



C4.2 Programación Microcontrolador NodeMCU ESP32

Comunicación por medio de la conexión Wi-Fi



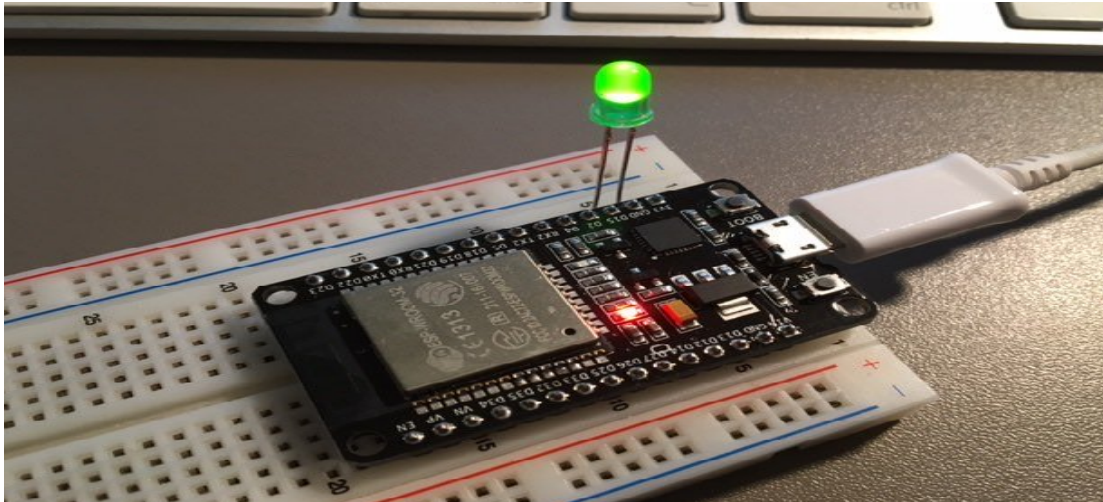
Instrucciones

- De acuerdo con la información presentada por el asesor referente al tema, desarrollar lo que se indica dentro del apartado siguiente.
- Toda actividad o reto se deberá realizar utilizando el estilo **Markdown con extension .md** y el entorno de desarrollo VSCode, debiendo ser elaborado como un documento **single page**, es decir si el documento cuanta con imágenes, enlaces o cualquier documento externo debe ser accedido desde etiquetas y enlaces.
- Es requisito que el archivo .md contenga una etiqueta del enlace al repositorio de su documento en Github, por ejemplo **Enlace a mi GitHub**
- Al concluir el reto el reto se deberá subir a github el archivo .md creado.
- Desde el archivo .md se debe exportar un archivo .pdf con la nomenclatura **C4.2_NombreAlumno_Equipo.pdf**, el cual deberá subirse a classroom dentro de su apartado correspondiente, para que sirva como evidencia de su entrega; siendo esta plataforma **oficial** aquí se recibirá la calificación de su actividad por individual.
- Considerando que el archivo .pdf, fue obtenido desde archivo .md, ambos deben ser idénticos y mostrar el mismo contenido.
- Su repositorio además de que debe contar con un archivo **readme.md** dentro de su directorio raíz, con la información como datos del estudiante, equipo de trabajo, materia, carrera, datos del asesor, e incluso logotipo o imágenes, debe tener un apartado de contenidos o índice, los cuales realmente son ligas o **enlaces a sus documentos .md**, *evite utilizar texto* para indicar enlaces internos o externo.
- Se propone una estructura tal como esta indicada abajo, sin embargo puede utilizarse cualquier otra que le apoye para organizar su repositorio.

```
| readme.md
| | blog
| | | C4.1_TituloActividad.md
| | | C4.2_TituloActividad.md
| | | C4.3_TituloActividad.md
| | | C4.4_TituloActividad.md
| | | C4.5_TituloActividad.md
| | img
| | docs
| | | A4.1_TituloActividad.md
| | | A4.2_TituloActividad.md
```

Desarrollo

1. Basado en el siguiente circuito, ensamblarlo, utilizando los elementos electrónicos observados.



Fuente de consulta: [Random Nerd Tutorials](#)

2. Analice y apóyese del programa que se muestra a continuación para elaborar el reto.

```
/*
  WiFi Web Server Simple
  */

#include <WiFi.h>
#include <WebServer.h>

const char* ssid = "<identificador>";
const char* password = "<password>";

WebServer server(80); // Object of WebServer(HTTP port, 80 is default)

void setup() {
  Serial.begin(115200);
  Serial.println("Try Connecting to ");
  Serial.println(ssid);

  // Connect to your wi-fi modem
  WiFi.begin(ssid, password);

  // Check wi-fi is connected to wi-fi network
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected successfully");
  Serial.print("Got IP: ");
  Serial.println(WiFi.localIP()); //Show ESP32 IP on serial
```

```

server.on("/", handle_root);

server.begin();
Serial.println("HTTP server started");
delay(100);
}

void loop() {
  server.handleClient();
}

// HTML & CSS contents which display on web server
String HTML = "<!DOCTYPE html>\n
<html>\n
<body>\n
<h1>Mi Primer Servidor Web with ESP32 - Station Mode &#128522;</h1>\n
</body>\n
</html>";

// Handle root url (/)
void handle_root() {
  server.send(200, "text/html", HTML);
}

```

3. Pruebe y observe los resultados obtenidos explicándolos en esta sección.

Segun los resultados que obtuve es que realizamos dentro de el mismo arduino una pagina o mejor dicho podemos mandar un web server y mandamos el html me parecio muy interesante la manera en la que funciona y no tuve mucho problema en lograrlo

4. Al programa anterior agregue las instrucciones necesarias para que se despliegue en la interface un botón que permita encender y apagar un Led tal como se muestra en la figura 1.

```

#include <WiFi.h>

// la red y el pasword
const char* ssid = "INFINITUM4524_2.4";
const char* password = "Peepeepoopoocheck77.";

WiFiServer server(80);
String header;
String output26State = "off";
const int output26 = 26;

// Current time
unsigned long currentTime = millis();
unsigned long previousTime = 0;
const long timeoutTime = 2000;

void setup() {

```

```

Serial.begin(115200);
pinMode(output26, OUTPUT);
digitalWrite(output26, LOW);

// Nos conetamos a la res wiffi
Serial.print("Connecting to ");
Serial.println(ssid);
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}
// Imprime la red y si se logro conectar
Serial.println("");
Serial.println("WiFi connected.");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
server.begin();
}

void loop(){
    WiFiClient client = server.available(); // Escuchando al cliente

    if (client) { // Si nos logramos conectar
        currentTime = millis();
        previousTime = currentTime;
        Serial.println("New Client."); // Imprimimos si ya entraron al
        puerto
        String currentLine = ""; // Lo que mandara el cliente
        while (client.connected() && currentTime - previousTime <= timeoutTime)
        { // Ciclo para el cliente cpnectado
            currentTime = millis();
            if (client.available()) { // Si esta leyendo con el
            cliente
                char c = client.read(); // lee y despues
                Serial.write(c); // escribe en el serial
                header += c;
                if (c == '\n') { // Si es una nueva linea de
                caracteres
                    if (currentLine.length() == 0) {
                        client.println("HTTP/1.1 200 OK");
                        client.println("Content-type:text/html");
                        client.println("Connection: close");
                        client.println();

                        // Apagar y encender la luz
                        if (header.indexOf("GET /26/on") >= 0) {
                            Serial.println("GPIO 26 on");
                            output26State = "on";
                            digitalWrite(output26, HIGH);
                        } else if (header.indexOf("GET /26/off") >= 0) {
                            Serial.println("GPIO 26 off");
                            output26State = "off";
                            digitalWrite(output26, LOW);

```

```

    }

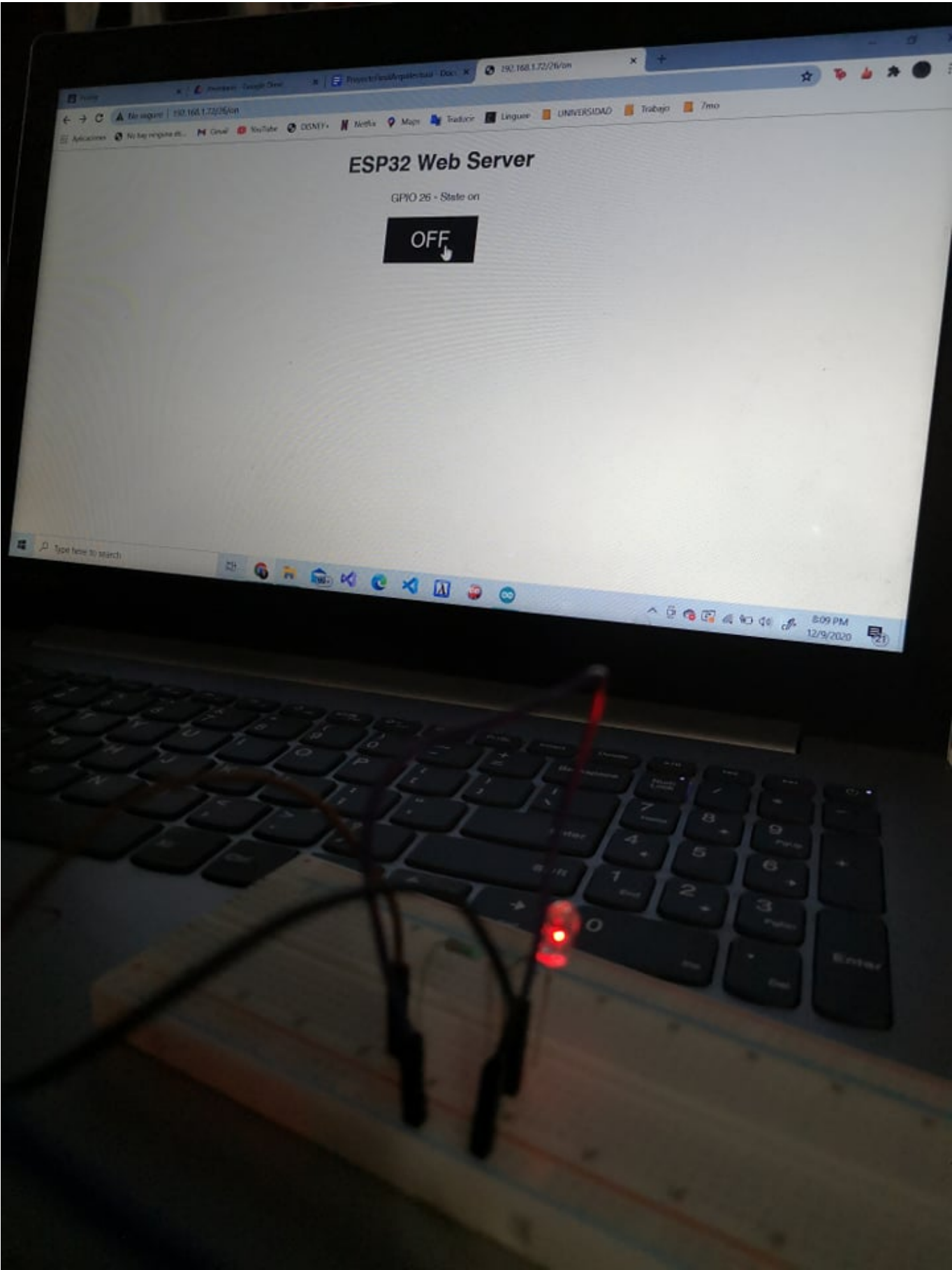
    // HTM de la pagina
    client.println("<!DOCTYPE html><html>");
    client.println("<head><meta name=\"viewport\" "
content=\"width=device-width, initial-scale=1\">");
    client.println("<link rel=\"icon\" href=\"data:,\">");
    //CSS que usan los botones y la pagina en general
    client.println("<style>html { font-family: Helvetica; display:
inline-block; margin: 0px auto; text-align: center;}");
    client.println(".button { background-color: #4CAF50; border:
none; color: white; padding: 16px 40px;");
    client.println("text-decoration: none; font-size: 30px; margin:
2px; cursor: pointer;}");
    client.println(".button2 {background-color: #555555;}</style>
</head>");
    client.println("<body><h1>ESP32 Web Server</h1>");

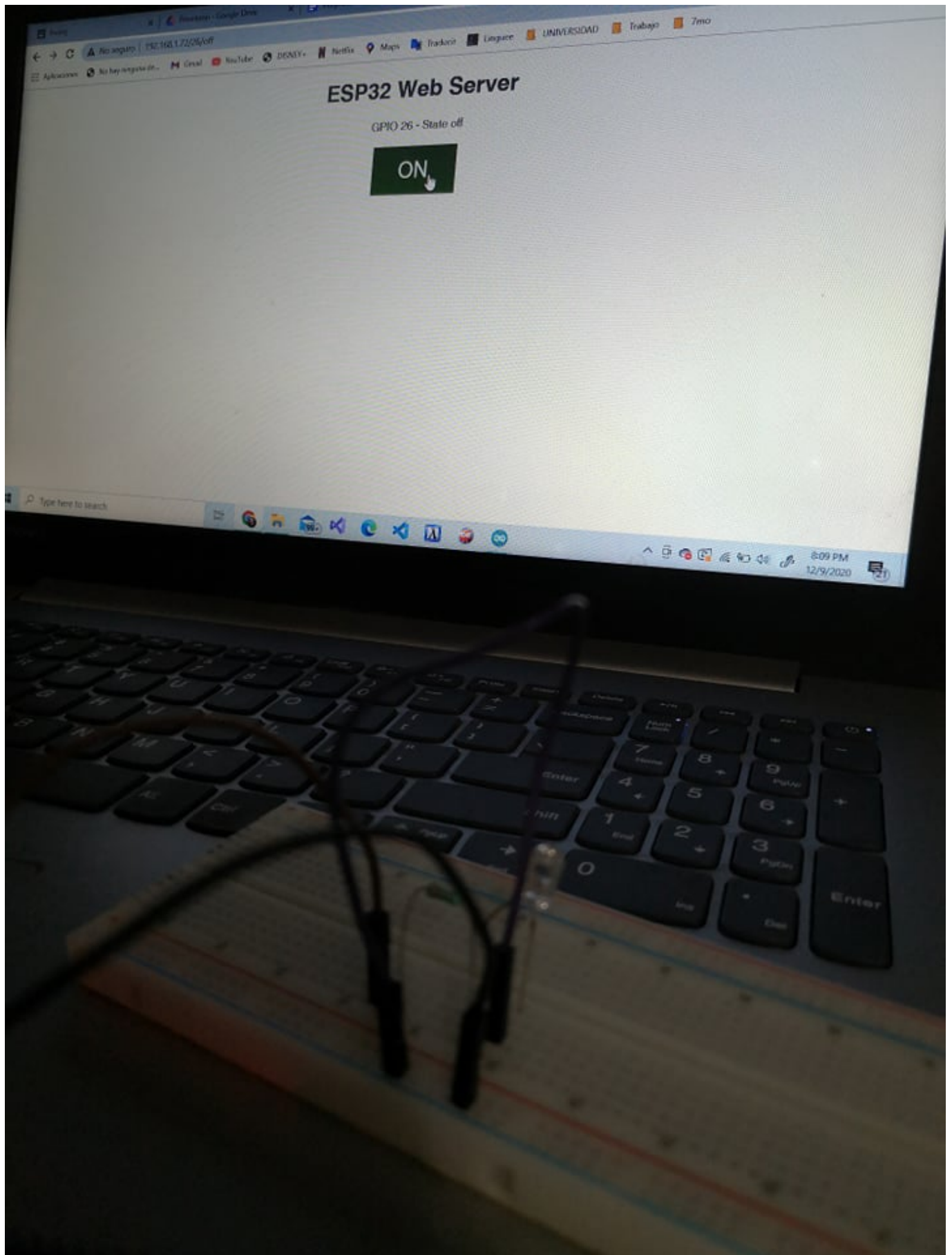
    // Para los botones off/on o mejor dicho el boton
    client.println("<p>GPIO 26 - State " + output26State + "</p>");
    if (output26State=="off") {
        client.println("<p><a href=\"/26/on\"><button
class=\"button\">ON</button></a></p>");
    } else {
        client.println("<p><a href=\"/26/off\"><button class=\"button
button2\">OFF</button></a></p>");
    }
    client.println("</body></html>");

    client.println();
    // Break de nuestro ciclo
    break;
} else { //
    currentLine = "";
}
} else if (c != '\r') {
    currentLine += c;
}
}
}
// Limpiar la variable
header = "";
// Cerrar la conexion
client.stop();
Serial.println("Client disconnected.");
Serial.println("");
}
}

```

5. Inserte aquí las imágenes que considere como evidencias para demostrar el resultado obtenido.





Video

[Ir a repositorio](#)



Rubrica

Criterios	Descripción	Puntaje
Instrucciones	Se cumple con cada uno de los puntos indicados dentro del apartado Instrucciones?	20
Desarrollo	Se respondió a cada uno de los puntos solicitados dentro del desarrollo de la actividad?	80

[Ir a readme](#)