**LAPORAN PRAKTIKUM SIMULASI INTERNET OF THINGS (IOT) PEMANTAUAN SUHU DAN KELEMBAPAN BERBASIS ESP32 MENGGUNAKAN NODE-RED DAN INFLUXDB**

*Fredlina Devhania Kholishah1*

*Fakultas Vokasi, Universitas Brawijaya*

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

**ABSTRAK**

Pada praktik ini, ESP32 dimanfaatkan untuk melakukan pemantauan suhu dan kelembapan menggunakan simulasi sensor DHT22 pada platform Wokwi. Data hasil pengukuran dikirimkan ke Node-RED melalui protokol MQTT untuk kemudian diproses dan disimpan dalam InfluxDB sebagai basis data deret waktu. Node-RED juga digunakan untuk menampilkan data secara real-time melalui dashboard interaktif. Melalui simulasi ini, sistem Internet of Things (IoT) dapat diuji tanpa memerlukan perangkat keras fisik. Hasil praktik menunjukkan bahwa kolaborasi antara ESP32, Node-RED, dan InfluxDB mampu menjalankan pemantauan lingkungan secara efektif. Praktikum ini menggambarkan konsep dasar dari IoT yang dapat diterapkan dalam berbagai bidang, seperti rumah pintar, pertanian cerdas, serta pemantauan kualitas udara, dengan sistem efisien yang berjalan secara waktu nyata.

***Kata Kunci****: ESP32, Wokwi, MQTT, Node-RED, InfluxDB*

**ABSTRACT**

This project utilizes the ESP32 to monitor temperature and humidity using a DHT22 sensor simulation on the Wokwi platform. Sensor data is transmitted via MQTT to Node-RED, where it is processed and stored in InfluxDB as a time-series database. Node-RED also enables real-time visualization of the data through an interactive dashboard. The simulation makes it possible to test IoT systems without physical hardware. The results confirm that integrating ESP32, Node-RED, and InfluxDB performs well for environmental monitoring. This practicum illustrates the fundamental concept of the Internet of Things (IoT), which can be applied to areas such as smart homes, precision agriculture, and air quality monitoring through an efficient and real-time system.

***Keywords****: ESP32, Wokwi, MQTT, Node-RED, InfluxDB*

### **Pendahuluan**

**1.1 Latar Belakang**

Perkembangan teknologi Internet of Things (IoT) telah membawa perubahan signifikan dalam berbagai bidang, terutama dalam sistem pemantauan dan otomasi. IoT memungkinkan perangkat untuk saling terhubung dan berkomunikasi melalui internet, sehingga data dari berbagai sensor dapat dikumpulkan, dianalisis, dan ditindaklanjuti secara real-time. Salah satu perangkat yang banyak digunakan dalam pengembangan IoT adalah ESP32, sebuah mikrokontroler yang memiliki konektivitas Wi-Fi dan Bluetooth.

Dalam praktik ini, dilakukan simulasi sistem IoT menggunakan ESP32 untuk memantau suhu dan kelembapan lingkungan. Sensor DHT22 digunakan sebagai sumber data yang kemudian dikirimkan melalui protokol MQTT ke platform Node-RED. Node-RED bertugas sebagai pengolah data dan penyedia visualisasi secara real-time melalui dashboard. Data yang diterima juga disimpan di dalam InfluxDB, sebuah basis data yang dirancang khusus untuk menyimpan data dalam bentuk deret waktu (time-series).

Simulasi ini dijalankan pada platform Wokwi yang memungkinkan pengujian dan pengembangan sistem IoT tanpa memerlukan perangkat fisik. Dengan pendekatan ini, pengguna dapat memahami alur komunikasi data antarperangkat IoT secara efisien dan hemat biaya. Praktikum ini bertujuan untuk memberikan pemahaman dasar tentang integrasi perangkat keras dan perangkat lunak dalam membangun sistem IoT yang dapat diterapkan pada berbagai bidang seperti rumah pintar, pertanian, dan pemantauan kualitas udara.

### **1.2 Tujuan**

### Tujuan dari praktikum ini adalah:

1. Membangun sistem monitoring suhu dan kelembapan berbasis ESP32 dan sensor DHT22 secara simulatif menggunakan platform Wokwi.
2. Mengirimkan data sensor melalui protokol MQTT ke Node-RED untuk diproses dan divisualisasikan.
3. Menyimpan data secara otomatis ke dalam basis data time-series InfluxDB.

### **2. Metodologi**

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

**Alat:**

* Laptop
* Platform Wokwi
* Visual Studio Code (VSCode)
* Platform Node-RED
* Platform InfluxDB

**Bahan:**

* ESP32 (simulasi melalui Wokwi)
* Sensor DHT22
* LED merah
* Bahasa pemrograman C++ dengan pustaka Arduino
* Library:
  + DHTesp.h
  + WiFi.h
  + PubSubClient.h (meskipun tidak digunakan secara aktif dalam koneksi MQTT, masih terpasang di kode)

#### **2.2 Langkah Implementasi**

1. Membuka situs [Wokwi](https://wokwi.com) dan membuat proyek baru dengan board ESP32.
2. Menambahkan sensor DHT22 dan menghubungkannya ke pin GPIO 15.
3. Menambahkan LED merah dan menghubungkannya ke GPIO 2.
4. Mengunduh dan menginstal ekstensi PlatformIO serta Wokwi di VSCode.
5. Membuat file diagram.json dan menyalin konfigurasi rangkaian dari Wokwi.
6. Menulis kode utama pada file main.cpp untuk membaca data dari sensor dan menyalakan LED berdasarkan suhu tertentu.
7. Menginstal Node-RED melalui terminal dengan perintah:  
   bash “npm install -g --unsafe-perm node-red”
8. Menjalankan Node-RED dengan perintah:  
   bash “node-red”
9. Menginstal node tambahan di Node-RED seperti influxdb dan dashboard.
10. Membuat flow di Node-RED dengan node function, influxdb out, dan ui\_gauge atau ui\_chart.
11. Membuat konfigurasi InfluxDB dengan detail berikut:

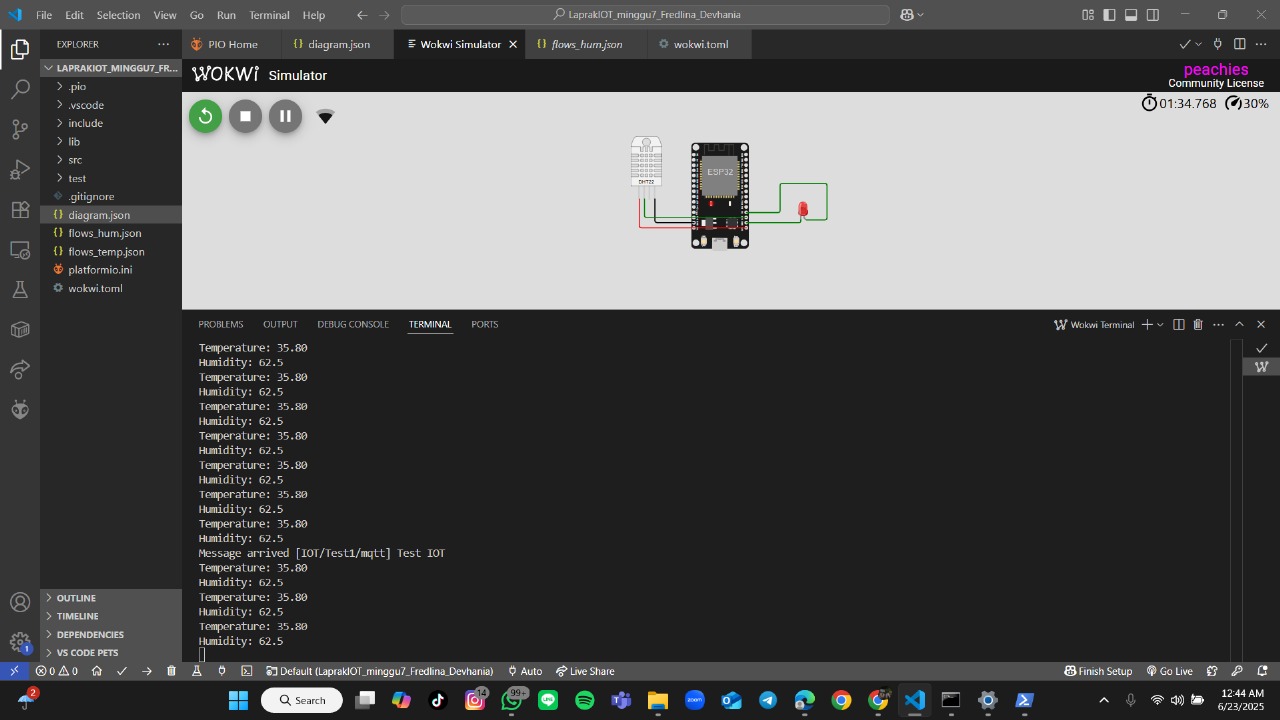
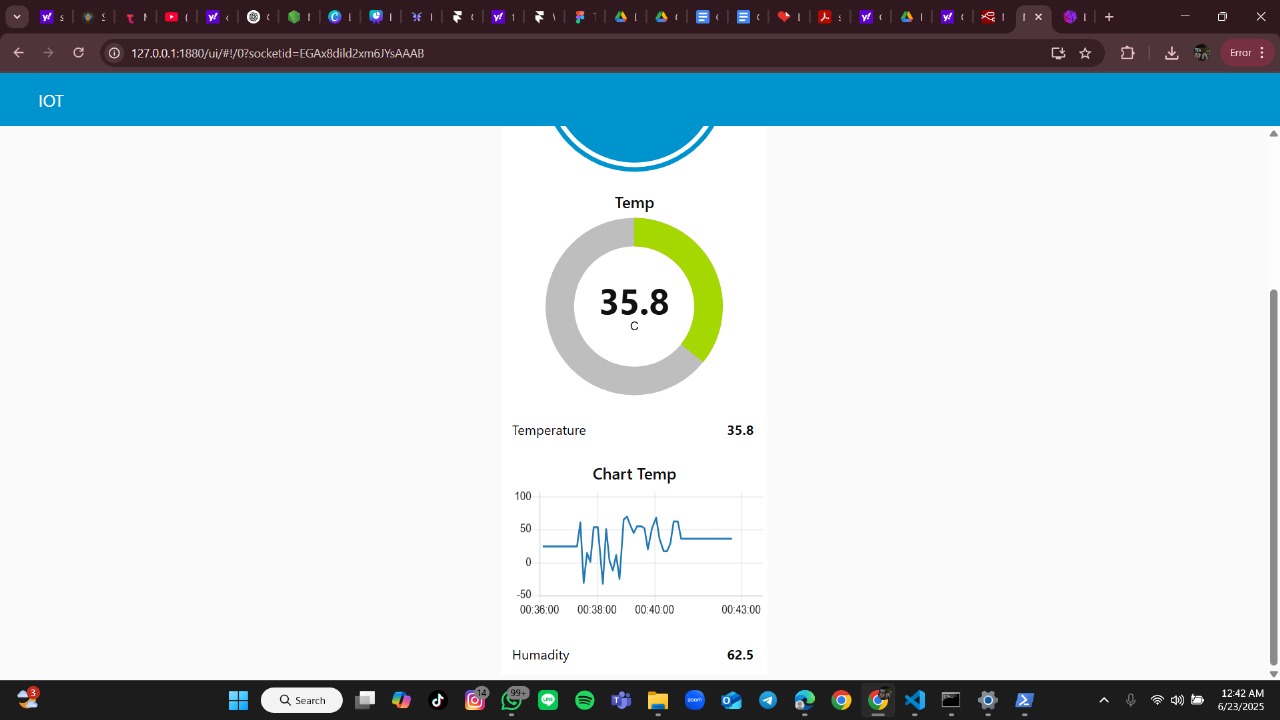
* **URL Server**: https://us-east-1-1.aws.cloud2.influxdata.com
* **Organization**: brawijaya university
* **Bucket**: iotfredlina
* **Measurement**: Temp

1. Menghubungkan node function ke influxdb out untuk menyimpan data, serta ke UI untuk visualisasi.
2. Melakukan deploy flow, dan menjalankan program untuk mulai mengirimkan data dari simulasi ke Node-RED dan InfluxDB.

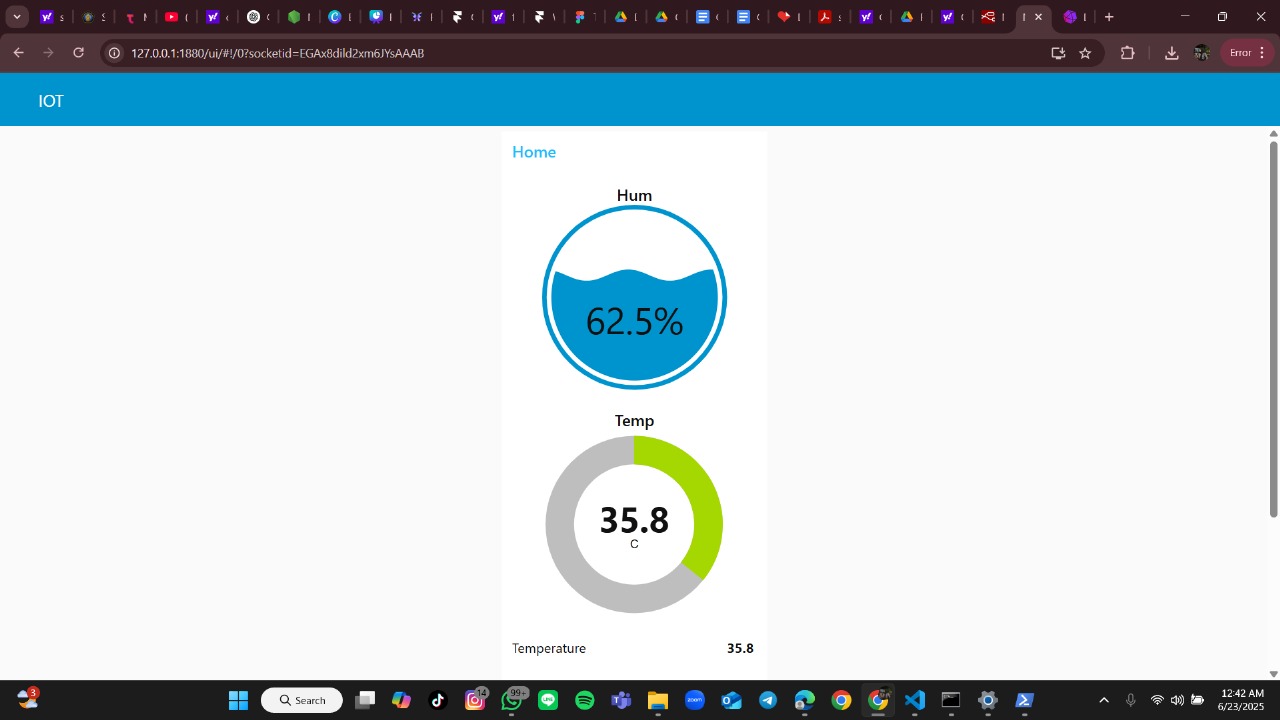
### **3. Hasil dan Pembahasan**

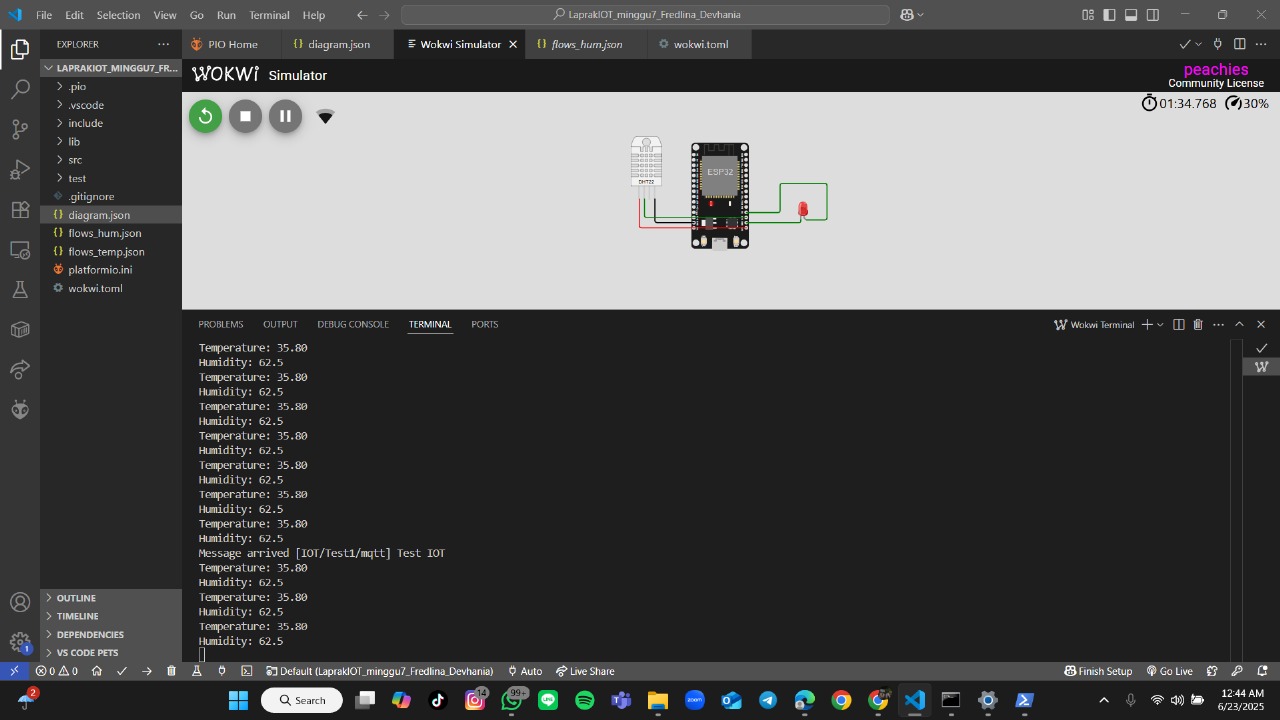
**3.1 Hasil Eksperimen**

**a. Sistem berhasil terhubung dengan Node-Red (Temp)**

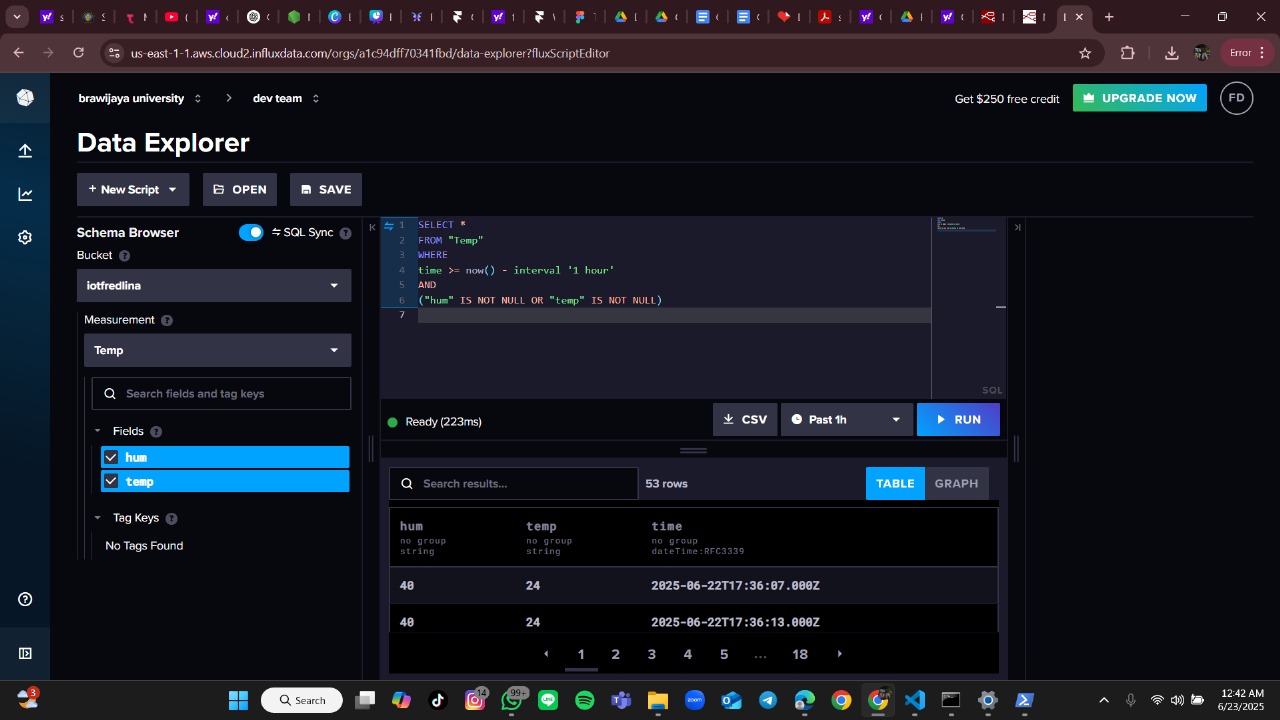
****

**b. Sistem berhasil terhubung dengan Node-Red (Hum)**

****

****

**c. Database di InfluxDB**

****

**3.2 Pembahasan**

Berdasarkan hasil eksperimen, sistem berhasil mensimulasikan pemantauan suhu menggunakan ESP32 dan sensor DHT22 melalui platform Wokwi. Meskipun tidak menggunakan protokol MQTT dalam proses pengiriman data, integrasi antara ESP32 dan Node-RED dapat berjalan dengan baik berkat konektivitas internal yang disimulasikan. Node-RED digunakan sebagai alat pemrosesan dan visualisasi data secara real-time, sementara InfluxDB berfungsi sebagai penyimpanan data historis berbasis deret waktu.

LED yang dikendalikan melalui kondisi suhu menunjukkan bahwa logika pemrosesan berjalan dengan benar. Flow yang dibuat di Node-RED juga menunjukkan responsivitas yang baik terhadap input data yang dikirimkan secara periodik. Dengan menggunakan simulasi ini, sistem IoT dapat dikembangkan dan diuji tanpa memerlukan perangkat keras fisik, sehingga sangat cocok untuk keperluan pembelajaran dan pengujian awal.

**4. 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 = "broker.emqx.io";//"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++) {**

**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);**

**} else {**

**digitalWrite(LED\_RED, LOW);**

**}**

**}**

**void reconnect() {**

**// 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");**

**// ... 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 Devhania",

"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": {}

}

Kode flows\_temp.json

[

{

"id": "8b88f65241fd2f4e",

"type": "tab",

"label": "IOT MQTT",

"disabled": false,

"info": "",

"env": []

},

{

"id": "b7be3fd5d19d83a0",

"type": "mqtt in",

"z": "8b88f65241fd2f4e",

"name": "MQTT data",

"topic": "IOT/Test1/temp",

"qos": "0",

"datatype": "auto-detect",

"broker": "fd4cbcbcd29913ab",

"nl": false,

"rap": true,

"rh": 0,

"inputs": 0,

"x": 390,

"y": 340,

"wires": [

[

"02e211b7bb351f90",

"5b6b7f5c5d77d1a6",

"f933b932defe4689",

"1de42b0ad1ab856b",

"bd388ae6b35881f9"

]

]

},

{

"id": "6dc2c7101b73f23e",

"type": "inject",

"z": "8b88f65241fd2f4e",

"name": "",

"props": [

{

"p": "payload"

},

{

"p": "topic",

"vt": "str"

}

],

"repeat": "",

"crontab": "",

"once": false,

"onceDelay": 0.1,

"topic": "",

"payload": "",

"payloadType": "date",

"x": 380,

"y": 200,

"wires": [

[

"02e211b7bb351f90"

]

]

},

{

"id": "02e211b7bb351f90",

"type": "ui\_text",

"z": "8b88f65241fd2f4e",

"group": "6cb91646811ccc32",

"order": 0,

"width": 0,

"height": 0,

"name": "",

"label": "Temperature",

"format": "{{msg.payload}}",

"layout": "row-spread",

"className": "",

"style": false,

"font": "",

"fontSize": 16,

"color": "#000000",

"x": 650,

"y": 320,

"wires": []

},

{

"id": "5b6b7f5c5d77d1a6",

"type": "ui\_gauge",

"z": "8b88f65241fd2f4e",

"name": "",

"group": "6cb91646811ccc32",

"order": 1,

"width": 0,

"height": 0,

"gtype": "donut",

"title": "Temp",

"label": "C",

"format": "{{value}}",

"min": 0,

"max": "100",

"colors": [

"#00b500",

"#e6e600",

"#ca3838"

],

"seg1": "",

"seg2": "",

"diff": false,

"className": "",

"x": 630,

"y": 360,

"wires": []

},

{

"id": "f933b932defe4689",

"type": "ui\_chart",

"z": "8b88f65241fd2f4e",

"name": "",

"group": "6cb91646811ccc32",

"order": 2,

"width": 0,

"height": 0,

"label": "Chart Temp",

"chartType": "line",

"legend": "false",

"xformat": "HH:mm:ss",

"interpolate": "linear",

"nodata": "",

"dot": false,

"ymin": "",

"ymax": "",

"removeOlder": 1,

"removeOlderPoints": "",

"removeOlderUnit": "3600",

"cutout": 0,

"useOneColor": false,

"useUTC": false,

"colors": [

"#1f77b4",

"#aec7e8",

"#ff7f0e",

"#2ca02c",

"#98df8a",

"#d62728",

"#ff9896",

"#9467bd",

"#c5b0d5"

],

"outputs": 1,

"useDifferentColor": false,

"className": "",

"x": 650,

"y": 400,

"wires": [

[]

]

},

{

"id": "1de42b0ad1ab856b",

"type": "debug",

"z": "8b88f65241fd2f4e",

"name": "debug 1",

"active": true,

"tosidebar": true,

"console": false,

"tostatus": false,

"complete": "payload",

"targetType": "msg",

"statusVal": "",

"statusType": "auto",

"x": 640,

"y": 240,

"wires": []

},

{

"id": "bd388ae6b35881f9",

"type": "function",

"z": "8b88f65241fd2f4e",

"name": "function 1",

"func": "var xx = msg.payload;\nvar Newobject = {};\nNewobject = {\n \"temp\": msg.payload.toString()\n}\nmsg.payload = Newobject;\nreturn msg;\n",

"outputs": 1,

"timeout": 0,

"noerr": 0,

"initialize": "",

"finalize": "",

"libs": [],

"x": 640,

"y": 480,

"wires": [

[

"efc79a152f28c53b"

]

]

},

{

"id": "efc79a152f28c53b",

"type": "influxdb out",

"z": "8b88f65241fd2f4e",

"influxdb": "2fe9efe09dfa8dfd",

"name": "InfluxDB",

"measurement": "Temp",

"precision": "",

"retentionPolicy": "",

"database": "database",

"precisionV18FluxV20": "s",

"retentionPolicyV18Flux": "",

"org": "organisation",

"bucket": "NodeRed",

"x": 840,

"y": 480,

"wires": []

},

{

"id": "fd4cbcbcd29913ab",

"type": "mqtt-broker",

"name": "",

"broker": "broker.emqx.io",

"port": 1883,

"clientid": "",

"autoConnect": true,

"usetls": false,

"protocolVersion": 4,

"keepalive": 60,

"cleansession": true,

"autoUnsubscribe": true,

"birthTopic": "",

"birthQos": "0",

"birthRetain": "false",

"birthPayload": "",

"birthMsg": {},

"closeTopic": "",

"closeQos": "0",

"closeRetain": "false",

"closePayload": "",

"closeMsg": {},

"willTopic": "",

"willQos": "0",

"willRetain": "false",

"willPayload": "",

"willMsg": {},

"userProps": "",

"sessionExpiry": ""

},

{

"id": "6cb91646811ccc32",

"type": "ui\_group",

"name": "IOT",

"tab": "acc17c2264463766",

"order": 1,

"disp": true,

"width": 6,

"collapse": false,

"className": ""

},

{

"id": "2fe9efe09dfa8dfd",

"type": "influxdb",

"hostname": "127.0.0.1",

"port": 8086,

"protocol": "http",

"database": "database",

"name": "InfluxDB",

"usetls": false,

"tls": "",

"influxdbVersion": "2.0",

"url": "https://us-east-1-1.aws.cloud2.influxdata.com/",

"timeout": 10,

"rejectUnauthorized": true

},

{

"id": "acc17c2264463766",

"type": "ui\_tab",

"name": "Home",

"icon": "dashboard",

"disabled": false,

"hidden": false

}

]

Kode flows\_hum.json

[

{

"id": "9ee437252d2b0c81",

"type": "tab",

"label": "test",

"disabled": false,

"info": "",

"env": []

},

{

"id": "f1172be69061b358",

"type": "mqtt in",

"z": "9ee437252d2b0c81",

"name": "MQTT data",

"topic": "IOT/Test1/hum",

"qos": "0",

"datatype": "auto-detect",

"broker": "fd4cbcbcd29913ab",

"nl": false,

"rap": true,

"rh": 0,

"inputs": 0,

"x": 230,

"y": 260,

"wires": [

[

"4db5b2a12e92f52f",

"3fbf3dbcbd4b18e1",

"9fc136140df8a5b4",

"3923030809a0bd4a"

]

]

},

{

"id": "0b9c798ef7faf830",

"type": "inject",

"z": "9ee437252d2b0c81",

"name": "",

"props": [

{

"p": "payload"

},

{

"p": "topic",

"vt": "str"

}

],

"repeat": "",

"crontab": "",

"once": false,

"onceDelay": 0.1,

"topic": "",

"payload": "",

"payloadType": "date",

"x": 220,

"y": 120,

"wires": [

[

"4db5b2a12e92f52f"

]

]

},

{

"id": "4db5b2a12e92f52f",

"type": "ui\_text",

"z": "9ee437252d2b0c81",

"group": "66596c22c53900d8",

"order": 0,

"width": 0,

"height": 0,

"name": "",

"label": "Humadity",

"format": "{{msg.payload}}",

"layout": "row-spread",

"className": "",

"style": false,

"font": "",

"fontSize": 16,

"color": "#000000",

"x": 480,

"y": 240,

"wires": []

},

{

"id": "3fbf3dbcbd4b18e1",

"type": "ui\_gauge",

"z": "9ee437252d2b0c81",

"name": "",

"group": "66596c22c53900d8",

"order": 1,

"width": 0,

"height": 0,

"gtype": "wave",

"title": "Hum",

"label": "%",

"format": "{{value}}",

"min": 0,

"max": "100",

"colors": [

"#00b500",

"#e6e600",

"#ca3838"

],

"seg1": "",

"seg2": "",

"diff": false,

"className": "",

"x": 470,

"y": 280,

"wires": []

},

{

"id": "9fc136140df8a5b4",

"type": "debug",

"z": "9ee437252d2b0c81",

"name": "debug 2",

"active": true,

"tosidebar": true,

"console": false,

"tostatus": false,

"complete": "payload",

"targetType": "msg",

"statusVal": "",

"statusType": "auto",

"x": 480,

"y": 160,

"wires": []

},

{

"id": "3923030809a0bd4a",

"type": "function",

"z": "9ee437252d2b0c81",

"name": "function 2",

"func": "var xx = msg.payload;\nvar Newobject = {};\nNewobject = {\n \"hum\": msg.payload.toString()\n}\nmsg.payload = Newobject;\nreturn msg;\n",

"outputs": 1,

"timeout": 0,

"noerr": 0,

"initialize": "",

"finalize": "",

"libs": [],

"x": 480,

"y": 400,

"wires": [

[

"95ab26848d5e0e2e"

]

]

},

{

"id": "95ab26848d5e0e2e",

"type": "influxdb out",

"z": "9ee437252d2b0c81",

"influxdb": "2fe9efe09dfa8dfd",

"name": "InfluxDB",

"measurement": "Temp",

"precision": "",

"retentionPolicy": "",

"database": "database",

"precisionV18FluxV20": "s",

"retentionPolicyV18Flux": "",

"org": "organisation",

"bucket": "NodeRed",

"x": 680,

"y": 400,

"wires": []

},

{

"id": "fd4cbcbcd29913ab",

"type": "mqtt-broker",

"name": "",

"broker": "broker.emqx.io",

"port": 1883,

"clientid": "",

"autoConnect": true,

"usetls": false,

"protocolVersion": 4,

"keepalive": 60,

"cleansession": true,

"autoUnsubscribe": true,

"birthTopic": "",

"birthQos": "0",

"birthRetain": "false",

"birthPayload": "",

"birthMsg": {},

"closeTopic": "",

"closeQos": "0",

"closeRetain": "false",

"closePayload": "",

"closeMsg": {},

"willTopic": "",

"willQos": "0",

"willRetain": "false",

"willPayload": "",

"willMsg": {},

"userProps": "",

"sessionExpiry": ""

},

{

"id": "66596c22c53900d8",

"type": "ui\_group",

"name": "Home",

"tab": "33e1fa2b35d5f28e",

"order": 1,

"disp": true,

"width": 6,

"collapse": false,

"className": ""

},

{

"id": "2fe9efe09dfa8dfd",

"type": "influxdb",

"hostname": "127.0.0.1",

"port": 8086,

"protocol": "http",

"database": "database",

"name": "InfluxDB",

"usetls": false,

"tls": "",

"influxdbVersion": "2.0",

"url": "https://us-east-1-1.aws.cloud2.influxdata.com/",

"timeout": 10,

"rejectUnauthorized": true

},

{

"id": "33e1fa2b35d5f28e",

"type": "ui\_tab",

"name": "IOT",

"icon": "dashboard",

"disabled": false,

"hidden": false

}

]