PERANCANGAN PROTOTYPE ALAT MONITOR DAN KENDALI LEVEL CAIRAN DENGAN INTERNET OF THINGS

SKRIPSI

Diajukan Sebagai Salah Satu Syarat Untuk Menyelesaikan
Pendidikan Strata Satu (S-I) Jurusan Teknik Elektro
Fakultas Teknologi Industri
Universitas Bung Hatta

Oleh :

ADI SUARDIMAN
2410017111063

PROGRAM STUDI TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS BUNG HATTA
PADANG
2026

LEMBAR PENGESAHAN

PERANCANGAN PROTOTYPE ALAT MONITOR DAN KENDALI LEVEL CAIRAN DENGAN INTERNET OF THINGS

SKRIPSI

Diajukan Sebagai Salah Satu Syarat Untuk Menyelesaikan
Pendidikan Strata Satu (S-1) Jurusan Teknik Elektro
Fakultas Teknologi Industri
Universitas Bung Hatta

Oleh :

ADI SUARDIMAN
NPM : 2410017111063

Disetujui Oleh:
Pembimbing

(Dr. Ir. Indra Nisja, M.Sc)
NIDN : 10280765018

Diketahui Oleh :

Fakultas Teknologi Industri
Dekan,

Prof. Dr. Eng. Reni Desmiarti, S.T., M.T.
NIDN : 1012097403

Prodi Teknik Elektro
Ketua,

Dr. Ir. Indra Nisja, M.Sc
NIDN : 1028076501

LEMBAR PENGUJI

PERANCANGAN PROTOTYPE ALAT MONITOR DAN KENDALI LEVEL CAIRAN DENGAN INTERNET OF THINGS

SKRIPSI

ADI SUARDIMAN
NPM : 2410017111063

Dipertahankan di Depan Penguji Skripsi
Pendidikan Strata Satu (S-1) Jurusan Teknik Elektro
Fakultas Teknologi Industri Universitas Bung Hatta
Hari : Jumat, 27 Februari 2026

No	Nama	Tanda Tangan
1.	**Dr. Ir. Indra Nisja, M.Sc** (Ketua dan Penguji)	.....
2.	**Ir. Arnita, M.T** (Penguji)	.....
3.	**Mirzazoni, S.T., M.T** (Penguji)	.....

PERNYATAAN KEASLIAN SKRIPSI

Dengan ini saya menyatakan bahwa ini sebagian maupun keseluruhan Skripsi saya dengan judul " **PERANCANGAN PROTOTYPE ALAT MONITOR DAN KENDALI LEVEL CAIRAN DENGAN INTERNET OF THINGS**" adalah benar- benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan- bahan yang tidak diizinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka. Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Padang, 7 April 2026

ADI SUARDIMAN

NPM : 2410017111063

KATA PENGANTAR

Puji syukur ke hadirat Allah SWT atas rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan skripsi berjudul **"Perancangan Prototype Alat Monitor Dan Kendali Level Cairan Dengan *Internet Of Things* "** sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik pada Jurusan Teknik Elektro, Fakultas Teknologi Industri, Universitas Bung Hatta, Padang.

Dalam penyusunan skripsi ini, penulis memperoleh bantuan, dukungan, dan bimbingan dari berbagai pihak. Untuk itu penulis menyampaikan terima kasih yang sebesar-besarnya kepada:

1. Keluarga tercinta, yang selalu memberikan doa, semangat, dan dukungan moral maupun materiil selama proses penyusunan skripsi ini.
2. Ibu Prof. Dr. Reni Desmiarti, S.T., M.T. selaku Dekan Fakultas Teknologi Industri Universitas Bung Hatta.
3. Bapak Dr. Ir. Indra Nisja, M.Sc. selaku Ketua Jurusan Teknik Elektro Universitas Bung Hatta.
4. Bapak Dr. Ir. Indra Nisja, M.Sc.. selaku Pembimbing yang telah bersedia meluangkan waktunya untuk dapat membimbing, mengkoreksi, memberi saran, dan memberikan petunjuk selama proses pembuatan proposal ini.
5. Seluruh Bapak/Ibu dosen jurusan Teknik Elektro Universitas Bung Hatta.
6. Rekan - rekan yang telah banyak membantu dalam pelaksanaan pembuatan proposal ini.

Pekanbaru, Februari 2026

Adi Suardiman

ABSTRAK

Adi Suardiman : Perancangan Prototype Alat Monitor Dan Kendali Level Cairan Dengan Internet Of Things

Penelitian ini merancang alat "Prototype Sistem Monitor dan Kontrol Pompa pada Instalasi Penampungan Cairan Berbasis Internet of Things (IoT)" untuk mengatasi keterbatasan pemantauan level secara manual yang berisiko *human error*. Sistem ini menggunakan arsitektur *Hybrid Monitoring* dengan tiga antarmuka pemantauan data secara *real-time*, yaitu layar OLED, dashboard menu setting pada TFT ILI9341, dan *Web Dashboard*. Integrasi sensor dilakukan secara menyeluruh mencakup sensor JSN-SR04T untuk monitoring level cairan secara non kontak, MQ-135 untuk deteksi kualitas udara, serta sensor DHT22 dan BMP180 untuk memantau parameter lingkungan seperti suhu, kelembapan, dan tekanan udara di dalam area penampungan.

Seluruh data sensor dan aktivitas utama sistem dicatat secara otomatis ke dalam Log Data Memory Card sebagai rekam jejak operasional dan dapat di unduh diweb dashboard. Perintah kontrol pompa ditransmisikan secara nirkabel menggunakan modul nRF24L01 ke unit kontrol utama untuk menggerakkan pompa secara otomatis berdasarkan *set point* yang dapat dikalibrasi dan disimpan dalam EEPROM. Hasil pengujian menunjukkan bahwa sistem memiliki redundansi yang baik; kontrol lokal pada layar TFT tetap berfungsi penuh meskipun koneksi WiFi terputus, sistem memiliki limitasi pada waktu pemanasan (*pre-heating*) sensor MQ-135 karena untuk pembacaan sensor pertama kali memeanaskan elemen sensitivitas sensor terhadap gas, alat memiliki ketergantungan fitur jarak jauh pada stabilitas sinyal WiFi, serta potensi gangguan akurasi sensor JSN-SR04T jika posisi sensor tidak simetris di atas tanki. Secara keseluruhan, prototipe ini berhasil mensimulasikan efisiensi kontrol pompa dan akurasi monitoring berbagai parameter fisik cairan secara komprehensif.

Kata Kunci: IOT, JSN-SR04T, MQ-135, DHT22, BMP180, nRF24L01, Kontrol Pompa.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Dalam era industri modern, efisiensi dan otomatisasi menjadi kunci utama dalam menjaga produktivitas kerja. Salah satu aspek krusial dalam operasional industri maupun fasilitas umum adalah pengelolaan level cairan dalam bejana penampungan (tangki). Pengelolaan yang buruk tidak hanya berisiko menyebabkan pemborosan sumber daya, tetapi juga dapat mengakibatkan kerusakan sistem hingga potensi bahaya kerja.

Namun, pada realitanya, banyak sistem pemantauan level cairan yang masih mengandalkan metode konvensional atau manual. Beberapa kendala utama yang sering muncul antara lain:

- **Keterbatasan Aksesibilitas:** Operator harus mendatangi lokasi tangki secara fisik untuk mengecek level cairan, yang tentu tidak efisien jika lokasi tangki sulit dijangkau.
- **Risiko Human Error:** Pemantauan manual rentan terhadap kesalahan pembacaan data, yang berakibat pada keterlambatan dalam menyalakan atau mematikan pompa.
- **Kendala Saat Maintenance:** Proses perawatan (*maintenance*) seringkali terhambat karena tidak adanya sistem kendali jarak jauh yang responsif, sehingga pengosongan atau pengisian bejana memakan waktu lebih lama dan memerlukan pengawasan ketat secara terus-menerus.

Perkembangan teknologi *Internet of Things* (*IoT*) menawarkan solusi cerdas untuk mengatasi hambatan tersebut. Dengan mengintegrasikan sensor level (seperti ultrasonik atau tekanan), mikrokontroler (seperti ESP32 atau NodeMCU), dan platform berbasis web, data ketinggian cairan dapat dikirimkan secara real-time ke perangkat pintar milik operator.

Sistem ini tidak hanya berfungsi sebagai alat pantau, tetapi juga sebagai alat kendali otomatis yang mampu menggerakkan pompa berdasarkan set-point yang

ditentukan. Dengan adanya prototipe sistem monitor dan kendali berbasis IoT ini, diharapkan proses pemantauan menjadi lebih akurat, beban kerja operator selama proses ***maintenance*** berkurang, dan risiko kegagalan sistem akibat *overflow* atau *dry-running* pada pompa dapat diminimalisir secara signifikan.

Keterbatasan sistem manual sering kali menyebabkan gangguan operasional akibat keterlambatan pengisian serta meningkatnya risiko keselamatan kerja, terutama pada area berbahaya dengan akses terbatas. Oleh karena itu, diperlukan sistem monitor level cairan yang cepat, akurat, dan dapat diakses jarak jauh secara *real-time* untuk meningkatkan efisiensi serta keandalan proses operasional (Reinandie & Pramudita, 2026).

Informasi ketinggian cairan merupakan parameter krusial dalam berbagai instalasi industri maupun domestik sebagai dasar pengambilan keputusan. Namun, metode pengecekan langsung di lokasi masih dominan, yang selain tidak efisien, juga berisiko menimbulkan keterlambatan penanganan kondisi kritis seperti luapan atau kekurangan cairan. Selain itu, diperlukan presisi data terkait waktu dan laju aliran untuk kebutuhan analisis serta penyajian grafik hasil pengujian sistem (Reinandie & Pramudita, 2026; Az-Zikri et al., 2026).

Teknologi *Internet of Things* (*Internet of Things*) hadir sebagai solusi melalui integrasi sensor, mikrokontroler ESP32, dan jaringan internet untuk memantau level cairan secara otomatis. Dengan memanfaatkan sensor jarak sebagai pendeteksi permukaan, data dapat dikirim secara *real-time* ke platform web atau *smartphone*. Pendekatan ini memungkinkan pengguna memantau kondisi cairan tanpa harus hadir secara fisik di lokasi, sekaligus memperoleh peringatan dini saat cairan mencapai batas tertentu.

Berdasarkan kondisi tersebut, diperlukan rancang bangun Prototype Sistem Monitor Level Liquid berbasis IoT sebagai solusi alternatif sistem manual. Sistem ini dikembangkan untuk meningkatkan efektivitas pemantauan secara real-time, meminimalkan risiko kerugian akibat keterlambatan informasi, serta menciptakan sistem kontrol yang lebih modern, praktis, dan responsif bagi operator.

1.2 Rumusan Masalah

Berdasarkan latar belakang tersebut, maka rumusan masalah dalam penelitian ini adalah

- Bagaimana merancang dan membangun sebuah alat / *Prototype* sistem monitor serta kendali *level* cairan berbasis *Internet of Things* (*Internet of Things*) sehingga mampu menyediakan data secara *real-time*, akurat, membuat alat yang dapat mempermudah operator terkait pemantauan level Cairan dan pengontrolan pompa di bejana penampungan selama peroses maintenance.

1.3 Batasan Masalah

Agar pembahasan penelitian lebih terarah dan sesuai dengan tujuan yang ingin dicapai, maka penelitian ini dibatasi pada beberapa ruang lingkup berikut:

- Sistem yang dirancang berupa Prototype skala simulasi, bukan sistem permanen pada instalasi industri berskala besar.
- Sensor yang digunakan untuk pengukuran level cairan dibatasi pada sensor ultrasonik, sehingga tidak membahas jenis sensor level lain seperti pressure sensor, float switch, atau radar level.
- Objek pengujian dibatasi pada cairan bersifat *liquid*(non-korosif dan tidak berbahaya), sehingga tidak mencakup cairan kimia atau fluida dengan viskositas tinggi.
- Mikrokontroler yang digunakan dalam penelitian ini dibatasi pada ESP32 dengan konektivitas Wi-Fi sebagai media pengiriman data.
- Sistem monitor hanya difokuskan pada:
    - pembacaan level cairan,
    - pengiriman data ke platform *Internet of Things* ,
    - serta penampilan data berupa indikator level, grafik, dan notifikasi sederhana.
- Pengujian performa sistem dibatasi pada:

  - rentang jarak tertentu sesuai spesifikasi sensor,
  - pengukuran akurasi,
  - dan respons sistem terhadap perubahan level cairan.
- Aspek keamanan perangkat keras, perlindungan lingkungan lapangan, dan integrasi dengan sistem kontrol otomatis (seperti pengendali pompa) tidak menjadi fokus utama, namun hanya dibahas pada tingkat rekomendasi pengembangan.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah:

- Merancang dan membangun prototipe sistem monitoring dan kendali level cairan berbasis Internet of Things (IoT) yang mampu menyediakan data secara *real-time*, akurat, dan dapat diakses jarak jauh guna meningkatkan efisiensi operasional.

1.5 Manfaat Penelitian

Manfaat yang dapat diperoleh setelah tujuan penelitian ini tercapai adalah sebagai berikut:

- Tersedianya Alat Monitor Otomatis: Menghasilkan Prototype yang menggantikan pemantauan manual, sehingga efisiensi kerja di area LHF (Cairan Handling Facility) meningkat secara signifikan.
- Kemudahan Kendali Jarak Jauh: Memberikan kemampuan bagi operator untuk mengontrol pompa vakum tanpa harus berada di lokasi fisik, yang berdampak pada penghematan waktu dan tenaga.
- Peningkatan Akurasi Data: Memperoleh data level cairan yang presisi secara *real-time*, sehingga risiko kesalahan manusia (*human error*) dalam proses *maintenance* dapat diminimalisir.
- Terjaminnya Keselamatan Kerja: Menciptakan lingkungan kerja yang lebih aman karena operator dapat memantau kondisi bejana dari area yang tidak berbahaya.

BAB II
TINJAUAN PUSTAKA

2.1 Tinjauan Penelitian

Penelitian terkait monitor dan pengontrolan level cairan telah banyak dilakukan pada berbagai lingkungan aplikasi, baik pada sistem industri, tangki penyimpanan, maupun fasilitas pengolahan fluida. Namun, sebagian besar penelitian sebelumnya lebih menitikberatkan pada sistem yang dipasang secara permanen (fixed installation) dan terintegrasi dengan unit proses tertentu, sehingga tidak memiliki fleksibilitas penggunaan pada berbagai lokasi. Hal ini menjadi pembeda utama dengan penelitian Rancang Bangun Sistem Portable Monitor Level Liquid Berbasis Internet of Things dengan Dukungan Modul Komunikasi NRF24L01 sebagai komunikasi antara rangkaian kontrol dan rangkaian daya, yang difokuskan pada mobilitas alat dan kemudahan di lapangan.

Penelitian yang dilakukan oleh Azriel Nuriswara Az-Zikri, Slamet Indriyanto, dan Agung Wicaksono dalam jurnal berjudul *"Perancangan Prototype Sistem Monitor Level LiquidTandon Berbasis Internet of Things (Internet of Things) Menggunakan Sensor Ultrasonik JSN-SR04T"* merancang sistem monitor level cairan menggunakan sensor ultrasonik dan modul ESP8266 yang terhubung ke platform *Internet of Things* untuk menampilkan data secara real-time. Sistem monitor pada penelitian ini dipasang secara permanen pada tandon *liquid*rumah tangga sehingga hanya berfungsi pada satu titik pengukuran. Dengan demikian, karakter sistem masih bersifat fixed-installation dan belum mengakomodasi portabilitas alat untuk kebutuhan pengukuran pada beberapa lokasi berbeda. [1]

Penelitian lainnya oleh Naufal Fakhrie Rienandie dan Resa Pramudita berjudul *"Design of an Internet of Things-Based Liquid Level Monitor System"* mengembangkan sistem monitor level cairan berbasis ESP32 yang dilengkapi fungsi pengontrolan pompa otomatis. Sistem ini efektif diterapkan pada instalasi

reservoir tetap dan terintegrasi langsung dengan sistem pompa proses. Fokus penelitian lebih diarahkan pada kestabilan operasi dan otomatisasi pengendalian fluida sehingga perangkat masih bersifat plant-based dan tidak dirancang sebagai alat monitor yang portable.

Pada penelitian Prasanth P. P., Athul Thomas, dan James Kurian yang berjudul *"Scalable and Robust Internet of Things -Based Liquid Level Monitor /Control System for Home and Industrial Automation"*, dirancang sistem monitor dan kontrol level cairan berbasis *Internet of Things* dengan komunikasi MQTT yang mendukung beberapa node sensor. Penelitian ini lebih menitikberatkan pada aspek skalabilitas jaringan, keandalan komunikasi data, dan integrasi sistem otomasi. Namun perangkat monitor tetap dipasang sebagai bagian dari instrumentasi permanen pada instalasi tertentu sehingga tidak diarahkan untuk mobilitas penggunaan di berbagai lokasi.

Selanjutnya, Ali Sahat Pardomuan dan Umar Zaky dalam penelitian berjudul *"Automated* Cairan *Level Control System Using IOT Under Diverse Conditions"* mengembangkan sistem monitor dan kontrol level cairan berbasis sensor ultrasonik HY-SRF05 dengan mekanisme pengaturan katup otomatis. Tujuan utama penelitian ini adalah menjaga kestabilan level cairan pada tangki penampungan melalui respon aktuator. Walaupun sistem memiliki akurasi pengukuran yang baik, perangkat tetap dipasang secara permanen pada tangki yang sama sehingga belum mendukung konsep alat monitor portable.

Dalam penelitian *"Internet of Things -Based* Cairan *Level Monitor System of Situ Rawa Besar"* oleh Erna Kusuma Wati, Hari Hadi Santoso, dan Aryo Laksono, sistem monitor dirancang untuk memantau ketinggian muka *liquid*pada area Situ Rawa Besar dengan antarmuka web berbasis *Internet of Things* . Sistem ini mampu memberikan informasi kondisi *liquid*secara real-time sebagai pendukung pengawasan lingkungan. Namun perangkat monitor dipasang secara tetap pada

lokasi tertentu sehingga bersifat site-specific dan tidak ditujukan untuk penggunaan mobile di area pengukuran lain.

Penelitian oleh Annisa Putri Hasanah, Muhammad Irfan Sarif, dan Hafni berjudul *"Perancangan Sistem Monitor Level LiquidMenggunakan Sensor Ultrasonik Berbasis Internet of Things dengan Aplikasi Dashboard"* juga mengembangkan monitor level cairan berbasis *Internet of Things* dengan fitur notifikasi dan kontrol pompa melalui aplikasi Dashboard. Sistem ini memiliki kemampuan pemantauan jarak jauh, namun instalasi perangkat tetap difokuskan pada satu unit tangki. Oleh karena itu, sistem masih bersifat non-portable dan tidak dirancang untuk kebutuhan inspeksi lapangan pada beberapa titik pengukuran. [

Secara umum, penelitian-penelitian sebelumnya menunjukkan bahwa pengembangan sistem monitor level cairan berbasis *Internet of Things* masih berfokus pada instalasi tetap, integrasi proses, serta fungsi kontrol otomatis, sehingga belum banyak yang mengembangkan perangkat monitor level liquid yang bersifat portable, fleksibel dipindahkan, dan ditujukan untuk kebutuhan inspeksi lapangan sebagaimana yang dikembangkan dalam penelitian ini.

2.2 Landasan Teori

Bagian ini membahas teori dan konsep dasar yang digunakan untuk mendukung penelitian mengenai sistem kendali level cairan. Dasar-dasar teori yang akan dijelaskan meliputi prinsip sistem kendali terbuka dan tertutup serta perangkat pendukung operasional alat lainnya

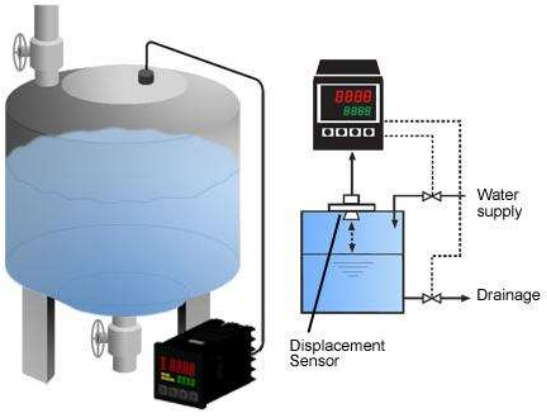
2.2.1 Liquid Level Control

Level Control merupakan perangkat elektronik yang berfungsi untuk mendeteksi, memonitor, dan mengendalikan ketinggian cairan di dalam suatu tangki atau wadah secara otomatis. Perangkat ini sering disebut sebagai *floatless relay*, *liquid level relay*, atau *automatic liquid tank level controller*.

Secara teknis, sistem pengendalian ini dapat dibagi menjadi dua metode utama, yaitu:

1. **Sistem Kendali Terbuka (*Open Loop*):** Pada sistem ini, pengendalian dilakukan tanpa adanya umpan balik dari kondisi nyata di dalam tangki. Misalnya, pompa hanya diatur bekerja berdasarkan durasi waktu tertentu. Kelemahannya, sistem tidak dapat mengantisipasi jika terjadi luapan atau kekosongan mendadak karena tidak ada sensor yang memberikan informasi balik ke pengontrol.
2. **Sistem Kendali Tertutup (*Close Loop*):** Sistem ini menggunakan sensor sebagai umpan balik (*feedback*) untuk memantau perubahan level cairan secara *real-time*. Data dari sensor diteruskan ke unit kontrol (seperti mikrokontroler atau relay) yang selanjutnya mengatur aktuator, seperti pompa atau katup, sesuai dengan kondisi level cairan yang terdeteksi. Sistem ini jauh lebih akurat karena unit kontrol selalu menyesuaikan kinerjanya berdasarkan input sensor.

Penggunaan Level Control dengan metode *Close Loop* bertujuan untuk menjaga level cairan tetap berada pada batas operasi yang diizinkan, mencegah terjadinya meluap (*overflow*), menghindari kekosongan tangki (*dry condition*), serta meningkatkan efisiensi operasi sistem distribusi cairan, baik pada lingkungan industri maupun domestik.



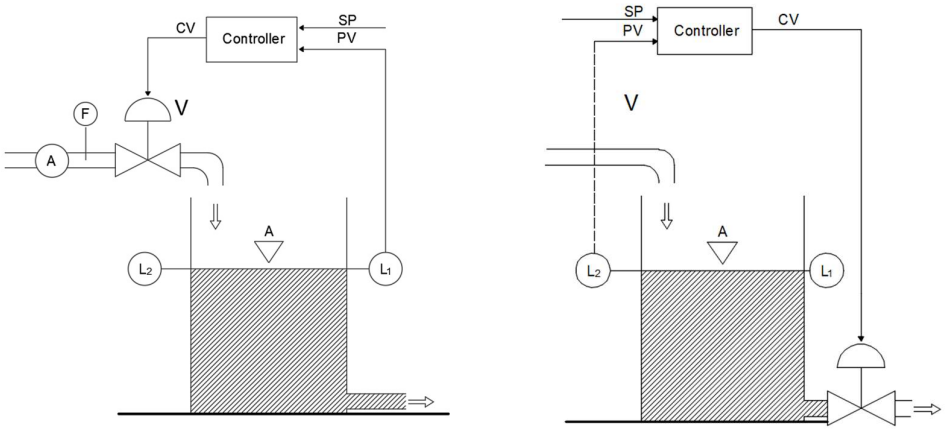
Gambar 2. 1 Diagram skematik sistem pengendalian tingkat cairan

A. Prinsip Kerja Cairan Level Control

Prinsip kerja Cairan Level Control didasarkan pada sifat *liquid*sebagai penghantar listrik. Sensor akan memberikan respon ketika permukaan *liquid*menyentuh range sensor tertentu sehingga sensor membaca jarak. Nilai resistansi antara dua elektroda yang terendam akan dibandingkan dengan nilai ambang (*threshold*) pada rangkaian kontrol.

Apabila level *liquid*berada pada batas tertentu, sinyal listrik akan diteruskan ke relay untuk menentukan kondisi pengendalian, misalnya:

1. Fungsi Pengosongan (Discharge Control)
   Pompa akan aktif ketika *liquid*mencapai batas maksimum untuk menurunkan level cairan hingga berada di bawah batas minimum.
2. Fungsi Pengisian (Filling Control)
   Pompa akan menyala ketika *liquid*turun di bawah batas minimum dan berhenti ketika level mencapai posisi maksimum.



Filling Control Discharge Control

Gambar 2. 2 Prinsip Kerja Cairan Level Control

Dengan mekanisme ini, proses pengisian dan pengosongan tangki dapat berjalan secara otomatis, stabil, dan aman.

B. Komponen Utama Cairan Level Control

Secara umum, sistem Cairan Level Control terdiri dari beberapa komponen utama, yaitu:

1. Unit Kontrol / Relay Level

   Berfungsi sebagai pengolah sinyal dari sensor dan pengendali aktuator.
2. Sensor Elektroda / Probe Level

   Dipasang pada titik batas minimum, tengah, dan maksimum sesuai kebutuhan pengukuran.
3. Pompa atau Katup (Actuator)

   Berfungsi untuk mengisi atau mengosongkan cairan pada tangki.
4. Indikator Visual / Alarm

   Dapat berupa LED atau buzzer sebagai penanda kondisi level cairan.

C. Jenis–Jenis Cairan Level Control

Berdasarkan fungsi pengendaliannya, Cairan Level Control dapat diklasifikasikan menjadi:

1. Relay Kontrol Level Minimum

   Digunakan untuk mendeteksi kondisi cairan berada di bawah batas minimum.
2. Relay Kontrol Level Maksimum

   Digunakan untuk mendeteksi kondisi cairan yang melebihi batas maksimum.
3. Relay Kombinasi Minimum–Maksimum

   Mampu melakukan fungsi pengisian sekaligus pengosongan secara otomatis.

Pemilihan jenis relay disesuaikan dengan kebutuhan proses dan karakteristik sistem fluida.

D. Aplikasi Cairan Level Control

Cairan Level Control banyak digunakan pada berbagai bidang, di antaranya:

- Sistem penyimpanan dan distribusi *liquid*bersih
- Instalasi pengolahan *liquid*limbah
- Tangki penyimpanan cairan industri
- Sistem irigasi dan pertanian

- Industri makanan dan minuman
- Sistem boiler dan suplai *liquid*panas

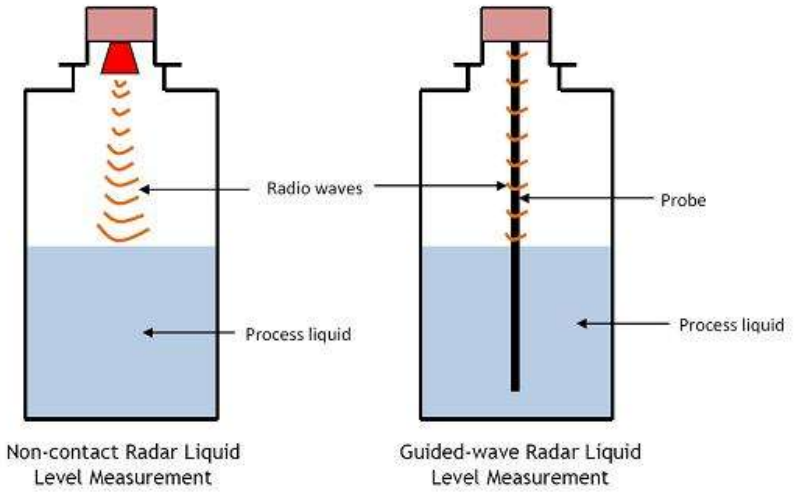
Penerapan sistem kontrol level cairan memberikan manfaat berupa peningkatan keselamatan operasi, efisiensi energi, pengurangan risiko kerusakan peralatan, serta meminimalkan intervensi manual operator.

2.2.2 Sensor untuk Cairan Level Control

Sensor level adalah komponen utama yang digunakan untuk mendeteksi perubahan ketinggian cairan di dalam tangki. Pemilihan jenis sensor harus disesuaikan dengan:

- jenis fluida,
- kondisi lingkungan,
- kebutuhan akurasi,
- dan karakteristik sistem.

Jenis sensor yang digunakan sangat bergantung pada karakteristik fluida, kondisi lingkungan, serta kebutuhan akurasi sistem. Secara umum, sensor level dapat dikategorikan menjadi dua kelompok, yaitu sensor kontak (contact type) dan sensor tanpa kontak (non-contact type).



Gambar 2. 3 Jenis sensor non kontak dan kontak untuk level liquid

Sensor kontak bekerja dengan cara bersentuhan langsung dengan cairan, seperti float switch dan elektrode konduktivitas. Float switch banyak digunakan pada aplikasi sederhana karena memiliki mekanisme kerja mekanik yang mudah, biaya rendah, namun tidak mendukung pengukuran kontinu dan kurang sesuai untuk lingkungan yang korosif atau mengandung bahan kimia.

Sebaliknya, sensor non-kontak seperti ultrasonic level sensor bekerja dengan memanfaatkan gelombang ultrasonik yang dipantulkan dari permukaan cairan. Sensor jenis ini memiliki keunggulan karena tidak bersentuhan langsung dengan cairan, mendukung pembacaan jarak secara kontinu, serta lebih aman digunakan pada media yang berpotensi berbahaya. Pada pengukuran level berbasis mikrokontroler, data hasil pengukuran jarak sensor akan dikonversi menjadi nilai ketinggian cairan melalui proses komputasi.

Sensor Ultrasonic JSN-SR04T merupakan sensor jarak yang bekerja berdasarkan prinsip pemantulan gelombang ultrasonik (*ultrasonic ranging*). Sensor ini banyak digunakan pada sistem monitor ketinggian cairan, robotika, serta aplikasi pengukuran jarak berbasis mikrokontroler karena memiliki tingkat akurasi yang cukup baik, konsumsi daya rendah, dan mudah diintegrasikan dengan sistem elektronik digital.

Sensor ini terdiri dari dua komponen utama, yaitu transmitter ultrasonik yang berfungsi memancarkan gelombang ultrasonik, dan receiver ultrasonik yang berfungsi menerima gelombang pantul dari objek atau permukaan cairan. JSN-SR04T bekerja pada frekuensi sekitar 40 kHz, sehingga aman digunakan dan tidak terdengar oleh manusia.

2.2.3 Sensor Ultrasonic JSN-SR04T

Proses kerja sensor JSN-SR04T diawali dengan pengiriman sinyal pemicu (*trigger*) dari mikrokontroler. Setelah menerima sinyal tersebut, sensor akan memancarkan gelombang ultrasonik selama durasi tertentu. Gelombang tersebut bergerak melalui udara dan akan dipantulkan kembali ketika mengenai permukaan objek atau permukaan cairan.

Gambar 3. 1 Sensor Ultrasonic JSN-SR04T

Sinyal pantulan kemudian diterima oleh bagian echo receiver, dan sensor akan menghasilkan pulsa keluaran dengan lebar pulsa yang sebanding dengan waktu tempuh gelombang ultrasonik.

Jarak objek dapat dihitung menggunakan persamaan:

$$\text{Jarak} = \frac{v \times t}{2}$$

dengan keterangan:

- v = kecepatan rambat suara di udara (± 340 m/s),
- t = waktu tempuh gelombang (pulsa echo),
- faktor pembagi 2 digunakan karena gelombang menempuh jarak pergi dan kembali.

Pada sistem monitor level cairan, nilai jarak yang diperoleh akan dikonversi menjadi ketinggian cairan berdasarkan selisih jarak sensor terhadap permukaan tangki.

B. Spesifikasi Umum Sensor JSN-SR04T

Beberapa karakteristik teknis sensor JSN-SR04T antara lain:

- Tegangan kerja : 5 V DC
- Jarak ukur : 2 cm – 400 cm
- Frekuensi kerja : 40 kHz
- Antarmuka data : sinyal digital (Trigger–Echo)
- Tipe pembacaan : pengukuran jarak non-kontak

Karena tidak bersentuhan langsung dengan cairan, sensor ini lebih aman digunakan pada aplikasi cairan kimia, limbah, maupun fluida yang bersifat korosif.

C. Kelebihan dan Keterbatasan Sensor JSN-SR04T

Kelebihan:

- tidak bersentuhan langsung dengan media cairan,
- mendukung pembacaan jarak secara kontinu,
- kompatibel dengan mikrokontroler seperti ESP32, Arduino, dan STM32.

Keterbatasan:

- hasil pembacaan dipengaruhi uap, busa, atau permukaan cairan bergejolak,
- akurasi menurun pada suhu lingkungan ekstrem,

2.2.4 Sensor Suhu dan Kelembapan (DHT22)

Sensor DHT22 merupakan sensor digital yang digunakan untuk mengukur suhu dan kelembapan udara secara bersamaan. Sensor ini memiliki elemen pengukur kelembapan berbasis kapasitor polimer dan termistor untuk pengukuran suhu. Data hasil pengukuran dikirimkan dalam bentuk sinyal digital ke mikrokontroler.

Gambar 3. 2 Sensor Suhu dan Kelembapan (DHT22)

Proses kerja sensor DHT22 diawali dengan pengiriman sinyal permintaan (start signal) dari mikrokontroler. Setelah menerima sinyal tersebut, sensor akan melakukan pembacaan kondisi lingkungan, kemudian mengirimkan data suhu dan kelembapan dalam bentuk paket data digital melalui satu jalur komunikasi.

Nilai suhu dan kelembapan yang diperoleh selanjutnya dapat digunakan sebagai parameter pemantauan kondisi lingkungan, khususnya pada sistem monitor berbasis *Internet of Things* .

B. Spesifikasi Umum Sensor DHT22

Beberapa karakteristik teknis sensor DHT22 antara lain:

- Tegangan kerja : 3.3 – 6 V DC
- Rentang pengukuran suhu : -40 °C sampai 80 °C
- Rentang pengukuran kelembapan : 0 – 100 % RH
- Akurasi suhu : ± 0.5 °C
- Akurasi kelembapan : ± 2 – 5 % RH
- Antarmuka data : Digital (single wire)
- Periode pembacaan : sekitar 2 detik

C. Kelebihan dan Keterbatasan Sensor DHT22

Kelebihan:

- Mengukur suhu dan kelembapan dalam satu modul
- Akurasi lebih baik dibanding DHT11
- Komunikasi digital sehingga mudah diintegrasikan dengan mikrokontroler seperti ESP32 dan Arduino

Keterbatasan:

- Waktu respon relatif lebih lambat
- Tidak cocok untuk lingkungan dengan perubahan suhu yang sangat cepat
- Sensitif terhadap kondensasi air

2.2.5 Sensor Kualitas Udara/Gas (MQ-135)

Sensor MQ-135 digunakan untuk mendeteksi kualitas udara dengan mengukur konsentrasi gas tertentu di lingkungan, seperti amonia (NH_3), nitrogen oksida (NOx), alkohol, benzena, asap, dan karbon dioksida (CO_2) dalam kisaran tertentu.

Prinsip kerja sensor MQ-135 berdasarkan perubahan resistansi material semikonduktor (SnO_2) akibat adanya gas di udara. Ketika konsentrasi gas

meningkat, resistansi sensor akan berubah, kemudian perubahan tersebut dikonversi menjadi sinyal analog yang dapat dibaca oleh mikrokontroler.

Gambar 3. 3 Sensor Kualitas Udara/Gas (MQ-135)

Dalam sistem monitor , data dari sensor MQ-135 dapat digunakan untuk mengetahui tingkat pencemaran udara di sekitar area tangki atau lingkungan industri.

B. Spesifikasi Umum Sensor MQ-135

Beberapa karakteristik teknis sensor MQ-135 antara lain:

- Tegangan kerja : 5 V DC
- Tipe output : Analog dan digital
- Sensitivitas gas : NH_3, NOx, alkohol, benzena, asap, CO_2
- Rentang deteksi : sekitar 10 – 1000 (tergantung jenis gas)
- Waktu pemanasan awal : ± 20 detik (stabilisasi optimal beberapa menit)

Penting untuk meluruskan bahwa sensor MQ-135 sebenarnya membaca kualitas udara secara umum, namun angka "mentah" yang dihasilkan oleh analogRead di Arduino bukanlah nilai absolut kecuali Anda sudah melakukan kalibrasi menggunakan R (resistansi udara bersih).

Berikut adalah panduan batas normal kualitas udara dalam satuan (Parts Per Million) yang umumnya dirujuk untuk sensor MQ-135:

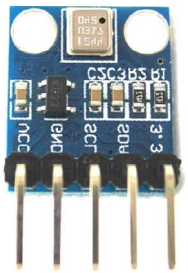
Tabel 2 1 Tabel Kualitas Udara (Skala)

Nilai	Status	Deskripsi
< 400	Sangat Baik	Udara bersih luar ruangan (normalnya sekitar 350-450 CO2).

400 - 1.000	Baik	Standar udara dalam ruangan dengan sirkulasi yang cukup.
1.000 - 2.000	Waspada	Udara mulai terasa pengap, bisa menyebabkan kantuk atau kurang fokus.
2.000 - 5.000	Buruk	Konsentrasi gas (CO2/Amonia) tinggi. Bisa menyebabkan sakit kepala dan mual.
> 5.000	Bahaya	Batas paparan kerja (OSHA). Udara beracun jika dihirup dalam waktu lama.

2.2.6 Sensor Tekanan Udara (BMP180)

Sensor BMP180 merupakan sensor digital yang digunakan untuk mengukur tekanan udara dan suhu. Sensor ini sering digunakan untuk aplikasi pengukuran ketinggian, cuaca, serta monitor kondisi lingkungan.



Gambar 3. 4 Sensor Tekanan Udara (BMP180)

Prinsip kerja BMP180 didasarkan pada perubahan tekanan yang mempengaruhi elemen piezoresistif internal. Perubahan tekanan tersebut diolah oleh ADC internal dan dikirimkan ke mikrokontroler melalui komunikasi I2C.

Pada sistem monitor berbasis *Internet of Things* , data tekanan udara dapat dimanfaatkan untuk analisis kondisi lingkungan serta kompensasi pengukuran lainnya.

B. Spesifikasi Umum Sensor BMP180

Beberapa karakteristik teknis sensor BMP180 antara lain:

- Tegangan kerja : 3.3 V DC
- Rentang tekanan : 300 – 1100 hPa

- Akurasi tekanan : ± 1 hPa
- Antarmuka komunikasi : I2C
- Resolusi tinggi dengan konsumsi daya rendah

C. Kelebihan dan Keterbatasan Sensor BMP180

Kelebihan:

- Akurasi cukup tinggi untuk pengukuran tekanan udara
- Konsumsi daya rendah
- Komunikasi I2C sehingga hemat pin mikrokontroler
- Dapat digunakan untuk estimasi ketinggian

Keterbatasan:

- Sensitif terhadap perubahan suhu ekstrem
- Memerlukan proses kalibrasi internal
- Rentang tekanan terbatas pada aplikasi atmosfer

2.2.7 RF RX/TX Radio Controller Communication (Modul NRF24L01)

Radio Frequency (RF) Communication merupakan teknologi komunikasi nirkabel yang memanfaatkan gelombang radio sebagai media transmisi data antara perangkat pengirim (*transmitter*) dan penerima (*receiver*). Sistem RF umumnya digunakan pada aplikasi *remote control*, *wireless sensor network*, sistem telemetri, hingga Internet of Things (*Internet of Things*) karena mampu mentransmisikan data tanpa membutuhkan koneksi kabel dan memiliki jangkauan komunikasi yang relatif lebih luas dibandingkan komunikasi infra-merah atau *wired connection*.

Pada sistem kontrol berbasis mikrokontroler, komunikasi RF berperan sebagai media pertukaran data antara *controller unit* (pengendali) dengan *end device* seperti sensor, aktuator, atau modul monitor . Proses komunikasi RF melibatkan dua fungsi utama, yaitu proses pemancaran data (TX – Transmitter) dan proses penerimaan data (RX – Receiver). Data digital yang dikirimkan akan dimodulasi ke dalam sinyal RF, kemudian dipancarkan melalui antena dan diterima oleh modul penerima untuk dikonversi kembali menjadi data digital.

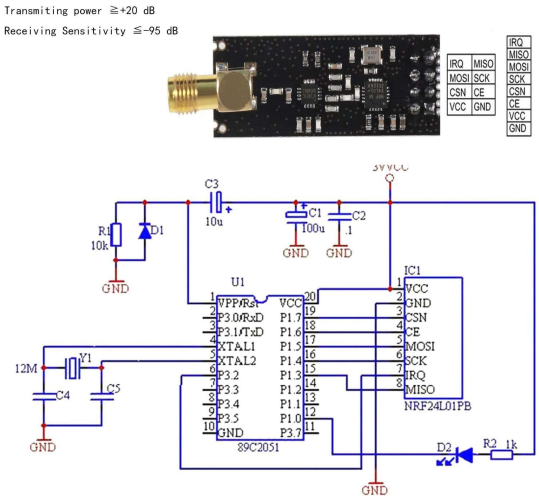
Modul NRF24L01

Modul NRF24L01 adalah perangkat *transceiver* RF yang bekerja pada pita frekuensi *Industrial, Scientific, and Medical* (ISM) 2.4 GHz secara global tanpa memerlukan lisensi. Modul ini merupakan pengembangan dari seri nRF2401A dengan penambahan fitur berupa *pipeline* ekstra, *buffer*, serta fungsi *auto-retransmit* untuk meningkatkan keandalan transmisi. Perangkat ini dirancang untuk memudahkan komunikasi nirkabel pada proyek mikrokontroler tanpa memerlukan desain sirkuit nirkabel yang rumit

Modul NRF24L01 memiliki beberapa karakteristik penting, antara lain:

- Jarak Jauh: Mencapai 1.000+ meter berkat integrasi PA dan LNA.
- Frekuensi: Beroperasi pada pita 2.4GHz ISM bebas lisensi.
- Kecepatan: Mendukung laju data 250kbps, 1Mbps, hingga 2Mbps.
- Koneksi: Menggunakan antarmuka SPI untuk kontrol data.
- Daya Rendah: Beroperasi pada tegangan 1.9V hingga 3.6V.
- Fitur: Mendukung *auto-retransmit* dan memiliki input toleran 5V

Dengan karakteristik tersebut, NRF24L01 sering digunakan pada sistem kontrol jarak jauh, monitor data sensor, serta komunikasi antar perangkat *Internet of Things* .



Gambar 2. 4 Port NRF24L01 2,4 Ghz

B. Prinsip Kerja Komunikasi RF pada NRF24L01

Proses komunikasi pada modul nRF24L01 berlangsung melalui tahapan berikut:

1. Perangkat pengirim (TX) menerima data dari mikrokontroler melalui antarmuka SPI.
2. Modul mengemas data ke dalam paket nirkabel dan memancarkannya pada frekuensi 2,4 GHz.
3. Perangkat penerima (RX) menangkap sinyal RF tersebut melalui antena dan memverifikasi integritas datanya.
4. Data yang telah diterima kemudian dikirimkan kembali ke mikrokontroler melalui jalur SPI untuk diproses lebih lanjut

Mekanisme ini memungkinkan pertukaran data berlangsung secara dua arah (bidirectional) sehingga mendukung fungsi *monitor* maupun *control* pada sistem.

C. Kelebihan Penggunaan Modul NRF24L01

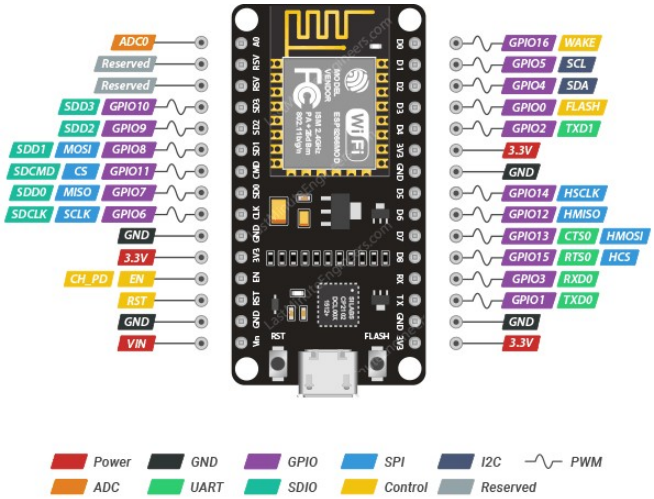
Penggunaan modul nRF24L01 dalam sistem monitor dan kontrol memberikan beberapa kelebihan, di antaranya:

- Memiliki jangkauan komunikasi yang sangat jauh, mencapai lebih dari 1.000 meter (line of sight) berkat integrasi PA dan LNA.
- Mendukung kecepatan transmisi data tinggi hingga 2Mbps pada pita frekuensi 2.4GHz yang bebas lisensi global.
- Dilengkapi fitur *auto-retransmit* dan pengecekan CRC otomatis untuk menjamin keandalan pengiriman paket data.
- Menggunakan antarmuka SPI yang kompatibel dengan berbagai jenis mikrokontroler dan mendukung integrasi sistem *Internet of Things* .

Oleh karena itu, modul nRF24L01 menjadi salah satu alternatif yang efektif dalam implementasi sistem kontrol jarak jauh berbasis RF, khususnya pada aplikasi monitor level cairan, sistem otomasi, dan perangkat *Internet of Things* yang membutuhkan transmisi data jarak jauh dengan konsumsi daya rendah.

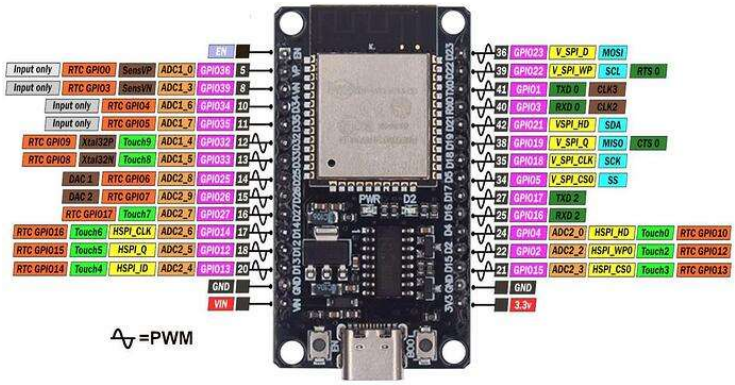
2.2.8 Esp 32 dan ESP 8266 NodeMCU

ESP32 dan ESP8266 NodeMCU merupakan mikrokontroler berbasis sistem pada chip (*System on Chip / SoC*) yang dikembangkan oleh Espressif Systems dan banyak digunakan dalam pengembangan sistem tertanam serta aplikasi Internet of Things (*Internet of Things*). Kedua perangkat ini memiliki kemampuan komunikasi nirkabel yang terintegrasi, sehingga mendukung pengembangan sistem monitor dan kontrol jarak jauh secara efisien.



Gambar 2. 5 Esp 8266 Pinout

ESP8266 NodeMCU merupakan mikrokontroler yang telah dilengkapi modul Wi-Fi IEEE 802.11 b/g/n dan antarmuka USB-to-Serial, sehingga memudahkan proses pemrograman. Mikrokontroler ini memiliki satu inti prosesor 32-bit dengan frekuensi hingga 80 MHz, jumlah GPIO yang terbatas, serta konsumsi daya yang relatif rendah. ESP8266 umumnya digunakan pada aplikasi sederhana seperti pengiriman data sensor, sistem monitor berbasis web, dan komunikasi data nirkabel jarak dekat.



Gambar 2. 6 Esp 32 pinout

ESP32 merupakan pengembangan lanjutan dari ESP8266 dengan spesifikasi yang lebih tinggi. Mikrokontroler ini memiliki prosesor dual-core 32-bit dengan frekuensi hingga 240 MHz, serta mendukung konektivitas Wi-Fi dan Bluetooth (Classic dan BLE). Selain itu, ESP32 menyediakan lebih banyak pin GPIO, fitur ADC dan DAC, serta mendukung mode hemat daya (*deep sleep*). Dengan kemampuan tersebut, ESP32 sangat sesuai digunakan pada aplikasi yang membutuhkan pemrosesan data lebih kompleks, integrasi berbagai sensor, dan sistem kontrol yang lebih stabil.

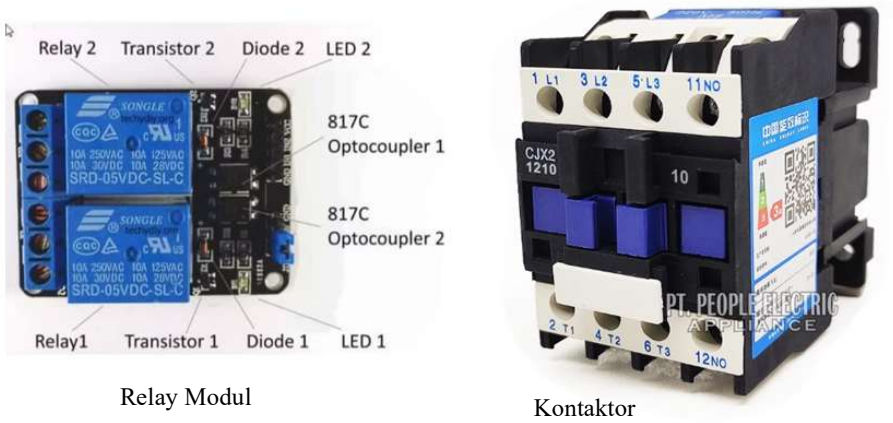
Dalam penelitian ini, penggunaan ESP32 dan ESP8266 NodeMCU dipilih karena kemampuannya dalam mendukung komunikasi data nirkabel, kemudahan pemrograman, serta kompatibilitas dengan berbagai sensor dan modul pendukung yang digunakan dalam sistem yang dirancang.

2.2.9 Penghubung / Pemutus Arus (Contactor & Relay)

Relay merupakan saklar elektromagnetik yang bekerja pada level tegangan sinyal rendah dan digunakan sebagai penghubung antara mikrokontroler dengan beban bertegangan lebih tinggi. Relay berperan sebagai elemen pengaman agar rangkaian kontrol tidak terhubung langsung dengan beban listrik.

Sementara itu, contactor digunakan pada rangkaian beban dengan arus yang lebih besar, misalnya motor pompa industri. Contactors dirancang khusus untuk

operasi switching berulang dengan kemampuan hantar arus tinggi serta dilengkapi komponen proteksi tambahan.



Relay Modul Kontaktor

Gambar 2. 7 Relay dan kontaktor

Penggunaan relay atau contactor pada sistem pengendalian level cairan memberikan beberapa keuntungan, antara lain:

1. Meningkatkan keamanan pemisahan rangkaian kontrol dan rangkaian daya,
2. Memungkinkan pengendalian beban listrik secara otomatis, dan
3. Mendukung integrasi dengan sistem *Internet of Things* maupun kontrol manual.

Dengan demikian, keberadaan pemutus arus menjadi komponen penting dalam memastikan operasi pengendalian level cairan berjalan aman, andal, dan sesuai standar kelistrikan.

BAB III
METODOLOGI PENELITIAN

3.1 Alat dan Bahan

Pada bab ini akan dijelaskan mengenai metode untuk merancang, mengimplementasikan, dan menguji Prototype alat monitor dan kendali level cairan berbasis *Internet of Things* (*Internet of Things*). Proses pembuatan sistem ini akan dibagi menjadi beberapa tahapan sistematis, antara lain: persiapan alat dan bahan penelitian, alur penelitian, perancangan perangkat keras (*hardware*), perancangan perangkat lunak (*software*), serta deskripsi kerja sistem dan analisa pengujian.

Dalam perancangan ini dibutuhkan beberapa alat dan bahan pendukung untuk merealisasikan sistem monitor dan kendali yang optimal. Berikut adalah rincian kebutuhan penelitian:

3.1.1 Alat Penelitian (*Tools*)

Alat penelitian merupakan perangkat kerja yang digunakan untuk membantu proses pembuatan, perakitan, dan pemrograman sistem, meliputi:

1. Laptop / PC: Sebagai perangkat utama untuk menulis kode program, desain skematik, dan memonitor data serial.
2. Solder dan Timah: Untuk merakit komponen elektronik pada PCB atau menyambung kabel.
3. Multimeter (Avometer): Untuk mengukur tegangan, arus, dan hambatan guna memastikan sirkuit bekerja dengan benar.
4. Obeng dan Tang Potong: Untuk instalasi mekanik dan pemotongan kabel jumper.
5. Perangkat Lunak (*Software*):
    - Arduino IDE: Sebagai *Integrated Development Environment* untuk memprogram mikrokontroler.
    - Proteous: Untuk merancang skema rangkaian elektronik dan desain PCB.

- Driver USB: Untuk komunikasi antara laptop dengan mikrokontroler.

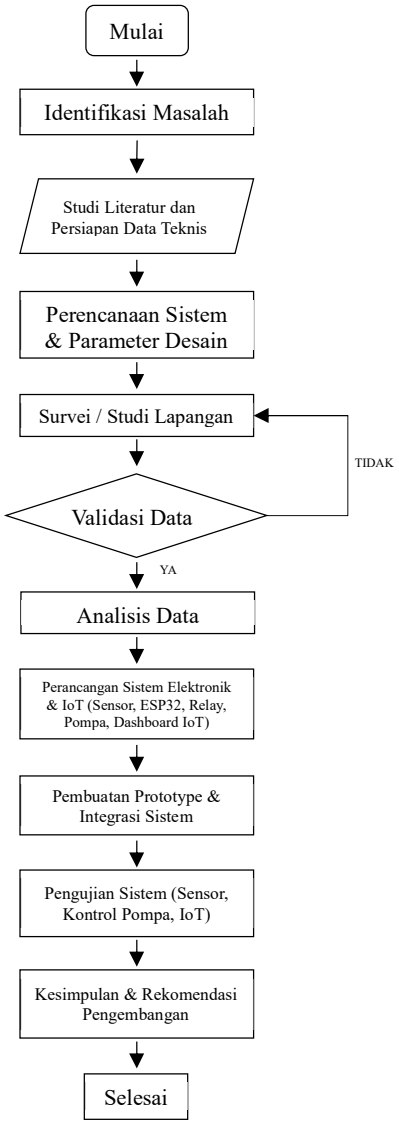
3.1.2 Bahan Penelitian (*Materials*)

Bahan penelitian adalah komponen-komponen utama dan pendukung yang disusun untuk membentuk sistem Prototype, yaitu:

1. Mikrokontroler (ESP32 / ESP8266): Sebagai otak utama sistem yang memproses data sensor dan mengirimkannya ke internet (*Internet of Things*).
2. Sensor Ultrasonik (HC-SR04): Sebagai sensor utama untuk mendeteksi jarak permukaan *liquid*(mengukur level liquid).
3. Modul Relay (2 Channel): Saklar elektronik untuk mengaktifkan dan mematikan pompa *liquid*berdasarkan logika program.
4. Layar Tampilan (LCD ILI9341 atau OLED): Untuk menampilkan status level *liquid*dan status pompa secara lokal (*real-time*).
5. Power Supply (Adaptor 12V / 5V): Sumber tegangan untuk menyuplai daya ke pompa dan mikrokontroler.
6. Kabel Jumper (Male-to-Male, Male-to-Female): Untuk menghubungkan jalur antar komponen pada *breadboard*.
7. Breadboard / PCB Dot Matrix: Papan sirkuit cetak untuk menempatkan komponen.
8. Wadah / Tangki Prototype: Sebagai simulasi tempat penampungan cairan (cairan *storage*).
9. Komponen Pasif: Resistor, Kapasitor, LED Indikator, dan Buzzer (alarm peringatan level kritis).

3.2 Alur Penelitian

Metodologi penelitian ini dibagi dalam beberapa tahapan utama yang ditunjukkan pada Gambar dibawah ini.



Gambar 3. 1 Alur Penelitian

Metodologi penelitian ini dibagi dalam beberapa tahapan utama yang ditunjukkan pada Gambar 3.1 di atas. Tahapan penelitian dimulai dengan Identifikasi Masalah untuk menentukan batasan dan tujuan penelitian terkait sistem monitoring dan kendali level liquid. Setelah masalah terdefinisi, dilakukan Studi Literatur dan Persiapan Data Teknis guna mengumpulkan referensi teori, spesifikasi komponen, serta jurnal pendukung yang relevan dengan teknologi IoT dan mikrokontroler.

Langkah selanjutnya adalah Perencanaan Sistem dan Parameter Desain, di mana spesifikasi alat dan kriteria fungsional sistem ditetapkan. Setelah perencanaan

matang, dilakukan Survei atau Studi Lapangan untuk memahami kondisi nyata di lokasi penempatan alat seperti pada wadah/tandon air. Hasil dari survei ini kemudian masuk ke tahap Validasi Data. Jika data lapangan dirasa belum cukup atau tidak akurat (pilihan "Tidak"), maka proses akan kembali ke tahap survei. Namun, jika data sudah valid (pilihan "Ya"), maka penelitian dilanjutkan ke tahap Analisis Data untuk menentukan logika kendali dan integrasi sistem yang optimal.

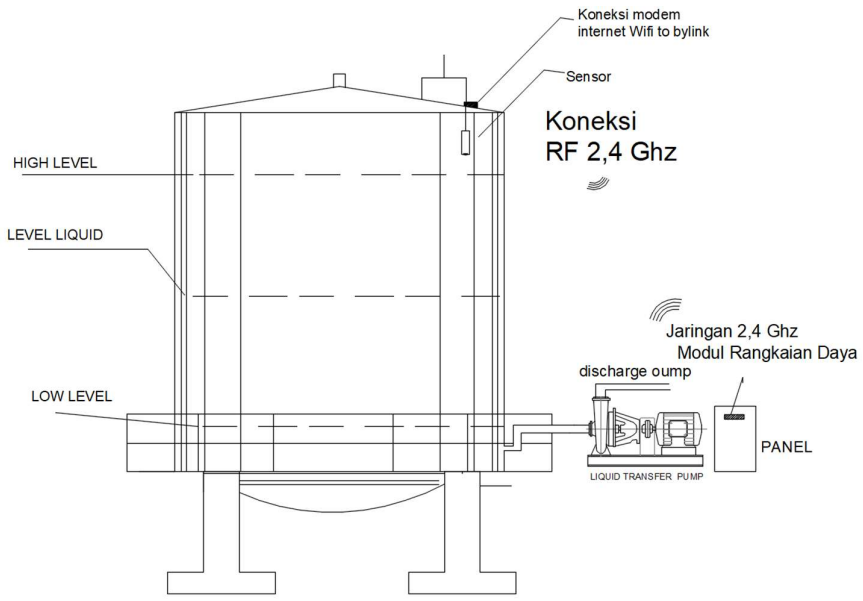
Setelah analisis selesai, dilakukan Perancangan Sistem Elektronik dan IoT yang mencakup integrasi berbagai sensor, ESP32, relay, pompa, hingga pembuatan *dashboard* monitoring. Tahap berikutnya adalah Pembuatan Prototype dan Integrasi Sistem, di mana seluruh komponen dirakit menjadi satu kesatuan alat yang utuh. Alat yang telah jadi kemudian masuk ke tahap Pengujian Sistem untuk memastikan akurasi pembacaan sensor, kestabilan kontrol pompa, serta kelancaran pengiriman data melalui jaringan IoT. Sebagai tahap akhir, disusun Kesimpulan dan Rekomendasi Pengembangan berdasarkan hasil pengujian untuk memberikan saran perbaikan di masa mendatang sebelum penelitian dinyatakan Selesai.

3.3 Perancangan Hardware Dan Perangkat Keras

Tahap perancangan ini merupakan langkah krusial untuk merealisasikan sistem yang telah direncanakan ke dalam bentuk fisik. Perancangan hardware meliputi pemilihan komponen, pemetaan jalur sirkuit, hingga integrasi antarmuka antar perangkat. Penjelasan lebih mendalam mengenai alur pemikirannya akan dibahas pada poin-poin berikut

3.3.1 Konsep Perancangan Alat

Berdasarkan Gambar dibawah ini, perancangan sistem ini difokuskan pada pemantauan level cairan dan pengendalian pompa secara otomatis berbasis *Internet of Things* (IoT). Secara garis besar, sistem terdiri dari tangki penyimpanan, unit sensor, modul kendali daya, dan antarmuka monitoring.



Gambar 3. 5 Integrasi Sistem Kendali Level Cairan Otomatis

Sistem ini mengintegrasikan pemantauan level secara *real-time* dengan kontrol pompa otomatis menggunakan komunikasi *hybrid* (Wi-Fi dan RF).

1. **Deteksi Level Cairan** Sensor ultrasonik di atas tangki membagi volume menjadi tiga zona kritis:

- High Level: Batas atas (interlock) untuk mencegah *overflow*.
- Actual Level: Data volume *real-time*.
- Low Level: Batas bawah sebagai pemicu (*trigger*) pengisian.

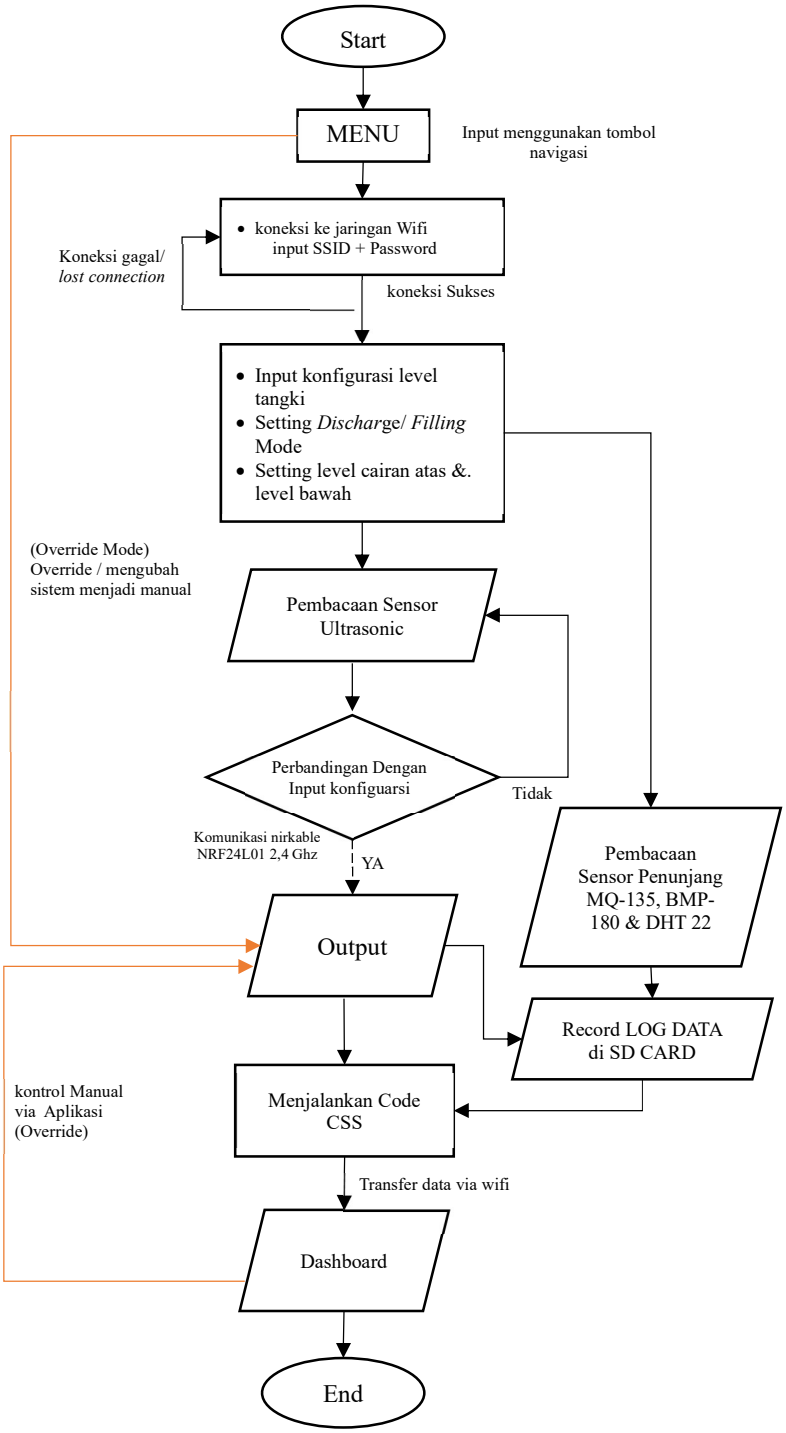
2. Arsitektur Komunikasi Dua Jalur

- Wi-Fi (IoT): Mengirim data ke *dashboard* untuk pemantauan jarak jauh via Web/Smartphone.
- Radio Frequency (RF 2.4 GHz): Jalur komunikasi lokal (nirkabel) antara sensor dan panel kontrol untuk respons cepat tanpa ketergantungan internet.

3. Kontrol kontaktor (Pompa) Modul Rangkaian Daya menerima instruksi via RF untuk menggerakkan Liquid Transfer Pump:

- Auto-Start: Pompa menyala saat menyentuh *Low Level*.
- Auto-Stop: Pompa mati saat menyentuh *High Level* untuk keamanan sistem.

Flowchart sistem



Gambar 3. 6 Flowchart sistem

Alur kerja sistem ini mencakup proses otomatisasi kendali level cairan serta monitoring parameter lingkungan secara simultan. Penjelasan detail mengenai tahapan sistem adalah sebagai berikut:

1. Inisialisasi, Koneksi, dan Konfigurasi

Sistem memulai operasional melalui tampilan MENU utama dengan input menggunakan tombol navigasi. Langkah pertama adalah menghubungkan perangkat ke jaringan Wi-Fi melalui input *SSID* dan *Password*. Apabila koneksi berhasil, pengguna dapat masuk ke tahap pengaturan parameter operasional yang meliputi:

- Konfigurasi Level Tangki: Menentukan dimensi fisik tangki penyimpanan.
- Mode Operasi: Memilih antara mode *Discharge* (pengosongan) atau *Filling* (pengisian).
- Threshold Level: Mengatur batas ambang atas dan bawah sebagai acuan sistem kendali.

2. Akuisisi Data Sensor Utama dan Logika Kendali

Sistem melakukan pembacaan pada Sensor Ultrasonic untuk mendapatkan data ketinggian level cairan. Data ini kemudian dibandingkan dengan input konfigurasi yang telah ditetapkan.

- Jika kondisi ambang batas belum terpenuhi (TIDAK), sistem akan terus melakukan pengulangan pembacaan sensor.
- Jika kondisi terpenuhi (YA), instruksi akan dikirimkan melalui komunikasi nirkabel menggunakan modul NRF24L01 2,4 GHz menuju bagian Output (aktivasi relay/pompa).

3. Akuisisi Sensor Penunjang dan Penyimpanan Data (*Data Logging*)

Secara paralel setelah tahap konfigurasi, sistem juga melakukan pemantauan kondisi lingkungan melalui Sensor Penunjang yang terdiri dari:

- MQ-135: Untuk memantau kualitas udara.
- BMP-180: Untuk mengukur tekanan atmosfer dan suhu.
- DHT 22: Untuk mengukur kelembapan dan suhu udara di dalam bejana.

Seluruh data hasil pembacaan, baik dari sensor utama maupun sensor penunjang, serta status *output*, kemudian disimpan secara lokal dalam format (.csv) Record LOG DATA di SD CARD. Hal ini berfungsi sebagai pencatatan data untuk dianalisa di kemudian waktu.

4. Transmisi Data, *WiFi Tunneling*, dan Visualisasi *Dashboard*

Data yang telah diakuisisi oleh sensor diproses secara internal oleh ESP32 yang berfungsi sebagai *Host* atau *Web Server* lokal. Untuk membangun antarmuka pengguna, ESP32 menjalankan instruksi pemrograman HTML dan CSS yang tersimpan dalam memori perangkat guna menghasilkan *dashboard* visual yang informatif.

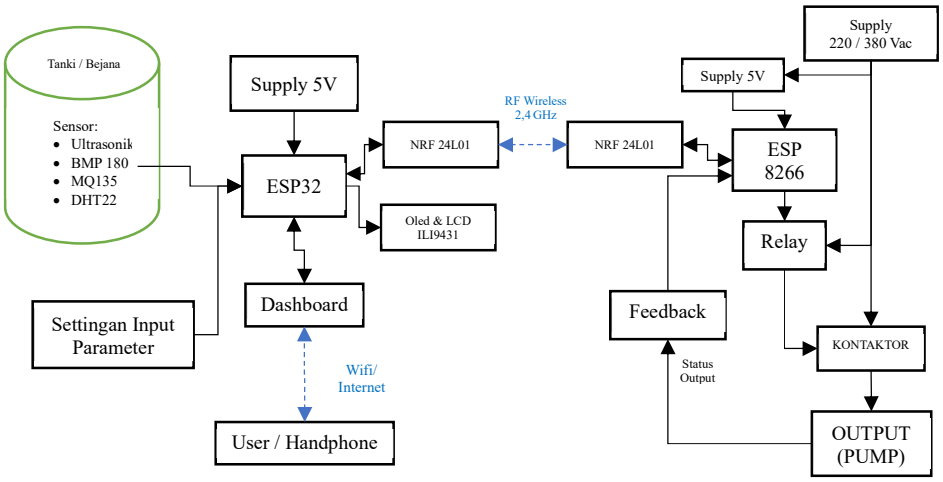
Agar sistem ini dapat diakses melalui jaringan internet secara luas (tidak terbatas pada jaringan lokal), diterapkan metode WiFi Tunneling melalui *router*. Teknik ini berfungsi untuk memetakan jalur komunikasi dari alamat IP lokal ESP32 ke sebuah *domain* atau alamat publik yang dapat dijangkau dari luar jaringan. Dengan adanya *tunneling* ini, pengguna dapat melakukan pemantauan status level cairan serta parameter lingkungan (suhu, tekanan, dan kualitas udara) secara *real-time* dari mana saja selama tersedia koneksi internet, tanpa memerlukan server fisik eksternal.

5. Mekanisme Kendali Manual (*Override*)

Sistem menyediakan fitur Override Mode yang memungkinkan pengguna untuk mengambil alih kendali sistem menjadi manual. Fitur ini dapat diakses langsung melalui menu pada alat atau dikendalikan dari jarak jauh melalui aplikasi (*Dashboard*), yang secara langsung akan menginstruksikan bagian Output tanpa menunggu logika dari sensor.

.

3.3.2 Perancangan Perangkat Keras



Gambar 3. 7 Diagram Blok Rangkaian

Deskripsi Blok Diagram

Sistem ini dirancang dengan struktur terpusat pada Mikrokontroler ESP32 sebagai otak utama. Secara garis besar, aliran kerjanya dibagi menjadi empat bagian:

- Sistem Input: Terdiri dari sensor lingkungan (Ultrasonik, BMP180, MQ135) serta Settingan Input Parameter (tombol analog) untuk memberikan data mentah ke kontroler.
- Unit Pemroses (NodeMCU ESP32): Mengolah data sensor, mengatur logika kendali, dan mengelola komunikasi data baik secara lokal maupun ke jaringan internet.
- Sistem Output & Display: Hasil pengolahan ditampilkan secara visual melalui OLED & LCD ILI9431. Terdapat juga output fisik berupa buzzer dan laser pointer untuk indikator alarm atau penanda.
- Komunikasi Data: Menggunakan modul nRF24L01 (2.4 GHz) untuk pengiriman data jarak pendek atau kontrol ke relay/kontaktor, dan Menggunakan koneksi Wi-Fi internal ESP32 untuk menghubungkan Dashboard yang dapat diakses oleh user melalui smartphone.

3.3.2.1 Schematik diagram Rangkaian Kontrol

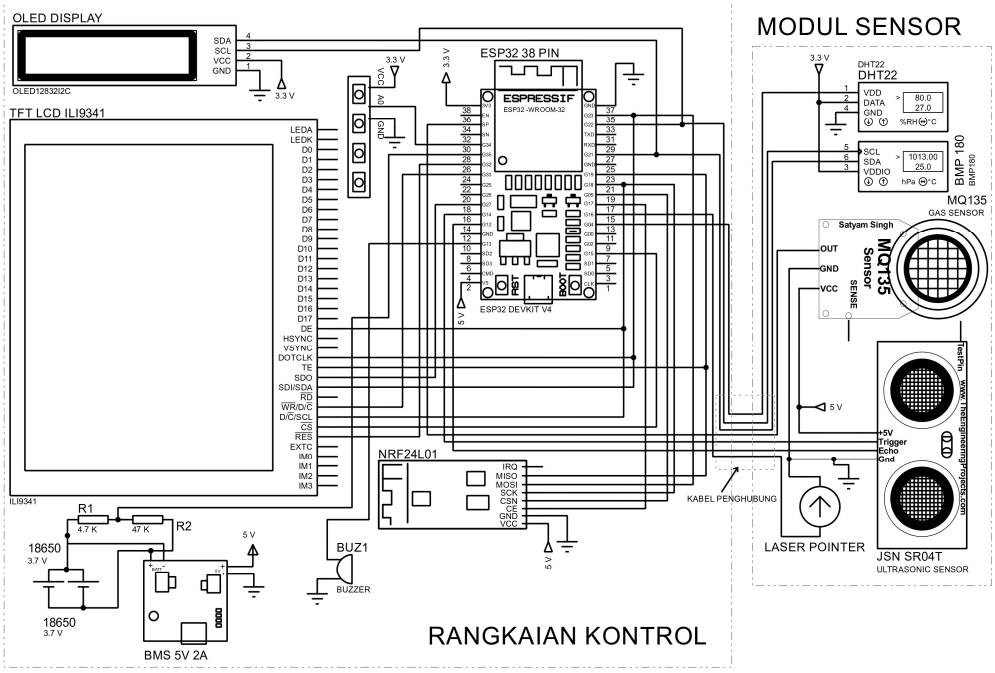
Untuk memastikan seluruh perangkat keras dapat berkomunikasi dengan pengontrol utama, dilakukan pemetaan *pinout* pada ESP32. Pemetaan ini mencakup alokasi pin digital, analog, serta jalur komunikasi bus seperti SPI dan I2C. Detail mengenai konfigurasi koneksi antar komponen terhadap pin mikrokontroler disajikan dalam Tabel 3.1 berikut ini

Tabel 3. 1 Tabel Perancangan pinout ESP 32

Komponen / Periferal	**Pin ESP32**	**Fungsi / Keterangan**
TFT LCD ILI9341	15	CS (Chip Select)
	32	RST (Reset)
	33	DC (Data/Command)
	23	MOSI (SPI Standard)
	18	SCK (SPI Standard)
SD Card Module	27	CS (Chip Select)
nRF24L01 (Radio)	17	CE (Chip Enable)
	5	CSN (Chip Select Not)
Sensor JSN-SR04T	14	TRIG (Trigger)
	12	ECHO (Echo)
Sensor DHT22	4	Data Pin
Sensor MQ-135	36 (VP)	Analog Output (Gas)
Sensor BMP180	21	SDA (I2C Standard)
	22	SCL (I2C Standard)
Layar OLED SSD1306	21, 22	I2C (Paralel dengan BMP180)
Tombol Analog (Input)	34	AD_PIN (Resistor Ladder)
Monitor Baterai	35	BATT_PIN (Analog Input)
Buzzer	13	Output Alarm
Laser Pointer	16	Output Laser

Pengalokasian *pin* di atas dirancang sedemikian rupa untuk menghindari konflik instruksi antar periferal, terutama pada modul yang berbagi bus data yang sama. Seluruh koneksi fisik antara komponen-komponen tersebut dengan ESP32 akan digambarkan secara visual melalui diagram skematik pada bagian selanjutnya.

Skema Rangkaian kontrol



Gambar 3. 8 Skema Rangkaian Kontrol

Perancangan pengabelan (wiring) pada ESP32 ini sangat padat dan memanfaatkan berbagai protokol komunikasi (SPI, I2C, Analog, dan GPIO Digital). Berikut adalah detail alokasi pin-nya:

A. Antarmuka Visual (Display)

Sistem ini menggunakan dua jenis display untuk efisiensi informasi:

- TFT LCD ILI9341: Menggunakan komunikasi SPI Standard. Pin yang digunakan adalah GPIO 15 (CS), 32 (RST), 33 (DC), 23 (MOSI), dan 18 (SCK). Ini digunakan untuk grafis yang lebih kompleks.
- OLED SSD1306: Menggunakan protokol I2C. Menariknya, OLED ini dipasang paralel dengan sensor BMP180 pada Pin 21 (SDA) dan 22 (SCL), menghemat penggunaan jumlah pin.

B. Komunikasi Radio & Penyimpanan

- nRF24L01: Menggunakan pin 17 (CE) dan 5 (CSN). Modul ini bekerja pada frekuensi 2.4 GHz untuk komunikasi nirkabel antar perangkat.

- SD Card Module: Terhubung pada Pin 27 (CS) untuk keperluan *data logging* (pencatatan data sensor secara offline).

C. Akuisisi Data Sensor

- Sensor JSN-SR04T (Ultrasonik): Menggunakan Pin 14 (Trigger) dan 12 (Echo) untuk mengukur jarak.
- Sensor Gas MQ-135: Terhubung ke Pin 36 (VP), yang merupakan pin Analog-to-Digital Converter (ADC) untuk membaca konsentrasi gas.
- Sensor Lingkungan: BMP180 (Tekanan/Suhu) menggunakan I2C, dan DHT22 (Kelembapan) menggunakan Pin 4.

D. Input Analog & Monitoring

Perancang menggunakan teknik Resistor Ladder pada Pin 34. Ini adalah teknik cerdas untuk membaca banyak tombol hanya dengan satu pin analog. Selain itu, Pin 35 digunakan sebagai Monitor Baterai untuk memantau sisa daya perangkat.

E. Output

Untuk indikator peringatan, perancang mengalokasikan Pin 13 untuk Buzzer (alarm suara) dan Pin 16 untuk Laser Pointer (indikator visual/target).

3.3.2.1 Schematik diagram Rangkaian Daya

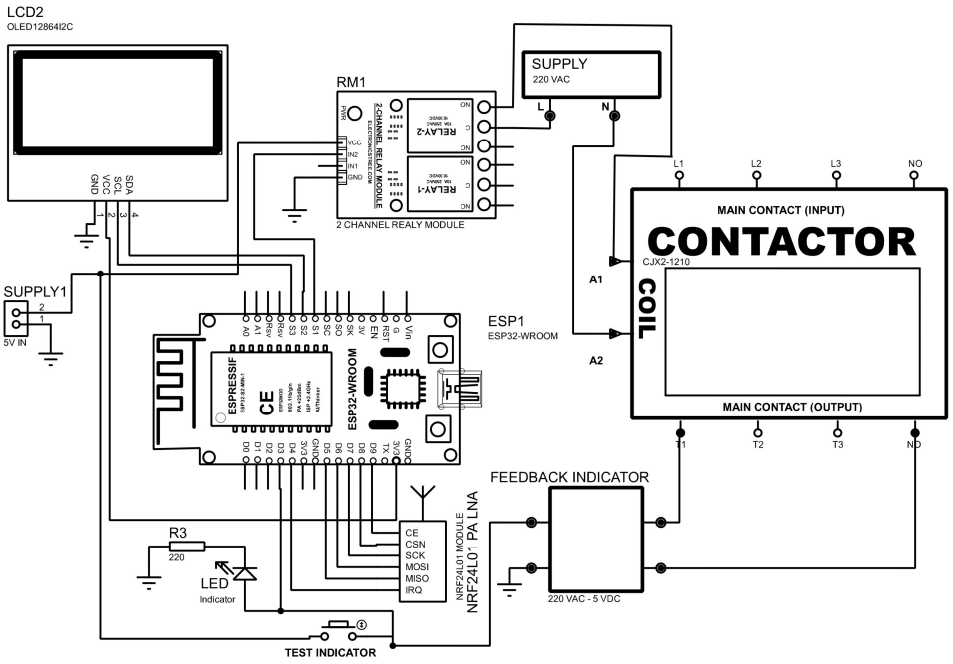
Untuk memastikan integrasi yang tepat antara mikrokontroler dengan perangkat eksternal, dilakukan pemetaan pin (pinout) pada ESP8266. Pemetaan ini mencakup koneksi untuk modul komunikasi nirkabel NRF24L01, modul tampilan OLED SSD1306, serta jalur input/output digital untuk kendali relay dan sinyal balik (*feedback*).

Tabel 3. 2 Tabel Perancangan Pinout Esp8266

Komponen	Pin modul	Pin ESP 8266	Keterangan
NRF24L01	VCC	3.3V	VCC
	GND	G / GND	GND
	CE	D3	CE_PIN D3.
	CSN	D8	CSN_PIN D8.
	SCK	D5	Hardware SPI Clock.
	MISO	D6	Hardware SPI MISO.
	MOSI	D7	Hardware SPI MOSI.

OLED SSD1306	SDA	D2	Jalur Data I2C.
	SCL	D1	Jalur Clock I2C.
Output/Input	Relay	D4	Pin Relay.
	Feedback	D0	Feedback.

Berdasarkan rincian pada tabel di atas, konfigurasi pin pada ESP8266 telah disesuaikan dengan standar protokol komunikasi perangkat. Pin D5, D6, dan D7 dialokasikan khusus untuk jalur *Hardware SPI* guna menjamin kecepatan transfer data yang stabil pada modul NRF24L01. Sementara itu, penggunaan pin D1 dan D2 untuk jalur I2C memungkinkan efisiensi pengalamatan data pada layar OLED tanpa mengganggu performa pin digital lainnya



Gambar 3. 9 Skema Rangkaian Daya

Berdasarkan gambar skema layout dan tabel referensi yang Anda berikan, berikut adalah deskripsi penggunaan kaki (pinout) pada sistem kendali berbasis ESP32-WROOM (pada layout) atau ESP8266 (pada tabel):

1. Modul Komunikasi NRF24L01 (SPI)

Modul ini digunakan untuk komunikasi nirkabel jarak jauh. Jalur yang digunakan adalah protokol SPI:

- VCC: Dihubungkan ke tegangan 3.3V.
- GND: Ground sistem.
- CE: Dihubungkan ke pin D3 (digunakan untuk mengontrol mode *standby* atau *active*).
- CSN: Dihubungkan ke pin D8 (*Chip Select Not* untuk memulai komunikasi SPI).
- SCK: Dihubungkan ke pin D5 (*Serial Clock*).
- MISO: Dihubungkan ke pin D6 (*Master In Slave Out*).
- MOSI: Dihubungkan ke pin D7 (*Master Out Slave In*).

2. Layar OLED SSD1306 (I2C)

Layar ini berfungsi sebagai penampil data/status menggunakan protokol I2C:

- SDA (Data): Dihubungkan ke pin D2.
- SCL (Clock): Dihubungkan ke pin D1.
- VCC & GND: Dihubungkan ke jalur daya 3.3V/5V dan Ground.

3. Kontrol Output (Relay & Contactor)

Sistem ini menggunakan relay untuk menggerakkan kontaktor AC yang lebih besar:

- Pin Relay (D4): Pin ini mengirim sinyal ke modul RM1 (Relay Module).
- Relay 1: Memutus atau menyambung arus dari Supply 220VAC menuju Coil A1/A2 pada Contactor.
- Contactor: Digunakan untuk memutus/menyambung beban utama (L1, L2, L3) ke terminal output (T1, T2, T3).

4. Feedback & Indikator

- Feedback (D0): Mendapat input dari blok Feedback Indicator. Alat ini mengonversi tegangan 220VAC (dari output kontaktor) menjadi sinyal rendah (3.3VDC) agar ESP bisa mengetahui apakah beban benar-benar menyala.
- LED Indicator: Terhubung ke salah satu pin digital (melalui resistor R3) sebagai penanda status perangkat secara visual Test Indicator/Push Button.

3.4 Perancangan Perangkat Lunak (*Software Design*)

Perangkat lunak sistem ini dibangun menggunakan bahasa C++ pada framework Arduino IDE dengan arsitektur *Super-loop* (non-blocking). Sistem menjalankan tugas *multitasking* memanfaatkan fungsi *millis()* untuk menghindari *delay* saat pemrosesan sensor dan *web server*.

Beberapa pustaka utama yang digunakan meliputi:

- *WiFi.h & WebServer.h*: Pengelolaan koneksi internet dan *dashboard* monitoring.
- *Adafruit_ILI9341 & Adafruit_GFX*: Antarmuka grafis pada layar TFT 2.4".
- *RF24.h*: Protokol komunikasi nirkabel antara *node* sensor dan aktuator.
- *SD.h & EEPROM.h*: Penyimpanan log data (.csv) dan memori konfigurasi permanen.

3.4.1 Perancangan Program ESP32 (Rangkaian Control)

Pada bagian ini, fokus utama dialokasikan pada pengembangan perangkat lunak (firmware) yang ditanamkan ke dalam mikrokontroler ESP32. Sebagai otak atau pusat kendali dari seluruh sistem, ESP32 bertanggung jawab untuk mengoordinasikan kerja seluruh komponen elektronik agar berjalan sesuai skenario yang diinginkan.

Program yang dirancang pada ESP32 ini mengintegrasikan lima logika utama yang memastikan sistem bekerja secara presisi, responsif, dan aman:

3.4.1.1 Algoritma Akuisisi Data dan Proteksi Sensor

Proses konversi data pada sistem ini melibatkan transformasi besaran faktuil (waktu tempuh gelombang) menjadi besaran fisis (jarak) dan variabel informatif (volume/tinggi air).

A. Perhitungan Fisik Sensor Ultrasonik

Sistem menggunakan sensor ultrasonik yang bekerja dengan prinsip pantulan gelombang suara. Kalkulasi jarak dilakukan dengan mengukur durasi waktu antara

pengiriman gelombang (*Trigger*) hingga diterimanya pantulan (*Echo*). Rumus yang diterapkan pada *firmware* adalah:

$$s = \frac{t \times v}{2} \quad \text{.....(3.1)}$$

Di mana:

- s = Jarak antara sensor dan permukaan cairan (cm).
- t = Durasi waktu tempuh gelombang (*microsecond*).
- v = Kecepatan suara (di udara ≈ 0,034 cm/µs).
- Pembagi 2 digunakan karena gelombang menempuh jarak pulang-pergi.

B. Konversi Tinggi Cairan Aktual

Untuk mendapatkan tinggi air sesungguhnya dalam wadah, digunakan parameter kalibrasi dasar tangki (setLow). Jarak yang terbaca oleh sensor merupakan ruang kosong (*headspace*), sehingga tinggi air dihitung dengan rumus:

$$Tinggi_{Air} = setLow - s \quad \text{.....(3.2)}$$

- $setLow$: Jarak dari sensor ke dasar tangki (titik nol air).
- s: Jarak permukaan air ke sensor hasil pembacaan ultrasonik.

C. Kalkulasi Persentase Volume

Sistem menghitung persentase level cairan untuk memudahkan pemantauan operator dengan membandingkan tinggi air terhadap rentang operasional (*setLow* ke *setFull*):

$$Level(\%) = \left(\frac{Tinggi_{Air}}{setLow - setFull}\right) \times 100\% \quad \text{.....(3.3)}$$

d. Fitur *Error Protection* & Validasi Rasional

Untuk menjamin keamanan aktuator (pompa), sistem dilengkapi dengan algoritma **Validasi Data Rasional**. Fitur ini bekerja dengan mendeteksi anomali pada hasil pembacaan:

- **Deteksi Nilai Negatif:** Jika sensor membaca jarak yang lebih besar dari batas dasar tangki ($s > setLow$), yang mengakibatkan hasil kalkulasi $Tinggi_{Air}$ bernilai negatif, maka sistem akan mengaktifkan flag calibError.
- **Deteksi *Out of Range*:** Jika durasi pantulan tidak diterima (sensor terhalang atau rusak), sistem akan memberikan status *N/A* (Not Available).
- **Aksi *Fail-Safe*:** Saat salah satu kondisi error di atas terpenuhi, sistem secara otomatis menjalankan prosedur keselamatan dengan memutus aliran listrik ke pompa (OFF) secara permanen hingga data kembali valid. Hal ini mencegah kerusakan pompa akibat fenomena *dry running* atau pengisian berlebih yang tidak terdeteksi.

3.4.1.2 Algoritma Kendali Aktuator (*Relay Control Logic*)

Logika kendali relai pada sistem ini dirancang untuk bekerja secara mandiri (*autonomous*) dengan mempertimbangkan aspek keamanan alat dan fleksibilitas operasional. Kontrol relai dibagi menjadi tiga mekanisme utama:

A. Mode Otomatis 1: *Filling* (Sistem Pengisian) Mode ini digunakan untuk menjaga ketersediaan cairan dalam tangki (misal: *ground tank*).

- **Aktivasi:** Pompa akan berubah ke status ON secara otomatis jika persentase level cairan menyentuh atau kurang dari ambang batas bawah (setLimitLow).
- **Deaktivasi:** Pompa akan berubah ke status OFF hanya setelah cairan mencapai atau melebihi ambang batas atas (setLimitHigh).
- **Prinsip Histeresis:** Penggunaan dua batas (atas dan bawah) bertujuan untuk mencegah kondisi *chattering* (pompa mati-nyala secara cepat) saat permukaan air bergelombang.

B. Mode Otomatis 2: *Discharging* (Sistem Pengosongan) Mode ini digunakan untuk aplikasi drainase atau pembuangan cairan (misal: *pit* limbah).

- **Aktivasi:** Pompa akan aktif (ON) saat cairan mencapai batas atas (setLimitHigh) untuk mencegah luapan (*overflow*).

- **Deaktivasi:** Pompa akan berhenti (OFF) saat cairan telah terkuras hingga menyentuh batas bawah (setLimitLow) untuk mencegah kerusakan pompa akibat berjalan tanpa air (*dry running*).

C. Mekanisme *Manual Override* & Interupsi Prioritas Sistem dilengkapi dengan fitur kendali manual yang memiliki prioritas lebih tinggi daripada logika otomatis:

- **Interupsi Fisik:** Operator dapat menekan dan menahan tombol *Select* pada perangkat selama 4 detik. Hal ini akan memicu *loop* program khusus yang menghentikan logika sensor dan memaksa status relai sesuai keinginan operator.
- **Interupsi nirkabel (Web Dashboard):** Melalui antarmuka HTTP, operator dapat mengirimkan permintaan *Force Start/Stop*. Saat status *Override* aktif, sistem akan mengabaikan pembacaan sensor hingga operator secara manual mengembalikan sistem ke mode otomatis melalui perintah *Exit Override*.
- **Indikator Visual:** Saat mode *Override* aktif, layar TFT akan menampilkan kotak peringatan merah berkedip sebagai bentuk prosedur keselamatan kerja bagi operator di lapangan.

D. Proteksi Kegagalan Kalibrasi (*Fail-Safe*) Sebagai lapisan keamanan tambahan, logika relai terintegrasi dengan status calibError. Jika sensor ultrasonik membaca nilai yang tidak valid (misal: jarak lebih besar dari tinggi tangki akibat gangguan teknis), sistem secara otomatis memutus aliran listrik ke pompa (OFF) untuk mencegah malfungsi yang tidak terprediksi.

3.4.1.3 *Web Server* & Antarmuka Monitoring Asinkron

Sistem ini mengimplementasikan pusat pemantauan berbasis *Internet of Things* (IoT) yang memungkinkan interaksi dua arah antara operator dan perangkat melalui jaringan nirkabel.

A. Arsitektur Komunikasi JSON & HTTP

- **Enkapsulasi Data JSON:** Seluruh variabel sensor (jarak, persentase level, suhu, kelembaban, tekanan, dan kualitas gas) dikemas dalam format *JavaScript Object Notation* (JSON) menggunakan fungsi snprintf(). Format ini dipilih karena ringan dan standar dalam pertukaran data *web-based*.

- Pertukaran Data Asinkron (AJAX/Fetch): Untuk menampilkan grafik dan nilai secara *real-time*, antarmuka *dashboard* menggunakan metode *asynchronous* (Fetch API). Hal ini memungkinkan data di halaman *web* diperbarui setiap 1 detik tanpa perlu melakukan penyegaran (*refresh*) seluruh halaman, sehingga menghemat *bandwidth* dan beban kerja ESP32.

B. Visualisasi Grafis *Real-Time*

- Client-Side Recording**:** Grafik pemantauan pada *dashboard* dibangun menggunakan pustaka *Highcharts*. Proses *rendering* grafik dilakukan pada sisi klien (*browser*), sementara ESP32 hanya bertindak sebagai penyedia data mentah. Hal ini menjaga stabilitas memori RAM mikrokontroler meskipun visualisasi data sangat kompleks (mencakup 4 parameter sekaligus).
- Dashboard Multi-Card**:** Antarmuka dirancang menggunakan *CSS Flexbox* untuk tata letak yang responsif, mencakup kartu status tangki (visualisasi cairan), kartu data lingkungan, dan panel kontrol pompa.

C. Fitur *Floating Chat* & Manajemen *Buffer*

- Operator Chat System: Tersedia jendela pesan mengambang (*Floating Chat Window*) untuk komunikasi antar operator lapangan yang mengakses *IP Address* yang sama.
- RAM-Based Message Buffer**:** Pesan teks dikelola menggunakan algoritma *Circular Buffer* (kapasitas 10 pesan terakhir) di dalam memori RAM ESP32. Setiap pesan diidentifikasi berdasarkan *Global Message ID* dan *Unique IP Sender* untuk memastikan urutan percakapan tetap sinkron secara asinkron.

D. Kontrol Interupsi & *Web Override*

- Bi-directional Control: Selain monitoring, *web server* menyediakan fitur *Force Start/Stop* melalui permintaan HTTP khusus (/override_on dan /override_off). Perintah ini memiliki prioritas interupsi tinggi, yang mampu mengesampingkan logika otomatis sensor saat terjadi kondisi darurat yang terdeteksi oleh operator secara visual.

3.4.1.4 Mekanisme *Data Logging* & Sistem *Anti-Crash*

Sistem ini mengintegrasikan penyimpanan eksternal dan retensi memori internal untuk menjamin kontinuitas data dan operasional.

A. *Data Logging* Terjadwal & Dinamis

- **Pencatatan Periodik:** Data sensor diarsip ke *MicroSD Card* setiap 60 detik dalam format .csv untuk analisis tren jangka panjang.
- **Stempel Waktu NTP:** Penamaan file dan *timestamp* data disinkronisasi secara otomatis melalui *Network Time Protocol* (NTP). Hal ini mencegah duplikasi file dan mempermudah audit data kronologis (Contoh: Senin-16-02-2026.csv).
- **Struktur Data Komprehensif:** Baris log mencatat parameter mentah (*raw*), kalkulasi tinggi air, persentase volume, kondisi lingkungan (Suhu/Kelembaban/Gas), hingga status *relay*.

B. Algoritma *Anti-Crash* & *Self-Healing* (EEPROM) Untuk mengatasi kegagalan daya (*power outage*), sistem menerapkan manajemen memori non-volatil:

- **Retensi Parameter:** Nilai kalibrasi (*setFull*, *setLow*), ambang batas kontrol, dan mode operasi disimpan menggunakan fungsi EEPROM.put(). Data tetap tersimpan meskipun suplai daya terputus.
- **Validasi Booting:** Saat *startup*, sistem menjalankan algoritma *Self-Healing*. Jika data pada EEPROM terdeteksi korup atau tidak rasional (NaN/*Not a Number*), sistem secara otomatis memulihkan parameter ke nilai *default* pabrik.
- **Operasional Non-Blocking:** Seluruh proses penulisan data dilakukan tanpa fungsi delay(), sehingga sistem tetap responsif terhadap interupsi keamanan (seperti kebocoran gas) dan kendali *Web Dashboard* secara *real-time*.

Sinkronisasi NRF24L01: Menjamin pengiriman data stabil dengan menyamakan dua parameter utama:

Kanal (*Channel*): Disetel pada kanal yang sama (contoh: kanal 115) menggunakan *radio.setChannel(115)* untuk menghindari interferensi sinyal WiFi, Channel: 115 (Frekuensi 2.400 + 115 = 2.515 GHz). Ini

adalah channel yang bagus karena berada di luar jangkauan standar Wi-Fi umum (2.400 - 2.4835GHz), sehingga minim interferensi.

- **Alamat (*Address*)**: Menggunakan alamat pipa 5-byte yang identik (contoh: "00001") pada unit pengirim dan penerima melalui fungsi *radio.openWritingPipe(rfAddress)*dan *radio.openReadingPipe()*.
- **Data Rate:** 250 kbps. Ini adalah kecepatan terendah yang tersedia, namun memberikan jarak jangkauan (range) terjauh dan sensitivitas penerimaan terbaik

3.4.2 Perancangan Program ESP8266 (Rangkaian Daya)

Pada bagian ini, fokus perancangan diarahkan pada unit pemrosesan di sisi beban atau daya yang menggunakan mikrokontroler ESP8266. Perangkat ini berfungsi sebagai *node receiver* yang menerima instruksi dari pusat kendali untuk mengoperasikan beban elektrik berdaya tinggi.

3.4.2.1 Implementasi Logika Kendali Relai

Pada penelitian ini, aktuator utama berupa pompa air dikendalikan oleh mikrokontroler melalui modul relai. Untuk memastikan keandalan dan keamanan sistem, khususnya saat terjadi gangguan komunikasi nirkabel, perangkat lunak pada *node receiver* (penerima) dirancang menggunakan pendekatan sistem berbasis aturan (*rule-based system*) yang dilengkapi dengan protokol keamanan mandiri.
Secara garis besar, logika perancangan perangkat lunak pada *node receiver* dibagi menjadi empat mekanisme utama:

3.4.2.2 Prinsip Kendali Relai Aktif-Rendah (*Active-Low Control*)

Sistem ini menggunakan relai dengan karakteristik *active-low*, di mana aktuator akan terhubung (aktif) ketika pin mikrokontroler memberikan sinyal logika LOW, dan terputus (mati) saat diberikan sinyal logika HIGH.
Sebagai langkah mitigasi perangkat keras saat sistem pertama kali dihidupkan (*booting*), pin relai (RELAY_PIN) secara bawaan (*default*) langsung diinisialisasi ke kondisi HIGH pada blok setup(). Hal ini bertujuan untuk mencegah terjadinya lonjakan arus sesaat (*glitch*) yang dapat menyebabkan pompa menyala secara tidak sengaja saat mikrokontroler sedang melakukan inisialisasi modul komunikasi

nRF24L01. Setelah sistem berjalan, eksekusi perintah pembacaan menggunakan operator *ternary* untuk menerjemahkan nilai *boolean* dari data masuk menjadi sinyal tegangan yang sesuai.

3.4.2.3 Algoritma Pengambilan Keputusan (Sistem Berbasis Aturan)

Sistem kendali dirancang menggunakan logika deterministik yang secara terus-menerus mengevaluasi *state* (keadaan) dari paket data yang masuk. Mikrokontroler melakukan validasi kondisi operasi menggunakan parameter incomingData.pump yang diterima dari *node transmitter*.

Tabel 3. 3 Matriks Keputusan Kendali Relai

Kondisi Data Masuk *(incomingData.pump)*	**Status Koneksi**	**Output Sinyal Relai**	**Status Pompa**
True (ON)	Terhubung	LOW	Menyala
False (OFF)	Terhubung	HIGH	Mati
Apapun nilainya	Terputus (> 3 detik)	HIGH	Mati (Failsafe)

Setelah mikrokontroler mengeksekusi perintah pada relai, sistem melakukan validasi aliran aktual menggunakan sensor aliran (*flow sensor*) melalui FEEDBACK_PIN. Data umpan balik ini (isPumpOn dan sensorStatus) kemudian dikemas ke dalam variabel FeedbackPacket dan ditransmisikan kembali ke *node transmitter* melalui fitur AckPayload pada nRF24L01. Mekanisme ini membentuk sistem kendali tertutup (*closed-loop control*).

3.4.2.4 Sistem Interupsi Keamanan Perangkat Lunak (***Software Watchdog Failsafe***)

Untuk memitigasi risiko kerusakan aktuator atau luapan air akibat putusnya koneksi antara pengirim dan penerima, sistem dilengkapi dengan interupsi berbasis waktu (*software watchdog*).

Setiap kali paket data berhasil diterima, mikrokontroler mencatat waktu absolut ke dalam variabel penampung menggunakan fungsi millis(). Pada siklus pemrosesan utama, sistem membandingkan waktu aktual dengan waktu penerimaan terakhir. Jika selisih waktu melampaui ambang batas toleransi sebesar 3.000

milidetik (3 detik), sistem menyimpulkan bahwa komunikasi terputus (*connection loss*). Mikrokontroler kemudian mengambil alih kendali dengan secara sepihak memutus aliran listrik ke pompa (mengirimkan sinyal HIGH ke relai) sebagai tindakan pengamanan kritis darurat.

3.4.2.5 Protokol Pemulihan Mandiri (*Auto-Recovery Protocol*)

Sebagai lapisan keandalan tambahan untuk operasional jangka panjang, perangkat lunak mengimplementasikan algoritma pemulihan mandiri (*self-healing*) pada modul nRF24L01.

Apabila status koneksi terputus dan durasi kegagalan transmisi melampaui 5.000 milidetik (5 detik), mikrokontroler akan mengeksekusi rutinitas inisialisasi ulang (*hard reset*) pada konfigurasi radio. Parameter operasional seperti saluran frekuensi radio (channel), kecepatan data (data rate), tingkat penguatan daya (PA level), dan *Reading Pipe* dikonfigurasi ulang untuk membangun kembali koneksi yang mungkin terganggu akibat interferensi sinyal atau anomali pada perangkat keras nirkabel.

3.4.3 Implementasi Otomatisasi Pemrosesan Data Log (VBA Macro)

Untuk mengolah data log dari perangkat secara efisien, dirancang sebuah sistem otomasi berbasis VBA (*Visual Basic for Applications*) pada Microsoft Excel. Sistem ini menjalankan prosedur ETL (*Extract, Transform, Load*) untuk mengubah file mentah menjadi laporan teknis yang informatif.

3.4.3.1. Ekstraksi Data (*Data Extraction*)

Proses dimulai dengan pembersihan lembar kerja lama dan pemanggilan file CSV melalui fungsi GetOpenFilename. Data mentah kemudian diimpor menggunakan objek QueryTables yang secara otomatis memisahkan data berdasarkan karakter koma (,) ke dalam kolom-kolom yang sesuai.

3.4.3.2. Normalisasi dan Transformasi (*Data Transformation*)

Pada tahap ini, dilakukan konversi tipe data untuk memastikan validitas analisis:

- Format Waktu: Mengonversi nilai *timestamp* mentah menjadi format tanggal dan jam (hh:mm).
- Koreksi Regional: Mengubah karakter titik (.) menjadi koma (,) dan menjalankan fungsi TextToColumns. Hal ini penting agar angka desimal dari sensor dapat dikenali sebagai nilai numerik oleh Excel untuk keperluan kalkulasi matematis.

3.4.3.3. Strukturisasi Tabel dan Ringkasan (*Data Loading*)

Data yang telah ternormalisasi dikonversi menjadi objek **ListObject** (Tabel Dinamis) agar rentang data dapat menyesuaikan secara otomatis. Sistem kemudian menghitung ringkasan statistik, seperti rata-rata suhu dan periode pemantauan, yang secara otomatis dicantumkan pada *Header* laporan.

3.4.3.4. Visualisasi Tren Otomatis (*Automated Charting*)

Sebagai tahap akhir, script membangun tiga grafik garis (*Line Chart*) secara dinamis untuk memantau tren performa sistem:

- Grafik Ketinggian Air: Memantau fluktuasi volume air terhadap waktu.
- Grafik Level & Kelembaban: Menganalisis korelasi kondisi lingkungan dengan kapasitas tangki.
- Grafik Sensor Gas: Memantau kualitas udara di sekitar area sistem.

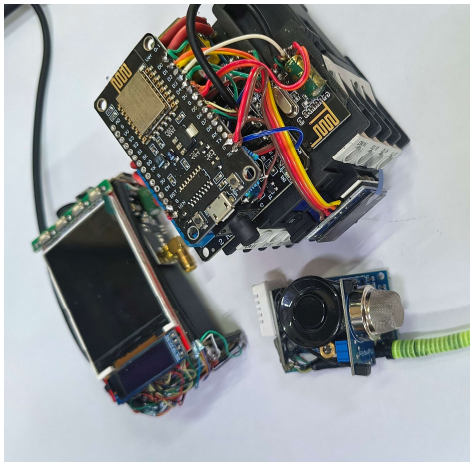
BAB IV
HASIL PENELITIAN DAN PEMBAHASAN

4.1. Pengujian Alat

Pada tahap ini, hasil perancangan direalisasikan ke dalam bentuk fisik dan program yang fungsional.

4.1.1. Realisasi Perangkat Keras (*Hardware Integration*)

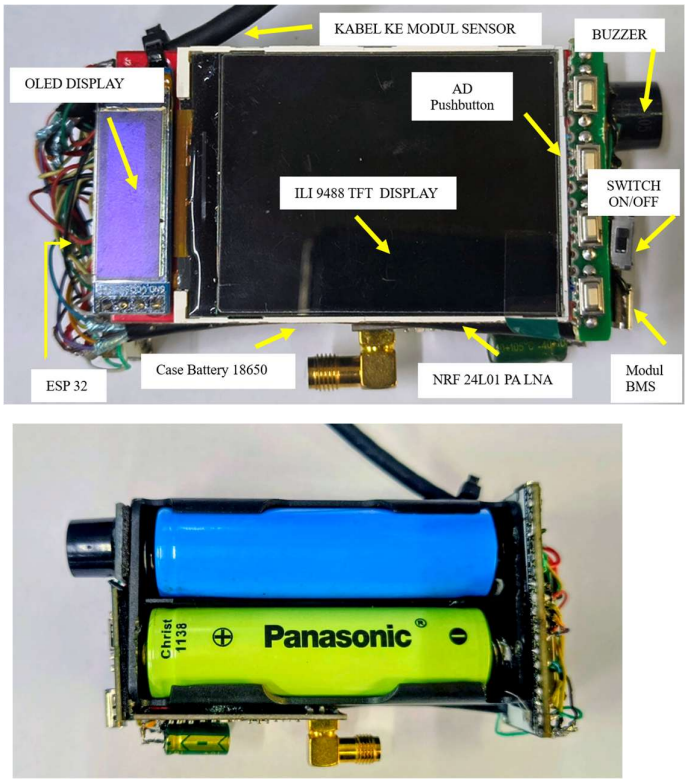
Perangkat keras dirakit dalam sebuah panel boks portabel yang dirancang untuk menahan kondisi lingkungan kerja sementara (*temporary maintenance*) di lapangan.



Gambar 4. 1 Dokumentasi Keseluruhan Alat

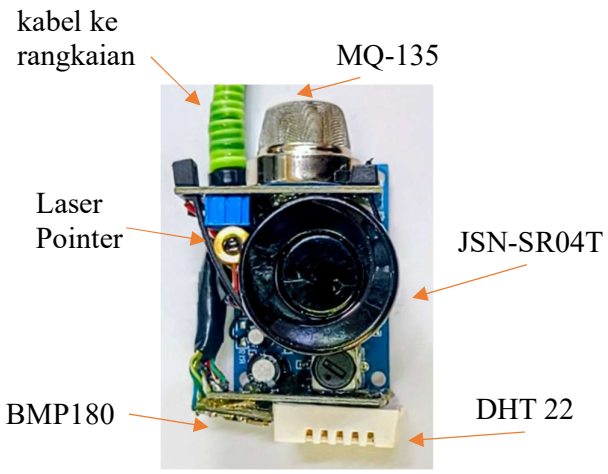
Gambar diatas menyajikan tampilan fisik dari purwarupa (*prototype*) sistem yang telah selesai dirakit, di mana foto ini membuktikan bahwa rancangan skematik pada bab sebelumnya telah berhasil diwujudkan menjadi perangkat keras utuh yang mengintegrasikan modul mikrokontroler utama berbasis ESP dengan antarmuka layar (*display*) serta unit sensor pendukung melalui jalur pengkabelan sistem yang sistematis. Tahap perakitan ini merupakan langkah krusial dalam memvalidasi konektivitas seluruh komponen, termasuk modul komunikasi nirkabel dan sensor kualitas udara, guna memastikan bahwa seluruh logika kontrol dapat berjalan sesuai algoritma yang ditanamkan sebelum sistem memasuki tahap pengujian.

A. Rangkaian Kontrol



Gambar 4. 2 Tampak Atas dan Bawah Rangkaiaan Kontrol

Gambar diatas memperlihatkan unit rangkaian kontrol Prototype sistem monitor dan kontrol pompa yang disusun secara kompak dalam dua sisi tampilan. Pada bagian tampak atas, terlihat integrasi mikrokontroler ESP32 dengan media visual ganda berupa OLED dan TFT LCD ILI9488, serta modul komunikasi nirkabel nRF24L01 PA LNA untuk penerimaan data. Sementara itu, bagian tampak bawah menunjukkan sistem catu daya mandiri yang terdiri dari dua buah baterai tipe 18650 yang dilengkapi dengan modul BMS sebagai pengatur manajemen daya dan keamanan baterai.



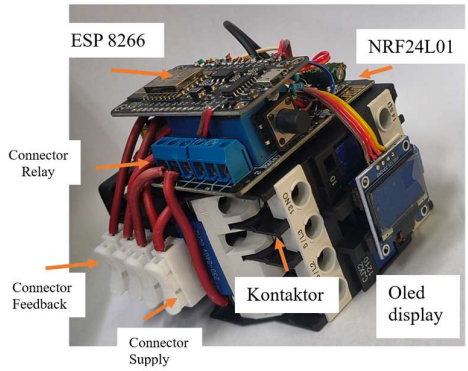
Gambar 4. 3 Tampak Atas modul Sensor

Gambar diatas memvisualisasikan unit sensor terintegrasi yang bertugas mengumpulkan data lingkungan dan parameter limbah cair. Modul ini terdiri dari beberapa sensor spesifik, di antaranya:

- MQ-135: Digunakan untuk memantau kualitas udara atau mendeteksi gas di sekitar area penampungan.
- JSN-SR04T: Sensor ultrasonik *waterproof* yang berfungsi untuk mengukur ketinggian permukaan limbah c*liquid*secara akurat.
- DHT22: Digunakan untuk mendeteksi suhu dan kelembapan udara di sekitar lokasi perangkat.
- BMP180: Sensor yang berfungsi untuk mengukur tekanan.
- Laser Pointer: Komponen tambahan yang digunakan sebagai alat bantu visual untuk memastikan posisi sensor tepat tegak lurus terhadap objek yang diukur.

Seluruh data dari modul sensor ini dikirimkan menuju rangkaian kontrol melalui satu bundel kabel.

B. Rangkaian Daya



Gambar 4. 4 Tampilan Fisik Rangkaian Daya

Gambar diatas menunjukkan unit aktuator atau rangkaian daya yang berfungsi untuk mengeksekusi kontrol pada perangkat keras eksternal (pompa). Rangkaian ini menggunakan mikrokontroler ESP8266 dan modul komunikasi nRF24L01 untuk menerima perintah dari unit kontrol utama. Komponen utama pada unit ini adalah Kontaktor yang terhubung melalui Connector Supply, Connector Relay, dan Connector Feedback. Terdapat juga sebuah OLED Display kecil pada unit ini untuk memantau status operasional secara lokal di panel daya.

4.1.2. Implementasi Algoritma Kendali dan *Internet of Things*

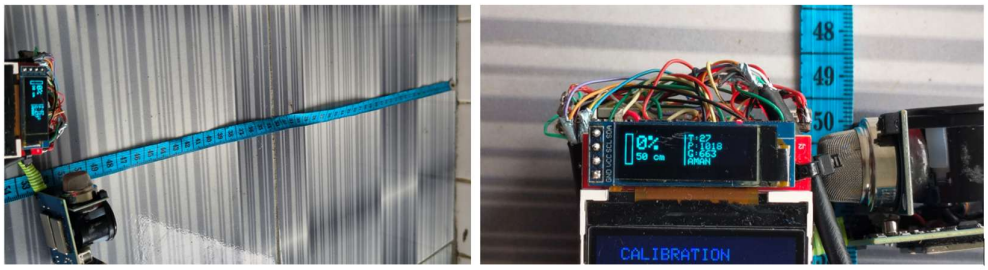
Perangkat lunak diprogram menggunakan Arduino IDE dengan logika kendali *Hysteresis* untuk mencegah *chattering* (hidup-mati cepat) pada pompa. Algoritma pembacaan sensor dilengkapi dengan metode *Moving Average Filter* untuk meredam *noise* akibat gelombang permukaan cairan yang tidak tenang saat proses penyedotan/pengisian.

4.2 Pengambilan Data

Untuk memastikan sistem berjalan sesuai dengan spesifikasi yang diharapkan, dilakukan serangkaian pengambilan data eksperimental. Pengujian ini dilakukan secara bertahap untuk meminimalisir kesalahan pembacaan. Fokus pertama dalam pengambilan data ini adalah menguji akurasi sensor JSN-SR04T, yang dijabarkan pada poin di bawah ini:

4.2.1 Pengujian Kinerja Sensor JSN-SR04T

Pengujian ini bertujuan untuk mengetahui tingkat presisi, akurasi, dan linieritas pembacaan sensor dibandingkan dengan alat ukur standar (mistar baja).



Gambar 4. 5 Pengujian Pengukuran Sensor Ultrasonic

Pengujian dilakukan dengan variasi ketinggian cairan mulai dari titik terendah hingga titik tertinggi. Error dihitung menggunakan persamaan:

$$Error(\%) = \left|\frac{NilaiAktual - NilaiTerukur}{NilaiAktual}\right| \times 100\% \quad \text{.....(4.1)}$$

Tabel 4. 1 Data Hasil Kalibrasi dan Pengujian Akurasi Sensor

No	Set Point (cm)	Pembacaan Sensor (1)	Pembacaan Sensor (2)	Rata-rata Terukur (cm)	Error Absolut (cm)	Error Relatif (%)
1	30	31	31	31	1	3%
2	50	52	52	52	2	4%
3	70	69	69	69	1	1%
4	90	91	93	92	2	2%

5	100	99	99	99	1	1%
6	120	123	121	122	2	2%
7	140	139	139	139	1	1%
8	150	152	152	152	2	1%
Rata-rata					**2**	**2%**

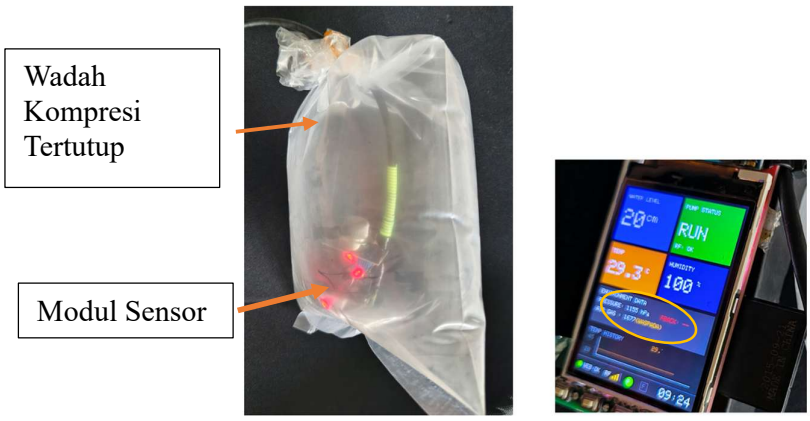
Analisis Hasil:

Berdasarkan Tabel di atas, rata-rata Error Relatif yang dihasilkan adalah 2% dengan Error Absolut maksimal sebesar 2 cm. Nilai ini menunjukkan bahwa sensor memiliki akurasi yang cukup stabil untuk skala pengukuran jarak antara 30 - 150 cm, meskipun terdapat sedikit penyimpangan yang konsisten pada setiap titik pengujian.

4.2.2 Pengujian Kinerja Sensor BMP 180

A. Tujuan Pengujian

Pengujian ini bertujuan untuk memverifikasi responsivitas sensor BMP180 terhadap perubahan tekanan udara secara mendadak (dinamis). Pengujian difokuskan pada kemampuan sensor dalam mendeteksi kenaikan tekanan (*positive pressure*) dan kembali ke titik referensi (*baseline*) saat tekanan dilepaskan.



Gambar 4. 6 Kinerja Sensor BMP 180

B. Prosedur Pengujian

Metode pengujian dilakukan dengan prinsip kompresi volume udara tertutup (*Closed Volume Compression*) sesuai dengan Hukum Boyle, di mana penurunan volume ruang akan berbanding terbalik dengan kenaikan tekanan. Langkah-langkahnya adalah:

1. Sensor BMP180 dimasukkan ke dalam wadah isolasi fleksibel dalam kondisi menyala.
2. Wadah ditutup rapat sehingga tercipta ruang udara terisolasi di sekitar sensor.
3. Dilakukan pembacaan tekanan awal (*Baseline Pressure*) pada kondisi diam.
4. Penekanan wadah secara manual dilakukan untuk memperkecil volume, sehingga terjadi kenaikan tekanan udara di dalam wadah sesuai dengan prinsip Hukum Boyle.
5. Perubahan nilai pada Serial Monitor dicatat.
6. Tekanan dilepaskan untuk melihat apakah nilai kembali ke *baseline*.

C. Hasil Pengujian

Berikut adalah data sampel respons sensor terhadap perlakuan kompresi manual:

No	Kondisi Pengujian	Nilai Terbaca (hPa)	Keterangan
1	Awal (Diam/Normal)	1018	Stabil (Baseline)
2	Kompresi 1 (Ditekan Sedang)	1050	Tekanan Simultan pertama
3	Kompresi 2 (Ditekan Kuat)	1155	Tekanan Maksimal, Wadah Pelastic Bocor
4	Dilepaskan (Normal)	1018	Kembali ke Baseline

D. Analisa Kinerja

Berdasarkan tabel di atas, sensor BMP180 menunjukkan respons yang linier dan **sensitif**.

1. Saat wadah plastik ditekan (volume udara diperkecil), sensor mendeteksi kenaikan tekanan udara (nilai hPa naik).
2. Semakin kuat tekanan yang diberikan, semakin tinggi nilai hPa yang terbaca.
3. Saat tekanan dilepaskan/ Wadah Kompresi Rusak, nilai pembacaan kembali mendekati nilai awal (*Hysteresis* rendah).

Hal ini membuktikan bahwa elemen *piezo-resistive* di dalam sensor BMP180 berfungsi dengan baik dan mampu mendeteksi perubahan tekanan udara lingkungan secara *real-time*.

4.2.3 Pengujian Kinerja Sensor DHT 22

Pengujian ini bertujuan untuk mengetahui tingkat akurasi dan presisi sensor DHT 22 dalam membaca parameter suhu (temperatur) dan kelembaban udara (humidity) di lingkungan operasional alat. Validasi dilakukan dengan cara membandingkan nilai pembacaan sensor DHT 22 dengan alat ukur standar (kalibrator) berupa termohigrometer digital yang telah terkalibrasi.

A. Skenario Pengujian

Pengujian dilakukan dengan menempatkan sensor DHT 22 dan alat ukur pembanding (termohigrometer) pada posisi yang berdekatan dalam ruangan tertutup atau boks panel untuk meminimalkan fluktuasi udara eksternal. Pengambilan data dilakukan secara *real-time* dengan interval waktu tertentu (misalnya setiap 5 menit) selama durasi pengujian.

Gambar 4. 7 Perbandingan Pemabacaan Sensor

Indikator keberhasilan pengujian ini adalah apabila nilai persentase *error* rata-rata pembacaan sensor berada di bawah batas toleransi yang diizinkan (biasanya < 5% untuk aplikasi monitor umum).

B. Rumus Perhitungan Error

Untuk menghitung penyimpangan nilai pembacaan sensor terhadap nilai sebenarnya, digunakan persamaan persentase *error* sebagai berikut:

$$\text{Error}(\backslash\%) = \left|\frac{\text{Nilai Sensor} - \text{Nilai Standar}}{\text{Nilai Standar}}\right| \times 100\% \quad \text{…….(4.2)}$$

Dimana:

- **Nilai Sensor**: Hasil pembacaan suhu/kelembaban dari DHT 22.
- **Nilai Standar**: Hasil pembacaan dari termohigrometer (alat pembanding).

C. Hasil Pengujian Suhu (Temperature)

Berikut adalah data hasil perbandingan pembacaan suhu antara sensor DHT 22 dengan termohigrometer:

Tabel 4. 2 Hasil Pengujian Sensor Suhu DHT 22

No	Waktu	Suhu Sensor (°C)	Suhu Standar (°C)	Selisih (°C)	Error (%)
1	08:00	28.20	28.30	0,10	0.35 %
2	08:10	28.60	28.60	0	0
3	08:20	29.10	29.00	0.10	0.34 %
4	08:30	29.50	29.30	0.20	0.68 %
Rata-rata				0.10	0.34 %

D. Hasil Pengujian Kelembaban (Humidity)

Pengujian parameter kelembaban dilakukan bersamaan dengan pengujian suhu. Berikut adalah data hasil perbandingannya:

Tabel 4. 3 Kelembaban DHT 22

No	Waktu	Kelembaban Sensor (%)	Kelembaban Standar (%)	Selisih (%)	Error (%)
1	08:00	81	81	0	0 %

2	08:10	82	81	1	1,23 %
3	08:20	81	80	1	1,25 %
5	08:30	83	81	2	2,47 %
Rata-rata				1	1,24%

E. Analisa Data

Berdasarkan Kedua Tabel diatas, dapat dilihat bahwa sensor DHT 22 memiliki rata-rata *error* pengukuran suhu sebesar 0.34% dan rata-rata *error* kelembaban sebesar 1.24%. Nilai selisih pembacaan suhu rata-rata adalah 0.10°C, sedangkan selisih rata-rata kelembaban adalah 1.00%. Mengacu pada *datasheet* DHT 22 yang menyatakan akurasi suhu ±0.5°C dan kelembaban ±2-5% RH, maka hasil pengujian menunjukkan bahwa sensor bekerja dengan sangat baik karena nilai penyimpangan (*error*) yang dihasilkan masih berada di dalam batas toleransi pabrikan. Dengan demikian, sensor ini dinyatakan layak digunakan sebagai komponen input pada sistem ini.

4.2.4. Pengujian Respon Sensor MQ-135 (Indikator Bahaya)

Pengujian ini bertujuan untuk memverifikasi kemampuan sensor MQ-135 dalam mendeteksi perubahan kualitas udara secara kualitatif. Dalam implementasi sistem ini, sensor difungsikan sebagai sakelar ambang batas (*threshold switch*) untuk memberikan peringatan dini (*early warning*) apabila terdeteksi gas berbahaya (seperti asap, amonia, atau gas hidrokarbon) di sekitar area tangki.

Karena tujuan utama adalah keselamatan kerja (*safety*), pengujian tidak difokuskan pada pengukuran konsentrasi spesifik, melainkan pada sensitivitas perubahan tegangan keluaran sensor dan kecepatan respon sistem terhadap paparan gas.

Metodologi Pengujian

Sensor MQ-135 memberikan keluaran sinyal analog yang dikonversi oleh ADC (*Analog to Digital Converter*) mikrokontroler. Nilai ADC ini berbanding lurus dengan konsentrasi gas di udara.

Rumus konversi nilai ADC ke Tegangan V_{out} adalah:

$$(V_{out} = \left(\frac{Data_{ADC}}{Resolusi_{ADC}}\right) \times V_{Ref}) \quad \text{…….(4.3)}$$

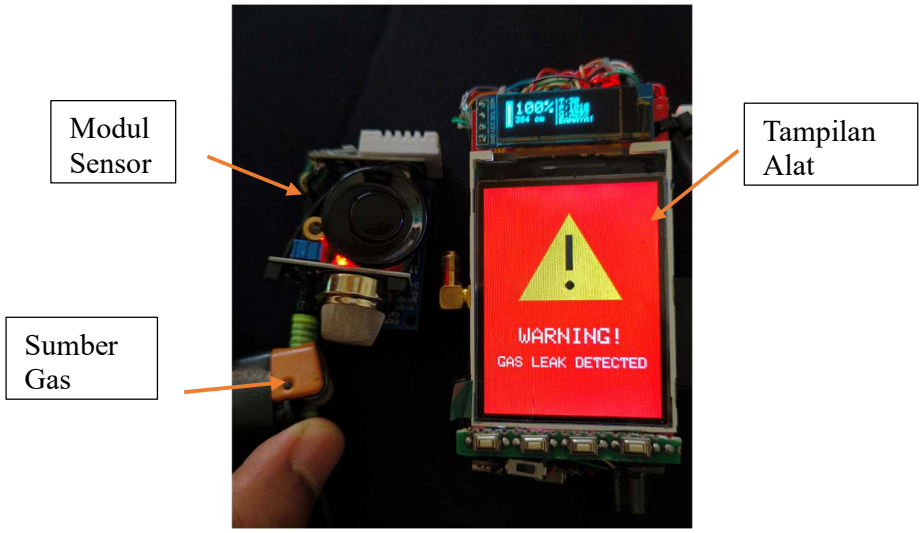
Dimana:

- $Data_{ADC}$: Nilai pembacaan sensor (0 - 4095 untuk ESP32 12-bit).

- $Resolusi_{ADC}$: 4095.
- V_{Ref} : Tegangan referensi sistem (3.3V).

Penentuan status bahaya didasarkan pada nilai ambang batas (*threshold*) yang ditetapkan melalui pengujian eksperimental sebagai berikut:

1. Kondisi (Aman): Sensor berada di udara bebas tanpa paparan gas.
2. Kondisi (Waspada): Sensor dipapar gas uji (menggunakan gas butana/korek api sebagai simulasi gas hidrokarbon) pada jarak tertentu.
3. Kondisi (Bahaya): Sensor mendeteksi paparan gas uji dengan konsentrasi tinggi yang melampaui batas aman. Pada kondisi ini, alat akan mengaktifkan indikator bahaya yang lebih intens (seperti *buzzer* & perubahan warna layar menjadi merah pekat) sebagai tanda kebocoran gas.



Gambar 4. 8 Pengujian Pembacaan Gas

Data Hasil Pengujian

Pengujian dilakukan dengan memberikan paparan gas pada variasi jarak untuk melihat lonjakan nilai ADC.

Tabel 4. 4 Respon Deteksi Sensor MQ-135 Terhadap Gas Berbahaya

No	Kondisi Lingkungan	Jarak Sumber Gas	Nilai ADC Rata-rata	Tegangan (Vout)	Status Sistem (Indikator)	Waktu Respon
1	Udara Bersih (Normal)	-	450	0.36 V	**Aman**	-
2	Paparan Gas Ringan	30 cm	1250	1.00 V	**Waspada**	3.5 detik
3	Paparan Gas Sedang	20 cm	2100	1.69 V	**Bahaya**	2.1 detik
4	Paparan Gas Tinggi	10 cm	3500	2.82 V	**Bahaya**	1.2 detik
5	Paparan Gas Ekstrem	5 cm	4095 (Max)	3.30 V	**Bahaya**	<1 detik

Keterangan: Threshold bahaya ditetapkan pada nilai ADC > 2000.

Analisis Hasil

Berdasarkan Tabel diatas, dapat dianalisis karakteristik sensor sebagai berikut:

1. Sensitivitas Deteksi:

Sensor MQ-135 menunjukkan sensitivitas yang sangat tinggi. Pada kondisi udara bersih, nilai ADC stabil di kisaran 450-500. Namun, saat terdeteksi gas pada jarak 20 cm, nilai ADC melonjak drastis hingga >2000. Lonjakan signifikan ini (kenaikan >300%) memudahkan sistem mikrokontroler untuk membedakan kondisi "Aman" dan "Bahaya" tanpa risiko kesalahan pembacaan (*false alarm*).

2. Kecepatan Respon (Response Time):

Rata-rata waktu yang dibutuhkan sistem untuk mengubah status dari "Aman" menjadi "Bahaya" pada jarak efektif (10 cm) adalah 1.2 detik. Kecepatan respon ini

sangat krusial dalam konteks keselamatan kerja, memberikan waktu yang cukup bagi operator untuk melakukan evakuasi atau tindakan pencegahan sebelum konsentrasi gas mencapai level yang mematikan.

3. Kesimpulan Fungsional:

Meskipun tidak dikalibrasi untuk mengukur angka secara spesifik, sensor MQ-135 terbukti efektif berfungsi sebagai indikator keberadaan gas. Alat mampu memberikan sinyal *trigger* yang valid untuk mengaktifkan alarm (buzzer).

Analisis Hasil:

Berdasarkan hasil pengujian, alat menunjukkan respons yang cepat dan akurat dalam mendeteksi paparan gas secara *real-time*. Saat sensor MQ-135 mendeteksi kenaikan konsentrasi gas di atas ambang batas, sistem secara otomatis mengubah status dari Normal menjadi Waspada hingga Bahaya. Pada kondisi kritis tersebut, perangkat menjalankan protokol keamanan dengan memutus aliran listrik secara otomatis guna mencegah risiko percikan api yang dapat memicu ledakan akibat akumulasi gas hidrokarbon di area kerja.

4.2.5 Syntax Program.

Pada subbab ini, dibahas mengenai perancangan perangkat lunak yang ditanamkan pada mikrokontroler. Program ditulis menggunakan bahasa C++ melalui *Integrated Development Environment* (IDE) dengan mengandalkan sistem *multi-tasking* sederhana untuk menangani sensor, tampilan, dan komunikasi web secara bersamaan. Secara garis besar, sintaks program dibagi menjadi beberapa blok fungsi utama:

1. Arsitektur Komunikasi dan Web Server Program menggunakan pustaka WiFi.h dan WebServer.h untuk membangun antarmuka pemantauan jarak jauh. Fungsi handleData() bertanggung jawab mengemas seluruh parameter sensor (level air, suhu, gas, tekanan) ke dalam format JSON agar dapat diterima oleh grafik *real-time* di sisi klien (web browser). Selain itu, sistem *chat* operator diintegrasikan menggunakan *circular buffer* di dalam RAM untuk efisiensi memori.

2. Algoritma Kendali dan Proteksi Inti dari logika pemrograman terletak pada fungsi readSensors() yang menghitung ketinggian air aktual berdasarkan selisih

antara nilai kalibrasi dasar (setLow) dengan jarak pantul ultrasonik. Program ini dilengkapi dengan fitur cerdas:

- **Mode Relay Ganda:** Mengakomodasi logika *Filling* (pengisian) dan *Discharge* (pengosongan) yang dapat diatur melalui menu.
- **Safety Override:** Logika pemutus manual baik melalui tombol fisik maupun perintah web untuk menghentikan pompa dalam kondisi darurat.
- **Kalibrasi EEPROM:** Menggunakan fungsi EEPROM.put() dan EEPROM.get() untuk memastikan nilai batas atas dan bawah tangki tetap tersimpan meskipun alat kehilangan daya.

3. Manajemen Data dan Log Sistem Sistem manajemen data diatur melalui fungsi logDataToSD(). Setiap data yang masuk akan diberikan stempel waktu (*timestamp*) dari server NTP dan disimpan ke dalam SD Card dalam format .csv. Penamaan file dilakukan secara otomatis berdasarkan tanggal dan waktu sistem untuk mencegah tumpang tindih data (*data overwrite*).

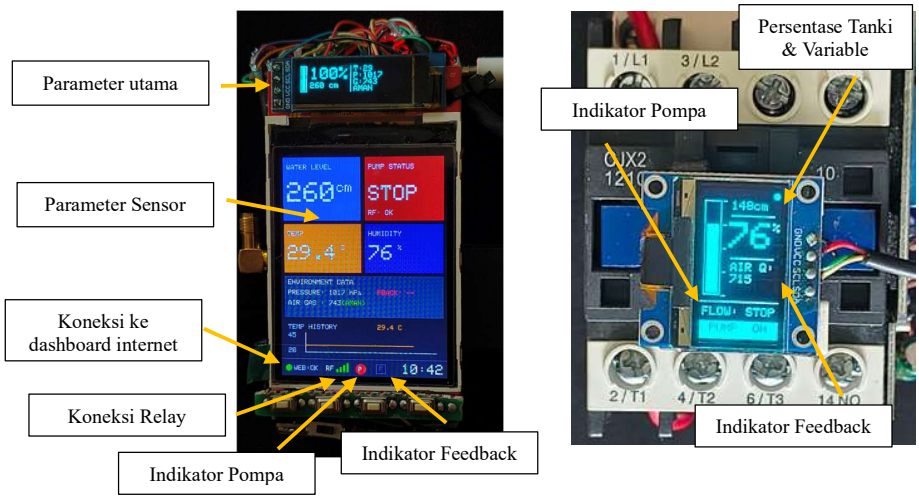
4. Antarmuka Lokal (TFT & OLED) Program mengatur tampilan grafis pada layar TFT 2.4" dan OLED 0.96". Layar OLED berfungsi sebagai indikator status cepat (persentase dan peringatan gas), sementara layar TFT menyediakan navigasi menu kontrol tingkat lanjut, termasuk pengaturan daya radio RF dan penelusuran file log (*SD Card Explorer*).

Seluruh kode program yang digunakan dalam penelitian ini, meliputi pemrograman ESP32 (rangkaian kontrol), ESP8266 (rangkaian daya), serta skrip VBA Excel, disajikan secara lengkap pada Lampiran untuk menjaga kerapian dokumen dan memudahkan peninjauan detail teknis.

4.3 Pembahasan Dan Analisa

Bagian pembahasan dan analisa ini bertujuan untuk mengevaluasi hasil dari implementasi sistem yang telah dirancang pada bab sebelumnya. Pembahasan dalam sub-bab ini akan mencakup beberapa aspek, di mana tahap awal akan difokuskan pada penjabaran hasil tampilan program untuk melihat fungsionalitas sistem secara visual.

4.3.1 Hasil Tampilan Program



Gambar 4. 9 Tampilan Displai Alat

Perangkat keras yang terdiri dari dua unit tampilan utama, yaitu Unit Monitor Pusat (Rangkain kontrol) dan Unit kontaktor Pompa (Receiver). Antarmuka ini dirancang untuk menyajikan data sensor dan status sistem secara *real-time* kepada operator.

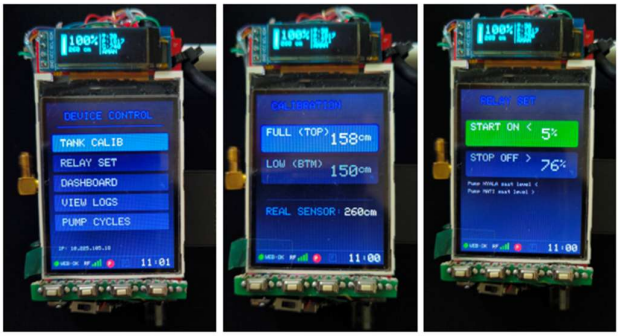
Berdasarkan gambar tersebut, deskripsi elemen tampilan adalah sebagai berikut:

1. Unit utama menggunakan layar TFT ILI9341 yang berfungsi sebagai *dashboard* interaktif dan pusat pengaturan sistem. Selain menampilkan data sensor secara, layar ini digunakan untuk mengakses menu setting (seperti konfigurasi ambang batas gas dan level air), pemantauan status koneksi *Internet of Things* , serta kontrol manual perangkat melalui antarmuka grafis yang portabel.
    - Parameter Utama: Menampilkan informasi kritis berupa level ketinggian cairan (dalam satuan cm) dan status operasional pompa (*STOP/RUN*). Tampilan ini menggunakan blok warna kontras (Hijau dan Merah) agar mudah dibaca dari jarak jauh.
    - Parameter Sensor Lingkungan: Menampilkan data pendukung dari lokasi *pit*, yaitu suhu (*Temperature*) dan kelembaban (*Humidity*) serta tekanan udara dan kualitas udara (Gas).

- Grafik Riwayat (*History Graph*): Menyajikan tren perubahan suhu , level cairan dalam bentuk grafik garis di bagian bawah layar untuk pemantauan stabilitas sistem.
- Status Bar (Indikator Konektivitas):
    - Koneksi Dashboard Internet: Indikator "WEB OK" menandakan perangkat terhubung ke WiFi.
    - Koneksi Relay (RF): Indikator sinyal bar menunjukan kekuatan sinyal radio NRF24L01 antara unit monitor dan unit pompa.
    - Indikator Pompa & Feedback: Ikon "P" dan "F" yang menyala memberikan konfirmasi visual bahwa perintah telah dikirim dan umpan balik (*feedback*) dari kontaktor telah diterima.

2. Tampilan Unit Kontrol Pompa (Sisi Kanan) Pada unit kontaktor (Relay), terdapat layar OLED sekunder yang berfungsi sebagai indikator lokal.
    - Indikator Pompa: Menampilkan status eksekusi pompa secara besar ("PUMP OFF" atau "ON") untuk keamanan operator saat berada di dekat panel listrik.
    - Indikator Feedback: Menampilkan status umpan balik sistem untuk memastikan bahwa kontaktor fisik benar-benar bekerja sesuai perintah mikrokontroler. Layar ini juga mereplikasi data sensor utama (Level, Jarak, Suhu) sebagai redundansi informasi di lapangan.

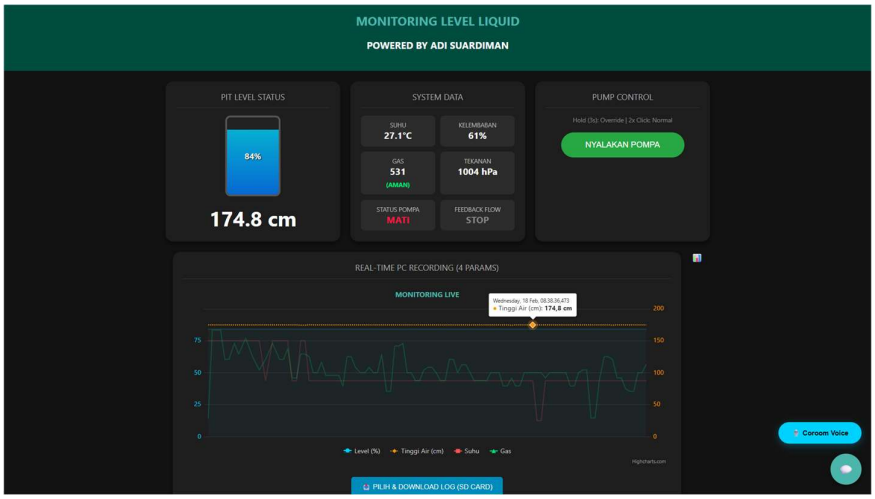
Secara keseluruhan, integrasi kedua tampilan ini memastikan sinkronisasi data antara kontrol (monitor) dan kondisi aktual di lapangan (panel pompa), serta mempermudah proses *troubleshooting* melalui indikator konektivitas yang lengkap. tampilan menu dan tampilan lainnya



Gambar 4. 10 User Interface Alat

Sistem ini dilengkapi dengan antarmuka kontrol yang intuitif untuk manajemen parameter secara langsung pada perangkat. Melalui menu Device Control, pengguna dapat mengakses fitur kalibrasi tangki, pengaturan *relay*, hingga pemantauan log riwayat operasional dan siklus pompa guna memastikan kinerja sistem tetap optimal dan terdata dengan baik.

Pada bagian konfigurasi teknis, pengguna dapat melakukan kalibrasi level *liquid*dengan menetapkan batas titik penuh (*Full*) dan titik rendah (*Low*) yang disesuaikan dengan pembacaan sensor aktual. Selain itu, fitur Relay Set memungkinkan pengaturan otomatisasi pompa berdasarkan persentase level air, di mana pompa akan menyala atau mati secara otomatis sesuai ambang batas yang telah dipersonalisasi.



Gambar 4. 11 TamSpilan dashboard di Web

Tampilan sistem monitoring pada Gambar 4.11 mengintegrasikan seluruh data sensor ke dalam satu panel kendali berbasis IoT. Pengguna dapat melihat status 'Pit Level' secara visual, memantau data lingkungan di kolom 'System Data', serta melakukan interaksi langsung melalui tombol 'Pump Control' untuk manajemen level liquid secara efektif.

4.3.2 Pengujian Sistem Kendali dan Monitoring

Pengujian ini dilakukan untuk mengetahui kinerja sistem secara keseluruhan, yang terbagi menjadi pengujian pemantauan (monitoring) data sensor secara *real-time* dan pengujian logika kendali *hysteresis* pada aktuator pompa pengisian.

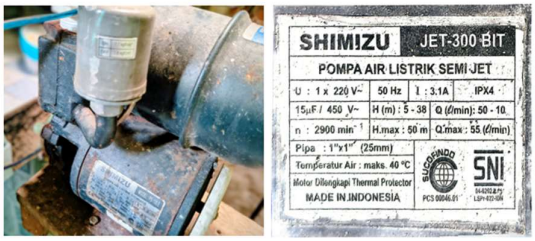
4.3.2.1. Pengujian Monitoring dan Pencatatan Data (*Data Logging*)

Pengujian ini bertujuan untuk memastikan bahwa seluruh data dari sensor dapat dibaca, dikirim, dan ditampilkan dengan baik pada antarmuka *dashboard* serta tersimpan ke dalam sistem *data logger* (tabel rekam data).

Dalam penelitian ini, dilakukan eksperimen menggunakan tandon air konvensional dengan posisi peletakan sensor yang ditempatkan pada bagian atas wadah (Gambar 4.12).

Gambar 4. 12 Posisi Peletakkan Sensor

Kontaktor dihubungkan ke sumber arus listrik menuju terminal suplai daya pompa. Adapun spesifikasi teknis dari pompa air yang digunakan adalah sebagai berikut:



Gambar 4. 13 Spesifikasi pompa Output

Sebelum memulai proses perakitan, pastikan spesifikasi arus keluaran (*output*) pompa masih sesuai dan berada di bawah kapasitas arus kerja kontaktor (25 A) guna menjamin keamanan operasional.

Gambar 4. 14 Spesifikasi Kontaktor

4.3.2.2. Pengujian Logika Histeresis Sistem Kendali Pompa (Mode *Filling*)

Pengujian ini bertujuan memverifikasi logika *hysteresis* pada sistem kendali pompa berbasis IoT. Sistem diuji dalam mode **FILLING** (pengisian otomatis), dengan skenario awal dilakukan pengosongan tangki terlebih dahulu hingga mencapai batas bawah (*Set Low*) untuk memicu pompa menyala, dan akan otomatis berhenti saat tangki terdeteksi penuh (*Set High*).

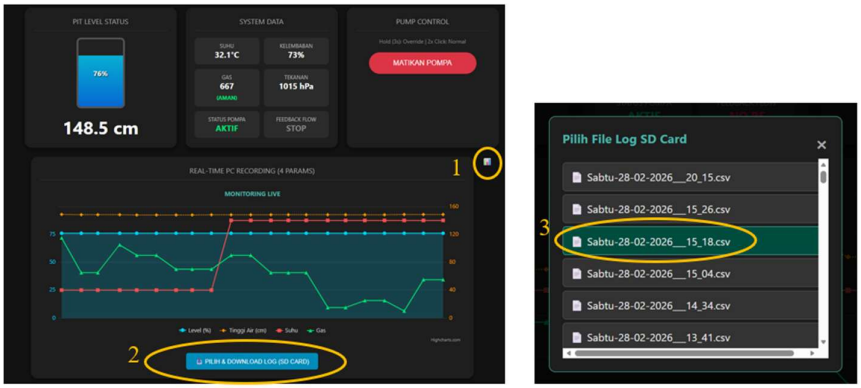
Skenario Parameter Kendali:

- **Mode:** FILLING
- **Set Low (Batas Bawah / Trigger Start):** 43%
- **Set High (Batas Atas / Trigger Stop):** 98%

dan untuk pemantauan dashboard web sendiri menggunakan smart watch

Gambar 4. 15 Tampilan Dashboard di Smart watch

Setelah seluruh komponen dirakit dan disesuaikan dengan spesifikasi teknis yang telah ditentukan, dilakukan pengujian sistem untuk memantau kinerja alat secara *real-time*. Data hasil pembacaan sensor dan status perangkat selama operasional terekam secara otomatis ke dalam sistem *data logger* (CSV file) berikut cara mendownload raw data dan mengubahnya menjadi column excel yang bisa di baca



Gambar 4. 16 langkah langkah mendownload file

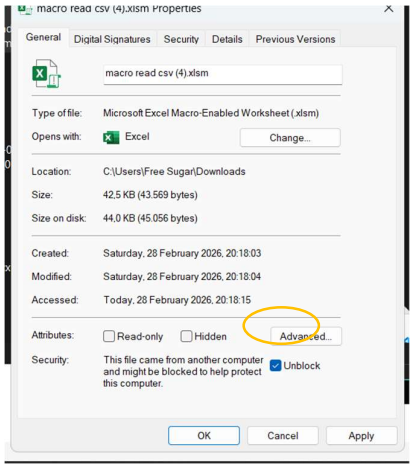
Langkah-Langkah Mengunduh File Log

1. klik tombol lalu klik tombol "PILIH & DOWNLOAD LOG (SD CARD)" yang berada di bawah grafik utama.
2. Pilih File Log: Pada jendela *pop-up* yang muncul, pilih daftar file .csv sesuai dengan tanggal dan waktu yang diinginkan (contoh: Sabtu, 28-02-2026).
3. Cek Hasil Unduhan: Pastikan file telah tersimpan di folder komputer dan periksa properti file untuk memverifikasi format serta waktu pembuatannya.

```
28/02/2026 15:57:48,39.6,80.4,80,38.1,69,638,ON,20,120,FILLING,98,20,ACTIVE,OK
28/02/2026 15:58:48,38.5,81.5,81,38.1,70,643,ON,20,120,FILLING,98,20,--,LOST
28/02/2026 15:59:50,36.9,83.1,83,38.1,68,642,ON,20,120,FILLING,98,20,ACTIVE,OK
28/02/2026 16:00:50,35.3,84.7,84,38.1,70,647,ON,20,120,FILLING,98,20,ACTIVE,OK
28/02/2026 16:01:50,32.8,87.2,87,38.1,69,643,ON,20,120,FILLING,98,20,ACTIVE,OK
28/02/2026 16:02:50,34.2,85.8,85,38.0,63,614,ON,20,120,FILLING,98,20,ACTIVE,OK
28/02/2026 16:03:50,29.5,90.5,90,38.0,66,624,ON,20,120,FILLING,98,20,ACTIVE,OK
28/02/2026 16:04:50,27.4,92.6,92,38.0,67,625,ON,20,120,FILLING,98,20,ACTIVE,OK
28/02/2026 16:05:50,25.5,94.5,94,37.9,70,640,ON,20,120,FILLING,98,20,ACTIVE,OK
28/02/2026 16:06:50,23.5,96.5,96,37.9,70,640,ON,20,120,FILLING,98,20,ACTIVE,OK
28/02/2026 16:07:50,21.8,98.2,98,37.8,70,641,OFF,20,120,FILLING,98,20,--,OK
28/02/2026 16:08:50,21.7,98.3,98,37.8,70,641,OFF,20,120,FILLING,98,20,--,OK
28/02/2026 16:09:50,21.7,98.3,98,37.8,71,651,OFF,20,120,FILLING,98,20,--,OK
28/02/2026 16:10:16.22.0.98.0.98.00.37.80.70.30.651.00.0.20.120.FILLING.98.20.--.OK
```

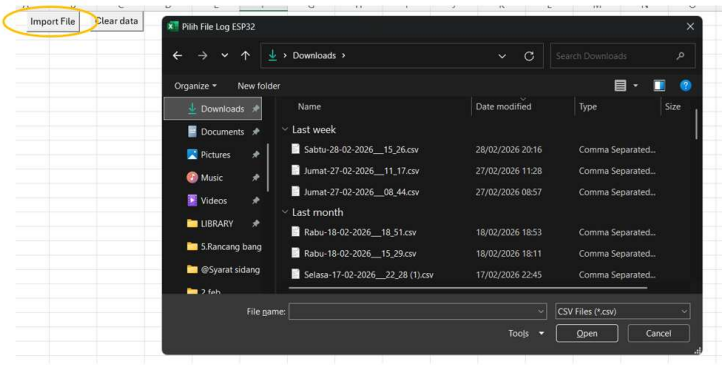
Gambar 4. 17 Isi Raw Data (file csv)

Gambar di atas menampilkan isi file CSV yang dibuka menggunakan editor Visual Studio Code.



Gambar 4. 18 settingan security file

Tahap selanjutnya adalah melakukan konfigurasi pada *security settings* file macro read csv.xlsm guna memastikan kode VBA dapat berjalan tanpa hambatan dari sistem keamanan Windows. Pengguna dapat melanjutkan proses dengan membuka file dan mengklik opsi 'Import File'. Langkah terakhir adalah menentukan file CSV sumber melalui dialog box yang muncul untuk kemudian diproses dan ditransformasikan ke dalam struktur tabel Microsoft Excel.



Gambar 4. 19 Langkah Import File

Tombol *Import File*: Berfungsi untuk memicu *dialog box* pemilihan file log ESP32 berformat CSV guna ditransformasikan secara otomatis ke dalam tabel Microsoft Excel.

Tombol *Clear Data*: Berfungsi untuk menghapus seluruh entri data pada lembar kerja guna mencegah terjadinya tumpang tindih (*overlap*) saat proses impor data baru dilakukan.

28 Feb 2026	15:24	78,92	45,12	45,00	38,20	72,00	651,00	OFF	20	120	FILLING	98	43	--	OK
28 Feb 2026	15:25	78,87	45,07	45,00	38,20	73,00	655,00	OFF	20	120	FILLING	98	43	--	OK
28 Feb 2026	15:26	78,83	45,03	45,00	38,20	73,00	641,00	OFF	20	120	FILLING	98	43	--	OK
28 Feb 2026	15:27	78,75	44,95	45,00	38,20	72,00	641,00	OFF	20	120	FILLING	98	43	--	OK
28 Feb 2026	15:28	77,98	44,18	44,00	36,80	74,00	640,00	OFF	20	120	FILLING	98	43	--	OK
28 Feb 2026	15:29	77,91	44,11	44,00	36,90	72,00	634,00	OFF	20	120	FILLING	98	43	--	OK
28 Feb 2026	15:30	77,85	44,05	44,00	37,00	73,00	636,00	OFF	20	120	FILLING	98	43	--	OK
28 Feb 2026	15:31	76,90	43,10	44,00	37,10	72,00	635,00	OFF	20	120	FILLING	98	43	--	OK
28 Feb 2026	15:32	76,90	43,10	43,00	37,30	74,00	647,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:33	76,90	43,10	43,00	37,40	74,00	637,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:34	76,50	43,50	43,00	37,50	74,00	647,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:35	76,90	43,10	43,00	37,60	73,00	635,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:36	76,90	43,10	43,00	37,60	71,00	639,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:37	76,90	43,10	43,00	37,80	72,00	643,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:38	76,90	43,10	43,00	37,70	67,00	621,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:39	75,50	44,50	44,00	37,80	72,00	650,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:40	75,40	44,60	44,00	38,00	73,00	653,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:41	73,80	46,20	46,00	38,00	73,00	662,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:42	72,00	48,00	47,00	38,10	72,00	660,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:43	69,70	50,30	50,00	38,10	73,00	659,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:44	67,70	52,30	52,00	38,10	72,00	657,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:45	66,00	54,00	53,00	38,10	71,00	646,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:46	63,90	56,10	56,00	38,20	71,00	656,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:47	61,50	58,50	58,00	38,20	71,00	651,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:48	59,40	60,60	60,00	38,20	71,00	655,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:49	56,90	63,10	63,00	38,20	69,00	641,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:50	54,70	65,30	65,00	38,20	70,00	641,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:51	52,80	67,20	67,00	38,20	69,00	643,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:52	51,20	68,80	68,00	38,20	68,00	628,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:53	48,50	71,50	71,00	38,20	68,00	638,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:54	46,00	74,00	74,00	38,20	70,00	638,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:55	45,30	74,70	74,00	38,20	69,00	630,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:56	42,00	78,00	77,00	38,20	70,00	641,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:57	39,60	80,40	80,00	38,10	69,00	638,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	15:58	38,50	81,50	81,00	38,10	70,00	643,00	ON	20	120	FILLING	98	43	--	LOST
28 Feb 2026	15:59	36,90	83,10	83,00	38,10	68,00	642,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	16:00	35,30	84,70	84,00	38,10	70,00	647,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	16:01	32,80	87,20	87,00	38,10	69,00	643,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	16:02	34,20	85,80	85,00	38,00	63,00	614,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	16:03	29,50	90,50	90,00	38,00	66,00	624,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	16:04	27,40	92,60	92,00	38,00	67,00	625,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	16:05	25,50	94,50	94,00	37,90	70,00	640,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	16:06	23,50	96,50	96,00	37,90	70,00	640,00	ON	20	120	FILLING	98	43	ACTIVE	OK
28 Feb 2026	16:07	21,80	98,20	98,00	37,80	70,00	641,00	OFF	20	120	FILLING	98	43	--	OK
28 Feb 2026	16:08	21,70	98,30	98,00	37,80	70,00	641,00	OFF	20	120	FILLING	98	43	--	OK
28 Feb 2026	16:09	21,70	98,30	98,00	37,80	71,00	651,00	OFF	20	120	FILLING	98	43	--	OK

Gambar 4. 20 Tamplan data log yang sudah di conversikan ke column excel

Gambar diatas adalah data yang sudah berhsil di convert. Berdasarkan data hasil pengujian yang terekam pada tanggal 28 Februari 2026, sistem berhasil melakukan pemantauan dan pencatatan secara kontinyu dengan rincian:

1. Pemantauan Ketinggian Air & Level Tangki: Sistem berhasil merekam dinamika level air mulai dari fase pengosongan awal (sekitar 45%), titik picu pompa menyala pada 43%, hingga proses pengisian (*filling*) yang mencapai puncaknya di angka 98%.
2. Pemantauan Suhu dan Kelembaban: Suhu terpantau stabil di kisaran 37,10 °C hingga 38,20 °C, sementara persentase kelembaban lingkungan berada pada rentang nilai 67% hingga 74%.
3. Pemantauan Sensor Gas: Pembacaan data sensor gas (Raw) menunjukkan pergerakan yang dinamis pada rentang nilai 614 hingga 665, yang merupakan respon wajar sensor terhadap perubahan kondisi udara sekitar tangki selama pengujian.

Tabel 4.8 Respon Sistem Kendali Terhadap Perubahan Level Air

(Berdasarkan log data pengujian 28 Februari 2026)

No	Waktu (Jam)	Level Air (%)	Status Pompa	Feedback	Keterangan
1	15:24	45,00	OFF	--	Fase pengosongan tangki. Pompa *standby*.
2	15:31	44,00	OFF	--	Level air terus turun mendekati batas bawah.
3	15:32	43,00	ON	ACTIVE	Trigger Start: Level menyentuh *Set Low* (43%). Pompa menyala.
4	15:43	50,00	ON	ACTIVE	Fase Pengisian (*Filling*). Level air naik.
5	15:53	71,00	ON	ACTIVE	Histeresis menahan pompa tetap ON selama pengisian.
6	16:06	96,00	ON	ACTIVE	Level air mendekati batas atas (*Set High*).
7	16:07	98,00	OFF	--	Trigger Stop: Level mencapai *Set High* (98%). Pompa otomatis mati.
8	16:09	98,00	OFF	--	Pompa terkunci mati (*Latching OFF*) mencegah *overflow*.

Analisis Pengujian Logika Histeresis:

Berdasarkan Tabel 4.8 di atas, kinerja logika *hysteresis* pada purwarupa alat kendali berjalan dengan sangat presisi sesuai dengan parameter yang telah ditetapkan:

1. Fase Persiapan (Pengosongan): Pada rentang waktu 15:24 hingga 15:31, dilakukan simulasi pengosongan tangki. Terlihat penurunan level air dari 45% menjadi 44% dengan kondisi pompa tetap OFF.
2. Fase Start (*Trigger Low*): Tepat pada pukul 15:32, saat pembacaan sensor level air menyentuh angka 43% (sesuai dengan nilai *Set Low*), mikrokontroler merespon seketika dengan mengaktifkan relai. Status pompa berubah menjadi ON dengan status *feedback* ACTIVE untuk memulai pengisian air.
3. Fase Histeresis (*Latching / Penguncian*): Selama proses pengisian dari pukul 15:32 hingga 16:06, level air berangsur naik dari 43% hingga 96%. Meskipun level air terus mengalami perubahan nilai secara *real-time*, status pompa berhasil ditahan/dikunci pada kondisi ON. Fitur histeresis ini terbukti efektif mencegah aktuator dari kondisi "cetak-cetek" akibat riak gelombang air di dalam tangki.
4. Fase Stop (Proteksi *Overflow*): Kinerja batas atas teruji pada pukul 16:07. Saat sensor mendeteksi level cairan menyentuh persentase 98% (sesuai nilai *Set High*), sistem seketika memutus arus relai sehingga pompa berubah menjadi OFF. Pada menit-menit berikutnya (16:08 hingga 16:09), pompa tetap dalam keadaan mati, memastikan bahwa sistem terproteksi dengan baik dari risiko kelebihan muatan (*overflow*).

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil perancangan, pembuatan, dan pengujian "Rancang Bangun Prototype Sistem Monitor dan Kontrol Pompa pada Pit Penampungan Limbah Cairan Berbasis Internet of Things", dapat ditarik beberapa kesimpulan mengenai fitur, kelebihan, dan kekurangan alat sebagai berikut:

5.1.1 Fitur dan Kemampuan Alat

1. Sistem Monitor *Hybrid*: Alat mampu menampilkan data *real-time* melalui tiga antarmuka sekaligus: layar OLED (data ringkas), layar TFT ILI9341 (menu dan Tampilan), dan Web Server (*dashboard* grafik lengkap dan akses jarak jauh via *smartphone*/PC).
2. Telemetri Nirkabel: Menggunakan modul nRF24L01 untuk mengirimkan data dari node sensor ke unit kontrol utama, sehingga sensor dapat ditempatkan di area yang sulit dijangkau kabel.
3. Kontrol Pompa Otomatis & Manual: Sistem memiliki logika kontrol pompa otomatis berdasarkan ketinggian level *liquid*(set point *Full* dan *Low* yang dapat dikalibrasi) serta fitur kontrol manual jarak jauh melalui Web Server.
4. Sistem Keamanan Gas (*Safety*): Dilengkapi sensor MQ-135 dengan algoritma *warm-up* (pemanasan awal) selama 60 detik untuk mencegah *false alarm*, serta indikator status "Aman", "Waspada", dan "Bahaya" dengan pemicu *buzzer* jika terdeteksi kebocoran gas di atas ambang batas (2100).
5. Kemudahan Akses: Tersedia fitur QR Code pada menu perangkat yang dapat dipindai untuk mempercepat akses login ke *dashboard* monitor berbasis web.

5.1.2 Kelebihan Sistem

1. Antarmuka Informatif: Penggunaan layar TFT berwarna dengan memudahkan operator memantau tren kenaikan level *liquid*dan gas secara visual langsung di lapangan tanpa perlu membuka laptop.
2. Fleksibilitas Kontrol: Pengguna dapat mengubah nilai kalibrasi tangki (batas atas/bawah) dan *set point* relai pompa langsung melalui tombol menu di perangkat tanpa perlu memprogram ulang koding (nilai tersimpan di EEPROM).
3. Redundansi Monitor : Jika koneksi WiFi terputus, alat tetap berfungsi penuh secara lokal (Layar TFT dan kontrol pompa otomatis tetap berjalan), sehingga keandalan sistem tetap terjaga.
4. Efisiensi Data: Sistem transmisi data hanya mengirimkan perubahan status yang signifikan untuk menghemat *bandwidth* dan pemrosesan, serta dilengkapi fitur *clipping* grafik agar tampilan tetap rapi.

5.1.3 Kekurangan Sistem

1. Waktu Pemanasan Sensor: Sensor gas MQ-135 memerlukan waktu *pre-heating* (pemanasan) saat pertama kali dinyalakan agar pembacaan akurat, sehingga sistem tidak dapat langsung mendeteksi gas seketika setelah *booting*.
2. Keterbatasan Jangkauan WiFi: Akses kontrol jarak jauh sangat bergantung pada kekuatan sinyal WiFi di lokasi penempatan alat. Jika sinyal lemah atau server Ngrok *down*, fitur *remote control* tidak dapat digunakan.
3. Akurasi Sensor Ultrasonik: Pembacaan level *liquid*menggunakan sensor ultrasonik dapat terganggu jika terdapat busa tebal pada limbah c*liquid*atau uap panas yang memantulkan gelombang suara secara tidak presisi.

5.2 Saran

Untuk pengembangan lebih lanjut agar sistem ini dapat diimplementasikan pada skala industri yang lebih besar, penulis memberikan beberapa saran sebagai berikut:

1. Integrasi Database Cloud: Penyimpanan data sebaiknya ditingkatkan dari kartu SD lokal ke database *cloud* (seperti Firebase atau MySQL) agar data historis dapat diakses dari mana saja dan tidak hilang jika kartu SD rusak.
2. Pembuatan Aplikasi Mobile Native: Mengembangkan aplikasi berbasis Android/iOS secara *native* (bukan hanya web view) untuk mendapatkan fitur notifikasi *push* langsung ke HP operator saat terjadi kondisi bahaya (level *liquid*meluap atau kebocoran gas).
3. Desain PCB Cetak: Untuk mengurangi risiko kabel kendor (loose connection) akibat getaran mesin pompa, sebaiknya rangkaian dipindahkan dari *breadboard* atau kabel *jumper* ke PCB cetak (*Printed Circuit Board*) yang lebih kokoh dan terlindung *casing* standar industri (IP65).

DAFTAR PUSTAKA

[1] Az-Zikri, A. N., Indriyanto, S., & Wicaksono, A. (2026). Perancangan Prototype sistem monitor level *liquid*tandon berbasis Internet of Things (*Internet of Things*) menggunakan sensor ultrasonik JSN-SR04T. *Jurnal SINTA: Sistem Informasi dan Teknologi Komputasi*, 2(1), 1–10.

[2] Rienandie, N. F., & Pramudita, R. (2026). Design of an Internet of Things-based cairan level monitor system. *Jurnal Teknologi dan Sistem Informasi (JURTEKSI)*, 11(2), 115–124.

[3] A. N. Az-Zikri, S. Indriyanto, dan A. Wicaksono, "Perancangan Prototype Sistem Monitor Level *Liquid*Tandon Berbasis Internet of Things (*Internet of Things*) Menggunakan Sensor Ultrasonik JSN-SR04T," *Jurnal Teknik Elektro dan Komputer*, vol. 9, no. 2, pp. 85–92, 2021.

[4] N. F. Rienandie dan R. Pramudita, "Design of an Internet of Things-Based Cairan Level Monitor System," *Jurnal Teknik dan Sistem Komputer*, vol. 3, no. 1, pp. 14–20, 2022.

[5] P. P. Prasanth, A. Thomas, dan J. Kurian, "Scalable and Robust *Internet of Things* -Based Liquid Level Monitor /Control System for Home and Industrial Automation," *Journal of Scientific and Technical Research*, vol. 6, no. 1, pp. 45–52, 2019.

[6] A. S. Pardomuan dan U. Zaky, "Automated Cairan Level Control System Using *Internet of Things* Under Diverse Conditions," *Journal of Research in Engineering and Technology*, vol. 8, no. 4, pp. 112–118, 2020.

[7] E. K. Wati, H. H. Santoso, dan A. Laksono, "*Internet of Things* -Based Cairan Level Monitor System of Situ Rawa Besar," *Multidisciplinary Scientific Journal*, vol. 4, no. 2, pp. 67–74, 2021.

[8] A. P. Hasanah, M. I. Sarif, dan Hafni, "Perancangan Sistem Monitor Level *Liquid*Menggunakan Sensor Ultrasonik Berbasis *Internet of Things* dengan Aplikasi Dashboard," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 10, no. 3, pp. 201–208, 2022.

[9] Rajawali Utama, “Prinsip Kerja, Jenis, dan Aplikasi Cairan Level Control,” 2023.

[10] Instrumentation Tools, “Contact and Non-Contact Level Sensors,” 2022.

[11] INAparts, “Level Sensor: Definisi, Fungsi, dan Jenis,” 2021.

[12] EPT Technology, “How Does the Non-Contact Liquid Level Sensor Work,” 2022.

[13] Cytron Technologies, HC-SR04 Ultrasonic Sensor Datasheet, 2019.

[14] Elecrow Electronics, Ultrasonic Ranging Module HC-SR04 Datasheet, 2018.

[15] Nordic Semiconductor. (2008). nRF24L01 Single Chip 2.4GHz Transceiver Product Specification (Ver. 2.0)

[16] Espressif Systems. (2023). *ESP32 Series Datasheet*. Espressif Systems Inc.

[17] Espressif Systems. (2022). *ESP8266EX Datasheet*. Espressif Systems Inc.

[18] Dashboard Inc., “Getting Started with Dashboard,” 2023.

[19] Siemens AG, Contactor and Relay Technical Guide, 2019.

[20] Schneider Electric, Electrical Installation Guide – Contactors and Relays, 2020.

[21] Bosch Sensortec, BMP180 Digital Pressure Sensor Datasheet, 2015.

[22] Aosong Electronics, DHT22 Digital Temperature and Humidity Sensor Product Manual, 2021.

[23] Zhengzhou Winsen Electronics Technology, MQ-135 Gas Sensor for *Liquid*Quality Datasheet, 2018.

[24] Ilitek Technology, ILI9341 a-Si TFT LCD Single Chip Driver Datasheet, 2011.

[25] Universal-Science, JSN-SR04T Waterproof Ultrasonic Range Finder User Manual, 2019.

LAMPIRAN A

Code Program Esp 32 (Rangkaian Kontrol)

```
/*
   PROJECT:LEVEL LIQUID MONITOR
   Author: Adi Suardiman
   Board: ESP32 Dev Module
*/

#include <WiFi.h>
#include <WiFiClient.h>
#include <WebServer.h>
#include <Adafruit_GFX.h>
#include <Adafruit_ILI9341.h>
#include <Adafruit_SSD1306.h>
#include <EEPROM.h>
#include <SPI.h>
#include <Wire.h>
#include <DHT.h>
#include <Adafruit_BMP085.h>
#include <nRF24L01.h>
#include <RF24.h>
#include "FS.h"
#include "SD.h"
#include "time.h"

//   KONFIGURASI WEB SERVER
WebServer server(80);

//   DEFINISI WARNA TFT
#define C_BG            0x10A2
#define C_METRO_BLUE    0x039B
#define C_METRO_GREEN   0x0500
#define C_METRO_RED     0xC000
#define C_METRO_ORANGE  0xDC00
#define C_METRO_PURPLE  0x6010
#define C_METRO_GRAY    0x4208
#define C_TEXT_W        0xFFFF
#define C_TEXT_B        0x0000
#define C_TEXT_G        0xBDF7
#define C_ALERT         0xF800
#define C_OK            0x07E0
#define C_WARN          0xFD20

//   KONFIGURASI TAMPILAN
#define GRAPH_X 30
#define GRAPH_Y 80
#define GRAPH_W 200
#define GRAPH_H 100
#define MAX_POINTS 50
#define STATUS_BAR_Y  295

//   PINOUT ESP32
#define TFT_CS     15
#define TFT_RST    32
#define TFT_DC     33
#define SD_CS      27
#define BATT_PIN   35
#define AD_PIN     34
#define TRIG_PIN   14
```

```
#define ECHO_PIN   12
#define BUZZER_PIN 13
#define DHT_PIN    4
#define MQ135_PIN  36
#define LASER_PIN  16
#define RF_CE      17
#define RF_CSN     5

#define DHTTYPE    DHT22

//   KONFIGURASI GAS SENSOR
#define GAS_THRESHOLD 2100
const unsigned long MQ_WARMUP_TIME = 40000;

//   NTP & TIME
const char* ntpServer = "pool.ntp.org";
const long  gmtOffset_sec = 25200; // WIB (GMT+7)
const int   daylightOffset_sec = 0;

char daysOfTheWeek[7][12] = {"Minggu", "Senin", "Selasa", "Rabu",
"Kamis", "Jumat", "Sabtu"};

// Hardware Objects
Adafruit_SSD1306 oled(128, 32, &Wire, -1);
Adafruit_ILI9341 tft = Adafruit_ILI9341(TFT_CS, TFT_DC, TFT_RST);
DHT dht(DHT_PIN, DHTTYPE);
Adafruit_BMP085 bmp;
RF24 radio(RF_CE, RF_CSN);

//   NRF24
const byte rfAddress[6] = "00001";
struct DataPacket { float dist; int pct; float temp; float hum; int
airQuality; bool pump; };
DataPacket sensorData;
struct FeedbackPacket { bool isPumpOn; int sensorStatus; };
FeedbackPacket feedbackData;

//   GLOBAL VARIABLES
String wifi_ssid = "ADDSpoenya";
String wifi_pass = "11111111";

//   [MODIFIED] Added waterLevelCM and calibError variables
float distance, temperature, humidity, pressure;
float waterLevelCM = 0; // Menyimpan tinggi air (bukan jarak sensor)
bool calibError = false; // Flag jika pembacaan sensor melebihi
setLow (negatif)
//                    --

float liquidHeight = 0;
int airQuality, lastPct = 0;
int battPercent;

//   [MODIFIED] SETTINGAN EEPROM
int setFull = 20, setLow = 100;
// Rename variabel limit agar sesuai fungsi baru
int setLimitLow = 20;  // Dulu setPumpON (Batas Bawah %)
int setLimitHigh = 90; // Dulu setPumpOFF (Batas Atas %)
int setRFPower = 0;
int relayMode = 0;     // 0: FILLING (ISI), 1: DISCHARGE (KURAS)
[NEW]
//            --
```

```
int menuIndex = 0;
bool inSubMenu = false;
bool pumpState = false;
bool isOverride = false;

// UI Flags
bool needsFullRedraw = true;
bool needsMenuUpdate = true;
bool needsStatusUpdate = true;
bool needsLogRedraw = true;
int lastDrawnMenuIndex = -1;
bool isGasAlertActive = false;
bool wasGasAlertActive = false;

// WEB CONTROL OVERRIDE
volatile bool webOverrideReq = false;
volatile bool webExitOverrideReq = false;

// Graph & Logs
float graphDist[MAX_POINTS];
int graphGas[MAX_POINTS];
int dataCount = 0;
int graphMode = 0;
bool logLoaded = false;

//   SCROLL & BUFFER
int scrollY = 0;
const int MAX_SCROLL = 380;
const int LOG_SIZE = 40;
float logTemp[LOG_SIZE], logHum[LOG_SIZE], logGas[LOG_SIZE],
logPct[LOG_SIZE];
unsigned long lastLogTime = 0;

//   FILE SYSTEM VARIABLES
String currentLogFileName = "/datalog.csv";
String sdFileList[15];
int sdFileCount = 0;
bool isFileView = false;
String selectedLogFile = "";

// Event Logs
struct EventItem { char timeStr[6]; String status; uint16_t color; };
EventItem pumpLogs[10];
int lastSensorStatus = -1;
bool lastRfState = false;
bool rfConnected = false;
bool rfStrongSignal = false;
unsigned long lastRfRecv = 0;

// Input
unsigned long lastRepeatTime = 0;
int repeatDelay = 400;

//   TIMERS
unsigned long lastSensorRead = 0;
const long sensorInterval = 2000;

// VARIAN LASER POINTER
unsigned long lastMorseTime = 0;
int morseStep = 0;
```

```
struct MorseStep { bool state; int duration; };
MorseStep morseADI[] = {
  {LOW, 150}, {HIGH, 150}, {LOW, 450}, {HIGH, 600},
  {LOW, 450}, {HIGH, 150}, {LOW, 150}, {HIGH, 150}, {LOW, 150},
{HIGH, 600},
  {LOW, 150}, {HIGH, 150}, {LOW, 150}, {HIGH, 2000}
};
const int totalMorseSteps = sizeof(morseADI) / sizeof(morseADI[0]);

// Dashboard Cache
float l_temp = -99, l_hum = -99, l_dist = -99;
int l_pump_state = -1;
bool l_rf_stat = false;
int bootProgress = 0;

//   CHAT SYSTEM VARIABLES (NEW)
struct ChatMsg { int id; String user; String text; String ip; };
ChatMsg chatBuffer[10]; // Buffer untuk 10 pesan terakhir di RAM
int msgCount = 0;
int globalMsgId = 0;
const char* macroFileName = "/macro read csv.xlsm";

// FUNGSI PROTOTYPE
void dumpRamToSD();
void savePumpCycleToCSV();
String readButtons();
void drawStatusBar();
void updateOLED(int pct, float dist);
void checkPumpEvents();
void logDataToSD();

//   CODE AUDIO & SHUTDOWN
void playTone(int type) {
  if (type == 1) {
    tone(BUZZER_PIN, 500, 100); delay(100);
    tone(BUZZER_PIN, 1000, 100); delay(100);
    tone(BUZZER_PIN, 2000, 200); delay(200);
  }
  else if (type == 2) {
    tone(BUZZER_PIN, 2000); delay(150);
    tone(BUZZER_PIN, 1500); delay(150);
    noTone(BUZZER_PIN);
  }
  else if (type == 3) {
    tone(BUZZER_PIN, 2500, 50); delay(50);
  }
}

void performSafeShutdown() {
  tft.fillScreen(C_TEXT_B);
  tft.setTextColor(C_TEXT_W); tft.setTextSize(2);
  tft.setCursor(60, 100); tft.print("SAVING...");

  dumpRamToSD();
  savePumpCycleToCSV();
  delay(500);

  tft.fillRect(40, 140, 160, 10, C_METRO_BLUE);
  for(int i=160; i>0; i-=40) {
      tft.fillRect(40, 140, i, 10, C_METRO_GREEN);
      delay(100);
```

```
    }

    tft.fillScreen(C_TEXT_B);
    tft.setCursor(50, 140); tft.setTextColor(C_METRO_RED);
    tft.print("SYSTEM OFF");

    radio.powerDown();
    oled.clearDisplay(); oled.display();
    playTone(2);
    esp_deep_sleep_start();
}

void showShutdownMenu() {
    int popX = 30, popY = 80, popW = 180, popH = 120;
    tft.fillRect(popX+5, popY+5, popW, popH, 0x2104);
    tft.fillRect(popX, popY, popW, popH, C_BG);
    tft.drawRect(popX, popY, popW, popH, C_METRO_RED);

    tft.setTextColor(C_TEXT_W); tft.setTextSize(2);
    tft.setCursor(popX + 35, popY + 20); tft.print("POWER OFF?");

    tft.setTextSize(1); tft.setTextColor(C_TEXT_G);
    tft.setCursor(popX + 20, popY + 50); tft.print("Data will be
saved.");

    tft.fillRoundRect(popX + 20, popY + 75, 60, 30, 4, C_METRO_GRAY);
// NO
    tft.setCursor(popX + 35, popY + 82); tft.setTextColor(C_TEXT_W);
tft.print("NO");

    tft.fillRoundRect(popX + 100, popY + 75, 60, 30, 4, C_METRO_RED);
// YES
    tft.setCursor(popX + 115, popY + 82); tft.setTextColor(C_TEXT_W);
tft.print("YES");

    bool waiting = true;
    while(waiting) {
        server.handleClient();
        String b = readButtons();
        if (b == "SELECT") {
            playTone(3); performSafeShutdown(); waiting = false;
        }
        else if (b == "BACK") {
            playTone(3); waiting = false; needsFullRedraw = true;
tft.fillScreen(C_BG);
        }
        delay(50);
    }
}

//   WEB SERVER & DASHBOARD CODE
String getDashboardHTML() {
    String p = "<!DOCTYPE html><html><head><meta charset='UTF-8'><meta
name='viewport' content='width=device-width, initial-scale=1'>";
    p += "<title>TA - ADI SUARDIMAN</title>";
    p += "<script
src='https://code.highcharts.com/highcharts.js'></script>";

    //   STYLE (CSS)
    p += "<style>";
```

```
  p += "body { font-family: 'Segoe UI', sans-serif; background-color:
#121212; color: white; text-align: center; margin: 0; padding: 0;
user-select: none; }";
  p += ".header { background: #004d40; padding: 20px; }";
  p += "h1 { margin: 0; font-size: 24px; color: #4db6ac; }";

  p += ".container { display: flex; flex-wrap: wrap; justify-content:
center; padding: 20px; gap: 20px; }";
  p += ".card { background: #1e1e1e; border-radius: 15px; padding:
20px; width: 300px; box-shadow: 0 4px 10px rgba(0,0,0,0.5); }";
  p += ".card-wide { width: 95%; max-width: 950px; }";
  p += ".card-title { font-size: 16px; color: #aaa; margin-bottom:
15px; border-bottom: 1px solid #333; padding-bottom: 10px; }";
  p += ".value { font-size: 40px; font-weight: bold; }";

  p += ".tank-container { width: 100px; height: 150px; border: 4px
solid #555; border-radius: 10px; position: relative; margin: 0 auto;
background: #222; overflow: hidden; }";
  p += ".liquid { width: 100%; position: absolute; bottom: 0;
background: linear-gradient(to top, #007bff, #00d2ff); transition:
height 1s ease-in-out; opacity: 0.8; }";
  p += ".liquid-text { position: absolute; top: 50%; left: 50%;
transform: translate(-50%, -50%); font-weight: bold; text-shadow: 1px
1px 2px black; z-index: 10; }";

  p += ".btn { display: inline-block; padding: 15px 40px; font-size:
16px; color: white; border: none; border-radius: 50px; cursor:
pointer; transition: 0.3s; margin-top: 5px; width: 80%; text-
decoration: none; }";
  p += ".btn-on { background: #28a745; } .btn-off { background:
#dc3545; }";
  p += ".btn-override { background: #ff8f00; border: 2px solid white;
animation: blink 1s infinite; }";
  p += ".btn-down { background: #008CBA; border-radius: 5px; width:
auto; padding: 10px 20px; color: white; border: none; cursor:
pointer;}";
  p += ".btn-file { background: #333; color: #eee; width: 100%;
margin: 5px 0; padding: 10px; text-align: left; border: 1px solid
#555; border-radius: 5px; cursor: pointer; }";
  p += ".btn-file:hover { background: #004d40; border-color: #4db6ac;
}";

  p += ".modal { display: none; position: fixed; z-index: 100; left:
0; top: 0; width: 100%; height: 100%; background-color:
rgba(0,0,0,0.8); }";
  p += ".modal-content { background-color: #1e1e1e; margin: 10% auto;
padding: 20px; border: 1px solid #4db6ac; width: 80%; max-width:
400px; border-radius: 10px; text-align: left;}";
  p += ".close { color: #aaa; float: right; font-size: 28px; font-
weight: bold; cursor: pointer; }";

  p += ".override-alert { color: #ff3d00; font-weight: bold; margin-
top: 10px; display: none; animation: blink 1s infinite; }";
  p += "@keyframes blink { 50% { opacity: 0; } }";
  p += ".grid { display: grid; grid-template-columns: 1fr 1fr; gap:
10px; }";
  p += ".grid-item { background: #2c2c2c; padding: 10px; border-
radius: 8px; }";
  p += ".grid-lbl { font-size: 12px; color: #bbb; }";
  p += ".grid-val { font-size: 18px; font-weight: bold; color: #fff;
}";
```

```
  p += "#chat-container { position: fixed; bottom: 20px; right: 20px;
width: 300px; background: #1e1e1e; border: 1px solid #4db6ac; border-
radius: 10px; box-shadow: 0 5px 15px rgba(0,0,0,0.5); z-index: 1000;
display: none; flex-direction: column; overflow: hidden; }";
  p += "#chat-header { background: #004d40; color: white; padding:
10px; cursor: pointer; font-weight: bold; display: flex; justify-
content: space-between; }";
  p += "#chat-messages { height: 250px; overflow-y: auto; padding:
10px; display: flex; flex-direction: column; gap: 8px; background:
#121212; }";
  p += ".msg { padding: 8px; border-radius: 5px; font-size: 13px;
max-width: 85%; word-wrap: break-word; }";
  p += ".msg-other { background: #333; align-self: flex-start; color:
#eee; }";
  p += ".msg-me { background: #00796b; align-self: flex-end; color:
white; }";
  p += "#chat-input-area { display: flex; padding: 10px; background:
#1e1e1e; border-top: 1px solid #333; }";
  p += "#chat-input { flex: 1; background: #121212; border: 1px solid
#444; color: white; padding: 5px; border-radius: 3px; }";
  p += "#btn-send { background: #4db6ac; border: none; color: black;
margin-left: 5px; cursor: pointer; border-radius: 3px; width: 40px;
}";
  p += "#chat-trigger { position: fixed; bottom: 20px; right: 20px;
width: 60px; height: 60px; background: #4db6ac; border-radius: 50%;
display: flex; align-items: center; justify-content: center; cursor:
pointer; box-shadow: 0 4px 10px rgba(0,0,0,0.3); z-index: 999; font-
size: 24px; }";

  p += "</style>";

  //   JAVASCRIPT
  p += "<script>";
  p += "var timer; var holdDuration = 3000; var isHolding = false;";
  p += "var chart; var maxDataPoints = 100;";

  p += "function startHold() { isHolding = false; timer =
setTimeout(function() { isHolding = true; fetch('/override_on'); },
holdDuration); }";
  p += "function endHold() { clearTimeout(timer); }";
  p += "function handleDbClick() { fetch('/override_off'); }";
  p += "function handleClick() { if(!isHolding) { fetch('/toggle'); }
}";

  p += "var lastMsgId = 0;";
  p += "function toggleChat() { var chat =
document.getElementById('chat-container'); chat.style.display =
(chat.style.display === 'none' || chat.style.display === '') ? 'flex'
: 'none'; }";
  p += "function sendMessage() { var inp =
document.getElementById('chat-input'); var msg = inp.value.trim();
if(msg === '') return; fetch('/send_chat?msg=' +
encodeURIComponent(msg)).then(() => { inp.value = ''; getMessages();
}); }";
  p += "function getMessages() { fetch('/get_chat?last=' +
lastMsgId).then(r => r.json()).then(data => { var box =
document.getElementById('chat-messages'); data.forEach(m => { var div
= document.createElement('div'); div.className = 'msg ' + (m.self ?
'msg-me' : 'msg-other'); div.innerHTML = '<b>' + m.user + '</b><br>'
```

```
    + m.text; box.appendChild(div); lastMsgId = m.id; }); if(data.length
    > 0) box.scrollTop = box.scrollHeight; }); }";
      p += "setInterval(getMessages, 3000);";

      p += "function toggleCoroom() {";
      p += "  var box = document.getElementById('coroomBox');";
      p += "  var btn = document.getElementById('btnCoroom');";
      p += "  if (box.style.display === 'none' || box.style.display ===
    '') {";
      p += "    box.style.display = 'block';";
      p += "    btn.innerHTML = '✕ Tutup Voice';";
      p += "    btn.style.background = '#dc3545';";
      p += "  } else {";
      p += "    box.style.display = 'none';";
      p += "    btn.innerHTML = ' Coroom Voice';";
      p += "    btn.style.background = '#00d2ff';";
      p += "  }";
      p += "}";

      p += "function initChart() {";
      p += "  chart = Highcharts.chart('chart-live', {";
      p += "    chart: { type: 'line', backgroundColor: '#1e1e1e',
    animation: Highcharts.svg, marginLeft: 40, marginRight: 40 },";
      p += "    title: { text: 'MONITORING LIVE', style: {color:
    '#4db6ac', fontSize: '14px'} },";
      p += "    xAxis: { type: 'datetime', visible: false },";
      p += "    yAxis: [{";
      p += "       title: { text: null },";
      p += "       min: 0, max: 100, gridLineColor: '#333', labels: {
    style: {color: '#00d2ff'}, align: 'right', x: -5 }";
      p += "    }, {";
      p += "       title: { text: null },";
      p += "       min: 0, opposite: true, gridLineWidth: 0, labels: {
    style: {color: '#ff9800'}, align: 'left', x: 5 }";
      p += "    }, {";
      p += "       title: { text: null },";
      p += "       gridLineWidth: 0, labels: { enabled: false }";
      p += "    }, {";
      p += "       title: { text: null },";
      p += "       opposite: true, gridLineWidth: 0, labels: { enabled:
    false }";
      p += "    }],";
      p += "    legend: { enabled: true, itemStyle: {color: '#ccc'} },";
      p += "    series: [{";
      p += "       name: 'Level (%)', yAxis: 0, data: [], color:
    '#00d2ff', type: 'area', fillOpacity: 0.2, tooltip: { valueSuffix: '
    %' }";
      p += "    }, {";
      p += "       name: 'Tinggi Air (cm)', yAxis: 1, data: [], color:
    '#ff9800', dashStyle: 'ShortDot', tooltip: { valueSuffix: ' cm' }";
      p += "    }, {";
      p += "       name: 'Suhu', yAxis: 2, data: [], color: '#ff5252',
    tooltip: { valueSuffix: ' °C' }";
      p += "    }, {";
      p += "       name: 'Gas', yAxis: 3, data: [], color: '#00e676',
    tooltip: { valueSuffix: '  ' }";
      p += "    }]";
      p += "  });";
      p += "}";

    p += "function openFileModal() {";
```

```
  p += "  document.getElementById('fileModal').style.display =
'block';";
  p += "  document.getElementById('fileList').innerHTML = 'Scanning
SD Card...';";
  p += "  fetch('/list_files').then(r => r.json()).then(files => {";

  //   BAGIAN SORTING (LOGIKA BARU)
  p += "    files.sort(function(a, b) {";
  // Regex untuk mengambil angka: dd-mm-yyyy___hh_mm dari nama file
  p += "       var getTs = function(s) {";
  p += "         var m = s.match(/-(\\d{2})-(\\d{2})-
(\\d{4})___(\\d{2})_(\\d{2})/);";
  // Format Date JS: Year, Month(0-11), Day, Hour, Minute
  p += "         return m ? new Date(m[3], m[2]-1, m[1], m[4],
m[5]).getTime() : 0;";
  p += "       };";
  // Sort Descending (Terbaru di atas) -> b - a
  p += "       return getTs(b) - getTs(a);";
  p += "    });";
  //

  p += "    var html = '';";
  p += "    if(files.length == 0) html = '<p>Tidak ada file
CSV.</p>';";
  p += "    else { files.forEach(f => { html += '<div class=\"btn-
file\" onclick=\"window.location=\\'/download?file='+f+'\\'\">
'+f+'</div>'; }); }";
  p += "    document.getElementById('fileList').innerHTML = html;";
  p += "  }).catch(e => {
document.getElementById('fileList').innerHTML = 'Error reading SD';
});";
  p += "}";

  p += "function closeFileModal() {
document.getElementById('fileModal').style.display = 'none'; }";

  p += "function getData() {";
  p += "  fetch('/data?nocache=' + new Date().getTime())";
  p += "    .then(r => r.json())";
  p += "    .then(data => {";
  p += "    if(data.error) {";
  p += "       document.getElementById('dist').innerHTML = 'LOW
ERROR!';";
  p += "       document.getElementById('dist').style.color =
'#ff1744';";
  p += "       document.getElementById('pct').innerHTML = 'ERR';";
  p += "       document.getElementById('tank-fill').style.height =
'0%';";
  p += "    } else if(data.dist > 600) {";
  p += "       document.getElementById('dist').innerHTML = 'N/A';";
  p += "       document.getElementById('dist').style.color =
'white';";
  p += "       document.getElementById('pct').innerHTML = '--';";
  p += "       document.getElementById('tank-fill').style.height =
'0%';";
  p += "    } else {";
  p += "       document.getElementById('dist').innerHTML = data.dist
+ ' cm';";
  p += "       document.getElementById('dist').style.color =
'white';";
```

```
  p += "       document.getElementById('pct').innerHTML = data.pct +
'%';";
  p += "       document.getElementById('tank-fill').style.height =
data.pct + '%';";
  p += "    }";
  p += "    document.getElementById('temp').innerHTML = data.temp +
'°C';";
  p += "    document.getElementById('hum').innerHTML = data.hum +
'%';";
  p += "    document.getElementById('press').innerHTML = data.press +
' hPa';";
  p += "    var gVal = data.gas; var gSts = 'AMAN'; var gCol =
'#00e676';";
  p += "    if(gVal > 1250) { gSts = 'WASPADA'; gCol = '#ffea00'; }";
  p += "    if(gVal > 2100) { gSts = 'BAHAYA'; gCol = '#ff1744'; }";
  p += "    document.getElementById('gas').innerHTML = gVal + '
<br><span style=\"font-size:12px; color:'+gCol+'\">(' + gSts +
')</span>';";
  p += "    var pSts = data.pump ? 'AKTIF' : 'MATI';";
  p += "    var pCol = data.pump ? '#00e676' : '#ff1744';";
  p += "    var elPump = document.getElementById('p-stat');";
  p += "    if(elPump) elPump.innerHTML = '<span
style=\"color:'+pCol+'\">' + pSts + '</span>';";
  p += "    var fSts = 'STOP'; var fCol = '#888';";
  p += "    if(!data.rf) { fSts = 'NO RF'; fCol = '#ff1744'; }";
  p += "    else if(data.flow) { fSts = 'ACTIVE'; fCol = '#00e676';
}";
  p += "    var elFlow = document.getElementById('f-stat');";
  p += "    if(elFlow) elFlow.innerHTML = '<span
style=\"color:'+fCol+'\">' + fSts + '</span>';";
  p += "    var btn = document.getElementById('btn-pump');";
  p += "    var alertTxt = document.getElementById('override-txt');";
  p += "    if(data.override) {";
  p += "       alertTxt.style.display = 'block';";
  p += "       btn.innerHTML = data.pump ? 'FORCE STOP' : 'FORCE
START';";
  p += "       btn.className = 'btn btn-override';";
  p += "    } else {";
  p += "       alertTxt.style.display = 'none';";
  p += "       if(data.pump) { btn.innerHTML = 'MATIKAN POMPA';
btn.className = 'btn btn-off'; }";
  p += "       else { btn.innerHTML = 'NYALAKAN POMPA'; btn.className
= 'btn btn-on'; }";
  p += "    }";
  p += "    var x = (new Date()).getTime();";
  p += "    var shift = chart.series[0].data.length >
maxDataPoints;";
  p += "    chart.series[0].addPoint([x, data.pct], true, shift);";
  p += "    chart.series[1].addPoint([x, data.dist], true, shift);";
  p += "    chart.series[2].addPoint([x, data.temp], true, shift);";
  p += "    chart.series[3].addPoint([x, data.gas], true, shift);";
  p += "  }).catch(err => console.log(err))";
  p += "  .finally(() => { setTimeout(getData, 1000); });";
  p += "}";

  p += "window.onload = function() { initChart(); getData(); };";
  p += "</script>";

  //   HTML BODY
  p += "</head><body>";
```

```
  p += "<div id='fileModal' class='modal'>";
  p += "  <div class='modal-content'>";
  p += "    <span class='close' onclick='closeFileModal()'>×</span>";
  p += "    <h3 style='color:#4db6ac; margin-top:0'>Pilih File Log SD
Card</h3>";
  p += "    <div id='fileList' style='max-height:300px; overflow-
y:auto;'>Loading...</div>";
  p += "  </div>";
  p += "</div>";

  p += "<div class='header'><h1>MONITORING LEVEL
LIQUID</h1><h3>POWERED BY ADI SUARDIMAN</h3></div>";
  p += "<div class='container'>";

  //   KARTU 1: TANGKI
  p += "<div class='card'><div class='card-title'>PIT LEVEL
STATUS</div>";
  p += "<div class='tank-container'><div id='tank-fill'
class='liquid' style='height: " + String(lastPct) + "%;'></div>";
  p += "<div id='pct' class='liquid-text'>" + String(lastPct) +
"%</div></div>";
  p += "<div style='margin-top:15px;'><span class='value' id='dist'>"
+ String(waterLevelCM, 1) + " cm</span></div></div>";

  //   KARTU 2: SYSTEM DATA
  p += "<div class='card'><div class='card-title'>SYSTEM
DATA</div><div class='grid'>";
  p += "<div class='grid-item'><div class='grid-lbl'>SUHU</div><div
class='grid-val' id='temp'>" + String(temperature, 1) +
"°C</div></div>";
  p += "<div class='grid-item'><div class='grid-
lbl'>KELEMBABAN</div><div class='grid-val' id='hum'>" +
String((int)humidity) + "%</div></div>";
  p += "<div class='grid-item'><div class='grid-lbl'>GAS</div><div
class='grid-val' id='gas'>" + String(airQuality) + "  </div></div>";
  p += "<div class='grid-item'><div class='grid-
lbl'>TEKANAN</div><div class='grid-val' id='press'>" +
String((int)pressure) + " hPa</div></div>";
  p += "<div class='grid-item'><div class='grid-lbl'>STATUS
POMPA</div><div class='grid-val' id='p-stat'>--</div></div>";
  p += "<div class='grid-item'><div class='grid-lbl'>FEEDBACK
FLOW</div><div class='grid-val' id='f-stat'>--</div></div>";
  p += "</div></div>";

  //   KARTU 3: KONTROL POMPA
  p += "<div class='card'><div class='card-title'>PUMP
CONTROL</div>";
  p += "<div id='override-txt' class='override-alert'>⚠ MODE
OVERRIDE ⚠</div>";
  p += "<p style='color:#888; font-size:12px;'>Hold (3s): Override |
2x Click: Normal</p>";
  p += "<button id='btn-pump' class='btn btn-on' ";
  p += "onmousedown='startHold()' onmouseup='endHold()'
ontouchstart='startHold()' ontouchend='endHold()' ";
  p += "ondblclick='handleDbClick()'
onclick='handleClick()'>LOADING...</button>";
  p += "</div>";

  //   KARTU 4: GRAFIK LIVE
  p += "<div class='card card-wide'><div class='card-title'>REAL-TIME
PC RECORDING (4 PARAMS)</div>";
```

```
  p += "<div id='chart-live' style='height:350px;
width:100%;'></div>";
  p += "<br><button onclick='openFileModal()' class='btn-down'> PILIH
& DOWNLOAD LOG (SD CARD)</button>";
  p += "</div>";

  p += "<a id='macro-trigger' href='/download_macro' title='Download
Macro Read CSV'></a>";

  //   CHAT WIDGETS
  p += "<div id='chat-trigger' onclick='toggleChat()'></div>";
  p += "<div id='chat-container'>";
  p += "  <div id='chat-header' onclick='toggleChat()'>";
  p += "    <span>OPERATOR CHAT</span>";
  p += "    <span>✖</span>";
  p += "  </div>";
  p += "  <div id='chat-messages'></div>";
  p += "  <div id='chat-input-area'>";
  p += "    <input type='text' id='chat-input' placeholder='Ketik
pesan...' onkeypress='if(event.keyCode==13) sendMessage()'>";
  p += "    <button id='btn-send'
onclick='sendMessage()'>➤</button>";
  p += "  </div>";
  p += "</div>";

  //   COROOM.APP VOICE WIDGET
  p += "<div id='coroom-container' style='position:fixed;
bottom:100px; right:20px; z-index:1001;'>";
  p += "  <div id='coroomBox' style='width:350px; height:500px;
background:#1e1e1e; border:2px solid #00d2ff; border-radius:15px;
display:none; margin-bottom:10px; overflow:hidden; box-shadow: 0 10px
30px rgba(0,0,0,0.5);'>";
  p += "    <div style='background:#00d2ff; padding:8px; color:black;
font-weight:bold; font-size:12px; text-align:center;'>OPERATOR VOICE
CHANNEL</div>";
  p += "    <iframe src='https://coroom.app/go/righteous-punishment-
2577' allow='microphone; camera; display-capture' style='width:100%;
height:465px; border:none;'></iframe>";
  p += "  </div>";
  p += "  <button id='btnCoroom' onclick='toggleCoroom()'
style='background:#00d2ff; color:black; font-weight:bold;
border:none; border-radius:50px; padding:12px 25px; cursor:pointer;
width:100%; box-shadow: 0 4px 15px rgba(0,210,255,0.3);'>";
  p += "     Coroom Voice";
  p += "  </button>";
  p += "</div>";

  p += "</div></body></html>";

  return p;
}

void handleRoot() {
  server.sendHeader("Cache-Control", "no-cache, no-store, must-
revalidate");
  server.sendHeader("Pragma", "no-cache");
  server.sendHeader("Expires", "-1");

  server.setContentLength(CONTENT_LENGTH_UNKNOWN);
  server.send(200, "text/html; charset=utf-8", "");
  server.sendContent(getDashboardHTML());
```

```
  server.sendContent("");
}

void handleDownloadMacro() {
  if (SD.exists(macroFileName)) {
    File file = SD.open(macroFileName, FILE_READ);
    server.sendHeader("Content-Type", "application/vnd.ms-
excel.sheet.macroEnabled.12");
    server.sendHeader("Content-Disposition", "attachment;
filename=\"macro read csv.xlsm\"");
    server.streamFile(file, "application/octet-stream");
    file.close();
  } else {
    server.send(404, "text/plain", "File Macro Tidak Ditemukan di SD
Card!");
  }
}

void handleListFiles() {
  String json = "[";
  File root = SD.open("/");
  if(root){
      File file = root.openNextFile();
      bool first = true;
      while(file){
          String fileName = String(file.name());
          // Hanya ambil file .csv atau .CSV
          if(fileName.indexOf(".csv") > 0 || fileName.indexOf(".CSV")
> 0) {
              if(!first) json += ",";
              json += "\"" + fileName + "\""; // Menambahkan nama
file ke JSON array
              first = false;
          }
          file = root.openNextFile();
      }
      root.close();
  }
  json += "]";
  server.send(200, "application/json", json);
}

void handleToggle() { pumpState = !pumpState; needsStatusUpdate =
true; server.sendHeader("Location", "/"); server.send(303); }

void handleWebOverrideOn() { webOverrideReq = true; server.send(200,
"text/plain", "OK"); }
void handleWebOverrideOff() { webExitOverrideReq = true;
server.send(200, "text/plain", "OK"); }

//   HANDLER CHAT (ADDED)
void handleSendChat() {
  if (server.hasArg("msg")) {
    String message = server.arg("msg");
    String senderIP = server.client().remoteIP().toString();

    // Geser buffer jika penuh (Circular Buffer logic simple)
    if (msgCount >= 10) {
      for (int i = 0; i < 9; i++) chatBuffer[i] = chatBuffer[i+1];
      msgCount = 9;
    }
```

```
    // Simpan pesan baru ke RAM (ID otomatis naik, User diambil dari
akhiran IP)
    chatBuffer[msgCount] = { ++globalMsgId, "User IP " +
senderIP.substring(senderIP.lastIndexOf('.')+1), message, senderIP };
    msgCount++;
    server.send(200, "text/plain", "OK");
  }
}

void handleGetChat() {
  int lastId = server.arg("last").toInt();
  String json = "[";
  bool first = true;
  // Loop buffer pesan
  for (int i = 0; i < msgCount; i++) {
    if (chatBuffer[i].id > lastId) {
      if (!first) json += ",";
      bool isSelf = (chatBuffer[i].ip ==
server.client().remoteIP().toString());
      // Bangun JSON manual
      json += "{\"id\":" + String(chatBuffer[i].id) +
              ",\"user\":\"" + chatBuffer[i].user +
              "\",\"text\":\"" + chatBuffer[i].text +
              "\",\"self\":" + (isSelf ? "true" : "false") + "}";
      first = false;
    }
  }
  json += "]";
  server.send(200, "application/json", json);
}

//   UPDATE: DOWNLOAD FILE SPESIFIK
void handleDownloadLog() {
  String targetFile = currentLogFileName; // Default file

  // Jika Web mengirim request file tertentu (misal:
/download?file=/data.csv)
  if (server.hasArg("file")) {
    targetFile = server.arg("file");
    if (!targetFile.startsWith("/")) targetFile = "/" + targetFile;
  }

  if (SD.exists(targetFile)) {
    File file = SD.open(targetFile, FILE_READ);
    if (file) {
      server.sendHeader("Content-Type", "text/csv");
      server.sendHeader("Content-Disposition", "attachment;
filename=" + targetFile.substring(1)); // Hapus slash depan untuk
nama file download
      server.sendHeader("Connection", "close");
      server.streamFile(file, "text/csv");
      file.close();
    } else server.send(500, "text/plain", "Fail Read");
  } else server.send(404, "text/plain", "File Not Found: " +
targetFile);
}

void handleSDData() {
  if (SD.exists(currentLogFileName)) {
    File file = SD.open(currentLogFileName, FILE_READ);
```

```
    if (file) { server.streamFile(file, "text/plain"); file.close();
}
  } else server.send(404, "text/plain", "No Data");
}

void handleData() {
  // 1. CEK ERROR SENSOR
  if (isnan(temperature) || isnan(humidity)) {
    temperature = 0.0; humidity = 0.0;
  }

  //   [MODIFIED]
  // Menggunakan waterLevelCM untuk dikirim ke web sebagai "dist"
agar grafik naik
  float displayValue = waterLevelCM;
  //

  // 2. LOGIKA FLOW
  bool isFlowing = (rfConnected && feedbackData.sensorStatus == 1);

  // 3. SUSUN JSON PAKAI SNPRINTF
  //   [MODIFIED] Menambahkan field error untuk web
  char jsonBuffer[350];
  snprintf(jsonBuffer, sizeof(jsonBuffer),

"{\"dist\":%.1f,\"pct\":%d,\"temp\":%.1f,\"hum\":%d,\"press\":%d,\"ga
s\":%d,\"override\":%s,\"pump\":%s,\"flow\":%s,\"rf\":%s,\"error\":%s
}",
    displayValue, // Ini sekarang tinggi air
    lastPct,
    temperature,
    (int)humidity,
    (int)pressure,
    airQuality,
    isOverride ? "true" : "false",
    pumpState ? "true" : "false",
    isFlowing ? "true" : "false",
    rfConnected ? "true" : "false",
    calibError ? "true" : "false" // Field baru
  );

  server.send(200, "application/json", jsonBuffer);
}

//   FUNGSI HELPER & LOG
void writeString(int add, String data) {
  int _size = data.length();
  for (int i = 0; i < _size; i++) { EEPROM.write(add + i, data[i]); }
  EEPROM.write(add + _size, '\0'); EEPROM.commit();
}
String readString(int add) {
  String data; char k; int len = 0; unsigned char k_c;
  k_c = EEPROM.read(add);
  while (k_c != '\0' && len < 64) { k = k_c; data += k; len++; k_c =
EEPROM.read(add + len); }
  return data;
}

int logY = 0;
void bootLog(String text) {
  tft.setTextSize(1); tft.setTextColor(C_OK);
```

```
  tft.setCursor(5, logY); tft.print("> "); tft.println(text);
  logY += 12;
  if(logY > 300) { tft.fillScreen(C_TEXT_B); logY = 0; }
  oled.clearDisplay(); oled.setTextSize(1);
oled.setTextColor(SSD1306_WHITE);
  oled.setCursor(0, 0); oled.print(F("SYSTEM BOOTING..."));
  oled.setCursor(0, 15); oled.print(text);
  oled.drawRect(0, 25, 128, 7, SSD1306_WHITE);
  bootProgress += 10; if (bootProgress > 124) bootProgress = 124;
  oled.fillRect(2, 27, bootProgress, 3, SSD1306_WHITE);
  oled.display(); delay(50);
}

//   [MODIFIED] Generate File Name with Date/Time Safety
void generateFileName() {
  struct tm timeinfo;
  if(!getLocalTime(&timeinfo)) { currentLogFileName = "/Default.csv";
}
  else {
    char buffer[64];
    // Format: Hari-dd-mm-yyyy___hh_mm.csv (Garis miring diganti
strip agar valid di SD Card)
    sprintf(buffer, "/%s-%02d-%02d-%04d___%02d_%02d.csv",
            daysOfTheWeek[timeinfo.tm_wday],
            timeinfo.tm_mday,
            timeinfo.tm_mon + 1,
            timeinfo.tm_year + 1900,
            timeinfo.tm_hour,
            timeinfo.tm_min
    );
    currentLogFileName = String(buffer);
  }
}

//   [MODIFIED] DUMP RAM TO SD with Enhanced Columns
void dumpRamToSD() {
  File file = SD.open(currentLogFileName, FILE_APPEND);

  if (!file) {
    file = SD.open(currentLogFileName, FILE_WRITE);

file.println("Waktu,RawSensor,TinggiAir,Level,Temp,Hum,Gas,Pump,FullT
op,LowBtm,Mode,SetHigh,SetLow,Feedback,RF");
  }

  // Ambil Waktu NTP Saat Ini untuk Snapshot
  struct tm timeinfo;
  char timeStr[25];
  if(getLocalTime(&timeinfo)) {
      sprintf(timeStr, "%02d/%02d/%04d %02d:%02d:%02d",
        timeinfo.tm_mday, timeinfo.tm_mon + 1, timeinfo.tm_year +
1900,
        timeinfo.tm_hour, timeinfo.tm_min, timeinfo.tm_sec);
  } else {
      strcpy(timeStr, "01/01/2000 00:00:00"); // Default jika NTP
error
  }

  for (int i=0; i<LOG_SIZE; i++) {
      if(logTemp[i] > 0) {
         //   [MODIFIED] Gunakan Format Tanggal NTP
```

```
            file.print(timeStr); file.print(","); // Kolom Waktu

            // Re-Calculate Data
            float recalcLevel = (logPct[i] / 100.0) * (setLow -
setFull);
            float recalcRaw = setLow - recalcLevel;

            // 1. Raw & Tinggi Air
            file.print(recalcRaw, 1); file.print(",");
            file.print(recalcLevel, 1); file.print(",");

            // 2. Data Env
            file.print(logPct[i]); file.print(",");
            file.print(logTemp[i]); file.print(",");
            file.print(logHum[i]); file.print(",");
            file.print(logGas[i]); file.print(",");
            file.print("0"); file.print(",");

            // 3. Settings & Status Snapshot
            file.print(setFull); file.print(",");
            file.print(setLow); file.print(",");
            file.print(relayMode == 0 ? "FILLING" : "DISCHARGE");
file.print(",");
            file.print(setLimitHigh); file.print(",");
            file.print(setLimitLow); file.print(",");

            bool rfOk = rfConnected;
            bool flowActive = (rfOk && feedbackData.sensorStatus == 1);
            file.print(flowActive ? "ACTIVE" : "--"); file.print(",");
            file.println(rfOk ? "OK" : "LOST");
        }
    }
    file.close();
}

void initSDCard() {
    if(!SD.begin(SD_CS)) { bootLog("ERR: SD Card Failed!"); return; }
    generateFileName(); bootLog("File: " + currentLogFileName);
}

unsigned long lastSDLogTime = 0;
const long sdLogInterval = 60000;

//   [MODIFIED] LOG DATA TO SD with Enhanced Columns
void logDataToSD() {
    if (millis() - lastSDLogTime < sdLogInterval) {
      return;
    }
    lastSDLogTime = millis();

    File file = SD.open(currentLogFileName, FILE_APPEND);

    if (!file) {
      file = SD.open(currentLogFileName, FILE_WRITE);
      // Header Lengkap

file.println("Waktu,RawSensor,TinggiAir,Level,Temp,Hum,Gas,Pump,FullT
op,LowBtm,Mode,SetHigh,SetLow,Feedback,RF");
    }

    if(file) {
```

```
    struct tm timeinfo;
    char timeStr[25]; // Buffer diperbesar untuk menampung dd/mm/yyyy
hh:mm:ss

    //   [MODIFIED] Format Waktu Lengkap Sesuai NTP
    if(getLocalTime(&timeinfo)) {
        // Format: dd/mm/yyyy hh:mm:ss (Contoh: 16/02/2026 14:30:05)
        sprintf(timeStr, "%02d/%02d/%04d %02d:%02d:%02d",
            timeinfo.tm_mday,           // Tanggal
            timeinfo.tm_mon + 1,        // Bulan (tm_mon mulai dari
0)
            timeinfo.tm_year + 1900,    // Tahun (tm_year mulai dari
1900)
            timeinfo.tm_hour,           // Jam
            timeinfo.tm_min,            // Menit
            timeinfo.tm_sec             // Detik
        );
    } else {
        strcpy(timeStr, "01/01/2000 00:00:00"); // Jika NTP belum
sync
    }

    // 1. Tulis Waktu
    file.print(timeStr); file.print(",");

    // 2. Sensor Utama
    file.print(distance, 1); file.print(",");      // Raw Sensor
    file.print(waterLevelCM, 1); file.print(",");  // Tinggi Air

    // 3. Env Data
    file.print(lastPct);  file.print(",");
    file.print(temperature, 1); file.print(",");
    file.print(humidity, 0); file.print(",");
    file.print(airQuality); file.print(",");
    file.print(pumpState ? "ON" : "OFF"); file.print(",");

    // 4. Settings
    file.print(setFull); file.print(",");
    file.print(setLow); file.print(",");
    file.print(relayMode == 0 ? "FILLING" : "DISCHARGE");
file.print(",");
    file.print(setLimitHigh); file.print(",");
    file.print(setLimitLow); file.print(",");

    // 5. Status RF & Flow
    bool flowActive = (rfConnected && feedbackData.sensorStatus ==
1);
    file.print(flowActive ? "ACTIVE" : "--"); file.print(",");
    file.println(rfConnected ? "OK" : "LOST");

    file.close();
  }
}

void scanLogFiles() {
  sdFileCount = 0; File root = SD.open("/"); if(!root){ return; }
  File file = root.openNextFile();
  while(file){
      String fileName = String(file.name());
      if(fileName.indexOf(".csv") > 0 || fileName.indexOf(".CSV") >
0) {
```

```
            if(sdFileCount < 15) { sdFileList[sdFileCount] = fileName;
sdFileCount++; }
        }
        file = root.openNextFile();
    }
}

void loadLogData(String fName) {
    tft.fillRect(20, 100, 200, 40, C_BG); tft.setCursor(60, 110);
tft.setTextColor(C_METRO_BLUE); tft.print("Reading File...");
    File file = SD.open(fName, FILE_READ);
    if (!file) { tft.setCursor(60, 110); tft.setTextColor(C_METRO_RED);
tft.print("Open Error!"); delay(1000); return; }
    size_t fileSize = file.size(); size_t seekPos = 0;
    if (fileSize > 2500) seekPos = fileSize - 2500;
    file.seek(seekPos);
    if (fileSize > 2500) file.readStringUntil('\n');
    dataCount = 0;
    while (file.available()) {
      String line = file.readStringUntil('\n');
      if (line.length() > 5) {
        int c1 = line.indexOf(','); int c2 = line.indexOf(',', c1 + 1);
        int c3 = line.indexOf(',', c2 + 1); int c4 = line.indexOf(',',
c3 + 1);
        int c5 = line.indexOf(',', c4 + 1); int c6 = line.indexOf(',',
c5 + 1); int c7 = line.indexOf(',', c6 + 1);
        if (c2 > 0 && c7 > 0) {
          String sDist = line.substring(c1 + 1, c2); String sGas =
line.substring(c6 + 1, c7);
          if (dataCount < MAX_POINTS) { graphDist[dataCount] =
sDist.toFloat(); graphGas[dataCount] = sGas.toInt(); dataCount++; }
          else {
            for (int i = 0; i < MAX_POINTS - 1; i++) { graphDist[i] =
graphDist[i+1]; graphGas[i] = graphGas[i+1]; }
            graphDist[MAX_POINTS - 1] = sDist.toFloat();
graphGas[MAX_POINTS - 1] = sGas.toInt();
          }
        }
      }
    }
    file.close();
}

void logPumpEvent(String status, uint16_t color) {
    for (int i = 9; i > 0; i--) { pumpLogs[i] = pumpLogs[i-1]; }
    struct tm timeinfo;
    if(!getLocalTime(&timeinfo)){ sprintf(pumpLogs[0].timeStr, "--:--
"); }
    else { sprintf(pumpLogs[0].timeStr, "%02d:%02d", timeinfo.tm_hour,
timeinfo.tm_min); }
    pumpLogs[0].status = status; pumpLogs[0].color = color;
    needsLogRedraw = true;
}

void checkPumpEvents() {
    if (rfConnected != lastRfState) {
      if (!rfConnected) logPumpEvent("RF LOST", C_METRO_RED);
      else logPumpEvent("RF CONN", C_OK);
      lastRfState = rfConnected;
    }
    if (rfConnected) {
```

```
      if (feedbackData.sensorStatus != lastSensorStatus) {
        if (feedbackData.sensorStatus == 1) logPumpEvent("FLOW ON",
C_METRO_GREEN);
        else if (feedbackData.sensorStatus == 0) logPumpEvent("FLOW
OFF", C_METRO_GRAY);
        lastSensorStatus = feedbackData.sensorStatus;
      }
    }
  }

  void updateEnvLog() {
    if (millis() - lastLogTime > 1000) {
      for (int i = 0; i < LOG_SIZE - 1; i++) {
        logTemp[i] = logTemp[i+1]; logHum[i] = logHum[i+1]; logGas[i] =
logGas[i+1]; logPct[i] = logPct[i+1];
      }
      logTemp[LOG_SIZE - 1] = temperature; logHum[LOG_SIZE - 1] =
humidity;
      logGas[LOG_SIZE - 1] = (float)airQuality; logPct[LOG_SIZE - 1] =
(float)lastPct;
      lastLogTime = millis();
    }
  }

  bool getClippedRect(int &y, int &h) {
    if (y >= STATUS_BAR_Y) return false; if (y + h <= 0) return false;
    if (y + h > STATUS_BAR_Y) h = STATUS_BAR_Y - y; if (h <= 0) return
false;
    return true;
  }

  void drawLiveGraph(int x, int rawY, int w, int rawH, float *data,
uint16_t color, String title, String unit, String xLabel, String
yLabel, float minVal, float maxVal) {
    int y = rawY; int h = rawH;
    if (y >= STATUS_BAR_Y || y + h <= 0) return; if (!getClippedRect(y,
h)) return;
    tft.fillRect(x, y, w, h, C_BG);
    tft.setTextSize(1); tft.setTextColor(C_TEXT_G, C_BG);
tft.setCursor(x, y); tft.print(title);
    tft.setTextColor(color, C_BG); tft.setCursor(x + 130, y);
    float currentVal = data[LOG_SIZE-1]; tft.print(currentVal, 1);
tft.print(" "); tft.print(unit);
    int gX = x + 25; int gY = y + 12; int gW = w - 30; int gH = h - 22;
    if (gH < 10) return;
    tft.drawFastVLine(gX, gY, gH, C_TEXT_W); tft.drawFastHLine(gX, gY +
gH, gW, C_TEXT_W);
    tft.setTextColor(C_TEXT_G, C_BG); tft.setCursor(x, gY);
tft.print((int)maxVal); tft.setCursor(x, gY + gH - 6);
tft.print((int)minVal);
    int xStep = gW / LOG_SIZE;
    for (int i = 0; i < LOG_SIZE - 1; i++) {
      float val1 = constrain(data[i], minVal, maxVal); float val2 =
constrain(data[i+1], minVal, maxVal);
      int yMap1 = map((long)(val1 * 10), (long)(minVal * 10),
(long)(maxVal * 10), 0, gH);
      int yMap2 = map((long)(val2 * 10), (long)(minVal * 10),
(long)(maxVal * 10), 0, gH);
      int y1 = gY + gH - yMap1; int y2 = gY + gH - yMap2;
```

```
      if (y1 < STATUS_BAR_Y && y2 < STATUS_BAR_Y && y1 > 0 && y2 > 0) {
tft.drawLine(gX + (i * xStep), y1, gX + ((i + 1) * xStep), y2,
color); }
    }
  }

  void drawChart(int mode) {
    tft.fillRect(0, 55, 240, 200, C_BG);
    tft.setCursor(10, 60); tft.setTextSize(1);
tft.setTextColor(C_TEXT_W);
    tft.print("FILE: "); tft.println(selectedLogFile.substring(1, 15));
    tft.setCursor(10, 75);
    // [MODIFIED] Update Text
    if (mode == 0) tft.print("GRAPH: WATER LEVEL (cm)"); else
tft.print("GRAPH: GAS LEVEL ( )");
    int gY = 90; tft.drawRect(GRAPH_X, gY, GRAPH_W, GRAPH_H,
C_METRO_GRAY);
    if (dataCount < 2) { tft.setCursor(80, 130); tft.print("No Data");
return; }
    float minVal = 9999, maxVal = -9999;
    for (int i = 0; i < dataCount; i++) {
      float val = (mode == 0) ? graphDist[i] : (float)graphGas[i];
      if (val < minVal) minVal = val; if (val > maxVal) maxVal = val;
    }
    if (maxVal == minVal) maxVal += 10; float range = maxVal - minVal;
    tft.setTextSize(1); tft.setTextColor(C_TEXT_G); tft.setCursor(0,
gY); tft.print((int)maxVal); tft.setCursor(0, gY + GRAPH_H - 8);
tft.print((int)minVal);
    int xStep = GRAPH_W / MAX_POINTS;
    for (int i = 0; i < dataCount - 1; i++) {
      float val1 = (mode == 0) ? graphDist[i] : (float)graphGas[i];
      float val2 = (mode == 0) ? graphDist[i+1] : (float)graphGas[i+1];
      int y1 = gY + GRAPH_H - (int)((val1 - minVal) / range * GRAPH_H);
      int y2 = gY + GRAPH_H - (int)((val2 - minVal) / range * GRAPH_H);
      int x1 = GRAPH_X + (i * xStep); int x2 = GRAPH_X + ((i + 1) *
xStep);
      uint16_t color = (mode == 0) ? C_METRO_BLUE : C_METRO_RED;
      tft.drawLine(x1, y1, x2, y2, color);
    }
    tft.setCursor(10, 200); tft.setTextColor(C_TEXT_G);
tft.print("UP/DWN:Switch  BACK:List");
  }

  void savePumpCycleToCSV() {
    String pFileName = "/PumpLogs.csv";
    File file = SD.open(pFileName, FILE_APPEND);
    if (!file) file = SD.open(pFileName, FILE_WRITE);
    if (file) {
      if (file.size() == 0) file.println("Waktu,Status,EventColor");
      for(int i=0; i<10; i++) {
         if (pumpLogs[i].status.length() > 0) {
           file.print(pumpLogs[i].timeStr); file.print(",");
file.print(pumpLogs[i].status); file.print(",");
file.println(pumpLogs[i].color);
         }
      }
      file.close();
    }
  }

  void exportManualSnapshot() {
```

```
  tft.fillRect(40, 100, 160, 60, C_METRO_BLUE);
tft.setTextColor(C_TEXT_W); tft.setCursor(60, 120);
tft.print("SAVING...");
  dumpRamToSD(); savePumpCycleToCSV();
  tft.fillScreen(C_OK); tft.setTextColor(C_TEXT_B);
tft.setTextSize(2); tft.setCursor(50, 140); tft.print("DATA SAVED!");
  digitalWrite(BUZZER_PIN, HIGH); delay(200);
digitalWrite(BUZZER_PIN, LOW); delay(1000); needsFullRedraw = true;
}

void drawCombinedLogView(int scroll) {
  //   [MODIFIED] Log View sekarang menampilkan LEVEL AIR
  drawLiveGraph(10, 5 - scroll, 220, 60, logPct, C_METRO_BLUE, "LEVEL
AIR (%)", "%", "", "", 0.0, 100.0);
  drawLiveGraph(10, 70 - scroll, 220, 60, logGas, C_WARN, "GAS ( )",
"", "", "", 0.0, 1000.0);
  drawLiveGraph(10, 135 - scroll, 220, 60, logTemp, C_METRO_ORANGE,
"SUHU (C)", "C", "", "", 20.0, 50.0);
  int tableY = 210 - scroll; int tableH = 200;
  if (tableY < STATUS_BAR_Y && (tableY + tableH) > 0) {
      tft.fillRect(10, tableY, 220, 25, C_METRO_PURPLE);
      tft.setTextColor(C_TEXT_W); tft.setCursor(20, tableY + 8);
tft.print("LOG SIKLUS POMPA (RAM)");
      int rowY = tableY + 30; bool hasData = false;
      for (int i = 0; i < 8; i++) {
         if (rowY >= STATUS_BAR_Y) break;
         if (pumpLogs[i].status.length() > 0 && rowY > 0) {
           hasData = true;
           tft.drawRect(10, rowY, 220, 20, C_METRO_GRAY);
           tft.setTextColor(C_TEXT_G, C_BG); tft.setCursor(15, rowY +
5); tft.print(pumpLogs[i].timeStr);
           tft.setTextColor(pumpLogs[i].color, C_BG);
tft.setCursor(80, rowY + 5); tft.print(pumpLogs[i].status);
         }
         rowY += 22;
      }
      if (!hasData && rowY < STATUS_BAR_Y && rowY > 0) {
tft.setTextColor(C_METRO_GRAY, C_BG); tft.setCursor(60, rowY);
tft.print("- No Events -"); }
  }
}

String readButtons() {
  int val = analogRead(AD_PIN);
  if (val < 200) return "IDLE";
  if (val > 3800) return "UP";
  if (val > 1400) return "DOWN";
  if (val > 700)  return "SELECT";
  if (val > 300)  return "BACK";
  return "IDLE";
}

char getNextChar(char c, bool up) {
  const String validChars = " ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789";
  int idx = validChars.indexOf(c); if (idx == -1) idx = 0;
  if (up) { idx++; if (idx >= validChars.length()) idx = 0; }
  else { idx--; if (idx < 0) idx = validChars.length() - 1; }
  return validChars[idx];
}

void handleLaserMorse() {
```

```
  if (inSubMenu && (menuIndex >= 0 && menuIndex <= 1)) {
digitalWrite(LASER_PIN, LOW); return; }
  if (millis() - lastMorseTime >= morseADI[morseStep].duration) {
    lastMorseTime = millis(); morseStep++;
    if (morseStep >= totalMorseSteps) morseStep = 0;
    digitalWrite(LASER_PIN, morseADI[morseStep].state);
  }
}

//   [MODIFIED] OLED UPDATE (STATUS TEXT AMAN/WASPADA/BAHAYA + ERROR
LIMIT)
void updateOLED(int pct, float levelAir) {
  oled.clearDisplay(); oled.drawRect(0, 0, 8, 32, SSD1306_WHITE);
  int h = map(constrain(pct, 0, 100), 0, 100, 0, 28);
  if(h > 0) oled.fillRect(2, 30-h, 4, h, SSD1306_WHITE);

  oled.setTextColor(SSD1306_WHITE);

  if (calibError) {
    // Tampilan Error di OLED
    oled.setTextSize(1); oled.setCursor(12, 5); oled.print("ERR:
LOW");
    oled.setCursor(12, 15); oled.print("LIMIT");
  } else if (distance > 600) {
    oled.setTextSize(2); oled.setCursor(20, 10); oled.print("N/A");
  } else {
    oled.setTextSize(2); oled.setCursor(12, 2); oled.print(pct);
oled.print("%");
    oled.setTextSize(1); oled.setCursor(12, 20);
    oled.print((int)levelAir); oled.print(" cm"); // Menampilkan
Tinggi Air
  }

  oled.drawFastVLine(62, 0, 32, SSD1306_WHITE);

  // Layout Kanan OLED (4 Baris)
  int xPos = 66;
  oled.setTextSize(1);
  oled.setCursor(xPos, 0); oled.print("T:"); oled.print(temperature,
0);
  oled.setCursor(xPos, 8); oled.print("P:");
oled.print((int)pressure);

  oled.setCursor(xPos, 16);
  if (millis() < MQ_WARMUP_TIME) { oled.print("G:HEAT"); } else {
oled.print("G:"); oled.print(airQuality); }

  oled.setCursor(xPos, 24);
  if (millis() < MQ_WARMUP_TIME) { oled.print("WAIT.."); }
  else {
    if(airQuality <= 1250) oled.print("AMAN");
    else if(airQuality <= 2100) oled.print("WASPADA");
    else oled.print("BAHAYA!");
  }
  oled.display();
}

void drawGasAlert() {
  if (!wasGasAlertActive) {
    tft.fillScreen(C_ALERT);
```

```
      int centerX = 120; int centerY = 100;
      tft.fillTriangle(centerX, centerY-60, centerX-70, centerY+60,
centerX+70, centerY+60, 0xFFE0);
      tft.fillRect(centerX-5, centerY-30, 10, 50, C_TEXT_B);
      tft.fillCircle(centerX, centerY+40, 6, C_TEXT_B);

      tft.setTextColor(C_TEXT_W); tft.setTextSize(3);
      tft.setCursor(50, 200); tft.print("WARNING!");
      tft.setTextSize(2);
      tft.setCursor(20, 240); tft.print("GAS LEAK DETECTED");
    }

    tone(BUZZER_PIN, 2000); delay(150);
    tone(BUZZER_PIN, 1500); delay(150);
    noTone(BUZZER_PIN);
  }

  //   [MODIFIED] BACA SENSOR (DENGAN LOGIKA KETINGGIAN AIR & MODE
RELAY BARU)
  void readSensors() {
    // 1. Ultrasonik
    digitalWrite(TRIG_PIN, LOW); delayMicroseconds(2);
    digitalWrite(TRIG_PIN, HIGH); delayMicroseconds(10);
    digitalWrite(TRIG_PIN, LOW);
    long duration = pulseIn(ECHO_PIN, HIGH, 40000);
    distance = (duration == 0) ? 0 : (duration * 0.034 / 2);

    //   [NEW LOGIC] HITUNG KETINGGIAN AIR
    waterLevelCM = (float)setLow - distance;

    // Cek Logika Kacau (Jika sensor baca > setLow)
    if (distance > setLow || waterLevelCM < 0) {
        calibError = true;
        waterLevelCM = 0; // Force 0 agar tidak minus
    } else {
        calibError = false;
    }
    //

    // 2. Env Sensors
    humidity = dht.readHumidity();
    temperature = dht.readTemperature();
    pressure = bmp.readPressure() / 100.0F;

    // 3. Gas
    airQuality = analogRead(MQ135_PIN);
    if (millis() < MQ_WARMUP_TIME) { isGasAlertActive = false; }
    else { if (airQuality > 2100) isGasAlertActive = true; else
isGasAlertActive = false; }

    // 4. Battery
    int raw = analogRead(BATT_PIN);
    float v = (raw / 4095.0) * 3.3 * 2.10;
    battPercent = (v > 2.0) ? map(constrain(v * 100, 330, 420), 330,
420, 0, 100) : 0;

    // 5. Pct Calculation
    float pctCalc = 0;
    if (setLow != setFull) {
        // [MODIFIED] Rumus persentase pakai waterLevelCM
```

```
      if (!calibError) pctCalc = (waterLevelCM / (float)(setLow -
setFull)) * 100.0;
  }
  int currentPct = constrain((int)pctCalc, 0, 100);
  lastPct = currentPct;

  // 6. Logic Pump (Auto)
  // [MODIFIED] UPDATE FILLING / DISCHARGING Logic
  if (!isOverride && !calibError) {
      if (relayMode == 0) {
          //   MODE FILLING (PENGISIAN)
          // Nyala jika Kering (< Low), Mati jika Penuh (> High)
          if (currentPct <= setLimitLow && !pumpState) { pumpState =
true; needsStatusUpdate = true; }
          else if (currentPct >= setLimitHigh && pumpState) {
pumpState = false; needsStatusUpdate = true; }
      }
      else {
          //   MODE DISCHARGE (PENGOSONGAN)
          // Nyala jika Penuh (> High), Mati jika Kering (< Low)
          if (currentPct >= setLimitHigh && !pumpState) { pumpState =
true; needsStatusUpdate = true; }
          else if (currentPct <= setLimitLow && pumpState) {
pumpState = false; needsStatusUpdate = true; }
      }
  } else if (calibError && !isOverride) {
      pumpState = false; // Force Stop Pump jika pembacaan kacau
  }

  // 7. Radio Send
  sensorData.dist = waterLevelCM; // [MODIFIED] Kirim ketinggian air
ke RF
  sensorData.pct = currentPct;
  sensorData.temp = temperature; sensorData.hum = humidity;
  sensorData.airQuality = airQuality; sensorData.pump = pumpState;

  if (radio.write(&sensorData, sizeof(DataPacket))) {
    rfStrongSignal = radio.testRPD();
    if (radio.isAckPayloadAvailable()) { radio.read(&feedbackData,
sizeof(FeedbackPacket)); rfConnected = true; lastRfRecv = millis(); }
    else { rfConnected = true; lastRfRecv = millis(); }
  } else rfConnected = false;

  if (millis() - lastRfRecv > 2500) rfConnected = false;

  checkPumpEvents();
  updateOLED(currentPct, waterLevelCM);
  logDataToSD();
  needsStatusUpdate = true;
}

void drawStatusBar() {
  tft.fillRect(0, STATUS_BAR_Y, 240, 320 - STATUS_BAR_Y, C_BG);
  tft.drawFastHLine(0, STATUS_BAR_Y, 240, C_METRO_GRAY);
  int baseY = STATUS_BAR_Y + 5;

  tft.fillCircle(10, baseY + 8, 4, C_OK);
  tft.setTextSize(1); tft.setTextColor(C_TEXT_G);
  tft.setCursor(18, baseY + 5); tft.print("WEB:OK");

  int sigX = 65; int sigY = baseY + 14;
```

```
  tft.setTextColor(C_TEXT_W); tft.setCursor(sigX, baseY + 5);
tft.print("RF");
  int barW = 3; int gap = 2; int startX = sigX + 15;
  uint16_t sigColor = (!rfConnected) ? C_METRO_GRAY : (rfStrongSignal
? C_OK : C_WARN);

  tft.fillRect(startX, sigY - 4, barW, 4, rfConnected ? sigColor :
C_METRO_GRAY);
  tft.fillRect(startX + (barW+gap), sigY - 7, barW, 7, rfConnected ?
sigColor : C_METRO_GRAY);
  tft.fillRect(startX + (barW+gap)*2, sigY - 10, barW, 10,
(rfConnected && (rfStrongSignal || rfConnected)) ? sigColor :
C_METRO_GRAY);
  tft.fillRect(startX + (barW+gap)*3, sigY - 13, barW, 13,
(rfConnected && rfStrongSignal) ? sigColor : C_METRO_GRAY);

  uint16_t pumpIconColor = (!rfConnected) ? C_METRO_GRAY : (pumpState
? C_OK : C_METRO_RED);
  int px = 115, py = baseY + 9;
  tft.fillCircle(px, py, 7, pumpIconColor);
  tft.setTextColor(C_TEXT_W); tft.setCursor(px - 3, py - 3);
tft.print("P");

  // FEEDBACK
  int fx = 138, fy = baseY + 2;
  bool isFlowing = (rfConnected && feedbackData.sensorStatus == 1);
  if (isFlowing) { tft.fillRect(fx, fy, 14, 14, C_METRO_BLUE);
tft.setTextColor(C_TEXT_W); }
  else { tft.drawRect(fx, fy, 14, 14, C_METRO_GRAY);
tft.setTextColor(C_METRO_GRAY); }
  tft.setCursor(fx + 4, fy + 3); tft.print("F");

  struct tm timeinfo;
  tft.fillRect(170, baseY, 70, 25, C_BG);
  tft.setTextColor(C_TEXT_W); tft.setTextSize(2); tft.setCursor(175,
baseY + 1);
  if(!getLocalTime(&timeinfo)) tft.print("--:--");
  else tft.printf("%02d:%02d", timeinfo.tm_hour, timeinfo.tm_min);

  needsStatusUpdate = false;
}

void displayMenu() {
  if (needsFullRedraw) {
    tft.fillScreen(C_BG); tft.fillRect(0, 0, 240, 50, C_BG);
    tft.setTextColor(C_METRO_BLUE); tft.setTextSize(2);
tft.setCursor(35, 18); tft.print("DEVICE CONTROL");
    tft.drawFastHLine(20, 45, 200, C_METRO_BLUE);
    needsFullRedraw = false; needsMenuUpdate = true;
needsStatusUpdate = true; lastDrawnMenuIndex = -1;
  }
  if (needsMenuUpdate) {
    String menus[] = {"TANK CALIB", "RELAY SET", "DASHBOARD", "VIEW
LOGS", "PUMP CYCLES", "RF RANGE SET"};
    int menuCount = 6;
    for (int i = 0; i < menuCount; i++) {
      bool isSelected = (i == menuIndex);
      if (lastDrawnMenuIndex == -1 || isSelected || (i ==
lastDrawnMenuIndex)) {
        int yPos = 60 + (i * 38);
```

```
        if (isSelected) { tft.fillRoundRect(10, yPos, 220, 32, 2,
C_METRO_BLUE); tft.setTextColor(C_TEXT_W); }
        else { tft.fillRoundRect(10, yPos, 220, 32, 2, C_METRO_GRAY);
tft.setTextColor(C_TEXT_G); }
        tft.setTextSize(2); tft.setCursor(25, yPos + 8);
tft.print(menus[i]);
      }
    }

    tft.setCursor(15, 275);
    tft.setTextColor(C_TEXT_G, C_BG); tft.setTextSize(1);
    tft.print("IP: "); tft.print(WiFi.localIP());

    lastDrawnMenuIndex = menuIndex; needsMenuUpdate = false;
  }
}

void handleSubMenu(String btn) {
  static int lastDisplayedVal1 = -999; static int lastDisplayedVal2 =
-999; static float lastDistDraw = -1.0;
  static int subCursor = 0; static int lastSubCursor = -1;
  static int fileCursor = 0;
  static int lastDrawnRF = -1;

  if (needsFullRedraw) {
    if (menuIndex != 2) tft.fillScreen(C_BG); else tft.fillRect(0, 0,
240, STATUS_BAR_Y, C_BG);
    needsFullRedraw = false; needsStatusUpdate = true;
    l_temp = -99; l_hum = -99; l_dist = -99; l_pump_state = -1;
l_rf_stat = !rfConnected;
    lastDisplayedVal1 = -999; lastDisplayedVal2 = -999; lastDistDraw
= -1.0; subCursor = 0; lastSubCursor = -1;
    lastDrawnRF = -1;
    needsLogRedraw = true;
  }

  //   MENU 0: CALIB
  if (menuIndex == 0) {
      if (lastSubCursor == -1) {
        tft.setTextColor(C_METRO_BLUE, C_BG); tft.setTextSize(2);
tft.setCursor(30, 15); tft.print("CALIBRATION"); tft.drawFastHLine(0,
45, 240, C_METRO_GRAY);
      }
      if (btn == "SELECT") {
          unsigned long pressStart = millis(); bool longPressDetected
= false;
          while (analogRead(AD_PIN) > 700) { if (millis() -
pressStart > 1500) { longPressDetected = true; break; } delay(50); }

          if (longPressDetected) {
              //   [DIPERBAIKI] SIMPAN MENGGUNAKAN EEPROM.put AGAR
BISA > 250
              int* targetVal = (subCursor == 0) ? &setFull : &setLow;
              *targetVal = (int)distance;

              // Simpan ke EEPROM menggunakan .put()
              EEPROM.put(0, setFull);
              EEPROM.put(10, setLow);
              EEPROM.commit(); // Wajib commit agar tersimpan
permanen
```

```
                tft.fillRoundRect(30, 90, 180, 80, 8, C_OK);
tft.setTextColor(C_TEXT_B); tft.setTextSize(2);
                tft.setCursor(50, 110); tft.print("CALIBRATED!");
tft.setCursor(65, 135); tft.print("SAVED.");
                playTone(3); delay(1000); tft.fillScreen(C_BG);
needsFullRedraw = true; return;
            } else { subCursor = !subCursor; }
        }

        int* targetVal = (subCursor == 0) ? &setFull : &setLow;
        if (btn == "UP") (*targetVal)++; if (btn == "DOWN")
(*targetVal)--;

        if (subCursor != lastSubCursor || lastSubCursor == -1) {
            uint16_t col1 = (subCursor == 0) ? C_METRO_BLUE :
C_METRO_GRAY; tft.fillRoundRect(10, 60, 220, 50, 4, col1);
            uint16_t col2 = (subCursor == 1) ? C_METRO_BLUE :
C_METRO_GRAY; tft.fillRoundRect(10, 120, 220, 50, 4, col2);
            tft.setTextColor((subCursor == 0) ? C_TEXT_W : C_TEXT_G,
col1); tft.setTextSize(2); tft.setCursor(20, 65); tft.print("FULL
(TOP)");
            tft.setTextColor((subCursor == 1) ? C_TEXT_W : C_TEXT_G,
col2); tft.setTextSize(2); tft.setCursor(20, 125); tft.print("LOW
(BTM)");
            lastDisplayedVal1 = -999; lastDisplayedVal2 = -999;
lastSubCursor = subCursor;
        }
        uint16_t bg1 = (subCursor == 0) ? C_METRO_BLUE : C_METRO_GRAY;
uint16_t txt1 = (subCursor == 0) ? C_TEXT_W : C_TEXT_G;
        if (setFull != lastDisplayedVal1) { tft.setTextColor(txt1,
bg1); tft.setTextSize(3); tft.setCursor(140, 75); tft.print(setFull);
tft.setTextSize(2); tft.print("cm "); lastDisplayedVal1 = setFull; }
        uint16_t bg2 = (subCursor == 1) ? C_METRO_BLUE : C_METRO_GRAY;
uint16_t txt2 = (subCursor == 1) ? C_TEXT_W : C_TEXT_G;
        if (setLow != lastDisplayedVal2) { tft.setTextColor(txt2, bg2);
tft.setTextSize(3); tft.setCursor(140, 135); tft.print(setLow);
tft.setTextSize(2); tft.print("cm "); lastDisplayedVal2 = setLow; }

        tft.drawFastHLine(20, 200, 200, C_METRO_GRAY);
        tft.setTextColor(C_METRO_BLUE, C_BG);
        tft.setTextSize(2);
        tft.setCursor(20, 215);
        tft.print("RAW SENSOR:");

        if (abs(distance - lastDistDraw) > 0.1 || (distance > 600 &&
lastDistDraw <= 600) || (distance <= 600 && lastDistDraw > 600)) {
            tft.setCursor(168, 215);
            tft.setTextColor(C_TEXT_W, C_BG);
            if (distance > 600) {
               tft.print("N/A    ");
            } else {
               tft.print((int)distance); tft.print(" cm ");
            }
            lastDistDraw = distance;
        }
    }

    //   MENU 1: RELAY SET
    else if (menuIndex == 1) {
        if (lastSubCursor == -1) {
          tft.fillRect(0, 0, 240, 320, C_BG);
```

```
            tft.setTextColor(C_METRO_BLUE, C_BG); tft.setTextSize(2);
            tft.setCursor(20, 15); tft.print("RELAY CONTROL");
            tft.drawFastHLine(0, 45, 240, C_METRO_GRAY);

            lastSubCursor = 0;
            subCursor = 0;
            btn = "FORCE_DRAW";
          }

          if (btn == "SELECT") {
              subCursor++;
              if(subCursor > 2) subCursor = 0;
          }

          if (btn == "UP") {
              if (subCursor == 0) { relayMode = !relayMode; }
              else if (subCursor == 1) { setLimitHigh++;
    if(setLimitHigh>100) setLimitHigh=100; }
              else if (subCursor == 2) { setLimitLow++;
    if(setLimitLow>setLimitHigh-5) setLimitLow=setLimitHigh-5; }
          }
          if (btn == "DOWN") {
              if (subCursor == 0) { relayMode = !relayMode; }
              else if (subCursor == 1) { setLimitHigh--;
    if(setLimitHigh<setLimitLow+5) setLimitHigh=setLimitLow+5; }
              else if (subCursor == 2) { setLimitLow--; if(setLimitLow<0)
    setLimitLow=0; }
          }

          if (btn != "IDLE" || lastSubCursor == -1) {
              int itemX = 10;

              // Item 0: MODE
              int y0 = 60;
              tft.setTextColor(subCursor==0 ? C_TEXT_W : C_TEXT_G,
    subCursor==0 ? C_METRO_BLUE : C_BG);
              tft.setTextSize(1); tft.setCursor(itemX, y0);
    tft.print("CONTROL MODE:");
              tft.setTextSize(2); tft.setCursor(itemX, y0+15);
              if(relayMode == 0) tft.print("FILLING  ");
              else tft.print("DISCHARGE");

              // Item 1: HIGH LEVEL
              int y1 = 120;
              tft.setTextColor(subCursor==1 ? C_TEXT_W : C_TEXT_G,
    subCursor==1 ? C_METRO_BLUE : C_BG);
              tft.setTextSize(1); tft.setCursor(itemX, y1);
    tft.print("HIGH LEVEL (%):");
              tft.setTextSize(2); tft.setCursor(itemX, y1+15);
    tft.print(setLimitHigh); tft.print("% ");

              // Item 2: LOW LEVEL
              int y2 = 180;
              tft.setTextColor(subCursor==2 ? C_TEXT_W : C_TEXT_G,
    subCursor==2 ? C_METRO_BLUE : C_BG);
              tft.setTextSize(1); tft.setCursor(itemX, y2);
    tft.print("LOW LEVEL (%):");
              tft.setTextSize(2); tft.setCursor(itemX, y2+15);
    tft.print(setLimitLow); tft.print("% ");

              int tx = 140; int ty = 70; int tw = 80; int th = 140;
```

```
            tft.fillRect(tx-10, ty-10, tw+20, th+20, C_BG);
            tft.drawRect(tx, ty, tw, th, C_TEXT_W);
            tft.drawLine(tx, ty, tx+tw/2, ty-10, C_TEXT_W);
            tft.drawLine(tx+tw, ty, tx+tw/2, ty-10, C_TEXT_W);

            int pxHigh = ty + th - map(setLimitHigh, 0, 100, 0, th);
            int pxLow  = ty + th - map(setLimitLow, 0, 100, 0, th);

            tft.fillRect(tx+2, pxHigh, tw-4, pxLow-pxHigh, 0x18E3);
            for(int i=tx; i<tx+tw; i+=4) tft.drawPixel(i, pxHigh,
C_METRO_GREEN);
            tft.setCursor(tx+tw+2, pxHigh-3); tft.setTextSize(1);
tft.setTextColor(C_METRO_GREEN); tft.print("H");

            for(int i=tx; i<tx+tw; i+=4) tft.drawPixel(i, pxLow,
C_METRO_RED);
            tft.setCursor(tx+tw+2, pxLow-3); tft.setTextSize(1);
tft.setTextColor(C_METRO_RED); tft.print("L");

          tft.setCursor(tx+5, ty+th+10); tft.setTextColor(C_TEXT_G);
            if(relayMode == 0) tft.print("IN -> [Tank]");
            else tft.print("[Tank] -> OUT");

            lastSubCursor = subCursor;
        }
    }

    //   MENU 2: DASHBOARD
    else if (menuIndex == 2) {
      updateEnvLog(); bool isScrolling = false;
      if (btn == "DOWN") { if (scrollY < MAX_SCROLL) { scrollY += 20;
isScrolling = true; needsFullRedraw = true; } }
      if (btn == "UP") { if (scrollY > 0) { scrollY -= 20; isScrolling
= true; needsFullRedraw = true; } }
      int yPos, h, drawH;
      yPos = 5 - scrollY; h = 90; drawH = h;
      if (getClippedRect(yPos, drawH)) {
          if (distance != l_dist || isScrolling || calibError) {
            tft.fillRect(5, yPos, 112, drawH, calibError ? C_METRO_RED
: C_METRO_BLUE);
            tft.setTextColor(C_TEXT_W);
            if (drawH > 20) {
                tft.setTextSize(1); tft.setCursor(10, yPos + 10);
                tft.print(calibError ? "CALIB ERROR!" : "WATER DEPTH");
            }
            if (drawH > 50) {
              tft.setTextSize(4); tft.setCursor(10, yPos + 35);
              if (distance > 600) {
                  tft.print("N/A");
              } else if (calibError) {
                  tft.setTextSize(2); tft.setCursor(10, yPos + 45);
                  tft.print("LOW LIMIT");
              } else {
                  tft.print(waterLevelCM, 0);
              }
              if(!calibError) { tft.setTextSize(2); tft.print("cm"); }
            }
            l_dist = distance;
          }
```

```
        bool currentPumpDisplay = rfConnected ? feedbackData.isPumpOn
: pumpState;
        int currentPumpColor = currentPumpDisplay ? C_METRO_GREEN :
C_METRO_RED;
        if (l_pump_state != (int)currentPumpDisplay || l_rf_stat !=
rfConnected || isScrolling) {
          tft.fillRect(123, yPos, 112, drawH, currentPumpColor);
tft.setTextColor(C_TEXT_W);
          if (drawH > 20) { tft.setTextSize(1); tft.setCursor(128,
yPos + 10); tft.print("PUMP STATUS"); }
          if (drawH > 60) {
              tft.setTextSize(3); tft.setCursor(128, yPos + 40);
tft.print(currentPumpDisplay ? "RUN" : "STOP");
              tft.setTextSize(1); tft.setCursor(128, yPos + 75);
tft.print(rfConnected ? "RF: OK" : "RF: --");
          }
          l_pump_state = (int)currentPumpDisplay; l_rf_stat =
rfConnected;
        }
    }
    yPos = 100 - scrollY; h = 70; drawH = h;
    if (getClippedRect(yPos, drawH)) {
        if (temperature != l_temp || isScrolling) {
          tft.fillRect(5, yPos, 112, drawH, C_METRO_ORANGE);
tft.setTextColor(C_TEXT_W);
          if(drawH > 20) { tft.setTextSize(1); tft.setCursor(10, yPos
+ 10); tft.print("TEMP"); }
          if(drawH > 50) { tft.setTextSize(3); tft.setCursor(10, yPos
+ 30); tft.print(temperature, 1); tft.setTextSize(1); tft.print("
C"); }
          l_temp = temperature;
        }
        if (humidity != l_hum || isScrolling) {
          tft.fillRect(123, yPos, 112, drawH, C_METRO_PURPLE);
tft.setTextColor(C_TEXT_W);
          if(drawH > 20) { tft.setTextSize(1); tft.setCursor(128,
yPos + 10); tft.print("HUMIDITY"); }
          if(drawH > 50) { tft.setTextSize(3); tft.setCursor(128,
yPos + 30); tft.print((int)humidity); tft.setTextSize(1); tft.print("
%"); }
          l_hum = humidity;
        }
    }
    yPos = 175 - scrollY; h = 60; drawH = h;
    static unsigned long lastInfoDraw = 0;
    if (getClippedRect(yPos, drawH)) {
        if (millis() - lastInfoDraw > 1000 || isScrolling) {
          tft.fillRect(5, yPos, 230, drawH, C_METRO_GRAY);
tft.setTextColor(C_TEXT_W);
          if(drawH > 15) { tft.setTextSize(1); tft.setCursor(10, yPos
+ 5); tft.print("ENVIRONMENT DATA"); }
          if(drawH > 40) {
              tft.setCursor(10, yPos + 20); tft.print("PRESSURE: ");
tft.print((int)pressure); tft.print(" hPa");
              tft.setCursor(10, yPos + 35); tft.print("AIR GAS : ");
              if (millis() < MQ_WARMUP_TIME) {
tft.setTextColor(C_METRO_ORANGE); tft.print("HEATING");
tft.setTextColor(C_TEXT_W); }
              else {
                  tft.print(airQuality);
                  String stsText; uint16_t stsColor;
```

```
                    if(airQuality <= 1250) { stsText="AMAN";
stsColor=C_OK; }
                    else if(airQuality <= 2100) { stsText="WASPADA";
stsColor=C_WARN; }
                    else { stsText="BAHAYA"; stsColor=C_METRO_RED; }
                    tft.setTextColor(stsColor); tft.print("(" + stsText
+ ")");
                }
                tft.setCursor(140, yPos + 20);
                if(rfConnected && feedbackData.sensorStatus == 1) {
tft.setTextColor(C_METRO_BLUE); tft.print("FBACK: OK"); }
                else { tft.setTextColor(C_METRO_RED); tft.print("FBACK:
--"); }
            }
            lastInfoDraw = millis();
          }
      }
      if ((millis() - lastLogTime < 100) || isScrolling) {
         drawLiveGraph(10, 245 - scrollY, 220, 65, logTemp,
C_METRO_ORANGE, "TEMP HISTORY", "C", "Time", "Tmp", 20.0, 45.0);
         drawLiveGraph(10, 315 - scrollY, 220, 65, logHum,
C_METRO_PURPLE, "HUMIDITY HIST", "%", "Time", "Hum", 0.0, 100.0);
         drawLiveGraph(10, 385 - scrollY, 220, 65, logGas, C_WARN, "GAS
LEVEL ( )", "", "Time", "Gas", 0.0, 1000.0);
         drawLiveGraph(10, 455 - scrollY, 220, 65, logPct,
C_METRO_BLUE, "TANK LEVEL (%)", "%", "Time", "Lvl", 0.0, 100.0);
      }
    }

    //   MENU 3: VIEW LOGS
    else if (menuIndex == 3) {
        if (needsFullRedraw) {
            tft.fillScreen(C_BG); tft.setTextColor(C_METRO_BLUE);
tft.setCursor(60, 140); tft.print("Auto-Saving...");
            dumpRamToSD(); tft.fillScreen(C_BG); isFileView = false;
needsLogRedraw = true; needsFullRedraw = false; scrollY = 0;
        }
        if (!isFileView) {
            if (btn == "DOWN") { scrollY += 20; if(scrollY > 200)
scrollY = 200; tft.fillRect(0, 0, 240, STATUS_BAR_Y, C_BG);
needsLogRedraw = true; }
            if (btn == "UP") { scrollY -= 20; if(scrollY < 0) scrollY =
0; tft.fillRect(0, 0, 240, STATUS_BAR_Y, C_BG); needsLogRedraw =
true; }
            if (btn == "SELECT") {
                unsigned long pressStart = millis();
                while(analogRead(AD_PIN) > 700) { if(millis() -
pressStart > 1500) { exportManualSnapshot(); btn = "IDLE"; break; }
delay(50); }
            }
            if (btn == "BACK") {
                unsigned long pressStart = millis(); bool longPress =
false;
                while(analogRead(AD_PIN) > 300 && analogRead(AD_PIN) <
700) { if(millis() - pressStart > 1000) { longPress = true; break; }
delay(50); }
                if (longPress) { isFileView = true; scanLogFiles();
needsLogRedraw = true; btn = "IDLE"; tft.fillScreen(C_BG); }
            }
            if (needsLogRedraw || millis() - lastLogTime < 200) {
drawCombinedLogView(scrollY); needsLogRedraw = false; }
```

```
        }
        else { // File Explorer
            if (needsLogRedraw) {
                tft.fillScreen(C_BG); tft.setTextColor(C_METRO_BLUE,
C_BG); tft.fillRect(0, 0, 240, 30, C_BG);
                tft.setCursor(10, 10); tft.setTextSize(2);
tft.print("SD CARD EXPLORER"); tft.drawFastHLine(0, 35, 240,
C_METRO_GRAY);
                if (sdFileCount == 0) { tft.setCursor(50, 100);
tft.setTextColor(C_METRO_RED); tft.print("FILE KOSONG"); }
                else {
                    for(int i=0; i<sdFileCount; i++) {
                        int yPos = 50 + (i * 25); if (yPos > 280)
break;
                        if (i == fileCursor) { tft.fillRect(10, yPos,
220, 22, C_METRO_BLUE); tft.setTextColor(C_TEXT_W); }
                        else { tft.fillRect(10, yPos, 220, 22, C_BG);
tft.setTextColor(C_TEXT_G); }
                        tft.setCursor(20, yPos + 5);
tft.setTextSize(1); tft.print(sdFileList[i].substring(1));
                    }
                }
                needsLogRedraw = false;
            }
            if (btn == "DOWN") { fileCursor++; if (fileCursor >=
sdFileCount) fileCursor = 0; needsLogRedraw = true; }
            if (btn == "UP")    { fileCursor--; if (fileCursor < 0)
fileCursor = sdFileCount - 1; needsLogRedraw = true; }
            if (btn == "SELECT" && sdFileCount > 0) {
                selectedLogFile = sdFileList[fileCursor];
tft.fillScreen(C_BG); loadLogData(selectedLogFile); drawChart(0);
                bool viewingChart = true;
                while(viewingChart) {
                    String b = readButtons();
                    if (b == "BACK") viewingChart = false;
                    if (b == "UP" || b == "DOWN") { graphMode =
!graphMode; drawChart(graphMode); delay(200); }
                    delay(50);
                }
                needsLogRedraw = true;
            }
            if (btn == "BACK") { isFileView = false; needsLogRedraw =
true; tft.fillScreen(C_BG); btn = "IDLE"; }
        }
    }

    //   MENU 4: PUMP LOGS
    else if (menuIndex == 4) {
        if (lastSubCursor == -1) {
            tft.setTextColor(C_METRO_BLUE, C_BG); tft.setTextSize(2);
tft.setCursor(50, 20); tft.print("PUMP LOGS");
            tft.drawFastHLine(0, 50, 240, C_METRO_BLUE);
tft.fillRect(10, 60, 220, 25, C_METRO_GRAY);
            tft.setTextColor(C_TEXT_W); tft.setTextSize(1);
tft.setCursor(20, 68); tft.print("TIME"); tft.setCursor(120, 68);
tft.print("EVENT STATUS");
            lastSubCursor = 0; needsLogRedraw = true;
        }
        if (needsLogRedraw) {
            tft.fillRect(0, 86, 240, 209, C_BG); bool empty = true;
            for (int i = 0; i < 10; i++) {
```

```
                int yRow = 90 + (i * 20); if (yRow + 20 >=
STATUS_BAR_Y) break;
                if (pumpLogs[i].status.length() > 0) {
                    empty = false;
                    tft.setTextColor(C_TEXT_G, C_BG); tft.setCursor(20,
yRow + 5); tft.print(pumpLogs[i].timeStr);
                    tft.setTextColor(pumpLogs[i].color, C_BG);
tft.setCursor(120, yRow + 5); tft.print(pumpLogs[i].status);
                    tft.drawFastHLine(10, yRow + 19, 220, 0x2104);
                }
            }
            if (empty) { tft.setTextColor(C_TEXT_G, C_BG);
tft.setCursor(80, 130); tft.print("No Events Yet"); }
            needsLogRedraw = false;
        }
    }

    // === NEW MENU: RF RANGE SET (INDEX 5) ===
    else if (menuIndex == 5) {
        if (lastSubCursor == -1) {
            tft.setTextColor(C_METRO_BLUE, C_BG); tft.setTextSize(2);
tft.setCursor(40, 15); tft.print("RF POWER SET");
            tft.drawFastHLine(0, 45, 240, C_METRO_GRAY);
            tft.setTextColor(C_TEXT_G, C_BG); tft.setTextSize(1);
            tft.setCursor(15, 180); tft.print("MIN : Jarak < 5 Meter
(Lab)");
            tft.setCursor(15, 195); tft.print("MAX : Jarak Jauh /
Tembok");
            lastSubCursor = 0;
        }

        bool powerChanged = false;
        if (btn == "UP") {
            setRFPower++; if (setRFPower > 3) setRFPower = 3;
            powerChanged = true;
        }
        if (btn == "DOWN") {
            setRFPower--; if (setRFPower < 0) setRFPower = 0;
            powerChanged = true;
        }

        if (powerChanged) {
            if(setRFPower == 0) radio.setPALevel(RF24_PA_MIN);
            else if(setRFPower == 1) radio.setPALevel(RF24_PA_LOW);
            else if(setRFPower == 2) radio.setPALevel(RF24_PA_HIGH);
            else if(setRFPower == 3) radio.setPALevel(RF24_PA_MAX);
        }

        if (powerChanged || lastDrawnRF != setRFPower) {
            int barY = 80;
            tft.fillRect(10, 60, 220, 100, C_BG);
            tft.fillRect(30, barY+45, 30, 15, (setRFPower >= 0) ?
C_METRO_GREEN : C_METRO_GRAY);
            tft.fillRect(70, barY+30, 30, 30, (setRFPower >= 1) ?
C_METRO_BLUE : C_METRO_GRAY);
            tft.fillRect(110, barY+15, 30, 45, (setRFPower >= 2) ?
C_METRO_ORANGE : C_METRO_GRAY);
            tft.fillRect(150, barY, 30, 60, (setRFPower >= 3) ?
C_METRO_RED : C_METRO_GRAY);
            tft.setTextSize(2); tft.setCursor(60, 150);
tft.setTextColor(C_TEXT_W, C_BG);
```

```
            if(setRFPower == 0) tft.print("MIN (LOW)");
            else if(setRFPower == 1) tft.print("LOW (MED)");
            else if(setRFPower == 2) tft.print("HIGH (FAR)");
            else if(setRFPower == 3) tft.print("MAX (XTREME)");
            lastDrawnRF = setRFPower;
        }
    }

    if (btn == "BACK" && !isFileView) {
      // [DIPERBAIKI] Gunakan put() agar data berapapun (termasuk >
250) aman
      EEPROM.put(0, setFull);
      EEPROM.put(10, setLow);
      EEPROM.put(20, setLimitLow);
      EEPROM.put(30, setLimitHigh);
      EEPROM.put(40, setRFPower);
      EEPROM.put(50, relayMode);
      EEPROM.commit();

      // Play sound save
      tone(BUZZER_PIN, 2000, 100);

      logLoaded = false; inSubMenu = false; needsFullRedraw = true;
needsMenuUpdate = true; needsStatusUpdate = true;
      lastDrawnMenuIndex = -1; lastSubCursor = -1; subCursor = 0;
scrollY = 0;
    }
}

void handleWifiMenu() {
  tft.fillScreen(C_BG); tft.setTextColor(C_ALERT);
tft.setTextSize(2);
  tft.setCursor(30, 20); tft.print("CONNECTION FAILED");
  tft.setTextColor(C_TEXT_W); tft.setTextSize(1); tft.setCursor(30,
45); tft.print("Please configure WiFi");
  int cursorIdx = 0; bool editing = false; String tempSSID =
wifi_ssid; String tempPASS = wifi_pass; int charPos = 0;
  while(true) {
    tft.fillRect(10, 70, 220, 150, C_BG);
    tft.drawRect(20, 80, 200, 30, cursorIdx == 0 ? C_METRO_BLUE :
C_METRO_GRAY);
    tft.setCursor(25, 90); tft.setTextColor(editing && cursorIdx==0 ?
C_OK : C_TEXT_W);
    tft.print(tempSSID); if(editing && cursorIdx == 0) {
tft.drawFastHLine(25 + (charPos*6), 100, 5, C_METRO_BLUE); }

    tft.drawRect(20, 120, 200, 30, cursorIdx == 1 ? C_METRO_BLUE :
C_METRO_GRAY);
    tft.setCursor(25, 130); tft.setTextColor(editing && cursorIdx==1
? C_OK : C_TEXT_W);
    tft.print(tempPASS); if(editing && cursorIdx == 1) {
tft.drawFastHLine(25 + (charPos*6), 140, 5, C_METRO_BLUE); }

    tft.fillRoundRect(60, 170, 120, 30, 5, cursorIdx == 2 ? C_OK :
C_METRO_GRAY);
    tft.setTextColor(C_TEXT_B); tft.setCursor(85, 180);
tft.print("CONNECT");
    tft.setTextColor(C_TEXT_G); tft.setCursor(20, 210);
    if(editing) tft.print("UP/DWN:Change SEL:Next"); else
tft.print("UP/DWN:Select SEL:Edit");
```

```
    delay(200); String btn = "IDLE"; while(btn == "IDLE") { btn =
readButtons(); delay(50); }
    if (editing) {
       String *target = (cursorIdx == 0) ? &tempSSID : &tempPASS;
       if (btn == "UP") (*target)[charPos] =
getNextChar((*target)[charPos], true);
       if (btn == "DOWN") (*target)[charPos] =
getNextChar((*target)[charPos], false);
       if (btn == "SELECT") { charPos++; if(charPos >= target-
>length()) { if (target->length() < 30) *target += " "; else {
charPos = 0; editing = false; } } }
       if (btn == "BACK") { target->trim(); editing = false; charPos
= 0; } delay(150);
    } else {
       if (btn == "UP") { cursorIdx--; if(cursorIdx < 0) cursorIdx =
2; }
       if (btn == "DOWN") { cursorIdx++; if(cursorIdx > 2) cursorIdx
= 0; }
       if (btn == "SELECT") {
          if (cursorIdx == 2) {
             wifi_ssid = tempSSID; wifi_pass = tempPASS;
             writeString(100, wifi_ssid); writeString(200,
wifi_pass);
             tft.fillScreen(C_BG); tft.setCursor(50, 100);
tft.setTextColor(C_METRO_BLUE); tft.print("Connecting..."); return;
          } else { editing = true; charPos = 0; }
       } delay(200);
    }
  }
}

//   MAIN SETUP
void setup() {
  Serial.begin(115200); Wire.begin(21, 22);
  oled.begin(SSD1306_SWITCHCAPVCC, 0x3C); oled.clearDisplay();
oled.display();
  tft.begin(); tft.setRotation(2); tft.fillScreen(C_TEXT_B);
  bootLog("BIOS CHECK... OK");

  //   PERBAIKAN EEPROM DISINI
  EEPROM.begin(512);
  bootLog("Loading Settings...");

  // Gunakan get untuk mengambil data
  EEPROM.get(0, setFull);
  EEPROM.get(10, setLow);
  EEPROM.get(20, setLimitLow);
  EEPROM.get(30, setLimitHigh);
  EEPROM.get(40, setRFPower);
  EEPROM.get(50, relayMode);

  // === VALIDASI ANTI-CRASH ===
  // Jika data ngawur (hasil bacaan sampah), paksa ke default
  bool needSave = false;

  // Cek apakah angka tidak masuk akal (misal negatif, atau > 5000cm,
atau 0 semua)
  if (isnan(setFull) || setFull < 0 || setFull > 3000) { setFull =
20; needSave = true; }
  if (isnan(setLow) || setLow < 0 || setLow > 3000) { setLow = 100;
needSave = true; }
```

```
  // PENTING: Mencegah pembagian dengan nol (setLow tidak boleh sama
dengan setFull)
  if (setLow == setFull) {
    setLow = setFull + 50;
    needSave = true;
  }

  // Validasi Limit Relay
  if (isnan(setLimitLow) || setLimitLow < 0 || setLimitLow > 100) {
setLimitLow = 20; needSave = true; }
  if (isnan(setLimitHigh) || setLimitHigh < 0 || setLimitHigh > 100)
{ setLimitHigh = 90; needSave = true; }

  // Validasi RF & Mode
  if (setRFPower < 0 || setRFPower > 3) { setRFPower = 0; needSave =
true; }
  if (relayMode < 0 || relayMode > 1) { relayMode = 0; needSave =
true; }

  // Jika ada data yang diperbaiki, simpan ulang agar restart
berikutnya aman
  if (needSave) {
    bootLog("Fixing Corrupt Data...");
    EEPROM.put(0, setFull);
    EEPROM.put(10, setLow);
    EEPROM.put(20, setLimitLow);
    EEPROM.put(30, setLimitHigh);
    EEPROM.put(40, setRFPower);
    EEPROM.put(50, relayMode);
    EEPROM.commit();
    delay(100);
  }
  //  ========

  bootLog("Init Sensors..."); dht.begin(); bmp.begin();

  String e_ssid = readString(100); String e_pass = readString(200);
  if (e_ssid.length() > 1 && e_ssid.length() < 32) { wifi_ssid =
e_ssid; wifi_pass = e_pass; bootLog("Loaded WiFi."); }
  else bootLog("Using Default WiFi.");
  bootLog("Connecting to WiFi...");

  WiFi.begin(wifi_ssid.c_str(), wifi_pass.c_str());
  int attempts = 0; bool wifiConnected = false;
  while (attempts < 20) {
    if (WiFi.status() == WL_CONNECTED) { wifiConnected = true; break;
}
    oled.fillRect(2, 27, (bootProgress + attempts), 3,
SSD1306_WHITE); oled.display();
    delay(500); tft.print("."); attempts++;
  }
  if (!wifiConnected) {
      bootLog("CONN FAILED! Setup Mode");
      while(!wifiConnected) {
        handleWifiMenu(); WiFi.begin(wifi_ssid.c_str(),
wifi_pass.c_str()); attempts = 0;
        while(attempts < 20) { if (WiFi.status() == WL_CONNECTED) {
wifiConnected = true; break; } delay(500); attempts++; }
      }
  }
```

```
    bootLog("WiFi Connected!");

    server.on("/", handleRoot);
    server.on("/toggle", handleToggle);
    server.on("/download_macro", handleDownloadMacro);
    server.on("/data", handleData);
    server.on("/override_on", handleWebOverrideOn);
    server.on("/override_off", handleWebOverrideOff);

    //   UPDATE ROUTING
    server.on("/list_files", handleListFiles);
    server.on("/download", handleDownloadLog);
    server.on("/history", handleSDData);

    //   CHAT ROUTES (NEW)
    server.on("/send_chat", handleSendChat);
    server.on("/get_chat", handleGetChat);

    server.begin();

    bootLog("Web Server Started.");

    bootLog("Syncing Clock...");
    configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);

    struct tm timeinfo;
    int timeRetry = 0;
    bool timeSuccess = false;

    while (timeRetry < 20) {
      if (getLocalTime(&timeinfo)) {
        timeSuccess = true;
        break;
      }

      oled.clearDisplay();
      oled.setTextSize(1); oled.setTextColor(SSD1306_WHITE);
      oled.setCursor(10, 10); oled.print("GETTING TIME...");
      oled.setCursor(50, 25); oled.print(20 - timeRetry);
oled.print("s");
      oled.fillRect(0, 30, map(timeRetry, 0, 20, 0, 128), 2,
SSD1306_WHITE);
      oled.display();

      Serial.println("Waiting for NTP...");
      delay(1000);
      timeRetry++;
    }

    if (!timeSuccess) {
      tft.fillScreen(0xF800);
      tft.setTextColor(0xFFFF);
      tft.setTextSize(3); tft.setCursor(20, 100); tft.print("NO
INTERNET");
      tft.setTextSize(2); tft.setCursor(30, 140); tft.print("Time Sync
Fail");

      oled.clearDisplay();
      oled.setTextSize(2); oled.setCursor(10, 10); oled.print("NO
NET!");
      oled.display();
```

```
    tone(BUZZER_PIN, 1000); delay(2000); noTone(BUZZER_PIN);

    // [DIPERBAIKI] Jangan stuck di sini selamanya, lanjut saja pakai
waktu default
    // while(true) { delay(100); }
    bootLog("SKIP TIME SYNC");
  } else {
    bootLog("Time Loaded OK!");
  }

  bootLog("Mounting SD Card..."); initSDCard();

  bootLog("Checking RF Radio...");
  if (!radio.begin()) {
    bootLog("WARN: RF Radio Fail");
  } else {
    bootLog("RF Radio OK.");
    radio.setChannel(115);
    radio.setDataRate(RF24_250KBPS);

    // APPLY POWER FROM EEPROM
    if(setRFPower == 0) radio.setPALevel(RF24_PA_MIN);
    else if(setRFPower == 1) radio.setPALevel(RF24_PA_LOW);
    else if(setRFPower == 2) radio.setPALevel(RF24_PA_HIGH);
    else radio.setPALevel(RF24_PA_MAX);

    radio.setRetries(15, 15);
    radio.enableDynamicPayloads();
    radio.enableAckPayload();
    radio.openWritingPipe(rfAddress);
    radio.stopListening();
  }
  pinMode(TRIG_PIN, OUTPUT); pinMode(ECHO_PIN, INPUT);
  pinMode(BATT_PIN, INPUT); pinMode(BUZZER_PIN, OUTPUT);
  pinMode(MQ135_PIN, INPUT); pinMode(LASER_PIN, OUTPUT);
  digitalWrite(LASER_PIN, HIGH);

  bootLog("SYSTEM READY.");
  logPumpEvent("SYSTEM ON", C_OK);
  delay(500);

  // NADA STARTUP
  playTone(1);

  tft.fillScreen(C_BG); needsFullRedraw = true;
  lastSensorRead = millis();
}

//   MAIN LOOP
void loop() {
  server.handleClient();

  if (millis() - lastSensorRead >= sensorInterval) {
    lastSensorRead = millis();
    readSensors();
  }

  handleLaserMorse();
```

```
  if (isGasAlertActive) { drawGasAlert(); wasGasAlertActive = true;
return; }
  if (wasGasAlertActive && !isGasAlertActive) { wasGasAlertActive =
false; needsFullRedraw = true; tft.fillScreen(C_BG); }

  String btn = readButtons();
  static String lastBtn = "IDLE";

  //   LOGIKA EMERGENCY OVERRIDE REVISI
  bool triggerOverride = false;

  // 1. Cek Tombol Fisik (SELECT Tahan 4 Detik)
  if (btn == "SELECT" && !inSubMenu) {
      unsigned long pressStart = millis();
      while(readButtons() == "SELECT") {
          server.handleClient(); // PENTING: Anti-Freeze saat menahan
tombol
          if (millis() - pressStart > 4000) { triggerOverride = true;
playTone(2); break; }
          delay(50);
      }
  }

  // 2. Cek Trigger Web
  if (webOverrideReq) {
      triggerOverride = true;
      playTone(2);
      webOverrideReq = false; // Reset flag
  }

  //   MASUK KE LOOP OVERRIDE
  if (triggerOverride) {
      isOverride = true;
      pumpState = !pumpState; // Default action saat masuk override
(flip state)

      // UI OVERRIDE (Gambar kotak merah)
      tft.fillRect(40, 100, 160, 60, C_TEXT_W);
      tft.drawRect(40, 100, 160, 60, C_METRO_RED);
      tft.setTextSize(2); tft.setTextColor(C_METRO_ORANGE, C_TEXT_W);
      tft.setCursor(75, 115); tft.print("OVERRIDE");
      tft.setTextSize(1); tft.setTextColor(C_TEXT_B, C_TEXT_W);
      tft.setCursor(65, 140); tft.print("PUMP FORCED: ");
tft.print(pumpState ? "ON" : "OFF");

      // Tunggu tombol fisik dilepas dulu
      while(readButtons() == "SELECT") { server.handleClient();
delay(10); }

      bool inOverrideLoop = true;
      while(inOverrideLoop) {
          // 1. PENTING: Handle Web Client agar tidak Freeze
          server.handleClient();

          // 2. Baca Sensor (tetap update data di web)
          if (millis() - lastSensorRead >= sensorInterval) {
             lastSensorRead = millis();
             readSensors();
          }

          // 3. Cek Tombol Fisik "BACK" untuk Keluar
```

```
            if (readButtons() == "BACK") {
               playTone(3);
               isOverride = false;
               inOverrideLoop = false;
            }

            // 4. Cek Web Exit Request (Double Click)
            if (webExitOverrideReq) {
               playTone(3);
               isOverride = false;
               inOverrideLoop = false;
               webExitOverrideReq = false;
            }

            // 5. Update UI di Alat (Kedip-kedip)
            if (millis() % 1000 < 500) tft.drawRect(42, 102, 156, 56,
C_METRO_RED);
            else tft.drawRect(42, 102, 156, 56, C_TEXT_W);

            tft.setTextSize(1); tft.setTextColor(C_TEXT_B, C_TEXT_W);
            tft.setCursor(65, 140); tft.print("PUMP FORCED: ");
tft.print(pumpState ? "ON " : "OFF");

            delay(50);
         }

         tft.fillScreen(C_BG); needsFullRedraw = true;
         return;
      }
      //   END LOGIKA EMERGENCY OVERRIDE

      //   2. SAFE SHUTDOWN (BACK 2s)
      if (btn == "BACK" && !inSubMenu) {
         unsigned long pressStart = millis();
         bool longPressed = false;
         while(readButtons() == "BACK") {
            server.handleClient(); // Anti Freeze
            if (millis() - pressStart > 2000) {
               playTone(3);
               while(readButtons() == "BACK") { delay(10); }
               showShutdownMenu();
               longPressed = true;
               break;
            }
            delay(50);
         }
         if (longPressed) { lastBtn = "IDLE"; btn = "IDLE"; return; }
      }

      if (btn != "IDLE") {
        unsigned long now = millis(); bool act = false;
        if (btn != lastBtn) { act = true; lastRepeatTime = now;
repeatDelay = 400; digitalWrite(BUZZER_PIN, HIGH); delay(15);
digitalWrite(BUZZER_PIN, LOW); }
        else if (now - lastRepeatTime > repeatDelay) { if (btn == "UP" ||
btn == "DOWN") { act = true; lastRepeatTime = now; repeatDelay = 100;
} }

        if (act) {
            if (!inSubMenu) {
```

```
            if (btn == "UP") { menuIndex--; if(menuIndex < 0) menuIndex
= 5; needsMenuUpdate = true; }
            else if (btn == "DOWN") { menuIndex++; if(menuIndex > 5)
menuIndex = 0; needsMenuUpdate = true; }
            else if (btn == "SELECT") { inSubMenu = true;
needsFullRedraw = true; }
          } else handleSubMenu(btn);
      }
    }
    lastBtn = btn;
    if (!inSubMenu) displayMenu(); else handleSubMenu("IDLE");
    if(needsStatusUpdate) drawStatusBar();
  }
```

LAMPIRAN B

Code Program Esp 8266 (Rangkaian Daya)

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

//   KONFIGURASI PIN
#define CE_PIN        D3
#define CSN_PIN       D8
#define RELAY_PIN     D4
#define FEEDBACK_PIN  D0

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define OLED_RESET    -1
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

RF24 radio(CE_PIN, CSN_PIN);
const byte rfAddress[6] = "00001";

struct DataPacket {
  float dist;
  int pct;
  float temp;
  float hum;
  int airQuality;
  bool pump;
};
DataPacket incomingData;

struct FeedbackPacket {
  bool isPumpOn;
  int sensorStatus;
};
FeedbackPacket feedbackData;

unsigned long lastRecvTime = 0;
bool isConnected = false;
bool currentPumpState = false;
int currentFlowState = 0;
int displayMode = 0;
bool blinkState = false;
unsigned long lastSwitchTime = 0;
unsigned long lastBlinkTime = 0;
int logY = 0;

//   FUNGSI HELPER BOOTING

void updateLog(const char* msg, int progress) {
  display.fillRect(0, 45, 64, 35, SSD1306_BLACK);
  display.setTextSize(1);
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(0, 1);
  display.print("by sADI");
  display.setCursor(0, 50);
  display.print("STATUS:");
```

```
  display.setCursor(0, 65);
  display.print(">> ");
  display.println(msg);

  int barW = 50;
  int barH = 8;
  int barX = (64 - barW) / 2;
  int barY = 110;
  display.fillRect(barX + 2, barY + 2, barW - 4, barH - 4,
SSD1306_BLACK);
  display.drawRect(barX, barY, barW, barH, SSD1306_WHITE);
  display.fillRect(barX + 2, barY + 2, map(progress, 0, 100, 0, barW -
4), barH - 4, SSD1306_WHITE);
  display.display();
  delay(400);
}

//   FUNGSI LAYOUT UTAMA

void drawSketchLayout() {
  display.clearDisplay();
  display.setRotation(3);
  display.setTextColor(SSD1306_WHITE);

  //   LOGIKA JIKA SINYAL HILANG (DENGAN IKON SEGITIGA)
  if (!isConnected) {
    int cx = 32; // Center X
    int cy = 40; // Center Y
    int s = 15;  // Size

    // Gambar Segitiga
    display.drawTriangle(cx, cy-s, cx-s, cy+s, cx+s, cy+s,
SSD1306_WHITE);
    // Gambar Tanda Seru
    display.setTextSize(2);
    display.setCursor(cx-3, cy-2);
    display.print("!");

    display.setTextSize(1);
    display.setCursor(5, 70);
    display.print("NO SIGNAL");
    display.display();
    return; // Keluar dari fungsi agar layout bawah tidak digambar
  }

  //   1. BAGIAN KIRI: TANGKI
  int barX = 2;
  int barY = 10;
  int barW = 14;
  int barH = 80;

  display.drawRect(barX, barY, barW, barH, SSD1306_WHITE);
  for(int i=0; i<=4; i++) {
    int tickY = barY + (i * (barH/4));
    display.drawLine(barX + barW + 1, tickY, barX + barW + 3, tickY,
SSD1306_WHITE);
  }

  int fillH = map(incomingData.pct, 0, 100, 0, barH - 4);
  fillH = constrain(fillH, 0, barH - 4);
```

```
  display.fillRect(barX + 2, (barY + barH - 2) - fillH, barW - 4, fillH,
SSD1306_WHITE);

  //   2. BAGIAN KANAN: DATA
  int dataX = 24;
  display.setTextSize(1);
  display.setCursor(dataX, 10);
  display.print((int)incomingData.dist); display.print("cm");

  display.drawFastHLine(dataX, 22, 35, SSD1306_WHITE);
  display.setCursor(dataX - 2, 28);
  if(incomingData.pct >= 100) display.setTextSize(2); else
display.setTextSize(3);
  display.print(incomingData.pct);
  display.setTextSize(1);
  display.print("%");

  int paramY = 62;
  display.drawFastHLine(dataX, 58, 35, SSD1306_WHITE);
  display.setCursor(dataX, paramY);
  if(displayMode == 0) {
    display.print("TEMP:");
    display.setCursor(dataX, paramY + 10);
    display.print(incomingData.temp, 1); display.print("C");
  } else if(displayMode == 1) {
    display.print("HUMI:");
    display.setCursor(dataX, paramY + 10);
    display.print((int)incomingData.hum); display.print("%");
  } else {
    display.print("AIR Q:");
    display.setCursor(dataX, paramY + 10);
    display.print(incomingData.airQuality);
  }

  //   3. BAGIAN BAWAH: STATUS
  display.drawFastHLine(0, 95, 64, SSD1306_WHITE);
  display.setCursor(2, 100);
  display.print("FLOW: ");
  display.print(currentFlowState == HIGH ? "OK" : "STOP");

  int pumpY = 114;
  if(currentPumpState) {
    display.fillRect(0, pumpY - 2, 64, 15, SSD1306_WHITE);
    display.setTextColor(SSD1306_BLACK);
    display.setCursor(8, pumpY);
    display.print("PUMP: ON");
  } else {
    display.setTextColor(SSD1306_WHITE);
    display.setCursor(8, pumpY);
    display.print("PUMP: OFF");
  }

  if(blinkState) display.fillCircle(60, 5, 2, SSD1306_WHITE);
  display.display();
}

//   SETUP & LOOP

void setup() {
  Serial.begin(115200);
  pinMode(RELAY_PIN, OUTPUT); digitalWrite(RELAY_PIN, LOW);
```

```
  pinMode(FEEDBACK_PIN, INPUT);

  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) { for(;;); }
  display.setRotation(3);
  display.clearDisplay();

  updateLog("BOOT", 10);
  if (!radio.begin()) {
    updateLog("RF FAIL!", 40);
    while(1);
  }

  updateLog("RF OK", 60);
  radio.setChannel(115);
  radio.setDataRate(RF24_250KBPS);
  radio.setPALevel(RF24_PA_MAX);
  radio.enableDynamicPayloads();
  radio.enableAckPayload();
  radio.openReadingPipe(1, rfAddress);
  radio.startListening();

  updateLog("READY!", 100);

  feedbackData.isPumpOn = false;
  feedbackData.sensorStatus = 0;
  radio.writeAckPayload(1, &feedbackData, sizeof(FeedbackPacket));
  delay(1000);
}

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

  if (now - lastBlinkTime > 500) {
    blinkState = !blinkState;
    lastBlinkTime = now;
    drawSketchLayout();
  }

  if (now - lastSwitchTime > 3000) {
    displayMode = (displayMode + 1) % 3;
    lastSwitchTime = now;
  }

  if (radio.available()) {
    radio.read(&incomingData, sizeof(DataPacket));
    lastRecvTime = now;
    isConnected = true;

    currentPumpState = incomingData.pump;
    digitalWrite(RELAY_PIN, currentPumpState ? HIGH : LOW);
    currentFlowState = digitalRead(FEEDBACK_PIN);

    feedbackData.isPumpOn = currentPumpState;
    feedbackData.sensorStatus = currentFlowState;
    radio.writeAckPayload(1, &feedbackData, sizeof(FeedbackPacket));
  }

  if (now - lastRecvTime > 3000) {
    if (isConnected) {
      isConnected = false;
      digitalWrite(RELAY_PIN, LOW);
```

```
        }
    }
}
```

LAMPIRAN C

kode vba (untuk mengubah file ekstensi *.csv*

agar terbaca di tabel excel)

MODUL 1

```
Sub ImportSmartMonitorData()
    Dim ws As Worksheet
    Dim filePath As Variant
    Dim lastRow As Long
    Dim tbl As ListObject
    Dim chartObj As ChartObject
    Dim i As Long, colIndex As Long
    Dim rngData As Range
    Dim shp As Shape
    Dim srs As Series

    ' 1. Setting Awal & Error Handling
    On Error GoTo ErrorHandler
    Application.ScreenUpdating = False
    Set ws = ActiveSheet

    ' Kunci Posisi Tombol
    For Each shp In ws.Shapes
        shp.Placement = xlFreeFloating
    Next shp

    ' Hapus Data Lama (Baris 3 ke bawah)
    ws.Rows("3:" & ws.Rows.Count).Clear

    ' Hapus Grafik Lama
    For Each chartObj In ws.ChartObjects
        chartObj.Delete
    Next chartObj

    ' 2. Pilih File CSV
    filePath = Application.GetOpenFilename("CSV Files (*.csv),
*.csv", , "Pilih File Log ESP32")
    If filePath = False Then
        Application.ScreenUpdating = True
        Exit Sub
    End If

    ' 3. Import Data
    With ws.QueryTables.Add(Connection:="TEXT;" & filePath,
Destination:=ws.Range("A4"))
        .TextFileParseType = xlDelimited
        .TextFileCommaDelimiter = True
        .Refresh BackgroundQuery:=False
    End With
    If ws.QueryTables.Count > 0 Then ws.QueryTables(1).Delete

    ' 4. Header & Formatting Data
    ws.Range("A3:O3").value = Array("Tanggal", "Raw Sensor", "Tinggi
Air (%)", "Level (%)", _
                                    "Suhu (C)", "Kelembaban (%)",
"Gas (Raw)", "Pompa", _
```

```
                                        "Full Top", "Low Bottom", "Mode",
"Set High", _
                                        "Set Low", "Feedback", "RF")

    ' Styling Header Tabel
    With ws.Range("A3:O3")
        .Font.Bold = True
        .Interior.color = RGB(40, 40, 40)
        .Font.color = RGB(255, 255, 255)
        .HorizontalAlignment = xlCenter
    End With

    lastRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row
    If lastRow < 4 Then lastRow = 4

    ' 5. Insert Kolom Jam
    ws.Columns("B:B").Insert Shift:=xlToRight
    ws.Range("B3").value = "Jam"
    With ws.Range("B3")
        .Font.Bold = True
        .Interior.color = RGB(40, 40, 40)
        .Font.color = RGB(255, 255, 255)
    End With

    ' 6. Loop Konversi Data
    For i = 4 To lastRow
        ' Tanggal
        If IsNumeric(ws.Cells(i, 1).value) And ws.Cells(i, 1).value
<> "" Then
            ws.Cells(i, 1).value = ws.Cells(i, 1).value / 86400000
        End If
        ' Jam (Format Text '14:00)
        If ws.Cells(i, 1).value <> "" Then
            ws.Cells(i, 2).value = "'" & Format(ws.Cells(i, 1).value,
"hh:mm")
        End If
    Next i

    ' 7. Text To Columns (PENTING: Ubah Text jadi Angka)
    For colIndex = 3 To 8
        Set rngData = ws.Range(ws.Cells(4, colIndex),
ws.Cells(lastRow, colIndex))
        rngData.Replace What:=".", Replacement:=",", LookAt:=xlPart
        rngData.TextToColumns Destination:=rngData.Cells(1, 1),
DataType:=xlDelimited, _
                              FieldInfo:=Array(1, 1),
TrailingMinusNumbers:=True
        rngData.NumberFormat = "0.00"
    Next colIndex

    ' Format Kolom Waktu
    ws.Range("A4:A" & lastRow).NumberFormat = "[$-en-ID,1]d mmm
yyyy;@"
    ws.Range("B4:B" & lastRow).NumberFormat = "@"

    ' 8. Buat Tabel & Kop
    Set tbl = ws.ListObjects.Add(xlSrcRange, ws.Range("A3:P" &
lastRow), , xlYes)
    tbl.TableStyle = "TableStyleMedium1"

    ' ============================================================
```

```
        ' HITUNG DATA RINGKASAN UNTUK KOP
        ' ============================================================
        Dim avgSuhu As String
        Dim tglAwal As String, tglAkhir As String
        Dim jamAwal As String, jamAkhir As String

        On Error Resume Next
        ' Hitung Rata-rata Suhu (Kolom F)
        avgSuhu =
    Format(Application.WorksheetFunction.Average(ws.Range("F4:F" &
    lastRow)), "0.0")

        ' Ambil Tanggal & Jam (Baris 4 = Awal, lastRow = Akhir)
        tglAwal = Format(ws.Cells(4, 1).value, "dd/mm/yy")
        tglAkhir = Format(ws.Cells(lastRow, 1).value, "dd/mm/yy")
        ' Bersihkan tanda petik (') dari jam agar rapi
        jamAwal = Replace(ws.Cells(4, 2).value, "'", "")
        jamAkhir = Replace(ws.Cells(lastRow, 2).value, "'", "")
        On Error GoTo ErrorHandler
        ' ============================================================

        ' Buat Kop Laporan
        Dim headerInfo As String
        headerInfo = "Lokasi: Area Tangki  |  Rata-rata Suhu: " & avgSuhu
    & "°C  |  " & _
                     "Periode: " & tglAwal & " - " & tglAkhir & " (" &
    jamAwal & " s/d " & jamAkhir & ")"

        With ws.Range("A1:P1")
            .Merge
            .value = "LAPORAN MONITORING SISTEM KONTROL AIR"
            .Font.Size = 16
            .Font.Bold = True
            .HorizontalAlignment = xlCenter
            .VerticalAlignment = xlCenter
            .Interior.color = RGB(230, 230, 230)
            .RowHeight = 25
        End With

        With ws.Range("A2:P2")
            .Merge
            .value = headerInfo
            .HorizontalAlignment = xlCenter
            .VerticalAlignment = xlCenter
            .Interior.color = RGB(242, 242, 242)
            .Font.Size = 11
            .RowHeight = 20
            .Borders(xlEdgeBottom).LineStyle = xlContinuous
        End With

        ' ============================================================
        ' 9. PEMBUATAN GRAFIK
        ' ============================================================
        Dim chartLeft As Double, chartTop As Double, chartW As Double,
    chartH As Double
        chartLeft = ws.Range("S3").Left
        chartTop = ws.Range("S3").Top
        chartW = 600: chartH = 250

        ' Pastikan ada data untuk digrafikkan
        If lastRow < 5 Then GoTo SkipCharts
```

```
    ' --- GRAFIK 1: AIR (Kolom D) ---
    Set chartObj = ws.ChartObjects.Add(Left:=chartLeft,
Top:=chartTop, Width:=chartW, Height:=chartH)
    With chartObj.Chart
        .ChartType = xlLine
        Do While .SeriesCollection.Count > 0:
.SeriesCollection(1).Delete: Loop

        With .SeriesCollection.NewSeries
            .Name = "Ketinggian Air"
            .Values = ws.Range("D4:D" & lastRow)
            .XValues = ws.Range("B4:B" & lastRow)
        End With
    End With
    Call ApplyModernStyle(chartObj, "MONITORING KETINGGIAN AIR",
RGB(0, 255, 255)) ' CYAN

    ' --- GRAFIK 2: LEVEL & KELEMBABAN (Kolom E & G) ---
    chartTop = chartTop + chartH + 15
    Set chartObj = ws.ChartObjects.Add(Left:=chartLeft,
Top:=chartTop, Width:=chartW, Height:=chartH)
    With chartObj.Chart
        .ChartType = xlLine
        Do While .SeriesCollection.Count > 0:
.SeriesCollection(1).Delete: Loop

        ' Series 1
        With .SeriesCollection.NewSeries
            .Name = "Level Tangki (%)"
            .Values = ws.Range("E4:E" & lastRow)
            .XValues = ws.Range("B4:B" & lastRow)
        End With
        ' Series 2
        With .SeriesCollection.NewSeries
            .Name = "Kelembaban (%)"
            .Values = ws.Range("G4:G" & lastRow)
        End With
    End With
    Call ApplyModernStyle(chartObj, "LEVEL TANGKI & KELEMBABAN",
RGB(255, 165, 0))

    ' Override Warna Garis Kedua (Hijau Lime)
    On Error Resume Next
    With chartObj.Chart.SeriesCollection(2).Format.Line
        .ForeColor.RGB = RGB(50, 205, 50)
        .Weight = 2
    End With
    On Error GoTo ErrorHandler

    ' --- GRAFIK 3: GAS (Kolom H) ---
    chartTop = chartTop + chartH + 15
    Set chartObj = ws.ChartObjects.Add(Left:=chartLeft,
Top:=chartTop, Width:=chartW, Height:=chartH)
    With chartObj.Chart
        .ChartType = xlLine
        Do While .SeriesCollection.Count > 0:
.SeriesCollection(1).Delete: Loop

        With .SeriesCollection.NewSeries
            .Name = "Gas Sensor"
```

```
            .Values = ws.Range("H4:H" & lastRow)
            .XValues = ws.Range("B4:B" & lastRow)
        End With
    End With
    Call ApplyModernStyle(chartObj, "MONITORING GAS SENSOR", RGB(255,
0, 100)) ' PINK

SkipCharts:
    ' Finishing
    ws.UsedRange.HorizontalAlignment = xlCenter
    ws.Columns("A:Q").AutoFit
    ws.Range("A1").Select
    Application.ScreenUpdating = True

    Exit Sub

ErrorHandler:
    Application.ScreenUpdating = True
    MsgBox "Terjadi kesalahan: " & Err.Description, vbCritical
End Sub

' ============================================================
' SUB KHUSUS DESAIN (TIDAK PERLU DIUBAH)
' ============================================================
Sub ApplyModernStyle(chtObj As ChartObject, titleText As String,
mainColor As Long)
    On Error Resume Next

    With chtObj.Chart
        ' 1. Judul & Background
        .HasTitle = True
        .ChartTitle.Text = titleText
        .ChartTitle.Font.color = RGB(255, 255, 255)
        .ChartTitle.Font.Bold = True

        .ChartArea.Format.Fill.ForeColor.RGB = RGB(35, 35, 45)
        .PlotArea.Format.Fill.ForeColor.RGB = RGB(35, 35, 45)
        .ChartArea.Border.Line.Visible = msoFalse

        ' 2. Sumbu X & Y
        With .Axes(xlValue)
            .TickLabels.Font.color = RGB(200, 200, 200)
            .MajorGridlines.Format.Line.ForeColor.RGB = RGB(60, 60,
60)
            .MajorGridlines.Format.Line.DashStyle = msoLineSysDot
        End With
        With .Axes(xlCategory)
            .TickLabels.Font.color = RGB(200, 200, 200)
            .TickLabels.NumberFormat = "hh:mm"
        End With

        ' 3. Warna Garis Series
        If .SeriesCollection.Count > 0 Then
            With .SeriesCollection(1).Format.Line
                .ForeColor.RGB = mainColor
                .Weight = 2
            End With
            .SeriesCollection(1).MarkerStyle = xlMarkerStyleNone
        End If

        ' 4. Legend
```

```
            If .HasLegend Then
                .Legend.Position = xlLegendPositionTop
                .Legend.Font.color = RGB(255, 255, 255)
            End If
        End With
        chtObj.RoundedCorners = True
    End Sub
```

MODUL 2

```
Sub ResetWorksheet()
    Dim ws As Worksheet
    Dim chrt As ChartObject
    Dim tbl As ListObject
    Dim shp As Shape
    Set ws = ActiveSheet

    ' 1. Konfirmasi sebelum reset (opsional tapi disarankan)
    If MsgBox("Apakah Anda yakin ingin menghapus SEMUA data dan
grafik?", _
              vbQuestion + vbYesNo, "Konfirmasi Reset") = vbNo Then
Exit Sub

    ' 2. Matikan proteksi dan updating untuk kecepatan
    Application.ScreenUpdating = False

    ' 3. Hapus semua Tabel (ListObjects) agar tidak merusak
formatting
    For Each tbl In ws.ListObjects
        tbl.Unlist ' Ubah tabel jadi range biasa dulu
    Next tbl

    ' 4. Hapus semua konten, warna, border, dan format sel
    ws.Cells.Clear
    ws.Cells.ClearFormats

    ' 5. Hapus semua Grafik/Chart
    For Each chrt In ws.ChartObjects
        chrt.Delete
    Next chrt

    ' 6. Mengembalikan tampilan standar Excel
    With ActiveWindow
        .DisplayGridlines = True ' Munculkan garis kotak-kotak lagi
        .Zoom = 100              ' Kembalikan zoom ke 100%
    End With

    ' 7. Kembalikan ukuran kolom dan baris ke standar
    ws.Rows.RowHeight = 15
    ws.Columns.ColumnWidth = 8.43

    ' 8. Pindahkan kursor ke A1
    ws.Range("A1").Select

    Application.ScreenUpdating = True
End Sub
```