

# Proyecto ‘AIr condIoTioner’ Parte IoT

Diego Iván Perea Montealegre, Carlos Ivan Osorio Moreno, Gabriel Jeannot Viaña, Samir Hassan Ordóñez

Facultad de Ingeniería, Universidad Autónoma de Occidente  
Cali, Valle del Cauca

## DEFINICIÓN DEL PROBLEMA

A medida que avanza la tecnología trae consigo soluciones a diversas situaciones que nos han permitido mejorar nuestra calidad de vida y/o del ecosistema. Pero con cada solución llegan otras situaciones más en forma de daño colateral. Una de ellas es la excesiva dependencia del servicio de energía eléctrica, y con ello el consumo masivo de este servicio que en su mayoría continúa proviniendo de fuentes de generación no sustentables.

El mayor problema de esta situación es que una parte de la energía consumida en el día a día es malgastada y desaprovechada debido al estilo de vida de la gente y la manera en cómo funcionan las cosas que nos rodean. Por ello desde esta materia, se ha buscado sacar partido de las nuevas tecnologías con el fin de aportar a la necesidad de generar dispositivos con mayor eficiencia energética.



*Figura.* El 66% del consumo de energía se utiliza principalmente para la calefacción, refrigeración y ventilación.

## DESCRIPCIÓN DE LA SOLUCIÓN PLANTEADA

Por tanto, haciendo uso de esta temática y enfocándonos en mejorar día a día el ecosistema que nos rodea, se presenta ‘**AIr condIoTioner**’, un nuevo modelo en el funcionamiento que presentan los aires acondicionados que existen actualmente. El cual pretende asegurar un mayor ahorro de energía en el proceso de enfriamiento de estos equipos a lo largo del día.

Pero, ¿Cómo funciona?

Su proceso de funcionamiento se basa en la incorporación de un sistema de sensado de variables de temperatura y humedad del entorno en el que se encuentre (habitaciones, salas

de estar, etc), y a partir del procesamiento de estos datos por medio de un modelo de aprendizaje automático, logre predecir las condiciones climáticas que se avecinan, y a través de estos resultados enviados al aire acondicionado poder realizar un autoajuste para hallar la mejor curva de enfriamiento en un determinado tiempo y/o su manutención en una temperatura dada. Permitiendo al final, gracias al internet de las cosas, desplegar esta información al usuario a través de una aplicación web.

## **DEFINICIÓN DE LAS VARIABLES**

Como se mencionó anteriormente, las variables a utilizarse dentro del sistema son la temperatura y humedad, las cuales son consideradas como las mediciones más importantes dentro del contexto de un sistema de enfriamiento, y mediante las cuales, a partir de su correlación, se lograra determinar un punto de equilibrio ideal para un óptimo funcionamiento del aire acondicionado.

## **DEFINICIÓN DE PROCESAMIENTO Y ANÁLISIS DE LOS DATOS**

El inicio de esta fase empieza con la recolección y tratamiento de los datos, a los cuales se les hará pasar por diferentes “filtros” con el fin de obtener una información mucho más generalizada y fácil de procesar, entre los más importantes destaca el proceso de normalización, inherente a casi cualquier dataset, para poder tener en un rango definido el valor de los datos, eficientando el tiempo de procesamiento en el modelo de IA y el costo computacional requerido.

Posterior al tratamiento de los datos, estos se encuentran listos para ingresar al modelo de IA, el cual generará la extracción de características a partir de la generalización de estos y la identificación de patrones de comportamiento que permitan la predicción futura de dichas variables. Para ello, en este contexto, se hará uso de técnicas de procesamiento de datos secuenciales para predicción, y el uso particular de dos arquitecturas de ‘Deep learning’, las cuales son **redes convolucionales 1D** y **redes recurrentes RNN**.

## **DEFINICIÓN DE LOS USUARIOS Y DESPLIEGUE DE LA INFORMACIÓN**

El aplicativo se encuentra orientado a toda la comunidad que en su hogar, trabajo o espacio cerrado posea un sistema de enfriamiento (escalable a sistemas de calefacción), y que con su uso permita un ahorro eficiente de la energía eléctrica gastada por estos equipos, contribuyendo a la sustentabilidad y, por qué no, al ahorro económico.

Pero dentro del diseño de la aplicación, los dos tipos de usuarios los cuales tendrán diferentes posibilidades de uso, gestión y despliegue del sistema se encontraran divididos en:

- **Usuario:** Este usuario tendrá la facultad de poder crear/borrar y gestionar su cuenta, obtener información (gráficas en un tiempo determinado) de datos asociados a sus nodos (dispositivos enlazados a su cuenta), así mismo tendrá la posibilidad de ajustar variables en la aplicación como tiempo de llegada y temperatura ideal, para que el sistema pueda realizar la predicción con ese tiempo y temperatura para encontrar el

punto ideal de la curva de enfriamiento y una vez lograda esa temperatura poder mantenerla automáticamente.

- Administrador: Podrá realizar todas las anteriores operaciones y adicionalmente podrá gestionar las bases de datos de todos los usuarios, pudiendo eliminar nodos, consultar, modificar y gestionar cuentas de todos los usuarios del sistema.

Tomando en cuenta lo anterior, y a manera de referencia en el usuario principal ‘Usuario’, se presenta a continuación un prototipo de baja fidelidad sobre el diseño de la aplicación móvil del sistema desplegado en un smartphone.

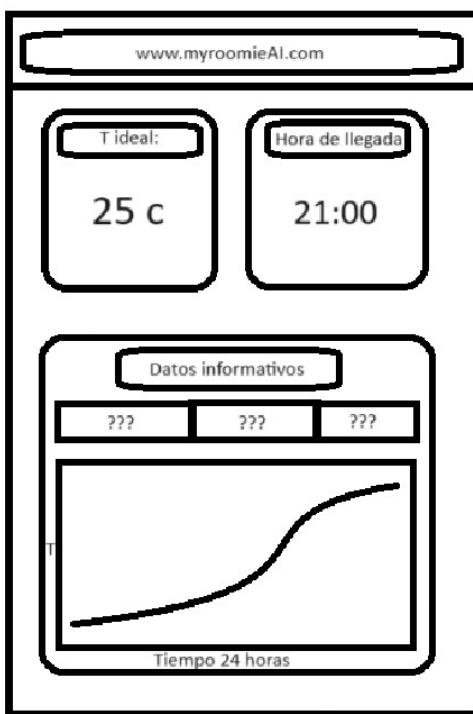


Figura. Interfaz móvil de usuario del sistema.

#### Sensores de temperatura

Sensor de temperatura LM35 está calibrado directamente en grados Celsius (centígrados) y no debemos hacer ningún tipo de conversión ni de calibración externa. La salida es analógica y la mediremos con una entrada analógica de Arduino. El gran problema de este sensor es que realmente solo podemos medir temperaturas entre 2° C y 150° C a no ser que utilicemos voltajes negativos. Voltaje de operación: de 4 V a 30 V , Rango de temperaturas: -55° C a 150° C , Precisión: ±0,5° C , Conversión: 10 mV / °C, Tiempo de respuesta (100%): 4 min. , Offset: 0 V. Costo: 1 euro aprox.

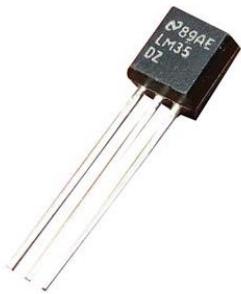


Figura. LM35

Sensor temperatura TMP36 , Este sensor es analógico y algo más caro, su precio ronda los 2,5€ pero si le sumas los gastos de envío, [los encuentras por casi 6€](#). Voltaje de operación: de 2,7 V a 5,5 V ,Rango de temperaturas: -40° C a 150° C aunque a partir de los 125°C ya no es lineal ,Precisión:  $\pm 2^\circ \text{C}$  ,Conversión: 10 mV / °C ,Tiempo de respuesta (100%): 8 min. ,Offset: 0.5 V



Figura. TMP36

Sensor temperatura Arduino TC74 , sensor digital y cada uno a unos 5€ con gastos de envío incluidos .Es un sensor de temperatura digital especialmente adecuado para aplicaciones de bajo coste. Es capaz de convertir la temperatura dentro del propio sensor y se transmite a través de un palabra digital de 8-bit. Voltaje de operación: de 2,7 V a 5,5 V Rango de temperaturas: -40° C a 125° C ,Precisión:  $\pm 2^\circ \text{C}$  de 25° C a 85° C y  $\pm 3^\circ \text{C}$  de 0° C a 125° C ,Resolución: 8-bit ,Muestras/segundo: 8.



Figura. TC74

#### Sensores de humedad

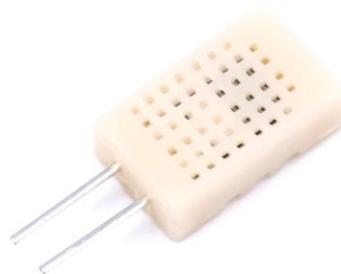
Sensor de humedad relativa hs1101, útil en todas las aplicaciones donde se necesita compensación de la humedad, presenta comportamiento capacitivo, a través de este se pueden generar mediciones

de humedad en aplicaciones de control de aire, como lo son las cabinas de los automóviles, electrodomésticos, sistemas de control que tengan que ver con procesos industriales, entre otros. Su diseño y fabricación le proporcionan prolongada vida útil y gran fidelidad en cuanto a las mediciones realizadas. Rango de Humedad :1~99% RH ,Sensibilidad:0.34pF/% RH,Exactitud: $\pm 2\%$  a 10 ~ 90% RH a 25°C ,Tiempo de respuesta:5 segundos (33% ~ 76% RH estático a 63%) ,Capacitancia180pF a 55% RH ,Voltaje de salida:5V~10V (Capacitivo) ,Rango de temperatura: -40°C ~100°C , con costo de 5 USD aproximadamente.



*Figura 6. hs1101*

Sensor de humedad HR202 es un nuevo tipo de componente sensible a la humedad que utiliza material de polímero orgánico. Tiene un amplio rango de absorción de humedad y un rendimiento estable a largo plazo. Se puede utilizar en almacenamiento, compartimentos, control de calidad del aire interior, automatización de edificios, sistema de control médico e industrial. Y una amplia gama de aplicaciones en el campo de la investigación científica. Parámetros característicos: (a 1kHz) Unidad: ohm ,Voltaje de alimentación: 1,5 V CA (onda sinusoidal máxima) ,Frecuencia de funcionamiento: 500Hz...2kHz ,Potencia nominal: 0,2 mW (onda sinusoidal máxima) ,Valor Central: (at25 °C 1kHz 1V CA 60% onda sinusoidal RH) 31.0kΩ. ,Rango de impedancia: (at25 °C 1kHz 1V CA 60% onda sinusoidal RH) 19,8... 50.2kΩ ,Precisión de detección de humedad:  $\pm 5\%$  RH ,costo de 3,50 USD aproximadamente



*Figura HR202*

Sensor de humedad HDS10 revela que el sensor es un componente de cambio positivo, que es insensible a la baja humedad y sensible solo a la alta humedad. Puede funcionar bajo voltaje de CC, con calidad estable y alta fiabilidad. ,Voltaje de alimentación: 0,8 v CC (voltaje de seguridad) ,Rango de temperatura de funcionamiento: 1-80 grados ,Rango de humedad de funcionamiento: 1-100% RH ,Rango de prueba de condensación: 94-100% RH ,costo de 3 USD aproximadamente.



Figura. HDS10

#### Sensores de temperatura y humedad

Sensor de temperatura y humedad HDC1080 , sensor digital , El HDC1080 tiene dos modos de funcionamiento: modo dormido y modo midiendo. Cuando el sensor de temperatura y humedad se enciende, automáticamente entra en modo dormido consumiendo una media de 100nA. En este modo, el sensor espera cualquier comando que le llegue por el protocolo I2C para despertar.Cuando recibe un comando para realizar una medida, pasa del modo dormido al modo midiendo. Una vez completada la medida, el sensor de temperatura y humedad regresa al modo dormido. Gracias a que todo esto lo hace de forma automática, se consiguen unos consumos muy bajos. Pero en realidad, el consumo también varía según el voltaje de alimentación y de la temperatura.

Relative humidity accuracy (Typ) (%RH) :  $\pm 2$

Temperature accuracy (Typ) ( $^{\circ}\text{C}$ ) :  $\pm 0.2$

Relative humidity operating range (Typ) (%RH): 0 - 100

Supply voltage (Min): 2.7

Supply voltage (Max): 5.5

Average supply current (Typ) (uA): 1.2 @ 1 sample/sec

Operating temperature range (C): -40 to 125

Cost : 10 USD aprox

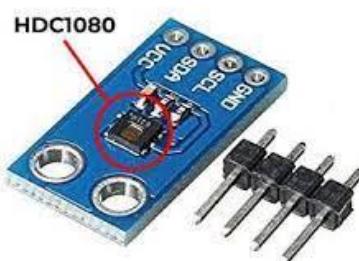
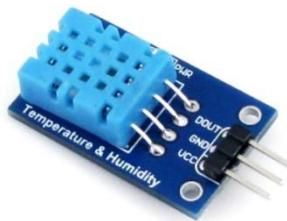


Figura . HDC1080

Los sensores DHT11 y DHT22 son los más básicos y los más utilizados para implementarlos con Arduino, estos sensores están compuestos en dos partes, un sensor de humedad capacitivo y un termistor, también constan de un circuito integrado básico en el interior que hace la conversión de

analógico a digital y este envía una señal digital con la temperatura y la humedad. El DHT11 es un sensor de temperatura y humedad digital básico y de muy bajo costo. Es capaz de detectar la temperatura y también la humedad relativa, que es la cantidad de vapor de agua en el aire frente al punto de saturación del vapor de agua en el aire. DHT11 es el módulo de temperatura y humedad más común para Arduino y Raspberry Pi. Por lo tanto, ampliamente favorecido por los entusiastas del hardware por sus muchas ventajas.



*Figura DHT11*

El DHT22 El DHT22 también conocido como AM2302 o RHT03, incluye un sensor de humedad capacitivo y un sensor de temperatura de alta precisión. Utiliza tecnología de adquisición de módulo digital dedicada y tecnología de detección de temperatura y humedad para garantizar una alta confiabilidad y una excelente estabilidad a largo plazo. El DHT22 también tiene un elemento de detección capacitivo y un elemento de medición de temperatura de alta precisión conectado a un microcontrolador de 8 bits de alto rendimiento. Por lo tanto, tiene las ventajas de una excelente calidad, una respuesta ultrarrápida, una gran capacidad anti-interferencias y un rendimiento de alto costo.



*Figura DHT22*

Los sensores de temperatura y humedad son muy parecidos , pero ¿ cual es la diferencia entre cada uno? (Observar Tabla 3)

	DHT11	DHT22
Rango de temperatura	-20 a 60°C	-40 a 80°C
Precisión de temperatura	$\pm 2\%$	$\pm 0.5\%$
Rango de humedad	5 a 95% RH	0 a 100% RH
Precisión de humedad	$\pm 5\%$	$\pm 2\%$
Costo	5 USD	11 USD

### Tabla Comparativa DHT11 vs DHT22

Teniendo en cuenta todas la comparaciones y a nivel de costos y rangos de temperatura para implementar en nuestro objeto iot se seleccionó al DHT11 .

## OBTENCIÓN DE DATOS



```
1 #include <Arduino.h>
2 #include <ArduinoJson.h>
3 //LIBERIAS PARA DHT11 (TEMPERATURA Y HUMEDAD)
4 #include <WiFi.h>
5 #include <NTPClient.h>
6 #include <WiFiUdp.h>
7 //LIBERIAS PARA DHT11 (TEMPERATURA Y HUMEDAD)
8 #include <dht/dht.h>
9 #include <dht/DHT.h>
10
11 //DEFINICION DE PINES DHT11
12 #define DHTPIN 4 //A = PIN D4
13 #define DHTTYPE DHT11
14 DHT dht(DHTPIN, DHTTYPE);
15
16 //CONFIG PARA ----FECHA Y HORA-----
17 //Replace with your network credentials
18 const char* ssid = "*****your*name*ssid*"; //name wifi
19 const char* password = "*****your*password*password*"; // clave de wifi
20 // Define NTP Client to get time
21 WiFiUDP ntptime;
22 NTPClient timeClient(ntptime);
23
24 // Variables to save date and time
25 String formattedDate;
26 String dayStamp;
27 String timeStamp;
28
29 ;
30 ;
31 ;
32
33 void setup() {
34 // Initialize Serial Monitor
35 Serial.begin(9600);
36 //-----CONFIG PARA -----HORA-----
37 Serial.begin(WIFI_S7S);
38 WiFi.begin(ssid, password);
39 while (WiFi.status() != WL_CONNECTED) {
40 delay(500);
41 }
42 // Initialize a NTPClient to get time
43 timeClient.begin();
44 // Set offset time in seconds to adjust for your timezone, for example:
45 // COLOMBIA -5 entones -5*3600 => -18000
46 timeClient.setTimeOffset(-18000); //Thailand +7 = 25200
47
48 }
49
50 ;
51
52 void loop() {
53 while(timeClient.update()) {
54 timeClient.forceUpdate();
55 }
56 // The formattedDate comes with the following format:
57 // 2018-05-28T16:00:12Z
58 // We need to extract date and time
59 formattedDate = timeClient.getFormattedDate();
60 // Extract date
61 int splitT = formattedDate.indexOf("T");
62 dayStamp = formattedDate.substring(0, splitT);
63 //Serial.println(dayStamp);
64 //Extract time
65 timeStamp = formattedDate.substring(splitT+1, formattedDate.length()-1);
66 //Serial.print("HOUR: ");
67 //Serial.println(timeStamp);
68
69 //-----CODIGO...TEMPERATURA Y HUMEDAD-----
70 float h = dht.readHumidity();
71 float t = dht.readTemperature();
72
73 /**
74 -----
75
76 //-----CODIGO JSON-----
77 String variable;
78
79 DynamicJsonDocument doc(1024);
80
81
82
83
84 doc["Fecha"] = dayStamp;
85 doc["Hora"] = timeStamp;
86 doc["Temperatura(C)"] = t;
87 doc["Humedad(% )"] = h;
88
89
90
91 serializeJson(doc, variable);
92 Serial.println(variable);
93 delay(1000);
94 }
```

Figura 1 Código de ESP32 con JSON de fecha, hora, temperatura y humedad

```
{"Fecha": "2022-08-21", "Hora": "13:20:16", "Temperatura(°C)": 28.5, "Humedad(%)": 69}  
 {"Fecha": "2022-08-21", "Hora": "13:20:17", "Temperatura(°C)": 28.5, "Humedad(%)": 69}  
 {"Fecha": "2022-08-21", "Hora": "13:20:18", "Temperatura(°C)": 28.5, "Humedad(%)": 71}  
 {"Fecha": "2022-08-21", "Hora": "13:20:19", "Temperatura(°C)": 28.5, "Humedad(%)": 71}  
 {"Fecha": "2022-08-21", "Hora": "13:20:20", "Temperatura(°C)": 28.5, "Humedad(%)": 72}  
 {"Fecha": "2022-08-21", "Hora": "13:20:21", "Temperatura(°C)": 28.5, "Humedad(%)": 72}  
 {"Fecha": "2022-08-21", "Hora": "13:20:22", "Temperatura(°C)": 28.89999962, "Humedad(%)": 72}  
 {"Fecha": "2022-08-21", "Hora": "13:20:23", "Temperatura(°C)": 28.89999962, "Humedad(%)": 72}  
 {"Fecha": "2022-08-21", "Hora": "13:20:24", "Temperatura(°C)": 28.89999962, "Humedad(%)": 73}  
 {"Fecha": "2022-08-21", "Hora": "13:20:25", "Temperatura(°C)": 28.89999962, "Humedad(%)": 73}
```

Figura 2 Visualización del Serial Monitor ESP32 de datos adquiridos

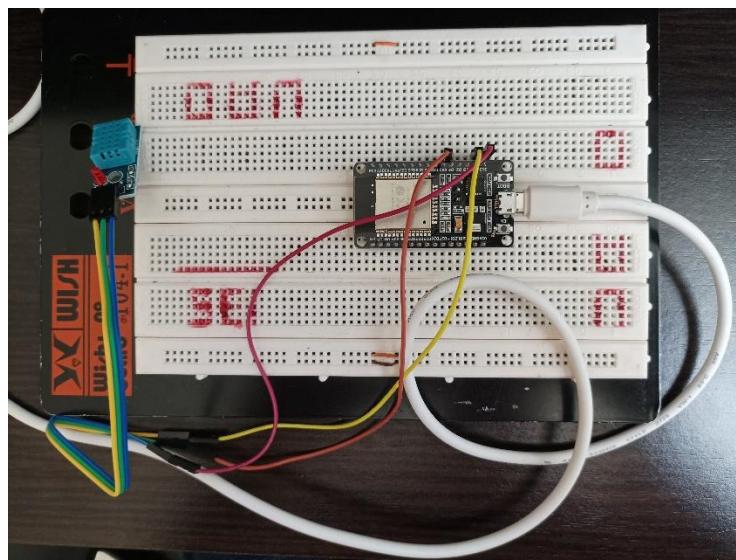
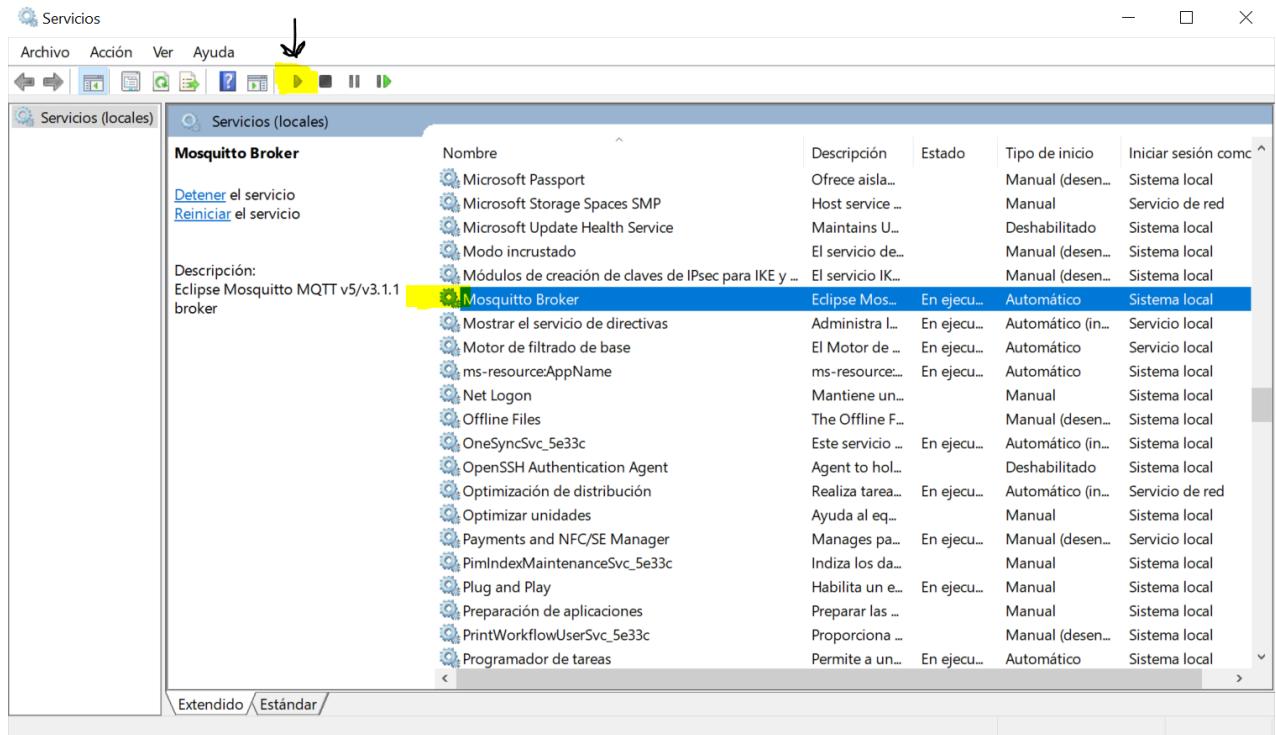


Figura 3 Montaje de ESP32 con sensor DHT11

## TRANSMISION DE DATOS MQTT

Instalación y ejecución de Mosquito : <https://mosquitto.org/files/binary/>

Es aconsejable tener la última versión



En la cmd se va a la ruta de mosquito (donde se instaló mosquito) y se realiza la prueba de creación de la suscripción y de la publicación

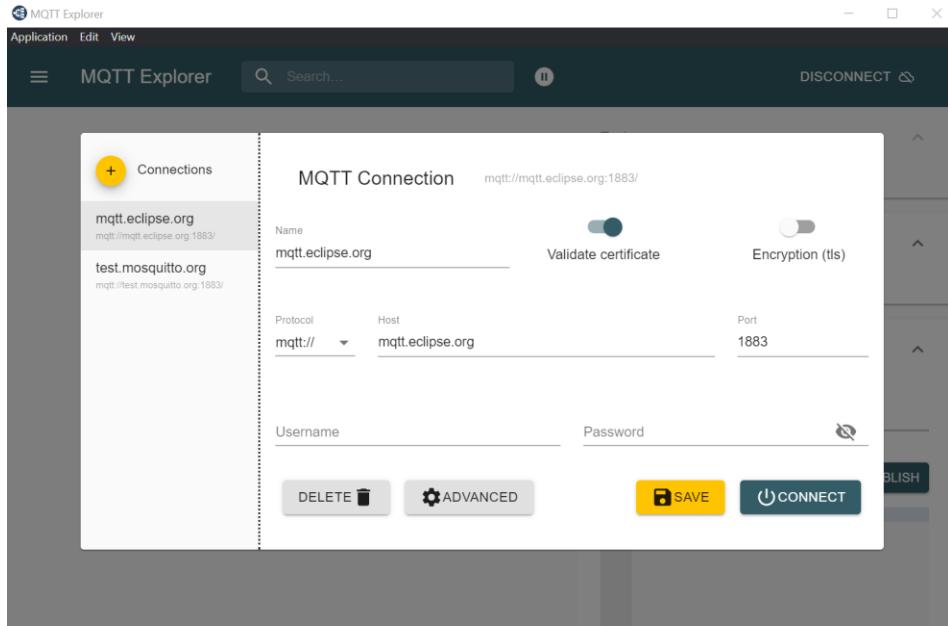
```
C:\Program Files\Mosquitto>mosquitto_pub -t prueba1 -m "hola prueba1"
```

Comando de publicación a prueba1

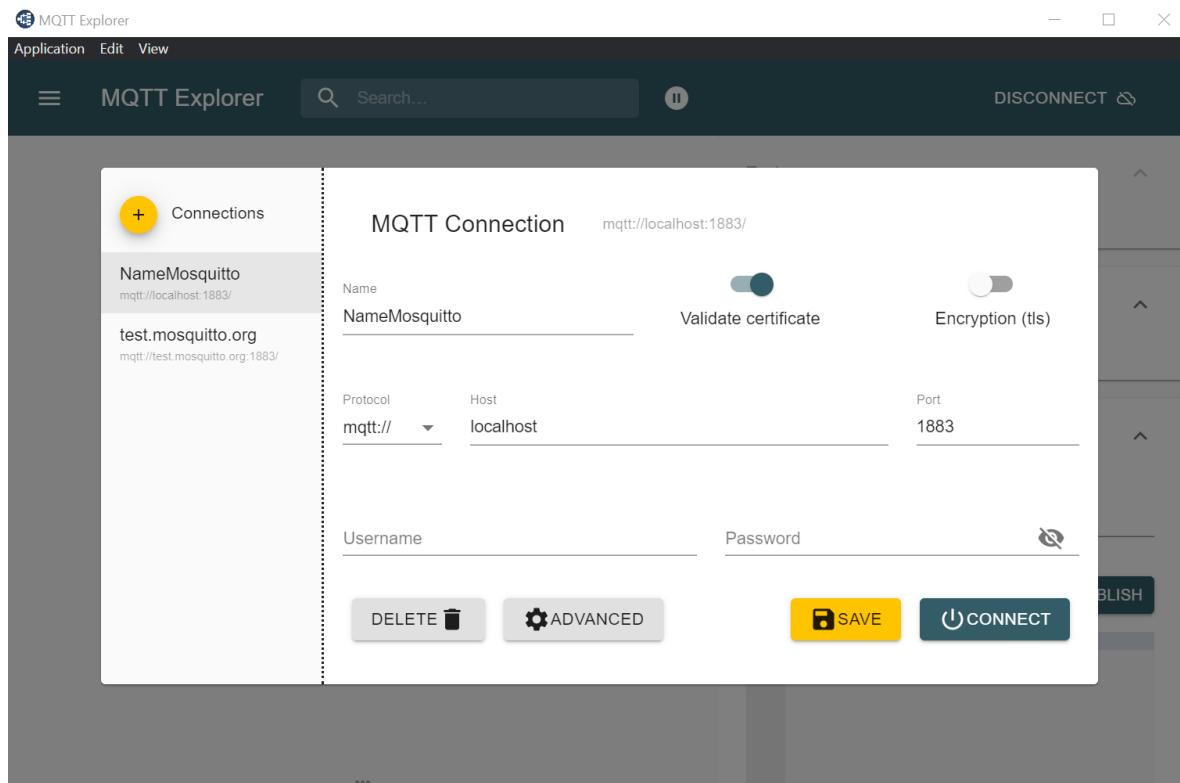
```
C:\Program Files\Mosquitto>mosquitto_sub -t prueba1
hola prueba1
```

Comando de suscripción llamado prueba 1

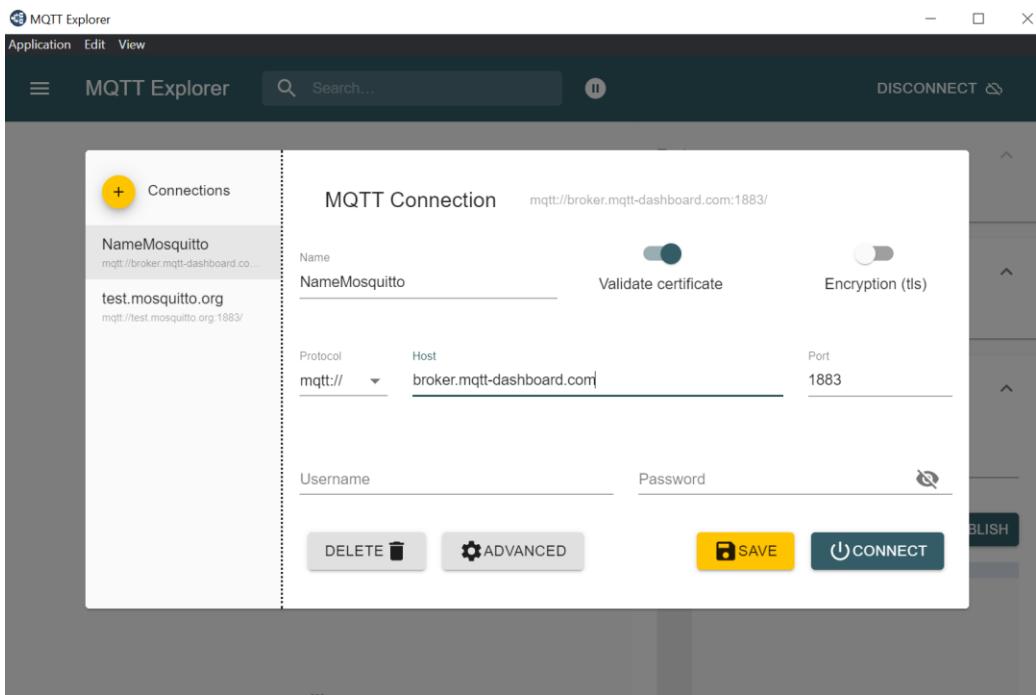
Instalación de mqtt explorer : <https://mqtt-explorer.com/>



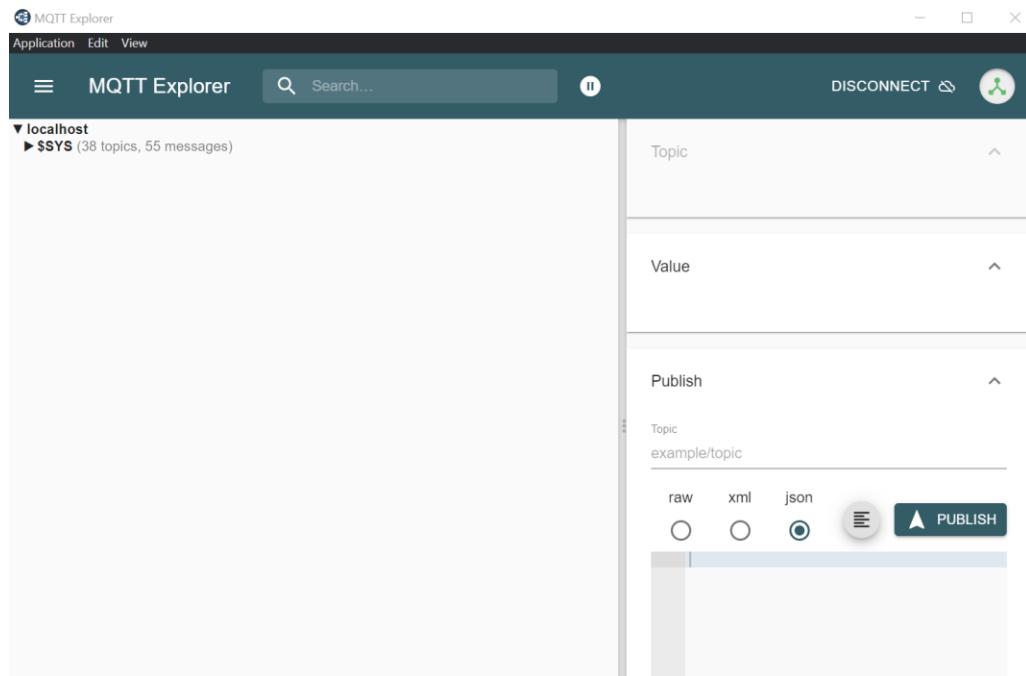
Para conectarse con el bróker mosquitto de forma local seria :



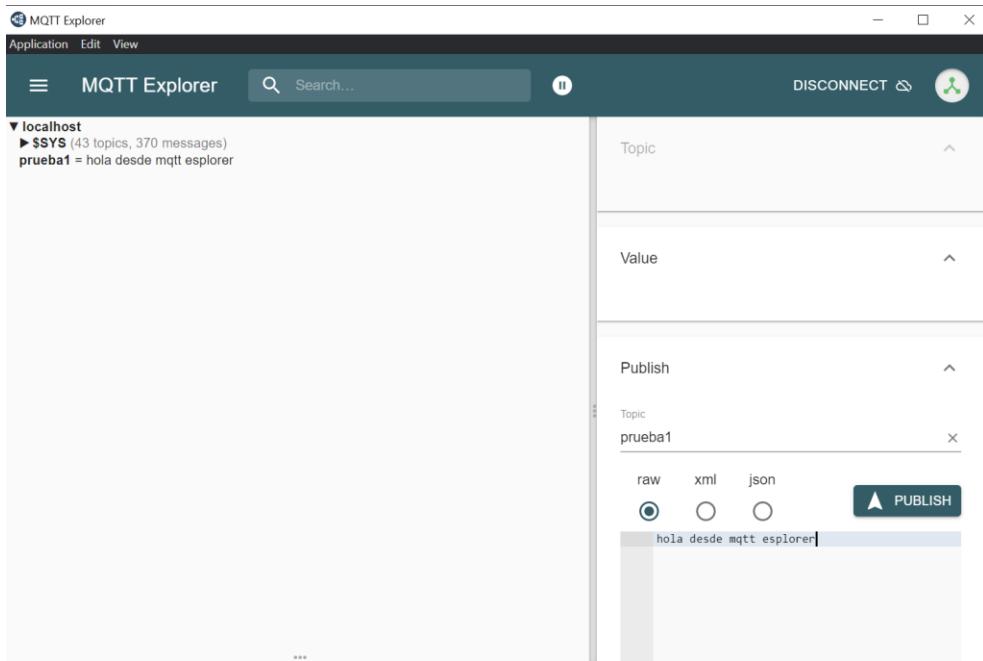
Pero si no se tiene mosquitto instalado de forma local , se puede usar uno en la nube que seria el HOST: broker.mqtt-dashboard.com



Y no se pone nada en username y password y darle en connect :



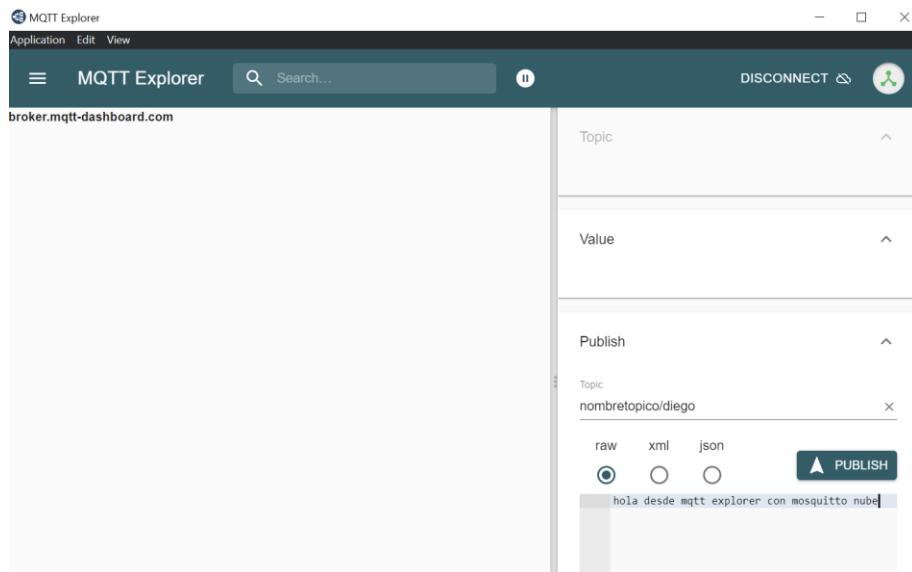
Para publicarlo seria de esta forma :



Y se visualiza que se tomo correctamente y se evidencia en el cmd :

```
C:\Program Files\Mosquitto>mosquitto_sub -t prueba1
hola prueba1
hola desde mqtt esplorer
```

Ahora probando con mosquitto en nube :



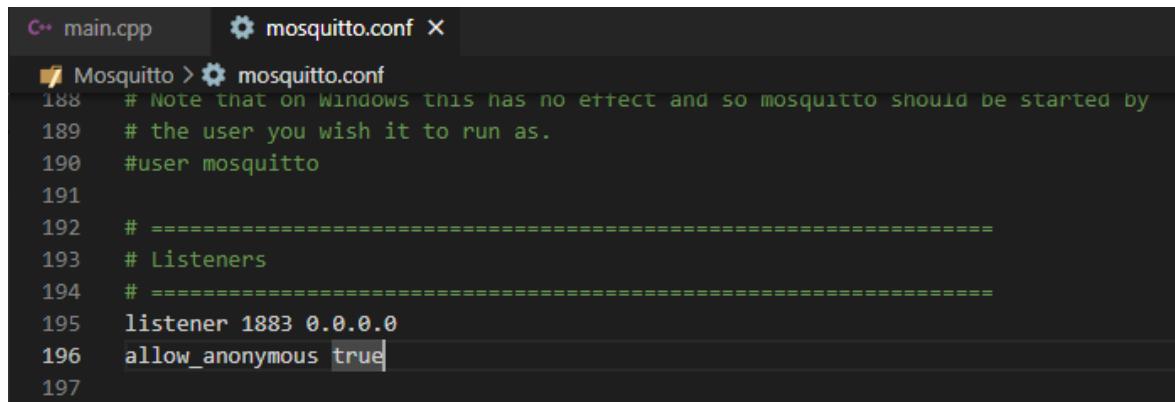
```
C:\Program Files\Mosquitto>mosquitto_sub -h broker.mqtt-dashboard.com -t nombretopico/diego
hola desde mqtt explorer con mosquitto nube
```

Para ESP32:

Si la conexión de mosquitto no se da con el esp32 hacer:

Desde la carpeta mosquito , modificar archivo mosquito.conf y en parte Listeners agregar :  
listener 1883 0.0.0.0

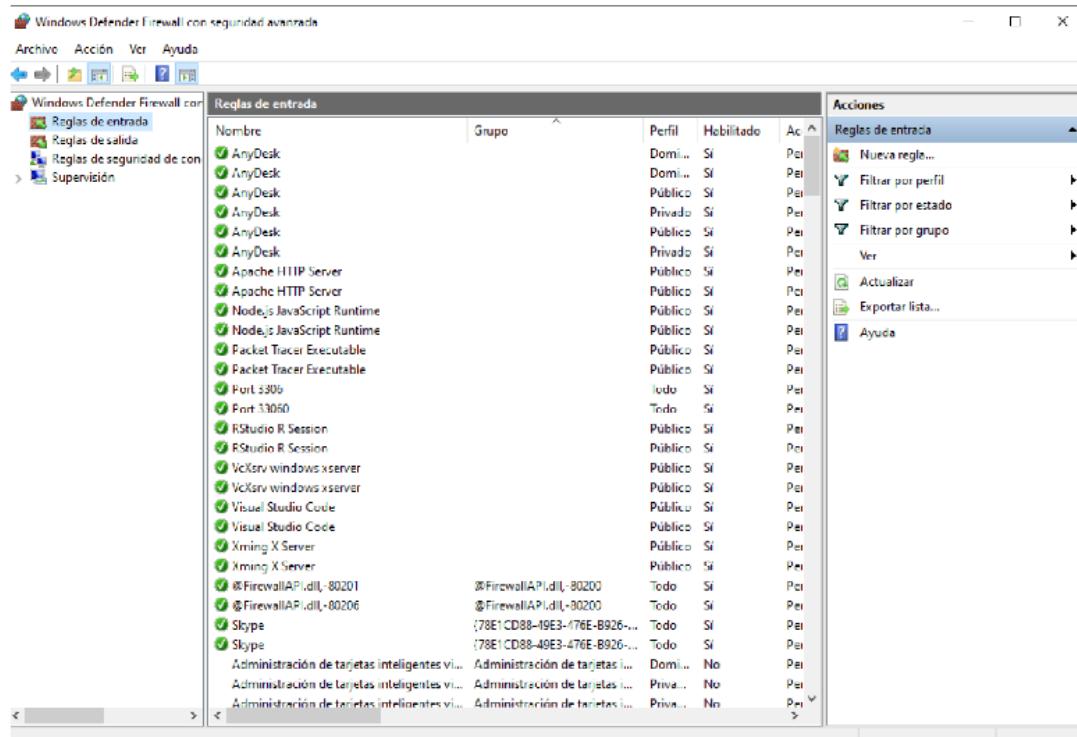
allow\_anonymous true



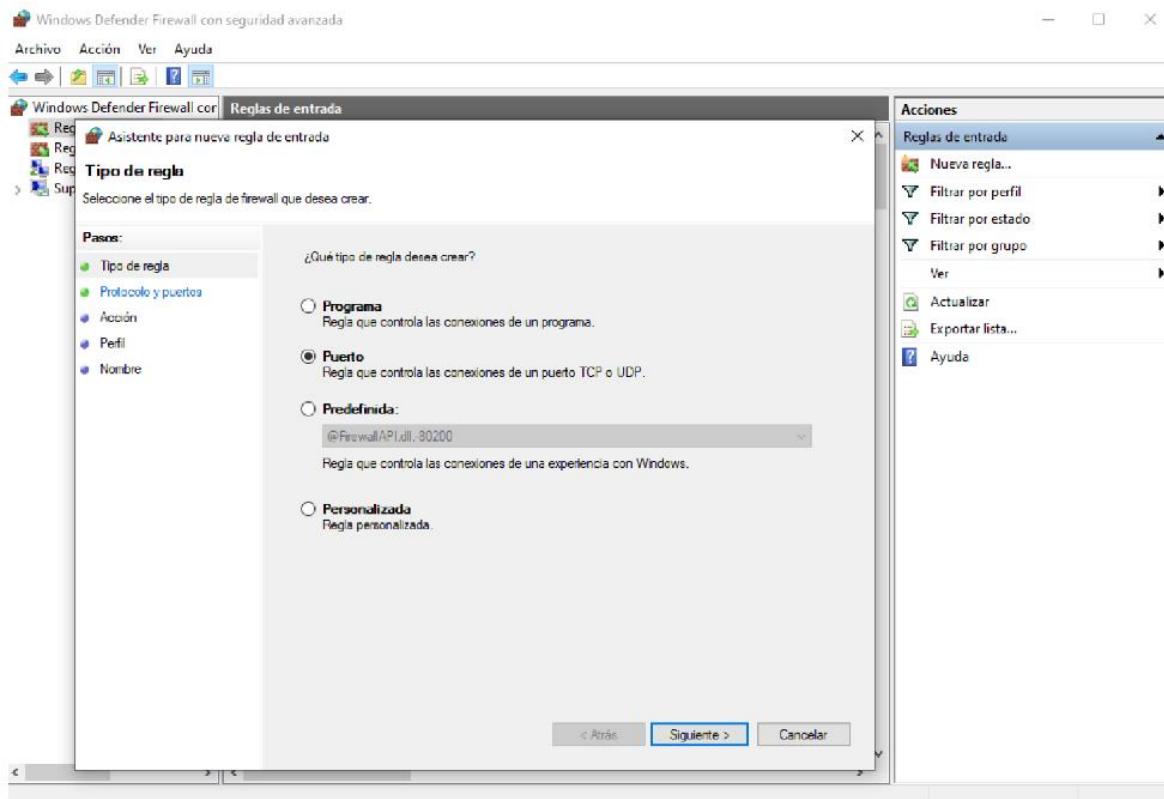
```
main.cpp      mosquito.conf

Mosquitto > mosquito.conf
188  # Note that on Windows this has no effect and so mosquitto should be started by
189  # the user you wish it to run as.
190  #user mosquitto
191
192  # =====
193  # Listeners
194  # =====
195  listener 1883 0.0.0.0
196  allow_anonymous true
197
```

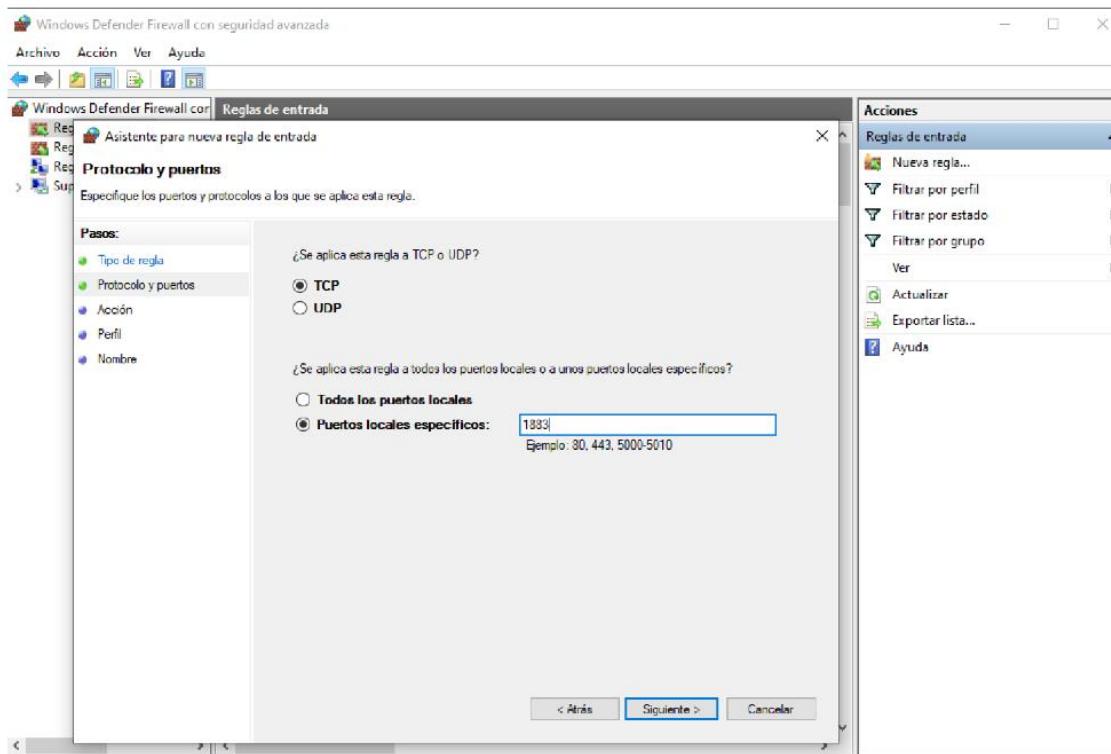
Ir a “windows defender firewall con seguridad avanzada”



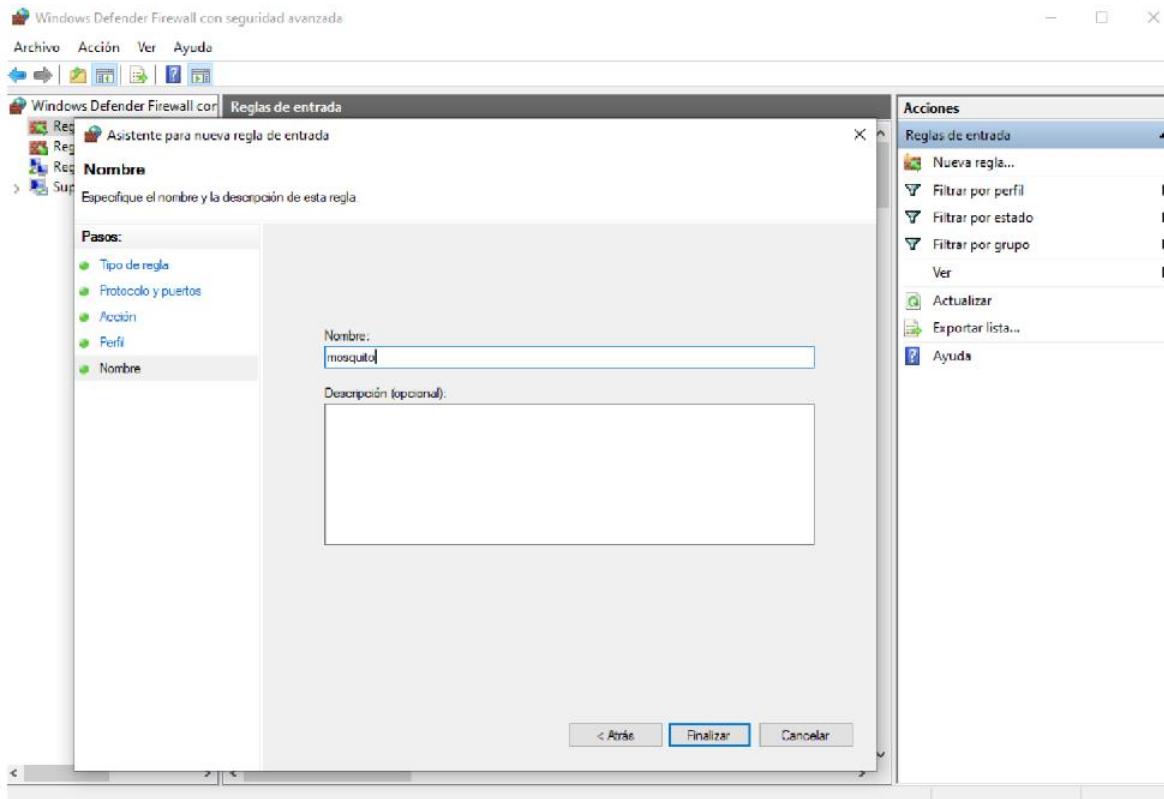
Crear nueva regla , seleccionar puerto



Seleccionar TCP e introducir el pueto “1883” en puertos locales



Dar un nombre , y darle en finalizar :



Reiniciar equipo y ejecutar código de ESP32

Cmd con suscripción de mosquito de topic “prueba1”

```
C:\Program Files\Mosquitto>mosquitto_sub -t prueba1
{"Fecha": "2023-10-28", "Hora": "22:27:58", "temperatura": 24.1000038, "humedad": 63, "idnodo": 1}
{"Fecha": "2023-10-28", "Hora": "22:27:58", "temperatura": 24.1000038, "humedad": 63, "idnodo": 1}
{"Fecha": "2023-10-28", "Hora": "22:28:09", "temperatura": 24.1000038, "humedad": 63, "idnodo": 1}
 {"Fecha": "2023-10-28", "Hora": "22:28:09", "temperatura": 24.1000038, "humedad": 63, "idnodo": 1}
 {"Fecha": "2023-10-28", "Hora": "22:28:09", "temperatura": 24.1000038, "humedad": 63, "idnodo": 1}
```

Envio de datos desde terminal Vscode:

```
Attempting MQTT connection...connected
 {"Fecha": "2023-10-28", "Hora": "22:28:09", "temperatura": 24.1000038, "humedad": 63, "idnodo": 1}
Attempting MQTT connection...failed, rc=-2 try again in 5 seconds
Attempting MQTT connection...connected
 {"Fecha": "2023-10-28", "Hora": "22:28:09", "temperatura": 24.1000038, "humedad": 63, "idnodo": 1}
Attempting MQTT connection...connected
 {"Fecha": "2023-10-28", "Hora": "22:28:09", "temperatura": 24.1000038, "humedad": 63, "idnodo": 1}
Attempting MQTT connection...connected
 {"Fecha": "2023-10-28", "Hora": "22:28:09", "temperatura": 24.1000038, "humedad": 62, "idnodo": 1}
Attempting MQTT connection...connected
 {"Fecha": "2023-10-28", "Hora": "22:28:09", "temperatura": 24.1000038, "humedad": 62, "idnodo": 1}
Attempting MQTT connection...connected
 {"Fecha": "2023-10-28", "Hora": "22:28:09", "temperatura": 24.1000038, "humedad": 62, "idnodo": 1}
```

Usando Platformio Codigo de ESP32:

```
#include <Arduino.h>

#include <ArduinoJson.h>
#include <WiFi.h>
#include <PubSubClient.h>
//LIBRERIAS PARA DHT11 (TEMPERATURA Y HUMEDAD)
#include <Adafruit_Sensor.h>
#include <DHT.h>
//LIBRERIAS PARA FECHA Y HORA
#include <WiFi.h>
#include <NTPClient.h>
#include <WiFiUdp.h>
//DEFINICION DE PINES DHT11
#define DHTPIN 14 // 4 = PIN D4
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

// Define NTP Client to get time
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP);

// Variables to save date and time
String formattedDate;
String dayStamp;
String timeStamp;

#define mqttUser ""
#define mqttPass ""
#define mqttPort 1883
const char* ssid = "**NAME_WIFI**"; //name wifi
const char* password = "*PASSWORD_WIFI*"; // clave de wifi
char mqttBroker[] = "192.168.*.*"; //ip del servidor
char mqttClientId[] = "prueba1"; //cualquier nombre
char inTopic[] = "prueba1"; //topico a suscribirse

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();
}

WiFiClient BClient;
PubSubClient client(BClient);
void reconnect() {
// Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (client.connect("", mqttUser, mqttPass)) {
      Serial.println("connected");
      // Once connected, publish an announcement...
      // Once connected, publish an announcement...
      float h= dht.readHumidity();
      float t =dht.readTemperature();

      String variable;
      StaticJsonDocument<256> doc;

      doc["Fecha"] = dayStamp;
      doc["Hora"] = timeStamp;
      doc["temperatura"] = t;
      doc["humedad"] = h;
      doc["idnodo"] = 1;

      serializeJson(doc, variable);
      int lon = variable.length()+1;
      Serial.println(variable);
      char datojson[lon];
      variable.toCharArray(datojson, lon);
      client.publish(inTopic,datojson);
      client.disconnect();
      delay(5000);
      // ... and resubscribe
      //client.subscribe("topic2");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}
```

```
}

void setup_wifi() {
    delay(10);
    // We start by connecting to a WiFi network
    Serial.println();
    Serial.print("Connecting to ");
    Serial.println(ssid);
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println("");
    Serial.println("WiFi connected");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());
    // Initialize a NTPClient to get time
    timeClient.begin();
    // Set offset time in seconds to adjust for your timezone, for example:
    // COLOMBIA -5 , entonces -5*3600 -> -18000
    timeClient.setTimeOffset(-18000); //Thailand +7 = 25200
}

void setup()
{
    Serial.begin(9600); //Serial connection
    setup_wifi(); //WiFi connection
    client.setServer(mqttBroker, mqttPort );
    client.setCallback( callback );
    Serial.println("Setup done");
    delay(1500);
}

void loop(){
    while(!timeClient.update()) {
        timeClient.forceUpdate();
    }
    // The formattedDate comes with the following format:
    // 2018-05-28T16:00:13Z
    // We need to extract date and time
    formattedDate = timeClient.getFormattedDate();
    // Extract date
    int splitT = formattedDate.indexOf("T");
```

```
dayStamp = formattedDate.substring(0, splitT);
//Serial.print("DATE: ");
//Serial.println(dayStamp);
// Extract time
timeStamp = formattedDate.substring(splitT+1, formattedDate.length()-1);
if (!client.connected()) {
  reconnect();
}
client.loop();
}
```

## RECEPCION DE DATOS CON NODE RED

Se aplica el comando npm install -g --unsafe-perm node-red en cmd :

```
C:\Users\User>npm install -g --unsafe-perm node-red

added 302 packages in 29s

45 packages are looking for funding
  run `npm fund` for details
npm notice
npm notice New major version of npm available! 9.5.0 -> 10.2.1
npm notice Changelog: https://github.com/npm/cli/releases/tag/v10.2.1
npm notice Run npm install -g npm@10.2.1 to update!
npm notice
```

Se abre con el comando node-red :

```
C:\Users\User>node-red
29 Oct 15:45:33 - [info]

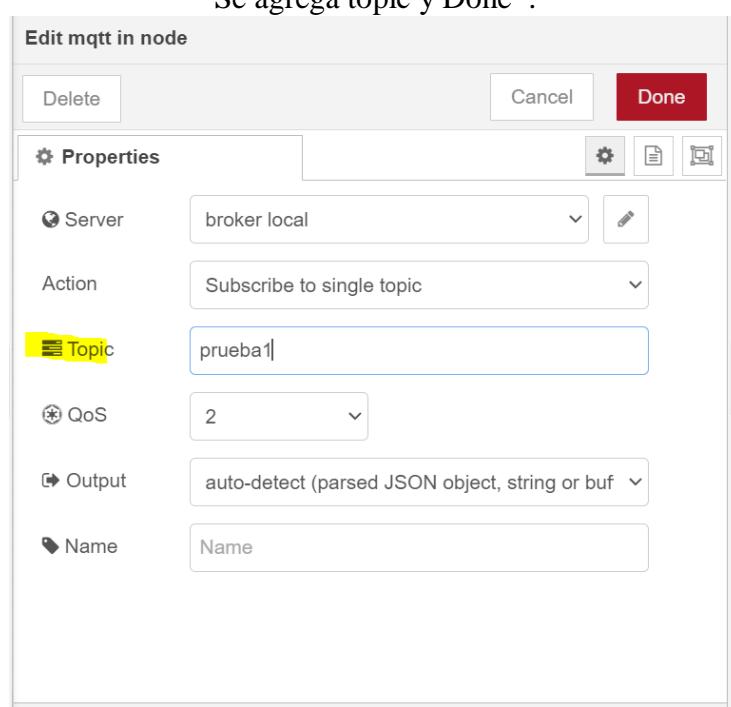
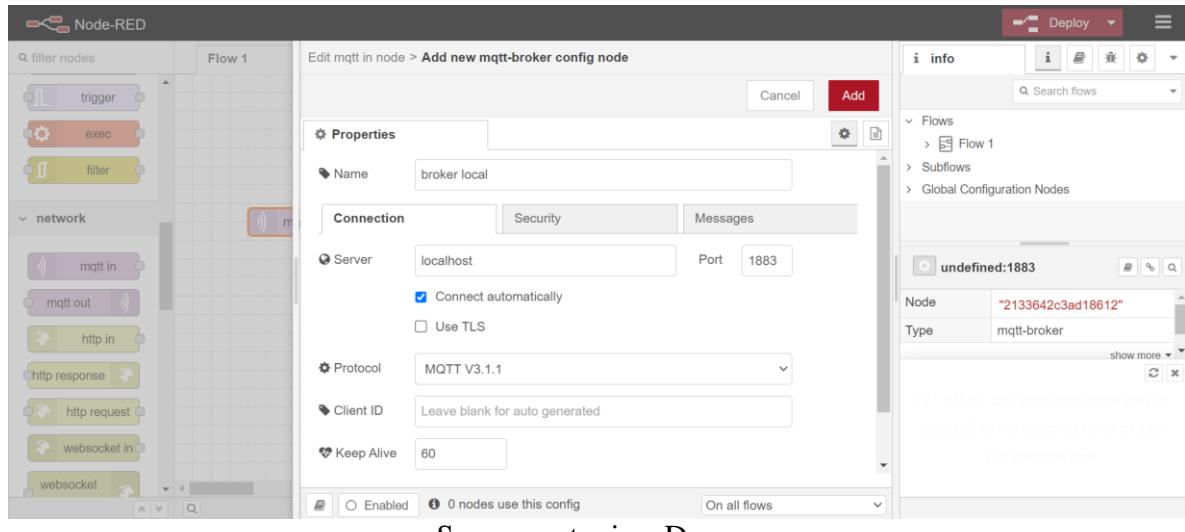
Welcome to Node-RED
=====
29 Oct 15:45:33 - [info] Node-RED version: v3.1.0
29 Oct 15:45:33 - [info] Node.js version: v18.15.0
29 Oct 15:45:33 - [info] Windows_NT 10.0.19045 x64 LE
29 Oct 15:45:36 - [info] Loading palette nodes
29 Oct 15:45:38 - [info] Settings file : C:\Users\User\.node-red\settings.js
29 Oct 15:45:38 - [info] Context store : 'default' [module=memory]
29 Oct 15:45:38 - [info] User directory : C:\Users\User\.node-red
29 Oct 15:45:38 - [warn] Projects disabled : editorTheme.projects.enabled=false
29 Oct 15:45:38 - [info] Flows file : C:\Users\User\.node-red\flows.json
29 Oct 15:45:38 - [info] Creating new flow file
29 Oct 15:45:38 - [warn]
```

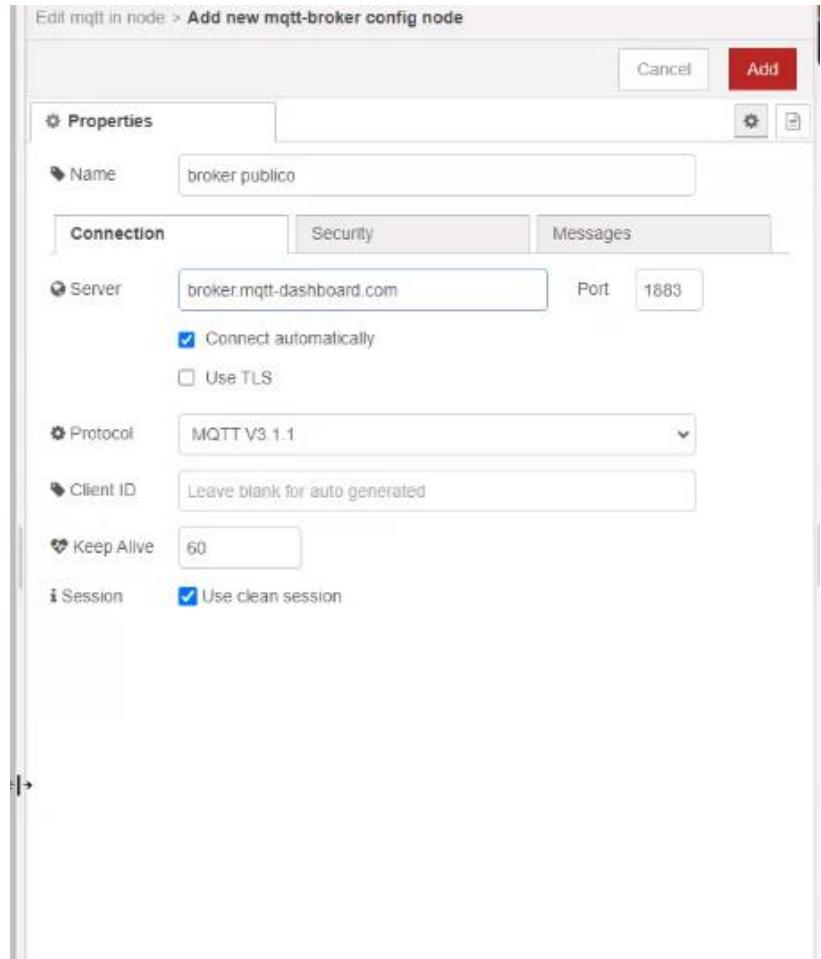
Y dndando la dirección de localhost:1880 se visualiza node red:

The screenshots illustrate the configuration of an MQTT node in Node-RED:

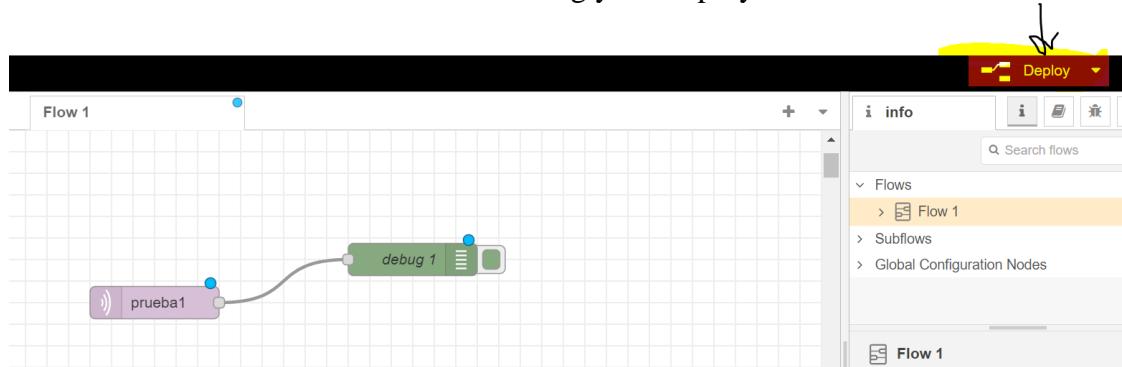
- Screenshot 1:** Shows the Node-RED interface with a flow editor containing a "trigger" node, an "exec" node, a "filter" node, and an "mqtt" node. The "mqtt" node is highlighted with a yellow box. The properties panel shows the "Topic" field is set to "Topic".
- Screenshot 2:** Shows the "Edit mqtt in node" dialog. The "Properties" tab is selected, showing fields for "Server" (set to "Add new mqtt-broker..."), "Action" (set to "Subscribe to single topic"), "Topic" (set to "Topic"), "QoS" (set to 2), "Output" (set to "auto-detect (parsed JSON object, string or buf)"), and "Name" (empty). A yellow box highlights the "Server" dropdown.
- Screenshot 3:** Shows the "Edit mqtt in node > Add new mqtt-broker config node" dialog. The "Properties" tab is selected, showing fields for "Name" (set to "Name"), "Connection" (server set to "e.g. localhost", port set to 1883, "Connect automatically" checked, "Use TLS" unchecked), "Protocol" (set to "MQTT V3.1.1"), "Client ID" (set to "Leave blank for auto generated"), and "Keep Alive" (set to 60). A yellow box highlights the "Server" input field.

Si se usa local se pone y se da Add:

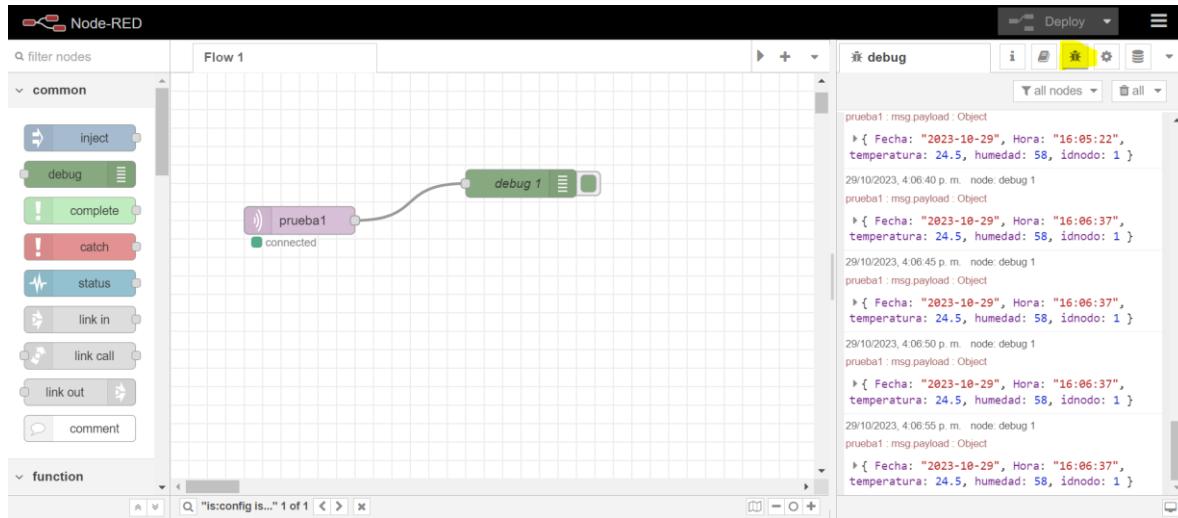




Se conecta debug y dar deploy :



Dar en debug y ejecutar código de ESP32



### Código ESP32 Platformio:

```
#include <Arduino.h>

#include <ArduinoJson.h>
#include <WiFi.h>
#include <PubSubClient.h>
//LIBRERIAS PARA DHT11 (TEMPERATURA Y HUMEDAD)
#include <Adafruit_Sensor.h>
#include <DHT.h>
//LIBRERIAS PARA FECHA Y HORA
#include <WiFi.h>
#include <NTPClient.h>
#include <WiFiUdp.h>
//DEFINICION DE PINES DHT11
#define DHTPIN 14 // 4 = PIN D4
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

// Define NTP Client to get time
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP);

// Variables to save date and time
String formattedDate;
String dayStamp;
String timeStamp;

#define mqttUser ""
#define mqttPass ""
#define mqttPort 1883
```

```

const char* ssid = "NAME_WIFI"; //name wifi
const char* password = "PASSWORD_WIFI"; // clave de wifi
char mqttBroker[] = "192.168.*.*"; //ip del servidor

char mqttClientId[] = "prueba1"; //cualquier nombre
char inTopic[] = "prueba1"; //topico a suscribirse

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();
}

WiFiClient BClient;
PubSubClient client(BClient);
void reconnect() {
// Loop until we're reconnected
    while (!client.connected()) {
        Serial.print("Attempting MQTT connection...");
        // Attempt to connect
        if (client.connect("", mqttUser, mqttPass)) {
            Serial.println("connected");
            // Once connected, publish an announcement...
            // Once connected, publish an announcement...
            float h= dht.readHumidity();
            float t =dht.readTemperature();

            String variable;
            StaticJsonDocument<256> doc;

            doc["Fecha"] = dayStamp;
            doc["Hora"] = timeStamp;
            doc["temperatura"] = t;
            doc["humedad"] = h;
            doc["idnodo"] = 1;

            serializeJson(doc, variable);
            int lon = variable.length()+1;
            Serial.println(variable);
        }
    }
}

```

```

char datojson[lon];
variable.toCharArray(datojson, lon);
client.publish(inTopic,datojson);
client.disconnect();
delay(5000);
// ... and resubscribe
//client.subscribe("topic2");
} else {
Serial.print("failed, rc=");
Serial.print(client.state());
Serial.println(" try again in 5 seconds");
// Wait 5 seconds before retrying
delay(5000);
}
}
}

void setup_wifi() {
delay(10);
// We start by connecting to a WiFi network
Serial.println();
Serial.print("Connecting to ");
Serial.println(ssid);
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
delay(500);
Serial.print(".");
}
Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
// Initialize a NTPClient to get time
timeClient.begin();
// Set offset time in seconds to adjust for your timezone, for example:
// COLOMBIA -5 , entonces -5*3600 -> -18000
timeClient.setTimeOffset(-18000); //Thailand +7 = 25200
}

void setup()
{
Serial.begin(9600); //Serial connection
setup_wifi(); //WiFi connection
client.setServer(mqttBroker, mqttPort );
client.setCallback( callback );

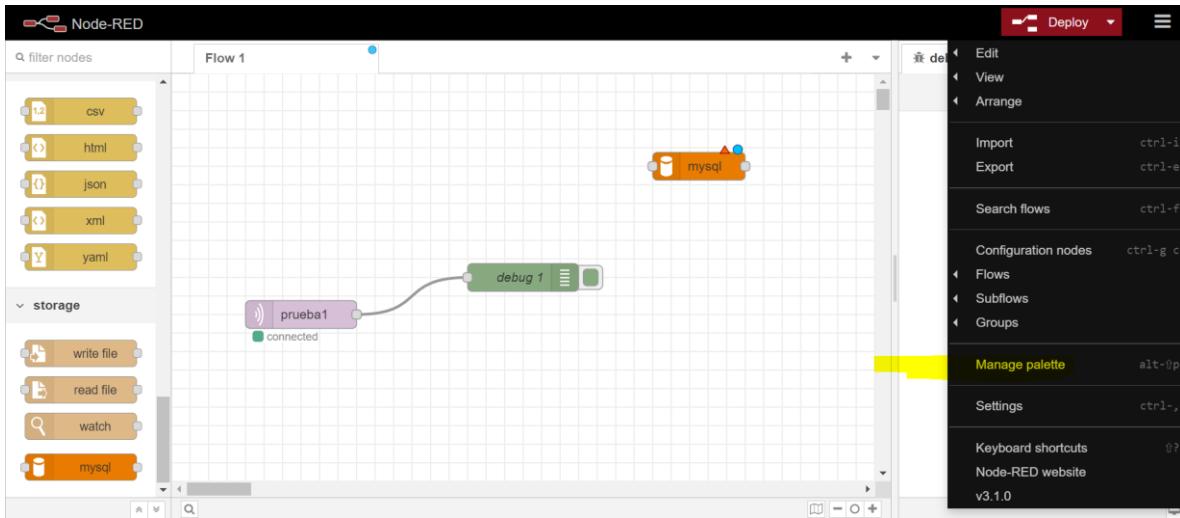
```

```
Serial.println("Setup done");
delay(1500);
}

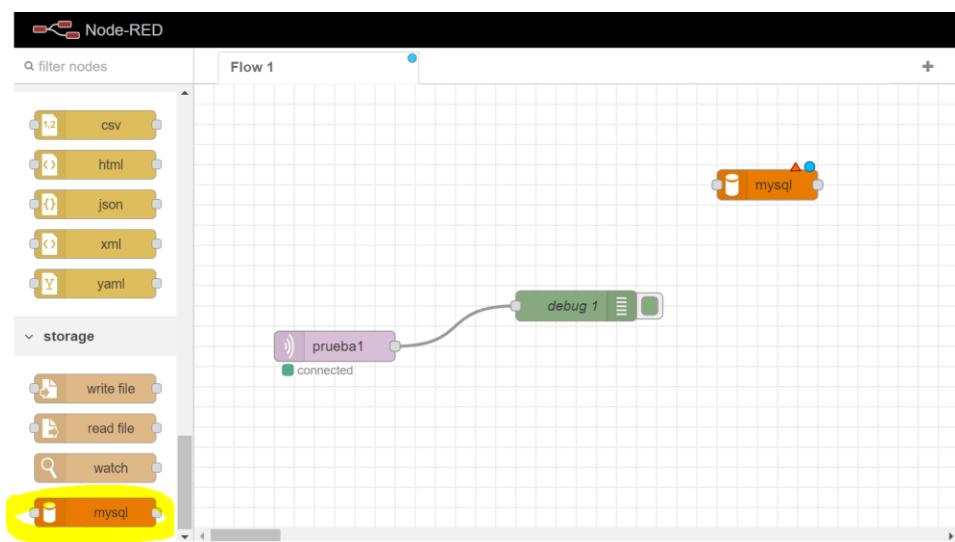
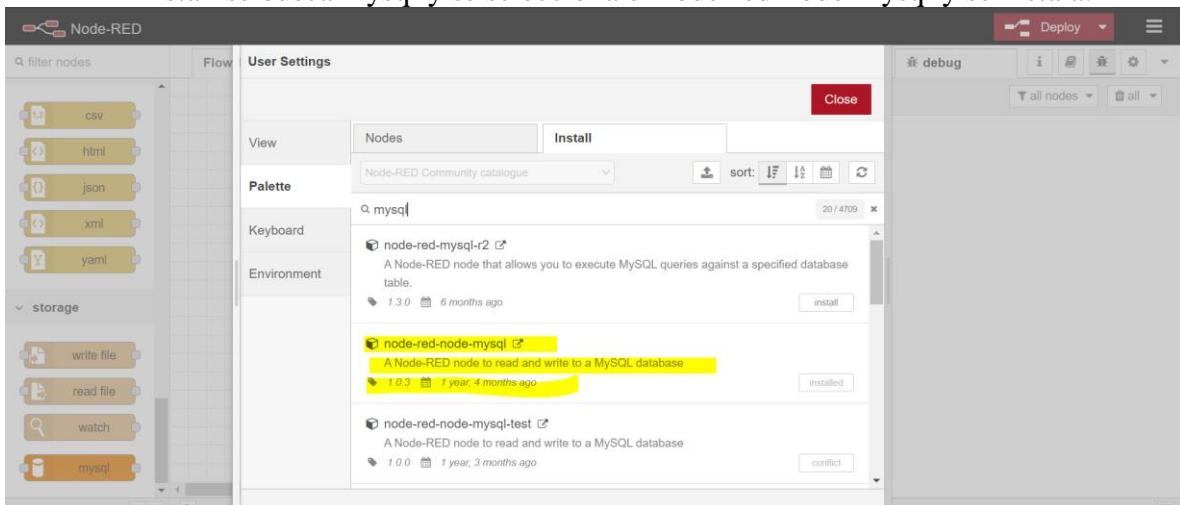
void loop(){
    while(!timeClient.update()) {
        timeClient.forceUpdate();
    }
    // The formattedDate comes with the following format:
    // 2018-05-28T16:00:13Z
    // We need to extract date and time
    formattedDate = timeClient.getFormattedDate();
    // Extract date
    int splitT = formattedDate.indexOf("T");
    dayStamp = formattedDate.substring(0, splitT);
    //Serial.print("DATE: ");
    //Serial.println(dayStamp);
    // Extract time
    timeStamp = formattedDate.substring(splitT+1, formattedDate.length()-1);
    if (!client.connected()) {
        reconnect();
    }
    client.loop();
}
```

## ALMACENAMIENTO Y PROCESAMIENTO DE DATOS

Para usar mysql en node red dar en el menú en Manage palette :



En install se busca mysql y se selecciona el node-red-node-mysql y se instala:



Se clickea mysql y se edita:

**Edit mysql node**

Delete Cancel Done

**Properties**

Database Add new MySQLdatabase...

Name

Enabled

Edit mysql node > **Add new MySQLdatabase config node**

Cancel Add

**Properties**

Host 127.0.0.1

Port 3306

User

Password

Database

Timezone ±hh:mm

Charset UTF8

Name

Enabled 0 nodes use this config

Se crea la base de datos , en este caso sea Docker para crear un contenedor de mysql, en el cual el nombre del contenedor es mymysql , el usuario es root y se le da una contraseña :

```
docker run --name mymysql -e MYSQL_ROOT_PASSWORD=mypassword -p 3306:3306  
-d mysql:latest
```

Se ingresa al contenedor mymysql:  
docker exec -it mymysql bash

Se ingresa a mysql :  
mysql -u root -p

Se pone la contraseña y se crea la base de datos llamada “iaiot” :

```
bash-4.4# mysql -u root -p  
Enter password:  
Welcome to the MySQL monitor. Commands end with ; or \g.  
Your MySQL connection id is 9  
Server version: 8.0.32 MySQL Community Server - GPL  
  
Copyright (c) 2000, 2023, Oracle and/or its affiliates.  
  
Oracle is a registered trademark of Oracle Corporation and/or its  
affiliates. Other names may be trademarks of their respective  
owners.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
  
mysql> create database iaiot;  
Query OK, 1 row affected (0.02 sec)
```

Se crea la tabla “datos” y sus parámetros :

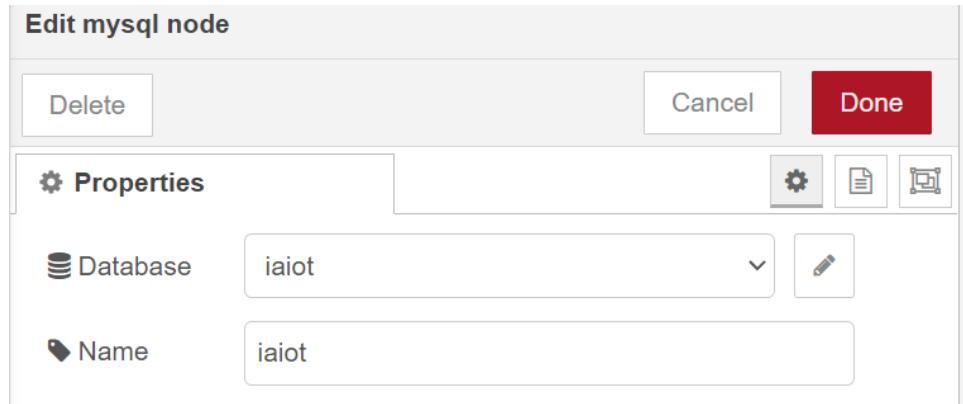
```
mysql> use iaiot;  
Database changed  
mysql> create table datos (   
    -> id int auto_increment,  
    -> idnodo int,  
    -> temperatura float,  
    -> humedad float,  
    -> fecha datetime default now(),  
    -> primary key(id));  
Query OK, 0 rows affected (0.09 sec)
```

```
mysql> desc datos;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id    | int  | NO   | PRI | NULL    | auto_increment |
| idnodo | int  | YES  |     | NULL    |             |
| temperatura | float | YES  |     | NULL    |             |
| humedad | float | YES  |     | NULL    |             |
| fecha  | datetime | YES  |     | CURRENT_TIMESTAMP | DEFAULT_GENERATED |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.02 sec)
```

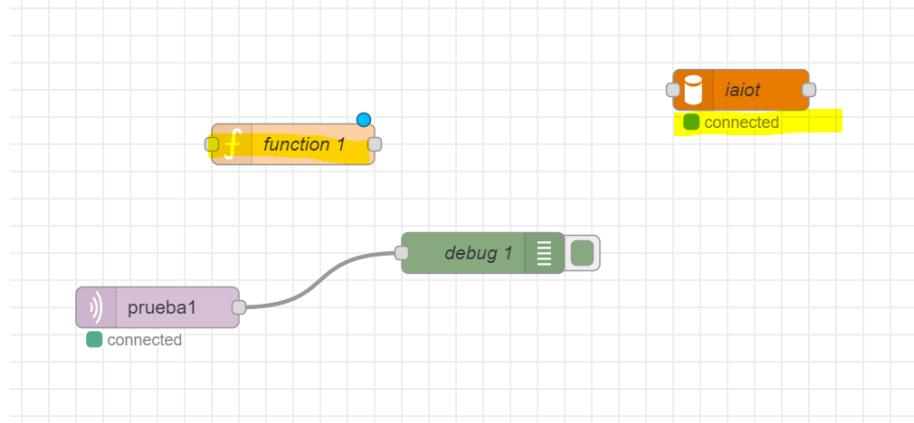
Para node red con mysql :

Edit mysql node > **Edit MySQLdatabase node**

<input type="button" value="Delete"/>	<input type="button" value="Cancel"/>	<input style="background-color: #e63333; color: white; font-weight: bold; padding: 2px 5px; border-radius: 5px; border: none;" type="button" value="Update"/>
<span style="font-size: 1em; font-weight: bold;">Properties</span> <span style="float: right; margin-top: -1em;"> <input style="border: none; border-radius: 50%; width: 1em; height: 1em; background-color: #ccc; color: #ccc; vertical-align: middle;" type="button" value="gear"/> <input style="border: none; width: 1em; height: 1em; background-color: #ccc; color: #ccc; vertical-align: middle;" type="button" value="file"/> </span>		
<span style="color: #ccc;">Host</span>	localhost	
<span style="color: #ccc;">Port</span>	3306	
<span style="color: #ccc;">User</span>	root	
<span style="color: #ccc;">Password</span>	.....	
<span style="color: #ccc;">Database</span>	iaiot	
<span style="color: #ccc;">Timezone</span>	±hh:mm	
<span style="color: #ccc;">Charset</span>	UTF8	
<span style="color: #ccc;">Name</span>	Name	



Se pone desplay para observar si se conectó , y se agrega la función :



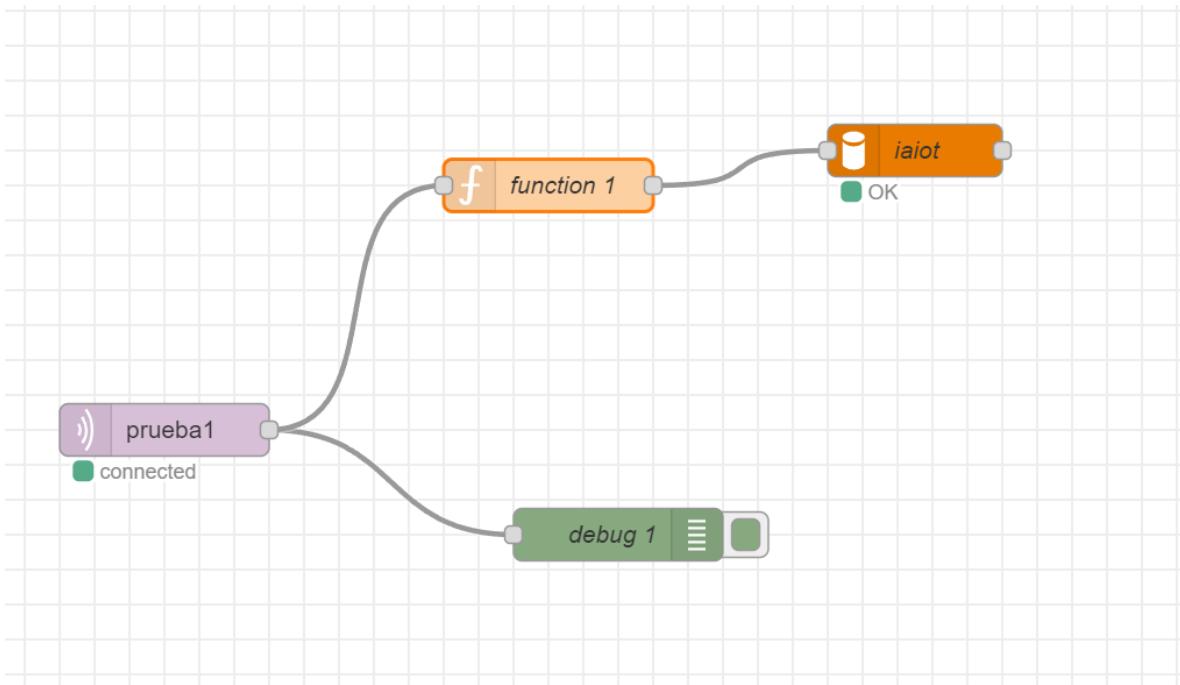
Se edita la función :

El código es el siguiente:

```
var json = msg.payload;
msg.topic = "insert into datos values (null," + json.idnodo + "," +
json.temperatura + "," + json.humedad +
",now())";

return msg;
```

Se conecta la función



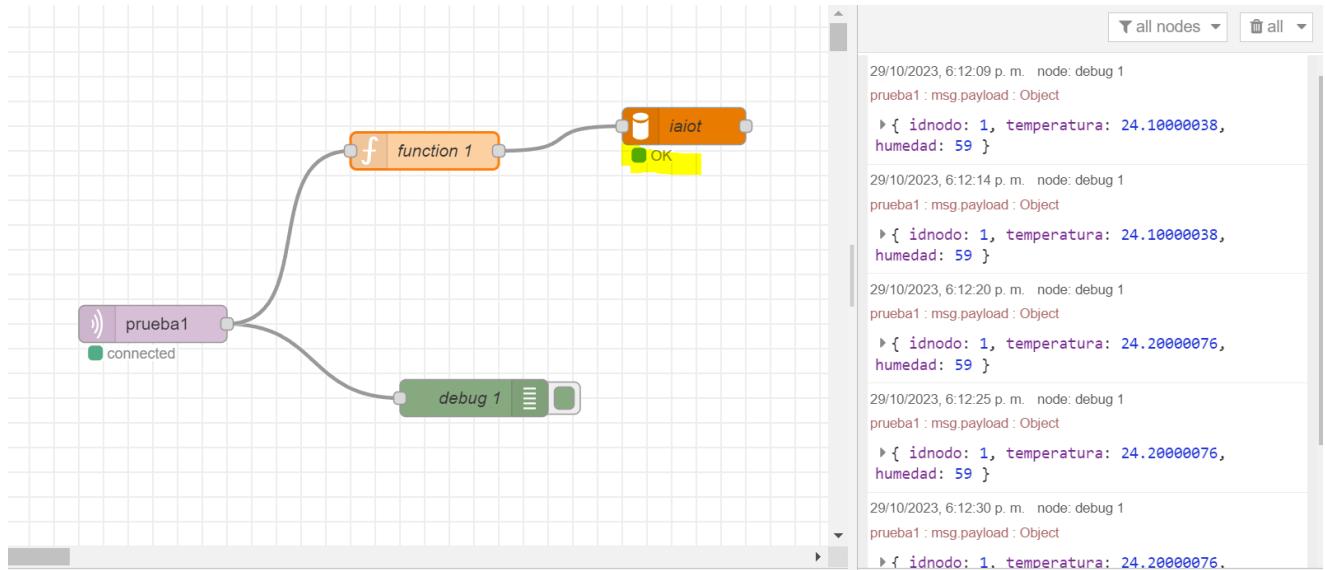
Se ejecuta código ESP32:

```

Attempting MQTT connection...connected
{"idnodo":1,"temperatura":24.10000038,"humedad":59}
Attempting MQTT connection...connected
{"idnodo":1,"temperatura":24.10000038,"humedad":59}
Attempting MQTT connection...connected
{"idnodo":1,"temperatura":24.20000076,"humedad":59}
Attempting MQTT connection...connected
 {"idnodo":1,"temperatura":24.20000076,"humedad":59}
Attempting MQTT connection...connected
 {"idnodo":1,"temperatura":24.20000076,"humedad":59}
Attempting MQTT connection...connected
 {"idnodo":1,"temperatura":24.29999924,"humedad":59}

```

Y se puede visualizar el buen funcionamiento en Node-RED dando en la base de datos un OK :



Se visualiza en la base de datos , que fue exitosamente agregado:

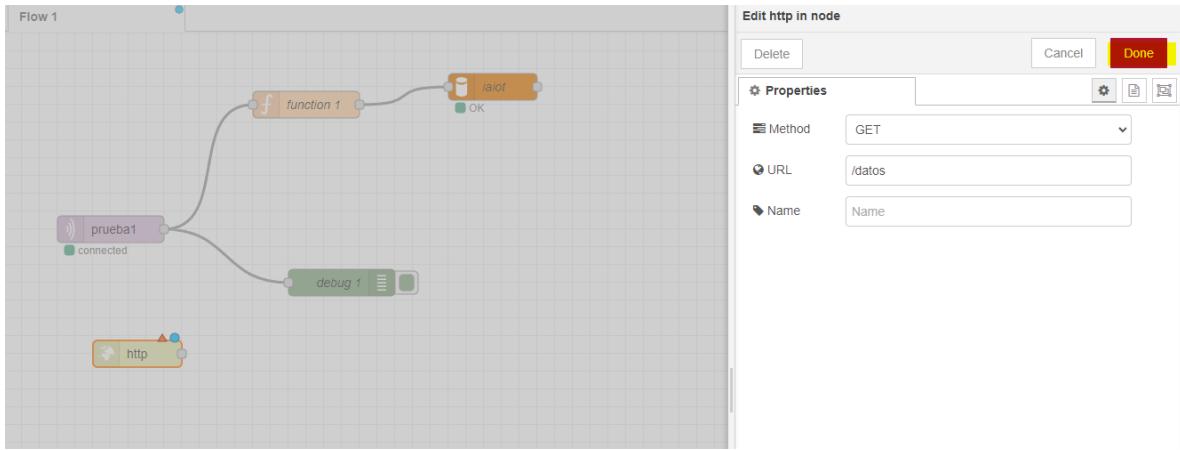
```

mysql> select * from datos;
+----+----+-----+-----+-----+
| id | idnodo | temperatura | humedad | fecha      |
+----+----+-----+-----+-----+
| 1  | 1       | 24.1        | 59      | 2023-10-29 23:12:09 |
| 2  | 1       | 24.1        | 59      | 2023-10-29 23:12:14 |
| 3  | 1       | 24.2        | 59      | 2023-10-29 23:12:20 |
| 4  | 1       | 24.2        | 59      | 2023-10-29 23:12:25 |
| 5  | 1       | 24.2        | 59      | 2023-10-29 23:12:30 |
| 6  | 1       | 24.3        | 59      | 2023-10-29 23:12:36 |
+----+----+-----+-----+-----+
6 rows in set (0.00 sec)

```

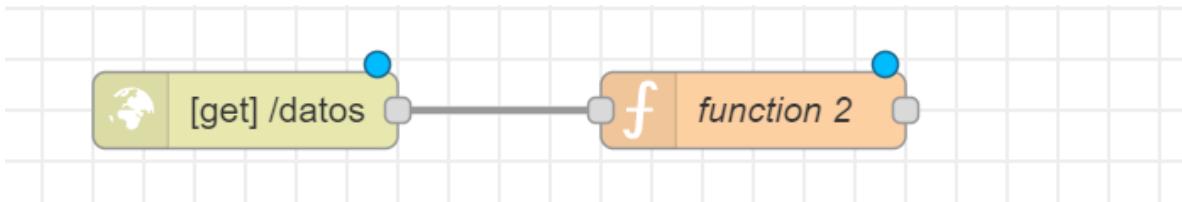
## Usando API REST en Node-RED

Se agrega el bloque ‘http in’ y dar ruta “/datos” :

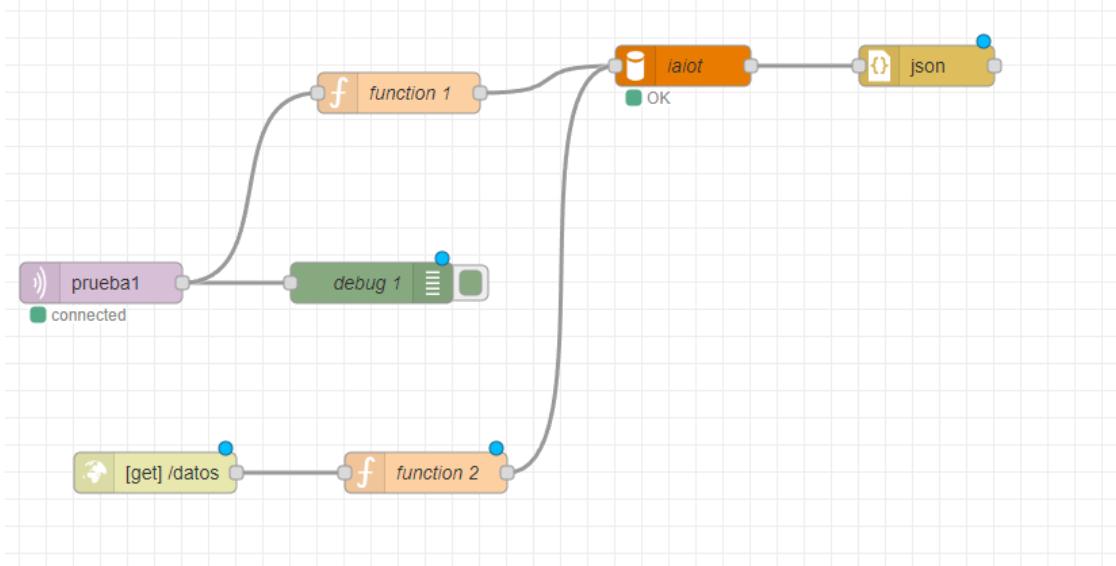


Se agrega una función para esa ruta :  
Se da con el siguiente código debido a GET :

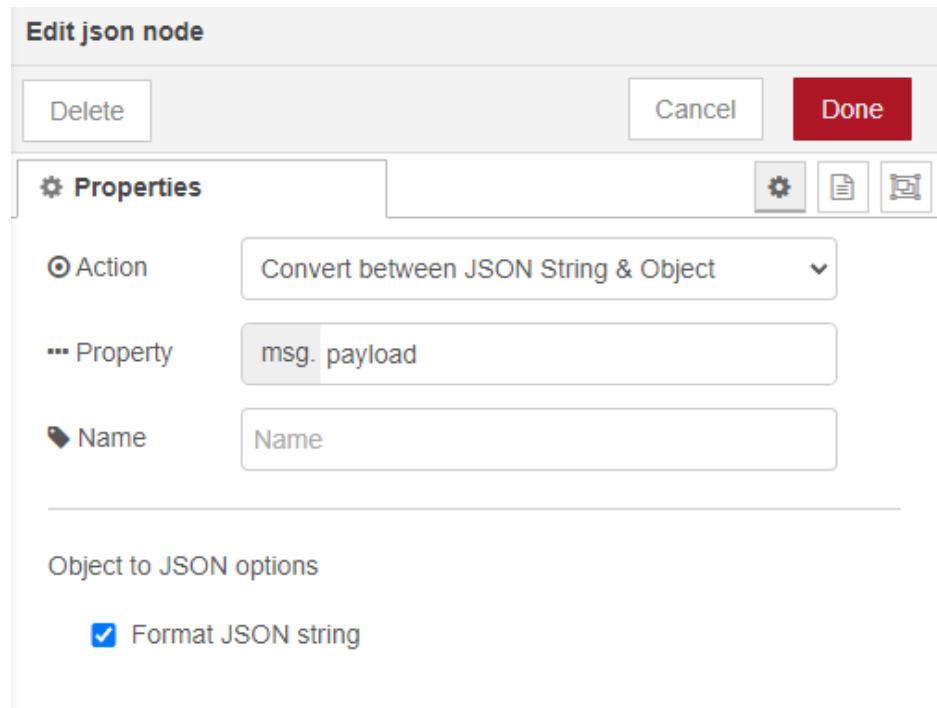
```
msg.topic = "select * from datos";
return msg;
```



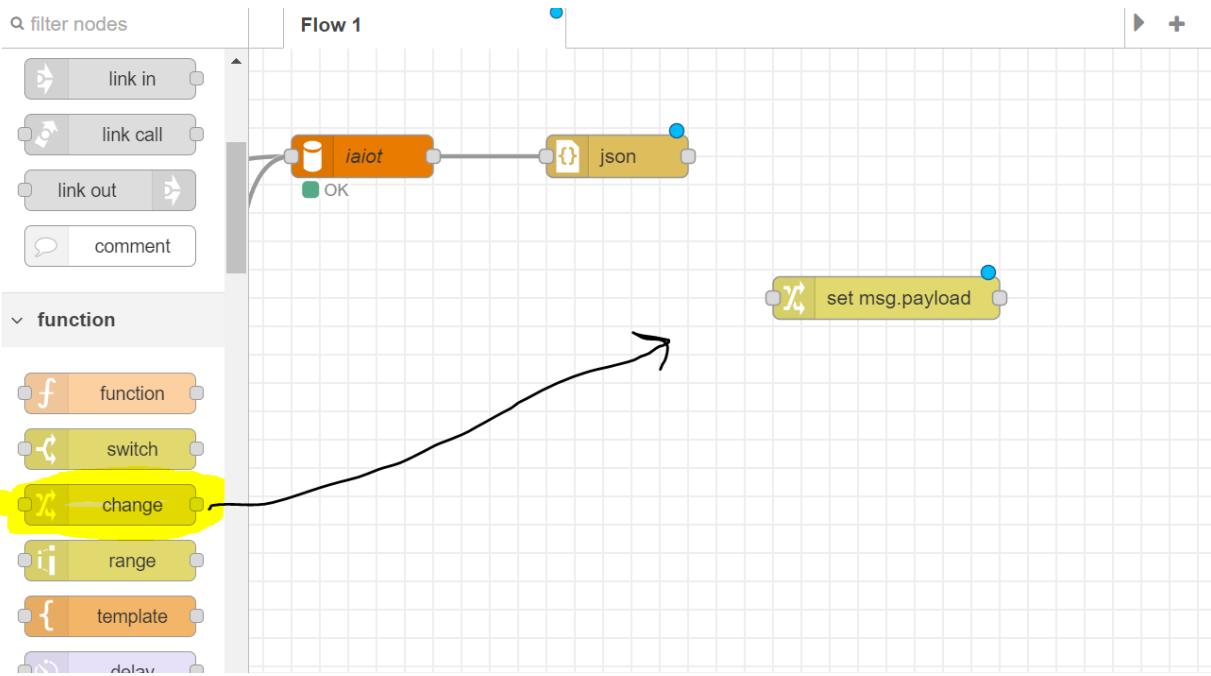
Conectamos de forma exitosa los bloques:

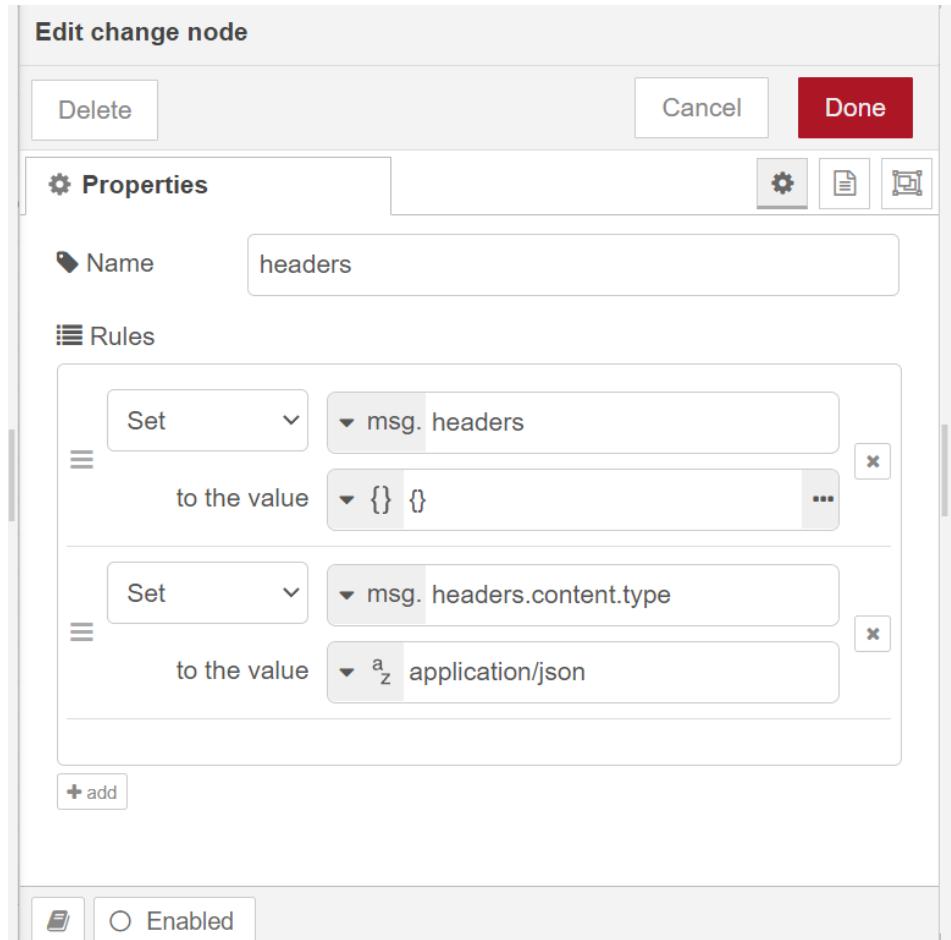


Editamos el JSON para sea formateado a json la respuesta:

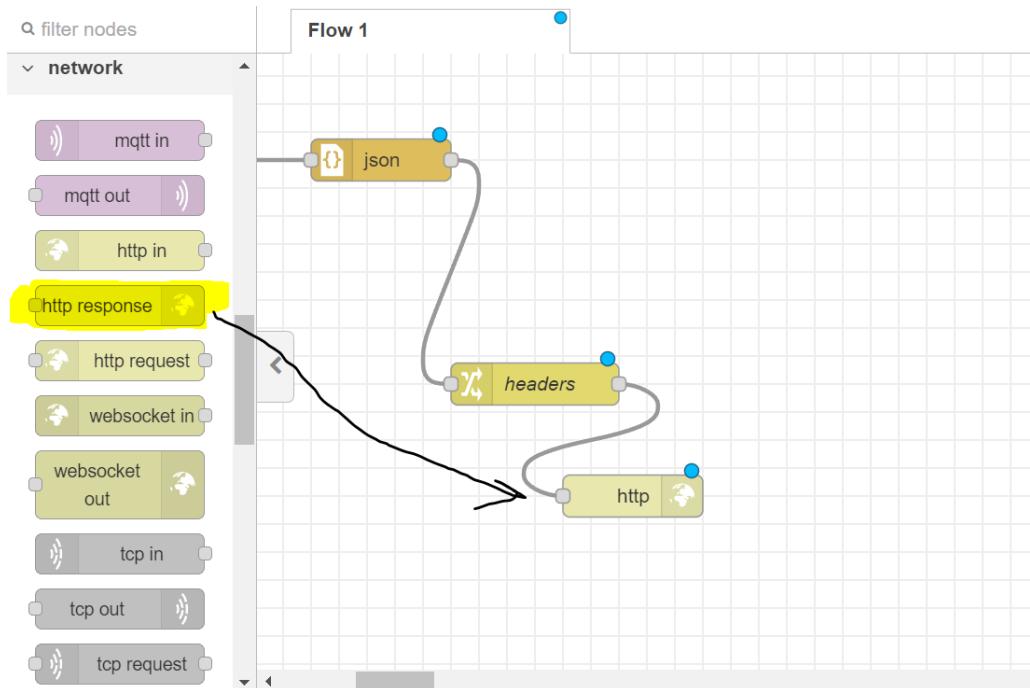


Se agrega “change” para la respuesta http y se edita ‘change’ :

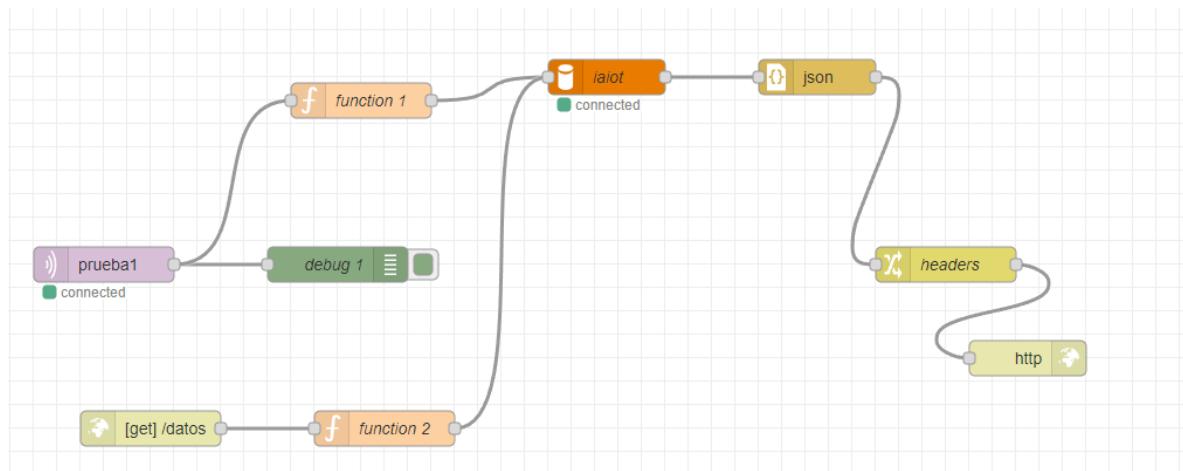
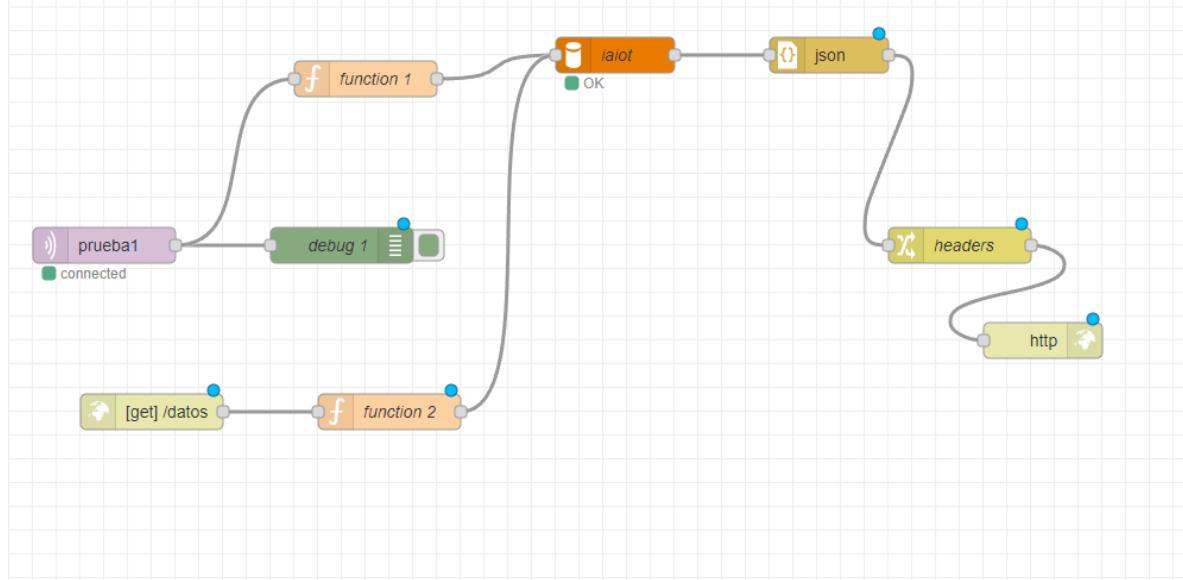




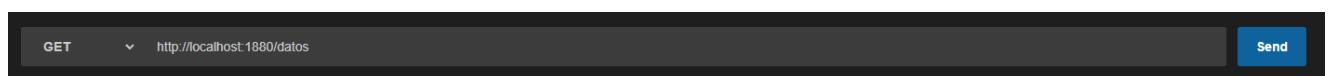
Se agrega por ultimo el http response :



El resultado quedaría así y se le da deploy :



Usando request (ejemplo Postman u otros) en la url con GET :  
<http://localhost:1880/datos>



Body Headers JSON Status: 200 OK Duration: 80 ms

```

1   [
2     {
3       "id": 1,
4       "idnodo": 1,
5       "temperatura": 24.1,
6       "humedad": 59,
7       "fecha": "2023-10-30T04:12:00.000Z"

```

Body Headers JSON Status: 200 OK Duration: 80 ms

```

1   [
2     [
3       {
4         "id": 1,
5         "idnodo": 1,
6         "temperatura": 24.1,
7         "humedad": 59,
8         "fecha": "2023-10-30T04:12:00.000Z"
9       },
10      {
11        "id": 2,
12        "idnodo": 1,

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS + ⌂ docker.html\_email.bson PlatformIO: Upload... PlatformIO: Monit...

id	idnodo	temperatura	humedad	fecha
1	1	24.1	59	2023-10-29 23:12:00
2	1	24.1	59	2023-10-29 23:12:14
3	1	24.2	59	2023-10-29 23:12:20
4	1	24.2	59	2023-10-29 23:12:25
5	1	24.2	59	2023-10-29 23:12:30
6	1	24.3	59	2023-10-29 23:12:36

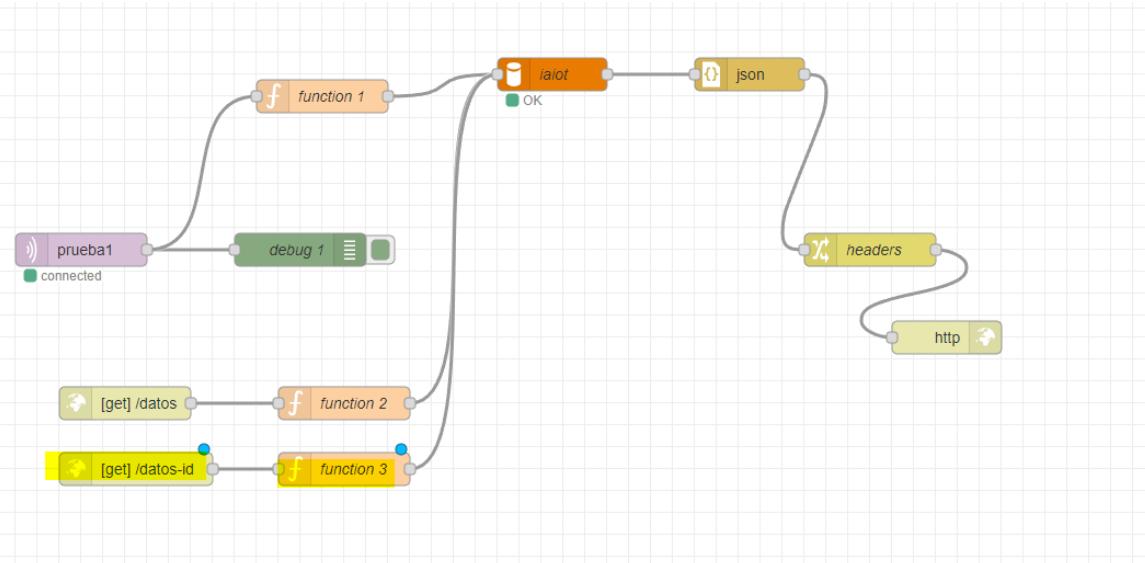
Aquí se muestra toda la respuesta del GET:

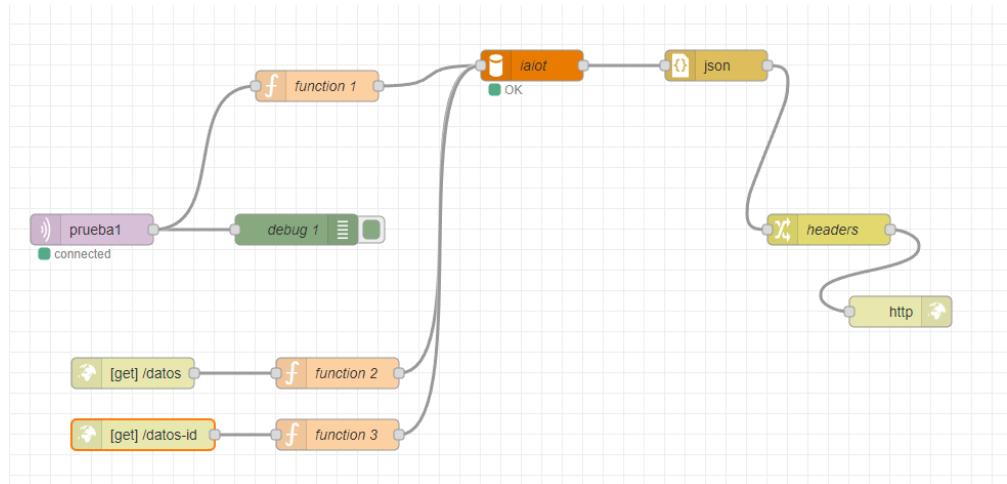
```
[
  {
    "id": 1,
    "idnodo": 1,
    "temperatura": 24.1,
    "humedad": 59,
    "fecha": "2023-10-30T04:12:09.000Z"
  },
  {
    "id": 2,
    "idnodo": 1,
    "temperatura": 24.1,
    "humedad": 59,
    "fecha": "2023-10-30T04:12:14.000Z"
  },
  {
    "id": 3,
    "idnodo": 1,
    "temperatura": 24.2,
    "humedad": 59,
    "fecha": "2023-10-30T04:12:20.000Z"
  },
  {
    "id": 4,
    "idnodo": 1,
    "temperatura": 24.2,
    "humedad": 59,
    "fecha": "2023-10-30T04:12:25.000Z"
  },
  {
    "id": 5,
    "idnodo": 1,
    "temperatura": 24.2,
    "humedad": 59,
    "fecha": "2023-10-30T04:12:30.000Z"
  },
  {
    "id": 6,
    "idnodo": 1,
    "temperatura": 24.3,
    "humedad": 59,
    "fecha": "2023-10-30T04:12:36.000Z"
  }
]
```

```
{
  "id": 4,
  "idnodo": 1,
  "temperatura": 24.2,
  "humedad": 59,
  "fecha": "2023-10-30T04:12:25.000Z"
},
{
  "id": 5,
  "idnodo": 1,
  "temperatura": 24.2,
  "humedad": 59,
  "fecha": "2023-10-30T04:12:30.000Z"
},
{
  "id": 6,
  "idnodo": 1,
  "temperatura": 24.3,
  "humedad": 59,
  "fecha": "2023-10-30T04:12:36.000Z"
}
]
```

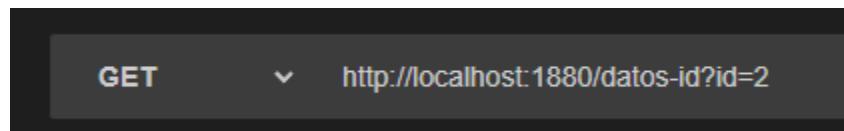
Se agrega otra ruta /datos-id con la función del siguiente código :

```
var id = msg.payload.id;
msg.topic = "select * from datos where id =" + id;
return msg;
```





Se realiza el GET con <http://localhost:1880/datos-id?id=2>



```

1 [[
2   {
3     "id": 2,
4     "idnodo": 1,
5     "temperatura": 24.1,
6     "humedad": 59,
7     "fecha": "2023-10-30T04:12:14.000Z"
8   }
9 ]]

```

```

[
  {
    "id": 2,
    "idnodo": 1,
    "temperatura": 24.1,
    "humedad": 59,
    "fecha": "2023-10-30T04:12:14.000Z"
  }
]

```

Por ultimo se hace un filtro GET de fecha de dos parámetros, de fecha de inicio y fecha de fin .

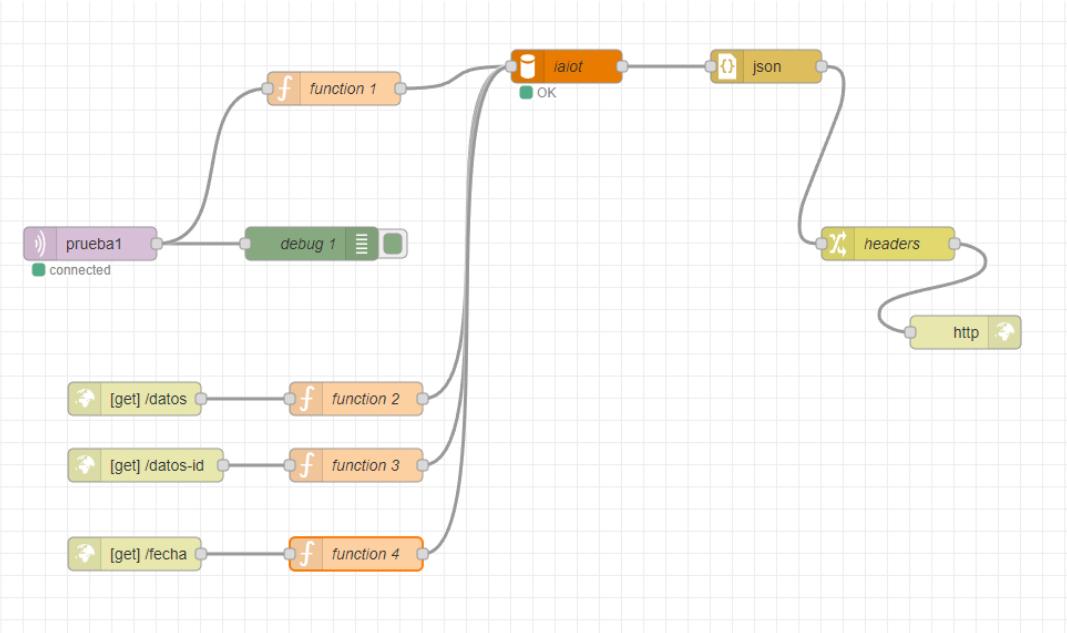
Por lo cual se crea ruta “/fecha” y se agrega función con el siguiente código :

```
var fechaInicio = msg.req.query.fechaInicio;
var fechaFin = msg.req.query.fechaFin;

if (fechaInicio && fechaFin) {
    msg.topic = "SELECT * FROM datos WHERE fecha BETWEEN '" + fechaInicio +
    "' AND '" + fechaFin + "'";
} else {

    msg.topic = "SELECT * FROM datos";
}

return msg;
```



Sabemos que la información guardada es :

```
mysql> select * from datos;
+----+-----+-----+-----+-----+
| id | idnodo | temperatura | humedad | fecha          |
+----+-----+-----+-----+-----+
| 1  | 1      | 24.1        | 59       | 2023-10-29 23:12:09 |
| 2  | 1      | 24.1        | 59       | 2023-10-29 23:12:14 |
| 3  | 1      | 24.2        | 59       | 2023-10-29 23:12:20 |
| 4  | 1      | 24.2        | 59       | 2023-10-29 23:12:25 |
| 5  | 1      | 24.2        | 59       | 2023-10-29 23:12:30 |
| 6  | 1      | 24.3        | 59       | 2023-10-29 23:12:36 |
+----+-----+-----+-----+-----+
```

En la url se pone los parámetros de la fecha y hora , por ejemplo si queremos de fecha inicio 2023-10-29 23:12:09 y hasta 2023-10-29 23:12:20 , se pone en el request :

http://localhost:1880/fecha?fechaInicio=2023-10-29 23:12:09 &fechaFin=2023-10-29 23:12:20

The screenshot shows a POSTMAN interface. At the top, it says "GET" and the URL "http://localhost:1880/fecha?fechaInicio=2023-10-29 23:12:09 &fechaFin=2023-10-29 23:12:20". Below this, there are tabs for "Params", "Authorization", "Headers (4)", "Body", and "Options". The "Params" tab is selected, displaying a table with two rows: "fechaInicio" with value "2023-10-29 23:12:09" and "fechaFin" with value "2023-10-29 23:12:20". There is also a row for "Key" with "Value". A "Send" button is at the top right. Below the table, under "Body", is a "JSON" dropdown set to "JSON". The response body shows a JSON array with three elements, each containing an object with fields: "id", "idnodo", "temperatura", "humedad", and "fecha". The first element's "fecha" field is highlighted in red.

La salida es :

```
[  
  {  
    "id": 1,  
    "idnodo": 1,  
    "temperatura": 24.1,  
    "humedad": 59,  
    "fecha": "2023-10-30T04:12:09.000Z"  
  },  
  {  
    "id": 2,  
    "idnodo": 1,  
    "temperatura": 24.1,  
    "humedad": 59,  
    "fecha": "2023-10-30T04:12:14.000Z"  
  },  
  {  
    "id": 3,  
    "idnodo": 1,  
    "temperatura": 24.1,  
    "humedad": 59,  
    "fecha": "2023-10-30T04:12:14.000Z"  
  }]
```

```

        "temperatura": 24.2,
        "humedad": 59,
        "fecha": "2023-10-30T04:12:20.000Z"
    }
]
```

Que es correctamente a los datos

```
mysql> select * from datos;
+----+-----+-----+-----+-----+
| id | idnodo | temperatura | humedad | fecha          |
+----+-----+-----+-----+-----+
| 1  | 1      | 24.1       | 59     | 2023-10-29 23:12:09 |
| 2  | 1      | 24.1       | 59     | 2023-10-29 23:12:14 |
| 3  | 1      | 24.2       | 59     | 2023-10-29 23:12:20 |
| 4  | 1      | 24.2       | 59     | 2023-10-29 23:12:25 |
| 5  | 1      | 24.2       | 59     | 2023-10-29 23:12:30 |
| 6  | 1      | 24.3       | 59     | 2023-10-29 23:12:36 |
+----+-----+-----+-----+-----+
```

Codigo ESP32 Platformio :

```
#include <Arduino.h>

#include <ArduinoJson.h>
#include <WiFi.h>
#include <PubSubClient.h>
//LIBRERIAS PARA DHT11 (TEMPERATURA Y HUMEDAD)
#include <Adafruit_Sensor.h>
#include <DHT.h>
//LIBRERIAS PARA FECHA Y HORA
#include <WiFi.h>
#include <NTPClient.h>
#include <WiFiUdp.h>
//DEFINICION DE PINES DHT11
#define DHTPIN 14 // 4 = PIN D4
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

// Define NTP Client to get time
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP);

// Variables to save date and time
```

```
String formattedDate;
String dayStamp;
String timeStamp;

#define mqttUser ""
#define mqttPass ""
#define mqttPort 1883
const char* ssid = "**NAME_WIFI**;//name wifi
const char* password = "*PASSWORD_WIFI*"; // clave de wifi
char mqttBroker[] = "192.168.*.*"; //ip del servidor

char mqttClientId[] = "prueba1"; //cualquier nombre
char inTopic[] = "prueba1";//topcico a suscribirse

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();
}
WiFiClient BClient;
PubSubClient client(BClient);
void reconnect() {
// Loop until we're reconnected
    while (!client.connected()) {
        Serial.print("Attempting MQTT connection...");
// Attempt to connect
        if (client.connect("", mqttUser, mqttPass)) {
            Serial.println("connected");
// Once connected, publish an announcement...
// Once connected, publish an announcement...
            float h= dht.readHumidity();
            float t =dht.readTemperature();

            String variable;
            StaticJsonDocument<256> doc;

/*  doc["Fecha"] = dayStamp;
    doc["Hora"] = timeStamp; */
```

```
doc["idnodo"] = 1;
doc["temperatura"] = t;
doc["humedad"] = h;

serializeJson(doc, variable);
int lon = variable.length()+1;
Serial.println(variable);
char datojson[lon];
variable.toCharArray(datojson, lon);
client.publish(inTopic,datojson);
client.disconnect();
delay(5000);
// ... and resubscribe
//client.subscribe("topic2");
} else {
Serial.print("failed, rc=");
Serial.print(client.state());
Serial.println(" try again in 5 seconds");
// Wait 5 seconds before retrying
delay(5000);
}
}
}

void setup_wifi() {
delay(10);
// We start by connecting to a WiFi network
Serial.println();
Serial.print("Connecting to ");
Serial.println(ssid);
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
delay(500);
Serial.print(".");
}
Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
// Initialize a NTPClient to get time
timeClient.begin();
// Set offset time in seconds to adjust for your timezone, for example:
```

```

// COLOMBIA -5 , entonces -5*3600 -> -18000
timeClient.setTimeOffset(-18000); //Thailand +7 = 25200
}

void setup()
{
    Serial.begin(9600); //Serial connection
    setup_wifi(); //WiFi connection
    client.setServer(mqttBroker, mqttPort );
    client.setCallback( callback );
    Serial.println("Setup done");
    delay(1500);
}

void loop(){
    while(!timeClient.update()) {
        timeClient.forceUpdate();
    }
    // The formattedDate comes with the following format:
    // 2018-05-28T16:00:13Z
    // We need to extract date and time
    formattedDate = timeClient.getFormattedDate();
    // Extract date
    int splitT = formattedDate.indexOf("T");
    dayStamp = formattedDate.substring(0, splitT);
    //Serial.print("DATE: ");
    //Serial.println(dayStamp);
    // Extract time
    timeStamp = formattedDate.substring(splitT+1, formattedDate.length()-1);
    if (!client.connected()) {
        reconnect();
    }
    client.loop();
}

```

## FASE DE VISUALIZACION WEB

Creación de tabla de usuarios

```

1 row in set (0.00 sec)

mysql> create table usuarios(
    -> user varchar(100),
    -> nombre varchar(100),
    -> password varchar(100),
    -> tipo int ,
    -> primary key(user));
Query OK, 0 rows affected (0.06 sec)

mysql>

```

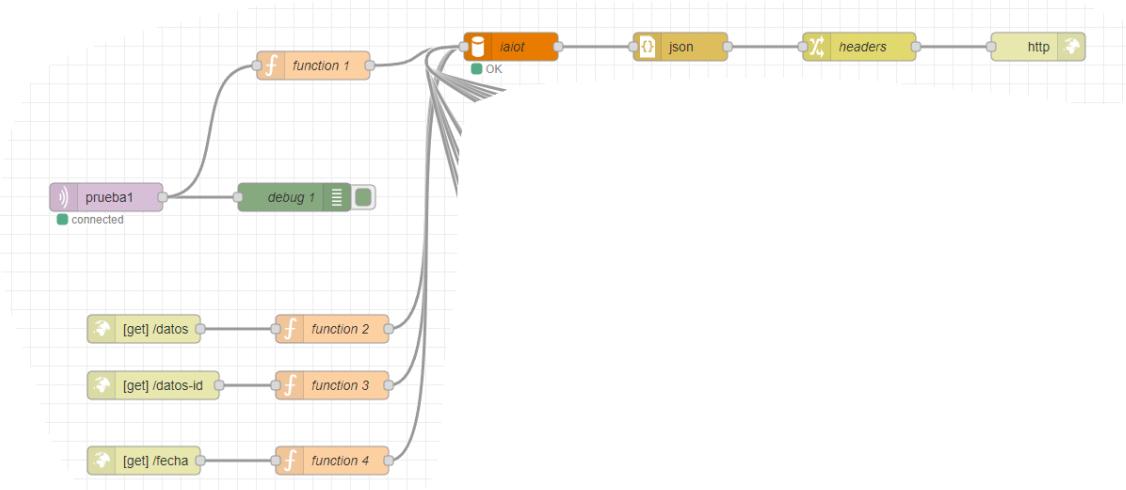
Creación tabla nodos :

```

mysql> create table nodos (
    -> idnodo int,
    -> nombreNodo varchar(100),
    -> ubicacion varchar(100),
    -> estado int,
    -> user varchar(100),
    -> primary key(idnodo));
Query OK, 0 rows affected (0.07 sec)

```

Se usa `C:\Users\User>node-red` para ejecutar node red:



Se cre la primera ruta y función



```

var user = msg.payload.user;
var nombre = msg.payload.nombre;
var pass = msg.payload.password;
var tipo = msg.payload.tipo;

```

```

msg.topic = `insert into usuarios values
("${user}","${nombre}","${pass}","${tipo})`;
return msg;

```

POST http://localhost:1880/crearUsuario

Params Authorization Headers (4) Body Options Code

Body Type: JSON

```

1 {
2   "user": "admin",
3   "nombre": "administrador",
4   "password": "123",
5   "tipo": 1
6 }

```

Status: 200 OK Duration: 42 ms

```

1 {
2   "fieldCount": 0,
3   "affectedRows": 1,
4   "insertId": 0,
5   "info": "",
6   "serverStatus": 2,
7   ...
8 }

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ... docker - prueba1 + ×

Empty set (0.01 sec)

mysql> select \* from usuarios;

user	nombre	password	tipo
admin	administrador	123	1

1 row in set (0.00 sec)

mysql> [put] /modificarUsuario

Se crea ruta y función



```

var user = msg.payload.user;
var nombre = msg.payload.nombre;
var pass = msg.payload.password;
var tipo = msg.payload.tipo;

```

```
msg.topic = `update usuarios set nombre="${nombre}", password="${pass}", tipo=${tipo} where user="${user}"`;
return msg;
```

PUT http://localhost:1880/modificarUsuario Send

Params Authorization Headers (4) Body Options Code

none  form-data  X-www-form-urlencoded  raw  binary  GraphQL JSON [Beautify](#)

```
1 {  
2   "user": "admin",  
3   "nombre": "administradorChanged",  
4   "password": "123",  
5   "tipo": 1  
6 }
```

Body Headers JSON Status: 200 OK Duration: 159 ms

```
1 {  
2   "fieldCount": 0,  
3   "affectedRows": 1,  
4   "insertId": 0,  
5   "info": "Rows matched: 1  Changed: 1  Warnings: 0",  
6   "serverStatus": 2,  
7 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ... docker - prueba1 + ▾

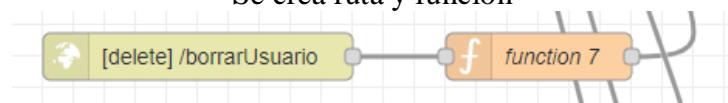
```
1 row in set (0.00 sec)
```

```
mysql> select * from usuarios;
```

user	nombre	password	tipo
admin	administradorChanged	123	1

```
1 row in set (0.01 sec)
```

```
var user = msg.payload.user;
msg.topic = `delete from usuarios where user="${user}"`;
return msg;
```



**DELETE** http://localhost:1880/borrarUsuario?user=admin **Send**

Params Authorization Headers (4) **Body** Options Code

none  form-data  X-WWW-Form-Urlencoded  raw  binary  GraphQL **JSON**  Beautify

```
1
```

Body Headers **JSON** Status: 200 OK Duration: 65 ms

```
1 [
2   "fieldCount": 0,
3   "affectedRows": 1,
4   "insertId": 0,
5   "info": "",
6   "serverStatus": 2,
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ... docker - prueba1 + ×

```
+-----+-----+-----+
| user | nombre           | password | tipo |
+-----+-----+-----+
| admin | administradorChanged | 123      | 1  |
+-----+-----+-----+
1 row in set (0.01 sec)

mysql> select * from usuarios;
Empty set (0.00 sec)

mysql> []
```

Se crea ruta y función



```
var user = msg.payload.user;
var pass = msg.payload.password;
```

```
msg.topic = `select * from usuarios where user="${user}" and  
password="${pass}"`;  
return msg;
```

The screenshot shows a browser-based API testing interface. At the top, it displays a POST request to the URL `http://localhost:1880/validarUsuario`. The "Headers" tab is selected, showing four headers: `Content-Type: application/json`, `Accept: application/json`, `Content-Type: application/x-www-form-urlencoded`, and `Content-Type: application/json`. The "Body" tab is selected, showing the JSON payload:

```
1 {  
2   "user": "user1",  
3   "password": "123"  
4 }  
5  
6 }
```

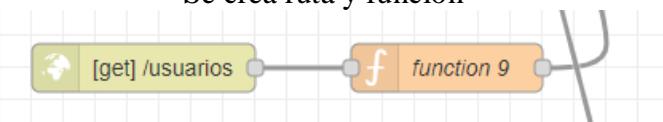
The response section shows the status as 200 OK with a duration of 27 ms. The "JSON" tab is selected, displaying the response body:

```
2 {  
3   "user": "user1",  
4   "nombre": "diego",  
5   "password": "123",  
6   "tipo": 1  
7 }  
8  
9 }
```

At the bottom, the terminal window shows the MySQL command `select * from usuarios;` and its output:

```
mysql> select * from usuarios;  
+-----+-----+-----+-----+  
| user | nombre | password | tipo |  
+-----+-----+-----+-----+  
| user1 | diego | 123 | 1 |  
+-----+-----+-----+-----+  
1 row in set (0.00 sec)  
  
mysql> |
```

Se crea ruta y función



```
msg.topic = "select * from usuarios";
return msg;
```

GET http://localhost:1880/usuarios Send

Params Authorization Headers (4) Body Options Code

none  form-data  X-www-form-urlencoded  raw  binary  GraphQL JSON ▼ Beautify

```
1
```

Body Headers JSON ▼ Status: 200 OK Duration: 26 ms

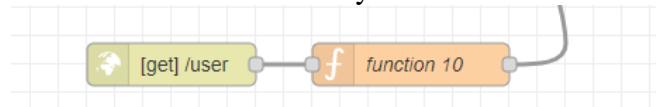
```
3 |     "user": "user1",
4 |     "nombre": "diego",
5 |     "password": "123",
6 |     "tipo": 1
7 | },
8 | {
9 |     "user": "user2",
10|     "nombre": "sara",
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ... docker - prueba1 + ▾ ...

```
+-----+-----+-----+
| user1 | diego | 123    | 1 |
| user2 | sara  | 123    | 2 |
+-----+-----+-----+
2 rows in set (0.00 sec)
```

mysql> [ ]

Se crea ruta y función



```
var user = msg.payload.user;
msg.topic = `select * from usuarios where user="${user}"`;
return msg;
```

GET http://localhost:1880/user?user=user1 **Send**

Params Authorization Headers (4) **Body** Options Code

none  form-data  X-WWW-Form-Urlencoded  raw  binary  GraphQL **JSON**  Beautify

```
1
```

Body Headers **JSON** Status: 200 OK Duration: 34 ms

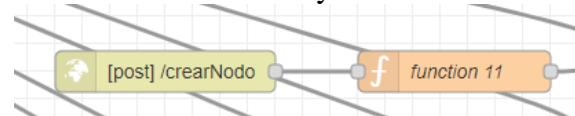
```
2 {  
3   "user": "user1",  
4   "nombre": "diego",  
5   "password": "123",  
6   "tipo": 1  
7 }
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS ... docker - prueba1 + ×

```
+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> select * from usuarios;
+-----+-----+-----+-----+
| user | nombre | password | tipo |
+-----+-----+-----+-----+
| user1 | diego | 123      | 1    |
| user2 | sara  | 123      | 2    |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

Se crea ruta y función



```
var idnodo = msg.payload.idnodo;
var nombre = msg.payload.nombre;
var ubic = msg.payload.ubicacion;
```

```
var estado = msg.payload.estado;
var user = msg.payload.user;
msg.topic = `insert into nodos values
(${idnodo} , "${nombre}" , "${ubic}" , ${estado} , "${user}" )`;
return msg;
```

POST <http://localhost:1880/crearNodo> Send

Params Authorization Headers (4) Body Options Code

```
1 [ {  
2     "idnodo":1,  
3     "nombre": "Nodo Uno",  
4     "ubicacion": "Cali Sur",  
5     "estado":1,  
6     "user": "user1"  
7 } ]
```

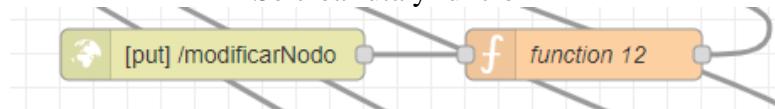
Body Headers JSON Status: 200 OK Duration: 26 ms

```
1 {  
2     "fieldCount": 0,  
3     "affectedRows": 1,  
4     "insertId": 0,  
5     "info": "",  
6     "serverStatus": 2,
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ... docker - prueba1 + ×

```
mysql> select * from nodos;  
Empty set (0.06 sec)  
  
mysql> select * from nodos;  
+-----+-----+-----+-----+-----+  
| idnodo | nombreNodo | ubicacion | estado | user |  
+-----+-----+-----+-----+-----+  
|      1 | Nodo Uno   | Cali Sur   |      1 | user1 |  
+-----+-----+-----+-----+-----+  
1 row in set (0.00 sec)
```

## Se crea ruta y función



```
var idnodo = msg.payload.idnodo;  
var nombre = msg.payload.nombre;  
var ubic = msg.payload.ubicacion;
```

```
var estado = msg.payload.estado;
var user = msg.payload.user;

msg.topic = `update nodos set
nombreNodo="${nombre}",ubicacion="${ubic}",estado="${estado}",user="${user}"
where idnodo=${idnodo}`;
return msg;
```

PUT <http://localhost:1880/modificarNodo> Send

Params Authorization Headers (4) Body Options Code

```
1 {  
2     "idnodo":1,  
3     "nombre": "chnaged",  
4     "ubicacion": "changed",  
5     "estado":1,  
6     "user": "user1"  
7 }  
8
```

Body Headers JSON Status: 200 OK Duration: 26 ms

```
1 {  
2     "fieldCount": 0,  
3     "affectedRows": 1,  
4     "insertId": 0,  
5     "info": "Rows matched: 1  Changed: 1  Warnings: 0",  
6     "serverStatus": 2.
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ... docker - prueba1 + ×

```
+-----+-----+-----+-----+-----+  
1 row in set (0.00 sec)  
  
mysql> select * from nodos;  
+-----+-----+-----+-----+-----+  
| idnodo | nombreNodo | ubicacion | estado | user |  
+-----+-----+-----+-----+-----+  
|      1 | chnaged   | changed   |       1 | user1 |  
+-----+-----+-----+-----+-----+  
1 row in set (0.00 sec)
```

## Se crea ruta y función



```
var idnodo = msg.payload.idnodo;
msg.topic = `delete from nodos where idnodo="${idnodo}"`;
return msg;
```

**DELETE**  **Send**

Params    Authorization    Headers (4)    **Body**    Options    Code

none     form-data     X-www-form-urlencoded     raw     binary     GraphQL    **JSON**

```

1  [
2    "idnodo":2,
3    "nombre": "Nodo Dos",
4    "ubicacion": "Cali Sur",
5    "estado":1,

```

Body    Headers    **JSON**  Status: 200 OK Duration: 31 ms

```

1  {
2    "fieldCount": 0,
3    "affectedRows": 1,
4    "insertId": 0,
5    "info": ""

```

PROBLEMS    OUTPUT    DEBUG CONSOLE    **TERMINAL**    PORTS    ...    docker - prueba1    +    ⌂    ⌂    ...    ^

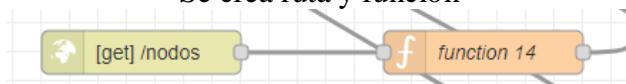
```

| idnodo | nombreNodo | ubicacion | estado | user |
+-----+-----+-----+-----+
|     1 | chnaged   | changed   |      1 | user1 |
|     2 | Nodo Dos  | Cali Sur  |      1 | user2 |
+-----+-----+-----+-----+
2 rows in set (0.01 sec)

mysql> select * from nodos;
+-----+-----+-----+-----+
| idnodo | nombreNodo | ubicacion | estado | user |
+-----+-----+-----+-----+
|     1 | chnaged   | changed   |      1 | user1 |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

```

Se crea ruta y función



```

msg.topic = "select * from nodos";
return msg;

```

GET http://localhost:1880/nodos Send

Params Authorization Headers (4) Body Options Code

data urlencoded

```
1 {  
2     "idnodo":2,  
3     "nombre": "Nodo Dos",  
4     "ubicacion":"Cali Sur",  
5     "estado":1,  
6     "user":"user2"  
7 }  
8 }
```

Body Headers JSON Status: 200 OK Duration: 13 ms

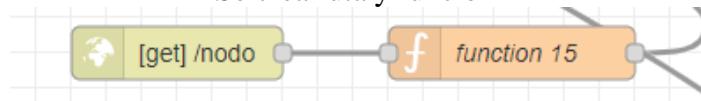
```
1 | idnodo | nombreNodo | ubicacion | estado | user |  
2 | 1 | chnaged | changed | 1 | user1 |  
3 | 2 | Nodo Dos | Cali Sur | 1 | user2 |  
4 |  
5 |  
6 |  
7 |  
8 |  
9 |  
10 |
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ... docker - prueba1 + ×

	idnodo	nombreNodo	ubicacion	estado	user
1	1	chnaged	changed	1	user1
2	2	Nodo Dos	Cali Sur	1	user2

2 rows in set (0.01 sec)

Se crea ruta y función



```

var idnodo = msg.payload.idnodo;
msg.topic = "select * from nodos where idnodo=" + idnodo;
return msg;

```

The screenshot shows a Node-RED interface with the following components:

- HTTP Request Node:** GET method, URL: http://localhost:1880/nodo?idnodo=1.
- JSON Node:** Body tab, showing a JSON object with fields: idnodo:2, nombre: "Nodo Dos", ubicacion: "Cali Sur", estado:1, user:"user2".
- MySQL Terminal:** Status: 200 OK Duration: 14 ms. The terminal shows the following MySQL query results:
 

idnodo	nombreNodo	ubicacion	estado	user
1	chnaged	changed	1	user1
2	Nodo Dos	Cali Sur	1	user2

 The message also includes the text: 2 rows in set (0.01 sec).

Se crea ruta y función



```
var user = msg.payload.user;
```

```
msg.topic = `select * from nodos where user ="${user}"`;
return msg;
```

GET http://localhost:1880/nodoUser?user=user1 Send

Params Authorization Headers (4) Body Options Code

data urlencoded

1

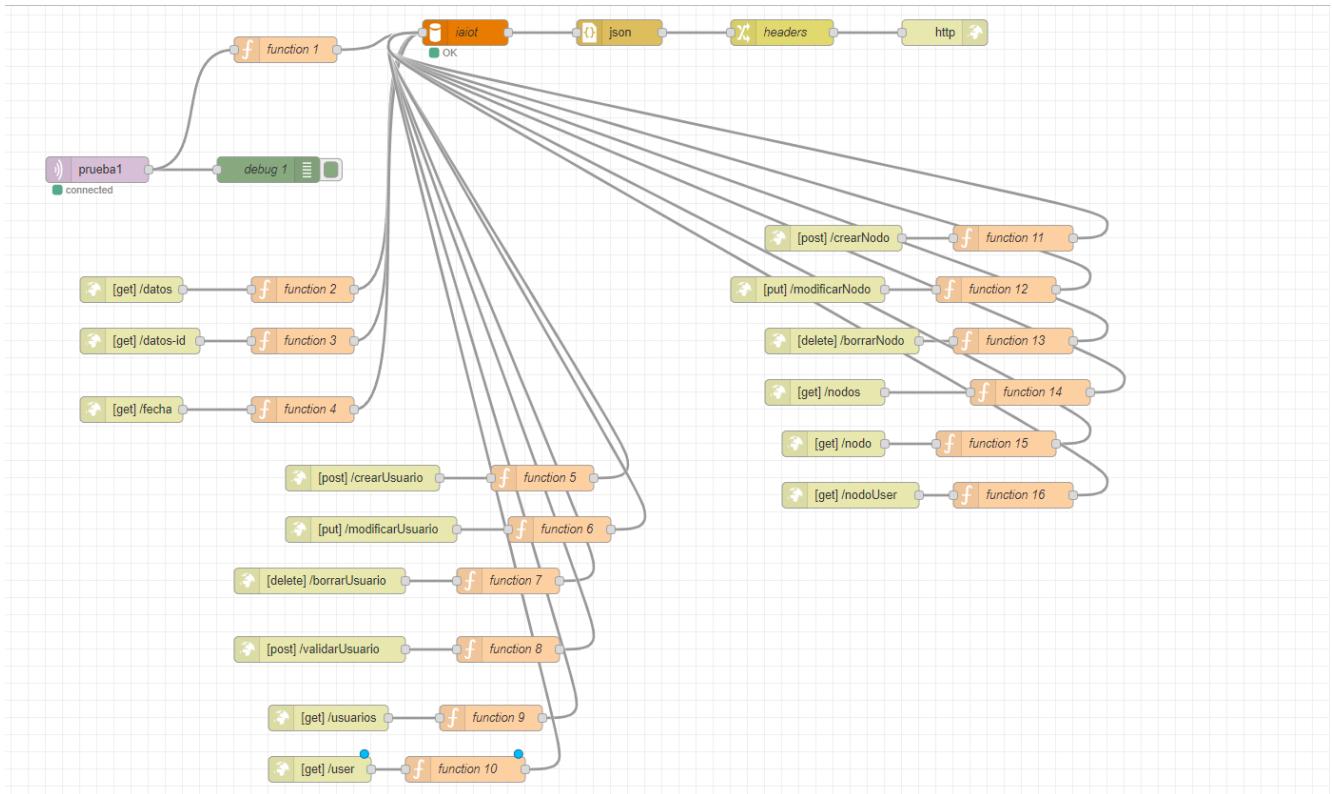
Status: 200 OK Duration: 14 ms

Body Headers JSON

```
1 [ ]
2 {
3     "idnodo": 1,
4     "nombreNodo": "chnaged",
5     "ubicacion": "changed",
6     "estado": 1,
7     "user": "user1"
8 ]
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ... docker - prueba1 + ×

```
| idnodo | nombreNodo | ubicacion | estado | user |
+-----+-----+-----+-----+
| 1 | chnaged | changed | 1 | user1 |
| 2 | Nodo Dos | Cali Sur | 1 | user2 |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```



Este proyecto de node-red también se puede importar en el cual esta en un nodered.json

Para visualizar la información de una forma didáctica se crean archivos html y php para la parte backend , en este caso ya que se tenia Docker , para evitar instalar php y apache se creo un docker-compose.yml , en el cual hay una imagen que tiene php y apache juntos.

```
version: '3'
services:
  webserver:
    image: php:7.4-apache # Utilizamos una imagen que incluye PHP y Apache
    ports:
      - "80:80" # Mapeamos el puerto 80 del contenedor al puerto 80 del host
    volumes:
      - C:\path-to\php-files:/var/www/html
```

Pero como los archivos php daban a la ruta de localhost se debe cambiar a la ip del pc , por ejemplo si en el archivo ingresar.php esta :

```
// URL de la solicitud POST
$url = 'http://localhost:1880/validarUsuario';
```

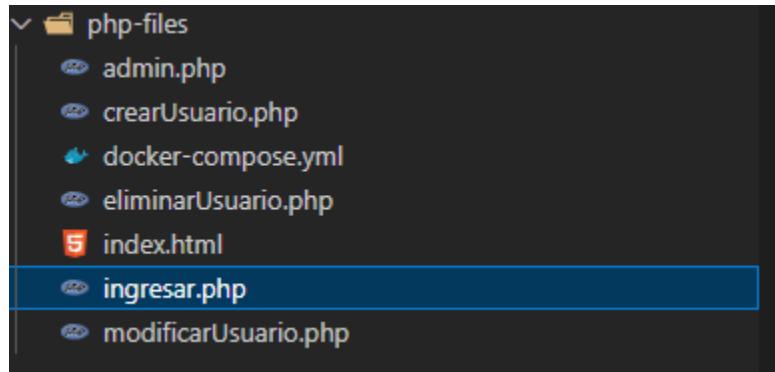
Tocaría cambiarlo por la ip del pc

```
// URL de la solicitud POST
```

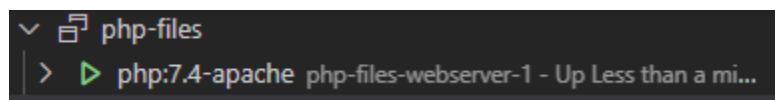
```
$url = 'http://IP_DE_LOCAL:1880/validarUsuario';
```

Y así con los demás archivo que tenga la ruta .

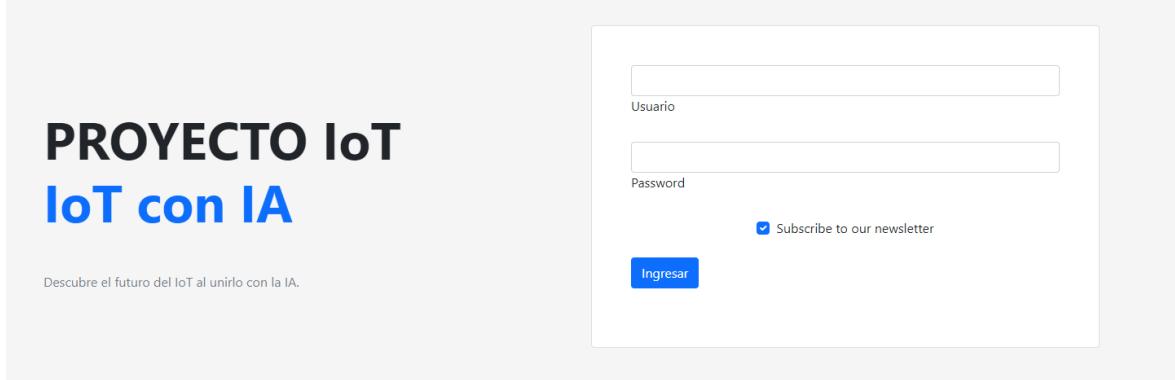
Teniendo ya todos los archivos ordenados en el mismo directorio del Docker compose



Se realiza “docker-compose up”



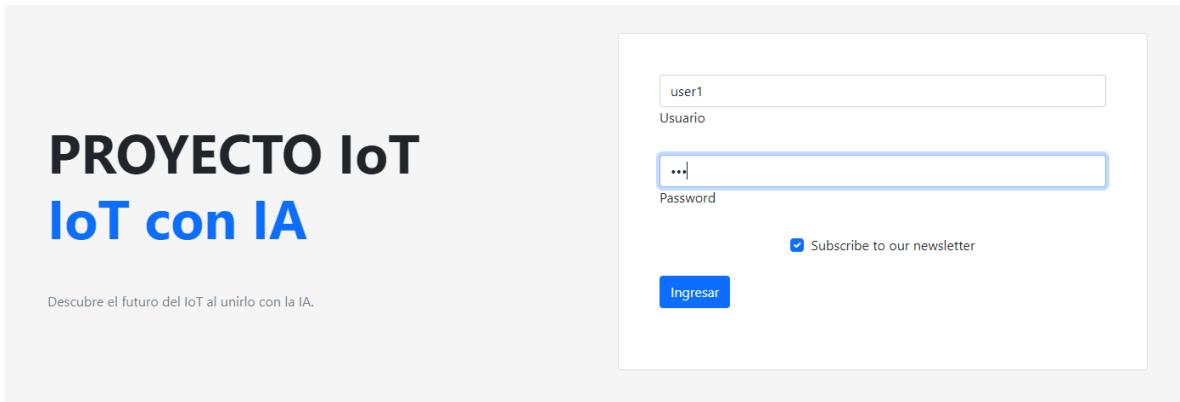
Y dirigiendo al localhost se visualiza:



Recorda que en el mysql es :

```
mysql> select * from usuarios;
+-----+-----+-----+
| user | nombre | password | tipo |
+-----+-----+-----+
| user1 | diego | 123      |    1 |
| user2 | sara   | 123      |    2 |
+-----+-----+-----+
2 rows in set (0.00 sec)
```

Por eso se intenta ingresar con el user1



Como admin tendrá esta interfaz en el cual puede crear , modificar s y eliminar usuarios y nodos, y puede dar una visualización de todos los datos.

Proyecto IoT						Logout diego
Nombre	Usuario	Password	tipo	Editar	Eliminar	
diego	user1	123	1	MODIFICAR	ELIMINAR	
sara	user2	123	3	MODIFICAR	ELIMINAR	
<a href="#">CREAR USUARIO</a>						

Dando crear usuario:

The image shows a modal dialog box titled "CREAR USUARIO". It contains four input fields: "Nombre" (placeholder: " "), "Usuario" (placeholder: " "), "Contraseña" (placeholder: " "), and "Tipo" (placeholder: " "). At the bottom of the dialog are two buttons: "Close" and "Crear Usuario". The background of the dialog is dark gray, matching the background of the main application window.

Proyecto IoT Usuarios Nodos Datos Logout diego

Nombre	Usuario	Password		Editar	Eliminar
diego	user1	123		<input type="button" value="MODIFICAR"/>	<input type="button" value="ELIMINAR"/>
sara	user2	123		<input type="button" value="MODIFICAR"/>	<input type="button" value="ELIMINAR"/>
<b>CREAR USUARIO</b>					

**CREAR USUARIO**

Nombre:

Usuario:

Contraseña:

Tipo:

Proyecto IoT Usuarios Nodos Datos Logout diego

Nombre	Usuario	Password	tipo	Editar	Eliminar
<input type="text" value="diego"/>	user1	<input type="text" value="123"/>	<input type="text" value="1"/>	<input type="button" value="MODIFICAR"/>	<input type="button" value="ELIMINAR"/>
<input type="text" value="sara"/>	user2	<input type="text" value="123"/>	<input type="text" value="3"/>	<input type="button" value="MODIFICAR"/>	<input type="button" value="ELIMINAR"/>
<input type="text" value="maria"/>	user3	<input type="text" value="123"/>	<input type="text" value="2"/>	<input type="button" value="MODIFICAR"/>	<input type="button" value="ELIMINAR"/>

**CREAR USUARIO**

Se evidencia la creación del usuario en la base de datos

user	nombre	password	tipo
user1	diego	123	1
user2	sara	123	3
user3	maria	123	2

Si se quisiera modificar por ejemplo un usuario en este caso el user 3 maria con el tipo ahora de 4 :

Proyecto IoT Usuarios Nodos Datos Logout diego

Nombre	Usuario	Password	tipo	Editar	Eliminar
diego	user1	<input type="text" value="123"/>	<input type="text" value="1"/>	<input type="button" value="MODIFICAR"/>	<input type="button" value="ELIMINAR"/>
sara	user2	<input type="text" value="123"/>	<input type="text" value="3"/>	<input type="button" value="MODIFICAR"/>	<input type="button" value="ELIMINAR"/>
maria	user3	<input type="text" value="123"/>	<input type="text" value="4"/>	<input type="button" value="MODIFICAR"/>	<input type="button" value="ELIMINAR"/>
<b>CREAR USUARIO</b>					

user	nombre	password	tipo
user1	diego	123	1
user2	sara	123	3
user3	maria	123	4

## Y eliminándolo

Proyecto lot						Logout diego
Nombre	Usuario	Password	tipo	Editar	Eliminar	
diego	user1	123	1	<button>MODIFICAR</button>	<button>ELIMINAR</button>	
sara	user2	123	3	<button>MODIFICAR</button>	<button>ELIMINAR</button>	
maria	user3	123	4	<button>MODIFICAR</button>	<button>ELIMINAR</button>	

[CREAR USUARIO](#)

```
+-----+-----+-----+-----+
| user | nombre | password | tipo |
+-----+-----+-----+-----+
| user1 | diego | 123      | 1   |
| user2 | sara  | 123      | 3   |
+-----+-----+-----+-----+
```

Lo mismo se puede hacer con los nodos:

Proyecto lot						Logout diego
IdNodo	Nombre	Ubicacion	Estado	Usuario	Editar	Eliminar
1	chnaged	changed	1	user1	<button>MODIFICAR</button>	<button>ELIMINAR</button>
2	nodo2	popayan	1	user2	<button>MODIFICAR</button>	<button>ELIMINAR</button>

[CREAR NODO](#)

Se crea un nodo

Proyecto lot

Usuarios			Nodos	Datos
IdNodo	Nombre	Ubicacion	Estado	Usuario
1	chnaged	changed	1	user1
2	nodo2	popayan	1	user2

[CREAR NODO](#)

**CREAR USUARIO**

IdNodo  
3

Nombre  
nodo3

Ubicacion  
cali

Estado  
1

Usuario

Close Crear Nodo

Proyecto lot						
IdNodo		Nombre	Ubicacion	Estado	Usuario	Editar
1	chnaged	changed	1	user1	<a href="#">MODIFICAR</a>	<a href="#">ELIMINAR</a>
2	nodo2	popayan	1	user2	<a href="#">MODIFICAR</a>	<a href="#">ELIMINAR</a>
3	nodo3	cali	1	user8	<a href="#">MODIFICAR</a>	<a href="#">ELIMINAR</a>

[CREAR NODO](#)

```
+-----+-----+-----+-----+
| idnodo | nombreNodo | ubicacion | estado | user |
+-----+-----+-----+-----+
|      1 | chnaged    | changed   |      1 | user1  |
|      2 | nodo2      | popayan   |      1 | user2  |
|      3 | nodo3      | cali      |      1 | user8  |
+-----+-----+-----+-----+
```

Ahora modificar el usuario del idnodo 3 :

Proyecto lot						
IdNodo		Nombre	Ubicacion	Estado	Usuario	Editar
1	chnaged	changed	1	user1	<a href="#">MODIFICAR</a>	<a href="#">ELIMINAR</a>
2	nodo2	popayan	1	user2	<a href="#">MODIFICAR</a>	<a href="#">ELIMINAR</a>
3	nodo3	cali	1	user7	<a href="#">MODIFICAR</a>	<a href="#">ELIMINAR</a>

[CREAR NODO](#)

```
+-----+-----+-----+-----+
| idnodo | nombreNodo | ubicacion | estado | user |
+-----+-----+-----+-----+
|      1 | chnaged    | changed   |      1 | user1  |
|      2 | nodo2      | popayan   |      1 | user2  |
|      3 | nodo3      | cali      |      1 | user7  |
+-----+-----+-----+-----+
```

Y eliminar

Proyecto lot						
IdNodo		Nombre	Ubicacion	Estado	Usuario	Editar
1	chnaged	changed	1	user1	<a href="#">MODIFICAR</a>	<a href="#">ELIMINAR</a>
2	nodo2	popayan	1	user2	<a href="#">MODIFICAR</a>	<a href="#">ELIMINAR</a>
3	nodo3	cali	1	user7	<a href="#">MODIFICAR</a>	<a href="#">ELIMINAR</a>

[CREAR NODO](#)

```
+-----+-----+-----+-----+
| idnodo | nombreNodo | ubicacion | estado | user |
+-----+-----+-----+-----+
|      1 | chnaged    | changed   |      1 | user1  |
|      2 | nodo2      | popayan   |      1 | user2  |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

En Datos se puede observar la visualización de diferentes formas de todos los datos:

Proyecto lot    Usuarios    Nodos    **Datos**

## Selecciona Visualizador de datos

Selecciona una opción:

Tabla de datos

[Ir a la página seleccionada](#)

Se selecciona la forma de visualización y se le da click en “ir a la pagina seleccionada” , en este caso Tabla de datos:

### DATOS CAPTURADOS

ID	IDNODO	TEMPERATURA	HUMEDAD	FECHA
1	1	24.1	59	2023-10-30T04:12:09.000Z
2	1	24.1	59	2023-10-30T04:12:14.000Z
3	1	24.2	59	2023-10-30T04:12:20.000Z
4	1	24.2	59	2023-10-30T04:12:25.000Z
5	1	24.2	59	2023-10-30T04:12:30.000Z
6	1	24.3	59	2023-10-30T04:12:36.000Z
7	2	25.8	62	2023-11-11T23:04:04.000Z
8	2	26.2	61	2023-11-11T23:04:23.000Z
9	2	26.2	61	2023-11-11T23:04:29.000Z
10	2	26.2	61	2023-11-11T23:04:35.000Z

En Cards datos se visualiza :

## Selecciona Visualizador de datos

Selecciona una opción:

Cards datos

[Ir a la página seleccionada](#)

### DATO ESPECIFICO

Ingrese un número ID:

[Actualizar ID](#)

Se indica el id a observar , en este caso el 1

## DATO ESPECIFICO

Ingrese un número ID:

1

Actualizar ID

2023-10-30T04:12:09.000Z

### TEMPERATURA

24.1

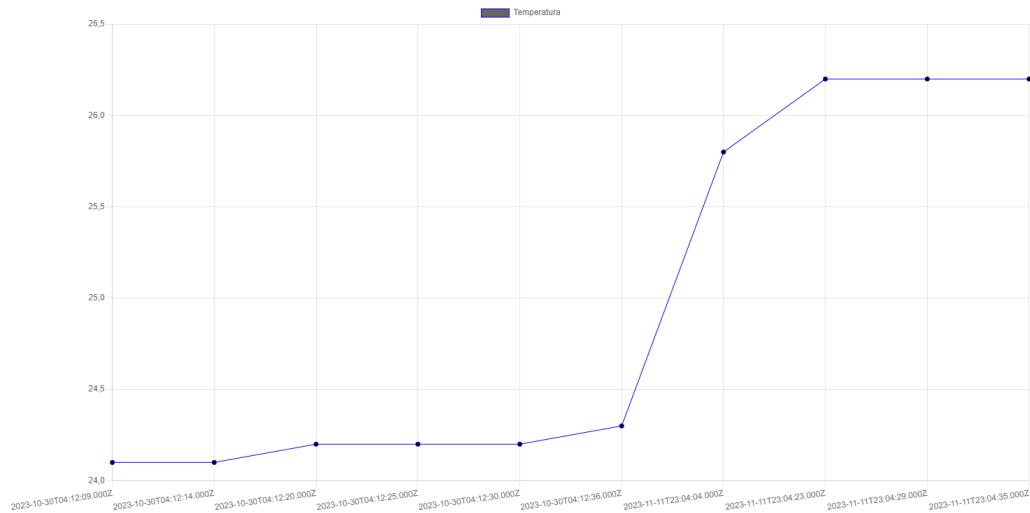
[Ver gráfica](#)

### HUMEDAD

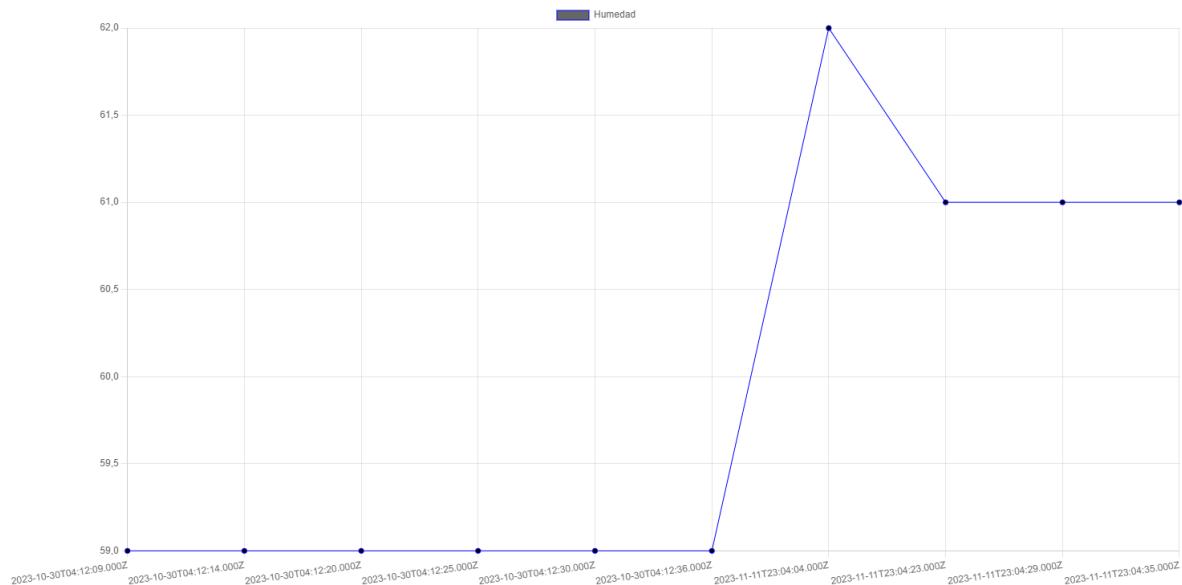
59

[Ver gráfica](#)

Se observa la fecha , temperatura y humedad en ese id específico .  
En ver grafica se visualiza todos los datos de temperatura y humedad respectivos



### Humedad



Estas graficas también se pueden observar en el seleccionador :

### Selecciona Visualizador de datos

Selecciona una opción:

Grafica de temperatura

[Ir a la página seleccionada](#)

### Selecciona Visualizador de datos

Selecciona una opción:

Grafica de humedad

[Ir a la página seleccionada](#)

Para salir de la sesión con logout :

Proyecto IoT							Logout diego
IdNodo	Nombre	Ubicacion	Estado	Usuario	Editar	Eliminar	
1	chnaged	changed	1	user1	<a href="#">MODIFICAR</a>	<a href="#">ELIMINAR</a>	
2	nodo2	popayan	1	user2	<a href="#">MODIFICAR</a>	<a href="#">ELIMINAR</a>	
<a href="#">CREAR NODO</a>							

Ingresando con un usuario normal , no admin :

En este caso es el user2:

The image shows a login form for a website titled "PROYECTO IoT IoT con IA". The form has fields for "Usuario" (user2) and "Password". There is a checkbox for "Subscribe to our newsletter" which is checked. A blue "Ingresar" button is at the bottom.

user2  
Usuario

...  
Password

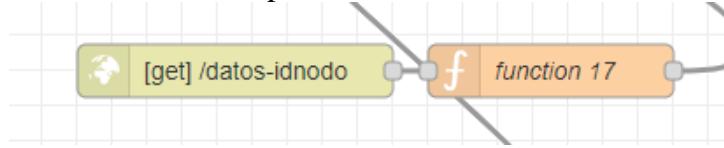
Subscribe to our newsletter

Ingresar

Observará los datos de su ID Nodo correspondiente en este caso su ID Nodo es el 2

ID Nodo	Temperatura	Humedad	Fecha
2	25.8	62	2023-11-11T23:04:04.000Z
2	26.2	61	2023-11-11T23:04:23.000Z
2	26.2	61	2023-11-11T23:04:29.000Z
2	26.2	61	2023-11-11T23:04:35.000Z

Para la realización de esta parte se añade en Node-red dos funciones:



```
var idnodo = msg.payload.idnodo;
msg.topic = "select * from datos where idnodo =" + idnodo;
return msg;
```

GET http://localhost:1880/datos-idnodo?idnodo=2

Params Authorization Headers (4) Body Options

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

1

Body Headers JSON Status: 200 OK Duration: 26 ms

```
[{"id": 7, "idnodo": 2, "temperatura": 25.8, "humedad": 62, "fecha": "2023-11-11T23:04:04.000Z"}]
```



```
var name = msg.payload.name;
msg.topic = `select * from usuarios where nombre="${name}"`;
return msg;
```

GET http://localhost:1880/userName?name=sara Send

Params Authorization Headers (4) Body Options

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

1

Body Headers JSON Status: 200 OK Duration: 96 ms

```
[{"user": "user2", "nombre": "sara", "password": "123", "tipo": 3}]
```

Codigo ESP32 Platformio con el idnodo 2 :

```
#include <Arduino.h>

#include <ArduinoJson.h>
#include <WiFi.h>
#include <PubSubClient.h>
//LIBRERIAS PARA DHT11 (TEMPERATURA Y HUMEDAD)
#include <Adafruit_Sensor.h>
#include <DHT.h>
//LIBRERIAS PARA FECHA Y HORA
#include <WiFi.h>
#include <NTPClient.h>
#include <WiFiUdp.h>
//DEFINICION DE PINES DHT11
#define DHTPIN 14 // 4 = PIN D4
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

// Define NTP Client to get time
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP);

// Variables to save date and time
String formattedDate;
String dayStamp;
String timeStamp;
```

```

#define mqttUser ""
#define mqttPass ""
#define mqttPort 1883
const char* ssid = "**NAME_WIFI**;//name wifi
const char* password = "*PASSWORD_WIFI*"; // clave de wifi
char mqttBroker[] = "192.168.*.*"; //ip del servidor

char mqttClientId[] = "prueba1"; //cualquier nombre
char inTopic[] = "prueba1";//topico a suscribirse

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();
}
WiFiClient BClient;
PubSubClient client(BClient);
void reconnect() {
// Loop until we're reconnected
    while (!client.connected()) {
        Serial.print("Attempting MQTT connection...");
// Attempt to connect
        if (client.connect("", mqttUser, mqttPass)) {
            Serial.println("connected");
            // Once connected, publish an announcement...
            // Once connected, publish an announcement...
            float h= dht.readHumidity();
            float t =dht.readTemperature();

            String variable;
            StaticJsonDocument<256> doc;

/*  doc["Fecha"] = dayStamp;
    doc["Hora"] = timeStamp; */

    doc["idnodo"] = 2;
    doc["temperatura"] = t;
    doc["humedad"] = h;
}

```

```
serializeJson(doc, variable);
int lon = variable.length()+1;
Serial.println(variable);
char datojson[lon];
variable.toCharArray(datojson, lon);
client.publish(inTopic,datojson);
client.disconnect();
delay(5000);
// ... and resubscribe
//client.subscribe("topic2");
} else {
Serial.print("failed, rc=");
Serial.print(client.state());
Serial.println(" try again in 5 seconds");
// Wait 5 seconds before retrying
delay(5000);
}
}
}

void setup_wifi() {
delay(10);
// We start by connecting to a WiFi network
Serial.println();
Serial.print("Connecting to ");
Serial.println(ssid);
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
delay(500);
Serial.print(".");
}
Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
// Initialize a NTPClient to get time
timeClient.begin();
// Set offset time in seconds to adjust for your timezone, for example:
// COLOMBIA -5 , entonces -5*3600 -> -18000
timeClient.setTimeOffset(-18000); //Thailand +7 = 25200
}
```

```

void setup()
{
    Serial.begin(9600); //Serial connection
    setup_wifi(); //WiFi connection
    client.setServer(mqttBroker, mqttPort );
    client.setCallback( callback );
    Serial.println("Setup done");
    delay(1500);
}

void loop(){
    while(!timeClient.update()) {
        timeClient.forceUpdate();
    }
    // The formattedDate comes with the following format:
    // 2018-05-28T16:00:13Z
    // We need to extract date and time
    formattedDate = timeClient.getFormattedDate();
    // Extract date
    int splitT = formattedDate.indexOf("T");
    dayStamp = formattedDate.substring(0, splitT);
    //Serial.print("DATE: ");
    //Serial.println(dayStamp);
    // Extract time
    timeStamp = formattedDate.substring(splitT+1, formattedDate.length()-1);
    if (!client.connected()) {
        reconnect();
    }
    client.loop();
}

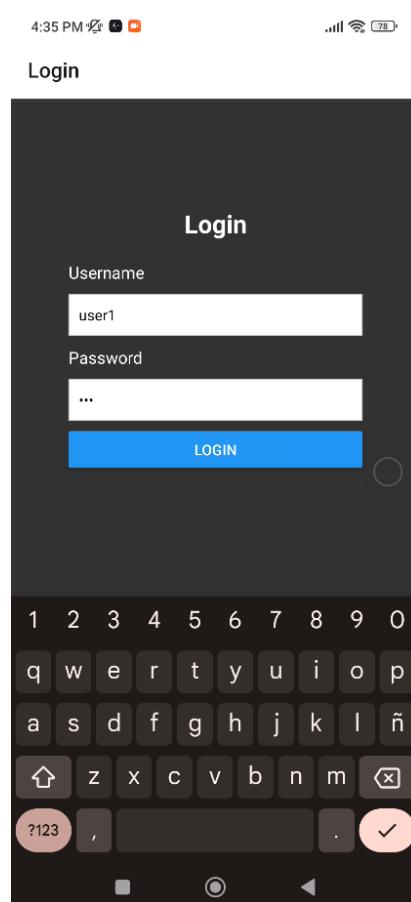
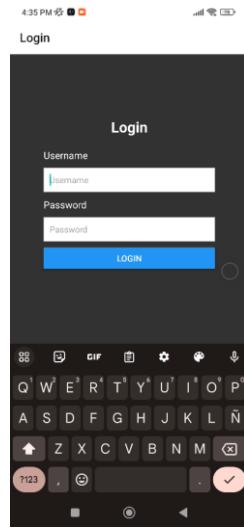
```

## FASE DE VISUALIZACION MOVIL

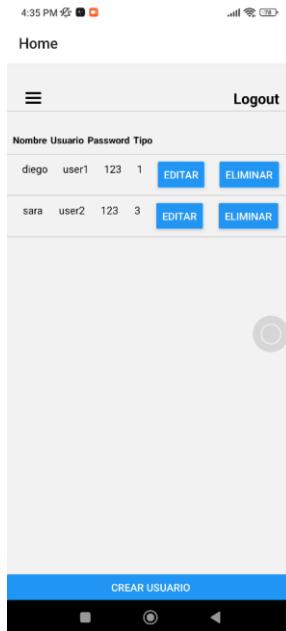
Se desarrolló la visualización en móvil usando react native con expo , debido a que sirve para sistema operativos de Android y ios ,utilizando la pasrte backend de node-red en donde se quería realizar las mismas acciones que en la web.

La primera parte era el login en donde se daba dos usuarios el usuario admin con tipo 1 y el cliente el usuario con tipo diferente de 1

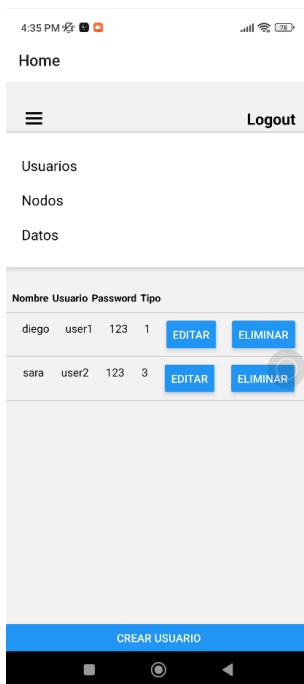
En este caso primero se muestra con el usuario admin



El usuario “user1” es de tipo 1 con lo cual es admin en donde la primera pantalla mostrará el apartado de los usuarios , donde puede crear , editar y eliminar un usuario.



Dando al icono ios se visualiza los diferentes apartados de la aplicación en el cual son Usuarios, Nodos y Datos



Al darle al botón “CREAR USUARIO” desplegará la interfaz para crear el usuario

4:36 PM   

### CREAR USUARIO

Nombre

Username

Password

Tipo

**CREAR**

**CANCELAR**



Introduciendo los valores específicos para crear el usuario este debe darle al botón “CREAR” para confirmar la creación del usuarios o “CANCELAR” si se desea cancelar la creación del usuario

4:36 PM   

### CREAR USUARIO

Nombre

samir

Username

user4

Password

123

Tipo

3

**CREAR**

**CANCELAR**



Al darle “CREAR” se visualiza inmediatamente el usuario creado en el apartado de “Usuarios”

## Home

Nombre Usuario Password Tipo					Logout
diego	user1	123	1	<a href="#">EDITAR</a>	<a href="#">ELIMINAR</a>
sara	user2	123	3	<a href="#">EDITAR</a>	<a href="#">ELIMINAR</a>
samir	user4	123	3	<a href="#">EDITAR</a>	<a href="#">ELIMINAR</a>

CREAR USUARIO

Al darle en “EDITAR” se despliega la interfaz para modificar el usuario en este caso el user4

Nombre Usuario Password Tipo					Logout
diego	user1	123	1	<a href="#">EDITAR</a>	<a href="#">ELIMINAR</a>
sara	user2	123	3	<a href="#">EDITAR</a>	<a href="#">ELIMINAR</a>
samir	user4	123	3	<a href="#">EDITAR</a>	<a href="#">ELIMINAR</a>

4:36 PM

## EDITAR USUARIO

Nombre

samir

Username

user4

Password

123



Tipo

3

**GUARDAR CAMBIOS**

**CANCELAR**

■ ◎ ◀

4:36 PM

## EDITAR USUARIO

Nombre

samir

Username

user4

Password

123



Tipo

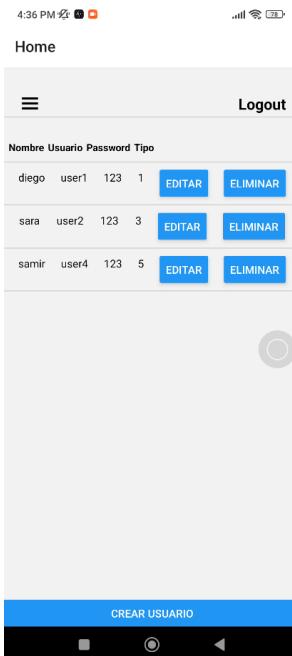
5

**GUARDAR CAMBIOS**

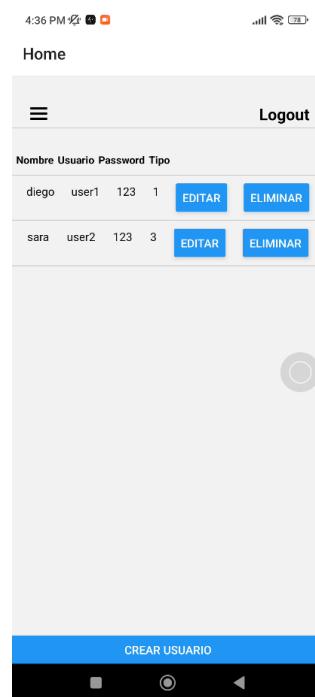
**CANCELAR**

■ ◎ ◀

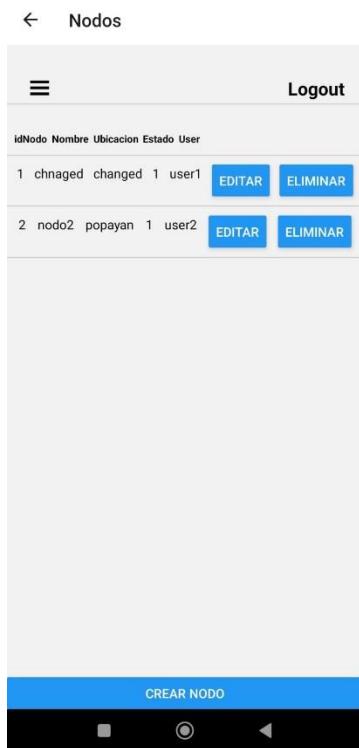
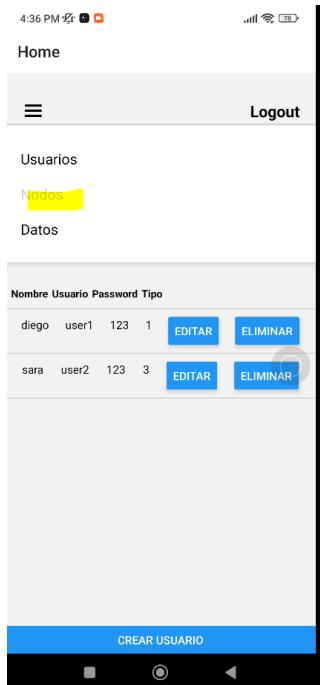
Confirmando los cambios con “GUARDAR CAMBIOS” se muestra efectivamente el cambio en el apartado usuarios , en este caso de modiflico el tipo en donde antes era 3 ahora será 5



Y para eliminar el usuario solo darle al botón “ELIMINAR” , en este caso se eliminara el user4



Pasando al apartado de nodos , este se puede crear nodos , modificar y borrar



Al darle "CREAR NODO" se muestra la interfaz con diferentes parámetros para crear el nodo como lo son idnodo, nombre, ubicación ,estado y user

### CREAR NODO

idNodo

Nombre

Ubicacion

Estado

User

CREAR

CANCELAR

■ ◎ ◀

### CREAR NODO

idNodo

Nombre

Ubicacion

Estado

User

CREAR

CANCELAR

■ ◎ ◀

*Para confirmación la creación del nodo dar en “CREAR”*

← Nodos

Nodos				
idNodo	Nombre	Ubicacion	Estado	User
1	chnaged	changed	1	user1
2	nodo2	popayan	1	user2
3	nametest	popayan	1	user8

CREAR NODO

Al darle “EDITAR” se despliega la interfaz para modificar el nodo

### EDITAR NODO

idNodo	<input type="text" value="3"/>
Nombre	<input type="text" value="nametest"/>
Ubicacion	<input type="text" value="popayan"/>
Estado	<input type="text" value="1"/>
User	<input type="text" value="user8"/>

**GUARDAR CAMBIOS**

**CANCELAR**

En este caso se editará el user8 a user7

## EDITAR NODO

idNodo

3

Nombre

nametest

Ubicacion

popayan

Estado

1

User

user7

**GUARDAR CAMBIOS**

**CANCELAR**

The screenshot shows a mobile application interface. At the top, there is a navigation bar with icons for signal strength, battery level, and other status indicators. Below the navigation bar, the title "Nodos" is displayed next to a back arrow icon. On the right side of the title, there are "Logout" and a menu icon (three horizontal lines). The main content area displays a table of nodes with columns: idNodo, Nombre, Ubicacion, Estado, and User. Each row contains a node with its details and two buttons: "EDITAR" and "ELIMINAR". The first node has the following values: idNodo: 1, Nombre: chnaged, Ubicacion: changed, Estado: 1, User: user1. The second node has: idNodo: 2, Nombre: nodo2, Ubicacion: popayan, Estado: 1, User: user2. The third node has: idNodo: 3, Nombre: nametest, Ubicacion: popayan, Estado: 1, User: user7. At the bottom of the screen, there is a blue button labeled "CREAR NODO". The bottom of the screen also features a black navigation bar with standard Android icons for back, home, and recent apps.

idNodo	Nombre	Ubicacion	Estado	User
1	chnaged	changed	1	user1
2	nodo2	popayan	1	user2
3	nametest	popayan	1	user7

**CREAR NODO**

Dando al botón “ELIMINAR” se eliminará el nodo en este caso se eliminará el idnodo 3

← Nodos

idNodo	Nombre	Ubicacion	Estado	User
1	chnaged changed	1	user1	<a href="#">EDITAR</a> <a href="#">ELIMINAR</a>
2	nodo2	popayan	1	user2

CREAR NODO

Dando en el apartado de datos

← Nodos

Logout

Usuarios

Nodos

Datos

idNodo	Nombre	Ubicacion	Estado	User
1	chnaged changed	1	user1	<a href="#">EDITAR</a> <a href="#">ELIMINAR</a>
2	nodo2	popayan	1	user2

4:37 PM 🔋 🔋 🔋

← Datos



Logout

### Selecciona el visualizador de Datos

Tabla de Datos

Card datos

Grafica Temperatura

Grafica Humedad



Seleccionando la tabla de datos muestra todos los datos del id , idnodo , temperatura , humedad y fecha en una tabla

4:37 PM 🔋 🔋 🔋

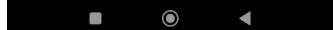
← TableData



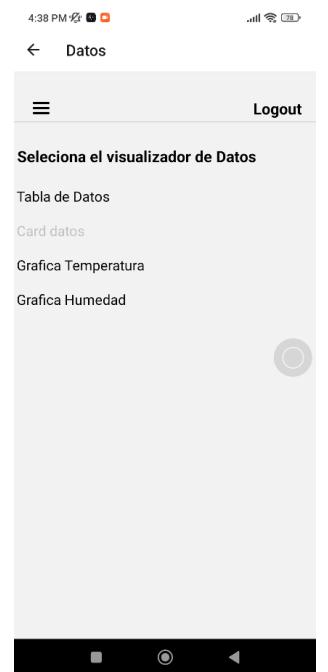
Logout

ID IDNODO TEMPERATURA HUMEDAD FECHA

1	1	24.1	59	2023-10-30T04:12:09.000Z
2	1	24.1	59	2023-10-30T04:12:14.000Z
3	1	24.2	59	2023-10-30T04:12:20.000Z
4	1	24.2	59	2023-10-30T04:12:25.000Z
5	1	24.2	59	2023-10-30T04:12:30.000Z
6	1	24.3	59	2023-10-30T04:12:36.000Z
7	2	25.8	62	2023-11-11T23:04:04.000Z
8	2	26.2	61	2023-11-11T23:04:23.000Z
9	2	26.2	61	2023-11-11T23:04:29.000Z
10	2	26.2	61	2023-11-11T23:04:35.000Z
11	2	26.2	63	2023-11-12T01:02:46.000Z
12	2	26.2	62	2023-11-12T01:03:46.000Z
13	2	26.2	62	2023-11-12T01:03:51.000Z
14	2	26.2	62	2023-11-12T01:03:56.000Z
15	2	26.2	62	2023-11-12T01:04:07.000Z



## Seleccionando Card datos



Se debe ingresar el id donde se quiere ver la fecha , temperatura y humedad que tiene , en este caso se introdujo el id 1y dar en “VER ID”

4:38 PM

← CardData

## DATO ESPECIFICO

Ingrese un número ID:

1

VER ID

2023-10-30T04:12:09.000Z

TEMPERATURA      HUMEDAD  
24.1                59

VER GRÁFICA

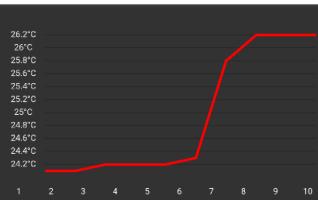
VER GRÁFICA



En donde hay dos botones de ver grafica en donde el izquierdo lado de la temperatura mostrara la grafica de todos los datos de la temperatura y el lado derecho de la humedad mostrara la grafica de todos los datos de la humedad. En este caso se seleccionara la grafica de temperatura

4:38 PM

← GraphTemperature



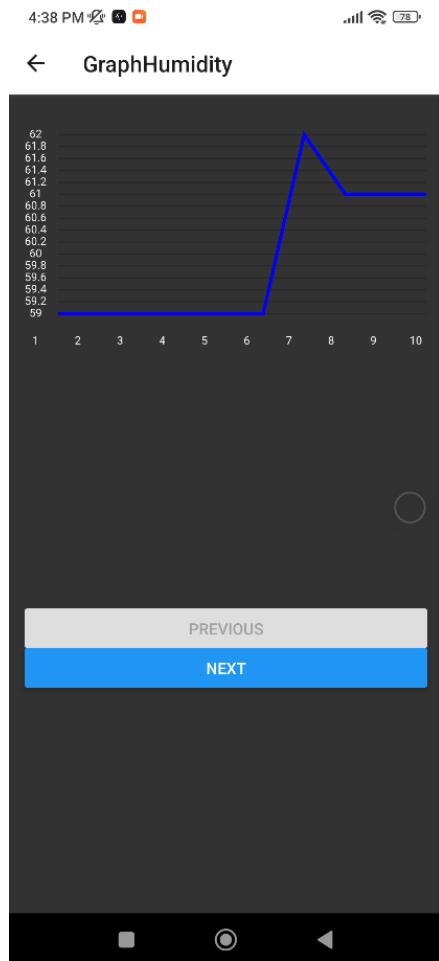
PREVIOUS

NEXT

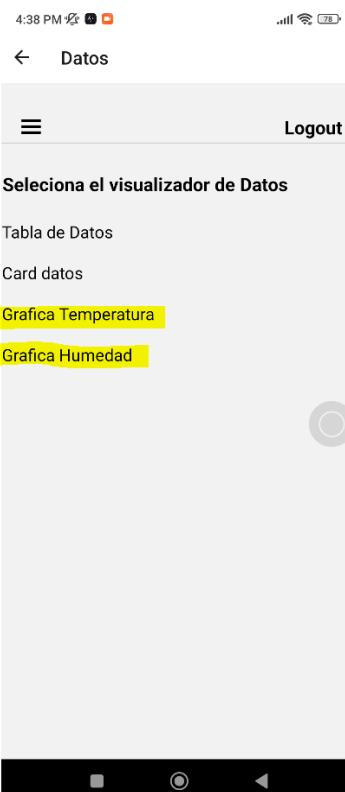


En donde se observa los datos de la temperatura en °C en el eje y y el eje x esta el id y para ver mas datos se le da NEXT en donde se ve los otros 10 hasta completar todos los datos y con el PREVIOUS retrocederá 10 datos hasta iniciar con los 10 primeros datos.

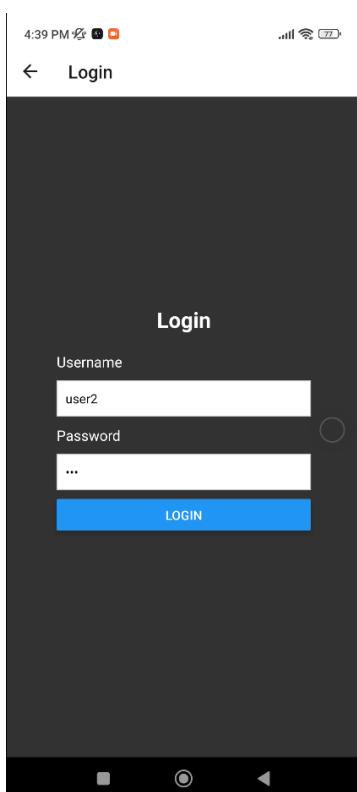
Ahora con la grafica de humedad



Todas estas graficas de temperatura y humedad se pueden ver también en la selección de Datos



Ahora para el otro usuario cliente que es de diferente tipo 1 , en este caso el user2



Donde después de ingresar se visualizará una tabla de los datos de temperatura, humedad y fecha , según el idnodo asignado al usuario , en este caso el idnodo asignado es el 2 , por lo que solo vera los datos del idnodo 2

The screenshot shows a mobile application interface titled "Client". At the top, there are status icons for battery level, signal strength, and connectivity. Below the title is a "Logout" button. The main content is a table with the following columns: IDNODO, TEMPERATURA, HUMEDAD, and FECHA. The data is as follows:

IDNODO	TEMPERATURA	HUMEDAD	FECHA
2	25.8	62	2023-11-11T23:04:04.000Z
2	26.2	61	2023-11-11T23:04:23.000Z
2	26.2	61	2023-11-11T23:04:29.000Z
2	26.2	61	2023-11-11T23:04:35.000Z
2	26.2	63	2023-11-12T01:02:46.000Z
2	26.2	62	2023-11-12T01:03:46.000Z
2	26.2	62	2023-11-12T01:03:51.000Z
2	26.2	62	2023-11-12T01:03:56.000Z
2	26.2	62	2023-11-12T01:04:07.000Z
2	26.2	61	2023-11-12T01:04:12.000Z
2	26.2	61	2023-11-12T01:04:17.000Z
2	26.2	61	2023-11-12T01:04:23.000Z
2	26.2	61	2023-11-12T01:04:29.000Z
2	25.8	61	2023-12-04T03:01:42.000Z
2	25.8	61	2023-12-04T03:01:49.000Z
2	25.8	61	2023-12-04T03:02:01.000Z

## Bibliografía

[1]"Capítulo 3. Almacenamiento", *Fao.org*, 2022. [Online]. Available: <https://www.fao.org/3/y4893s/y4893s06.htm>. [Accessed: 22- Aug- 2022]

[2]"Capítulo 35: Mejoramiento de la seguridad alimentaria en el hogar", *Fao.org*, 2022. [Online]. Available: <https://www.fao.org/3/w0073s/w0073s13.htm>. [Accessed: 22- Aug- 2022]

[3]*Procolombia.co*, 2022. [Online]. Available: [https://procolombia.co/sites/all/modules/custom/mccann/mccann\\_ruta\\_exportadora/files/06-cartilla-cadena-frio.pdf](https://procolombia.co/sites/all/modules/custom/mccann/mccann_ruta_exportadora/files/06-cartilla-cadena-frio.pdf). [Accessed: 22- Aug- 2022]

[4]"Consumo y producción sostenibles - Desarrollo Sostenible", *Desarrollo Sostenible*, 2022. [Online]. Available: <https://www.un.org/sustainabledevelopment/es/sustainable-consumption-production/>. [Accessed: 22- Aug- 2022]

[5]"Hambre y seguridad alimentaria - Desarrollo Sostenible", *Desarrollo Sostenible*, 2022. [Online]. Available: <https://www.un.org/sustainabledevelopment/es/hunger/>. [Accessed: 22- Aug- 2022]

[6]"Sensor de temperatura, escoge el mejor para tus proyectos con Arduino", *Programar fácil con Arduino*, 2022. [Online]. Available: <https://programarfacil.com/podcast/82-escoger-mejor-sensor-temperatura-arduino/>. [Accessed: 22- Aug- 2022]

[7]"HDC1080 Arduino y ESP8266 sensor de temperatura y humedad", *Programar fácil con Arduino*, 2022. [Online]. Available: <https://programarfacil.com/blog/arduino-blog/hdc1080-arduino-esp8266/>. [Accessed: 22- Aug- 2022]

[8]"Sensor de Humedad Relativa HS1101", *VISTRONICA S.A.S.*, 2022. [Online]. Available: <https://www.vistronica.com/sensores/sensor-de-humedad-relativa-hs1101-detail.html>. [Accessed: 22- Aug- 2022]

[9]"0.9 £ 24% de DESCUENTO|Higrómetro con Sensor de humedad, resistencia sensible, módulo con funda, HR31 HR31D HR202 HR202L HJ3180B HDS10, 5 piezas|Sensores| - AliExpress", *aliexpress.com*, 2022. [Online]. Available: [https://es.aliexpress.com/item/32951082181.html?spm=a2g0o.productlist.0.0.4f462a48CJN2YE&algo\\_pvid=3bcceb09-5679-4253-86c8-80d8cd0ccbec&algo\\_exp\\_id=3bcceb09-5679-4253-86c8-80d8cd0ccbec-16&pdp\\_ext\\_f=%7B%22sku\\_id%22%3A%2212000027223692462%22%7D&pdp\\_npi=2%40dis%21COP%217438.98%215646.45%21%21%2117970.06%21%21%402101d64d16610413989422613e8d0e%2112000027223692462%21sea&curPageLogUid=p6va0gBBweOo](https://es.aliexpress.com/item/32951082181.html?spm=a2g0o.productlist.0.0.4f462a48CJN2YE&algo_pvid=3bcceb09-5679-4253-86c8-80d8cd0ccbec&algo_exp_id=3bcceb09-5679-4253-86c8-80d8cd0ccbec-16&pdp_ext_f=%7B%22sku_id%22%3A%2212000027223692462%22%7D&pdp_npi=2%40dis%21COP%217438.98%215646.45%21%21%2117970.06%21%21%402101d64d16610413989422613e8d0e%2112000027223692462%21sea&curPageLogUid=p6va0gBBweOo). [Accessed: 22- Aug- 2022]

[10]*Youtube.com*, 2022. [Online]. Available: [https://www.youtube.com/watch?v=SKg\\_4ggqz2U](https://www.youtube.com/watch?v=SKg_4ggqz2U). [Accessed: 22- Aug- 2022]

[11]*Youtube.com*, 2022. [Online]. Available: [https://www.youtube.com/watch?v=mlJxILi\\_xds](https://www.youtube.com/watch?v=mlJxILi_xds). [Accessed: 22- Aug- 2022]

[12]"ESP32: Date and Time (NTP Client)", *Phatiphatt Thounthong*, 2022. [Online]. Available: <https://phatiphatt.wordpress.com/esp32-date-and-time-ntp-client/>. [Accessed: 22- Aug- 2022]

[13]"PlatformIO Registry", *PlatformIO Registry*, 2022. [Online]. Available: <https://registry.platformio.org/libraries/paulstoffregen/Time>. [Accessed: 22- Aug- 2022]