UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL VILLA MARÍA
Avenida Universidad 450 – 5900 Villa María (Cba.)

Informe Trabajo Práctico Arduino

Asignatura: Teoría de control

Año: 2023

Fecha: 23/11/23

Docentes:
- Ing. Nicolas Oliva
- Ing. José Luis Catalano

Grupo: H

Alumnos:
- Nicolas Mamani - niko_mamani@hotmail.com
- Ulises Cerquatti - cerquattiulises@gmail.com
- Martin Coser - cosermartin849@gmail.com
- Hernán Arias - hernanarias126@gmail.com
- Federico Fasani - fedefasani@gmail.com
- Celia Carranza - carranzacelia69@gmail.com

Índice:

Resumen

Breve descripción del proyecto:

El concepto central de este proyecto implica la implementación de un microcontrolador ESP8266 para monitorear la temperatura a través de un sensor DHT11. Cuando la temperatura detectada cae dentro de un rango predefinido, se activa un sistema de iluminación mediante el uso de un relé. Este relé, al gestionar la apertura y cierre de la conexión, controla el encendido y apagado de una luz LED, ofreciendo así una respuesta automática a las variaciones térmicas específicas.

Objetivos Principales:

- Establecer una conexión inalámbrica mediante Wi-Fi con el microcontrolador ESP8266.
- Configurar un circuito que integre un relé, un LED y un sensor de humedad DHT11 para facilitar la interacción.
- Establecer comunicación utilizando Packet Sender con el microcontrolador ESP8266 para gestionar y monitorizar datos.
- Implementar la técnica de modulación por ancho de pulso (PWM) para controlar el encendido y apagado del LED de manera efectiva.

Resultados destacados:

Conexión Inalámbrica Eficiente:

- Implementación de un circuito funcional con relé, LED y sensor de humedad DHT11, asegurando una interconexión efectiva entre estos componentes clave.

Funcionamiento Integrado del Circuito:

- Implementación de un circuito funcional con relé, LED y sensor de humedad DHT11, asegurando una interconexión efectiva entre estos componentes clave.

Comunicación Efectiva con Packet Sender:

- Empleo de Packet Sender para recibir paquetes de datos y gestionarlos según las condiciones establecidas en el código de Arduino, facilitando la interacción entre el sistema y el entorno controlador.

Control Dinámico del LED con PWM:

- Aplicación de la técnica de modulación por ancho de pulso (PWM) para controlar dinámicamente la iluminación del LED, permitiendo una gestión eficiente del encendido y apagado.

Introducción

Contexto y Justificación del Proyecto

En el marco de nuestro proyecto, exploramos la integración de tecnologías como el ESP8266, el sensor DHT11, un relé y un LED para abordar la gestión eficiente de la iluminación basada en la temperatura. Este informe detalla el proceso de diseño y desarrollo de esta solución práctica, destacando la sencillez y utilidad de nuestra propuesta.

Breve Descripción de la Problemática a Resolver

En este contexto, surge la necesidad de crear un sistema que permita la supervisión y control a distancia de dispositivos electrónicos, focalizándose especialmente en la gestión de la temperatura y la humedad.
Nuestro proyecto aborda la falta de sistemas automáticos y accesibles para ajustar la iluminación según las condiciones térmicas. La solución propuesta ofrece una respuesta eficiente y remota a esta problemática, mejorando tanto la eficiencia energética como la experiencia del usuario.

Objetivo

Objetivo General del Proyecto

El objetivo del proyecto fue diseñar, implementar y documentar un sistema automatizado mediante la plataforma Arduino. En el proyecto, se ha trabajado en la integración de diversos componentes, entre ellos el microcontrolador ESP8266, un sensor de humedad DHT11, un relé y un LED. La ejecución del proyecto abarcó el desarrollo del código correspondiente utilizando el entorno Arduino IDE, con el objetivo de gestionar la lectura del sensor y controlar el encendido del LED a través del relé.

Además, nos enfocamos en la creación de una maqueta que refleja la conexión física de cada componente. Esta maqueta proporciona una representación física del sistema implementado, permitiendo una comprensión de cada componente utilizado.

Objetivos Específicos

Conexión Inalámbrica Eficiente:

- Lograr conectar el dispositivo de control al microcontrolador ESP8266 mediante una conexión Wi-Fi. Esto significa que el sistema puede

comunicarse de manera efectiva sin depender de cables físicos, lo que simplifica la implementación y facilita la movilidad del sistema.

Funcionamiento Integrado del Circuito:

- Crear y poner en funcionamiento un circuito que incluye componentes como un relé, un LED y un sensor de humedad DHT11. Este diseño debe garantizar que estos elementos están interconectados de manera efectiva, permitiendo que el sistema opere de manera cohesiva y cumpla con sus funciones previstas.

Comunicación Efectiva con Packet Sender:

- Utilizar Packet Sender como herramienta central para la recepción de paquetes de datos. Este software permite manejar la información de acuerdo con las condiciones programadas en el código de Arduino. En esencia, Packet Sender actúa como el intermediario que posibilita la interacción y el intercambio de datos entre el sistema implementado y el entorno controlador definido por el código de Arduino.

Control Dinámico del LED con PWM:

- Implementar la técnica de modulación por ancho de pulso (PWM) para controlar de manera dinámica la iluminación del LED. Esta técnica nos brinda la capacidad de gestionar el encendido y apagado del dispositivo LED, permitiendo reflejar el control que realiza el sistema en cuanto a las condiciones establecidas en el código para la entrada de datos.

Marco Teórico

Fundamentos teóricos relacionados con el proyecto

- **Microcontroladores**: Un microcontrolador es un circuito integrado digital que puede ser usado para muy diversos propósitos debido a que es programable.
- **El ESP8266** es un microcontrolador de bajo costo y bajo consumo de energía que incluye capacidades de conectividad Wi-Fi.
- **Sensor de Temperatura y Humedad DHT11**: El sensor DHT11 es un sensor económico que proporciona mediciones de temperatura y humedad. Utiliza un termistor para medir la temperatura y un sensor de humedad capacitivo.
- **Relé:** Un relé es un interruptor controlado por un circuito eléctrico. En este proyecto, se utilizará para controlar la conexión y desconexión de la luz LED en respuesta a las lecturas del sensor de temperatura.
- T**écnica de modulación por ancho de pulso (PWM):** Esta técnica se utiliza para controlar la cantidad de energía entregada a un dispositivo eléctrico, como en el caso de la luz LED. Al modular el ancho de los pulsos de la señal

eléctrica, se puede controlar la intensidad luminosa de la luz LED de manera efectiva.

Tecnologías o conceptos claves utilizados

- Microcontrolador
- Sensor
- Relé
- LED
- Resistencia

Diseño y Metodología

Descripción detallada del diseño del sistema.

El diseño del sistema se centra en la implementación de un sistema de monitoreo y control de iluminación mediante un microcontrolador ESP8266, un sensor de temperatura y humedad DHT11, un relé y un LED. A continuación, se presenta una descripción detallada de cada componente y su interconexión:

1. ESP8266:

El ESP8266 sirve como la unidad central del sistema. Se encarga de la lectura de datos del sensor DHT11, la toma de decisiones basada en las condiciones de temperatura y la gestión del relé para controlar la iluminación. Además, establece una conexión Wi-Fi para permitir la monitorización remota y la gestión de datos.

2. Sensor de Temperatura y Humedad DHT11:

El sensor DHT11 está conectado al ESP8266 para proporcionar mediciones precisas de temperatura y humedad. La interfaz digital del sensor facilita la integración con el microcontrolador. Las lecturas del sensor son fundamentales para activar el sistema de iluminación cuando la temperatura cae fuera de un rango predefinido.

3. Relé:

El relé actúa como un interruptor controlado por el ESP8266. Cuando el sensor detecta una temperatura fuera del rango deseado, el relé se activa, cerrando la conexión eléctrica y permitiendo que la corriente fluya hacia el LED. De esta manera, el relé es el elemento clave para el control remoto de la iluminación.

4. LED:

El LED es la fuente de iluminación controlada por el sistema. Está conectado al relé de manera que, cuando el relé está activado, el LED se enciende, y cuando el relé está desactivado, el LED se apaga. La técnica de modulación por ancho de pulso (PWM) se utiliza para controlar la intensidad luminosa del LED, permitiendo una respuesta gradual y eficiente a las variaciones de temperatura.

5. Circuito y Conexiones:

- El sensor DHT11 está conectado al ESP8266 mediante pines digitales para la transferencia de datos.
- El relé se conecta al ESP8266 a través de pines digitales que permiten el control de la conexión eléctrica hacia el LED.
- El LED está conectado al relé, y la técnica de PWM se aplica para controlar la luminosidad.
- Se establece una fuente de energía eléctrica para alimentar el sistema.

6. Comunicación Wi-Fi:

El ESP8266 utiliza su capacidad de conexión Wi-Fi para establecer una comunicación inalámbrica. Esto posibilita la monitorización remota y la gestión de datos, permitiendo que el sistema sea controlado y supervisado desde ubicaciones externas.

7. Técnica de Modulación por Ancho de Pulso (PWM):

La modulación por ancho de pulso se implementa en el control del LED para ajustar su brillo de manera eficiente. La variación en el ancho de los pulsos eléctricos determina la cantidad de luz emitida por el LED, proporcionando un control preciso y una respuesta gradual.

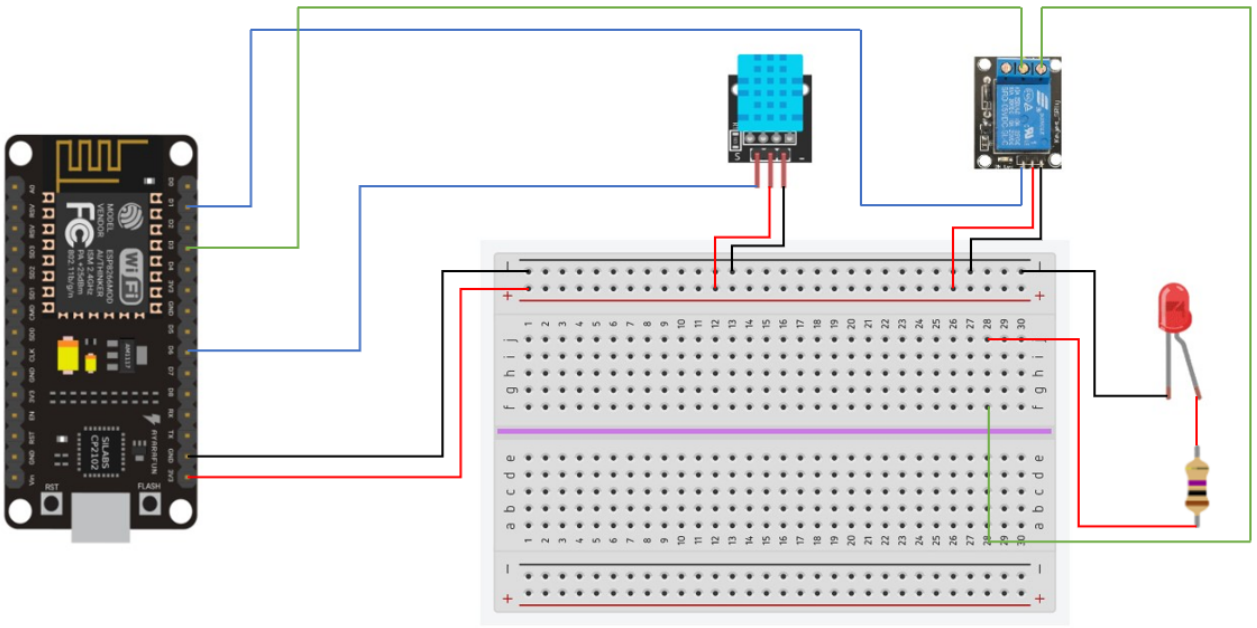
Metodología utilizada en el desarrollo.

Para desarrollar el proyecto, se siguió una metodología basada en los siguientes pasos:

1. **Definición de requerimientos:** Se establecieron los requerimientos del sistema, que incluyen la detección de la temperatura ambiente y el control preciso de la iluminación del LED.
2. **Selección de componentes:** Se seleccionaron los componentes necesarios para el proyecto, incluyendo el sensor de temperatura, el microcontrolador ESP8266, el relé, el LED y la resistencia.
3. **Diseño del circuito:** Se diseñó el circuito electrónico que permite la conexión de los componentes seleccionados y el control de la iluminación del LED en función de las lecturas del sensor de temperatura.
4. **Programación del microcontrolador:** Se programó el microcontrolador ESP8266 para procesar las lecturas del sensor de temperatura y enviar señales de control al relé.
5. **Pruebas y ajustes:** Se realizaron pruebas del sistema para verificar su correcto funcionamiento y se realizaron ajustes necesarios para lograr un control preciso de la iluminación del LED en función de las variaciones de temperatura detectadas.
6. **Implementación:** Finalmente, se implementó el sistema en un protoboard o placa de circuito impreso para su uso en aplicaciones prácticas.

Esquema de Conexionado

Diagrama de conexión de los componentes de Arduino



Descripción de la conexión de cada componente

- **Placa ESP8266 (NodeMCU 1.0):** La placa que utilizamos es una ESP8266 en la cual conectamos los distintos componentes. Esta placa tiene la función de conexión mediante WiFi, la que utilizamos para enviar mensajes de las lecturas tomadas.
    - Conectamos el pin 3.3v (+) a la línea positiva de la protoboard para alimentar a los demás componentes.
    - Conectamos el pin GND (-) a la línea negativa de la protoboard.

- **Relé:** Utilizamos el relé para alimentar con energía la luz led que nos va a indicar los cambios de temperatura de acuerdo a los rangos asignados.
    - Conectamos el pin de señal del relé al pin D1 (GPIO5) de nuestra placa. Este pin nos va a dar la señal de activar y desactivar el relé.
    - El pin (+) del relé lo conectamos a la línea positiva (3.3v) de la protoboard.
    - El pin (-) lo conectamos a la línea negativa de la protoboard.
    - Alimentamos con (+) el pin superior del relé y conectamos el pin NO (Normal Abierto) con la resistencia que se encuentra con el led. Este cable va a alimentar con tensión (+) el led por medio de la resistencia.

- **Sensor DHT11 (HW-507):**
    - Conectamos el pin señal del sensor al pin D6 (GPIO12) que va a ser por donde la placa va a tomar los datos de lectura de temperatura y humedad.
    - Conectamos el pin (+) del sensor a la línea positiva de la protoboard.
    - Conectamos el pin (-) del sensor a la línea negativa de la protoboard.

- **Resistencia y Led:**
    - Conectamos el pin (+) del led a la resistencia, y esta resistencia recibe alimentación del contacto NO (Normal Abierto) del relé.
    - Contactamos el pin (-) a la línea negativa de la protoboard.

Desarrollo del Código

Extractos clave del código utilizado.

```
//LIBRERIAS
#include <DHT.h>
#include "ESP8266WiFi.h"
```

1) En esta sección del código, se incluyen las librerías necesarias para el proyecto. **DHT.h** es la librería del sensor de temperatura y humedad, y **ESP8266WiFi.h** es la librería de la placa ESP8266.

Librería DHT.h

Propósito:

- Esta librería facilita la interacción con sensores de humedad y temperatura de la serie DHT. En tu código, utilizas un sensor DHT11.

Funcionalidades clave:

- Lectura de la humedad y la temperatura del sensor DHT.
- Es compatible con varios modelos de sensores DHT.

Libreria ESP8266WiFi

Propósito:

- Esta librería proporciona funciones para la configuración y gestión de la conexión Wi-Fi en dispositivos basados en el módulo ESP8266, como el NodeMCU o el ESP-01.

Funcionalidades clave:

- Conexión a redes Wi-Fi.
- Configuración de modos de conexión.
- Obtención de información de red (IP, estado, etc.).

```
// CONFIGURACION DE PARAMETROS WIFI
const char *ssid = "w-ala2-A"; // NOMBRE DE LA RED
const char *password = "ala2frvm"; // CONTRASEÑA DE LA RED
```

Configuración de parámetros Wi-Fis

- En esta parte del código se definen el nombre y la contraseña de la red Wi-Fi a la que se conectará el dispositivo. En este caso será **w-ala2-A**, y su contraseña: **ala2frvm.**

```
#define DHTPIN 12     // PIN DEL SENSOR
#define DHTTYPE DHT11 // SENSOR UTILIZADO
#define RELAY_PIN 5   // PIN DEL RELE
#define PWM_LED 0 // PIN DE ALIMENTACION DEL LED
```

Definiciones de pines

- Se definen los pines utilizados en el proyecto.

DHTPIN: Es el pin de la placa donde está conectado el sensor DHT.

DHTTYPE: Esta constante define el tipo específico de sensor DHT que estás utilizando. En este caso, estás utilizando un sensor DHT11.

RELAY_PIN: es el pin de la placa donde está conectado el relé.

PWM_LED: Es el pin de la placa utilizado para controlar la intensidad del LED mediante modulación por ancho de pulso (PWM)

```
void setup() {
  Serial.begin(9600);
  dht.begin();

  pinMode(RELAY_PIN, OUTPUT); // CONFIGURAR PIN DE RELE COMO SALIDA
  pinMode(PWM_LED, OUTPUT); // CONDIFURAR PIN PWM COMO SALIDA

  digitalWrite(PWM_LED, HIGH);  //SETEAR PWM EN HIGH
  digitalWrite(RELAY_PIN, HIGH); //SETEAR EL RELE EN HIGH

  WiFi.mode(WIFI_STA);  //TIPO DE CONEXION
  WiFi.begin(ssid, password); //ESTABLECER CONEXION WIFI


  while (WiFi.status() != WL_CONNECTED) { //CONECTANDO...
      delay(700);
      Serial.print(".");
  }

  //CUANDO SE ESTABLECE LA CONEXION
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println(WiFi.localIP());  //MOSTRAR IP PLACA
}
```

Función setup()

En la función setup(), se inicia la comunicación serial, se configuran los pines como salidas, se establece la conexión Wi-Fi y se espera hasta que se conecte.

- **Serial.begin(9600):**
  Inicia la comunicación serie a una velocidad de 9600 baudios. Esto es útil para imprimir mensajes en el monitor serial para propósitos de depuración.

- **dht.begin():**
  Inicia la comunicación con el sensor DHT. Esto es necesario antes de realizar lecturas del sensor.

- **pinMode(RELAY_PIN, OUTPUT):**
  Configura el pin RELAY_PIN (en este caso, el pin 5) como una salida. Este pin probablemente está conectado a un relé que controla algún dispositivo externo.

- **pinMode(PWM_LED, OUTPUT):**Configura el pin PWM_LED (en este caso, el pin 0) como una salida. Este pin se usará para controlar la intensidad de un LED mediante la técnica de modulación por ancho de pulso (PWM).

- **digitalWrite(PWM_LED, HIGH):**
  Establece el pin PWM_LED en alto.

- **digitalWrite(RELAY_PIN, HIGH):**
  Establece el pin RELAY_PIN en alto.

- **WiFi.mode(WIFI_STA):**
  Configura el módulo ESP8266 en modo estación (WIFI_STA), lo que significa que se conectará a una red Wi-Fi existente como cliente.

- **WiFi.begin(ssid, password):**
  Inicia la conexión a la red Wi-Fi especificada por ssid y password. Esta es una operación que puede llevar un tiempo dependiendo de la calidad de la señal y otros factores.

- **while (WiFi.status() != WL_CONNECTED):**
  Este bucle espera hasta que el dispositivo se conecte exitosamente a la red Wi-Fi. Mientras está esperando, imprime un punto en el monitor serial cada 700 milisegundos.

- **Serial.println("WiFi connected"):**
  Imprime un mensaje indicando que la conexión Wi-Fi se ha establecido con éxito.

- **Serial.println(WiFi.localIP()):**
  Imprime la dirección IP local asignada al dispositivo por el enrutador Wi-Fi. Esta dirección es útil para acceder al dispositivo a través de la red.

```
void loop() {
  delay(2000);

  //VARIABLES DE LECTURA
  float humedad = dht.readHumidity();
  float temperatura = dht.readTemperature();

  //MOSTRAR DATOS EN MONITOR
  if (isnan(humedad) || isnan(temperatura)) {
    Serial.println("Error al leer el sensor DHT");
  } else {
    Serial.print("Humedad: ");
    Serial.print(humedad);
    Serial.print("%\t");
    Serial.print("Temperatura: ");
    Serial.print(temperatura);
    Serial.println("°C");


  //VALOR DE INTENSIDAD DE LED
    int intensidadLed = 0;

  //VALORES DE INTENSIDAD EN FUNCION DE LA TEMPERATURA
    if (temperatura < 29.0) {
      intensidadLed = 0; // Apaga el LED
      sendTcp("T:" + String(temperatura) + "Led Apagado");
    }
    //SI LA TEMPERATURA ES SUPERIOR A 29°
    else if (temperatura >= 29.0 && temperatura < 30.0) {
      intensidadLed = 64; // 25% de intensidad
      sendTcp("T:" + String(temperatura) + "Led Encendido 25%");
    }
```

```
    //SI LA TEMPERATURA ES SUPERIOR A 30° Y MENOR A 31°
    else if (temperatura >= 30.0 && temperatura < 31.0) {
      intensidadLed = 128; // 50% de intensidad
      sendTcp("T:" + String(temperatura) + "Led Encendido 50%");
    }
    //SI LA TEMPERATURA ES SUPERIOR A 31° Y MENOR A 32°
    else if (temperatura >= 31.0 && temperatura < 32.0) {
      intensidadLed = 191; // 75% de intensidad
      sendTcp("T:" + String(temperatura) + "Led Encendido 75%");
    }
    //SI LA TEMPERATURA ES SUPERIOR A 32°
    else {
      intensidadLed = 255; // 100% de intensidad
      sendTcp("T:" + String(temperatura) + "Led Encendido 100%");
    }

  //SETEAR EL NIVEL DE INTENSIDAD EN EL LED CON PWM
    analogWrite(PWM_LED, intensidadLed);
    delay(1000);

  }
}
```

Función loop():

En la función loop(), se realiza la lectura del sensor DHT, se muestran los datos en el monitor serial, se determina la intensidad del LED según la temperatura y se envía un mensaje TCP.

float humedad = dht.readHumidity();: Lee la humedad del sensor DHT y almacena el valor en la variable humedad.

float temperatura = dht.readTemperature();: Lee la temperatura del sensor DHT y almacena el valor en la variable temperatura.

If

Se utiliza una serie de condiciones if-else para determinar la intensidad del LED (intensidadLed) en función de la temperatura leída:

- Si la temperatura es menor a 29.0°C, apaga el LED (intensidadLed = 0).

- Si la temperatura está entre 29.0°C y 30.0°C, enciende el LED al 25% de intensidad (intensidadLed = 64).

- Si la temperatura está entre 30.0°C y 31.0°C, enciende el LED al 50% de intensidad (intensidadLed = 128).

- Si la temperatura está entre 31.0°C y 32.0°C, enciende el LED al 75% de intensidad (intensidadLed = 191).

- Si la temperatura es mayor o igual a 32.0°C, enciende el LED al 100% de intensidad (intensidadLed = 255).

Se llama la función **sendTcp()** con un mensaje que incluye información sobre la temperatura y el estado del LED. Este mensaje se envía a través de una conexión TCP.

analogWrite(PWM_LED, intensidadLed); se usa para establecer la intensidad del LED mediante modulación por ancho de pulso (PWM). El valor de intensidadLed determina la cantidad de tiempo que el LED permanece encendido durante cada ciclo.

```
//ENVIO DE MENSAJES TCP
void sendTcp(String message) {
  if (WiFi.status() == WL_CONNECTED) {
    WiFiClient client;
    //SETEAMOS DIRECCION IP Y PUERTO
    if (client.connect("192.168.201.94", 8081)) {
      //SI LA CONEXION SE ESTABLECE SE ENVIA EL MENSAJE
      client.print(message);
    } else {
      Serial.println("Fallo en la conexión");
    }
    //CIERRA LA CONEXION
    client.stop();
  }
}
```

if (WiFi.status() == WL_CONNECTED) {:
Esta condición verifica si el dispositivo está actualmente conectado a una red Wi-Fi. WiFi.status() devuelve el estado de la conexión Wi-Fi, y WL_CONNECTED es un valor que indica que la conexión está establecida.

WiFiClient client;:
Se crea un objeto WiFiClient llamado client. Esta clase se utiliza para establecer una conexión cliente en una red Wi-Fi.

if (client.connect("192.168.201.94", 8081)) {:
Este bloque de código intenta establecer una conexión con un servidor remoto que tiene la dirección IP 192.168.201.94 y el puerto 8081. Si la conexión es exitosa, entra en el bloque if.

client.print(message);:
Si la conexión es exitosa, el contenido de la variable message se envía al servidor utilizando el método print() del objeto client. En este caso, message contiene la información sobre la temperatura y el estado del LED que se desea enviar.

else { Serial.println("Fallo en la conexión"); }: Si la conexión no se establece correctamente, imprime un mensaje de error en el monitor serial indicando que ha habido un fallo en la conexión.

client.stop();:
Después de enviar el mensaje o si hubo un fallo en la conexión, se llama a client.stop() para cerrar la conexión con el servidor. Esto libera recursos y prepara la conexión para futuros intentos.

Resultados

La salida desde el IDE Arduino nos permite visualizar que se ha conectado correctamente al WiFi, así como visualizar la IP que tiene asignada el ESP8266 junto con la humedad y la temperatura.

- A continuación adjuntamos los resultados de la consola de Arduino IDE con el programa en ejecución.

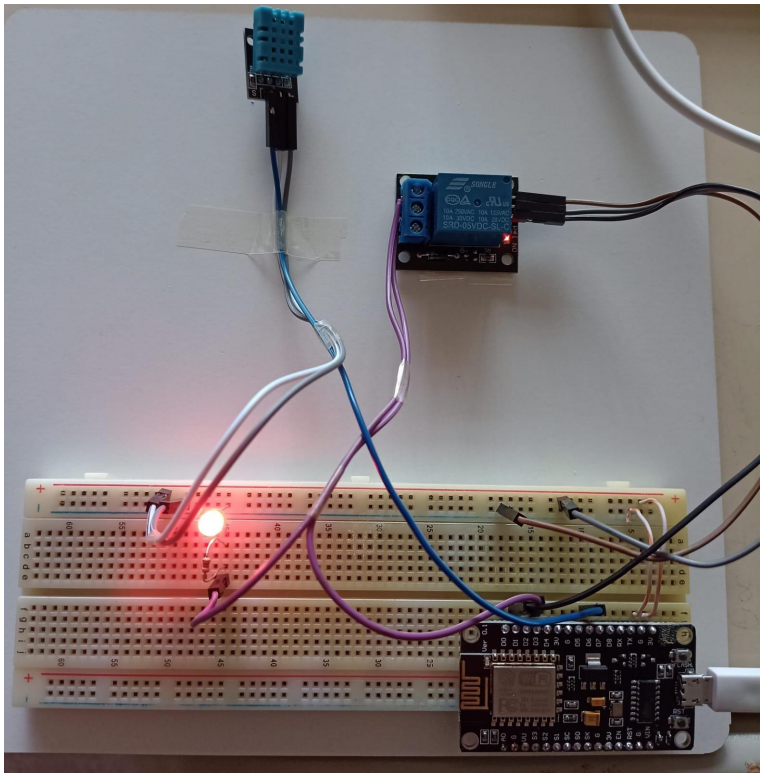
```
.........................................
WiFi connected
192.168.201.193
Humedad: 66.00% Temperatura: 25.80°C
Humedad: 35.00% Temperatura: 23.70°C
Humedad: 35.00% Temperatura: 23.70°C
Humedad: 35.00% Temperatura: 23.60°C
Humedad: 35.00% Temperatura: 23.60°C
Humedad: 35.00% Temperatura: 23.60°C
Humedad: 40.00% Temperatura: 23.60°C
Humedad: 52.00% Temperatura: 23.60°C
Humedad: 58.00% Temperatura: 23.60°C
Humedad: 61.00% Temperatura: 23.70°C
Humedad: 64.00% Temperatura: 23.80°C
Humedad: 66.00% Temperatura: 24.00°C
Humedad: 68.00% Temperatura: 24.10°C
Humedad: 69.00% Temperatura: 24.20°C
Humedad: 65.00% Temperatura: 24.40°C
Humedad: 67.00% Temperatura: 24.50°C
Humedad: 69.00% Temperatura: 24.60°C
Humedad: 72.00% Temperatura: 24.70°C
Humedad: 73.00% Temperatura: 24.70°C
Humedad: 74.00% Temperatura: 24.80°C
```

Packet Sender
Desde el programa Packet Sender abrimos el puerto 8081 con el protocolo TCP para luego desde el IDE Arduino enviar la temperatura, el estado del LED y la intensidad del LED al programa Packet Sender.

- A continuación adjuntamos la recepción de paquetes desde Packet Sender que incluye en el mensaje la temperatura sensada y el estado del led.

You	8081	TCP		T:26.60Led Encendido 75%
You	8081	TCP		T:26.90Led Encendido 75%
You	8081	TCP		T:26.50Led Encendido 75%
You	8081	TCP		T:26.40Led Encendido 75%
You	8081	TCP		T:26.30Led Encendido 75%
You	8081	TCP		T:26.40Led Encendido 75%
You	8081	TCP		T:26.10Led Encendido 75%
You	8081	TCP		T:26.00Led Encendido 75%
You	8081	TCP		T:25.90Led Encendido 50%
You	8081	TCP		T:25.80Led Encendido 50%
You	8081	TCP		T:25.70Led Encendido 50%
You	8081	TCP		T:25.60Led Encendido 50%
You	8081	TCP		T:25.70Led Encendido 50%
You	8081	TCP		T:25.40Led Encendido 50%

A continuación insertamos una imagen de la maqueta terminada



Conclusiones

- Se logró el monitoreo de la temperatura con el sensor DHT11
- Se implementó con éxito el envío inalámbrico de los datos de temperatura al módulo ESP8266.
- Se automatizó el encendido y apagado de un circuito LED mediante el control de un relé utilizando el ESP8266.
- Se logró implementar la comunicación del Packet Sender con el microcontrolador ESP8266.