PERANCANGAN DAN IMPLEMENTASI SISTEM MONITORING *AIR QUALITY INDEX* (AQI) BERBASIS SENSOR BME688 DENGAN VISUALISASI DATA P5 DISLPAY DI AREA PELAYANAN PUBLIK

SKRIPSI

Diajukan Untuk Memenuhi Syarat Mendapatkan Gelar Sarjana Strata Satu (S1) Pada Program Studi Teknik Elektro Fakultas Teknik Universitas Islam Nusantara

Oleh :

Hidayat

41037002211008

PROGRAM STUDI TEKNIK ELEKTRO

FAKULTAS TEKNIK

UNIVERSITAS ISLAM NUSANTARA

2025

LEMBAR KEASLIAN SKRIPSI

Yang bertanda tangan di bawah ini:

Nama: Hidayat

NIM: 41037002211008

Program Studi: Teknik Elektro

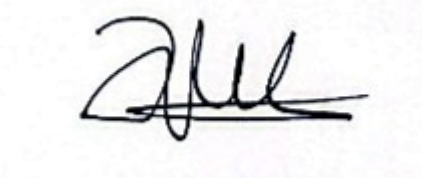
Menyatakan bahwa Skripsi yang berjudul:

PERANCANGAN DAN IMPLEMENTASI SISTEM MONITORING *AIR QUALITY INDEX* (AQI) BERBASIS SENSOR BME688 DENGAN VISUALISASI DATA P5 DISLPAY DI AREA PELAYANAN PUBLIK

Dibuat dengan sebenar-benarnya dari penelitian, pemikiran, dan pemaparan hasil saya sendiri, untuk melengkapi sebagai pernyataan menjadi Sarjana (S1) pada jurusan Teknik Elektro Fakultas Teknik Universitas Islam Nusantara Bandung, sejauh yang saya ketahui bukan merupakan tiruan atau duplikasi dari buku Skripsi yang sudah dipublikasikan dan atau pernah dipakai untuk mendapatkan jenjang Sarjana (S1) di lingkungan Teknik Elektro Fakultas Teknik Universitas Islam Nusantara Bandung maupun perguruan-perguruan tinggi atau instansi manapun kecuali bagian yang sumber informasi dicantumkan sebagaimana mestinya.

Bandung, 12 Juni 2025

Yang membuat pernyataan,



HIDAYAT
NIM 41037002211008

LEMBAR PENGESAHAN

PERANCANGAN DAN IMPLEMENTASI SISTEM MONITORING *AIR QUALITY INDEX* (AQI) BERBASIS SENSOR BME688 DENGAN VISUALISASI DATA P5 DISLPAY DI AREA PELAYANAN PUBLIK

Disusun dan diajukan oleh :

HIDAYAT

41037002211008

Disetujui dan disahkan pada

pada tanggal :

Bandung, 12 Juni 2025

Pembimbing 1	Pembimbing 2
Dr. Iksal Rachman, M.T.	Agung M Toha S.ST, M.T

Mengetahui :

Dekan Fakultas Teknik	Ketua Prodi Teknik Elektro
Dr. Ricky Yoseptry, S.T., M.M.Pd.	Muhammad Zimamul Adli, M.Si.

LEMBAR PENGESAHAN

REVISIAN SKRIPSI

PERANCANGAN DAN IMPLEMENTASI SISTEM MONITORING *AIR QUALITY INDEX* (AQI) BERBASIS SENSOR BME688 DENGAN VISUALISASI DATA P5 DISLPAY DI AREA PELAYANAN PUBLIK

Telah Direvisi oleh :

HIDAYAT

41037002211008

Bandung, 26 juni 2025

Mengesahkan,

Penguji 1

Osphanie Mentari, S.T., M.T.

Penguji 2

Ir. Joko Haryatno, M.T.

Ketua Sidang

Muhammad Zipsamul Adli, M.Si.

BIODATA PENULIS

`

Nama	: Hidayat
Tempat, Tanggal Lahir	: Bandung, 10 Maret 2000
Telepon	: 0821-1611-5711
Email	: jajangh475@gmail.com
Riwayat Pendidikan	: SDN Patrol 3
	SMPN 1 Solokanjeruk
	SMK Wirakarya 1 Ciparay

KATA PENGANTAR

Puji dan syukur penulis panjatkan ke hadirat Allah SWT, Tuhan Yang Maha Esa, atas limpahan rahmat, kasih sayang, serta hidayah-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul *"Perancangan dan Implementasi Sistem QAIR untuk Pemantauan dan Analisis Kualitas Udara di Tempat Pelayanan Masyarakat secara Real-time"*. Skripsi ini disusun sebagai salah satu syarat untuk menyelesaikan studi jenjang Strata Satu (S1) di Universitas Islam Nusantara, Fakultas Teknik, Program Studi Teknik Elektro.

Dalam proses penyusunan skripsi ini, penulis menyadari bahwa capaian ini tidak lepas dari dukungan, doa, serta bantuan dari berbagai pihak yang telah memberikan semangat dan dorongan moral. Dengan penuh rasa hormat dan tulus, penulis ingin menyampaikan ucapan terima kasih yang sebesar-besarnya kepada:

1. Allah SWT, yang senantiasa memberikan kekuatan, kesehatan, dan petunjuk dalam setiap langkah penulis. Segala sesuatu adalah berkat kehendak-Nya, dan semoga karya ini menjadi amal yang bermanfaat.
2. Orang tua dan seluruh anggota keluarga besar bapak YOYO yang menjadi sumber kekuatan, cinta, dan inspirasi. Terima kasih atas segala doa, dukungan moril, semangat, serta bantuan materi yang tidak ternilai.
3. Bapak Ganis Sanhaji, S.Si., M.Sc., selaku dosen wali selama massa perkuliahan yang selalu senantiasa memberikan arahan dan bimbingan pada saat proses pembuatan skripsi ini.
4. Dr. Iksal Rachman, M.T. beserta bapak Agung M Toha S.ST, M.T, selaku dosen pembimbing skripsi, yang telah memberikan arahan, bimbingan, dan kesabaran dalam membimbing penulis hingga skripsi ini terselesaikan.
5. Bapak Muhammad Zimamul Adli, M.Si. selaku Ketua Program Studi Teknik Elektro atas segala dukungan dan apresiasi akan proses pembuatan skripsi ini.

6. Dr. Ricky Yoseptry, S.T., M.M.Pd., selaku Dekan Fakultas Teknik, yang senantiasa memberikan motivasi dan dukungan selama masa perkuliahan.
7. Seluruh dosen di Fakultas Teknik yang telah membagikan ilmu dan pengalaman berharga, yang menjadi bekal penting bagi penulis dalam menyusun karya ilmiah ini.
8. Seluruh staf Tata Usaha Fakultas Teknik, yang telah membantu dalam urusan administrasi dengan penuh kesabaran dan pelayanan yang baik.
9. Prof. Dr. Endang Komara, M.Si., selaku Rektor Universitas Islam Nusantara, yang telah memberikan kesempatan dan fasilitas untuk belajar serta berkembang di lingkungan kampus ini.
10. Rekan-rekan seperjuangan dalam menyusun skripsi: Julian, Hamdi Sholahudin, Herlan Syah, Anisa Febrianti, Vito Dwi Nur Hidayat, dan Decky Putra Kurnia, yang telah menjadi teman diskusi, motivator, dan sahabat sejati selama proses penyusunan skripsi ini.
11. Teman-teman Teknik Elektro angkatan 2021, atas kebersamaan, dukungan, serta semangat yang telah mewarnai perjalanan studi penulis.
12. Seluruh pihak lain yang tidak dapat disebutkan satu per satu, yang telah memberikan bantuan secara langsung maupun tidak langsung.

Penulis menyadari bahwa skripsi ini masih jauh dari kesempurnaan. Oleh karena itu, segala bentuk saran dan kritik yang membangun sangat diharapkan demi perbaikan di masa mendatang. Semoga skripsi ini dapat memberikan manfaat dan menjadi referensi yang berguna bagi pembaca maupun pengembang teknologi serupa di masa depan.

Bandung, 27 Mei 2025

Penulis

ABSTRAK

Pencemaran udara menjadi salah satu faktor utama penyebab gangguan kesehatan pernapasan dan kematian dini, terutama di kawasan padat aktivitas seperti area pelayanan publik. Penelitian ini bertujuan untuk merancang dan mengimplementasikan sistem pemantauan kualitas udara berbasis sensor BME688 yang terintegrasi dengan mikrokontroler ESP32, panel LED P5 untuk tampilan data real-time, serta modul RTC dan kartu SD untuk pencatatan waktu dan penyimpanan data historis.

Sistem ini mampu mengukur parameter suhu, kelembapan, tekanan udara, indeks kualitas udara (IAQ), konsentrasi CO_2, dan senyawa organik volatil (VOC). Metode yang digunakan adalah pendekatan Research and Development (R&D) dengan teknik kuantitatif. Hasil pengujian yang dilakukan selama tiga hari menunjukkan bahwa rata-rata suhu mencapai 27,13°C, kelembapan 77,9%, tekanan udara 936,32 hPa, IAQ sebesar 51,3 (kategori baik), CO_2 sebesar 610,3 ppm, dan VOC sebesar 0,52 ppm.

Sensor BME688 memiliki tingkat akurasi ±0,5°C untuk suhu, ±3% RH untuk kelembapan, dan ±0,6 hPa untuk tekanan udara, dengan deviasi IAQ antar sensor sebesar ±15%. Hasil menunjukkan bahwa sistem ini efektif dalam memberikan informasi kualitas udara secara real-time dan dapat dijadikan media edukatif untuk meningkatkan kesadaran masyarakat terhadap lingkungan dan kesehatan.

Kata kunci: Air Quality Index, Sensor BME688, ESP32, Panel P5, Kualitas Udara, IAQ, CO_2, VOC, Monitoring Real-Time, Akurasi Sensor.

Abstract

Air pollution ranks among the primary contributors to respiratory illnesses and untimely fatalities, especially in densely populated areas such as public service venues. This research is focused on creating and deploying an air quality monitoring system that utilizes the BME688 sensor in conjunction with the ESP32 microcontroller, a P5 LED display for real-time visualization of data, and RTC and SD card components for time tracking and archival data storage. The system is designed to assess variables including temperature, humidity, atmospheric pressure, Indoor Air Quality (IAQ), CO_2 levels, and Volatile Organic Compounds (VOC). The study employs a Research and Development (R&D) methodology, utilizing a quantitative framework. Over a testing period of three days, the system recorded an average temperature of 27.13°C, a humidity level of 77.9%, atmospheric pressure measured at 936.32 hPa, IAQ scored 51.3 (deemed good), CO_2 concentration was at 610.3 ppm, and VOC measured 0.52 ppm. The BME688 sensor delivers accuracy levels of ±0.5°C for temperature, ±3% RH for humidity, ±0.6 hPa for pressure, and a discrepancy of ±15% on IAQ readings between sensors. The results indicate that the system successfully delivers real-time air quality data and has the potential to act as an educational resource to enhance public understanding of environmental and health-related matters.

Keywords: Air Quality Index, BME688 Sensor, ESP32, P5 Panel, Air Quality, IAQ, CO_2, VOC, Real-Time Monitoring, Sensor Accuracy.

DAFTAR ISI

LEMBAR KEASLIAN SKRIPSI i
LEMBAR PENGESAHAN ii
LEMBAR PENGESAHAN iii
BIODATA PENULIS iv
KATA PENGANTAR v
ABSTRAK vii
DAFTAR ISI ix
DAFTAR GAMBAR xi
DAFTAR TABEL xii
1.1. Latar Belakang 1
1.2. Rumusan Masalah 5
1.3. Batasan Masalah dan Asumsi 6
1.4. Tujuan Penelitian 6
1.5. Manfaat Penelitian 7
BAB 2 LANDASAN TEORI 8
2.1. Indeks Kualitas Udara (AQI) 8
2.2. Sensor BME688 12
2.3. Mikrokontroler ESP32 16
2.4. Panel P5 (LED Matrix Display) 17
2.5. Modul Kartu SD 18
2.6. Modul Step-Down (DC-DC Buck Converter) 19
2.7. RTC DS1307 (Real Time Clock) 20
2.8. Bahasa Pemrograman Arduino (C/C++) 21
2.9. Platform.io 22
2.10. Kajian penelitian sebelumnya 23
2.11. Kerangka Berpikir 27
BAB 3 PERANCANGAN SISTEM DAN IMPLEMENTASI SISTEM 28
3.1. Metode Penelitian 28
3.2. Flowchart sistem 44

BAB 4 PEMBAHASAN DAN ANALISIS 49
4.1. Gambaran Umum Perancangan 49
4.2. Proses Perakitan PCB.....50
4.3. Rancangan Pengujian 52
4.4. Hasil Pengujian 53
4.5. Hasil analisis 57
BAB 5 KESIMPULAN DAN SARAN.....62
5.1. Kesimpulan.....62
5.2. Saran 62
DAFTAR PUSTAKA.....64
LAMPIRAN.....67

DAFTAR GAMBAR

Gambar 2.1. Modul Sensor BME688 12
Gambar 2.2. Modul ESP32 16
Gambar 2.3. P5 LED Matrix Display 17
Gambar 2.4. Modul Kartu SD 18
Gambar 2.5. Modul Step-Down 19
Gambar 2.6. Modul RTC DS1307 20
Gambar 2.7. Bahasa Pemrograman C++ 21
Gambar 2.8 Platform.io 22
Gambar 3.1. Datasheet ESP32 30
Gambar 3.2. Datasheet BME688 32
Gambar 3.3. Datasheet RTC 32
Gambar 3.4. Datasheet Module MikroSD 33
Gambar 3.5. Datasheet Panel P5 34
Gambar 3.6. Desain 3D komponen 37
Gambar 3.7. Diagram Blok Sistem 38
Gambar 3.8. Proses pembutan Program 40
Gambar 3.9. Hasil Pengujian Pada Terminal Monitor 43
Gambar 3.10. Flowchart Sistem 45
Gambar 4.1. Jalur rangkaian PCB 50
Gambar 4.2. Hasil Rangkaian pada PCB 50
Gambar 4.3. Dokumentasi Alat 53
Gambar 4.4. Grafik Perbandingan Data Pada 10 Tempat Yang Berbeda 55
Gambar 4.5. Grafik Data Kualitas Udara Pada Tanggal 25 Mei 2025 61

DAFTAR TABEL

Tabel 2.1. Konveri Nilai Konsentrasi.....9
Tabel 2.2. Indeks Standar Pencemaran Udara 10
Tabel 2.3. Indeks Kualitas Udara Berdasarkan IAQ.....14
Tabel 3.1. Alat yang Digunakan dalam Perancangan.....35
Tabel 3.2. Bahan yang Digunakan dalam Perancangan 35
Tabel 3.3. Konfigurasi pin pada rangkaian.....37
Tabel 4.1. Perolehan Data Rata-Rata Selama 1 Jam (13:36-14:40).....53
Tabel 4.2. Data Hasil Pengujian Pada 10 Tempat Dengan Rentang Waktu 30 Menit 54
Tabel 4.3. Hasil Pengujian Pada Tempat Pembakaran Sampah.....56
Tabel 4.4. Ringkasan Akurasi Utama BME688.....58
Tabel 4.5. Rekap Rata-rata data per 1 Jam pada tanggal 25 Mei 2025 58

BAB 1 PENDAHULUAN

1.1. Latar Belakang

Pencemaran udara merupakan salah satu ancaman terbesar terhadap kesehatan masyarakat di dunia. Organisasi Kesehatan Dunia (WHO) mencatat bahwa lebih dari 4,2 juta kematian dini setiap tahun disebabkan oleh paparan polusi udara ambien (outdoor). Di Indonesia sendiri, jumlah kematian akibat pencemaran udara diperkirakan mencapai lebih dari 123.000 kasus per tahun. Paparan jangka panjang terhadap polusi udara, terutama partikel halus seperti PM2.5, dapat menyebabkan berbagai penyakit serius seperti kanker paru-paru, penyakit jantung, stroke, dan gangguan pernapasan kronis(Syuhada dkk., 2023). Salah satu organ yang paling terdampak oleh polusi udara adalah paru-paru, karena sistem pernapasan menjadi jalur utama masuknya partikel dan gas pencemar ke dalam tubuh. Partikel halus berukuran kurang dari 2,5 mikrometer (PM2.5) mampu masuk hingga ke alveoli paru-paru dan mengganggu pertukaran oksigen, memicu peradangan kronis, serta merusak jaringan paru secara bertahap. Paparan jangka panjang terhadap polutan ini telah dikaitkan dengan peningkatan risiko Penyakit Paru Obstruktif Kronik (PPOK), kanker paru-paru, serta penurunan fungsi paru secara umum, bahkan pada individu yang tidak memiliki riwayat merokok. Terdapat hubungan kausal yang signifikan antara paparan polusi udara dan kejadian penyakit saluran pernapasan pada masnyarkat, terutama pada kelompok anak-anak dan lansia[1]

Selain PM2.5, polutan lain seperti nitrogen dioksida (NO_2) dan ozon troposferik (O_3) juga memberikan kontribusi besar terhadap kerusakan sistem pernapasan. NO_2 yang banyak dihasilkan dari pembakaran kendaraan bermotor dapat menyebabkan iritasi saluran pernapasan dan memperburuk gejala asma. Sementara itu, ozon pada tingkat permukaan tanah bersifat toksik dan dapat menyebabkan penyempitan saluran pernapasan, yang berakibat pada sesak napas, batuk, dan bahkan penurunan fungsi paru dalam jangka panjang. Kelompok yang paling rentan terhadap dampak pencemaran udara ini adalah anak-anak, lansia, dan penderita penyakit kronis. Anak-anak yang tinggal di daerah dengan tingkat polusi tinggi memiliki risiko lebih besar mengalami gangguan perkembangan paru-paru serta lebih rentan terserang pneumonia dan bronkitis. Studi oleh

Syuhada et al. (2023) menunjukkan bahwa polusi udara di Jakarta secara signifikan berhubungan dengan peningkatan kasus penyakit saluran pernapasan, khususnya pada anak-anak dan lansia[2], serta menimbulkan beban ekonomi kesehatan yang sangat besar.

Dengan demikian, pencemaran udara tidak hanya berdampak pada lingkungan tetapi juga memiliki konsekuensi serius terhadap kesehatan manusia, terutama pada sistem pernapasan. Penelitian ini bertujuan memberikan informasi kualitas udara kepada pengguna secara real-time. Sistem ini dirancang untuk membantu masyarakat mengetahui kualitas udara di sekitar tempat tinggal mereka terutama di tempat-tempat pelayanan masyarakat (Hidayati dkk., 2024)Pembuatan alat sistem monitoring air quality index (AQI) berbasis sensor BME688 dengan visualisasi data P5 dislpay di area pelayanan public menjadi solusi untuk masalah tersebut, alat ini di rancang dengan sistem yang memudahkan orang-orang untuk melihat hasil data dari pengukuran kualitas udara secara langsung lewat panel display P5 yang digunkan, tidak hanya itu sensor yang digunakan pada alat ini pun merupakan sebuah sensor yang dapat mengukur tingkat kualitas udara dengan sangat baik dan memiliki berbagai macam fungsi serta fitur yang sangat bermamfaat guna mengukur kualitas secara efektif dan akurat. Menurut Rizal et al. (2023), sistem pemantauan kualitas udara yang dirancang dengan sensor berbasis gas mampu memberikan informasi akurat mengenai kondisi udara di lingkungan sekolah, serta efektif dalam mendeteksi keberadaan gas berbahaya dan perubahan kualitas udara secara real-time[3].

Sensor BME688 adalah sebuah sensor yang dirancang khusus untuk mengukur kualitas udara secara akurat dan efisien. Sensor BME688 ini memiliki kemampuan mendeteksi berbagai parameter lingkungan seperti suhu, kelembapan, tekanan udara, serta gas-gas berbahaya seperti senyawa organik volatil (VOC), karbon monoksida (CO), dan hidrogen (H_2) dalam konsentrasi sangat rendah (ppb). Selain itu, sensor ini dilengkapi dengan kecerdasan buatan (AI) yang memungkinkan pengenalan pola gas secara spesifik, menjadikannya ideal untuk aplikasi monitoring kualitas udara secara real-time(Klibanov & Klibanov, t.t.). Kelebihan dari sensor ini adalah ukurannya yang kompleks, harga yang terjangkau, dan kemampuan untuk memberikan data secara real-time dengan akurasi yang tinggi, sehingga sangat cocok untuk diterapkan dalam sistem AQI "Hasil

eksperimen juga menunjukkan bahwa sensor BME688 mampu memberikan informasi yang berguna dan detail mengenai kualitas udara, baik di lingkungan terkontrol maupun tidak terkontrol. Sensor ini cocok untuk mendeteksi kualitas udara yang dirasakan manusia, dengan pembacaan CO_2 yang juga signifikan untuk gas lain seperti CO[4].

Sehingga sensor BME688 memiliki keunggulan dalam kemampuannya untuk mendeteksi polusi udara secara lebih detail dan memungkinkan pemantauan kualitas udara di berbagai kondisi dan lokasi. Sensor ini dilengkapi dengan teknologi AI (Artificial Intelligence) yang dapat meningkatkan akurasi pengukuran dengan menganalisis data dari lingkungan secara lebih cerdas. Dalam konteks ini, sensor BME688 diharapkan dapat berperan sebagai alat utama dalam sistem AQI yang akan dirancang, memberikan data yang lebih akurat mengenai kondisi udara di sekitar wilayah yang dipantau.

Penelitian ini bertujuan untuk merancang dan mengembangkan sistem AQI berbasis sensor BME688 yang dapat mengukur dan memantau kualitas udara dengan akurat dan efisien. Melalui penerapan teknologi ini, diharapkan dapat memberikan kontribusi yang signifikan dalam meningkatkan kualitas hidup masyarakat, memperbaiki lingkungan, serta memberikan manfaat bagi keberlangsungan alam.

Pemantauan kualitas udara bukan hanya masalah teknis, tetapi juga menjadi landasan untuk pengambilan kebijakan publik yang berkelanjutan. Di banyak negara, termasuk Indonesia, kurangnya data akurat mengenai polusi udara menyebabkan kebijakan yang diambil tidak selalu efektif dalam mengurangi dampak polusi. Oleh karena itu, penerapan sistem pemantauan udara berbasis teknologi canggih, seperti sensor BME688, dapat membantu pemerintah dan instansi terkait dalam merumuskan kebijakan yang lebih tepat sasaran. Data real-time yang diperoleh melalui sistem AQI memungkinkan pengambilan keputusan yang lebih cepat dan responsif terhadap situasi polusi yang berbahaya.

Sistem AQI yang berbasis sensor BME688 dapat diintegrasikan dengan teknologi Internet of Things (IoT) untuk menciptakan jaringan pemantauan kualitas udara yang lebih luas dan efisien. Dengan memanfaatkan teknologi IoT, kita bisa terus-menerus mengumpulkan informasi tentang kualitas udara, menganalisis pola polusi, dan

mengambil tindakan yang diperlukan untuk menjaga lingkungan yang sehat dan aman(Fredy Agung Dwi Cahyono & Denny Irawan, 2024), sensor-sensor yang terhubung dapat mengirimkan data secara langsung ke platform cloud, memungkinkan pemantauan kualitas udara di berbagai lokasi secara real-time. Ini memberikan keuntungan dalam hal akurasi data yang lebih tinggi dan memberikan kemampuan untuk mendeteksi polusi udara secara lebih detail, bahkan pada area yang sulit dijangkau. Konsep smart city yang mengintegrasikan pemantauan kualitas udara berbasis IoT dapat menjadi langkah maju untuk menciptakan kota yang lebih sehat dan ramah lingkungan.

Penurunan kualitas udara tidak hanya berdampak pada kesehatan fisik, tetapi juga memiliki konsekuensi ekonomi yang besar. Kerugian ekonomi akibat dampak kesehatan dari pencemaran udara contohnya di Jakarta diperkirakan mencapai USD 2,94 miliar per tahun, atau sekitar 2,2% dari total Produk Domestik Regional Bruto (PDRB)(Syuhada dkk., 2023), Bahkan Polusi udara di Provinsi DKI Jakarta memberikan kontribusi negatif terhadap perekonomian sebesar IDR 8,98 triliun, menunjukkan perlunya sinergi kebijakan antara pemerintah, dunia usaha, akademisi, media, dan tokoh masyarakat(Sri Susilo dkk., 2024). Pencemaran udara dapat menurunkan produktivitas kerja karena meningkatkan angka ketidakhadiran akibat penyakit pernapasan atau kondisi medis lainnya. Data dari Organisasi Kesehatan Dunia (WHO) menunjukkan bahwa polusi udara dapat menurunkan produktivitas secara signifikan, terutama di sektor-sektor yang bergantung pada pekerja lapangan atau outdoor. Dalam konteks ini, sistem AQI yang efektif dapat memberikan informasi penting untuk mencegah dampak negatif terhadap produktivitas, serta untuk melindungi tenaga kerja dan menciptakan lingkungan kerja yang lebih sehat.[5]

Walaupun teknologi pemantauan kualitas udara berbasis sensor sudah ada, tantangan besar masih muncul dalam memastikan kualitas dan akurasi pengukuran, terutama di area yang terpengaruh oleh faktor-faktor eksternal seperti perubahan cuaca, polusi lokal, dan tingkat kebisingan. Oleh karena itu, penelitian ini juga mengkaji solusi untuk mengoptimalkan penggunaan sensor BME688 dalam kondisi yang berbeda-beda, baik di area urban maupun pedesaan. Ini bertujuan untuk memberikan data yang lebih

valid dan konsisten, yang pada gilirannya akan membantu dalam pengambilan kebijakan yang lebih baik.

Selain teknologi yang digunakan, penting juga untuk meningkatkan kesadaran masyarakat akan pentingnya pemantauan kualitas udara dan dampak negatif polusi udara terhadap kesehatan. Sistem AQI berbasis sensor BME688 dapat menjadi alat yang sangat berguna untuk edukasi publik. Dengan memberikan informasi secara real-time mengenai kualitas udara, masyarakat dapat mengambil langkah-langkah preventif, seperti mengurangi aktivitas di luar ruangan ketika kualitas udara buruk, atau menggunakan alat pelindung pernapasan. Kampanye pendidikan yang dilengkapi dengan teknologi pemantauan udara dapat mendorong perilaku yang lebih sadar akan lingkungan.

Pemantauan kualitas udara yang terus-menerus melalui sistem seperti AQI akan membantu dalam merencanakan langkah-langkah jangka panjang untuk pengelolaan kualitas udara yang berkelanjutan. Dengan mengumpulkan data yang lebih terperinci dalam jangka waktu yang lama, sistem ini dapat memberikan gambaran mengenai tren perubahan kualitas udara dan pola polusi. Hal ini menjadi dasar untuk merancang kebijakan atau program yang lebih terarah, seperti pengurangan emisi kendaraan atau perbaikan pengelolaan sampah, yang pada akhirnya akan berkontribusi pada kualitas udara yang lebih baik dan keberlanjutan lingkungan.[6]

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, terdapat beberapa permasalahan utama yang menjadi fokus penelitian ini, yaitu:

1. Bagaimana sensor BME688 dapat digunakan untuk mengukur kualitas udara secara akurat dan efisien di area pelayanan publik, termasuk parameter suhu, kelembapan, tekanan, dan gas berbahaya?
2. Apa saja tantangan yang dihadapi dalam pengukuran kualitas udara menggunakan sensor BME688, khususnya pada lingkungan dengan tingkat polusi dan kondisi cuaca yang berubah-ubah?

3. Bagaimana data kualitas udara dapat ditampilkan secara real-time melalui panel P5 Display dan disimpan secara lokal menggunakan kartu SD, serta sejauh mana efektivitas dan akurasinya dalam menyampaikan informasi kepada pengguna?
4. Bagaimana alat ini dapat dimanfaatkan sebagai media edukasi untuk meningkatkan kesadaran masyarakat terhadap pentingnya pemantauan kualitas udara dan dampaknya terhadap kesehatan?

1.3. Batasan Masalah dan Asumsi

Adapun batasan masalah dalam penelitian ini adalah:

1. Sistem dirancang masih berupa prototipe dan masih dalam tahap uji coba sehingga belum mendapatkan nilai yang *actual*.
2. Hanya membahas bagaimana perancangan dan penelitian pembuatan skema alat dan estimasi penggunaan alat dalam jangka waktu dan konsep kerja.
3. Data yang dihasilkan belum sepenuhnya spesifik karena dalam masa perancangan dan uji coba.
4. Pengujian system fokus pada kemampuan BME688 dalam membaca suhu, kelembaban, tekanan udara, dan berbagai macam gas uji coba yang digunakan sebagai indeks pengukur kualitas udara.

1.4. Tujuan Penelitian

Penelitian ini memiliki beberapa tujuan utama, yaitu:

1. Mengimplementasikan sensor BME688 sebagai alat ukur kualitas udara yang mampu mendeteksi parameter suhu, kelembapan, tekanan udara, dan konsentrasi gas berbahaya secara efisien di area pelayanan publik.
2. Mengidentifikasi dan menganalisis tantangan teknis dalam pengukuran kualitas udara menggunakan sensor BME688, terutama pada kondisi lingkungan dengan variasi tingkat polusi dan perubahan cuaca.
3. Merancang dan mengembangkan sistem visualisasi data kualitas udara secara real-time menggunakan panel P5 Display, serta menyimpan data secara lokal melalui kartu SD untuk mendukung keandalan dan efektivitas penyampaian informasi kepada pengguna.

4. Mengevaluasi potensi pemanfaatan alat sebagai sarana edukasi masyarakat, dalam rangka meningkatkan kesadaran terhadap pentingnya pemantauan kualitas udara dan dampaknya terhadap kesehatan lingkungan.

1.5. Manfaat Penelitian

Manfaat yang diharapkan dari penelitian ini antara lain:

1. Berkontribusi terhadap Ilmu Pengetahuan dan Teknologi Terapan sehingga penelitian ini memperkaya literatur ilmiah dalam bidang pemantauan kualitas udara berbasis sensor, dengan pendekatan sederhana namun efektif.
2. Sebagai referensi untuk penelitian selanjutnya, Rancangan sistem ini dikembangkan untuk menjadi dasar atau model untuk pengembangan fitur yang lebih kompleks di masa mendatang.
3. Sebagai pengembang kompetensi teknis sehingga peneliti mendapatkan pengalaman praktis dalam penggabungan sensor BME688, mikrokontroler ESP32, RTC, dan modul penyimpanan, memperkuat keterampilan teknis dan analisis sistem.
4. Rancangan sistem membuka peluang kolaborasi antara bidang teknologi dan kebijakan publik, khususnya dalam konteks lingkungan dan kesehatan masyarakat.
5. Meningkatkan Kesadaran dan Pemahaman Lingkungan Alat ini menjadi media edukatif untuk meningkatkan kesadaran masyarakat terhadap pentingnya kualitas udara, serta memahami parameter lingkungan secara langsung.
6. Masyarakat dapat mengurangi risiko gangguan pernapasan akibat kualitas udara buruk, sehingga menekan beban biaya kesehatan. Selain itu, data dari alat ini dapat menjadi masukan untuk kebijakan lokal yang lebih sehat dan ramah lingkungan.
7. Penggunaan alat ini diharapkaan dapat mendorong munculnya kesadaran terhadap pentingnya inovasi teknologi yang berkelanjutan dan berbasis kebutuhan lingkungan sekitar.

BAB 2 LANDASAN TEORI

2.1. Indeks Kualitas Udara (AQI)

Kualitas udara adalah parameter krusial yang menunjukkan sejauh mana udara di suatu daerah bebas dari bahan pencemar atau terpapar pencemaran. Untuk secara teratur dan kuantitatif menilai kualitas ini, diterapkan sebuah norma yang disebut sebagai Indeks Kualitas Udara (Air Quality Index/AQI).

2.1.1. Definisi dan Fungsi AQI

Indeks Kualitas Udara (AQI) ialah sebuah sistem pengukuran yang mencerminkan tingkat kebersihan atau pencemaran udara di suatu area. Nilai AQI mulai dari 0, yang menandakan udara sangat bersih, sampai lebih dari 300 yang menunjukkan kondisi udara sangat berbahaya. Semakin tinggi angka AQI, semakin buruk kualitas udara yang ada dan potensi risiko bagi kesehatan manusia semakin tinggi (nafas.co.id, 2024). AQI dibedakan menjadi enam level, masing-masing diwarnai berbeda untuk membantu masyarakat dengan mudah memahami tingkat kualitas udara di lokasi mereka. Umumnya, angka AQI di bawah 100 masih dianggap aman bagi mayoritas orang. AQI memiliki rumus perhitungan sebagai berikut:

$$\text{AQI} = \left(\frac{I_{\text{high}} - I_{\text{low}}}{C_{\text{high}} - C_{\text{low}}} \right) \times (C - C_{\text{low}}) + I_{\text{low}}$$

Keterangan:

- C = Konsentrasi polutan yang terukur (misalnya PM2.5 dalam µg/m³)
- C_low = Ambang batas bawah dari kategori konsentrasi tempat nilai C berada
- C_high = Ambang batas atas dari kategori konsentrasi tempat nilai C berada
- I_low = Nilai AQI pada batas bawah (kategori AQI yang sesuai)
- I_high = Nilai AQI pada batas atas (kategori AQI yang sesuai)

Di Indonesia, alat yang dipakai untuk menilai kualitas udara disebut Indeks Standar Pencemaran Udara (ISPU). Sistem ini juga menggunakan angka tak beraturan untuk menunjukkan kondisi udara, yang dihitung berdasarkan sejauh mana dampaknya

pada kesehatan manusia, lingkungan, dan makhluk hidup lainnya. ISPU juga berfungsi sebagai sistem peringatan dini, khususnya di area yang rentan terhadap kebakaran hutan dan lahan. Ketentuan untuk ISPU diatur dalam Peraturan Menteri Lingkungan Hidup dan Kehutanan No. 14 Tahun 2020, yang menggantikan regulasi sebelumnya dari tahun 1997. Dalam peraturan tersebut, ISPU dihitung berdasarkan tujuh jenis polutan utama, yaitu PM10, PM2.5, NO_2, SO_2, CO, O_3, dan HC, dengan penambahan parameter PM2.5 dan HC dari ketentuan sebelumnya.

Indeks AQI dirancang berdasarkan konsentrasi berbagai parameter pencemar udara yang umum di sekitar kita, seperti:

- PM2.5 (partikel halus dengan ukuran ≤2.5 mikrometer)
- PM10 (partikel lebih besar sampai 10 mikrometer)
- Karbon Monoksida (CO)
- Ozon (O_3)
- Nitrogen Dioksida (NO_2)
- Sulfur Dioksida (SO_2)
- Senyawa Organik Volatil (VOCs)

Tabel 2.1. Konveri Nilai Konsentrasi

ISPU	24 Jam PM10 (µg/m³)	24 Jam PM2.5 (µg/m³)	24 Jam SO_2 (µg/m³)	24 Jam CO (µg/m³)	24 Jam O_3 (µg/m³)	24 Jam NO_2 (µg/m³)	24 Jam HC (µg/m³)
0 - 50	50	15,5	52	4000	120	80	45
51 - 100	150	55,4	180	8000	235	200	100
101 - 200	350	150,4	400	15000	400	1130	215
201 - 300	420	250,4	800	30000	800	2260	432
>300	500	500	1200	45000	1000	3000	648

Keterangan:
- Data pengukuran selama 24 jam secara terus-menerus.
- Hasil perhitungan ISPU parameter partikulat (PM2.5) disampaikan tiap jam selama 24 jam.
- Hasil perhitungan ISPU parameter partikulat (PM10), sulfur dioksida (SO_2), karbon monoksida (CO), ozon (O_3), nitrogen dioksida (NO_2) dan hidrokarbon (HC), diambil nilai ISPU parameter tertinggi dan paling sedikit disampaikan setiap jam 09.00 dan jam 15.00.

Perhitungan ISPU ini didasarkan dari nilai ISPU batas atas, ISPU batas bawah, ambien batas atas, ambien batas bawah, dan konsentrasi ambien hasil pengukuran. Persamaan matematika perhitungan ISPU sebagai berikut:[6]

$$I = \frac{I_a - I_b}{X_a - X_b}(X_x - X_b) + I_b$$

Dengan keterangan sebagai berikut:

- I = ISPU terhitung
- Ia = ISPU batas atas
- Ib = ISPU batas bawah
- Xa = Konsentrasi ambien batas atas (µg/m3)
- Xb = Konsentrasi ambien batas bawah (µg/m3)
- Xx = Konsentrasi ambien nyata hasil pengukuran (µg/m3)

Tabel 2.2. Indeks Standar Pencemaran Udara

	Rentang	Kategori Polusi Udara	Keterangan
1	1-50	Baik	Tingkat mutu udara yang sangat baik, tidak memberikan efek negatif terhadap manusia, hewan dan tumbuhan
2	51-100	Sedang	Tingkat mutu udara masih dapat diterima pada kesehatan manusia, hewan, dan tumbuhan
3	101-200	Tidak Sehat	Tingkat mutu udara yang bersifat merugikan pada manusia, hewan, dan tumbuhan
4	201-300	Sangat Tidak Sehat	Tingkat mutu udara yang dapat meningkatkan risiko kesehatan pada sejumlah segmen populasi yang terpapar
5	301+	Berbahaya	Tingkat mutu udara yang dapat merugikan kesehatan serius pada populasi dan perlu penanganan cepat.

2.1.2. Dampak Kualitas Udara terhadap Kesehatan

Menurut pedoman yang ditetapkan oleh World Health Organization (WHO), paparan jangka panjang terhadap udara yang mengandung polutan di atas ambang batas dapat menyebabkan berbagai gangguan kesehatan serius, antara lain:

- Penyakit pernapasan kronis seperti asma dan bronkitis
- Peningkatan risiko penyakit jantung dan stroke
- Kanker paru-paru pada paparan ekstrem dan berkelanjutan

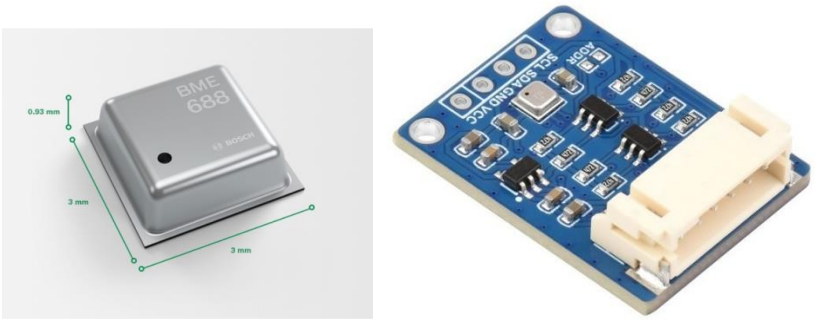
Oleh karena itu, pengukuran kualitas udara secara periodik menjadi penting, tidak hanya untuk tujuan ilmiah, tetapi juga untuk perlindungan kesehatan masyarakat.

2.1.3. Relevansi AQI dalam Sistem Monitoring

Dalam konteks penelitian ini, AQI dijadikan sebagai acuan atau tolok ukur dalam menilai data kualitas udara yang diperoleh dari sensor BME688. Meskipun alat yang dirancang belum mengukur seluruh parameter AQI secara penuh, beberapa indikator seperti gas VOC, suhu, dan kelembapan memberikan gambaran awal terhadap profil udara yang berkaitan erat dengan kondisi lingkungan mikro suatu lokasi.[7]

Nilai IAQ yang diperoleh dari sensor ini yang nantinya akan di konversi menjadi nilai AQI untuk menentukan nilai kualitas udara yang diukur oleh alat ini, perlu di garis bawahi bahwasanya nilai AQI yang di ukur dari alat ini belum mencakup seluruh parameter yang dibutuhkan untuk menentukan nilai AQI yang sebenarnya sehingga perhitungan yang dilakukan dalam alat ini mengacu pada rumus berdasarkan datasheet yang dimana perhitungan ini merupakan hasil algoritma BSEC dan tidak dilakukan perhitungan secara manual karena tidak sesuai dengan jumlah parameter yang digunakan serta nilai yang diperoleh tidak seakurat hasil algoritma. dan rumus perhitungan pada algoritma pun tidak bisa di bongkar karena merupakan otoritas milik perusahaan dan memiliki hak cipta.

2.2. Sensor BME688



Gambar 2.1. Modul Sensor BME688

Sumber: https://www.bosch-sensortec.com/products/environmental-sensors/gas-sensors/bme688/

Sensor BME688 merupakan modul sensor canggih yang dikembangkan oleh Bosch Sensortec, dirancang untuk mengukur berbagai parameter lingkungan secara bersamaan. Perbedaan antara BME688 dan pendahulunya yaitu BME680 terletak dalam hal kemampuan kecerdasan buatan (AI) yang dimiliki oleh BME688. Sensor ini memungkinkan pengembangan aplikasi yang lebih cerdas dalam pemantauan kualitas udara(Klibanov & Klibanov, t.t.). Sensor ini menggabungkan kemampuan pengukuran suhu, kelembapan, tekanan udara, serta kemampuan deteksi gas, menjadikannya alat yang sangat komprehensif untuk pemantauan kualitas udara.[8]

2.2.1. Karakteristik Umum

BME688 adalah pengembangan dari pendahulunya, yaitu BME680. Selain mempertahankan semua fitur utama dari BME680, sensor BME688 dilengkapi dengan kemampuan pemrosesan kecerdasan buatan (AI) yang memungkinkan klasifikasi profil gas secara lebih cerdas dan adaptif. Hal ini memungkinkan BME688 untuk digunakan dalam aplikasi yang membutuhkan identifikasi pola gas atau pengenalan lingkungan berbasis udara, seperti:

- Pemantauan kualitas udara dalam ruangan
- Deteksi VOC (Volatile Organic Compounds)

- Analisis lingkungan berbasis profil gas

2.2.2. Prinsip Kerja

Sensor ini bekerja berdasarkan prinsip pemanasan mikro dan pembacaan resistansi. Di dalamnya terdapat elemen pemanas kecil (gas sensing heater) yang digunakan untuk mengaktifkan material sensitif terhadap gas. Ketika udara mengalir ke sensor, gas-gas tertentu akan berinteraksi dengan material sensor, mengubah nilai resistansinya. Perubahan resistansi ini kemudian dikonversi menjadi data digital untuk dianalisis.

Selain itu, sensor ini mampu bekerja dalam multi-mode dengan pola pemanasan yang dapat disesuaikan, memungkinkan untuk mendeteksi berbagai jenis gas dengan lebih tepat. Proses ini sangat penting dalam membedakan antara jenis gas seperti hidrogen (H_2), karbon monoksida (CO), metana (CH_4), dan senyawa VOC lainnya.

2.2.3. Parameter yang Dihasilkan

Sensor BME688 mampu menghasilkan data lingkungan sebagai berikut:

- Suhu (°C)
  Diukur menggunakan termistor internal dan dikompensasi dengan kalibrasi perangkat lunak.
- Kelembapan Relatif (%RH)
  Menunjukkan kadar uap air di udara, penting untuk kenyamanan dan kesehatan manusia.
- Tekanan Udara (hPa)
  Berguna dalam analisis kondisi atmosfer dan ketinggian (altitude estimation).
- Gas Resistance (Ohm)
  Indikator tingkat polutan di udara, yang dikonversi menjadi Indeks Kualitas Udara (IAQ) oleh algoritma.

Sensor BME688 ini jika menggunakan BSEC (Bosch Software Environmental Cluster), secara otomatis menghitung nilai IAQ berdasarkan data gas, suhu, kelembaban,

dan tekanan. Nilai ini disajikan dalam bentuk angka yang merepresentasikan tingkat polusi udara, biasanya dalam skala 0–500 (semakin tinggi, semakin buruk kualitasnya), dengan klasifikasi seperti:[9].

Tabel 2.3. Indeks Kualitas Udara Berdasarkan IAQ

Nilai IAQ	Kategori Kualitas Udara
0–50	Sangat baik
51–100	Baik
101–150	Sedang
151–200	Buruk
201–300	Sangat buruk
301–500	Berbahaya

Nilai IAQ ini nantinya akan dikonversi menjadi nilai AQI pada alat pendeteksi kualitas udara ini sebagai indeks pengukur kualitas udara yang menjadi acuan pengukur kualitas udara pada alat ini.

2.2.4. Keunggulan Sensor BME688

Sensor BME688 buatan Bosch Sensortec merupakan sensor canggih generasi terbaru yang dirancang khusus untuk kebutuhan sistem pemantauan lingkungan yang komprehensif, termasuk aplikasi pemantauan kualitas udara berbasis Internet of Things (IoT). Keunggulan utama BME688 terletak pada kemampuannya untuk mengukur berbagai parameter lingkungan dalam satu chip serta dukungan kecerdasan buatan (AI) untuk deteksi gas yang lebih spesifik. Dalam konteks rangkaian yang dirancang

menggunakan mikrokontroler seperti ESP32, sensor ini sangat cocok karena menawarkan performa tinggi dengan konsumsi daya rendah.

Berikut adalah kelebihan utama BME688 dalam sistem tersebut:

- Multifungsi dalam satu chip (pengukuran 4 parameter lingkungan sekaligus)
- Dukungan AI (Artificial Intelligence) untuk klasifikasi profil gas
- Konsumsi daya rendah, cocok untuk perangkat portabel dan IoT
- Kompak dan ringan, mudah diintegrasikan ke dalam sistem mikrokontroler seperti ESP32
- Kompatibel dengan antarmuka I2C dan SPI, memudahkan integrasi dengan berbagai platform pengembangan

2.2.5. Aplikasi dalam Penelitian

Dalam konteks penelitian ini, BME688 digunakan untuk memantau kualitas udara secara real-time di lingkungan tempat pelayanan masyarakat. Data yang dihasilkan dari sensor ini ditampilkan secara langsung pada panel display P5 dan disimpan secara berkala ke kartu SD, sehingga pengguna dapat mengevaluasi kualitas udara dari waktu ke waktu tanpa perlu koneksi internet.

2.3. Mikrokontroler ESP32

Gambar 2.2. Modul ESP32

Sumber: https://www.edukasielektronika.com/2019/07/arsitektur-dan-fitur-esp32-module-esp32.html

ESP32 adalah mikrokontroler berbasis SoC (System on Chip) yang dikembangkan oleh Espressif Systems, dikenal karena kinerjanya yang tinggi dan konsumsi daya yang rendah. Mikrokontroler ESP32 digunakan sebagai pusat kendali sistem, mengintegrasikan berbagai sensor untuk mendeteksi gas berbahaya. Data yang dikumpulkan oleh sensor kemudian diproses oleh ESP32 dan dikirimkan secara real-time memungkinkan pemantauan kualitas udara secara langsung dan pemberian peringatan dini apabila terjadi peningkatan kadar gas berbahaya di atas ambang batas yang ditentukan(Putra Pratama & Faiq Muhammad, t.t.). ESP32 ini merupakan salah satu mikrokontroler yang banyak digunakan dalam pengembangan sistem berbasis Internet of Things (IoT). Menurut Babiuch, Foltýnek, dan Smutný (2019), ESP32 memiliki kemampuan yang baik dalam pengukuran dan pemrosesan data, serta dapat terintegrasi dengan berbagai sensor pintar dan modul IoT untuk menyediakan data secara efisien ke sistem pusat(Babiuch dkk., 2019).

2.3.1. Spesifikasi Teknis

ESP32 memiliki fitur berikut yang menjadikannya cocok untuk proyek monitoring berbasis sensor:

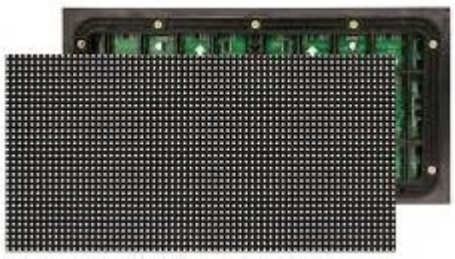
- Prosesor dual-core 32-bit Xtensa LX6 dengan frekuensi hingga 240 MHz

- Memori SRAM dan flash internal
- GPIO (General Purpose Input Output) yang banyak dan fleksibel
- Dukungan antarmuka I2C, SPI, UART, ADC, dan PWM

2.3.2. Fungsi dalam Sistem

Dalam penelitian ini, ESP32 bertugas sebagai otak dari sistem, yakni mengontrol pembacaan data dari sensor BME688, mengelola waktu dengan RTC DS1307, mengolah dan menampilkan data ke panel P5, serta merekam data ke dalam modul kartu SD untuk penyimpanan historis. Sistem monitoring kualitas udara berbasis NodeMCU ESP32 juga dapat dikembangkan untuk mendeteksi gas-gas berbahaya di kawasan industri, seperti karbon monoksida (CO), nitrogen dioksida (NO_2), dan hidrogen sulfida (H_2S), dengan menambahkan sensor MQ dan peringatan buzzer jika batas ambang dilampaui(Harpad dkk., 2022). Sehingga mikrokontroler ini sangat cocok digunakan pada alat pendeteksi kualias udara ini karena sangat kompleks dan mudah untuk di kembangkan kedepannya dengan fitur-fitur yang memadai.

2.4. Panel P5 (LED Matrix Display)



Gambar 2.3. P5 LED Matrix Display

Sumber: https://digilog.pk/products/outdoor-p5-led-display-module

Panel P5 adalah jenis panel LED matriks dengan pitch 5 mm antar-piksel. Panel ini umumnya digunakan untuk menampilkan informasi teks atau grafik dalam bentuk tampilan dinamis yang mudah dibaca dari jarak jauh.

2.4.1. Spesifikasi Umum

- Resolusi: biasanya 64×32 piksel
- Antarmuka komunikasi: HUB75
- Tegangan operasi: 5V
- Dapat menampilkan huruf, angka, grafik bergerak, dan animasi sederhana

2.4.2. Fungsi dalam Sistem

Panel P5 berperan sebagai media output visual, di mana data suhu, kelembapan, tekanan, dan indeks kualitas udara ditampilkan secara real-time. Informasi ini membantu pengguna memahami kondisi udara secara langsung tanpa perlu membuka aplikasi atau perangkat tambahan.

2.5. Modul Kartu SD

Gambar 2.4. Modul Kartu SD

Sumber: https://www.nn-digital.com/blog/2019/08/01/contoh-program-micro-sd-card-dengan-arduino/

Modul SD card berperan sebagai media penyimpanan eksternal di dalam sistem. Modul ini mendukung koneksi antarmuka SPI, memungkinkan hubungan yang mudah dengan ESP32.

2.5.1. Spesifikasi Umum

- Mendukung penggunaan kartu microSD berkapasitas hingga 32 GB (FAT16/FAT32)

• Kompatibel dengan antarmuka SPI

• Tegangan operasional: 3.3-5V (menggunakan konverter level tegangan)

2.5.2. Fungsi dalam Sistem

Modul ini berfungsi untuk menyimpan hasil pembacaan sensor secara berkala dalam bentuk file teks atau CSV. Penyimpanan lokal ini sangat bermanfaat ketika sistem tidak terhubung ke internet, memastikan data tetap dapat dikumpulkan dan dianalisis secara manual.

2.6. Modul Step-Down (DC-DC Buck Converter)

Gambar 2.5. Modul Step-Down

Sumber: https://ecadio.com/jual-modul-step-down-dc-lm2596

Modul step-down atau buck converter adalah rangkaian konversi daya DC yang digunakan untuk menurunkan tegangan input menjadi tegangan output yang lebih rendah, tanpa banyak membuang energi sebagai panas seperti pada regulator linier.

2.6.1. Prinsip Kerja

Modul ini bekerja dengan cara:

- Menghidupkan dan mematikan saklar elektronik (biasanya transistor) dengan frekuensi tinggi
- Menyaring sinyal menggunakan induktor dan kapasitor untuk menghasilkan tegangan output yang stabil

- Mengatur output sesuai kebutuhan beban melalui potensiometer atau kontrol otomatis

Contoh: jika sistem diberi input 12V dari adaptor atau baterai, modul step-down dapat menurunkan tegangan ke 5V atau 3.3V yang dibutuhkan oleh mikrokontroler dan komponen lainnya.

2.6.2. Fungsi dalam Sistem

Dalam sistem QAIR, modul step-down digunakan untuk menyesuaikan tegangan dari catu daya utama agar sesuai dengan kebutuhan ESP32 (3.3V) dan perangkat lain seperti panel P5 (5V). Dengan demikian, stabilitas dan keandalan sistem tetap terjaga, serta mencegah kerusakan akibat tegangan berlebih.

2.7. RTC DS1307 (Real Time Clock)

Gambar 2.6. Modul RTC DS1307

Sumber: https://abc-rc.pl/pl/products/modul-czasu-rtc-ds1307-zegar-czasu-rzeczywistego-arduino-6190.html

RTC DS1307 merupakan modul waktu yang berfungsi untuk menyimpan serta mengawasi waktu dengan akurat, bahkan ketika sistem utama tidak aktif, berkat dukungan daya dari baterai. Jam Real-Time (RTC) adalah alat yang dibuat untuk mempertahankan data waktu secara tepat, meskipun perangkat utama tidak berfungsi. RTC menggunakan sumber energi cadangan seperti baterai kancing, yang memungkinkan pencatatan waktu terus menerus tanpa bergantung pada pasokan listrik dari mikrokontroler. Dalam perangkat elektronik seperti alat ukur, RTC sangat penting untuk memberikan informasi waktu yang diperlukan saat pengumpulan data, penjadwalan otomatis, dan pencatatan waktu pada data sensor(Sundaresan & Perumal, t.t.).

2.7.1. Spesifikasi Teknis

- Format waktu 24 jam atau 12 jam (AM/PM)
- Komunikasi menggunakan I2C
- Baterai cadangan CR2032 untuk menjaga waktu tetap berjalan saat catu daya utama mati

2.7.2. Fungsi dalam Sistem

RTC DS1307 digunakan untuk menandai setiap data hasil pembacaan sensor dengan waktu yang akurat. Dengan demikian, pengguna dapat mengetahui kapan data tersebut diambil dan memungkinkan analisis tren berdasarkan waktu (misalnya per jam atau per hari).

2.8. Bahasa Pemrograman Arduino (C/C++)

Gambar 2.7. Bahasa Pemrograman C++

Sumber: https://fti.ars.ac.id/blog/content/program-sederhana-c-menghitung-luas-persegi

Dalam pengembangan sistem QAIR, digunakan bahasa pemrograman Arduino yang berbasis C/C++. Bahasa ini dirancang untuk memudahkan pemrograman perangkat keras mikrokontroler, terutama pada platform Arduino dan ESP32.[10]

2.8.1. Karakteristik Bahasa

- Struktur sintaks sederhana dan modular, memudahkan pengembangan program secara bertahap
- Mendukung pemrograman prosedural dan fungsional

- Didukung oleh banyak pustaka (library) yang disediakan oleh komunitas terbuka, seperti untuk RTC, SD Card, sensor BME688, dan panel LED
- Kompatibel dengan Arduino IDE dan PlatformIO, yang menyediakan antarmuka pengguna dan manajemen proyek yang intuitif

2.8.2. Peran dalam Sistem

Bahasa pemrograman ini digunakan untuk menulis skrip yang mengatur seluruh alur kerja sistem: mulai dari inisialisasi komponen, pembacaan sensor, pengelolaan waktu, penyimpanan data, hingga tampilan ke panel LED. Pendekatan ini memungkinkan pengendalian perangkat keras secara efisien dan responsif.

2.9. Platform.io

Gambar 2.8 Platform.io

Sumber: https://marketplace.visualstudio.com/.items?itemName=platformio.platformio-ide

PlatformIO berfungsi sebagai lingkungan pengembangan terpadu (*Integrated Development Environment/IDE*) yang mendukung pemrograman mikrokontroler secara efisien dan fleksibel. Dalam proyek ini, PlatformIO digunakan untuk menulis, mengatur, dan mengunggah program ke papan mikrokontroler seperti ESP32 yang mengendalikan sensor BME688 dan modul display seperti P5 RGB Matrix.

Dengan menggunakan PlatformIO, pengguna dapat:

- Mengelola pustaka (library) secara otomatis, seperti pustaka untuk sensor BME688, komunikasi I²C/SPI, serta pustaka tampilan grafis untuk P5 display.

- Mengatur berbagai konfigurasi proyek dan dependensi melalui file platformio.ini, sehingga mempermudah pengelolaan proyek secara modular dan skalabel.
- Melakukan pemrograman dan debugging secara langsung melalui antarmuka Visual Studio Code yang telah terintegrasi dengan PlatformIO.
- Melakukan kompilasi lintas platform (*cross-compilation*), sehingga proyek dapat dikembangkan di berbagai sistem operasi seperti Windows, Linux, maupun

2.10. Kajian penelitian sebelumnya

No	Penulis	Judul	Jenis Penulisan	Hasil Pembahasan
1	Yokesh Babu Sundaresan, Ashish Kumar Jaiswal, Kumaresan P	*A Smart Multi-purpose Remote Controlled Digital Clock*	Artikel Ilmiah	Pengembangan jam digital dengan RTC dan kontrol jarak jauh menggunakan Arduino, menjadi dasar untuk penggunaan modul waktu nyata (RTC) dalam sistem AQI.
2	Dita Putra Pratama, Nurchim, Nibras Faiq Muhammad	*Sistem Pemantauan Kualitas Udara IoT untuk Mitigasi Risiko Pekerja dan Masyarakat di Industri Genteng*	Artikel Ilmiah	Sistem monitoring kualitas udara berbasis IoT untuk lingkungan industri, relevan sebagai model mitigasi risiko bagi pekerja dan masyarakat sekitar.
3	Muhammad Rizal , Arham Arifin, Muhammad Furqan Rasyd, Andi Asvin	*Design and Implementation of a Real-Time Air Pollution Monitoring*	Artikel Ilmiah	Sistem monitoring udara real-time dengan antarmuka pengguna berbasis Android, memperluas pilihan UI

No	Penulis	Judul	Jenis Penulisan	Hasil Pembahasan
	Mahersatillah Suradi, Akbar Bahtiar	*System Based on Android*		dalam monitoring kualitas udara.
4	Andiko Pridiantoko Putro, Danu Arrival Hidayat, Farras Fauzan Heratama, Andy Dwi Cahyo, Dhany Eka Yulian, Yuliyanto Agung Prabowo	*Sistem Monitoring Kualitas Udara Menggunakan Mikrokontroler ESP32 dengan Sensor MQ2*	Artikel Ilmiah	Implementasi mikrokontroler ESP32 dengan sensor MQ2 untuk mendeteksi polusi gas, sesuai dengan teknologi dasar dalam monitoring kualitas udara.
5	Marek Babiuch. Petr Foltýnek. Pavel Smutný	*Using the ESP32 Microcontroller for Data Processing*	Artikel Ilmiah	Menjelaskan kemampuan ESP32 dalam memproses data sensor secara efisien, mendukung sistem AQI dalam akuisisi data real-time.
6	Fredy Agung Dwi Cahyono & Denny Irawan	*Rancang Bangun Sistem Pemantauan dan Kontrol Kualitas Udara Dalam Ruangan Berbasis IoT*	Artikel Ilmiah	Monitoring kualitas udara dalam ruangan disertai kontrol otomatis berbasis IoT, dapat menjadi pengembangkan lebih lanjut untuk system monitoring kualitas udara karena sangat kompitable

No	Penulis	Judul	Jenis Penulisan	Hasil Pembahasan
				dengan komponen yang digunakan.
7	Muhamad Reski Wijaya Kusuma	*Monitoring Kualitas Udara Ruangan dengan Sensor Debu dan Cahaya Menggunakan Metode Fuzzy Mamdani*	Tugas Akhir	Penerapan logika fuzzy dalam pengolahan data sensor debu dan cahaya, memberi referensi untuk pengambilan keputusan cerdas pada Pembuatan monitoring kualitas udara menggunakan sensor BME688 ini.
8	Bartolomius Harpad, Salmon, Rizky Meizal Saputra	*Sistem Monitoring Kualitas Udara di Kawasan Industri dengan NodeMCU ESP32*	Artikel Ilmiah	Pemantauan kualitas udara berbasis NodeMCU ESP32 di lingkungan industri, memberikan model implementasi sistem AQI di lapangan dengan system yang mendukung pengukuran gas bebahaya dan indikator buzzer untuk memberi peringatan Ketika indeks kualitas udara melebihi batas aman.
9	Rahmi Hidayati, Kartika Sari , Yasman Halawa ,	*Sistem Pemantauan Kualitas Udara Secara Real-Time*	Artikel Ilmiah	Sistem monitoring kualitas udara secara real-time dengan ESP32,

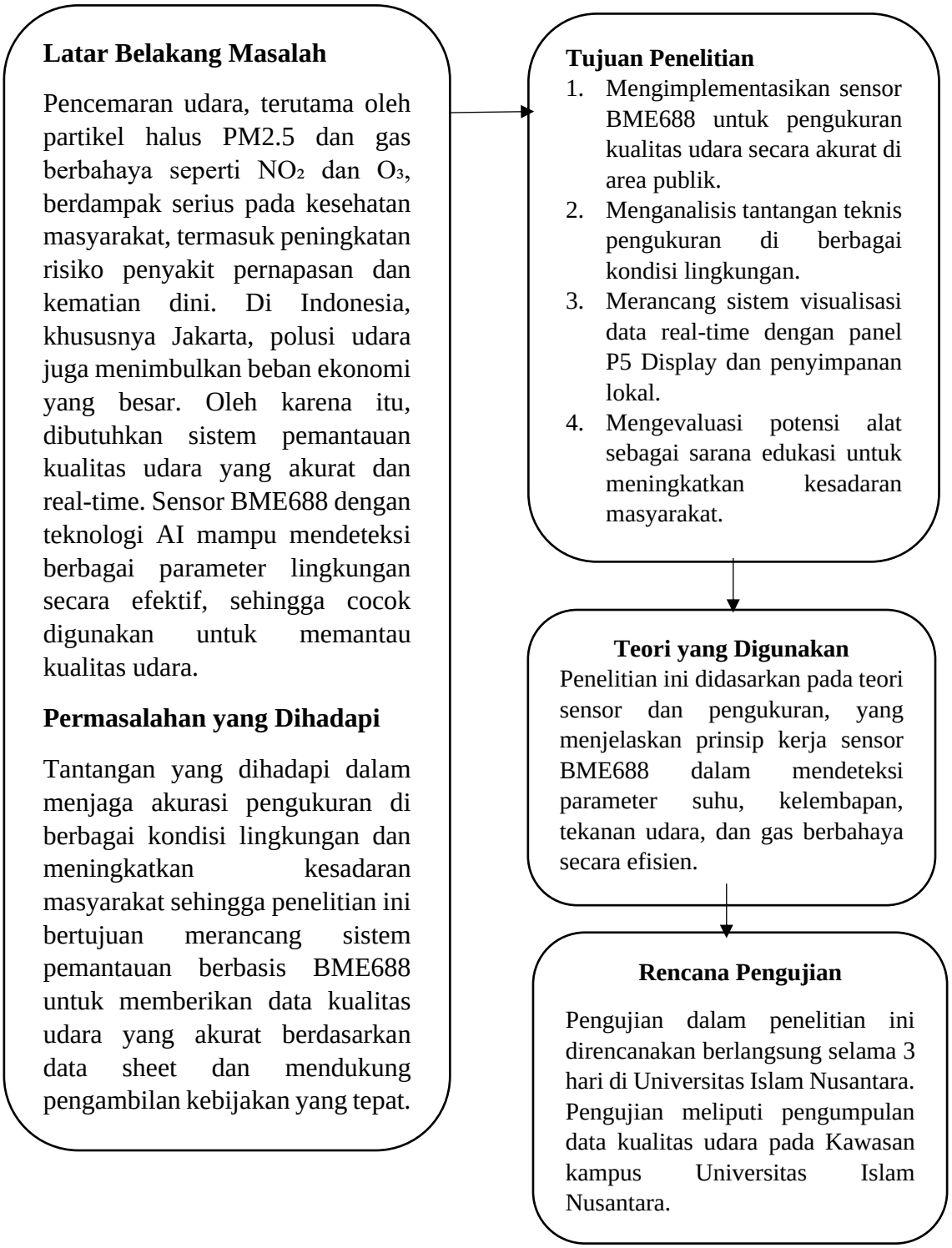
No	Penulis	Judul	Jenis Penulisan	Hasil Pembahasan
	Athif Tafrihan A. , M. Abdurrahman Harits	*Menggunakan ESP32 dan Teknologi IoT*		menunjukkan efisiensi data akuisisi yang cocok untuk AQI.
10	L.Klibanov, PaulBoldt	*Preliminary Analysis of Bosch BME688 4-in-1 Environmental Sensor with AI*	Artikel Ilmiah	Mengkaji keunggulan sensor BME688 yang mendukung AI dan deteksi empat parameter lingkungan, mendukung keandalan sistem AQI.
11	Hidayat	*Perancangan Dan Implementasi Sistem Monitoring Air Quality Index (AQI) Berbasis Sensor BME688 Dengan Visualisasi Data P5 Dislpay Di Area Pelayanan Publik*	Skripsi	Mengkaji keunggulan sensor BME688 dalam deteksi empat parameter lingkungan yang dapat menjadi acuan AQI dalam pembuatan alat pendeteksi kualitas udara.

Kesimpulan Kajian

Berdasarkan studi terdahulu, dapat disimpulkan bahwa:

- Mikrokontroler ESP32 terbukti menjadi pilihan yang populer dan efektif dalam pengembangan sistem pemantauan kualitas udara karena keunggulannya dalam pemrosesan data dan konektivitas.
- Berbagai sensor gas seperti MQ2, MQ135, dan terbaru BME688, telah digunakan untuk mendeteksi parameter lingkungan dengan akurasi yang memadai.
- Sistem pemantauan kualitas udara telah diaplikasikan baik di ruang publik, industri, hingga ruang kelas, menunjukkan potensi luas penerapannya.

2.11. Kerangka Berpikir



BAB 3 PERANCANGAN SISTEM DAN IMPLEMENTASI SISTEM

3.1. Metode Penelitian

Penelitian ini menggunakan pendekatan ”Kuantitatif” dengan tujuan untuk merancang dan menguji alat pengukur kualitas udara berbasis sensor yang dapat memberikan data akurat mengenai parameter udara seperti kadar CO_2, VOC, suhu, dan kelembaban. Pendekatan ini dipilih karena memungkinkan pengumpulan dan analisis data numerik untuk mengukur kinerja alat secara objektif.

3.1.1. Jenis dan Pendekatan Penelitian

Penelitian ini termasuk dalam kategori penelitian rekayasa (engineering research) dengan pendekatan Research and Development (R&D). Fokus utama dari penelitian ini adalah merancang, membangun, dan mengimplementasikan sistem AQI (Air Quality Monitoring System) yang mampu memantau kualitas udara secara langsung menggunakan mikrokontroler ESP32 dan sensor lingkungan canggih dengan tampilan data melalui Display panel P5 dengan penyimpanan data pada micro SD. [11]

3.1.2. Manfaat Pendekatan Penelitian

Pendekatan penelitian Research and Development (R&D) memberikan sejumlah manfaat yang signifikan, terutama dalam pengembangan sistem teknologi seperti AQI untuk pemantauan kualitas udara. Beberapa manfaat tersebut antara lain:

1. Menghasilkan Produk yang Relevan dan Aplikatif

   Metode R&D memungkinkan peneliti untuk menghasilkan sebuah produk atau sistem yang langsung menjawab permasalahan di lapangan. Dalam penelitian ini, sistem AQI dikembangkan secara nyata untuk memenuhi kebutuhan monitoring kualitas udara di tempat pelayanan masyarakat.

2. Menggabungkan Teori dan Praktik Secara Langsung

   Pendekatan ini mengintegrasikan pemahaman teoritis yang diperoleh dari studi literatur dengan penerapannya dalam pengembangan sistem. Hal ini mendorong

peneliti untuk tidak hanya memahami konsep, tetapi juga mampu mewujudkannya dalam bentuk aplikasi nyata.

3. Mendorong Inovasi dan Kreativitas

   Melalui proses perancangan, pengujian, dan penyempurnaan, pendekatan R&D mendorong peneliti untuk berpikir kritis dan kreatif dalam menciptakan solusi baru yang efektif dan efisien.

4. Evaluasi Produk Secara Langsung

   Dalam metode ini, produk yang dikembangkan diuji secara langsung di lapangan, sehingga kelebihan dan kekurangannya dapat diketahui secara objektif. Hal ini memungkinkan adanya perbaikan yang berkelanjutan sebelum produk benar-benar digunakan secara luas.

5. Kontribusi Praktis terhadap Masyarakat

   Sistem yang dihasilkan tidak hanya menjadi hasil akademik semata, tetapi juga memiliki manfaat praktis bagi masyarakat, khususnya dalam meningkatkan kesadaran akan pentingnya kualitas udara dan menyediakan data yang dapat digunakan dalam pengambilan keputusan.

6. Meningkatkan Kapasitas Peneliti sebagai Problem Solver

   Dengan terlibat langsung dalam proses pengembangan teknologi, peneliti dilatih untuk menjadi pemecah masalah yang mampu melihat tantangan secara menyeluruh dan menawarkan solusi berbasis teknologi.

3.1.3. Tahapan Metode Penelitian

Metodologi yang digunakan ini terdiri dari beberapa tahapan utama sebagai berikut:

1. Identifikasi Masalah

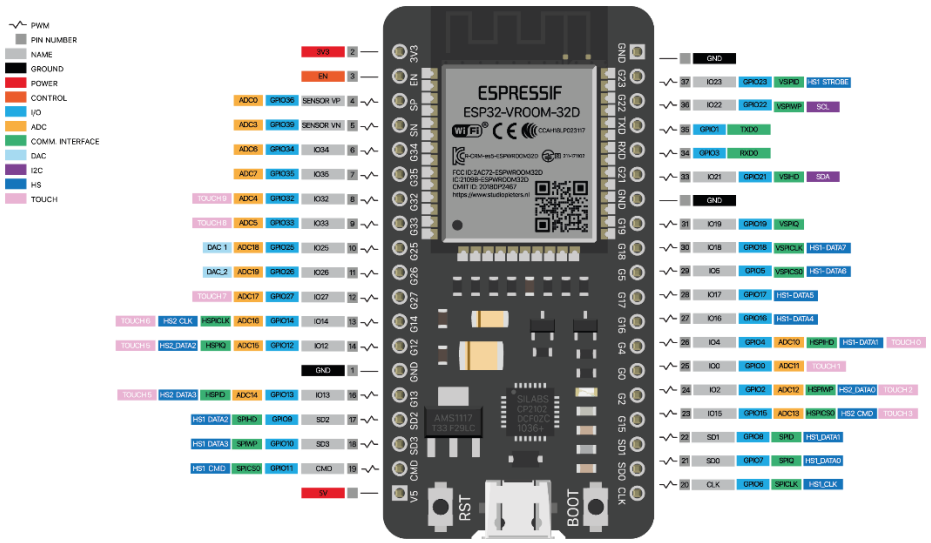
Langkah awal adalah mengidentifikasi permasalahan di lingkungan pelayanan masyarakat, yaitu belum adanya sistem monitoring kualitas udara yang mampu

memberikan informasi secara langsung kepada pengguna maupun pengelola fasilitas demi meningkatkan kesdaran masnyarakat dengan bahaya pencemaran udara bagi lingkungan, kesehatan, maupun ekonomi.[12]

2. Studi Literatur

Studi dilakukan untuk memahami fungsi dan datasheet dari setiap komponen yang akan digunakan pada penelitian kali ini, meliputi beberapa komponen inti sebagai berikut:

A. Mikrokontroler ESP32



Gambar 3.1. Datasheet ESP32

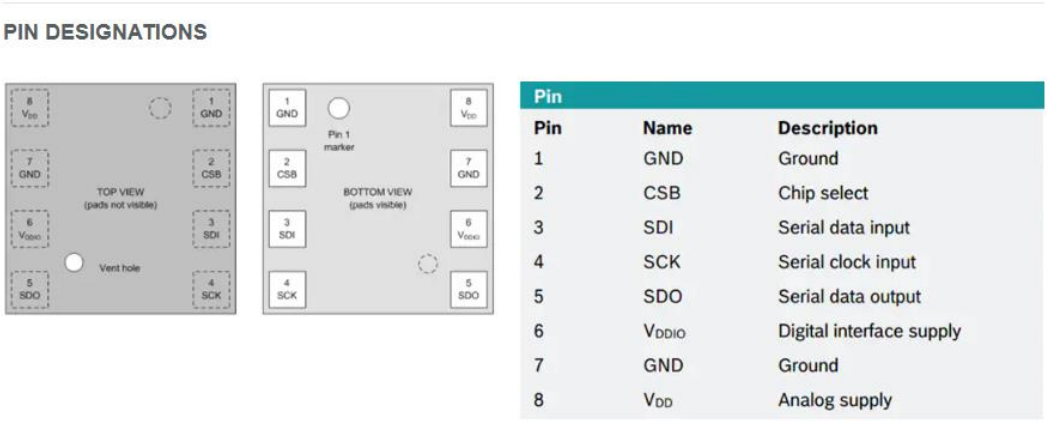
Sumber: https://mischianti.org/esp32-nodemcu-32s-esp-32s-kit-high-resolution-pinout-datasheet-and-specs/

Penjelasan Umum Fungsi Pin:

- Power Pins
    - 3V3: Output 3.3V (juga bisa digunakan sebagai input).
    - GND: Ground (tanah).
    - EN: Enable pin, digunakan untuk mengaktifkan chip.
- GPIO (General Purpose Input Output)
    - ESP32 memiliki lebih dari 30 GPIO yang dapat dikonfigurasi sebagai input, output, PWM, I²C, SPI, dll.
    - GPIO34–GPIO39 hanya input (tidak bisa output).

    - Beberapa pin memiliki fungsi khusus seperti ADC (analog), DAC, Touch Sensor, dsb.
- UART (Serial Communication)
    - TX0 (GPIO1), RX0 (GPIO3): Default serial (UART0).
    - UART1 dan UART2 juga tersedia dan dapat dipetakan ke GPIO lain.
- I2C
    - Biasanya digunakan: SDA: GPIO21 dan SCL: GPIO22
- SPI
    - Default SPI pins: MOSI: GPIO23, MISO: GPIO19, SCLK: GPIO18, CS: GPIO5
- PWM (Pulse Width Modulation)
    - Hampir semua GPIO bisa digunakan sebagai PWM.
- ADC (Analog to Digital Converter)
    - Tersedia pada GPIO32 hingga GPIO39 (ADC1 dan ADC2).
    - ADC2 tidak bisa digunakan saat WiFi aktif.
- DAC (Digital to Analog Converter)
    - GPIO25 dan GPIO26 mendukung DAC output.
- Touch Sensor
    - Tersedia pada GPIO4, GPIO0, GPIO2, GPIO15, dsb.

B. Sensor BME688



PIN DESIGNATIONS

Pin	Name	Description
1	GND	Ground
2	CSB	Chip select
3	SDI	Serial data input
4	SCK	Serial clock input
5	SDO	Serial data output
6	V_{DDIO}	Digital interface supply
7	GND	Ground
8	V_{DD}	Analog supply

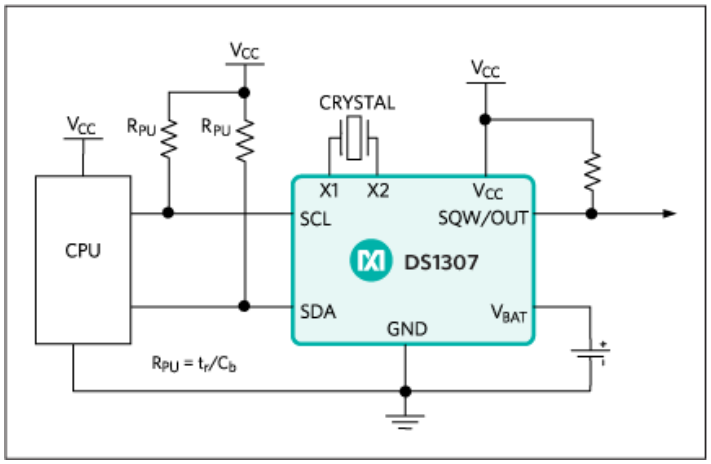
Gambar 3.2. Datasheet BME688

Sumber: https://www.mouser.co.id/new/bosch/bosch-bme688-ai-gas-sensor/ \

BME688 (Sensor Kualitas Udara)

- Komunikasi: I2C
- Pin SDA: GPIO21 (ESP32)
- Pin SCL: GPIO22 (ESP32)
- Fungsi: Mengukur suhu, kelembapan, tekanan udara, dan gas (VOC, CO, H_2).

C. Real-Time Clock (RTC DS1307)



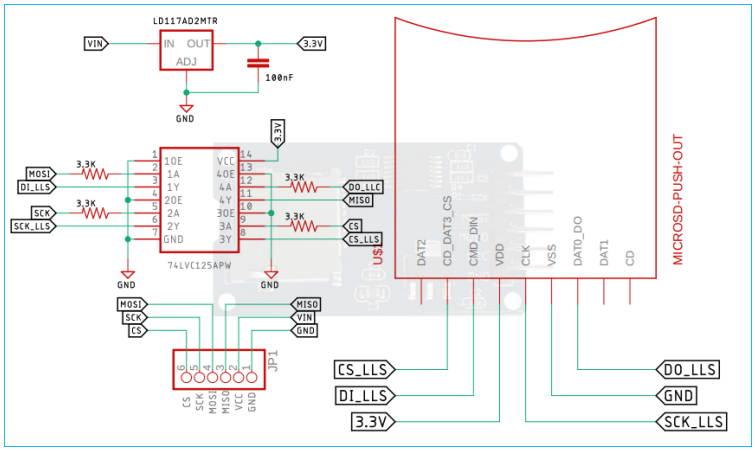
Gambar 3.3. Datasheet RTC

Sumber; https://www.analog.com/en/products/ds1307.html

RTC DS1307 (Modul Jam Real-Time)

- Komunikasi: I2C
- Pin SDA: GPIO21 (ESP32)
- Pin SCL: GPIO22 (ESP32)
- Fungsi: Menyediakan waktu dan tanggal akurat secara real-time, bahkan saat ESP32 mati.

D. Modul Kartu SD



Gambar 3.4. Datasheet Module MikroSD

Sumber: https://pauzan.com/interface-arduino-dengan-micro-sd/#google_vignette

Modul SD Card

- Komunikasi: SPI
- MOSI: GPIO23
- MISO: GPIO19
- SCK: GPIO18
- CS (Chip Select): GPIO5
- Fungsi: Menyimpan data sensor secara lokal.

E. Panel Display P5

■ **Interface Definition**

				Pin	Signal Definition	Function
R1	1	2	G1	1. 2. 3	R1.G1.B1	1 Group RGB Data
B1	3	4	GND	5. 6. 7	R2.G2.B1	2 Group RGB Data
R2	5	6	G2	9.10.11.12	A.B.C.D	Scan(Row) Control Signal
B2	7	8	GND	4. 8. 16	GND	Signal Ground
A	9	10	B	13	CLK	Clock Signal
C	11	12	D	14	LAT	Latch Signal
CLK	13	14	LAT	15	OE	Output Enable Signal
ROE	15	16	GND			

Gambar 3.5. Datasheet Panel P5

Sumber: https://id.aliexpress.com/item/1005004577111028.html

Panel P5 (HUB75 RGB LED Display)

- Komunikasi: GPIO Digital (paralel)
- Pin yang umum digunakan: GPIO12, 13, 14, 15, 16, 17, 25, 26, 27, 32, 33
- Fungsi: Menampilkan data kualitas udara secara real-time.

3. Analisis Kebutuhan Sistem

Analisis ini mencakup penentuan kebutuhan perangkat keras seperti alat dan bahan yang dibutuhkan pada saat penelitianini yang meliputi:

A. Alat dan Bahan

Dalam proses perancangan sistem pemantauan kualitas udara diperlukan sejumlah komponen elektronik dan perangkat lunak pendukung yang saling terintegrasi untuk menunjang kinerja alat secara optimal. Pemilihan alat dan bahan ini disesuaikan dengan kebutuhan sistem, spesifikasi teknis, serta parameter lingkungan yang akan diukur. Setiap komponen yang digunakan memiliki fungsi spesifik dan berperan penting dalam keberhasilan perancangan alat, baik dari sisi pembacaan data, pemrosesan informasi, penyimpanan, hingga sistem peringatan. Pemahaman terhadap fungsi dan karakteristik masing-masing alat dan bahan sangat diperlukan agar perancangan berjalan sesuai rencana dan hasil yang diperoleh dapat diandalkan.[13]

Tabel berikut menunjukkan alat dan bahan yang digunakan dalam penelitian ini:

Tabel 3.1. Alat yang Digunakan dalam Perancangan

No.	Nama Alat	Fungsi
1	Solder 80W	Untuk menyambung komponen elektronik ke PCB
2	Alat penyedot timah	Menghilangkan timah saat kesalahan penyolderan
3	Cutter	Memotong kabel dan komponen kecil
4	Pembersih mata solder	Membersihkan ujung solder dari sisa timah
5	Penjepit PCB	Menahan PCB agar stabil saat penyolderan
6	Tang capit	Menjepit kabel dan komponen
7	Tang pemotong	Memotong kaki komponen setelah dipasang di PCB
8	Pinset	Mengambil dan menempatkan komponen kecil
9	Obeng kecil	Memasang atau melepas sekrup pada casing dan sambungan kabel
10	Multimeter	Mengukur tegangan, arus, dan resistansi pada rangkaian
11	Laptop + Platform.io	Untuk memprogram mikrokontroler ESP32
12	Glue gun (opsional)	Menempelkan komponen di dalam kotak pelindung (black box)

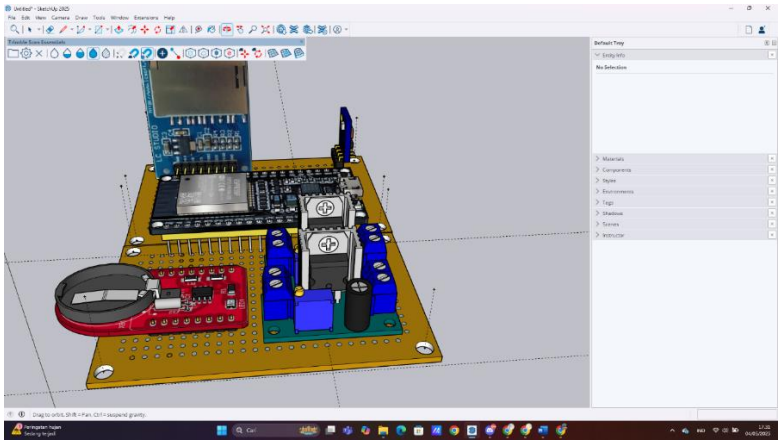
Tabel 3.2. Bahan yang Digunakan dalam Perancangan

No.	Nama Bahan	Fungsi
1	ESP32	Mikrokontroler utama pengendali system
2	Sensor BME688	Mendeteksi suhu, kelembapan, tekanan, dan gas VOC
3	RTC DS1307	Menyediakan waktu dan tanggal untuk pencatatan data

No.	Nama Bahan	Fungsi
4	SD Card + SD Card Module	Menyimpan data log hasil pemantauan
5	Panel LED P5	Menampilkan informasi kualitas udara secara real-time
6	Step-down module 12V to 5V	Mengubah tegangan input 12V menjadi 5V untuk ESP32 dan sensor
7	Power supply 12V	Sumber utama tegangan untuk seluruh sistem
8	PCB dual-layer 4x6 cm	Tempat merakit semua komponen elektronik
9	Timah solder	Media penyambung komponen ke jalur PCB
10	Header dan pin header	Konektor yang bisa dilepas pasang untuk ESP32 dan modul lainnya
11	Saklar push button	Tombol kontrol manual (jika diperlukan)
12	Transistor NPN	Mengendalikan aktifasi beban seperti kipas dan buzzer
13	Buzzer	Memberi peringatan suara saat udara buruk
14	Fan 3x3 cm	Menggerakkan udara kedalam alat agar dapat mendapatkan data yang lebih optimal
15	Elco 10µF	Menstabilkan tegangan dan mengurangi noise di rangkaian
16	Kapasitor keramik 100nF	Filter sinyal frekuensi tinggi
17	Resistor 10kΩ	Membatasi arus dalam sirkuit
18	Dioda	Melindungi rangkaian dari arus balik
19	Kabel jumper	Menghubungkan antar modul pada breadboard atau PCB
20	Kabel catu daya + soket 2 lubang	Mengalirkan arus dari adaptor ke system

No.	Nama Bahan	Fungsi
21	Black Box	Pelindung fisik semua komponen
22	Sekrup dan baut kecil	Mengunci komponen di dalam black box

4. Perancangan Sistem



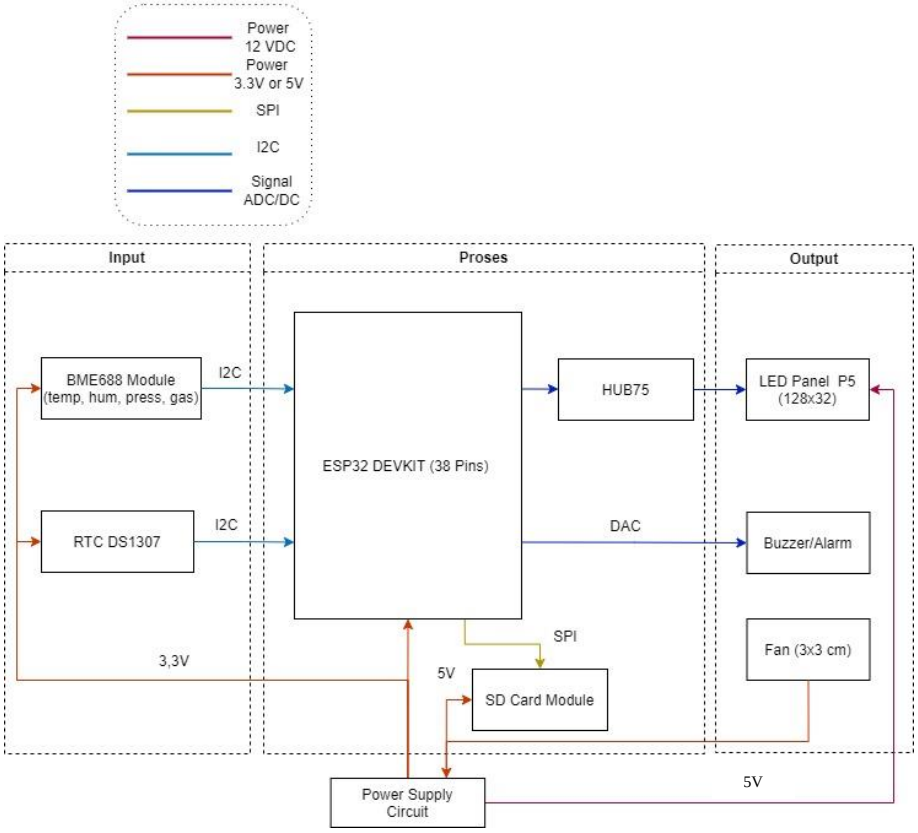
Gambar 3.6. Desain 3D komponen

Sumber: Primer

Tabel 3.3. Konfigurasi pin pada rangkaian

BME688	RTC DS1307	MICRO SD MODUL	PANEL LED P5	BUZZER	GPIO ESP32
3.3 VDC	3.3 VDC				3.3 VDC
GND	GND	GND	GND	GND	GND
		5VDC			5 VDC
SDA	SDA				GPIO 21
SCL	SCL				GPIO 22
		CS			GPIO 5
		SCK			GPIO 18
		MOSI			GPIO 23
		MISO			GPIO 19
			R1		GPIO 17
			R2		GPIO 12
			B1		GPIO 13
			B2		GPIO 14
			G1		GPIO 16
			G2		GPIO 4
			A		GPIO 27
			B		GPIO 0
			C		GPIO 26
			D		GPIO 2
			LAT		GPIO 15
			CLK		GPIO 25
			OE		GPIO 33
				PIN BUZZER	GPIO 32

Perancangan meliputi desain dan konfigurasi jalur yang akan digunakan pada alat ini. Selain itu dalam perancangan ini melipuiti pembutan dari blok diagram sistem yang akan digunakan sebagai panduan fungsi dari sistem sebagai berikut:



Gambar 3.7. Diagram Blok Sistem

Sumber: Primer

Diagram tersebut memperlihatkan bagaimana sensor BME688 sebagai sensor utama dari alat ini membaca parameter udara (suhu, kelembaban, tekanan, gas VOC), lalu data akan dikirim ke mikrokontroler ESP32. Kemudian waktu akan dicatat menggunakan RTC DS1307 serta data kemudian akan disimpan di SD card. Informasi hasil pengukuran ditampilkan pada Panel P5 setiap 2 detik. Jika nilai kualitas udara berada di atas ambang batas, buzzer akan berbunyi sebagai sistem peringatan.[14]

Untuk alur sistem nya sendiri dapat diartikan sebagai berikut:

1. Input (Sensor dan waktu)

• BME688 Module: Mengukur faktor lingkungan seperti suhu, kelembapan, tekanan udara, dan gas. Informasi ini ditransmisikan ke mikrokontroler ESP32 melalui I2C.

• RTC DS1307: Modul jam yang menunjukkan waktu dan tanggal dengan tepat, juga mengomunikasikan informasi ini ke ESP32 via I2C.

• Kedua modul ini memperoleh daya 3.3V dari Sirkuit Power Supply.

• Kipas (3×3 cm): Berfungsi menarik udara menuju sensor. Tujuannya adalah untuk memastikan bahwa aliran udara ke sensor BME688 lebih stabil dan merata, sehingga hasil pengujian kualitas udara menjadi lebih akurat dan dapat diandalkan.

2. Proses (Pengolahan Data oleh ESP32)

• ESP32 Devkit (38 Pin) berperan sebagai otak sistem yang:

a. Menerima informasi sensor dari BME688 serta waktu dari RTC.
b. Menyimpan data yang diukur ke Modul SD Card melalui komunikasi SPI.
c. Mengirim informasi tampilan ke HUB75 untuk ditampilkan pada Panel LED P5.
d. Mengelola pengaktifan Buzzer/Alarm melalui sinyal dari jalur DAC.

3. Output (Tampilan dan Respons)

• LED Panel P5 (128×32): Menyajikan data kualitas udara secara langsung.

• Buzzer/Alarm: Memberikan sinyal peringatan jika kualitas udara melebihi batas yang telah ditentukan.

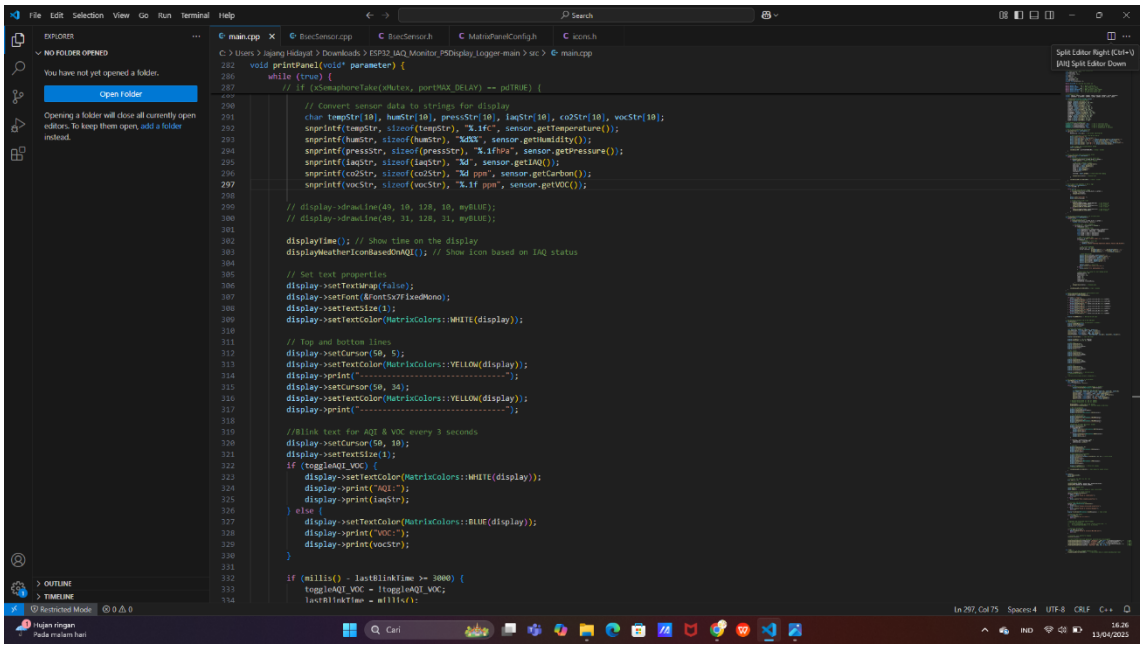
4. Power Supply

• Power Supply Circuit: Menyediakan arus yang diperlukan agar sistem dapat beroperasi dengan baik.

5. Implementasi Sistem

Setelah desain selesai, sistem dirakit dan diprogram menggunakan Platform.io. berikut alur tahapan pemerograman yang dilakukan:

A. Implementasi Perangkat Lunak



Gambar 3.8. Proses pembutan Program

Sumber: Primer

Program utama sistem ini menggunakan bahasa C++ dengan deplpmen melalui platform.io. adapun berikut adalah alur logika dari program yang berfungsi pada ESP32 sebagai mikrokontroler:

Program ini dirancang untuk berintegrasi dengan komponen-komponen sebagai berikut:

- ESP32 sebagai mikrokontroler utama.
- Sensor BME688 sebagai sensor utama yang akan berfungsi untuk mengukur nilai dari IAQ
- RTC DS1307 untuk pencatatan waktu.
- Panel LED matrix untuk menampilkan data secara nyata.

- Buzzer sebagai alarm jika kualitas udara buruk.
- module SD Card untuk menyimpan data.

1. Langkah pertama adalah menyiapkan library yang dibutuhkan seperti:

- Wire, SPI, SD untuk komunikasi.
- RTClib, BsecSensor untuk sensor dan waktu.
- ESP32-HUB75-MatrixPanel-I2S-DMA untuk layar LED.
- freertos/task.h untuk multitasking dengan FreeRTOS.

2. Kemudian melakukan inisialisasi pada Objek dan Variabel yang akan di proses oleh program

Di awal kode hal yang perlu kita lakukan adalah :

- Membuat objek untuk display matrix, RTC, sensor BME680, dan File SD.
- Membuat mutex (xMutex) untuk menghindari konflik antar task saat mengakses data sensor.
- Menyimpan data kumulatif (suhu, tekanan, IAQ, dll.) untuk dihitung rata-ratanya sebelum disimpan.

3. Lalu setelah itu kita harus mendefinisian Task (Tugas Paralel)

ESP32 mendukung multitasking (berkat FreeRTOS), sehingga beberapa proses berjalan secara bersamaan dengan pembacaan sebagai berikut:

- readSensorTask

  Membaca data dari sensor setiap 2 detik dan menambahkannya ke variabel akumulasi. Mutex digunakan untuk menghindari konflik akses data.

- printDataTask

  Mencetak waktu dan data sensor ke Serial Monitor untuk debugging atau pemantauan manual.

- AlarmTask

  Memonitor nilai IAQ terbaru. Jika IAQ > 150 (kualitas udara buruk), buzzer akan berbunyi setiap detik sebagai peringatan.

- SaveDataTask

  Menyimpan data rata-rata dari pembacaan sensor setiap 1 jam ke kartu SD dalam format CSV. Menghitung rata-rata dari akumulasi dan menuliskannya ke file.

- PrintPanel

  Menampilkan data ke panel LED, termasuk waktu, tanggal, suhu, kelembapan, CO2, VOC, dan ikon kualitas udara yang berubah warna tergantung status.

4. Adapun beberapa fungsi Tambahan

Beberapa fungsi tambahan berikut mendukung tampilan dan logika pada tampilan display P5:

- InitColors() untuk mengatur warna RGB yang digunakan di panel LED.
- DisplayWeatherIconBasedOnAQI() menampilkan ikon berdasarkan status IAQ.
- DisplayTime() menampilkan jam dan tanggal saat ini pada LED.
- Logika berkedip untuk pergantian tampilan AQI ↔ VOC setiap 3 detik.

5. Kemudian dilakukan penyestrukturan Sinkronisasi dan Pengamanan

Karena banyak task mengakses variabel yang sama, digunakan:

- XSemaphoreTake() dan xSemaphoreGive(): mencegah task mengakses data secara bersamaan dan menyebabkan konflik (race condition).

6. terakhir dilakukan integrasi

Setelah semua task didefinisikan, pada fungsi setup(), akan dilakukan:

- Inisialisasi sensor, RTC, SD card, buzzer PWM, dan LED matrix.

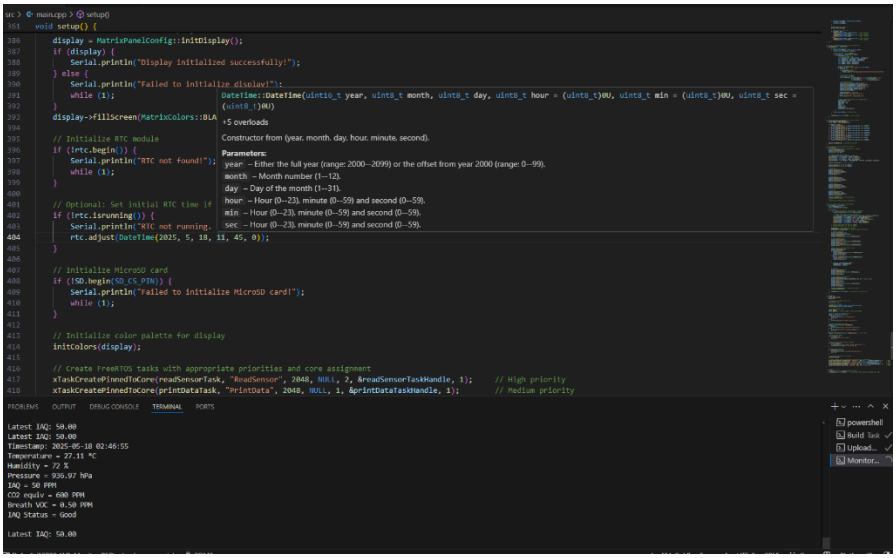
- Membuat task-task dengan xTaskCreatePinnedToCore() agar dapat berjalan secara paralel di inti (core) prosesor ESP32.

Kesimpulan yang dapat kita ketahui dari tahapan proses program di atas alat ini merupakan sistem pemantauan kualitas udara secara real-time dengan spesifikasi:

- Membaca sensor lingkungan secara berkala.
- Menampilkan data dan status ke panel LED.
- Menyimpan data ke kartu SD untuk keperluan dokumentasi.
- Mengaktifkan alarm jika kualitas udara berbahaya.
- Menggunakan FreeRTOS multitasking agar semua proses dapat berjalan efisien dan bersamaan.

Setelah program selesai dilakukanlah tahapan pengujian, pengujian awal dilakukan dengan mengaktifkan sensor secara berkala, mencatat data pada kartu SD, dan menampilkan informasi kualitas udara pada panel LED P5.

6. Pengujian Sistem



Gambar 3.9. Hasil Pengujian Pada Terminal Monitor

Sumber: Primer

Pengujian dilaksanakan guna memastikan bahwa setiap bagian beroperasi dengan baik serta untuk mengecek sejauh mana sensor peka saat terjadi perubahan pada lingkungan dan untuk memastikan semua sistem berfungsi dengan optimal.

7. Evaluasi dan Analisis Hasil

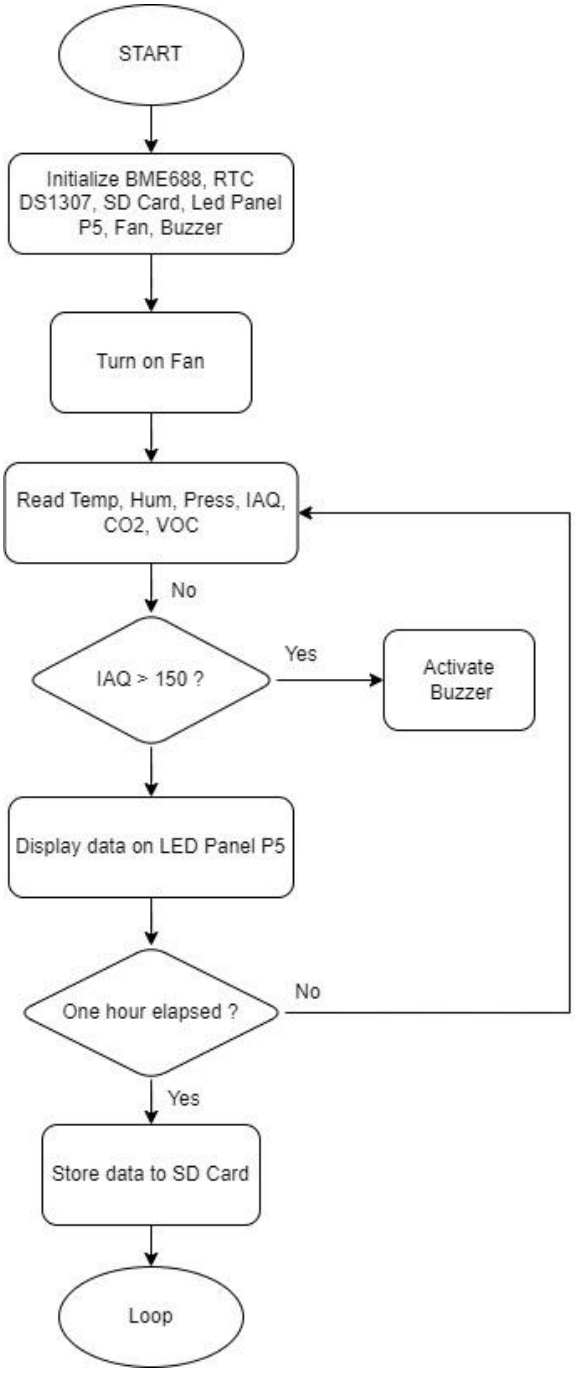
Data dari uji coba dianalisis untuk mengukur kinerja sensor, akurasi dalam pencatatan waktu, efektivitas penyajian data, dan kestabilan dari keseluruhan sistem.

3.1.3. Lokasi dan Waktu Penelitian

Penelitian dilakukan di Universitas Islam Nusantara selama periode 3 hari, yang dimana hal ini mencakup tahap pengumpulan data, perakitan sistem, dan pengujian dengan peroses pengujian alat pada 10 tempat yang berbeda seperti rumah, sekolah, lapangan terbuka, tempat penggilingan padi, dll. Hal ini bertujuan untuk meningkatkan tingkat akurasi dari alat yang di uji di berbagai tempat.

3.2. Flowchart sistem

Tahapan ini merupakan suatu pendekatan yang secara terperinci menjelaskan seluruh prosedur yang terlibat dalam pembuatan program yang bertujuan untuk melakukan pengukuran kualitas udara. Proses ini melibatkan peneliti dalam mengumpulkan data yang relevan dan signifikan untuk kemudian diolah menjadi sebuah informasi yang dapat digunakan untuk pengklasifikasian tingkat kualitas udara. Seluruh langkah-langkah dalam proses ini, sebagaimana diilustrasikan dalam Gambar 3.9 yang terdapat dalam flowchart penelitian berikut:



Gambar 3.10. Flowchart Sistem

1. Mulai

Pada tahapan awal ini dimulai ketika sistem dinyalakan. maka Mikrokontroler ESP32 mulai menjalankan proses utama setelah mendapatkan suplai daya dari catu daya.

Sistem masuk dalam mode aktif dan siap menginisialisasi semua perangkat pendukung. Tahap ini sangat penting karena menandai transisi dari kondisi diam (idle) ke operasional (aktif) yang memungkinkan sistem mulai bekerja secara otomatis.

2. Inisialisasi Komponen (BME688, RTC DS1307, SD Card, LED Panel P5, Kipas, dan Buzzer)

Setelah dinyalakan, sistem akan langsung melakukan proses inisialisasi terhadap seluruh perangkat keras yang terhubung dengan rangkaian.

- Sensor BME688 diaktifkan untuk mulai membaca parameter lingkungan seperti suhu, kelembapan, indeks kualitas udara (IAQ), CO_2, dan gas VOC.
- RTC DS1307 dikonfigurasi untuk mencatat waktu dan tanggal secara real-time, yang diperlukan untuk log data ke kartu SD.
- Modul SD Card dicek untuk memastikan bahwa penyimpanan tersedia dan dapat digunakan.
- Panel LED P5 diaktifkan sebagai sarana tampilan informasi.
- Buzzer disiapkan dalam mode siaga agar dapat diaktifkan sesuai dengan kondisi kualitas udara yang terbaca. Tahap ini memastikan semua perangkat siap beroperasi secara sinkron.

3. Menyalakan fan (kipas)

Setelah sistem dihidupkan, fan(kipas) akan menyala untuk membantu menyalurkan sirkulasi udara di sekitar sensor yang digunakan. Aliran udara yang relatif stabil sangat penting agar sensor BME688 dapat membaca data kualitas udara secara akurat. Tanpa sirkulasi yang baik, sensor berpotensi menerima data yang tidak mencerminkan kondisi udara sebenarnya.

4. Pembacaan Parameter Lingkungan disekitar

Sistem kemudian membaca berbagai parameter lingkungan sekitar menggunakan sensor BME688. Parameter yang diukur meliputi:

- Suhu (temperature) – untuk mengetahui panas/dingin udara.
- Kelembapan (humidity) – untuk mengetahui tingkat kelembapan udara.
- Indeks kualitas udara (IAQ) – indikator gabungan berdasarkan gas VOC dan kondisi lingkungan.
- Konsentrasi CO_2 dan VOC – sebagai faktor utama penentu polusi udara. Data ini dibaca secara periodik dan menjadi dasar pengambilan keputusan sistem.

5. Pembacaan Nilai Parameter IAQ

Data yang diperoleh kemudian akan dianalisis, terutama untuk nilai IAQ (Indeks Kualitas Udara). Sistem menggunakan ambang batas IAQ > 150 sebagai indikator kualitas udara yang tergolong tidak sehat. Ambang ini didasarkan pada standar umum di mana nilai IAQ tinggi menunjukkan konsentrasi polutan udara yang membahayakan kesehatan.

6. Menyalakan Buzzer sebagai respon pada tingkat kualitas udara yang buruk

Jika nilai IAQ lebih dari 150, maka sistem secara otomatis mengaktifkan buzzer sebagai sinyal peringatan bagi pengguna. Buzzer berfungsi sebagai alat notifikasi berbunyi yang menandakan bahwa udara di sekitar sistem tidak layak untuk dihirup dalam jangka panjang. Ini memberikan informasi real-time kepada pengguna agar segera mengambil tindakan, seperti membuka ventilasi atau meninggalkan area.

7. Visualisasi Hasil Pengukuran pada Panel Display P5

Setelah proses analisis data selesai, sistem secara otomatis menampilkan hasil pengukuran dari sensor pada panel display P5. Informasi yang ditampilkan mencakup parameter suhu, kelembapan, dan nilai Indeks Kualitas Udara (IAQ). Desain tampilan ini dirancang untuk memudahkan pengguna dalam memahami kondisi kualitas udara secara langsung, tanpa memerlukan akses ke aplikasi atau perangkat tambahan lainnya.

8. Penjadwalan Penyimpanan Data Setiap Satu Jam

Sistem dilengkapi dengan mekanisme pemantauan waktu yang terintegrasi dengan modul RTC DS1307 guna memastikan proses penyimpanan data berlangsung secara tepat dan terjadwal. Data hasil pengukuran akan disimpan ke dalam kartu SD setiap satu jam sekali. Apabila interval waktu belum terpenuhi, sistem akan menunda proses penyimpanan dan kembali melanjutkan siklus pembacaan sensor hingga waktu yang ditentukan tercapai.

9. Penyimpanan Informasi ke Kartu SD

Ketika periode penyimpanan telah selesai (setiap 60 menit), sistem akan mengalihkan seluruh informasi dari sensor ke dalam modul kartu SD. Informasi tersebut akan direkam dalam format file (csv) yang mencakup waktu, suhu, kelembapan, tekanan, IAQ, CO_2, dan VOC. Tujuan dari proses ini adalah untuk melakukan dokumentasi, melakukan analisis jangka panjang, serta memenuhi kebutuhan pelaporan.

10. Siklus (Pengulangan)

Setelah proses penyimpanan informasi selesai atau jika waktu satu jam belum tercapai, sistem akan kembali ke langkah awal dalam membaca sensor. Proses ini akan terus berlangsung selama perangkat mendapatkan suplai daya dan berfungsi dengan baik.

BAB 4 PEMBAHASAN DAN ANALISIS

4.1. Gambaran Umum Perancangan

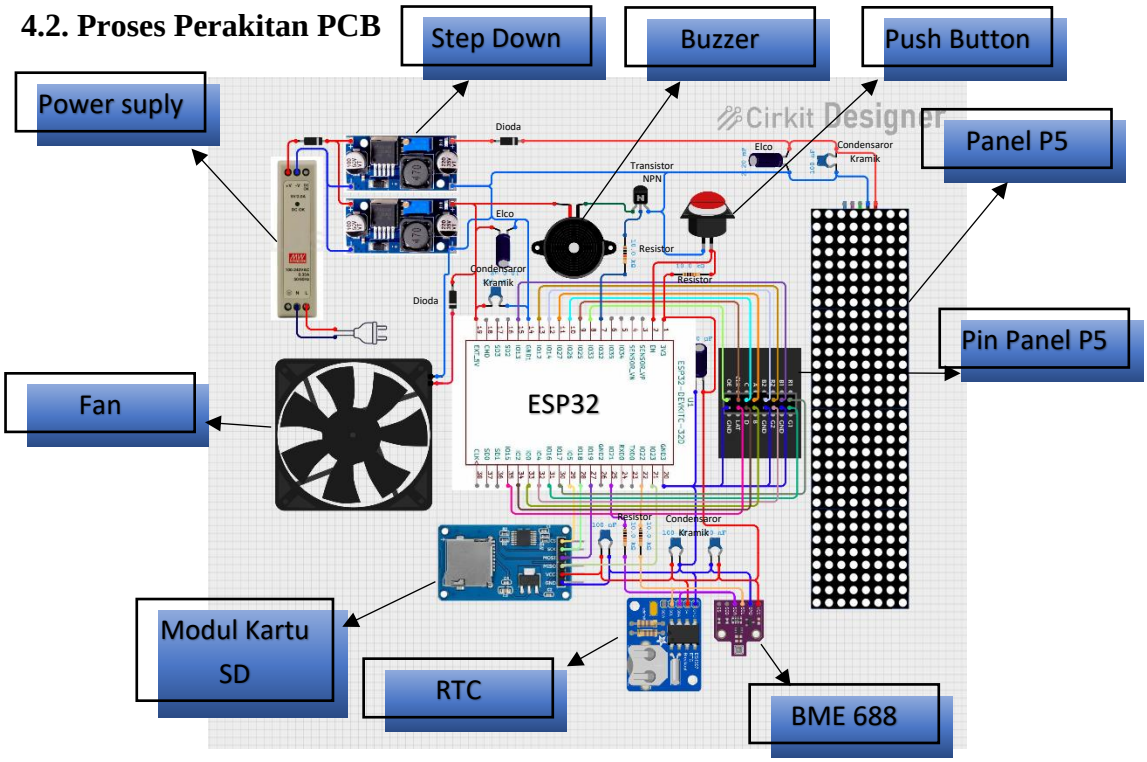
Pada sistem yang telah dirancang terdapat performa yang perlu diuji, yaitu kemampuan pembacaan sensor BME688 terhadap parameter suhu, kelembaban, dan resistansi gas untuk mendeteksi tingkat kualitas udara. Sistem ini dirancang untuk digunakan di area pelayanan publik sebagai sistem pemantauan kualitas udara (AQI) yang mampu menampilkan data secara langsung, dan menyimpan hasil data secara langsung kedalam kartu SD, serta memberikan peringatan dini apabila kualitas udara memburuk.

Komponen utama dalam sistem meliputi sensor BME688 sebagai pendeteksi lingkungan, RTC DS3107 untuk pencatatan waktu real-time, SD Card module sebagai media penyimpanan data, ESP32 sebagai mikrokontroler pusat, panel LED P5 untuk tampilan visual, serta buzzer sebagai indikator kondisi darurat.

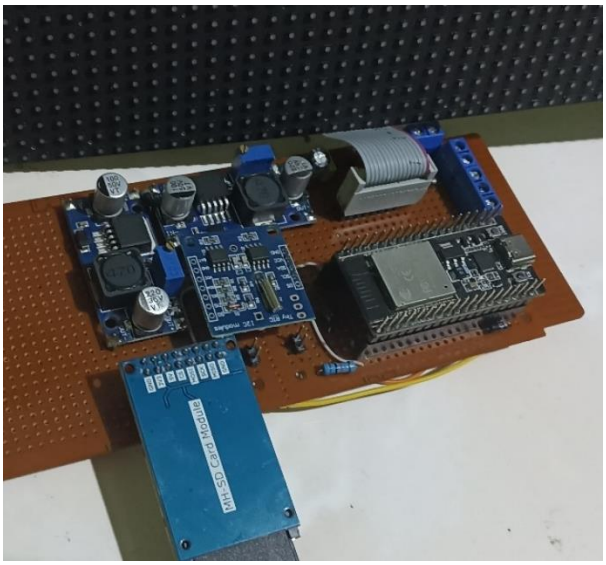
Setiap komponen dikonfigurasi dengan protokol komunikasi yang sesuai, seperti I2C untuk sensor dan RTC, serta SPI untuk modul penyimpanan. Penyusunan rangkaian mengacu pada datasheet masing-masing komponen, memastikan tegangan kerja sesuai, koneksi pin akurat, dan tidak terjadi gangguan kelistrikan.

Perangkat lunak mikrokontroler ESP32 dikembangkan untuk membaca data lingkungan dari BME688, mencatat waktu pengukuran dari RTC, menyimpan hasil ke kartu SD, mengaktifkan buzzer saat ambang batas gas terlampaui, serta menampilkan data ke panel LED secara langsung.

4.2. Proses Perakitan PCB



Gambar 4.1. Jalur rangkaian PCB



Gambar 4.2. Hasil Rangkaian pada PCB

Proses perakitan dimulai dengan penyusunan tata letak komponen pada PCB dual-layer ukuran 4x6 cm. Penyusunan dilakukan dengan mempertimbangkan efisiensi jalur sinyal dan kestabilan distribusi daya. Langkah-langkah perakitan adalah sebagai berikut:

1. Pemasangan Modul Catu Daya
    - Catu daya 12V AC-DC (adaptor) dipasang terlebih dahulu dan dihubungkan ke modul step-down regulator (buck converter) untuk mengatur tegangan ke 5V dan 3.3V.
    - Output 5V dialirkan langsung ke kipas, modul LED matrix, buzzer, dan ESP32. Tegangan 3.3V digunakan untuk sensor BME688 dan RTC DS1307.
2. Pemasangan Mikrokontroler ESP32
    - Modul ESP32 Devkit V1 (38 pin) dipasang di tengah-tengah sebagai pusat kendali. Pin header dipasang agar ESP32 dapat dilepas-pasang dengan mudah dari rangkaian PCB jika perlu dilakukan pemrograman ulang.
3. Pemasangan Sensor dan Modul Pendukung
    - Sensor BME688 dan RTC DS1307 dihubungkan ke pin I2C ESP32 (SDA dan SCL). Keduanya mendapatkan daya dari jalur 3.3V.
    - Modul SD Card dihubungkan ke jalur SPI ESP32 untuk penyimpanan data.
    - Semua koneksi data dan daya disolder ke jalur PCB sesuai urutan dan arah aliran data.
4. Pemasangan Output
    - LED Matrix (Panel P5) dihubungkan ke pin digital ESP32 melalui HUB75 dengan soket konektor. Jalur data dan power dipisah untuk menghindari interferensi.

- Buzzer/Alarm disolder ke jalur DAC dari ESP32, dengan saklar sebagai tombol reset untuk ESP32.
- Kipas 3x3 cm dihubungkan ke jalur power 5V sebagai penarik udara ke arah sensor.

5. Perakitan Tombol dan Saklar
    - Push button digunakan sebagai tombol reset/manual input, disambungkan ke salah satu pin GPIO dan ground dengan resistor pull-down.
    - Saklar pengaman juga disertakan di sisi input power utama.
6. Penyusunan Komponen ke dalam Box
    - Semua komponen yang telah dirakit di atas papan PCB dimasukkan ke dalam kotak pelindung (black box).
    - Bagian depan kotak diberi lubang untuk jalur udara (dekat sensor dan kipas), serta jendela tampilan untuk LED matrix.
7. Pengujian dan Finalisasi
    - Setelah semua komponen terpasang, dilakukan pengujian rangkaian untuk memastikan setiap komponen bekerja sesuai fungsinya.
    - Penyesuaian perangkat lunak dilakukan untuk sinkronisasi pembacaan sensor, penyimpanan data, dan tampilan.

4.3. Rancangan Pengujian

Pengujian system dilakukan dengan pengujian pada sensitivitas sensor BME688, dimana pengujian ini dilakukan di Lokasi pembuatan alat pendeteksi kualitas udara di Kawasan solokanjeruk, kabupaten Bandung dan tempat tempat lain sebgagai lokasi pengujian. Pengujian dilakukan dengan melihat bagaimana sensor BME688 ini dapat mengukur Tingkat kualitas udara di sekitar Kawasan tersebut dengan mula-mula pengetesan awal pada sensor dilakukan dengan cara memberikan input berupa gas yang di dapat dari gas korek api untuk melihat bagaimana sensitivitas dari sensor pada saat diberikan input

tersebu. Selain itu pengujian selanjutnya dilakukan dengan membiarkan alat menyala selama 3 hari untuk melihat bagaimana kinerja dari alat pendeteksi kualitas udara tersebut.[15]

4.4. Hasil Pengujian

Gambar 4.3. Dokumentasi Alat

Setelah semua komponen dirasa berfungsi dengan normal, dilakukanlah pengujian untuk mengetahuai berapa nilai yang di dapat dari hasil pengukuran alat pendeteksi kualitas udara ini. Berikut adalah table hasil pengukuran dari alat pengukur kualitas udara yang dihasilkan dari hasil pengukuran selama 1jam pada tanggal 28 mei 2025 di universitas Isalam Nusantara:

Tabel 4.1. Perolehan Data Rata-Rata Selama 1 Jam (13:36-14:40)

Parameter	Rata-rata Selama 1 Jam
Suhu (°C)	27,13
Kelembaban (%)	77,9
Tekanan (hPa)	936,32
IAQ	51,3
CO2 (ppm)	610,3
VOC (ppm)	0,52

Catatan: Nilai rata-rata di atas dihitung dengan mengakumulasi seluruh data pada rentang waktu 1 jam, lalu dibagi dengan jumlah data yang tersedia untuk masing-masing parameter pada periode tersebut.

Dari data di atas kita hapat menarik beberapa Kesimpulan diantara lain :

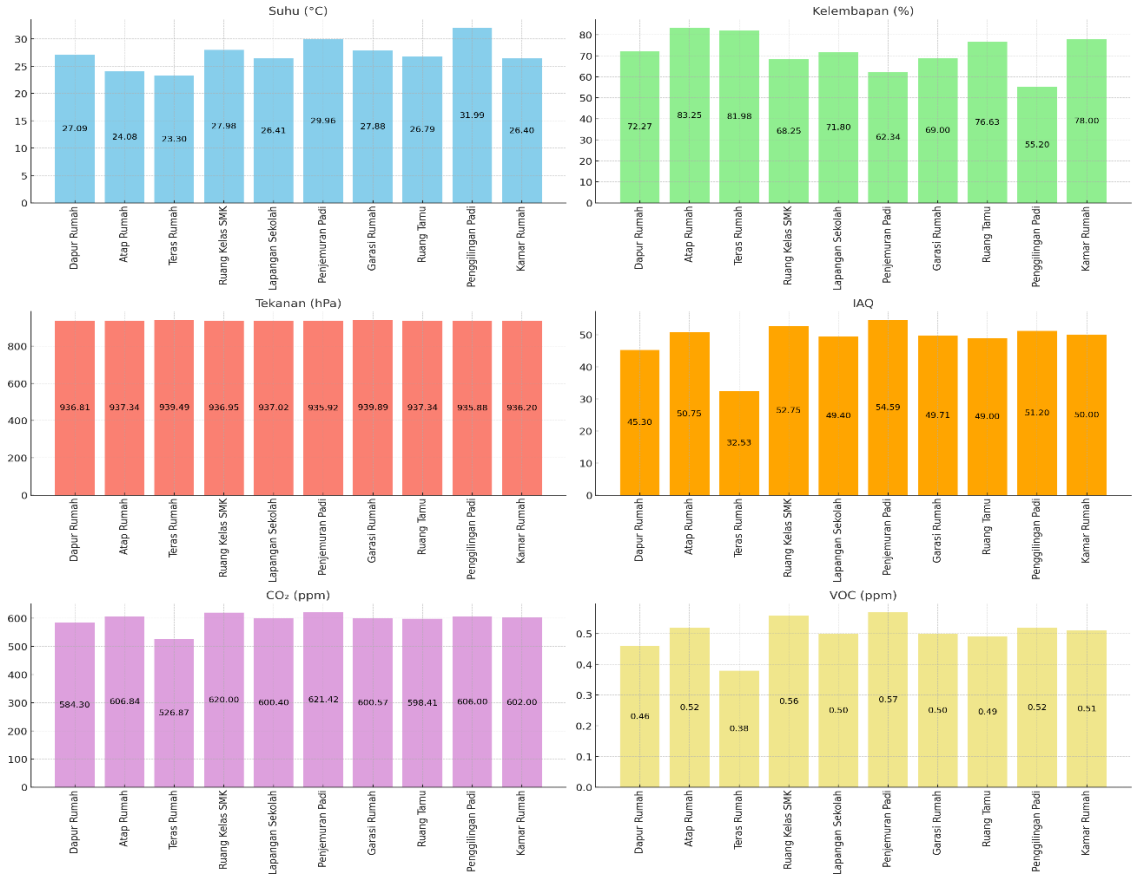
- Suhu rata-rata yang di dapat selama 1 jam berada di kisaran 27°C, menunjukkan kondisi ruangan yang relatif stabil.
- Kelembaban pada tempat pengujian cukup tinggi, mendekati 78%, yang umum pada lingkungan tropis.
- IAQ (Indoor Air Quality) dan CO_2 menunjukkan kualitas udara dalam ruangan yang masih dalam batas wajar untuk aktivitas normal.
- VOC (Volatile Organic Compounds) berada pada level sangat rendah, menandakan minimnya polutan organik di udara.

Oleh karena itu dari hasil pengujian di atas kita dapat mengetahuai bahwa pada saat pengambilan data, nilai kualitas udara berada pada tingkat yang cukup normal sebagaimana ditunjukan oleh tabel 2.3 sehingga menandakan pada jam dan lokasi pengambilan data tidak ada indeks yang menunjukan terjadinya pencemaran udara yang parah pada lokasi tersebut.[16]

Tabel 4.2. Data Hasil Pengujian Pada 10 Tempat Dengan Rentang Waktu 30 Menit

Tempat	tanggal/ Waktu	Suhu (°C)	Kelembapan (%)	Tekanan (hPa)	IAQ	CO_2 (ppm)	VOC (ppm)
Dapur Rumah	03-06-25/17:50	27.09	72.27	936.81	45.30	584.30	0.46
Atap Rumash	03-06-25/18:40	24.08	83.25	937.34	50.75	606.84	0.52
Teras Rumah	03-05-25/22:54	23.30	81.98	939.49	32.53	526.87	0.38
Ruang Kelas SMK Pasundan Rancaekek	04-06-25/16:00	27.98	68.25	936.95	52.75	620.00	0.56
Lapangan Sekolah	04-06-25/16:00	26.41	71.80	937.02	49.40	600.40	0.50

Tempat	tanggal/ Waktu	Suhu (°C)	Kelembapan (%)	Tekanan (hPa)	IAQ	CO_2 (ppm)	VOC (ppm)
Lapangan Penjemuran Padi	05-06-25/15:30	29.96	62.34	935.92	54.59	621.42	0.57
Garasi Rumah	03-06-25/23:23	27.88	69.00	939.89	49.71	600.57	0.50
Ruang Tamu Rumah	05-06-25/11:40	26.79	76.63	937.34	49.00	598.41	0.49
Tempat Penggilingan Padi	05-06-25/14:40	31.99	55.20	935.88	51.20	606.00	0.52
Kamar Rumah	03-06-25/17:17	26.40	78.00	936.20	50.00	602.00	0.51



Gambar 4.4. Grafik Perbandingan Data Pada 10 Tempat Yang Berbeda

Selain melakukan ujicoba pada 10 tempat yang berbeda dilakukan juga ujicoba pada tempat yang dinilai memiliki potensi nilai AQI yang tinggi seperti tempat pembakaran sampah dengan nialai sebagai berikut :

Tabel 4.3. Hasil Pengujian Pada Tempat Pembakaran Sampah

Tanggal/Waktu	Suhu	Kelembapan	Tekanan	IAQ	CO_2	VOC
28/06/2025 11.12	26.49	78.00	937.64	168.00	1685.00	7.48
28/06/2025 11.13	25.32	78.00	937.52	171.00	1718.00	8.31
28/06/2025 11.14	25.19	78.00	937.47	153.00	1535.00	5.46
28/06/2025 11.15	25.69	75.00	937.22	163.00	1642.00	6.68
28/06/2025 11.16	25.74	75.00	937.12	153.00	1537.00	5.28
28/06/2025 11.17	25.53	76.00	936.09	156.00	1572.00	5.80
28/06/2025 11.18	25.61	75.00	936.06	153.00	1535.00	5.29
28/06/2025 11.19	25.73	74.00	935.92	152.00	1533.00	5.23
28/06/2025 11.20	26.01	74.00	935.91	178.00	1794.00	9.73
28/06/2025 11.21	26.27	73.00	935.93	183.00	1839.00	10.51
28/06/2025 11.22	26.40	73.00	935.91	159.00	1600.00	6.16
28/06/2025 11.23	26.88	73.00	935.62	154.00	1546.00	5.37
28/06/2025 11.24	27.02	73.00	935.62	158.00	1593.00	5.99
28/06/2025 11.25	27.16	73.00	935.64	161.00	1624.00	6.40
28/06/2025 11.26	27.21	72.00	935.68	162.00	1624.00	6.40
28/06/2025 11.27	27.18	72.00	935.69	162.00	1630.00	6.49
28/06/2025 11.28	27.13	72.00	935.68	164.00	1646.00	6.73
28/06/2025 11.29	27.08	72.00	935.67	164.00	1651.00	6.82
28/06/2025 11.30	27.03	72.00	935.66	164.00	1652.00	6.83
28/06/2025 11.31	26.98	72.00	935.64	164.00	1652.00	6.83
28/06/2025 11.32	26.93	72.00	935.64	162.00	1633.00	6.54
28/06/2025 11.33	26.90	72.00	935.63	163.00	1634.00	6.55
28/06/2025 11.34	26.86	72.00	935.61	164.00	1654.00	6.86
28/06/2025 11.35	26.82	72.00	935.60	167.00	1678.00	7.24
28/06/2025 11.36	26.78	72.00	935.59	167.00	1679.00	7.26
28/06/2025 11.37	26.75	72.00	935.58	165.00	1656.00	6.89
28/06/2025 11.38	26.72	73.00	935.58	165.00	1656.00	6.89
28/06/2025 11.39	26.73	73.00	935.57	167.00	1677.00	7.22
28/06/2025 11.40	26.74	73.00	935.56	168.00	1691.00	7.46
28/06/2025 11.41	26.72	73.00	935.56	167.00	1680.00	7.27

4.5. Hasil analisis

Seperti yang telah dijelaskan pada bagian 1.3, sistem ini masih berada pada tahap prototipe awal dan perancangan, sehingga data yang diperoleh masih berupa data awal tanpa kalibrasi lanjutan. Berdasarkan hasil pengujian, sensor BME688 membutuhkan waktu pemanasan sekitar 30.000 ms atau sekitar 5 menit agar resistansinya mencapai nilai yang lebih akurat. Namun, untuk pembacaan gas yang lebih spesifik masih belum dapat dilakukan karena memerlukan riset, pengambilan sampel, dan kalibrasi yang lebih mendalam.[17]

Perlu diketahui bahwa untuk mendapatkan nilai sensor yang akurat, terutama dalam pembacaan gas, dibutuhkan algoritma yang cukup canggih. Hal ini dikarenakan berbagai macam gas yang menguap saat terjadi pembakaran hutan, seperti CO, CO2, metana, dan gas lainnya, memiliki karakteristik yang berbeda-beda. Saat ini, sensor hanya mampu mendeteksi perubahan resistansi gas secara umum yang dapat dipengaruhi oleh berbagai jenis gas secara drastis. Oleh karena itu, kalibrasi lebih lanjut sangat diperlukan untuk meningkatkan akurasi pembacaan gas.

Secara garis besar:

- Sistem prototipe saat ini sudah berfungsi sesuai tujuan awal.
- Komponen dan algoritma dapat diubah atau ditingkatkan untuk kebutuhan yang lebih kompleks.
- Algoritma fuzzy dapat membantu mengolah data sensor dengan ketidakpastian dan variabilitas, meningkatkan akurasi dan respons sistem.
- BME AI Studio memungkinkan pelatihan khusus sensor BME688 untuk mendeteksi berbagai gas dengan sensitivitas dan selektivitas yang disesuaikan.

Dengan demikian, pengembangan lebih lanjut dengan integrasi AI dan algoritma fuzzy sangat direkomendasikan untuk mendapatkan hasil yang lebih maksimal dan akurat sesuai kebutuhan.

untuk tingkat akurasi Sensor BME688 memiliki tingkat akurasi berdasarkan datasheet resmi Bosch Sensortec dan beberapa sumber terkait:

- Akurasi Temperatur: ±0,5 °C ada pada rentang 0 hingga 65 °C.
- Akurasi Kelembaban: ±3% Relative Humidity (RH) pada rentang 20%–80% RH, dengan hysteresis ±1,5% RH.
- Akurasi Tekanan: ±0,6 hPa pada rentang 300–1100 hPa dan suhu 0–65 °C, dengan noise sekitar 0,12 hPa.

- Akurasi Indeks Kualitas Udara (IAQ): Sensor memberikan resolusi IAQ 1 point dengan rentang 0–500, deviasi antar sensor ±15%, dan drift ±1%–4% tergantung konsentrasi gas.
- Deteksi Gas VOC dan Gas Lain: Sensor dapat mendeteksi VOC, gas sulfur volatil, karbon monoksida, hidrogen, dan gas lain dalam konsentrasi bagian per miliar (ppb), dengan algoritma AI untuk meningkatkan akurasi dan selektivitas.

Tabel 4.4. Ringkasan Akurasi Utama BME688

Parameter	Akurasi / Toleransi	Rentang Operasi
Temperatur	±0,5 °C	-40 °C hingga +85 °C
Kelembaban	±3% RH (20–80% RH)	0% hingga 100% RH
Tekanan	±0,6 hPa	300 hPa hingga 1100 hPa
IAQ	Deviasi antar sensor ±15%, drift ±1–4%	IAQ 0–500
Gas VOC	Deteksi di kisaran ppb dengan AI	—

sehingga dapat di simpulkan bahwa sensor ini memiliki tingkat akurasi yang baik untuk digunakan sebagai sensor pengukur tingkat kualitas udara dengan sfesifikasi yang memadai dan mudah di kembangkan.

Dari hasil analisis tersebut didapati datasheet dari hasil pengambilan data selama 1 hari tepatnya pada tanggal 25 Mei 2025 pada ruang tamu di sebuah rumah dikecamatan solokan jeruk sebagaimana berikut:

Tabel 4.5. Rekap Rata-rata data per 1 Jam pada tanggal 25 Mei 2025

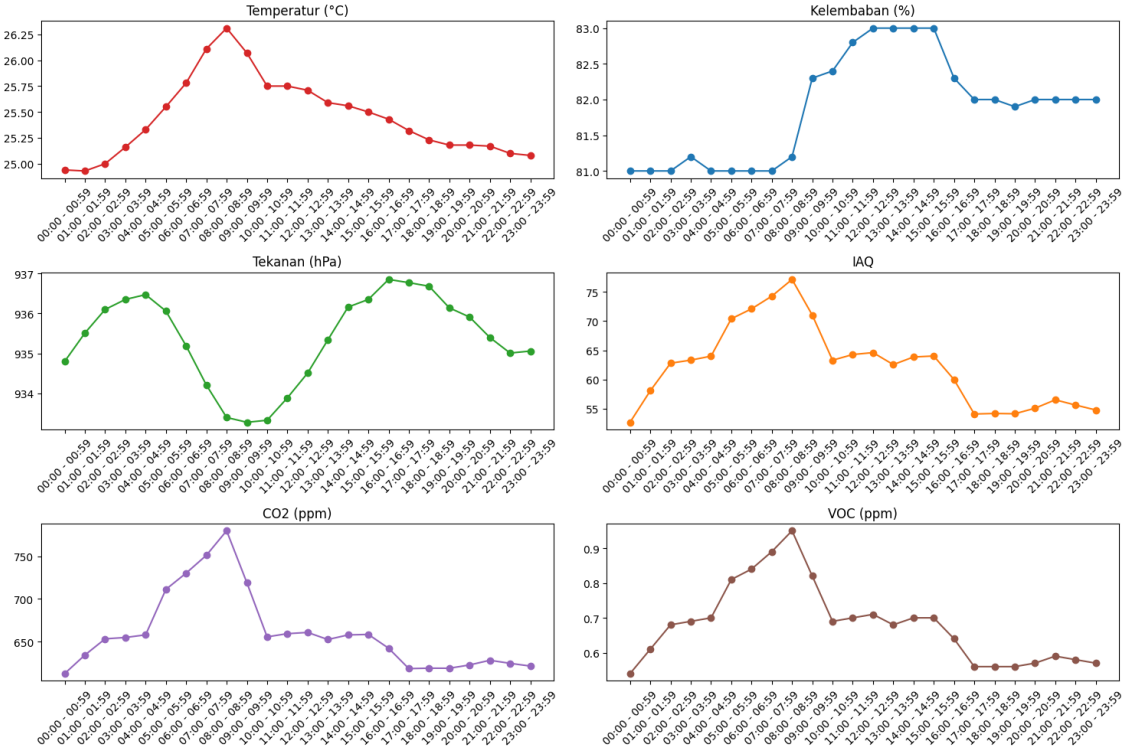
Jam	Suhu (°C)	Kelembaban (%)	Tekanan (hPa)	IAQ	CO2 (ppm)	VOC (ppm)	Keterangan
00:00	24.94	81.0	934.80	52.64	612.78	0.54	Baik

Jam	Suhu (°C)	Kelembaban (%)	Tekanan (hPa)	IAQ	CO2 (ppm)	VOC (ppm)	Keterangan
01:00	24.93	81.0	935.51	58.05	634.44	0.61	Baik
02:00	25.00	81.0	936.10	62.78	653.39	0.68	Baik
03:00	25.16	81.2	936.35	63.31	654.97	0.69	Baik
04:00	25.33	81.0	936.47	63.99	658.04	0.70	Baik
05:00	25.55	81.0	936.06	70.41	711.19	0.81	Baik
06:00	25.78	81.0	935.19	72.11	730.21	0.84	Baik
07:00	26.11	81.0	934.21	74.29	751.06	0.89	Baik
08:00	26.31	81.2	933.40	77.16	779.73	0.95	Baik
09:00	26.07	82.3	933.28	71.04	718.98	0.82	Baik
10:00	25.75	82.4	933.33	63.29	655.71	0.69	Baik
11:00	25.75	82.8	933.89	64.27	659.45	0.70	Baik
12:00	25.71	83.0	934.51	64.60	660.87	0.71	Baik
13:00	25.59	83.0	935.34	62.54	652.60	0.68	Baik
14:00	25.56	83.0	936.16	63.85	657.98	0.70	Baik
15:00	25.50	83.0	936.35	64.02	658.54	0.70	Baik
16:00	25.43	82.3	936.85	59.97	642.42	0.64	Baik
17:00	25.32	82.0	936.77	54.06	618.57	0.56	Baik
18:00	25.23	82.0	936.68	54.15	619.08	0.56	Baik

Jam	Suhu (°C)	Kelembaban (%)	Tekanan (hPa)	IAQ	CO2 (ppm)	VOC (ppm)	Keterangan
19:00	25.18	81.9	936.14	54.10	619.02	0.56	Baik
20:00	25.18	82.0	935.91	55.05	622.88	0.57	Baik
21:00	25.17	82.0	935.40	56.49	628.32	0.59	Baik
22:00	25.10	82.0	935.01	55.60	624.77	0.58	Baik
23:00	25.08	82.0	935.06	54.75	621.55	0.57	Baik

Dari hasil data tersebut dapat ditarik beberapa Kesimpulan Data Kualitas Udara yang di peroleh dari alat pendeteksi kualitas udara tersebut:

1. Suhu (Temperature) yang di peroleh cenderung meningkat secara bertahap dari dini hari hingga siang hari, hal ini mungkin dipengaruhi oleh curah hujan yang cukup intens sehingga suhu mencapai puncak sekitar pukul 08:00–09:00 (sekitar 26.3°C) sebelum hujan turun dan pada saat jam sibuk aktivitas, lalu perlahan menurun hingga malam hari karena cuaca.
2. Kelembaban (Humidity) stabil di kisaran 81–83%, sedikit meningkat pada siang hingga sore hari, kemudian stabil kembali di malam hari.
3. Kualitas Udara (IAQ) memburuk seiring meningkatnya aktivitas manusia di pagi hingga siang hari (nilai IAQ naik hingga ~77), lalu membaik di malam hari.
4. Konsentrasi CO2 dan VOC juga meningkat pada pagi hingga siang hari, menunjukkan adanya aktivitas manusia seperti penggunaan alat elektronik, kendaraan, atau aktivitas industri ringan. Nilai tertinggi CO2 terjadi sekitar pukul 08:00, dan menurun perlahan setelah pukul 12:00.



Gambar 4.5. Grafik Data Kualitas Udara Pada Tanggal 25 Mei 2025

BAB 5 KESIMPULAN DAN SARAN

5.1. Kesimpulan

Desain perancangan sistem monitoring Air Quality Index (AQI) berbasis sensor BME688 dengan visualisasi data P5 Display di area pelayanan publik yang telah dibuat dan diuji memberikan hasil yang cukup baik. Namun, masih banyak hal yang perlu diperbaiki dan diperhatikan ketika akan merancang sistem ini untuk pembacaan data yang lebih akurat. Pada bab ini dipaparkan pencapaian dari sistem beserta saran-saran untuk pengembangan agar dapat digunakan sesuai kebutuhan.

Dari hasil perancangan dan pengujian desain sistem yang telah dilakukan, dapat disimpulkan beberapa hal sebagai berikut:

1. Rancangan sistem monitoring AQI berbasis sensor BME688 dengan visualisasi data P5 Display telah berhasil dibuat dan diimplementasikan sebagai alat pemantau kualitas udara secara real-time di area pelayanan publik, sesuai dengan tujuan penelitian ini.
2. Sistem mampu mengukur beberapa parameter lingkungan seperti suhu, kelembapan, dan gas VOC secara cukup akurat serta menampilkan data tersebut secara langsung melalui panel display P5.
3. Akurasi pembacaan sensor BME688 cukup baik, namun untuk hasil yang lebih presisi terutama pada deteksi gas tertentu, diperlukan proses pemanasan sensor selama beberapa menit dan kalibrasi lanjutan secara berkala.
4. Fitur alarm berupa buzzer berjalan dengan baik sebagai peringatan dini jika kualitas udara memasuki kategori tidak sehat, sehingga dapat meningkatkan kewaspadaan masyarakat di area pelayanan publik.
5. Sistem telah dilengkapi dengan penyimpanan data otomatis menggunakan SD Card dan pencatatan waktu real-time dengan RTC DS1307, sehingga data historis dapat terdokumentasi dengan baik untuk analisis lebih lanjut.
6. Sistem ini aplikatif, efisien, dan dapat diandalkan sebagai solusi monitoring kualitas udara yang dapat diadopsi di berbagai lingkungan pelayanan publik.

5.2. Saran

Agar sistem monitoring AQI ini dapat lebih optimal dan bermanfaat di masa mendatang, berikut beberapa saran pengembangan:

1. Integrasi dengan teknologi Internet of Things (IoT): Sistem dapat dikembangkan dengan mengirimkan data secara online ke platform web atau aplikasi yang

digunakan oleh instansi terkait sehingga pemantauan dapat dilakukan secara jarak jauh dan multi-lokasi.

2. Penambahan jenis sensor: Untuk cakupan parameter yang lebih luas, dapat ditambahkan sensor lain seperti PM2.5, NO_2, atau O_3 agar hasil monitoring lebih komprehensif dan relevan dengan kebutuhan lingkungan.
3. Peningkatan antarmuka pengguna: Pengembangan aplikasi mobile atau web untuk visualisasi data yang lebih informatif dan mudah diakses oleh masyarakat umum maupun pihak terkait.
4. Kalibrasi dan pengujian lebih lanjut: Lakukan kalibrasi ulang secara berkala dan uji coba di berbagai kondisi lingkungan untuk memastikan keandalan dan akurasi alat dalam berbagai situasi nyata.
5. Kolaborasi dengan instansi terkait: Mendorong kerja sama dengan pemerintah daerah atau lembaga lingkungan hidup untuk pemanfaatan data sebagai dasar pengambilan kebijakan publik dan edukasi masyarakat tentang pentingnya kualitas udara.
6. Penambahan fitur otomatisasi: Seperti pengendalian ventilasi ruangan otomatis atau sistem notifikasi berbasis aplikasi jika terdeteksi kualitas udara buruk.
7. Mengkalibrasi alat dengan teknologi atau alat pengukur kualitas udara yang sudah terjamin kualitas dan akurasinya seperti TSI DustTrak™ DRX Aerosol Monitor atau Thermo Scientific™ Air Quality Monitors (seperti TEOM™ 1405).

Dengan pengembangan dan perbaikan berkelanjutan, diharapkan sistem monitoring AQI ini dapat menjadi solusi teknologi tepat guna yang berkontribusi nyata dalam menjaga kesehatan dan kualitas lingkungan di area pelayanan publik.

DAFTAR PUSTAKA

[1] A. P. Putro, D. A. Hidayat, F. F. Heratama, A. Dwi, D. E. Yulian, dan Y. A. Prabowo, "Sistem Monitoring Kualitas Udara Menggunakan Mikrokontroler ESP32 dengan Sensor MQ2 Berbasis Internet of Things".

[2] B. Harpad, S. Salmon, dan R. M. Saputra, "SISTEM MONITORING KUALITAS UDARA DI KAWASAN INDUSTRI DENGAN NODEMCU ESP32 BERBASIS IOT," J. Inform. Wicida, vol. 12, no. 2, hlm. 39–47, Jul 2022, doi: 10.46984/inf-wcd.1955.

[3] D. D. Akbar, "Pollution and Stunting (Case Study at DKI Jakarta)," Interact. J. Pendidik. Bhs., vol. 10, no. 2, hlm. 764–776, Okt 2023, doi: 10.36232/jurnalpendidikanbahasa.v10i2.4899.

[4] D. Putra Pratama, N. Nurchim, dan N. Faiq Muhammad, "Sistem Pemantauan Kualitas Udara IoT Untuk Mitigasi Risiko Pekerja dan Masyarakat di Industri Genteng : Studi Kasus Desa Talesan Purwantoro Wonogiri," Inf. J. Inform. Dan Sist. Inf., vol. 16, no. 2, hlm. 184–196, Okt 2024, doi: 10.37424/informasi.v16i2.310.

[5] F. K. Sari, "The Relationship Between Air Pollution Exposure And The Incidence Of Chronic Respiratory Disease: A Case Study In An Urban Area".

[6] Fredy Agung Dwi Cahyono dan Denny Irawan, "Rancang Bangun Sistem Pemantauan Dan Kontrol Kualitas Udara Dalam Ruangan Berbasis Iot," Elkom J. Elektron.

Dan Komput., vol. 17, no. 2, hlm. 468–476, Des 2024, doi: 10.51903/elkom.v17i2.2208.

[7] G. Syuhada dkk., “Impacts of Air Pollution on Health and Cost of Illness in Jakarta, Indonesia,” Int. J. Environ. Res. Public. Health, vol. 20, no. 4, hlm. 2916, Feb 2023, doi: 10.3390/ijerph20042916.

[8] K. Bandung, “Figure 3 · Potential gain in life expectancy from reducing PM2.5 from 2022 levels to the WHO guideline in 10 most populous provinces of Indonesia”.

[9] L. Klibanov, “Preliminary analysis of Bosch BME688 4-in-1 environmental sensor with AI”.

[10] M. Babiuch, P. Foltynek, dan P. Smutny, “Using the ESP32 Microcontroller for Data Processing,” dalam 2019 20th International Carpathian Control Conference (ICCC), Krakow-Wieliczka, Poland: IEEE, Mei 2019, hlm. 1–6. doi: 10.1109/CarpathianCC.2019.8765944.

[11] M. Rizal, A. Arifin, M. F. Rasyd, A. A. M. Suradi, dan A. Bahtiar, “Desain dan Implementasi Sistem Pemantauan Polusi Udara Berbasis Android Real-Time di SMKS Darul Ulum Layoa Bantaeng: Design and Implementation of A Real-Time Air Pollution Monitoring System Based on Android at SMKS Darul Ulum Layoa Bantaeng,” MALCOM Indones. J. Mach. Learn. Comput. Sci., vol. 3, no. 2, hlm. 143–152, Okt 2023, doi: 10.57152/malcom.v3i2.894.

[12] “Monitoring Kualitas Udara Ruangan dengan Sensor Debu dan Cahaya Menggunakan Metode Fuzzy Mamdani.pdf.”

[13] R. Fajar Nugraha, F. Nurul Husna, S. Sandi, A. Fairuz Syahla, Y. Aldi Saputra, dan R. Hidayat, “Smart Air Quality Guardian: Pengawasan Polusi Udara Berbasis ESP32 dengan Sensor Gas MQ-2 dan MQ-135,” J. Komput. Dan Elektro Sains, vol. 2, no. 2, hlm. 1–7, Jan 2024, doi: 10.58291/komets.v2i2.175.

[14] R. Hidayati, “SISTEM PEMANTAUAN KUALITAS UDARA SECARA REAL-TIME MENGGUNAKAN ESP32 DAN TEKNOLOGI IOT,” Djtechno J. Teknol. Inf., vol. 5, no. 2, hlm. 232–245, Agu 2024, doi: 10.46576/djtechno.v5i2.4619.

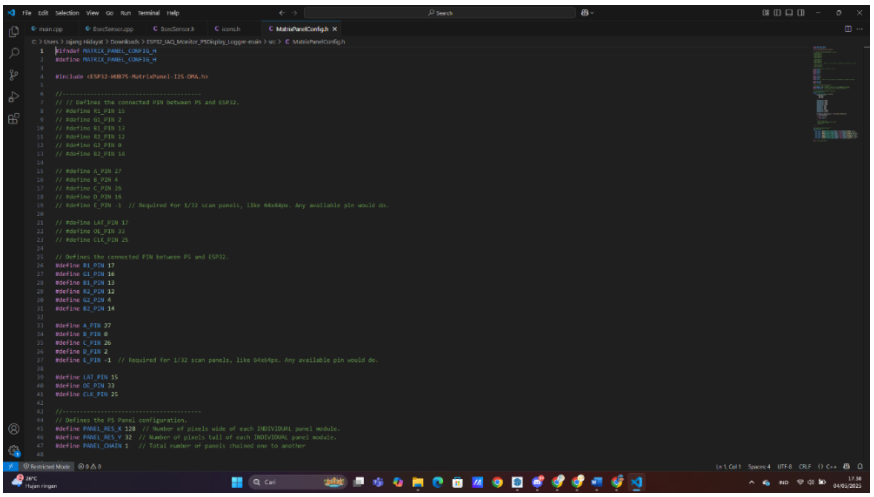
[15] V. C. Poekoel dan A. H. J. Ontowirjo, “Pembuatan Sistem Monitoring Kualitas Udara Dalam Ruangan”.

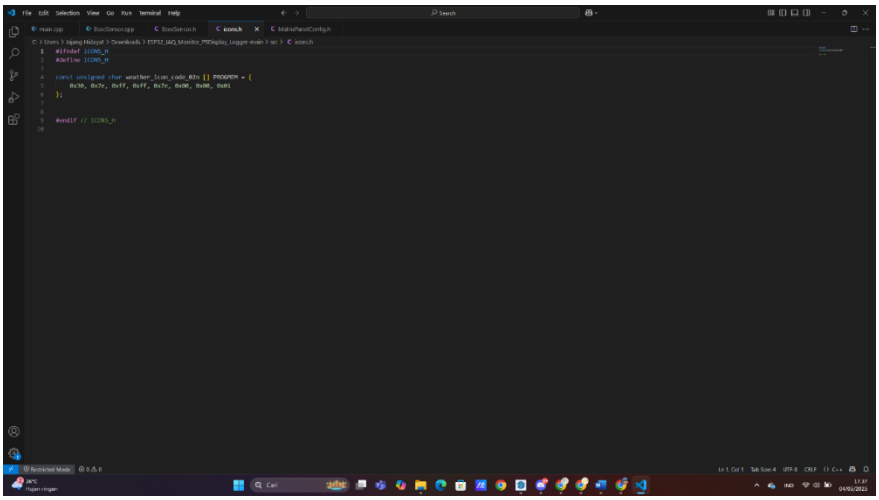
[16] Y. B. Sundaresan dan A. K. Jaiswal, “A Smart Multi-purpose Remote Controlled Digital Clock”.

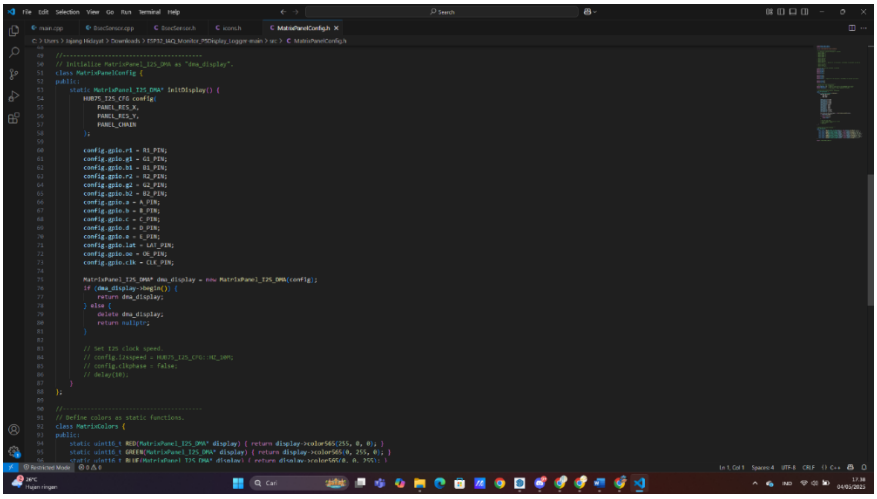
[17] Y. S. Susilo, J. Herawan, dan F. Parningotan, “Valuation of the Economic Impact of Air Pollution to Promote Public Welfare in Jakarta,” vol. 6, no. 2, 2024.

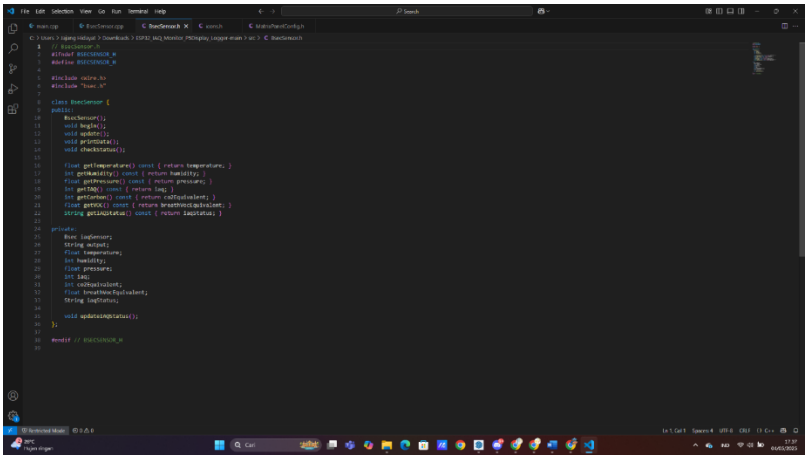
LAMPIRAN

Foto Pembuatan Program

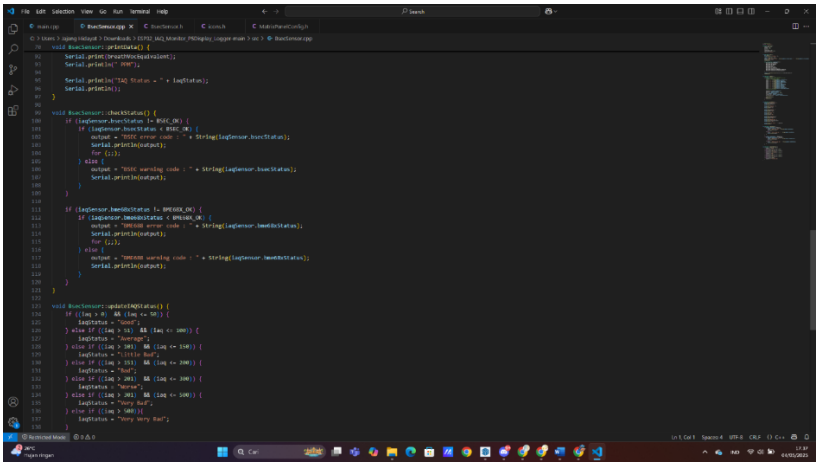




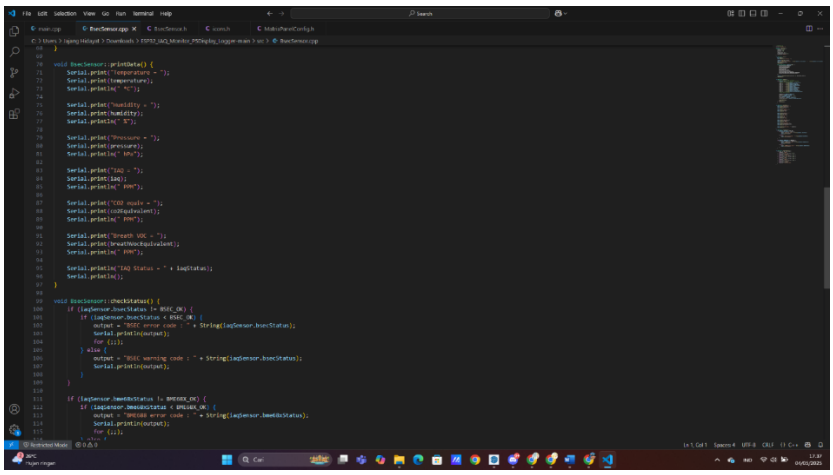




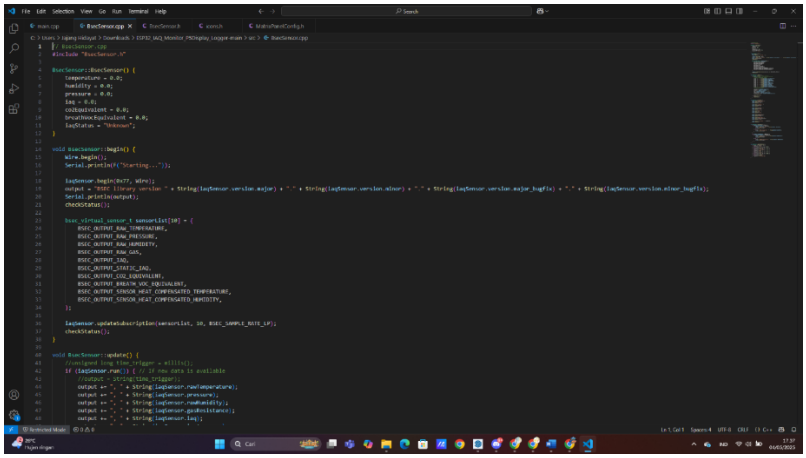
```
// BsecSensor.h
#ifndef BSECSENSOR_H
#define BSECSENSOR_H

#include <Wire.h>
#include "bsec.h"

class BsecSensor {
public:
    BsecSensor();
    void begin();
    void update();
    void printData();
    void checkStatus();

    float getTemperature() const { return temperature; }
    int getHumidity() const { return humidity; }
    float getPressure() const { return pressure; }
    int getIAQ() const { return iaq; }
    int getCarbon() const { return co2Equivalent; }
    float getVOC() const { return breathVocEquivalent; }
    String getIAQStatus() const { return iaqStatus; }

private:
    Bsec iaqSensor;
    String output;
    float temperature;
    int humidity;
    float pressure;
    int iaq;
    int co2Equivalent;
    float breathVocEquivalent;
    String iaqStatus;

    void updateIAQStatus();
};

#endif // BSECSENSOR_H
```

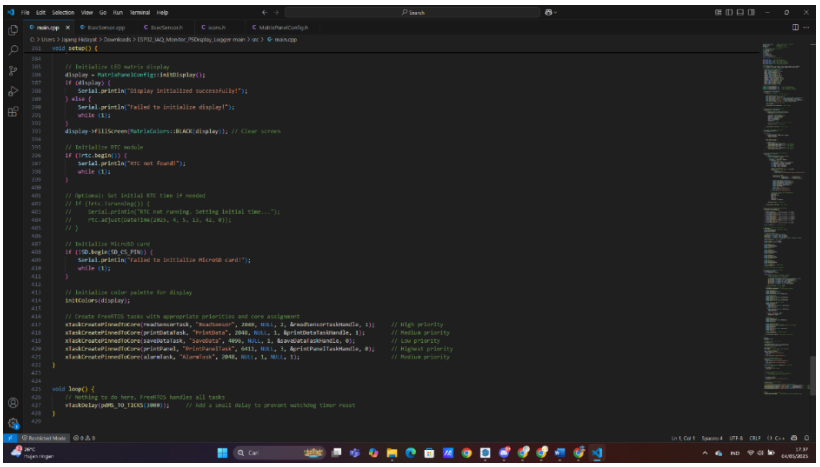


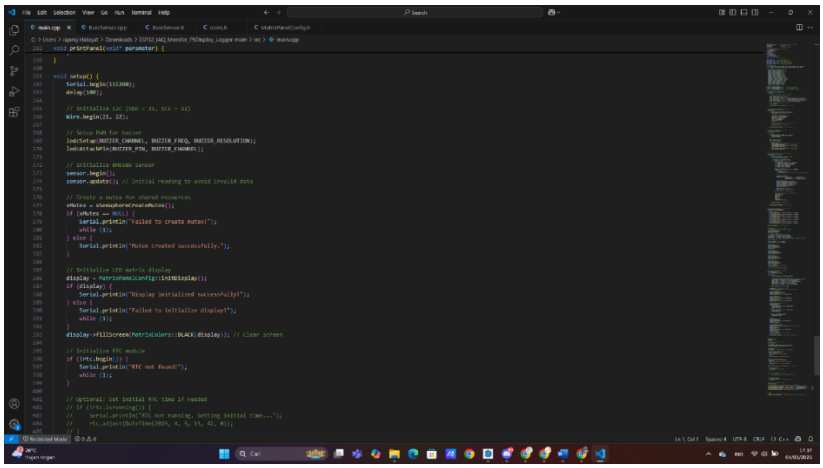
```
void BsecSensor::printData() {
    Serial.print(breathVocEquivalent);
    Serial.println(" PPM");

    Serial.println("IAQ Status = " + iaqStatus);
    Serial.println();
}

void BsecSensor::checkStatus() {
    if (iaqSensor.bsecStatus != BSEC_OK) {
        if (iaqSensor.bsecStatus < BSEC_OK) {
            output = "BSEC error code : " + String(iaqSensor.bsecStatus);
            Serial.println(output);
            for (;;);
        } else {
            output = "BSEC warning code : " + String(iaqSensor.bsecStatus);
            Serial.println(output);
        }
    }

    if (iaqSensor.bme68xStatus != BME68X_OK) {
        if (iaqSensor.bme68xStatus < BME68X_OK) {
            output = "BME68X error code : " + String(iaqSensor.bme68xStatus);
            Serial.println(output);
            for (;;);
        } else {
            output = "BME68X warning code : " + String(iaqSensor.bme68xStatus);
            Serial.println(output);
        }
    }
}

void BsecSensor::updateIAQStatus() {
    if ((iaq > 0)  && (iaq <= 50)) {
        iaqStatus = "Good";
    } else if ((iaq > 51)  && (iaq <= 100)) {
        iaqStatus = "Average";
    } else if ((iaq > 101)  && (iaq <= 150)) {
        iaqStatus = "Little Bad";
    } else if ((iaq > 151)  && (iaq <= 200)) {
        iaqStatus = "Bad";
    } else if ((iaq > 201)  && (iaq <= 300)) {
        iaqStatus = "Worse";
    } else if ((iaq > 301)  && (iaq <= 500)) {
        iaqStatus = "Very Bad";
    } else if ((iaq > 500)){
        iaqStatus = "Very Very Bad";
    }
}
```

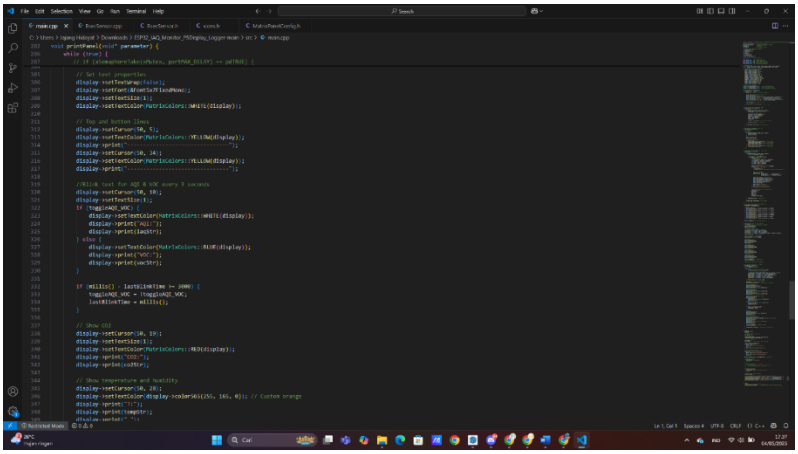


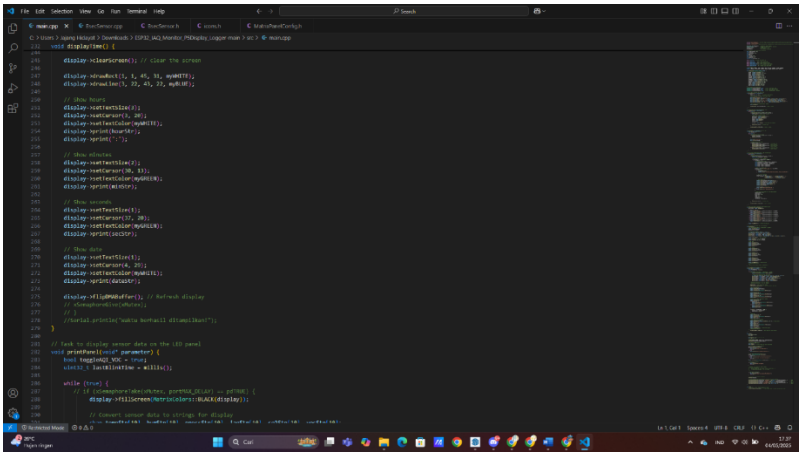
```
}

void BsecSensor::printData() {
    Serial.print("Temperature = ");
    Serial.print(temperature);
    Serial.println(" °C");

    Serial.print("Humidity = ");
    Serial.print(humidity);
    Serial.println(" %");

    Serial.print("Pressure = ");
    Serial.print(pressure);
    Serial.println(" hPa");

    Serial.print("IAQ = ");
    Serial.print(iaq);
    Serial.println(" PPM");

    Serial.print("CO2 equiv = ");
    Serial.print(co2Equivalent);
    Serial.println(" PPM");

    Serial.print("Breath VOC = ");
    Serial.print(breathVocEquivalent);
    Serial.println(" PPM");

    Serial.println("IAQ Status = " + iaqStatus);
    Serial.println();
}

void BsecSensor::checkStatus() {
    if (iaqSensor.bsecStatus != BSEC_OK) {
        if (iaqSensor.bsecStatus < BSEC_OK) {
            output = "BSEC error code : " + String(iaqSensor.bsecStatus);
            Serial.println(output);
            for (;;);
        } else {
            output = "BSEC warning code : " + String(iaqSensor.bsecStatus);
            Serial.println(output);
        }
    }

    if (iaqSensor.bme68xStatus != BME68X_OK) {
        if (iaqSensor.bme68xStatus < BME68X_OK) {
            output = "BME68X error code : " + String(iaqSensor.bme68xStatus);
            Serial.println(output);
            for (;;);
```

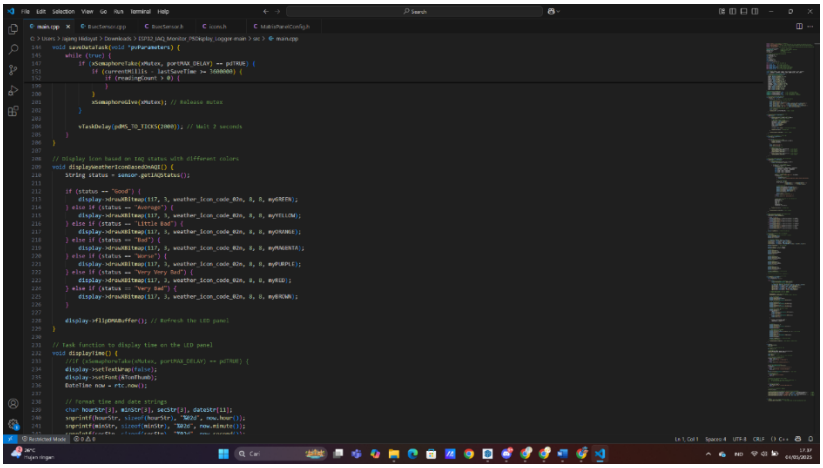


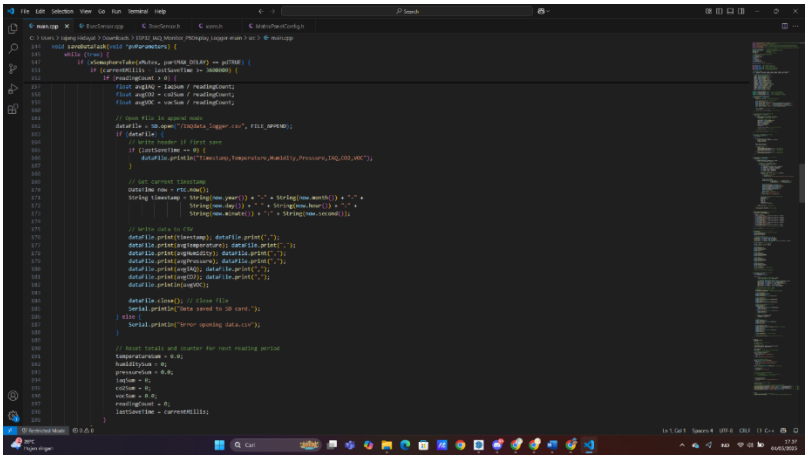


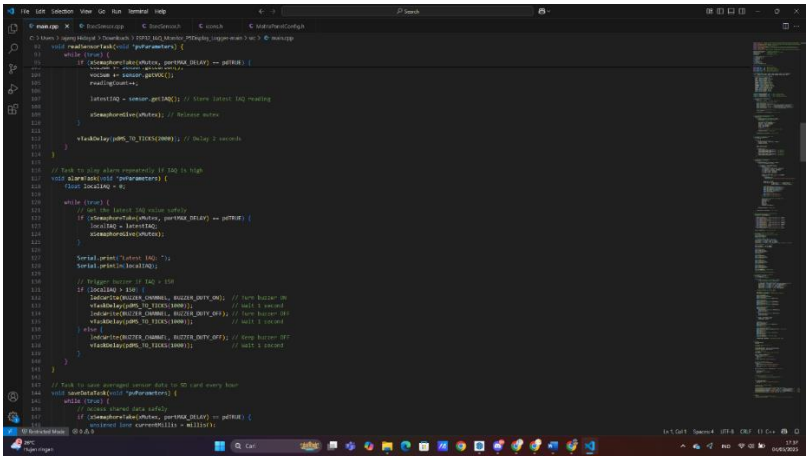


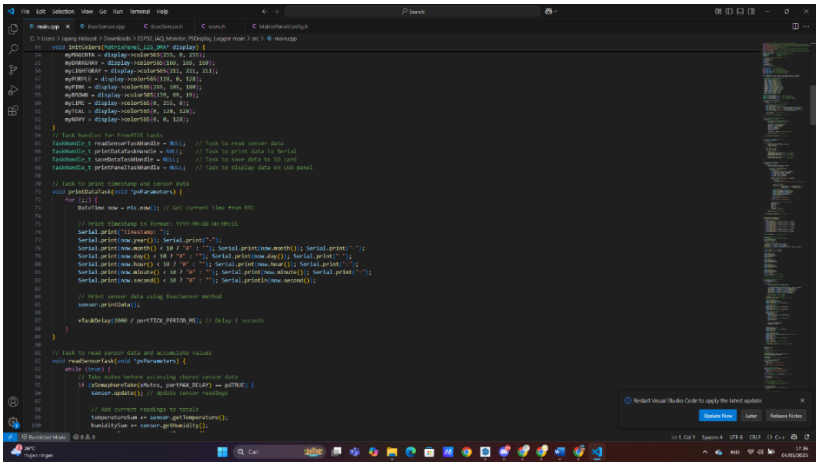


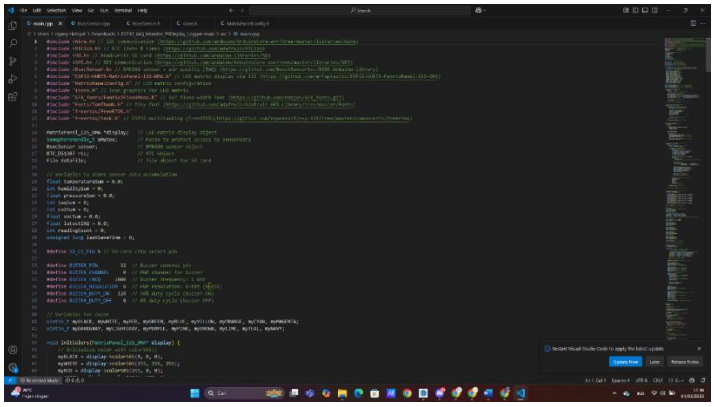


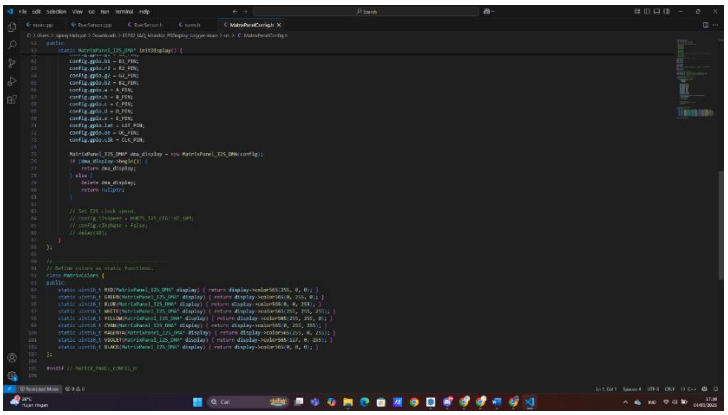




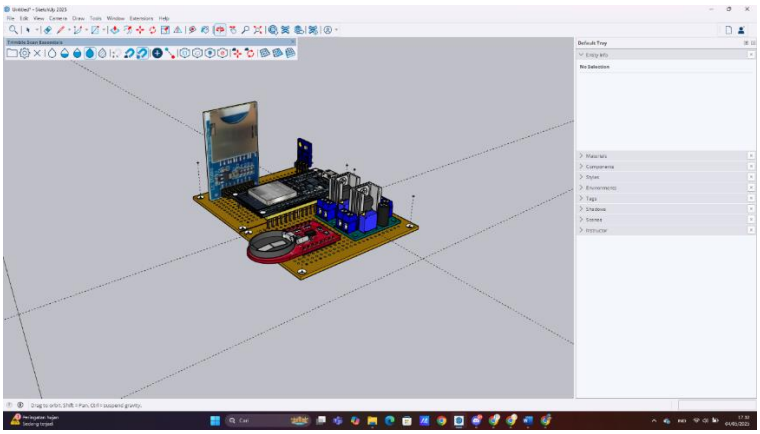


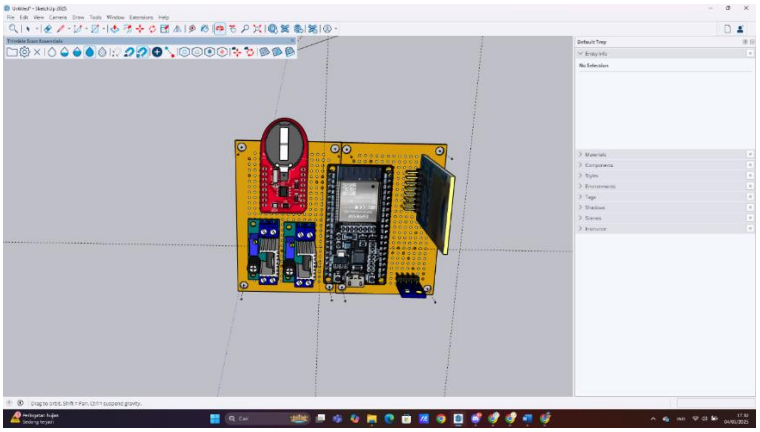


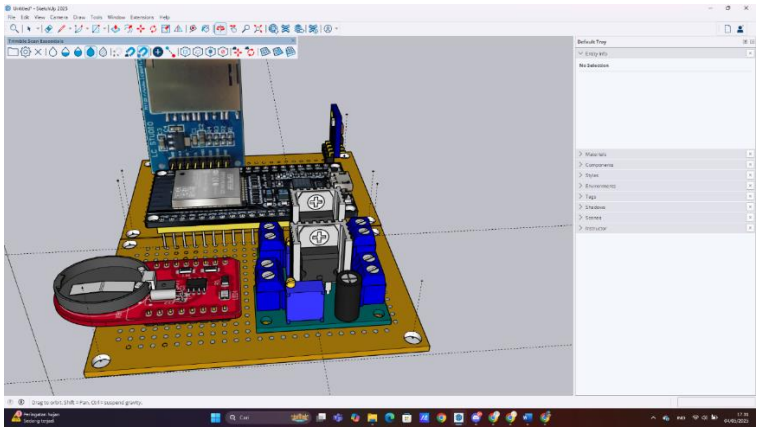


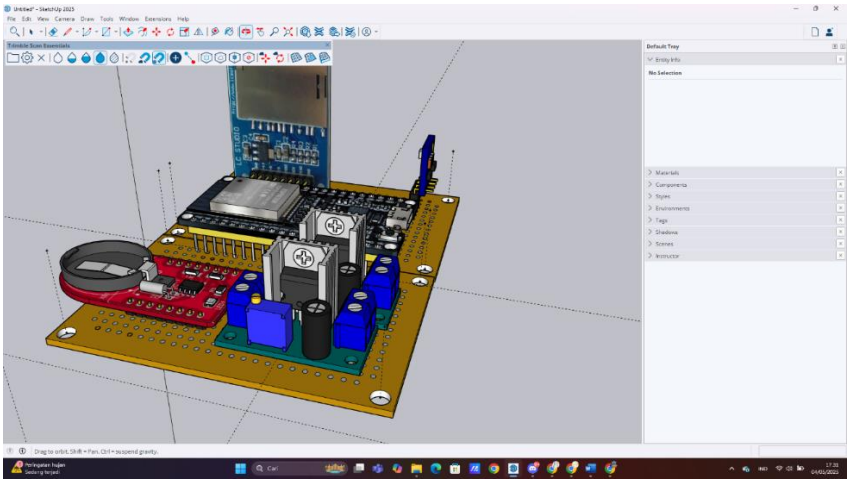


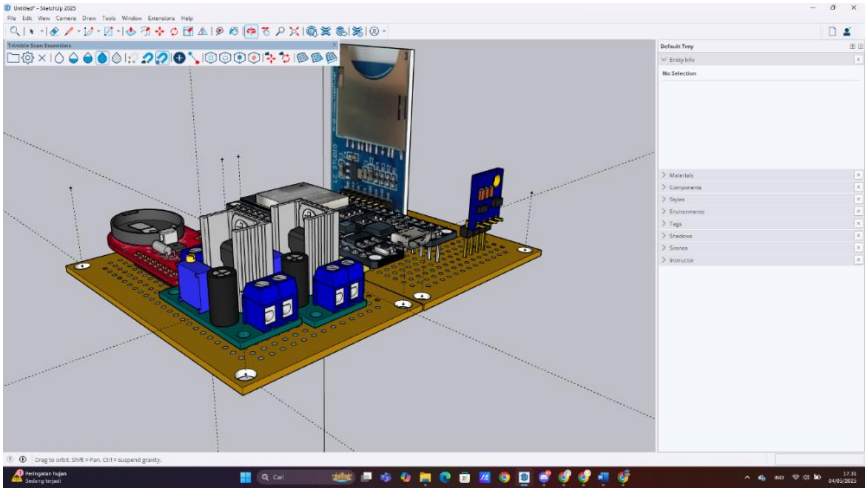
Desain Bentuk 3D

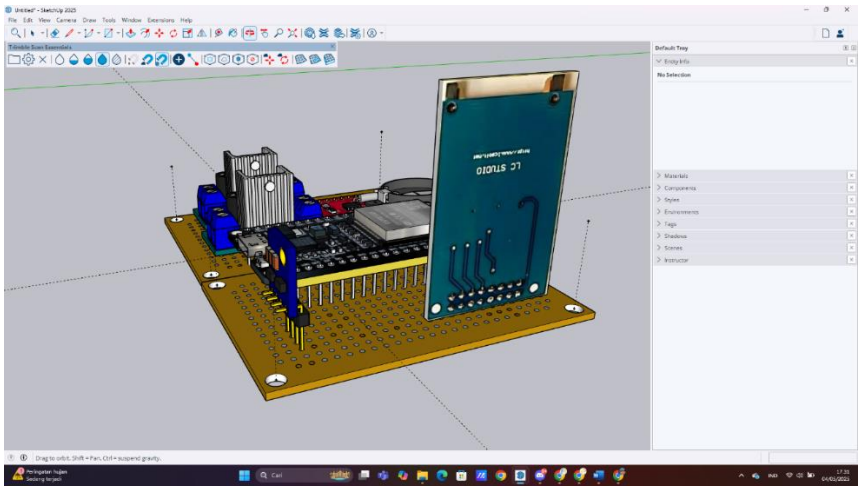


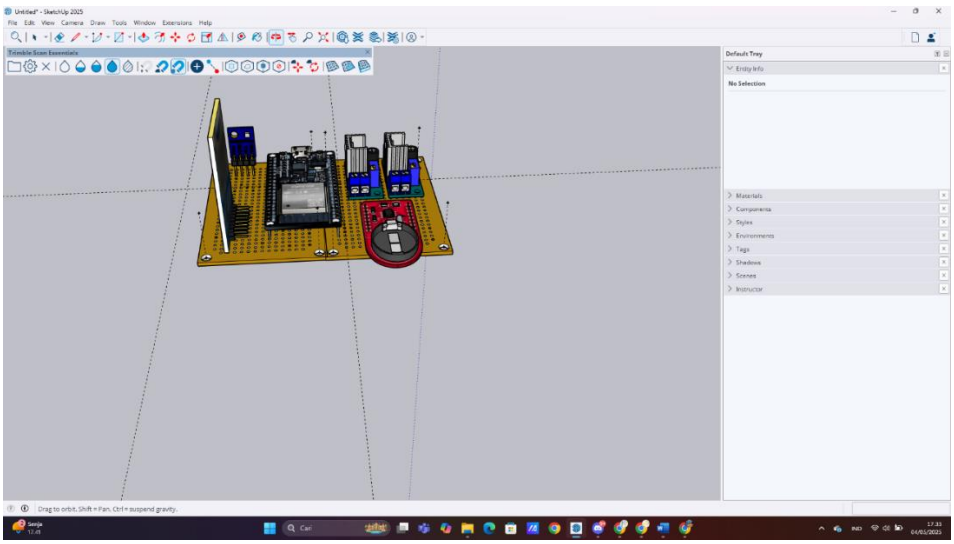




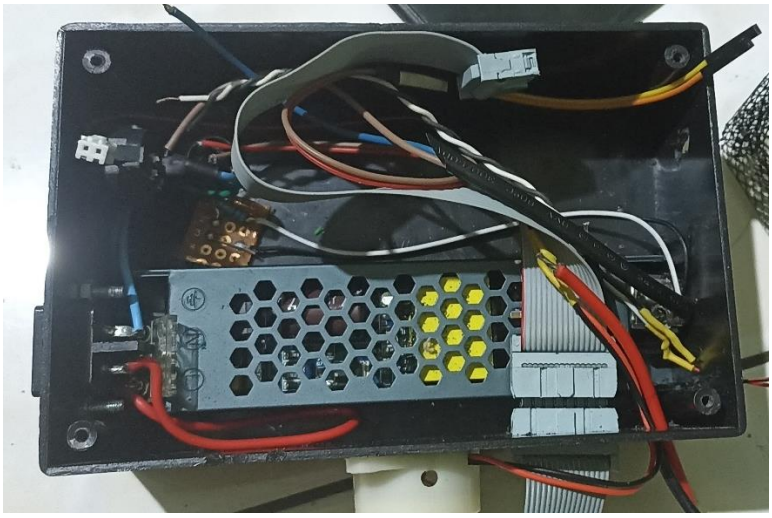


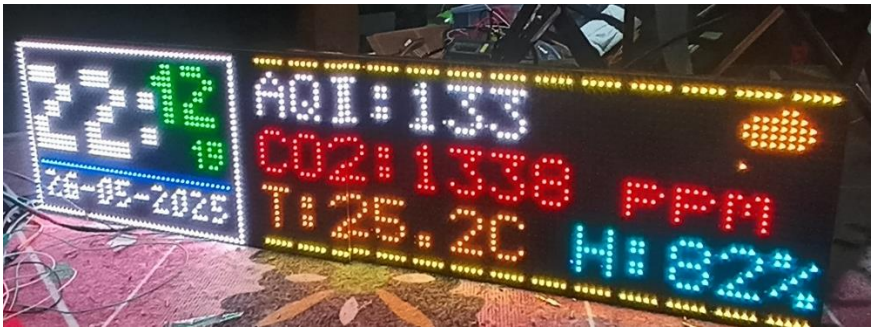






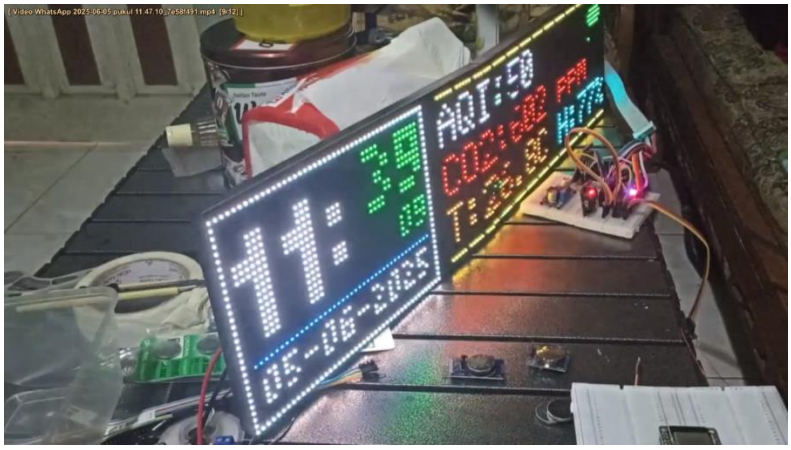
Dokumentasi Alat





Dokumentasi pengambilan data

Vidio Pengambilan data pada : https://drive.google.com/drive/folders/1bAnck9_e-fuLmMX22UOuAOF4tJ-lI1kk




22:54
25
03-06-2025
AQI:0
CO2:400 PPM
T:23.7C H:82%


18:40
44
03-06-2025
VOC:0.5 PPM
CO2:599 PPM
T:24.5C H:82%


17:17
26
03-06-2025
VOC:0.5 PPM
CO2:616 PPM
T:26.4C H:79%


AQI:47
CO2:590 PPM
T:27.4C H:72%
03-06-2025


16:08
40
04-06-2025
AQI:50
CO2:602 PPM
T:26.4C H:72%

Kode pemerograman yang digunakan

Kode program:

1. Main program

#include <Wire.h> // I2C communication (https://github.com/arduino/ArduinoCore-avr/tree/master/libraries/Wire)

#include <RTClib.h> // RTC (date & time) (https://github.com/adafruit/RTClib)

#include <SD.h> // Read/write SD card (https://github.com/arduino-libraries/SD)

#include <SPI.h> // SPI communication (https://github.com/arduino/ArduinoCore-avr/tree/master/libraries/SPI)

#include <BsecSensor.h> // BME680 sensor + air quality (IAQ) (https://github.com/BoschSensortec/BSEC-Arduino-library)

#include "ESP32-HUB75-MatrixPanel-I2S-DMA.h" // LED matrix display via I2S (https://github.com/mrfaptastic/ESP32-HUB75-MatrixPanel-I2S-DMA)

#include "MatrixPanelConfig.h" // LED matrix configuration

#include "icons.h" // Icon graphics for LED matrix

#include "GFX_fonts/Font5x7FixedMono.h" // 5x7 fixed-width font (https://github.com/robjen/GFX_fonts.git)

#include "Fonts/TomThumb.h" // Tiny font (https://github.com/adafruit/Adafruit-GFX-Library/tree/master/Fonts)

#include "freertos/FreeRTOS.h"

#include "freertos/task.h" // ESP32 multitasking (FreeRTOS)(https://github.com/espressif/esp-idf/tree/master/components/freertos)

MatrixPanel_I2S_DMA *display; // LED matrix display object

SemaphoreHandle_t xMutex; // Mutex to protect access to sensorData

```
BsecSensor sensor;              // BME680 sensor object
RTC_DS1307 rtc;                 // RTC object
File dataFile;                  // File object for SD card

// Variables to store sensor data accumulation
float temperatureSum = 0.0;
int humiditySum = 0;
float pressureSum = 0.0;
int iaqSum = 0;
int co2Sum = 0;
float vocSum = 0.0;
float latestIAQ = 0.0;
int readingCount = 0;
unsigned long lastSaveTime = 0;

#define SD_CS_PIN 5 // SD card chip select pin

#define BUZZER_PIN        32  // Buzzer control pin
#define BUZZER_CHANNEL     0  // PWM channel for buzzer
#define BUZZER_FREQ     1000  // Buzzer frequency: 1 kHz
#define BUZZER_RESOLUTION  8  // PWM resolution: 8-bit (0–255)
#define BUZZER_DUTY_ON   128  // 50% duty cycle (buzzer ON)
#define BUZZER_DUTY_OFF    0  // 0% duty cycle (buzzer OFF)
```

```
// Variables for color
uint16_t myBLACK, myWHITE, myRED, myGREEN, myBLUE, myYELLOW, myORANGE, myCYAN, myMAGENTA;
uint16_t myDARKGRAY, myLIGHTGRAY, myPURPLE, myPINK, myBROWN, myLIME, myTEAL, myNAVY;

void initColors(MatrixPanel_I2S_DMA* display) {
  // Initialize color with color565()
  myBLACK = display->color565(0, 0, 0);
  myWHITE = display->color565(255, 255, 255);
  myRED = display->color565(255, 0, 0);
  myGREEN = display->color565(0, 255, 0);
  myBLUE = display->color565(0, 0, 255);
  myYELLOW = display->color565(255, 255, 0);
  myORANGE = display->color565(255, 165, 0);
  myCYAN = display->color565(0, 255, 255);
  myMAGENTA = display->color565(255, 0, 255);
  myDARKGRAY = display->color565(169, 169, 169);
  myLIGHTGRAY = display->color565(211, 211, 211);
  myPURPLE = display->color565(128, 0, 128);
  myPINK = display->color565(255, 105, 180);
  myBROWN = display->color565(139, 69, 19);
```

```
    myLIME = display->color565(0, 255, 0);
    myTEAL = display->color565(0, 128, 128);
    myNAVY = display->color565(0, 0, 128);
}
// Task handles for FreeRTOS tasks
TaskHandle_t readSensorTaskHandle = NULL;   // Task to read sensor data
TaskHandle_t printDataTaskHandle = NULL;    // Task to print data to Serial
TaskHandle_t saveDataTaskHandle = NULL;     // Task to save data to SD card
TaskHandle_t printPanelTaskHandle = NULL;   // Task to display data on LED panel

// Task to print timestamp and sensor data
void printDataTask(void *pvParameters) {
    for (;;) {
        DateTime now = rtc.now(); // Get current time from RTC

        // Print timestamp in format: YYYY-MM-DD HH:MM:SS
        Serial.print("Timestamp: ");
        Serial.print(now.year()); Serial.print("-");
        Serial.print(now.month() < 10 ? "0" : ""); Serial.print(now.month()); Serial.print("-
");
        Serial.print(now.day() < 10 ? "0" : ""); Serial.print(now.day()); Serial.print(" ");
        Serial.print(now.hour() < 10 ? "0" : ""); Serial.print(now.hour()); Serial.print(":");
```

```
        Serial.print(now.minute() < 10 ? "0" : ""); Serial.print(now.minute()); Serial.print(":");
        Serial.print(now.second() < 10 ? "0" : ""); Serial.println(now.second());

        // Print sensor data using BsecSensor method
        sensor.printData();

        vTaskDelay(2000 / portTICK_PERIOD_MS); // Delay 2 seconds
    }
}

// Task to read sensor data and accumulate values
void readSensorTask(void *pvParameters) {
    while (true) {
        // Take mutex before accessing shared sensor data
        if (xSemaphoreTake(xMutex, portMAX_DELAY) == pdTRUE) {
            sensor.update(); // Update sensor readings

            // Add current readings to totals
            temperatureSum += sensor.getTemperature();
            humiditySum += sensor.getHumidity();
            pressureSum += sensor.getPressure();
            iaqSum += sensor.getIAQ();
```

```
            co2Sum += sensor.getCarbon();
            vocSum += sensor.getVOC();
            readingCount++;

            latestIAQ = sensor.getIAQ(); // Store latest IAQ reading

            xSemaphoreGive(xMutex); // Release mutex
        }

        vTaskDelay(pdMS_TO_TICKS(2000)); // Delay 2 seconds
    }
}

// Task to play alarm repeatedly if IAQ is high
void alarmTask(void *pvParameters) {
    float localIAQ = 0;

    while (true) {
        // Get the latest IAQ value safely
        if (xSemaphoreTake(xMutex, portMAX_DELAY) == pdTRUE) {
            localIAQ = latestIAQ;
            xSemaphoreGive(xMutex);
        }
```

```
        Serial.print("Latest IAQ: ");
        Serial.println(localIAQ);

        // Trigger buzzer if IAQ > 150
        if (localIAQ > 150) {
            ledcWrite(BUZZER_CHANNEL, BUZZER_DUTY_ON);  // Turn buzzer ON
            vTaskDelay(pdMS_TO_TICKS(1000));            // Wait 1 second
            ledcWrite(BUZZER_CHANNEL, BUZZER_DUTY_OFF); // Turn buzzer OFF
            vTaskDelay(pdMS_TO_TICKS(1000));            // Wait 1 second
        } else {
            ledcWrite(BUZZER_CHANNEL, BUZZER_DUTY_OFF); // Keep buzzer OFF
            vTaskDelay(pdMS_TO_TICKS(1000));            // Wait 1 second
        }
    }
}

// Task to save averaged sensor data to SD card every hour
void saveDataTask(void *pvParameters) {
    while (true) {
        // Access shared data safely
        if (xSemaphoreTake(xMutex, portMAX_DELAY) == pdTRUE) {
            unsigned long currentMillis = millis();
```

```
    // Save data every 1 hour (3600000 ms)
    if (currentMillis - lastSaveTime >= 3600000) {
      if (readingCount > 0) {
        // Calculate average values
        float avgTemperature = temperatureSum / readingCount;
        float avgHumidity = humiditySum / readingCount;
        float avgPressure = pressureSum / readingCount;
        float avgIAQ = iaqSum / readingCount;
        float avgCO2 = co2Sum / readingCount;
        float avgVOC = vocSum / readingCount;

        // Open file in append mode
        dataFile = SD.open("/IAQdata_logger.csv", FILE_APPEND);
        if (dataFile) {
          // Write header if first save
          if (lastSaveTime == 0) {

dataFile.println("Timestamp,Temperature,Humidity,Pressure,IAQ,CO2,VOC");
          }

          // Get current timestamp
          DateTime now = rtc.now();
```

```
    String timestamp = String(now.year()) + "-" + String(now.month()) + "-"
+
                String(now.day()) + " " + String(now.hour()) + ":" +
                String(now.minute()) + ":" + String(now.second());

    // Write data to CSV
    dataFile.print(timestamp); dataFile.print(",");
    dataFile.print(avgTemperature); dataFile.print(",");
    dataFile.print(avgHumidity); dataFile.print(",");
    dataFile.print(avgPressure); dataFile.print(",");
    dataFile.print(avgIAQ); dataFile.print(",");
    dataFile.print(avgCO2); dataFile.print(",");
    dataFile.println(avgVOC);

    dataFile.close(); // Close file
    Serial.println("Data saved to SD card.");
} else {
    Serial.println("Error opening data.csv");
}

// Reset totals and counter for next reading period
temperatureSum = 0.0;
humiditySum = 0;
```

```
            pressureSum = 0.0;
            iaqSum = 0;
            co2Sum = 0;
            vocSum = 0.0;
            readingCount = 0;
            lastSaveTime = currentMillis;
          }
        }
        xSemaphoreGive(xMutex); // Release mutex
      }

      vTaskDelay(pdMS_TO_TICKS(2000)); // Wait 2 seconds
    }
}

// Display icon based on IAQ status with different colors
void displayWeatherIconBasedOnAQI() {
    String status = sensor.getIAQStatus();

    if (status == "Good") {
        display->drawXBitmap(117, 3, weather_icon_code_02n, 8, 8, myGREEN);
    } else if (status == "Average") {
        display->drawXBitmap(117, 3, weather_icon_code_02n, 8, 8, myYELLOW);
```

```
    } else if (status == "Little Bad") {
        display->drawXBitmap(117, 3, weather_icon_code_02n, 8, 8, myORANGE);
    } else if (status == "Bad") {
        display->drawXBitmap(117, 3, weather_icon_code_02n, 8, 8, myMAGENTA);
    } else if (status == "Worse") {
        display->drawXBitmap(117, 3, weather_icon_code_02n, 8, 8, myPURPLE);
    } else if (status == "Very Very Bad") {
        display->drawXBitmap(117, 3, weather_icon_code_02n, 8, 8, myRED);
    } else if (status == "Very Bad") {
        display->drawXBitmap(117, 3, weather_icon_code_02n, 8, 8, myBROWN);
    }

    display->flipDMABuffer(); // Refresh the LED panel
}

// Task function to display time on the LED panel
void displayTime() {
    //if (xSemaphoreTake(xMutex, portMAX_DELAY) == pdTRUE) {
    display->setTextWrap(false);
    display->setFont(&TomThumb);
    DateTime now = rtc.now();

    // Format time and date strings
```

```
    char hourStr[3], minStr[3], secStr[3], dateStr[11];
    snprintf(hourStr, sizeof(hourStr), "%02d", now.hour());
    snprintf(minStr, sizeof(minStr), "%02d", now.minute());
    snprintf(secStr, sizeof(secStr), "%02d", now.second());
    snprintf(dateStr, sizeof(dateStr), "%02d-%02d-%04d", now.day(), now.month(), now.year());

    display->clearScreen(); // clear the screen

    display->drawRect(1, 1, 45, 31, myWHITE);
    display->drawLine(3, 22, 43, 22, myBLUE);

    // Show hours
    display->setTextSize(3);
    display->setCursor(3, 20);
    display->setTextColor(myWHITE);
    display->print(hourStr);
    display->print(":");

    // Show minutes
    display->setTextSize(2);
    display->setCursor(30, 13);
    display->setTextColor(myGREEN);
```

```
    display->print(minStr);

    // Show seconds
    display->setTextSize(1);
    display->setCursor(37, 20);
    display->setTextColor(myGREEN);
    display->print(secStr);

    // Show date
    display->setTextSize(1);
    display->setCursor(4, 29);
    display->setTextColor(myWHITE);
    display->print(dateStr);

    display->flipDMABuffer(); // Refresh display
    // xSemaphoreGive(xMutex);
    // }
    //Serial.println("Waktu berhasil ditampilkan!");
}

// Task to display sensor data on the LED panel
void printPanel(void* parameter) {
    bool toggleAQI_VOC = true;
```

```
uint32_t lastBlinkTime = millis();

while (true) {
  // if (xSemaphoreTake(xMutex, portMAX_DELAY) == pdTRUE) {
      display->fillScreen(MatrixColors::BLACK(display));

      // Convert sensor data to strings for display
      char tempStr[10], humStr[10], pressStr[10], iaqStr[10], co2Str[10], vocStr[10];
      snprintf(tempStr, sizeof(tempStr), "%.1fC", sensor.getTemperature());
      snprintf(humStr, sizeof(humStr), "%d%%", sensor.getHumidity());
      snprintf(pressStr, sizeof(pressStr), "%.1fhPa", sensor.getPressure());
      snprintf(iaqStr, sizeof(iaqStr), "%d", sensor.getIAQ());
      snprintf(co2Str, sizeof(co2Str), "%d ppm", sensor.getCarbon());
      snprintf(vocStr, sizeof(vocStr), "%.1f ppm", sensor.getVOC());

  // display->drawLine(49, 10, 128, 10, myBLUE);
  // display->drawLine(49, 31, 128, 31, myBLUE);

  displayTime(); // Show time on the display
  displayWeatherIconBasedOnAQI(); // Show icon based on IAQ status

  // Set text properties
  display->setTextWrap(false);
```

```
display->setFont(&Font5x7FixedMono);
display->setTextSize(1);
display->setTextColor(MatrixColors::WHITE(display));

// Top and bottom lines
display->setCursor(50, 5);
display->setTextColor(MatrixColors::YELLOW(display));
display->print("--------------------------------");
display->setCursor(50, 34);
display->setTextColor(MatrixColors::YELLOW(display));
display->print("--------------------------------");

//Blink text for AQI & VOC every 3 seconds
display->setCursor(50, 10);
display->setTextSize(1);
if (toggleAQI_VOC) {
   display->setTextColor(MatrixColors::WHITE(display));
   display->print("AQI:");
   display->print(iaqStr);
} else {
   display->setTextColor(MatrixColors::BLUE(display));
   display->print("VOC:");
   display->print(vocStr);
```

```
}

if (millis() - lastBlinkTime >= 3000) {
   toggleAQI_VOC = !toggleAQI_VOC;
   lastBlinkTime = millis();
}

// Show CO2
display->setCursor(50, 19);
display->setTextSize(1);
display->setTextColor(MatrixColors::RED(display));
display->print("CO2:");
display->print(co2Str);

// Show temperature and humidity
display->setCursor(50, 28);
display->setTextColor(display->color565(255, 165, 0)); // Custom orange
display->print("T:");
display->print(tempStr);
display->print(" ");
display->setTextColor(MatrixColors::CYAN(display));
display->print("H:");
display->print(humStr);
```

```
        display->flipDMABuffer(); // Refresh the display
        //xSemaphoreGive(xMutex);
    //}
        vTaskDelay(pdMS_TO_TICKS(50)); // Small delay for smooth refresh
    }
}

void setup() {
    Serial.begin(115200);
    delay(100);

    // Initialize I2C (SDA = 21, SCL = 22)
    Wire.begin(21, 22);

    // Setup PWM for buzzer
    ledcSetup(BUZZER_CHANNEL, BUZZER_FREQ, BUZZER_RESOLUTION);
    ledcAttachPin(BUZZER_PIN, BUZZER_CHANNEL);

    // Initialize BME688 sensor
    sensor.begin();
    sensor.update(); // Initial reading to avoid invalid data
```

```
// Create a mutex for shared resources
xMutex = xSemaphoreCreateMutex();
if (xMutex == NULL) {
    Serial.println("Failed to create mutex!");
    while (1);
} else {
    Serial.println("Mutex created successfully.");
}

// Initialize LED matrix display
display = MatrixPanelConfig::initDisplay();
if (display) {
    Serial.println("Display initialized successfully!");
} else {
    Serial.println("Failed to initialize display!");
    while (1);
}
display->fillScreen(MatrixColors::BLACK(display)); // Clear screen

// Initialize RTC module
if (!rtc.begin()) {
    Serial.println("RTC not found!");
    while (1);
```

```
}

// Optional: Set initial RTC time if needed
// if (!rtc.isrunning()) {
//     Serial.println("RTC not running. Setting initial time...");
//     rtc.adjust(DateTime(2025, 4, 5, 13, 42, 0));
// }

// Initialize MicroSD card
if (!SD.begin(SD_CS_PIN)) {
    Serial.println("Failed to initialize MicroSD card!");
    while (1);
}

// Initialize color palette for display
initColors(display);

// Create FreeRTOS tasks with appropriate priorities and core assignment
xTaskCreatePinnedToCore(readSensorTask, "ReadSensor", 2048, NULL, 2, &readSensorTaskHandle, 1);     // High priority
xTaskCreatePinnedToCore(printDataTask, "PrintData", 2048, NULL, 1, &printDataTaskHandle, 1);        // Medium priority
xTaskCreatePinnedToCore(saveDataTask, "SaveData", 4096, NULL, 1, &saveDataTaskHandle, 0);           // Low priority
```

```
  xTaskCreatePinnedToCore(printPanel, "PrintPanelTask", 6411, NULL, 3, &printPanelTaskHandle, 0);     // Highest priority
  xTaskCreatePinnedToCore(alarmTask, "AlarmTask", 2048, NULL, 1, NULL, 1);     // Medium priority
}


void loop() {
  // Nothing to do here, FreeRTOS handles all tasks
  vTaskDelay(pdMS_TO_TICKS(3000));     // Add a small delay to prevent watchdog timer reset
}
```

2. Sensor

```
// BsecSensor.h
#ifndef BSECSENSOR_H
#define BSECSENSOR_H

#include <Wire.h>
#include "bsec.h"

class BsecSensor {
public:
  BsecSensor();
  void begin();
```

```
    void update();
    void printData();
    void checkStatus();

    float getTemperature() const { return temperature; }
    int getHumidity() const { return humidity; }
    float getPressure() const { return pressure; }
    int getIAQ() const { return iaq; }
    int getCarbon() const { return co2Equivalent; }
    float getVOC() const { return breathVocEquivalent; }
    String getIAQStatus() const { return iaqStatus; }

private:
    Bsec iaqSensor;
    String output;
    float temperature;
    int humidity;
    float pressure;
    int iaq;
    int co2Equivalent;
    float breathVocEquivalent;
    String iaqStatus;
```

```
    void updateIAQStatus();
};

#endif // BSECSENSOR_H

// BsecSensor.cpp
#include "BsecSensor.h"

BsecSensor::BsecSensor() {
    temperature = 0.0;
    humidity = 0.0;
    pressure = 0.0;
    iaq = 0.0;
    co2Equivalent = 0.0;
    breathVocEquivalent = 0.0;
    iaqStatus = "Unknown";
}

void BsecSensor::begin() {
    Wire.begin();
    Serial.println(F("Starting..."));

    iaqSensor.begin(0x77, Wire);
```

```
    output = "BSEC library version " + String(iaqSensor.version.major) + "." + String(iaqSensor.version.minor) + "." + String(iaqSensor.version.major_bugfix) + "." + String(iaqSensor.version.minor_bugfix);
  Serial.println(output);
  checkStatus();

  bsec_virtual_sensor_t sensorList[10] = {
    BSEC_OUTPUT_RAW_TEMPERATURE,
    BSEC_OUTPUT_RAW_PRESSURE,
    BSEC_OUTPUT_RAW_HUMIDITY,
    BSEC_OUTPUT_RAW_GAS,
    BSEC_OUTPUT_IAQ,
    BSEC_OUTPUT_STATIC_IAQ,
    BSEC_OUTPUT_CO2_EQUIVALENT,
    BSEC_OUTPUT_BREATH_VOC_EQUIVALENT,
    BSEC_OUTPUT_SENSOR_HEAT_COMPENSATED_TEMPERATURE,
    BSEC_OUTPUT_SENSOR_HEAT_COMPENSATED_HUMIDITY,
  };

  iaqSensor.updateSubscription(sensorList, 10, BSEC_SAMPLE_RATE_LP);
  checkStatus();
}
```

```
void BsecSensor::update() {
    //unsigned long time_trigger = millis();
    if (iaqSensor.run()) { // If new data is available
        //output = String(time_trigger);
        output += ", " + String(iaqSensor.rawTemperature);
        output += ", " + String(iaqSensor.pressure);
        output += ", " + String(iaqSensor.rawHumidity);
        output += ", " + String(iaqSensor.gasResistance);
        output += ", " + String(iaqSensor.iaq);
        output += ", " + String(iaqSensor.iaqAccuracy);
        output += ", " + String(iaqSensor.temperature);
        output += ", " + String(iaqSensor.humidity);
        output += ", " + String(iaqSensor.staticIaq);
        output += ", " + String(iaqSensor.co2Equivalent);
        output += ", " + String(iaqSensor.breathVocEquivalent);
        //Serial.println(output);

        temperature = iaqSensor.temperature;
        humidity = iaqSensor.humidity;
        pressure = iaqSensor.pressure / 100.0;
        iaq = iaqSensor.staticIaq;
        co2Equivalent = iaqSensor.co2Equivalent;
        breathVocEquivalent = iaqSensor.breathVocEquivalent;
```

```
        updateIAQStatus();
    } else {
        checkStatus();
    }
}

void BsecSensor::printData() {
    Serial.print("Temperature = ");
    Serial.print(temperature);
    Serial.println(" *C");

    Serial.print("Humidity = ");
    Serial.print(humidity);
    Serial.println(" %");

    Serial.print("Pressure = ");
    Serial.print(pressure);
    Serial.println(" hPa");

    Serial.print("IAQ = ");
    Serial.print(iaq);
    Serial.println(" PPM");
```

```
    Serial.print("CO2 equiv = ");
    Serial.print(co2Equivalent);
    Serial.println(" PPM");

    Serial.print("Breath VOC = ");
    Serial.print(breathVocEquivalent);
    Serial.println(" PPM");

    Serial.println("IAQ Status = " + iaqStatus);
    Serial.println();
}

void BsecSensor::checkStatus() {
    if (iaqSensor.bsecStatus != BSEC_OK) {
        if (iaqSensor.bsecStatus < BSEC_OK) {
            output = "BSEC error code : " + String(iaqSensor.bsecStatus);
            Serial.println(output);
            for (;;);
        } else {
            output = "BSEC warning code : " + String(iaqSensor.bsecStatus);
            Serial.println(output);
        }
```

```
    }

    if (iaqSensor.bme68xStatus != BME68X_OK) {
        if (iaqSensor.bme68xStatus < BME68X_OK) {
            output = "BME688 error code : " + String(iaqSensor.bme68xStatus);
            Serial.println(output);
            for (;;);
        } else {
            output = "BME688 warning code : " + String(iaqSensor.bme68xStatus);
            Serial.println(output);
        }
    }
}

void BsecSensor::updateIAQStatus() {
    if ((iaq > 0)  && (iaq <= 50)) {
        iaqStatus = "Good";
    } else if ((iaq > 51)  && (iaq <= 100)) {
        iaqStatus = "Average";
    } else if ((iaq > 101)  && (iaq <= 150)) {
        iaqStatus = "Little Bad";
    } else if ((iaq > 151)  && (iaq <= 200)) {
        iaqStatus = "Bad";
```

```
    } else if ((iaq > 201)  && (iaq <= 300)) {
        iaqStatus = "Worse";
    } else if ((iaq > 301)  && (iaq <= 500)) {
        iaqStatus = "Very Bad";
    } else if ((iaq > 500)){
        iaqStatus = "Very Very Bad";
    }
}
```

3. Matrix panel

```
#ifndef MATRIX_PANEL_CONFIG_H
#define MATRIX_PANEL_CONFIG_H

#include <ESP32-HUB75-MatrixPanel-I2S-DMA.h>

//---------------------------------------
// // Defines the connected PIN between P5 and ESP32.
// #define R1_PIN 15
// #define G1_PIN 2
// #define B1_PIN 13
// #define R2_PIN 12
// #define G2_PIN 0
// #define B2_PIN 14
```

```
// #define A_PIN 27
// #define B_PIN 4
// #define C_PIN 26
// #define D_PIN 16
// #define E_PIN -1  // Required for 1/32 scan panels, like 64x64px. Any available pin would do.

// #define LAT_PIN 17
// #define OE_PIN 33
// #define CLK_PIN 25

// Defines the connected PIN between P5 and ESP32.
#define R1_PIN 17
#define G1_PIN 16
#define B1_PIN 13
#define R2_PIN 12
#define G2_PIN 4
#define B2_PIN 14

#define A_PIN 27
#define B_PIN 0
#define C_PIN 26
#define D_PIN 2
```

```
#define E_PIN -1  // Required for 1/32 scan panels, like 64x64px. Any available pin would do.

#define LAT_PIN 15
#define OE_PIN 33
#define CLK_PIN 25

//---------------------------------------
// Defines the P5 Panel configuration.
#define PANEL_RES_X 128  // Number of pixels wide of each INDIVIDUAL panel module.
#define PANEL_RES_Y 32  // Number of pixels tall of each INDIVIDUAL panel module.
#define PANEL_CHAIN 1  // Total number of panels chained one to another

//---------------------------------------
// Initialize MatrixPanel_I2S_DMA as "dma_display".
class MatrixPanelConfig {
public:
   static MatrixPanel_I2S_DMA* initDisplay() {
      HUB75_I2S_CFG config(
         PANEL_RES_X,
         PANEL_RES_Y,
         PANEL_CHAIN
```

```
);

config.gpio.r1 = R1_PIN;
config.gpio.g1 = G1_PIN;
config.gpio.b1 = B1_PIN;
config.gpio.r2 = R2_PIN;
config.gpio.g2 = G2_PIN;
config.gpio.b2 = B2_PIN;
config.gpio.a = A_PIN;
config.gpio.b = B_PIN;
config.gpio.c = C_PIN;
config.gpio.d = D_PIN;
config.gpio.e = E_PIN;
config.gpio.lat = LAT_PIN;
config.gpio.oe = OE_PIN;
config.gpio.clk = CLK_PIN;

MatrixPanel_I2S_DMA* dma_display = new MatrixPanel_I2S_DMA(config);
if (dma_display->begin()) {
   return dma_display;
} else {
   delete dma_display;
   return nullptr;
```

```
    }

    // Set I2S clock speed.
    // config.i2sspeed = HUB75_I2S_CFG::HZ_10M;
    // config.clkphase = false;
    // delay(10);
  }
};

//--------------------------------------
// Define colors as static functions.
class MatrixColors {
public:
  static uint16_t RED(MatrixPanel_I2S_DMA* display) { return display->color565(255, 0, 0); }
  static uint16_t GREEN(MatrixPanel_I2S_DMA* display) { return display->color565(0, 255, 0); }
  static uint16_t BLUE(MatrixPanel_I2S_DMA* display) { return display->color565(0, 0, 255); }
  static uint16_t WHITE(MatrixPanel_I2S_DMA* display) { return display->color565(255, 255, 255); }
  static uint16_t YELLOW(MatrixPanel_I2S_DMA* display) { return display->color565(255, 255, 0); }
```

```
  static uint16_t CYAN(MatrixPanel_I2S_DMA* display) { return display->color565(0, 255, 255); }
  static uint16_t MAGENTA(MatrixPanel_I2S_DMA* display) { return display->color565(255, 0, 255); }
  static uint16_t VIOLET(MatrixPanel_I2S_DMA* display) { return display->color565(127, 0, 255); }
  static uint16_t BLACK(MatrixPanel_I2S_DMA* display) { return display->color565(0, 0, 0); }
};

#endif // MATRIX_PANEL_CONFIG_H
```

4. Icon

```
#ifndef ICONS_H
#define ICONS_H

const unsigned char weather_icon_code_02n [] PROGMEM = {
        0x30, 0x7e, 0xff, 0xff, 0x7e, 0x00, 0x00, 0x01
};


#endif // ICONS_H
```