Monitoring Cuaca Menggunakan Weather API dengan Wokwi Arduino di Visual Studio Code

Fransiska Natasya Desyanti

Program Studi Teknologi Informasi, Fakultas Vokasi Universitas Brawijaya

francenats@gmail.com

Abstract

This practical project aims to design and implement a weather monitoring system based on the Internet of Things (IoT) using ESP32, which retrieves weather data from a Weather API and displays it on a virtual 16x2 LCD using the Wokwi Simulator and PlatformIO in Visual Studio Code. The system replaces the need for physical sensors by utilizing cloud-based data from OpenWeatherMap, offering a more flexible and cost-efficient alternative. Simulation results show that the ESP32 successfully connects to virtual WiFi and displays real-time weather data, including temperature and weather conditions. By leveraging APIs, JSON parsing, and virtual tools, this system serves as an effective prototype for developing cloud-based environmental monitoring solutions without physical hardware.

Keywords*: Internet of Things, Weather API, ESP32, Wokwi, Visual Studio Code, LCD, OpenWeatherMap, PlatformIO..*

Abstrak

Praktikum ini bertujuan untuk merancang dan mengimplementasikan sistem monitoring cuaca berbasis Internet of Things (IoT) menggunakan ESP32 yang mengambil data dari Weather API dan divisualisasikan melalui LCD 16x2 secara virtual menggunakan Wokwi Simulator dan PlatformIO di Visual Studio Code. Sistem ini menggantikan kebutuhan akan sensor fisik dengan memanfaatkan data cloud dari OpenWeatherMap, sehingga lebih fleksibel dan efisien. Hasil simulasi menunjukkan bahwa ESP32 berhasil terhubung ke jaringan WiFi virtual dan mampu mengambil serta menampilkan data suhu dan kondisi cuaca secara real-time. Dengan pemanfaatan API, parsing JSON, serta antarmuka virtual, sistem ini dapat menjadi solusi awal untuk pengembangan perangkat monitoring lingkungan berbasis data daring tanpa perangkat keras tambahan.

Kata Kunci: Internet of Things, Weather API, ESP32, Wokwi, Visual Studio Code, LCD, OpenWeatherMap, PlatformIO.

1. PENDAHULUAN

1.1. LATAR BELAKANG

Dalam dunia yang semakin terhubung, kebutuhan akan informasi cuaca yang akurat dan up-to-date sangat penting di berbagai bidang seperti pertanian, transportasi, dan aktivitas harian. Namun, penggunaan sensor fisik terkadang terbatas oleh biaya, ketersediaan, dan kondisi lingkungan. Oleh karena itu, memanfaatkan *Weather API* sebagai sumber data cuaca merupakan solusi alternatif yang efisien dan fleksibel. Dengan bantuan platform simulasi seperti Wokwi dan editor seperti Visual Studio Code, pengembangan sistem berbasis Arduino menjadi lebih praktis dan mudah diakses. Sistem ini tidak hanya memungkinkan pengambilan data cuaca secara real-time, tetapi juga dapat dikembangkan lebih lanjut untuk pengambilan keputusan otomatis berbasis kondisi lingkungan.Melalui praktikum ini, diharapkan dapat dihasilkan prototipe sistem lampu lalu lintas berbasis IoT yang dapat dikontrol secara otomatis dan dimonitor melalui jaringan. Implementasi ini tidak hanya memberikan pemahaman tentang IoT dan pemrograman ESP32, tetapi juga menunjukkan potensi pemanfaatan teknologi dalam kehidupan sehari-hari.

1.2. TUJUAN PRAKTIKUM

Praktikum ini bertujuan untuk merancang dan mengimplementasikan sistem monitoring cuaca yang mengambil data dari Weather API dan menampilkannya melalui mikrokontroler Arduino secara virtual menggunakan Wokwi dan Visual Studio Code. Melalui praktikum ini, peserta diharapkan memahami proses pengambilan data melalui API, pengolahan data di Arduino, serta visualisasi hasil secara sederhana. Selain itu, praktikum ini juga bertujuan untuk melatih keterampilan integrasi perangkat lunak dan perangkat keras virtual dalam pengembangan sistem IoT berbasis data cloud.

2. METODOLOGI

2.1. ALAT DAN BAHAN

Dalam pelaksanaan praktikum ini, beberapa alat dan bahan digunakan untuk mengimplementasikan Monitoring Cuaca Menggunakan Weather API dengan Wokwi Arduino di Visual Studio Code berbasis Internet of Things menggunakan ESP32. Alat dan bahan tersebut dibagi menjadi dua kategori utama, yaitu perangkat lunak (software) dan komponen virtual yang digunakan dalam simulasi Wokwi. Berikut adalah penjelasan detailnya:

a. Wokwi Simulator
   Platform simulasi online yang memungkinkan pengguna untuk merancang dan menguji sirkuit elektronik secara virtual. Wokwi digunakan untuk membuat sirkuit lampu lalu lintas dengan ESP32 sebagai mikrokontroler utama.
b. Visual Studio Code (VSCode)
   Editor kode yang digunakan untuk menulis, mengelola, dan mengembangkan program. VSCode dipilih karena fleksibilitas dan dukungannya yang luas terhadap berbagai bahasa pemrograman.
c. PlatformIO
   Ekstensi di VSCode yang menyediakan lingkungan pengembangan terintegrasi untuk pemrograman mikrokontroler, termasuk ESP32. PlatformIO memudahkan proses kompilasi, upload, dan debugging kode.
d. Library ESP32
   Beberapa library seperti WiFi.h digunakan untuk mengimplementasikan fitur IoT, seperti konektivitas Wi-Fi dan kontrol jarak jauh.
e. ESP32
   Mikrokontroler virtual yang berfungsi sebagai otak dari sistem lampu lalu lintas. ESP32 dipilih karena kemampuannya dalam menghubungkan perangkat ke jaringan IoT.
f. LCD 16x2
   Komponen tampilan virtual yang digunakan untuk menampilkan teks atau data dari mikrokontroler secara real-time dalam simulasi Wokwi.
g. Kabel Virtual
   Digunakan untuk menghubungkan ESP32 dengan LED dan resistor dalam sirkuit virtual.
h. Koneksi Internet
   Diperlukan untuk mengakses Wokwi Simulator dan menguji fitur IoT seperti kontrol jarak jauh.

2.2. LANGKAH IMPLEMENTASI

Implementasi Monitoring Cuaca Menggunakan Weather API dengan Wokwi Arduino di Visual Studio Code berbasis Internet of Things menggunakan ESP32 pada Wokwi Simulator dengan PlatformIO di Visual Studio Code (VSCode) dilakukan melalui beberapa tahapan. Adapun tahapan yang akan dilakukan dalam praktikum ini adalah sebagai berikut:

A. Persiapan Lingkungan Pengembangan

1. Install Visual Studio Code dan ekstensi PlatformIO.
2. Buat project baru dengan board ESP32 Dev Module di PlatformIO.
3. Siapkan koneksi internet dan akses ke Wokwi untuk kebutuhan simulasi.
4. Tambahkan library HTTPClient dan WiFi untuk mengakses API cuaca.

B. Konfigurasi Weather API

1. Daftar akun di layanan *Weather API* seperti OpenWeatherMap**.**
2. Dapatkan API key dan endpoint URL yang berisi data cuaca berdasarkan lokasi (misal berdasarkan kota).
3. Uji endpoint API menggunakan browser atau Postman untuk memastikan koneksi berhasil.

C. Menulis dan Mengembangkan Kode Program

1. Tulis kode program di PlatformIO dengan logika:
2. Menghubungkan ESP32 ke WiFi.
3. Mengirim permintaan HTTP GET ke Weather API.
4. Mengambil dan mengurai data suhu, kelembapan, dan cuaca.
5. Tambahkan logika untuk menampilkan data ke serial monitor atau LCD Wokwi.
6. Gunakan library seperti ArduinoJson untuk parsing JSON dari API.

3. HASIL DAN PEMBAHASAN

3.1. HASIL EKSPERIMEN

Table 1. Hasil Implementasi Sistem Monitoring Cuaca Menggunakan Weather API

Aspek	Hasil	Keterangan
Simulasi di Wokwi	Data cuaca berhasil ditampilkan di Serial Monitor dan LCD.	Wokwi efektif sebagai simulasi tanpa perangkat fisik untuk menguji respon data dari API cuaca.
Pengujian Kode di PlatformIO	Kode berhasil dikompilasi dan dijalankan tanpa error.	PlatformIO di VSCode mempermudah proses pemrograman dengan fitur auto-build dan debugging.
Koneksi WiFi dan API	ESP32 berhasil terhubung ke WiFi dan mengakses Weather API.	Modul ESP32 mampu melakukan permintaan HTTP GET dan menerima respon JSON dari server API.
Parsing Data JSON	Data suhu dan kondisi cuaca berhasil diambil dan ditampilkan.	Library ArduinoJson berhasil mengurai struktur JSON dengan akurat sesuai format dari API.

Dari hasil praktikum, sistem monitoring cuaca berhasil dijalankan menggunakan ESP32 di Wokwi dan dikembangkan melalui PlatformIO. Data cuaca dari Weather API dapat ditampilkan dengan baik di Serial Monitor dan LCD. Kode berjalan tanpa error, koneksi WiFi virtual berhasil, dan data JSON berhasil diproses. Sistem juga mampu memperbarui data secara berkala, menunjukkan bahwa monitoring cuaca dapat dilakukan secara efisien tanpa sensor fisik.

4. LAMPIRAN

4.1. KODE PROGRAM

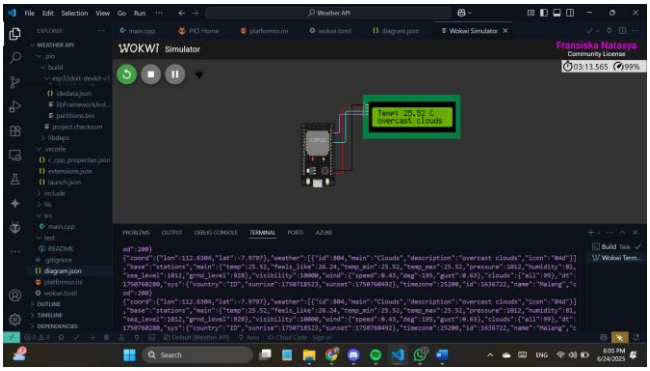
```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <WiFi.h>          // Ganti dengan ESP8266WiFi.h jika menggunakan ESP8266
#include <HTTPClient.h>
const char* ssid = "Wokwi-GUEST";       // Ganti dengan SSID Wi-Fi kamu
const char* password = ""; // Ganti dengan password Wi-Fi kamu
String apiKey = "20ca0ff523294dcdeb424dfc5802e21b"; // API Key dari OpenWeatherMap
String city = "Malang";              // Kota yang ingin ditampilkan
String units = "metric";             // Untuk Celsius gunakan "metric", untuk Fahrenheit "imperial"
String server = "http://api.openweathermap.org/data/2.5/weather?q=" + city + "&units=" + units + "&appid=" + apiKey;
//String server= "https://api.openweathermap.org/data/2.5/weather?q=Malang&appid=20ca0ff523294dcdeb424dfc5802e21b";

LiquidCrystal_I2C lcd(0x27, 16, 2);   // Inisialisasi LCD dengan alamat I2C 0x27

void setup() {
Serial.begin(115200);
// Inisialisasi LCD
lcd.init();
lcd.backlight();
lcd.setCursor(0, 0);
lcd.print("Weather Info:");
// Inisialisasi 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))
{  // Check WiFi connection status
HTTPClient http;
http.begin(server);     // Specify the
URL
int httpCode = http.GET();  // Make
the request

if (httpCode > 0) { // Check for the
returning code

String payload = http.getString();
Serial.println(payload);    // Print
the response payload

// Parse data (extract temperature)
int tempIndex =
payload.indexOf("temp");
String temp =
payload.substring(tempIndex + 6,
payload.indexOf(",", tempIndex));

// Display temperature on LCD
lcd.setCursor(0, 0);
lcd.print("Temp:");
lcd.setCursor(6, 0);
lcd.print(temp);
lcd.print(" C");

// Extract weather description
int descIndex =
payload.indexOf("description");
String desc =
payload.substring(descIndex + 14,
payload.indexOf("\"", descIndex +
14));

// Display description on LCD
lcd.setCursor(0, 1);
lcd.print(desc);

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

http.end();  // Free the resources
}
delay(60000);  // Update every minute
}
```

4.2. KODE DIAGRAM

```
{
  "version": 1,
  "author": "Fransiska Natasya",
  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-esp32-devkit-
v1", "id": "esp", "top": 4.7, "left":
4.6, "attrs": {} },
    {
      "type": "wokwi-lcd1602",
      "id": "lcd1",
      "top": -80,
      "left": 197.6,
      "attrs": { "pins": "i2c" }
    }
  ],
  "connections": [
    [ "esp:TX0", "$serialMonitor:RX",
"", [] ],
    [ "esp:RX0", "$serialMonitor:TX",
"", [] ],
    [ "lcd1:SCL", "esp:D22",
"violet", [ "h-76.8", "v67.8" ] ],
    [ "lcd1:VCC", "esp:3V3", "red", [
"h-67.2", "v206.6" ] ],
    [ "lcd1:GND", "esp:GND.1",
"black", [ "h-38.4", "v211.3" ] ],
    [ "lcd1:SDA", "esp:D21", "cyan",
[ "h-57.6", "v101.5" ] ]
  ],
  "dependencies": {}
}
```

4.3. HASIL SIMULASI



Gambar 1. Tampilan Hasil Monitoring Cuaca