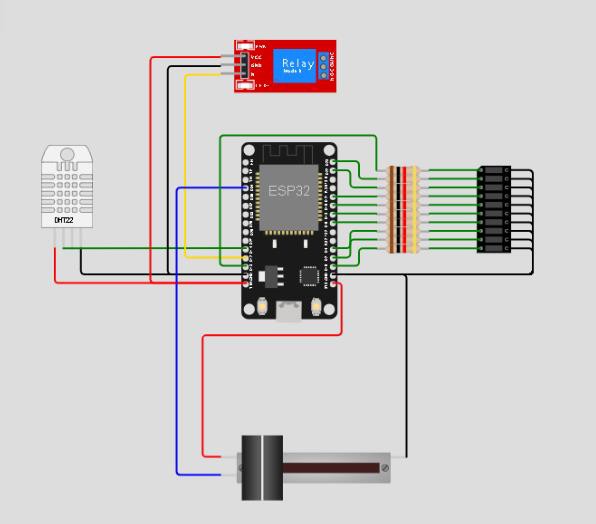


Evidencia 11 - Sistema completo

# \*\* Fecha de entrega (03/11/23) \*\*

Probar la comunicación completa entre el dispositivo IoT y la PC.

Realizar las correcciones necesarias y la optimización del sistema que consideres.  
Proporcionar el código fuente y documentación de los programas que se han escrito.

Documentar las pruebas realizadas, las correcciones y cambios realizados a lo largo del proyecto.

**Código del proyecto**

/\*====================================================================

; "Sensor de temperatura con nivel de batería"

;

; Creado:   lunes 9 de octubre del 2023

; Procesador: ESP32

; Compilador:  Wokwi

; Autor: Ulises Ale

; Descripción:

; El dispositivo está diseñando para dar la información producida

; por el sensor DHT22, el nivel de Batería a través del led gráfico

; y el accionar de un Actuador.

; Esquema eléctrico con los respectivos componentes al microcontrolador

;

;====================================================================\*/

#include <Adafruit\_Sensor.h>

#include <DHT.h>

#include <DHT\_U.h>

// Pin para el relé

const int relayPin = 12;

// Pin para el potenciómetro

const int potPin = 34;

// Pins para la barra de LED del nivel

const int ledPins[] = {15, 2, 4, 5, 18, 19, 21, 22, 23, 13};

// Configuración del sensor DHT22

#define DHTPIN 14

#define DHTTYPE DHT22

DHT\_Unified dht(DHTPIN, DHTTYPE);

void setup() {

  pinMode(relayPin, OUTPUT);

  dht.begin();

**Serial**.begin(9600);

  // Configurar los pines de la barra de LED del nivel como salidas

  for (int i = 0; i < sizeof(ledPins) / sizeof(ledPins[0]); i++) {

    pinMode(ledPins[i], OUTPUT);

  }

}

void loop() {

  sensors\_event\_t event;

  dht.temperature().getEvent(&event);

  float temperatura = event.temperature;

  dht.humidity().getEvent(&event);

  float humedad = event.relative\_humidity;

  int valorPotenciometro = analogRead(potPin);

  int porcentajePotenciometro = map(valorPotenciometro, 0, 4095, 0, 100); // El ESP32 tiene un ADC de 12 bits

**Serial**.print("Temperatura: ");

**Serial**.print(temperatura);

**Serial**.print(" °C | Humedad: ");

**Serial**.print(humedad);

**Serial**.print(" % | Potenciómetro: ");

**Serial**.print(porcentajePotenciometro);

**Serial**.println(" %");

  if (porcentajePotenciometro >= 50) { // Si el potenciómetro supera o

    digitalWrite(relayPin, HIGH); // iguala al 50% enciende LED

  } else {

    digitalWrite(relayPin, LOW); // Sino permanece apagado

  }

  // Control de la barra de LED del nivel

int ledsEncendidos = map(porcentajePotenciometro, 0, 100, 0, sizeof(ledPins) / sizeof(ledPins[0]));

  for (int i = 0; i < sizeof(ledPins) / sizeof(ledPins[0]); i++) {

    if (i < ledsEncendidos) {

      digitalWrite(ledPins[i], HIGH);

    } else {

      digitalWrite(ledPins[i], LOW);

    }

  }

  delay(1000);

}

**En este caso, se incorpora la luz LED para mostrar que cuando tiene mas del 50% de batería la luz se enciende.**

# Enlace al simulador [Wokwi]

https://wokwi.com/projects/378165306553327617