SERVIDOR WEB CON BOTONES ESTILIZADOS, BARRA Y TARJETA

ARCHIVO main.cpp

```
#include <Arduino.h>
#include <ESP8266WiFi.h>
#include <ESPAsyncTCP.h>
#include <ESPAsyncWebServer.h>
#include <LittleFS.h>
const char* ssid = "MEGACABLE-F231F";
const char* password = "Ersdsw443";

// Crea el objeto AsyncWebServer en el puerto 80
AsyncWebServer server(80);
// Define el pin del LED
const int ledPin = 2;
// Variable cadena para almacenar el estado del LED
String EstadoDeLED;
// Inicializa LitleFS
void initFS() {
if (!LittleFS.begin()) {
Serial.println("Un error ha ocurrido mientras se montaba LitleFS");
}
Serial.println("LitleFS montado exitosamente");
}
void initWiFi() {
WiFi.mode(WIFI_STA);
WiFi.begin(ssid, password);
Serial.print("Conectando a la red WiFi ..");
while (WiFi.status() != WL_CONNECTED) {
Serial.print('.');
delay(1000);
}
Serial.println(WiFi.localIP());
}
String processor(const String& var)
{
//La función procesador () reemplaza los marcadores de posición
// en el texto HTML con los valores actuales.
/* Cuando se solicita la página web, el ESP8266 comprueba si el texto HTML
 tiene marcadores de posición. Si encuentra el marcador de posición  %ESTADO%,
 lea el estado actual de GPIO con digitalRead (ledPin) y configure la variable
 de valor EstadoDeLED en consecuencia. La función devuelve el estado actual de GPIO
 como una variable de cadena. */
if(var == "ESTADO") {
if(digitalRead(ledPin)) {
EstadoDeLED = "APAGADO";
}
else {
EstadoDeLED = "ENCENDIDO";
```

```
}
return EstadoDeLED;
}
return String();
}
void setup() {
Serial.begin(115200);
initWiFi();
initFS();
// Establece GPIO2 como una SALIDA
pinMode(ledPin, OUTPUT);
server.on("/", HTTP_GET, [](AsyncWebServerRequest *request){
request->send(LittleFS, "/index.html", "text/html",false, processor);
});
/*
Cuando recibe esa solicitud, envía el texto HTML guardado en el archivo index.html
para construir la página web. También necesita pasar la función de procesador,
que reemplaza a todos los marcadores de posición en el texto HTML con los valores
correctos.
El primer argumento de la función send () es el sistema de archivos donde se guardan
los archivos, en este caso se guarda en SPIFFS (o LittleFS para el ESP8266).
El segundo argumento es la ruta donde se encuentra el archivo.
El tercer argumento se refiere al tipo de contenido (texto HTML).
El cuarto argumento significa descarga = falso.
Finalmente, el último argumento es la función del procesador.
Cuando el archivo HTML se carga en su navegador, solicitará el archivo CSS y
el favicon. Estos son archivos estáticos guardados en el mismo directorio
(SPIFFS o LittleFS). Entonces, podemos simplemente agregar la siguiente línea
para servir archivos en un directorio cuando lo solicite
la URL raíz. Sirve los archivos CSS y favicon automáticamente.
*/
server.serveStatic("/", LittleFS, "/");
// Procedimiento para establecer el GPIO 2 en estado ALTO (lógica invertida en ESP8266)
server.on("/encendido", HTTP_GET, [](AsyncWebServerRequest *request)
{
digitalWrite(ledPin, LOW); // lógica invertida
request->send(LittleFS, "/index.html", "text/html", false, processor);
}
);
// Procedimiento para establecer el GPIO 2 en estado BAJO (lógica invertida en ESP8266)
server.on("/apagado", HTTP_GET, [](AsyncWebServerRequest *request)
{
digitalWrite(ledPin, HIGH); // lógica invertida
request->send(LittleFS, "/index.html", "text/html", false, processor);
}
);
server.begin();
}
void loop() {   }
```

ARCHIVO index.html

```
<!DOCTYPE html>
<html>
<head>
<title>Servidor Web con botones</title>
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" type="text/css" href="style.css">
<link rel="icon" type="image/png" href="favicon.png">

</head>
<body>
<div class="barra">
<h1>WEB SERVER CON BOTONES sin card title</h1>
</div>
<div class="contenido">
<div class="cuadriculaentarjeta">
<div class="tarjeta">
<p> GPIO 2 del ESP8266 en Wemos D1</p>
<p>
<a href="encendido"><button class="button-on">ENCENDER</button></a>
<a href="apagado"><button class="button-off">APAGAR</button></a>
</p>
<p class="estado">Estado: %ESTADO%</p>
</div>
</div>
</div>
</body>
</html>
```

ARCHIVO style.css

```
html {
font-family: Arial, Helvetica, sans-serif;
text-align: center;
}
h1 {
font-size: 1.8rem;
color: white;
/*
La unidad rem para el tamaño de fuente es relativa al tamaño de fuente
del elemento raíz. Esto significa que el tamaño de fuente es 1,8 veces
mayor que el tamaño predeterminado utilizado por el navegador.
*/
}
.barra {
/*
El div barra contiene el texto  WEB SERVER CON BOTONES.
es un poco más grande que la fuente predeterminada del título 1.
*/
overflow: hidden;
background-color: #204a8f;
/*
Se estiliza el color de fondo de barra usando la propiedad background-color.
Usted puede elegir cualquier color de fondo, estamos usando # 0A1128.
Además, configure el desbordamiento propiedad oculta de esta manera.
*/

}
body {
margin: 0;
}
.contenido {
    /*
Aquí se agrega algo de relleno al div de contenido, para que
la cuadrícula de la tarjeta no se contraiga a los márgenes
del cuerpo.
    */
padding: 50px;
}
.cuadriculaentarjeta {

    max-width: 800px;
    margin: 0 auto;
    display: grid;
    grid-gap: 2rem;
    grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));
 /* La propiedad de ancho máximo max-width, como su nombre indica, establece
  el ancho   máximo de la cuadrícula.
  Para colocar todo alineado en el centro, establezca el margen de mygrid en
```

```
  0 automático.
  Al establecer la propiedad de margen en 0, se centra automáticamente
  la división dentro de su contenedor principal. Es lo mismo que establecer
  los márgenes superior e inferior en cero y los márgenes izquierdo y derecho
  en automático. El navegador distribuye automáticamente la cantidad correcta
  de margen a cada lado.

La propiedad de visualización establece el elemento mygrid como un contenedor
de cuadrícula.
La grid-gap define el espacio entre cada celda de la cuadrícula (cuadro);
en este caso, solo tenemos una celda de la cuadrícula.
La propiedad grid-template-column establece el ancho de las columnas.
La repetición significa que aplicará la misma configuración a todas las columnas.
El minmax (200px, 1fr) significa que el ancho de las columnas se ajusta al tamaño
de la ventana del navegador a un mínimo de 200px.
El 1fr le permite establecer el tamaño de una pista como una fracción del espacio
 libre del contenedor de la cuadrícula. Por ejemplo, si tuviera tres tarjetas, establecería cada
elemento en un tercio del ancho del contenedor de la cuadrícula. Por el momento, es posible que
esto no se note porque solo estamos usando una celda de la cuadrícula; comprenderá mejor cómo
esto funciona cuando agregamos más tarjetas en proyectos futuros.
    */
    }
 .tarjeta    { /*Tarjeta donde se incluyen los botones*/
    background-color: rgb(102, 160, 145);
    box-shadow: 2px 2px 12px 1px rgba(140, 140, 140, 0.459);
/*
El color de fondo de la tarjeta se establece en blanco,
pero puede usar cualquier otro color.
La propiedad box-shadow adjunta una o más sombras al elemento.
Esto es lo que significa cada valor de izquierda a derecha:
• offset-x: distancia horizontal;
• offset-y: distancia vertical;
• radio de desenfoque: cuanto mayor sea este valor, mayor será el desenfoque,
por lo que la sombra se vuelve más grande y más clara;
• spread-radius: los valores positivos harán que la sombra se expanda y crezca,
los valores negativos harán que la sombra se encoja.
• color: estamos usando el código de color RGBA. RGBA significa alfa rojo,
verde, azul. La alpha corresponde a la opacidad del color.
*/
    }
 .estado { /*aplica estilo al párrafo que muestra el estado:
        el color y el tamaño de fuente propiedades. */
    font-size: 1.2rem;
    color: #2293b3;
    }
  Button { /*para agrandar el tamaño de los botones */
    border: none;
    color: #FEFCFB;
    padding: 15px 32px;
```

```
    text-align: center;
    text-decoration: none;
    font-size: 16px;
    width: 200px;
    border-radius: 4px;
    transition-duration: 0.4s;
    }

    .button-on {  /* color del botón de encendido */
    background-color: #034078;
    }
    .button-off { /* color del botón de apagado */
        background-color: #858585;
      }
```

ARCHIVO platformio.ini

```
[env:d1]
platform = espressif8266
board = d1
framework = arduino
monitor_speed = 115200
board_build.filesystem = littlefs
lib_deps = ottowinter/ESPAsyncWebServer-esphome@^3.0.0
```

Al seguir los pasos de grabación co archives separados debe aparecer:

```
Conectando a la red WiFi .....192.168.1.145
LitleFS montado exitosamente
```