

BAB IV

PENGUJIAN DAN ANALISIS

4.1 Hasil Implementasi Sistem

Bagian ini berisi hasil implementasi sistem, pengujian sensor dan instrumentasi, pengujian komunikasi dan antarmuka HMI, pengujian keseluruhan sistem, serta analisis terhadap hasil pengujian yang diperoleh.

4.1.1 Implementasi Perangkat Keras

Implementasi perangkat keras merupakan tahapan realisasi fisik dari desain sistem kontrol yang telah dirancang. Tahapan ini mencakup perakitan kerangka mekanik, instalasi sistem perpipaan dan tangki, penempatan instrumen sensor dan aktuator, serta perakitan sirkuit elektronik di dalam panel kendali. Struktur utama alat ini ditopang oleh kerangka aluminium yang dirancang agar kokoh dan mampu menahan beban operasional dari dua buah tangki air, yaitu tangki utama berkapasitas 10 liter dan tangki penampungan berkapasitas 5 liter. Kedua tangki tersebut dihubungkan melalui sistem perpipaan yang dirangkai sedemikian rupa untuk memfasilitasi sirkulasi fluida yang efisien.

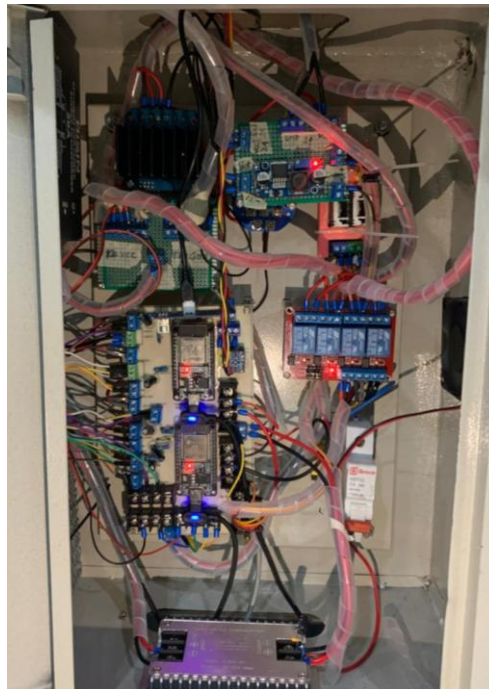


Gambar 4. 1 Foto Alat keseluruhan

Pada sistem sirkulasi, aktuasi aliran fluida digerakkan oleh pompa air DC 12V yang dikendalikan secara proporsional. Untuk mengakuisisi data variabel proses, sistem ini mengintegrasikan beberapa sensor yang ditempatkan secara strategis.

Sensor laju aliran air (*flow sensor*) YFS201 dipasang pada jalur pipa keluaran pompa untuk membaca debit air secara *real-time*. Pada tangki utama, dipasang sensor suhu PT100 yang terhubung dengan *RTD transmitter* untuk mengukur temperatur air, serta sensor ultrasonik JSN-SR04T yang dipasang di bagian atas tangki untuk memantau ketinggian permukaan air (*water level*). Selain itu, sistem ini juga dilengkapi dengan *heater* 220V 500W sebagai elemen pemanas utama dan modul pendingin termoelektrik Peltier TEC1-12706 untuk proses stabilisasi suhu.

Seluruh komponen komputasi dan manajemen daya dipusatkan di dalam sebuah *box panel* elektrik. Penggunaan *box panel* ini bertujuan untuk melindungi rangkaian dari paparan lingkungan eksternal, kelembapan fluida, serta meminimalisir gangguan interferensi elektromagnetik. Di dalam panel tersebut, terdapat mikrokontroler ESP32 yang berfungsi sebagai otak utama pengolah data. Sistem daya disuplai oleh *Power Supply Unit* (PSU) 24V yang kemudian didistribusikan menggunakan *step-down converter* untuk menghasilkan tegangan 12V dan 5V sesuai dengan kebutuhan kerja masing-masing perangkat. Rangkaian ini juga memuat *Solid State Relay* (SSR) untuk pengaturan daya *heater*, serta *driver motor* BTS7960 untuk mengeksekusi sinyal PWM menuju pompa dan Peltier.



Gambar 4. 2 Komponen Elektrikal di Dalam Panel Kendali

Sebagai antarmuka fisik antara mesin dan operator, layar *Human Machine Interface* (HMI) Haiwell B7H-W berukuran 7 inci dipasang pada sebuah *bracket* (dudukan) khusus yang difabrikasi menggunakan teknologi *3D printing*. Dudukan HMI ini diletakkan tepat di tengah meja utama, di antara kedua tangki. Penempatan ini dirancang secara ergonomis agar sejajar dengan pandangan operator, sehingga memudahkan pengamatan visual dan interaksi sentuh saat memantau status operasional alat maupun mengubah nilai *setpoint*.



Gambar 4. 3 Pemasangan HMI Haiwell pada Dudukan 3D Print

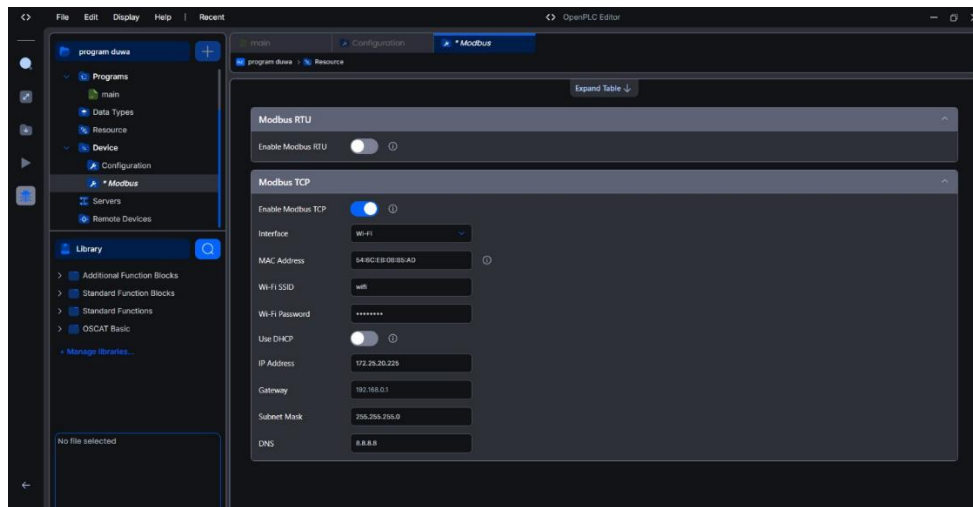
Secara keseluruhan, implementasi perangkat keras ini telah memenuhi rancangan topologi fisik yang diharapkan. Perakitan struktur mekanik yang kokoh serta penempatan komponen elektrikal di dalam *box panel* terpisah memastikan sistem dapat beroperasi secara aman dan terhindar dari risiko korsleting akibat paparan fluida. Selain itu, tata letak antarmuka HMI yang terpusat dan ergonomis sangat menunjang efektivitas interaksi operator. Dengan selesainya integrasi perangkat keras fisik beserta seluruh jalur instrumentasinya, alat simulator ini telah siap untuk dikonfigurasi dan diintegrasikan dengan logika algoritma kontrol pada tahap implementasi perangkat lunak.

4.1.2 Implementasi Perangkat Lunak

Pengembangan perangkat lunak pada sistem ini dilakukan secara terdistribusi menggunakan tiga platform utama untuk mengonfigurasi layanan komunikasi *controller*, antarmuka visualisasi, dan *gateway* penerjemah protokol.

Untuk memungkinkan *gateway* Node-RED mengakses data dari sistem kendali logika, dilakukan konfigurasi layanan *Modbus TCP Server* pada perangkat lunak OpenPLC Editor. Konfigurasi ini bertujuan untuk membuka jalur komunikasi data

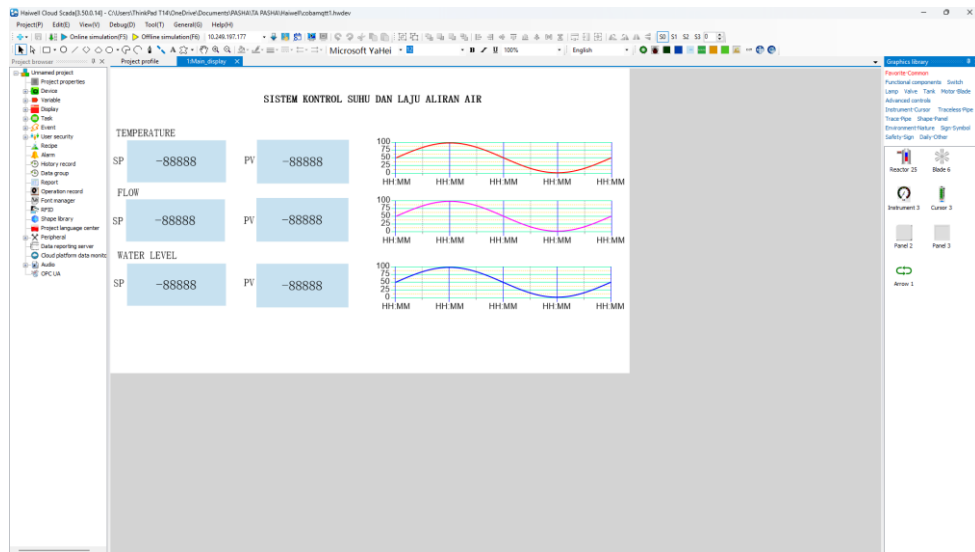
industri berbasis IP pada jaringan lokal. Tampilan jendela pengaturan Modbus TCP pada OpenPLC Editor ditunjukkan pada Gambar 4.4.



Gambar 4. 4 Konfigurasi layanan ModbusTCP server pada OpenPLC

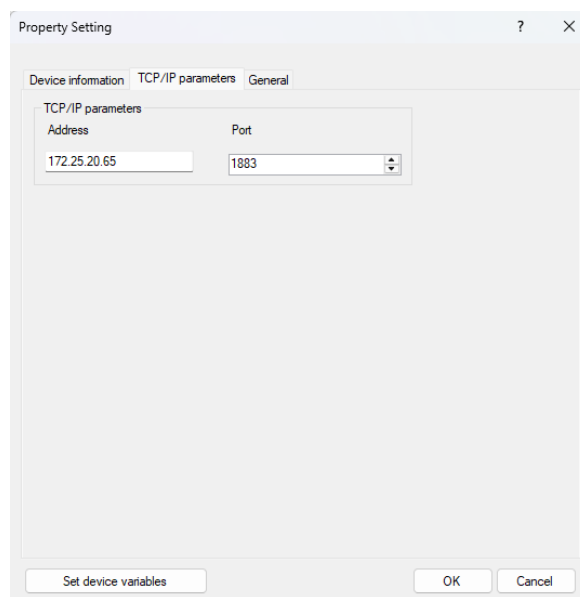
Gambar 4.4 menunjukkan pengaktifan fitur *Enable Modbus TCP* yang berjalan di atas antarmuka jaringan *Wi-Fi*. Pada pengaturan tersebut, perangkat OpenPLC dialokasikan pada alamat IP statis 172.25.20.225 dengan *Subnet Mask* 255.255.255.0 dan *Gateway* 192.168.0.1. Pengaturan IP statis ini sangat krusial agar *Node Modbus Read* dan *Modbus Write* pada *gateway* Node-RED dapat melakukan pendaftaran koneksi (*handshake*) secara konsisten ke alamat memori *holding register* OpenPLC tanpa terputus akibat perubahan IP dinamis.

Desain antarmuka dikembangkan menggunakan *software* Haiwell Cloud SCADA dan HMI Haiwell untuk memberikan representasi visual yang komprehensif kepada operator. Antarmuka ini dirancang untuk memantau temperatur air, laju aliran air, dan ketinggian air secara *real-time*, sekaligus memfasilitasi intervensi nilai *Setpoint* (SP) dan pengamatan grafik tren (*Trend Graph*). Tampilan rancangan *dashboard* visual (*Main_display*) ditunjukkan pada Gambar 4.5.



Gambar 4. 5 Tampilan Antarmuka Visual (*Dashboard UI*) pada Haiwell Cloud SCADA

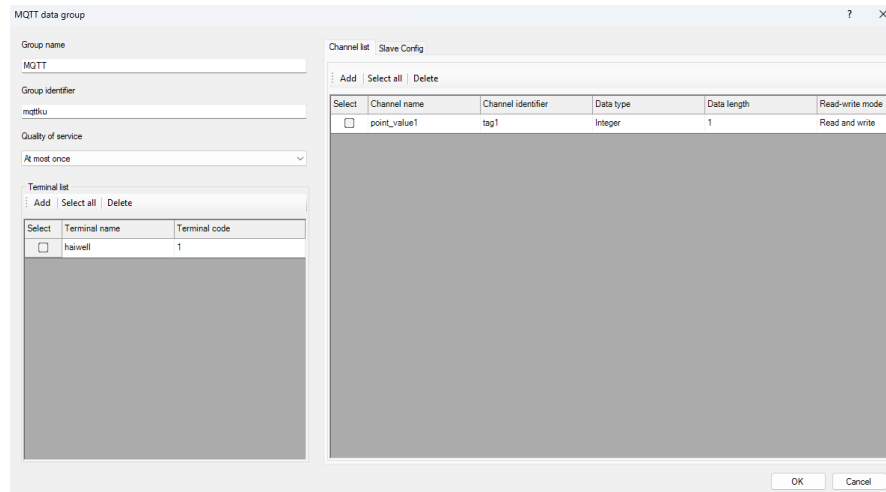
Untuk menghubungkan antarmuka visual tersebut dengan *gateway* Node-RED, dilakukan beberapa tahapan konfigurasi parameter komunikasi MQTT di dalam Haiwell Cloud SCADA:



Gambar 4. 6 Konfigurasi Parameter TCP/IP Device Driver MQTT pada Haiwell Cloud SCADA

Gambar 4.6 menunjukkan pengaturan parameter TCP/IP pada *device driver* MQTT di dalam aplikasi Haiwell Cloud SCADA. Konfigurasi ini berfungsi untuk membangun jalur komunikasi *socket* TCP antara instrumen HMI dengan *gateway*

utama. Parameter *Address* diisi dengan alamat IP 172.25.20.65 yang merupakan alamat IP lokal tempat *broker* MQTT dan *gateway* Node-RED beroperasi. Sementara itu, parameter *Port* disetel pada angka 1883 yang merupakan *port* standar internasional untuk transmisi data protokol MQTT.



Gambar 4. 7 Tampilan Pengaturan MQTT Data Group dan Channel List pada Haiwell Cloud SCADA

Konfigurasi *MQTT Data Group* pada Haiwell Cloud SCADA dilakukan dengan mendaftarkan empat grup utama. Gambar 4.7 menampilkan contoh pengaturan parameter *Group Identifier*, *Terminal List*, serta *Channel List* pada salah satu grup. Rincian konfigurasi parameter untuk seluruh grup komunikasi MQTT dirangkum pada Tabel 4.1.

Tabel 4. 1 Data Group dan Channel List pada Haiwell Cloud SCADA

No	Nama Group	Group Identifier	Terminal Name	Channel Name	Channel Identifier	Tipe Data	Mode Akses
1	MQTT	mqttku	haiwell	point_value1	tag1	Integer	Read and write
2	MQTT2	mqttku2	haiwell2	point_value2	tag2	Integer	Read and write
3	MQTT3	mqttku3	haiwell3	point_value3	tag3	Integer	Read and write

4	MQTT4	mqttku4	haiwell4	point_value4	tag4	Integer	Read and write
---	-------	---------	----------	--------------	------	---------	----------------

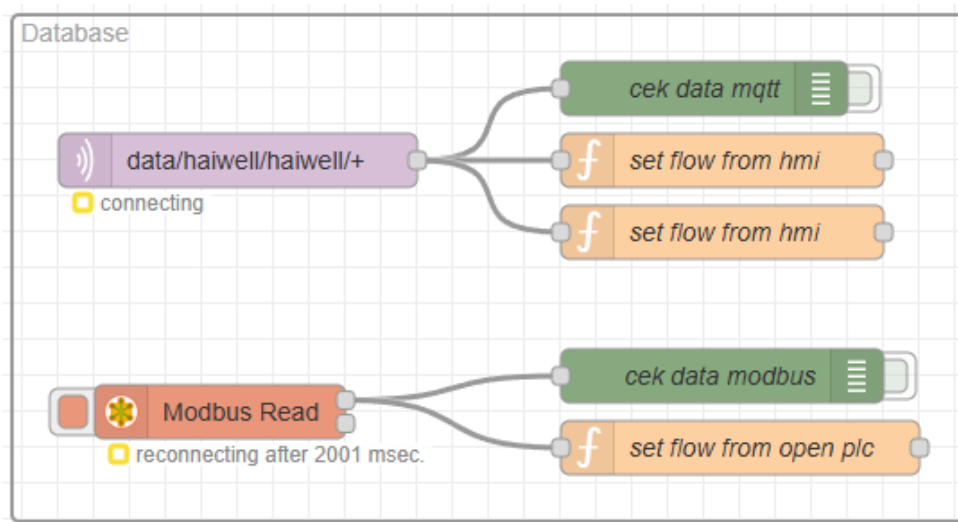
Berdasarkan Tabel 4.1, keempat grup komunikasi MQTT pada Haiwell Cloud SCADA terbagi ke dalam dua arah alur transmisi data, yaitu alur penerimaan data telemetri (*downstream*) dan alur pengiriman instruksi kendali (*upstream*). Grup 1 hingga Grup 3 (MQTT, MQTT2, MQTT3) dikonfigurasi untuk bertindak sebagai penerima data (*subscriber*) yang menarik pembaruan nilai variabel proses (PV) berupa laju aliran, suhu, dan ketinggian air dari *gateway* Node-RED secara *real-time*. Sementara itu, Grup 4 (MQTT4) difungsikan khusus pada jalur *upstream* untuk mengakomodasi seluruh intervensi operator—baik masukan angka *Setpoint* (SP) maupun penekanan tombol operasional—yang dipublikasikan (*publish*) kembali ke *gateway* Node-RED melalui topik terpusat *data/haiwell/haiwell/+*. Pemisahan grup ini bertujuan untuk menjaga efisiensi lalu lintas data nirkabel di jaringan lokal agar tidak terjadi penumpukan *payload* (*topic collision*) saat proses pemantauan dan pengontrolan berjalan bersamaan.

The screenshot shows a web-based configuration interface for an MQTT server. The title bar reads 'Edit server configuration'. The 'Server' dropdown is set to 'MQTT'. Under 'Server information configuration', the 'Server desc' is 'MQTT', 'Project identifier' is 'haiwell' (with a 'Random' button), 'Server host' is '172.25.20.65', and 'Port' is '1883'. The 'Client ID' is set to 'PN Code'. The 'Data push' section shows 'Data push network interface' set to 'WIFI'. There is an unchecked checkbox for 'Enable escalation verification'. At the bottom, the 'User info' section has empty fields for 'User name' and 'Password', with a 'Show' checkbox.

Gambar 4. 8 Konfigurasi *Server Host* MQTT Broker pada *Data Reporting Server* Haiwell

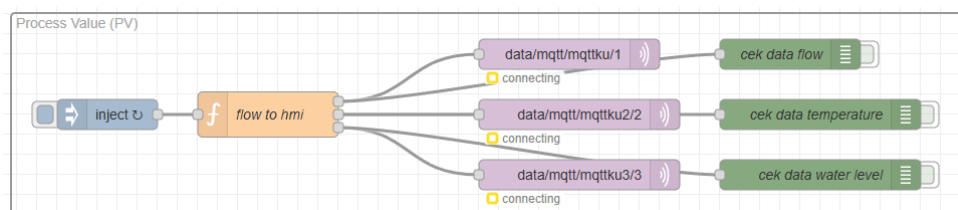
Gambar 4.8 menunjukkan konfigurasi *Server Host* pada menu *Data reporting server* yang difungsikan sebagai jembatan komunikasi utama antara Haiwell Cloud SCADA dengan *broker* MQTT pada *gateway* Node-RED. Parameter yang diatur mencakup *Server Host* dengan alamat IP 172.25.20.65 dan *Port* 1883. Selain itu, penetapan *Project Identifier* dengan nama haiwell secara otomatis membentuk basis struktur topik MQTT *upstream* (data/haiwell/haiwell/+) untuk menyalurkan instruksi dari HMI, serta pemilihan *Data Push Network Interface* berbasis WIFI untuk transmisi data nirkabel.

Untuk mengintegrasikan kedua perangkat lunak tersebut, platform Node-RED dikonfigurasi sebagai *gateway* komunikasi (*middleware*) penerjemah protokol antara Modbus TCP dan MQTT. *Workspace* Node-RED dirancang secara rinci ke dalam lima grup fungsional (*grouping*) utama:



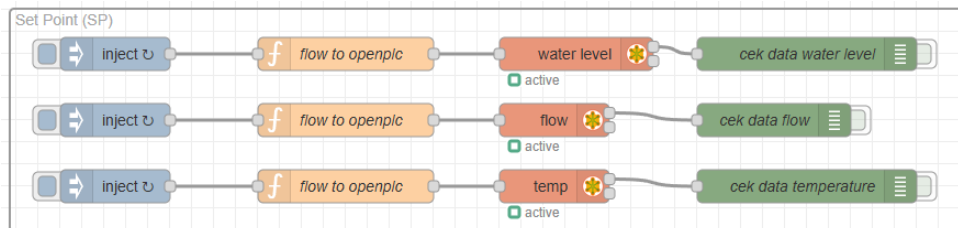
Gambar 4. 9 *Group Database*

Gambar 4.9 Mengelola penerjemahan protokol dua arah (*protocol converter*). *Node MQTT In* (data/haiwell/haiwell/+) dan *Function Node* set flow from hmi menangkap instruksi antarmuka, sedangkan *Node Modbus Read* dan *Function Node* set flow from open plc membaca *register* mentah dari OpenPLC



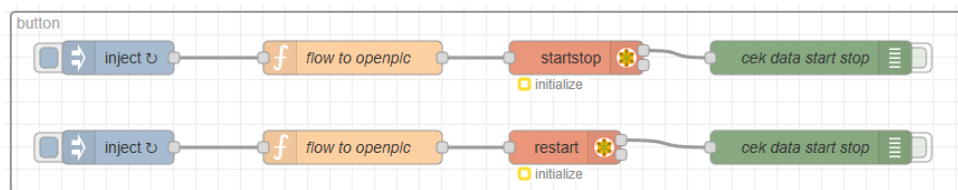
Gambar 4. 10 *Group Process Value (PV)*

Gambar 4.10 Menangani transmisi data telemetri sensor. *Node Inject* secara periodik memicu *Function Node* flow to hmi untuk memecah *payload*, yang kemudian dipublikasikan oleh *Node MQTT Out* ke topik data/mqtt/mqttku/1 (flow), data/mqtt/mqttku2/2 (suhu), dan data/mqtt/mqttku3/3 (level).



Gambar 4. 11 *Group Set Point (SP)*

Gambar 4.11 Meneruskan parameter nilai acuan dari operator. Menggunakan kombinasi *Node Inject*, *Function Node* flow to openplc, serta *Node Modbus Write* (water level, flow, dan temp) untuk menuliskan nilai target ke memori OpenPLC.



Gambar 4. 12 *Group Button*

Gambar 4.12 Memproses instruksi kontrol digital. Sinyal tombol dari HMI diterjemahkan melalui *Function Node* flow to openplc dan dieksekusi ke *coil/register* OpenPLC menggunakan *Node Modbus Write* (startstop dan restart).



Gambar 4. 13 *Group Utilitas (Waktu & Restart)*

Gambar 4.13 Menyediakan fungsi *time* untuk sinkronisasi stempel waktu (*timestamp*) sistem serta fungsi *restart* internal untuk memulihkan alur *gateway* secara otomatis apabila terjadi kendala koneksi.

4.1.3 Implementasi Keseluruhan Sistem

Integrasi keseluruhan sistem diuji dengan mengaktifkan seluruh komponen secara serentak untuk memverifikasi sinkronisasi antara perangkat keras dan

perangkat lunak secara *real-time*. Pengujian diawali dengan memasok daya utama 220V AC yang dikonversi oleh *Power Supply Unit* (PSU) dan *buck converter* untuk menghidupkan layar HMI (24V DC), aktuator pompa/Peltier (12V DC), serta mikrokontroler ESP32 dan instrumen sensor (5V DC). Dokumentasi pengujian keseluruhan sistem dalam kondisi beroperasi ditunjukkan pada Gambar 4.14.



Gambar 4. 14 Pengujian Implementasi Keseluruhan Sistem dalam Kondisi Beroperasi (*Power ON*)

Berdasarkan Gambar 4.14, mikrokontroler ESP32 bertugas mengakuisisi data mentah dari sensor suhu PT100, sensor laju aliran air YF-S201, serta sensor ketinggian air JSN-SR04T secara kontinu. Data variabel proses (*Process Value / PV*) tersebut diolah oleh OpenPLC dan ditransmisikan menuju *gateway* Node-RED menggunakan protokol Modbus TCP melalui jaringan lokal. Di dalam Node-RED, *payload* data diurai dan diterjemahkan menjadi format pesan MQTT, lalu dipublikasikan secara nirkabel menuju antarmuka HMI Haiwell B7H-W serta Haiwell Cloud SCADA. Tampilan visualisasi monitoring dan grafik tren saat sistem berjalan ditunjukkan pada Gambar 4.15.



Gambar 4. 15 Visualisasi Data Sensor dan Grafik Tren pada Layar HMI Saat Sistem Berjalan

Sebaliknya, alur kendali supervisi (*supervisory control*) diuji dengan mengintervensi nilai acuan (*Setpoint* / SP) serta penekanan tombol operasi (*Start/Stop* dan *Restart*) langsung dari layar sentuh HMI. Perintah berbasis MQTT dari HMI ditangkap oleh Node-RED, diterjemahkan menjadi perintah penulisan *register* (*Modbus Write*), dan disalurkan ke memori OpenPLC untuk menggerakkan daya pada SSR *heater*, *driver* MOSFET Peltier, maupun *driver* pompa BTS7960. Hasil implementasi menunjukkan bahwa pada kondisi operasi normal seluruh subsistem mampu melakukan pertukaran data dan mengeksekusi fungsi monitoring serta kendali supervisi dengan baik. Data variabel proses dapat diteruskan menuju HMI dan instruksi *Setpoint* dari operator dapat diterima kembali oleh sistem kontrol. Meskipun demikian, pada pengoperasian dalam waktu yang lebih panjang masih ditemukan *lag* atau *freeze* sementara pada antarmuka HMI. Temuan tersebut dibahas sebagai salah satu keterbatasan system pada Subbab 4.4.3

4.2 Pengujian Transmisi dan Kesesuaian Data Monitoring

Pengujian transmisi dan kesesuaian *data monitoring* dilakukan untuk mengetahui keberhasilan proses pengiriman data referensi (*Setpoint*) dari *Human Machine Interface* (HMI) menuju context data Node-RED. Pengujian ini bertujuan untuk memastikan bahwa nilai masukan yang diberikan oleh operator melalui HMI

dapat diterima oleh Node-RED dengan nilai yang sama tanpa mengalami perubahan nilai selama proses komunikasi berlangsung.

Pada sistem yang dirancang, HMI digunakan sebagai antarmuka operator untuk memberikan nilai referensi terhadap parameter proses yang akan dikendalikan. Nilai *Setpoint* yang diberikan melalui HMI selanjutnya dikirimkan menuju Node-RED dan disimpan pada context data sebagai data masukan sistem. Oleh karena itu, pengujian transmisi dilakukan dengan membandingkan nilai yang dikirimkan melalui HMI dengan nilai yang diterima pada context data Node-RED untuk mengetahui tingkat kesesuaian data selama proses komunikasi berlangsung.

Pengujian dilakukan dengan memberikan beberapa variasi nilai referensi pada masing-masing parameter yang digunakan dalam sistem, yaitu suhu, laju aliran air, dan ketinggian air. Setiap nilai yang dimasukkan melalui HMI kemudian diamati pada context data Node-RED untuk mengetahui apakah data yang diterima memiliki nilai yang sama dengan data yang dikirimkan.

Tingkat kesesuaian data transmisi dianalisis berdasarkan nilai selisih antara data yang dikirimkan dan data yang diterima. Nilai selisih tersebut digunakan untuk mengetahui apakah terjadi perubahan nilai selama proses transmisi data dari HMI menuju Node-RED. Apabila nilai yang diterima pada Node-RED sama dengan nilai yang diberikan melalui HMI, maka proses transmisi data dinyatakan berhasil.

Berdasarkan pengujian tersebut, proses transmisi data dilakukan pada tiga parameter masukan sistem, yaitu nilai referensi suhu, laju aliran air, dan ketinggian air. Masing-masing parameter diuji dengan memberikan variasi nilai *Setpoint* melalui HMI, kemudian dilakukan pengamatan terhadap nilai yang diterima dan tersimpan pada context data Node-RED. Hasil pengujian tersebut digunakan untuk mengetahui tingkat keberhasilan proses transmisi data antara HMI dan Node-RED pada setiap parameter yang diuji.

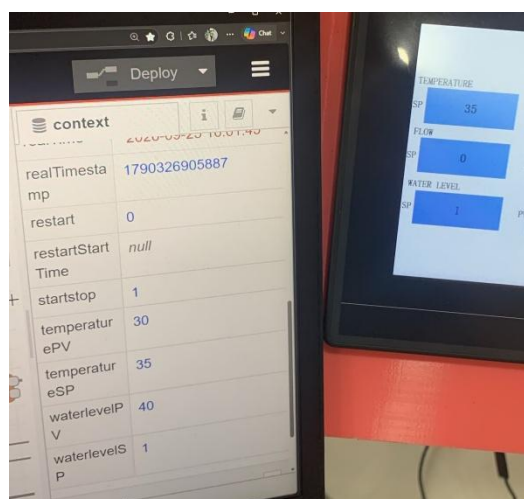
4.2.1 Pengujian Transmisi *Setpoint* Suhu

Pengujian transmisi setpoint suhu dilakukan untuk mengetahui tingkat kesesuaian nilai *Setpoint* suhu yang diberikan melalui *Human Machine Interface* (HMI) dengan nilai yang diterima dan tersimpan pada context data Node-RED. Pengujian ini bertujuan untuk memastikan bahwa proses pengiriman data referensi suhu dari HMI menuju Node-RED dapat berjalan dengan baik tanpa mengalami

perubahan nilai selama proses komunikasi berlangsung. Pengujian ini tidak melibatkan pembacaan sensor PT100, melainkan nilai referensi yang dimasukkan operator melalui HMI.

Pada pengujian ini, nilai referensi suhu (Temperature Setpoint) diberikan melalui HMI Haiwell dengan beberapa variasi nilai masukan. Nilai yang dikirimkan melalui HMI kemudian dibandingkan dengan nilai suhu yang diterima pada context data Node-RED. Perbandingan tersebut dilakukan untuk mengetahui apakah terdapat perbedaan nilai antara data yang dikirimkan oleh HMI dan data yang diterima oleh Node-RED.

Proses pengujian transmisi data suhu PT100 dilakukan dengan memberikan nilai *Setpoint* suhu sebesar 35°C, 45°C, dan 55°C melalui HMI. Tampilan nilai *Setpoint* pada HMI dan nilai yang tersimpan pada context data Node-RED digunakan sebagai bukti bahwa proses transmisi data telah berlangsung sesuai dengan rancangan sistem.



Gambar 4. 16 Transmisi Suhu

Berdasarkan Gambar 4.16, terlihat bahwa nilai *Temperature Setpoint* yang diberikan melalui HMI berhasil diterima oleh *context data* Node-RED. Pada gambar tersebut, nilai *Setpoint* suhu yang dimasukkan pada HMI sebesar 35°C terbaca pada variabel *temperatureSP* di Node-RED dengan nilai yang sama, yaitu 35. Hal ini menunjukkan bahwa proses transmisi data suhu dari HMI menuju Node-RED berhasil dilakukan tanpa perubahan nilai. Hasil pengujian transmisi data suhu PT100 pada beberapa variasi nilai *Setpoint* ditunjukkan pada Tabel 4.2.

Tabel 4. 2 Pengujian Transmisi Data Suhu PT100

No	Suhu Referensi HMI (°C)	Pembacaan Node-RED (°C)	Selisih Data (°C)	Sesuai
1	35°C	35°C	0°C	Sesuai
2	45°C	45°C	0°C	Sesuai
3	55°C	55°C	0°C	Sesuai

Berdasarkan hasil pengujian pada Tabel 4.2, terlihat bahwa nilai *Setpoint* suhu yang dikirimkan melalui HMI Haiwell dapat diterima oleh context data Node-RED dengan nilai yang sama pada seluruh variasi pengujian. Pengujian dilakukan dengan memberikan tiga variasi nilai referensi suhu, yaitu 35°C, 45°C, dan 55°C. Pada setiap pengujian, nilai yang terbaca pada Node-RED menunjukkan nilai yang identik dengan nilai yang diberikan melalui HMI.

Hasil perbandingan antara nilai referensi HMI dan nilai yang diterima oleh Node-RED menunjukkan bahwa seluruh pengujian menghasilkan nilai selisih data sebesar 0°C. Hal tersebut menunjukkan bahwa tidak terdapat perubahan nilai selama proses transmisi data berlangsung. Berdasarkan persamaan persentase kesalahan pada Bab II, nilai error transmisi adalah 0% dengan tingkat keberhasilan transmisi (success rate) sebesar $3/3 \times 100\% = 100\%$. Berdasarkan hasil tersebut, seluruh data pengujian dikategorikan **sesuai** karena nilai yang dikirimkan oleh HMI berhasil diterima oleh Node-RED tanpa mengalami perubahan.

Nilai selisih data yang diperoleh kemudian menunjukkan bahwa proses komunikasi transmisi data *Setpoint* suhu dari HMI menuju Node-RED berjalan dengan tingkat kesesuaian yang baik. Dengan demikian, sistem mampu melakukan pengiriman data referensi suhu secara akurat dan dapat digunakan sebagai masukan pada proses kendali sistem.

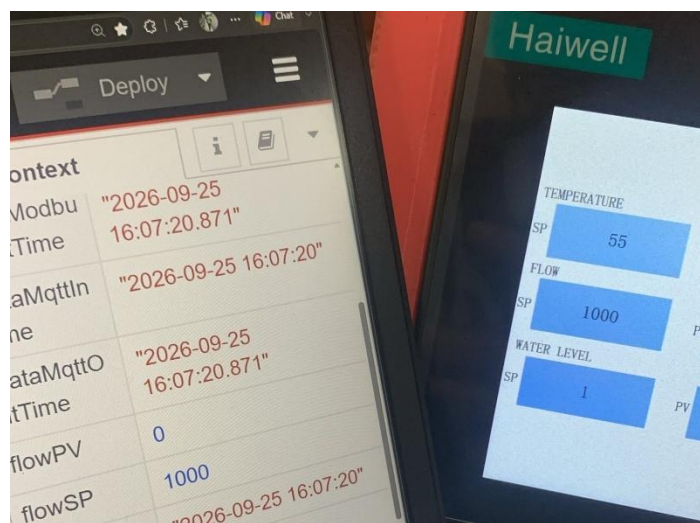
4.2.2 Pengujian Transmisi Laju Aliran Air YFS201

Pengujian transmisi *setpoint* laju aliran air dilakukan untuk mengetahui tingkat kesesuaian nilai referensi (*Setpoint*) laju aliran air yang diberikan melalui *Human*

Machine Interface (HMI) dengan nilai yang diterima pada context data Node-RED. Pengujian ini bertujuan untuk memastikan bahwa proses pengiriman data referensi laju aliran air dari HMI menuju Node-RED dapat berlangsung dengan baik tanpa mengalami perubahan nilai selama proses komunikasi data.

Pada pengujian ini, nilai referensi laju aliran air (*Flow Setpoint*) diberikan melalui HMI Haiwell dengan beberapa variasi nilai masukan. Nilai yang dikirimkan melalui HMI kemudian dibandingkan dengan nilai yang diterima dan tersimpan pada context data Node-RED. Perbandingan tersebut dilakukan untuk mengetahui apakah terdapat perbedaan antara nilai yang dimasukkan oleh operator pada HMI dengan nilai yang diterima oleh sistem.

Proses pengujian transmisi data laju aliran air dilakukan dengan memberikan variasi nilai referensi sebesar 1000 ml, 2000 ml, dan 3000 ml melalui HMI. Tampilan nilai referensi pada HMI dan nilai yang tersimpan pada context data Node-RED digunakan sebagai bukti bahwa proses transmisi data laju aliran air berhasil dilakukan.



Gambar 4. 17 Transmisi Laju Aliran

Berdasarkan Gambar 4.17, terlihat bahwa nilai referensi laju aliran air yang diberikan melalui HMI berhasil diterima oleh *context data* Node-RED. Nilai *Flow Setpoint* yang dikirimkan melalui HMI dapat terbaca pada Node-RED dengan nilai yang sama, sehingga menunjukkan bahwa proses transmisi data berjalan sesuai

dengan rancangan sistem. Hasil pengujian transmisi data laju aliran air YFS201 pada beberapa variasi nilai referensi ditunjukkan pada Tabel 4.3.

Tabel 4. 3 Pengujian Transmisi Laju Aliran Air YFS201

No	Nilai Referensi HMI (ml)	Pembacaan		Status
		Node-RED (ml)	Selisih (ml)	
1	1000	1000	0	Sukses
2	2000	2000	0	Sukses
3	3000	3000	0	Sukses

Berdasarkan hasil pengujian pada Tabel 4.3, nilai referensi laju aliran air yang dikirimkan melalui HMI berhasil diterima oleh Node-RED dengan nilai yang sama pada seluruh variasi pengujian. Pada nilai referensi 1000 ml, 2000 ml, dan 3000 ml, hasil pembacaan Node-RED menunjukkan nilai yang identik dengan nilai masukan dari HMI.

Hasil perhitungan selisih data menunjukkan bahwa seluruh pengujian memiliki nilai selisih sebesar 0 ml. Hal ini menunjukkan bahwa tidak terjadi perubahan nilai selama proses transmisi data berlangsung. Berdasarkan hasil tersebut, seluruh proses pengiriman data laju aliran air dari HMI menuju Node-RED dinyatakan berhasil dengan status sukses.

Dengan demikian, sistem mampu meneruskan nilai referensi laju aliran air secara akurat dari antarmuka HMI menuju Node-RED dan dapat digunakan sebagai data masukan pada proses pengendalian sistem.

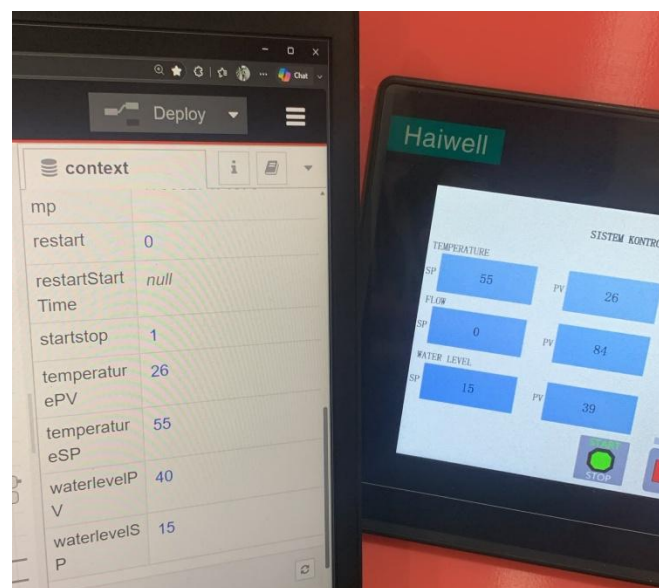
4.2.3 Pengujian Transmisi Ketinggian Air JSN-SR04T

Pengujian transmisi data ketinggian air JSN-SR04T dilakukan untuk mengetahui tingkat kesesuaian nilai referensi (*Setpoint*) ketinggian air yang diberikan melalui *Human Machine Interface* (HMI) dengan nilai yang diterima pada context data Node-RED. Pengujian ini bertujuan untuk memastikan bahwa proses pengiriman data referensi ketinggian air dari HMI menuju Node-RED dapat

berjalan dengan baik tanpa mengalami perubahan nilai selama proses komunikasi data berlangsung.

Pada pengujian ini, nilai referensi ketinggian air (*Water Level Setpoint*) diberikan melalui HMI Haiwell dengan beberapa variasi nilai masukan. Nilai yang dikirimkan melalui HMI kemudian dibandingkan dengan nilai yang diterima dan tersimpan pada context data Node-RED. Perbandingan tersebut dilakukan untuk mengetahui kesesuaian nilai antara data yang diberikan oleh operator melalui HMI dengan data yang diterima oleh sistem.

Proses pengujian transmisi data ketinggian air dilakukan dengan memberikan variasi nilai referensi sebesar 5 cm, 10 cm, dan 15 cm melalui HMI. Tampilan nilai referensi pada HMI dan nilai yang tersimpan pada context data Node-RED digunakan sebagai bukti bahwa proses transmisi data ketinggian air berhasil dilakukan.



Gambar 4. 18 Transmisi *Level Air*

Berdasarkan Gambar 4.18, terlihat bahwa nilai referensi ketinggian air yang diberikan melalui HMI berhasil diterima oleh context data Node-RED. Nilai *Water Level Setpoint* yang dikirimkan melalui HMI dapat terbaca pada Node-RED dengan nilai yang sama, sehingga menunjukkan bahwa proses transmisi data berjalan sesuai dengan rancangan sistem. Hasil pengujian transmisi data ketinggian air JSN-SR04T pada beberapa variasi nilai referensi ditunjukkan pada Tabel 4.4.

Tabel 4. 4 Pengujian Kesesuaian Data Pembacaan Sensor JSN-SR04T

No	Level Referensi HMI (Cm)	Pembacaan Node-RED (Cm)	Selisih (Cm)	Status
1	5	5	0	Sukses
2	10	10	0	Sukses
3	15	15	0	Sukses

Berdasarkan hasil pengujian pada Tabel 4.4, nilai referensi ketinggian air yang dikirimkan melalui HMI berhasil diterima oleh Node-RED dengan nilai yang sama pada seluruh variasi pengujian. Pada nilai referensi 5 cm, 10 cm, dan 15 cm, hasil pembacaan pada Node-RED menunjukkan nilai yang identik dengan nilai yang diberikan melalui HMI.

Hasil perhitungan selisih data menunjukkan bahwa seluruh pengujian memiliki nilai selisih sebesar 0 cm. Hal tersebut menunjukkan bahwa tidak terjadi perubahan nilai selama proses transmisi data berlangsung. Berdasarkan hasil tersebut, seluruh proses pengiriman data referensi ketinggian air dari HMI menuju Node-RED dinyatakan berhasil dengan status sukses, dengan nilai error 0% dan success rate $3/3 \times 100\% = 100\%$.

Dengan demikian, sistem mampu meneruskan nilai referensi ketinggian air secara akurat dari antarmuka HMI menuju Node-RED dan dapat digunakan sebagai data masukan pada proses monitoring maupun pengendalian sistem.

4.3 Pengujian Kinerja Komunikasi dan Antarmuka

Pengujian kinerja komunikasi dan antarmuka dilakukan untuk mengetahui kemampuan sistem dalam melakukan pertukaran data serta mempertahankan proses komunikasi antara perangkat yang terintegrasi pada sistem *monitoring* berbasis Node-RED. Pengujian ini bertujuan untuk mengevaluasi performa komunikasi yang digunakan pada sistem, terutama dari sisi waktu respons, jangkauan komunikasi, kestabilan koneksi, serta kemampuan antarmuka dalam menampilkan *data monitoring*.

Pada sistem yang dirancang, proses komunikasi melibatkan beberapa perangkat utama, yaitu *Human Machine Interface* (HMI), Node-RED sebagai *gateway* komunikasi, dan OpenPLC sebagai sistem kontrol. Setiap perangkat memiliki peran dalam proses pertukaran data sesuai dengan protokol komunikasi yang digunakan. Pengujian dilakukan untuk mengetahui karakteristik komunikasi data pada sistem sehingga dapat diketahui kemampuan sistem dalam mendukung proses *monitoring* dan kendali secara *real-time*.

Parameter pengujian pada bagian ini meliputi pengukuran waktu *respons* komunikasi (*latency*), pengujian jarak komunikasi pada jalur komunikasi yang digunakan, pengamatan tampilan antarmuka *monitoring* pada HMI, serta pengujian kemampuan sistem dalam mempertahankan dan memulihkan komunikasi ketika terjadi gangguan koneksi.

Hasil pengujian kinerja komunikasi dan antarmuka digunakan sebagai dasar evaluasi terhadap kemampuan sistem dalam melakukan pertukaran data secara stabil serta mendukung proses *monitoring* dan pengendalian pada sistem kontrol suhu dan laju aliran air yang telah dirancang.

4.3.1 Pengujian Latency Komunikasi MQTT

Pengujian latency komunikasi MQTT dilakukan untuk mengetahui waktu *respons* yang dibutuhkan dalam proses pengiriman data *monitoring* dari Node-RED menuju *Human Machine Interface* (HMI). Pengujian ini bertujuan untuk mengetahui kemampuan protokol MQTT dalam melakukan pertukaran data secara *real-time* pada sistem *monitoring* berbasis Node-RED yang telah dirancang.

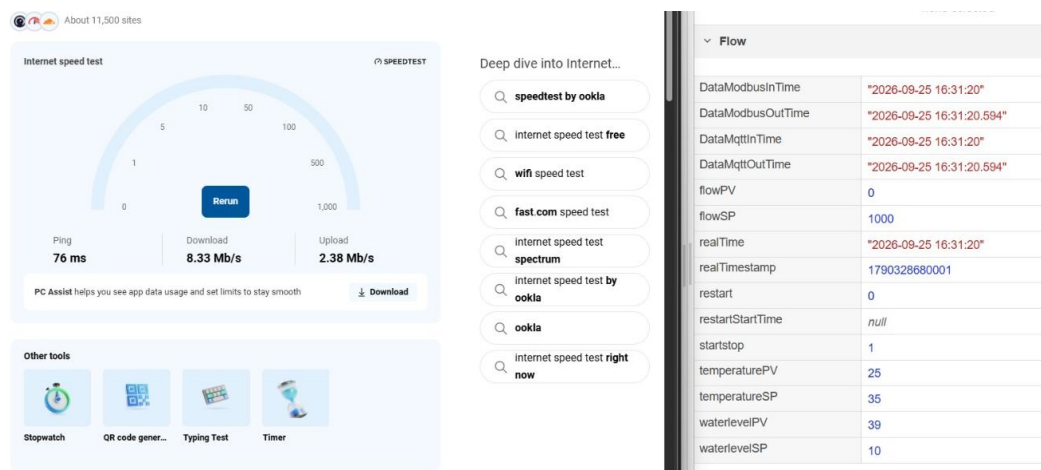
Pada sistem yang dikembangkan, Node-RED berperan sebagai perangkat pengirim (*publisher*) yang mengirimkan data *monitoring* melalui protokol MQTT, sedangkan HMI Haiwell berperan sebagai perangkat penerima (*subscriber*) yang menerima dan menampilkan data hasil komunikasi. Data yang dikirimkan melalui MQTT terdiri dari beberapa parameter proses yang dikirimkan secara bersamaan dalam satu siklus komunikasi.

Pengukuran latency dilakukan berdasarkan waktu yang diperlukan sejak data dikirimkan oleh Node-RED hingga data tersebut berhasil diterima oleh HMI. Pengujian tidak dilakukan berdasarkan masing-masing parameter proses, melainkan berdasarkan satu siklus pengiriman data secara keseluruhan. Hal tersebut

dilakukan karena komunikasi MQTT mengirimkan data dalam bentuk pesan (message) yang berisi kumpulan data sehingga nilai latency yang diperoleh merepresentasikan waktu respons komunikasi secara keseluruhan.

Pengujian dilakukan dengan memberikan variasi jarak komunikasi antara perangkat Node-RED dan HMI, yaitu 1 meter, 3 meter, 6 meter, 9 meter, 12 meter, 15 meter, 18 meter, dan 21 meter. Pada setiap variasi jarak dilakukan pencatatan waktu pengiriman data (timestamp kirim) dari Node-RED dan waktu penerimaan data (timestamp terima) pada HMI. Selisih waktu dari kedua proses tersebut digunakan sebagai nilai latency komunikasi MQTT.

Proses pencatatan waktu pengiriman dan penerimaan data dilakukan menggunakan variabel timestamp pada context data Node-RED. Data tersebut digunakan sebagai parameter pengukuran untuk mengetahui waktu tempuh proses komunikasi MQTT pada setiap variasi jarak pengujian.



Gambar 4. 19 Gambar Pengujian *Latency* Komunikasi MQTT Pada Node-RED

Berdasarkan Gambar 4.19, terlihat bahwa sistem mencatat waktu pengiriman dan penerimaan data MQTT melalui variabel `DataMqttInTime` dan `DataMqttOutTime` pada *context data* Node-RED. Variabel tersebut digunakan sebagai acuan dalam menentukan waktu respons komunikasi MQTT.

Pada pengujian yang ditampilkan pada Gambar 4.19, data MQTT tercatat masuk pada waktu 16:31:20 (`DataMqttInTime`) dan keluar pada waktu 16:31:20.594 (`DataMqttOutTime`). Berdasarkan selisih waktu antara proses pengiriman dan penerimaan data tersebut, diperoleh nilai *latency* komunikasi

sebesar 594 ms. Nilai tersebut menunjukkan waktu satu siklus pertukaran data MQTT yang tercatat pada Node-RED. Hasil pengujian *latency* komunikasi MQTT pada setiap variasi jarak komunikasi ditunjukkan pada Tabel 4.5.

Perlu diperhatikan bahwa stempel waktu data masuk (DataMqttInTime) hanya tercatat hingga satuan detik, sedangkan stempel waktu data keluar (DataMqttOutTime) tercatat hingga milidetik. Oleh karena itu, bagian milidetik dari waktu masuk tidak diketahui dan nilai *latency* sebenarnya berada pada rentang 0 ms hingga nilai yang tercatat, sehingga nilai *latency* pada Tabel 4.5 merupakan nilai pendekatan (batas atas). Selain itu, kedua stempel waktu dicatat pada Node-RED sehingga nilai tersebut belum mengukur secara langsung waktu tampilnya data pada layar HMI.

Tabel 4. 5 Pengujian *Latency* Komunikasi MQTT

No	Jarak Komunikasi (m)	<i>Timestamp</i> Kirim Node-RED	<i>Timestamp</i> Terima HMI	Latency (ms)	Status Komunikasi
1	1 meter	16:27:11.000	16:27:11.668	668ms	<i>Success</i>
2	3 meter	16:28:13.000	16:28:13.200	200ms	<i>Success</i>
3	6 meter	16:29:39.000	16:29:39.915	915ms	<i>Success</i>
4	9 meter	16:31:20.000	16:31:20.594	594ms	<i>Success</i>
5	12 meter	16:33:12.000	16:33:12.444	444ms	<i>Success</i>
6	15 meter	16:33:58.000	16:33:58.815	815ms	<i>Success</i>
7	18 meter	16:39:50.000	16:39:50.675	675ms	<i>Success</i>
8	21 meter	16:41:33.000	16:41:33.437	437ms	<i>Success</i>

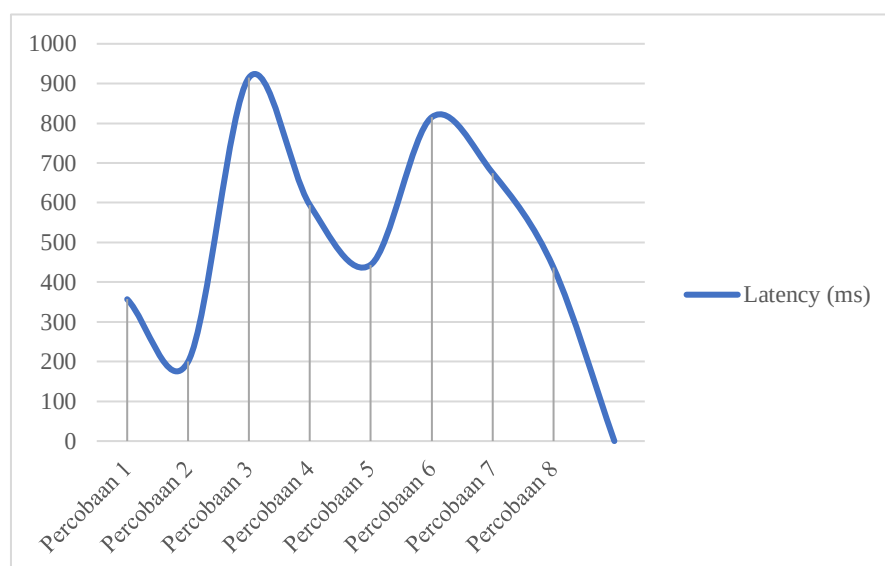
Berdasarkan hasil pengujian pada Tabel 4.5, komunikasi MQTT berhasil melakukan proses pengiriman data dari Node-RED menuju HMI pada seluruh variasi jarak pengujian. Seluruh pengujian menunjukkan status *Success*, yang

menunjukkan bahwa data berhasil diterima oleh HMI tanpa mengalami kegagalan komunikasi.

Nilai *latency* yang diperoleh pada setiap variasi jarak menunjukkan adanya perubahan waktu respons komunikasi selama proses pengujian berlangsung. Nilai *latency* terendah diperoleh pada jarak 3 meter dengan nilai sebesar 200 ms, sedangkan nilai *latency* tertinggi diperoleh pada jarak 6 meter dengan nilai sebesar 915 ms. Berdasarkan persamaan rata-rata *latency* pada Bab II, diperoleh nilai rata-rata sebesar $(668 + 200 + 915 + 594 + 444 + 815 + 675 + 437) / 8 = 593,5$ ms, dengan tingkat keberhasilan transmisi (success rate) sebesar $8/8 \times 100\% = 100\%$.

Perbedaan nilai *latency* pada setiap pengujian tidak menunjukkan pola peningkatan yang linier terhadap pertambahan jarak komunikasi. Hal tersebut menunjukkan bahwa selain faktor jarak, kondisi jaringan komunikasi dan proses pengiriman data pada saat pengujian juga dapat memengaruhi waktu respons komunikasi MQTT.

Untuk mengetahui perubahan nilai *latency* pada setiap variasi pengujian, data hasil pengukuran kemudian disajikan dalam bentuk grafik hubungan antara jarak komunikasi terhadap nilai *latency* seperti ditunjukkan pada Gambar 4.20.



Gambar 4. 20 Grafik Pengujian *Latency* MQTT

Berdasarkan Gambar 4.20, terlihat bahwa nilai *latency* komunikasi MQTT mengalami perubahan pada setiap variasi jarak pengujian. Grafik menunjukkan bahwa nilai *latency* tidak mengalami peningkatan secara linier seiring

bertambahnya jarak komunikasi. Pada beberapa variasi jarak, nilai *latency* mengalami kenaikan, namun pada jarak berikutnya dapat kembali mengalami penurunan. Hal ini menunjukkan bahwa perubahan jarak komunikasi tidak menjadi satu-satunya faktor yang memengaruhi waktu respons komunikasi MQTT.

Pada pengujian pertama dengan jarak komunikasi 1 meter, diperoleh nilai *latency* sebesar 668 ms. Ketika jarak komunikasi diperbesar menjadi 3 meter, nilai *latency* mengalami penurunan menjadi 200 ms. Selanjutnya, pada jarak 6 meter nilai *latency* meningkat secara signifikan menjadi 915 ms yang merupakan nilai tertinggi selama proses pengujian. Setelah itu, nilai *latency* kembali mengalami perubahan pada jarak 9 meter hingga 21 meter dengan rentang nilai antara 437 ms hingga 815 ms.

Nilai *latency* tertinggi diperoleh pada jarak 6 meter dengan nilai sebesar 915 ms, sedangkan nilai *latency* terendah diperoleh pada jarak 3 meter dengan nilai sebesar 200 ms. Perbedaan nilai tersebut menunjukkan bahwa waktu respons komunikasi dapat dipengaruhi oleh kondisi jaringan pada saat pengujian berlangsung, seperti kestabilan koneksi, proses pengiriman paket data, maupun beban komunikasi yang terjadi pada sistem.

Meskipun nilai *latency* mengalami variasi pada setiap jarak pengujian, seluruh proses komunikasi MQTT tetap menunjukkan status berhasil (*Success*). Hal tersebut menunjukkan bahwa peningkatan jarak komunikasi hingga 21 meter masih mampu dipertahankan oleh sistem tanpa menyebabkan kegagalan dalam proses transmisi data dari Node-RED menuju HMI.

Berdasarkan hasil pengujian tersebut, dapat diketahui bahwa protokol MQTT yang diterapkan pada sistem mampu mendukung proses transmisi data monitoring secara *real-time*. Variasi nilai *latency* yang terjadi tidak mengganggu proses penerimaan data pada HMI, sehingga komunikasi MQTT masih dapat digunakan sebagai media pertukaran data antara Node-RED dan HMI pada sistem monitoring yang dirancang.

4.3.2 Pengujian *Latency* Komunikasi Modbus TCP

Pengujian *latency* komunikasi Modbus TCP dilakukan untuk mengetahui waktu respons yang dibutuhkan dalam proses pertukaran data pada komunikasi Modbus TCP yang digunakan pada sistem. Pengujian ini bertujuan untuk

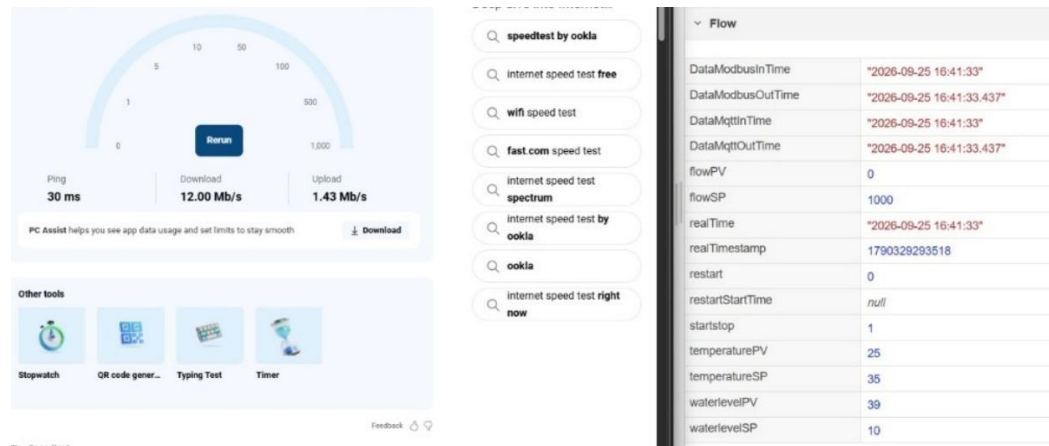
mengetahui performa komunikasi Modbus TCP dalam melakukan proses pengiriman dan penerimaan data antara perangkat yang terhubung pada sistem monitoring dan kontrol.

Pada sistem yang dirancang, komunikasi Modbus TCP digunakan sebagai salah satu jalur komunikasi data antara perangkat sistem dengan Node-RED sebagai gateway komunikasi. Node-RED berfungsi sebagai penghubung dalam proses pertukaran data sehingga informasi yang diterima dapat diproses dan digunakan oleh sistem sesuai dengan fungsi yang telah dirancang.

Pengukuran latency dilakukan berdasarkan waktu yang diperlukan dalam satu siklus komunikasi Modbus TCP. Nilai latency diperoleh dari selisih waktu antara proses pengiriman data dan proses penerimaan data pada sistem. Pengujian dilakukan berdasarkan satu siklus komunikasi, sehingga nilai yang diperoleh menunjukkan waktu respons keseluruhan komunikasi Modbus TCP, bukan berdasarkan masing-masing parameter data.

Pengujian dilakukan dengan memberikan variasi jarak komunikasi pada jalur Modbus TCP, yaitu 1 meter, 3 meter, 6 meter, 9 meter, 12 meter, 15 meter, 18 meter, dan 21 meter. Pada setiap variasi jarak dilakukan pencatatan waktu komunikasi menggunakan data timestamp untuk mengetahui perubahan nilai latency yang terjadi pada setiap kondisi pengujian.

Proses pencatatan waktu komunikasi dilakukan menggunakan data waktu yang tersimpan pada Node-RED. Data tersebut digunakan sebagai acuan untuk mengetahui waktu tempuh komunikasi Modbus TCP pada setiap variasi jarak pengujian.



Gambar 4. 21 Gambar Pengujian *Latency* Komunikasi Modbus TCP Pada Node-RED

Berdasarkan Gambar 4.21, terlihat bahwa sistem melakukan pencatatan waktu proses komunikasi Modbus TCP melalui data *timestamp* yang terdapat pada Node-RED. Data waktu tersebut digunakan sebagai acuan dalam menentukan nilai *latency* komunikasi berdasarkan selisih waktu antara proses pengiriman dan penerimaan data.

Pada pengujian yang ditampilkan pada Gambar 4.21, proses komunikasi tercatat dikirim pada waktu 16:41:33.000 dan diterima pada waktu 16:41:33.437. Berdasarkan perbedaan waktu antara kedua proses tersebut, diperoleh nilai *latency* komunikasi Modbus TCP sebesar 437 ms. Nilai tersebut menunjukkan waktu yang dibutuhkan sistem untuk menyelesaikan satu siklus proses pertukaran data pada kondisi pengujian tersebut.

Data hasil pencatatan *timestamp* pada setiap variasi jarak selanjutnya digunakan untuk memperoleh nilai *latency* komunikasi Modbus TCP. Hasil pengujian tersebut ditunjukkan pada Tabel 4.6.

Tabel 4. 6 Pengujian *Latency* Komunikasi Modbus TCP

No	Jarak Komunikasi	<i>Timestamp</i> Kirim Setpoint (HMI)	<i>Timestamp</i> Terima Node-RED	Latency (ms)	Status Komunikasi
1	1 meter	16:27:11.000	16:27:11.668	668ms	Success
2	3 meter	16:28:13.000	16:28:13.200	200ms	Success

3	6 meter	16:29:39.000	16:29:39.915	915ms	<i>Success</i>
4	9 meter	16:31:20.000	16:31:20.594	594ms	<i>Success</i>
5	12 meter	16:33:12.000	16:33:12.444	444ms	<i>Success</i>
6	15 meter	16:33:58.000	16:33:58.815	815ms	<i>Success</i>
7	18 meter	16:39:50.000	16:39:50.675	675ms	<i>Success</i>
8	21 meter	16:41:33.000	16:41:33.437	437ms	<i>Success</i>

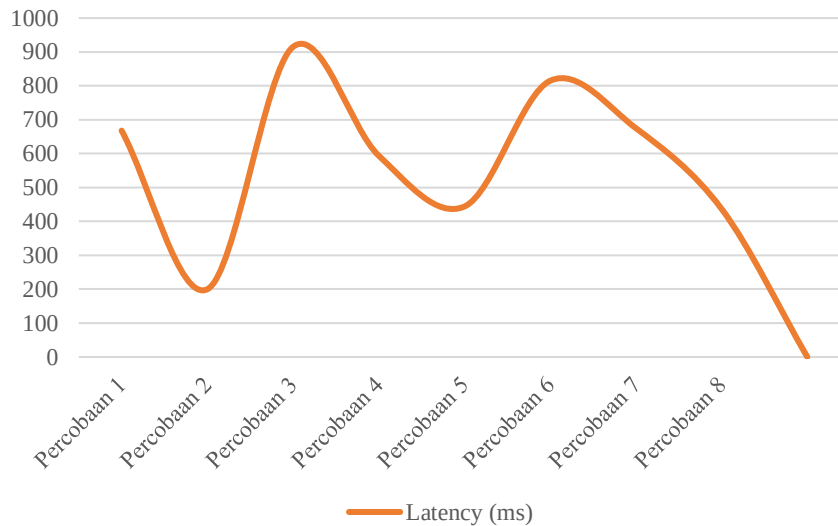
Berdasarkan hasil pengujian pada Tabel 4.6, komunikasi Modbus TCP berhasil melakukan proses pertukaran data pada seluruh variasi jarak pengujian. Seluruh pengujian menunjukkan status *Success*, yang menunjukkan bahwa komunikasi masih dapat berjalan dengan baik tanpa mengalami kegagalan transmisi data.

Apabila dibandingkan dengan Tabel 4.5, seluruh stempel waktu dan nilai latency pada Tabel 4.6 identik dengan hasil pengujian MQTT. Hal ini terjadi karena stempel waktu `DataModbusInTime/DataModbusOutTime` dan `DataMqttInTime/DataMqttOutTime` tercatat bersamaan dalam satu siklus flow pada Node-RED, sebagaimana terlihat pada Gambar 4.19 dan Gambar 4.21. Dengan demikian, nilai latency pada Tabel 4.5 dan Tabel 4.6 merepresentasikan waktu satu siklus pertukaran data pada gateway Node-RED secara keseluruhan dan belum dapat digunakan untuk membandingkan waktu respons protokol MQTT dan Modbus TCP secara terpisah.

Nilai *latency* yang diperoleh pada setiap variasi jarak menunjukkan adanya perubahan waktu respons komunikasi. Nilai *latency* terendah diperoleh pada jarak 3 meter dengan nilai sebesar 200 ms, sedangkan nilai *latency* tertinggi diperoleh pada jarak 6 meter dengan nilai sebesar 915 ms.

Perbedaan nilai *latency* pada setiap variasi jarak menunjukkan bahwa waktu respons komunikasi tidak selalu meningkat seiring bertambahnya jarak komunikasi. Hal tersebut menunjukkan bahwa terdapat faktor lain yang dapat memengaruhi waktu respons komunikasi, seperti kondisi jaringan, proses pengiriman paket data, dan aktivitas komunikasi yang berlangsung selama proses pengujian.

Untuk mengetahui perubahan nilai *latency* pada setiap variasi jarak komunikasi, data hasil pengujian kemudian disajikan dalam bentuk grafik hubungan antara jarak komunikasi terhadap nilai *latency* seperti ditunjukkan pada Gambar 4.22.



Gambar 4. 22 Grafik Pengujian Latency Modbus TCP

Berdasarkan Gambar 4.22, terlihat bahwa nilai *latency* komunikasi Modbus TCP mengalami perubahan pada setiap variasi jarak pengujian. Grafik menunjukkan bahwa peningkatan jarak komunikasi tidak secara langsung menyebabkan peningkatan nilai *latency* secara linier. Nilai *latency* mengalami perubahan yang fluktuatif pada setiap titik pengujian, dimana pada beberapa jarak tertentu terjadi peningkatan waktu respons, namun pada jarak berikutnya nilai *latency* dapat kembali mengalami penurunan.

Pada pengujian dengan jarak 1 meter, nilai *latency* yang diperoleh sebesar 668 ms. Ketika jarak komunikasi diperbesar menjadi 3 meter, nilai *latency* mengalami penurunan menjadi 200 ms. Namun pada jarak 6 meter, nilai *latency* meningkat hingga mencapai 915 ms yang menjadi nilai tertinggi selama pengujian. Setelah melewati jarak tersebut, nilai *latency* kembali mengalami perubahan dengan rentang nilai 437 ms hingga 815 ms pada variasi jarak berikutnya.

Nilai *latency* tertinggi terjadi pada jarak 6 meter dengan nilai sebesar 915 ms, sedangkan nilai terendah diperoleh pada jarak 3 meter dengan nilai sebesar 200 ms. Sementara itu, pada jarak terjauh yang diuji yaitu 21 meter, nilai *latency* yang

diperoleh sebesar 437 ms. Hasil tersebut menunjukkan bahwa meskipun jarak komunikasi semakin besar, sistem masih mampu mempertahankan waktu respons komunikasi yang relatif stabil dan tidak mengalami peningkatan *latency* secara signifikan.

Fluktuasi nilai *latency* yang terjadi pada pengujian dapat disebabkan oleh beberapa faktor, seperti kondisi jaringan komunikasi pada saat pengujian berlangsung, waktu pemrosesan data pada perangkat komunikasi, serta mekanisme pertukaran data yang terjadi pada protokol Modbus TCP. Oleh karena itu, perubahan nilai *latency* tidak hanya dipengaruhi oleh faktor jarak, tetapi juga dipengaruhi oleh kondisi operasional sistem saat proses pengujian dilakukan.

Meskipun terdapat variasi nilai *latency* pada setiap jarak pengujian, seluruh proses komunikasi Modbus TCP menunjukkan status **Success**. Hal tersebut menunjukkan bahwa komunikasi antara perangkat yang terhubung melalui Modbus TCP masih dapat berjalan dengan baik pada seluruh variasi jarak yang diuji tanpa mengalami kegagalan pertukaran data.

Berdasarkan hasil pengujian tersebut, dapat diketahui bahwa implementasi komunikasi Modbus TCP pada sistem yang dirancang mampu mempertahankan proses pertukaran data dengan baik. Kemampuan komunikasi tersebut mendukung fungsi Node-RED sebagai *gateway* dalam sistem *monitoring* dan kontrol, sehingga data dapat diterima dan diproses sesuai dengan kebutuhan sistem yang telah dirancang.

4.3.3 Pengujian Antarmuka Haiwell Cloud SCADA

Pengujian ini difokuskan pada evaluasi fungsionalitas antarmuka pengguna (*User Interface*) pada aplikasi Haiwell Cloud SCADA. Pengujian dilakukan untuk mengetahui kemampuan antarmuka dalam menampilkan data *monitoring*, memberikan masukan nilai *setpoint*, serta menampilkan perubahan kondisi sistem secara visual.

Antarmuka yang dirancang menampilkan beberapa parameter utama sistem, yaitu nilai *Setpoint* (SP), nilai proses (*Process Value*/PV), grafik perubahan variabel secara *real-time*, serta indikator status operasi sistem. Tampilan antarmuka Haiwell Cloud SCADA yang digunakan dalam proses pengujian ditunjukkan pada Gambar 4.22.



Gambar 4. 23 Tampilan Antarmuka Haiwell Cloud SCADA Saat *Monitoring* Sistem

Berdasarkan Gambar 4.23, antarmuka Haiwell Cloud SCADA mampu menampilkan nilai variabel proses berupa suhu, laju aliran air, dan ketinggian air dalam bentuk indikator numerik. Selain itu, grafik tren (*Trend Graph*) dapat menampilkan perubahan nilai proses selama sistem berjalan sehingga operator dapat melakukan pemantauan kondisi sistem secara langsung.

Pada antarmuka tersebut juga tersedia fitur pengaturan nilai *setpoint* yang memungkinkan operator memberikan nilai masukan untuk parameter sistem. Perubahan nilai masukan tersebut kemudian diteruskan melalui jalur komunikasi menuju Node-RED sebagai *gateway* sebelum diteruskan ke sistem kontrol.

Pengujian terhadap fungsi-fungsi antarmuka tersebut dilakukan dengan beberapa skenario pengujian yang meliputi tampilan indikator numerik, grafik tren (*Real-Time Trend Graph*), input parameter *setpoint*, indikator status sistem, dan sinkronisasi Cloud SCADA. Hasil pengujian fungsionalitas antarmuka ditunjukkan pada Tabel 4.7.

Tabel 4. 7 Pengujian Antarmuka Haiwell Cloud SCADA

No	Fitur Antarmuka	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengamatan Visual	Status
1	Indikator Numerik Suhu, <i>Flow</i> , & <i>Level</i>	Variabel proses fisik diubah pada siklus interval 1 detik	Angka numerik pada <i>dashboard</i> memperbaiki ui nilai PV	Tampilan angka memperbaiki ui nilai PV secara tepat pada setiap pergantian detik	Berhasil
			Angka Kurva grafik membentuk titik plot data baru setiap siklus interval	Angka Kurva Grafik membentuk garis tren kontinu dengan <i>update point</i> per detik	Berhasil
2	Grafik Tren (<i>Real-Time Trend Graph</i>)	Mengamati perubahan data kontinu selama rentang pengujian	Nilai SP terkirim ke <i>gateway</i> Node-RED secara instan	Register SP pada Node-RED berubah langsung saat tombol <i>Enter</i> ditekan	Berhasil
			Nilai SP terkirim ke <i>gateway</i> Node-RED secara instan	Register SP pada Node-RED berubah langsung saat tombol <i>Enter</i> ditekan	Berhasil
3	Input Parameter Setpoint (SP)	Memasukkan nilai batas suhu/level baru via <i>touchscreen</i> HMI	Indikator visual On / Off pada HMI	Perubahan indikator terjadi secara <i>real-</i>	Berhasil
			Indikator visual On / Off pada HMI	Perubahan indikator terjadi secara <i>real-</i>	Berhasil
4	Sistem On / Off	Mengubah status kerja	Indikator visual On / Off pada HMI	Perubahan indikator terjadi secara <i>real-</i>	Berhasil
			Indikator visual On / Off pada HMI	Perubahan indikator terjadi secara <i>real-</i>	Berhasil

		berubah	<i>time</i>	tanpa
		(Merah	=	<i>lag</i> visual
		OFF, Hijau		
		= ON)		
		Tampilan	Data	dan
		<i>dashboard</i>	tampilan	
		<i>remote</i>	grafik	pada
	Sinkronisasi	via Haiwell	aplikasi	Berha
5	Cloud	Cloud	seluler	sil
	SCADA	SCADA	identik	
		(Smartphone)	dengan	
		layar HMI		
		fisik	HMI fisik	

Berdasarkan hasil pengujian pada Tabel 4.7, antarmuka Haiwell Cloud SCADA telah berhasil menjalankan fungsi *monitoring* dan pengendalian sesuai dengan rancangan sistem. Pengujian dilakukan terhadap beberapa fitur utama antarmuka, yaitu tampilan indikator numerik, grafik tren perubahan data, input parameter *setpoint*, indikator status sistem, serta sinkronisasi tampilan melalui layanan Cloud SCADA.

Pada pengujian indikator numerik, nilai variabel proses yang diterima oleh antarmuka dapat ditampilkan secara visual sesuai dengan perubahan data yang terjadi pada sistem. Tampilan nilai pada indikator suhu, laju aliran air, dan ketinggian air mengalami pembaruan secara berkala sesuai interval pembacaan yang telah ditentukan.

Pengujian grafik tren menunjukkan bahwa Haiwell Cloud SCADA mampu menampilkan perubahan data proses dalam bentuk grafik secara kontinu. Data yang diterima dapat membentuk pola perubahan sesuai dengan nilai variabel yang dikirimkan, sehingga fitur grafik dapat digunakan sebagai media pemantauan kondisi proses secara *real-time*.

Pada pengujian *input* parameter *setpoint*, nilai batas baru yang dimasukkan melalui antarmuka HMI berhasil diteruskan melalui Node-RED menuju sistem kontrol. Perubahan nilai *setpoint* dapat diterima oleh *gateway* komunikasi dan

diteruskan sesuai jalur komunikasi yang dirancang tanpa mengalami perubahan data.

Selain itu, pengujian indikator status sistem menunjukkan bahwa perubahan kondisi operasi dapat ditampilkan melalui perubahan visual indikator pada antarmuka. Status sistem dapat berubah sesuai kondisi yang diberikan sehingga operator dapat mengetahui keadaan operasi melalui tampilan HMI.

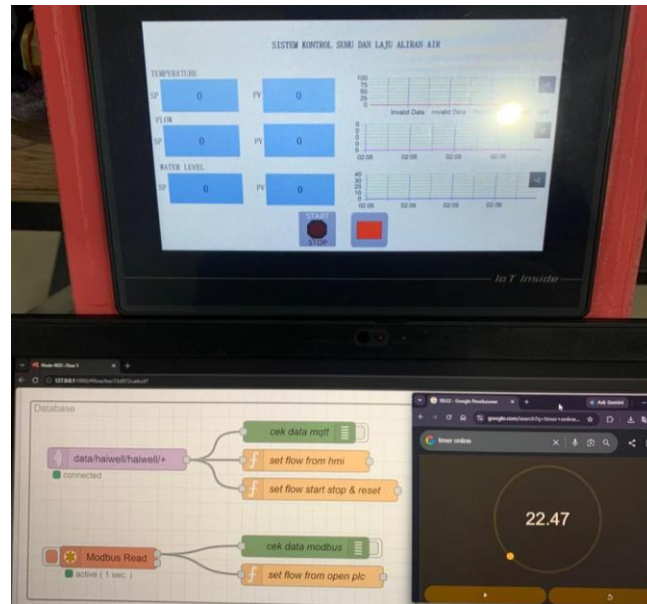
Pengujian terakhir dilakukan terhadap fitur sinkronisasi Cloud SCADA. Hasil pengujian menunjukkan bahwa tampilan antarmuka dapat terhubung dengan layanan Cloud SCADA dan menampilkan data monitoring secara konsisten. Dengan demikian, Haiwell Cloud SCADA dapat berfungsi sebagai antarmuka supervisi yang mendukung proses pemantauan variabel sistem dan pengiriman instruksi *setpoint* melalui Node-RED sebagai *gateway* komunikasi.

4.3.4 Pengujian Pemulihan Komunikasi Modbus TCP dan MQTT

Pengujian pemulihan komunikasi dilakukan untuk mengetahui kemampuan sistem dalam menangani gangguan koneksi jaringan serta mengevaluasi mekanisme pemulihan (*recovery*) pada jalur komunikasi MQTT dan Modbus TCP. Pengujian dilakukan dengan memberikan gangguan berupa pemutusan koneksi WiFi selama beberapa durasi tertentu, kemudian diamati respons sistem setelah koneksi jaringan kembali tersedia.

Parameter pengujian yang diamati meliputi respons antarmuka HMI selama gangguan berlangsung, kemampuan sistem melakukan koneksi ulang secara otomatis (*auto recovery*), waktu yang diperlukan hingga komunikasi kembali aktif, serta kondisi data yang ditampilkan setelah proses pemulihan komunikasi.

Proses pemulihan komunikasi diamati melalui perubahan status koneksi pada Node-RED dan pembaruan data pada antarmuka HMI. Dokumentasi kondisi sistem saat proses pemulihan komunikasi ditunjukkan pada Gambar 4.24.



Gambar 4. 24 Sistem *Auto-Recovery*

Berdasarkan Gambar 4.24, sistem menunjukkan proses pemulihan komunikasi setelah terjadi gangguan jaringan. Mekanisme pemulihan diamati melalui kembalinya koneksi komunikasi antara perangkat kontrol, Node-RED, dan antarmuka HMI setelah jaringan kembali tersedia. Hasil pengujian pemulihan komunikasi Modbus TCP dan MQTT ditunjukkan pada Tabel 4.8.

Tabel 4. 8 Pengujian Pemulihan Komunikasi Modbus TCP dan MQTT

No	Skenario Pengujian	Jalur komunikasi	Durasi Putus	Perilaku HMI	Status <i>Recovery</i>
1	WiFi Off (Uji 1)	MQTT	30s	HMI kehilangan pembaruan data selama koneksi terputus. Setelah jaringan kembali tersedia, MQTT melakukan <i>reconnect</i> otomatis, namun nilai	<i>Auto-Recovery</i> (± 24.88 detik) <i>with State Reset</i>

				variabel proses kembali ke kondisi awal (0) sebelum menerima pembaruan data berikutnya.
				Komunikasi Modbus TCP terputus selama gangguan jaringan.
2	WiFi Off (Uji 1)	Modbus TCP	30s	Setelah koneksi jaringan pulih, komunikasi antara Node-RED dan OpenPLC kembali tersambung sehingga proses pembacaan data dapat berjalan kembali.
				HMI kehilangan pembaruan data selama koneksi terputus. Setelah jaringan WiFi kembali tersedia, MQTT melakukan <i>reconnect</i> otomatis, namun nilai variabel proses kembali ke kondisi awal (0) sebelum
3	WiFi Off (Uji 2)	MQTT	60s	Setelah jaringan WiFi kembali tersedia, MQTT melakukan <i>reconnect</i> otomatis, namun nilai variabel proses kembali ke kondisi awal (0) sebelum

Auto-Recover red ($\pm$ 3.49 detik)

Auto-Recover red ($\pm$ 24.13 detik)

with State Reset

				menerima pembaruan data berikutnya.
				Komunikasi Modbus TCP terputus selama gangguan jaringan.
4	WiFi Off (Uji 2)	Modbus TCP	60s	Setelah koneksi jaringan pulih, komunikasi antara Node-RED dan OpenPLC kembali tersambung sehingga proses pembacaan data dapat berjalan kembali.
				HMI kehilangan pembaruan data selama koneksi terputus. Setelah jaringan WiFi kembali tersedia, MQTT melakukan <i>reconnect</i> otomatis, namun nilai variabel proses kembali ke kondisi awal (0) sebelum menerima pembaruan data berikutnya.
5	WiFi Off (Uji 3)	MQTT	120s	Setelah jaringan WiFi kembali tersedia, MQTT melakukan <i>reconnect</i> otomatis, namun nilai variabel proses kembali ke kondisi awal (0) sebelum menerima pembaruan data berikutnya.

Auto-Recover (± 3.57 detik)

Auto-Recover (± 22.47 detik) *with State Reset*

				Komunikasi
				Modbus TCP
				terputus selama
				gangguan jaringan.
				Setelah koneksi
				jaringan pulih, <i>Auto-</i>
				komunikasi antara <i>Recover</i>
6	WiFi Off (Uji 3)	Modbus TCP	120s	Node-RED dan <i>red</i> ($\pm$
				OpenPLC kembali 3.21
				tersambung detik)
				sehingga proses
				pembacaan data
				dapat berjalan
				kembali.

Berdasarkan hasil pengujian pada Tabel 4.8, gangguan koneksi WiFi memberikan pengaruh terhadap proses pertukaran data pada sistem komunikasi MQTT dan Modbus TCP. Pengujian dilakukan dengan memutus koneksi jaringan selama beberapa durasi untuk mengetahui kemampuan sistem dalam melakukan pemulihan komunikasi setelah jaringan kembali tersedia. Hasil pengamatan menunjukkan bahwa kedua jalur komunikasi mampu melakukan proses pemulihan secara otomatis, namun memiliki karakteristik waktu dan perilaku sistem yang berbeda.

Pada jalur komunikasi MQTT, ketika koneksi WiFi terputus, HMI mengalami kehilangan pembaruan data karena tidak lagi menerima data dari Node-RED melalui broker MQTT. Kondisi tersebut menyebabkan HMI tidak lagi menerima pembaruan nilai variabel proses dan menampilkan notifikasi bahwa perangkat MQTT tidak online (Lampiran 15). Setelah koneksi jaringan kembali tersedia, sistem MQTT mampu melakukan proses *reconnect* secara otomatis sehingga komunikasi antara HMI, broker MQTT, dan Node-RED dapat kembali berjalan.

Meskipun proses *reconnect* MQTT berhasil dilakukan, hasil pengamatan menunjukkan bahwa sistem belum mempertahankan kondisi terakhir (*state*) data pada antarmuka HMI. Setelah koneksi berhasil dipulihkan, nilai variabel proses dan

nilai setpoint pada tampilan HMI kembali ke kondisi awal yaitu 0 dan status tombol kendali kembali ke kondisi *OFF*. Data *monitoring* baru akan kembali diperbarui setelah Node-RED mengirimkan data terbaru melalui komunikasi MQTT. Hal tersebut menunjukkan bahwa mekanisme pemulihan MQTT telah berjalan pada sisi konektivitas, tetapi penyimpanan kondisi terakhir pada antarmuka belum dipertahankan ketika terjadi gangguan komunikasi.

Pada pengujian dengan durasi gangguan 30 detik, 60 detik, dan 120 detik, karakteristik pemulihan MQTT menunjukkan pola yang relatif sama. Perbedaan lama pemutusan jaringan tidak menyebabkan kegagalan proses *reconnect*, namun tetap menyebabkan hilangnya kondisi data terakhir pada HMI selama proses gangguan. Hal ini menunjukkan bahwa lama gangguan jaringan lebih berpengaruh terhadap waktu kehilangan pembaruan data dibandingkan terhadap kemampuan sistem dalam melakukan koneksi ulang.

Berbeda dengan jalur MQTT, komunikasi Modbus TCP menunjukkan respons pemulihan yang lebih cepat setelah koneksi jaringan kembali tersedia. Selama gangguan berlangsung, proses pembacaan dan pengiriman data antara Node-RED dan OpenPLC mengalami penghentian sementara karena jalur komunikasi TCP terputus. Namun, setelah jaringan kembali normal, koneksi Modbus TCP dapat terbentuk kembali secara otomatis sehingga Node-RED dapat kembali melakukan pembacaan *register* dan pengiriman data menuju sistem kontrol.

Berdasarkan variasi durasi gangguan pada pengujian Modbus TCP, yaitu 30 detik, 60 detik, dan 120 detik, sistem tetap mampu melakukan pemulihan komunikasi tanpa memerlukan konfigurasi ulang. Waktu pemulihan yang diperoleh relatif stabil pada setiap pengujian, menunjukkan bahwa mekanisme koneksi ulang pada jalur Modbus TCP mampu mengembalikan komunikasi antara *gateway* Node-RED dan perangkat kontrol secara konsisten.

Secara kuantitatif, waktu pemulihan jalur MQTT berada pada rentang 22,47–24,88 detik dengan rata-rata $\pm 23,83$ detik, sedangkan jalur Modbus TCP berada pada rentang 3,21–3,57 detik dengan rata-rata $\pm 3,42$ detik. Lama gangguan (30, 60, dan 120 detik) tidak menunjukkan pengaruh yang berarti terhadap waktu pemulihan pada kedua jalur komunikasi.

Perbedaan karakteristik pemulihan antara MQTT dan Modbus TCP dipengaruhi oleh fungsi masing-masing protokol dalam sistem. MQTT digunakan sebagai jalur komunikasi antara *gateway* dengan antarmuka HMI yang memiliki proses sinkronisasi *data* melalui *broker*, sehingga proses *reconnect* juga melibatkan pemulihan sesi komunikasi dan pembaruan data tampilan. Sedangkan Modbus TCP digunakan sebagai komunikasi langsung antara Node-RED dan OpenPLC untuk pertukaran *data register*, sehingga setelah koneksi TCP kembali tersedia, proses komunikasi dapat dilanjutkan kembali.

Secara keseluruhan, hasil pengujian menunjukkan bahwa sistem memiliki kemampuan pemulihan komunikasi secara otomatis pada kedua protokol yang digunakan. MQTT memiliki kemampuan *reconnect* yang baik pada sisi komunikasi HMI, namun belum mampu mempertahankan state terakhir ketika terjadi gangguan jaringan. Sementara itu, Modbus TCP menunjukkan proses pemulihan komunikasi yang lebih cepat dan mampu mengembalikan proses pertukaran data antara Node-RED dan OpenPLC secara otomatis.

4.3.5 Pengujian Keseluruhan Sistem

Pengujian keseluruhan sistem (End-to-End System Testing) dilakukan untuk mengevaluasi kemampuan integrasi seluruh komponen yang telah dirancang, mulai dari perangkat akuisisi data, sistem kontrol, *gateway* komunikasi, hingga antarmuka monitoring dan kendali. Pengujian ini bertujuan untuk memastikan bahwa ESP32, OpenPLC, Node-RED, HMI Haiwell, dan Haiwell Cloud SCADA dapat bekerja secara terpadu dalam menjalankan fungsi monitoring dan kendali supervisi.

Pengujian dilakukan dengan menjalankan sistem pada beberapa kondisi operasi, yaitu kondisi normal (*standby*), perubahan nilai *setpoint* melalui HMI, perubahan parameter proses, serta pengujian akses monitoring melalui layanan Cloud SCADA. Parameter yang diamati meliputi respons perangkat keras dalam membaca dan mengolah data, keberhasilan komunikasi antarperangkat, perubahan tampilan pada HMI, serta kesesuaian tampilan antara HMI fisik dengan dashboard Cloud SCADA. Dokumentasi kondisi HMI ketika seluruh sistem berjalan ditunjukkan pada Gambar 4.25.



Gambar 4. 25 Kondisi HMI Saat Keseluruhan Sistem Berjalan

Berdasarkan Gambar 4.25, antarmuka HMI mampu menampilkan parameter proses berupa suhu, laju aliran, dan *level* air secara real-time. Selain indikator numerik, sistem juga menampilkan grafik tren perubahan nilai proses sehingga operator dapat melakukan pemantauan kondisi sistem selama proses berlangsung. Hasil pengujian keseluruhan sistem ditunjukkan pada Tabel 4.9.

Tabel 4. 9 Pengujian Keseluruhan Sistem

No	Skenario Pengujian	Parameter Acuan	Resppon Perangkat Keras	Respon HMI & Cloud SCADA	Status
1	Skenario Normal (<i>Standby</i>)	Sistem aktif dengan nilai <i>monitoring</i> awal	ESP32 membaca data sensor, OpenPLC menjalankan program, dan Node-RED meneruskan data	Nilai suhu, <i>level</i> air, dan laju aliran tampil pada HMI serta grafik berjalan	Berhasil
2	Perubahan <i>Level</i> Air	<i>Setpoint level</i> air diberikan	Sistem menerima	Nilai <i>setpoint</i>	Berhasil

		sebesar 10 cm	nilai <i>setpoint</i>	10 cm	
		melalui HMI	dan perangkat menjalankan proses sesuai program kontrol	tampil pada HMI dan perubahan nilai proses dapat diamati melalui grafik	
3	Perubahan <i>Setpoint</i> Suhu	<i>Setpoint</i> suhu diubah melalui <i>input</i> HMI	Node-RED meneruskan data <i>setpoint</i> menuju OpenPLC melalui jalur komunikasi yang dirancang	Nilai <i>setpoint</i> baru tampil pada HMI dan <i>data monitorin</i> g diperbarui	Berhasil
4	Pengujian Perubahan Parameter	Perubahan nilai parameter melalui sistem kontrol	Perangkat membaca perubahan nilai proses sesuai kondisi pengujian	Perubahan nilai proses dapat diamati melalui tampilan numerik dan grafik trend pada HMI	Berhasil

5	Akses Cloud SCADA	Akses via <i>Smartphone</i> / Browser Web	Server	Tampilan	Berha sil
			Haiwell Cloud menarik <i>register</i> HMI fisik	<i>dashboard</i> pada <i>smartphon</i> <i>e</i> identik dengan HMI fisik	

Berdasarkan hasil pengujian pada Tabel 4.9, seluruh komponen sistem berhasil bekerja secara terintegrasi sesuai dengan rancangan yang telah dibuat. Pengujian ini menunjukkan bahwa komunikasi antara perangkat lapangan, sistem kontrol, *gateway* Node-RED, dan antarmuka HMI dapat berjalan dengan baik pada berbagai kondisi pengujian.

Pada skenario normal (*standby*), ESP32 berhasil melakukan pembacaan data sensor yang digunakan sebagai variabel proses, yaitu suhu, laju aliran, dan *level* air. Data hasil pembacaan sensor kemudian diproses oleh OpenPLC sesuai program kontrol yang telah dibuat. Selanjutnya, Node-RED berfungsi sebagai *gateway* komunikasi yang meneruskan data dari sistem kontrol menuju HMI melalui protokol komunikasi yang telah dirancang. Hasil pengamatan menunjukkan bahwa nilai variabel proses dapat ditampilkan pada HMI secara *real-time* serta perubahan data dapat divisualisasikan melalui grafik tren.

Pada pengujian perubahan nilai *setpoint level* air, nilai masukan sebesar 10 cm yang diberikan melalui HMI berhasil diteruskan menuju sistem kontrol. OpenPLC dapat menerima nilai *setpoint* tersebut dan menjalankan proses sesuai logika kontrol yang telah diprogram. Perubahan nilai proses kemudian dapat diamati kembali melalui tampilan numerik dan grafik pada HMI. Hal tersebut menunjukkan bahwa jalur komunikasi dua arah antara operator dan sistem kontrol telah berjalan dengan baik.

Pengujian perubahan *setpoint* suhu juga menunjukkan hasil yang serupa. Nilai *setpoint* yang diberikan oleh operator melalui HMI dapat diteruskan oleh Node-RED menuju OpenPLC tanpa mengalami kegagalan komunikasi. Setelah nilai diterima oleh sistem kontrol, perubahan parameter dapat ditampilkan kembali pada antarmuka HMI sehingga operator dapat mengetahui kondisi terbaru dari sistem.

Pada pengujian perubahan parameter proses, sistem mampu merespons perubahan nilai yang terjadi dan menampilkan informasi tersebut pada HMI. Tampilan numerik memberikan informasi nilai proses secara langsung, sedangkan grafik tren memberikan gambaran perubahan parameter selama sistem beroperasi. Kombinasi kedua tampilan tersebut mendukung fungsi *monitoring* sehingga operator dapat melakukan pengamatan terhadap kondisi sistem secara lebih mudah.

Selain *monitoring* melalui HMI fisik, pengujian akses Haiwell Cloud SCADA menunjukkan bahwa sistem mampu melakukan sinkronisasi data antara HMI lokal dengan perangkat eksternal melalui jaringan internet. Tampilan *dashboard* pada *smartphone* atau browser web memiliki informasi yang sama dengan tampilan HMI fisik, sehingga operator dapat melakukan pemantauan sistem dari lokasi yang berbeda.

Berdasarkan keseluruhan hasil pengujian, sistem HMI berbasis Node-RED yang dirancang telah mampu menjalankan fungsi *monitoring* dan kendali supervisi secara terintegrasi. Sistem berhasil melakukan pertukaran data dua arah, yaitu pengiriman data variabel proses dari perangkat kontrol menuju HMI serta pengiriman instruksi operator berupa nilai *setpoint* dan perintah kendali menuju sistem control.

Dengan demikian, hasil pengujian keseluruhan sistem menunjukkan bahwa rancangan integrasi ESP32, OpenPLC, Node-RED, HMI Haiwell, dan Haiwell Cloud SCADA telah memenuhi tujuan perancangan sebagai sistem *monitoring* dan kendali supervisi pada simulator kontrol proses.

4.4 Analisis Hasil Pengujian dan Pembahasan

Berdasarkan hasil implementasi dan pengujian yang telah dilakukan pada sistem monitoring berbasis Human Machine Interface (HMI) Haiwell, Node-RED, dan OpenPLC, selanjutnya dilakukan analisis terhadap hasil pengujian untuk mengevaluasi kinerja sistem secara menyeluruh. Tahap analisis ini dilakukan untuk mengetahui sejauh mana sistem yang dirancang mampu memenuhi fungsi utama sebagai sistem monitoring dan kontrol berbasis komunikasi jaringan.

Analisis hasil pengujian dilakukan dengan mempertimbangkan beberapa aspek utama, yaitu kemampuan sistem dalam melakukan transmisi data, kemampuan komunikasi antarperangkat dalam mempertahankan pertukaran informasi, serta

kemampuan antarmuka HMI dalam menampilkan dan menerima data dari sistem. Ketiga aspek tersebut saling berkaitan karena keberhasilan sistem monitoring tidak hanya ditentukan oleh kemampuan perangkat secara individu, tetapi juga oleh keberhasilan integrasi antarbagian sistem.

Pada sistem yang dikembangkan, HMI Haiwell digunakan sebagai media interaksi antara operator dengan sistem, sedangkan Node-RED berperan sebagai pusat pengelolaan komunikasi data antara HMI dan perangkat kontrol. Oleh karena itu, analisis dilakukan untuk mengetahui bagaimana setiap bagian sistem bekerja dalam mendukung proses monitoring parameter suhu, laju aliran air, dan ketinggian air.

Analisis pertama dilakukan terhadap kesesuaian transmisi data monitoring berdasarkan hasil pengujian pengiriman setpoint suhu, laju aliran air, dan ketinggian air. Pengujian tersebut digunakan untuk mengevaluasi kemampuan sistem dalam mempertahankan nilai data mulai dari proses pemberian nilai referensi melalui HMI hingga data diterima dan diproses pada Node-RED.

Selanjutnya, analisis kinerja komunikasi dan antarmuka dilakukan berdasarkan hasil pengujian komunikasi MQTT dan Modbus TCP, pengujian antarmuka Haiwell Cloud SCADA, pengujian pemulihan komunikasi, pengujian keseluruhan sistem, serta pengujian pengaruh jarak komunikasi terhadap proses transmisi data. Analisis pada bagian ini bertujuan untuk mengetahui kemampuan sistem dalam melakukan pertukaran data, memberikan respons komunikasi, serta mempertahankan fungsi monitoring ketika terjadi perubahan kondisi komunikasi.

Selain mengevaluasi keberhasilan sistem dalam melakukan proses komunikasi, analisis juga dilakukan untuk mengetahui faktor-faktor yang dapat memengaruhi kinerja sistem selama proses pengujian berlangsung. Faktor tersebut dapat berasal dari mekanisme komunikasi yang digunakan, kondisi jaringan, proses pengolahan data pada Node-RED, maupun kemampuan perangkat dalam melakukan pertukaran informasi.

Berdasarkan keseluruhan hasil pengujian, pembahasan pada subbab ini akan menjelaskan keterkaitan antara rancangan sistem dengan hasil implementasi yang diperoleh. Pembahasan tersebut mencakup tingkat keberhasilan transmisi data, performa komunikasi antarperangkat, kemampuan antarmuka dalam mendukung

proses monitoring, serta kelebihan dan keterbatasan sistem yang telah dikembangkan.

4.4.1 Analisis Kesesuaian Transmisi Data *Monitoring*

Berdasarkan hasil pengujian transmisi data monitoring yang telah dilakukan, sistem menunjukkan kemampuan yang baik dalam mempertahankan kesesuaian nilai data antara perangkat pengirim dan perangkat penerima. Analisis kesesuaian transmisi dilakukan dengan membandingkan nilai yang diberikan melalui Human Machine Interface (HMI) dengan nilai yang diterima dan tersimpan pada context data Node-RED. Perbandingan tersebut dilakukan untuk mengetahui apakah terdapat perubahan nilai selama proses pengiriman data berlangsung.

Pada sistem yang dirancang, HMI berperan sebagai antarmuka bagi operator untuk memberikan nilai referensi (Setpoint) terhadap parameter proses, sedangkan Node-RED berfungsi sebagai penerima sekaligus pengelola data sebelum diteruskan ke bagian sistem lainnya. Oleh karena itu, kesesuaian nilai antara HMI dan Node-RED menjadi salah satu parameter penting untuk memastikan bahwa informasi yang diberikan oleh operator dapat diteruskan secara akurat tanpa mengalami perubahan selama proses komunikasi.

Berdasarkan hasil pengujian transmisi setpoint suhu, nilai referensi suhu yang diberikan melalui HMI pada beberapa variasi pengujian berhasil diterima oleh Node-RED dengan nilai yang sama. Tidak ditemukan adanya perbedaan antara nilai yang dimasukkan melalui HMI dengan nilai yang diterima pada context data Node-RED. Hal tersebut menunjukkan bahwa proses transmisi data suhu mampu mempertahankan nilai informasi yang dikirimkan sehingga tidak terjadi perubahan data selama proses komunikasi berlangsung.

Hasil yang sama juga ditunjukkan pada pengujian transmisi setpoint laju aliran air. Nilai referensi laju aliran air yang diberikan melalui HMI dapat diterima oleh Node-RED sesuai dengan nilai masukan awal. Tidak adanya perbedaan nilai antara data pengirim dan penerima menunjukkan bahwa komunikasi data pada parameter laju aliran air berjalan dengan baik serta mampu menjaga keutuhan informasi selama proses transmisi.

Pada pengujian transmisi setpoint ketinggian air, sistem juga menunjukkan hasil yang serupa. Data yang dikirimkan melalui HMI berhasil diterima pada Node-RED

tanpa mengalami perubahan nilai. Kondisi tersebut menunjukkan bahwa sistem komunikasi yang digunakan mampu mempertahankan informasi data meskipun terdapat proses pengiriman dan penerimaan data antarperangkat.

Berdasarkan ketiga parameter pengujian tersebut, yaitu suhu PT100, laju aliran air, dan ketinggian air, dapat diketahui bahwa proses transmisi data dari HMI menuju Node-RED memiliki tingkat kesesuaian yang baik. Seluruh data yang dikirimkan (9 dari 9 data) dapat diterima dengan nilai yang sama (error 0%, success rate 100%), sehingga tidak ditemukan adanya penyimpangan nilai selama proses komunikasi berlangsung.

Perlu dicatat bahwa pengujian ini hanya mencakup arah transmisi setpoint dari HMI menuju Node-RED. Transmisi data variabel proses (PV) dari sensor menuju HMI belum diuji secara kuantitatif dan diverifikasi secara fungsional pada pengujian keseluruhan sistem (Tabel 4.9).

Keberhasilan transmisi data tersebut menunjukkan bahwa mekanisme komunikasi yang diterapkan pada sistem mampu menjaga integritas data (data integrity). Dalam sistem monitoring dan kontrol, integritas data menjadi faktor penting karena perubahan nilai sekecil apa pun pada proses transmisi dapat menyebabkan perbedaan antara kondisi yang diamati operator dengan kondisi sebenarnya pada sistem.

Selain itu, hasil pengujian ini juga menunjukkan bahwa Node-RED mampu menjalankan perannya sebagai gateway komunikasi dengan baik. Node-RED tidak hanya menerima data dari HMI, tetapi juga mampu mempertahankan informasi yang diterima sebelum digunakan untuk proses komunikasi berikutnya. Dengan demikian, proses pertukaran data antara antarmuka pengguna dan sistem kontrol dapat berlangsung secara akurat.

Secara keseluruhan, hasil analisis menunjukkan bahwa sistem monitoring yang dirancang telah mampu melakukan transmisi data dengan tingkat kesesuaian yang baik antara HMI dan Node-RED. Keberhasilan tersebut menjadi dasar bahwa sistem telah memenuhi kebutuhan awal dalam proses *monitoring*, yaitu memastikan data yang diberikan oleh operator dapat diterima dan diproses oleh sistem tanpa mengalami perubahan nilai.

4.4.2 Analisis Kinerja HMI dan Keandalan Komunikasi Data

Berdasarkan hasil pengujian yang telah dilakukan, sistem *Human Machine Interface* (HMI) berbasis Node-RED yang dikembangkan mampu menjalankan fungsi *monitoring* dan supervisi terhadap sistem kontrol suhu dan laju aliran air secara terintegrasi. Sistem yang dirancang menggunakan arsitektur komunikasi *hybrid* dengan Node-RED sebagai *gateway* komunikasi yang menghubungkan protokol Modbus TCP pada sisi sistem kontrol dengan protokol MQTT pada sisi antarmuka HMI. Pada arsitektur tersebut, data variabel proses dari OpenPLC diteruskan menuju Node-RED melalui komunikasi Modbus TCP, kemudian dikonversikan menjadi data MQTT untuk dikirimkan menuju HMI Haiwell. Arsitektur ini memungkinkan proses pertukaran data antara perangkat kontrol dan antarmuka *monitoring* dapat dilakukan tanpa ketergantungan langsung antarprotokol komunikasi yang berbeda.

Kinerja sistem komunikasi dianalisis berdasarkan beberapa parameter, yaitu kemampuan transmisi data, waktu *respons* komunikasi (*latency*), kestabilan pertukaran data, serta kemampuan sistem dalam mempertahankan koneksi ketika terjadi gangguan komunikasi. Berdasarkan hasil pengujian *latency* komunikasi MQTT dan Modbus TCP, seluruh proses pengiriman data dapat berjalan dengan status berhasil (Success), dengan success rate 100% dan *latency* tercatat pada rentang 200–915 ms (rata-rata 593,5 ms). Hal tersebut menunjukkan bahwa Node-RED mampu menjalankan fungsi sebagai *protocol converter* yang membaca data dari sistem kontrol dan meneruskannya menuju HMI dengan baik.

Pada pengujian komunikasi MQTT, nilai *latency* yang diperoleh menunjukkan adanya perubahan waktu *respons* pada setiap pengujian. Perubahan nilai tersebut menunjukkan bahwa waktu pengiriman data tidak selalu konstan meskipun sistem menggunakan jaringan komunikasi yang sama. Kondisi ini dapat terjadi karena proses komunikasi tidak hanya dipengaruhi oleh jarak perangkat, tetapi merupakan akumulasi dari beberapa proses, mulai dari pembentukan paket data, proses transmisi melalui jaringan Wi-Fi, penerimaan data oleh perangkat tujuan, hingga proses pengolahan data pada Node-RED sebelum ditampilkan pada HMI.

Berdasarkan hasil dokumentasi pengujian, kondisi jaringan selama proses pengambilan data juga mengalami perubahan nilai kualitas koneksi. Perubahan

nilai ping, kecepatan unggah (*upload*), dan kecepatan unduh (*download*) menunjukkan bahwa jaringan Wi-Fi yang digunakan memiliki kondisi yang dinamis selama proses pengujian. Namun, perubahan kondisi jaringan tersebut tidak dapat dijadikan sebagai satu-satunya faktor penyebab perubahan nilai *latency*. Hal ini dikarenakan waktu *respons* sistem juga dipengaruhi oleh proses *internal* perangkat, seperti pembacaan *register* Modbus TCP pada Node-RED, proses konversi data menjadi format MQTT, serta mekanisme penerimaan pesan pada HMI Haiwell.

Pada beberapa pengujian, nilai *latency* yang diperoleh cukup besar, namun komunikasi tetap berjalan dengan status berhasil. Kondisi tersebut menunjukkan bahwa sistem memiliki kemampuan dalam mempertahankan proses pertukaran data meskipun terdapat variasi waktu *respons*. Dalam implementasi sistem *monitoring*, kestabilan komunikasi tidak hanya ditentukan berdasarkan nilai *latency* terkecil, tetapi juga berdasarkan kemampuan sistem dalam memastikan data tetap diterima oleh perangkat tujuan tanpa mengalami kehilangan komunikasi.

Selain faktor jaringan, karakteristik masing-masing protokol komunikasi juga berpengaruh terhadap performa sistem. Modbus TCP digunakan pada sisi komunikasi kontrol antara OpenPLC dan Node-RED, sehingga memiliki fungsi utama dalam pertukaran *data register* sistem. Sementara itu, MQTT digunakan pada sisi *monitoring* karena memiliki mekanisme komunikasi berbasis *publish-subscribe* yang sesuai untuk pengiriman data menuju perangkat antarmuka. Perbedaan fungsi tersebut menyebabkan kedua protokol memiliki karakteristik waktu *respons* yang berbeda, meskipun keduanya berada dalam satu sistem yang sama.

Pada sisi antarmuka, hasil pengujian HMI Haiwell Cloud SCADA menunjukkan bahwa sistem mampu menampilkan data parameter proses secara *real-time*. Data suhu, laju aliran air, dan ketinggian air dapat divisualisasikan melalui tampilan *monitoring* sehingga operator dapat mengetahui kondisi aktual sistem tanpa melakukan pengamatan langsung pada perangkat kontrol. Selain menampilkan data proses, HMI juga mampu menerima masukan nilai referensi (*Setpoint*) dari operator yang kemudian diteruskan kembali menuju sistem kontrol melalui jalur komunikasi yang telah dirancang. Hal ini menunjukkan bahwa HMI

tidak hanya berfungsi sebagai media visualisasi, tetapi juga sebagai media supervisi yang memungkinkan operator melakukan interaksi terhadap sistem.

Kemampuan HMI dalam mempertahankan tampilan data selama sistem berjalan menunjukkan bahwa proses integrasi antara perangkat kontrol, *gateway* komunikasi, dan antarmuka telah berjalan dengan baik. Node-RED berperan sebagai komponen penting dalam menjaga aliran data dua arah, yaitu meneruskan data pembacaan sensor menuju HMI serta meneruskan perintah operator dari HMI menuju sistem kontrol. Dengan adanya fungsi tersebut, sistem yang dikembangkan mampu menggabungkan proses *monitoring* dan supervisi dalam satu antarmuka terpusat.

Berdasarkan pengujian pemulihan komunikasi Modbus TCP dan MQTT, sistem juga menunjukkan kemampuan untuk melakukan koneksi kembali setelah terjadi gangguan komunikasi. Mekanisme pemulihan tersebut menunjukkan bahwa gangguan sementara pada jaringan tidak menyebabkan sistem berhenti secara permanen. Setelah koneksi kembali tersedia, proses pertukaran data dapat dilanjutkan sehingga fungsi *monitoring* dapat kembali berjalan. Kemampuan pemulihan ini menjadi salah satu aspek penting dalam sistem berbasis jaringan karena gangguan komunikasi merupakan salah satu faktor yang dapat memengaruhi kontinuitas proses *monitoring*.

Meskipun sistem telah mampu menjalankan fungsi *monitoring* dan komunikasi dengan baik, masih terdapat beberapa keterbatasan yang dapat memengaruhi performa sistem. Salah satunya adalah ketergantungan terhadap kualitas jaringan Wi-Fi yang digunakan. Perubahan kondisi jaringan dapat menyebabkan variasi waktu *respons* komunikasi sehingga nilai *latency* tidak selalu berada pada nilai yang sama. Selain itu, karena sistem masih berupa purwarupa skala laboratorium, hasil pengujian belum sepenuhnya menggambarkan kondisi komunikasi pada lingkungan industri dengan jumlah perangkat yang lebih banyak dan tingkat gangguan jaringan yang lebih kompleks.

Secara keseluruhan, hasil analisis menunjukkan bahwa integrasi HMI Haiwell, Node-RED, OpenPLC, Modbus TCP, dan MQTT mampu membangun sistem *monitoring* dan supervisi yang berjalan secara fungsional. Sistem mampu melakukan pertukaran data, menampilkan parameter proses secara *real-time*,

menerima masukan operator, serta mempertahankan komunikasi setelah terjadi gangguan. Dengan demikian, rancangan sistem yang dikembangkan telah memenuhi tujuan penelitian dalam membangun sistem *monitoring* berbasis HMI dengan komunikasi jaringan yang terintegrasi dan memiliki kemampuan adaptasi terhadap perubahan kondisi komunikasi.

4.4.3 Kelebihan dan Keterbatasan Sistem

Berdasarkan hasil implementasi dan pengujian yang telah dilakukan, sistem *monitoring* berbasis *Human Machine Interface* (HMI) Haiwell, Node-RED, dan OpenPLC memiliki beberapa kelebihan yang mendukung penerapan sistem *monitoring* dan kontrol berbasis komunikasi jaringan. Namun demikian, sistem yang dikembangkan masih memiliki beberapa keterbatasan yang perlu diperhatikan sebagai bahan evaluasi dan pengembangan lebih lanjut.

Salah satu kelebihan utama dari sistem yang dirancang adalah kemampuan integrasi beberapa perangkat dengan protokol komunikasi yang berbeda. Pada sistem ini, Node-RED berperan sebagai *gateway* komunikasi yang menghubungkan komunikasi Modbus TCP pada sisi perangkat kontrol dengan komunikasi MQTT pada sisi antarmuka *monitoring*. Dengan adanya mekanisme tersebut, perangkat kontrol dan perangkat *monitoring* dapat berkomunikasi meskipun menggunakan karakteristik protokol yang berbeda.

Kelebihan lainnya adalah kemampuan sistem dalam melakukan *monitoring* parameter proses secara terpusat melalui HMI Haiwell. Parameter seperti suhu, laju aliran air, dan ketinggian air dapat ditampilkan dalam bentuk visual sehingga operator dapat melakukan pemantauan kondisi sistem dengan lebih mudah. Selain fungsi *monitoring*, HMI juga memungkinkan operator memberikan nilai referensi (*Setpoint*) yang dapat diteruskan menuju sistem kontrol, sehingga sistem tidak hanya berfungsi sebagai media pengamatan tetapi juga mendukung proses supervisi.

Dari sisi komunikasi data, hasil pengujian menunjukkan bahwa sistem mampu mempertahankan proses transmisi data dengan tingkat kesesuaian yang baik. Pengujian transmisi data menunjukkan bahwa nilai yang diberikan melalui HMI dapat diterima pada Node-RED tanpa mengalami perubahan nilai. Selain itu, pengujian komunikasi MQTT dan Modbus TCP menunjukkan bahwa sistem

mampu melakukan pertukaran data dengan status berhasil pada seluruh kondisi pengujian yang dilakukan.

Penggunaan Node-RED juga memberikan fleksibilitas dalam pengembangan sistem karena proses komunikasi dan pengolahan data dapat dikonfigurasi melalui lingkungan pemrograman berbasis *flow*. Hal tersebut mempermudah proses integrasi perangkat, perubahan konfigurasi komunikasi, serta pengembangan fitur tambahan tanpa harus melakukan perubahan besar pada perangkat keras yang digunakan.

Meskipun memiliki beberapa kelebihan, sistem yang dikembangkan masih memiliki beberapa keterbatasan. Salah satu keterbatasan utama adalah ketergantungan terhadap kondisi jaringan komunikasi yang digunakan. Karena proses pertukaran data MQTT dan Modbus TCP dilakukan melalui jaringan berbasis Wi-Fi, perubahan kualitas koneksi dapat memengaruhi waktu *respons* komunikasi (*latency*). Hal tersebut terlihat dari adanya variasi nilai *latency* pada hasil pengujian, meskipun seluruh proses komunikasi masih dapat berjalan dengan baik.

Keterbatasan lainnya adalah sistem masih dikembangkan dalam skala purwarupa laboratorium sehingga jumlah perangkat dan kondisi pengujian masih terbatas. Pada implementasi industri dengan jumlah perangkat yang lebih banyak, terdapat kemungkinan peningkatan beban komunikasi yang dapat memengaruhi performa sistem. Oleh karena itu, diperlukan pengujian lanjutan dengan jumlah perangkat yang lebih besar serta kondisi jaringan yang lebih mendekati lingkungan industri.

Selain itu, sistem yang dikembangkan belum dilengkapi dengan penyimpanan data historis (*historical data logging*) secara terintegrasi. Data monitoring pada sistem saat ini digunakan untuk pemantauan kondisi secara langsung (*real-time monitoring*), namun belum mencakup pengolahan data historis untuk analisis tren, evaluasi performa jangka panjang, maupun prediksi kondisi sistem.

Keterbatasan lain terdapat pada aspek keamanan komunikasi. Sistem yang dirancang berfokus pada implementasi fungsi *monitoring* dan komunikasi data, sehingga aspek keamanan seperti autentikasi pengguna, enkripsi komunikasi, dan pengamanan akses jaringan belum menjadi bagian utama dalam penelitian ini. Pada

penerapan sistem secara lebih luas, aspek keamanan perlu diperhatikan untuk menjaga keandalan dan integritas data.

Secara keseluruhan, sistem yang dikembangkan telah mampu memenuhi fungsi utama sebagai sistem *monitoring* berbasis HMI dengan komunikasi Modbus TCP dan MQTT. Sistem memiliki kemampuan dalam melakukan transmisi data, menampilkan informasi proses secara *real-time*, serta mendukung interaksi operator melalui antarmuka HMI. Namun, pengembangan lebih lanjut masih diperlukan terutama pada aspek kestabilan komunikasi, penyimpanan *data historis*, skalabilitas sistem, dan peningkatan keamanan agar sistem dapat diterapkan pada lingkungan yang lebih kompleks.