LAPORAN PROJECT
INTERKONEKSI SISTEM INSTRUMENTASI

Smart Farming : Integrasi IoT dan Blockchain untuk Monitoring Hidroponik Berbasis Web3

Kelompok 10:

Axel Fitra Ananda	2042231002
Lu'lu' Rusyida Hamudyah	2042231058
Mushthafa Ali Akbar	2042231082

PRODI D4 TEKNOLOGI REKAYASA INSTRUMENTASI
DEPARTEMEN TEKNIK INSTRUMENTASI
FAKULTAS VOKASI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER

2025

BAB 1

Pendahuluan

1.1 Latar Belakang

Perkembangan teknologi di era industri 4.0 telah mendorong transformasi signifikan dalam berbagai sektor, termasuk pertanian. Salah satu bentuk inovasi tersebut adalah konsep smart farming yang didukung *Internet of Things (*IoT) untuk monitoring kondisi lingkungan secara otomatis dan real time. Dalam sistem hidroponik, kestabilan parameter seperti pH, suhu, kelembaban , dan tingkat nutrisi menjadi sangat krusial untuk menjamin pertumbuhan tanaman yang optimal. Namun, tantangan utama dari sistem ini adalah keterbatasan transparansi dan kepercayaan dan data yang dikumpulkan, terutama jika digunakan dalam skala besar atau melibatkan banyak pemangku kepentingan.

Teknologi Blockchain hadir menjadi solusi untuk menciptakan sistem monitoring yang tidak hanya cerdas tetapi juga transparan dan akuntabel. Dengan menggunakan smart contract dan decentralized application, data sensor dapat dicatat secara otomatis dalam jaringan blockchain, sehingga dapat manipulasi dan dapat diverifikasi oleh berbagai pihak dalam rantai pasok pertanian. Integrasi teknologi Web3 memungkinkan pengguna untuk berinteraksi langsung dengan data secara aman dan terdesentralisasi melalui dompet digital. Oleh karena itu, proyek ini mengambangkan sistem monitoring hidroponik berbasis IoT yang terintegrasi dengan Blockchain dan Web3 untuk mendukung praktik pertanian hidroponik yang berkelanjutan, transparan, dan modern.

1.2 Rumusan Masalah

1. Bagaimana merancang sistem monitoring pertanian hidroponik berbasis IoT untuk memantau parameter lingkungan secara real-time?
2. Bagaimana cara mengintegrasikan data sensor dari IoT ke dalam blockchain menggunakan smart contract?
3. Bagaimana membangun antarmuka DApp berbasis Web3 yang pengguna dapat mengakses dan memverifikasi data monitoring secara transparan dan aman?

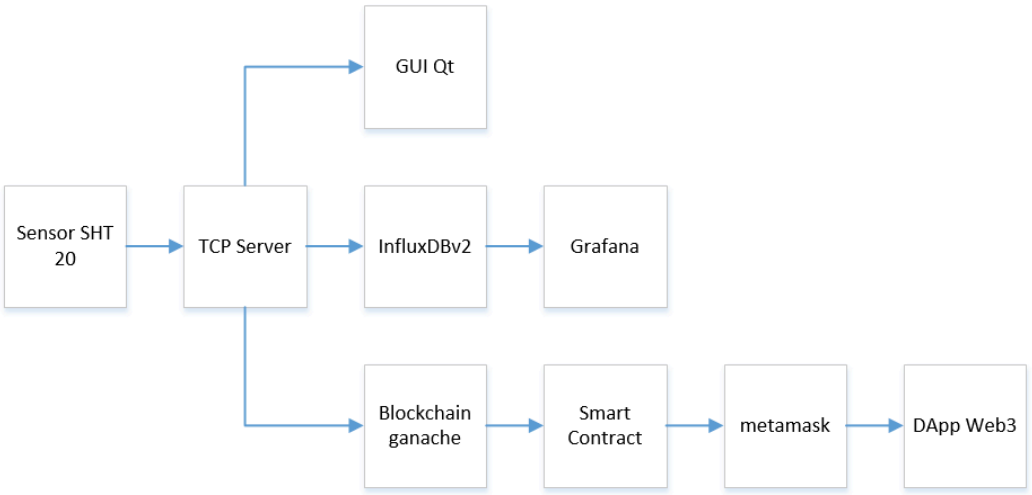
1.3 Tujuan Proyek

1. Mengembangkan sistem monitoring hidroponik berbasis IoT untuk membaca parameter seperti suhu, kelembaban, pH, dan nutrisi air secara otomatis
2. Mengimplementasikan teknologi blockchain untuk mencatat data sensor secara immutable menggunakan smart contract
3. Membangun DApp berbasis Web3 yang memungkinkan pengguna mengakses data monitoring secara real-time, transparan, dan terdesentralisasi.

BAB 2

TINJAUAN PUSTAKA

2.1 Arsitektur Sistem



Desain arsitektur sistem yang ditampilkan menggambarkan integrasi antara perangkat IoT, server backend, database time-series, antarmuka pengguna desktop dan web, serta teknologi blockchain. Sistem diawali dari sensor SHT20, yang berfungsi untuk membaca data suhu dan kelembapan lingkungan. Sensor ini terhubung ke TCP Server yang dikembangkan menggunakan bahasa pemrograman Rust dan bertugas sebagai pusat komunikasi data.

Dari TCP Server, data sensor didistribusikan ke beberapa komponen sistem secara paralel. Jalur pertama mengarah ke InfluxDB v2, yaitu database time-series yang menyimpan data historis pengukuran sensor. Data yang tersimpan kemudian divisualisasikan melalui Grafana, memungkinkan pengguna untuk melakukan pemantauan tren suhu dan kelembapan secara real-time melalui antarmuka berbasis web.

Jalur kedua mengarah ke Blockchain Ganache, yang digunakan sebagai jaringan blockchain lokal untuk keperluan pengujian dan pencatatan data secara transparan dan tidak dapat diubah. Data sensor dikirim ke smart contract yang telah dideploy pada jaringan Ganache. Proses ini memungkinkan pencatatan data secara terdesentralisasi dan aman. Smart contract terhubung dengan Metamask, sebuah dompet digital berbasis ekstensi browser yang menjadi perantara antara pengguna dan blockchain. Melalui Metamask, pengguna dapat mengakses data melalui DApp

Web3, yaitu antarmuka web yang dibangun menggunakan framework modern seperti React.

Selain itu, sistem juga menyediakan antarmuka GUI berbasis Qt (PyQt), yang menampilkan data sensor secara real-time langsung dari TCP Server tanpa memerlukan koneksi ke internet atau browser. Antarmuka ini dirancang dengan tampilan minimalis dan interaktif untuk memudahkan pengguna dalam memantau data secara langsung dari perangkat desktop.

Dengan kombinasi komponen-komponen tersebut, sistem ini dirancang untuk mendukung monitoring data sensor secara real-time, transparan, dan multi-platform. Integrasi antara teknologi IoT, database time-series, blockchain, dan antarmuka pengguna memungkinkan sistem untuk digunakan dalam berbagai skenario seperti pemantauan lingkungan, riset ilmiah, dan otomasi industri.

2.2. Deskripsi Komponen

2.2.1 SHT20

SHT20 adalah sensor suhu dan kelembaban digital berkinerja tinggi yang populer, dikembangkan oleh Sensirion. Sensor ini memanfaatkan teknologi CMOSens® yang mengintegrasikan elemen sensor kelembaban kapasitif, sensor suhu *band-gap*, dan sirkuit analog/digital pada satu *chip*, menghasilkan akurasi dan stabilitas yang sangat baik dengan konsumsi daya minimal. SHT20 mengirimkan data yang sudah dikalibrasi dan dilinearisasi dalam format digital melalui antarmuka I2C, memudahkan integrasinya dengan mikrokontroler seperti Arduino atau ESP32. Berkat ukurannya yang kompak dan efisiensi dayanya, SHT20 banyak digunakan dalam aplikasi pemantauan iklim, rumah pintar, pertanian, dan industri yang membutuhkan pembacaan suhu dan kelembaban yang akurat.

2.1.2 RS485

RS485 adalah standar komunikasi serial pada lapisan fisik yang sangat andal, terutama untuk transmisi data jarak jauh di lingkungan industri. Keunggulan utamanya terletak pada sistem transmisi diferensial yang menggunakan sepasang kabel terpilin; perbedaan tegangan antara kedua kabel ini merepresentasikan data, sehingga sangat efektif dalam menekan *noise* (gangguan) elektromagnetik. RS485 juga mendukung topologi multi-drop, memungkinkan banyak perangkat (node) untuk terhubung pada satu bus komunikasi yang sama, di mana satu perangkat *master* dapat berkomunikasi dengan beberapa *slave*. Standar ini mampu mengirimkan data hingga jarak 1200 meter dengan kecepatan yang bervariasi, mencapai hingga 10 Mbps pada jarak yang lebih pendek. Umumnya beroperasi dalam mode half-duplex (mengirim atau menerima, tidak bersamaan), RS485 sering menjadi fondasi komunikasi di sistem otomatisasi industri seperti SCADA, kontrol motor, dan sistem keamanan, sering kali dipasangkan dengan protokol seperti Modbus RTU.

2.1.3 InfluxDB

InfluxDB adalah database yang dioptimalkan untuk menyimpan data time series, sangat cocok untuk aplikasi pemantauan seperti ini. Data suhu dan kelembaban yang diterima dari TCP Server disimpan dengan efisien, memungkinkan akses dan analisis data yang cepat.

2.1.4 Grafana

Grafana berfungsi sebagai platform visualisasi data yang memungkinkan pengguna untuk membuat dashboard interaktif. Dengan Grafana, pengguna dapat memantau data suhu dan kelembaban secara real-time dan melakukan analisis mendalam terhadap data yang telah dikumpulkan.

BAB 3

HASIL PERCOBAAN

3.1 Kode Rust Modbus Client

Untuk membaca data dari sensor suhu dan kelembaban, kelompok kami menggunakan bahasa Rust dengan pendekatan asynchronous. Sensor SHT20 dihubungkan ke sistem melalui antarmuka RS-485 ke USB, dan dikenali sebagai /dev/ttyUSB0 di sistem operasi Linux. Komunikasi dilakukan menggunakan protokol Modbus RTU, dengan alamat slave sensor sebesar 0x01

Data tersebut dikonversi ke dalam format JSON menggunakan pustaka serde_json dan dikirim ke server lokal melalui koneksi TCP (port 9000). Proses ini dilakukan setiap 15 detik secara berkala menggunakan fitur tokio::time::sleep, sehingga sistem dapat melakukan pengiriman data sensor secara real-time dan efisien. Contoh payload JSON yang dikirim:

```
let data = SensorData {
                timestamp: timestamp.clone(),
                sensor_id: "SHT20-PascaPanen-001".into(),
                location: "Gudang Fermentasi 1".into(),
                process_stage: "Fermentasi".into(),
                temperature_celsius: temp,
                humidity_percent: rh,
            };
```

Data tersebut dikonversi ke dalam format JSON menggunakan pustaka serde_json dan dikirim ke server lokal melalui koneksi TCP (port 7878). Proses ini dilakukan setiap 15 detik secara berkala menggunakan fitur tokio::time::sleep, sehingga sistem dapat melakukan pengiriman data sensor secara real-time dan efisien. Contoh payload JSON yang dikirim:

3. 2 InfluxDB

Server dibangun untuk berjalan pada port 9000 dan menerima data dari client secara asynchronous. Ketika koneksi masuk diterima, server membaca data dalam format JSON dan memprosesnya menggunakan pustaka serde_json untuk diubah menjadi objek Rust terstruktur. Data sensor tersebut kemudian ditulis ke dalam InfluxDB dengan struktur sebagai berikut:

- Measurement: monitoring SHT20
- Tags (untuk proses filter & identifikasi):

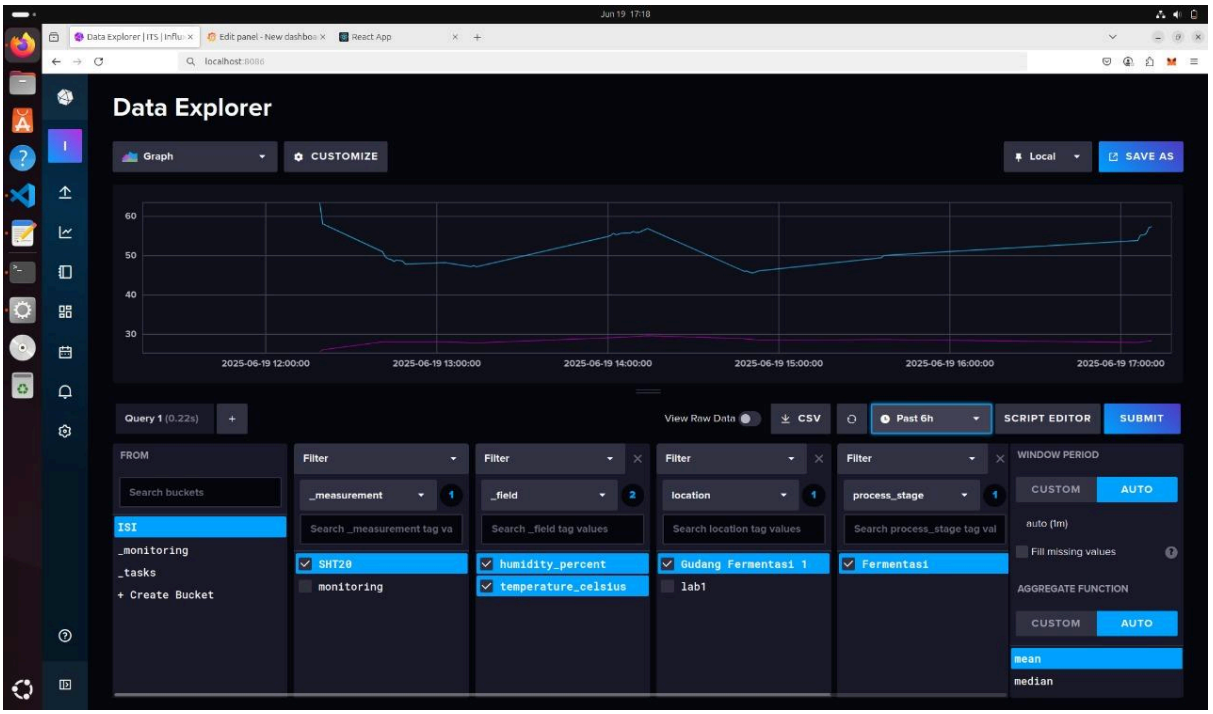
o sensor_id: ID unik sensor, contoh SHT20-PascaPanen-001

o shipment_id: Identitas kontainer atau lokasi fisik, seperti Gudang Fermentasi 1

o process_stage: Tahapan proses, contoh Fermentasi

• Fields (nilai aktual yang akan dianalisis): o temperature_celsius: Suhu dalam °C o humidity_percent: Kelembaban dalam %

• Timestamp: Diambil dari waktu pembacaan sensor dalam format RFC3339, lalu dikonversi menjadi Unix timestamp (dalam detik) untuk memenuhi format precision=s di InfluxDB v2.



3.3 Grafana

Dashboard Grafana ini digunakan untuk memvisualisasikan data sensor secara real-time.

Grafana terhubung langsung dengan InfluxDB, yang berfungsi sebagai sumber data utama dan menyimpan parameter suhu serta kelembaban dari berbagai pengiriman buah.

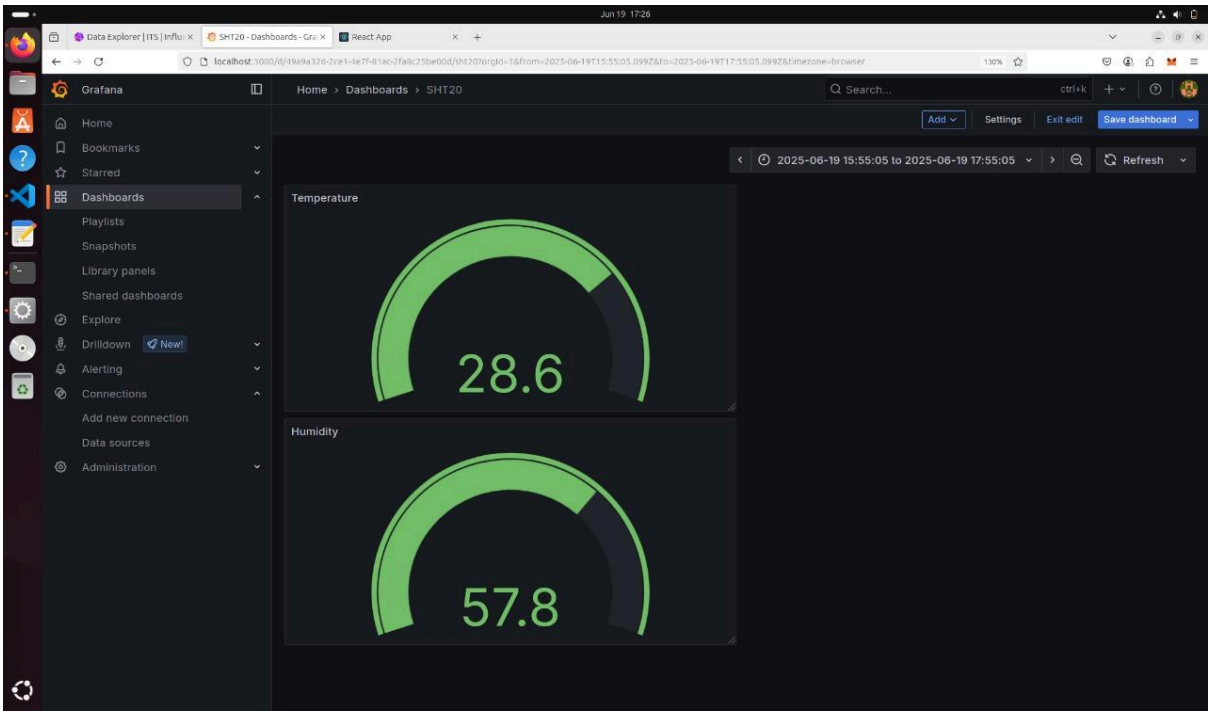
Pada dashboard, terdapat dua grafik utama:

- Grafik suhu terhadap waktu (*temperature_celsius*)

- Grafik kelembaban terhadap waktu (*humidity_percent*)

Setiap grafik dilengkapi fitur penyaringan, sehingga pengguna dapat memantau kondisi masing-masing field secara individual. Selain grafik, terdapat panel informasi statis yang menampilkan pembacaan sensor terbaru.

Grafana juga dikonfigurasi dengan filter waktu (seperti 1 jam terakhir, hari ini, atau 7 hari terakhir) untuk mempermudah pemantauan tren kondisi. Sistem ini dapat dikembangkan lebih lanjut dengan fitur *alerting*, yaitu notifikasi otomatis apabila suhu atau kelembaban melebihi batas yang telah ditentukan, guna mendukung pengambilan keputusan cepat selama proses pemberdayaan hidroponik



3.4 QtGUI

1. Inisialisasi Proyek GUI
   - Membuat file Python utama (`main.py`) yang akan menjalankan antarmuka GUI menggunakan `QWidget`.

   - Mengatur window dengan judul, ukuran, dan tata letak menggunakan `QVBoxLayout`.

2. Desain Antarmuka Pengguna (UI)
   - Menambahkan tiga label utama: suhu, kelembaban, dan status koneksi.

- Mengatur tampilan label dengan `setStyleSheet` untuk memperjelas informasi.

- Mengintegrasikan `pyqtgraph.PlotWidget` untuk menampilkan grafik real-time.

3. Menyiapkan Grafik Waktu Nyata
   - Menambahkan dua kurva pada grafik: satu untuk suhu (`temperature`) dan satu untuk kelembaban (`humidity`).

   - Mengaktifkan grid, legenda, dan latar belakang putih agar grafik mudah dibaca.

   - Menyimpan data dalam list `temp_data`, `hum_data`, dan `time_data` untuk plotting.

4. Timer Real-Time
   - Menggunakan `QTimer` untuk menjalankan fungsi `update_plot()` setiap 1 detik.

   - Fungsi `update_plot()` memperbarui grafik berdasarkan data terbaru yang diterima.

5. Komunikasi TCP dengan Server
   - Membuat thread terpisah (`threading.Thread`) agar GUI tetap responsif saat mendengarkan data TCP.

   - Fungsi `tcp_listener()` melakukan:

     - Koneksi ke server (misal di `127.0.0.1:9000`)

     - Mengirim identifikasi `"GUI_CLIENT\n"` ke server

     - Menerima data JSON dari server secara terus-menerus

     - Memancarkan sinyal ke thread GUI jika data diterima

6. Sistem Sinyal & Slot (Thread-Safe)
   - Menggunakan `pyqtSignal` dari `SignalEmitter` untuk memastikan pembaruan UI hanya dilakukan di thread GUI utama.

   - Dua sinyal utama:

     - `data_received_signal`: membawa data JSON

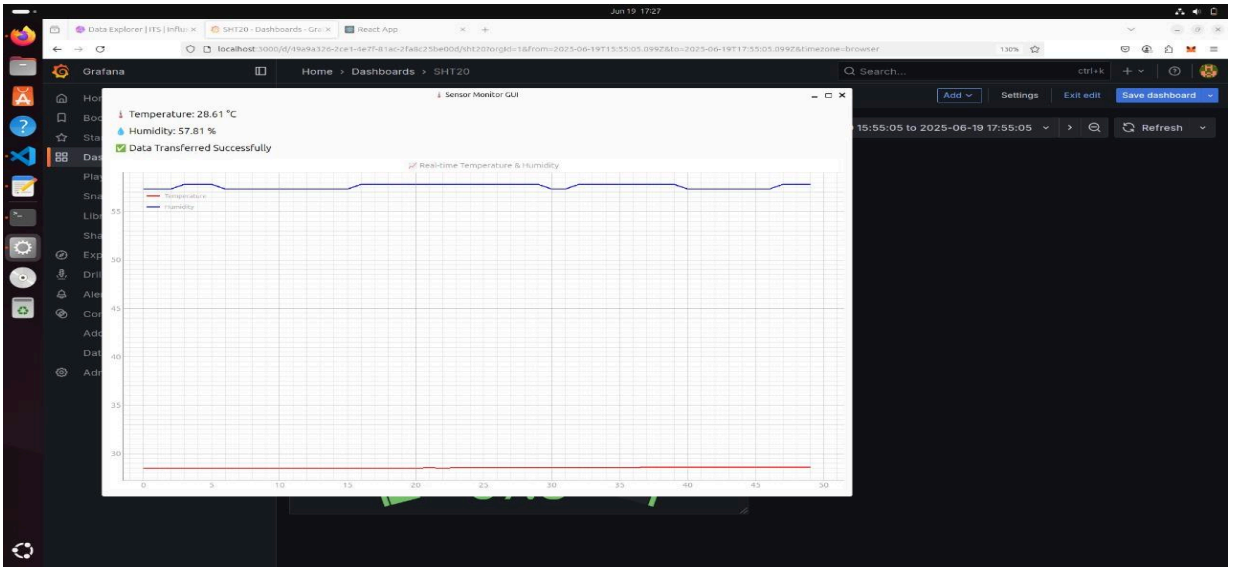
    - `connection_status_signal`: memperbarui status koneksi

7. Pemrosesan Data JSON
    - Fungsi `process_json()` dipanggil saat data JSON diterima:
        - Melakukan parsing JSON
        - Mengambil nilai `temperature_celsius` dan `humidity_percent`
        - Memperbarui label dan menambahkan data ke list grafik
        - Menangani error seperti format JSON salah atau key hilang

8. Visualisasi Real-Time
    - Data terbaru ditampilkan secara langsung pada grafik dan label GUI.
    - Hanya 50 titik data terakhir yang ditampilkan untuk menjaga performa.

9. Penanganan Error
    - Seluruh bagian program dibungkus dengan `try-except` untuk menangani kemungkinan:
        - Koneksi TCP gagal
        - Format data JSON salah
        - Key JSON tidak ditemukan

10. Eksekusi Aplikasi
    - Pada bagian `if __name__ == "__main__":`, GUI diinisialisasi dan dijalankan dengan `QApplication`.



3.5 Web3

Persiapan Lingkungan Pengembangan Blockchain Sebelum melakukan integrasi dengan blockchain, dilakukan beberapa persiapan:

• Instalasi react.js dan npm untuk mengelola dependensi.

• Instalasi dan konfigurasi Hardhat, yaitu framework pengembangan smart contract berbasis Ethereum. • Instalasi MetaMask, dompet digital berbasis browser untuk menghubungkan DApp dengan jaringan blockchain lokal.

1. Pengembangan Smart Contract Monitoring.sol

Smart contract dikembangkan menggunakan bahasa **Solidity versi ^0.8.20**, dengan struktur yang mendukung pencatatan data sensor ke dalam jaringan blockchain. Di dalam kontrak, didefinisikan sebuah struct bernama DataPoint yang menyimpan parameter penting yaitu *timestamp*, *shipment_id*, suhu (dalam bentuk integer dikalikan 10), dan kelembaban (juga dalam bentuk integer dikalikan 10).

Seluruh data tersebut disimpan dalam array allDataPoints, sementara variabel owner berfungsi sebagai identitas kepemilikan kontrak untuk mencegah akses tidak sah. Penambahan data dilakukan melalui fungsi addDataPoint() yang hanya dapat dijalankan oleh pemilik kontrak, berkat penggunaan modifier onlyOwner.

Setiap kali data baru ditambahkan, event DataPointAdded dipicu sebagai sinyal keberhasilan. Jumlah data yang tercatat dapat diketahui melalui fungsi getDataPointCount(). Seluruh kontrak dikompilasi menggunakan perintah:

npx hardhat compile

2. Deployment ke Jaringan Lokal Hardhat

Untuk melakukan *deployment*, jaringan lokal dijalankan dengan perintah:

npx hardhat node

Kemudian dibuat skrip scripts/deploy.js menggunakan library ethers.js untuk mengatur proses deploy. Skrip dijalankan dengan:
npx hardhat run scripts/deploy.js --network localhost

Setelah berhasil, alamat kontrak (contract address) dicatat untuk digunakan oleh sistem backend (Rust server) dan frontend (DApp).

3. Integrasi Smart Contract dengan Server TCP (Rust)
Agar data dari sensor dapat langsung tercatat di blockchain, server TCP (ditulis dengan Rust) dimodifikasi. Beberapa dependensi ditambahkan, antara lain:

- ethers: untuk melakukan interaksi dengan blockchain Ethereum,
- dotenv: untuk memuat variabel lingkungan,
- hex: untuk konversi dan manipulasi data dalam format heksadesimal.

File .env dibuat untuk menyimpan informasi penting seperti:

- PRIVATE_KEY dari akun pertama yang tersedia di jaringan Hardhat lokal,
- CONTRACT_ADDRESS hasil dari proses deploy sebelumnya.

Untuk menghubungkan Rust dengan kontrak Solidity, file ABI (Monitoring.json) diletakkan di folder tcp_server/abi/, kemudian abigen! macro digunakan untuk membuat binding kontrak:
rust
SalinEdit
abigen!(Monitoring, "./abi/Monitoring.json");

Sebuah fungsi baru bernama write_to_blockchain() dikembangkan untuk:

- Terhubung dengan node Hardhat di http://127.0.0.1:8545,
- Menginisialisasi wallet dari private key,
- Membuat instance kontrak berdasarkan ABI dan alamat,
- Memanggil fungsi addDataPoint() dengan data dari sensor.

Fungsi ini kemudian dipanggil dalam handle_connection untuk memastikan bahwa setiap data yang diterima dari sensor akan langsung tercatat di blockchain.
4. Pengembangan DApp Sederhana
Untuk memudahkan pemantauan data blockchain melalui browser, dikembangkan aplikasi desentralisasi (DApp) sederhana yang terdiri dari:

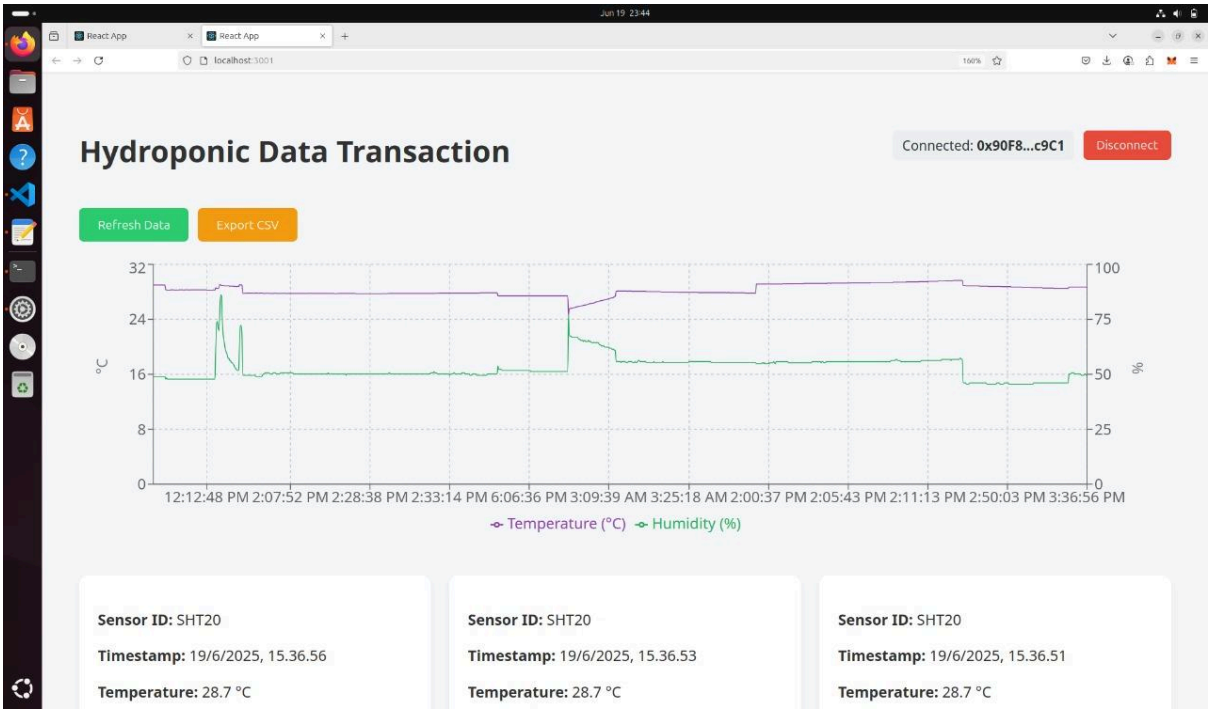
- index.html: berisi struktur HTML dan tampilan tabel data,
- app.js: berisi logika untuk menghubungkan dengan blockchain dan mengambil data.

DApp menggunakan ethers.js di sisi klien untuk:
- Terkoneksi ke dompet MetaMask,
- Mengakses kontrak pintar menggunakan ABI dan alamat kontrak,
- Menjalankan fungsi getDataPointCount() dan allDataPoints() dari kontrak,
- Menampilkan hasil pembacaan dalam bentuk tabel berisi waktu, ID pengiriman, suhu, dan kelembaban.

Agar MetaMask dapat berfungsi dengan jaringan lokal, dikonfigurasikan sebagai berikut:
- **Network Name**: Hardhat Local
- **RPC URL**: http://127.0.0.1:8545
- **Chain ID**: 31337

5. Pengujian Sistem End-to-End

Setelah seluruh komponen dikembangkan, sistem diuji secara menyeluruh dengan langkah berikut:
1. Menjalankan node lokal Hardhat.
2. Memulai server TCP berbasis Rust.
3. Menjalankan klien Modbus Rust untuk membaca dan mengirimkan data sensor.
4. Membuka DApp di browser dan memastikan data dari blockchain ditampilkan secara real-time.



3.6 Hasil yang dicapai

- Prototype sistem monitoring suhu dan kelembaban yang berjalan secara real-time dan terintegrasi dari sensor dan penyimpanan data terpusat dan terdesentralisasi
- Visualisasi data melalui Grafana dan aplikasi desktop PyQt
- Smart Contract yang berfungsi mencatat data sensor secara transparan dan tidak dapat diubah di blockchain simulasi
- DApp sederhana yang menunjukkan fungsi pembacaan data dari smart contract, membuktikan transparansi dan aksesibilitas data yang tercatat pada blockchain
- Konektivitas antara sistem instrumentasi, server pengolah data, penyimpanan data, dan pencatatan di blockchain

BAB 4

WKESIMPULAN DAN REKOMENDASI

Sistem pemantauan suhu dan kelembaban berbasis sensor SHT20 yang terintegrasi dengan berbagai platform modern seperti Rust, InfluxDB, Grafana, PyQt, dan teknologi blockchain, telah menunjukkan kinerja yang efektif dan andal dalam menjaga mutu buah-buahan selama proses distribusi. Penggunaan sensor dengan tingkat akurasi tinggi memungkinkan pengukuran kondisi lingkungan secara real-time. Sementara itu, integrasi dengan basis data time-series InfluxDB serta visualisasi melalui dashboard Grafana memberikan kemudahan bagi operator maupun manajemen dalam memantau kondisi secara komprehensif.

Pengembangan aplikasi desktop menggunakan PyQt memberikan fleksibilitas tambahan dalam pemantauan lokal, serta didukung oleh fitur alarm otomatis dan pencatatan data historis yang memperkuat sistem dalam aspek pengendalian mutu. Lebih lanjut, penerapan teknologi blockchain melalui smart contract di jaringan Ethereum menjamin integritas, transparansi, dan keterlacakan data yang tidak dapat diubah, yang sangat krusial dalam rantai pasok produk segar.

Dengan berbagai keunggulan tersebut, sistem ini direkomendasikan untuk diimplementasikan secara nyata dalam distribusi buah-buahan maupun produk hortikultura lainnya. Sinergi antara teknologi Internet of Things (IoT) dan blockchain tidak hanya meningkatkan efisiensi serta ketelitian pemantauan, tetapi juga membuka peluang untuk penerapan sertifikasi mutu berbasis data serta peningkatan kepercayaan konsumen. Ke depan, pengembangan sistem dapat diarahkan pada penerapan teknologi *edge computing* guna pemrosesan data secara lokal, serta integrasi *machine learning* untuk mendeteksi potensi kerusakan produk secara dini.