**PRAKTIKUM SIMULASI MONITORING SUHU DAN KELEMBAPAN DENGAN NODE-RED DAN INFLUXDB DALAM MATA KULIAH INTERNET OF THINGS**

**(IOT)**



Muhammad Fa’iz Ramadhan

Fakultas Vokasi, Universitas Brawijaya

[faizramadhan23@student.ub.ac.id](mailto:faizramadhan23@student.ub.ac.id)

**Abstrak**

Dengan menggunakan sistem monitoring suhu dan kelembapan berbasis Internet of Things (IoT), kondisi lingkungan dapat dipantau secara historis dan secara real-time. Dalam praktikum ini, platform Node-RED mengatur alur data mulai dari input, pemrosesan, penyimpanan ke InfluxDB, dan visualisasi melalui grafik dan gauge. Selanjutnya, Node-RED mengatur pemrosesan data, penyimpanan ke InfluxDB, dan visualisasi melalui Dashboard UI. Hasil praktik menunjukkan bahwa sistem dapat mengumpulkan, menyimpan, dan menyajikan informasi suhu dan kelembapan secara historis dan real-time dengan cara yang optimal.

**Kata kunci**: IoT, Node-RED, InfluxDB, suhu, kelembapan, dashboard, time-series

1. **Pendahuluan**
   1. **Latar belakang**

Dalam era Internet of Things, data lingkungan seperti suhu dan kelembapan sangat penting. Analisis tren cuaca dan pemantauan saat ini dipermudah dengan penyimpanan data historis. Node-RED menawarkan alur pemrograman visual untuk membaca dan memproses data, sedangkan InfluxDB adalah database rangkaian waktu yang dimaksudkan untuk menyimpan data berbasis waktu, seperti suhu dan kelembapan. Kombinasi keduanya memastikan sistem pengawasan yang kuat dan efektif.

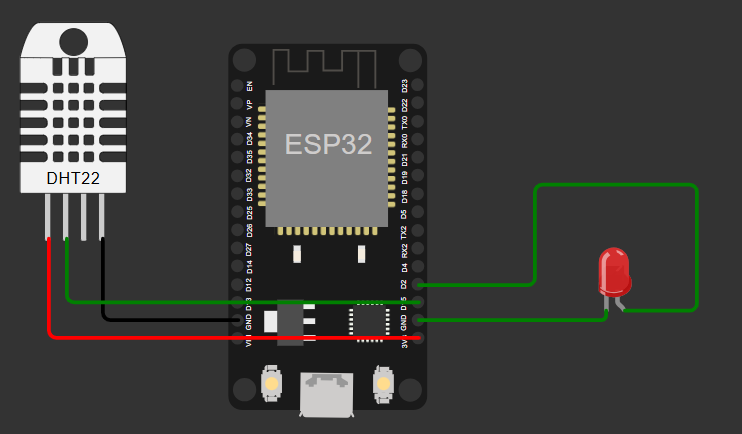
* 1. **Tujuan eksperimen**

Tujuan dilakukannya praktikum ini adalah :

1. Menyimulasikan pengumpulan data suhu dan kelembapan
2. Menyimpan data tersebut ke dalam InfluxDB
3. Menampilkan data secara real-time dan historis melalui dashboard
4. Menggunakan Node-RED sebagai orchestrator antar komponen
5. Memverifikasi fungsionalitas sistem melalui simulasi di platform Wokwi.
6. Menganalisis performa dan akurasi sistem dalam kondisi simulasi.
7. **Metodologi**
   1. **Alat dan Bahan**
8. Mikrokontroler ESP32
9. Sensor DHT22 (temperatur dan kelembapan).
10. Koneksi Wi-Fi (Wokwi Guest)
11. node-red-dashboard (untuk UI)
12. node-red-contrib-influxdb (untuk koneksi ke InfluxDB)
13. function, inject, gauge, chart, dll
    1. **Langkah Implempentasi**
14. tulis kode program untuk membaca data suhu dan kelembapan, lalu kirimkan via WiFi ke Node-RED (gunakan protokol MQTT atau HTTP).
15. Mengunggah dan menjalankan program di ESP32.
16. Klik **"Start Simulation"** untuk memulai simulasi.
17. Jalankan Node-RED di laptop
18. Gunakan node **MQTT in**
19. Gunakan node **function** untuk parsing data suhu dan kelembapan.
20. Gunakan node **influxdb out** untuk menyimpan data ke InfluxDB.
21. Gunakan node **ui\_gauge** atau **ui\_chart** untuk menampilkan data suhu & kelembapan secara real-time.
22. Install dan jalankan InfluxDB
23. Buat database baru
24. Konfigurasikan node InfluxDB di Node-RED
25. Pastikan data dari Node-RED masuk ke InfluxDB.
26. Melakukan debugging jika terjadi kesalahan dalam program atau koneksi perangkat keras.
27. Mengamati hasil dan mencatat performa sistem selama siklus operasional berlangsung.
28. **Hasil dan Pembahasan**

Wokwi berfungsi sebagai simulator mikrokontroler ESP32 yang terhubung dengan sensor DHT22 untuk melacak data suhu dan kelembapan secara real-time dalam proyek ini. Selanjutnya, data dikirim ke Node-RED, sebuah platform berbasis aliran (flow-based) yang memungkinkan integrasi antarperangkat IoT tanpa menulis banyak kode. Di dalam Node-RED, data ditampilkan secara visual melalui dashboard dan disimpan ke InfluxDB, sebuah database time-series yang dikhususkan untuk merekam data sensor dalam interval waktu tertentu. Kombinasi ketiga ini menawarkan solusi simulasi Internet of Things yang efektif dan berguna tanpa memerlukan perangkat keras fisik.

* 1. **Hasil Eksperimen**



* 1. **Kode Program**

#include <WiFi.h>

#include <PubSubClient.h>

#include <DHTesp.h>

const int LED\_RED = 2;

const int DHT\_PIN = 15;

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;

float temp = 0;

float hum = 0;

void setup\_wifi() {

delay(10);

Serial.println();

Serial.print("Connecting to ");

Serial.println(ssid);

WiFi.mode(WIFI\_STA);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

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) {

Serial.print("Message arrived [");

Serial.print(topic);

Serial.print("] ");

for (int i = 0; i < length; i++) {

Serial.print((char)payload[i]);

}

Serial.println();

if ((char)payload[0] == '1') {

digitalWrite(LED\_RED, HIGH);

} else {

digitalWrite(LED\_RED, LOW);

}

}

void reconnect() {

while (!client.connected()) {

Serial.print("Attempting MQTT connection...");

String clientId = "ESP32Client-";

clientId += String(random(0xffff), HEX);

if (client.connect(clientId.c\_str())) {

Serial.println("Connected");

client.publish("IOT/Test1/mqtt", "Test IOT");

client.subscribe("IOT/Test1/mqtt");

} else {

Serial.print("failed, rc=");

Serial.print(client.state());

Serial.println(" try again in 5 seconds");

delay(5000);

}

}

}

void setup() {

pinMode(LED\_RED, OUTPUT);

Serial.begin(115200);

setup\_wifi();

client.setServer(mqtt\_server, 1883);

client.setCallback(callback);

dht.setup(DHT\_PIN, DHTesp::DHT22);

}

void loop() {

if (!client.connected()) {

reconnect();

}

client.loop();

unsigned long now = millis();

if (now - lastMsg > 2000) {

lastMsg = now;

TempAndHumidity data = dht.getTempAndHumidity();

String temp = String(data.temperature, 2);

client.publish("IOT/Test1/temp", temp.c\_str());

String hum = String(data.humidity, 1);

client.publish("IOT/Test1/hum", hum.c\_str());

Serial.print("Temperature: ");

Serial.println(temp);

Serial.print("Humidity: ");

Serial.println(hum);

}

}