**LAPORAN PRAKTIKUM**

**IOT MINGGU KE-5**

**Memantau Cuaca Terkini Dari Open Weather Map API**

**Melalui Koneksi Wifi Bebasis Esp32**

****

**Dosen Pengampu :**

**Ir. Subairi, ST., MT., IPM**

**Disusun Oleh:**

**Muhammad Alif Aris**

**(233140707111077)**

**PROGRAM STUDI TEKNOLOGI INFORMASI**

**FAKULTAS VOKASI**

**UNIVERSITAS BRAWIJAYA**

**2025**

**Abstrak**

Proyek ini mengembangkan sistem pemantauan cuaca yang menggunakan API ESP32 dan OpenWeatherMap. Dengan menggunakan teknologi Internet of Things (IoT), sistem ini dirancang untuk membaca data cuaca secara real-time melalui koneksi Wi-Fi dan menampilkan informasi seperti suhu, kelembapan, dan deskripsi cuaca pada layar LCD. Proses pengembangan dilakukan secara virtual menggunakan platform Wokwi untuk menyusun dan menguji rangkaian dan kode program. Hasil eksperimen menunjukkan bahwa sistem dapat mengambil data dari API server bahkan dalam kondisi jaringan yang tidak stabil. Sistem ini menjadi prototipe pemantauan cuaca yang efektif dan fleksibel dengan fitur tambahan seperti tampilan gulir dan tombol navigasi.

*Keywords - Internet of Things, ESP32, OpenWeatherMap API, koneksi Wi-Fi, tampilan real-time, layar LCD, simulasi Wokwi, data cuaca, sistem otomatis, navigasi tombol.*

**BAB I**

**PENDAHULUAN**

**1.1 Latar Belakang**

Di era modern, pemantauan cuaca secara real-time sangat penting untuk kebutuhan individu, bisnis, dan pengambilan keputusan dalam kegiatan tertentu. Dengan kemajuan teknologi saat ini, data cuaca dapat diakses melalui layanan berbasis API seperti Open Weather Map. Mikrokontroler ESP32 dengan konektivitas Wi-Fi memungkinkan sistem untuk mengambil dan menampilkan data cuaca dari API Open Weather Map, yang mempermudah integrasi perangkat keras dengan layanan cloud, sehingga perangkat pintar dapat dengan mudah mengakses informasi cuaca. Solusi untuk berbagai kebutuhan, seperti pengelolaan energi, perencanaan aktivitas di luar ruangan, dan aplikasi pertanian cerdas, disediakan oleh sistem ini. Dengan inovasi ini, pengguna belajar tentang pengelolaan data dan aplikasi IoT.

**1.2 Tujuan Eksperimen**

1. Mengembangkan sistem pemantauan cuaca berbasis ESP32 yang dapat menggunakan Open Weather Map API untuk mengumpulkan data saat ini.
2. Memahami cara mengintegrasikan layanan API dengan mikrokontroler untuk menampilkan data cuaca seperti suhu, kelembapan, dan kondisi atmosfer lainnya.
3. Mengkaji penggunaan teknologi Internet of Things untuk membuat model sistem yang dapat digunakan untuk berbagai aplikasi, seperti pengelolaan lingkungan, pertanian cerdas, dan perencanaan kegiatan berbasis cuaca.

**BAB II**

**METODOLOGI**

**2.1 Alat dan Bahan**

* ESP32 DevKit v1
* Resistor (1kΩ dan 10kΩ)
* LED (opsional)
* Kabel Penghubung Virtual
* Platform Simulasi Wokwi
* Komputer/Laptop dengan Internet
* OpenWeatherMap API Key
* Sensor Virtual

**2.3 Implementasi Sistem**

1. Hubungkan ESP32 ke breadboard virtual di Wokwi untuk menyusun rangkaian simulasi.
2. Integrasikan API OpenWeatherMap dengan memasukkan API Key dan URL ke dalam program.
3. Atur koneksi Wi-Fi ESP32 dengan menambahkan SSID dan password pada kode.
4. Jalankan simulasi untuk menampilkan data cuaca real-time di Serial Monitor.

**BAB III**

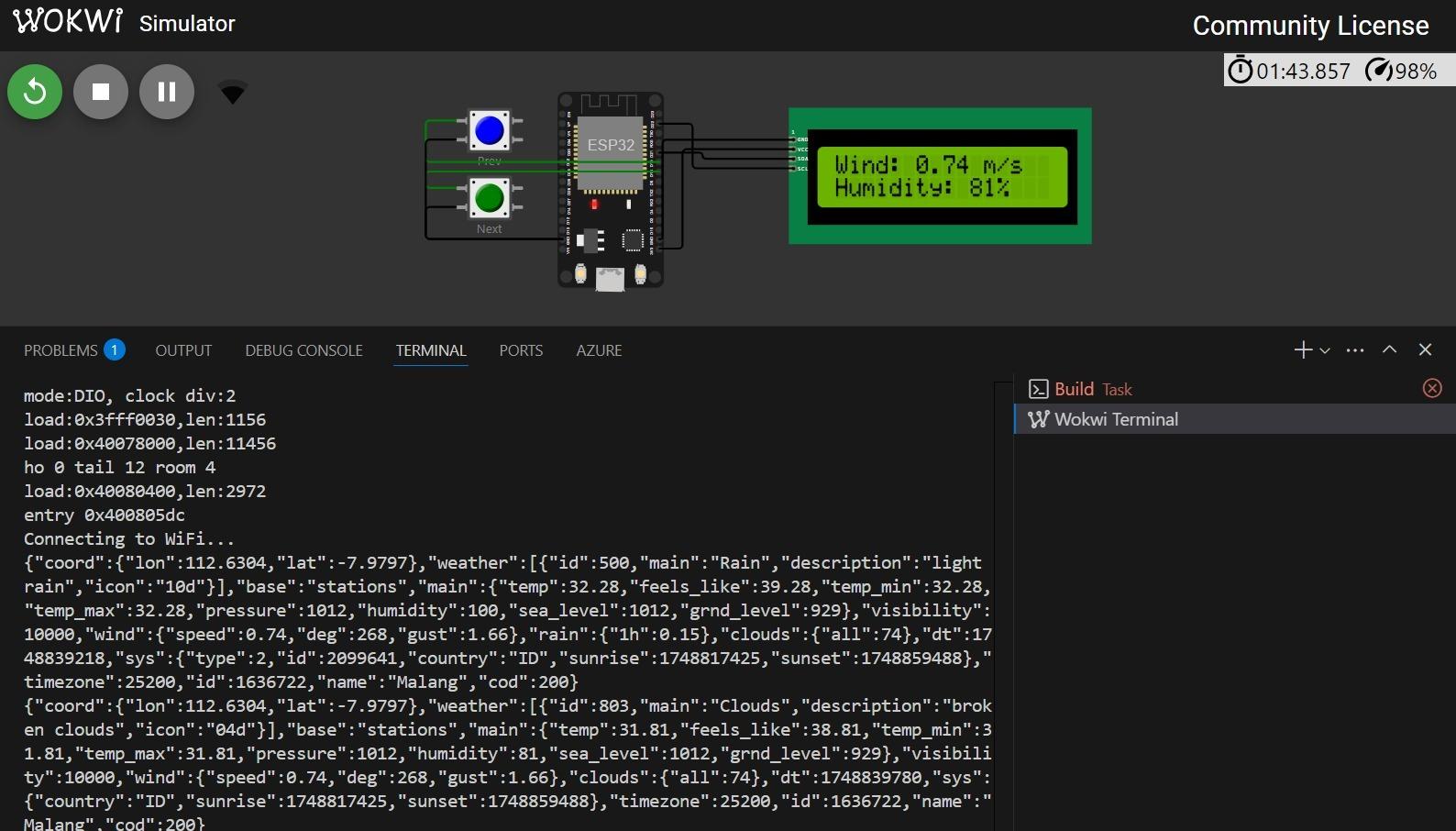
**HASIL DAN PEMBAHASAN**

**3.1 Hasil Eksperimen**

Eksperimen ini menunjukkan bahwa ESP32 dapat mengakses data cuaca secara real-time dari API OpenWeatherMap melalui koneksi Wi-Fi. Suhu, kelembapan, dan kondisi cuaca ditampilkan dengan jelas di Serial Monitor, menunjukkan bahwa integrasi antara perangkat lunak dan perangkat keras berhasil. Sistem berjalan seperti yang direncanakan, dengan permintaan data API dikirim pada interval tertentu dan dijawab dengan benar oleh server. Simulasi di Wokwi mempermudah pengujian tanpa mengorbankan perangkat.

Sistem ini menanggapi perubahan data cuaca di server API dengan cepat. Jika jaringan Wi-Fi tidak stabil, program dirancang untuk mencoba kembali koneksi hingga data diperbarui dengan sukses. Eksperimen ini menunjukkan pemahaman yang baik tentang pemrograman Internet of Things dasar dan menunjukkan kekuatan ESP32 untuk aplikasi yang lebih kompleks.

**3.2 Dokumentasi eksperimen meliputi screenshoot simulasi :**

****

**Lampiran**

**Kode Program 1 :**

{

"version": 1,

"author": "Cagita Dian A'yunin",

"editor": "wokwi",

"parts": [

{ "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": -50, "left": -100, "attrs": {} },

{

"type": "wokwi-lcd1602",

"id": "lcd",

"top": -32,

"left": 130.4,

"attrs": { "pins": "i2c" }

},

{

"type": "wokwi-pushbutton",

"id": "btnNext",

"top": 35,

"left": -201.6,

"attrs": { "color": "green", "label": "Next" }

},

{

"type": "wokwi-pushbutton",

"id": "btnPrev",

"top": -32.2,

"left": -201.6,

"attrs": { "color": "blue", "label": "Prev" }

}

],

"connections": [

[ "esp:TX0", "$serialMonitor:RX", "", [] ],

[ "esp:RX0", "$serialMonitor:TX", "", [] ],

[ "lcd:SCL", "esp:D22", "black", [ "h-100", "v-40" ] ],

[ "lcd:SDA", "esp:D21", "black", [ "h-90", "v-20" ] ],

[ "lcd:VCC", "esp:3V3", "black", [ "h-110", "v60" ] ],

[ "lcd:GND", "esp:GND.1", "black", [ "h-130", "v40" ] ],

[ "btnNext:1.l", "esp:D18", "green", [ "h-30", "v-20" ] ],

[ "btnNext:2.l", "esp:GND.2", "black", [ "h-30", "v20" ] ],

[ "btnPrev:1.l", "esp:D19", "green", [ "h-30", "v-20" ] ],

[ "btnPrev:2.l", "esp:GND.2", "black", [ "h-30", "v20" ] ]

],

"dependencies": {}

}

**Kode Program 2 :**

#include <Arduino.h>

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

#include <WiFi.h>

#include <HTTPClient.h>

#include <ArduinoJson.h>

const char\* ssid = "Wokwi-GUEST";

const char\* password = "";

String apiKey = "aa183d09b9792e4ef1d50d33a2222699";

String city = "Malang City, East Java";

String units = "metric";

String server = "http://api.openweathermap.org/data/2.5/weather?q=Malang&units=metric&appid=aa183d09b9792e4ef1d50d33a2222699";

LiquidCrystal\_I2C lcd(0x27, 16, 2);

String displayText = "";

int scrollIndex = 0;

unsigned long lastUpdateTime = 0;

unsigned long lastScrollTime = 0;

const long updateInterval = 60000;

const long scrollInterval = 300;

const int buttonNextPin = 18;

const int buttonPrevPin = 19;

int currentPage = 0;

const int totalPages = 4;

unsigned long lastDebounceTime = 0;

const long debounceDelay = 200;

String temp = "";

String desc = "";

String humidity = "";

String wind = "";

void updateWeather();

void showPage(int page);

void scrollDisplay();

void setup() {

Serial.begin(115200);

lcd.init();

lcd.backlight();

lcd.setCursor(0, 0);

lcd.print("Weather Info:");

pinMode(buttonNextPin, INPUT\_PULLUP);

pinMode(buttonPrevPin, INPUT\_PULLUP);

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();

updateWeather();

showPage(currentPage);

}

void loop() {

unsigned long currentMillis = millis();

if (currentMillis - lastUpdateTime >= updateInterval) {

updateWeather();

lastUpdateTime = currentMillis;

showPage(currentPage);

}

if (currentPage == 2) {

if (currentMillis - lastScrollTime >= scrollInterval) {

scrollDisplay();

lastScrollTime = currentMillis;

}

}

if (digitalRead(buttonNextPin) == LOW) {

if (millis() - lastDebounceTime > debounceDelay) {

currentPage++;

if (currentPage >= totalPages) currentPage = 0;

showPage(currentPage);

lastDebounceTime = millis();

}

}

if (digitalRead(buttonPrevPin) == LOW) {

if (millis() - lastDebounceTime > debounceDelay) {

if (currentPage == 0) currentPage = totalPages - 1;

else currentPage--;

showPage(currentPage);

lastDebounceTime = millis();

}

}

}

void updateWeather() {

if (WiFi.status() == WL\_CONNECTED) {

HTTPClient http;

http.begin(server);

int httpCode = http.GET();

if (httpCode > 0) {

String payload = http.getString();

Serial.println(payload);

StaticJsonDocument<1024> doc;

deserializeJson(doc, payload);

temp = String(doc["main"]["temp"].as<float>());

desc = doc["weather"][0]["description"].as<String>();

humidity = String(doc["main"]["humidity"].as<int>());

wind = String(doc["wind"]["speed"].as<float>());

displayText = "| Temp: " + temp + " C | " + desc;

scrollIndex = 0;

} else {

Serial.println("Error on HTTP request");

}

http.end();

}

}

void showPage(int page) {

lcd.clear();

switch (page) {

case 0:

lcd.setCursor(0, 0);

lcd.print("Location:");

lcd.setCursor(0, 1);

lcd.print(city);

break;

case 1:

lcd.setCursor(0, 0);

lcd.print("Temperature:");

lcd.setCursor(0, 1);

lcd.print(temp + " C");

break;

case 2:

lcd.setCursor(0, 0);

lcd.print("Weather:");

lcd.setCursor(0, 1);

if (desc.length() > 16) {

lcd.print(desc.substring(0, 16));

} else {

lcd.print(desc);

}

break;

case 3:

lcd.setCursor(0, 0);

lcd.print("Wind: " + wind + " m/s");

lcd.setCursor(0, 1);

lcd.print("Humidity: " + humidity + "%");

break;

}

}

void scrollDisplay() {

if (displayText.length() > 0) {

lcd.setCursor(0, 1);

if (scrollIndex + 16 <= displayText.length()) {

lcd.print(displayText.substring(scrollIndex, scrollIndex + 16));

} else {

String part1 = displayText.substring(scrollIndex);

String part2 = displayText.substring(0, 16 - part1.length());

lcd.print(part1 + part2);

}

scrollIndex++;

if (scrollIndex >= displayText.length()) {

scrollIndex = 0;

}

}

}