Sistem SCADA berbasis *Internet of Things*

Hernan S. Idris[1], Agung Saputra[1], dan Indra Hardian Mulyadi[1*]

[1]Politeknik Negeri Batam, Batam, Indonesia

[*]Email: indra@polibatam.ac.id

Abstrak—Pengendalian *supervisory control and data acquisition* (SCADA) pada PLC umumnya tidak dapat dikontrol secara IoT melalui *platform* IoT. Oleh karena itu pada penelitian ini dibuatlah sebuah pengendalian SCADA berbasis PLC yang akan ditampilkan pada *web*. PLC yang digunakan pada sistem ini yaitu Schneider TM241CE24R, dan pengontrolan sistem IoT dengan memanfaatkan modul ESP32 sebagai perantaranya. Untuk komunikasi antara ESP32 dengan *device* PLC dan *Power Meter* digunakan protokol komunikasi Modbus RTU yang mana data akan diolah di ESP32 sebelum mengirim data ke server. Sistem monitoring SCADA menggunakan *web* dengan Node.JS sebagai sisi server dan Express.JS sebagai framework, serta menggunakan Message Queuing Telemetry Transport (MQTT) sebagai protokol komunikasi antara ESP32 dan server. Pengujian sistem terdiri dari komunikasi antara PLC dan *power meter* ke ESP32, komunikasi ESP32 ke Node.JS, pengujian *delay* pengiriman data dari *web* ke *device* PLC, pengujian *delay* pengiriman PLC ke *web* dan pengujian *packet loss* pada pengiriman data menggunakan *software* Wireshark. Hasilnya penelitian ini bisa memudahkan pada sistem SCADA yang menggunakan RS485 sebagai komunikasi yang bisa dimonitoring secara *realtime*, rata-rata *delay respond time* pada komunikasi *device* PLC dan *Power Meter* ke *web* maupun sebaliknya yaitu 0,3 detik pada saat terhubung internet baik dan rata-rata *delay respond time* 0,5-1 detik pada saat internet buruk dan sistem ini bisa diakses di mana saja selama alat ini terhubung dengan internet.

Kata kunci: IoT, Mikrokontroler, PLC, Sistem SCADA, *Web*

Abstract—Supervisory Control and Data Acquisition (SCADA) control on PLCs generally cannot be controlled by IoT via the IoT platform. Therefore, in this study, a PLC-based SCADA control was created which will be displayed on the web. The PLC used in this system is Schneider TM241CE24R, and controlling the IoT system by utilizing the ESP32 module as an intermediary. For communication between the ESP32 and the PLC and Power meter devices, the Modbus RTU communication protokol is used where data will be processed on ESP32 before sending data to the server. The SCADA monitoring system uses the web with Node.JS as the server side and Express.JS as the framework, and uses Message Queuing Telemetry Transport (MQTT) as the communication protokol between ESP32 and the server. System testing consists of communication between the PLC and the Power meter to ESP32, ESP32 communication to Node.JS, testing the delay in sending data from the web to the PLC device, testing the delay in sending PLC to the web and testing packet loss in sending data using the Wireshark software. The results of this research can make it easier for SCADA systeMS that use RS485 as a communication that can be monitored in realtime, the average delay response time on PLC and Power meter communication devices to the web is 0.3 seconds when connected to the good internet and the average delay response time of 0.5-1 second when the internet is bad and this system can be accessed anywhere as long as this device is connected to the internet.

Keywords: IoT, Microcontroler, PLC, SCADA system, Web

I. PENDAHULUAN

PADA era industri 4.0 ini perkembangan teknologi mengalami percepatan yang signifikan, banyak teknologi yang dirancang untuk mempermudah sistem kerja [1]. IoT didefinisikan sebagai pemanfaatan media yang dapat diakses untuk menyederhanakan berbagai aktivitas dan menghubungkannya dengan layanan *cloud*, sehingga meningkatkan efisiensi [2].

SCADA adalah sebuah sistem yang mampu mengawasi, mengendalikan, dan mengakuisisi data dari sebuah *plant* [3].

Perkembangan IoT di bidang industri diharapkan dapat menciptakan lapangan kerja baru dan memberikan dampak ekonomis yang besar pada tahun 2030 [4]. Pada masa kini, konsep IoT telah menjadi lebih mudah untuk dikembangkan berkat kemajuan infrastruktur jaringan yang semakin luas dan akses internet yang mudah. Oleh karena itu, dalam penelitian ini akan dibuat sebuah sistem SCADA berbasis *web* yang mengadopsi konsep IoT. Sistem ini akan memungkinkan pengguna untuk memantau dan mengontrolnya dari jarak jauh selama perangkat terhubung ke internet.

Dalam pembuatan sistem ini, akan menggunakan beberapa komponen elektronik seperti PLC, *Power Meter*, dan mikrokontroler. *Software* Arduino IDE digunakan untuk memprogram mikrokontroler, sementara Express.JS digunakan sebagai *framework* dalam pembuatan aplikasi *web*. Sistem ini memiliki kemampuan untuk melakukan monitoring dan mengontrol melalui sistem SCADA. Selain itu, sistem ini juga dapat terhubung kejaringan *Wifi* yang dapat mengirimkan *input*, *output* PLC, dan data *Power Meter* ke ESP32 melalui Modbus RTU RS485. Hal ini memungkinkan kita untuk melakukan monitoring dan mengontrol sistem SCADA secara *realtime* melalui aplikasi *web*.

Penelitian yang dilakukan oleh F. Ardiansyah, M. F. Lawasi, and C. F. Hadi, pada tahun 2019 yang berjudul "Sistem Monitoring Inkubator Penetas Telur Berbasis Android", menjelaskan bahwa penerapan sistem IoT bisa digunakan sebagai pengawas mesin penetas telur dengan menggunakan kontroler NodeMCU. Kontroler ini mampu memberikan respons waktu yang relatif lambat, yakni sekitar 18,84 detik untuk proses pengiriman dan penerimaan data [5]. Untuk aplikasi SCADA diperlukan respon waktu yang lebih cepat, mendekati *realtime*.

Berdasarkan penelitian yang telah dilakukan oleh M. C. Afrian, M. T. Asron and R. Wicaksono pada tahun 2018 yang berjudul "Prototipe Pengangkut Sampah Otomatis pada Pintu Ali dengan Sistem Informasi Menggunakan NodeMCU ESP8266 Berbasis PLC," menjelaskan dilakukannya pengembangan pada sistem IoT yang diterapkan menggunakan PLC. Metode ini diterapkan pada sistem pengangkut sampah, di mana pengawasan dan pengontrolan dapat dilakukan melalui pengiriman email yang didukung oleh mikrokontroler yang terpasang di dalamnya [6]. Untuk penggunaan *e-mail* dirasa kurang praktis.

Berdasarkan penelitian yang telah dilakukan oleh Angga Wahyu Aditya, Nur Rani Alham, Restu Mukti Utomo, dan Hilmansyah pada tahun 2023 yang berjudul "Sistem Pemantauan Konsumsi Energi Listrik Berbasis *web* Sebagai Upaya Konservasi Energi,", Penelitian ini menjelaskan penggunaan *Power Meter* PM2120 yang dilengkapi dengan ESP32 memungkinkan dapat koneksi ke internet [7]. Sistem hanya dapat mengontrol satu arah.

Pada penelitian yang dilakukan oleh P. Avina, H. Wicaksono and P. Santoso, pada tahun 2018 yang berjudul "Sistem Keamanan Bangunan Multi Lokasi Berbasis IoT Menggunakan Siemens Logo dan Raspberry Pi", Penelitian ini menjelaskan bahwa peneliti menggunakan PLC Siemens Logo dan Raspberry Pi sebagai *web* server. Hasil penelitian menunjukkan bahwa respon waktu yang didapatkan adalah 4,15 detik. Metode pemantauan yang menggunakan notifikasi pada aplikasi [8]. Kontroler yang digunakan relatif mahal.

Pada penelitian yang dilakukan oleh Alif Ghifari Priambodo, pada tahun 2018 yang berjudul "Rancang Bangun Protokol Komunikasi On Board Unit (Obu) Untuk *Intelligent Transport System* (ITS) di Surabaya", menjelaskan bahwa peneliti menggunakan protokol komunikasi MQTT dan VPS sebagai server [9]. Tidak adanya *interface* untuk pengguna.

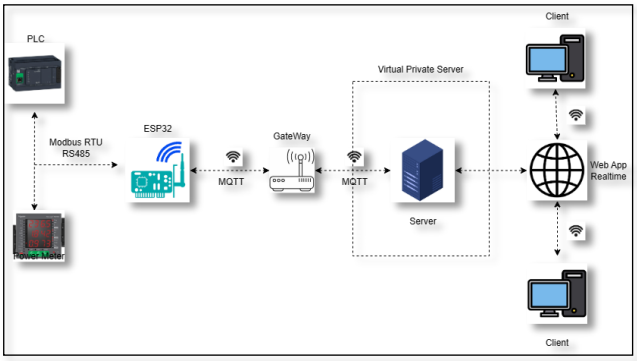
Pada penelitian yang dilakukan oleh Mus Mulyadi Usman, Xaverius B.N. Najoan, Meicsy E. I. Najoan , pada tahun 2020 yang berjudul "Rancang Bangun Aplikasi Monitoring Ketinggian Air Sungai Berbasis Internet of Things Menggunakan Amazon *web* Service", menjelaskan bahwa peneliti menggunakan layanan AWS EC2 [10]. Layanan yang digunakan relatif mahal.

Dengan dasar tersebut, tujuan dari penelitian ini adalah menciptakan sistem IoT-SCADA berbasis PLC yang dapat diakses melalui *smartphone* Android dan komputer menggunakan modul ESP32. Diharapkan sistem yang akan dibuat ini mampu memantau tanaman secara langsung dan dapat digunakan dengan mudah.

II. METODE

A. Perancangan Diagram Blok Sistem

Diagram blok terkait perancangan *web* aplikasi untuk menampilkan data *input* dan *output* PLC dan data *voltage power meter*, data tersebut merupakan data *realtime* yang dikirim melalui jaringan internet. Berikut blok diagram terkait IoT SCADA Sistem Berbasis *web* yang ditunjukan pada Gambar. 1.

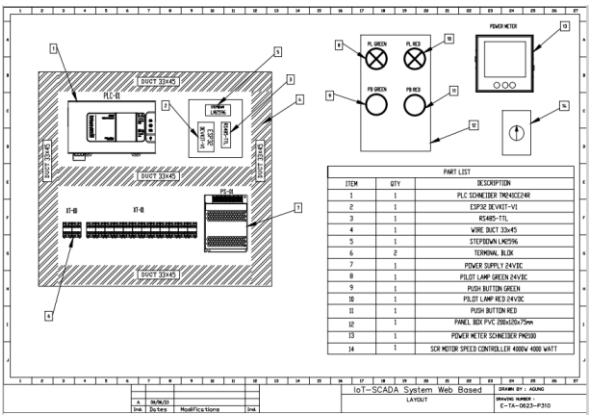


Gambar. 1. Diagram blok sistem

Gambar. 1. merupakan Diagram Blok untuk sistem yang dibuat yaitu IoT SCADA *System web Based*. Pada gambar tersebut terlihat bahwa terdapat sistem *web* Aplikasi sebagai *interface* pada client. Kemudian terdapat dua buah *slave* yaitu PLC dan *power meter*, yang berkomunikasi dengan ESP32 menggunakan protokol Modbus RTU RS485-TTL, kontroler PLC digunakan untuk *handle input* dan *output* eksternal, *power meter* digunakan untuk memberikan dan menampilkan nilai parameter voltage dan ESP32 digunakan sebagai pemberi dan penerima perintah atau data ke *web* aplikasi melalui Node.JS.

B. Perancangan Sistem Elektrikal

Untuk memulai perancangan sistem elektrik, langkah pertama yang harus diambil adalah membuat desain *wiring* yang terperinci. Dengan melakukan ini, proses instalasi panel dan pemasangan *wiring* dapat berjalan dengan terencana dan terstruktur dengan baik. Untuk mendukung pembuatan *wiring* panel diagram, desain *layout* panel ditujukkan pada Gambar. 2.

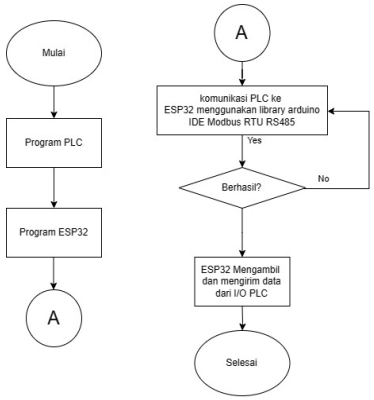


Gambar. 2. Layout panel *wiring* diagram

C. Perancangan Komunikasi dua arah PLC ke ESP32

Langkah pertama dalam perancangan komunikasi dua arah antara PLC dan ESP32 adalah membuat program untuk PLC. Hal ini diperlukan agar PLC dapat mengontrol *input* dan *output* dengan baik, selanjutnya membuat program ESP32 yang nantinya akan mengambil dan mengirim data *input* dan *output* PLC menggunakan Modbus RTU RS485, setelah membuat program PLC dan ESP32 yang harus dilakukan adalah memilih *library* Modbus RTU RS485 pada program ESP32 agar saat pengiriman data PLC ke ESP32 berhasil terbaca, *library* yang digunakan Modbus Master. Berikut *flowchart* terkait

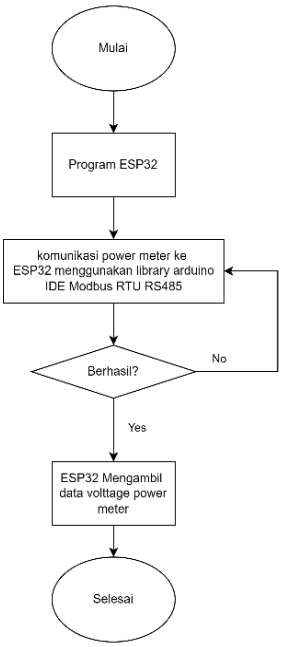
perancangan komunikasi dua arah PLC ke ESP32 yang ada pada Gambar. 3.



Gambar. 3. *Flowchart* komunikasi antara PLC dan ESP32

D. Perancangan Komunikasi Power meter ke ESP32

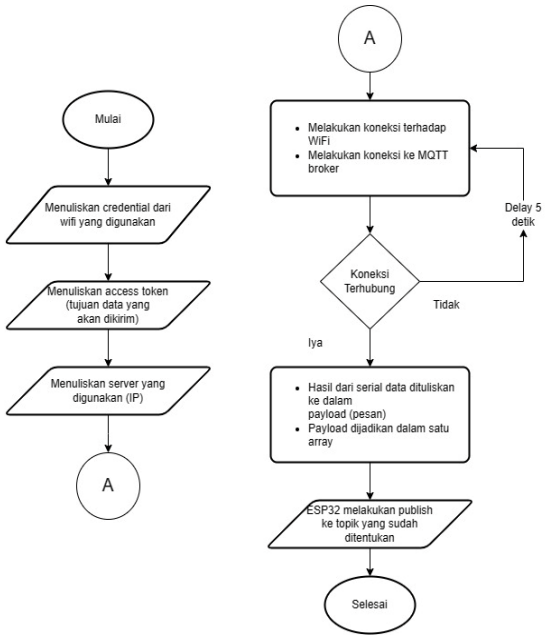
Perancangan komunikasi *power meter* ke ESP32 ini yang pertama harus dilakukan adalah membuat program ESP32 agar dapat membaca parameter *voltage* pada *power meter* menggunakan komunikasi Modbus RTU RS485, Langkah selanjutnya adalah memilih *library* Modbus RTU RS485 pada program ESP32 agar data yang diambil dari *power meter* berhasil terbaca pada ESP32, *library* yang digunakan ModbusMaster yang ada pada link berikut, https://github.com/420ma/ModbusMaster/blob/master/src/ModbusMaster.h. Berikut *flowchart* terkait perancangan komunikasi *Power Meter* ke ESP32 yang ada pada Gambar. 4.



Gambar. 4. *Flowchart* komunikasi antara power meter dan ESP32

E. Perancangan komunikasi ESP32 ke MQTT Broker

Sebelum memulai memprogram di ESP32, ada *library* yang harus di pasang terlebih dahulu untuk mendukung ESP32 dapat melakukan pengiriman data ke MQTT broker. *Library* tersebut yaitu PubSubClient.h yang dikembangkan oleh Nicholas O'Leary dari tahun 2008-2020. Pengiriman data dilakukan oleh ESP32 dengan cara mengirim data melalui jaringan *wifi* yang terhubung pada ESP32. Selain itu, untuk mengirim data ke MQTT broker menggunakan metode MQTT. Sebelum mengirim ke MQTT broker, terlebih dahulu harus memasukkan topik. *Flowchart* dalam melakukan pengiriman data dapat ditunjukan pada Gambar. 5.

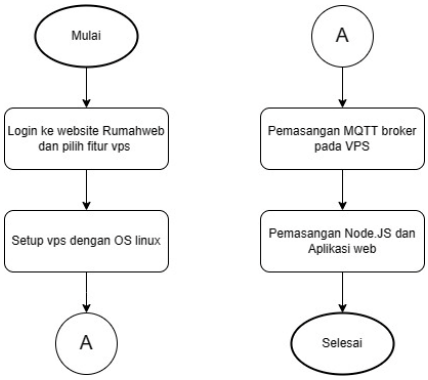


Gambar. 5. *Flowchart* komunikasi ESP32 ke MQTT broker

Dari *flowchart* yang ditunjukan pada Gambar. 5, untuk komunikasi ESP32 ke MQTT broker yang pertama dilakukan yaitu menuliskan *credential*. *Flowchart* komunikasi antara *Power Meter* dan ESP32. *Flowchart* komunikasi antara *Power Meter* dan ESP32 28 dari *wifi* yang digunakan setelah itu menuliskan *access token* kemudian menuliskan server yang akan digunakan setelah semua sudah dilakukan ESP32 akan melakukan koneksi ke *wifi* dan MQTT broker jika koneksi gagal maka ESP32 akan menunggu 5 detik untuk koneksi ulang, jika berhasil maka hasil data dari serial dituliskan kedalam *payload* dan *payload* akan dijadikan dalam satu *array*, apabila sudah maka ESP32 akan *publish* data ke topik yang sudah ditentukan.

F. Perancangan Sistem Virtual Private Server

Server yang peneliti gunakan untuk tugas akhir terletak di iot.scada.com dengan alamat IP 203.194.113.202. Peneliti memakai VPS sebagai server yang berfungsi sebagai broker MQTT dan difungsikan untuk men*deploy* aplikasi *web* berbasis node js. Broker server memiliki tanggung jawab sebagai perantara antara *publisher* dan *subscriber*. Server dapat diakses melalui *website* rumah *web*, *flowchart* sistem ditunjukkan pada Gambar. 6.

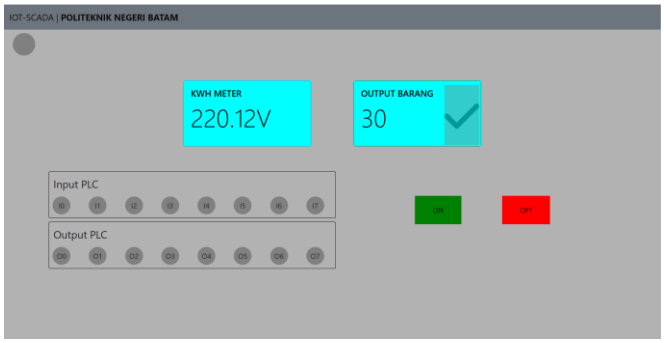


Gambar. 6. Flowchart perancangan sistem VPS

Dari Gambar. 6. kita diharuskan *login* pada *website* rumah *web* dan pilih fitur VPS setelah itu setup VPS dengan OS linux selanjutnya, kita lakukan pemasangan MQTT broker pada VPS, serta kita juga melakukan pemasangan Node.JS dan aplikasi *web* pada VPS.

G. Perancangan Aplikasi web

Pada halaman *web* digunakan untuk memonitoring data yang masuk secara *realtime* termasuk data I/O pada PLC, data tegangan yang dihasilkan oleh *power meter* dan *output* barang yang berasal dari PLC, Tampilan desain *web* ditunjukkan pada Gambar. 7.



Gambar. 7. Tampilan pengguna aplikasi *web*

Desain *web* dari Gambar. 7. menggunakan Express.js sebagai *framework*, Node.JS sebagai sisi server dan html sebagai desain tampilan. Untuk paket data yang dikirim dari ESP32 ke *web* berbentuk *string* yang dimana akan diubah menjadi *array*, data tersebut ditampilkan pada Gambar. 8.

Array	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Data	I0 PLC	I1 PLC	I2 PLC	I3 PLC	I4 PLC	I5 PLC	I6 PLC	I7 PLC	O0 PLC	O1 PLC	O2 PLC	O3 PLC	O4 PLC	O5 PLC	O6 PLC	O7 PLC	Voltage

Gambar. 8. Format *packet* data yang dikirim dari ESP32 ke *web*

III. HASIL DAN PEMBAHASAN

A. Sistem Elektronik

Sistem elektronik merupakan komponen utama yang berfungsi untuk mengolah sinyal PLC dan *power meter* lalu dikirimkan ke ESP32 melalui RS485-TTL, selanjutnya ESP32 meneruskan data yang dikirim dari *slave* ke *web* Based. Selama pengujian, semua peerangkat keras yang digunakan dipastikan berfungsi dengan baik, sesuai dengan rancangan yang ditunjukkan pada Gambar. 9.

Gambar. 9. Rangkaian alat secara fisik

Dalam sistem ini, terdapat berbagai alat yang memiliki fungsi masing-masing, sebagai berikut:

1. PLC berfungsi untuk mengolah sinyal *input* dan *output*, data tersebut dikirim ke ESP32
2. ESP32 berfungsi mengolah data *slave*, dan data akan dikirim ke Node.JS
3. *Stepdown* LM2596 sebagai penurun tegangan dari 24 V DC ke 5 V DC, dan memberikan sumber tegangan 5V DC pada ESP32.
4. RS485 to TTL sebagai alat komunikasi antara *slave* ke EPS32.
5. *Power Meter* PM2100 berperan untuk mengukur tegangan *input* yang dikontrol oleh SCR Motor *Speed Control*, dan data pengukuran tersebut dikirim ke ESP32.
6. SCR Motor Speed Control alat yang digunakan untuk mengubah tegangan beban agar nilai yang terbaca pada *Power Meter* bervariasi.
7. Terminal blok sebagai alat terminasi/penghubung kabel.
8. *Power supply* 24 VDC berfungsi sebagai pemberi tegangan pada PLC.
9. *Pilot Lamp* berfungsi sebagai *output* pada PLC.
10. *Push Button* Berfungsi sebagai *Input* pada PLC.

B. Pengujian pengiriman data dari input PLC ke ESP32

Pengujian pengiriman data dari *input* PLC ke ESP32 ditujukan untuk mengamati pengujian data yang dikirim dari berhasil terbaca atau tidak, dengan cara monitoring menggunakan *software machine expert* dan Arduino IDE. Tabel I berikut merupakan penunjukan data monitoring ESP32 pada *input* PLC.

TABLE I
HASIL MONITORING PEMBACAAN *INPUT* PLC KE ESP32

Elemen	Percobaan ke-	Monitor PLC	Monitor ESP32	Status
Push Button green	1	ON	ON	OK
	2	ON	ON	OK
	3	ON	ON	OK
	4	ON	ON	OK
	5	ON	ON	OK
Push Button Red	1	ON	ON	OK
	2	ON	ON	OK
	3	ON	ON	OK
	4	ON	ON	OK
	5	ON	ON	OK

Dari Tabel I yang telah dilampirkan, dapat disimpulkan bahwa pengujian nilai yang dikirimkan oleh PLC dan dimonitor pada ESP32 dapat terbaca secara *realtime* dan semua pengiriman data berhasil.

C. Pengujian pengiriman data dari ESP32 ke output PLC

Pengujian pengiriman data dari ESP32 *output* PLC ditujukan untuk mengamati pengujian data yang dikirim berhasil terbaca atau tidak dengan cara monitoring menggunakan *software machine expert* dan Arduino IDE. Tabel II berikut merupakan penunjukan data monitoring ESP32 pada *output* PLC.

TABLE II
PENGIRIMAN DATA DARI ESP32 KE *OUTPUT* PLC

Elemen	Percobaan ke-	Monitor PLC	Monitor ESP32	Status
Lamp Green	1	ON	ON	OK
	2	ON	ON	OK
	3	ON	ON	OK
	4	ON	ON	OK
	5	ON	ON	OK
Lamp Red	1	ON	ON	OK
	2	ON	ON	OK
	3	ON	ON	OK
	4	ON	ON	OK
	5	ON	ON	OK

Berdasarkan Tabel II yang diberikan, dapat disimpulkan bahwa pengujian nilai yang dikendalikan pada ESP32 dan dikirimkan ke PLC dapat terbaca secara *realtime* dan semua pengiriman data berhasil.

D. Pengujian pengiriman data dari Power meter PM2100 ke ESP32

Pengujian pengiriman data dari *power meter* PM2100 ke ESP32 ditujukan untuk mengamati pengujian data parameter yang dikirim berhasil terbaca atau tidak dengan cara monitoring menggunakan Arduino IDE. Tabel III berikut merupakan penunjukan data parameter monitoring ESP32 pada *power meter* PM2100.

TABLE III
PENGIRIMAN DATA DARI *POWER METER* PM2100 KE ESP32

Elemen	Percobaan ke-	*Input*	Monitor *Power Meter*	Monitor ESP32	Status
Tegangan AC (monitor)	1	41.9VAC	41.9VAC	41.89VAC	OK
	2	42.9VAC	42.9VAC	42.87VAC	OK
	3	59.4VAC	59.4VAC	59.43VAC	OK
	4	88.8VAC	88.8VAC	88.80VAC	OK
	5	112.5VAC	112.5VAC	112.50VAC	OK
	1	132.0VAC	132.0VAC	132.0VAC	OK
	2	165.1VAC	165.1VAC	165.10VAC	OK
	3	184.0VAC	184.0VAC	184.0VAC	OK
	4	199.3VAC	199.3VAC	199.30VAC	OK
	5	199.3VAC	199.3VAC	199.30VAC	OK

Dari Tabel III yang telah dilampirkan, dapat disimpulkan bahwa pengujian nilai parameter voltage yang ada pada *power meter* PM2100 dapat terbaca oleh ESP32 secara *realtime*, dan semua pengiriman data berhasil.

E. Pengujian pengiriman data dari input PLC ke web

Pengujian pengiriman data dari *input* PLC ke *web based* ditujukan untuk mengamati pengujian data yang dikirim berhasil terbaca atau tidak pada *web* menggunakan jaringan baik dan buruk. Tabel IV, V, dan VI merupakan penunjukan data *monitoring web* pada *input* PLC.

TABLE IV
HASIL MONITORING PEMBACAAN NILAI PADA WEB DARI PLC

Elemen		Percobaan ke-	Monitor PLC	Monitor Aplikasi *web*
			Internet Baik di bawah 200 ms	
Maksimal *Push Button* dihidupkan selama 2 detik	*Push Button Green*	1	ON	ON
		2	ON	ON
		3	ON	ON
		4	ON	ON
		5	ON	ON
		6	ON	ON
		7	ON	ON
		8	ON	ON
		9	ON	ON
		10	ON	ON
	Push Button Red	1	ON	ON
		2	ON	ON
		3	ON	ON
		4	ON	ON
		5	ON	ON
		6	ON	ON
		7	ON	ON
		8	ON	ON
		9	ON	ON
		10	ON	ON
Tingkat keberhasilan			100%	

TABLE V
HASIL MONITORING PEMBACAAN NILAI PADA WEB DARI PLC

Elemen		Percobaan ke-	Monitor PLC	Monitor Aplikasi *web*
			Internet Baik di atas 500 ms	
Maksimal *Push Button* dihidupkan selama 2 detik	*Push Button Green*	1	ON	ON
		2	ON	ON
		3	ON	OFF
		4	ON	ON
		5	ON	OFF
		6	ON	OFF
		7	ON	ON
		8	ON	ON
		9	ON	ON
		10	ON	OFF
	Push Button Red	1	ON	OFF
		2	ON	ON
		3	ON	ON
		4	ON	OFF
		5	ON	ON
		6	ON	ON
		7	ON	OFF
		8	ON	OFF
		9	ON	ON
		10	ON	ON
Tingkat keberhasilan			60%	

TABLE VI
HASIL MONITORING PEMBACAAN NILAI PADA WEB DARI PLC

Elemen		Percobaan ke-	Monitor PLC	Monitor aplikasi *web*
			Internet buruk di atas 800 ms	
Maksimal *Push Button* dihidupkan selama 2 detik	*Push Button Green*	1	ON	OFF
		2	ON	OFF
		3	ON	OFF
		4	ON	OFF
		5	ON	OFF
		6	ON	OFF
		7	ON	OFF
		8	ON	OFF
		9	ON	OFF
		10	ON	OFF
	Push Button Red	1	ON	OFF
		2	ON	OFF
		3	ON	OFF
		4	ON	OFF
		5	ON	OFF
		6	ON	OFF
		7	ON	OFF
		8	ON	OFF
		9	ON	OFF
		10	ON	OFF
Tingkat keberhasilan			0%	

Dari Tabel IV, V, dan VI di atas pengujian pengiriman data dari *input* PLC ke *web* ditampilkan untuk mengamati data yang dikirim berhasil terbaca atau tidak, pada *web* menggunakan jaringan baik dan buruk. pada pengujian ini terdapat tiga *mode* jaringan:

1. Jaringan baik dengan kecepatan internet di bawah 200 ms, terdapat monitoring *input push button* dengan jeda waktu *on* selama 2 detik, mendapatkan tingkat keberhasilan 100% pada sinkronisasi PLC ke *web*
2. Jaringan dengan kecepatan internet di atas 500 ms, mendapatkan tingkat keberhasilan 60%, penyebab 40% gagal karena jaringan yang tidak stabil.
3. Jaringan dengan kecepatan internet di atas 800 ms, mendapatkan tingkat keberhasilan 0%, dikarenakan jaringan yang sangat buruk dan *delay* yang sangat lama.

Berdasarkan Tabel IV, V, dan VI yang telah dilampirkan, dapat disimpulkan bahwa pengujian pengiriman data dari PLC ke *web* melalui ESP32 dapat terbaca secara *realtime* ketika jaringan dalam kondisi di bawah 200 ms. Terdapat perbedaan dalam beberapa sampel data antara PLC dan *web*, yang mana kesalahan tersebut disebabkan oleh jaringan *wifi* yang memiliki waktu respon di atas 500 ms. Hal ini mengakibatkan kegagalan dalam pengiriman data ke *web*.

F. Pengujian pengiriman data dari web ke Output PLC

Pengujian pengiriman data dari *web* based ke *output* PLC *input* PLC ke *web* Based ditujukan untuk mengamati pengujian data yang dikirim berhasil terbaca atau tidak pada PLC menggunakan jaringan baik dan buruk. Tabel VII, VIII, IX merupakan penunjukkan data *controlling web based* pada *output* PLC beserta kecepatan jaringan internet baik atau buruknya.

TABLE VII
HASIL CONTROLING PEMBACAAN NILAI PADA WEB BASED DARI PLC

Elemen		Percobaan ke-	Monitor PLC	Monitor Aplikasi *web*
			Internet Baik di bawah 200 ms	
Maksimal *Pilot Lamp* dihidupkan selama 2 detik	*Pilot Lamp Green*	1	ON	ON
		2	ON	ON
		3	ON	ON
		4	ON	ON
		5	ON	ON
		6	ON	ON
		7	ON	ON
		8	ON	ON
		9	ON	ON
		10	ON	ON
	Pilot Lamp Red	1	ON	ON
		2	ON	ON
		3	ON	ON
		4	ON	ON
		5	ON	ON
		6	ON	ON
		7	ON	ON
		8	ON	ON
		9	ON	ON
		10	ON	ON
Tingkat keberhasilan			100%	

TABLE VIII
HASIL CONTROLING PEMBACAAN NILAI PADA WEB BASED DARI PLC

Elemen		Percobaan ke-	Monitor PLC	Monitor Aplikasi *web*
			Internet Baik di bawah 200 ms	
Maksimal *Pilot Lamp* dihidupkan selama 2 detik	*Pilot Lamp Green*	1	ON	OFF
		2	ON	ON
		3	ON	ON
		4	ON	OFF
		5	ON	OFF
		6	ON	ON
		7	ON	OFF
		8	ON	ON
		9	ON	ON
		10	ON	ON
	Pilot Lamp Red	1	ON	OFF
		2	ON	OFF
		3	ON	ON
		4	ON	ON
		5	ON	ON
		6	ON	OFF
		7	ON	OFF
		8	ON	OFF
		9	ON	ON
		10	ON	ON
Tingkat keberhasilan			55%	

TABLE IX
HASIL CONTROLING PEMBACAAN NILAI PADA WEB BASED DARI PLC

Elemen		Percobaan ke-	Monitor PLC	Monitor Aplikasi *web*
			Internet Baik di bawah 200 ms	
Maksimal *Pilot Lamp* dihidupkan selama 2 detik	*Pilot Lamp Green*	1	ON	OFF
		2	ON	OFF
		3	ON	OFF
		4	ON	OFF
		5	ON	OFF
		6	ON	OFF
		7	ON	OFF
		8	ON	OFF
		9	ON	OFF
		10	ON	OFF
	Pilot Lamp Red	1	ON	OFF
		2	ON	OFF
		3	ON	OFF
		4	ON	OFF
		5	ON	OFF
		6	ON	OFF
		7	ON	OFF
		8	ON	OFF
		9	ON	OFF
		10	ON	OFF
Tingkat keberhasilan			0%	

Tujuan dari pengujian ini adalah untuk memeriksa apakah data yang dikirim dari *web* ke *output* PLC bisa terbaca atau tidak pada PLC menggunakan jaringan yang baik maupun buruk. pada pengujian ini terdapat tiga mode jaringan.

Hasil *controling* pembacaan nilai pada *web* Based dari PLC.

1. Jaringan baik dengan kecepatan internet di bawah 200 ms, terdapat monitoring *output* pilot *lamp* dengan menghidupkan dari *web* selama 2 detik, mendapatkan tingkat keberhasilan 100% pada sinkronisasi PLC ke *web*.
2. Jaringan dengan kecepatan internet di atas 500 ms, mendapatkan tingkat keberhasilan 55%, penyebab 45% gagal karena jaringan yang tidak stabil.
3. Jaringan dengan kecepatan internet di atas 800 ms, mendapatkan tingkat keberhasilan 0%, dikarenakan jaringan yang sangat buruk dan *delay* yang sangat lama.

Berdasarkan Tabel VII, VIII, dan IX dapat disimpulkan bahwa kontrol yang diuji pada *web* dan dikirim ke PLC melalui ESP32 dapat dibaca dalam waktu *realtime* jika dalam kondisi jaringan di bawah 200 ms. Terdapat beberapa perubahan data sampling yang tidak sama antara PLC dan *web*, *error* tersebut disebabkan oleh jaringan *wifi* yang respon waktunya melebihi 500 ms, sehingga mempengaruhi kegagalan pengiriman data ke PLC.

G. Pengujian pengiriman data dari Power meter ke web

Pengujian pengiriman data pada *Power Meter* dilakukan dengan menggunakan SCR *speed control* 4000 W sebagai pengubah nilai tegangan. Tujuan dari pengujian ini adalah untuk mengamati apakah data dapat diterima atau tidak, pada aplikasi berbasis *web*. Tabel X, XI, XII merupakan penunjukkan data *monitoring web* pada *output power meter* beserta kecepatan jaringan internet baik atau buruknya.

TABLE X
HASIL MONITORING PEMBACAAN NILAI PADA *WEB* DARI *POWER METER*

Elemen		Percobaan ke-	*Power Meter*	Monitor aplikasi *web*
			Internet baik di bawah 200 ms	
Maksimal *delay time* pengiriman data selama 2 detik	Tegangan AC (monitor)	1	46.10VAC	46.10VAC
		2	61.84VAC	61.84VAC
		3	79.25VAC	79.25VAC
		4	127.6VAC	127.6VAC
		5	211.9VAC	211.9VAC
		6	38.83VAC	38.83VAC
		7	57.71VAC	57.71VAC
		8	78.69VAC	78.69VAC
		9	123.8VAC	123.8VAC
		10	215.1VAC	215.1VAC
Tingkat keberhasilan			100%	

TABLE XI
HASIL MONITORING PEMBACAAN NILAI PADA WEB DARI *POWER METER*

Elemen		Percobaan ke-	*Power Meter*	Monitor aplikasi *web*
			Internet baik di atas 500 ms	
Maksimal *delay* time pengiriman data selama 2 detik	Tegangan AC (monitor)	1	39.20VAC	39.20VAC
		2	44.2VAC	44.2VAC
		3	50.5VAC	47.10VAC
		4	80.1VAC	80.1VAC
		5	177.2VAC	177.2VAC
		6	50.3VAC	50.3VAC
		7	80.7VAC	65.22VAC
		8	101.2VAC	86.6VAC
		9	154.9VAC	154.9VAC
		10	201.4VAC	201.4VAC
Tingkat keberhasilan			70%	

TABLE XII
HASIL MONITORING PEMBACAAN NILAI PADA WEB DARI *POWER METER*

Elemen		Percobaan ke-	*Power Meter*	Monitor aplikasi *web*
			Internet buruk di atas 800 ms	
Maksimal *delay* time pengiriman data selama 2 detik	Tegangan AC (monitor)	1	46.10VAC	10.1VAC
		2	61.84VAC	51.8VAC
		3	79.25VAC	72.3VAC
		4	127.6VAC	81.9VAC
		5	211.9VAC	147.2VAC
		6	38.83VAC	15.8VAC
		7	57.71VAC	41.6VAC
		8	78.69VAC	62.1VAC
		9	123.8VAC	103.2VAC
		10	215.1VAC	173.7VAC
Tingkat keberhasilan			0%	

Pada pengujian ini terdapat tiga mode jaringan:

1. Jaringan baik dengan kecepatan internet di bawah 200 MS, terdapat monitoring nilai volt pada *power meter* ke *web* dengan jeda waktu sinkronisasi selama 2 detik, mendapatkan tingkat keberhasilan 100% pada sinkronisasi PLC ke *web*.
2. Jaringan dengan kecepatan internet di atas 500 ms, mendapatkan tingkat keberhasilan 70%, penyebab 30% gagal karena jaringan yang tidak stabil.
3. Jaringan dengan kecepatan internet diatas 800 ms, mendapatkan tingkat keberhasilan 0%, dikarenakan jaringan yang sangat buruk dan *delay* yang sangat lama.

Berdasarkan Tabel X, XI, dan XII, dapat disimpulkan bahwa pengujian pengiriman data nilai tegangan pada *power meter* dan dikirimkan ke *web* melalui ESP32 dapat terbaca secara *realtime* dalam kondisi jaringan dengan respon waktu di bawah 200 ms. Namun, terdapat beberapa perbedaan pada data antara *power meter* dan *web*. Perbedaan ini disebabkan oleh jaringan *wifi* dengan waktu respon di atas 500 ms. Akibatnya, pengiriman data ke *web based* menjadi gagal.

H. Pengujian packet loss jaringan

Pada pengujian ini, kita perlu menggunakan aplikasi Wireshark. Untuk langkah pertama, kita perlu membuka aplikasi terlebih dahulu. Setelah itu, *capture* jaringan dan masukkan perintah "tcp && ip.dst==185.201.9.130 && ip.src==192.168.43.108" pada kolom *filter* di halaman aplikasi. Kemudian, tekan tombol enter agar Wireshark hanya menampilkan data dari jaringan protokol TCP dan MQTT. Waktu yang digunakan yaitu 8 kali percobaan pada 1 hari yang ditunjukkan pada Tabel XIII.

TABLE XIII
WAKTU PENGUJIAN *PACKET LOSS*

No.	Waktu ke	Lama Pengujian
1	1	08.00 - 08.10 WIB
2	2	09.00 - 09.10 WIB
3	3	10.00 - 10.10 WIB
4	4	11.00 - 11.10 WIB
5	5	12.00 - 12.10 WIB
6	6	13.00 - 13.10 WIB
7	7	14.00 - 14.10 WIB
8	8	15.00 - 15.10 WIB

Setelah mengamati sesuai dengan skenario yang telah dibuat, ditemukan nilai *packet loss* yang tercatat dan ditampilkan dalam aplikasi Wireshark, hasil tersebut kemudian dipaparkan di dalam Tabel XIV.

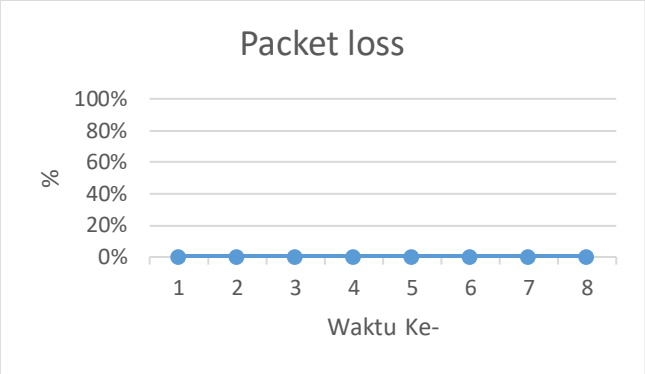
TABLE XIV
HASIL DATA PACKET LOSS DARI *SOFTWARE* WIRESHARK

Waktu ke-	*Captured*	*Displayed*	*Packet* loss *rate*
1	1818	1818	0%
2	1764	1764	0%
3	1666	1666	0%
4	1875	1875	0%
5	1789	1789	0%
6	1735	1735	0%
7	1766	1766	0%
8	1809	1809	0%

Nilai *packet loss rate* didapat menggunakan rumus *packet loss rate*

$$Packet\ loss\ rate = (total\ packet\ loss / total\ packet\ sent) * 100\ \% \quad (1)$$

Tidak terjadi kehilangan data yang ditandai pada parameter *captured* and *displayed* pada perangkat lunak Wireshark, disajikan pada Tabel XIV, menunjukkan keandalan sistem komunikasi. Jaringan ini menunjukkan kinerja yang sangat baik dan mendapatkan nilai indeks 4 sesuai dengan ketentuan dari TIPHON.



Gambar. 10. Grafik pengujian packet loss

Jaringan protokol ini memiliki tingkat keandalan yang tinggi, diindikasikan dengan tidak adanya paket data yang hilang ditampilkan pada Tabel XIV dimana parameter *captured* dan *displayed* memiliki nilai yang sama dan memiliki *rate* 0% di setiap waktu percobaan seperti yang ditampilkan dalam Gambar. 10. Oleh karena itu, jaringan ini cocok untuk digunakan dalam modul IoT SCADA yang membutuhkan transmisi data dengan tingkat data hilang yang rendah.

IV. KESIMPULAN DAN SARAN

A. Kesimpulan

Dalam rangkaian pengujian dan implementasi yang dilakukan, dapat disimpulkan bahwa sistem komunikasi antara perangkat seperti PLC, *Power Meter*, ESP32, dan server berjalan 100% *realtime* dengan *delay* internet dibawah 500MS. Data berhasil dikirim dan diterima dalam berbagai situasi, termasuk jaringan yang baik dan jaringan yang buruk Selama pengujian, tidak terjadi kehilangan data yang ditandai pada parameter *captured and displayed* pada perangkat lunak Wireshark, menunjukkan keandalan sistem komunikasi. Desain protokol komunikasi IoT SCADA, yang memanfaatkan RS485 untuk menghubungkan PLC dan *Power Meter*, serta menggunakan protokol MQTT melalui TCP/IP untuk komunikasi dengan server, terbukti efektif. Keseluruhan sistem berfungsi sesuai yang diharapkan, menjadikannya solusi yang handal untuk pengiriman dan penerimaan data dalam lingkungan yang beragam.

B. Saran

Dari kesimpulan di atas maka penelitian ini disarankan untuk mengembangkan protokol keamanan agar terhindar dari adanya pencurian data, serta lebih banyak menggunakan fitur desain yang menarik agar ramah terhadap pengguna, dan kemudian mengembangkan fitur penyimpanan data pada database agar sistem ini bisa lebih canggih untuk digunakan.

REFERENCES

[1] A. Abdullah, C. Cholish, and M. Zainul haq, "Pemanfaatan IoT (Internet of Things) Dalam Monitoring Kadar Kepekatan Asap dan Kendali Pergerakan Kamera," *CIRCUIT J. Ilm. Pendidik. Tek. Elektro*, vol. 5, no. 1, p. 86, Feb. 2021, doi: 10.22373/crc.v5i1.8497.

[2] N. Widyaningrum, S. Pemantauan, and U. Y. Oktiawati, "Sistem Pemantauan dan Pengendalian Debit Fluida Berbasis Arduino dan *website* (Monitoring and Controling Fluid System Based on Arduino and *website*)," 2020.

[3] F. Ciasaka, S. D. Panjaitan, and B. W. Sanjaya, "Perancangan Sistem Kendali Supervisi Dan Akuisisi Data (Scada) Pada Panel Surya Berbasis Internet of Things," *J. Tek. Elektro Univ. ...*, [Online]. Available: https://jurnal.untan.ac.id/index.php/jteuntan/article/view/62649%0Ahttps://jurnal.untan.ac.id/index.php/jteuntan/article/viewFile/62649/75676596399

[4] S. W. Jadmiko, D. N. Suharno, and S. K. Nugraha, "Aplikasi Internet of Things (IoT) untuk Pemantauan Simulator Plant Berbasis PLC – *web* Server," *... Teknol. dan Ris. ...*, pp. 28–33, 2020, [Online]. Available: https://semnastera.polteksmi.ac.id/index.php/semnastera/article/view/85

[5] F. Ardiansyah, M. F. Lawasi, and C. F. Hadi, "Sistem Monitoring Inkubator Penetas Telur Berbasis Android," *Zetroem*, vol. 01, pp. 8–16, 2019.

[6] M. Choir Afrian, M. T. Asron, R. Wicaksono,) Diii, and T. Elektronika, "Prototipe Pengangkut Sampah Otomatis Pada Pintu Ali Dengan Sistem Informasi Menggunakan Node Mcu Esp8266 Berbasis Plc (Programmable Logic Controller)", doi: 10.21009/autocracy.05.2.5.

[7] Angga Wahyu Aditya, Nur Rani Alham, Restu Mukti Utomo, Hilmansyah, "Sistem Pemantauan KonsuMSi Energi Listrik Berbasis *web* Sebagai Upaya Konservasi Energi" 2023.

[8] Pricillia Alvina, Handy Wicaksono, Petrus Santoso "Sistem Keamanan Bangunan Multi Lokasi Berbasis IoT Menggunakan Siemens LOGO! dan Raspberry Pi" 2019.

[9] A. G. Priambodo, "Rancang Bangun Protokol Komunikasi On Board Unit (Obu) Untuk Intelligent Transport Sytem (Its) Di Surabaya."

[10] Mus Mulyadi Usman, Xaverius B.N. Najoan, Meicsy E. I. Najoan, "Rancang Bangun Aplikasi Monitoring Ketinggian Air Sungai Berbasis Internet of Things Menggunakan Amazon *web* Service"2020.