Praktik Pengiriman dan Penerimaan Data Sensor melalui MQTT menggunakan ESP32

Ranindya Dwi Qintari
Fakultas Vokasi, Universitas Brawijaya
Email: ranindyadq@student.ub.ac.id

Abstrak: Praktikum ini membahas implementasi sistem Internet of Things (IoT) sederhana menggunakan mikrokontroler ESP32 untuk membaca data suhu dan kelembapan dari sensor DHT22 serta mengirimkannya ke broker MQTT melalui koneksi Wi-Fi. Selain itu, ESP32 juga berfungsi sebagai subscriber yang menerima perintah dari broker MQTT untuk mengendalikan LED secara remote. Tujuan dari praktikum ini adalah untuk memahami konsep komunikasi data pada IoT menggunakan protokol MQTT, termasuk proses publish dan subscribe data antar perangkat. Metode yang digunakan meliputi inisialisasi koneksi Wi-Fi, pembacaan data sensor DHT22, pengiriman data melalui topik MQTT, serta penerimaan pesan dan eksekusi perintah dari broker. Hasil praktikum menunjukkan bahwa sistem dapat mengirim dan menerima data secara real-time dan stabil, selama koneksi internet tersedia. Praktikum ini menunjukkan penerapan komunikasi MQTT sebagai fondasi penting dalam sistem IoT modern.

Abstract: This practicum discusses the implementation of a simple Internet of Things (IoT) system using the ESP32 microcontroller to read temperature and humidity data from the DHT22 sensor and transmit it to an MQTT broker via Wi-Fi. In addition, the ESP32 acts as a subscriber that receives commands from the MQTT broker to remotely control an LED. The objective of this practicum is to understand data communication in IoT using the MQTT protocol, including the process of publishing and subscribing data between devices. The methods used include Wi-Fi initialization, DHT22 sensor data acquisition, MQTT topic-based data transmission, and receiving commands from the broker to trigger outputs. The results show that the system is capable of sending and receiving data in real time and remains stable as long as the internet connection is available. This practicum demonstrates the use of MQTT communication as a fundamental component of modern IoT systems.

Keywords- Internet of Things, ESP32, MQTT, DHT22, Wi-Fi, Publish-Subscribe

1. PENDAHULUAN

1.1 Latar Belakang

Internet of Things (IoT) merupakan paradigma teknologi yang memungkinkan perangkat fisik seperti sensor dan aktuator saling terhubung dan berkomunikasi melalui jaringan internet. Salah satu komponen penting dalam sistem IoT adalah protokol komunikasi yang digunakan untuk mengirim dan menerima data secara efisien dan real-time. Protokol Message Queuing Telemetry Transport (MQTT) menjadi salah satu protokol yang paling banyak digunakan dalam pengembangan IoT karena sifatnya yang ringan, cepat, dan hemat bandwidth.

Pada praktik ini, digunakan mikrokontroler ESP32 yang memiliki fitur Wi-Fi untuk menghubungkan perangkat ke jaringan internet. ESP32 bertugas membaca data suhu dan kelembapan dari sensor DHT22 dan mengirimkan data tersebut ke broker MQTT melalui metode publish. Selain itu, ESP32 juga mampu menerima data dari broker (subscribe) untuk mengendalikan LED secara jarak jauh. Praktikum ini bertujuan untuk memberikan pemahaman praktis mengenai komunikasi dua arah dalam sistem IoT menggunakan protokol MQTT.

1.2 Tujuan Eksperimen

- Menghubungkan ESP32 ke jaringan internet menggunakan Wi-Fi.
- Mengukur suhu dan kelembapan menggunakan sensor DHT22.
- Mengirimkan data sensor ke broker MQTT menggunakan metode publish.
- Menerima data dari broker MQTT dan mengendalikan LED berdasarkan perintah yang diterima (subscribe).
- Menampilkan data sensor dan status LED di Serial Monitor secara real-time.

2. METODOLOGI

2.1 Alat dan Bahan

- Laptop
- Visual Studio Code
- PlatformIO Extension atau Arduino IDE
- Wokwi Simulator (untuk simulasi)
- Mikrokontroler ESP32 (simulasi)
- Sensor DHT22
- LED (merah)
- Kabel jumper (simulasi)
- Koneksi internet/Wi-Fi
- Broker MQTT publik (contoh: broker.emqx.io)

2.2 Langkah Implementasi

2.2.1 Simulasi Rangkaian di Wokwi

1. Membuka situs Wokwi Simulator dan memilih template ESP32.
2. Menambahkan komponen:
   - Sensor DHT22
   - LED Merah
3. Melakukan wiring virtual:
   - DHT22 data → GPIO15
   - LED Merah → GPIO2
   - VCC dan GND sesuai dengan power ESP32
4. Mengatur koneksi Wi-Fi simulator:
   - SSID: Wokwi-GUEST
   - Password: "" (kosong)
5. Menulis kode program untuk:
   - Koneksi ESP32 ke Wi-Fi
   - Pembacaan data suhu dan kelembapan dari DHT22
   - Publish data ke broker MQTT pada topik IOT/Test1/temp dan IOT/Test1/hum
   - Subscribe ke topik IOT/Test1/mqtt untuk menerima perintah kendali LED
   - Menyalakan/mematikan LED berdasarkan pesan dari broker
6. Mengamati hasil pada **Serial Monitor** berupa:
   - Output suhu dan kelembapan setiap 2 detik
   - Status koneksi Wi-Fi dan MQTT
   - Status LED (ON/OFF) berdasarkan pesan yang diterima melalui MQTT

2.2.2 Penggunaan Broker MQTT

1. Menggunakan broker MQTT publik seperti:
   broker.emqx.io
2. Topik MQTT yang digunakan:
   - IOT/Test1/temp → untuk publish data suhu
   - IOT/Test1/hum → untuk publish data kelembapan
   - IOT/Test1/mqtt → untuk subscribe perintah kendali LED
3. Format pesan:
   - Nilai suhu dan kelembapan dalam string (misal "27.3" dan "65.1")
   - Pesan "1" untuk menyalakan LED, pesan lainnya untuk mematikan

3. HASIL DAN PEMBAHASAN

3.1 Hasil Praktik

Praktikum ini berhasil mengimplementasikan sistem pengiriman dan penerimaan data sensor secara real-time menggunakan protokol MQTT. Mikrokontroler ESP32 digunakan untuk membaca data suhu dan kelembapan dari sensor DHT22, kemudian mengirimkan data tersebut ke broker MQTT publik (contoh: broker.emqx.io) melalui topik IOT/Test1/temp dan IOT/Test1/hum.

Selain sebagai publisher, ESP32 juga bertindak sebagai subscriber untuk menerima perintah melalui topik IOT/Test1/mqtt. Jika data yang diterima bernilai '1', maka LED merah menyala; jika sebaliknya,

LED mati. Sistem berhasil menunjukkan komunikasi dua arah melalui MQTT berjalan dengan baik dan stabil selama koneksi internet tersedia.

Tabel berikut menunjukkan contoh hasil data yang berhasil dipublish ke broker MQTT diawal:

Parameter	Nilai Sebelum Perubahan
Suhu	24.00
Kelembapan	40.0

Tabel berikut menunjukkan contoh hasil data yang berhasil dipublish ke broker MQTT LED nyala:

Parameter	Nilai Sebelum Perubahan
Suhu	24.00
Kelembapan	40.0
Status LED	1 (nyala)

Tabel berikut menunjukkan contoh hasil data yang berhasil dipublish ke broker MQTT LED mati:

Parameter	Nilai Sebelum Perubahan
Suhu	24.00
Kelembapan	40.0
Status LED	0 (mati)

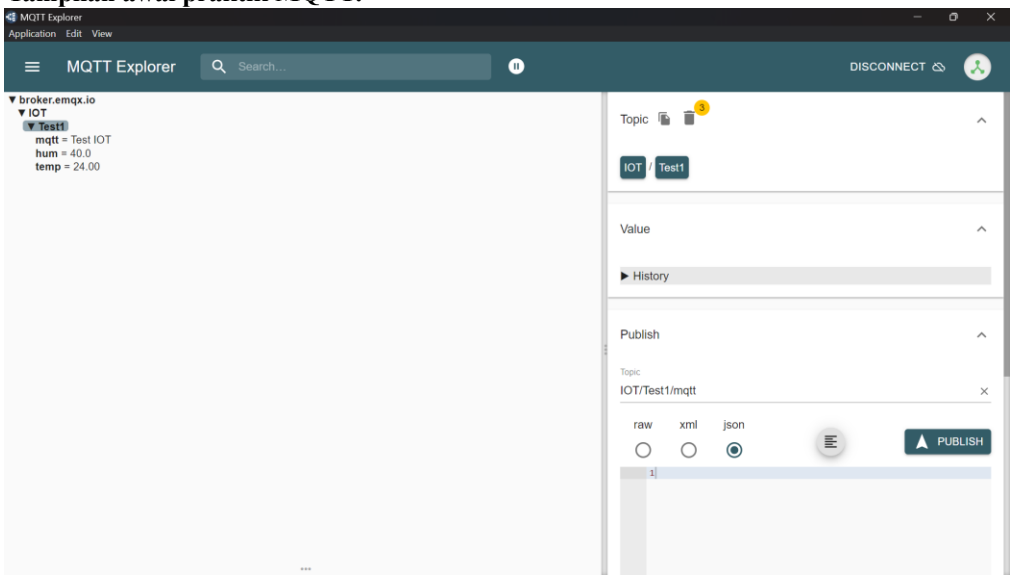
Data diperbarui secara berkala setiap 2 detik. Status LED berubah tergantung pesan yang diterima dari broker MQTT.

3.2 Tampilan Hasil Praktik

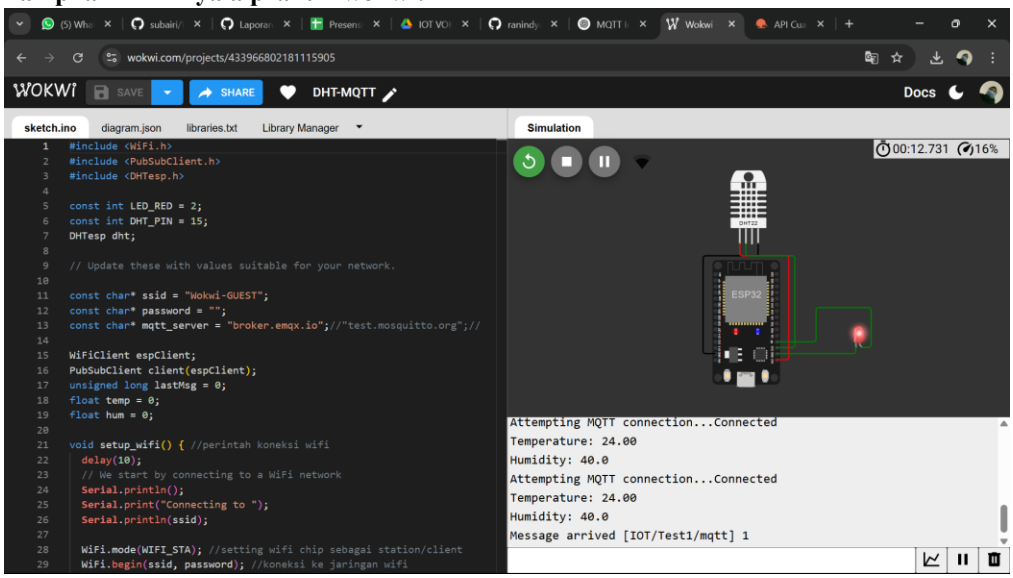
- LED menyala saat menerima pesan "1" dari topik MQTT.
- LED mati saat menerima pesan "0" dari topik MQTT.
- Serial monitor menampilkan data suhu dan kelembapan secara real-time.

3.2.1 Tampilan awal praktik wokwi:

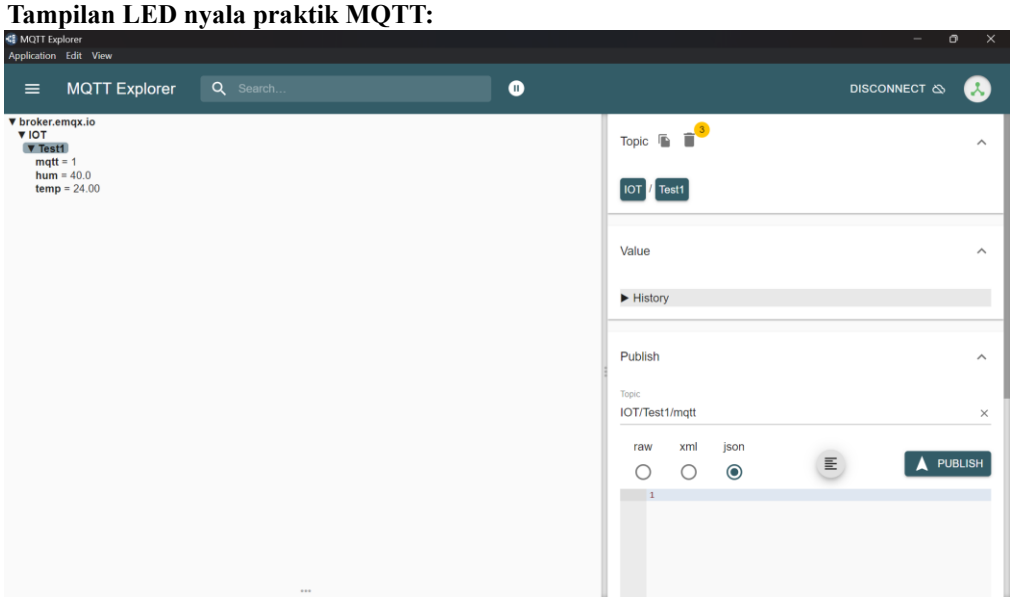
3.2.2 Tampilan awal praktik MQTT:



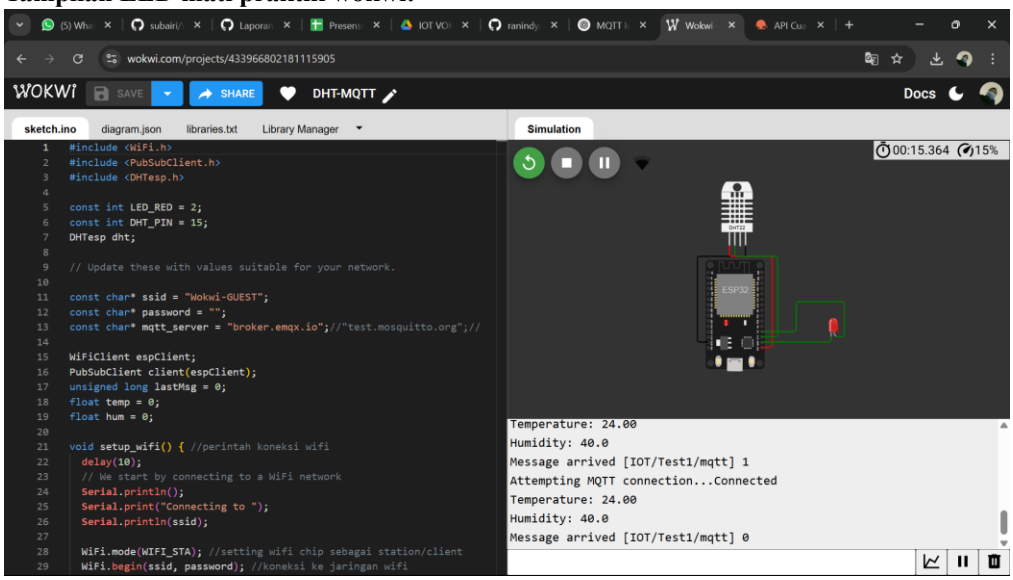
3.2.3 Tampilan LED nyala praktik wokwi:



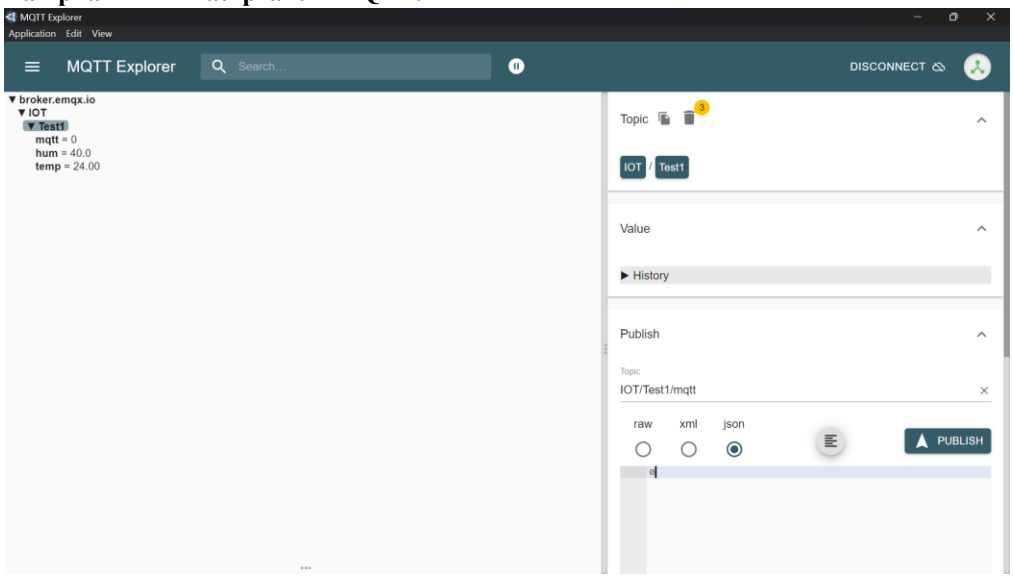
3.2.4 Tampilan LED nyala praktik MQTT:



3.2.5 Tampilan LED mati praktik wokwi:



3.2.6 Tampilan LED mati praktik MQTT:



4. LAMPIRAN

4.1 Kode Program

- **main.cpp**

```
#include <WiFi.h>
#include <PubSubClient.h>
#include <DHTesp.h>

const int LED_RED = 2;
const int DHT_PIN = 15;
DHTesp dht;

// Update these with values suitable for your network.

const char* ssid = "Wokwi-GUEST";
const char* password = "";
const char* mqtt_server = "broker.emqx.io";//"test.mosquitto.org";//

WiFiClient espClient;
PubSubClient client(espClient);
unsigned long lastMsg = 0;
float temp = 0;
float hum = 0;

void setup_wifi() { //perintah koneksi wifi
  delay(10);
  // We start by connecting to a WiFi network
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.mode(WIFI_STA); //setting wifi chip sebagai station/client
```

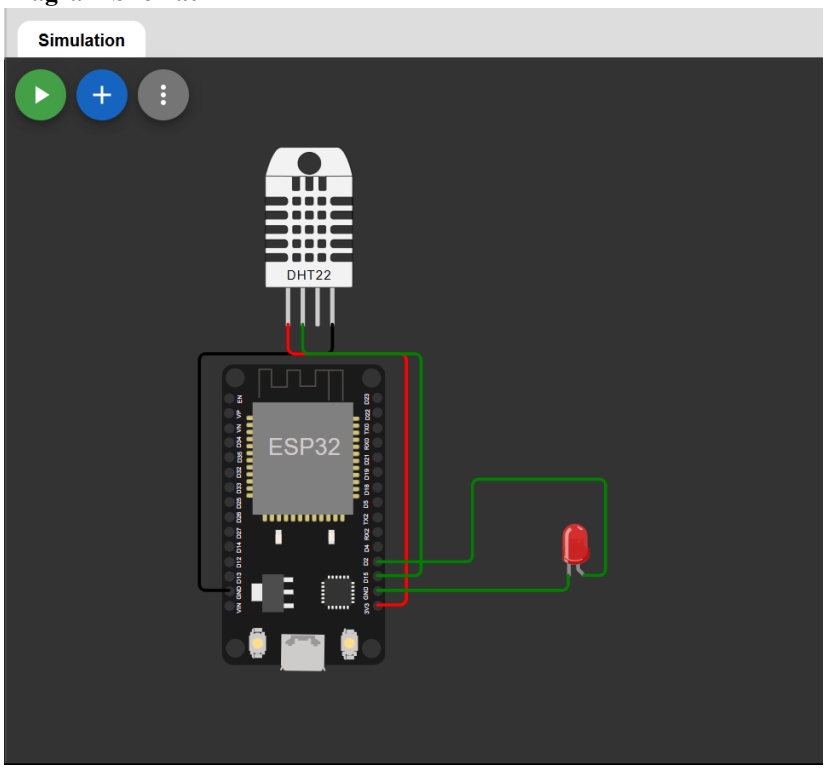
```
  WiFi.begin(ssid, password); //koneksi ke jaringan wifi

  while (WiFi.status() != WL_CONNECTED) { //perintah tunggu esp32 sampi terkoneksi ke wifi
    delay(500);
    Serial.print(".");
  }

  randomSeed(micros());

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

void callback(char* topic, byte* payload, unsigned int length) { //perintah untuk menampilkan
data ketika esp32 di setting sebagai subscriber
  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] ");
  for (int i = 0; i < length; i++) { //mengecek jumlah data yang ada di topik mqtt
    Serial.print((char)payload[i]);
  }
  Serial.println();

  // Switch on the LED if an 1 was received as first character
  if ((char)payload[0] == '1') {
    digitalWrite(LED_RED, HIGH);   // Turn the LED on
  } else {
    digitalWrite(LED_RED, LOW);  // Turn the LED off
  }
}

void reconnect() { //perintah koneksi esp32 ke mqtt broker baik itu sebagai publusher atau
subscriber
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // perintah membuat client id agar mqtt broker mengenali board yang kita gunakan
    String clientId = "ESP32Client-";
    clientId += String(random(0xffff), HEX);
    // Attempt to connect
    if (client.connect(clientId.c_str())) {
      Serial.println("Connected");
      // Once connected, publish an announcement...
      client.publish("IOT/Test1/mqtt", "Test IOT"); //perintah publish data ke alamat topik
yang di setting
      // ... and resubscribe
      client.subscribe("IOT/Test1/mqtt"); //perintah subscribe data ke mqtt broker
```

```
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}

void setup() {
  pinMode(LED_RED, OUTPUT);     // inisialisasi pin 2 / ledbuiltin sebagai output
  Serial.begin(115200);
  setup_wifi(); //memanggil void setup_wifi untuk dieksekusi
  client.setServer(mqtt_server, 1883); //perintah connecting / koneksi awal ke broker
  client.setCallback(callback); //perintah menghubungkan ke mqtt broker untuk subscribe data
  dht.setup(DHT_PIN, DHTesp::DHT22);//inisialiasi komunikasi dengan sensor dht22
}

void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();

  unsigned long now = millis();
  if (now - lastMsg > 2000) { //perintah publish data
    lastMsg = now;
    TempAndHumidity  data = dht.getTempAndHumidity();

    String temp = String(data.temperature, 2); //membuat variabel temp untuk di publish ke
broker mqtt
    client.publish("IOT/Test1/temp", temp.c_str()); //publish data dari varibel temp ke
broker mqtt

    String hum = String(data.humidity, 1); //membuat variabel hum untuk di publish ke broker
mqtt
    client.publish("IOT/Test1/hum", hum.c_str()); //publish data dari varibel hum ke broker
mqtt

    Serial.print("Temperature: ");
    Serial.println(temp);
    Serial.print("Humidity: ");
    Serial.println(hum);
  }
}
```

- **diagram.json**

```
{
  "version": 1,
  "author": "Ranindya Dwi Qintari",
  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": 43.1, "left": -5, "attrs": {}
},
    { "type": "wokwi-dht22", "id": "dht1", "top": -95.7, "left": 23.4, "attrs": {} },
    {
      "type": "wokwi-led",
      "id": "led1",
      "top": 140.4,
      "left": 205.4,
      "attrs": { "color": "red" }
    }
  ],
  "connections": [
    [ "esp:TX0", "$serialMonitor:RX", "", [] ],
    [ "esp:RX0", "$serialMonitor:TX", "", [] ],
    [ "dht1:GND", "esp:GND.2", "black", [ "v19.2", "h-86.4", "v153.7" ] ],
    [ "dht1:VCC", "esp:3V3", "red", [ "v19.2", "h76.8", "v163.2" ] ],
    [ "dht1:SDA", "esp:D15", "green", [ "v19.2", "h76.9", "v144.2" ] ],
    [ "led1:C", "esp:GND.1", "green", [ "v0" ] ],
    [ "esp:D2", "led1:A", "green", [ "h61.9", "v-53.6", "h86.4", "v57.6" ] ]
  ],
  "dependencies": {}
}
```

4.2 Diagram Skematik



5. PENUTUP

5.1 Kesimpulan

Dari hasil eksperimen, dapat disimpulkan bahwa mikrokontroler ESP32 berhasil digunakan untuk mengirim dan menerima data sensor secara real-time menggunakan protokol MQTT. ESP32 mampu membaca data dari sensor DHT22, yaitu suhu dan kelembapan, dan mengirimkannya ke broker MQTT melalui topik tertentu. Selain itu, ESP32 juga mampu menerima pesan dari topik MQTT yang sama untuk mengontrol perangkat output, yaitu LED.

Sistem berjalan dengan stabil dan responsif selama koneksi internet tersedia. Penggunaan MQTT terbukti efisien dalam komunikasi dua arah antara ESP32 dan platform cloud MQTT broker. Praktik ini menunjukkan penerapan konsep dasar Internet of Things (IoT) dalam hal komunikasi data, monitoring lingkungan, dan kontrol perangkat secara remote menggunakan MQTT.

Dengan demikian, praktikum ini berhasil mengilustrasikan implementasi sistem IoT sederhana yang dapat dikembangkan lebih lanjut untuk aplikasi monitoring atau kontrol skala kecil hingga besar.