**LAPORAN PRAKTIKUM**

**IOT MINGGU KE-8**

**TEMPERATURE AND HUMIDITY MONITORING SIMULATION BASED ON IOT USING ESP32 AND NODERED**

****

**Dosen Pengampu :**

**Ir. Subairi, ST., MT., IPM**

**Disusun Oleh:**

**Muhammad Alif Aris**

**(233140707111077)**

**PROGRAM STUDI TEKNOLOGI INFORMASI**

**FAKULTAS VOKASI**

**UNIVERSITAS BRAWIJAYA**

**2025**

**Abstrak**

Solusi modern untuk pengelolaan lingkungan adalah pemantauan suhu dan kelembapan berbasis Internet of Things (IoT). Eksperimen ini menggunakan ESP32 sebagai mikrokontroler untuk membaca data sensor DHT22. Untuk visualisasi dalam bentuk grafik dan indikator, data dikirim ke platform Node-RED melalui protokol MQTT. Eksperimen simulasi dilakukan pada platform Wokwi, yang memungkinkan pengujian sistem tanpa perangkat fisik. Hasilnya menunjukkan bahwa sistem memiliki kemampuan untuk mengumpulkan dan mengirim data secara real-time dengan tingkat akurasi yang tinggi, serta kemampuan untuk memperbaiki koneksi Wi-Fi dengan cepat. Node-RED membuat sistem ini menjadi solusi murah dan efisien untuk aplikasi Internet of Things seperti rumah pintar, pertanian, dan pengelolaan energi. Ini karena ia memudahkan pengaturan alur kerja dan visualisasi data.

*Keywords - ESP32, DHT22, Node-RED, IoT, Protokol MQTT, Pemantauan Lingkungan.*

**BAB I**

**PENDAHULUAN**

**1.1 Latar Belakang**

Teknologi Internet of Things (IoT) telah membuat pengelolaan data lingkungan menjadi lebih mudah di zaman modern. Salah satu teknologi utama yang digunakan adalah mikrokontroler ESP32, yang dapat mengumpulkan data suhu dan kelembapan melalui sensor dan kemudian mengirimkannya ke platform visualisasi seperti Node-RED. Kombinasi ESP32 dan Node-RED menghasilkan sistem monitoring yang menggabungkan data suhu dan kelembapan. Aplikasi seperti manajemen energi dan rumah pintar mendapat manfaat besar dari sistem ini, yang memungkinkan pemantauan jarak jauh yang cepat dan akurat. Node-RED juga membantu pengguna membuat alur kerja otomatis.

**1.2 Tujuan Eksperimen**

1. Mengaktifkan integrasi ESP32 dan Node-RED untuk melacak suhu dan kelembapan dalam waktu nyata.
2. Mengevaluasi bagaimana sistem mengirim dan menampilkan data sensor melalui platform Internet of Things berbasis Node-RED.
3. Memverifikasi bahwa ESP32 adalah solusi yang hemat energi untuk aplikasi pemantauan lingkungan berbasis IoT.

**BAB II**

**METODOLOGI**

**2.1 Alat dan Bahan**

* SP32 DevKit V1
* Sensor DHT22
* Breadboard
* Kabel Jumper (Virtual)
* Wokwi
* Koneksi Internet
* Software Arduino IDE atau editor yang kompatibel
* Pustaka DHTesp untuk pengolahan data sensor
* Platform Node-RED untuk visualisasi data.

**2.3 Implementasi Sistem**

1. Di breadboard virtual Wokwi, hubungkan ESP32 dengan sensor DHT22 menggunakan pin VCC, GND, dan GPIO untuk data.
2. Dengan Arduino IDE, program ESP32 dapat membaca data suhu dan kelembapan DHT22.
3. Untuk mengirim data ke dashboard Node-RED, masukkan koneksi Wi-Fi ke kode.
4. Untuk menerima data melalui protokol MQTT atau HTTP dan menampilkannya pada grafik atau indikator, integrasikan Node-RED.
5. Untuk memvalidasi data yang diterima di dashboard Node-RED, lakukan simulasi di Wokwi.

**BAB III**

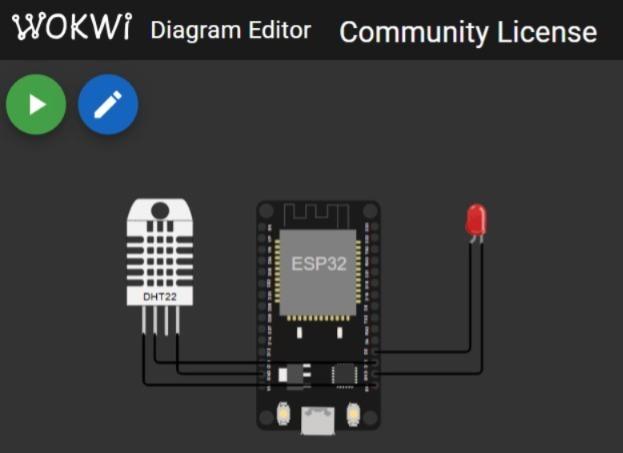
**HASIL DAN PEMBAHASAN**

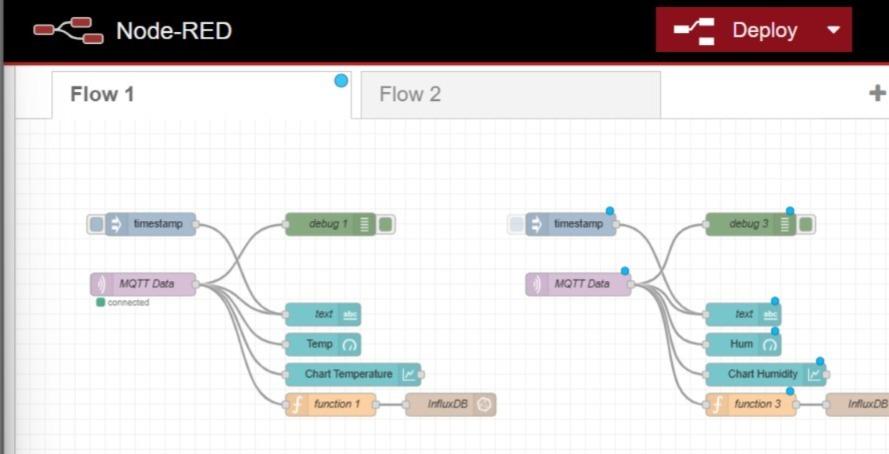
**3.1 Hasil Eksperimen**

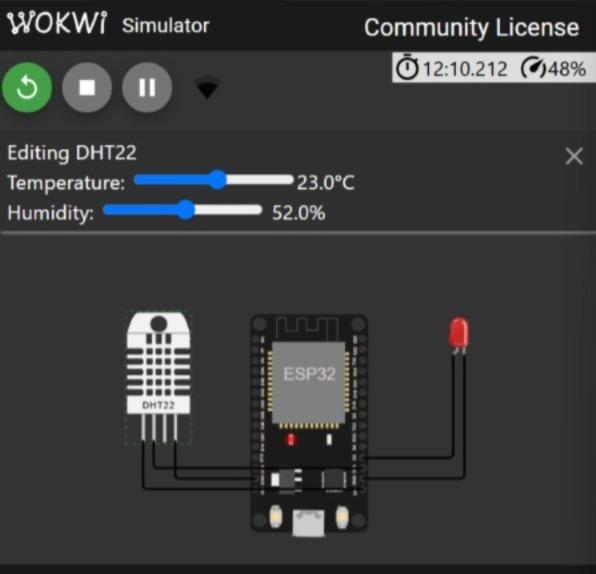
Hasil eksperimen menunjukkan bahwa sistem IoT berbasis ESP32, DHT22, dan Node-RED dapat dengan akurat membaca dan mengirim data suhu dan kelembapan. Data dikirim ke Node-RED melalui protokol MQTT, dan dashboard menunjukkan hasilnya. Meskipun diuji dalam berbagai kondisi jaringan Wi-Fi, sistem menunjukkan respons cepat dan keandalan.

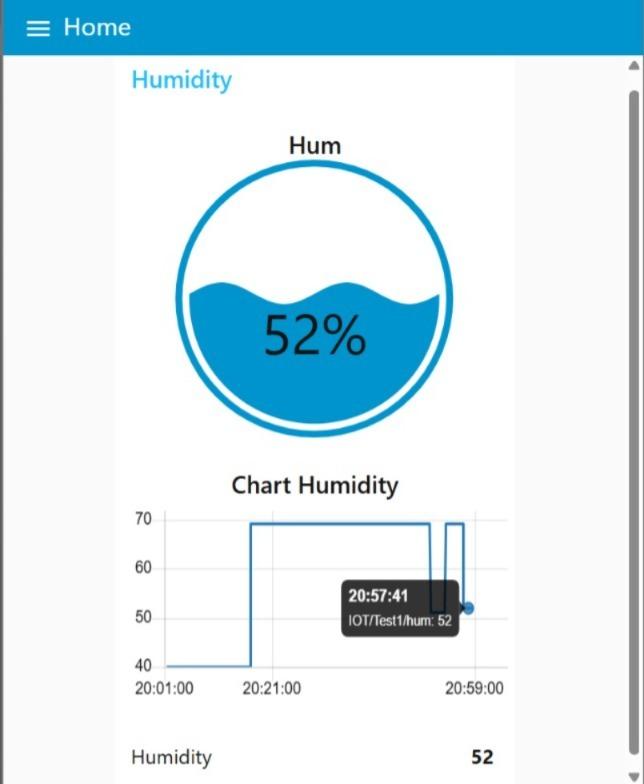
Dengan menggunakan platform Wokwi, simulasi menjadi lebih mudah untuk menemukan kesalahan dan memungkinkan debugging kode tanpa mengorbankan perangkat keras. Dashboard Node-RED menyediakan antarmuka yang mudah dipahami yang membuat analisis data lebih mudah. Dengan sistem ini, aplikasi pemantauan lingkungan yang cocok untuk rumah pintar, pertanian cerdas, atau proyek berbasis Internet of Things lainnya dapat dibuat.

**3.2 Dokumentasi eksperimen meliputi screenshoot simulasi :**

****

****

****

****

**Lampiran**

**Kode Program 1 :**

"v134.4",

"h-9.6",

"v0",

"h0",

"v-115.2",

"h-28.8",

"v134.4",

"h0",

"v9.6"

]

]

],

"dependencies": {}

}

**Kode Program 2 :**

#include <Arduino.h>

#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"; //server

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()); //Topic Temperature

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

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

Serial.print("Temperature: ");

Serial.println(temp);

Serial.print("Humidity: ");

Serial.println(hum);

}

}