Arduino y ESP32
IOT

Segunda parte WiFi y Node-Red

TOPICOS PREVIOS SENSORISTICA

ARDUINO Y TECNOLOGÍAS INALÁMBRICAS PARA IOT

Tutor: L. A. Ángeles-Hurtado

QUERÉTARO, ENERO 2024

0.1 Cofiguración estática en esp32-c6

En la siguiente imagen se puede ver como queda el panel en el que se puede ver gráficamente y con dos identificadores indicadores uno con datos en HEX y otro con datos en mili volts, además de dos leds a modo de muestra de lo que se puede agregar al panel. Node-red permite agregar una gran cantidad de nodos que nos ayudan a visualizar por medio de un servidor local. Una vez que exportes el proyecto vas a poder acceder con la siguiente dirección http://localhost:1880/ui/. Véase los códigos en el repositorio con nombres sketch_wifi01.ino y flow_wifi01.json

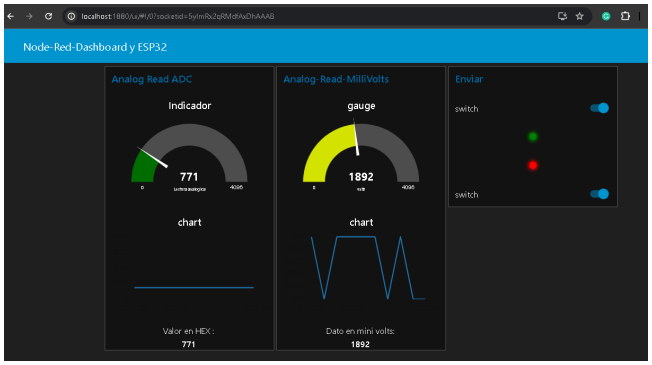

Figure 1: Para la segunda etapa se diseño un panel para visualizar los datos obtenidos por el ADC de la esp32-c6

Para importar el código realiza la siguiente acción.

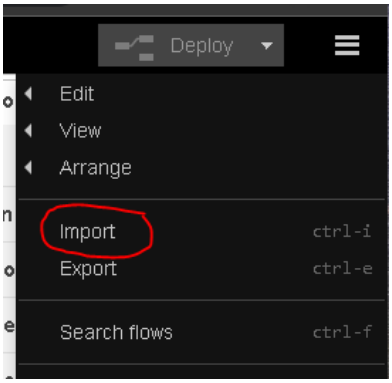

Figure 2: Selecciona importar y después busca el archivo a importar flow_wifi01.json

```
[
    {
        "id": "03da84c4eac89217",
        "type": "tab",
        "label": "eje04",
        "disabled": false,
        "info": "",
```

```
8           "env": []
9       },
```

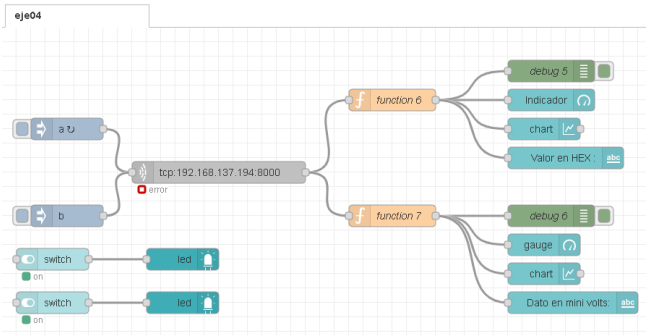

Figure 3: Si realizas correctamente la exportación del archivo flow_wifi01.json podras visualizar los nodos que se agregaron

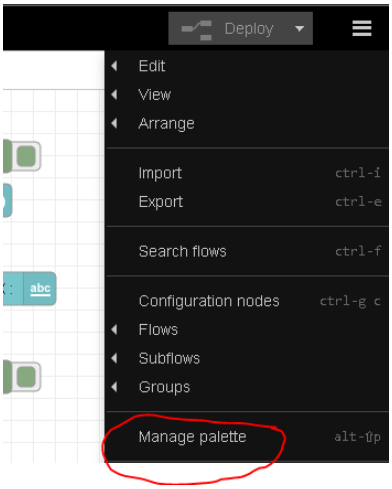

Figure 4: Posteriormente tienes que descargar el dasboard, da click en configuración y después Manage palette

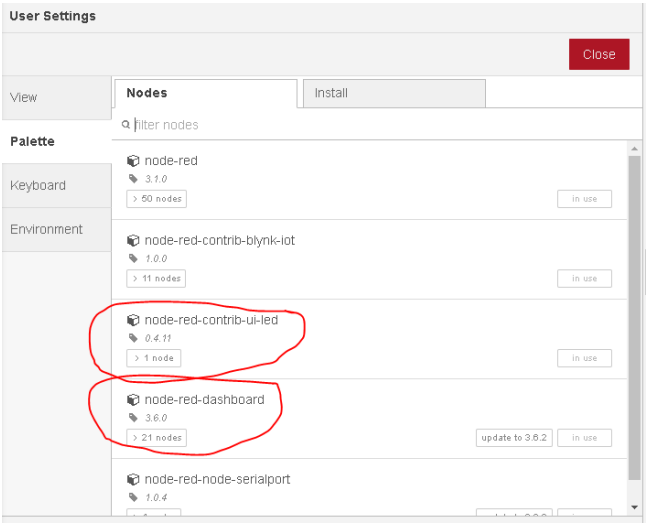

Figure 5: Si no tienes instalados estos nodos ve a la sección de install y escribe los nombres tal y como aparecen en la imagen

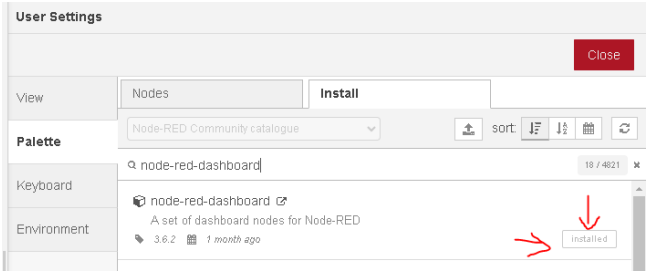

Figure 6: node-red-dashboard

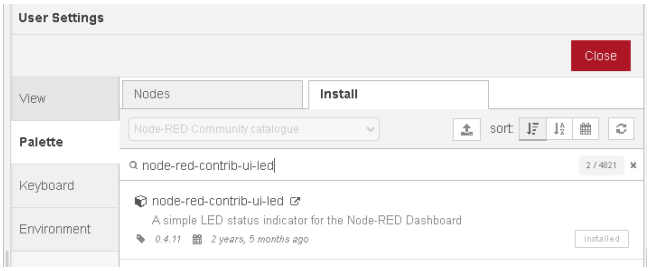

Figure 7: node-red-contrib-ui-led

El siguiente código se compilo en el IDE de Arduino para la esp32-c6 es importante que se verifique el modelo de la esp32 porque pueden cambiar algunos metodos por ejemplo para leer el ADC. Para que en conjunto con Node-Red se comuniquen por medio del protocolo TCP y se puedan visualizar los datos en un servidor local.

```
1  void loop() {
2
3    // para escuchar las peticiones del cliente se crea un objeto de WiFiClient
```

```
4     WiFiClient cliente = server.available(); // detecta al cliente que en este ejemplo es
    Node-Red
5     if (cliente) {
6       while(cliente.connected()){
7         if(cliente.available()){
8           char c = cliente.read();
9
10          String stringZero = String(analogRead(0), HEX);          // using an int and a
    base (hexadecimal)
11          String stringOne =  String(analogReadMilliVolts(0), DEC); // using an int and a
    base
12          String stringTWO =  String(stringZero + "," + stringOne); // concatenating
    strings
13
14          Serial.println(c);
15          Serial.println(stringTWO);
16
17          if(c == 'a'){
18            cliente.print(stringTWO);
19          }else {
20            cliente.print("Conexión correcta con la ESP32 ");
21          }
22        }
23      }
24    }
25    //---------------------------------------
26    if(WiFi.status() == WL_DISCONNECTED){
27      Serial.println("WiFi desconectado ");
28      delay(5000);
29    }
30    //---------------------------------------
31  }
```