SOURCE CODE

WOKWI (C++)

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#include "DHT.h"// Library for dht11
#define DHTPIN 15     // what pin we're connected to
#define DHTTYPE DHT22   // define type of sensor DHT 22
#define BUZZER 2
DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and typr of dht connected
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
//-------credentials of IBM Accounts------
#define ORG "6yafic"//IBM ORGANITION ID
#define DEVICE_TYPE "Sensor"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "Sensorid"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "VghKTvPaS!bz+vlCyz"     //Token
String data3;
float h, t;
//-------- 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);
  dht.begin();
  pinMode(BUZZER,OUTPUT);
  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
}
void loop()// Recursive Function
{
  h = dht.readHumidity();
  t = dht.readTemperature();
  Serial.print("Temperature:");
  Serial.println(t);
  Serial.print("Humidity:");
  Serial.println(h);
  PublishData(t, h);
  delay(3000);
  if (!client.loop()) {
    mqttconnect();
  }
}
/.....retrieving to Cloud...../
void PublishData(float temp, float humid) {
  mqttconnect();//function call for connecting to ibm
  /* creating the String in in form JSon to update the data to ibm cloud  */
  String payload = "{\"Temperature\":";
  payload += temp;
  payload += "," "\"Humidity\":";
  payload += humid;
  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(3000);
    }
     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(3000);
    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 == "alarmon")
  {
    pinMode(BUZZER,HIGH);
    delay(1000);
    tone(BUZZER,80);
    delay(10000);
    noTone(BUZZER);
    delay(1000);
    Serial.println(data3);
  }
  else {
  Serial.println(data3);
  pinMode(BUZZER,LOW);
  delay(500);
  }
data3="";
}
```

PYTHON

```
import wiotp.sdk.device
import time
import random
myConfig = {
    "identity": {
        "orgId": "6yafic",
        "typeId": "Sprint1",
        "deviceId":"SprintID"
    },
    "auth": {
        "token": "sW(iQhEK*t)4!jgrjD"
    }
}
def myCommandCallback(cmd):
    print("Message received from IBM IoT Platform: %s" % cmd.data['command'])
    m=cmd.data['command']

client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()
while True:
    temp=random.randint(0,50)
    heart=random.randint(60,100)
    myData={'temperature':temp, 'heartrate':heart}
    client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0,
onPublish=None)
    print("Published data Successfully: %s", myData)
    client.commandCallback = myCommandCallback
    time.sleep(5)
client.disconnect()
```