SPRINT 4

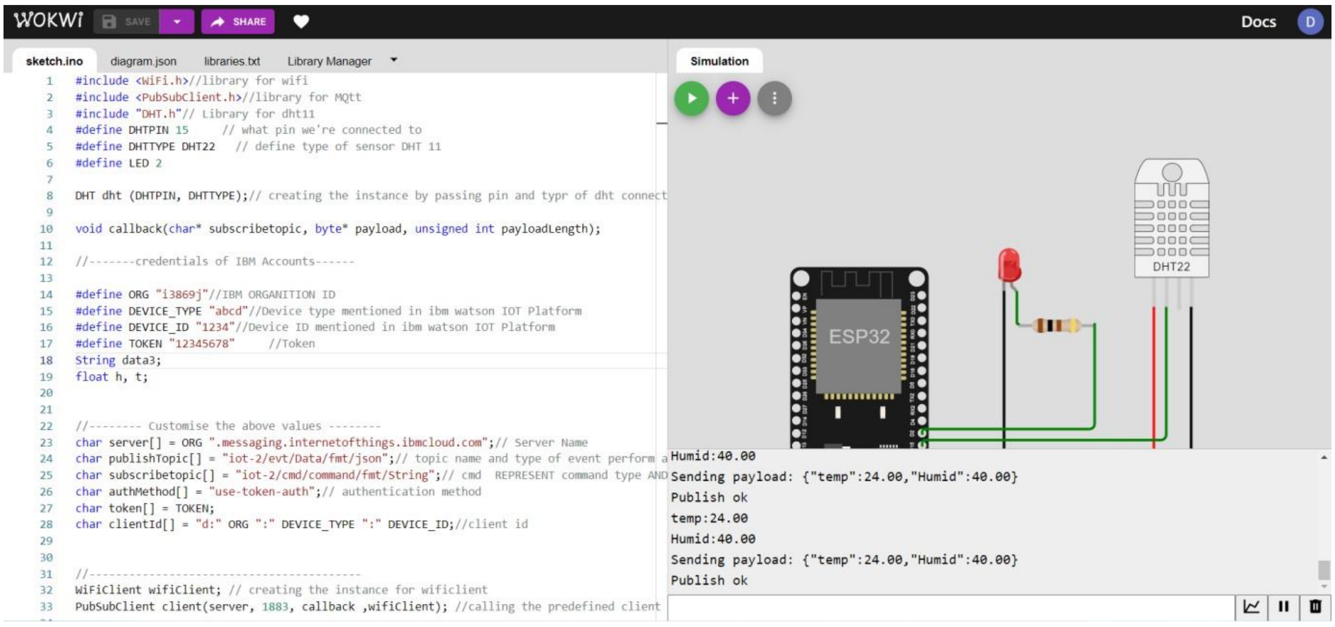
Date	17 November 2022
Team ID	PNT2022TMID21831
Project Name	Hazardous Area Monitoring for Industrial Plant powered by IoT

WOKWI CODE:

```
#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 "iagqzu"//IBM ORGANITION ID
#define DEVICE_TYPE "Deepak"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "123"//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 < payloadLength; 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="";
}
```

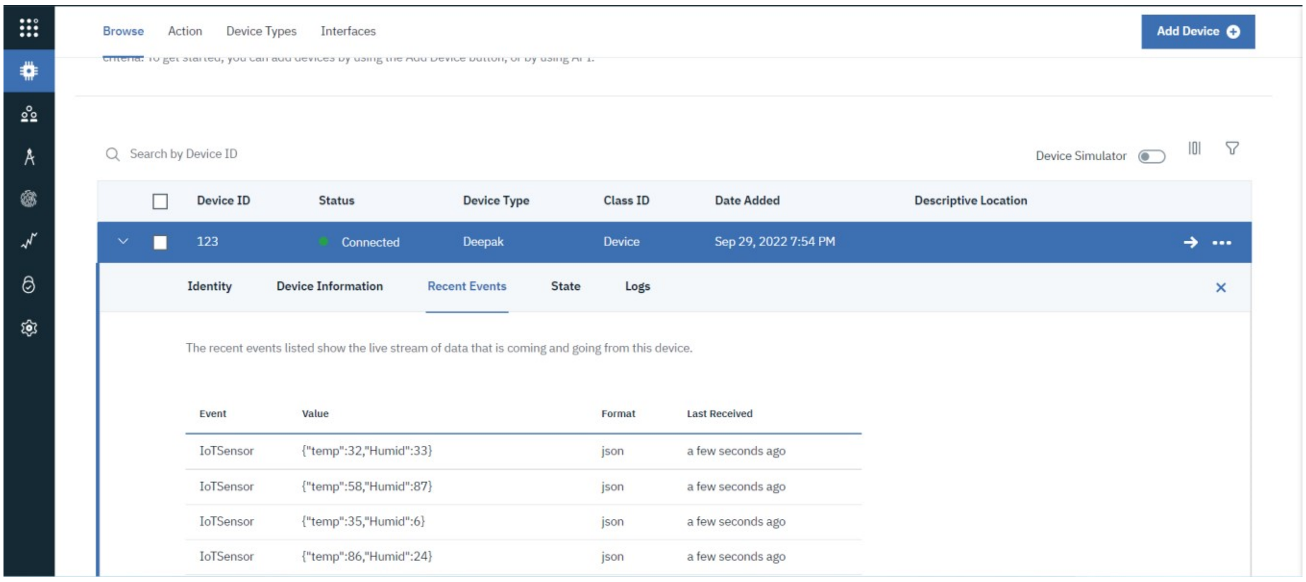
WOKWI OUTPUT:



```
#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 connect

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

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

#define ORG "i3869j"//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 a
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd  REPRESENT command type AND
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
```

```
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
```

IBM WATSON PLATFORM   DEVICE EVENT LOG:



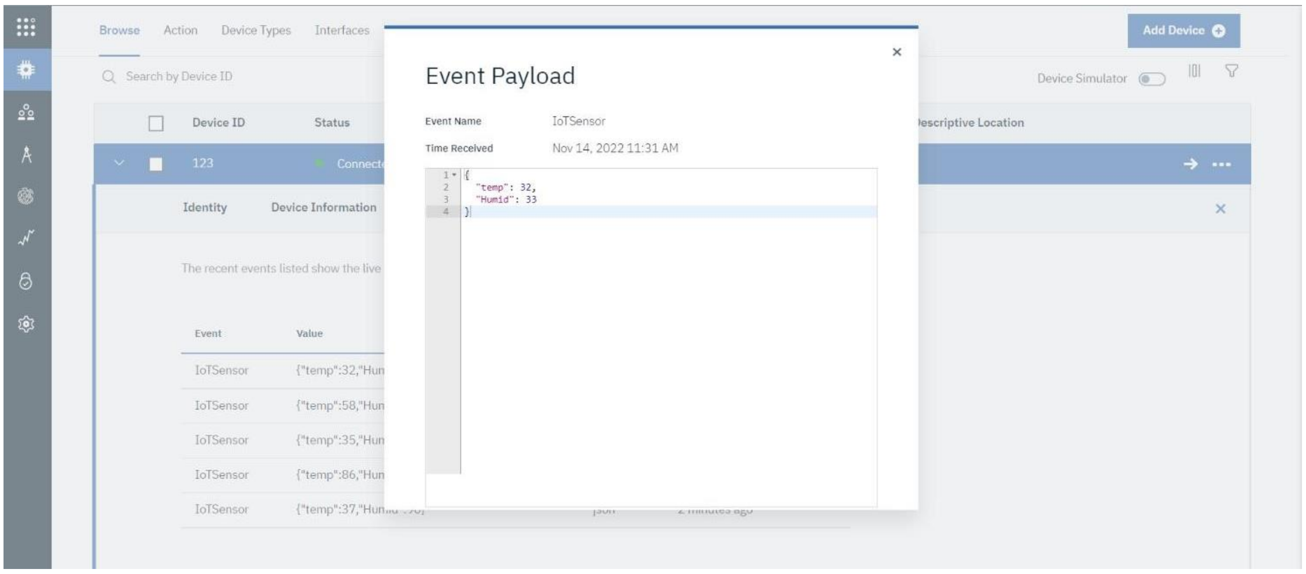
Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
123	Connected	Deepak	Device	Sep 29, 2022 7:54 PM	

Identity Device Information Recent Events State Logs

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

Event	Value	Format	Last Received
IoTSensor	{"temp":32,"Humid":33}	json	a few seconds ago
IoTSensor	{"temp":58,"Humid":87}	json	a few seconds ago
IoTSensor	{"temp":35,"Humid":6}	json	a few seconds ago
IoTSensor	{"temp":86,"Humid":24}	json	a few seconds ago

DEVICE EVENT PAYLOAD:



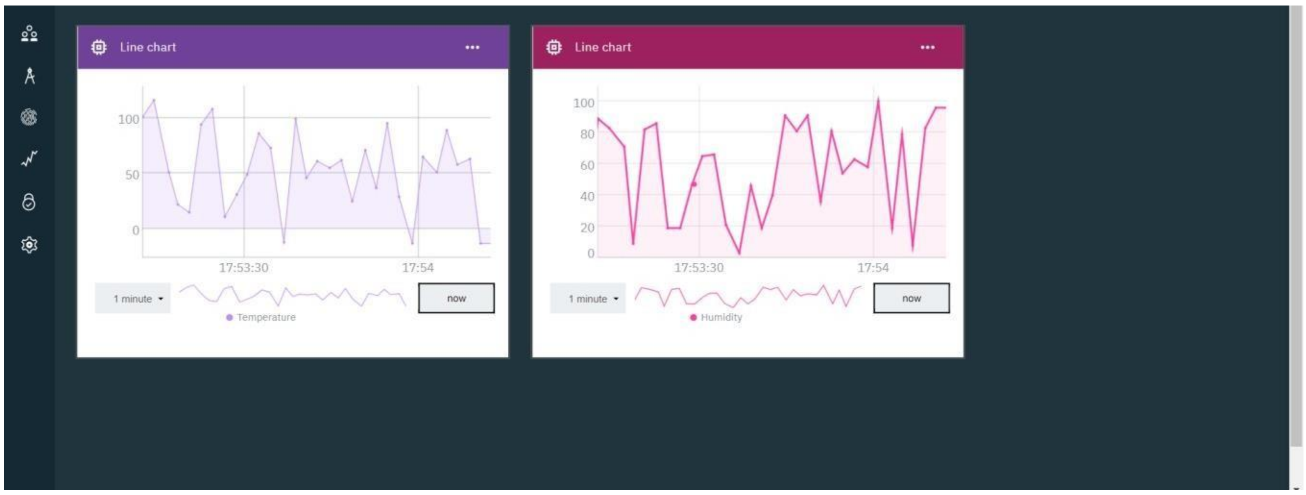
Event Payload

Event Name IoTSensor

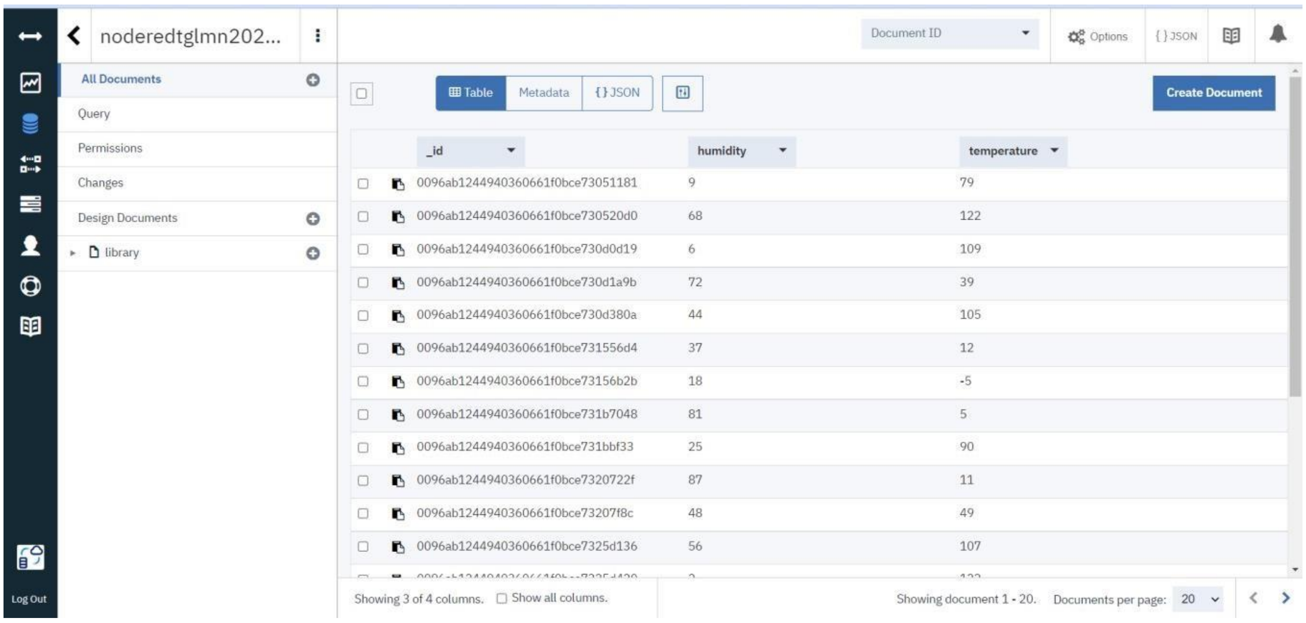
Time Received Nov 14, 2022 11:31 AM

```
{
  "temp": 32,
  "Humid": 33
}
```

DEVICE- BOARD:



IBM CLOUDANT DB LOG:



_id	humidity	temperature
0096ab1244940360661f0bce73051181	9	79
0096ab1244940360661f0bce730520d0	68	122
0096ab1244940360661f0bce730d0d19	6	109
0096ab1244940360661f0bce730d1a9b	72	39
0096ab1244940360661f0bce730d380a	44	105
0096ab1244940360661f0bce731556d4	37	12
0096ab1244940360661f0bce73156b2b	18	-5
0096ab1244940360661f0bce731b7048	81	5
0096ab1244940360661f0bce731bbf33	25	90
0096ab1244940360661f0bce7320722f	87	11
0096ab1244940360661f0bce73207f8c	48	49
0096ab1244940360661f0bce7325d136	56	107

Showing 3 of 4 columns. Show all columns. Showing document 1 - 20. Documents per page: 20