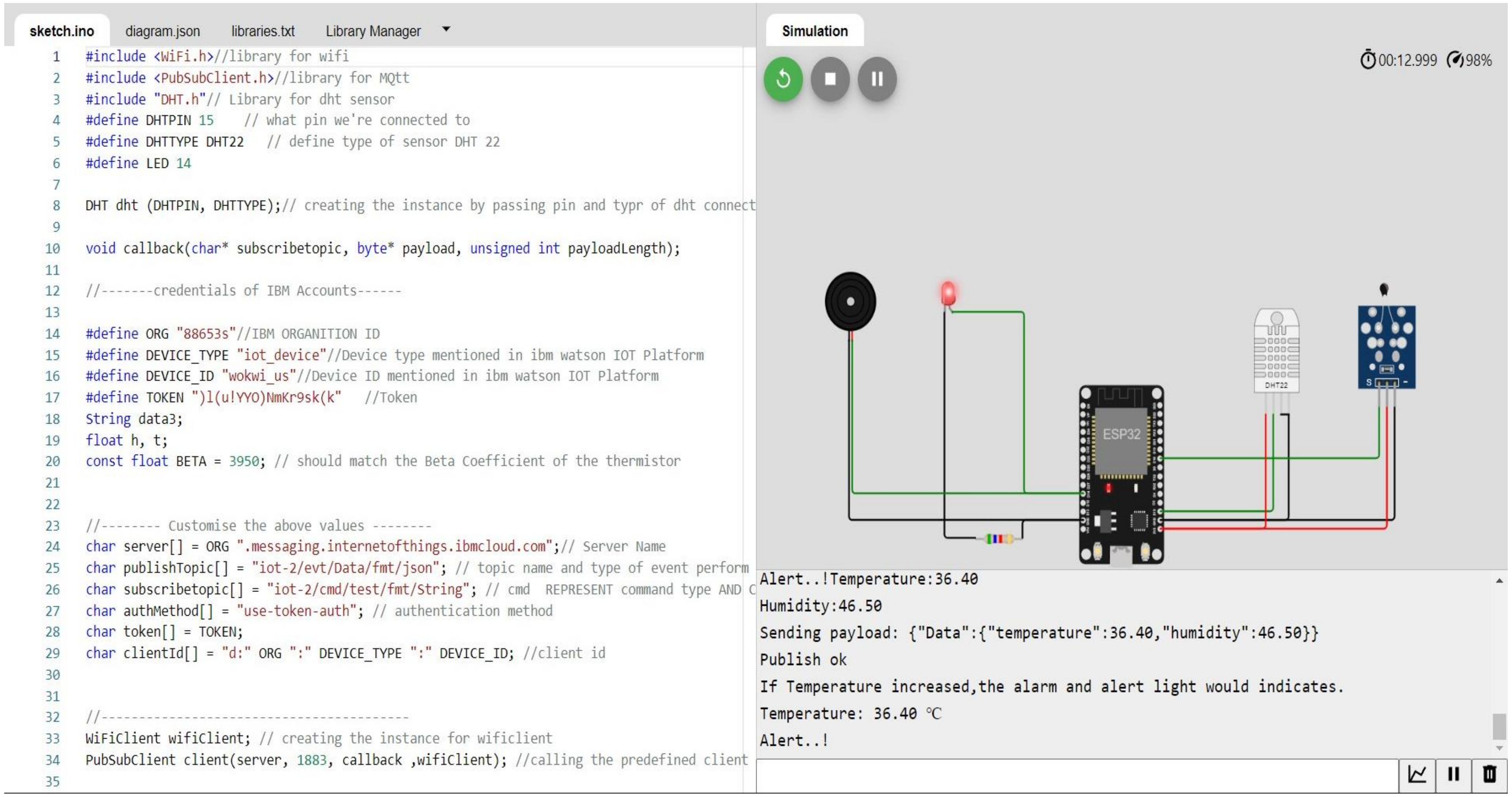
Project Development Phase

Sprint 3

Date	11 November 2022
Team ID	PNT2022TMID35665
Project Name	Industry-Specific Intelligent Fire Management System

OUTPUT:

WOKWI SIMULATOR

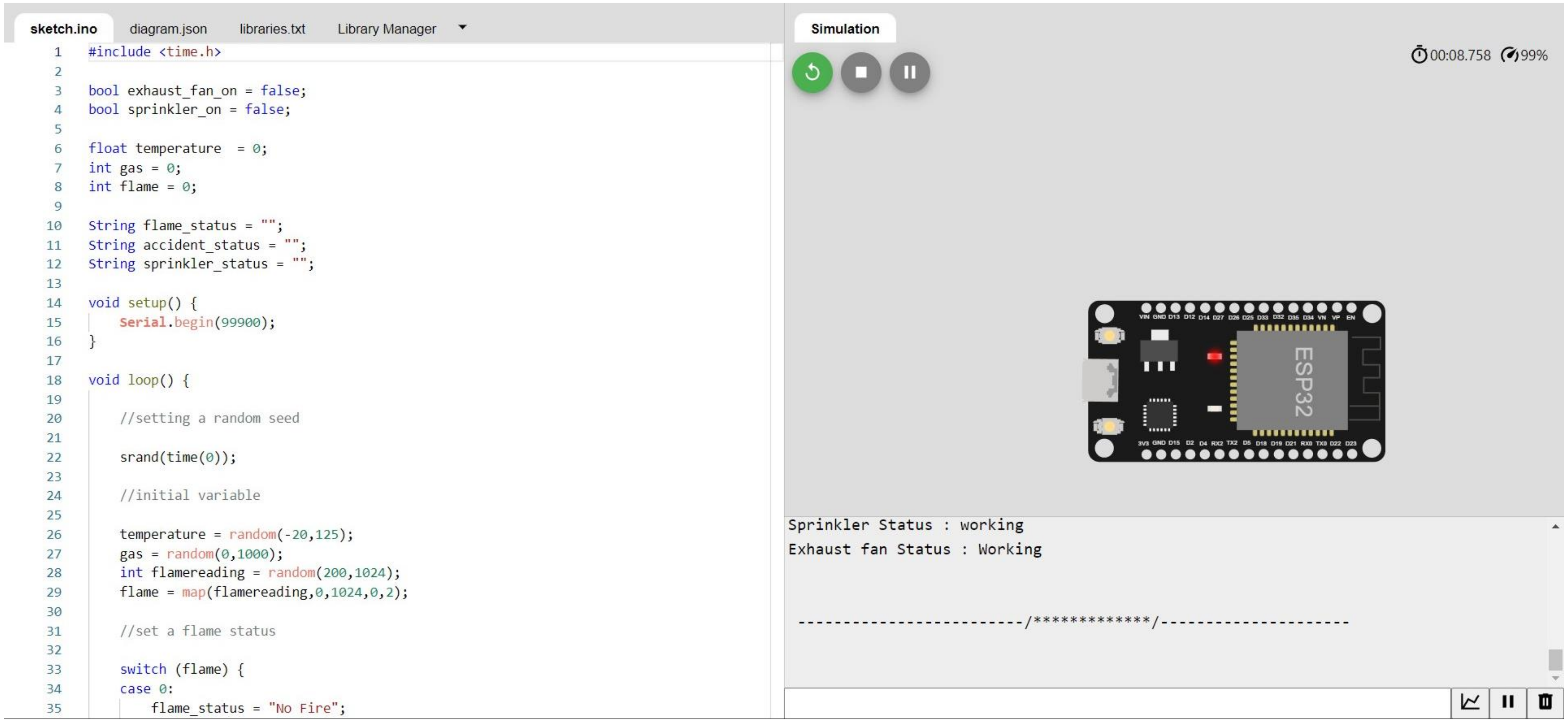


```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#include "DHT.h"// Library for dht sensor
#define DHTPIN 15    // what pin we're connected to
#define DHTTYPE DHT22   // define type of sensor DHT 22
#define LED 14

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

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

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

#define ORG "88653s"//IBM ORGANITION ID
#define DEVICE_TYPE "iot_device"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "wokwi_us"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN ")l(u!YYO)NmKr9sk(k"   //Token
String data3;
float h, t;
const float BETA = 3950; // should match the Beta Coefficient of the thermistor


//-------- 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
char subscribetopic[] = "iot-2/cmd/test/fmt/String"; // cmd  REPRESENT command type AND C
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
```

```
Alert..!Temperature:36.40
Humidity:46.50
Sending payload: {"Data":{"temperature":36.40,"humidity":46.50}}
Publish ok
If Temperature increased,the alarm and alert light would indicates.
Temperature: 36.40 ℃
Alert..!
```

OUTPUT:

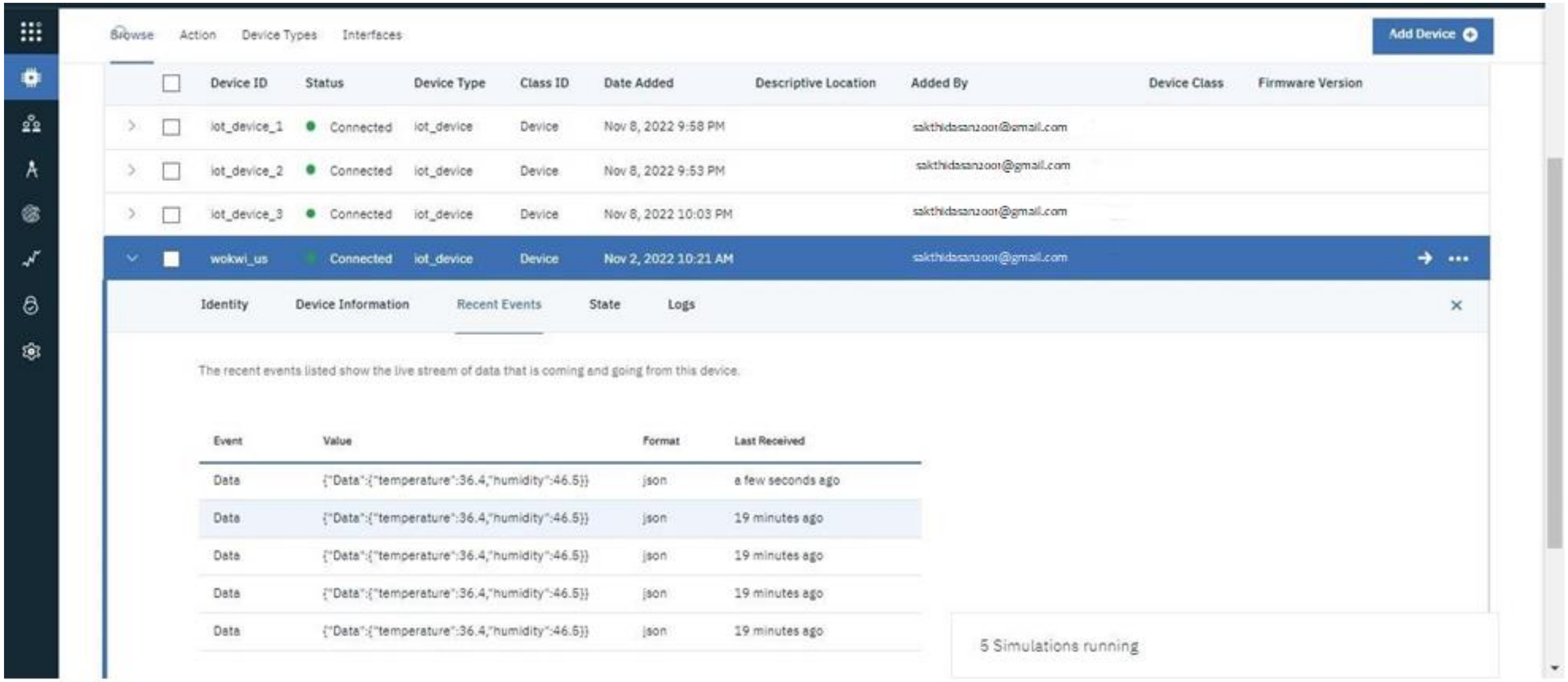
WOKWI SIMULATOR



```
#include <time.h>

bool exhaust_fan_on = false;
bool sprinkler_on = false;

float temperature  = 0;
int gas = 0;
int flame = 0;

String flame_status = "";
String accident_status = "";
String sprinkler_status = "";

void setup() {
    Serial.begin(99900);
}

void loop() {

    //setting a random seed

    srand(time(0));

    //initial variable

    temperature = random(-20,125);
    gas = random(0,1000);
    int flamereading = random(200,1024);
    flame = map(flamereading,0,1024,0,2);

    //set a flame status

    switch (flame) {
    case 0:
        flame_status = "No Fire";
```

```
Sprinkler Status : working
Exhaust fan Status : Working


 -------------------------/*************/---------------------
```

IBM WATSON OUTPUT



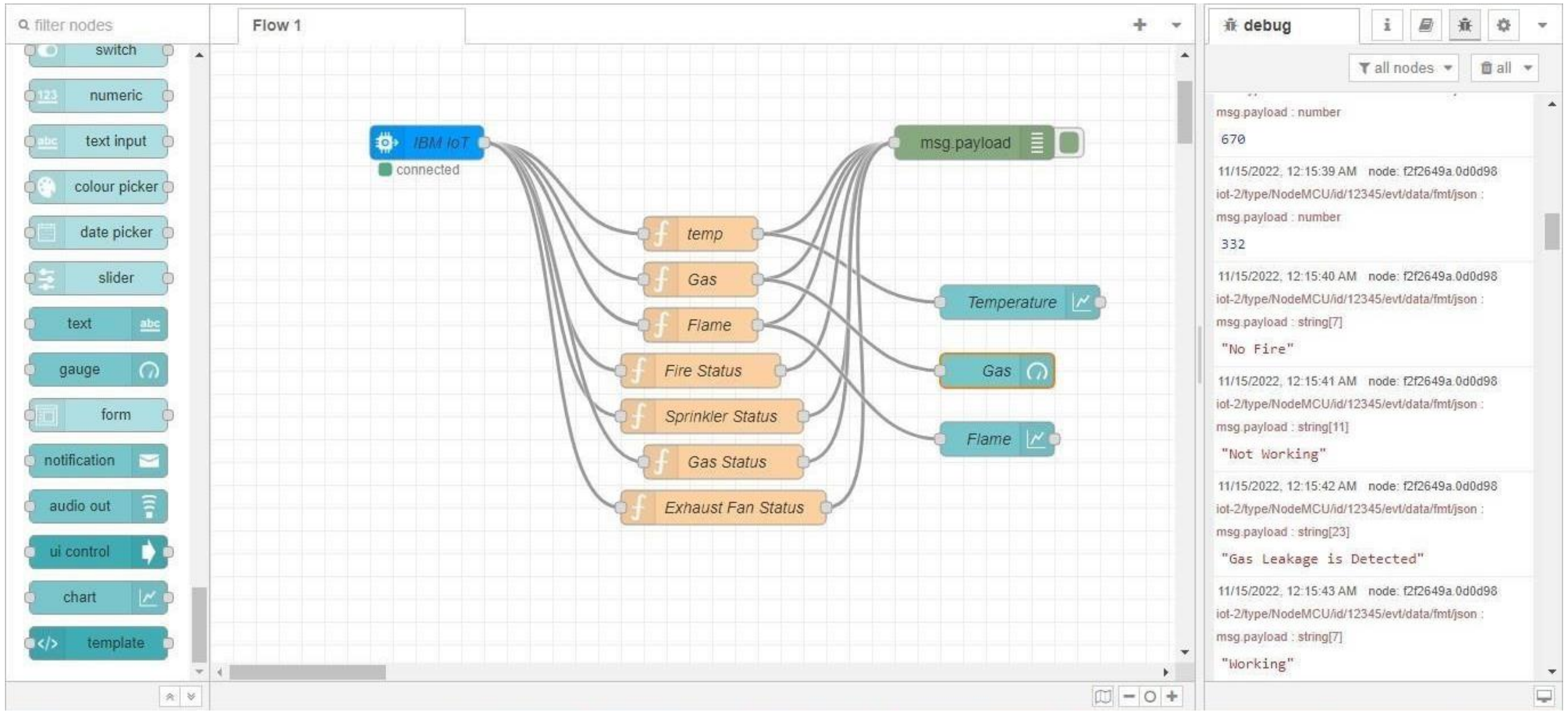
	Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location	Added By	Device Class	Firmware Version
	iot_device_1	● Connected	iot_device	Device	Nov 8, 2022 9:58 PM		sakthidasan2001@gmail.com		
	iot_device_2	● Connected	iot_device	Device	Nov 8, 2022 9:53 PM		sakthidasan2001@gmail.com		
	iot_device_3	● Connected	iot_device	Device	Nov 8, 2022 10:03 PM		sakthidasan2001@gmail.com		
	wokwi_us	● Connected	iot_device	Device	Nov 2, 2022 10:21 AM		sakthidasan2001@gmail.com		

The recent events listed show the live stream of data that is coming and going from this device.

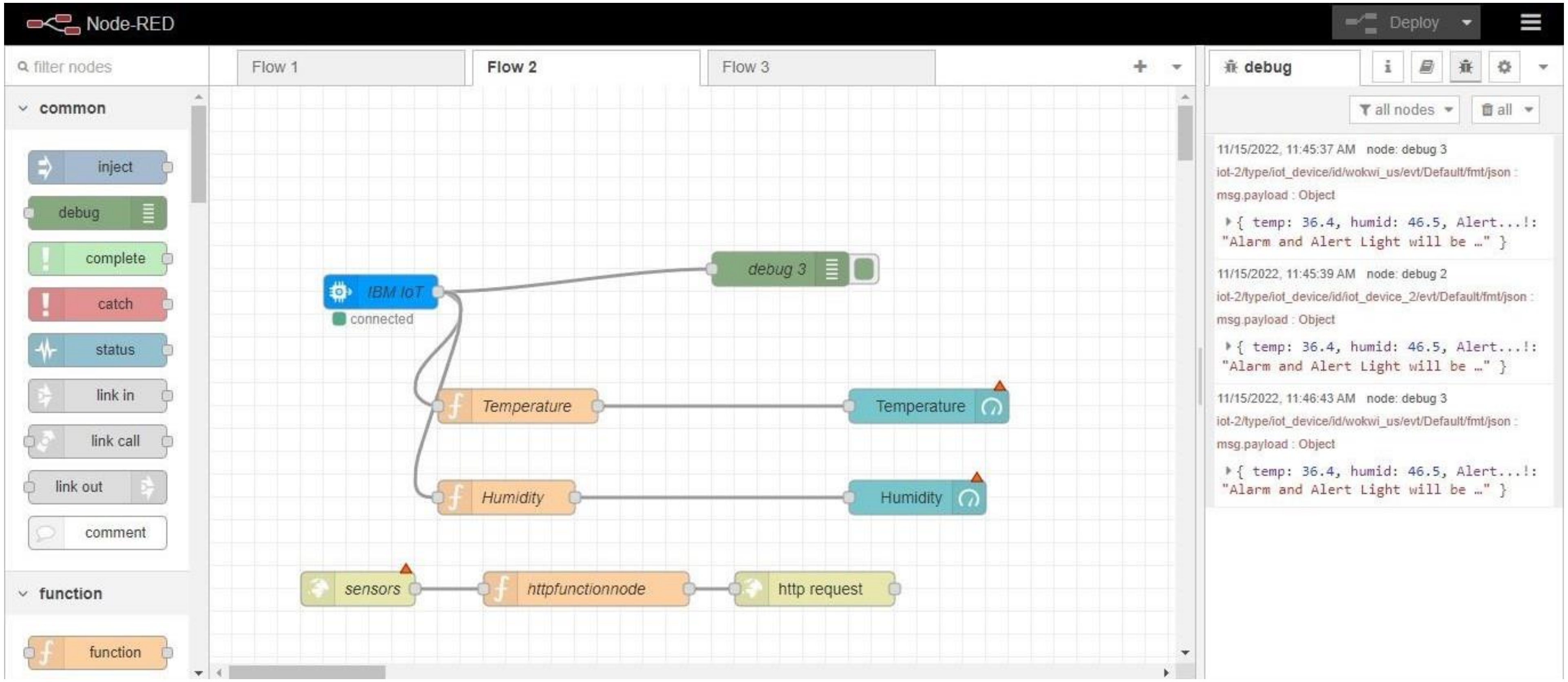
Event	Value	Format	Last Received
Data	{"Data":{"temperature":36.4,"humidity":46.5}}	json	a few seconds ago
Data	{"Data":{"temperature":36.4,"humidity":46.5}}	json	19 minutes ago
Data	{"Data":{"temperature":36.4,"humidity":46.5}}	json	19 minutes ago
Data	{"Data":{"temperature":36.4,"humidity":46.5}}	json	19 minutes ago
Data	{"Data":{"temperature":36.4,"humidity":46.5}}	json	19 minutes ago

5 Simulations running

TRANSFERRING DATA FROM IBM WATSON INTO NODE-RED

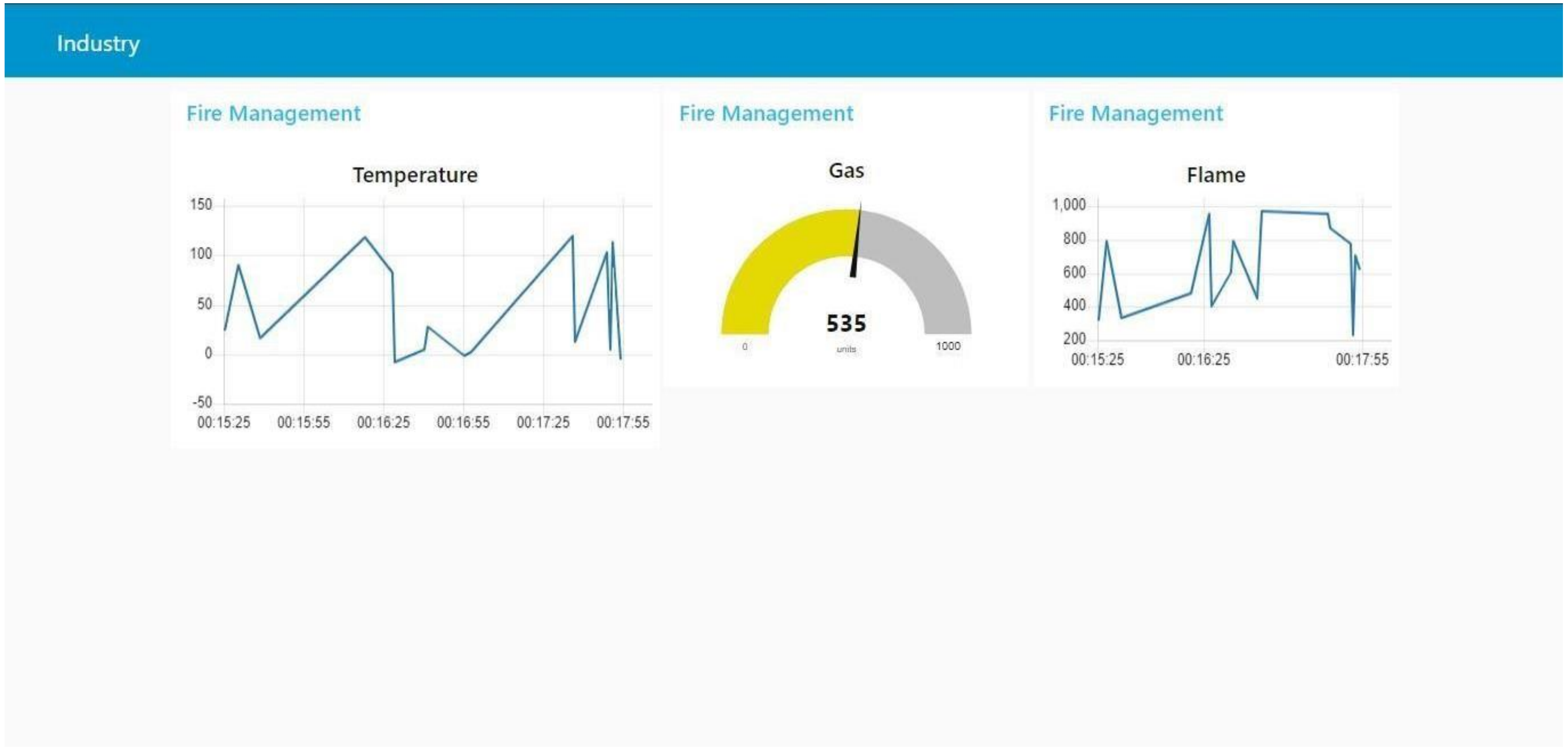


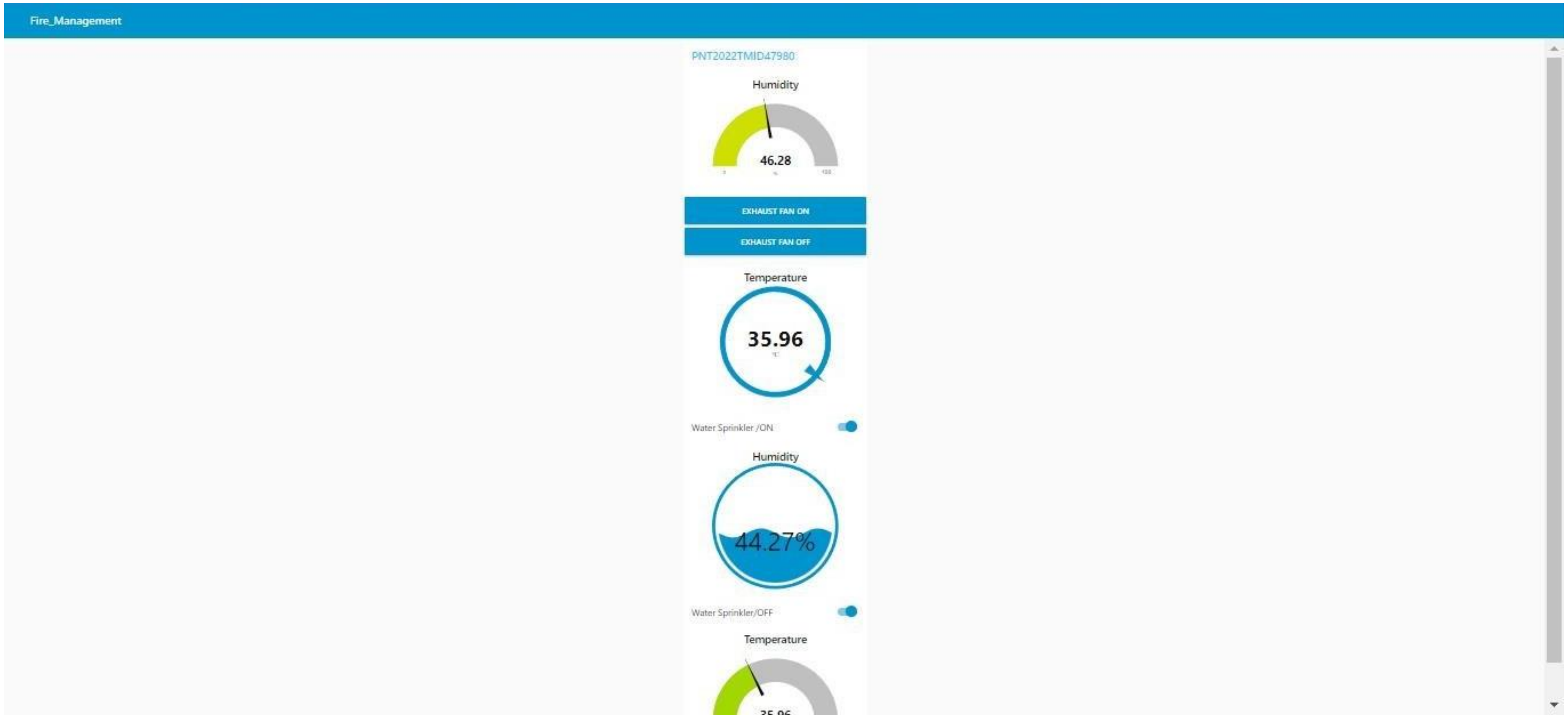
```
msg.payload : number
670
11/15/2022, 12:15:39 AM   node: f2f2649a.0d0d98
iot-2/type/NodeMCU/id/12345/evt/data/fmt/json :
msg.payload : number
332
11/15/2022, 12:15:40 AM   node: f2f2649a.0d0d98
iot-2/type/NodeMCU/id/12345/evt/data/fmt/json :
msg.payload : string[7]
"No Fire"
11/15/2022, 12:15:41 AM   node: f2f2649a.0d0d98
iot-2/type/NodeMCU/id/12345/evt/data/fmt/json :
msg.payload : string[11]
"Not Working"
11/15/2022, 12:15:42 AM   node: f2f2649a.0d0d98
iot-2/type/NodeMCU/id/12345/evt/data/fmt/json :
msg.payload : string[23]
"Gas Leakage is Detected"
11/15/2022, 12:15:43 AM   node: f2f2649a.0d0d98
iot-2/type/NodeMCU/id/12345/evt/data/fmt/json :
msg.payload : string[7]
"Working"
```



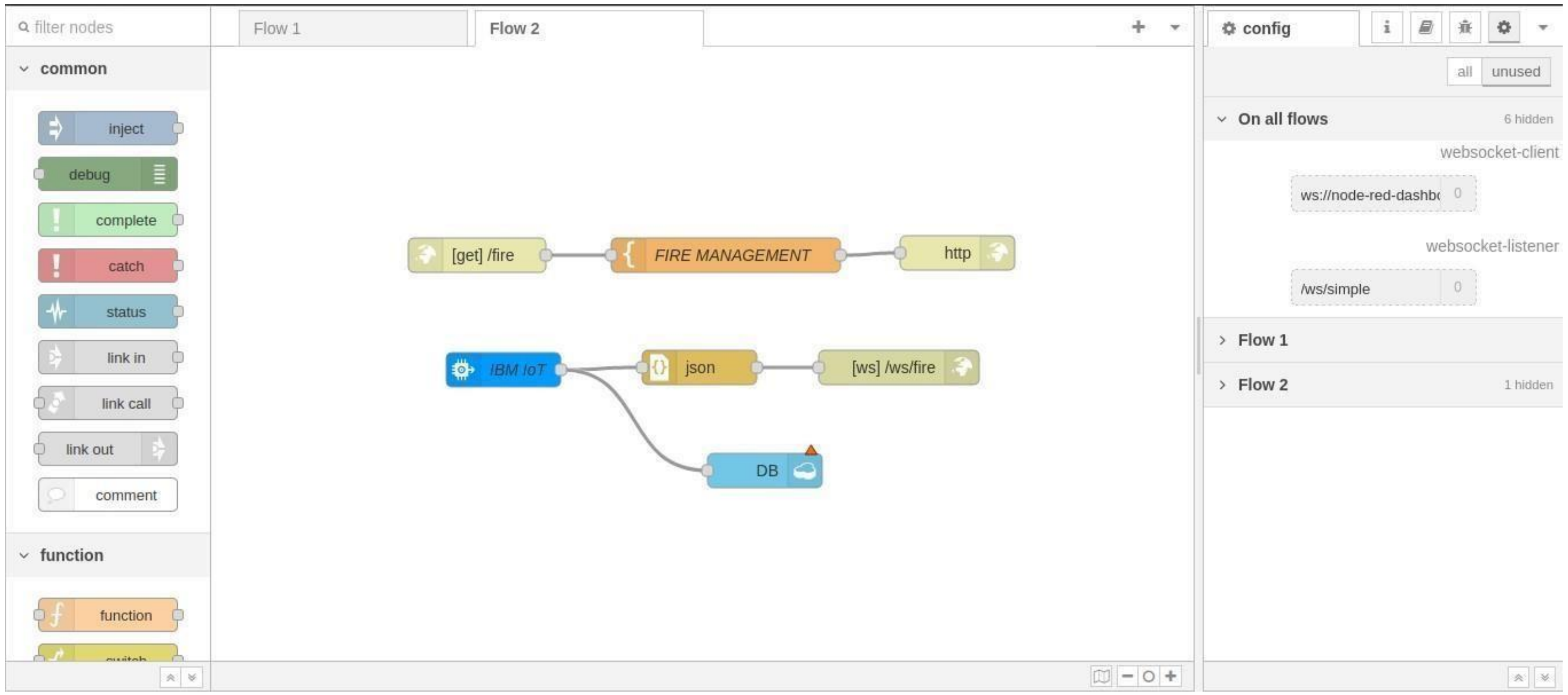
```
11/15/2022, 11:45:37 AM   node: debug 3
iot-2/type/iot_device/id/wokwi_us/evt/Default/fmt/json :
msg.payload : Object
 ▸{ temp: 36.4, humid: 46.5, Alert...!:
"Alarm and Alert Light will be …" }

11/15/2022, 11:45:39 AM   node: debug 2
iot-2/type/iot_device/id/iot_device_2/evt/Default/fmt/json :
msg.payload : Object
 ▸{ temp: 36.4, humid: 46.5, Alert...!:
"Alarm and Alert Light will be …" }

11/15/2022, 11:46:43 AM   node: debug 3
iot-2/type/iot_device/id/wokwi_us/evt/Default/fmt/json :
msg.payload : Object
 ▸{ temp: 36.4, humid: 46.5, Alert...!:
"Alarm and Alert Light will be …" }
```

NODE DASHBOARD



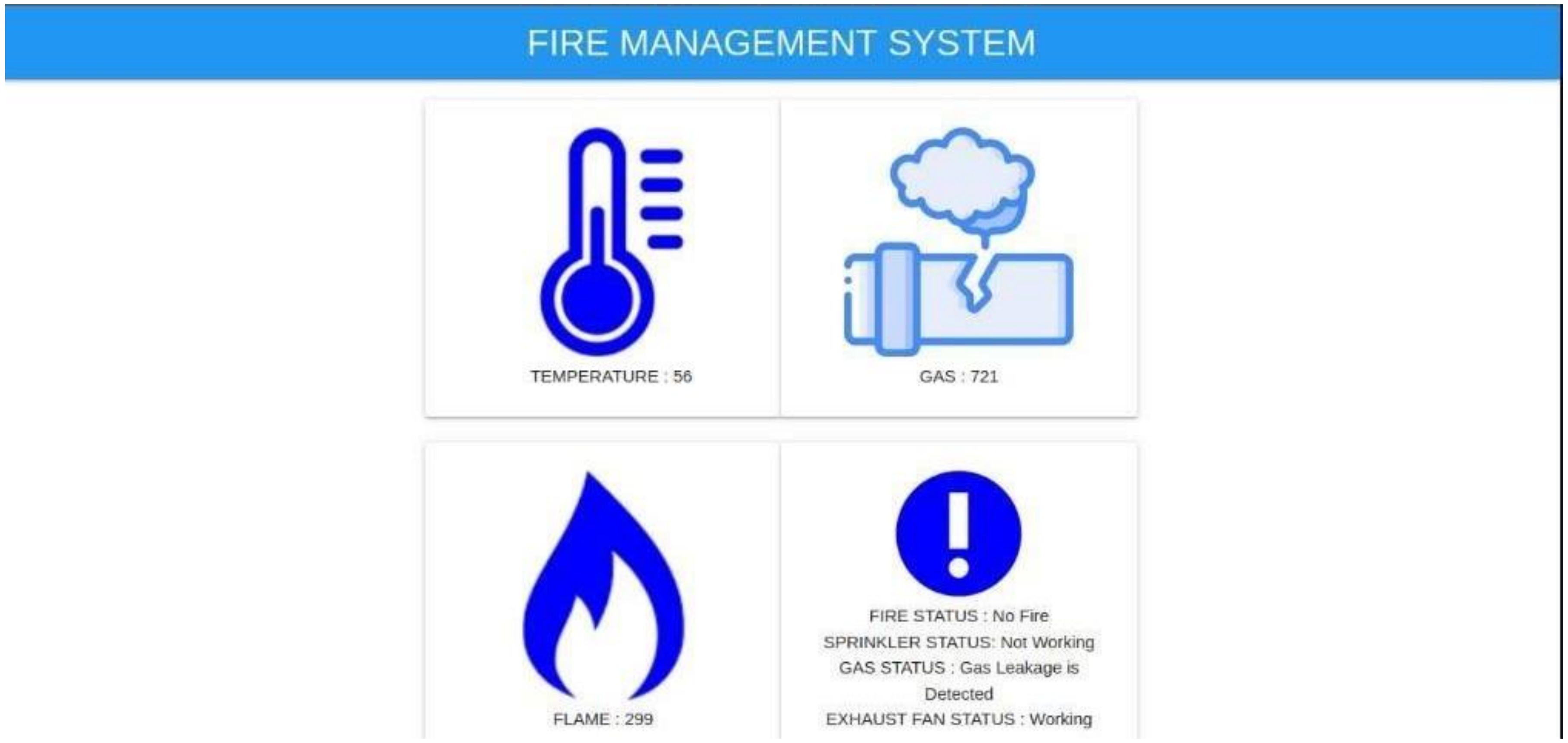

Fire_Management
PNT2022TMID47980
Humidity
46.28
EXHAUST FAN ON
EXHAUST FAN OFF
Temperature
35.96
Water Sprinkler /ON
Humidity
44.27%
Water Sprinkler/OFF
Temperature

TRANSFERRING DATA FROM NODE-RED INTO WEB UI



WEB UI

DESKTOP VIEW



FIRE MANAGEMENT SYSTEM

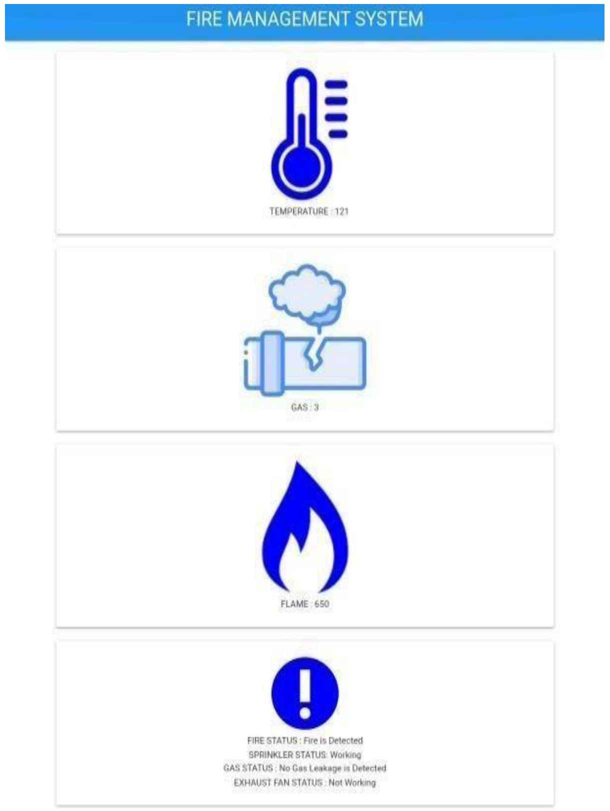
TEMPERATURE : 56

GAS : 721

FLAME : 299

FIRE STATUS : No Fire
SPRINKLER STATUS: Not Working
GAS STATUS : Gas Leakage is Detected
EXHAUST FAN STATUS : Working

MOBILE VIEW


FIRE MANAGEMENT SYSTEM

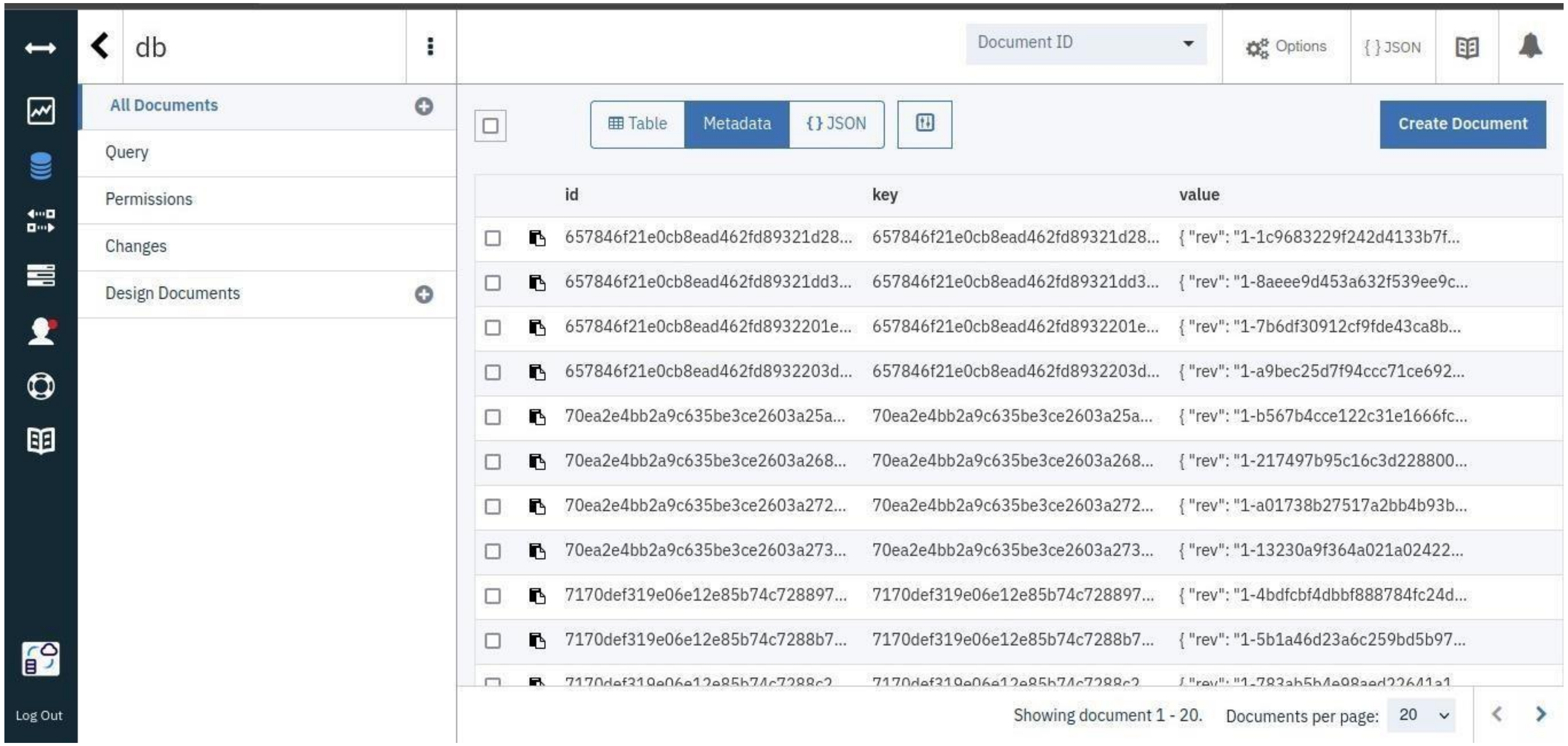
TEMPERATURE : 121

GAS : 3

FLAME : 650

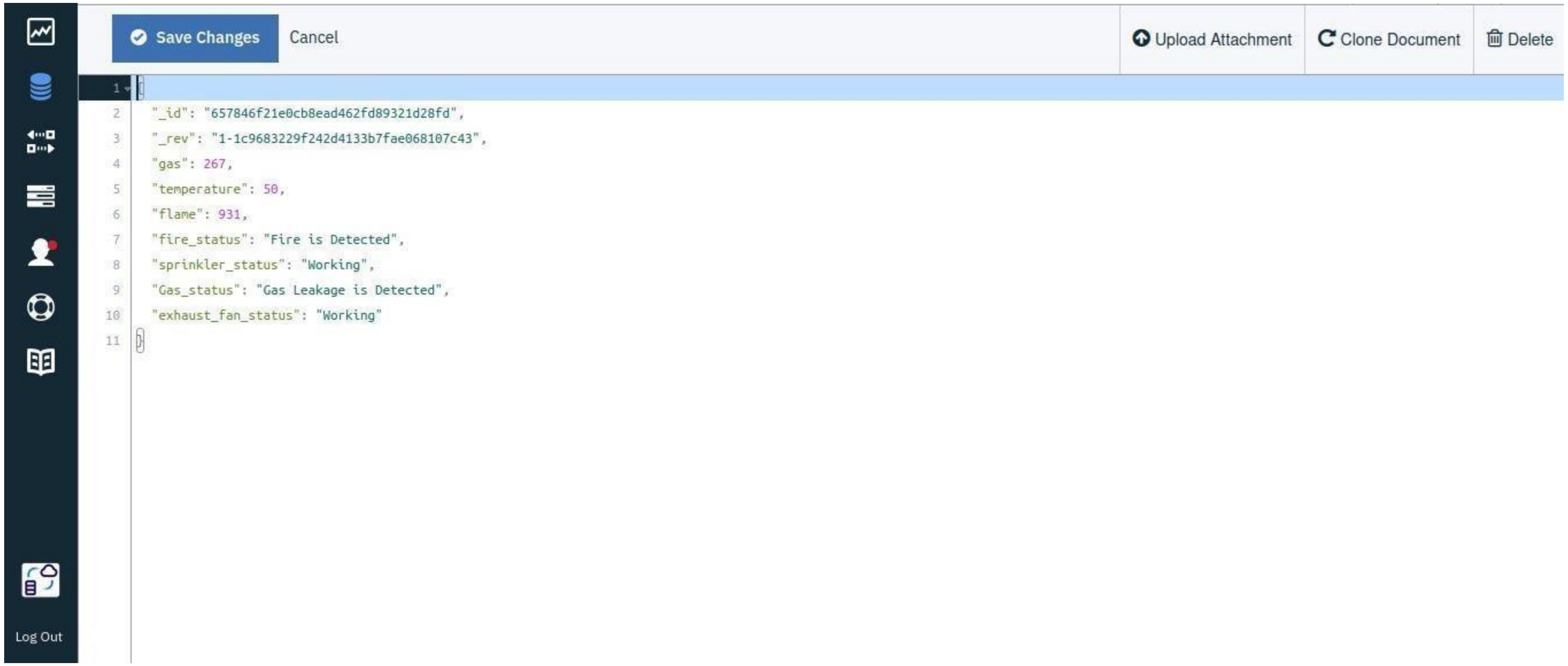
FIRE STATUS : Fire is Detected
SPRINKLER STATUS: Working
GAS STATUS : No Gas Leakage is Detected
EXHAUST FAN STATUS : Not Working

CLOUDANT:



id	key	value
657846f21e0cb8ead462fd89321d28...	657846f21e0cb8ead462fd89321d28...	{ "rev": "1-1c9683229f242d4133b7f...
657846f21e0cb8ead462fd89321dd3...	657846f21e0cb8ead462fd89321dd3...	{ "rev": "1-8aeee9d453a632f539ee9c...
657846f21e0cb8ead462fd8932201e...	657846f21e0cb8ead462fd8932201e...	{ "rev": "1-7b6df30912cf9fde43ca8b...
657846f21e0cb8ead462fd8932203d...	657846f21e0cb8ead462fd8932203d...	{ "rev": "1-a9bec25d7f94ccc71ce692...
70ea2e4bb2a9c635be3ce2603a25a...	70ea2e4bb2a9c635be3ce2603a25a...	{ "rev": "1-b567b4cce122c31e1666fc...
70ea2e4bb2a9c635be3ce2603a268...	70ea2e4bb2a9c635be3ce2603a268...	{ "rev": "1-217497b95c16c3d228800...
70ea2e4bb2a9c635be3ce2603a272...	70ea2e4bb2a9c635be3ce2603a272...	{ "rev": "1-a01738b27517a2bb4b93b...
70ea2e4bb2a9c635be3ce2603a273...	70ea2e4bb2a9c635be3ce2603a273...	{ "rev": "1-13230a9f364a021a02422...
7170def319e06e12e85b74c728897...	7170def319e06e12e85b74c728897...	{ "rev": "1-4bdfcbf4dbbf888784fc24d...
7170def319e06e12e85b74c7288b7...	7170def319e06e12e85b74c7288b7...	{ "rev": "1-5b1a46d23a6c259bd5b97...
7170def319e06e12e85b74c7288c2	7170def319e06e12e85b74c7288c2	{ "rev": "1-783ab5b4e98aed22641a1

Showing document 1 - 20. Documents per page: 20



```
{
  "_id": "657846f21e0cb8ead462fd89321d28fd",
  "_rev": "1-1c9683229f242d4133b7fae068107c43",
  "gas": 267,
  "temperature": 50,
  "flame": 931,
  "fire_status": "Fire is Detected",
  "sprinkler_status": "Working",
  "Gas_status": "Gas Leakage is Detected",
  "exhaust_fan_status": "Working"
}
```

<u>CODE:</u>

```
  #include <time.h>
  #include <WiFi.h>
  #include <PubSubClient.h>

#define ORG "88653s"
#define DEVICE_TYPE "iot_device"
#define DEVICE_ID "wokwi_us"
#define TOKEN ")l(u!YYO)NmKr9sk(k"

  char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
  char publishTopic[] = "iot-2/evt/data/fmt/json";
  char authMethod[] = "use-token-auth";
  char token[] = TOKEN;
  char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;

  WiFiClient wifiClient;
  PubSubClient client(server, 1883, wifiClient);

  float temperature = 0;
  int gas = 0;
  int flame = 0;
  String flame_status = "";


  String Gas_status = "";
  String exhaust_fan_status = "";
  String sprinkler_status = "";
```

```
   void setup() {
   Serial.begin(99900);
   wifiConnect();
   mqttConnect();
   }
   void loop() {
   srand(time(0));
                          //initial variables and random generated data

              temperature = random(-20,125);
gas = random(0,1000);
int flamereading = random(200,1024);
flame = map(flamereading,200,1024,0,2);

        //set a flame status
        switch (flame) {
        case 0: flame_status = "No Fire";
        break;
        case  1:  flame_status  =  "Fire  is
   Detected";
        break;
        }

        //send the sprinkler status

        if(flame==1){
            sprinkler_status = "Working";
        }
   else{
   sprinkler_status = "Not Working";
        }
```

```
    //toggle the fan according to gas reading

    if(gas > 100){
        Gas_status = "Gas Leakage is Detected";
        exhaust_fan_status = "Working";
    }
else{
        Gas_status = "No Gas Leakage is Detected";
        exhaust_fan_status = "Not Working";
    }
```

//Wokwi Project

```
#include <time.h>
#include <WiFi.h>
#include <PubSubClient.h>

#define ORG "wt19pm"
#define DEVICE_TYPE "NodeMCU"
#define DEVICE_ID "12345"
#define TOKEN "12345678"

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/data/fmt/json";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;

WiFiClient wifiClient;
PubSubClient client(server, 1883, wifiClient);

float temperature  = 0;
int gas = 0;
int flame = 0;

String flame_status = "";
String Gas_status = "";
String exhaust_fan_status = "";
String sprinkler_status = "";



void setup() {
  Serial.begin(99900);
   wifiConnect();
   mqttConnect();
}

void loop() {

  srand(time(0));

    //initial variables and random generated data

    temperature = random(-20,125);
    gas = random(0,1000);
    int flamereading = random(200,1024);
    flame = map(flamereading,200,1024,0,2);

    //set a flame status
```

```
switch (flame) {
case 0:
   flame_status = "No Fire";
   break;
case 1:
   flame_status = "Fire is Detected";
   break;
}

//send the sprinkler status

if(flame==1){
   sprinkler_status = "Working";
}
else{
   sprinkler_status = "Not Working";

}

//toggle the fan according to gas reading

if(gas > 100){
   Gas_status = "Gas Leakage is Detected";
   exhaust_fan_status = "Working";
```

```
}
else{
   Gas_status = "No Gas Leakage is Detected";
   exhaust_fan_status = "Not Working";
}

//json format for IBM Watson

String payload = "{";
payload+="\"gas\":";
payload+=gas;
payload+=",";
payload+="\"temperature\":";
payload+=(int)temperature;
payload+=",";
payload+="\"flame\":";
payload+=flamereading;
payload+=",";
payload+="\"fire_status\":\""+flame_status+"\",";
payload+="\"sprinkler_status\":\""+sprinkler_status+"\",";
payload+="\"Gas_status\":\""+Gas_status+"\",";
payload+="\"exhaust_fan_status\":\""+exhaust_fan_status+"\"}";

if(client.publish(publishTopic, (char*) payload.c_str()))
{
    Serial.println("Publish OK");
  }
  else{
    Serial.println("Publish failed");
  }
  delay(1000);


  if (!client.loop())
  {
   mqttConnect();
  }
}


void wifiConnect()
{
 Serial.print("Connecting to ");
 Serial.print("Wifi");
 WiFi.begin("Wokwi-GUEST", "", 6);
 while (WiFi.status() != WL_CONNECTED)
 {
  delay(500);
  Serial.print(".");
 }
```

```
  Serial.print("WiFi connected, IP address: ");
  Serial.println(WiFi.localIP());

}



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

    Serial.println();
  }
}
```

//.........Project Data in json Format. ..... /

```
{
  "version": 1,
  "author": "Jagadish K",
  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": 10, "left": -60.67, "attrs": {} },
    {
      "type": "wokwi-led",
      "id":  "led1",  "top":  -
      109,
      "left": -244.4,
      "attrs": { "color": "red" } },
    {
      "type": "wokwi-dht22",
      "id": "dht1",
      "top": -70.9,
      "left": 157.2,
      "attrs": { "temperature": "36.4", "humidity": "46.5" }
    },
    {
      "type": "wokwi-ntc-temperature-sensor",
      "id": "ntc1",
      "top": -69.55,
      "left": 253.55,
      "rotate": 90,
      "attrs": {}
    },
    {
      "type": "wokwi-resistor",
      "id": "r1",
      "top": 169.5,
      "left": -190.59,
      "attrs": { "value": "5600" }
    },
    {
      "type": "wokwi-buzzer",
      "id": "bz1",
      "top": -118.83,
      "left": -378.64,
      "attrs": { "volume": "0.1" }
    }
  ],
  "connections": [
    [ "esp:TX0", "$serialMonitor:RX", "", [] ],
    [ "esp:RX0", "$serialMonitor:TX", "", [] ],
    [ "dht1:GND", "esp:GND.1", "black", [ "v0" ] ],
    [ "dht1:SDA", "esp:D15", "green", [ "v0" ] ],
    [ "ntc1:GND", "esp:GND.1", "black", [ "v0" ] ],
    [ "ntc1:VCC", "esp:3V3", "red", [ "v0" ] ],
    [ "led1:C", "r1:1", "black", [ "v0" ] ],
    [ "r1:2", "esp:GND.2", "black", [ "v0" ] ],
    [ "led1:A", "esp:D14", "green", [ "v-0.86", "h89.56", "v199.46" ] ],
    [ "ntc1:OUT", "esp:D18", "green", [ "v0" ] ],
    [ "bz1:1", "esp:GND.2", "black", [ "v0" ] ],
    [ "bz1:2", "esp:D14", "green", [ "v0" ] ],
    [ "dht1:VCC", "esp:3V3", "red", [ "v0" ] ],
    [ "dht1:NC", "dht1:GND", "black", [ "v0" ] ]
  ]
}
```

//.........Python Script for Random Outputs of Temperature and Humidity......

```
import time import sys
import          ibmiotf.application
import ibmiotf.device
import random


#Provide your IBM Watson Device Credentials
organization = "bxobbs"
deviceType = "b5ibm"
deviceId = "b5device"
authMethod = "token"
authToken = "b55m1eibm"

# Initialize GPIO

def myCommandCallback(cmd):
    print("Command       received:       %s"       %       cmd.data['command'])
    status=cmd.data['command']
    if status=="lighton":
        print ("led is on")
    else :

        print ("led is off")

    #print(cmd)
```

```
try:
        deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth- token": authToken}
        deviceCli = ibmiotf.device.Client(deviceOptions)
          #..............................................

except Exception as e:
          print("Caught exception connecting device: %s" % str(e))
          sys.exit()


# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times deviceCli.connect()

while True:
          #Get Sensor Data from DHT11

          temp=random.randint(0,100)
          Humid=random.randint(0,100)

          data = { 'temp' : temp, 'Humid': Humid }
          #print data
          def myOnPublishCallback():
               print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % Humid, "to IBM Watson")

          success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCallback)
          if not success:
               print("Not connected to IoTF")
          time.sleep(1)

          deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud
deviceCli.disconnect()
```