

Final Project

14.05.2025.

Anika Petrovic

Erasmus+ Student at FEI, STU

Advanced Information Technologies

SS 2024/2025

Overview

The aim of this project is to visualize the outputs from measuring air humidity using NodeMCU and ThingsBoard – a platform for IoT device management and IoT data collection, processing, and visualization.

Goals

1. Read air humidity using an ESP8266 NodeMCU and a DHT11 sensor.
2. Implement a simple application with ThingsBoard Community Edition Demo Live server to perform telemetry injection and visualization of the provided sensor and NodeMCU microcontroller.

Specifications of equipment and tools

NodeMCU ESP8266

DHT11 temperature and humidity sensor

Jumper wires

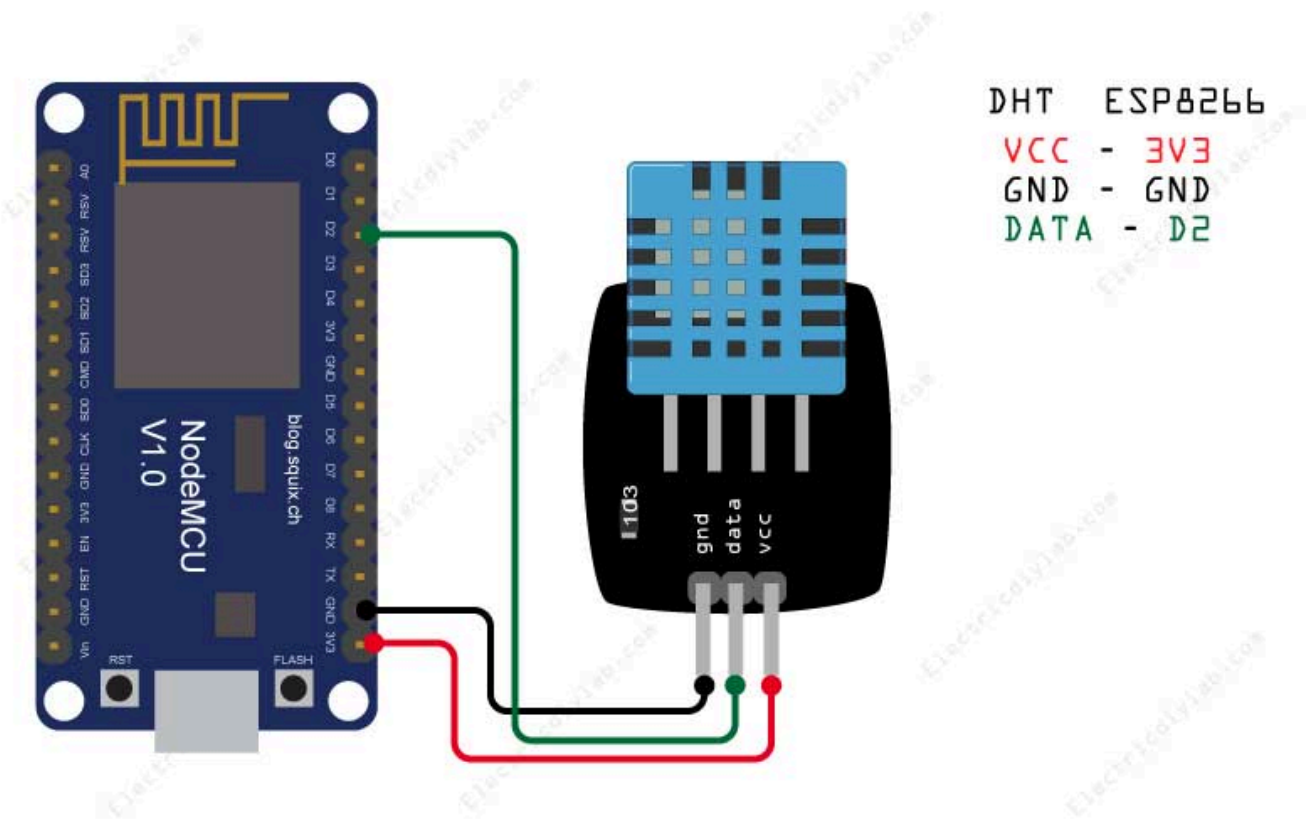
USB cable for uploading code

Thingsboard - IoT platform

Arduino IDE - development environment

Connecting NodeCMU and sensor

To connect NodeCMU ESP8266 and sensor DHT11 we need 3 jumper wires. Connect as shown in picture:



Use a USB cable to connect it to your laptop so we can load code onto NodeMCU.

First, we need to set up an Arduino IDE. Open Arduino IDE, go to Files and click on the Preferences. Copy following link to the Additional Boards Manager and click OK:

http://arduino.esp8266.com/stable/package_esp8266com_index.json

Now, go to Tools and board and select Board Manager. Navigate to esp8266 by esp8266 community and install the software for Arduino.

Once all the above processes have been completed we are ready to program our esp8266 with Arduino IDE.

If the DHT library is not installed, install it via /Project/Include Libraries/Manage Libraries (choose Adafruit DHT Sensor Library).

Now we can create a new sketch and load it into NodeMCU by clicking Verify and Upload.

Code

Let's review what our code should look like:

- Read humidity values from sensor
- Publish results to ThingsBoard platform (PubSubClient - MQTT client library)

Once physical parts are connected, reading values from the sensor is easy.

Keep in mind ESP8266 pin mapping, in code we use DPIN 4 which is D2 pin on ESP8266 board. To create a DHT object we also need to specify DTYPE in this case it is DHT11. Now we can just use the readHumidity() function.

To publish results of measurement to the ThingsBoard platform we will use PubSubClient Arduino Library. It uses the MQTT protocol, and MQTT protocol runs over WiFi protocol on a physical level. So we need a WiFi Arduino Library.

For now, our code looks like this:

```
#include <PubSubClient.h>
#include <ESP8266WiFi.h>
#include <stdlib.h>

#include <DHT.h>

#include <credentials.h>

#define DPIN 4
#define DTYPE DHT11

DHT dht(DPIN,DTYPE);

WiFiClient client;
PubSubClient pubsub_client(client);

void setup() {
  Serial.begin(9600);
  dht.begin();

  delay(10);

  Serial.println("Connecting to ");
```

```

Serial.println(ssid);

WiFi.begin(ssid, pass);
while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}
Serial.println("");
Serial.println("WiFi connected");
}

void loop() {
    delay(2000);

    float humidity = dht.readHumidity();
    Serial.println(humidity);
}

```

For security reasons, we created **credentials.h** file and added it to our sketch.

```

#define ssid myssidwhatever
#define pass mypassword

```

ThingsBoard Platform

For the purpose of this project, we are using the ThingsBoard Demo Live server. Before we can connect to this server using the MQTT protocol, we need to set up a few things.

- Create new device, get access token and device id
- Use credentials.h file to store those
- Create new dashboard and show data

Create new device and copy device id and access token to credentials.h file.

Add new device?×

1 Device details

2 Credentials
Optional

Name*

humidity-sensor

Label

Device profile*

default

×

☐ Is gateway

Assign to customer

Next: Credentials

Cancel

Add

humidity-sensor?×

Device details

<

Details

Attributes

Latest telemetry

Calculated fields

Alarms

Events

>

Open details page

Make device public

Assign to customer

Manage credentials

Check connectivity

Delete device

Copy device Id

Copy access token

Name*

humidity-sensor

Device profile*

default

Label

Assigned firmware

To create new dashboard and add select telemetry data to visualize follow next steps.

Add dashboard

Title*

Humidity sensor

Description

Assigned customers

Mobile application settings

☐ Hide dashboard in mobile application

Dashboard order in mobile application

Cancel

Add

humidity-sensor

Device details

Details

Attributes

Latest telemetry

Calculated fields

Alarms

Events

1 telemetry unit selected

Show on widget

Show on widget

<input checked="" type="checkbox"/>	Last update time	Key ↑	Value	
<input checked="" type="checkbox"/>	2025-05-14 13:43:51	humidity	31	

Add widget to dashboard

Select existing dashboard

Create new dashboard

☒ Dashboard*
Humidity sensor

☐ New dashboard title

☐ Open dashboard

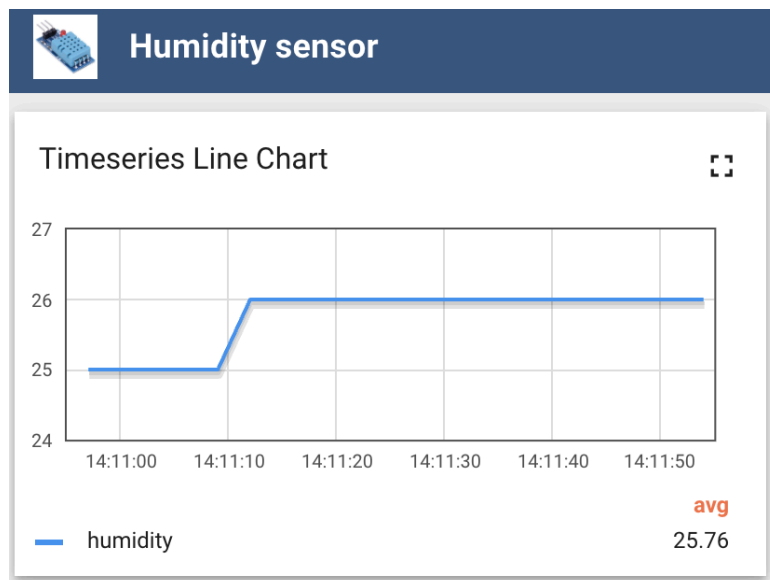
Cancel

Add

From Current bundle dropdown choose Charts. Then click Add to dashboard.



To further expand functionality of dashboards you can go to Edit mode. You can edit the whole dashboard or just individual widgets. Here, we removed the default widget title and added a logo to be shown when the dashboard is in fullscreen mode.



Let's add a simple alarm that will notify us when humidity levels are too high. Go to Devices and select device, choose edit device profile.

humidity-sensor

Device details

?

×

✓

×

Name*

humidity-sensor

Device profile*

default

×

✎

default

Edit device profile

×

Default device profile

✎

Transport type*

Default

▼

Alarm rules (0)

^

No alarm rules configured

Add alarm rule

Device provisioning

^

Cancel

Save

Edit alarm rule condition

Key filters

Key name

Key type

humidity

Time series

✎

×

Add key filter

Filter preview

humidity greater than 60

Condition type

Simple

Cancel

Now we have created an alarm. We can add new widget to our dashboard.

Add widget: Alarm count
Basic
Advanced
?
X

Datasource

Entity alias
humidity-sensor

Filter
Create new

Filter

Reset

Alarm status list

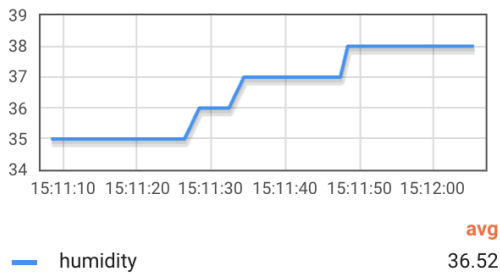
Active
Cleared
Acknowledged
Unacknowledged

Alarm severity list

Critical
Major
Minor
Warning
Indeterminate

Cancel
Preview
Add

Humidity sensor



Alarms

🕒 Realtime - last 2 hours

<input type="checkbox"/>	Created time ↓	Originator	Type	Severity	Status	Assignee
<input type="checkbox"/>	2025-05-14 14:38:17	humidity-sensor	High humidity	Warning	Active Unacknowledged	Unas... ▼ ⋮
<input type="checkbox"/>	2025-05-14 14:23:36	humidity-sensor	High humidity	Warning	Cleared Unacknowledged	Unas... ▼ ⋮

Items per page:

10 ▼

1 – 2 of 2

⏪ ⏴ ⏵ ⏩

To verify everything check from mobile phone or other device.

Keep nodeMCU plugged to the power source.

Final code

```
#include <PubSubClient.h>
#include <ESP8266WiFi.h>
#include <stdlib.h>

#include <DHT.h>

#include <credentials.h>

#define DPIN 4
#define DTYPE DHT11

DHT dht(DPIN,DTYPE);

WiFiClient client;
PubSubClient pubsub_client(client);

void setup() {
  Serial.begin(9600);
  dht.begin();

  delay(10);

  Serial.println("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, pass);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  pubsub_client.setServer("demo.thingsboard.io", 1883);

  while (!pubsub_client.connect(device_id, access_token, NULL)) {
    delay(500);
  }
}
```

```

    Serial.print(".");
}
    Serial.println("");
    Serial.println("Thingsboard connected");

}

void loop() {
    delay(2000);

    float humidity = dht.readHumidity();
    Serial.println(humidity);

    String payload = "{";  payload += "\"humidity\":";
    payload += humidity;
    payload += "}";
    Serial.println(payload);

    if(pubsub_client.publish("v1/devices/me/telemetry",payload.c_str()))

        Serial.println("Published");
}

```

Sources

Github link to the project github.com/anikapet/humidity-sensor

ThingsBoard quick guide [Getting Started with ThingsBoard](#)

Guide to measure WiFi signal in dB by NodeMCU and publish it though ThingsBoard

[Visualizing telemetry data using NodeMCU IoT device with ThingsBoard IoT Dashboard | by Isura Nirmal | Medium](#)

[NodeMCU Language Reference Manual](#)

<https://thingsboard.io/docs/user-guide/rule-engine-2-0/tutorials/create-clear-alarms/>

ESP8266 pin mapping <https://randomnerdtutorials.com/esp8266-pinout-reference-gpios/>

DHT sensor Library <https://github.com/adafruit/DHT-sensor-library>

PubSubClient <https://docs.arduino.cc/libraries/pubsubclient/>

WiFi Client <https://docs.arduino.cc/language-reference/en/functions/wifi/client/>