**LAPORAN PRAKTIKUM**

**IOT MINGGU KE-6**

**MQTT-Based Environment Monitoring With Esp32 dan DHT22**

****

**Dosen Pengampu :**

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

**Disusun Oleh:**

**Muhammad Alif Aris**

**(233140707111077)**

**PROGRAM STUDI TEKNOLOGI INFORMASI**

**FAKULTAS VOKASI**

**UNIVERSITAS BRAWIJAYA**

**2025**

**Abstrak**

Metode komunikasi yang andal dan hemat daya disediakan oleh teknologi Internet of Things (IoT) melalui protokol Message Queuing Telemetry Transport (MQTT). Dalam eksperimen ini, ESP32 berfungsi sebagai mikrokontroler utama, dan DHT22 berfungsi sebagai sensor suhu dan kelembapan. Hasil eksperimen menunjukkan bahwa sistem dapat berfungsi dengan baik dalam skenario simulasi Wokwi bahkan ketika ada gangguan koneksi. Aplikasi monitoring seperti HiveMQ menerima data yang diperoleh secara real-time dari broker MQTT melalui koneksi Wi-Fi. Solusi pemantauan lingkungan yang efisien, murah, dan mudah diterapkan dalam skala yang berbeda dibuat dengan kombinasi ESP32, DHT22, dan MQTT. Pengelolaan energi, rumah pintar, dan sistem pertanian presisi adalah beberapa contoh aplikasi yang dapat digunakan.

*Keywords - ESP32, DHT22, MQTT, IoT, pemantauan suhu, broker MQTT, aplikasi IoT.*

**BAB I**

**PENDAHULUAN**

**1.1 Latar Belakang**

Pemantauan lingkungan sangat penting untuk banyak hal, seperti pengelolaan energi, pertanian, dan layanan kesehatan. Protokol MQTT memberikan metode yang ringan dan andal untuk mengirimkan data sensor secara real-time dalam konteks Internet of Things (IoT). Mikrokontroler ESP32 mendukung konektivitas Wi-Fi dan sensor DHT22 melacak suhu dan kelembapan. Data yang diambil dari sensor dikirim ke broker MQTT, yang kemudian dikirim ke dashboard monitoring untuk dianalisis. Sistem ini dimaksudkan untuk menghemat uang, mudah digunakan, dan terjangkau. Dengan fitur ini, monitoring berbasis MQTT dapat dilaksanakan dalam skala kecil hingga besar, menawarkan solusi kreatif untuk teknologi otomatisasi dan pengelolaan lingkungan.

**1.2 Tujuan Eksperimen**

1. Membuat model sistem IoT yang dapat mengukur dan mengirimkan data suhu dan kelembapan dengan menggunakan protokol MQTT dan ESP32.
2. Mengevaluasi stabilitas koneksi dan keandalan antara perangkat penerima, broker MQTT, dan ESP32 dalam sistem monitoring lingkungan.
3. Demonstrasi penggunaan teknologi Internet of Things untuk aplikasi pemantauan lingkungan yang murah dan ramah pengguna.

**BAB II**

**METODOLOGI**

**2.1 Alat dan Bahan**

* ESP32 DevKit V1
* Sensor Suhu dan Kelembapan (DHT22)
* Resistor (1kΩ dan 10kΩ)
* Kabel Penghubung Virtual
* Breadboard Virtual
* Platform Simulasi Wokwi

**2.3 Implementasi Sistem**

1. Menghubungkan ESP32 dengan sensor DHT22 pada Wokwi: Anda dapat menggunakan platform Wokwi untuk menyusun rangkaian virtual dengan menggunakan breadboard virtual untuk menghubungkan pin VCC, GND, dan DATA dari sensor DHT22 ke ESP32.
2. Mengaktifkan Koneksi Wi-Fi pada ESP32: Tambahkan kode ke pustaka WiFi.h untuk menghubungkan ESP32 ke jaringan Wi-Fi, termasuk informasi SSID dan password jaringan.
3. Mengatur Pengiriman Data ke Broker MQTT menggunakan program ESP32 untuk membaca data dari DHT22 dan mempublikasikannya ke topik MQTT tertentu.
4. Jalankan Simulasi di Wokwi: Gunakan jaringan Wi-Fi di Wokwi untuk memantau pembacaan data DHT22 dan pengiriman data ke broker MQTT.
5. Memeriksa Data dengan Client MQTT Gunakan alat seperti HiveMQ untuk melihat data suhu dan kelembapan yang diterima dari ESP32, memastikan bahwa data dikirim secara konsisten.

**BAB III**

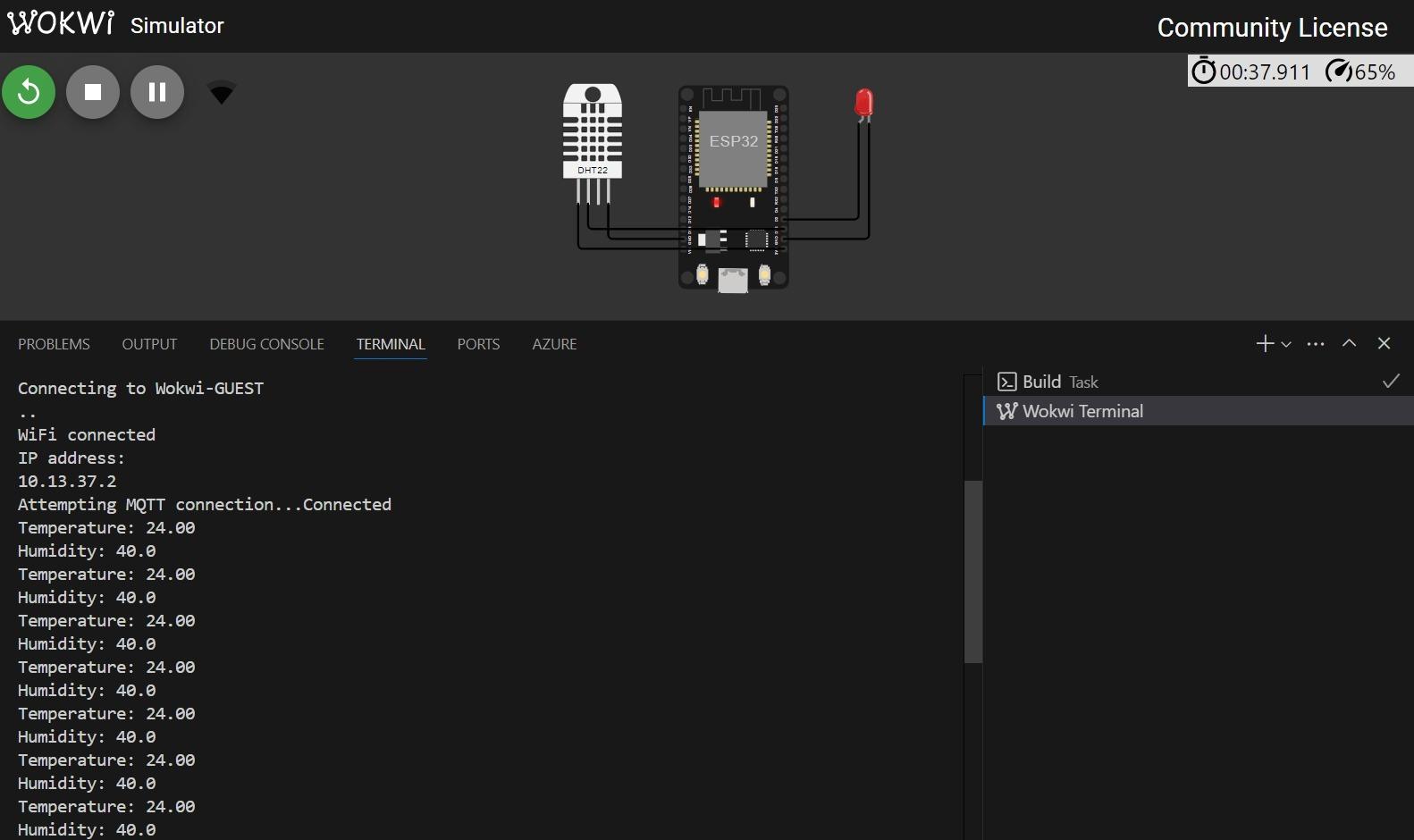
**HASIL DAN PEMBAHASAN**

**3.1 Hasil Eksperimen**

Hasil eksperimen menunjukkan bahwa sistem pemantauan berbasis MQTT yang menggunakan ESP32 dan DHT22 dapat dengan akurat memberikan data suhu dan kelembapan. Dengan konfigurasi yang tepat, ESP32 dapat terhubung ke jaringan Wi-Fi dan mempublikasikan data ke broker MQTT. Aplikasi MQTT client, seperti HiveMQ, dapat mengakses data yang dikirim dengan hasil yang konsisten.

Meskipun dilakukan uji pengambilan data selama waktu yang cukup lama, sistem tetap beroperasi dengan baik. Untuk memastikan pengiriman data yang berkelanjutan, koneksi Wi-Fi yang terputus dapat dipulihkan secara otomatis. Eksperimen ini menunjukkan potensi teknologi Internet of Things (IoT) untuk pemantauan lingkungan; aplikasi otomatisasi dan pengelolaan sumber daya yang lebih efisien dapat terus dikembangkan. Sebelum penerapan sistem secara fisik, platform simulasi virtual sangat membantu.

**3.2 Dokumentasi eksperimen meliputi screenshoot simulasi :**

****

**Lampiran**

**Kode Program 1 :**

{

"version": 1,

"author": "Subairi",

"editor": "wokwi",

"parts": [

{ "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": 0, "left": 0, "attrs": {} },

{ "type": "wokwi-dht22", "id": "dht1", "top": 0.3, "left": -111, "attrs": {} },

{

"type": "wokwi-led",

"id": "led1",

"top": -3.6,

"left": 157.8,

"attrs": { "color": "red", "flip": "1" }

}

],

"connections": [

[ "esp:TX0", "$serialMonitor:RX", "", [] ],

[ "esp:RX0", "$serialMonitor:TX", "", [] ],

[ "dht1:GND", "esp:GND.2", "black", [ "v0" ] ],

[ "dht1:VCC", "esp:3V3", "black", [ "v0" ] ],

[ "dht1:SDA", "esp:D15", "black", [ "v0" ] ],

[ "led1:A", "esp:D2", "black", [ "v0" ] ],

[ "led1:C", "esp:GND.1", "black", [ "v0" ] ]

],

"dependencies": {}

}

**Kode Program 2 :**

#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 = "test.mosquitto.org";

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);

}

}