Sprint 2

Date	09 November 2022
Team ID	PNT2022TMID54458
Project Name	Hazardous Area Monitoring for Industrial Plant powered by IoT

In this sprint, we are getting Temperature and Humidity as input from the ESP32 using wokwi which is considered to get the input from the environment condition.

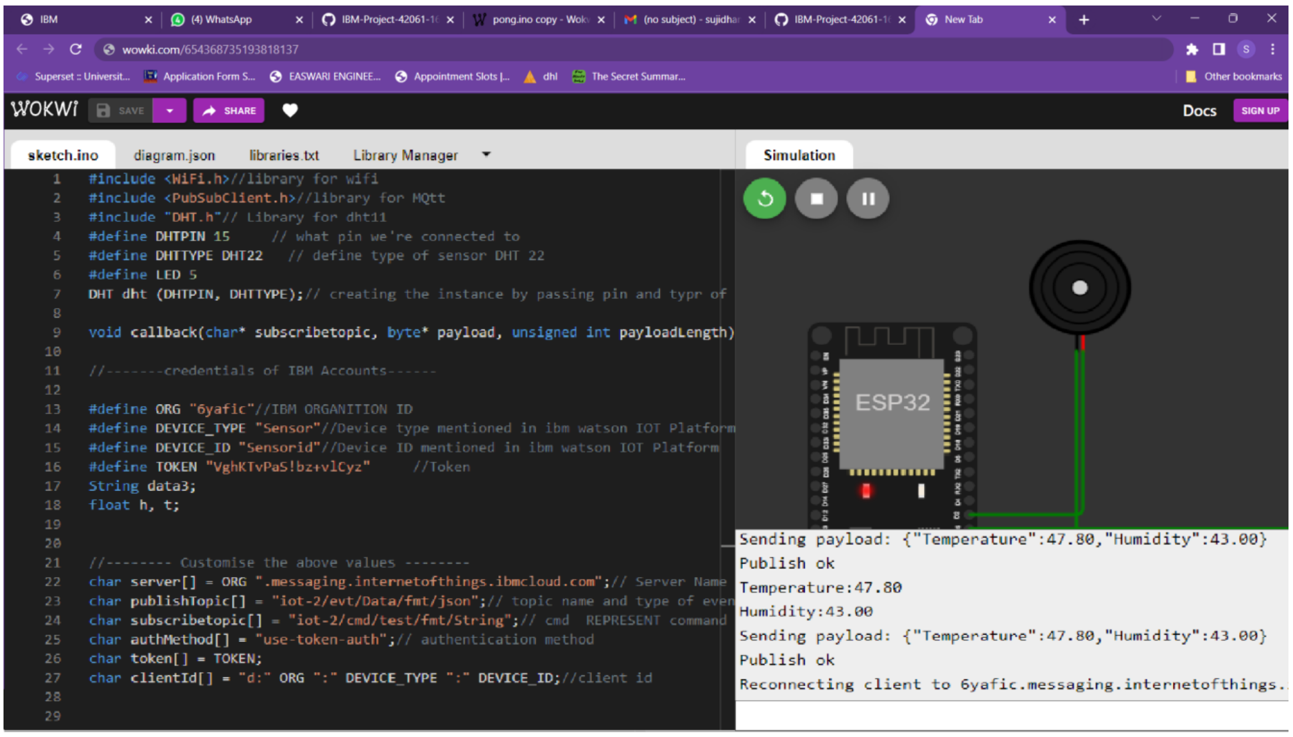
Solution:

```
#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 22
#define LED 5
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 "6yafic"//IBM ORGANITION ID
#define DEVICE_TYPE "Sensor"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "Sensorid"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "VghKTvPaS!bz+vlCyz"     //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/test/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("Temperature:");
  Serial.println(t);
  Serial.print("Humidity:");
  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 = "{\"Temperature\":";
  payload += temp;
  payload += "," "\"Humidity\":";
  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=="Alarmon")
 {
 Serial.println(data3);
 digitalWrite(LED,HIGH);

 }

 else
 {
   Serial.println(data3);
   digitalWrite(LED,LOW);

 }
data3="";
}
```

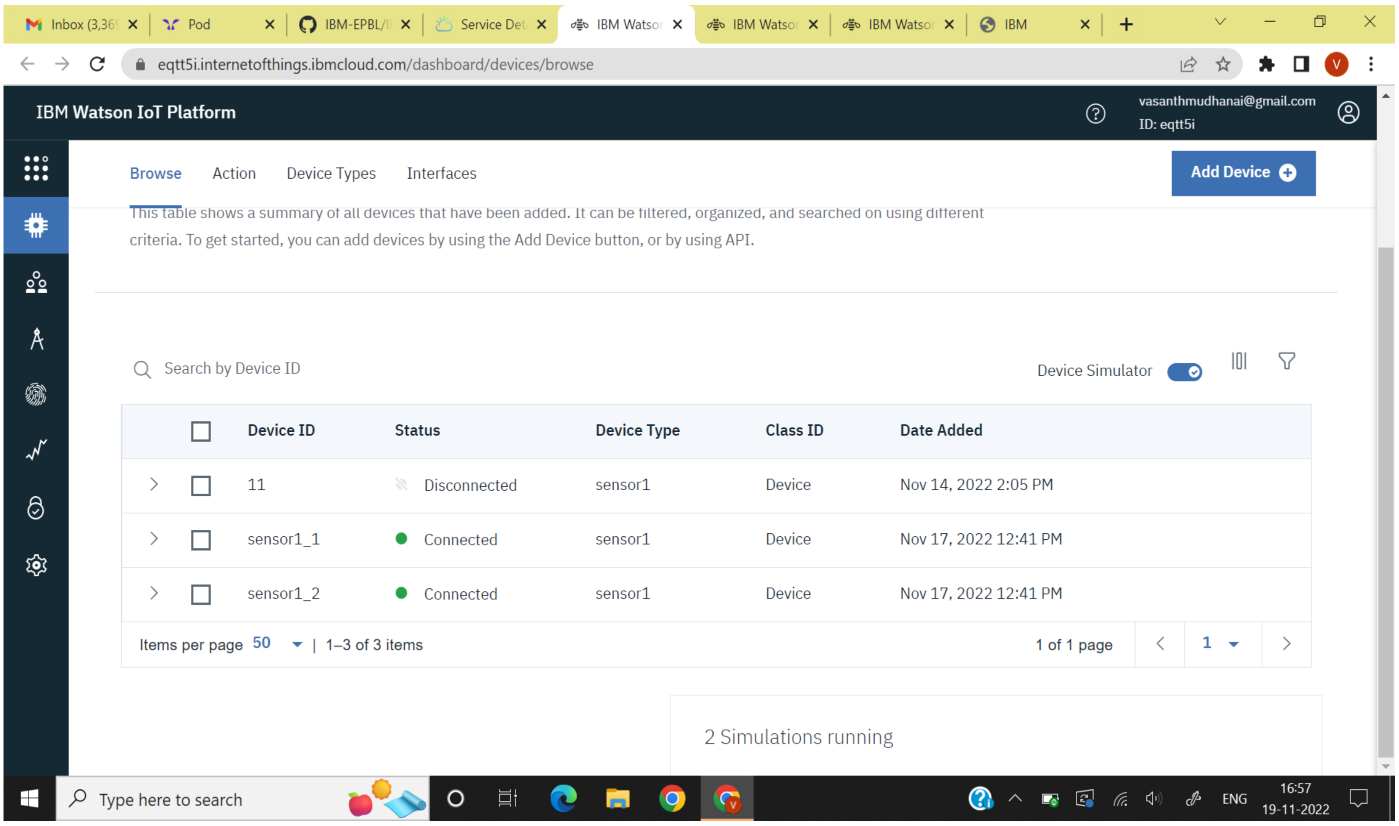
Data gathering from sensors (wokwi)



```
#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 22
#define LED 5
DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and typr of

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

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

#define ORG "6yafic"//IBM ORGANITION ID
#define DEVICE_TYPE "Sensor"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "Sensorid"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "VghKTvPaS!bz+vlCyz"     //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 even
char subscribetopic[] = "iot-2/cmd/test/fmt/String";// cmd  REPRESENT command
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id


```

```
Sending payload: {"Temperature":47.80,"Humidity":43.00}
Publish ok
Temperature:47.80
Humidity:43.00
Sending payload: {"Temperature":47.80,"Humidity":43.00}
Publish ok
Reconnecting client to 6yafic.messaging.internetofthings.
```

Uploaded data in Cloud from sensors (wokwi - ESP32)



This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different criteria. To get started, you can add devices by using the Add Device button, or by using API.

Device ID	Status	Device Type	Class ID	Date Added
11	Disconnected	sensor1	Device	Nov 14, 2022 2:05 PM
sensor1_1	● Connected	sensor1	Device	Nov 17, 2022 12:41 PM
sensor1_2	● Connected	sensor1	Device	Nov 17, 2022 12:41 PM

Items per page 50 | 1–3 of 3 items   1 of 1 page   1

2 Simulations running