TUGAS AKHIR (EE33410)

RANCANG BANGUN SISTEM *AUTOMARKING* BERBASIS ESP32 DENGAN SENSOR PROXIMITY DAN SENSOR TEGANGAN SEBAGAI UMPAN BALIK PENGENDALI MOTOR DC

Dony Dwi Yulianto
0423030039

Dosen Pembimbing
Purwidi Asri, S.ST., M.T.

PROGRAM STUDI D3 TEKNIK KELISTRIKAN KAPAL
JURUSAN TEKNIK KELISTRIKAN KAPAL
POLITEKNIK PERKAPALAN NEGERI SURABAYA SURABAYA
2026

TUGAS AKHIR (EE33410)

RANCANG BANGUN SISTEM AUTOMARKING BERBASIS ESP32 DENGAN SENSOR PROXIMITY DAN SENSOR TEGANGAN SEBAGAI UMPAN BALIK PENGENDALI MOTOR DC

Dony Dwi Yulianto
0423030039

Dosen Pembimbing
Purwidi Asri, S.ST., M.T.

PROGRAM STUDI D3 TEKNIK KELISTRIKAN KAPAL
JURUSAN TEKNIK KELISTRIKAN KAPAL
POLITEKNIK PERKAPALAN NEGERI SURABAYA SURABAYA
2026

(Halaman ini sengaja dikosongkan)

LEMBAR PENGESAHAN
TUGAS AKHIR

RANCANG BANGUN SISTEM *AUTOMARKING* BERBASIS ESP32 DENGAN SENSOR PROXIMITY DAN SENSOR TEGANGAN SEBAGAI UMPAN BALIK PENGENDALI MOTOR DC

Disusun Oleh:
Dony Dwi Yulianto
0423030039

Diajukan Untuk Memenuhi Salah Satu Syarat Kelulusan
Program Studi D3 Teknik Kelistrikan Kapal
Jurusan Teknik Kelistrikan Kapal
POLITEKNIK PERKAPALAN NEGERI SURABAYA

Disetujui oleh Tim penguji Tugas Akhir Tanggal Ujian : 15 Juli 2026
Periode Wisuda : Oktober 2026

Menyetujui,

Dosen Penguji	NIDN	Tanda Tangan
1. Dwi Sasmita Aji Pambudi, S.T., M.T.	(0021049206)	(.....)
2. Mohammad Basuki Rahmat, S.T., M.T.	(0022057304)	(.....)
3. Purwidi Asri, S.ST., M.T.	(0003097105)	(.....)
4. Thomas Brian, S.ST., M.Kom.	(8648769670130322)	(.....)

Dosen Pembimbing	NIDN	Tanda Tangan
1. Purwidi Asri, S.ST., M.T.	(0003097105)	(.....)

Menyetujui
Ketua Jurusan,

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

Mengetahui
Koordinator Program Studi,

Ii Munadhif, S.ST., M.T.
NIP. 199107102018031001

(Halaman ini sengaja dikosongkan)

PPNS	**PERNYATAAN BEBAS PLAGIAT**	No. : F.WD I. 021 Date : 18 Agustus 2026 Rev. : 01 Page : 1 dari 1

Yang bertandatangan dibawah ini :

Nama : Dony Dwi Yulianto

NRP. : 0423030039

Jurusan/Prodi : Teknik Kelistrikan Kapal/D3 – Teknik Kelistrikan Kapal

Dengan ini menyatakan dengan sesungguhnya bahwa :

Tugas Akhir yang akan saya kerjakan dengan judul :

RANCANG BANGUN SISTEM *AUTOMARKING* BERBASIS ESP32 DENGAN SENSOR PROXIMITY DAN SENSOR TEGANGAN SEBAGAI UMPAN BALIK PENGENDALI MOTOR DC

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, 18 Agustus 2026

Yang membuat pernyataan,

(Dony Dwi Yulianto)

NRP. 0423030039

(Halaman ini sengaja dikosongkan)

KATA PENGANTAR

Puji syukur penulis panjatkan ke hadirat Tuhan Yang Maha Esa atas segala Rahmat, karunia, dan hidayah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir berjudul "Rancang Bangun Sistem Automarking Berbasis ESP32 dengan Sensor Proximity dan Sensor Tegangan Sebagai Umpan Balik Pengendali Motor DC" dengan baik. Tugas Akhir ini disusun sebagai salah satu bentuk persyaratan untuk menyelesaikan Pendidikan pada Program Studi Teknik Kelistrikan Kapal, Program Diploma III, Politeknik Perkapalan Negeri Surabaya. Penulis berharap, melalui penelitian ini agar dapat memberikan manfaat serta kontribusi bagi pengembangan ilmu pengetahuan, khususnya dalam bidang sistem kelistrikan pada kapal.

Dalam proses penyusunan dan penyelesaiian Tugas Akhir ini, penulis menyadari bahwa penelitian ini tidak dapat terselesaikan denfan baik tanpa adanya bantuan, bimbingan, dukungan, doa, serta motivasi dari berbagai pihak. Oleh karena itu, dengan segala kerendahan hati, penulis menyampaikan ucapan terima kasih kepada:

1. Bapak Rachmad Tri Soelistijono, S.T., M.T., selaku Direktur Politeknik Perkapalan Negeri Surabaya.
2. Bapak Isa Rachman, S.T., M.T., selaku Ketua Jurusan Teknik Kelistrikan Kapal Politeknik Perkapalan Negeri Surabaya.
3. Ibu Purwidi Asri, S.ST., M.T. selaku Dosen Pembimbing yang telah meluangkan waktu, tenaga, dan pikiran dalam memberikan bimbingan, arahan, serta masukan selama proses penyusunan Tugas Akhir ini.
4. Bapak Dimas Pristovani Riananda, S.ST., M.T., selaku Koordinator Tugas Akhir yang telah memberikan arahan dalam pelaksanaan Tugas Akhir.
5. Seluruh dosen Program Studi Teknik Kelistrikan Kapal yang telah memberikan ilmu pengetahuan selama masa perkuliahan.

(Halaman ini sengaja dikosongkan)

Rancang Bangun Sistem Automarking Berbasis ESP32 dengan Sensor Proximity dan Sensor Tegangan sebagai Umpan Balik Pengendali Motor DC

Dony Dwi Yulianto

ABSTRAK

Proses *marking* pada industri masih banyak dilakukan secara manual sehingga hasil penandaan sering tidak konsisten dan membutuhkan waktu yang lebih lama. Oleh karena itu, pada penelitian ini dirancang dan dibangun sistem *automarking* berbasis ESP32 yang dapat melakukan proses penandaan secara otomatis. Sistem ini menggunakan sensor *proximity* untuk mendeteksi keberadaan benda logam dan sensor tegangan untuk memantau kondisi sumber tegangan sebagai umpan balik agar sistem bekerja dengan aman dan stabil. Metode penelitian meliputi perancangan perangkat keras dan perangkat lunak, pembuatan alat, pemrograman menggunakan Arduino IDE, serta pengujian setiap komponen dan pengujian sistem secara keseluruhan. Motor DC konveyor dikendalikan menggunakan metode PWM, sedangkan proses penyemprotan dilakukan secara otomatis ketika sensor mendeteksi objek. Hasil pengujian menunjukkan bahwa sensor tegangan memiliki tingkat kesalahan rata-rata sebesar 0,007% sehingga mampu mengukur tegangan dengan baik. Sensor *proximity* mampu mendeteksi objek logam hingga jarak 2 mm secara konsisten. Nilai PWM sebesar 130 menghasilkan kecepatan konveyor yang paling stabil sehingga proses deteksi dan penyemprotan berjalan dengan baik. Sistem juga mampu menghentikan motor, mematikan pompa, dan mengaktifkan buzzer ketika tegangan berada di bawah batas yang ditentukan. Selain itu, data kondisi sistem dapat dipantau melalui web secara *real-time*. Berdasarkan hasil penelitian, dapat disimpulkan bahwa sistem *automarking* berbasis ESP32 dapat bekerja secara otomatis, stabil, dan mampu meningkatkan efisiensi serta konsistensi proses *marking* pada skala prototipe.

Kata kunci: automarking, ESP32, sensor *proximity*, sensor tegangan, motor DC, PWM.

(Halaman ini sengaja dikosongkan)

Design and Construction of an ESP32-Based Automarking System with a Proximity Sensor and Voltage Sensor as Feedback for Controlling a DC Motor

Dony Dwi Yulianto

ABSTRACT

The marking process in industry is still commonly carried out manually, resulting in inconsistent marking quality and lower production efficiency. Therefore, this study aims to design and develop an ESP32-based automarking system that can perform the marking process automatically. The system uses a proximity sensor to detect the presence of metal objects and a voltage sensor to monitor the power supply as feedback to ensure stable and safe system operation. The research method includes hardware and software design, system assembly, programming using Arduino IDE, component testing, and integrated system testing. The conveyor is driven by a DC motor controlled using the Pulse Width Modulation (PWM) method. At the same time, the spray marking process is automatically activated when the proximity sensor detects an object. The test results show that the voltage sensor has an average error of 0.007%, indicating good measurement accuracy. The proximity sensor can detect metal objects consistently at a maximum distance of 2 mm. A PWM value of 130 provides the most stable conveyor speed, allowing the detection and marking processes to operate properly. The protection system is also able to stop the conveyor motor, turn off the spray pump, and activate the buzzer when the supply voltage drops below the specified limit. In addition, the system status can be monitored in real time through a web-based interface. Based on the results, it can be concluded that the proposed ESP32-based automarking system operates automatically, reliably, and consistently, making it suitable for improving the efficiency and quality of the marking process in a prototype-scale application.

Keywords: automarking, ESP32, proximity sensor, voltage sensor, DC motor, PWM.

.

(Halaman ini sengaja dikosongkan)

DAFTAR ISI

DAFTAR TABEL

(Halaman ini sengaja dikosongkan)

DAFTAR GAMBAR

(Halaman ini sengaja dikosongkan)

DAFTAR NOTASI

(Halaman ini sengaja dikosongkan)

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan dunia industri saat ini menuntut proses produksi yang semakin cepat, konsisten, dan efisien. Salah satu proses yang masih sering dijumpai dilakukan secara manual adalah pemberian tanda atau *marking* pada produk. Di beberapa industri, proses marking masih dilakukan dengan cara penyemprotan manual oleh operator, baik menggunakan alat semprot sederhana maupun spray gun. Metode ini memang mudah diterapkan, namun dalam praktiknya sering menimbulkan berbagai permasalahan. Permasalahan utama dari proses marking manual terletak pada ketergantungan terhadap operator. Ketepatan posisi dan waktu penyemprotan sangat bergantung pada fokus dan keterampilan pekerja, sehingga hasil marking sering kali tidak seragam. Selain itu, proses manual membutuhkan waktu lebih lama karena operator harus menghentikan konveyor, memastikan posisi produk, lalu melakukan penyemprotan secara langsung. Kondisi tersebut dapat menurunkan efisiensi produksi, terutama pada lini produksi dengan volume tinggi dan pekerjaan yang berulang. Selain masalah ketelitian dan efisiensi, di lingkungan industri juga sering ditemukan kendala pada sisi kelistrikan. Tegangan suplai yang tidak stabil atau mengalami penurunan sering kali menyebabkan peralatan otomasi tidak bekerja secara optimal. Pada proses marking, kondisi daya yang masuk kurang atau tidak stabil dapat mengakibatkan aktuator penyemprotan tidak bekerja, motor tidak berputar dengan baik, atau sistem berhenti secara tiba-tiba. Permasalahan ini sering kali sulit terdeteksi secara langsung karena tidak adanya sistem monitoring daya yang ditampilkan secara real-time.

Berdasarkan permasalahan tersebut, diperlukan suatu sistem automarking yang tidak hanya mampu mengotomatisasi proses penyemprotan, tetapi juga mampu memantau kondisi daya listrik yang digunakan oleh sistem. Pada penelitian ini, mikrokontroler ESP32 dipilih sebagai pusat kendali karena memiliki kemampuan pemrosesan yang baik, jumlah pin input-output yang

memadai, serta mudah dikembangkan untuk sistem otomasi berbasis prototipe. ESP32 berfungsi mengolah data dari sensor dan mengendalikan seluruh proses kerja sistem automarking secara otomatis. Untuk mendukung kinerja sistem, digunakan sensor proximity sebagai pendeteksi keberadaan produk pada konveyor. Sensor ini memungkinkan sistem mengetahui waktu yang tepat untuk menghentikan konveyor dan menjalankan proses marking tanpa campur tangan operator. Selain itu, sebagai bentuk improvisasi terhadap permasalahan yang sering terjadi di industri, sistem ini juga dilengkapi dengan sensor tegangan. Sensor tegangan digunakan untuk memonitor daya masuk ke sistem secara terus-menerus, kemudian menampilkan informasinya melalui LCD. Dengan adanya tampilan tegangan pada LCD, operator dapat mengetahui secara langsung kondisi suplai daya yang digunakan oleh sistem. Penggunaan sensor tegangan ini menjadi nilai tambah dibandingkan sistem konvensional yang umumnya tidak dilengkapi monitoring daya. Dengan memantau tegangan secara real-time, potensi kegagalan proses automarking akibat daya yang kurang atau tidak stabil dapat diidentifikasi lebih awal. Hal ini diharapkan dapat meningkatkan keandalan sistem serta mengurangi gangguan produksi yang disebabkan oleh masalah kelistrikan. Melalui perancangan sistem automarking berbasis ESP32 yang dilengkapi sensor proximity dan sensor tegangan, diharapkan proses marking dapat berjalan lebih otomatis, konsisten, dan aman. Sistem ini dirancang sebagai solusi atas permasalahan marking manual dan kendala daya listrik yang sering terjadi di industri, sehingga mampu meningkatkan efisiensi dan kualitas proses produksi.

Penggunaan ESP32 pada tugas akhir ini dipilih sebagai alternatif pengendali dibandingkan PLC yang umum digunakan di industri karena sistem yang dirancang berfokus pada pengembangan prototipe dengan fitur improvisasi, salah satunya adalah monitoring tegangan secara real-time menggunakan sensor tegangan. Fitur monitoring ini lebih mudah diintegrasikan pada ESP32 karena ESP32 memiliki input analog internal yang dapat langsung membaca data sensor tegangan tanpa memerlukan modul tambahan seperti pada PLC.

Selain itu, ESP32 memungkinkan proses pemantauan dan pengolahan data tegangan dilakukan secara lebih fleksibel, baik untuk ditampilkan pada LCD maupun dikembangkan lebih lanjut. Pada sistem PLC, pembacaan tegangan umumnya memerlukan modul analog khusus dengan biaya relatif tinggi dan konfigurasi yang lebih kompleks. Oleh karena itu, penggunaan ESP32 dinilai lebih sesuai untuk pengembangan sistem automarking dalam bentuk prototipe yang menekankan aspek monitoring daya sebagai bentuk peningkatan dari sistem konvensional di industri.

1.2 Rumusan Masalah

Berdasarkan latar belakang tersebut, maka rumusan masalah dalam tugas akhir ini adalah:

1. Bagaimana merancang sistem automarking berbasis ESP32 yang bekerja secara otomatis?
2. Bagaimana sensor proximity dan sensor tegangan digunakan sebagai input dalam sistem automarking?
3. Bagaimana mengendalikan konveyor dan proses marking agar berjalan sesuai dengan kondisi sistem?

1.3 Batasan Masalah

Agar pembahasan lebih terfokus, maka batasan masalah dalam tugas akhir ini adalah:

1. Sistem yang dibuat berupa prototipe automarking skala laboratorium.
2. Proses marking dibatasi pada mekanisme aktif-nonaktif (ON/OFF) penyemprotan menggunakan dynamo sprayer dan nozzle tanpa mongondisikan variasi bentuk geometris hasil cat.
3. Sistem dikendalikan menggunakan ESP32 tanpa membahas sistem komunikasi jarak jauh.

1.4 Tujuan Penelitian

Tujuan dari pelaksanaan tugas akhir ini adalah:

1. Merancang dan membangun sistem automarking berbasis ESP32.
2. Menerapkan sensor proximity dan sensor tegangan sebagai pendukung sistem kontrol.
3. Menghasilkan sistem marking otomatis yang lebih efisien dibandingkan metode manual.

1.5 Manfaat Penelitian

Manfaat yang diharapkan dari tugas akhir ini antara lain:

1. Memberikan pemahaman mengenai penerapan otomasi sederhana berbasis mikrokontroler.
2. Menjadi referensi pembelajaran dalam bidang sistem kontrol dan otomasi.
3. Menjadi dasar pengembangan sistem automarking yang lebih lanjut

BAB II
DASAR TEORI

2.1 Kajian Pustaka

2.1.1 Penelitian Terdahulu

Beberapa penelitian yang berkaitan tentang *"Rancang Bangun Sistem Automarking Berbasis ESP32"* sudah pernah dilakukan antara lain, penelitian yang dilakukan oleh Sharma, R., Kumar, P., & Singh, A. (2021) berjudul "*IoT-Based Smart Conveyor System for Integrated Process Control"*. Penelitian ini membahas perancangan sistem konveyor pintar berbasis Internet of Things (IoT) yang digunakan untuk mengendalikan proses industri secara terintegrasi. Sistem yang dikembangkan memanfaatkan mikrokontroler sebagai pusat pengendali, sensor sebagai input sistem, serta aktuator motor sebagai penggerak konveyor. Selain itu, penelitian ini menekankan pentingnya integrasi antar komponen agar proses berjalan secara otomatis dan terkontrol. Jurnal ini relevan dengan tugas akhir yang dilakukan karena memberikan gambaran penerapan sistem otomasi pada konveyor yang dikendalikan oleh mikrokontroler, khususnya dalam pengaturan alur kerja berbasis sensor.

Penelitian yang dilakukan oleh Regina,dkk. (2025) berjudul "Rancang Bangun Sistem Pemantauan Ketersediaan Tempat Parkir Sepeda Motor Berbasis ESP32". Penelitian ini membahas mengenai perancangan sistem pendeteksi ketersediaan tempat parkir menggunakan sensor proximity dan berbasis ESP32. Dengan sensor proximity mendeteksi lalu akan ditampilkan melalui LCD.

Penelitian yang dilakukan oleh Muhammad Gilang Satria Adiguna dkk. (2023) berjudul *"Color and Size Sorting System with ESP32-Based Robotic Arm"*. Penelitian ini merancang sistem sortir otomatis berbasis ESP32 untuk robotic arm 5-DOF yang mendeteksi warna dan ukuran objek pada konveyor menggunakan kamera dan sensor proximity optic, lalu menggerakkan motor DC/servo untuk pick and place marking/sorting dengan akurasi 95% pada

kecepatan 10 objek/menit.

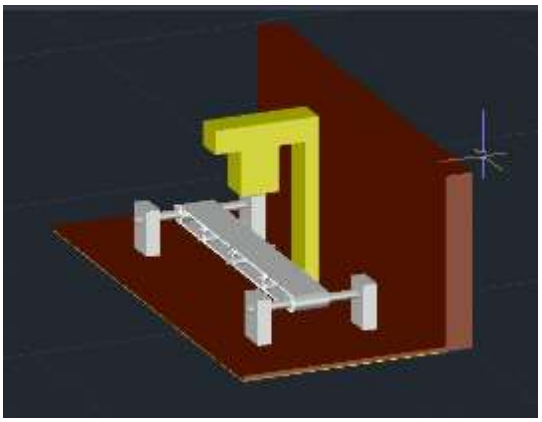
Penelitian yang dilakukan oleh Miranty, Andy Nasriadi (2024) berjudul *"Espduino-32 Utilization in the Warehouse Storage Information System of the Metal Processing Industry Department"* Artikel ini membahas sistem manajemen gudang lokal menggunakan ESP32(Espduino32) yang mengontrol relay untuk menyalakan lampu marker rak secara otomatis saat ada pencarian barang. ESP32 memproses input sensor proximity, mengaktifkan indikator visual (motor relay), LCD menampilkan status rak aktif, dan buzzer konfirmasi selesai.

2.2 Dasar Teori

2.2.1 Plan dari Rancang Bangun Sistem Automarking berbasis ESP32 menggunakan sensor proximity dan sensor tegangan dengan aktuator motor DC

1. Gambar Plan Tampak Atas

   Pada Gambar 2.1, kami membuat plan yang menampilkan desain rangkaian komponen saya , saya tampilkan plan tersebut pada gambar berikut :



Gambar 2. 1 Plan Tampak Atas

Pada tahap perancangan alat, salah satu bagian penting yang perlu ditampilkan dalam laporan adalah gambar plan tampak atas. Gambar ini digunakan untuk memperlihatkan susunan dan posisi setiap komponen ketika alat sudah dirakit secara keseluruhan. Dengan adanya tampak atas,

pembaca dapat memahami dengan jelas bagaimana perangkat seperti ESP32, sensor proximity, motor DC, driver motor, buzzer, LCD I2C, dan konveyor mini ditempatkan dalam satu sistem yang terintegrasi. Gambar plan tampak atas ini dibuat untuk memberikan gambaran visual mengenai tata letak alat dari sudut pandang atas, sehingga semua komponen terlihat dalam satu bidang. Tampilan seperti ini memudahkan proses analisis desain, karena kita dapat melihat apakah jarak antar komponen sudah sesuai, apakah kabel memiliki ruang yang cukup untuk diatur dengan rapi, dan apakah posisi sensor maupun motor memungkinkan alat bekerja tanpa hambatan. Dengan kata lain, gambar tampak atas membantu memastikan bahwa alat tidak hanya berfungsi dengan baik secara elektronik, tetapi juga tertata dengan rapi secara fisik. Dalam pembuatan tampak atas ini, posisi komponen sengaja dirancang agar alur kerja sistem terlihat jelas.

Selain berfungsi sebagai acuan saat perakitan, gambar tampak atas ini juga membantu untuk dokumentasi dan presentasi tugas akhir. Gambarnya dibuat dalam skala yang proporsional sehingga ukuran komponen terlihat realistis dan tidak membingungkan. Hal ini penting agar pembaca laporan dapat membayangkan alat secara utuh meskipun tidak memegang perangkat aslinya.

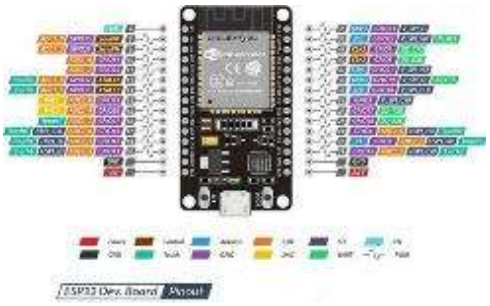
2.2.2 Penjelasan *Hardware* yang digunakan

Pada sistem automarking berbasis ESP32 ini, terdapat beberapa perangkat keras utama yang saling bekerja sama untuk menghasilkan proses penandaan objek secara otomatis. Setiap perangkat memiliki fungsi dan peran yang berbeda, tetapi semuanya saling terhubung dan dikendalikan secara terpusat oleh mikrokontroler. Bagian ini menjelaskan secara rinci setiap komponen hardware yang digunakan, alasan pemilihannya, serta perannya dalam keseluruhan sistem.

1. Mikrokontroller ESP32

Modul ESP32 merupakan komponen utama yang bertugas sebagai otak dari keseluruhan sistem. Pada penelitian ini, ESP32 dipilih karena kemampuannya yang cukup lengkap untuk ukuran mikrokontroler modern, terutama untuk

aplikasi yang membutuhkan koneksi nirkabel, proses data cepat, dan jumlah input - output yang banyak. ESP32 sendiri adalah mikrokontroler berbasis arsitektur dual-core yang dikembangkan oleh Espressif, sehingga mampu menjalankan beberapa proses secara bersamaan tanpa membebani kinerja sistem. Secara umum, ESP32 sudah dilengkapi dengan modul WiFi dan Bluetooth bawaan. Keberadaan dua fitur ini membuat perangkat sangat fleksibel, karena dapat digunakan untuk mengirim data secara real-time ke platform IoT, melakukan monitoring dari jarak jauh, atau sekadar berkomunikasi antarperangkat tanpa memerlukan modul tambahan. Dalam penelitian ini, fitur WiFi dari ESP32 dimanfaatkan untuk mengirimkan data status sensor serta kondisi motor ke sistem monitoring, sehingga operator dapat melihat perkembangan alat tanpa harus berada dekat dengan perangkat. Dari sisi I/O, ESP32 menyediakan cukup banyak pin digital dan analog yang bisa digunakan untuk menghubungkan sensor-sensor maupun aktuator seperti pada Gambar 2.2. Beberapa pin mendukung fungsi PWM (Pulse Width Modulation), sehingga sangat berguna ketika mengendalikan motor DC melalui driver atau inverter mini. Pin ADC pada ESP32 juga dimanfaatkan untuk membaca tegangan, arus, atau sinyal lain yang membutuhkan pengukuran analog. Dengan jumlah pin yang cukup banyak, penelitian ini dapat menempatkan sensor proximity, limit switch, buzzer, indikator LED, serta driver motor dalam satu rangkaian yang tertata tanpa kekurangan port.



Gambar 2. 2 Mikrokontroller ESP32

(Sumber : https://www.ardutech.com/wp-content/uploads/2020/03/4.-ESP32-Pinout.jpg)

Keunggulan lain dari ESP32 adalah kecepatannya dalam memproses data. Mikrokontroler ini bekerja pada kecepatan hingga 240 MHz, sehingga perubahan status sensor dapat direspons hampir seketika. Hal ini penting dalam sistem automasi, terutama pada sistem yang membutuhkan reaksi cepat seperti deteksi objek dan proses penandaan otomatis. Ketika sensor proximity mendeteksi objek, ESP32 dapat langsung mengolah sinyal tersebut dan memberikan perintah ke motor atau aktuator tanpa jeda yang berarti. Dari segi konsumsi daya, ESP32 termasuk cukup efisien. Walaupun memiliki fitur lengkap dan prosesor cepat, konsumsi dayanya masih tergolong rendah. Fitur deep-sleep dan mode hemat daya juga tersedia, meskipun pada penelitian ini fitur tersebut tidak digunakan karena sistem bekerja secara terus-menerus. Namun, jika pada penelitian lanjutan ingin dikembangkan versi portable atau berbasis baterai, kemampuan ini sangat membantu. ESP32 juga mudah diprogram karena mendukung berbagai platform seperti Arduino IDE, PlatformIO, hingga MicroPython. Dalam penelitian ini, pemrograman dilakukan menggunakan Arduino IDE karena sintaksnya sederhana, banyak referensinya, serta mendukung library komunikasi IoT dan kendali motor. Proses pemrograman ESP32 juga mudah karena hanya membutuhkan kabel USB tanpa memerlukan perangkat tambahan seperti downloader eksternal. Selain itu, ESP32 memiliki memori penyimpanan yang cukup besar untuk ukuran mikrokontroler, baik untuk menyimpan program maupun variabel. Hal ini memungkinkan logika kontrol yang digunakan dalam sistem automarking dapat ditulis dengan struktur yang kompleks tanpa khawatir kehabisan ruang memori.

2. Sensor Proximity NPN

Sensor proximity merupakan salah satu komponen utama yang digunakan dalam sistem automarking berbasis ESP32 ini. Sensor ini berfungsi sebagai pendeteksi objek berbahan logam yang melewati area kerja mesin. Pemilihan sensor proximity dilakukan karena sensor jenis ini mampu mendeteksi keberadaan logam tanpa harus melakukan kontak fisik dengan benda tersebut. Hal ini tentu membuat proses deteksi jauh lebih aman, cepat, dan minim kerusakan, terutama pada sistem yang bekerja terus-menerus seperti konveyor atau mesin produksi otomatis. Sensor proximity yang digunakan pada penelitian ini termasuk tipe

induktif, yaitu sensor yang khusus dirancang untuk mendeteksi logam melalui perubahan medan elektromagnetik. Cara kerjanya cukup sederhana namun efektif. Di dalam sensor terdapat kumparan yang menghasilkan medan magnet kecil. Ketika sebuah benda logam melintas di depan sensor, medan magnet tersebut akan berubah. Perubahan inilah yang dibaca oleh rangkaian internal sensor, lalu diubah menjadi sinyal elektrik berupa output HIGH atau LOW. Sinyal ini kemudian diteruskan ke ESP32 untuk diproses lebih lanjut.

Gambar 2. 3 Sensor Proximity

(Sumber : Ubuy Indonesia.com)

Keunggulan sensor proximity induktif seperti pada Gambar 2.3 adalah sifatnya yang tidak terpengaruh oleh cahaya, debu, atau kondisi lingkungan yang kurang ideal. Sensor ini sangat cocok ditempatkan di area industri yang biasanya banyak getaran, serpihan logam, ataupun percikan cairan. Dibandingkan sensor mekanik seperti limit switch, sensor proximity lebih tahan lama karena tidak memiliki bagian yang bergerak. Dengan begitu, perawatan alat menjadi lebih mudah dan umur pakainya lebih panjang. Dalam sistem automarking ini, sensor proximity dipasang pada posisi yang strategis di sisi konveyor atau jalur benda kerja. Ketika benda logam melintas di depannya, sensor akan memberikan sinyal ke ESP32 sebagai penanda bahwa proses marking harus segera dimulai. Output sensor ini menjadi pemicu utama agar motor marking bergerak tepat waktu. Jika tidak ada sinyal dari sensor, maka ESP32 menganggap bahwa konveyor sedang kosong, sehingga motor tidak perlu bekerja dan buzzer dapat diaktifkan untuk

memberikan peringatan kepada operator. Dari sisi kelistrikan, sensor proximity yang digunakan memiliki konfigurasi tiga kabel: VCC, GND, dan kabel output. Sensor jenis NPN atau PNP bisa digunakan, namun pada penelitian ini dipilih tipe NPN karena lebih umum dan mudah diintegrasikan dengan input digital pada ESP32. Tegangan kerja sensor biasanya berkisar antara 6–36 VDC, sehingga diperlukan penyesuaian untuk memastikan sinyal output aman ketika masuk ke pin input mikrokontroler. Pada penelitian ini digunakan rangkaian pembatas tegangan sederhana agar output sensor tetap berada dalam batas yang dapat diterima oleh ESP32. Kecepatan respon sensor proximity juga menjadi alasan kenapa sensor ini cocok digunakan pada sistem. Sensor dapat membaca objek dengan sangat cepat, sehingga meskipun benda logam bergerak cukup cepat di atas konveyor, proses deteksi tetap berlangsung akurat. Ketelitian ini penting karena sistem marking bekerja berdasarkan waktu, dan sedikit keterlambatan dapat menyebabkan marking tidak tepat sasaran. Pemasangan sensor proximity juga dilakukan dengan memperhatikan jarak sensitif (sensing distance). Jarak maksimal yang bisa dideteksi oleh sensor biasanya berkisar 2–8 mm untuk logam besi atau baja. Oleh karena itu, penempatan sensor harus benar-benar sejajar dan cukup dekat dengan benda agar deteksi berlangsung optimal. Selain itu, orientasi sensor juga dijaga agar tidak terkena getaran berlebih yang dapat menyebabkan pembacaan tidak stabil.

3. MOTOR DC

Motor DC merupakan komponen yang memegang peranan penting dalam sistem automarking berbasis ESP32 ini. Motor ini bertugas sebagai aktuator utama yang menghasilkan gerakan mekanis untuk melakukan proses marking pada objek yang lewat di atas konveyor. Tanpa keberadaan motor DC, proses penandaan tidak dapat dilakukan karena alat membutuhkan gerakan fisik yang presisi dan cukup kuat untuk menekan atau membuat tanda pada permukaan benda kerja. Oleh sebab itu, motor DC dipilih sebagai penggerak utama yang dapat diandalkan dalam sistem ini. Dilihat dari cara kerjanya, motor DC memanfaatkan prinsip elektromagnetik yang terjadi ketika arus listrik mengalir ke

dalam kumparan rotor. Arus tersebut menciptakan medan magnet yang berinteraksi dengan magnet permanen pada motor, sehingga rotor berputar. Putaran inilah yang kemudian diteruskan ke mekanisme marking. Salah satu alasan motor DC banyak digunakan pada berbagai sistem otomatisasi skala kecil hingga menengah adalah karena motor ini mampu memberikan respons yang cepat terhadap perubahan sinyal kontrol. Ketika ESP32 mengirimkan perintah untuk bergerak, motor akan langsung berputar tanpa jeda yang berarti, dan ketika sinyal dihentikan, motor juga dapat berhenti dengan cepat sehingga proses marking tetap sesuai waktu dan posisi yang diharapkan.

Gambar 2. 4 Motor DC

(Sumber : Shopee.co.id)

Motor DC juga dapat dikendalikan dengan sangat fleksibel. Kecepatannya dapat diatur menggunakan sinyal PWM yang dikeluarkan oleh ESP32 melalui driver motor. Hal ini sangat membantu proses marking, karena durasi dan kekuatan penekanan bisa disesuaikan sesuai kebutuhan. Pengaturan yang tepat membuat hasil marking tidak terlalu dalam atau terlalu ringan. Kemampuan motor DC untuk menyesuaikan kecepatan secara halus membuatnya lebih unggul dibandingkan motor jenis lain yang membutuhkan pengontrol lebih kompleks. Selain mudah dikendalikan, motor DC seperti pada Gambar 2.4 juga dikenal memiliki torsi yang cukup kuat untuk ukuran fisiknya yang relatif kecil. Ini sangat sesuai dengan kebutuhan pada sistem automarking, di mana mekanisme marking membutuhkan tenaga yang cukup untuk menekan permukaan benda tanpa membuat sistem bekerja terlalu keras. Motor DC memberikan tenaga yang stabil dan tidak mudah mengalami penurunan performa selama digunakan dalam durasi

yang wajar. Dari sisi biaya dan perawatan, motor DC juga tergolong lebih ekonomis dibandingkan motor AC atau motor stepper. Harga motor DC relatif terjangkau, mudah ditemukan di pasaran, dan proses pemasangannya tidak memerlukan rangkaian yang rumit. Hal ini tentu sangat mendukung pengembangan alat berbasis prototype seperti alat automarking ini, yang membutuhkan komponen dengan ketersediaan tinggi namun tetap andal. Motor DC juga mempunyai konstruksi yang sederhana sehingga kerusakan dapat diminimalkan, dan jika terjadi masalah, perbaikannya dapat dilakukan dengan cepat. Untuk menunjang kinerjanya, motor DC pada sistem ini dihubungkan dengan driver motor yang berfungsi mengatur arus serta melindungi ESP32 dari arus balik yang bisa muncul ketika motor berhenti mendadak. Dengan adanya driver, motor dapat dikendalikan secara aman tanpa membebani mikrokontroler. Driver motor juga menjamin bahwa motor mendapatkan suplai arus yang cukup agar tidak mengalami drop tegangan yang dapat mempengaruhi kecepatan putaran. Motor DC yang digunakan pada sistem ini bekerja pada tegangan sekitar 12V, sehingga torsi yang dihasilkan sudah cukup kuat tanpa membuat motor bekerja berlebihan. Secara keseluruhan, motor DC seperti pada Gambar 2.4 memiliki kombinasi karakteristik yang cocok untuk aplikasi automarking seperti respons cepat, kontrol kecepatan yang halus, torsi cukup besar, serta biaya yang ekonomis.

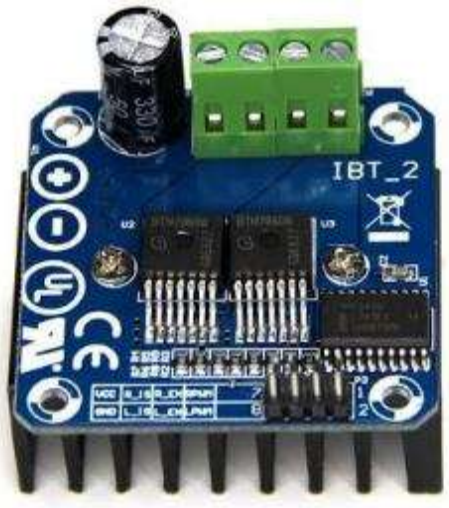
4. Driver Motor

Driver motor BTS7960 merupakan modul pengendali motor DC yang digunakan untuk mengatur arah putaran dan kecepatan motor berdasarkan sinyal yang diberikan oleh mikrokontroler. Pada penelitian ini, modul BTS7960 berfungsi sebagai penghubung antara ESP32 dengan motor DC konveyor. Penggunaan driver ini diperlukan karena pin keluaran ESP32 hanya menghasilkan arus yang kecil, sehingga tidak dapat langsung digunakan untuk menggerakkan motor DC yang membutuhkan arus lebih besar.

Modul BTS7960 bekerja dengan menerima sinyal logika dan sinyal PWM (Pulse Width Modulation) dari ESP32. Sinyal tersebut kemudian diproses oleh

driver untuk mengatur besar tegangan dan arah arus yang diberikan ke motor. Dengan cara ini, kecepatan motor dapat diatur sesuai kebutuhan, sedangkan arah putaran motor dapat diubah apabila diperlukan. Dibandingkan dengan driver motor L298N, BTS7960 memiliki kemampuan menangani arus yang jauh lebih besar sehingga lebih sesuai digunakan pada motor DC yang memiliki beban kerja tinggi. Selain itu, modul ini memiliki efisiensi yang lebih baik karena menggunakan komponen MOSFET sebagai saklar elektronik, sehingga rugi-rugi daya dan panas yang dihasilkan lebih kecil. Kondisi tersebut membuat kerja motor menjadi lebih stabil, terutama ketika digunakan dalam waktu yang lama.

Pada sistem automarking yang dirancang, BTS7960 digunakan untuk mengendalikan motor DC konveyor berdasarkan perintah dari ESP32. Saat sensor proximity mendeteksi adanya benda, ESP32 akan mengirimkan sinyal ke BTS7960 untuk menghentikan putaran motor sehingga proses penyemprotan dapat dilakukan. Setelah proses marking selesai, ESP32 kembali memberikan sinyal PWM kepada BTS7960 sehingga motor berputar kembali dan konveyor melanjutkan proses pengangkutan benda menuju tahap berikutnya.



Gambar 2. 5 Driver Motor

(Sumber : yic-electronics.com)

Selain sebagai penguat arus, driver motor seperti pada Gambar 2.5 juga

memberikan kemampuan kepada sistem untuk mengatur arah putaran motor. Pada mekanisme automarking, meskipun motor biasanya hanya berputar satu arah, kontrol arah ini tetap bermanfaat untuk operasi tertentu seperti pengembalian posisi atau pengaturan ulang alat marking. Driver motor bekerja berdasarkan prinsip rangkaian H-bridge, yaitu konfigurasi empat transistor yang memungkinkan motor diputar maju ataupun mundur hanya melalui perubahan sinyal logika dari ESP32. Sistem ini menawarkan fleksibilitas yang cukup besar dibandingkan menyuplai motor langsung menggunakan sumber DC. Kontrol kecepatan motor juga sangat bergantung pada driver motor. ESP32 mengirimkan sinyal PWM (Pulse Width Modulation) ke pin input driver, kemudian driver mengubah sinyal tersebut menjadi pengaturan kecepatan yang efektif untuk motor DC. Dengan cara ini, operator dapat mengatur seberapa cepat atau seberapa kuat tekanan marking harus dilakukan. Hal ini sangat membantu menjaga konsistensi hasil marking, karena tekanan yang terlalu cepat dapat menyebabkan ketidaktepatan posisi, sedangkan tekanan yang terlalu lambat bisa membuat marking kurang jelas atau tidak presisi. Driver motor yang digunakan dalam sistem ini biasanya sudah dirancang agar kompatibel dengan berbagai tegangan, sehingga mampu memberikan performa yang stabil meskipun motor menarik arus cukup besar saat awal putaran. Kondisi ini dikenal sebagai “arus start” yang merupakan salah satu faktor yang membuat motor DC membutuhkan driver motor yang cukup kuat. Driver yang baik akan memastikan tegangan pada motor tetap stabil sehingga pergerakan motor dimulai dengan halus tanpa tersendat.

Selain itu, penggunaan driver motor membuat sistem automarking menjadi lebih aman secara kelistrikan. Setiap jalur daya dan sinyal pada driver sudah dipisahkan secara fungsional sehingga gangguan pada satu sisi tidak langsung merusak perangkat lain. Cara kerja ini membantu menjaga ESP32 tetap aman selama digunakan dalam jangka waktu lama. Dengan demikian, stabilitas sistem secara keseluruhan dapat dipertahankan meskipun alat digunakan terus-menerus dalam lingkungan yang memiliki tingkat getaran atau beban mekanis yang cukup tinggi. Dalam implementasi fisik, driver motor dipasang berdekatan dengan motor DC agar jalur arus pendek dan minim hambatan. Driver juga diberi sumber

tegangan tersendiri yang biasanya lebih besar daripada tegangan untuk ESP32. Pemisahan sumber tegangan ini diperlukan agar tidak terjadi penurunan daya pada mikrokontroler setiap kali motor aktif. Ketika motor bekerja, terutama pada saat start, konsumsi arusnya meningkat tajam. Jika sumber daya digabungkan, ESP32 bisa mengalami reset mendadak yang akan mengganggu proses marking.

5. Buzzer

Buzzer merupakan salah satu komponen pendukung yang cukup penting dalam sistem automarking berbasis ESP32, khususnya sebagai alat pemberi tanda ketika terjadi kondisi tertentu yang tidak sesuai dengan alur kerja mesin. Dalam penelitian ini, buzzer digunakan untuk memberikan peringatan suara ketika konveyor tidak membawa objek atau ketika sensor proximity tidak mendeteksi logam dalam jangka waktu tertentu. Dengan demikian, buzzer berfungsi sebagai media komunikasi sederhana antara sistem dan operator, agar operator dapat segera mengetahui adanya ketidaksesuaian proses tanpa harus terus-menerus memperhatikan layar atau sistem monitoring. Secara umum, buzzer bekerja dengan cara mengubah sinyal listrik menjadi getaran suara melalui elemen piezoelektrik atau rangkaian internal yang menghasilkan frekuensi tertentu. Ketika ESP32 memberikan sinyal logika kepada buzzer, komponen ini akan menghasilkan bunyi yang cukup jelas untuk didengar oleh operator di sekitar mesin. Hal ini sangat membantu karena suara peringatan dapat langsung menarik perhatian, terutama ketika operator sedang melakukan aktivitas lain atau berada sedikit jauh dari posisi mesin. Peran buzzer dalam sistem automarking bukan sekadar memberikan suara peringatan, tetapi juga membantu menjaga efisiensi proses. Misalnya, ketika tidak ada benda logam yang lewat dalam selang waktu tertentu, sistem menganggap bahwa konveyor sedang kosong atau terjadi hambatan pada aliran benda kerja. Jika kondisi ini tidak cepat diketahui, motor marking bisa saja terus menunggu perintah tanpa melakukan proses apapun, yang pada akhirnya dapat membuang energi dan memperlambat proses produksi. Dengan adanya alarm suara, operator dapat langsung mengecek jalur konveyor dan memperbaiki masalah yang mungkin terjadi.

Gambar 2. 6 Buzzer Aktif

(Sumber : techshopbd.com)

Buzzer seperti pada Gambar 2.6 juga digunakan untuk memberikan tanda bahwa proses marking berjalan normal atau menunjukkan transisi tertentu dalam alur kerja alat. Walaupun fungsinya terlihat sederhana, keberadaan buzzer ternyata sangat membantu dalam memberikan feedback langsung kepada pengguna. Suara dari buzzer memberikan konfirmasi cepat bahwa sistem memang merespons perintah atau mendeteksi kondisi tertentu.

6. LCD I2C

Dalam sistem automarking berbasis ESP32 ini, salah satu komponen yang berperan penting dalam penyampaian informasi kepada operator adalah modul LCD berbasis komunikasi I2C. Penggunaan LCD diperlukan agar sistem dapat menampilkan data operasional secara langsung, seperti jumlah benda yang berhasil diproses atau status perangkat saat marking sedang berlangsung. Dengan adanya tampilan visual ini, operator dapat memantau alat tanpa harus terhubung ke perangkat lain seperti laptop atau aplikasi tambahan, sehingga proses kerja menjadi lebih ringkas dan praktis.

Gambar 2. 7 LCD I2C

(Sumber : electronicwings.com)

LCD yang digunakan dalam proyek ini umumnya merupakan LCD 16x2 seperti padad Gambar 2.7 yang sudah dilengkapi dengan modul antarmuka I2C di bagian belakangnya. Modul tambahan tersebut memungkinkan LCD berkomunikasi dengan mikrokontroler hanya melalui dua jalur utama, yaitu SDA dan SCL. Keberadaan modul I2C ini membuat kebutuhan pin pada ESP32 menjadi jauh lebih efisien. Tanpa antarmuka I2C, LCD jenis ini biasanya memakan banyak pin digital jika menggunakan mode paralel. Dengan I2C, cukup dua pin saja sudah bisa mengelola keseluruhan tampilan, sehingga sisa pin dapat dipakai untuk sensor dan aktuator lain. LCD I2C bekerja dengan prinsip mengirimkan data karakter melalui protokol komunikasi serial yang sudah distandarkan. Protokol ini sangat membantu dalam merampingkan rangkaian dan mempermudah proses pemrograman karena data tidak lagi dikirim sebagai sinyal paralel, melainkan dalam bentuk paket data yang lebih sederhana. Mikrokontroler kemudian menerjemahkan data tersebut menjadi karakter yang ditampilkan pada layar. Dalam sistem ini, LCD diprogram untuk menampilkan jumlah marking secara real-time, status sensor proximity, maupun indikator lain yang dianggap penting.

7. Buck Converter

Buck converter adalah sebuah rangkaian elektronik yang berfungsi untuk menurunkan tegangan dari nilai yang lebih tinggi ke nilai yang lebih rendah

dengan efisiensi yang jauh lebih baik dibandingkan regulator linear biasa. Komponen ini cukup penting digunakan dalam sistem yang menggabungkan perangkat dengan kebutuhan tegangan yang berbeda, seperti motor DC yang biasanya membutuhkan 12 volt, sementara mikrokontroler seperti ESP32 memerlukan tegangan yang jauh lebih rendah di sekitar 5 volt atau 3,3 volt.

Gambar 2. 8 Buck converter

(Sumber : Shopee.co.id)

Pada proyek automarking berbasis ESP32, buck converter seperti pada Gambar 2.10 digunakan sebagai perantara antara sumber daya utama dan perangkat elektronika yang lebih sensitif. Misalnya, ketika seluruh sistem menggunakan PSU 12 volt untuk motor dan konveyor, ESP32 tentu tidak bisa langsung dihubungkan ke tegangan sebesar itu. Dengan bantuan buck converter, tegangan 12 volt tersebut diturunkan menjadi 5 volt yang lebih aman, stabil, dan sesuai dengan kebutuhan ESP32, LCD I2C, sensor proximity, dan buzzer. Keberadaan buck converter juga membantu menjaga agar tegangan yang masuk ke perangkat elektronik tetap konsisten meskipun motor sedang bekerja dan menyebabkan fluktuasi arus yang cukup besar.

Cara kerja buck converter memanfaatkan proses *switching* yang sangat cepat menggunakan komponen seperti transistor MOSFET, induktor, dan dioda. Dengan metode ini, buck converter mampu menghasilkan efisiensi yang tinggi, bahkan bisa mencapai 85–95%. Efisiensi ini membuat buck converter tidak cepat panas meskipun bekerja dalam jangka waktu lama. Hal ini berbeda dengan regulator linear seperti 7805 yang cenderung membuang energi berlebih dalam

bentuk panas ketika menurunkan tegangan. Dalam sistem automarking, penggunaan buck converter juga memberikan keuntungan dari sisi kestabilan. Motor DC memiliki karakteristik beban yang berubah-ubah, terutama saat start atau ketika torsi yang dibutuhkan meningkat. Jika mikrokontroler disuplai dari jalur yang sama tanpa dipisahkan buck converter, lonjakan arus motor dapat menyebabkan tegangan drop dan membuat ESP32 *reset* atau hang. Dengan menempatkan buck converter, jalur logika menjadi lebih bersih dan terjaga dari gangguan noise yang berasal dari aktuator. Dari segi bentuk dan harga, buck converter umumnya tersedia dalam ukuran yang kecil dan sangat terjangkau. Banyak model populer seperti modul berbasis LM2596 yang sudah dilengkapi dengan potensiometer untuk mengatur tegangan keluaran secara presisi. Beberapa tipe juga menyediakan indikator voltmeter langsung di papan, sehingga memudahkan pengecekan tanpa alat tambahan.

8. Modul Relay

Modul relay merupakan salah satu komponen penting yang digunakan dalam sistem automarking ini, khususnya untuk mengendalikan beban listrik yang membutuhkan arus dan tegangan lebih besar dibandingkan kemampuan keluaran langsung dari mikrokontroler ESP32. Secara sederhana, relay dapat diibaratkan sebagai saklar otomatis yang dapat dikendalikan secara elektronik. Dengan adanya relay, ESP32 tidak perlu terhubung langsung ke beban seperti motor konveyor atau aktuator marking, sehingga sistem menjadi lebih aman dan terlindungi.

Gambar 2. 9 Module Relay

(Sumber : Shopee.co.id)

Pada sistem ini, modul relay seperti pada Gambar 2.11 digunakan untuk mengatur kondisi hidup dan mati pada konveyor serta proses marking. Ketika ESP32 memberikan sinyal logika tertentu ke input relay, kontak di dalam relay akan berubah posisi, sehingga aliran listrik ke beban dapat disambungkan atau diputus. Cara kerja ini memungkinkan konveyor dihentikan sementara saat proses marking berlangsung dan dihidupkan kembali setelah proses selesai. Penggunaan relay sangat membantu dalam mengatur alur kerja sistem tanpa harus menggunakan mekanisme mekanis tambahan. Modul relay yang digunakan umumnya sudah dilengkapi dengan rangkaian pendukung seperti transistor driver, diode proteksi, dan optocoupler. Keberadaan rangkaian ini membuat relay dapat dikendalikan langsung oleh pin digital ESP32 tanpa membebani mikrokontroler. Selain itu, diode proteksi berfungsi untuk melindungi rangkaian dari lonjakan arus balik yang dapat muncul saat relay berpindah kondisi, sehingga umur komponen elektronik menjadi lebih panjang.

Dalam penerapannya, relay bekerja dengan memutus atau menghubungkan suplai tegangan ke motor konveyor. Saat sistem mendeteksi adanya objek melalui sensor proximity, ESP32 akan mematikan relay sehingga konveyor berhenti dan objek berada pada posisi yang tetap. Setelah proses marking selesai, ESP32 kembali mengaktifkan relay dan konveyor dapat berjalan kembali. Pola kerja ini membuat sistem automarking lebih terkontrol dan memastikan proses penandaan dilakukan pada posisi yang tepat. Keunggulan penggunaan modul relay dalam sistem ini adalah fleksibilitasnya. Relay dapat digunakan untuk mengendalikan berbagai jenis beban, baik beban DC maupun AC, selama masih sesuai dengan spesifikasi relay yang digunakan. Dengan memanfaatkan modul relay sebagai pengendali beban, sistem automarking tidak hanya menjadi lebih aman, tetapi juga lebih rapi dari sisi perancangan rangkaian. ESP32 cukup berperan sebagai pengendali logika, sementara relay menangani beban daya yang lebih besar. Pendekatan ini menjadikan sistem lebih andal, mudah dikembangkan, dan sesuai dengan konsep otomasi yang umum diterapkan di dunia industri.

9. *Dinamo Spray Socket*

Dinamo sprayer socket merupakan komponen aktuator yang digunakan dalam sistem automarking ini untuk menjalankan proses pemberian tanda pada objek. Komponen ini bekerja sebagai penggerak utama pada alat penyemprot, di mana dinamo akan mengalirkan cairan marking melalui nozzle sehingga tanda dapat diberikan secara otomatis tanpa campur tangan operator. Penggunaan dinamo sprayer dipilih karena konstruksinya sederhana, mudah didapatkan, dan sudah banyak digunakan pada aplikasi penyemprotan cairan skala kecil.

Gambar 2. 10 Dinamo Sprayer

(Sumber : Shopee.co.id)

Pada sistem yang dirancang, dinamo sprayer socket seperti pada Gambar 2.12 berfungsi untuk menggerakkan pompa penyemprot ketika proses marking berlangsung. Dinamo ini bekerja menggunakan sumber tegangan DC yang relatif rendah, sehingga cocok digunakan pada sistem berbasis mikrokontroler seperti ESP32. Namun karena arus kerja dinamo cukup besar, dinamo tidak dikendalikan secara langsung oleh ESP32, melainkan melalui modul relay sebagai penghubung antara mikrokontroler dan beban. Dengan cara ini, ESP32 hanya berfungsi sebagai pengendali logika, sementara suplai daya ke dinamo dikontrol oleh relay.

Cara kerja dinamo sprayer dalam sistem automarking cukup sederhana. Ketika sensor proximity mendeteksi adanya benda yang akan diberi marking, ESP32 mengirimkan sinyal ke modul relay untuk mengaktifkan dinamo sprayer.

Dinamo kemudian bekerja selama waktu tertentu untuk menyemprotkan cairan marking ke permukaan objek. Setelah durasi penyemprotan terpenuhi, ESP32 mematikan relay sehingga dinamo berhenti bekerja. Proses ini berlangsung secara otomatis dan berulang setiap kali objek terdeteksi. Penggunaan dinamo sprayer socket memberikan keuntungan dari segi efisiensi dan konsistensi hasil marking. Dibandingkan dengan proses manual, penyemprotan menggunakan dinamo dapat menghasilkan tanda yang lebih merata karena tekanan dan durasi penyemprotan dapat diatur melalui program. Selain itu, sistem ini juga mengurangi ketergantungan pada tenaga manusia dan meminimalkan kesalahan akibat faktor kelelahan operator. Dari sisi perancangan sistem, dinamo sprayer socket juga relatif mudah diintegrasikan dengan komponen lain. Koneksi listriknya cukup sederhana, hanya memerlukan suplai tegangan dan jalur kontrol melalui relay. Hal ini memudahkan proses perakitan dan perawatan alat. Apabila terjadi kerusakan, dinamo sprayer dapat diganti dengan cepat tanpa perlu melakukan perubahan besar pada rangkaian.

10. *Nozzle*

Nozzle merupakan salah satu komponen penting dalam sistem automarking karena berfungsi sebagai media akhir untuk mengeluarkan cairan marking ke permukaan objek. Walaupun bentuknya terlihat sederhana, *nozzle* memiliki peran besar dalam menentukan kualitas hasil penandaan. *Nozzle* yang digunakan pada sistem ini dirancang untuk mengatur arah, tekanan, serta pola semprotan agar cairan marking dapat mengenai objek secara tepat dan merata.

Gambar 2. 11 Nozzle

(Sumber : moglix.com)

Dalam sistem automarking berbasis ESP32 ini, *nozzle* seperti pada Gambar 2.13 dipasang pada ujung saluran penyemprotan yang terhubung langsung dengan dinamo sprayer. Ketika dinamo aktif, cairan marking akan terdorong menuju *nozzle* dan kemudian dikeluarkan dalam bentuk semprotan. Pola semprotan yang dihasilkan sangat bergantung pada desain lubang *nozzle*, baik dari segi ukuran maupun bentuknya. Oleh karena itu, pemilihan *nozzle* yang sesuai menjadi faktor penting agar hasil marking tidak terlalu menyebar atau justru terlalu sempit. Penggunaan *nozzle* memungkinkan proses marking dilakukan secara lebih terkontrol dibandingkan penyemprotan manual. Dengan bantuan *nozzle*, arah semprotan dapat difokuskan ke titik tertentu sehingga cairan marking tidak mengenai area lain yang tidak diinginkan. Hal ini sangat membantu dalam menjaga kerapian hasil marking, terutama ketika objek bergerak di atas konveyor dan membutuhkan ketepatan posisi saat diberi tanda.

Dari sisi integrasi sistem, *nozzle* bekerja secara pasif dan tidak memerlukan sumber daya listrik tersendiri. *Nozzle* hanya merespons tekanan cairan yang dihasilkan oleh dinamo sprayer. Meskipun demikian, keberadaan *nozzle* tetap sangat berpengaruh terhadap performa sistem secara keseluruhan. Jika *nozzle* tersumbat atau ukurannya tidak sesuai, hasil marking bisa menjadi tidak konsisten. Oleh karena itu, dalam perancangan alat ini, *nozzle* dipilih dengan mempertimbangkan kemudahan perawatan dan pembersihan.

11. Sensor Tegangan

Sensor tegangan merupakan salah satu komponen pendukung yang digunakan dalam sistem automarking berbasis prototipe menggunakan ESP32. Keberadaan sensor ini berfungsi untuk memantau kondisi tegangan kerja pada sistem, khususnya pada jalur catu daya yang menyuplai aktuator seperti motor konveyor dan dinamo sprayer. Dengan adanya pemantauan tegangan, sistem tidak hanya bekerja secara otomatis, tetapi juga memiliki aspek pengawasan terhadap kestabilan sumber daya listrik yang digunakan. Pada penelitian ini, jenis sensor

tegangan yang digunakan adalah modul voltage sensor DC berbasis pembagi tegangan dengan rentang pengukuran 0–25 V DC. Sensor ini dipilih karena sesuai dengan kebutuhan sistem yang menggunakan sumber tegangan 12 V DC sebagai suplai utama. Selain itu, sensor tegangan tipe ini memiliki bentuk yang sederhana, mudah diintegrasikan dengan mikrokontroler ESP32, serta cukup akurat untuk keperluan monitoring pada skala prototipe.

Gambar 2. 12 Sensor Tegangan

(Sumber : Alibaba.com)

Prinsip kerja sensor tegangan seperti pada Gambar 2.14 ini didasarkan pada rangkaian pembagi tegangan yang menurunkan nilai tegangan input menjadi tegangan yang aman untuk dibaca oleh pin analog ESP32. Tegangan hasil pembagian tersebut kemudian dikonversi menjadi data digital melalui fitur ADC (Analog to Digital Converter) yang terdapat pada ESP32. Data inilah yang selanjutnya diolah oleh sistem untuk mengetahui kondisi tegangan aktual pada rangkaian. Penggunaan sensor tegangan dalam sistem automarking memiliki peran yang cukup penting, terutama dalam menjaga kestabilan proses kerja. Apabila terjadi penurunan atau kenaikan tegangan di luar batas yang telah ditentukan, ESP32 dapat memberikan respon berupa penghentian sementara proses marking atau memberikan peringatan kepada operator melalui buzzer maupun tampilan LCD. Dengan cara ini, potensi kerusakan komponen akibat tegangan tidak normal dapat diminimalkan. Dari sisi penerapan, sensor tegangan dihubungkan ke salah satu pin analog ESP32 sehingga pembacaan nilai tegangan dapat dilakukan secara real-time. Nilai tegangan yang terbaca kemudian ditampilkan pada LCD sebagai informasi tambahan bagi pengguna. Informasi ini berguna untuk mengetahui apakah sistem bekerja dalam kondisi normal atau

sedang mengalami gangguan pada sumber daya. Pemilihan sensor tegangan tipe modul DC 0–25 V pada penelitian ini dinilai tepat karena sesuai dengan karakteristik sistem automarking yang menggunakan aktuator berbasis motor DC. Sensor ini juga mendukung konsep arus kuat yang terdapat pada tugas akhir, karena digunakan untuk memantau jalur daya yang menyuplai beban berarus relatif besar. Dengan demikian, penggunaan sensor tegangan tidak hanya berfungsi sebagai alat monitoring, tetapi juga sebagai bagian dari sistem pengaman dan evaluasi kinerja alat secara keseluruhan.

2.2.3 Penjelasan *Software* yang digunakan

1. Arduino IDE

Dalam proses pembuatan sistem automarking berbasis ESP32, perangkat lunak utama yang digunakan adalah Arduino Integrated Development Environment (Arduino IDE**)**. Arduino IDE merupakan software yang dirancang khusus untuk memudahkan proses pengembangan aplikasi berbasis mikrokontroler. Walaupun tampilannya sederhana, Arduino IDE menawarkan fasilitas yang lengkap untuk menulis program, melakukan kompilasi, mengunggah kode ke mikrokontroler, serta memantau jalannya program secara langsung. Karena sistem automarking ini melibatkan beberapa komponen seperti sensor proximity, motor DC, driver motor, buzzer, dan LCD I2C, keberadaan software yang praktis dan mudah digunakan sangat membantu selama proses pembuatan dan pengujian alat.

Gambar 2. 13 Arduino

(Sumber : Arduino-Ide.id)

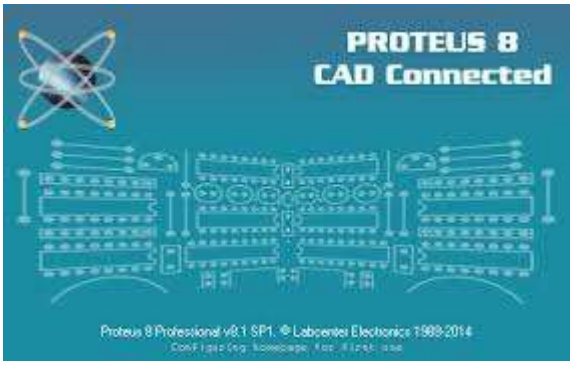
Software pada Gambar 2.13 bekerja sebagai pusat pengolahan program. Semua instruksi untuk ESP32, mulai dari cara membaca sensor, menggerakkan motor, menghitung jumlah marking, menampilkan informasi ke LCD, hingga mengatur buzzer saat kondisi tertentu muncul, ditulis dalam bentuk sketsa (sketch) pada software ini. Penulisan program di Arduino IDE tidak membutuhkan konfigurasi yang rumit karena bahasa pemrogramannya sudah disederhanakan. Banyak fungsi dasar sudah tersedia dan dapat langsung digunakan, sehingga saya bisa fokus pada logika kerja alat tanpa harus berurusan dengan detail teknis yang terlalu kompleks. Hal ini membuat proses pengembangan berjalan lebih cepat dan lebih stabil. Selain digunakan untuk menulis kode program, Arduino IDE juga menyediakan berbagai library yang sangat membantu. Library tersebut ibarat tambahan alat bantu yang tinggal dipasang ketika dibutuhkan. Misalnya, untuk menghubungkan LCD berbasis I2C, saya tinggal menambahkan library LiquidCrystal_I2C. Begitu juga ketika memerlukan fungsi PWM untuk mengatur kecepatan motor atau memproses input dari sensor proximity. Library-library ini membuat proses coding lebih praktis, sekaligus mengurangi potensi kesalahan karena sebagian besar fungsi dasar sudah diuji dan dipastikan berjalan dengan baik. Fitur lain dari Arduino IDE yang sangat berguna dalam tugas akhir ini adalah Serial Monitor dan Serial Plotter. Melalui fitur Serial Monitor, saya bisa memantau nilai sensor secara langsung ketika alat dijalankan. Ini sangat membantu saat menentukan batas pembacaan sensor proximity, memastikan logika perhitungan jumlah produk berjalan benar, atau melihat respons sistem ketika motor mulai bekerja. Sementara itu, Serial Plotter menampilkan data sensor dalam bentuk grafik sehingga memudahkan dalam memahami pola pembacaan dan memastikan sensor berfungsi stabil. Kedua fitur ini sangat mempersingkat waktu debugging karena saya bisa mengetahui masalah tanpa harus membuka casing alat atau melakukan pengukuran manual.

Arduino IDE juga memiliki kemampuan untuk ditambahkan tipe board lain seperti ESP32. Untuk dapat memprogram ESP32, saya hanya perlu

menambahkan URL board manager ke menu pengaturan, lalu mengunduh dukungan board ESP32. Setelah itu, proses pemrograman dapat langsung dilakukan hanya dengan menghubungkan ESP32 ke komputer menggunakan kabel USB. Proses upload pun cukup cepat dan jika terjadi kesalahan, Arduino IDE akan menampilkan pesan error yang mudah dipahami, sehingga langkah perbaikan bisa dilakukan segera. Keandalan Arduino IDE membuatnya sangat cocok digunakan pada proyek seperti sistem automarking ini. Tidak hanya berfungsi sebagai software pemrograman, tetapi juga sebagai alat utama untuk pengujian, analisis, dan penyempurnaan kerja sistem. Selama proses pengerjaan, Arduino IDE membantu saya memastikan bahwa setiap komponen bekerja sesuai fungsinya, dan keseluruhan sistem dapat berjalan dengan stabil. Kemudahan penggunaan, dukungan library yang lengkap, serta fitur monitoring yang sangat membantu menjadikan Arduino IDE sebagai software yang sangat berperan penting dalam keberhasilan pembuatan tugas akhir ini.

2. Proteus

Dalam proses perancangan sistem automarking berbasis ESP32, software yang juga digunakan sebagai alat bantu adalah Proteus. Proteus merupakan aplikasi simulasi rangkaian elektronik yang cukup dikenal dan banyak dipakai oleh mahasiswa maupun praktisi teknik. Perangkat lunak ini memudahkan pengguna untuk membuat rangkaian elektronik secara virtual sebelum benar-benar dirakit di dunia nyata. Dengan adanya Proteus, proses perancangan dapat dilakukan lebih aman, terukur, dan efisien, karena kesalahan pada rangkaian bisa diketahui lebih awal tanpa takut merusak komponen asli.



Gambar 2. 14 Proteus

(Sumber : Xtronic.org)

Software pada Gambar 2.14 memiliki dua fitur utama, yaitu ISIS (simulasi rangkaian) dan ARES (desain PCB). Dalam pengerjaan tugas akhir ini, bagian yang paling sering dimanfaatkan adalah fitur simulasi rangkaian. Di dalam ISIS, saya dapat menggambar skema wiring seperti penempatan ESP32, sensor proximity, driver motor, LCD I2C, dan beberapa komponen lain yang digunakan dalam sistem automarking. Setiap komponen memiliki simbol dan parameter yang bisa diatur sesuai kebutuhan, sehingga rangkaian yang dibuat mendekati kondisi sebenarnya ketika alat dirakit. Penggunaan Proteus sangat membantu terutama pada tahap awal, saat saya perlu memastikan hubungan antar komponen sudah benar dan tidak ada jalur yang tertukar. Misalnya, ketika menghubungkan pin sensor ke ESP32, Proteus memungkinkan saya mengecek apakah rangkaian sudah sesuai atau apakah ada kesalahan seperti jalur short, polaritas salah, atau kebutuhan tegangan yang tidak cocok. Dengan simulasi ini, risiko kegagalan saat alat dirakit dapat ditekan seminimal mungkin. Selain itu, Proteus juga mendukung simulasi perilaku beberapa komponen elektronik tertentu. Walaupun ESP32 tidak dapat disimulasikan secara penuh seperti mikrokontroler AVR, Proteus tetap sangat berguna untuk memastikan koneksi dasar seperti rel tegangan, jalur input-output, dan tata letak rangkaian secara umum. Dengan begitu, ketika proses perakitan dilakukan di breadboard atau PCB, saya sudah memiliki gambaran yang jelas mengenai susunan rangkaiannya.

Proteus juga menyediakan tampilan visual yang rapi dan mudah dipahami.

Dengan fitur zoom, grouping, dan pengaturan layer, rangkaian dapat disusun dengan lebih terstruktur dan profesional. Hasil gambar rangkaian dari Proteus ini kemudian digunakan sebagai bahan untuk laporan, baik sebagai wiring diagram, skema sistem, maupun tata letak rangkaian elektronik. Gambar yang dihasilkan terlihat lebih jelas dan formal, sehingga lebih mudah dipahami oleh pembaca laporan, termasuk dosen pembimbing atau penguji. Keunggulan lain dari Proteus adalah kemampuannya memperkirakan apakah suplai daya dalam rangkaian sudah mencukupi. Ketika sebuah komponen membutuhkan arus lebih tinggi, Proteus memberi peringatan melalui notifikasi sehingga saya dapat memperbaiki rangkaian terlebih dahulu. Hal ini cukup membantu ketika menentukan penggunaan power supply, buck converter, dan fuse untuk alat automarking ini.

3. Autocad

AutoCAD merupakan salah satu perangkat lunak yang sangat populer dalam bidang perancangan teknik, terutama untuk membuat gambar kerja yang presisi dan mudah dipahami. Pada penyusunan tugas akhir ini, AutoCAD digunakan untuk membantu proses pembuatan gambar rancangan alat, baik yang berkaitan dengan tata letak komponen, wiring diagram, maupun gambar tampak atas dari keseluruhan sistem automarking berbasis ESP32. Perangkat lunak ini memberikan kemudahan dalam menuangkan ide desain ke bentuk visual yang rapi, akurat, dan sesuai standar gambar teknik. Pemilihan AutoCAD didasarkan pada kemampuannya dalam menghasilkan gambar teknik 2D/3D yang detail dan memiliki tingkat ketelitian yang tinggi. Ketika merancang sistem automarking, saya perlu memastikan bahwa setiap komponen dalam alat, seperti sensor proximity, ESP32, motor DC, driver motor, buzzer, LCD I2C, dan konveyor mini, ditempatkan pada posisi yang tepat.

Gambar 2. 15 Autocad

(Sumber : Autodesk.com)

Selain itu, software pada Gambar 2.15 juga mendukung penggunaan layer, yang membuat penyusunan gambar menjadi lebih terorganisir. Setiap layer bisa digunakan untuk kategori tertentu, misalnya layer khusus kabel, layer komponen utama, layer rangka mekanik, atau layer anotasi. Dengan pembagian seperti ini, proses revisi dan pengecekan gambar menjadi jauh lebih mudah. Ketika ada perubahan desain, saya tidak perlu mengulangi gambar dari awal cukup memodifikasi bagian pada layer yang berkaitan. Pengaturan layer ini juga sangat membantu ketika gambar sudah kompleks dan memiliki banyak detail. Dalam konteks pembuatan wiring diagram, AutoCAD memberikan fleksibilitas untuk menggambar jalur kabel, terminal, dan koneksi antar komponen dengan tampilan yang jelas dan profesional.

BAB III
METODE PENELITIAN

3.1 Konsep Penelitian

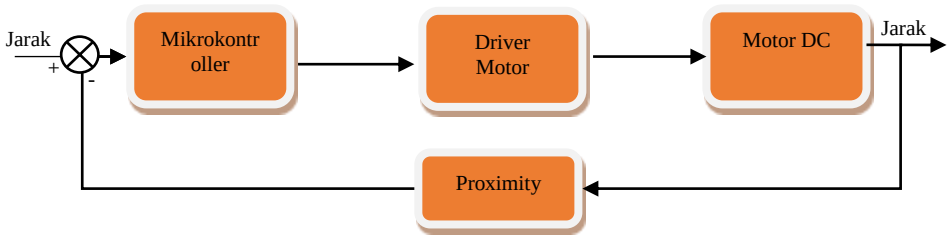
Konsep penelitian pada tugas akhir ini berfokus pada penerapan sistem otomasi untuk meningkatkan efisiensi proses kerja di lingkungan perusahaan. Permasalahan yang sering dijumpai di lapangan adalah masih digunakannya metode manual dalam proses marking, yang membutuhkan keterlibatan operator secara terus-menerus dan berpotensi menurunkan efisiensi waktu serta konsistensi hasil. Kondisi tersebut dapat menyebabkan proses produksi berjalan kurang optimal, terutama ketika jumlah objek yang harus diproses cukup banyak.

Pemecahan masalah tersebut dilakukan dengan merancang sistem automarking berbasis mikrokontroler ESP32 yang mampu bekerja secara otomatis berdasarkan kondisi objek. Sistem ini memanfaatkan sensor proximity untuk mendeteksi keberadaan objek pada konveyor, sehingga proses marking hanya dilakukan ketika diperlukan. Dengan adanya kontrol otomatis ini, konveyor dan alat marking tidak bekerja secara terus-menerus tanpa kebutuhan, sehingga penggunaan energi listrik dan waktu kerja dapat lebih terkendali.

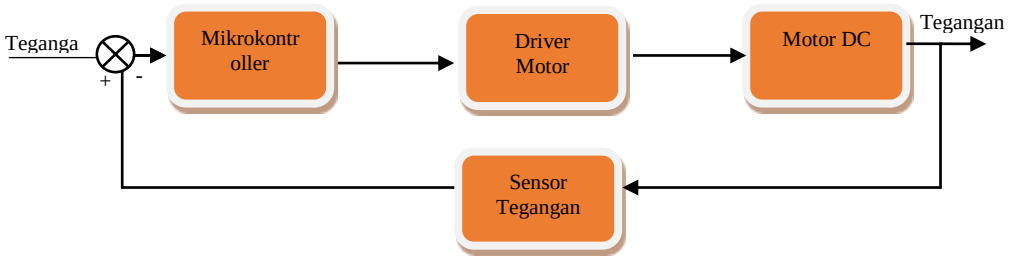
Melalui penerapan sistem automarking di lingkungan perusahaan, diharapkan proses marking dapat berjalan lebih efisien, konsisten, dan terstruktur. Selain meningkatkan efektivitas kerja, sistem ini juga dapat mengurangi ketergantungan pada proses manual serta meminimalkan pemborosan energi dan sumber daya. Dengan demikian, konsep penelitian ini tidak hanya berfokus pada aspek teknis perancangan alat, tetapi juga pada peningkatan efisiensi dan produktivitas proses produksi di perusahaan.

3.1.1 Prinsip dan Mekanisme Kerja Sistem

Prinsip kerja sistem automarking ini dimulai dari proses pendeteksian objek menggunakan sensor proximity. Sensor proximity berfungsi untuk mendeteksi keberadaan objek logam yang bergerak di atas konveyor. Ketika objek berada dalam jangkauan sensor, sensor proximity akan mengirimkan sinyal ke mikrokontroler ESP32 sebagai tanda bahwa objek siap untuk diproses. Selain sensor proximity, sistem juga dilengkapi dengan sensor tegangan yang berfungsi untuk memonitor kondisi suplai daya pada sistem, khususnya pada beban seperti motor konveyor dan dinamo sprayer. Sensor tegangan ini digunakan karena ESP32 tidak dapat membaca tegangan tinggi secara langsung, sehingga diperlukan rangkaian sensor sebagai pengaman dan pemantau kondisi kerja sistem. ESP32 berfungsi sebagai pusat kendali yang menerima data dari sensor proximity dan sensor tegangan. Data tersebut kemudian diolah untuk menentukan apakah proses marking dapat dijalankan atau tidak. Jika kondisi objek terdeteksi dan tegangan berada dalam batas normal, ESP32 akan mengaktifkan aktuator melalui modul relay. Proses marking dilakukan menggunakan dinamo sprayer yang menyemprotkan cairan penanda melalui nozzle. Dengan prinsip kerja ini, sistem hanya akan melakukan marking ketika objek benar-benar terdeteksi dan kondisi sistem dinyatakan aman, sehingga proses marking menjadi lebih efisien dan terkontrol dibandingkan metode manual.



Gambar 3. 1 Diagram Blok Sensor Proximity



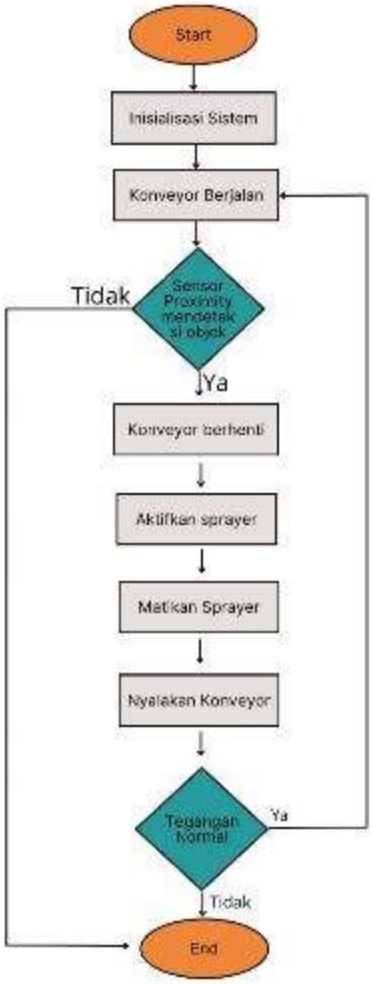
Gambar 3. 2 Diagram Blok Sensor Tegangan

Mekanisme kerja sistem seperti pada Gambar 3.1 dan Gambar 3.2 diawali ketika konveyor bergerak membawa objek menuju area sensor. Pada kondisi awal, ESP32 berada dalam mode siaga dan terus menerima data dari sensor proximity serta sensor tegangan. Selama tidak ada objek yang terdeteksi, konveyor akan terus berjalan dan sistem marking dalam keadaan mati. Ketika objek logam melewati area sensor, sensor proximity mendeteksi keberadaan objek tersebut dan mengirimkan sinyal ke ESP32. Setelah menerima sinyal ini, ESP32 melakukan pengecekan kondisi tegangan melalui sensor tegangan untuk memastikan suplai daya berada dalam kondisi normal. Pengecekan ini bertujuan untuk mencegah gangguan sistem akibat tegangan yang tidak stabil. Apabila kondisi tegangan dinyatakan aman, ESP32 mengirimkan perintah ke modul relay untuk menghentikan sementara motor konveyor. Setelah konveyor berhenti dan posisi objek sesuai, ESP32 mengaktifkan relay yang terhubung dengan dinamo sprayer. Dinamo sprayer kemudian menyemprotkan cairan marking melalui nozzle selama waktu tertentu sesuai pengaturan program. Setelah proses marking selesai, ESP32 mematikan dinamo sprayer dan kembali mengaktifkan motor konveyor. Objek kemudian bergerak keluar dari area marking, dan sistem kembali ke kondisi awal untuk menunggu objek berikutnya. Seluruh proses ini berlangsung secara otomatis tanpa intervensi langsung dari operator.

3.1.2 Flowchart Sistem

Flowchart sistem automarking berbasis ESP32 seperti pada Gambar 3.3 diawali dengan proses mulai, yang menandakan sistem siap dijalankan. Pada tahap awal, sistem melakukan inisialisasi terhadap seluruh komponen utama, meliputi mikrokontroler ESP32, sensor proximity, sensor tegangan, modul relay, serta aktuator berupa motor konveyor dan dinamo sprayer. Proses inisialisasi ini bertujuan untuk memastikan seluruh perangkat berada dalam kondisi siap operasi. Setelah proses inisialisasi selesai, sistem masuk ke kondisi konveyor berjalan, yang menunjukkan bahwa sistem berada dalam mode siaga. Pada tahap ini, konveyor bergerak secara kontinu sambil menunggu adanya objek yang akan diproses. Sensor proximity secara terus-menerus melakukan pendeteksian terhadap keberadaan objek di area marking. Apabila sensor proximity belum mendeteksi objek, sistem akan tetap berada pada kondisi konveyor berjalan dan kembali mengulangi proses pendeteksian. Namun, ketika sensor proximity mendeteksi adanya objek, sistem akan melanjutkan ke tahap pembacaan sensor tegangan. Pembacaan ini dilakukan untuk memastikan bahwa tegangan kerja sistem berada dalam kondisi normal dan aman sebelum proses marking dijalankan. Jika hasil pembacaan sensor tegangan menunjukkan kondisi tidak normal, sistem tidak melanjutkan ke proses marking dan akan kembali ke mode siaga sebagai langkah pengamanan. Sebaliknya, apabila tegangan berada dalam kondisi normal, ESP32 akan memberikan perintah kepada modul relay untuk menghentikan motor konveyor. Penghentian konveyor bertujuan agar posisi objek tetap stabil saat proses marking berlangsung. Setelah konveyor berhenti, sistem akan mengaktifkan dinamo sprayer untuk melakukan proses marking pada objek. Sprayer akan bekerja selama waktu tertentu yang telah ditentukan dalam program. Setelah durasi marking tercapai, sistem akan mematikan sprayer dan selanjutnya mengaktifkan kembali motor konveyor melalui relay. Tahap terakhir dari flowchart menunjukkan bahwa sistem kembali ke mode siaga. Pada kondisi ini,

konveyor kembali berjalan dan sistem siap untuk memproses objek berikutnya. Seluruh tahapan tersebut akan berlangsung secara berulang selama sistem masih dalam kondisi aktif.

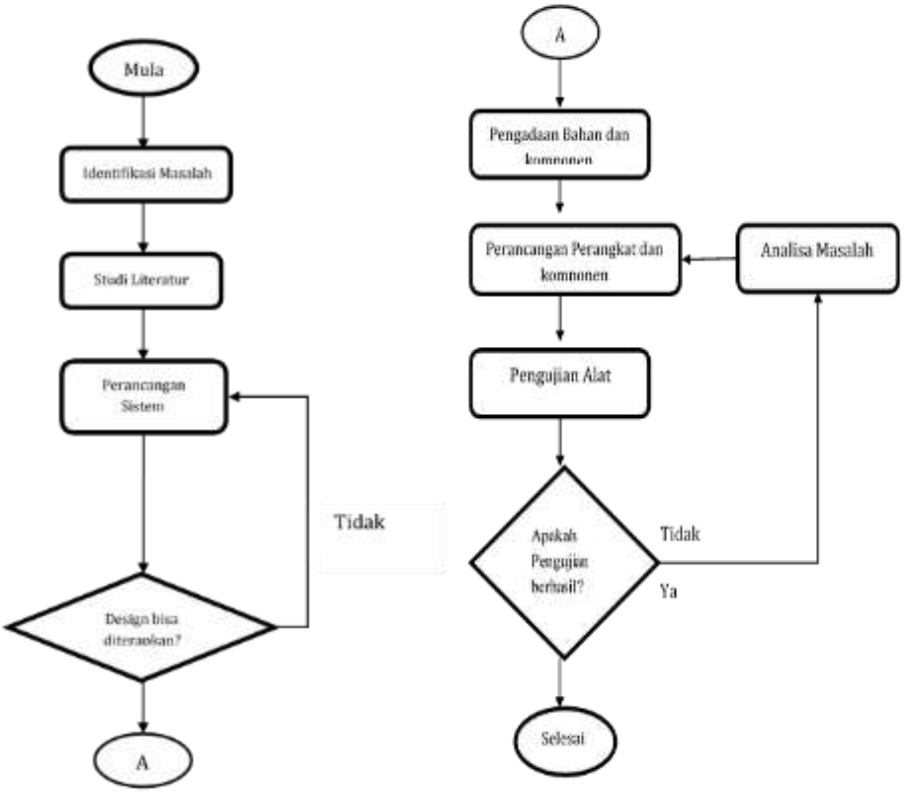


Gambar 3. 3 Flowchart Sistem

3.2 Tahapan Penelitian

Pada sub bab ini alur penelitian menjelaskan tentang perancangan sistem secara keseluruhan melalui *flowchart* pada Gambar 3.4. Penentuan alur penelitian bertujuan untuk mempermudah dalam memahami alur tindakan. Diagram alir penelitian adalah representasi visual dari langkah-langkah yang diambil dalam sebuah penelitian dari awal hingga akhir, sehingga

memudahkan pemahaman tentang bagaimana penelitian tersebut dilakukan dan bagaimana setiap tahapan berhubungan satu sama lain.



Gambar 3. 4 Diagram Alir Penelitian

Tahapan penelitian pada tugas akhir ini disusun secara sistematis agar proses perancangan dan pembuatan sistem automarking berbasis ESP32 dapat berjalan terarah dan sesuai dengan tujuan yang ingin dicapai. Setiap tahapan dilakukan secara berurutan, mulai dari identifikasi permasalahan hingga pengujian sistem secara keseluruhan. Tahap awal penelitian dimulai dengan identifikasi masalah yang dilakukan melalui pengamatan langsung terhadap proses marking yang masih dilakukan secara manual. Pada tahap ini, peneliti mengamati alur kerja, keterlibatan operator, serta kendala yang sering muncul selama proses marking berlangsung. Hasil dari tahap ini digunakan sebagai dasar untuk menentukan kebutuhan sistem dan arah pengembangan alat yang akan dibuat. Tahap berikutnya adalah studi literatur, yaitu

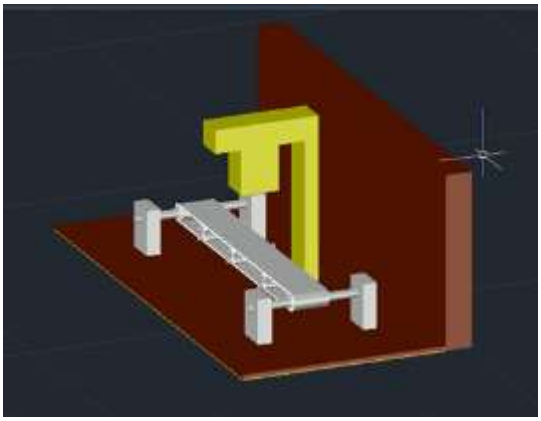
mengumpulkan dan mempelajari berbagai referensi yang berkaitan dengan sistem automasi, mikrokontroler ESP32, sensor proximity, sistem konveyor, serta metode marking otomatis. Referensi diperoleh dari jurnal ilmiah, buku, dan sumber pendukung lainnya. Studi literatur ini bertujuan untuk memperkuat landasan teori sekaligus menjadi acuan dalam merancang sistem agar sesuai dengan konsep yang telah dikembangkan sebelumnya.

Setelah landasan teori diperoleh, penelitian dilanjutkan ke tahap perancangan sistem. Pada tahap ini dilakukan perancangan diagram blok, flowchart, serta perancangan perangkat keras dan perangkat lunak. Perancangan hardware meliputi pemilihan komponen dan penyusunan rangkaian, sedangkan perancangan software meliputi pembuatan logika program yang akan dijalankan pada ESP32 sesuai dengan alur kerja sistem. Tahap selanjutnya adalah pembuatan dan perakitan alat. Komponen yang telah dirancang sebelumnya dirakit menjadi satu kesatuan sistem automarking. Proses perakitan dilakukan secara bertahap untuk memastikan setiap komponen dapat terhubung dan berfungsi dengan baik. Setelah perakitan selesai, dilakukan pengecekan awal untuk memastikan tidak terjadi kesalahan pada rangkaian.

Setelah alat selesai dirakit, penelitian dilanjutkan ke tahap pengujian sistem. Pengujian dilakukan untuk mengetahui apakah sistem dapat bekerja sesuai dengan perancangan yang telah dibuat. Pengujian meliputi pengujian sensor proximity dalam mendeteksi objek, pengujian relay dalam mengendalikan motor konveyor dan dinamo sprayer, serta pengujian keseluruhan alur kerja sistem automarking. Tahap terakhir adalah analisis dan evaluasi hasil. Pada tahap ini, data hasil pengujian dianalisis untuk mengetahui kinerja sistem, kelebihan, serta keterbatasan yang masih ada. Hasil analisis tersebut kemudian digunakan sebagai bahan pembahasan dan penarikan kesimpulan, sekaligus menjadi dasar untuk memberikan saran pengembangan sistem di masa mendatang.

3.3 Perencanaan dan Desain

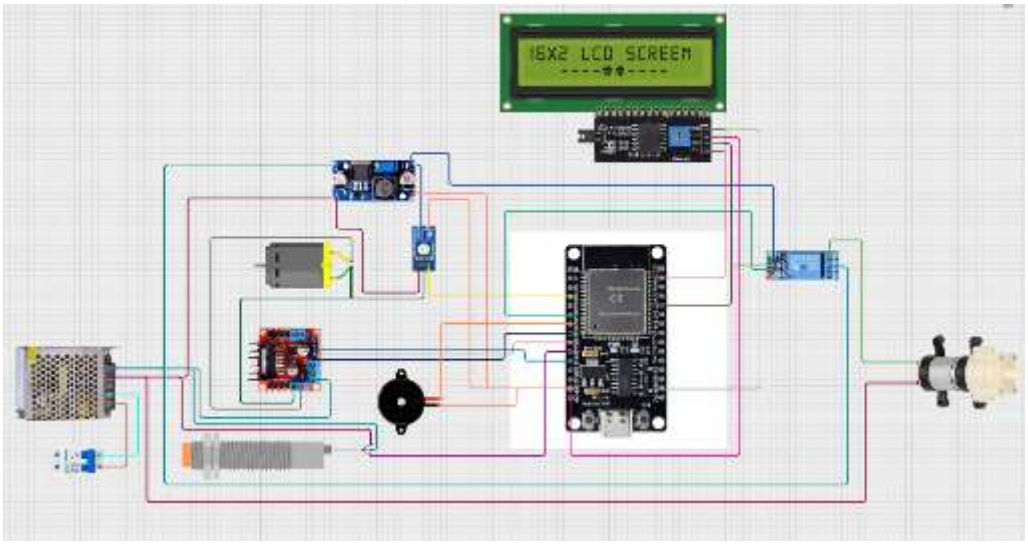
Tahap perencanaan dan desain merupakan bagian awal yang sangat menentukan keberhasilan dalam pembuatan sistem automarking berbasis prototipe menggunakan ESP32. Pada tahap ini, peneliti mulai dengan mengidentifikasi kebutuhan sistem yang ada di lapangan, khususnya kondisi proses marking yang sebelumnya masih dilakukan secara manual. Proses manual tersebut dinilai kurang efisien karena bergantung pada ketelitian operator, waktu pengerjaan yang tidak konsisten, serta potensi terjadinya kesalahan saat proses penyemprotan tanda pada produk.Berdasarkan permasalahan tersebut, dilakukan perencanaan sistem otomasi yang mampu bekerja secara lebih terkontrol dan berulang dengan hasil yang seragam.



Gambar 3. 5 Perencanaan Desain Alat

Pada Gambar 3.5 ditentukan fungsi utama sistem, yaitu mendeteksi keberadaan objek logam menggunakan sensor proximity, menghentikan pergerakan konveyor saat objek berada di posisi marking, lalu mengaktifkan proses penyemprotan tanda secara otomatis. Selain itu, sistem juga dirancang agar mampu memantau kondisi tegangan kerja sebagai bagian dari aspek keselamatan dan kestabilan sistem.Setelah konsep sistem ditentukan, tahapan dilanjutkan dengan desain hardware

dan software. Desain hardware mencakup pemilihan dan penempatan komponen seperti ESP32 sebagai pengendali utama, sensor proximity sebagai pendeteksi objek, modul relay untuk mengontrol aktuator penyemprotan, driver motor untuk konveyor, serta catu daya yang sesuai dengan kebutuhan tegangan dan arus. Penempatan komponen dirancang secara sistematis agar mudah dalam proses perakitan, perawatan, dan pengujian.



Gambar 3. 6 Perencanaan Wiring

Sementara itu, Gambar 3.6 difokuskan pada pembuatan alur logika program yang sesuai dengan kebutuhan sistem. Program dirancang agar ESP32 mampu membaca input dari sensor proximity dan sensor tegangan, kemudian mengambil keputusan untuk mengendalikan konveyor, proses marking, buzzer, serta menampilkan informasi pada LCD. Seluruh desain tersebut kemudian disatukan pada tahap implementasi dan integrasi, sehingga sistem dapat bekerja sebagai satu kesatuan yang saling terhubung dan mendukung proses automarking secara otomatis.

3.3.1 Perencanaan Desain Web

Pada sistem automarking ini, web monitoring dirancang sebagai media untuk memantau kondisi alat secara langsung melalui jaringan

WiFi. Halaman web dihubungkan dengan ESP32 sehingga setiap perubahan kondisi pada sistem dapat ditampilkan secara otomatis tanpa perlu melakukan pembaruan halaman secara manual. Dengan adanya web monitoring, pengguna tidak harus selalu berada di dekat alat untuk mengetahui kondisi sistem yang sedang berjalan. Tampilan web dibuat dengan konsep yang sederhana agar informasi yang ditampilkan mudah dipahami oleh pengguna. Halaman utama menampilkan beberapa informasi penting, yaitu nilai tegangan yang dibaca oleh sensor tegangan, status sensor proximity, kondisi motor konveyor, status pompa sprayer, serta jumlah benda yang telah berhasil diproses. Seluruh informasi tersebut diperbarui secara *real-time* sesuai dengan data yang dikirimkan oleh ESP32.

Gambar 3. 7 Perencanaan Desain Web

Selain menampilkan data monitoring seperti padda Gambar 3.7, halaman web juga berfungsi sebagai media untuk memastikan bahwa setiap komponen bekerja sesuai dengan kondisi yang terjadi pada sistem. Sebagai contoh, ketika sensor proximity mendeteksi adanya benda logam, status sensor pada web akan berubah, motor konveyor berhenti, dan pompa sprayer aktif untuk melakukan proses penandaan. Sebaliknya, apabila tidak ada benda yang terdeteksi atau tegangan berada di bawah batas yang telah ditentukan, informasi tersebut juga akan ditampilkan pada halaman web sehingga operator dapat segera mengetahui kondisi sistem.

Perancangan tampilan web dibuat dengan mempertimbangkan kemudahan penggunaan. Susunan informasi dibuat secara teratur sehingga operator dapat melihat kondisi alat dengan cepat tanpa harus membaca banyak menu. Dengan rancangan seperti ini, proses monitoring menjadi lebih praktis dan dapat membantu pengguna dalam mengawasi kinerja sistem automarking selama proses pengujian maupun saat alat dioperasikan.

3.4 Rencana Pengujian

3.4.1 Sensor Proximity

Pengujian sensor proximity direncanakan untuk mengetahui kemampuan sensor dalam mendeteksi keberadaan objek logam pada area konveyor. Sensor akan diuji dalam beberapa kondisi, yaitu saat tidak terdapat objek, saat objek berada dalam jarak deteksi, serta saat objek bergerak melewati sensor mengikuti alur konveyor. Parameter utama yang diamati adalah perubahan logika output sensor yang diterima oleh ESP32. Melalui pengujian ini diharapkan dapat diketahui apakah sensor mampu bekerja secara stabil dan responsif terhadap objek logam, sehingga dapat dijadikan acuan dalam menentukan waktu penghentian konveyor dan proses marking.

3.4.2 Sensor Tegangan

Pengujian sensor tegangan direncanakan untuk memastikan sensor mampu membaca nilai tegangan pada sistem secara akurat. Pengujian dilakukan dengan mengamati pembacaan tegangan saat sistem berada dalam kondisi tanpa beban, saat motor konveyor aktif, serta saat seluruh beban sistem bekerja. Nilai tegangan yang terbaca oleh sensor akan menjadi dasar bagi sistem dalam memantau kondisi kerja motor dan catu daya. Pengujian ini penting untuk memastikan bahwa perubahan tegangan dapat terdeteksi dengan baik dan tidak mengganggu kinerja ESP32 maupun komponen lainnya.

3.4.3 Motor DC Konveyor

Pengujian motor DC konveyor dilakukan untuk memastikan motor dapat beroperasi sesuai perintah dari sistem kontrol. Motor akan diuji dalam kondisi menyala, berhenti, serta berhenti otomatis saat sensor proximity mendeteksi objek. Parameter yang diamati meliputi respon motor terhadap perintah kontrol dan kestabilan putaran saat sistem berjalan. Pengujian ini bertujuan untuk memastikan bahwa motor konveyor mampu mendukung alur kerja automarking secara konsisten dan tidak mengalami keterlambatan respon yang dapat memengaruhi proses marking.

BAB IV
HASIL DAN PEMBAHASAN

Pada Bab 4 berikut akan dibahas mengenai hasil dan pembahasan dari pengujian komponen yang digunakan dalam penelitian. Setiap komponen diuji untuk memastikan fungsinya berjalan sesuai dengan perancangan sistem. Hasil pengujian ditampilkan dalam bentuk data dan pengamatan langsung sebagai dasar evaluasi performa masing-masing komponen. Selain itu, hasil ini menjadi acuan untuk memastikan bahwa sistem dapat bekerja secara optimal saat diintegrasikan secara keseluruhan.

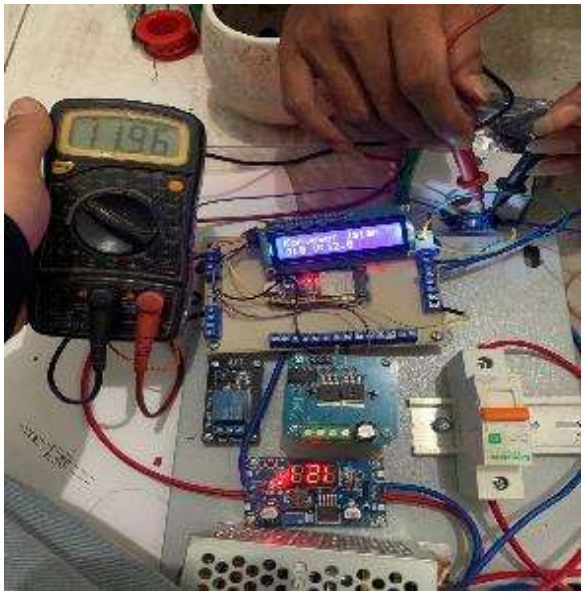
4.1 Pengujian Sensor

Pengujian sensor dilakukan untuk memastikan bahwa komponen sensor yang digunakan dalam sistem bekerja sesuai dengan fungsinya. Tujuan dari pengujian ini adalah untuk mengetahui tingkat keakuratan dan konsistensi sensor dalam membaca parameter yang diukur. Proses pengujian melibatkan pembandingan antara hasil pembacaan sensor yang ditampilkan oleh mikrokontroler dan nilai acuan dari LCD.

4.1.1 Pengujian Sensor Tegangan

Pengujian sensor tegangan dilakukan untuk memastikan bahwa sensor yang digunakan mampu membaca nilai tegangan dengan baik dan sesuai dengan kondisi sebenarnya di sistem. Pengujian ini penting karena sensor tegangan berfungsi sebagai umpan balik untuk mengetahui apakah tegangan yang masuk ke sistem berada dalam kondisi normal atau tidak.

Pada tahap pengujian, sensor dihubungkan dengan sumber tegangan yang sama seperti saat sistem digunakan, kemudian nilai tegangan yang terbaca oleh sensor ditampilkan melalui LCD. Untuk mengetahui tingkat keakuratannya, hasil pembacaan sensor dibandingkan dengan alat ukur seperti multimeter. Dari perbandingan tersebut dapat dilihat apakah terdapat selisih pembacaan, serta seberapa besar deviasi yang terjadi. Berikut adalah Gambar 4.1 menunjukan aktivitas pengujian sensor Tegangan .



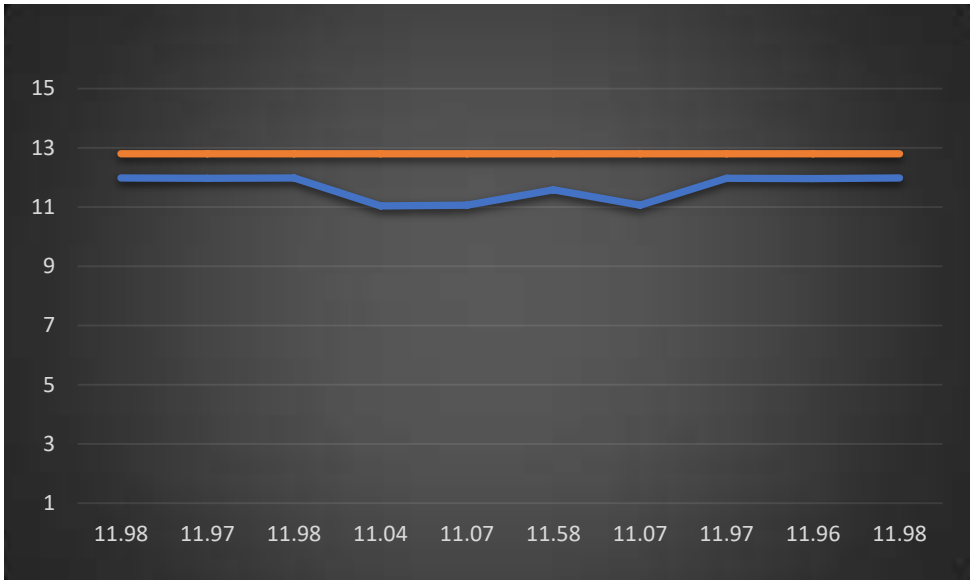
Gambar 4. 1 Aktivitas pengujian sensor Tegangan

Pengujian ini membandingkan tegangan yang akan diukur dengan multimeter dan nilai yang terbaca oleh sensor melalui LCD I2C. Selisih keduanya digunakan untuk menghitung presentase error. Hasil ini menunjukan akurasi dan konsitensi sensor dalam membaca tegangan. Tabel 4.1 menampilkan hasil pengujian 1 sensor pada rentang 12VDC beserta presentase errornya.

Tabel 4. 1 Data Pengujian Sensor Tegangan

	Tampilan Multimeter	Persentase Error
12.8	11.98 VDC	0,068%
12.8	11.97 VDC	0,069%
12.8	11.98 VDC	0,068%
12.8	11.04 VDC	0,159%
12.8	11.70 VDC	0,094%
12.8	11.58 VDC	0,105%
12.8	11.97 VDC	0,069%
12.8	11.97 VDC	0,069%
12.8	11.96 VDC	0,070%
12.8	11.98 VDC	0,068%
Rata Rata		0,007%

Jadi, rata – rata presentease error dari sensor tegangan pada sistem ini adalah 0,007%.



Gambar 4. 2 Grafik Data Pengujian Sensor Tegangan

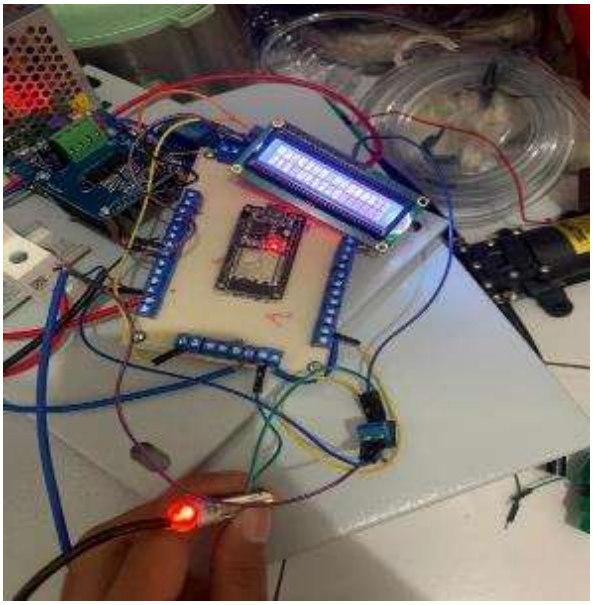
Gambar 4.2 menunjukkan hasil pembacaan sensor tegangan pada sepuluh kali pengujian. Berdasarkan grafik, nilai tegangan yang terbaca berada pada kisaran 11,04 V hingga 11,98 V ketika sumber tegangan yang digunakan sebesar 12,8 V. Hasil tersebut memperlihatkan bahwa pembacaan sensor cenderung stabil dengan selisih yang relatif kecil terhadap nilai acuan yang diukur menggunakan multimeter.

Perbedaan pembacaan yang muncul dipengaruhi oleh karakteristik sensor tegangan, toleransi komponen elektronik, serta proses konversi sinyal analog ke digital pada ESP32. Meskipun demikian, hasil pengujian menunjukkan bahwa nilai error rata-rata masih sangat kecil sehingga tidak memberikan pengaruh yang berarti terhadap kinerja sistem. Dengan tingkat akurasi tersebut, sensor tegangan dinilai mampu memberikan informasi kondisi catu daya secara andal dan dapat digunakan sebagai umpan balik untuk mendukung proses pengendalian pada sistem automarking.

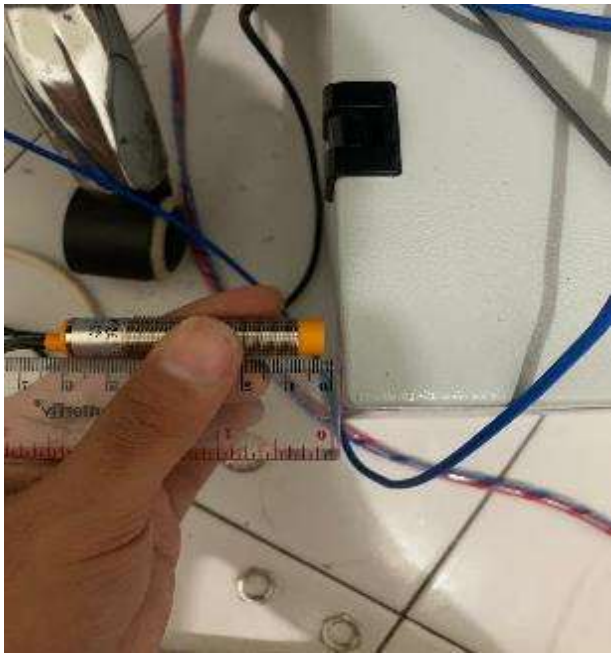
4.1.2 Pengujian Sensor Proximity

Pengujian sensor proximity dilakukan untuk memastikan bahwa sensor dapat mendeteksi objek logam dengan baik sesuai dengan kebutuhan sistem. Sensor ini memiliki peran penting karena digunakan sebagai pemicu utama dalam proses automarking, sehingga keakuratannya harus benar-benar

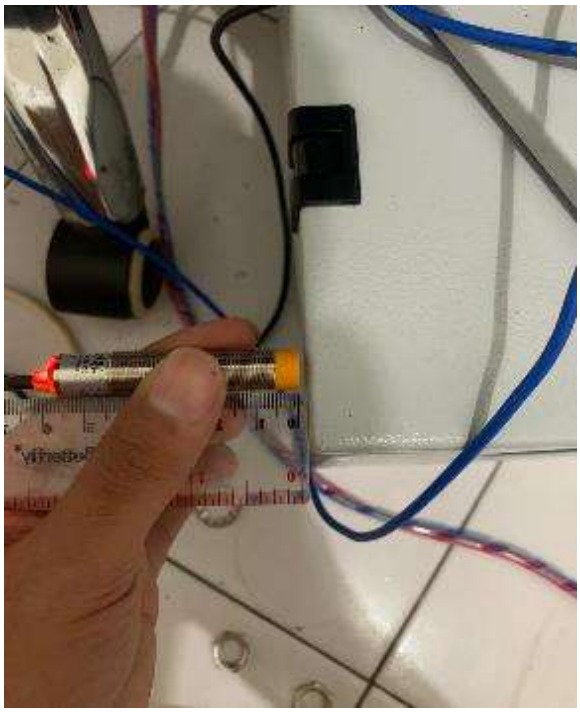
diperhatikan. Proses pengujian dilakukan dengan cara mendekatkan benda berbahan logam ke area deteksi sensor, kemudian diamati perubahan sinyal yang dihasilkan. Saat objek berada dalam jarak tertentu, sensor seharusnya memberikan sinyal aktif yang kemudian dibaca oleh ESP32 sebagai tanda bahwa objek terdeteksi seperti pada Gambar 4.3. Sebaliknya, ketika tidak ada objek, sensor akan kembali ke kondisi normal. Selain itu, dilakukan juga pengujian pada beberapa variasi jarak untuk mengetahui seberapa jauh sensor masih dapat mendeteksi objek dengan stabil seperti pada Gambar 4.4 dan Gambar 4.5. Dari pengujian yang menghasilkan data pada Tabel 4.2 bisa dilihat jarak efektif sensor dalam bekerja, serta apakah terdapat gangguan atau pembacaan yang tidak konsisten.



Gambar 4. 3 Tampilan ESP32 menerima sinyal dari proximity



Gambar 4. 4 Jarak proximity masih belum memenuhi



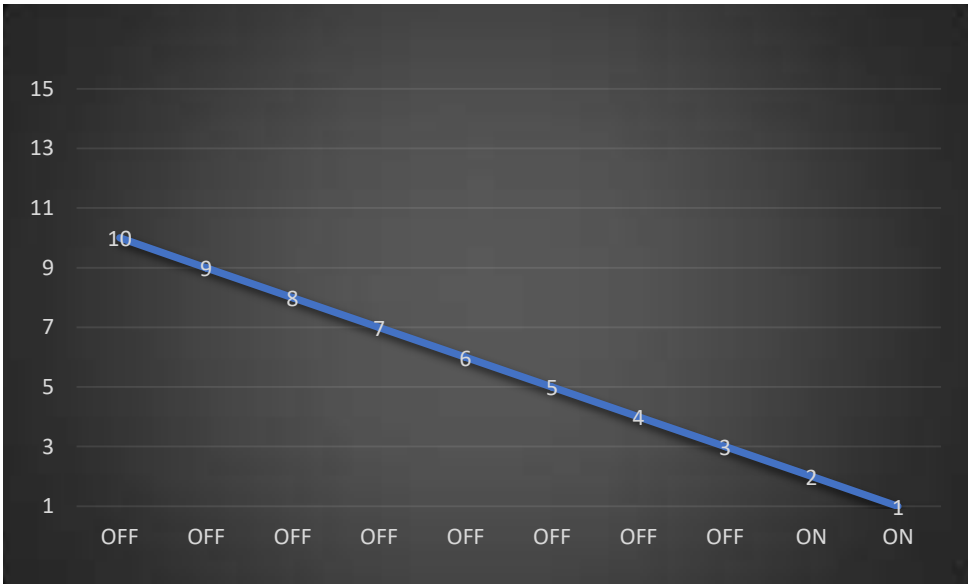
Gambar 4. 5 Jarak Proximity sudah memenuhi

Tabel 4. 2 Data Hasil Pengujian Sensor Proximity

NO	JARAK PROXIMITY DAN BARANG	HASIL
1	10 mm	OFF
2	9 mm	OFF

3	8 mm	OFF
4	7 mm	OFF
5	6 mm	OFF
6	5 mm	OFF
7	4 mm	OFF
8	3 mm	OFF
9	2 mm	ON
10	1 mm	ON

Jadi, dapat disimpulkan bahwa sensor proximity mampu mendeteksi barang dengan jarak maksimal 2mm.



Gambar 4. 6 Grafik Data Pengujian Sensor Proximity

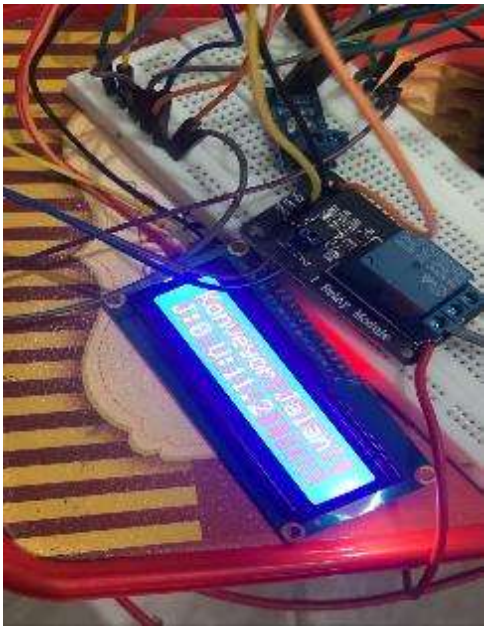
Gambar 4.6 memperlihatkan hubungan antara jarak objek terhadap sensor proximity dengan kondisi keluaran sensor selama proses pengujian. Berdasarkan grafik tersebut, sensor belum mampu mendeteksi objek pada jarak 10 mm hingga 3 mm sehingga keluaran sensor berada pada kondisi OFF. Ketika jarak objek diperkecil menjadi 2 mm dan 1 mm, keluaran sensor berubah menjadi ON, yang menunjukkan bahwa objek telah berhasil terdeteksi.

Hasil pengujian ini menunjukkan bahwa sensor proximity yang digunakan memiliki jarak deteksi efektif maksimum sekitar 2 mm untuk objek yang diuji. Pada jarak tersebut sensor dapat memberikan sinyal secara konsisten kepada ESP32 sebagai pemicu proses automarking. Oleh karena itu, pemasangan sensor pada sistem perlu memperhatikan posisi dan jarak terhadap objek agar proses pendeteksian berlangsung secara optimal dan mampu mengurangi kemungkinan terjadinya kegagalan deteksi selama konveyor beroperasi.

4.2 Pengujian LCD

Pengujian LCD dilakukan untuk memastikan bahwa layar yang digunakan dapat menampilkan informasi dengan jelas dan sesuai dengan data yang dikirim dari ESP32. LCD dalam sistem ini berfungsi sebagai media monitoring, sehingga sangat penting untuk memastikan tampilannya terbaca dengan baik oleh pengguna. Pada tahap pengujian, LCD dihubungkan ke ESP32 menggunakan komunikasi I2C, kemudian dilakukan pengiriman beberapa jenis data seperti teks, nilai tegangan, status sistem, serta jumlah barang yang sudah diproses. Tampilan tersebut diamati apakah muncul dengan benar, tidak terpotong, dan tidak terjadi delay yang mengganggu.

Selain itu, pengujian juga dilakukan untuk melihat kestabilan tampilan saat sistem berjalan secara terus-menerus. Hal ini penting karena LCD akan digunakan secara real-time selama proses automarking berlangsung. Jika tampilan sering berkedip atau tidak sinkron dengan kondisi sistem, maka akan menyulitkan dalam proses monitoring.

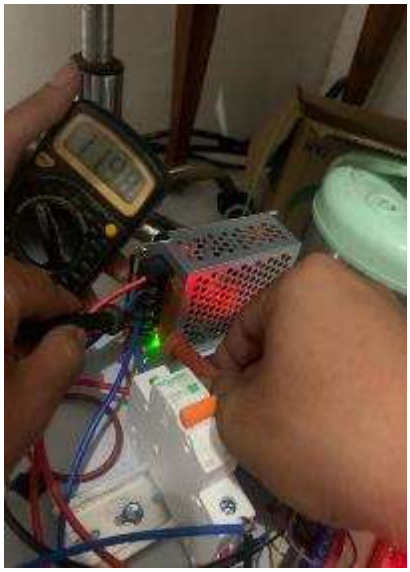


Gambar 4. 7 Tampilan LCD

Dari hasil pengujian yang dilakukan pada Gambar 4.7, LCD mampu menampilkan informasi dengan cukup jelas dan stabil. Data yang dikirim dari ESP32 dapat ditampilkan tanpa kendala yang berarti, baik berupa teks maupun angka. Dengan demikian, LCD dinilai sudah dapat digunakan sebagai media tampilan dalam sistem automarking untuk membantu memantau kondisi alat secara langsung.

4.3 Pengujian Power Supply

Pengujian power supply dilakukan untuk memastikan bahwa sumber tegangan yang digunakan dalam sistem benar-benar stabil dan sesuai dengan kebutuhan masing-masing komponen. Karena semua bagian sistem bergantung pada suplai tegangan ini, maka pengecekan di awal jadi hal yang cukup penting. Pengujian dimulai dengan menghubungkan power supply ke sumber listrik AC, kemudian mengukur tegangan output yang dihasilkan menggunakan multimeter. Nilai tegangan yang keluar diamati apakah sudah sesuai dengan spesifikasi, yaitu sekitar 12V. Berikut pada Gambar 4.8 untuk hasil dari pengujian parsial power supply.



Gambar 4. 8 Gambar Pengujian Tegangan output Power Supply

Dari hasil pengujian yang dilakukan, power supply mampu menghasilkan tegangan yang cukup stabil. Tegangan yang dihasilkan juga masih berada dalam batas yang aman untuk komponen yang digunakan. Dengan begitu, power supply ini dapat digunakan sebagai sumber utama dalam sistem automarking yang dirancang.

4.4 Pengujian buck converter

Pengujian buck converter dilakukan untuk memastikan bahwa modul penurun tegangan dapat bekerja dengan baik dalam mengubah tegangan dari 12V menjadi 5V sesuai kebutuhan sistem. Modul ini digunakan untuk menyuplai tegangan ke ESP32 dan beberapa komponen lain yang membutuhkan tegangan lebih rendah, sehingga kestabilannya cukup penting. Proses pengujian dimulai dengan menghubungkan input buck converter ke power supply 12V seperti pada Gambar 4.9, kemudian mengatur output menggunakan potensiometer hingga didapatkan tegangan sekitar 5V, seperti pada Gambar 4.10. Setelah itu, tegangan output diukur menggunakan multimeter untuk memastikan nilai yang dihasilkan sudah sesuai dan tidak melebihi batas aman untuk ESP32.

Gambar 4. 9 Tampilan input buck converter

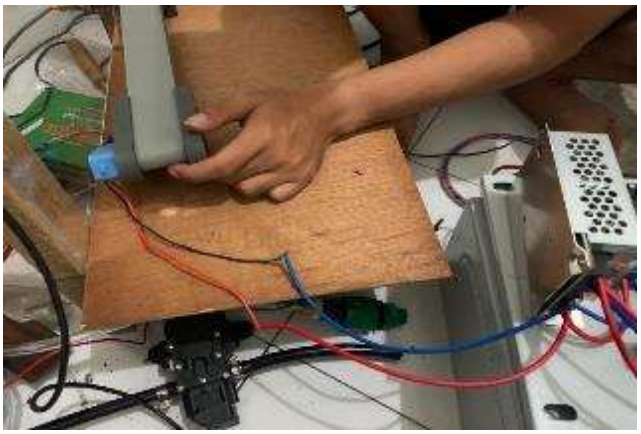
Gambar 4. 10 Tampilan output buck converter

Dari hasil pengujian yang dilakukan, buck converter mampu menurunkan tegangan dengan baik dan menghasilkan output yang cukup stabil di sekitar 5V. Saat diberikan beban, tegangan yang dihasilkan juga tidak mengalami penurunan yang signifikan. Dengan demikian, modul step-down ini dapat digunakan dengan aman sebagai penyedia tegangan untuk bagian kontrol pada sistem automarking.

4.5 Pengujian Motor DC konveyor

Pengujian motor DC dilakukan untuk memastikan bahwa motor yang

digunakan sebagai penggerak konveyor dapat bekerja dengan baik sesuai dengan perintah dari sistem. Motor ini memiliki peran penting karena berfungsi untuk menggerakkan benda menuju area proses marking, sehingga kinerjanya harus stabil dan mudah dikontrol. Berikut tampilan pengujian parsial motor DC off pada Gambar 4.11 serta motor DC on pada Gambar 4.12.



Gambar 4. 11 Tampilan Pengujian Parsial motor DC Off

Gambar 4. 12 Tampilan Pengujian Parsial motor DC On

Dari hasil pengujian yang dilakukan, motor DC dapat berputar dengan baik dan merespon perintah dari ESP32 dengan cukup cepat. Proses berhenti dan berjalan juga berlangsung dengan stabil, sehingga motor dinilai sudah layak digunakan sebagai penggerak konveyor dalam sistem automarking.

4.6 Pengujian Terintegrasi

Pengujian terintegrasi dilakukan setelah seluruh komponen selesai diuji secara terpisah dan dinyatakan bekerja dengan baik. Tujuan dari pengujian ini adalah untuk mengetahui apakah semua komponen dapat saling berkomunikasi

dan bekerja sebagai satu sistem yang utuh sesuai dengan alur kerja yang telah dirancang. Pada pengujian ini, sistem dijalankan mulai dari kondisi awal hingga proses automarking selesai. Saat alat diaktifkan, power supply akan memberikan tegangan ke seluruh rangkaian, kemudian ESP32 mulai menjalankan program yang telah diunggah. Motor DC akan menggerakkan konveyor sehingga benda dapat bergerak menuju area pendeteksian. Ketika obyek memasuki area sensor proximity, sensor akan mengirimkan sinyal ke ESP32 sebagai tanda bahwa objek telah terdeteksi. Setelah menerima sinyal tersebut, ESP32 akan menghentikan putaran motor konveyor melalui driver L298N sehingga posisi benda tepat berada di area penyemprotan. Selanjutnya, modul relay akan mengaktifkan pompa DC untuk menyemprotkan cat ke permukaan benda sesuai waktu yang telah ditentukan pada program. Setelah proses penyemprotan selesai, pompa akan berhenti dan motor konveyor kembali dijalankan sehingga benda dapat bergerak menuju proses berikutnya, seperti pada Gambar 4.13.

Gambar 4. 13 Pengujian Keseluruhan Alat

Selama sistem bekerja, sensor tegangan juga melakukan pembacaan secara terus-menerus terhadap tegangan yang masuk ke rangkaian. Nilai tegangan tersebut ditampilkan pada LCD sebagai informasi bagi pengguna. Selain itu, LCD juga menampilkan status kerja sistem, seperti kondisi konveyor serta jumlah benda yang telah berhasil diproses. Apabila sensor tegangan mendeteksi nilai tegangan berada di bawah batas minimum yang telah ditentukan, sistem akan

menghentikan proses kerja dan buzzer akan aktif sebagai tanda adanya gangguan pada sumber daya.

4.6.1 Pengujian Motor Konveyor

Pengujian kecepatan motor DC dilakukan untuk mengetahui pengaruh perubahan nilai Pulse Width Modulation (PWM) yang diberikan oleh ESP32 terhadap kecepatan putaran motor konveyor. Semakin besar nilai PWM yang diberikan, maka semakin tinggi kecepatan putaran motor yang dihasilkan. Proses pengujian dilakukan dengan memberikan beberapa variasi nilai PWM melalui program. Setiap nilai PWM diuji beberapa kali untuk memperoleh hasil yang lebih konsisten.

Tabel 4. 3 Data Pengujian Motor Konveyor

No	PWM	Kondisi Motor	Hasil
1	120	Lambat	Stabil
2	130	Lambat	Stabil
3	140	Lambat	Stabil
4	150	Lambat	Stabil
5	160	Lambat	Stabil
6	170	Cepat	Resiko miss deteksi
7	180	Cepat	Resiko miss deteksi
8	200	Cepat	Resiko miss deteksi
9	235	Sangat Cepat	Miss deteksi
10	255	Sangat Cepat	Miss deteksi

Hasil pengujian Tabel 4.3 menunjukkan bahwa kecepatan motor meningkat seiring bertambahnya nilai PWM yang diberikan oleh ESP32. Pada nilai PWM yang rendah, motor berputar lebih lambat sehingga konveyor bergerak dengan kecepatan rendah. Sebaliknya, ketika nilai PWM dinaikkan, putaran motor menjadi lebih cepat dan pergerakan konveyor juga meningkat. Kondisi tersebut menunjukkan bahwa driver

L298N mampu menerima sinyal PWM dari ESP32 dengan baik dan mengendalikan motor sesuai nilai yang telah ditentukan.

Pengujian ini dilakukan untuk menentukan nilai PWM yang paling sesuai digunakan pada sistem automarking. Nilai PWM yang dipilih harus mampu menghasilkan kecepatan konveyor yang stabil sehingga sensor proximity memiliki waktu yang cukup untuk mendeteksi benda, serta proses penyemprotan cat dapat berlangsung dengan tepat pada posisi yang telah ditentukan. Berdasarkan data tersebut kecepatan PWM yang digunakan adalah 130.

4.6.2 Pengujian Sensor Tegangan

Pengujian terintegrasi sensor tegangan dilakukan untuk mengetahui apakah sensor tegangan dapat bekerja dengan baik saat dihubungkan dengan seluruh komponen pada sistem automarking. Pengujian ini tidak hanya bertujuan untuk melihat kemampuan sensor dalam membaca nilai tegangan, tetapi juga untuk memastikan bahwa data yang diperoleh dapat digunakan oleh ESP32 sebagai umpan balik dalam mengendalikan sistem ketika terjadi penurunan tegangan. Pengujian dimulai dengan mengaktifkan seluruh rangkaian sehingga sistem bekerja dalam kondisi normal. Sensor tegangan akan membaca nilai tegangan yang masuk dari sumber catu daya, kemudian hasil pembacaan tersebut dikirim ke ESP32 untuk ditampilkan pada LCD. Selain pengujian pada kondisi normal, dilakukan juga simulasi penurunan tegangan dengan mengubah nilai tegangan masukan hingga berada di bawah batas minimum yang telah ditentukan pada program. Ketika kondisi tersebut terjadi, sensor tegangan akan mengirimkan informasi ke ESP32, kemudian sistem secara otomatis menghentikan putaran motor konveyor sebagai langkah pengamanan. Pada saat yang sama, buzzer akan aktif sebagai indikator adanya gangguan pada sumber daya, sedangkan LCD menampilkan pesan peringatan beserta nilai tegangan yang terbaca. Pengujian ini dilakukan untuk memastikan bahwa

seluruh komponen dapat merespon perubahan tegangan secara bersamaan sesuai dengan logika program yang telah dibuat.

Tabel 4. 4 Data Pengujian Sensor Tegangan

No	Tegangan (V)	Konveyor	Pompa	Buzzer
1	11 V	Berjalan	Nyala	Mati
2	10 V	Berjalan	Nyala	Mati
3	9 V	Berjalan	Nyala	Mati
4	8 V	Berjalan	Nyala	Mati
5	7 V	Berjalan	Nyala	Mati
6	6 V	Berjalan	Nyala	Mati
7	5V	Berjalan	Nyala	Mati
8	4 V	Berhenti	Mati	Bunyi
9	3 V	Berhenti	Mati	Bunyi
10	0 V	Berhenti	Mati	Bunyi

Berdasarkan hasil pengujian Tabel 4.4 sensor tegangan mampu bekerja dengan baik saat diintegrasikan dengan sistem automarking. Nilai tegangan yang terbaca dapat ditampilkan secara real-time pada LCD dan respon sistem terhadap kondisi tegangan rendah berjalan sesuai dengan yang diharapkan. Motor konveyor berhenti secara otomatis, buzzer memberikan peringatan, dan informasi kondisi tegangan dapat langsung diketahui oleh pengguna melalui LCD. Hasil tersebut menunjukkan bahwa sensor tegangan tidak hanya berfungsi sebagai alat monitoring, tetapi juga berperan sebagai sistem proteksi untuk menjaga agar proses automarking tetap berjalan dengan aman.

4.6.3 Pengujian Dinamo Sprayer

Pengujian waktu penyemprotan pompa dilakukan untuk mengetahui kesesuaian antara durasi penyemprotan yang telah diprogram pada ESP32 dengan waktu kerja pompa saat proses marking berlangsung, seperti pada Gambar 4.14. Pada penelitian ini, waktu penyemprotan ditetapkan sebesar 0,20 detik. Durasi tersebut dipilih berdasarkan hasil

pengamatan awal yang menunjukkan bahwa waktu tersebut mampu menghasilkan semprotan cat yang cukup untuk memberikan tanda pada permukaan benda kerja tanpa menyebabkan penumpukan cat.

Gambar 4. 14 Pengujian Waktu Penyemprotan

Proses pengujian dilakukan dengan menjalankan sistem secara berulang menggunakan kondisi yang sama. Ketika sensor proximity mendeteksi adanya objek, ESP32 mengaktifkan relay sehingga pompa menyala selama 200ms atau 0,20 detik sesuai dengan nilai yang telah ditentukan dalam program. Setelah waktu tersebut tercapai, relay kembali memutus arus listrik ke pompa sehingga proses penyemprotan berhenti secara otomatis. Pengujian dilakukan sebanyak 10 kali seperti pada Tabel 4.5 untuk memastikan bahwa durasi penyemprotan tetap konsisten pada setiap siklus kerja. Selama pengujian berlangsung, waktu aktif pompa diamati menggunakan stopwatch dan dibandingkan dengan nilai yang telah diprogram.

Tabel 4. 5 Data Pengujian Waktu Penyemprotan

No	Stopwatch	Program	Hasil
1	0,19 detik	0,20 detik	Terdapat Selisih 0,01 detik antara stopwatch dan program.
2	0,19 detik	0,20 detik	Terdapat Selisih 0,01 detik

			antara stopwatch dan program.
3	0,19 detik	0,20 detik	Terdapat Selisih 0,01 detik antara stopwatch dan program.
4	0,19 detik	0,20 detik	Terdapat Selisih 0,01 detik antara stopwatch dan program.
5	0,19 detik	0,20 detik	Terdapat Selisih 0,01 detik antara stopwatch dan program.
6	0,19 detik	0,20 detik	Terdapat Selisih 0,01 detik antara stopwatch dan program.
7	0,19 detik	0,20 detik	Terdapat Selisih 0,01 detik antara stopwatch dan program.
8	0,19 detik	0,20 detik	Terdapat Selisih 0,01 detik antara stopwatch dan program.
9	0,19 detik	0,20 detik	Terdapat Selisih 0,01 detik antara stopwatch dan program.
10	0,19 detik	0,20 detik	Terdapat Selisih 0,01 detik antara stopwatch dan program.

Durasi penyemprotan sebesar 0,20 detik menghasilkan volume cat yang cukup untuk membentuk tanda pada permukaan objek tanpa menimbulkan cat yang berlebihan. Selain itu, pompa dapat berhenti tepat setelah waktu yang ditentukan sehingga tidak terjadi pemborosan cairan marking. Hal ini menunjukkan bahwa pengaturan waktu penyemprotan pada program ESP32 telah bekerja dengan baik dan mampu mengendalikan pompa secara stabil. Hasil pengujian menunjukkan bahwa pompa mampu bekerja sesuai dengan target waktu 0,20 detik dengan penyimpangan yang sangat kecil sebesar 0,01 detik sehingga tidak memengaruhi proses marking secara keseluruhan.

4.6.4 Pengujian Web Monitoring Sistem

Pengujian web monitoring dilakukan untuk mengetahui apakah data yang dikirim oleh ESP32 dapat ditampilkan pada halaman web dengan baik. Sebelum pengujian dimulai, ESP32 dihubungkan ke jaringan WiFi sehingga komunikasi antara perangkat dan web monitoring dapat berjalan dengan normal. Setelah koneksi berhasil dilakukan, halaman web dapat diakses menggunakan perangkat yang berada pada jaringan yang sama. Pada saat sistem dijalankan, setiap perubahan kondisi pada alat akan langsung dikirimkan oleh ESP32 ke halaman web. Informasi yang ditampilkan meliputi nilai tegangan yang terbaca oleh sensor tegangan, jumlah benda yang telah diproses. Selama proses pengujian, seluruh data dapat ditampilkan sesuai dengan kondisi sebenarnya pada system. Selain itu, jumlah benda yang telah diproses juga bertambah secara otomatis setiap kali proses marking selesai dilakukan.

Simulasi juga dilakukan pada kondisi ketika tegangan berada di bawah batas yang telah ditentukan. Pada kondisi tersebut, halaman web menampilkan nilai tegangan yang terbaca, sedangkan sistem menghentikan kerja motor konveyor, mematikan pompa sprayer, dan mengaktifkan buzzer sebagai tanda adanya gangguan pada sumber tegangan.

BAB V
KESIMPULAN

5.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan pengujian yang telah dilakukan, dapat diambil beberapa kesimpulan sebagai berikut.

1. Sistem automarking berbasis ESP32 berhasil dirancang dan direalisasikan sesuai dengan perencanaan. Sistem mampu bekerja secara otomatis dengan mengintegrasikan sensor proximity, sensor tegangan, motor DC konveyor, pompa sprayer, buzzer, LCD, dan web monitoring sehingga proses penandaan dapat berlangsung tanpa campur tangan operator.
2. Sensor proximity dan sensor tegangan berhasil diterapkan sebagai masukan sistem. Sensor proximity mampu mendeteksi objek logam secara konsisten pada jarak maksimal 2 mm sebagai pemicu proses marking, sedangkan sensor tegangan mampu memantau kondisi catu daya dengan rata-rata kesalahan pengukuran sebesar 0,007% sehingga dapat digunakan sebagai umpan balik untuk menjaga kestabilan dan keamanan sistem.
3. Pengendalian konveyor dan proses marking berjalan sesuai dengan kondisi sistem. Motor DC konveyor dapat dikendalikan menggunakan metode PWM dengan nilai optimal sebesar 130 sehingga proses deteksi dan penyemprotan berlangsung stabil. Sistem juga mampu menghentikan konveyor, mengaktifkan pompa penyemprot saat objek terdeteksi, serta menjalankan fungsi proteksi dengan menghentikan aktuator dan mengaktifkan buzzer ketika tegangan berada di bawah batas yang telah ditentukan. Selain itu, kondisi sistem dapat dipantau secara real-time melalui web monitoring.

5.2 Saran

Dari pengerjaan Tugas Akhir yang telah dilakukan terdapat beberapa kekurangan dan kelemahan pada sistem. Oleh karena itu penulis memberikan saran apabila dapat dikembangkan kembali terkait pembahasan alat automarking ini. Saran yang diberikan yaitu.

1. Menambahkan fitur penyimpanan data berbasis database atau cloud

sehingga riwayat jumlah produksi, nilai tegangan, dan kondisi sistem dapat direkam secara otomatis.

2. Mengembangkan web monitoring menjadi sistem kendali jarak jauh yang memungkinkan operator mengatur parameter seperti nilai PWM, waktu penyemprotan, maupun proses start dan stop konveyor melalui antarmuka web.
3. Menambahkan sensor pendukung, seperti sensor posisi atau kamera, agar proses penentuan titik marking menjadi lebih akurat pada berbagai bentuk dan ukuran objek.
4. Menggunakan nozzle dengan spesifikasi yang lebih sesuai atau menambahkan pengatur tekanan cairan sehingga hasil penyemprotan menjadi lebih seragam dan penggunaan cairan marking lebih efisien.
5. Mengembangkan sistem agar mampu mendeteksi berbagai jenis material, tidak hanya logam, sehingga alat dapat diterapkan pada aplikasi industri yang lebih luas.
6. Menambahkan sistem notifikasi melalui perangkat seluler ketika tegangan berada di luar batas yang telah ditentukan atau ketika terjadi gangguan pada proses automarking sehingga operator dapat segera melakukan tindakan.

DAFTAR PUSTAKA

Bolton, W. (2015). *Mechatronics: Electronic Control Systems in Mechanical and Electrical Engineering* (6th ed.). Pearson Education Limited.

Espressif Systems. (2023). *ESP32 Series Datasheet*. Espressif Systems Official Documentation.https://www.espressif.com

Fraden, J. (2016). *Handbook of Modern Sensors: Physics, Designs, and Applications* (5th ed.). Springer.

Groover, M. P. (2016). *Automation, Production Systems, and Computer-Integrated Manufacturing* (4th ed.). Pearson Education.

Hughes, A., & Drury, B. (2019). *Electric Motors and Drives: Fundamentals, Types and Applications* (5th ed.). Butterworth-Heinemann.

Johnson, C. D. (2014). *Process Control Instrumentation Technology* (8th ed.). Pearson Education.

Monk, S. (2020). *Programming Arduino: Getting Started with Sketches* (2nd ed.). McGraw-Hill Education.

Oladapo, A. A., Akinwale, O. S., & Adeyemi, O. A. (2016). Model design and simulation of automatic sorting machine using proximity sensor. *Journal of Engineering Science and Technology (JESTECH)*, 11(4), 512–523.

Sharma, P., Verma, A., & Singh, R. (2021). IoT-based smart conveyor system for integrated process control. *Journal of Industrial Automation and Smart Manufacturing*, 5(2), 101–110.

Lee, J., Park, S., & Kim, H. (2020). Design and implementation of automated marking and sorting system using microcontroller. *International Journal of Advanced Manufacturing Technology*, 109(3–4), 845–856. https://doi.org/10.1007/s00170-020-05621-4

Patel, R., Shah, K., & Mehta, D. (2022). Automation of industrial conveyor system using embedded controller. *International Journal of Electrical and Computer Engineering*, 12(2), 1560–1568. https://doi.org/10.11591/ijece.v12i2.pp1560-1568

Arduino. (2024). *Arduino IDE Documentation*. https://docs.arduino.cc

Autodesk. (2024). *AutoCAD User Guide*. https://www.autodesk.com

Espressif Systems. (2024). *ESP32 Series Datasheet*. https://www.espressif.com

Kadir, A. (2018). *Pemrograman Arduino dan Processing*. Jakarta: PT Elex Media Komputindo.

Miranty, & Nasriadi, A. (2024). *Espduino-32 Utilization in the Warehouse Storage Information System of the Metal Processing Industry Department*.

Pratama, I. P. A. E. (2020). *Internet of Things: Konsep dan Implementasi*. Bandung: Informatika.

Putra, D. A. (2021). *Mikrokontroler ESP32 untuk Aplikasi Internet of Things*. Yogyakarta: Andi Offset.

Setiawan, R. (2022). *Sistem Kontrol Motor DC Menggunakan Metode PWM Berbasis Mikrokontroler*. Jurnal Teknik Elektro, 11(2), 101-109.

Siswanto, E. (2021). *Sensor dan Aktuator pada Sistem Otomasi Industri*. Yogyakarta: Deepublish.

Texas Instruments. (2024). *BTS7960 High Current Motor Driver Datasheet*. https://www.ti.com

The MathWorks, Inc. (2023). *Pulse Width Modulation (PWM) Fundamentals*. https://www.mathworks.com

Tjahjono, H. (2020). *Otomasi Industri Berbasis Mikrokontroler*. Jakarta: PT Gramedia.

Utomo, B., & Nugroho, A. (2022). Implementasi ESP32 sebagai Web Server untuk Monitoring Sistem Berbasis Internet of Things. *Jurnal Teknologi Informasi dan Komputer*, 8(1), 55-63.

Widodo, S. (2021). *Sistem Kendali Motor Listrik*. Yogyakarta: Graha Ilmu.

Yuliana, T., & Hidayat, R. (2023). Perancangan Sistem Monitoring Tegangan DC Menggunakan ESP32 Berbasis Web. *Jurnal Elektronika dan Instrumentasi*, 15(2), 88-97.

(Halaman ini sengaja dikosongkan)

LAMPIRAN

Lampiran Kode Program Arduino

```
#define PROXIMITY_PIN  15
#define RPWM           26
#define LPWM           33
#define R_EN           14
#define L_EN           12
#define RELAY_POMPA    13
#define BUZZER         4

//=================================================================
=========
// VARIABLE & CONSTANTS
//=================================================================
=========
// WiFi Credentials (Silakan sesuaikan dengan jaringan Anda
jika diperlukan)
const char* ssid     = "Automarking_AP";
const char* password = "PasswordAutomarking";

// System Configuration
const float teganganSupply = 11.45;
const int pwmNormal = 130;

// Operational Variables
float teganganMotor = 0.0;
int pwmMotor = 0;
int jumlahBarang = 0;
String statusSistem = "BOOTING";

// Relay and Buzzer Electrical States (Relay: Active LOW,
Buzzer: Active HIGH)
bool relayActive = false;
bool buzzerActive = false;

// Debounce and Filter Timers
unsigned long lastDebounceTime = 0;
const unsigned long debounceDelay = 30;
bool lastSensorState = HIGH;
bool sensorStableState = HIGH;
bool barangSudahLewat = true;

// LEDC PWM Configurations
const int channelMaju = 0;
const int channelMundur = 1;
const int pwmFreq = 5000;
const int pwmResolution = 8;

// System States
enum EngineState {
  STATE_NORMAL,
  STATE_MARKING,
  STATE_LOW_VOLTAGE
```

```
};
EngineState currentState = STATE_NORMAL;

// Marking State Timers
unsigned long markingStartTime = 0;
const unsigned long waktuSemprot = 200;

// LCD Telemetry State Refresh Control
unsigned long lastLcdUpdate = 0;
const unsigned long lcdRefreshInterval = 200;
int lastLcdPrintedJumlah = -1;
float lastLcdPrintedTegangan = -1.0;
String lastLcdPrintedStatus = "";

//================================================================
=========
// LCD MODULE
//================================================================
=========
LiquidCrystal_I2C lcd(0x27, 16, 2);

void initLCD() {
  lcd.init();
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("  AUTOMARKING   ");
  lcd.setCursor(0, 1);
  lcd.print("  SYSTEM READY  ");
}

void updateLCDTelemetry() {
  unsigned long currentMillis = millis();
  if (currentMillis - lastLcdUpdate < lcdRefreshInterval)
return;
  lastLcdUpdate = currentMillis;

  if (currentState == STATE_LOW_VOLTAGE) {
    if (lastLcdPrintedStatus != "LOW") {
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("TEGANGAN RENDAH ");
      lcd.setCursor(0, 1);
      lcd.print("V: ");
      lcd.print(teganganMotor, 2);
      lcd.print("V Lock");
      lastLcdPrintedStatus = "LOW";
    }
    return;
  }

  if (currentState == STATE_MARKING) {
    if (lastLcdPrintedStatus != "MARKING") {
      lcd.setCursor(0, 0);
      lcd.print("Marking Proses  ");
      lastLcdPrintedStatus = "MARKING";
    }
  } else if (currentState == STATE_NORMAL) {
    if (lastLcdPrintedStatus != "NORMAL") {
```

```
      lcd.setCursor(0, 0);
      lcd.print("Konveyor Jalan  ");
      lastLcdPrintedStatus = "NORMAL";
    }
  }

  // Update dynamic numeric variables dynamically to minimize
visual flicker
  if (jumlahBarang != lastLcdPrintedJumlah || abs(teganganMotor
- lastLcdPrintedTegangan) >= 0.1) {
    lcd.setCursor(0, 1);
    char buffer[17];
    snprintf(buffer, sizeof(buffer), "J:%-4d  V:%-5.2fV",
jumlahBarang, teganganMotor);
    lcd.print(buffer);
    lastLcdPrintedJumlah = jumlahBarang;
    lastLcdPrintedTegangan = teganganMotor;
  }
}

//================================================================
=========
// MOTOR CONTROL
//================================================================
=========
void driveMotor(int targetPwm) {
  if (currentState == STATE_LOW_VOLTAGE) {
    pwmMotor = 0;
    ledcWrite(channelMaju, 0);
    ledcWrite(channelMundur, 0);
    return;
  }

  pwmMotor = targetPwm;
  ledcWrite(channelMundur, 0);
  ledcWrite(channelMaju, pwmMotor);
}

void stopMotor() {
  pwmMotor = 0;
  ledcWrite(channelMaju, 0);
  ledcWrite(channelMundur, 0);
}

void calculateVoltage() {
  teganganMotor = (teganganSupply * (float)pwmMotor) / 255.0;
}

//================================================================
=========
// SIMULATION ENGINE
//================================================================
=========
enum SimStep {
  SIM_IDLE,
  SIM_T1_RAMP_UP,
  SIM_T2_HOLD,
  SIM_T3_RAMP_DOWN
```

```
};
SimStep currentSimStep = SIM_IDLE;

bool simulationActive = false;
unsigned long simStepStartTime = 0;

void startSimulation() {
  if (currentState == STATE_LOW_VOLTAGE) return;
  simulationActive = true;
  currentSimStep = SIM_T1_RAMP_UP;
  simStepStartTime = millis();
  statusSistem = "SIM RAMP UP";
}

void resetSimulation() {
  simulationActive = false;
  currentSimStep = SIM_IDLE;
  if (currentState != STATE_LOW_VOLTAGE) {
    currentState = STATE_NORMAL;
    statusSistem = "NORMAL";
    driveMotor(pwmNormal);
  }
}

void runSimulationMachine() {
  if (!simulationActive || currentState == STATE_LOW_VOLTAGE)
return;

  unsigned long elapsed = millis() - simStepStartTime;

  switch (currentSimStep) {
    case SIM_T1_RAMP_UP:
      if (elapsed <= 2000) {
        int calculatedPwm = pwmNormal + (int)(((255 -
pwmNormal) * elapsed) / 2000);
        driveMotor(calculatedPwm);
      } else {
        currentSimStep = SIM_T2_HOLD;
        simStepStartTime = millis();
        statusSistem = "SIM HOLD";
        driveMotor(255);
      }
      break;

    case SIM_T2_HOLD:
      if (elapsed <= 5000) {
        driveMotor(255);
      } else {
        currentSimStep = SIM_T3_RAMP_DOWN;
        simStepStartTime = millis();
        statusSistem = "SIM RAMP DOWN";
      }
      break;

    case SIM_T3_RAMP_DOWN:
      if (elapsed <= 5000) {
        int calculatedPwm = 255 - (int)(((255 - 100) * elapsed)
/ 5000);
```

```
        driveMotor(calculatedPwm);
      } else {
        resetSimulation();
      }
      break;

    default:
      break;
  }
}

//================================================================
=========
// WEB SERVER & DASHBOARD STORAGE (PROGMEM HTML/CSS/JS)
//================================================================
=========
WebServer server(80);

const char PAGE_Dashboard[] PROGMEM = R"rawliteral(
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Automarking System Telemetry</title>
    <style>
        :root {
            --bg-primary: #0f172a;
            --bg-secondary: #1e293b;
            --text-main: #f8fafc;
            --text-muted: #94a3b8;
            --accent-green: #10b981;
            --accent-yellow: #f59e0b;
            --accent-red: #ef4444;
            --accent-blue: #3b82f6;
            --border-color: #334155;
        }
        * { box-sizing: border-box; margin: 0; padding: 0;
font-family: 'Segoe UI', Roboto, sans-serif; }
        body { background-color: var(--bg-primary); color:
var(--text-main); min-height: 100vh; padding: 2rem 1rem;
transition: background 0.3s ease; }
        body.low-voltage { --bg-primary: #450a0a; --bg-
secondary: #7f1d1d; --border-color: #b91c1c; }
        .wrapper { max-width: 1200px; margin: 0 auto; }
        header { text-align: center; margin-bottom: 2.5rem; }
        header h1 { font-size: 2.5rem; font-weight: 800;
letter-spacing: -0.05em; text-transform: uppercase; color:
var(--text-main); margin-bottom: 0.5rem; }
        header p { color: var(--text-muted); font-size: 1rem; }
        .grid { display: grid; grid-template-columns:
repeat(auto-fit, minmax(280px, 1fr)); gap: 1.5rem; margin-
bottom: 2rem; }
        .card { background-color: var(--bg-secondary); border:
1px solid var(--border-color); border-radius: 1rem; padding:
1.5rem; position: relative; overflow: hidden; display: flex;
flex-direction: column; justify-content: space-between;
```

```
transition: all 0.2s ease; }
        .card-title { font-size: 0.875rem; font-weight: 600;
text-transform: uppercase; letter-spacing: 0.05em; color: var(-
-text-muted); margin-bottom: 1rem; }
        .card-value { font-size: 2.25rem; font-weight: 700;
line-height: 1; margin-bottom: 0.5rem; }
        .status-badge { display: inline-block; padding: 0.5rem
1rem; border-radius: 2rem; font-weight: 700; font-size:
1.125rem; text-align: center; width: 100%; text-transform:
uppercase; }
        .bg-green { background-color: rgba(16, 185, 129, 0.2);
color: var(--accent-green); border: 1px solid var(--accent-
green); }
        .bg-yellow { background-color: rgba(245, 158, 11, 0.2);
color: var(--accent-yellow); border: 1px solid var(--accent-
yellow); }
        .bg-red { background-color: rgba(239, 68, 68, 0.2);
color: var(--accent-red); border: 1px solid var(--accent-red);
}
        .progress-container { width: 100%; background-color:
var(--border-color); height: 0.75rem; border-radius: 1rem;
overflow: hidden; margin-top: 0.5rem; }
        .progress-bar { height: 100%; width: 0%; background-
color: var(--accent-blue); transition: width 0.1s linear; }
        .progress-bar.voltage-bar { background-color: var(--
accent-green); }
        .state-indicator { display: flex; align-items: center;
gap: 0.7rem; font-size: 1.5rem; font-weight: 700; }
        .dot { width: 1rem; height: 1rem; border-radius: 50%;
background-color: var(--text-muted); }
        .dot.active { animation: pulse 1s infinite alternate; }
        .dot.active-pump { background-color: var(--accent-
blue); box-shadow: 0 0 8px var(--accent-blue); }
        .dot.active-buzzer { background-color: var(--accent-
red); box-shadow: 0 0 8px var(--accent-red); }
        .control-panel { background-color: var(--bg-secondary);
border: 1px solid var(--border-color); border-radius: 1rem;
padding: 2rem; display: flex; flex-wrap: wrap; gap: 1rem;
justify-content: center; align-items: center; }
        .btn { padding: 0.75rem 2rem; font-size: 1rem; font-
weight: 700; border-radius: 0.5rem; border: none; cursor:
pointer; text-transform: uppercase; transition: all 0.2s ease;
width: 100%; max-width: 250px; }
        .btn-start { background-color: var(--accent-blue);
color: #fff; box-shadow: 0 4px 6px -1px rgba(59, 130, 246,
0.3); }
        .btn-start:hover:not(:disabled) { background-color:
#2563eb; }
        .btn-reset { background-color: rgba(148, 163, 184,
0.2); color: var(--text-main); border: 1px solid var(--border-
color); }
        .btn-reset:hover:not(:disabled) { background-color:
rgba(148, 163, 184, 0.3); }
        .btn:disabled { opacity: 0.4; cursor: not-allowed; }
        @keyframes pulse { from { opacity: 0.4; } to { opacity:
1; } }
        @media(min-width: 768px) { .btn { width: auto; } }
    </style>
```

```
</head>
<body>
    <div class="wrapper">
        <header>
            <h1>Automarking System</h1>
            <p>Realtime HMI & Hardware Telemetry Interface</p>
        </header>

        <div class="grid">
            <div class="card">
                <div class="card-title">System Operational
Status</div>
                <div id="status-card" class="status-badge bg-
green">BOOTING</div>
            </div>

            <div class="card">
                <div class="card-title">Motor Terminal
Voltage</div>
                <div id="voltage-value" class="card-value">0.00
V</div>
                <div class="progress-container">
                    <div id="voltage-progress" class="progress-
bar voltage-bar"></div>
                </div>
            </div>

            <div class="card">
                <div class="card-title">PWM Duty Cycle
Input</div>
                <div id="pwm-value" class="card-value">0 /
255</div>
                <div class="progress-container">
                    <div id="pwm-progress" class="progress-
bar"></div>
                </div>
            </div>

            <div class="card">
                <div class="card-title">Processed Items
Count</div>
                <div id="items-value" class="card-value"
style="color:var(--accent-green)">0</div>
            </div>

            <div class="card">
                <div class="card-title">Actuator Stack
Status</div>
                <div style="display:flex; flex-
direction:column; gap: 0.5rem;">
                    <div class="state-indicator">
                        <div id="pump-dot" class="dot"></div>
                        <span style="font-size:1rem;
color:var(--text-muted)">Marking Pump</span>
                    </div>
                    <div class="state-indicator">
                        <div id="buzzer-dot" class="dot"></div>
                        <span style="font-size:1rem;
```

```
color:var(--text-muted)">Alarm Buzzer</span>
                    </div>
                </div>
            </div>
        </div>

        <div class="control-panel">
            <button id="btn-start" class="btn btn-start"
onclick="triggerSimulation()">Start Simulation</button>
            <button id="btn-reset" class="btn btn-reset"
onclick="triggerReset()">Reset System</button>
        </div>
    </div>

    <script>
        function updateDashboard() {
            fetch('/status')
                .then(response => response.json())
                .then(data => {
                    const statusCard =
document.getElementById('status-card');
                    statusCard.innerText = data.status;

                    statusCard.className = "status-badge";
                    if(data.status.includes("LOW VOLTAGE") ||
data.status.includes("FAULT")) {
                        statusCard.classList.add("bg-red");
                        document.body.classList.add("low-
voltage");
                        document.getElementById('btn-
start').disabled = true;
                    } else if(data.status.includes("SIM") ||
data.status.includes("MARKING")) {
                        statusCard.classList.add("bg-yellow");
                        document.body.classList.remove("low-
voltage");
                        document.getElementById('btn-
start').disabled = true;
                    } else {
                        statusCard.classList.add("bg-green");
                        document.body.classList.remove("low-
voltage");
                        document.getElementById('btn-
start').disabled = false;
                    }

                    document.getElementById('items-
value').innerText = data.jumlah;
                    document.getElementById('pwm-
value').innerText = data.pwm + " / 255";

                    let pwmPct = (data.pwm / 255) * 100;
                    document.getElementById('pwm-
progress').style.width = pwmPct + "%";

                    document.getElementById('voltage-
value').innerText = data.tegangan.toFixed(2) + " V";
                    let voltPct = (data.tegangan / 12.0) * 100;
```

```
                    document.getElementById('voltage-
progress').style.width = Math.min(voltPct, 100) + "%";

                    const pumpDot =
document.getElementById('pump-dot');
                    if(data.relay) {
                        pumpDot.className = "dot active active-
pump";
                    } else {
                        pumpDot.className = "dot";
                    }

                    const buzzerDot =
document.getElementById('buzzer-dot');
                    if(data.buzzer) {
                        buzzerDot.className = "dot active
active-buzzer";
                    } else {
                        buzzerDot.className = "dot";
                    }
                })
                .catch(err => console.error("Telemetry Endpoint
Fetch Error: ", err));
        }

        function triggerSimulation() {
            fetch('/start-sim', { method: 'POST' });
        }

        function triggerReset() {
            fetch('/reset-sim', { method: 'POST' });
        }

        setInterval(updateDashboard, 100);
    </script>
</body>
</html>
)rawliteral";

//================================================================
=========
// JSON TELEMETRY ENDPOINT
//================================================================
=========
void handleStatusEndpoint() {
  String json = "{";
  json += "\"tegangan\":" + String(teganganMotor, 2) + ",";
  json += "\"pwm\":" + String(pwmMotor) + ",";
  json += "\"jumlah\":" + String(jumlahBarang) + ",";
  json += "\"status\":\"" + statusSistem + "\",";
  json += "\"relay\":" + String(relayActive ? "true" : "false")
+ ",";
  json += "\"buzzer\":" + String(buzzerActive ? "true" :
"false") + ",";
  json += "\"simulasi\":" + String(simulationActive ? "true" :
"false");
  json += "}";
  server.send(200, "application/json", json);
```

```
}

void handleStartSimEndpoint() {
  if (currentState != STATE_LOW_VOLTAGE) {
    startSimulation();
    server.send(200, "text/plain", "OK");
  } else {
    server.send(403, "text/plain", "LOCKED_LOW_VOLTAGE");
  }
}

void handleResetSimEndpoint() {
  if (currentState == STATE_LOW_VOLTAGE) {
    currentState = STATE_NORMAL;
    jumlahBarang = 0;
  }
  resetSimulation();
  server.send(200, "text/plain", "OK");
}

void handleRootDashboard() {
  server.send_P(200, "text/html", PAGE_Dashboard);
}

void setupWebServer() {
  server.on("/", HTTP_GET, handleRootDashboard);
  server.on("/status", HTTP_GET, handleStatusEndpoint);
  server.on("/start-sim", HTTP_POST, handleStartSimEndpoint);
  server.on("/reset-sim", HTTP_POST, handleResetSimEndpoint);
  server.begin();
}

//=============================================================
==========
// SETUP
//=============================================================
==========
void setup() {
  Serial.begin(115200);
  delay(500);


Serial.println("\n========================================");
  Serial.println("[SYSTEM] SYSTEM INIT - AUTOMARKING ESP32");
  Serial.println("========================================");

  pinMode(PROXIMITY_PIN, INPUT_PULLUP);
  pinMode(RELAY_POMPA, OUTPUT);
  pinMode(BUZZER, OUTPUT);
  pinMode(R_EN, OUTPUT);
  pinMode(L_EN, OUTPUT);

  digitalWrite(RELAY_POMPA, HIGH);
  digitalWrite(BUZZER, LOW);
  digitalWrite(R_EN, HIGH);
  digitalWrite(L_EN, HIGH);

  Wire.begin();
```

```
  Wire.setClock(100000);
  initLCD();
  delay(1500);

  ledcSetup(channelMaju, pwmFreq, pwmResolution);
  ledcAttachPin(RPWM, channelMaju);

  ledcSetup(channelMundur, pwmFreq, pwmResolution);
  ledcAttachPin(LPWM, channelMundur);

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Connecting WiFi ");

  Serial.print("[WIFI] Menghubungkan ke SSID: ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  unsigned long connectionStart = millis();
  int dotCount = 0;

  while (WiFi.status() != WL_CONNECTED && millis() -
connectionStart < 12000) {
    delay(400);
    lcd.setCursor(dotCount % 16, 1);
    lcd.print(".");
    dotCount++;
    Serial.print(".");
  }

  lcd.clear();
  if (WiFi.status() == WL_CONNECTED) {
    statusSistem = "NORMAL";

    Serial.println("\n[STATUS] WiFi Berhasil Tersambung!");
    Serial.print("[IP ADDRESS] ");
    Serial.println(WiFi.localIP());
    Serial.println("----------------------------------------
");

    lcd.setCursor(0, 0);
    lcd.print("WiFi Connected! ");
    lcd.setCursor(0, 1);
    lcd.print(WiFi.localIP().toString());
  } else {
    statusSistem = "NORMAL (OFFLINE)";
    WiFi.disconnect();
    WiFi.mode(WIFI_AP);
    WiFi.softAP("ESP32_Automarking", "12345678");

    Serial.println("\n[STATUS] WiFi Gagal Terkoneksi
(Timeout/Tidak Ditemukan)!");
    Serial.println("[STATUS] Mengaktifkan Mode Access Point
(SoftAP)...");
    Serial.print("[SSID AP] ESP32_Automarking | [IP ADDRESS AP]
");
    Serial.println(WiFi.softAPIP());
```

```
    Serial.println("----------------------------------------
");

    lcd.setCursor(0, 0);
    lcd.print("AP Mode Active  ");
    lcd.setCursor(0, 1);
    lcd.print(WiFi.softAPIP().toString());
  }

  delay(3500);
  lcd.clear();

  setupWebServer();

  driveMotor(pwmNormal);
  calculateVoltage();

  Serial.println("[SYSTEM] Mesin Siap Digunakan. Konveyor
Berjalan.");
}

//================================================================
==========
// LOOP (STATE MACHINE & CORE PROCESSOR)
//================================================================
==========
void loop() {
  server.handleClient();


//================================================================
=======
  // 1. NON-BLOCKING SENSOR PROXIMITY DEBOUNCE FILTER

//================================================================
=======
  bool currentReading = digitalRead(PROXIMITY_PIN);

  if (currentReading != lastSensorState) {
    lastDebounceTime = millis();
  }

  if ((millis() - lastDebounceTime) > debounceDelay) {
    if (currentReading != sensorStableState) {
      sensorStableState = currentReading;
    }
  }
  lastSensorState = currentReading;

  if (sensorStableState == HIGH) {
    barangSudahLewat = true;
  }


//================================================================
=======
  // 2. STATE MACHINE CONFIGURATION (LOGIC TRACKING ONLY)
```

```
//============================================================
=======
  switch (currentState) {

    case STATE_NORMAL:
      if (sensorStableState == LOW && barangSudahLewat) {
        currentState = STATE_MARKING;
        markingStartTime = millis();
        barangSudahLewat = false;
        jumlahBarang++;

        Serial.print("[LOG] Barang Terdeteksi! Jumlah: ");
        Serial.println(jumlahBarang);

        if (!simulationActive) {
          statusSistem = "MARKING ACTIVE";
        }
      }
      break;

    case STATE_MARKING:
      if (millis() - markingStartTime >= waktuSemprot) {
        // Proses marking selesai, kembalikan state ke Normal
        currentState = STATE_NORMAL;
      }
      break;

    case STATE_LOW_VOLTAGE:
      break;
  }


//============================================================
=======
  // 3. ACTUATOR STACK CONTROL & ENGINE DRIVERS

//============================================================
=======
  if (currentState == STATE_MARKING) {
    stopMotor();                    // Motor aman di-set 0V
karena proteksi dipagari logis
    digitalWrite(RELAY_POMPA, LOW); // Pompa Aktif (Active LOW)
    relayActive = true;
  }
  else if (currentState == STATE_NORMAL) {
    digitalWrite(RELAY_POMPA, HIGH); // Pompa Nonaktif
    relayActive = false;
    digitalWrite(BUZZER, LOW);
    buzzerActive = false;

    if (simulationActive) {
      runSimulationMachine();        // Jalankan kalkulasi
kurva PWM simulasi secara paralel
    } else {
      driveMotor(pwmNormal);         // Jalankan motor pada
kecepatan default
      statusSistem = "NORMAL";
    }
```

```
  }


//================================================================
=======
  // 4. TELEMETRY COMPUTATION

//================================================================
=======
  calculateVoltage();


//================================================================
=======
  // 5. CRITICAL UNDER-VOLTAGE PROTECTIVE SAFETY CHECKS (UVLO)
  // MODIFIKASI: Hanya aktif saat simulasi berjalan, pada tahap
3 (ramp down),
  // dan memastikan konveyor tidak sedang melakukan interupsi
marking fisik.

//================================================================
=======
  if (simulationActive && currentSimStep == SIM_T3_RAMP_DOWN &&
currentState == STATE_NORMAL) {
    if (teganganMotor < 5.0) {
      if (currentState != STATE_LOW_VOLTAGE) {
        Serial.println("[CRITICAL ALERT] Terjadi Drop Tegangan
Simulasi! < 5.0V. Sistem Dikunci!");
      }
      currentState = STATE_LOW_VOLTAGE;
      statusSistem = "LOW VOLTAGE (FAULT)";

      stopMotor();
      calculateVoltage();
      digitalWrite(RELAY_POMPA, HIGH);
      relayActive = false;

      digitalWrite(BUZZER, HIGH);
      buzzerActive = true;
    }
  }


//================================================================
=======
  // 6. DISPLAY SYNC

//================================================================
=======
  updateLCDTelemetry();
}
```

Lampiran Datasheet ESP32 DevKit V1

Tabel Spesifikasi ESP32 DevKit V1

Parameter	Spesifikasi

Mikrokontroler	ESP32-WROOM-32
CPU	Dual-Core Xtensa LX6
Frekuensi Clock	Hingga 240 MHz
Flash Memory	4 MB
SRAM	520 KB
Tegangan Operasi	3,3 V
Tegangan Input	5 V USB
WiFi	IEEE 802.11 b/g/n
Bluetooth	Bluetooth v4.2 BLE
GPIO	34 Pin
ADC	12-bit
DAC	2 Channel
PWM	16 Channel
Interface	UART, SPI, I2C, I2S

Sumber: Espressif Systems, *ESP32-WROOM-32 Datasheet*.

Lampiran Datasheet Sensor Proximity

Tabel Spesifikasi Sensor Proximity LJ12A3-4-Z/BX

Parameter	Spesifikasi
Model	LJ12A3-4-Z/BX
Jenis	Inductive Proximity Sensor
Tegangan Kerja	6–36 V DC
Jarak Deteksi	4 mm
Output	NPN Normally Open
Arus Maksimum	300 mA
Frekuensi Respon	500 Hz
Diameter Sensor	12 mm

Sumber: Datasheet LJ12A3-4-Z/BX.

Lampiran Datasheet Sensor Tegangan

Tabel Spesifikasi Sensor Tegangan DC 0–25 Volt

Parameter	Spesifikasi
Jenis Sensor	Voltage Sensor Module
Rentang Pengukuran	0–25 V DC
Output	Analog
Prinsip Kerja	Voltage Divider

Tegangan Output	0–3,3 V
Kompatibel	ESP32, Arduino

Sumber: DFRobot Voltage Sensor Documentation.

Lampiran Biodata

1. Nama Lengkap : Dony Dwi Yulianto
2. Nama : Dony
3. Program Studi : D3 - Teknik Kelistrikan Kapal
4. Jenis Kelamin : Laki - Laki
5. Agama : Islam
6. TTL : Probolinggo,29 Juli 2004
7. Alamat : Jln. Pelita 1, Sumberkedawung, Leces, Kab Probolinggo
8. No Telp/HP : 081334078897
9. Email : arblogenfc@gmail.com
10. Nama Orang Tua/ Wali
    - Ayah : Sakriyadi
    - Ibu : Sulastri

Alamat Orang Tua/Wali : Jln. Pelita 1, Su
11. berkedawung, Leces, Kab Probolinggo
12. Riwayat Pendidikan
    - 2011 – 2017 : SDN Sumberkedawung III
    - 2017 – 2020 : SMPN 1 Leces
    - 2020 – 2023 : SMA Taruna Dra. Zulaeha
    - 2023 – Sekarang : D3 Teknik Kelistrikan Kapal PPNS