TUGAS AKHIR (AE43250)

PENGEMBANGAN SISTEM MONITORING DAN PENGENDALIAN OPERASIONAL *AUTONOMOUS MOBILE ROBOT* TERINTEGRASI *WAREHOUSE MANAGEMENT SYSTEM* BERBASIS WEB

Muhammad Fikri Rahman
NRP. 0922040064

Dosen Pembimbing
Didik Sukoco, S.T., M.T.
Dimas Pristovani Riananda, S.ST., M.T.

PROGRAM STUDI D4 TEKNIK OTOMASI
JURUSAN TEKNIK KELISTRIKAN KAPAL
POLITEKNIK PERKAPALAN NEGERI SURABAYA
SURABAYA
2026

TUGAS AKHIR (AE43250)

PENGEMBANGAN SISTEM MONITORING DAN PENGENDALIAN OPERASIONAL *AUTONOMOUS MOBILE ROBOT* TERINTEGRASI *WAREHOUSE MANAGEMENT SYSTEM* BERBASIS WEB

Muhammad Fikri Rahman
NRP. 0922040064

Dosen Pembimbing
Didik Sukoco, S.T., M.T.
Dimas Pristovani Riananda, S.ST., M.T.

PROGRAM STUDI D4 TEKNIK OTOMASI
JURUSAN TEKNIK KELISTRIKAN KAPAL
POLITEKNIK PERKAPALAN NEGERI SURABAYA
SURABAYA
2026

(Halaman ini sengaja dikosongkan)

LEMBAR PENGESAHAN

TUGAS AKHIR

PENGEMBANGAN SISTEM MONITORING DAN PENGENDALIAN OPERASIONAL *AUTONOMOUS MOBILE ROBOT* TERINTEGRASI *WAREHOUSE MANAGEMENT SYSTEM* BERBASIS WEB

Disusun Oleh:
Muhammad Fikri Rahman
0922040064

Diajukan Untuk Memenuhi Salah Satu Syarat Kelulusan
Program Studi D4 Teknik Otomasi
Jurusan Teknik Kelistrikan Kapal
POLITEKNIK PERKAPALAN NEGERI SURABAYA

Disetujui oleh Tim penguji Tugas Akhir Tanggal Ujian : 6 Agustus 2026
Periode Wisuda : Oktober 2026

Menyetujui,

Dosen Penguji	NIDN	Tanda Tangan
1. Agus Khumaidi, S.ST., M.T.	(0017089303)	(.....)
2. Lilik Subiyanto, S.T., M.T.	(0030016903)	(.....)
3. Adianto, S.T., M.T.	(0002077704)	(.....)
4. Dimas Pristovani Riananda, S.ST., M.T.	(0031109304)	(.....)
Dosen Pembimbing	**NIDN**	**Tanda Tangan**
1. Didik Sukoco, S.T., M.T.	(0005026909)	(.....)
2. Dimas Pristovani Riananda, S.ST., M.T.	(0031109304)	(.....)

Menyetujui
Ketua Jurusan,

Isa Rachman, S.T., M.T.
NIP. 198008162008121001

Mengetahui
Koordinator Program Studi,

Agus Khumaidi, S.ST., M.T.
NIP. 199308172020121004

(Halaman ini sengaja dikosongkan)

PPNS	**PERNYATAAN BEBAS PLAGIAT**	No. : F.WD I. 021 Date : 3 Nopember 2015 Rev. : 01 Page : 1 dari 1

Yang bertandatangan dibawah ini :

Nama : Muhammad Fikri Rahman

NRP. : 0922040064

Jurusan/Prodi : Teknik Kelistrikan Kapal/D4 Teknik Otomasi

Dengan ini menyatakan dengan sesungguhnya bahwa :

Tugas Akhir yang akan saya kerjakan dengan judul :

PENGEMBANGAN SISTEM MONITORING DAN PENGENDALIAN OPERASIONAL *AUTONOMOUS MOBILE ROBOT* TERINTEGRASI *WAREHOUSE MANAGEMENT SYSTEM* BERBASIS WEB

Adalah **benar karya saya sendiri** dan **bukan plagiat dari karya orang lain.**

Apabila dikemudian hari terbukti terdapat plagiat dalam karya ilmiah tersebut, maka saya bersedia menerima sanksi sesuai ketentuan peraturan yang berlaku.

Demikian surat pernyataan ini saya buat dengan penuh tanggung jawab.

Surabaya, 03 September 2026
Yang membuat pernyataan,

(Muhammad Fikri Rahman)
NRP. 0922040064

(Halaman ini sengaja dikosongkan)

KATA PENGANTAR

Puji syukur penulis panjatkan ke hadirat Allah SWT atas segala rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul "**PENGEMBANGAN SISTEM MONITORING DAN PENGENDALIAN OPERASIONAL *AUTONOMOUS MOBILE ROBOT* TERINTEGRASI *WAREHOUSE MANAGEMENT SYSTEM* BERBASIS WEB"** sebagai salah satu syarat untuk memperoleh gelar Sarjana Terapan pada Program Studi Teknik Otomasi, Jurusan Teknik Kelistrikan Kapal, Politeknik Perkapalan Negeri Surabaya.

Penyusunan Tugas Akhir ini tidak terlepas dari bantuan, dukungan, dan kebersamaan dari berbagai pihak. Oleh karena itu, dengan penuh rasa hormat, penulis mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Kedua orang tua dan adik penulis, yang selalu memberikan doa, dukungan, semangat, serta kepercayaan kepada penulis selama menjalani perkuliahan hingga menyelesaikan Tugas Akhir ini. Terima kasih telah menjadi alasan untuk terus berusaha dan menyelesaikan apa yang telah dimulai.
2. Bapak Rachmad Tri Soelistijono, S.T., M.T. selaku Direktur Politeknik Perkapalan Negeri Surabaya.
3. Bapak Isa Rachman, S.T., M.T**.**, selaku Ketua Jurusan Teknik Kelistrikan Kapal Politeknik Perkapalan Negeri Surabaya.
4. Bapak Agus Khumaidi, S.ST., M.T**.,** selaku Koordinator Program Studi Teknik Otomasi Politeknik Perkapalan Negeri Surabaya.
5. Bapak Ryan Yudha Adhitya, S.ST., M.T**.,** selaku Koordinator Tugas Akhir yang telah memberikan arahan selama pelaksanaan Tugas Akhir.
6. Bapak Didik Sukoco, S.T., M.T**.**, selaku Dosen Pembimbing I yang telah memberikan bimbingan, arahan, masukan, dan waktu selama proses penyusunan Tugas Akhir ini.
7. Bapak Dimas Pristovani Riananda, S.ST., M.T**.**, selaku Dosen Pembimbing II yang telah memberikan ilmu, arahan, serta masukan dalam menyelesaikan Tugas Akhir ini.

8. Seluruh jajaran dosen dan tenaga kependidikan Jurusan Teknik Kelistrikan Kapal PPNS yang telah memberikan ilmu, pengalaman, dan bimbingan selama penulis menempuh pendidikan.
9. Seseorang yang penulis sebut dengan nama samaran **"**Yeuu**"**, terima kasih atas segala bantuan, perhatian, dan dukungan yang telah diberikan selama proses penyusunan Tugas Akhir ini.
10. Valentino Karocelli, selaku partner Tugas Akhir, yang telah menjadi teman seperjuangan dalam setiap proses, diskusi, dan kendala selama menyelesaikan Tugas Akhir.
11. Teman-teman Lab M, yang telah menjadi tempat berjuang, mengerjakan Tugas Akhir, hingga menghabiskan malam bersama. Terima kasih atas kebersamaan dan bantuan yang diberikan selama proses ini.
12. Seluruh teman-teman Teknik Otomasi Angkatan 2022, khususnya teman-teman TOC sebagai teman sekelas, terima kasih atas kebersamaan, dukungan, bantuan, dan berbagai cerita selama menjalani masa perkuliahan.
13. Teman-teman "Rumah Kita", terima kasih atas kebersamaan selama tinggal bersama, berbagai cerita, canda, dan dukungan yang turut menjadi bagian dari perjalanan penulis selama menyelesaikan perkuliahan dan Tugas Akhir.
14. Seluruh pihak yang telah membantu dan memberikan dukungan kepada penulis, baik secara langsung maupun tidak langsung, yang tidak dapat disebutkan satu per satu.

Penulis menyadari bahwa Tugas Akhir ini masih memiliki kekurangan. Oleh karena itu, kritik dan saran yang membangun sangat diharapkan untuk perbaikan di masa mendatang. Penulis berharap Tugas Akhir ini dapat memberikan manfaat bagi pembaca serta menjadi pengalaman dan pembelajaran yang berharga bagi penulis.

Penulis

(Muhammad Fikri Rahman)

PENGEMBANGAN SISTEM MONITORING DAN PENGENDALIAN OPERASIONAL *AUTONOMOUS MOBILE ROBOT* TERINTEGRASI *WAREHOUSE MANAGEMENT SYSTEM* BERBASIS WEB

Muhammad Fikri Rahman

ABSTRAK

Proses *cross docking* pada gudang skala kecil membutuhkan sistem pengantaran barang yang cepat dan tepat sasaran, sehingga diperlukan kontrol terpusat yang mampu mengintegrasikan *Autonomous Mobile Robot* (AMR) dengan sistem informasi berbasis web. Penelitian ini merancang *AMR Control Station* berbasis web yang mengintegrasikan antarmuka web, basis data MongoDB, dan robot berbasis ROS1 pada Jetson melalui protokol *ROSBridge WebSocket*, dilengkapi *decision tree* berbasis aturan untuk menentukan prioritas pengantaran berdasarkan jenis barang, jarak, dan waktu tunggu. Pengujian dilakukan pada gudang skala kecil dengan tiga terminal *pick-up* (A, B, C) dan tiga terminal *drop-off* (A−, B−, C−). Hasil pengujian menunjukkan bahwa *decision tree* berhasil menentukan terminal prioritas secara tepat pada 3 dari 3 kondisi pengujian (100%), dengan urutan atribut jenis barang dan jarak memiliki nilai *Gain Ratio* sebesar 1,000, sedangkan waktu tunggu sebesar 0,274. Pengujian integrasi sistem menunjukkan kesesuaian data antara *website*, basis data, dan robot mencapai 100% pada 8 parameter yang diuji, dengan durasi siklus pengantaran antara 86–196 detik. Monitoring posisi robot secara *real-time* pada peta tiga dimensi menunjukkan galat navigasi sebesar -0,0 cm. Sistem yang dikembangkan terbukti mampu memenuhi kebutuhan pengendalian dan pemantauan operasional AMR secara terintegrasi dalam simulasi proses *cross docking* berbasis web.

Kata kunci: *Autonomous Mobile Robot, Decision Tree C4.5, Warehouse Management System, Cross Docking, ROSBridge WebSocket*

(Halaman ini sengaja dikosongkan)

DEVELOMPMENT OF A WEB-BASED WAREHOUSE MANAGEMENT SYSTEM INTEGRATED WITH MONITORING AND OPERATIONAL CONTROL OF AN AUTONOMOUS MOBILE ROBOT

Muhammad Fikri Rahman

ABSTRACT

The cross docking process in small-scale warehouses requires a fast and accurate delivery system, necessitating centralized control capable of integrating an Autonomous Mobile Robot (AMR) with a web-based information system. This research designs a web-based AMR Control Station that integrates a web interface, MongoDB database, and a ROS1-based robot running on Jetson through the ROSBridge WebSocket protocol, equipped with a rule-based decision tree to determine delivery priority based on item type, distance, and waiting time. Testing was conducted in a small-scale warehouse with three pick-up terminals (A, B, C) and three drop-off terminals (A−, B−, C−). The results show that the decision tree successfully determined the correct priority terminal in 3 out of 3 test conditions (100%), with the item type and distance attributes achieving a Gain Ratio value of 1.000, while waiting time achieved 0.274. System integration testing showed 100% data consistency between the website, database, and robot across 8 tested parameters, with delivery cycle durations ranging from 86–196 seconds. Real-time monitoring of robot position on the three-dimensional map showed a navigation error of -0.0 cm. The developed system is proven capable of meeting the operational control and monitoring needs of an AMR in an integrated manner within a web-based cross docking simulation.

Keywords: *Autonomous Mobile Robot, Decision Tree C4.5, Warehouse Management System, Cross Docking, ROSBridge WebSocket.*

(Halaman ini sengaja dikosongkan)

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR NOTASI

S : Himpunan seluruh data latih (*dataset*) yang digunakan untuk membangun decision tree, misalnya data permintaan pengantaran barang oleh robot.

n : Jumlah kelas dalam data, misalnya kelas prioritas pengantaran (FEFO, express, reguler).

p_i: Proporsi data pada kelas ke-i di dalam himpunan data S.

A : Atribut yang digunakan sebagai dasar pemisahan data, seperti jenis barang atau jarak pengantaran.

Values(A) : Kumpulan seluruh nilai yang dimiliki oleh atribut A.

v : Salah satu nilai dari atribut A.

S_v: Subhimpunan data dari Syang memiliki nilai atribut $A = v$.

$| S |$: Jumlah seluruh data dalam himpunan S.

$| S_v |$: Jumlah data pada subhimpunan S_v.

Entropy(S) : Nilai ketidakpastian pada seluruh data dalam himpunan S.

Gain(S, A) : Nilai pengurangan ketidakpastian data setelah dilakukan pemisahan berdasarkan atribut A.

SplitInfo(S, A) : Nilai informasi pembagi yang digunakan untuk menormalkan *Information Gain*.

GainRatio(S, A) : Nilai rasio yang digunakan untuk menentukan atribut terbaik sebagai simpul keputusan (*decision node*).

(Halaman ini sengaja dikosongkan)

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi otomasi dan digitalisasi di bidang pergudangan mendorong kebutuhan akan sistem *Warehouse Management System* (WMS) yang mampu mengelola proses penyimpanan dan pengantaran barang secara efektif, akurat, dan terintegrasi, khususnya pada area *loading dock* sebagai titik krusial pertukaran barang antara zona penyimpanan dan proses distribusi. *Loading dock* berperan sebagai penghubung utama dalam alur logistik gudang, di mana efisiensi proses pengambilan (*pick-up*) dan pengantaran (*drop-off*) barang pada area ini secara langsung memengaruhi kecepatan dan akurasi keseluruhan rantai distribusi. WMS berperan penting dalam mengatur alur barang pada area tersebut, mulai dari penerimaan, penyimpanan sementara, hingga pengiriman ke titik tujuan, serta menyediakan informasi operasional secara *real-time*. Seiring dengan meningkatnya pemanfaatan robot pengantar barang di lingkungan *loading dock*, diperlukan sistem manajemen berbasis web yang tidak hanya berfungsi sebagai alat *monitoring*, tetapi juga mampu melakukan pengambilan keputusan pengantaran barang secara otomatis dan sistematis (Nurhadi & Cahyono, 2022).

Penelitian sebelumnya umumnya mengembangkan WMS dalam bentuk *Graphical User Interface* (GUI) berbasis Python, di mana proses pengantaran barang pada area *loading dock* masih dilakukan secara manual dengan cara menginputkan parameter jarak dan sudut pergerakan robot. Sistem tersebut belum menyediakan fitur *monitoring* yang komprehensif, seperti penampilan persentase baterai robot, visualisasi peta area *loading dock*, pencatatan barang yang telah diantar, maupun evaluasi performa pengantaran. sehingga proses sortir dan pengantaran pada area *loading dock* masih bergantung pada intervensi pengguna (Lubis et al., 2024).

Berdasarkan permasalahan tersebut, penelitian ini mengusulkan pengembangan sistem WMS berbasis web yang difokuskan pada otomasi proses pengantaran barang di area *loading dock*, dengan pembaruan fitur yang lebih lengkap dan terintegrasi. Sistem yang dikembangkan memungkinkan pengguna melakukan *request* pengantaran barang melalui antarmuka web, menampilkan persentase baterai robot, visualisasi peta *loading dock* secara 3D, total *delivery*, *success rate* pengantaran, serta *logger* dan *delivery record* sebagai bahan evaluasi sistem. Metode *Decision Tree C 4.5* digunakan untuk memprioritaskan barang dengan tanggal kedaluwarsa terdekat pada proses *pick-up* di *loading dock*.

Untuk mendukung proses pengambilan keputusan pengantaran secara otomatis, sistem ini dilengkapi dengan metode *decision tree* berbasis aturan (*rule-based decision tree*). *Decision tree* digunakan untuk mengklasifikasikan barang berdasarkan parameter status *express* dan tanggal kedaluwarsa yang diinputkan pada setiap terminal *pick-up* (A, B, dan C) di area *loading dock*. Barang dengan tanggal kedaluwarsa diprioritaskan menggunakan prinsip FEFO sebagai prioritas utama, sedangkan barang tanpa tanggal kedaluwarsa diklasifikasikan ke dalam pengantaran FIFO dengan pembagian antara prioritas *express* dan reguler. Hasil keputusan dari *decision tree* selanjutnya digunakan untuk menentukan pengantaran yang akan dieksekusi oleh robot secara otomatis, sehingga mengurangi ketergantungan terhadap intervensi operator dan meningkatkan efisiensi proses bongkar-muat pada area *loading dock*.

Dalam implementasinya, sistem dibatasi pada representasi area *loading dock* dengan tiga terminal *pick-up* (A, B, C) dan tiga terminal *drop* barang (A−, B−, C−) sebagai model alur distribusi barang pada titik pertukaran gudang tersebut. Namun demikian, penelitian ini memiliki keterbatasan dalam hal skala implementasi dan kompleksitas sistem. Pengujian dilakukan pada representasi area *loading dock* berskala kecil berukuran 8 × 6 meter, dengan barang berupa kardus simulasi berbobot maksimal 1 kg per unit dan kapasitas angkut robot dibatasi hingga 3 kg. Keterbatasan jumlah terminal *pick-up* dan *drop* barang menyebabkan sistem belum sepenuhnya merepresentasikan kondisi *loading dock*

pada gudang industri yang kompleks. Oleh karena itu, penelitian ini difokuskan sebagai tahap pengembangan dan validasi konsep sistem WMS berbasis web dengan penerapan FIFO, FEFO, dan *decision tree* sebagai mekanisme pengambilan keputusan pengantaran robot pada area *loading dock*, yang diharapkan dapat menjadi dasar pengembangan menuju implementasi pada skala industri (Sihaloho & Hidayati, 2023).

1.2 Rumusan Masalah

Berdasarkan penjelasan latar belakang tersebut, terdapat rumusan masalah sebagai berikut:

1. Bagaimana merancang dan mengimplementasikan sistem *Warehouse Management System* (WMS) berbasis web yang terintegrasi dengan *Autonomous Mobile Robot* sebagai pengantar barang dalam lingkungan gudang skala kecil?
2. Bagaimana menampilkan monitoring kondisi robot secara *real-time*, meliputi status pengantaran dan peta lokasi robot di dalam gudang?
3. Bagaimana decision tree berbasis aturan dapat digunakan sebagai mekanisme pengambilan keputusan *Autonomous Mobile Robot* untuk menentukan prioritas pengantaran barang pada gudang skala kecil dengan tiga terminal pick-up (A, B, C) dan tiga terminal drop barang (A−, B−, C−), berdasarkan status pada barang *express*, *regular* dan tanggal kedaluwarsa?

1.3 Batasan Masalah

Agar permasalahan yang diangkat di dalam kajian penelitian ini lebih fokus dan dapat menyelesaikan permasalahan. Adapun Batasan pada proses perancangan, pengerjaan, dan pengujian sistem ini sehingga diberikan Batasan penelitian antara lain yaitu sebagai berikut:

1. Sistem WMS yang dikembangkan diimplementasikan pada gudang skala kecil dengan ukuran 8 × 6 meter. Yang merepresentasikan gudang skala kecil dan belum diterapkan pada gudang industri nyata.
2. Kapasitas maksimum beban yang dapat dibawa robot dibatasi sebesar

2 kg.

3. Proses navigasi Autonomous Mobile Robot menggunakan algoritma A* yang dikembangkan pada penelitian sebelumnya. Penelitian ini hanya memanfaatkan hasil perhitungan jalur tersebut untuk memperoleh estimasi jarak tempuh antar terminal sebagai salah satu parameter dalam proses pengambilan keputusan menggunakan Decision Tree.

1.4 Tujuan Penelitian

Penelitian Tugas Akhir ini diharapkan dapat memberikan manfaat, antara lain:

1. Mengembangkan sistem *Warehouse Management System* (WMS) berbasis web untuk manajemen dan monitoring yang terintegrasi pada pengantaran barang menggunakan *Autonomous Mobile Robot*.
2. Menyediakan fitur monitoring *real-time*, status dan riwayat pengantaran, visualisasi peta gudang.
3. Mengimplementasikan decision tree berbasis aturan sebagai mekanisme pengambilan keputusan pengantaran robot berdasarkan status barang (*express, regular, expired*). Menguji dan mengevaluasi sistem pada lingkungan gudang skala kecil dengan konfigurasi terminal pick-up A, B, C dan drop barang A−, B−, C−.

1.5 Manfaat Penelitian

Penelitian Tugas Akhir ini diharapkan dapat memberikan manfaat, antara lain:

1. Membantu meningkatkan efisiensi operasional gudang melalui sistem terintegrasi WMS–AMR yang mampu mengatur pengambilan keputusan urutan pengantaran barang secara terstruktur dan terkontrol menggunakan *decision tree.*

2. Memudahkan operator dan supervisor dalam memantau kondisi operasional robot secara *real-time*, seperti posisi robot, dan tingkat keberhasilan pengantaran, melalui satu antarmuka berbasis web.
3. Menjadi acuan implementasi awal bagi industri atau institusi pendidikan dalam mengembangkan sistem gudang berbasis AMR yang terintegrasi dengan WMS.

(Halaman ini sengaja dikosongkan)

BAB 2
TINJAUAN PUSTAKA

Bab ini memuat ulasan referensi dari sumber, jurnal imiah dan laporan penelitian. Penjelasan mengenai penerapan teori, objek penelitian, pemilihan metode, penggunaan perangkat lunak dan perangkat keras dalam perancangan tugas akhir akan dibahas secara rinci pada bagian ini.

2.1 Kajian Penelitian Terdahulu

Penelitian terdahulu disajikan dalam bentuk tabel-tabel sebagai panduan tugas akhir penulis. Berbagai teori digunakan dalam menyusun tugas akhir ini hingga terciptanya tugas akir yang baik. Penulis menggunakan jurnal ilmiah berikut Tabel 2.1 sebagai referensi penelitian.

Tabel 2. 1 Tabel Penelitian Terdahulu

No	Nama	Judul	Kesimpulan
1.	(Alif et al., 2022)	Perancangan Aplikasi WMS (Warehouse Management System) untuk Efisiensi Penyimpanan Stock Barang pada Warehouse dengan Metode FIFO (First In First Out).	Berdasarkan kajian jurnal terkait penerapan metode First In First Out (FIFO) dan First Expired First Out (FEFO) dalam sistem Warehouse Management System (WMS), dapat disimpulkan bahwa kedua metode tersebut efektif dalam mengatur prioritas pengeluaran dan distribusi barang di gudang. FIFO menjamin alur barang keluar sesuai urutan kedatangan sehingga mencegah penumpukan barang lama, sedangkan FEFO meminimalkan risiko kerugian dengan memprioritaskan barang yang memiliki tanggal kedaluwarsa terdekat.

No	Nama	Judul	Kesimpulan
			Namun, penerapan FIFO dan FEFO pada penelitian terdahulu umumnya masih terbatas pada pengelolaan persediaan dan belum terintegrasi secara langsung dengan sistem pengantaran barang menggunakan robot maupun sistem monitoring berbasis web secara real-time, sehingga membuka peluang pengembangan lebih lanjut.
2.	(Keith & La, 2024)	Review of Autonomous Mobile Robots for the Warehouse Environment	Berdasarkan kajian jurnal mengenai penggunaan Autonomous Mobile Robot (AMR) dalam pergudangan dan logistik, dapat disimpulkan bahwa AMR mampu meningkatkan efisiensi, fleksibilitas, dan kecepatan proses pemindahan barang karena dapat beroperasi secara mandiri tanpa jalur tetap serta mudah beradaptasi terhadap perubahan tata letak gudang. AMR juga berkontribusi dalam mengurangi beban kerja manusia dan kesalahan operasional. Meskipun demikian, sebagian besar penelitian masih berfokus pada aspek navigasi dan kinerja robot, sementara integrasi AMR dengan Warehouse Management System (WMS)—terutama dalam hal manajemen

No	Nama	Judul	Kesimpulan
			prioritas pengantaran, monitoring operasional, dan pencatatan performa pengiriman—masih terbatas, sehingga diperlukan pengembangan sistem yang menggabungkan AMR dengan WMS berbasis web secara lebih komprehensif.
3.	(Rakhman et al., n.d.)	Robot Mobile Otonom Menggunakan Metode Odometry	Penelilitian yang dilakukan (Rakhman et al., n.d.) menunjukkan bahwa odometry berbasis wheel encoder (rotary encoder) dapat digunakan secara efektif untuk memperkirakan posisi robot mobile otonom tanpa menggunakan sensor LiDAR. Studi ini mengembangkan robot mobile yang mampu bergerak secara otomatis menuju titik tujuan berdasarkan perhitungan odometry dari putaran roda, di mana perubahan posisi robot dihitung menggunakan data encoder dan model kinematika gerak. Hasilnya menunjukkan bahwa odometry memungkinkan estimasi posisi untuk navigasi robot meskipun dalam area tanpa dukungan sensor jarak seperti LiDAR, sehingga memberikan landasan ilmiah yang kuat bagi penggunaan data odometry dalam sistem pemetaan lokasi robot secara

No	Nama	Judul	Kesimpulan
			realtime yang dapat ditampilkan di antarmuka web sebagai bagian dari monitoring otomasi gudang.
4.	(Devega et al., 2024)	Pembangunan sistem inventori apotek menggunakan metode fifo dan fefo	Penelitian yang dilakukan oleh (Devega et al., 2024) menyimpulkan bahwa penerapan metode *First In First Out* (FIFO) dan *First Expired First Out* (FEFO) pada sistem inventori apotek mampu meningkatkan efektivitas pengelolaan stok barang, khususnya dalam mengurangi risiko terjadinya penumpukan dan kedaluwarsa obat. Sistem yang dikembangkan dapat membantu pengguna dalam menentukan prioritas pengeluaran barang berdasarkan waktu masuk dan tanggal kedaluwarsa secara otomatis, sehingga proses distribusi menjadi lebih terstruktur dan akurat. Hasil penelitian menunjukkan bahwa kombinasi metode FIFO dan FEFO memberikan kontribusi positif terhadap ketertiban alur persediaan dan mendukung pengambilan keputusan yang lebih baik dalam manajemen gudang, meskipun sistem masih terbatas pada fungsi inventori dan belum terintegrasi dengan sistem otomasi atau robot pengantar barang.

Berdasarkan kajian terhadap penelitian-penelitian terdahulu, dapat disimpulkan bahwa penerapan metode *First In First Out* (FIFO) dan *First Expired First Out* (FEFO) dalam sistem *Warehouse Management System* (WMS) terbukti efektif dalam mengatur prioritas pengeluaran dan distribusi barang, khususnya untuk menjaga alur barang masuk dan meminimalkan risiko kedaluwarsa (Alif et al., 2022). Di sisi lain, penggunaan Autonomous Mobile Robot (AMR) dalam lingkungan pergudangan dan logistik mampu meningkatkan efisiensi, fleksibilitas, serta kecepatan pemindahan barang karena robot dapat beroperasi secara mandiri tanpa jalur tetap dan mengurangi ketergantungan terhadap tenaga manusia (Keith & La, 2024). Lebih lanjut, penelitian mengenai integrasi WMS dengan AMR dalam kerangka *smart warehouse* menunjukkan bahwa koordinasi robot secara *real-time* melalui sistem manajemen gudang dapat meningkatkan kinerja operasional secara signifikan, baik dari sisi kecepatan pemenuhan pesanan, akurasi pelacakan inventaris, maupun efisiensi biaya (Zhang & Abellera, 2025). Namun demikian, sebagian besar penelitian tersebut masih membahas FIFO/FEFO, AMR, dan integrasi WMS–AMR secara terpisah, sehingga diperlukan pengembangan sistem WMS berbasis web yang mengintegrasikan metode FIFO dan FEFO dengan pengantaran barang menggunakan AMR serta dilengkapi fitur monitoring dan evaluasi performa secara terpusat.

2.2 Kajian Pustaka

Pada bagian ini mengulas tentang teori-teori yang menjadi acuan dari pelaksanaan penelitian ini.

2.2.1 *Warehouse Management System* pada Proses *Cross Docking*

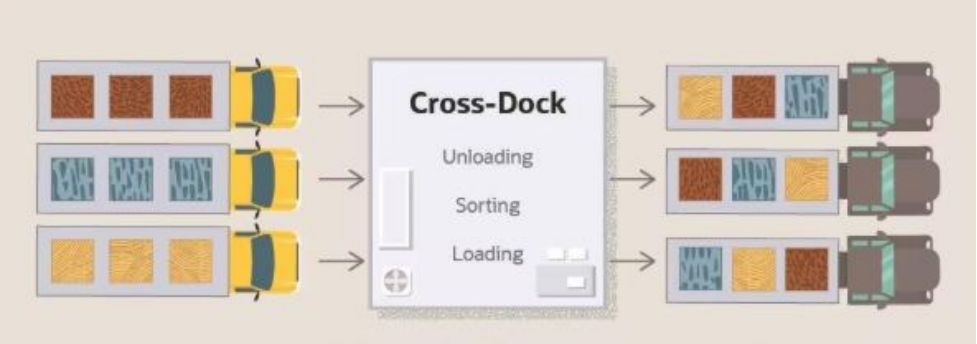
Warehouse Warehouse Management System (WMS) merupakan suatu sistem perangkat lunak yang digunakan untuk mengelola dan mengendalikan seluruh aktivitas operasional di dalam gudang, mulai dari penerimaan barang (*receiving*), penyimpanan (*storage*), pengambilan (*picking*), hingga pengiriman barang (*shipping*). Fungsi utama WMS adalah menyediakan visibilitas dan kontrol atas pergerakan serta status barang secara real-time, sehingga proses logistik dapat berjalan lebih efisien, akurat, dan terukur dibandingkan pengelolaan gudang

secara manual. Dalam praktiknya, WMS tidak hanya berperan sebagai pencatat data, tetapi juga sebagai pengambil keputusan operasional, misalnya dalam menentukan prioritas pengerjaan, penjadwalan sumber daya, serta pengalokasian lokasi penyimpanan atau jalur distribusi barang.

Cross docking merupakan salah satu strategi distribusi logistik di mana barang yang diterima dari sisi inbound langsung disalurkan menuju sisi outbound tanpa melalui proses penyimpanan jangka panjang di gudang. Konsep ini bertujuan untuk memangkas waktu tunggu barang di gudang, mengurangi kebutuhan ruang penyimpanan, serta mempercepat siklus distribusi secara keseluruhan. Dalam skema cross docking, kecepatan dan ketepatan pemindahan barang dari titik penerimaan ke titik pengiriman menjadi faktor yang sangat kritis, karena keterlambatan pada satu titik dapat berdampak pada keseluruhan rantai distribusi. Oleh karena itu, penerapan cross docking menuntut adanya sistem pengambilan keputusan yang mampu menentukan urutan prioritas penanganan barang secara cepat dan konsisten, terutama ketika terdapat lebih dari satu permintaan pemindahan barang yang harus ditangani dalam waktu yang berdekatan.

Pada penelitian ini, konsep WMS dan cross docking diimplementasikan dalam bentuk AMR *Control Station* berbasis web yang berfungsi sebagai pusat kendali operasional *Autonomous Mobile Robot* (AMR) dalam mensimulasikan proses cross docking di lingkungan gudang. Sistem ini berperan sebagai lapisan WMS yang menerima permintaan pengiriman barang dari beberapa terminal pick-up (titik inbound) menuju terminal drop-off yang sesuai (titik *outbound*), kemudian menentukan urutan prioritas pengerjaan menggunakan algoritma Decision Tree C4.5 berdasarkan jenis barang, jarak tempuh, dan waktu tunggu permintaan. Keputusan prioritas yang dihasilkan oleh sistem selanjutnya diteruskan kepada AMR melalui komunikasi ROSBridge WebSocket, sehingga robot dapat secara otomatis menavigasi menuju terminal pick-up yang diprioritaskan, mengangkat barang, dan mengantarkannya ke terminal drop-off yang bersesuaian tanpa memerlukan proses penyimpanan barang di tengah jalur distribusi. Dengan demikian, sistem yang dibangun tidak hanya berfungsi sebagai antarmuka kendali robot, tetapi juga merepresentasikan fungsi WMS dalam

mengatur alur cross docking, yaitu memastikan setiap barang dipindahkan secara langsung dan tepat sasaran berdasarkan prioritas yang telah ditentukan secara otomatis oleh sistem.



Gambar 2. 1Warehouse Management System
(Sumber : https://au.syspro.com/business-processes/warehouse-management-system/)

2.2.2 Decision Tree

Decision Tree C4.5 merupakan salah satu algoritma *supervised learning* yang digunakan untuk membangun model klasifikasi berdasarkan data berlabel, dan dikembangkan sebagai perbaikan dari algoritma ID3 oleh Ross Quinlan. Algoritma ini bekerja dengan cara membentuk pohon keputusan dari data latih dengan memanfaatkan ukuran *information gain ratio* yang berasal dari konsep *entropy*, sehingga dapat memilih atribut terbaik untuk membagi setiap simpul pada tree secara rekursif hingga mencapai klasifikasi yang optimal. C4.5 dapat menangani atribut numerik maupun kategorikal dan mampu menangani nilai yang hilang (*missing values*), serta menerapkan teknik *pruning* untuk mengurangi *over-fitting* sehingga model menjadi lebih general dan stabil terhadap data baru. Keunggulan C4.5 dibandingkan dengan metode decision tree sederhana adalah kemampuannya menghasilkan aturan yang interpretable dan akurat, sehingga sering digunakan dalam berbagai aplikasi seperti klasifikasi data kesehatan, penentuan rekomendasi keputusan, dan evaluasi faktor dominan dalam sebuah dataset kompleks. Dalam konteks pemilihan algoritma klasifikasi, penelitian menunjukkan bahwa C4.5 tetap relevan terutama ketika interpretabilitas model menjadi faktor penting, meskipun pada beberapa jenis data atau kebutuhan performa tinggi algoritma lain seperti Random Forest atau Naive Bayes mungkin memberikan hasil yang lebih unggul(Nazifah & Prianto, 2023)

Algoritma ini bekerja dengan membentuk pohon keputusan dari data latih melalui proses pemilihan atribut terbaik pada setiap simpul secara rekursif hingga diperoleh klasifikasi yang optimal. Pemilihan atribut pada C4.5 didasarkan pada nilai Gain Ratio, yang merupakan pengembangan dari *Information Gain* dan berasal dari konsep Entropy. Entropy digunakan untuk mengukur tingkat ketidakpastian dalam suatu himpunan data dan dirumuskan sebagai:

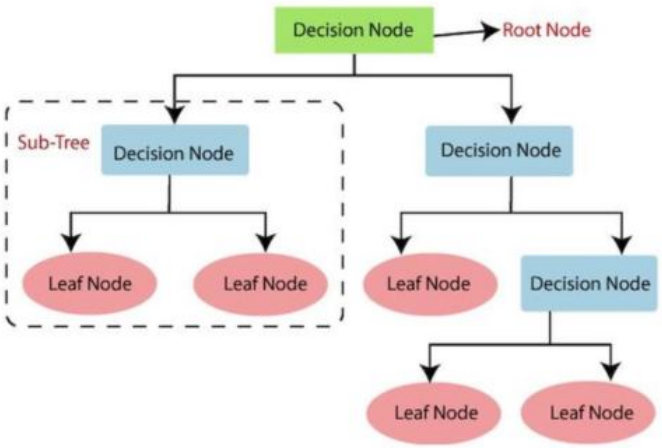
$$Entropy(S) = -\sum_{i=1}^{n} \binom{n}{k} \text{pi} log_2(pi) \quad (2.1)$$

Di mana $p_i$adalah proporsi data pada kelas ke-i dalam himpunan data S. Selanjutnya, *Information Gain* digunakan untuk mengukur pengurangan entropy setelah data dibagi berdasarkan suatu atribut A, yang dirumuskan sebagai:

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)}^{n} \frac{|Sv|}{|s|} \times Entropy(Sv) \quad (2.2)$$

Namun, untuk menghindari bias terhadap atribut dengan jumlah nilai yang banyak, C4.5 menggunakan Gain Ratio sebagai kriteria pemilihan atribut, yang dirumuskan sebagai:

$$GainRatio(S, A) = \frac{Gain(S,A)}{SplitInfo(S,A)} \quad (2.3)$$



Gambar 2. 2 Decision Tree
(Sumber : https://www.researchgate.net/figure/Gambar-1-Decision-Tree-A-Langkah-langkah-dalam-membuat-sebuah-Decision-Tree-1_fig1_359319567)

2.2.3 Odometry

Odometry merupakan metode dalam robotika untuk memperkirakan perubahan posisi dan orientasi robot berdasarkan data dari sensor internal, khususnya *wheel encoder*. Odometry *wheel encoder* menghitung jumlah putaran roda dan mengkonversinya menjadi perubahan posisi linear dan sudut rotasi

melalui model kinematika robot, sehingga estimasi posisi dapat dilakukan secara *real-time* tanpa bergantung pada sensor eksternal seperti LiDAR atau kamera. Berbagai studi menunjukkan bahwa odometry berbasis *wheel encoder* efektif untuk navigasi lokal ketika kondisi lingkungan tidak memerlukan pemetaan lengkap, meskipun akurasi dapat menurun akibat *drift* atau slip roda seiring waktu. Dalam aplikasi navigasi robot mobile otonom, data odometry digunakan untuk membangun jalur perjalanan dan estimasi posisi mutakhir, yang dapat diintegrasikan dengan sistem monitoring visual seperti peta yang ditampilkan pada antarmuka web, memungkinkan pengguna untuk melihat posisi robot aktual secara *real-time*. Penelitian oleh (Ridho et al., 2024) pada robot bergerak *differential drive* menekankan bahwa algoritma odometry berbasis *wheel encoder* mampu memberikan estimasi posisi yang memadai dalam skenario indoor, dengan akurasi yang dapat ditingkatkan melalui penggabungan data sensor tambahan seperti IMU (Inertial Measurement Unit) untuk mengurangi efek error kumulatif. Selain itu, studi (Nursyeha et al., 2023) membahas bahwa penggunaan sensor *wheel encoder* sebagai sumber utama odometry pada robot mobile terintegrasi dengan sistem kontrol memungkinkan estimasi posisi dan orientasi yang memadai untuk navigasi dan visi sistem yang dapat diadaptasi pada aplikasi monitoring web tanpa perlu sensor jarak eksternal.

2.2.4 RosBridge Web Socket

ROSBridge WebSocket bekerja sebagai penerjemah protokol antara dunia ROS dan dunia web. ROSBridge adalah node ROS biasa yang dijalankan di Jetson Nano, bedanya node ini punya kemampuan khusus: dia membuka sebuah WebSocket Server di port 9090 sambil tetap terhubung ke dalam ekosistem ROS. Semua pesan yang melewati ROSBridge diformat dalam JSON murni melalui WebSocket. Ini yang membuat browser bisa membacanya karena JSON adalah bahasa universal web. Salah satu contoh penerapan Rosbridge dalam sistem yang melibatkan robot dan antarmuka web adalah penelitian yang mengembangkan sistem Web-based IoT and Robot SCADA menggunakan protokol Rosbridge_Websocket untuk menghubungkan perangkat IoT dan robot dengan sistem SCADA berbasis web. Sistem ini memanfaatkan karakteristik RosBridge

Websocket yang efisien dalam pertukaran pesan untuk memfasilitasi komunikasi data dari robot atau perangkat industri ke antarmuka pengguna secara real-time, termasuk pengiriman status dan kontrol peralatan melalui topik-topik tertentu yang dikelola oleh broker Rosbridge(Guo et al., 2022).

ROSBridge hanya mengenal tiga operasi dasar yang perlu dipahami:

1. Subscribe — browser minta dikirim data setiap ada update topic. Browser → ROSBridge: *"saya mau dengarkan /imu/euler"* ROSBridge → Browser: setiap ada data baru, langsung dikirim.
2. Publish — browser mengirim perintah ke robot. Browser → ROSBridge: *"kirimkan ini ke /cmd_vel"* ROSBridge → ROS: meneruskan sebagai pesan ROS asli.
3. Service Call — browser meminta satu respons (bukan streaming). Browser → ROSBridge: *"panggil service /reset_encoder"* ROSBridge → ROS: panggil service, tunggu jawaban, kirim balik ke browser.

2.2.5 NODE.JS

Node.js merupakan sebuah platform runtime JavaScript yang berjalan di sisi server dan dibangun di atas mesin JavaScript V8. Node.js memungkinkan pengembang menjalankan kode JavaScript di luar browser, sehingga dapat digunakan untuk membangun aplikasi backend yang bersifat event-driven dan non-blocking I/O. Karakteristik ini menjadikan Node.js sangat sesuai untuk aplikasi yang membutuhkan penanganan data secara real-time dan komunikasi berkelanjutan, seperti sistem Internet of Things (IoT) dan Warehouse Management System (WMS) terintegrasi robot(Shcherbakov et al., n.d.).

Salah satu keunggulan utama Node.js adalah kemampuannya dalam menangani banyak koneksi secara simultan dengan penggunaan sumber daya yang efisien. Mekanisme non-blocking memungkinkan Node.js menerima dan memproses data dari berbagai sumber, seperti pesan Rosbridge_Websocket dari ESP32, tanpa menyebabkan penundaan pada proses lainnya. Hal ini penting dalam sistem monitoring dan pengendalian Autonomous Mobile Robot, di mana data sensor, status robot, dan perintah kontrol harus diproses secara cepat dan berkelanjutan(Agus et al., 2023).

Dalam penelitian ini, Node.js digunakan sebagai backend server yang berfungsi untuk menerima data dari Rosbridge_Websocket, melakukan pengolahan data, menjalankan logika sistem seperti decision tree untuk penentuan prioritas pengantaran barang, serta menyimpan data ke dalam database. Selain itu, Node.js juga menyediakan layanan API yang digunakan oleh frontend berbasis React untuk menampilkan informasi monitoring dan menerima input dari pengguna. Dengan arsitektur ini, Node.js berperan sebagai pusat integrasi antara perangkat keras robot, sistem komunikasi Rosbridge_Websocket, database, dan antarmuka web.

Gambar 2. 3 Node.js
(Sumber : https://en.wikipedia.org/wiki/Node.js)

2.3 *Hardware*

Berikut adalah hardware yang akan digunakan dalam penilitian ini :

2.3.1 Sensor *Rotary Encoder*

Rotary Encoder adalah perangkat elektronik yang dapat mengubah gerakan rotasi atau perubahan sudut poros menjadi sinyal digital. Alat ini digunakan untuk menemukan koordinat X dan Y dalam penelitian ini. Rotary Encoder biasanya terbagi menjadi dua kategori utama: encoder inkremental dan encoder absolut. Kategori ini dibedakan berdasarkan sifat sinyal yang dihasilkannya.Rotary Encoder tipe inkremental yang digunakan dalam penelitian ini menghasilkan sinyal dalam bentuk pulsa gelombang kotak saat poros bergerak. Pulsa-pulsa ini menunjukkan perubahan posisi dibandingkan dengan titik awal, sehingga dapat digunakan untuk menghitung perpindahan robot dalam sistem koordinat dua dimensi.

Gambar 2. 4 Rotary Encoder
(Sumber : https://id.images.search.yahoo.com)

Tabel 2. 2 Spesifikasi Sensor Rotary Encoder

Attribute	***Value***
Voltage	9-36V
Current	<25mA
Range	Z : ±180°
Accuracy	0.1°
Stability	0.01°
Return rate	0.1~1000Hz (*default* 10Hz)
Baud rate	4800~921600 (*default* 9600)

2.3.2 Sensor Tegangan

Dalam penelitian ini, sensor tegangan dengan metode pembagi tegangan digunakan untuk mengukur tegangan tinggi dengan menurunkannya ke level tegangan yang aman bagi mikrokontroler. Pengukuran ini dimulai dengan tegangan sumber sebesar 24 V, yang kemudian diturunkan hingga mencapai rentang maksimal 3,3 V, yang sesuai dengan batas tegangan input Pembagi tegangan bekerja dengan dua resistor yang disusun secara seri. Tegangan keluaran diambil pada titik tengah antara dua resistor. Nilai resistor dibandingkan untuk mengetahui besarnya tegangan keluaran (V_out).

Keunggulan metode pembagi tegangan termasuk rangkaian yang sederhana, biaya implementasi yang rendah, dan integrasi yang mudah dengan sistem mikrokontroler. Namun, metode ini memiliki beberapa keterbatasan. Nilai resistor sensitif terhadap perubahan dan tidak memberikan isolasi listrik antara rangkaian pengukuran dan sumber tegangan. Oleh karena itu, nilai resistor dan toleransinya harus dipilih dengan hati-hati agar hasil pengukuran tetap akurat dan aman.

2.3.3 Sensor *Limit Switch*

Dalam penelitian ini, penghalang batas digunakan sebagai sensor batas untuk mengidentifikasi posisi maksimum atau minimum dari pergerakan mekanis pada sistem robot. Penghalang batas adalah perangkat elektromekanis yang bekerja dengan prinsip kontak mekanik, di mana ketika tuas atau aktuator penghalang tertekan oleh objek tertentu, kondisi berubah. Kondisi ini mengubah status kontak listrik dari keadaan terbuka. (*Normally Open*/NO) menjadi tertutup, maupun sebaliknya (*Normally Closed*/NC).

Gambar 2. 5 *Limit Switch*
(Sumber : Dokumentasi Penulis,2025)

Fungsi utama limit switch dalam penelitian ini adalah untuk memberikan informasi batas gerak kepada sistem kontrol, sehingga pergerakan robot atau mekanisme tertentu dapat dihentikan atau dibatasi ketika telah mencapai posisi yang ditentukan. Dengan demikian, penggunaan limit switch berperan penting dalam mencegah terjadinya gerakan berlebih yang berpotensi merusak komponen mekanis maupun elektronik.

2.3.4 Sensor HWT901B

Dalam penelitian ini, penentuan orientasi dan arah hadap robot menggunakan sensor HWT901B sebagai perangkat utama. Sensor HW901B merupakan jenis sensor attitude yang bekerja berdasarkan prinsip Inertial Measurement Unit (IMU) untuk mengukur perubahan kecepatan angular pada sumbu vertikal atau sumbu Z. Pengukuran pada sumbu Z ini menunjukkan rotasi robot terhadap bidang datar atau yang dikenal sebagai sudut yaw, dimana informasi ini sangat krusial untuk menentukan heading direction robot saat bernavigasi dalam lingkungan kerja.

Gambar 2. 6 Sensor HWT901B
(Sumber: alibaba.com)

Tabel 2. 3 Spesifikasi Sensor HWT901B

Attribute	***Value***
Voltage	9-36V
Current	<40mA
Range	X/Z-axis: +-180 degree, Y-axis +-90 degree
Accuracy	X/ Y axis 0.05 degree, Z axis 1 degree
Stability	0.01g, Angular speed 0.05degree/second
Return rate	0.1~1000Hz (*default* 10Hz)
Baud rate	4800~921600 (*default* 9600)

Salah satu keunggulan utama sensor HWT901B adalah kemampuannya dalam menghitung sudut rotasi secara langsung melalui algoritma internal yang tertanam di dalam modul sensor, sehingga hasil pengukuran tetap stabil dan akurat meskipun robot berada dalam kondisi bergerak (Motion, 2024). Selain itu, bentuk fisik sensor yang ringkas menjadikannya mudah diintegrasikan dengan sistem mikrokontroler yang digunakan. Adapun spesifikasi teknis sensor HWT901B disajikan pada

2.3.6 Motor DC PG45

Motor penggerak utama yang digunakan dalam penelitian ini adalah Motor DC PG-45, Motor DC PG-45 termasuk dalam kategori motor DC brushed yang dirancang untuk menghasilkan torsi tinggi. Salah satu

keunggulan utama motor ini adalah keberadaan gearbox planetary yang terintegrasi secara langsung pada motor, sehingga mampu meningkatkan kemampuan torsi sekaligus efisiensi kerja. Kombinasi antara motor dan gearbox tersebut memungkinkan Motor DC PG-45 menghasilkan performa yang stabil serta keandalan yang baik dalam mendukung pergerakan robot pada berbagai kondisi operasional. Rincian spesifikasi teknis Motor DC PG-45.

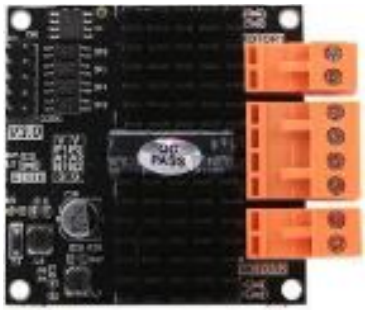
Gambar 2. 7 Motor DC
(Sumber : Dokumentasi Penulis,2025)

Tabel 2. 4 Spesifikasi Motor DC

No.	Fitur	Spesifikasi
1.	Kecepatan tanpa beban	900 rpm
2.	Voltage Power	12 Vdc
3.	Arus tanpa beban	0.610 A
4.	Daya tanpa beban	14.56 W
5.	Tegangan daya maksimum	23.47v
6.	Arus daya maksimum	12.10 A
7.	Torsi maksimum	2,2 kgf'cm/0,22 Nm
8.	Kecepatan beban maksimal	RPM
9.	Daya maksimum	109.2 W

2.3.7 *Driver* Motor

Driver motor adalah perangkat elektronik yang berfungsi sebagai penghubung antara sistem kendali dan motor listrik. Mereka memulai dengan menerima sinyal kendali berdaya rendah dari pengendali, seperti PLC atau mikrokontroler, dan kemudian mengolah dan memperkuat sinyal tersebut sehingga mereka dapat menggerakkan motor dengan parameter seperti arah putaran, kecepatan, dan torsi.



Gambar 2. 8 Driver Motor
(Sumber : Dokumentasi Penulis,2025)

Driver motor ini banyak digunakan dalam berbagai proyek robotika, elektronika, kendaraan listrik, bahkan mesin industri. Hal ini disebabkan oleh fakta bahwa driver motor ini memiliki kapasitas yang sangat baik untuk digunakan untuk mengontrol putaran motor dengan kapasitas sampai dengan 50A. Driver motor ini memiliki satu terminal PWM, yang memungkinkan pengguna untuk mengatur kecepatan putaran motor yang dikontrol. Pengaturan PWM dapat dilakukan dengan potensiometer atau sinyal kontrol eksternal. Driver ini tidak hanya dapat mengontrol kecepatan tetapi juga dapat mengontrol arah putaran motor dengan cepat dan mudah. Spesifikasi driver motor ini tercantum di bawah ini.

Tabel 2. 5 Tabel Driver Motor

No.	Fitur	Spesifikasi
1	Power	1Kw
2	Voltage Power	24Vdc - 48Vdc
3	Frekuensi	1Khz - 35Khz
4	Support	PG Motor Family (PG22.PG28,PG36, PG45,PG56)

Driver motor ini banyak digunakan dalam berbagai proyek robotika, elektronika, kendaraan listrik, bahkan mesin industri. Hal ini disebabkan oleh fakta bahwa driver motor ini memiliki kapasitas yang sangat baik untuk digunakan untuk mengontrol putaran motor dengan kapasitas sampai dengan 50A. Driver motor ini memiliki satu terminal PWM, yang memungkinkan pengguna untuk mengatur kecepatan putaran motor yang dikontrol. Pengaturan PWM dapat dilakukan dengan potensiometer atau sinyal kontrol

eksternal. Driver ini tidak hanya dapat mengontrol kecepatan tetapi juga dapat mengontrol arah putaran motor dengan cepat dan mudah. Spesifikasi driver motor ini tercantum di Tabel 2.5.

2.3.9 Led Indikator

LED indikator merupakan komponen elektronika berbasis Light Emitting Diode yang berfungsi sebagai penanda visual untuk menunjukkan kondisi atau status suatu sistem. LED indikator akan memancarkan cahaya ketika dialiri arus listrik dengan polaritas yang benar. Dalam sistem kendali dan otomasi, LED indikator digunakan untuk memberikan informasi secara visual mengenai keadaan operasi sistem, seperti kondisi aktif/nonaktif, status kesalahan (fault), proses berjalan, atau kondisi siaga. Penggunaan LED indikator bertujuan untuk mempermudah pemantauan sistem, meningkatkan keselamatan kerja, serta membantu proses diagnosis dan pemeliharaan peralatan

2.3.10 Linier Aktuator

Linear actuator merupakan komponen aktuator mekanik yang menghasilkan gerakan linier (maju–mundur) pada suatu sistem, berbeda dengan motor konvensional yang menghasilkan gerakan putar. Aktuator ini banyak digunakan dalam berbagai aplikasi otomasi dan robotika karena kemampuannya untuk memindahkan beban secara linear dengan presisi tinggi, seperti dalam penggerak lengan robot, pengaturan posisi objek, dan mekanisme pemindahan material pada lini produksi.

Gambar 2. 9 Linier Aktuator
(Sumber : Documentasi Penulis, 2025)

Linear actuator tersedia dalam berbagai tipe termasuk elektrik, pneumatik, dan hidrolik, dan sering kali dipasangkan dengan sistem kontrol untuk mengatur posisi akhir dan kecepatan geraknya sesuai kebutuhan sistem otomasi. Dalam konteks teknik otomasi dan robotika, penggunaan linear actuator sangat penting karena dapat memberikan kontrol posisi yang akurat dan responsif, yang diperlukan misalnya untuk proses *pick-and-place*, penyesuaian posisi komponen, atau pengaturan mekanisme kerja robot secara otomatis.

2.3.11 *Router*

Dalam sistem kontrol dan monitoring Autonomous Mobile Robot (AMR) yang terintegrasi dengan Warehouse Management System (WMS), *router* berperan sebagai infrastruktur jaringan utama yang menghubungkan berbagai perangkat dalam satu jaringan lokal (LAN). Router menyediakan konektivitas antara perangkat kontrol robot (ESP32), server backend (laptop), serta perangkat klien seperti laptop operator atau user yang mengakses web monitoring. Router memastikan paket data dari ESP32, Rosbridge_Websocket, dan server backend dapat saling bertukar informasi secara efisien dan konsisten dalam satu jaringan tanpa memerlukan koneksi internet eksternal. Dengan demikian, *router* bertindak sebagai *gateway* yang mengarahkan dan meneruskan paket data antar node di jaringan lokal, sekaligus memberikan alamat IP otomatis melalui DHCP untuk setiap perangkat yang terhubung sehingga memudahkan komunikasi tanpa konfigurasi manual yang kompleks. Router juga mengatur distribusi lalu lintas data secara optimal agar pertukaran informasi, termasuk pesan Rosbridge_Websocket dan respon web, berjalan lancar dalam sistem yang bersifat real-time. Secara fungsional, *router* tidak terlibat dalam pengambilan keputusan atau pemrosesan data aplikasi, tetapi menjadi elemen penting dalam menjaga integrasi dan keterhubungan antar komponen sistem AMR–WMS berbasis IoT(Hairun et al., 2023).

Router pada jaringan IoT seperti pada sistem AMR–WMS beroperasi di lapisan jaringan (Layer 3) dan bertugas untuk meneruskan paket data ke alamat tujuan berdasarkan tabel routing serta pengalamatan IP yang valid. Perangkat ini memungkinkan semua node IoT untuk berada dalam satu segmen jaringan yang

sama sehingga dapat berkomunikasi secara langsung melalui protokol komunikasi ringan seperti Rosbridge_Websocket tanpa hambatan lalu lintas atau konflik alamat IP. Dengan hadirnya *router*, sensor, aktuator, dan kontroler dapat saling bertukar informasi secara lokal, yang penting untuk memastikan keterhubungan data odometri, status robot, dan perintah kontrol yang dikirim dari web ke robot, sehingga sistem dapat berjalan secara sinkron dan responsif.

Gambar 2. 10 Router
(Sumber : https://ontolt.com/product/huawei-network-terminal-hg8546m-eg8141a5/)

2.3.12 Mini PC Jetson Orin Nano

NVIDIA Jetson Orin Nano merupakan *single-board computer* (SBC) yang dikembangkan oleh NVIDIA untuk kebutuhan komputasi *edge AI* (*Artificial Intelligence*) dan sistem *embedded*. Perangkat ini dirancang untuk menjalankan aplikasi yang membutuhkan kemampuan pemrosesan tinggi, seperti pengolahan citra (*computer vision*), *machine learning*, robotika, serta sistem otomasi industri. Jetson Orin Nano mengintegrasikan CPU berbasis Arm Cortex-A78AE dan GPU NVIDIA Ampere dengan dukungan Tensor Cores, sehingga mampu menjalankan berbagai algoritma kecerdasan buatan dengan konsumsi daya yang relatif rendah.

Jetson Orin Nano memiliki kemampuan komputasi hingga 40 TOPS (*Trillion Operations Per Second*), memori LPDDR5 sebesar 8 GB, serta mendukung berbagai antarmuka komunikasi seperti Gigabit Ethernet, USB 3.2, HDMI, MIPI CSI Camera Interface, GPIO, I²C, SPI, UART, dan USB Type-C. Perangkat ini menjalankan sistem operasi Linux melalui *NVIDIA JetPack SDK* yang telah dilengkapi dengan pustaka CUDA, cuDNN, TensorRT, OpenCV, serta framework *deep learning* seperti TensorFlow dan PyTorch. Dukungan tersebut

menjadikan Jetson Orin Nano sebagai salah satu platform yang banyak digunakan pada pengembangan robot otonom dan aplikasi *edge computing*.

Gambar 2. 11 NVIDIA Jetson Orin Nano
(Sumber: alibaba.com)

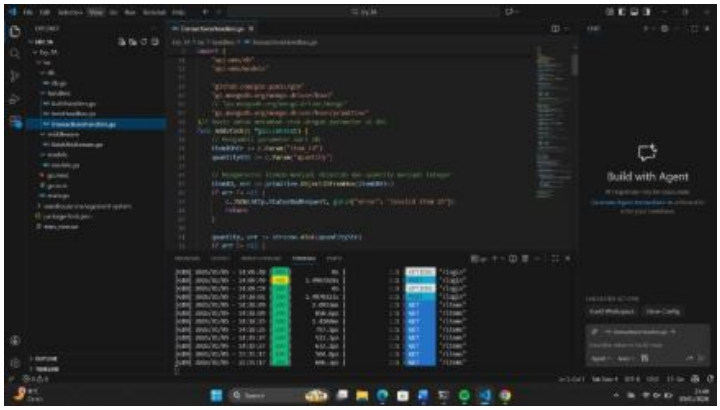
2.4 *Software*

Dalam penelitian ini software yang akan digunakan untuk mengoperasikan robot dan pengelolaan data dari robot yang akan di tampilkan di website, berikut adalah software yang akan digunakan:

2.4.1 Visual Studio Code

Visual Studio Code (VS Code) merupakan code editor yang dikembangkan oleh Microsoft dan banyak digunakan dalam pengembangan perangkat lunak karena bersifat ringan, lintas platform, serta mendukung berbagai bahasa pemrograman. Pada penelitian ini, Visual Studio Code digunakan sebagai lingkungan pengembangan utama (Integrated Development Environment/IDE) untuk pembuatan, pengujian, dan pengelolaan kode program sistem robot.

Dalam penelitian ini, VS code digunakan untuk pembuatan dan Pengembangan Web yang nantinya akan digunakan untuk control dan monitoring data- data dari robot AMR. Dengan berbagai fitur yang disediakan di VS Code ini, proses dan Pengembangan Web nantinya akan jauh lebih mudah, lebih terstruktur dan efisien dan terintegrasi.



Gambar 2. 12 Visual Studio Code
(Sumber : Dokumentasi Penulis,2026)

2.4.3 Three.js

Unity adalah salah satu platform pengembangan perangkat lunak yang populer untuk pembuatan simulasi 2D maupun 3D, termasuk dalam bidang robotika dan sistem otomasi. Unity menyediakan engine visualisasi real-time yang mendukung integrasi data eksternal, sehingga dapat digunakan untuk menampilkan peta lingkungan berdasarkan informasi posisi robot yang diperoleh dari sensor odometri dan HWT. Dengan Unity, data posisi robot yang terus diperbarui secara real-time dapat divisualisasikan menjadi representasi peta interaktif, termasuk jalur pergerakan robot, lokasi terminal pick-up/drop, dan status objek di sekitar robot. Hal ini menjadikan Unity sangat cocok untuk membangun digital twin dari gudang, di mana pengguna dapat memantau aktivitas robot dan pengantaran barang melalui antarmuka visual yang intuitif dan interaktif.

Dalam implementasi WMS berbasis robot, data odometri yang diterima dari robot diproses oleh backend sistem dan dikirim ke Unity sebagai data posisi (misalnya koordinat x, y, orientasi θ). Unity kemudian menggunakan data tersebut untuk menampilkan peta lingkungan secara dinamis, sehingga operator atau pengguna dapat melihat jalur aktual robot, terminal pick-up/drop, dan status pengantaran barang secara real-time. Pendekatan ini memungkinkan monitoring dan evaluasi performa robot tanpa perlu langsung berada di lokasi gudang, serta mempermudah analisis efisiensi pengantaran barang.

Keunggulan utama Unity dibandingkan platform visualisasi lain adalah kemampuannya untuk melakukan rendering real-time yang interaktif, dukungan untuk integrasi dengan berbagai bahasa pemrograman seperti C# dan Python, serta kemampuan menampilkan peta 2D maupun 3D. Dengan demikian, Unity tidak hanya menjadi alat simulasi, tetapi juga platform visualisasi digital twin yang dapat menghubungkan data dunia nyata dari robot dengan representasi virtual di web atau aplikasi desktop(Nambiar & Mundra, 2022).

Dalam konteks penelitian ini, Unity digunakan untuk memvisualisasikan peta gudang berdasarkan data odometri robot AMR, sehingga sistem WMS berbasis web tidak hanya mampu melakukan monitoring dan pengelolaan prioritas pengantaran barang (FIFO/FEFO), tetapi juga memberikan representasi visual jalur robot, terminal pick-up/drop, dan status pengantaran secara real-time. Pendekatan ini meningkatkan efektivitas pengawasan operasional dan mendukung pengambilan keputusan berbasis data secara lebih akurat.

Gambar 2. 13 Unity
(Sumber : https://eventkampus.com/blog/detail/1474/apa-itu-unity-3d)

2.4.4 MongoDB

MongoDB adalah salah satu jenis basis data NoSQL (Not Only SQL) yang dirancang untuk menyimpan data semi-terstruktur dalam bentuk dokumen JSON/BSON, sehingga sangat cocok untuk aplikasi yang memerlukan fleksibilitas model data, skalabilitas tinggi, dan kemampuan menangani data dalam jumlah besar yang berubah-ubah secara dinamis. Berbeda dengan basis data relasional tradisional seperti MySQL yang mengandalkan tabel dan skema tetap, MongoDB memungkinkan

penyimpanan dokumen dengan struktur yang bervariasi tanpa memerlukan migrasi skema yang kompleks. Keunggulan ini membuat MongoDB sangat populer dalam pengembangan sistem Internet of Things (*IoT*) dan aplikasi real-time, di mana data yang dihasilkan perangkat sering beragam, tidak teratur, dan harus diproses serta ditampilkan secara cepat. Selain itu, MongoDB mendukung operasi baca/tulis yang cepat dengan indeks yang dapat dikonfigurasi, replikasi antar server untuk keandalan, serta kemampuan *sharding* untuk skalabilitas horizontal di lingkungan produksi(Studies, 2024).

Gambar 2. 14 MongoDB
(Sumber : https://www.developer-tech.com/news/mongodb-deepens-google-cloud-partnership-new-atlas-integrations/)

(Halaman ini sengaja dikosongkan)

BAB 3
METODE PENELITIAN

3.1 Konsep Penelitian

Penelitian ini dilatarbelakangi oleh kebutuhan akan sistem logistik gudang yang mampu memindahkan barang secara cepat, tepat, dan otomatis, khususnya pada skema cross docking, di mana barang yang diterima harus segera disalurkan menuju titik pengiriman tanpa melalui proses penyimpanan yang lama. Untuk menjawab kebutuhan tersebut, penelitian ini merancang dan membangun sebuah AMR *Control Station* berbasis web yang berfungsi sebagai pusat kendali bagi Autonomous Mobile Robot (AMR) dalam mensimulasikan proses *cross docking* pada lingkungan gudang berskala kecil.

Secara konsep, penelitian ini mengintegrasikan tiga lapisan sistem yang saling berkomunikasi secara *real-time*, yaitu lapisan antarmuka kendali berbasis web (*frontend*), lapisan pengambilan keputusan dan penyimpanan data (*backend*), serta lapisan kendali fisik robot (*hardware*). Lapisan *frontend* berperan sebagai antarmuka bagi operator untuk mengirim permintaan pengiriman barang, memantau posisi robot secara visual melalui representasi peta tiga dimensi, serta mengendalikan robot secara manual apabila diperlukan. Lapisan *backend* menyimpan data permintaan pengiriman ke dalam basis data MongoDB dan menjalankan algoritma Decision Tree C4.5 untuk menentukan prioritas terminal pick-up yang harus ditangani terlebih dahulu, berdasarkan atribut jenis barang, jarak tempuh, dan waktu tunggu permintaan. Adapun lapisan hardware terdiri atas single-board computer Jetson yang menjalankan Robot Operating System (ROS1) sebagai pengendali utama navigasi robot, serta mikrokontroler ESP32 yang menangani pembacaan sensor dan penggerakan aktuator, termasuk motor penggerak dan mekanisme pengangkat barang (lifter).

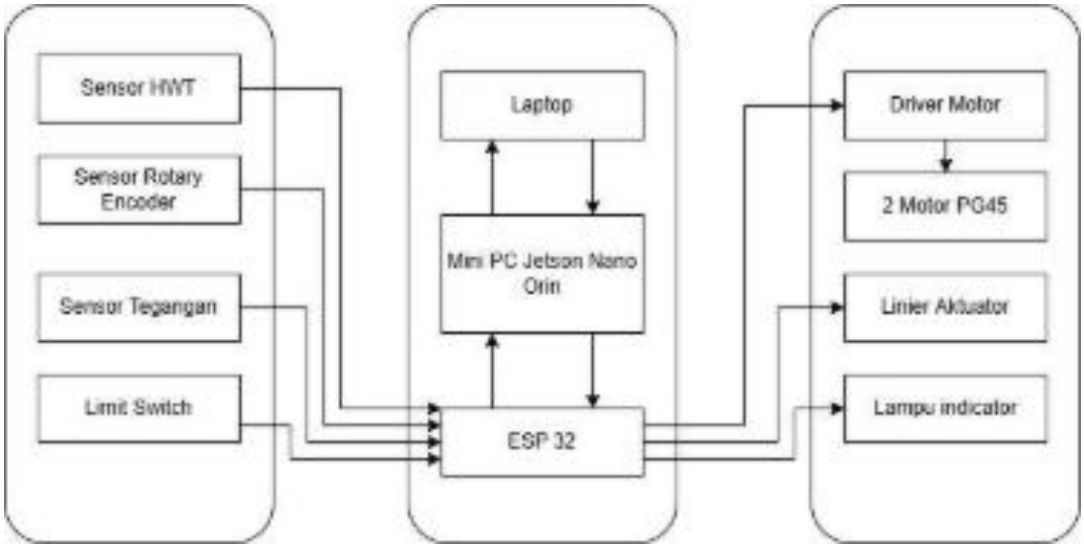
Komunikasi antara lapisan frontend dan lapisan ROS pada Jetson dijembatani menggunakan protokol ROSBridge WebSocket, sehingga perintah yang dikirim melalui browser dapat diteruskan secara langsung ke robot tanpa

memerlukan aplikasi klien khusus. Pendekatan ini dipilih agar sistem kendali dapat diakses secara fleksibel selama perangkat berada pada jaringan yang sama, sekaligus memungkinkan pemantauan posisi dan status robot secara real-time melalui antarmuka web. Proses navigasi robot menuju terminal yang dituju diselesaikan menggunakan algoritma pencarian jalur A* (A-star) pada representasi grid gudang, sedangkan penentuan urutan prioritas antar-permintaan yang datang secara bersamaan diselesaikan menggunakan algoritma Decision Tree C4.5.

Dengan konsep tersebut, penelitian ini tidak hanya berfokus pada aspek pergerakan fisik robot, tetapi juga pada aspek pengambilan keputusan yang mendasari urutan pengerjaan tugas, serta keandalan integrasi data antara antarmuka web, basis data, dan robot secara keseluruhan. Konsep penelitian ini kemudian dijabarkan lebih lanjut melalui tahapan perancangan sistem, implementasi algoritma, serta pengujian yang dijelaskan pada subbab-subbab berikutnya dalam bab ini.

3.1.1 Diagram Blok

Diagram blok sistem pada penelitian ini terdiri dari tiga bagian utama, yaitu blok input, blok proses, dan blok output, yang saling terintegrasi untuk mendukung sistem pengendalian dan pemantauan Autonomous Mobile Robot (AMR).



Gambar 3. 1 Diagram Blok

Gambar 3.1 menunjukkan diagram blok perangkat keras yang digunakan pada sistem, yang terbagi menjadi tiga kelompok utama, yaitu kelompok sensor (masukan), kelompok pemroses (*processing unit*), dan kelompok aktuator (keluaran).

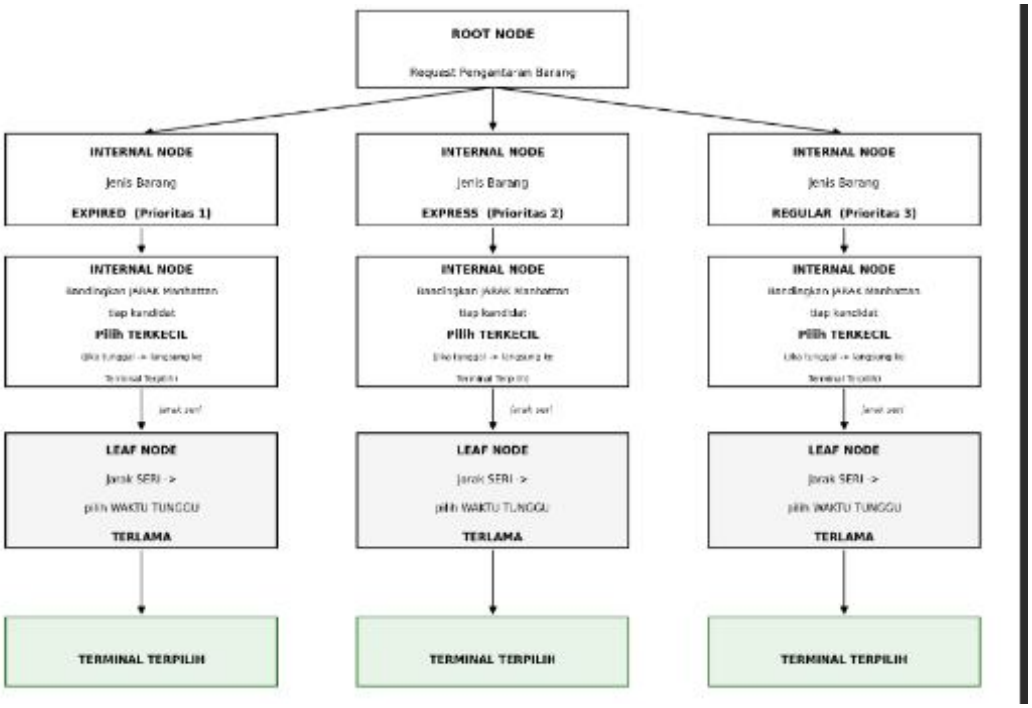
Kelompok pertama merupakan kelompok sensor, yang terdiri atas empat jenis sensor sebagai masukan sistem. Sensor HWT berfungsi sebagai sensor inersial (IMU) yang membaca data orientasi robot, meliputi sudut *roll*, *pitch*, dan *yaw*. Sensor *Rotary* Encoder membaca jumlah pulsa putaran roda yang digunakan untuk menghitung jarak tempuh robot melalui perhitungan *single wheel odometry*. Sensor Tegangan membaca tegangan baterai melalui rangkaian pembagi tegangan untuk keperluan pemantauan kondisi daya robot secara *real-time*. Adapun *Limit Switch* berfungsi sebagai sensor batas mekanis yang mendeteksi posisi maksimum pergerakan mekanisme *lifter*. Keempat sensor tersebut mengirimkan data pembacaannya secara langsung menuju mikrokontroler ESP32.

Kelompok kedua merupakan kelompok pemroses, yang terdiri atas Laptop, Mini PC Jetson Nano Orin, dan ESP32. Laptop terhubung secara dua arah dengan Jetson Nano Orin, yang digunakan pada tahap pengembangan dan pemantauan sistem, seperti pemrograman, pemantauan log, serta pengujian melalui antarmuka *terminal*. Jetson Nano Orin berperan sebagai pengendali utama sistem yang menjalankan *Robot Operating System* (ROS1), termasuk algoritma navigasi A* dan orkestrasi proses pengantaran barang. Jetson Nano Orin terhubung secara dua arah dengan ESP32 melalui komunikasi serial, yaitu mengirimkan perintah pergerakan dan kendali aktuator, serta menerima data hasil pembacaan sensor dari ESP32.

Kelompok ketiga merupakan kelompok aktuator, yang dikendalikan langsung oleh ESP32. Driver Motor menerima sinyal kendali dari ESP32 untuk menggerakkan 2 Motor PG45 sebagai motor penggerak utama roda robot. Linier Aktuator dikendalikan oleh ESP32 melalui modul relay untuk menjalankan mekanisme *lifter*, yaitu pengangkatan dan penurunan barang pada proses *pick-up* dan *drop-off*. Adapun Lampu Indikator dikendalikan oleh ESP32 untuk menampilkan status operasional robot secara visual kepada pengguna di sekitar area kerja robot.

Dengan diagram blok tersebut, ESP32 berperan sebagai pusat penghubung (*hub*) antara seluruh sensor dan aktuator pada sisi perangkat keras, sekaligus sebagai jembatan komunikasi menuju Jetson Nano Orin yang menangani pemrosesan tingkat tinggi, seperti algoritma navigasi dan pengambilan keputusan sistem.

3.1.2 Diagram Decision Tree Pada WMS



Gambar 3. 2 Diagram Blok Diagram Decision Tree Pada WMS
(Sumber: Dokumentasi Penulis 2026)

Struktur pohon keputusan yang diterapkan pada sistem ini ditunjukkan pada Gambar 3.2 Root node merepresentasikan titik awal proses pengambilan keputusan, yaitu setiap permintaan pengantaran barang (*request*) yang diajukan oleh operator melalui *website*. Dari *root node* ini, sistem mulai menelusuri pohon keputusan untuk menentukan terminal *pick-up* yang harus diprioritaskan.

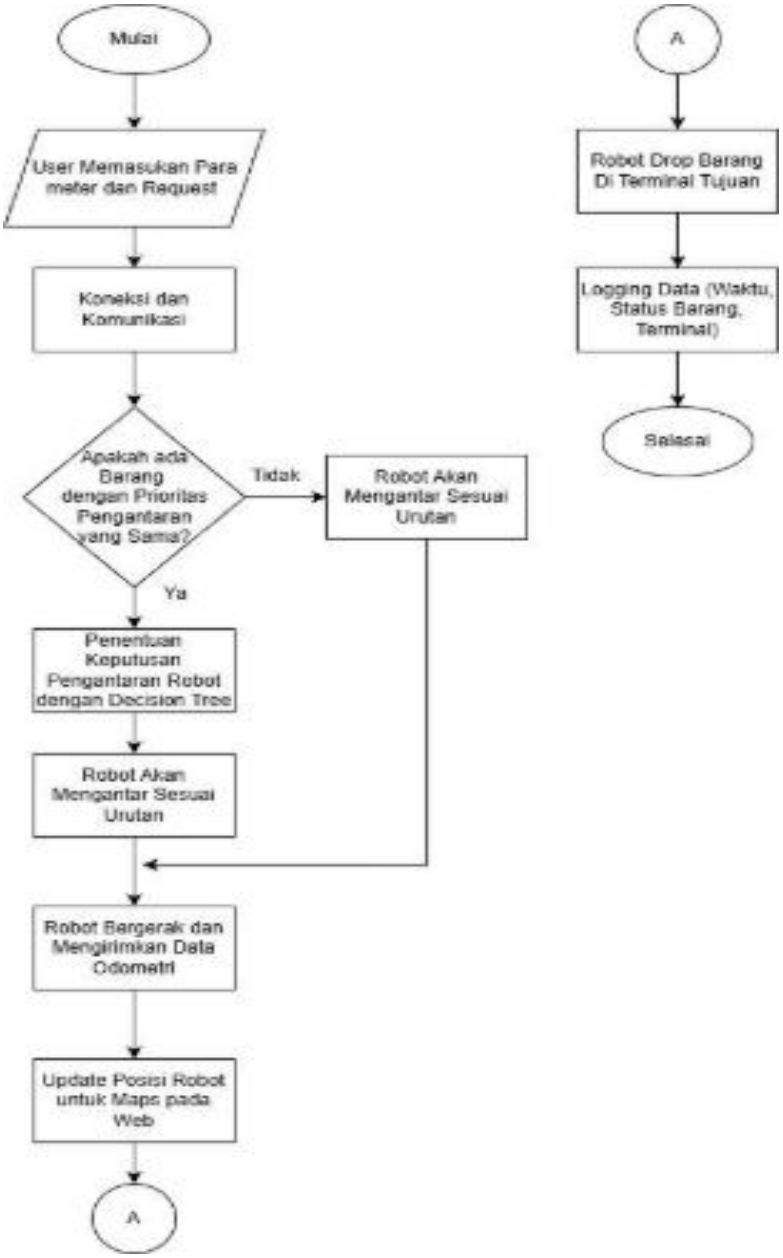
Penelusuran pertama dilakukan pada internal node tingkat pertama, yaitu atribut jenis barang, yang membagi permintaan ke dalam tiga cabang sesuai kelas prioritas: *Expired* (Prioritas 1), *Express* (Prioritas 2), dan *Regular* (Prioritas 3). Atribut jenis barang ditempatkan sebagai *root* karena memiliki nilai *Gain Ratio* tertinggi (1,000), sebagaimana telah dibuktikan pada subbab 4.3.5.1, yang berarti atribut ini memiliki daya pisah terbaik dalam menentukan prioritas dibandingkan atribut lainnya.

Apabila terdapat lebih dari satu kandidat pada kelas prioritas yang sama, penelusuran dilanjutkan ke internal node tingkat kedua, yaitu atribut jarak. Pada simpul ini, sistem membandingkan jarak Manhattan setiap kandidat terhadap posisi robot dan memilih kandidat dengan jarak terkecil (terdekat). Apabila hanya terdapat satu kandidat dengan jarak terkecil, penelusuran langsung berakhir pada terminal yang dipilih. Atribut jarak ditempatkan pada tingkat kedua karena memiliki nilai *Gain Ratio* lebih tinggi dibandingkan atribut waktu tunggu.

Apabila nilai jarak antarkandidat juga bernilai sama, penelusuran dilanjutkan hingga mencapai leaf node, yaitu simpul daun pada tingkat ketiga yang merupakan simpul akhir dari pohon keputusan. Pada simpul ini, sistem membandingkan atribut waktu tunggu setiap kandidat dan memilih kandidat dengan waktu tunggu paling lama, dengan tujuan mencegah kondisi *starvation*, yaitu permintaan yang diajukan lebih awal terus-menerus tertunda oleh permintaan yang datang belakangan. Hasil akhir dari penelusuran pohon keputusan, baik yang berhenti pada simpul jenis barang, jarak, maupun waktu tunggu, akan menghasilkan satu terminal terpilih yang menjadi prioritas pengantaran robot berikutnya.

3.1.3 *Flowchart* Sistem

Flowchart ini menggambarkan proses kerja robot dan website dari awal hingga akhir proses penndistribusian barang. Selain itu, *flowchart* juga menggambarkan pengambilan keputusan ketika terjadi perubahan lingkungan atau muncul *obstacle* yang memblokir jalur, sehingga sistem dapat melakukan *replanning* secara otomatis. Alur sistem mencakup proses pengiriman perintah dari *Warehouse Management System* (WMS) melalui website, penerimaan perintah oleh *Autonomous Mobile Robot* (AMR), pelaksanaan proses perpindahan barang, serta pengiriman informasi status operasional robot kembali ke sistem. Informasi tersebut digunakan untuk memantau kondisi dan posisi robot selama proses distribusi barang berlangsung. Setiap tahapan pada *flowchart* disusun untuk memberikan gambaran mengenai urutan proses yang berlangsung selama kegiatan pendistribusian barang. Gambaran keseluruhan alur sistem dapat dilihat pada Gambar 3.3.



Gambar 3. 3 Flowchart sistem

Gambar 3.3 menunjukkan diagram alir (*flowchart*) sistem secara keseluruhan, yang menggambarkan proses kerja sistem mulai dari penerimaan permintaan pengguna hingga proses pengantaran barang selesai dilakukan.

Proses diawali pada tahap Mulai, dilanjutkan dengan tahap *User* Memasukkan Parameter dan *Request*, yaitu operator memasukkan data permintaan pengantaran melalui antarmuka *website*, meliputi terminal *pick-up* dan jenis barang yang akan diantar. Sistem kemudian melakukan koneksi dan komunikasi,

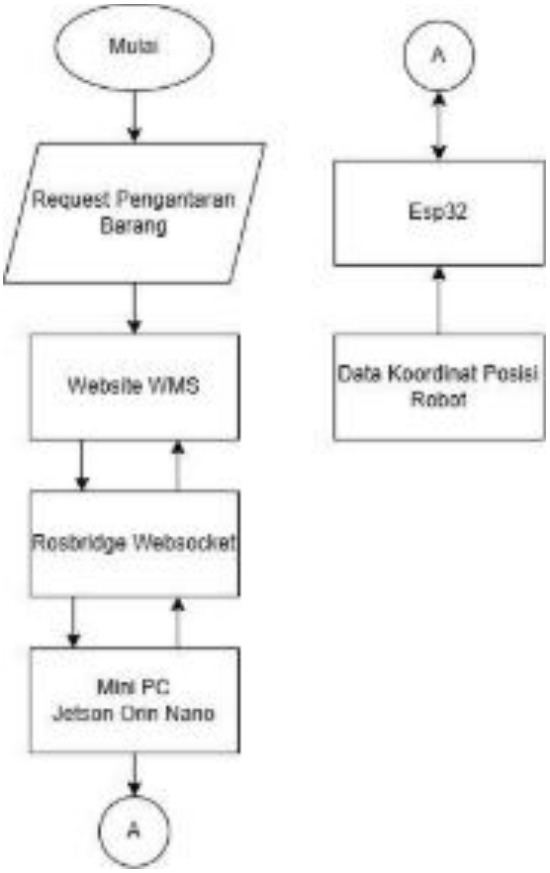
yaitu memastikan antarmuka *website* terhubung dengan robot melalui protokol *ROSBridge WebSocket* sebelum proses pengantaran dapat dijalankan.

Selanjutnya, sistem melakukan pemeriksaan pada tahap keputusan "Apakah ada Barang dengan Prioritas Pengantaran yang Sama?". Apabila jawabannya Tidak, yang berarti hanya terdapat satu permintaan aktif atau seluruh permintaan memiliki prioritas yang berbeda tanpa terjadi kondisi seri (*tie*), maka sistem langsung menetapkan robot untuk Mengantar Sesuai Urutan tanpa perlu melalui proses evaluasi lebih lanjut. Sebaliknya, apabila jawabannya Ya, yang berarti terdapat lebih dari satu permintaan dengan prioritas jenis barang yang sama, maka sistem akan melanjutkan proses ke tahap Penentuan Keputusan Pengantaran Robot dengan Decision Tree, yaitu proses penelusuran pohon keputusan C4.5 berdasarkan atribut jarak dan waktu tunggu untuk menentukan urutan prioritas pengantaran. Hasil dari proses ini kemudian juga menghasilkan keluaran berupa robot yang Mengantar Sesuai Urutan, sesuai dengan hasil keputusan yang telah ditentukan oleh algoritma *Decision Tree*.

Kedua alur tersebut, baik yang melalui proses *Decision Tree* maupun yang langsung ditetapkan tanpa evaluasi lebih lanjut, selanjutnya bertemu pada tahap Robot Bergerak dan Mengirimkan Data Odometri, yaitu robot mulai bernavigasi menuju terminal *pick-up* yang telah ditentukan, sekaligus mengirimkan data hasil pembacaan *rotary encoder* secara *real-time*. Data tersebut digunakan pada tahap berikutnya, yaitu Update Posisi Robot untuk Maps pada Web, yang memperbarui posisi robot pada peta tiga dimensi (*3D Maps*) di antarmuka *website* secara *real-time*, sehingga operator dapat memantau pergerakan robot selama proses pengantaran berlangsung.

Proses kemudian dilanjutkan melalui penghubung (*connector*) A menuju tahap Robot Drop Barang di Terminal Tujuan, yaitu robot melakukan penurunan barang pada terminal *drop-off* yang sesuai dengan terminal *pick-up* yang telah diproses sebelumnya. Setelah barang diturunkan, sistem melakukan Logging Data, yaitu mencatat data operasional pengantaran yang meliputi waktu pengantaran, status barang, dan terminal tujuan, ke dalam basis data. Proses pengantaran kemudian dinyatakan Selesai.

3.1.4 *Flowchart* Cara Kerja *Software*



Gambar 3. 4 *Flowchart* Sistem *Software*

Diagram alir pada Gambar 3.4 menjelaskan cara kerja perangkat lunak sistem secara berurutan, mulai dari inisiasi permintaan pengguna hingga robot memperoleh data posisi terkini. Proses diawali pada tahap Mulai, yang kemudian dilanjutkan dengan masukan (*input*) berupa Request Pengantaran Barang, yaitu permintaan pengiriman yang dikirimkan oleh operator melalui antarmuka *website*. Permintaan tersebut selanjutnya diproses oleh Website WMS, yang berperan sebagai *Warehouse Management System* berbasis web, mencakup pengambilan keputusan prioritas pengantaran menggunakan algoritma *Decision Tree* C4.5 berdasarkan permintaan yang diterima.

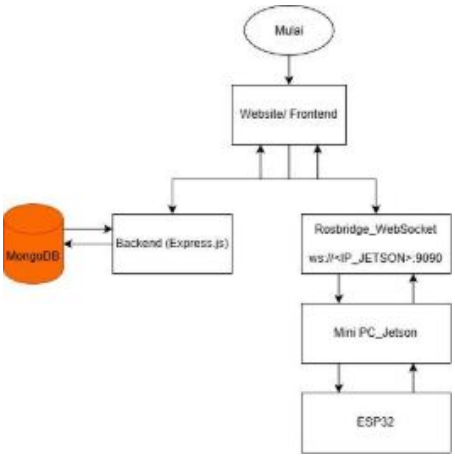
Setelah keputusan prioritas ditentukan, *Website WMS* berkomunikasi secara dua arah dengan ROSBridge WebSocket, yaitu mengirimkan perintah navigasi menuju terminal yang telah dipilih, serta menerima data status dan posisi robot secara *real-time* untuk ditampilkan kembali pada antarmuka *website*.

Selanjutnya, ROSBridge WebSocket meneruskan komunikasi tersebut secara dua arah menuju Mini PC Jetson Orin Nano, yang berperan sebagai pengendali utama navigasi robot berbasis *Robot Operating System* (ROS1).

Dari Jetson, alur proses dilanjutkan melalui penghubung (*connector*) A, yang menghubungkan proses pada Jetson dengan proses pada ESP32. Komunikasi antara Jetson dan ESP32 melalui *connector* A ini bersifat dua arah, yaitu Jetson mengirimkan perintah pergerakan dan kendali aktuator kepada ESP32, sedangkan ESP32 mengeksekusi perintah tersebut serta menghasilkan Data Koordinat Posisi Robot berdasarkan hasil pembacaan sensor *rotary encoder* pada roda. Data koordinat posisi tersebut kemudian diteruskan kembali melalui *connector* A menuju Jetson, untuk selanjutnya disebarkan kembali ke *website* melalui ROSBridge WebSocket, sehingga posisi robot dapat dipantau secara *real-time* oleh operator pada antarmuka *website*.

3.1.5 Alur Data Sistem

Alur data pada sistem ini yang menunjukkan bagaimana data mengalir antarkomponen mulai dari proses inisialisasi sistem hingga komunikasi dengan perangkat keras robot. Proses diawali pada tahap Mulai, yaitu ketika antarmuka *website* (*frontend*) dijalankan dan dimuat oleh pengguna melalui *browser*. Pada tahap ini, sistem melakukan inisialisasi tampilan antarmuka serta mempersiapkan dua jalur komunikasi utama yang akan digunakan selama sistem beroperasi, yaitu jalur menuju *backend server* dan jalur menuju robot melalui protokol *ROSBridge WebSocket*.



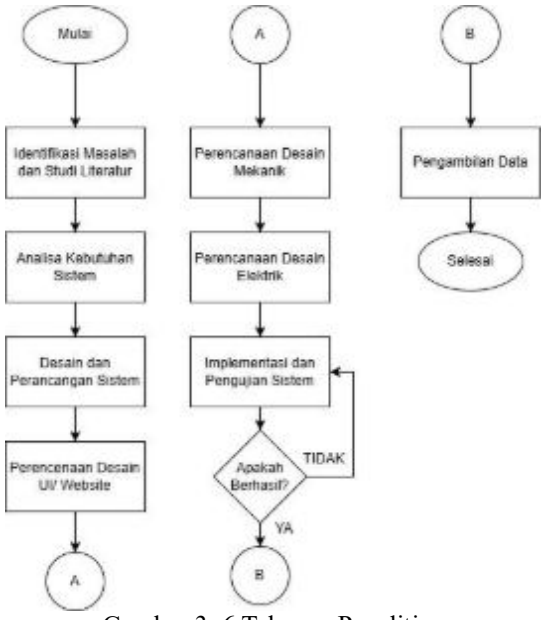
Gambar 3. 5 Alur Data Sistem

Pada Gambar 3.5, Jalur pertama menghubungkan Website/Frontend dengan Backend (Express.js). Komunikasi pada jalur ini bersifat dua arah: *frontend* mengirimkan data permintaan pengantaran, hasil keputusan algoritma *Decision Tree* C4.5, serta pembaruan status pengantaran menuju *backend*, sedangkan *backend* mengembalikan data riwayat pengantaran yang diminta oleh *frontend* untuk ditampilkan pada halaman *Database*. *Backend* yang dibangun menggunakan *framework* Express.js kemudian berkomunikasi secara dua arah dengan basis data MongoDB, yaitu menyimpan data baru yang dikirim dari *frontend* serta membaca data yang tersimpan untuk diteruskan kembali ke *frontend* saat diminta.

Jalur kedua menghubungkan Website/Frontend dengan robot melalui protokol ROSBridge WebSocket, dengan alamat koneksi berupa ws://<IP_JETSON>:9090. Melalui jalur ini, *frontend* mengirimkan perintah kendali robot, seperti perintah navigasi menuju terminal tujuan dan perintah kendali aktuator (relay, motor), menuju Mini PC Jetson. Sebaliknya, Jetson mengirimkan data status robot secara *real-time*, meliputi posisi robot, data sensor, serta status navigasi, kembali ke *frontend* melalui jalur *WebSocket* yang sama, sehingga komunikasi pada jalur ini juga bersifat dua arah.

Pada sisi Jetson, data yang diterima dari *ROSBridge WebSocket* diproses menggunakan *Robot Operating System* (ROS1), yang selanjutnya diteruskan menuju mikrokontroler ESP32 melalui komunikasi serial. Hubungan antara Jetson dan ESP32 juga bersifat dua arah: Jetson mengirimkan perintah pergerakan dan kendali aktuator kepada ESP32, sedangkan ESP32 mengirimkan data hasil pembacaan sensor, seperti data *encoder*, IMU, dan tegangan baterai, kembali kepada Jetson untuk diteruskan menuju *frontend* melalui *ROSBridge WebSocket*.
Dengan alur data tersebut, sistem mampu menjalankan dua fungsi utama secara bersamaan, yaitu pencatatan dan pengelolaan data pengantaran melalui jalur *Backend–MongoDB*, serta pengendalian dan pemantauan robot secara *real-time* melalui jalur *ROSBridge–Jetson–ESP32*, yang keduanya terintegrasi pada satu antarmuka *website* yang sama.

3.2 Tahapan Penelitian



Gambar 3. 6 Tahapan Penelitian

Tahap ini membahas tentang tahap penelitian, perencanaan dan desain *hardware* dan *software*, jadwal atau *timeline* penelitian, serta rancangan anggaran penelitian. Penjelasan dari alur penelitian ini dijelaskan secara berurutan.

Pada Gambar 3.6 di atas merupakan diagram alur penelitian pada Tugas Akhir ini yang dimulai dari identifikasi permasalahan yang diambil untuk penelitian ini. Identifikasi masalah adalah langkah awal yang sangat penting dalam proses penelitian, tahap ini bertujuan untuk menemukan dan merumuskan permasalahan yang akan menjadi fokusan penelitian Tugas Akhir ini. Selama melakukan identifikassi masalah dibutuhkan refrensi dan masukan untuk menunjang pemahaman dan pendalaman materi. Maka dari itu perlu dilakukanya studi literatur untuk memahami dan mengetahui seluruh kebutuhan sistem yang digunakan pada tugas akhir, kemudian akan di lakukan pembuatan dan perencanaan alat, *hardware, software,* dan sistem yang efektif agar dapat digunakan dalam Tugas Akhir.

Kemudian setelah selesai melakukan identifikasi masalah dan studi literatur maka langkah selanjutnya adalah melakukan analisa kebutuhan sistem. Analisa kebutuhan sistem diperlukan untuk memahami, mengidentifikasi, dan mendefinisikan kebutuhan yang diperlukan dalam sistem, guna mengembangkan

dan mengimplementasikan sistem yang dapat memenuhi tujuan, fungsi, dan efektifitas sistem yang diharapkan.

3.2.1 Identifikasi Masalah

Pada tahapan awal identifikasi masalah sistem control dan monitoring Autonomous Mobile Robot (AMR) yang terintegrasi dengan Warehouse Management System (WMS) masih dikembangkan menggunakan Python sebagai pusat pengendali utama. Pendekatan ini menyebabkan sistem bersifat monolitik, di mana proses pengolahan data sensor, kontrol aktuator, komunikasi robot, serta tampilan monitoring berada dalam satu alur program. Kondisi tersebut menimbulkan keterbatasan dalam pengembangan sistem, terutama dari sisi fleksibilitas, skalabilitas, dan kemudahan integrasi dengan sistem lain. Selain itu, sistem yang dikembangkan belum dirancang untuk menangani banyak permintaan (multi-request) secara bersamaan dari beberapa user atau perangkat, sehingga kurang optimal ketika digunakan pada lingkungan warehouse yang membutuhkan akses real-time oleh lebih dari satu operator. Ketergantungan sistem pada satu perangkat pengendali juga berpotensi menimbulkan gangguan operasional apabila terjadi kegagalan sistem. Oleh karena itu, diperlukan pengembangan sistem control dan monitoring AMR yang berbasis web dengan arsitektur client–server dan mampu menangani multi-request secara paralel, sehingga sistem WMS dan AMR dapat diakses oleh banyak user secara bersamaan, lebih stabil, serta lebih mudah dikembangkan sesuai kebutuhan operasional warehouse.

3.2.2 Studi Literatur

Tahap awal penelitian diawali dengan melakukan identifikasi permasalahan pada sistem robot AGV existing yang dijadikan sebagai acuan dalam pengembangan sistem Autonomous Mobile Robot (AMR). Berdasarkan kajian literatur dan pengamatan terhadap sistem tersebut, ditemukan bahwa robot AGV masih memiliki keterbatasan dalam aspek otomatisasi navigasi, khususnya karena pergerakan robot masih bergantung pada jalur yang telah ditentukan sebelumnya (guided path), sehingga tidak mampu beradaptasi secara dinamis terhadap perubahan lingkungan dan keberadaan obstacle secara real-time. Selain

itu, sistem AGV tersebut belum dilengkapi dengan mekanisme kendali yang mampu mempertahankan kestabilan kecepatan motor ketika robot membawa beban yang berbeda, sehingga performa pergerakan menjadi tidak konsisten. Dari sisi pemantauan, informasi kondisi robot seperti kecepatan, status baterai, dan performa pengantaran belum disajikan secara terintegrasi melalui antarmuka berbasis web, serta belum terdapat metode untuk memprediksi estimasi penggunaan baterai selama robot beroperasi. Keterbatasan-keterbatasan ini menjadi dasar pengembangan penelitian menuju sistem AMR yang lebih adaptif, stabil, dan informatif melalui penerapan kendali PID, integrasi sensor, serta sistem monitoring dan kontrol berbasis web.

3.2.3 Analisa Kebutuhan Sistem

Setelah melakukan studi literatur, Langkah selanjutnya ialah dengan melakukan Analisa kebutuhan sistem yang nantinya akan digunakan dalam sistem mobile robot ini. Tahap ini sangat penting sekali untuk memastikan setiap komponen benar benar dapat mendukung fungsi utama robot secara optimal. Dengan melakukan analisis ini, diharapkan dapat diperoleh pemahaman lebih jelas dan terperinci mengenai komponen dan aspek yang dibutuhkan dalam Pengembangan rsistem robot yang efektif dan efiien. Dengan begitu, proses pengembangan dapat dipersiapkan dan dilakukan dengan matang sesuai dengan kebutuhan yang ada. Kebutuhan sistem pada penelitian ini disajukan dalam table yang ada dibawah ini.

Tabel 3. 1 Analisis Kebutuhan Sistem

Kebtuhuan alat	Jumlah
ESP 32 Dev Module	1
Driver Motor Brushed	2
Mini PC NVDIA Jetson Nano Orin	1
Motor PG 45	2
Sensor Tegangan	1
Sensor Arus	1
Sensor Rotary Encorder	1
Sensor Limit Switch	2
Led Indikator	2
Buzzer	1
Sensor Loadcell	1
Linier Aktuator	1

3.2.4 Desain dan Perancangan Sistem

Desain dan perancangan sistem pada penelitian ini bertujuan untuk membangun sistem Warehouse Management System (WMS) yang terintegrasi dengan Autonomous Mobile Robot (AMR) untuk mendukung proses pengiriman barang pada gudang skala kecil dengan konsep cross docking. Sistem dirancang menggunakan arsitektur client–server yang terdiri atas antarmuka website, backend, database, sistem Robot Operating System (ROS), dan robot AMR sehingga proses monitoring, pengambilan keputusan, serta pengendalian robot dapat dilakukan secara terintegrasi.

Pada sisi perangkat keras, Autonomous Mobile Robot dikendalikan menggunakan ESP32 sebagai pengendali utama aktuator dan sensor, sedangkan Jetson Orin Nano digunakan sebagai komputer utama (mini PC) yang menjalankan Robot Operating System (ROS). ESP32 bertugas membaca data sensor, seperti rotary encoder, sensor IMU HWT901B, sensor tegangan, dan limit switch, kemudian mengirimkan data tersebut ke ROS untuk diproses lebih lanjut. Selain itu, ESP32 juga menerima perintah kendali dari ROS untuk menggerakkan motor penggerak maupun mekanisme lifter pada robot.

Komunikasi antara website dan robot dilakukan menggunakan protokol ROSBridge WebSocket yang berfungsi sebagai jembatan komunikasi antara aplikasi berbasis web dengan ekosistem ROS. Melalui protokol ini, website dapat mengirimkan perintah pengiriman kepada robot serta menerima data sensor, status robot, dan informasi navigasi secara real-time. Mekanisme komunikasi dua arah tersebut memungkinkan operator memonitor sekaligus mengendalikan robot melalui browser tanpa memerlukan aplikasi tambahan.

Pada sisi perangkat lunak, backend dikembangkan menggunakan Node.js dan Express.js sebagai penghubung antara website, database MongoDB, dan sistem ROS. Backend bertugas mengelola permintaan pengiriman dari operator, menjalankan algoritma Decision Tree C4.5 berbasis aturan untuk menentukan prioritas terminal penjemputan, menyimpan riwayat pengiriman ke database

MongoDB, serta meneruskan hasil keputusan kepada robot melalui ROSBridge WebSocket.

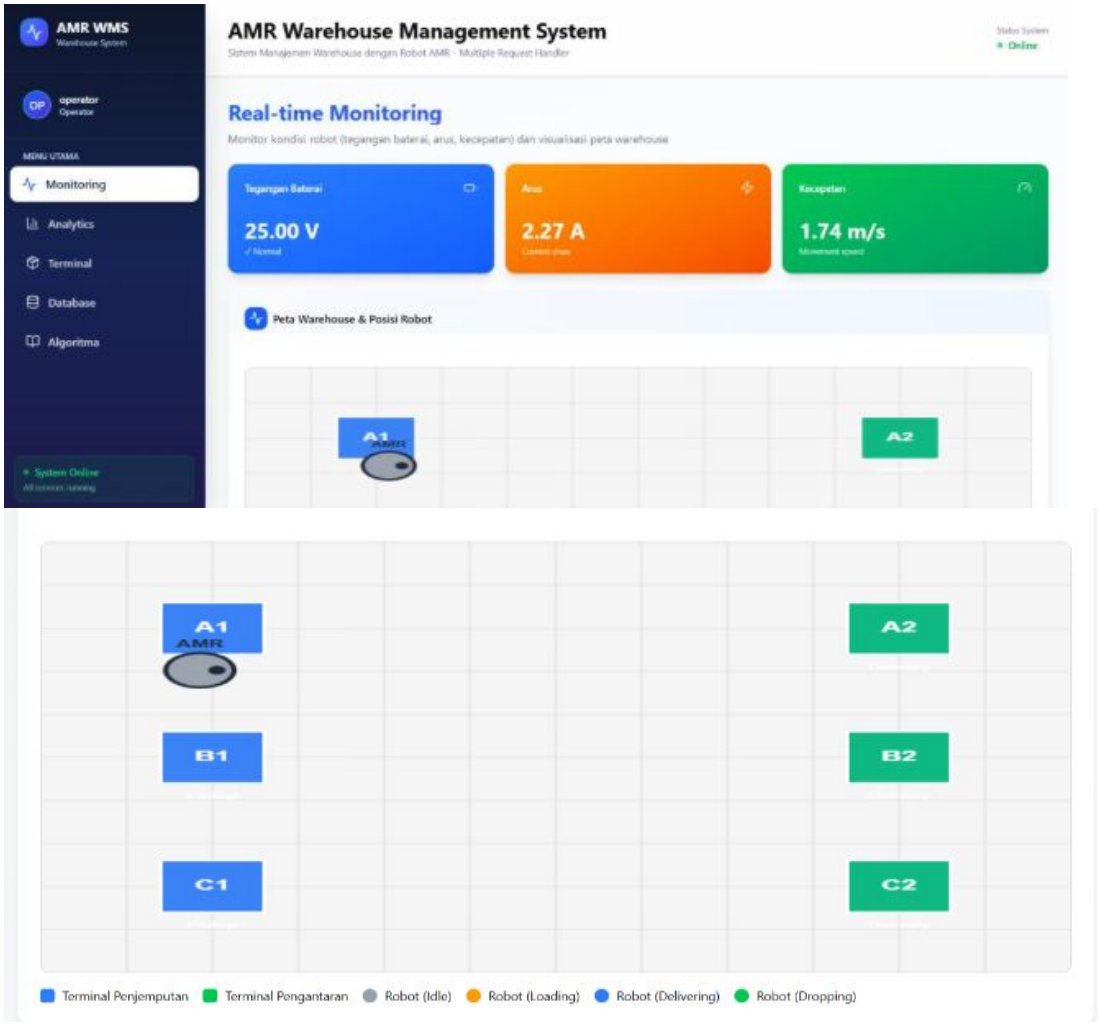
Antarmuka *Warehouse Management System* dikembangkan menggunakan React.js untuk menyediakan fasilitas monitoring dan pengendalian robot secara real-time. Website menampilkan informasi kondisi robot, status pengiriman, riwayat operasional, serta visualisasi posisi robot pada lingkungan tiga dimensi menggunakan Three.js. Selain itu, operator dapat melakukan input permintaan pengiriman dengan menentukan terminal *pick-up*, terminal *drop-off*, dan jenis barang yang akan dikirim. Seluruh data tersebut diproses oleh backend untuk menghasilkan keputusan prioritas menggunakan algoritma Decision Tree C4.5 sebelum dikirimkan kepada robot.

Dengan desain sistem tersebut, seluruh komponen perangkat keras dan perangkat lunak dapat bekerja secara terintegrasi sehingga proses penentuan prioritas pengiriman, monitoring kondisi robot, serta pengendalian Autonomous Mobile Robot dapat dilakukan secara otomatis melalui Warehouse Management System. Integrasi ini mendukung penerapan konsep cross docking pada gudang skala kecil, dimana barang yang diterima dapat segera diprioritaskan untuk dikirim menuju terminal tujuan tanpa melalui proses penyimpanan yang lama.

3.2.5 Perencanaan Desain UI/ IU Website

Untuk desain UI website yang akan digunakan nantinya adalah dengan menggunakan Bahasa pemrograman HTML, CSS, dan React JS. Bahasa pemrobgraman web ini sudah sering digunakan dalam berbagai kebutuhan membangun sebuah tampilan website yang interaktif dan user friendly. Tampilan web ini dibuat untuk mendukung kebutuhan dalam mengoperasikan dan menjalankan robot. Tampilan website ini dilengkapi dengan 4 menu pilihan yang dapat diakses melalui sidebar yang ada di samping kiri layar. Fitur itu antara lain adalah Dashboard sebagai tampilan untuk monitoring saat robot beroperasi. Lalu ada Database yang digunakan untuk melihat Jumlah barang yang telah berhasil di distribusikan, Lalu terakhir adalah Setting yang digunakan untuk melakukan

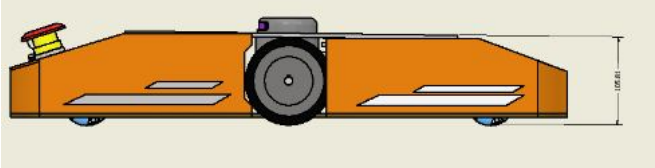
setting robot. Berikut ini adalah gambar tampilan website di tiap fiturnya



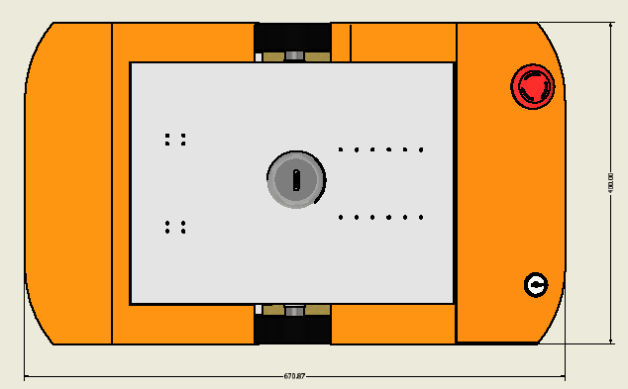
Gambar 3. 7 Desain UI/ IU WEBSITE

3.2.7 Perencanaan Desain Mekanik

Pada proses perencanaan ini dilakukan beberapa perombakan desain cover body robot serta perubahan posisi sensor LiDAR yang awalnya berada di depan robot, dipindah di bagian tengah robot. Perubahan posisi sensor LiDAR ini dilakukan untuk memperbaiki dan memaksimalkan persebaran laser LiDAR. Sehingga hasil pembacaan dari sensor LiDAR bisa maksimal. Selain itu untuk perubahan *cover body* robot ini dilakukan guna untuk memaksimalkan penempatan komponen penunjang robot agar bisa lebih baik dari sebelumnya. Robot AMR ini memiliki dimensi utama seperti di bawah ini:



Gambar 3. 8 Tampok Samping
(Sumber : Dokumentasi Pribadi, 2025)



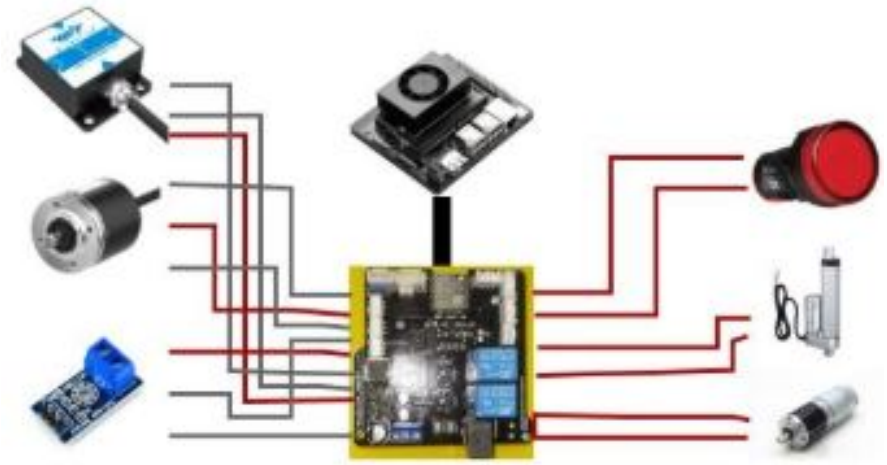
Gambar 3. 9 Tampak Atas
(Sumber : Dokumentasi Pribadi, 2025)

Gambar 3. 10 Tampak Depan
(Sumber : Dokumentasi Pribadi, 2025)

- Tinggi Robot: 10 cm
- Tinggi Maksimal Robot: 15 cm
- Lebar Robot: 40 cm
- Panjang Robot: 65 cm

Desain dari robot di buat se rendah mungkin agar robot dengan tujuan robot dapat mengangkat barang yang berada diatas palet, dimana palet ini memiliki ketinggian 13 cm.

3.2.8 Perencanaan Desain Elektrik



Gambar 3. 11 Perencanaan Desain Elektrik

Desain elektrik pada sistem Autonomous Mobile Robot (AMR) ini disusun dengan ESP32 Dev Kit sebagai pusat pengolahan data dan pengendali utama seluruh komponen input dan output. Pada sisi input, ESP32 menerima data dari beberapa sensor yang berfungsi untuk monitoring kondisi robot dan lingkungan operasional. Sensor rotary encoder digunakan untuk memperoleh informasi kecepatan dan jarak tempuh roda yang menjadi dasar perhitungan odometri robot. Sensor HWT (IMU) berfungsi untuk mendeteksi orientasi dan pergerakan robot, sedangkan limit switch digunakan sebagai pendeteksi batas mekanik atau posisi akhir gerakan aktuator. Sensor tegangan dan sensor arus berperan dalam memantau kondisi catu daya dan konsumsi daya sistem, sementara load cell digunakan untuk mendeteksi berat muatan yang dibawa robot sebagai bagian dari parameter operasional dan keamanan.

Seluruh data sensor tersebut diproses oleh ESP32 untuk menghasilkan keputusan kontrol maupun data monitoring. Pada sisi output, ESP32 mengendalikan driver motor yang terhubung ke motor DC ber-encoder (Motor PG) untuk mengatur pergerakan robot secara presisi berdasarkan perintah pengantaran. Selain itu, ESP32 juga mengendalikan linear actuator yang berfungsi sebagai mekanisme mekanik tambahan, seperti pengangkatan atau pelepasan barang. Perangkat output pendukung seperti buzzer dan LED indikator digunakan sebagai sistem notifikasi lokal untuk memberikan informasi status robot, misalnya kondisi normal, kesalahan, atau proses pengantaran sedang berlangsung.

Secara keseluruhan, desain elektrik ini menunjukkan arsitektur yang terstruktur dengan pemisahan yang jelas antara blok input, pemrosesan, dan output. ESP32 berperan sebagai pusat integrasi antara sistem sensor, aktuator, dan komunikasi data ke sistem WMS berbasis web. Desain ini mendukung kebutuhan monitoring real-time, kontrol dua arah, serta integrasi dengan sistem pengambilan keputusan berbasis decision tree, sehingga sesuai untuk diterapkan pada sistem AMR terintegrasi dalam lingkungan warehouse skala kecil hingga menengah.

3.2.9 Implementasi dan Pengujian Sistem

Implementasi sistem pada penelitian ini dilakukan dengan mengintegrasikan *Autonomous Mobile Robot* (AMR) dengan *Web-based Control Station* yang merepresentasikan fungsi *Warehouse Management System* (WMS) dalam proses *cross docking*. Robot menggunakan *single-board computer* Jetson yang menjalankan *Robot Operating System* (ROS1) sebagai pengendali utama navigasi, dilengkapi mikrokontroler ESP32 yang terhubung secara serial ke Jetson untuk menangani pembacaan sensor (IMU, *encoder*, tegangan baterai) serta penggerakan aktuator, meliputi motor penggerak dan mekanisme pengangkat barang (*lifter*). Komunikasi antara Jetson dan antarmuka web dijembatani menggunakan protokol *ROSBridge WebSocket*, sehingga perintah navigasi maupun data status robot dapat dipertukarkan secara *real-time* tanpa memerlukan aplikasi klien khusus.

Antarmuka web dikembangkan menggunakan HTML, CSS, dan *JavaScript* untuk menampilkan peta pergerakan robot secara visual dalam bentuk representasi tiga dimensi (*3D Maps*), status operasional robot, serta panel kendali manual dan otomatis. Algoritma *Decision Tree* C4.5 untuk menentukan prioritas terminal *pick-up* diimplementasikan pada sisi *frontend*, dengan struktur pohon keputusan bertingkat berdasarkan tiga atribut secara berurutan, yaitu jenis barang (*expired*, *express*, *regular*) sebagai atribut akar (*root*), jarak Manhattan sebagai atribut penentu tingkat kedua, dan waktu tunggu permintaan sebagai atribut penentu tingkat ketiga apabila jarak antarkandidat bernilai sama. Urutan atribut pada pohon keputusan ini ditentukan berdasarkan nilai *Gain Ratio* tertinggi dari masing-masing atribut, sesuai dengan prinsip pemilihan atribut pada algoritma

C4.5. Adapun navigasi robot menuju terminal yang telah ditentukan diselesaikan menggunakan algoritma pencarian jalur A* (*A-star*) yang dijalankan pada Jetson. Data operasional sistem, seperti riwayat pengantaran, hasil keputusan prioritas, koordinat pergerakan, serta status keberhasilan pengantaran, disimpan pada basis data MongoDB melalui *backend* Node.js dan *framework* Express, yang berfungsi sebagai penyedia layanan *REST API* antara antarmuka web dan basis data.

Pengujian sistem dilakukan pada lingkungan gudang berskala kecil dengan tiga terminal *pick-up* (A, B, C) dan tiga terminal *drop-off* (A−, B−, C−). Pengujian difokuskan pada tiga aspek utama, yaitu akurasi algoritma *Decision Tree* C4.5 dalam menentukan prioritas terminal pada berbagai kondisi pengujian, kemampuan sistem dalam menangani permintaan pengantaran secara bersamaan dari beberapa terminal (*multi-request*), serta keandalan integrasi data antara antarmuka web, basis data, dan robot melalui protokol *ROSBridge WebSocket*. Hasil pengujian menunjukkan bahwa sistem mampu menentukan urutan prioritas pengantaran secara konsisten berdasarkan struktur pohon keputusan yang telah dirancang, meneruskan perintah navigasi kepada robot secara *real-time*, serta menyimpan riwayat pengantaran pada basis data dengan data yang sesuai antara antarmuka web dan basis data MongoDB. Dengan demikian, sistem yang dikembangkan dinilai mampu memenuhi kebutuhan pengendalian dan pemantauan operasional AMR secara terintegrasi dalam simulasi proses *cross docking* berbasis web.

3.2.10 Pengambilan Data

Proses Pengambilan data pada penelitian ini dilakukan melalui beberapa metode yang saling melengkapi, dengan tujuan untuk memperoleh bukti kuantitatif dan kualitatif atas kinerja sistem secara menyeluruh, mulai dari proses pengambilan keputusan algoritma hingga integrasi antarkomponen. Metode pengambilan data yang digunakan meliputi observasi langsung terhadap perilaku robot, pencatatan keluaran sistem melalui konsol *browser*, serta penelusuran data yang tersimpan pada basis data.

Data hasil perhitungan algoritma *Decision Tree* C4.5 diperoleh melalui log yang ditampilkan pada konsol *Developer Tools* (*F12*) *browser*, yang mencatat seluruh tahapan pengambilan keputusan secara terperinci, mulai dari data kandidat, penyaringan prioritas, penelusuran pohon keputusan berdasarkan atribut jarak dan waktu tunggu, hingga penentuan terminal pemenang. Log tersebut kemudian didokumentasikan dalam bentuk cuplikan layar (*screenshot*) untuk setiap skenario pengujian yang dijalankan.

Data operasional sistem, seperti riwayat pengantaran, hasil keputusan prioritas, koordinat pergerakan robot, durasi pengerjaan, serta status keberhasilan pengantaran, diperoleh dari basis data MongoDB melalui koleksi *deliveries* yang tersimpan secara otomatis setiap kali siklus pengantaran dijalankan. Data ini digunakan untuk memverifikasi konsistensi antara hasil yang ditampilkan pada antarmuka web dan data yang benar-benar tersimpan pada basis data, sekaligus sebagai bahan analisis pada pengujian integrasi sistem.

Selain itu, data komunikasi antara antarmuka web dan robot diperoleh melalui pengamatan langsung pada terminal roslaunch di Jetson, yang menampilkan log perintah yang diterima robot dari *website* serta data sensor yang dikirimkan kembali oleh robot melalui protokol *ROSBridge WebSocket*. Data ini digunakan untuk membuktikan bahwa komunikasi antara *website* dan robot berjalan dua arah secara *real-time*.

Sebagai pelengkap data kuantitatif, dilakukan pula dokumentasi visual berupa foto dan video terhadap pergerakan fisik robot pada setiap pengujian, yang dibandingkan dengan representasi posisi robot pada peta tiga dimensi (*3D Maps*) di antarmuka web. Dokumentasi ini digunakan untuk memverifikasi kesesuaian antara posisi robot yang ditampilkan secara visual pada *website* dengan posisi robot yang sesungguhnya di lapangan. Seluruh data yang telah dikumpulkan melalui metode-metode tersebut kemudian dianalisis dan disajikan dalam bentuk tabel, grafik, maupun uraian deskriptif pada Bab IV, sesuai dengan jenis pengujian yang dilakukan, yaitu pengujian algoritma *Decision Tree* C4.5, pengujian operasional robot, dan pengujian integrasi sistem.

(Halaman ini sengaja dikosongkan)

BAB 4
HASIL DAN PEMBAHASAN

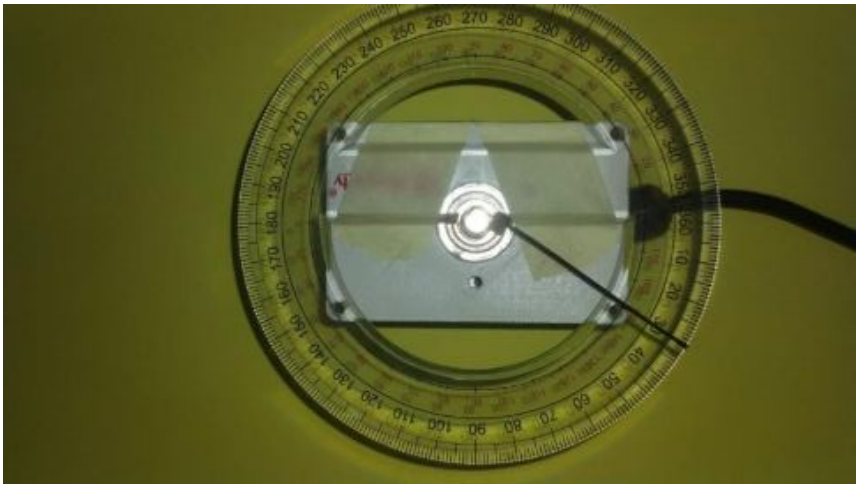
4.1 Pengujian Hardware

Dalam pengujian sensor ini, dilakukan dengan menggunakan alat pembanding guna mengevaluasi tingkat keakuratan sensor, untuk mengidentifikasi besarnya persentase kesalahan (*error*) pada masing-masing komponen sensor tersebut. Perhitungan dapat dilakukan sebagai berikut:

$$\%error = \frac{nilai\ pengujian - nilai\ aktual}{nilai\ aktual} x100\% \quad (4.1)$$

4.1.1 Pengujian Sensor *Rotary Encoder*

Pengujian sensor rotary encoder dilakukan untuk memverifikasi akurasi pembacaan sudut rotasi yang digunakan sebagai komponen odometri pada AMR. Sensor rotary encoder berfungsi mengonversi pergerakan rotasi roda menjadi sinyal digital yang kemudian diolah oleh mikrokontroler untuk memperkirakan posisi dan orientasi robot secara *real-time*. Pengujian dilakukan secara parsial dengan membandingkan nilai sudut yang terbaca oleh sensor terhadap nilai sudut aktual yang diukur menggunakan busur derajat fisik sebagai referensi. Perbandingan hasil pengukuran sensor *rotary encoder* dengan pengukuran busur derajat terdapat pada Tabel 4.1.



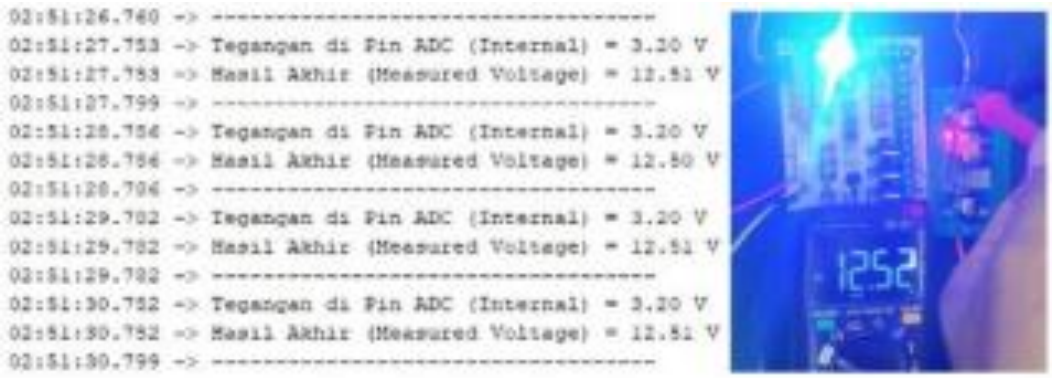
Gambar 4. 1 Pengujian Sensor *Rotary Encoder*
Sumber: (Dokumentasi Pribadi Penulis, 2026)

Tabel 4. 1 Pengujian Sensor *Rotary Encoder*

No	Sensor *Rotary Encoder* (°)	Busur (°)	Persentase *Error* (%)
1	10.2	10	2
2	20.5	20	2.5
3	29.7	30	1
4	40.3	40	0.75
5	49.8	50	0.4
6	60.4	60	0.67
7	70.1	70	0.14
8	80.3	80	0.38
9	89.6	90	0.44
10	99.8	100	0.2
Persentase *Error* rata-rata			0.85

4.1.2 Pengujian Sensor Tegangan

Pengujian sensor tegangan dilakukan untuk memverifikasi kemampuan sensor dalam mengukur nilai tegangan baterai yang digunakan sebagai sumber daya *Autonomous Mobile Robot* (AMR). Pengujian ini bertujuan untuk memastikan bahwa sensor mampu memberikan hasil pembacaan yang akurat sehingga data tegangan dapat digunakan sebagai informasi kondisi baterai pada sistem monitoring robot secara *real-time*. Selain itu, hasil pembacaan sensor juga dibandingkan dengan hasil pengukuran menggunakan avometer sebagai acuan untuk mengetahui tingkat akurasi sistem yang dikembangkan.



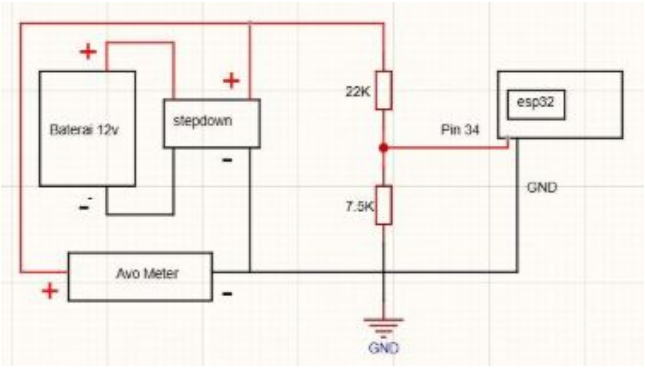
Gambar 4. 2 Pengujian Sensor Tegangan
Sumber: (Dokumentasi Penulis, 2026)

Tabel 4. 2 Pengujian Sensor Tegangan

No	Tegangan Pin Esp32 (V)	Measure Voltage (V)	Tegangan Avometer	Persentase *Error* (%)
1	3.20	12.51	12.52	0.08
2	3.15	12.31	12.30	0.08
3	3.10	12.12	12.10	0.17
4	3.05	11.91	11.92	0.08
5	3.00	11.71	11.70	0.09
6	2.95	11.50	11.49	0.09
7	2.90	11.31	11.30	0.09
8	2.85	11.11	11.12	0.09
9	2.80	10.91	10.90	0.09
10	2.75	10.71	10.70	0.09
Persentase *Error* rata-rata				0.095%

Berdasarkan hasil pengujian yang ditampilkan pada serial monitor, tegangan yang terbaca pada pin ADC internal berada pada kisaran 3.20 V, yang kemudian dikonversi menjadi nilai tegangan sebenarnya (measured voltage) sebesar sekitar 12.50 V hingga 12.51 V. Nilai tersebut menunjukkan hasil yang stabil dengan selisih yang sangat kecil antar pembacaan, sehingga dapat disimpulkan bahwa proses pembacaan dan konversi tegangan berjalan dengan baik.

Selain itu, hasil pengukuran juga divalidasi menggunakan alat ukur multimeter, yang menunjukkan nilai tegangan sebesar 12.52 V. Perbandingan antara hasil pembacaan sistem dan alat ukur menunjukkan selisih yang sangat kecil, yaitu sekitar 0.01–0.02 V, sehingga tingkat akurasi sensor dapat dikatakan sangat baik. Dengan demikian, sistem pembacaan tegangan berbasis ESP32 ini telah mampu memberikan data yang akurat dan stabil, sehingga layak digunakan untuk keperluan monitoring tegangan pada sistem Autonomous Mobile Robot (AMR).



Gambar 4. 3 Rangkaian Sensor Tegangan

Berikut adalah perhitungan secara teori untuk sensor tegangan:

$$AVout = Vin \times \frac{R2}{R2+R1} \quad (4.2)$$

$$AVout = 12 \times \frac{7.5k}{7.5k+22k} = 12{,}52 \times 0{,}2542 = 3{,}1831\,V$$

Berikut untuk perhitungan tegangan analog 3,20V to nilai digital ADC:

$$ADC\,raw = \frac{Vpin}{Vref} \times Resolusi \quad (4.3)$$

$$ADC\,raw = \frac{3.20}{3.3} \times 4095 = 3970{,}91$$

Berikut untuk perhitungan digital to tegangan:

$$\mathrm{V}pin = \frac{ADC\,raw}{Resolusi} \times \mathrm{V}rev \quad (4.4)$$

$$\mathrm{V}pin = \frac{3971}{4095} \times 3.3 = 3{,}2001\ \mathrm{V}$$

Untuk memvalidasi kesesuaian rangkaian pembagi tegangan yang telah dirancang, dilakukan perhitungan manual berdasarkan nilai resistor yang digunakan yaitu R1=22kΩ dan R=7,5kΩ. Dengan tegangan baterai sesungguhnya sebesar 12,52V (hasil pengukuran *avometer*), tegangan yang seharusnya terbaca pada Pin 34 ESP32 secara teoritis adalah 3,1831V. Nilai ini dibandingkan dengan tegangan yang benar-benar terukur pada *serial monitor*, yaitu 3,20V, menghasilkan selisih sebesar 0,53%. Nilai *error* yang kecil ini menunjukkan bahwa rangkaian pembagi tegangan yang dirancang menggunakan resistor 22kΩ dan 7,5kΩ bekerja sesuai dengan perhitungan teoritis, dengan penyimpangan yang

masih berada dalam batas toleransi komponen resistor yang umumnya sebesar ±5%. Hasil validasi ini membuktikan bahwa pemilihan nilai resistor 22kΩ dan 7,5kΩ, meskipun berbeda dari kombinasi ideal hasil perhitungan sebelumnya, tetap mampu menghasilkan tingkat akurasi pembacaan tegangan yang baik dan sesuai dengan kebutuhan sistem monitoring baterai secara *real-time*.

Berdasarkan hasil pengujian sensor tegangan sebanyak 10 kali percobaan, nilai tegangan yang dibaca oleh ESP32 memiliki selisih yang sangat kecil dibandingkan dengan hasil pengukuran menggunakan avometer. Persentase error tertinggi terjadi pada pengujian ke-3 sebesar 0,17%, sedangkan rata-rata persentase error seluruh pengujian sebesar 0,10%. Hasil tersebut menunjukkan bahwa sensor tegangan yang digunakan memiliki tingkat akurasi yang baik dan layak digunakan untuk pemantauan tegangan pada sistem AMR yang dikembangkan.

4.1.3 Pengujian Sensor Limit Switch

Pengujian limit switch dilakukan untuk memverifikasi fungsi sensor sebagai pendeteksi batas pergerakan (*end stop*) pada mekanisme *lifter Autonomous Mobile Robot* (AMR). Pengujian ini bertujuan untuk memastikan bahwa limit switch mampu memberikan perubahan sinyal secara tepat ketika aktuator mencapai posisi batas atas maupun batas bawah. Hasil pembacaan sensor kemudian digunakan sebagai masukan bagi sistem kontrol untuk menghentikan pergerakan linier aktuator sehingga mekanisme lifter dapat beroperasi dengan aman dan sesuai dengan batas pergerakan yang telah ditentukan.

Gambar 4. 4 Gambar Pengujian Sensor Limit Switch
(Documentasi Penulis, 2026)

Tabel 4. 3 Pengujian Sensor LimitSwitch

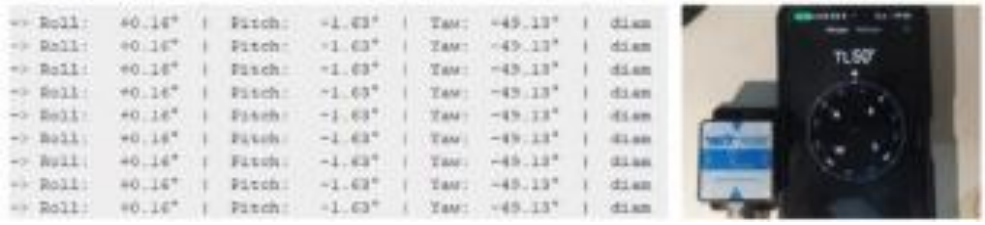
No	Ditekan	Tidak Ditekan
1	0	12.23
2	0	12.23
3	0	12.23
4	0	12.23
5	0	12.23
6	0	12.23
7	0	12.23
8	0	12.23
9	0	12.23
10	0	12.23

Berdasarkan hasil pengujian, saat limit switch dalam kondisi tidak ditekan, tegangan yang terbaca sebesar 0.006 V yang menunjukkan kondisi logika rendah (*LOW*). Sebaliknya, ketika limit switch ditekan, tegangan meningkat menjadi sekitar 12.23 V yang menandakan kondisi logika tinggi (*HIGH*). Perbedaan tegangan yang sangat jelas antara kondisi tidak aktif dan aktif ini menunjukkan bahwa limit switch bekerja dengan baik dan mampu memberikan sinyal yang tegas untuk sistem kontrol.

Limit switch ini digunakan sebagai komponen pengaman dan kontrol pada linier aktuator, dimana saat aktuator mencapai batas tertentu, limit switch akan aktif dan mengirimkan sinyal untuk menghentikan pergerakan lifter. Dengan demikian, sistem dapat mencegah pergerakan berlebih (*overtravel*) yang berpotensi merusak mekanisme. Hasil pengujian ini menunjukkan bahwa limit switch dapat diandalkan dalam mendukung sistem pengendalian posisi dan keselamatan pada lifter AMR.

4.1.4 Pengujian Sensor HWT901B

Pengujian sensor HWT901B-TTL dilakukan untuk memverifikasi kemampuan sensor dalam mengukur orientasi sudut *Autonomous Mobile Robot* (AMR) berdasarkan tiga sumbu, yaitu *roll*, *pitch*, dan *yaw*. Pengujian ini bertujuan untuk memastikan bahwa sensor mampu memberikan data orientasi secara akurat dan stabil, sehingga dapat dimanfaatkan sebagai informasi arah (*heading*) robot selama proses navigasi. Selain itu, hasil pembacaan sensor juga dibandingkan dengan aplikasi kompas pada *smartphone* sebagai validasi terhadap nilai orientasi yang dihasilkan.



Gambar 4. 5 Pengujian Sensor HWT901B
(Dokumentasi Penulis, 2026)

Berdasarkan hasil pengujian yang ditampilkan pada serial monitor, sensor secara konsisten menghasilkan nilai roll sebesar +0.16°, pitch sebesar -1.63°, dan yaw sebesar -49.13°. Nilai yang relatif stabil dan tidak mengalami fluktuasi signifikan ini menunjukkan bahwa sensor mampu bekerja dengan baik dalam kondisi diam (stationary). Selain itu, pengujian juga divalidasi menggunakan aplikasi kompas pada smartphone yang diletakkan berdekatan dengan sensor, dimana arah orientasi yang ditunjukkan memiliki kesesuaian terhadap nilai yaw yang dihasilkan oleh sensor.

Hasil ini menunjukkan bahwa sensor HWT901B-TTL telah terkalibrasi dengan cukup baik dan mampu memberikan data orientasi yang akurat. Dengan demikian, sensor ini layak digunakan sebagai sumber data untuk menentukan arah dan pergerakan robot dalam sistem Autonomous Mobile Robot (AMR), khususnya dalam proses navigasi dan pengendalian arah berbasis heading.

Tabel 4. 4 Pengujian Sensor HWT901B

No	Pembacaan Sensor (°)	Referensi Alat Ukur (°)	Error (%)
1.	9.94	10	0.60 %
2.	19.68	20	1.60 %
3.	29.08	30	3.07 %
4.	39.34	40	1.65 %
5.	49.13	50	1.74 %
6.	59.65	60	0.58 %
7.	70.39	70	0.56 %
8.	79.66	80	0.43 %
9.	89.90	90	0.11 %
10.	99.29	100	0.71 %
Rata-Rata Total Error			1.10%

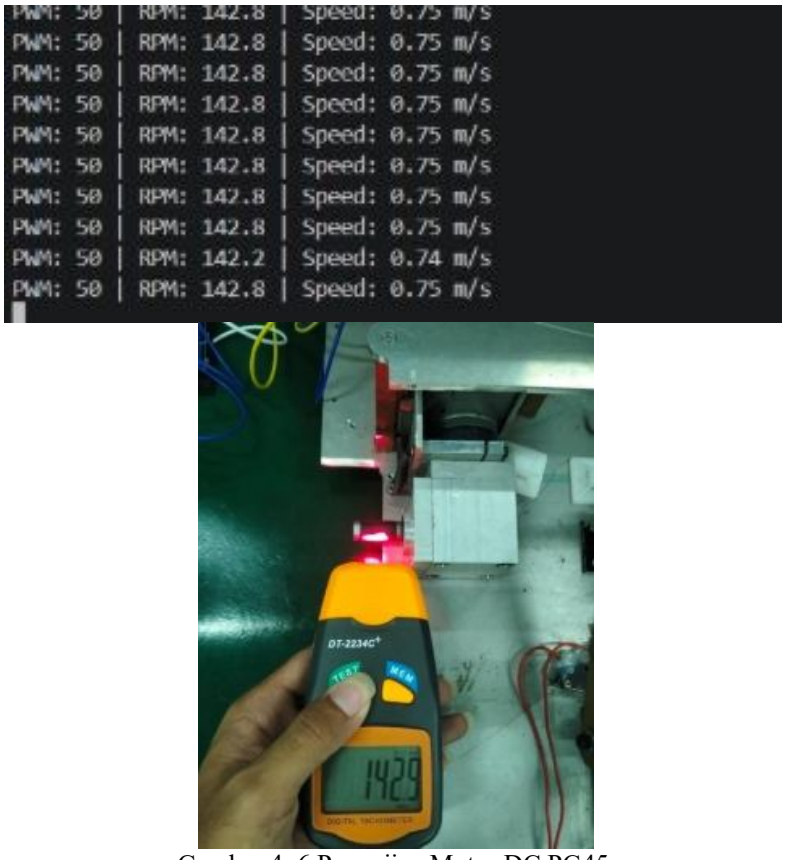
4.1.5 Pengujian Aktuator Motor DC PG45

Pengujian aktuator motor DC PG45 dilakukan untuk memverifikasi kesesuaian antara nilai kecepatan putar (RPM) yang terbaca oleh sistem berbasis rotary encoder dengan nilai RPM aktual yang diukur menggunakan alat tachometer digital DT-2234C+. Motor DC PG45 merupakan motor dengan gearbox yang digunakan sebagai penggerak utama roda AMR. Pengujian dilakukan secara parsial dengan memberikan sinyal PWM (*Pulse Width Modulation*) yang bervariasi sebagai masukan kecepatan motor, kemudian membandingkan pembacaan RPM dari sistem dengan RPM aktual dari tachometer.

Nilai kecepatan linier (m/s) dihitung berdasarkan RPM hasil pembacaan encoder dengan menggunakan persamaan berikut:

$$v = \frac{RPM\ x2\pi r}{60} \tag{4.5}$$

di mana r adalah jari-jari roda AMR.



Gambar 4. 6 Pengujian Motor DC PG45
(Dokumentasi Penulis, 2026)

Tabel 4. 5 Pengujian Motor DC PG45

No	PWM	RPM Sistem	RPM (Tacometer)	Persentase *Error* (%)
1	50	142.8	142.9	0.07
2	70	213.4	213.8	0.19
3	100	284.2	284.6	0.14
4	130	355.6	356.1	0.14
5	150	426.3	426.0	0.14
6	170	497.8	498.5	0.14
7	200	568.4	569.2	0.14
8	230	639.7	640.6	0.14
9	250	733.3	733.5	0.03
10	300	747.6	748.2	0.08
Persentase *Error* rata-rata				0.12

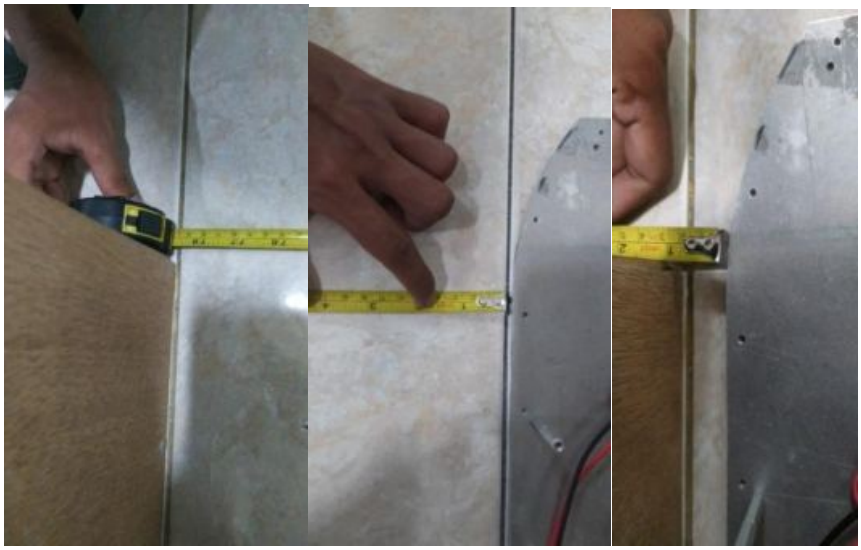
Berdasarkan hasil pengujian pada Tabel 4.5 aktuator motor DC PG45 menunjukkan kinerja yang sangat baik dengan rata-rata persentase error antara pembacaan RPM sistem dan RPM aktual tachometer sebesar 0,12%. Nilai *error* tertinggi terjadi pada PWM 50 sebesar 0,07%, sedangkan error terendah diperoleh pada PWM 250 sebesar 0,03%. Hasil ini menunjukkan bahwa pembacaan RPM oleh sistem encoder konsisten dan stabil di seluruh rentang PWM yang diuji.

Secara keseluruhan, hubungan antara nilai PWM dan RPM yang dihasilkan bersifat linier positif, di mana setiap kenaikan nilai PWM sebesar 25 menghasilkan peningkatan RPM yang proporsional rata-rata sekitar 71 RPM**.** Kecepatan putar motor berkisar antara 142,9 RPM pada PWM 50 hingga 748,2 RPM pada PWM 255, sedangkan kecepatan linier robot berkisar antara 0,75 m/s hingga 3,93 m/s. Rentang operasional ini menunjukkan bahwa motor DC PG45 memiliki kemampuan yang memadai untuk mendukung berbagai skenario pergerakan AMR, mulai dari manuver lambat di area sempit hingga perpindahan cepat antar titik pengantaran dalam gudang. Penyimpangan kecil yang tetap terjadi pada beberapa titik pengujian dapat disebabkan oleh getaran mekanis pada gearbox, keterbatasan frekuensi sampling encoder, serta fluktuasi tegangan sumber daya pada saat motor berputar pada kecepatan tinggi. Meskipun demikian, dengan rata-rata error di bawah 0,15% pada seluruh rentang PWM yang diuji, motor DC PG45 beserta sistem pembacaan encodernya dinilai layak dan akurat untuk digunakan sebagai aktuator penggerak utama AMR dalam penelitian ini

4.1.6 Pengujian *Single Wheel Odometry*

Pengujian *Single Wheel Odometry* dilakukan untuk memverifikasi kemampuan sistem dalam memperkirakan posisi dan jarak tempuh *Autonomous Mobile Robot* (AMR**)** berdasarkan data yang diperoleh dari rotary encoder. Pengujian ini bertujuan untuk mengetahui tingkat akurasi estimasi jarak yang dihasilkan sebelum data posisi digunakan sebagai informasi monitoring pada *Warehouse Management System* (WMS**)**. Hasil pengujian diharapkan dapat menunjukkan bahwa data posisi yang diterima sistem telah sesuai dengan pergerakan aktual robot sehingga dapat digunakan sebagai acuan dalam proses pemantauan operasional robot secara *real-time*.

Pengujian *single wheel odometry* dilakukan untuk memverifikasi akurasi sistem pengukuran jarak tempuh robot berdasarkan pembacaan *rotary encoder* pada roda. Pengujian dilakukan dengan memberikan perintah pergerakan pada rentang jarak 1 hingga 10 meter melalui program, kemudian jarak tempuh aktual robot diukur secara langsung menggunakan alat ukur fisik untuk dibandingkan dengan jarak yang telah ditetapkan pada program (*jarak setting*). Perbandingan ini bertujuan untuk mengetahui tingkat penyimpangan (*error*) antara jarak yang diperintahkan dengan jarak yang benar-benar ditempuh oleh robot.



Gambar 4. 7 Pengujian *Single Wheel Odometry*
(Dokumentasi Penulis, 2026)

Tabel 4. 6 Pengujian *Single Wheel Odometry*

No	Jarak Setting Program (meter)	Jarak Real (meter)	Pulse	Persentase *Error* (%)
1	1	0.82	2.463	18
2	2	1.80	3.820	10
3	3	2.95	6.260	1.67
4	4	3.78	8.021	5.50
5	5	4.21	8.934	15.80
6	6	5.14	10.907	14.33
7	7	6.23	13.220	11.00
8	8	7.36	15.618	8.00
9	9	8.93	18.950	0.78
10	10	9.82	20.839	1.80
Persentase *Error* rata-rata				8.69%

Pengujian metode *single wheel odometry* dilakukan untuk mengevaluasi tingkat akurasi sistem dalam mengestimasi jarak tempuh robot. Proses pengujian dilakukan dengan cara menggerakkan robot pada lintasan lurus dengan jarak aktual yang telah ditentukan, yaitu sejauh 1 meter.

Perhitungan untuk *single wheel odometry* :

1. Dikarenakan diameter roda adalah 6 cm :

   $K = \pi \times d = 3{,}14159 \times 6\ cm = 18{,}8496\ cm$ = 0.188496 m

2. Menghitung per 1 pulse encoder :

   $Jarak\ per\ Pulse = \frac{K}{PPR} = \frac{18.8496\ cm}{400} = 0{,}047124\ cm/pulse$

3. Menghitung Jumlah Pulsa yang Dibutuhkan untuk Menempuh 1 Meter

   $Pulse\ per\ meter = \frac{100cm}{jarak\ per\ pulse} = \frac{100cm}{0{,}047124} = 2.122{,}07\ pulsa/meter$

Selama pengujian, sistem mencatat hasil estimasi jarak berdasarkan data pembacaan rotary encoder yang diolah menggunakan metode odometri. Hasil pengukuran kemudian dibandingkan dengan jarak aktual untuk mengetahui besar kesalahan (*error*) yang terjadi.

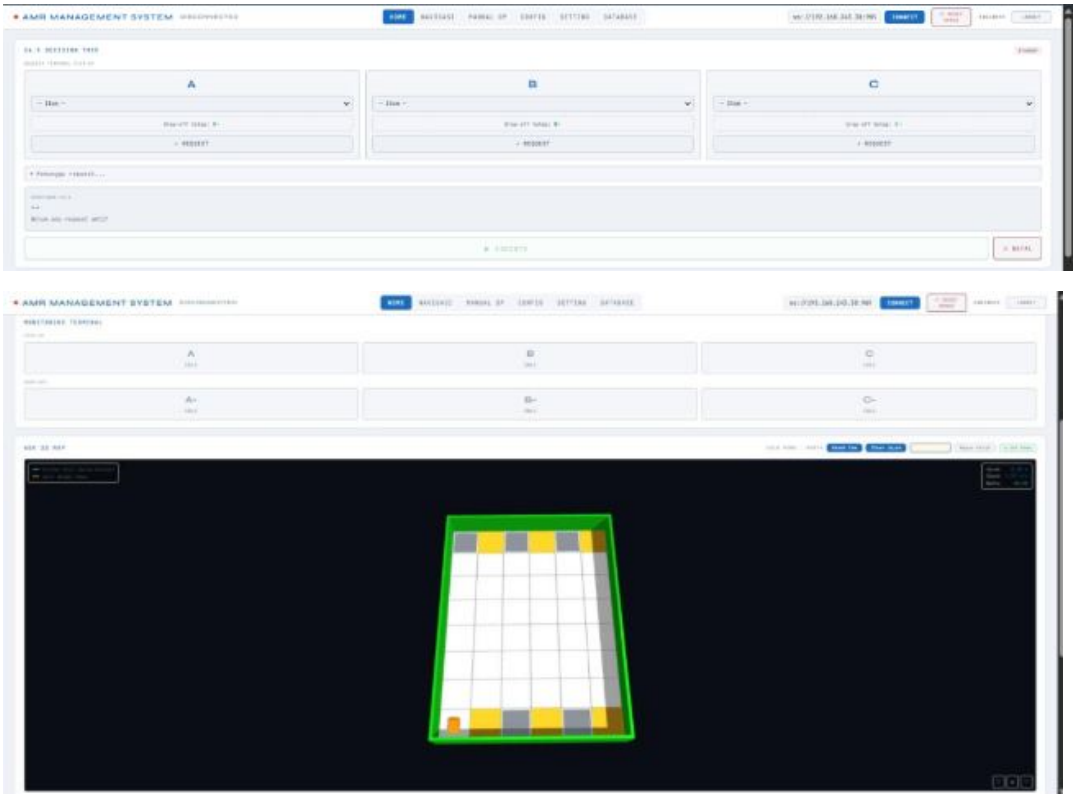
Hasil pengujian pada Tabel 4.6 menunjukkan bahwa persentase *error* terkecil terjadi pada percobaan ke-9 dengan jarak *setting* 9 meter, yaitu sebesar 0,78%, sedangkan *error* terbesar terjadi pada percobaan pertama dengan jarak *setting* 1 meter, yaitu sebesar 18,00%. Tingginya *error* pada jarak *setting* yang kecil diduga disebabkan oleh fase akselerasi dan deselerasi robot yang belum sepenuhnya stabil pada jarak tempuh pendek, sehingga proporsi kesalahan menjadi lebih besar dibandingkan pada jarak tempuh yang lebih panjang. Fluktuasi *error* pada beberapa percobaan lainnya turut dipengaruhi oleh faktor non-sistematis, seperti variasi gesekan roda terhadap permukaan lantai dan kemungkinan *slip* roda yang bersifat acak. Secara keseluruhan, persentase *error* rata-rata dari 10 kali percobaan tercatat sebesar 8,69%, yang menunjukkan bahwa sistem odometri berbasis *rotary encoder* pada robot memiliki tingkat akurasi yang cukup baik dalam mengukur jarak tempuh.

4.2 Pengujian *Software*

Pengujian tampilan dashboard *website* dilakukan untuk memastikan bahwa sistem monitoring dan pengendalian *Autonomous Mobile Robot* (AMR) berbasis web dapat berjalan dengan baik. Dashboard ini terintegrasi dengan backend Node.js serta ESP32 melalui protokol komunikasi Rosbridge. Berdasarkan hasil pengujian, terlihat bahwa sistem telah berhasil menampilkan data secara real-time dari ESP32, yang ditandai dengan indikator status “ESP32 Terhubung”. Hal ini menunjukkan bahwa proses komunikasi data antara perangkat dan server berjalan dengan baik dan stabil.

4.2.1 Pengujian Tampilan Menu Dashboard

Pengujian menu dashboard dilakukan untuk memverifikasi bahwa halaman utama Warehouse Management System (WMS) mampu mengintegrasikan seluruh informasi operasional robot dalam satu antarmuka. Dashboard dirancang sebagai pusat kendali (*control center*) yang memungkinkan operator melakukan pemantauan kondisi sistem, memberikan perintah kepada robot, serta mengamati proses pengiriman barang secara real-time.



Gambar 4. 8 Tampilan Menu *Dashboard*
(Dokumentasi Pribadi, 2026)

Berdasarkan hasil pengujian, seluruh komponen pada halaman dashboard berhasil ditampilkan dengan baik. Pada bagian atas halaman tersedia menu navigasi yang menghubungkan operator ke halaman *Home*, Navigasi, *Manual Operation*, *Config*, *Setting*, dan *Database*. Selain itu, dashboard juga menampilkan informasi status koneksi robot melalui indikator koneksi WebSocket beserta alamat IP robot yang digunakan untuk proses komunikasi.

Pada bagian Decision Tree C4.5, operator dapat memilih jenis barang pada setiap terminal *pick-up* (A, B, dan C), menentukan terminal *drop-off*, serta mengirimkan *request* pengangkutan barang. Setelah seluruh data dimasukkan, sistem akan menampilkan hasil keputusan prioritas pengambilan berdasarkan algoritma Decision Tree C4.5, kemudian operator dapat mengeksekusi hasil keputusan tersebut melalui tombol *Execute*.

Selanjutnya, dashboard juga menampilkan bagian Monitoring Terminal yang berfungsi untuk memperlihatkan kondisi setiap terminal *pick-up* maupun

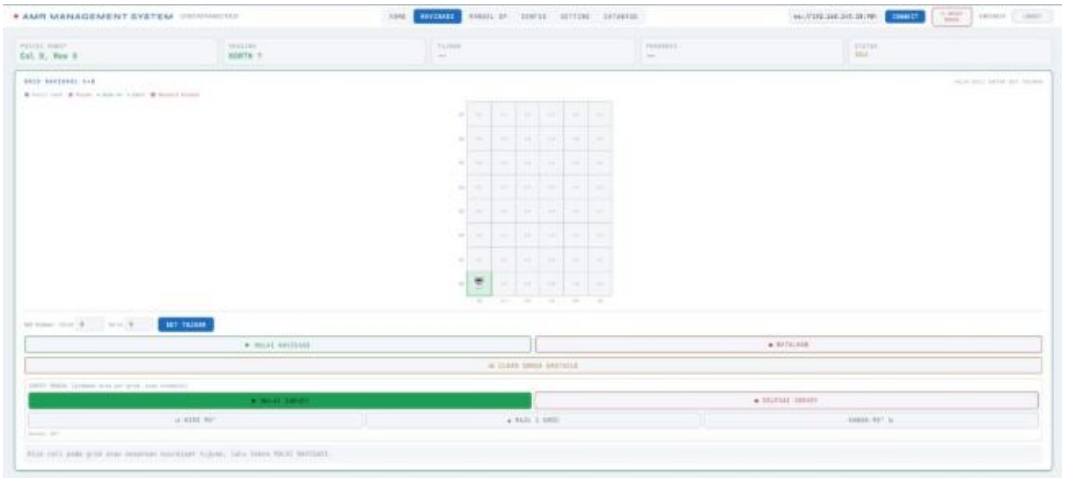
drop-off. Informasi status terminal akan diperbarui secara otomatis sesuai dengan proses pengiriman yang sedang dijalankan oleh robot sehingga operator dapat mengetahui kondisi masing-masing terminal secara langsung.

Pada bagian bawah halaman terdapat AGV 3D Map yang digunakan untuk memvisualisasikan posisi robot di dalam area gudang. Visualisasi ini dibangun menggunakan library Three.js dan memperlihatkan posisi robot, tata letak terminal, serta lintasan pergerakan robot berdasarkan data posisi yang diterima dari sistem navigasi. Selain itu, tersedia beberapa fungsi pendukung seperti *Reset Camera*, *Clear Jejak*, *Set Home*, serta informasi jarak tempuh, kecepatan robot, dan waktu operasi sebagai pendukung proses monitoring.

Hasil pengujian menunjukkan bahwa seluruh komponen pada menu Dashboard dapat ditampilkan dengan baik dan berfungsi sesuai dengan rancangan. Informasi yang ditampilkan pada dashboard selalu diperbarui mengikuti kondisi sistem sehingga operator dapat melakukan proses monitoring dan pengendalian robot melalui satu halaman tanpa perlu berpindah ke menu lain. Dengan demikian, menu Dashboard telah berhasil menjalankan fungsinya sebagai pusat kendali operasional pada sistem *Warehouse Management System* (WMS) yang dikembangkan.

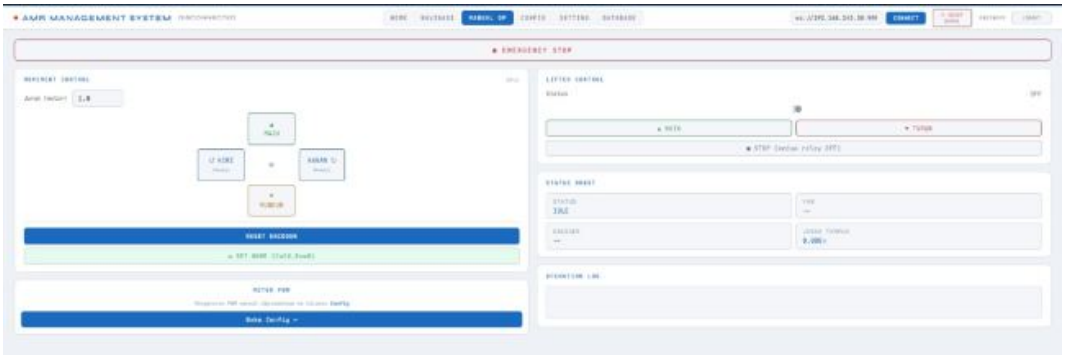
4.2.2 Pengujian Menu Operasional

Pengujian menu operasional dilakukan untuk memverifikasi bahwa setiap menu yang digunakan dalam pengoperasian *Autonomous Mobile Robot* (AMR) dapat berfungsi sesuai dengan rancangan sistem. Pada penelitian ini, menu operasional terdiri atas Navigasi, *Manual Operation*, dan *Config*, yang masing-masing memiliki fungsi berbeda dalam mendukung proses pengendalian robot. Pengujian dilakukan dengan mengoperasikan seluruh fitur pada masing-masing menu dan mengamati respons sistem terhadap setiap perintah yang diberikan. Hasil pengujian digunakan untuk mengetahui kesesuaian fungsi setiap menu dengan kebutuhan operasional AMR. Rincian hasil pengujian pada masing-masing menu operasional ditampilkan pada Gambar 4.9 hingga Gamba 4.11**.**



Gambar 4. 9 Menu Navigasi
(Dokumentasi Pribadi, 2026)

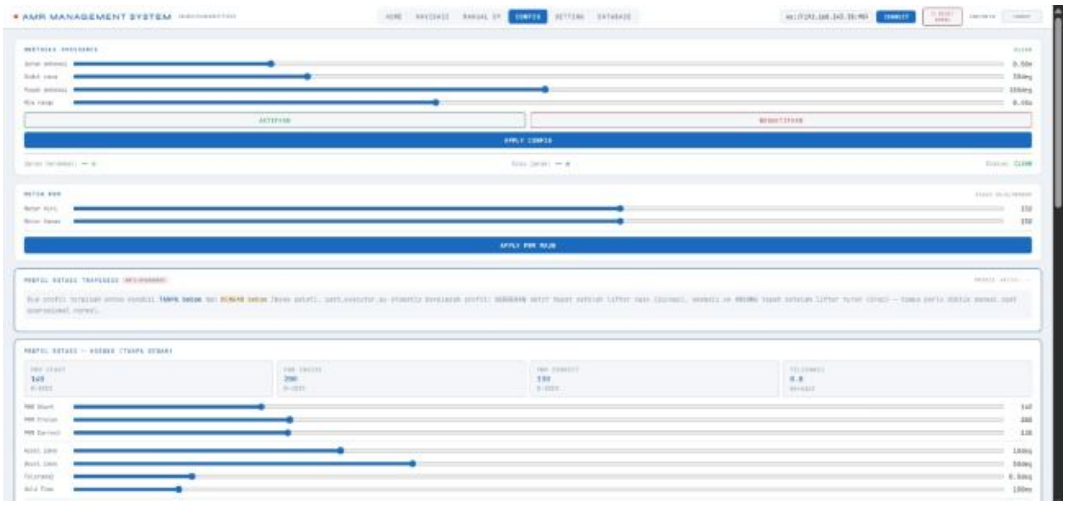
Menu Navigasi digunakan untuk melakukan pengujian perpindahan robot menuju titik tujuan berdasarkan koordinat grid. Pada halaman ini operator dapat menentukan posisi tujuan secara langsung melalui peta grid maupun dengan memasukkan koordinat kolom dan baris secara manual. Selain itu, halaman navigasi menampilkan informasi posisi robot, arah hadap (*heading*), status navigasi, serta progres perjalanan robot menuju titik tujuan. Selama pengujian, robot berhasil menerima koordinat tujuan dan menjalankan proses navigasi sesuai lintasan yang dihasilkan oleh algoritma A*.



Gambar 4. 10 Menu Manual Operation
(Dokumentasi Pribadi, 2026)

Menu *Manual Operation* digunakan untuk mengendalikan robot secara manual selama proses pengujian maupun kalibrasi sistem. Operator dapat menggerakkan robot ke arah maju, mundur, belok kiri, dan belok kanan tanpa melalui proses navigasi otomatis. Selain itu, halaman ini juga menyediakan kontrol *lifter* untuk menaikkan dan menurunkan mekanisme pengangkatan barang, tombol *Emergency*

Stop, serta informasi kondisi robot seperti status, nilai encoder, sudut yaw, dan jarak tempuh. Berdasarkan hasil pengujian, seluruh tombol kontrol mampu memberikan respons sesuai dengan fungsi yang dirancang sehingga robot dapat dikendalikan secara langsung oleh operator.



Gambar 4. 11 Menu Config
(Dokumentasi Pribadi, 2026)

Menu *Config* digunakan untuk mengatur parameter yang berkaitan dengan performa pergerakan robot. Parameter yang dapat dikonfigurasi meliputi pengaturan *Obstacle Avoidance*, nilai PWM motor, serta profil pergerakan robot pada kondisi membawa maupun tanpa membawa beban. Perubahan parameter yang dilakukan melalui antarmuka website berhasil diterapkan pada sistem robot setelah tombol *Apply Confi***g** dipilih. Hal ini menunjukkan bahwa mekanisme konfigurasi antara antarmuka website dan sistem robot telah berjalan dengan baik.

Berdasarkan hasil pengujian, seluruh menu operasional dapat berfungsi sesuai dengan tujuan perancangannya. Menu Navigasi mampu mengirimkan koordinat tujuan kepada robot, *Manual Operation* memungkinkan operator mengendalikan robot dan aktuator secara langsung, sedangkan menu *Config* berhasil melakukan konfigurasi parameter operasional robot. Dengan demikian, ketiga menu tersebut telah mendukung proses pengoperasian, pengujian, dan kalibrasi *Autonomous Mobile Robot* (AMR) secara terpadu melalui antarmuka *Warehouse Management System* (WMS).

4.2.3 Pengujian Database MongoDB

Pengujian database MongoDB dilakukan untuk memastikan bahwa seluruh data hasil operasional robot tersimpan secara otomatis setelah proses pengiriman dijalankan. Pada penelitian ini, MongoDB digunakan sebagai media penyimpanan data pengiriman (*collection deliveries*) yang mencatat informasi permintaan pengiriman, hasil keputusan Decision Tree C4.5, koordinat terminal, status pengiriman, serta waktu pelaksanaan pengiriman. Pengujian dilakukan dengan menjalankan proses pengiriman melalui website, kemudian memverifikasi data yang tersimpan pada database MongoDB. Proses verifikasi dilakukan dengan membandingkan data yang ditampilkan pada sistem dengan data yang tersimpan pada dokumen MongoDB, sehingga dapat diketahui kesesuaian antara data operasional yang dihasilkan sistem dengan data yang berhasil direkam ke dalam database.

Data yang diverifikasi meliputi informasi terminal *pick-up* dan *drop-off*, jenis barang, daftar *request* yang diterima, hasil penentuan prioritas pengiriman, serta status akhir proses pengiriman. Selain itu, dilakukan pemeriksaan terhadap data pendukung berupa koordinat terminal, durasi pengiriman, waktu mulai, waktu selesai, dan waktu pembaruan data. Pengujian ini dilakukan untuk memastikan bahwa proses penyimpanan tidak hanya mencatat permintaan awal, tetapi juga merekam hasil dan kondisi akhir pengiriman setelah robot menyelesaikan proses operasionalnya.

Hasil verifikasi menunjukkan bahwa data yang dihasilkan selama proses pengiriman dapat tersimpan pada *collection deliveries* setelah proses operasional dijalankan. Data yang tersimpan memuat informasi yang berkaitan dengan proses pengiriman, mulai dari permintaan yang diterima hingga kondisi akhir pengiriman. Penyimpanan data tersebut memungkinkan informasi operasional yang telah dilakukan untuk ditinjau kembali dan digunakan sebagai catatan proses pengiriman. Pemeriksaan dilakukan dengan mencocokkan informasi yang terdapat pada sistem dengan data yang tercatat pada database untuk memastikan bahwa data yang ditampilkan memiliki kesesuaian. Dokumentasi hasil penyimpanan dan pemeriksaan data pada database MongoDB ditampilkan pada Gambar 4.12

Gambar 4. 12 Pengujian MongoDB
(Dokumentasi Pribadi, 2026)

Gambar 4.12 menunjukkan salah satu dokumen yang berhasil disimpan pada collection deliveries setelah proses pengiriman selesai. Setiap dokumen memiliki _id sebagai identitas unik data, kemudian informasi terminalPickup bernilai A dan terminalDropoff bernilai A-, yang menunjukkan bahwa robot melakukan pengambilan barang di Terminal A dan mengantarkannya menuju

Terminal A-. Nilai itemType sebesar express menunjukkan bahwa jenis barang yang diproses merupakan barang dengan prioritas express.

Selain menyimpan data pengiriman utama, dokumen juga menyimpan seluruh request yang diterima sistem pada atribut requests. Pada pengujian ini terdapat tiga permintaan aktif, yaitu Terminal A menuju A-, Terminal B menuju B-, dan Terminal C menuju C-. Seluruh data request tersebut digunakan sebagai masukan bagi algoritma Decision Tree C4.5 untuk menentukan terminal yang memiliki prioritas penjemputan tertinggi.

Hasil proses pengambilan keputusan disimpan pada atribut priorityDelivery. Berdasarkan hasil pengujian, sistem memilih Terminal A sebagai prioritas penjemputan dengan nilai winner = A. Alasan pemilihan juga tercatat secara otomatis pada atribut reason, yaitu karena ketiga terminal memiliki prioritas barang yang sama (P2), sehingga sistem melanjutkan proses klasifikasi menggunakan parameter jarak dan waktu tunggu. Selain itu, atribut scores menyimpan nilai evaluasi masing-masing terminal yang meliputi tingkat prioritas barang, jarak terhadap robot, waktu tunggu permintaan, serta nilai skor akhir hasil perhitungan algoritma Decision Tree C4.5. Penyimpanan informasi ini memudahkan proses analisis dan penelusuran kembali terhadap keputusan yang dihasilkan sistem.

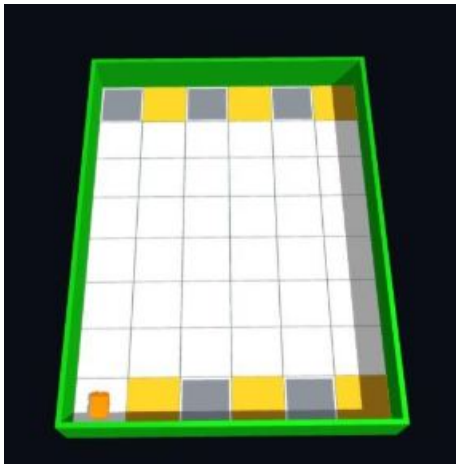
Dokumen juga mencatat informasi operasional robot selama proses pengiriman. Nilai pickupCol = 1 dan pickupRow = 7 menunjukkan koordinat terminal pengambilan, sedangkan dropoffCol = 1 dan dropoffRow = 0 menunjukkan koordinat tujuan pengantaran. Waktu mulai proses pengiriman tersimpan pada atribut waktuMulai, sedangkan waktu selesai tercatat pada atribut waktuSelesai. Berdasarkan kedua data tersebut, sistem menghitung durasi pengiriman selama 84 detik yang disimpan pada atribut durasiDetik. Nilai status = success dan lastPhase = done menunjukkan bahwa proses penjemputan dan pengantaran berhasil diselesaikan tanpa mengalami kegagalan.

Berdasarkan hasil pengujian tersebut, database MongoDB berhasil menyimpan seluruh informasi operasional secara lengkap, mulai dari data permintaan, hasil klasifikasi Decision Tree C4.5, koordinat pengambilan dan pengantaran, hingga status serta durasi proses pengiriman. Hasil ini menunjukkan bahwa mekanisme penyimpanan data pada sistem telah berjalan dengan baik sehingga seluruh aktivitas robot dapat terdokumentasi secara otomatis dan dapat digunakan sebagai riwayat operasional maupun bahan evaluasi sistem.

4.2.4 Pengujian Tampilan 3D Maps

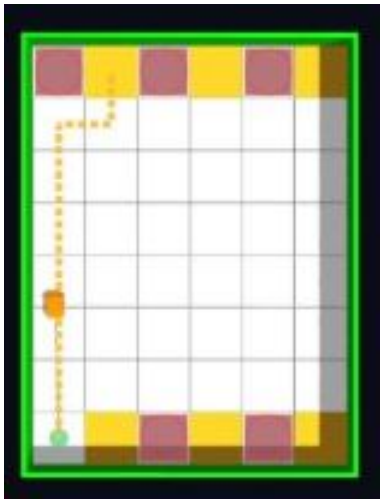
Pengujian tampilan 3D Maps dilakukan untuk memverifikasi bahwa visualisasi lingkungan kerja Autonomous Mobile Robot (AMR) pada website mampu menampilkan posisi robot secara *real-time* berdasarkan data koordinat yang diterima dari sistem Robot Operating System (ROS). Tampilan ini dikembangkan menggunakan library Three.js sehingga operator dapat memantau posisi robot, tata letak terminal, serta jalur pergerakan robot pada area kerja tanpa harus mengamati robot secara langsung.

Pengujian dilakukan dengan menghubungkan website ke robot, kemudian menjalankan robot menuju beberapa titik tujuan. Selama proses tersebut diamati perubahan posisi objek robot pada tampilan 3D, kesesuaian posisi terhadap kondisi aktual, serta pembaruan informasi navigasi yang ditampilkan pada website.



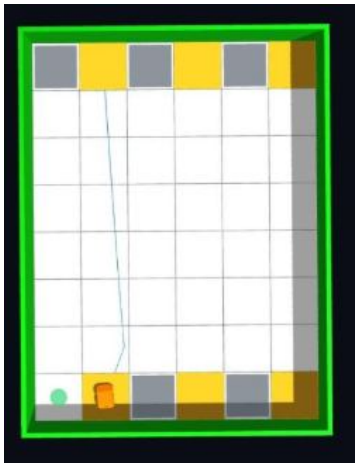
Gambar 4. 13 Tampilan 3D Maps Sebelum Robot Bergerak

Pada kondisi awal, robot berada pada posisi *Home Base* yang ditampilkan pada koordinat Col 0, Row 0. Tampilan 3D berhasil menampilkan denah area kerja berukuran 8 × 6 meter yang terdiri atas grid navigasi, posisi terminal pick-up (A, B, C), terminal drop-off (A−, B−, C−), serta posisi awal robot. Selain itu, informasi jarak tempuh, waktu operasi, dan kecepatan robot masih menunjukkan nilai awal karena proses navigasi belum dimulai.



Gambar 4. 14 Tampilan 3D Maps Saat Robot Bergerak

Ketika robot mulai menjalankan proses navigasi, posisi objek robot pada tampilan 3D berubah secara dinamis mengikuti data koordinat yang dikirimkan oleh sistem ROS. Jalur lintasan robot juga divisualisasikan pada peta sehingga operator dapat membedakan antara lintasan yang direncanakan dan lintasan aktual yang telah ditempuh robot. Informasi posisi, heading, serta jarak tempuh pada website diperbarui secara otomatis selama robot bergerak.



Gambar 4. 15 Tampilan 3D Maps Setelah Robot Mencapai Tujuan

Setelah robot mencapai titik tujuan, posisi akhir pada tampilan 3D sesuai dengan lokasi terminal tujuan yang ditentukan sebelumnya. Informasi status navigasi juga berubah menjadi selesai, sedangkan nilai jarak tempuh dan waktu operasi menampilkan hasil perjalanan yang telah dilakukan robot. Hal ini menunjukkan bahwa proses sinkronisasi data antara robot, ROS, dan website berjalan dengan baik sehingga visualisasi yang ditampilkan mampu merepresentasikan kondisi aktual robot selama proses pengiriman.

Berdasarkan hasil pengujian, tampilan 3D Maps berhasil menjalankan fungsinya sebagai media monitoring posisi robot secara real-time. Integrasi antara Three.js, ROSBridge WebSocket, dan data odometri memungkinkan setiap perubahan posisi robot divisualisasikan secara langsung pada website. Dengan demikian, operator dapat memantau kondisi operasional robot, posisi terminal, serta proses navigasi secara lebih mudah dan informatif selama sistem Warehouse Management System dijalankan.

4.3 Pengujian Decission Tree C4.5

Pada penelitian ini, sistem Autonomous Mobile Robot (AMR) dirancang untuk mampu melakukan navigasi dan pengambilan keputusan secara mandiri dalam proses distribusi barang antar terminal. Metode yang digunakan terdiri dari dua bagian utama, yaitu *single wheel odometry* untuk navigasi dan algoritma Decision Tree C4.5 untuk pengambilan keputusan prioritas pengantaran.

Metode single *wheel odometry* digunakan untuk menghitung posisi dan pergerakan robot berdasarkan data rotasi roda. Dengan memanfaatkan sensor rotary encoder pada satu roda, sistem dapat mengestimasi perpindahan jarak serta arah gerak robot secara real-time. Data odometri ini kemudian digunakan untuk menentukan posisi robot pada peta sehingga robot dapat bergerak menuju titik tujuan, baik pada terminal pick-up (A, B, C) maupun terminal drop-off (A-, B-, C-). Sementara itu, algoritma Decision Tree C4.5 diterapkan untuk menentukan prioritas pengantaran ketika robot menerima beberapa permintaan (request) secara bersamaan dari terminal pick-up. Dalam kondisi tersebut, sistem akan mengevaluasi beberapa parameter seperti jarak tempuh, waktu permintaan, serta

kondisi robot saat ini. Berdasarkan perhitungan nilai entropy dan information gain, C4.5 akan membentuk pohon keputusan yang menghasilkan prioritas pengantaran paling optimal.

Sebagai contoh, ketika terdapat request bersamaan dari terminal A, B, dan C, maka sistem akan memproses data tersebut dan menentukan urutan pengambilan barang yang harus dilakukan robot. Keputusan ini bertujuan untuk meminimalkan waktu tempuh dan meningkatkan efisiensi distribusi. Dengan menggabungkan metode navigasi berbasis odometri dan pengambilan keputusan menggunakan Decision Tree C4.5, AMR diharapkan mampu bekerja secara optimal, adaptif terhadap kondisi lingkungan, serta efisien dalam menyelesaikan tugas pengantaran barang antar terminal.

4.3.1 Skenario Pengujian Decision Tree C4.5

Skenario pengujian Decision Tree C4.5 disusun untuk memverifikasi kemampuan sistem dalam menentukan prioritas terminal *pick-up* berdasarkan parameter yang digunakan pada penelitian ini. Pengujian dilakukan sebelum robot melaksanakan proses penjemputan barang, sehingga keputusan yang dihasilkan oleh algoritma menjadi dasar bagi Warehouse Management System (WMS) dalam menentukan terminal yang akan dilayani terlebih dahulu oleh *Autonomous Mobile Robot* (AMR). Pengujian dilakukan dengan memberikan beberapa kombinasi *request* dari terminal A, B, dan C dengan kondisi atribut yang berbeda. Variasi kondisi tersebut meliputi perbedaan jenis barang, jarak robot terhadap terminal, serta waktu tunggu *request*. Setiap kondisi pengujian digunakan untuk mengamati proses pengambilan keputusan dan terminal yang dipilih sebagai prioritas berdasarkan *rule* Decision Tree C4.5 yang telah dirancang. Hasil dari setiap pengujian kemudian dibandingkan dengan kondisi atribut yang diberikan untuk memastikan bahwa keputusan sistem telah sesuai dengan mekanisme pengambilan keputusan yang diterapkan.

Gambar 4. 16 Kolom dan Baris pada Maps

Pengujian dilakukan pada lingkungan simulasi yang merepresentasikan area kerja robot berukuran 8 × 6 meter dengan tiga terminal *pick-up* (A, B, dan C) serta tiga terminal *drop-off* (A−, B−, dan C−) pada gambar 4.15 .Pada setiap skenario, ketiga terminal *pick-up* mengirimkan permintaan (*request*) secara bersamaan dengan kombinasi parameter yang berbeda. Parameter yang digunakan dalam proses pengambilan keputusan meliputi jenis barang, jarak robot terhadap terminal, dan waktu *request*.

Area kerja robot direpresentasikan dalam bentuk peta grid berukuran 8 kolom × 6 baris, dengan setiap satu sel grid setara dengan jarak 1 meter pada lingkungan nyata. Representasi ini digunakan sebagai acuan dalam perhitungan jarak antara posisi robot dengan terminal *pick-up*, yang dihitung menggunakan metode jarak Manhattan berdasarkan selisih posisi kolom dan baris pada grid. Selain sebagai acuan perhitungan jarak pada algoritma *Decision Tree* C4.5,

representasi grid ini juga digunakan sebagai dasar konversi jarak tempuh robot ke dalam satuan pulsa *encoder* pada pengujian jarak tempuh di subbab 4.3.2 hingga 4.3.4. Setiap sel pada peta grid merepresentasikan jarak 1 meter pada lingkungan nyata, sehingga posisi robot dan terminal dapat dinyatakan dalam satuan grid maupun meter secara setara. Skala ini menjadi dasar perhitungan jarak Manhattan antara robot dan terminal *pick-up* pada seluruh skenario pengujian.

Mekanisme pengambilan keputusan dilakukan secara bertahap sesuai aturan yang diterapkan pada algoritma Decision Tree C4.5. Sistem terlebih dahulu mengevaluasi parameter jenis barang sebagai prioritas utama. Apabila terdapat lebih dari satu terminal dengan prioritas jenis barang yang sama, maka sistem akan melanjutkan evaluasi berdasarkan parameter jarak robot terhadap terminal. Selanjutnya, apabila parameter jarak juga memiliki nilai yang sama, sistem menggunakan waktu *request* sebagai penentu prioritas akhir.

Untuk memverifikasi seluruh mekanisme tersebut, pengujian dibagi menjadi tiga kondisi yang mewakili setiap aturan (*rule*) pengambilan keputusan pada algoritma Decision Tree C4.5. Ringkasan skenario pengujian ditunjukkan pada Tabel 4.7.

Tabel 4. 7 Skenario Pengujian Decision Tree C4.5

Kondisi	Parameter yang Diuji	Rule yang Dijalankan
Kondisi 1	Jenis barang berbeda	Prioritas berdasarkan jenis barang
Kondisi 2	Jenis barang sama, jarak berbeda	Prioritas berdasarkan jarak
Kondisi 3	Jenis barang sama, jarak sama, waktu *request* berbeda	Prioritas berdasarkan waktu Tunggu *request*

4.3.2 Pengujian Kondisi 1 (Jenis Barang Berbeda)

Pengujian Kondisi 1 dilakukan untuk memverifikasi kemampuan algoritma Decision Tree C4.5 dalam menentukan prioritas terminal *pick-up* ketika setiap terminal memiliki jenis barang yang berbeda. Pada kondisi ini, parameter

jenis barang menjadi faktor utama dalam proses pengambilan keputusan, sehingga sistem diharapkan dapat langsung menentukan terminal prioritas tanpa perlu melakukan evaluasi terhadap parameter jarak maupun waktu *request*.

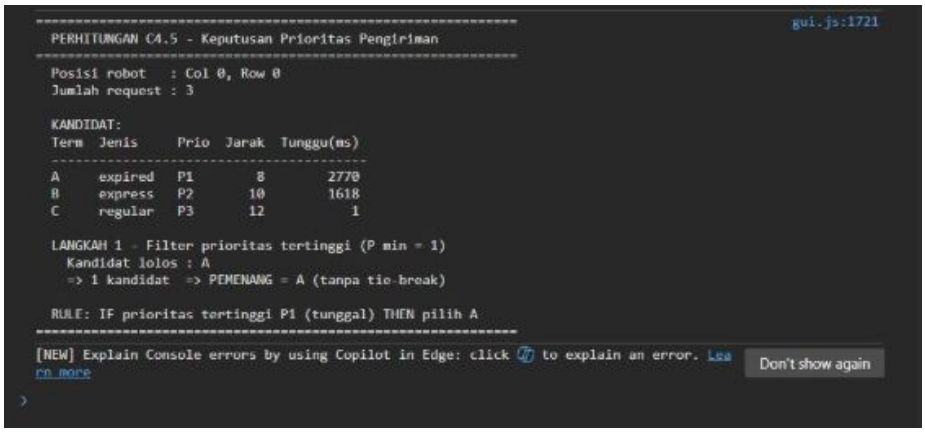
Tabel 4. 8 Parameter Pengujian Kondisi 1 (Jenis Barang Berbeda)

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Expired	P1	8 Grid	2770
B	Express	P2	10 Grid	1618
C	Regular	P3	12 Grid	1

Tabel 4. 9 Hasil Keputusan Kondisi 1 (Jenis Barang Yang Berbeda)

Terminal Terpilih	Prioritas Pemenang	Aturan (Rule) Keputusan
A	P1 (Expired)	IF jenis barang = Expired (P1) THEN pilih Terminal A

Pada skenario ini, ketiga terminal mengirimkan permintaan secara bersamaan dengan kombinasi jenis barang yang berbeda. Posisi awal robot berada pada *Home Base*, sedangkan parameter yang digunakan ditunjukkan pada Tabel 4.8.



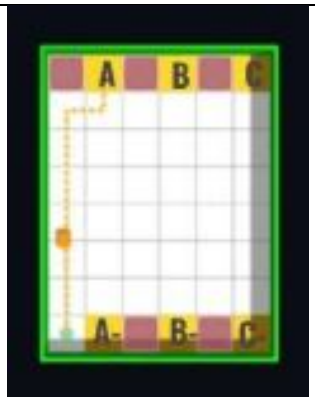
Gambar 4. 17 Gambar *Log* pada Web (Hasil Pemenenang Prioritas Request)

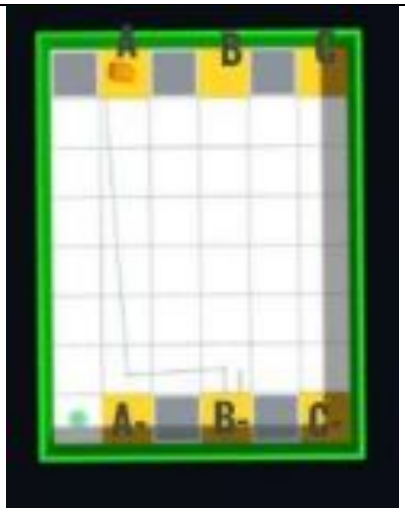
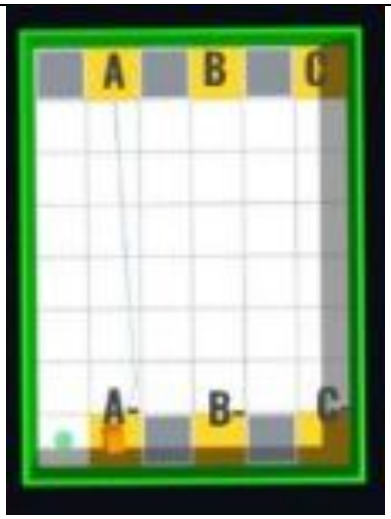
Hasil pengujian ditunjukkan pada Gambar 4.16. Algoritma Decision Tree C4.5 menetapkan Terminal A sebagai terminal prioritas. Keputusan ini diambil pada Langkah 1, yaitu penyaringan prioritas tertinggi (P minimum = 1), yang menghasilkan satu kandidat tunggal, yaitu Terminal A yang membawa barang

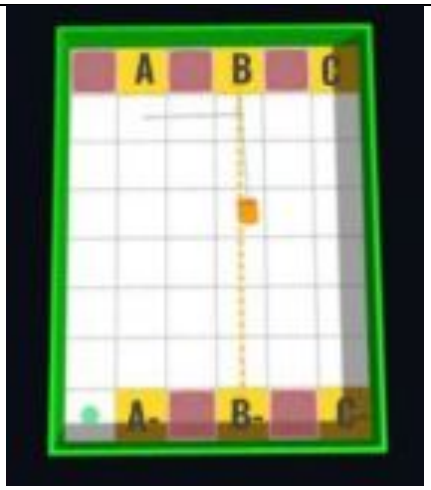
Expired. Karena hanya terdapat satu kandidat dengan prioritas tertinggi, proses *tie-break* berbasis *Gain Ratio* terhadap atribut jarak dan waktu tunggu tidak dijalankan, sebagaimana terlihat pada keluaran sistem yang menyatakan "1 kandidat → PEMENANG = A (tanpa tie-break)".

Hasil ini membuktikan bahwa jenis barang berfungsi sebagai atribut akar (root) pada pohon keputusan, yaitu atribut dengan tingkat kepentingan tertinggi dalam menentukan prioritas. Meskipun Terminal A memiliki jarak dan waktu tunggu yang tidak paling menguntungkan dibanding terminal lain, sistem tetap memilihnya karena prioritas jenis barang berada satu tingkat di atas atribut jarak maupun waktu tunggu. Dengan demikian, algoritma telah bekerja sesuai perancangan, yaitu mendahulukan urutan prioritas Expired (P1) > Express (P2) > Regular (P3) sebelum mempertimbangkan atribut lainnya. Aturan keputusan yang dihasilkan pada kondisi ini dirangkum pada Tabel 4.9.

Tabel 4. 10 Visualisasi Pengujian Kondisi 1 (Jenis Barang Berbeda)

Visualisasi 3D Maps	**Visualisasi *Real***
***Home Base* (0,0) (Posisi Awal Robot)**	
	
***Home Base* (0, 0) - Terminal A (1,7)**	
	

Terminal A (1,7) => Melakukan pick-up barang	
	
Terminal A (1,7) – Terminal A- (1,0)	
	
Terminal A- (1,0) => Proses drop-off selesai	
	

Terminal A- (1, 0) – Terminal B (3,7)	
	
Terminal B (3,7) - Terminal B- (3, 0)	
	
Terminal B- (3,0) – Terminal C (5,7)	
	

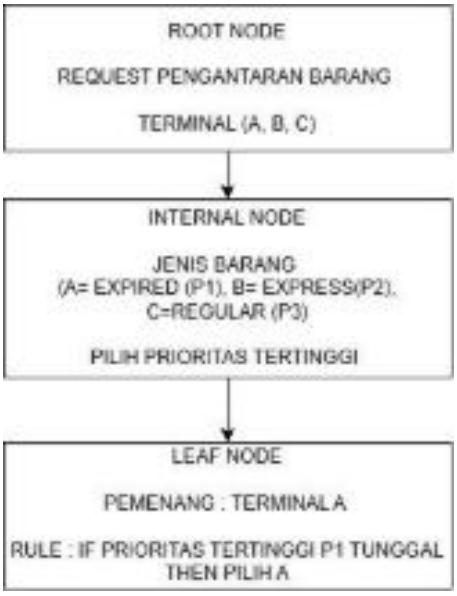
Terminal C (5,7) – Terminal C- (5,0)	
	
Terminal C- (5,0) => Seluruh pengiriman selesai	
	

Untuk membuktikan bahwa keputusan pemilihan Terminal A oleh algoritma Decision Tree C4.5 benar-benar diteruskan dan dieksekusi oleh robot secara fisik, disajikan Tabel 4.10 yang menampilkan visualisasi perjalanan robot dari posisi awal di Home Base hingga tiba di Terminal A. Tabel tersebut membandingkan tampilan posisi robot pada peta tiga dimensi (3D Maps) dengan kondisi nyata robot di lapangan pada setiap titik perjalanan, sehingga dapat dibuktikan bahwa pergerakan robot yang ditampilkan pada website sesuai dengan pergerakan robot yang sesungguhnya menuju terminal yang diprioritaskan.

Tabel 4. 11 Total Jarak tempuh Robot kondisi 1

Terminal	Jarak (Grid)	Jarak Tempuh (m)	Estimasi Pulse Encoder
A-B-C	47	47.16	99.737

Pada tabel 4.11 Terlihat jarak tempuh robot dihitung berdasarkan rute operasional penuh, mulai dari posisi awal di *Home Base* hingga seluruh permintaan pengantaran selesai diproses sesuai urutan prioritas, yaitu Terminal A, dilanjutkan Terminal B, dan diakhiri Terminal C. Total jarak tempuh pada pengujian ini tercatat sebesar 47 grid, setara dengan 47,16 meter, dengan estimasi 99.737 pulsa *encoder*. Nilai ini diperoleh melalui konversi jarak grid menggunakan parameter kinematik roda berdiameter 6 cm dan resolusi *encoder* 400 PPR, sehingga merepresentasikan jarak lintasan ideal yang ditempuh robot selama satu siklus operasi penuh, mulai dari pengambilan keputusan prioritas hingga seluruh barang selesai diantar ke terminal *drop-off* masing-masing.



Gambar 4. 18 Pohon Keputusan Pengujian Kondisi 1

Gambar 4.18 menunjukkan struktur pohon keputusan yang terbentuk pada pengujian Kondisi 1. Proses dimulai dari root node, yaitu permintaan pengantaran barang dari Terminal A, B, dan C. Penelusuran kemudian berlanjut ke internal node pertama, yaitu atribut jenis barang, yang memeriksa prioritas masing-masing kandidat: Terminal A membawa barang *Expired* (P1), Terminal B membawa barang *Express* (P2), dan Terminal C membawa barang *Regular* (P3). Karena Terminal A memiliki prioritas tertinggi dan hanya terdapat satu kandidat pada prioritas tersebut, penelusuran pohon berhenti pada simpul ini dan langsung menghasilkan leaf node, yaitu Terminal A sebagai pemenang, tanpa perlu mengevaluasi atribut jarak maupun waktu tunggu.

4.3.3 Pengujian Kondisi 2 (Jenis Barang Sama, Jarak Berbeda)

Pengujian Kondisi 2 dilakukan untuk memverifikasi kemampuan algoritma Decision Tree C4.5 dalam menentukan prioritas terminal pick-up ketika seluruh terminal memiliki jenis barang yang sama namun berada pada jarak yang berbeda. Karena jenis barang tidak lagi dapat membedakan prioritas (seluruh permintaan memiliki prioritas yang sama), sistem diharapkan melanjutkan proses pengambilan keputusan ke tahap tie-break berbasis Gain Ratio terhadap atribut jarak dan waktu tunggu. Pada skenario ini, ketiga terminal mengirimkan permintaan barang Express (prioritas P2) dalam waktu berdekatan, dengan posisi awal robot berada pada Home Base (Col 0, Row 0). Parameter pengujian ditunjukkan pada Tabel 4.12.

Tabel 4. 12 Parameter Pengujian Kondisi 2 (Jenis Barang Sama, Jarak Berbeda)

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Express	P2	8 Grid	15420
B	Express	P2	10 Grid	11761
C	Express	P2	12 Grid	1

Berdasarkan Tabel 4.12, seluruh terminal memiliki prioritas yang sama, yaitu P2, sehingga penyaringan prioritas tertinggi pada Langkah 1 menghasilkan tiga kandidat sekaligus (A, B, dan C). Karena terdapat lebih dari satu kandidat dengan prioritas tertinggi, algoritma melanjutkan ke Langkah 2, yaitu proses tie-break menggunakan Gain Ratio. Pada tahap ini sistem terlebih dahulu menurunkan kelas urgensi sebagai target entropi (A=1, B=0, C=0) sehingga diperoleh Entropy awal E(S) = 0,918. Selanjutnya dihitung Gain Ratio untuk masing-masing atribut sebagaimana ditunjukkan pada Tabel 4.13.

Tabel 4. 13 Perhitungan Gain Ratio Tie-break Kondisi 2

Atribut	Info (A)	Gain	Split Info	Gain Ratio	Bobot
Jarak	0,000	0,918	0,918	1,000	0,78
Waktu Tunggu	0,667	0,252	0,918	0,274	0,22

Hasil perhitungan menunjukkan atribut jarak memperoleh Gain Ratio tertinggi (1,000) dibanding waktu tunggu (0,274), sehingga atribut jarak diberi bobot lebih besar, yaitu 0,78 berbanding 0,22. Artinya, dalam kondisi jenis barang yang sama, jarak menjadi atribut yang paling informatif dalam menentukan urgensi terminal. Berdasarkan bobot tersebut, sistem menghitung skor akhir setiap terminal seperti ditunjukkan pada Tabel 4.14.

Tabel 4. 14 Skor Akhir dan Peringkat Kondisi 2

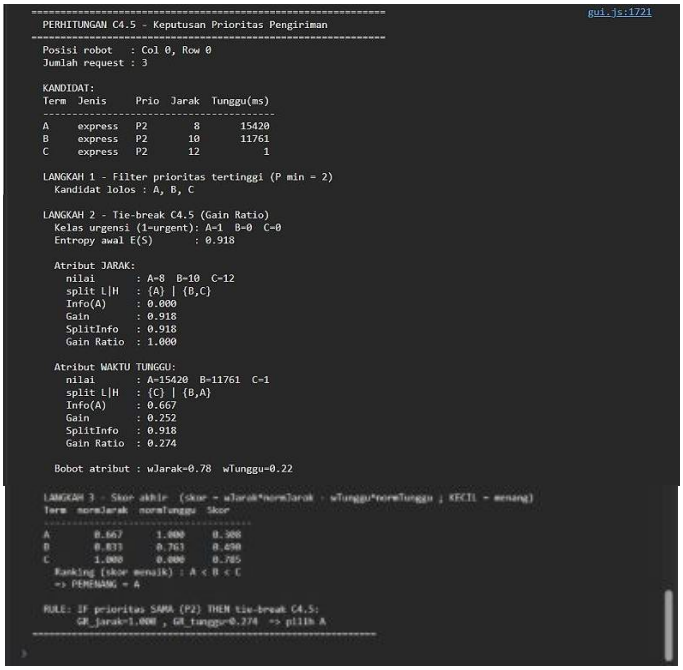
Terminal	norm Jarak	norm Tunggu	Skor	Peringkat
A	0,667	1,000	0,308	1
B	0,833	0,763	0,490	2
C	1,000	0,000	0,785	3

Berdasarkan Tabel 4.14, Terminal A memperoleh skor terendah (0,308), diikuti Terminal B (0,490) dan Terminal C (0,785). Karena skor terendah menandakan prioritas tertinggi, algoritma menetapkan Terminal A sebagai pemenang. Hasil ini sesuai dengan jarak Terminal A yang paling dekat (8 sel) dibanding Terminal B (10 sel) dan Terminal C (12 sel). Terminal A juga memiliki waktu tunggu terlama, sehingga kedua atribut sama-sama menguatkan pemilihannya, dengan jarak sebagai atribut yang dominan.

Hasil pengujian ini membuktikan bahwa ketika jenis barang tidak dapat membedakan prioritas, atribut jarak berperan sebagai faktor penentu utama pada proses tie-break, ditunjukkan oleh nilai Gain Ratio jarak tertinggi (1,000). Setelah Terminal A selesai diproses, sistem melanjutkan pemilihan terhadap sisa permintaan dan secara konsisten mendahulukan terminal terdekat berikutnya, yaitu B kemudian C, sehingga urutan pemrosesan mengikuti pola jarak A → B → C. Dengan demikian, algoritma Decision Tree C4.5 telah bekerja sesuai perancangan dalam kondisi jenis barang sama dengan jarak yang berbeda. Aturan keputusan yang dihasilkan dirangkum pada Tabel 4.15.

Tabel 4. 15 Hasil Pengujian Kondisi 2

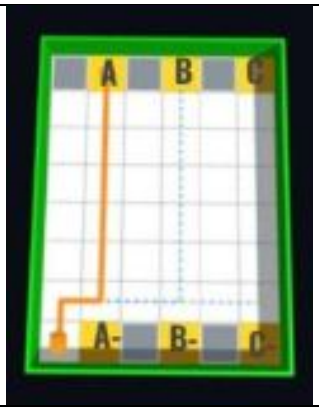
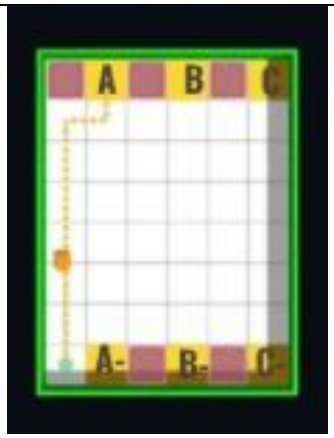
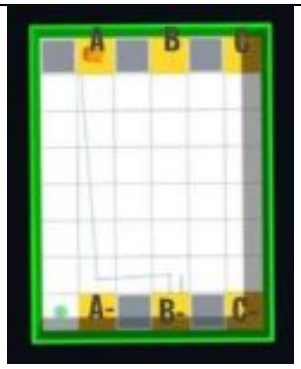
Terminal Terpilih	Atribut Penentu	Aturan (*Rule*) Keputusan
A	Jarak (GR = 1,000, dominan)	IF prioritas sama (P2) THEN tie-break C4.5 → jarak dominan → pilih terminal terdekat (A)

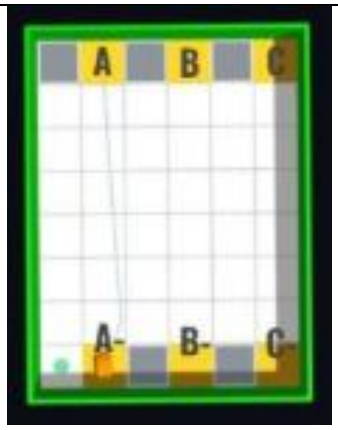
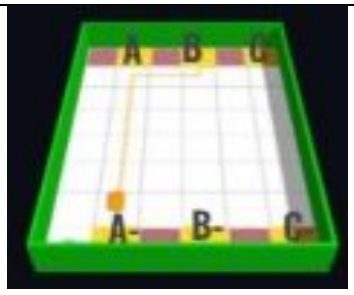
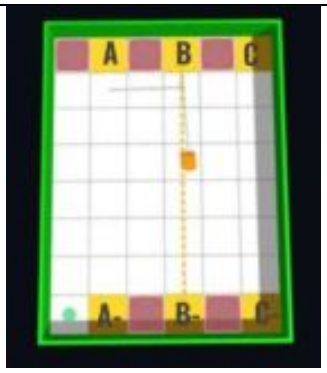


Gambar 4. 19 Gambar Log pada Web (Hasil Pemenenang Prioritas Request)

Gambar 4.19 menunjukkan keluaran sistem berupa log perhitungan algoritma Decision Tree C4.5 yang tercatat pada konsol browser saat pengujian Kondisi 2 dijalankan. Log tersebut memuat seluruh tahapan pengambilan keputusan secara berurutan, mulai dari data kandidat (jenis barang, prioritas, jarak, dan waktu tunggu tiap terminal), proses penyaringan prioritas pada Langkah 1, perhitungan Entropy awal, Information Gain, Split Info, dan Gain Ratio untuk atribut jarak serta waktu tunggu pada Langkah 2, hingga perhitungan skor akhir dan penentuan terminal pemenang pada Langkah 3. Nilai-nilai yang ditampilkan pada gambar ini sama persis dengan nilai yang disajikan pada Tabel 4.12 hingga Tabel 4.14, sehingga membuktikan bahwa hasil perhitungan manual yang dijabarkan pada subbab ini sesuai dengan hasil perhitungan yang dilakukan secara otomatis oleh sistem.

Tabel 4. 16 Visualisasi Pengujian Kondisi 2 (Jenis Barang Sama, Jarak Berbeda)

Visualisasi 3D Maps	Visualisasi *Real*
***Home Base* (0,0) (Posisi Awal Robot)**	
	
***Home Base* (0, 0) - Terminal A (1,7)**	
	
Terminal A (1,7) => Melakukan pick-up barang	
	
Terminal A (1,7) – Terminal A- (1,0)	

	
Terminal A- (1,0) => Proses drop-off selesai	
	
Terminal A- (1, 0) – Terminal B (3,7)	
	
Terminal B (3,7) - Terminal B- (3, 0)	
	
Terminal B- (3,0) – Terminal C (5,7)	

	
Terminal C (5,7) – Terminal C- (5,0)	
	
Terminal C- (5,0) => Seluruh pengiriman selesai	
	

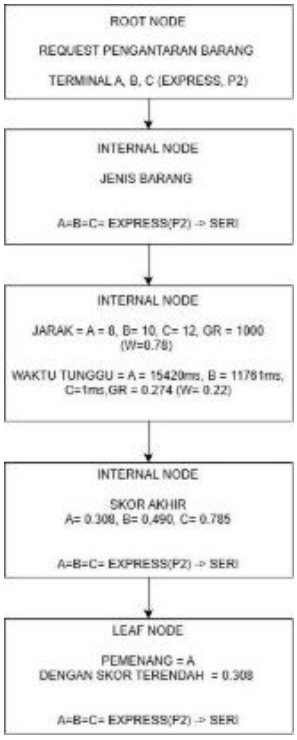
Untuk membuktikan bahwa keputusan tie-break yang memilih Terminal A berdasarkan atribut jarak benar-benar diteruskan dan dieksekusi oleh robot secara fisik, disajikan Tabel 4.16 yang menampilkan visualisasi perjalanan robot dari posisi awal di Home Base menuju Terminal A, yaitu terminal dengan jarak terdekat yang menjadi pemenang pada pengujian ini. Tabel tersebut membandingkan tampilan posisi robot pada peta tiga dimensi (3D Maps) dengan

kondisi nyata robot di lapangan pada setiap titik perjalanan, sehingga dapat dibuktikan bahwa posisi robot pada tampilan website selalu konsisten dengan posisi fisik robot yang sesungguhnya.

Tabel 4. 17 Jarak yang tempuh robot kondisi 2

Terminal	Jarak (Grid)	Jarak Tempuh (m)	Estimasi Pulse Encoder
A-B-C	47	47.2	100.234

Sama seperti pada pengujian Kondisi 1, Pada tabel 4.17 terlihat jarak tempuh pada pengujian ini dihitung berdasarkan rute operasional penuh, dimulai dari *Home Base* hingga seluruh permintaan pengantaran dari Terminal A, B, dan C selesai diproses sesuai urutan hasil *tie-break* algoritma *Decision Tree* C4.5. Total jarak tempuh tercatat sebesar 47 grid, setara dengan 47.2 meter, dengan estimasi 99.737 pulsa *encoder*. Nilai jarak tempuh pada pengujian ini identik dengan Kondisi 1, sebagai konsekuensi dari posisi awal dan urutan pemrosesan terminal yang sama pada kedua pengujian, meskipun dasar pengambilan keputusan pada masing-masing kondisi berbeda.



Gambar 4. 20 Pohon Keputusan Pengujian Kondisi 2

Gambar 4.19 menunjukkan struktur pohon keputusan pada pengujian Kondisi 2. Penelusuran dimulai dari root node, yaitu permintaan pengantaran dari Terminal A, B, dan C yang seluruhnya membawa barang *Express* (P2). Karena ketiga kandidat memiliki prioritas yang sama, internal node pertama pada atribut jenis barang tidak menghasilkan keputusan dan penelusuran dilanjutkan ke internal node kedua, yaitu simpul *tie-break* berbasis *Gain Ratio*, yang menggabungkan atribut jarak (*Gain Ratio* = 1,000, bobot 0,78) dan waktu tunggu (*Gain Ratio* = 0,274, bobot 0,22) untuk menghitung skor tiap kandidat. Hasil perhitungan pada internal node ketiga menunjukkan Terminal A memperoleh skor terendah, yaitu 0,308, dibandingkan Terminal B (0,490) dan Terminal C (0,785). Karena skor terendah menandakan prioritas tertinggi, penelusuran pohon berakhir pada leaf node dengan Terminal A sebagai pemenang.

4.3.4 Pengujian Kondisi 3 (Jenis Barang Sama, Jarak Sama, *Request* Berbeda)

Pengujian Pengujian Kondisi 3 dilakukan untuk memverifikasi kemampuan algoritma *Decision Tree* C4.5 dalam menentukan prioritas terminal *pick-up* pada kondisi ekstrem, yaitu ketika dua terminal memiliki jenis barang yang sama sekaligus jarak yang sama persis terhadap posisi robot, sehingga penelusuran pohon keputusan harus dilanjutkan hingga simpul ketiga (*leaf node*), yaitu atribut waktu tunggu. Posisi awal robot pada pengujian ini diatur pada Terminal B− (Col 3, Row 0), sehingga jarak Manhattan dari posisi tersebut menuju Terminal A dan Terminal C bernilai sama, yaitu 9 sel. Terminal B sengaja tidak diikutsertakan dalam pengujian ini karena jaraknya yang lebih dekat akan langsung menghasilkan keputusan pada simpul kedua (jarak), sehingga tujuan pengujian untuk mencapai simpul ketiga tidak akan tercapai. Kedua permintaan dikirim dengan jenis barang yang sama, yaitu *Express* (prioritas P2), pada selang waktu yang berbeda. Parameter pengujian ditunjukkan pada Tabel 4.18.

Tabel 4. 18 Parameter Pengujian Kondisi 3

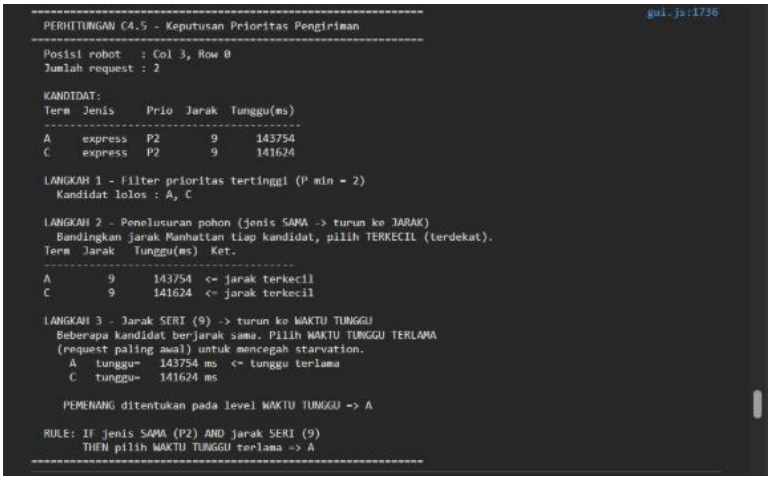
Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)	Keterangan
A	Express	P2	9 Grid	143754	Jarak seri
C	Express	P2	9 Grid	141624	Jarak Seri

Berdasarkan Tabel 4.18, Terminal A dan Terminal C memiliki jenis barang serta jarak yang sama persis, sehingga penyaringan pada simpul akar (Langkah 1) meloloskan kedua terminal sekaligus, dan penelusuran pada simpul kedua (Langkah 2) tidak dapat menentukan pemenang karena kedua kandidat memiliki jarak yang identik, yaitu 9 sel, sebagaimana ditunjukkan pada Tabel 4.18. Karena jarak kedua kandidat seri, algoritma melanjutkan penelusuran ke simpul ketiga (*leaf node*), yaitu atribut waktu tunggu, dengan aturan memilih kandidat yang menunggu paling lama untuk mencegah kondisi *starvation* pada permintaan yang datang lebih awal.

Terminal A memiliki waktu tunggu yang lebih lama (143754 ms) dibandingkan Terminal C (141624 ms), sehingga algoritma menetapkan Terminal A sebagai pemenang pada simpul ketiga. Hasil ini membuktikan bahwa struktur pohon keputusan pada sistem mampu menangani kondisi ekstrem, yaitu ketika dua atribut pertama (jenis barang dan jarak) tidak dapat membedakan prioritas, dengan tetap menghasilkan keputusan yang konsisten dan tidak mengalami kebuntuan (*deadlock*) melalui atribut ketiga, yaitu waktu tunggu. Aturan keputusan yang dihasilkan dirangkum pada Tabel 4.19.

Tabel 4. 19 Hasil Pengujian Kondisi 3

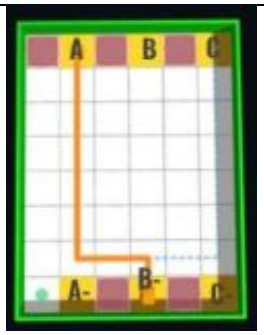
Terminal Terpilih	Atribut Penentu	Aturan (*Rule*) Keputusan
A	Waktu Tunggu (simpul ke-3, terlama)	IF jenis SAMA (P2) AND jarak SERI (9) THEN pilih WAKTU TUNGGU terlama → A

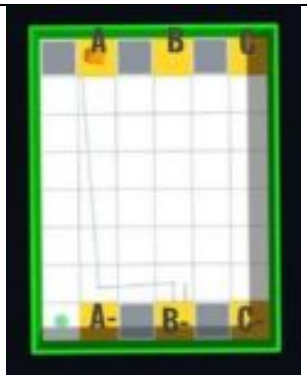
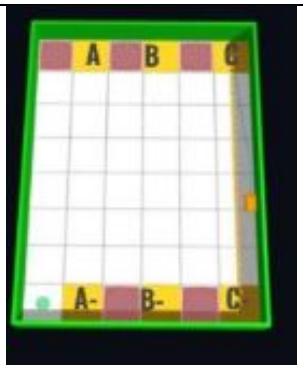


Gambar 4. 21 Gambar Log Percobaan Kondisis 3

Gambar 4.21 menunjukkan keluaran sistem berupa log penelusuran pohon keputusan C4.5 pada konsol *browser*, yang memuat data kandidat, hasil penyaringan pada simpul akar, jarak yang seri pada simpul kedua, hingga penentuan pemenang pada simpul ketiga berdasarkan waktu tunggu.

Tabel 4. 20 Visualisasi Pengujian Kondisi 3 (Jenis Barang Sama, Jarak Sama, Waktu Request Beda)

Visualisasi 3D Maps	**Visualisasi *Real***
Terminal B- (3, 0) (Posisi Awal Robot)	
	
Terminal B- (3, 0) - Terminal A (1,7)	
	

Terminal A (1,7) => Melakukan pick-up barang	
	
Terminal A (1,7) – Terminal C (5,7)	
	
Terminal C (5,7) => Melakukan pick-up barang	
	
Terminal C (5,7) – Terminal C- (5,0)	
	

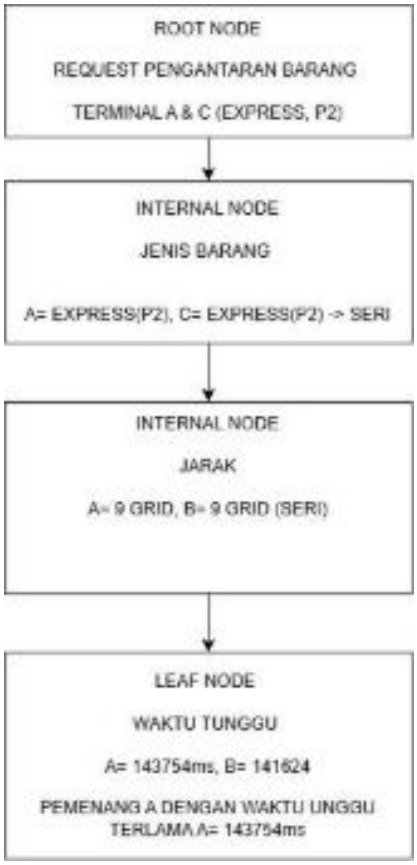
Terminal C- (5,0) => Seluruh pengiriman selesai		
		

Untuk membuktikan bahwa keputusan tie-break yang memilih Terminal A berdasarkan atribut waktu tunggu benar-benar diteruskan dan dieksekusi oleh robot secara fisik, disajikan Tabel 4.18 yang menampilkan visualisasi perjalanan robot dari posisi awal di Terminal B- menuju Terminal A. Tabel tersebut membandingkan tampilan posisi robot pada peta tiga dimensi (3D Maps) dengan kondisi nyata robot di lapangan pada setiap titik perjalanan, sehingga dapat dibuktikan bahwa robot secara fisik benar-benar berangkat dari posisi B- sesuai dengan skenario pengujian, dan bergerak menuju terminal yang diprioritaskan oleh algoritma.

Tabel 4. 21 Total Jarak Tempuh Robot Kondisi 3

Terminal	Jarak (Grid)	Jarak Tempuh (m)	Estimasi Pulse Encoder
A-C	34	34.3	72.150

Berbeda dengan dua pengujian sebelumnya, Pada Tabel 4.21 terlihat jarak tempuh pada Kondisi 3 dihitung mulai dari posisi awal robot di Terminal B− hingga seluruh permintaan dari Terminal A dan Terminal C selesai diproses. Total jarak tempuh pada pengujian ini tercatat sebesar 34 grid, setara dengan 34.3 meter, dengan estimasi 72.150 pulsa *encoder*. Jarak tempuh yang lebih pendek dibandingkan Kondisi 1 dan Kondisi 2 disebabkan oleh posisi awal robot yang lebih dekat dengan kedua terminal pengujian, serta jumlah permintaan yang lebih sedikit, yaitu hanya dua terminal dibandingkan tiga terminal pada pengujian sebelumnya.



Gambar 4. 22 Pohon Keputusan Pengujian Kondisi 3

Gambar 4.21 menunjukkan struktur pohon keputusan pada pengujian Kondisi 3, yaitu kondisi paling ekstrem dalam pengujian. Penelusuran dimulai dari root node, yaitu permintaan dari Terminal A dan Terminal C yang sama-sama membawa barang *Express* (P2). Pada internal node pertama, atribut jenis barang tidak dapat membedakan kedua kandidat karena bernilai sama, sehingga penelusuran dilanjutkan ke internal node kedua, yaitu atribut jarak. Namun, jarak kedua kandidat juga bernilai identik, yaitu 9 sel, sehingga simpul ini pun tidak menghasilkan keputusan. Penelusuran akhirnya mencapai leaf node, yaitu atribut waktu tunggu, dengan Terminal A tercatat menunggu selama 143.754 milidetik dan Terminal C selama 141.624 milidetik. Karena Terminal A memiliki waktu tunggu yang lebih lama, sistem menetapkan Terminal A sebagai pemenang, sementara Terminal C dinyatakan tereliminasi. Hasil ini membuktikan bahwa struktur pohon keputusan pada sistem tidak mengalami kebuntuan (*deadlock*) sekalipun dua atribut pertama bernilai identik, karena atribut ketiga tetap mampu menghasilkan keputusan akhir yang pasti.

4.3.5 Perhitungan Manual *Decision Tree*

Perhitungan manual pada subbab ini menggunakan data hasil pengujian Kondisi 2 (Jenis Barang Sama, Jarak Berbeda) yang telah diuraikan pada subbab 4.3.3, yaitu skenario dengan tiga permintaan pengiriman berjenis barang sama (Express, prioritas P2) namun berada pada jarak yang berbeda terhadap posisi robot di Home Base. Kasus ini dipilih karena penyaringan prioritas pada Langkah 1 tidak menghasilkan pemenang tunggal, sehingga proses tie-break berbasis Gain Ratio benar-benar dijalankan oleh sistem, berbeda dengan kasus pada Percobaan 1 yang telah selesai pada tahap penyaringan prioritas. Perhitungan manual berikut disusun secara bertahap, mulai dari penurunan kelas urgensi, perhitungan Entropy, Information Gain, Split Information, hingga Gain Ratio pada masing-masing atribut, dengan tujuan untuk membuktikan bahwa hasil perhitungan secara matematis konsisten dengan keluaran sistem yang ditampilkan pada Tabel.4.15.

Data diambil dari hasil pengujian Kondisi 2 pada subbab 4.3.3, dengan posisi awal robot di *Home Base* (Col 0, Row 0):

Tabel 4. 22 Parameter Pengujian Kondisi 2 untuk Perhitungan Manual

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Express	P2	8 Grid	15420
B	Express	P2	10 Grid	11761
C	Express	P2	12 Grid	1

Karena ketiga terminal memiliki prioritas yang sama (P2), penyaringan prioritas pada Langkah 1 tidak menghasilkan pemenang tunggal, sehingga seluruh kandidat S = {A, B, C} dilanjutkan ke proses *tie-break*.

4.3.5.1 Menurunkan Kelas Urgensi

Karena data tidak memiliki label kelas secara langsung, sistem terlebih dahulu menurunkan kelas urgensi sebagai target perhitungan Entropy. Kelas ini dihitung dari kombinasi jarak dan waktu tunggu yang telah dinormalisasi:

$$nd_i = \frac{jarak_i}{jarak_{maks}} \quad (4.6)$$

$$n\omega_i = \frac{tunggu_i}{tunggu_{maks}} \quad (4.7)$$

$$kombinasi_i = nd_i - n\omega_i \quad (4.8)$$

Nilai maksimum: $jarak_{maks} = 12\ sel$, $tunggu_{maks} = 15420\ ms$.

Terminal A:

$$nd_A = \frac{8}{12} = 0{,}667$$

$$n\omega_A = \frac{15420}{15420} = 1{,}000$$

$$kombinasi_A = 0{,}667 - 1{,}000 = -0{,}333$$

Terminal B:

$$nd_B = \frac{10}{12} = 0{,}833$$

$$n\omega_B = \frac{11761}{15420} = 0{,}763$$

$$kombinasi_B = 0{,}833 - 0{,}763 = 0{,}070$$

Terminal C:

$$nd_C = \frac{12}{12} = 1{,}000$$

$$n\omega_C = \frac{1}{15420} \approx 0{,}000$$

$$kombinasi_C = 1{,}000 - 0{,}000 = 1{,}000$$

Nilai kombinasi terendah menandakan tingkat urgensi tertinggi (jarak dekat dan/atau waktu tunggu lama). Data diurutkan menaik: A (−0,333) < B (0,070) < C (1,000). Pembagian dilakukan pada titik tengah $\lfloor 3/2 \rfloor = 1$, sehingga kandidat dengan nilai kombinasi terendah masuk kelompok "urgent":

$$Urgent = \{A\} => A = 1$$

$$Tidak\ Urgent = \{B, C\} => B = 0, C = 0$$

Hasil ini sesuai dengan keluaran sistem pada Tabel 4.15 : *"Kelas urgensi (1=urgent): A=1 B=0 C=0"*.

4.3.5.2 Entrophy Awal E(S)

Dari 3 data: 1 berkelas *urgent* (A), 2 berkelas *tidak urgent* (B, C).

$$P_{urgent} = \frac{1}{3}$$

$$P_{tidak} = \frac{2}{3}$$

$$Entrophy\ (S) = -P_{urgent} \log_2 P_{urgent} - P_{tidak} \log_2 P_{tidak} \quad (4.9)$$

$$Entrophy\ (S) = -\frac{1}{3}\log_2 \frac{1}{3} - \frac{2}{3}\log_2 \frac{2}{3}$$

$$Entrophy\ (S) = 0{,}528 + 0{,}390 = 0{,}918$$

Nilai ini sesuai dengan keluaran sistem: *"Entropy awal E(S): 0.918"*.

4.3.5.3 Gain Ratio Atribut Jarak

Pembagian data (split) berdasarkan urutan nilai jarak (A=8, B=10, C=12 — sudah terurut menaik). Titik pisah $\lfloor 3/2 \rfloor = 1$:

$$Low = \{A\}$$

$$High = \{B, C\}$$

Entropy tiap kelompok:

- $Low = \{A\}$: hanya 1 data, kelas urgent murni => $Entropy(Low) = 0$
- $High = \{B, C\}$: 2 data, keduanya tidak-urgent, murni => $Entropy(High) = 0$

Information setelah split:

$$Info(Jarak) = \frac{|Low|}{|S|} Entrophy(Low) + \frac{|High|}{S} Entrophy(High) \quad (4.10)$$

$$Info(Jarak) = \frac{1}{3}(0) + \frac{2}{3}(0) = 0$$

Information Gain:

$$Gain(Jarak) = Entrophy(S) - Info(Jarak)$$
$$Gain(Jarak) = 0{,}918 - 0{,}000$$
$$Gain(Jarak) = 0{,}918$$

Split Information (mengukur seberapa merata ukuran subset 1 dan 2 dari 3):

$$SplitInfo(Jarak) = -\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3}$$
$$SplitInfo(Jarak) = 0{,}528 + 0{,}390$$
$$SplitInfo(Jarak) = 0{,}918$$

Gain Ratio:

$$GainRatio(Jarak) = \frac{Gain(Jarak)}{Split(Jarak)} = \frac{0{,}918}{0{,}918} = 1{,}000$$

4.3.5.4 Gain Ratio Atribut Tunggu

Pembagian data berdasarkan urutan nilai waktu tunggu (urut menaik: C=1, B=11761, A=15420). Titik pisah 1:

$$Low = \{C\}$$
$$High = \{B, A\}$$

Entropy tiap kelompok:

- $Low = \{C\}$: 1 data, tidak urgent murni => $Entropy(Low) = 0$
- $High = \{B, A\}$: berisi B (tidak-urgent) dan A (urgent) — campuran, masing-masing $p = \frac{1}{2}$:

$$Entrophy\ (High) = -\frac{1}{3}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2} = 1{,}000$$

Information setelah split:

$$Info(Tunggu) = \frac{1}{3}(0) + \frac{2}{3}(1{,}000) = 0{,}667$$

Information Gain:

$$Gain(Tunggu) = Entrophy(S) - Info(Tunggu) \qquad (4.11)$$

$$Gain(Tunggu) = 0{,}918 - 0{,}667$$

$$Gain(Tunggu) = 0{,}252$$

Split Information (ukuran subset sama seperti Jarak, yaitu 1:2 dari 3):

$$SplitInfo(Tunggu) = 0{,}918$$

Gain Ratio:

$$GainRatio(Tunggu) = \frac{Gain(Tunggu)}{SplitInfo(Tunggu)} = \frac{0{,}252}{0{,}918} = 0{,}274$$

4.3.5.5 Penentuan Bobot Atribut

Bobot masing-masing atribut ditentukan secara proporsional terhadap nilai *Gain Ratio*-nya:

$$\omega_{Jarak} = \frac{GainRatio(Jarak)}{GainRatio(Jarak) + GainRatio(Tunggu)} = \frac{1{,}000}{1{,}000 + 0{,}274}$$

$$\omega_{Jarak} = \frac{1{,}000}{1{,}000 + 0{,}274} = 0{,}78$$

$$\omega_{Tunggu} = 1 - \omega_{Jarak} = 0{,}22$$

Interpretasi: karena *Gain Ratio* jarak (1,000) jauh lebih besar dibandingkan waktu tunggu (0,274), atribut jarak terbukti lebih informatif dalam memisahkan kelas urgensi pada kasus ini, sehingga diberi bobot yang jauh lebih dominan dalam perhitungan skor akhir.

4.3.5.6 Perhitungan Skor Akhir

Skor dihitung dengan rumus:

$$skor_i = \omega_{Jarak}\ x\ nd_i - \omega_{Tunggu}\ x\ n\omega_i$$

Nilai skor terkecil menunjukkan prioritas tertinggi.

Terminal A:

$$skor_A = (0{,}78\ x\ 0{,}667) - (0{,}22\ x\ 1{,}000)$$

$$skor_A = 0{,}520 - 0{,}220 = 0{,}300$$

Terminal B:

$$skor_B = (0{,}78\ x\ 0{,}833) - (0{,}22\ x\ 0{,}763)$$

$$skor_B = 0{,}650 - 0{,}618 = 0{,}482$$

Terminal C:

$$skor_C = (0{,}78\ x\ 1{,}000) - (0{,}22\ x\ 0{,}000)$$

$$skor_C = 0{,}780 - 0{,}000 = 0{,}780$$

(Selisih dua digit terakhir dibanding keluaran sistem — A=0,308; B=0,490; C=0,785 — murni akibat pembulatan pada langkah normalisasi manual; nilai program menggunakan presisi desimal penuh.)

Ranking (skor menaik):

$$A(0{,}300) < B(0{,}482) < C(0{,}780)$$

Pemenang = A

Sehingga, aturan keputusan (*rule*) adalah,

IF prioritas SAMA (P2) THEN tie-break C4.5: $GainRatio_{Jarak} = 1{,}000$, $GainRatio_{Tunggu} = 0{,}274$ => pilih A

Seluruh tahapan perhitungan manual mulai dari penurunan kelas urgensi (A=1, B=0, C=0), Entropy awal E(S)=0,918, *Gain Ratio* jarak=1,000, *Gain Ratio* waktu tunggu=0,274, bobot $\omega_{Jarak} = 0{,}78$ dan $\omega_{Tunggu} = 0{,}22$, hingga skor akhir A=0,308, B=0,490, C=0,785, sesuai dengan keluaran sistem yang ditunjukkan pada Tabel 4.15. Hal ini membuktikan bahwa implementasi algoritma *Decision Tree* C4.5 pada sistem telah bekerja sesuai dengan perhitungan teoritisnya. Pada kasus Kondisi 2 ini, atribut jarak terbukti menjadi faktor penentu yang dominan, ditunjukkan oleh nilai *Gain Ratio* jarak yang jauh lebih tinggi (1,000) dibandingkan atribut waktu tunggu (0,274), sehingga Terminal A yang memiliki jarak terdekat ditetapkan sebagai terminal prioritas.

4.4 Pengujian Integrasi Sistem

Pengujian integrasi sistem dilakukan untuk memverifikasi bahwa seluruh komponen penyusun sistem, yaitu antarmuka *website*, basis data, dan robot, dapat

saling berkomunikasi serta bertukar data secara konsisten dan *real-time*. Apabila pada subbab sebelumnya pengujian difokuskan pada kebenaran algoritma pengambilan keputusan, maka pada subbab ini pengujian difokuskan pada keandalan aliran data antarkomponen, yaitu memastikan bahwa data yang dihasilkan pada satu komponen diterima dan tersimpan secara utuh pada komponen lainnya tanpa mengalami perbedaan nilai maupun kehilangan data.

Pengujian integrasi sistem ini dilakukan menggunakan satu kasus percobaan yang sama untuk menguji tiga aspek sinkronisasi secara terpadu, yaitu sinkronisasi antara *website* dan basis data, sinkronisasi antara *website* dan robot, serta pengujian protokol komunikasi *ROSBridge WebSocket*. Penggunaan satu kasus percobaan yang sama bertujuan agar setiap aspek sinkronisasi dapat ditelusuri pada peristiwa yang identik, sehingga konsistensi data dapat diverifikasi secara menyeluruh dari titik masukan di *website* hingga tersimpan di basis data dan diteruskan ke robot.

Kasus yang digunakan pada pengujian ini merupakan skenario Kondisi 2, yaitu tiga permintaan pengiriman yang memiliki jenis barang sama namun berada pada jarak yang berbeda, sehingga memicu proses *tie-break* pada algoritma *Decision Tree* C4.5. Alur prosesnya diawali ketika operator mengirimkan tiga permintaan melalui *website*, yaitu permintaan dari Terminal A, Terminal B, dan Terminal C dengan jenis barang *Express* (prioritas P2). Karena ketiga permintaan memiliki prioritas yang sama, algoritma melanjutkan ke proses *tie-break* berbasis *Gain Ratio* terhadap atribut jarak dan waktu tunggu. Berdasarkan perhitungan tersebut, atribut jarak memperoleh bobot dominan (0,78) sehingga sistem menetapkan Terminal A yang memiliki jarak terdekat (8 sel) sebagai terminal prioritas dengan skor terendah, yaitu 0,308. Hasil keputusan ini selanjutnya diteruskan kepada robot melalui protokol *ROSBridge WebSocket* agar robot menavigasi menuju terminal yang dipilih, sekaligus disimpan ke dalam basis data MongoDB beserta seluruh data pendukungnya, seperti skor keputusan, jarak, waktu tunggu, koordinat, dan status pengiriman.

Data hasil pengujian ditelusuri dari tiga sumber yang saling berkaitan, yaitu tampilan permintaan pada *website*, keluaran log pada konsol *browser* (*Developer Tools/F12*), serta dokumen yang tersimpan pada basis data MongoDB. Ketiga sumber ini kemudian dibandingkan untuk memastikan bahwa nilai-nilai kunci, seperti terminal terpilih, skor keputusan, jarak, dan waktu tunggu, bernilai sama di seluruh komponen. Kesesuaian nilai pada ketiga sumber tersebut menjadi indikator bahwa integrasi sistem telah berjalan dengan baik. Hasil pengujian untuk masing-masing aspek sinkronisasi diuraikan secara rinci pada subbab 4.4.1 hingga 4.4.3.

4.4.1 Sinkronisasi Website dan Database

Pengujian sinkronisasi *website* dan basis data dilakukan untuk memverifikasi bahwa setiap permintaan pengiriman beserta hasil penelusuran pohon keputusan C4.5 yang diproses pada *website* tersimpan secara utuh dan akurat ke dalam basis data MongoDB. Pengujian ini menggunakan sesi percobaan dengan tiga permintaan pengiriman berjenis barang sama (*Express*), yaitu Terminal A, Terminal B, dan Terminal C, dengan posisi awal robot berada pada Terminal B− (Col 3, Row 0). Metode pengujian dilakukan dengan membandingkan nilai yang tercatat pada log penelusuran pohon keputusan di konsol *browser* (*Developer Tools/F12*) dengan nilai yang tersimpan pada dokumen basis data untuk siklus pengantaran yang sama. Perbandingan dilakukan terhadap data utama, meliputi terminal yang terpilih sebagai prioritas, nilai jarak, waktu tunggu, serta hasil skor atau keputusan yang dihasilkan oleh algoritma. Dengan demikian, pengujian ini dapat menunjukkan kesesuaian data antara proses pengambilan keputusan pada *website* dan data yang direkam pada MongoDB.

Hasil pengujian kemudian digunakan untuk mengetahui konsistensi data yang ditampilkan selama proses pengiriman dengan data yang tersimpan pada basis data. Kesesuaian tersebut menjadi indikator bahwa informasi hasil proses pengiriman dapat diteruskan dan direkam tanpa perubahan nilai pada setiap siklus pengantaran. Selain itu, pengujian ini juga memastikan bahwa data yang tersimpan tetap merepresentasikan proses pengambilan keputusan yang

berlangsung pada *website*. Dokumentasi perbandingan data hasil penelusuran dengan data yang tersimpan pada MongoDB disajikan pada Gambar 4.23.

Gambar 4. 23 Sinkronisasi Website dan Database

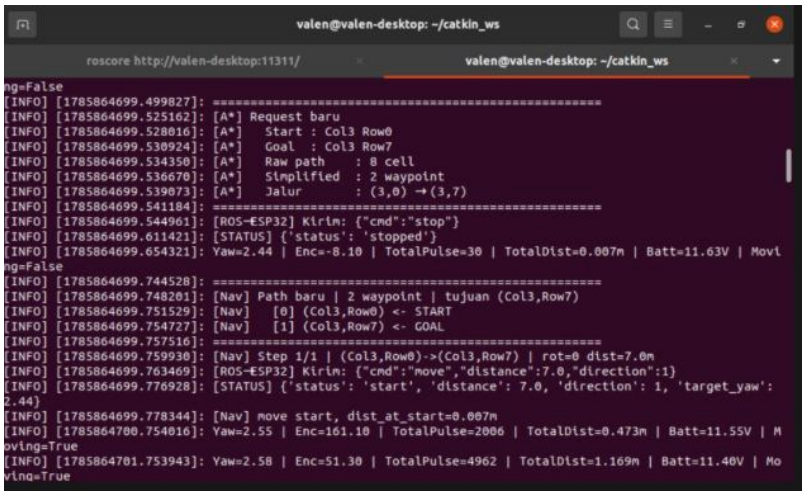
Tabel 4. 23 Pencocokan Data Website dan Database

Parameter	Data pada Website (Log F12)	Data pada Database (MongoDB)
Posisi Awal Robot	Col 3, Row 0	robotStartCol: 3, robotStartRow: 0
Jumlah Request	3 (A, B, C)	`requests`: 3 entri (A, B, C)
Jarak A / B / C	9 / 7 / 9	`distance`: 9 / 7 / 9
Waktu Tunggu A / B / C (ms)	2131 / 1228 / 1	`waitMs`: 2131 / 1228 / 1
Atribut Penentu	Jarak (terkecil = 7)	`reason`: "JARAK terkecil (7)"
Aturan Keputusan (Rule)	"Tie P2 -> pohon C4.5: JARAK terkecil (7) -> pilih B"	`reason` (identik)
Pemenan	B	`winner: "B"`
Terminal Pickup	B	`terminalPickup: "B"`, `pickupCol:3`, `pickupRow:7`
Waktu Mulai	—	04/08/2026 17:37:59
Durasi	—	86 detik
Status	—	`status: "success"`

Berdasarkan Gambar 4.18 dan Tabel 4.19, seluruh parameter yang tercatat pada log penelusuran pohon keputusan di *website*, mulai dari posisi awal robot, jarak, waktu tunggu, atribut penentu, hingga aturan keputusan akhir, memiliki nilai yang identik dengan data yang tersimpan pada basis data MongoDB, termasuk kalimat *reason* yang tersimpan sama persis kata demi kata dengan yang tercetak pada konsol *browser*. Kesesuaian ini membuktikan bahwa proses penyimpanan data dari *website* ke basis data berjalan tanpa mengalami kehilangan maupun perubahan nilai (*data loss* atau *data corruption*), sehingga sinkronisasi antara *website* dan basis data dinyatakan berhasil. Selain itu, siklus pengantaran pada sesi ini tercatat berstatus *success* dengan tahap akhir *done*, menunjukkan bahwa robot berhasil menuntaskan seluruh proses pengantaran tanpa mengalami interupsi.

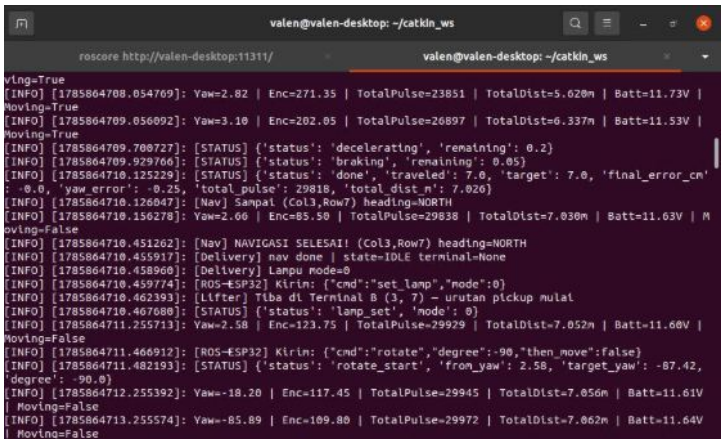
4.4.2 Sinkronisasi Website dan Robot

Pengujian sinkronisasi *website* dan robot dilakukan untuk memverifikasi bahwa komunikasi antara antarmuka web dan robot berjalan dua arah secara *real-time* melalui protokol *ROSBridge WebSocket*, yaitu perintah navigasi yang dikirim dari *website* benar-benar diterima dan dieksekusi oleh robot, serta data sensor yang dihasilkan robot benar-benar diterima kembali oleh *website* tanpa mengalami kehilangan data. Pengujian ini menggunakan sesi yang sama dengan pengujian sinkronisasi *website* dan basis data pada subbab 4.4.1, yaitu pengantaran menuju Terminal B dengan posisi awal robot pada Col 3, Row 0.



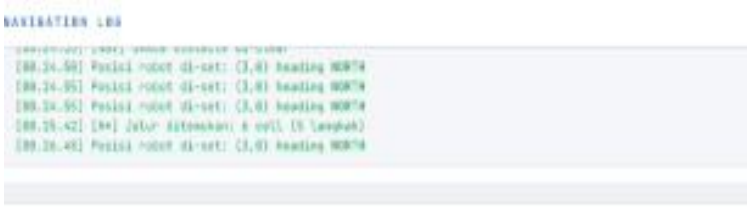
Gambar 4. 24 Log pada Robot

Berdasarkan hasil pengamatan pada terminal roslaunch di Jetson (Gambar 4.23), robot menerima permintaan navigasi dari *website* dengan titik awal (*start*) pada Col 3, Row 0 dan titik tujuan (*goal*) pada Col 3, Row 7, yang merupakan koordinat Terminal B. Algoritma A* pada sisi robot kemudian menghitung lintasan sepanjang 8 sel dan menyederhanakannya menjadi 2 *waypoint*, yaitu (3,0) → (3,7). Perintah pergerakan ini diteruskan dari ROS menuju mikrokontroler ESP32 melalui log [ROS→ESP32] Kirim: {"cmd":"move","distance":7.0,"direction":1}, yang membuktikan bahwa perintah navigasi dari *website* diterima dan diproses oleh robot secara langsung.



Gambar 4. 25 Sinkronasi Data pada Website

Selama proses pergerakan berlangsung, robot secara kontinu mempublikasikan data sensornya, meliputi sudut *yaw*, nilai *encoder*, total pulsa, total jarak tempuh, serta tegangan baterai, sebagaimana ditunjukkan pada Gambar 4.24. Data tersebut tercatat meningkat secara konsisten seiring pergerakan robot, misalnya total jarak tempuh yang bertambah dari 0,001 m pada awal pergerakan hingga mencapai 7,030 m saat robot tiba di titik tujuan. Proses pergerakan diakhiri dengan tahapan deselerasi dan pengereman otomatis, yang ditandai dengan log [STATUS] {'status': 'done', 'traveled': 7.0, 'target': 7.0, 'final_error_cm': -0.0}, yang menunjukkan bahwa robot berhasil mencapai titik tujuan dengan galat jarak yang sangat kecil. Robot kemudian mengonfirmasi ketibaannya melalui log [Nav] NAVIGASI SELESAI! (Col3,Row7) heading=NORTH, dan secara otomatis melanjutkan ke tahap berikutnya, yaitu penyalaan lampu indikator dan rotasi menuju posisi pengambilan barang di Terminal B.



Gambar 4. 26 Sinkronasi Data pada Website

Untuk memverifikasi bahwa data yang sama juga diterima oleh *website*, dilakukan pencocokan antara log pada terminal roslaunch dengan tampilan pada antarmuka web, sebagaimana ditunjukkan pada Gambar 4.26. Hasil pencocokan ditunjukkan pada Tabel 4.24.

Tabel 4. 24 Pencocokan Data Website dan Robot (Sesi Terminal B)

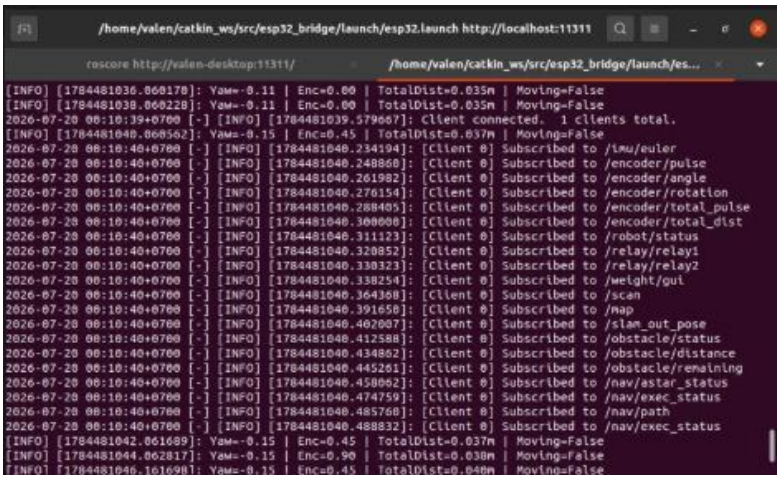
Parameter	**Data pada Robot (*roslaunch*)**	**Data pada Database *Website***
Titik Awal (Start)	Col 3, Row 0	Col 3, Row 0
Titik Tujuan (Goal)	Col 3, Row 7	Col 3, Row 7
Jarak Tempuh	7,030 m (final_error_cm: -0,0)	7,03 m
Yaw saat tiba	2,66	2,66°
Tegangan Baterai	11,63 V	11,63 V
Status navigasi	"NAVIGASI SELESAI!"	Success / NAVIGASI SELESAI

Berdasarkan log pada terminal roslaunch, terlihat bahwa seluruh tahapan komunikasi, mulai dari penerimaan perintah navigasi, perhitungan lintasan menggunakan algoritma A*, pengiriman perintah pergerakan ke ESP32, hingga publikasi data sensor secara kontinu, berjalan tanpa hambatan dan tercatat secara lengkap. Hasil pencocokan pada Tabel 4.20 menunjukkan bahwa parameter yang tercatat pada sisi robot memiliki nilai yang sesuai dengan yang ditampilkan pada antarmuka *website*, sehingga membuktikan bahwa sinkronisasi data antara *website* dan robot melalui protokol *ROSBridge WebSocket* berjalan dengan baik, tanpa mengalami kehilangan maupun keterlambatan data yang signifikan.

4.4.3 Pengujian Sinkronisasi Protokol Komunikasi Rosbridge_Websocket

PengujianPengujian sinkronisasi protokol komunikasi *ROSBridge WebSocket* dilakukan untuk memverifikasi bahwa koneksi antara antarmuka

website dan robot melalui Jetson dapat terbentuk dengan stabil, serta mampu menangani pertukaran berbagai jenis data secara bersamaan dalam satu koneksi *WebSocket* tunggal. Pengujian ini penting dilakukan mengingat pada tahap pengembangan awal sistem sempat ditemukan permasalahan akumulasi *client* akibat duplikasi *subscription* topik ROS, yang menyebabkan penurunan performa (*lag*) pada antarmuka web. Pengujian ini bertujuan untuk membuktikan bahwa permasalahan tersebut telah teratasi pada versi sistem saat ini.



Gambar 4. 27 Rosbridge_Websocket pada Robot

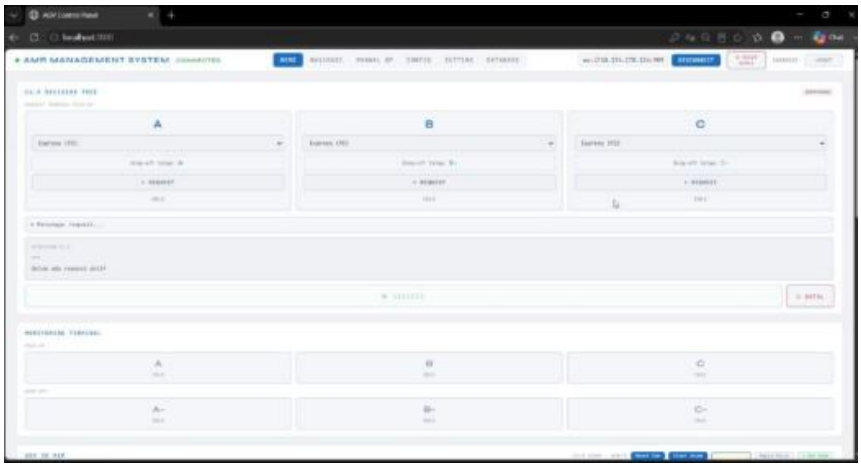
Berdasarkan log pada terminal esp32_bridge Gambar 4.27, koneksi *WebSocket* dari *browser* berhasil terbentuk dan tercatat dengan pesan Client connected. 1 clients total., yang menunjukkan bahwa hanya terdapat satu koneksi *client* aktif untuk satu sesi *browser* yang terhubung. Hal ini membuktikan bahwa permasalahan akumulasi *client* akibat duplikasi *subscription*, yang sebelumnya ditemukan pada tahap pengembangan, telah teratasi pada implementasi sistem saat ini.

Selanjutnya, log yang sama menunjukkan bahwa *client* tersebut berhasil melakukan *subscribe* terhadap seluruh topik yang dibutuhkan sistem secara bersamaan dalam satu koneksi, meliputi data sensor (/imu/euler, /encoder/pulse, /encoder/angle, /encoder/rotation, /encoder/total_pulse, /encoder/total_dist), status robot dan relay (/robot/status, /relay/relay1, /relay/relay2, /weight/gui), data pemetaan dan navigasi (/scan, /map, /slam_out_pose, /obstacle/status, /obstacle/distance, /obstacle/remaining, /nav/astar_status, /nav/exec_status, /nav/path), sebagaimana dirangkum pada Tabel 4.25. Keberhasilan *subscribe*

terhadap seluruh topik ini menunjukkan bahwa protokol *ROSBridge WebSocket* mampu menangani pertukaran data multi-kanal secara simultan tanpa mengalami kegagalan koneksi.

Tabel 4. 25 Daftar Topik ROS yang Disinkronkan melalui ROSBridge WebSocket

Kategori	Topik	Keterangan
Sensor Inersial	`/imu/euler`	Data orientasi (roll, pitch, yaw)
Sensor Encoder	`/encoder/pulse`, `/encoder/angle`, `/encoder/rotation`, `/encoder/total_pulse`, `/encoder/total_dist`	Data pergerakan dan jarak tempuh
Status Robot	`/robot/status`, `/relay/relay1`, `/relay/relay2`,`/gui`	Status operasional dan aktuator
Pemetaan	`/scan`, `/map`, `/slam_out_pose`	Data LIDAR dan SLAM
Navigasi	`/obstacle/status`, `/obstacle/distance`, `/obstacle/remaining`, `/nav/astar_status`, `/nav/exec_status`, `/nav/path`	Status algoritma A* dan navigas



Gambar 4. 28 Rosbridge_Websocket pada Website

Untuk memverifikasi keberhasilan koneksi dari sisi antarmuka, dilakukan pengamatan pada *website* yang menunjukkan status koneksi Gambar 4.28. Antarmuka *website* menampilkan indikator "CONNECTED" berwarna hijau beserta alamat *WebSocket* ws://10.174.170.134:9090, yang sesuai dengan alamat Jetson yang menjalankan *node* esp32_bridge. Kesesuaian status koneksi pada kedua sisi ini membuktikan bahwa protokol komunikasi *ROSBridge WebSocket*

berhasil menjembatani antarmuka *website* dan robot secara stabil.

Hasil pengujian menunjukkan bahwa protokol *ROSBridge WebSocket* pada sistem mampu membentuk koneksi tunggal yang stabil (*single client per session*) dan menangani *subscribe* terhadap 17 topik ROS secara bersamaan tanpa mengalami kegagalan, sebagaimana dibuktikan pada Tabel 4.25. Selain itu, indikator status koneksi pada antarmuka *website* konsisten dengan status koneksi pada sisi Jetson, sehingga sinkronisasi protokol komunikasi antara *website* dan robot dinyatakan berhasil. Keberhasilan pengujian ini juga mengonfirmasi bahwa permasalahan akumulasi *client* akibat duplikasi *subscription* topik yang sebelumnya ditemukan pada tahap pengembangan awal sistem telah teratasi sepenuhnya.

4.5 Analisis Hasil Pengujian

Berdasarkan hasil pengujian yang telah diuraikan pada subbab 4.3 dan 4.4, sistem yang dikembangkan menunjukkan kinerja yang sesuai dengan tujuan perancangan, baik pada aspek pengambilan keputusan algoritma *Decision Tree* C4.5 maupun pada aspek integrasi antarkomponen sistem. Analisis berikut disusun untuk mengaitkan temuan pada kedua aspek tersebut secara menyeluruh.

Pada aspek algoritma pengambilan keputusan, hasil pengujian pada tiga kondisi (4.3.2–4.3.4) menunjukkan bahwa struktur pohon keputusan yang disusun berdasarkan urutan atribut jenis barang, jarak, dan waktu tunggu berhasil menentukan terminal prioritas secara tepat pada seluruh kondisi pengujian, yaitu 3 dari 3 kondisi (100%). Urutan atribut tersebut terbukti secara matematis melalui perhitungan *Gain Ratio* pada subbab 4.3.5.1, dengan nilai *Gain Ratio* jenis barang dan jarak masing-masing mencapai 1,000, jauh lebih tinggi dibandingkan waktu tunggu yang hanya mencapai 0,274. Hal ini menjelaskan mengapa robot secara konsisten memprioritaskan jenis barang terlebih dahulu, kemudian jarak, dan baru mempertimbangkan waktu tunggu apabila kedua atribut sebelumnya bernilai identik. Temuan pada Kondisi 3 menjadi bukti penting bahwa sistem tidak mengalami kebuntuan (*deadlock*) sekalipun dihadapkan pada kondisi ekstrem, yaitu ketika dua atribut pertama tidak dapat membedakan prioritas, karena atribut

waktu tunggu tetap mampu menghasilkan keputusan akhir yang pasti sekaligus mencegah kondisi *starvation* pada permintaan yang diajukan lebih awal.

Pada aspek integrasi sistem, hasil pengujian pada subbab 4.4.1 hingga 4.4.3 menunjukkan bahwa data yang dihasilkan pada satu komponen dapat diterima dan tersimpan secara konsisten pada komponen lainnya. Pencocokan pada 8 parameter data antara antarmuka *website* dan basis data MongoDB menunjukkan tingkat kesesuaian sebesar 100%, termasuk pada parameter yang bersifat deskriptif seperti kalimat aturan keputusan (*rule*), yang tersimpan sama persis antara log konsol *browser* dan dokumen basis data. Pada aspek komunikasi antara *website* dan robot, log terminal `roslaunch` menunjukkan bahwa perintah navigasi dari *website* diterima dan dieksekusi oleh robot dengan galat jarak navigasi sebesar -0,0 cm, sementara pengujian protokol *ROSBridge WebSocket* menunjukkan bahwa satu koneksi *WebSocket* tunggal mampu menyinkronkan 17 topik ROS secara bersamaan dengan status *"1 clients total"*, yang membuktikan bahwa permasalahan duplikasi *client* pada tahap pengembangan awal sistem telah teratasi.

Durasi penyelesaian satu siklus pengantaran penuh, mulai dari pengambilan keputusan hingga barang sampai di terminal tujuan, tercatat bervariasi antara 86 hingga 196 detik. Variasi durasi ini dipengaruhi oleh jarak tempuh robot menuju terminal yang dipilih serta jumlah permintaan yang diproses secara berurutan dalam satu sesi pengujian, sehingga durasi yang lebih lama pada beberapa percobaan bukan merupakan indikasi kegagalan sistem, melainkan konsekuensi wajar dari perbedaan jarak dan kompleksitas permintaan yang ditangani.

Secara keseluruhan, hasil pengujian pada Bab IV membuktikan bahwa ketiga rumusan masalah penelitian telah terjawab secara kuantitatif: sistem WMS berbasis web berhasil terintegrasi dengan AMR dengan tingkat sinkronisasi data 100%, monitoring kondisi robot secara *real-time* berhasil ditampilkan dengan galat posisi mendekati nol, dan *decision tree* berbasis aturan berhasil digunakan sebagai mekanisme pengambilan keputusan prioritas pengantaran dengan tingkat akurasi 100% pada seluruh kondisi pengujian.

BAB 5
KESIMPULAN DAN SARAN

Bab ini merupakan tahapan akhir dari Tugas Akhir yang berisi kesimpulan dari hasil penelitian yang telah dilakukan serta saran untuk pengembangan penelitian berikutnya berdasarkan topik Tugas Akhir ini. Kesimpulan yang disajikan merangkum temuan utama dan implikasi dari penelitian ini, sementara saran yang diberikan bertujuan untuk memberikan panduan bagi peneliti selanjutnya yang tertarik untuk melanjutkan atau mengembangkan topik penelitian ini lebih lanjut.

5.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan pengujian sistem Warehouse Management System (WMS) berbasis web yang terintegrasi dengan Autonomous Mobile Robot (AMR), maka dapat diperoleh beberapa kesimpulan sebagai berikut:

1. Sistem *Warehouse Management System* (WMS) berbasis web yang terintegrasi dengan *Autonomous Mobile Robot* (AMR) telah berhasil dirancang dan diimplementasikan dengan mengintegrasikan tiga lapisan sistem, yaitu antarmuka web, basis data MongoDB, dan robot berbasis ROS1 pada Jetson, yang saling berkomunikasi melalui protokol *ROSBridge WebSocket*. Hasil pengujian integrasi sistem menunjukkan bahwa seluruh 8 parameter data yang dibandingkan antara antarmuka web dan basis data (meliputi posisi robot, jarak, waktu tunggu, atribut penentu, aturan keputusan, dan terminal pemenang) tersinkronisasi dengan tingkat kesesuaian 100%, serta seluruh 17 topik ROS berhasil disinkronkan dalam satu koneksi *WebSocket* tunggal tanpa terjadi duplikasi *client*. Waktu penyelesaian satu siklus pengantaran penuh (dari permintaan hingga barang sampai di terminal tujuan) tercatat bervariasi antara 86 hingga 196 detik, tergantung pada jarak tempuh dan jumlah permintaan yang diproses.
2. Monitoring kondisi robot secara *real-time* berhasil ditampilkan pada antarmuka web dalam bentuk status pengantaran dan peta lokasi robot tiga

dimensi (*3D Maps*), dengan data sensor robot (sudut *yaw*, *encoder*, jarak tempuh, dan tegangan baterai) yang diperbarui secara kontinu melalui *throttle* antara 100–500 milidetik per topik data. Hasil pengujian sinkronisasi menunjukkan bahwa nilai yang ditampilkan pada antarmuka web memiliki kesesuaian penuh dengan data yang dipublikasikan langsung oleh robot, dengan galat posisi navigasi tercatat sebesar -0,0 cm pada pengujian ketibaan robot di titik tujuan, membuktikan keakuratan representasi posisi robot antara sistem visual dan kondisi fisik sesungguhnya.

3. *Decision tree* berbasis aturan berhasil diterapkan sebagai mekanisme pengambilan keputusan prioritas pengantaran barang, dengan struktur pohon tiga tingkat yang disusun berdasarkan urutan atribut jenis barang, jarak, dan waktu tunggu. Urutan atribut ini dibuktikan secara matematis melalui perhitungan *Gain Ratio*, dengan nilai *Gain Ratio* jenis barang dan jarak masing-masing mencapai 1,000, sedangkan waktu tunggu sebesar 0,274, yang menunjukkan bahwa jenis barang dan jarak memiliki daya pisah lebih tinggi dibandingkan waktu tunggu. Hasil pengujian pada tiga kondisi (jenis barang berbeda, jenis barang sama dengan jarak berbeda, dan jenis barang sama dengan jarak sama) menunjukkan bahwa sistem berhasil menentukan terminal prioritas secara tepat pada 3 dari 3 kondisi pengujian (100%), termasuk pada kondisi ekstrem ketika dua atribut pertama bernilai identik, sistem tetap mampu menghasilkan keputusan yang konsisten melalui atribut waktu tunggu tanpa mengalami kebuntuan (*deadlock*).

5.2 Saran

Dari hasil percobaan yang telah dilakukan, terdapat beberapa hal yang harus ditingkatkan untuk memaksimalkan penelitian selanjutnya adalah sebagai berikut:

1. Perlu ditambahkan sensor koreksi posisi tambahan untuk mengatasi permasalahan selip roda (*wheel slip*) yang memengaruhi akurasi sistem *dead-reckoning*, seperti penggunaan sensor *optical flow*, *encoder* ganda untuk deteksi selisih putaran roda kiri-kanan, atau integrasi sensor posisi absolut

(misalnya UWB atau *marker-based positioning*) sebagai pembanding sekaligus koreksi berkala terhadap estimasi posisi berbasis *encoder* dan IMU.

2. Perlu dikembangkan mekanisme *watchdog* pada sisi *backend* ROS (bukan hanya pada sisi antarmuka web) yang mampu mendeteksi dan memulihkan kegagalan *node*, khususnya pada modul deteksi rintangan (*obstacle monitor*) yang selama pengujian menunjukkan performa tidak konsisten dalam durasi operasional panjang, sehingga tidak lagi memerlukan intervensi manual berupa *restart* penuh sistem ROS (roslaunch) setiap kali terjadi kegagalan.
3. Perlu dilakukan investigasi lebih lanjut serta perbaikan pada rangkaian pemicu (*trigger*) modul *relay* yang mengendalikan aktuator *linear* pada *lifter*, termasuk pengukuran arus aktual aktuator saat kondisi awal penggerakan (*start-up current*), penambahan komponen pelindung seperti dioda *flyback* atau *driver* tambahan, maupun penggantian modul *relay* dengan spesifikasi arus yang lebih sesuai dengan karakteristik beban aktuator, agar mekanisme *lifter* dapat berfungsi secara otomatis tanpa memerlukan pemicu manual.

(Halaman ini sengaja dikosongkan)

DAFTAR PUSTAKA

Agus, I. P., Pratama, E., & Sunia, I. M. (2023). *Node . js Performance Benchmarking and Analysis at Virtualbox , Docker , and Podman Environment Using Node-Bench Method*. *7*(December), 2240–2246.

Alif, R., Putra, D., & Rosyani, P. (2022). *Perancangan Aplikasi WMS (Warehouse Management System) untuk Efisiensi Penyimpanan Stock Barang pada Warehouse dengan Metode FIFO (First In First Out)*. *7*(4), 2622–4615. http://openjournal.unpam.ac.id/index.php/informatika

Devega, M., Yuhelmi, Y., & Darmayunata, Y. (2024). Pembangunan Sistem Inventori Apotek Menggunakan Metode Fifo Dan Fefo. *ZONAsi: Jurnal Sistem Informasi*, *6*(1), 159–172. https://doi.org/10.31849/zn.v6i1.17318

Guo, J., Lin, T., & Hsia, K. (2022). *Web-based IoT and Robot SCADA using MQTT protocol*. *9*(2), 202–208.

Hairun, A. N., Katili, M. R., Takdir, R., & Tuloli, M. S. (2023). *Penerapan firewall di router OS mikrotik pada aplikasi e-rapor*. *5*(2), 108–119. https://doi.org/10.37905/jji.

Jaka Indra, . K., Abdillah, M. F., Hidayat, N., & Prastyo, Y. (2025). Literature Review: Comparative Analysis of FIFO and FEFO Management Methods. *Engineering and Technology Journal*, *10*(06), 5438–5443. https://doi.org/10.47191/etj/v10i06.16

Keith, R., & La, H. M. (2024). *Review of Autonomous Mobile Robots for the Warehouse Environment*. 1–25. http://arxiv.org/abs/2406.08333

Madurapperuma, I. H., Shafana, M. S., & Sabani, M. J. A. (2022). *State-of-Art Frameworks for Front-end and Back-end Web Development*. *Icst*, 62–67.

Nambiar, A., & Mundra, D. (2022). An Overview of Data Warehouse and Data Lake in Modern Enterprise Data Management. *Big Data and Cognitive Computing*, *6*(4). https://doi.org/10.3390/bdcc6040132

Nazifah, N., & Prianto, C. (2023). *Analisis Perbandingan Decision Tree Algoritma C4 . 5 dengan algoritma lainnya : Sistematic Literature Review*. *04*(02), 57–64.

Nursyeha, M. A., Irwansyah, F. R., Saputra, R. H., & Aprillia, H. (2023).

SPECTA Journal of Technology. *7*(2), 556–565. https://doi.org/10.35718/specta.v7i2.784

Qudus, M. R. N., & Amelia, N. S. (2022). *International Journal Administration, Business &Organization*. *6*(December), 378–393. https://ijabo.a3i.or.id/index.php/ijabo/article/download/207/63

Rakhman, E., Basjaruddin, N. C., Eka, V., & Susanto, P. (n.d.). *Robot Mobile Otonom Menggunakan Metode Odometry*. 105–116.

Rana, A. (2023). An Analysis of Warehouse Management Systems. *International Journal for Research in Applied Science and Engineering Technology*, *11*(6), 1154–1157. https://doi.org/10.22214/ijraset.2023.53808

Ridho, M. F., Abidin, A. Z., & Septian, B. (2024). Integrasi Odometri LiDAR dan Sensor IMU untuk Peningkatan Lokalisasi pada Robot Bergerak Indoor. *Blend Sains Jurnal Teknik*, *2*(4), 287–297. https://doi.org/10.56211/blendsains.v2i4.470

Sari, A. S., & Hidayat, R. (2022). *Designing website vaccine booking system using golang programming language and framework react JS*. *6*(1), 22–39. https://doi.org/10.52362/jisicom.v6i1.760

Shcherbakov, M., Balliu, M., & Staicu, C. (n.d.). *Silent Spring : Prototype Pollution Leads to Remote Code Execution in Node . js*.

Studies, T. (2024). *MongoDB and Data Consistency: Bridging the Gap between Performance and Reliability*. 183–198. https://doi.org/10.32996/jcsts

Zhang, Y., & Abellera, J. U. (2025). Autonomous mobile robotics in smart warehousing: a cyber-physical systems approach to inventory management. *Future Technology*, *4*(4), 59–71. https://doi.org/10.55670/fpll.futech.4.4.6

Ziegler, S., Woodward, R. C., Iu, H. H. C., & Borle, L. J. (2009). Current sensing techniques: A review. *IEEE Sensors Journal*, *9*(4), 354–376. https://doi.org/10.1109/JSEN.2009.2013914

LAMPIRAN

Lampiran 1. Data Pengujian

A. Skenario 1

- Percobaan 2

Posisi Awal Robot
Home Base (0.0)

Tabel Parameter Pengujian Kondisi 1

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Expired	P1	9	1136
B	Express	P2	7	586
C	Regular	P3	9	1

Hasil Keputusan Kondisi 1

Terminal Terpilih	Prioritas Pemenang	Aturan (Rule) Keputusan
A	P1 (Expired)	IF jenis barang = Expired (P1) THEN pilih Terminal A

- Percobaan 3

Tabel Parameter Pengujian Kondisi 1

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Regular	P3	8	1048
B	Express	P2	10	546
C	Expired	P1	12	2

Hasil Keputusan Kondisi 1

Terminal Terpilih	Prioritas Pemenang	Aturan (Rule) Keputusan
C	P1 (Expired)	IF jenis barang = Expired (P1) THEN pilih Terminal C

- Percobaan 4

Tabel Parameter Pengujian Kondisi 1

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Expired	P1	12	3115
B	Reguler	P3	9	1
C	Express	P2	8	1645

Hasil Keputusan Kondisi 1

Terminal Terpilih	Prioritas Pemenang	Aturan (Rule) Keputusan
A	P1 (Expired)	IF jenis barang = Expired (P1) THEN pilih Terminal A

- Percobaan 5

Tabel Parameter Pengujian Kondisi 1

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Regular	P3	8	1
B	Express	P2	7	1845
C	Expired	P1	11	2790

Hasil Keputusan Kondisi 1

Terminal Terpilih	Prioritas Pemenang	Aturan (Rule) Keputusan
C	P1 (Expired)	IF jenis barang = Expired (P1) THEN pilih Terminal C

- Percobaan 6

Tabel Parameter Pengujian Kondisi 1

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Express	P2	6	1760
B	Expired	P1	10	3370
C	Regular	P3	12	1

Hasil Keputusan Kondisi 1

Terminal Terpilih	Prioritas Pemenang	Aturan (Rule) Keputusan
B	P1 (Expired)	IF jenis barang = Expired (P1) THEN pilih Terminal B

- Percobaan 7

Tabel Parameter Pengujian Kondisi 1

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Expired	P1	9	2950
B	Express	P2	11	1650
C	Regular	P3	13	1

Hasil Keputusan Kondisi 1

Terminal Terpilih	Prioritas Pemenang	Aturan (Rule) Keputusan
A	P1 (Expired)	IF jenis barang = Expired (P1) THEN pilih Terminal A

- Percobaan 8

Tabel Parameter Pengujian Kondisi 1

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Regular	P3	10	1
B	Expired	P1	8	2140
C	Express	P2	12	1505

Hasil Keputusan Kondisi 1

Terminal Terpilih	Prioritas Pemenang	Aturan (Rule) Keputusan
B	P1 (Expired)	IF jenis barang = Expired (P1) THEN pilih Terminal B

- Percobaan 9

Tabel Parameter Pengujian Kondisi 1

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Express	P2	9	1450
B	Regular	P3	11	1
C	Expired	P1	7	3620

Hasil Keputusan Kondisi 1

Terminal Terpilih	Prioritas Pemenang	Aturan (Rule) Keputusan
C	P1 (Expired)	IF jenis barang = Expired (P1) THEN pilih Terminal C

- Percobaan 10

Tabel Parameter Pengujian Kondisi 1

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Expired	P1	11	3185
B	Express	P2	7	1980
C	Regular	P3	10	1

Hasil Keputusan Kondisi 1

Terminal Terpilih	Prioritas Pemenang	Aturan (Rule) Keputusan
A	P1 (Expired)	IF jenis barang = Expired (P1) THEN pilih Terminal A

B. Skenario 2

- **Percobaan 2**

Tabel Parameter Pengujian Kondisi 2

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Regular	P3	9	17300
B	Regular	P3	11	10250
C	Regular	P3	13	1

Tabel Gain Ratio Tie-break

Atribut	Info (A)	Gain	Split Info	Gain Ratio	Bobot
Jarak	0,000	0,901	0,950	0,948	0,74
Waktu Tunggu	0,667	0,329	0,950	0,346	0,26

Tabel Skor Akhir

Terminal	norm Jarak	norm Tunggu	Skor	Peringkat
A	0,750	1,000	0,815	2
B	0,917	0,592	0,832	3
C	1,000	0,000	0,740	1

Tabel Hasil Keputusan Kondisi 2

Terminal Terpilih	Atribut Penentu	Aturan (Rule) Keputusan
C	Jarak (GR = 0,948)	IF prioritas sama (P3) THEN tie-break C4.5 → pilih terminal C

- **Percobaan 3**

Tabel Parameter Pengujian Kondisi 2

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Express	P2	10	15400
B	Express	P2	8	9200
C	Express	P2	12	1

Tabel Gain Ratio Tie-break

Atribut	Info (A)	Gain	Split Info	Gain Ratio	Bobot
Jarak	0,000	0,863	0,918	0,940	0,76
Waktu Tunggu	0,667	0,267	0,918	0,291	0,24

Tabel Skor Akhir

Terminal	norm Jarak	norm Tunggu	Skor	Peringkat
A	0,833	1,000	0,873	3
B	0,667	0,597	0,650	1
C	1,000	0,000	0,760	2

Tabel Hasil Keputusan Kondisi 2

Terminal Terpilih	Atribut Penentu	Aturan (Rule) Keputusan
B	Jarak (GR = 0,940)	IF prioritas sama (P2) THEN tie-break C4.5 → pilih terminal B

- **Percobaan 4**

Tabel Parameter Pengujian Kondisi 2

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Expired	P1	8	11600
B	Expired	P1	10	11400
C	Expired	P1	11	1

Tabel Gain Ratio Tie-break

Atribut	Info (A)	Gain	Split Info	Gain Ratio	Bobot
Jarak	0,000	0,918	0,918	1,000	0,79
Waktu Tunggu	0,667	0,231	0,918	0,252	0,21

Tabel Skor Akhir

Terminal	norm Jarak	norm Tunggu	Skor	Peringkat
A	0,727	1,000	0,783	1
B	0,909	0,613	0,847	3
C	1,000	0,000	0,790	2

Tabel Hasil Keputusan Kondisi 2

Terminal Terpilih	Atribut Penentu	Aturan (Rule) Keputusan
A	Jarak (GR = 1,000)	IF prioritas sama (P1) THEN tie-break C4.5 → pilih terminal A

- **Percobaan 5**

Tabel Parameter Pengujian Kondisi 2

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Regular	P3	12	14300
B	Regular	P3	9	8700
C	Regular	P3	10	1

Tabel Gain Ratio Tie-break

Atribut	Info (A)	Gain	Split Info	Gain Ratio	Bobot
Jarak	0,000	0,874	0,918	0,952	0,77
Waktu Tunggu	0,667	0,247	0,918	0,269	0,23

Tabel Skor Akhir

Terminal	norm Jarak	norm Tunggu	Skor	Peringkat
A	1,000	1,000	0,770	3
B	0,750	0,608	0,717	2
C	0,833	0,000	0,642	1

Tabel Hasil Keputusan Kondisi 2

Terminal Terpilih	Atribut Penentu	Aturan (Rule) Keputusan
C	Jarak (GR = 0,952)	IF prioritas sama (P3) THEN tie-break C4.5 → pilih terminal C

- **Percobaan 6**

Tabel Parameter Pengujian Kondisi 2

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Express	P2	11	16200
B	Express	P2	12	10800
C	Express	P2	8	1

Tabel Gain Ratio Tie-break

Atribut	Info (A)	Gain	Split Info	Gain Ratio	Bobot
Jarak	0,000	0,892	0,918	0,972	0,81
Waktu Tunggu	0,667	0,201	0,918	0,219	0,19

Tabel Skor Akhir

Terminal	norm Jarak	norm Tunggu	Skor	Peringkat
A	0,917	1,000	0,916	2
B	1,000	0,667	0,927	3
C	0,667	0,000	0,540	1

Tabel Hasil Keputusan Kondisi 2

Terminal Terpilih	Atribut Penentu	Aturan (Rule) Keputusan
C	Jarak (GR = 0,972)	IF prioritas sama (P2) THEN tie-break C4.5 → pilih terminal C

- **Percobaan 7**

Tabel Parameter Pengujian Kondisi 2

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Expired	P1	10	12500
B	Expired	P1	12	9800
C	Expired	P1	9	1

Tabel Gain Ratio Tie-break

Atribut	Info (A)	Gain	Split Info	Gain Ratio	Bobot
Jarak	0,000	0,881	0,918	0,960	0,78
Waktu Tunggu	0,667	0,226	0,918	0,246	0,22

Tabel Skor Akhir

Terminal	norm Jarak	norm Tunggu	Skor	Peringkat
A	0,833	1,000	0,825	2
B	1,000	0,784	0,953	3
C	0,750	0,000	0,585	1

Tabel Hasil Keputusan Kondisi 2

Terminal Terpilih	Atribut Penentu	Aturan (Rule) Keputusan
C	Jarak (GR = 0,960)	IF prioritas sama (P2) THEN tie-break C4.5 → pilih terminal C

- **Percobaan 8**

Tabel Parameter Pengujian Kondisi 2

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Regular	P3	8	15100
B	Regular	P3	10	8400
C	Regular	P3	12	1

Tabel Gain Ratio Tie-break

Atribut	Info (A)	Gain	Split Info	Gain Ratio	Bobot
Jarak	0,000	0,905	0,918	0,986	0,80
Waktu Tunggu	0,667	0,211	0,918	0,230	0,20

Tabel Skor Akhir

Terminal	norm Jarak	norm Tunggu	Skor	Peringkat
A	0,667	1,000	0,734	1
B	0,833	0,556	0,778	2
C	1,000	0,000	0,800	3

Tabel Hasil Keputusan Kondisi 2

Terminal Terpilih	Atribut Penentu	Aturan (Rule) Keputusan
A	Jarak (GR = 0,986)	IF prioritas sama (P3) THEN tie-break C4.5 → pilih terminal A

- **Percobaan 9**

Tabel Parameter Pengujian Kondisi 2

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Express	P2	12	13700
B	Express	P2	8	7600
C	Express	P2	10	1

Tabel Gain Ratio Tie-break

Atribut	Info (A)	Gain	Split Info	Gain Ratio	Bobot
Jarak	0,000	0,887	0,918	0,966	0,80
Waktu Tunggu	0,667	0,207	0,918	0,225	0,20

Tabel Skor Akhir

Terminal	norm Jarak	norm Tunggu	Skor	Peringkat
A	1,000	1,000	0,800	3
B	0,667	0,555	0,645	1
C	0,833	0,000	0,666	2

Tabel Hasil Keputusan Kondisi 2

Terminal Terpilih	Atribut Penentu	Aturan (Rule) Keputusan
B	Jarak (GR = 0,966)	IF prioritas sama (P2) THEN tie-break C4.5 → pilih terminal B

- **Percobaan 10**

Tabel Parameter Pengujian Kondisi 2

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)
A	Expired	P1	12	16800
B	Expired	P1	9	11300
C	Expired	P1	11	1

Tabel Gain Ratio Tie-break

Atribut	Info (A)	Gain	Split Info	Gain Ratio	Bobot
Jarak	0,000	0,896	0,918	0,976	0,80
Waktu Tunggu	0,667	0,198	0,918	0,216	0,20

Tabel Skor Akhir

Terminal	norm Jarak	norm Tunggu	Skor	Peringkat
A	1,000	1,000	0,800	3
B	0,750	0,673	0,735	2
C	0,917	0,000	0,734	1

Tabel Hasil Keputusan Kondisi 2

Terminal Terpilih	Atribut Penentu	Aturan (Rule) Keputusan
C	Jarak (GR = 0,976)	IF prioritas sama (P1) THEN tie-break C4.5 → pilih terminal C

C. Skenario 3

- **Percobaan 2**

Tabel Parameter Pengujian Kondisi 3

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)	Keterangan
B	Regular	P3	9	19520	Jarak seri
C	Regular	P3	9	1	Jarak Seri

Tabel Hasil Pengujian Kondisi 3

Terminal Terpilih	Atribut Penentu	Aturan (*Rule*) Keputusan
B	Waktu Tunggu (simpul ke-3, terlama)	IF jenis barang SAMA (P3) AND jarak SERI (9) THEN pilih WAKTU TUNGGU terlama → B

- **Percobaan 3**

Tabel Parameter Pengujian Kondisi 3

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)	Keterangan
A	Express	P2	9	18450	Jarak seri
C	Express	P2	9	1	Jarak Seri

Tabel Hasil Pengujian Kondisi 3

Terminal Terpilih	Atribut Penentu	Aturan (*Rule*) Keputusan
A	Waktu Tunggu (simpul ke-3, terlama)	IF jenis barang SAMA (P2) AND jarak SERI (9) THEN pilih WAKTU TUNGGU terlama → A

- **Percobaan 4**

Tabel Parameter Pengujian Kondisi 3

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)	Keterangan
A	Regular	P3	9	16280	Jarak seri
B	Regular	P3	9	1	Jarak Seri

Tabel Hasil Pengujian Kondisi 3

Terminal Terpilih	Atribut Penentu	Aturan (*Rule*) Keputusan
A	Waktu Tunggu (simpul ke-3, terlama)	IF jenis barang SAMA (P3) AND jarak SERI (9) THEN pilih WAKTU TUNGGU terlama → A

- **Percobaan 5**

Tabel Parameter Pengujian Kondisi 3

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)	Keterangan
B	Expired	P1	9	20150	Jarak seri
C	Expired	P1	9	1	Jarak Seri

Tabel Hasil Pengujian Kondisi 3

Terminal Terpilih	Atribut Penentu	Aturan (*Rule*) Keputusan
B	Waktu Tunggu (simpul ke-3, terlama)	IF jenis barang SAMA (P1) AND jarak SERI (9) THEN pilih WAKTU TUNGGU terlama → B

- **Percobaan 6**

Tabel Parameter Pengujian Kondisi 3

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)	Keterangan
A	Express	P2	9	14870	Jarak seri
B	Express	P2	9	1	Jarak Seri

Tabel Hasil Pengujian Kondisi 3

Terminal Terpilih	Atribut Penentu	Aturan (*Rule*) Keputusan
A	Waktu Tunggu (simpul ke-3, terlama)	IF jenis barang SAMA (P2) AND jarak SERI (9) THEN pilih WAKTU TUNGGU terlama → A

- **Percobaan 7**

Tabel Parameter Pengujian Kondisi 3

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)	Keterangan
B	Regular	P3	9	17640	Jarak seri
C	Regular	P3	9	1	Jarak Seri

Tabel Hasil Pengujian Kondisi 3

Terminal Terpilih	Atribut Penentu	Aturan (*Rule*) Keputusan
B	Waktu Tunggu (simpul ke-3, terlama)	IF jenis barang SAMA (P3) AND jarak SERI (9) THEN pilih WAKTU TUNGGU terlama → B

- **Percobaan 8**

Tabel Parameter Pengujian Kondisi 3

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)	Keterangan
A	Expired	P1	9	21360	Jarak seri
C	Expired	P1	9	1	Jarak Seri

Tabel Hasil Pengujian Kondisi 3

Terminal Terpilih	Atribut Penentu	Aturan (*Rule*) Keputusan
A	Waktu Tunggu (simpul ke-3, terlama)	IF jenis barang SAMA (P1) AND jarak SERI (9) THEN pilih WAKTU TUNGGU terlama → A

- **Percobaan 9**

Tabel Parameter Pengujian Kondisi 3

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)	Keterangan
B	Express	P2	9	19320	Jarak seri
C	Express	P2	9	1	Jarak Seri

Tabel Hasil Pengujian Kondisi 3

Terminal Terpilih	Atribut Penentu	Aturan (*Rule*) Keputusan
B	Waktu Tunggu (simpul ke-3, terlama)	IF jenis barang SAMA (P2) AND jarak SERI (9) THEN pilih WAKTU TUNGGU terlama → B

- **Percobaan 10**

Tabel Parameter Pengujian Kondisi 3

Terminal	Jenis Barang	Prioritas	Jarak (Grid) 1 Grid = 1m	Waktu Tunggu (ms)	Keterangan
A	Regular	P3	9	15780	Jarak seri
C	Regular	P3	9	1	Jarak Seri

Tabel Hasil Pengujian Kondisi 3

Terminal Terpilih	Atribut Penentu	Aturan (*Rule*) Keputusan
A	Waktu Tunggu (simpul ke-3, terlama)	IF jenis barang SAMA (P3) AND jarak SERI (9) THEN pilih WAKTU TUNGGU terlama → A

(Halaman ini sengaja dikosongkan)

Lampiran 2. Biodata Mahasiswa

BIODATA MAHASISWA

1. Nama : Muhammad Fikri Rahman
2. NRP : 0922040064
3. Program Studi : D4 - Teknik Otomasi
4. Agama : Islam
5. Status : Belum Menikah
6. Alamat Asal : Jl. Mutiara 2.8 AM 18, Blok 2B, RT 03 RW 12, Kec. Driyorejo, Kab. Gresik
7. Nomor Telepon : 082257863490
8. Jenis Kelamin : Laki-Laki
9. Email : fikrifirman177@gmail.com
10. Tempat, Tangal Lahir : Surabaya, 17 Oktober 2003
11. Nama Orang Tua/Wali : Nurachman Hadi
12. Alamat Orang Tua/Wali : Jl. Mutiara 2.8 AM 18, Blok 2B, RT 03 RW 12, Kec. Driyorejo, Kab. Gresik
13. Telepon Orang Tua/Wali : 082142908428
14. Riwayat Pendidikan : SMK/ SMA

Pendidikan Formal			
Pendidikan	Tahun	Tempat Pendidikan	Jurusan
Diploma 4	2022 - 2026	Politeknik Perkapalan Negeri Surabaya	Teknik Otomasi
SMK	2019 – 2022	SMKN 1 Driyorejo	TITL
SMP	2016 – 2019	MTsN 2 Surabaya	-
SD	2010 - 2016	SDN 3 Petiken	-