```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#include "DHT.h"// Library for dht11
#define DHT_SENSOR_PIN  21 // ESP32 pin GIOP21 connected to DHT22 sensor
#define DHT_SENSOR_TYPE DHT22
#define RELAY_PIN    17 // ESP32 pin GIOP17 that connects to relay
#define AOUT_PIN 36
#define MOISTURE_PIN 36 // ESP32 pin GIOP36 (ADC0) that connects to AOUT pin
of moisture sensor
#define THRESHOLD 1000
#define POWER_PIN  17 // ESP32 pin GIOP17 connected to sensor's VCC pin
#define SIGNAL_PIN 36 // ESP32 pin GIOP36 (ADC0) connected to sensor's signal
pin
#define SENSOR_MIN 0
#define SENSOR_MAX 521
#define LED 23

DHT dht_sensor(DHT_SENSOR_PIN, DHT_SENSOR_TYPE);// 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 "31b6or"//IBM ORGANITION ID
#define DEVICE_TYPE "smart_farm"//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "Agriculture"//Device ID mentioned in ibm watson IOT
Platform
#define TOKEN "-4+ZnfIxFunHoCkyIu"     //Token
#define METHOD "use-token-auth"
String data4;
float h, t, m, w;


//-------- 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); //calling the
predefined client id by passing parameter like server id,portand
wificredential
int value = 0; // variable to store the sensor value
int level = 0; // variable to store the water level

void setup()// configureing the ESP32
{
  Serial.begin(115200);
  dht_sensor.begin();// initialize the DHT sensor
  pinMode(RELAY_PIN, OUTPUT);
  pinMode(POWER_PIN, OUTPUT);   // configure D7 pin as an OUTPUT
  digitalWrite(POWER_PIN, LOW); // turn the sensor OFF
  pinMode(LED,OUTPUT);
  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
}

void loop()
{
  // read humidity
  float hum  = dht_sensor.readHumidity();
  // read temperature in Celsius
  float tempC = dht_sensor.readTemperature();

  // check whether the reading is successful or not
  if ( isnan(tempC) || isnan(hum))
  {
    Serial.println("Failed to read from DHT sensor!");
  }
  else
   {
    Serial.print("Humidity: ");
    Serial.print(hum);
    Serial.print("%");

    Serial.print("  |  ");

    Serial.print("Temperature: ");
    Serial.print(tempC);
    Serial.print("°C  ~  ");

   }
  m = analogRead(AOUT_PIN); // read the analog value from sensor
  m = analogRead(MOISTURE_PIN); // read the analog value from sensor
  if (m < THRESHOLD)
```

```
  {
    Serial.print("The soil is DRY (");
    Serial.print("The soil is DRY => turn pump ON");
    digitalWrite(RELAY_PIN, HIGH);
  }
  else
  {
    Serial.print("The soil is WET (");
    Serial.print("The soil is WET => turn pump OFF");
    digitalWrite(RELAY_PIN, LOW);
  }
  Serial.print(" (");
  Serial.print(m);
  Serial.println(")");
  delay(500);

  digitalWrite(POWER_PIN, HIGH);  // turn the sensor ON
  delay(10);                      // wait 10 milliseconds
  value = analogRead(SIGNAL_PIN); // read the analog value from sensor
  digitalWrite(POWER_PIN, LOW);   // turn the sensor OFF

  level = map(value, SENSOR_MIN, SENSOR_MAX, 0, 4); // 4 levels
  Serial.print("Water level: ");
  Serial.println(level);
  delay(1000);

  PublishData(t, h, m, w);
  delay(1000);
  if (!client.loop())
  {
    mqttconnect();
  }
}

/*.....................................retrieving to
Cloud.............................*/

void PublishData(float temp, float hum, float moist, float water)
{
  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 += hum;
  payload = "{\"moisture\":";
```

```
  payload += moist;
  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]);
    data4 += (char)payload[i];
  }

  Serial.println("data: "+ data4);
  if(data4=="lighton")
  {
Serial.println(data4);
digitalWrite(LED,HIGH);

  }

  else
  {
Serial.println(data4);
digitalWrite(LED,LOW);

  }
data4="";
}
```