Final Code

Team ID	PNT2022TMID05121
Project Name	Iot Based Smart Crop Protection System for Agriculture

Source Code :

IoT Code

```
#include <AsyncTCP.h>
#include <ESPAsyncWebServer.h>
#include <WiFi.h>
#include <WiFiClient.h>
#include <PubSubClient.h>
#include <Adafruit_BMP280.h>
#include <math.h>
#include <Wire.h>
#define BMP_SDA 21
#define BMP_SCL 22
#include <DFRobot_DHT11.h>
DFRobot_DHT11 DHT;
#define DHT11_PIN 4
#define rainAnalog 35
#define rainDigital 34
#define moistureDigital 32

Adafruit_BMP280 bmp280;
const char* ssid     = "";
const char* password = "";
AsyncWebServer server(80);
AsyncEventSource events("/events");
unsigned long lastTime = 0;
unsigned long timerDelay = 1000;
int soil;
int rain;
int rainA;
float temperature;
float humidity;
float pressure;
float  altitude;
long lastMsg = 0;
int pumpRelayPin = 26;

#define ORG "6jw3v9"
#define DEVICE_TYPE "ESP32"
#define DEVICE_ID "############################"
#define TOKEN "############################"

char servers[] = ORG ".messaging.internetofthings.ibmcloud.com";
char pubTopic1[] = "iot-2/evt/temperature/fmt/json";
char pubTopic2[] = "iot-2/evt/humidity/fmt/json";
char pubTopic3[] = "iot-2/evt/pressure/fmt/json";
char pubTopic4[] = "iot-2/evt/altitude/fmt/json";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;

WiFiClient wifiClient;
PubSubClient client(servers, 1883, NULL, wifiClient);
```

```
// Init BME280
void initBME() {
  if (!bmp280.begin(0x76)) {
    Serial.println("Could not find a valid BMP280 sensor, check wiring!");
    while (1);
  }
}

void getSensorReadings() {
  DHT.read(DHT11_PIN);
  temperature = DHT.temperature;
  humidity = DHT.humidity;
  pressure = bmp280.readPressure() / 100;
  soil = digitalRead(moistureDigital);
  rain = digitalRead(rainDigital);
  rainA = analogRead(rainAnalog);
  altitude = bmp280.readAltitude(1011.18);
  if(soil == 1){
    digitalWrite(pumpRelayPin, LOW);
  }
  else{
    digitalWrite(pumpRelayPin, HIGH);
  }
}

// Initialize WiFi
void initWiFi() {
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  Serial.print("Connecting to WiFi ..");
  while (WiFi.status() != WL_CONNECTED) {
  Serial.print('.');
    delay(1000);
  }
  Serial.println(WiFi.localIP());
}

String processor(const String& var) {
  getSensorReadings();
  //Serial.println(var);
  if (var == "TEMPERATURE") {
    return String(temperature);
  }
  else if (var == "HUMIDITY") {
    return String(humidity);
  }
  else if (var == "PRESSURE") {
    return String(pressure);
  }
  else if (var == "ALTITUDE") {
    return String(altitude);
  }
  else if (var == "RAINING") {
    return String(rain);
  }
  else if (var == "SOIL") {
    return String(soil);
  }
  return String();
}

const char index_html[] PROGMEM = R"rawliteral(
<!DOCTYPE HTML><html>
<head>
  <title>Grow Greens Smart</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.2/css/all.css"
integrity="sha384-fnmOCqbTlWIlj8LyTjo7mOUStjsKC4pOpQbqyi7RrhN7udi9RwhKkMHpvLbHG9
Sr" crossorigin="anonymous">
  <link rel="icon" href="data:,">
  <style>
    html {font-family: Arial; display: inline-block; text-align: center; background-color:#FCF8E8}
    p { font-size: 1.2rem;}
    body {  margin: 0;}
    .topnav { overflow: hidden; color: #6D9886; font-size: 1rem; }
    .content { padding: 20px; }
    .card { background-color: #F2AA4CFF; box-shadow: 2px 2px 12px 1px rgba(140,140,140,.5);
border-radius: 30px;}
    .cards { max-width: 800px; margin: 0 auto; display: grid; grid-gap: 2rem;
grid-template-columns: repeat(auto-fit, minmax(200px, 1fr)); }
    .reading { font-size: 1.4rem; }
  </style>
</head>
<body>
  <div class="topnav">
    <h1>Grow Greens Smart</h1>
  </div>

  <div class="content">

    <div class="cards">

      <div class="card">

        <p><i class="fas fa-thermometer-half" style="color:#101820FF; font-size:25px"></i>
Temperature</p><p><span class="reading" style = "color:#101820FF"><span id="temp"
style="font-size:1rem; font-weight:bolder;">%TEMPERATURE%</span> &deg;C</span></p>

      </div>

      <div class="card">

        <p><i class="fas fa-tint" style="color:#101820FF; font-size:25px"></i>
Humidity</p><p><span class="reading" style="color:#101820FF; font-size:1rem;"><span
id="hum" style="font-size:1rem; font-weight:bolder;">%HUMIDITY%</span>
&percnt;</span></p>

      </div>

      <div class="card">

        <p><i class="fas fa-angle-double-down" style="color:#101820FF; font-size:25px"></i>
Pressure</p><p><span class="reading" style="color:#101820FF; font-size:1rem;"><span
id="pres" style="font-size:1rem; font-weight:bolder;">%PRESSURE%</span> hPa</span></p>

      </div>

      <div class="card">

        <p><i class="fas fa-mountain" style="color:#101820FF; font-size:25px"></i>
Altitude</p><p><span class="reading" style="color:#101820FF"><span id="alti"
style="font-size:1rem; font-weight:bolder;">%ALTITUDE%</span> m</span></p>

      </div>

      <div class="card">

        <p><i class="fas fa-cloud-rain" style="color:#101820FF; font-size:25px"></i>
Raining</p><p><span class="reading" style="color:#101820FF"><span id="rain"
style="font-size:1rem; font-weight:bolder;">%RAINING%</span></span></p>

      </div>

      <div class="card">

        <p><i class="fas fa-tree" style="color:#101820FF; font-size:25px"></i>
Moisture</p><p><span class="reading" style="color:#101820FF"><span id="soil"
style="font-size:1rem; font-weight:bolder;">%SOIL%</span></span></p>
```

```
      </div>
    </div>
  </div>
<script>
if (!!window.EventSource) {
 var source = new EventSource('/events');

 source.addEventListener('open', function(e) {
  console.log("Events Connected");
 }, false);
 source.addEventListener('error', function(e) {
  if (e.target.readyState != EventSource.OPEN) {
  console.log("Events Disconnected");
  }
 }, false);

 source.addEventListener('message', function(e) {
  console.log("message", e.data);
 }, false);

 source.addEventListener('temperature', function(e) {
  console.log("temperature", e.data);
  document.getElementById("temp").innerHTML = e.data;
 }, false);

 source.addEventListener('humidity', function(e) {
  console.log("humidity", e.data);
  document.getElementById("hum").innerHTML = e.data;
 }, false);

 source.addEventListener('pressure', function(e) {
  console.log("pressure", e.data);
  document.getElementById("pres").innerHTML = e.data;
 }, false);

 source.addEventListener('altitude', function(e) {
  console.log("latitude", e.data);
  document.getElementById("alti").innerHTML = e.data;
 }, false);

 source.addEventListener('rain', function(e) {
 console.log("Rain", e.data);
  if(e.data == '0')
  document.getElementById("rain").innerHTML = "Raining";
  else
    document.getElementById("rain").innerHTML = "Not Raining";
 }, false);

 source.addEventListener('soil', function(e) {
 console.log("Soil Moisture", e.data);
 if(e.data == '1')
    document.getElementById("soil").innerHTML = "Less Water";
  else
    document.getElementById("soil").innerHTML = "Enough Water";
 }, false);
}
```

```
</script>
</body>
</html>)rawliteral";

void setup()  {
  Serial.begin(115200);
  pinMode(rainDigital, INPUT);
  pinMode(moistureDigital, INPUT);
  pinMode(pumpRelayPin, OUTPUT);
  initWiFi();
  initBME();
  // Handle Web Server
  server.on("/", HTTP_GET, [](AsyncWebServerRequest * request) {
    request->send_P(200, "text/html", index_html, processor);
  });

  // Handle Web Server Events
  events.onConnect([](AsyncEventSourceClient * client) {
  if (client->lastId()) {
      Serial.printf("Client reconnected! Last message ID that it got is: %u\n", client->lastId());
    }
    // send event with message "hello!", id current millis
    // and set reconnect delay to 1 second
    client->send("hello!", NULL, millis(), 10000);
  });
  server.addHandler(&events);
  server.begin();
  if (!client.connected()) {
    Serial.print("Reconnecting client to ");
    Serial.println(servers);
    while (!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
      delay(500);
    }
    Serial.println("Bluemix connected");
  }
}
void loop() {
  client.loop();
  long now = millis();
  if (now - lastMsg > 3000) {
    lastMsg = now;
    String payload = "{\"temperature\":";
    payload += temperature;
    payload += "}";
    Serial.print("Sending payload: ");
    Serial.println(payload);
    if (client.publish(pubTopic1, (char*) payload.c_str())) {
      Serial.println("Publish ok");
    } else {
      Serial.println("Publish failed");
    }
```

```
String payload1 = "{\"humidity\":";

payload1 += humidity;

payload1 += "}";

Serial.print("Sending payload: ");

Serial.println(payload1);

if (client.publish(pubTopic2, (char*) payload1.c_str())) {

  Serial.println("Publish ok");

} else {

  Serial.println("Publish failed");

}

String payload2 = "{\"pressure\":";

payload2 += pressure;

payload2 += "}";

Serial.print("Sending payload: ");

Serial.println(payload2);

if (client.publish(pubTopic3, (char*) payload2.c_str())) {

  Serial.println("Publish ok");

} else {

  Serial.println("Publish failed");

}
String payload3 = "{\"altitude\":";

payload3 += altitude;

payload3 += "}";

Serial.print("Sending payload: ");
```

```
  Serial.println(payload3);
  if (client.publish(pubTopic4, (char*) payload3.c_str())) {
    Serial.println("Publish ok");
  } else {
    Serial.println("Publish failed");
  }
}
if ((millis() - lastTime) > timerDelay) {
  getSensorReadings();
  Serial.printf("Temperature = %.2f ºC \n", temperature);
  Serial.printf("Humidity = %.2f \n", humidity);
  Serial.printf("Pressure = %.0f hPa \n", pressure);
  Serial.printf("Altitude = %.0f m \n", altitude);
  Serial.printf("Rain = %d\n", rain);
  Serial.printf("Rain = %d\n", rainA);
  Serial.printf("Soil = %d\n", soil);
  Serial.println();
  // Send Events to the Web Server with the Sensor Readings
  events.send("ping", NULL, millis());
  events.send(String(temperature).c_str(), "temperature", millis());
  events.send(String(humidity).c_str(), "humidity", millis());
  events.send(String(pressure).c_str(), "pressure", millis());
  events.send(String(altitude).c_str(), "altitude", millis());
  events.send(String(rain).c_str(), "rain", millis());
  events.send(String(soil).c_str(), "soil", millis());
```

```
lastTime = millis();
   }
 }
```