SPRINT 1

Team ID	PNT2022TMID46404
Project Name	Smart Farmer - IOT Enabled Smart Farming Application

PYTHON CODE:

```
import time import
sys
import ibmiotf.application
import ibmiotf.device import
random
#Provide your IBM Watson Device
Credentials organization = "4tot8b"
deviceType = "smart_farming"
deviceId = "farm_today"
authMethod = "token" authToken
=
"oiJYpRYqYNUC)E2eAt" #
Initialize GPIO def myCommandCallback(cmd):
print("Command received: %s" % cmd.data['command'])
status=cmd.data['command'] if status=="motoron":
    print ("motor is on")
  elif status == "motoroff":
    print ("motor is off")
  else :
    print ("please send proper command")

try:
  deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod,
"auth-token": authToken}
  deviceCli =
ibmiotf.device.Client(deviceOptions)
#................................................. except
Exception as e:
  print("Caught exception connecting device: %s" %
  str(e)) sys.exit()
```

```
# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting"
10 times

deviceCli.connect(

) while True:

    #Get Sensor Data from DHT11



    temp=random.randint(0,100)

    Humid=random.randint(0,100)

    Mois=random.randint(0,100)

    data = {"d":{'temp' : temp, 'Humid': Humid, 'Mois' :Mois}}

    #print data def

    myOnPublishCallback():

      print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % Humid, "Moisture =%s deg
      c"
%Mois, "to IBM Watson") success =

      deviceCli.publishEvent("IoTSensor", "json", data,

qos=0,

on_publish=myOnPublishCallback)  if  not

success:  print("Not  connected  to  IoTF")

time.sleep(1)

   deviceCli.commandCallback =

myCommandCallback # Disconnect the device and application

from the cloud deviceCli.disconnect()
```

ibm iot code.py - C:\Users\PC\Desktop\ibm iot code.py (3.7.0)

File Edit Format Run Options Window Help

```python
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
#Provide your IBM Watson Device Credentials
organization = "4tot8b"
deviceType = "smart_farming"
deviceId = "farm_today"
authMethod = "token"
authToken = "oiJYpRYqYNUC)E2eAt"
# Initialize GPIO
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="motoron":
        print ("motor is on")
    elif status == "motoroff":
        print ("motor is off")
    else :
        print ("please send proper command")

try:
    deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}
    deviceCli = ibmiotf.device.Client(deviceOptions)
#..............................................
except Exception as e:
    print("Caught exception connecting device: %s" % str(e))
    sys.exit()
# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times
deviceCli.connect()
while True:
     #Get Sensor Data from DHT11

     temp=random.randint(0,100)
     Humid=random.randint(0,100)
     Mois=random.randint(0,100)
     data = {"d":{'temp' : temp, 'Humid': Humid, 'Mois' :Mois}}
     #print data
     def myOnPublishCallback():
        print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % Humid, "Moisture =%s deg c" %Mois, "to IBM Watson")
```

ibm iot code.py - C:\Users\PC\Desktop\ibm iot code.py (3.7.0)

File Edit Format Run Options Window Help

```python
deviceId = "farm_today"
authMethod = "token"
authToken = "oiJYpRYqYNUC)E2eAt"
# Initialize GPIO
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="motoron":
        print ("motor is on")
    elif status == "motoroff":
        print ("motor is off")
    else :
        print ("please send proper command")

try:
    deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}
    deviceCli = ibmiotf.device.Client(deviceOptions)
#..............................................
except Exception as e:
    print("Caught exception connecting device: %s" % str(e))
    sys.exit()
# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times
deviceCli.connect()
while True:
     #Get Sensor Data from DHT11

     temp=random.randint(0,100)
     Humid=random.randint(0,100)
     Mois=random.randint(0,100)
     data = {"d":{'temp' : temp, 'Humid': Humid, 'Mois' :Mois}}
     #print data
     def myOnPublishCallback():
        print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % Humid, "Moisture =%s deg c" %Mois, "to IBM Watson")
        success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCallback)
if not success:
    print("Not connected to IoTF")
    time.sleep(1)
    deviceCli.commandCallback = myCommandCallback
# Disconnect the device and application from the cloud
deviceCli.disconnect()
```

Python 3.7.0 Shell

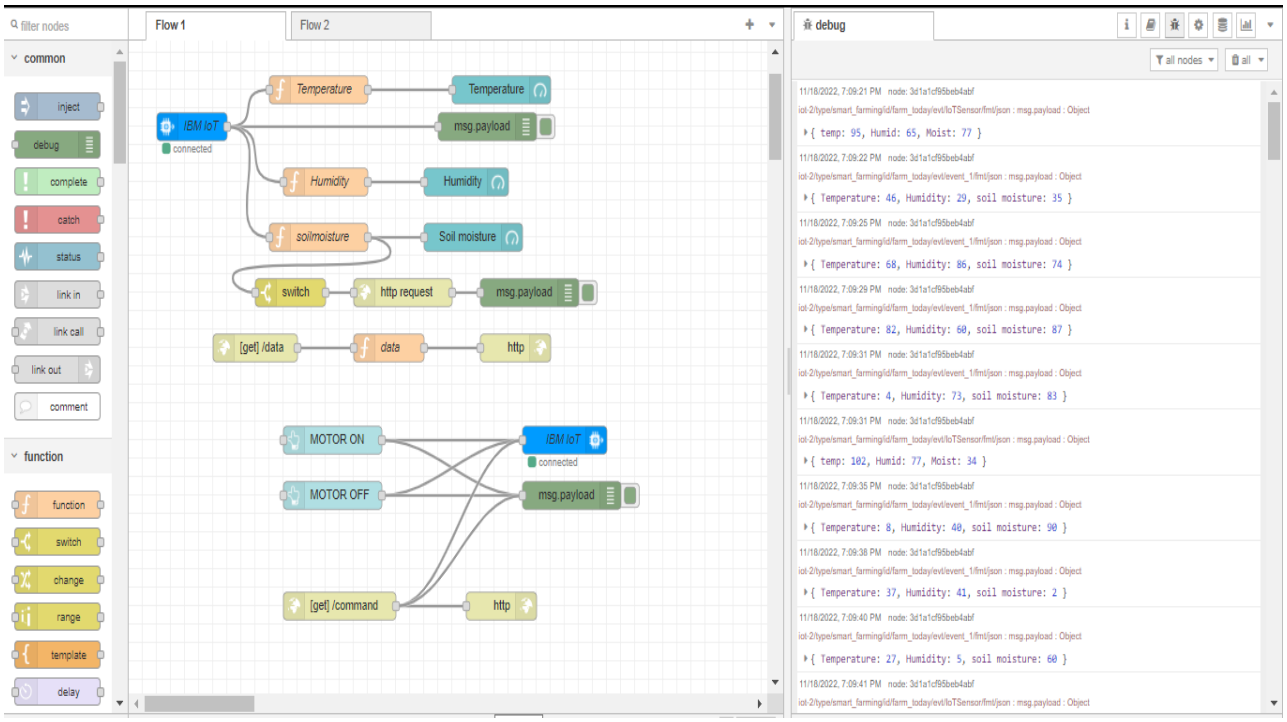
File Edit Shell Debug Options Window Help

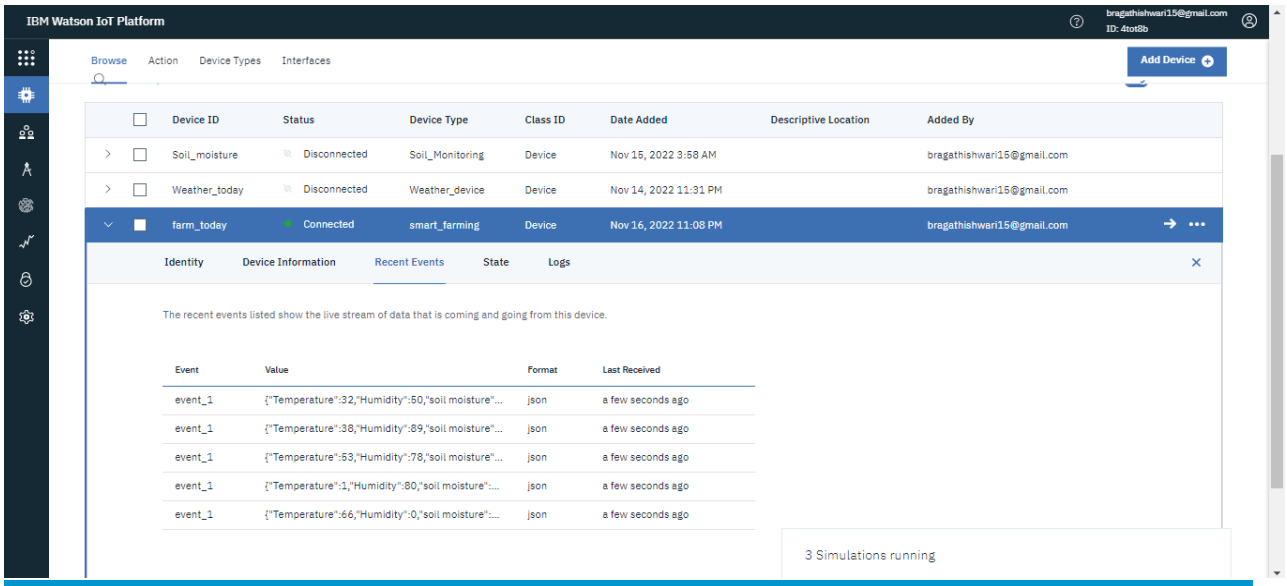
```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
================ RESTART: C:\Users\PC\Desktop\ibm iot code.py ================
2022-11-17 12:42:39,934   ibmiotf.device.Client      INFO    Connected successfully: d:4tot8b:smart_farming:farm_today
```

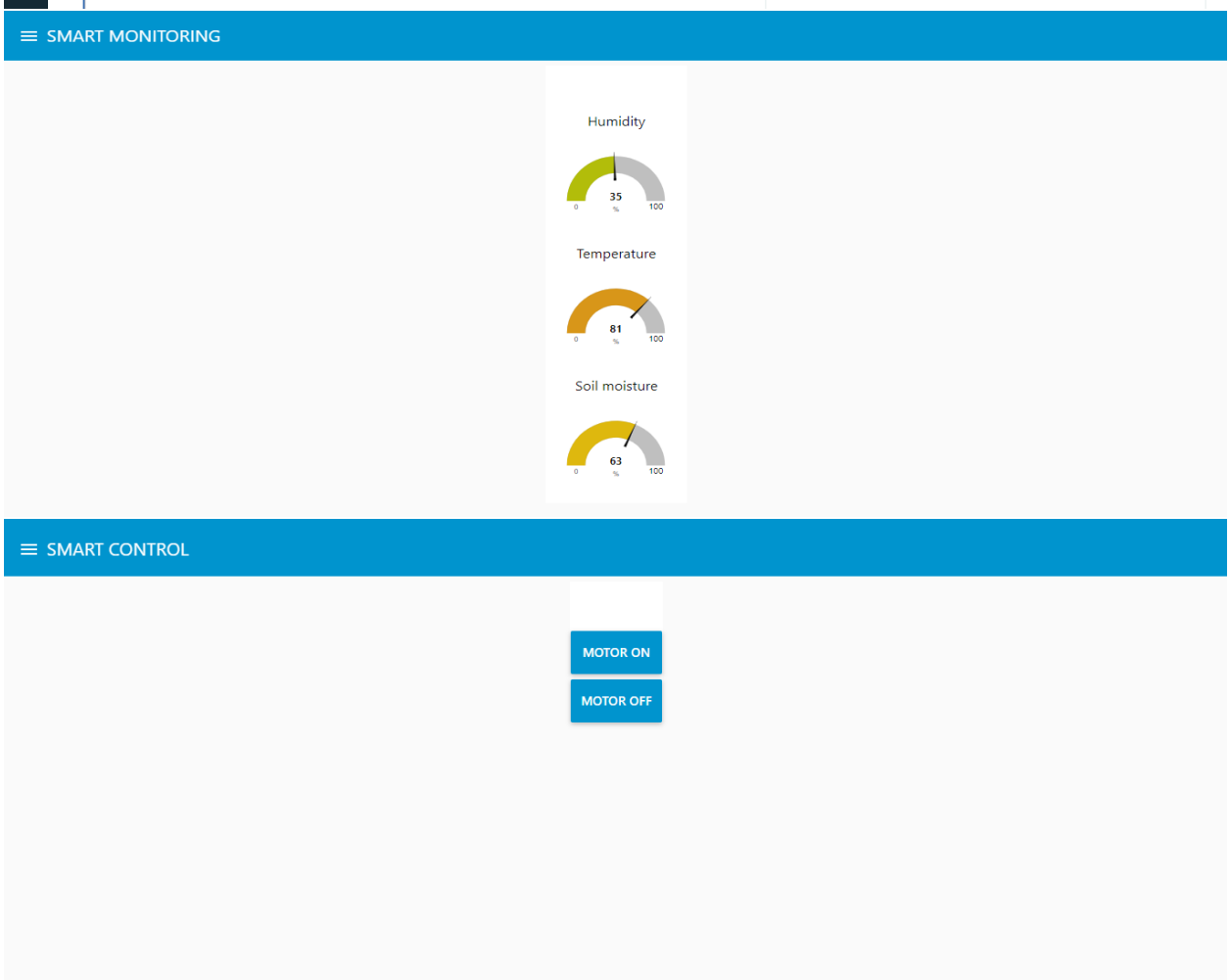
Python 3.7.0 Shell

File Edit Shell Debug Options Window Help

```
Published Temperature = 107 C Humidity = 69 % Soilmoisture= 19 deg c to IBM Watson
Published Temperature = 95 C Humidity = 79 % Soilmoisture= 61 deg c to IBM Watson
Published Temperature = 108 C Humidity = 100 % Soilmoisture= 83 deg c to IBM Watson
Published Temperature = 106 C Humidity = 72 % Soilmoisture= 39 deg c to IBM Watson
Published Temperature = 107 C Humidity = 60 % Soilmoisture= 25 deg c to IBM Watson
Published Temperature = 107 C Humidity = 86 % Soilmoisture= 32 deg c to IBM Watson
Published Temperature = 94 C Humidity = 62 % Soilmoisture= 104 deg c to IBM Watson
Published Temperature = 102 C Humidity = 71 % Soilmoisture= 58 deg c to IBM Watson
Published Temperature = 101 C Humidity = 63 % Soilmoisture= 58 deg c to IBM Watson
Published Temperature = 102 C Humidity = 76 % Soilmoisture= 38 deg c to IBM Watson
Published Temperature = 95 C Humidity = 79 % Soilmoisture= 42 deg c to IBM Watson
Published Temperature = 97 C Humidity = 79 % Soilmoisture= 66 deg c to IBM Watson
Published Temperature = 106 C Humidity = 61 % Soilmoisture= 89 deg c to IBM Watson
Published Temperature = 97 C Humidity = 70 % Soilmoisture= 98 deg c to IBM Watson
Published Temperature = 92 C Humidity = 100 % Soilmoisture= 34 deg c to IBM Watson
Published Temperature = 99 C Humidity = 93 % Soilmoisture= 18 deg c to IBM Watson
Published Temperature = 110 C Humidity = 85 % Soilmoisture= 55 deg c to IBM Watson
Published Temperature = 96 C Humidity = 61 % Soilmoisture= 76 deg c to IBM Watson
Published Temperature = 109 C Humidity = 83 % Soilmoisture= 102 deg c to IBM Watson
Published Temperature = 103 C Humidity = 99 % Soilmoisture= 94 deg c to IBM Watson
Published Temperature = 104 C Humidity = 65 % Soilmoisture= 83 deg c to IBM Watson
Published Temperature = 107 C Humidity = 61 % Soilmoisture= 22 deg c to IBM Watson
Published Temperature = 109 C Humidity = 86 % Soilmoisture= 59 deg c to IBM Watson
Published Temperature = 95 C Humidity = 89 % Soilmoisture= 100 deg c to IBM Watson
Published Temperature = 103 C Humidity = 93 % Soilmoisture= 92 deg c to IBM Watson
Published Temperature = 95 C Humidity = 81 % Soilmoisture= 104 deg c to IBM Watson
Published Temperature = 92 C Humidity = 78 % Soilmoisture= 10 deg c to IBM Watson
Command received: motor on
please send proper command
Command received: motor off
please send proper command
Published Temperature = 94 C Humidity = 86 % Soilmoisture= 36 deg c to IBM Watson
Published Temperature = 95 C Humidity = 93 % Soilmoisture= 81 deg c to IBM Watson
Published Temperature = 107 C Humidity = 75 % Soilmoisture= 10 deg c to IBM Watson
Published Temperature = 108 C Humidity = 65 % Soilmoisture= 19 deg c to IBM Watson
Published Temperature = 101 C Humidity = 96 % Soilmoisture= 45 deg c to IBM Watson
Published Temperature = 105 C Humidity = 82 % Soilmoisture= 15 deg c to IBM Watson
Command received: motor on
please send proper command
Published Temperature = 94 C Humidity = 86 % Soilmoisture= 105 deg c to IBM Watson
Published Temperature = 101 C Humidity = 83 % Soilmoisture= 19 deg c to IBM Watson
```







Connecting Sensors with ESP32 RASP using C++ code

```
#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 11
#define LED 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 "i3869j"//IBM ORGANITION ID
#define DEVICE_TYPE "abcd"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "1234"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //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(LED,OUTPUT);
delay(10);   Serial.println();
wificonnect();
mqttconnect();
}  void loop()// Recursive
Function
{    h = dht.readHumidity();
t = dht.readTemperature();
  Serial.print("temp:");
  Serial.println(t);
  Serial.print("Humid:");
  Serial.println(h);
  PublishData(t, h);
delay(1000);   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 = "{\"temp\":";
payload += temp;   payload +=
"," "\"Humid\":";   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(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=="lighton")   {
Serial.println(data3);
digitalWrite(LED,HIGH);
  }
else
  {
Serial.println(data3); digitalWrite(LED,LOW);
  } data3="";
}
```

WOKWi SAVE SHARE sketch.ino

sketch.ino diagram.json libraries.txt Library Manager

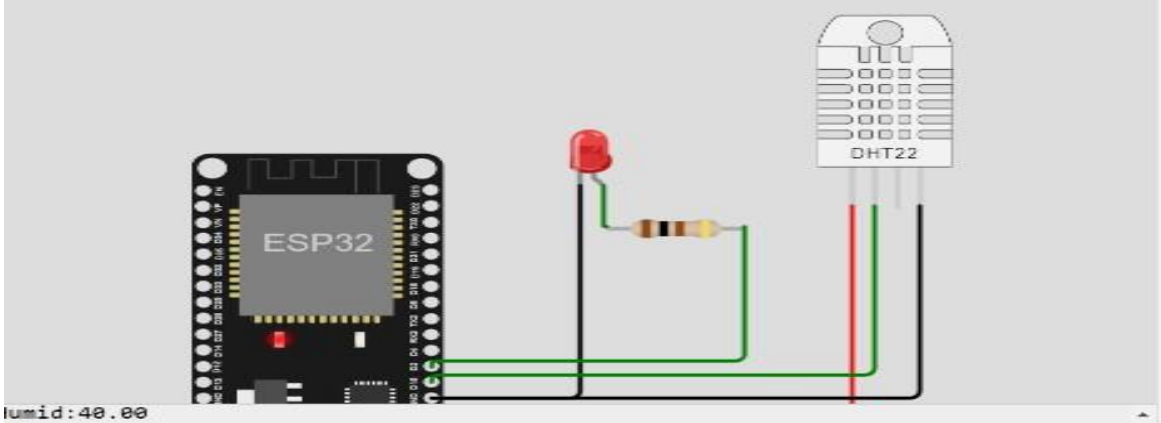
```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#include "DHT.h"// Library for dht11
#define DHTPIN 15     // what pin we're connected to
#define DHTTYPE DHT22   // define type of sensor DHT 11
#define LED 2

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 "4tot8b"//IBM ORGANITION ID
#define DEVICE_TYPE "smart_farming"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "farm_today"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "oiJYpRYqYNUC)E2eAt"     //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
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd  REPRESENT command type AND COMMAND IS TEST OF FORMAT ST
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


```

28°C Partly sunny Search

Docs

Simulation

00:27.090 99%




```
umid:40.00
ending payload: {"temp":24.00,"Humid":40.00}
ublish ok
emp:24.00
umid:40.00
ending payload: {"temp":24.00,"Humid":40.00}
ublish ok
```

ENG IN 10:42 18-11-2022