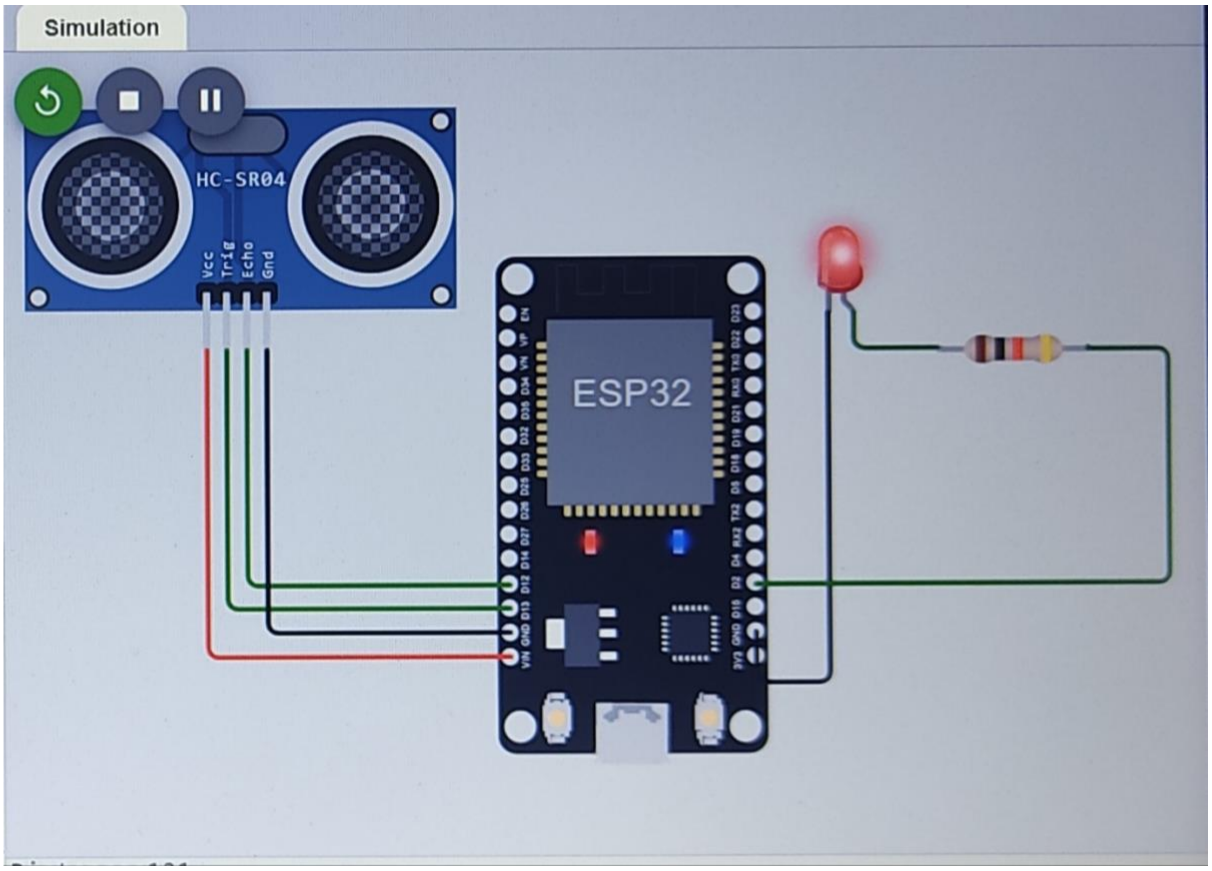
ASSIGNMENT 4

NAME	JIBISHA.P
REGISTER NUMBER	962819106024

Write code and connections in wokwi for ultrasonic sensor. Whenever distance is less than 100 cms send "alert" to ibm cloud and display in device recent events.Upload document with wokwi share link and images of ibm cloud.

CIRCUIT :



PROGRAM:

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

```
#include <PubSubClient.h>//library for MQtt
#define echoPin 12    // what pin we're connected to
#define trigPin 13   // define type of sensor DHT 11
#define LED 2
long duration;

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

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

#define ORG "ywyg3r"//IBM ORGANITION ID
#define DEVICE_TYPE "Jibisha2001"//Device type mentioned in ibm
watson IOT Platform
#define DEVICE_ID "Jibisha01"//Device ID mentioned in ibm watson
IOT Platform
#define TOKEN "zp2JUc7Y-0CUItpuAs"     //Token
String data3;
int distance;
//-------- 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
```

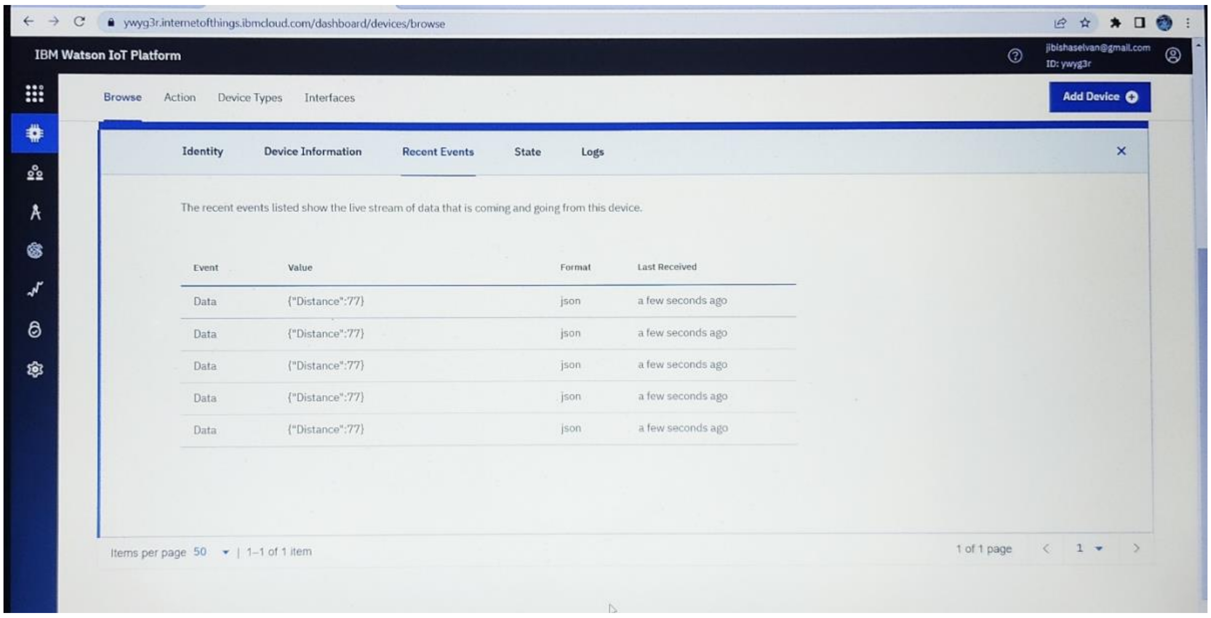
```
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd
REPRESENT command type AND COMMAND IS TEST OF FORMAT
STRING

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);

  pinMode(trigPin,OUTPUT);

  pinMode(echoPin,INPUT);

        pinMode(LED,OUTPUT);

        delay(10);

        Serial.println();

        wificonnect();

  mqttconnect();

}


void loop()// Recursive Function
```

```
{
  digitalWrite(trigPin,LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin,HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin,LOW);
  duration=pulseIn(echoPin,HIGH);
  distance=duration*0.034/2;
  Serial.println("duration:"+String(distance)+"cm");
  if(distance<100)
  {
  digitalWrite(LED,HIGH);
  }
  else{
  digitalWrite(LED,LOW);
  }
          Serial.print("Distance:");
    Serial.println(distance);



          PublishData(distance);
          delay(1000);
                if(!client.loop())
  {
    mqttconnect();
  }

}

/*...................................retrieving to Cloud..............................*/
void PublishData(int distance) {
           mqttconnect();//function call for connecting to ibm
           /*
              creating the String in in form JSon to update the data to ibm
cloud
           */
           String payload = "{\"Distance\":";
           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);
digitalWrite(LED,HIGH);

  }



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



  }
data3="";
```

}

OUTPUT:



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

Event	Value	Format	Last Received
Data	{"Distance":77}	json	a few seconds ago
Data	{"Distance":77}	json	a few seconds ago
Data	{"Distance":77}	json	a few seconds ago
Data	{"Distance":77}	json	a few seconds ago
Data	{"Distance":77}	json	a few seconds ago

Items per page 50 | 1–1 of 1 item　　1 of 1 page　1

https://wokwi.com/projects/347233177012535890