

# **UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE - L**



**DEPARTAMENTO DE ENERGÍA Y MECÁNICA  
CARRERA DE INGENIERÍA MECATRÓNICA**

**DISEÑO ELECTRÓNICO**

**NRC: 2058**

**TEMA**

---

**INFORME APLICACIONES IOT**

---

**DOCENTE:**

Ing. David Rivas

**NOMBRE:**

Steven Curipallo

Gregory Dávalos

**PERIODO ACADÉMICO:**

Noviembre 20 – Abril 21

## 1. Tema

Aplicación IOT con el ESP8266

## 2. Objetivos

### Objetivo General:

Desarrollar un circuito con un controlador ESP8266 para el desarrollo de una aplicación IOT

### Objetivos Específicos:

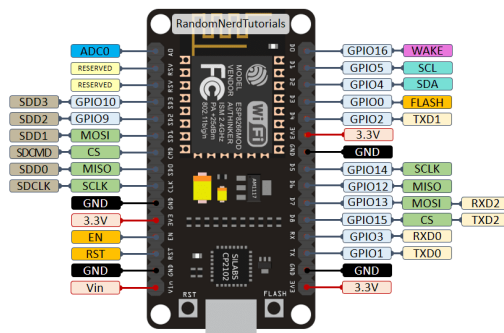
Diseñar una placa PCB del circuito para el futuro desarrollo de la misma

Comprobar el funcionamiento de la plataforma y el controlador desde un acceso remoto

Verificar el funcionamiento del circuito en otra plataforma IOT

## 3. Marco Teórico

### ESP8266



El ESP8266 es un chip de bajo costo Wi-Fi con un stack TCP/IP completo y un microcontrolador, fabricado por Espressif, una empresa afincada en Shanghai, China.

El primer chip se hace conocido en los mercados alrededor de agosto de 2014 con el módulo ESP-01, desarrollado por la empresa AI-Thinker. Este pequeño módulo permite a otros microcontroladores conectarse a una red inalámbrica Wi-Fi y realizar conexiones simples con TCP/IP usando comandos al estilo Hayes.

El ESP8285 es como un ESP8266 pero con 1 MB de memoria flash interna, para permitir a dispositivos de un chip conexiones de Wi-Fi.

El ESP8266 normalmente viene integrado en un módulo. Esto es debido a que el propio SoC ESP8266 no tiene memoria Flash integrada. El primero que vio la luz fue

el ESP-01 el cual estaba pensado para funcionar como interfaz WiFi de las placas de Arduino. Sin embargo, enseguida se hizo muy popular en la comunidad Maker.

A partir de este módulo surgieron muchos más hasta que finalmente irrumpió en el mercado el ESP-12, el más popular de todos los módulos. Este módulo se utiliza en multitud de placas siendo las más famosas NodeMCU y Wemos.

## DHT22



El DHT22 es un sensor de temperatura y humedad con unas prestaciones que lo acercan mucho a los de alta precisión. Lo puedes encontrar fácilmente en tiendas especializadas o grandes superficies, donde lo puedes comprar por unos cuantos euros. Eso te permite no tener que depender de un sensor de temperatura y otro de humedad por separado, sino tenerlo todo integrado en un mismo dispositivo.

Lo puedes encontrar suelto o en módulos especialmente diseñados para Arduino, es decir, el DHT22 montado sobre una placa PCB ya lista para usar, sin tener que agregar resistencias pull-up, etc. Hasta aquí todo se parece bastante el DHT11. Y también tendrás una alta fiabilidad y estabilidad en las mediciones debido a la señal digital calibrada que usa.

## Relé



El relé (en francés, relais ‘relevo’) es un dispositivo electromagnético. Funciona como un interruptor controlado por un circuito eléctrico en el que, por medio de una bobina y un electroimán, se acciona un juego de uno o varios contactos que permiten abrir o

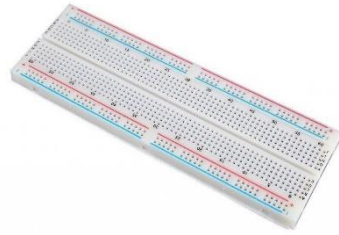
cerrar otros circuitos eléctricos independientes. Fue inventado por Joseph Henry en 1835.

Dado que el relé es capaz de controlar un circuito de salida de mayor potencia que el de entrada, puede considerarse, en un amplio sentido, como un amplificador eléctrico. Como tal se emplearon en telegrafía, haciendo la función de repetidores que generaban una nueva señal con corriente procedente de pilas locales a partir de la señal débil recibida por la línea. Se les llamaba «relevadores».

#### 4. Materiales

Componente	Ilustración
1 x ESP8266	
1 x DHT22	
1x Módulo Relé	

1 x Protoboard



1 x LED



Resistencias Eléctricas  
1 x 330 $\Omega$



1 x Potenciometro de 10K  $\Omega$



1 x Foco





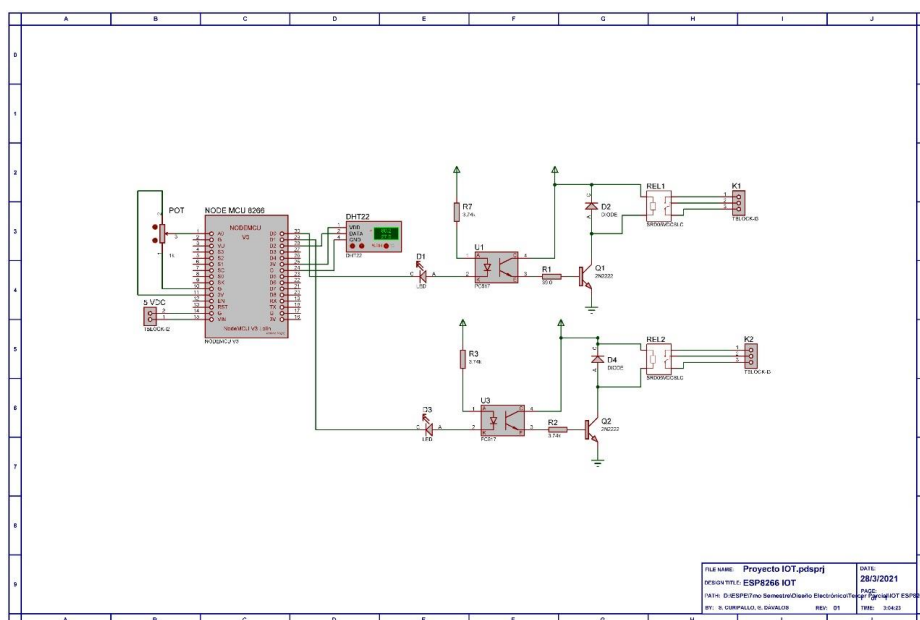
1 x Fuente de energía	
Varios Cables Jumper	

Tabla 1: Tabla de Materiales

## 5. Desarrollo

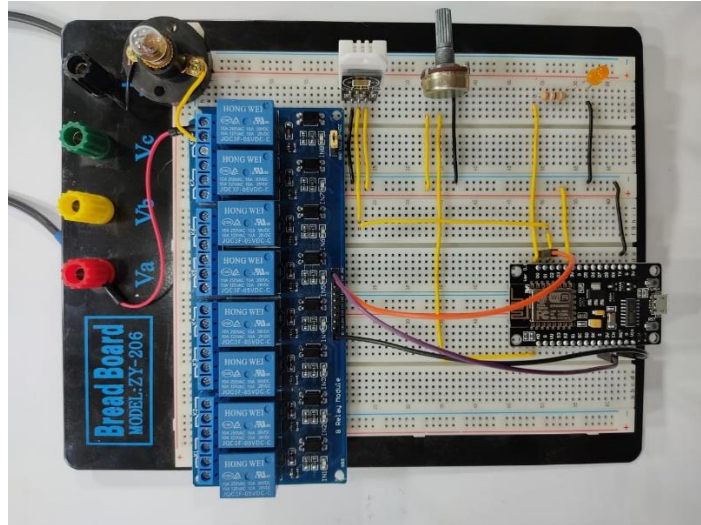
### Desarrollo del circuito

En el software Proteus, el siguiente circuito en el cuál se tiene al microcontrolador ESP8266, sus pines de alimentación están conectados a borneras. Además, tiene un potenciómetro y un sensor de temperatura-humedad conectados a los puertos de entradas analógicas. En las salidas digitales están conectados 2 relés con sus respectivas borneras para el futuro utilización de estos.



## Construcción del circuito

Con el diseño del circuito de procede a conectar los componentes, en este caso a una de las salidas del relé se conectó un foco de 12 v y la otra salida digital se ha conectado un led con su respectiva resistencia.



## Programación del controlador

Para programar esta tarjeta lo que se utiliza es la IDE de Arduino con las modificaciones para que nos permita receptar e identificar los puertos de la tarjeta.

Se presenta los códigos con los comentarios de la programación de cada una de las plataformas IOT.

## CÓDIGO UBIDOTS

```
#include <Ubidots.h> //Librería de Ubidots
#include <DHT.h> //Librería Sensor DHT22

#define DEVICE_LABEL "Node_MCU_8266" //Creamos una constante para almacenar
el nombre del dispositivo empleado en Ubidots

#define LAB_LED "led" //Se define la etiqueta que identificará al LED
#define LAB_TEMP "temperatura" //Se define la etiqueta que identificará a la
Temperatura
#define LAB_HUM "humedad" //Se define la etiqueta que identificará a la
Humedad
#define LAB_POT "potenciometro" //Se define la etiqueta que identificará al
Potenciometro
```

```
#define PIN_LED D0 //Se define el Pin del Microcontrolador donde se encuentra
conectado el LED
#define PIN_DHT D2 //Se define el Pin del Microcontrolador donde se encuentra
conectado el sensor DHT
```

```
const char* UBIDOTS_TOKEN = "BBFF-HSVxN87fAgwIE5WH0VJcxQimEMEst8";
//Token Cuenta de Ubidots
const char* WIFI_SSID = "NETLIFE-Creareco."; // SSID Red Wifi
const char* WIFI_PASS = "#JaqueDavalos1992#"; // Contraseña del Wifi
```

```
Ubidots ubidots (UBIDOTS_TOKEN, UBI_HTTP); //Creamos un objeto para enviar
datos a Ubidots
```

```
DHT dht (PIN_DHT, DHT22); //Creamos un objeto para manejar el sensor DHT22
seteando el tipo de sensor y el pin en el que se encuentra conectado
```

```
int poten, led, humedad; //Variable para almacenar el valor recibido del potenciómetro y
el sensor DH22, además de la variable para almacenar el estado del LED
```

```
float temperatura;
```

```
void setup() {
```

```
    Serial.begin(115200); //Configuro la velocidad de transferencia de datos para la
comunicacion serial
```

```
    ubidots.wifiConnect(WIFI_SSID, WIFI_PASS); //Llamamos al objeto ubidots con el
metodo wificonnect enviandole los datos de la red wifi
```

```
    Serial.println(""); //Mensaje mostrado en el monitor serie para verificar que el
microcontrolador se ha conectado exitosamente a la red Wifi
```

```
    Serial.println("*****");
```

```
    Serial.print("Conectado a la red Wi-fi: ");
```

```
    Serial.println(WiFi.SSID());
```

```
    Serial.print("IP: ");
```

```
    Serial.println(WiFi.localIP());
```

```
    Serial.print("MAC: ");
```

```
    Serial.println(WiFi.macAddress());
```

```
    Serial.println("*****");
```

```
dht.begin(); //Inicializamos el sensor DHT22
```

```
poten = 0; //Inicializamos la variable poten en cero
```

```
temperatura = 0; //Inicializamos la variable temperatura en cero
```

```
humedad = 0; //Inicializamos la variable humedad en cero
```

```
led = 0; //Inicializamos la variable led en cero
```



```
pinMode(PIN_LED,OUTPUT); //Indicamos que el pin al que se encuentra conectado
el LED como salida
digitalWrite(PIN_LED,led); //Escribimos en el pin donde esta conectado en led con el
estado de la variable led inicialmente en cero para que se encuentre apagado
```

```
}
```

```
void loop() { //Inicio del ciclo loop
  led = ubidots.get(DEVICE_LABEL, LAB_LED); //Obtencion de los datos de Ubidots
sobre la variable led
  poten = analogRead(A0); //Lectura del potenciómetro en el puerto Analógico 0 del
Node MCU ESP8266
  temperatura = dht.readTemperature(); //Leemos el valor de temperatura del sensor
DHT22 y lo almacenamos en la variable temperatura
  humedad = dht.readHumidity(); //Leemos el valor de humedad del sensor DHT22 y lo
almacenamos en la variable temperatura
  Serial.println("Temperatura="+String(temperatura)+",
Humedad="+String(humedad)+", Potenciómetro="+String(poten)); //Imprimimos en el
monitor serie los valores de temperatura, humedad y del potenciómetro

  digitalWrite(PIN_LED,led); //Escribimos sobre el pin del LED con la variable led que
almacena los datos enviados por Ubidots
```

```
  ubidots.add(LAB_POT, poten); //Llamamos al objeto ubidots con el metodo add para
enviar la etiqueta y el valor de la variable del potenciómetro a Ubidots
  ubidots.add(LAB_TEMP, temperatura); //Llamamos al objeto ubidots con el metodo
add para enviar la etiqueta y el valor de la variable temperatura a Ubidots
  ubidots.add(LAB_HUM, humedad); //Llamamos al objeto ubidots con el metodo add
para enviar la etiqueta y el valor de la variable thumedad a Ubidots
  ubidots.add(LAB_LED, led); //Llamamos al objeto ubidots con el metodo add para
enviar la etiqueta y el valor de la variable led a Ubidots
```

```
  bool bufferSent = false; //Declaracion de una variable booleana bufferSent para tener un
metodo de confirmacion del envio de los datos
  bufferSent = ubidots.send(DEVICE_LABEL); //Envia todos los datos que se hayan
empleado con el metodo add
```

```
  if (bufferSent){ //Condicion de envio de datos del buffer
    Serial.println("Valores enviados por el dispositivo"); //Muestra un mensaje de
confirmacion del envio de los datos
  }
  delay(1000); //Espera de un segundo
}
```

## CÓDIGO CAYENNE

```
#include <CayenneMQTTESP8266.h> //Incluimos la libreria de Cayenne
#include <DHT.h> //Incluimos la libreria del sensor DHT
#define CAYENNE_DEBUG //Definimos el Debug de Cayenne
#define CAYENNE_PRINT Serial //Definimos la comunicacion Serial de Cayenne
#define VIRTUAL_CHANNEL_FOCO 1 //Definimos el canal virtual para el foco
#define VIRTUAL_CHANNEL_LED 2 //Definimos el canal virtual para el LED
#define VIRTUAL_CHANNEL_POT 3 //Definimos el canal virtual para el
potenciometro
#define FOCO_PIN D1 //Definimos el pin donde se encuentra conectado el Foco
#define LED_PIN D0 //Definimos el pin donde se encuentra conectado el LED
#define POT_PIN A0 //Definimos el pin donde se encuentra conectado el
Potenciometro

char ssid[] = "NETLIFE-Creareco."; //Ingresamos el SSID de la red Wifi a la que se
desea conectar el microcontrolador
char password[] = "#JaqueDavalos1992#"; //Ingresamos el password de la red Wifi a la
que se desea conectar el microcontrolador

char username[] = "360d7000-814c-11eb-a2e4-b32ea624e442"; //Ingresamos el
username de nuestra cuenta de Cayenne
char mqtt_password[] = "937f1cb958494f4bc8392dbfac9c108ef618a0fa"; //Ingresamos
el password de nuestra cuenta de Cayenne
char client_id[] = "66ea97c0-814c-11eb-b767-3f1a8f1211ba"; //Ingresamos el ID de
cliente de nuestra cuenta de Cayenne

DHT dht(D2, DHT22); //Definimos el pin y el sensor de temperatura y humedad con el
que vamos a trabajar en este caso DHT en el pin D2

void setup() { //Inicio del setup
    Serial.begin(9600); //Configuramos la velocidad de la comunicacion serial a 9600
bauds
    Cayenne.begin(username, mqtt_password, client_id, ssid, password); //Creamos un
objeto para almacenar las credenciales de autenticacion de nuestra cuenta de Cayenne
    pinMode(FOCO_PIN, OUTPUT); //Definimos el pin del Foco como salida
    pinMode(LED_PIN, OUTPUT); //Definimos el pin del LED como salida
    pinMode(POT_PIN, INPUT); //Definimos el pin del Potenciometro como salida
}

void loop() { //Inicio del ciclo loop
    Cayenne.loop(); //Inicializamos el objeto Cayenne en ciclo loop
    float temp = (dht.readTemperature(true)-32)/1.8000; //Obtención y escalamiento de la
variable temperatura del DHT22
    float hum = dht.readHumidity(); //Obtención de la variable humedad del DHT22
    Cayenne.virtualWrite(1, temp, TYPE_TEMPERATURE,
UNIT_CELSIUS); //Escribimos sobre el canal virtual 1 la variable temperatura del
DHT22
    Cayenne.virtualWrite(2, hum, TYPE_RELATIVE_HUMIDITY, UNIT_PERCENT);
//Escribimos sobre el canal virtual 1 la variable temperatura del DHT22
```

```

Cayenne.virtualWrite(VIRTUAL_CHANNEL_POT,      analogRead(POT_PIN),
"analog_sensor", "null"); //Escribimos sobre el canal virtual del potenciómetro la lectura
del potenciómetro obtenida mediante analogRead
Serial.println(analogRead(POT_PIN)); //Escribimos en el monitor serie el valor
obtenido del potenciómetro
}

```

```

CAYENNE_IN(VIRTUAL_CHANNEL_FOCO) //Creamos una subrutina para el
accionar del foco
{
    int value = getValue.asInt(); //Creamos una variable value para almacenar el valor que
proviene de Cayenne mediante el boton para accionar el Foco
    CAYENNE_LOG("Channel %d, pin %d, value %d", VIRTUAL_CHANNEL_FOCO,
FOCO_PIN, value); //Nos logeamos en el canal virtual del foco con su pin y el valor
obtenido previamente

    if (value == 0) //Condicion si el valor es igual a cero
    {digitalWrite(FOCO_PIN, HIGH); //El relé al que se encuentra conectado el foco se
apaga
    }
    else if (value == 1){ //Condicion si el valor es igual a uno
        digitalWrite(FOCO_PIN, LOW); //El relé al que se encuentra conectado el foco se
enciende
    }
}

```

```

CAYENNE_IN(VIRTUAL_CHANNEL_LED) //Creamos una subrutina para el
accionar del LED
{
    int value1 = getValue.asInt(); //Creamos una variable value1 para almacenar el estado
de el boton led ue proviene de Cayenne mediante el boton para accionar el LED
    CAYENNE_LOG("Channel %d, pin %d, value %d", VIRTUAL_CHANNEL_LED,
LED_PIN, value1); //Nos logeamos en el canal virtual del LED con su pin y el valor
obtenido previamente
    digitalWrite(LED_PIN, value1); //Escribimos sobre el pin del microcontrolador el valor
obtenido previamente referente al LED
}

```

```

CAYENNE_OUT(0) //Creamos una subrutina para el envio de los datos a Cayenne
{
    CAYENNE_LOG("Send data for Virtual Channel 0"); //Se muestra un mensaje si la
comunicacion y envio de datos ha sido exitoso
    Cayenne.virtualWrite(0, millis() / 1000); //Envio de datos a Cayenne por el canal virtual
cero en tiempo de segundos
}

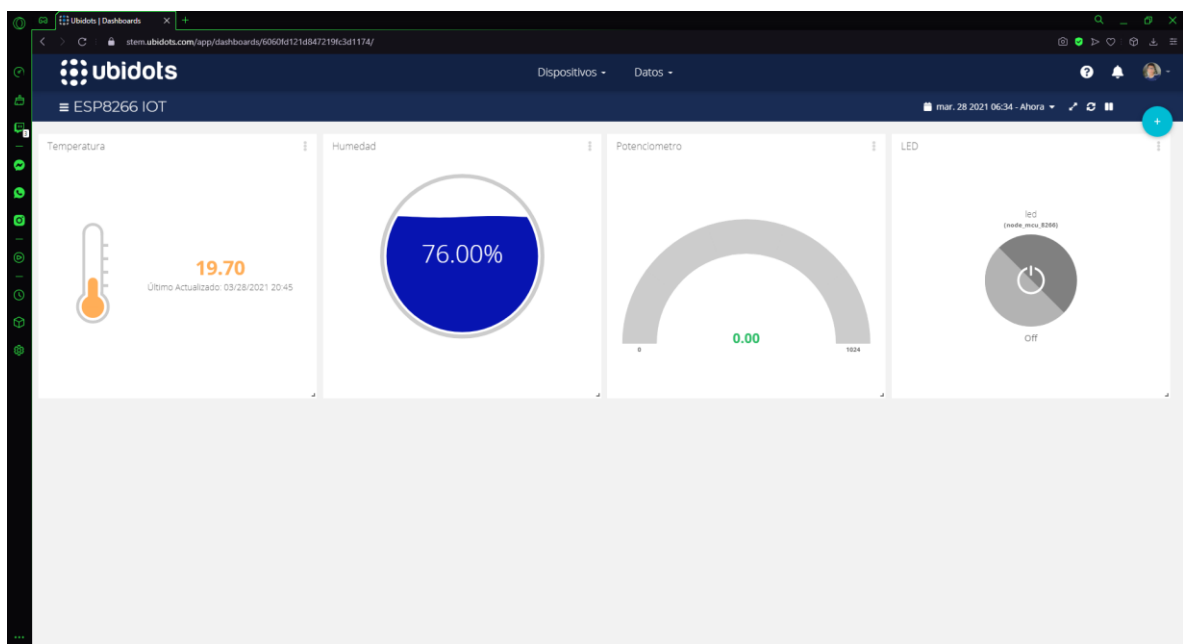
```

## Diseño del panel de control en la plataforma IOT

Dentro de las plataformas IOT se procede a diseñar el panel de control con los componentes necesarios, en este caso un botón para el encendido del led o relé, un indicador numérico tanto para el valor de temperatura y otro para la humedad. Y un medidor o gauge para el valor del potenciómetro.

### Ubidots

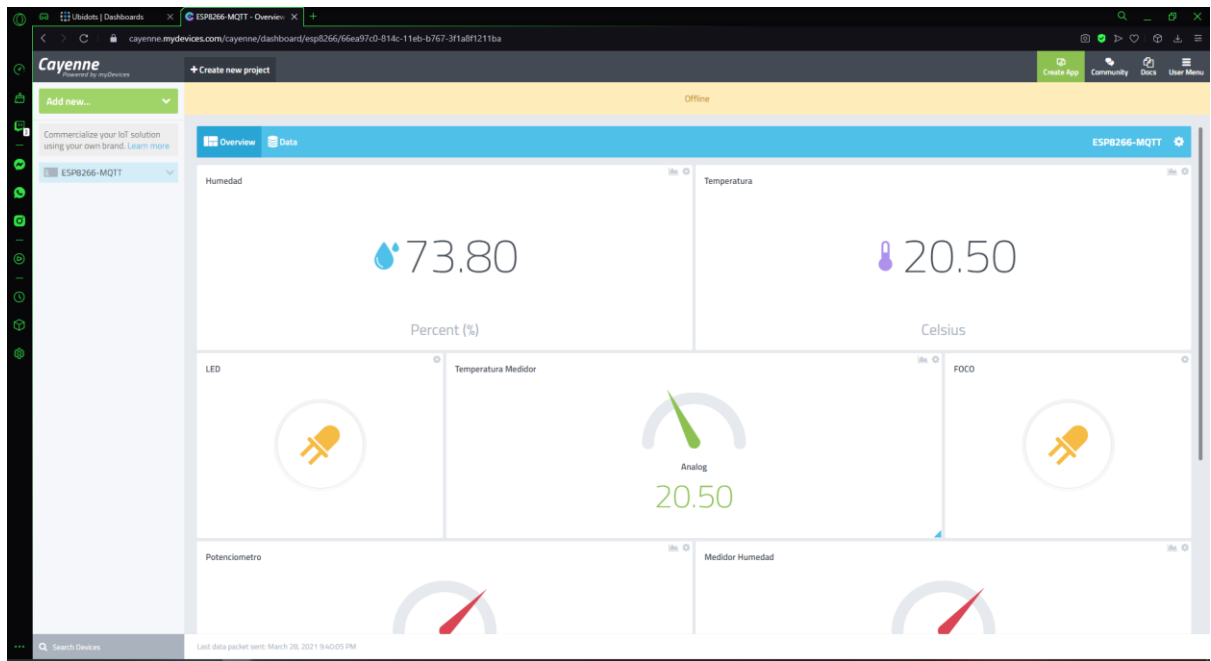
Dentro de la dashboard se puede agregar los diferentes medidores e indicadores, tanto para la temperatura como para la humedad, además la entrada para el potenciómetro y la salida de activación de un led.



### Cayenne

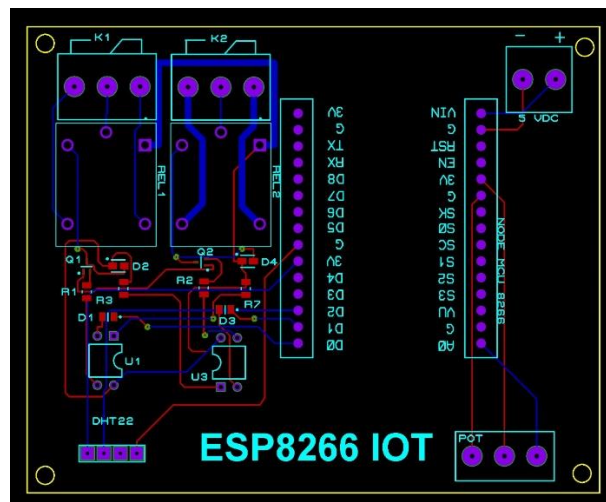
Se puede agregar una gran cantidad de indicadores y una gama específica para los diferentes sensores que se encuentran en el mercado, facilitando el control y visualización de las señales.

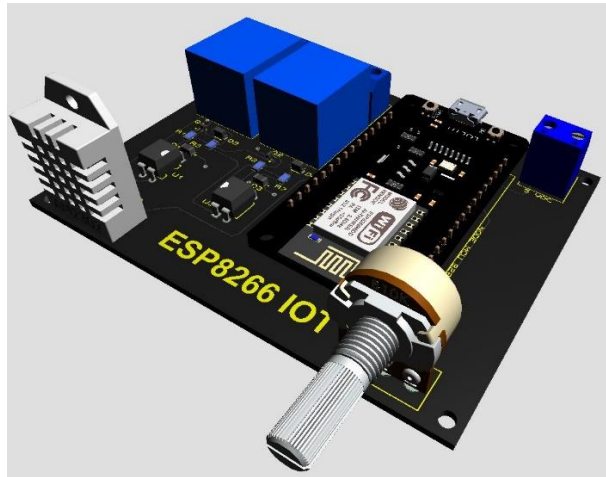
Posee una interfaz simple para el manejo y como complemento, una aplicación para móviles.



## Diseño de la placa PCB

Una vez que se comprobó el funcionamiento del circuito con la plataforma IOT, se procedió a diseñar la placa PCB, con las diferentes pistas y componentes de su posterior elaboración.





## 6. Análisis de costos

Descripción	Precio
Componentes	
<b>ESP8266</b>	\$8.50
<b>Sensor Humedad Temperatura DHT 12</b>	\$11.00
<b>Relé x2</b>	\$4.00
<b>Potenciómetro</b>	\$1.00
<b>Tiempo de diseño de la PCB</b>	\$40.00
<b>Total</b>	\$64,5

## 7. Conclusiones

- Las plataformas IOT nos permiten desarrollar una infinidad de proyectos con distintas finalidades.
- Después de probar y testear las 2 plataformas IOT, se visualizó que la plataforma Cayenne es superior a Ubidots, ya que en Cayenne podemos manipular 2 variables de salida mientras que Ubidots se limita a 1, otra razón es la rapidez al enviar o recibir un comando.
- El mayor problema que tuvimos con la plataforma Ubidots es la limitación en la cantidad de envío de datos, ya que solo nos permite enviar 4000 unidades de datos por día, una vez superado este valor, el control se desactiva hasta el día siguiente.

## **8. Enlace**

### **Enlace al Video de Funcionamiento**

[https://drive.google.com/file/d/1zt6BS\\_1whufqAr6cgTKTOUaJk\\_pCWVZ4/view?usp=sharing](https://drive.google.com/file/d/1zt6BS_1whufqAr6cgTKTOUaJk_pCWVZ4/view?usp=sharing)

## **9. Bibliografía**

[1]: Community for Businesses in Latin America and the Caribbean | ConnectAmericas. (2021). Retrieved 1 April 2021, from <https://connectamericas.com/es/company/ubidots>

[2]: ESP8266 - Wikipedia, la enciclopedia libre. (2021). Retrieved 1 April 2021, from <https://es.wikipedia.org/wiki/ESP8266>

[3]: humedad, T., (AM2302), S., & (AM2302), S. (2021). Sensor de temperatura y humedad relativa DHT22 (AM2302). Retrieved 1 April 2021, from <https://naylampmechatronics.com/sensores-temperatura-y-humedad/58-sensor-de-temperatura-y-humedad-relativa-dht22-am2302.html>

[4]: Relé - Wikipedia, la enciclopedia libre. (2021). Retrieved 1 April 2021, from <https://es.wikipedia.org/wiki/Rel%C3%A9>

[5]: soy?, ¿., & [GRATIS], C. (2021). Cayenne myDevices y Arduino para monitorizar sensores del IoT. Retrieved 1 April 2021, from [https://programarfacil.com/blog/arduino-blog/cayenne-mydevices-arduino-sensores-iot/#Que\\_es\\_Cayenne\\_My\\_Devices](https://programarfacil.com/blog/arduino-blog/cayenne-mydevices-arduino-sensores-iot/#Que_es_Cayenne_My_Devices)