

***PROTOTYPE* SISTEM UNTUK MENDETEKSI JARAK AMAN  
KENDARAAN dan KONTROL KECEPATAN BERBASIS IoT  
MENGUNAKAN ESP32**

**PROYEK AKHIR**

Laporan akhir ini dibuat dan diajukan untuk memenuhi salah satu syarat  
Kelulusan Sarjana Terapan/Diploma IV Politeknik Manufaktur Negeri Bangka  
Belitung



Diusulkan oleh :

Hebri

NIM : 1052212

Candra

NIM : 1052205

**POLITEKNIK MANUFAKTUR NEGERI  
BANGKA BELITUNG  
TAHUN 2025**

## LEMBAR PENGESAHAN

### **PROTOTYPE SISTEM UNTUK MENDETEKSI JARAK AMAN KENDARAAN DAN KONTROL KECEPATAN BERBASIS IoT MENGUNAKAN ESP32**

Oleh :

Hebri/1052212

Candra/1052205

Laporan akhir ini telah disetujui dan disahkan sebagai salah satu syarat kelulusan  
Program Sarjana/Diploma IV Politeknik Manufaktur Negeri Bangka Belitung

Menyetujui,

Pembimbing 1




(Yudhi M.T.)

Pembimbing 2



(Rizki Meliya Ningsi, S.P., M.Si.)

Penguji 1



( Aan Febriansyah, S.ST.,M.Tr.T.)

Penguji 2



(Wahid Hasyimi, S.Pi., M.Si)

## PERYATAAN BUKAN PLAGIAT

Yang bertanda tangan di bawah ini:

Nama Mahasiswa 1 : Hebri

NIM: 1052212

Nama Mahasiswa 2 : Candra

NIM: 1052205

Dengan Judul : *Prototype* Sistem Untuk Mendeteksi Jarak Aman Kendaraan dan Kontrol Kecepatan Berbasis IoT Menggunakan ESP32.

Menyatakan bahwa laporan akhir ini adalah hasil kerja keras kami sendiri dan bukan merupakan plagiat. Peryataan ini kami buat untuk sebenarnya dan bila ternyata di kemudian hari melanggar peryataan ini, kami bersedia menerima sanksi yang berlaku.

Sungailiat, 24 juli 2025

Nama Mahasiswa

Tanda Tangan

1. Hebri



2. Candra



.....

## ABSTRAK

Mobil memudahkan mobilitas manusia, namun pengemudi sering kali kesulitan menjaga jarak aman dan mengendalikan kecepatan, terutama saat terjadi kemacetan. Jarak yang tidak aman antar kendaraan merupakan salah satu penyebab utama terjadinya kecelakaan. Untuk mengatasi permasalahan tersebut, dikembangkan sebuah *prototype* sistem deteksi jarak aman dan kontrol kecepatan berbasis IoT menggunakan *mikrokontroler* ESP32. Sistem ini memanfaatkan sensor ultrasonik HC-SR04 dan JSN-SR04T untuk mengukur jarak dari empat arah (depan, belakang, kiri, kanan) dan sensor *photo-interrupter* untuk mendeteksi kecepatan kendaraan secara *real-time*. Data dari sensor ditampilkan pada LCD dan dikirim ke aplikasi Blynk melalui jaringan *WiFi*, sehingga pengemudi dapat memantau kondisi kendaraan secara langsung melalui *smartphone*. Sistem juga dilengkapi dengan LED dan *buzzer* sebagai alat peringatan ketika jarak atau kecepatan kendaraan melebihi ambang batas yang telah ditentukan. Hasil pengujian menunjukkan bahwa sistem mampu mendeteksi jarak dengan akurasi  $\pm 2$  cm dan merespons secara cepat dan stabil terhadap perubahan kondisi. Dengan mengintegrasikan pemantauan jarak dan kontrol kecepatan, sistem ini diharapkan dapat membantu pengemudi menghindari potensi tabrakan dan meningkatkan keselamatan berkendara.

Kata Kunci: Blynk, ESP32, HC-SR04, IoT, Jarak Aman kendaraan.

## ABSTRACT

*Cars significantly enhance human mobility, but drivers often struggle to maintain a safe following distance and control speed, especially in traffic congestion. Unsafe distance between vehicles is one of the leading causes of collisions. To address this issue, a prototype system for safe distance detection and speed control based on the IoT was developed using the ESP32 microcontroller. The system utilizes HC-SR04 and JSN-SR04T ultrasonic sensors to measure distances from four directions (front, rear, left, and right) and an opto-interrupter sensor to detect vehicle speed in real-time. Sensor data is displayed on an LCD and transmitted via WiFi to the Blynk application, enabling drivers to monitor vehicle conditions directly from their smartphones. The system is also equipped with LED indicators and a buzzer to alert the driver when the distance or speed exceeds predefined safety thresholds. Test results show the system can detect distances with an accuracy of  $\pm 2$  cm and respond quickly and reliably to changes in surroundings. By integrating distance monitoring and speed control, this prototype is expected to help drivers avoid potential collisions and enhance driving safety.*

*Keywords: Blynk, ESP32, HC-SR04, IoT, Vehicle Safe Distance.*

## KATA PENGANTAR

Bismillahirrahmanirrahim

Penulis ucapkan puji dan syukur atas kehadiran Tuhan Yang Maha Esa karena berkat rahmat dan karunia-Nya Penulis dapat menyelesaikan makalah proyek akhir ini dengan baik dan sesuai dengan waktu yang telah ditentukan. Untuk memenuhi salah satu syarat wajib kelulusan Diploma IV di Politeknik Manufaktur Negeri Bangka Belitung maka diperlukan untuk membuat makalah proyek akhir ini. Melalui makalah ini bertujuan untuk membuat pembaca dapat mengetahui gambaran proyek akhir yang dibuat oleh penulis. Makalah proyek akhir ini dibuat berdasarkan pengembangan dari jurnal-jurnal penelitian yang sudah pernah dilakukan sebelumnya. Dalam proyek akhir ini, penulis mencoba menerapkan ilmu pengetahuan yang sudah didapat selama 4 tahun menempuh pendidikan di Politeknik Manufaktur Negeri Bangka Belitung. Selain itu, penulis juga menjadikan makalah-makalah proyek akhir mahasiswa Politeknik Manufaktur Negeri Bangka Belitung pada tahun-tahun sebelumnya untuk dijadikan informasi mengenai data-data pendukung dari proyek akhir ini.

Dalam penyusunan makalah proyek akhir ini, penulis mendapatkan bantuandan bimbingan dari berbagai pihak yang membuat proses pembuatan makalah ini dapat diselesaikan dengan baik. Oleh sebab itu, penulis ingin mengucapkan terima kasih kepada:

1. Orang tua dan keluarga tercinta yang selalu memberikan dukungan baik secara moral maupun materi sehingga penulis dapat memberikan hasil yang terbaik.
2. Bapak I Made Andik Setiawan, M.Eng., Ph.D. selaku Direktur Politeknik Manufaktur Negeri Bangka Belitung.
3. Bapak Zanu Saputra, S.ST., M.Tr.T. selaku ketua jurusan Rekayasa Elektro dan Industri Pertanian Negeri Bangka Belitung.
4. Bapak Aan Febriansyah, S.ST., M.T. selaku Koordinator Program Studi D-IV Teknik Elektronika Politeknik Manufaktur Negeri Bangka Belitung.
5. Bapak Yudhi, M.T. selaku Dosen pengajar Politeknik Manufaktur Negeri Bangka Belitung sekaligus dosen pembimbing 1 dalam proyek akhir ini.

6. Ibu Riski Meliya Ningsih, S.P.,M.Si. selaku Dosen Pengajar Politeknik Manufaktur Negeri Bangka Belitung Sekaligus dosen pembimbing 2 dalam proyek akhir ini.
7. Seluruh tenaga pendidik dan ke pendidikan Politeknik Manufaktur Negeri Bangka Belitung.
8. Teman-teman Mahasiswa Politeknik Manufaktur Negeri Bangka Belitung yang telah membantu dalam penyelesaian proyek akhir ini.
9. Seluruh pihak yang ikut terlibat baik secara langsung maupun tidak langsung dalam penyelesaian proyek akhir ini yang tidak bisa di sebutkan satu persatu.

Dalam penulisan laporan ini, penulis menyadari masih terdapat kekurangan dikarenakan masih terbatasnya pengetahuan dan kemampuan penulis. Oleh karena itu, untuk perbaikan dan pengembangan penulisan makalah ini di masa yang akan datang, penulis mengharapkan kritik dan saran dari semua pihak. Penulis berharap makalah ini dapat bermanfaat bagi pihak yang berkepentingan pada khususnya dan bagi perkembangan ilmu pengetahuan dan teknologi.

Sungailiat, 24 Juli 2025



Penulis

## DAFTAR ISI

LEMBAR PENGESAHAN.....	ii
PERYATAAN BUKAN PLAGIAT .....	iii
ABSTRAK .....	iv
ABSTRACT .....	v
KATA PENGANTAR.....	vi
DAFTAR ISI.....	viii
DAFTAR TABEL .....	x
DAFTAR GAMBAR.....	xi
DAFTAR LAMPIRAN .....	xii
<b>BAB 1 PENDAHULUAN .....</b>	<b>1</b>
1.1. Latar Belakang .....	1
1.2. Prumusan Masalah .....	2
1.3. Batasan Masalah.....	2
1.4. Tujuan.....	2
<b>BAB II LANDASAN TEORI .....</b>	<b>3</b>
2.1 Jarak Aman Kendaraan .....	3
2.2. <i>Internet of Things</i> .....	4
2.3. Sensor Ultrasonik HC-SR04 .....	4
2.4. <i>Microkontroler</i> ESP32 .....	5
2.5. <i>Blynk</i> .....	6
2.6. Sensor Ultrasonik JSN-sr04T .....	7
<b>BAB III METODE PELAKSANAAN.....</b>	<b>9</b>
3.1. Pengumpulan dan Pengolahan Data.....	9
3.2. Rancangan <i>Hardware</i> dan <i>Software</i> .....	10
3.2.1. Rancangan <i>Hardware</i> .....	10
3.2.2. Rancangan <i>Software</i> .....	10
3.3. Pembuatan <i>Hardware</i> dan <i>Software</i> .....	10
3.3.1. Pembuatan <i>Hardware</i> .....	11
3.3.2. Pembuatan <i>Software</i> .....	11
3.4. Pengujian <i>Hardware</i> dan <i>Software</i> .....	11
3.4.1. Pengujian <i>Hardware</i> .....	11
3.4.2. Pengujian <i>Software</i> .....	12



3.5. Pengujian Keseluruhan Sistem.....	12
3.6. Analisa Data.....	13
3.7. Pembuatan Makalah Projek Akhir.....	13
<b>BAB IV PEMBAHASAN.....</b>	<b>14</b>
4.1. Deskripsi Alat.....	14
4.2. Perancangan dan Pembuatan <i>Hardware</i> Pendeteksi Jarak Aman pada kendaraan dan Pengaturan Kecepatan.....	15
4.2.1. Perancangan <i>Hardware</i> secara <i>Mekanik</i> .....	15
4.2.2. Pembuatan <i>Hardware</i> secara <i>Mekanik</i> .....	17
4.2.3. Perancangan dan Pembuatan <i>Hardware Elektrik</i> .....	18
4.3. Pengujian Sensor Ultrasonik.....	20
4.3.1. <i>Prosedure</i> Pengujian Sensor HC-SR04 dan JSN-SR04t.....	21
4.3.2. Hasil Pengujian Sensor HC-SR04 dan JSN-SR04t.....	22
4.4. Pengujian Sensor <i>Photo Interrupter</i> .....	23
4.4.1. <i>Prosedure</i> Pengujian Sensor <i>Photo Interrupter</i> .....	23
4.4.2. Hasil Pengujian Sensor <i>Photo Interrupter</i> .....	24
4.5. Perancangan dan Pembuatan <i>Software</i> Deteksi Jarak Aman dan Pengaturan Kecepatan.....	25
4.5.1. Perancangan <i>Software</i> .....	26
4.5.2. Pembuatan <i>Software</i> .....	26
4.6. Pengujian <i>Software</i> Deteksi Jarak Aman pada Kendaraan dan Pengaturan Kecepatan.....	27
4.6.1. Pembacaan Data <i>Blynk</i> Secara <i>Real-Time</i> .....	28
4.7. Pengujian Alat Keseluruhan .....	28
4.7.1. Pengujian Sensor Jarak Kiri, Kanan, Depan, dan Belakang....	29
4.7.2. Pengujian Sistem <i>Blynk</i> .....	30
4.7.2.1. <i>Monitoring</i> Data Sensor.....	32
4.7.2.2. Kontrol Kecepatan Melalui Aplikasi.....	32
<b>BAB V KESIMPULAN DAN SARAN.....</b>	<b>33</b>
5.1 Kesimpulan.....	33
5.2 Saran.....	33

## DAFTAR GAMBAR

Gambar 1. 1 Proses Kerja Ultrasonik.....	5
Gambar 1. 2 <i>Microcontroller</i> Unit NodeMCU (ESP32).....	6
Gambar 1. 3 Aplikasi <i>Blynk</i> .....	7
Gambar 1. 4 Proses kerja sensor ultrasonik JSN-sr04t .....	7
Gambar 3. 1 Tahapan-tahapan Pelaksanaan.....	8
Gambar 4.1 Diagram blok cara kerja <i>prototype</i> .....	14
Gambar 4. 2 Tampak dalam Perancangan <i>Hardware</i> .....	17
Gambar 4. 3 Tampak Samping Perancangan <i>Hardware</i> .....	17
Gambar 4.4 Tampilan <i>Hardware Mekanik</i> Tampak Dalam .....	18
Gambar 4. 5 Tampilan <i>Hardware Mekanik</i> Tampak Luar.....	18
Gambar 4.6 Skema Rangkaian Sistem Kontrol.....	19
Gambar 4.7 Hasil dari sistem kontrol elektrik <i>prototype</i> .....	20
Gambar 4.8 Diagram Blok Pengujian Sensor HC-SR04 .....	21
Gambar 4. 9 Pengujian Sensor Terhadap Objek .....	21
Gambar 4. 10 Pengujian Sensor HC-SR04 .....	22
Gambar 4. 11 Pengujian Sensor JSN-SR04t.....	22
Gambar 4. 12 Diagram Pengujian Sensor <i>photo interrupter</i> .....	25
Gambar 4. 13 Pengujian Sensor <i>photo interrupter</i> .....	25
Gambar 4. 14 Tampilan Program Arduino IDE.....	27
Gambar 4. 15 Tampilan Perancangan <i>Software</i> Aplikasi <i>Blynk</i> .....	26
Gambar 4. 16 Tampilan <i>Monitoring</i> Jarak Pada Aplikasi <i>Blynk</i> .....	31
Gambar 4. 17 Tampilan <i>Monitoring</i> Kecepatan Pada Aplikasi <i>Blynk</i> .....	32

## DAFTAR TABEL

Tabel 4. 1 Hasil Pengujian perbandingan sensor dengan Alat ukur meteran,.....	23
Tabel 4. 2 Data Hasil Percobaan terhadap <i>Buzzer</i> .....	25
Tabel 4. 3 Hasil Pengujian jarak sensor kiri, kanan, depan, dan belakang .....	30



## **DAFTAR LAMPIRAN**

Lampiran 1. 1 Daftar Riwayat Hidup.....	38
Lampiran 1. 2 Program .....	39



# BAB 1

## PENDAHULUAN

### 1.1. Latar Belakang

Setiap tahun, jumlah kendaraan di Indonesia terus meningkat baik kendaraan bermotor maupun kendaraan roda empat, berdasarkan data Badan Pusat Statistik (BPS) pada 29 Februari 2024, jumlah kendaraan mencapai 157.080.504 unit[1]. sehingga meningkatkan kepadatan lalu lintas dan risiko kecelakaan. Kurangnya jarak aman antar pengendara, terutama saat terjadi kemacetan lalu lintas saat berhenti mendadak, merupakan salah satu penyebab utama kecelakaan. Bahaya ini meningkat akibat kelalaian pengemudi, seperti kurang konsentrasi dan kegagalan mengantisipasi lalu lintas yang datang dari arah berlawanan[2].

Menurut Kevin Ashton pada tahun 1999, IoT adalah sebuah konsep atau skenario dari objek yang memiliki *transfer* data melalui jaringan tanpa memerlukan interaksi manusia ke manusia atau manusia ke sumber[3]. Teknologi berbasis IoT menawarkan cara untuk meningkatkan keselamatan berkendara. Salah satunya adalah sistem pendeteksi jarak otomatis, sistem ini dapat melacak jarak antar mobil secara *real time* dan memberi peringatan kepada pengemudi saat jaraknya terlalu dekat. Agar pengemudi dapat melihat keadaan dari dekat, sensor ultrasonik digunakan untuk mengukur jarak dari berbagai arah. Data tersebut kemudian ditampilkan di telepon pintar menggunakan aplikasi *Blynk*.

Untuk membantu pengemudi menjaga jarak aman dari kendaraan lain, maka kami rancang proyek ini dengan judul “*Prototype Deteksi Jarak Aman Kendaraan dan Kontrol Kecepatan Berbasis IoT*”. Jika jarak minimum terlampaui, bel dan layar LCD akan memberi tahu pengemudi. Diharapkan metode ini akan meningkatkan keselamatan berkendara dan mengurangi kemungkinan tabrakan, terutama dalam situasi lalu lintas padat.

## 1.2. Perumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan maka, ditariklah beberapa rumusan masalah sebagai berikut;

1. Bagaimana merancang sistem pendeteksi jarak aman kendaraan yang akurat menggunakan sensor ultrasonik dan JSN Sr04t dan ESP32?
2. Bagaimana mengintegrasikan IoT untuk mengirim notifikasi jarak aman ke pengemudi via *smartphone*?
3. Bagaimana sistem dapat mengontrol kecepatan dan memberikan notifikasi otomatis saat jarak atau kecepatan melebihi batas yang di tentukan?

## 1.3 Batasan Masalah

Untuk memperjelas arah pembahasan dari proyek akhir ini, maka di berikan batasan masalah sebagai berikut.

1. Bagaimana merancang sistem deteksi jarak aman kendaraan yang akurat menggunakan sensor ultrasonik HC-SR04, JSN-SR04T, dan ESP32?.
2. Bagaimana mengintegrasikan *teknologi* IoT untuk mengirim *notifikasi* jarak aman kepada pengemudi melalui aplikasi *smartphone*?
3. Bagaimana menentukan batas jarak aman secara dinamis berdasarkan kecepatan kendaraan?

## 1.4 Tujuan

Tujuan yang ingin dicapai dari proyek pengontrolan kecepatan dan jarak kendaraan ini diuraikan di bawah ini.

1. Merancang dan membuat sistem peringatan jarak aman serta pengontrol kecepatan kendaraan berbasis sensor ultrasonik dan IoT.
2. Mengintegrasikan sistem sensor dengan *platform* IoT untuk memantau jarak dan kecepatan kendaraan secara *real-time* melalui *smartphone*.
3. Mengembangkan mekanisme notifikasi yang dapat memperingatkan pengemudi secara otomatis jika jarak dan kecepatan kendaraan berada di luar batas aman.

## BAB II DASAR TEORI

### 2.1. Jarak Aman Kendaraan

Saat mengoperasikan kendaraan, mengutamakan keselamatan adalah hal yang penting. Jarak aman antar kendaraan mengacu pada jarak seminimal mungkin yang harus dijaga antara mobil dan objek apa pun di depan untuk mencegah kecelakaan[4]. Jarak ini di pengaruhi oleh faktor-faktor seperti kecepatan kendaraan, waktu reaksi pengemudi, dan kondisi jalan, berdasarkan standar keselamatan, jarak aman dapat dihitung menggunakan rumus:

$$\text{Jarak Aman} = (\text{Waktu Reaksi} \times \text{Kecepatan}) + \text{Jarak Pengereman}$$

Waktu reaksi manusia rata – rata adalah 1,5 detik, sedangkan jarak pengereman tergantung pada kondisi permukaan jalan dan kecepatan kendaraan[5]. Berikut ini menampilkan jarak yang umumnya dianggap aman dalam berkendara[6].

Tabel 2.1 Jarak Aman Berkendara

Kecepatan	Jarak Minimal	Jarak Aman
30 km/jam	15 m	30 m
40 km/jam	20 m	40 m
50 km/jam	25 m	50 m
60 km/jam	40 m	60 m
70 km/jam	50 m	70 m
80 km/jam	60 m	80 m
90 km/jam	70 m	90 m
100 km/jam	80 m	100 m
120 km/jam	100 m	120 m

## 2.2. *Internet of Things*

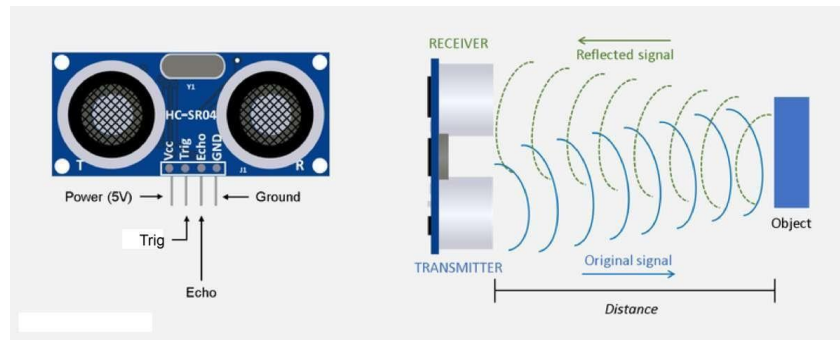
*Internet of Things*, atau IoT, adalah jaringan perangkat yang saling terkait yang menghubungkan dan bertukar data dengan perangkat IoT lain dan *could*. Perangkat IoT biasanya memiliki fitur seperti sensor dan perangkat lunak yang terdiri dari mesin mekanik dan *elektronik*, serta barang-barang konsumen sehari-hari[7]. Perangkat ini mencakup segala hal mulai dari peralatan rumah tangga sehari-hari hingga peralatan industri yang rumit. Semakin banyak organisasi di berbagai industri yang menggunakan IoT untuk beroperasi lebih *efisien* memberikan layanan pelanggan yang lebih baik, meningkatkan pengambilan keputusan, dan meningkatkan nilai bisnis[8]. Dengan IoT data dapat di *transfer* melalui jaringan tanpa memerlukan interaksi manusia-ke manusia atau manusia-ke komputer.

IoT adalah konsep dimana perangkat fisik terhubung ke *internet* dan dapat saling bertukar data, dalam proyek ini, IoT di gunakan untuk memantau jarak kendaraan secara *real-time* melalui *platform* seperti *Blynk* atau *ThingSpeak*. Keunggulan IoT meliputi: kemampuan *monitoring* jarak secara *real-time*. *Notifikasi* langsung ke *smartphone* pengemudi. Penyimpanan data untuk analisis lebih lanjut[9].

## 2.3. Sensor Ultrasonik HC-SR04

Terdapat empat koneksi pada sensor ultrasonik HC-SR04: *Vcc*, *Gnd*, *Trigger*, dan *Echo*. Setiap koneksi memiliki fungsi yang berbeda. Koneksi *Gnd* berfungsi sebagai *ground*, dan koneksi *Vcc* menyediakan tegangan positif. Koneksi *Echo* menangkap sinyal pantulan dari objek yang berada di depan sensor, sedangkan koneksi *trigger* memulai sinyal dari sensor. Dua *transduser* ultrasonik yang berfungsi sebagai pemancar gelombang dan penerima pantulan gelombang dan disertakan dengan sensor ultrasonik HC-SR04[10]. Melalui Gambar 1.1 berikut ini.





Gambar 1. 1 Proses Kerja Ultrasonik Sumber : <https://pendidikan-elektro.ft.unesa.ac.id/post/bagian-bagian-sensor-ultrasonik>

Mengatur jarak pada kendaraan sangat penting untuk memastikan kendaraan beroperasi dalam batas aman dan meminimalkan risiko tabrakan. Untuk menentukan seberapa jauh jarak mobil dari objek di depan, diperlukan sensor untuk mengukur jarak tersebut. Pada model ini, sensor ultrasonik digunakan untuk mengidentifikasi rintangan yang terletak di depan, di sebelah kanan, di sebelah kiri, dan di belakang kendaraan. Sensor ultrasonik HC-SR04 secara khusus dibuat untuk mengukur jarak ke suatu objek. Sensor ini dapat mengenali objek pada jarak mulai dari 2 cm hingga 200 cm[11].

Objek berukuran apa pun yang dapat memantulkan gelombang ultrasonik kembali ke sensor akan terdeteksi oleh sensor ultrasonik ini. Sensor ultrasonik telah banyak digunakan secara luas dalam berbagai proyek penelitian karena keakuratan dan jangkauan deteksi yang relatif luas 2 cm – 200 cm[12].

#### 2.4. Mikrokontroler ESP32

ESP 32 merupakan *mikrokontroler* yang telah dilengkapi dengan jaringan *bluetooth* dan *WiFi*, sehingga memudahkan pembuatan sistem IoT yang membutuhkan akses *nirkabel*. Data input sensor dibaca oleh *mikrokontroler* ESP32 dan dikirimkan ke server atau *platform* IoT untuk diproses melalui *modul WiFi* ESP32. Setelah itu, *platform* akan menampilkan data yang diterima untuk dilihat oleh pengguna[13].

Yang sudah dilengkapi prosesor inti ganda, kecepatan maksimum 240 MHz, memori akses acak 520 KB, dan *memori flash* hingga 4 MB merupakan fitur-fitur dari *mikrokontroler* ESP32. Karena konsumsi dayanya yang rendah dan rentang

tegangan operasi 2,2 V hingga 3,6 V sehingga *mikrokontroler* ini cocok untuk aplikasi IoT.

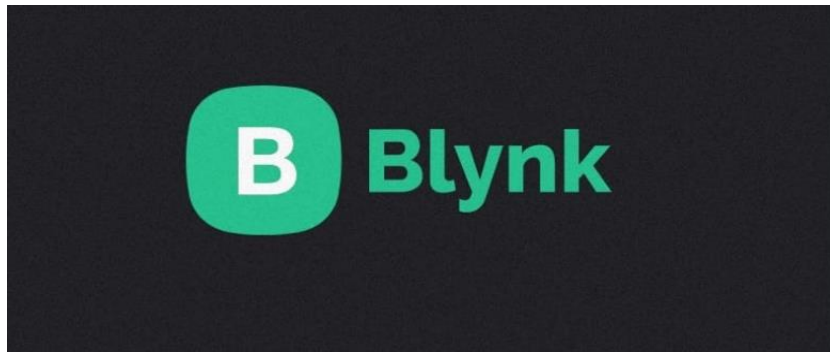


Gambar 1. 2. *Microcontroller Unit Node MCU(ESP32)* Sumber : <https://shopdelta.eu/development-kit-esp32-esp-wroom-32-espressif-12-p21391.html>

Dalam proyek ini, beberapa tegangan analog sensor jarak dibaca menggunakan fungsi *Analog-to-Digital Converter* (ADC) ESP32. Agar *mikrokontroler* dapat mengevaluasi data dan menentukan jarak objek yang terdeteksi dalam cm, ADC mengubah tegangan analog sensor menjadi nilai digital 12-bit. Selain itu, data digital diproses menggunakan rumus untuk setiap sensor dan ditampilkan secara *real-time* pada *platform* IoT seperti *Node-RED*. Pengguna dapat memperoleh informasi bahaya yang tepat dan tepat waktu berkat implementasinya, yang menjamin pemantauan jarak aman yang akurat dan efektif.

## **2.5. Blynk**

*Blynk* adalah *platform* IoT yang di rancang untuk memudahkan pengguna dalam membuat antarmuka kontrol dan *monitoring* perangkat berbasis *smartphone*. *Platform* ini tepat untuk proyek IoT sederhana hingga menengah dengan dukungan *fitur real-time*[14].



Gambar 1. 3 Aplikasi *Blynk*. Sumber : <https://community.dfrobot.com/makelog-312970.html>

Data akan tampil ke *smartphone* dalam bentuk *gauge*, data dan teks. Yang terintegrasi dengan ESP32 dengan dukungan *library* khusus untuk *microkontroler* berbasis *WiFi*. Pada proyek ini *Blynk* digunakan untuk *memonitoring* Jarak aman kendaraan dan pengontrolan kecepatan.

## 2.6. Sensor Ultrasonik JSN-sr04T

Sensor ultrasonik JSN-sr04t adalah perangkat yang menggunakan gelombang ultrasonik untuk mendeteksi keberadaan dan jarak objek di lingkungan yang membutuhkan ketahanan terhadap air dan kondisi *external* lainnya. Sensor ini memiliki pelindung atau pelapis khusus yang membuatnya tidak terpengaruh oleh kelembapan, hujan atau bahkan terendam air dalam jangka waktu tertentu[15].



Gambar 1. 4 proses kerja sensor ultrasonik JSN-sr04t. Sumber : <https://www.manorshi.com/id/JSN-SR04T-Ultrasonic-Module-Ranging-Module-Distance-Sensor-pd46813559.html>

Sehingga sangat cocok untuk di gunakan dengan alat pendeteksi jarak agar dapat mengukur jarak aman kendaraan terhadap objek yang menghalanginya.

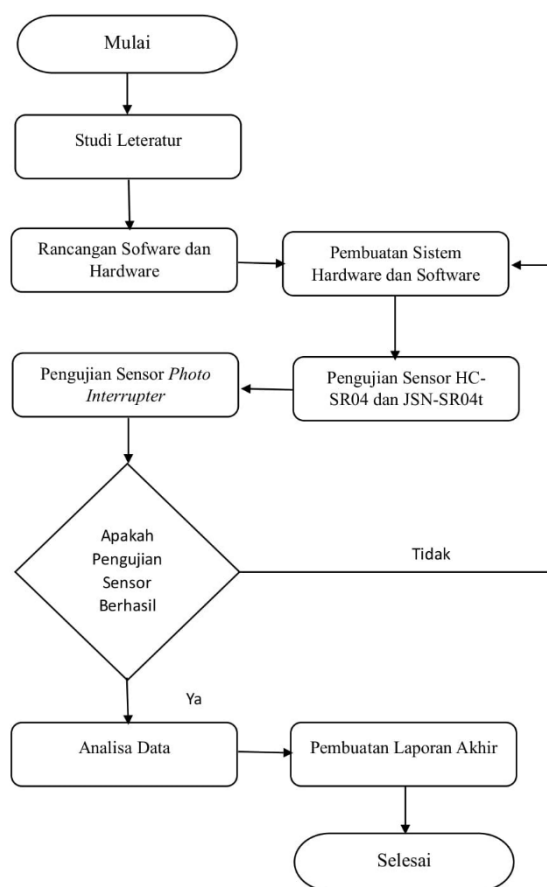
Sensor ultrasonik bekerja dengan memancarkan sinyal ke arah target atau objek, yang kemudian memantulkan kembali sinyal tersebut setelah mengenai objek.



### BAB III

## METODE PELAKSANAAN

Pada metode pelaksanaan yang dilakukan dalam pelaksanaan proyek akhir ini bertujuan untuk membantu dalam pembuatan proyek akhir. Informasi mengenai langkah-langkah yang terlibat dalam pelaksanaan proyek akhir ini dapat ditunjukkan dengan diagram alir. Pada Gambar 3. 1 berikut ini.



Gambar 3. 1 Tahapan-tahapan pelaksanaan

Tahapan-tahapan yang dilakukan untuk penyelesaian proyek *Prototype* Sistem untuk Mendeteksi Jarak Aman Kendaraan dan Kontrol Kecepatan Berbasis IoT diawali dengan *studi literatur* untuk mengumpulkan referensi dan teori terkait sensor jarak, sistem pengontrol kecepatan kendaraan, serta *platform* IoT guna

memahami konsep dasar dan teknologi yang digunakan. Setelah itu, dilakukan perancangan alat yang mencakup pembuatan desain rangkaian elektronik, pemilihan sensor, perancangan sistem kontrol, dan skema komunikasi antara perangkat keras dengan *platform* IoT.

Tahap berikutnya adalah pembuatan sistem yang meliputi perakitan komponen *hardware* seperti sensor, *mikrokontroler*, dan modul IoT, serta pembuatan *software* untuk membaca data sensor, mengontrol kecepatan kendaraan, dan mengirim data ke *platform* IoT secara *real-time*. Setelah sistem selesai dibuat, dilakukan pengujian alat untuk memastikan semua komponen berfungsi dengan baik, mulai dari akurasi sensor, keandalan pengontrol kecepatan, hingga kestabilan komunikasi data. Selanjutnya, dilakukan analisis hasil untuk mengevaluasi apakah sistem telah memenuhi kriteria yang diharapkan, seperti akurasi pendeteksian jarak, efektivitas pengontrolan kecepatan, dan ketepatan pengiriman data. Terakhir, seluruh proses tersebut didokumentasikan dalam laporan akhir sebagai bentuk pertanggung jawaban dan dokumentasi resmi dari proyek ini.

### **3.1. Pengumpulan dan Pengolahan Data**

Pengumpulan data dilakukan dengan tujuan memperoleh informasi serta pemahaman yang berkaitan dengan topik proyek akhir ini, yang nantinya digunakan sebagai referensi dalam proses pembuatan alat maupun penyusunan laporan proyek. Melalui referensi tersebut, dapat diperoleh gambaran umum mengenai sistem pengaturan kecepatan dan pengendalian jarak aman secara otomatis pada kendaraan untuk mencegah terjadinya bahaya, dengan menerapkan metode uji coba alat yang kemudian akan dikembangkan lebih lanjut dalam penelitian ini.

Proses pengumpulan data yang digunakan sebagai acuan melibatkan dua tahap, yaitu pengumpulan data *primer* yang diperoleh melalui konsultasi langsung dengan dosen pembimbing terkait pengerjaan proyek akhir, serta pengumpulan data sekunder yang diperoleh dengan mencari dan mempelajari berbagai *artikel* atau sumber *literatur* lain yang memiliki kaitan dengan tema proyek.

Setelah seluruh data berhasil dikumpulkan, langkah berikutnya adalah melakukan analisis dan perumusan terhadap data tersebut untuk kemudian dikembangkan menjadi sebuah ide baru yang akan diaplikasikan dalam proses

pembuatan dan pengembangan proyek akhir ini.

### **3.2. Rancangan *Hardware* dan *Software***

Rancangan *hardware* dan *software* merupakan tahapan di mana desain alat yang ditetapkan sebelumnya menjadi gambaran untuk pembuatan alat selanjutnya berikut di bawah ini tahapan rancangan *hardware* dan *software*.

#### **3.2.1. Rancangan *Hardware***

Perancangan *hardware prototype* pengendali kecepatan dan pengendali jarak otomatis pada mobil remot kontrol terdiri dari perangkat keras *mekanik* dan perangkat keras *elektrik*. Pada perancangan *hardware* ini, komponen yang akan digunakan di tentukan berdasarkan ukuran dan bentuk *prototype*. *Software* yang digunakan yaitu *Sketch Up* yang digunakan untuk merancang bentuk (struktur) aktual kendaraan mobil remot ini.

Berikut ini tahapan perancangan rangkaian *hardware* secara *mekanik* yang akan dibuat sebagai berikut.

1. untuk membuat *prototype* ini kami menggunakan Mobil remot *control* yang berukuran 30 cm x 15 cm x 5 cm.
2. Melengkapi mobil kendali jarak jauh dengan DFPlayerMini, LCD, LED, *buzzer*, motor *driver*, sensor *ultrasonik HC-SR04*, sensor kecepatan *photo interrupter*, dan *mikrokontroler* (ESP32).

Untuk perancangan *hardware* secara *elektrik*, pengendalian kecepatan otomatis dan pengendalian jarak terhadap bahaya kecelakaan yaitu dengan cara membuat rangkaian *wiring control* dengan sistem *control* yang memanfaatkan ESP32 dibuat selama tahap desain *hardware* secara *elektrik*.

#### **3.2.2. Rancangan *Software***

Rancangan *Software* pada proyek akhir ini menentukan perangkat mana yang akan digunakan. *Software* Arduino IDE digunakan untuk mengendalikan seluruh sistem, termasuk pengaturan kecepatan dan deteksi jarak bahaya pada mobil. Aplikasi *Blynk* digunakan untuk membuat IoT. Yang kemudian akan di *monitoring* lewat tampilan LCD dan apk *Blynk*.



### **3.3. Pembuatan *Hardware* dan *Software***

Tahap pembuatan perangkat keras dan perangkat lunak adalah tempat desain alat yang dikembangkan pada tahap awal menjadi alat dalam bentuk nyata berikut dibawah ini tahapan pembuatan *hardware*.

#### **3.3.1. Pembuatan *Hardware***

Komponen yang ditentukan dipasang ke dalam kendaraan *pengontrolan* kecepatan jarak jauh berdasarkan gambaran yang dibuat sebelumnya dalam tahap pembuatan mekanis. *Metode* yang digunakan dalam pembuatan *hardware mekanik* adalah sebagai berikut:

1. Modifikasi bentuk sensor ultrasonik, LCD, dan LED yang akan dipasang sebelum melubangi bodi kendaraan.
2. Terdapat LCD di atas bodi mobil, satu LED di bagian atas mobil, ESP32, sensor *photo interrupter*, motor driver, dan *buzzer* yang dipasang di dalam kendaraan, serta satu sensor ultrasonik di masing-masing sisi kanan, kiri, depan, dan belakang.

#### **3.3.2. Pembuatan *Software***

1. Pemrogram ESP32 untuk menampilkan data dari sensor jarak ultrasonik dan sensor kecepatan yang kemudian akan ditampilkan pada layar LCD dan *Blynk* Apk.
2. Pemograman ESP32 *dikonfigurasi* untuk mengatur kecepatan dan jarak mobil secara otomatis dengan menggunakan *buzzer*, motor DC, driver motor L298N, dan LED sebagai tindakan dan *indikator*.

### **3.4. Pengujian *Hardware* dan *Software***

Pengujian *hardware* dan *software* pada *prototype* alat pengendali kecepatan dan pendeteksi jarak aman kendaraan terhadap bahaya kecelakaan untuk mengetahui apakah *prototype* alat pengendali kecepatan dan pendeteksi jarak aman kendaraan terhadap bahaya kecelakaan dapat bekerja sesuai dengan target yang diinginkan.



#### **3.4.1. Pengujian *Hardware***

Ada beberapa pengujian *hardware* yang dilakukan pada alat deteksi jarak aman kendaraan dan *pengontrolan* kecepatan ini.

1. Untuk menguji kemampuan deteksi sensor maka diperlukan ESP32 dan sensor ultrasonik terhadap jarak objek.
2. Pengujian sensor *photo interrupter* dengan ESP32 untuk menentukan kemampuan deteksi kecepatan motor dari sensor *photo interrupter*.
3. Menggunakan *Blynk* untuk menguji ESP32 guna menentukan apakah kemampuan pemantauan *kompatibel* dengan data yang dihasilkan oleh sensor untuk memungkinkan pemantauan.

#### **3.4.2. Pengujian *Software***

Evaluasi *software* dalam proyek akhir ini dilakukan dengan memeriksa hasil yang dihasilkan dari pengkodean Arduino.

##### **Cara Kerja dalam proyek ini**

1. ESP32 mengirim data jarak dari sensor *HC-SR04* ke server *Blynk* via *wifi*.
2. Data di tampilkan di *wedget value display* atau *graph* pada aplikasi *Blynk*.
3. Jika jarak < batas aman, *Blynk* mengaktifkan *notifikasion widget* untuk mengirim *alert* ke pengemudi.

#### **3.5. Pengujian Keseluruhan Sistem**

Pengujian sistem secara menyeluruh dilakukan untuk mengevaluasi kinerja *prototype* dalam hal pengaturan kecepatan dan deteksi jarak aman kendaraan terhadap potensi bahaya kecelakaan. Sistem ini juga *dimonitor* melalui aplikasi *Blynk* dengan tujuan untuk memastikan apakah seluruh fungsi dapat berjalan dengan baik sesuai perancangan.

Langkah pengujian dilakukan dengan meletakkan objek di setiap sisi kendaraan, yaitu pada bagian depan, belakang, kiri, dan kanan, guna mengamati respon dari *buzzer* dan indikator LED akan menyala sebagai tanda peringatan apabila jarak objek sudah mencapai batas yang ditentukan.

Selain itu, pengujian jarak juga dilakukan dengan menempatkan objek di depan kendaraan, mulai dari posisi sangat dekat hingga jauh, sesuai dengan nilai

input yang telah diatur. Uji coba kecepatan dilakukan dengan mengubah tingkat kecepatan dari rendah hingga tinggi untuk melihat perubahan *output* PWM yang dihasilkan.

Selanjutnya, dilakukan pengamatan terhadap respon *buzzer* dan kondisi LED dalam berbagai skenario input jarak tertentu. Hasil dari seluruh proses pengujian ini kemudian dicatat dan direkap dalam bentuk tabel sebagai dokumentasi pengujian.

### **3.6. Analisa Data**

Prosedur analisis data didasarkan pada temuan pengujian, yang melibatkan pengujian sistem secara keseluruhan mulai *Hardware* dan *Software* yang digunakan dalam sistem deteksi jarak aman dan kontrol kecepatan kendaraan. Langkah ini akan mencakup *identifikasi* perbaikan dan penyempurnaan untuk mendapatkan hasil yang diinginkan jika data yang dikumpulkan memiliki kekurangan.

### **3.7. Pembuatan Makalah Proyek Akhir**

Makalah di buat mulai dari pendahuluan, teori, tahap implementasi, hasil pengujian dan pembahasan, sampai pada kesimpulan dan rekomendasi, setiap aspek konstruksi proyek akhir telah diperhitungkan dalam penyusunan makalah proyek akhir ini.

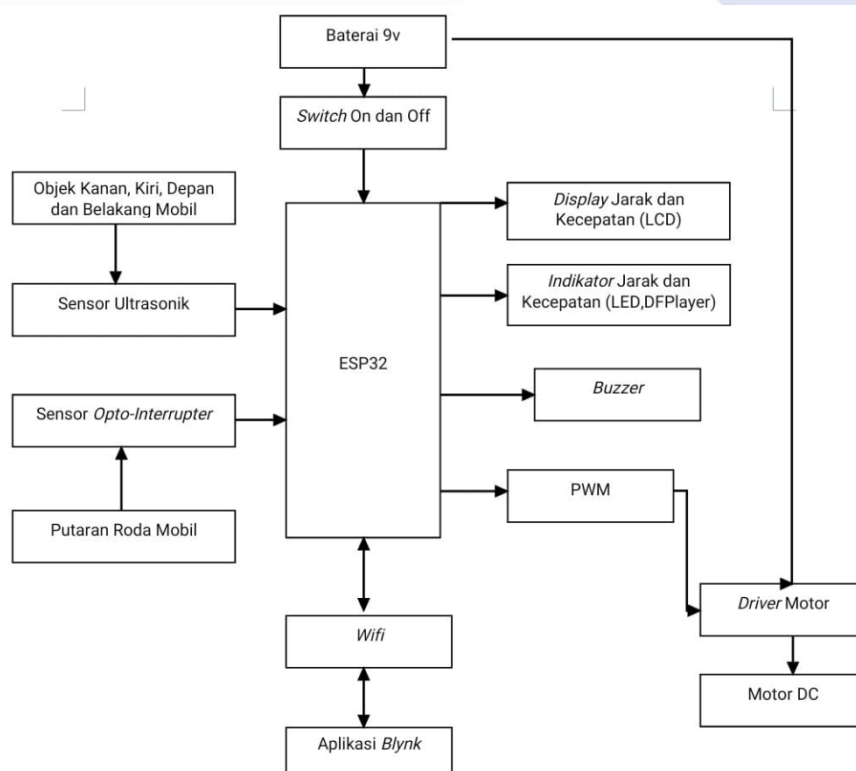
## BAB IV

### PEMBAHASAN

Tahapan - tahapan yang dilakukan dalam pembuatan proyek akhir ini akan dibahas dalam bab ini, termasuk merancang dan membangun *prototype* untuk kontrol kecepatan otomatis dan deteksi jarak aman kendaraan melalui pengujian alat.

#### 3.1. Deskripsi Alat

Perangkat *prototype* ini dirancang agar dapat mengendalikan kecepatan dan jarak secara otomatis ketika mendeteksi objek di sekitarnya dalam batas jarak yang telah ditentukan. Diagram blok yang ditampilkan pada Gambar 4.1 berikut memberikan gambaran tentang prinsip kerja sistem ini. Cara kerja sistem pengontrolan kecepatan dan deteksi jarak aman secara otomatis dapat di rancang sebagai berikut:



Gambar 4.1 Diagram blok cara kerja *prototype*

Diagram blok di atas menggambarkan cara kerja sistem deteksi jarak aman dan kontrol kecepatan kendaraan berbasis IoT menggunakan ESP32. Sistem ini diawali dari *Power* yang menyalakan seluruh rangkaian melalui saklar *On dan Off*. Sensor Ultrasonik dipasang di depan, belakang, kiri, dan kanan mobil untuk mendeteksi jarak objek disekitar kendaraan, sedangkan sensor *photo interrupter* membaca putaran roda untuk mengetahui kecepatan mobil.

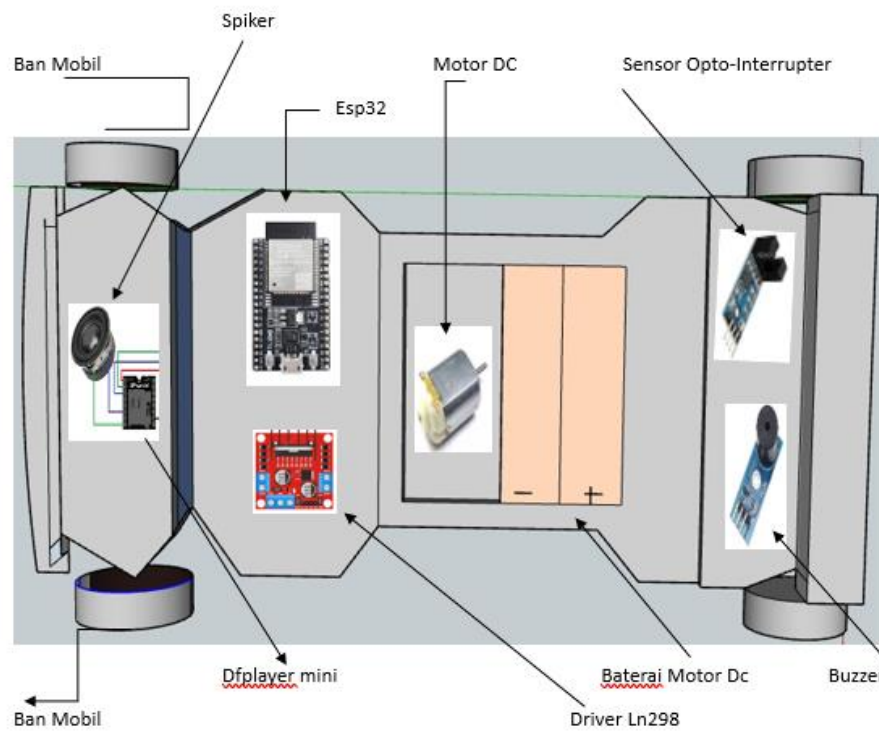
Semua data sensor dikirim ke ESP32 untuk diproses. Hasil ditampilkan pada LCD sebagai informasi jarak dan kecepatan, serta mengaktifkan *indicator LED*, DFPlayer mini, dan *buzzer* jika ada kondisi bahaya. Selain itu ESP32 mengatur kecepatan motor DC melalui sinyal PWM yang diteruskan ke *driver* motor. Data juga dikirim melalui *WiFi* ke aplikasi *Blynk* di *smartphone* sehingga pengguna bisa memantau dan mengendalikan sistem secara otomatis.

### **3.2. Perancangan dan Pembuatan *Hardware Prototype* Pengaturan Kecepatan dan Deteksi Jarak Otomatis Pada Kendaraan.**

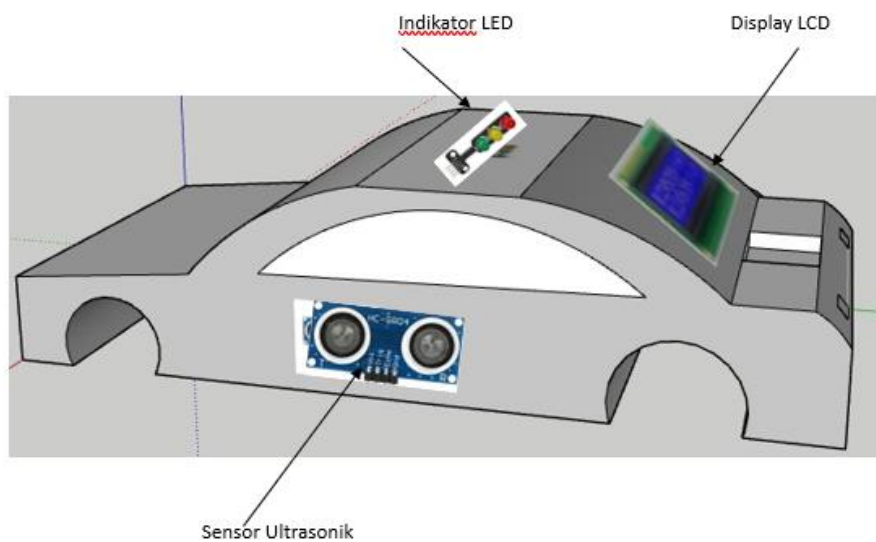
*Hardware* untuk *prototype* pengendali kecepatan otomatis dan pendeteksi jarak aman kendaraan ini dirancang dan diproduksi dalam dua tahap: *mekanik* dan *elektrik*. Berikut ini adalah langkah-langkah dalam tahap perancangan dan pembuatan *hardware*.

#### **3.2.1. Perancangan *Hardware* secara *Mekanik***

Pada tahap perancangan *mekanik hardware*, perancangan struktur fisik untuk *prototype* pengendali kecepatan dan jarak otomatis dilakukan dengan bantuan perangkat lunak *SketchUp*. Mobil mainan dengan dimensi 30 cm x 15 cm x 5 cm menjadi dasar pembuatan *prototype* ini. Berikut ini Gambar 4.2 dan 4.3 adalah perancangan *prototype* yang akan dibuat.



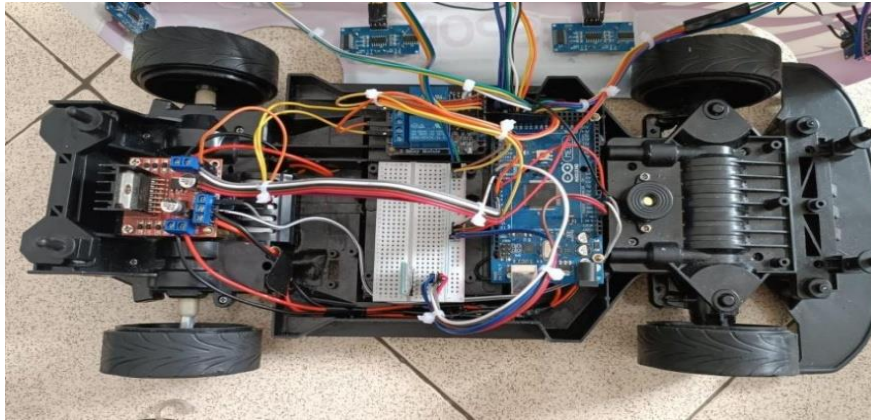
Gambar 4. 2 Tampak dalam pengaturan kecepatan dan deteksi jarak



Gambar 4. 3 Tampak Samping pengaturan kecepatan dan deteksi jarak

### 3.2.2. Pembuatan *Hardware* secara *Mekanik*

*Prototype* yang dibuat ini menggunakan sebuah mobil mainan dengan ukuran 30 cm x 15 cm x 5 cm. Pada bagian bodi mobil, dipasang 4 buah sensor ultrasonik yang diletakkan di setiap sisi, yaitu di bagian depan, belakang, kiri, dan kanan. Selain itu, di dalam mobil juga dipasangkan komponen seperti *mikrokontroler* ESP32, sensor *photo-interrupter*, *buzzer*, lampu lalu lintas (*traffic LED*), motor DC, modul *DFPlayer mini*, serta baterai. Berikut adalah merupakan tampilan akhir dari *prototype* sistem pengaturan kecepatan dan deteksi jarak aman kendaraan secara otomatis.



Gambar 4. 4Tampilan *Hardware Mekanik* Tampak Dalam



Gambar 4. 5 Tampilan *Hardware Mekanik* Tampak Luar

### 3.2.3. Perancangan dan Pembuatan *Hardware Elektrik*

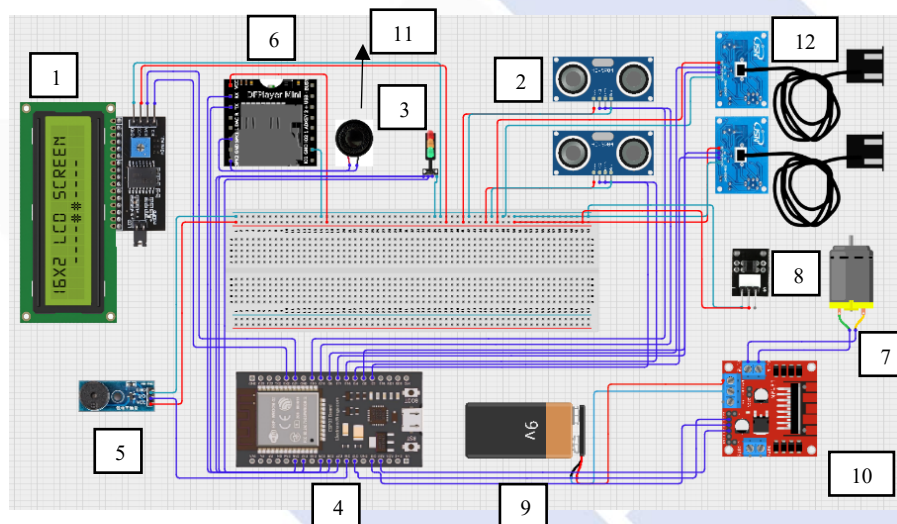
Pada tahap ini, desain sistem diimplementasikan dengan tujuan untuk mengendalikan deteksi kecepatan dan jarak kendaraan. Terdapat dua jenis data



input dalam desain ini, input jarak dari sensor ultrasonik dan input kecepatan dari sensor *photo interrupter*. Mikrokontroler ESP32 akan memproses kedua data tersebut untuk mengatur LED, *buzzer*, jarak dan kecepatan.

Perancangan dan pembuatan sistem untuk mengontrol sistem pengaturan kecepatan dan deteksi jarak meliputi tahap berikut ini.

1. Menggunakan aplikasi *Circuit Designer* untuk membuat tata letak kabel atau *wiring*. Sistem *wiring* ini dikembangkan untuk memudahkan pembuatan dan perbaikan instalasi kabel yang sebenarnya. Desain rangkaian untuk komponen yang digunakan dalam proyek ini ditampilkan pada Gambar 4.6 di bawah ini.

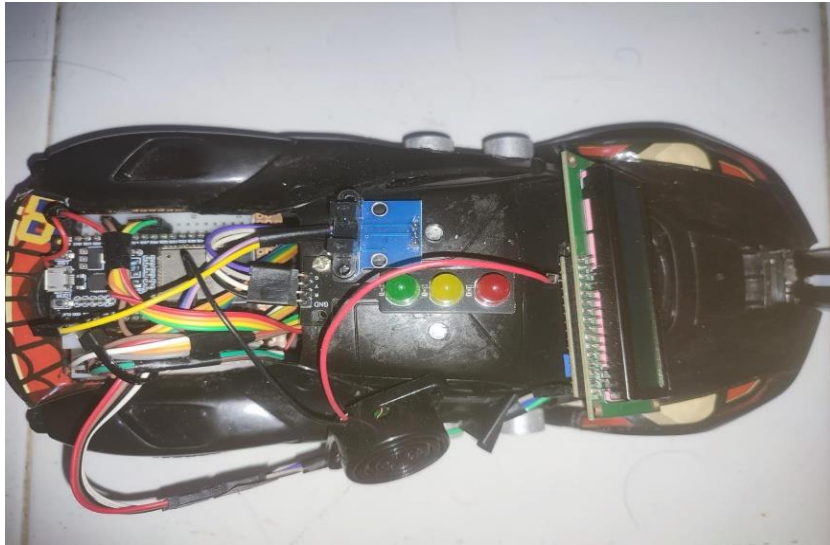


Gambar 4. 6 Skema Rangkaian Sistem Kontrol

**Keterangan:**

- |                  |                                   |
|------------------|-----------------------------------|
| 1.LCD            | 7.Motor DC                        |
| 2.Sensor HC-SR04 | 8.Sensor <i>photo Interrupter</i> |
| 3.LED            | 9.Baterai 9V                      |
| 4.ESP32          | 10.Driver l29N                    |
| 5. <i>Buzzer</i> | 11.Spiker                         |
| 6.DFPlayerMini   | 12. JSN-SR04t                     |

2. Pengkabelan komponen melibatkan pemasangan motor DC, modul DFPlayerMini, *buzzer*, LED, LCD, sensor HC-SR04/JSN-sr04t, dan sensor *photo interrupter* ke pin ESP32. Hasil pemasangan sistem kontrol pada *prototype* adalah sebagai berikut.



Gambar 4. 7 Hasil dari sistem kontrol *elektrik prototype*

Gambar diatas menunjukkan tampilan akhir dari rangkaian kontrol *elektrik* pada *prototype* kendaraan. Seluruh komponen seperti *microkontroler* ESP32, sensor jarak HC-SR04 dan JSN-SR04T, sensor kecepatan *photo interrupter*, *buzzer*, LCD, DFPlayer mini, motor DC, dan LED *indikator* telah terpasang pada body kendaraan mainan sesuai dengan rancangan sebelumnya. Kabel-kabel disusun rapi untuk memudahkan *troubleshooting* serta mencegah gangguan *fungsi* antar komponen. Sistem ini berfungsi sebagai pusat pemrosesan sinyal dari sensor untuk pengaturan kecepatan dan peringatan dini jika kendaraan berada dalam kondisi tidak aman

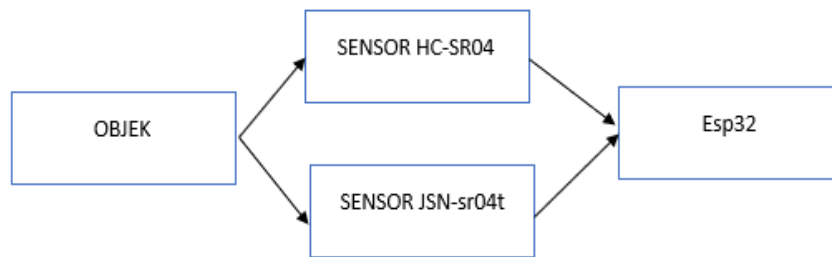
### 3.3. Pengujian Sensor HC-SR04

Untuk mempelajari lebih lanjut tentang kapasitas sensor ultrasonik HC-SR04 dalam mengukur jarak objek, pengujian dilakukan. Memperkirakan pendekatan objek pada jarak dekat, sedang, dan jauh. Akan lebih mudah jika anda mengetahui batas deteksi sensor ultrasonik HC-SR04. Berikut ini langkah-langkah dalam pengujian sensor ultrasonik HC-SR04.

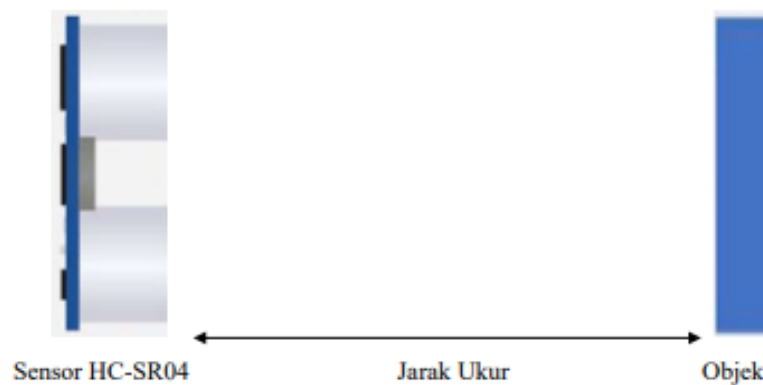
#### 3.3.1. Prosedur Pengujian Sensor HC-SR04 dan JSN-sr04t

Pengujian ini melibatkan input objek yang ditempatkan di depan sensor ultrasonik HC-SR04, yang akan dinilai berdasarkan jarak sebenarnya. Berikut adalah *representasi skema evaluasi* sensor ultrasonik yang melibatkan objek yang terletak pada jarak tertentu.





Gambar 4. 8 Diagram Blok Pengujian Sensor HC-SR04



Gambar 4. 9 Pengujian Sensor Terhadap Objek

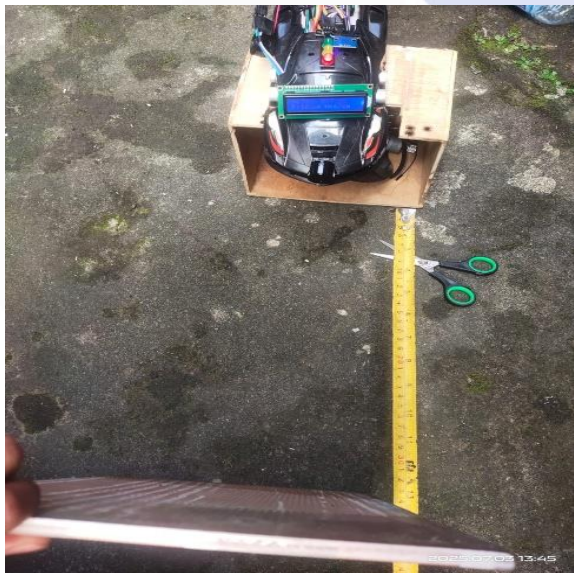
Untuk cara kerjanya apabila sensor mendeteksi objek di depannya maka sensor tersebut akan menangkap kembali *sinyal* yang di pantulkan pada objek kemudian *output* yang di hasilkan sensor akan di peroses oleh *microkontroler* dan dikonversikan dalam bentuk nilai berupa jarak antara sensor dengan objek yang dideteksinya.

### 3.3.2. Hasil Pengujian Sensor HC-SR04 dan JSN-sr04t

Pengujian ini mengumpulkan informasi dari benda yang terletak di depan sensor ultrasonik HC-SR04, yang kemudian akan dibandingkan dengan jarak sebenarnya. Berikut adalah *representasi skema* yang menunjukkan cara kerja pengujian sensor ultrasonik dengan benda yang ditempatkan pada jarak tertentu.



Gambar 4. 10 Pengujian Sensor HC-SR04



Gambar 4. 11 Pengujian Sensor JSN-SR04t

Pengujian dilakukan mulai dari jarak minimum 10 cm yang dapat dibaca oleh sensor ultrasonik dan berlanjut hingga objek berada pada jarak 300 cm, atau tidak dapat mendeteksinya lagi. Hasil pengujian sensor HC-SR 04 dan JSN-SR04t pada objek pada jarak tertentu ditampilkan dalam tabel di bawah ini.

Tabel 4. 1 Hasil Pengujian perbandingan sensor dengan Alat ukur meteran,

Input	Jarak terhadap objek	Output LCD	Selisih	Error	Rata-rata Error
JSN-SR04t	26 cm	26 cm	0 cm	0%	2,01 %
	40 cm	39 cm	1 cm	2,5%	
	60 cm	63 cm	3 cm	5%	
	80 cm	79 cm	1 cm	1,25%	
	100 cm	98 cm	2 cm	2%	
	150 cm	148 cm	2 cm	1,33%	
	200 cm	204 cm	4 cm	2%	
	300 cm	294 cm	6 cm	2%	
HC-SR04	20 cm	20 cm	0 cm	0%	2,49 %
	40 cm	40 cm	0 cm	0%	
	60 cm	65 cm	5 cm	3,33%	
	80 cm	81 cm	1 cm	1,25%	
	100 cm	98 cm	2 cm	2%	
	150 cm	145 cm	5 cm	3,33%	

*Persentase Error* (1,25%) hasil yang ditentukan dalam uji coba pengukuran sensor HC-SR04 dan JSN-sr04 ditemukan menggunakan rumus ini.

$$error (\%) = \frac{\text{Selisih}}{\text{Input}} \times 100\%$$

$$error (\%) = \frac{1 \text{ cm}}{80 \text{ cm}} \times 100\%$$

$$error (\%) = 1,25\%$$

Tingkat rata-rata *error* adalah 2.01% dan 2,49% untuk JSN-sr04t, seperti yang ditunjukkan oleh data dalam bagan *evaluasi* sensor HC-SR04 dan pembacaan jarak sebenarnya dari objek tersebut. Dari hasil pengujian, dapat disimpulkan bahwa kemampuan sensor HC-SR04 untuk mendeteksi jarak cukup akurat,

sehingga memungkinkan proyek untuk memanfaatkan sensor ultrasonik untuk mengukur jarak.

#### 4.4. Pengujian Sensor *Photo interrupter*

Pada tahap ini, sensor *Photo interrupter* dirancang untuk mendeteksi jumlah putaran poros motor DC yang kemudian *dikonversikan* menjadi satuan kecepatan kendaraan dalam km/jam. Rangkaian pengujian terdiri dari *mikrokontroler* ESP32 yang menerima *input* dari sensor optik, serta sebuah motor DC sebagai objek penggerak utama. Sensor dipasang pada bagian roda penggerak mobil mainan untuk mendeteksi tiap kali terdapat putaran penuh.

*Output* sensor berupa pulsa logika *hight dan low* yang akan diolah oleh *mikrokontroler* untuk menghitung *frekuensi* putaran dalam satuan rotasi per menit (RPM), lalu dikonversikan ke dalam satuan km/jam dengan menggunakan rumus konversi sesuai keliling roda mobil mainan yang digunakan.

##### 4.4.1. Prosedur Pengujian Sensor *Photo interrupter*

Langkah pengujian dilakukan dengan cara menjalankan motor DC pada berbagai tingkat kecepatan yang sudah ditentukan sebelumnya. Data RPM dari sensor *photo interrupter* dikonversi ke km/jam menggunakan *formula kalibrasi* berdasarkan diameter roda mobil.

Berbeda dari penelitian sebelumnya yang membandingkan akurasi sensor dengan *tachometer*, dalam pengujian ini sensor *photo interrupter* difokuskan untuk menentukan kategori kecepatan berdasarkan jarak tempuh kendaraan, yaitu *diklasifikasikan* menjadi tiga kategori kecepatan:

- **Jarak Dekat:** di bawah 30 km/jam
- **Jarak Sedang:** antara 30–60 km/jam
- **Jarak Jauh:** di atas 60 km/jam

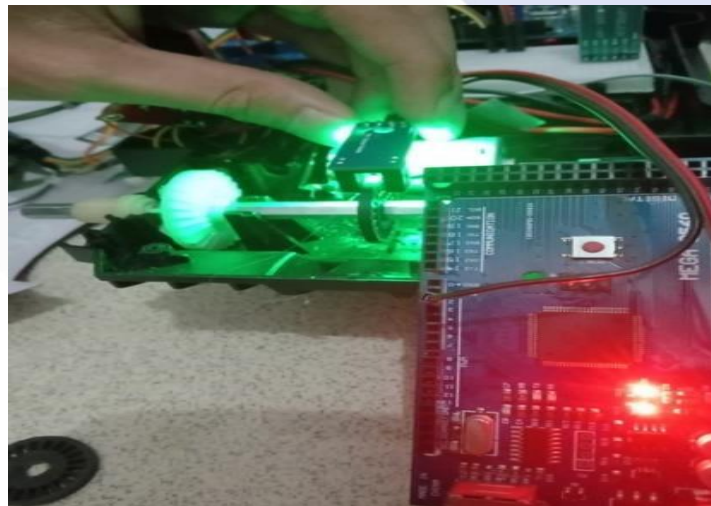
Sebagai bagian dari sistem peringatan dini, *buzzer* akan otomatis *aktif* ketika kecepatan mobil melebihi 60 km/jam meskipun jarak ke objek masih dalam kategori aman.



Gambar 4. 12 Diagram Pengujian Sensor *Photo interrupter*

#### 4.4.2. Hasil Pengujian Sensor *Photo interrupter*

Hasil dari pengujian menunjukkan bahwa sensor *photo interrupter* mampu membaca variasi kecepatan mobil dari posisi diam hingga lebih dari 60 km/jam secara *real-time*. Data kecepatan dalam satuan km/jam kemudian ditampilkan di LCD sebagai *indikator* kecepatan kendaraan.



Gambar 4. 13 Pengujian Sensor *photo interrupter*

Tabel 4. 2 Data Hasil Percobaan sensor *Photo interrupter* terhadap *Buzzer*

RPM Terbaca	Kecepatan (km/jam)	Kategori Jarak	Status Buzzer
501	20 km/jam	Dekat	Mati
796	30 km/jam	Sedang	Mati
1006	40 km/jam	Sedang	Mati
1569	60 km/jam	Jauh	Menyala

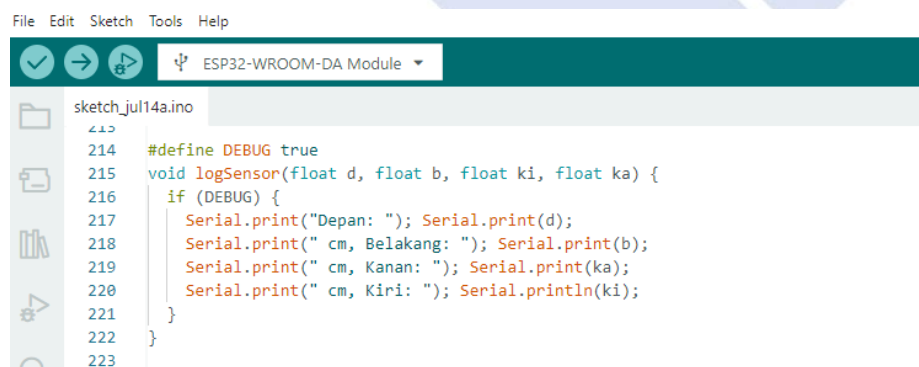
Selama pengujian, sistem memperlihatkan *respons* yang baik sesuai dengan rentang pengelompokan jarak yang telah ditentukan. Ketika kecepatan kendaraan mencapai lebih dari 60 km/jam, *buzzer aktif* sebagai peringatan bahaya. Sistem ini membantu dalam pengaturan kecepatan saat mobil berada dalam kondisi jarak jauh terhadap objek, untuk menghindari potensi kecelakaan akibat laju kendaraan yang terlalu tinggi.

#### 4.5. Perancangan dan Pembuatan *Software* Pengaturan Kecepatan dan Deteksi Jarak Aman Kendaraan

Tahap ini bertujuan untuk mengembangkan perangkat lunak (*software*) yang berfungsi sebagai pengendali utama dalam sistem deteksi jarak aman dan pengaturan kecepatan kendaraan. Seluruh proses pemrograman dilakukan menggunakan Arduino IDE, sedangkan aplikasi *Blynk* dimanfaatkan untuk kebutuhan *monitoring* data jarak, kecepatan, serta kendali sistem secara remote melalui *smartphone*. Dengan demikian, pengguna dapat memantau kondisi kendaraan dan *mengaktifkan* atau *menonaktifkan* sistem dari jarak jauh.

##### 4.5.1. Perancangan *Software*

Langkah awal dalam perancangan *software* ini adalah membuat logika pemrosesan data dari sensor jarak HC-SR04 dan JSN-SR04T dan sensor kecepatan *photo interrupter*. Seluruh logika pengolahan sensor, pengendalian motor DC, pengaktifan *buzzer* dan LED indikator ditulis dalam program berbasis bahasa pemrograman C/C++ di Arduino IDE.



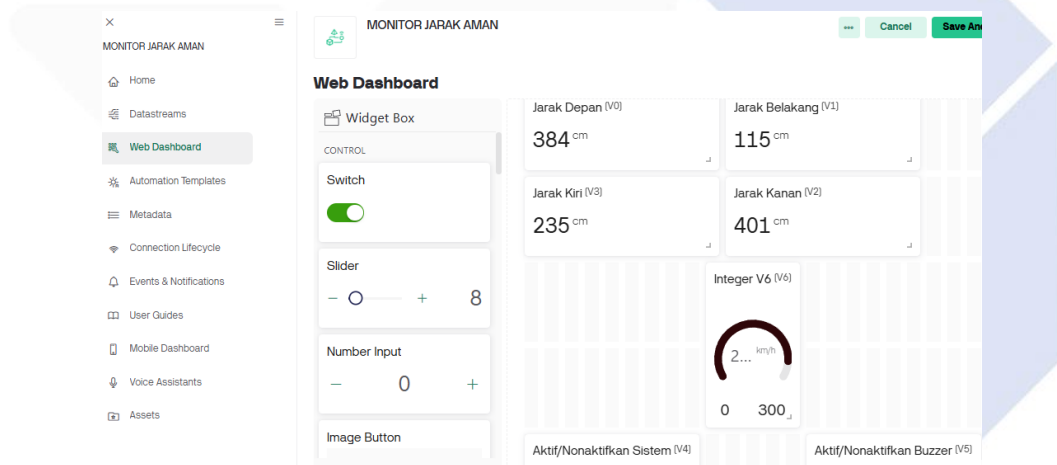
Gambar 4. 14 Tampilan Program Arduino IDE



Selain itu, perancangan *software* ini juga mencakup integrasi dengan aplikasi *Blynk*, yang berfungsi menampilkan data hasil pembacaan sensor dan status sistem secara *real-time* melalui jaringan *WiFi*. Dalam aplikasi *Blynk*, terdapat berbagai *widget* seperti *value display*, *gauge*, dan tombol *virtual button* yang digunakan untuk *monitoring* jarak, kecepatan, serta kendali *On* dan *Off* sistem.

#### 4.5.2. Pembuatan Software

Pada tahap implementasi ini, dilakukan proses penulisan program (*coding*) pada Arduino IDE, kemudian diunggah ke *mikrokontroler* ESP32. Program ini bertugas untuk membaca data sensor jarak dan kecepatan, mengontrol motor DC, serta mengirimkan data ke aplikasi *Blynk*. Selain sebagai pengirim data *monitoring*, ESP32 juga *dikonfigurasi* agar dapat menerima input dari *Blynk* berupa perintah *On* dan *Off* sistem yang dikirim melalui *virtual button*.



Gambar 4. 15 Tampilan Perancangan *Software* Aplikasi *Blynk*

##### 4.5.2.1. Logika Keputusan Program

Walaupun tidak menggunakan metode *fuzzy* Mamdani seperti pada penelitian lain, dalam sistem ini logika keputusan dibangun secara manual melalui percabangan *if-else* di dalam program Arduino. Logika ini mengatur *respon* sistem berdasarkan kombinasi antara data jarak dan kecepatan yang diterima dari sensor.

Contoh penerapan logika:

- Jika kecepatan tinggi, maka *buzzer* akan menyala, dan LED merah aktif.
- Jika jarak aman dan kecepatan rendah, maka sistem tetap berjalan normal tanpa peringatan.

- Untuk *opsional* disini bisa menggunakan pengaturan aplikasi *Blynk* untuk mengatur kecepatan/PWM

Selain itu, pengguna juga dapat mematikan atau mengaktifkan seluruh sistem melalui tombol *virtual* (V4) pada aplikasi *Blynk*.

#### **4.6. Pengujian *Software* Pengontrolan Kecepatan dan Jarak Aman Pada Kendaraan.**

Pengujian *software* dilakukan untuk memastikan bahwa program yang telah dikembangkan menggunakan Arduino IDE dapat berjalan sesuai dengan tujuan sistem. Pada tahapan ini, dilakukan serangkaian pengujian untuk mengetahui apakah data dari sensor ultrasonik dan sensor kecepatan dapat terbaca dengan baik oleh *mikrokontroler* ESP32, serta dapat terkirim dan ditampilkan pada aplikasi.

##### **4.6.1. Pembacaan data *Blynk* secara *Real-Time***

Pengujian ini juga melibatkan pengecekan fungsi-fungsi utama sistem, seperti:

- Pembacaan data jarak dari empat arah (depan, belakang, kiri, dan kanan) menggunakan sensor HC-SR04 dan JSN-SR04T.
- Pengolahan data kecepatan kendaraan yang diukur menggunakan sensor *photo interrupter*.
- Pengiriman data ke aplikasi *blynk* melalui koneksi *wifi* ESP32.
- Fungsi *On* dan *Off* sistem yang dapat dikontrol oleh pengguna dari aplikasi *Blynk*.
- Aktivasi *buzzer* dan LED indikator berdasarkan kondisi jarak dan kecepatan tertentu.

Hasil dari setiap pengujian akan diamati melalui dua media, yaitu:

1. Melalui serial monitor arduino IDE di sini data jarak, kecepatan, dan status sistem dapat diamati dalam bentuk teks langsung dari *mikrokontroler*.
2. Melalui dashboard aplikasi *Blynk* di *Smartphone* Tampilan *Blynk* akan menunjukkan nilai jarak dan kecepatan dalam bentuk *value display* atau *gauge*, serta status sistem dalam bentuk *indicator widget* dan *virtual button*.

Selama pengujian, sistem diuji dengan memberikan variasi kondisi jarak dan kecepatan. Jika jarak kendaraan terlalu dekat dengan objek di sekitarnya atau



kecepatan melebihi batas yang ditetapkan, maka sistem secara otomatis akan: Mengaktifkan *buzzer* sebagai *alarm* dan mengubah status LED *indikator* (misalnya, LED merah untuk kondisi bahaya) Selain itu, pengguna juga dapat mengaktifkan atau menonaktifkan sistem melalui tombol *virtual* di aplikasi *Blynk* yang terhubung langsung dengan ESP32. Secara keseluruhan, hasil pengujian menunjukkan bahwa *software* mampu menjalankan fungsi-fungsi *monitoring* dan kendali sesuai dengan perancangan awal.

#### **4.7. Pengujian Alat Keseluruhan**

Pengujian alat secara keseluruhan melibatkan upaya memastikan bahwa setiap komponen sistem terintegrasi dengan aplikasi *Blynk* dan beroperasi sesuai dengan tujuan yang diinginkan. Melalui fitur kontrol *dasbor*, aplikasi *Blynk* mengontrol kecepatan motor DC dan menampilkan data jarak dari sensor HC-SR04. Hasil pengujian menunjukkan bahwa kontrol kecepatan beroperasi secara *responsif* dan nilai jarak ditampilkan secara *real-time*. Lebih jauh lagi, saat sistem mensimulasikan pemutusan dan penyambungan kembali koneksi *internet*, sistem tetap stabil. Hal ini menunjukkan bahwa kinerja sistem dapat didukung oleh aplikasi *Blynk*.

##### **4.7.1. Pengujian Sensor Jarak Kiri, Kanan, Depan dan Belakang Mobil**

Untuk memeriksa Jarak, suara dari DFPlayer mini akan mengingatkan pengendara saat batas tercapai, dan pemasangan sensor diposisikan di sisi kiri, kanan, dan belakang kendaraan. pada pengujian ini masing-masing sensor akan di berikan objek di depannya dengan tujuan untuk mengetahui apakah sensor dapat bekerja dengan baik sekaligus uji coba DFPlayer mini apakah nyala apabila jarak ditentukan telah mencapai batas jarak yang di tentukan berikut ini data yang kami ambil berdasarkan uji coba yang telah dilakukan dapat dilihat pada tabel 4.3 di bawah ini.

Tabel 4. 3 Hasil Pengujian DFPlayer mini terhadap jarak kiri, kanan, depan, dan belakang

Posisi	Jarak Objek	DFPlayer mini
Depan	300 cm	Mati
	250 cm	Mati
	200 cm	Mati
	100 cm	Nyala
	50 cm	Nyala
	10 cm	Nyala
Belakang	300 cm	Mati
	250 cm	Mati
	200 cm	Mati
	100 cm	Nyala
	50 cm	Nyala
	10 cm	Nyala
Kiri	200 cm	Mati
	150 cm	Mati
	100 cm	Nyala
	75 cm	Nyala
	50 cm	Nyala
	25 cm	Nyala
	10 cm	Nyala
Kanan	200 cm	Mati
	150 cm	Mati
	100 cm	Nyala
	75 cm	Nyala
	50 cm	Nyala
	25 cm	Nyala
	0 cm	Nyala

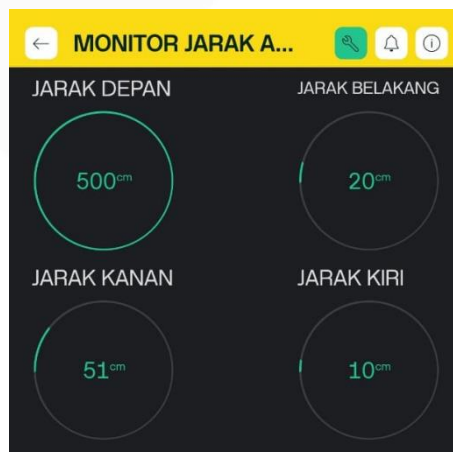
Dari tabel di atas terlihat jelas bahwa jika objek telah melewati batas aman, maka DFPlayer mini akan berbunyi. Jika jarak objek kurang dari 100 cm, maka *buzzer* di sisi kiri dan kanan mobil akan aktif. Sementara itu, jika jarak objek lebih dari 100 cm, maka *buzzer* mobil tidak akan nyala.

#### 4.7.2. Pengujian Sistem *Blynk*

Tujuan pengujian sistem berbasis web *Blynk* adalah untuk memastikan bahwa sistem dapat mengendalikan kecepatan dan melacak jarak secara *real-time*, dan bahwa komunikasi antara sensor, *mikrokontroler* (seperti Arduino/ESP32), dan aplikasi *Blynk* berfungsi dengan baik. Langkah-langkah umum untuk pengujian *Blynk* adalah sebagai berikut:

##### 4.7.2.1. Monitoring Data Sensor HC-SR04 dan JSN-SR04T

Data yang berhasil dikirim ke aplikasi *Blynk* oleh sensor jarak (HC-SR04). Secara *real time*, nilai jarak ditampilkan berdasarkan keadaan saat ini (misalnya, objek dipindahkan → nilai berubah).



Gambar 4. 16 Tampilan *Monitoring Jarak* Pada Aplikasi *Blynk*

Data yang ditampilkan kurang dari 100 cm maka *indikator* LED menyala berwarna merah dan DFPlayer akan berbunyi untuk pemberitahuan kepada pengemudi jarak yang ditentukan telah mencapai batas.

#### 4.7.2.2. Kontrol Kecepatan Melalui Aplikasi *Blynk*

Tombol atau *slider* *Blynk* secara *efektif* mengubah kecepatan motor DC. Reaksi fisik motor perubahan RPM yang dapat dikontrol sesuai dengan apa yang diinginkan untuk mensimulasikan apakah *buzzer* berfungsi apabila sudah mencapai batas.



Gambar 4.17 Tampilan *Monitoring* Kecepatan Pada Aplikasi *Blynk*

Apabila kecepatan yang ditentukan telah mencapai batas maka *buzzer* akan menyala sebagai *indicator* peringatan kepada pengemudi untuk mengurangi kecepatan.

## BAB V

### KESIMPULAN DAN SARAN

#### 5.1. Kesimpulan

Setelah selesai pembuatan alat dan telah dilakukan pengujian, maka hasil dari proyek akhir ini yaitu pembuatan *prototype* kontrol kecepatan dan deteksi jarak aman kendaraan berbasis IoT ini dapat disimpulkan bahwa dengan menggunakan sensor ultrasonik HC-SR04 dan JSN-SR04T, sistem yang dikembangkan ini secara efektif mengukur jarak secara *real-time* antara kendaraan dan objek di sekitar bagian depan, belakang, kiri, dan kanan.

1. Platform *Blynk* bekerja secara efektif sebagai alat pemantauan, yang memungkinkan data jarak sensor dikirimkan dan ditampilkan dengan benar dan cepat melalui aplikasi telepon pintar.
2. Jika jarak yang terdeteksi kurang dari ambang batas aman yang telah ditetapkan, sistem akan segera memberi tahu atau memperingatkan pengguna menggunakan LED, *buzzer*, dan aplikasi *Blynk*.
3. Terdapat kekurangan dari *prototype* yang dimana *prototype* masih ketergantungan penuh terhadap koneksi *WiFi* yang menyebabkan sistem tidak berfungsi maksimal.
4. Pengemudi dapat lebih waspada terhadap kemungkinan tabrakan dengan adanya teknologi ini, terutama saat berkendara di daerah yang padat. Pendekatan ini dapat menjadi cara yang bermanfaat untuk meningkatkan keselamatan berkendara.
5. Aplikasi *Blynk* memiliki kontrol berbasis slider atau tombol untuk mengatur kecepatan motor. Saat input aplikasi berubah, kecepatan motor DC bereaksi secara langsung dan stabil, yang menunjukkan bahwa sistem kontrol kecepatan berfungsi dengan baik.
6. Sistem belum mampu untuk mengendalikan kecepatan kendaraan secara otomatis sistem ini masih bersifat manual melalui aplikasi.
7. Seluruh sistem, termasuk perangkat lunak (*Blynk* dan Arduino IDE) dan

perangkat keras (sensor, motor, dan ESP32), berhasil *diintegrasikan* dan dilakukan secara *konsisten* selama pengujian.

## 5.2. Saran

1. *Konektivitas* ke sistem kendaraan yang telah dirancang masih berupa *prototype*. Disarankan agar sistem ini dikombinasikan dengan mobil sungguhan untuk pengembangan di masa mendatang, termasuk tautan ke unit kontrol *elektronik* (ECU) untuk kecepatan waktu nyata dan kontrol pengereman yang lebih baik.
2. Menggunakan *platform* IoT yang lebih aman dan stabil meskipun *Blynk* cukup untuk aplikasi pemantauan dasar, *platform* IoT dengan tingkat keamanan dan keandalan yang lebih tinggi disarankan untuk penggunaan yang lebih lama atau *ekstensif*.
3. Pengujian dalam berbagai kondisi cuaca dan jalan saat ini, hanya beberapa kondisi lingkungan yang digunakan untuk pengujian. Untuk memastikan sistem terus berfungsi dengan baik, diperlukan lebih banyak pengujian dalam berbagai kondisi cuaca (panas, hujan, lubang, dan jalan licin).

## DAFTAR PUSTAKA

- [1] Badan Pusat Statistik. (2023, 29 Februari). Jumlah kendaraan bermotor menurut provinsi dan jenis kendaraan – 2023. Diakses pada 7 Juni 2024, <https://www.bps.go.id/id/statisticstable/3/VjJ3NGRGa3dkRk5MTIU1bVNFOtVVbmQyVURSTVFUMDkjMw==>
- [2] Sugiharti, E., Saleh, M., & Mustikasari. (2019). Analisis faktor potensi penyebab kecelakaan lalu lintas kendaraan bermotor. *Jurnal Manajemen Bisnis Transportasi dan Logistik*, 5(3), 367–374.
- [3] Najib, W., Sulisty, S., & Widyawan. (2020). Tinjauan ancaman dan solusi keamanan pada teknologi Internet of Things (IoT). *Jurnal Nasional Teknik Elektro dan Teknologi Informasi*, 9(4), 375–384.
- [4] Umar, A. (2022). Penanganan perkara kecelakaan lalu lintas yang mengakibatkan korban luka ringan di wilayah hukum Kepolisian Resor Jepara [Skripsi, Universitas Islam Nahdlatul Ulama Jepara].
- [5] Agam, O. (2019). Analisis tingkat keselamatan penyeberang menggunakan Pedestrian Risk Index (PRI). Diakses pada 7 Juni 2024, dari <https://repository.uir.ac.id/8828/>
- [6] Widiaputra, P. M., Wisanggeni, S. P., & Krishna, A. (2022). Pengembangan sistem LiDAR pendeteksi jarak aman berkendara. *Jurnal Rekayasa Sipil dan Lingkungan*, 6(2), 190–198. <https://doi.org/10.19184/jrsl.v6i2.36607>.
- [7] Nahdi, F., & Dhika, H. (2021). Analisis dampak Internet of Things (IoT) pada perkembangan teknologi di masa yang akan datang. *INTEGER: Jurnal Information Technology*, 6(1), 33–40. <https://doi.org/10.31284/j.integer.2021.v6i1.1423>.
- [8] Judijanto, L., Hiswara, A., Aini, M. A., & Nanjar, A. (2024). Pengaruh implementasi Internet of Things terhadap pengambilan keputusan bisnis pada perusahaan teknologi di Jakarta. *Jurnal Multidisiplin West Science*, 3(3), 389–397. <https://doi.org/10.58812/jmws.v3i03.1075>

- [9] Selay, A., dkk. (2022). IoT dalam perspektif tauhid. *Karimah Tauhid*, 1(6), 861–862.
- [10] Mukhammad, Y., Santika, A., & Haryuni, S. (2022). Analisis akurasi modul amplifier HX711 untuk timbangan bayi. *MedTek: Jurnal Teknik Elektromedik Indonesia*, 4(1), 24–28. <https://doi.org/10.18196/mt.v4i1.15148>
- [11] Soni, A., & Aman, A. (2018). Distance measurement of an object by using ultrasonic sensors with Arduino and GSM module. *IJSTE: International Journal of Science Technology & Engineering*, 4(11), 23–28. Diakses pada 7 Juni 2024, dari [www.ijste.org](http://www.ijste.org)
- [12] Putra, I. U., Saefulloh, S., Bakri, M., & Darwis, D. (2022). Pengukur tinggi badan digital ultrasonik berbasis Arduino dengan LCD dan output suara. *Jurnal Teknik dan Sistem Komputer*, 2(2), 1–14. <https://doi.org/10.33365/jtikom.v2i2.69>
- [13] Wiguna, R. (2023). Rancang bangun modul pembelajaran Internet of Things (IoT) berbasis ESP32. Diakses pada 7 Juni 2024.
- [14] Djaksana, Y. M., & Gunawan, K. (2021). Perancangan sistem monitoring dan kontroling pompa air berbasis Android. *SINTECH (Science and Information Technology Journal)*, 4(2), 146–154. <https://doi.org/10.31598/sintechjournal.v4i2.741>
- [15] Nyebar Ilmu. (2023). Sensor Ultrasonik JSN-SR04T. Diakses pada 7 Juni 2024, dari <https://www.nyebarilmu.com/tutorial-mengakses-sensor-ultrasonic-jsn-sr04t/>



**LAMPIRAN 1**  
**(Riwayat Hidup Perorangan)**



## DAFTAR RIWAYAT HIDUP

### 1. Data Pribadi

Nama lengkap : Hebri  
Tempat & tanggal lahir : Dendang, 8 Februari 2004  
Alamat rumah : Desa Dendang  
Telp: 083157568491  
Hp: 083157568491  
Email: [heprihepri203@gmail.com](mailto:heprihepri203@gmail.com)  
Jenis kelamin : Laki - laki  
Agama : Islam



### 2. Riwayat Pendidikan

1. SD Negeri 1 Dendang	Lulus 2016
2. SMP Negeri 2 Dendang	Lulus 2019
3. SMK Negeri 1 Dendang	Lulus 2022
4. Politeknik Manufaktur Negeri Bangka Belitung	2025 – Sekarang

### 3. Pendidikan Non-Formal

-

Sungailiat, 11 juli 2025

Hebri

## DAFTAR RIWAYAT HIDUP

### 1. Data Pribadi

Nama lengkap : Candra  
Tempat & tanggal lahir : KERETAK, 22 Oktober 2000  
Alamat rumah : JL. SUNGAISELAN  
Telp: 081279908961  
Hp: 081279908961  
Email: [chandrasam221100@gmail.com](mailto:chandrasam221100@gmail.com)  
Jenis kelamin : Laki - laki  
Agama : Islam



### 2. Riwayat Pendidikan

1. SD Negeri 5 SUNGAISELAN	Lulus 2016
2. SMP Negeri 6 SUNGAISELAN	Lulus 2019
3. SMK Negeri 1 SIMPANG KATIS	Lulus 2022
4. Politeknik Manufaktur Negeri Bangka Belitung	2025 – Sekarang

### 3. Pendidikan Non-Formal

-

Sungailiat, 11 juli 2025

Candra



## **LAMPIRAN 2**

### **(KODE PROGRAM)**

## PROGRAM ARDUINO

```
#define BLYNK_TEMPLATE_ID "TMPL6oW-xhlg_"
#define BLYNK_TEMPLATE_NAME "MONITOR JARAK AMAN"
#define BLYNK_AUTH_TOKEN "KJPULiv3wVj_8GfkMzLt7lx6Ahx18xD4"

#include <WiFi.h>
#include <BlynkSimpleEsp32.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <EEPROM.h>
#include "DFRobotDFPlayerMini.h"

// ==== DFPLAYER MINI CONFIG ====
#define RXD2 33
#define TXD2 32
HardwareSerial mySerial(2);
DFRobotDFPlayerMini myDFPlayer;

// WiFi
char ssid[] = "ROG PHONE 9 PRO";
char pass[] = "NNNNNNN6";

// EEPROM
#define EEPROM_SIZE 4
#define EEPROM_ADDR 0
#define ADDR_BUZZER_ENABLED 1
#define ADDR_DFPLAYER_ENABLED 2

// LCD I2C
LiquidCrystal_I2C lcd(0x27, 16, 2);

// Sensor JSN-SR04T
#define TRIG_DEPAN 19
#define ECHO_DEPAN 18
#define TRIG_BELAKANG 5
#define ECHO_BELAKANG 17
#define TRIG_KIRI 16
#define ECHO_KIRI 4
#define TRIG_KANAN 2
#define ECHO_KANAN 15

// LED & Buzzer
#define LED_MERAH 27
#define LED_KUNING 26
#define LED_HIJAU 25
#define BUZZER_PIN 13
```

```

// Encoder Kecepatan (HC-020K)
#define ENCODER_PIN 12
volatile unsigned long pulseCount = 0;
unsigned long lastSpeedCheck = 0;
float kecepatan = 0.0;
const float RODA_KELILING = 0.20;
const int PULSES_PER_REV = 20;

bool buzzerEnabled = true;
bool systemActive = true;
bool dfplayerEnabled = true;

float batasKecepatan = 80.0; // default, bisa diubah dari Blynk

BlynkTimer timer;

void IRAM_ATTR countPulse() {
    pulseCount++;
}

float readDistance(int trigPin, int echoPin) {
    digitalWrite(trigPin, LOW); delayMicroseconds(2);
    digitalWrite(trigPin, HIGH); delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    long duration = pulseIn(echoPin, HIGH, 30000);
    if (duration == 0) return 999.0;
    float distance = duration * 0.034 / 2;
    return (distance > 999.0) ? 999.0 : distance;
}

void hitungKecepatan() {
    static unsigned long lastPulseCount = 0;
    unsigned long now = millis();
    if (now - lastSpeedCheck >= 1000) {
        noInterrupts();
        unsigned long currentCount = pulseCount;
        interrupts();

        unsigned long deltaCount = currentCount - lastPulseCount;
        lastPulseCount = currentCount;

        float putaranPerDetik = deltaCount / (float)PULSES_PER_REV;
        float jarakPerDetik = putaranPerDetik * RODA_KELILING;
        kecepatan = jarakPerDetik * 3.6;
    }
}

```

```

    Blynk.virtualWrite(V6, kecepatan);
    lastSpeedCheck = now;
  }
}

void tampilkanLCD(float d, float b, float ki, float ka) {
  static String lastLine1 = "", lastLine2 = "";
  char line1[17], line2[17];
  snprintf(line1, sizeof(line1), "D:%.0fcm B:%.0fcm", d, b);
  snprintf(line2, sizeof(line2), "KI:%.0fcm KA:%.0fcm", ki, ka);

  String newLine1 = String(line1);
  String newLine2 = String(line2);

  if (newLine1 != lastLine1) {
    lcd.setCursor(0, 0); lcd.print("          ");
    lcd.setCursor(0, 0); lcd.print(newLine1);
    lastLine1 = newLine1;
  }

  if (newLine2 != lastLine2) {
    lcd.setCursor(0, 1); lcd.print("          ");
    lcd.setCursor(0, 1); lcd.print(newLine2);
    lastLine2 = newLine2;
  }
}

void updateTrafficLight(float minDistance) {
  digitalWrite(LED_MERAH, minDistance < 150.0);
  digitalWrite(LED_KUNING, minDistance >= 150.0 && minDistance < 250.0);
  digitalWrite(LED_HIJAU, minDistance >= 250.0);
}

void kirimDataSensorKeBlynk() {
  float d = readDistance(TRIG_DEPAN, ECHO_DEPAN);
  float b = readDistance(TRIG_BELAKANG, ECHO_BELAKANG);
  float ka = readDistance(TRIG_KANAN, ECHO_KANAN);
  float ki = readDistance(TRIG_KIRI, ECHO_KIRI);

  Blynk.virtualWrite(V0, d);
  Blynk.virtualWrite(V1, b);
  Blynk.virtualWrite(V2, ka);
  Blynk.virtualWrite(V3, ki);
  Blynk.virtualWrite(V6, kecepatan);
  Blynk.virtualWrite(V9, batasKecepatan); // kirim batas kecepatan
}

```

```

Serial.print("Depan: "); Serial.print(d); Serial.print(" cm, ");
Serial.print("Belakang: "); Serial.print(b); Serial.print(" cm, ");
Serial.print("Kanan: "); Serial.print(ka); Serial.print(" cm, ");
Serial.print("Kiri: "); Serial.print(ki); Serial.print(" cm, ");
Serial.print("Kecepatan: "); Serial.print(kecepatan); Serial.println(" km/jam");
}

BLYNK_WRITE(V4) {
  systemActive = param.asInt();
  EEPROM.write(EEPROM_ADDR, systemActive);
  EEPROM.commit();
}

BLYNK_WRITE(V5) {
  buzzerEnabled = param.asInt();
  EEPROM.write(ADDR_BUZZER_ENABLED, buzzerEnabled);
  EEPROM.commit();
}

BLYNK_WRITE(V7) {
  dfplayerEnabled = param.asInt();
  EEPROM.write(ADDR_DFPLAYER_ENABLED, dfplayerEnabled);
  EEPROM.commit();
}

BLYNK_WRITE(V8) {
  batasKecepatan = param.asFloat();
  Serial.print("Batas kecepatan baru dari Blynk: ");
  Serial.println(batasKecepatan);
}

void setup() {
  Serial.begin(115200);
  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);

  EEPROM.begin(EEPROM_SIZE);
  systemActive = EEPROM.read(EEPROM_ADDR);
  buzzerEnabled = EEPROM.read(ADDR_BUZZER_ENABLED);
  dfplayerEnabled = EEPROM.read(ADDR_DFPLAYER_ENABLED);

  Serial.print("EEPROM systemActive: "); Serial.println(systemActive);
  Serial.print("EEPROM buzzerEnabled: "); Serial.println(buzzerEnabled);
  Serial.print("EEPROM dfplayerEnabled: "); Serial.println(dfplayerEnabled);

  pinMode(TRIG_DEPAN, OUTPUT);  pinMode(ECHO_DEPAN, INPUT);
  pinMode(TRIG_BELAKANG, OUTPUT); pinMode(ECHO_BELAKANG,

```



```

INPUT);
pinMode(TRIG_KANAN, OUTPUT); pinMode(ECHO_KANAN, INPUT);
pinMode(TRIG_KIRI, OUTPUT); pinMode(ECHO_KIRI, INPUT);

pinMode(LED_MERAH, OUTPUT);
pinMode(LED_KUNING, OUTPUT);
pinMode(LED_HIJAU, OUTPUT);
pinMode(BUZZER_PIN, OUTPUT);
digitalWrite(LED_MERAH, LOW);
digitalWrite(LED_KUNING, LOW);
digitalWrite(LED_HIJAU, LOW);
digitalWrite(BUZZER_PIN, LOW);

pinMode(ENCODER_PIN, INPUT_PULLUP);
attachInterrupt(digitalPinToInterrupt(ENCODER_PIN), countPulse, RISING);

Wire.begin(21, 22);
lcd.begin(16, 2);
lcd.backlight();

mySerial.begin(9600, SERIAL_8N1, RXD2, TXD2);
if (!myDFPlayer.begin(mySerial)) {
  Serial.println("DFPlayer gagal terhubung!");
  while (true);
}
myDFPlayer.volume(50);

timer.setInterval(1000L, kirimDataSensorKeBlynk);
}

#define DEBUG true
void logSensor(float d, float b, float ki, float ka) {
  if (DEBUG) {
    Serial.print("Depan: "); Serial.print(d);
    Serial.print(" cm, Belakang: "); Serial.print(b);
    Serial.print(" cm, Kanan: "); Serial.print(ka);
    Serial.print(" cm, Kiri: "); Serial.println(ki);
  }
}

unsigned long lastPlayTime = 0;
const unsigned long playInterval = 3000; // 3 detik jeda antar suara
unsigned long lastDebugTime = 0;

void loop() {
  Blynk.run();

```

```

timer.run();
hitungKecepatan();

unsigned long now = millis();
if (now - lastDebugTime > 1000) {
    Serial.println("Loop berjalan... systemActive: " + String(systemActive));
    lastDebugTime = now;
}

if (kecepatan >= batasKecepatan) {
    if (buzzerEnabled) {
        digitalWrite(BUZZER_PIN, HIGH);
    } else {
        digitalWrite(BUZZER_PIN, LOW);
    }
    lcd.setCursor(0, 0); lcd.print("!! KECEPATAN !! ");
    lcd.setCursor(0, 1);
    lcd.print(" >"); lcd.print(batasKecepatan); lcd.print(" km/jam ");
    return;
} else {
    digitalWrite(BUZZER_PIN, LOW);
}

if (systemActive) {
    float jarakDepan = readDistance(TRIG_DEPAN, ECHO_DEPAN);
    float jarakBelakang = readDistance(TRIG_BELAKANG,
ECHO_BELAKANG);
    float jarakKanan = readDistance(TRIG_KANAN, ECHO_KANAN);
    float jarakKiri = readDistance(TRIG_KIRI, ECHO_KIRI);

    logSensor(jarakDepan, jarakBelakang, jarakKanan, jarakKiri);

    float minDist = min(min(jarakDepan, jarakBelakang), min(jarakKanan,
jarakKiri));
    updateTrafficLight(minDist);

    if (minDist < 150.0 && dfplayerEnabled) {
        unsigned long currentTime = millis();
        if (currentTime - lastPlayTime > playInterval) {
            if (minDist == jarakDepan) myDFPlayer.play(1);
            else if (minDist == jarakBelakang) myDFPlayer.play(2);
            else if (minDist == jarakKiri) myDFPlayer.play(3);
            else if (minDist == jarakKanan) myDFPlayer.play(4);
            lastPlayTime = currentTime;
        }
    }
}

```

```
tampilkanLCD(jarakDepan, jarakBelakang, jarakKanan, jarakKiri);
}
else {
    digitalWrite(LED_MERAH, LOW);
    digitalWrite(LED_KUNING, LOW);
    digitalWrite(LED_HIJAU, LOW);
    digitalWrite(BUZZER_PIN, LOW);
    lcd.setCursor(0, 0); lcd.print("...SISTEM OFF...");
    lcd.setCursor(0, 1); lcd.print("Aktifkan Sistem ");
}

delay(200);
}
```



# 18%

SIMILARITY INDEX

### PRIMARY SOURCES

1	<a href="https://repository.polman-babel.ac.id">repository.polman-babel.ac.id</a> Internet	589 words — 10%
2	<a href="https://jurnal.unej.ac.id">jurnal.unej.ac.id</a> Internet	54 words — 1%
3	<a href="https://middelkamp.gerahrtekunst.de">middelkamp.gerahrtekunst.de</a> Internet	45 words — 1%
4	<a href="https://www.akreditasi.org">www.akreditasi.org</a> Internet	40 words — 1%
5	<a href="https://tekno.tempo.co">tekno.tempo.co</a> Internet	33 words — 1%
6	Dian Nova Kusuma Hardani, Latiful Hayat. "Penerapan Internet of Things (IoT) pada Sistem Pengendali dan Pengaman Pintu Berbasis Android", Jurnal Riset Rekayasa Elektro, 2020 Crossref	20 words — < 1%
7	Noer Soedjarwanto. "Prototipe Smart Door Lock Menggunakan Motor Stepper Berbasis IoT (Internet Of Things)", Electrician, 2021 Crossref	20 words — < 1%
8	<a href="https://repository.unmuhjember.ac.id">repository.unmuhjember.ac.id</a> Internet	18 words — < 1%

9	<a href="http://vdocuments.mx">vdocuments.mx</a> Internet	14 words — < 1%
10	<a href="http://datadosen.com">datadosen.com</a> Internet	13 words — < 1%
11	<a href="http://docplayer.info">docplayer.info</a> Internet	13 words — < 1%
12	<a href="http://dspace.ups.edu.ec">dspace.ups.edu.ec</a> Internet	13 words — < 1%
13	<a href="http://repository.ppns.ac.id">repository.ppns.ac.id</a> Internet	13 words — < 1%
14	<a href="http://data.epo.org">data.epo.org</a> Internet	12 words — < 1%
15	<a href="http://repository.untag-sby.ac.id">repository.untag-sby.ac.id</a> Internet	12 words — < 1%
16	<a href="http://pressenumerique.viapresse.com">pressenumerique.viapresse.com</a> Internet	11 words — < 1%
17	<a href="http://journal.unj.ac.id">journal.unj.ac.id</a> Internet	10 words — < 1%
18	<a href="http://repository.ittelkom-pwt.ac.id">repository.ittelkom-pwt.ac.id</a> Internet	10 words — < 1%
19	<a href="http://www.sal-4u.de">www.sal-4u.de</a> Internet	10 words — < 1%
20	<a href="http://fe-akuntansi.unila.ac.id">fe-akuntansi.unila.ac.id</a> Internet	9 words — < 1%
21	<a href="http://id.scribd.com">id.scribd.com</a> Internet	9 words — < 1%

22	<a href="http://www.coursehero.com">www.coursehero.com</a> Internet	9 words — < 1%
23	<a href="http://core.ac.uk">core.ac.uk</a> Internet	8 words — < 1%
24	<a href="http://digilib.uin-suka.ac.id">digilib.uin-suka.ac.id</a> Internet	8 words — < 1%
25	<a href="http://expertsinuncertainty.net">expertsinuncertainty.net</a> Internet	8 words — < 1%
26	<a href="http://repository.its.ac.id">repository.its.ac.id</a> Internet	8 words — < 1%
27	<a href="http://www.kelasteknisi.com">www.kelasteknisi.com</a> Internet	8 words — < 1%
28	<a href="http://www.manorshi.com">www.manorshi.com</a> Internet	8 words — < 1%
29	<a href="http://www.scribd.com">www.scribd.com</a> Internet	8 words — < 1%
30	<a href="http://jesty.pubmedia.id">jesty.pubmedia.id</a> Internet	7 words — < 1%

EXCLUDE QUOTES OFF

EXCLUDE BIBLIOGRAPHY OFF

EXCLUDE SOURCES OFF

EXCLUDE MATCHES OFF