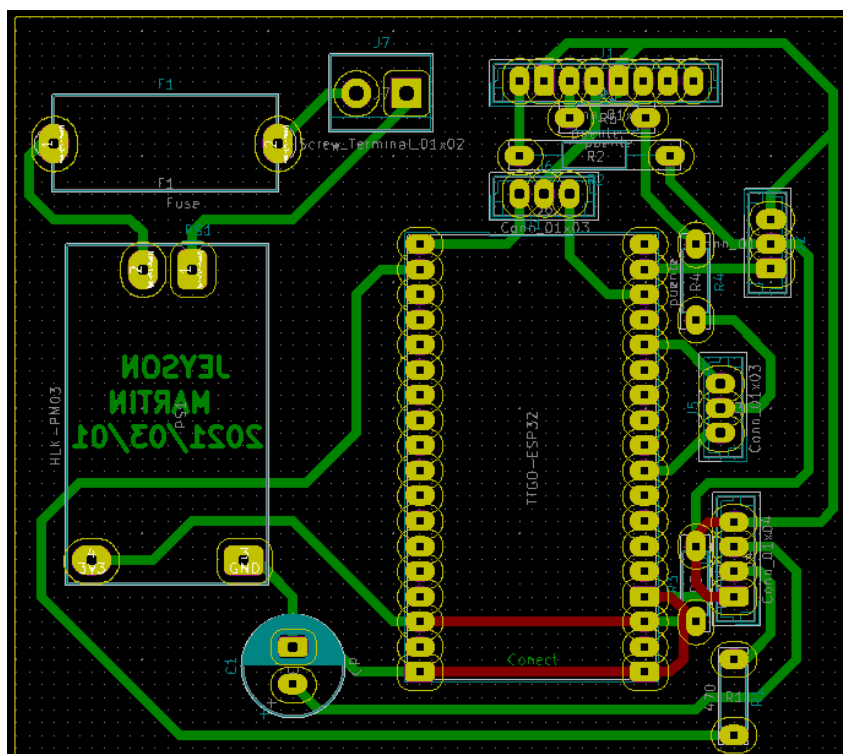
```



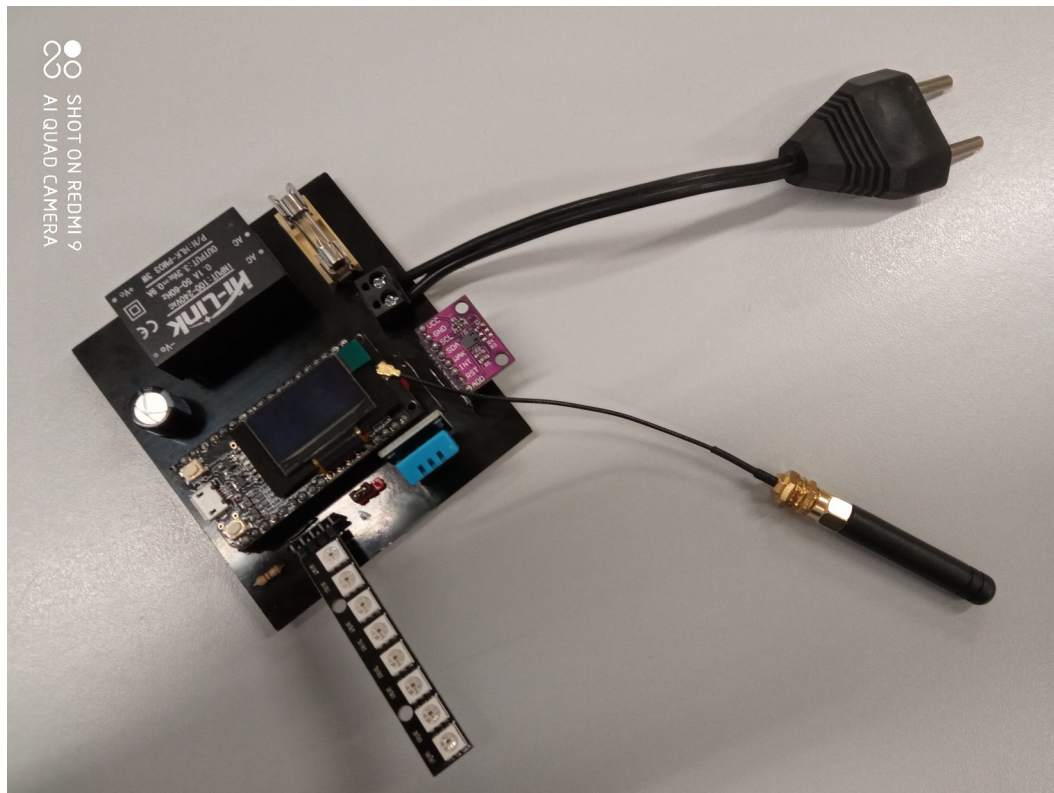
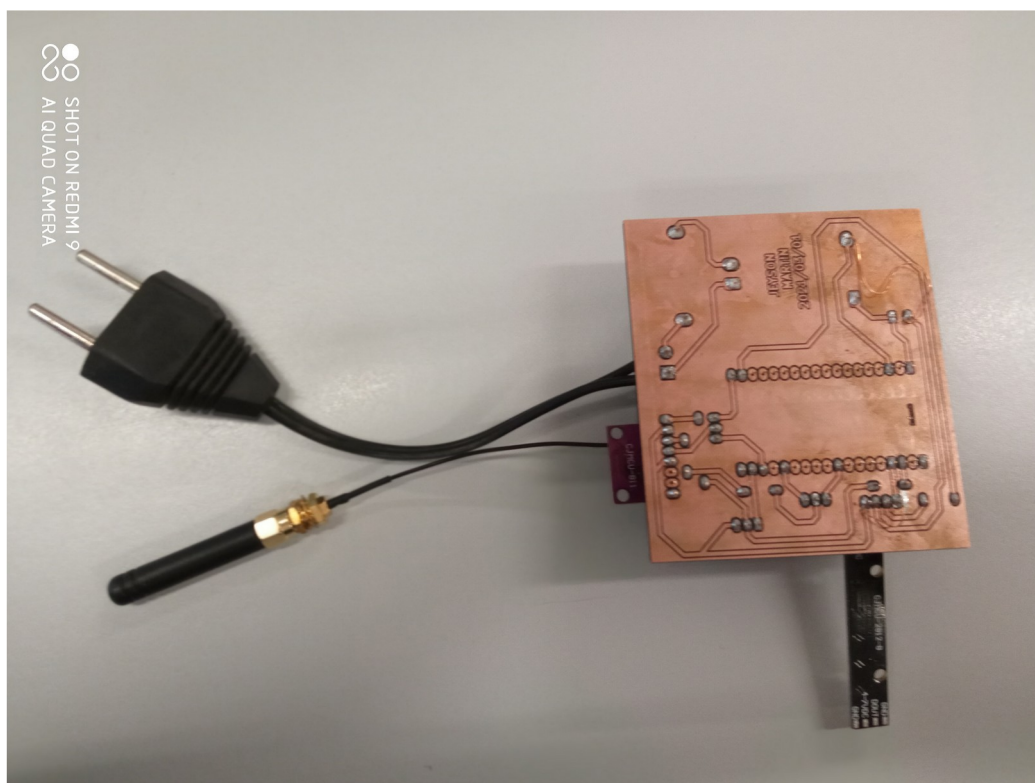
13.1.Kicad sch.



13.2.KiCad pcb.



14.Sistemaren Argazkiak



15.Gorabeherak

Oled pantaila ez zihoan, DHT 11 sentsorea aldatu izan behar dugu eta zihoalako. Horretaik aparte, lora TTGO kontrolagailuaren Oled pantaila ez zihoan. Plaka probatzerakoan CO2 sentsorea ez zihoan eta arazoak eukin ditugu.



16.Hobetzeko proposamenak

Ez.

