KELOMPOK 1

RANCANG BANGUN SISTEM MONITORING KELEMBABAN DAN SUHU BERBASIS IOT (HUMIDITY SENSOR PROJECT)

Disusun Oleh Kelompok 1:

Ananda Agung Ismail	20230140123
Nashrul Fikri	20230140105
Bagus Adnan Siraid	20230140101
Damar Sadewa Putra Purwanta	20230140110
Lalu Idrak Yadafi Fatan Nuraga	20230140134

PROGRAM STUDI TEKNOLOGI INFORMASI

FAKULTAS TEKNIK

UNIVERSITAS MUHAMMADIYAH YOGYAKARTA

2026

DAFTAR ISI

Contents

BAB 1: PENDAHULUAN

1.1 Latar Belakang

Pemantauan suhu dan kelembaban merupakan aspek penting dalam berbagai bidang, mulai dari pertanian, pergudangan, hingga kenyamanan ruangan. Teknologi Internet of Things (IoT) memungkinkan proses pemantauan ini dilakukan secara real-time dan terpusat. Dalam proyek ini, kami mengembangkan sistem monitoring menggunakan sensor DHT11 dan mikrokontroler ESP32 yang terintegrasi dengan backend server berbasis Node.js dan database MySQL.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, maka rumusan masalah dalam penelitian proyek akhir ini adalah:

- Bagaimana merancang arsitektur sistem IoT yang mampu mengintegrasikan perangkat keras sensor (DHT11) dengan Web Server berbasis Node.js secara real-time?
- Bagaimana membangun layanan Backend API menggunakan Node.js dan Native MySQL Driver (tanpa bantuan Object-Relational Mapping/ORM) untuk menangani permintaan HTTP POST dari perangkat IoT?

- Bagaimana merancang skema basis data relasional (Entity Relationship Diagram) yang efisien untuk menyimpan data log suhu dan kelembaban dalam jumlah besar?
- Bagaimana memodelkan alur kerja sistem dan interaksi pengguna menggunakan standar Unified Modeling Language (UML) yang mencakup Use Case, Activity, Sequence, dan Class Diagram?

1.3 Tujuan Penelitian

Tujuan yang ingin dicapai dari pembuatan sistem monitoring ini adalah:

- Implementasi Arsitektur IoT: Menghasilkan purwarupa sistem monitoring lingkungan yang menghubungkan sisi client (mikrokontroler) dan server melalui protokol komunikasi HTTP.
- Pengembangan Backend Native: Membuktikan kemampuan pengelolaan data sensor menggunakan kueri SQL manual (Raw Query) pada lingkungan Node.js untuk optimalisasi performa dan pemahaman logika dasar basis data.
- Visualisasi Data: Menyediakan antarmuka (Dashboard) bagi pengguna untuk memantau kondisi suhu dan kelembaban terkini serta melihat riwayat data (logs) secara terstruktur.
- Dokumentasi Sistem: Menghasilkan dokumen perancangan perangkat lunak yang lengkap yang terdiri dari 7 diagram UML sebagai acuan pengembangan sistem.

1.4 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat bagi berbagai pihak, antara lain:

1. Bagi Penulis:
- Memperdalam pemahaman mengenai implementasi arsitektur Internet of Things (IoT) secara menyeluruh dari sisi perangkat keras hingga perangkat lunak.
- Meningkatkan kemampuan teknis dalam pemrograman backend menggunakan Node.js dan manajemen basis data MySQL menggunakan kueri SQL murni (Native Query) tanpa ketergantungan pada library pihak ketiga (ORM).
2. Bagi Akademisi:
- Sebagai referensi atau rujukan bagi penelitian selanjutnya yang berkaitan dengan sistem monitoring berbasis IoT dan pengembangan RESTful API.
- Menjadi contoh implementasi perancangan sistem menggunakan standar UML (Unified Modeling Language) yang komprehensif.
3. Bagi Masyarakat/Industri:
- Menyediakan solusi sistem pemantauan suhu dan kelembaban yang berbiaya rendah (low-cost) namun memiliki performa yang handal.
- Membantu proses digitalisasi pencatatan data lingkungan yang sebelumnya dilakukan secara manual menjadi otomatis dan real-time.

1.5 Batasan Masalah

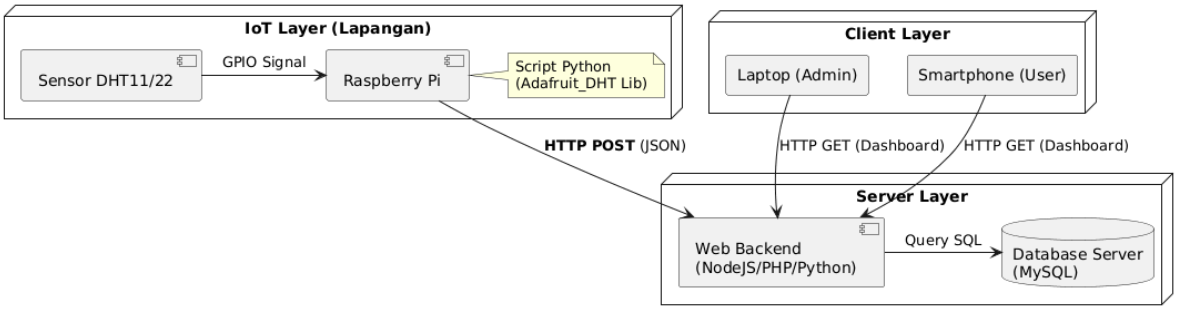
Agar penyusunan laporan dan pengembangan sistem lebih terarah dan tidak meluas dari topik utama, maka penulis menetapkan batasan masalah sebagai berikut:

1. Fokus Perangkat Keras: Sensor yang digunakan adalah DHT11/DHT22 untuk parameter suhu dan kelembaban. Mikrokontroler yang digunakan sebagai pengirim data adalah ESP32 atau Raspberry Pi (disesuaikan dengan simulasi).
2. Lingkup Backend:
   Bahasa pemrograman sisi server adalah JavaScript (Node.js). Tidak menggunakan Framework ORM (seperti Prisma atau Sequelize), melainkan menggunakan library mysql2 untuk eksekusi kueri SQL secara langsung (Native).
3. Lingkup Database: Basis data yang digunakan adalah MySQL/MariaDB yang berjalan pada local server. Fokus perancangan data terbatas pada tabel pengguna (users), perangkat (devices), dan log sensor (logs).
4. Protokol Komunikasi: Pengiriman data dari alat ke server menggunakan metode HTTP REST API (Method POST), bukan menggunakan protokol MQTT atau WebSocket.
5. Fungsionalitas Sistem: Sistem hanya berfokus pada fungsi Monitoring (pembacaan data), tidak mencakup fungsi Controlling (seperti menyalakan kipas/pompa secara otomatis) atau notifikasi kompleks (seperti WA/Email Gateway).

BAB 2: PERANCANGAN SISTEM (UML & DIAGRAM)

Pada bab ini dijelaskan desain sistem menggunakan 7 diagram utama yang telah dirancang.

2.1 Arsitektur Sistem



Gambar 1. Arsitektur Sistem

Gambar berikut menjelaskan topologi jaringan secara keseluruhan. Sistem terdiri dari IoT Device (Sensor DHT) yang mengirim sinyal GPIO ke mikrokontroler, kemudian data dikirim via HTTP POST ke Cloud Server (Node.js), dan disimpan ke Database. User mengakses data melalui Laptop atau HP.

Perancangan sistem monitoring suhu dan kelembaban ini dibangun menggunakan pendekatan arsitektur Client-Server yang terintegrasi dengan perangkat Internet of Things (IoT). Arsitektur sistem menggambarkan bagaimana komponen perangkat keras (hardware) dan

perangkat lunak (software) saling berinteraksi untuk melakukan pengambilan data, pemrosesan, penyimpanan, hingga penyajian informasi kepada pengguna.

Secara garis besar, arsitektur sistem ini terbagi menjadi tiga lapisan utama (layers), yaitu:

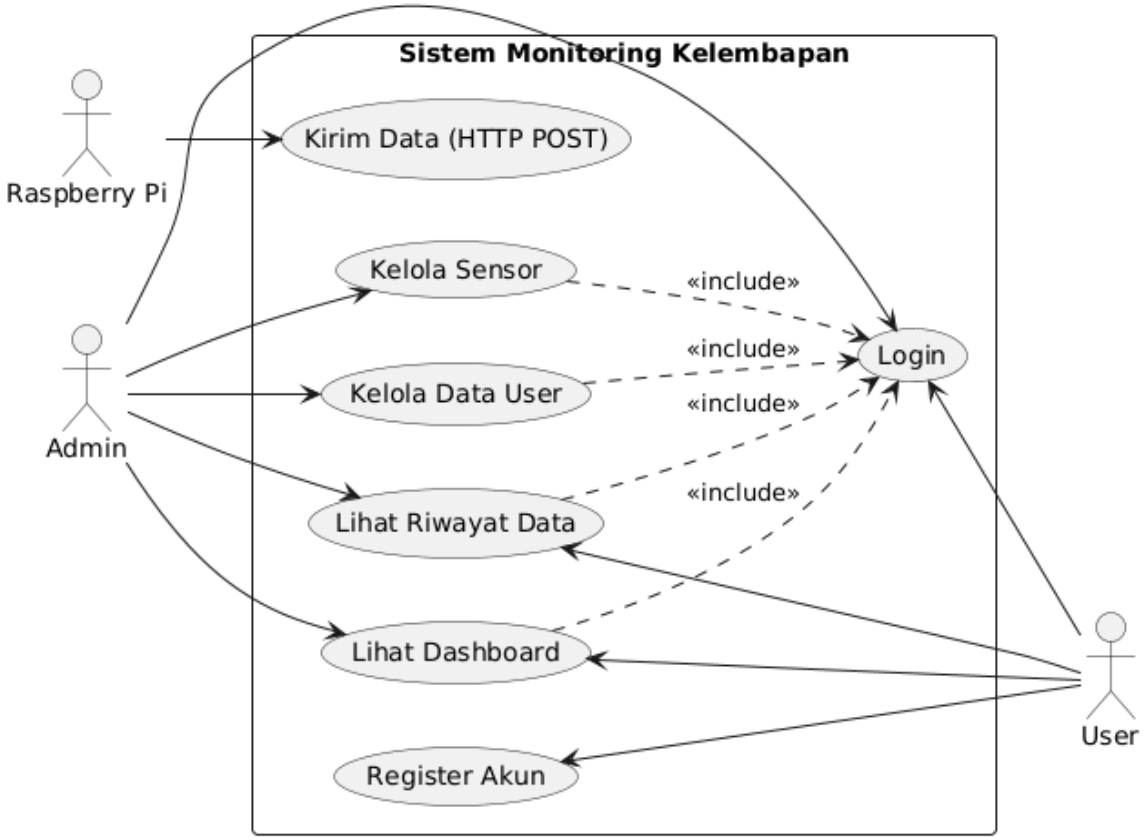
1. Lapisan Perangkat Keras (Device Layer) Lapisan ini merupakan garda terdepan yang berinteraksi langsung dengan lingkungan fisik. Komponen utamanya adalah Sensor DHT22 yang berfungsi mengakuisisi data analog berupa suhu dan kelembaban udara. Data tersebut kemudian dibaca oleh Raspberry Pi 4 Model B melalui pin GPIO (General Purpose Input Output). Berbeda dengan sistem berbasis mikrokontroler sederhana, Raspberry Pi di sini bertindak sebagai Smart Gateway yang tidak hanya membaca sensor, tetapi juga memiliki kemampuan komputasi untuk memvalidasi data sebelum dikirimkan.

2. Lapisan Server dan Pemrosesan (Server & Processing Layer) Lapisan ini bertanggung jawab atas logika bisnis aplikasi. Data yang diterima dari lapisan perangkat keras akan diproses oleh aplikasi backend yang dibangun menggunakan Node.js dengan kerangka kerja Express.js. Fungsi utama lapisan ini meliputi:

- Menerima permintaan data (request) dari perangkat IoT.
- Melakukan pencatatan data (logging) menggunakan pustaka Winston.
- Menyimpan data secara persisten ke dalam basis data MySQL agar dapat diakses kembali sebagai data historis.

3. Lapisan Antarmuka Pengguna (Application/Presentation Layer) Lapisan ini adalah bagian yang berinteraksi langsung dengan pengguna (user). Menggunakan teknologi Web Browser, pengguna dapat memantau kondisi suhu dan kelembaban secara real-time. Halaman antarmuka dibangun menggunakan template engine Mustache yang merender data dari server menjadi tampilan visual berupa tabel atau grafik yang informatif. Lapisan ini memungkinkan pengguna untuk mengakses sistem dari berbagai perangkat (laptop atau smartphone) selama terhubung dalam jaringan yang sama.

2.2 Use Case Diagram



Gambar 2. Use Case Diagram

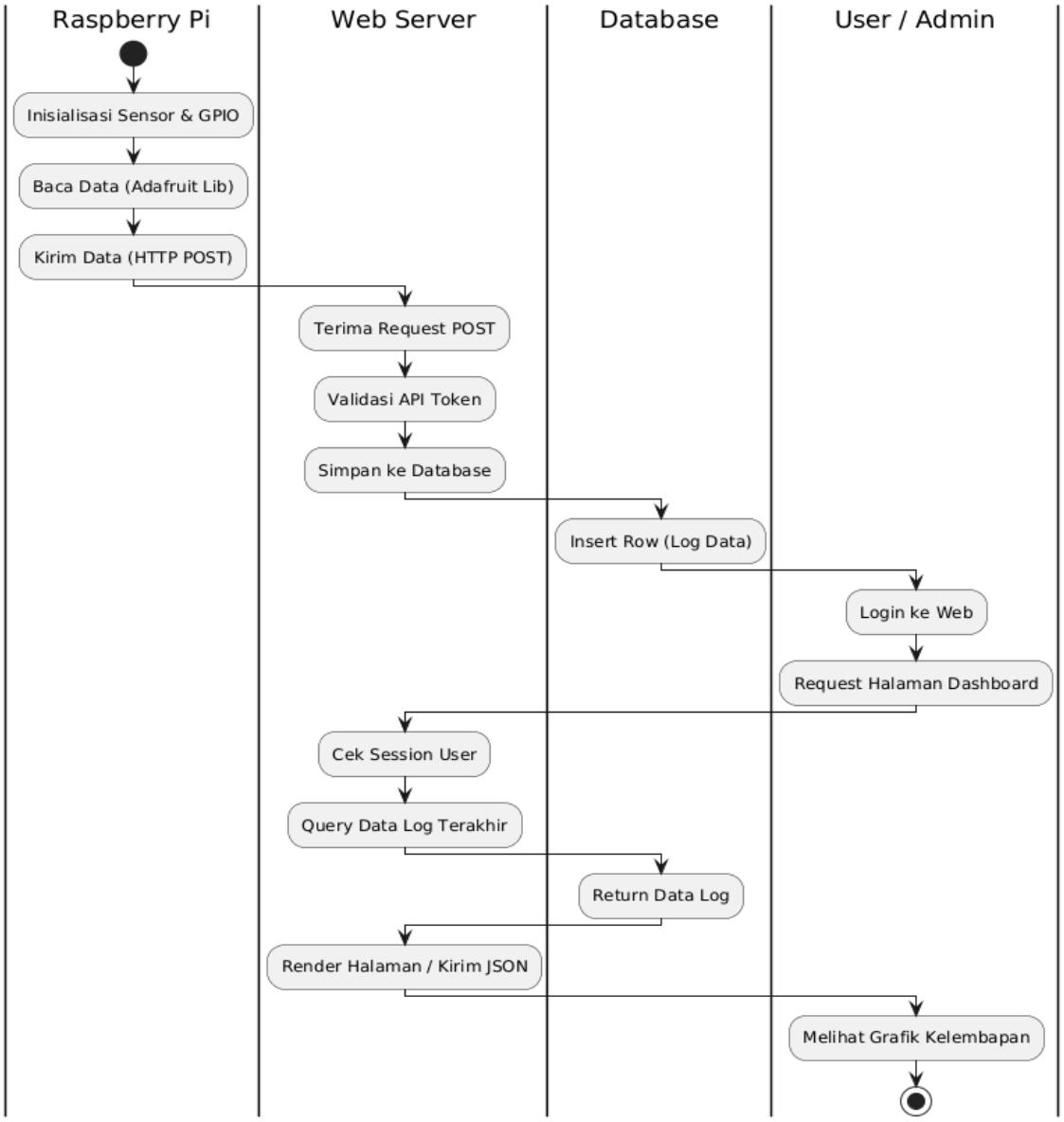
Diagram ini menggambarkan interaksi aktor dengan sistem.

- User: Dapat melakukan Register, Login, melihat Dashboard, dan melihat Riwayat Data.
- Admin: Memiliki hak akses penuh termasuk mengelola data user dan perangkat sensor.
- Raspberry Pi/Device: Bertugas khusus untuk mengirim data sensor ke server.

Use Case Diagram merepresentasikan interaksi fungsional antara aktor (pengguna) dengan sistem yang dibangun. Pada sistem monitoring suhu dan kelembaban ini, terdapat dua aktor utama, yaitu Admin/Operator dan Perangkat IoT (Sistem Otomatis). Diagram ini menggambarkan hak akses dan fitur-fitur yang tersedia dalam aplikasi web.

Aktor Admin memiliki akses penuh untuk melakukan autentikasi (Login) guna masuk ke dalam dashboard sistem. Setelah berhasil masuk, Admin dapat mengakses fitur Monitoring Real-time untuk melihat data suhu dan kelembaban terkini yang dikirimkan oleh sensor. Selain itu, Admin juga memiliki akses ke fitur Laporan Historis, di mana data lampau dapat diakses dan dianalisis berdasarkan rentang waktu tertentu. Di sisi lain, aktor Perangkat IoT bekerja di latar belakang (background service) dengan use case utama Send Sensor Data, yaitu proses otomatis pengiriman data dari Raspberry Pi ke server tanpa intervensi manusia.

2.3 Activity Diagram (Workflow Sistem)



Gambar 3. Activity Diagram

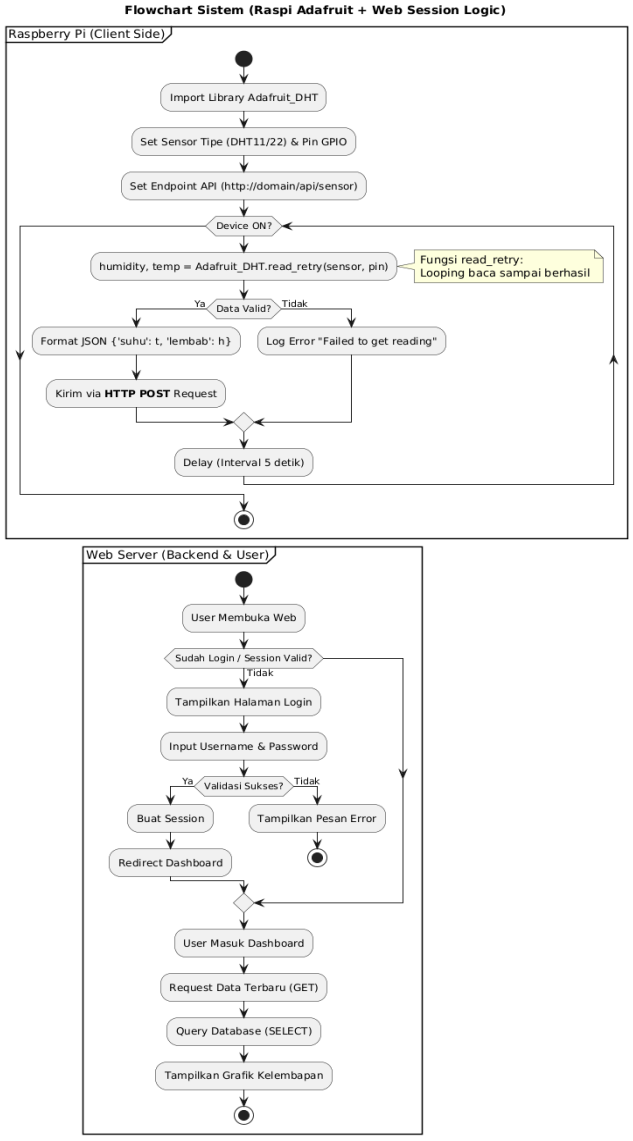
Diagram ini menjelaskan alur kerja sistem secara end-to-end menggunakan swimlane. Dimulai dari Raspberry Pi membaca sensor, server menerima request POST, validasi token, hingga data tersimpan dan ditampilkan ke User dalam bentuk grafik.

Activity Diagram menggambarkan alur kerja (workflow) prosedural dari sistem, mulai dari inisialisasi hingga penyajian data. Diagram ini memvisualisasikan logika operasional sistem secara step-by-step.

Alur aktivitas dimulai ketika sistem Raspberry Pi dinyalakan (Start Node). Sistem akan melakukan inisialisasi sensor DHT22 dan mencoba terhubung ke jaringan WiFi. Setelah koneksi berhasil, sistem memasuki siklus pembacaan sensor (Reading Loop). Sensor membaca parameter fisik suhu dan kelembaban, kemudian program melakukan validasi data.

Jika data valid, sistem akan mengirimkan data tersebut ke server melalui protokol HTTP POST. Di sisi server, aktivitas berlanjut dengan penerimaan data, penyimpanan ke dalam database MySQL, dan pembaruan tampilan pada antarmuka web pengguna. Alur ini berulang secara terus-menerus (looping) selama perangkat menyala, memastikan data yang tampil di layar pengguna selalu up-to-date.

2.4 Flowchart Logika Program



Gambar 4. Flowchart Sistem

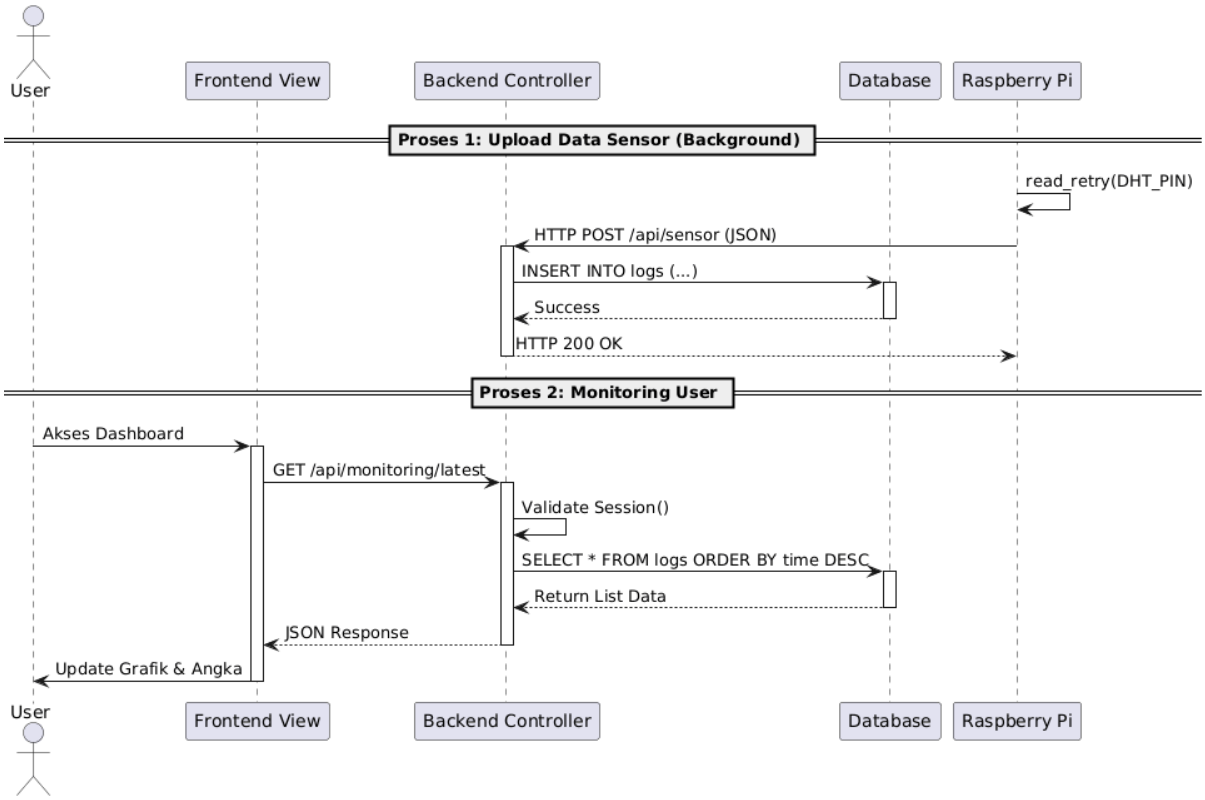
Diagram ini berfokus pada logika algoritma di sisi perangkat (Client) dan Server.

- Sisi Client: Terdapat logika retry (baca ulang) jika sensor gagal membaca, dan pengiriman JSON payload.
- Sisi Server: Terdapat pengecekan API Key. Jika Authorized, data disimpan (INSERT). Jika tidak, kembalikan error 401.

Sequence Diagram memperlihatkan interaksi dinamis antar objek di dalam sistem berdasarkan urutan waktu (time sequence). Diagram ini menjelaskan bagaimana pesan (messages) dikirimkan antar komponen untuk menyelesaikan satu skenario tertentu, misalnya skenario "Menampilkan Data Sensor".

Pada diagram ini, objek User mengirimkan permintaan (request) melalui Web Browser untuk mengakses halaman utama. Browser kemudian meneruskan permintaan tersebut ke Controller (Node.js Server). Controller memproses permintaan dengan memanggil fungsi pada Model untuk mengambil data terbaru dari Database. Database merespons dengan mengembalikan result set berisi data suhu dan kelembaban. Selanjutnya, Controller mengirimkan data tersebut ke View (Template Engine Mustache) untuk dirender menjadi halaman HTML. Akhirnya, halaman yang sudah berisi data visual dikirimkan kembali ke Web Browser untuk ditampilkan kepada User. Diagram ini menegaskan penerapan arsitektur MVC (Model-View-Controller) dalam sistem.

2.5 Sequence Diagram



Gambar 5. Sequence Diagram

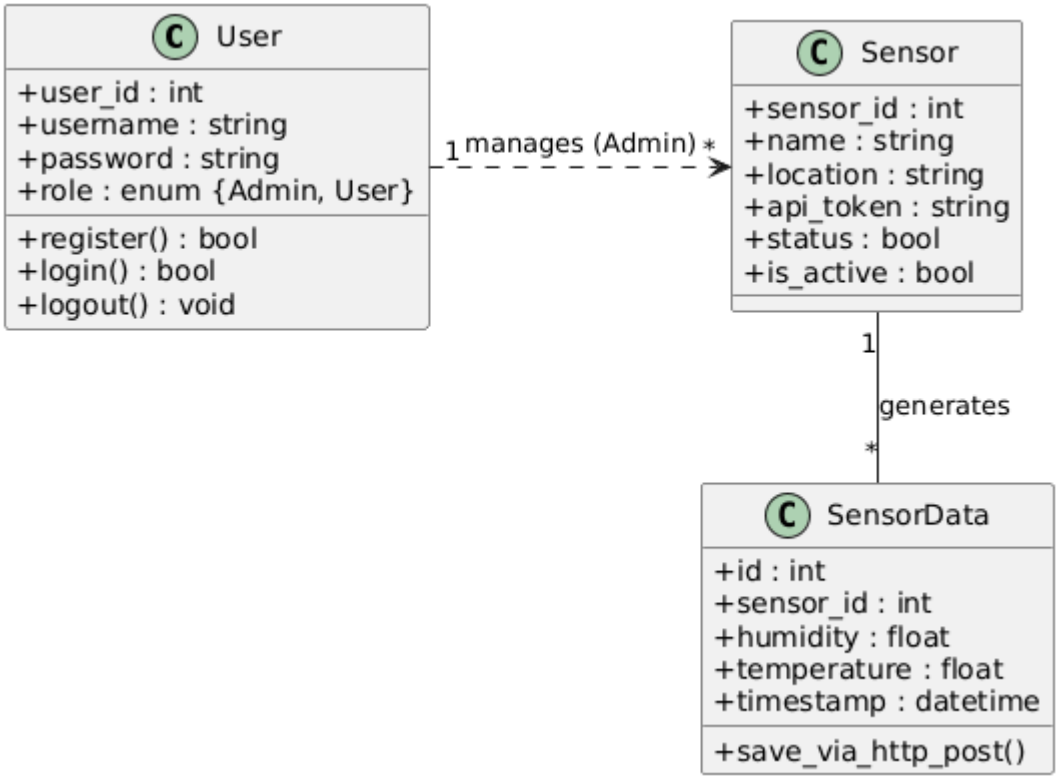
Diagram ini menjabarkan urutan waktu pengiriman pesan.

1. Loop Pengiriman: Device terus menerus mengirim HTTP POST ke API.
2. Akses Monitoring: Saat User membuka dashboard, Frontend meminta data (GET) ke Backend, Backend melakukan query SELECT ke Database, lalu data dikembalikan ke User.

Class Diagram menggambarkan struktur statis sistem dengan menunjukkan kelas-kelas (classes) yang membangun perangkat lunak beserta atribut dan metodenya. Diagram ini menjadi cetak biru (blueprint) dari struktur kode program dan basis data.

Sistem ini terdiri dari beberapa kelas utama. Kelas User memiliki atribut seperti username, password, dan role dengan metode login() dan logout(). Kelas SensorData berfungsi menampung data pembacaan dengan atribut id, suhu, kelembaban, dan timestamp. Kelas ini memiliki relasi asosiasi dengan kelas Device, yang merepresentasikan identitas perangkat Raspberry Pi dengan atribut deviceID dan lokasi. Hubungan antar kelas menunjukkan bahwa satu Device dapat menghasilkan banyak SensorData (relasi One-to-Many), yang memungkinkan sistem untuk dikembangkan lebih lanjut dengan banyak perangkat sensor di masa depan.

2.6 Class Diagram



Gambar 6. Class Diagram

Deskripsi Class Diagram Berdasarkan rancangan Class Diagram, sistem dibangun di atas tiga kelas utama yang saling berinteraksi:

- User: Bertanggung jawab menangani otentikasi pengguna, meliputi proses register, login, dan logout. Kelas ini juga menyimpan atribut role untuk membedakan hak akses antara 'Admin' dan 'User', di mana Admin memiliki hubungan association untuk mengelola (manages) perangkat sensor.

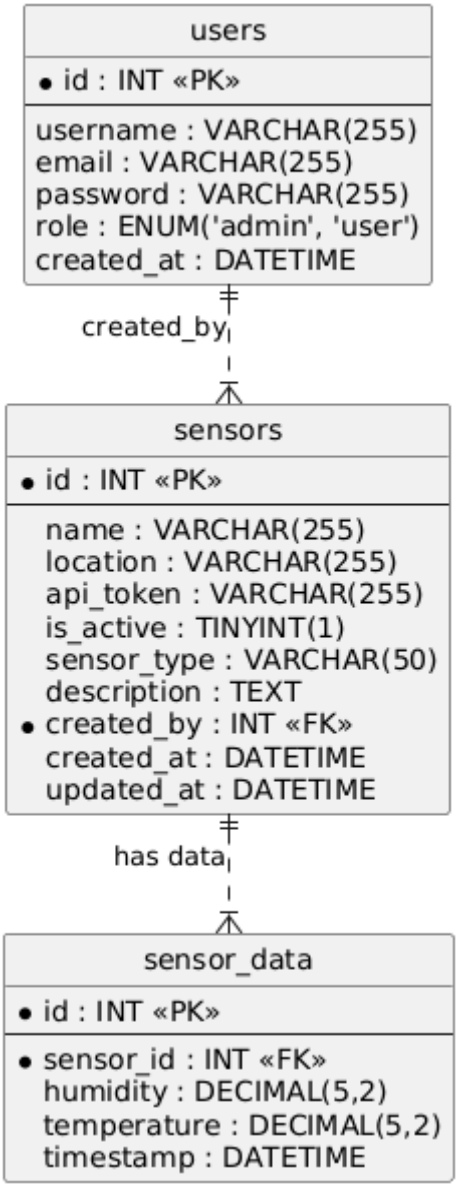
- Sensor: Merepresentasikan identitas perangkat keras (Raspberry Pi/DHT). Kelas ini menyimpan konfigurasi statis seperti nama perangkat, lokasi pemasangan, api_token untuk keamanan, serta status aktif perangkat (is_active).
- SensorData: Berfungsi sebagai objek log yang menangani penyimpanan data telemetri. Kelas ini memiliki metode save_via_http_post() untuk mengirimkan atribut suhu (temperature), kelembapan (humidity), dan waktu perekaman (timestamp) ke server.

Implementasi Entity Relationship Diagram (ERD) Entity Relationship Diagram (ERD) memodelkan struktur data yang disimpan dalam basis data MySQL berdasarkan Class Diagram di atas. Rancangan database kini terdiri dari tiga tabel utama untuk memastikan normalisasi data yang baik:

1. Tabel users: Berfungsi menyimpan data kredensial pengguna. Tabel ini memuat kolom username, password, dan role untuk keperluan keamanan dan otorisasi sistem.
2. Tabel sensors: Tabel ini merupakan representasi dari kelas Sensor. Berfungsi menyimpan metadata perangkat agar sistem dapat mengelola banyak sensor sekaligus. Kolom utamanya meliputi sensor_id (PK), name, location, dan api_token.
3. Tabel sensor_logs (Representasi SensorData): Berfungsi sebagai tabel transaksional untuk menyimpan riwayat pembacaan. Tabel ini memiliki kolom id (PK), temperature, humidity, dan recorded_at. Tabel ini memiliki relasi Foreign Key ke tabel sensors (via kolom sensor_id) untuk memetakan data mana yang dihasilkan oleh perangkat mana.

Struktur tiga tabel ini dirancang untuk menjaga integritas referensial dan memudahkan skalabilitas jika di kemudian hari jumlah perangkat sensor bertambah.

2.7 Entity Relationship Diagram (ERD)



Gambar 7. Entity Relationship Diagram (ERD)

Perancangan basis data menggunakan MySQL. Sistem ini menggunakan tabel relasional:

- tb_users: Menyimpan data pengguna.
- tb_sensors: Menyimpan token API perangkat dan informasi Perangkat.
- tb_sensors_data: Tabel transaksi utama yang menyimpan nilai suhu (temp_val) dan kelembaban (humid_val) yang berelasi dengan tabel device.

Deployment Diagram memvisualisasikan arsitektur fisik dari sistem, menunjukkan bagaimana komponen perangkat lunak didistribusikan ke node perangkat keras (hardware nodes).

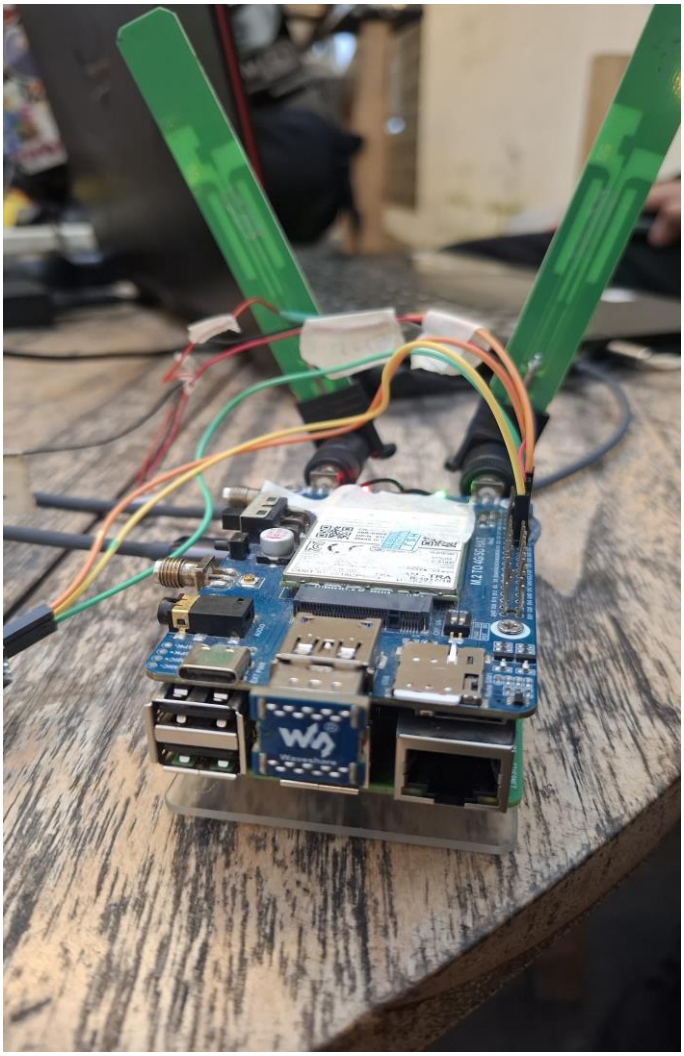
Diagram ini terdiri dari tiga node utama yang saling terhubung melalui jaringan TCP/IP. Node pertama adalah IoT Node (Raspberry Pi 4B), yang menjalankan runtime environment untuk pembacaan sensor. Node kedua adalah Server Node (Laptop/PC atau Cloud Server), yang menjadi tempat berjalannya Web Server (Node.js) dan Database Server (MySQL). Node ketiga adalah Client Node (Laptop/Smartphone Pengguna), yang menjalankan aplikasi Web Browser. Komunikasi antar node menggunakan protokol HTTP standar, di mana IoT Node bertindak sebagai pengirim data (Publisher) dan Client Node sebagai penerima informasi (Subscriber). Diagram ini memberikan gambaran jelas mengenai topologi jaringan dan infrastruktur yang dibutuhkan agar sistem dapat berjalan dengan baik.

BAB 3: IMPLEMENTASI APLIKASI

3.1 Alat dan Bahan

Implementasi Antarmuka (Screenshot Aplikasi)

Berikut adalah tampilan antarmuka sistem yang telah berhasil dibangun:



Gambar 8. Sensor yang dipakai

Dalam pelaksanaan praktikum dan pengembangan sistem monitoring suhu dan kelembaban ini, dibutuhkan perangkat keras (hardware) dan perangkat lunak (software) yang saling terintegrasi. Adapun rincian alat dan bahan yang digunakan adalah sebagai berikut:

3.1.1 Alat

Alat merupakan perangkat utama yang digunakan untuk memproses data dan menjalankan logika program. Alat yang digunakan meliputi:

1. Raspberry Pi 4 Model B

Berfungsi sebagai Single Board Computer (SBC) yang menjadi otak utama sistem. Perangkat ini menggantikan fungsi mikrokontroler konvensional dengan kemampuan komputasi yang lebih tinggi, memiliki fitur konektivitas WiFi built-in, dan menjalankan sistem operasi Linux untuk eksekusi program backend.

2. Laptop / PC

Digunakan sebagai media untuk menulis kode program (coding), mengakses Raspberry Pi secara jarak jauh (remote SSH), serta memonitor hasil pembacaan sensor melalui terminal atau web browser.

3. Power Supply (USB Type-C)

Digunakan sebagai sumber daya listrik utama untuk menyalakan Raspberry Pi 4B dengan tegangan 5V dan arus minimal 3A agar sistem berjalan stabil.

4. Micro SD Card (16 GB / 32 GB)

Berfungsi sebagai media penyimpanan sistem operasi (Raspberry Pi OS) dan tempat penyimpanan file kode program beserta database.

3.1.2 Bahan

Bahan merupakan komponen pendukung dan sensor yang digunakan untuk pengambilan data fisik. Bahan yang digunakan meliputi:

1. Sensor DHT22 (AM2302)

Sensor ini digunakan untuk mengukur parameter suhu (temperature) dan kelembaban udara (humidity) di sekitar alat. DHT22 dipilih karena memiliki akurasi yang lebih baik dan jangkauan pembacaan yang lebih luas dibandingkan seri DHT11.

2. Kabel Jumper (Female-to-Female)

Digunakan sebagai media penghantar listrik dan data untuk menghubungkan kaki-kaki (pin) sensor DHT22 ke header GPIO (General Purpose Input Output) pada Raspberry Pi.

3. Breadboard (Opsional)

Papan rangkaian yang digunakan untuk menyusun komponen elektronik sementara tanpa perlu melakukan penyolderan.

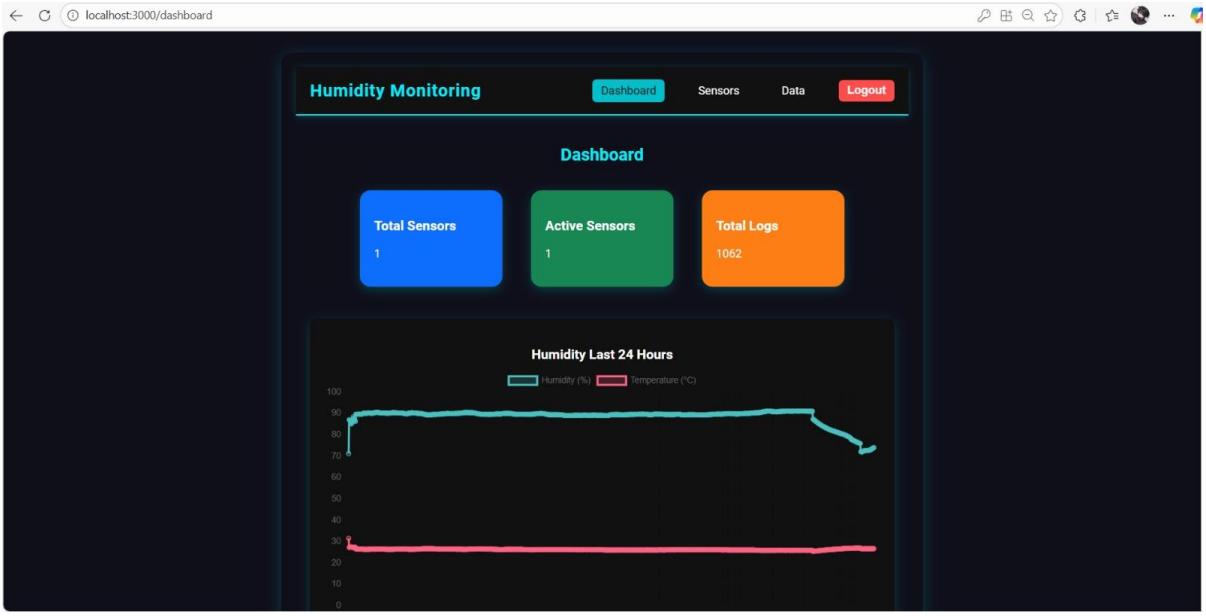
3.2 Gambaran Umum Sistem

Sistem yang dirancang adalah sistem pemantauan kondisi lingkungan berbasis Internet of Things (IoT). Konsep kerja sistem ini dimulai dari sensor DHT22 yang mendeteksi perubahan suhu dan kelembaban udara. Data analog yang ditangkap oleh sensor dikonversi menjadi sinyal digital dan dikirimkan ke Raspberry Pi 4B.

Di dalam Raspberry Pi, data tersebut dibaca oleh program aplikasi berbasis Node.js. Program akan memvalidasi data yang masuk, menampilkannya pada console sistem, dan menyimpannya ke dalam basis data (database) untuk keperluan pencatatan historis (logging). Pengguna dapat memantau data tersebut secara real-time melalui antarmuka perangkat lunak.

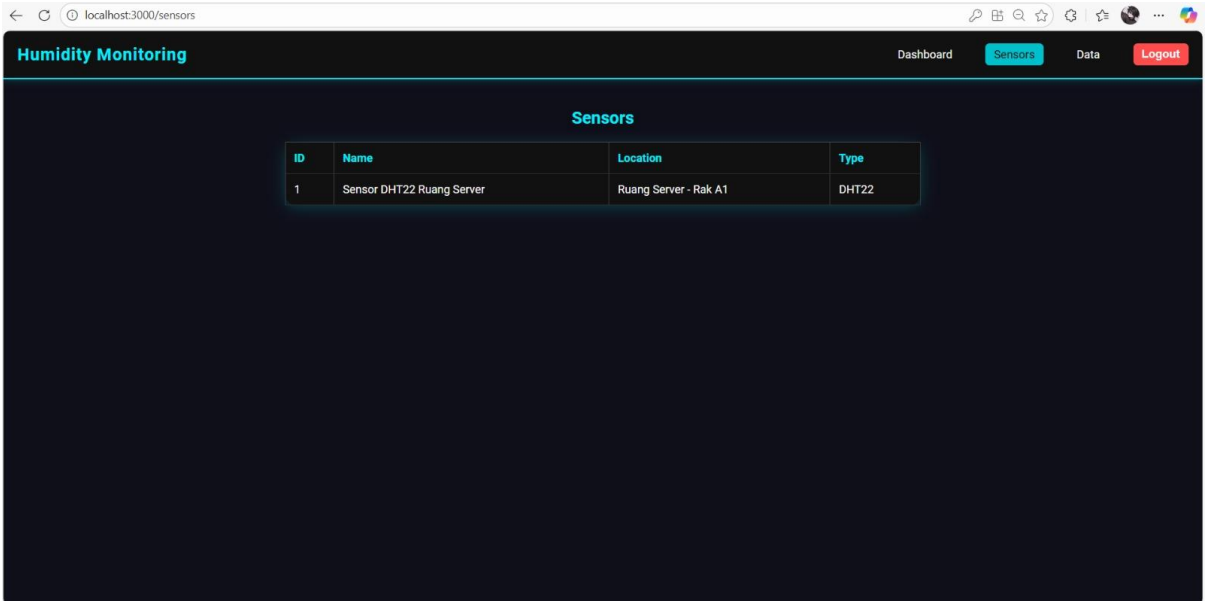
3.3 Tampilan Web per Homepage

3.3.1 Tampilan Dashboard



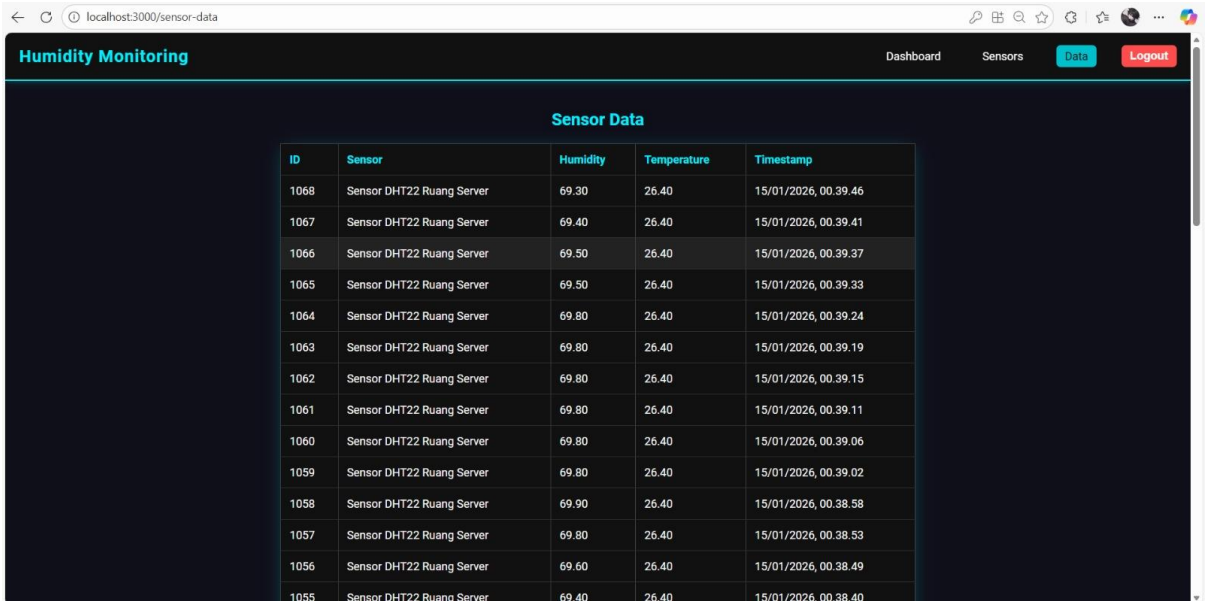
Gambar 9. Dashboard User

3.3.2 Tampilan Sensors Users



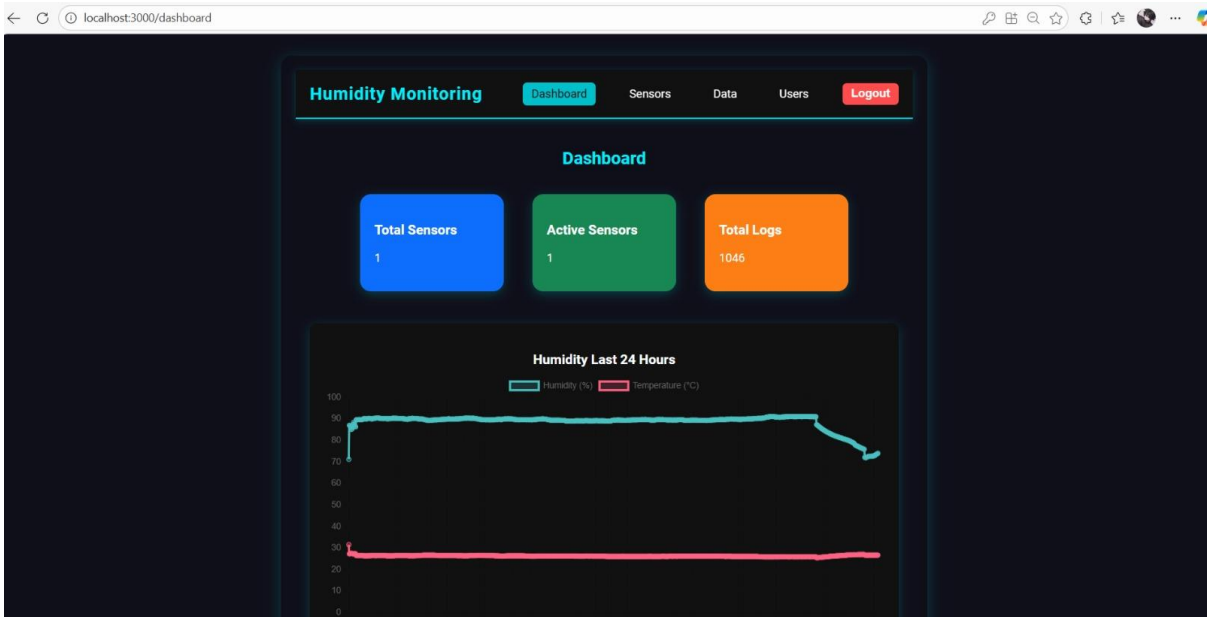
Gambar 10. Tampilan Sensor User

3.3.3 Tampilan Data Users



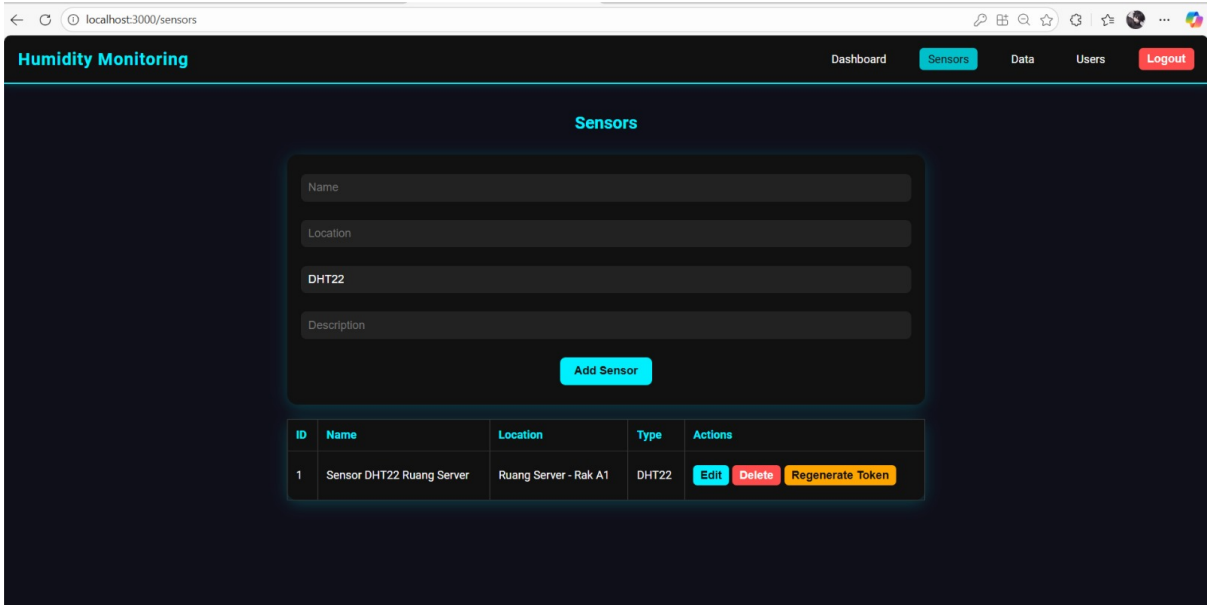
Gambar 11. Sensor Data User

3.3.4 Tampilan Dashboard Admin



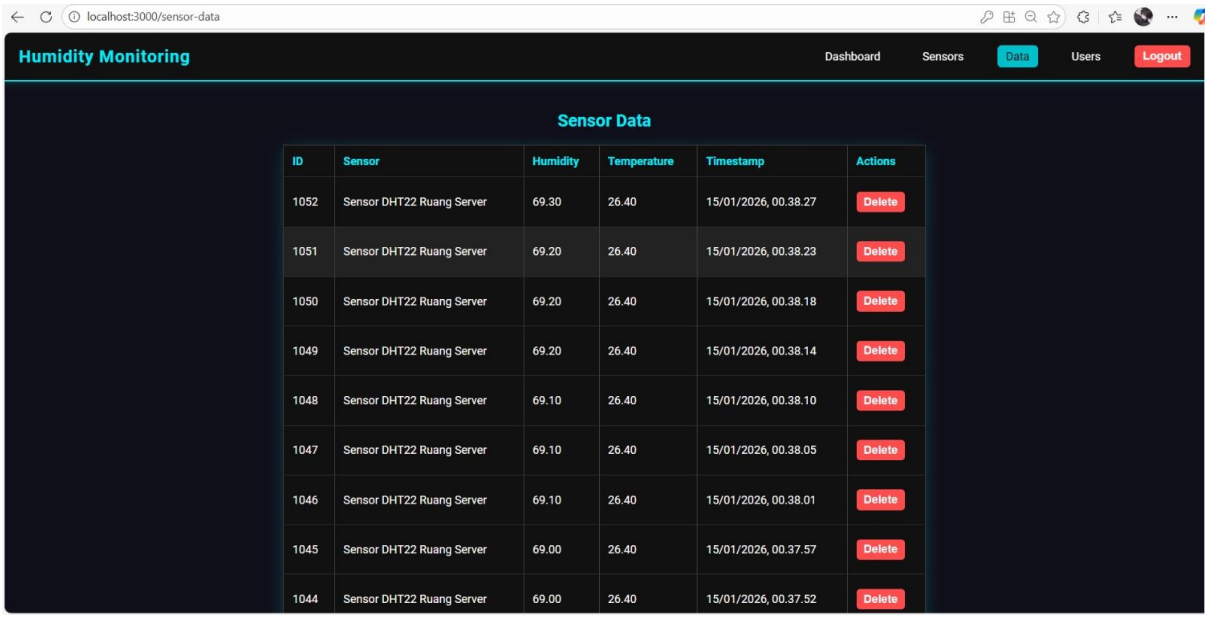
Gambar 12. Dashboard Admin

3.3.5 Tampilan Sensors Admin



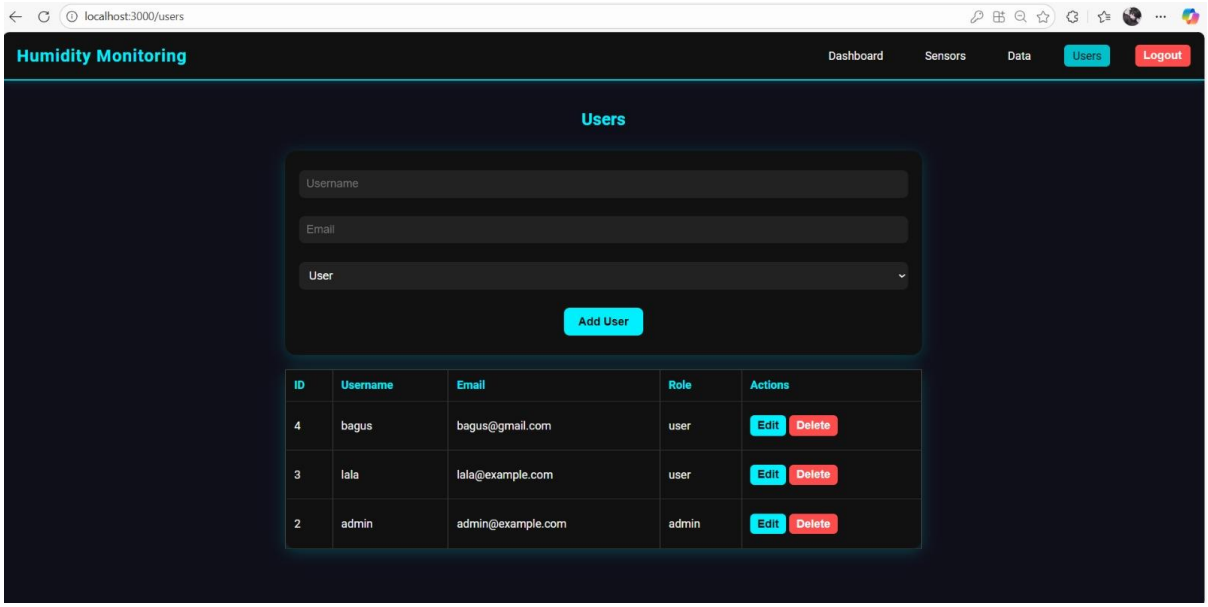
Gambar 13. Sensor Admin

3.3.6 Tampilan Data Admin



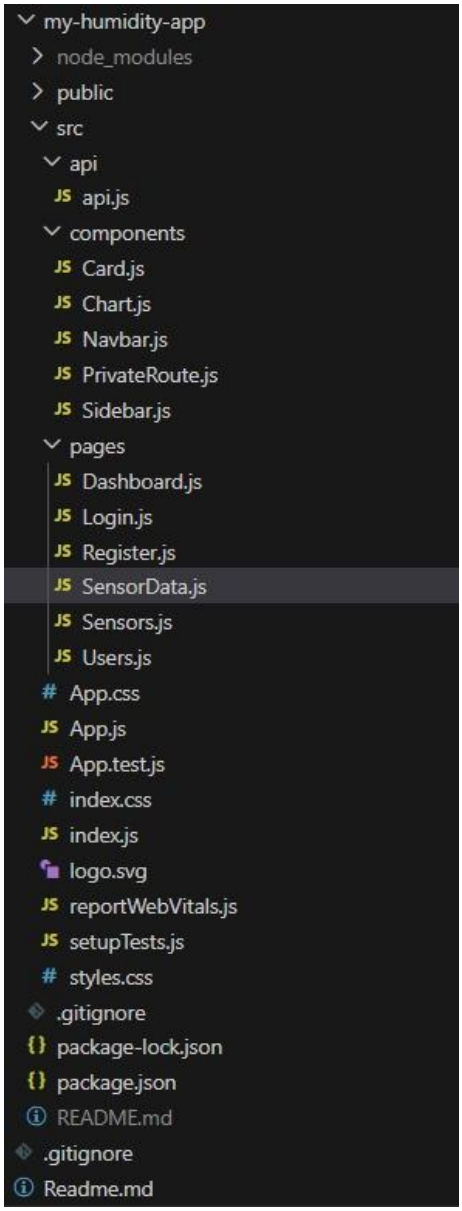
Gambar 14. Sensor Data Admin

3.3.7 Tampilan Tambah Users (Admin)

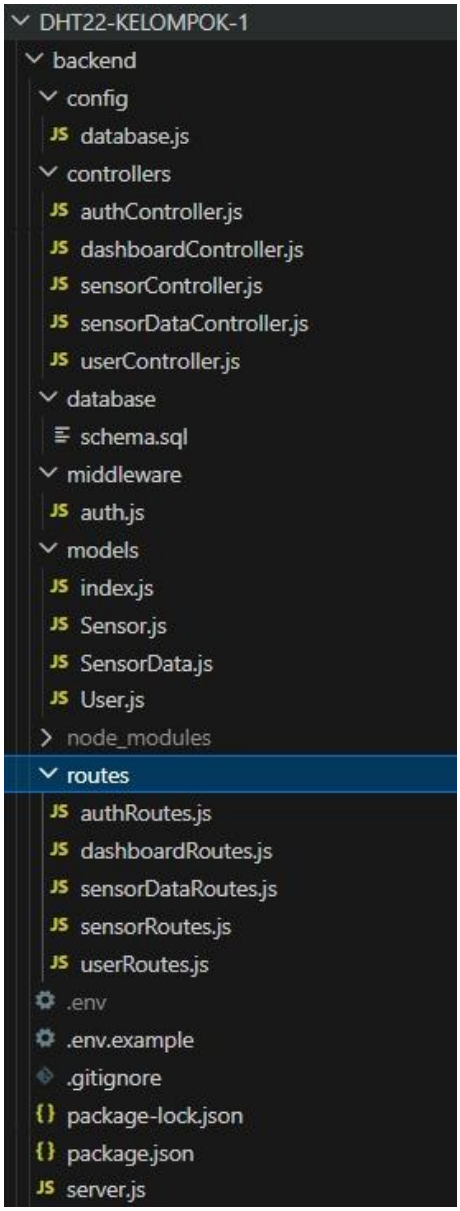


Gambar 15. Add User Admin

3.3.8 Stuktur Folder



Gambar 16. Struktur Folder Pt.1



Gambar 17. Struktur Folder Pt.2

3.3.9 Source Code

https://github.com/PAW-KELOMPOK1-DHT/DHT22-KELOMPOK-1

3.3.10 Code Raspberry

Kode ini berfungsi sebagai **klien IoT** yang bertugas membaca data fisik dan mengirimkannya ke server. Berikut rincian alurnya:

1. Persiapan & Konfigurasi (Setup)
    a. Kode mengimpor library penting: adafruit_dht (untuk komunikasi dengan sensor) dan requests (untuk mengirim data via internet).

    b. Mengatur konfigurasi target: URL API tujuan (API_URL), Token keamanan (API_TOKEN), dan interval pengiriman data setiap 60 detik.
    c. Menginisialisasi sensor DHT22 pada pin GPIO 23.
2. Fungsi Pengiriman (send_to_server)
    a. Fungsi ini membungkus data suhu dan kelembapan ke dalam format JSON.
    b. Melakukan request HTTP POST ke server.
    c. Menyertakan Header berisi X-API-Token untuk autentikasi (agar server tahu ini data valid dari perangkatmu).
3. Loop Utama (Main Loop)
    a. Program berjalan terus-menerus (while True).
    b. Membaca Data: Mengambil nilai temperature dan humidity dari sensor.
    c. Monitoring Lokal: Menampilkan nilai tersebut di layar terminal (console) agar kamu bisa melihat statusnya secara langsung.
    d. Error Handling (Penting): Menggunakan blok try-except RuntimeError. Sensor DHT seringkali gagal membaca data sesekali (timing sensitive). Kode ini dirancang agar tidak crash saat gagal baca, melainkan hanya mencetak pesan error dan mencoba lagi nanti.
    e. Jeda: Istirahat selama 60 detik (time.sleep) sebelum mengulang proses.

sensor.py > ...

```python
import time
import board
import adafruit_dht
import requests
from datetime import datetime

# ========== KONFIGURASI ==========
API_URL = "http://100.96.103.85:5000/api/sensor-data"
API_TOKEN = "your_sensor_api_token_here"  # Ganti dengan API Token dari admin
SEND_INTERVAL = 60  # Kirim data setiap 60 detik
# ================================

# Initialize DHT22 sensor on GPIO 23
dht = adafruit_dht.DHT22(board.D23)

print("=" * 50)
print("  RASPBERRY PI HUMIDITY MONITORING")
print("=" * 50)
print(f"Server: {API_URL}")
print(f"Interval: {SEND_INTERVAL} seconds")
print("Press Ctrl+C to stop\n")

def send_to_server(temperature, humidity):
    """Kirim data ke backend API"""
    try:
        headers = {
            "Content-Type": "application/json",
            "X-API-Token": API_TOKEN
        }
        data = {
            "temperature": round(temperature, 2),
            "humidity": round(humidity, 2)
        }

        response = requests.post(API_URL, json=data, headers=headers, timeout=10)

        if response.status_code == 201:
            print("✓ Data sent to server successfully")
            return True
        else:
            print(f"✗ Server error: {response.status_code}")
            return False

    except Exception as e:
        print(f"✗ Error sending data: {e}")
        return False
```

```
# Main loop
try:
    while True:
        try:
            temperature = dht.temperature
            humidity = dht.humidity

            timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
            print(f"\n[{timestamp}]")
            print(f"Suhu: {temperature:.1f}°C")
            print(f"Kelembaban: {humidity:.1f}%")

            # Kirim ke server
            send_to_server(temperature, humidity)

        except RuntimeError as e:
            print(f"✗ Reading error: {e}")

        time.sleep(SEND_INTERVAL)

except KeyboardInterrupt:
    print("\n\nStopping...")
    dht.exit()
    print("✓ Stopped gracefully")
```

BAB 4: KESIMPULAN DAN SARAN

4.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan pengujian sistem yang telah dilakukan, maka dapat ditarik beberapa kesimpulan untuk menjawab rumusan masalah, yaitu:

1. Integrasi IoT dan Server: Arsitektur sistem yang dirancang berhasil menghubungkan sensor DHT11/22 dengan server berbasis Node.js. Protokol HTTP POST terbukti efektif untuk mengirimkan payload data JSON dari mikrokontroler ke server dengan latensi yang dapat diterima untuk kebutuhan monitoring.
2. Kinerja Backend Native: Implementasi Backend API menggunakan Node.js dengan driver mysql2 (Native) memberikan kontrol penuh terhadap eksekusi kueri. Sistem mampu menangani operasi Create (Insert Log) dan Read (View Dashboard) tanpa kendala overhead yang biasanya muncul pada penggunaan ORM, sehingga performa aplikasi lebih ringan.
3. Struktur Data: Perancangan ERD (Entity Relationship Diagram) dengan tiga entitas utama (users, devices, logs) telah mampu mengakomodasi kebutuhan penyimpanan data historis suhu dan kelembaban. Relasi antar tabel menjamin integritas data, di mana setiap log sensor tervalidasi kepemilikannya berdasarkan ID perangkat.
4. Dokumentasi Sistem: Penggunaan 7 diagram UML (Use Case, Activity, Sequence, Class, dll.) berhasil memvisualisasikan alur kerja sistem secara detail. Diagram

tersebut memudahkan proses pengembangan (coding) karena logika bisnis dan struktur kelas telah terdefinisi dengan jelas sebelum tahap implementasi.

4.2 Saran

Demi pengembangan sistem yang lebih baik di masa mendatang, penulis memberikan beberapa saran sebagai berikut:

1. Keamanan Data: Disarankan untuk menambahkan lapisan keamanan enkripsi (seperti HTTPS/SSL) pada jalur komunikasi data dan menerapkan mekanisme JWT (JSON Web Token) yang lebih kompleks untuk autentikasi perangkat.
2. Fitur Notifikasi: Sistem dapat dikembangkan dengan menambahkan fitur peringatan dini (Early Warning System) berupa notifikasi via WhatsApp atau Email jika suhu/kelembaban melewati ambang batas tertentu.
3. Implementasi Protokol MQTT: Untuk efisiensi daya dan bandwidth yang lebih baik pada skala perangkat yang masif, protokol komunikasi dapat dipertimbangkan untuk beralih dari HTTP REST ke MQTT (Message Queuing Telemetry Transport).
4. Visualisasi Data Lanjutan: Dashboard monitoring dapat diperkaya dengan fitur ekspor data (PDF/Excel) dan grafik analisis tren bulanan/tahunan untuk membantu pengambilan keputusan pengguna.

DAFTAR PUSTAKA

1. Pressman, R. S., & Maxim, B. R. (2020). *Software Engineering: A Practitioner's Approach* (9th ed.). New York: McGraw-Hill Education.
2. Sommerville, I. (2016). *Software Engineering* (10th ed.). London: Pearson Education.
3. Connolly, T., & Begg, C. (2015). Database Systems: A Practical Approach to Design, Implementation, and Management (6th ed.). Boston: Pearson.
4. Kurniawan, A. (2018). Node.js & Socket.io: Membangun Aplikasi Web Real-Time. Yogyakarta: Lokomedia.
5. Espressif Systems. (2024). *ESP32 Technical Reference Manual*. Tersedia di: **https://www.espressif.com**.

6. Node.js Foundation. (2024). *Node.js Documentation*. Tersedia di: **https://nodejs.org/en/docs/**.
7. MySQL Documentation. (2024). *MySQL 8.0 Reference Manual*. Tersedia di: **https://dev.mysql.com/doc/**.
8. Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015). "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications". *IEEE Communications Surveys & Tutorials*, 17(4), 2347-2376.
9. Dennis, A., Wixom, B. H., & Tegarden, D. (2015). *Systems Analysis and Design: An Object-Oriented Approach with UML* (5th ed.). Wiley.
10. Fowler, M. (2004). UML Distilled: A Brief Guide to the Standard Object Modeling Language. Addison-Wesley Professional.
11. Pratama, I. P. A. E. (2014). *Sistem Informasi dan Implementasinya*. Bandung: Informatika.
12. Sidik, B. (2017). *Pemrograman Web dengan Node.js*. Bandung: Informatika.
13. Enterprise, J. (2016). *Belajar Python untuk Pemula*. Jakarta: Elex Media Komputindo. (Referensi untuk script Raspberry Pi/ESP).
14. W. W. W. H. W. (2019). *Designing the Internet of Things*. John Wiley & Sons.
15. Fielding, R. T., & Taylor, R. N. (2002). "Principled Design of the Modern Web Architecture". *ACM Transactions on Internet Technology (TOIT)*, 2(2), 115-150. (Referensi Teori REST API).
16. Dokumen Spesifikasi Sensor DHT11/DHT22. *Aosong Electronics Co., Ltd*.

Lampiran

Arsitektur Utama

https://github.com/PAW-KELOMPOK1-DHT/DHT22-KELOMPOK-1/blob/01c048febcc7cfbf3ed1ad47e2239932d42cf38c/Gambar%20Diagram/ARSITEKTUR%20DIAGRAM%20KEL%201.png

Use Case

https://github.com/PAW-KELOMPOK1-DHT/DHT22-KELOMPOK-1/blob/01c048febcc7cfbf3ed1ad47e2239932d42cf38c/Gambar%20Diagram/USE%20CASE%20DIAGRAM%20KEL%201.png

Activity Diagram

https://github.com/PAW-KELOMPOK1-DHT/DHT22-KELOMPOK-1/blob/af04da698e64fa4e5bd1dd988e4afda050170215/Gambar%20Diagram/ACTIVITY%20DIAGRAM%20KEL%201.png

Flowchart Logika Pemrograman

https://github.com/PAW-KELOMPOK1-DHT/DHT22-KELOMPOK-1/blob/af04da698e64fa4e5bd1dd988e4afda050170215/Gambar%20Diagram/FLOWCHART%20SYSTEM%20KEL%201.png

Sequence Diagram

https://github.com/PAW-KELOMPOK1-DHT/DHT22-KELOMPOK-1/blob/af04da698e64fa4e5bd1dd988e4afda050170215/Gambar%20Diagram/SEQUENCE%20DIAGRAM%20KEL%201.png

Class Diagram

https://github.com/PAW-KELOMPOK1-DHT/DHT22-KELOMPOK-1/blob/af04da698e64fa4e5bd1dd988e4afda050170215/Gambar%20Diagram/CLASS%20DIAGRAM.PNG

Entity Relationship

https://github.com/PAW-KELOMPOK1-DHT/DHT22-KELOMPOK-1/blob/af04da698e64fa4e5bd1dd988e4afda050170215/Gambar%20Diagram/ERD.PNG