LAPORAN PRAKTIKUM
MATA KULIAH INTERNET OF THINGS

Monitoring Suhu dan Kelembaban Berbasis ESP32, DHT22, MQTT, dan Dashboard Website Interaktif

Dosen Pengampu :
Ir. Subairi, ST., MT., IPM

Disusun Oleh :

Desi Eka Mardiani
233140707111084

Fakultas Vokasi, Universitas Brawijaya
Email : desiekaa71@student.ub.ac.id

Laporan Praktikum Mata Kuliah Internet of Things

Monitoring Suhu dan Kelembaban Berbasis ESP32, DHT22, MQTT, dan Dashboard Website Interaktif

Desi Eka Mardiani

Program Studi Teknologi Informasi, Fakultas Vokasi, Universitas Brawijaya

Abstrak : Praktikum ini bertujuan merancang sistem Internet of Things (IoT) untuk monitoring suhu dan kelembaban secara real-time menggunakan sensor DHT22 yang terhubung ke mikrokontroler ESP32. Data sensor dikirimkan melalui protokol MQTT ke broker publik broker.emqx.io, sehingga dapat dipantau secara remote. Selain menggunakan aplikasi MQTT Explorer, pada praktikum ini juga dikembangkan dashboard website interaktif yang mampu menampilkan data suhu dan kelembaban secara real-time dalam bentuk angka, grafik, dan gauge, serta menyediakan fitur kontrol LED ESP32 langsung dari web. Sistem ini melatih pemahaman konsep IoT, pemrograman mikrokontroler, komunikasi MQTT, serta integrasi sensor, jaringan, dan visualisasi data berbasis web. Dengan sistem ini, peserta dapat memahami cara menghubungkan sensor dengan mikrokontroler, mengirim data secara real-time, memantau kondisi lingkungan dari jarak jauh, serta mengendalikan perangkat IoT melalui website.

Kata Kunci : IoT, ESP32, DHT22, MQTT, dashboard web, sensor suhu, kelembaban, monitoring real-time, kontrol LED.

Abstract : *This practicum aims to design an Internet of Things (IoT) system for real-time temperature and humidity monitoring using a DHT22 sensor connected to an ESP32 microcontroller. Sensor data is sent via the MQTT protocol to the broker.emqx.io public broker, so that it can be monitored remotely. In addition to using the MQTT Explorer application, this practicum also developed an interactive website dashboard that is able to display temperature and humidity data in real-time in the form of numbers, graphs, and gauges, and provides an ESP32 LED control feature directly from the web. This system trains the understanding of IoT concepts, microcontroller programming, MQTT communication, and web-based sensor, network, and data visualization integration. With this system, participants can understand how to connect sensors to microcontrollers, send data in real-time, monitor environmental conditions remotely, and control IoT devices via the website.*

1. PENDAHULUAN

1.1. Latar Belakang

Internet of Things (IoT) merupakan konsep teknologi yang menghubungkan berbagai perangkat fisik ke jaringan internet, sehingga memungkinkan perangkat-perangkat tersebut untuk saling bertukar data secara otomatis dan real-time tanpa campur tangan manusia secara langsung. Dengan adanya konektivitas ini, perangkat-perangkat IoT dapat saling berkomunikasi dan berkolaborasi untuk menjalankan berbagai fungsi cerdas yang sangat bermanfaat dalam kehidupan sehari-hari.

Salah satu aplikasi IoT yang sangat penting dan banyak digunakan adalah dalam bidang monitoring lingkungan. Monitoring ini meliputi pengukuran dan pemantauan kondisi-kondisi lingkungan seperti suhu, kelembaban udara, kualitas udara, tekanan atmosfer, dan parameter lainnya yang sangat krusial untuk berbagai sektor, termasuk pertanian, kesehatan, dan pengelolaan gedung pintar (smart building). Misalnya, dalam bidang pertanian, data suhu dan kelembaban yang akurat membantu petani mengoptimalkan irigasi dan menjaga kesehatan tanaman. Di bidang kesehatan, pemantauan lingkungan dapat meningkatkan kualitas udara di rumah sakit atau ruang perawatan. Sedangkan di gedung pintar, data lingkungan digunakan untuk mengatur sistem pendingin udara dan pencahayaan secara otomatis demi kenyamanan dan efisiensi energi.

Untuk mengimplementasikan sistem monitoring lingkungan berbasis IoT, diperlukan perangkat sensor yang mampu mengukur parameter lingkungan secara akurat dan mikrokontroler yang dapat mengolah serta mengirimkan data tersebut ke internet. Sensor DHT22 adalah salah satu sensor populer yang banyak digunakan untuk mengukur suhu dan kelembaban udara. Sensor ini memiliki keunggulan dalam hal akurasi yang baik, harga yang terjangkau, serta kemudahan integrasi dengan berbagai mikrokontroler.

Sebagai mikrokontroler, ESP32 menjadi pilihan ideal karena dilengkapi dengan modul WiFi terintegrasi yang memungkinkan perangkat untuk langsung terhubung ke jaringan internet tanpa memerlukan modul tambahan. Selain itu, ESP32 memiliki performa yang cukup tinggi, konsumsi daya yang efisien, serta dukungan berbagai

protokol komunikasi, sehingga sangat cocok untuk aplikasi IoT yang membutuhkan konektivitas dan pemrosesan data secara real-time.

1.2. Tujuan Praktikum

- Memahami konsep komunikasi IoT menggunakan protokol MQTT (publish/subscribe).
- Mengintegrasikan sensor DHT22 dengan ESP32 untuk membaca data suhu dan kelembaban.
- Mengirim data sensor secara periodik ke broker MQTT publik.
- Mengembangkan dashboard website untuk monitoring real-time dan kontrol perangkat (LED) dari web.
- Melatih kemampuan pemrograman mikrokontroler, komunikasi data, dan pengembangan aplikasi web IoT.

2. METODOLOGI

2.1. Alat dan Bahan

Praktikum ini dilaksanakan menggunakan platform simulasi Wokwi, yang membantu melakukan pengujian dan pengembangan sistem IoT secara virtual tanpa memerlukan perangkat keras fisik. Dalam simulasi ini, perangkat keras yang digunakan meliputi mikrokontroler ESP32 dan sensor DHT22 untuk pengukuran suhu serta kelembaban udara. ESP32 dipilih karena memiliki kemampuan WiFi terintegrasi yang memudahkan pengiriman data secara nirkabel, sementara sensor DHT22 dikenal memiliki akurasi yang baik dan harga yang terjangkau, sehingga sangat cocok untuk aplikasi monitoring lingkungan. Data yang diperoleh dari sensor DHT22 kemudian dikirimkan secara periodik melalui protokol MQTT ke broker MQTT publik, yaitu broker.emqx.io. Penggunaan broker publik ini memberikan kemudahan akses data secara real-time dari lokasi manapun tanpa perlu membangun infrastruktur server sendiri.

Untuk memantau data yang dikirim, praktikum ini juga memanfaatkan aplikasi MQTT Explorer, sebuah aplikasi yang memudahkan visualisasi dan monitoring data MQTT secara langsung dan interaktif. Selain itu, dashboard website interaktif yang dikembangkan menggunakan platform OneCompiler menampilkan berbagai fitur penting seperti kontrol lampu LED secara remote, grafik (chart) dan gauge untuk memvisualisasikan data suhu dan kelembaban, serta riwayat data dalam bentuk grafik

yang membantu analisis tren lingkungan secara lebih mendalam. Dengan kombinasi platform simulasi Wokwi, mikrokontroler ESP32, sensor DHT22, broker MQTT publik, aplikasi MQTT Explorer, dan dashboard OneCompiler, praktikum ini memberikan pengalaman belajar yang lengkap dan komprehensif mulai dari pengukuran sensor, pengiriman data, hingga monitoring dan kontrol perangkat secara real-time dalam ekosistem IoT yang sesungguhnya.

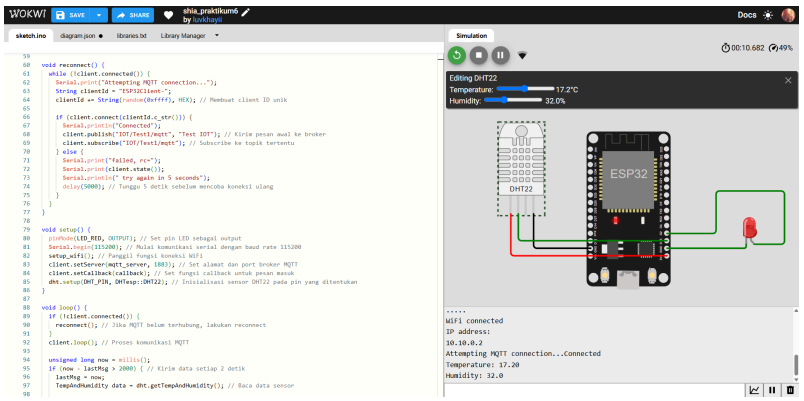
2.2. Langkah Implementasi

A. Simulasi di Wokwi

Dalam praktikum ini, perangkat keras tidak menggunakan komponen fisik, melainkan simulasi di platform Wokwi. Komponen yang digunakan meliputi:

- Mikrokontroler ESP32 sebagai otak sistem yang memiliki fitur WiFi terintegrasi.
- Sensor DHT22 untuk mengukur suhu dan kelembaban. Sensor ini terhubung ke pin GPIO 15 ESP32 pada simulasi.
- LED merah sebagai indikator output yang terhubung ke pin GPIO 2 ESP32.

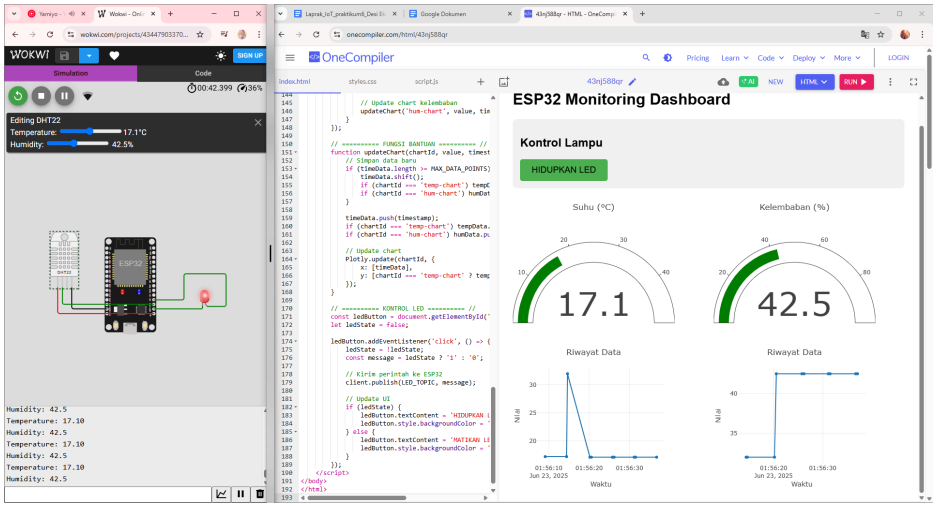
Koneksi antar komponen sudah diatur dalam file konfigurasi JSON Wokwi, sehingga simulasi dapat berjalan tanpa perakitan fisik.



B. Pengembangan Dashboard Website Interaktif

Pengembangan dashboard website interaktif dalam praktikum ini dilakukan menggunakan HTML dan JavaScript, dengan memanfaatkan pustaka MQTT.js untuk menghubungkan website secara langsung ke broker MQTT melalui protokol WebSocket. Dengan pendekatan ini, dashboard mampu melakukan komunikasi real-time dengan perangkat IoT tanpa perlu refresh halaman secara manual.

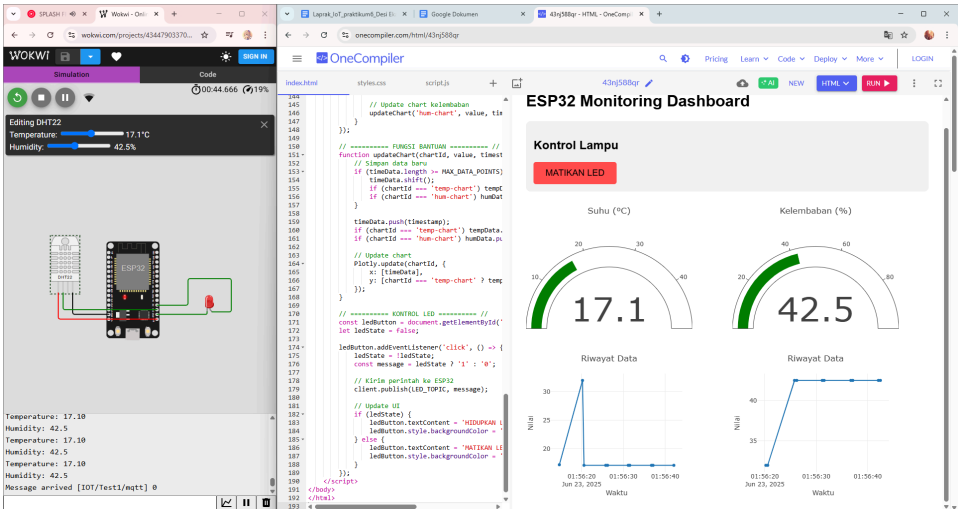
Fitur utama dashboard meliputi gauge yang menampilkan data suhu dan kelembaban secara langsung, serta chart yang memperlihatkan grafik riwayat perubahan data tersebut dari waktu ke waktu, sehingga memudahkan pemantauan tren lingkungan secara visual. Selain itu, terdapat fitur kontrol LED berupa tombol on/off yang memungkinkan pengguna mengendalikan LED yang terhubung ke ESP32 secara remote melalui antarmuka web.



3. PEMBAHASAN

3.1. Hasil

- ESP32 berhasil terhubung ke jaringan WiFi simulasi dan broker MQTT publik. Sensor DHT22 mampu membaca data suhu dan kelembaban dengan baik. Data sensor berhasil dikirim ke broker MQTT dan dapat di-subscribe dari aplikasi MQTT Explorer maupun dashboard website.



- Dashboard website mampu menampilkan data suhu dan kelembaban secara real-time dalam bentuk angka, gauge, dan grafik.
- Fitur kontrol LED dari website berjalan lancar: tombol pada web mengirim perintah on/off ke ESP32 melalui MQTT, dan LED menyala/mati sesuai perintah.
- Sistem dapat diakses dan dikontrol dari mana saja selama terhubung internet.

3.2. Pembahasan

Praktikum ini membuktikan bahwa ESP32 dapat berfungsi sebagai node IoT yang mengumpulkan dan mengirim data sensor ke broker MQTT secara efisien. Dengan penambahan dashboard website interaktif, sistem menjadi lebih user-friendly dan powerful, karena pengguna dapat :

- Melihat data sensor secara visual (gauge & chart)
- Melakukan kontrol perangkat (LED) secara remote
- Memantau data dari berbagai perangkat (PC, HP) tanpa aplikasi tambahan

Penggunaan MQTT sebagai protokol komunikasi sangat efisien dan fleksibel untuk aplikasi IoT. Dashboard berbasis web juga sangat mudah dikembangkan dan diakses, sehingga sangat cocok untuk aplikasi monitoring dan kontrol jarak jauh.

4. KESIMPULAN

Sistem monitoring suhu dan kelembaban berbasis ESP32 dan sensor DHT22 berhasil diimplementasikan dan diuji menggunakan simulasi di platform Wokwi. Data sensor dapat dikirim secara real-time ke broker MQTT publik dan dimonitor melalui dashboard website interaktif yang mendukung visualisasi data (gauge, chart) serta kontrol perangkat (LED) secara remote. Praktikum ini efektif dalam memperdalam pemahaman konsep dasar IoT, pemrograman mikrokontroler, komunikasi data dengan MQTT, serta pengembangan aplikasi web IoT.

Keberhasilan praktikum ini membuka peluang pengembangan sistem IoT yang lebih kompleks, seperti penambahan sensor lain, fitur notifikasi, atau integrasi dengan platform cloud untuk aplikasi monitoring dan kontrol yang lebih luas.

LAMPIRAN

Kode HTML

```
<!DOCTYPE html>
<html>
<head>
    <title>ESP32 Dashboard</title>
                                                                <script
src="https://cdnjs.cloudflare.com/ajax/libs/mqtt/4.3.7/mqtt.min.js"></s
cript>
    <script src="https://cdn.plot.ly/plotly-latest.min.js"></script>
    <style>
        body { font-family: Arial, sans-serif; padding: 20px; }
        .dashboard {
            display: grid;
            grid-template-columns: 1fr 1fr;
            gap: 20px;
        }
        .gauge, .chart { height: 300px; }
        #led-control {
            padding: 15px;
            background: #f0f0f0;
            border-radius: 10px;
            margin-bottom: 20px;
        }
        #led-button {
            padding: 12px 24px;
            font-size: 18px;
            cursor: pointer;
            border: none;
            border-radius: 5px;
            transition: all 0.3s;
        }
    </style>
</head>
<body>
    <h1>ESP32 Monitoring Dashboard</h1>

    <!-- Kontrol LED -->
    <div id="led-control">
        <h2>Kontrol Lampu</h2>
```

```
                    <button   id="led-button"   style="background-color:
#ff4d4d;">MATIKAN LED</button>
    </div>

    <!-- Gauges -->
    <div class="dashboard">
        <div id="temp-gauge" class="gauge"></div>
        <div id="hum-gauge" class="gauge"></div>
    </div>

    <!-- Charts -->
    <div class="dashboard">
        <div id="temp-chart" class="chart"></div>
        <div id="hum-chart" class="chart"></div>
    </div>

    <script>
        // ========== KONFIGURASI MQTT ========== //
        const client = mqtt.connect('wss://broker.emqx.io:8084/mqtt', {
                                        clientId:   'web-dashboard-'   +
Math.random().toString(16).substr(2, 8)
        });

        // Topik sesuai kode Arduino
        const TEMP_TOPIC = 'IOT/Test1/temp';
        const HUM_TOPIC = 'IOT/Test1/hum';
        const LED_TOPIC = 'IOT/Test1/mqtt';

        // ========== VARIABEL DATA ========== //
        let tempData = [];
        let humData = [];
        let timeData = [];
        const MAX_DATA_POINTS = 20; // Maksimal data di chart

        // ========== INISIALISASI PLOTLY ========== //
        // Layout gauge suhu
        const tempGaugeLayout = {
            title: 'Suhu (°C)',
            margin: { t: 50, b: 0, l: 0, r: 0 }
        };

        // Layout gauge kelembaban
        const humGaugeLayout = {
```

```
    title: 'Kelembaban (%)',
    margin: { t: 50, b: 0, l: 0, r: 0 }
};

// Layout chart
const chartLayout = {
    title: 'Riwayat Data',
    xaxis: { title: 'Waktu' },
    yaxis: { title: 'Nilai' },
    margin: { t: 50, l: 50, r: 20, b: 50 }
};

// Inisialisasi plot
Plotly.newPlot('temp-gauge', [{
    type: "indicator",
    mode: "gauge+number",
    value: 0,
    gauge: { axis: { range: [0, 50] } }
}], tempGaugeLayout);

Plotly.newPlot('hum-gauge', [{
    type: "indicator",
    mode: "gauge+number",
    value: 0,
    gauge: { axis: { range: [0, 100] } }
}], humGaugeLayout);

Plotly.newPlot('temp-chart', [{
    x: [],
    y: [],
    type: 'scatter',
    name: 'Suhu'
}], chartLayout);

Plotly.newPlot('hum-chart', [{
    x: [],
    y: [],
    type: 'scatter',
    name: 'Kelembaban'
}], chartLayout);

// ========== HANDLER MQTT ========== //
client.on('connect', () => {
```

```
    console.log('Terhubung ke broker MQTT');
    client.subscribe(TEMP_TOPIC);
    client.subscribe(HUM_TOPIC);
});

client.on('message', (topic, message) => {
    const value = parseFloat(message.toString());
    const timestamp = new Date();

    if (topic === TEMP_TOPIC) {
        // Update gauge suhu
        Plotly.update('temp-gauge', {
            value: [value]
        });

        // Update chart suhu
        updateChart('temp-chart', value, timestamp);

    } else if (topic === HUM_TOPIC) {
        // Update gauge kelembaban
        Plotly.update('hum-gauge', {
            value: [value]
        });

        // Update chart kelembaban
        updateChart('hum-chart', value, timestamp);
    }
});

// ========== FUNGSI BANTUAN ========== //
function updateChart(chartId, value, timestamp) {
    // Simpan data baru
    if (timeData.length >= MAX_DATA_POINTS) {
        timeData.shift();
        if (chartId === 'temp-chart') tempData.shift();
        if (chartId === 'hum-chart') humData.shift();
    }

    timeData.push(timestamp);
    if (chartId === 'temp-chart') tempData.push(value);
    if (chartId === 'hum-chart') humData.push(value);

    // Update chart
```

```
            Plotly.update(chartId, {
                x: [timeData],
                y: [chartId === 'temp-chart' ? tempData : humData]
            });
        }

        // ========== KONTROL LED ========== //
        const ledButton = document.getElementById('led-button');
        let ledState = false;

        ledButton.addEventListener('click', () => {
            ledState = !ledState;
            const message = ledState ? '1' : '0';

            // Kirim perintah ke ESP32
            client.publish(LED_TOPIC, message);

            // Update UI
            if (ledState) {
                ledButton.textContent = 'HIDUPKAN LED';
                ledButton.style.backgroundColor = '#4CAF50';
            } else {
                ledButton.textContent = 'MATIKAN LED';
                ledButton.style.backgroundColor = '#ff4d4d';
            }
        });
    </script>
</body>
</html>
```