Assignment - 4

ESP 32 – Ultrasonic Sensor

Assignment Date	3 NOVEMBER 2022
Student Name	MAHASHREE S
Student Roll Number	621319106050
Maximum Marks	2 Marks

Question-1:

Write code and Connection in wokwi for ultrasonic sensor. Whenever distance is less than 100cms send “alert” to the ibm cloud and display in device recent events.

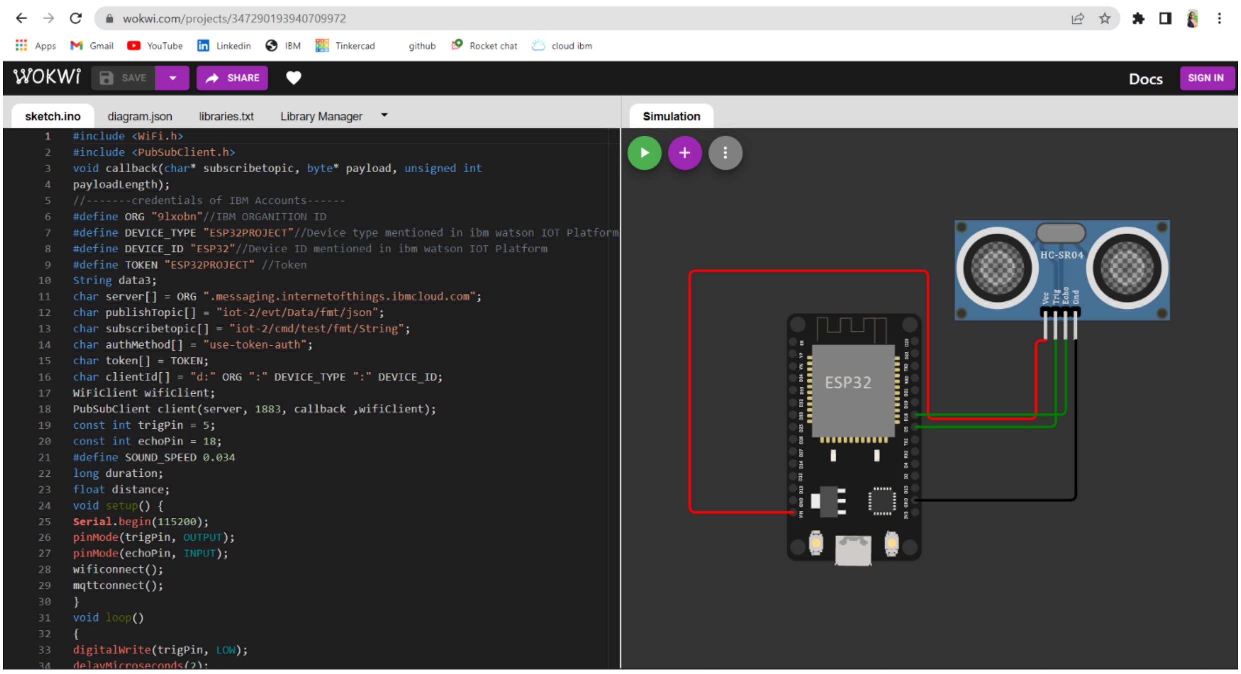
Solution:

Program:

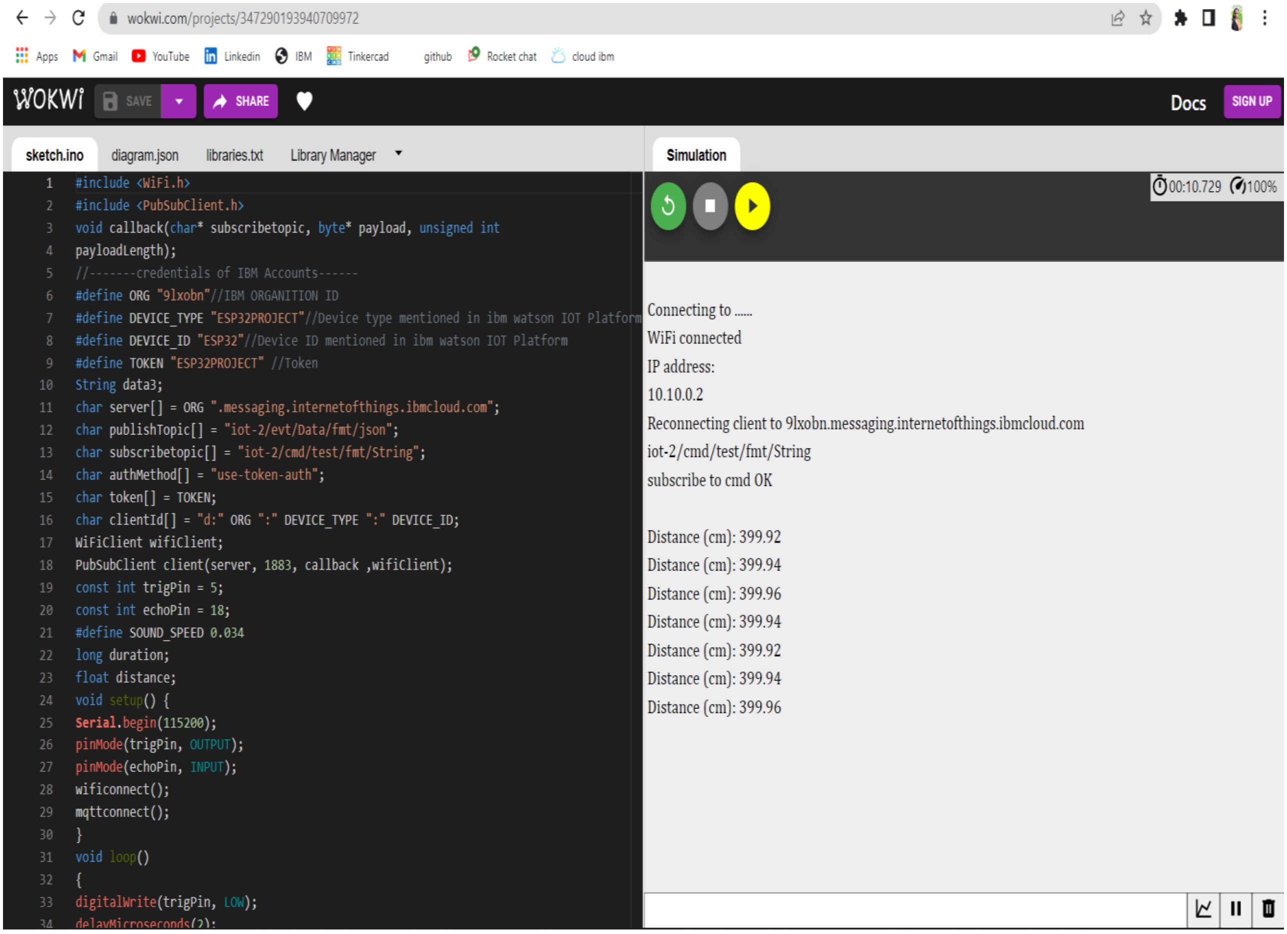
```
#include <WiFi.h>
#include <WiFiClient.h>
#include <PubSubClient.h>
const int trigPin = 5;
const int echoPin = 18;
//define sound speed in cm/uS
#define SOUND_SPEED 0.034
#define CM_TO_INCH 0.393701
long duration;
float distanceCm;
float distanceInch;


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

#define ORG “b31tni”//IBM ORGANITION ID
#define DEVICE_TYPE "Assignment4"//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "assignment"//Device ID mentioned in ibm watson IOT
Platform#define TOKEN "6r?TKCIuy+okJ?9B+7" //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);

void setup() {
  Serial.begin(115200); // Starts the serial communication
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
  Serial.println();
  wificonnect();
  mqttconnect();

}

void loop() {
  // Clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);

  // Calculate the distance
  distanceCm = duration * SOUND_SPEED/2;

  // Convert to inches
  distanceInch = distanceCm * CM_TO_INCH;

  // Prints the distance in the Serial Monitor
  Serial.print("Distance (cm): ");
  Serial.println(distanceCm);
  Serial.print("Distance (inch): ");
  Serial.println(distanceInch);

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

  void PublishData(float Cm) {
```

```
  mqttconnect();//function call for connecting to ibm
  /*
     creating the String in in form JSon to update the data to ibm cloud
  */
  String payload = "{\"Distance (cm)\":";
  payload += Cm;
  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];
  }
}
```

Wokwi Simulation:



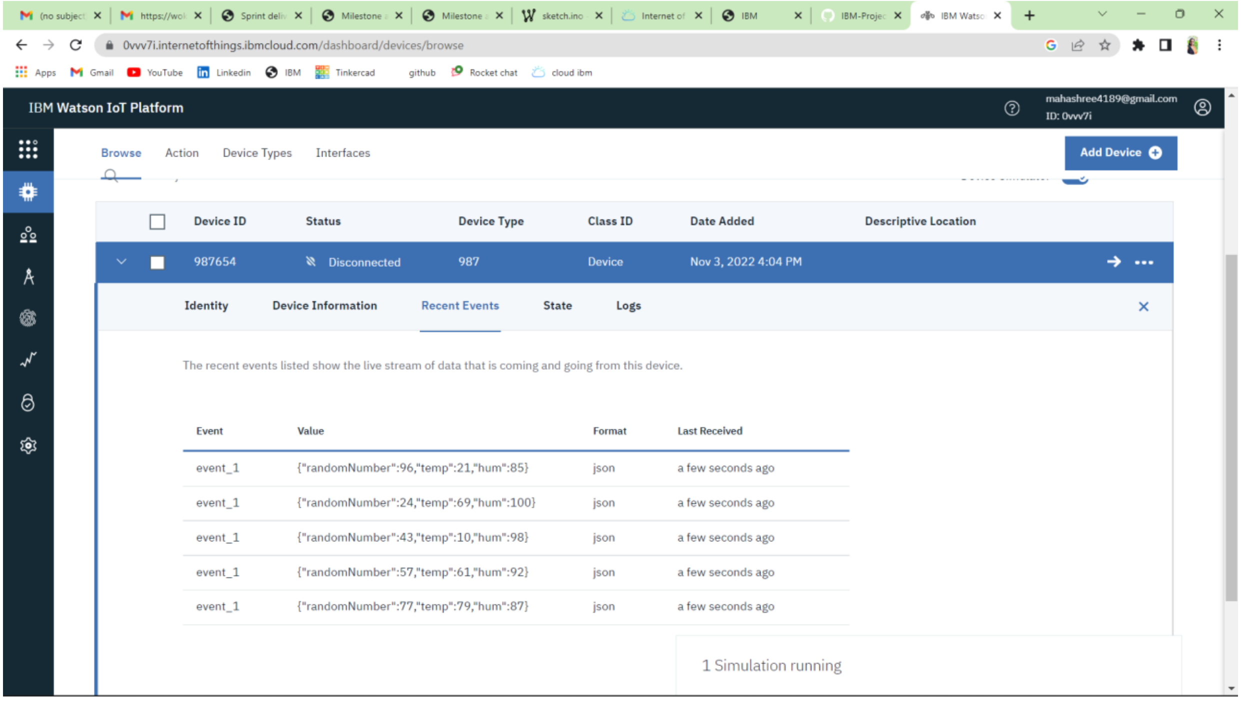
```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);
//-------credentials of IBM Accounts------
#define ORG "9lxobn"//IBM ORGANITION ID
#define DEVICE_TYPE "ESP32PROJECT"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "ESP32"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "ESP32PROJECT" //Token
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/fmt/json";
char subscribetopic[] = "iot-2/cmd/test/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
WiFiClient wifiClient;
PubSubClient client(server, 1883, callback ,wifiClient);
const int trigPin = 5;
const int echoPin = 18;
#define SOUND_SPEED 0.034
long duration;
float distance;
void setup() {
Serial.begin(115200);
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
wificonnect();
mqttconnect();
}
void loop()
{
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
```



```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);
//-------credentials of IBM Accounts------
#define ORG "9lxobn"//IBM ORGANITION ID
#define DEVICE_TYPE "ESP32PROJECT"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "ESP32"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "ESP32PROJECT" //Token
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/fmt/json";
char subscribetopic[] = "iot-2/cmd/test/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
WiFiClient wifiClient;
PubSubClient client(server, 1883, callback ,wifiClient);
const int trigPin = 5;
const int echoPin = 18;
#define SOUND_SPEED 0.034
long duration;
float distance;
void setup() {
Serial.begin(115200);
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
wificonnect();
mqttconnect();
}
void loop()
{
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
```

```
Connecting to ......
WiFi connected
IP address:
10.10.0.2
Reconnecting client to 9lxobn.messaging.internetofthings.ibmcloud.com
iot-2/cmd/test/fmt/String
subscribe to cmd OK

Distance (cm): 399.92
Distance (cm): 399.94
Distance (cm): 399.96
Distance (cm): 399.94
Distance (cm): 399.92
Distance (cm): 399.94
Distance (cm): 399.96
```

IoT Watson Platform:



0vvv7i.internetofthings.ibmcloud.com/dashboard/devices/browse

IBM Watson IoT Platform

mahashree4189@gmail.com
ID: 0vvv7i

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
987654	Disconnected	987	Device	Nov 3, 2022 4:04 PM	

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

Event	Value	Format	Last Received
event_1	{"randomNumber":96,"temp":21,"hum":85}	json	a few seconds ago
event_1	{"randomNumber":24,"temp":69,"hum":100}	json	a few seconds ago
event_1	{"randomNumber":43,"temp":10,"hum":98}	json	a few seconds ago
event_1	{"randomNumber":57,"temp":61,"hum":92}	json	a few seconds ago
event_1	{"randomNumber":77,"temp":79,"hum":87}	json	a few seconds ago

1 Simulation running

https://wokwi.com/projects/347290193940709972