```
#include <WiFi.h>
#include <AsyncMqttClient.h>
#include <Ticker.h>

// Configuración de la red WiFi
const char* ssid = "xxxxxxxxxxx";
const char* password = "xxxxxxxxxx";


// Configuración del servidor MQTT
const char* mqttServer = "broker.emqx.io";
const int mqttPort = 1883;

// Tema MQTT para controlar el LED
const char* ServoTopic = "LLENAR_TANQUE"; // Este tema recibe los
mensajes para controlar el LED

// Pin donde está conectado el LED
const int ELECTRO_BOMBA = 4;
int value=0;
AsyncMqttClient mqttClient;

// Conexión WiFi
void setupWifi() {
 WiFi.begin(ssid, password);
 while (WiFi.status() != WL_CONNECTED) {
   delay(500);
   Serial.print(".");
 }
 Serial.println("\nConectado a la red WiFi");
}

// Evento al conectar con el servidor MQTT
void onMqttConnect(bool sessionPresent) {
 Serial.println("Conectado al servidor MQTT");

 // Suscribirse al tema del SERVO
 mqttClient.subscribe(ServoTopic, 2);
```

```
}

// Evento al desconectar del servidor MQTT
void onMqttDisconnect(AsyncMqttClientDisconnectReason reason) {
 Serial.println("Desconectado del servidor MQTT. Intentando reconectar...");
 delay(5000);  // Espera 5 segundos antes de reconectar
 mqttClient.connect();
}

// Evento para manejar los mensajes recibidos
void onMqttMessage(char* topic, char* payload,
AsyncMqttClientMessageProperties properties, size_t len, size_t index, size_t
total) {
 Serial.print("Mensaje recibido en el tema: ");
 Serial.println(topic);

 // Convertir el payload en un String para compararlo fácilmente
 String message;
 for (size_t i = 0; i < len; i++) {
   message += (char)payload[i];
 }

 Serial.print("Contenido del mensaje: ");
 Serial.println(message);

 // Encender o apagar Electrobomba según el mensaje recibido
(LLENAR_TANQUE)
 if (String(topic) == ServoTopic){
   if (message == "OFF") {
       digitalWrite(ELECTRO_BOMBA, LOW);
   } else if (message == "ON") {
     digitalWrite(ELECTRO_BOMBA, HIGH);
```

```
  } else {
    Serial.println("Mensaje no reconocido"); // En caso de mensaje
desconocido
  }
 }



 if (String(topic) == ServoTopic){
value=ServoTopic;
  }


}




void setup() {
 Serial.begin(115200);

//CambiarContador.attach(2, FuncionContar);
 // Configuración del pin del LED como salida
 pinMode(ELECTRO_BOMBA, OUTPUT);

 // Configurar la conexión WiFi
 setupWifi();

 // Configuración del cliente MQTT y eventos
 mqttClient.onConnect(onMqttConnect);
 mqttClient.onDisconnect(onMqttDisconnect);
 mqttClient.onMessage(onMqttMessage);
 mqttClient.setServer(mqttServer, mqttPort);

 // Conectar al servidor MQTT
 mqttClient.connect();
}
```

```
void loop() {
 // No se necesita código en el loop debido a que usamos eventos asíncronos
}
```