RANCANG BANGUN SISTEM KEAMANAN PEKERJA PROYEK DENGAN HELM PINTAR BERBASIS ESP

SKRIPSI

Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh
Gelar Komputer

DI AJUKAN OLEH :

FARHAN KENNEDY
21101152620019

JURUSAN SISTEM KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS PUTRA INDONESIA "YPTK" PADANG

PADANG, 2025

LEMBAR PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : FARHAN KENNEDY

No. BP : 21101152620019

Fakultas : ILMU KOMPUTER

Jurusan : SISTEM KOMPUTER

Menyatakan bahwa :

1. Sesungguhnya skripsi yang saya susun ini merupakan hasil karya tulis saya sendiri. Adapun bahagian-bahagian tertentu dalam skripsi yang saya peroleh dan hasil karya tulis orang lain, telah saya tuliskan sumbernya dengan jelas, sesuai dengan kaidah penulisan ilmiah.
2. Jika dalam pembuatan skripsi baik pembuatan program maupun skripsi secara keseluruhan terbukti dibuatkan oleh orang lain, maka saya bersedia menerima sanksi yang diberikan akademik, berupa pembatalan skripsi dan mengulang penelitian serta mengajukan judul baru.

Demikian surat pernyataan ini saya buat dengan sesungguhnya tanpa ada paksaan dari pihak manapun.

Padang, Agustus 2025

FARHAN KENNEDY
21101152620019

HALAMAN PENGESAHAN SKRIPSI

RANCANG BANGUN SISTEM KEAMANAN PEKERJA PROYEK DENGAN HELM PINTAR BERBASIS ESP

Yang dipersiapkan dan disusun oleh

FARHAN KENNEDY
21101152620019

Telah dipertahankan di depan dewan penguji

Pada tanggal :……..

dan dinyatakan telah lulus memenuhi syarat

Pembimbing I	**Pembimbing II**
(Ir. Emil Naf'an, S.Kom., M.Kom., Ph.D)	**(Riska Robianto, S.Kom.,M.Kom.)**
NIDN : 1017127401	**NIDN : 1004077904**

Padang, …….

Dekan Fakultas Ilmu Komputer

Universitas Putra Indonesia "YPTK" Padang

(Dr. Rini Sovia, S.Kom., M.Kom)

NIDN : 1005047601

HALAMAN PENGESAHAN PENGUJI SIDANG SKRIPSI

RANCANG BANGUN SISTEM KEAMANAN PEKERJA PROYEK DENGAN HELM PINTAR BERBASIS ESP

OLEH

FARHAN KENNEDY
21101152620019

PROGRAM STUDI SISTEM KOMPUTER

Skripsi ini telah dinyatakan LULUS oleh Penguji Materi Pada Sidang Skripsi
Program Studi Sistem Komputer Strata I Fakultas Ilmu Komputer
Universitas Putra Indonesia "YPTK" Padang
Pada Hari/ Tanggal: . . . . / . . . . . /2025

TIM PENGUJI

1. **Dr. Ondra Eka Putra, S.Kom., M.Kom.** **……**
   NIDN : 1006068701

2. **Sepsa Nur Rahman S.Kom., M.Kom.** **……**
   NIDN : 1028099201

Padang, Agustus 2025

Mengetahui
Dekan Fakultas Ilmu Komputer
Universitas Putra Indonesia "YTPK" Padang

(Dr. Rini Sovia, S.Kom., M.Kom.)
NIDN : 1005047601

HALAMAN PENGESAHAN LULUS SIDANG SKRIPSI

RANCANG BANGUN SISTEM KEAMANAN PEKERJA PROYEK DENGAN HELM PINTAR BERBASIS ESP

Yang dipersiapkan dan disusun oleh

FARHAN KENNEDY
21101152620019

Telah dipertahankan di depan dewan penguji

Pada tanggal :……..

dan dinyatakan telah lulus memenuhi syarat

Pembimbing I	**Pembimbing II**
(Ir. Emil Naf'an, S.Kom., M.Kom., Ph.D.)	**(Riska Robianto, S.Kom.,M.Kom.)**
NIDN : 1017127401	**NIDN : 1004077904**

Padang, …….
Dekan Fakultas Ilmu Komputer
Universitas Putra Indonesia "YPTK" Padang

(Dr. Rini Sovia, S.Kom, M.Kom)
NIDN : 1005047601

ABSTRACT

Thesis Title	**: RANCANG BANGUN SISTEM KEAMANAN PEKERJA PROYEK DENGAN HELM PINTAR BERBASIS ESP**
Student Name	**: FARHAN KENNEDY**
Student Number	**: 21101152620019**
Study Program	**: Computer Engineering**
Degree Granted	**: Strata 1 (S1)**
Adivisor	**: 1. Ir. Emil Naf'an, S.Kom., M.Kom., Ph.D**
	2. Riska Robianto, S.Kom.,M.Kom

Occupational safety in construction projects is crucial to reduce the risk of accidents. This research designed an ESP32-based smart helmet equipped with an MPU6050 sensor for fall detection, a MAX30102 for heart rate, an MQ-135 for hazardous gases, and GPS for location tracking. The system provides alerts via a Buzzer, LED, Vibration motor, and Telegram notifications. Test results showed the system was able to detect hazards and send notifications in real time. This helmet is expected to improve worker safety in the field.

***Keywords**: Smart Helmet, ESP32, Sensors, Project Security, Telegram.*

ABSTRACT

Judul Skripsi : **RANCANG BANGUN SISTEM KEAMANAN PEKERJA PROYEK DENGAN HELM PINTAR BERBASIS ESP**

Nama : **FARHAN KENNEDY**

No BP : **21101152620019**

Program Studi : **Computer Engineering**

Jenjang Pendidikan : **Strata 1 (S1)**

Pembimbing : **1. Ir. Emil Naf'an, S.Kom., M.Kom., Ph.D**

2. Riska Robianto, S.Kom.,M.Kom

Keselamatan kerja di proyek konstruksi sangat penting untuk mengurangi risiko kecelakaan. Penelitian ini merancang helm pintar berbasis ESP32 yang dilengkapi sensor MPU6050 untuk deteksi jatuh, MAX30102 untuk detak jantung, MQ-135 untuk gas berbahaya, dan GPS untuk pelacakan lokasi. Sistem memberikan peringatan melalui *Buzzer*, LED, motor getar, dan notifikasi ke Telegram. Hasil pengujian menunjukkan sistem mampu mendeteksi bahaya dan mengirimkan notifikasi secara real-time. Helm ini diharapkan dapat meningkatkan keamanan pekerja di lapangan.

Kata Kunci: Helm Pintar, ESP32, Sensor, Keamanan Proyek, Telegram.

KATA PENGANTAR

Puji syukur penulis ucapkan kepada Allah S.W.T yang telah memberikan rahmat dan karunianya, sehingga bisa menyelesaikan penulisan skripsi penulis dengan judul **"Rancang Bangun Sistem Keamanan Pekerja Proyek Dengan Helm Pintar Berbasis ESP"**

Penulisan laporan ini bertujuan dalam rangka untuk memperoleh gelar Sarjana Komputer pada Fakultas Ilmu Komputer Universita Putra Indonesia "YPTK" Padang.

Penulisan laporan ini diwujudkan dalam bentuk suatu perancangan alat, yang tidak lepas dari kata kekurangan, keterbatasan, dan jauh dari kata sempurna, baik dari segi penulisan, bahasa, dan alatnya. Hal ini dikarenakan oleh ilmu, pengetahuan dan pengalaman penulis miliki masih kurang serta keterbatasan fasilitas yang disediakan. Oleh karena itu penulis berharap adanya kritik saran yang mendukung agar dapat diperbaiki kembali dan sempurna hendaknya. Selanjutnya didalam penulisan laporan ini penulis sadar bahwa keberhasilan penulis dalam menyelesaikan skripsi ini berkat dari dorongan dan bimbingan dari pembimbing, orang tua, teman, dan berbagai pihak, oleh karena itu penulis ingin mengucapkan rasa terima kasih, khususnya kepada:

1. Bapak **H. Herman Nawas** (Alm) selaku pendiri yayasan perguruan tinggi komputer (yptk) padang yang menaungi Universitas Putra Indonesia YPTK Padang.
2. Ibu **Dr. Hj. Zerni Melmusi, S.E, M.M, Ak, C.A** sebagai Ketua Pembina Yayasan Perguruan Tinggi Komputer Padang
3. Ibu **Sitti Rizki Mulyani, S.Pd., MM** sebagai Ketua Yayasan Perguruan Tinggi Komputer (YPTK) Padang

4. Bapak **Assoc. Prof. Dr. Muhammad Ridwan, SE., MM** selaku Rektor Universitas Putra Indonesia “YPTK” Padang.
5. Ibu **Dr. Rini Sovia, S.Kom., M.Kom** selaku Dekan Fakultas Ilmu Komputer Universitas Putra Indonesia “YPTK” Padang.
6. Ibu **Dr. Retno Devita, S.Kom., M.Kom** selaku Ketua Program Studi Sistem Komputer Universitas Putra Indonesia “YPTK” Padang
7. Bapak **Ir. Emil Naf’an, S.Kom., M.Kom., Ph.D** selaku Pembimbing I yang telah membimbing dan sangat banyak memberi masukan kepada penulis.
8. Bapak **Riska Robianto, S.Kom,M.Kom** selaku Pembimbing II yang telah membimbing dan sangat banyak memberi masukan kepada penulis.
9. Bapak dan Ibu Dosen serta Karyawan dan Karyawati Universitas Putra Indonesia “YPTK” Padang.

Akhir kata, penulis berharap semoga kebaikan dan amal yang mereka berikan kepada penulis mendapatkan balasan yang berlipat ganda hendaknya dari Allah S.W.T, Aamiiiin ya rabbal ‘Alamiin.

Padang, 4 Agustus 2025

Farhan Kennedy

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Keselamatan kerja merupakan aspek krusial yang harus diperhatikan dalam setiap proyek konstruksi. Lingkungan kerja proyek yang kompleks dan penuh risiko menuntut adanya sistem keamanan yang mampu melindungi para pekerja dari potensi kecelakaan, terutama akibat kelalaian dalam penggunaan alat pelindung diri (APD) seperti helm keselamatan. Kesadaran akan penggunaan Alat Pelindung Diri (APD) perlu ditanamkan pada setiap tenaga kerja karena keselamatan kerja adalah hal yang utama, namun demikian di lapangan adanya perasaan tidak nyaman (risih, panas, berat, terganggu) yang dirasakan karyawan saat menggunakan APD merupakan salah satu alasan mengapa pekerja tidak menggunakan APD (Nisa et al., 2022).

Seiring perkembangan teknologi informasi dan komunikasi, pemanfaatan *Internet of Things* (IoT) dalam bidang keselamatan kerja mulai banyak dikembangkan. Teknologi ini memungkinkan pemantauan kondisi pekerja secara *real-time* melalui integrasi sensor dan perangkat nirkabel. Mikrokontroler seperti ESP8266 atau ESP32 sangat ideal untuk digunakan dalam pengembangan sistem helm pintar karena memiliki kemampuan komunikasi jaringan dan mendukung berbagai jenis sensor (Santosa & Fajar, 2021).

Meskipun beberapa sistem keamanan berbasis sensor telah tersedia, banyak di antaranya masih bersifat pasif dan tidak memberikan umpan balik secara langsung kepada pekerja maupun pengawas proyek. Menurut penelitian oleh Wibowo (2021),

sistem keselamatan kerja berbasis IoT yang dilengkapi dengan fitur deteksi otomatis dan pengiriman notifikasi secara *real-time* terbukti lebih efektif dalam mengurangi risiko kecelakaan di lapangan. Sistem ini dapat mendeteksi suhu tubuh pekerja, posisi kepala, bahkan mendeteksi jika helm dilepas saat bekerja.

Selain itu, integrasi antara sensor suhu, sensor kejatuhan, dan GPS pada helm pintar memungkinkan pengawasan yang lebih detail terhadap kondisi pekerja. Penelitian oleh Rahmawati dan Hidayat (2020) menunjukkan bahwa sistem pemantauan kondisi fisik pekerja menggunakan perangkat *wearable* dapat meningkatkan efisiensi dan respons pengawasan di area konstruksi. Hal ini diperkuat oleh studi dari Arifin & Dewi (2023) yang menekankan bahwa penggunaan teknologi adaptif dalam sistem keamanan kerja memberikan dampak signifikan terhadap keselamatan dan produktivitas pekerja proyek.

Oleh karena itu, penelitian ini bertujuan untuk mengembangkan sistem keamanan pekerja proyek menggunakan helm pintar berbasis ESP yang dapat mendeteksi kondisi kritis seperti suhu tinggi, posisi tidak normal, atau pelepasan helm, serta mengirimkan notifikasi ke pusat pengawasan secara otomatis. Alat ini dilengkapi dengan sensor MAX30102 untuk mengetahui detak jantung dan kadar oksigen pada pekerja secara *real-time.* Untuk sensor MPU6050 sendiri berfungsi untuk mendeteksi gerakan abnormal atau jatuh pada pekerja proyek. Untuk sensor gas sendiri berfungsi sebagai pendeteksi adanya gas yang berbahaya bagi pekerja dalam pekerjaan sautu proyek. Untuk GPS berfungsi untuk mendeteksi Lokasi pekerja Ketika terjadi suatu kecelakan dalam suatu pekerjaan. Dan untuk *Buzzer, Vibration motor,* dan LED digunakan untuk memberikan peringatan langsung kepada pekerja saat terjadinya

kondisi berbahaya dalam pekerjaan. Pendekatan ini diharapkan dapat menjadikan pekerjaan terasa lebih aman kepada pekerja proyek. Dengan teknologi modern ini, pekerjaan dapat dilakukan secara efesien tanpa adanya rasa cemas saat terjadinya bahaya dalam suatu pekerjaan.

1.2 Perumusan Masalah

Berdasarkan Berdasarkan latar belakang masalah yang telah diuraikan di atas, dalam rangka melakukan penelitian ini, dibuat rumusan masalah yang dirancang secara sistematis dan relevan sebagai berikut :

1. Bagaimana merancang sistem keamanan berbasis helm pintar menggunakan mikrokontroler ESP32 yang terintegrasi dengan berbagai sensor dan aktuator untuk meningkatkan keselamatan pekerja proyek?
2. Bagaimana kerja sensor MPU6050 dalam mendeteksi gerakan abnormal atau jatuh pada pekerja proyek?
3. Bagaimana kerja sensor MAX30102 dalam memantau detak jantung dan kadar oksigen dalam darah pekerja secara *real-time*?
4. Bagaimana kerja modul GPS dalam menentukan dan mengirimkan lokasi pekerja ketika terjadi kondisi darurat?
5. Bagaimana kerja sensor gas dalam mendeteksi keberadaan gas berbahaya di lingkungan kerja proyek?
6. Bagaimana kerja *Buzzer, Vibration motor*, dan LED sebagai sistem peringatan langsung bagi pekerja saat terdeteksi kondisi bahaya?

7. Bagaimana kerja Telegram sebagai media notifikasi untuk mengirimkan informasi kondisi darurat kepada pihak pengawas proyek secara otomatis?
8. Bagaimana bahasa pemrograman C/C++ digunakan dalam merancang sistem pengendalian dan komunikasi data antar komponen pada helm pintar?

1.3 Batasan Masalah

Agar penelitian ini tetap fokus dan tidak menyimpang dari tujuan yang telah ditetapkan, maka pembahasan akan difokuskan pada masalah-masalah yang berkaitan dengan hal-hal berikut :

1. Menggunakan mikrokontroler ESP32 sebagai sistem pengontrol utama pada helm pintar.
2. Menggunakan sensor MPU6050 untuk mendeteksi gerakan tidak normal atau kejatuhan pekerja.
3. Menggunakan sensor MAX30102 untuk memantau detak jantung dan kadar oksigen dalam darah pekerja.
4. Menggunakan modul GPS untuk menentukan dan mengirimkan lokasi pekerja proyek.
5. Menggunakan sensor gas untuk mendeteksi adanya gas berbahaya di lingkungan kerja.
6. Menggunakan *Buzzer, Vibration motor*, dan LED sebagai sistem peringatan langsung kepada pekerja.
7. Menggunakan aplikasi Telegram sebagai media notifikasi kepada pengawas proyek.

8. Menggunakan bahasa pemrograman C/C++ sebagai sistem pengendalian dan komunikasi antar komponen pada helm pintar.

1.4 Hipotesa

Berdasarkan pada perumusan masalah yang telah dijelaskan di atas, maka dapat dirumuskan hipotesis penelitian yang relevan dan mendukung sebagai berikut :

1. Diharapkan perancangan sistem keamanan helm pintar berbasis ESP32 yang terintegrasi dengan berbagai sensor dan aktuator dapat berjalan dengan baik.
2. Diharapkan kerja sensor MPU6050 dalam mendeteksi gerakan tidak normal atau kejatuhan pekerja berjalan dengan baik.
3. Diharapkan kerja sensor MAX30102 dalam memantau detak jantung dan kadar oksigen dalam darah pekerja berjalan dengan baik.
4. Diharapkan kerja modul GPS dalam menentukan dan mengirimkan lokasi pekerja proyek berjalan dengan baik.
5. Diharapkan kerja sensor gas dalam mendeteksi gas berbahaya di lingkungan kerja berjalan dengan baik.
6. Diharapkan kerja *Buzzer, Vibration motor*, dan LED sebagai sistem peringatan berjalan dengan baik.
7. Diharapkan pengiriman notifikasi melalui aplikasi Telegram kepada pengawas proyek berjalan dengan baik.
8. Diharapkan kerja bahasa pemrograman C/C++ sebagai sistem pengendalian pada helm pintar berjalan dengan baik.

1.5 Tujuan Penelitian

Adapun tujuan utama yang ingin dicapai dalam proses perancangan dan pembuatan alat ini, agar penelitian yang dilakukan dapat memberikan kontribusi yang relevan, dapat dirumuskan secara rinci sebagai berikut:

1. Memahami konsep kerja mikrokontroler ESP32 serta integrasi dengan berbagai sensor dan aktuator pada sistem helm pintar. Menganalisa setiap permasalahan yang ada dan pemanfaatan alat–alat elektronika yang digunakan pada sistem yang dibuat.
2. Menganalisa kebutuhan dan permasalahan keselamatan kerja di lingkungan proyek yang dapat diatasi melalui penggunaan teknologi helm pintar.
3. Merancang sistem helm pintar berbasis ESP32 yang terintegrasi dengan sensor detak jantung, oksigen, gerakan, gas, dan GPS serta sistem notifikasi otomatis.
4. Membangun alat helm pintar yang dapat mendeteksi kondisi kritis pekerja dan memberikan peringatan secara langsung maupun melalui notifikasi ke pengawas.
5. Menguji kinerja sistem helm pintar dalam kondisi simulasi dan nyata untuk memastikan fungsionalitas semua komponen berjalan sesuai perencanaan.

1.6 Manfaat Penelitian

Adapun manfaat yang diharapkan dari penelitian ini dalam perancangan dan pengembangan sistem, agar dapat memberikan kontribusi yang relevan, dapat dijelaskan secara terperinci sebagai berikut:

A. Bagi Penulis

1. Sebagai sarana untuk menerapkan ilmu dan mengembangkan potensi diri dalam menambah pengetahuan terutama dibidang elektronika dan kontroller.

2. Sebagai syarat untuk memenuhi mata kuliah Skripsi.
3. Dapat mengetahui cara kerja komponen yang digunakan pada alat yang diproses oleh ESP32.

B. Bagi Program Studi

1. Mengaplikasikan ilmu pengetahuan di bidang komputer dalam pengontrolan alat menggunakan ESP32 dan menjadi salah satu contoh aplikasi pada mata kuliah yang dipelajari.
2. Dengan penelitian ini diharapkan dapat menambah motivasi bagi mahasiswa sistem komputer untuk berkarya lebih baik lagi dan menggali ilmu pengetahuan khususnya dalam bidang teknologi komputer.

C. Bagi Masyarakat

1. Dapat meningkatkan keamanan pekerja proyek.
2. Dapat mempercepat tindakan darurat.

BAB II

TINJAUAN PUSTAKA

2.1 Konsep Dasar Sistem

Rancang adalah kegiatan yang memiliki tujuan untuk mendesain sistem baru yang dapat menyelesaikan masalah-masalah yang dihadapi perusahaan yang diperoleh dari pemilihan alternatif sistem terbaik. Bangun adalah kegiatan menciptakan sistem baru maupun mengganti atau memperbaiki sistem yang telah ada baik secara keseluruhan maupun sebagian (Caritas Ziliwu, dkk, 2021).

Jadi dapat disimpulkan bahwa Rancang Bangun adalah proses pengembangan sistem untuk menciptakan sistem baru maupun mengganti atau memperbaiki sistem yang telah ada baik secara keseluruhan maupun hanya sebagian.

Sistem adalah kumpulan elemen yang saling terkait atau terpadu yang dimaksudkan untuk mencapai suatu tujuan. Sebagai gambaranya, jika dalam sebuah sistem terdapat elemen yang tidak memberikan manfaat dalam mencapai tujuan yang sama, maka elemen tersebut dapat dipastikan bukanlah bagian dari sistem. Sistem didefenisikan sebagai sekumpulan prosedur yang saling berkaitan dan saling terhubung untuk melakukan suatu tugas bersama-sama. Secara garis besar, sebuah sistem informasi terdiri atas tiga komponen utama. Ketiga komponen tersebut mencakup *software, hardware, dan brainware*. Ketiga komponen ini saling berkaitan satu sama lain. (Yuli Hartati, dkk, 2023).

2.1.1 Karakteristik Sistem

Sistem memiliki karakteristik tertentu sebagai ciri-ciri terbentuknya sebuah sistem, Menurut Dika Widiyanto (2023) Supaya Sistem itu dikatakan sistem yang baik memiliki karakteristik yaitu:

1. Komponen Sistem

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi yang artinya saling bekerja sama membentuk satu kesatuan. Setiap sistem selalu mengandung komponen atau subsistem.

2. Batasan Sistem

   Merupakan area yang membatasi antara sistem dengan sistem yang lain. Batas sistem memungkinkan sistem dilihat sebagai satu kesatuan. Batas suatu sistem menunjukkan lingkup dari sistem tersebut.

3. Lingkungan Luar Sistem

   Lingkungan luar dari sistem ialah apapun diluar batas yang berpengaruh pada operasi sistem. Lingkungan luar dapat bersifat menguntungkan ataupun merugikan.

4. Penghubung Sistem

   Penghubung merupakan media penghubung antara satu subsistem dengan subsitem lainnya. Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari suatu subsistem ke subsistem. Keluaran dari suatu sistem dapat menjadi masukan untuk subsistem lain dengan melalui penghubung.

5. Sasaran atau Tujuan

   Sistem mempunyai tujuan atau sasaran. Jika tidak mempunyai sasaran maka operasi sistem tidak ada gunanya. Sasaran sistem menentukan sekali, dapat dikatakan berhasil jika mencapai sasaran ataupun tujuannya.

2.1.2 Alur Pengembangan Sistem

Model prosedur pelaksanaan penelitian dan pengembangan dilakukan dengan *waterfall model*. *Waterfall model* memiliki model pengembangan yang berurutan dalam menyelesaikan suatu pengembangan perangkat lunak sehingga mempermudah pengerjaan (Hendri Susilo, dkk, 2021). Tahapan-tahapan dari *metode waterfall* yaitu :

1. *Requirement Analysis and Definition*

   Requirement Analysis and Definition adalah tahapan penetapan fitur, kendala dan tujuan sistem melalui konsultasi dengan pengguna sistem. Semua hal tersebut akan ditetapkan secara rinci dan berfungsi sebagai spesifikasi sistem. *Requirement analysis* atau analisis kebutuhan apa saja yang diperlukan dalam mengembangkan aplikasi perangkat lunak. Informasi dari analisis kebutuhan didapatkan dengan malakukan wawancara, diskusi atau survei langsung. Analisis yang dilakukan antara lain dengan membuat konsep aplikasi yang dapat digunakan dalam mendeskripsikan objek 3D.

2. *Sistem and Software Design*

   Proses Pada tahapan *Sistem and Software Design* ini akan dibentuk suatu arsitektur sistem berdasarkan persyaratan yang telah ditetapkan. Selain itu juga, dilakukan identifikasi dan penggambaran terhadap abstraksi dasar sistem perangkat lunak

beserta hubungan-hubungannya. Desain sistem dibuat berdasarkan *experience* (*user experience*) dan berdasarkan tampilan antarmuka (*user interface*).

3. *Implementation and Unit Testing*

   Dalam tahapan *Implementation and Unit Testing* ini, hasil dari desain dikembangkan dan ditranslasikan kedalam barisan program yang akan direalisasikan sebagai satu set program atau unit program. Setiap unit akan diuji apakah sudah memenuhi spesifikasinya. Sehingga dapat disimpulkan bahwa pada tahap ini terdapat dua jenis kegiatan, pemrograman dan pengujian sistem aplikasi.

4. *Integration and Sistem Testing*

   Proses Dalam tahap *Integration and Sistem Testing* ini, setiap unit program akan diintegrasikan satu sama lain dan diuji sebagai satu sistem yang utuh untuk memastikan sistem sudah memenuhi persyaratan yang ada. Setelah itu sistem akan dikirim ke pengguna sistem. Peneliti akan mencoba aplikasi di beberapa *smartphone* android untuk mengetahui *error* yang terjadi dan segera memperbaikinya sebelum diimplementasikan.

5. *Operation and Maintenance*

   Dalam tahap *Operation and Maintenance* ini, sistem mulai digunakan. Selain itu juga memperbaiki *error* yang tidak ditemukan pada tahap pembuatan. Dalam tahap ini juga dilakukan pengembangan sistem seperti penambahan fitur dan fungsi baru.

2.2 Sistem Kontrol

Kontrol menurut Kamus Besar Bahasa Indonesia adalah pengawasan, pemeriksaan, atau pengendalian. Sistem kontrol berarti kumpulan dari beberapa

komponen yang terhubung satu sama lainnya, sehingga membentuk suatu tujuan tertentu yaitu mengendalikan atau mengatur suatu sistem. Menurut prosesnya, sistem kontrol terdiri dari dua jenis, yaitu sistem *loop* terbuka dan *loop* tertutup.

2.2.1 Sistem Kendali Loop Terbuka

Sistem kendali *loop* terbuka merupakan suatu sistem pengaturan yang keluarannya tidak mempunyai pengaruh terhadap aksi kontrol. Diperkuat oleh pernyataan berikut, pada sistem pengaturan *loop* terbuka tidak terdapat umpan balik. Dengan kata lain, pada sistem pengaturan *loop* terbuka keluarannya tidak dapat digunakan sebagai pembanding umpan balik dengan acuan masukan (*set point*). Sebuah sistem kontrol *loop* terbuka menggunakan *controller* atau *actuator control* untuk mendapatkan respon yang diinginkan. Keuntungan dari sistem *loop* terbuka yaitu konstruksi sistem kendali yang sederhana dan harga yang relatif murah dan cocok digunakan pada sistem yang memiliki kekurangan yang sulit untuk diukur secara ekonomi tidak layak. Sedangkan kelemahan dari sistem *loop* terbuka yaitu adanya gangguan dan perubahan kalibrasi yang akan menimbulkan kesalahan, sehingga keluaran dapat berbeda dari yang diinginkan (Fina Ayu Lestari dan Bagus Dwi Cahyono, 2022). Berikut merupakan diagram dari sistem kendali *loop* terbuka:

Sumber: Fina Ayu Lestari dan Bagus Dwi Cahyono, Sistem Pengendali Mesin Solar Cells Automatic Tabber Stringer pada Penyolderan String di PT. Indonesia Solar Global, 2022, ISSN : 2828-4984

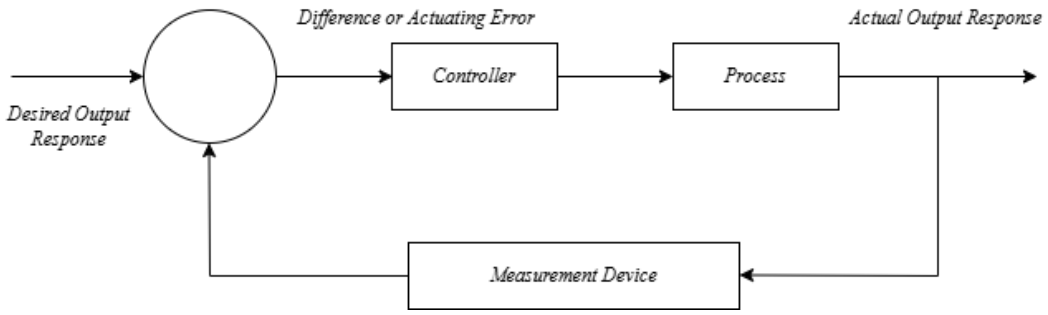
Gambar 2. 1 Contoh Sistem *Loop* Terbuka

2.2.2 Sistem Kendali Loop Tertutup

Loop tertutup merupakan sistem kontrol di mana sinyal *output* nya memberikan pengaruh pada sistem pengontrolan. *Loop* tertutup juga adalah sistem kontrol yang memiliki f*eedback*. Sinyal kesalahan penggerak, yang merupakan selisih antara sinyal masukan dan sinyal *feedback*. Diumpankan ke kontroler untuk memperkecil *error* yang membuat agar *output* sistem mendekati tujuan yang diinginkan.

Dengan istilah lain "*loop* tertutup" berarti menggunakan pengaruh *feedback* untuk mengurangi kesalahan sistem. Pada sistem kendali tertutup, memanfaatkan variabel yang sebanding dengan selisih respon yang terjadi terhadap respon yang diinginkan. Sistem umpan balik ini banyak dipergunakan pada sistem kemudi kapal dan pesawat terbang. Perangkat sehari-hari yang juga menerapkan sistem ini adalah penyetelan temperatur pada almari es, oven, valve dan pemanas air. Dalam sistem kendali tertutup dapat diketahui dengan adanya sensor sebagai umpan balik dalam proses sistem kendali ini yang memberikan pengaruh terhadap masukan.

Sensor merupakan bagian dari suatu piranti pengukuran maupun sistem pengendalian yang langsung berhubungan baik secara kontak langsung maupun tidak langsung dengan lingkungan di luar piranti atau sistem. Keuntungan dari sistem *loop* tertutup yaitu sistem *loop* tertutup dapat mengurangi kesalahan dengan otomatis menyesuaikan *input* sistem. Selain itu, sistem *loop* tertutup dapat meningkatkan stabilitas sistem yang tidak stabil, dapat juga menambah atau mengurangi sensitivitas sistem. Sedangkan kelemahan dari sistem *loop* tertutup yaitu untuk memberikan jumlah kontrol yang diperlukan, sistem *loop* tertutup harus lebih kompleks dengan memiliki satu atau lebih jalur umpan balik serta jika gain pengontrol terlalu sensitif terhadap perubahan perintah *input* atau sinyal, hal tersebut dapat menjadi tidak stabil dan mulai berosilasi ketika pengontrol mencoba untuk memperbaiki sendiri, dan akhirnya sesuatu akan pecah. Berikut merupakan diagram dari sistem kendali *loop* tertutup:



Sumber: Fina Ayu Lestari dan Bagus Dwi Cahyono, Sistem Pengendali Mesin Solar Cells Automatic Tabber Stringer pada Penyolderan String di PT. Indonesia Solar Global, 2022, ISSN : 2828-4984

Gambar 2. 2 Contoh Sistem *Loop* Terbuka

2.3 Alat Bantu Perancangan Sistem

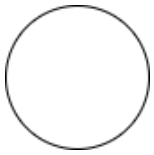
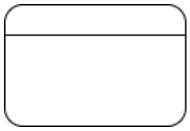
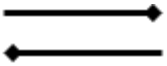
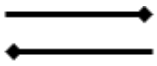
Proses penganalisaan terhadap suatu sistem dilakukan dengan aturan dasar yang mendefenisikan secara menyeluruh sistem yang akan dirancang. Hal ini mengandung arti bahwa harus ada gambran yang jelas mengenai ruang lingkup tentang sistem yang dibahas. Media yang digunakan untuk menggambarkan sistem tersebut adalah *Context Diagram*, *Data Flow Diagram* (DFD), dan *Flowchart*.

2.3.1 Context Diagram

Context Diagram adalah diagram yang terdiri dari suatu proses dan menggambarkan ruang lingkup suatu sistem. Diagram konteks merupakan level tertinggi dari dfd yang menggambarkan seluruh *input* ke sistem atau *output* dari sistem. Ia akan memberi gambaran tentang keseluruan sistem. Sistem dibatasi oleh *boundary* (dapat digambarkan dengan garis putus). Dalam diagram konteks hanya ada satu proses. Tidak boleh ada *store* dalam *Context Diagram*. Gambar yang di gunakan dapat dilihat di tabel 2.1 sebagai berikut :

Tabel 2. 1 Simbol *Context Diagram*

Gambar		**Keterangan**
Notasi Yourdan/Demiarco	**Notasi Gane & Sarson**	
		simbol *external entity* / *terminal* menggambarkan asal atau tujuan data di luar sistem

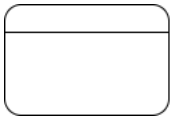
		Sistem lingkaran menggambarkan entitas atau proses dimana aliran data masuk ditansformasikan ke aliran data keluar
		Simbol aliran data meggambarkan aliran data
		simbol file menggambarakan tempat data disimpan

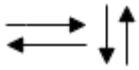
Sumber: Nofri Wandi Al-hafiz & Erlinda, Perancangan Sistem Penyiraman Otomatis Menggunakan Arduino, 2020, ISSN: 2655-7592

2.3.2 Data Flow Diagram

Data Flow Diagram adalah komponen yang digunakan dalam pembuatan suatu konteks diagram dan DFD dari rancangan sistem yang dibahas dalam penulisan tugas akhir. Simbol dari *flow diagram* dapat dilihat pada tabel 2.2.

Tabel 2. 2 Simbol Pada *Data Flow Diagram*

Simbol	Nama	Keterangan
	*Eksternal entity* (kesatuan luar)	Digunakan Simbol ini merupakan sumber atau tujuan data suatu bagian /orang yang berada di luar sistem tapi berhubungan dengan sistem tersebut, baik itu memasukkan data maupun mengambil data dari sistem.
	Proses	Simbol ini digunakan untuk melakukan proses pengolahan data yang menunjukkan suatu kegiatan yang mengubah aliran data masuk (*input*) menjadi aliran data keluar (*output*).

	Penyimpanan data	Simbol ini berfungsi sebagai tempat penyimpanan dokumen/file yang dibutuhkan dalam suatu sistem informasi.
	Aliran data	Simbol ini menunjukkan arus dalam proses, dimana simbol aliran data ini mempunyai nama sendiri.

Sumber: Nofri Wandi Al-hafiz & Erlinda, Perancangan Sistem Penyiraman Otomatis Menggunakan Arduino, 2020, ISSN: 2655-7592

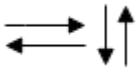
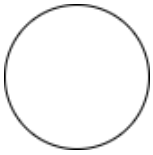
2.3.3 Flowchart

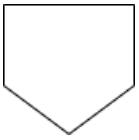
Flowchart atau dalam bahasa Indonesia disebut dengan Diagram Alir ini dipergunakan dalam industri manufakturing untuk menggambarkan proses-proses operasionalnya sehingga mudah dipahami dan mudah dilihat berdasarkan urutan langkah dari suatu proses ke proses lainnya. Berikut gambar simbol-simbol *Flowchart.*

a. *Flow Direction Symbols*

Yaitu, simbol yang dipakai untuk menghubungkan antara simbol yang satu dengan simbol lainnya atau disebut juga *connecting line*. *Flow Direction Symbols* dapat dilihat pada Tabel 2.3.

Tabel 2. 3 *Flowchart*

Simbol	Nama	Keterangan
	Arus / *Flow*	Penghubung antara prosedur / proses
	*Connector*	Simbol keluar / masuk prosedur atau proses dalam lembar / halaman yang sama

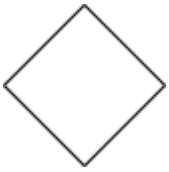
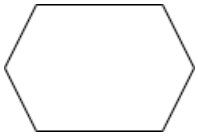
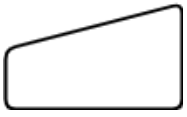
	*Off-line Connector*	Simbol keluar / masuk prosedur atau proses dalam lembar / halaman yang lain

Sumber: Nofri Wandi Al-hafiz & Erlinda, Perancangan Sistem Penyiraman Otomatis Menggunakan Arduino, 2020, ISSN: 2655-7592

b. *Processing Symbols*

Merupakan simbol yang menunjukkan jenis operasi pengolahan dalam suatu prosedur. *Processing Symbols* dapat dilihat pada tabel 2.4.

Tabel 2. 4 *Flowchart*

Simbol	Nama	Keterangan
	*Process*	Simbol yang menunjukkan pengolahan yang dilakukan Komputer
	*Decision*	Simbol untuk kondisi yang akan menghasilkan beberapa kemungkinan jawaban / aksi
	*Predefined Process*	Simbol untuk mempersiapkan penyimpanan yang akan digunakan sebagai tempat pengolahan di dalam *storage*
	*Terminal*	Simbol untuk permulaan atau akhir dari suatu program
	*Manual Input*	Simbol untuk pemasukan data secara manual *on-line keyboard*

Sumber: Nofri Wandi Al-hafiz & Erlinda, Perancangan Sistem Penyiraman Otomatis Menggunakan Arduino, 2020, ISSN: 2655-7592

c. *Input Output Symbols*

Simbol yang dipakai untuk menyatakan jenis peralatan yang digunakan sebagai media *input* atau *output*. *Input Output Symbols* dapat dilihat pada tabel 2.5.

Tabel 2. 5 *Flowchart*

	*Input-Output*	Simbol yang menyatakan proses *input* dan *output* tanpa tergantung dengan jenis peralatannya
	*Document*	Simbol yang menyatakan *input* berasal dari dokumen dalam bentuk kertas atau *output* di cetak dikertas
	*Disk and On-line Storage*	Simbol untuk menyatakan *input* berasal dari *disk* atau *output* di simpan ke *disk*

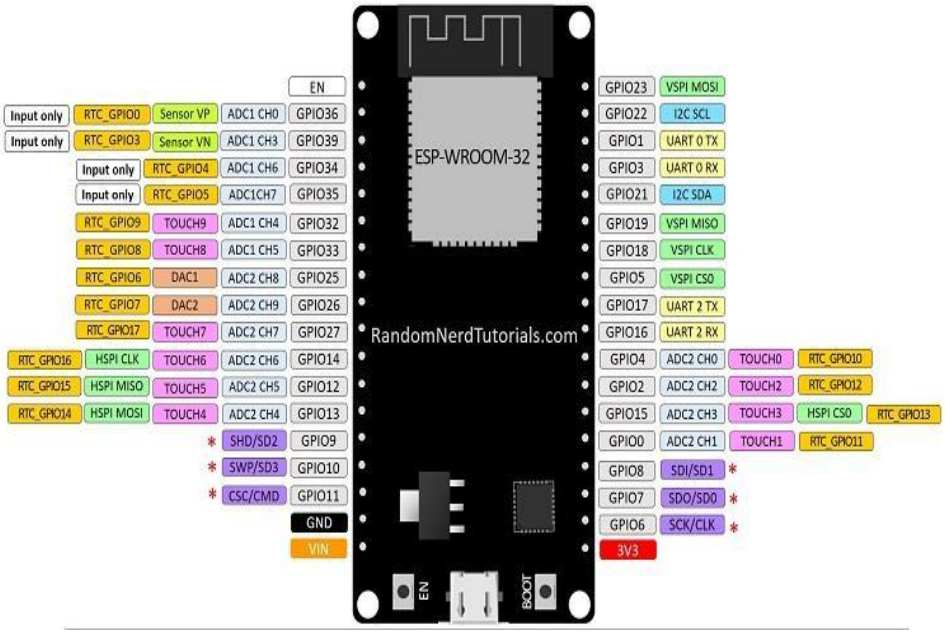
Sumber: Nofri Wandi Al-hafiz & Erlinda, Perancangan Sistem Penyiraman Otomatis Menggunakan Arduino, 2020, ISSN: 2655-7592

2.4 Komponen Utama

Komponen utama yang digunakan pada perancangan sistem ini akan dijelaskan sebagai berikut :

2.4.1 ESP32

ESP32 adalah sebuah *development board* seperti Arduino dan dikembangkan khusus untuk aplikasi dan solusi *Internet of Things*. Jenis ini sangat cocok dignakan untuk pembelajaran dan hobi *project Internet Of Things*. ESP32 ini dapat diprogram menggunakan Arduino IDE. ESP32 menggunakan *core* ESP32, sebuah *core* mikrokontroler berbasis komunikasi WiFI.



Sumber: Demi Adidrana, dkk, Perancangan Sistem Smart Lamp berbasis Internet of Things Menggunakan Ubidots, 2020, ISSN : 2686-1089

Gambar 2. 3 ESP32

Spesifikasi yang ada pada papan ESP32 terdapat pada tabel 2.6 berikut.

Tabel 2. 6 Spesifikasi ESP32

No	Specification	Description
1	Tegangan	3.3 *Volt*
2	CPU	Xtens a *dual core* LX6 - 160M Hz
3	Arsitektur	32 bit
4	*Flash Memory*	16MB
5	SRAM	512kb
6	GPIO Pin (ADC/DAC)	36 (18/2)
7	*Bluetooth*	Ada

8	WiFi	Ada
9	SPI/I2C/UAR T	4/2/2

Sumber: Muliadi, dkk, Pengembangan tempat sampah pintar menggunakan ESP32, 2020, ISSN : 2721-9100

2.4.2 MPU-6050

Sensor MPU-6050 merupakan sebuah sensor yang mampu mendeteksi sebuah kemiringan yang data didapatkan melalui sensor *gyroscope* dan *accelerometer*. *Range* kemiringan yang dapat dibaca oleh sensor ini *plus minus* 250, 500, 100, dan 2000. Sensor ini memiliki 16 bit analog ADC yang berfungsi untuk merubah nilai analog menjadi nilai digital. MPU 6050 memiliki kinerja yang optimal dikarenakan memiliki Digital *Motion Processor*, yang membantu mengintegrasikan magnetometer dengan sensor lainya melalui I2C. Modul ini menggunakan tegangan kerja sebesar 3.3 *Volt*. Karena MPU 6050 memiliki regulator, sehingga dapat dihubungkan secara langsung menggunakan tegangan sebesar 5 *Volt* (Fikri Akmal Trisetio, dkk, 2021). Spesifikasi dari MPU 6050 adalah sebagai berikut:

1. Berbasis Chip MPU-6050
2. *Supply* tegangan berkisar 3-5V
3. *Gyroscope range* + 250 500 1000 2000 ° / s
4. *Acceleration range*: ± 2 ± 4 ± 8 ± 16 g
5. *Communication standard* I2C
6. *Chip built-in* 16 bit AD *converter*, 16 bits data *output*
7. Jarak antar pin *header* 2.54 mm

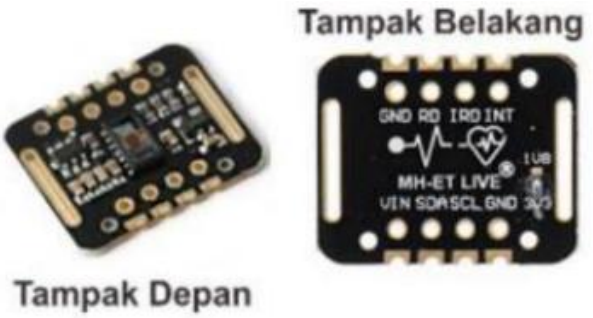
8. Dimensi modul 20.3mm x 15.6mm

Sumber: Fikri Akmal Trisetio, dkk, Implementasi Solenoid dan Sensor Getar Pada Sistem Keamanan Sepeda Menggunakan Modul Bluetooth dan GSM Berbasis Mikrokontroler, 2021, ISSN: 2355-9365

Gambar 2. 4 MPU-6050

2.4.3 MAX30102

Modul Sensor MAX30102 adalah modul darig *Maxim Integrated* yang digunakan untuk mengukur konsentrasi oksigen dalam darah dan pengukuran detak jantung. Keunggulan dari sensor MAX30102 adalah memiliki *noise* yang rendah sehingga akan mudah untuk dikalibrasi. Sensor ini juga banyak digunakan dalam sistem pemantauan terutama dalam bidang kebugaran. Memantau detak jantung dan suhu tubuh pada saat berolahraga sangat penting untuk mengetahui kondisi kesehatan (Maulida Uswatun Jannah, dkk, 2024).



Sumber: Maulida Uswatun Jannah, dkk, Sistem Deteksi Detak Jantung Berbasis Sensor MAX30102, ARDUINO UNO, Dan Oled Display Untuk Pemantauan Detak Jantung Secara Real-Time, 2024, eISSN: 2830-7062

Gambar 2. 5 MAX30102

2.4.4 GPS UBLOX NEO-7M

GPS UBLOX NEO-7M merupakan modul yang dapat membantu untuk mengetahui lokasi suatau tempat atau koordinat modul GPS itu berada. Modul tersebut mampu membuat berbagai macam alat yang memerlukan lokasi yang membutuhkan koordninat (Muhammad Kilau Natsyaqov, dkk, 2020).

Sumber: Muhammad Kilau Natsyaqov, dkk, Pembangunan Perangkat Pelacak Truk Pengankut Limbah Tinja Dengan Modul Gps, 2022, ISSN : 2442-5826

Gambar 2. 6 GPS UBLOX NEO 7M

2.4.5 Sensor MQ-135

Sensor MQ-135 adalah jenis sensor kimia yang sensitif terhadap senyawa NH3, NOx, alkohol, benzol, asap (CO), CO2, dan lain-lain. Sensor ini bekerja dengan cara menerima perubahan nilai resistansi (analog) bila terkena gas. Sensor ini memiliki daya tahan yang baik untuk penggunaan penanda bahaya polusi karena praktis dan tidak memakan daya yang besar. Penyesuaian sensitifitas sensor ditentukan oleh nilai resistansi dari MQ-135 yang berbeda-beda untuk berbagai konsentrasi gas-gas. Jadi, Ketika menggunakan komponen ini, penyesuaian sensitifitas sangat diperlukan. Selain itu, kalibrasi pendeteksian konsentrasi NH3 sebesar 100 ppm atau alkohol sebesar 50 ppm di udara (Arida Amalia Rosa, dkk, 2020).

Sumber: Arida Amalia Rosa, dkk, Sistem Pendeteksi Pencemar Udara Portabel Menggunakan Sensor MQ-7 dan MQ-135, 2020, ISSN 2355-3286

Gambar 2. 7 Sensor MQ-135

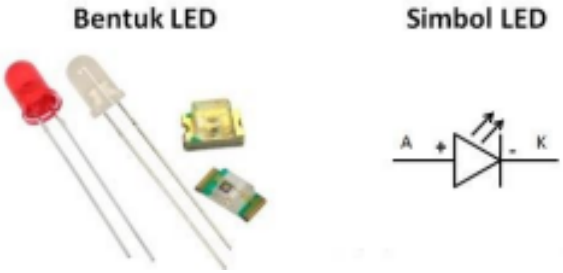
Tabel 2. 7 Spesifikasi standar kerja Sensor MQ-135

Parameter	Kondisi Teknis	Keterangan
Circuit Voltage	5V ± 0,1	AC atau DC
Heating Voltage	5V ± 0,1	AC atau DC

Load Resistance	Bisa menyesuakan	
Heater Resistance	33 Ω ± 5%	Suhu ruangan
Heating Consumption	<500 mW	
Jangkauan Pengukuran	10-300 ppm ammonia 10-1000 ppm benzol 10-300 ppm alkohol	

2.4.6 LED

Light-Emitting Diode (LED) adalah suatu *device* semikonduktor yang memancarkan cahaya monokromatik yang tidak koheren ketika diberi tegangan maju. Karakteristik chip LED pada umumnya adalah sama dengan karakteristik *diode* yang hanya memerlukan tegangan tertentu untuk dapat beroperasi. Namun bila diberikan tegangan yang terlalu besar, LED bisa rusak (terbakar) walaupun tegangan yang diberikan adalah tegangan maju.



Sumber: Joni Welman Simatupang, dkk, Lampu LED sebagai Pilihan yang Lebih Efisien untuk Lampu Utama Sepeda Motor, 2021, ISSN: 2502-8464

Gambar 2. 8 LED

Seperti yang telah dijelaskan sebelumnya bahwa LED merupakan keluarga dari dioda (*family of diodes*) yang terbuat dari material semikonduktor. Cara kerjanya pun hampir sama dengan dioda yang memiliki dua kutub yaitu kutub Positif (P) dan Kutub

Negatif (N). LED hanya akan memancarkan cahaya apabila dialiri tegangan maju (*bias forward*) dari Anoda menuju ke Katoda. Contoh bentuk dan simbol lampu LED ditunjukkan pada Gambar 2.8.

LED terdiri dari sebuah chip semikonduktor yang di doping sehingga menciptakan junction P dan N (PN *junction*). Hal ini bisa terjadi karena adanya pemberian doping kepada masing-masing elemental *semiconductor*, misalnya *Silicon* (Si). Yang dimaksud dengan proses doping dalam semikonduktor adalah proses untuk menambahkan ketidakmurnian (*impurity*) pada semikonduktor yang murni sehingga menghasilkan karakteristik kelistrikan yang diinginkan. Ketika LED dialiri tegangan maju atau *bias forward* yaitu dari Anoda (P) menuju ke Katoda (K), Kelebihan Elektron pada N-*Type* material akan berpindah ke wilayah yang kelebihan *Hole* (lubang) yaitu wilayah yang bermuatan positif (P-*Type* material). Saat Elektron berjumpa dengan *Hole* akan melepaskan photon dan memancarkan cahaya monokromatik (satu warna). LED untuk fisiknya ada 2 yaitu LED *type radial* dan LED type SMD (*Surface Mounted Device*).

2.4.7 Buzzer

Buzzer merupakan komponen elektronika yang dapat menghasilkan getaran suara berupa gelombang bunyi. *Buzzer* akan menghasilkan getaran suara ketika diberikan sejumlah tegangan listrik dengan taraf tertentu sesuai dengan spesifikasi bentuk dan ukuran *Buzzer* elektronika itu sendiri. Setiap *Buzzer* memerlukan *input* berupa tegangan listrik yang kemudian diubah menjadi getaran suara atau gelombang bunyi yang memiliki frekuensi dengan kisaran antara 1-5 KHz. *Buzzer* memiliki 2 buah

kaki yaitu positif dan negatif. Secara sederhana, kita bisa menggunakannya dengan memberikan tegangan positif dan negatif 3V-12V (Willy Oktodinata dan Yunianto Purnomo, 2024).

Sumber: Willy Oktodinata dan Yulianto Purnomo, Perangkat Jam Portabel dengan Fungsi Pembaca Suhu dan Pelacakan Suara Melalui Buzzer Menggunakan Modul Nrf Berbasis Arduino, 2024, E-ISSN: 2963-2293

Gambar 2. 9 *Buzzer*

2.4.8 Vibration motor

Vibration motor adalah motor DC tanpa inti dan ukuran motor ini kompak. Tujuan utama motor ini adalah untuk memperingatkan pengguna agar dapat merespon secara fisik yang dapat dirasakan pada indra (Arham Arifin, dkk, 2024). Fitur utama motor ini memiliki sifat magnetik ringan dan ukuran motor kecil, ditunjukkan pada gambar di bawah ini :

Sumber: Arham Arifin, dkk, Rancang Bangun Jam Weker Portable Untuk Tunarungu Tanpa Gelombang Elektromagnetik Berbasis Arduino Nano, 2024, e-ISSN : 2797-3298

Gambar 2. 10 *Vibration motor*

2.4.9 Telegram

Telegram adalah layanan pesan instan berbasis *cloud* yang gratis dan nirlaba. Telegram ini dapat digunakan oleh pengguna dengan sistem operasi berbeda seperti iOS, Windows *Phone*, Android, Ubuntu *Touch*, macOS, Linux, Windows. Aplikasi ini dapat mengirim pesan teks, audio, stiker, foto, dokumen dan berkasi lainny (Tri Sulistyorinia, dkk, 2024).

Sumber: Tri Sulistyorinia, dkk, Pemanfaatan Telegram Untuk Pendeteksi Kebocoran Gas Berbasis Nodemcu, 2024, E-ISSN: 2828-6901

Gambar 2. 11 Telegram

2.5 Komponen Pendukung

Komponen pendukung digunakan sebagai pelengkap dari sebuah sistem agar dapat berjalan dengan maksimal. Komponen–komponen pendukung tersebut akan dijabarkan sebagai berikut :

2.5.1 Kabel Jumper

Kabel *jumpe*r adalah suatu istilah kabel berdiameter kecil yang di dalam dunia elektronika digunakan untuk menghubungkan dua titik atau lebih dan dapat juga untuk menghubungkan 2 komponen elektronika.

Ada beberapa jenis kabel *jumper* yang dibedakan berdasarkan konektor kabelnya, yaitu :

1. *Male–Male*

   Kabel *jumper* ini digunakan untuk koneksi *male to male* pada kedua ujung kabelnya.

Sumber: Feri Irawan, Perancangan sistem alat sterilisasi pintu keluar ruang isolasi berbasis arduino di RS Hastien, 2022, ISSN: 3021-8209

Gambar 2. 12 Kabel *Male – Male*

2. *Male–Female*

   Kabel *jumper* ini digunakan untuk koneksi *male to female* dengan salah satu ujung kabel di koneksi *male* dan satu ujungnya lagi dengan koneksi *female*.

Sumber: Feri Irawan, Perancangan sistem alat sterilisasi pintu keluar ruang isolasi berbasis arduino di RS Hastien, 2022, ISSN: 3021-8209

Gambar 2. 13 Kabel *Male – Female*

3. *Female–Female*

   Kabel *jumper* jenis ini digunakan untuk koneksi *female to female* pada kedua ujung kabel nya.

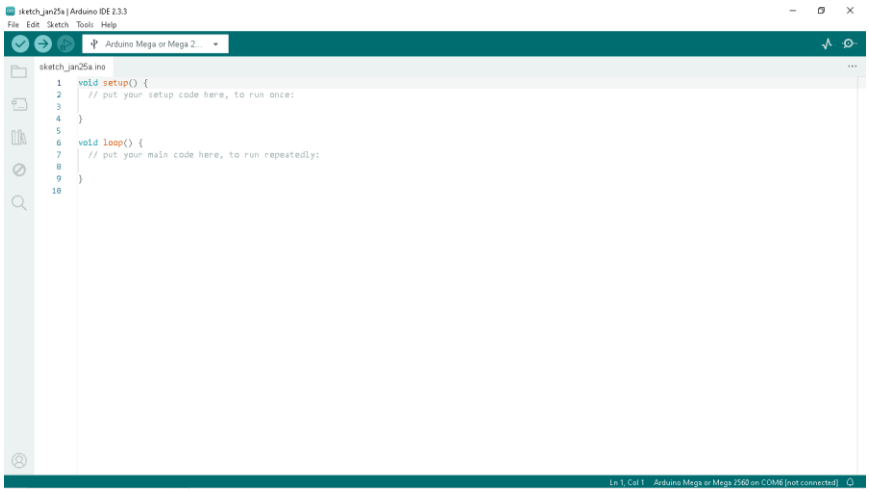
Sumber: Feri Irawan, Perancangan sistem alat sterilisasi pintu keluar ruang isolasi berbasis arduino di RS Hastien, 2022, ISSN: 3021-8209

Gambar 2. 14 Kabel *Female – Female*

2.6 IDE Arduino

Perancangan sistem pada *software* Arduino sangat lah penting sebab dari sinilah program dibuat dan di *upload* mengunakan *software* Arduino, hal ini bertujuan untuk menyisipkan kode program kedalam Arduino (Anantama, dkk, 2020). Kata IDE merupakan kependekan dari *Integrated Development Environment*, yaitu lingkungan terintegrasi yang digunakan untuk melakukan pengembangan program.

Arduino IDE memudahkan pengguna Arduino untuk dapat membuat, memperbarui, menghapus, dan mengirimkan kode pemrograman sketsa dari personal computer ke papan Arduino. Bahasa pemrograman yang dipakai untuk menyusun sketsa adalah bahasa C atau C++. Bentuk dari Arduino IDE versi 2.0.0 terdapat pada gambar 2.15 berikut.



Sumber: Dani Sasmoko, Arduino dan Sensor pada Project Arduino DIY, 2019, ISBN : 978-623-6141-37-3

Gambar 2. 15 Arduino IDE

2.7 Pemrograman Arduino

Pembuatan proyek berbasis Arduino tidak bisa lepas dari pemrograman. Arduino memakai bahasa C sebagai bahasa dasarnya. Bahasa C adalah bahasa yang sangat terkenal karena perannya yang besar dalam perkembangan teknologi. Pada bab ini, dasar pemrograman Arduino akan dijelaskan sebagai berikut:

2.7.1 Struktur

Pada Arduino terdapat dua fungsi yang wajib ada, yaitu fungsi *setup* dan *loop*. Fungsi *setup* akan dijalankan pertama kali setelah alat dihidupkan selama satu kali. Fungsi *loop* akan terjus berjalan sampai alat dimatikan.

2.7.2 Tipe Data

Data pada umumnya dibedakan menjadi dua golongan, yaitu variabel dan konstanta. Variabel berguna untuk menyimpan suatu nilai yang dapat diubah selama pemrograman dieksekusi. Konstanta dinyatakan sebagai data tetap sehingga nilai yang ada di dalamnya tidak dapat diubah selama pemrograman dieksekusi. Ada beberapa tipe data menurut yang dijelaskan pada tabel 2.8 berikut.

Tabel 2. 8 Tipe Data

No	Tipe Data	Keterangan	Kebutuhan Memori
1	*Boolean*	Berguna untuk menyimpan data yang berkemungkinan dua, yaitu *true* atau *false* saja.	1 *byte*
2	*Char*	Berguna untuk menyatakan sebuah karakter	1 *byte*

3	*Byte*	Berguna untuk menyimpan data yang berkisar antara 0 sampai dengan 255	1 *byte*
4	*Int*	Berguna untuk menampung bilangan bulat yang berkisar antara -32768 sampai dengan 32767.	2 *byte*
5	*Unsigned Int*	Berguna untuk menampung bilangan bulat yang berkisar antara 0 sampai dengan 65535.	2 *byte*
6	*Word*	Identik dengan *unsigned int*.	2 *byte*
7	*Long*	Berguna untuk menampung bilangan bulat yang berkisar antara -2.147.483.648 sampai dengan 2.147.483.647.	4 *byte*
8	*Unsigned Long*	Berguna untuk menampung bilangan bulat yang berkisar antara 0 sampai dengan 4.294.967.295 (232-1).	4 *byte*
9	*Float atau Double*	Berguna untuk menyimpan bilangan real. Angka yang bisa disimpan dari -3.4028235E+38 sampai dengan 3.4028235E+38. Tingkat presisi hingga 6-7 digit.	4 *byte*

Sumber: Abdul Kadir, Arduino Mega Panduan untuk Mempelajari Pembuatan Berbagai Proyek Elektronika, 2020, ISBN: 9789792966435

2.7.3 Operator

Operator merupakan simbol yang biasa dilibatkan dalam program untuk melakukan suatu operasi. Terdapat dua jenis operator pada Arduino, yaitu operator matematik yang digunakan untuk operasi matematik dan operator pembanding yang bisanya digunakan untuk membandingkan dua nilai.

2.7.3.1 Operator Matematik

Operator matematik digunakan untuk memanipulasi angka seperti penambahan, pengurangan, perkalian dan pembagian. Operator matematik yang terdapat pada Arduino terdapat pada tabel 2.9 berikut.

Tabel 2. 9 Operator Matematik

No	Operator	Keterangan
1	*	Perkalian
2	/	Pembagian
3	%	Modulo, Sisa pembagian
4	+	Penjumlahan
5	-	pengurangan

Sumber: Abdul Kadir, Arduino Mega Panduan untuk Mempelajari Pembuatan Berbagai Proyek Elektronika, 2020, ISBN: 9789792966435

2.7.3.2 Operator Pembanding

Operator pembanding digunakan untuk operasi logika perbandingan antara dua nilai atau antara dua variabel. Operator pembanding yang terdapat pada Arduino terdapat pada tabel 2.10 berikut.

Tabel 2. 10 Operator Pembanding

No	Operator	Keterangan
1	==	Sama dengan (*equal to*)
2	!=	Tidak sama dengan (*not equal to*)
3	<	Lebih besar (*greater than*)

4	>	Lebih kecil atau sama dengan (*less than or equal*)
5	<=	pengurangan
6	>=	Lebih besar atau sama dengan (*greater than or equal*)

Sumber: Abdul Kadir, Arduino Mega Panduan untuk Mempelajari Pembuatan Berbagai Proyek Elektronika, 2020, ISBN: 9789792966435

2.7.3.3 Operator Boolean

Operator boolean adalah pembanding yang digunakan untuk kondisi di dalam *statement* percabangan *IF*. Operator boolean yang terdapat pada Arduino terdapat pada tabel 2.9 berikut. Boolean akan mengeluarkan nilai *true* (1) atau *false* (0).

Tabel 2. 11 Operator Boolean

No	Operator	Keterangan
1	&&	Dan
2	||	Atau
3	!	Tidak

Sumber: Abdul Kadir, Arduino Mega Panduan untuk Mempelajari Pembuatan Berbagai Proyek Elektronika, 2020, ISBN: 9789792966435

2.7.4 Kondisi

Struktur kondisi biasa digunakan untuk menyeleksi kondisi yang akan dieksekusi selanjutnya oleh program berdasarkan kondisi yang terpenuhi. Struktur ini umumnya dipakai untuk memutuskan satu pilihan pada program dari banyaknya pilihan kondisi. Beberapa struktur kondisi dijelaskan sebagai berikut:

2.7.4.1 Kondisi If

Pada struktur kondisi ini akan dilakukan pemeriksaan kondisi. Jika kondisi pada struktur ini tidak terpenuhi (*false*), maka program akan melewati kode yang ada dalam struktur. Jika kondisi pada struktur ini terpenuhi (*true*), maka program akan mengeksekusi kode yang ada dalam struktur. Berikut ini struktur kondisi *if*:

```
if(kondisi) {
//pernyataan terpenuhi
}
```

2.7.4.2 Kondisi If-Else

Strukur ini adalah lanjutan dari struktur kondisi *if*. Perbedaannya terletak pada kondisi yang tidak terpenuhi (*false*) mempunyai kode di dalamnya dan program akan mengeksekusi kode tersebut tanpa melewati kode yang ada dalam struktur kondisi ini. Berikut ini struktur kondisi *if-else*:

```
if(kondisi) {
//pernyataan terpenuhi
} else {
//pernyataan tidak terpenuhi
}
```

2.7.4.3 Kondisi Switch Case

Struktur ini mengizinkan untuk memilih salah satu dari beberapa pilihan. Perbedaan mendasar terletak pada eksekusi *switch case* yang mencari kondisi yang

cocok sementara eksekusi *if-else* memeriksa setiap kondisi yang ada. Berikut ini struktur kondisi *switch case*:

Switch(nilai yang diperiksa) {

case kondisi1: //pernyataan1; *break*;

case kondisi2: //pernyataan2; *break*;

}

2.7.5 Perulangan

Perulangan digunakan untuk mengulang satu pernyataan atau beberapa pernyataan yang terdapat di dalam blok perulangan selama kondisi yang diberikan masih terpenuhi. Beberapa struktur perulangan dijelaskan sebagai berikut:

2.7.5.1 Perulangan For

Perulangan *for* adalah perulangan satu atau sekumpulan pernyataan selama kondisi pengujian pada nilai memberikan nilai *true* dan nilai inisialisasi akan terus bertambah atau berkurang sampai pengujian menghasilkan nilai *false*. Perulangan *for* mempunyai bentuk seperti berikut:

For(inisialisasi nilai; pengujian; penambahan/pengurangan) {

//pernyataan

}

2.7.5.2 Perulangan While

Perulangan *while* adalah perulangan yang akan menjalankan pernyataan selama kondisi menghasilkan nilai *false*. Berbeda dengan *for* yang hanya bisa melakukan

pengujian nilai bilangan *real*, *while* dapat dipakai untuk pengujian banyak pernyataan. Perulangan *while* mempunyai bentuk seperti berikut:

```
while(kondisi) {
//pernyataan
}
```

2.7.5.3 Perulangan Do-While

Perulangan ini adalah lanjutan dari perulangan *while*. Perbedaannya terletak pada pengujian *while* dilakukan di awal, sedangkan *do-while* dilakukan di akhir. Perulangan *do-while* mempunyai bentuk seperti berikut:

```
do {
//pernyataan
} while(kondisi);
```

2.8 Sistem Keamanan Pekerja Proyek

Keselamatan dan keamanan kerja merupakan aspek krusial dalam lingkungan proyek konstruksi dan industri. Pekerja proyek sering dihadapkan pada berbagai risiko seperti jatuh dari ketinggian, tertimpa material, tersengat listrik, atau terpapar zat berbahaya. Oleh karena itu, penerapan sistem keamanan pekerja proyek sangat penting untuk mencegah kecelakaan dan mengurangi dampak cedera.

2.8.1 Tujuan Sistem Keamanan

Tujuan utama dari sistem keamanan pekerja proyek adalah untuk melindungi keselamatan dan kesehatan para pekerja selama menjalankan aktivitas di lokasi proyek.

Dengan sistem keamanan yang baik, risiko kecelakaan kerja seperti jatuh dari ketinggian, tertimpa material, atau terpapar bahan berbahaya dapat diminimalkan. Selain itu, sistem ini juga bertujuan untuk memastikan setiap kegiatan di lapangan berjalan sesuai dengan standar keselamatan kerja yang berlaku, baik nasional maupun internasional. Penerapan sistem keamanan yang disiplin dapat menciptakan lingkungan kerja yang lebih aman, produktif, dan profesional. Tidak hanya itu, sistem keamanan juga membantu membangun budaya kerja yang peduli terhadap keselamatan, baik di tingkat pekerja lapangan maupun manajemen proyek secara keseluruhan.

2.8.2 Komponen Sistem Keamanan Pekerja Proyek

Untuk menciptakan lingkungan kerja yang aman dan meminimalkan risiko kecelakaan, sistem keamanan pekerja proyek harus disusun secara menyeluruh melalui berbagai komponen pendukung. Komponen-komponen ini saling melengkapi dan berperan penting dalam menjaga keselamatan pekerja di lapangan, mulai dari perlindungan fisik, pengawasan aktivitas, hingga pemanfaatan teknologi. Berikut ini adalah beberapa komponen utama dalam sistem keamanan pekerja proyek:

a. Alat Pelindung Diri (APD)

Alat Pelindung Diri merupakan komponen paling dasar dan wajib dalam sistem keamanan proyek. Perlengkapan seperti helm proyek, rompi reflektif, sepatu *safety*, sarung tangan, dan masker berfungsi melindungi pekerja dari potensi cedera akibat kecelakaan di lokasi kerja.

b. Sistem Monitoring dan Pengawasan

Pengawasan yang efektif sangat diperlukan untuk memastikan semua pekerja mematuhi prosedur keselamatan. Sistem ini dapat berupa pengawasan langsung oleh petugas keselamatan (*safety officer*) maupun melalui teknologi seperti kamera CCTV dan sensor otomatis untuk mendeteksi kondisi berbahaya.

c. Pelatihan dan Sosialisasi K3

Pekerja proyek wajib mengikuti pelatihan keselamatan dan kesehatan kerja (K3) sebelum mulai bekerja. Pelatihan ini mencakup penggunaan APD, cara kerja yang aman, penanganan darurat, dan pemahaman terhadap bahaya di lingkungan kerja.

d. Prosedur Standar Operasional (SOP)

SOP merupakan panduan tertulis yang mengatur tata cara bekerja dengan aman sesuai jenis aktivitasnya. SOP harus diterapkan secara disiplin, baik dalam penggunaan alat berat, pekerjaan di ketinggian, pekerjaan listrik, maupun pekerjaan dengan bahan kimia berbahaya.

2.8.3 Contoh Sistem Keamanan Berbasis Teknologi

Dalam era digital, sistem keamanan proyek mulai memanfaatkan teknologi modern untuk meningkatkan keselamatan kerja. Beberapa contoh teknologi yang digunakan antara lain:

1. Helm Pintar: Dilengkapi sensor untuk mendeteksi benturan, suhu tubuh, posisi jatuh, dan lokasi GPS pekerja secara *real-time*.
2. Kamera CCTV: Digunakan untuk memantau aktivitas pekerja dan mengidentifikasi potensi pelanggaran keselamatan di area proyek.

3. Sensor Gas Berbahaya: Mendeteksi keberadaan gas beracun atau mudah meledak di area tertutup, sehingga dapat segera diatasi.
4. Aplikasi Laporan K3: Aplikasi *mobile* yang memungkinkan pekerja melaporkan kondisi berbahaya, insiden, atau pelanggaran SOP secara cepat.
5. Sistem Alarm Darurat Otomatis: Mengeluarkan peringatan jika terdeteksi situasi darurat, seperti kebocoran gas, kebakaran, atau kecelakaan kerja.

2.8.4 Manfaat Sistem Keamanan Pekerja Proyek

Penerapan sistem keamanan yang baik tidak hanya melindungi pekerja, tetapi juga memberikan dampak positif bagi kelancaran proyek secara keseluruhan. Beberapa manfaat utama dari sistem keamanan pekerja proyek antara lain:

1. Mengurangi kecelakaan kerja dan biaya pengobatan.
2. Meningkatkan produktivitas karena pekerja merasa aman.
3. Mengurangi risiko hukum akibat pelanggaran keselamatan.
4. Meningkatkan citra perusahaan terhadap mitra dan publik.

2.8.5 Tantangan dalam Implementasi

Meskipun penting, penerapan sistem keamanan di proyek tidak selalu berjalan mulus. Terdapat berbagai tantangan yang sering dihadapi, seperti:

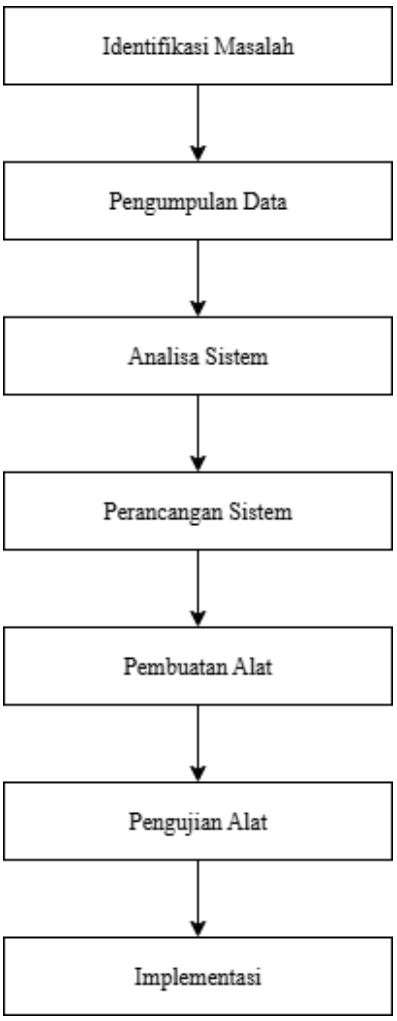
1. Kurangnya kesadaran pekerja akan pentingnya K3.
2. Biaya tinggi dalam pengadaan teknologi keamanan modern.
3. Sulitnya memantau kepatuhan SOP di proyek besar.

BAB III

METODE PENELITIAN

3.1 Kerangka Kerja Penelitian

Dalam kerangka kerja penelitian akan digambarkan langkah-langkah dari sebuah penelitian yang dilakukan untuk mempermudah tahapan-tahapan dari pembuatan alat yang akan dibuat. Kerangka kerja penelitian yang akan dilakukan, dibagi pada gambar 3.1 di bawah ini



Gambar 3. 1 Kerangka Penelitian

3.2 Uraian Kerangka Kerja Penelitian

Adapun tahapan-tahapan yang ditempuh dalam penelitian ini disusun secara sistematis, dilakukan dengan perencanaan yang matang, serta dikaji secara cermat guna memperoleh hasil yang optimal sebagai berikut:

3.2.1 Identifikasi Masalah

Identifikasi masalah dilakukan dengan pendekatan langsung terhadap objek penelitian untuk memahami permasalahan secara menyeluruh. Tujuan dari tahap ini adalah untuk mengetahui permasalahan yang terjadi secara cepat, sehingga diharapkan penelitian dapat memberikan solusi yang paling optimal terhadap pemecahan masalah tersebut.

Permasalahan yang berhasil diidentifikasi adalah bagaimana merancang dan membangun sistem keamanan pekerja proyek dengan menggunakan helm pintar berbasis ESP. Saat ini, kecelakaan kerja pada proyek konstruksi masih sering terjadi akibat minimnya pengawasan langsung dan kurangnya pemantauan terhadap penggunaan alat pelindung diri seperti helm keselamatan. Selain itu, tidak ada sistem yang mampu memberikan peringatan dini atau pelaporan otomatis jika pekerja berada dalam kondisi tidak aman seperti tidak memakai helm, jatuh, atau suhu tubuh meningkat.

Kondisi ini menjadi tantangan tersendiri dalam meningkatkan efektivitas keselamatan kerja di lapangan. Padahal, dengan memanfaatkan teknologi *Internet of Things* (IoT) seperti ESP32 yang terintegrasi dengan sensor-sensor tertentu, dapat

dibangun sebuah sistem helm pintar yang mampu memantau kondisi pekerja dan mengirimkan data secara *real-time* kepada pengawas proyek. Oleh karena itu, dibutuhkan sebuah sistem yang dapat memberikan kontribusi nyata dalam mengurangi tingkat kecelakaan kerja dan meningkatkan pengawasan terhadap pekerja proyek melalui pendekatan teknologi berbasis helm pintar.

3.2.2 Pengumpulan Data

Berdasarkan metode yang dilakukan pada saat melakukan proses penelitian untuk memperoleh data yang akurat dan juga dalam pembuatan alat dan pengerjaan laporan ini peneliti menggunakan metodologi penelitian sebagai berikut :

3.2.1.1 Waktu Penelitian

Penelitian dilakukan dengan memproses data–data yang didapat oleh peneliti, pengambilan data yang dilakukan dari bulan Maret 2025 hingga Agustus 2025.

Tabel 3. 1 Jadwal Penelitian

No	Kegiatan	2025																			
		Maret		April				Mei				Juni				Juli				Agust	
		3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2
1	Identifikasi Masalah	■	■	■																	
2	Pengumpulan Data			■	■	■	■														
3	Analisa Sistem						■	■	■	■	■										
4	Perancangan Sistem										■	■	■	■							
5	Pembuatan Sistem													■	■	■	■				
6	Pengujian Sistem																■	■	■	■	
7	Implementasi Sistem																		■	■	■

3.2.2.2 Metode Penelitian

Dalam melakukan penelitian agar mendapatkan hasil seperti yang diinginkan, maka sekiranya diperlukan suatu metodologi penelitian yang umum dilakukan yaitu :

1. Penelitian Perpustakaan *(Library Research)*

   Penelitian perpustakaan ini dilakukan dengan cara membaca, membahas, meringkas dan membuat kesimpulan dari jurnal tentang komponen apa saja yang digunakan pada alat dan *programming* yang berkaitan dengan analisa dan perancangan alat yang dibuat dengan algoritma pemrograman C++ untuk mendapatkan bahan-bahan yang secara ilmiah dapat dijadikan landasan dalam menyusun penelitian ini.

2. Penelitian Lapangan *(Field Research)*

   Pada penelitian alat ini dilakukan pemilihan alat yang terkait sesuai dengan alat yang dibuat serta mengetahui bagaimana seseorang dapat mengerti dalam penggunaan alat yang akan digunakan dan memudahkan seseorang dalam suatu pekerjaan.

3. Penelitian Laboratorium *(Laboratory Research)*

   Metode ini dilakukan untuk menguji konsep-konsep yang ada dengan menggunakan peralatan yang dipakai dan sesuai. Adapun objek yang diuji spesifikasi *hardware* dan *software* yang digunakan dalam penulisan ini dapat dilihat pada tabel 3.2. :

Tabel 3. 2 Spesifikasi *Hardware* dan *software*

Hardware	***Software***
Laptop MSI GF63 Thin 10UC Intel(R) Core(TM) i5-105005H CPU @ 2.50GHz (12 CPus), ~2.5GHz RAM 16384MB SSD 1024GB	Windows 11 *Home Single Language* 64-bit Microsoft Office 2019 Fritzing SketchUp Arduino IDE

3.2.3 Analisis Sistem

Tahap analisa merupakan tahap yang paling penting dalam pengembangan sebuah sistem, karena pada tahap inilah nantinya dilakukan evaluasi dari sistem, serta rancangan sistem dan langkah-langkah yang dibutuhkan untuk perancangan sistem dari alat yang akan dibuat sampai mendapatkan analisis yang diharapkan. Peneliti melakukan analisa data yang bertujuan untuk memecahkan masalah agar dapat menghasilkan sebuah solusi yang baru sebelum melakukan sebuah perancangan alat.

Dalam penelitian ini, peneliti ingin menjelaskan *input, output*, dan proses dari sistem yang akan dibuat. *Input* adalah sebuah masukan yang akan diproses dan menghasilkan keluaran yaitu *output. Input* yang digunakan adalah sensor MPU-6050, sensor MAX30102, GPS Ublox Neo 7M, sensor MQ-135, *Power Supply,* Sedangkan *output* yang digunakan ialah *Vibration motor, Buzzer, LED,* dan Telegram.

3.2.4 Perancangan Sistem

Perancangan sistem merupakan tahap penting yang bertujuan untuk menggambarkan bentuk awal dari sistem keamanan yang akan dibangun pada helm

pintar berbasis ESP. Perancangan ini mencakup komponen-komponen perangkat keras (*hardware*) dan perangkat lunak (*software)*, serta alur kerja dari sistem agar dapat berjalan sesuai dengan kebutuhan di lapangan proyek.

Helm pintar ini dirancang untuk meningkatkan keselamatan kerja di area proyek konstruksi melalui pemantauan kondisi pekerja secara *real-time*. Sistem ini disusun berdasarkan sejumlah kriteria yang ditentukan berdasarkan kebutuhan di lapangan dan kemampuan teknologi yang tersedia, yaitu sebagai berikut:

a. Menggunakan sensor MQ-135 untuk mendeteksi konsentrasi gas berbahaya seperti karbon monoksida, amonia, dan asap. Sistem harus memberikan peringatan jika ambang batas gas berbahaya terlampaui.
b. Memanfaatkan sensor MPU6050 (*accelerometer dan gyroscope*), sistem dapat mendeteksi jika pekerja mengalami jatuh atau gerakan tidak normal, lalu mengirim sinyal bahaya ke sistem monitoring. Komponen yang mudah dibuat, serta mudah dalam pemeliharaan dan pengaplikasianya.
c. GPS UBLOX NEO-7M digunakan untuk memperoleh koordinat lokasi pekerja secara akurat. Lokasi ini dikirimkan saat terjadinya suatu bahaya kepada pekerja agar pengawas dapat mengetahui posisi pekerja.
d. Menggunakan sensor MAX30102 untuk membaca denyut jantung dan kadar oksigen dalam darah. Sistem akan mengirimkan peringatan apabila data menunjukkan kondisi abnormal (misalnya detak terlalu rendah/tinggi atau SpO2 rendah).
e. ESP32 digunakan sebagai pusat pengendali dan pengirim data dari seluruh sensor ke server atau *dashboard monitoring* secara *real-time* melalui jaringan WiFi.

f. Semua komponen dirancang agar ringan, nyaman digunakan dalam bentuk helm, dan ditenagai oleh baterai isi ulang yang tahan lama agar tetap aktif di lingkungan proyek.

3.2.5 Pembuatan Alat

Pembuatan alat didasari oleh beberapa pertimbangan-pertimbangan yang telah disebutkan pada perancangan alat sebelumnya seperti *Context Diagram*, *Data Flow Diagram*, blok diagram, *Flowchart*, dan *design*.

Pada proses pembuatan alat, semua data yang telah dikumpulkan dan riset yang sudah dilakukan akan diolah dan semua komponen utama seperti Helm Proyek, Sensor MPU6050, Sensor MAX30102, Sensor GPS UBLOX NEO7-M, *Power Supply*, dan Sensor MQ-135 untuk *input,* dan untuk *output* menggunakan *Buzzer, Vibration motor*, LED, dan Telegram.

Komponen ini kemudian akan disusun kedalam helm proyek yang sesuai dengan letaknya. Sensor MPU6050 berada didalam helm bagian atas, sensor MAX30102 dibagian kening dari penahan kepala, sensor MQ-135 dibagian depan helm yang telah di lubangi agar ujung sensor bisa mudah mendeteksi gas, dan untuk sensor GPS Ublox NEO7-M dibagian dalam helm yang berada di posisi samping dengan antennya berada diposisi luar pada helm. Posisi untuk *output*, seperti LED, *Buzzer*, dan *Vibration motor* akan diletakkan pada helm sesuai dengan *output* yang dikeluarkan agar mudah diketahui oleh pekerja apa yang terjadi. Komponen tersebut nantinya akan digabungkan dengan kabel *jumper* yang telah di solder ke ESP32 dengan

penghubungnya menggunakan PCB (*Printed Circuit Board*) agar komponen tidak mudah lepas.

Selanjutnya, dilakukan penyempurnaan terhadap sistem, baik dari sisi fisik maupun dari sisi perangkat lunak. Jika terjadi kendala dalam logika program atau sambungan rangkaian, makan akan dilakukan perbaikan dan penuyusan ulang hingga alat bekerja dengan baik.

Secara keseluruhan, tahapan pembuatan sistem merupakan perwujudan dari seluruh konsep, rancangan, dan analisis yang telah dilakukan pada tahap-tahap sebelumnya, hingga menghasilkan sebuah alat atau *prototype* yang siap diuji dan digunakan.

3.2.6 Pengujian Alat

Pengujian alat dibutuhkan agar alat yang dibuat sesuai dengan kebutuhan sehingga dapat digunakan oleh pengguna. Tujuan dari pengujian alat yaitu untuk melihat seberapa tepat alat yang dibuat sehingga sesuai dengan kebutuhan dan mengidentifikasi masalah yang terjadi pada sistem. Pengujian dilakukan secara bertahap dengan berbagai skenario untuk memastikan semua fitur berfungsi dengan baik.

Uji coba yang dilakukan pertama kali adalah mencoba apakah Sensor dari UBLOX NEO7-M mampu menangkap kordinat sekitar. Kemudian dilakukan nya uji coba pada sensor lainnya seperti Sensor MPU6050, MAX30102, dan MQ-135 yang dimana sensor tersebut memiliki masing-masing *output* seperti *Buzzer, Vibration motor*, dan LED. Telegram akan mengeluarkan notifikasi yang sesuai dengan kendala

yang terjadi pada Lokasi. Hasil dari pengujian ini akan menjadi dasar untuk melakukan perbaikan atau penyempurnaan agar alat dapat bekerja secara optimal.

3.2.7 Impelementasi

Implementasi sistem merupakan tahap meletakkan sistem sehingga siap untuk dioperasikan. Implementasi sistem juga merupakan hasil dari perwujudan sistem yang dibangun, apakah sistem tersebut sudah mampu diterapkan dan beroperasi sesuai dengan yang diinginkan peneliti. Hal ini dibutuhkan agar alat yang dibuat sesuai dengan kebutuhan agar dapat digunakan oleh pengguna.

Tujuan dari implementasi yaitu untuk melihat seberapa tepat alat yang dibuat sehingga sesuai dengan kebutuhan dan mengidentifikasi masalah yang terjadi pada sistem. Implementasi alat ini ditujukan kepada pekerja proyek yang ingin pekerjaannya lebih aman saat berada dilokasi.

Pada bab ini dijelaskan proses implementasi dari sistem keamanan pekerja proyek dengan helm pintar berbasis ESP32. Implementasi mencakup pemasangan perangkat keras, pemograman mikrokontroler, serta pengujian awal fungsi dasar alat. Tujuan dari implementasi ini adalah merealisasikan helm pintar yang dapat membantu pekerja proyek melakukan pekerjaan.

BAB IV

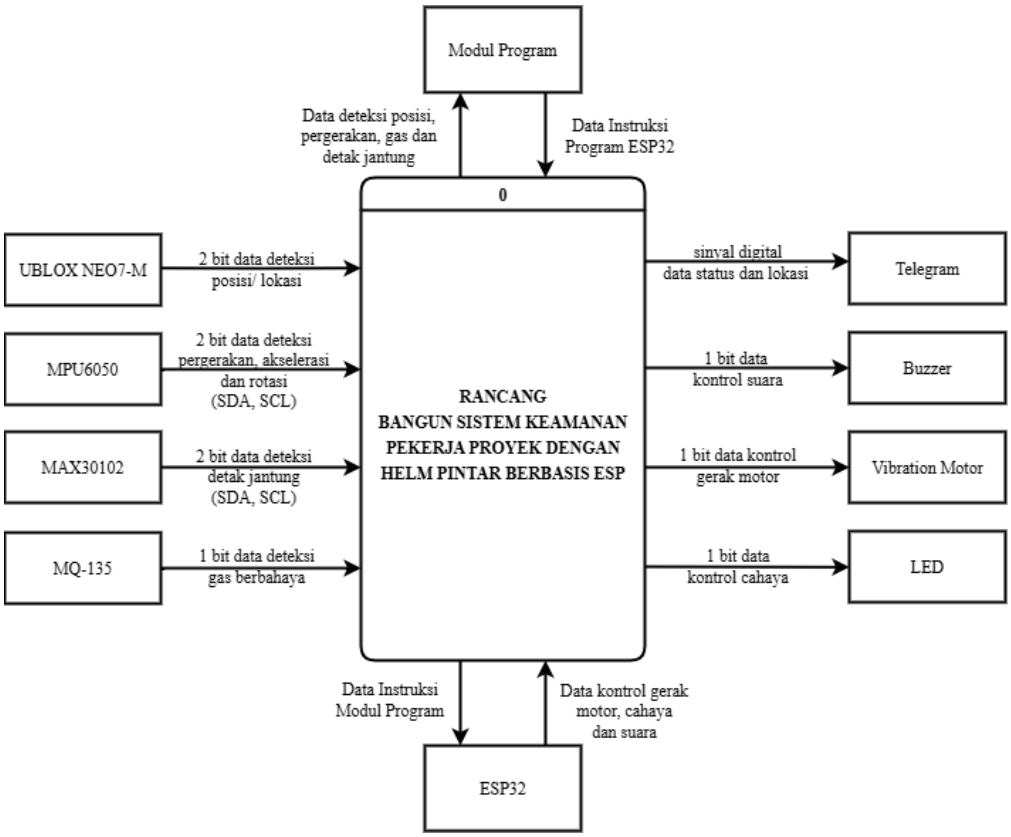
ANALISA DAN HASIL

4.1 Desain Sistem Secara Umum

Desain sistem merupakan rancangan gambaran sistem secara keseluruhan. Analisa sistem setidaknya membutuhkan pendefinisian dari sistem yang akan dirancang melalui desain sistem yang ada. Hal ini diperuntukan untuk memberikan gambaran secara jelas mengenai ruang lingkup permasalahan dan pembahasan dari penelitian yang dilakukan melalui media berupa *Context Diagram, Data Flow Diagram, dan block diagram.*

4.1.1 Context Diagram

Adapun *Context Diagram* merupakan gambaran dari sistem yang akan dirancang dan bersifat umum serta menyeluruh. *Context Diagram* memberikan gambaran berupa batasan entitas yang digunakan dalam sistem dan memberikan gambaran interaksi salah satu entitas dalam sistem terhadap entitas lainnya. Hal ini ditujukan untuk memudahkan proses penganalisaan sistem yang dirancang. *Context Diagram* dari sistem yang dirancang dapat dilihat pada Gambar 4.1:



Gambar 4. 1 *Context Diagram*

Sistem akan saling berinteraksi dengan beberapa entitas yaitu ESP32, MPU6050, MAX30102, UBLOX NEO-7M, MQ-135, *Buzzer, Vibration motor*, LED, dan Telegram. Adapun fungsi dari entitas yang digunakan adalah sebagai berikut.

1. ESP32

   ESP32 berfungsi sebagai pusat pengolahan data intruksi dan pengontrol sistem yang dirancang.

2. Modul Program

   Modul program berfungsi sebagai pengendali utama logika sistem.

3. MPU6050

   MPU6050 berfungsi untuk mendeteksi adanya perubahan rotasi pada pekerja.

4. MAX30102

   Untuk mendeteksi detak jantung pada pekerja.

5. GPS UBLOX NEO-7M

   UBLOX NEO-7M berfungsi untuk memberikan Lokasi disaat kondisi pekerja mengalami bahaya.

6. MQ-135

   MQ-135 untuk mendeteksi adanya gas berbahaya pada Lokasi kerja.

7. *Buzzer*

   Buzzer berfungsi sebagai *output* dari sensor GPS UBLOX NEO-7M dan sensor MPU6050.

8. *Vibration motor*

   Vibration motor berfungsi sebagai *output* dari sensor MAX30102.

9. LED

   LED berfungsi sebagai *output* dari sensor MQ-135.

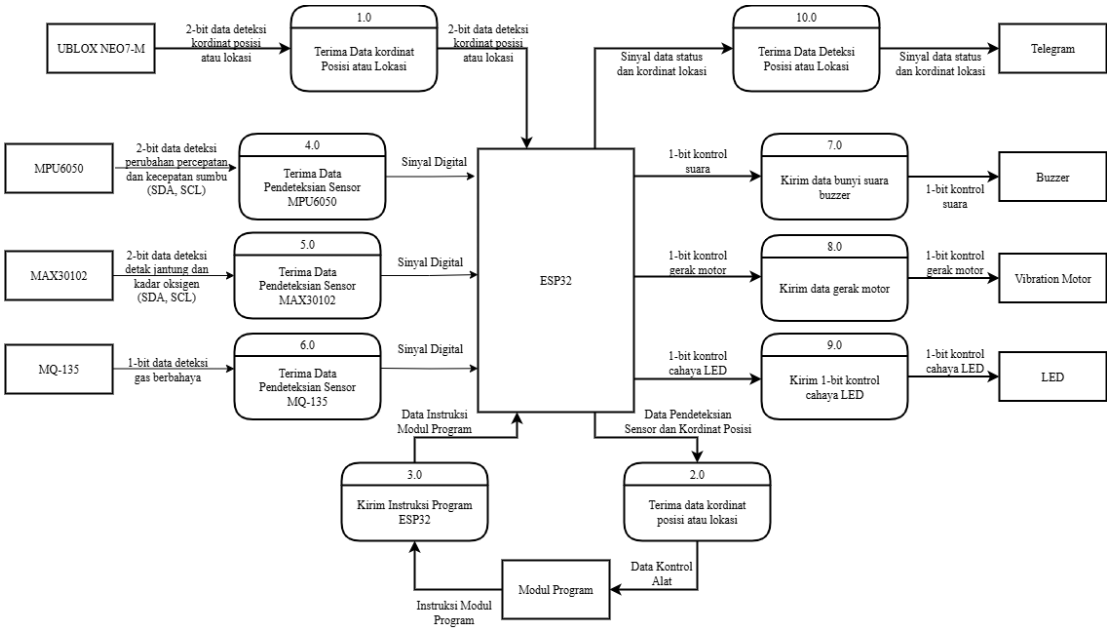
10. Telegram

    Telegram berfungsi untuk memberikan status pekerja disaat kondisi pekerja buruk dengan memberikan Lokasi nya menggunakan GPS UBLOX NEO-7M.

4.1.2 Data Flow Diagram

Diagram alir data merupakan gambaran yang lebih rinci tentang sistem yang akan dirancang dan berorientasi pada aliran data yang bergerak dalam suatu sistem. Diagram alir data digambarkan berdasarkan desain diagram konteks yang telah

dijelaskan sebelumnya. *Data Flow Diagram* yang akan dirancang pada sistem terdapat pada Gambar 4.2.



Gambar 4. 2 *Data Flow Diagram*

Berdasarkan DFD diatas dapa dijelaskan bahwa sistem ini memiliki beberapa rangkaian proses instruksi sebagai berikut.

1 Sensor GPS mendeteksi koordinat lokasi dan mengirimkan datanya ke ESP32 (1.0).

2 ESP32 mengirimkan data ke modul program untuk diproses(2.0).

3 Hasil Proses pada modul program dikirimkan Kembali ke ESP32 (3.0).

4 Sensor MPU6050 mendeteksi sumbu rotasi dan mengirimkan datanya ke ESP32 (4.0)

5 Sensor MAX30102 mendeteksi denyut jantung dan mengirimkan datanya ke ESP32 (5.0).

6 Sensor MQ-135 mendeteksi gas berbahaya dan mengirimkan datanya ke ESP32 (6.0).

7 ESP32 mengirimkan data ke *Buzzer* yang berasal dari data MPU6050 (7.0).

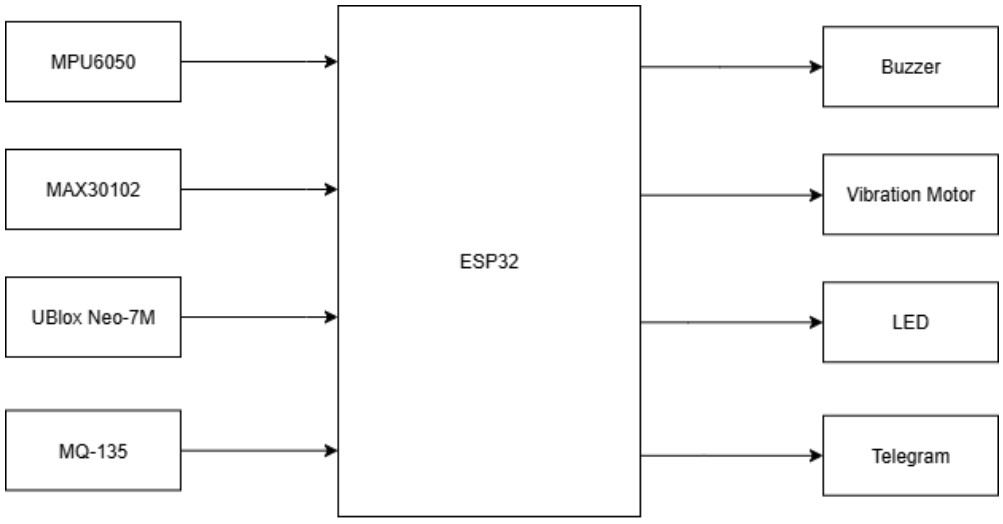
8 ESP32 mengirimkan data ke *Vibration Motor* yang berasal dari data MAX30102 (8.0).

9 ESP32 mengirimkan data ke LED yang berasal dari data MQ-135 (9.0).

10 ESP32 memberikan data instruksi ke Telegram berupa notifikasi yang berasal dari sensor MPU6050, MAX30102, MQ-135, dan UBLOX NEO-7M (10.0).

4.1.3 Blok Diagram

Diagram blok merupakan penggambaran suatu sistem yang dirancang dan disajikan dalam bentuk yang lebih sederhana untuk memberikan pemahaman dengan menjelaskan prinsip kerja alat secara keseluruhan, yaitu gabungan prinsip elektronika dan mekanika yang dikendalikan dengan pemrograman. Blok diagram pada alat akan dapat dilihat pada Gambar 4.3



Gambar 4. 3 Blok Diagram

Dari data blok diagram diatas dapat dilihat bahwa sistem terdiri atas *input*, *process*, dan *output*. ESP32 berfungsi sebagai pusat pengontrolan, MPU6050, MAX30102, UBLOX NEO-7M, dan MQ-135 merupakan *entity input.* Sedangkan *Buzzer*, *Vibration motor*, LED, dan Telegram merupakan *entity output.*

4.2 Prinsip Kerja Sistem

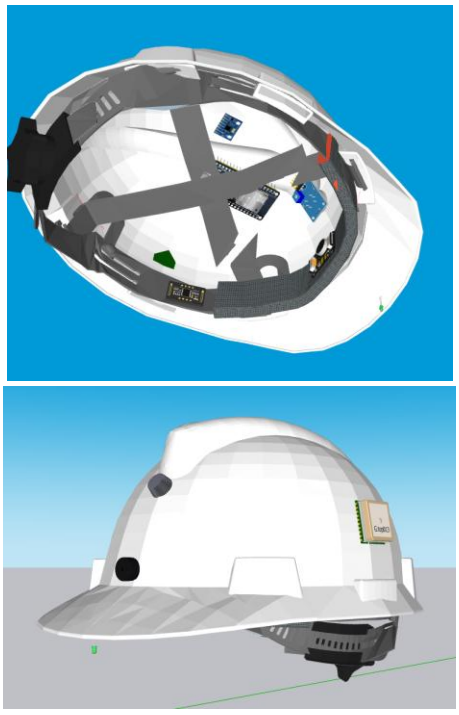
Prinsip dan sistem kerja alat dari helm pintar pekerja proyek sebagai berikut :

1. Ketika alat aktif, UBLOX NEO-7M akan mencari kordinat terlebih dahulu.
2. Sensor yang lain tidak akan aktif sampai GPS mendapatkan kordinat lokasinya.
3. Ketika mendapatkan kordinat, MPU6050, MAX30102, dan MQ-135 akan aktif
4. MPU6050 bekerja jika Sensor mendeteksi adanya benturan atau posisi terbalik.
5. *Buzzer* dari *output* MPU6050 berguna Ketika terjadi benturan atau posisi miring kurang dari 10 detik
6. Saat helm berada di posisi miring selama 10 detik, ESP32 akan mengirimkan data notifikasi ke telegram bahwa helm terbalik beserta dengan Lokasi nya.
7. MAX30102 berguna untuk mendeteksi denyut jantung pekerja.
8. *Vibration motor* sebagai *output* dari MAX30102 berguna ketika denyut jantung sedikit dibawah atau diatas rata-rata pada manusia kurang dari 10 detik.
9. Saat denyut jantung semakin menurun atau semakin meningkat dari biasanya selama 10 detik, ESP32 akan mengirimkan instruksi untuk mengirimkan notifikasi ke telegram bahwa denyut rendah atau tinggi beserta dengan Lokasi nya.
10. MQ-135 berguna untuk mendeteksi gas yang berbahaya pada Lokasi.

11. LED sebagai *ouput* dari Sensor MQ-135 akan aktif Ketika mendeteksi gas berbahaya berada diantara 75%-99%.
12. Saat gas berbahaya mencapai nilainya 100%, ESP32 akan memberikan instruksi ketelegram untuk mengirimkan notifikasi bahwasannya ada gas berbahaya pada Lokasi dan disertai juga Lokasi pekerja.

4.3 Rancangan Fisik Sistem

Perancangan alat ini merupakan tahap awal instalasi dan menganalisis permasalahan yang dihadapi berdasarkan literatur. Hal ini dilakukan untuk memberikan gambaran yang jelas mengenai alat yang akan dibuat sehingga tidak akan ada masalah dalam proses pembuatannya. Desain fisik alat dibuat menggunakan *software* Google Sketchup yang dapat dilihat pada Gambar 4.4



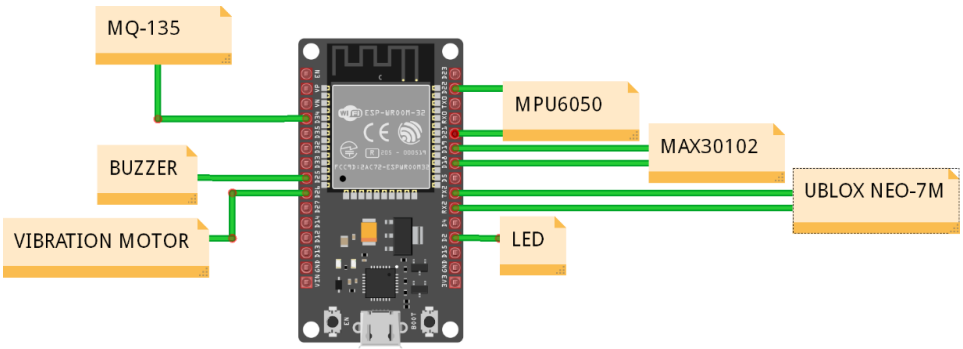
Gambar 4. 4 Rancangan Fisik Sistem

4.4 Desain Sistem Terperinci

Perancangan sistem merupakan gambaran sistem secara keseluruhan. Dengan gambaran umum, prinsip dan entitas sistem dapat dilihat dengan sangat jelas.

4.4.1 Rangkaian Microcontroller ESP32

Pada sistem ini digunakan mikrokontroler ESP32 sebagai pengontrol atau pengontrol sistem. ESP32 menggunakan chip Xtens a *dual core* LX6 - 160M Hz. Papan ini sangat lengkap untuk memenuhi kebutuhan sistem yang dibuat pada penelitian ini. Rangkaian ESP32 bisa terlihat pada Gambar 4.5.



Gambar 4. 5 Rangkaian *Microcontroller* ESP32

Berdasarkan gambar diatas, pin yang digunakan pada ESP32 adalah sebagai berikut:

1 Pin 21-22 pada ESP32 terhubung ke MPU6050.

2 Pin 18-19 pada ESP32 terhubung ke MAX30102.

3 Pin 16-17 pada ESP32 terhubung ke GPS UBLOX NEO-7M.

4 Pin 2 pada ESP32 terhubung ke LED.

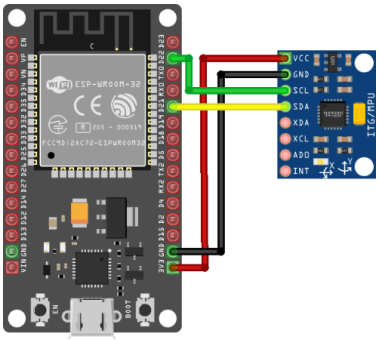
5 Pin 34 pada ESP32 terhubung ke MQ-135.

6 Pin 25 pada ESP32 terhubung ke *Buzzer*.

7 Pin 26 padaESP32 terhubung ke *Vibration motor.*

4.4.2 Rangkaian MPU6050

MPU6050 digunakan untuk mendeteksi rotasi pada pekerja dapat dilihat pada Gambar 4.6.



Gambar 4. 6 Rangkaian MPU6050

Berdasarkan gambar pin diatas, pin yang digunakan pada ESP32 adalah sebagai berikut:

1 Pin 3.3V ESP32 terhubung ke VCC

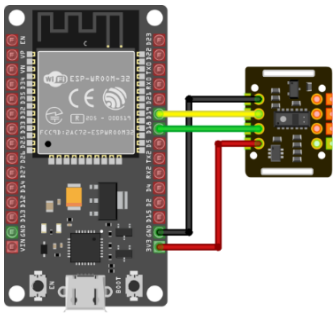
2 Pin GND ESP32 terhubung ke GND

3 Pin 21 ESP32 terhubung ke SDA

4 Pin 22 ESP32 terhubung ke SDL

4.4.3 Rangkaian MAX30102

Pada sistem ini, MAX30102 berfungsi sebagai pendeteksi denyut jantung yang dapat dilihat pada Gambar 4.7.



Gambar 4. 7 Rangkaian MAX30102

Berdasarkan gambar pin diatas, pin yang digunakan pada sensor MAX30102 adalah sebagai berikut:

1 Pin 3.3V ESP32 terhubung ke VCC

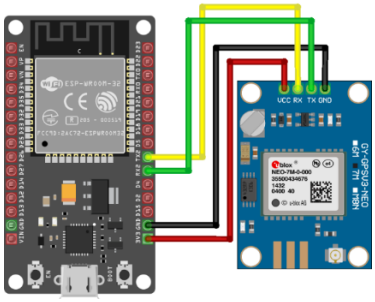
2 Pin GND ESP32 terhubung ke GND

3 Pin 19 ESP32 terhubung ke SCL

4 Pin 18 ESP32 terhubung ke SDA

4.4.4 Rangkaian GPS UBLOX NEO-7M

Pada sistem ini, GPS UBLOX NEO-7M berfungsi sebagai pendeteksi lokasi, rangkaian dari GPS UBLOX NEO-7M dapat dilihat pada Gambar 4.8.



Gambar 4. 8 Rangkaian UBLOX NEO-7M

Berdasarkan gambar pin diatas, pin yang digunakan pada GPS UBLOX NEO-7M adalah sebagai berikut:

1 Pin 5V ESP32 terhubung ke VCC

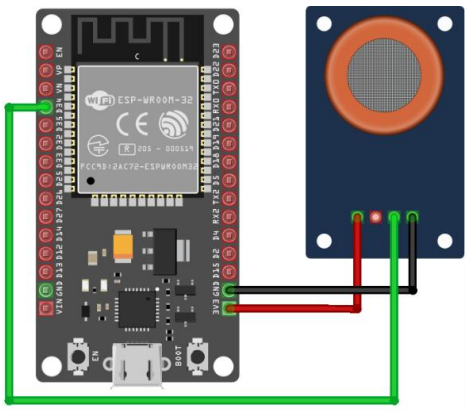
2 Pin GND ESP32 terhubung ke GND

3 Pin 16 ESP32 terhubung ke TX

4 Pin 17 ESP32 terhubung ke RX

4.4.5 Rangkaian MQ-135

Pada sistem ini, MQ-135 berfungsi sebagai pendeteksi gas berbahaya, rangkaian dari MQ-135 dapat dilihat pada Gambar 4.9.



Gambar 4. 9 Rangkaian MQ-135

Berdasarkan gambar pin diatas, pin yang digunakan pada Sensor MQ-135 adalah sebagai berikut:

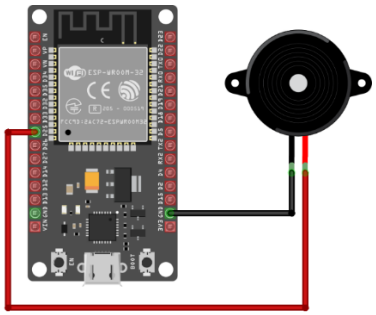
1 Pin 3.3V ESP32 terhubung ke VCC

2 Pin GND ESP32 terhubung ke GND

3 Pin 34 ESP32 terhubung ke AOUT

4.4.6 Rangkaian Buzzer

Pada sistem ini, *Buzzer* berfungsi sebagai keluaran pada *input* berupa suara, rangkaian dari *Buzzer* dapat dilihat pada Gambar 4.10.



Gambar 4. 10 Rangkaian *Buzzer*

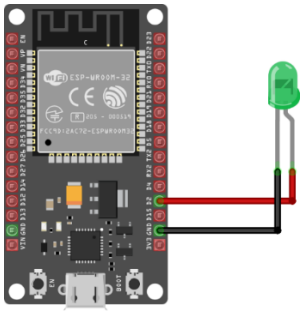
Berdasarkan gambar pin diatas, pin yang digunakan pada *Buzzer* adalah sebagai berikut:

1 Pin 25 ESP32 terhubung ke *Positive*

2 Pin GND ESP32 terhubung ke *Negatif*

4.4.7 Rangkaian LED

Pada sistem ini, LED berfungsi sebagai keluaran pada *input*, rangkaian dari LED dapat dilihat pada Gambar 4.11.



Gambar 4. 11 Rangkaian LED

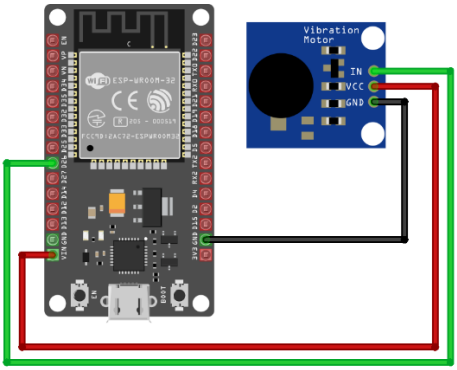
Berdasarkan gambar pin diatas, pin yang digunakan pada LED adalah sebagai berikut:

1 Pin 2 ESP32 terhubung ke anoda

2 Pin GND ESP32 terhubung ke katoda

4.4.8 Rangkaian Vibration motor

Pada sistem ini, *Vibration motor* berfungsi sebagai keluaran pada *input*, rangkaian dari *Vibration motor* dapat dilihat pada Gambar 4.12.



Gambar 4. 12 Rangkaian *Vibration motor*

Berdasarkan gambar pin diatas, pin yang digunakan pada *Vibration motor* adalah sebagai berikut:

1 Pin VIN pada ESP32 terhubung ke VCC

2 Pin GND pada ESP32 terhubung ke GND

3 Pin 26 Pada ESP32 terhubung ke IN

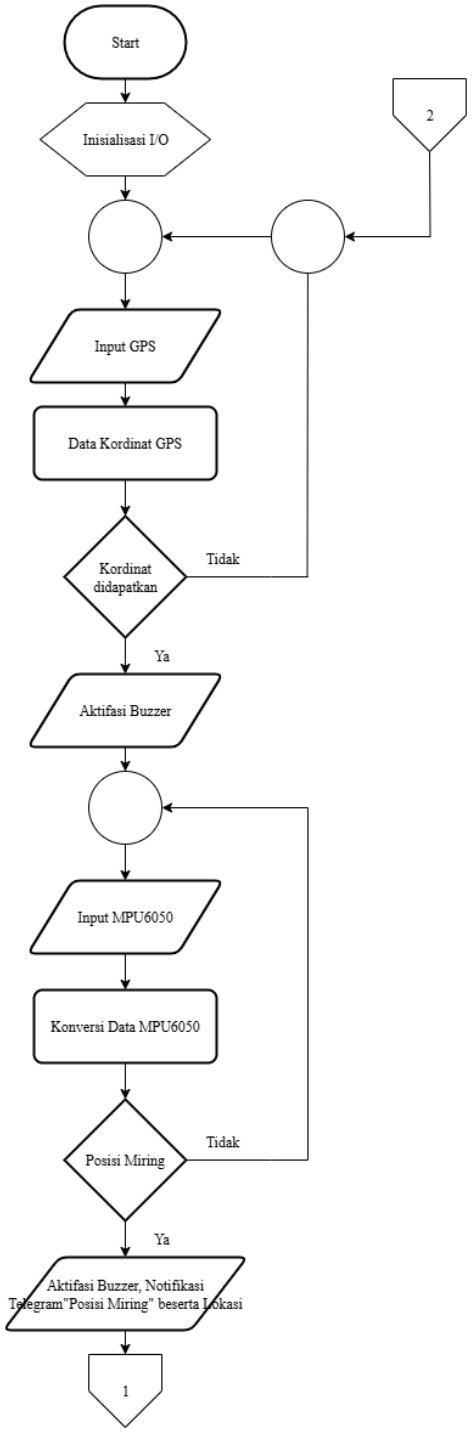
4.5 Rancangan Modul Program

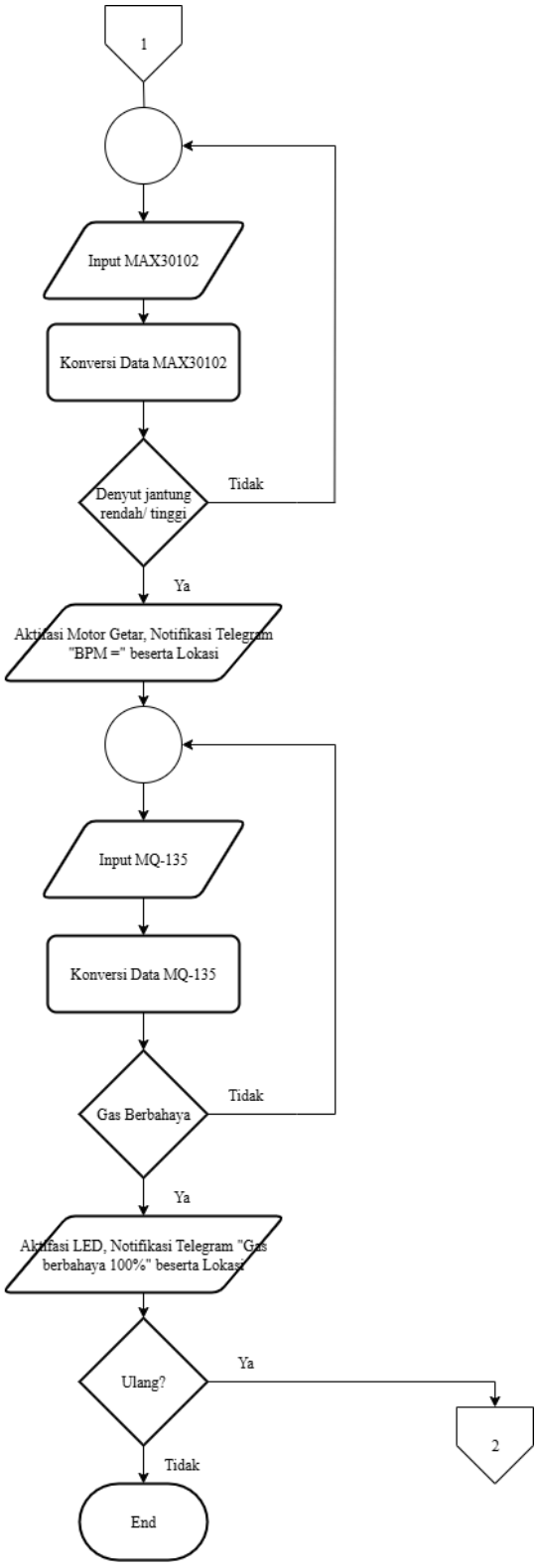
Perancangan modul program berisi penjelasan tentang modul program yang digunakan pada sistem yang telah dirancang. Desain modul program dibagi menjadi 2 yaitu *Flowchart* dan *listing program*.

4.5.1 Flowchart

Modul program pada dasarnya dirancang dengan baik dan mudah dipahami. Jadi sebelum membuat *listing program*, perlu ditentukan logika programnya. Logika

program ini adalah gambaran dasar penulisan dalam membuat *listing program*. Deskripsi dasar ini menggunakan diagram alur sebagai alat untuk membantu menentukan hal tersebut logika program yang dapat dilihat pada Gambar 4.15.





Gambar 4. 13 ***Flowchart***

4.5.2 Listing Program

Daftar program pada sub bab ini akan menjelaskan kode-kodenya modul program yang digunakan untuk membuat helm pintar untuk pekerja proyek menggunakan *software* Arduino IDE pada ESP32. Program keseluruhan adalah sebagai berikut.

```
#include <Wire.h>
#include <WiFi.h>
#include <HTTPClient.h>
#include "MAX30105.h"
#include "heartRate.h"
#include <MPU6050_light.h>
#include <TinyGPS++.h>

TwoWire WireMAX = TwoWire(0); // MAX30102 on pins set later
TwoWire WireMPU = TwoWire(1); // MPU6050 on pins set later

#define SDA_MAX 18
#define SCL_MAX 19
#define SDA_MPU 21
#define SCL_MPU 22
#define MQ135_PIN 34
#define BUZZER_PIN 25
#define MOTOR_PIN 26
#define LED_PIN 2

#define GPS_RX 16
#define GPS_TX 17

const char* ssid = "FrhnKnndy_";
const char* password = "knndy21019";
const char*botToken =
"7548099378:AAGC0IuwsMdVtvuLwcofCGu2_CoDEwpEKrI";
const char* chatID = "1199486758";

const int bpmDangerLow = 50;
const int bpmDangerHigh = 130;
const int gasWarning = 1300;
const int gasDanger = 1500;
const unsigned long TELEGRAM_COOLDOWN_MS = 15000; // 15s
const unsigned long I2C_CHECK_INTERVAL = 2000;
```

```
const long IR_DETECT_THRESHOLD = 5000;
const unsigned long BPM_STABILIZE_MS = 10000;
const unsigned long BPM_DANGER_DURATION = 5000;
const unsigned long MPU_DANGER_DURATION = 10000;
const unsigned long GAS_DANGER_DURATION = 10000;
const unsigned long MOTOR_PULSE_MS = 1000;
const float REBESAH_THRESHOLD_DEG = 70.0;

MAX30105 particleSensor;
MPU6050 mpu(WireMPU);
TinyGPSPlus gps;
HardwareSerial gpsSerial(1);

bool gpsReady = false;
bool koordinatPernahDidapat = false;
float gpsLat = 0, gpsLng = 0;

float beatsPerMinute = 0, beatAvg = 0;
long lastBeat = 0;
bool adaJari = false;
bool bpmBahaya = false, bpmKirim = false;
bool bpmStabil = false;
bool pernahAdaBPM = false;
unsigned long startDeteksiKulit = 0;
unsigned long startBpmBahaya = 0;

bool mpuBahaya = false, mpuKirim = false;
unsigned long startMpuBahaya = 0;
float prevAx = 0, prevAy = 0, prevAz = 0;
unsigned long lastMpuRead = 0;

bool gasBahaya = false, gasKirim = false;
unsigned long startGasBahaya = 0;
unsigned long lastGasRead = 0;
int mqBaseline = 0;

unsigned long motorOnTime = 0;
bool motorAktif = false;

unsigned long lastI2cCheck = 0;
unsigned long lastTelegramTime = 0;

bool maxReady = false;
bool mpuReady = false;
```

```
unsigned long motorToggleInterval = 400;
unsigned long motorLastToggle = 0;
bool motorState = false;

void setup() {
  Serial.begin(115200);
  delay(50);

  pinMode(BUZZER_PIN, OUTPUT);
  pinMode(MOTOR_PIN, OUTPUT);
  pinMode(LED_PIN, OUTPUT);
  matikanSemua();

  Serial.println(F("=== STARTUP ==="));

  Serial.printf(" 🪫 Menghubungkan ke WiFi: %s\n", ssid);
  WiFi.begin(ssid, password);
  unsigned long wifiStart = millis();
  while (WiFi.status() != WL_CONNECTED && millis() - wifiStart < 10000) {
    delay(300); Serial.print(".");
  }
  if (WiFi.status() == WL_CONNECTED) Serial.println("\n✅ WiFi Terhubung");
  else Serial.println("\n⚠️ WiFi gagal terhubung");

  gpsSerial.begin(9600, SERIAL_8N1, GPS_RX, GPS_TX);
  Serial.println(" 🔎 Inisialisasi GPS NEO-7M...");

  WireMAX.begin(SDA_MAX, SCL_MAX, 100000);
  WireMPU.begin(SDA_MPU, SCL_MPU, 100000);

  Serial.println(" 🔎 Inisialisasi MAX30102...");
  if (particleSensor.begin(WireMAX)) {
    particleSensor.setup();
    particleSensor.setPulseAmplitudeRed(0x3F);
    particleSensor.setPulseAmplitudeGreen(0);
    maxReady = true;
    Serial.println(" ✅ MAX30102 siap.");
  } else {
    Serial.println(" ❌ MAX30102 tidak terdeteksi!");
    maxReady = false;
  }

  Serial.println(" 🔎 Inisialisasi MPU6050...");
```

```
  if (mpu.begin() == 0) {
    mpu.calcOffsets();
    mpuReady = true;
    Serial.println("✅ MPU6050 siap.");
  } else {
    Serial.println("❌ MPU6050 gagal inisialisasi!");
    mpuReady = false;
  }

  Serial.println("🔪 Kalibrasi MQ135 baseline...");
  long total = 0;
  const int samples = 100;
  for (int i = 0; i < samples; ++i) {
    total += analogRead(MQ135_PIN);
    delay(10);
  }
  mqBaseline = total / samples;
  Serial.printf("🔪 MQ135 baseline = %d\n", mqBaseline);

  Serial.println(F("=== SETUP COMPLETE ==="));
}

void loop() {
  unsigned long now = millis();

  cekGPS();

  if (!koordinatPernahDidapat) {
    if (gpsReady) {
      koordinatPernahDidapat = true;
      digitalWrite(BUZZER_PIN, HIGH);
      delay(200);
      digitalWrite(BUZZER_PIN, LOW);
      Serial.printf("✅ GPS fix pertama: %.6f, %.6f\n", gpsLat, gpsLng);
    } else {
      Serial.println("⏳ Menunggu GPS fix...");
      delay(500);
      return;
    }
  }

  if (now - lastI2cCheck >= I2C_CHECK_INTERVAL) {
    lastI2cCheck = now;
    cekKoneksiPerangkat();
```

```
  }
  if (maxReady) cekBPM();
  if (mpuReady && now - lastMpuRead >= 200) {
    lastMpuRead = now;
    cekMPU();
  }
  if (now - lastGasRead >= 1000) {
    lastGasRead = now;
    cekGas();
  }
  kontrolMotor();
  delay(10);
}

void cekGPS() {
  while (gpsSerial.available() > 0) {
    gps.encode(gpsSerial.read());
  }
  if (gps.location.isValid()) {
    gpsLat = gps.location.lat();
    gpsLng = gps.location.lng();
    gpsReady = true;
    Serial.printf("[GPS] Koordinat: %.6f, %.6f\n", gpsLat, gpsLng);
  } else {
    gpsReady = false;
    Serial.println("[GPS] Belum mendapatkan fix...");
  }
}

bool i2cDevicePresent(TwoWire &bus, uint8_t address) {
  bus.beginTransmission(address);
  uint8_t e = bus.endTransmission();
  return (e == 0);
}

void cekKoneksiPerangkat() {
  if (!i2cDevicePresent(WireMAX, 0x57)) {
    if (particleSensor.begin(WireMAX)) {
      particleSensor.setup();
      particleSensor.setPulseAmplitudeRed(0x3F);
      particleSensor.setPulseAmplitudeGreen(0);
      maxReady = true;
    } else maxReady = false;
  }
  if (!i2cDevicePresent(WireMPU, 0x68)) {
```

```
    if (mpu.begin() == 0) {
      mpu.calcOffsets();
      mpuReady = true;
    } else mpuReady = false;
  }
}

void cekBPM() {
  long irValue = particleSensor.getIR();
  adaJari = (irValue > IR_DETECT_THRESHOLD);

  if (!adaJari) {
    Serial.println("[BPM] Tidak ada jari terdeteksi.");
    if (bpmStabil && (millis() - startDeteksiKulit > 10000)) bpmStabil = false;
    return;
  }

  if (!bpmStabil) {
    if (startDeteksiKulit == 0) startDeteksiKulit = millis();
    if (millis() - startDeteksiKulit < BPM_STABILIZE_MS) {
      Serial.println("[BPM] Menstabilkan pembacaan...");
      return;
    }
    bpmStabil = true;
    Serial.println("[BPM] Stabil, mulai membaca BPM...");
  }

  if (checkForBeat(irValue)) {
    long delta = millis() - lastBeat;
    lastBeat = millis();
    beatsPerMinute = 60.0 / (delta / 1000.0);
    if (beatsPerMinute >= 20 && beatsPerMinute <= 200) {
      beatAvg = (beatAvg * 3.0 + beatsPerMinute) / 4.0;
      pernahAdaBPM = true;
    }
  }

  Serial.printf("[BPM] IR: %ld | BPM: %.2f | Avg: %.2f\n", irValue, beatsPerMinute,
beatAvg);

  unsigned long now = millis();

  if ((beatAvg >= 50 && beatAvg <= 60) || (beatAvg >= 120 && beatAvg <= 130)) {
    if (!motorAktif) {
      motorAktif = true;
```

```
    motorOnTime = now;
    digitalWrite(MOTOR_PIN, HIGH);
   }
  }

  if ((beatAvg < 50 || beatAvg > 130) && pernahAdaBPM) {
   if (!bpmBahaya) {
    bpmBahaya = true;
    startBpmBahaya = now;
    bpmKirim = false;
   }

   if (!motorAktif) {
    motorAktif = true;
    motorOnTime = now;
    digitalWrite(MOTOR_PIN, HIGH);
   }

   if (now - startBpmBahaya >= BPM_DANGER_DURATION && !bpmKirim) {
    kirimTelegramWithCooldown("🚨 BPM Bahaya: " + String(beatAvg));
    bpmKirim = true;
    Serial.println("[BPM] Telegram dikirim karena BPM bahaya!");
   }
  } else if (!((beatAvg >= 40 && beatAvg <= 55) || (beatAvg >= 125 && beatAvg <=
140))) {
   // Reset flag jika BPM normal
   bpmBahaya = false;
   bpmKirim = false;
  }

  if (motorAktif && now - motorOnTime >= MOTOR_PULSE_MS) {
   digitalWrite(MOTOR_PIN, LOW);
   motorAktif = false;
  }
}

void cekMPU() {
  mpu.update();
  float ax = mpu.getAccX();
  float ay = mpu.getAccY();
  float az = mpu.getAccZ();
  float delta = abs(ax - prevAx) + abs(ay - prevAy) + abs(az - prevAz);
  prevAx = ax; prevAy = ay; prevAz = az;
```

```
  float pitch = atan2(ax, sqrt(ay*ay + az*az)) * 180.0 / PI;
  float roll  = atan2(ay, sqrt(ax*ax + az*az)) * 180.0 / PI;

   bool miring = (abs(pitch) > REBESAH_THRESHOLD_DEG || abs(roll) > REBESAH_THRESHOLD_DEG);
  bool benturan = (delta > 3.0);

  Serial.printf("[MPU] Accel(X: %.2f, Y: %.2f, Z: %.2f) | Pitch: %.2f | Roll: %.2f | Miring: %d | Benturan: %d\n",
                ax, ay, az, pitch, roll, miring, benturan);

  static unsigned long lastBuzzerToggle = 0;
  unsigned long now = millis();

  if (benturan) {
      digitalWrite(BUZZER_PIN, HIGH); delay(200); digitalWrite(BUZZER_PIN, LOW);
    if (!mpuBahaya) { startMpuBahaya = now; mpuKirim = false; mpuBahaya = true; }
    if (now - startMpuBahaya >= MPU_DANGER_DURATION && !mpuKirim) {
      kirimTelegramWithCooldown("🪖 Helm terbentur keras!");
      mpuKirim = true;
    }
  } else if (miring) {
    if (!mpuBahaya) { startMpuBahaya = now; mpuKirim = false; mpuBahaya = true; }
    if (now - startMpuBahaya >= MPU_DANGER_DURATION && !mpuKirim) {
      kirimTelegramWithCooldown("🪖 Helm miring/rebahan!");
      mpuKirim = true;
    }
    if (now - lastBuzzerToggle >= 1000) {
      lastBuzzerToggle = now;
      int currentState = digitalRead(BUZZER_PIN);
      digitalWrite(BUZZER_PIN, !currentState);
    }
  } else {
    mpuBahaya = false;
    mpuKirim = false;
    digitalWrite(BUZZER_PIN, LOW);
  }
}

void updateLedGasBlink(int gasPersen) {
  static unsigned long lastToggle = 0;
  static bool ledState = LOW;
  unsigned long now = millis();
```

```
  if (gasPersen >= 75) {
    if (now - lastToggle >= 1000) {
      lastToggle = now;
      ledState = !ledState;
      digitalWrite(LED_PIN, ledState);
    }
  } else {
    digitalWrite(LED_PIN, LOW);
    ledState = LOW;
    lastToggle = now;
  }
}

void cekGas() {
  int nilaiGas = analogRead(MQ135_PIN);
  int gasPersen = constrain(map(nilaiGas, 500, 1500, 0, 100), 0, 100);

  Serial.printf("[GAS] Nilai: %d | Persen: %d%%\n", nilaiGas, gasPersen);

  updateLedGasBlink(gasPersen);

  if (nilaiGas > gasWarning) {
    if (!gasBahaya) { startGasBahaya = millis(); gasKirim = false; gasBahaya = true; }
        if (nilaiGas > gasDanger && millis() - startGasBahaya >=
GAS_DANGER_DURATION && !gasKirim) {
      kirimTelegramWithCooldown("☠️ Gas beracun terdeteksi! (" + String(gasPersen)
+ "%)");
      gasKirim = true;
    }
  } else {
    gasBahaya = false;
    gasKirim = false;
  }
}

void aktifkanMotor() {
  if (!motorAktif) {
    motorAktif = true;
    motorOnTime = millis();
    digitalWrite(MOTOR_PIN, HIGH);
  }
}

void kontrolMotor() {
```

```
  if (motorAktif && millis() - motorOnTime >= MOTOR_PULSE_MS) {
    digitalWrite(MOTOR_PIN, LOW);
    motorAktif = false;
  }
}

void matikanSemua() {
  digitalWrite(LED_PIN, LOW);
  digitalWrite(BUZZER_PIN, LOW);
  digitalWrite(MOTOR_PIN, LOW);
  motorAktif = false;
}

void kirimTelegramWithCooldown(String pesan) {
  unsigned long now = millis();
  if (now - lastTelegramTime < TELEGRAM_COOLDOWN_MS) {
    return;
  }
  lastTelegramTime = now;
  String lokasi = "";
  if (gpsReady) {
     lokasi = "\nLokasi: https://maps.google.com/?q=" + String(gpsLat, 6) + "," +
String(gpsLng, 6);
  } else {
    lokasi = "\nLokasi: GPS tidak tersedia";
  }
  kirimTelegram(pesan + lokasi);
}

void kirimTelegram(String pesan) {
  if (WiFi.status() != WL_CONNECTED) return;
  HTTPClient http;
  String url = "https://api.telegram.org/bot" + String(botToken) + "/sendMessage";
  http.begin(url);
  http.addHeader("Content-Type", "application/x-www-form-urlencoded");
  String body = "chat_id=" + String(chatID) + "&text=" + " ⚠️ " + pesan;
  http.POST(body);
  http.end();
}
```

BAB V

PENGUJIAN SISTEM

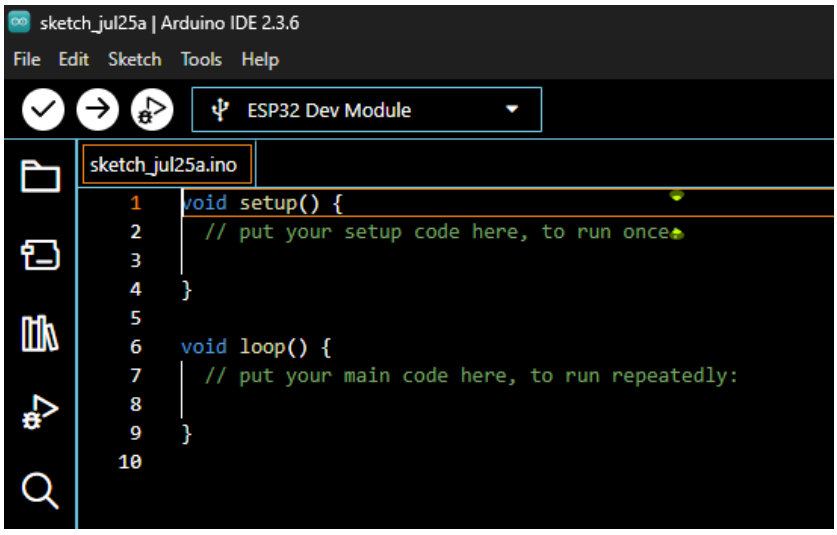
5.1 Pengujian Permodul

Pengujian permodul bertujuan untuk melihat hasil entitas yang dirancang sudah sesuai dengan tujuan yang ingin dicapai. Pengujian permodul dilakukan mulai dari pengujian kompnen hingga modul program secara keseluruhan. Berikut merupakan langkah-langkah dari pengujian permodul pada sistem yang dirancang.

5.1.1 Pengujian Software Arduino IDE

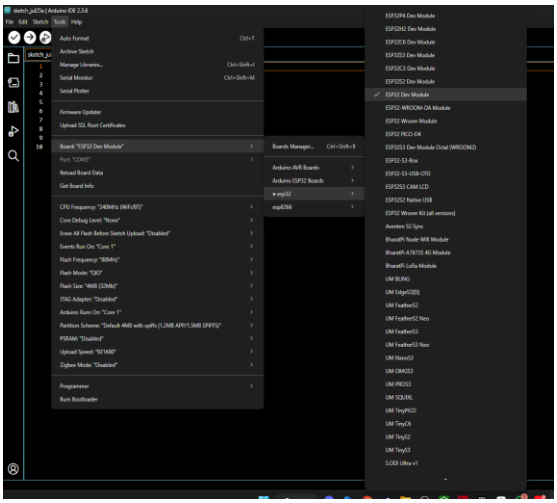
ESP32 menggunakan *software* Arduino IDE pada dalam pembuatan program dan *download* program ke Arduino *Board*. Beberapa Langkah yang perlu dilakukan dalam pemrograman C pada ESP32 adalah sebagai berikut:

1. Jalankan aplikasi Arduino IDE, sehingga akan muncul tampilan seperti Gambar 5.1.



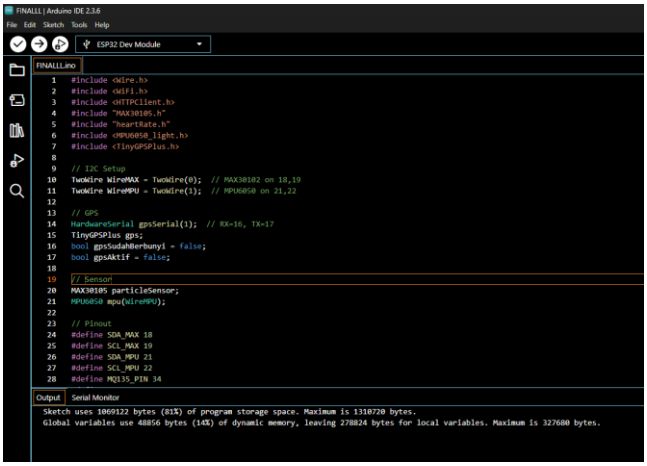
Gambar 5. 1 Tampilan Awal Arduino IDE

2. Masukkan program helm proyek pada jendela sketch yang terbuka.
3. Simpan program dengan mengklik tombol "*file*", lalu pilih "*save*".
4. Pasangkan kabel penghubung antara ESP32 dan PC untuk mengunggah program yang akan digunakan.
5. Cek dan pilih tipe *board* dan *port* yang sesuai dengan ESP yang digunakan seperti pada Gambar 5.2.



Gambar 5. 2 Pemilihan Tipe *Board* dan *Port*

6. Upload program hingga terdapat jendela seperti Gambar 5.3.



Gambar 5. 3 Proses Upload Progam Selesai

5.1.2 Hasil Pengujian Sensor MPU6050

Pengujian sensor MPU6050 dilakukan untuk mengetahui *output* yang akan diberikan. Hasil pengujian sensor MPU6050 dapat dilihat pada Tabel 5.1.

Tabel 5. 1 Hasil Pengujian Sensor MPU6050

No	*Input*	Nilai	*Output*
1	Sumbu Z	Diatas 0.15	Tidak Aktif
		Dibawah 0.15	Buzzer aktif dan mengirimkan notifikasi jika sudah 10 detik
2	Delta	Dibawah 3.0	Tidak Aktif
		Diatas 3.0	*Buzzer* Aktif

Berdasarkan hasil pengujian pada table 5.1, diketahui jika sumbu z menerima nilai diatas 0.15, *Buzzer* akan aktif dan mengirimkan ke telegram jika kondisi nilai 0.15 sudah 10 detik. Delta sebagai pendeteksi nilai benturan jika nilai berada diatas 3.0, *Buzzer* akan aktif.

5.1.3 Hasil Pengujian Sensor MAX30102

Pengujian sensor MAX30102 dilakukan untuk mengetahu *output* yang akan diberikan. Hasil pengujian sensor MAX30102 dapat dilihat pada Tabel 5.2.

Tabel 5. 2 Hasil Pengujian Sensor MAX30102

No	Nilai BPM	*Output*
1	40-55 dan 125-140	*Vibration Motor* Aktif
2	Bpm > 40 dan Bpm <140	Notifikasi ke Telegram

Berdasarkan table 5.2 diatas diketahui jika detak jantung berada di antara 40-55 dan 125-140 *Vibration Motor* akan aktif. Pengujian notifikasi ke Telegram dilakukan Ketika denyut jantung sudah berada di batas tidak normal pada manusia (dibawah 30 dan diatas 150).

5.1.4 Hasil Pengujian Sensor GPS UBLOX NEO-7M

Pengujian sensor GPS UBLOX NEO-7M dilakukan untuk mengetahu *output* yang akan diberikan. Hasil pengujian sensor GPS UBLOX NEO-7M dapat dilihat pada Tabel 5.3

Tabel 5. 3 Hasil Pengujian Pengambilan Kordinat Sensor GPS

No	Jumlah Kordinat	Logika	*Ouput*
1	0	*Low*	Sensor MPU6050, MAX30102, dan MQ-135 Tidak Aktif
2	1	*High*	Sensor MPU6050, MAX30102, dan MQ-135 Aktif

Berdasarkan tabel 5.3 diatas, diketahui jika sensor GPS tidak mendapatkan kordinatnya, maka ketiga sensor tidak akan aktif. Sensor akan aktif jika sensor GPS bisa mendapatkan 1 kordinatnya. Hasil pengujian sensor lainnya dapat dilihat pada Tabel 5.4.

Tabel 5. 4 Hasil Pengujian Sensor GPS UBLOX NEO-7M

No	*Input*	Nilai Kondisi	*Output*
1	MPU6050	Nilai Z dibawah 0.15 selama 10 detik	Pengiriman Lokasi ke Telegram beserta indikasi

2	MAX30102	BPM dibawah 30 dan diatas 150	Pengiriman Lokasi ke Telegram beserta indikasi
3	MQ-135	Nilai Gas 100%	Pengiriman Lokasi ke Telegram beserta indikasi

Berdasarkan pada tabel 5.4 diatas, Sensor GPS akan mengirimkan Lokasi jika ada indikasi yang ditimbulkan dari sensor MPU6050, MAX30102, dan MQ-135.

5.1.5 Hasil Pengujian Sensor MQ-135

Pengujian sensor MQ-135 dilakukan untuk mengetahui *output* yang akan diberikan. Hasil pengujian sensor MQ-135 dapat dilihat pada Tabel 5.5.

Tabel 5. 5 Hasil Pengujian Sensor MQ-135

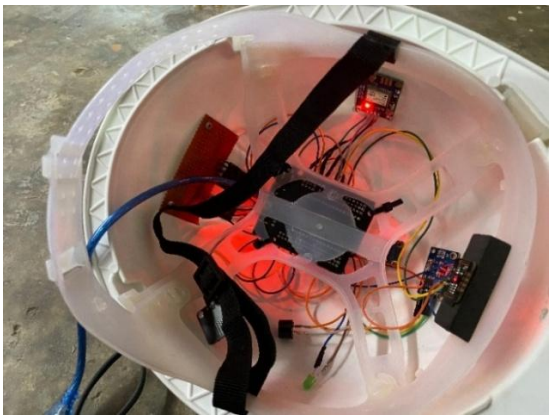
No	Persentase Gas	*Output*
1	75%	LED Aktif
2	100%	LED Aktif dan mengirimkan notifikasi ke Telegram

Berdasarkan hasil pengujian pada Tabel 5.5, diketahui saat nilai gas berada di 75%, LED akan aktif untuk memberikan peringatan langsung kepada pekerja. Saat persentase mencapai 100%, LED akan aktif dan memberikan notifikasi ke Telegram.

5.2 Pengujian Sistem Keseluruhan

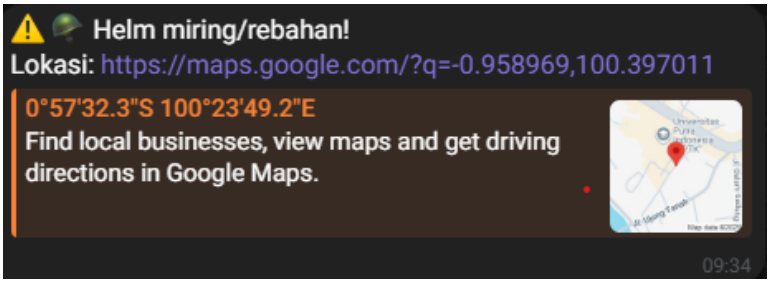
Pengujian keseluruhan adalah pengujian yang dilakukan menggunakan semua entitas yang ada berdasarkan langkah-langkah yang telah ditentukan sebelumnya. Langkah pengujian system keseluruhan dapat dilihat sebagai berikut.

1. Pertama adalah menghubungkan *power supply* dengan sumber arus listrik agar sistem dapat berfungsi dengan baik.
2. *Buzzer* akan berbunyi saat GPS mendapatkan kordinat agar bisa melacak Lokasi pekerja nantinya. Ketiga sensor akan aktif bersamaan saat kordinat telah didapatkan seperti pada Gambar 5.4.



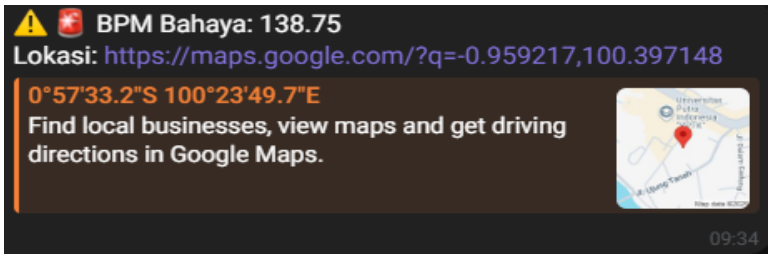
Gambar 5. 4 Alat Aktif

3. Pada sensor pertama, sensor MPU6050 akan mendeteksi jika helm terbentur atau terbalik akan mengaktifkan *Buzzer*. Pada saat helm terbalik selama 10 detik, Telegram akan mengirimkan notifikasi bahwa helm terbalik beserta dengan Lokasi nya seperti pada Gambar 5.5



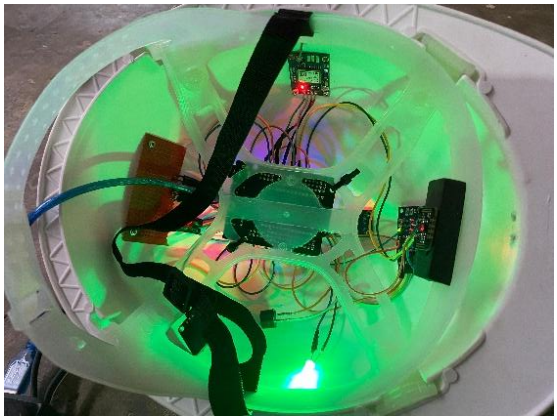
Gambar 5. 5 Notifikasi dari Sensor MPU6050 beserta Lokasi

4. Sensor MAX30102 akan mendeteksi denyut jantung pada pekerja proyek, saat denyut jantung rendah tinggi tinggi, *Vibration motor* akan hidup jika belum mencapai 10 detik. Ketika sensor mendeteksi denyut jantungnya rendah atau tinggi selama 10 detik, Telegram akan mengirimkan notifikasi bpm dan Lokasi nya seperti pada Gambar 5.6.

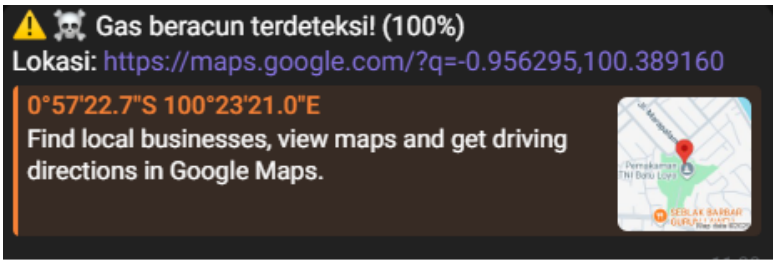


Gambar 5. 6 Notifikasi dari Sensor MAX30102 beserta Lokasi

5. Sensor MQ-135 akan mengeluarkan *output* berupa LED jika gas berbahaya pada Lokasi masih batas pada manusia seperti Gambar 5.7. Jika gas berbahaya terdeteksi bahwa nilai gas nya berbahaya pada manusia, Telegram akan mengirimkan notifikasi mengenai sensor ini beserta Lokasi nya seperti pada Gambar 5.8.



Gambar 5. 7 LED dari Sensor MQ-135 beserta Lokasi



Gambar 5. 8 Notifikasi dari Sensor MQ-135 beserta Lokasi

BAB VI

PENUTUP

6.1 Kesimpulan

Dari hasil pengujian terhadap realisasi alat, maka dapat diambil beberapa kesimpulan sebagai berikut :

1 Sistem keamanan helm pintar berbasis ESP32 yang dirancang mampu berfungsi dengan baik dan memberikan perlindungan tambahan bagi pekerja proyek.

2 Sensor MPU6050 berhasil mendeteksi gerakan tidak normal atau kejatuhan pekerja secara responsif dan akurat.

3 Sensor MAX30102 mampu memantau detak jantung dan kadar oksigen dalam darah secara real-time, sehingga dapat membantu mendeteksi kondisi kesehatan pekerja.

4 Modul GPS dapat menentukan posisi pekerja dengan cukup akurat dan mengirimkan lokasi saat terjadi kondisi darurat.

5 Sensor gas berfungsi dengan baik dalam mendeteksi keberadaan gas berbahaya di sekitar lingkungan kerja.

6 Sistem peringatan berupa *buzzer*, motor getar, dan LED dapat memberikan notifikasi langsung kepada pekerja saat terdeteksi kondisi bahaya.

7 Notifikasi yang dikirim melalui aplikasi Telegram kepada pengawas proyek dapat berjalan dengan lancar dan memberikan informasi yang cepat saat terjadi keadaan darurat.

8 Bahasa pemrograman C/C++ yang digunakan dalam mikrokontroler ESP32 mampu mengendalikan seluruh sistem secara stabil dan efisien.

6.2 Saran

Berdasarkan pengalaman yang diperoleh selama perancangan dan pembuatan alat ini. Beberapa kendala yang dihadapi oleh penulis, oleh karena itu penulis memberikan beberapa saran yang bermanfaat sebagai referensi bagi pengembang selanjutnya dalam menyempurnakan hasil karya untuk penelitian selanjutnya.

1 Sistem dapat dikembangkan lebih lanjut dengan pengaturan sleep mode ESP32 dan manajemen daya untuk mendukung penggunaan dalam waktu lama.

2 Penambahan sensor suhu dan kelembaban, serta perekaman data ke *cloud* sebagai *log* histori kondisi pekerja.

3 Perlu perbaikan desain fisik helm agar nyaman dipakai, tidak mengganggu aktivitas pekerja, dan tetap melindungi dengan standar keamanan industri.

4 Disarankan untuk mengembangkan aplikasi Android/iOS sebagai *dashboard monitoring* kondisi *real-time* seluruh pekerja di lapangan.

5 Pengujian sistem perlu dilakukan secara langsung di lapangan dengan berbagai skenario nyata untuk melihat keandalan dan akurasinya dalam kondisi lingkungan kerja sebenarnya.

DAFTAR PUSTAKA

Al Hafiz, N. W., & Erlinda, E. (2020). Perancangan sistem penyiraman tanaman otomatis menggunakan arduino. *Jurnal Teknologi Dan Open Source*, 3(2).

Anantama, A., Apriyantina, A., Samsugi, S., & Rossi, F. (2020). Alat pantau jumlah pemakaian daya listrik pada alat elektronik berbasis Arduino UNO. *Jurnal Teknologi dan Sistem Tertanam*, 1(1).

Arifin, A., Harlina, S., & Bahtiar, A. (2024). Rancang Bangun Jam Weker Portable Untuk Tunarungu Tanpa Gelombang Elektromagnetik Berbasis Arduino Nano. *Jurnal Minfo Polgan*, 13(2).

Hakim, A. R., Suryoprayogo, H., & Yansyah, I. R. (2023). Perancangan Sistem Smart Lamp berbasis Internet of Things Menggunakan Ubidots. *Journal of Informatics and Communication Technology* (JICT), 5(1).

Hartati, Y., Ipriadi, I., & Wijaya, A. H. (2023). Perancangan Sistem Informasi E-Learning pada SMA NEGERI 1 TIGO NAGARI Dengan Menggunakan Bahasa Pemrograman PHP Dan Database Mysql. *Jurnal Sistem Informasi Dan Informatika*, 1(2).

Irawan, F., Pangkal Perjuangan, J., & Pass, B. K. (2023). Perancangan Sistem Alat Sterilisasi Pintu Keluar Ruang Isolasi Berbasis Arduino Di Rs Hastien. Scientifica: *Jurnal Ilmiah Sains Dan Teknologi*, 1.

Kadir, A. (2018). ARDUINO MEGA Panduan Untuk Mempelajari Pembuatan Berbagai Proyek Elektronika.

Lestari, F. A., & Cahyono, B. D. (2022). Sistem Pengendali Mesin Solar Cells Automatic Tabber Stringer pada Penyolderan String di PT. Indonesia Solar Global. INSOLOGI: *Jurnal Sains dan Teknologi*, 1(5).

Muliadi, M., Imran, A., & Rasul, M. (2020). Pengembangan tempat sampah pintar menggunakan ESP32. *Jurnal Media Elektrik*, 17(2).

Natsyaqov, M. K., Sularsa, A., & Handayani, R. (2020). Pembangunan Perangkat Pelacak Truk Pengangkut Limbah Tinja Dengan Modul GPS. *Eproceedings Of Applied Science*, 6(1).

Nisa, J., Anwar, R., & Yuanita. (2022). Evaluasi Tingkat Kepatuhan Penerapan Instruksi Penggunaan Apd Pada Karyawan Panen Di Perkebunan Kelapa Sawit. *Jurnal Agriment*, 7(1).

Nur'aidha, A. C., & Kumarajati, D. Y. H. (2024). Sistem Deteksi Detak Jantung Berbasis Sensor Max30102, Arduino Uno, dan Oled Display Untuk Pemantauan Detak Jantung Secara Real-Time. *Jurnal Informatika dan Teknik Elektro Terapan.*

Oktodinata, W., & Purnomo, Y. (2024). Perangkat Jam Portabel dengan Fungsi Pembaca Suhu dan Pelacakan Suara Melalui *Buzzer* Menggunakan Modul Nrf Berbasis Arduino. *Jurnal Teknik Indonesia*, 3(3).

Rosa, A. A., Simon, B. A., & Lieanto, K. S. (2020). Sistem Pendeteksi Pencemaran Udara Portabel Menggunakan Sensor MQ-7 dan MQ-135. *Ultima Computing: Jurnal Sistem Komputer,* 12(1), 23-28.

Sasmoko, D. (2021). Arduino dan sensor pada project Arduino DIY. *Penerbit Yayasan Prima Agus Teknik.*

Simatupang, J. W., Santoso, F. H., Afristanto, S. D., Bramasto, R., & Maheli, H. B. (2022). Lampu LED sebagai pilihan yang lebih efisien untuk lampu utama sepeda motor.

Sulistyorini, T., Sofi, N., & Sova, E. (2024). Pemanfaatan Telegram Untuk Pendeteksi Kebocoran Gas Berbasis Nodemcu. *Jurnal Ilmiah Teknik*, 3(2).

Susilo, H., Sudargo, S., & Menarianti, I. (2021). APLIKASI PENGENALAN AKSARA JAWA "HANACARAKA" BERBASIS AUGMENTED REALITY. JIPETIK: *Jurnal Ilmiah Penelitian Teknologi Informasi & Komputer*, 2(2).

Trisetio, F. A., Suryani, V., & Yasirandi, R. (2021). Implementasi Solenoid Dan Sensor Getar Pada Sistem Keamanan Sepeda Menggunakan Modul Bluetooth Dan Gsm Berbasis Mikrokontroler. *eProceedings of Engineering*, 8(2).

Widiyanto, D. (2023). Sistem Informasi Pendaftaran Mahasiswa Baru Berbasis Web Pada Politeknik Sawunggalih Aji Kutoarjo. *Jurnal Ekonomi Dan Teknik Informatika*, 11(2).

Ziliwu, C., Sitanggang, R., Ginting, R. U., & Sibero, A. F. (2021). Rancang Bangun Sistem Informasi Penjualan Produk Handmade Berbasis Web. *Jurnal Mahajana Informasi*, 6(1).

LAMPIRAN

Program ESP32

```
#include <Wire.h>
#include <WiFi.h>
#include <HTTPClient.h>
#include "MAX30105.h"
#include "heartRate.h"
#include <MPU6050_light.h>
#include <TinyGPS++.h>

TwoWire WireMAX = TwoWire(0); // MAX30102 on pins set later
TwoWire WireMPU = TwoWire(1); // MPU6050 on pins set later

#define SDA_MAX 18
#define SCL_MAX 19
#define SDA_MPU 21
#define SCL_MPU 22
#define MQ135_PIN 34
#define BUZZER_PIN 25
#define MOTOR_PIN 26
#define LED_PIN 2

#define GPS_RX 16
#define GPS_TX 17

const char* ssid = "FrhnKnndy_";
const char* password = "knndy21019";
const char*botToken =
"7548099378:AAGC0IuwsMdVtvuLwcofCGu2_CoDEwpEKrI";
const char* chatID = "1199486758";

const int bpmDangerLow = 50;
const int bpmDangerHigh = 130;
const int gasWarning = 1300;
const int gasDanger = 1500;
const unsigned long TELEGRAM_COOLDOWN_MS = 15000; // 15s
const unsigned long I2C_CHECK_INTERVAL = 2000;
const long IR_DETECT_THRESHOLD = 5000;
const unsigned long BPM_STABILIZE_MS = 10000;
const unsigned long BPM_DANGER_DURATION = 5000;
const unsigned long MPU_DANGER_DURATION = 10000;
const unsigned long GAS_DANGER_DURATION = 10000;
const unsigned long MOTOR_PULSE_MS = 1000;
```

```
const float REBESAH_THRESHOLD_DEG = 70.0;

MAX30105 particleSensor;
MPU6050 mpu(WireMPU);
TinyGPSPlus gps;
HardwareSerial gpsSerial(1);

bool gpsReady = false;
bool koordinatPernahDidapat = false;
float gpsLat = 0, gpsLng = 0;

float beatsPerMinute = 0, beatAvg = 0;
long lastBeat = 0;
bool adaJari = false;
bool bpmBahaya = false, bpmKirim = false;
bool bpmStabil = false;
bool pernahAdaBPM = false;
unsigned long startDeteksiKulit = 0;
unsigned long startBpmBahaya = 0;

bool mpuBahaya = false, mpuKirim = false;
unsigned long startMpuBahaya = 0;
float prevAx = 0, prevAy = 0, prevAz = 0;
unsigned long lastMpuRead = 0;

bool gasBahaya = false, gasKirim = false;
unsigned long startGasBahaya = 0;
unsigned long lastGasRead = 0;
int mqBaseline = 0;

unsigned long motorOnTime = 0;
bool motorAktif = false;

unsigned long lastI2cCheck = 0;
unsigned long lastTelegramTime = 0;

bool maxReady = false;
bool mpuReady = false;

unsigned long motorToggleInterval = 400;
unsigned long motorLastToggle = 0;
bool motorState = false;

void setup() {
  Serial.begin(115200);
```

```
delay(50);

pinMode(BUZZER_PIN, OUTPUT);
pinMode(MOTOR_PIN, OUTPUT);
pinMode(LED_PIN, OUTPUT);
matikanSemua();

Serial.println(F("=== STARTUP ==="));

Serial.printf(" 🪫 Menghubungkan ke WiFi: %s\n", ssid);
WiFi.begin(ssid, password);
unsigned long wifiStart = millis();
while (WiFi.status() != WL_CONNECTED && millis() - wifiStart < 10000) {
  delay(300); Serial.print(".");
}
if (WiFi.status() == WL_CONNECTED) Serial.println("\n✅ WiFi Terhubung");
else Serial.println("\n⚠️ WiFi gagal terhubung");

gpsSerial.begin(9600, SERIAL_8N1, GPS_RX, GPS_TX);
Serial.println(" 🔎 Inisialisasi GPS NEO-7M...");

WireMAX.begin(SDA_MAX, SCL_MAX, 100000);
WireMPU.begin(SDA_MPU, SCL_MPU, 100000);

Serial.println(" 🔎 Inisialisasi MAX30102...");
if (particleSensor.begin(WireMAX)) {
  particleSensor.setup();
  particleSensor.setPulseAmplitudeRed(0x3F);
  particleSensor.setPulseAmplitudeGreen(0);
  maxReady = true;
  Serial.println(" ✅ MAX30102 siap.");
} else {
  Serial.println(" ❌ MAX30102 tidak terdeteksi!");
  maxReady = false;
}

Serial.println(" 🔎 Inisialisasi MPU6050...");
if (mpu.begin() == 0) {
  mpu.calcOffsets();
  mpuReady = true;
  Serial.println(" ✅ MPU6050 siap.");
} else {
  Serial.println(" ❌ MPU6050 gagal inisialisasi!");
```

```
    mpuReady = false;
  }

  Serial.println("🪫 Kalibrasi MQ135 baseline...");
  long total = 0;
  const int samples = 100;
  for (int i = 0; i < samples; ++i) {
    total += analogRead(MQ135_PIN);
    delay(10);
  }
  mqBaseline = total / samples;
  Serial.printf("🪫 MQ135 baseline = %d\n", mqBaseline);

  Serial.println(F("=== SETUP COMPLETE ==="));
}

void loop() {
  unsigned long now = millis();

  cekGPS();

  if (!koordinatPernahDidapat) {
    if (gpsReady) {
      koordinatPernahDidapat = true;
      digitalWrite(BUZZER_PIN, HIGH);
      delay(200);
      digitalWrite(BUZZER_PIN, LOW);
      Serial.printf("✅ GPS fix pertama: %.6f, %.6f\n", gpsLat, gpsLng);
    } else {
      Serial.println("⏳ Menunggu GPS fix...");
      delay(500);
      return;
    }
  }

  if (now - lastI2cCheck >= I2C_CHECK_INTERVAL) {
    lastI2cCheck = now;
    cekKoneksiPerangkat();
  }
  if (maxReady) cekBPM();
  if (mpuReady && now - lastMpuRead >= 200) {
    lastMpuRead = now;
    cekMPU();
  }
```

```
  if (now - lastGasRead >= 1000) {
    lastGasRead = now;
    cekGas();
  }
  kontrolMotor();
  delay(10);
}

void cekGPS() {
  while (gpsSerial.available() > 0) {
    gps.encode(gpsSerial.read());
  }
  if (gps.location.isValid()) {
    gpsLat = gps.location.lat();
    gpsLng = gps.location.lng();
    gpsReady = true;
    Serial.printf("[GPS] Koordinat: %.6f, %.6f\n", gpsLat, gpsLng);
  } else {
    gpsReady = false;
    Serial.println("[GPS] Belum mendapatkan fix...");
  }
}

bool i2cDevicePresent(TwoWire &bus, uint8_t address) {
  bus.beginTransmission(address);
  uint8_t e = bus.endTransmission();
  return (e == 0);
}

void cekKoneksiPerangkat() {
  if (!i2cDevicePresent(WireMAX, 0x57)) {
    if (particleSensor.begin(WireMAX)) {
      particleSensor.setup();
      particleSensor.setPulseAmplitudeRed(0x3F);
      particleSensor.setPulseAmplitudeGreen(0);
      maxReady = true;
    } else maxReady = false;
  }
  if (!i2cDevicePresent(WireMPU, 0x68)) {
    if (mpu.begin() == 0) {
      mpu.calcOffsets();
      mpuReady = true;
    } else mpuReady = false;
  }
}
```

```
void cekBPM() {
  long irValue = particleSensor.getIR();
  adaJari = (irValue > IR_DETECT_THRESHOLD);

  if (!adaJari) {
    Serial.println("[BPM] Tidak ada jari terdeteksi.");
    if (bpmStabil && (millis() - startDeteksiKulit > 10000)) bpmStabil = false;
    return;
  }

  if (!bpmStabil) {
    if (startDeteksiKulit == 0) startDeteksiKulit = millis();
    if (millis() - startDeteksiKulit < BPM_STABILIZE_MS) {
      Serial.println("[BPM] Menstabilkan pembacaan...");
      return;
    }
    bpmStabil = true;
    Serial.println("[BPM] Stabil, mulai membaca BPM...");
  }

  if (checkForBeat(irValue)) {
    long delta = millis() - lastBeat;
    lastBeat = millis();
    beatsPerMinute = 60.0 / (delta / 1000.0);
    if (beatsPerMinute >= 20 && beatsPerMinute <= 200) {
      beatAvg = (beatAvg * 3.0 + beatsPerMinute) / 4.0;
      pernahAdaBPM = true;
    }
  }

   Serial.printf("[BPM] IR: %ld | BPM: %.2f | Avg: %.2f\n", irValue, beatsPerMinute,
beatAvg);

  unsigned long now = millis();

  if ((beatAvg >= 50 && beatAvg <= 60) || (beatAvg >= 120 && beatAvg <= 130)) {
    if (!motorAktif) {
      motorAktif = true;
      motorOnTime = now;
      digitalWrite(MOTOR_PIN, HIGH);
    }
  }

  if ((beatAvg < 50 || beatAvg > 130) && pernahAdaBPM) {
```

```
    if (!bpmBahaya) {
      bpmBahaya = true;
      startBpmBahaya = now;
      bpmKirim = false;
    }

    if (!motorAktif) {
      motorAktif = true;
      motorOnTime = now;
      digitalWrite(MOTOR_PIN, HIGH);
    }

    if (now - startBpmBahaya >= BPM_DANGER_DURATION && !bpmKirim) {
      kirimTelegramWithCooldown("🚨 BPM Bahaya: " + String(beatAvg));
      bpmKirim = true;
      Serial.println("[BPM] Telegram dikirim karena BPM bahaya!");
    }
  } else if (!((beatAvg >= 40 && beatAvg <= 55) || (beatAvg >= 125 && beatAvg <=
140))) {
    // Reset flag jika BPM normal
    bpmBahaya = false;
    bpmKirim = false;
  }

  if (motorAktif && now - motorOnTime >= MOTOR_PULSE_MS) {
    digitalWrite(MOTOR_PIN, LOW);
    motorAktif = false;
  }
}

void cekMPU() {
  mpu.update();
  float ax = mpu.getAccX();
  float ay = mpu.getAccY();
  float az = mpu.getAccZ();
  float delta = abs(ax - prevAx) + abs(ay - prevAy) + abs(az - prevAz);
  prevAx = ax; prevAy = ay; prevAz = az;

  float pitch = atan2(ax, sqrt(ay*ay + az*az)) * 180.0 / PI;
  float roll  = atan2(ay, sqrt(ax*ax + az*az)) * 180.0 / PI;

   bool miring = (abs(pitch) > REBESAH_THRESHOLD_DEG || abs(roll) >
REBESAH_THRESHOLD_DEG);
  bool benturan = (delta > 3.0);
```

```
  Serial.printf("[MPU] Accel(X: %.2f, Y: %.2f, Z: %.2f) | Pitch: %.2f | Roll: %.2f |
Miring: %d | Benturan: %d\n",
                ax, ay, az, pitch, roll, miring, benturan);

  static unsigned long lastBuzzerToggle = 0;
  unsigned long now = millis();

  if (benturan) {
      digitalWrite(BUZZER_PIN, HIGH); delay(200); digitalWrite(BUZZER_PIN,
LOW);
    if (!mpuBahaya) { startMpuBahaya = now; mpuKirim = false; mpuBahaya = true; }
    if (now - startMpuBahaya >= MPU_DANGER_DURATION && !mpuKirim) {
      kirimTelegramWithCooldown("🪖 Helm terbentur keras!");
      mpuKirim = true;
    }
  } else if (miring) {
    if (!mpuBahaya) { startMpuBahaya = now; mpuKirim = false; mpuBahaya = true; }
    if (now - startMpuBahaya >= MPU_DANGER_DURATION && !mpuKirim) {
      kirimTelegramWithCooldown("🪖 Helm miring/rebahan!");
      mpuKirim = true;
    }
    if (now - lastBuzzerToggle >= 1000) {
      lastBuzzerToggle = now;
      int currentState = digitalRead(BUZZER_PIN);
      digitalWrite(BUZZER_PIN, !currentState);
    }
  } else {
    mpuBahaya = false;
    mpuKirim = false;
    digitalWrite(BUZZER_PIN, LOW);
  }
}

void updateLedGasBlink(int gasPersen) {
  static unsigned long lastToggle = 0;
  static bool ledState = LOW;
  unsigned long now = millis();
  if (gasPersen >= 75) {
    if (now - lastToggle >= 1000) {
      lastToggle = now;
      ledState = !ledState;
      digitalWrite(LED_PIN, ledState);
    }
```

```
  } else {
    digitalWrite(LED_PIN, LOW);
    ledState = LOW;
    lastToggle = now;
  }
}

void cekGas() {
  int nilaiGas = analogRead(MQ135_PIN);
  int gasPersen = constrain(map(nilaiGas, 500, 1500, 0, 100), 0, 100);

  Serial.printf("[GAS] Nilai: %d | Persen: %d%%\n", nilaiGas, gasPersen);

  updateLedGasBlink(gasPersen);

  if (nilaiGas > gasWarning) {
    if (!gasBahaya) { startGasBahaya = millis(); gasKirim = false; gasBahaya = true; }
        if (nilaiGas > gasDanger && millis() - startGasBahaya >=
GAS_DANGER_DURATION && !gasKirim) {
     kirimTelegramWithCooldown("☠️ Gas beracun terdeteksi! (" + String(gasPersen)
+ "%)");
      gasKirim = true;
    }
  } else {
    gasBahaya = false;
    gasKirim = false;
  }
}

void aktifkanMotor() {
  if (!motorAktif) {
    motorAktif = true;
    motorOnTime = millis();
    digitalWrite(MOTOR_PIN, HIGH);
  }
}

void kontrolMotor() {
  if (motorAktif && millis() - motorOnTime >= MOTOR_PULSE_MS) {
    digitalWrite(MOTOR_PIN, LOW);
    motorAktif = false;
  }
}
```

```
void matikanSemua() {
  digitalWrite(LED_PIN, LOW);
  digitalWrite(BUZZER_PIN, LOW);
  digitalWrite(MOTOR_PIN, LOW);
  motorAktif = false;
}

void kirimTelegramWithCooldown(String pesan) {
  unsigned long now = millis();
  if (now - lastTelegramTime < TELEGRAM_COOLDOWN_MS) {
    return;
  }
  lastTelegramTime = now;
  String lokasi = "";
  if (gpsReady) {
     lokasi = "\nLokasi: https://maps.google.com/?q=" + String(gpsLat, 6) + "," +
String(gpsLng, 6);
  } else {
    lokasi = "\nLokasi: GPS tidak tersedia";
  }
  kirimTelegram(pesan + lokasi);
}

void kirimTelegram(String pesan) {
  if (WiFi.status() != WL_CONNECTED) return;
  HTTPClient http;
  String url = "https://api.telegram.org/bot" + String(botToken) + "/sendMessage";
  http.begin(url);
  http.addHeader("Content-Type", "application/x-www-form-urlencoded");
  String body = "chat_id=" + String(chatID) + "&text=" + " ⚠️ " + pesan;
  http.POST(body);
  http.end();
}
```