
ISPC INSTITUTO SUPERIOR POLITÉCNICO CÓRDOBA

Dirección General de EDUCACIÓN TÉCNICA Y FORMACIÓN PROFESIONAL · Ministerio de EDUCACIÓN · CÓRDOBA entre todos

Profesor: Gonzalo Vera

**Prueba la Comunicación entre el ESP32 y la API**

**Explicación detallada del código**

1. Importar las librerías necesarias

```
#include <WiFi.h>
#include <HTTPClient.h>
```

Estas librerías son fundamentales para gestionar la conexión Wi-Fi y para realizar solicitudes HTTP a la API.

2. Definir las credenciales Wi-Fi

```
const char* ssid = "tu_SSID";
const char* password = "tu_PASSWORD";
```

Estas variables almacenan el nombre y la contraseña de tu red Wi-Fi. Debes reemplazar "tu_SSID" y "tu_PASSWORD" por tus credenciales reales de red.

3. Definir la URL del servidor:

```
const char* serverName = "https://api.gonaiot.com/plata/datos_dispositivos";
```

Aquí defines la URL de la API donde se enviarán los datos de prueba.

4. Variables de prueba de temperatura y humedad:

```
float temperatura = 24.5;
float humedad = 60.2;
```

Estos son valores ficticios para simular los datos que se enviarían a la API. En tu proyecto real, estos valores vendrán de sensores conectados al ESP32.

5. Función setup(): Esta función se ejecuta una vez cuando se enciende el ESP32:

```
void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Intentando conectarse...");
  }
  Serial.println("Conectado a la red WiFi");
}
```

- Inicializa la comunicación serie para monitorear el estado del ESP32.
- Establece la conexión con la red Wi-Fi usando las credenciales definidas.
- Si la conexión Wi-Fi no se ha establecido, imprime un mensaje hasta que se conecte.

6. Función loop(): Esta función se ejecuta continuamente, y aquí es donde se realiza la solicitud POST:

```
void loop() {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;
    http.begin(serverName);
    http.addHeader("Content-Type", "application/json");
    String httpRequestData = "{\"temperatura\":" + String(temperatura) + ", \"humedad\
    int httpResponseCode = http.POST(httpRequestData);
    if (httpResponseCode > 0) {
      String response = http.getString();
      Serial.println(httpResponseCode);
      Serial.println(response);
    } else {
      Serial.print("Error en la solicitud POST: ");
      Serial.println(httpResponseCode);
    }
    http.end();
  } else {
    Serial.println("Error en la conexión Wi-Fi");
  }
  delay(10000);
}
```

- Conexión Wi-Fi: Antes de hacer una solicitud HTTP, se comprueba que el ESP32 esté conectado a la red.

- Solicitud POST: Se construye la solicitud HTTP POST para enviar los datos de temperatura y humedad en formato JSON a la API.
- Manejo de Errores: Si la solicitud falla, se imprime un mensaje de error con el código correspondiente.

5. Compilar y cargar el código en el ESP32