Praktik Sistem Monitoring dan Kontrol Berbasis Web Menggunakan ESP32, MQTT, dan Sensor DHT22

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

Abstrak: Praktikum ini membahas implementasi sistem Internet of Things (IoT) berbasis web yang menggunakan mikrokontroler ESP32, protokol MQTT, dan antarmuka HTML. Sistem dirancang untuk memantau suhu dan kelembapan secara real-time menggunakan sensor DHT22, serta mengirimkan data ke broker MQTT publik (broker.emqx.io). Data kemudian divisualisasikan melalui grafik dinamis di antarmuka web menggunakan Highcharts. Selain itu, sistem memungkinkan kendali LED berbasis PWM secara jarak jauh menggunakan kontrol slider di web, di mana nilai PWM dikirim melalui topik MQTT dan dikonversi menjadi sinyal yang mengatur intensitas LED di ESP32. Praktikum ini menunjukkan bahwa komunikasi dua arah antara web dan perangkat IoT dapat dilakukan secara efisien melalui MQTT, tanpa memerlukan server backend atau cloud khusus. Hasilnya, sistem dapat menampilkan data sensor secara akurat dan mengontrol perangkat aktuator secara real-time.

Abstract: This practicum discusses the implementation of a web-based Internet of Things (IoT) system using the ESP32 microcontroller, MQTT protocol, and HTML interface. The system monitors temperature and humidity in real time using a DHT22 sensor and transmits the data to a public MQTT broker (broker.emqx.io). The data is visualized dynamically on a web dashboard using Highcharts. Additionally, the system allows remote control of an LED's brightness through a slider interface on the web page, where PWM values are published via MQTT and translated into output signals to control the LED on the ESP32. This practicum demonstrates efficient bidirectional communication between the web and IoT devices using MQTT without relying on a dedicated backend or cloud server. The results show that the system accurately displays sensor data and performs real-time actuator control.

Keywords- Internet of Things, ESP32, MQTT, DHT22, Web Dashboard, PWM, Highcharts

1. PENDAHULUAN

1.1 Latar Belakang

Internet of Things (IoT) merupakan konsep di mana perangkat fisik seperti sensor dan aktuator dapat saling terhubung, bertukar data, dan dikendalikan melalui jaringan internet. Perkembangan teknologi IoT memberikan kemudahan dalam membangun sistem monitoring dan kontrol secara real-time, yang dapat diakses dan dikendalikan dari berbagai lokasi.

ESP32 adalah salah satu mikrokontroler populer dalam pengembangan sistem IoT karena telah dilengkapi dengan konektivitas Wi-Fi dan kemampuan pemrosesan yang cukup tinggi. Dalam praktik ini, ESP32 digunakan untuk membaca data suhu dan kelembapan dari sensor DHT22 dan mengirimkan data tersebut ke broker MQTT (Message Queuing Telemetry Transport), yaitu broker.emqx.io, melalui protokol MQTT yang ringan dan efisien. Selain itu, sistem juga dirancang untuk dapat mengontrol intensitas cahaya LED secara jarak jauh menggunakan slider pada antarmuka web yang dibangun dengan HTML dan JavaScript.

Antarmuka web yang dibuat memungkinkan pengguna untuk melihat grafik suhu dan kelembapan secara real-time, serta mengatur nilai PWM (Pulse Width Modulation) yang dikirim melalui MQTT ke ESP32 untuk mengatur kecerahan LED. Pendekatan ini memperlihatkan bagaimana sistem IoT dapat dibangun tanpa harus menggunakan layanan cloud atau server backend seperti Laravel, karena seluruh komunikasi dilakukan melalui broker MQTT publik.

1.2 Tujuan Eksperimen

- Menghubungkan ESP32 ke jaringan Wi-Fi.
- Mengintegrasikan sensor DHT22 untuk membaca data suhu dan kelembapan.
- Mengirimkan data sensor ke broker MQTT secara berkala menggunakan protokol MQTT.
- Membangun antarmuka web berbasis HTML dan Highcharts untuk menampilkan data suhu dan kelembapan secara real-time.
- Mengembangkan kontrol slider pada halaman web untuk mengatur nilai PWM LED.
- Mengimplementasikan komunikasi dua arah antara web dan ESP32 melalui MQTT.
- Memahami konsep komunikasi publish-subscribe menggunakan broker MQTT dalam sistem IoT.

2. METODOLOGI

2.1 Alat dan Bahan

- Laptop
- Visual Studio Code dengan ekstensi PlatformIO
- Wokwi Simulator (untuk simulasi perangkat keras)
- Mikrokontroler ESP32 (dalam simulasi)
- Sensor DHT22 (sensor suhu dan kelembapan)
- LED (merah)
- Kabel jumper virtual (melalui Wokwi)
- Broker MQTT publik (broker.emqx.io)
- Web browser (untuk menjalankan antarmuka index.html)
- File index.html yang menampilkan grafik dan kontrol PWM LED**Langkah Implementasi**

2.2.1 Simulasi Rangkaian di Wokwi

1. Mengakses situs https://wokwi.com dan membuat proyek baru menggunakan board ESP32 DevKit v1.
2. Menambahkan komponen:
   - Sensor DHT22
   - LED (merah)
3. Melakukan wiring virtual:
   DHT22:
   - VCC → 3V3 ESP32
   - GND → GND ESP32
   - Data → GPIO15 (atau GPIO yang ditentukan dalam kode)

   LED:
   - Anoda (A) → GPIO2
   - Katoda (C) → GND
4. Mengatur konfigurasi jaringan Wi-Fi default simulator:
   - SSID: Wokwi-GUEST
   - Password: (kosong)

2.2.2 Implementasi di Visual Studio Code + PlatformIO

1. Membuka Visual Studio Code dan membuat proyek baru menggunakan PlatformIO.
2. Menulis program untuk:
   - Menghubungkan ESP32 ke jaringan Wi-Fi
   - Membaca suhu dan kelembapan dari sensor DHT22
   - Mengirimkan data sensor ke broker MQTT melalui topik IOT/temp dan IOT/hum
   - Menerima data PWM dari topik IOT/mqtt untuk mengatur intensitas LED
   - Mengatur logika LED menyala jika nilai PWM > 128, dan kecerahan berkuang jika ≤ 128, sedangkan LED mati saat nilai PWM 0

2.2.3 Pengembangan dan Pengujian Web Antarmuka

1. Membuat file index.html yang berisi:
   - Grafik suhu dan kelembapan menggunakan Highcharts
   - Slider untuk mengatur intensitas LED
   - Koneksi MQTT menggunakan MQTT.js melalui wss://broker.emqx.io:8084/mqtt

2. Antarmuka akan:
    - Menerima data suhu dari topik IOT/temp dan menampilkannya dalam grafik suhu
    - Menerima data kelembapan dari topik IOT/hum dan menampilkannya dalam grafik kelembapan
    - Mengirim data PWM ke topik IOT/mqtt ketika slider digeser
3. Menjalankan file index.html di browser secara lokal dengan cara:
    - Klik dua kali file atau buka melalui ekstensi Live Server
    - Pastikan koneksi internet aktif agar dapat terhubung ke broker MQTT

2.2.4 Pengujian Sistem Monitoring dan Kontrol via MQTT

1. Menjalankan simulasi ESP32 di Wokwi dan membuka Serial Monitor untuk melihat log koneksi dan data.
2. Membuka index.html di browser dan mengamati tampilan grafik serta slider kontrol.
3. Melihat hasil:
    - Data suhu dan kelembapan tampil secara real-time
    - Nilai PWM dapat dikirim ke ESP32
    - LED menyala/mati tergantung nilai PWM yang dikirim
4. Memastikan komunikasi dua arah MQTT berjalan antara ESP32 dan antarmuka web.

3. HASIL DAN PEMBAHASAN

3.1 Hasil Praktik

Pada praktik ini, sistem berhasil mengimplementasikan pemantauan suhu dan kelembapan serta kontrol intensitas LED berbasis web menggunakan komunikasi protokol MQTT. Sistem terdiri dari mikrokontroler ESP32 yang terhubung ke jaringan Wi-Fi dan berfungsi untuk membaca data suhu dan kelembapan dari sensor DHT22, serta menerima perintah kontrol LED dari antarmuka web yang dikembangkan dalam file index.html.

ESP32 mempublikasikan data sensor ke broker MQTT publik (broker.emqx.io) melalui topik IOT/temp untuk suhu dan IOT/hum untuk kelembapan. Di sisi lain, halaman web yang berjalan di browser melakukan subscribe pada kedua topik tersebut dan menampilkan datanya dalam bentuk grafik secara real-time menggunakan Highcharts.

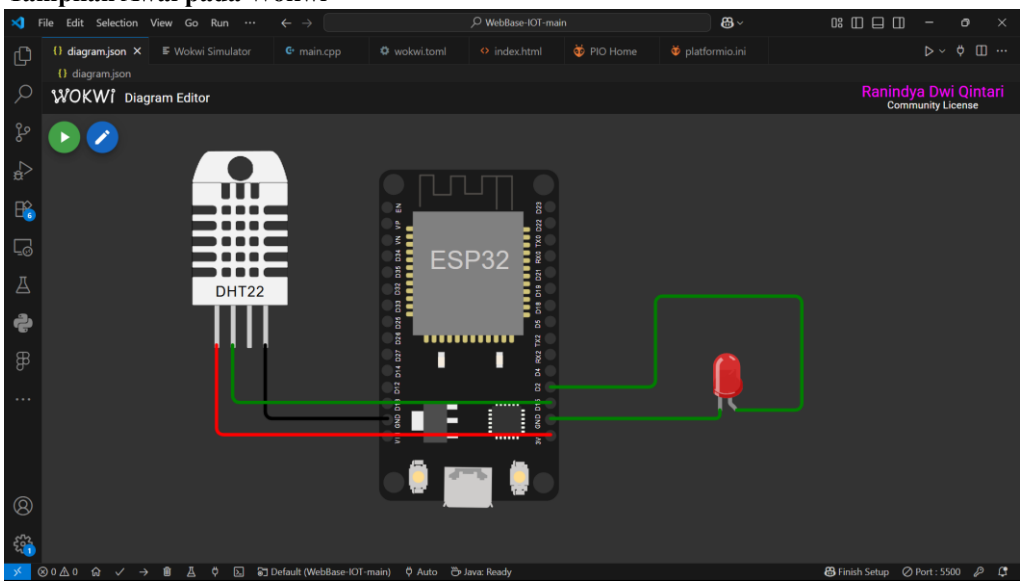
Kontrol LED dilakukan dengan mengatur nilai slider pada halaman web, yang kemudian mengirimkan nilai PWM melalui topik IOT/mqtt. ESP32 melakukan subscribe terhadap topik tersebut dan mengubah intensitas LED sesuai nilai PWM yang diterima. Selain itu, juga diterapkan logika tambahan: LED akan menyala jika nilai PWM > 128, dan mati jika ≤ 128.

Berikut adalah contoh data yang berhasil dibaca dari sensor dan ditampilkan melalui antarmuka web:

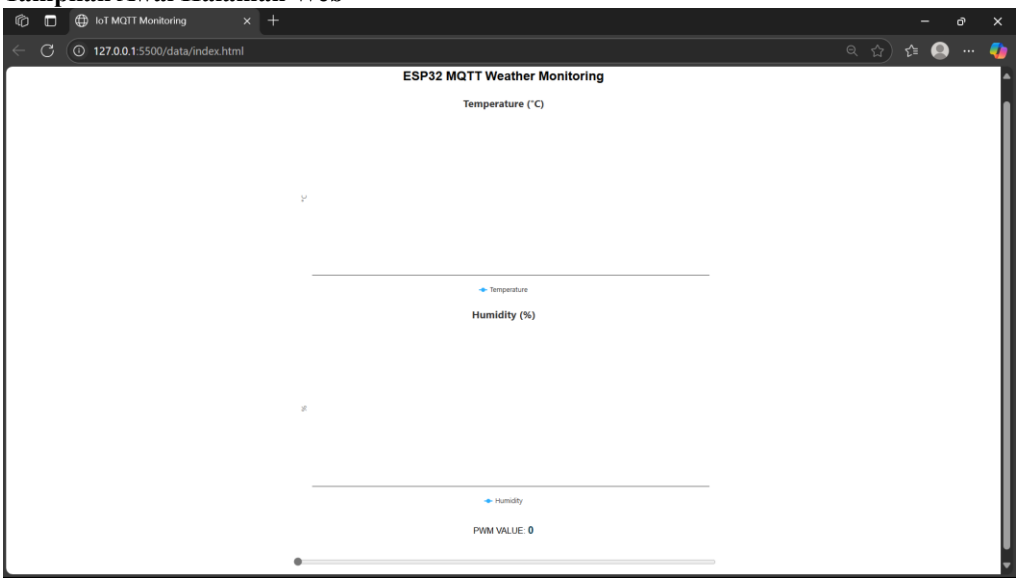
Parameter	Nilai
Suhu	24.00
Kelembapan	40.0
PWM	0

3.2 Tampilan Hasil Praktik

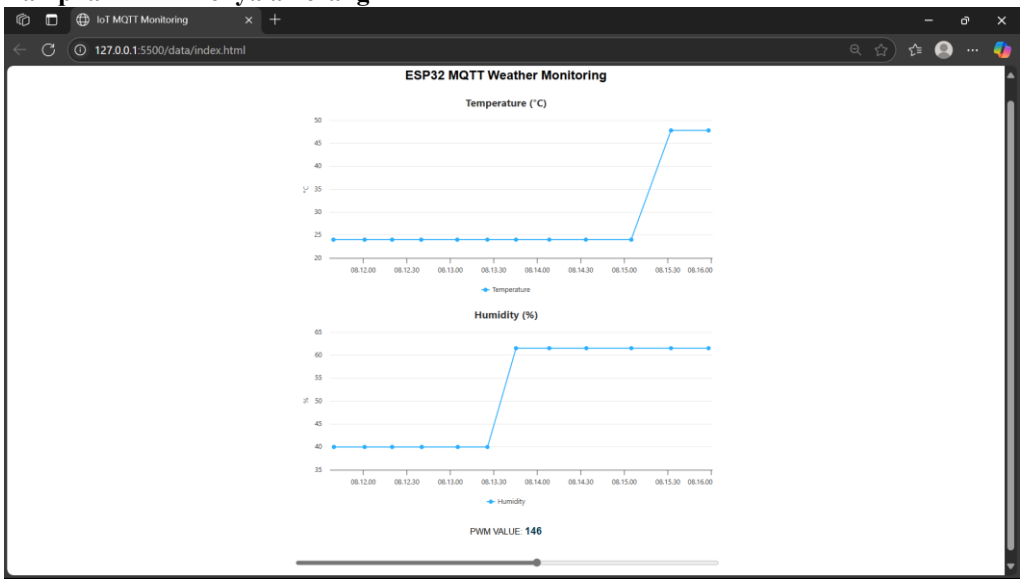
3.2.1 Tampilan Awal pada Wokwi

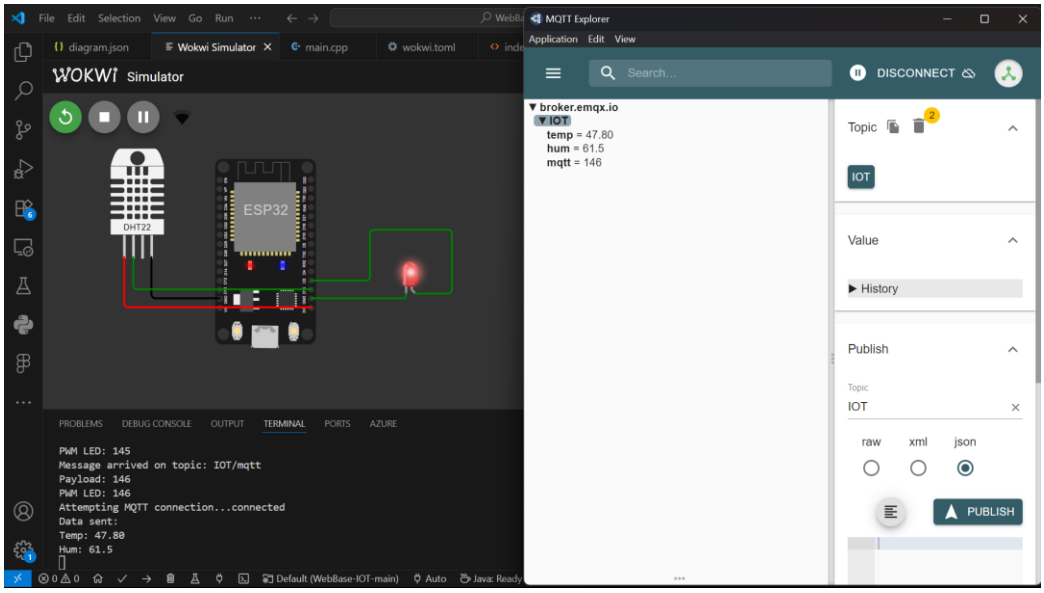


3.2.2 Tampilan Awal Halaman Web

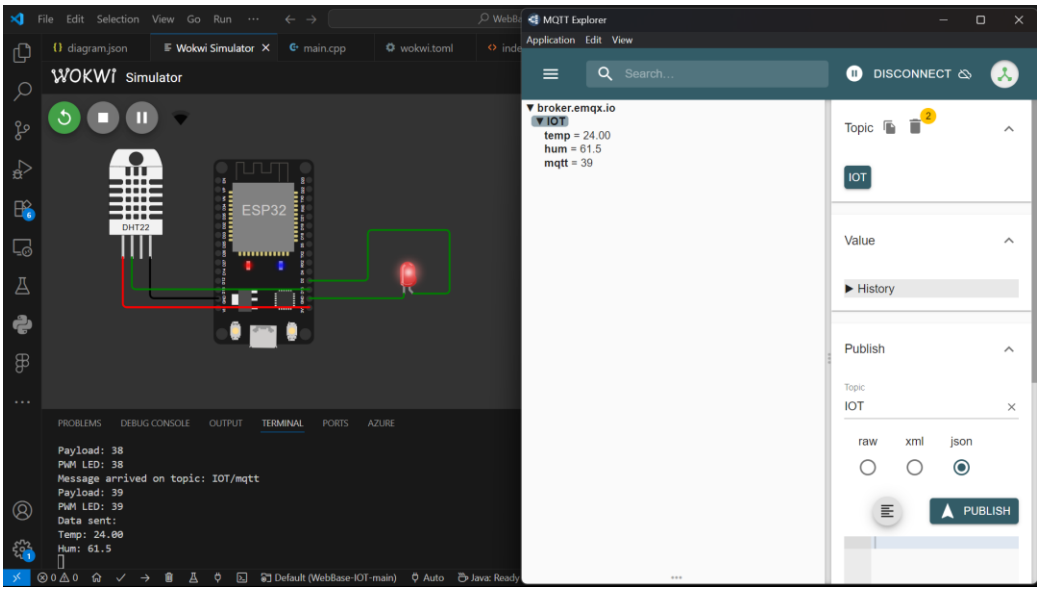


3.2.3 Tampilan LED Menyala Terang

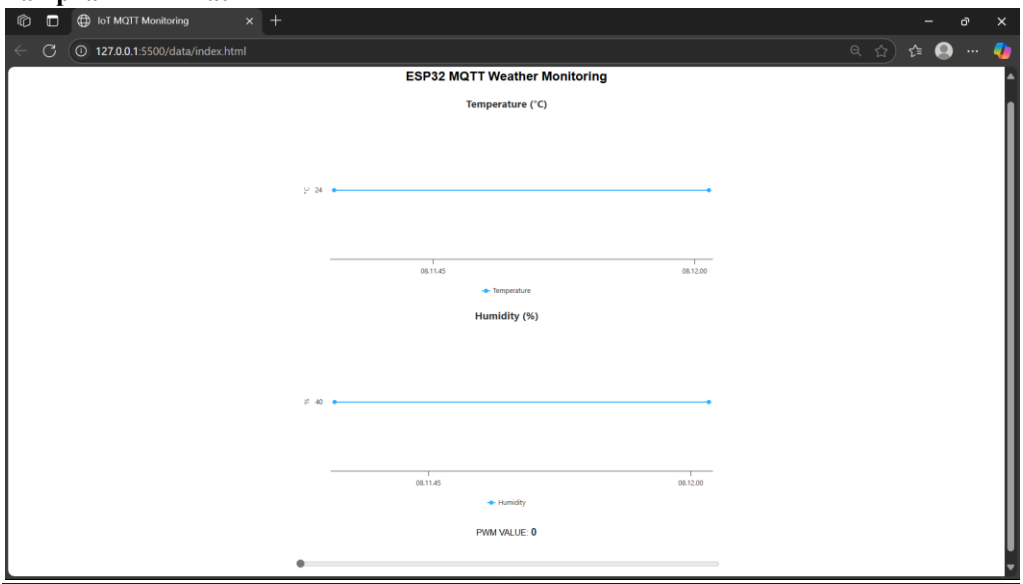


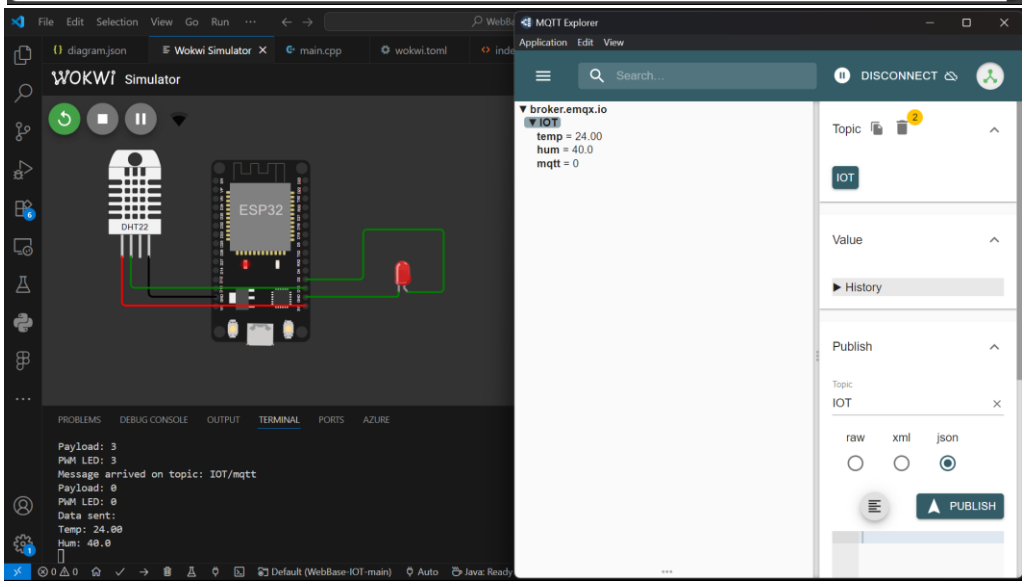


3.2.4 Tampilan LED Redup



3.2.5 Tampilan LED Mati





4. LAMPIRAN

4.1 Kode Program

- **main.cpp**

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

const int LED_RED = 2;       // Pin LED (PWM)
const int DHT_PIN = 15;      // Pin DHT22
DHTesp dht;

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

WiFiClient espClient;
```

```
PubSubClient client(espClient);

unsigned long lastMsg = 0;
int pwmValue = 0;

const int pwmChannel = 0;
const int freq = 5000;
const int resolution = 8;

void setup_wifi() {
  Serial.print("Connecting to WiFi");
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(300);
    Serial.print(".");
  }
  Serial.println("\nWiFi connected!");
}

void callback(char* topic, byte* payload, unsigned int length) {
  Serial.print("Message arrived on topic: ");
  Serial.println(topic);

  String msg;
  for (unsigned int i = 0; i < length; i++) {
    msg += (char)payload[i];
  }
  Serial.print("Payload: ");
  Serial.println(msg);

  if (String(topic) == "IOT/mqtt") {
    pwmValue = msg.toInt();                  // Ubah ke integer
    pwmValue = constrain(pwmValue, 0, 255); // Pastikan dalam range

    ledcWrite(pwmChannel, 255 - pwmValue);

    if (pwmValue > 128) {
      Serial.println("LED: ON (PWM tinggi)");
    } else {
      Serial.println("LED: OFF (PWM rendah)");
    }

    Serial.print("PWM Value: ");
    Serial.println(pwmValue);
  }
}

void reconnect() {
  while (!client.connected()) {
```

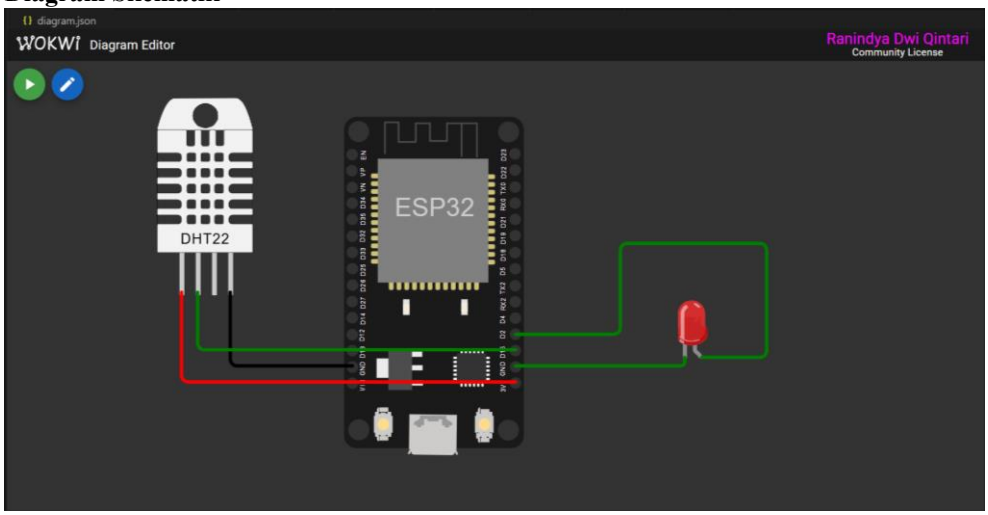
```
    Serial.print("Attempting MQTT connection...");
    String clientId = "esp32-client-" + String(random(0xffff), HEX);
    if (client.connect(clientId.c_str())) {
      Serial.println("Connected to MQTT broker");
      client.subscribe("IOT/mqtt");  // Topic kontrol LED
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      delay(2000);
    }
  }
}

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

  // Setup PWM untuk LED
  ledcSetup(pwmChannel, freq, resolution);
  ledcAttachPin(LED_RED, pwmChannel);

  // Inisialisasi sensor DHT
  dht.setup(DHT_PIN, DHTesp::DHT22);

  // Setup MQTT
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}

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

  unsigned long now = millis();
  if (now - lastMsg > 3000) {
    lastMsg = now;

    TempAndHumidity data = dht.getTempAndHumidity();

    String tempStr = String(data.temperature, 2);
    String humStr = String(data.humidity, 1);

    client.publish("IOT/temp", tempStr.c_str());
    client.publish("IOT/hum", humStr.c_str());

    Serial.println("Data sent:");
    Serial.println("Temp: " + tempStr);
```

```
    Serial.println("Hum: " + humStr);
  }
}
```

- **diagram.json**

```
{
  "version": 1,
  "author": "Ranindya Dwi Qintari",
  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": 0, "left": 0, "attrs": {} },
    { "type": "wokwi-dht22", "id": "dht1", "top": -9.3, "left": -111, "attrs": {} },
    { "type": "wokwi-led", "id": "led1", "top": 102, "left": 186.2, "attrs": { "color":
"red" } }
  ],
  "connections": [
    [ "esp:TX0", "$serialMonitor:RX", "", [] ],
    [ "esp:RX0", "$serialMonitor:TX", "", [] ],
    [ "dht1:GND", "esp:GND.2", "black", [ "v0" ] ],
    [ "dht1:VCC", "esp:3V3", "red", [ "v0" ] ],
    [ "dht1:SDA", "esp:D15", "green", [ "v0" ] ],
    [ "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 yang telah dilakukan, dapat disimpulkan bahwa mikrokontroler ESP32 berhasil digunakan untuk membangun sistem pemantauan suhu dan kelembapan serta pengendalian intensitas LED secara real-time melalui komunikasi MQTT. Sistem ini membaca data dari sensor DHT22 dan mempublikasikan data ke broker MQTT broker.emqx.io, kemudian antarmuka web menampilkan grafik suhu dan kelembapan berdasarkan data yang diterima melalui protokol MQTT.

Selain itu, sistem juga memungkinkan pengguna untuk mengontrol intensitas LED dari halaman web eksternal (index.html) menggunakan elemen slider. Nilai PWM dikirim ke ESP32 melalui topik MQTT dan langsung digunakan untuk mengatur kecerahan LED secara responsif.

Berbeda dengan pendekatan web server lokal, sistem ini tidak menggunakan ESPAsyncWebServer atau SPIFFS, melainkan memanfaatkan komunikasi MQTT penuh, yang menjadikannya lebih fleksibel dan dapat diakses dari berbagai perangkat selama terkoneksi ke broker MQTT dan internet.

Praktikum ini berhasil mengilustrasikan konsep dasar Web-Based IoT menggunakan MQTT, meliputi:

- Pengumpulan data sensor secara berkala dari mikrokontroler.
- Pengiriman dan penerimaan data melalui protokol MQTT.
- Visualisasi data dan kontrol perangkat melalui antarmuka web yang berjalan di browser.