

Assignment -4
Python Programming

Assignment Date	28 October 2022
Student Name	Mr. GIRIDHARAN P
Student Roll Number	310619205030
Maximum Marks	2 Marks

Question-1:

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.

Solution:

sketch.ino diagram.json libraries.txt Library Manager

Sim

```

1 #include <WiFi.h> //library for wifi
2 #include <PubSubClient.h> //library for MQTT
3 void callback(char* subscribetopic, byte* payload, unsigned int payloadlength);
4 //-----credentials of IBM Account-----
5 #define ORG "izyy6o" // IBM ORGANIZATION ID
6 #define DEVICE_TYPE "iotdeviceproject" //DEVICE TYPE MENTIONED IN IOT WATSON PLATFORM
7 #define DEVICE_ID "229714" //DEVICE ID MENTIONED IN IOT WATSON PLATFORM
8 #define TOKEN "24681012" //Token
9 String data3;
10 float dist;
11 //-----customize the above value-----
12 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; //server name
13 char publishtopic[] = "ultrasonic/evt/Data/fmt/json"; //topic name and type of event perform
14 //and format in which data to be send*/
15 char subscribetopic[] = "ultrasonic/cmd/test/fmt/String"; //cmd REPRESENT Command type and
16 //COMMAND IS TEST OF FORMAT STRING*/
17 char authMethod[] = "use-token-auth"; //authentication method
18 char token[] = TOKEN;
19 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //CLIENT ID
20 //-----
21 WiFiClient wificlient; // creating an instance for wificlient
22 PubSubClient client(server, 1883, callback, wificlient); //calling the predefined client id
23 //by passing parameter like server id, port and wificredential*/
24 int LED = 4;
25 int trig = 5;
26 int echo = 18;
27 void setup()
28 {
29   Serial.begin(115200);
30   pinMode(trig, OUTPUT);

```

Co
nn
ec
ti
ng
to

← → ↺ wokwi.com/projects/346566226034557523

WOKWI

SAVE

SHARE

♥

Docs

V

sketch.ino ● diagram.json libraries.txt ● Library Manager

61Serial.println("no object is near");
62object="Near";
63}
64else
65{
66digitalWrite(LED,LOW);
67Serial.println("no object found");
68object="No";
69}
70String payload="{\"distance\":";
71payload +=dist;
72payload +=",\" \"object\":\":";
73payload += object;
74payload += "\":";
75
76Serial.print("Sending payload: ");
77Serial.println(payload);
78if(client.publish(publishtopic, (char*) payload.c_str())){
79Serial.println("Publish ok");/* If its sucessfully upload data on the cloud then it will print
80publish ok in serial monitor or else it will print poblish failed*/
81}
82else{
83Serial.println("Publish failed");
84}
85
86void mqttconnect(){
87if(!client.connected()){
88Serial.print("Reconnecting client to ");
89Serial.println(server);
90while(!client.connect(clientid,authMethod, token)){
91Serial.print(".");
92delay(500);
93

Simu

▶

Co
nn
ec
ti
ng
to

← → ↺ wokwi.com/projects/346566226034557523

WOKWI

SAVE

SHARE

♥

Docs

V

sketch.ino ● diagram.json libraries.txt ● Library Manager

92}
93initManagedDevice();
94Serial.println();
95}
96}
97void wificonnect();//function defenition for wificonnect
98{
99Serial.println();
100Serial.print("Connecting to ");
101WiFi.begin("Wokwi-GUEST", "",6);//PASSING THE WIFI CREDENTIALS TO ESTABLISH CONNECTION
102while (WiFi.status() !=WL_CONNECTED){
103delay(500);
104Serial.print(".");
105}
106Serial.println("");
107Serial.println("WiFi connected");
108Serial.println("IP address");
109Serial.println(WiFi.localIP());
110}
111void initManagedDevice(){
112if(client.subscribe(subscribetopic)){
113Serial.println((subscribetopic));
114Serial.println("subscribe to cmd OK");
115}
116else{
117Serial.println("subscribe to cmd failed");
118}
119}
120void callback(char* subscribetopic,byte*payload,unsigned int payloadLength)
121{
122Serial.print("callback invoked for topic: ");
123Serial.println(subscribetopic);
124

Simu

▶

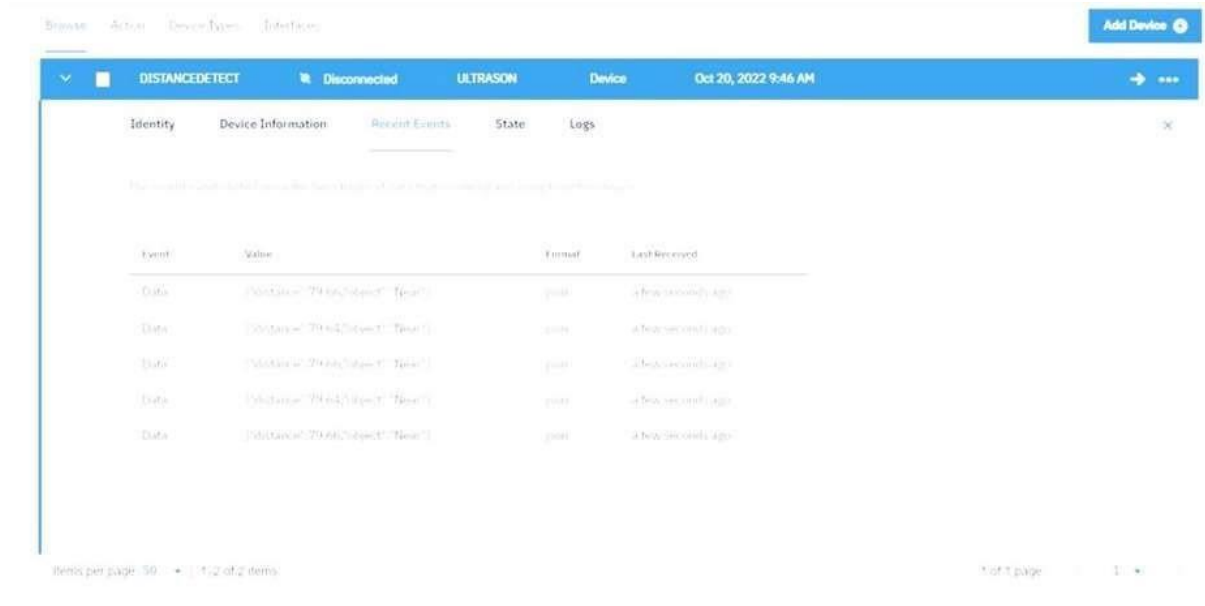
Co
nn
ec
ti
ng
to

```
123 for(int i=0; i< payloadLength; i++){
124   //Serial.print((char)payload[i]);
125   data3 +=(char)payload[i];
126 }
127 //Serial.println("dta: "+ data3);
128 //if(data3=="near")
129 //{
130 //Serial.println(data3);
131 //digitalwrite(LED,HIGH);
132 //}
133 //else
134 //{
135 //Serial.println(data3);
136 //digitalwrite(LED,LOW);
137 //}
138 data3="";
139 }
```

OUTPUT:
DATA IS SENT TO IBM CLOUD WHEN NO OBJECT IS DETECTED

Event	Value	Format	Last Received
Data	"Distance=79.64/object: "near"	json	4 hours 55 mins ago
Data	"Distance=79.64/object: "near"	json	4 hours 55 mins ago
Data	"Distance=79.64/object: "near"	json	4 hours 55 mins ago
Data	"Distance=79.64/object: "near"	json	4 hours 55 mins ago
Data	"Distance=79.64/object: "near"	json	4 hours 55 mins ago

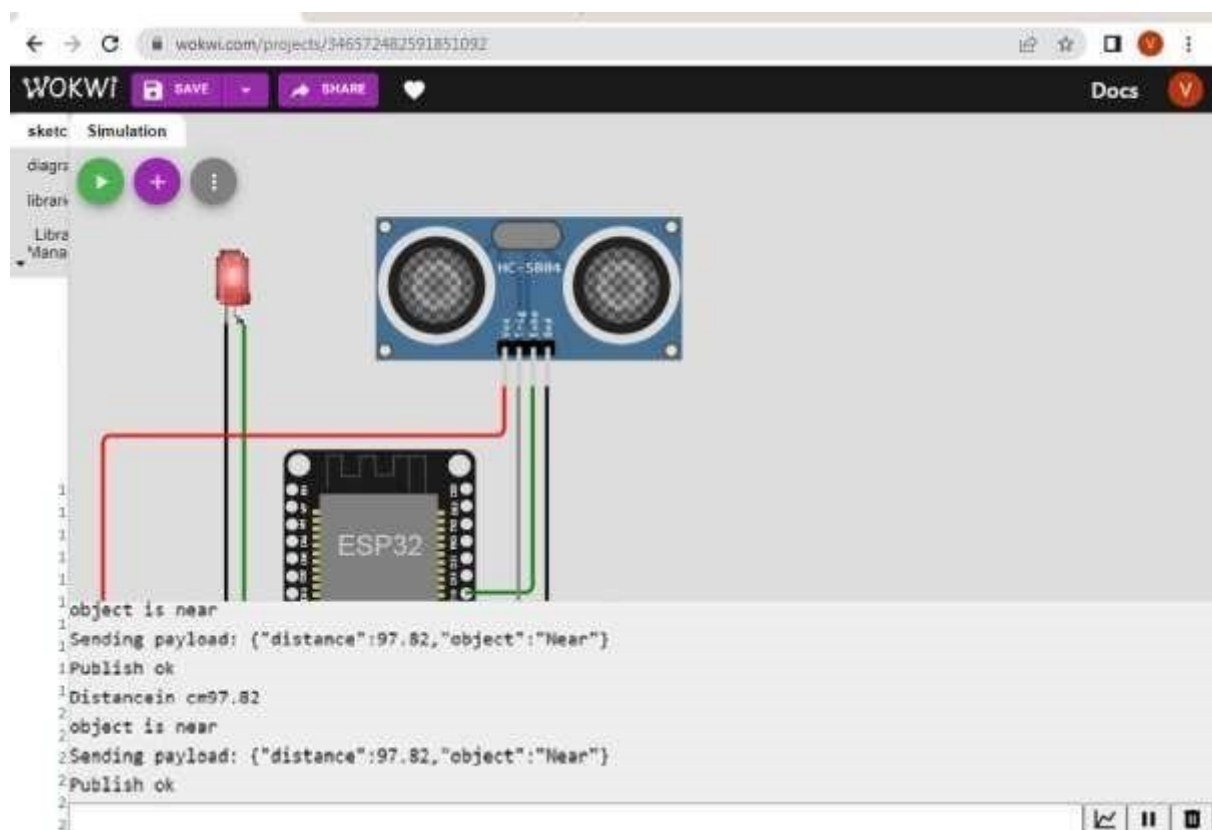
When no object is detected



The screenshot shows the Wokwi web interface for a project named "DISTANCEDETECT". The device is an "ULTRASON" sensor, currently "Disconnected". The interface includes tabs for "Identity", "Device Information", "Recent Events", "State", and "Logs". The "Recent Events" tab is active, displaying a table of events. The table has columns for "Event", "Value", "Format", and "Last Received". The events are all "Data" type, with values indicating a distance of 79.66 cm and an object status of "None". The "Last Received" times range from 4 hours and 54 minutes to 4 hours and 55 minutes ago. The interface also shows a "Add Device" button in the top right and a "1 of 1 page" indicator at the bottom right.

Event	Value	Format	Last Received
Data	{Distance: 79.66, object: "None"}	json	4 hours and 54 minutes ago
Data	{Distance: 79.66, object: "None"}	json	4 hours and 54 minutes ago
Data	{Distance: 79.66, object: "None"}	json	4 hours and 54 minutes ago
Data	{Distance: 79.66, object: "None"}	json	4 hours and 54 minutes ago
Data	{Distance: 79.66, object: "None"}	json	4 hours and 55 minutes ago

When object is detected in ultrasonic detector



The screenshot shows the Wokwi web interface for a project named "wokwi.com/projects/346572482591851092". The device is an "HC-SR04" ultrasonic sensor, currently "Connected". The interface includes tabs for "sketch", "Simulation", "diagram", "library", and "Libra Mana". The "Simulation" tab is active, displaying a circuit diagram of the sensor connected to an "ESP32" microcontroller. The "diagram" tab is also active, showing a circuit diagram with a red LED and a green wire. The "Simulation" tab shows the following log output:

```
1 object is near
1 Sending payload: {"distance":97.82,"object":"Near"}
1 Publish ok
1 Distancein cm97.82
2 object is near
2 Sending payload: {"distance":97.82,"object":"Near"}
2 Publish ok
```