

BAB II

DASAR TEORI

2.1 Tinjauan Pustaka

Setelah melakukan studi literatur terhadap topik terkait baik dari jurnal dan paper nasional maupun internasional, didapati beberapa penelitian yang menjadi inspirasi oleh penulis. Berikut beberapa penelitian tersebut.

Tabel 2. 1 Tinjauan Pustaka

No	Judul Penelitian	Keterangan	Target Pembaharuan
1.	<i>An Open-Source Supervisory Control and Data Acquisition Architecture for Photovoltaic System Monitoring Using ESP32, Banana Pi M4, and Node-RED</i>	Penelitian ini mengembangkan arsitektur SCADA open-source untuk pemantauan sistem fotovoltaiik. ESP32-E mengirimkan data tegangan, arus, dan daya melalui MQTT ke Banana Pi M4, sedangkan Node-RED digunakan untuk membangun dashboard HMI yang mendukung pemantauan secara real-time dan kendali beban jarak jauh. Hasil pengujian menunjukkan bahwa fungsi akuisisi data, visualisasi, dan kendali dapat berjalan pada skala laboratorium.	Penelitian ini menerapkan Node-RED sebagai gateway komunikasi dua arah antara Modbus TCP pada OpenPLC dan MQTT pada HMI Haiwell, bukan hanya sebagai dashboard. Sistem digunakan untuk memantau suhu, laju aliran, dan level air serta meneruskan nilai setpoint operator melalui HMI Haiwell B7H-W dan Haiwell Cloud SCADA.
2.	<i>Temperature and Humidity Monitoring System Using</i>	Penelitian ini merancang sistem <i>monitoring</i> suhu dan kelembapan berbasis IoT. Data sensor dikirim melalui	Penelitian ini memperluas parameter yang dipantau menjadi suhu, laju aliran, dan

	<i>Node-Red Based on MQTT Protocol</i>	protokol MQTT dan diolah oleh Node-RED untuk ditampilkan pada <i>dashboard</i> interaktif secara <i>real-time</i> . Pengujian menunjukkan akurasi sistem di atas 98,4% dengan rata-rata selisih pengukuran suhu sebesar 0,4 °C.	level air serta mengintegrasikannya dengan OpenPLC. Sistem juga menambahkan fungsi kendali supervisi berupa pengiriman nilai setpoint dari HMI, dengan Node-RED sebagai penghubung komunikasi Modbus TCP dan MQTT.
3.	<i>Sistem Pemantauan dan Kontrol Otomatis Kualitas Air Berbasis IoT Menggunakan Platform Node-RED untuk Budidaya Udang</i>	Penelitian ini memantau dan mengendalikan kualitas air budidaya udang menggunakan sensor dissolved oxygen, pH, kekeruhan, suhu, dan ketinggian air. Data dikirimkan ke Node-RED melalui MQTT dan diolah menggunakan metode IFTTT untuk menghasilkan perintah kepada aktuator. Pengujian mencatat rata-rata delay pengiriman data dari publisher ke subscriber sebesar 260 ms.	Penelitian ini berfokus pada simulator kontrol proses industri. Node-RED digunakan untuk memetakan data dua arah antara OpenPLC dan HMI Haiwell melalui Modbus TCP dan MQTT untuk monitoring suhu, laju aliran, level air, serta pengiriman nilai setpoint dari operator.
4.	<i>Automatic Water Level and Pressure Control System Prototype Design Using</i>	Penelitian ini mengembangkan prototipe pengendalian otomatis level dan tekanan air menggunakan PLC Omron CP2E dan HMI layar sentuh Omron. HMI	Penelitian ini menggunakan ESP32 dan OpenPLC serta mengintegrasikan sistem kontrol suhu dan laju aliran air ke dalam satu

	<i>Programmable Logic Controller and Human Machine Interface</i>	digunakan untuk memvisualisasikan kondisi plant secara real-time sekaligus sebagai media kendali jarak jauh. Hasil pengujian menunjukkan bahwa program ladder dan desain HMI dapat menjalankan sistem dengan baik.	HMI Haiwell. Node-RED digunakan sebagai gateway protokol, sedangkan Haiwell Cloud SCADA menyediakan pemantauan jarak jauh dan mendukung pertukaran variabel proses serta nilai setpoint secara dua arah.
5.	<i>Supervisory Control and Data Acquisition Approach in Node-RED: Application and Discussions</i>	Penelitian ini mengembangkan aplikasi supervisi berbasis Node-RED yang menyerupai fungsi SCADA. Modbus TCP digunakan untuk berkomunikasi dengan perangkat industri, sedangkan MQTT digunakan untuk komunikasi pada tingkat yang lebih tinggi menggunakan pola publish-subscribe. Sistem juga menyediakan dashboard, pencatatan data, dan pengarsipan menggunakan InfluxDB.	Penelitian ini menerapkan integrasi Modbus TCP dan MQTT secara khusus pada simulator kontrol proses dua tangki menggunakan ESP32, OpenPLC, dan HMI Haiwell B7H-W. Pengujian meliputi monitoring tiga parameter proses, pengiriman setpoint dua arah, latensi komunikasi, fungsi antarmuka, dan pemulihan koneksi.

Berdasarkan lima penelitian terdahulu yang disajikan pada Tabel 2.1, Node-RED telah dimanfaatkan untuk melakukan akuisisi, pengolahan, visualisasi, dan supervisi data pada sistem berbasis Internet of Things (IoT). Penerapannya mencakup dashboard SCADA untuk pemantauan sistem fotovoltaik, monitoring suhu dan kelembapan, serta pemantauan dan kontrol kualitas air dengan pengiriman

data melalui MQTT[7]. Penelitian lainnya menunjukkan bahwa HMI yang terhubung dengan PLC dapat digunakan untuk menampilkan kondisi plant secara real-time sekaligus memberikan perintah kendali[8]. Selain itu, Node-RED juga telah diterapkan sebagai platform supervisi yang menghubungkan perangkat industri melalui Modbus TCP dengan sistem pada tingkat yang lebih tinggi melalui MQTT[4].

Walaupun penelitian-penelitian tersebut telah menunjukkan kemampuan Node-RED, MQTT, PLC, dan HMI dalam sistem monitoring dan kendali, masing-masing masih memiliki ruang untuk dikembangkan. Sebagian penelitian menggunakan Node-RED terutama sebagai dashboard visualisasi atau pusat pemrosesan data MQTT, sedangkan penelitian mengenai PLC dan HMI masih menggunakan perangkat dalam satu sistem kendali[2], [7], [8], [9]. Pada lima penelitian yang ditinjau, integrasi Modbus TCP dan MQTT melalui Node-RED belum diterapkan secara khusus pada simulator kontrol suhu dan laju aliran air yang menggunakan OpenPLC, HMI Haiwell fisik, dan Cloud SCADA dalam satu arsitektur komunikasi dua arah[4].

Berdasarkan celah tersebut, penelitian ini mengembangkan sistem HMI terpusat menggunakan Haiwell B7H-W dan Haiwell Cloud SCADA untuk memantau parameter suhu, laju aliran air, dan ketinggian air pada simulator kontrol proses. Node-RED difungsikan sebagai gateway penerjemah protokol yang membaca dan menulis register OpenPLC melalui Modbus TCP serta melakukan pertukaran data dengan HMI melalui MQTT. Arsitektur tersebut mendukung komunikasi dua arah sehingga data variabel proses dapat ditampilkan secara real-time, sedangkan operator dapat mengirimkan nilai setpoint dan perintah operasi menuju sistem kontrol. Evaluasi sistem dilakukan terhadap fungsi antarmuka, keberhasilan transmisi data, latensi komunikasi, dan pemulihan koneksi.

2.2 Sistem Kontrol Proses



Gambar 2. 1 Konsep Dasar Sistem Kontrol Proses

Pada sistem kontrol proses, pengendalian dilakukan terhadap parameter tertentu agar tetap berada pada kondisi yang diinginkan[8]. Pada penelitian ini, parameter yang dikontrol meliputi suhu dan laju aliran air dalam suatu sistem berbasis tangki dan sirkulasi fluida.

Berdasarkan Gambar 2.1, sistem terdiri dari dua buah tangki yang saling terhubung melalui pipa aliran. Tangki pertama berfungsi sebagai tempat proses pemanasan air, sedangkan tangki kedua digunakan sebagai bagian dari sistem sirkulasi dan distribusi fluida. Air dialirkan menggunakan pompa melalui pipa yang telah dilengkapi dengan *flow* sensor untuk mengukur laju aliran air secara *real-time*.

Pengukuran suhu dilakukan menggunakan sensor suhu yang terpasang pada tangki utama. Data suhu yang diperoleh digunakan untuk mengetahui kondisi aktual sistem, khususnya dalam proses pemanasan yang dilakukan oleh *heater*. Selain itu, sistem juga dilengkapi dengan sensor *water level* untuk memantau ketinggian air di dalam tangki.

Seluruh data dari sensor dikirimkan ke mikrokontroler ESP32 untuk diproses. Mikrokontroler berfungsi sebagai pusat pengendali yang mengolah data masukan dan menghasilkan sinyal keluaran untuk mengatur kerja aktuator seperti *heater*, pompa, dan modul pendingin (*peltier*).

Sistem ini dilengkapi dengan antarmuka Human Machine Interface (HMI) untuk menampilkan parameter utama berupa suhu, ketinggian air, dan laju aliran air. Selain melakukan monitoring, operator dapat memasukkan nilai Setpoint (SP) melalui HMI sebagai nilai acuan bagi sistem kontrol.

Pertukaran data antara sistem kontrol dan antarmuka dilakukan menggunakan arsitektur komunikasi hibrid. OpenPLC berkomunikasi dengan gateway Node-RED melalui protokol Modbus TCP pada jaringan lokal, sedangkan Node-RED berkomunikasi dengan HMI Haiwell menggunakan protokol MQTT[1], [4]. Dengan arsitektur tersebut, data variabel proses dapat ditampilkan pada HMI dan instruksi dari operator dapat diteruskan kembali menuju sistem kontrol.

Integrasi tersebut memungkinkan proses monitoring dan kendali supervisi dilakukan melalui satu antarmuka sehingga operator dapat mengamati kondisi proses dan memberikan perubahan nilai acuan tanpa berinteraksi langsung dengan perangkat kontrol di lapangan.

2.3 Mikrokontroler ESP32

Mikrokontroler merupakan sebuah sistem komputer kecil yang terintegrasi dalam satu *chip* yang berfungsi untuk mengendalikan berbagai perangkat elektronik. Salah satu mikrokontroler yang banyak digunakan dalam sistem IoT adalah ESP32[10].



Gambar 2. 2 ESP32 [10]

ESP32 memiliki berbagai keunggulan seperti kemampuan komunikasi WiFi dan Bluetooth, kecepatan pemrosesan yang tinggi, serta jumlah pin *input-output* yang cukup banyak. Dengan kemampuan tersebut, ESP32 sangat cocok digunakan dalam sistem *monitoring* berbasis IoT[10].

Tabel 2. 2 Spesifikasi ESP32 [10]

Parameter	Nilai
Prosesor	Dual-core Tensilica LX6
Frekuensi	Hingga 240 MHz
Tegangan kerja	3.0V – 3.3V
<i>Input supply</i> (VIN)	5V
GPIO	± 30 pin (tergantung board)
ADC	12-bit <i>resolution</i>
Komunikasi	WiFi 802.11 b/g/n, Bluetooth v4.2 (BLE + Classic)
Memori	SRAM 520KB + Flash (tergantung modul)
<i>Interface</i>	UART, SPI, I2C, PWM
Fitur tambahan	<i>Deep sleep mode, touch sensor, RTC</i>

Dalam penelitian ini, ESP32 berfungsi sebagai unit akuisisi dan pemrosesan data lapangan yang terhubung dengan sensor suhu PT100, sensor laju aliran YF-S201, dan sensor ketinggian air JSN-SR04T. Data hasil pembacaan sensor digunakan sebagai variabel proses pada sistem kontrol berbasis OpenPLC. Data tersebut selanjutnya dapat diakses oleh gateway Node-RED melalui komunikasi Modbus TCP untuk diteruskan menuju sistem HMI.

2.4 OpenPLC

OpenPLC merupakan perangkat lunak *open-source* yang digunakan untuk mengimplementasikan sistem kontrol berbasis *Programmable Logic Controller* (PLC). OpenPLC memungkinkan pengguna untuk merancang dan menjalankan logika kontrol menggunakan standar internasional bagian ketiga dari seri IEC 61131 yang mendefinisikan bahasa pemrograman untuk *Programmable Logic Controller* (PLC) dan sistem otomasi industri (IEC 61131-3)[1].



Gambar 2. 3 Softwre Open PLC Editor

Tabel 2. 3 Spesifikasoi Software Open PLC Editor

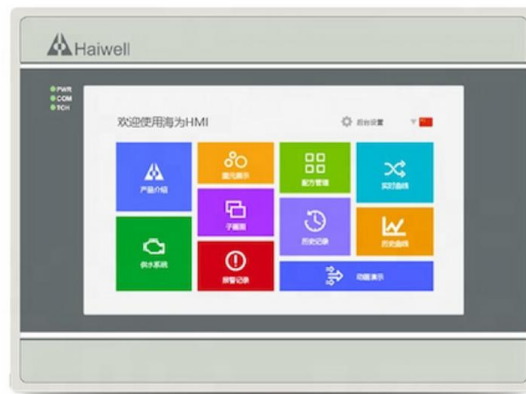
Parameter	Keterangan
Tipe software	Open-source PLC programming tool
Standar	IEC 61131-3
Bahasa pemrograman	Ladder Diagram, FBD, ST, SFC, IL
Platform	Windows, Linux, Raspberry Pi
Komunikasi	Modbus TCP/IP, MQTT, Ethernet
Fungsi utama	Programming & runtime PLC
Integrasi	Arduino, ESP32, Raspberry Pi, industrial PLC
Kegunaan	Otomasi industri, IoT, kontrol proses

Dalam penelitian ini, OpenPLC digunakan sebagai kontroler logika yang menjalankan fungsi pengolahan variabel proses dan logika pengendalian pada sistem. Selain menjalankan logika kontrol, OpenPLC menyediakan data register yang dapat dibaca maupun ditulis melalui protokol Modbus TCP sehingga dapat diintegrasikan dengan gateway Node-RED[1], [5].

2.5 Human Machine Interface (HMI)

Human Machine Interface (HMI) merupakan suatu sistem yang digunakan sebagai penghubung antara manusia dengan mesin atau sistem otomatis. HMI memungkinkan operator untuk memantau kondisi sistem serta memberikan perintah pengendalian melalui tampilan antarmuka yang mudah dipahami[8].

Dalam sistem industri, HMI biasanya digunakan untuk menampilkan informasi seperti nilai sensor, status sistem, grafik data, serta alarm apabila terjadi kondisi tertentu pada sistem. Dengan adanya HMI, proses pengawasan terhadap sistem dapat dilakukan dengan lebih mudah dan efisien[11].



Gambar 2. 4 HMI Haiwell B7H-W 7 Inch

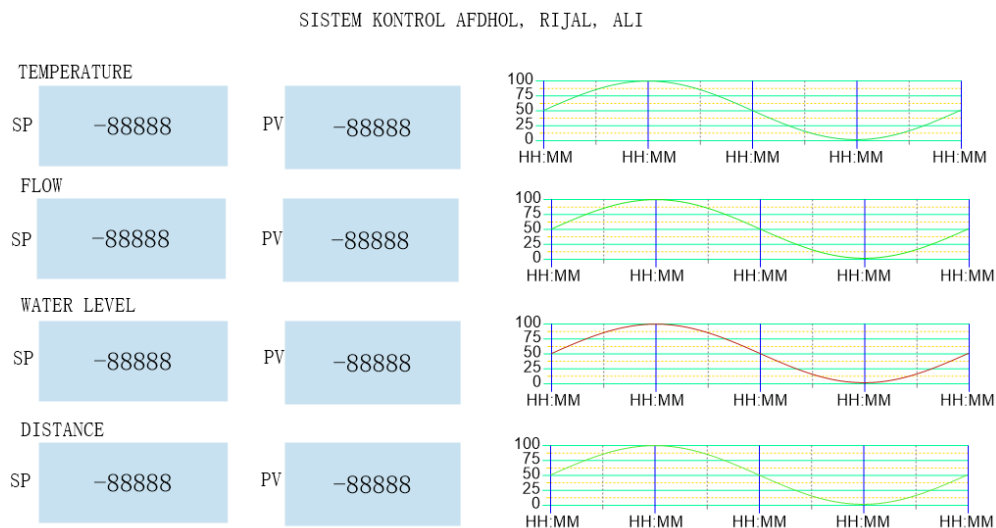
Tabel 2. 4 Spesifikasi HMI B7H-W Haiwell 7 Inch

Parameter	Nilai
Model	B7H-W Haiwell IoT Cloud HMI
Ukuran layar	7 inch TFT LCD Touchscreen
Resolusi	1024 × 600 (HD)
Warna	16.7 juta warna
Touchscreen	Resistive touch
CPU	ARM Cortex-A8
RAM	512 MB
Storage	4 GB Flash
Komunikasi	RS485 / RS232 / Ethernet (LAN) / WiFi
Protokol	Modbus RTU, Modbus TCP, MQTT, HTTP
Tegangan kerja	24V DC ±20%
Konsumsi daya	±7 Watt
Sistem operasi	Embedded Linux
Software	Haiwell Cloud SCADA

Parameter	Nilai
Fitur tambahan	IoT cloud, remote monitoring, alarm system

Dalam sistem kendali *loop* tertutup (seperti kontrol PID), HMI tidak hanya berfungsi sebagai sarana visualisasi data (*monitoring*), tetapi juga sebagai media intervensi operator (*supervisory control*). Melalui HMI, operator dapat melakukan pengaturan *Setpoint* (SP) secara *real-time*, memantau respon variabel proses (PV) terhadap gangguan, serta melihat performa algoritma kontrol PID secara langsung melalui grafik tren, sehingga HMI menjadi pusat kendali yang krusial dalam menjaga kestabilan proses[8].

2.6 Sistem *monitoring* Berbasis *Dashboard*



Gambar 2. 5 Tampilan Sistem Monitoring Berbasis Dashboard

Sistem *monitoring* berbasis *dashboard* HMI (*Human Machine Interface*) merupakan sistem yang digunakan untuk menampilkan kondisi proses secara visual dan *real-time* sehingga memudahkan pengguna dalam melakukan pemantauan dan pengendalian. *Dashboard* HMI dapat menampilkan berbagai informasi, seperti nilai sensor, status aktuator, serta parameter pengendalian dalam satu antarmuka yang terintegrasi[7].

Pada penelitian ini, sistem *monitoring* menggunakan Haiwell Cloud SCADA yang terintegrasi dengan HMI Haiwell B7H-W. Sistem ini digunakan untuk

memantau temperatur air, *level* air, laju aliran, serta status aktuator secara *real-time*. Selain itu, Haiwell Cloud SCADA memiliki beberapa fitur utama, yaitu:

1. *Monitoring* parameter proses secara *real-time*.
2. Visualisasi data dalam bentuk pengaturan *setpoint* dan grafik.
3. *Remote monitoring* melalui jaringan internet[12].

Dalam penelitian ini, konsep Internet of Things diterapkan pada proses pertukaran data antara sistem kontrol, gateway komunikasi, dan antarmuka HMI berbasis jaringan. Data variabel proses dari perangkat lapangan diteruskan melalui Node-RED menuju HMI Haiwell dan Haiwell Cloud SCADA sehingga kondisi proses dapat dipantau secara *real-time* melalui jaringan.

2.7 Internet of Things (IoT)



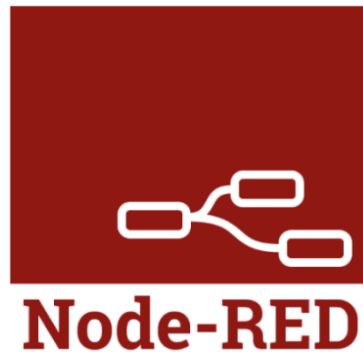
Gambar 2. 6 *Internet of Things (IoT)*

Internet of Things (IoT) merupakan konsep teknologi yang memungkinkan berbagai perangkat elektronik untuk saling terhubung dan bertukar data melalui jaringan internet. Dengan teknologi IoT, berbagai perangkat seperti sensor, aktuator, dan sistem kontrol dapat terintegrasi dalam satu sistem yang terhubung secara digital[4].

Penerapan IoT dalam sistem *monitoring* memungkinkan proses pengumpulan dan pengiriman data dilakukan secara otomatis tanpa memerlukan interaksi manusia secara langsung. Data yang diperoleh dari sensor dapat dikirimkan ke *server* atau *dashboard monitoring* sehingga kondisi sistem dapat dipantau dari jarak jauh[7], [13].

Dalam penelitian ini, konsep IoT diterapkan pada pertukaran data antara sistem kontrol, Node-RED sebagai *gateway* komunikasi, HMI Haiwell B7H-W, dan Haiwell Cloud SCADA. Node-RED membaca data register OpenPLC melalui Modbus TCP dan meneruskan data tersebut menuju HMI Haiwell melalui MQTT sehingga kondisi proses dapat dipantau secara *real-time* melalui jaringan.

2.8 Node-RED



Gambar 2. 7 Logo *Software* Node-RED

Node-RED merupakan *platform* pemrograman berbasis *flow* yang digunakan untuk menghubungkan berbagai perangkat keras, layanan web, dan aplikasi IoT. Node-RED dikembangkan dengan konsep *flow-based programming* sehingga pengguna dapat membuat sistem dengan cara menghubungkan berbagai node secara visual[4].

Node-RED banyak digunakan dalam pengembangan sistem IoT karena memiliki berbagai fitur seperti pengolahan data, integrasi dengan berbagai protokol komunikasi, serta pembuatan *dashboard monitoring* secara interaktif. Node-RED memiliki keunggulan:

1. mudah digunakan (*visual programming*)
2. mendukung berbagai protokol
3. dapat membuat *dashboard* interaktif[4], [7]

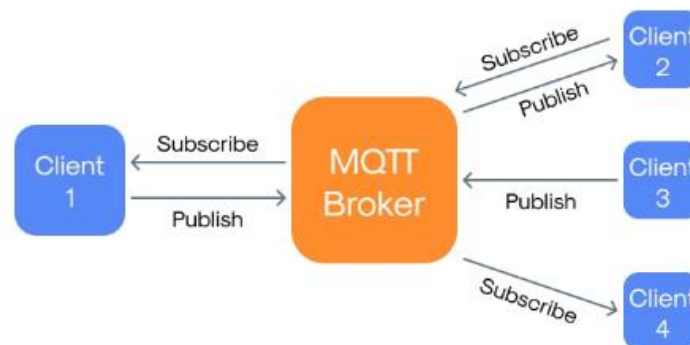
Pada penelitian ini, Node-RED digunakan sebagai *gateway* komunikasi antara OpenPLC dan HMI Haiwell. Pada jalur *monitoring*, Node-RED membaca data *register* OpenPLC melalui Modbus TCP, melakukan pengolahan dan pemetaan data, kemudian memublikasikan data tersebut melalui MQTT menuju HMI Haiwell. Pada jalur kendali, Node-RED menerima nilai *setpoint* dan perintah

operasi dari HMI melalui MQTT, kemudian meneruskannya menjadi perintah penulisan *register* pada OpenPLC melalui Modbus TCP.

Dalam sistem otomasi industri, Node-RED dapat digunakan sebagai *middleware* untuk mengintegrasikan protokol komunikasi yang berbeda. Modbus TCP menerapkan komunikasi *client-server* dengan pola *request-response*, sedangkan MQTT menggunakan pola *publish-subscribe* melalui *broker*. Melalui *flow* yang dirancang, Node-RED dapat memetakan nilai *register* Modbus TCP menjadi pesan MQTT dan melakukan proses sebaliknya. Mekanisme tersebut memungkinkan pertukaran data dua arah antara perangkat kontrol di lapangan dan sistem monitoring berbasis HMI atau cloud[4], [5], [6].

2.9 Protokol MQTT

Message Queuing Telemetry Transport (MQTT) merupakan protokol komunikasi yang banyak digunakan dalam sistem *Internet of Things*. MQTT bekerja dengan menggunakan metode *publish* dan *subscribe* dimana perangkat pengirim data (*publisher*) mengirimkan data ke *broker*, kemudian perangkat penerima (*subscriber*) menerima data dari *broker* tersebut[6].



Gambar 2. 8 Protokol MQTT

Protokol MQTT memiliki keunggulan dalam efisiensi penggunaan *bandwidth* serta kemampuan bekerja pada jaringan dengan keterbatasan koneksi. Oleh karena itu, MQTT sangat cocok digunakan pada sistem *monitoring* berbasis IoT[6].

Dalam penelitian ini, MQTT digunakan sebagai media komunikasi antara mikrokontroler dan sistem *dashboard* Node-RED sehingga data sensor dapat dikirimkan secara *real-time*.

2.9.1 Konsep *Publish-Subscribe*

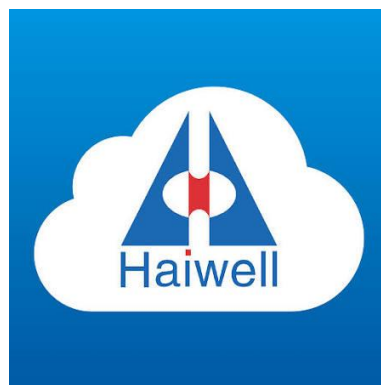
MQTT menggunakan mekanisme *publish-subscribe* yang terdiri atas *publisher*, *broker*, dan *subscriber*. *Publisher* berfungsi mengirimkan pesan ke suatu topik, *broker* bertugas menerima dan mendistribusikan pesan, sedangkan *subscriber* menerima pesan berdasarkan topik yang telah didaftarkan[14].

Pada jalur *monitoring* dalam penelitian ini, Node-RED bertindak sebagai *publisher* yang mengirimkan data variabel proses menuju *broker* MQTT, sedangkan HMI Haiwell bertindak sebagai *subscriber*. Sebaliknya, pada jalur pengiriman instruksi operator, HMI bertindak sebagai *publisher* untuk mengirimkan nilai *Setpoint* dan perintah operasi, sedangkan Node-RED bertindak sebagai *subscriber*.

Penggunaan mekanisme tersebut memungkinkan komunikasi data berlangsung secara dua arah tanpa mengharuskan perangkat pengirim dan penerima berkomunikasi secara langsung[14].

Dalam penelitian ini, MQTT digunakan sebagai protokol komunikasi antara *gateway* Node-RED dengan HMI Haiwell B7H-W. Pada jalur *monitoring*, Node-RED mempublikasikan data variabel proses berupa suhu, laju aliran air, dan ketinggian air melalui *broker* MQTT untuk diterima oleh HMI. Pada jalur kendali, HMI mempublikasikan nilai *Setpoint* serta perintah operasi yang kemudian diterima oleh Node-RED untuk diteruskan menuju OpenPLC.

2.10 Haiwell Cloud SCADA



Gambar 2. 9 *Software* Haiwell Cloud SCADA

Haiwell Cloud SCADA merupakan platform *Supervisory Control and Data Acquisition* (SCADA) berbasis cloud yang dikembangkan oleh Haiwell untuk

mendukung proses monitoring dan pengendalian sistem otomasi industri. Platform ini memungkinkan pengguna untuk memantau parameter sistem secara real-time melalui antarmuka grafis yang terintegrasi dengan perangkat HMI Haiwell. Selain itu, Haiwell Cloud SCADA mendukung komunikasi data melalui jaringan internet sehingga memudahkan proses pemantauan dari berbagai lokasi[7].

Pada penelitian ini, Haiwell Cloud SCADA digunakan sebagai media monitoring untuk menampilkan data suhu, level air, dan laju aliran air yang diperoleh dari sistem kontrol. Integrasi antara Haiwell B7H-W dan Haiwell Cloud SCADA memungkinkan pengguna memperoleh informasi kondisi sistem secara cepat, akurat, dan mudah dipahami melalui tampilan antarmuka yang interaktif.

2.11 Protokol Modbus TCP

Modbus TCP adalah salah satu protokol komunikasi standar industri yang beroperasi di atas lapisan TCP/IP (*Transmission Control Protocol / Internet Protocol*). Protokol ini dikembangkan dari protokol Modbus konvensional agar dapat diaplikasikan pada jaringan Ethernet, memungkinkan komunikasi data antarperangkat industri secara andal[5].



Gambar 2. 10 Logo Modbus

Modbus TCP menggunakan arsitektur *client-server* (atau *master-slave*) dengan pola komunikasi *request-response*, di mana perangkat *client* mengirimkan permintaan pembacaan atau penulisan data (*register*) kepada perangkat *server*. Protokol ini sangat handal dan banyak digunakan dalam sistem otomasi industri karena bersifat deterministik, stabil, serta dapat diintegrasikan secara langsung dengan perangkat kontrol seperti PLC (*Programmable Logic Controller*) di dalam jaringan lokal (*Local Area Network*)[5].

Dalam penelitian ini, Modbus TCP berfungsi sebagai jembatan komunikasi lokal yang menghubungkan sistem *gateway* dengan perangkat kontroler lapangan

untuk memastikan pertukaran data kontrol berjalan secara akurat tanpa penundaan yang signifikan.

2.12 Parameter Pengujian Sistem

Pengujian sistem komunikasi dilakukan untuk mengetahui performa pertukaran data antara perangkat kontrol, gateway Node-RED, dan antarmuka Human Machine Interface (HMI). Parameter pengujian yang digunakan meliputi kesesuaian transmisi data, nilai error komunikasi, latency pengiriman data, serta tingkat keberhasilan transmisi. Parameter tersebut digunakan sebagai dasar analisis performa sistem monitoring yang telah dirancang.

2.12.1 Pengujian Kesesuaian Transmisi Data

Pengujian kesesuaian transmisi data bertujuan untuk mengetahui tingkat kesamaan nilai antara data yang dikirimkan oleh perangkat pengirim dengan data yang diterima oleh perangkat penerima. Pada penelitian ini, pengujian dilakukan dengan membandingkan nilai setpoint yang dikirimkan melalui HMI dengan nilai yang diterima oleh Node-RED. Nilai selisih antara data pengirim dan penerima dapat dihitung menggunakan persamaan:

$$E = |X_s - X_r|$$

Keterangan:

E = nilai selisih data

X_s = nilai data yang dikirimkan (sender)

X_r = nilai data yang diterima (receiver)

Nilai error yang semakin kecil menunjukkan bahwa proses transmisi data memiliki tingkat kesesuaian yang semakin baik. Untuk mengetahui tingkat kesalahan dalam bentuk persentase digunakan persamaan:

$$Error (\%) = \frac{X_s - X_r}{X_s} \times 100\%$$

Keterangan:

$(\text{Error}(\%)) = \text{persentase kesalahan transmisi data}$

$(X_s) = \text{nilai data yang dikirimkan}$

$(X_r) = \text{nilai data yang diterima}$

2.12.2 Pengujian Latency Komunikasi

Latency merupakan waktu jeda yang dibutuhkan suatu data mulai dari proses pengiriman hingga data tersebut diterima oleh perangkat tujuan. Pengujian latency digunakan untuk mengetahui kecepatan respons komunikasi pada sistem yang menggunakan protokol MQTT dan Modbus TCP. Nilai latency dapat dihitung menggunakan persamaan:

$$L = T_r - T_s$$

Keterangan:

L = nilai latency komunikasi (ms)

T_s = waktu data dikirimkan (send time)

T_r = waktu data diterima (receive time)

Pada penelitian ini, pengukuran latency dilakukan pada dua jalur komunikasi, yaitu:

1. Latency MQTT

Pengukuran dilakukan pada jalur komunikasi MQTT antara Node-RED dan HMI. Data yang diamati adalah selisih antara stempel waktu data masuk (DataMqttInTime) dan stempel waktu data keluar (DataMqttOutTime) yang dicatat pada context data Node-RED.

2. Latency Modbus TCP

Pengukuran dilakukan pada jalur komunikasi Modbus TCP antara Node-RED dan OpenPLC. Data yang diamati adalah selisih antara stempel waktu data masuk (DataModbusInTime) dan stempel waktu data keluar (DataModbusOutTime) yang dicatat pada context data Node-RED.

2.12.3 Rata-rata Latency Komunikasi

Pengujian latency dilakukan dengan beberapa kali percobaan untuk memperoleh nilai representatif. Nilai rata-rata latency dihitung menggunakan persamaan:

$$\bar{L} = \frac{\sum_{i=1}^n L_i}{n}$$

Keterangan:

$\bar{L}$ = rata-rata latency (ms)

L_i = latency pada pengujian ke-i (ms)

n = jumlah pengujian

Nilai rata-rata latency digunakan untuk menggambarkan performa komunikasi sistem secara keseluruhan.

2.12.4 Pengujian Keberhasilan Transmisi

Tingkat keberhasilan transmisi digunakan untuk mengetahui kemampuan sistem dalam mengirimkan dan menerima data sesuai jumlah data yang diuji. Persamaan keberhasilan transmisi:

$$SuccessRate = \frac{N_b}{N_t} \times 100\%$$

Keterangan:

SuccessRate = tingkat keberhasilan transmisi data (%)

N_b = jumlah data berhasil diterima

N_t = jumlah total data yang dikirim

Semakin tinggi nilai *success rate*, menunjukkan bahwa sistem komunikasi memiliki tingkat keandalan yang semakin baik dalam melakukan pertukaran data.