IMPLEMENTASI SISTEM MONITORING CUACA REAL-TIME MENGGUNAKAN ESP32 DAN API OPENWEATHERMAP

Rumiris Butarbutar
Program Studi Teknologi Informasi, Fakultas Vokasi,Universitas Brawijaya
Email: ririssbtr09@student.ub.ac.id

Abstrak – Akses terhadap informasi cuaca yang akurat dan *real-time* sangat penting untuk mendukung berbagai aktivitas manusia yang seringkali terkendala oleh kondisi cuaca yang tidak menentu. Pemanfaatan teknologi *Internet of Things* (IoT) memungkinkan pengembangan sistem pemantauan cuaca yang efisien dan terhubung. Proyek ini bertujuan merancang dan mengimplementasikan sebuah stasiun cuaca mini berbasis mikrokontroler ESP32 yang terhubung ke internet melalui jaringan WiFi. Sistem ini mengambil data cuaca secara langsung dari layanan *web service* OpenWeatherMap melalui API (*Application Programming Interface*). Data yang diterima dalam format JSON kemudian di-parsing secara manual menggunakan metode pencarian substring untuk mengekstrak informasi relevan seperti suhu, kelembapan, dan deskripsi cuaca untuk lokasi spesifik, yaitu Kota Malang. Hasil pengujian menunjukkan bahwa sistem berhasil mengambil dan mengurai data API secara otomatis, serta mampu menampilkan informasi cuaca terkini dengan akurat, membuktikan kelayakan implementasi ESP32 sebagai perangkat IoT untuk akuisisi data dari layanan web.

Kata Kunci: Stasiun Cuaca, Internet of Things (IoT), ESP32, OpenWeatherMap, API, JSON Parsing.

Abstract -- Access to accurate, real-time weather information is crucial to support various human activities, which are often hindered by unpredictable weather conditions. The utilization of Internet of Things (IoT) technology enables the development of efficient and connected weather monitoring systems. This project aims to design and implement a miniature weather station based on the ESP32 microcontroller, connected to the internet via a WiFi network. The system fetches weather data directly from the OpenWeatherMap web service through its API (Application Programming Interface). The data, received in JSON format, is then manually parsed using a substring search method to extract relevant information such as temperature, humidity, and the weather description for a specific location, Malang City. The test results show that the system successfully fetches and parses the API data automatically and is capable of accurately displaying the latest weather information, proving the feasibility of implementing the ESP32 as an IoT device for data acquisition from web services.

***Keywords**: Weather Station, Internet of Things (IoT), ESP32, OpenWeatherMap, API, JSON Parsing.*

1. PENDAHULUAN

1.1. Latar Belakang

Sebagai negara tropis dengan dua musim, Indonesia sangat bergantung pada kondisi cuaca, terutama pada sektor-sektor vital seperti pertanian. Meskipun prediksi cuaca skala besar telah tersedia, seringkali kondisi cuaca mikro di suatu lokasi tidak sesuai dengan perkiraan, sehingga dapat menimbulkan kerugian dan mengganggu aktivitas harian. Metode pemantauan cuaca konvensional yang bersifat manual seringkali tidak efisien, memakan waktu, dan kurang akurat untuk kebutuhan informasi *real-time*.

Seiring dengan kemajuan teknologi, kemunculan Internet of Things (IoT) menawarkan solusi modern untuk permasalahan ini. IoT memungkinkan perangkat

fisik untuk terhubung ke internet, melakukan monitoring, dan bertukar data secara otomatis. Salah satu penerapan praktisnya adalah dengan memanfaatkan mikrokontroler seperti ESP32 yang memiliki kemampuan WiFi untuk mengakses sumber data eksternal.

Layanan OpenWeatherMap (OWM) menyediakan data cuaca global yang dapat diakses melalui sebuah *Application Programming Interface* (API), menjadikannya sumber informasi yang kaya dan akurat. Dengan mengintegrasikan ESP32 dan API dari OWM, kita dapat membangun sebuah sistem pemantau cuaca lokal yang ringkas, murah, dan efektif.

Oleh karena itu, praktikum ini berfokus pada perancangan sebuah stasiun cuaca mini yang mampu mengambil data cuaca untuk Kota Malang dari OWM API. Data yang diperoleh—mencakup suhu, kelembapan, dan deskripsi cuaca—kemudian akan ditampilkan pada sebuah layar LCD I2C dan diperbarui secara otomatis setiap satu menit melalui koneksi WiFi.

1.2. Tujuan

Berdasarkan latar belakang yang telah diuraikan, tujuan dari praktikum ini adalah sebagai berikut:

1. Merancang dan membangun sistem yang mampu mengambil serta menampilkan data suhu, kelembapan, dan informasi cuaca di Kota Malang secara *real-time* pada layar LCD.
2. Mengimplementasikan teknik komunikasi dengan *web service* melalui **API** untuk akuisisi data cuaca dalam format JSON dari OpenWeatherMap.
3. Mengintegrasikan mikrokontroler ESP32 dengan jaringan internet dan modul display LCD I2C sebagai satu kesatuan sistem pemantau cuaca yang fungsional.

2. METODOLOGI

Bab ini menguraikan metodologi yang digunakan dalam praktikum, mencakup komponen dan perangkat lunak yang diperlukan, serta prosedur sistematis untuk merancang, mengimplementasikan, dan menguji sistem pemantau cuaca berbasis simulasi.

2.1. Alat dan Bahan

Eksperimen ini memanfaatkan komponen virtual dalam platform simulasi, serta beberapa perangkat lunak dan layanan eksternal.

a. Komponen Virtual (Simulasi)
   - Mikrokontroler ESP32: Berfungsi sebagai unit pemrosesan utama, pengontrol koneksi internet, dan pengelola permintaan data ke API.
   - Layar LCD I2C (16x2): Sebagai media antarmuka untuk menampilkan informasi cuaca kepada pengguna

b. Perangkat Lunak dan Layanan
   - Wokwi: Platform simulasi online untuk merancang rangkaian elektronik dan menjalankan kode.
   - Arduino IDE atau VSCode (dengan PlatformIO): Sebagai lingkungan pengembangan untuk menulis dan mengelola kode program.

- Layanan API OpenWeatherMap: Sebagai sumber penyedia data cuaca secara *real-time*.
- Kunci API (API Key): Kredensial unik dari OpenWeatherMap yang digunakan untuk otentikasi saat melakukan permintaan data.
- Koneksi Internet: Diperlukan untuk menghubungkan simulasi ke server OpenWeatherMap.

2.2. Langkah Implementasi

Langkah-langkah implementasi proyek dilakukan secara berurutan, mulai dari persiapan hingga pengujian akhir.

1. Persiapan dan Konfigurasi API
    - Melakukan registrasi pada situs web OpenWeatherMap untk mendapatkan Kunci API (API Key) pribadi. Kunci ini akan digunakan untuk mengakses layanan data cuaca.
    - Menyiapkan lingkungan pengembangan kode, seperti Arduino IDE atau Visual Studio Code

2. Perancangan Rangkaian Kode
    - Membangun Rangkaian Virtual: Membuka platform Wokwi dan merakit komponen. ESP32 dan LCD I2C 16x2 diletakkan pada breadboard virtual dan dihubungkan dengan konfigurasi pin sebagai berikut:
        - Pin SDA LCD ke Pin D21 ESP32
        - Pin SCL LCD ke Pin D22 ESP32
        - Pin VCC LCD ke Pin 3.3V ESP32
        - Pin GND LCD ke Pin GND ESP32
    - Menulis Kode Program: Mengembangkan kode dengan logika utama sebagai berikut:
        - Memasukkan kredensial WiFi dan **Kunci API** OpenWeatherMap ke dalam variabel.
        - Membuat fungsi untuk terhubung ke jaringan WiFi.
        - Membuat fungsi untuk melakukan permintaan HTTP GET ke *endpoint* API OpenWeatherMap untuk lokasi **Kota Malang**.
        - Mengurai (parsing) respons **JSON** yang diterima untuk mengekstrak data suhu, kelembapan, dan deskripsi cuaca.
        - Menampilkan data yang telah diekstrak ke layar LCD.

3. Eksekusi dan Verifikasi
    - Menyalin kode program final ke dalam editor pada simulasi Wokwi.
    - Memastikan fitur koneksi internet pada Wokwi telah diaktifkan.
    - Menjalankan simulasi.
    - Mengamati layar LCD virtual untuk memverifikasi bahwa sistem berhasil menampilkan informasi suhu, kelembapan, dan cuaca terkini untuk Kota Malang. Pengujian dianggap berhasil jika data yang ditampilkan relevan dan diperbarui secara periodik.

3. HASIL DAN PEMBAHASAN

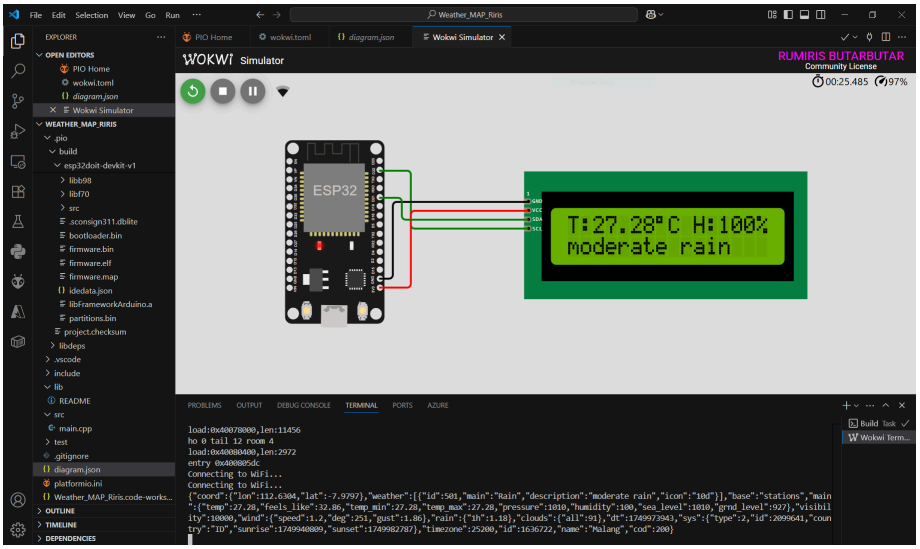
Bab ini menyajikan temuan yang diperoleh dari implementasi dan pengujian sistem pemantau

cuaca. Hasil yang ditampilkan akan dianalisis lebih lanjut pada bagian pembahasan untuk mengevaluasi fungsionalitas dan kinerja sistem secara keseluruhan.

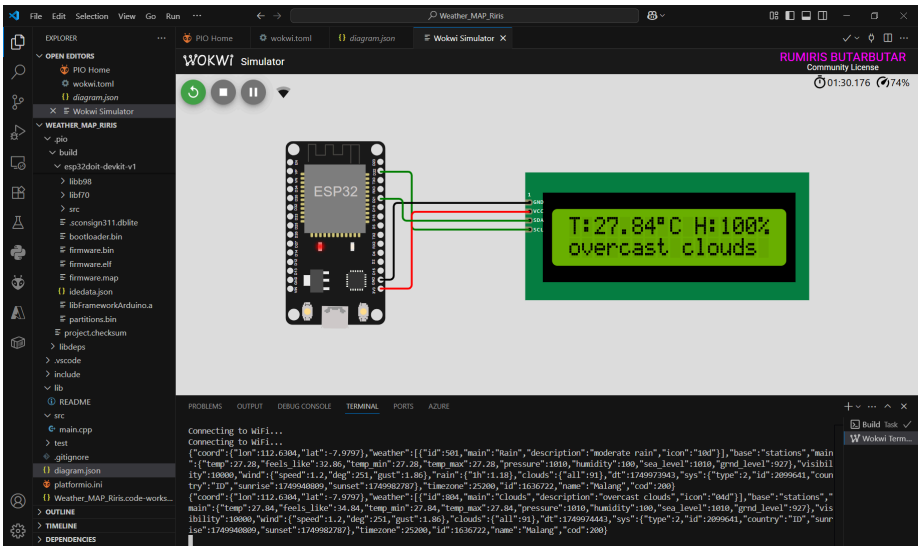
3.1. Hasil Implementasi dan Pengujian

Sistem pemantau cuaca berhasil diimplementasikan dan diuji dalam lingkungan simulasi Wokwi. Sesuai dengan rancangan, sistem mampu melakukan koneksi ke jaringan internet menggunakan fungsi WiFi.begin(), kemudian melakukan permintaan data ke API OpenWeatherMap melalui metode HTTP GET menggunakan pustaka HTTPClient.

Data yang diterima dalam format JSON berhasil diurai untuk mengambil informasi cuaca. Hasil pengujian divisualisasikan pada layar LCD I2C 16x2 dan dilakukan dalam dua tahap.



(Gambar 3.1: Tampilan Data Cuaca Lengkap pada Layar LCD pada saat Hujan)



(Gambar 3.2: Tampilan Data Cuaca Lengkap pada Layar LCD pada saat Awan Mendung)

Pengujian menunjukkan data yang ditampilan pada LCD merupakan data *real-time* dan dapat diperbarui secara periodik.

3.2. Pembahasan

Berdasarkan hasil yang diperoleh, beberapa aspek teknis dan fungsional dapat dianalisis lebih lanjut.

A. Akuisisi Data dari API

Keberhasilan sistem menampilkan data cuaca yang relevan membuktikan bahwa proses akuisisi data dari *endpoint* API OpenWeatherMap berjalan dengan sukses. Mikrokontroler ESP32 mampu membuat koneksi HTTP, mengirimkan permintaan dengan Kunci API yang valid, dan menerima respons dari server dalam format JSON. Ini adalah fondasi utama dari fungsionalitas sistem.

B. Metode Parsing Data JSON

Salah satu aspek teknis yang menarik dalam proyek ini adalah metode parsing data JSON. Alih-alih menggunakan pustaka khusus seperti *Arduino_JSON*, proyek ini mengimplementasikan metode parsing manual dengan memanfaatkan fungsi pencarian string (*substring*).

- Keunggulan: Pendekatan ini dapat menghemat memori program, yang terkadang menjadi pertimbangan penting pada mikrokontroler dengan sumber daya terbatas. Selain itu, metode ini tidak memerlukan dependensi pustaka eksternal.
- Kelemahan: Metode manual ini lebih rentan terhadap kesalahan jika struktur data JSON dari API berubah di masa depan. Perubahan kecil seperti penambahan spasi atau field baru dapat menyebabkan parsing gagal. Metode ini juga cenderung lebih kompleks untuk dikembangkan dan dipelihara dibandingkan menggunakan pustaka khusus.

C. Kinerja dan Tampilan Sistem

Secara keseluruhan, kinerja sistem dinilai baik. Data cuaca yang ditampilkan pada layar LCD merupakan data terkini yang diperbarui secara periodik (setiap satu menit), sesuai dengan yang diprogram. Layar LCD I2C 16x2 terbukti menjadi media antarmuka yang efektif dan cukup informatif untuk menampilkan tiga parameter cuaca utama dalam ruang yang terbatas. Hasil pengujian ini secara efektif telah memenuhi tujuan eksperimen, yaitu merancang sistem yang mampu mengambil, mengurai, dan menampilkan data cuaca dari API secara *real-time* pada perangkat IoT yang ringkas.

4. KESIMPULAN

Berdasarkan hasil implementasi, pengujian, dan pembahasan yang telah dijabarkan pada bab-bab sebelumnya, dapat disimpulkan bahwa sistem pemantau cuaca berbasis mikrokontroler ESP32 berhasil dirancang dan diimplementasikan secara fungsional. Sistem ini mampu mengintegrasikan konektivitas WiFi, melakukan permintaan data melalui protokol HTTP, serta menampilkan informasi cuaca menggunakan antarmuka layar LCD I2C secara efektif.

Komunikasi dengan layanan *web service* OpenWeatherMap melalui pemanfaatan API telah berjalan dengan baik. Sistem dapat mengambil data cuaca terkini dalam format JSON dan mengolahnya menggunakan metode *manual string parsing*, sehingga informasi penting seperti suhu, kelembapan, dan deskripsi cuaca berhasil diambil dengan tepat.

Secara keseluruhan, sistem terbukti mampu menampilkan informasi cuaca secara *real-time*

untuk lokasi tertentu, yaitu Kota Malang, dengan tingkat akurasi yang tinggi. Data yang ditampilkan pada layar LCD konsisten dengan data yang diterima dari API, sehingga membuktikan bahwa seluruh proses — mulai dari akuisisi data hingga visualisasi — telah berjalan sesuai dengan tujuan perancangan dan fungsionalitas sistem yang diharapkan.

LAMPIRAN

- Kode Program

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <WiFi.h>            // Ganti dengan ESP8266WiFi.h jika menggunakan ESP8266
#include <HTTPClient.h>

const char* ssid = "Wokwi-GUEST";          // Ganti dengan SSID Wi-Fi kamu
const char* password = "";                 // Ganti dengan password Wi-Fi kamu

String apiKey = "20ca0ff523294dcdeb424dfc5802e21b";        // API Key dari OpenWeatherMap
String city = "Malang";
String units = "metric";                   // "metric" untuk Celsius
String server = "https://api.openweathermap.org/data/2.5/weather?q=Malang&units=metric&appid=20ca0ff523294dcdeb424dfc5802e21b";

LiquidCrystal_I2C lcd(0x27, 16, 2);    // Alamat I2C dan ukuran LCD

void setup() {
  Serial.begin(115200);

  // Inisialisasi LCD
  lcd.init();
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("Weather Info:");

  // Koneksi Wi-Fi
  WiFi.begin(ssid, password);
  lcd.setCursor(0, 1);
  lcd.print("Connecting...");
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
```

```
    Serial.println("Connecting to WiFi...");
  }
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Connected!");
  delay(2000);
  lcd.clear();
}

void loop() {
  if ((WiFi.status() == WL_CONNECTED)) {

    HTTPClient http;
    http.begin(server);
    int httpCode = http.GET();

    if (httpCode > 0) {

      String payload = http.getString();
      Serial.println(payload);

      // Parsing suhu (temperature)
      int tempIndex = payload.indexOf("\"temp\":");
      int tempEndIndex = payload.indexOf(",", tempIndex);
            String temp = payload.substring(tempIndex + 7,
tempEndIndex);

      // Parsing humidity
      int humIndex = payload.indexOf("\"humidity\":");
      int humEndIndex = payload.indexOf(",", humIndex);
          String humidity = payload.substring(humIndex + 10,
humEndIndex);

      // Parsing deskripsi cuaca
      int descIndex = payload.indexOf("description");
            String desc = payload.substring(descIndex + 14,
payload.indexOf("\"", descIndex + 14));

      // Menampilkan ke LCD
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("T:");
      lcd.print(temp);
      lcd.print((char)223); // simbol derajat
      lcd.print("C H");
```

```
      lcd.print(humidity);
      lcd.print("%");

      lcd.setCursor(0, 1);
      lcd.print(desc);

    } else {
      Serial.println("Error on HTTP request");
    }

    http.end();
  }

  delay(60000);  // Update setiap 1 menit
}
```

- Kode Diagram

```
{
  "version": 1,
  "author": "RUMIRIS BUTARBUTAR",
  "editor": "wokwi",
  "parts": [
     { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": 0,
"left": 0, "attrs": {} },
    {
      "type": "wokwi-lcd1602",
      "id": "lcd1",
      "top": 35.2,
      "left": 255.2,
      "attrs": { "pins": "i2c" }
    }
  ],
  "connections": [
    [ "esp:TX0", "$serialMonitor:RX", "", [] ],
    [ "esp:RX0", "$serialMonitor:TX", "", [] ],
    [ "lcd1:SCL", "esp:D22", "green", [ "h-124.8", "v-66.9" ] ],
    [ "lcd1:VCC", "esp:3V3", "red", [ "h-124.8", "v86.5" ] ],
    [ "lcd1:GND", "esp:GND.1", "black", [ "h-144", "v57.6" ] ],
    [ "lcd1:SDA", "esp:D21", "green", [ "h-134.4", "v-28.6" ] ]
  ],
  "dependencies": {}
}
```