

Guía IoT Diseño y creación de Interfaces 2022

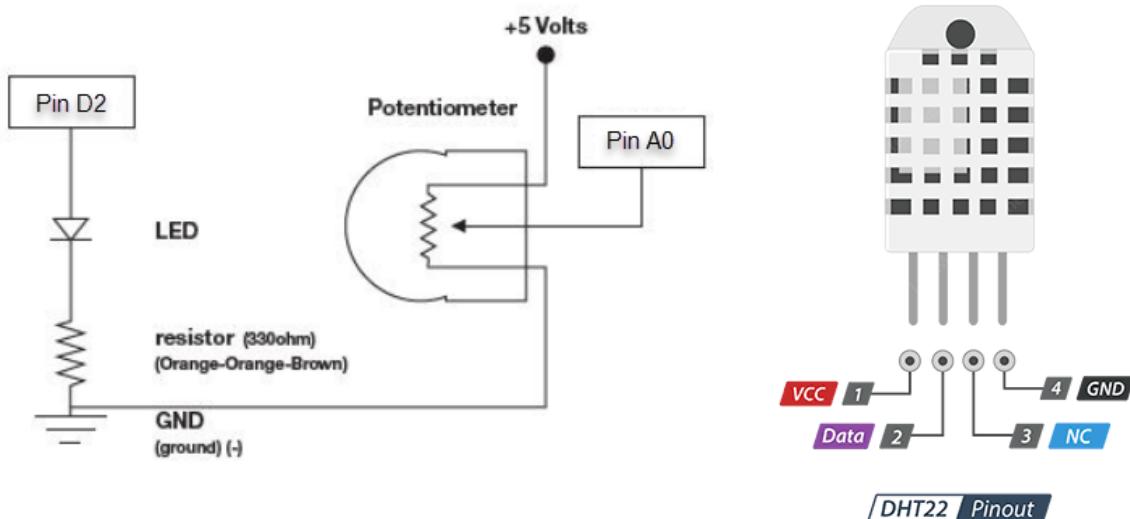
Preparado por: Camilo Becerra y Paul Aguayo

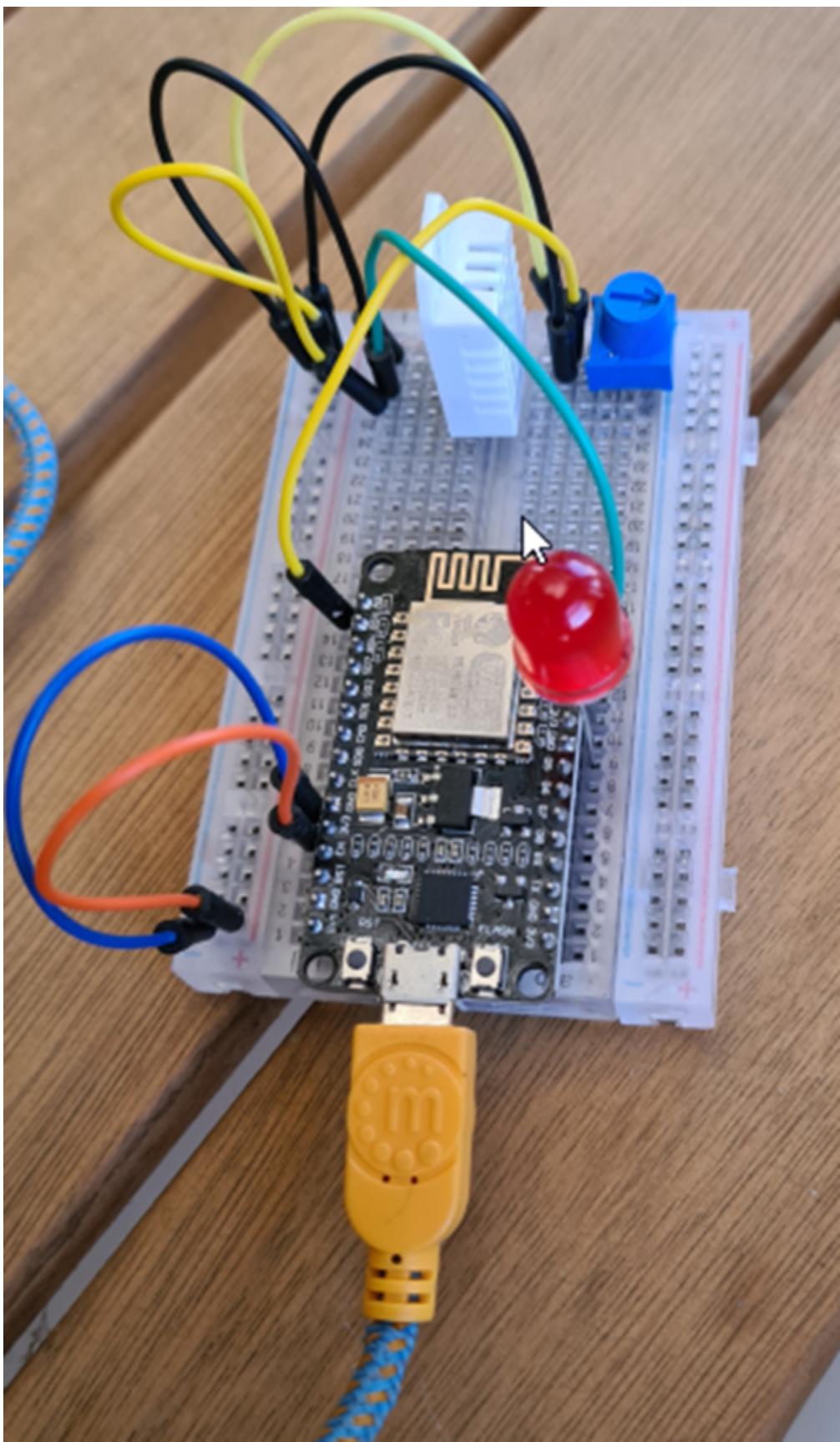
paguayo@edu.uai.cl

La siguiente guía nos muestra cómo utilizar la NodeMCU para leer datos análogos o digitales y enviarlos a la plataforma cloud de Arduino, la cual nos permitirá crear dashboards para monitorear las variables que estamos sensando permitiéndonos acceder desde cualquier parte del mundo a la información que recolectemos. Adicionalmente desde la misma plataforma podremos generar en forma automática el código para nuestra tarjeta y programarla.

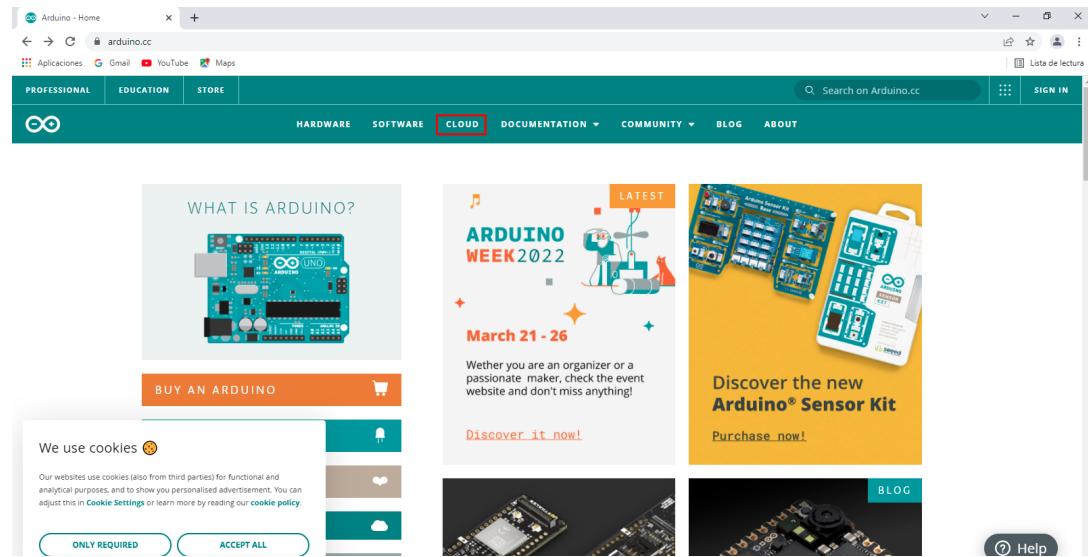
NodeMCU es una tarjeta de desarrollo basada en el módulo ESP8266 la cual cuenta con conectividad WiFi y además es programable con el IDE de Arduino.

Para poder demostrar el uso de la plataforma, crearemos un circuito simple que nos permita leer una entrada analógica (potenciómetro A0) y también controlar una salida digital (LED pin D2). Adicionalmente conectaremos un sensor de Temperatura y Humedad DHT22 (D1).

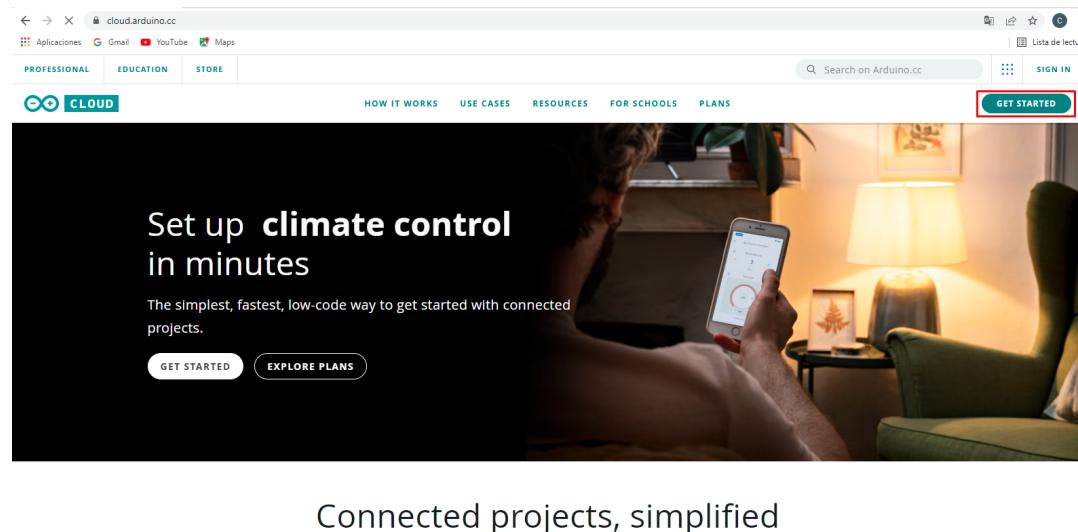




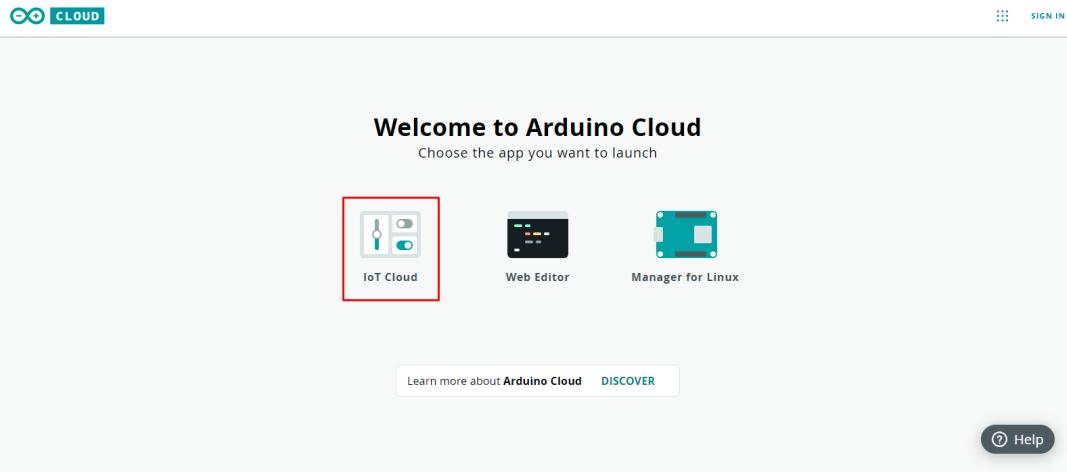
Una vez tengamos armado nuestro circuito ingresamos al sitio web arduino.cc y seleccionamos **CLOUD**



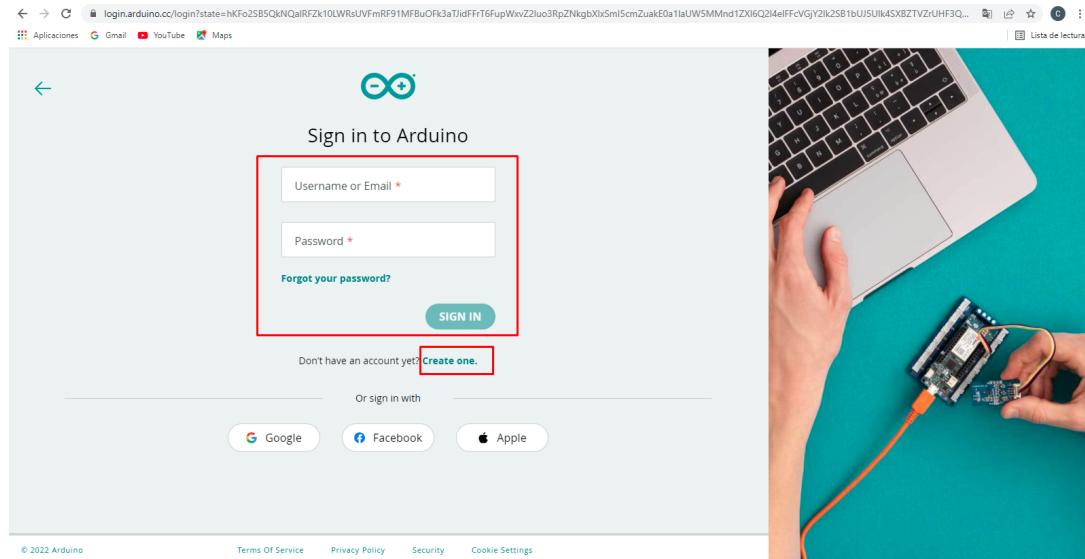
en la página del cloud debemos seleccionar “**GET STARTED**” en la esquina superior derecha



Veremos 3 opciones. En esta guía utilizaremos dos de ellas, el IoT Cloud y el Web Editor. Comenzaremos escogiendo la opción “**IoT cloud**”



Si ya tienes cuenta en arduino.cc puedes usarla para ingresar, de lo contrario debes crear una; es gratuita. Al crear una cuenta se solicita la fecha de nacimiento, esto es porque las cuentas de menores de edad no registran datos manteniendo su privacidad.



Una vez creada la cuenta podemos crear y configurar nuestros dispositivos en la plataforma. La versión gratuita permite crear 2 “things”, que corresponden a los dispositivos IoT o tarjetas de control. Cada una puede tener conectada uno o más sensores o actuadores. Para la versión gratuita el máximo de sensores es de 5 por dispositivo. Para obtener más información sobre los planes disponibles puedes visitar <https://cloud.arduino.cc/plans>



Create your first Thing

A Thing is a connected device that can communicate with the cloud. You can make your Things interact with other Things or anything else in the physical world.

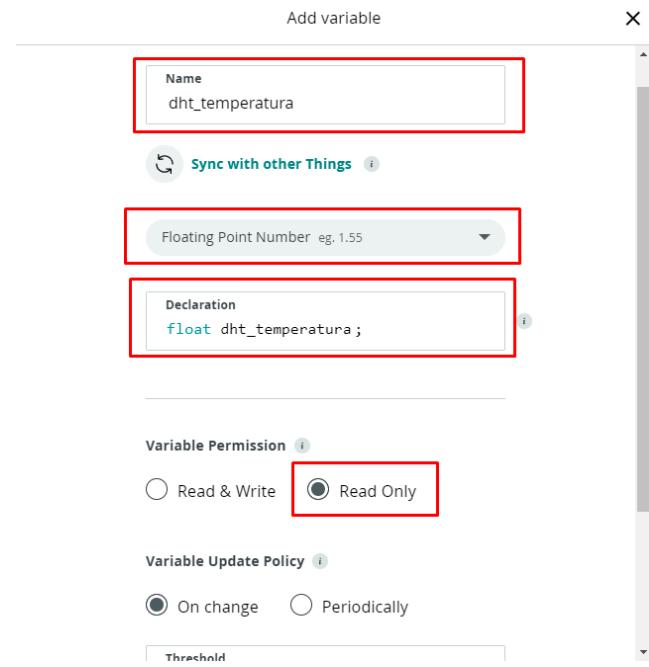
[CREATE THING](#)

A continuación, crearemos y asignaremos la información de nuestro dispositivo, para ello haz click en “**CREATE THING**”

Asignamos un nombre cambiando “**Untitled**” por el que nosotros queramos. Cada lectura ya sea digital o análoga que queramos enviar a la nube debemos asociarla con una variable para que la plataforma pueda identificarla, para hacer esto debemos presionar “**ADD VARIABLE**”

The screenshot shows the 'Setup' tab for a 'Untitled' thing. The 'Variables' section contains a button labeled 'ADD VARIABLE' with a red box around it. The 'Device' section has a 'Select Device' button. The 'Network' section has a placeholder 'Enter your network credentials to'. A sidebar on the right has a 'Feedback' button.

Para este ejemplo comenzaremos con el sensor de temperatura y humedad relativa, DHT22. El nombre de la variable lo escogemos nosotros, en este caso será **dht_temperatura** y la definiremos de tipo flotante (float) ya que de esta forma nos permitirá el uso decimales en la medición. Configuraremos el sensor como solo lectura y le pediremos que nos envíe un mensaje cuando se produzca un cambio en la información del mismo. La magnitud del cambio que queramos detectar es configurable por medio del parámetro “**Threshold**”. La plataforma permite también el envío de datos en una periodicidad establecida.



Presiona “ADD VARIABLE” para crear la variable

Variable Permission

Read & Write Read Only

Variable Update Policy

On change Periodically

Threshold
0

ADD VARIABLE CANCEL

Realizamos el mismo procedimiento para crear variable **dht_humedad**

Add variable X

Name
dht_humedad

 Sync with other Things i

Floating Point Number eg. 1.55

Declaration
`float dht_humedad ;`

Variable Permission i

Read & Write Read Only

Variable Update Policy i

On change Periodically

Variable Permission i

Read & Write Read Only

Variable Update Policy i

On change Periodically

Threshold
0

ADD VARIABLE CANCEL

Creamos la variable con “**ADD VARIABLE**”

Ahora creamos una nueva variable para una lectura analógica, en este caso será de tipo entero (int) y solo lectura

Add variable X

Name
lectura_analogica

 Sync with other Things i

Integer Number eg. 1

Declaration
`int lectura_analogica;` i

Variable Permission i

Read & Write Read Only

Variable Update Policy i

On change Periodically

Variable Permission i

Read & Write Read Only

Variable Update Policy i

On change Periodically

Threshold
0

ADD VARIABLE CANCEL

Creamos la variable con “**ADD VARIABLE**”

La variable led se crea de la misma manera, pero en este caso será de tipo “**bool**” y de lectura y escritura puesto que queremos poder consultar el estado del LED además de encenderlo y apagarlo en forma remota.

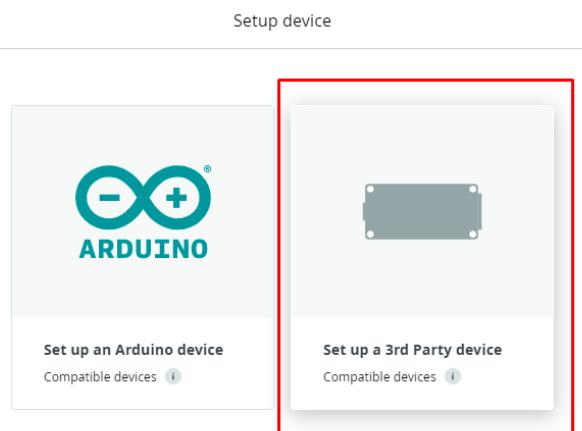
Con las variables creadas seleccionamos el dispositivo (NodeMCU) que vamos a utilizar.

The screenshot shows the IoT Cloud setup interface. On the left, under 'Variables', there is a table with four entries:

Name	Last Value	Last Update
dht_humedad float dht_humedad;	-	-
dht_temperatura float dht_temperatura;	-	-
lectura_analogica int lectura_analogica;	-	-

On the right, under 'Device', it says 'Select the device you want to use or configure a new one.' with a 'Select Device' button highlighted by a red box. Below that is the 'Network' section with a 'Configure' button.

En este caso seleccionaremos las tarjetas compatibles. Si tuviéramos alguna tarjeta arduino original como con WiFi como la MKR1000, MKR1010, Nano33 IoT, etc deberíamos seleccionar la opción de la izquierda “Set up an Arduino device”



Seleccionamos el tipo de tarjeta y el modelo. Para el NodeMCU escogemos ES8266

[←](#) Setup device [X](#)

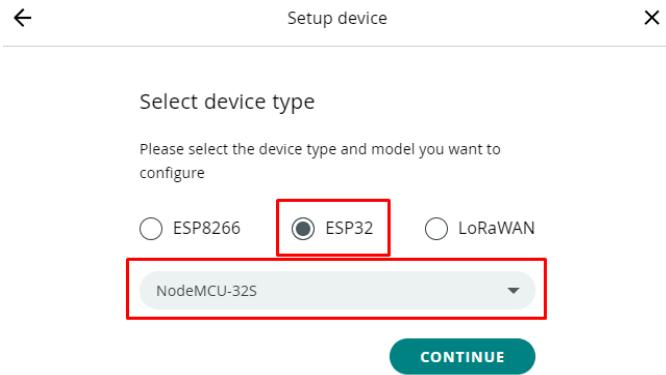
Select device type

Please select the device type and model you want to configure

ESP8266 **ESP32** LoRaWAN

NodeMCU-32S [▼](#)

CONTINUE



Elegimos un nombre para nuestro dispositivo

[←](#) Setup device [X](#)

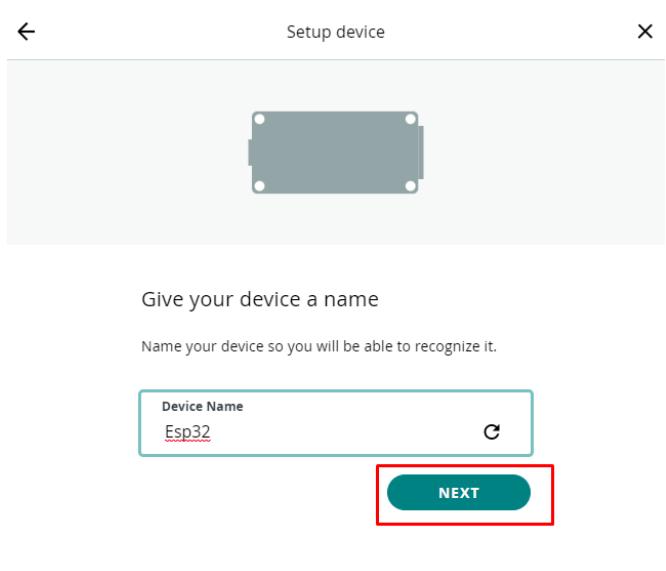


Give your device a name

Name your device so you will be able to recognize it.

Device Name
Esp32 [C](#)

NEXT



Tras presionar “**NEXT**” obtendremos el mensaje de dispositivo listo. **Es importante descargar el pdf que allí aparece porque tendremos una clave para utilizar el dispositivo de forma segura.** Esta clave no es recuperable por lo que debes mantenerla en un lugar seguro. Si la pierdes la única opción es volver a crear el dispositivo.

Make your device IoT-ready

To use this board you will need a Device ID and a Secret Key, please copy and save them or [download the PDF.](#)

Also, keep in mind that this device authentication has a lower security level compared to other Arduino devices.

Device ID

08d599e1-b4e4-47aa-a542-c959c05a0c1e

Secret Key

4EWSTYWUYR0QLWEB8Y5X



Secret key cannot be recovered

Please keep it safe, if you lose it you will have to delete and setup your device again.

I saved my device ID and Secret Key

CONTINUE

El pdf que descargas va a contener la información que se muestra a continuación. La "Secret Key" es la clave de la que se hizo mención anteriormente. Estos datos nos permitirán identificar en forma única nuestro dispositivo.



Esp32

ESP32
NodeMCU-32S

Configured on March 9, 2022

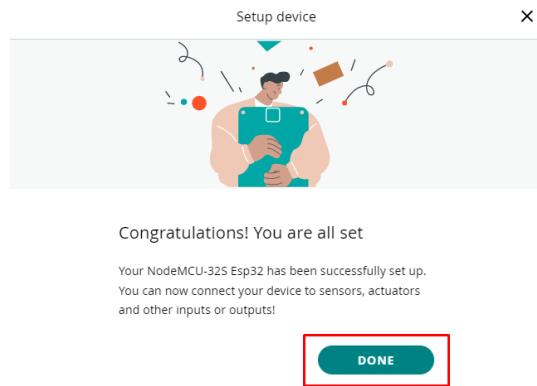
Device ID

08d599e1-b4e4-47aa-a542-c959c05a0c1e

Secret Key

4EWSTYWUYR0QLWEB8Y5X

Terminado este paso presionamos “DONE”



Lo siguiente es configurar la red WiFi en nuestro dispositivo para eso vamos a **Network**, presionamos “**configure**”

A screenshot of the IoT Cloud interface showing the "Setup" tab. On the left, there's a "Variables" section with four entries: "dht_humedad", "dht_temperatura", "lectura_analogica", and "led". On the right, there's a "Device" section for an "Esp32" device with ID 08d599e1-bd4d-47aa-a542-... and Type NodeMCU-32S. The status is Offline. There are "Change" and "Detach" buttons. Below the device section is a "Network" section with a note: "Enter your network credentials to connect your device." At the bottom, there are "Set webhook" and "Timezone: America/NewYork" buttons.

Agregamos las credenciales de red, nombre de red (SSID), password y serial key del pdf que descargamos

A screenshot of the "Configure network" dialog. It contains three input fields: "Wi-Fi Name *" with value "MC11101B", "Password *" with value "*****", and "Secret Key *" with value "4EWSTYWUYR0QLWEB8Y5X". Each field has a red box drawn around it. At the bottom is a green "SAVE" button.

Así se vería la pantalla con todo lo que se ha realizado hasta ahora

The screenshot shows the IoT Cloud interface. On the left, under 'Variables', there is a table with four rows:

Name	Last Value	Last Update
dht_humedad float dht_humedad;	-	-
dht_temperatura float dht_temperatura;	-	-
lectura_analogica int lectura_analogica;	-	-

On the right, under 'Device', it shows an 'Esp32' device with the following details:

- ID: 08d599e1-b4e4-47aa-a542-...
- Type: NodeMCU-32S
- Status: Offline

Below the device section is a 'Network' section with Wi-Fi settings:

- Wi-Fi Name: MC110...
- Password:
- Secret Key:

At the bottom, there are buttons for 'Set webhook' and 'Timezone: America/New York'.

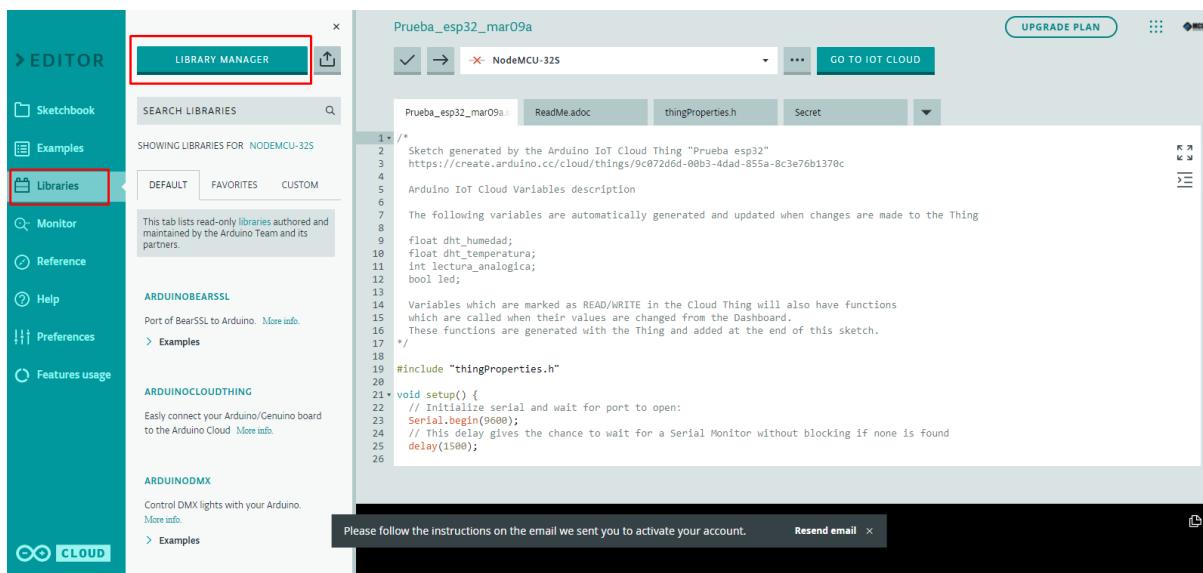
Para continuar, seleccionamos Sketch y Open full editor

The screenshot shows the IoT Cloud interface with the 'Sketch' tab selected. It displays a code editor with the following Arduino sketch:

```
1 //*
2 * Sketch generated by the Arduino IoT Cloud Thing "Prueba esp32"
3 * https://create.arduino.cc/cloud/things/9c072d6d-00b3-4dad-855a-8c3e76b1370c
4 *
5 * Arduino IoT Cloud Variables description
6 *
7 * The following variables are automatically generated and updated when changes are made to the Thing
8 *
9 * float dht_humedad;
10 * float dht_temperatura;
11 * int lectura_analogica;
12 * bool led;
13 *
```

Below the code editor, there is a message: 'No associated device found'. At the bottom right of the code editor area, there is a button labeled '</> Open full editor'.

El siguiente paso es necesario para poder utilizar el sensor DHT22 puesto que necesita una librería especial. Si no lo vas a utilizar, puedes omitir este paso. En el editor online, seleccionar “Libraries”, luego “LIBRARY MANAGER”



Buscar dht y seleccionar estrella en “DHT SENSOR LIBRARY FOR ESPX”, luego done

Favorite the Contributed libraries you use the most

All the libraries included in the Library Manager are already available for your sketches. Star your favorite ones to download them, see the related examples, and select a specific version (if available).

dht

DHT SENSOR LIBRARY
Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors

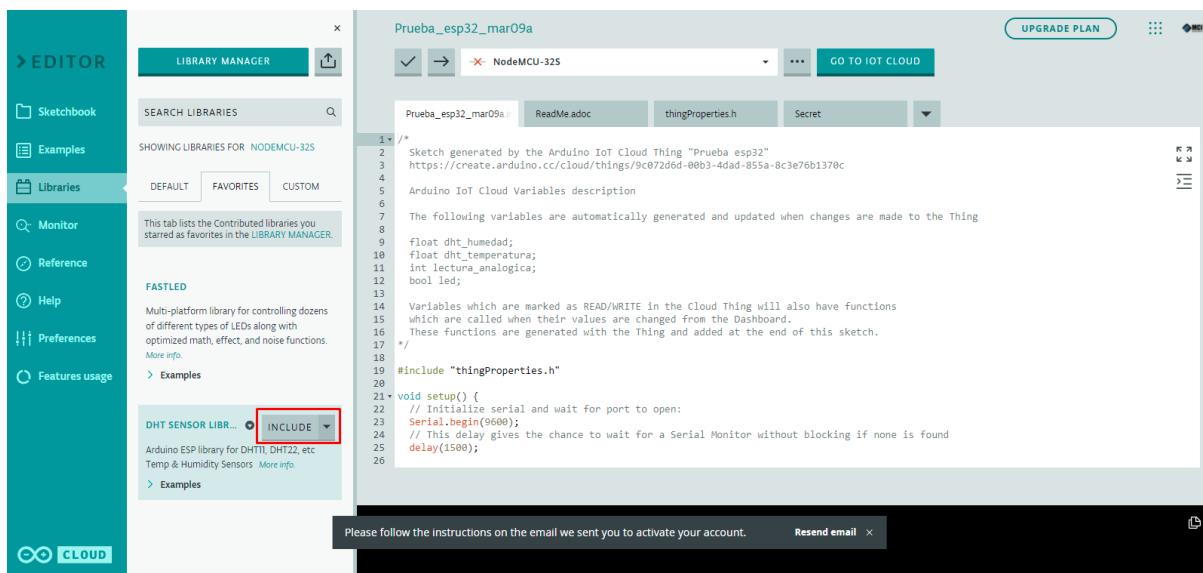
DHT SENSOR LIBRARY FOR ESPX
Arduino ESP library for DHT11, DHT22, etc Temp & Humidity Sensors

DHT12
Arduino library for I2C DHT12 temperature and humidity sensor.

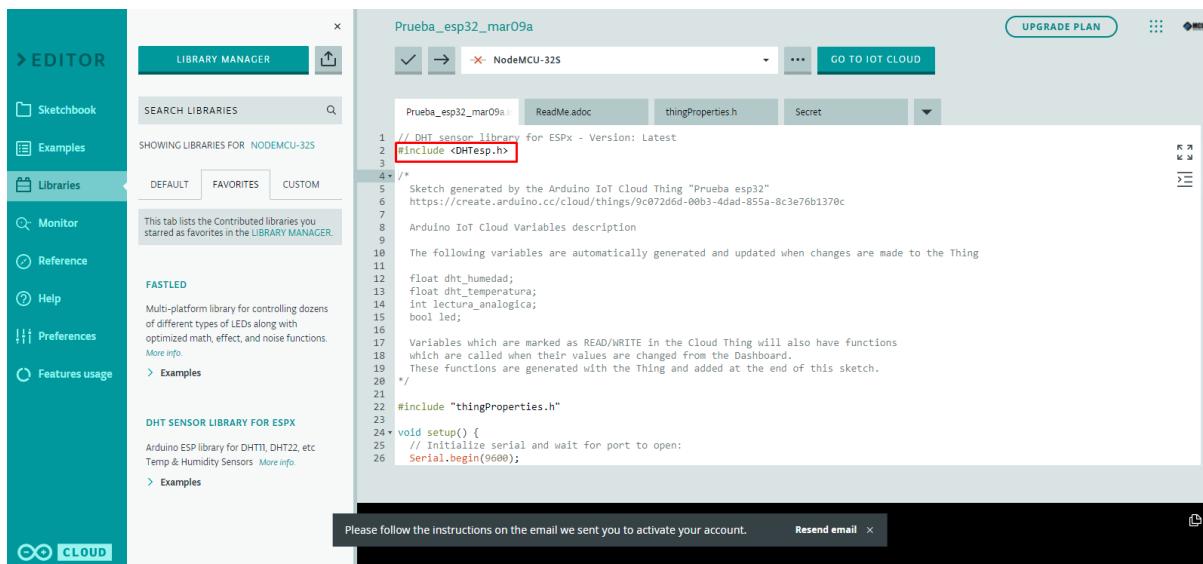
DHT12 SENSOR LIBRARY
Arduino, ESP8266 and ESP32 library for DHT12 Temp & Humidity Sensors

DONE

De vuelta en la ventana del editor online, presionar “INCLUDE” en el recuadro de la librería



De esta forma, la librería será incluida en el código que vamos a cargar a la tarjeta.



Creamos el objeto dht de tipo **DHTesp** antes del void setup () como se muestra en la siguiente imagen y agregamos la siguiente línea dentro del setup para configurar la librería.

`dht.setup(pin, DHTesp::DHT22);`

donde **pin** corresponde a pin donde está físicamente conectado el sensor DHT22.

```

Prueba_esp32_mar09a
✓ → x- NodeMCU-32S ... GO TO IOT CLOUD
Prueba_esp32_mar09a ReadMe.adoc thingProperties.h Secret
22 #include "thingProperties.h"
23 DHTesp dht; // objeto dht de tipo DHTesp
24
25 void setup() {
26   // Initialize serial and wait for port to open:
27   Serial.begin(9600);
28   // This delay gives the chance to wait for a Serial Monitor without blocking if none is found
29   delay(1500);
30
31   // Defined in thingProperties.h
32   initProperties();
33
34   // Connect to Arduino IoT Cloud
35   ArduinoCloud.begin(ArduinoIoTPREFERREDConnection);
36
37 /*
38   The following function allows you to obtain more information
39   related to the state of network and IoT Cloud connection and errors
40   the higher number the more granular information you'll get.
41   The default is 0 (only errors).
42   Maximum is 4
43 */
44 setDebugMessageLevel(2);
45 ArduinoCloud.printDebugInfo();
46 dht.setup(13,DHTesp::DHT22);
47 }

```

Please follow the instructions on the email we sent you to activate your account. [Resend email](#)

En el loop asignamos la lectura de datos a las variables que creamos previamente

```

dht_temperatura = dht.getTemperature();
dht_humedad = dht.getHumidity();
lectura_analógica = analogRead(35);
delay(1000);

```

Importante: debes verificar que los pines que estés utilizando en las funciones de lectura analógica y digital correspondan a la conexión física que tengas en la tarjeta NodeMCU

```

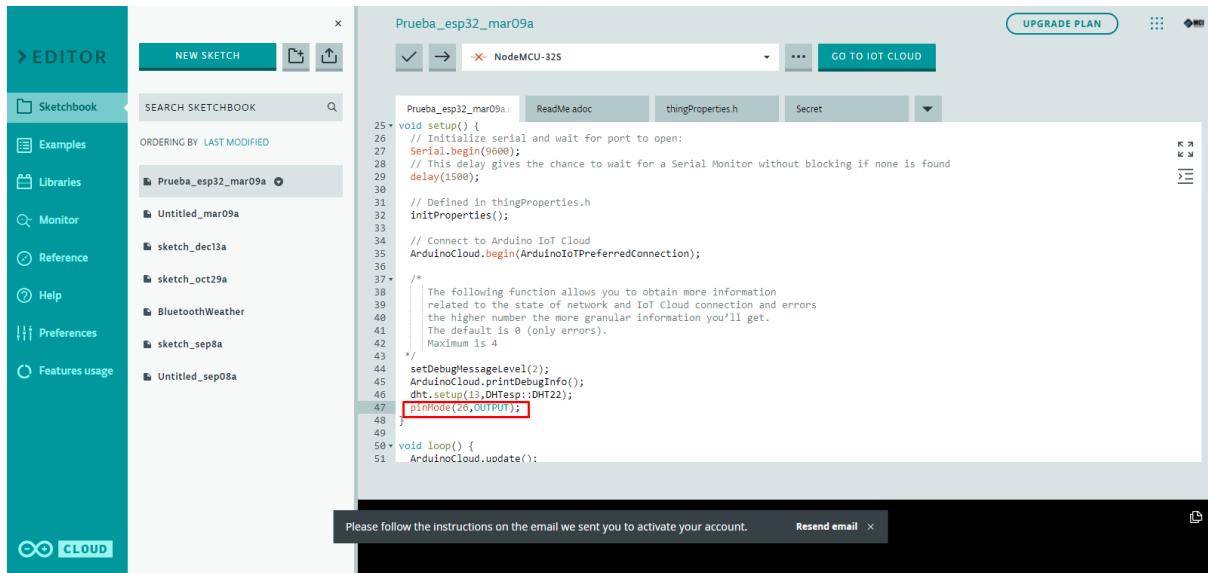
Prueba_esp32_mar09a
✓ → x- NodeMCU-32S ... GO TO IOT CLOUD
Prueba_esp32_mar09a ReadMe.adoc thingProperties.h Secret
33
34 // Connect to Arduino IoT Cloud
35 ArduinoCloud.begin(ArduinoIoTPREFERREDConnection);
36
37 /*
38   The following function allows you to obtain more information
39   related to the state of network and IoT Cloud connection and errors
40   the higher number the more granular information you'll get.
41   The default is 0 (only errors).
42   Maximum is 4
43 */
44 setDebugMessageLevel(2);
45 ArduinoCloud.printDebugInfo();
46 dht.setup(13,DHTesp::DHT22);
47 }
48
49 void loop() {
50   ArduinoCloud.update();
51   // Your code here
52   dht_temperatura = dht.getTemperature();
53   dht_humedad = dht.getHumidity();
54   lectura_analogica = analogRead(35);
55   delay(1000);
56 }
57
58

```

Success: Saved on your online Sketchbook and done verifying Prueba_esp32_mar09a

on this architecture - ArduinoIoTCloud will not keep track of local change timestamps .
pragma message "No RTC available on this architecture - ArduinoIoTCloud will not keep track of local change timestamps ."

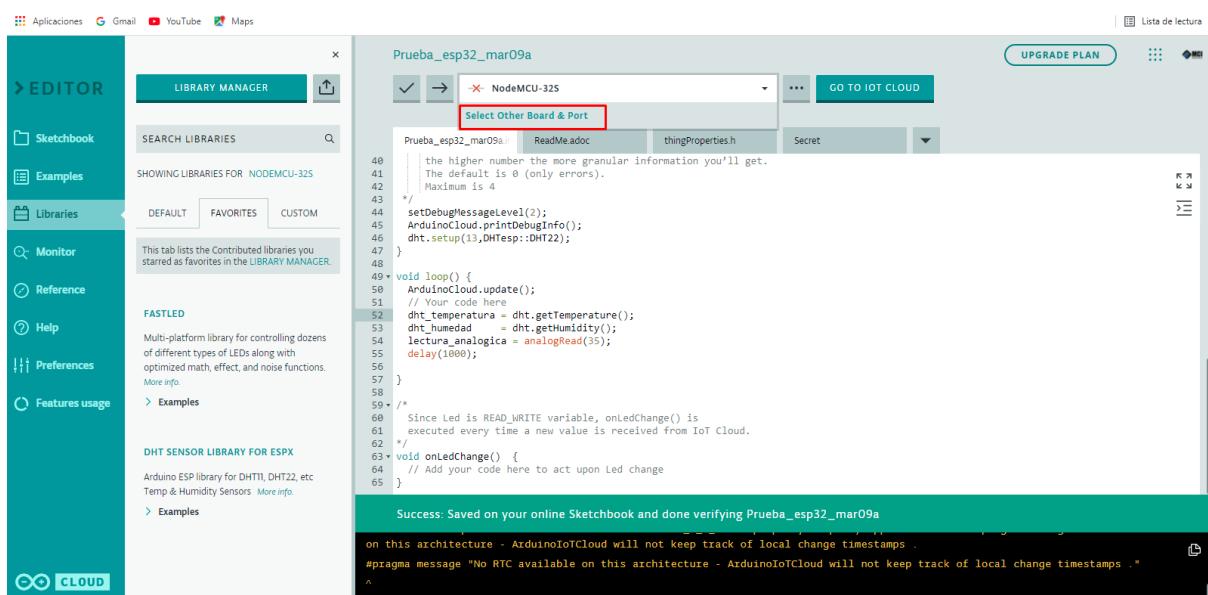
En setup, se debe definir pin donde tenemos conectado el LED como salida



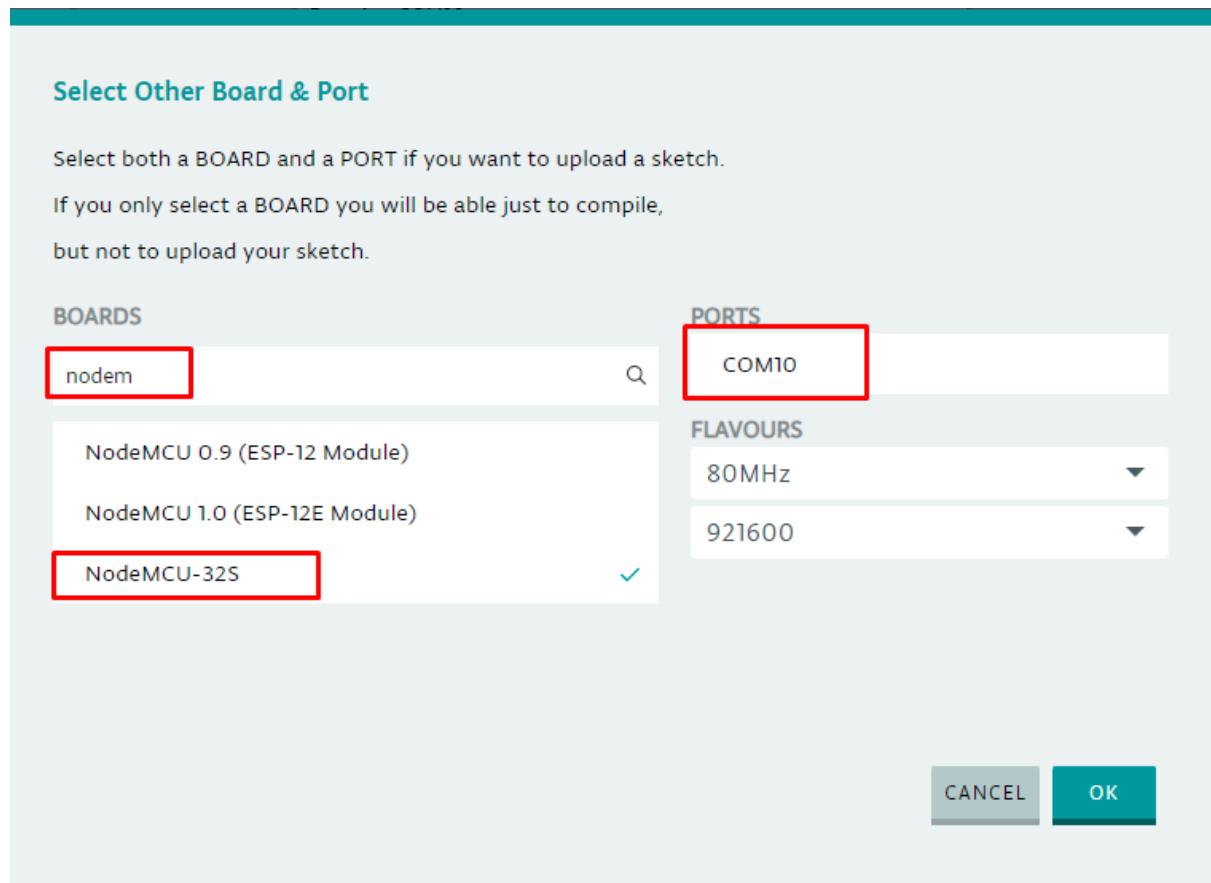
Luego vamos a la función **void onLedChange()** y agregamos el código que queremos que se ejecute cuando cambie la variable Led. En nuestro caso

```
62 Since Led is READ_WRITE variable, onLedChange() is
63 executed every time a new value is received from IoT Cloud.
64 */
65 void onLedChange() {
66     // Add your code here to act upon Led change
67     if(led){
68         digitalWrite(4, HIGH);
69     }
70     else{
71         digitalWrite(4, LOW);
72     }
73 }
```

Finalmente seleccionamos la tarjeta y el puerto



Buscamos el modelo de la tarjeta que vamos a utilizar y el puerto al que está conectada. **Para la NodeMCU se debe seleccionar como tarjeta “Generic ESP8266 Module” para cargar el programa**



Con ello ya es posible cargar el programa a la tarjeta.

Finalmente nos queda configurar cómo se van a ver los datos que vamos a enviar desde nuestra tarjeta. Para eso vamos de vuelta a la página cloud, seleccionamos dashboard y creamos uno nuevo haciendo click en “**Build Dashboard**”

IOT CLOUD

Things Dashboards Devices Integrations Templates UPGRADE PLAN

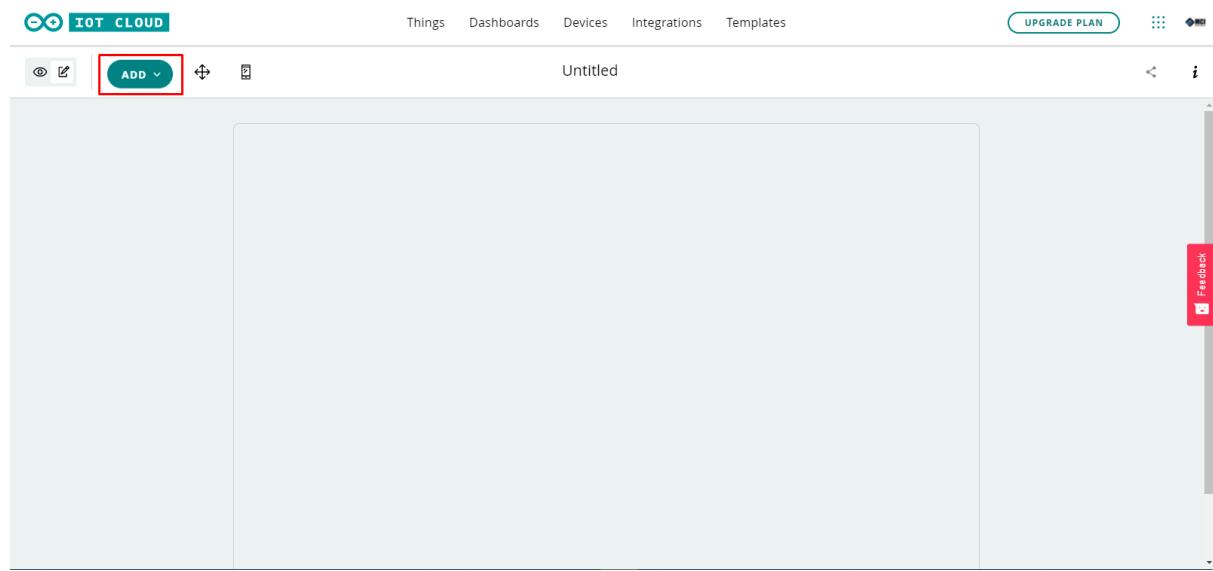
Monitor your Things

Build a Dashboard to easily monitor the status of your Things and control them.

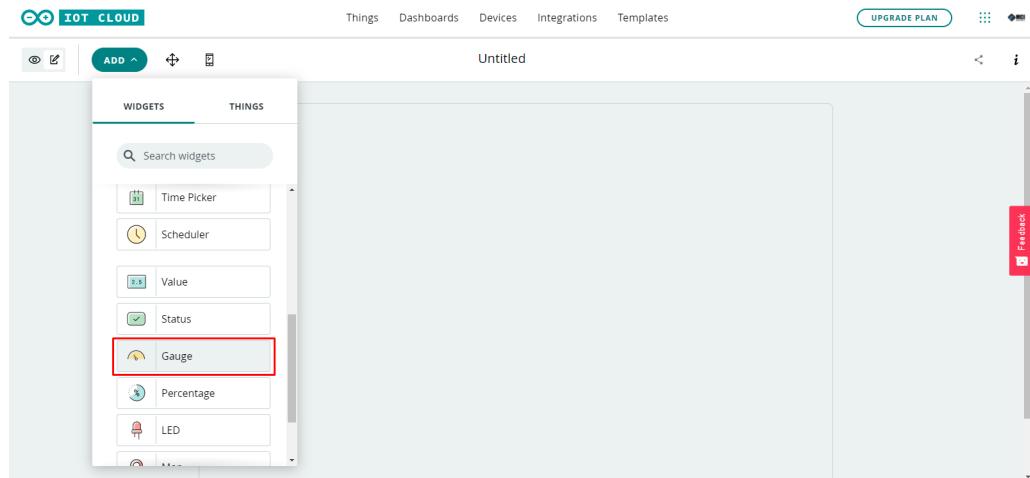
BUILD DASHBOARD

Feedback

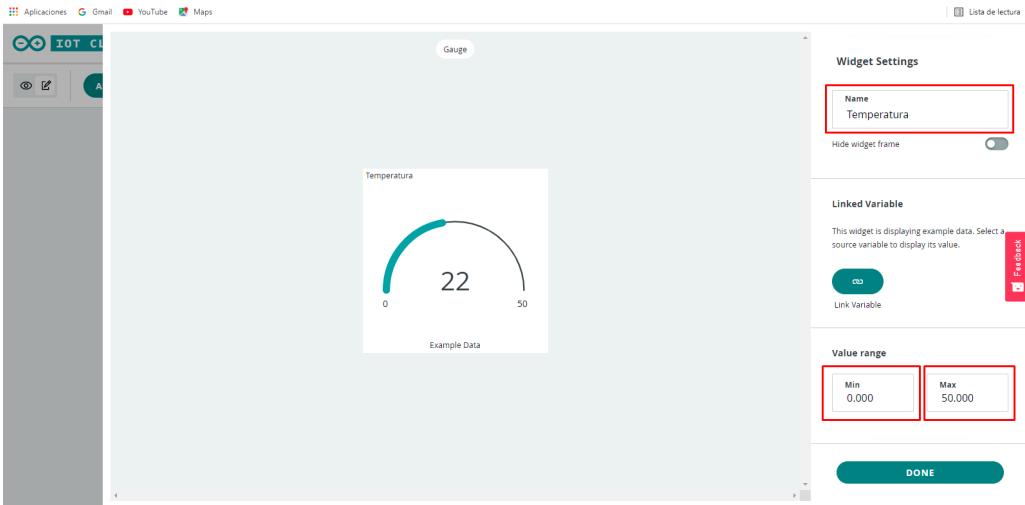
En el cloud agregaremos los widgets presionando “ADD”



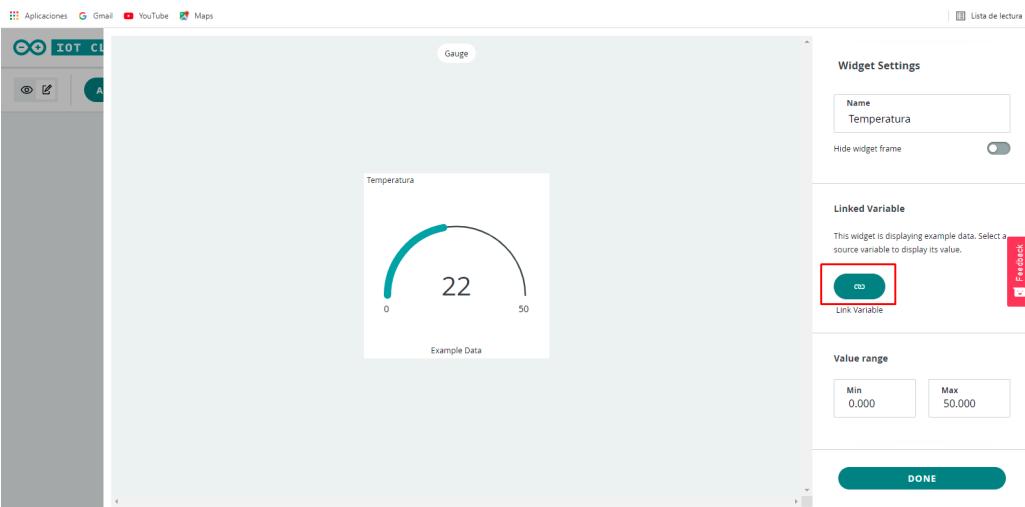
Para mostrar la temperatura seleccionaremos “Gauge”



En los parámetros del widget le damos un nombre, definimos los valores mínimo y máximo



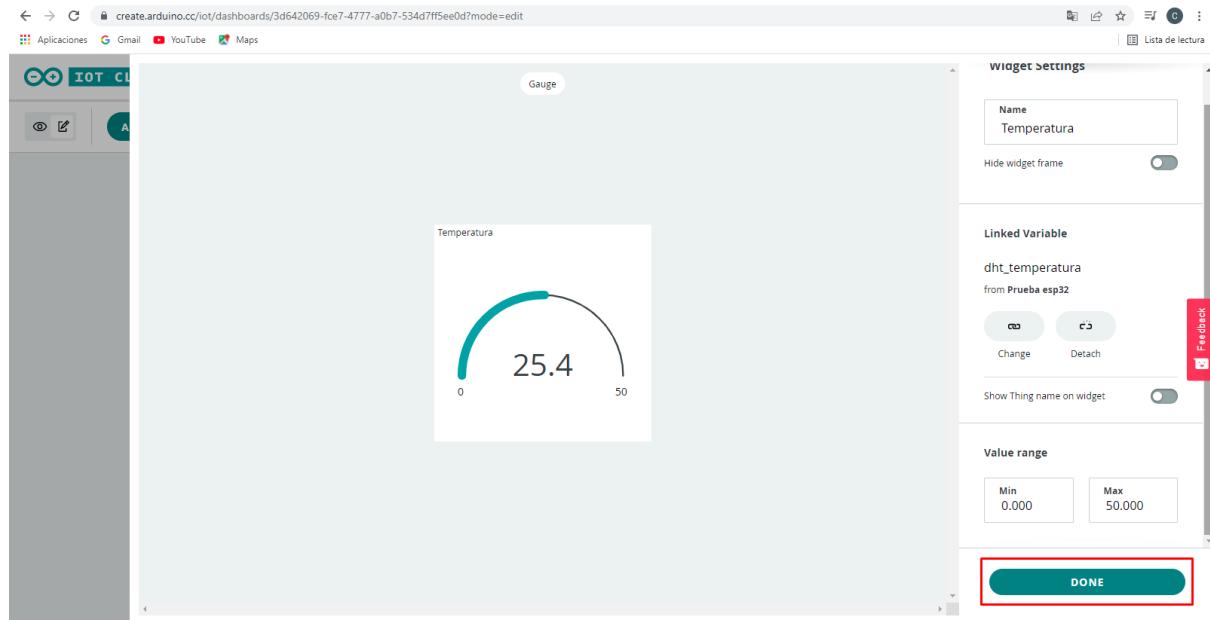
Tenemos que asociar el gauge a la variable que habíamos creado anteriormente



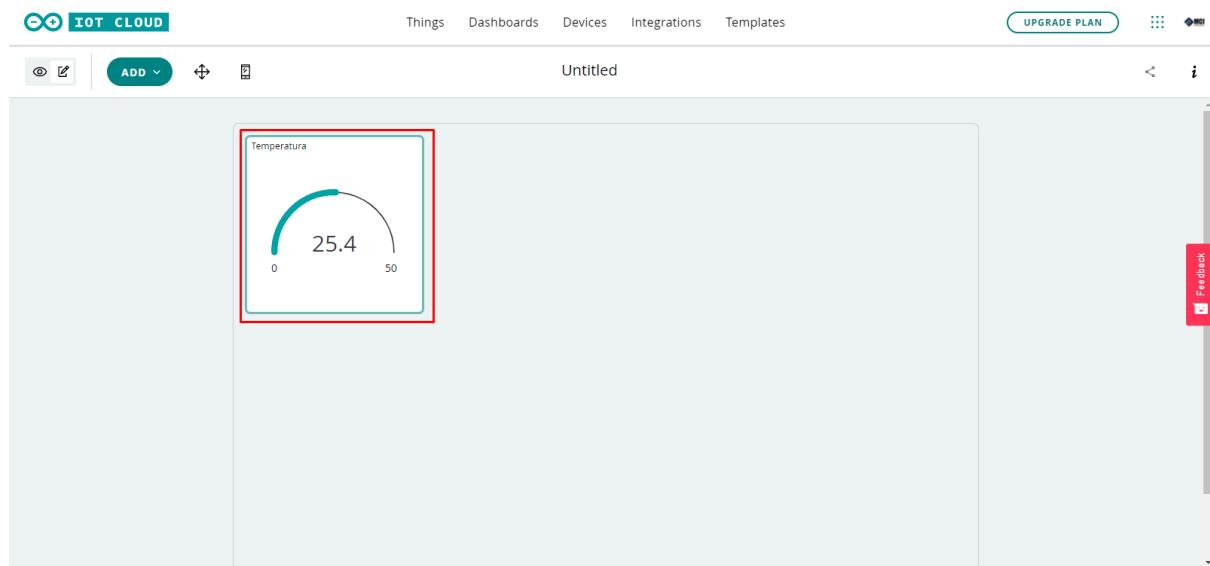
Seleccionamos la variable dht_temperature, luego damos click a “LINK VARIABLE”

The screenshot shows the 'Link Variable to Temperatura' screen. On the left, under 'Things', 'Prueba esp32' is selected. In the center, under 'Variables', 'dht_temperatura' is selected and highlighted with a red box. On the right, the details for 'dht_temperatura' are shown, including its Thing ('Prueba esp32'), Type ('Float'), Last value ('25.4'), and Update policy ('On change'). At the bottom right, a large teal 'LINK VARIABLE' button is highlighted with a red box.

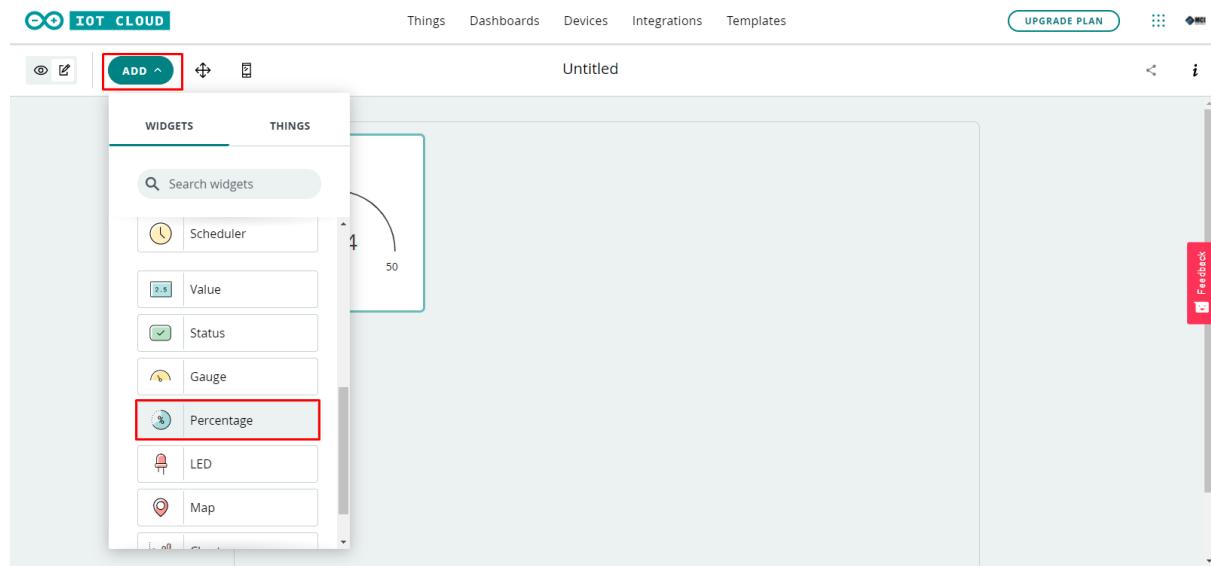
Damos click en “DONE”



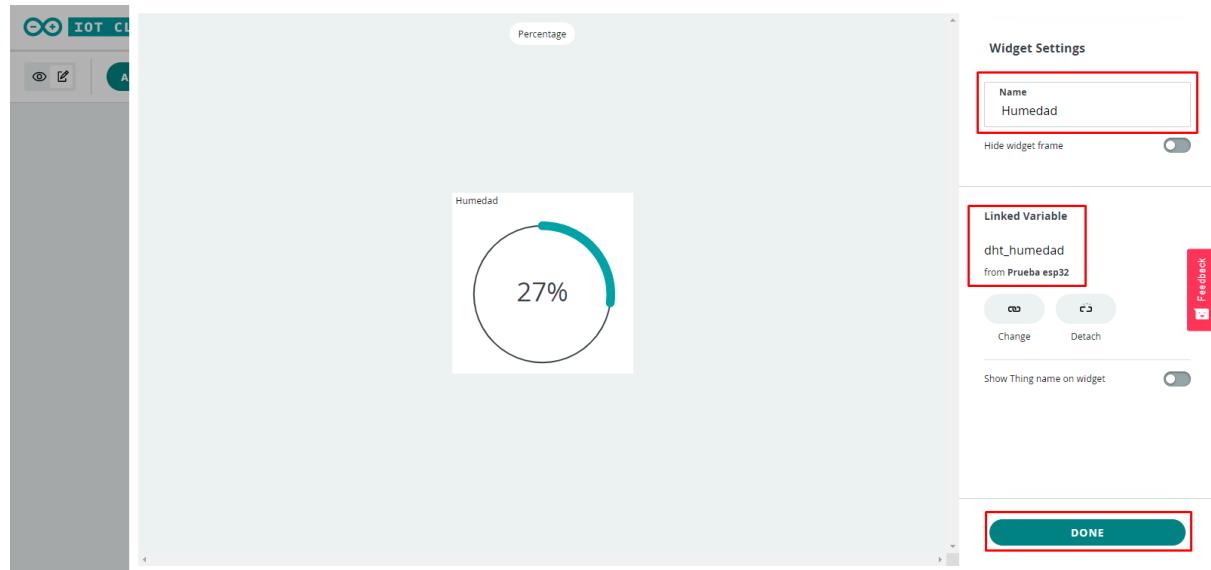
El dashboard ya tiene nuestra variable



Para la humedad hacemos el mismo procedimiento, pero seleccionaremos el widget de porcentaje “Percentage”

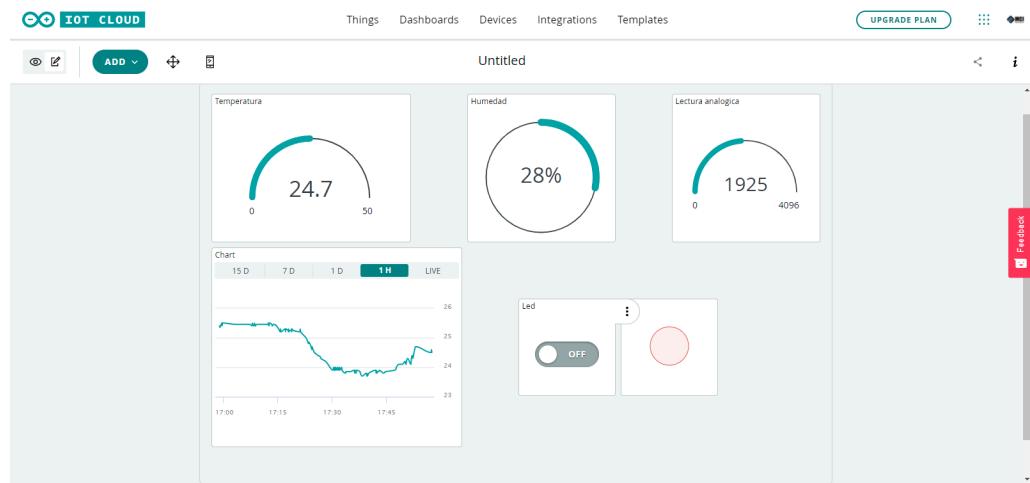


Seguimos el mismo procedimiento anterior

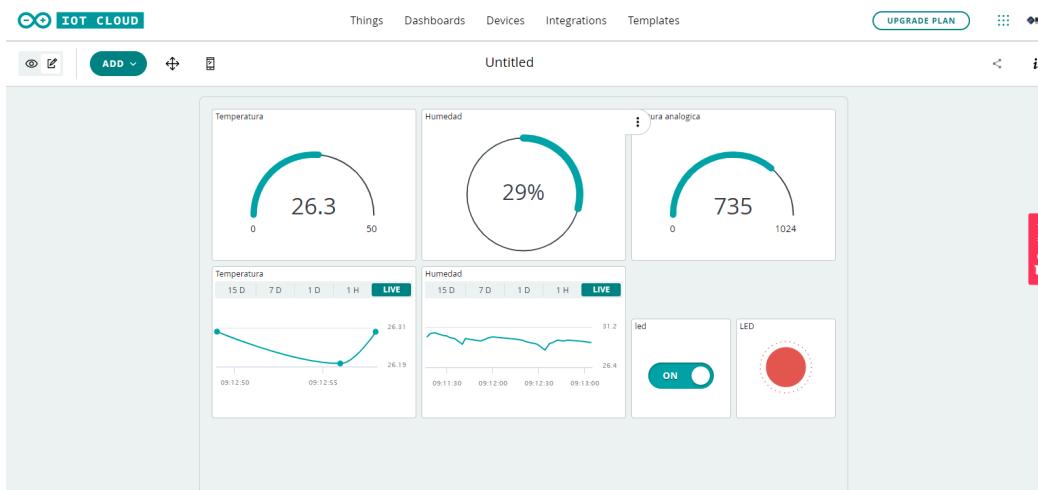


Podemos agregar otros widget asociados a la misma variable para tener más información o comportamiento, por ejemplo, en gráficas

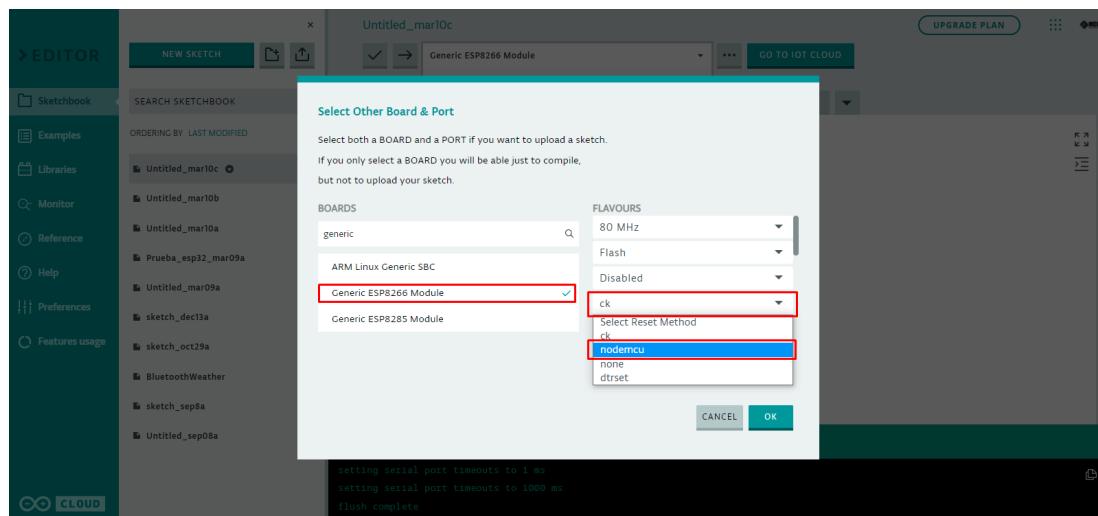
El dashboard completo con las variables quedaría así



El procedimiento para el **esp8266** en vez del **esp32** que se muestra en las imágenes anteriores es el mismo obteniendo los mismos resultados,



Sin embargo existen algunos pequeños cambios. Se debe seleccionar como tarjeta “Generic ESP8266 Module” para cargar el programa



- Lo mismo para la selección del dispositivo

The screenshot shows the TOT Cloud interface with the title "Untitled". At the top, there are tabs for "Setup" and "Sketch" (with a red notification badge). Below these are sections for "Variables" and "Device".

Variables:

Name	Last Value	Last Update
dht_humedad float dht_humedad;	29.2	11 Mar 2022 09:23:56
dht_temperatura float dht_temperature;	26	11 Mar 2022 09:19:08
lectura_analogica int lectura_analogica;	735	11 Mar 2022 09:24:08

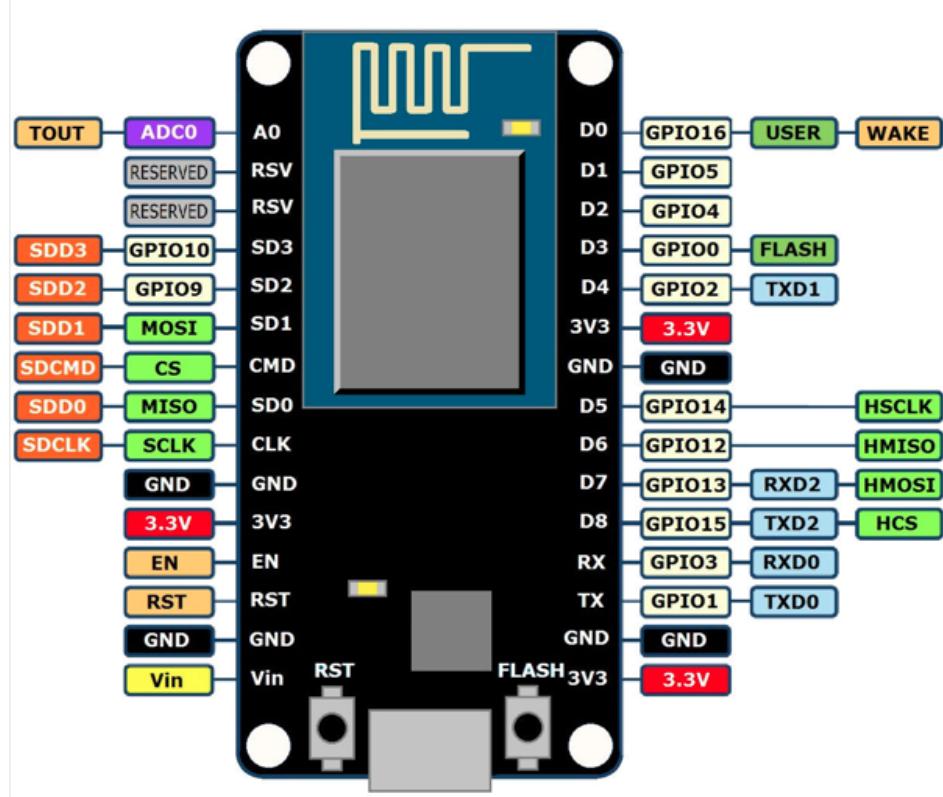
Device:

Melantha

- ID: 8dd2e083-cefe-4584-ac60-d...
- Type: Generic ESP8266 Module
- Status: Online

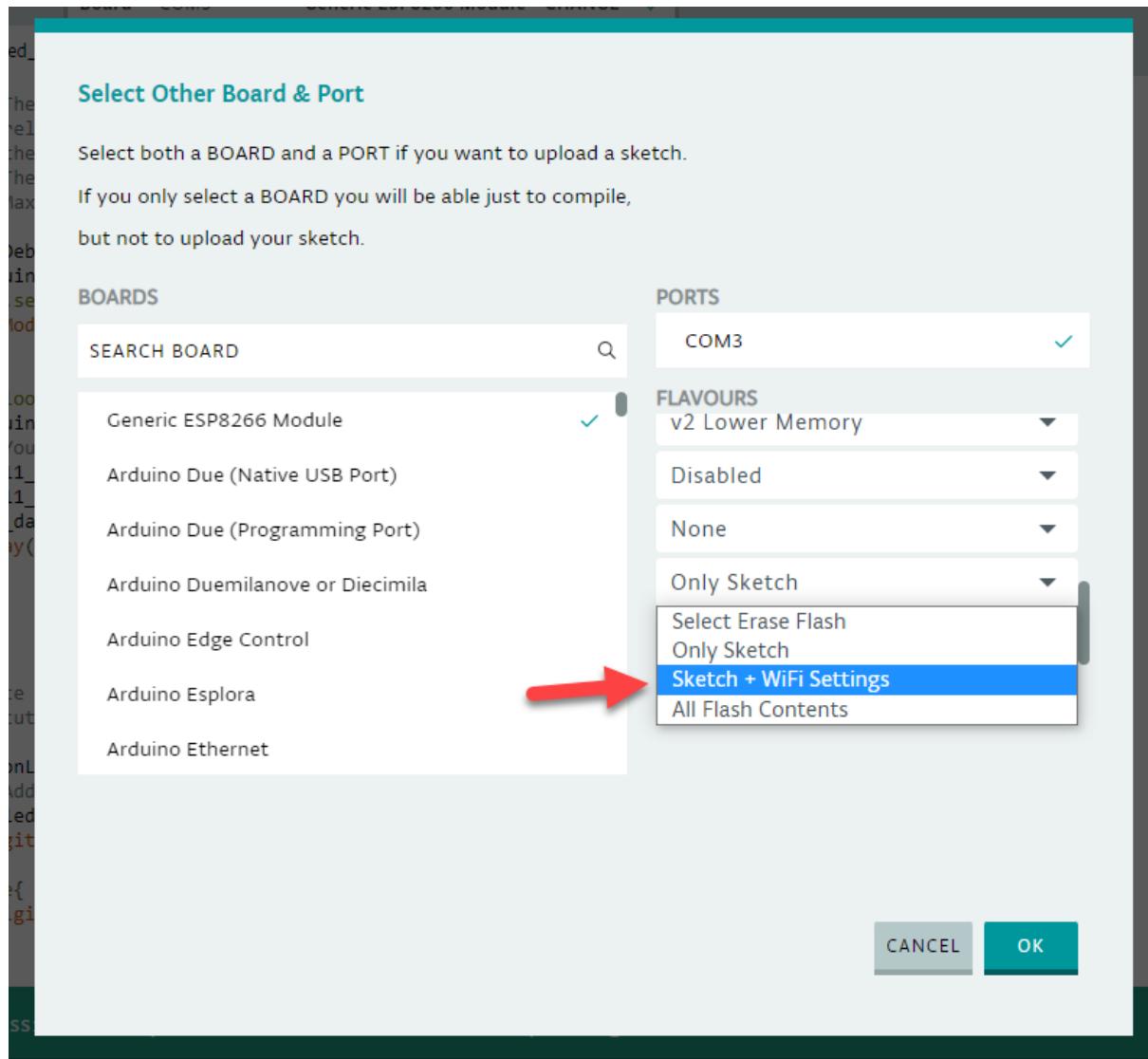
Buttons: Change, Detach

- Los pines usados en el programa se basan en el GPIO no en Dx impresos en la tarjeta

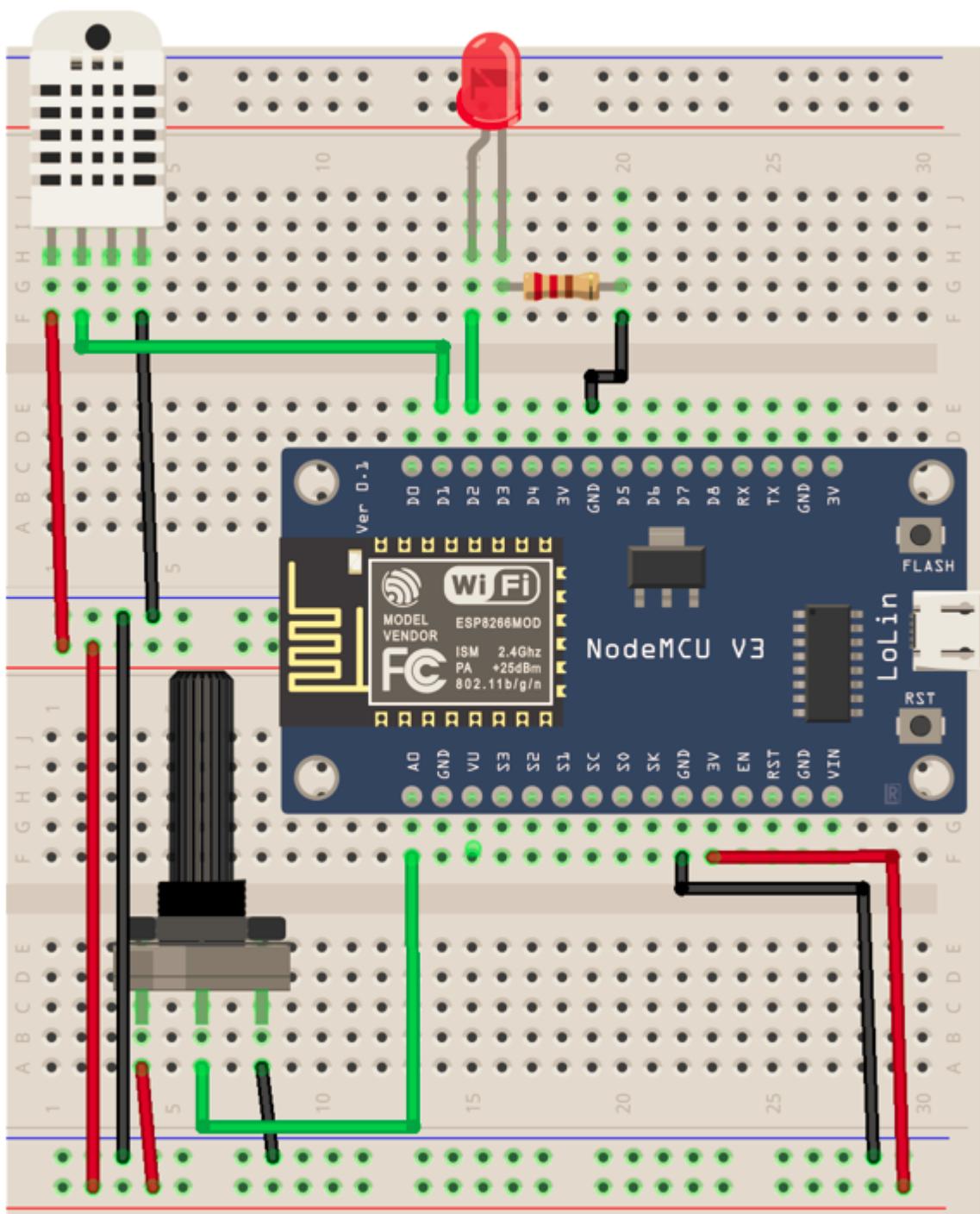


- Se deben usar los pines entre el D0 y D4.
- Programa usado para el Esp8266,
<https://create.arduino.cc/editor/camilobf/8c00add9-4193-4a8a-9034-f11cf5bb08f3/preview>

Si no es la primera vez que utilizas la placa recuerda no solo cargar en sketch en ella si no también la configuración del Wifi



El circuito construido es el siguiente:



El circuito armado se ve de la siguiente manera:

