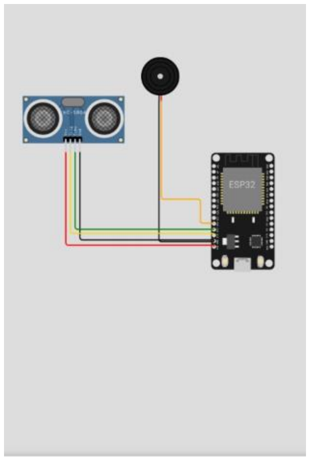
. # Assignment 4

Assignment Date	02-11-2022
Student Name	Saahitha Jinu G S
Student Register Number	962819106034

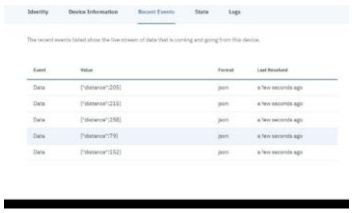
Circuit Diagram:



Program:

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#define trigPin 15
#define echoPin 2     // what pin we're connected to
#define BUZZER 4


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

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

#define ORG "xkxvud"//IBM ORGANITION ID
#define DEVICE_TYPE "Trackdevicetype"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "new1"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "Sw()41Ao5gzAj!CK7-"
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/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
{
  pinMode(echoPin,OUTPUT);
  pinMode(trigPin,INPUT);
  pinMode(BUZZER,OUTPUT);
  Serial.begin(115200);
  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
}

void loop()// Recursive Function
{
  digitalWrite(trigPin,LOW);
  delayMicroseconds(3);
  digitalWrite(trigPin,HIGH);
  delayMicroseconds(10);
  distance=random(0,300);
  if(distance<100)
  {
    digitalWrite(BUZZER, HIGH);
  }
  else
  {
    digitalWrite(BUZZER, 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=="alarm")
  {
  Serial.println(data3);
  digitalWrite(BUZZER,HIGH);

  }

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

  }
data3="";


}
```



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":205}	json	a few seconds ago
Data	{"distance":215}	json	a few seconds ago
Data	{"distance":258}	json	a few seconds ago
Data	{"distance":79}	json	a few seconds ago
Data	{"distance":152}	json	a few seconds ago

https://wokwi.com/projects/347175315941884498