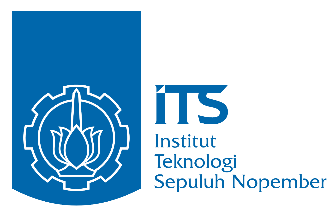
HALAMAN

JUDUL



**INTERKONEKSI SISTEM INSTRUMENTASI-VI231418**

**Sistem Monitoring Kelembaban dan Suhu Gudang Beras Berbasis Sensor IoT dan Teknologi Blockchain untuk Menjamin Kualitas dalam Rantai Pasok Pangan**

**KANIA NAYAKA UTAMAYSARAH (2042231033)**

**ALAMSYAH PRADANA PUTRA (2042231053)**

**WILDAN RIZKI AUZAY (2042231061)**

Dosen Pengampu

**Ahmad Radhy, S.Si., M.Si.**

NPP. 1983201711054

**Program Studi Rekayasa Teknologi Instrumentasi**

Departemen Teknik Instrumentasi

Fakultas Vokasi

Institut Teknologi Sepuluh Nopember

Surabaya

2025



**INTERKONEKSI SISTEM INSTRUMENTASI-VI231418**

**Sistem Monitoring Kelembaban dan Suhu Gudang Beras Berbasis Sensor IoT dan Teknologi Blockchain untuk Menjamin Kualitas dalam Rantai Pasok Pangan**

**KANIA NAYAKA UTAMAYSARAH (2042231033)**

**ALAMSYAH PRADANA PUTRA (2042231053)**

**WILDAN RIZKI AUZAY (2042231061)**

Dosen Pengampu

**Ahmad Radhy, S.Si., M.Si.**

NPP. 1983101711054

**Program Studi** **D4 Rekayasa Teknologi Instrumentasi**

Departemen Teknik Instrumentasi

Fakultas Vokasi

Institut Teknologi Sepuluh Nopember

Surabaya

2025

**KATA PENGANTAR**

**ABSTRAK**

**DAFTAR ISI**

**DAFTAR GAMBAR**

**DAFTAR TABEL**

**BAB I**

**PENDAHULUAN**

* 1. **Latar Belakang**

Beras merupakan komoditas pangan strategis dan bahan pokok utama di Indonesia yang berperan penting dalam ketahanan pangan nasional. Namun, persoalan pasca-panen, khususnya pada fase penyimpanan di gudang, sering kali luput dari perhatian. Kerusakan beras yang terjadi di tahap penyimpanan dapat disebabkan oleh kondisi suhu dan kelembaban yang tidak sesuai standar, yang menyebabkan berkembangnya mikroorganisme, serangan hama, serta penurunan kualitas fisik dan nutrisi beras (Suliantoro et al., 2021).

Menurut data Badan Pangan Nasional (Bapanas), lebih dari 5% hasil panen beras mengalami kehilangan pascapanen (postharvest loss), sebagian besar disebabkan oleh sistem penyimpanan yang tidak memadai (*Tempo*, 2023). Di sisi lain, Perum Bulog sebagai penyedia cadangan pangan nasional menghadapi tantangan dalam memastikan kualitas beras di lebih dari 1.500 gudang miliknya di seluruh Indonesia (CNN Indonesia, 2024). Hal ini menunjukkan urgensi pengembangan sistem monitoring penyimpanan yang dapat bekerja secara real-time, presisi, dan akuntabel.

Untuk menjaga kualitas beras, dua parameter lingkungan utama yang harus dimonitor secara ketat adalah suhu dan kelembaban relatif udara. Menurut FAO (2019), suhu gudang ideal adalah di bawah 25°C, dengan kisaran optimal 15–25°C. Sementara itu, kelembaban relatif yang aman untuk penyimpanan beras berada pada kisaran maksimum 65–70%. Suhu di atas 30°C berpotensi mempercepat perkembangan hama gudang seperti *Sitophilus oryzae*, sedangkan kelembaban di atas 75% akan mendorong pertumbuhan jamur dan meningkatkan risiko kontaminasi aflatoksin (Prasetyo & Darmawan, 2021).

Teknologi Internet of Things (IoT) menjadi solusi potensial untuk menghadirkan sistem monitoring suhu dan kelembaban secara otomatis dan berkelanjutan. Sensor industri dengan output komunikasi Modbus RTU atau TCP/IP dapat langsung terhubung ke komputer server untuk membaca kondisi lingkungan secara akurat. Data ini kemudian disimpan dalam *time-series database* seperti InfluxDB dan divisualisasikan secara real-time melalui platform seperti Grafana, sehingga memudahkan pengelola gudang dalam mengambil keputusan preventif (Siregar & Putra, 2022).

Namun demikian, penyimpanan data secara lokal dan terpusat masih menyisakan potensi risiko manipulasi atau hilangnya data. Oleh karena itu, diperlukan integrasi sistem monitoring ini dengan teknologi blockchain. Dengan mencatat *hash* data suhu dan kelembaban ke dalam blockchain publik seperti Ethereum, data lingkungan dalam gudang dapat menjadi immutable dan transparan, serta dapat diaudit oleh seluruh pihak yang berkepentingan dalam rantai pasok—mulai dari petani, distributor, pemerintah, hingga konsumen akhir (Tian, 2016).

Lebih jauh lagi, pendekatan ini sejalan dengan inisiatif pemerintah dalam mendorong digitalisasi pertanian berbasis *smart farming* dan *Agriculture 4.0*, di mana teknologi informasi digunakan untuk meningkatkan efisiensi, ketertelusuran, dan kualitas produk pertanian (Kementerian Pertanian RI, 2023). Dengan mengintegrasikan instrumentasi berbasis sensor, jaringan komunikasi, visualisasi data, dan Web3 (decentralized application), proyek ini diharapkan dapat menjadi prototipe sistem cerdas yang mampu menjawab tantangan nyata dalam pengelolaan kualitas pangan nasional.

* 1. **Rumusan Masalah**

Berdasarkan latar belakang yang telah diuraikan, proyek ini berangkat dari permasalahan utama dalam pengelolaan gudang beras yang masih belum optimal, khususnya dalam hal pemantauan kondisi lingkungan dan transparansi data. Adapun rumusan masalah yang dapat dirumuskan adalah sebagai berikut:

1. Bagaimana merancang sistem monitoring suhu dan kelembaban yang mampu memantau kondisi gudang beras secara real-time dan berkelanjutan menggunakan sensor berbasis IoT?
2. Bagaimana cara menyimpan dan memvisualisasikan data suhu dan kelembaban secara efisien dan akurat agar dapat membantu pengambilan keputusan oleh pengelola gudang?
3. Bagaimana mengintegrasikan sistem monitoring tersebut dengan teknologi blockchain agar data yang tercatat dapat dijamin keasliannya (immutable), transparan, dan dapat diverifikasi oleh pihak-pihak dalam rantai pasok?
4. Bagaimana merancang antarmuka pengguna berbasis Web3 (DApp) yang memungkinkan stakeholder seperti petani, distributor, atau pemerintah untuk mengakses histori kondisi gudang secara aman dan desentralistik?
   1. **Tujuan Penelitian**
5. Merancang sistem monitoring suhu dan kelembaban gudang beras dengan memanfaatkan sensor industri yang terhubung langsung ke komputer server melalui protokol komunikasi Modbus RTU atau TCP/IP, untuk memperoleh data lingkungan secara berkala dan akurat.
6. Mengembangkan sistem penyimpanan dan visualisasi data berbasis InfluxDB dan Grafana, yang mampu menampilkan informasi suhu dan kelembaban secara time-series dan real-time untuk keperluan pemantauan dan pengambilan keputusan.
7. Mengintegrasikan sistem monitoring dengan teknologi blockchain publik, dengan cara mencatat hash data lingkungan pada interval waktu tertentu ke dalam smart contract berbasis Ethereum, guna menjamin transparansi dan integritas data dalam rantai pasok beras.
8. Membangun aplikasi antarmuka pengguna berbasis Web3 (DApp) yang memungkinkan stakeholder seperti pemilik gudang, distributor, dan regulator untuk memverifikasi kondisi lingkungan penyimpanan secara terdesentralisasi dan dapat diakses publik.
   1. **Manfaat Penelitian**
9. Manfaat Akademik
   1. Memberikan kontribusi dalam pengembangan ilmu di bidang interkoneksi sistem instrumentasi, khususnya dalam integrasi antara sistem monitoring lingkungan, database time-series, dan teknologi blockchain.
   2. Menjadi contoh penerapan nyata konsep transparansi data dan traceability dalam rantai pasok berbasis teknologi Web3, yang dapat menjadi referensi untuk penelitian sejenis di bidang pertanian dan pangan.
   3. Meningkatkan pemahaman mahasiswa terhadap penggunaan protokol komunikasi industri (seperti Modbus RTU/TCP), pengolahan data sensor, dan integrasi dengan sistem visualisasi serta smart contract.
10. Manfaat Praktis
    1. Menyediakan solusi sistematis untuk monitoring suhu dan kelembaban gudang beras secara real-time tanpa ketergantungan pada perangkat mikrokontroler, memanfaatkan infrastruktur industri yang telah ada.
    2. Meningkatkan kualitas penyimpanan beras dengan menghadirkan sistem peringatan dini (early warning) dan dokumentasi lingkungan yang dapat digunakan sebagai bukti kepatuhan terhadap standar penyimpanan pangan.
    3. Mendukung kepercayaan antar pihak dalam rantai pasok, seperti petani, distributor, dan pemerintah, melalui pencatatan hash data ke blockchain yang tidak dapat dimanipulasi dan dapat diaudit publik.
11. Manfaat Sosial dan Ekonomi
    1. Membantu mengurangi kerugian pascapanen akibat penyimpanan yang tidak sesuai standar, sehingga berdampak pada peningkatan daya saing petani dan UMKM pangan.
    2. Menunjang program pemerintah dalam transformasi digital sektor pertanian dan pangan, sekaligus membuka peluang pengembangan model bisnis berbasis transparansi dan akuntabilitas data.
    3. Mendorong penggunaan teknologi decentralized ledger dan open-source monitoring di sektor-sektor kritis, sebagai bagian dari adopsi Industri 4.0 dan Agriculture 4.0 di Indonesia.
    4. **Batasan Penelitian**
12. Sistem hanya menggunakan sensor industri yang mendukung protokol Modbus RTU atau Modbus TCP/IP, yang terhubung langsung ke komputer server atau gateway. Proyek ini tidak menggunakan mikrokontroler (seperti ESP32, Arduino, atau STM32) dalam tahap akuisisi data
13. Parameter lingkungan yang dimonitor terbatas pada suhu dan kelembaban relatif udara di dalam ruang penyimpanan gudang beras. Parameter lain seperti kadar gas, tekanan, atau intensitas cahaya tidak termasuk dalam cakupan sistem ini.
14. Sistem hanya diterapkan dan diuji coba pada simulasi ruang penyimpanan berskala kecil atau skenario gudang tunggal. Pengujian multi-lokasi atau integrasi antar gudang belum menjadi fokus penelitian ini.
15. Visualisasi data dilakukan melalui dashboard Grafana yang terhubung ke database InfluxDB. Sistem tidak menyediakan fitur mobile app atau dashboard berbasis cloud dalam lingkup proyek ini.
16. Integrasi dengan blockchain dibatasi pada pencatatan hash data suhu dan kelembaban dalam interval waktu yang pendek ke dalam smart contract berbasis Ethereum. Proyek ini tidak mencatat seluruh data sensor ke blockchain secara langsung, guna menghindari beban transaksi dan biaya gas tinggi.

**BAB II**

**TINJAUAN PUSTAKA**

1. ***State of Art***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Judul Penelitian** | **Penulis (Tahun)** | **Metode yang Digunakan** | **Hasil yang Didapatkan** |
| 1 | Real-Time Temperature and Humidity Monitoring System for Agricultural Storage | Yasin, Bahar, & Aziz (2020) | Metode eksperimental: perancangan sistem berbasis IoT menggunakan ESP8266, sensor SHT11, serta integrasi ThingSpeak Cloud. | Sistem dapat menampilkan data suhu dan kelembaban secara real-time melalui dashboard daring. |
| 2 | Real-time Environmental Monitoring System for Cold Chain Warehouses Using Modbus-RTU and SCADA | Cheng, Zhang, & Liu (2021) | Metode rekayasa sistem: penerapan sensor Modbus RTU yang terhubung ke PLC, visualisasi melalui SCADA untuk cold storage. | Sistem dapat memantau lingkungan secara akurat namun kurang fleksibel karena tergantung pada hardware PLC. |
| 3 | Development of a Web-Based IoT Platform for Real-Time Environmental Monitoring with Grafana | Hasan & Islam (2020) | Metode prototyping: pengembangan sistem berbasis Raspberry Pi, sensor DHT22, penyimpanan dengan InfluxDB, dan visualisasi Grafana. | Sistem berhasil menyajikan visualisasi suhu dan kelembaban real-time dengan efisiensi dan kemudahan akses. |
| 4 | An Agri-Food Supply Chain Traceability System for China Based on RFID & Blockchain Technology | Tian (2016) | Metode desain sistem: integrasi RFID, sensor lingkungan, dan pencatatan hash ke blockchain melalui smart contract. | Sistem memungkinkan traceability dalam rantai pasok dan menjamin integritas data melalui teknologi blockchain. |

Berdasarkan kajian literatur sebelumnya, telah banyak penelitian yang mengembangkan sistem monitoring suhu dan kelembaban berbasis Internet of Things (IoT), baik dalam konteks gudang pertanian, rumah kaca, maupun cold storage industri. Sebagian besar sistem tersebut menggunakan metode pemantauan berbasis mikrokontroler (seperti ESP8266 atau Raspberry Pi) yang mengirimkan data ke cloud server seperti ThingSpeak atau visualisasi SCADA internal. Selain itu, sistem-sistem tersebut umumnya hanya menitikberatkan pada akuisisi dan visualisasi data secara real-time, tanpa memperhatikan aspek keandalan data jangka panjang, transparansi, dan auditabilitas lintas pemangku kepentingan.

Salah satu kelemahan umum yang teridentifikasi adalah tidak adanya mekanisme untuk menjamin integritas data dari sensor. Dalam banyak kasus, data suhu dan kelembaban hanya tersimpan secara lokal atau cloud tanpa sistem yang mampu menjamin bahwa data tersebut tidak dimanipulasi. Penelitian oleh Tian (2016) menjadi satu-satunya yang mengusulkan integrasi blockchain untuk mendukung traceability data lingkungan, namun implementasinya lebih fokus pada penggunaan RFID dan belum secara spesifik mengadopsi arsitektur time-series database dan visualisasi lokal.

Selain itu, sistem yang menggabungkan penyimpanan time-series (InfluxDB), visualisasi dual-mode (Grafana + Qt), dan pencatatan hashed summary ke dalam smart contract Ethereum, belum ditemukan pada literatur terdahulu. Tidak ada juga sistem yang secara spesifik dirancang untuk monitoring suhu dan kelembaban di gudang beras—sektor penting dalam rantai pasok pangan di negara berkembang.

1. **Pergudangan Beras sebagai Plant Monitoring**

Pergudangan beras merupakan salah satu komponen penting dalam rantai pasok pangan nasional yang berfungsi sebagai tempat penyimpanan hasil panen sebelum didistribusikan ke pasar, distributor, atau konsumen akhir. Kualitas beras selama masa penyimpanan sangat dipengaruhi oleh kondisi lingkungan di dalam gudang, terutama suhu dan kelembaban, yang dapat menyebabkan penurunan mutu fisik, biologis, dan nutrisi apabila tidak dikendalikan secara tepat (FAO, 2019).

Dalam konteks sistem instrumentasi, gudang penyimpanan beras diposisikan sebagai plant—yakni objek fisik yang menjadi fokus pemantauan dan pengendalian sistem. Sebagai plant, gudang memiliki karakteristik lingkungan tertutup, berskala luas, dan menyimpan komoditas pangan dalam jumlah besar yang sensitif terhadap fluktuasi mikroklimat. Oleh karena itu, dibutuhkan sistem monitoring yang mampu melakukan pengukuran secara berkelanjutan, real-time, dan terintegrasi.

Menurut Suliantoro et al. (2021), kelembaban di atas 70% dapat memicu pertumbuhan jamur seperti Aspergillus flavus yang menghasilkan aflatoksin, sementara suhu yang melebihi 30°C akan mempercepat aktivitas hama gudang seperti Sitophilus oryzae. Jika kondisi ini tidak dimonitor, kerusakan produk dapat terjadi dalam waktu singkat.

Dalam proyek ini, pergudangan beras tidak hanya menjadi objek pengukuran, tetapi juga dijadikan sebagai simulasi nyata penerapan teknologi monitoring dan Web3 dalam rantai pasok pertanian. Dengan demikian, pergudangan tidak hanya dilihat sebagai ruang penyimpanan pasif, melainkan sebagai entitas aktif dalam ekosistem digital yang dapat membuktikan bahwa penyimpanan dilakukan sesuai dengan standar mutu nasional dan internasional.

1. **Sistem Monitoring Lingkungan**

Sistem monitoring lingkungan merupakan sistem terintegrasi yang bertujuan untuk mengamati, mencatat, dan menganalisis parameter lingkungan secara berkala dan real-time. Dalam konteks penyimpanan beras, parameter yang paling berpengaruh terhadap stabilitas mutu adalah suhu dan kelembaban relatif udara. Kedua variabel ini berperan langsung dalam mencegah pertumbuhan mikroorganisme, serangan hama, serta proses degradasi yang dapat menyebabkan kehilangan pascapanen (*postharvest loss*).

Berdasarkan laporan FAO (2019), suhu gudang yang ideal untuk penyimpanan beras berkisar antara 15–25°C, sedangkan kelembaban relatif harus dijaga di bawah 70%. Jika kelembaban melebihi ambang tersebut, kondisi lingkungan menjadi optimal untuk pertumbuhan jamur seperti *Aspergillus flavus*, yang menghasilkan aflatoksin berbahaya bagi kesehatan manusia (Tian, 2016).

Oleh karena itu, sistem monitoring lingkungan yang presisi diperlukan agar pengelola gudang dapat mengambil tindakan korektif lebih awal, seperti menyalakan sistem ventilasi, pengering udara, atau melakukan pemindahan stok. Sistem ini juga dapat berfungsi sebagai bukti dokumentasi digital bahwa kondisi penyimpanan telah sesuai standar, terutama jika terintegrasi dengan blockchain.

Pada proyek ini, sistem menggunakan **sensor industri tipe SHT20** atau sensor sejenis yang mendukung **protokol komunikasi Modbus RTU**. Sensor ini berfungsi sebagai komponen utama dalam proses akuisisi data lingkungan. SHT20 merupakan sensor digital yang menggabungkan **pengukuran suhu dan kelembaban** dalam satu chip dan dirancang untuk lingkungan industri dengan **akurasi tinggi** (±0.3°C untuk suhu dan ±3% untuk RH).

Menurut Yasin et al. (2020), penggunaan sensor digital seperti SHT20 atau DHT22 dalam sistem monitoring pangan memberikan keunggulan dalam hal **presisi, daya tahan terhadap lingkungan ekstrem, serta kemampuan integrasi ke sistem SCADA atau IoT**. Sensor ini sangat cocok digunakan untuk kebutuhan industri skala kecil maupun besar karena memiliki **kalibrasi pabrik dan protokol standar industri.**

1. **Komunikasi Data Industri**

Komunikasi data industri merupakan sistem pertukaran informasi antar perangkat di lingkungan otomasi dan kontrol industri. Dalam konteks sistem monitoring lingkungan, komunikasi ini menghubungkan sensor industri dengan sistem pusat seperti server, PLC, atau HMI agar data dapat diolah, disimpan, dan divisualisasikan. Keandalan dan standar protokol komunikasi yang digunakan sangat menentukan kualitas dan akurasi sistem secara keseluruhan (Palau et al., 2021).

Modbus RTU adalah protokol komunikasi serial berbasis RS-485 yang banyak digunakan dalam industri untuk menghubungkan perangkat seperti sensor, aktuator, dan kontroler dalam topologi *master-slave*. Dalam protokol ini, sensor bertindak sebagai slave yang merespons permintaan dari master (komputer/server atau gateway), dan pertukaran data terjadi dalam format biner yang efisien.

Menurut Kumar et al. (2020), Modbus RTU tetap menjadi standar komunikasi penting dalam sistem monitoring karena konsumsi bandwidth-nya rendah, cocok untuk implementasi real-time, dan mudah dikonversi ke protokol lain menggunakan converter/bridge.

Modbus TCP/IP adalah versi berbasis jaringan Ethernet dari protokol Modbus. Protokol ini bekerja dengan mengemas data Modbus RTU ke dalam format TCP/IP dan memungkinkan pengiriman data melalui jaringan LAN atau WAN. Ini sangat cocok untuk sistem modern yang terhubung ke jaringan komputer atau cloud.

Menurut Palau et al. (2021), Modbus TCP/IP sangat sesuai untuk sistem instrumentasi cerdas karena mudah diintegrasikan dengan *software stack* modern dan memiliki skalabilitas tinggi untuk sistem multisensor. Dalam proyek ini, jika sensor atau gateway sudah mendukung komunikasi Ethernet, maka Modbus TCP/IP digunakan untuk menghubungkan sensor ke program TCP Server yang berjalan di komputer. Program ini kemudian menyimpan data ke InfluxDB, dan mencatat *hash* ke blockchain.

1. **InfluxDB**

InfluxDB adalah database time-series (TSDB) open-source yang dirancang khusus untuk menangani data yang berkaitan dengan waktu (*timestamped data*), seperti suhu, kelembaban, tegangan, dan tekanan. Berbeda dengan database relasional (SQL), InfluxDB dioptimalkan untuk kecepatan penulisan (*write performance*), efisiensi penyimpanan data sensor secara kontinu, serta analisis data berbasis waktu secara real-time (Barksdale & Stone, 2019).

Dalam sistem monitoring lingkungan, InfluxDB berperan penting sebagai media penyimpanan data suhu dan kelembaban yang diperoleh dari sensor. Setiap data dicatat bersama dengan waktu pengambilan data, sehingga memungkinkan pengguna untuk menganalisis perubahan kondisi lingkungan secara historis maupun real-time.

InfluxDB juga mendukung pengaturan retention policy, yaitu kebijakan penyimpanan data agar otomatis terhapus setelah waktu tertentu, dan continuous query yang dapat digunakan untuk membuat ringkasan data seperti rata-rata suhu per 6 jam. Ringkasan ini bermanfaat sebagai sumber hash yang akan dicatat ke blockchain untuk menjamin integritas data.

Menurut Srinivasan et al. (2022), arsitektur InfluxDB sangat cocok digunakan dalam sistem monitoring industri karena mampu menyimpan jutaan entri sensor dengan efisien dan tetap mendukung integrasi dengan platform visualisasi dan blockchain.

1. **Visualisasi Data dengan Grafana**

Grafana adalah sebuah platform open-source untuk visualisasi dan analisis data yang dirancang khusus untuk data time-series. Dengan Grafana, pengguna dapat membuat dashboard interaktif, menyusun grafik, dan memantau data sensor secara real-time dari berbagai sumber seperti InfluxDB, Prometheus, MySQL, atau API eksternal (Gupta & Singh, 2020).

Dalam sistem monitoring, visualisasi berperan penting dalam membantu pengguna memahami pola data dan membuat keputusan secara cepat. Grafana memungkinkan penyajian informasi yang kompleks menjadi bentuk grafik yang mudah dipahami melalui widget seperti line chart, bar graph, gauge, table, dan alert panel.

Menurut Gupta dan Singh (2020), Grafana menjadi pilihan utama dalam proyek-proyek pemantauan berbasis IoT karena antarmukanya yang mudah digunakan, dukungan terhadap banyak jenis data, serta kemampuannya dalam memberikan notifikasi otomatis yang kritis terhadap sistem yang dimonitor. Babu & Rao (2022) menunjukkan bahwa sistem pemantauan suhu dan kelembaban berbasis InfluxDB dan Grafana memiliki akurasi visual tinggi dan dapat diimplementasikan pada berbagai sektor penyimpanan, termasuk logistik dan pertanian. Sementara itu, Singh et al. (2021) menegaskan bahwa Grafana sangat cocok digunakan untuk smart agriculture karena kemampuannya dalam mengelola data yang fluktuatif dan dinamis seperti suhu gudang.

1. **Qt sebagai Antarmuka Monitoring Lokal**

Qt adalah framework pengembangan aplikasi cross-platform berbasis GUI (Graphical User Interface) yang digunakan secara luas untuk membangun aplikasi desktop interaktif dan responsif. Qt ditulis dengan bahasa C++ dan mendukung berbagai platform seperti Windows, Linux, dan macOS. Framework ini juga menyediakan binding untuk bahasa lain seperti Python melalui PyQt atau PySide (Blanchette & Summerfield, 2006).

Dalam konteks sistem monitoring lingkungan, Qt digunakan untuk membangun aplikasi desktop lokal yang menampilkan data suhu dan kelembaban dari sensor secara real-time, sebagai alternatif atau pelengkap dari dashboard berbasis web seperti Grafana. Antarmuka berbasis Qt memungkinkan pengguna melihat data secara langsung di komputer lokal tanpa perlu koneksi internet, sehingga cocok digunakan untuk pengawasan internal gudang atau laboratorium penyimpanan. Dalam proyek ini, Qt digunakan untuk:

1. Menampilkan data suhu dan kelembaban secara lokal melalui tampilan grafik, tabel, atau indikator digital.
2. Menghubungkan aplikasi desktop dengan database InfluxDB atau langsung membaca data dari program TCP Server.
3. Menyediakan user interface (UI) yang intuitif dan ringan untuk pengguna yang tidak membutuhkan akses blockchain atau web.

Qt juga mendukung fitur pemrograman event-driven dan pemrosesan paralel, yang membuatnya ideal untuk menangani aliran data sensor secara kontinu dengan antarmuka yang tetap responsif. Selain itu, dengan adanya Qt Designer, pengembangan tampilan antarmuka menjadi lebih mudah tanpa harus menulis kode UI secara manual.

1. **Tekologi Blockchain dalam Rantai Pasok**

Blockchain merupakan teknologi penyimpanan data terdistribusi yang memungkinkan setiap transaksi atau data yang dicatat bersifat immutabel (tidak dapat diubah), terdesentralisasi, dan terverifikasi oleh banyak pihak secara independen. Teknologi ini bekerja dengan prinsip ledger digital bersama, di mana setiap blok berisi informasi (data) yang terhubung secara kriptografis ke blok sebelumnya dalam sebuah rantai, sehingga membentuk rekam jejak yang tidak dapat dimanipulasi (Tian, 2016).

Dalam konteks sistem informasi, blockchain tidak hanya digunakan untuk mata uang kripto seperti Bitcoin atau Ethereum, tetapi juga telah berkembang luas di bidang logistik, pertanian, keuangan, dan kesehatan, terutama untuk mendukung prinsip transparansi, akuntabilitas, dan traceability dalam rantai pasok (Casino et al., 2019).

Berikut adalah karakteristik utama teknologi blockchain yang menjadikannya relevan dalam sistem rantai pasok pangan:

1. Decentralized: Tidak ada otoritas tunggal yang mengontrol sistem; semua node saling berbagi dan memverifikasi data.
2. Immutable: Setelah data tercatat dalam blok dan tervalidasi, data tidak dapat diubah atau dihapus.
3. Transparency: Seluruh pihak yang terhubung dapat melihat riwayat data yang tercatat.
4. Security: Setiap blok terlindungi oleh algoritma hash dan kriptografi yang kompleks.
5. Auditability: Seluruh proses dapat ditelusuri kembali ke waktu dan tempat kejadian dengan bukti kriptografis.

Menurut Lin et al. (2020), keunggulan blockchain dalam memastikan integritas dan keaslian data menjadikannya ideal untuk sektor pertanian, khususnya dalam pelacakan kualitas produk dan dokumentasi kondisi lingkungan selama penyimpanan atau pengiriman.

Dalam rantai pasok pangan seperti beras, kondisi penyimpanan (suhu dan kelembaban) sangat memengaruhi kualitas produk. Namun, sistem monitoring konvensional sering kali tidak menyediakan **bukti otentik** bahwa kondisi penyimpanan telah memenuhi standar. Tian (2016) menyatakan bahwa traceability berbasis blockchain dalam sektor agri-food memberikan pengaruh signifikan terhadap **keamanan pangan, pengendalian mutu, dan efisiensi logistik.** Menurut Altmeyer & Konietzko (2018), penggunaan Qt dalam sistem SCADA dan monitoring industri terbukti efektif dalam menyediakan antarmuka pengguna yang stabil dan mudah digunakan untuk operasi lapangan.

1. **Decentralized Application (DApp) dan Web3**

Decentralized Application (DApp) adalah jenis aplikasi digital yang berjalan di atas jaringan blockchain publik dan tidak bergantung pada server pusat. DApp memanfaatkan kontrak pintar (smart contract) sebagai mesin logika yang berjalan secara otomatis dan dapat diverifikasi oleh semua pihak dalam jaringan. DApp biasanya memiliki frontend seperti aplikasi web pada umumnya, tetapi backend-nya berjalan di blockchain dan dikontrol oleh kode sumber terbuka, bukan perusahaan terpusat (Zheng et al., 2020).

Web3, di sisi lain, merupakan konsep generasi ketiga dari teknologi web yang memperkenalkan interaksi langsung antara pengguna dan blockchain melalui protokol seperti Web3.js atau Ethers.js. Web3 memberi kendali penuh kepada pengguna atas data dan aset digital mereka, menjadikannya sangat cocok untuk aplikasi yang menuntut transparansi, keamanan, dan desentralisasi (Buterin, 2020).

Dalam proyek sistem monitoring suhu dan kelembaban gudang beras ini, DApp dan Web3 digunakan sebagai antarmuka verifikasi publik untuk stakeholder (seperti distributor, pengelola gudang, dan konsumen) agar dapat:

* Melihat hash data lingkungan yang tercatat ke dalam blockchain.
* Mengecek keaslian dan integritas data suhu dan kelembaban pada waktu tertentu.
* Memastikan bahwa produk disimpan sesuai standar sebelum didistribusikan.

Pengguna dapat mengakses DApp melalui browser yang mendukung ekstensi wallet (seperti MetaMask), dan melihat **log hash data** beserta timestamp-nya. Ini memberikan **akses publik terhadap bukti penyimpanan** yang telah dilakukan sesuai standar, tanpa harus membuka data sensor mentah.

Menurut Xie et al. (2019), DApp memberikan kemampuan unik dalam menyediakan kepercayaan tanpa pihak ketiga, yang sangat dibutuhkan dalam skenario rantai pasok global yang kompleks. Dalam konteks pertanian, ini berarti kepercayaan terhadap kualitas produk dapat dibangun langsung antara produsen dan konsumen akhir.

**BAB III**

**METODOLOGI PENELITIAN**

1. **Metode Penelitian**

Penelitian ini menggunakan pendekatan penelitian terapan (applied research) dengan metode perancangan sistem prototipe yang bertujuan untuk mengembangkan solusi teknologi yang dapat diimplementasikan secara nyata dalam konteks monitoring lingkungan gudang penyimpanan beras. Fokus utama penelitian ini adalah membangun sistem yang mampu memantau suhu dan kelembaban secara real-time, menyimpan data dengan efisien, menampilkannya melalui dashboard visual, dan mengintegrasikan transparansi data melalui pencatatan hash ke blockchain.

Metode yang digunakan menggabungkan tahapan dari System Development Life Cycle (SDLC) model prototyping, yang memungkinkan iterasi cepat dari pengembangan sistem, evaluasi fungsional, serta validasi terhadap kebutuhan pengguna.

Alur pengerjaan proyek ini disusun menjadi beberapa tahap sebagai berikut:

1. Studi Literatur dan Analisis Kebutuhan

Tahap pertama dari penelitian ini dimulai dengan studi literatur dan analisis kebutuhan, yaitu mengkaji berbagai referensi akademik dan standar industri terkait penyimpanan beras, suhu dan kelembaban optimal, serta penerapan teknologi seperti blockchain, time-series database, dan visualisasi data. Analisis ini bertujuan untuk merumuskan spesifikasi sistem yang akan dikembangkan.

1. Perancangan Arsitektur Sistem

Selanjutnya dilakukan perancangan arsitektur sistem, yang mencakup penyusunan alur data dari sensor ke server, pengolahan dan penyimpanan data, hingga pencatatan hash ke blockchain. Arsitektur ini dirancang agar mendukung aliran data secara berkesinambungan dan dapat terintegrasi dengan teknologi Web3 secara modular.

1. Implementasi Sistem Monitoring

Tahap ketiga adalah **implementasi sistem monitoring**, di mana sensor industri yang terhubung melalui protokol Modbus mengirimkan data ke program TCP Server. Program ini berfungsi sebagai penghubung antara perangkat fisik dan sistem penyimpanan berbasis InfluxDB.

1. Visualisasi Data

Setelah sistem penyimpanan aktif, dilakukan pembangunan dashboard visualisasi menggunakan Grafana. Dashboard ini dirancang untuk menampilkan suhu dan kelembaban secara real-time dan historis, dengan fitur pemantauan ambang batas serta ringkasan data periodik.

1. Integrasi Blockchain dan DApp

Tahap berikutnya adalah integrasi blockchain dan pengembangan DApp, yaitu mencatat *hash* data suhu dan kelembaban ke dalam smart contract di jaringan Ethereum. DApp yang dikembangkan berfungsi sebagai antarmuka publik untuk memverifikasi keaslian data berdasarkan *hash* tersebut.

1. Pengujian dan Evaluasi Sistem

Setelah semua komponen berfungsi, dilakukan pengujian dan evaluasi sistem untuk memastikan setiap modul berjalan sesuai fungsinya. Pengujian ini meliputi uji fungsional TCP Server, akurasi visualisasi, pencatatan *hash* ke blockchain, serta pengujian antarmuka pengguna pada DApp.

1. Dokumentasi dan Pelaporan

Terakhir, hasil sistem yang telah diuji didokumentasikan dalam bentuk laporan proyek, yang mencakup analisis keberhasilan sistem, pembahasan hasil, serta saran pengembangan lebih lanjut untuk implementasi skala lebih besar atau integrasi multi-lokasi.

1. **Alat dan Bahan**

Dalam implementasi sistem monitoring suhu dan kelembaban untuk gudang beras ini, digunakan kombinasi perangkat keras dan perangkat lunak yang membentuk satu kesatuan sistem monitoring real-time berbasis jaringan dan blockchain. Setiap alat dan bahan dipilih berdasarkan fungsinya dalam mendukung akuisisi data, penyimpanan, visualisasi, serta transparansi informasi.

1. Perangkat Keras

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Alat** | **Spesifikasi / Fungsi** |
| 1 | **Sensor Suhu & Kelembaban** | Sensor industri tipe SHT20 atau sejenis yang mendukung protokol Modbus RTU. Digunakan untuk membaca kondisi lingkungan gudang secara kontinu. |
| 2 | **Converter RS485 to USB** | Digunakan untuk menghubungkan sensor Modbus RTU ke komputer server melalui port USB. |
| 3 | **Komputer Server / Laptop** | Minimal prosesor Intel Core i3, RAM 4 GB. Digunakan sebagai host untuk aplikasi monitoring, database, server TCP, visualisasi, dan aplikasi blockchain. |
| 4 | **Jaringan LAN / Internet (opsional)** | Digunakan jika komunikasi menggunakan Modbus TCP/IP atau untuk koneksi ke jaringan Ethereum saat mencatat hash ke blockchain. |
| 5 | **Power Supply Sensor** | Menyediakan suplai daya stabil untuk sensor agar dapat beroperasi secara terus-menerus. |

1. Perangkat Lunak

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Perangkat Lunak** | **Deskripsi / Fungsi** |
| 1 | **Rust Programming Language** | Digunakan untuk membuat program TCP Server yang menerima data dari sensor dan menyimpannya ke InfluxDB. |
| 2 | **InfluxDB** | Database time-series untuk menyimpan data suhu dan kelembaban berdasarkan timestamp. |
| 3 | **Grafana** | Visualisasi berbasis web yang menampilkan data real-time dan historis dari InfluxDB dalam bentuk grafik dan indikator. |
| 4 | **Qt (Qt5/Qt6)** | Framework GUI untuk membangun **aplikasi desktop lokal** guna memantau suhu dan kelembaban secara real-time, tanpa koneksi internet. |
| 5 | **Solidity** | Digunakan untuk menulis smart contract dalam pencatatan hash data ke jaringan blockchain Ethereum. |
| 6 | **Remix IDE** | Editor berbasis web untuk menulis, menguji, dan melakukan deploy smart contract ke Ethereum. |
| 7 | **MetaMask** | Wallet digital untuk mengelola akun Ethereum dan berinteraksi dengan DApp dari browser. |
| 8 | **Web3.js / Ethers.js** | Library JavaScript untuk menghubungkan frontend DApp ke smart contract di blockchain. |
| 9 | **Node.js** | Environment untuk menjalankan aplikasi DApp dan server lokal frontend. |
| 10 | **Visual Studio Code / Qt Creator** | Digunakan sebagai IDE untuk menulis kode Rust, Solidity, JavaScript, dan aplikasi GUI Qt. |
| 11 | **Git & GitHub (opsional)** | Untuk manajemen versi kode dan kolaborasi pengembangan tim. |

Dengan dukungan alat dan bahan di atas, sistem dapat dibangun secara end-to-end mulai dari pengambilan data sensor, penyimpanan dan visualisasi lokal menggunakan Qt, pelaporan berbasis web melalui Grafana, hingga pencatatan bukti data ke blockchain. Penggunaan Qt dalam sistem ini melengkapi kebutuhan pengawasan di lingkungan lokal yang tidak selalu terkoneksi internet, sedangkan integrasi Web3 mendukung transparansi publik dalam rantai pasok beras.

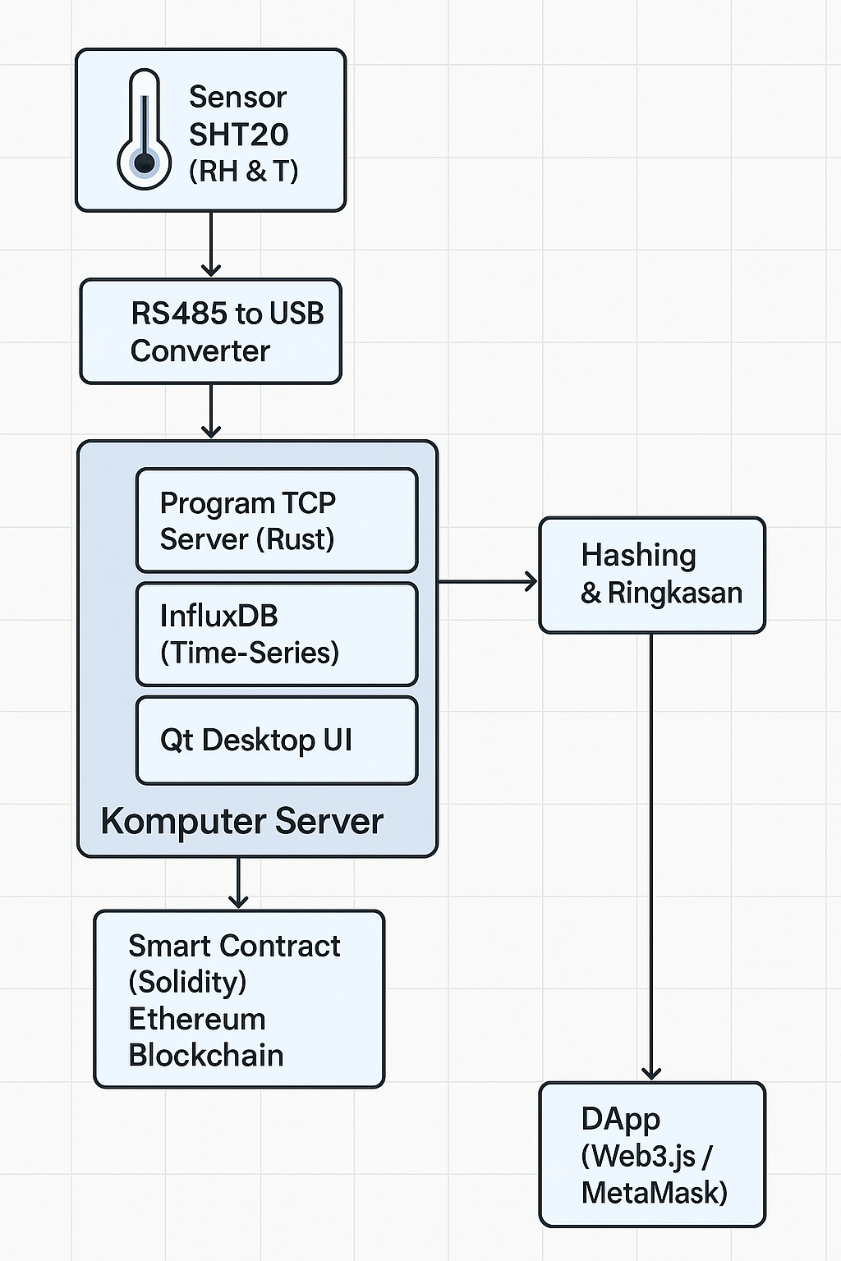
1. **Arsitektur Sistem**

Arsitektur sistem merupakan rancangan struktural dari seluruh komponen yang terlibat dalam proyek, mulai dari perangkat input (sensor), perangkat pemrosesan data, sistem penyimpanan, hingga antarmuka pengguna dan integrasi blockchain. Arsitektur ini dirancang secara modular dan terintegrasi, sehingga memungkinkan sistem bekerja secara real-time, fleksibel, dan transparan.

Tujuan dari Arsitektur Sistem:

1. Memastikan data suhu dan kelembaban dapat diakuisisi, disimpan, ditampilkan, dan diverifikasi.
2. Menghubungkan sistem monitoring lokal (Qt) dengan sistem visualisasi web (Grafana).
3. Menjamin integritas data melalui pencatatan hash ke blockchain Ethereum.

Diagram Arsitektur Sistem



Sistem ini terdiri atas beberapa komponen utama sebagai berikut:

1. Sensor Suhu dan Kelembaban (SHT20 / Modbus RTU)

Bertugas mengukur parameter lingkungan (suhu dan kelembaban) di dalam gudang secara kontinu.

1. Converter RS485 to USB

Menghubungkan sensor berbasis Modbus RTU ke komputer server untuk memungkinkan komunikasi data.

1. Program TCP Server (Rust)

Aplikasi yang dijalankan di komputer server untuk membaca data dari sensor, memprosesnya, dan menyimpan ke database InfluxDB.

1. Database Time-Series (InfluxDB)

Menyimpan data lingkungan secara historis berdasarkan timestamp dengan efisiensi tinggi.

1. Dashboard Web (Grafana)

Menyediakan tampilan visual real-time dan historis yang dapat diakses melalui browser untuk keperluan pemantauan.

1. Aplikasi Monitoring Lokal (Qt5/Qt6)

Menyediakan antarmuka desktop untuk pengguna yang membutuhkan pemantauan tanpa koneksi internet. Tampilan meliputi data suhu dan kelembaban terkini dalam bentuk tabel, grafik, atau indikator.

1. Smart Contract (Solidity)

Kontrak digital yang ditanamkan di blockchain Ethereum untuk menyimpan hash ringkasan data pada interval waktu tertentu (misalnya setiap 6 jam).

1. DApp (Web3.js / Ethers.js)

Aplikasi berbasis web terdesentralisasi yang memungkinkan pengguna memverifikasi hash data dari blockchain dan memastikan data belum dimanipulasi.

1. **Alur Sistem**

Alur sistem menggambarkan urutan proses dan interaksi antar komponen utama dalam sistem monitoring suhu dan kelembaban gudang beras berbasis InfluxDB, Grafana, Qt, dan Blockchain. Setiap komponen berperan dalam memastikan data lingkungan dapat diperoleh secara akurat, ditampilkan secara real-time, dan diverifikasi secara transparan.

Proses dimulai dari pengukuran suhu dan kelembaban oleh sensor industri sensor SHT20 yang ditempatkan di dalam gudang penyimpanan beras. Sensor ini bekerja secara kontinu dan mengirimkan data melalui protokol Modbus RTU menggunakan jalur komunikasi RS485. Untuk menghubungkannya ke komputer, digunakan konverter RS485-to-USB yang memungkinkan komunikasi serial antara sensor dan sistem pemrosesan

Data dari sensor diterima oleh program TCP Server yang dikembangkan menggunakan bahasa pemrograman Rust. Program ini bertugas membaca data dari port serial (USB), melakukan parsing, validasi, dan selanjutnya menyimpan data ke dalam InfluxDB, yaitu database time-series yang dirancang khusus untuk menyimpan data berdasarkan waktu.

Setelah tersimpan, data suhu dan kelembaban ditampilkan melalui dua jalur antarmuka pengguna. Pertama, secara lokal melalui aplikasi desktop berbasis Qt, yang menampilkan data dalam bentuk grafik, indikator, atau tabel untuk pengguna internal gudang. Kedua, secara daring menggunakan Grafana, yang terhubung langsung dengan InfluxDB untuk menyajikan data historis dan real-time melalui dashboard visual berbasis web.

Setiap interval waktu tertentu, sistem melakukan proses ringkasan data berupa nilai rata-rata, maksimum, dan minimum suhu serta kelembaban. Ringkasan ini kemudian dikonversi menjadi nilai hash (misalnya SHA-256) sebagai sidik jari digital atas kondisi gudang dalam rentang waktu tersebut. Nilai hash tersebut kemudian dicatat ke smart contract yang telah dideploy di jaringan Ethereum, menjadikan data tidak dapat diubah dan dapat diverifikasi publik.Untuk memfasilitasi transparansi, dibangun Decentralized Application (DApp) yang memungkinkan pengguna seperti distributor, pengelola, atau konsumen untuk memverifikasi hash data dari blockchain. Dengan demikian, seluruh pihak yang berkepentingan dapat memastikan bahwa kondisi penyimpanan dilakukan sesuai standar dan tidak mengalami manipulasi data. Alur ini tidak hanya menciptakan sistem monitoring fungsional, tetapi juga mendukung prinsip transparansi, traceability, dan akuntabilitas dalam rantai pasok pangan berbasis teknologi Web3

1. **Implementasi dan Kode Program**
   1. **Kode Rust Modbus Client**

Untuk mengambil data dari sensor menggunakan protokol Modbus RTU, langkah pertama adalah menghubungkan sensor ke sistem melalui koneksi serial menggunakan antarmuka RS-485 ke USB. Di sistem operasi Linux, koneksi ini umumnya dikenali sebagai port serial dengan path /dev/ttyUSB0. Sensor seperti SHT20 biasanya memiliki alamat slave 0x01 dalam jaringan Modbus. Selanjutnya, pembacaan data dilakukan dengan mengakses register tertentu pada sensor, seperti alamat register 0x0000 untuk data suhu dan 0x0001 untuk kelembaban. Nilai yang diperoleh dari sensor biasanya berbentuk integer dan perlu dikonversi menjadi float dengan membaginya dengan angka 10, karena sensor mengirimkan data dalam format nilai dikali 10. Setelah data suhu dan kelembaban terbaca dan dikonversi, data tersebut diubah menjadi format JSON untuk memudahkan pengolahan dan transmisi. Data JSON ini kemudian dikirim ke server melalui koneksi TCP menggunakan port 7878 setiap 10 detik, sehingga server menerima pembaruan data sensor secara real-time.

Kode Rust Modbus Client

|  |
| --- |
|  |

* 1. **Kode Rust TCP Server**

Tahap berikutnya adalah menerima data dari client menggunakan program Rust TCP Server. Server ini terus-menerus memonitor koneksi pada port USB. Ketika data JSON diterima, program memprosesnya dengan pustaka serde\_json untuk mengurai informasi yang dikirim oleh client. Setelah parsing, data disiapkan dan dikirimkan ke InfluxDB, sebuah database time-series yang dirancang untuk menyimpan data berurutan berdasarkan waktu. Data disimpan dengan struktur measurement untuk mencatat nama data, tags untuk metadata seperti ID sensor, dan fields untuk nilai-nilai aktual seperti suhu dan kelembaban.

Kode Rust TCP Server

|  |
| --- |
|  |

* 1. **Konfigurasi InfluxDB dan Integrasi**

Tahap berikutnya adalah menerima data dari client menggunakan program Rust TCP Server. Server ini terus-menerus memonitor koneksi pada port USB. Ketika data JSON diterima, program memprosesnya dengan pustaka serde\_json untuk mengurai informasi yang dikirim oleh client. Setelah parsing, data disiapkan dan dikirimkan ke InfluxDB, sebuah database time-series yang dirancang untuk menyimpan data berurutan berdasarkan waktu. Data disimpan dengan struktur measurement untuk mencatat nama data, tags untuk metadata seperti ID sensor, dan fields untuk nilai-nilai aktual seperti suhu dan kelembaban.

Hasil Query di InfluxDB CLI

|  |
| --- |
|  |

* 1. **Dashboard Grafana**

Tahap berikutnya adalah menerima data dari client menggunakan program Rust TCP Server. Server ini terus-menerus memonitor koneksi pada port USB. Ketika data JSON diterima, program memprosesnya dengan pustaka serde\_json untuk mengurai informasi yang dikirim oleh client. Setelah parsing, data disiapkan dan dikirimkan ke InfluxDB, sebuah database time-series yang dirancang untuk menyimpan data berurutan berdasarkan waktu. Data disimpan dengan struktur measurement untuk mencatat nama data, tags untuk metadata seperti ID sensor, dan fields untuk nilai-nilai aktual seperti suhu dan kelembaban.

Dashnoard Grafama

|  |
| --- |
|  |

* 1. **Dashboard QT**

Connecting ke InfluxDB

|  |
| --- |
|  |

Struktur Program

|  |
| --- |
|  |

**BAB IV**

**HASIL PENELITIAN**

* + 1. Hasil Pengujian

|  |
| --- |
|  |

* + 1. Hasil Penyimpanan InfluxDB

|  |
| --- |
|  |

* + 1. Hasil Real-time Dashboard Grafana

|  |
| --- |
|  |
|  |

* + 1. Hasil Real-time Dashboard Qt

|  |
| --- |
|  |

* + 1. Hasil Real-time Dashboard Web3

|  |
| --- |
|  |
|  |

**BAB V**

**PENUTUP**

* 1. **Kesimpulan**

Proyek ini berhasil mengembangkan sistem monitoring suhu dan kelembaban untuk gudang penyimpanan beras yang menggabungkan beberapa teknologi modern, seperti Internet of Things (IoT), blockchain Ethereum, InfluxDB, dan Grafana. Sistem ini bertujuan untuk memastikan bahwa kualitas beras tetap terjaga selama masa penyimpanan, dengan memberikan transparansi dan kepercayaan kepada semua pihak yang terlibat dalam rantai pasok. Sistem yang dibangun memiliki beberapa fitur utama, yaitu:

1. Pemantauan suhu dan kelembaban secara real-time melalui sensor SHT20.
2. Penyimpanan data menggunakan InfluxDB yang memungkinkan penyimpanan time-series dengan efisien.
3. Visualisasi data menggunakan Grafana dan Qt, yang memungkinkan pengguna untuk memantau kondisi gudang baik secara daring maupun offline.
4. Pencatatan hash data ke blockchain Ethereum untuk menjamin keaslian dan integritas data yang tidak dapat dimanipulasi.
5. DApp berbasis Web3 yang memungkinkan verifikasi publik atas data yang tercatat di blockchain.

Dengan penerapan blockchain dan Web3, sistem ini memberikan keamanan, transparansi, dan auditabilitas yang lebih tinggi dibandingkan dengan sistem konvensional yang hanya mengandalkan penyimpanan dan visualisasi data tradisional.

Secara keseluruhan, proyek ini menunjukkan potensi besar dalam menciptakan sistem yang efisien dan dapat diandalkan untuk mengurangi kerugian pangan, memastikan kualitas penyimpanan, dan meningkatkan kepercayaan konsumen.

* 1. **Saran**

Proyek ini berhasil membuat system monitoring kelembaban dan suhu untuk Gudang penyimpanan beras yang sesuai dengan tema proyek ini dengan penggunaan teknologi blockchain, Ethereum, InfluxDB dan Grafana. Namun, tidak menutup kemungkinan akan adanya saran dari penelitian ini meliputi:

1. Untuk meningkatkan keakuratan monitoring, disarankan untuk menambahkan sensor tambahan yang dapat memantau parameter lain seperti tekanan, gas, atau kondisi tanah (untuk pertanian). Hal ini akan memperluas penerapan sistem ini di sektor lain, seperti pertanian atau penyimpanan bahan pangan lainnya.
2. Meskipun blockchain memberikan tingkat keamanan yang tinggi, perlu dilakukan peningkatan sistem enkripsi untuk melindungi data sensitif dan privasi pengguna yang terhubung dengan sistem. Penggunaan private blockchain atau hybrid blockchain juga bisa menjadi opsi untuk meningkatkan kontrol atas data.
3. Proyek ini dapat dikembangkan lebih lanjut untuk mendukung skala yang lebih besar, seperti jaringan gudang atau sistem yang melibatkan banyak sensor dan lokasi berbeda. Diperlukan peningkatan dalam manajemen jaringan dan skalabilitas sistem agar dapat mengatasi volume data yang lebih besar.
4. Sistem ini bisa diperluas untuk mendukung sektor logistk, khususnya dalam distribusi barang sensitif seperti obat-obatan atau produk makanan beku. Integrasi dengan sistem pelacakan dan manajemen distribusi akan memberikan keamanan tambahan dalam pengiriman produk.
5. Pengujian lebih lanjut pada sistem secara keseluruhan perlu dilakukan di beberapa gudang atau lokasi yang berbeda untuk mengevaluasi keandalan, akurasi, dan ketahanan sistem dalam jangka panjang.

Dengan adanya pengembangan lebih lanjut dan penerapan teknologi ini di sektor lain, diharapkan sistem monitoring berbasis blockchain dan IoT ini dapat lebih luas diterima dan diimplementasikan untuk mendukung keberlanjutan rantai pasok pangan yang efisien dan transparan.

**DAFTAR PUSTAKA**

Altmeyer, S., & Konietzko, J. (2018). Qt-Based SCADA System for Industrial Applications. Journal of Industrial Control and Automation, 9(2), 56–61.

Babu, G. V., & Rao, M. M. (2022). Real-Time Smart Monitoring of Temperature and Humidity Using InfluxDB and Grafana. International Journal of Engineering and Advanced Technology, 11(3), 35–40.

Barksdale, A., & Stone, T. (2019). *Scalable Time-Series Data Infrastructure Using InfluxDB for IoT Sensor Monitoring*. *Journal of Sensor Networks*, 5(4), 202–210.

Blanchette, J., & Summerfield, M. (2006). C++ GUI Programming with Qt 4 (2nd ed.). Prentice Hall.

Buterin, V. (2020). The Meaning of Decentralization. Ethereum Foundation. https://ethereum.org/en/developers/docs/intro-to-web3/

Casino, F., Dasaklis, T. K., & Patsakis, C. (2019). A systematic literature review of blockchain-based applications: Current status, classification and open issues. Telecommunications Policy, 43(10), 101848. https://doi.org/10.1016/j.telpol.2019.101848

CNN Indonesia. (2024, February 5). *Bulog Akui Gudang Beras Sering Kelembaban Tinggi, Kualitas Menurun*. https://www.cnnindonesia.com/ekonomi/bulog-akui-kualitas-beras-menurun-karena-gudang-lembap

Dahalin, M., et al. (2021). IoT-Based Real-Time Environmental Monitoring System Using MQTT, InfluxDB, and Grafana. Indonesian Journal of Electrical Engineering and Computer Science, 22(1), 151–158. https://doi.org/10.11591/ijeecs.v22.i1.pp151-158

FAO. (2019). *Postharvest Management of Rice and Grains*. Food and Agriculture Organization of the United Nations.

Gupta, R., & Singh, H. (2020). Real-time IoT Dashboard Design Using Grafana and InfluxDB. International Journal of Computer Applications, 176(22), 1–6. https://doi.org/10.5120/ijca2020919934

Kementerian Pertanian RI. (2023). *Transformasi Pertanian Menuju Agriculture 4.0*. Jakarta: Pusat Data dan Sistem Informasi Pertanian.

Kumar, R., Singh, A., & Sharma, M. (2020). Implementation of Real-Time Industrial Data Monitoring System Using Modbus RTU Protocol. *Journal of Industrial Automation*, 12(3), 45–52.

Lin, Q., Wang, H., Pei, X., & Wang, X. (2020). Food safety traceability system based on blockchain and EPCIS. IEEE Access, 8, 131681–131692. https://doi.org/10.1109/ACCESS.2020.3009030

Palau, C. E., Esteve, M., & Esteve, A. (2021). Analysis of Modbus Protocols in Smart Industrial Environments. *International Journal of Distributed Sensor Networks*, 17(1), 1–10. https://doi.org/10.1177/1550147721992002

Prasetyo, A., & Darmawan, R. (2021). Analisa suhu dan kelembaban gudang penyimpanan beras menggunakan IoT. *Jurnal Teknologi Pertanian*, 22(3), 145–152.

Singh, R., et al. (2021). A Lightweight IoT Framework for Monitoring Smart Agriculture Using Node-RED, InfluxDB, and Grafana. International Journal of Advanced Computer Science and Applications, 12(6), 370–376. https://doi.org/10.14569/IJACSA.2021.0120647

Siregar, M., & Putra, R. (2022). IoT-Based Monitoring System for Rice Storage Warehouses Using Temperature and Humidity Sensors. *International Journal of Smart Agriculture*, 3(2), 75–83. https://doi.org/10.31289/ijsa.v3i2.6837

Srinivasan, S., Al-Jabery, K., & Velayutham, R. (2022). *Design and Evaluation of Time-Series Database Architectures for Industrial IoT Monitoring*. *Sensors*, 22(7), 2652. https://doi.org/10.3390/s22072652

Suliantoro, H., Pratama, M., & Widodo, S. (2021). Postharvest Losses of Rice in Indonesia: Identification and Mitigation. *Agricultural Science & Technology Journal*, 5(1), 1–10. https://doi.org/10.17977/um015v5i12021p1

Tempo. (2023, November 22). *Pemerintah Akui Kehilangan Pasca-Panen Beras Mencapai 5 Persen*. https://bisnis.tempo.co/read/1776475

Tian, F. (2016). An agri-food supply chain traceability system for China based on RFID & blockchain technology. *13th International Conference on Service Systems and Service Management (ICSSSM)*, 1–6.

Xie, J., Tang, M., Huang, T., & Yu, F. R. (2019). A survey of blockchain technology applied to smart cities: Research issues and challenges. IEEE Communications Surveys & Tutorials, 21(3), 2794–2830. https://doi.org/10.1109/COMST.2019.2899617

Yasin, M., Bahar, A., & Aziz, F. (2020). Real-Time Temperature and Humidity Monitoring System for Agricultural Storage Using Industrial-Grade Sensors. *International Journal of Smart Agriculture*, 4(1), 15–22.

Zheng, Z., Xie, S., Dai, H., Chen, X., & Wang, H. (2020). An overview of blockchain technology: Architecture, consensus, and future trends. IEEE International Congress on Big Data, 557–564. https://doi.org/10.1109/BigDataCongress.2020.00085

**LAMPIRAN**

**CODINGAN MODBUS CLIENT**

**1. Cargo.toml**

|  |
| --- |
| [package]  name = "sensor\_client"  version = "0.1.0"  edition = "2021"  [dependencies]  tokio = { version = "1.32", features = ["full"] }  tokio-serial = "5.4.0"  tokio-modbus = "0.5.0"  serde = { version = "1", features = ["derive"] }  serde\_json = "1"  chrono = { version = "0.4", features = ["serde"] } |

**2. main.rs**

|  |
| --- |
| use tokio\_modbus::{client::rtu, prelude::\*};  use tokio\_serial::{SerialPortBuilderExt, Parity, StopBits, DataBits};  use tokio::net::TcpStream;  use tokio::io::AsyncWriteExt;  use serde::Serialize;  use chrono::Utc;  use std::error::Error;  use tokio::time::{sleep, Duration};  #[derive(Serialize)]  struct SensorData {  timestamp: String,  sensor\_id: String,  location: String,  process\_stage: String,  temperature\_celsius: f32,  humidity\_percent: f32,  }  async fn read\_sensor(slave: u8) -> Result<Vec<u16>, Box<dyn Error>> {  let builder = tokio\_serial::new("/dev/ttyUSB0", 9600)  .parity(Parity::None)  .stop\_bits(StopBits::One)  .data\_bits(DataBits::Eight)  .timeout(std::time::Duration::from\_secs(1));  let port = builder.open\_native\_async()?;  let mut ctx = rtu::connect\_slave(port, Slave(slave)).await?;  let response = ctx.read\_input\_registers(1, 2).await?;  Ok(response)  }  #[tokio::main]  async fn main() -> Result<(), Box<dyn Error>> {  loop {  match read\_sensor(1).await {  Ok(response) if response.len() == 2 => {  let temp = response[0] as f32 / 10.0;  let rh = response[1] as f32 / 10.0;  println!("📡 Temp: {:.1} °C | RH: {:.1} %", temp, rh);  let data = SensorData {  timestamp: Utc::now().to\_rfc3339(),  sensor\_id: "SHT20-Gudang Beras".into(),  location: "Gudang Penyimpanan Beras".into(),  process\_stage: "Pasca Panen Beras".into(),  temperature\_celsius: temp,  humidity\_percent: rh,  };  let json = serde\_json::to\_string(&data)?;    match TcpStream::connect("127.0.0.1:9000").await {  Ok(mut stream) => {  stream.write\_all(json.as\_bytes()).await?;  stream.write\_all(b"\n").await?;  println!(" Data dikirim ke TCP server");  },  Err(e) => {  println!(" Gagal kirim ke TCP server: {}", e);  }  }  },  Ok(other) => {  println!(" Data tidak lengkap: {:?}", other);  },  Err(e) => {  println!(" Gagal baca sensor: {}", e);  }  }  sleep(Duration::from\_secs(5)).await;      }  } |

**CODINGAN TCP SERVER**

**1.cargo.toml**

|  |
| --- |
| [package]  name = "sensor\_server\_blockchain"  version = "0.1.0"  edition = "2021"  [dependencies]  tokio = { version = "1", features = ["full"] }  serde = { version = "1.0", features = ["derive"] }  serde\_json = "1.0"  reqwest = { version = "0.11", features = ["json"] }  chrono = "0.4"  ethers = { version = "2.0", features = ["abigen"] }  anyhow = "1.0" |

**2. main.rs**

|  |
| --- |
| use tokio::net::TcpListener;  use tokio::io::{AsyncBufReadExt, BufReader};  use serde::Deserialize;  use reqwest::Client;  use ethers::prelude::\*;  use ethers::abi::Abi;  use std::{fs, sync::Arc};  use chrono::DateTime;  #[derive(Deserialize, Debug)]  struct SensorData {  timestamp: String,  sensor\_id: String,  location: String,  process\_stage: String,  temperature\_celsius: f32,  humidity\_percent: f32,  }  #[tokio::main]  async fn main() -> anyhow::Result<()> {  // --- InfluxDB setup ---  let influx\_url = "http://localhost:8086/api/v2/write?org=kelisi5&bucket=iisi5&precision=s";  let influx\_token = "DF3XZjyfxca7uguMUCjhKVrxTmPYQjwmR9YJi9oHKObk5TStGO1P-0aaK1SY1Q0GhMT2pmoGNlH7JLgnKfspgg==";  let http\_client = Client::new();  // --- Ethereum setup ---  let provider = Provider::<Http>::try\_from("http://localhost:8545")?;  let wallet: LocalWallet = "0xa750a966afb44c4220bd6cd5bbff5a73eff1e2b8ba85098532a71744b4262431"  .parse::<LocalWallet>()?  .with\_chain\_id(1337u64);  let client = Arc::new(SignerMiddleware::new(provider, wallet));  // Debug path  let abi\_path = "build/SensorStorage.abi";  let bytecode\_path = "build/SensorStorage.bin";  println!(" Membaca ABI dari path: {}", abi\_path);  println!(" Membaca Bytecode dari path: {}", bytecode\_path);  let abi\_str = fs::read\_to\_string(abi\_path)  .map\_err(|e| anyhow::anyhow!(" Gagal baca ABI di '{}': {}", abi\_path, e))?;  let bytecode\_str = fs::read\_to\_string(bytecode\_path)  .map\_err(|e| anyhow::anyhow!(" Gagal baca Bytecode di '{}': {}", bytecode\_path, e))?;  println!(" File ABI dan Bytecode berhasil dibaca");  let abi: Abi = serde\_json::from\_str(&abi\_str)?;  let bytecode = bytecode\_str.trim().parse::<Bytes>()?;  let factory = ContractFactory::new(abi, bytecode, client.clone());  let contract = factory.deploy(())?.send().await?;  println!(" Smart contract deployed at: {:?}", contract.address());  // --- TCP Server ---  let listener = TcpListener::bind("0.0.0.0:9000").await?;  println!("TCP Server listening on port 9000...");  loop {  let (socket, addr) = listener.accept().await?;  println!(" New connection from {}", addr);  let influx\_url = influx\_url.to\_string();  let influx\_token = influx\_token.to\_string();  let http\_client = http\_client.clone();  let contract = contract.clone();  tokio::spawn(async move {  let reader = BufReader::new(socket);  let mut lines = reader.lines();  while let Ok(Some(line)) = lines.next\_line().await {  match serde\_json::from\_str::<SensorData>(&line) {  Ok(data) => {  println!(" Received sensor data: {:?}", data);  // --- InfluxDB Write ---  let timestamp = DateTime::parse\_from\_rfc3339(&data.timestamp)  .unwrap()  .timestamp();  let line\_protocol = format!(  "monitoring,sensor\_id={},location={},stage={} temperature={},humidity={} {}",  data.sensor\_id.replace(" ", "\\ "),  data.location.replace(" ", "\\ "),  data.process\_stage.replace(" ", "\\ "),  data.temperature\_celsius,  data.humidity\_percent,  timestamp  );  match http\_client  .post(&influx\_url)  .header("Authorization", format!("Token {}", influx\_token))  .header("Content-Type", "text/plain")  .body(line\_protocol)  .send()  .await  {  Ok(resp) if resp.status().is\_success() => {  println!(" InfluxDB: data written");  }  Ok(resp) => {  println!("⚠ InfluxDB error: {}", resp.status());  }  Err(e) => {  println!(" InfluxDB HTTP error: {}", e);  }  }  // --- Ethereum Contract Write ---  let method\_call = contract  .method::<\_, H256>("storeData", (  timestamp as u64,  data.sensor\_id.clone(),  data.location.clone(),  data.process\_stage.clone(),  (data.temperature\_celsius \* 100.0) as i64,  (data.humidity\_percent \* 100.0) as i64,  ))  .unwrap();  let tx = method\_call.send().await;  match tx {  Ok(pending\_tx) => {  println!("📡 Ethereum: tx sent: {:?}", pending\_tx);  }  Err(e) => {  println!(" Ethereum tx error: {:?}", e);  }  }  }  Err(e) => println!(" Invalid JSON received: {}", e),  }  }       });      }  } |

**CODINGAN QT**

**1. main.py**

|  |
| --- |
| import tkinter as tk  from tkinter import ttk  import requests  import threading  import time  import csv  from io import StringIO  from matplotlib.backends.backend\_tkagg import FigureCanvasTkAgg  from matplotlib.figure import Figure  from collections import deque  # Konfigurasi InfluxDB  INFLUX\_QUERY\_URL = "http://localhost:8086/api/v2/query"  ORG = "kelisi5"  BUCKET = "iisi5"  TOKEN = "DF3XZjyfxca7uguMUCjhKVrxTmPYQjwmR9YJi9oHKObk5TStGO1P-0aaK1SY1Q0GhMT2pmoGNlH7JLgnKfspgg=="  # Riwayat data  history\_length = 50  temp\_history = deque(maxlen=history\_length)  rh\_history = deque(maxlen=history\_length)  time\_history = deque(maxlen=history\_length)  def get\_latest\_data():  flux\_query = f'''  from(bucket: "{BUCKET}")  |> range(start: -1m)  |> filter(fn: (r) => r.\_measurement == "monitoring")  |> filter(fn: (r) => r.\_field == "temperature" or r.\_field == "humidity")  |> last()  '''  headers = {  "Authorization": f"Token {TOKEN}",  "Content-Type": "application/vnd.flux",  "Accept": "application/csv"  }  try:  response = requests.post(  INFLUX\_QUERY\_URL,  params={"org": ORG},  headers=headers,  data=flux\_query  )  reader = csv.DictReader(StringIO(response.text))  data = {}  for row in reader:  try:  field = row["\_field"]  value = float(row["\_value"])  data[field] = value  except:  continue  if "temperature" in data and "humidity" in data:  return data["temperature"], data["humidity"]  return None  except Exception as e:  print(" Exception query Influx:", e)  return None  def get\_data\_range(start\_time, end\_time):  flux\_query = f'''  from(bucket: "{BUCKET}")  |> range(start: {start\_time}, stop: {end\_time})  |> filter(fn: (r) => r.\_measurement == "monitoring")  |> filter(fn: (r) => r.\_field == "temperature" or r.\_field == "humidity")  '''  headers = {  "Authorization": f"Token {TOKEN}",  "Content-Type": "application/vnd.flux",  "Accept": "application/csv"  }  try:  response = requests.post(  INFLUX\_QUERY\_URL,  params={"org": ORG},  headers=headers,  data=flux\_query  )  reader = csv.DictReader(StringIO(response.text))  temp\_map = {}  rh\_map = {}  for row in reader:  try:  t = row["\_time"]  field = row["\_field"]  value = float(row["\_value"])  if field == "temperature":  temp\_map[t] = value  elif field == "humidity":  rh\_map[t] = value  except:  continue  sorted\_keys = sorted(set(temp\_map.keys()) & set(rh\_map.keys()))  temps = [temp\_map[t] for t in sorted\_keys]  rhs = [rh\_map[t] for t in sorted\_keys]  times = [t[11:19] for t in sorted\_keys] # jam:menit:detik  return temps, rhs, times  except Exception as e:  print(" Exception query Influx:", e)  return [], [], []  def update\_data():  while True:  result = get\_latest\_data()  current\_time = time.strftime('%H:%M:%S')  if result:  temp, rh = result  label\_temp.config(text=f"Suhu: {temp:.1f} °C")  label\_rh.config(text=f"Kelembaban: {rh:.1f} %")  status\_label.config(text=" Data dari Influx")  temp\_history.append(temp)  rh\_history.append(rh)  time\_history.append(current\_time)  plot\_graph()  else:  label\_temp.config(text="Suhu: ---")  label\_rh.config(text="Kelembaban: ---")  status\_label.config(text=" Gagal ambil data")  time.sleep(2)  def plot\_graph():  ax1.clear()  ax2.clear()  # Set background ke hitam  fig.patch.set\_facecolor('black')  ax1.set\_facecolor('black')  ax2.set\_facecolor('black')  x = list(range(len(time\_history)))  times = list(time\_history)  # Plot data dengan garis dan warna yang kontras  ax1.plot(x, list(temp\_history), label='Suhu (°C)', color='red', marker='o', linestyle='-')  ax2.plot(x, list(rh\_history), label='Kelembaban (%)', color='cyan', marker='x', linestyle='-')  ax1.set\_title("Grafik Suhu", color='white')  ax2.set\_title("Grafik Kelembaban", color='white')  ax1.set\_ylabel("°C", color='white')  ax2.set\_ylabel("%", color='white')  # Tampilkan hanya label waktu setiap 5 data  interval = 5  tick\_positions = x[::interval]  tick\_labels = times[::interval]  ax1.set\_xticks(tick\_positions)  ax2.set\_xticks(tick\_positions)  ax1.set\_xticklabels(tick\_labels, rotation=45, ha="right", color='white')  ax2.set\_xticklabels(tick\_labels, rotation=45, ha="right", color='white')  for ax in [ax1, ax2]:  ax.tick\_params(axis='y', colors='white')  ax.tick\_params(axis='x', colors='white')  ax.grid(True, linestyle='--', alpha=0.3, color='gray')  for spine in ax.spines.values():  spine.set\_color('white')  fig.tight\_layout()  canvas.draw()  def show\_history():  start = entry\_start.get()  end = entry\_end.get()  temps, rhs, times = get\_data\_range(start, end)  if temps and rhs:  temp\_history.clear()  rh\_history.clear()  time\_history.clear()  temp\_history.extend(temps)  rh\_history.extend(rhs)  time\_history.extend(times)  label\_temp.config(text="(Hist) Suhu: -- °C")  label\_rh.config(text="(Hist) RH: -- %")  status\_label.config(text=" Menampilkan data historis")  plot\_graph()  else:  status\_label.config(text=" Tidak ada data historis")  # GUI Setup  root = tk.Tk()  root.title("Monitor SHT20 dari InfluxDB")  root.geometry("800x650")  label\_temp = tk.Label(root, text="Suhu: -- °C", font=("Helvetica", 16))  label\_temp.pack(pady=5)  label\_rh = tk.Label(root, text="Kelembaban: -- %", font=("Helvetica", 16))  label\_rh.pack(pady=5)  status\_label = tk.Label(root, text="Status: ---", fg="blue")  status\_label.pack(pady=5)  frame\_input = tk.Frame(root)  frame\_input.pack(pady=5)  tk.Label(frame\_input, text="Start (RFC3339):").grid(row=0, column=0, padx=5)  entry\_start = tk.Entry(frame\_input, width=30)  entry\_start.grid(row=0, column=1)  tk.Label(frame\_input, text="End (RFC3339):").grid(row=1, column=0, padx=5)  entry\_end = tk.Entry(frame\_input, width=30)  entry\_end.grid(row=1, column=1)  btn\_show = tk.Button(frame\_input, text="Tampilkan Riwayat", command=show\_history)  btn\_show.grid(row=0, column=2, rowspan=2, padx=10)  fig = Figure(figsize=(6, 4), dpi=100)  ax1 = fig.add\_subplot(211)  ax2 = fig.add\_subplot(212)  canvas = FigureCanvasTkAgg(fig, master=root)  canvas.get\_tk\_widget().pack(pady=10)  # Mulai update realtime  threading.Thread(target=update\_data, daemon=True).start()  root.mainloop() |