

# **Laporan Praktikum Rangkaian Pengukuran Suhu Kelembapan,dan Cuaca Menggunakan Website Open Weather App**

*Maradu Denny S. Hutahae*

*Fakultas Vokasi , Universitas Brawijaya*

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

## **Abstrak :**

Penelitian ini memiliki tujuan untuk merancang sekaligus mengimplementasikan sebuah sistem pemantauan informasi cuaca berbasis mikrokontroler ESP32. Sistem ini mampu menampilkan data suhu dan kondisi cuaca secara waktu nyata (real-time) melalui integrasi sensor fisik dan API berbasis internet. Komponen utama yang digunakan mencakup ESP32 sebagai inti pengendali, sensor DHT11 untuk mendeteksi suhu dan kelembapan, serta layar LCD 16x2 berprotokol I2C sebagai media tampilan. Sistem bekerja dengan menggabungkan data suhu lokal dari sensor DHT11 dan data cuaca eksternal yang diperoleh dari layanan OpenWeatherMap API menggunakan koneksi Wi-Fi. Pengambilan data berlangsung secara berkala setiap 60 detik. Informasi suhu dan kondisi cuaca ditampilkan secara langsung di layar LCD, memudahkan pengguna untuk memantau cuaca tanpa memerlukan perangkat tambahan. Proses penguraian data JSON dari API dilakukan menggunakan pustaka ArduinoJson, yang membantu mempercepat dan mempermudah ekstraksi data secara akurat. Berdasarkan hasil implementasi, sistem terbukti berjalan dengan baik dalam menyajikan informasi suhu dan kelembapan secara langsung, serta dapat digunakan sebagai prototipe awal dalam pengembangan sistem IoT untuk pemantauan cuaca yang lebih lanjut.

**Kata Kunci :** *Arduino, ESP32, Wokwi, open weather app , DHT22, Internet Of Things.*

## **Abstract:**

This research aims to design and implement a weather monitoring system based on the ESP32 microcontroller, capable of displaying temperature and weather conditions in real-time by integrating physical sensors with an online API. The hardware components used include the ESP32 microcontroller, a DHT11 temperature and humidity sensor, and a 16x2 I2C LCD display. The system combines local temperature readings from the DHT11 sensor with global weather data retrieved from the OpenWeatherMap API via a Wi-Fi connection. Data is collected periodically every 60 seconds. Temperature and weather descriptions are displayed directly on the LCD screen, allowing users to monitor conditions quickly without the need for other devices. JSON data parsing from the API is handled using the ArduinoJson library to ensure accurate and efficient data extraction. The implementation results show that the system functions effectively in displaying real-time weather and humidity information and serves as a potential prototype for more advanced IoT-based weather monitoring systems.

**Key Word :** *Arduino, ESP32, Wokwi, Open Weather App , Internet Of Thi*

## PENDAHULUAN

Perkembangan teknologi Internet of Things (IoT) semakin pesat dan telah memberikan dampak signifikan dalam berbagai bidang, termasuk sistem monitoring cuaca. Informasi cuaca yang akurat dan real-time sangat penting untuk berbagai aktivitas, baik di bidang pertanian, perikanan, transportasi, hingga kehidupan sehari-hari. Namun, tidak semua masyarakat memiliki akses terhadap perangkat atau aplikasi canggih untuk mendapatkan informasi tersebut secara cepat dan mudah.

Dalam hal ini, mikrokontroler ESP32 menjadi solusi alternatif yang efisien dan terjangkau untuk membangun sistem monitoring berbasis IoT. Dengan dukungan koneksi Wi-Fi dan kemampuan pemrosesan yang mumpuni, ESP32 dapat digunakan untuk mengakses data cuaca secara daring melalui API publik seperti OpenWeatherMap, serta menggabungkannya dengan data dari sensor fisik seperti DHT11.

Pada proyek ini, dirancang sebuah prototipe sistem monitoring cuaca sederhana yang mampu menampilkan suhu lokal, kelembapan, dan deskripsi cuaca secara real-time menggunakan ESP32, sensor DHT11, dan LCD I2C 16x2. Sistem ini diharapkan dapat menjadi solusi praktis yang dapat digunakan dalam skala rumah tangga maupun skala kecil lainnya, sekaligus sebagai dasar pengembangan sistem monitoring cuaca yang lebih kompleks di masa depan.

## METODOLOGI

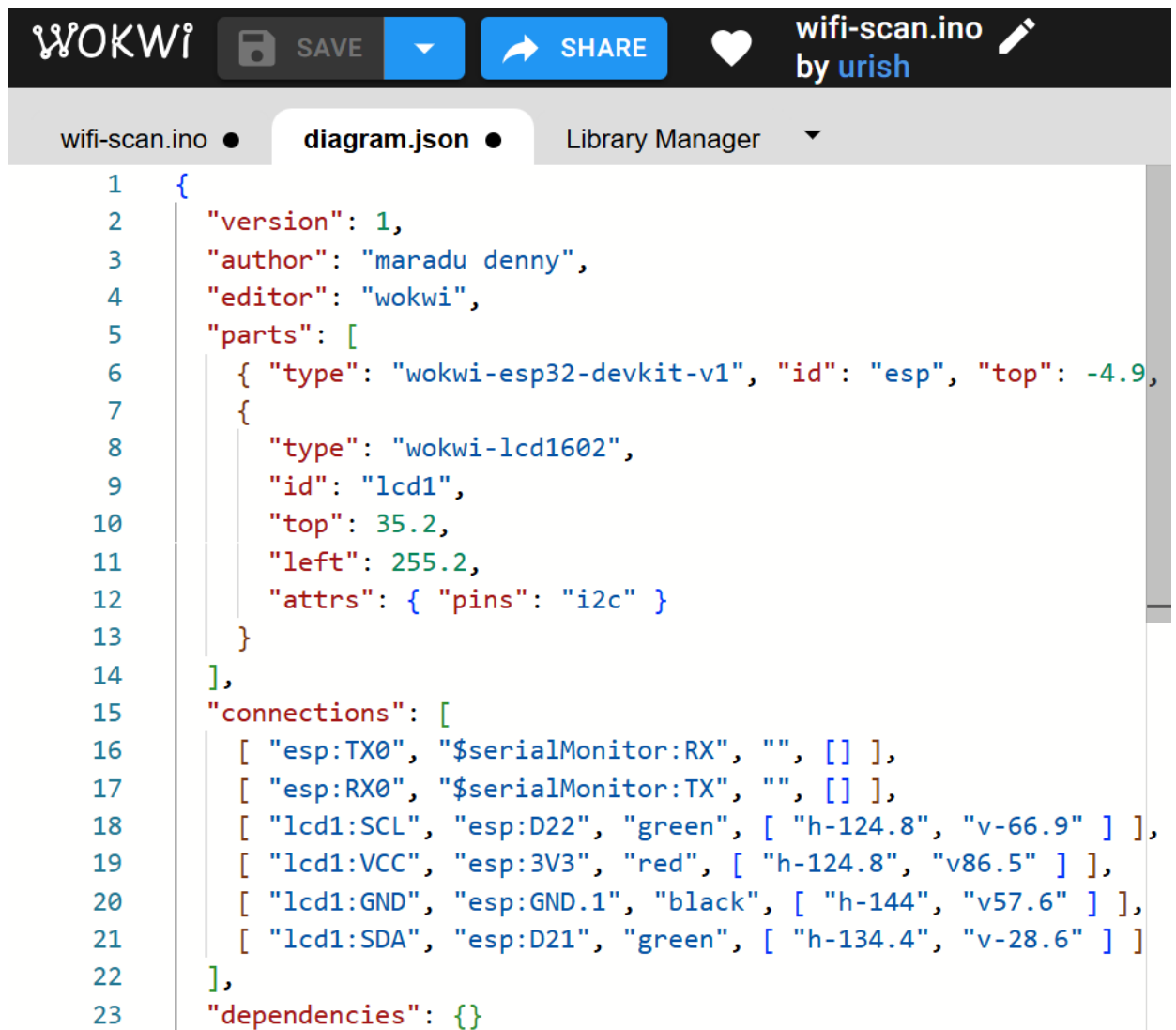
### A. Alat dan Bahan

Mikrokontroler ESP32, Library Blynk, Sensor DHT, Kabel jumper, Breadboard, software Visual Studio Code, Website Blynk, PlatformIo, dan platform Wokwi .

### B. Langkah Perancangan

1. Pilih ESP32 sebagai mikrokontroler pada situs [Wokwi](#).
2. Tambahkan komponen pendukung untuk menyusun rangkaian yang lengkap. Pastikan seluruh komponen tersambung dengan benar ke ESP32.
3. Buka website wokwi pada browser

4. Copy script diagram.json di website wokwi.



The screenshot shows the Wokwi web interface. At the top, there's a header with the Wokwi logo, a 'SAVE' button, a 'SHARE' button, a heart icon, and the text 'wifi-scan.ino by urish'. Below the header, there's a tab bar with 'wifi-scan.ino', 'diagram.json' (which is selected), and 'Library Manager'. The main area displays the content of 'diagram.json' with line numbers 1 through 23 on the left. The JSON code is as follows:

```
1  {
2    "version": 1,
3    "author": "maradu denny",
4    "editor": "wokwi",
5    "parts": [
6      { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": -4.9,
7        {
8          "type": "wokwi-lcd1602",
9          "id": "lcd1",
10         "top": 35.2,
11         "left": 255.2,
12         "attrs": { "pins": "i2c" }
13       }
14     ],
15     "connections": [
16       [ "esp:TX0", "$serialMonitor:RX", "", [ ] ],
17       [ "esp:RX0", "$serialMonitor:TX", "", [ ] ],
18       [ "lcd1:SCL", "esp:D22", "green", [ "h-124.8", "v-66.9" ] ],
19       [ "lcd1:VCC", "esp:3V3", "red", [ "h-124.8", "v86.5" ] ],
20       [ "lcd1:GND", "esp:GND.1", "black", [ "h-144", "v57.6" ] ],
21       [ "lcd1:SDA", "esp:D21", "green", [ "h-134.4", "v-28.6" ] ]
22     ],
23     "dependencies": {}
```

5. Buka website openweather pada browser dan get API keys.

6. Copy script diagram.json di website wokwi.

```
wokwi
SAVE
SHARE
wifi-scan.ino
by urish

wifi-scan.ino • diagram.json • Library Manager
1 #include <Wire.h>
2 #include <LiquidCrystal_I2C.h>
3 #include <WiFi.h>
4 #include <HTTPClient.h>
5 #include "DHT.h"
6
7 // Wi-Fi credentials
8 const char* ssid = "Wokwi-GUEST";
9 const char* password = "";
10
11 // OpenWeatherMap API
12 String apiKey = "7139e220a9966e69475dd62ef063b3ff";
13 String city = "Malang";
14 String units = "metric";
15 String server = "http://api.openweathermap.org/data/2.5/weather?";
16
17 // LCD setup
18 LiquidCrystal_I2C lcd(0x27, 16, 2);
19
20 // DHT11 setup
21 #define DHTPIN 4
22 #define DHTTYPE DHT11
23 DHT dht(DHTPIN, DHTTYPE);
24
25 void setup() {
26   Serial.begin(115200);
27
28   // LCD init
29   lcd.init();
30   lcd.backlight();
31   lcd.setCursor(0, 0);
32   lcd.print("Weather Info:");
33   delay(1000);
34
35   // Wi-Fi connection
36   WiFi.begin(ssid, password);
37   lcd.setCursor(0, 1);
38   lcd.print("Connecting...");
39   while (WiFi.status() != WL_CONNECTED) {
40     delay(1000);
41     Serial.println("Connecting to WiFi...");
42   }
43
44   lcd.clear();
45   lcd.setCursor(0, 0);
46   lcd.print("Connected!");
47   delay(2000);
48   lcd.clear();
49
50   // DHT11 start
51   dht.begin();
52   delay(2000); // Tambahkan delay agar sensor siap
53 }
54
55 void loop() {
56   if (WiFi.status() == WL_CONNECTED) {
57     HTTPClient http;
58     http.begin(server);
59     int httpCode = http.GET();
60
61     if (httpCode > 0) {
62       String payload = http.getString();
63       Serial.println(payload);
64
65       // Ambil suhu dari API
66       int tempIndex = payload.indexOf("temp");
67       String temp = payload.substring(tempIndex + 6, payload.inde
68
69   // Ambil deskripsi cuaca
70   int descIndex = payload.indexOf("description");
71   String desc = payload.substring(descIndex + 14, payload.in
72
73   // Baca kelembapan dari DHT11
74   float humidity = dht.readHumidity();
75   if (isnan(humidity)) {
76     delay(1000); // tunggu 1 detik
77     humidity = dht.readHumidity(); // coba baca lagi
78   }
79
80   // Tampilkan data di LCD
81   lcd.clear();
82   lcd.setCursor(0, 0);
83   lcd.print("T: " + temp + "C H: ");
84
85   lcd.setCursor(0, 1);
86   if (desc.length() > 16) {
87     lcd.print(desc.substring(0, 16));
88   } else {
89     lcd.print(desc);
90   }
91 }
```

```

90     }
91
92     } else {
93         Serial.println("Error on HTTP request");
94         lcd.clear();
95         lcd.setCursor(0, 0);
96         lcd.print("HTTP Error");
97     }
98
99     http.end();
100 } else {
101     Serial.println("WiFi disconnected");
102     lcd.clear();
103     lcd.setCursor(0, 0);
104     lcd.print("WiFi lost...");
105 }
106
107 delay(60000); // Update every 60 detik
108 }
109

```

7. Jalankan simulasi dan jika tidak dapat dirunning maka copy semua kode yang telah ditulis dan masukkan ke dalam VSCode Arduino.

## HASIL DAN PEMBAHASAN

Setelah melalui proses perancangan perangkat keras dan pemrograman perangkat lunak, sistem monitoring cuaca berbasis ESP32 berhasil diimplementasikan dengan baik. Sistem ini terdiri dari beberapa komponen utama, yaitu mikrokontroler ESP32, sensor suhu dan kelembapan DHT11, serta LCD I2C 16x2 sebagai media tampilan informasi. Ketika sistem pertama kali dinyalakan, ESP32 akan langsung mencoba terhubung ke jaringan Wi-Fi yang telah ditentukan. Setelah koneksi berhasil, sistem akan melakukan permintaan data ke API OpenWeatherMap untuk memperoleh data cuaca berdasarkan lokasi yang telah ditentukan, dalam hal ini adalah kota Malang. Secara bersamaan, sensor DHT11 juga akan membaca suhu dan kelembapan udara dari lingkungan sekitar perangkat.

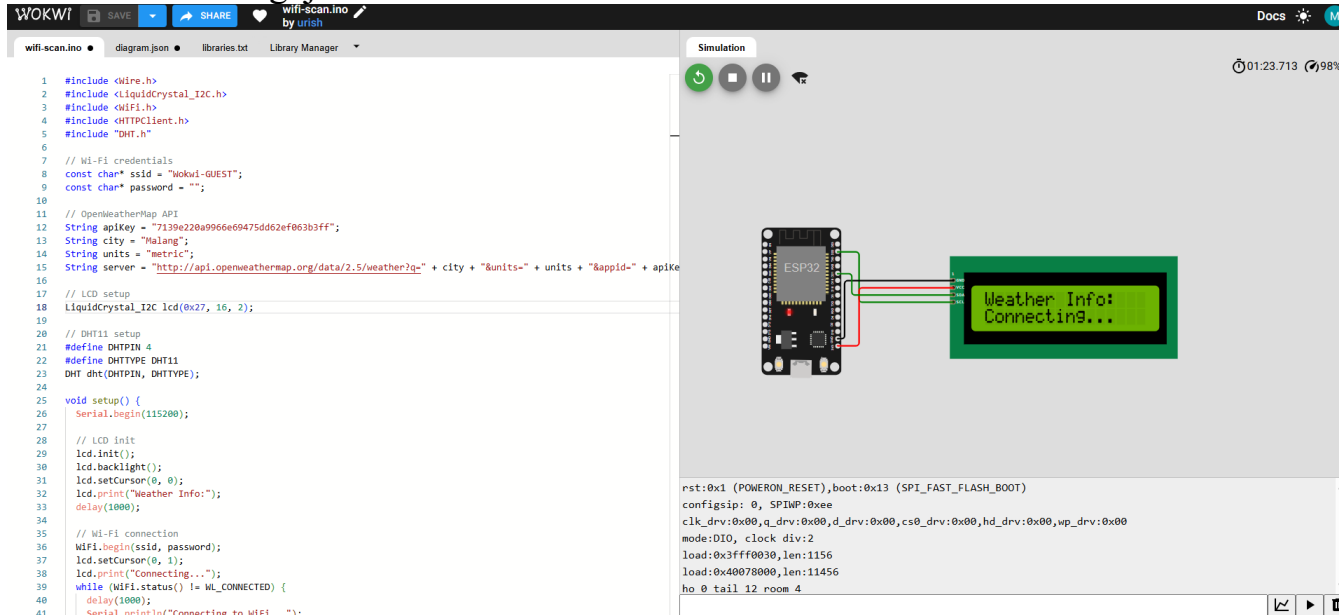
Informasi yang ditampilkan pada layar LCD terbagi menjadi dua baris. Baris pertama menampilkan suhu (yang diambil dari API) dan kelembapan (yang dibaca dari DHT11), sedangkan baris kedua menampilkan deskripsi cuaca, seperti "light rain" atau "clear sky". Proses pembaruan data dilakukan secara otomatis setiap 60 detik, sehingga informasi yang ditampilkan tetap up-to-date. Berdasarkan hasil pengujian di lingkungan nyata, tampilan pada LCD menunjukkan informasi yang dapat dibaca dengan jelas, serta sistem mampu mempertahankan koneksi Wi-Fi dengan stabil selama pengoperasian.

Secara umum, integrasi antara data lokal dan data daring dapat berjalan dengan baik. ESP32 mampu mengambil data cuaca secara real-time dari OpenWeatherMap menggunakan metode HTTP GET, kemudian memproses data JSON untuk mengekstrak nilai suhu dan deskripsi cuaca. Library ArduinoJson digunakan untuk mempermudah parsing data. Di sisi lain, sensor DHT11 memberikan data kelembapan dengan cukup baik meskipun dengan tingkat akurasi yang masih terbatas. Untuk kebutuhan monitoring dasar, sensor ini sudah memadai.

Namun demikian, terdapat beberapa keterbatasan dalam sistem ini. Salah satunya adalah keterbatasan jumlah karakter pada layar LCD, sehingga deskripsi cuaca yang panjang harus dipotong agar dapat ditampilkan dalam satu baris. Selain itu, sistem belum dilengkapi dengan mekanisme pemulihan otomatis jika koneksi Wi-Fi terputus di tengah pengambilan data. Di sisi sensor, DHT11 tidak cocok untuk aplikasi yang memerlukan presisi tinggi karena keterbatasan resolusi dan rentang pembacaannya. Meski begitu, secara keseluruhan sistem ini menunjukkan kinerja yang stabil dan dapat dijadikan sebagai prototipe awal untuk pengembangan sistem IoT monitoring cuaca yang lebih kompleks di masa mendatang.

# LAMPIRAN

## 1. Dokumentasi Pengujian



## 2. Link Wokwi :

<https://wokwi.com/projects/434665332883366913>





