Jurnal Riset Multidisiplin Edukasi
https://journal.hasbaedukasi.co.id/index.php/jurmie
Halaman : 1584 - 1601

PENERAPAN INTERNET OF THINGS (IOT) PADA ROBOT PELAYANAN MENGGUNAKAN MQTT (MESSAGE QUEUING TELEMETRY TRANSPORT) UNTUK MONITORING REALTIME

Harry Abi Bayu[1], Mhd. Zulfansyuri Siambaton[2], Aulia Ichsan[3]
Prodi Teknik Informatika Fakultas Teknik Universitas Islam Sumatera Utara Medan-Indonesia [1,2,3]
Email: harryabibayu@gmail.com[1], zulfansyuri@ft.isu.ac.id[2], auliaichsan15@gmail.com[3]

Informasi

Volume : 3
Nomor : 6
Bulan : Juni
Tahun : 2026
E-ISSN : 3062-9624

Abstract

Internet of Things (IoT) technology has driven innovation in automation systems, including service-based robotic applications. This research aims to design and implement an IoT-based service robot system capable of receiving real-time commands and performing automatic navigation to designated locations. The system is developed using multiple ESP32 microcontrollers interconnected via a Wi-Fi network and communicating through the Message Queuing Telemetry Transport (MQTT) protocol. The research methodology includes system architecture design, hardware and software implementation, and system testing. The robot navigation is designed using a Finite State Machine (FSM) approach based on time-based navigation, while obstacle detection is handled using an ultrasonic sensor. A web-based user interface is implemented to send commands and monitor the robot's status in real time. The results show that the MQTT-based communication system is able to transmit and receive data reliably with delay values within acceptable limits for real-time applications. Furthermore, the robot successfully navigates to the target tables and returns to the starting position according to the designed system, while also responding effectively to obstacles detected along its path.In conclusion, this study demonstrates that the integration of IoT, FSM-based control, and time-based navigation can be effectively applied to a service robot system with a high level of operational success.

Keyword: *Internet of Things, ESP32, MQTT, Service Robot, Finite State Machine*

Abstrak

Teknologi Internet of Things (IoT) telah mendorong inovasi dalam sistem otomasi, termasuk pada bidang pelayanan berbasis robot. Penelitian ini bertujuan untuk merancang dan mengimplementasikan sistem robot pelayan berbasis IoT yang mampu menerima perintah secara realtime serta melakukan navigasi menuju lokasi tujuan secara otomatis. Sistem dikembangkan menggunakan beberapa modul ESP32 yang terintegrasi melalui jaringan Wi-Fi dengan protokol komunikasi Message Queuing Telemetry Transport (MQTT). Metode yang digunakan dalam penelitian ini meliputi perancangan arsitektur sistem, implementasi perangkat keras dan perangkat lunak, serta pengujian sistem. Navigasi robot dirancang menggunakan pendekatan Finite State Machine (FSM) berbasis waktu (time-based navigation), sedangkan deteksi hambatan dilakukan menggunakan sensor ultrasonik. Antarmuka pengguna berbasis web digunakan untuk mengirimkan perintah serta melakukan monitoring kondisi robot secara realtime. Hasil pengujian menunjukkan bahwa sistem komunikasi MQTT mampu mengirim dan menerima data dengan baik dengan delay yang masih dalam batas toleransi sistem realtime. Selain itu, robot berhasil menjalankan navigasi menuju meja tujuan dan kembali ke posisi awal sesuai dengan perancangan, serta mampu merespons keberadaan hambatan di jalur pergerakan.Dengan demikian, penelitian ini membuktikan bahwa integrasi IoT, metode FSM, dan sistem navigasi berbasis waktu dapat diterapkan secara efektif pada robot pelayan sederhana dengan tingkat keberhasilan operasional yang tinggi.

Kata Kunci: *Internet of Things, ESP32, MQTT, Robot Pelayan, Finite State Machine*

A. PENDAHULUAN

Teknologi Internet of Things (IoT) telah membawa perubahan signifikan dalam berbagai bidang, khususnya pada sistem otomasi dan integrasi perangkat cerdas. Menurut IoT memungkinkan perangkat fisik untuk saling terhubung, bertukar data, serta dikendalikan dan di pantau secara realtime melalui jaringan internet tanpa memerlukan interaksi manusia secara langsung (Selay et al., 2022) (Munawar et al., 2023). Dengan kemampuan tersebut, IoT banyak diterapkan pada berbagai sektor, seperti industri, kesehatan, rumah pintar, serta sistem otomasi berbasis robotika.

Pada bidang robotika, penerapan IoT berperan penting dalam meningkatkan fleksibilitas dan kemampuan sistem robot agar tidak hanya bekerja secara lokal, tetapi juga dapat dipantau dan dikendalikan dari jarak jauh melalui jaringan internet (Annur et al., 2021). Integrasi IoT pada sistem robot memungkinkan proses pengiriman perintah, pemantauan status, serta pengambilan data dilakukan secara terpusat dan realtime. Hal ini menjadikan robot lebih adaptif terhadap kebutuhan lingkungan dan pengguna.

Salah satu bentuk penerapan robotika yang berkembang pesat adalah robot pelayanan. Robot pelayanan dirancang untuk membantu aktivitas pelayanan, seperti pengantaran makanan atau barang, khususnya pada lingkungan indoor seperti restoran, rumah sakit, dan fasilitas pelayanan lainnya. Penggunaan robot pelayan dapat meningkatkan efisiensi pelayanan, mengurangi beban kerja manusia, serta memberikan konsistensi dalam proses layanan (Amrulloh & Kurniadi, 2022), (Ramadhan, A., Tamsir Ariyadi, 2023). Meskipun demikian, beberapa penelitian dan implementasi robot pelayan yang telah dikembangkan masih memiliki keterbatasan, antara lain sistem kendali yang bersifat lokal, komunikasi yang belum terintegrasi secara optimal, serta keterbatasan dalam pemantauan kondisi robot secara realtime. Keterbatasan tersebut menyebabkan robot kurang fleksibel dan sulit dikembangkan untuk kebutuhan pelayanan yang dinamis dan terintegrasi dengan sistem jaringan (Rofi et al., 2025).

Untuk mengatasi permasalahan tersebut, diperlukan sebuah sistem robot pelayan yang terintegrasi dengan teknologi Internet of Things (IoT) dan didukung oleh protokol komunikasi yang efisien. Salah satu protokol komunikasi yang banyak digunakan pada sistem IoT adalah Message Queuing Telemetry Transport (MQTT). MQTT dirancang sebagai protokol komunikasi ringan dengan mekanisme publish–subscribe yang mampu mendukung pertukaran data secara

cepat dan realtime (Rohman et al., 2021), (Fadhilah et al., 2023). Protokol ini sangat sesuai untuk sistem robot pelayan yang membutuhkan komunikasi yang responsif dan stabil. Selain protokol komunikasi, pemilihan perangkat kendali juga menjadi faktor penting dalam pengembangan robot pelayan berbasis IoT. Mikrokontroler ESP32 merupakan salah satu perangkat yang banyak digunakan karena telah dilengkapi dengan modul Wi-Fi dan memiliki kemampuan pemrosesan yang memadai untuk mendukung sistem IoT dan robotika (Suhardi et al., 2022), (Suhaeb et al., 2024). Dengan menggunakan ESP32, sistem robot pelayan dapat diintegrasikan langsung dengan jaringan internet untuk proses kendali dan monitoring.

Pada penelitian ini, robot pelayan dirancang menggunakan arsitektur sistem terdistribusi dengan memanfaatkan beberapa unit mikrokontroler ESP32 yang saling terhubung melalui protokol MQTT. Pemisahan fungsi kendali, navigasi, dan aktuator dilakukan untuk meningkatkan stabilitas sistem, responsivitas komunikasi, serta keandalan operasional robot dalam menjalankan tugas pengantaran pada lingkungan indoor terbatas. Sistem ini memungkinkan robot menerima perintah secara realtime, melakukan navigasi dasar dengan

Berdasarkan latar belakang penelitian, rumusan masalah dalam penelitian ini berfokus pada bagaimana merancang robot pelayan berbasis Internet of Things (IoT) yang mampu membantu proses pengantaran di lingkungan indoor terbatas, mengembangkan sistem kendali dan komunikasi menggunakan protokol MQTT agar proses pengiriman perintah serta pemantauan status robot dapat dilakukan secara realtime dan terdistribusi, serta menganalisis cara kerja robot dalam menjalankan fungsi pengantaran berdasarkan hasil pengujian. Agar penelitian lebih terarah, ruang lingkup penelitian dibatasi pada perancangan dan pembuatan prototipe robot pelayan, bukan pengembangan sistem komersial, dengan pengoperasian di lingkungan indoor terbatas yang memiliki jalur telah ditentukan, menggunakan sistem navigasi sederhana tanpa pemetaan lingkungan maupun algoritma navigasi kompleks, serta menerapkan konsep IoT melalui protokol MQTT sebagai media komunikasi dan pertukaran data. Sejalan dengan itu, tujuan penelitian ini adalah merancang dan membangun prototipe robot pelayan berbasis IoT untuk membantu proses pengantaran di lingkungan indoor, mengembangkan sistem kendali dan komunikasi berbasis MQTT sehingga robot mampu menerima perintah dan mengirimkan informasi status secara realtime, serta melakukan pengujian terhadap kinerja robot dalam menjalankan fungsi pengantaran sesuai skenario yang telah ditetapkan.

B. METODE PENELITIAN

Penelitian ini menggunakan metode rancang bangun (engineering design) untuk merancang, membangun, dan menguji prototipe robot pelayan berbasis Internet of Things (IoT) yang mampu membantu proses pengantaran di lingkungan indoor. Tahapan penelitian meliputi analisis kebutuhan sistem, perancangan perangkat keras dan perangkat lunak, implementasi, hingga pengujian kinerja sistem. Perangkat keras yang digunakan terdiri atas ESP32 sebagai mikrokontroler utama, sensor ultrasonik HC-SR04, motor driver BTS7960, motor DC gear, servo, LCD I2C, DFPlayer Mini, baterai lithium, serta modul pendukung lainnya. Sementara itu, perangkat lunak yang digunakan meliputi Arduino IDE untuk pemrograman, HiveMQ Cloud sebagai broker MQTT, Laragon sebagai server web, dan aplikasi monitoring berbasis web. Arsitektur sistem dibangun menggunakan tiga unit ESP32 yang memiliki fungsi berbeda, yaitu ESP32 Remote sebagai pengirim perintah melalui antarmuka web, ESP32 Navigasi sebagai pengendali pergerakan robot menggunakan metode Finite State Machine (FSM), dan ESP32 Aktuator yang mengendalikan servo, LCD, serta modul audio. Seluruh unit saling berkomunikasi melalui protokol MQTT dengan mekanisme publish–subscribe sehingga proses kendali dan pemantauan robot dapat dilakukan secara realtime dan terintegrasi.

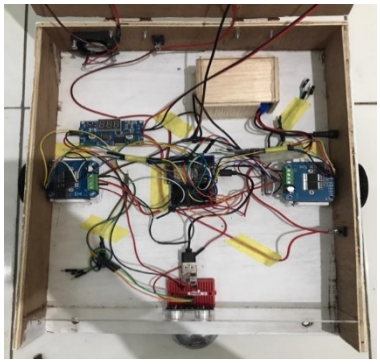
Perancangan perangkat lunak difokuskan pada sistem komunikasi MQTT, monitoring realtime, logika navigasi berbasis FSM, serta mekanisme penanganan hambatan. Komunikasi data dilakukan melalui tiga topik utama, yaitu robot/command, robot/status, dan robot/event, yang memungkinkan pertukaran perintah, status, dan notifikasi antarperangkat secara efisien. Sistem navigasi dirancang menggunakan FSM dengan beberapa state, seperti IDLE, GO_TABLE, WAIT_RETURN, BACK_COUNTER, dan OBSTACLE, sehingga robot mampu bergerak secara terstruktur sesuai jalur yang telah ditentukan. Sensor ultrasonik digunakan untuk mendeteksi hambatan dengan batas jarak 30 cm; apabila terdeteksi penghalang, robot akan berhenti sementara, mengirimkan notifikasi melalui MQTT, kemudian melanjutkan perjalanan setelah jalur kembali aman atau membatalkan misi jika hambatan berlangsung lebih dari 60 detik. Pengujian sistem dilakukan terhadap konektivitas MQTT, fungsi monitoring realtime, performa navigasi, dan mekanisme penanganan hambatan untuk memastikan seluruh komponen perangkat keras maupun perangkat lunak bekerja sesuai dengan rancangan serta menghasilkan sistem robot pelayan yang andal, responsif, dan mampu beroperasi secara otomatis.

C. HASIL DAN PEMBAHASAN

Hasil Perancangan Fisik Robot

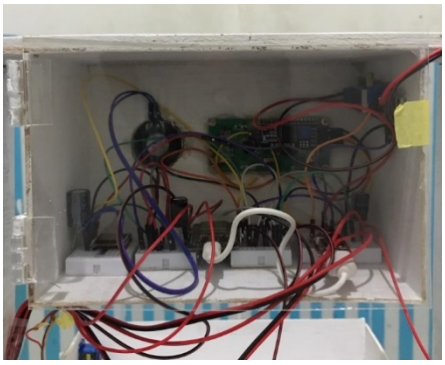
Gambar 1 Bentuk Fisik Robot Pelayanan Hasil Implementasi

Bagian robot navigasi dikendalikan oleh ESP32 Navigasi yang terhubung dengan driver motor dan motor DC sebagai penggerak roda. Sensor ultrasonik dipasang pada bagian depan robot dan digunakan untuk mendeteksi adanya hambatan di jalur pergerakan robot. Data jarak yang diperoleh dari sensor tersebut digunakan sebagai masukan untuk sistem navigasi dalam menentukan tindakan yang harus dilakukan oleh robot.



Gambar 2 Penempatan sensor dan sistem penggerak robot

Selain sistem navigasi, robot juga dilengkapi dengan sistem aktuator yang dikendalikan oleh ESP32 Aktuator. Sistem ini terdiri dari motor servo yang digunakan untuk membuka dan menutup rak makanan, modul audio sebagai pemberi notifikasi suara, serta LCD sebagai media penapil informasi. Komponen-komponen tersebut memungkinkan robot memberikan umpan balik visual dan audio kepada pengguna selama proses pelayanan berlangsung.



Gambar 3 Implementasi sistem aktuator pada robot pelayan

Pengujian Konektivitas dan Komunikasi IoT

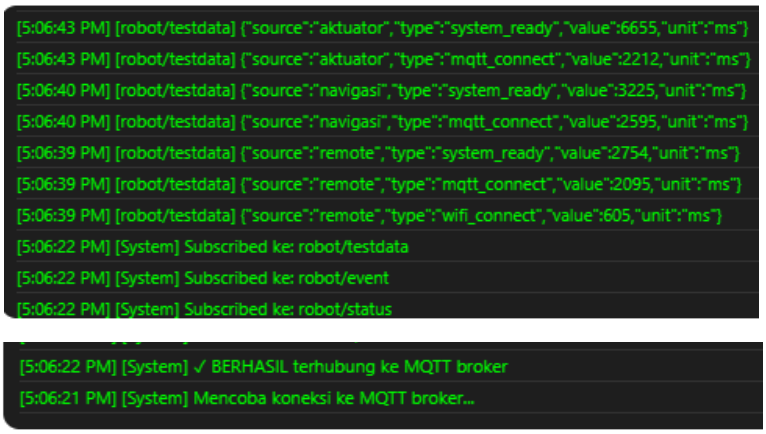
Pengujian konektivitas dilakukan untuk memastikan seluruh unit ESP32 (Remote, Navigasi, Aktuator) dapat terhubung ke jaringan WiFi dan berkomunikasi menggunakan protokol MQTT melalui broker *HiveMQ Cloud*.

Gambar dibawah ini menunjukkan apabila Robot belum terkoneksi dengan komunikasi IoT yaitu MQTT maka tampilan LCD pada Robot "Robot Waiter Booting".

Gambar 4 Kondisi Robot Ketika Belum Terhubung

Gambar dibawah ini menunjukan apabila Robot sudah terkoneksi dengan MQTT maka tampilan di LCD akan menamplikan "

Gambar 5 Kondisi Robot Berhasil Terhubung



Gambar 6 Log MQTT pada *Dashboard* Saat Koneksi Berhasil

Tabel 1 Hasil Pengujian Konektivitas IoT

No	**Skenario Pengujian**	**Percobaan 1 (ms)**	**Percobaan 2 (ms)**	**Percobaan 3 (ms)**	**Rata-rata (ms)**	**Status**
1.	Koneksi WiFi ESP32 *Remote*	3245	3187	3302	3245	Berhasil
2.	Koneksi MQTT ESP32 *Remote*	238	245	252	245	Berhasil
3.	Koneksi WiFi ESP32 Navigasi	3562	3489	3621	3557	Berhasil
4.	Koneksi MQTT ESP32 Navigasi	252	248	258	253	Berhasil
5.	Koneksi WiFi ESP32 Aktuator	3412	3356	3478	3415	Berhasil
6.	Koneksi MQTT ESP32 Aktuator	258	262	255	258	Berhasil
No	**Skenario Pengujian**	**Kondisi Jaringan**	**Hasil Jaringan yang Diharapkan**	**Hasil Pengujian**		**Status**
6.	Memutus koneksi Wi-Fi secara sengaja saat robot	Terputus (*Disconnect*)	Serial Monitor menampilkan status kegagalan koneksi ke	ESP32 gagal melakukan *publish/subscribe*, sistem *stuck* pada *state* terakhir sebelum terputus.		Tidak Berhasil

	sedang siaga		MQTT broker dan mencoba *reconnect*.			
7.	Membawa robot ke area *blind spot* (jarak > 20 meter dari router)	Sinyal Lemah (*RSSI < -85 dBm*)	Penundaan (*delay*) pengiriman data monitoring meningkat signifikan.	Terjadi *delay* pengiriman data posisi robot ke *interface* web selama lebih dari 5 detik.		Tidak Berhasil

Tabel 2 Pengukuran *Delay* Komunikasi MQTT

No	**Jenis Komunikasi**	**Percobaan 1 (ms)**	**Percobaan 2 (ms)**	**Percobaan 3 (ms)**	**Rata-rata (ms)**
1.	Remote → Navigasi (perintah)	115	118	112	115
2.	Navigasi → *Remote* (status)	98	102	105	102
3.	Navigasi → Aktuator (event)	105	110	108	108
4.	Remote → Navigasi (go_counter)	118	115	120	118
Rata-rata Keseluruhan					111

Pengujian konektivitas dilakukan dengan mengamati proses *startup* ketiga ESP32. Gambar 4.4 menunjukkan LCD pada robot saat masih dalam proses menghubungkan diri ke jaringan WiFi, sementara Gambar 5 menunjukkan LCD setelah berhasil terhubung ke broker MQTT. Gambar 6 menampilkan *log* MQTT pada *dashboard* yang merekam semua pesan yang dikirim dan diterima.

Berdasarkan Tabel 1, unit ESP32 remote berhasil terhubung ke wifi yang berbeda dengan SSID “Remoterobot”, serta 2 unit ESP32 yaitu navigasi dan aktuator berhasil terhubung ke jaringan WiFi dengan SSID "WAITERUISU" dan ke broker *HiveMQ Cloud*. Waktu koneksi WiFi bervariasi antara 3200-3600 ms, sedangkan waktu koneksi MQTT berkisar antara 240-260 ms. Semua koneksi berhasil dalam 3 kali percobaan.

Tabel 2 menunjukkan pengukuran *delay* komunikasi yang diambil dari *log* MQTT. Rata-rata *delay* pengiriman perintah dari *Remote* ke Navigasi adalah 115 ms, *delay* status dari Navigasi ke *Remote* 102 ms, dan *delay event* dari Navigasi ke Aktuator 108 ms. Rata-rata keseluruhan *delay* komunikasi adalah 111 ms, yang masih jauh di bawah batas 500 ms untuk sistem *realtime*. Hal ini membuktikan bahwa infrastruktur komunikasi IoT menggunakan protokol MQTT berjalan stabil dan responsif. pada pengujian nomor 6 dan 7, sistem menunjukkan ketergantungan yang sangat tinggi pada stabilitas titik akses (*access point*).

Ketika koneksi terputus, protokol MQTT tidak dapat mengirimkan data telemetri secara *real-time*, dan robot tidak dapat menerima perintah baru hingga koneksi berhasil dipulihkan kembali (*reconnected*).".

Pengujian ESP32 *Remote* (Antarmuka Pengguna)

Pengujian ini bertujuan untuk memastikan ESP32 *Remote* dapat berfungsi sebagai web server dan mengirimkan perintah melalui MQTT.

Gambar 7 Tampilan Web Server ESP32 *Remote*

Tabel 3 Hasil Pengujian ESP32 *Remote*

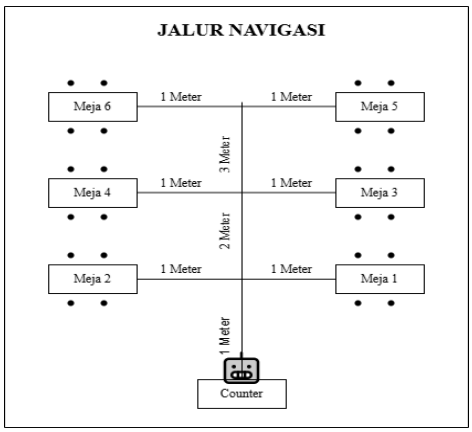
No	Skenario Pengujian	Prosedur	Hasil yang Diharapkan	Hasil Aktual	Status
1.	Akses halaman web	Buka browser, ketik IP ESP32 Remote	Halaman web muncul dengan tombol meja 1-6	Halaman web tampil	Berhasil
2.	Tombol Meja 1	Klik tombol "Meja 1"	Perintah "1" terkirim via MQTT	*Log* mencatat "1"	Berhasil
3.	Tombol Meja 2	Klik tombol "Meja 2"	Perintah "2" terkirim via MQTT	*Log* mencatat "2"	Berhasil
4.	Tombol Meja 3	Klik tombol "Meja 3"	Perintah "3" terkirim via MQTT	*Log* mencatat "3"	Berhasil
5.	Tombol Meja 4	Klik tombol "Meja 4"	Perintah "4" terkirim via MQTT	*Log* mencatat "4"	Berhasil
6.	Tombol Meja 5	Klik tombol "Meja 5"	Perintah "5" terkirim via MQTT	*Log* mencatat "5"	Berhasil
7.	Tombol Meja 6	Klik tombol "Meja 6"	Perintah "6" terkirim via MQTT	*Log* mencatat "6"	Berhasil
8.	Tombol *RETURN*	Klik tombol "*RETURN*"	Perintah "*go_counter*" terkirim	*Log* mencatat "*go_counter*"	Berhasil

9.	Tombol *STOP*	Klik tombol "*STOP*"	Perintah "STOP" terkirim	*Log* mencatat "*STOP*"	Berhasil

Pengujian ESP32 Remote dilakukan dengan mengakses halaman web melalui browser pada perangkat yang terhubung dalam jaringan yang sama. Gambar 4.7 menunjukkan tampilan web server yang menyediakan tombol pemilihan meja (1-6), tombol RETURN, dan tombol STOP. Berdasarkan Tabel 4.3, seluruh tombol berfungsi dengan baik dan mengirimkan perintah yang sesuai ke topik robot/command. Data pengiriman perintah ini juga tercatat di dashboard melalui topic robot/testdata dengan type command_count. Hal ini membuktikan bahwa ESP32 Remote berhasil diimplementasikan sebagai antarmuka pengguna untuk mengirim perintah ke robot.

Pengujian ESP32 Navigasi (Pergerakan Motor)

Pengujian ini bertujuan untuk memastikan ESP32 Navigasi dapat menerima perintah dari MQTT, menjalankan logika Finite State Machine (FSM), mengendalikan motor DC, serta mendeteksi halangan menggunakan sensor ultrasonik. Selain pengujian pada permukaan dan kondisi ideal, dilakukan pula pengujian batasan (stress test) pada sistem navigasi untuk mengidentifikasi kelemahan sensor jarak dan metode pergerakan berbasis waktu (time-based).



Gambar 8 Jalur Navigasi Robot dari *Counter* ke Meja 1-6

Tabel 4 Pengujian *State Machine* FSM pada ESP32 Navigasi

No	***State***	**Pemicu**	**Aksi Motor**	**Status**	**Hasil**
1.	*IDLE*	Robot siap	Motor *Stop*	"Robot Siap"	Berhasil
2.	*GO_MAIN*	Terima perintah meja	Motor Maju	"Menuju Meja X"	Berhasil
3.	*TURN_TO_TABLE*	Timer T_M1 selesai	Motor Belok	"Membelok"	Berhasil

4.	*GO_TABLE*	Timer T_90 selesai	Motor Maju	"Mendekati Meja X"	Berhasil
5.	*WAIT_RETURN*	Timer T_MEJA selesai	Motor *Stop*	"Tiba di Meja X"	Berhasil
6.	*TURN_180*	Terima "*go_counter*"	Motor Putar Balik	"Kembali"	Berhasil
7.	*BACK_MAIN*	Timer T_180 selesai	Motor Maju	"Kembali"	Berhasil
8.	*TURN_COUNTER*	Timer T_MEJA selesai	Motor Belok	"Kembali"	Berhasil
9.	*GO_COUNTER*	Timer T_90 selesai	Motor Maju	"Kembali"	Berhasil
10.	*TURN_ALIGN*	Timer jarak selesai	Motor Belok	"Robot Siap"	Berhasil
11.	*OBSTACLE*	Sensor ≤30 cm	Motor *Stop*	"HALANGAN!"	Berhasil

Tabel 5 Hasil Pengujian Navigasi dan Waktu Tempuh ke Meja

No	**Tujuan**	**Percobaan 1 (Detik)**	**Percobaan 2 (Detik)**	**Rata-rata (Detik)**	**Status**
1.	Meja 1	8.25	8.25	8.25	Sampai
2.	Meja 2	8.24	8.24	8.24	Sampai
3.	Meja 3	12.25	12.25	12.25	Sampai
4.	Meja 4	12.24	12.24	12.24	Sampai
5.	Meja 5	16.25	16.25	16.25	Sampai
6.	Meja 6	16.24	16.24	16.24	Sampai
No	**Skenario Pengujian**	**Kondisi Lingkungan / Alat**	**Hasil yang Diharapkan**	**Hasil Pengujian**	**Status**
7.	Menempatkan penghalang berupa kain tebal/spons atau objek dengan kemiringan sudut ekstrim di depan robot	Objek penyerap gelombang/miring	Sensor ultrasonik mendeteksi objek < 30 cm dan robot berhenti otomatis.	Gelombang ultrasonik terserap/terpantul ke arah lain, nilai jarak terbaca tidak akurat, dan robot menabrak penghalang.	Tidak Berhasil
8.	Mengirim perintah ke Meja 1 saat kondisi daya baterai motor DC sudah lemah (Tegangan < 11V)	Kapasitas baterai rendah	Robot mencapai koordinat Meja 1 tepat waktu dalam 8,25 detik.	Putaran roda (RPM) melambat akibat penurunan tegangan, robot berhenti sekitar 40 cm sebelum mencapai Meja 1 saat waktu habis.	Tidak Berhasil

Gambar 9 Pengujian Sensor Ultrasonik dengan Jarak 30cm

Gambar 10 LCD Menampilkan "HALANGAN" Saat *Obstacle* Terdeteksi

Pengujian ESP32 Navigasi dilakukan untuk memverifikasi implementasi *Finite State Machine* (FSM) dengan 11 *state*. Tabel 4 menunjukkan bahwa setiap state bertransisi sesuai pemicu dan menjalankan aksi motor yang tepat. Gambar 8 menunjukkan gambaran jalur Navigasi.

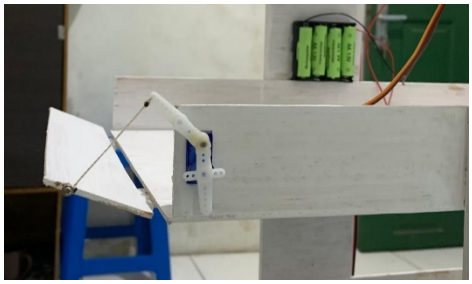
Tabel 5 menunjukkan robot berhasil mencapai seluruh meja tujuan (1-6) dengan waktu tempuh yang konsisten. Data ini diambil dari *topic robot/testdata* dengan *type mission_duration* yang dikirim oleh ESP32 Navigasi. Meja 1 membutuhkan waktu 8,25 detik, Meja 2 membutuhkan 8,24 detik, Meja 3 membutuhkan 12,25 detik, Meja 4 membutuhkan 12,24 detik, Meja 5 membutuhkan 16,25 detik, dan Meja 6 membutuhkan 16,24 detik. Perbedaan waktu antara meja kanan dan kiri disebabkan oleh perbedaan waktu belok (T_90_R = 1,05 detik vs T_90_L = 1,04 detik). Berdasarkan hasil pengujian pada nomor 7 dan 8, ditemukan dua batasan penting pada sistem navigasi robot. Pertama, sensor HC-SR04 mengalami kegagalan baca ketika berhadapan dengan material penyerap suara atau bidang miring. Kedua, metode navigasi berbasis waktu sangat rentan terhadap penurunan tegangan baterai karena tidak adanya umpan balik (*feedback*) dari roda, sehingga akurasi pemberhentian robot berkurang seiring habisnya daya baterai.

Gambar 4.9 dan 4.10 mendokumentasikan proses pengujian halangan.

Pengujian ESP32 Aktuator (Perangkat Pendukung)

Pengujian ini bertujuan untuk memastikan ESP32 Aktuator dapat menerima status dari MQTT dan mengendalikan motor servo, DFPlayer, serta LCD.

Gambar 11 Rak dalam Keadaan Tertutup



Gambar 12 Rak Terbuka di Meja 1 (Servo 1)

Gambar 13 LCD Menampilkan "Pesanan Tiba Meja 1"

Gambar 14 LCD Menampilkan "Kembali ke Counter"

Tabel 6 Hasil Pengujian ESP32 Aktuator

No	Status MQTT Diterima	Servo Aktif	Posisi Servo	Waktu Respon (ms)	Tampilan LCD	Status
1.	arrived_meja_1	Servo 1	90° (buka)	180	"Pesanan Tiba Meja 1"	Berhasil
2.	arrived_meja_2	Servo 1	90° (buka)	182	"Pesanan Tiba Meja 2"	Berhasil

3.	arrived_meja_3	Servo 2	90° (buka)	178	"Pesanan Tiba Meja 3"	Berhasil
4.	arrived_meja_4	Servo 2	90° (buka)	181	"Pesanan Tiba Meja 4"	Berhasil
5.	arrived_meja_5	Servo 3	90° (buka)	179	"Pesanan Tiba Meja 5"	Berhasil
6.	arrived_meja_6	Servo 3	90° (buka)	180	"Pesanan Tiba Meja 6"	Berhasil
7.	back_to_counter	Semua Servo	0° (tutup)	185	"Kembali ke Counter"	Berhasil

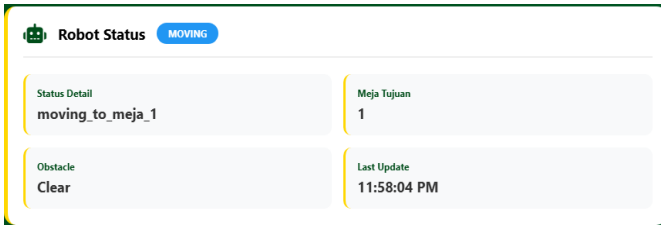
Pengujian ESP32 Aktuator dilakukan dengan memantau *respons* setiap komponen saat ESP32 menerima status dari topik robot/status. Tabel 4.6 menunjukkan bahwa servo merespons sesuai dengan meja tujuan, di mana Servo 1 aktif untuk meja 1-2, Servo 2 untuk meja 3-4, dan Servo 3 untuk meja 5-6. Waktu respon servo diambil dari topic robot/testdata dengan type *servo_open_duration* dan *servo_close_duration*. Rata-rata waktu respon servo adalah 180 ms.

Gambar 11 dan 12 menunjukkan perbandingan rak saat tertutup dan terbuka. Gambar 13 dan 14 menunjukkan tampilan LCD yang sesuai. Hal ini membuktikan bahwa ESP32 Aktuator berhasil mengendalikan perangkat pendukung berdasarkan data MQTT dari ESP32 Navigasi.

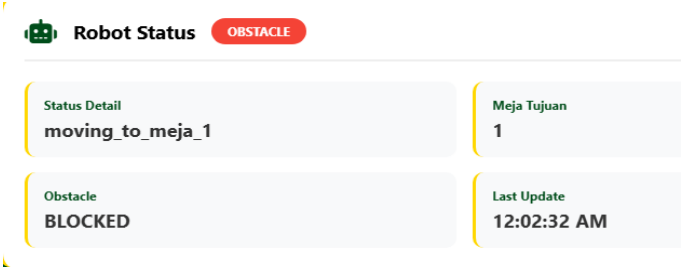
Pengujian *Dashboard* Monitoring

Pengujian ini bertujuan untuk memastikan *dashboard* dapat menampilkan data *realtime* dari ketiga unit ESP32 dan menyimpan data pengujian untuk keperluan analisis.

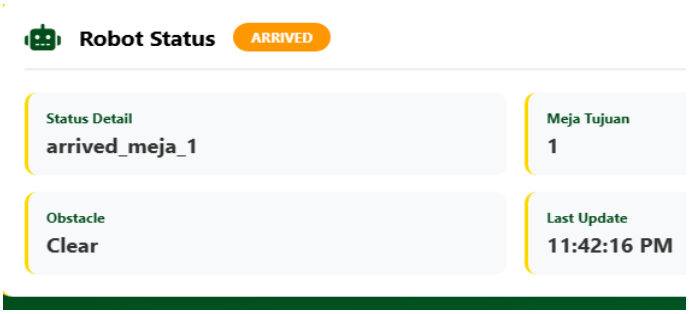
Gambar 15 Tampilan *Dashboard* Monitoring Terhubung ke MQTT



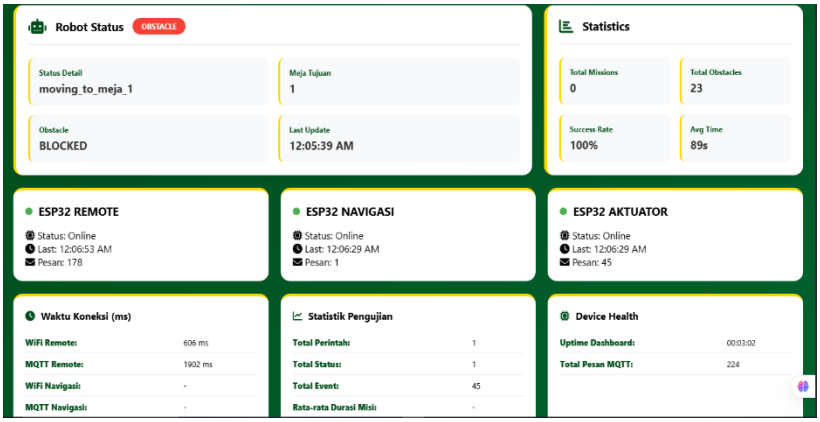
Gambar 16 *Dashboard* Menampilkan Status "*MOVING*"



Gambar 17 *Dashboard* Menampilkan Status "*OBSTACLE*"



Gambar 18 *Dashboard* Menampilkan Status "*ARRIVED*"



Gambar 19 Tampilan Data Pengujian (Waktu Koneksi, Statistik, *Device Health*)

Tabel 7 Hasil Pengujian *Dashboard* Monitoring

No	Fitur	Sumber Data	Hasil Pengujian	Status
1.	Koneksi MQTT	Indikator LED	Berubah hijau saat *connect*	Berhasil
2.	Status Robot	Topik robot/status	Berubah sesuai kondisi	Berhasil
3.	Status ESP32	Topik MQTT	*Online/Offline* sesuai aktivitas	Berhasil
4.	*Log* MQTT	Semua topik	Semua pesan tercatat	Berhasil
5.	Waktu Koneksi	Topik robot/testdata	Tampil di *card* pengujian	Berhasil
6.	Total Perintah	Topik robot/testdata	Terhitung otomatis	Berhasil
7.	Rata-rata Durasi	Perhitungan otomatis	Tampil di *card* pengujian	Berhasil
8.	*Export* CSV	Tombol *export*	File ter*download*	Berhasil

Pengujian *dashboard* monitoring dilakukan dengan mengamati setiap fitur saat robot beroperasi. Gambar 15 menunjukkan dashboard saat berhasil terhubung ke MQTT broker. Gambar 16 hingga 18 menunjukkan perubahan status robot sesuai dengan kondisi yang diterima dari topic robot/status. Gambar 19 menampilkan card data pengujian yang berisi informasi waktu koneksi ketiga ESP32, statistik perintah, dan rata-rata durasi.

Tabel 7 menunjukkan bahwa seluruh fitur *dashboard* berfungsi dengan baik. Data konektivitas dari topic robot/testdata berhasil ditampilkan, termasuk waktu koneksi WiFi dan MQTT dari ketiga ESP32. Total perintah, status, dan event terhitung secara otomatis. Rata-rata durasi misi, servo, dan *obstacle* dihitung berdasarkan data yang masuk. Fitur *export* CSV memungkinkan pengguna mendownload data pengujian untuk analisis lebih lanjut.

D. KESIMPULAN

Berdasarkan hasil perancangan, implementasi, dan pengujian, penelitian ini berhasil merancang serta membangun prototipe robot pelayan berbasis Internet of Things (IoT) menggunakan tiga unit ESP32 yang saling terintegrasi melalui protokol MQTT. Setiap ESP32 memiliki fungsi yang berbeda, yaitu sebagai web server pengirim perintah (ESP32 Remote), pengendali navigasi berbasis Finite State Machine (FSM) dan sensor ultrasonik (ESP32 Navigasi), serta pengendali aktuator seperti motor servo, modul DFPlayer, dan LCD (ESP32 Aktuator). Sistem komunikasi menggunakan mekanisme publish-subscribe melalui broker HiveMQ Cloud dengan tiga topik utama, yaitu robot/command, robot/status, dan robot/event. Hasil pengujian menunjukkan rata-rata waktu tunda komunikasi sebesar 111 ms, sehingga memenuhi kebutuhan sistem kendali realtime. Integrasi perangkat keras, perangkat lunak, serta komunikasi MQTT membuktikan bahwa konsep IoT dapat diterapkan secara efektif untuk mendukung sistem pelayanan otomatis berbasis jaringan.

Hasil pengujian juga menunjukkan bahwa robot mampu menjalankan seluruh proses pengantaran secara otomatis, mulai dari menerima perintah melalui antarmuka web, melakukan navigasi menuju meja tujuan, mendeteksi dan menangani hambatan menggunakan sensor ultrasonik, membuka rak sesuai tujuan pengantaran melalui motor servo, hingga kembali ke posisi awal (counter) setelah tugas selesai. Dashboard monitoring berbasis web mampu menampilkan status robot, tujuan pengantaran, statistik misi, serta data operasional secara realtime sehingga memudahkan proses pemantauan. Secara keseluruhan, sistem bekerja dengan baik pada kondisi lingkungan dan jaringan yang ideal. Namun, pengujian juga menunjukkan beberapa keterbatasan, seperti sensor ultrasonik yang kurang optimal dalam

mendeteksi material penyerap suara, menurunnya akurasi navigasi berbasis waktu ketika tegangan baterai melemah, serta munculnya peningkatan delay saat koneksi internet tidak stabil atau sistem menerima perintah secara berlebihan (spam command). Temuan ini menunjukkan bahwa keberhasilan implementasi IoT, MQTT, dan FSM sangat dipengaruhi oleh kestabilan perangkat keras dan kualitas jaringan, sehingga aspek tersebut perlu menjadi perhatian dalam pengembangan sistem pada penelitian selanjutnya.

E. DAFTAR PUSTAKA

Al Tahtawi, A. R., Somantri, Y., & Haritman, E. (2016). Design and Implementation of PID Control-based FSM Algorithm on Line Following Robot. JTERA (Jurnal Teknologi Rekayasa), 1(1), 23–30. https://doi.org/10.31544/jtera.v1.i1.2016.23-30

Amrulloh, S., & Kurniadi, H. (2022). Rancang Bangun Robot Pelayan Pasien Berbasis Internet of Things (Iot). 15(3), 333–339.

Annur, M. R., Hidayat, N., & Soebroto, A. A. (2021). Implementasi Deteksi Hujan dan Banjir Real-Time berbasis MQTT Over Websocket dengan ESP32. https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/download/8840/3834

Ashton, K. (2009). That 'Internet of Things' thing. RFID Journal, 22(7), 97–114.

Electronics, P. (2025). 1602 16x2 HD44780 LCD Display with Soldered I2C Module Support Documentation. Phipps Electronics. https://www.phippselectronics.com/support/1602-16x2-hd44780-lcd-display-with-soldered-i2c-module-support-documentation/

Espressif, 2020. (n.d.). ESP32 Series.

Fadhilah, M., Arifin, R., & Nurhadi, S. (2023). Implementasi Multi-Node ESP32 Berbasis MQTT untuk Sistem IoT Terdistribusi. Jurnal Informatika Dan Sistem Cerdas.

Hayubi, R. Al, Aulia, S., & Gunawan, D. A. (2024). Implementasi Sistem Penggerak Servo SG 90 Berbasis Arduino Uno dengan Kontrol Sudut Dinamis.

Index, C. (2026). DFPlayer Mini MP3 Player Module. ComponentIndex. https://componentindex.net/components/dfplayer-mini/

Lucet, E., Lucazeau, A., & Chemin, J. (2024). Autonomous Forklift Navigation Inside a Cluttered Logistics Factory. Proceedings of the 21st International Conference on Informatics in Control, Automation and Robotics (ICINCO), 2, 327–335. https://doi.org/10.5220/0013067600003822

Munawar, Z., Sastradipraja, C. K., Komalasari, R., Putri, N. I., Ma'sum, H., Mandowen, S. A., Mogi, I. K. A., Muliantara, A., Rahmad, I. F., Kmurawak, R. M. B., & others. (2023). Fundamental

internet of things (IoT): memahami teori dan penerapannya. Kaizen Media Publishing.

Paradigm, V. (2024). What is a Flowchart. Visual Paradigm International. https://www.visual-paradigm.com/guide/flowchart/what-is-a-flowchart/

Politeknik Elektronika Negeri Surabaya, P. E. S. P. (2019). Modul 1 pengenalan esp32 board 1.1. 6.

Ramadhan, A., Tamsir Ariyadi, et al. (2023). PROTOTYPE ROBOT PELAYAN PENERIMA DAN. Prototype Robot Pelayan Penerima Dan Pengantar Pesanan Berbasis Arduino.

Reynaldi Fakhri Pratama, R. Sunu Raihan W., A. N. P. (2023). Perancangan dan implementasi protokol mqtt pada sistem parkir cerdas berbasis iot. 11(3), 475–483.

Rofi, A., Afianah, N., Kautsar, S., Pratama, A. W., Winardi, A. W., Purnomo, F. E., & Putri, S. L. (2025). Development of Robotics Learning Modules based on IoT and. 2(3).

Rohman, M., Hartono, A., & Aziz, F. (2021). Protokol Komunikasi IoT Menggunakan MQTT. Jurnal Teknologi Elektro.

Selay, A., Andgha, G. D., Alfarizi, M. A., Bintang, M. I., Falah, M. N., Khaira, M., & Encep, M. (2022). Karimah Tauhid, Volume 1 Nomor 6 (2022), e-ISSN 2963-590X. 1, 860–868.

Suhaeb, S., Risal, A., & Makassar, U. N. (2024). Implementation of ESP32-Based Web Host For Control and Monitoring of Robotic Arm. 5(3), 249–254.

Suhardi, A., Nugroho, S., & Wijaya, R. Y. R.-2022. (2022). Perancangan Sistem Monitoring Berbasis IoT Menggunakan ESP32. Jurnal Teknik Elektro.

Suryani, D. (2020). Penerapan Internet of Things dalam Meningkatkan Efisiensi Sistem Informasi. Jurnal Teknologi Informasi Dan Pendidikan, 13(2), 45–52.

Utami, R., & Saputra, L. (2023). Sistem Kendali Perangkat Berbasis Dashboard IoT. Jurnal Teknologi Informasi.