**Monitoring Suhu dan Kelembapan Berbasis ESP32 dengan Protokol MQTT**

*Ibnu Jaisyurrahman Faiz*

*Program Studi Teknologi Informasi, Fakultas Vokasi, Universitas Brawijaya*

*Email: (*[*ibnufaiz72ub@student.ub.ac.id*](mailto:ibnufaiz72ub@student.ub.ac.id)*)*

**Abstrak:** Proyek ini bertujuan untuk merancang dan mengimplementasikan sistem Internet of Things (IoT) menggunakan mikrokontroler ESP32, sensor DHT22, dan protokol komunikasi MQTT. Sistem ini mampu melakukan monitoring suhu dan kelembapan secara real-time serta mengendalikan perangkat (LED) dari jarak jauh melalui MQTT broker publik test.mosquitto.org. Data suhu dan kelembapan dikirim secara berkala ke topik MQTT, sementara perintah untuk menghidupkan atau mematikan LED dikirim melalui topik terpisah. Dengan pendekatan ini, sistem dapat berfungsi sebagai dasar pengembangan aplikasi IoT yang hemat biaya dan fleksibel, seperti smart home atau pertanian cerdas.

Kata Kunci : *Internet of Things, ESP32, MQTT, DHT22, Sensor, Kendali Jarak Jauh*

**Abstract:** This project aims to design and implement an Internet of Things (IoT) system using the ESP32 microcontroller, DHT22 sensor, and MQTT communication protocol. The system is capable of real-time temperature and humidity monitoring, as well as remote device control (LED) through a public MQTT broker (test.mosquitto.org). Temperature and humidity data are periodically published to MQTT topics, while LED control commands are received via a separate topic. This approach demonstrates a low-cost and flexible foundation for developing practical IoT applications such as smart homes or precision agriculture.

Key Word: *Internet of Things, ESP32, MQTT, DHT22, Sensor, Remote Control*

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

Perkembangan teknologi Internet of Things (IoT) telah memberikan kemudahan dalam membangun sistem monitoring dan kendali jarak jauh yang efisien dan real-time. Salah satu protokol komunikasi yang banyak digunakan dalam IoT adalah Message Queuing Telemetry Transport (MQTT), karena bersifat ringan, cepat, dan cocok untuk perangkat dengan sumber daya terbatas seperti mikrokontroler. Dalam eksperimen ini, digunakan mikrokontroler ESP32 yang memiliki konektivitas WiFi bawaan, serta sensor DHT22 untuk mengukur suhu dan kelembapan lingkungan. Data hasil pengukuran dikirim secara berkala ke broker MQTT publik yaitu test.mosquitto.org, sehingga dapat diakses oleh perangkat lain di jaringan. Selain itu, ESP32 juga dapat menerima perintah dari broker untuk mengaktifkan atau mematikan LED. Eksperimen ini menunjukkan bagaimana integrasi antara perangkat keras dan komunikasi berbasis internet dapat diterapkan dalam pengembangan sistem IoT yang sederhana dan efektif.

1. **Tujuan eksperimen**

* Menerapkan protokol MQTT untuk komunikasi data antara ESP32 dan broker.
* Mengirim data suhu dan kelembapan dari sensor DHT22 ke broker MQTT secara berkala.
* Mengontrol LED secara jarak jauh melalui perintah yang dikirim ke topik MQTT.
* Menguji kemampuan ESP32 sebagai perangkat IoT untuk monitoring dan kendali berbasis internet.

1. **Metodologi**
2. **Alat dan Bahan**

Adapun alat dan bahan yang digunakan dalam eksperimen ini adalah sebagai berikut:

* Sensor DHT22 - Berfungsi untuk mengukur suhu dan kelembapan udara secara digital.
* LED Merah - Digunakan sebagai indikator output yang dikendalikan melalui perintah dari broker MQTT.
* Visual Studio Code (VS Code) - Digunakan sebagai lingkungan pengembangan utama untuk menulis, mengedit, dan menjalankan kode program ESP32.
* PlatformIO Extension - Ekstensi pada VS Code yang digunakan untuk memprogram dan membangun proyek berbasis mikrokontroler seperti ESP32.
* MQTT Client (misalnya: MQTT Explorer / MQTT Dashboard) - Alat bantu eksternal untuk memantau data yang dikirim dari ESP32 dan mengirim perintah balik.
* Library Arduino (WiFi.h, PubSubClient.h, DHTesp.h) - Digunakan untuk mendukung komunikasi WiFi, komunikasi dengan MQTT broker, dan pembacaan data dari sensor DHT secara virtual.
* Koneksi Internet - Diperlukan untuk menghubungkan simulasi ESP32 dengan broker MQTT publik melalui jaringan internet.

1. **Langkah Implementasi**

Langkah implementasi untuk Monitoring Suhu dan Kelembapan Berbasis ESP32 dengan Protokol MQTT sebagai berikut:

* Buka Wokwi Simulator dan buat proyek baru
* Tambahkan library berikut ke file platformio: WiFi.h untuk koneksi ke jaringan WiFi. PubSubClient.h untuk komunikasi dengan MQTT broker. DHTesp.h untuk simulasi pembacaan sensor suhu dan kelembapan
* Buat program utama untuk:

Menghubungkan ESP32 ke jaringan WiFi

Menghubungkan ke broker MQTT publik (test.mosquitto.org)

Membaca data suhu dan kelembapan (secara simulatif) dari DHT22

Mengirim data ke topik MQTT secara berkala

* Jalankan kode melalui PlatformIO dan pantau hasilnya pada Serial Monitor.

Pastikan koneksi WiFi dan koneksi ke broker MQTT berhasil dilakukan.

* Pastikan data suhu dan kelembapan dikirim dengan interval yang konsisten (misalnya setiap 2 detik).

1. **Hasil dan Pembahasan**
2. **Hasil Eksperimen**

Eksperimen dilakukan dengan menjalankan program ESP32 secara simulatif melalui Visual Studio Code menggunakan ekstensi PlatformIO. Program berhasil dikompilasi dan dijalankan, serta menunjukkan fungsionalitas yang sesuai dengan tujuan eksperimen.

Pada saat program dijalankan, Serial Monitor menampilkan status koneksi ke WiFi dan broker MQTT. Setelah koneksi berhasil, ESP32 mulai membaca data suhu dan kelembapan dari sensor DHT22 (secara simulatif), kemudian mengirimkannya ke broker MQTT publik test.mosquitto.org melalui topik IOT/Test1/temp dan IOT/Test1/hum.

Connecting to Wokwi-GUEST

..

WiFi connected

IP address:

10.13.37.2

Attempting MQTT connection...Connected

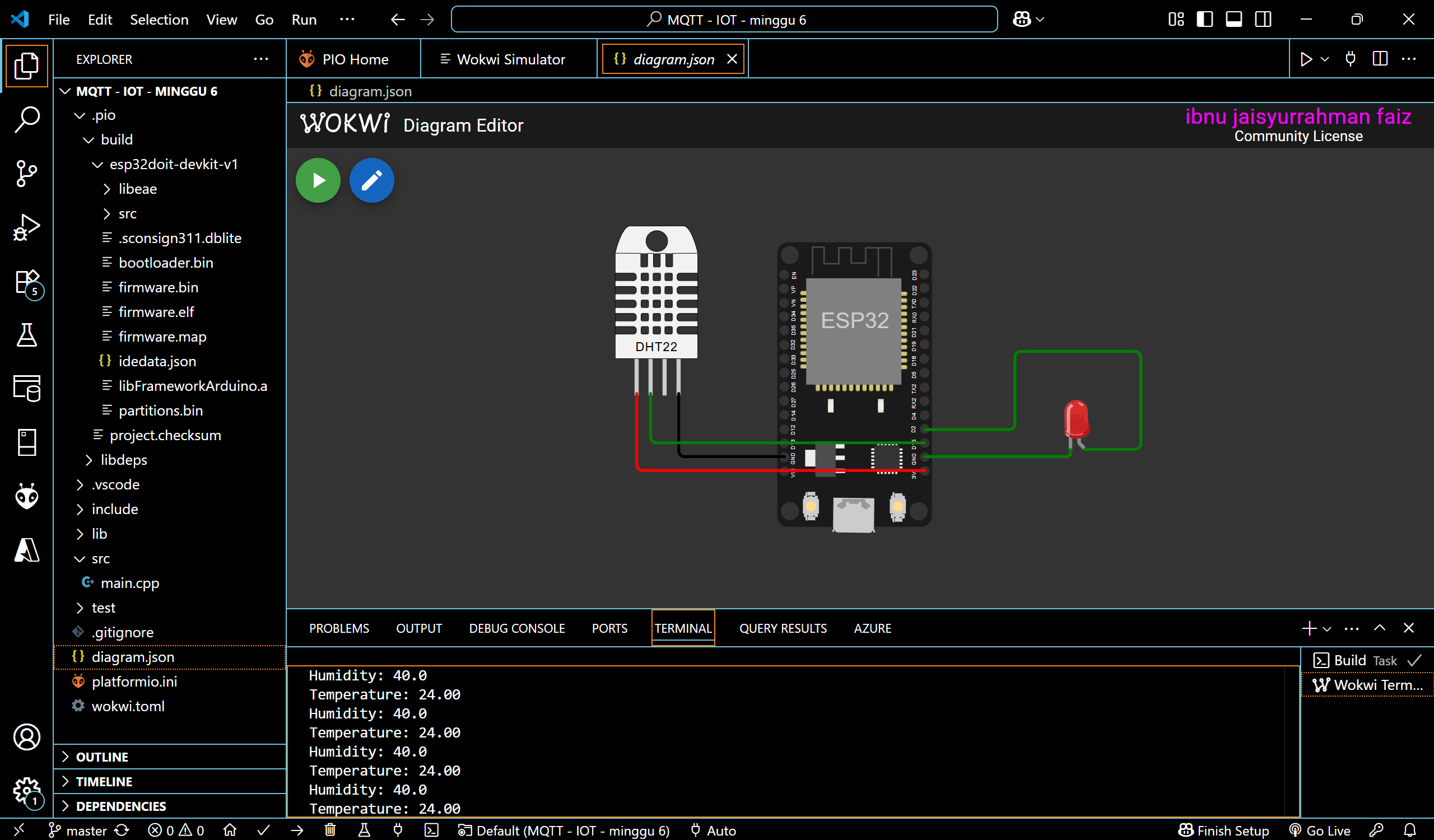
Temperature: 24.00

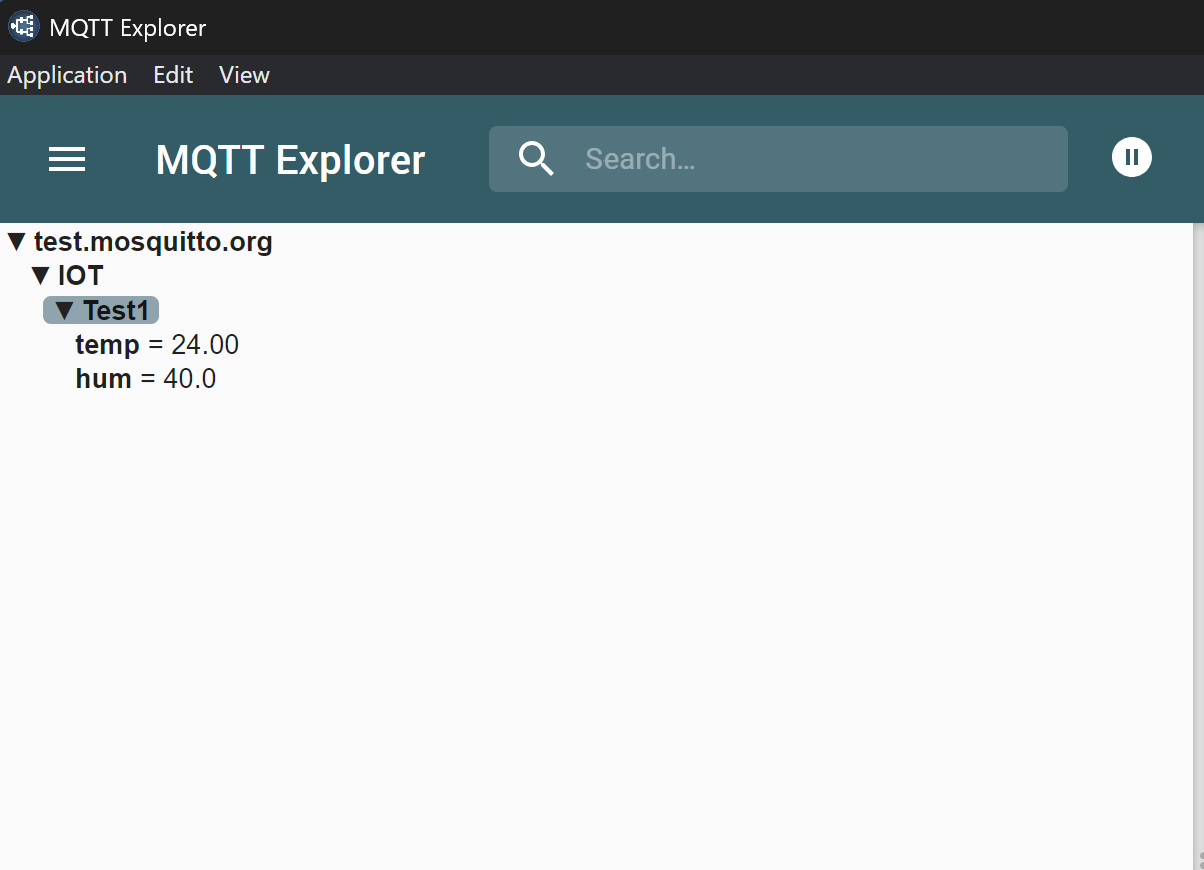
Humidity: 40.0

Temperature: 24.00

Humidity: 40.0

Temperature: 24.00





1. **Lampiran**

* 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";
* 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": " Anonymous maker",
* "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"
* }
* }
* ],
* "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"
* ]
* ]
* ],
* "dependencies": {}
* }