```
#include <ESP8266WiFi.h>
#include "DHT.h"
#include <ArduinoJson.h>
#include <PubSubClient.h>

// Watson IoT connection details
#define MQTT_HOST
"xpb9eu.messaging.internetofthings.ibmcloud.com" //Organization
ID.messaging.internetofthings.ibmcloud.com
//change  3xr4l4
#define MQTT_PORT 1883
#define MQTT_DEVICEID "d:xpb9eu:ESP8266:dev1"
//d:Organization ID:Device Type:Device ID
//change  3xr4l4
#define MQTT_USER "use-token-auth"
#define MQTT_TOKEN "karthikproject" // change your auth_id :
#define MQTT_TOPIC "iot-2/evt/status/fmt/json"
#define MQTT_TOPIC_DISPLAY "iot-2/cmd/display/fmt/json"

// Add GPIO pins used to connect devices

#define DHT_PIN 2 // GPIO pin the data line of the DHT sensor is
connected to

// Specify DHT11 (Blue) or DHT22 (White) sensor
#define DHTTYPE DHT11


// Add WiFi connection information
char ssid[] = "karthick";     //  your network SSID (name)
char pass[] = "87654321";  // your network password


DHT dht(DHT_PIN, DHTTYPE);


// MQTT objects
void callback(char* topic, byte* payload, unsigned int length);
```

```
WiFiClient wifiClient;
PubSubClient mqtt(MQTT_HOST, MQTT_PORT, callback, wifiClient);

// variables to hold data
StaticJsonDocument<100> jsonDoc;
JsonObject payload = jsonDoc.to<JsonObject>();
JsonObject status = payload.createNestedObject("d");
static char msg[50];

float h = 0.0;
float t = 0.0;



void callback(char* topic, byte* payload, unsigned int length) {
  // handle message arrived
  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] : ");

  payload[length] = 0; // ensure valid content is zero terminated so
can treat as c-string
  Serial.println((char *)payload);
}
void sendSMS(String msg)
{
  Serial.print("AT");  //Start Configuring GSM Module
  delay(1000);        //One second delay
  Serial.println();
  Serial.println("AT+CMGF=1");
  delay(1000);
  Serial.println("AT+CMGS=\"+916385808140\"\r");
  delay(1000);
  Serial.println(msg);
  delay(100);
  Serial.println((char)26);
  delay(1000);
}
```

```
void setup() {
 // Start serial console
  Serial.begin(115200);
  Serial.setTimeout(2000);
  while (!Serial) { }
  Serial.println();
  Serial.println("ESP8266 IBM Cloud Application");

  // Start WiFi connection
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, pass);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi Connected");

  // Start connected devices
  dht.begin();

  // Connect to MQTT - IBM Watson IoT Platform
  if (mqtt.connect(MQTT_DEVICEID, MQTT_USER, MQTT_TOKEN)) {
    Serial.println("MQTT Connected");
    mqtt.subscribe(MQTT_TOPIC_DISPLAY);

  } else {
    Serial.println("MQTT Failed to connect!");
    ESP.reset();
  }
}

void loop() {
  mqtt.loop();
  while (!mqtt.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (mqtt.connect(MQTT_DEVICEID, MQTT_USER, MQTT_TOKEN)) {
```

```
    Serial.println("MQTT Connected");
    mqtt.subscribe(MQTT_TOPIC_DISPLAY);
    mqtt.loop();
   } else {
    Serial.println("MQTT Failed to connect!");
    delay(5000);
   }
 }
 int sensorValue = analogRead(A0);
 Serial.println(sensorValue);
 delay(1000);
 h = dht.readHumidity();
 t = dht.readTemperature(); // uncomment this line for centigrade
 Serial.print("Current humidity = ");
  Serial.print(h);
  Serial.print("%  ");
  Serial.print("temperature = ");
  Serial.print(t);
  Serial.println("C  ");
  // t = dht.readTemperature(true); // uncomment this line for
Fahrenheit

 // Check if any reads failed and exit early (to try again).
 if (sensorValue<50) {
  Serial.println("WATER POLLUTED");
  sendSMS("WATER POLLUTED");
    delay(3000);
 }
 if (isnan(h) || isnan(t)) {
  Serial.println("Failed to read from DHT sensor!");
 } else {
  // Send data to Watson IoT Platform
  status["temp"] = t;
  status["humidity"] = h;
  status["TURBIDITY"] = sensorValue;
  serializeJson(jsonDoc, msg, 50);
  Serial.println(msg);
  if (!mqtt.publish(MQTT_TOPIC, msg)) {
```

```
      Serial.println("MQTT Publish failed");
    }
  }

  // Pause - but keep polling MQTT for incoming messages
  for (int i = 0; i < 10; i++) {
    mqtt.loop();
    delay(1000);
  }
}
```