Parte 3: Construção de um Painel de Monitoramento

Neste tutorial vamos criar uma aplicação Node-RED que irá permitir realizar o monitoramento de um conjunto de dados enviados por sensores através de um broker MQTT.

Este projeto tem o intuito de demonstrar mais algumas possibilidade do dashboard do Node-RED. Vamos desenvolver um sistema simples de monitoramento a partir de um conjunto de dados obtidos dos sensores BMP180 e BH1750 e enviados através do broker MQTT. Para a montagem do circuito eletrônico, identifique e separe os seguintes materiais.

Relação de materiais

- 1 NodeMCU (ESP8266 ou ESP32).
- 2 Resistores de 10 kOhms (marrom, preto, laranja).
- 1 Sensor BMP180.
- 1 Sensor BH1750.
- 1 Protoboard.
- Cabos de ligação.

Realize a montagem da maneira indicada pela Figura 1, caso esteja utilizando o NodeMCU (ESP8266).

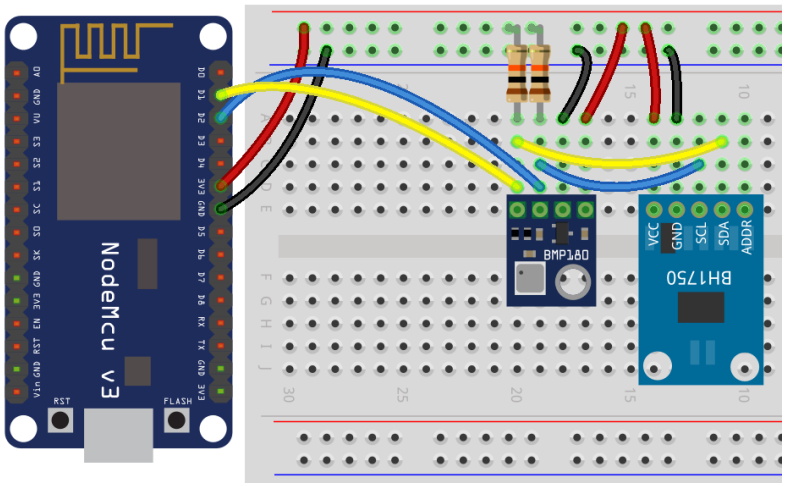

Figura 1: Conexões (NodeMCU-ESP8266)

Adote como referência a Figura 2, se estiver utilizando o NodeMCU (ESP32).

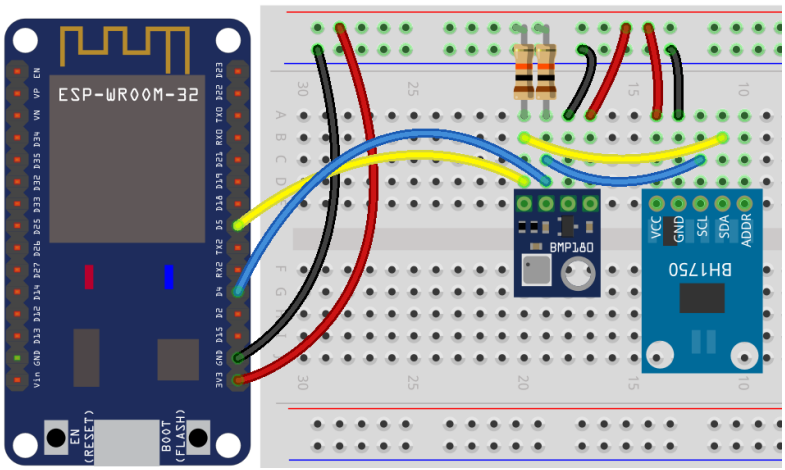

Figura 2: Conexões (NodeMCU-ESP32)

Em seguida, implemente o seguinte programa no ambiente de desenvolvimento Thonny.

```
from machine import Pin, I2C
from time import sleep
from bmp180 import BMP180
from bh1750 import BH1750
import network
from umqtt.simple import MQTTClient
import gc
gc.collect()

print('Iniciando...')
i2c = I2C(sda=Pin(5), scl=Pin(4))
bmp = BMP180(i2c)
bmp.oversample_sett = 2
bmp.baseline = 101325
bh = BH1750(i2c)

estacao = network.WLAN(network.STA_IF)
estacao.active(True)
estacao.connect('ap', 'senha')
while estacao.isconnected() == False:
  pass
print('Conexao realizada.')
print(estacao.ifconfig())

servidor = 'test.mosquitto.org'
topico = 'sensor/bmp_bh'
cliente = MQTTClient('NodeMCU', servidor, 1883)

try:
  while True:
    temperatura = int(bmp.temperature)
    pressao = int(bmp.pressure / 100)
    altitude = int(bmp.altitude)
    luminosidade = int(bh.luminance(BH1750.ONCE_HIRES_1))
    conteudo = '{"temperatura" : ' + str(temperatura) + ', "pressao" : ' + str(pressao)
+ ', "altitude" : ' + str(altitude) + ', "luminosidade" : ' + str(luminosidade) + '}'
    print('Publicando no servidor MQTT')
    cliente.connect()
    cliente.publish(topico.encode(), conteudo.encode())
    cliente.disconnect()
    print ('Envio realizado.')
    gc.collect()
    sleep(60.0)
except KeyboardInterrupt:
  estacao.disconnect()
  estacao.active(False)
  print("Fim.")
```

mqtt-bmp180-bh1750.py

Execute o programa criado em MicroPython e, na sequência, passamos para o desenvolvimento da aplicação no Node-RED. Conforme ilustra a Figura 3, realize a criação de um fluxo, insira e conecte os nós **mqtt in** e **debug**.

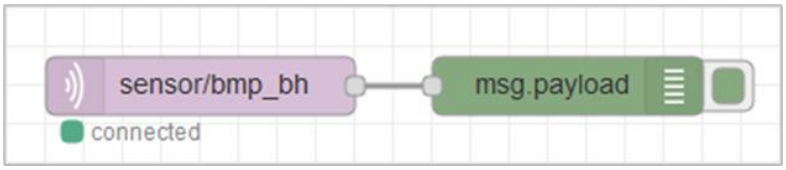

Figura 3: Leitura dos Dados do Broker MQTT

Realize um clique duplo o mouse sobre o nó **mqtt in** para entrar em modo de edição. Realize a configuração do nó especificando o servidor e o tópico, conforme podemos notar na Figura 4. É importante salientar que tanto o servidor quanto o tópico deverão ser os mesmos usados no programa em MicroPython.

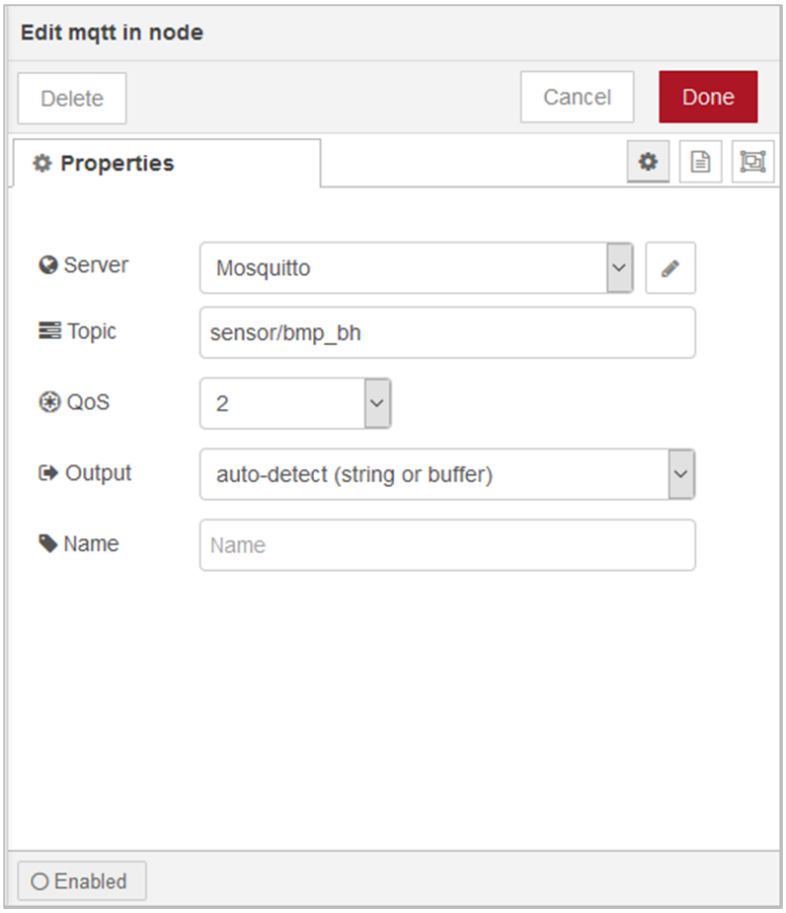

*Figura 4: Configuração do Nó **mqtt in***

No Node-RED Clique no botão **Deploy** para executar o fluxo, os dados deverão se exibidos no painel debug de maneira similar aos mostrados no Figura 5.

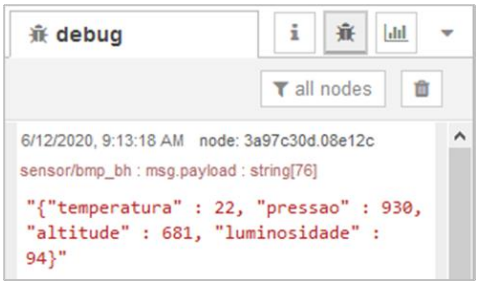

Figura 5: Dados Obtidos

Uma vez que certificamos que os dados já são obtidos do broker MQTT, iremos realizar a criação da aplicação que usará o módulo dashboard para criar uma representação gráfica. O primeiro passo consiste em excluir o nó **debug** e depois insirir um nó **json** e quatro nós **gauge**. Realize a conexão deles conforme indica a Figura 6.

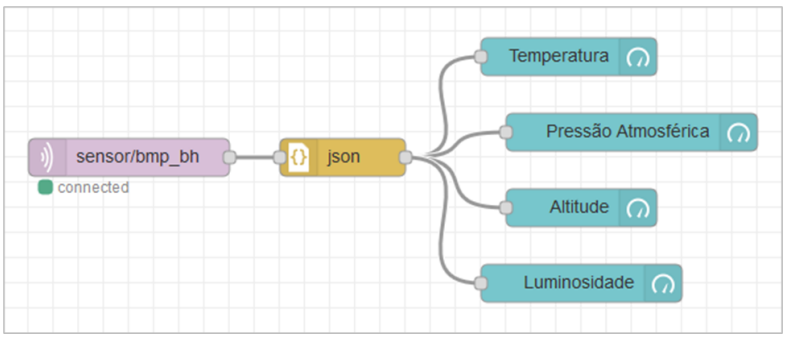

Figura 6: Criação do Dashboard

O nó **json** não precisa ser configurado e ele será responsável pela conversão dos dados recebidos em objetos JSON. Na sequência realize um clique duplo sobre o nó **gauge** que será usado para exibir da temperatura e crie um grupo. Para isso, selecione na propriedade **Group** a opção “Add new ui_group...” e clique no botão de edição (ícone do Lápis).

Na Figura 7 devemos notar que a propriedade **Name** deve receber o texto “Sistema de Monitoramento” enquanto a propriedade **Width** irá receber o valor “12”. Pressione o botão **Update** após concluir a edição do grupo.

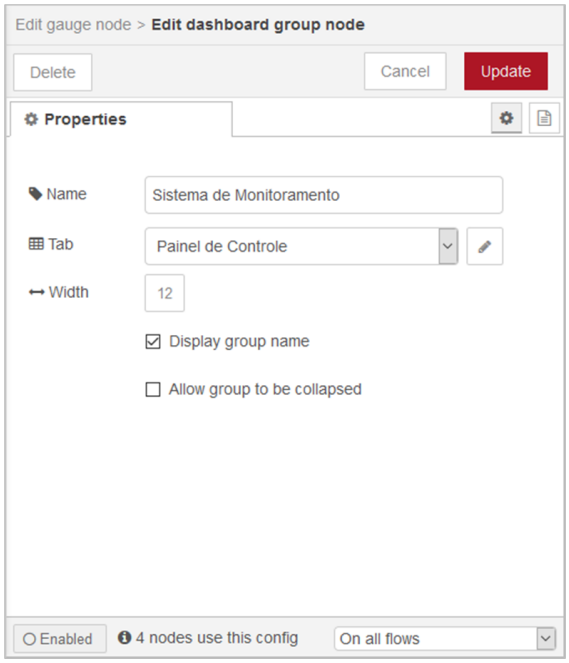

Figura 7: Criação do Grupo

Retornando a edição do nó **gauge**, ajuste as seguintes propriedades (Figura 8):

- Size: “6 x 4”.
- Label: “Temperatura”.
- Value format: “{{payload.temperatura}}” .
- Units: “°C”.
- Range min: “0”.
- Range max: “100”.

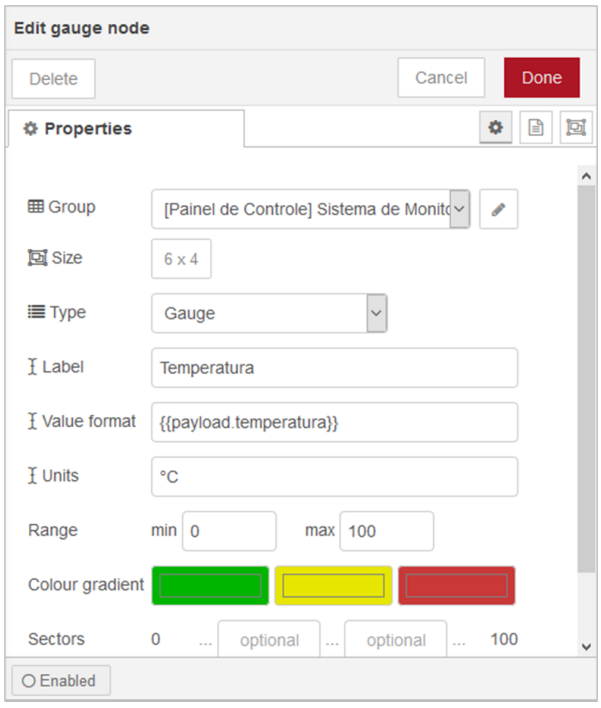

*Figura 8: Edição do Nó **gauge***

Repita este processo para os outros três nós **gauge**, ajustando devidamente os valores dos atributos, sendo que a unidade de pressão atmosférica é hPa e a faixa de valores está entre 300 e 1.100, altitude é expressa em metros com valores entre -1.000 e 9.000 e, por último a luminosidade é expressa em lux com valores entre 0 e 10.000.

Clique no botão **Deploy** para executar o fluxo. Abra uma nova aba no navegador e digite a URL da aplicação. Na Figura 9 apresentamos a exibição do painel de controle (dashboard) que foi criado.

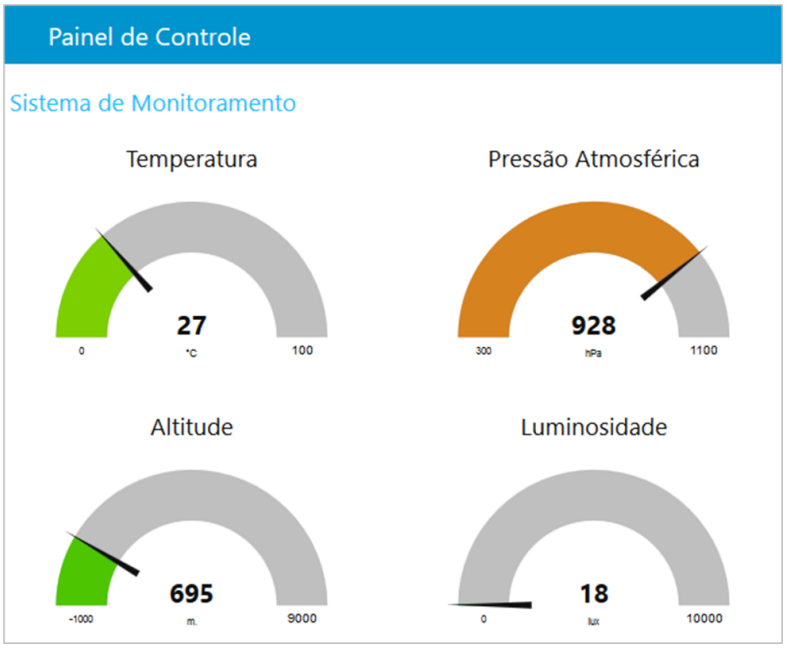

Figura 9: Painel de Controle