

Smart Farmer - IoT Enabled Smart Farming Application

ASSIGNMENT -4

Student Name	Arun Prakash T S
Roll No	412519106014

To 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.

Code:

```
#include<WiFi.h>// library for WIFI
#include<PubSubClient.h>// library for MQTT
//----- credentials of IBM Accounts -----
#define ORG "04gt4e"// IBM organisation id
#define DEVICE_TYPE "esp32"// Device type mentioned in ibmwatsoniot platform
#define DEVICE_ID "23456"// Device ID mentioned in ibmwatsoniot platform
#define TOKEN "zPS*0TV+fi0h)iq(sT"// Token
#define speed 0.034
#define led 14
String data3;
int LED =4;
//----- customise above values -----
charserver[]= ORG ".messaging.internetofthings.ibmcloud.com";// server name
charpublishTopic[]="iot-2/evt/Data/fmt/json";// topic name and type of event
perform and format in which data to be send
chartopic[]="iot-2/cmd/test/fmt/String";// cmd Represent type and command is
test format of strings
charauthMethod[]="use-token-auth";// authentication method char
chartoken[]= TOKEN;
charclientId[]="d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//Client id
//-----
-----
WiFiClientwifiClient;// creating instance for wificlient
PubSubClientclient(server,1883,wifiClient);// calling the predefined client id
by passing parameter like server id,port and wifi credential
constinttrigpin=5;const
intechopin=18;
String command;
String data="";
long duration;float
dist;
voidsetup()
```

```

{
  Serial.begin(115200);
  pinMode(led,OUTPUT);
  pinMode(trigpin,OUTPUT);
  pinMode(echopin,INPUT);
  wifiConnect();mqttConnect();
}
voidloop(){boolisNearby
=dist<100;
digitalWrite(led,isNearby);
publishData();
delay(500);
if(!client.loop())
{
  mqttConnect();// function call to connect to ibm
}
}
/* -----retrieving to cloud-----
-----*/
voidwifiConnect()
{
  Serial.print("Connecting to ");
  Serial.print("Wifi");
  WiFi.begin("Wokwi-GUEST","",6);
  while(WiFi.status()!= WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }
  Serial.print("WiFi connected, IP address: ");
  Serial.println(WiFi.localIP());
}
voidmqttConnect()
{
  if(!client.connected())
  {
    Serial.print("Reconnecting MQTT client to ");
    Serial.println(server);
    while(!client.connect(clientId,authMethod, token))
    {
      Serial.print(".");
      delay(500);
    }
    initManagedDevice();
    Serial.println();
  }
}
voidinitManagedDevice(){

```

```

if(client.subscribe(topic))
{
Serial.println("IBM subscribe to cmd OK");
}
else
{
Serial.println("subscribe to cmd FAILED");
}
}
void publishData()
{
digitalWrite(trigpin, LOW);
digitalWrite(trigpin, HIGH);
delayMicroseconds(10); digitalWrite(trigpin, LOW);
duration=pulseIn(echopin, HIGH);
dist=duration*speed/2;
if(dist<100)
{
digitalWrite(LED, HIGH); String
payload = "{\"Alert Distance\":\"";
payload +=dist;
payload +=}\"";
Serial.print("\n");
Serial.print("Sending payload:
");Serial.println(payload);if(client.publish(publishTopic,(char*)
payload.c_str()))// if data is uploaded to cloud successfully,prints publish
ok else prints publish failed
{
Serial.println("Publish OK");
}
}
if(dist>100)
{
digitalWrite(LED, HIGH);
String payload = "{\"Distance\":\"";
payload +=dist;
payload +=}\"";
Serial.print("\n");
Serial.print("Sending payload: ");
Serial.println(payload);
if(client.publish(publishTopic,(char*)payload.c_str()))
{
Serial.println("Publish OK");
}
}
else
{
digitalWrite(LED, LOW);
Serial.println("Publish FAILED");
}
}

```

```
}  
}  
}
```

Simulation Output:

<https://wokwi.com/projects/347571602979816019>

The screenshot displays the Wokwi web IDE interface. On the left, the 'sketch.ino' file contains the following code:

```
1 #include <WiFi.h> // library for WIFI  
2 #include <PubSubClient.h> // library for MQTT  
3 //----- credentials of IBM Accounts -----  
4 #define ORG "04gt4e" // IBM organisation id  
5 #define DEVICE_TYPE "esp32" // Device type mentioned in ibm watson iot platform  
6 #define DEVICE_ID "23456" // Device ID mentioned in ibm watson iot platform  
7 #define TOKEN "zPS*0TV+fi0h)iq(st" // Token  
8 #define speed 0.034  
9 #define led 14  
10 String data3;  
11 int LED = 4;  
12 //----- customise above values -----  
13 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // server name  
14 char publishTopic[] = "iot-2/evt/Data/fmt/json"; // topic name and type of event  
15 char topic[] = "iot-2/cmd/test/fmt/String"; // cmd Represent type and command  
16 char authMethod[] = "use-token-auth"; // authentication method char  
17 char token[] = TOKEN;  
18 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //Client id  
19 //-----  
20 WiFiClient wificlient; // creating instance for wificlient  
21 PubSubClient client(server, 1883, wificlient); // calling the predefined client  
22 const int trigpin=5; const  
23 int echopin=18;  
24 String command;  
25 String data="";  
26 long duration; float  
27 dist;  
28 void setup()  
29 {  
30 Serial.begin(115200);
```

The right side of the interface shows a simulation of the hardware. An 'Editing Ultrasonic Distance Sensor' dialog box is open, displaying a distance of 87 cm. Below the simulation, the output console shows the following messages:

```
Sending payload: {"Alert Distance":86.96}  
Publish OK  
  
Sending payload: {"Alert Distance":86.96}  
Publish OK  
  
Sending payload: {"Alert Distance":86.96}
```

IBM Watson IoT Platform

sec19ec035@sairamtap.edu.in
ID: 04gt4e

Browse

Action

Device Types

Interfaces

Add Device

23456

Connected

esp32

Device

6 Nov 2022 15:56

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	["Alert Distance":86.96]	json	a few seconds ago
Data	["Alert Distance":86.96]	json	a few seconds ago
Data	["Alert Distance":86.96]	json	a few seconds ago
Data	["Alert Distance":86.96]	json	a few seconds ago
Data	["Alert Distance":86.96]	json	a few seconds ago

0 Simulations running

16:18

ENG IN

06-11-2022