SPRINT - 03

Date :	07 November 2022
Team ID :	PNT2022TMID37707
Project Name	SIGNS WITH SMART CONNECTIVITY FOR BETTER ROAD SAFETY

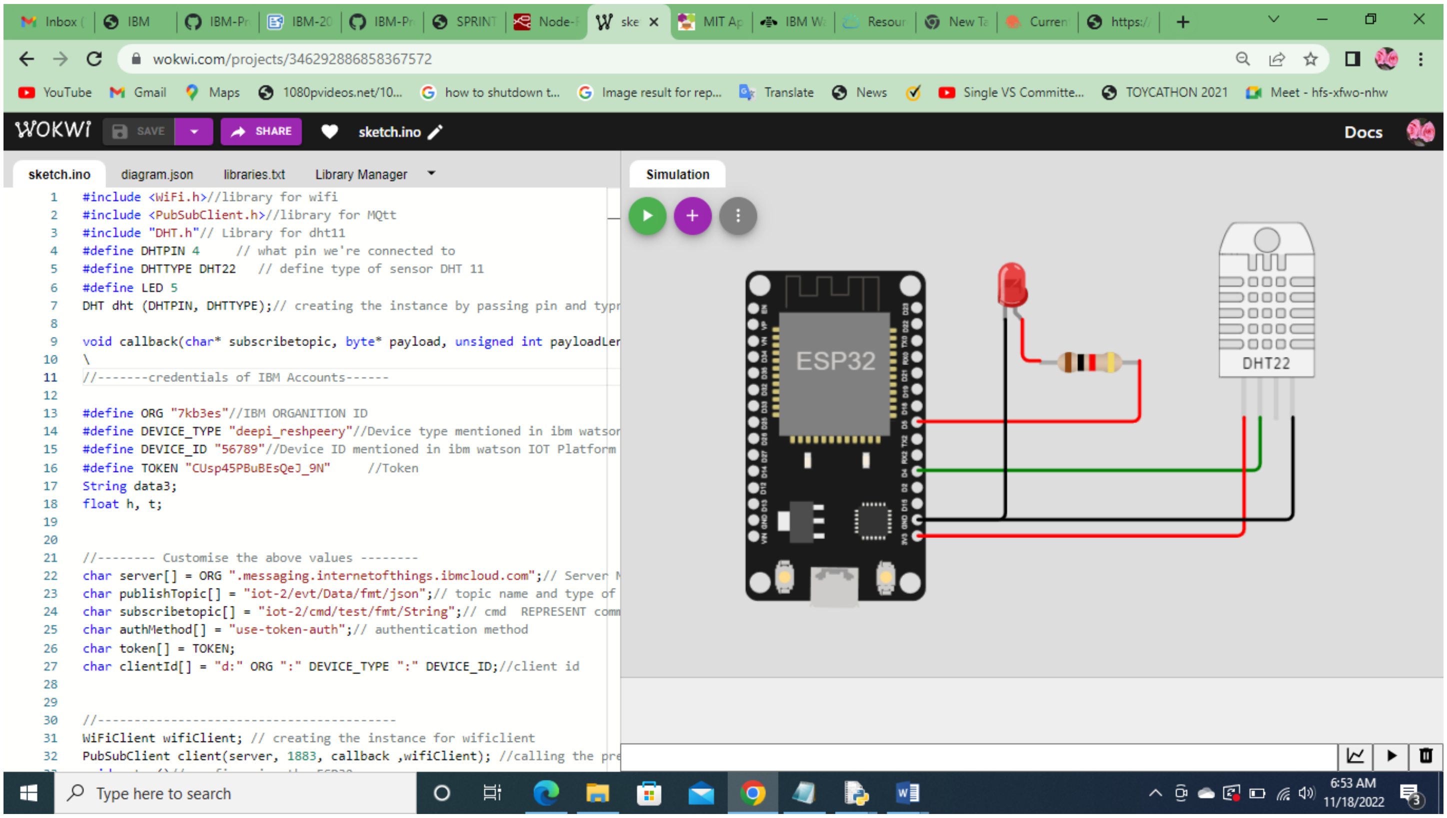
SPRINT GOAL:

Integrate the hardware to be able to access thecloud functions and provide inputs to the same.

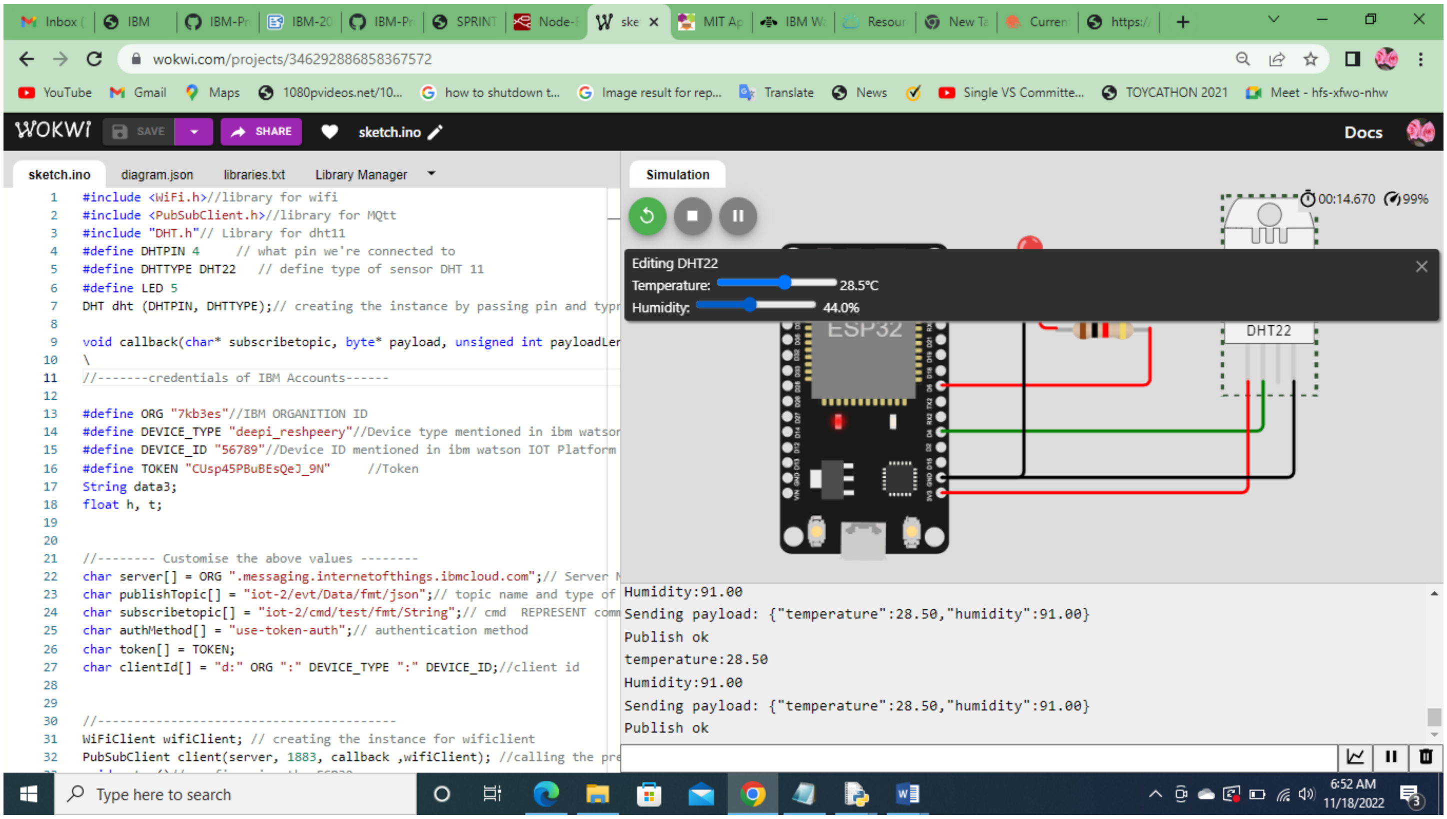
POGRAM 01:

AIM: To find the Temperature and Humidity DHT22 and ESP32

PLATFORM: WOKWI

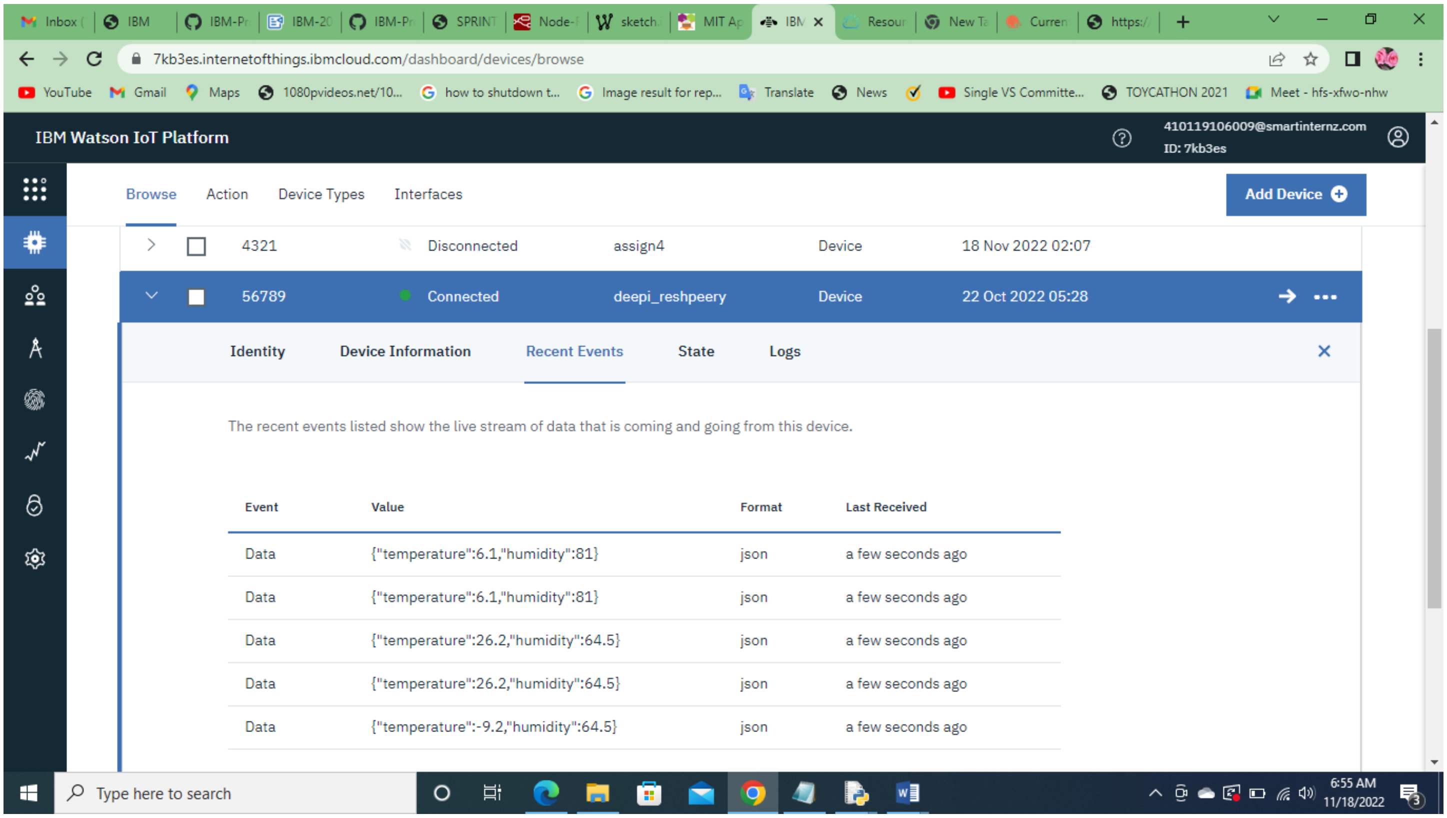


```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#include "DHT.h"// Library for dht11
#define DHTPIN 4     // what pin we're connected to
#define DHTTYPE DHT22   // define type of sensor DHT 11
#define LED 5
DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and typr

void callback(char* subscribetopic, byte* payload, unsigned int payloadLer
\
//-------credentials of IBM Accounts------

#define ORG "7kb3es"//IBM ORGANITION ID
#define DEVICE_TYPE "deepi_reshpeery"//Device type mentioned in ibm watsor
#define DEVICE_ID "56789"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "CUsp45PBuBEsQeJ_9N"     //Token
String data3;
float h, t;


//-------- Customise the above values --------
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server N
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of
char subscribetopic[] = "iot-2/cmd/test/fmt/String";// cmd  REPRESENT comm
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 pre
```



```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#include "DHT.h"// Library for dht11
#define DHTPIN 4     // what pin we're connected to
#define DHTTYPE DHT22   // define type of sensor DHT 11
#define LED 5
DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and typr

void callback(char* subscribetopic, byte* payload, unsigned int payloadLer
\
//-------credentials of IBM Accounts------

#define ORG "7kb3es"//IBM ORGANITION ID
#define DEVICE_TYPE "deepi_reshpeery"//Device type mentioned in ibm watsor
#define DEVICE_ID "56789"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "CUsp45PBuBEsQeJ_9N"     //Token
String data3;
float h, t;


//-------- Customise the above values --------
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server N
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of
char subscribetopic[] = "iot-2/cmd/test/fmt/String";// cmd  REPRESENT comm
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 pre
```

```
Humidity:91.00
Sending payload: {"temperature":28.50,"humidity":91.00}
Publish ok
temperature:28.50
Humidity:91.00
Sending payload: {"temperature":28.50,"humidity":91.00}
Publish ok
```



4321	Disconnected	assign4	Device	18 Nov 2022 02:07
56789	Connected	deepi_reshpeery	Device	22 Oct 2022 05:28

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
Data	{"temperature":6.1,"humidity":81}	json	a few seconds ago
Data	{"temperature":6.1,"humidity":81}	json	a few seconds ago
Data	{"temperature":26.2,"humidity":64.5}	json	a few seconds ago
Data	{"temperature":26.2,"humidity":64.5}	json	a few seconds ago
Data	{"temperature":-9.2,"humidity":64.5}	json	a few seconds ago

PYTHON CODE:

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#include "DHT.h"// Library for dht11
#define DHTPIN 4     // what pin we're connected to
#define DHTTYPE DHT22   // define type of sensor DHT 11
#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 "7kb3es"//IBM ORGANITION ID
#define DEVICE_TYPE "deepi_reshpeery"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "56789"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "CUsp45PBuBEsQeJ_9N"     //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=="lighton")
  {
Serial.println(data3);
digitalWrite(LED,HIGH);
  }
  else
  {
Serial.println(data3);
digitalWrite(LED,LOW);

  }
data3="";
}
```

Output link :

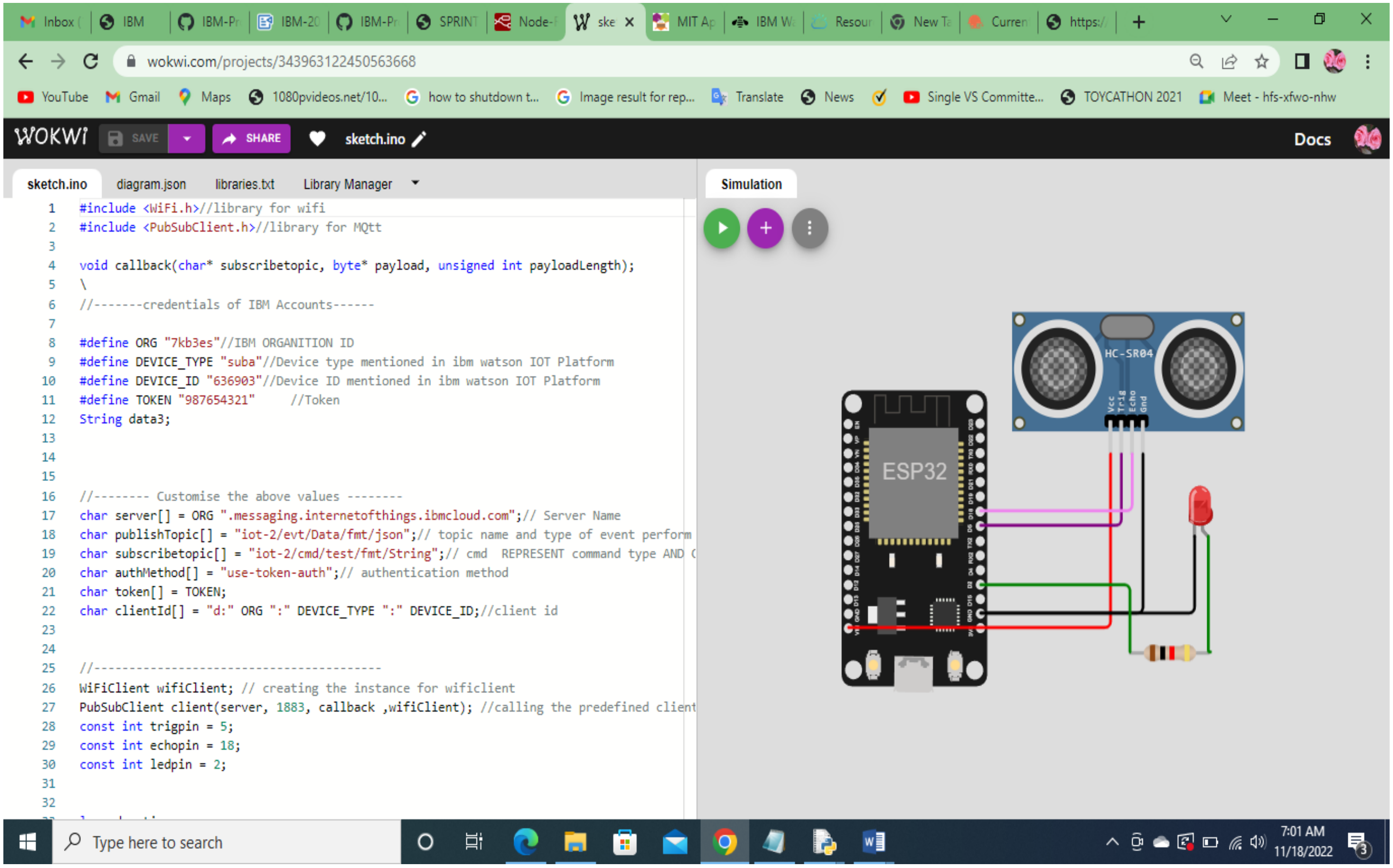
https://wokwi.com/projects/346292886858367572

By using this Wokwi we determined the Temperature and Humidity for better road safety.

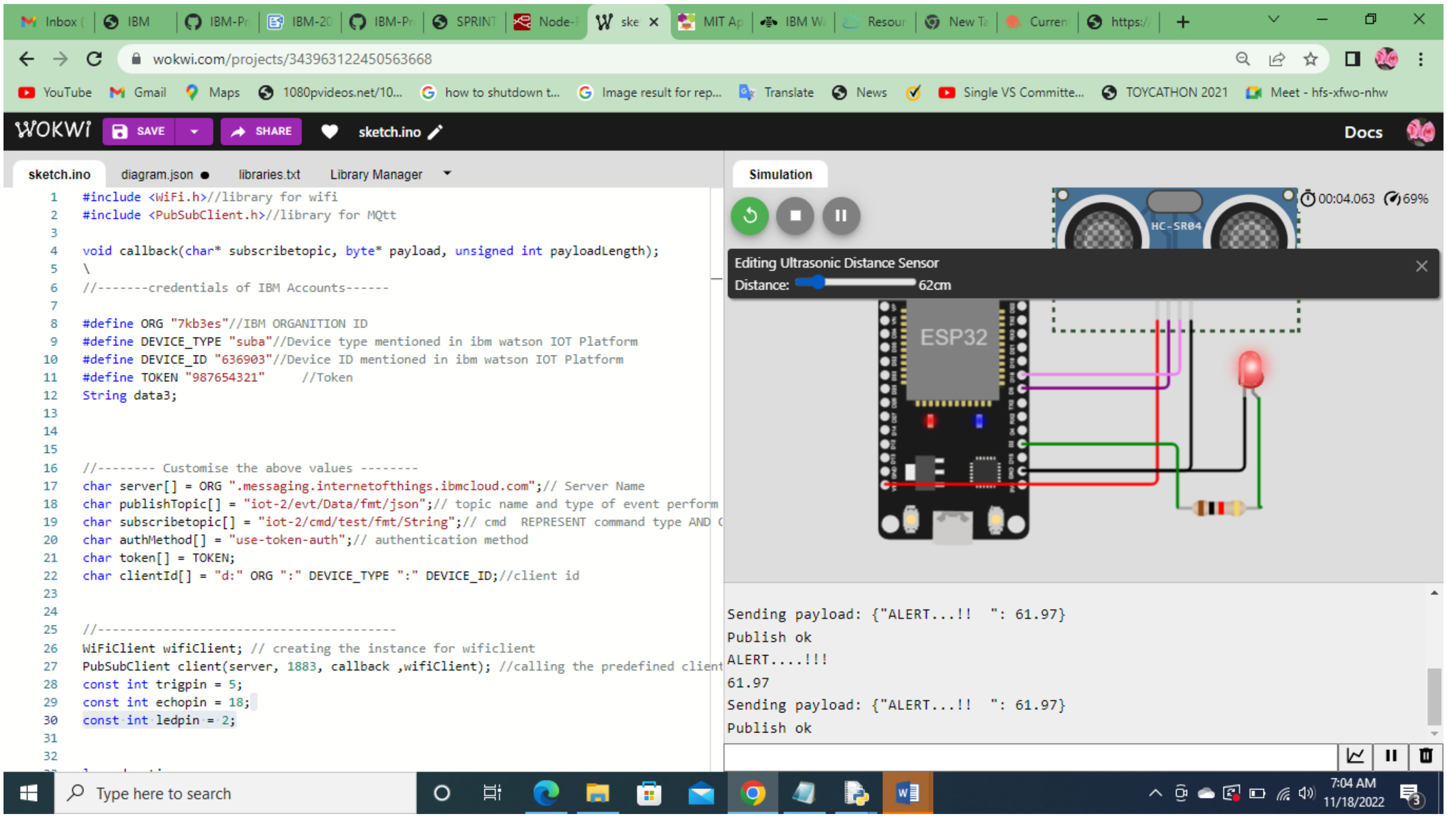
POGRAM 02

AIM: Write code and connection in Wowki for ultrasonic sensor. Whenever distance is less than 100 cms send “Alert” to IBM cloud and display in device recent events by using ESP32

PLATFORM: WOKWI



```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
\
//-------credentials of IBM Accounts------

#define ORG "7kb3es"//IBM ORGANITION ID
#define DEVICE_TYPE "suba"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "636903"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "987654321"     //Token
String data3;



//-------- 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
const int trigpin = 5;
const int echopin = 18;
const int ledpin = 2;
```

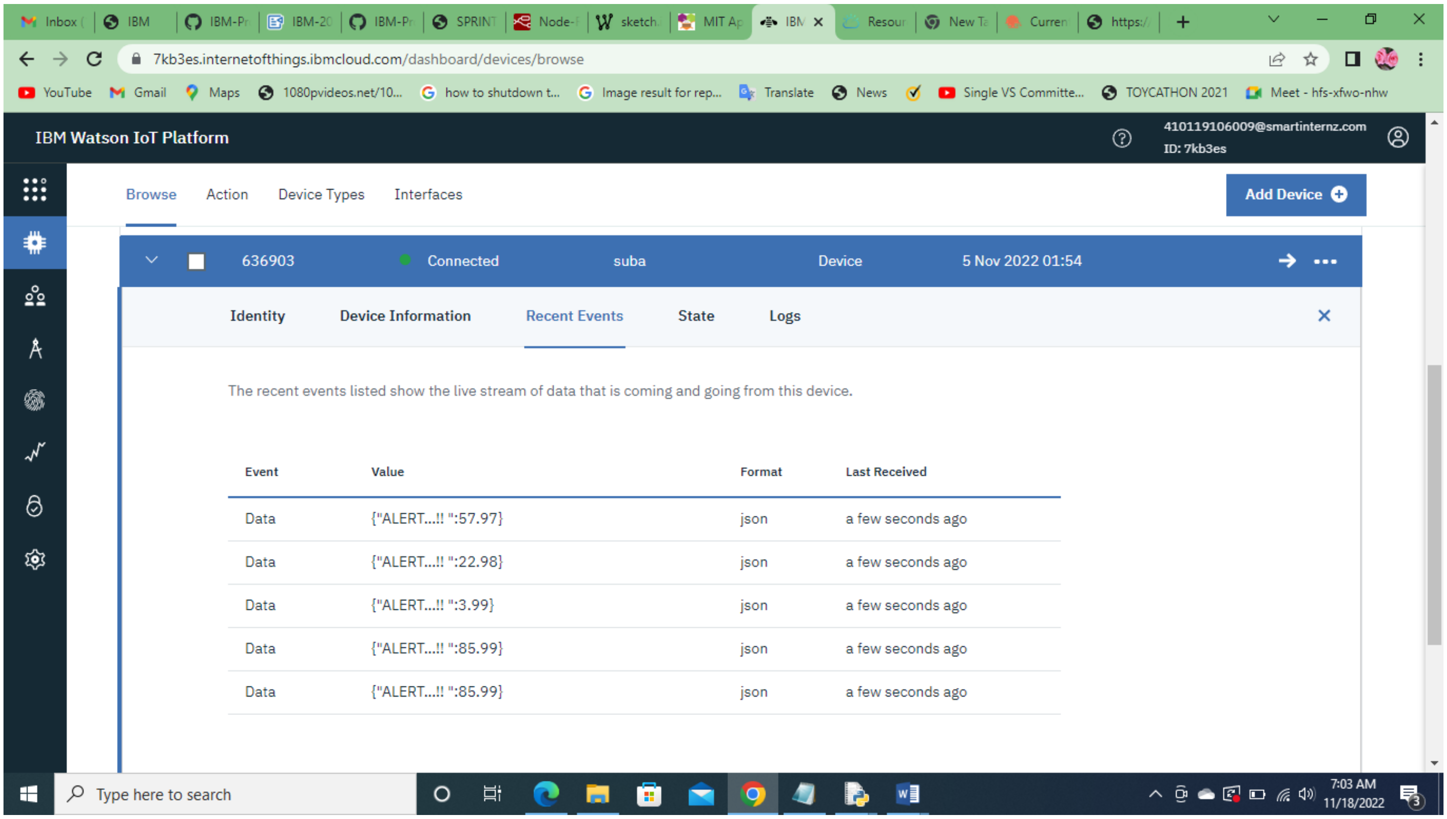


```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
\
//-------credentials of IBM Accounts------

#define ORG "7kb3es"//IBM ORGANITION ID
#define DEVICE_TYPE "suba"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "636903"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "987654321"     //Token
String data3;



//-------- 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
const int trigpin = 5;
const int echopin = 18;
const int ledpin = 2;
```

Editing Ultrasonic Distance Sensor

Distance: 62cm

```
Sending payload: {"ALERT...!!  ": 61.97}
Publish ok
ALERT....!!!
61.97
Sending payload: {"ALERT...!!  ": 61.97}
Publish ok
```



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

Event	Value	Format	Last Received
Data	{"ALERT...!! ":57.97}	json	a few seconds ago
Data	{"ALERT...!! ":22.98}	json	a few seconds ago
Data	{"ALERT...!! ":3.99}	json	a few seconds ago
Data	{"ALERT...!! ":85.99}	json	a few seconds ago
Data	{"ALERT...!! ":85.99}	json	a few seconds ago

PYHTON CODE:

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
\
//-------credentials of IBM Accounts------

#define ORG "7kb3es"//IBM ORGANITION ID
#define DEVICE_TYPE "suba"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "636903"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "987654321"     //Token
String data3;



//-------- 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
const int trigpin = 5;
const int echopin = 18;
const int ledpin = 2;


long duration ;
float distance;
#define sound_speed 0.034
void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  pinMode(trigpin, OUTPUT);
  pinMode(echopin, OUTPUT);
  pinMode(ledpin, OUTPUT);
  wificonnect();
  mqttconnect();
}

void loop() {
  digitalWrite(trigpin, LOW);
  digitalWrite(trigpin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigpin, LOW);

  duration= pulseIn(echopin,HIGH);
  distance = duration * sound_speed /2;
  if(distance<=100){
    PublishData(distance);
  delay(1000);
  if (!client.loop()) {
    mqttconnect();
  }
    digitalWrite(ledpin, HIGH);
    Serial.println("ALERT....!!!");
    Serial.println(distance);
  }
  else
  {
    digitalWrite(ledpin, LOW);
  }
  // put your main code here, to run repeatedly:
  delay(10); // this speeds up the simulation
}

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

```
void PublishData(float distance) {
  mqttconnect();//function call for connecting to ibm

    // creating the String in in form JSon to update the data to ibm cloud
  String payload = "{\"ALERT...!!  \": ";
  payload += distance;
  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);
  }
  else
  {
     Serial.println(data3);
  }
data3="";
}
```

Output link :

https://wokwi.com/projects/343963122450563668

Conclusion:

Here I showed the ALERT and DISTANCE in IBM cloud when vehicle has the distance is less than 100 cms

THANK YOU..!!