



**Instituto Tecnológico y de Estudios Superiores
de Monterrey**

“Campus Monterrey”

**TC1004B.506 – Implementación de internet de las
cosas (Gpo 506)**

Mtro. Rafael Emilio Dávalos Villarreal

Mtro. Sergio Ramírez San Vicente

“Documentación de reto”

- Conexión de caja de seguridad -

Marco Antonio Rodríguez Amezcua A00834672

Omar Damián Martínez Cordero A00832849

Santiago Quihui Rubio A00832880

Reynaldo Hernández González A00829814

ANTECEDENTES DE IOT

La Internet de las cosas es un tema emergente de importancia técnica, social y económica. En este momento se están combinando productos de consumo, bienes duraderos, automóviles y camiones, componentes industriales y de servicio públicos, sensores y otros objetos de uso cotidiano con conectividad a Internet y potentes capacidades de análisis de datos que prometen transformar el modo en que trabajamos, vivimos y jugamos. Las proyecciones del impacto de la IoT sobre Internet y la economía son impresionantes: hay quienes anticipan que en el año 2025 habrá hasta cien mil millones de dispositivos conectados a la IoT y que su impacto será de US \$11.000.000.000.000. Sin embargo, la Internet de las Cosas también plantea importantes desafíos que podrían dificultar la realización de sus potenciales beneficios. Noticias sobre ataques a dispositivos conectados a Internet, el temor a la vigilancia y las preocupaciones relacionadas con la privacidad ya han captado la atención del público. Los desafíos técnicos siguen allí, pero además están surgiendo nuevos desafíos.

El término “Internet de las Cosas” (IoT) fue empleado por primera vez en 1999 por el pionero británico Kevin Ashton para describir un sistema en el cual los objetos del mundo físico se podían conectar a Internet por medio de sensores. Ashton acuñó este término para ilustrar el poder de conectar a Internet las etiquetas de identificación por radiofrecuencia (RFID) que se utilizaban en las cadenas de suministro corporativas para contar y realizar un seguimiento de las mercancías sin necesidad de intervención humana. Hoy en día, el término Internet de las Cosas se ha popularizado para describir escenarios en los que la conectividad a Internet y la capacidad de cómputo se extienden a una variedad de objetos, dispositivos, sensores y artículos de uso diario.

Aunque el término “Internet de las Cosas” es relativamente nuevo, el concepto de combinar computadoras y redes para monitorear y controlar diferentes dispositivos ha existido durante décadas. Por ejemplo, a fines de la década de 1970 ya había en el mercado sistemas disponibles para monitorear los medidores conectados a la red eléctrica de forma remota a través de las líneas telefónicas. En la década de 1990, los avances en la tecnología inalámbrica permitieron la difusión de soluciones corporativas e industriales “máquina a máquina” (M2M) para monitorear y operar diferentes equipos. Sin embargo, muchas de estas primeras soluciones M2M se basaban en redes dedicadas especialmente construidas para este propósito y en estándares propietarios o específicos de la industria, no en redes basadas en el Protocolo de Internet (IP) y los estándares de Internet.

INTRODUCCIÓN

En este documento se plantea describir el proceso de desarrollo del proyecto de internet de las cosas del equipo 3, el cual se basa en una caja de seguridad, esta caja de seguridad busca la implementación de un sensor de fuerza y un sensor de movimiento, en el que el sensor de movimiento se encuentra conectado a un led que al momento de la detección de un mínimo movimiento el led que se encuentra en la protoboard conectado al nodeMCU se enciende y al no detectar movimiento se mantiene el led apagado, así también el sensor de fuerza tiene la finalidad de detectar y medir el momento en el que un objeto que se encuentre sobre el sensor de fuerza sea quitado, al quitar el objeto que se encuentra sobre el sensor de fuerza se activa una luz led que tiene la finalidad de alertar que se ha quitado un objeto. Esta propuesta se ideó gracias a una lluvia de ideas por parte del equipo, buscábamos un proyecto que tenga una funcionalidad cotidiana y que pudiera ayudar a las personas, así que una de las primeras ideas que se planteó fue la implementación de un sensor de temperatura y la implementación de un ventilador que se activa al momento en el cual el sensor de temperatura detecta un aumento en la temperatura del lugar y automáticamente el ventilador se activará, pero se optó por la creación de la caja de seguridad debido a la innovación que presenta el hacer uso de la unión del sensor de movimiento y el sensor de fuerza y a partir de ello poder lograr una conexión que se logra a través de herramientas de comunicación de datos que tienen la finalidad de mandar los datos que recibieron a una base de datos que nos ayudará a almacenar los datos de salida medidos por los sensores y que gracias a la base de datos asimismo nos permite visualizar aquellos datos recibidos y facilita el análisis de los datos.

DESCRIPCIÓN DE CADA ELEMENTO DEL PROYECTO

- **Microcontrolador nodeMCU:**

Es un microcontrolador que tiene un costo bajo. Este se encuentra inspirado en arduino por lo que el lenguaje de programación que utilizan sus diferentes ID's serán compatibles con este microcontrolador. Usa exclusivamente un módulo ESP8266 para realizar su conexión a internet de manera inalámbrica. Primeramente, podríamos considerar un nodeMCU como una plataforma de desarrollo de código abierto ya que es posible su utilización para la ejecución de distintas aplicaciones en relación al manejo de distintos sensores. Así mismo, cuenta con una API de red estilo NodeJS, por lo que es bastante flexible en su forma de uso y las herramientas y características que proporciona. Ahora bien, enfocándonos al uso y aplicación que le podemos dar a un nodeMCU es posible hacer proyectos de cualquier propósito general, de manera local (sin conexión wifi), control y automatización, y proyectos que tengan relación con el internet de las cosas. Esto último es lo más utilizado por parte de este microcontrolador por su capacidad de manejo.

- **Sensores:**

- **Fotoresistor:**

Se desempeña como una resistencia eléctrica la cual cambia su valor o medición en relación a la cantidad de luminosidad que recibe sobre su superficie. Así mismo, cuanto mayor llegue a ser la cantidad de luz que reciba el fotoresistor menor será su resistencia y viceversa. Las unidades que utilizan este sensor son los Ohms (Ω), los cuales pueden llegar a ser desde cero hasta 100 Ohms.

- **Sensor de temperatura y humedad DHT11:**

Es un sensor que convierte medidas atmosféricas en relación al ambiente, como lo es la temperatura de un espacio cerrado, este sensor hace uso de un sensor capacitivo de humedad y cuenta con un termistor para la medición de las condiciones ambientales, la salida de datos la muestra mediante una señal digital en el pin de datos y no posee una salida analógica.

- **Sensor de distancia ultrasónico HC-SR04:**

Es un sensor que mide cierta distancia utilizando ultrasonido para así determinar la distancia o el tamaño del intervalo entre el sensor y un objeto. El límite que maneja es en un rango entre 2 y 450 cm. Maneja un bajo consumo de energía y cuenta con una excelente lectura y precisión para la toma de sus mediciones. La salida de sus datos se muestra como el rango de distancia que detecta en centímetros hasta que cierto objeto obstruya su lectura ultrasónica.

- **Sensor de movimiento PIR:**

Es un detector pasivo infrarrojo que detecta las variaciones de la radiación infrarroja dentro del área de cobertura por el sensor. Suele ser bastante útil para detectar la presencia de personas o animales a través del calor que emiten sus cuerpos. Cabe destacar que no generan de forma activa ninguna señal y solo pueden recibir radiaciones para su correcto funcionamiento, el cual radica en comparar la temperatura que desprende un objeto con la de su alrededor, de manera que detecta un lugar en específico con este tipo de lecturas.

- **Sensor de peso y fuerza FSR:**

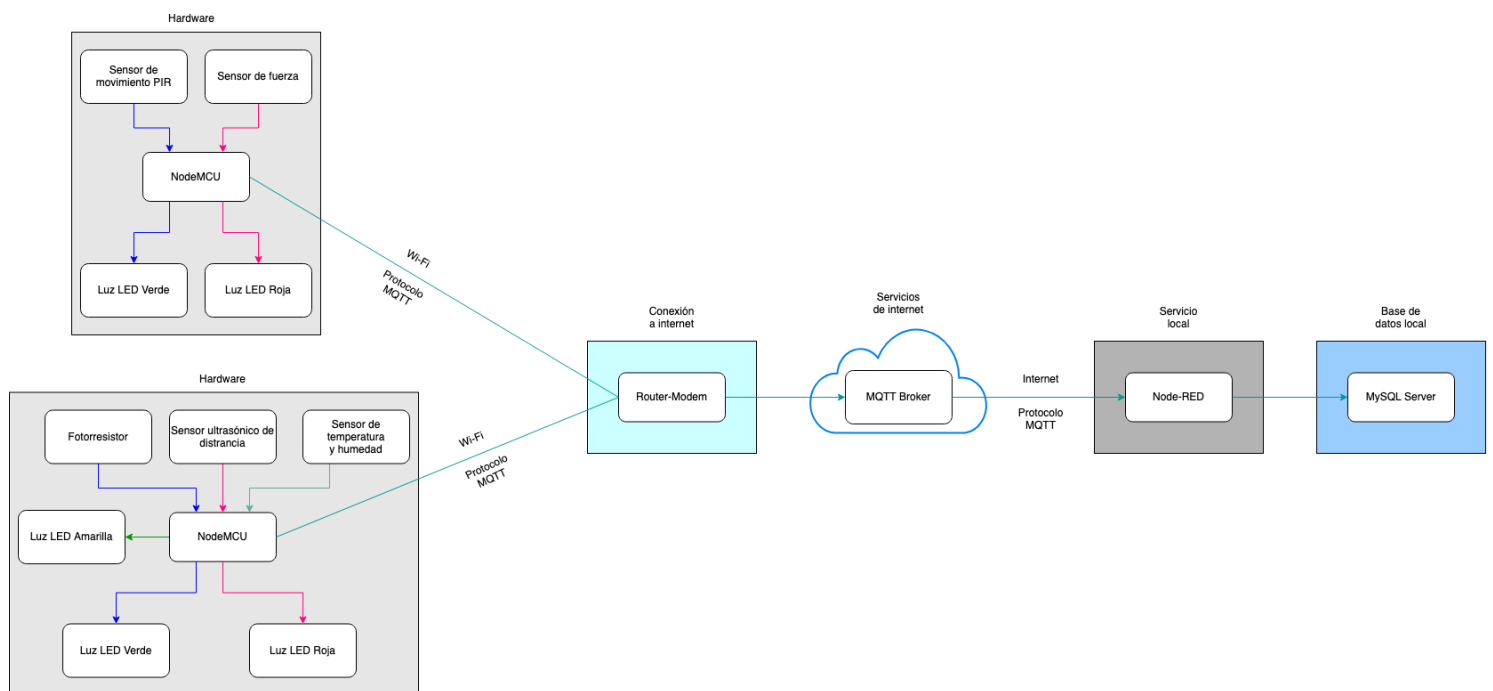
Es un sensor que varía sus lecturas en relación a la resistencia que puede cambiar conforme la presión o fuerza aplicada en el área sensitiva. En relación a su estructura, lleva consigo dos pines que hacen que este sensor sea posible de instalar en placas de pruebas.

- **Actuadores:**

En el caso de nuestro proyecto no se instalaron ningún tipo de actuadores, ya que todas nuestras salidas de los sensores fueron simuladas con leds como prueba de que funcionaban y marcaban una cierta instrucción en su lectura, dando como respuesta un valor booleano. Tomando en cuenta los sensores sería de la siguiente manera:

- Fotoresistor: se prendía el led cuando no había suficiente luz, y si detectaba luz se apagaba.
 - Sensor de temperatura y humedad DHT11: se prendía el led cuando detectaba una temperatura mayor a 26°.
 - Sensor de distancia ultrasónica HC-SR04: no tenía ni actuador ni led de respuesta, sólo envía las lecturas registradas.
 - Sensor de movimiento PIR: se prendía el led cuando detectaba el movimiento de algo dentro de su zona de cobertura.
 - Sensor de peso y fuerza FSR: se prendía el led cuando se supera la marca de 800 Ohmios (sus unidades son en relación a la resistencia que cambia su valor resistivo, en función a la presión ejercida).
- **Broker MQTT:**
Es una especie de servidor con el cual es posible que los clientes puedan comunicarse entre sí a través de internet con los diversos dispositivos vinculados y conectados a una red WIFI. A través de este broker es posible recibir toda la cantidad de información disponible por parte de otro usuario u otro dispositivo, como lo puede ser un microcontrolador. Su protocolo es controlado por eventos, donde no existe una transmisión de datos periódica o continua.
 - **NodeRED:**
Es una herramienta de desarrollo pública (open-source) que tiene como base principal una forma de programación muy bien ilustrada. Tienen como propósito establecer una conexión entre dispositivos de hardware, API's y servicios en línea. Es una aplicación de gestión y transformación de datos en tiempo real para soluciones que tengan relación con IoT. Se basa principalmente en la conexión de nodos para la ejecución de tareas específicas, generando una combinación de nodos de entrada, nodos de procesamiento y nodos de salida (flow).
 - **Base de Datos**
Primeramente, nosotros manejamos el ambiente de MySQL. Tomando esto en cuenta, ambiente es un sistema de gestión de base de datos relacional que está basado en código abierto. Su principal función es almacenar la información que le es entregada desde su redireccionamiento a otras páginas como broker's. De igual manera, es posible organizar la información en tablas de tal manera que sea más fácil la interpretación de los datos en la misma base de datos que se irá actualizando constantemente.

DISEÑO DE LOS ELEMENTOS INTERCONECTADOS



Se cuenta con dos microcontroladores NodeMCU ESP-12e, cada cumpliendo funciones diferentes. Nuestro primer NodeMCU, y el principal del proyecto, está conectado a dos sensores y dos actuadores: sensor de movimiento infrarrojo (PIR), sensor de fuerza, luz LED verde y luz LED roja. Estos 5 elementos están conectados con el apoyo de dos placas de prueba (protoboard) y varios cables jumper macho-macho y macho-hembra, al igual que 3 resistores para controlar la corriente que le damos a las luces LED y al sensor de fuerza. Con el apoyo de la conexión Wi-Fi que nos permite realizar el NodeMCU, los datos recopilados se envían por internet a un broker de MQTT proporcionado por la empresa HiveMQ.

Estos datos enviados al broker se reciben a través de suscripciones realizadas en la herramienta Node-RED, la cual se ejecuta en una computadora que funciona como nuestro propio servidor.

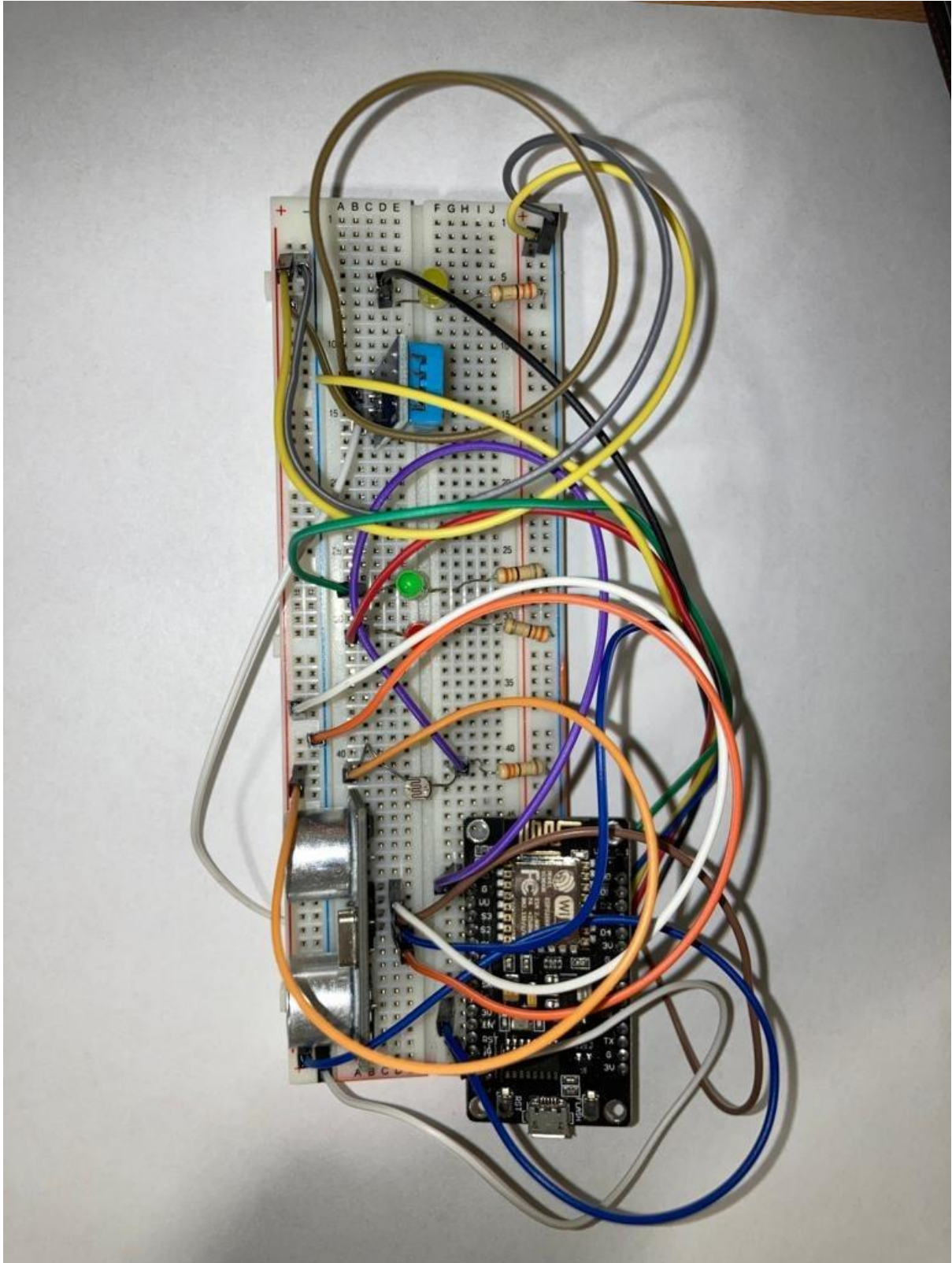
Esta computadora, aparte de ejecutar Node-RED, tiene nuestra base de datos relacional en un servidor MySQL. Esta base de datos es el destino final de todo lo que nuestro proyecto recopila. Para recibir los datos, se utiliza la misma herramienta mencionada anteriormente para procesar la información recibida por MQTT mediante funciones de JavaScript y enviarlas al servidor MySQL.

Este mismo proceso se realiza con nuestro segundo microcontrolador, excepto que este cuenta con otros sensores. Este prototipo tiene 3 luces LED, cada una con su resistor, un sensor ultrasónico de distancia, un sensor de temperatura/distancia y un fotorresistor, el cual también tiene un resistor de por medio al recibir la corriente.

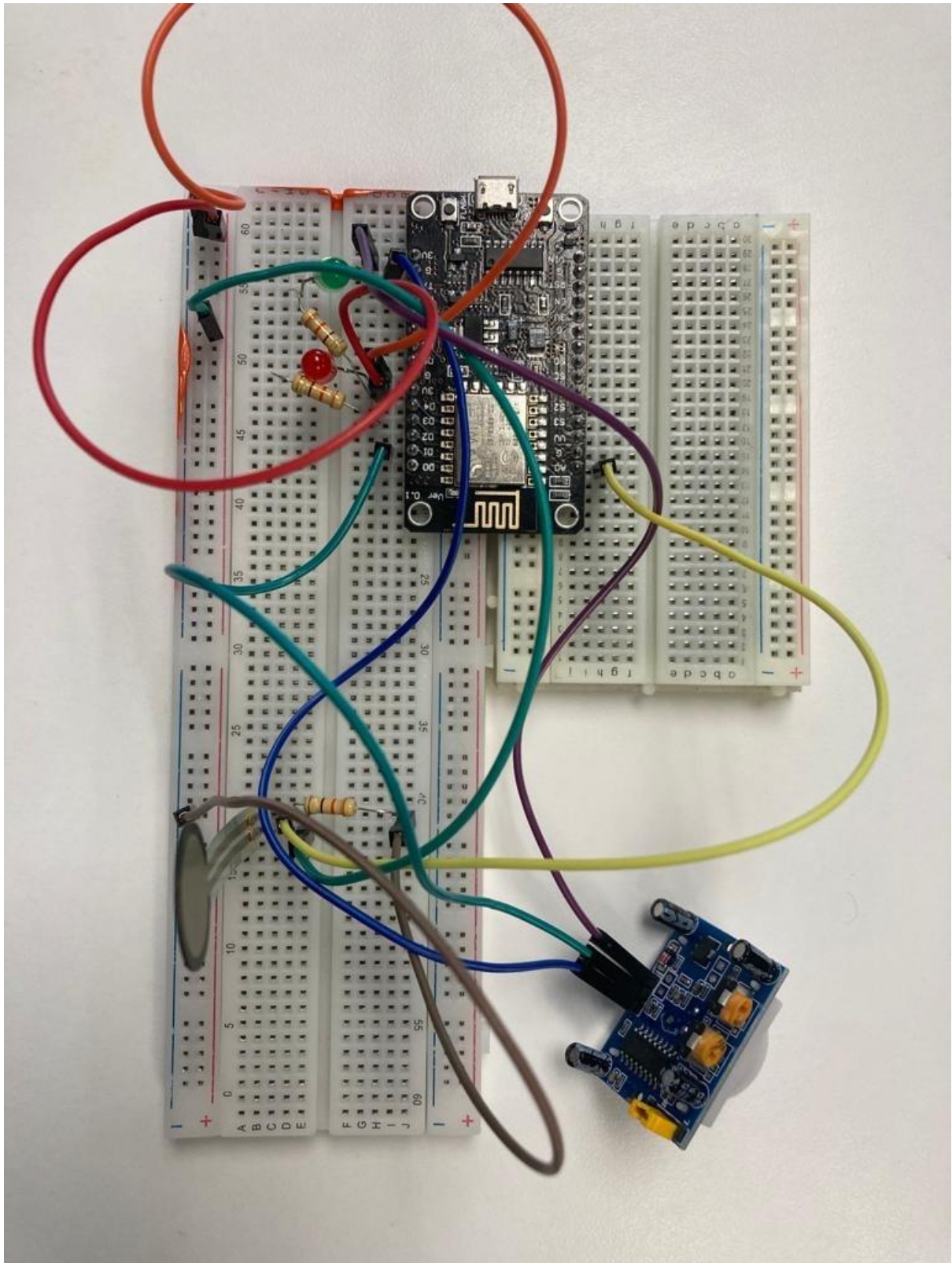
DESARROLLO DEL PROYECTO

La elaboración del proyecto se hizo en conjunto con las actividades del módulo del reto. La primera actividad consistió en conectar y programar el funcionamiento del sensor fotoresistor, el sensor de temperatura y humedad DHT11, y el sensor de distancia ultrasónico HC-SR04. Se realizaron las conexiones según las indicaciones de la actividad, y la mayoría del código lo proporcionó la actividad. Después trabajamos en la siguiente actividad, la cual fue establecer la conexión al broker MQTT para poder mandar las lecturas de los sensores hacia un tópico en el broker. El código para esta funcionalidad también fue proporcionado por la actividad. La siguiente actividad era descargar y conectar Node Red al MQTT para poder extraer los datos de los tópicos. Para esto se siguieron las indicaciones en la actividad para su cumplimiento. En la siguiente actividad aprendimos de manera breve a utilizar las funciones básicas de MySQL para generar y utilizar una base de datos. Finalmente hicimos la conexión de Node Red hacia la base de datos. Compramos el sensor de peso y fuerza FSR y lo utilizamos junto con el sensor de movimiento PIR para conectarlos a otro NodeMCU en otro protoboard. Con esto, todos los elementos principales del proyecto estaban listos, y el proyecto finalizado. Las lecturas de los sensores se mandaron al MQTT, y esos datos se extrajeron con Node Red, donde después se mandaron a la base de datos. A continuación se muestran los detalles de todas estas etapas del desarrollo del proyecto.

Imágenes de los circuitos

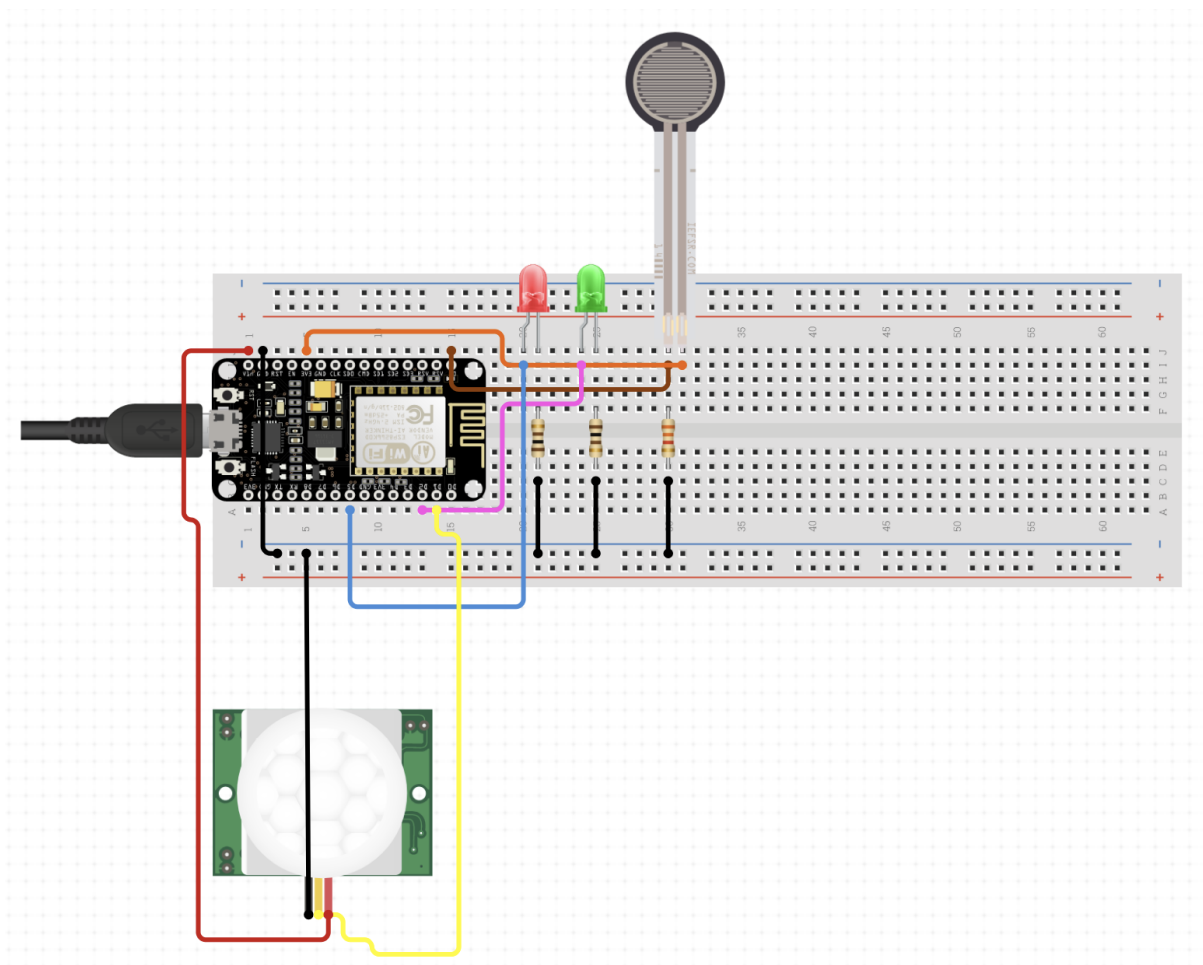
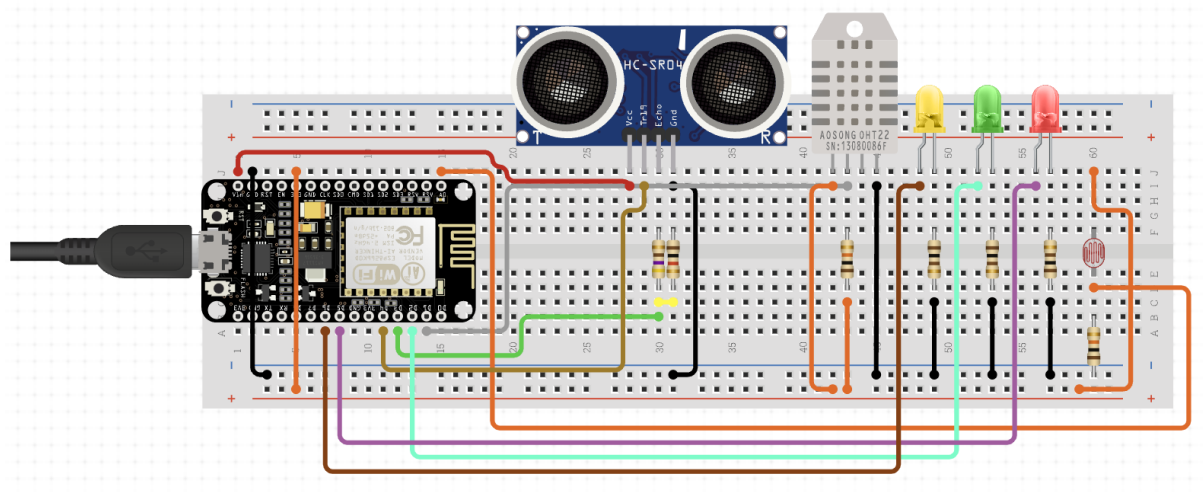


NodeMCU conectado a sensores de luminosidad, de temperatura y humedad, y de distancia



NodeMCU conectado a sensores de peso y de movimiento

Diagrama esquemático del proyecto



Programa desarrollado medición temperatura, distancia e iluminación

```

// Programa para NodeMCU
// Programa detector de temperatura
// Programa detector de iluminacion mediante un fotorresistor
// Programa detector de distancia

// ----- Definiciones de pins de NodeMCU (se cuenta de arriba hacia abajo)
#define D0 16 // Arriba derecha 1
#define D1 5 // Arriba derecha 2
#define D2 4 // Arriba derecha 3
#define D3 0 // Arriba derecha 4
#define D4 2 // Arriba derecha 5
// Posiciones 6 3V3, 7 GND
#define D5 14 // Arriba derecha 8
#define D6 12 // Arriba derecha 9
#define D7 13 // Arriba derecha 10
#define D8 15 // Arriba derecha 11
#define SD3 10 // Arriba izquierda 4
#define SD2 9 // Arriba izquierda 5
// el RX = GPIO3 y TX = GPIO1
// LED_BUILTIN = 16 //GPIO16

// ----- Definiciones del sensor de temperatura DHT11
#include "DHT.h"
#define DHTPIN 2 // Posicion equivalente a D4
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

// ----- Definiciones de fotorresistor
int photoPin = A0;
int ledphotoResistor = 16; // Equivale a D0
int x = -1; //0..1023

// ----- Definiciones de Sensor distancia
int trigPin = D2; //Conectado a D2, int trigPin =
int echoPin = D3; //Conectado a D1, int echoPin =
float v = 331.5+0.6*20; // m/s

// ----- Definiciones de ESP8266MQTT
/*
Basic ESP8266 MQTT example
This sketch demonstrates the capabilities of the pubsub library in combination
with the ESP8266 board/Library.
It connects to an MQTT server then:
- publishes "hello world" to the topic "outTopic" every two seconds
- subscribes to the topic "inTopic", printing out any messages
it receives. NB - it assumes the received payloads are strings not binary
- If the first character of the topic "inTopic" is an 1, switch ON the ESP Led,
else switch it off
It will reconnect to the server if the connection is lost using a blocking
reconnect function. See the 'mqtt_reconnect_nonblocking' example for how to
achieve the same result without blocking the main loop.
To install the ESP8266 board, (using Arduino 1.6.4+):
- Add the following 3rd party board manager under "File -> Preferences -> Additional Boards Manager URLs":
http://arduino.esp8266.com/stable/package_esp8266com_index.json
- Open the "Tools -> Board -> Board Manager" and click install for the ESP8266"
- Select your ESP8266 in "Tools -> Board"
*/

#include <ESP8266WiFi.h>
#include <PubSubClient.h> // Si no existe la biblioteca
// Nota bajar .zip https://www.arduino-libraries.info/Libraries/pub-sub-client
// Instalar Menu: Programa / Submenu: Incluir Libreria / Anadir biblioteca

PubSubClientxxx.zip
const char* ssid = "TpLink_M5"; //SE PONE EL SSID DE LA CASA O LA RED DONDE ESTEMOS
const char* password = "8m7sIs9e4S"; //SU RESPECTIVA CONTRASEÑA //B8A56299v7RPBkP6
const char* mqtt_server = "broker.mqtt-dashboard.com";
const char* topico_salida = "TopicoTemperatura"; //PUEDE CAMBIAR
const char* topico_salida_2 = "TopicoOutLuz"; //PUEDE CAMBIAR
const char* topico_salida_3 = "TopicoOutDist"; //PUEDE CAMBIAR
const char* topico_entrada = "Temp"; //Clima o aire acondicionado

```

```

char sTopicoOutLuz[50];
char sTopicoOutTemp[50];
char sTopicoOutDist[50];

WiFiClient espClient;
PubSubClient client(espClient);
unsigned Long LastMsg = 0;
#define MSG_BUFFER_SIZE (50)
char msg[MSG_BUFFER_SIZE];
int value = 0;

// =====
// ----- Funcion que Conecta a Wifi del Router
// We start by connecting to a WiFi network
void setup_wifi() {
  delay(10);
  Serial.println(); Serial.print("Connecting to "); Serial.println(ssid);
  WiFi.mode(WIFI_STA); WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  randomSeed(micros());
  Serial.println(""); Serial.println("WiFi connected"); Serial.print("IP address: ");
  Serial.println(WiFi.localIP());
}

// ----- Funcion que Conecta a Broker MQTT
void setup_mqtt() {
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
  if (!client.connected()) {
    reconnect();
  }
}

// ----- Funcion que Conecta a Broker MQTT
void callback(char* topic, byte* payload, unsigned int length) {
  Serial.print("Message arrived ["); Serial.print(topic); Serial.print("] ");
  for (int i = 0; i < length; i++) {
    Serial.print((char)payload[i]);
  }
  Serial.println();
  digitalWrite(D5, HIGH); delay(100); digitalWrite(D5, LOW); // Aviso Led azul en D5 Pin 8

  // Switch on the LED if an 1 was received as first character
  if ((char)payload[0] == '1') {
    digitalWrite(D6, HIGH); // Turn the LED on (Note that LOW is the voltage level
    // but actually the LED is on; this is because
    // it is active low on the ESP-01). No es cierto en D6.
  } else {
    digitalWrite(D6, LOW); // Turn the LED off by making the voltage HIGH
  }
}

// ----- Funcion que Conecta o reconecta a Broker MQTT
void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Create a random client ID
    String clientId = "ESP8266Client-";
    clientId += String(random(0xffff), HEX);
    // Attempt to connect
    if (client.connect(clientId.c_str())) {
      Serial.println("connected");
      // Once connected, publish an announcement...
      //client.publish("outTopic", "hello world");
      //client.subscribe(topico_entrada);
      //client.subscribe(topico_salida); //OJO Quitar el comentario para hacerlo suscribe
    }
  }
}

```

```

        client.publish(topico_salida, sTopicoOutTemp); //<-----
        client.publish(topico_salida_2, sTopicoOutLuz); //<-----
        client.publish(topico_salida_3, sTopicoOutDist); //<-----

        // ... and resubscribe
        //client.subscribe("inTopic");

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

void conectarMQTT() {
    if (!client.connected()) {
        reconnect();
    }
    client.loop();

    unsigned long now = millis();
    if (now - LastMsg > 5000) {
        LastMsg = now;
        ++value;

        //  snprintf (msg, MSG_BUFFER_SIZE, "hello world #%ld", value);

        Serial.print(now); Serial.print("Publish message: "); Serial.println(sTopicoOutTemp);
        client.publish(topico_salida, sTopicoOutTemp);

        client.publish(topico_salida_2, sTopicoOutLuz);

        client.publish(topico_salida_3, sTopicoOutDist);

        // client.publish(topico_salida, msg);
        //  client.publish(topico_entrada, msg);
    }
}

// =====
// ----- Funcion que lee temperatura y humedad con el sensor DHT11
void medirTemperatura() {
    delay(2000);
    float h = dht.readHumidity();
    float t = dht.readTemperature();
    float f = dht.readTemperature(true);

    if (isnan(h) || isnan(t) || isnan(f)) {
        Serial.print("Falló al leer el sensor DHT\n");
        return;
    }

    float hif = dht.computeHeatIndex(f, h);
    float hic = dht.computeHeatIndex(t, h, false);

    Serial.print("Humedad: "); Serial.print(h);
    Serial.print(", Temperatura: "); Serial.print(t);
    Serial.print("(C), "); Serial.print(f);
    Serial.print("(F), Indice de calor (C)"); Serial.print(hic);
    Serial.print(", Indice de calor (F)"); Serial.print(hif);
    Serial.print("\n");

    //Threshold o nivel de umbral para prender Led
    Serial.println(t);
    if (t > 26)
        digitalWrite(D6, HIGH);
    else
        digitalWrite(D6, LOW);

    //snprintf (sTopicoOutTemp, MSG_BUFFER_SIZE, "Temperatura(C) %5.2f, Humedad %5.2f #%ld", t, h, value);
    snprintf (sTopicoOutTemp, MSG_BUFFER_SIZE, "{ \"t\":%5.2f, \"h\":%5.2f}", t, h);
}

```

```

    Serial.print(sTopicoOutTemp);
}

// =====
// ----- Funcion detector de iluminacion mediante un fotorresistor
// Puerto A0, Lee valores de 0 a 1023 dependiendo de la resistencia
void medirLuzledPhotoResistor() {
    x = analogRead(photoPin);
    Serial.print("Valor de voltaje ");
    Serial.println(x);
    if (x < 100)
        digitalWrite(ledphotoResistor, HIGH);
    else
        digitalWrite(ledphotoResistor, LOW);

    snprintf (sTopicoOutLuz, MSG_BUFFER_SIZE, "%d", x);
}

// =====

float medirDistanciaCm() {
    digitalWrite(trigPin, LOW);
    delayMicroseconds(3);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(5);
    digitalWrite(trigPin, LOW);

    //listen for echo
    float tUs = pulseIn(echoPin, HIGH); //microseconds
    float t = tUs / 1000.0 / 1000.0 / 2.0;    //s
    float d = t*v; //m
    float dCm = d*100; // cm
    Serial.print("Distancia(cm): ");
    Serial.println(dCm);
    delay(200); //ms
    snprintf (sTopicoOutDist, MSG_BUFFER_SIZE, "%d", dCm);
}

// =====
void setup() {
    pinMode(D0,OUTPUT); // Led del fotorresistor
    pinMode(D5,OUTPUT); // Led blinking
    pinMode(D6,OUTPUT); // Led del sensor de temperatura

    // ----- Iniciar consola: Herramientas, Monitor serie
    Serial.begin(9600); // Iniciar consola

    // ----- Abrir Wifi y MQTT
    setup_wifi();
    setup_mqtt();

    // ----- Inicializacion sensor temperatura - humedad
    Serial.println(F("DHTxx test!"));
    dht.begin();

    // ----- Inicializacion fotorresistor
    pinMode(ledphotoResistor, OUTPUT);
    pinMode(photoPin, INPUT);
    Serial.println("Iniciando Lectura fotorresistor");

    // ----- Inicializacion sensor de distancia
    pinMode(trigPin,OUTPUT);
    pinMode(echoPin,INPUT);
    Serial.println("Iniciando Sensor Distancia");
}

void loop() {
    // blinking
    digitalWrite(D5,HIGH);
    delay(500);
    digitalWrite(D5,LOW);
}

```



```

    delay(500);

    medirTemperatura();
    medirLuzLedPhotoResistor();
    medirDistanciaCm();
    conectarMQTT();
    Serial.println("*****");
}
// =====

```

Programa desarrollado detección movimiento, fuerza

```

#define LED D4
#define PIR D1
#define FORCE_SENSOR_PIN A0
#define RED_LED D3
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

const char* ssid = "iPhone de Santiago";
const char* password = "1234567786";
const char* mqtt_server = "broker.mqtt-dashboard.com";
const char* topico_salida = "TopicoMovimiento";
const char* topico_salida2 = "TopicoPeso";
char sTopicoOutMovimiento[50];
char sTopicoOutPeso[50];

WiFiClient espClient;
PubSubClient client(espClient);
unsigned long lastMsg = 0;
#define MSG_BUFFER_SIZE (50)
int value = 0;

void setup_wifi() {
    delay(10);
    Serial.println(); Serial.print("Connecting to "); Serial.println(ssid);
    WiFi.mode(WIFI_STA); WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    randomSeed(micros());
    Serial.println(""); Serial.println("WiFi connected"); Serial.print("IP address: ");
    Serial.println(WiFi.localIP());
}

void setup_mqtt() {
    client.setServer(mqtt_server, 1883);
    client.setCallback(callback);
    if (!client.connected()) {
        reconnect();
    }
}

void callback(char* topic, byte* payload, unsigned int length) {
    Serial.print("Message arrived ["); Serial.print(topic); Serial.print("] ");
    for (int i = 0; i < length; i++) {
        Serial.print((char)payload[i]);
    }
    Serial.println();
}

void reconnect() {
    // Loop until we're reconnected
    while (!client.connected()) {
        Serial.print("Attempting MQTT connection...");
        // Create a random client ID
        String clientId = "ESP8266Client-";
    }
}

```

```

        clientId += String(random(0xffff), HEX);
        // Attempt to connect
        if (client.connect(clientId.c_str())) {
            Serial.println("connected");
            // Once connected, publish
            client.publish(topico_salida, sTopicoOutMovimiento);
            client.publish(topico_salida2, sTopicoOutPeso);
        } else {
            Serial.print("failed, rc="); Serial.print(client.state()); Serial.println(" try again in 5 seconds");
            // Wait 5 seconds before retrying
            delay(5000);
        }
    }
}

void conectarMQTT() {
    if (!client.connected()) {
        reconnect();
    }
    client.loop();
    unsigned long now = millis();
    if (now - lastMsg > 5000) {
        lastMsg = now;
        ++value;
        client.publish(topico_salida, sTopicoOutMovimiento);
        client.publish(topico_salida2, sTopicoOutPeso);
    }
}

void setup() {
    // put your setup code here, to run once:
    setup_wifi();
    setup_mqtt();
    pinMode(PIR, INPUT);
    pinMode(LED, OUTPUT);
    pinMode(REDA_LED, OUTPUT);
    Serial.begin(9600);
}

void loop() {
    // put your main code here, to run repeatedly:

    int senValue = digitalRead(PIR);
    int analogReading = analogRead(FORCE_SENSOR_PIN);
    Serial.println(senValue);
    Serial.print("The force sensor value = ");
    Serial.print(analogReading); // print the raw analog reading

    if (senValue == HIGH) {
        digitalWrite(LED, HIGH);
    }
    else {
        digitalWrite(LED, LOW);
    }
    if (analogReading < 800) {
        Serial.println(" -> no pressure");
        digitalWrite(REDA_LED, HIGH);
    } else {
        digitalWrite(REDA_LED, LOW);
    }
    snprintf(sTopicoOutMovimiento, MSG_BUFFER_SIZE, "%d", senValue);
    snprintf(sTopicoOutPeso, MSG_BUFFER_SIZE, "%d", analogReading);
    conectarMQTT();
    delay(1000);
}

```


Parámetros en la página MQTT

Connection

● disconnected

Host: Port: ClientID: Connect

Username: Password: Keep Alive: SSL: ☐ Clean Session: ☒

Last-Will Topic: Last-Will QoS: Last-Will Retain: ☐

Last-Will Message:

Publish

Topic: QoS: Retain: ☐ Publish

Message:

Subscriptions

Add New Topic Subscription

Qos: 2

TopicoMovimiento

X

Qos: 2

TopicoPeso

X

Qos: 2

TopicoTemperatura

X

Qos: 2

TopicoLuz

X

Qos: 2

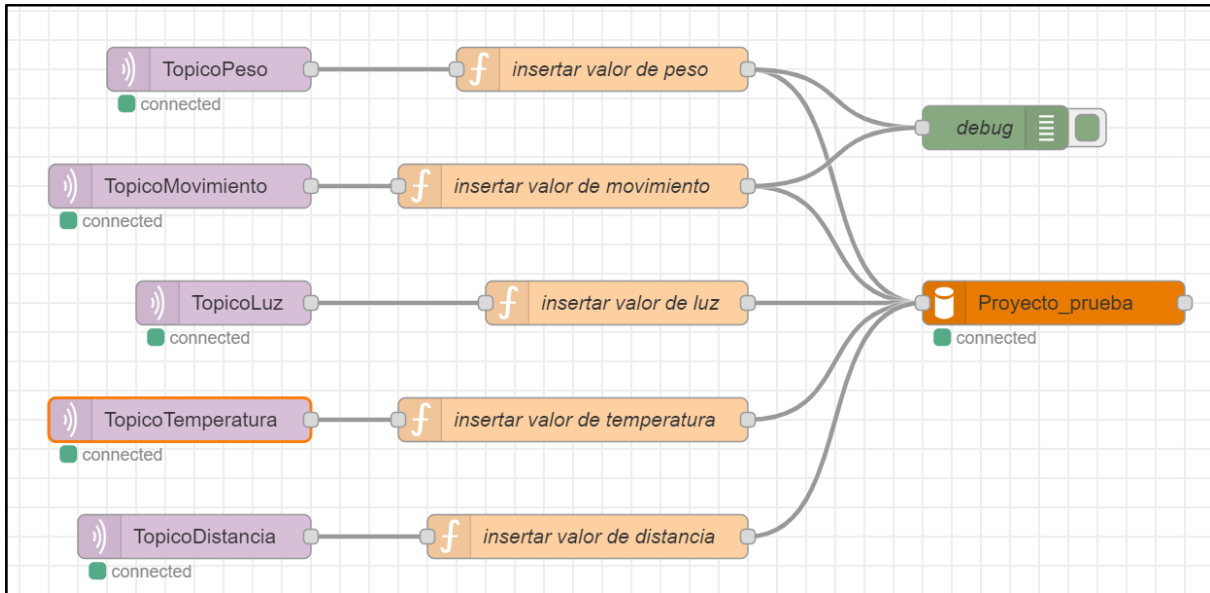
TopicoDistancia

X

Messages

Cada sensor está suscrito a un tópico distinto, por lo tanto se hacen las suscripciones a los tópicos respectivos. Todo lo demás se queda con la configuración predeterminada.

Elementos y descripción de campos de nodeRED



En la imagen anterior observamos nodos a la izquierda que se conectan a MQTT y están suscritos a distintos tópicos para extraer los datos que llegan a esos tópicos. En medio están los nodos que contienen la función para crear el comando SQL de insertar un dato a la base de datos. El nodo a la derecha es el nodo que se conecta a la base de datos y recibe los comandos generados por los nodos anteriores.

Las dos imágenes muestran la configuración de un nodo MQTT en Node-RED.

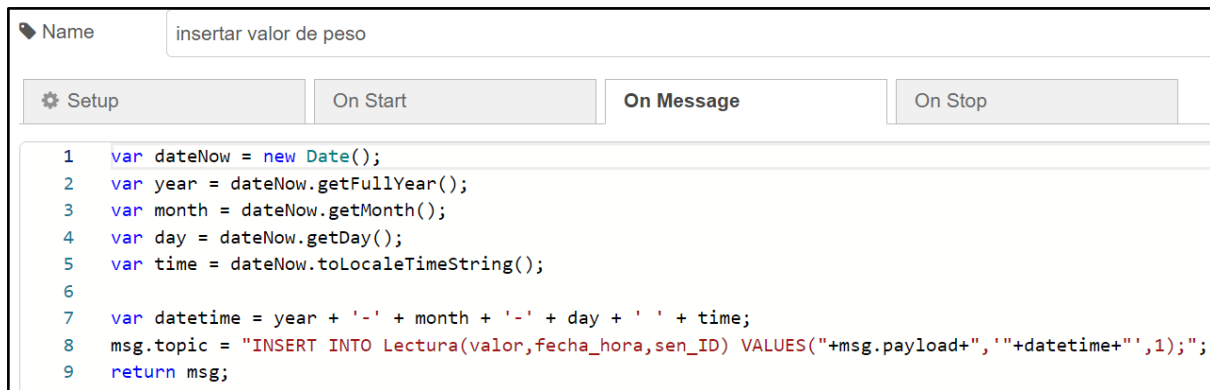
La imagen de la izquierda muestra el panel 'Edit mqtt in node' con las siguientes propiedades:

- Server: MQTT 01
- Action: Subscribe to single topic
- Topic: TopicoPeso
- QoS: 2
- Output: auto-detect (parsed JSON object, string or buf)
- Name: Name

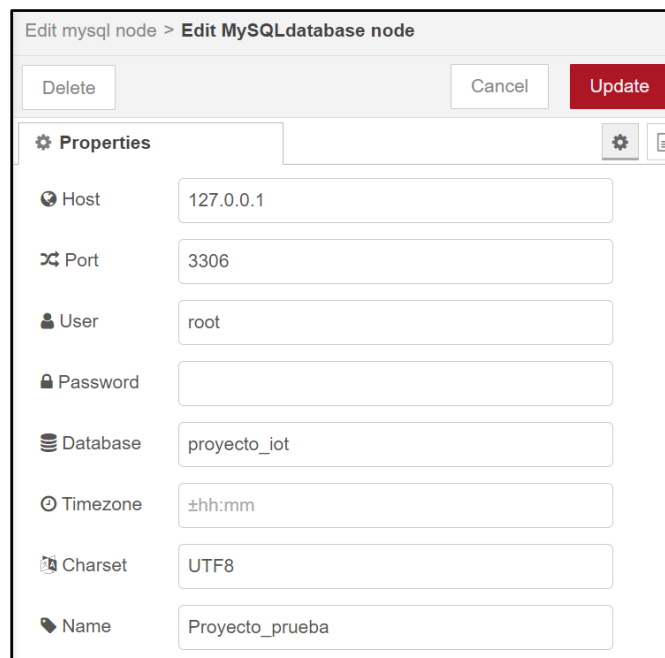
La imagen de la derecha muestra el panel 'Edit mqtt in node > Edit mqtt-broker node' con las siguientes propiedades:

- Name: MQTT 01
- Connection tab selected
- Server: broker.mqtdashboard.com
- Port: 1883
- Connect automatically: ☒
- Use TLS: ☐
- Protocol: MQTT V3.1.1
- Client ID: Leave blank for auto generated
- Keep Alive: 60
- Session: ☒ Use clean session

Las dos imágenes anteriores muestran el contenido de los nodos que se conectan a MQTT. En la izquierda observamos que el nodo está suscrito al tópico TopicoPeso. Los otros nodos están suscritos a los tópicos respectivos. En la imagen de la derecha observamos la configuración para que el nodo se pueda conectar al MQTT. Esto es exactamente igual para los otros nodos.

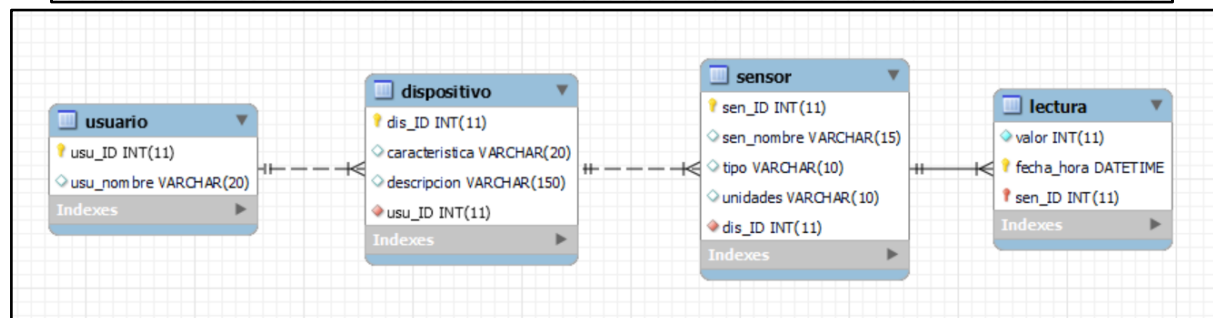
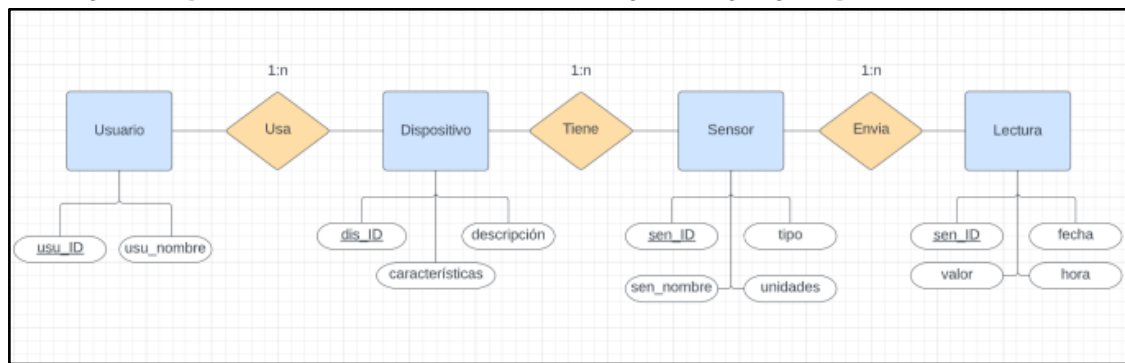


En la imagen anterior se observa el comando de SQL para insertar la lectura a la base de datos. Se genera una variable con la fecha y hora actual y se crea la estructura requerida para ingresarla a la base de datos. Esta función es exactamente igual para cada nodo función, con la excepción de que el número que se inserta para el campo *sen_ID* cambia para cada sensor (peso = 1, movimiento = 2, luz = 3, temperatura = 4, distancia = 5).















































Finalmente observamos en esta imagen la configuración del último nodo para conectarse a la base de datos. Las propiedades de Host y de Port estaban definidos de manera predeterminada, lo único que se agregó fue la propiedad de User (se ingresó "root") y la de Database (se ingresó el nombre de nuestra base de datos, "proyecto_iot").

Tablas y campos de base de datos en MySQL y ejemplos de datos reales



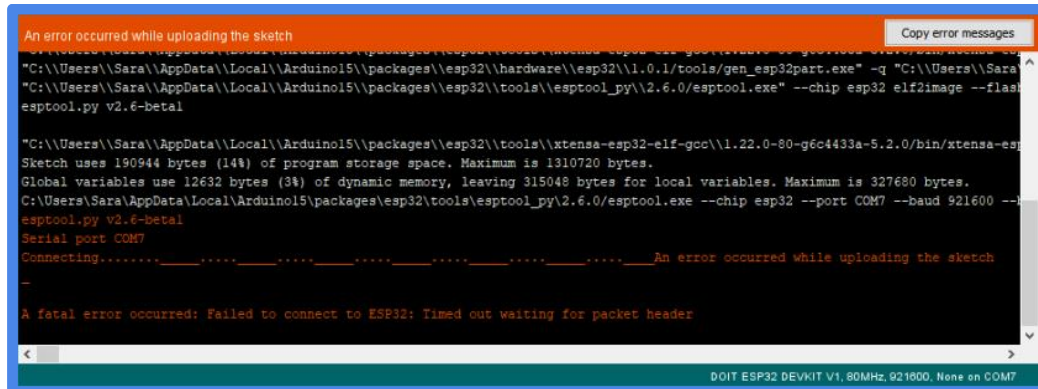
Nuestra base de datos consiste de cuatro tablas: usuario, dispositivo, sensor, y lectura. Se pueden observar las llaves primarias y las llaves foráneas en la imagen anterior.

<div><div><div>←</div><div>T</div><div>→</div></div><div></div></div>				valor	fecha_hora	sen_ID
<input type="checkbox"/>	 Edit	 Copy	 Delete	6	2022-11-04 12:23:05	1
<input type="checkbox"/>	 Edit	 Copy	 Delete	1	2022-11-04 12:23:05	2
<input type="checkbox"/>	 Edit	 Copy	 Delete	1024	2022-11-04 12:23:10	1
<input type="checkbox"/>	 Edit	 Copy	 Delete	1	2022-11-04 12:23:10	2
<input type="checkbox"/>	 Edit	 Copy	 Delete	1	2022-11-04 12:23:14	2
<input type="checkbox"/>	 Edit	 Copy	 Delete	12	2022-11-04 12:23:15	1
<input type="checkbox"/>	 Edit	 Copy	 Delete	12	2022-11-04 12:23:20	1
<input type="checkbox"/>	 Edit	 Copy	 Delete	1	2022-11-04 12:23:20	2
<input type="checkbox"/>	 Edit	 Copy	 Delete	0	2022-11-04 12:23:24	2
<input type="checkbox"/>	 Edit	 Copy	 Delete	12	2022-11-04 12:23:25	1
<input type="checkbox"/>	 Edit	 Copy	 Delete	0	2022-11-04 12:23:29	2
<input type="checkbox"/>	 Edit	 Copy	 Delete	12	2022-11-04 12:23:30	1
<input type="checkbox"/>	 Edit	 Copy	 Delete	1024	2022-11-04 12:24:01	1
<input type="checkbox"/>	 Edit	 Copy	 Delete	1	2022-11-04 12:24:01	2
<input type="checkbox"/>	 Edit	 Copy	 Delete	5	2022-11-04 12:24:06	1
<input type="checkbox"/>	 Edit	 Copy	 Delete	1	2022-11-04 12:24:06	2
<input type="checkbox"/>	 Edit	 Copy	 Delete	1024	2022-11-04 12:24:11	1

En esta imagen se observan las lecturas recibidas por los sensores de peso ($\text{sen_ID} = 1$) y de movimiento ($\text{sen_ID} = 2$).

Procesos o pruebas que no funcionaron

Durante la realización del proyecto únicamente contamos con un proceso que no funcionó completamente que fue en la implementación del nodeMCU con los sensores de temperatura, el de distancia y el de luminosidad, ya que algunas veces marca errores de conexión y no mandaba ningún dato de salida. *ANEXO EJEMPLO ERROR.*



```
An error occurred while uploading the sketch
Copy error messages

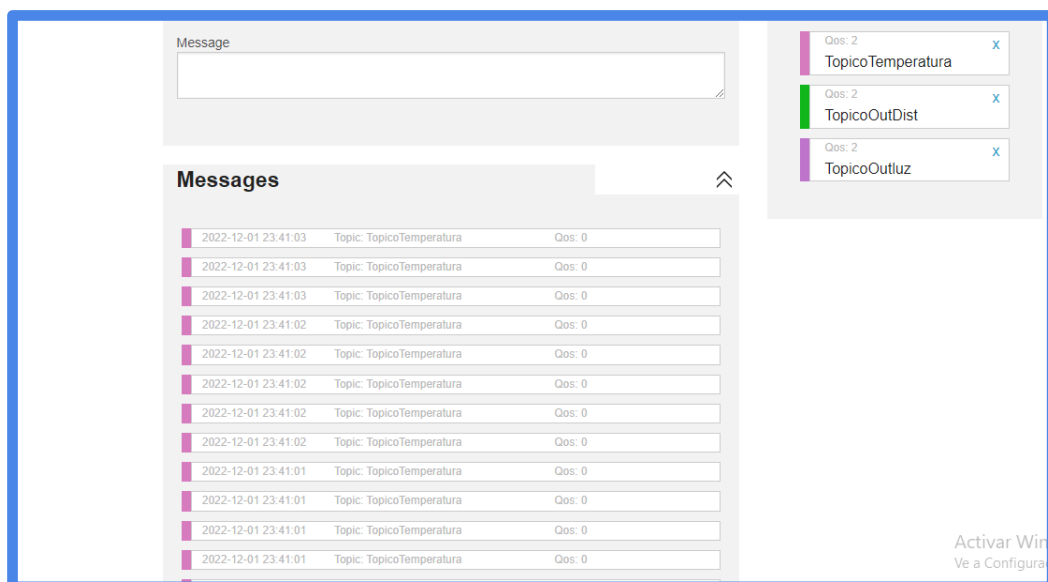
"C:\Users\Sara\AppData\Local\Arduino15\packages\esp32\hardware\esp32\1.0.1/tools/gen_esp32part.exe" -q "C:\Users\Sara\AppData\Local\Arduino15\packages\esp32\tools\esptool_py\2.6.0/esptool.exe" --chip esp32 elf2image --flash_size 4M --port COM7 --baud 921600 --esptool.py v2.6-beta1

"C:\Users\Sara\AppData\Local\Arduino15\packages\esp32\tools\xtensa-esp32-elf-gcc\1.22.0-80-g6c4433a-5.2.0/bin/xtensa-esp32-elf-gcc.exe" -x -c "C:\Users\Sara\AppData\Local\Arduino15\packages\esp32\hardware\esp32\1.0.1/tools/gen_esp32part.exe" -q "C:\Users\Sara\AppData\Local\Arduino15\packages\esp32\tools\esptool_py\2.6.0/esptool.exe" --chip esp32 --port COM7 --baud 921600 --esptool.py v2.6-beta1
Sketch uses 190944 bytes (14%) of program storage space. Maximum is 1310720 bytes.
Global variables use 12632 bytes (3%) of dynamic memory, leaving 315048 bytes for local variables. Maximum is 327680 bytes.
C:\Users\Sara\AppData\Local\Arduino15\packages\esp32\tools\esptool_py\2.6.0/esptool.exe --chip esp32 --port COM7 --baud 921600 --esptool.py v2.6-beta1
Serial port COM7
Connecting.....
An error occurred while uploading the sketch
A fatal error occurred: Failed to connect to ESP32: Timed out waiting for packet header

DOIT ESP32 DEVKIT V1, 80MHz, 921600, None on COM7
```

Este error se logró solucionar después de algunas pruebas y pudo mandar datos al monitor serie.

Otro de los errores que sucedió y que un principio si funciono fue el envío de los datos de salida arrojados al monitor serie y a partir de esos datos mandarlos al broker MQTT, este proceso funcionó temporalmente ya que mandaba datos de salida pero no se escribían en el broker MQTT. *ANEXO EJEMPLO ERROR*



Message

Messages

Time	Topic	QoS
2022-12-01 23:41:03	Topico: Temperatura	Qos: 0
2022-12-01 23:41:03	Topico: Temperatura	Qos: 0
2022-12-01 23:41:03	Topico: Temperatura	Qos: 0
2022-12-01 23:41:02	Topico: Temperatura	Qos: 0
2022-12-01 23:41:02	Topico: Temperatura	Qos: 0
2022-12-01 23:41:02	Topico: Temperatura	Qos: 0
2022-12-01 23:41:02	Topico: Temperatura	Qos: 0
2022-12-01 23:41:02	Topico: Temperatura	Qos: 0
2022-12-01 23:41:01	Topico: Temperatura	Qos: 0
2022-12-01 23:41:01	Topico: Temperatura	Qos: 0
2022-12-01 23:41:01	Topico: Temperatura	Qos: 0
2022-12-01 23:41:01	Topico: Temperatura	Qos: 0

Activar Win
Ve a Configura

ENLACE AL VIDEO DE LA EXPLICACIÓN DEL PROYECTO

<https://youtu.be/X0mnhQJRLFs>

CONCLUSIONES DEL RETO

Este proyecto consiste en el funcionamiento correcto de muchas partes; primero tenemos las conexiones realizadas desde nuestros sensores y actuadores hacia nuestro microcontrolador, luego tenemos la lectura de los datos de los mismos para poder enviarlos en un formato específico a un broker, que luego son recibidos por otra herramienta para finalmente enviarse a una base de datos. Hay muchas maneras en las que el proyecto puede fallar o darnos resultados que no tienen mucho sentido, así como nos pasó con uno de los microcontroladores. Desconocemos si la razón de la falla fue eléctrica, como una conexión mal hecha, un protoboard defectuoso, o incluso un microcontrolador defectuoso, o pudo haber sido algún error en código que no logramos ver lo que causó que los datos leídos y enviados al broker MQTT no se enviarán correctamente. Aun así, logramos cumplir nuestro objetivo con la parte principal del proyecto con todas las conexiones físicas y digitales funcionando como se esperaba.

REFLEXIÓN INDIVIDUAL

El desarrollo de este bloque fue sin duda alguna uno de los retos, aprendizajes y vivencia más significantes que he tenido en toda mi carrera, tanto por lo que se vió en relación a los tópicos de la materia como todo lo que logramos aplicar en nuestra vida gracias a lo que aprendimos y vivimos en ella. Sin embargo, el trabajo en equipo que se tuvo durante el desarrollo de todo el bloque me permitió tanto a mí como a mis compañeros el fuerte aprendizaje y valoración de los temas que se trataron, vivieron y evaluaron. Conforme al desarrollo del proyecto, considero que realmente fue un gran apoyo el que recibí por parte de mi equipo ya que la distribución de trabajo fue equitativa y aprendimos bastante de cada una de las actividades ya que siempre nos apoyamos entre todos para la resolución de una problemática en común, además de que se llegó a disfrutar de mejor manera el avance y desarrollo del proyecto de la mano de otras tres personas apoyándome.

En relación al aprendizaje que tuve del reto, me llevó bastante tiempo entender algunos de los temas, sobre todo la codificación en el ID de arduino para realizar las conexiones correctas del nodeMCU, ya que cada uno de los sensores que se utilizaron tenían un funcionamiento e instrucciones diferentes para aprovechar y ejecutar su correcto funcionamiento. De igual manera, las conexiones que se tuvieron con las distintas páginas web, como lo fueron el broker de MQTT y node-RED, tuvieron sus distintas complicaciones, sobre todo entre la conexión entre MQTT y el nodeMCU debido a los diversos errores con el protoboard y su cableado con los distintos sensores que muchas veces generaban falsos. Pero si no hubiera sido por esos fallos y errores no hubiera intentado y repasado la conexión varias veces, lo cual me fue de mucho beneficio para mi aprendizaje constante y la práctica de mis conocimientos de IoT. Por otro lado, la documentación del propio reto me fue bastante interesante debido a que mostré el verdadero propósito y aplicación de los diversos proyectos de IoT y unos de los muchos avances que pueden tener estos proyectos en nuestra vida y usos cotidianos que, en resumen, son posibles de realizar con mi práctica y aprendizaje continuo de estos temas. Realmente, considero muy valiosa mi participación y experiencia en este bloque ya que me ayudará a expandir y aspirar a más nuevos conocimientos a lo largo de mi carrera.

REFERENCIAS

- Rose, K., Eldridge, S., & Chapin, L. (2015, October 1). LA INTERNET DE LAS COSAS— UNA BREVE RESEÑA. Internet Society (ISOC). <https://www.internetsociety.org/wp-content/uploads/2017/09/report-InternetOfThings-20160817-es-1.pdf>
- GIMENO. (2022). GIMENO. Obtenido de SENSOR DE FUERZA RESISTIVO (FSR): [https://electronicagimeno.com/sensor-de-fuerza-resistivo-fsr#:~:text=El%20FSR%20\(Force%20Sensing%20Resistor,amigable%20con%20placas%20de%20pruebas.](https://electronicagimeno.com/sensor-de-fuerza-resistivo-fsr#:~:text=El%20FSR%20(Force%20Sensing%20Resistor,amigable%20con%20placas%20de%20pruebas.)
- MecatrónicaLATAM. (23 de Abril de 2021). MecatrónicaLATAM. Obtenido de LDR o fotoresistor: <https://www.mecatronicalatam.com/es/tutoriales/sensores/sensor-de-luz/ldr/>
- Microseguir. (2022). Microseguir. Obtenido de QUÉ ES UN SENSOR DE MOVIMIENTO PIR: <https://microseguir.com/que-es-un-sensor-de-movimiento-pir/#>
- Naylamp. (2022). Naylamp. Obtenido de SENSOR ULTRASONIDO HC-SR04: <https://naylampmechatronics.com/sensores-proximidad/10-sensor-ultrasonido-hc-sr04.html>
- PAESSLER. (2022). PAESSLER. Obtenido de ¿Qué es MQTT?: <https://www.paessler.com/es/it-explained/mqtt#:~:text=Arquitectura%20de%20MQTT,->

[MQTT%20se%20ejecuta&text=Un%20br%C3%B3ker%20es%20el%20servidor,e
ditor%2C%20un%20suscriptor%20o%20ambos.](#)

- Robledano, A. (24 de Septiembre de 2019). OpenWebinars. Obtenido de Qué es MySQL: Características y ventajas: <https://openwebinars.net/blog/que-es-mysql/>
- sinelec. (4 de Febrero de 2022). sinelec. Obtenido de ¿QUÉ ES NODE-RED Y PARA QUÉ SIRVE?: <https://blog.gruposinelec.com/actualidad/que-es-node-red-y-para-que-sirve/>
- TodoMicro. (2022). TodoMicro. Obtenido de Sensor De Temperatura Y Humedad Dht11 Arduino: [https://www.todomicro.com.ar/insumos/224-sensor-de-temperatura-y-humedad-dht11-arduino.html#:~:text=El%20DHT11%20es%20un%20sensor%20digital%20de%20temperatura%20y%20humedad,\(no%20posee%20salida%20anal%C3%B3gica\)](https://www.todomicro.com.ar/insumos/224-sensor-de-temperatura-y-humedad-dht11-arduino.html#:~:text=El%20DHT11%20es%20un%20sensor%20digital%20de%20temperatura%20y%20humedad,(no%20posee%20salida%20anal%C3%B3gica))

.