**PRAKTIKUM SIMULASI SISTEM MONITORING SUHU DAN**

**KELEMBAPAN DENGAN MQTT DAN ESP32**

*Fredlina Devhania Kholishah1*

*Fakultas Vokasi, Universitas Brawijaya*

[*devhania88@gmail.com*](mailto:devhania88@gmail.com)

### **ABSTRAK**

Praktikum ini bertujuan untuk merancang sistem monitoring suhu dan kelembaban berbasis Internet of Things (IoT) menggunakan mikrokontroler ESP32. Sistem ini memanfaatkan sensor DHT22 untuk memperoleh data suhu dan kelembaban, lalu menampilkannya pada layar OLED. Data juga dikirimkan ke broker MQTT secara berkala setelah ESP32 terhubung ke jaringan WiFi. LED digunakan sebagai indikator aktivitas sistem, sedangkan push button dapat difungsikan sebagai kontrol manual. Hasil simulasi menunjukkan bahwa sistem mampu menampilkan dan mengirimkan data dengan stabil dan akurat melalui jaringan MQTT.

***Kata Kunci****: Internet of Things, ESP32, MQTT, DHT22, OLED*

### **ABSTRACT**

This practicum aims to design a temperature and humidity monitoring system based on the Internet of Things (IoT) using an ESP32 microcontroller. The system utilizes a DHT22 sensor to obtain temperature and humidity data, which is then displayed on an OLED screen. The data is also periodically sent to an MQTT broker after the ESP32 connects to a WiFi network. An LED is used as a system activity indicator, while a push button can be used for manual control. Simulation results show that the system can reliably display and transmit data through the MQTT network.

***Keywords****: Internet of Things, ESP32, MQTT, DHT22, OLED*

## **1. PENDAHULUAN**

### **1.1 Latar Belakang**

Perkembangan teknologi Internet of Things (IoT) telah memungkinkan integrasi antara perangkat keras dan jaringan internet untuk memudahkan pengawasan serta kontrol jarak jauh secara real-time. Salah satu penerapan IoT yang banyak dikembangkan adalah sistem monitoring suhu dan kelembaban udara yang sangat penting dalam berbagai bidang, seperti pertanian, industri, kesehatan, hingga sistem rumah pintar (smart home).

Dalam praktikum ini, dirancang dan disimulasikan sebuah sistem monitoring berbasis ESP32 yang terhubung dengan sensor DHT22 untuk membaca suhu dan kelembaban. Data dari sensor ditampilkan melalui layar OLED dan dikirimkan ke broker MQTT menggunakan jaringan WiFi. Selain itu, sistem juga dilengkapi dengan LED indikator dan push button untuk interaksi tambahan. Pemanfaatan MQTT sebagai protokol komunikasi dipilih karena ringan dan efisien untuk perangkat IoT. Dengan proyek ini, mahasiswa diharapkan dapat memahami konsep dasar komunikasi MQTT, pemrograman mikrokontroler ESP32, serta integrasi sensor dan aktuator dalam suatu sistem IoT.

### **1.2 Tujuan**

Adapun tujuan dari pelaksanaan praktikum ini adalah sebagai berikut:

1. Merancang sistem monitoring suhu dan kelembaban menggunakan sensor DHT22 yang terintegrasi dengan mikrokontroler ESP32.
2. Menampilkan data suhu dan kelembaban pada layar OLED secara real-time.
3. Mengirimkan data sensor ke broker MQTT melalui koneksi WiFi menggunakan protokol komunikasi IoT.
4. Menguji respon dan kestabilan sistem, termasuk fungsi indikator LED dan tombol tekan (push button) dalam mendukung proses monitoring.
5. Memahami dan menerapkan konsep dasar komunikasi MQTT dalam sistem Internet of Things.

### **2. METODOLOGI**

#### **2.1 Alat dan Bahan**

Dalam praktikum ini digunakan beberapa alat dan bahan utama, yaitu:

* ESP32 Devkit V1, berfungsi sebagai mikrokontroler utama yang menangani komunikasi data dan kendali sistem.
* Sensor DHT22, digunakan untuk membaca data suhu dan kelembaban dari lingkungan sekitar.
* OLED Display SSD1306, digunakan untuk menampilkan hasil pembacaan sensor secara real-time.
* LED berwarna merah, kuning, dan biru, digunakan sebagai indikator sistem, misalnya saat data berhasil dikirim ke MQTT atau menunjukkan status tertentu.
* Push Button, sebagai tombol kontrol manual (misalnya untuk reset atau trigger pengiriman data).
* Buzzer (opsional), untuk memberikan notifikasi berbasis suara.
* Simulator Wokwi, digunakan sebagai platform simulasi virtual untuk merakit rangkaian dan menguji program.
* WiFi virtual dari Wokwi sebagai jaringan koneksi ke internet, serta broker MQTT (seperti test.mosquitto.org) sebagai penerima data dari ESP32.

#### **2.2 Diagram Rangkaian**

Rangkaian sistem dirancang di simulator Wokwi. ESP32 sebagai pusat sistem terhubung ke sensor DHT22 pada pin D15, OLED display menggunakan komunikasi I2C melalui pin D21 (SDA) dan D22 (SCL). LED indikator dihubungkan ke pin digital D2, D4, dan D5, sedangkan push button ke pin D13. Selain itu, ESP32 terhubung ke jaringan WiFi virtual dan mengirim data ke broker MQTT. Diagram visual terlampir dalam bentuk file .json pada sistem Wokwi.

#### **2.3 Langkah-Langkah Pelaksanaan**

Langkah-langkah yang dilakukan dalam praktikum ini meliputi:

1. Merancang rangkaian simulasi pada Wokwi sesuai kebutuhan sistem monitoring suhu dan kelembaban.
2. Mengimpor library yang dibutuhkan seperti WiFi.h, DHT.h, Adafruit\_SSD1306.h, dan PubSubClient.h ke dalam program.
3. Menulis program untuk membaca data dari sensor DHT22, menampilkannya ke OLED, serta mengirimkan data tersebut ke broker MQTT.
4. Menambahkan logika untuk menyalakan LED sebagai indikator ketika data berhasil dikirim.
5. Menggunakan tombol sebagai kontrol manual untuk fungsi tertentu, seperti mengatur ulang data atau memicu pengiriman ulang.
6. Menjalankan simulasi dan mengamati kinerja sistem pada Wokwi Terminal dan tampilan OLED.

#### **2.4 Alur Sistem**

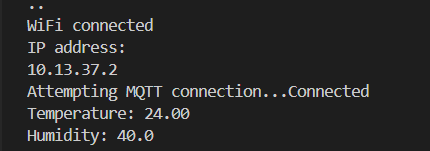
Alur kerja sistem secara umum adalah sebagai berikut:

* Sistem dimulai dan ESP32 akan mencoba terhubung ke jaringan WiFi.
* Jika koneksi berhasil, ESP32 melanjutkan untuk menghubungkan diri ke broker MQTT.
* Setelah terhubung, ESP32 mulai membaca data dari sensor DHT22.
* Data yang dibaca ditampilkan ke layar OLED.
* Data juga dikirim ke broker MQTT melalui koneksi internet.
* Saat pengiriman berhasil, LED indikator menyala sebagai tanda bahwa sistem berjalan normal.
* Push button dapat digunakan untuk memicu ulang proses tertentu (misalnya pengiriman manual).
* Proses ini berlangsung secara berulang (loop).

**3. HASIL DAN PEMBAHASAN**

Setelah merancang dan menyimulasikan sistem monitoring suhu dan kelembaban berbasis ESP32 pada platform Wokwi, sistem berhasil dijalankan dengan baik sesuai rancangan. Hasil yang diperoleh selama simulasi menunjukkan bahwa semua komponen berfungsi sebagaimana mestinya.

**3.1 Pembacaan Data Sensor**Sensor DHT22 mampu membaca suhu dan kelembaban dengan akurasi yang stabil. Data yang terbaca secara berkala ditampilkan pada OLED display dengan format yang mudah dibaca, seperti:



**3.2 Tampilan Data di OLED**Layar OLED menampilkan data sensor secara real-time. Dengan menggunakan library Adafruit\_SSD1306, tampilan yang muncul cukup informatif dan terbaca jelas pada simulasi. Hal ini memudahkan pengguna untuk memantau kondisi lingkungan sekitar secara langsung.

**3.3 Koneksi MQTT dan Pengiriman Data**ESP32 berhasil terhubung ke WiFi virtual yang disediakan oleh Wokwi, kemudian menghubungkan diri ke broker MQTT (seperti test.mosquitto.org). Data sensor dikirim dalam format payload MQTT dengan struktur JSON atau teks sederhana. Pengiriman berhasil ditandai dengan menyalanya LED indikator pada papan ESP32. Fungsi client.publish() berjalan baik tanpa error.

**3.4 Indikator LED**LED digunakan sebagai indikator status. Misalnya, LED merah menyala saat pengiriman data berhasil, LED kuning menyala ketika ada gangguan koneksi, dan LED biru untuk status sistem aktif. Penggunaan LED ini membantu dalam proses debugging dan memberikan umpan balik visual saat simulasi berlangsung.

**3.5 Tombol Tekan (Push Button)**Push button yang dihubungkan ke salah satu pin digital ESP32 digunakan sebagai input manual. Ketika tombol ditekan, sistem dapat dikonfigurasi untuk mengirim data secara langsung atau mengatur ulang tampilan OLED. Fungsi ini menambah interaktivitas sistem.

**3.6 Kelebihan Sistem**

* Sistem bekerja secara otomatis dan real-time.
* Tampilan OLED memudahkan monitoring langsung.
* Data dikirim ke MQTT yang dapat diakses jarak jauh.
* Rangkaian sederhana dan efisien untuk proyek awal IoT.

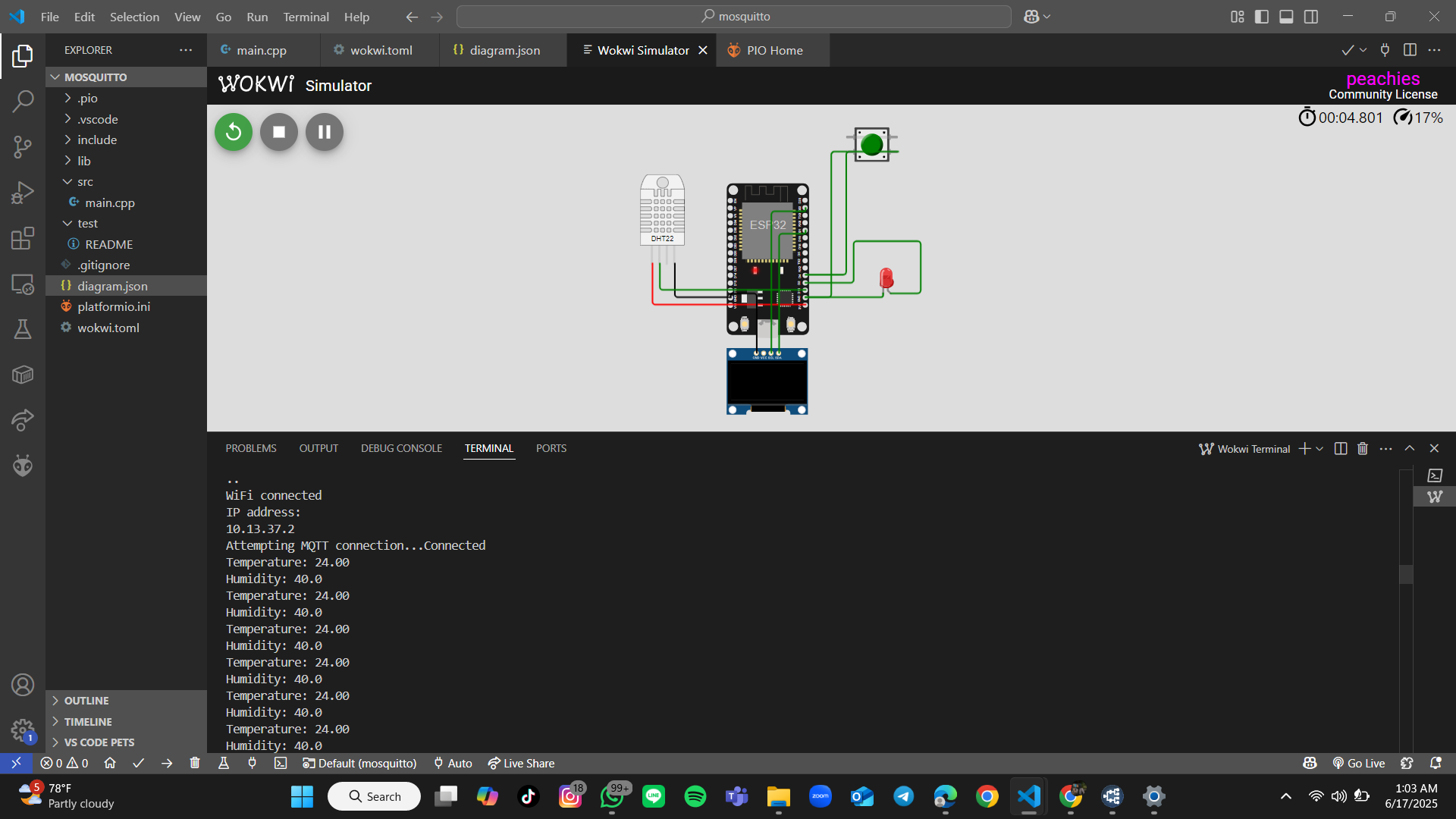
**3.7 Kekurangan dan Kendala**

* Karena berbasis simulasi, tidak ada gangguan jaringan nyata yang bisa diuji.
* Tidak semua fitur MQTT (seperti penerimaan pesan / subscribe) digunakan dalam proyek ini.
* Tidak ada penyimpanan data historis karena sistem belum terintegrasi dengan database atau aplikasi lain.

Secara keseluruhan, sistem berhasil memenuhi tujuan praktikum, yaitu membaca data lingkungan dan mengirimkannya secara nirkabel melalui protokol MQTT. Proyek ini dapat dikembangkan lebih lanjut untuk pemantauan jarak jauh secara online, termasuk integrasi dengan database atau dashboard visual.

**4. LAMPIRAN**

### **4.1 Gambar Rangkaian Wokwi**

****

### **4.2 Kode Program**

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 = "test.mosquitto.org";//"broker.emqx.io";**

**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);**

**}**

**}**

Kode Program diagram.json

**{**

**"version": 1,**

**"author": "Fredlina",**

**"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" } },**

**{**

**"type": "board-ssd1306",**

**"id": "oled1",**

**"top": 214.34,**

**"left": 0.23,**

**"attrs": { "i2cAddress": "0x3c" }**

**},**

**{**

**"type": "wokwi-pushbutton",**

**"id": "btn1",**

**"top": -70.6,**

**"left": 153.6,**

**"attrs": { "color": "green", "xray": "1" }**

**}**

**],**

**"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" ] ],**

**[ "oled1:SDA", "esp:D21", "green", [ "v-153.6", "h28.87" ] ],**

**[ "oled1:SCL", "esp:D22", "green", [ "v-182.4", "h38.7" ] ],**

**[ "oled1:GND", "esp:GND.2", "black", [ "v-76.8", "h-28.8" ] ],**

**[ "btn1:2.l", "esp:GND.1", "green", [ "h-19.2", "v187.6" ] ],**

**[ "btn1:2.r", "esp:D4", "green", [ "h-67", "v158.6" ] ]**

**],**

**"dependencies": {}**

**}**