**SIMULASI SISTEM INFORMASI CUACA REAL-TIME BERBASIS API OPENWEATHER MENGGUNAKAN ESP32 DAN LCD I2C**

*Azzam Beryl Nemesio Wijoyo  
Fakultas Vokasi, Universitas Brawijaya*[*nemesioberyl@gmail.com*](mailto:nemesioberyl@gmail.com)

**ABSTRAK**

Praktikum ini bertujuan untuk membangun sistem monitoring cuaca secara real-time menggunakan mikrokontroler ESP32 yang terhubung dengan layanan API OpenWeather. Data yang diperoleh mencakup berbagai parameter cuaca seperti suhu, kelembapan, kecepatan dan arah angin, serta waktu terbit dan terbenam matahari. Informasi tersebut ditampilkan pada layar LCD 16x2 dengan komunikasi I2C, serta dapat dinavigasi menggunakan tombol fisik Next dan Prev. Proyek ini disimulasikan secara penuh pada platform Wokwi, memungkinkan pengujian sistem tanpa perangkat keras fisik. Hasil simulasi menunjukkan sistem dapat berjalan dengan baik dan responsif.

**Kata Kunci:** IoT, ESP32, LCD I2C, API Cuaca, Wokwi

**1. PENDAHULUAN**

1.1 Latar Belakang

Internet of Things (IoT) memberikan peluang besar dalam pengembangan sistem monitoring lingkungan yang terhubung internet. Salah satu implementasinya adalah sistem pemantauan cuaca menggunakan data dari sumber terbuka seperti OpenWeather. Dengan menggabungkan mikrokontroler ESP32, LCD I2C, dan tombol navigasi, sistem ini dapat menampilkan informasi cuaca secara ringkas dan dinamis.

1.2 Tujuan Praktikum

* Mengkaji penggunaan API OpenWeather untuk memperoleh data cuaca.
* Mengembangkan tampilan cuaca berbasis LCD I2C.
* Menyimulasikan pengambilan dan penampilan data secara berurutan.
* Menerapkan logika scrolling data menggunakan tombol fisik.
* Menggunakan platform simulasi Wokwi dan VS Code.

**2. METODOLOGI**

2.1 Alat dan Bahan

**Perangkat Lunak:**

* Wokwi Simulator
* Visual Studio Code + PlatformIO

**Perangkat Keras (virtual):**

* ESP32 DevKit V1
* LCD 16x2 I2C
* 2 Pushbutton (Next dan Prev)
* Wi-Fi Wokwi

2.2 Langkah Implementasi

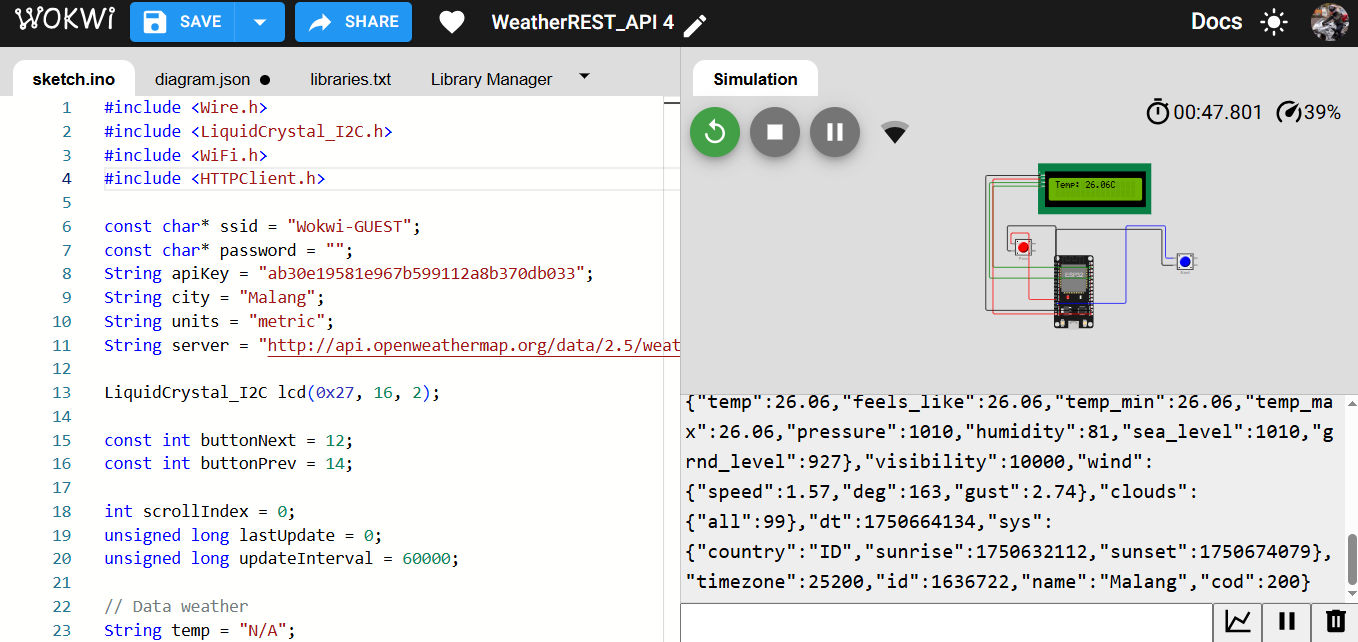
1. **Pengaturan Wokwi Web:**
   * Tambahkan ESP32, LCD I2C, dan 2 tombol.
   * Hubungkan SDA ke D21, SCL ke D22, VCC ke 3V3, dan GND ke GND.
   * Tombol Next ke D12, Prev ke D14.
2. **Pemrograman:**
   * Menggunakan HTTPClient untuk mengambil data JSON dari OpenWeather.
   * Parsing data secara manual tanpa library tambahan.
   * Menampilkan data sesuai index scroll.
3. **Simulasi:**
   * Uji tombol navigasi untuk scroll antar data.
   * Verifikasi penampilan data cuaca secara berurutan.
4. **Integrasi dengan VS Code:**
   * Buat project di PlatformIO.
   * Masukkan file diagram.json dan wokwi.toml.
   * Simulasikan melalui ikon Start Simulation di VS Code.

**3. HASIL DAN PEMBAHASAN**

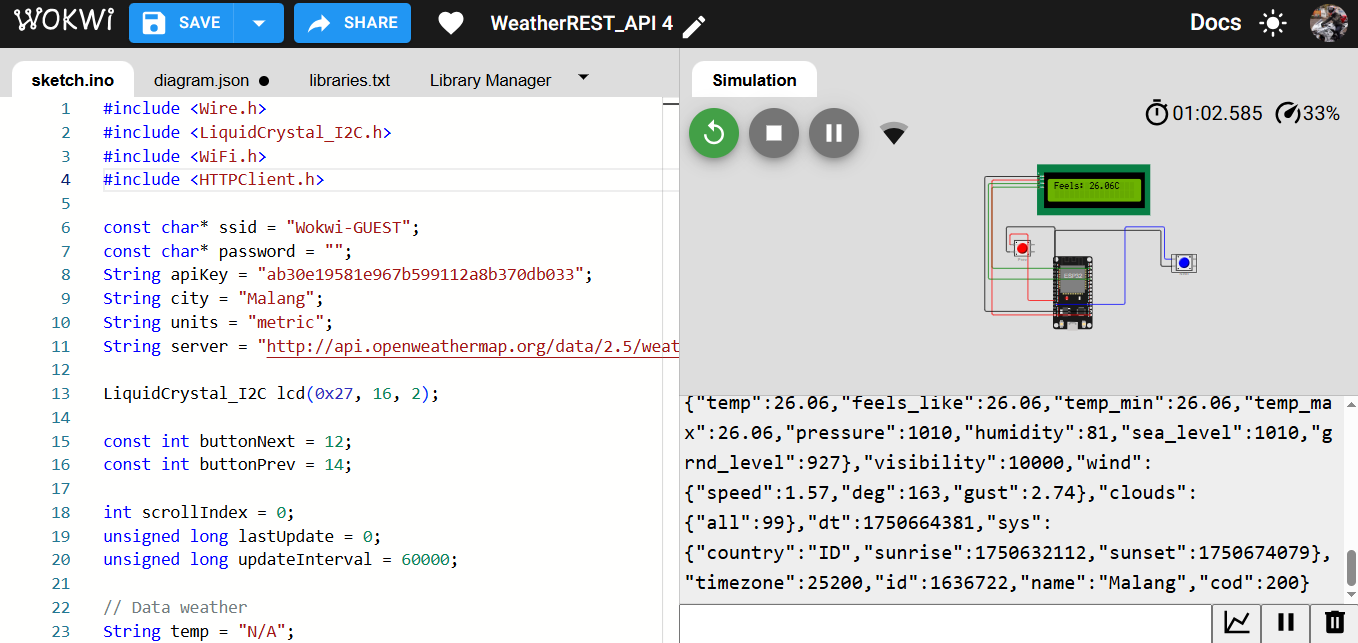
3.1 Hasil

LCD berhasil menampilkan:

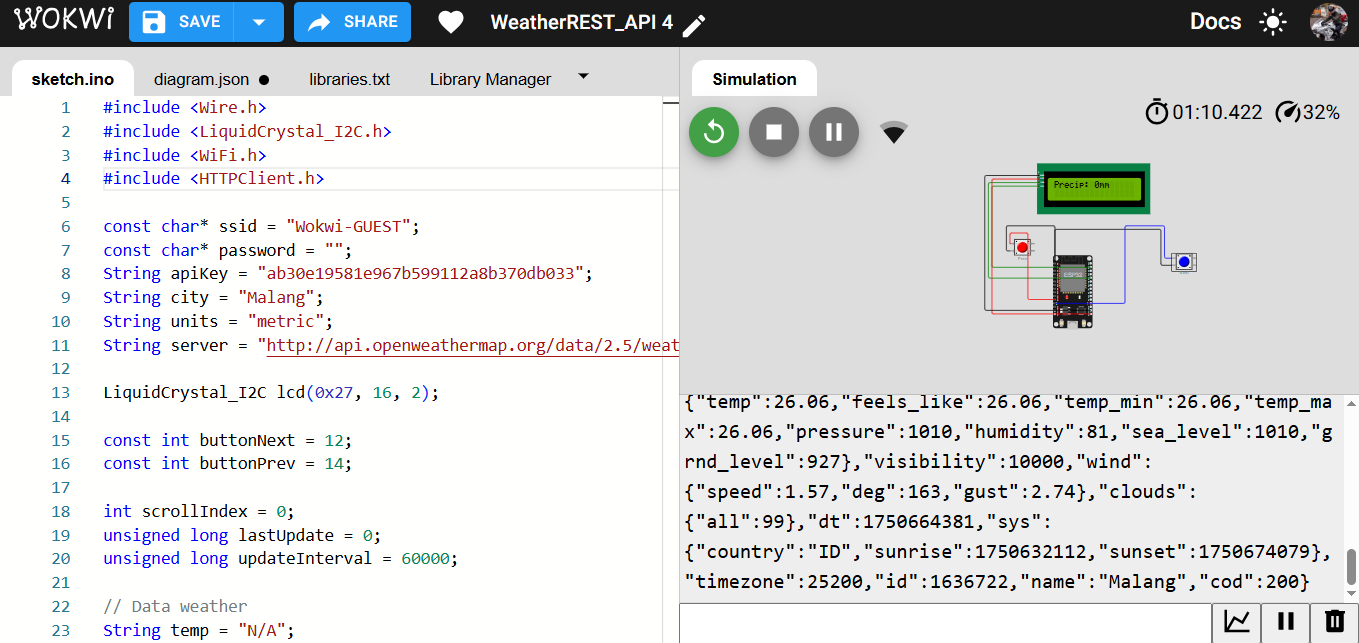
* Suhu (Temp)



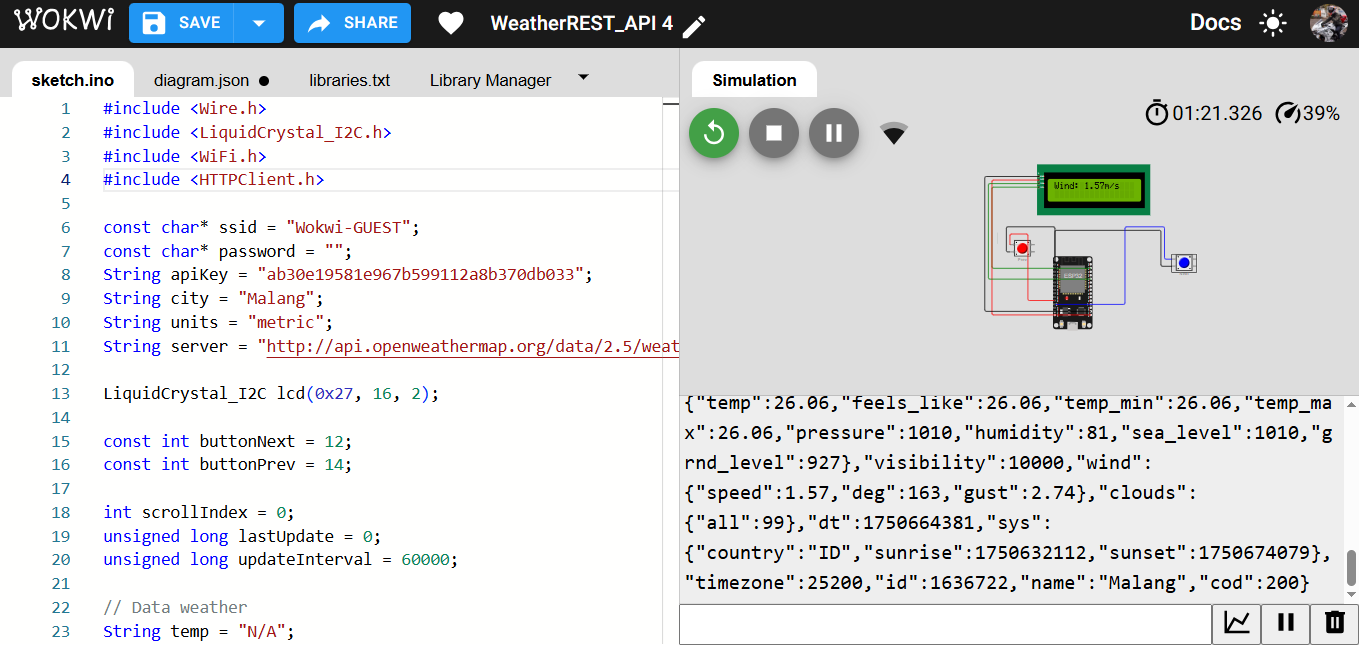
* Suhu terasa (Feels Like)



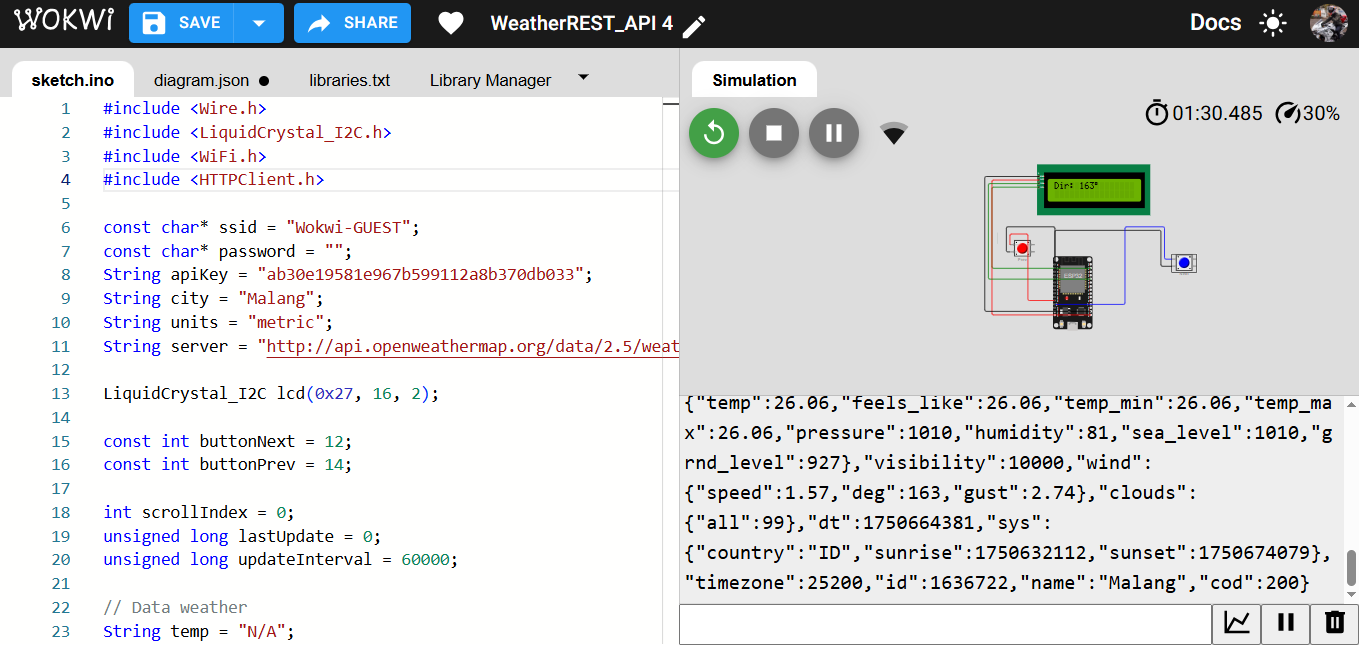
* Curah hujan



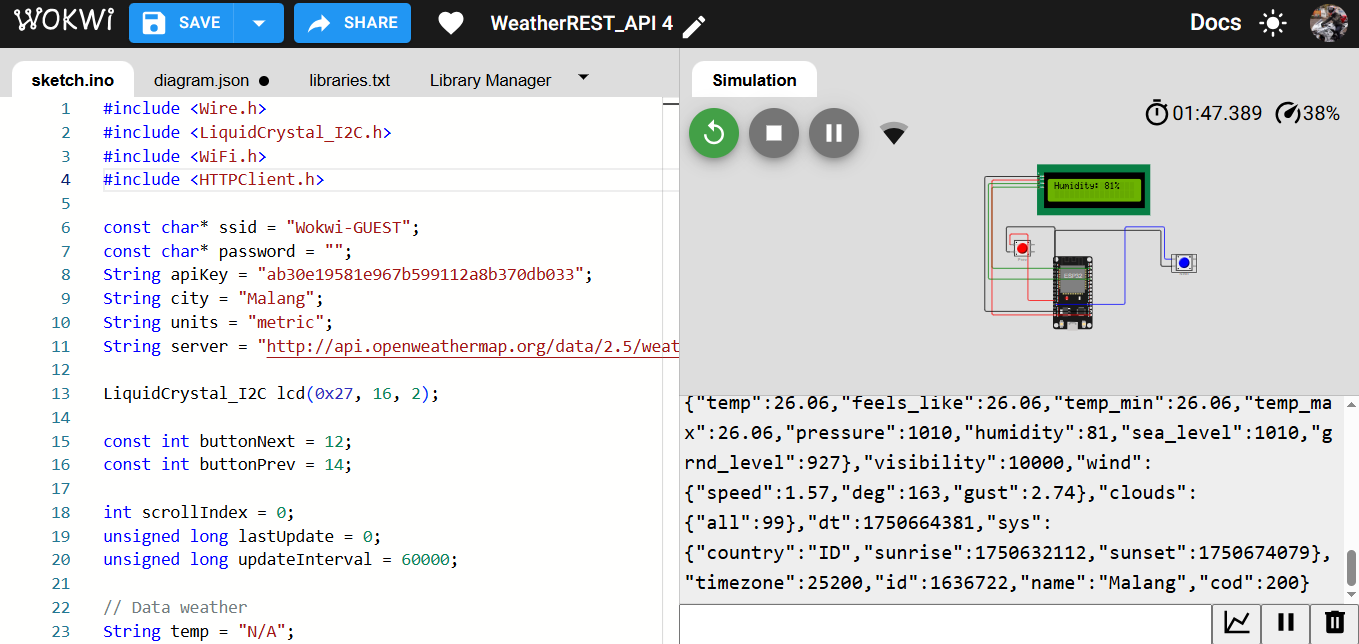
* Kecepatan angin



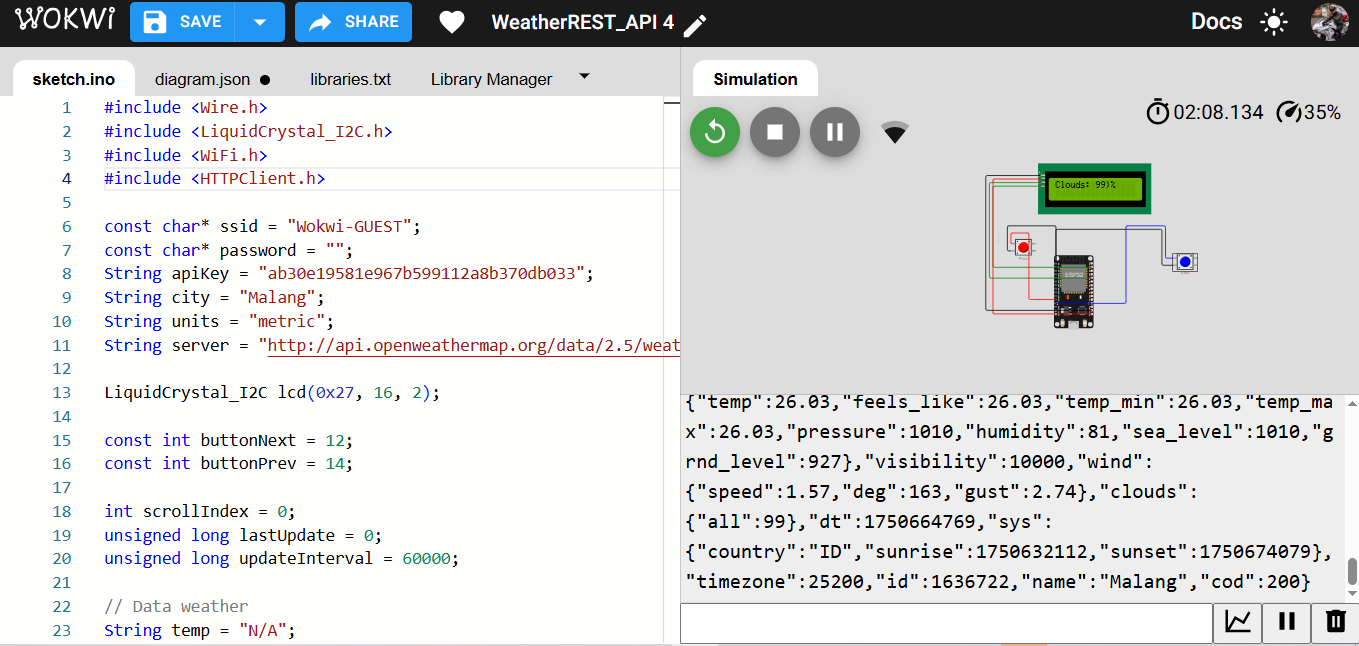
* Arah angin



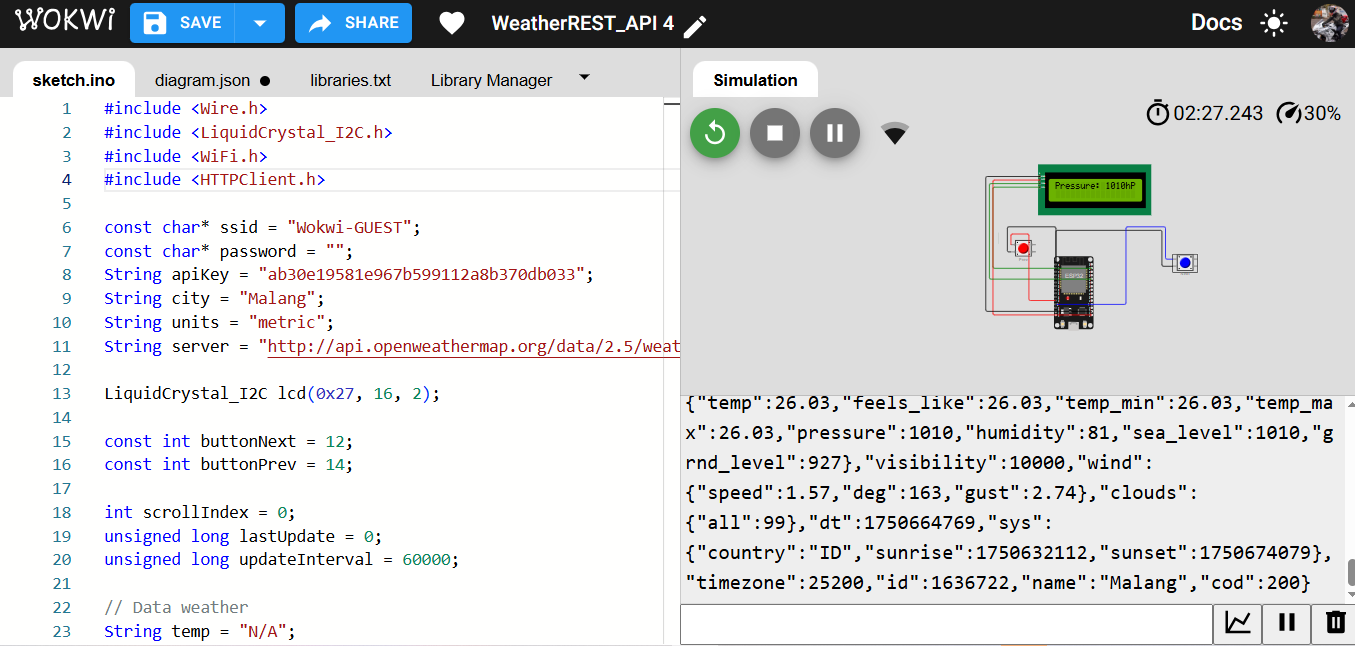
* Kelembapan



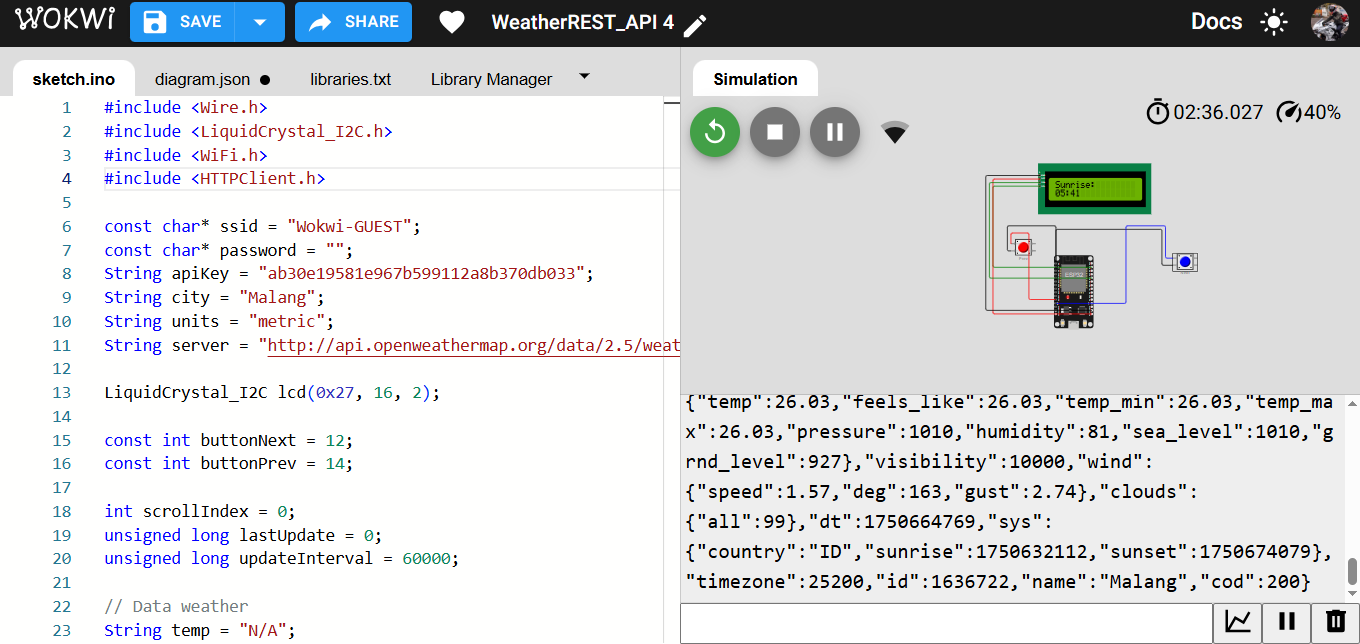
* Tutupan awan



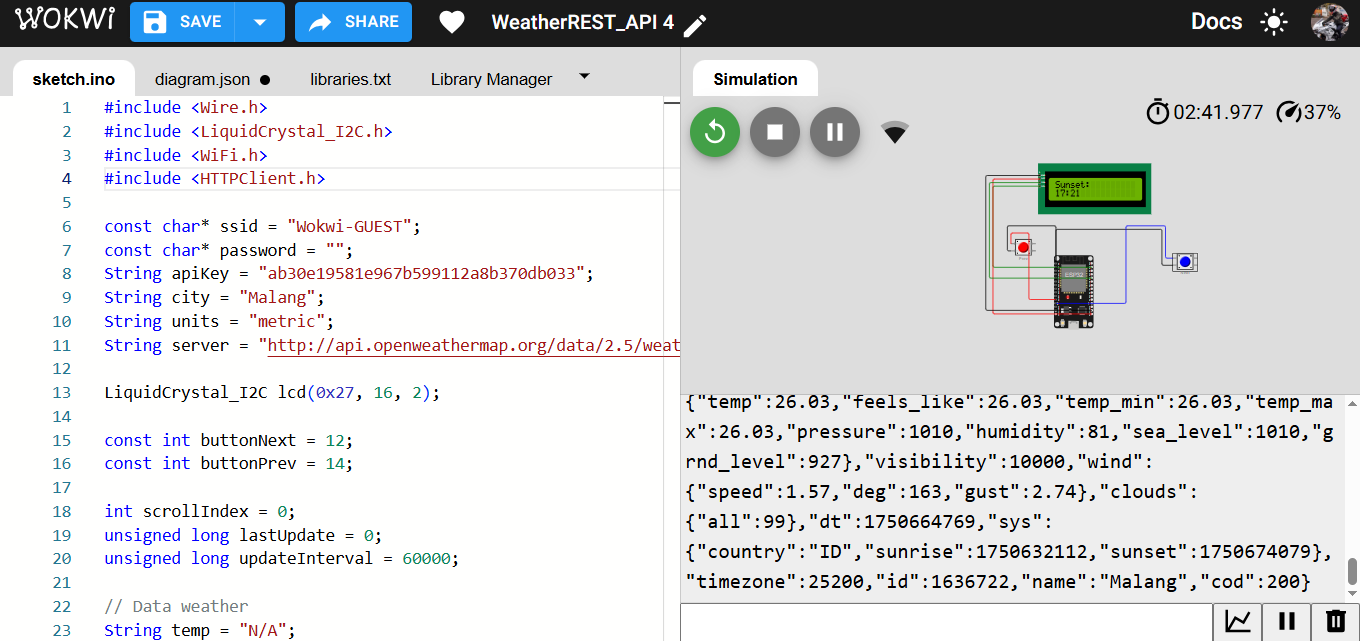
* Tekanan udara



* Waktu Sunrise



* Waktu Sunset



3.2 Pembahasan

Sistem mampu melakukan:

* Koneksi WiFi ke jaringan Wokwi Guest.
* Permintaan data cuaca melalui HTTP.
* Parsing string JSON secara manual.
* Navigasi antar data menggunakan 2 tombol fisik.
* Konversi UNIX timestamp ke format WIB untuk sunrise dan sunset.

Seluruh fungsionalitas diuji pada simulasi Wokwi tanpa error. Penggunaan LCD I2C memungkinkan efisiensi pin dan penyajian informasi secara ringkas.

**4. LAMPIRAN**

4.1 Cuplikan Kode Program Utama

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

#include <WiFi.h>

#include <HTTPClient.h>

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

const char\* password = "";

String apiKey = "ab30e19581e967b599112a8b370db033";

String city = "Malang";

String units = "metric";

String server = "http://api.openweathermap.org/data/2.5/weather?q=" + city + "&units=" + units + "&appid=" + apiKey;

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

const int buttonNext = 12;

const int buttonPrev = 14;

int scrollIndex = 0;

unsigned long lastUpdate = 0;

unsigned long updateInterval = 60000;

// Data weather

String temp = "N/A";

String feels\_like = "N/A";

String precipitation = "0"; // Default 0 mm

String windSpeed = "N/A";

String windDirection = "N/A";

String humidity = "N/A";

String clouds = "N/A";

String pressure = "N/A";

String sunrise = "N/A";

String sunset = "N/A";

// Function prototypes

void fetchWeather();

void displayData(int index);

String getValue(String data, String key);

String getOptionalValue(String data, String key);

String formatTime(String unixTimeStr);

void setup() {

**Serial**.begin(115200);

  lcd.init();

  lcd.backlight();

  pinMode(buttonNext, INPUT\_PULLUP);

  pinMode(buttonPrev, INPUT\_PULLUP);

  // Tampilan awal nama pembuat

  lcd.setCursor(0, 0);

  lcd.print("Created by");

  lcd.setCursor(0, 1);

  lcd.print("Beryl");

  delay(2000);

  lcd.clear();

  lcd.setCursor(0, 0);

  lcd.print("Connecting...");

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL\_CONNECTED) {

    delay(500);

**Serial**.print(".");

  }

  lcd.clear();

  lcd.setCursor(0, 0);

  lcd.print("Connected!");

  delay(1000);

  lcd.clear();

  fetchWeather(); // Fetch pertama

  displayData(scrollIndex);

}

void loop() {

  unsigned long currentMillis = millis();

  // Update data tiap 60 detik

  if (currentMillis - lastUpdate > updateInterval) {

    lastUpdate = currentMillis;

    fetchWeather();

    displayData(scrollIndex);

  }

  // Tombol Next

  if (digitalRead(buttonNext) == LOW) {

    scrollIndex++;

    if (scrollIndex > 9) scrollIndex = 0;

    displayData(scrollIndex);

    delay(300); // Debounce

  }

  // Tombol Prev

  if (digitalRead(buttonPrev) == LOW) {

    scrollIndex--;

    if (scrollIndex < 0) scrollIndex = 9;

    displayData(scrollIndex);

    delay(300); // Debounce

  }

}

void fetchWeather() {

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

    HTTPClient http;

    http.begin(server);

    int httpCode = http.GET();

    if (httpCode > 0) {

      String payload = http.getString();

**Serial**.println(payload);

      temp = getValue(payload, "temp");

      feels\_like = getValue(payload, "feels\_like");

      humidity = getValue(payload, "humidity");

      windSpeed = getValue(payload, "speed");

      windDirection = getValue(payload, "deg");

      pressure = getValue(payload, "pressure");

      clouds = getValue(payload, "all");

      precipitation = getOptionalValue(payload, "\"rain\":{\"1h\":");

      sunrise = formatTime(getValue(payload, "sunrise"));

      sunset = formatTime(getValue(payload, "sunset"));

    } else {

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

    }

    http.end();

  }

}

void displayData(int index) {

  lcd.clear();

  lcd.setCursor(0, 0);

  switch(index) {

    case 0:

      lcd.print("Temp: " + temp + "C");

      break;

    case 1:

      lcd.print("Feels: " + feels\_like + "C");

      break;

    case 2:

      lcd.print("Precip: " + precipitation + "mm");

      break;

    case 3:

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

      break;

    case 4:

      lcd.print("Dir: " + windDirection + (char)223);

      break;

    case 5:

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

      break;

    case 6:

      lcd.print("Clouds: " + clouds + "%");

      break;

    case 7:

      lcd.print("Pressure: " + pressure + "hPa");

      break;

    case 8:

      lcd.print("Sunrise:");

      lcd.setCursor(0, 1);

      lcd.print(sunrise);

      break;

    case 9:

      lcd.print("Sunset:");

      lcd.setCursor(0, 1);

      lcd.print(sunset);

      break;

  }

}

// Fungsi parsing data biasa

String getValue(String data, String key) {

  int startIndex = data.indexOf(key);

  if (startIndex == -1) return "N/A";

  startIndex = data.indexOf(":", startIndex) + 1;

  int endIndex = data.indexOf(",", startIndex);

  if (endIndex == -1) endIndex = data.indexOf("}", startIndex);

  String value = data.substring(startIndex, endIndex);

  value.trim();

  return value;

}

// Fungsi parsing rain opsional

String getOptionalValue(String data, String key) {

  int start = data.indexOf(key);

  if (start == -1) return "0"; // kalau tidak ada hujan, return 0

  start += key.length();

  int end = data.indexOf(",", start);

  if (end == -1) end = data.indexOf("}", start);

  String value = data.substring(start, end);

  value.trim();

  return value;

}

// Format UNIX Time ke WIB jam:menit

String formatTime(String unixTimeStr) {

  unsigned long unixTime = unixTimeStr.toInt();

  // Tambah offset 7 jam untuk WIB

  unixTime += 7 \* 3600;

  int hours = (unixTime % 86400L) / 3600;

  int minutes = (unixTime % 3600) / 60;

  char buffer[6];

  sprintf(buffer, "%02d:%02d", hours, minutes);

  return String(buffer);

}

4.2 Cuplikan diagram.json

{

  "version": 1,

  "author": "Rangga",

  "editor": "wokwi",

  "parts": [

    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": 0, "left": 0, "attrs": {} },

    {

      "type": "wokwi-lcd1602",

      "id": "lcd1",

      "top": -243.2,

      "left": -42.4,

      "attrs": { "pins": "i2c" }

    },

    {

      "type": "wokwi-pushbutton",

      "id": "btn1",

      "top": -41.8,

      "left": -115.2,

      "attrs": { "color": "red", "xray": "1", "label": "Prev" }

    },

    {

      "type": "wokwi-pushbutton",

      "id": "btn2",

      "top": -3.4,

      "left": 316.8,

      "attrs": { "color": "blue", "xray": "", "label": "Next" }

    }

  ],

  "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" ] ],

    [ "btn2:2.l", "esp:GND.2", "black", [ "h-28.8", "v-96", "h-172.8" ] ],

    [ "btn2:1.l", "esp:D12", "blue", [ "h-19.2", "v-86.4", "h-105.6", "v-105.6" ] ],

    [ "btn1:1.l", "esp:D14", "red", [ "h0", "v-28.8", "h48", "v-163.2" ] ],

    [ "btn1:2.l", "esp:GND.2", "black", [ "h-9.6", "v-67", "h86.4" ] ]

  ],

  "dependencies": {}

}

**5. PENUTUP**

Praktikum ini membuktikan bahwa sistem informasi cuaca real-time dapat disimulasikan secara efisien menggunakan ESP32 dan platform Wokwi. Dengan pengambilan data melalui API, parsing manual, serta interaksi tombol sederhana, sistem ini menjadi dasar pengembangan proyek IoT monitoring lingkungan yang lebih kompleks di masa depan.