SMARTBRIDGE EXTERNSHIP

Internet Of Things

Date: 07th June 2023

Jyoti Prakash Behura
20BCE7355
VIT-AP

Assignment 1 *: in wokwi add LED and switch on and off from node-red*

Code:

sketch.ino

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#include "DHT.h"// Library for dht11
#define DHTPIN 15     // what pin we're connected to
#define DHTTYPE DHT22   // define type of sensor DHT 11
#define LED 2

DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and typr of
dht connected

void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);

//-------credentials of IBM Accounts------

#define ORG "9f8wlx"//IBM ORGANITION ID
#define DEVICE_TYPE "abcd"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "1234"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //Token
String data3;
float h, t;


//-------- Customise the above values --------
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of
event perform and format in which data to be send
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd  REPRESENT
command type AND COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id
```

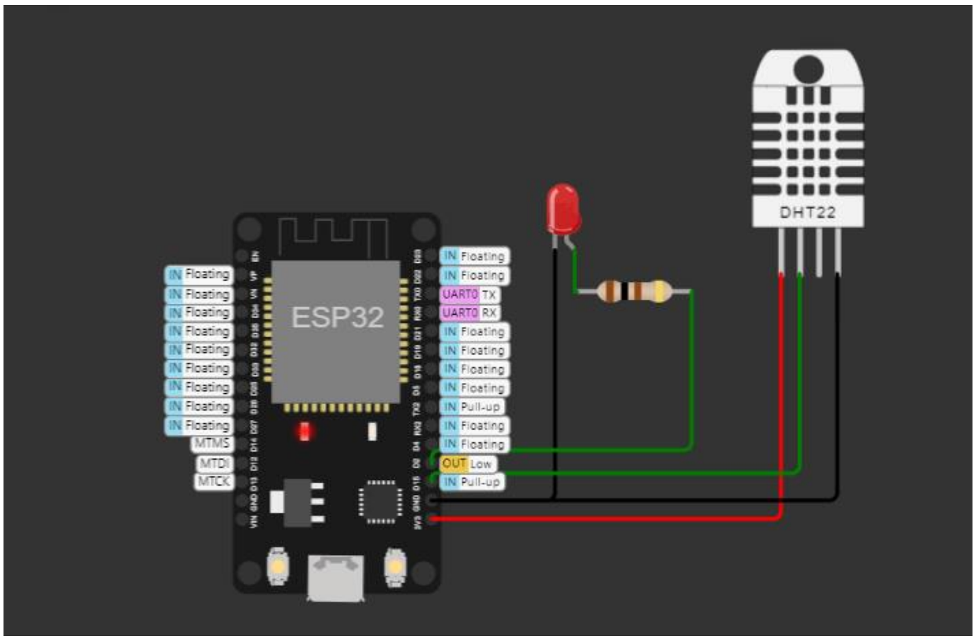
```
//-----------------------------------------
WiFiClient wifiClient; // creating the instance for wificlient
PubSubClient client(server, 1883, callback ,wifiClient); //calling the
predefined client id by passing parameter like server id,portand
wificredential


void setup()// configureing the ESP32
{
  Serial.begin(115200);
  dht.begin();
  pinMode(LED,OUTPUT);
  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
}

void loop()// Recursive Function
{

  h = dht.readHumidity();
  t = dht.readTemperature();
  Serial.print("temp:");
  Serial.println(t);
  Serial.print("Humid:");
  Serial.println(h);

  PublishData(t, h);
  delay(1000);
  if (!client.loop()) {
    mqttconnect();
  }
}



/*.....................................retrieving to
Cloud...............................*/

void PublishData(float temp, float humid) {
  mqttconnect();//function call for connecting to ibm
  /*
     creating the String in in form JSon to update the data to ibm cloud
  */
  String payload = "{\"temp\":";
  payload += temp;
  payload += "," "\"Humid\":";
  payload += humid;
  payload += "}";


  Serial.print("Sending payload: ");
  Serial.println(payload);


  if (client.publish(publishTopic, (char*) payload.c_str())) {
    Serial.println("Publish ok");// if it sucessfully upload data on the cloud
then it will print publish ok in Serial monitor or else it will print publish
failed
  } else {
    Serial.println("Publish failed");
  }

}


void mqttconnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting client to ");
    Serial.println(server);
    while (!!!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
      delay(500);
    }

     initManagedDevice();
     Serial.println();
  }
}
void wificonnect() //function defination for wificonnect
{
  Serial.println();
  Serial.print("Connecting to ");

  WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish
the connection
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
```

```
void initManagedDevice() {
  if (client.subscribe(subscribetopic)) {
    Serial.println((subscribetopic));
    Serial.println("subscribe to cmd OK");
  } else {
    Serial.println("subscribe to cmd FAILED");
  }
}

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
{

  Serial.print("callback invoked for topic: ");
  Serial.println(subscribetopic);
  for (int i = 0; i < 7; i++) {
    //Serial.print((char)payload[i]);
    data3 += (char)payload[i];
  }
  Serial.println("data: "+ data3);
  if(data3=="lighton")
  {
Serial.println(data3);
digitalWrite(LED,HIGH);
  }
  else
  {
Serial.println(data3);
digitalWrite(LED,LOW);
  }
data3="";
}
```

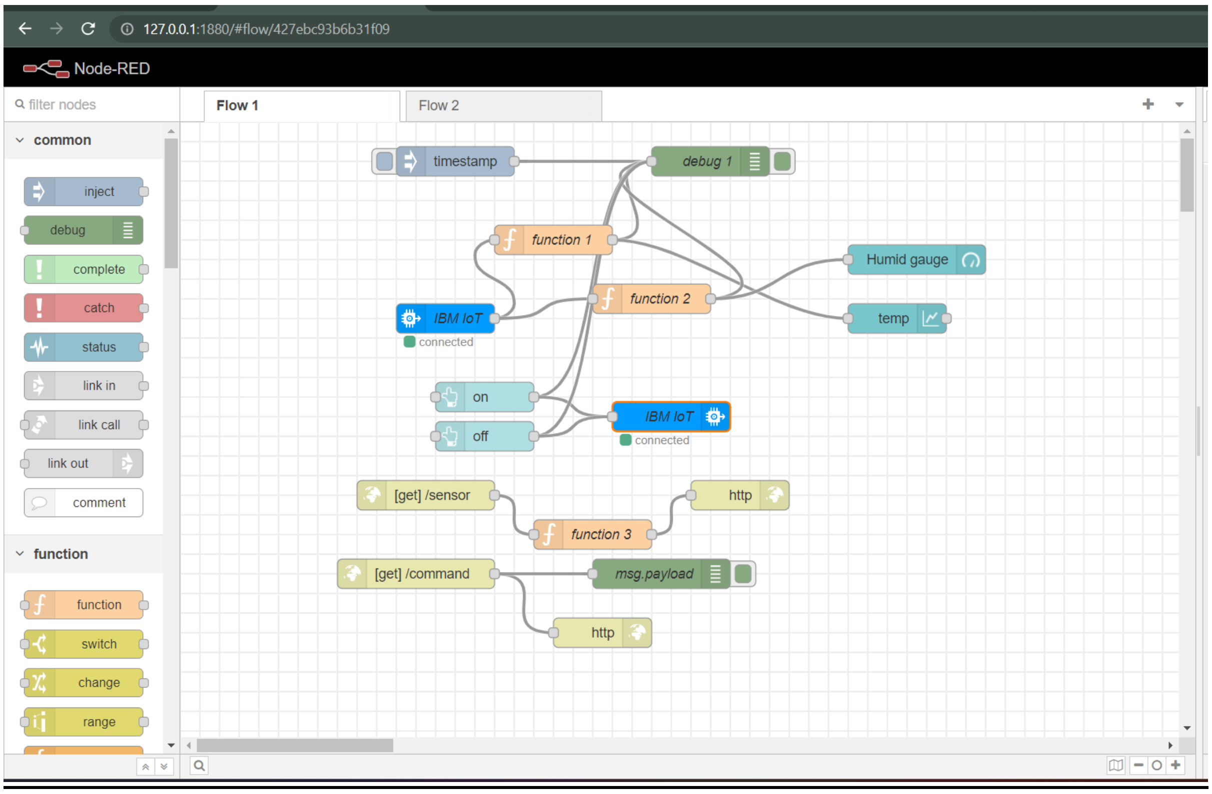
diagram.json

```
{
  "version": 1,
  "author": "JYOTI PRAKASH BEHURA 20BCE7355"

  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": 4.8, "left": -
127.69, "attrs": {} },
    { "type": "wokwi-dht22", "id": "dht1", "top": -76.72, "left": 137.76,
"attrs": {} },
    {
      "type": "wokwi-led",
      "id": "led1",
      "top": -16.04,
      "left": 21.83,
      "attrs": { "color": "red" }
    },
    {
      "type": "wokwi-resistor",
      "id": "r1",
      "top": 41.63,
      "left": 48.17,
      "attrs": { "value": "100" }
    }
  ],
  "connections": [
    [ "esp:TX0", "$serialMonitor:RX", "", [] ],
    [ "esp:RX0", "$serialMonitor:TX", "", [] ],
    [ "dht1:VCC", "esp:3V3", "red", [ "v0" ] ],
    [ "dht1:GND", "esp:GND.1", "black", [ "v0" ] ],
    [ "led1:A", "r1:1", "green", [ "v0" ] ],
    [ "led1:C", "esp:GND.1", "black", [ "v0" ] ],
    [ "dht1:SDA", "esp:D15", "green", [ "v101.76", "h-2.06" ] ],
    [ "r1:2", "esp:D2", "green", [ "v80.85", "h-3.49" ] ]
  ]
}
```

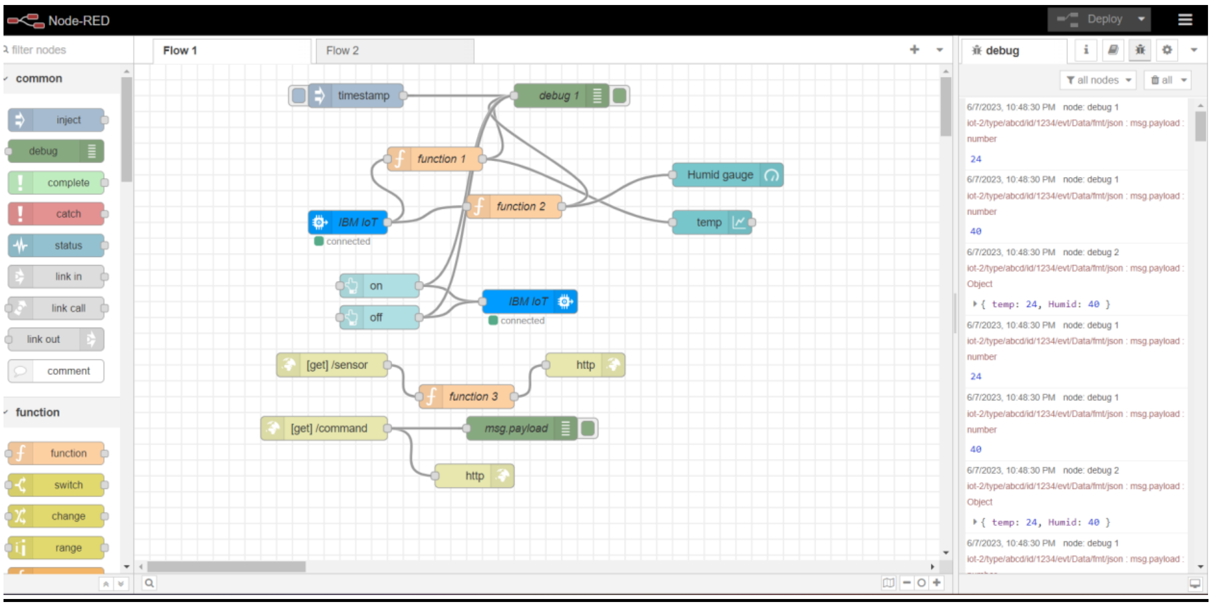
Diagram:



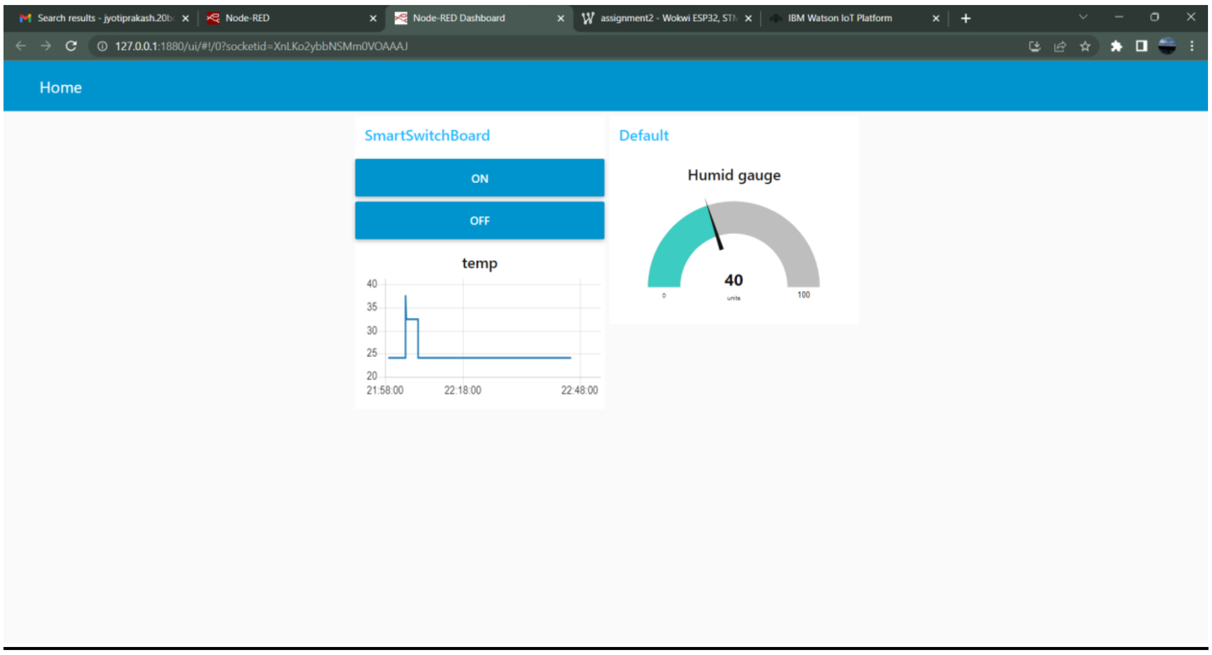
Node-Red Diagram:

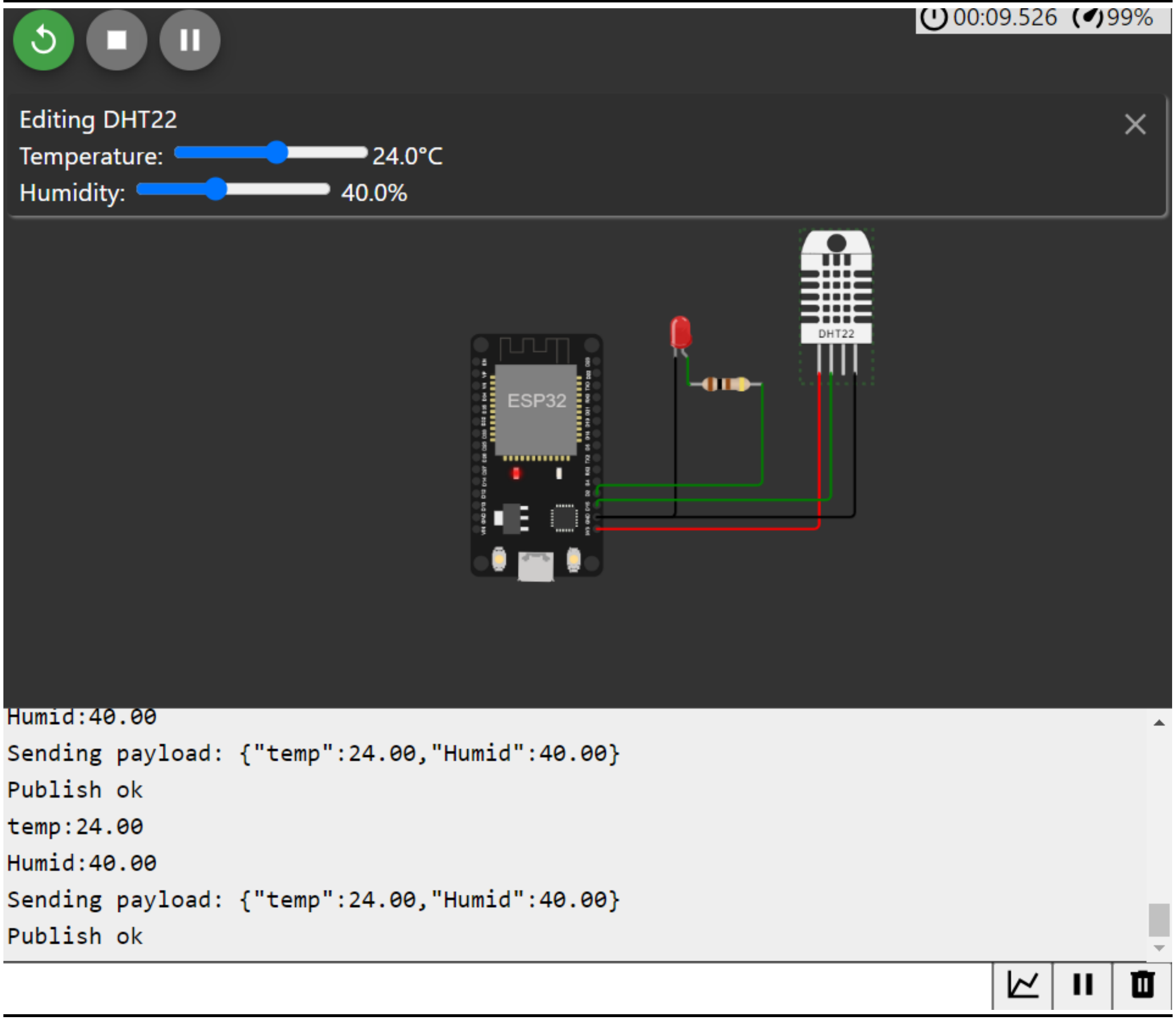


Outputs:



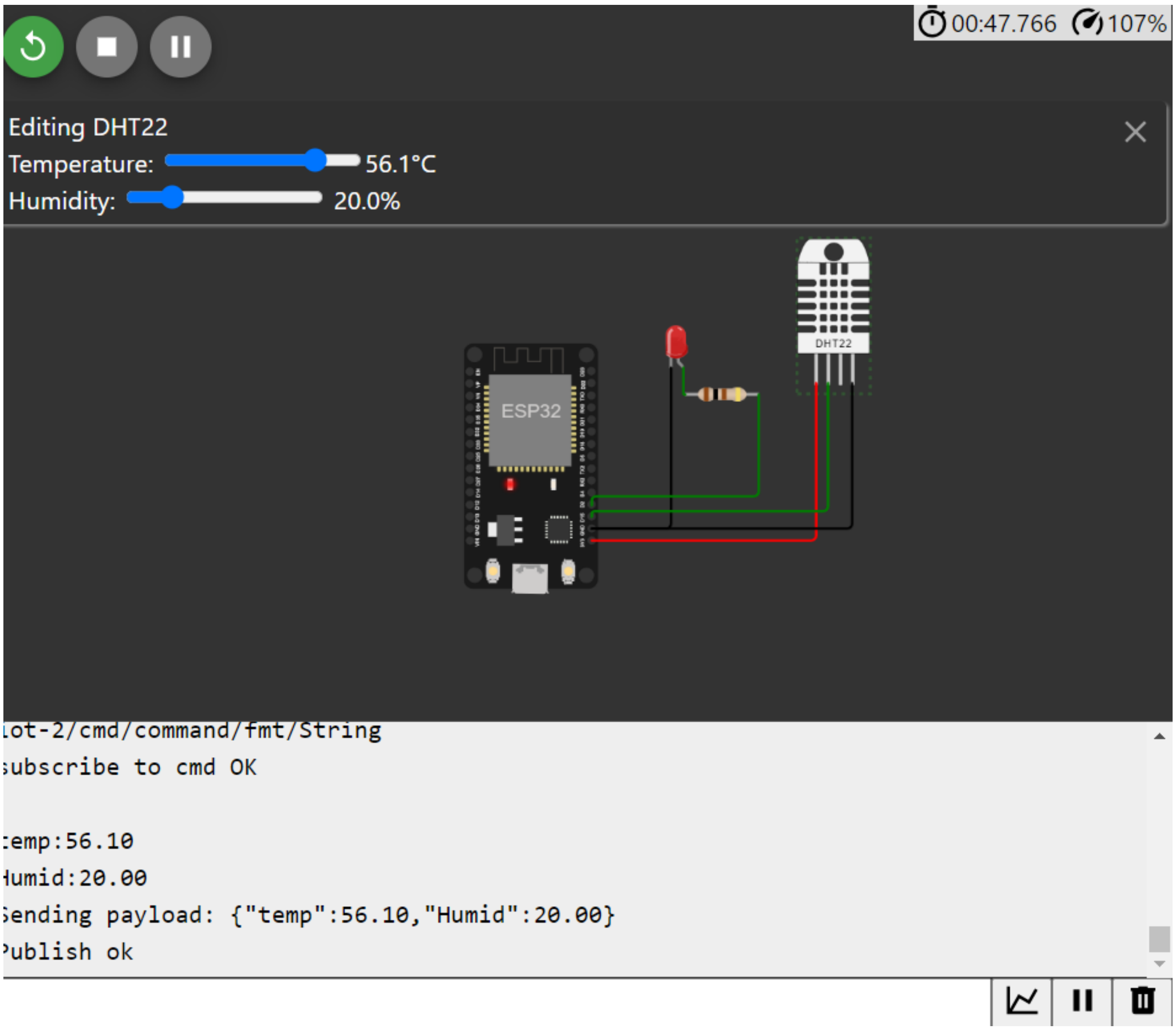
```
6/7/2023, 10:48:30 PM   node: debug 1
iot-2/type/abcd/id/1234/evt/Data/fmt/json : msg.payload :
number
 24
6/7/2023, 10:48:30 PM   node: debug 1
iot-2/type/abcd/id/1234/evt/Data/fmt/json : msg.payload :
number
 40
6/7/2023, 10:48:30 PM   node: debug 2
iot-2/type/abcd/id/1234/evt/Data/fmt/json : msg.payload :
Object
 ▸{ temp: 24, Humid: 40 }
6/7/2023, 10:48:30 PM   node: debug 1
iot-2/type/abcd/id/1234/evt/Data/fmt/json : msg.payload :
number
 24
6/7/2023, 10:48:30 PM   node: debug 1
iot-2/type/abcd/id/1234/evt/Data/fmt/json : msg.payload :
number
 40
6/7/2023, 10:48:30 PM   node: debug 2
iot-2/type/abcd/id/1234/evt/Data/fmt/json : msg.payload :
Object
 ▸{ temp: 24, Humid: 40 }
6/7/2023, 10:48:30 PM   node: debug 1
iot-2/type/abcd/id/1234/evt/Data/fmt/json : msg.payload :
```





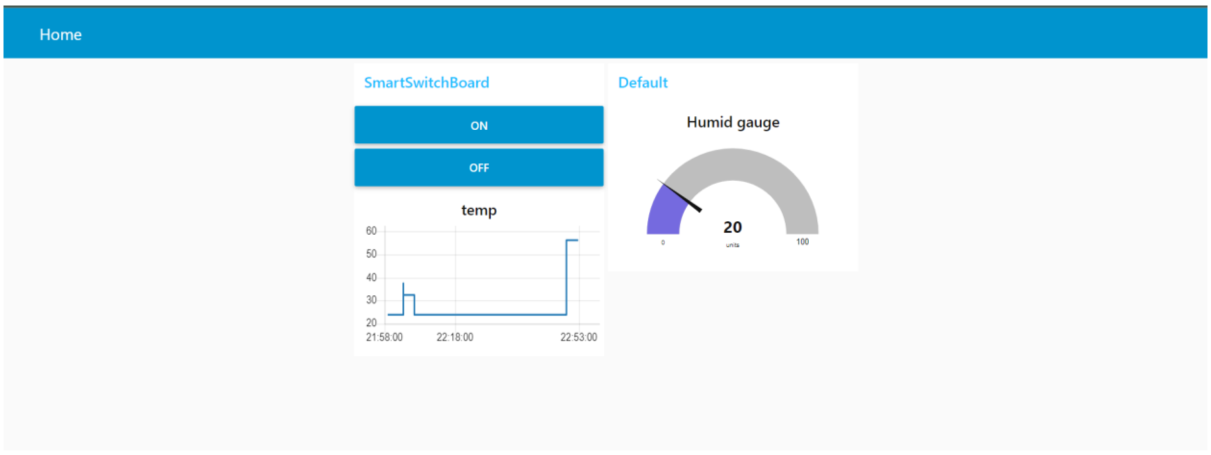
Editing DHT22
Temperature: 24.0°C
Humidity: 40.0%

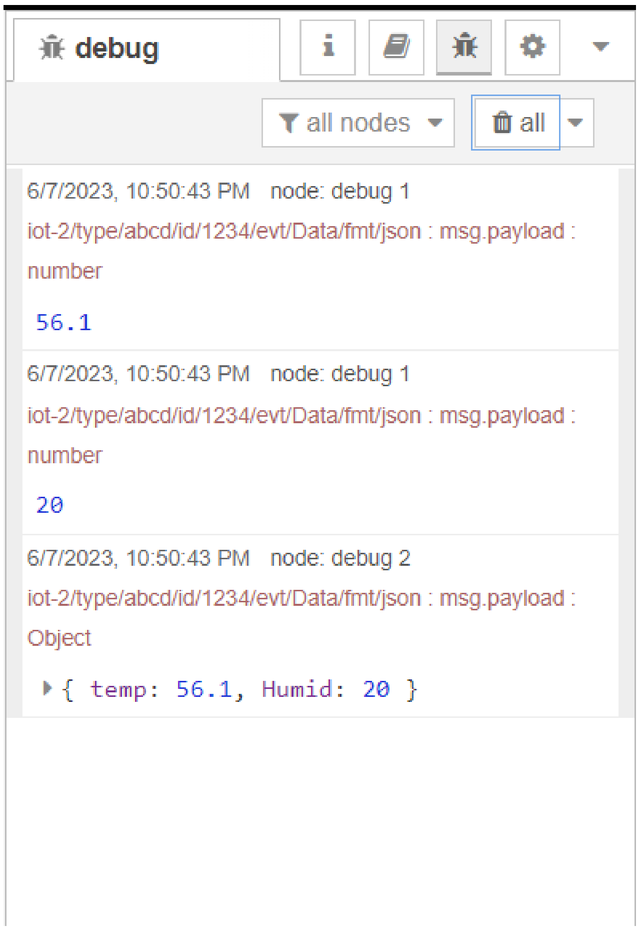
```
Humid:40.00
Sending payload: {"temp":24.00,"Humid":40.00}
Publish ok
temp:24.00
Humid:40.00
Sending payload: {"temp":24.00,"Humid":40.00}
Publish ok
```



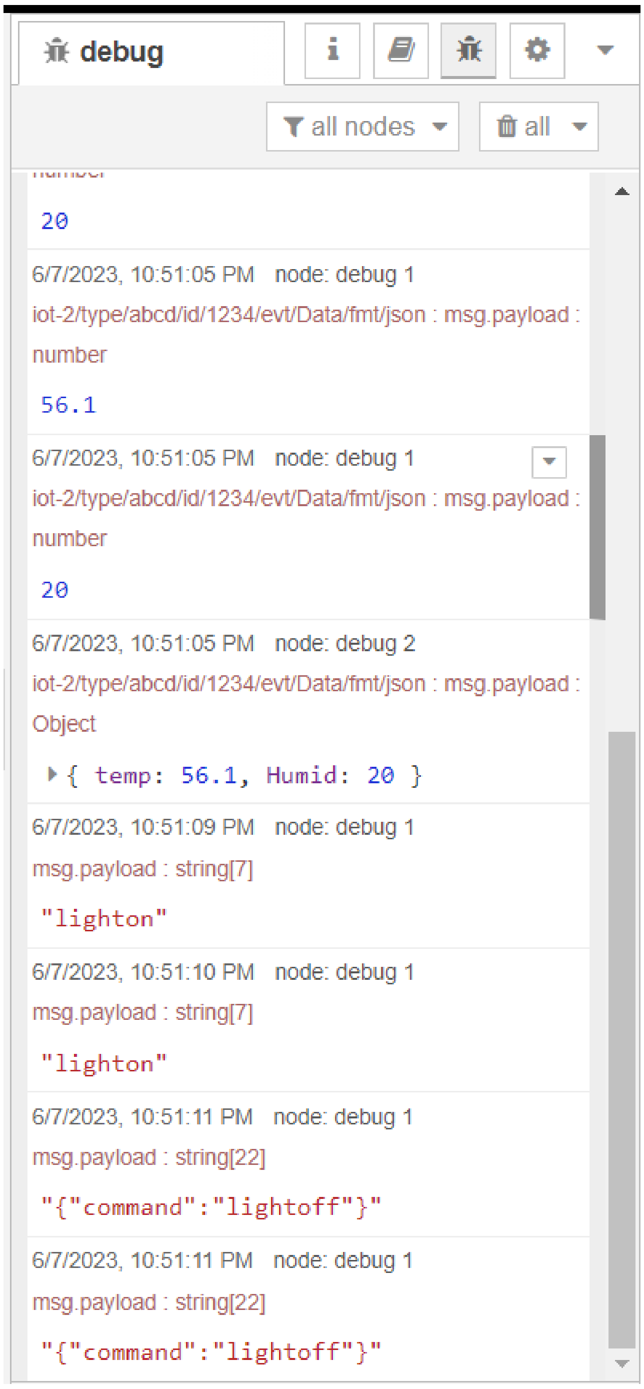
Editing DHT22
Temperature: 56.1°C
Humidity: 20.0%

```
iot-2/cmd/command/fmt/String
subscribe to cmd OK

temp:56.10
Humid:20.00
Sending payload: {"temp":56.10,"Humid":20.00}
Publish ok
```





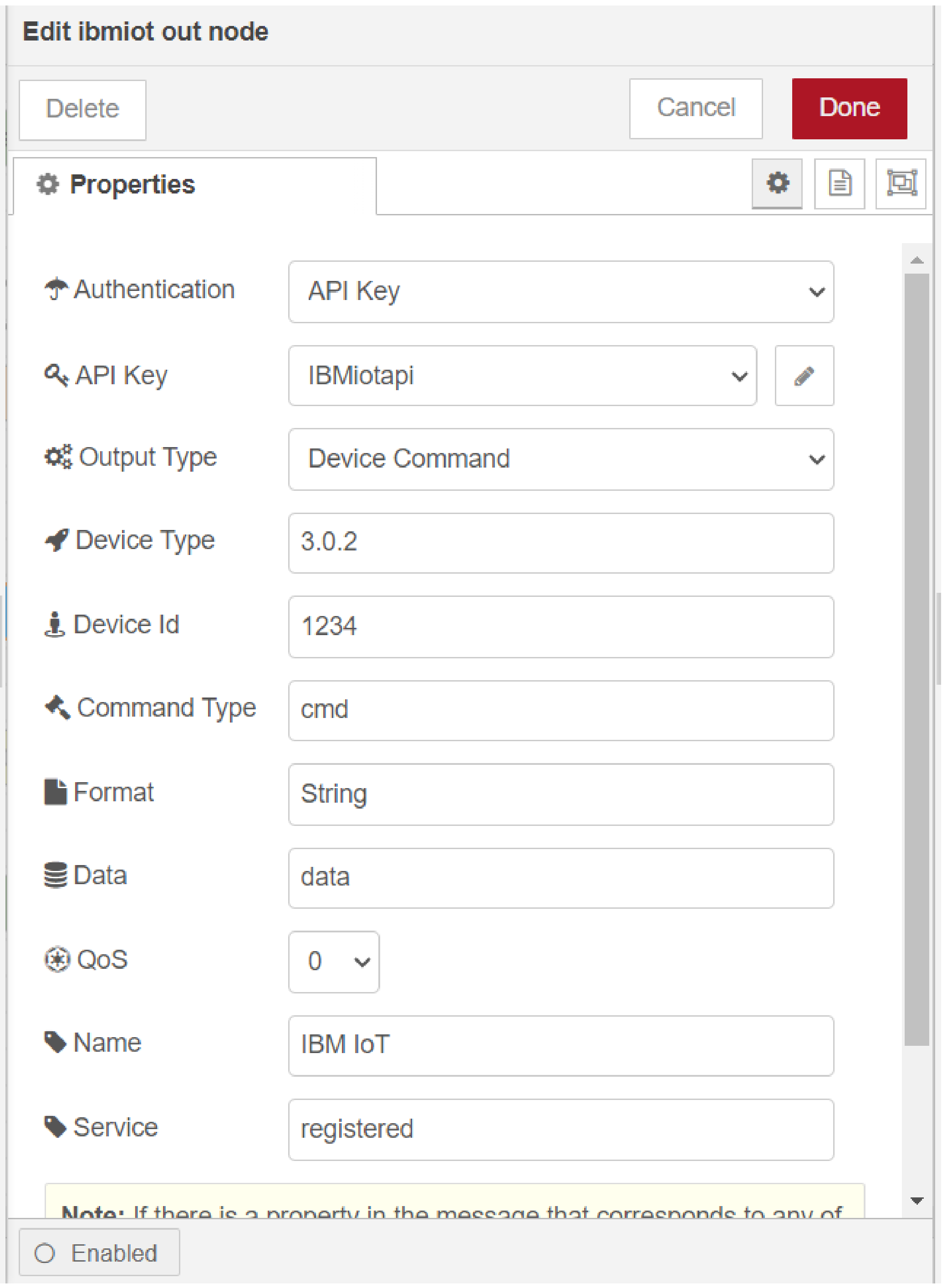
```
6/7/2023, 10:50:43 PM   node: debug 1
iot-2/type/abcd/id/1234/evt/Data/fmt/json : msg.payload :
number
 56.1
6/7/2023, 10:50:43 PM   node: debug 1
iot-2/type/abcd/id/1234/evt/Data/fmt/json : msg.payload :
number
 20
6/7/2023, 10:50:43 PM   node: debug 2
iot-2/type/abcd/id/1234/evt/Data/fmt/json : msg.payload :
Object
 ▸{ temp: 56.1, Humid: 20 }
```



debug

all nodes | all

```
20

6/7/2023, 10:51:05 PM   node: debug 1
iot-2/type/abcd/id/1234/evt/Data/fmt/json : msg.payload :
number
56.1

6/7/2023, 10:51:05 PM   node: debug 1
iot-2/type/abcd/id/1234/evt/Data/fmt/json : msg.payload :
number
20

6/7/2023, 10:51:05 PM   node: debug 2
iot-2/type/abcd/id/1234/evt/Data/fmt/json : msg.payload :
Object
▸{ temp: 56.1, Humid: 20 }

6/7/2023, 10:51:09 PM   node: debug 1
msg.payload : string[7]
"lighton"

6/7/2023, 10:51:10 PM   node: debug 1
msg.payload : string[7]
"lighton"

6/7/2023, 10:51:11 PM   node: debug 1
msg.payload : string[22]
"{"command":"lightoff"}"

6/7/2023, 10:51:11 PM   node: debug 1
msg.payload : string[22]
"{"command":"lightoff"}"
```



Edit ibmiot out node

Authentication	API Key
API Key	IBMiotapi
Output Type	Device Command
Device Type	3.0.2
Device Id	1234
Command Type	cmd
Format	String
Data	data
QoS	0
Name	IBM IoT
Service	registered