**LAPORAN PRAKTIKUM INTERNET OF THINGS (IOT)**

**SIMULASI MONITORING SUHU DAN KELEMBAPAN DENGAN NODE-RED DAN INFLUXDB**

**Oleh**

*Maynanda Elisa Pasya T*

*Fakultas Vokasi, Universitas Brawijaya*

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

**ABSTRAK**

Praktikum ini bertujuan untuk mensimulasikan sistem monitoring suhu dan kelembapan menggunakan Node-RED sebagai alat pemrosesan data dan InfluxDB sebagai media penyimpanan data berbasis waktu. Data sensor disimulasikan dan dikirimkan melalui MQTT broker untuk kemudian diproses dan divisualisasikan dalam bentuk grafik pada dashboard Node-RED. Hasil implementasi menunjukkan bahwa sistem mampu menerima, menyimpan, dan menampilkan data secara real-time dengan efisien. Node-RED mempermudah proses pengelolaan data melalui konsep flow-based programming, sementara InfluxDB memungkinkan penyimpanan data time-series secara optimal. Sistem ini berpotensi dikembangkan untuk kebutuhan monitoring lingkungan dalam skala lebih besar maupun sebagai bagian dari sistem smart home.

**Kata Kunci**: Internet of Things, Node-RED, InfluxDB, MQTT, suhu, kelembapan, monitoring real-time.

**ABSTRACT**

This practicum aims to simulate a temperature and humidity monitoring system using Node-RED as the data processing tool and InfluxDB as the time-series data storage. Sensor data is simulated and transmitted via an MQTT broker, then processed and visualized in graphical form on the Node-RED dashboard. The implementation results show that the system can efficiently receive, store, and display real-time data. Node-RED simplifies data management through its flow-based programming approach, while InfluxDB enables optimal time-series data storage. This system has the potential to be developed for large-scale environmental monitoring or as part of a smart home system.

**Keywords**: Internet of Things, Node-RED, InfluxDB, MQTT, temperature, humidity, real-time monitoring.

1. **PENDAHULUAN**
   1. **Latar Belakang**

Internet of Things (IoT) merupakan konsep yang memungkinkan perangkat elektronik saling terhubung dan saling berkomunikasi melalui jaringan internet. Salah satu aplikasi utama dari IoT adalah monitoring lingkungan secara real-time, seperti suhu dan kelembapan. Dalam dunia industri, pemantauan suhu dan kelembapan sangat penting untuk menjaga kestabilan proses produksi dan penyimpanan.

Node-RED adalah tool berbasis flow programming yang memudahkan integrasi dan pemrosesan data sensor dalam sistem IoT, sedangkan InfluxDB adalah database time-series yang cocok digunakan untuk menyimpan data lingkungan yang bersifat kontinu. Dengan kombinasi keduanya, kita dapat melakukan simulasi pemantauan suhu dan kelembapan secara efektif.

**1.2 Tujuan Praktikum**

Praktikum ini bertujuan untuk:

* Mempelajari penggunaan Node-RED untuk mengelola dan menampilkan data sensor.
* Mengimplementasikan penyimpanan data suhu dan kelembapan ke dalam InfluxDB.
* Menganalisis data lingkungan secara real-time melalui dashboard yang interaktif.

1. **METODOLOGI**

**2.1 Alat dan Bahan**

* **Alat:**

1. Laptop/computer
2. Wokwi Simulator
3. Node-RED (local)
4. InfluxDB

* **Bahan:**

1. ESP32 Devkit V1.
2. Sensor DHT22
3. Koneksi Wi-Fi (Wokwi Guest)
4. Node-RED nodes tambahan seperti:

* node-red-dashboard (untuk UI)
* node-red-contrib-influxdb (untuk koneksi ke InfluxDB)
* function, inject, gauge, chart, dll

**2.2 Langkah Implementasi**

Berikut adalah langkah-langkah implementasi secara rinci dalam melakukan simulasi sistem monitoring suhu dan kelembapan berbasis ESP32 dan DHT22 menggunakan Wokwi, Node-RED, dan InfluxDB:

1. **Instalasi Tools**

* Install Node-RED dan jalankan di localhost (biasanya pada <http://localhost:1880>).
* Install dan jalankan InfluxDB pada sistem.
* Pastikan MQTT broker aktif dan dapat menerima data dari publisher.

1. **Simulasi Data Sensor**

* Gunakan function node pada Node-RED untuk menghasilkan data suhu dan kelembapan secara acak (jika tidak menggunakan sensor fisik).
* Kirim data tersebut ke MQTT topic tertentu (contoh: sensor/suhu\_kelembapan).

1. **Pemrosesan Data di Node-RED**

* Gunakan node MQTT subscribe untuk menerima data.
* Pisahkan data suhu dan kelembapan menggunakan function node.
* Format data agar sesuai dengan struktur InfluxDB (menggunakan node influxdb out).

1. **Penyimpanan ke InfluxDB**

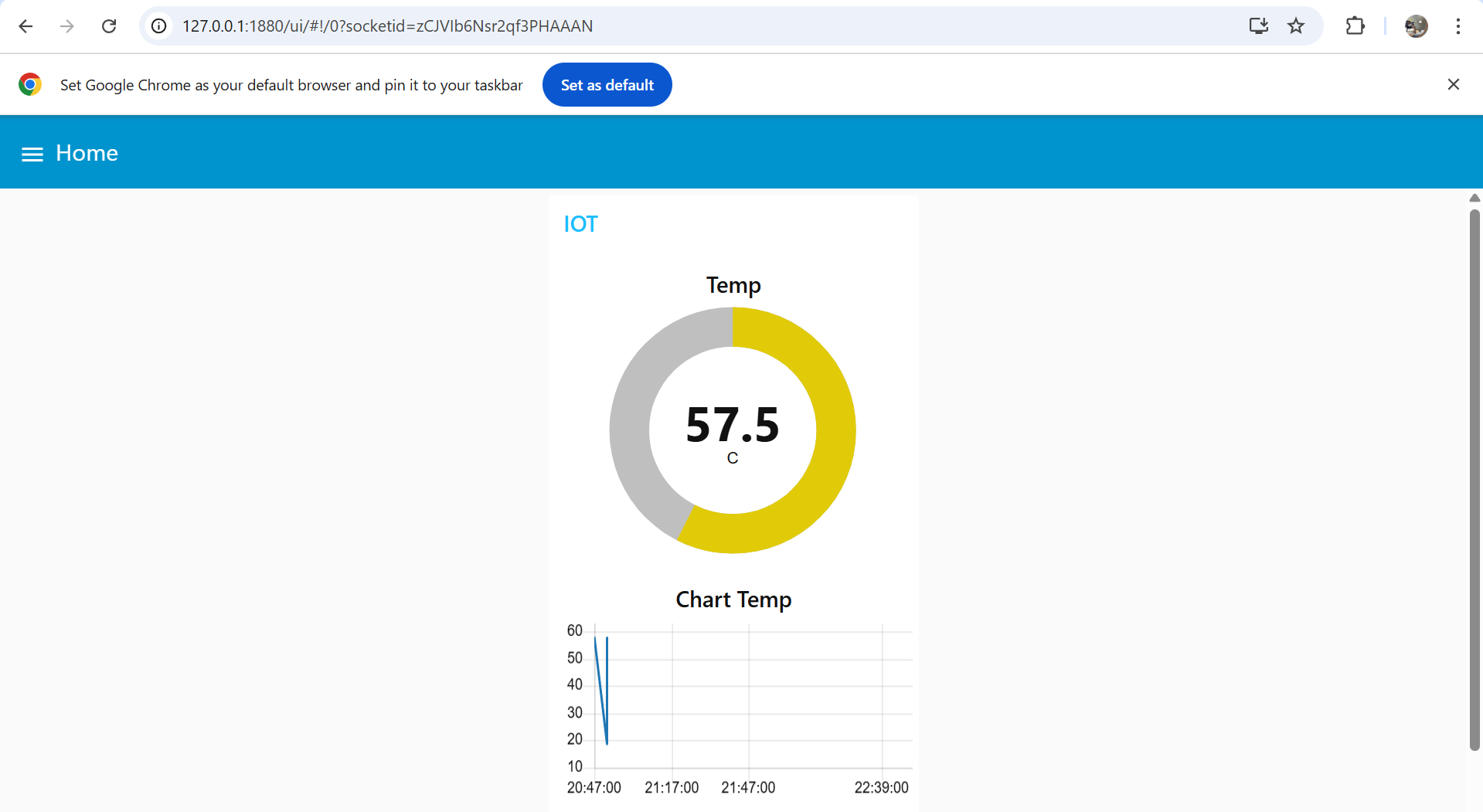
* Buat database bernama iot\_monitoring.
* Konfigurasikan node influxdb out untuk menyimpan data suhu dan kelembapan ke dalam measurement tertentu.

1. **Visualisasi Data**

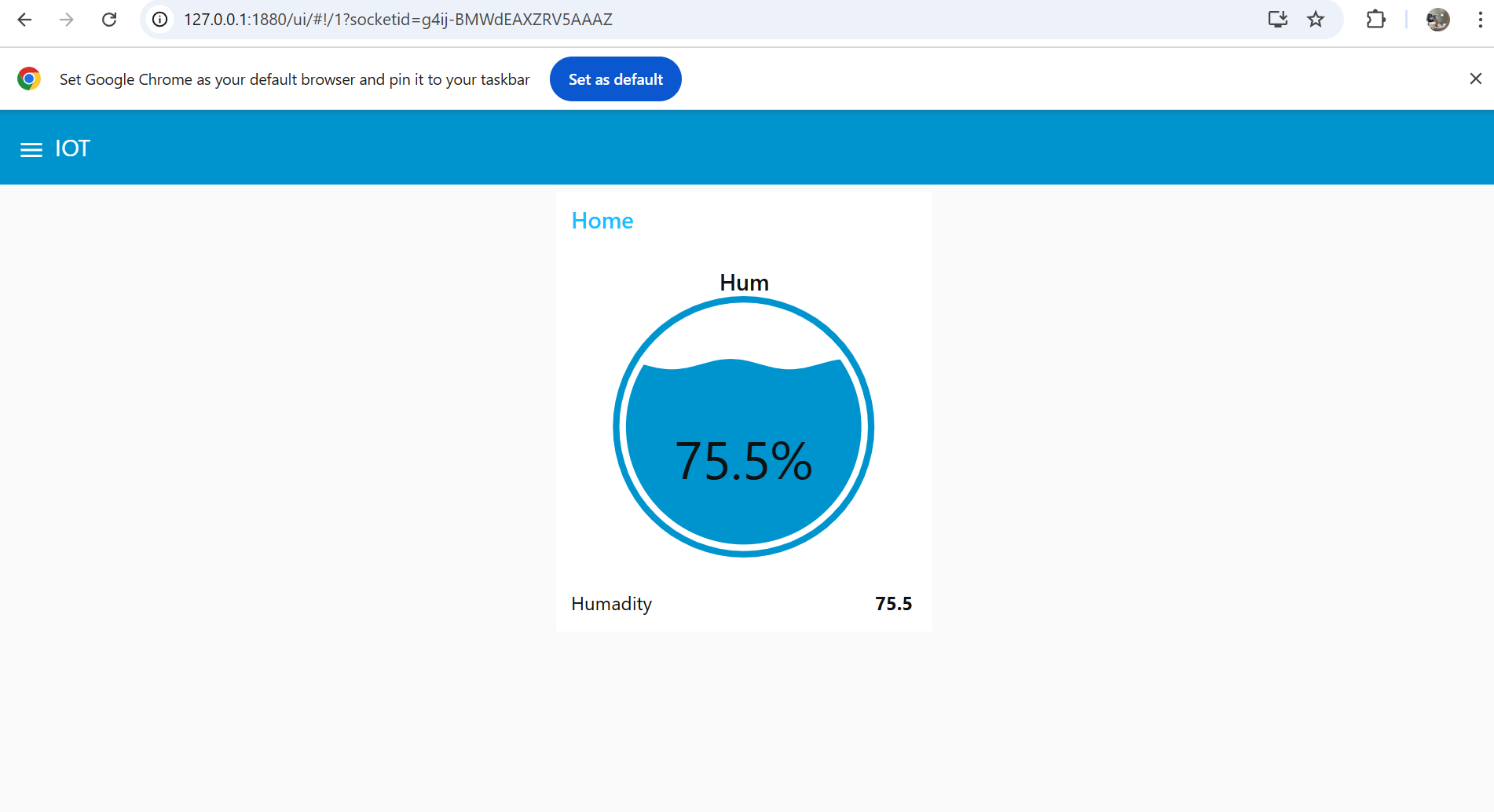
* Gunakan node ui\_chart dari dashboard Node-RED untuk menampilkan grafik suhu dan kelembapan.

1. **HASIL DAN PEMBAHASAN**

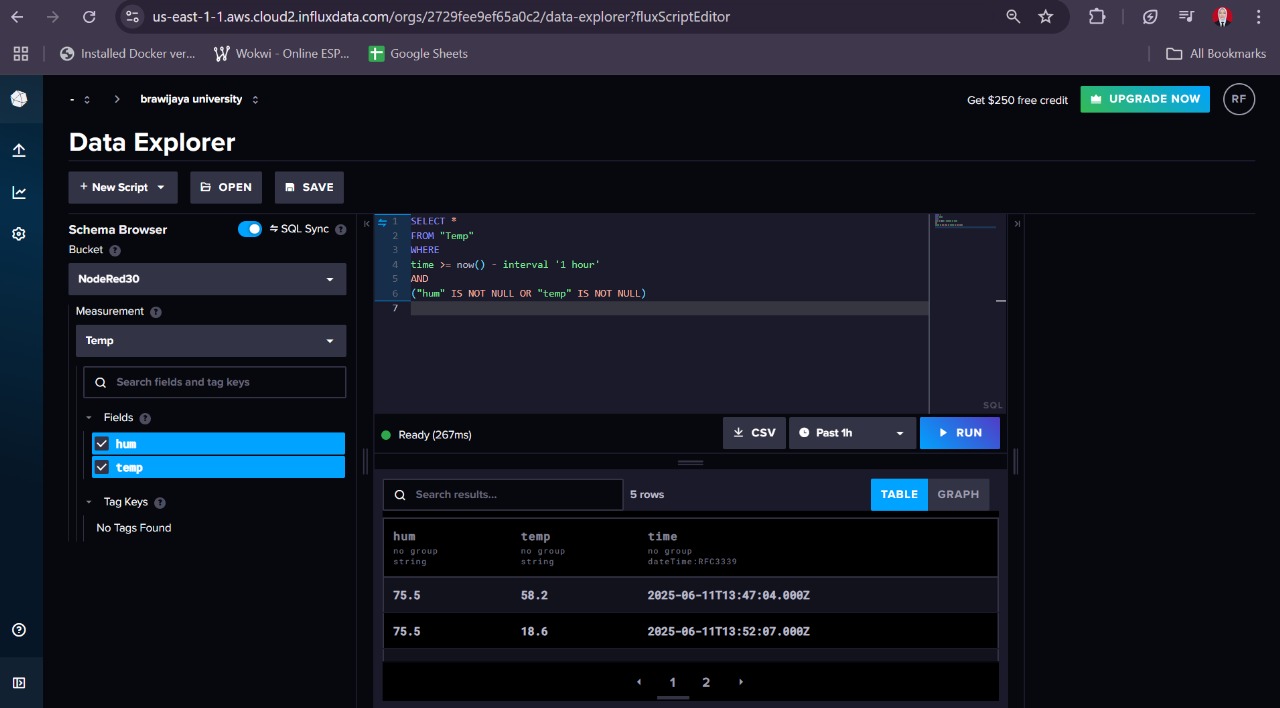
**3.1 Hasil UI Temperature di NodeRed**

****

* 1. **Hasil UI Humadity di NodeRed**

****

* 1. **Hasil Database menggunakan InfluxDB**



* 1. **Pembahasan**

Setelah implementasi, sistem berhasil menerima dan menampilkan data suhu serta kelembapan dari simulator secara real-time dalam bentuk grafik garis pada dashboard. Data tersebut juga tersimpan dengan baik di InfluxDB dalam format time-series, memungkinkan pencatatan perubahan data berdasarkan waktu. Node-RED terbukti mampu memproses data dengan cepat dan efisien berkat latensi yang rendah, sementara InfluxDB mendukung penyimpanan data secara optimal dan mendukung kueri berdasarkan rentang waktu tertentu. Secara keseluruhan, penggunaan Node-RED memudahkan pembuatan alur data tanpa banyak kode, dan InfluxDB sangat sesuai untuk menangani data sensor karena performanya yang tinggi.

1. **LAMPIRAN**
   1. **Kode program Wokwi Temp dan Hum**

#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++) { //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);

  }

}

* 1. **Kode diagram.json**

{

  "version": 1,

  "author": "Maynanda",

  "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": {}

}

* 1. **Kode Json NodeRed Temperature**

[{"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}]

* 1. **Kode Json NodeRed Humadity**

[{"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}]