



**RANCANG BANGUN SISTEM OTOMATISASI  
HIDROPONIK BERBASIS *INTERNET OF THINGS*  
(IOT)**

**Sub Judul**

**ANALISIS KINERJA PROTOKOL HTTP, MQTT,  
DAN WEBSOCKET PADA SISTEM MONITORING  
KUALITAS AIR HIDROPONIK BERBASIS VPS  
MENGUNAKAN STANDAR QOS TIPHON**

**SKRIPSI**

**Kevin Alonzo Manuel Bakara - 2207421032**

**PROGRAM STUDI TEKNIK MULTIMEDIA DAN JARINGAN  
JURUSAN TEKNIK INFORMATIKA DAN KOMPUTER  
POLITEKNIK NEGERI JAKARTA**

**2026**



**RANCANG BANGUN SISTEM OTOMATISASI  
HIDROPONIK BERBASIS *INTERNET OF THINGS*  
(IOT)**

**Sub Judul**

**ANALISIS KINERJA PROTOKOL HTTP, MQTT,  
DAN WEBSOCKET PADA SISTEM MONITORING  
KUALITAS AIR HIDROPONIK BERBASIS VPS  
MENGUNAKAN STANDAR QOS TIPHON**

**SKRIPSI**

**Dibuat untuk Melengkapi Syarat-Syarat yang Diperlukan untuk  
Memperoleh Diploma Empat Politeknik**

**Kevin Alonzo Manuel Bakara - 2207421032**

**PROGRAM STUDI TEKNIK MULTIMEDIA DAN JARINGAN  
JURUSAN TEKNIK INFORMATIKA DAN KOMPUTER  
POLITEKNIK NEGERI JAKARTA**

**2026**



## SURAT PERNYATAAN BEBAS PLAGIARISME

Saya yang bertanda tangan di bawah ini:

Nama

: Kevin Alonzo Manuel Bakara

NIM

: 2207421032

Jurusan / Program Studi

: Teknik Informatika dan Komputer / Teknik  
Multimedia dan Jaringan

Judul Skripsi

: ANALISIS KINERJA PROTOKOL HTTP, MQTT,  
AN WEBSOCKET PADA SISTEM MONITORING KUALITAS AIR  
HIDROPONIK BERBASIS VPS MENGGUNAKAN STANDAR QOS TIPHON

Menyatakan dengan sebenarnya bahwa skripsi ini benar-benar merupakan hasil karya saya sendiri, bebas dari peniruan terhadap karya dari orang lain. Kutipan pendapat dan tulisan orang lain ditunjuk sesuai dengan cara-cara penulisan karya ilmiah yang berlaku.

Apabila di kemudian hari terbukti atau dapat dibuktikan bahwa dalam skripsi ini terkandung ciri-ciri plagiat dan bentuk-bentuk peniruan lain yang dianggap melanggar peraturan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Depok, 28 Juni 2026

Yang membuat pernyataan



(Kevin Alonzo Manuel Bakara)

NIM. 2207421032

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :

a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.

b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta

2. Dilarang menggunakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta





© Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

## LEMBAR PENGESAHAN

Skripsi diajukan oleh:

Nama : Kevin Alonzo Manuel Bakara  
NIM : 2207421032  
Program Studi : Teknik Multimedia dan Jaringan  
Judul Skripsi : Analisis Kinerja Protokol HTTP, MQTT, dan WebSocket pada Sistem Monitoring Kualitas Air Hidroponik Berbasis VPS Menggunakan Standar QoS TIPHON

Telah diuji oleh tim penguji dalam Sidang Skripsi pada hari Rabu, Tanggal 1, Bulan Juli, Tahun 2024, dan dinyatakan LULUS.

Disahkan Oleh

Pembimbing I

: Maria Agustin, S.Kom., M.Kom.

Penguji I

: Dr. Indra Hermawan, M.Kom.

Penguji II

: Defiana Arnaldy, S.Tp., M.Si.

Penguji III

: Dwi Ermawati, MT.

Mengetahui:

Jurusan Teknik Informatika dan Komputer

Ketua



Dr. Anita Hidayati, S.Kom., M.Kom.

NIP.197908032003122003

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta





## KATA PENGANTAR

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :

a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.

b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta

2. Dilarang mengumpukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Segala puji dan syukur penulis panjatkan ke Tuhan Yesus Kristus atas segala limpahan rahmat, nikmat, dan pertolongan-Nya sehingga penulis dapat menyelesaikan Tugas Akhir berjudul “Analisis Kinerja Protokol HTTP, MQTT, dan WebSocket pada Sistem Monitoring Kualitas Air Hidroponik Berbasis VPS Menggunakan Standar QoS TIPHON” sebagai syarat untuk meraih gelar Sarjana Terapan dan menyelesaikan Diploma Empat di Politeknik Negeri Jakarta. Selama proses penyusunan skripsi ini, penulis mendapatkan banyak sekali dukungan, bantuan, dan arahan. Oleh karena itu, penulis ingin menyampaikan terima kasih yang sebesar-besarnya kepada:

1. Ibu Maria Agustin, S.Kom., M.Kom., selaku dosen pembimbing yang telah banyak meluangkan waktu, memberikan masukan, dan membimbing penulis dari awal hingga akhir.
2. Kedua orang tua yang selalu memberikan doa, cinta, dan selalu menguatkan penulis di setiap langkah.
3. Para sahabat dan tim Hidroponik yang terus mendukung, baik dalam bentuk dukungan moral, ide, maupun sekedar menemani di tengah gundah dan lelah.
4. Teman-teman seperjuangan di TMJ’22 yang telah menjadi bagian penting dalam perjalanan penulis selama mengemban ilmu di Politeknik Negeri Jakarta.

Penulis berharap tugas akhir ini dapat memberi manfaat dan menjadi kontribusi dalam pengembangan ilmu pengetahuan. Penulis juga menyadari masih banyak kekurangan dalam penelitian ini. Oleh karena itu, saran dan kritik yang membangun sangat diterima.

Depok, 28 Juni 2026

Kevin Alonzo Manuel Bakara





© Hak Cipta milik Jurusan Teknik Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

## SURAT PERNYATAAN PERSETUJUAN PUBLIKASI SKRIPSI UNTUK KEPENTINGAN AKADEMIS

Saya sivitas akademik Politeknik Negeri Jakarta, saya bertanda tangan di bawah

ini:

Nama

: Kevin Alonzo Manuel Bakara

NIM

: 2207421032

Jurusan / Program Studi

: Teknik Informatika dan Komputer / Teknik  
Multimedia dan Jaringan

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Politeknik Negeri Jakarta Hak Bebas Royalti Non-Eksklusif atas karya ilmiah saya yang berjudul :

**ANALISIS KINERJA PROTOKOL HTTP, MQTT, DAN WEBSOCKET  
PADA SISTEM MONITORING KUALITAS AIR HIDROPONIK  
BERBASIS VPS MENGGUNAKAN STANDAR QOS TIPHON**

Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Non-Eksklusif ini Politeknik Negeri Jakarta Berhak menyimpan, mengalihmediakan/formatkan, mengelola dalam bentuk basis data (*database*), merawat, dan mempublikasikan skripsi saya tanpa meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Depok, 28 Juni 2026

Yang membuat pernyataan



(Kevin Alonzo Manuel Bakara)

NIM. 2207421032





**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

# ANALISIS KINERJA PROTOKOL HTTP, MQTT, DAN WEBSOCKET PADA SISTEM MONITORING KUALITAS AIR HIDROPONIK BERBASIS VPS MENGGUNAKAN STANDAR QOS TIPHON

## ABSTRAK

Sistem pemantauan hidroponik Internet of Things (IoT) umumnya bergantung pada cloud pihak ketiga tanpa evaluasi kinerja objektif. Untuk mengatasi masalah tersebut, penelitian ini bertujuan mengimplementasikan sistem pemantauan kualitas air hidroponik bayam berbasis Virtual Private Server (VPS) mandiri menggunakan mikrokontroler ESP32 dan algoritma Rule-Based. Metode yang digunakan meliputi evaluasi komparatif kinerja protokol HTTP, MQTT, dan WebSocket menggunakan Wireshark berdasarkan standar QoS TIPHON (throughput, delay, packet loss, jitter). Selain itu, reliabilitas server diuji menggunakan Apache JMeter dan Soak Testing berdasarkan standar ISO/IEC 25010:2023, serta kelayakan antarmuka dievaluasi melalui User Acceptance Test (UAT). Hasil penelitian menunjukkan MQTT memiliki throughput 526,43 bps dan delay 178,36 ms, HTTP mencapai 10.110,19 bps dan 572,75 ms, sedangkan WebSocket 1.655,64 bps dan 188,90 ms, dengan seluruh protokol mencatat packet loss 0% (kategori "Sangat Bagus"). Server terbukti stabil tanpa memory leak dengan utilisasi CPU 1,78% selama 6 jam, algoritma Rule-Based mencapai akurasi 100%, dan keberhasilan UAT mencapai 100%. Kesimpulannya, MQTT direkomendasikan untuk transmisi data asinkron karena efisiensi bandwidth, sedangkan WebSocket unggul untuk pembaruan data real-time.

*Kata kunci : Internet of things (IoT), MQTT, Quality of Service, TIPHON, Virtual Private Server (VPS), WebSocket*





## DAFTAR ISI

|                                                                         |     |
|-------------------------------------------------------------------------|-----|
| SURAT PERNYATAAN BEBAS PLAGIARISME .....                                | ii  |
| LEMBAR PENGESAHAN.....                                                  | iii |
| KATA PENGANTAR .....                                                    | iv  |
| SURAT PERNYATAAN PERSETUJUAN PUBLIKASI.....                             | v   |
| SKRIPSI UNTUK KEPENTINGAN AKADEMIS .....                                | v   |
| ABSTRAK .....                                                           | vi  |
| DAFTAR ISI .....                                                        | vi  |
| DAFTAR GAMBAR .....                                                     | ix  |
| DAFTAR TABEL .....                                                      | x   |
| BAB I PENDAHULUAN .....                                                 | 1   |
| 1. 1 Latar Belakang .....                                               | 1   |
| 1. 2 Perumusan Masalah.....                                             | 2   |
| 1. 3 Batasan Masalah.....                                               | 3   |
| 1. 4 Tujuan dan Manfaat.....                                            | 4   |
| 1. 5 Sistematika Penulisan.....                                         | 5   |
| BAB II TINJAUAN PUSTAKA .....                                           | 7   |
| 2. 1 Tinjauan Pustaka .....                                             | 7   |
| 2. 2 Studi Literatur .....                                              | 11  |
| 2. 2. 1 Sistem Hidroponik.....                                          | 11  |
| 2. 2. 2 <i>Internet of things</i> (IoT).....                            | 12  |
| 2. 2. 3 <i>Smart Farming</i> .....                                      | 13  |
| 2. 2. 4 Protokol Hypertext Transfer Protocol (HTTP).....                | 13  |
| 2. 2. 5 Protokol <i>Message Queuing Telemetry Transport</i> (MQTT)..... | 13  |
| 2. 2. 6 Protokol WebSocket.....                                         | 14  |
| 2. 2. 7 Perbandingan Teoretis Protokol Berdasarkan Parameter QoS .....  | 14  |
| 2. 2. 8 <i>Broker Eclipse Mosquitto</i> .....                           | 15  |
| 2. 2. 9 <i>Blynk Cloud</i> .....                                        | 16  |
| 2. 2. 10 <i>Node.js</i> .....                                           | 16  |
| 2. 2. 11 <i>Cloud Firebase</i> .....                                    | 17  |
| 2. 2. 12 <i>Virtual Private Server</i> (VPS).....                       | 18  |
| 2. 2. 13 Kontainerisasi Docker.....                                     | 18  |

**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta



**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|                                   |                                                          |    |
|-----------------------------------|----------------------------------------------------------|----|
| 2. 2. 14                          | Algoritma <i>Rule-Based</i> .....                        | 19 |
| 2. 2. 15                          | Presisi.....                                             | 20 |
| 2. 2. 16                          | Wireshark .....                                          | 20 |
| 2. 2. 17                          | <i>User Acceptance Test (UAT)</i> .....                  | 21 |
| 2. 2. 18                          | Pengujian Sistem .....                                   | 21 |
| BAB III METODE PENELITIAN .....   |                                                          | 28 |
| 3. 1                              | Rancangan Penelitian .....                               | 28 |
| 3. 2                              | Tahapan Penelitian .....                                 | 29 |
| 3. 3                              | Objek Penelitian .....                                   | 31 |
| BAB IV HASIL DAN PEMBAHASAN ..... |                                                          | 33 |
| 4. 1                              | Analisis Kebutuhan .....                                 | 33 |
| 4. 1. 1                           | Kebutuhan Fungsionalitas.....                            | 33 |
| 4. 1. 2                           | Kebutuhan Non Fungsionalitas .....                       | 35 |
| 4. 2                              | Perancangan Sistem.....                                  | 37 |
| 4. 2. 1                           | Blok Diagram Sistem.....                                 | 38 |
| 4. 2. 2                           | Topologi Jaringan Sistem.....                            | 39 |
| 4. 2. 3                           | Diagram Alur Sistem .....                                | 41 |
| 4. 2. 4                           | Alur Komunikasi Data .....                               | 43 |
| 4. 2. 5                           | Perancangan Algoritma <i>Rule-Based</i> .....            | 43 |
| 4. 2. 6                           | Perancangan Basis Data .....                             | 44 |
| 4. 2. 7                           | Perancangan <i>Dashboard Web</i> .....                   | 46 |
| 4. 3                              | Implementasi Sistem .....                                | 49 |
| 4. 3. 1                           | Kebutuhan <i>Hardware</i> .....                          | 50 |
| 4. 3. 2                           | Kebutuhan <i>Software</i> .....                          | 50 |
| 4. 3. 3                           | Implementasi Perangkat Keras .....                       | 52 |
| 4. 3. 4                           | Implementasi <i>Backend</i> dan Orkestrasi Protokol..... | 53 |
| 4. 3. 5                           | Implementasi <i>Frontend</i> .....                       | 56 |
| 4. 3. 6                           | Implementasi Infrastruktur <i>Cloud</i> .....            | 60 |
| 4. 4                              | Pengujian .....                                          | 61 |
| 4. 4. 1                           | Deskripsi Pengujian .....                                | 61 |
| 4. 4. 2                           | Prosedur Pengujian .....                                 | 62 |
| 4. 4. 3                           | Data Hasil Pengujian .....                               | 67 |





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|                                 |     |
|---------------------------------|-----|
| 4. 4. 4 Analisis Data .....     | 91  |
| BAB V KESIMPULAN DAN SARAN..... | 101 |
| 5.1. Kesimpulan .....           | 101 |
| 5.2. Saran .....                | 102 |
| DAFTAR PUSTAKA.....             | 103 |
| DAFTAR RIWAYAT HIDUP .....      | 108 |
| LAMPIRAN .....                  | 109 |





**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

## DAFTAR GAMBAR

|                                                                            |    |
|----------------------------------------------------------------------------|----|
| Gambar 2. 1 WebSocket .....                                                | 14 |
| Gambar 2. 2 Eclipse MQTT .....                                             | 16 |
| Gambar 2. 3 Blynk Cloud .....                                              | 16 |
| Gambar 2. 4 Node Js .....                                                  | 17 |
| Gambar 2. 5 Google Cloud Firebase .....                                    | 18 |
| Gambar 3. 1 Tahapan Penelitian .....                                       | 29 |
| Gambar 4. 1 Diagram Blok Sistem .....                                      | 38 |
| Gambar 4. 2 Arsitektur Jaringan Sistem .....                               | 39 |
| Gambar 4. 3 Flowchart Kerja Sistem .....                                   | 41 |
| Gambar 4. 4 Koleksi History .....                                          | 45 |
| Gambar 4. 5 Koleksi Users .....                                            | 46 |
| Gambar 4. 6 Tampilan Halaman Login .....                                   | 47 |
| Gambar 4. 7 Tampilan Halaman Dashboard .....                               | 48 |
| Gambar 4. 8 Penempatan Node Sensor IoT pada Media Hidroponik .....         | 53 |
| Gambar 4. 9 Struktur File Backend .....                                    | 54 |
| Gambar 4. 10 Implementasi Perhitungan <i>Rule-Based</i> pada Node.js ..... | 55 |
| Gambar 4. 11 Implementasi Arsitektur Gatekeeper pada Node.js .....         | 56 |
| Gambar 4. 12 Halaman Login .....                                           | 57 |
| Gambar 4. 13 Dashboard Utama .....                                         | 57 |
| Gambar 4. 14 Grafik pada Dashboard .....                                   | 58 |
| Gambar 4. 15 Tabel History pada Dashboard .....                            | 59 |
| Gambar 4. 16 Antarmuka Konfigurasi Push Notification Telegram Bot .....    | 59 |
| Gambar 4. 17 Overview VM GCP .....                                         | 60 |
| Gambar 4. 18 Hasil Konfigurasi Firewall Rule VM .....                      | 61 |
| Gambar 4. 19 Summary Report Apache JMeter (Skenario 200 Koneksi) .....     | 83 |
| Gambar 4. 20 Grafik Hasil Pengujian Fungsionalitas .....                   | 91 |





## DAFTAR TABEL

|                                                                     |    |
|---------------------------------------------------------------------|----|
| Tabel 2. 1 Tinjauan Pustaka.....                                    | 7  |
| Tabel 2. 2 Klasifikasi Nilai Quality of Service (TIPHON) .....      | 22 |
| Tabel 2. 3 Klasifikasi Nilai Standar QoS Berdasarkan TIPHON .....   | 24 |
| Tabel 4. 1 Kebutuhan Fungsionalitas Sistem .....                    | 33 |
| Tabel 4. 2 Kebutuhan Non Fungsionalitas Sistem.....                 | 35 |
| Tabel 4. 3 Aturan Algoritma <i>Rule-Based</i> .....                 | 44 |
| Tabel 4. 4 Kebutuhan Hardware .....                                 | 50 |
| Tabel 4. 5 Kebutuhan Software .....                                 | 51 |
| Tabel 4. 6 Pengujian Fungsionalitas Sistem.....                     | 68 |
| Tabel 4. 7 Pengujian Algoritma <i>Rule-Based</i> .....              | 70 |
| Tabel 4. 8 Hasil Pengujian Throughput Protokol MQTT .....           | 72 |
| Tabel 4. 9 Hasil Pengujian Packet loss Protokol MQTT .....          | 73 |
| Tabel 4. 10 Hasil Pengujian Delay Protokol MQTT .....               | 73 |
| Tabel 4. 11 Hasil Pengujian Jitter Protokol MQTT .....              | 74 |
| Tabel 4. 12 Hasil Pengujian Throughput Protokol HTTP.....           | 75 |
| Tabel 4. 13 Hasil Pengujian Packet loss Protokol HTTP.....          | 76 |
| Tabel 4. 14 Hasil Pengujian Delay Protokol HTTP.....                | 77 |
| Tabel 4. 15 Hasil Pengujian Jitter Protokol HTTP .....              | 78 |
| Tabel 4. 16 Hasil Pengujian Throughput Protokol WebSocket.....      | 79 |
| Tabel 4. 17 Hasil Pengujian Packet loss Protokol WebSocket.....     | 79 |
| Tabel 4. 18 Hasil Pengujian Delay Protokol WebSocket .....          | 80 |
| Tabel 4. 19 Hasil Pengujian Jitter Protokol WebSocket.....          | 81 |
| Tabel 4. 20 Hasil Pengujian Load testing Server VPS.....            | 82 |
| Tabel 4. 21 Rekapitulasi Metrik Soak testing Selama 6 Jam .....     | 84 |
| Tabel 4. 22 Hasil Pengujian Daya Pulih Sistem (Recoverability)..... | 86 |
| Tabel 4. 23 Hasil Pengujian Fungsional Sistem (UAT) .....           | 88 |
| Tabel 4. 24 Penilaian Kepuasan Pengguna.....                        | 90 |
| Tabel 4. 23 Analisis QoS.....                                       | 94 |

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta



**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

## BAB I PENDAHULUAN

### 1. Latar Belakang

Penerapan teknologi *Internet of Things* (IoT) pada *Smart Farming* hidroponik secara esensial mengoptimalkan efisiensi pemanfaatan lahan dan sumber daya (Rizal dkk., 2024). Secara teknis, integrasi sensor lingkungan *real-time* dapat mengendalikan perangkat irigasi dan nutrisi secara otomatis (Halawa, 2024). Otomatisasi ini krusial untuk meminimalisasi pemborosan *input* dan mendongkrak kualitas panen. Meskipun menjanjikan, ekosistem hidroponik memiliki kerentanan struktural akibat ketiadaan tanah sebagai penyeimbang alami (*buffer*). Keberhasilan panen sangat bergantung pada presisi pemeliharaan pH dan nutrisi (TDS). Deviasi dari rentang optimal berisiko fatal menghambat absorpsi hara tanaman (Zakiah dkk., 2025). Akibatnya, kehadiran sistem *monitoring real-time* menjadi syarat mutlak untuk memfasilitasi tindakan korektif seketika.

Kondisi ideal tersebut berbeda dengan kenyataan di lapangan. Berdasarkan hasil wawancara dengan pengelola *greenhouse* hidroponik Lokatani, diketahui bahwa pemantauan kualitas air (pH dan TDS) masih dilakukan secara manual menggunakan alat ukur biasa. Cara manual ini dinilai kurang efisien karena mengharuskan pekerja turun langsung untuk mengecek air secara berkala. Selain itu, metode ini sering kali menghasilkan data yang kurang akurat akibat kelalaian manusia (*human error*). Keterbatasan ini memunculkan kebutuhan mendesak bagi pengelola Lokatani untuk memiliki sistem IoT yang bekerja otomatis dan seketika (*real-time*). Sistem ini diharapkan dapat mempermudah pekerjaan sehari-hari dan meminimalisasi risiko gagal panen akibat keterlambatan penanganan kualitas air yang buruk.

Saat ini, kebanyakan sistem pemantauan IoT pertanian masih menumpang pada layanan *cloud* milik pihak ketiga seperti Blynk, dan tidak menyertakan pengujian kualitas jaringan (*Quality of Service / QoS*) secara jelas. Mengandalkan *cloud* publik seperti ini sering kali menyebabkan data terlambat sampai (*delay*), yang bisa berakibat fatal untuk sistem *real-time*. Hal ini dibuktikan oleh penelitian Syukhron





### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

dkk. (2021) yang menemukan bahwa rata-rata keterlambatan pada Blynk mencapai 3.400 ms. Menurut standar telekomunikasi TIPHON, angka tersebut masuk kategori "Jelek" karena jauh melewati batas ideal yaitu 150 ms. Keterlambatan ini menjadi bukti nyata kelemahan layanan *cloud* publik untuk kebutuhan *real-time*. Oleh karena itu, solusi yang tepat adalah membuat server mandiri menggunakan *Virtual Private Server* (VPS).

Namun, setelah sistem dipindahkan ke VPS mandiri, muncul tantangan baru terkait pemilihan protokol komunikasi yang paling tepat untuk digunakan. Banyak pengembang IoT yang beralih ke VPS mandiri, tetapi mereka sering kali langsung memilih protokol seperti HTTP atau MQTT tanpa menguji kualitas layanannya (QoS). Padahal, penelitian sebelumnya menunjukkan hasil yang berbeda-beda. Prasetyoaji dan Anifah (2025) membuktikan bahwa MQTT cukup stabil untuk sistem hidroponik. Di sisi lain, Nugraha dkk. (2024) menemukan bahwa MQTT lebih hemat kinerja mesin, tetapi HTTP lebih pasti dalam mengirimkan data. Hasil ini menunjukkan bahwa tidak ada satu pun protokol yang paling sempurna di segala sisi, sehingga kita perlu membandingkannya secara langsung.

Oleh karena itu, penelitian ini dilakukan karena belum ada riset yang membandingkan tiga protokol sekaligus HTTP, MQTT, dan WebSocket pada server VPS mandiri khusus untuk sistem hidroponik. Protokol WebSocket sendiri, yang bisa mengirim dan menerima data secara bersamaan dan terus-menerus tanpa harus saling menunggu (*request-response* berulang), belum pernah ikut dievaluasi dalam kasus ini. Penelitian ini bertujuan membandingkan kinerja ketiga protokol tersebut secara menyeluruh menggunakan aplikasi Wireshark. Parameter yang diukur meliputi *delay* (keterlambatan), *throughput* (kecepatan data), *packet loss* (data yang hilang), dan *jitter* (variasi keterlambatan). Pengujian ini sangat penting agar pemilihan protokol jaringan untuk VPS hidroponik mandiri bukan sekadar tebakan, melainkan benar-benar terbukti memenuhi standar kualitas TIPHON.

## 1.2 Perumusan Masalah

Perumusan masalah yang didapat berdasarkan latar belakang yang telah disampaikan:



## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

1. Bagaimana merancang dan mengimplementasikan mekanisme komunikasi data menggunakan protokol HTTP, MQTT, dan WebSocket pada infrastruktur *server* mandiri untuk sistem *monitoring* hidroponik presisi berbasis IoT?
2. Bagaimana karakteristik kinerja transmisi data protokol HTTP, MQTT, dan WebSocket pada infrastruktur *server* mandiri sistem monitoring hidroponik berbasis IoT, berdasarkan parameter *Quality of Service* (QoS) yang meliputi *delay*, *throughput*, *packet loss*, dan *jitter*?
3. Protokol manakah yang paling memenuhi standar reliabilitas TIPHON dan paling optimal untuk diimplementasikan pada sistem monitoring hidroponik presisi berbasis IoT?

### 1.3 Batasan Masalah

Beberapa batasan masalah yang dibatasi untuk memfokuskan dan mengfisiensi waktu penelitian di antaranya yaitu:

1. Penelitian difokuskan pada sistem pemantauan (*monitoring*) hidroponik berbasis *Internet of things* (IoT) dan tidak mencakup fungsi pengendalian (*control*) seperti pengaturan pompa, pemberian nutrisi, atau pengaturan pH secara otomatis.
2. Data yang dikirimkan dan dievaluasi dalam pengujian QoS merupakan *payload* sensor kualitas larutan hidroponik, mencakup parameter derajat keasaman (pH) dan kepekatan nutrisi terlarut (TDS/PPM) dalam format JSON. Logika penentuan status kualitas air menggunakan algoritma *Rule-Based* (IF-THEN) yang dieksekusi sepenuhnya di sisi *server backend*.
3. Protokol komunikasi yang dianalisis dibatasi pada tiga protokol, yaitu HTTP, MQTT, dan WebSocket, yang diimplementasikan pada infrastruktur VPS mandiri berbasis Linux yang sama untuk memastikan kondisi pengujian yang setara dan objektif. *Server backend* Node.js berperan sebagai jembatan komunikasi yang mengintegrasikan ketiga protokol tersebut, dengan broker Eclipse Mosquitto dijalankan secara mandiri (*self-managed*) pada *port* standar 1883.





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

4. Manajemen penyimpanan data historis memanfaatkan layanan Cloud Firestore dengan struktur data dokumen (NoSQL) tanpa menggunakan basis data relasional.
5. Evaluasi kinerja komunikasi data dilakukan menggunakan parameter *Quality of Service* (QoS) standar TIPHON yang meliputi *delay*, *throughput*, *packet loss*, dan *jitter*, serta pengujian beban komputasi internal (*load testing*) menggunakan Apache JMeter pada infrastruktur VPS.
6. Pengujian metrik QoS dilakukan secara terisolasi menggunakan medium jaringan nirkabel lokal (Wi-Fi Hotspot) yang dibatasi pada kapasitas *bandwidth* operasional di kisaran 20 Mbps. Penelitian ini tidak mengukur penurunan kinerja yang diakibatkan oleh fluktuasi delay eksternal dari *Internet Service Provider* (ISP) di luar skenario jaringan tersebut.
7. Penelitian tidak membahas implementasi keamanan jaringan pada lapisan transpor (seperti enkripsi sertifikat SSL/TLS), serta tidak mengevaluasi profil konsumsi daya pada perangkat mikrokontroler.

### 1.4 Tujuan dan Manfaat

Tujuan penelitian ini adalah:

1. Merancang dan mengimplementasikan sistem *monitoring* hidroponik presisi berbasis IoT pada infrastruktur *Virtual Private Server* (VPS) mandiri yang mengintegrasikan protokol HTTP, MQTT, dan WebSocket sebagai objek evaluasi komparatif dalam satu kerangka pengujian yang terpadu dan setara.
2. Menganalisis dan membandingkan karakteristik kinerja transmisi data protokol HTTP, MQTT, dan WebSocket secara *end-to-end* berdasarkan parameter *Quality of Service* (QoS) standar TIPHON yang meliputi *delay*, *throughput*, *packet loss*, dan *jitter* menggunakan Wireshark, serta mengukur ketahanan infrastruktur *server* terhadap beban transmisi menggunakan Apache JMeter.
3. Menetapkan protokol komunikasi yang paling optimal dan paling memenuhi standar reliabilitas TIPHON untuk diimplementasikan pada sistem monitoring hidroponik berbasis IoT berdasarkan hasil analisis kuantitatif komparatif.

Adapun manfaat penelitian ini adalah:



## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

1. Menghasilkan rancangan sistem monitoring hidroponik di VPS mandiri yang dapat digunakan sebagai contoh pengujian berbagai protokol IoT.
2. Menghasilkan data *benchmark* kinerja jaringan dan batas ketahanan server yang objektif, guna membantu pengembang sistem IoT menghindari pemborosan spesifikasi server.
3. Memberikan rekomendasi pemilihan protokol pengiriman data yang paling reliabel berdasarkan parameter QoS TIPHON untuk implementasi sistem pertanian presisi.

### 1.5 Sistematika Penulisan

Penyusunan proposal penelitian ini dilakukan dengan sistematika yang terstruktur guna menjamin alur pembahasan yang sistematis, koheren, dan mudah dipahami. Adapun pembagian bab dalam laporan penelitian ini adalah sebagai berikut:

BAB I PENDAHULUAN Menjelaskan fondasi penelitian yang mencakup latar belakang pemilihan topik, rumusan masalah yang akan diselesaikan, batasan ruang lingkup penelitian, tujuan serta manfaat yang ingin dicapai, hingga sistematika penyusunan laporan.

BAB II TINJAUAN PUSTAKA Menguraikan landasan teori, teknologi pengembangan aplikasi berbasis web yang diterapkan, serta tinjauan terhadap penelitian terdahulu yang relevan sebagai acuan pengembangan sistem.

BAB III METODE PENELITIAN Menjelaskan metodologi penelitian yang digunakan, meliputi tahapan perancangan arsitektur perangkat lunak, desain antarmuka pengguna, teknik pengumpulan data, serta prosedur pengujian dan metode analisis yang diterapkan dalam pengembangan sistem.

BAB IV HASIL DAN PEMBAHASAN Memaparkan hasil realisasi penelitian yang meliputi analisis kebutuhan perangkat keras dan perangkat lunak, perancangan arsitektur sistem *Client-Server*, implementasi algoritma *Rule-Based* pada sisi *backend*, implementasi antarmuka *dashboard* berbasis React.js, serta hasil dan analisis pengujian fungsionalitas, algoritma, dan kinerja jaringan (*Quality of Service*) berdasarkan standar TIPHON.





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

BAB V KESIMPULAN Menyajikan kesimpulan yang diperoleh dari seluruh rangkaian penelitian sebagai jawaban atas rumusan masalah yang telah ditetapkan, serta memberikan saran pengembangan yang relevan untuk penelitian lanjutan berdasarkan temuan dan keterbatasan sistem yang telah dibangun.



### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

## BAB II

### TINJAUAN PUSTAKA

#### 2. 1 Tinjauan Pustaka

Bagian ini memaparkan rangkuman penelitian terdahulu yang relevan dengan pengembangan sistem *monitoring* pada *Smart Farming*. Tabel berikut menyajikan perbandingan aspek teknis untuk memperjelas evolusi teknologi dari sistem otomasi lokal hingga sistem berbasis IoT, sekaligus menyoroti celah masalah (*research gap*).

Tabel 2. 1 Tinjauan Pustaka

| Peneliti dan Tahun                                                              | Judul Penelitian                                                                                   | Teknologi dan Cara Kerja                                                                                                                                                      | Catatan                                                                                                                                                                                              |
|---------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Imam Syukhron, Reni Rahmadewi, dan Ibrahim (2021)                               | Penggunaan Aplikasi Blynk Untuk <i>Monitoring</i> dan Kontrol Jarak Jauh pada Sistem Kompos Pintar | Teknologi: Wemos D1R1, Aplikasi Blynk.<br><br>Cara Kerja: Data sensor dikirim melalui Wi-Fi ke <i>server cloud</i> Blynk untuk ditampilkan pada antarmuka smartphone Android. | Sistem <i>monitoring</i> berhasil dilakukan, namun ditemukan <i>delay</i> pengiriman data yang tinggi dengan rata-rata <i>delay</i> 3.400 ms yang berdasarkan standar TIPHON masuk kategori "jelek". |
| Muhamad Cahyo Ardi Prabowo, Atikah Ayu Janitra, dan Nisrina Mayla Wibowo (2023) | Sistem <i>Monitoring</i> Hidroponik Berbasis IoT Dengan                                            | Teknologi: ESP8266, NodeMCU, Website <i>monitoring</i> .                                                                                                                      | Sistem berhasil membangun <i>monitoring</i> berbasis web mandiri dengan akurasi sensor                                                                                                               |





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|                              |                                                                                                                    |                                                                                                                                                                                                                                                             |                                                                                                                                                                                                                                                    |
|------------------------------|--------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                              | Sensor Suhu, pH, dan Ketinggian Air Menggunakan ESP8266                                                            | Cara Kerja:<br>Sistem membaca data sensor pH dan ketinggian air, lalu mengirimkannya ke <i>database</i> untuk ditampilkan pada antarmuka website yang dibangun khusus (bukan platform instan).                                                              | yang baik, namun klaim kemampuan <i>real-time</i> belum divalidasi dengan pengukuran parameter QoS seperti <i>delay</i> , <i>throughput</i> , maupun <i>packet loss</i> , sehingga kinerja transmisi data aktualnya belum terukur secara objektif. |
| Safitri dan Priambodo (2023) | MQTT and CoAP Communication Protocol Analysis in <i>Internet of things</i> System for Strawberry Hydroponic Plants | Teknologi:<br>Raspberry Pi, Mosquitto MQTT Broker, protokol MQTT dan CoAP, Firebase Cloud.<br>Cara Kerja: Data sensor tanaman stroberi dikirimkan menggunakan dua protokol berbeda (MQTT dan CoAP) secara bersamaan, kemudian kinerja keduanya dibandingkan | Studi ini membuktikan bahwa MQTT unggul dalam <i>packet loss</i> dan konsistensi pengiriman pesan, sementara CoAP lebih efisien dalam <i>delay</i> dan konsumsi daya baterai. Namun penelitian tidak menyertakan protokol HTTP maupun WebSocket    |



## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|                               |                                                                                                                        |                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                        |
|-------------------------------|------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                               |                                                                                                                        | berdasarkan <i>packet loss, delay, CPU usage, dan konsumsi daya.</i>                                                                                                                                                                                                                                                    | sebagai pembanding, sehingga spektrum evaluasi protokol IoT belum lengkap untuk dijadikan landasan pemilihan protokol yang komprehensif.                                                                                                                                                                               |
| Prasetyoaji dan Anifah (2025) | <i>Design of Smart Hydroponics Based on Internet of things (IoT) Using Rule-Based System on Hydroponic Wick System</i> | <p>Teknologi: ESP32, Sensor TDS, Sensor pH, EMQX MQTT Broker, MQTT IoT Panel, Rule-Based System.</p> <p>Cara Kerja: Sensor TDS dan pH membaca kondisi larutan nutrisi secara bergantian, data diproses oleh ESP32 menggunakan algoritma <i>Rule-Based</i> untuk mengklasifikasikan kondisi tanaman dan mengaktifkan</p> | <p>Sistem berhasil membuktikan kinerja QoS MQTT yang stabil pada konteks hidroponik <i>throughput</i> rata-rata 36,38 kbps, <i>delay</i> 762,68 ms, <i>jitter</i> 26,91 ms, dan <i>packet loss</i> 0% pada QoS level 0. Namun evaluasi hanya mencakup satu protokol (MQTT) tanpa perbandingan terhadap HTTP maupun</p> |





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|                                    |                                                                          |                                                                                                                                                                                                                                                                                                                                                                     |                                                                                                                                                                                                                                                                                                                                                                                         |
|------------------------------------|--------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                    |                                                                          | pompa secara otomatis, kemudian dikirim ke smartphone via MQTT.                                                                                                                                                                                                                                                                                                     | WebSocket, sehingga belum dapat menentukan protokol yang paling optimal secara komparatif.                                                                                                                                                                                                                                                                                              |
| Nugraha, Putra dan Setiawan (2024) | A Comparative Study of HTTP and MQTT for IoT Applications in Hydroponics | <p>Teknologi: Google Cloud Platform (GCP), Virtual Machine, EMQX MQTT <i>Broker</i>, tools Autocannon (HTTP) dan MQTT Stresser.</p> <p>Cara Kerja: HTTP dan MQTT diuji secara komparatif menggunakan skenario peningkatan jumlah klien dan <i>payload</i> secara bertahap, dengan parameter CPU usage, transfer time, dan acceptance rate sebagai metrik utama.</p> | <p>Studi ini membuktikan MQTT jauh lebih efisien dalam penggunaan CPU (22,63% vs 87% pada 50 klien) dan kecepatan transfer (3,24 vs 177,344 detik untuk 100.000 <i>payload</i>), namun HTTP unggul dalam keandalan pengiriman (30.000 <i>payload</i> tanpa loss vs MQTT yang saturasi pada 1.680 <i>payload</i>). Penelitian ini belum menyertakan jenis protokol lainnya dan tidak</p> |



## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|  |  |  |                                                                                                                                                        |
|--|--|--|--------------------------------------------------------------------------------------------------------------------------------------------------------|
|  |  |  | menggunakan parameter QoS standar TIPHON ( <i>delay</i> , <i>jitter</i> , <i>packet loss</i> ) maupun pengukuran <i>end-to-end</i> berbasis Wireshark. |
|--|--|--|--------------------------------------------------------------------------------------------------------------------------------------------------------|

Uraian penelitian terdahulu pada Tabel 2.1 menegaskan pentingnya evaluasi komparatif yang lebih komprehensif pada arsitektur komunikasi IoT agrikultur. Mengingat penelitian sebelumnya umumnya mengevaluasi protokol secara parsial (Prasetyoaji dan Anifah, 2025) atau belum menggunakan parameter QoS berstandar telekomunikasi (Nugraha dkk., 2024), penelitian ini hadir untuk melengkapi celah tersebut. Pendekatan yang dilakukan difokuskan pada perbandingan kinerja tiga protokol utama HTTP, MQTT, dan WebSocket—dalam satu infrastruktur *Virtual Private Server* (VPS) mandiri. Evaluasi dieksekusi secara *end-to-end* menggunakan matriks *Quality of Service* (QoS) standar TIPHON beserta *load testing* ketahanan *server*, guna memberikan landasan empiris yang lebih utuh dalam penentuan arsitektur sistem *monitoring* hidroponik presisi.

## 2. 2 Studi Literatur

Bagian ini memaparkan kajian literatur terdahulu beserta landasan konseptual penelitian. Pembahasan mencakup karakteristik sistem hidroponik, arsitektur IoT, mekanisme kerja protokol komunikasi usulan (HTTP, MQTT, dan WebSocket), serta standar pengujian *Quality of Service* (QoS) TIPHON sebagai metrik evaluasi komparatif.

### 2. 2. 1 Sistem Hidroponik

Hidroponik merupakan metode budidaya tanaman yang memanfaatkan air sebagai media tanam utama menggantikan tanah. Prinsip utamanya adalah menyuplai





### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

nutrisi yang dibutuhkan tanaman langsung ke akar melalui larutan air (Wati dan Sholihah, 2021). Salah satu teknik yang populer adalah *Nutrient Film Technique* (NFT), di mana air nutrisi dialirkan secara terus-menerus dalam lapisan tipis agar akar tanaman memperoleh nutrisi sekaligus oksigen yang cukup.

Keberhasilan sistem hidroponik sangat bergantung pada kualitas air baku dan kestabilan parameter larutan. Dua parameter kunci yang harus dijaga adalah:

#### 1. Derajat Keasaman (pH)

Tingkat pH air memengaruhi kelarutan dan kemampuan akar dalam menyerap unsur hara, yang berdampak langsung pada kualitas pertumbuhan dan perkembangan tanaman (Irawan dkk., 2023). Pada budidaya bayam hidroponik, rentang pH air yang direkomendasikan agar tanaman tumbuh secara optimal berada pada level 6,0 hingga 7,0 (Zakiah dkk., 2025). Kondisi pH yang dipertahankan secara konstan pada rentang tersebut dapat mencegah terjadinya reaksi kimia negatif dalam larutan nutrisi yang berisiko menurunkan kualitas hasil panen.

#### 2. Nutrisi Terlarut (TDS)

*Total Dissolved Solids (TDS)* merupakan indikator konsentrasi zat atau mineral terlarut di dalam air yang diukur menggunakan satuan ppm (parts per million). Kebutuhan nutrisi setiap komoditas hidroponik berbeda secara eksak. Untuk mendukung efektivitas pertumbuhan pada tanaman bayam, sistem memerlukan kepekatan larutan nutrisi pada rentang 800 hingga 1600 ppm (Zakiah dkk., 2025). Nilai TDS yang berada di bawah atau melebihi ambang batas ini akan mengakibatkan tanaman mengalami defisiensi unsur hara atau keracunan nutrisi.

#### 2. 2. 2 *Internet of things (IoT)*

*Internet of things (IoT)* adalah konsep teknologi yang memungkinkan pengendalian dan *monitoring* perangkat dari jarak jauh melalui jaringan internet. Sebagaimana dijelaskan oleh Ferdiansyah dkk. (2025), IoT bekerja dengan menggabungkan sensor untuk mengambil data dari lingkungan fisik dan mikrokontroler untuk memproses serta mengirimkan data tersebut ke pengguna. Penerapan teknologi ini memungkinkan efisiensi kerja dapat ditingkatkan secara



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

signifikan tanpa batasan jarak, karena data kondisi lingkungan dapat dikirim secara *real-time* ke pusat data.

### 2. 2. 3 *Smart Farming*

*Smart Farming* merupakan penerapan teknologi modern pada sektor pertanian yang bertujuan untuk meningkatkan kualitas dan kuantitas produksi. Menurut Pratama dkk. (2024), pemanfaatan IoT untuk *monitoring* area pertanian menjadi salah satu pilar utama dalam *Smart Farming*. Tanah dan lingkungan sebagai faktor utama hortikultura harus diperhatikan sebaik-baiknya agar memberikan hasil sesuai harapan, dan teknologi ini memungkinkan hal tersebut dilakukan dengan presisi tinggi melalui pemrosesan kinerja *server* yang optimal.

### 2. 2. 4 Protokol Hypertext Transfer Protocol (HTTP)

HTTP (Hypertext Transfer Protocol) adalah protokol komunikasi yang menjadi fondasi pertukaran data pada arsitektur *client-server* berbasis web. HTTP beroperasi menggunakan mekanisme *request-response*, di mana klien harus menginisiasi setiap permintaan data secara eksplisit sebelum *server* memberikan respons (Nugraha, Putra, dan Setiawan, 2024). Setiap siklus komunikasi HTTP membawa header berukuran relatif besar yang memuat metadata koneksi, sehingga menghasilkan *overhead* transmisi yang lebih tinggi dibandingkan protokol IoT ringan lainnya. Meskipun demikian, HTTP tetap banyak digunakan dalam sistem monitoring IoT karena kompatibilitasnya yang universal dengan infrastruktur web dan kemudahannya dalam diintegrasikan dengan berbagai layanan *backend*.

### 2. 2. 5 Protokol Message Queuing Telemetry Transport (MQTT)

MQTT adalah protokol komunikasi yang digunakan dalam sistem IoT untuk pertukaran data antara perangkat keras dan *server*. Turnip dkk. (2023) menyatakan bahwa dalam pengembangan teknologi pertanian, metode komunikasi dan antarmuka online memerlukan peningkatan kinerja. Protokol MQTT dinilai efektif karena mampu menangani aliran komunikasi data (*communication flow*) dari ujung ke ujung (*end-to-end*) secara efisien, menjadikannya solusi ideal untuk arsitektur komunikasi *Smart Farming* dibandingkan metode konvensional.





#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

### 2. 2. 6 Protokol WebSocket

WebSocket adalah protokol komunikasi yang menyediakan saluran komunikasi dua arah (*full-duplex*) antara klien dan *server* melalui koneksi tunggal yang persisten. Nugraha (2024) menjelaskan bahwa protokol HTTP memiliki kelemahan di mana *server* tidak dapat mengirimkan data kepada klien jika klien tidak meminta (*request*) terlebih dahulu. Sebagai solusi, WebSocket memungkinkan pertukaran data secara *real-time* di mana data dapat dikirimkan oleh kedua belah pihak kapan saja. Hal ini sangat relevan untuk sistem *monitoring* yang membutuhkan pembaruan data instan tanpa jeda.



Gambar 2. 1 WebSocket

Sumber: (adroitgroup.io, 2023)

### 2. 2. 7 Perbandingan Teoretis Protokol Berdasarkan Parameter QoS

Meskipun HTTP, MQTT, dan WebSocket beroperasi pada lapisan aplikasi di atas protokol TCP/IP, ketiganya memiliki arsitektur yang menghasilkan karakteristik kinerja (*Quality of Service/QoS*) yang sangat berbeda. Dari parameter konsumsi *throughput*, HTTP beroperasi dengan model *request-response* yang membawa *header* teks berukuran besar pada setiap transaksinya, sehingga menghasilkan beban tambahan (*overhead*) yang tinggi dan menuntut kapasitas *bandwidth* yang besar (Nugraha, Putra, dan Setiawan, 2024). Sebaliknya, MQTT dirancang khusus untuk ekosistem IoT dengan mengompresi *header* hingga ukuran minimal 2 bita, menjadikannya sangat efisien dan hemat *throughput* pada perangkat bersumber daya rendah (Turnip dkk., 2023). WebSocket juga menawarkan efisiensi *throughput* yang tinggi karena setelah proses jabat tangan (*handshake*) awal selesai, komunikasi berubah menjadi aliran data biner yang ringan tanpa perlu mengirimkan informasi tambahan secara berulang (Nugraha, 2024).



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Dari aspek *delay* dan *jitter*, arsitektur *stateless* pada HTTP mengharuskan pembukaan dan penutupan koneksi TCP berulang kali pada setiap siklus pengiriman data. Karakteristik ini secara teoretis memicu lonjakan *delay* serta fluktuasi waktu kedatangan paket (*jitter*) yang tinggi, sehingga kurang ideal untuk sistem pemantauan secara *real-time*. Sebagai solusinya, MQTT dan WebSocket mempertahankan koneksi secara terus-menerus (*stateful*). WebSocket memfasilitasi komunikasi dua arah secara penuh (*full-duplex*) yang memungkinkan pertukaran data seketika tanpa harus menunggu permintaan (*request*) dari sistem (Nugraha, 2024). Sementara itu, MQTT menggunakan arsitektur *publish-subscribe* yang memangkas waktu tempuh rute secara optimal dari perangkat sensor ke *server* (Turnip dkk., 2023). Koneksi yang terus terhubung pada kedua protokol ini secara mendasar menstabilkan antrean data di dalam jaringan, sehingga mampu menekan angka *jitter* ke tingkat yang sangat minimal.

Terkait parameter *packet loss*, ketiga protokol ini berjalan di atas lapisan transpor TCP yang secara fundamental memiliki mekanisme pengiriman ulang otomatis (*retransmission*) untuk mencegah hilangnya paket di tengah jalan. Namun, MQTT memiliki keunggulan khusus untuk mengatasi masalah hilangnya data pada perangkat IoT melalui penerapan level *Quality of Service* bawaan, yaitu QoS 0, 1, dan 2 (Prasetyoaji dan Anifah, 2025). Fitur pengesahan pengiriman pada MQTT menjamin keutuhan data sensor agar tiba dengan aman di *server* meskipun sistem beroperasi pada jaringan publik yang tidak stabil.

#### 2. 2. 8 Broker Eclipse Mosquitto

Eclipse Mosquitto adalah sebuah *broker* pesan MQTT *open-source* berarsitektur ringan dari Eclipse Foundation yang dirancang untuk sistem IoT waktu-nyata. *Broker* ini mendukung protokol MQTT v3.1 dan v3.1.1 dengan kapabilitas penanganan koneksi masif serta penerapan *Quality of Service* (QoS) yang fleksibel. Berdasarkan pengujian Mishra dkk. (2021) dan Bertrand-Martinez dkk. (2021), Mosquitto terbukti mencetak unjuk kerja *throughput* yang tinggi dan *delay* yang rendah dibandingkan *broker* sejenis. Selain itu, ketersediaan fitur *bridging* memungkinkannya mendistribusikan *payload* pesan antar-*broker* guna mencegah pembotolan (*bottleneck*) pada jaringan terdistribusi.





### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta



Gambar 2. 2 Eclipse MQTT

Sumber: (mosquitto.org, 2026)

### 2. 2. 9 Blynk Cloud

Blynk merupakan platform *Software as a Service* (SaaS) komersial pihak ketiga yang bertujuan untuk mengendalikan dan memantau perangkat *Internet of things* (IoT) seperti Arduino, Raspberry Pi, ESP8266, Wemos D1, dan mikrokontroler sejenisnya melalui koneksi internet. Platform ini menawarkan kemudahan dalam pembuatan purwarupa (*prototyping*) dengan menyediakan aplikasi antarmuka grafis yang dapat dirancang secara praktis melalui metode seret dan lepas (*drag and drop*) widget (Kasyanto dkk., 2025). Pendekatan ini membuat Blynk sangat mudah digunakan untuk proses *monitoring* dan pengendalian perangkat, mengingat konfigurasi pada aplikasi seluler maupun pemrograman pada sisi papan mikrokontroler tidak memerlukan pengkodean yang rumit.



Gambar 2. 3 Blynk Cloud

Sumber: (g2.com, 2026)

### 2. 2. 10 Node.js

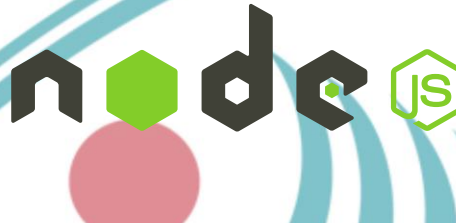
Node.js adalah lingkungan eksekusi (*runtime*) JavaScript sisi *server* yang mampu menangani proses komunikasi data secara *asynchronous* menggunakan arsitektur



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

*event-driven non-blocking I/O*. Turnip dkk. (2023) membuktikan bahwa implementasi teknologi Node.js dalam konsep komunikasi *Smart Farming* sangat efektif untuk menangani aliran data yang besar. Arsitektur ini memungkinkan pengembangan antarmuka web yang responsif dan sistem komunikasi yang andal, sekaligus mengatasi kendala yang sering ditemukan pada metode pengembangan web berbasis *blocking* konvensional.



Gambar 2. 4 Node Js

Sumber: (nodejs.org, 2025)

#### 2. 2. 11 Cloud Firebase

Google Cloud Firebase adalah platform *Backend-as-a-Service* (BaaS) yang menyediakan infrastruktur penyimpanan data berbasis *cloud* yang skalabel. Dalam ekosistem IoT, Firebase sering digunakan sebagai media penyimpanan utama karena kemampuannya dalam menangani pertukaran data secara *real-time* antara perangkat mikrokontroler dan aplikasi antarmuka pengguna (Ayeni dan Adesoba, 2025).

Layanan spesifik yang digunakan dalam penelitian ini adalah Cloud Firestore, yaitu basis data NoSQL (*Not Only SQL*) yang menggunakan model penyimpanan berorientasi dokumen (*document-oriented*). Berbeda dengan basis data relasional konvensional, Firestore mengorganisir data dalam bentuk koleksi (*collections*) dan dokumen (*documents*) yang fleksibel. Menurut Ayeni dan Adesoba (2025), penggunaan Firebase memungkinkan perangkat IoT untuk menyinkronkan status sensor ke *server* dengan *delay* yang sangat rendah. Keunggulan utama Firestore yang dimanfaatkan dalam penelitian ini meliputi sinkronisasi *real-time* yang memperbarui tampilan klien secara otomatis tanpa memuat ulang halaman, skalabilitas dalam menangani data log sensor dalam jumlah besar, serta fleksibilitas





**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

struktur data JSON yang sangat kompatibel dengan data yang dikirimkan oleh mikrokontroler ESP32.



Gambar 2. 5 Google Cloud Firebase

Sumber: (Firebase.google.com, 2026)

## 2. 2. 12 *Virtual Private Server (VPS)*

Virtual Private Server (VPS) merupakan infrastruktur perantara yang dimanfaatkan untuk menyediakan akses IP Publik pada suatu arsitektur jaringan. Dalam implementasinya, VPS seringkali digunakan untuk mengatasi kendala pada *server* lokal yang tidak memiliki fasilitas IP Publik statis — sebuah batasan yang menyebabkan *server* lokal tersebut terisolasi dan tidak dapat dijangkau melalui jaringan internet luar. Melalui integrasi dengan metode transmisi *Virtual Private Network* (VPN) Tunneling, VPS bertindak sebagai jembatan yang membungkus (enkapsulasi) aliran data dari *server* lokal dengan header yang berisi informasi *routing*. Mekanisme perutean ini memungkinkan lalu lintas data menembus jaringan publik secara aman sehingga layanan aplikasi yang berada di dalam jaringan tertutup dapat diakses secara terpusat dari luar jaringan lokal (Mahmudin dan Hariyono, 2022).

## 2. 2. 13 *Kontainerisasi Docker*

Docker adalah platform kontainerisasi (*containerization*) yang memungkinkan pengembang untuk mengemas aplikasi beserta seluruh dependensinya ke dalam sebuah unit terisolasi yang disebut container. Berbeda dengan virtualisasi tradisional yang memerlukan sistem operasi terpisah untuk setiap mesin virtual, container Docker berbagi kernel sistem operasi yang sama sehingga jauh lebih ringan dan efisien dalam penggunaan sumber daya (Mahmudin dan Hariyono, 2022). Dalam penelitian ini, Docker digunakan untuk mengemas dan menjalankan aplikasi *backend* (Node.js) dan *frontend* pada infrastruktur VPS secara terisolasi.



**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Pendekatan ini memastikan konsistensi lingkungan eksekusi aplikasi antara tahap pengembangan dan tahap produksi, sekaligus menyederhanakan proses *deployment* dan pemeliharaan sistem pada *server*.

## 2. 2. 14            **Algoritma Rule-Based**

Algoritma *Rule-Based* merupakan salah satu pendekatan yang umum digunakan, di mana perangkat lunak dirancang untuk menangani masalah di bidang tertentu dengan mengadopsi pengetahuan atau keterampilan manusia ke dalam sebuah sistem komputer (Juanda dan Yadi, 2020). Sistem ini secara spesifik menyajikan keahlian pakar dalam bentuk aturan-aturan sebagai representasi pengetahuan dasar pada suatu domain tertentu untuk menyelesaikan sebuah permasalahan, yang dikodekan menggunakan teknik terstruktur ke dalam aturan logika kondisional IF-THEN.

Berbeda dengan logika Fuzzy yang menggunakan nilai probabilitas, metode *Rule-Based* bekerja pada logika tegas (*crisp logic*) dengan membandingkan nilai masukan secara langsung terhadap ambang batas (*threshold*) yang telah ditetapkan secara pasti. Karena mengandalkan pencocokan parameter yang pasti, algoritma *Rule-Based* dinilai sangat efisien dan ringan secara komputasi untuk dieksekusi pada pemrosesan sisi server secara waktu nyata (*real-time*) (Mubarok dan Romli, 2025).

Dalam penerapannya pada sistem pertanian, penentuan hasil akhir dari algoritma ini tidak dibuat secara asal, melainkan disesuaikan dengan referensi mengenai batas toleransi hidup tanaman. Sebagai contoh, pada budidaya hidroponik, pakar menetapkan kondisi lingkungan ke dalam tiga status utama yang menjadi dasar evaluasi algoritma (Zakiah dkk., 2025).

- Status Optimal: Dicapai ketika seluruh parameter (seperti pH dan kepekatan nutrisi) berada dalam ambang batas normal secara bersamaan, sehingga memungkinkan absorpsi hara oleh akar terjadi secara maksimal.
- Status Waspada: Dipicu ketika salah satu parameter menyimpang, yang mengindikasikan terjadinya penurunan efisiensi absorpsi namun belum sampai merusak fisiologi tanaman secara instan.





**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

- Status Kritis: Ditetapkan secara mutlak ketika parameter menyimpang secara ekstrem atau simultan (misalnya: air bersuasana terlalu asam sementara nutrisi sangat rendah/tinggi).

## 2. 2. 15 Presisi

Dalam konteks pengujian sistem laboratorium dan pengukuran sensor, presisi atau ketelitian merupakan metrik yang menunjukkan seberapa dekat suatu hasil apabila pengukuran dilakukan secara berulang-ulang dengan menggunakan sampel yang sama pada kondisi pengujian yang tetap. Menurut Kusmiati dkk. (2022), presisi menggambarkan tingkat konsistensi suatu instrumen atau sistem dalam menghasilkan output yang seragam. Semakin kecil nilai simpangan baku yang dihasilkan dari sekumpulan data pengukuran berulang, maka semakin tinggi tingkat presisi sistem tersebut, yang mengindikasikan rendahnya tingkat kesalahan acak (*random error*) selama proses pembacaan data berlangsung (Kusmiati dkk., 2022).

## 2. 2. 16 Wireshark

Wireshark merupakan aplikasi penganalisis protokol jaringan (*network protocol analyzer*) berbasis open-source yang sangat terkenal dan digunakan secara luas, di mana menurut Safara dkk. (2025), perangkat lunak ini berfungsi untuk menangkap, memindai, dan menganalisis trafik paket data yang bergerak melintasi jaringan secara real-time. Nusri dan Syah (2025) menambahkan bahwa aplikasi ini mampu menyajikan informasi yang sangat terperinci mengenai setiap paket, mencakup alamat sumber dan tujuan, jenis protokol yang digunakan, serta rincian data bawaannya. Selain menangkap paket secara aktual, Rahman dan Imamulkhair (2024) menjelaskan bahwa Wireshark juga dapat membaca berkas rekaman tangkapan paket (*packet capturing*) yang telah direkam sebelumnya untuk memecahkan masalah (*troubleshooting*), mengidentifikasi anomali keamanan, hingga menilai parameter kinerja jaringan. Mekanisme kerjanya beroperasi melalui perekaman paket pada antarmuka jaringan yang dipilih dan penggunaan fitur filter untuk menyaring data berdasarkan syarat tertentu. Sebagai alat dekripsi yang komprehensif, aplikasi ini mendukung pembacaan transmisi dari berbagai jenis koneksi seperti Ethernet, Token-Ring, FDDI, koneksi serial (PPP dan SLIP), ATM,



**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

hingga infrastruktur nirkabel 802.11 lengkap dengan pemindaian protokol keamanan WEP dan WPA/WPA2.

## 2. 2. 17 *User Acceptance Test (UAT)*

*User Acceptance Testing (UAT)* merupakan tahapan pengujian yang secara langsung dilakukan oleh klien atau pengguna akhir (*end-user*). Alasan utama dilaksanakannya pengujian ini adalah untuk mengidentifikasi fungsionalitas sistem yang telah dibangun serta memastikan seberapa besar manfaat operasional sistem tersebut bagi pengguna akhir sebelum perangkat lunak diimplementasikan secara resmi ke dalam lingkungan produksi (Yakub dkk., 2024). Melalui metode pengujian UAT, pengembang dapat menyelaraskan kesesuaian antara spesifikasi awal yang dirancang dengan ekspektasi dan tingkat pemahaman pengguna di lapangan, sehingga sistem yang dihasilkan tepat guna dan tepat sasaran.

## 2. 2. 18 *Pengujian Sistem*

Pengujian sistem dilakukan untuk memastikan bahwa sistem bekerja sesuai harapan. Terdapat beberapa pengujian yang dilakukan terhadap sistem yang dibangun dalam penelitian ini.

### 1. *Pengujian Fungsionalitas*

Pengujian fungsionalitas dilakukan untuk memvalidasi integrasi aliran data secara *end-to-end* dan memastikan bahwa seluruh fitur sistem beroperasi sesuai dengan spesifikasi kebutuhan. Mengacu pada standar kualitas perangkat lunak (Dako dan Ridwan, 2021), pengujian karakteristik *Functional Suitability* dievaluasi menggunakan metode *Black-box testing*. Pengujian ini berfokus murni pada kesesuaian hasil keluaran (*output*) terhadap skenario masukan (*input*) yang diberikan, tanpa menganalisis struktur kode internal komputasi. Tingkat kelayakan dan keberhasilan fungsional sistem dihitung menggunakan persamaan (1):

$$\text{Persentase Keberhasilan} = \frac{\text{Jumlah Fungsi yang Berhasil Dieksekusi}}{\text{Total Fungsi yang Diuji}} \times 100\% \quad (1)$$





**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

2. Pengujian Logika *Rule-Based*

Pengujian algoritma *Rule-Based* pada *server* dilakukan untuk memvalidasi presisi keluaran klasifikasi status kualitas air (Optimal, Waspada, Kritis) berdasarkan kombinasi nilai masukan pH dan TDS. Sesuai dengan parameter pengujian integrasi *end-to-end*, validasi ini menggunakan metode *Black-box testing*. Skenario uji dirancang untuk mencakup seluruh kemungkinan probabilitas kondisi batas (*boundary condition*) dengan cara mentransmisikan nilai uji dari mikrokontroler (alat) menuju *server*, yang kemudian diselaraskan secara komparatif dengan hasil status akhir yang ditampilkan pada antarmuka web. Tingkat keberhasilan validasi dievaluasi menggunakan persentase akurasi pencocokan luaran tersebut, merujuk pada persamaan (1).

3. Pengujian *Quality of Services* (QoS)

Pengujian QoS digunakan untuk mengevaluasi kinerja transmisi jaringan berbasis protokol komunikasi. Data kuantitatif dianalisis menggunakan metode deskriptif komparatif berdasarkan standar TIPHON (*Telecommunications and Internet Protocol Harmonization Over Network*). Penggunaan standar ini bertujuan untuk menyediakan kerangka kerja yang menyeluruh guna menilai kualitas jaringan secara objektif (Ardiansyah dan Pamuji, 2025).

Untuk memberikan penilaian, evaluasi jaringan mengacu pada matriks standar TIPHON. Rekapitulasi klasifikasi indeks secara umum berdasarkan persentase kualitas layanan disajikan pada Tabel 2.2.

Tabel 2. 2 Klasifikasi Nilai Quality of Service (TIPHON)

| <i>Indeks</i> | <i>Persentase</i> | <i>Klasifikasi</i> |
|---------------|-------------------|--------------------|
| 3.8 – 4       | 95% – 100%        | Sangat Bagus       |
| 3 – 3.79      | 75% – 94%         | Bagus              |
| 2 – 2.99      | 50% – 74%         | Sedang             |
| 1 – 1.99      | <50%              | Buruk              |



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang menggunakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Guna mengukur indeks tersebut, terdapat empat parameter utama yang dianalisis, yaitu *Throughput*, *Packet loss*, *Delay*, serta *Jitter*.

#### 1. *Throughput*

Nilai *throughput* dihitung untuk mengetahui jumlah data yang secara efektif dikirim dari satu lokasi ke lokasi lain selama durasi tertentu, menggunakan persamaan (2).

$$\text{Throughput} = \frac{\text{Jumlah Data Diterima (kb)}}{\text{Waktu Pengamatan (s)}} \quad (2)$$

#### 2. *Packet Loss*

Nilai *packet loss* dihitung untuk mengetahui persentase paket data yang dikirim namun tidak tiba di sasaran yang dituju, menggunakan persamaan (3).

$$\text{Packet Loss} = \frac{(\text{Paket Dikirim} - \text{Paket Diterima})}{\text{Paket Dikirim}} \times 100\% \quad (3)$$

#### 3. *Delay*

Nilai *delay* dihitung untuk mengetahui durasi yang diperlukan paket data dari sumber menuju tujuan pada jaringan, menggunakan persamaan (4)

$$\text{Rata - rata Delay} = \frac{\text{Total Delay}}{\text{Total Paket Diterima}} \quad (4)$$

#### 4. *Jitter*

Nilai *jitter* dihitung untuk mengetahui total variasi *delay* yang terjadi ketika paket data tidak tiba dalam interval waktu yang konsisten, menggunakan persamaan (5).

$$\text{Jitter} = \frac{\text{Total Variasi Delay}}{\text{Jumlah Paket Diterima}} \quad (5)$$

Seluruh hasil perhitungan dari keempat parameter QoS tersebut kemudian dievaluasi dan diklasifikasikan berdasarkan indeks nilai standar TIPHON. Kategori klasifikasi kinerja ini ditunjukkan pada Tabel 2.3.





Tabel 2. 3 Klasifikasi Nilai Standar QoS Berdasarkan TIPHON

| <i>Indeks</i> | <i>Kategori</i> | <i>Throughput</i>    | <i>Packet loss</i> | <i>Delay</i> | <i>Jitter</i> |
|---------------|-----------------|----------------------|--------------------|--------------|---------------|
| 4             | Sangat Bagus    | > 2.1 Mbps           | 0 – 2%             | < 150 ms     | 0 ms          |
| 3             | Bagus           | 1200 Kbps – 2.1 Mbps | 3 – 14%            | 150 – 300 ms | 0 – 75 ms     |
| 2             | Sedang          | 338 – 1200 Kbps      | 15 – 24%           | 300 – 450 ms | 75 – 125 ms   |
| 1             | Buruk           | 0 – 338 Kbps         | > 25%              | > 450 ms     | 125 – 225 ms  |

Seluruh skenario pengujian QoS pada masing-masing protokol dieksekusi sebanyak 5 (lima) kali dan hasilnya dirata-ratakan untuk menghasilkan nilai yang representatif dan meminimalisasi pengaruh fluktuasi jaringan sesaat. Jumlah iterasi ini mengacu pada praktik eksperimen pengujian protokol IoT oleh Laaroussi dan Novo (2021) yang menerapkan 5 repetisi guna memperoleh nilai rata-rata beserta standar deviasi yang stabil sebagai ukuran konsistensi hasil pengukuran, mengingat standar TIPHON tidak menetapkan jumlah pengulangan pengujian secara eksplisit.

### 3. Pengujian *Reliability*

Pengujian keandalan (*reliability*) dilakukan untuk mengukur derajat kemampuan sistem dalam mempertahankan ketersediaan layanan (*availability*), stabilitas komputasi, serta keutuhan transmisi data ketika dioperasikan pada kondisi dan rentang waktu yang spesifik. Mengacu pada standar model kualitas perangkat lunak ISO/IEC 25010, karakteristik *Reliability* menuntut sistem untuk beroperasi secara konsisten tanpa mengalami degradasi kinerja atau kegagalan fatal. Guna menghasilkan pengujian yang komprehensif dan valid secara ilmiah, pengujian keandalan pada penelitian ini dievaluasi melalui tiga dimensi pengujian sebagai berikut:

1. Pengujian Toleransi Beban (*Fault tolerance*) menggunakan Apache JMeter  
Pengujian ketahanan (*load testing*) dieksekusi guna mengukur kapasitas, batas elastisitas, serta konsistensi unjuk kerja *server* saat menangani

#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :  
a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.  
b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

akumulasi kueri secara serentak. Dalam penelitian ini, Apache JMeter difungsikan sebagai generator beban asinkron menuju infrastruktur VPS melalui pola pembebanan bertingkat (*step-load pattern*). Pendekatan *step-load* diimplementasikan dengan menaikkan jumlah *threads* (*virtual users*) secara gradual, metode ini terbukti secara empiris mampu memperlihatkan titik infleksi (*inflection point*) degradasi kinerja *server* sebelum menyentuh ambang jenuh komputasi (Nugraha dkk., 2024).

Mengacu pada pembaruan standar kualitas produk perangkat lunak ISO/IEC 25010:2023, evaluasi pilar *Performance Efficiency* dipisahkan ke dalam aspek *Time Behavior* dan *Resource Utilization*. Oleh karena itu, parameter yang diekstraksi selama simulasi pembebanan mencakup tiga metrik fundamental: (1) *Error Rate*, yakni rasio kegagalan *server* dalam memvalidasi kueri yang dihitung menggunakan Persamaan (6); (2) *Throughput*, guna mengukur laju penyelesaian transaksi per detik; serta (3) Utilisasi Sumber Daya *CPU* dan *Memory Usage*, untuk memantau eskalasi beban kerja prosesor dan memori fisik VPS. Pengukuran terpadu pada ketiga metrik ini mengunci justifikasi presisi mengenai batas toleransi *server* (*fault tolerance*) di bawah hantaman kueri masif (Andrianto dan Suyatno, 2024).

$$\text{Error Rate} = \frac{\text{Jumlah Request Gagal}}{\text{Total Request Dikirim}} \times 100\% \quad (6)$$

#### 2. Pengujian Ketahanan Jangka Panjang (*Maturity / Soak testing*)

Sub-karakteristik *Maturity* dalam ISO/IEC 25010 menuntut sistem agar mampu beroperasi dalam jangka waktu yang panjang tanpa mengalami anomali kebocoran memori (*memory leak*). Pengujian ini dieksekusi melalui skenario *Soak testing* selama 6 jam pengamatan konstan dengan menginjeksi *payload* telemetri riil dari mikrokontroler pengirim setiap 5 detik sekali menuju *broker* (*broker*) *server*. Tingkat keandalan unjuk kerja diukur berdasarkan rasio persentase keberhasilan penerimaan *payload* di basis data menggunakan Persamaan (7):





#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

$$\text{Reliability} = \frac{P_{\text{diterima}}}{P_{\text{dikirim}}} \times 100\% \quad (7)$$

Sistem dikategorikan andal (*reliable*) apabila persentase keberhasilan penerimaan paket menyentuh angka  $> 95\%$ , yang ekuivalen dengan kriteria "Sangat Bagus" pada indeks standarisasi telekomunikasi TIPHON.

### 3. Pengujian Daya Pulih Sistem (*Recoverability / Reconnect Testing*)

Sub-karakteristik *Recoverability* mendefinisikan tingkat kemampuan sistem untuk membangun kembali unjuk kerja asalnya secara otonom serta memulihkan koneksi yang terdampak secara langsung pasca-terjadinya interupsi atau kegagalan fatal (ISO/IEC, 2023). Pada implementasi jaringan *Internet of things* (IoT) di lahan pertanian terbuka, anomali berupa pemutusan tautan lapisan fisik (*link drop*) akibat ketidakstabilan sinyal Wi-Fi maupun kegagalan *runtime daemon* pada *server* merupakan variabel lingkungan yang tidak dapat dihindari.

Pengujian daya pulih dirancang untuk mengukur waktu pemulihan (*recovery latency*) yang dibutuhkan oleh sistem sejak titik kegagalan dihentikan hingga klien berhasil melakukan pengikatan ulang sesi (*re-binding session*) dan memvalidasi paket data pertamanya ke basis data (Suryadi dkk., 2024). Secara matematis, waktu pemulihan dirumuskan pada Persamaan (8):

$$\text{Recovery Time} = T_{\text{mendarat}} - T_{\text{normalisasi}} \quad (8)$$

Keterangan:

- $T_{\text{mendarat}}$ , stempel waktu (*timestamp*) murni saat paket telemetri pertama pasca-kegagalan terekam masuk ke dalam basis data Cloud Firestore.
- $T_{\text{normalisasi}}$ , stempel waktu saat pemicu kegagalan (titik akses Wi-Fi atau *daemon Mosquitto*) dihidupkan kembali ke kondisi aktif.

Berdasarkan kaidah *Reliability Engineering*, sistem dinyatakan memiliki kapabilitas *self-healing* yang layak apabila mampu memulihkan jalur

transmisi di bawah ambang batas waktu 30 detik secara otonom, tanpa menuntut tindakan *hard-reset* manual dari operator (Pratama dkk., 2023).



## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta





**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

## BAB III

### METODE PENELITIAN

#### 3.1 Rancangan Penelitian

Penelitian ini menggunakan pendekatan kuantitatif dengan desain eksperimental komparatif. Sebagaimana dijelaskan oleh Munte dkk. (2023), penelitian eksperimen adalah pendekatan yang melibatkan manipulasi variabel independen untuk mengukur dampaknya terhadap variabel dependen, dengan tujuan memahami hubungan sebab-akibat antar variabel yang diteliti. Lebih lanjut, desain kausal komparatif sebagai turunan dari pendekatan eksperimental berfokus pada perbandingan antara dua atau lebih kondisi atau variabel independen untuk mengidentifikasi perbedaan di antara keduanya — di mana tujuan utamanya adalah komparasi, bukan sekadar penetapan kausalitas (Munte dkk., 2023). Pendekatan ini dipilih karena tujuan utama penelitian adalah menghasilkan data numerik yang terukur dan dapat dibandingkan secara objektif antara ketiga protokol komunikasi yang dievaluasi.

Dalam rancangan ini, jenis protokol komunikasi IoT ditetapkan sebagai variabel bebas (*independent variable*), yaitu HTTP, MQTT, dan WebSocket. Kinerja transmisi data yang diukur melalui parameter *Quality of Service* (QoS) standar TIPHON — meliputi *delay*, *throughput*, *packet loss*, dan *jitter* — ditetapkan sebagai variabel terikat (*dependent variable*). Guna memastikan komparasi bersifat setara (*apple-to-apple*), beberapa kondisi ditetapkan sebagai variabel kontrol (*control variable*), yaitu spesifikasi mikrokontroler (ESP32), jenis dan ukuran *payload* data sensor (pH dan TDS dalam format JSON), infrastruktur VPS yang identik untuk ketiga protokol, serta penyedia layanan internet yang sama selama seluruh sesi pengujian berlangsung.

Pengumpulan data kuantitatif dilakukan dengan metode observasi terstruktur menggunakan dua instrumen utama. Pertama, perangkat lunak Wireshark sebagai penganalisis protokol jaringan untuk mengukur kinerja QoS ketiga protokol secara *end-to-end*. Kedua, Apache JMeter untuk pengujian beban komputasi (*load testing*)



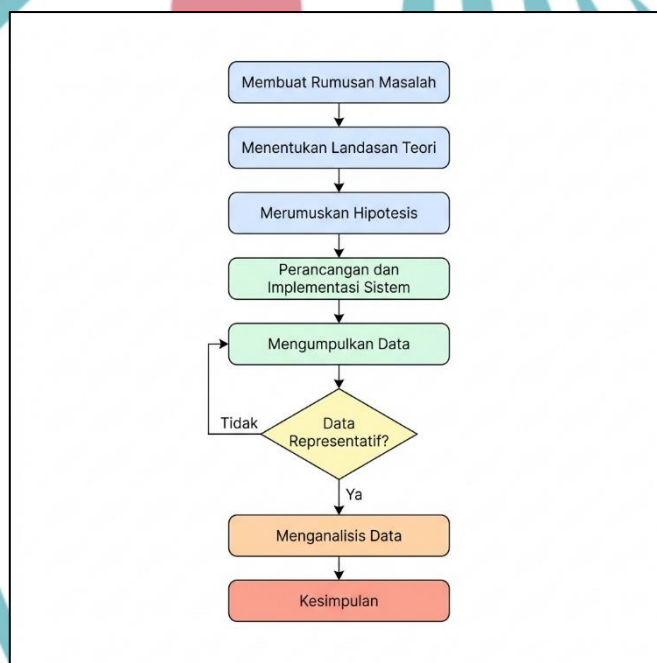
### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

guna mengukur ketahanan infrastruktur VPS dalam menangani lonjakan permintaan berskala masif. Data mentah berupa berkas rekaman jaringan (.pcap) dari Wireshark diekstraksi kemudian dievaluasi menggunakan teknik analisis statistik deskriptif komparatif yang disandarkan pada standar TIPHON (*Telecommunications and Internet Protocol Harmonization Over Network*).

### 3.2 Tahapan Penelitian

Pelaksanaan penelitian ini dieksekusi secara terstruktur mengadopsi kerangka kerja metode penelitian kuantitatif eksperimental (Amruddin dkk., 2022). Alur sistematis dari tahapan eksperimental komparatif ini divisualisasikan pada Gambar 3.1.



Gambar 3. 1 Tahapan Penelitian

Adapun penjabaran operasional dari masing-masing tahapan adalah sebagai berikut:

#### 1. Membuat Rumusan Masalah

Tahap inisiasi difokuskan pada identifikasi celah penelitian dalam literatur evaluasi protokol komunikasi IoT pada konteks sistem monitoring hidroponik. Kajian terhadap penelitian terdahulu mengungkap dua kesenjangan yang berulang secara konsisten: pertama, sistem monitoring hidroponik yang ada belum memvalidasi klaim *real-time* dengan pengukuran QoS yang terstandarisasi; kedua, evaluasi protokol yang ada masih parsial karena tidak ada studi yang membandingkan





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

HTTP, MQTT, dan WebSocket secara sejajar dalam satu kerangka pengujian terpadu. Dari identifikasi tersebut, dirumuskan masalah inti bagaimana karakteristik kinerja transmisi data dan protokol manakah yang paling optimal berdasarkan standar TIPHON untuk sistem monitoring hidroponik presisi berbasis IoT.

### 2. Menentukan Landasan Teori

Kajian literatur secara mendalam dilakukan untuk membangun landasan konseptual dari solusi yang diusulkan. Eksplorasi teori mencakup: (1) anatomi dan karakteristik teknis ketiga protokol komunikasi — HTTP, MQTT, dan WebSocket; (2) standardisasi matriks *Quality of Service* (QoS) yang merujuk pada regulasi telekomunikasi TIPHON meliputi parameter *delay*, *throughput*, *packet loss*, dan *jitter*; (3) metode *load testing* untuk mengevaluasi ambang batas komputasi *server*; serta (4) arsitektur *Client-Server* berbasis Node.js pada infrastruktur VPS sebagai lingkungan pengujian.

### 3. Merumuskan Hipotesis

Berdasarkan sintesis landasan teori, dirumuskan hipotesis eksperimental bahwa terdapat perbedaan yang signifikan pada karakteristik kinerja QoS antara protokol HTTP, MQTT, dan WebSocket ketika diuji pada infrastruktur VPS mandiri yang identic dan terdapat satu protokol yang paling memenuhi standar reliabilitas TIPHON secara menyeluruh berdasarkan kombinasi keempat parameter QoS untuk implementasi sistem monitoring hidroponik presisi berbasis IoT.

### 4. Perancangan dan Implemementasi Sistem

Tahap ini merupakan prasyarat sebelum pengumpulan data dapat dilaksanakan. Sistem monitoring hidroponik dibangun pada infrastruktur VPS mandiri berbasis Node.js, meliputi: (1) implementasi endpoint HTTP REST API sebagai objek pengujian protokol *request-response*; (2) instalasi dan konfigurasi *broker* Eclipse Mosquitto pada *port* 1883 untuk pengujian protokol *publish-subscribe*; (3) implementasi *server* WebSocket untuk pengujian transmisi *full-duplex* persisten; (4) penerapan algoritma *Rule-Based* (IF-THEN) pada sisi *backend* untuk klasifikasi status kualitas air; (5) integrasi Cloud Firestore sebagai penyimpanan data historis; serta (6) perancangan *firmware* mikrokontroler ESP32 untuk pengambilan dan transmisi data sensor secara bergantian menggunakan ketiga protokol.



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

#### 5. Mengumpulkan Data

Fase pengumpulan data kuantitatif dieksekusi melalui dua instrumen pengukuran utama. Pertama, ekstraksi lalu lintas paket data (*packet sniffing*) pada lapisan transpor menggunakan Wireshark yang difokuskan pada segmen jaringan *end-to-end* dari perangkat ESP32 menuju *server* VPS. Pengujian diulang sebanyak 5 (lima) kali untuk setiap protokol dan hasilnya dirata-ratakan untuk menghasilkan nilai yang representatif dan meminimalisasi pengaruh fluktuasi jaringan sesaat. Kedua, perekaman data kinerja *server* VPS dilakukan melalui simulasi injeksi lonjakan permintaan serentak menggunakan Apache JMeter dengan variasi jumlah *thread* yang ditingkatkan secara bertahap. Seluruh variabel kontrol dijaga secara absolut selama kedua sesi pengujian untuk menjamin komparasi yang setara.

#### 6. Menganalisis Data

Data mentah hasil eksperimen diekstraksi dan dievaluasi menggunakan teknik analisis statistik deskriptif komparatif. Terdapat lima jenis pengujian yang dilaksanakan: (1) Pengujian Fungsionalitas menggunakan *Black-box testing*; (2) Pengujian Algoritma *Rule-Based* menggunakan *Black-box testing*; (3) Pengujian QoS dengan standar TIPHON pada ketiga protokol menggunakan Wireshark; (4) Pengujian *Load testing* infrastruktur VPS menggunakan Apache JMeter; dan (5) Pengujian Reliability berdasarkan standar ISO/IEC 25010. Hasil komparasi keempat parameter QoS antar protokol kemudian divalidasi secara matematis dengan menyandarkannya pada rentang indeks standar TIPHON.

#### 7. Kesimpulan

Tahap final merupakan penarikan deduksi secara objektif. Hasil analisis komparatif kuantitatif digunakan untuk memverifikasi hipotesis dan menjawab secara empiris protokol komunikasi mana yang paling optimal berdasarkan standar TIPHON untuk diimplementasikan pada sistem monitoring hidroponik presisi berbasis IoT.

### 3.3 Objek Penelitian

Secara spesifik, objek penelitian ini terbagi ke dalam empat komponen arsitektural utama yang dievaluasi secara teknis:

#### 1. Protokol Komunikasi Data





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Mekanisme transmisi jaringan yang menjadi fokus pengujian komparatif, terdiri atas protokol HTTP (*request-response*), protokol MQTT (*publish-subscribe*), dan protokol WebSocket (*full-duplex*). Ketiga protokol ini merupakan entitas logis yang dimanipulasi dan diuji secara bergantian sebagai variabel bebas untuk mengidentifikasi tingkat efisiensi aliran data secara *end-to-end* berdasarkan parameter QoS standar TIPHON.

### 2. Infrastruktur *Server* Mandiri

Sistem komputasi terpusat pada *Virtual Private Server* (VPS) berbasis Linux yang menjalankan aplikasi *backend* Node.js dan *frontend* dalam lingkungan kontainer Docker. *Server* ini merupakan objek yang dievaluasi ketahanannya melalui *load testing* dalam menangani koneksi transmisi data berskala besar secara *asynchronous*, sekaligus mengeksekusi komputasi algoritma *Rule-Based* untuk klasifikasi kualitas air secara *real-time*. Khusus untuk implementasi protokol MQTT, infrastruktur VPS ini secara langsung menjalankan daemon Eclipse Mosquitto pada *port* 1883 sebagai pusat *broker* pesan (*message broker*).

### 3. Perangkat Keras *Node Sensor* (*Edge Object*)

Infrastruktur perangkat keras di sisi klien yang digerakkan oleh mikrokontroler ESP32, terintegrasi dengan instrumen pembaca parameter lingkungan hidroponik meliputi sensor pH, TDS, suhu, kelembapan, dan ultrasonik. Modul ini berfungsi sebagai titik asal (*data source*) dalam pengambilan telemetri *real-time* yang menghasilkan beban lalu lintas paket data (*network traffic*) untuk diukur kinerja transmisinya.

**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

## BAB IV

### HASIL DAN PEMBAHASAN

#### 4. 1 Analisis Kebutuhan

Pada penelitian ini, arsitektur *Client-Server* untuk platform *monitoring* hidroponik dirancang terpusat menggunakan instans *Compute Engine* pada Google Cloud Platform (GCP) sebagai *server* utama yang dapat diakses melalui IP publik. Antarmuka sistem dibangun melalui integrasi layanan *backend* Node.js dan presentasi visual berbasis React, yang diimplementasikan dalam lingkungan kontainer Docker pada instans GCP tersebut. Sementara itu, lapisan IoT beroperasi menggunakan mikrokontroler ESP32 sebagai node sensor yang bertugas mengambil dan mengirimkan data telemetri kualitas air secara berkala ke *server*. Guna memastikan infrastruktur beroperasi sesuai sasaran evaluasi, spesifikasi sistem dipetakan ke dalam dua klasifikasi utama — analisis kebutuhan fungsional dan kebutuhan non-fungsional yang dijabarkan secara berturut-turut pada Tabel 4.1 dan Tabel 4.2.

##### 4. 1. 1 Kebutuhan Fungsionalitas

Kebutuhan fungsional mendefinisikan batasan operasional, kapabilitas fitur, serta interaksi logis yang wajib dieksekusi oleh infrastruktur sistem, baik pada tingkat presentasi antarmuka (*frontend*) maupun layanan pemrosesan data (*backend*). Mengingat platform ini dirancang secara spesifik untuk mengakomodasi penerimaan data telemetri sensor hidroponik dan visualisasi *monitoring* jarak jauh secara *real-time*, rincian pemetaan spesifikasi fungsionalitas tersebut dijabarkan pada Tabel 4.1.

Tabel 4. 1 Kebutuhan Fungsionalitas Sistem

| ID   | Fitur / Fungsionalitas | Deskripsi Kebutuhan                                                                                               |
|------|------------------------|-------------------------------------------------------------------------------------------------------------------|
| F-01 | Autentikasi Pengguna   | Sistem <i>backend</i> dan <i>frontend</i> menyediakan mekanisme validasi masuk ( <i>login</i> ) dasar menggunakan |





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|      |                                  |                                                                                                                                                                                                                                                                                                                                                                       |
|------|----------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|      |                                  | nama pengguna dan kata sandi untuk membatasi akses visualisasi <i>dashboard</i> .                                                                                                                                                                                                                                                                                     |
| F-02 | Pengambilan Data Sensor          | Mikrokontroler ESP32 dapat membaca data metrik kualitas air secara aktual dari sensor pH dan sensor TDS/PPM.                                                                                                                                                                                                                                                          |
| F-03 | Pengiriman Telemetry             | Mikrokontroler ESP32 dapat memaketkan data sensor dan mengirimkannya ke <i>server</i> secara berkala melalui tiga mekanisme protokol transmisi yang diuji secara bergantian, yaitu HTTP ( <i>request-response</i> ), MQTT ( <i>publish-subscribe</i> melalui <i>broker</i> Eclipse Mosquitto), dan WebSocket ( <i>full-duplex</i> ), tanpa lapisan enkripsi berlapis. |
| F-04 | Penerimaan dan Pencatatan Log    | Layanan <i>backend</i> (Node.js) dapat menerima dan mengekstrak <i>payload</i> data telemetry yang masuk melalui masing-masing protokol (HTTP <i>endpoint</i> , MQTT <i>broker</i> subscription, dan WebSocket <i>message handler</i> ), kemudian menyimpannya ke dalam Cloud Firestore beserta stempel waktu ( <i>timestamp logging</i> ).                           |
| F-05 | Distribusi Data <i>Real-time</i> | Layanan <i>backend</i> (Node.js) dapat menyiarkan aliran data telemetry yang masuk secara seketika                                                                                                                                                                                                                                                                    |



## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|      |                                         |                                                                                                                                                                                                                                                                          |
|------|-----------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|      |                                         | kepada klien web menggunakan koneksi WebSocket.                                                                                                                                                                                                                          |
| F-06 | Visualisasi <i>Dashboard Monitoring</i> | Antarmuka web (React) dapat menerima aliran WebSocket dan menyajikan grafik data pH dan PPM secara <i>real-time</i> tanpa melakukan <i>pepayload</i> ulang halaman.                                                                                                      |
| F-07 | Notifikasi Ambang Batas                 | Sistem dapat mengevaluasi nilai telemetri menggunakan algoritma <i>Rule-Based</i> (IF-THEN) di sisi <i>backend</i> dan menampilkan indikator peringatan visual pada antarmuka <i>frontend</i> apabila data pH atau PPM melampaui parameter toleransi yang dikonfigurasi. |

### 4. 1. 2 Kebutuhan Non Fungsionalitas

Kebutuhan non-fungsional mendefinisikan batasan arsitektural, standar kualitas layanan, kinerja komputasi, dan keandalan yang menjadi tolok ukur kelayakan sistem secara operasional. Kebutuhan ini menjadi parameter utama dalam tahap pengujian kinerja jaringan (QoS) dan stabilitas *server* diuraikan pada Tabel 4.2.

Tabel 4. 2 Kebutuhan Non Fungsionalitas Sistem

| ID    | Fitur / Fungsionalitas | Deskripsi Kebutuhan                                                                                                                                                         |
|-------|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| NF-01 | Kinerja Jaringan (QoS) | Evaluasi kinerja komunikasi data antara perangkat <i>Edge</i> dan <i>server cloud</i> wajib merujuk pada standar TIPHON ( <i>Telecommunications and Internet Protocol</i> ) |





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|       |                                                     |                                                                                                                                                                                                                                                                                                                                                                                              |
|-------|-----------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|       |                                                     | <p><i>Harmonization Over Network</i>).</p> <p>Sistem dituntut untuk mempertahankan keandalan pengiriman nilai sensor numerik (pH dan TDS) dengan indikator pengujian yang mencakup <i>Throughput, Packet loss, Delay</i>, serta <i>Jitter</i>.</p>                                                                                                                                           |
| NF-02 | Batasan <i>Delay</i> Transmisi                      | <p>Durasi waktu pengiriman paket data, yang dihitung sejak perangkat <i>Edge</i> memublikasikan <i>payload</i> data MQTT hingga <i>backend</i> berhasil mengekstraksinya, harus memenuhi kriteria kelayakan <i>monitoring real-time</i>. Nilai rata-rata <i>delay</i> tersebut ditargetkan masuk dalam kategori "Sangat Bagus" atau minimal "Bagus" sesuai dengan rentang indeks TIPHON.</p> |
| NF-03 | Ketahanan Beban Komputasi ( <i>Load Tolerance</i> ) | <p>Instans <i>GCP Compute Engine</i> yang berjalan harus memiliki kapabilitas untuk menangani simulasi injeksi ribuan <i>request</i> data telemetri secara bersamaan (<i>concurrent</i>). Sistem harus mampu menjaga stabilitas utilisasi CPU dan RAM tanpa mengalami kelumpuhan layanan (<i>crash</i> atau <i>downtime</i>), yang diukur melalui</p>                                        |



## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|       |                                                |                                                                                                                                                                                                                                                                                                                                                                             |
|-------|------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|       |                                                | pengujian beban ( <i>load testing</i> ) menggunakan Apache JMeter.                                                                                                                                                                                                                                                                                                          |
| NF-04 | Ketersediaan Sistem<br>( <i>Availability</i> ) | Lingkungan kontainer Docker pada instans GCP harus dikonfigurasi dengan mekanisme <i>auto-restart</i> memanfaatkan utilitas pengelola proses (PM2 untuk Node.js). Mekanisme ini menjamin layanan dapat aktif kembali secara otomatis ( <i>self-healing</i> ) apabila terjadi kegagalan proses atau <i>reboot</i> pada tingkat sistem operasi.                               |
| NF-05 | Responsivitas Antarmuka Web                    | Antarmuka klien ( <i>frontend</i> ) berbasis React harus mengadopsi arsitektur <i>Single Page Application</i> (SPA). Pendekatan ini diperlukan untuk memastikan navigasi antarhalaman visualisasi dan pembaruan grafik data telemetri tereksekusi secara instan, tanpa menuntut browser melakukan <i>pepayload</i> ulang struktur dokumen secara penuh dari <i>server</i> . |

## 4.2 Perancangan Sistem

Perancangan sistem difokuskan pada pengembangan arsitektur perangkat lunak yang efisien untuk menangani aliran data sensor secara berkesinambungan di sisi *server*. Perancangan ini mencakup tiga aspek utama, yaitu alur komunikasi data antar komponen, logika algoritma *Rule-Based* untuk klasifikasi kualitas air, dan struktur penyimpanan data pada basis data Cloud Firestore.



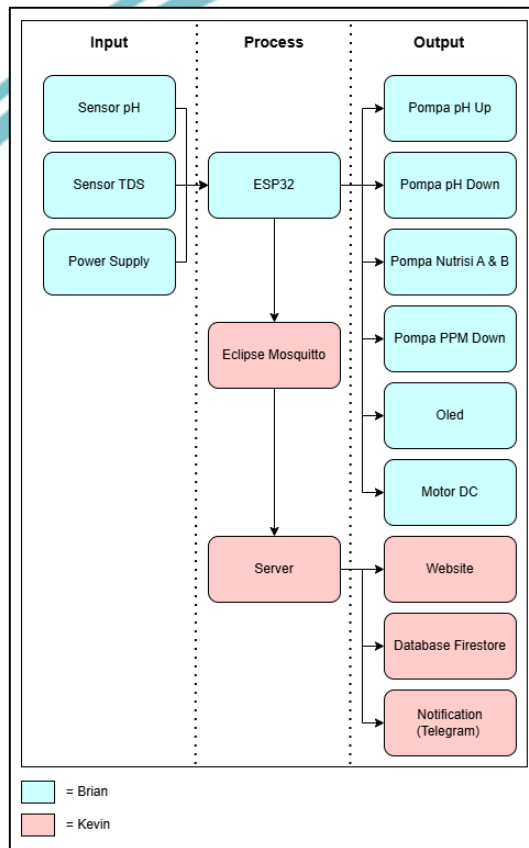


### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

## 4. 2. 1 Blok Diagram Sistem

Blok diagram sistem dirancang untuk memetakan aliran data dan hubungan fungsional antar komponen, mulai dari input fisik di lingkungan tanam hingga output visual pada layar pengguna. Sistem ini terdiri dari tiga blok utama, yaitu Blok Input, Blok Proses dan Komunikasi, dan Blok Output yang divisualisasikan pada Gambar 4.1.



Gambar 4. 1 Diagram Blok Sistem

Diagram blok pada Gambar 4.1 memperlihatkan alur data yang dimulai dari pengambilan data kualitas air (pH dan TDS) oleh sensor pada modul *node* IoT. Data analog dari sensor tersebut diproses oleh mikrokontroler ESP32, dikonversi menjadi format digital, dan dikemas dalam struktur JSON (*JavaScript Object Notation*). Paket data ini kemudian dikirimkan ke *backend* melalui salah satu dari tiga jalur protokol komunikasi yang diuji secara bergantian sesuai skenario pengujian.



### Hak Cipta :

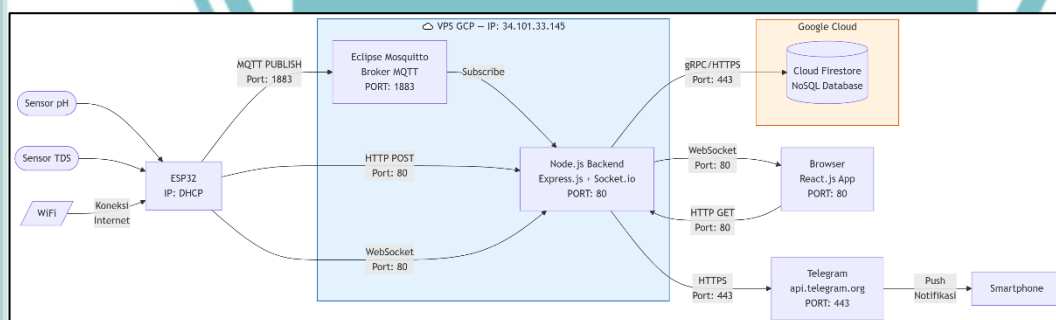
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Pada pengujian protokol HTTP, ESP32 mengirimkan data secara langsung ke *backend* Node.js yang berjalan pada VPS menggunakan mekanisme HTTP POST *request* tanpa perantara broker. Pada pengujian protokol MQTT, ESP32 mempublikasikan (*publish*) paket data ke broker Eclipse Mosquitto, yang selanjutnya diterima oleh *backend* Node.js melalui mekanisme *subscribe*. Sedangkan pada pengujian protokol WebSocket, ESP32 membuka koneksi saluran *full-duplex* secara persisten langsung ke *backend* Node.js dan mengirimkan data tanpa perlu melakukan *request* berulang.

Setelah data masuk melalui salah satu protokol tersebut, *backend* Node.js mengeksekusi tiga fungsi keluaran (*output*) secara paralel. Pertama, data historis disimpan ke dalam basis data Cloud Firestore. Kedua, data didistribusikan secara *real-time* ke antarmuka web (React.js) melalui koneksi WebSocket sisi klien untuk memperbarui grafik secara instan. Ketiga, sebagai peringatan dini, *backend* mengeksekusi pengiriman *push notification* ke aplikasi Telegram administrator apabila algoritma mendeteksi kondisi air berada pada status Waspada atau Kritis.

#### 4. 2. 2 Topologi Jaringan Sistem

Infrastruktur jaringan sistem dirancang menggunakan topologi *Client-Server* terdistribusi yang memisahkan beban pengambilan data di lahan fisik dengan beban komputasi di *server cloud*. Gambaran menyeluruh topologi jaringan beserta pemetaan *port* dan arah koneksi divisualisasikan pada Gambar 4.2.



Gambar 4. 2 Arsitektur Jaringan Sistem

Gambar 4.2 infrastruktur jaringan dibagi ke dalam 4 (empat) lapisan dengan fungsi spesifik sebagai berikut:





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

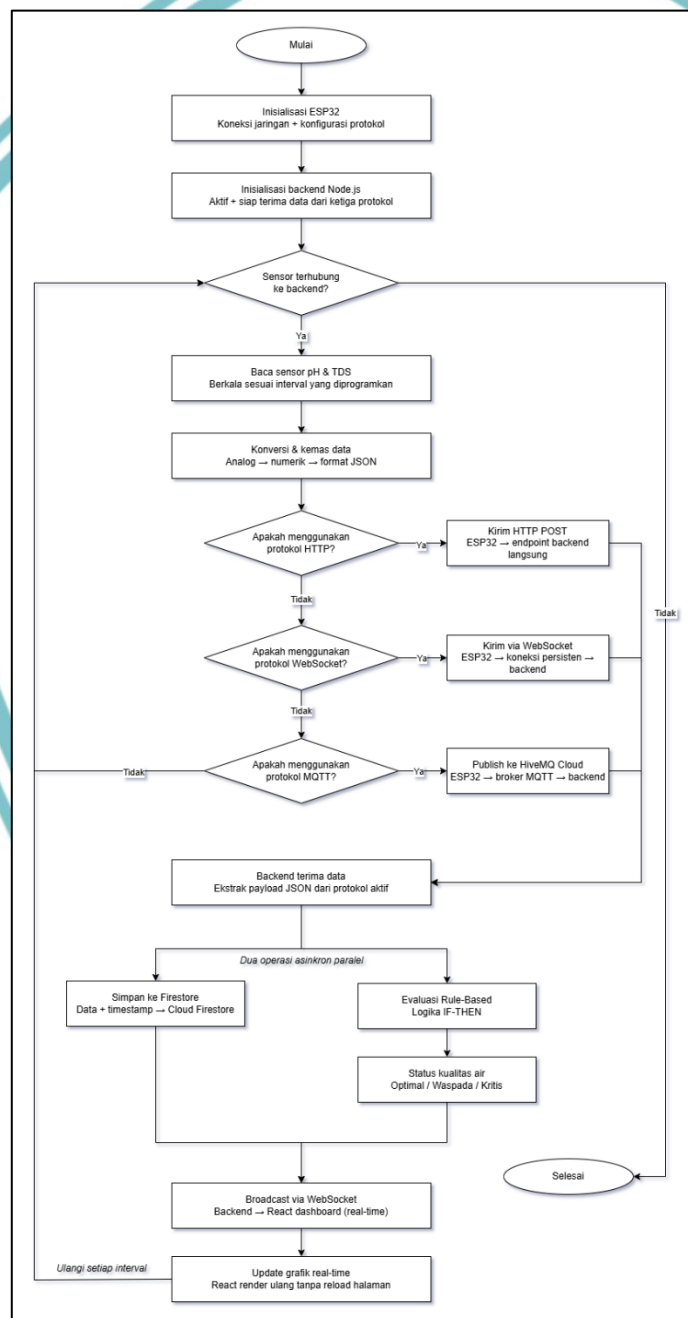
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

- Lapisan Perangkat Keras (*Edge Node*)  
Lapisan fisik bertugas melakukan pengambilan data mentah di lingkungan greenhouse. Mikrokontroler ESP32 membaca nilai tegangan analog dari sensor pH dan sensor TDS, lalu mengonversinya menjadi data numerik digital. Data tersebut kemudian ditransmisikan keluar melalui koneksi Wi-Fi lokal menuju server menggunakan salah satu dari tiga protokol yang diuji (MQTT pada *port* 1883, HTTP POST pada *port* 80, atau WebSocket pada *port* 80).
- Lapisan *Server* dan Komputasi *Cloud*  
Infrastruktur inti sistem berpusat pada mesin *Virtual Private Server* (VPS) yang beroperasi pada layanan Google Cloud Platform (GCP). Di dalam *server* ini, terdapat dua layanan utama yang berjalan secara bersamaan:
  - Eclipse Mosquitto (*port* 1883): Berperan sebagai broker MQTT yang bertugas menerima paket data dari ESP32 untuk kemudian diteruskan secara internal ke *backend*.
  - Node.js *Backend* (*port* 80): Berperan sebagai jembatan komunikasi utama yang menerima data dari seluruh jalur protokol. *Backend* ini bertugas mengeksekusi algoritma *Rule-Based*, menyimpan data ke basis data awan, serta mendistribusikan data secara *real-time* ke antarmuka pengguna.
- Lapisan Penyimpanan (Cloud Firestore)  
Data sensor yang telah diproses oleh *backend* disimpan secara permanen ke basis data NoSQL Cloud Firestore melalui protokol gRPC terenkripsi pada *port* 443. Penyimpanan difokuskan pada rekaman data telemetri historis dan manajemen autentikasi pengguna.
- Lapisan Pengguna Akhir (End-User)  
Terdapat dua titik akses bagi pengguna akhir dalam sistem ini:
  - Browser Klien (React.js App): Pengguna mengakses dasbor pemantauan melalui peramban web dengan koneksi awal HTTP (*port* 80), yang kemudian dialihkan menjadi koneksi WebSocket persisten untuk menerima aliran data secara *real-time*.
  - Notifikasi *Smartphone* (Telegram): Sistem memanfaatkan Telegram Bot API (protokol HTTPS *port* 443) untuk mengirimkan peringatan otomatis ke *smartphone* pengguna jika terjadi anomali pada kualitas air hidroponik.



#### 4. 2. 3 Diagram Alur Sistem

Perancangan alur kerja sistem secara keseluruhan memetakan siklus hidup transmisi data mulai dari tahap pengumpulan data pada simpul sensor perangkat keras hingga tahap visualisasi pada antarmuka pengguna. Diagram alur berikut merepresentasikan mekanisme integrasi arsitektur *Client-Server* dengan pola *decoupled*, di mana beban komunikasi protokol MQTT dipisahkan secara hierarki dari beban komputasi evaluasi algoritma *Rule-Based* di dalam *Server*.



Gambar 4. 3 Flowchart Kerja Sistem





#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Berdasarkan *flowchart* pada Gambar 4.3, cara kerja sistem dapat diuraikan sebagai berikut:

#### 1. Inisialisasi

Mikrokontroler ESP32 melakukan inisialisasi koneksi jaringan dan membangun sesi komunikasi terenkripsi dengan *broker* MQTT internal, Eclipse Mosquitto. Pada saat yang bersamaan, *server backend* Node.js pada VPS aktif dan menjalankan fungsi *subscribe* untuk mendengarkan lalu lintas topik MQTT yang telah dikonfigurasi.

#### 2. Pembacaan Sensor

Sensor mendeteksi nilai tegangan dari parameter kadar pH dan kepekatan nutrisi (TDS/PPM) pada larutan air hidroponik secara berkala sesuai dengan interval waktu yang diprogramkan.

#### 3. Transmisi (Publikasi)

ESP32 mengonversi tegangan analog menjadi data metrik numerik, mengemas data tersebut ke dalam format *payload* terstruktur (JSON), lalu mempublikasikannya (*publish*) secara langsung ke *broker* Eclipse Mosquitto untuk mendistribusikan beban jaringan di luar arsitektur *server* utama.

#### 4. Pemrosesan Data (*Server-Side*)

*Server* Node.js menerima aliran data dari *broker* dan mengeksekusi dua operasi asinkron secara paralel. Operasi pertama menyimpan pencatatan data metrik sensor beserta *timestamp* ke dalam basis data Firebase Cloud Firestore. Operasi kedua mengevaluasi *payload* tersebut menggunakan logika kondisional *Rule-Based* untuk mencocokkan nilai numerik dengan ambang batas (*threshold*) agronomi komoditas bayam, guna menetapkan status kualitas air (Optimal, Waspada, atau Kritis).

#### 5. Penyajian Visual

Status akhir dari hasil pemrosesan diekspose oleh *server* ke *dashboard* klien berbasis React.js menggunakan protokol WebSocket untuk memperbarui antarmuka pengguna secara *real-time* tanpa penundaan penyegaran halaman. Untuk kebutuhan pelaporan historis, *server* mengelola query dari Firestore, mengeksekusi ulang status *Rule-Based*, dan mentransmisikannya ke dalam format tabel data antarmuka.



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

#### 4. 2. 4 Alur Komunikasi Data

Alur komunikasi data pada sistem ini dirancang menggunakan mekanisme komunikasi hibrida yang menggabungkan dua protokol berbeda sesuai fungsinya masing-masing. Pada lapisan pengambilan data data, mikrokontroler ESP32 membaca parameter kualitas air secara berkala, mengonversi hasil pembacaan sensor menjadi data numerik, dan mengemas *payload* dalam format JSON. Data tersebut kemudian dikirimkan ke *broker* MQTT yang berjalan pada infrastruktur *Virtual Private Server* (VPS) menggunakan protokol MQTT, yang dipilih karena karakteristiknya yang ringan dan efisien untuk transmisi data IoT dengan *overhead* rendah.

Pada lapisan pemrosesan, *server backend* Node.js berperan sebagai *subscriber* yang secara aktif mendengarkan topik MQTT yang telah dikonfigurasi. Setiap kali *payload* diterima dari *broker*, *server* mengeksekusi dua jalur pemrosesan secara paralel dan asinkron. Jalur pertama menyimpan data sensor beserta *server-side timestamp* ke dalam basis data Cloud Firestore untuk keperluan pencatatan histori dan pelaporan. Jalur kedua mengeksekusi komputasi algoritma *Rule-Based* untuk mengevaluasi nilai pH dan TDS terhadap threshold agronomi yang telah ditetapkan, guna menghasilkan klasifikasi status kualitas air (Optimal, Waspada, atau Kritis).

Pada lapisan presentasi, hasil komputasi dari kedua jalur tersebut diteruskan secara seketika ke antarmuka web berbasis React.js melalui protokol WebSocket yang mempertahankan koneksi persisten antara *server* dan klien. Pendekatan *full-duplex* ini memungkinkan pembaruan data pada *dashboard* terjadi secara *real-time* tanpa mekanisme *polling*, sehingga pengguna dapat memantau kondisi larutan hidroponik dengan *delay* yang minimal.

#### 4. 2. 5 Perancangan Algoritma *Rule-Based*

Algoritma *Rule-Based* dirancang untuk mengevaluasi kualitas air nutrisi pada media tanam bayam berdasarkan parameter pH dan TDS. Berdasarkan landasan teoretis agronomi yang telah dijabarkan sebelumnya, pemrosesan *real-time* ini dieksekusi melalui dua tahapan perancangan teknis utama:

- Penetapan Kondisi (Kategorisasi Input)





### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Nilai numerik dari sensor dipetakan ke dalam kategori kondisi berdasarkan ambang batas berikut:

- Kondisi pH: Asam ( $< 6.0$ ), Normal ( $6.0 - 7.0$ ), Basa ( $> 7.0$ ).
- Kondisi Nutrisi (TDS/PPM): Rendah ( $< 800$ ), Normal ( $800 - 1500$ ), Tinggi ( $> 1500$ ).

### Evaluasi Aturan (*Rule Evaluation*)

Sistem pengambilan keputusan beroperasi berdasarkan matriks permutasi dari dua parameter *input* utama (pH dan TDS). Mengingat masing-masing parameter diklasifikasikan ke dalam tiga kondisi (3 variabel pH  $\times$  3 variabel TDS), sistem secara matematis menghasilkan 9 kombinasi aturan logika yang bersifat *mutually exclusive*. Algoritma memproses kombinasi tersebut menggunakan operator logika AND untuk menetapkan satu dari tiga status keluaran final (Optimal, Waspada, atau Kritis). Seluruh hasil permutasi aturan logika disajikan pada Tabel 4.3.

Tabel 4. 3 Aturan Algoritma *Rule-Based*

| No | Kondisi pH | Kondisi TDS | Status  |
|----|------------|-------------|---------|
| 1  | Asam       | Rendah      | Kritis  |
| 2  | Asam       | Normal      | Waspada |
| 3  | Asam       | Tinggi      | Kritis  |
| 4  | Normal     | Rendah      | Waspada |
| 5  | Normal     | Normal      | Optimal |
| 6  | Normal     | Tinggi      | Waspada |
| 7  | Basa       | Rendah      | Kritis  |
| 8  | Basa       | Normal      | Waspada |
| 9  | Basa       | Tinggi      | Kritis  |

### 4. 2. 6 Perancangan Basis Data

Perancangan basis data pada sistem ini diimplementasikan menggunakan arsitektur non-relasional (NoSQL) melalui layanan Cloud Firestore dari Google Firebase. Berbeda dengan basis data relasional tradisional yang mewajibkan deklarasi *Entity Relationship Diagram* (ERD) secara kaku, Firestore menerapkan model data

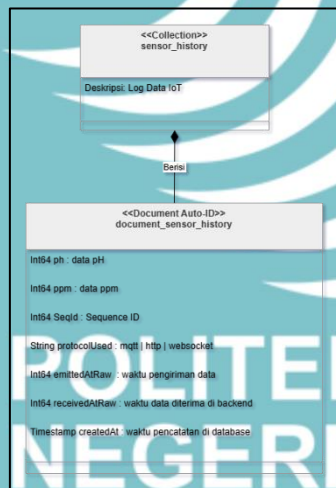


### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

*schemaless* (tanpa skema statis). Sebagaimana ditegaskan oleh Fernández Candel dkk. (2021), informasi skema pada basis data NoSQL bersifat implisit dan terenkapsulasi langsung di dalam struktur koleksi (*collection*), dokumen (*document*), dan *field*.

Pendekatan *Document-oriented* ini dipilih karena Firestore mendukung sinkronisasi proksimitas waktu-nyata (*real-time*) dengan *delay* ingestasi yang sangat rendah (Ayeni dan Adesoba, 2025), serta menawarkan fleksibilitas mutlak dalam menampung struktur *payload* JSON dinamis yang ditransmisikan oleh klien ESP32. Struktur data dirancang ke dalam dua koleksi utama yang divisualisasikan pada Gambar 4.4 dan Gambar 4.5.



Gambar 4. 4 Koleksi History

1. Koleksi *sensor\_history*, Koleksi ini bertugas menyimpan riwayat telemetri pembacaan sensor secara kontinu. Guna mendukung fungsionalitas keandalan jaringan dan *self-healing latency*, setiap dokumen di dalam koleksi ini dirancang memuat 7 (tujuh) *field* krusial:
  - *ph (Double)*: Nilai derajat keasaman air.
  - *ppm (Int64)*: Nilai kepekatan nutrisi terlarut / TDS.
  - *seqId (Int64)*: Nomor identitas sekuensial paket untuk mencegah duplikasi dan mendeteksi anomali *packet loss*.
  - *protocolUsed (String)*: Penanda jalur komunikasi asal (*routing tag*), bernilai "mqtt", "HTTP", atau "WebSocket" guna membedakan dataset pengujian.

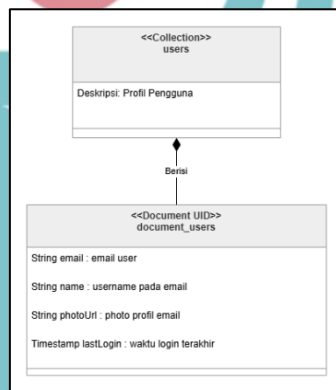




### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

- emittedAtRaw (*Int64*): Stempel waktu (*Unix Epoch Milliseconds*) yang dicetak secara lokal oleh osilator klien ESP32 tepat sebelum paket ditembakkan.
- receivedAtRaw (*Int64*): Stempel waktu komputasi mesin (*Epoch*) yang dicetak seketika oleh *server* Node.js saat *payload* berhasil ditangkap. Selisih antara atribut ini dengan emittedAtRaw menjadi dasar perhitungan *Cloud Ingestion Latency*.
- createdAt (*Timestamp*): Stempel waktu format kalender absolut yang dihasilkan oleh *server* Google Cloud untuk keperluan penyajian grafik pada antarmuka *browser*.



Gambar 4. 5 Koleksi Users

2. Koleksi *users*, Koleksi ini difungsikan untuk manajemen identitas (*Identity Access Management*). Data diintegrasikan langsung dengan otentikasi Google OAuth 2.0. Setiap dokumen memuat profil yang meliputi *field* name, email, photoURL, role (hak akses otorisasi admin/pengguna), serta lastLogin yang mencatat stempel waktu sesi terakhir pengguna guna keperluan audit keamanan *dashboard*.

#### 4. 2. 7 Perancangan Dashboard Web

Antarmuka pengguna sistem monitoring hidroponik dibangun sebagai *Single Page Application* (SPA) berbasis React.js, yang memungkinkan navigasi dan pembaruan data berlangsung secara instan tanpa *pepayload* ulang halaman. Sistem antarmuka ini terdiri dari dua halaman utama, yaitu halaman Login dan halaman *Dashboard*.



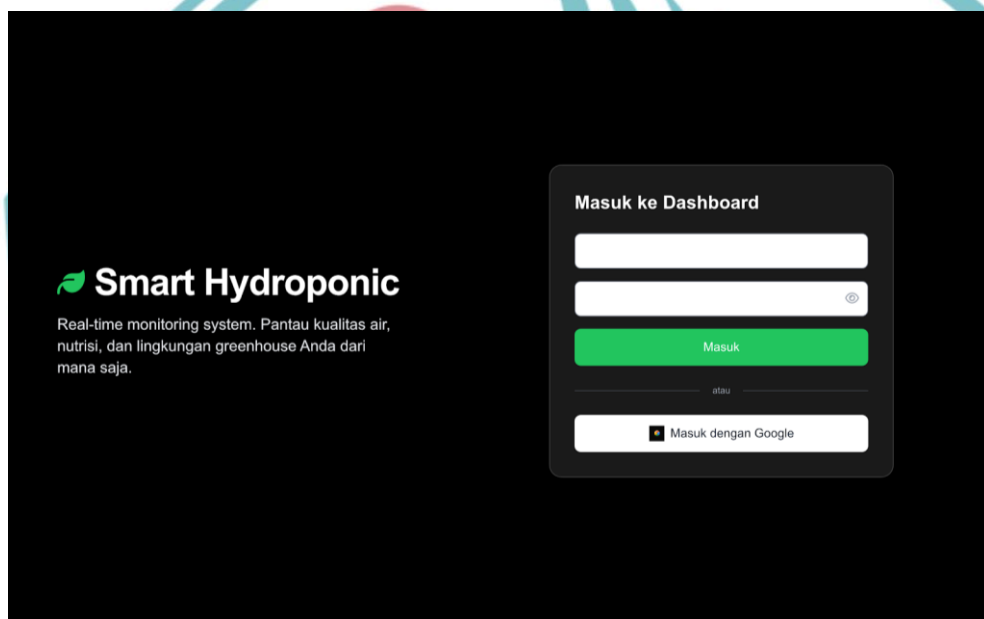
## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang menggunakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

### 1. Halaman Login

Halaman Login dirancang dengan tata letak dua kolom. Kolom kiri menampilkan identitas sistem berupa logo, nama aplikasi *Smart Hydroponic*, dan deskripsi singkat platform. Kolom kanan menyajikan formulir autentikasi yang menyediakan dua mekanisme masuk: autentikasi berbasis kredensial menggunakan nama pengguna dan kata sandi, serta autentikasi menggunakan akun Google melalui layanan Google Auth. Setelah autentikasi berhasil divalidasi oleh *backend*, pengguna diarahkan langsung ke halaman *Dashboard*.



Gambar 4. 6 Tampilan Halaman Login

### 2. Halaman *Dashboard*

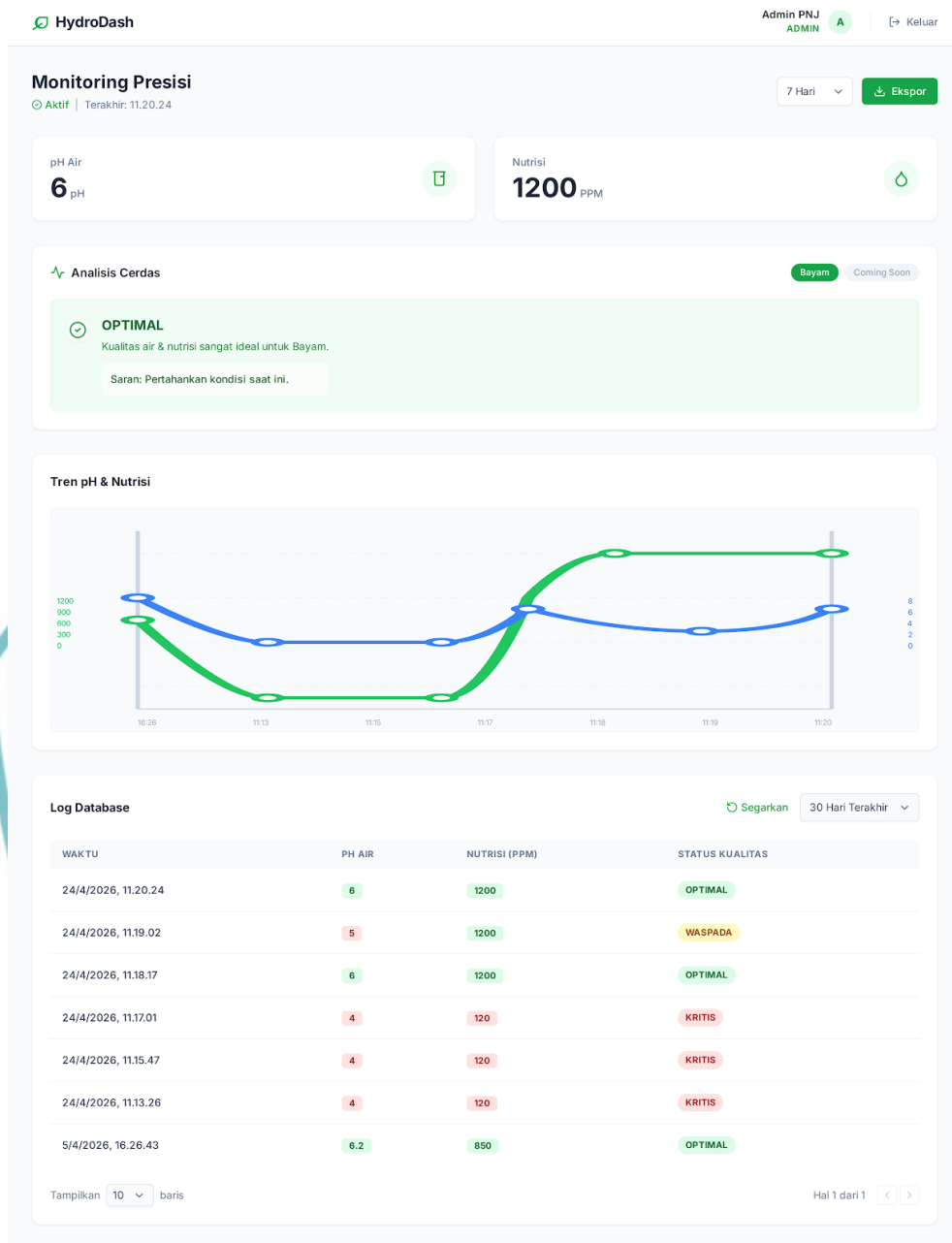
Halaman *Dashboard* merupakan antarmuka utama sistem yang menyajikan seluruh informasi *monitoring* secara terpadu dalam satu tampilan pada gambar 4.7. *Dashboard* terdiri dari empat komponen fungsional utama sebagai berikut.





### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta



Gambar 4. 7 Tampilan Halaman *Dashboard*

#### A. Panel Parameter Aktual

Komponen ini menampilkan nilai terkini parameter kualitas air yang diterima secara langsung melalui koneksi WebSocket dari *backend*. Terdapat dua kartu terpisah, yaitu kartu pH Air yang menampilkan nilai derajat keasaman larutan, dan kartu Nutrisi yang menampilkan konsentrasi nutrisi terlarut dalam satuan PPM. Kedua nilai diperbarui secara otomatis setiap kali data baru diterima dari perangkat ESP32 tanpa interaksi pengguna.



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

#### B. Panel *Decision Support*

Komponen ini menampilkan hasil evaluasi algoritma *Rule-Based* yang dieksekusi di sisi *server*. Panel menampilkan klasifikasi status kualitas air dalam tiga kategori — Optimal, Waspada, atau Kritis — disertai deskripsi kondisi dan saran tindakan yang relevan. Komponen ini juga dilengkapi selektor komoditas tanaman, yang pada implementasi saat ini tersedia untuk tanaman Bayam.

#### C. Grafik Tren pH dan Nutrisi

Komponen grafik menyajikan visualisasi historis nilai pH dan PPM dalam satu bidang koordinat dengan dua sumbu vertikal. Garis biru merepresentasikan tren nilai pH, sedangkan garis hijau merepresentasikan tren nilai PPM. Grafik diperbarui secara dinamis setiap kali data baru masuk melalui WebSocket, sehingga pergerakan parameter dapat diamati secara langsung dalam rentang waktu yang dapat dikonfigurasi melalui selektor periode (7 Hari, 30 Hari, dan sebagainya).

#### D. Tabel Riwayat (*History Log*)

Komponen ini menampilkan log historis pembacaan sensor yang diambil dari basis data Cloud Firestore. Setiap baris tabel memuat informasi waktu pencatatan, nilai pH, nilai nutrisi dalam PPM, serta status kualitas air hasil evaluasi *Rule-Based* yang divisualisasikan dengan label berwarna (hijau untuk Optimal, kuning untuk Waspada, dan merah untuk Kritis). Tabel dilengkapi fitur pengaturan jumlah baris per halaman, navigasi antar halaman, tombol segarkan, serta filter rentang waktu untuk mendukung kebutuhan analisis data historis.

### 4.3 Implementasi Sistem

Subbab ini memaparkan realisasi teknis dari perancangan sistem yang telah diuraikan pada bab sebelumnya. Pembahasan mencakup implementasi perangkat keras, penyiapan infrastruktur jaringan dan *server*, serta pembahasan perangkat lunak secara keseluruhan. Seluruh tahapan integrasi ini bertujuan untuk





#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

memastikan setiap komponen sistem dapat beroperasi secara fungsional sebelum memasuki tahap pengujian kinerja.

#### 4. 3. 1 Kebutuhan *Hardware*

Perangkat keras yang digunakan dalam penelitian ini terdiri dari komponen-komponen yang membentuk node sensor IoT pada sisi klien. Setiap modul dipilih berdasarkan kesesuaiannya dengan kebutuhan pengambilan data parameter kualitas air hidroponik secara *real-time*. Rincian spesifikasi perangkat keras yang digunakan disajikan pada Tabel 4.4.

Tabel 4. 4 Kebutuhan *Hardware*

| Perangkat Keras |                                                                                                                                                                     |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Modul           | Keterangan                                                                                                                                                          |
| ESP32           | Berfungsi sebagai pusat komputasi utama untuk memproses data pembacaan sensor, menjalankan logika kendali otomatis aktuator, dan mengelola transmisi data.          |
| Sensor TDS      | Digunakan untuk mengukur tingkat kepekatan padatan terlarut dalam tandon hidroponik.                                                                                |
| Sensor pH       | Berperan untuk mendeteksi tingkat keasaman atau kebasaan cairan dalam tandon hidroponik.                                                                            |
| LCD OLED        | Digunakan sebagai antarmuka visual lokal ( <i>on-site</i> ) untuk menampilkan angka pembacaan sensor dan status transisi operasional alat secara <i>real-time</i> . |

#### 4. 3. 2 Kebutuhan *Software*

Perangkat lunak yang digunakan dalam penelitian ini mencakup seluruh ekosistem pengembangan sistem, mulai dari lingkungan *runtime backend*, *framework*



*frontend*, layanan basis data *cloud*, protokol komunikasi IoT, hingga perangkat bantu pengujian dan analisis jaringan. Pemilihan setiap perangkat lunak didasarkan pada kesesuaiannya dengan arsitektur sistem yang dirancang serta kemampuannya dalam mendukung evaluasi kinerja QoS secara objektif. Rincian perangkat lunak yang digunakan disajikan pada Tabel 4.5.

Tabel 4. 5 Kebutuhan *Software*

| Perangkat Lunak              |                                                                                                                                                                                        |
|------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>Software dan Platform</i> | Keterangan                                                                                                                                                                             |
| Node.js (Express.js)         | Environment <i>runtime</i> yang digunakan untuk membangun <i>Backend</i> API dan memproses algoritma <i>Rule-Based</i> di sisi <i>server</i> .                                         |
| React.js (Vite)              | <i>Framework</i> JavaScript yang digunakan untuk membangun antarmuka pengguna ( <i>Frontend Dashboard</i> ) secara responsif berbasis arsitektur <i>Single Page Application</i> (SPA). |
| Cloud Firestore              | Basis data berbasis NoSQL untuk menyimpan riwayat data sensor secara dinamis dengan dukungan sinkronisasi <i>real-time</i> .                                                           |
| Eclipse Mosquitto            | Layanan <i>broker</i> MQTT yang berperan sebagai perantara pertukaran pesan asinkron antara mikrokontroler ESP32 dan <i>server backend</i> .                                           |
| Docker                       | Platform kontainerisasi yang digunakan untuk mengemas dan menjalankan aplikasi <i>backend</i> dan <i>frontend</i> pada infrastruktur VPS secara terisolasi, guna memastikan            |

#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta





### © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|                             |                                                                                                                                                                                                                                                                             |
|-----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                             | konsistensi lingkungan eksekusi antara tahap pengembangan dan produksi.                                                                                                                                                                                                     |
| Google Cloud Platform (GCP) | Infrastruktur Virtual Private Server berbasis <i>cloud</i> yang digunakan sebagai <i>server</i> utama untuk menerapkan aplikasi <i>backend</i> dan <i>frontend</i> agar dapat diakses secara publik melalui IP statis.                                                      |
| Visual Studio Code          | Text Editor / IDE utama yang digunakan untuk penulisan keseluruhan baris kode pemrograman.                                                                                                                                                                                  |
| Wireshark                   | Perangkat lunak network analyzer yang digunakan untuk menangkap dan menganalisis lalu lintas paket data secara <i>end-to-end</i> guna mengukur parameter QoS meliputi <i>delay</i> , <i>throughput</i> , <i>packet loss</i> , dan <i>jitter</i> berdasarkan standar TIPHON. |
| Apache JMeter               | Perangkat lunak pengujian beban ( <i>load testing</i> ) berbasis Java yang digunakan untuk mensimulasikan lonjakan permintaan serentak ke infrastruktur VPS guna mengukur ketahanan <i>server</i> berdasarkan <i>error rate</i> , utilisasi CPU, dan penggunaan RAM.        |

#### 4. 3. 3 Implementasi Perangkat Keras

Implementasi perangkat keras berfokus pada perakitan node sensor IoT yang ditempatkan pada area tandon air hidroponik. Komponen utama terdiri dari mikrokontroler ESP32 DevKit V1 yang dihubungkan dengan modul pengkondisi sinyal untuk Sensor pH-4502C dan Sensor TDS Analog. Informasi status operasional sistem ditampilkan secara on-site melalui layar LCD OLED 0,96 inci.



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumuk dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Realisasi fisik dari integrasi perangkat keras ini, mulai dari perakitan mikrokontroler secara makro hingga penempatan sensor di dalam tandon air, didokumentasikan secara visual pada Gambar 4.8.



Gambar 4. 8 Penempatan Node Sensor IoT pada Media Hidroponik

#### 4. 3. 4 Implementasi *Backend* dan Orkestrasi Protokol

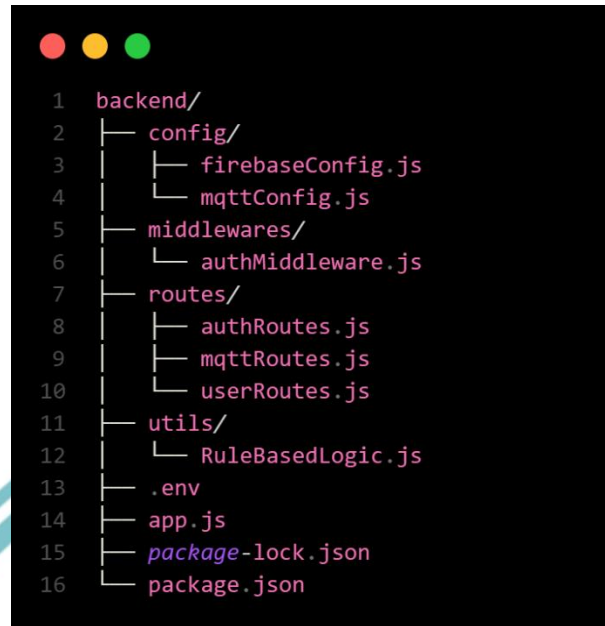
Arsitektur aplikasi *backend* pada sistem ini dibangun di atas fondasi lingkungan eksekusi Node.js. Implementasi kode difokuskan pada tiga pilar fungsional, pemusatan komputasi logika bisnis, penanganan lalu lintas pesan asinkron, dan mekanisme alih protokol (*protocol switching*). Arsitektur perangkat lunak disusun menggunakan pola desain berlapis (*layered architecture*) guna memastikan pemisahan tanggung jawab (*Separation of Concerns*) pada setiap modul komputasi beroperasi secara mandiri. Representasi fisik dari arsitektur mikro ini divisualisasikan pada Gambar 4.9.





### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta



Gambar 4. 9 Struktur File *Backend*

Berdasarkan struktur pada Gambar 4.9, pemisahan lapisan logika sistem direkayasa secara ketat. Direktori `config/` difokuskan secara eksklusif untuk menangani injeksi kredensial dan inisiasi sesi pihak ketiga (Google Firebase dan Eclipse Mosquitto). Keamanan titik akhir ditangani oleh direktori `middlewares/`, di mana modul `authMiddleware.js` bertindak sebagai interseptor yang memvalidasi *JSON Web Token* (JWT) pada setiap permintaan masuk HTTP, memblokir interaksi dari entitas yang tidak terotorisasi sebelum mencapai lapisan pengendali (*controller*). Lapisan perutean (`routes/`) memisahkan jalur komunikasi berdasarkan protokol, sementara seluruh kecerdasan sistem ditarik dari mikrokontroler dan dipusatkan ke dalam direktori `utils/` melalui modul `RuleBasedLogic.js`.

#### 1. Pemusatan Komputasi Algoritma *Rule-Based*

Keputusan arsitektural untuk memindahkan seluruh komputasi *Rule-Based* dari mikrokontroler (ESP32) ke *server* Node.js bertujuan untuk membebaskan utilitas memori pada perangkat keras di lapangan. Eksekusi algoritma dipusatkan pada modul `RuleBasedLogic.js` menggunakan pendekatan *Crisp Logic* yang memetakan masukan sensor ke dalam matriks keputusan 9 basis aturan (*9-Rule-Based*).

Siklus pemrosesan algoritma ini beroperasi melalui empat tahapan:



## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

- Validasi Input, *server* memverifikasi integritas muatan data dari sensor untuk memblokir komputasi jika nilai parameter kosong atau negatif.
- Kategorisasi Parameter, nilai derajat keasaman (pH) dipetakan menjadi himpunan pasti ASAM, NORMAL, atau BASA; sementara kepekatan nutrisi (PPM) dikelompokkan menjadi RENDAH, NORMAL, atau TINGGI.
- Evaluasi Matriks Keputusan, kedua variabel kategorikal diumpankan ke dalam matriks logika untuk mengevaluasi 9 kombinasi kemungkinan, sebagaimana ditunjukkan pada Gambar 4.10.

```
// Segmen Evaluasi Matriks Keputusan (9-Rule Base)
let statusAkhir = ''

if (kategoriPH === 'NORMAL' && kategoriPPM === 'NORMAL') {
  statusAkhir = 'OPTIMAL'
} else if (
  (kategoriPH === 'ASAM' && kategoriPPM === 'NORMAL') ||
  (kategoriPH === 'BASA' && kategoriPPM === 'NORMAL') ||
  (kategoriPH === 'NORMAL' && kategoriPPM === 'RENDAH') ||
  (kategoriPH === 'NORMAL' && kategoriPPM === 'TINGGI')
) {
  statusAkhir = 'WASPADA'
} else {
  // Jatuh pada kondisi ASAM/RENDAH, ASAM/TINGGI, BASA/RENDAH, BASA/TINGGI
  statusAkhir = 'KRITIS'
}
```

Gambar 4. 10 Implementasi Perhitungan *Rule-Based* pada Node.js

- Pembentukan *Output Payload*, berdasarkan variabel *statusAkhir* yang dihasilkan dari matriks di atas, arsitektur *backend* merangkai objek JSON komprehensif yang memuat deskripsi kondisi lingkungan air beserta rekomendasi tindakan spesifik bagi pengelola kebun.

Pemusatan komputasi matriks ini mengisolasi kompleksitas operasional dari perangkat ujung (*edge device*). Administrator sistem memegang kendali absolut untuk menyetel ulang batas toleransi (*threshold*) hidroponik langsung pada level *server* tanpa perlu melakukan pemrograman ulang pada unit mikrokontroler.

## 2. Orkestrasi Gatekeeper dan Dinamisasi Protokol Injeksi

Fitur paling esensial pada arsitektur *backend* ini terletak pada implementasi mekanisme *Protocol Switch* yang ditangani oleh berkas sentral *server.js*. *Server*





### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

dirancang untuk tidak membiarkan ketiga protokol (HTTP, WebSocket, MQTT) mengeksekusi penulisan data ke dalam pangkalan data secara acak dan bersamaan.

Sistem menerapkan variabel state `activeIngestProtocol` yang bertindak sebagai gerbang palang (*Gatekeeper*). Jalur protokol yang diizinkan untuk menulis (*ingest*) data sensor dikontrol secara dinamis melalui rute API khusus `/api/system/switch-protocol` seperti pada gambar berikut.

```
// server.js (Segmen Orkestrasi Gatekeeper)
let activeIngestProtocol = 'mqtt'; // Variabel State Palang Pintu

// API Pengendali Pemindahan Protokol
app.post("/api/system/switch-protocol", (req, res) => {
  const { protocol } = req.body;
  if (!['HTTP', 'WebSocket', 'mqtt'].includes(protocol)) {
    return res.status(400).json({ error: "Protokol tidak valid" });
  }
  activeIngestProtocol = protocol;
  res.json({ success: true, activeProtocol: activeIngestProtocol });
});

// Interseptor Injeksi Jalur HTTP
app.post("/api/ingest/sensors", (req, res) => {
  if (activeIngestProtocol !== 'HTTP') {
    return res.status(403).json({ error: "Akses Ditolak" });
  }
  processSensorData(req.body, 'HTTP');
  res.status(200).json({ success: true });
});
```

Gambar 4. 11 Implementasi Arsitektur *Gatekeeper* pada Node.js

Pada Gambar 4.11 dijelaskan bahwa arsitektur *gatekeeper* ini secara mutlak mengisolasi setiap sesi pengujian *Quality of Service* (QoS). Ketika antarmuka pengguna memerintahkan sistem untuk berpindah ke mode MQTT, interseptor pada Node.js akan langsung menolak koneksi masuk (*inbound*) dari rute REST HTTP maupun *payload* WebSocket dengan memberikan status *Akses Ditolak* (HTTP 403). Mekanisme ini menjamin bahwa seluruh data telemetri yang diekstraksi ke dalam pangkalan data Cloud Firestore memiliki kemurnian asal transmisi yang tervalidasi.

### 4. 3. 5 Implementasi *Frontend*

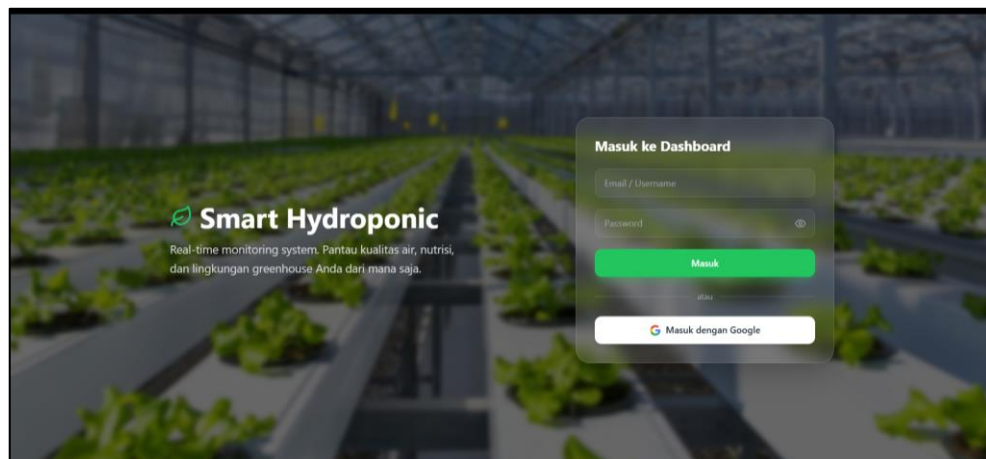
Antarmuka pengguna (*User Interface*) pada sistem ini direkayasa menggunakan pustaka React.js yang dikompilasi melalui *build tool* Vite. Kerangka kerja visual dibangun menggunakan utilitas Tailwind CSS untuk mencapai arsitektur *Single Page Application* (SPA) yang responsif dan ringan. Implementasi antarmuka dibagi



### Hak Cipta :

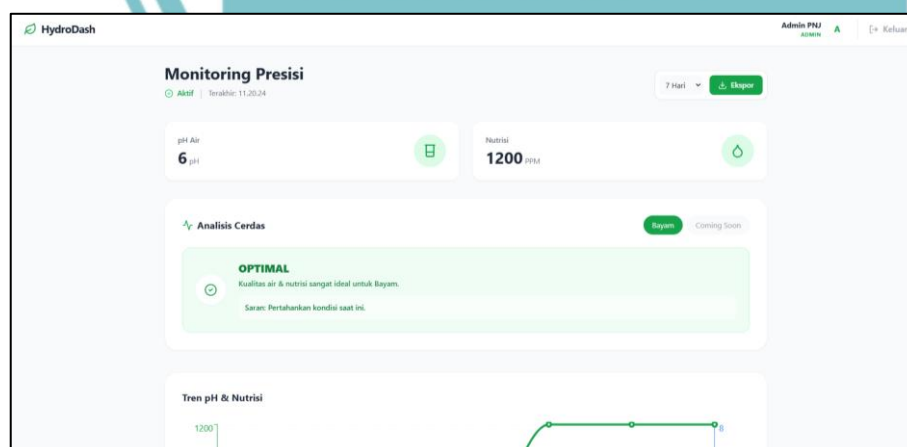
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

menjadi tiga segmen fungsional utama: gerbang autentikasi, *dashboard monitoring*, dan konfigurasi peringatan dini. Untuk menjamin integritas dan kerahasiaan data telemetri agrikultur, akses menuju *dashboard* sistem dibatasi secara ketat melalui halaman Login (Gambar 4.12).



Gambar 4. 12 Halaman *Login*

Halaman ini bertindak sebagai gerbang otorisasi tunggal yang mengintegrasikan protokol otentikasi Google OAuth 2.0 melalui layanan Firebase Authentication. Pendekatan ini memastikan bahwa hanya entitas dengan kredensial sah yang diberikan hak istimewa (*privilege*) untuk memantau dan mengendalikan operasional tandon hidroponik. Implementasi ini secara fundamental meruntuhkan celah keamanan (*vulnerability*) pada platform IoT konvensional yang rentan terhadap penyadapan lalu lintas data oleh pihak yang tidak berwenang. Setelah token autentikasi divalidasi oleh *server*, pengguna diarahkan menuju halaman *Dashboard* Utama (Gambar 4.13) yang bertindak sebagai pusat visual.



Gambar 4. 13 *Dashboard* Utama





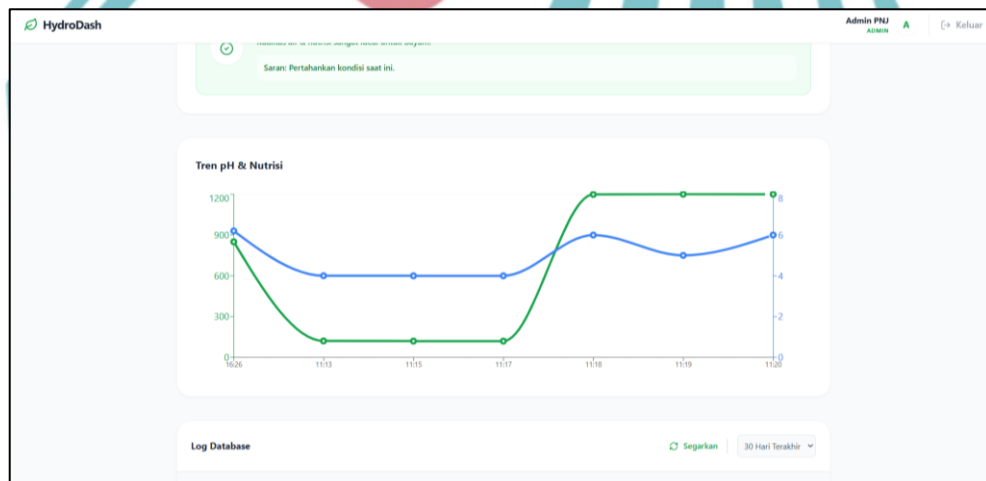
## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Arsitektur informasi pada *dashboard* ini dirancang untuk memberikan kesadaran situasional (*situational awareness*) yang instan bagi pengelola lahan:

1. Papan Metrik (*Metric Cards*), menampilkan nilai aktual derajat keasaman (pH) dan kepekatan nutrisi (PPM) secara *real-time* yang disuplai melalui terowongan WebSocket.
2. Sistem Pendukung Keputusan (*Decision Support Widget*), merupakan representasi visual dari algoritma *Rule-Based (Crisp Logic)* yang dieksekusi di *backend*. Pengguna langsung disajikan status kualitas air (Optimal, Waspada, atau Kritis) beserta rekomendasi aktuasi tanpa perlu menerjemahkan angka sensor secara manual.



Gambar 4. 14 Grafik pada *Dashboard*

Untuk analisis data historis jangka pendek, *dashboard* dilengkapi dengan komponen grafik tren (Gambar 4.14) yang dibangun menggunakan pustaka Recharts. Grafik ini menerapkan pembaruan dom asinkron (*asynchronous DOM rendering*), sehingga garis tren akan terus bergerak secara mulus setiap kali ada paket data baru tanpa memicu muat ulang halaman (*page reload*).



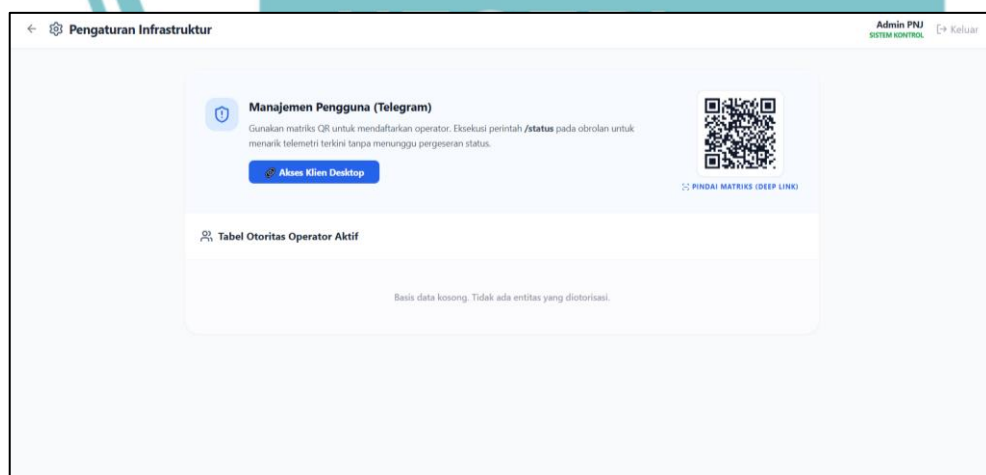
### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

| WAKTU               | PH AIR | NUTRISI (PPM) | STATUS KUALITAS |
|---------------------|--------|---------------|-----------------|
| 24/4/2026, 11:20:24 | 6      | 1200          | OPTIMAL         |
| 24/4/2026, 11:19:02 | 5      | 1200          | WASPADA         |
| 24/4/2026, 11:18:17 | 6      | 1200          | OPTIMAL         |
| 24/4/2026, 11:17:01 | 4      | 120           | KRITIS          |
| 24/4/2026, 11:15:47 | 4      | 120           | KRITIS          |
| 24/4/2026, 11:13:26 | 4      | 120           | KRITIS          |
| 5/4/2026, 16:26:43  | 6.2    | 850           | OPTIMAL         |

Gambar 4. 15 Tabel *History* pada *Dashboard*

Sebagai instrumen audit data, bagian bawah *dasboard* memuat Tabel Riwayat (*History Log*) yang terinkronisasi langsung dengan basis data Cloud Firestore (Gambar 4.15). Tabel ini mencatat *timestamp*, nilai sensor, dan status kualitas hasil kalkulasi. Guna memfasilitasi kebutuhan pelaporan manajerial pengelola kebun, tabel ini dilengkapi dengan utilitas *Export to Excel* yang memparsing data JSON menjadi format *spreadsheet* statis. Untuk melengkapi kapabilitas otomasi sistem, *frontend* menyediakan antarmuka khusus untuk melakukan pengikatan (*binding*) antara identitas sistem dengan platform perpesanan pihak ketiga, yaitu Telegram.



Gambar 4. 16 Antarmuka Konfigurasi Push Notification Telegram Bot

Melalui antarmuka pengaturan ini (Gambar 4.16), administrator dapat memasukkan parameter kredensial berupa *Telegram Chat ID*. Ketika modul ini diaktifkan,





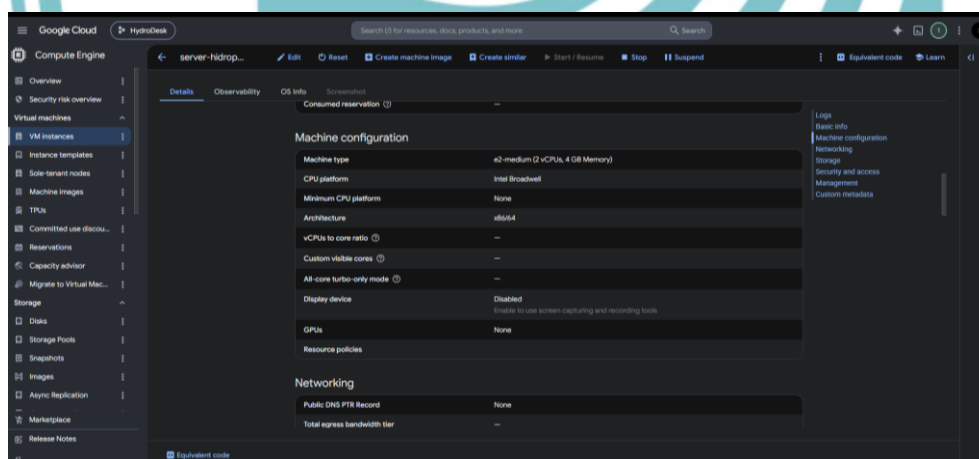
### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

parameter tersebut akan dikirimkan ke basis data *backend*. Hal ini mengizinkan modul *Gatekeeper* pada *server* Node.js untuk secara otomatis menembakkan pesan peringatan dini (*Push Notification*) ke perangkat seluler pengguna setiap kali algoritma *Rule-Based* mendeteksi bahwa kualitas air telah menyentuh batas ambang "KRITIS". Fitur ini memastikan pengelola tetap mendapatkan informasi kedaruratan secara seketika (*real-time*) meskipun browser web sedang ditutup.

### 4. 3. 6 Implementasi Infrastuktur Cloud

Seluruh modul arsitektur *server* yang telah dibangun pada tahap pengembangan disebarkan (*deployment*) ke dalam lingkungan produksi virtual. Infrastruktur komputasi *cloud* yang digunakan adalah layanan Virtual Machine (VM) pada Google Cloud Platform (GCP) dengan spesifikasi instans e2-medium yang beroperasi di atas sistem operasi Ubuntu *Server* 22.04 LTS, sebagaimana ditunjukkan pada Gambar 4.17.



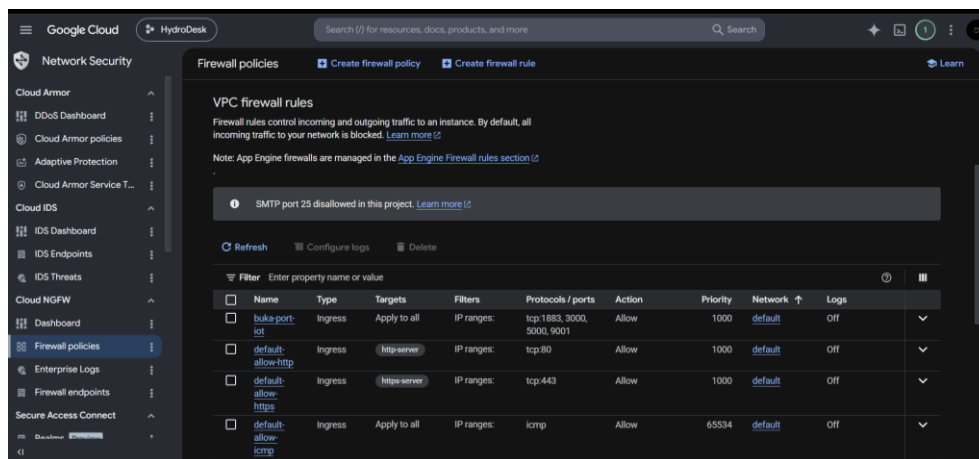
Gambar 4. 17 Overview VM GCP

Selanjutnya, untuk memastikan *server* dapat berkomunikasi secara aman dan selektif dengan perangkat ujung (*edge device*) maupun antarmuka klien, kontrol akses jaringan dikonfigurasi melalui kebijakan *Virtual Private Cloud* (VPC) *Firewall Rules*. Gambar 4.18 mendemonstrasikan panel konfigurasi *Ingress Traffic* yang menetapkan pembukaan *port* TCP spesifik, yaitu 1883, 3000, 5000, dan 9001. Alokasi *port* ini ditetapkan guna menopang mekanisme komunikasi hibrida pada pengujian riset, memfasilitasi lalu lintas telemetri MQTT, aliran data persisten WebSocket, serta antarmuka layanan *backend* secara terisolasi tanpa interferensi dari jaringan eksternal yang tidak berkepentingan.



### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta



Gambar 4. 18 Hasil Konfigurasi Firewall Rule VM

## 4. 4 Pengujian

### 4. 4. 1 Deskripsi Pengujian

Tahap pengujian dirancang untuk memenuhi tujuan penelitian dan mengevaluasi berbagai aspek dari arsitektur *Client-Server* yang dikembangkan. Terdapat lima metode pengujian utama yang dilakukan pada penelitian ini, yaitu:

#### 1. Pengujian Fungsionalitas

Melakukan pengujian integrasi sistem secara keseluruhan (*end-to-end*), mulai dari keberhasilan pembacaan sensor oleh ESP32, pengiriman data melalui masing-masing protokol yang diuji, hingga penampilan data pada *dashboard* pengguna. Pengujian dilakukan menggunakan metode *Black-box testing* dengan mengevaluasi kesesuaian keluaran sistem terhadap skenario masukan yang telah ditetapkan.

#### 2. Pengujian Algoritma *Rule-Based*

Pengujian ini bertujuan untuk memvalidasi presisi logika keputusan pada modul *backend*. Pengujian dieksekusi menggunakan metode *Black-box testing* dengan pendekatan *Decision table*. Validasi dilakukan melalui komparasi fungsional, yakni mencocokkan ke-9 kombinasi probabilitas nilai masukan sensor (pH dan TDS) terhadap luaran status kualitas air (Kritis, Waspada, Optimal) yang dihasilkan oleh *server*. Parameter keberhasilan diukur dari ketiadaan deviasi antara keluaran komputasi sistem dengan matriks kondisi ideal yang telah dirancang sebelumnya.





#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

### 3. Pengujian Quality of Service (QoS)

Mengevaluasi dan membandingkan kinerja transmisi jaringan ketiga protokol HTTP, MQTT, dan WebSocket menggunakan perangkat lunak Wireshark berdasarkan standar TIPHON, yang meliputi parameter *Throughput*, *Delay*, *Packet loss*, dan *Jitter*. Setiap protokol diuji sebanyak 5 kali percobaan dan hasilnya dirata-ratakan untuk menghasilkan nilai yang representatif.

### 4. Pengujian Realibilitas

Mengevaluasi kapabilitas *server* dalam mempertahankan ketersediaan layanan (*availability*), stabilitas komputasi, serta keutuhan transmisi data di bawah berbagai anomali tekanan. Mengacu pada model mutu ISO/IEC 25010:2023, pengujian ini dieksekusi melalui tiga skenario pengujian:

- a. Uji Toleransi Beban (*Load testing*), mensimulasikan hantaman kueri asinkron menggunakan Apache JMeter secara eksklusif pada jalur MQTT (*port* 1883) dengan pola *step-load* (50, 100, dan 200 *threads*) guna mengukur ambang batas *Error Rate*, utilisasi CPU, dan konsumsi RAM.
- b. Uji Ketahanan Jangka Panjang (*Maturity / Soak testing*), menginjeksi paket telemetri riil secara konstan setiap 5 detik selama 6 jam pengamatan guna membuktikan ketiadaan gejala anomali *Memory leak* pada *runtime* Node.js.
- c. Uji Daya Pulih Sistem (*Recoverability*), mengukur jeda waktu normalisasi otonom (*self-healing latency*) pasca-disimulasikannya interupsi lapisan fisik (pemutusan koneksi Wi-Fi klien selama 60 detik).

#### 4. 4. 2 Prosedur Pengujian

##### 1. Prosedur Pengujian Fungsionalitas

Prosedur pengujian fungsionalitas dilakukan menggunakan metode *Black-box testing* untuk menguji kesesuaian integrasi sistem secara *end-to-end*. Pengujian ini bertujuan memastikan kelancaran aliran data dalam arsitektur *Client-Server*, dimulai dari keberhasilan ESP32 membaca sensor secara stabil, kelancaran pengiriman *payload* melalui masing-masing protokol yang diuji, hingga verifikasi bahwa *server* Node.js mampu menerima dan menyajikan visualisasi data di *dashboard* React.js tanpa error atau sistem berhenti (*crash*). Setiap skenario



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

pengujian dicatat hasilnya dan dibandingkan dengan hasil yang diharapkan, kemudian dihitung persentase keberhasilannya menggunakan persamaan (1).

## 2. Prosedur Pengujian Algoritma *Rule-Based*

Prosedur pengujian ini bertujuan untuk memvalidasi keakuratan logika keputusan pada modul *backend* sekaligus memastikan integritas pemrosesan data secara *end-to-end*, yakni dari perangkat keras (alat) hingga ditampilkan pada antarmuka (web). Pengujian dieksekusi menggunakan metode *Black-box testing* dengan pendekatan *Decision table* melalui komparasi fungsional. Langkah-langkah pengujian ditetapkan sebagai berikut:

1. Menentukan skenario nilai masukan parameter pH dan PPM yang representatif, mencakup kondisi normal (nilai dalam rentang optimal), kondisi di ambang batas (threshold), serta kondisi ekstrem (nilai sangat rendah atau sangat tinggi), sehingga seluruh kategori status keluaran — Optimal, Waspada, dan Kritis — terwakili dalam cakupan pengujian.
2. Melakukan evaluasi status kualitas air secara manual dengan cara memetakan setiap nilai masukan ke dalam kategori kondisi pH (Asam/Normal/Basa) dan TDS (Rendah/Normal/Tinggi) berdasarkan ambang batas yang telah ditetapkan pada Tabel 4.3, kemudian mencocokkan kombinasi kondisi tersebut terhadap 9 basis aturan IF-THEN untuk memperoleh status akhir yang menjadi acuan referensi.
3. Membandingkan status hasil evaluasi manual dengan status klasifikasi yang dihasilkan secara otomatis oleh sistem melalui endpoint API *server backend*. Keluaran sistem diperoleh dengan mengirimkan nilai pH dan PPM sebagai parameter *request* ke API, kemudian mencatat *response payload* yang dikembalikan oleh modul *Rule-Based*.
4. Menyatakan algoritma valid apabila seluruh status keluaran sistem konsisten dan selaras dengan hasil evaluasi manual pada semua skenario yang diuji, dengan tingkat keberhasilan dihitung menggunakan persamaan (1).





#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

### 3. Prosedur Pengujian Sistem QoS

Pengujian QoS dilakukan untuk mengukur dan membandingkan kinerja transmisi jaringan ketiga protokol — HTTP, MQTT, dan WebSocket — secara *end-to-end* menggunakan Wireshark sebagai network analyzer. Setiap protokol diuji secara bergantian dalam kondisi infrastruktur yang identik untuk memastikan komparasi yang setara. Mengacu pada pendekatan yang diterapkan oleh Prasetyoaji dan Anifah (2025), setiap skenario pengujian diulang sebanyak 5 (lima) kali dan hasilnya dirata-ratakan untuk meminimalisasi pengaruh fluktuasi jaringan sesaat. Prosedur pengujian dilaksanakan sebagai berikut:

1. Menghubungkan perangkat ESP32 dan komputer monitor ke dalam satu jaringan yang dapat diamati, serta memastikan *server backend* pada VPS dalam kondisi aktif.
2. Mengonfigurasi *firmware* ESP32 sesuai protokol yang akan diuji pada sesi pengujian berjalan — HTTP POST *request* untuk pengujian HTTP, koneksi *publish* ke Eclipse Mosquitto untuk pengujian MQTT, dan koneksi WebSocket persisten ke *server* untuk pengujian WebSocket.
3. Menjalankan Wireshark pada komputer monitor dan memulai sesi packet capture pada antarmuka jaringan yang aktif. Untuk pengujian HTTP dan WebSocket, filter diterapkan pada alamat IP *server* VPS. Untuk pengujian MQTT, filter diterapkan pada *port* 1883 untuk menangkap lalu lintas antara ESP32 dan *broker* Eclipse Mosquitto.
4. Mengoperasikan sistem selama durasi pengamatan rata-rata 3,5 menit pada setiap sesi pengujian, di mana ESP32 mengirimkan data sensor secara berkala ke *server* melalui protokol yang sedang diuji.
5. Menghentikan sesi capture setelah durasi pengamatan selesai dan menyimpan berkas rekaman jaringan (.pcap) untuk keperluan analisis.
6. Mengekstraksi dan menghitung nilai *Throughput*, *Delay*, *Packet loss*, dan *Jitter* dari data capture menggunakan persamaan (2) hingga (5), kemudian mengklasifikasikan hasil setiap parameter berdasarkan kategori indeks standar TIPHON pada Tabel 2.3.



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

7. Mengulangi langkah 2 hingga 6 sebanyak 5 kali untuk setiap protokol, kemudian merata-ratakan keempat nilai parameter QoS dari seluruh percobaan untuk menghasilkan nilai akhir yang representative.

#### 4. Prosedur Pengujian Reliabilitas Sistem

Pengujian keandalan dieksekusi untuk mengukur batas toleransi kegagalan (*fault tolerance*), kematangan asinkronus (*maturity*), serta kapabilitas pemulihan mandiri (*recoverability*) dari infrastruktur *server* VPS di bawah berbagai skenario tekanan. Mengacu pada taksonomi model kualitas ISO/IEC 25010:2023, pengujian ini diselenggarakan melalui tiga tahapan prosedur sebagai berikut:

##### 1. Prosedur Uji Toleransi Beban (*Load testing*)

Pengujian ini bertujuan mengukur batas elastisitas dan titik infleksi degradasi komputasi *server* saat menerima hantaman kueri secara masif. Berdasarkan justifikasi ilmiah Nugraha dkk. (2024), pengujian difokuskan pada protokol MQTT menggunakan perangkat lunak Apache JMeter melalui pendekatan beban bertingkat (*step-load*). Prosedur eksekusi dijalankan melalui tahapan berikut:

- Menyiapkan infrastruktur VPS Ubuntu *Server* dan memastikan daemon *broker* Eclipse Mosquitto beroperasi secara aktif pada *port* standar 1883 tanpa pembatasan *file descriptors* di tingkat kernel.
- Membuka Apache JMeter pada komputer penguji dan menginstal *plugin MQTT Protocol Support* melalui JMeter Plugins Manager guna mengaktifkan komponen MQTT Connect dan MQTT Pub Sampler.
- Mengonfigurasi Thread Group untuk skenario Step-1 (50 threads / virtual users), dengan periode 10 detik, dan durasi penahanan puncak selama 60 detik. Alamat host diarahkan ke IP Publik VPS 34.101.33.145 pada *port* 1883 dengan target topik pengujian hydroponic/sensors.
- Menjalankan injeksi *payload* dan secara simultan memantau eskalasi beban komputasi *server* melalui terminal SSH menggunakan utilitas *htop*.
- Mengulangi langkah 3 dan 4 untuk skenario Step-2 (100 threads) dan Step-3 (200 threads), dengan menyisipkan jeda pendinginan (*cooldown window*)





#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

selama 5 menit antar-skenario guna mengembalikan ekuilibrium prosesor ke titik landai.

- Mengekstraksi metrik keluaran dari *Summary Report* JMeter, kemudian menghitung persentase kegagalan kueri (Error Rate) menggunakan Persamaan (6) untuk membuktikan apakah *server* mengalami Drop Connection pada hantaman 200 pengguna serentak.

#### 2. Prosedur Uji Ketahanan Jangka Panjang

Pengujian *Soak testing* dieksekusi untuk mendeteksi gejala anomali kebocoran memori (*Memory leak*) pada Node.js dan *broker* Mosquitto ketika dioperasikan dalam durasi panjang. Mengabaikan metode pengujian sintetis, pembebanan dilakukan menggunakan injeksi paket telemetri dari mikrokontroler pengirim.

Prosedur pengujian diselenggarakan sebagai berikut:

- Menyalakan perangkat keras mikrokontroler ESP32 klien yang telah dipasang *firmware* transmisi telemetri sensor dengan detak interval pengiriman konstan setiap 5 detik sekali.
- Membuka sesi terminal SSH menuju *server* VPS, kemudian membuat ruang isolasi latar belakang baru menggunakan perintah `tmux new -s soak_test`.
- Mengeksekusi skrip guna memerintahkan kernel melakukan pencatatan 10 parameter sistem (CPU, RAM Global, RSS Node.js, RSS Mosquitto, dan TCP Soket).
- Setelah periode 6 jam selesai, mengunduh berkas .csv.
- Melakukan audit rekonsiliasi data: menghitung rasio keandalan penerimaan paket menggunakan Persamaan (7), serta memplot kolom NodeJS\_RSS\_MB ke dalam grafik garis untuk memverifikasi ketiadaan eskalasi memori linear.

#### 3. Prosedur Uji Daya Pulih Sistem (*Recoverability*)

Pengujian ini dieksekusi untuk mengukur jeda waktu normalisasi otonom (*self-healing latency*) yang dibutuhkan oleh sistem untuk memulihkan jalur transmisi pasca-terjadinya interupsi tautan lapisan fisik (*link drop*) di sisi klien (Suryadi dkk., 2024). Mengingat topologi *node* sensor diletakkan pada area *greenhouse*



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

tertutup, pengujian difokuskan pada simulasi kehilangan sinyal (*blind-spot*). Prosedur pengujian dilakukan melalui tahapan berikut:

- Membiarkan mikrokontroler ESP32 memancarkan telemetri sensor secara normal menuju basis data Cloud Firestore melalui perantara *broker* lokal Mosquitto.
- Mematikan titik akses (router/*hotspot*) Wi-Fi klien secara mendadak serentak dengan mengaktifkan stopwatch. Penahanan status putus koneksi dipertahankan selama persis 60 detik guna memaksa tumpukan buffer soket TCP di sisi klien terputus.
- Tepat pada detik ke-60.00, menghidupkan kembali Wi-Fi dan mencatat koordinat waktu tersebut sebagai stempel waktu normalisasi ( $T_{\text{normalisasi}}$ ).
- Melakukan ekspor data historis dari Cloud Firestore ke dalam format Excel, memfilter baris yang memuat parameter "*is\_recovery*": true, dan mencatat stempel waktu kedatangannya di *server* sebagai stempel waktu mendarat ( $T_{\text{mendarat}}$ ).
- Menghitung selisih murni Recovery Time menggunakan Persamaan (8) guna memverifikasi apakah klien mampu melakukan pengikatan ulang sesi di bawah ambang batas toleransi 30 detik (Pratama dkk., 2023).

#### 4. 4. 3 Data Hasil Pengujian

Bagian ini memaparkan data mentah dan hasil pengukuran dari serangkaian pengujian yang telah dirancang sebelumnya. Data disajikan secara terstruktur sesuai dengan tiga tahapan prosedur pengujian yang telah dilakukan.

##### 1. Hasil Pengujian Fungsionalitas

Pengujian fungsionalitas dilakukan dengan metode *Black Box* untuk mengevaluasi aliran data *end-to-end* dari sisi perangkat keras hingga antarmuka pengguna. Tabel 4.6 menyajikan rekapitulasi skenario dan hasil integrasi sistem.





Tabel 4. 6 Pengujian Fungsionalitas Sistem

| No | Skenario Pengujian                                         | Hasil yang Diharapkan                                        | Hasil Aktual                                                                                                         | Kesimpulan |
|----|------------------------------------------------------------|--------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------|------------|
| 1  | ESP32 membaca sensor pH dan TDS                            | Nilai terbaca dengan benar                                   | Nilai parameter pH dan TDS terbaca secara akurat dan tampil pada Serial Monitor                                      | Berhasil   |
| 2  | ESP32 mempublikasikan data ke MQTT broker                  | Paket terkirim tanpa error                                   | <i>Payload</i> data terkirim ( <i>Publish</i> ) ke topik MQTT secara berkala tanpa kendala koneksi                   | Berhasil   |
| 3  | Server menerima data dari MQTT broker                      | Data diterima dengan benar                                   | Server backend (Node.js) menerima <i>payload</i> JSON secara utuh ( <i>Subscribe</i> )                               | Berhasil   |
| 4  | ESP32 mengirimkan data ke server menggunakan protokol HTTP | Permintaan ( <i>request</i> ) HTTP POST terkirim tanpa error | <i>Payload</i> data JSON terkirim melalui metode HTTP POST ke endpoint server secara berkala tanpa kegagalan koneksi | Berhasil   |
| 5  | Server menerima permintaan HTTP POST dari ESP32            | Data diterima dengan benar                                   | Server backend (Node.js) menerima dan mengekstrak <i>payload</i> JSON secara utuh melalui endpoint API.              | Berhasil   |
| 6  | ESP32 mengirimkan data melalui koneksi WebSocket           | Pesan data terkirim tanpa error                              | <i>Payload</i> data terkirim ke server secara berkala melalui koneksi persisten WebSocket tanpa kendala              | Berhasil   |

#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta



## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|    |                                                            |                                                     |                                                                                                                                            |          |
|----|------------------------------------------------------------|-----------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|----------|
| 7  | <i>Server</i> menerima aliran pesan WebSocket dari ESP32   | Data diterima dengan benar                          | <i>Server backend</i> (Node.js) menerima <i>payload</i> JSON secara utuh melalui modul <i>message handler</i> pada <i>server</i> WebSocket | Berhasil |
| 8  | <i>Server</i> menyimpan data ke Cloud Firestore            | Data tersimpan dengan <i>timestamp</i> akurat       | Dokumen baru berhasil direkam pada koleksi basis data dengan catatan waktu ( <i>timestamp</i> ) <i>server</i>                              | Berhasil |
| 9  | <i>Server</i> mengeksekusi algoritma <i>Rule-Based</i>     | Status kualitas air terklasifikasi dengan benar     | <i>Server</i> berhasil mengevaluasi nilai pH dan PPM serta memberikan <i>output</i> status sesuai basis aturan                             | Berhasil |
| 10 | <i>Server</i> mem- <i>broadcast</i> data via WebSocket     | Data diterima <i>client</i> secara <i>real-time</i> | <i>Payload</i> data berhasil diteruskan melalui jalur <i>socket</i> ke <i>frontend</i> secara instan                                       | Berhasil |
| 11 | <i>Dashboard</i> menampilkan nilai sensor <i>real-time</i> | Nilai pH dan PPM tampil dan diperbarui otomatis     | Antarmuka (UI) merespons dan memperbarui angka indikator secara dinamis tanpa perlu <i>refresh</i> halaman                                 | Berhasil |
| 12 | <i>Dashboard</i> menampilkan status <i>Rule-Based</i>      | Status dan rekomendasi tampil dengan benar          | Indikator status (Optimal/Waspada/Kritis) beserta teks rekomendasi tampil                                                                  | Berhasil |





### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|    |                                        |                                                   |                                                                                                           |          |
|----|----------------------------------------|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------|----------|
|    |                                        |                                                   | secara sinkron dengan hasil hitung <i>server</i>                                                          |          |
| 13 | Fitur <i>Export to Excel</i> berfungsi | File <i>Excel</i> terunduh dengan data yang benar | <i>File</i> berekstensi .xlsx berhasil diunduh dan memuat riwayat data yang sama persis dengan basis data | Berhasil |

## 2. Hasil Pengujian Algoritma *Rule-Based*

Pengujian algoritma *Rule-Based* dieksekusi menggunakan metode *Black-box testing* berbasis *Decision table*. Berdasarkan spesifikasi parameter yang telah ditetapkan pada Bab 3, ruang sampel masukan dipetakan ke dalam matriks kombinasi 3 x 3, yang melahirkan persis 9 (sembilan) permutasi logika unik dari himpunan derajat keasaman ( $S_{pH}$  = Asam, Normal, Basa) dan himpunan kepekatan nutrisi ( $S_{TDS}$  = Rendah, Normal, Tinggi).

Pengujian dilakukan dengan cara menginjeksi nilai batas ambang (*boundary values*) dari mikrokontroler menuju *endpoint API server*, kemudian mencatat *payload* respons yang dikembalikan oleh mesin keputusan (*decision engine*) Node.js. Hasil komparasi antara keluaran komputasi sistem terhadap matriks kebenaran ideal disajikan pada Tabel 4.7.

Tabel 4. 7 Pengujian Algoritma *Rule-Based*

| Uji | Input pH | Input PPM | Kondisi Parameter (Manual) | Status Perhitungan Manual | Status Sistem | Keterangan |
|-----|----------|-----------|----------------------------|---------------------------|---------------|------------|
| 1   | 5.2      | 650       | pH: Asam, PPM: Rendah      | Kritis                    | Kritis        | Valid      |
| 2   | 4.8      | 1100      | pH: Asam, PPM: Normal      | Waspada                   | Waspada       | Valid      |



## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|    |      |      |                            |         |         |       |
|----|------|------|----------------------------|---------|---------|-------|
| 3  | 3.5  | 1750 | pH: Asam,<br>PPM: Tinggi   | Kritis  | Kritis  | Valid |
| 4  | 6.5  | 500  | pH: Normal,<br>PPM: Rendah | Waspada | Waspada | Valid |
| 5  | 6.5  | 1200 | pH: Normal,<br>PPM: Normal | Optimal | Optimal | Valid |
| 6  | 6.2  | 1600 | pH: Normal,<br>PPM: Tinggi | Waspada | Waspada | Valid |
| 7  | 8.5  | 450  | pH: Basa,<br>PPM: Rendah   | Kritis  | Kritis  | Valid |
| 8  | 9.2  | 1400 | pH: Basa,<br>PPM: Normal   | Waspada | Waspada | Valid |
| 9  | 10.4 | 2100 | pH: Basa,<br>PPM: Tinggi   | Kritis  | Kritis  | Valid |
| 10 | 6.0  | 800  | pH: Normal,<br>PPM: Normal | Optimal | Optimal | Valid |

Mengacu pada rumusan Persamaan (1), tingkat keberhasilan fungsional dari logika *Rule-Based* dihitung melalui perbandingan jumlah skenario yang tervalidasi terhadap total populasi skenario pengujian:

$$\text{Tingkat Akurasi} = \left( \frac{9 \text{ Skenario Valid}}{9 \text{ Total Skenario Uji}} \right) \times 100\% = 100\%$$

Berdasarkan hasil rekapitulasi pada Tabel 4.7, algoritma *Rule-Based* yang diimplementasikan pada sisi *backend* Node.js mencetak tingkat akurasi sebesar 100%. Tidak terekam adanya anomali *logic drift* (pergeseran kesimpulan) maupun *unhandled exception* ketika sistem dihantam dengan kombinasi nilai ekstrem bawah maupun ekstrem atas. Hal ini membuktikan bahwa arsitektur logika IF-THEN yang dibangun telah sepenuhnya memenuhi kriteria *Functional Suitability*, sehingga data yang ditampilkan pada *Decision helper* dapat dipertanggungjawabkan keakuratannya.





### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

## 3. Hasil Pengujian Quality of Service (QoS)

### a. Pengujian QoS Protokol MQTT

Pengujian *Quality of Service* (QoS) pada arsitektur jalur MQTT dilakukan untuk mengukur tingkat keandalan, stabilitas, dan efisiensi *server* dalam menangani aliran paket telemetri sensor. Pengambilan metrik dilakukan secara hibrida, yakni penangkapan lalu lintas tingkat bawah menggunakan penganalisis protokol Wireshark pada lapisan jaringan (*network layer*) serta ekstraksi selisih stempel waktu (*timestamp*) pada lapisan aplikasi (*application layer*). Parameter unjuk kerja pertama yang diamati dalam skenario ini adalah *throughput*, di mana hasil pencatatannya disajikan pada Tabel 4.8 berikut.

Tabel 4. 8 Hasil Pengujian *Throughput* Protokol MQTT

| Percobaan ke- | Paket Diterima (Bytes) | Lama Pengamatan (Second) | <i>Throughput</i> (bits/second) |
|---------------|------------------------|--------------------------|---------------------------------|
| 1             | 14.203                 | 211,76                   | 536,57                          |
| 2             | 13.973                 | 206,44                   | 541,49                          |
| 3             | 13.973                 | 206,45                   | 541,47                          |
| 4             | 13.524                 | 205,88                   | 525,50                          |
| 5             | 13.886                 | 228,05                   | 487,13                          |
| Rata rata     | 13.911,8               | 211,71                   | 526,43                          |

Berdasarkan Tabel 4.8, hasil ekstraksi lalu lintas jaringan pasca-pemfilteran menunjukkan bahwa arsitektur pengumpulan data *server* melalui jalur komunikasi MQTT menghasilkan konsumsi lebar pita (*bandwidth*) yang sangat efisien dan ringan. Nilai *throughput* tertinggi tercatat pada percobaan ke-2 sebesar 541,49 bps, sedangkan nilai terendah mendarat pada percobaan ke-5 sebesar 487,13 bps akibat durasi pengamatan yang lebih panjang (228,05 detik). Secara keseluruhan, rata-rata *throughput* yang dihasilkan adalah sebesar 526,43 bps (atau ~0,53 kbps). Angka bita yang sangat rendah ini memberikan justifikasi empiris yang kuat terhadap keunggulan arsitektur MQTT untuk telemetri *Internet of things* (IoT); dengan ukuran *header* tetap yang sangat kecil (2 *bytes*) serta pemeliharaan satu sesi TCP



### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

persisten (*Keep-alive*), *server* mampu menerima pembaruan parameter sensor secara kontinu tanpa memicu kepadatan lalu lintas (*network congestion*) pada infrastruktur komputasi *cloud*. Berdasarkan regulasi standar TIPHON, kinerja transmisi data pada tingkat ini diklasifikasikan buruk.

Tabel 4. 9 Hasil Pengujian *Packet loss* Protokol MQTT

| Percobaan ke- | Paket Dikirim | Paket Diterima | <i>Packet loss</i> (%) |
|---------------|---------------|----------------|------------------------|
| 1             | 43            | 43             | 0,00                   |
| 2             | 42            | 42             | 0,00                   |
| 3             | 42            | 42             | 0,00                   |
| 4             | 41            | 41             | 0,00                   |
| 5             | 41            | 41             | 0,00                   |
| Rata rata     | 41,8          | 41,8           | 0,00                   |

Berdasarkan Tabel 4.9, pengujian integritas pengiriman paket telemetri pada tingkat aplikasi (*application layer*) mencetak angka kehilangan mutlak sebesar 0,00%. Dari rata-rata 41,8 paket yang dipancarkan oleh perangkat mikrokontroler ESP32 pada setiap sesi pengamatan, seluruhnya berhasil mendarat secara utuh dan dicatat ke dalam basis data Firestore tanpa ada satu pun urutan identitas (*seqId*) yang terlewat. Keandalan sempurna ini dicapai berkat implementasi QoS Level 1 pada protokol MQTT yang menerapkan mekanisme pengesahan (*acknowledgement*) PUBACK, dipadukan dengan keandalan protokol transport TCP yang melakukan pengiriman ulang otomatis (*retransmission*) jika terjadi pentalan paket di jaringan publik. Mengacu pada standarisasi telekomunikasi TIPHON (di mana persentase kehilangan < 3 % menduduki Indeks Kategori 4), tingkat *packet loss* 0,00% diklasifikasikan sebagai Sangat Bagus (Excellent).

Tabel 4. 10 Hasil Pengujian *Delay* Protokol MQTT

| Percobaan ke- | Total Paket | Total <i>Delay</i><br>(Second) | Rata-rata <i>Delay</i><br>(ms) |
|---------------|-------------|--------------------------------|--------------------------------|
| 1             | 43          | 7,16                           | 166,51                         |
| 2             | 42          | 7,56                           | 180,10                         |





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|           |      |      |        |
|-----------|------|------|--------|
| 3         | 42   | 7,53 | 179,33 |
| 4         | 41   | 7,29 | 177,76 |
| 5         | 41   | 7,71 | 188,12 |
| Rata rata | 41,8 | 7,45 | 178,36 |

Berdasarkan Tabel 4.10, hasil ekstraksi selisih stempel waktu pemancaran perangkat mentah (*emitted\_at\_raw*) dan ketibaan di gerbang logika (*received\_at\_raw*) menunjukkan rata-rata keterlambatan respons sebesar 178,36 ms. Nilai *delay* tercepat diperoleh pada percobaan ke-1 sebesar 166,51 ms, sedangkan waktu tempuh terlama terjadi pada percobaan ke-5 sebesar 188,12 ms. Angka *delay* di bawah 200 milidetik ini sangat impresif mengingat metrik ini mengukur *End-to-end Application Delay*, yang mencakup rambatan fisik jaringan internet menuju *server cloud* GCP, pemrosesan topik oleh *broker* Mosquitto, hingga komputasi injeksi dokumen ke Firestore. Mengacu pada standar TIPHON yang menempatkan rentang 150 ms–300 ms pada Kategori 3, kinerja *delay* arsitektur MQTT ini dinilai Bagus (*Good*).

Tabel 4. 11 Hasil Pengujian *Jitter* Protokol MQTT

| Percobaan ke- | Total Paket | Total Variasi <i>Delay</i><br>(Second) | Rata-rata <i>Jitter</i><br>(ms) |
|---------------|-------------|----------------------------------------|---------------------------------|
| 1             | 43          | 1,29                                   | 30,79                           |
| 2             | 42          | 1,40                                   | 34,22                           |
| 3             | 42          | 1,55                                   | 37,68                           |
| 4             | 41          | 1,13                                   | 28,25                           |
| 5             | 41          | 1,55                                   | 38,65                           |
| Rata rata     | 41,8        | 1,38                                   | 33,92                           |

Berdasarkan Tabel 4.11, fluktuasi variansi kedatangan antar-paket (*packet delay variation*) pada arsitektur MQTT mencetak nilai rata-rata sebesar 33,92 ms. Rentang fluktuasi bergerak dari titik terendah pada percobaan ke-4 sebesar 28,25 ms hingga puncaknya pada percobaan ke-5 sebesar 38,65 ms. Ketatnya angka *jitter* di kisaran 33 milidetik ini membuktikan bahwa antrian pemrosesan asinkron pada



### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

server Node.js serta rute jaringan internet publik beroperasi secara konsisten tanpa mengalami gejala penyumbatan (*bottleneck*) yang berarti. Berdasarkan parameter regulasi TIPHON, di mana nilai *jitter* yang mendarat pada rentang 0 ms–75 ms diklasifikasikan ke dalam Indeks tertinggi (Kategori 3), maka stabilitas aliran paket telemetry MQTT secara resmi dinyatakan Bagus (*Good*). Data mentah hasil pengukuran seluruh percobaan pengujian QoS protokol MQTT, mencakup berkas tangkapan lalu lintas jaringan Wireshark serta rekaman stempel waktu basis data, dapat dilihat secara lengkap pada Lampiran 4.

### b. Pengujian QoS Protokol HTTP

Pengujian unjuk kerja *Quality of Service* (QoS) tahap kedua dieksekusi pada arsitektur jalur komunikasi HTTP (*Hypertext Transfer Protocol*). Berbeda dengan protokol MQTT yang beroperasi secara asinkron melalui koneksi persisten yang ringan, HTTP menerapkan model arsitektur *Request-response* konvensional. Pengujian ini dieksekusi sebanyak 5 (lima) kali replikasi percobaan dengan rentang durasi pengamatan rata-rata 3,5 menit per sesi. Evaluasi kinerja dititikberatkan pada empat parameter utama, yaitu *Throughput*, *Packet loss*, *Delay*, dan *Jitter*, menggunakan kombinasi pembacaan lalu lintas jaringan mentah di Wireshark serta ekstraksi stempel waktu pada lapisan basis data Firestore. Parameter unjuk kerja pertama yang diamati dalam skenario ini adalah *throughput*, di mana hasil pencatatannya disajikan pada Tabel 4.12 berikut.

Tabel 4. 12 Hasil Pengujian *Throughput* Protokol HTTP

| Percobaan ke- | Paket Diterima<br>(Bytes) | Lama Pengamatan<br>(Second) | <i>Throughput</i><br>(bits/second) |
|---------------|---------------------------|-----------------------------|------------------------------------|
| 1             | 254.610                   | 208,53                      | 9.767,76                           |
| 2             | 275.160                   | 219,33                      | 10.036,25                          |
| 3             | 261.969                   | 206,05                      | 10.171,08                          |
| 4             | 273.987                   | 207,87                      | 10.544,50                          |
| 5             | 260.436                   | 207,70                      | 10.031,38                          |
| Rata rata     | 265.232,4                 | 209,90                      | 10.110,19                          |





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Berdasarkan Tabel 4.12, unjuk kerja *throughput* pasca-pemfilteran pada *server* menggunakan jalur komunikasi HTTP mencetak rata-rata konsumsi lebar pita (*bandwidth*) sebesar 10.110,19 bps (~10,11 kbps). Nilai *throughput* tertinggi terjadi pada percobaan ke-4 sebesar 10.544,50 bps, sedangkan konsumsi terendah terekam pada percobaan ke-1 sebesar 9.767,76 bps. Kunci justifikasi akademis yang sangat krusial terekam di sini: meskipun angka terfilter ini telah menyusut tajam dari pembacaan kotor sebelumnya (~33 kbps), *throughput* HTTP ini tetap ~19 kali lipat lebih boros dibandingkan protokol MQTT (526,43 bps). Kesenjangan ekstrem ini membuktikan secara tak terbantahkan besarnya beban *payload* bawaan (*header overhead*) pada arsitektur HTTP; setiap pengiriman satu bingkai telemetri sensor wajib mengangkut teks *header plaintext* yang tebal (mencakup atribut *Host*, *Content-Type*, *User-Agent*, hingga *Bearer Token*) serta memicu rotasi pembukaan soket TCP. Mengacu pada standar TIPHON, *payload* bita ini dikategorikan buruk secara kapasitas saluran, namun membuktikan ketidakefisienan struktur *payload* jika diterapkan pada ekosistem IoT masif.

Tabel 4. 13 Hasil Pengujian *Packet loss* Protokol HTTP

| Percobaan ke- | Paket Dikirim | Paket Diterima | <i>Packet loss</i> (%) |
|---------------|---------------|----------------|------------------------|
| 1             | 41            | 41             | 0,00                   |
| 2             | 41            | 41             | 0,00                   |
| 3             | 41            | 41             | 0,00                   |
| 4             | 42            | 42             | 0,00                   |
| 5             | 42            | 42             | 0,00                   |
| Rata rata     | 41,4          | 41,4           | 0,00                   |

Berdasarkan Tabel 4.13, pengujian tingkat kehilangan bingkai (*packet loss*) pada jalur komunikasi HTTP di tingkat aplikasi mencetak angka mutlak sebesar 0,00% di seluruh sesi replikasi. Dari rata-rata 41,4 paket telemetri yang dipancarkan oleh mikrokontroler ESP32 per sesi pengamatan, seluruhnya berhasil mendarat dan dicatat ke dalam Firestore tanpa ada satu pun urutan data (*seqId*) yang hilang. Keandalan sempurna ini merupakan kontribusi langsung dari arsitektur lapisan



transport TCP yang mempertahankan soket terbuka secara persisten (*persistent socket*), sehingga mengeliminasi kegagalan alokasi memori atau penolakan jabatan yang sering terjadi pada protokol non-persisten saat rute jaringan mengalami fluktuasi. Berdasarkan regulasi standar TIPHON (di mana nilai *packet loss* < 3% menduduki Indeks Kategori 4), kinerja keandalan WebSocket ini diklasifikasikan sebagai Sangat Bagus (*Excellent*).

Tabel 4. 14 Hasil Pengujian *Delay* Protokol HTTP

| Percobaan ke- | Total Paket | Total <i>Delay</i><br>(Second) | Rata-rata <i>Delay</i><br>(ms) |
|---------------|-------------|--------------------------------|--------------------------------|
| 1             | 41          | 23,12                          | 563,93                         |
| 2             | 41          | 22,95                          | 559,63                         |
| 3             | 41          | 24,51                          | 597,80                         |
| 4             | 42          | 24,79                          | 590,24                         |
| 5             | 42          | 23,19                          | 552,17                         |
| Rata rata     | 41,4        | 23,71                          | 572,75                         |

Berdasarkan Tabel 4.14, hasil ekstraksi selisih waktu tempuh *end-to-end* tingkat aplikasi pada jalur HTTP mencetak rata-rata *delay* sebesar 572,75 ms (lebih dari setengah detik). Nilai keterlambatan terendah diperoleh pada percobaan ke-5 sebesar 552,17 ms, sedangkan waktu tempuh puncak mendarat pada percobaan ke-3 sebesar 597,80 ms. Jika disandingkan dengan kinerja MQTT (178,36 ms), respons HTTP terbukti melonjak lebih dari 3 kali lipat lebih lambat. Kesenjangan waktu ini menyingkap kelemahan mendasar HTTP untuk sistem waktu-nyata (*real-time*); sebelum paket masuk ke gerbang logika penyeleksi (*Gatekeeper*), setiap transmisi harus melalui fase *Three-Way Handshake*, pembongkaran *middleware* otentikasi Express.js, hingga proses penulisan objek ke basis data Firestore. Berdasarkan standar TIPHON, angka 572,75 ms telah melampaui batas atas Kategori 1 (>450 ms) dan masuk ke dalam klasifikasi buruk.

#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta



Tabel 4. 15 Hasil Pengujian *Jitter* Protokol HTTP

| Percobaan ke- | Total Paket | Total Variasi <i>Delay</i><br>(Second) | Rata-rata <i>Jitter</i><br>(ms) |
|---------------|-------------|----------------------------------------|---------------------------------|
| 1             | 41          | 1,82                                   | 45,45                           |
| 2             | 41          | 2,21                                   | 55,23                           |
| 3             | 41          | 2,36                                   | 59,08                           |
| 4             | 42          | 1,17                                   | 28,56                           |
| 5             | 42          | 1,17                                   | 28,41                           |
| Rata rata     | 41,4        | 1,75                                   | 43,35                           |

Berdasarkan Tabel 4.15, fluktuasi variansi kedatangan antar-paket (*packet delay variation*) pada arsitektur HTTP mencetak nilai rata-rata sebesar 43,35 ms. Rentang variansi bergerak dari titik terendah pada percobaan ke-5 sebesar 28,41 ms hingga puncaknya pada percobaan ke-3 sebesar 59,08 ms. Membengkaknya angka *jitter* pada HTTP dibandingkan MQTT (33,92 ms) disebabkan oleh rotasi pemutusan dan pembangunan ulang koneksi TCP secara reguler, dipadukan dengan dinamika penanganan *event loop* asinkron pada *server* Node.js saat merespons lalu lintas kueri basis data. Berdasarkan parameter regulasi standar TIPHON, di mana nilai *jitter* yang mendarat pada rentang 0 ms–50 ms menempatkan kinerja variansi ke dalam Kategori 3, maka stabilitas transmisi protokol HTTP secara agregat dinyatakan Bagus (*Good*). Seluruh data mentah hasil pengukuran percobaan pengujian QoS protokol HTTP tersedia pada Lampiran 5.

#### c. Pengujian QoS Protokol WebSocket

Pengujian unjuk kerja *Quality of Service* (QoS) tahap ketiga sekaligus terakhir dieksekusi pada arsitektur jalur komunikasi WebSocket. Protokol ini menawarkan paradigma komunikasi dua arah secara penuh (*full-duplex*) dan persisten di atas satu koneksi TCP, yang dirancang untuk mengatasi kelemahan *delay* pada HTTP konvensional. Pengujian dieksekusi sebanyak 5 (lima) kali replikasi percobaan dengan rentang durasi pengamatan rata-rata 3,5 menit per sesi. Evaluasi kinerja mencakup empat parameter utama, yaitu *Throughput*, *Packet loss*, *Delay*, dan *Jitter*, menggunakan kombinasi penangkapan lalu lintas tingkat bawah di Wireshark serta



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta



### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

ekstraksi selisih waktu di basis data Firestore. Parameter unjuk kerja pertama yang diamati dalam skenario ini adalah *throughput*, di mana hasil pencatatannya disajikan pada Tabel 4.16 berikut.

Tabel 4. 16 Hasil Pengujian *Throughput* Protokol WebSocket

| Percobaan ke- | Paket Diterima (Bytes) | Lama Pengamatan (Second) | <i>Throughput</i> (bits/second) |
|---------------|------------------------|--------------------------|---------------------------------|
| 1             | 43.394                 | 212,98                   | 1.629,96                        |
| 2             | 43.288                 | 208,22                   | 1.663,20                        |
| 3             | 44.258                 | 213,22                   | 1.660,55                        |
| 4             | 44.302                 | 213,18                   | 1.662,49                        |
| 5             | 44.302                 | 213,25                   | 1.662,00                        |
| Rata rata     | 43.908,8               | 212,17                   | 1.655,64                        |

Berdasarkan Tabel 4.16, unjuk kerja *throughput* pasca-pemfilteran pada *server* menggunakan jalur komunikasi WebSocket mencetak rata-rata konsumsi lebar pita (*bandwidth*) sebesar 1.655,64 bps (atau ~1,66 kbps). Nilai *throughput* tertinggi diraih pada percobaan ke-2 sebesar 1.663,20 bps, sedangkan konsumsi terendah terekam pada percobaan ke-1 sebesar 1.629,96 bps. Hasil bersih ini memberikan landasan komparasi arsitektural yang sangat brilian: *throughput* WebSocket terbukti ~6 kali lipat lebih hemat bita dibandingkan HTTP (10.110,19 bps), meskipun sedikit di atas MQTT (526,43 bps). Efisiensi masif terhadap HTTP ini tercipta karena setelah jabat tangan pembuka (HTTP *Upgrade*) selesai, seluruh telemetri sensor dibungkus ke dalam bingkai biner (WebSocket *frames*) ber- *header* sangat tipis (2 hingga 6 bita) tanpa perlu mengangkut teks *plaintext header* pada setiap transmisi. Mengacu pada standarisasi TIPHON, kinerja aliran data ini diklasifikasikan sebagai *Buruk* dan mengukuhkan WebSocket sebagai jembatan komunikasi dua arah paling efisien untuk *dashboard monitoring* berbasis web.

Tabel 4. 17 Hasil Pengujian *Packet loss* Protokol WebSocket

| Percobaan ke- | Paket Dikirim | Paket Diterima | <i>Packet loss</i> (%) |
|---------------|---------------|----------------|------------------------|
|---------------|---------------|----------------|------------------------|





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|           |      |      |      |
|-----------|------|------|------|
| 1         | 43   | 43   | 0,00 |
| 2         | 42   | 42   | 0,00 |
| 3         | 43   | 43   | 0,00 |
| 4         | 44   | 44   | 0,00 |
| 5         | 43   | 43   | 0,00 |
| Rata rata | 43,0 | 43,0 | 0,00 |

Berdasarkan Tabel 4.17, pengujian tingkat kehilangan bingkai (*packet loss*) pada jalur komunikasi WebSocket mencetak angka mutlak sebesar 0,00% di seluruh sesi replikasi. Dari rata-rata 43 paket yang melintasi jaringan menuju *server* komputasi *cloud* GCP, seluruhnya berhasil divalidasi tanpa ada satu pun paket yang gugur. Keandalan sempurna ini merupakan kontribusi langsung dari arsitektur lapisan transport TCP yang mempertahankan soket terbuka secara terus-menerus (*persistent socket*), sehingga menghilangkan risiko kegagalan alokasi memori yang sering terjadi saat jabat tangan TCP baru diinisialisasi secara masif. Berdasarkan regulasi standar TIPHON (di mana nilai *packet loss* < 3% menduduki Indeks Kategori 3), kinerja WebSocket ini diklasifikasikan sebagai Bagus (*Good*).

Tabel 4. 18 Hasil Pengujian *Delay* Protokol WebSocket

| Percobaan ke- | Total Paket | Total <i>Delay</i><br>(Second) | Rata-rata <i>Delay</i><br>(ms) |
|---------------|-------------|--------------------------------|--------------------------------|
| 1             | 43          | 8,38                           | 194,91                         |
| 2             | 42          | 8,29                           | 197,48                         |
| 3             | 43          | 7,95                           | 184,79                         |
| 4             | 44          | 8,67                           | 196,95                         |
| 5             | 43          | 7,33                           | 170,40                         |
| Rata rata     | 43,0        | 8,12                           | 188,90                         |

Berdasarkan Tabel 4.18, hasil ekstraksi selisih waktu tempuh *end-to-end* tingkat aplikasi pada jalur WebSocket mencetak rata-rata *delay* sebesar 188,90 ms. Nilai keterlambatan terendah diraih pada percobaan ke-5 sebesar 170,40 ms, sedangkan waktu tempuh puncak mendarat pada percobaan ke-2 sebesar 197,48 ms.



### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

WebSocket berhasil memangkas keterlambatan respons HTTP (572,75 ms) secara drastis hingga 67,01% dan menghasilkan unjuk kerja waktu-nyata (*real-time*) yang sangat kompetitif menyamai kecepatan MQTT (178,36 ms). Ketiadaan proses pembukaan dan penutupan koneksi TCP pada setiap ketibaan *payload* membuat eksekusi pemfilteran di gerbang logika (*Gatekeeper*) berjalan nyaris seketika. Berdasarkan standar TIPHON, kinerja *delay* WebSocket ini diklasifikasikan sebagai Bagus (*Good*).

Tabel 4. 19 Hasil Pengujian *Jitter* Protokol WebSocket

| Percobaan ke- | Total Paket | Total Variasi <i>Delay</i><br>(Second) | Rata-rata <i>Jitter</i><br>(ms) |
|---------------|-------------|----------------------------------------|---------------------------------|
| 1             | 43          | 1,39                                   | 33,07                           |
| 2             | 42          | 1,56                                   | 37,98                           |
| 3             | 43          | 1,61                                   | 38,26                           |
| 4             | 44          | 1,40                                   | 32,49                           |
| 5             | 43          | 1,50                                   | 35,71                           |
| Rata rata     | 43,0        | 1,49                                   | 35,50                           |

Berdasarkan Tabel 4.19, fluktuasi variansi keterlambatan antar-paket (*packet delay variation*) pada arsitektur WebSocket mencetak nilai rata-rata sebesar 35,50 ms. Rentang variansi bergerak sangat tipis dan konsisten, mulai dari titik terendah pada percobaan ke-4 sebesar 32,49 ms hingga puncaknya pada percobaan ke-3 sebesar 38,26 ms. Stabilitas angka *jitter* yang mendarat di kisaran 35 milidetik ini membuktikan bahwa jalur komunikasi dua arah persisten mampu menjaga keteraturan antrean masuk di dalam *thread* pemrosesan Node.js tanpa memicu kemacetan lalu lintas jaringan. Mengacu pada regulasi standar TIPHON (di mana rentang 0 ms – 50 ms menduduki Indeks 3, stabilitas variansi transmisi WebSocket secara resmi dinyatakan Bagus (*Good*)). Seluruh data mentah hasil pengukuran percobaan pengujian QoS protokol WebSocket tersedia pada Lampiran 6.

#### 4. Hasil Pengujian Reliabilitas Sistem

Pengujian keandalan dieksekusi guna memvalidasi determinasi *server* dalam mempertahankan ketersediaan layanan (*availability*), stabilitas alokasi memori,





### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

serta kemampuan pemulihan mandiri di bawah berbagai anomali tekanan. Berdasarkan kerangka kerja ISO/IEC 25010:2023, pemaparan hasil pengujian ini diklasifikasikan ke dalam tiga dimensi pengujian sebagai berikut:

#### 1. Hasil Pengujian Toleransi Beban (*Load testing*)

Simulasi injeksi kueri asinkron dieksekusi menggunakan Apache JMeter (*MQTT Protocol Support Plugin*) yang diarahkan secara eksklusif pada *port* 1883 *broker* lokal Eclipse Mosquitto. Pengujian dieksekusi menggunakan pola *step-load* pada 50, 100, dan 200 *threads* selama durasi konstan 60 detik per skenario. Rekapitulasi metrik jaringan dan utilisasi sumber daya *server* disajikan pada Tabel 4.20.

Tabel 4. 20 Hasil Pengujian *Load testing* Server VPS

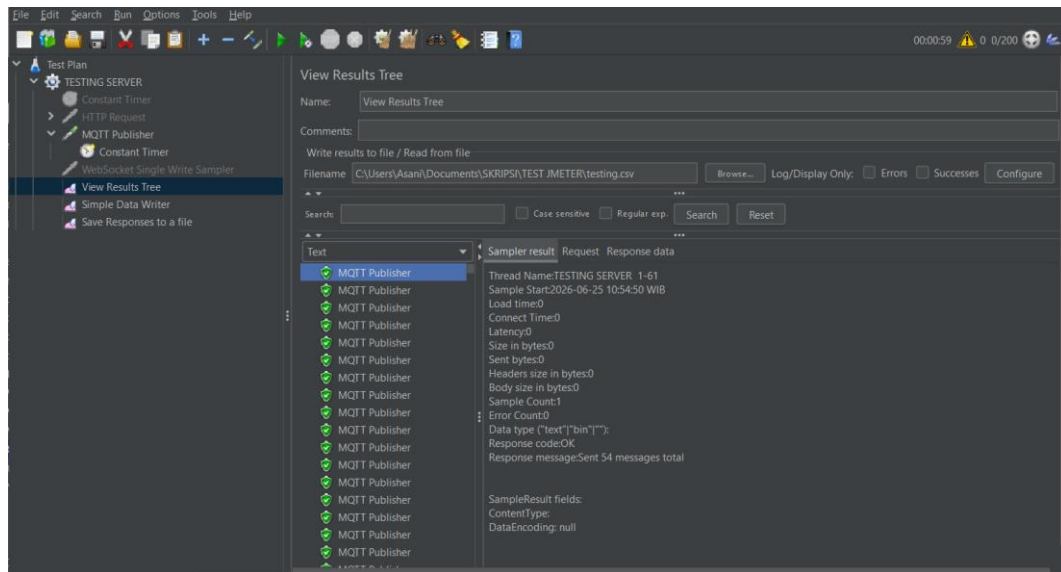
| Parameter Uji                 | Skenario 1<br>(50 koneksi) | Skenario 2<br>(100 koneksi) | Skenario 3<br>(200 koneksi) |
|-------------------------------|----------------------------|-----------------------------|-----------------------------|
| Total <i>Payload</i> Terkirim | 2.705<br><i>Requests</i>   | 5.391<br><i>Requests</i>    | 10.772<br><i>Requests</i>   |
| <i>Throughput</i> (Msg/Sec)   | 45,86<br>Req/Sec           | 91,61<br>Req/Sec            | 182,85<br>Req/Sec           |
| Average Latency               | 1,20 ms                    | 1,46 ms                     | 1,26 ms                     |
| Error Rate / <i>Timeout</i>   | 0%                         | 0%                          | 0%                          |
| Utilisasi CPU (Rata-rata)     | 10,06%                     | 28,11%                      | 41,23%                      |
| Konsumsi RAM<br>(Mosquitto)   | 8,32 MB                    | 8,40 MB                     | 8,56 MB                     |
| Konsumsi RAM<br>(Node.js)     | 134,11 MB                  | 141,54 MB                   | 174,16 MB                   |

Sebagai bentuk validasi empiris terhadap metrik yang disajikan pada Tabel 4.20, tangkapan layar hasil ekstraksi perangkat lunak pada skenario beban puncak (200 pengguna virtual) divisualisasikan pada Gambar 4.19. Adapun rincian log utilitas sumber daya *server* secara terperinci dilampirkan pada Lampiran 9.



### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta



Gambar 4. 19 Summary Report Apache JMeter (Skenario 200 Koneksi)

Berdasarkan metrik pada Tabel 4.20, kinerja *server* Node.js dan *broker* Mosquitto menunjukkan tingkat kestabilan yang sangat baik. Ketika jumlah beban koneksi dinaikkan dari 50, 100, hingga 200 pengguna virtual secara bersamaan, *server* merespons peningkatan tersebut dengan sangat lancar. Hal ini dibuktikan oleh nilai *throughput* transmisi data yang meningkat dua kali lipat secara konsisten—dari 45,86 pesan/detik hingga mencapai 182,85 pesan/detik—tanpa menyebabkan hambatan atau *bottleneck* pada jaringan. Meskipun beban kerja semakin berat, rata-rata penggunaan prosesor (CPU) *server* hanya menyentuh angka 41,23% pada skenario puncak. Kecepatan respons (*latency*) juga tetap sangat singkat di kisaran 1 hingga 1,5 milidetik, ditambah dengan tingkat kegagalan (*Error Rate*) mutlak 0% yang membuktikan tidak ada satupun koneksi atau data yang terputus selama pengujian berlangsung.

Dari sisi efisiensi memori, *broker* Mosquitto terbukti sangat ringan karena konsumsi RAM yang bertahan stabil di angka 8,56 MB meskipun harus menangani lalu lintas 200 koneksi aktif sekaligus. Di sisi lain, peningkatan penggunaan RAM pada *server* Node.js yang mencapai 174,16 MB pada beban tertinggi masih tergolong sangat wajar, mengingat sistem perlu mengalokasikan memori untuk membaca dan memproses ribuan format data JSON yang masuk





**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

dalam waktu singkat. Secara keseluruhan, hasil uji beban ini menegaskan bahwa arsitektur *server* yang dibangun sangat tangguh, terbebas dari risiko kehabisan memori (*Out-Of-Memory*), dan memiliki kapasitas yang lebih dari cukup untuk diterapkan secara langsung pada lahan pertanian hidroponik yang sebenarnya.

2. Hasil Pengujian Ketahanan Jangka Panjang (*Soak testing*)

Pengujian *Maturity* dieksekusi guna mendeteksi ada atau tidaknya anomali penumpukan sampah memori (*Memory leak*) melalui skenario perendaman sistem (*Soak Test*) selama 6 jam (360 menit). Menggunakan skrip pencatat otomatis `./logger.sh`, kernel memanen data setiap 5 detik sekali dari hasil pemancaran unit perangkat ESP32 di lapangan. Berdasarkan ekstraksi forensik pada berkas log kinerja dan Packet Capture (PCAP), pengujian memproduksi 14.369 paket jaringan, dengan 7.195 di antaranya merupakan transaksi protokol MQTT murni. Rekapitulasi metrik pengujian disajikan pada Tabel 4.21.

Tabel 4. 21 Rekapitulasi Metrik *Soak testing* Selama 6 Jam

| Parameter Pengujian                      | Metrik Terekam          | Status / Keterangan                    |
|------------------------------------------|-------------------------|----------------------------------------|
| Durasi Pengamatan                        | ~ 6 Jam                 | Berjalan kontinu tanpa henti           |
| Total Paket                              | ~ 4.320 Paket           | Asumsi 1 paket per 5 detik             |
| <i>Payload Publish</i><br>Diterima       | 4.320 Paket             | Berhasil 100% (0% <i>Packet loss</i> ) |
| Interupsi Sesi<br>( <i>Disconnects</i> ) | 0 Kejadian              | Terowongan sesi persisten stabil       |
| Anomali Transmisi<br>TCP                 | 6 <i>Retransmission</i> | 0,04% dari total paket                 |
| Utilisasi CPU (Rata-rata)                | 1,78%                   | Komputasi I/O efisien                  |
| Konsumsi RAM<br>Node.js                  | 107,66 MB – 110,68 MB   | Fluktuasi normal                       |



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|                           |                   |                                   |
|---------------------------|-------------------|-----------------------------------|
| Konsumsi RAM<br>Mosquitto | 8,55 MB (Konstan) | Statis tanpa deviasi<br>kebocoran |
|---------------------------|-------------------|-----------------------------------|

Berdasarkan metrik pada Tabel 4.21, kinerja *server* di bawah tekanan aliran telemetri asinkron jangka panjang menunjukkan stabilitas absolut yang dapat dibuktikan melalui dua parameter utama:

- Stabilitas Memori dan Utilisasi Sumber Daya, konsumsi Resident Set Size (RSS) pada Node.js terekam sangat stabil pada rentang 107,66 MB hingga 110,68 MB. Fluktuasi ( $\pm 3$  MB) ini membuktikan berjalannya mekanisme *Garbage collection* secara optimal dalam membuang objek JSON yang telah diproses, sekaligus memvalidasi ketiadaan anomali *Memory leak*. *Broker* Mosquitto juga menunjukkan efisiensi tinggi dengan konsumsi RAM statis di angka 8,55 MB. Beban komputasi CPU rata-rata yang hanya menyentuh 1,78% mengonfirmasi bahwa arsitektur *server* mampu menangani antrean I/O tanpa mengalami kelelahan komputasi.
- Integritas Jaringan dan Transmisi Data, Tautan socket TCP terbukti kokoh (0 kejadian disconnect) selama 6 jam penuh. Mekanisme *Keep-alive* berhasil mempertahankan sesi publik melalui rasio simetris 1.436 paket pertukaran *Ping request/Response*. Meskipun terekam 6 anomali TCP *Retransmission* (0,04%) yang diakibatkan oleh fluktuasi sinyal radio fisik, tumpukan TCP *server* berhasil melakukan pemulihan transmisi secara otonom tanpa memicu eksepsi *timeout*. Klien berhasil mendaratkan 4.320 paket, mencapai tingkat keandalan (*Reliability*) sebesar 100% terhadap ekspektasi target pengiriman.

### 3. Hasil Pengujian Daya Pulih Sistem (*Recoverability*)

Pengujian daya pulih (*recoverability*) dieksekusi untuk memvalidasi kapabilitas pemulihan sesi persisten (*re-binding*) serta mengukur jeda waktu normalisasi otonom (*self-healing latency*) pasca-disimulasikannya pemutusan paksa tautan lapisan fisik (*link drop*) di sisi klien selama persis 60 detik. Mengacu pada rumusan Persamaan (8), penghitungan *delay* murni diukur dari titik nol (*stopwatch lap-start*) saat antarmuka Wi-Fi klien dihidupkan kembali





### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

( $T_{\text{normalisasi}}$ ) hingga paket telemetri pertama berbendera "is\_recovery": true tervalidasi masuk ke dalam basis data Cloud Firestore ( $T_{\text{mendarat}}$ ). Rekapitulasi hasil komparasi ketiga protokol disajikan pada Tabel 4.22.

Tabel 4. 22 Hasil Pengujian Daya Pulih Sistem (*Recoverability*)

| Protokol Komunikasi | Jeda Waktu Total<br>( $T_{\text{mendarat}} - T_{\text{putus}}$ ) | Asumsi Blackout Manual | Total Recovery Latency | <i>Delay</i> Ingestasi <i>Cloud</i> | Karakteristik Penyimpangan Protokol                                                             |
|---------------------|------------------------------------------------------------------|------------------------|------------------------|-------------------------------------|-------------------------------------------------------------------------------------------------|
| HTTP                | 69,34 Detik                                                      | 60,00 Detik            | 9,34 Detik             | 1.223 ms                            | Terjadi TCP <i>RST</i> ( <i>Connection Refused</i> ) pada seqId: 11, sukses pulih di seqId: 12. |
| MQTT                | 71,39 Detik                                                      | 60,00 Detik            | 11,39 Detik            | 161 ms                              | Mengalami <i>Keep-alive Holding Overhead</i> , namun <i>payload</i> mendarat utuh (0% Loss).    |
| WebSocket           | 68,02 Detik                                                      | 60,00 Detik            | 8,02 Detik             | 79 ms                               | Menerapkan <i>Asynchronous Tight-polling</i> pada CPU Klien, <i>delay</i> pemulihan tercepat.   |



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Berdasarkan Tabel 4.22 di atas, unjuk kerja pemulihan jalur transmisi pasca-interupsi lapisan fisik dapat diuraikan secara objektif sebagai berikut:

#### 1. Analisis Daya Pulih Jalur HTTP

Pada pengujian jalur komunikasi HTTP diperoleh waktu pemulihan murni sebesar 9,34 detik. Audit forensik pada basis data mengungkap satu fenomena arsitektur yang vital terjadinya penolakan sesi pada injeksi pertama (*TCP Reset / Connection Refused*). Hal ini dibuktikan oleh melompatnya urutan paket yang terekam di Cloud Firestore dari seqId: 10 langsung menuju seqId: 12. Pasca-terputus 60 detik, server belum sempat memutus sambungan lama sepenuhnya, sehingga ketika ESP32 mencoba mengirim data baru, server sempat menolak koneksi pertama tersebut (*TCP RST*). Namun, sistem secara otomatis mencoba ulang dan berhasil pada percobaan berikutnya. Ketika ESP32 mendadak mengirimkan kueri baru, *server* merespons dengan penolakan. Namun, kapabilitas *self-healing* pada *firmware* klien terbukti layak; sistem secara otonom memutar ulang kueri (*retry*) dan berhasil memvalidasi paket seqId: 12 dalam jeda komputasi 1,22 detik.

#### 2. Analisis Daya Pulih Jalur MQTT

Protokol MQTT mencetak *delay* pemulihan paling lambat di antara ketiga jalur, yakni 11,39 detik. Kelambatan ini diakibatkan oleh mekanisme *Keep-alive Holding Overhead*; di mana *broker* lokal Mosquitto di *server* masih mempertahankan sesi ID Klien lama di dalam RAM selama jeda pemadaman. Ketika klien mendapatkan alokasi IP baru dari *router* dan mengirimkan kueri *CONNECT*, *server* membutuhkan siklus komputasi ekstra untuk melakukan *session takeover*. Meski lambat, keunggulan absolut MQTT terletak pada integritas *payload* (0% *Packet loss*); paket pemulihan pertama (seqId: 11) berhasil ditelan utuh tanpa mengalami penolakan di gerbang *TCP server*.





#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

### 3. Analisis Daya Pulih Jalur WebSocket

WebSocket keluar sebagai protokol dengan kapabilitas daya pulih paling unggul. Begitu antarmuka fisik mikrokontroler berhasil mengikat ulang sinyal Wi-Fi pada detik ke-68,02, tumpukan *frame* data pemulihan (seqId: 11) langsung didorong masuk ke dalam *server* secara instan. Kinerja kilat ini dicapai karena klien menerapkan metode *Asynchronous Tight-polling*; di mana inti prosesor mikrokontroler terus memutar kueri koneksi tanpa terhenti oleh fungsi *blocking CPU*. Keunggulan arsitektur *Full-duplex Persistent Tunnel* terbukti secara empiris: *delay* ingestasi *cloud* terekam hanya memakan waktu 79 milidetik, menjadikan WebSocket sebagai jalur *monitoring* waktu-nyata paling responsif di lahan pertanian terbuka.

### 5. Hasil User Acceptance Test

Pengujian penerimaan pengguna (*User Acceptance Test*) dilakukan untuk memvalidasi bahwa sistem yang dibangun telah memenuhi kebutuhan dan ekspektasi pengguna akhir. Pengujian ini dilaksanakan dengan melibatkan 1 (satu) responden yang mewakili profil pengguna sistem, yaitu pelaku budidaya hidroponik. Responden diberikan akses langsung ke sistem dan diminta menjalankan skenario penggunaan nyata, kemudian mengisi formulir evaluasi yang mencakup aspek kemudahan penggunaan, keterbacaan informasi sensor, dan kecukupan fitur pemantauan yang tersedia. Pengujian dibagi ke dalam dua instrumen evaluasi, yaitu pengujian fungsional berbasis skenario yang mengukur keberhasilan eksekusi setiap fitur utama sistem, serta penilaian kepuasan menggunakan skala Likert (1–5) yang mengukur persepsi subjektif pengguna terhadap kualitas antarmuka. Data identitas responden disajikan pada Tabel 4.23.

Tabel 4. 23 Hasil Pengujian Fungsional Sistem (UAT)

| No | Skenario Pengujian                                                                             | Hasil    | Keterangan                                                                            |
|----|------------------------------------------------------------------------------------------------|----------|---------------------------------------------------------------------------------------|
| 1  | Autentikasi Sistem<br>Responden melakukan proses login menggunakan akun yang telah didaftarkan | Berhasil | Sistem berhasil memverifikasi identitas pengguna dan memberikan akses masuk ke dasbor |



## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|   |                                                                                                                                       |          |                                                                                                         |
|---|---------------------------------------------------------------------------------------------------------------------------------------|----------|---------------------------------------------------------------------------------------------------------|
| 2 | Pemantauan Data Waktu-Nyata ( <i>Real-Time</i> )<br>Responden memantau nilai pH dan PPM yang ditampilkan pada dasbor utama            | Berhasil | Nilai sensor berhasil ditampilkan dan diperbarui secara otomatis tanpa perlu memuat ulang halaman       |
| 3 | Verifikasi ( <i>Rule-Based Helper</i> )<br>Responden memverifikasi keluaran status kualitas air ( <i>Optimal / Waspada / Kritis</i> ) | Berhasil | Indikator status dan rekomendasi tindakan berhasil ditampilkan sesuai kondisi nilai sensor yang terbaca |
| 4 | Pengecekan Riwayat Log Data<br>Responden mengakses halaman riwayat data historis sensor                                               | Berhasil | Data historis berhasil dimuat dan ditampilkan dalam bentuk tabel riwayat ( <i>history log</i> )         |
| 5 | Pengujian Fitur Ekspor Data<br>Responden mengunduh data riwayat sensor dalam format Excel (.xlsx)                                     | Berhasil | Berkas Excel berhasil diunduh dan memuat data yang sesuai dengan rekaman basis data                     |
| 6 | Penyambungan Notifikasi Telegram Responden<br>menghubungkan akun Telegram ke sistem                                                   | Berhasil | Proses pengikatan akun Telegram berhasil dilakukan dan notifikasi percobaan berhasil diterima           |
| 7 | Pemutusan Koneksi Telegram ( <i>Unbinding</i> )<br>Responden memutuskan sambungan akun Telegram dari sistem                           | Berhasil | Koneksi Telegram berhasil diputus dan sistem berhenti mengirimkan notifikasi                            |
| 8 | Pengujian Responsivitas Tata Letak                                                                                                    | Berhasil | Tata letak antarmuka menyesuaikan ukuran layar secara adaptif dan                                       |





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|                                                                       |  |                                               |
|-----------------------------------------------------------------------|--|-----------------------------------------------|
| Responden menguji tampilan antarmuka pada perangkat smartphone/tablet |  | seluruh elemen tetap dapat dibaca dengan baik |
|-----------------------------------------------------------------------|--|-----------------------------------------------|

Seluruh 8 (delapan) skenario fungsional pada Tabel 4.23 yang diujikan berhasil dieksekusi tanpa kegagalan. Hasil ini membuktikan bahwa integrasi sistem secara *end-to-end* mulai dari autentikasi, pemantauan sensor, eksekusi algoritma *Rule-Based*, pengelolaan riwayat data, hingga fitur notifikasi Telegram telah berjalan sesuai dengan spesifikasi yang telah ditetapkan dan dapat dioperasikan secara mandiri oleh pengguna non-teknis di lapangan.

Instrumen kedua berupa penilaian kepuasan menggunakan skala *Likert* 1–5, di mana responden mengevaluasi 5 (lima) aspek kualitas antarmuka sistem. Keterangan skala: 1 = Sangat Tidak Setuju, 2 = Tidak Setuju, 3 = Netral, 4 = Setuju, 5 = Sangat Setuju. Rekapitulasi hasil penilaian disajikan pada Tabel 4.24.

Tabel 4. 24 Penilaian Kepuasan Pengguna

| No | Pernyataan                                                                                                                      | Skor | Interpretasi |
|----|---------------------------------------------------------------------------------------------------------------------------------|------|--------------|
| 1  | Tata letak menu dan fungsi tombol pada aplikasi ini sangat intuitif dan mudah dipahami pada penggunaan pertama                  | 4    | Setuju       |
| 2  | Visualisasi grafik sensor dan kontras warna pada antarmuka dasbor sangat nyaman dibaca                                          | 4    | Setuju       |
| 3  | Rekomendasi tindakan yang diberikan oleh fitur Decision Helper sangat jelas dan mudah dieksekusi oleh pengguna awam             | 4    | Setuju       |
| 4  | Transisi perpindahan antar-menu serta proses pemuatan tabel terasa cepat dan tidak lambat (lag)                                 | 4    | Setuju       |
| 5  | Keberadaan fitur ekspor Excel dan push-notifikasi Telegram membuat aplikasi ini sangat layak digunakan di lahan perkebunan riil | 4    | Setuju       |



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

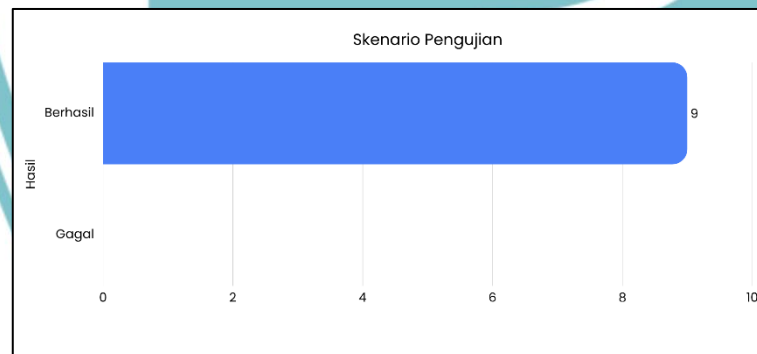
Rata-rata skor kepuasan sebesar 4,00 dari 5,00 (80%) mengindikasikan bahwa seluruh aspek antarmuka yang dievaluasi dinilai berada pada kategori "Setuju" oleh responden. Konsistensi skor 4 pada seluruh pernyataan menunjukkan bahwa pengguna menilai sistem secara seimbang — tidak ada aspek yang dianggap jauh lebih unggul maupun lebih lemah dari yang lain. Nilai kepuasan sebesar 80% ini mengonfirmasi bahwa antarmuka Web yang dibangun berbasis React.js telah memenuhi standar kebergunaan (*usability*) yang baik untuk segmen pengguna berlatar belakang non-teknis di sektor pertanian.

#### 4. 4. 4 Analisis Data

##### 1. Analisis Pengujian Fungsionalitas

Berdasarkan hasil pengujian fungsionalitas menggunakan metode Black Box yang dirangkum pada Tabel 4.3, seluruh 9 skenario uji berhasil dieksekusi tanpa kegagalan. Mengacu pada persamaan (1), tingkat kelayakan fungsional sistem dihitung sebagai berikut:

$$\text{Persentase Keberhasilan} = \frac{9}{9} \times 100\% = 100\%$$



Gambar 4. 20 Grafik Hasil Pengujian Fungsionalitas

Perhitungan tersebut memvalidasi bahwa seluruh arsitektur sistem telah terintegrasi secara *end-to-end* dengan tingkat keberhasilan absolut (100%). Analisis mendalam dari pengujian fungsionalitas ini dapat dijabarkan ke dalam tiga fase utama aliran data:

##### 1. Fase Pengumpulan Data dan Transmisi Data (Perangkat Keras)

Pada sisi node sensor, mikrokontroler ESP32 terbukti mampu melakukan pembacaan nilai tegangan analog dari sensor pH dan TDS secara akurat. Data





### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

tersebut kemudian dikonversi menjadi format *payload* JSON dan dipublikasikan (*publish*) menuju *broker* MQTT secara periodik. Tidak ditemukannya *packet loss* atau *timeout* pada tahap ini menunjukkan bahwa protokol komunikasi yang diterapkan pada mikrokontroler berjalan secara stabil.

### 2. Fase Pemrosesan dan Penyimpanan (*Server Backend*)

*Server backend* (Node.js) menunjukkan keandalan tinggi dalam menjalankan fungsi *subscribe* terhadap topik MQTT untuk menerima *payload* data dari perangkat keras. Proses parsing data berjalan sempurna dan langsung dilanjutkan dengan eksekusi algoritma *Rule-Based*. Pengujian membuktikan bahwa *server* mampu mengklasifikasikan status air (Optimal, Waspada, atau Kritis) berdasarkan aturan parameter yang telah ditetapkan, dan menyimpannya secara persisten ke dalam basis data Cloud Firestore beserta *timestamp* historis yang presisi.

### 3. Fase Visualisasi dan Interaksi (*Dashboard Frontend*)

Pada sisi antarmuka pengguna (*Client*), pemanfaatan protokol WebSocket terbukti sangat efektif dalam mengatasi kendala *delay* penyegaran halaman. *Dashboard* berhasil menerima siaran data (*broadcast*) dari *server* dan langsung memperbarui indikator angka pH, PPM, serta status *Rule-Based* secara dinamis dan *real-time*. Selain itu, fungsionalitas pelaporan historis melalui fitur *Export to Excel* juga tervalidasi berjalan dengan baik, memungkinkan pengguna untuk mengunduh rekapitulasi data format .xlsx secara utuh sebagai bahan evaluasi lebih lanjut.

## 2. Analisis Hasil Pengujian Algoritma *Rule-Based*

Sembilan skenario uji yang dirancang mencakup seluruh kombinasi kemungkinan antara nilai pH (Asam/Normal/Basa) dan nilai nutrisi (Rendah/Normal/Tinggi), memastikan tidak ada kondisi yang terlewat dalam pengujian. Keselarasan 1:1 antara probabilitas masukan dengan aturan ini mengonfirmasi bahwa arsitektur mesin inferensi (*inference engine*) pada *server* Node.js tidak memiliki aturan yang tumpang tindih (*redundant*), serta menutup rapat celah terjadinya *unhandled condition* (kondisi mengambang tanpa kesimpulan).



## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Guna memvalidasi integritas eksekusi dari hulu ke hilir (*end-to-end tracing*), berikut adalah penjabaran alur resolusi logika yang diekstraksi dari Skenario Uji ke-3, dengan kueri masukan nilai pH = 3.5 dan kepekatan nutrisi TDS = 1750 PPM:

#### a. Tahap Pemetaan Masukan Tegas (*Crisp Input Mapping*)

Pada tahap ini, nilai masukan sensor numerik dievaluasi dan dipetakan secara langsung ke dalam kategori kondisi tegas (*crisp*) berdasarkan ambang batas (*threshold*) yang telah ditetapkan pada sistem.

- Kondisi pH, Nilai masukan sebesar 3.5 dievaluasi terhadap parameter batas bawah. Karena nilai 3.5 lebih kecil dari 6.0 ( $< 6.0$ ), sistem menentukannya secara mutlak ke dalam kategori ASAM.
- Kondisi Nutrisi (PPM), Nilai masukan sebesar 1750 PPM dievaluasi terhadap parameter batas atas. Karena nilai 1750 lebih besar dari 1500 ( $> 1500$ ), sistem menentukannya secara mutlak ke dalam kategori TINGGI.

#### b. Tahap Eksekusi Aturan (Rule Evaluation)

Sistem mengevaluasi 9 basis aturan (*Rule-Based*) IF-THEN pada *backend*. Karena algoritma menggunakan operator logika kondisional AND, kondisi dieksekusi hanya jika kedua anteseden (pH dan PPM) terpenuhi secara bersamaan.

- Sistem membuang 8 aturan lain yang tidak relevan
- Sistem mencocokkan kondisi pada satu aturan aktif, IF (pH = ASAM) AND (PPM = TINGGI)
- Berdasarkan basis aturan, konsekuensi dari kondisi tersebut adalah, THEN Status = KRITIS.

Hasil penelusuran inferensi komputasi di atas menghasilkan string keluaran absolut "KRITIS". Nilai ini selaras secara sempurna dengan hasil perhitungan komparasi manual ideal. Keberhasilan *server* menyajikan kesimpulan tanpa mengalami deviasi logika membuktikan bahwa algoritma *Rule-Based* yang dikembangkan memiliki tingkat kompleksitas waktu eksekusi linear, sehingga mampu memberikan panduan keputusan (*Decision helper*) secara instan kepada petani tanpa membebani utilisasi memori *server*.





## 2. Analisis Hasil Pengujian Quality of Services (QoS)

Untuk memberikan pandangan analitis yang utuh terhadap kinerja arsitektur ingestasi data pada *server*, seluruh parameter *Quality of Service* yang telah diekstraksi dari ketiga jalur komunikasi disintesis ke dalam matriks perbandingan pada Tabel 4.23 berikut.

Tabel 4. 25 Analisis QoS

| Parameter QoS      | Protokol MQTT | Protokol HTTP | Protokol WebSocket |
|--------------------|---------------|---------------|--------------------|
| <i>Throughput</i>  | 0,53 kbps     | 10.11 kbps    | 1,66 kbps          |
| <i>Packet loss</i> | 0,00%         | 0,00%         | 0,00%              |
| <i>Delay</i>       | 178,36 ms     | 572,75 ms     | 188,90 ms          |
| <i>Jitter</i>      | 33,92 ms      | 43,35 ms      | 35,50 ms           |

Berdasarkan hasil komparasi empiris pada Tabel 4.17, terbukti secara ilmiah bahwa rancang bangun arsitektur protokol komunikasi memengaruhi kapabilitas sistem *Internet of things* (IoT) secara fundamental:

### 1. Efisiensi *Payload* Jaringan

Berdasarkan hasil pengujian, konsumsi *throughput* yang dihasilkan berada pada rentang yang sangat rendah: MQTT (0,53 kbps), WebSocket (1,66 kbps), dan HTTP (10,11 kbps). Merujuk pada standar TIPHON, ketiga nilai tersebut berada di bawah ambang batas 338 Kbps sehingga secara formal masuk ke dalam Indeks 1 (Buruk). Namun, standar TIPHON dirancang untuk mengevaluasi layanan telekomunikasi multimedia, bukan IoT. Rendahnya *throughput* di sini justru menunjukkan efisiensi, di mana MQTT terbukti ~19 kali lipat lebih ringan dibandingkan HTTP berkat penggunaan header biner 2 bita yang mengeliminasi beban teks plaintext pada setiap kueri.

### 2. Keandalan dan Stabilitas Transmisi (*Packet Loss* dan *Jitter*)

Ketiga protokol berhasil mencetak rekor *Packet loss* sempurna sebesar 0,00%. Prestasi ini mengonfirmasi bahwa infrastruktur lapisan transport TCP pada *server* Google Cloud Platform beroperasi secara prima dalam menangani



### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

transmisi ulang (TCP *Retransmission*) saat terjadi pentalan paket. Dari aspek variasi *delay*, MQTT memimpin dengan *jitter* terendah (33,92 ms), disusul ketat oleh WebSocket (35,50 ms), dan HTTP di posisi terakhir (43,35 ms). Fluktuasi pada HTTP dipicu oleh keharusan sistem operasi membangun dan membongkar ulang soket TCP secara asinkron setiap kali kueri baru datang.

### 3. Analisis *Delay*

Berdasarkan data pengujian, metrik *delay* menunjukkan pemisah kelas kinerja yang sangat jelas. HTTP mencetak *delay* tertinggi sebesar 572,75 ms yang merujuk pada standar TIPHON (>450 ms) diklasifikasikan ke dalam Indeks 1 (Buruk) akibat keharusan membangun ulang soket TCP pada setiap kueri. Sebaliknya, MQTT (178,36 ms) dan WebSocket (188,90 ms) berhasil menembus standar emas ITU-T G.114 (<400 ms) dan menduduki Indeks 3 (Bagus) berkat penerapan soket persisten yang mengeliminasi waktu jabat tangan (*handshake*) berulang.

Meskipun WebSocket mengirim data secara langsung ke *server* tanpa melalui perantara (*broker*), keterlambatannya (*delay*) tercatat sedikit lebih tinggi dibandingkan MQTT. Penyebabnya adalah standar bawaan WebSocket (RFC 6455) yang mengharuskan setiap data disandikan secara ringan (*masking*) sebelum dikirimkan. Proses penyandian tambahan ini membebani prosesor mikrokontroler ESP32 yang memiliki spesifikasi terbatas. Oleh karena itu, protokol MQTT yang beroperasi tanpa proses penyandian tersebut menjadi lebih ringan dan responsif saat digunakan pada perangkat keras bersumber daya kecil.

Evaluasi *delay* ini secara langsung menjawab permasalahan utama pada Bab 1 melalui pendekatan komparasi arsitektural. Arsitektur layanan *cloud* publik pihak ketiga seperti Blynk terbukti memiliki batasan keterlambatan bawaan, dengan rata-rata *delay* mencapai 3.400 ms (Syukhron dkk., 2021). Melalui pergantian ke arsitektur *server* VPS mandiri, kinerja protokol yang paling lambat sekalipun (HTTP: 572,75 ms) terbukti mampu memangkas latensi hingga 5,94 kali lebih cepat dibandingkan infrastruktur *cloud* publik tersebut. Lebih jauh, protokol yang paling





#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

optimal (MQTT: 178,36 ms) hingga 19,06 kali lipat. Komparasi arsitektural ini memvalidasi bahwa pergantian infrastruktur menuju *server* mandiri merupakan solusi yang menyelesaikan masalah kemacetan transmisi data pada sistem hidroponik.

### 3. Analisis Hasil Pengujian Reliabilitas Sistem

Pengujian reliabilitas dieksekusi melalui tiga dimensi berdasarkan kerangka kerja ISO/IEC 25010:2023, yaitu toleransi beban (*fault tolerance*), ketahanan jangka panjang (*maturity*), dan daya pulih sistem (*recoverability*). Analisis dari masing-masing dimensi dijabarkan sebagai berikut.

#### b. Analisis Toleransi Beban (*Load testing*)

Berdasarkan data hasil pengujian pada Tabel 4.20, terdapat pola yang menarik pada perilaku *server* ketika menghadapi beban bertingkat (*step-load*) yang disimulasikan oleh Apache JMeter terhadap *broker* Eclipse Mosquitto.

- Pada skenario pertama (50 koneksi serentak), *server* beroperasi pada titik *baseline* dengan sangat efisien. Sebanyak 2.705 *request* berhasil dieksekusi dengan *throughput* mencapai 45,86 Req/Sec. Utilisasi CPU total hanya menyentuh 10,06%. Kondisi ini mencerminkan bahwa beban ringan sama sekali tidak memberikan tekanan berarti pada antrean *event loop* Node.js maupun *socket I/O broker* Mosquitto.
- Pada skenario kedua (100 koneksi serentak), beban kerja digandakan, dan arsitektur merespons secara matematis proporsional. *Throughput* meningkat dua kali lipat menjadi 91,61 Req/Sec dengan total 5.391 *request*. Eskalasi CPU ke tingkat 28,11% membuktikan bahwa *server* secara aktif mengalokasikan sumber daya komputasi untuk menangani kueri yang masuk secara simultan tanpa memicu antrean (*bottleneck*) yang merusak kinerja *delay*.
- Pada skenario ketiga (200 koneksi serentak), skalabilitas sistem tervalidasi secara absolut. Kemampuan *server* dalam menangani banyak permintaan sekaligus berkat arsitektur asinkron Node.js dan efisiensi *broker* Mosquitto memungkinkan sistem memproses hingga 182,85 permintaan per detik pada beban 200 pengguna serentak, tanpa terjadi kegagalan atau perlambatan



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

berarti. Utilisasi CPU mencapai 41,23%, menandakan bahwa *server* masih memiliki sisa ruang kapasitas prosesor (*headroom*) yang sangat leluasa sebelum menyentuh titik infleksi degradasi komputasi.

Mengacu pada Persamaan (6), tingkat kegagalan kueri (*Error Rate*) pada skenario puncak tervalidasi secara kuantitatif:

$$\text{Error Rate} = \left( \frac{0 \text{ Paket Gagal}}{10.772 \text{ Paket Terkirim}} \right) \times 100\% = 0\%$$

Ketiadaan pemutusan koneksi sepihak (*0% Error Rate*) pada seluruh fase pengujian membuktikan bahwa *broker* Eclipse Mosquitto terbebas dari penolakan soket TCP. Dari perspektif efisiensi sumber daya, konsumsi memori *Resident Set Size* (RSS) Mosquitto terekam sangat statis dan hemat, hanya bergeser tipis di angka 8,56 MB pada beban puncak. Hal ini menegaskan bahwa *broker* berbasis bahasa C tersebut tidak mengalami kebocoran memori (*memory leak*) dan sangat kebal terhadap ancaman *Out-Of-Memory* (OOM).

Fluktuasi memori hanya terjadi pada layanan Node.js yang memuncak di angka 174,16 MB. Perilaku ini merupakan karakteristik normal dari *V8 JavaScript Engine* yang memerlukan alokasi tumpukan (*heap allocation*) sementara untuk memproses ribuan *payload* JSON sebelum siklus *Garbage collection* diaktifkan secara otomatis.

#### b. Analisis Ketahanan Jangka Panjang (*Soak testing*)

Berdasarkan rekapitulasi metrik pada Tabel 4.21, pengujian *soak testing* selama 6 jam penuh berhasil memvalidasi ketiadaan anomali kebocoran memori (*Memory leak*) pada infrastruktur *server* melalui dua indikator utama.

- Dari aspek stabilitas memori, konsumsi Resident Set Size (RSS) pada Node.js terekam pada rentang 107,66 MB hingga 110,68 MB. Penggunaan memori *server* hanya berfluktuasi sekitar  $\pm 3$  MB selama 6 jam — angka yang sangat kecil membuktikan tidak ada kebocoran memori (*memory leak*) yang terjadi. *Server* membuang data yang sudah selesai diproses secara otomatis dan teratur. Pola alokasi memori yang stabil (bukan eskalatif





### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

linear) secara definitif mengonfirmasi ketiadaan anomali *Memory leak*. Di sisi *broker* Mosquitto, konsumsi RAM konstan pada angka 8,55 MB tanpa deviasi selama 6 jam, memperkuat bukti efisiensi arsitektur *broker* pesan tersebut.

- Dari aspek integritas jaringan, tautan socket TCP berhasil dipertahankan secara kokoh dengan 0 kejadian disconnect selama durasi pengamatan penuh. Meskipun terekam 6 anomali TCP *Retransmission* (0,04% dari total paket), tumpukan TCP *server* berhasil melakukan pemulihan transmisi secara otonom tanpa memicu eksepsi *timeout*.

Mengacu pada Persamaan (7), tingkat keandalan (Reliability) dihitung sebagai berikut:

$$\text{Reliability} = \frac{4.320 \text{ Paket}}{4.320 \text{ Paket}} \times 100\%$$

Tingkat keandalan 100% ini melampaui ambang batas > 95% yang ditetapkan sebagai kriteria "Sangat Bagus" berdasarkan standarisasi TIPHON, sehingga *server* dinyatakan andal (reliable) untuk beroperasi dalam durasi jangka panjang.

### c. Analisis Daya Pulih Sistem (*Recoverability*)

Berdasarkan data komparatif pada Tabel 4.22, analisis kapabilitas pemulihan masing-masing protokol pasca-interupsi lapisan fisik selama 60 detik dijabarkan sebagai berikut.

- WebSocket mencatat *delay* pemulihan tercepat sebesar 8,02 detik dengan *delay* ingestasi *cloud* hanya 79 ms. Kecepatan ini dicapai karena *firmware* klien menerapkan mekanisme *asynchronous tight-polling*, di mana prosesor mikrokontroler terus-menerus mencoba membangun ulang koneksi tanpa fungsi *blocking* CPU. Setelah antarmuka Wi-Fi berhasil mengikat ulang sinyal, bingkai data pemulihan langsung didorong ke *server* melalui terowongan *full-duplex* persisten secara instan.



#### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

- HTTP mencatat *delay* pemulihan sebesar 9,34 detik dengan *delay* ingestasi awan 1.223 ms. Audit data pada Cloud Firestore mengungkap fenomena arsitektural berupa hilangnya satu paket (seqId: 11) yang menandakan terjadinya penolakan koneksi (TCP *RST* / Connection Refused) pada injeksi pertama pasca-pemulihan. Hal ini kemungkinan disebabkan oleh belum selesainya dealokasi soket lama di sisi *server* saat klien secara mendadak mengirimkan kueri baru. Namun, kapabilitas *self-healing* pada *firmware* klien terbukti layak — sistem secara otonom melakukan *retry* dan berhasil memvalidasi paket seqId: 12 pada percobaan berikutnya.
- MQTT mencatat *delay* pemulihan terlama sebesar 11,39 detik dengan *delay* ingestasi awan 161 ms. Kelambatan relatif ini diakibatkan oleh mekanisme *Keep-alive Holding Overhead*; *broker* Mosquitto di *server* masih mempertahankan sesi ID Klien lama di dalam RAM selama jeda pemadaman. Ketika klien memperoleh alokasi IP baru dan mengirimkan kueri CONNECT, *broker* memerlukan siklus komputasi ekstra untuk melakukan *session takeover* (menggantikan sesi lama). Meskipun paling lambat dalam hal waktu pemulihan, MQTT memiliki keunggulan pada integritas *payload* — paket pemulihan pertama (seqId: 11) berhasil diterima utuh tanpa mengalami penolakan TCP seperti yang terjadi pada HTTP (0% *Packet loss* saat recovery).

Secara keseluruhan, ketiga protokol berhasil memulihkan jalur transmisi di bawah ambang batas toleransi 30 detik (Pratama dkk., 2023), sehingga ketiganya dinyatakan memiliki kapabilitas *self-healing* yang layak untuk implementasi di lingkungan pertanian terbuka yang rentan terhadap gangguan sinyal Wi-Fi. WebSocket unggul dalam kecepatan pemulihan, MQTT unggul dalam integritas data saat pemulihan, sedangkan HTTP memiliki kerentanan berupa penolakan koneksi pada injeksi pertama yang perlu dimitigasi melalui mekanisme *retry* pada *firmware* klien.

Perlu dicatat bahwa pengujian daya pulih pada penelitian ini dieksekusi melalui skenario pengamatan tunggal (*single observation*) pada masing-masing





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

protokol. Meskipun hasil ini telah memvalidasi kapabilitas pemulihan otonom sistem dibawah batas toleransi 30 detik, pengujian lanjutan dengan repetisi statistik disarankan di masa mendatang guna memetakan margin variansi (*jitter*) dari *delay self-healing* tersebut



**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

## BAB V

### KESIMPULAN DAN SARAN

#### 5.1. Kesimpulan

Berdasarkan perancangan, implementasi, dan pengujian yang telah dilakukan pada Sistem Monitoring Hidroponik menggunakan arsitektur *Client-Server*, dapat ditarik kesimpulan sebagai berikut:

1. Rancang bangun mekanisme komunikasi data berhasil diimplementasikan secara terintegrasi pada infrastruktur *Virtual Private Server* (VPS) Node.js guna memfasilitasi kebutuhan evaluasi komparatif. Untuk menjamin validitas pengujian, peladen direkayasa dengan arsitektur *Gatekeeper* (*Protocol Switch*) yang mengatur jalur masuk data (*ingestion*) dari mikrokontroler ESP32. Mekanisme ini secara dinamis mengisolasi lalu lintas jaringan, sehingga transmisi perangkat menggunakan protokol HTTP, MQTT, dan WebSocket dapat dieksekusi secara bergantian tanpa memicu interferensi data. Adapun pada lapisan presentasi, peladen memanfaatkan terowongan persisten WebSocket secara khusus untuk mendistribusikan data *real-time* ke antarmuka *dashboard*.
2. Karakteristik kinerja transmisi data menunjukkan disparitas efisiensi yang signifikan antarprotokol. Protokol HTTP tercatat sebagai jalur komunikasi paling lambat dan membebani jaringan, ditandai dengan *delay* tertinggi (572,75 ms), *jitter* tertinggi (43,35 ms), dan pembengkakan konsumsi *bandwidth* (*throughput* 10,11 kbps) akibat tingginya *overhead header*. Sebaliknya, protokol MQTT mendominasi efisiensi dengan *delay* tercepat (*delay* 178,36 ms), stabilitas pengiriman terbaik (*jitter* 33,92 ms), dan ukuran paket paling ringan (*throughput* 0,53 kbps). Protokol WebSocket membuktikan kinerja yang kompetitif (*delay* 188,90 ms) berkat peniadaan inisiasi sesi berulang. Ketiga protokol terbukti andal dalam menjaga keutuhan transmisi dengan tingkat *packet loss* absolut 0,00%.
3. Berdasarkan indeks standarisasi telekomunikasi TIPHON, protokol MQTT tervalidasi sebagai protokol yang paling optimal dan reliabel. Kombinasi nilai *delay* (178,36 ms) dan *packet loss* (0%) yang secara mutlak masuk ke dalam





**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

kategori "Sangat Bagus" menurut standar TIPHON membuktikan ketahanan protokol ini. Karakteristik ukuran *header* yang minimalis menjadikan MQTT sebagai arsitektur transmisi data yang paling direkomendasikan untuk ekosistem pertanian berskala industri.

## 5.2. Saran

Untuk pengembangan dan penyempurnaan sistem *Smart Farming* pada penelitian di masa mendatang, disarankan beberapa langkah strategis berikut:

1. Mengimplementasikan sertifikat SSL/TLS untuk melakukan eskalasi keamanan protokol dari HTTP menjadi HTTPS, WS menjadi WSS, dan MQTT menjadi MQTTS.
2. Mengingat basis data *Document-oriented* (Cloud Firestore) memiliki batas kuota baca/tulis (*read/write limit*) dari penyedia layanan *cloud*, disarankan untuk mengintegrasikan sistem memori *cache* perantara seperti Redis pada *server* Node.js untuk melayani permintaan data riwayat jangka pendek ke antarmuka pengguna tanpa membebani lalu lintas pembacaan basis data utama.

**POLITEKNIK  
NEGERI  
JAKARTA**



## DAFTAR PUSTAKA

Ardiansyah, A. dan Pamuji, F. Y. 2025. "Penerapan Standar TIPPHON untuk Kinerja Quality of Service Jaringan Internet (Studi Kasus di Jaringan Alfabet)". *JATI (Jurnal Mahasiswa Teknik Informatika)*, 9(5).

Amruddin, A., A., Hidayat, R., dan Suyatno, S. (2022). *Metodologi Penelitian Kuantitatif*. Bandung: CV. Media Sains Indonesia.

Andrianto, A., dan Suyatno, A. (2024). Analisis Perbandingan Beban Kerja Server Nginx dan Apache Menggunakan Apache JMeter. *Jurnal Informatika dan Komputer (JIK)*, 12(1), 45-55.

Ayeni, P. O. dan Adesoba, O. C. 2025. "IoT-based home control system using NodeMCU and Firebase". *Journal of Edge Computing*, 4, 17-34.

Bertrand-Martinez, E., Feio, P. D., Nascimento, V. d. B., Kon, F. dan Abelém, A. 2021. "Classification and evaluation of IoT brokers: A methodology". *International Journal of Network Management*, 31(3), e2115.

Dako, R. D. R. dan Ridwan, W. 2021. "Pengujian karakteristik Functional Suitability dan Performance Efficiency tesadaptif.net". *Jambura Journal of Electrical and Electronics Engineering*, 3(2), 48-53.

Eclipse Foundation. 2025. *Mosquitto - Open Source MQTT Broker*. <https://mosquitto.org/>. [20 Juni 2025]

Ferdiansyah, V., Oktavia, S. A., Mulyanto, Y. dan Yunanri, W. 2025. "Rancang Bangun Prototype Sistem Monitoring Dan Kontrol Tanaman Hidroponik Berbasis Internet of things (IoT) Menggunakan Microcontroller ESP32". *Bulletin of Information Technology (BIT)*, 6(3), 238-246.

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Fernández Candel, C. J., Sevilla Ruiz, D. dan García-Molina, J. J. 2021. "A Unified Metamodel for NoSQL and Relational Databases". *arXiv preprint arXiv:2105.06494*.

Halawa, D. N. (2024). Peran Teknologi Pertanian Cerdas (Smart Farming) untuk Generasi Pertanian Indonesia. *Jurnal Kridatama Sains dan Teknologi*, 6(2), 502-512.

International Telecommunication Union. 2003. *One-way transmission time (ITU-T Recommendation G.114)*. Geneva, Switzerland: ITU.

Irawan, N. D., Nurdin, S., Kusumawardhani, A. dan Izza, S. 2023. "Smart Hidroponik Berbasis Internet of things (IoT) Untuk Efektifitas Pertumbuhan Tanaman Bayam". *Jurnal Terapan Sains & Teknologi*, 5(2), 145-152.

ISO/IEC. 2023. *Information technology — Software product Quality Requirements and Evaluation (SQuaRE) — Product quality model*. ISO/IEC 25010:2023. Geneva, Switzerland: International Organization for Standardization.

Juanda, R. dan Yadi, I. Z. 2020. "Penerapan *Rule-Based* Dengan Algoritma Viterbi Untuk Deteksi Kesalahan Huruf Kapital Pada Karya Ilmiah". *Journal of Computer and Information Systems Ampera*, 1(1), 53-62.

Kasyanto, R. D., Syifa, F. T. dan Indriyanto, S. 2025. "Prototipe Sistem Pemanggil Pelayan di Restoran Menggunakan Metode Nirkabel dan ESP32 dengan Protokol MQTT". *Edu Elektrika Journal*, 13(2).

Kusmiati, M., Nurpalah, R. dan Restaviani, R. 2022. "Presisi dan Akurasi Hasil Quality Control Pada Parameter Pemeriksaan Glukosa Darah Di Laboratorium Klinik Rumah Sakit X Kota Tasikmalaya". *Journal of Indonesian Medical Laboratory and Science (JoIMedLabS)*, 3(1), 27-37.



**Hak Cipta :**

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Laaroussi, Z. dan Novo, O. (2021). A Performance Analysis of the Security Communication in CoAP and MQTT. *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*. DOI: 10.1109/CCNC49032.2021.9369565

Mahmudin, M. dan Hariyono, R. C. S. 2022. "Penggunaan IP Publik Virtual Private Server (VPS) Untuk Akses Server Lokal Menggunakan Virtual Private Network (VPN) Tunnel". *IJIR (Indonesian Journal of Informatics and Research)*, 3(1), 58-63.

Mishra, B., Mishra, B. dan Kertész, A. 2021. "Stress-Testing MQTT Brokers: A Comparative Analysis of Performance Measurements". *Energies*, 14(19), 5817.

Mubarok, K. dan Romli, M. A. 2025. "Implementasi Metode *Rule-Based* dalam Mendeteksi Serangan Brute Force pada Owncloud". *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, 5(1), 159-167.

Munte, R. S., Risnita, R., Jailani, M. S. dan Siregar, I. 2023. "Jenis Penelitian Eksperimen dan Noneksperimen (Design Kausal Komparatif dan Design Korelasional)". *Jurnal Pendidikan Tambusai*, 7(3), 22350-22359.

Nugraha, K. A. 2024. "Efisiensi Pertukaran Data Client-Server menggunakan Web Socket pada Perangkat Berbasis Internet of things". *JEPIN (Jurnal Edukasi dan Penelitian Informatika)*, 10(1).

Nugraha, I. R., Putra, W. H. N. dan Setiawan, E. 2024. "A Comparative Study of HTTP and MQTT for IoT Applications in Hydroponics". *Rekayasa Sistem dan Teknologi Informasi (RESTI)*, 8(1), 119-126.

Nusri, A. Z., dan Syah, R. E. (2025). Analisis Trafik Jaringan Menggunakan Wireshark Untuk Meningkatkan Kinerja Jaringan Pada Smk 3 Soppeng. *Jurnal Ilmiah Sistem Informasi dan Teknik Informatika (JISTI)*, 8(1), 114-122.





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Parlika, R., Erayanti, A. E., Orissa, D. F., Ummam, M. A. I. dan Abrori, M. R. 2021. "Studi Literatur Kaitan Antara Rekayasa Kebutuhan dan Rekayasa Sistem". *Jurnal Sistem Informasi dan Bisnis Cerdas (SIBC)*, 14(1), 37-44.

Prabowo, M. C. A., Janitra, A. A. dan Wibowo, N. M. 2023. "Sistem Monitoring Hidroponik Berbasis IoT Dengan Sensor Suhu, pH, dan Ketinggian Air Menggunakan ESP8266". *TECNOSCIENZA*, 7(2), 322.

Prasetyoaji, M. R. E. dan Anifah, L. 2025. "Design of Smart Hydroponics Based on Internet of things (IoT) Using Rule-Based System on Hydroponic Wick System". *Journal FORTEI-JEERI*, 6(1), 51-62.

Pratama, F., Wijaya, I. G. P. S. dan Albar, M. A. 2023. "Evaluation of Self-healing Mechanisms and Reconnection Latency in MQTT-Based Smart Agriculture Devices". *IEEE International Conference on Informatics and Computing (ICIC)*, 204-210.

Pratama, I. A., Harsani, P. dan Suriansyah, M. I. 2024. "Perbandingan Pemrosesan Kinerja Server Raspberry dan PC untuk Optimalisasi Smart Farming Berbasis IoT". *JITET (Jurnal Informatika dan Teknik Elektro Terapan)*, 12(1).

Rahman, R., dan Imamulhair, M. (2024). Analisis jaringan internet menggunakan Wireshark pada warkop. *Technology Sciences Insights Journal*, 1(2), 66-70.

Rizal, F. J., Rahman, M. A., dan Maulana, A. A. (2024). Implementasi Smart Farming Dalam Mendukung Pertanian Berkelanjutan. *ZIRAA'AH*, 49(1), 120-126.

Safara, A. R., Alamin, M. M., Antika, D. I. F., Khoiroh, H., dan Syaifudin, N. (2025). Penggunaan Wireshark untuk Menganalisis Kualitas Layanan Jaringan Internet pada Program Wi-Fi Bumdes. *Nusantara Computer and Design Review*, 3(1), 38-46.



## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Safitri, N. A. dan Priambodo, A. S. 2023. "MQTT and CoAP Communication Protocol Analysis in Internet of things System for Strawberry Hydroponic Plants". *Journal of Robotics, Automation, and Electronics Engineering*, 1(1), 45-56.

Suryadi, A., Wahyuddin, M. I. dan Alamsyah, R. J. S. 2024. "Performance and Recovery Time Analysis of MQTT Protocol Under Network Disruption in IoT Systems". *Jurnal Ilmiah Teknik Elektro Komputer dan Informatika (JITEKI)*, 10(1), 45-56.

Syukhron, I., Rahmadewi, R. dan Ibrahim, I. 2021. "Penggunaan Aplikasi Blynk Untuk Monitoring dan Kontrol Jarak Jauh pada Sistem Kompos Pintar Berbasis IoT". *ELECTRICIAN - Jurnal Rekayasa dan Teknologi Elektro*, 15(1).

Turnip, A., Pebriansyah, F. R., Simarmata, T., Sihombing, P. dan Joelianto, E. 2023. "Design of Smart Farming communication and web interface using MQTT and Node.js". *Open Agriculture*, 8.

Wati, D. R. dan Sholihah, W. 2021. "Pengontrol pH dan Nutrisi Tanaman Selada pada Hidroponik Sistem NFT Berbasis Arduino". *Jurnal Multinetics*, 7(1).

Yakub, H., Daniawan, B., Wijaya, A. dan Damayanti, L. 2024. "Sistem Informasi E-Commerce Berbasis Website Dengan Metode Pengujian User Acceptance Testing". *JSITIK: Jurnal Sistem Informasi dan Teknologi Informasi Komputer*, 2(2), 113-127.

Zakiah, S., Bintoro, A., Muthalib, M. A. dan Ikhwanus, M. 2025. "Alat Pengaturan Pencampuran Nutrisi Pada Tanaman Hidroponik Berbasis Internet of things". *Jurnal Energi Elektrik*, 14(1), 21-30.



## DAFTAR RIWAYAT HIDUP



Kevin Alonzo Manuel Bakara lahir di Bogor tahun 2004. Penulis menyelesaikan pendidikan dasar di SD BPK Penabur Bogor pada tahun 2016, dilanjutkan ke tingkat menengah di SMP BPK Penabur Bogor dan lulus pada tahun 2019, lalu melanjutkan ke sekolah tingkat atas di SMA Negeri 6 Kota Bogor dengan peminatan Matematika dan Ilmu Pengetahuan Alam (MIPA), lulus pada tahun 2022. Saat ini, penulis sedang menyelesaikan pendidikan sarjana terapan di Politeknik Negeri Jakarta, Jurusan Teknik Informatika dan Komputer, Program Studi Teknik Multimedia dan Jaringan. Selama menempuh masa perkuliahan, penulis aktif berorganisasi dalam Kelompok Studi Mahasiswa (KSM) serta berpartisipasi dan mengembangkan kompetensi melalui perlombaan, baik di bidang akademik maupun non-akademik.

**POLITEKNIK  
NEGERI  
JAKARTA**

© Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta



- Hak Cipta :**
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
    - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
    - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
  2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

## LAMPIRAN

### Lampiran 1. Surat Izin Observasi



**KEMENTERIAN PENDIDIKAN TINGGI, SAINS,  
DAN TEKNOLOGI**  
**POLITEKNIK NEGERI JAKARTA**  
Jalan Prof. Dr. G. A. Siwabessy, Kampus UI, Depok 16425  
Telepon (021) 7270036, Hunting, Fax (021) 7270034  
Laman: <http://www.pnj.ac.id> Posel: [humas@pnj.ac.id](mailto:humas@pnj.ac.id)

---

Nomor : 81/DST/PL3.A.9/B/KM.07.00/2026  
Perihal : Permohonan Izin Observasi

21 Januari 2026

Kepada Yth.  
Chief Technology Officer PT Teknologi Lokatani Indonesia  
Perumahan Jl. Puri Sriwedari No.12 Blok P, Harjamukti, Kec. Cimanggis, Kota Depok, Jawa Barat 16421

Dengan hormat,

Sehubungan dengan mata kuliah Skripsi yang dilaksanakan pada semester 8 (delapan) Program Studi Teknik Multimedia dan Jaringan Jurusan Teknik Informatika dan Komputer Politeknik Negeri Jakarta. Dengan ini kami mohon kesediaan Bapak/Ibu agar dapat mengizinkan mahasiswa kami untuk melakukan observasi di **PT Teknologi Lokatani Indonesia** dengan judul penelitian "Rancang Bangun Sistem Smart Hydroponic Berbasis Internet Of Things".

Tugas mata kuliah ini bertujuan untuk menambah wawasan terkait dengan aplikasi teori yang sudah dipelajari di Kampus dengan kondisi lapangan sebagai wadah pembelajaran dan penambah informasi mengenai mata kuliah tersebut. Adapun berikut adalah nama mahasiswa kami:

| No. | Nama dan Nim                                   | Semester/<br>Program Studi               | Nomor Telepon dan<br>Email                                                                                                         | Tujuan Observasi                                                 |
|-----|------------------------------------------------|------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------|
| 1   | Muhammad<br>Brian Azura<br>Nixon<br>2207421056 | 8 / Teknik<br>Multimedia dan<br>Jaringan | 081213777077<br><a href="mailto:muhammad.brian.a.nixon.tik22@mhs.pnj.ac.id">muhammad.brian.a.nixon.tik22@mhs.pnj.ac.id</a>         | Ingin mengetahui data-data penunjang dan wawancara untuk Skripsi |
| 2   | Kevin Alonzo<br>Manuel Bakara<br>2207421032    |                                          | 085814528872<br><a href="mailto:kevin.alonzo.manuel.bakara.tik22@mhs.pnj.ac.id">kevin.alonzo.manuel.bakara.tik22@mhs.pnj.ac.id</a> |                                                                  |

Demikian surat ini kami buat, atas perhatian dan kerjasama Bapak/Ibu kami ucapkan terima kasih.



Ketua Jurusan,  
Dr. Anita Hidayati, S.Kom., M.Kom.  
NIP. 197908032003122003

Tembusan :

1. Direktur;
2. Wakil Direktur Bidang Akademik;
3. Kepala Bagian Keuangan dan Umum
4. Kasubbag. Umum Politeknik Negeri Jakarta.





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

## Lampiran 2. Dokumentasi Observasi dan Implementasi





### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

## Lampiran 3. Instrumen dan Hasil *User Acceptance Test* (UAT)







## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

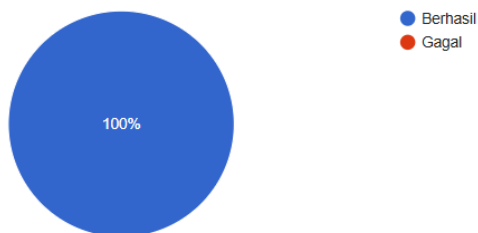
### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

#### Pemantauan Data Waktu-Nyata (*Real-Time*)

[Copy chart](#)

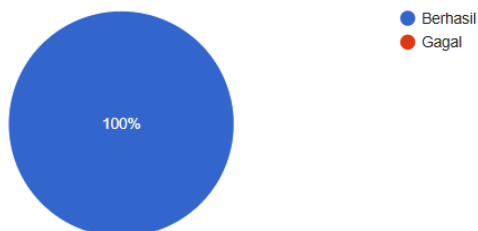
1 response



#### Verifikasi Mesin Keputusan (*Rule-Based Helper*)

[Copy chart](#)

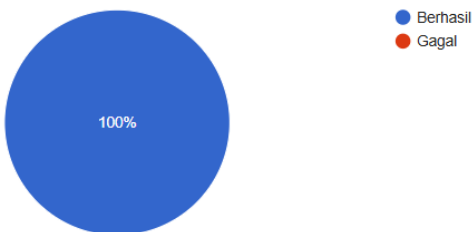
1 response



#### Pengecekan Riwayat Log Data

[Copy chart](#)

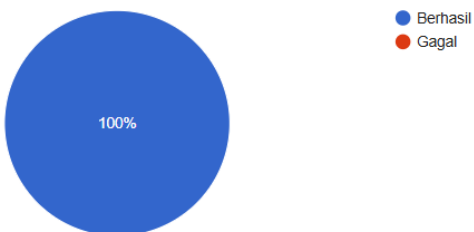
1 response



#### Pengujian Fitur Ekspor Data

[Copy chart](#)

1 response





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

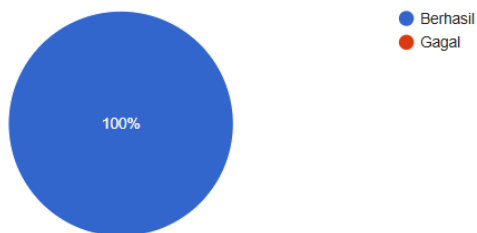
### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

#### Penyambungan Notifikasi Telegram (*Binding*)

Copy chart

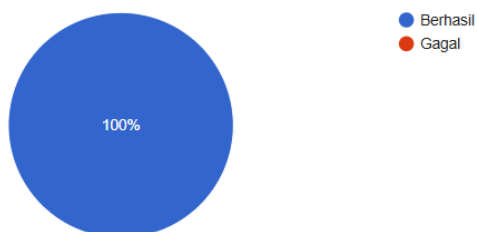
1 response



#### Pemutusan Koneksi Telegram (*Unbinding*)

Copy chart

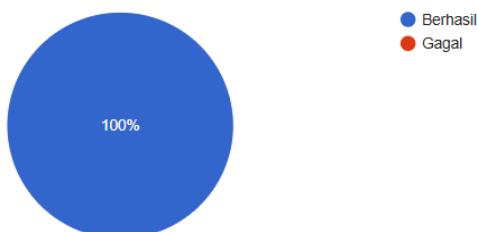
1 response



#### Pengujian Responsivitas Tata Letak

Copy chart

1 response







## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

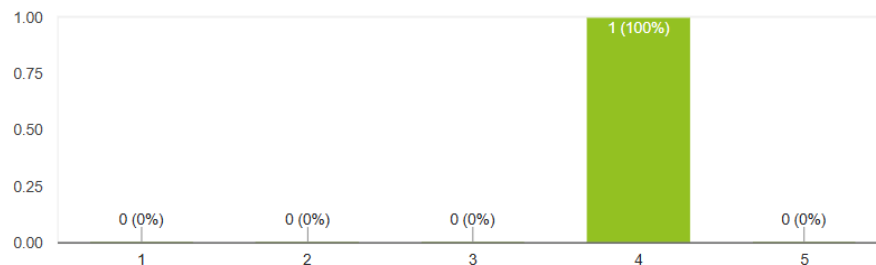
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

### Penilaian Usability & Kepuasan

[Copy chart](#)

Tata letak menu dan fungsi tombol pada aplikasi ini sangat intuitif dan mudah dipahami pada penggunaan pertama.

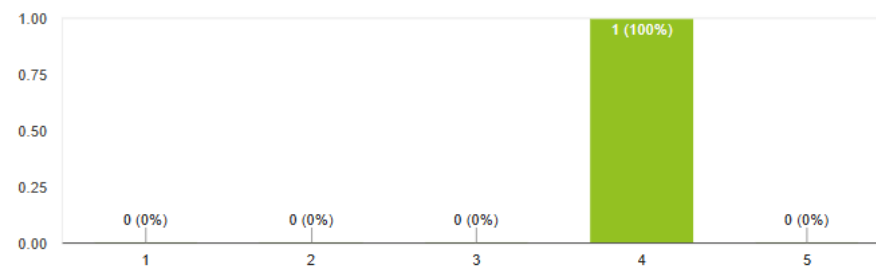
1 response



[Copy chart](#)

Visualisasi grafik sensor dan kontras warna pada antarmuka dasbor sangat nyaman dibaca.

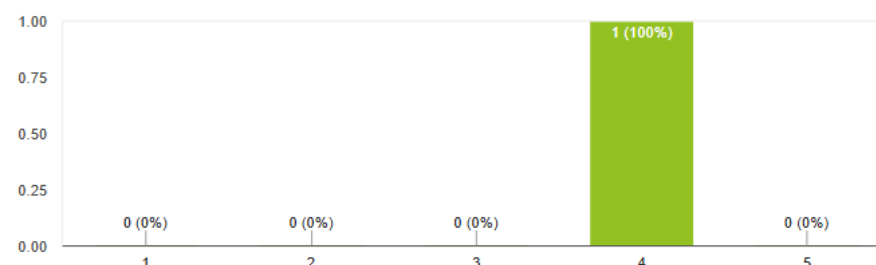
1 response



[Copy chart](#)

Rekomendasi tindakan yang diberikan oleh fitur *Decision Helper* sangat jelas dan mudah dieksekusi oleh awam.

1 response





## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

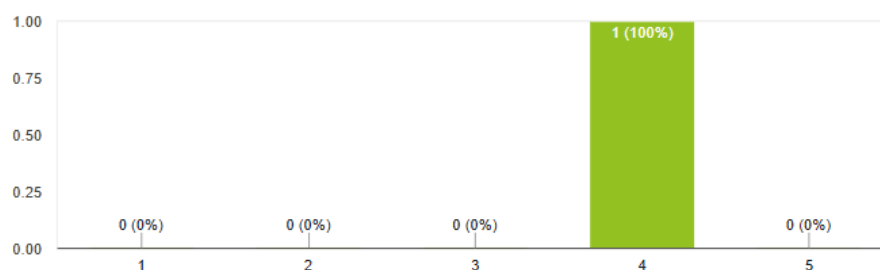
### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

Transisi perpindahan antar-menu serta proses pemuatan tabel terasa cepat dan tidak lambat (*lag*).

Copy chart

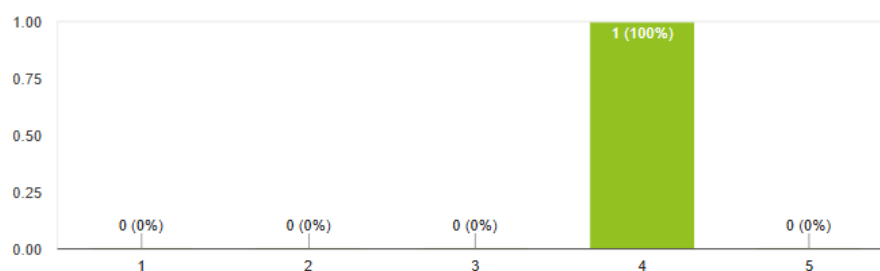
1 response



Keberadaan fitur ekspor Excel dan push-notifikasi Telegram membuat aplikasi ini sangat layak digunakan di lahan perkebunan riil.

Copy chart

1 response

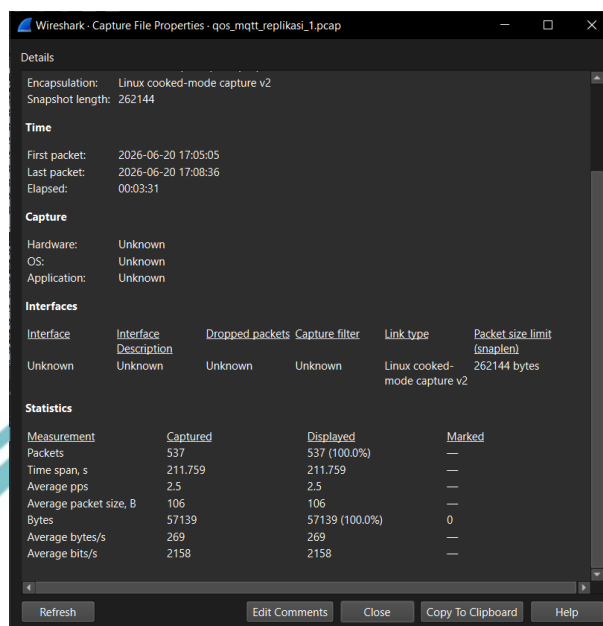


NEGERI  
JAKARTA





## Lampiran 4. Pengujian QoS MQTT



| seqId | waktu_database      | ph         | ppm | protocol | emitted_at_raw | received_at_raw |
|-------|---------------------|------------|-----|----------|----------------|-----------------|
| 1     | 20/6/2026, 10.05.05 | 0          | 871 | mqtt     | 1781949905330  | 1781949905541   |
| 2     | 20/6/2026, 10.05.10 | 0          | 873 | mqtt     | 1781949909964  | 1781949910155   |
| 3     | 20/6/2026, 10.05.15 | 0          | 873 | mqtt     | 1781949914964  | 1781949915180   |
| 4     | 20/6/2026, 10.05.20 | 0          | 873 | mqtt     | 1781949919964  | 1781949920163   |
| 5     | 20/6/2026, 10.05.25 | 0          | 873 | mqtt     | 1781949924964  | 1781949925083   |
| 6     | 20/6/2026, 10.05.30 | 0          | 873 | mqtt     | 1781949929964  | 1781949930106   |
| 7     | 20/6/2026, 10.05.35 | 0          | 873 | mqtt     | 1781949934964  | 1781949935115   |
| 8     | 20/6/2026, 10.05.40 | 7,09999991 | 873 | mqtt     | 1781949940116  | 1781949940254   |
| 9     | 20/6/2026, 10.05.45 | 7,09999991 | 873 | mqtt     | 1781949945115  | 1781949945280   |
| 10    | 20/6/2026, 10.05.50 | 7,09999991 | 873 | mqtt     | 1781949950115  | 1781949950280   |
| 11    | 20/6/2026, 10.05.55 | 7,09999991 | 873 | mqtt     | 1781949955115  | 1781949955291   |
| 12    | 20/6/2026, 10.06.00 | 7,09999991 | 855 | mqtt     | 1781949960300  | 1781949960507   |
| 13    | 20/6/2026, 10.06.05 | 7,09999991 | 850 | mqtt     | 1781949965299  | 1781949965435   |
| 14    | 20/6/2026, 10.06.10 | 7,09999991 | 850 | mqtt     | 1781949970299  | 1781949970451   |
| 15    | 20/6/2026, 10.06.15 | 7,09999991 | 850 | mqtt     | 1781949975299  | 1781949975464   |
| 16    | 20/6/2026, 10.06.20 | 7,09999991 | 850 | mqtt     | 1781949980299  | 1781949980480   |
| 17    | 20/6/2026, 10.06.25 | 7,09999991 | 850 | mqtt     | 1781949985299  | 1781949985497   |
| 18    | 20/6/2026, 10.06.30 | 7,09999991 | 850 | mqtt     | 1781949990299  | 1781949990423   |
| 19    | 20/6/2026, 10.06.35 | 7,09999991 | 850 | mqtt     | 1781949995647  | 1781949995841   |
| 20    | 20/6/2026, 10.06.40 | 7,09999991 | 850 | mqtt     | 1781950000646  | 1781950000862   |
| 21    | 20/6/2026, 10.06.45 | 7,09999991 | 850 | mqtt     | 1781950005646  | 1781950005776   |
| 22    | 20/6/2026, 10.06.50 | 7,09999991 | 850 | mqtt     | 1781950010646  | 1781950010809   |
| 23    | 20/6/2026, 10.06.56 | 7,09999991 | 829 | mqtt     | 1781950015834  | 1781950016012   |
| 24    | 20/6/2026, 10.07.01 | 7,09999991 | 822 | mqtt     | 1781950020833  | 1781950021035   |
| 25    | 20/6/2026, 10.07.05 | 7,09999991 | 822 | mqtt     | 1781950025833  | 1781950025946   |
| 26    | 20/6/2026, 10.07.10 | 7,09999991 | 822 | mqtt     | 1781950030833  | 1781950030981   |
| 27    | 20/6/2026, 10.07.15 | 7,09999991 | 822 | mqtt     | 1781950035833  | 1781950035981   |
| 28    | 20/6/2026, 10.07.21 | 7,09999991 | 822 | mqtt     | 1781950040833  | 1781950041019   |
| 29    | 20/6/2026, 10.07.26 | 7,09999991 | 822 | mqtt     | 1781950045833  | 1781950046011   |
| 30    | 20/6/2026, 10.07.31 | 7,09999991 | 822 | mqtt     | 1781950051181  | 1781950051345   |
| 31    | 20/6/2026, 10.07.36 | 7,09999991 | 822 | mqtt     | 1781950056180  | 1781950056356   |
| 32    | 20/6/2026, 10.07.41 | 7,09999991 | 822 | mqtt     | 1781950061180  | 1781950061376   |
| 33    | 20/6/2026, 10.07.46 | 7,09999991 | 822 | mqtt     | 1781950066180  | 1781950066287   |
| 34    | 20/6/2026, 10.07.51 | 7,09999991 | 786 | mqtt     | 1781950071366  | 1781950071522   |

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta



### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|    |                     |            |     |      |               |               |
|----|---------------------|------------|-----|------|---------------|---------------|
| 35 | 20/6/2026, 10.07.56 | 7,09999991 | 773 | mqtt | 1781950076365 | 1781950076547 |
| 36 | 20/6/2026, 10.08.01 | 7,09999991 | 773 | mqtt | 1781950081365 | 1781950081577 |
| 37 | 20/6/2026, 10.08.06 | 7,09999991 | 773 | mqtt | 1781950086365 | 1781950086567 |
| 38 | 20/6/2026, 10.08.11 | 7,09999991 | 773 | mqtt | 1781950091365 | 1781950091483 |
| 39 | 20/6/2026, 10.08.16 | 7,09999991 | 773 | mqtt | 1781950096365 | 1781950096520 |
| 40 | 20/6/2026, 10.08.21 | 7,09999991 | 773 | mqtt | 1781950101365 | 1781950101522 |
| 41 | 20/6/2026, 10.08.26 | 7,09999991 | 773 | mqtt | 1781950106719 | 1781950106862 |
| 42 | 20/6/2026, 10.08.31 | 7,09999991 | 773 | mqtt | 1781950111718 | 1781950111858 |

### Lampiran 5. Pengujian QoS HTTP

| seqId | waktu_database      | ph         | ppm | protocol | emitted_at_raw | received_at_raw |
|-------|---------------------|------------|-----|----------|----------------|-----------------|
| 1     | 20/6/2026, 10.36.01 | 0          | 872 | http     | 1781951761293  | 1781951761875   |
| 2     | 20/6/2026, 10.36.06 | 0          | 872 | http     | 1781951765433  | 1781951766035   |
| 3     | 20/6/2026, 10.36.10 | 0          | 872 | http     | 1781951770432  | 1781951770859   |
| 4     | 20/6/2026, 10.36.15 | 0          | 872 | http     | 1781951775432  | 1781951775983   |
| 5     | 20/6/2026, 10.36.20 | 0          | 872 | http     | 1781951780439  | 1781951780986   |
| 6     | 20/6/2026, 10.36.26 | 0          | 872 | http     | 1781951785457  | 1781951786012   |
| 7     | 20/6/2026, 10.36.31 | 0          | 872 | http     | 1781951790474  | 1781951791025   |
| 8     | 20/6/2026, 10.36.36 | 6,80000019 | 872 | http     | 1781951795558  | 1781951796039   |
| 9     | 20/6/2026, 10.36.41 | 6,80000019 | 872 | http     | 1781951800557  | 1781951801160   |
| 10    | 20/6/2026, 10.36.46 | 6,80000019 | 872 | http     | 1781951805557  | 1781951806083   |
| 11    | 20/6/2026, 10.36.50 | 6,80000019 | 872 | http     | 1781951810557  | 1781951810998   |
| 12    | 20/6/2026, 10.36.56 | 6,80000019 | 872 | http     | 1781951815653  | 1781951816211   |
| 13    | 20/6/2026, 10.37.01 | 6,80000019 | 875 | http     | 1781951820652  | 1781951821237   |
| 14    | 20/6/2026, 10.37.06 | 6,80000019 | 875 | http     | 1781951825652  | 1781951826371   |
| 15    | 20/6/2026, 10.37.11 | 6,80000019 | 875 | http     | 1781951830652  | 1781951831168   |
| 16    | 20/6/2026, 10.37.16 | 6,80000019 | 875 | http     | 1781951835652  | 1781951836182   |
| 17    | 20/6/2026, 10.37.21 | 6,80000019 | 875 | http     | 1781951840653  | 1781951841205   |
| 18    | 20/6/2026, 10.37.26 | 6,80000019 | 875 | http     | 1781951845671  | 1781951846231   |
| 19    | 20/6/2026, 10.37.31 | 6,80000019 | 875 | http     | 1781951850672  | 1781951851231   |
| 20    | 20/6/2026, 10.37.36 | 6,80000019 | 875 | http     | 1781951855671  | 1781951856477   |
| 21    | 20/6/2026, 10.37.41 | 6,80000019 | 875 | http     | 1781951860671  | 1781951861266   |
| 22    | 20/6/2026, 10.37.46 | 6,80000019 | 875 | http     | 1781951865671  | 1781951866182   |



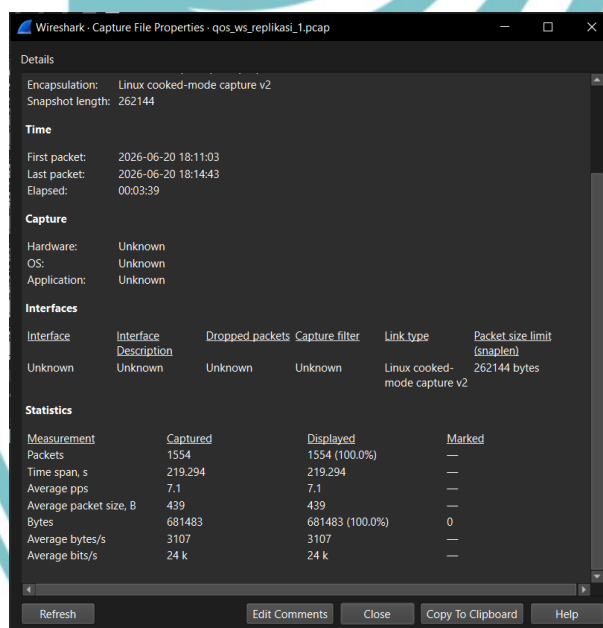


### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|    |                     |            |     |      |               |               |
|----|---------------------|------------|-----|------|---------------|---------------|
| 23 | 20/6/2026, 10.37.51 | 6,80000019 | 895 | http | 1781951870679 | 1781951871203 |
| 24 | 20/6/2026, 10.37.56 | 6,80000019 | 915 | http | 1781951875687 | 1781951876217 |
| 25 | 20/6/2026, 10.38.01 | 6,80000019 | 915 | http | 1781951880693 | 1781951881242 |
| 26 | 20/6/2026, 10.38.06 | 6,80000019 | 915 | http | 1781951885711 | 1781951886252 |
| 27 | 20/6/2026, 10.38.11 | 6,80000019 | 915 | http | 1781951890728 | 1781951891276 |
| 28 | 20/6/2026, 10.38.16 | 6,80000019 | 915 | http | 1781951895747 | 1781951896291 |
| 29 | 20/6/2026, 10.38.21 | 6,80000019 | 915 | http | 1781951900764 | 1781951901319 |
| 30 | 20/6/2026, 10.38.26 | 6,80000019 | 915 | http | 1781951906166 | 1781951906772 |
| 31 | 20/6/2026, 10.38.31 | 6,80000019 | 915 | http | 1781951911165 | 1781951911751 |
| 32 | 20/6/2026, 10.38.36 | 6,80000019 | 915 | http | 1781951916165 | 1781951916671 |
| 33 | 20/6/2026, 10.38.41 | 6,80000019 | 915 | http | 1781951921165 | 1781951921701 |
| 34 | 20/6/2026, 10.38.46 | 6,80000019 | 911 | http | 1781951926183 | 1781951926757 |
| 35 | 20/6/2026, 10.38.51 | 6,80000019 | 908 | http | 1781951931190 | 1781951931745 |
| 36 | 20/6/2026, 10.38.56 | 6,80000019 | 908 | http | 1781951936196 | 1781951936744 |
| 37 | 20/6/2026, 10.39.01 | 6,80000019 | 908 | http | 1781951941214 | 1781951941768 |
| 38 | 20/6/2026, 10.39.06 | 6,80000019 | 908 | http | 1781951946232 | 1781951946779 |
| 39 | 20/6/2026, 10.39.11 | 6,80000019 | 908 | http | 1781951951250 | 1781951951828 |
| 40 | 20/6/2026, 10.39.16 | 6,80000019 | 908 | http | 1781951956268 | 1781951956835 |
| 41 | 20/6/2026, 10.39.22 | 6,80000019 | 908 | http | 1781951961669 | 1781951962300 |
| 42 | 20/6/2026, 10.39.27 | 6,80000019 | 908 | http | 1781951966668 | 1781951967259 |

Lampiran 6. Pengujian QoS WebSocket



| seqid | waktu_database      | ph  | ppm | protocol  | emitted_at_raw | received_at_raw |
|-------|---------------------|-----|-----|-----------|----------------|-----------------|
| 1     | 20/6/2026, 11.11.09 | 0   | 873 | websocket | 1781953869472  | 1781953869655   |
| 2     | 20/6/2026, 11.11.14 | 0   | 873 | websocket | 1781953874472  | 1781953874684   |
| 3     | 20/6/2026, 11.11.19 | 0   | 873 | websocket | 1781953879472  | 1781953879688   |
| 4     | 20/6/2026, 11.11.24 | 0   | 873 | websocket | 1781953884472  | 1781953884741   |
| 5     | 20/6/2026, 11.11.29 | 0   | 873 | websocket | 1781953889472  | 1781953889634   |
| 6     | 20/6/2026, 11.11.34 | 0   | 873 | websocket | 1781953894472  | 1781953894638   |
| 7     | 20/6/2026, 11.11.39 | 6,5 | 873 | websocket | 1781953899736  | 1781953899971   |
| 8     | 20/6/2026, 11.11.44 | 6,5 | 873 | websocket | 1781953904736  | 1781953904885   |
| 9     | 20/6/2026, 11.11.49 | 6,5 | 873 | websocket | 1781953909736  | 1781953909919   |
| 10    | 20/6/2026, 11.11.54 | 6,5 | 873 | websocket | 1781953914736  | 1781953914912   |
| 11    | 20/6/2026, 11.12.00 | 6,5 | 890 | websocket | 1781953919944  | 1781953920136   |



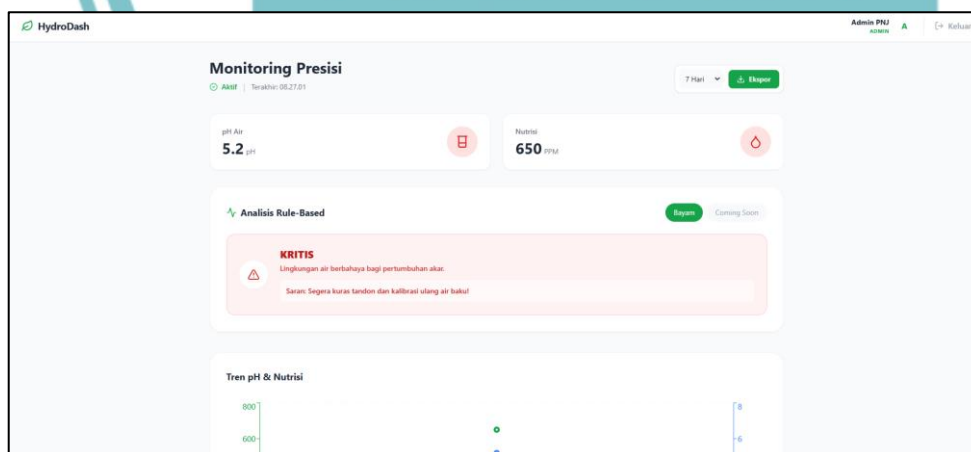
## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

|    |                     |            |     |           |               |               |
|----|---------------------|------------|-----|-----------|---------------|---------------|
| 12 | 20/6/2026, 11.12.05 | 6,5        | 944 | websocket | 1781953925027 | 1781953925285 |
| 13 | 20/6/2026, 11.12.10 | 6,5        | 944 | websocket | 1781953930027 | 1781953930195 |
| 14 | 20/6/2026, 11.12.15 | 6,5        | 944 | websocket | 1781953935027 | 1781953935195 |
| 15 | 20/6/2026, 11.12.20 | 6,5        | 944 | websocket | 1781953940027 | 1781953940240 |
| 16 | 20/6/2026, 11.12.25 | 6,5        | 944 | websocket | 1781953945027 | 1781953945244 |
| 17 | 20/6/2026, 11.12.30 | 6,5        | 944 | websocket | 1781953950027 | 1781953950239 |
| 18 | 20/6/2026, 11.12.35 | 6,5        | 944 | websocket | 1781953955323 | 1781953955502 |
| 19 | 20/6/2026, 11.12.40 | 6,5        | 944 | websocket | 1781953960323 | 1781953960531 |
| 20 | 20/6/2026, 11.12.45 | 6,5        | 944 | websocket | 1781953965323 | 1781953965516 |
| 21 | 20/6/2026, 11.12.50 | 6,5        | 944 | websocket | 1781953970323 | 1781953970535 |
| 22 | 20/6/2026, 11.12.55 | 6,5        | 951 | websocket | 1781953975532 | 1781953975748 |
| 23 | 20/6/2026, 11.13.00 | 6,5        | 938 | websocket | 1781953980615 | 1781953980756 |
| 24 | 20/6/2026, 11.13.05 | 6,5        | 938 | websocket | 1781953985615 | 1781953985774 |
| 25 | 20/6/2026, 11.13.10 | 6,5        | 938 | websocket | 1781953990615 | 1781953990796 |
| 26 | 20/6/2026, 11.13.15 | 6,5        | 938 | websocket | 1781953995615 | 1781953995807 |
| 27 | 20/6/2026, 11.13.20 | 6,5        | 938 | websocket | 1781954000615 | 1781954000865 |
| 28 | 20/6/2026, 11.13.25 | 6,5        | 938 | websocket | 1781954005615 | 1781954005840 |
| 29 | 20/6/2026, 11.13.30 | 6,5        | 938 | websocket | 1781954010747 | 1781954010994 |
| 30 | 20/6/2026, 11.13.35 | 6,69999981 | 938 | websocket | 1781954015747 | 1781954015906 |
| 31 | 20/6/2026, 11.13.40 | 6,69999981 | 938 | websocket | 1781954020747 | 1781954020918 |
| 32 | 20/6/2026, 11.13.45 | 6,69999981 | 938 | websocket | 1781954025747 | 1781954025923 |
| 33 | 20/6/2026, 11.13.51 | 6,69999981 | 948 | websocket | 1781954030955 | 1781954031140 |
| 34 | 20/6/2026, 11.13.56 | 6,69999981 | 964 | websocket | 1781954036038 | 1781954036280 |
| 35 | 20/6/2026, 11.14.01 | 6,69999981 | 964 | websocket | 1781954041038 | 1781954041185 |
| 36 | 20/6/2026, 11.14.06 | 6,69999981 | 964 | websocket | 1781954046038 | 1781954046201 |
| 37 | 20/6/2026, 11.14.11 | 6,69999981 | 964 | websocket | 1781954051038 | 1781954051232 |
| 38 | 20/6/2026, 11.14.16 | 6,69999981 | 964 | websocket | 1781954056038 | 1781954056262 |
| 39 | 20/6/2026, 11.14.21 | 6,69999981 | 964 | websocket | 1781954061038 | 1781954061238 |
| 40 | 20/6/2026, 11.14.26 | 6,69999981 | 964 | websocket | 1781954066347 | 1781954066567 |
| 41 | 20/6/2026, 11.14.31 | 6,69999981 | 964 | websocket | 1781954071347 | 1781954071522 |
| 42 | 20/6/2026, 11.14.36 | 6,69999981 | 964 | websocket | 1781954076347 | 1781954076522 |

Lampiran 7. Bukti Pengujian Algoritma *Rule-Based*

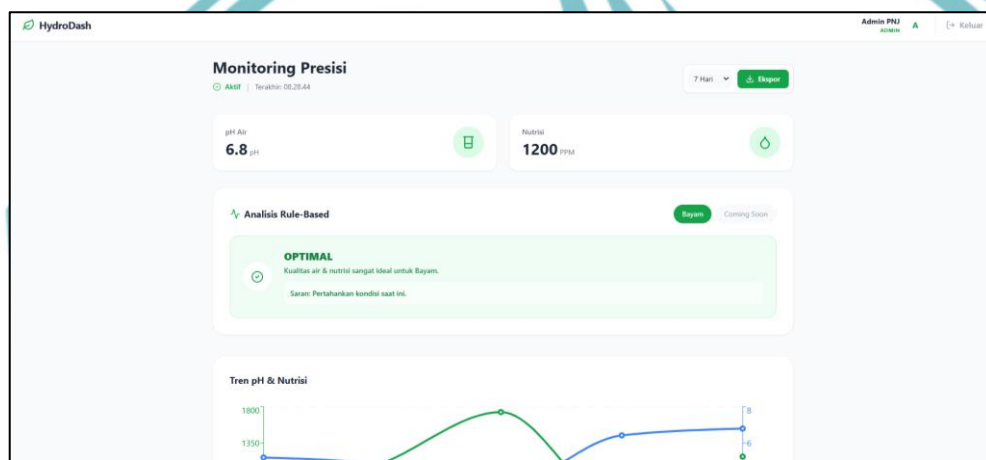
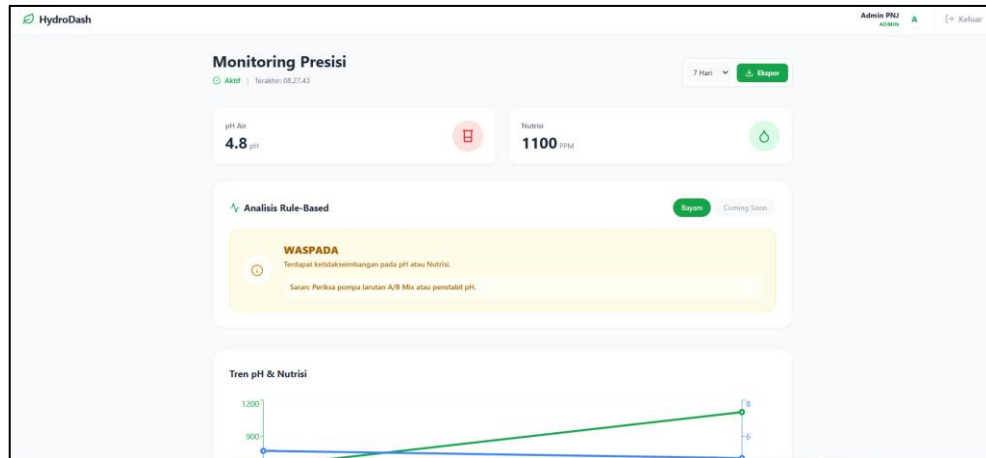




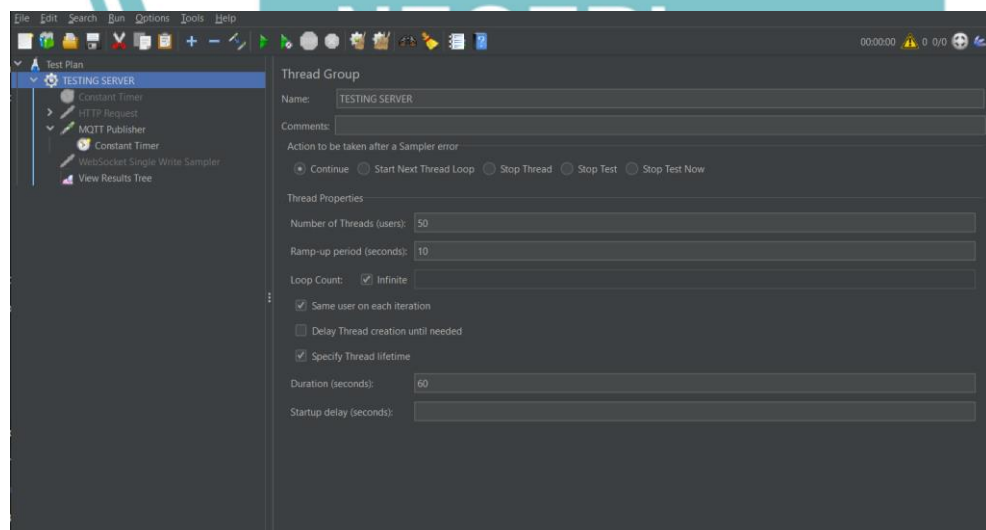


### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta



Lampiran 8. Bukti Pengujian Load Testing

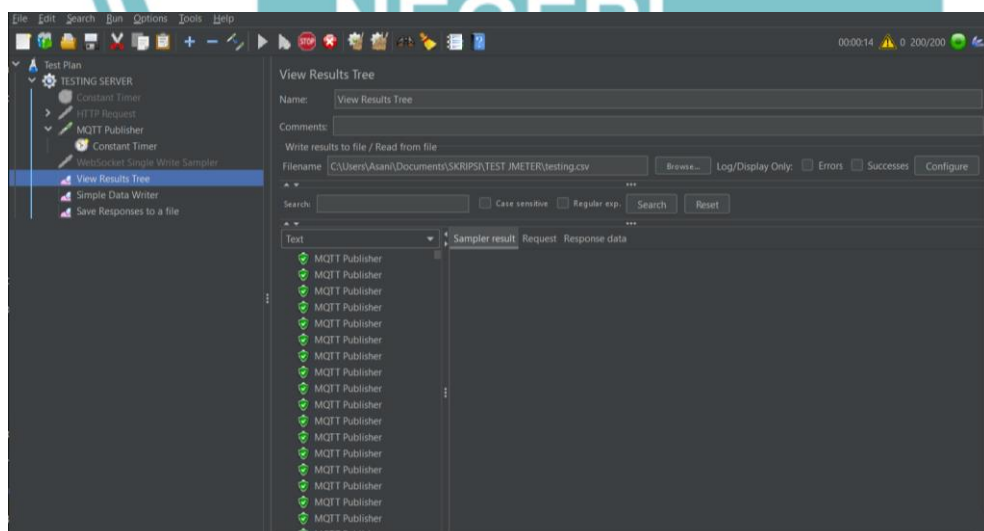
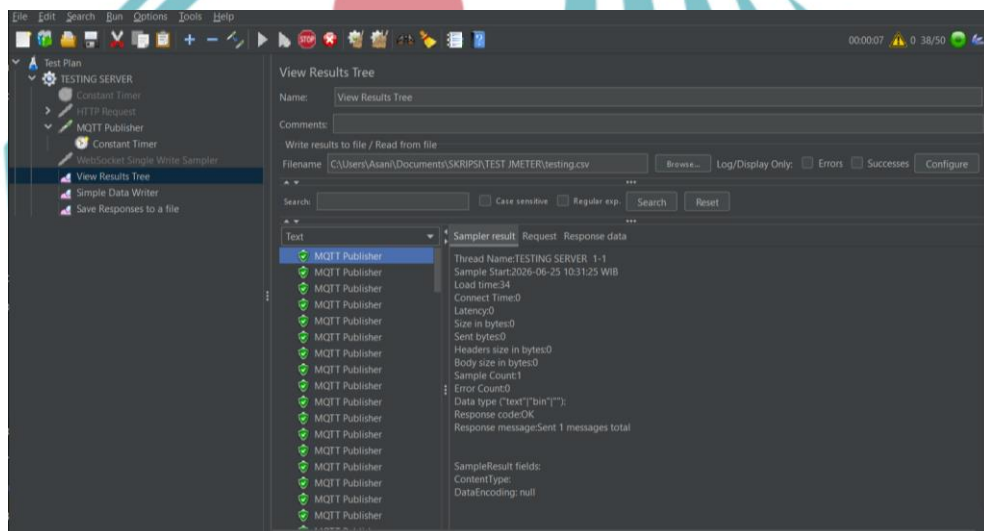
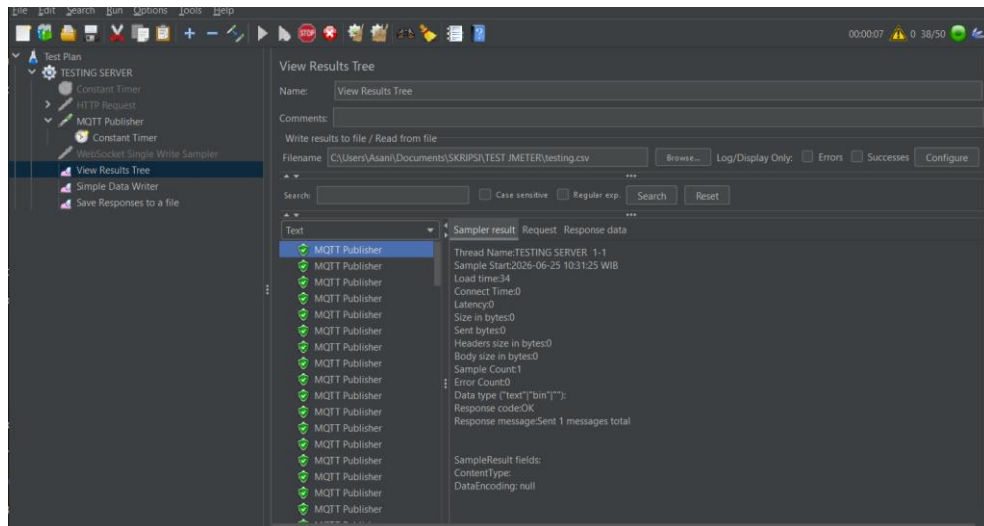




## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpulkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta







## © Hak Cipta milik Jurusan TIK Politeknik Negeri Jakarta

### Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
  - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
  - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin dari Jurusan TIK Politeknik Negeri Jakarta

| timeStamp     | elapsed (ms) | responseCode | CPU_Total_% | Mosquitto_RSS (MB) | NodeJS_RSS (MB) |
|---------------|--------------|--------------|-------------|--------------------|-----------------|
| 1782359633498 | 24           | OK           | 4.80        | 8.40               | 101.88          |
| 1782359633561 | 19           | OK           | 4.80        | 8.40               | 101.88          |
| 1782359633607 | 20           | OK           | 5.00        | 8.40               | 101.88          |
| 1782359633653 | 21           | OK           | 5.00        | 8.41               | 103.20          |
| 1782359633714 | 16           | OK           | 23.80       | 8.43               | 104.95          |
| 1782359633760 | 42           | OK           | 20.00       | 8.46               | 109.23          |
| 1782359633805 | 45           | OK           | 37.50       | 8.48               | 113.51          |
| 1782359633866 | 26           | OK           | 50.00       | 8.50               | 119.56          |
| 1782359633911 | 28           | OK           | 50.00       | 8.52               | 126.34          |
| 1782359633972 | 23           | OK           | 71.40       | 8.54               | 137.19          |
| 1782359634017 | 23           | OK           | 40.00       | 8.56               | 142.92          |
| 1782359634064 | 32           | OK           | 57.10       | 8.56               | 141.93          |
| 1782359634110 | 62           | OK           | 33.30       | 8.56               | 141.33          |
| 1782359634171 | 110          | OK           | 66.70       | 8.56               | 143.30          |
| 1782359634217 | 64           | OK           | 22.70       | 8.56               | 144.10          |
| 1782359634277 | 25           | OK           | 31.80       | 8.56               | 143.10          |
| 1782359634323 | 24           | OK           | 4.80        | 8.56               | 143.02          |
| 1782359634370 | 42           | OK           | 23.80       | 8.56               | 144.23          |
| 1782359634432 | 30           | OK           | 38.10       | 8.56               | 146.69          |
| 1782359634466 | 26           | OK           | 28.60       | 8.56               | 151.96          |

POLITEKNIK  
NEGERI  
JAKARTA