SMART WATER DISPENSER BERBASIS IOT: IMPLEMENTASI METODOLOGI LEAN DALAM PENGEMBANGAN SISTEM *MONITORING* DAN *NOTIFIKASI*

Disusun Oleh:

N a m a : Syifa Nursabilla Mezi Zahra

NIM : 21523205

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA FAKULTAS TEKNOLOGI INDUSTRI UNIVERSITAS ISLAM INDONESIA

2025

HALAMAN PENGESAHAN DOSEN PEMBIMBING

SMART WATER DISPENSER BERBASIS IOT: IMPLEMENTASI METODOLOGI LEAN DALAM PENGEMBANGAN SISTEM *MONITORING* DAN *NOTIFIKASI*

TUGAS AKHIR

Disusun Oleh:

N a m a : Syifa Nursabilla Mezi Zahra

NIM : 21523205

Yogyakarta, 21 Juli 2025

Pembimbing,

(Kurniawan Dwi Irianto, S.T., M.Sc.)

HALAMAN PENGESAHAN DOSEN PENGUJI

SMART WATER DISPENSER BERBASIS IOT: IMPLEMENTASI METODOLOGI LEAN DALAM PENGEMBANGAN SISTEM *MONITORING* DAN *NOTIFIKASI*

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika – Program Sarjana di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 30 Juli 2025

Tim Penguji

Kurniawan Dwi Irianto, S.T., M.Sc. _____

Anggota 1

Chandra Kusuma Dewa, S.Kom., M.Cs, Ph.D. _____

Anggota 2

Izzati Muhimmah, S.T., M.Sc., Ph.D. _____

Mengetahui,
Ketua Program Studi Informatika – Program Sarjana
Fakultas Teknologi Industri
Universitas Islam Indonesia

(Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Syifa Nursabilla Mezi Zahra

NIM : 21523205

Tugas akhir dengan judul:

SMART WATER DISPENSER BERBASIS IOT: IMPLEMENTASI METODOLOGI LEAN DALAM PENGEMBANGAN SISTEM *MONITORING* DAN *NOTIFIKASI*

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 21 Juli 2025

(Syifa Nursabilla Mezi Zahra)

HALAMAN PERSEMBAHAN

Segala puji dan syukur penulis panjatkan ke hadirat Allah SWT, Tuhan Yang Maha Esa, atas limpahan rahmat, hidayah, dan kekuatan yang telah diberikan sehingga penulis dapat menyelesaikan tugas akhir ini dengan sebaik-baiknya. Dengan penuh rasa hormat, cinta, dan penghargaan yang tulus, skripsi ini penulis persembahkan kepada:

1. Allah SWT. Atas segala rahmat, hidayah, dan kasih sayang-Nya yang tiada terbatas, sehingga saya dapat menyelesaikan tugas akhir ini. Segala pencapaian ini adalah berkat izin-Nya.
2. Papa tersayang, Ir. Dezi Purnama, S.T., M.T. Panutan dalam hidup penulis, yang selalu hadir dengan keteguhan dan ketenangan. Terima kasih atas dukungan, kerja keras, dan semangat yang Papa tanamkan dalam diam yang membuat penulis belajar bertanggung jawab, tetap kuat menghadapi tekanan, dan tidak mudah menyerah.Meski penulis jauh dari rumah, Papa terus berusaha memenuhi segala kebutuhan, baik secara materi maupun mental. Di balik proses panjang ini, ada pengorbanan Papa yang mungkin tidak selalu terlihat, tetapi sangat penulis rasakan dan syukuri.Restu dan keyakinan Papa telah menjadi fondasi kuat bagi setiap pencapaian yang berhasil diraih. Segala pencapaian ini tidak akan pernah terwujud tanpa cinta, doa, dan pengorbanan yang tulus dari Mama dan Papa. Semoga lembaran sederhana ini bisa menjadi bentuk kecil dari rasa terima kasih penulis atas segala yang telah kalian berikan yang nilainya tak pernah bisa diukur dengan apapun.
3. Mama tercinta, Merry Fitrisia, S.E. Sosok yang selalu hadir dengan cinta yang tak pernah habis, yang menjadi tempat ternyaman untuk berbagi dan bersandar. Dari jauh di Kerinci, Mama tak pernah lelah mendoakan dan menguatkan. Terima kasih atas segala doa, kesabaran, dan kasih sayang yang tak pernah putus mengiringi setiap langkah penulis.Walau hanya lewat video call, setiap kata Mama adalah penyemangat yang hadir di saat semangat mulai goyah di tanah rantau. Dari Mama, penulis belajar arti ketulusan, kekuatan dalam kelembutan, dan cinta yang diam-diam menjadi bahan bakar terbesar dalam menyelesaikan skripsi ini.
4. Kepada adik tercinta, Faiqa Nuzulul Rizqi yang mungkin belum benar-benar mengerti apa arti skripsi, tapi selalu jadi penghibur dengan canda dan tawamu. Meski kita terpisah jarak, kehadiranmu selalu menjadi alasan untuk pulang dan tersenyum.

Terima kasih telah menjadi pengingat bahwa hidup tak melulu soal pencapaian, tapi juga tentang kebahagiaan sederhana.

5. Untuk Nenek dan Kakek tercinta, dari pihak Mama dan Papa, Terima kasih atas segala doa dan kasih sayang yang senantiasa menyertai langkah ini, meski mungkin tidak selalu diucapkan secara langsung. Kehangatan, ketulusan, dan keteguhan hati kalian menjadi teladan dalam menjalani hidup dengan penuh kesabaran dan keikhlasan. Semoga pencapaian kecil ini dapat menjadi salah satu bentuk kebanggaan dan hadiah sederhana untuk cinta yang telah kalian berikan sepanjang hidup.
6. Bapak Kurniawan Dwi Irianto, S.T., M.Sc. Selaku dosen pembimbing, yang telah dengan sabar membimbing, mengarahkan, dan memberikan masukan yang sangat berarti dalam setiap tahap penyusunan skripsi ini.Terima kasih atas waktu, dedikasi, dan perhatian yang Bapak curahkan, yang tidak hanya membantu dalam menyelesaikan karya ini, tetapi juga memperkaya pemahaman penulis dalam proses belajar yang sesungguhnya. Semoga ilmu dan bimbingan yang telah diberikan menjadi amal jariyah serta membawa keberkahan bagi Bapak.
7. Partner tersayang, Hasan Rama Sagita, S.Kom. Sosok yang menjadi rumah sekaligus pendamping setia bagi penulis selama menempuh perjalanan di tanah rantau. Terima kasih telah banyak berkontribusi dalam proses penulisan skripsi ini meluangkan tenaga, waktu, dan pikiran, serta senantiasa sabar dalam menghadapi segala dinamika yang penulis alami. Kehadiranmu telah menjadi bagian penting, bukan hanya dalam penyusunan karya ini, tetapi sejak awal langkah penulis menapaki dunia perkuliahan hingga sampai pada titik ini.
8. Untuk seluruh teman dan sahabat, yang namanya tidak bisa disebutkan satu per satu, terima kasih telah menjadi bagian dari perjalanan ini. Baik yang hadir hanya sebentar maupun yang menetap hingga akhir, setiap pertemuan, dukungan, dan tawa bersama telah meninggalkan jejak berharga dalam proses penulisan skripsi ini. Kehadiran kalian dalam bentuk pesan singkat, candaan ringan, atau bahkan keheningan yang saling memahami telah membantu penulis melewati banyak rintangan. Terima kasih telah hadir, dengan cara masing-masing.
9. Almamater tercinta, yang telah menjadi wadah penulis dalam tumbuh, belajar, dan menempa diri menjadi insan yang siap menghadapi tantangan masa depan.
10. Untuk diriku sendiri, yang telah bertahan sejauh ini, meskipun tak sedikit badai datang silih berganti. Terima kasih karena memilih untuk tidak menyerah, bahkan saat

semuanya terasa ingin menghentikan langkah. Untuk hari-hari di tanah rantau yang sunyi dan sepi, saat jarak dari rumah begitu terasa, saat pelukan keluarga hanya bisa dibayangkan, dan saat semuanya harus dijalani sendiri. Kamu telah melalui banyak malam penuh keraguan, menangis dalam diam, menyembunyikan lelah, dan tetap berdiri meski dunia terasa berat. Terima kasih telah mampu mengendalikan diri dari tekanan yang datang dari luar ekspektasi, perbandingan, komentar, dan keraguan orang lain. Terima kasih telah tidak menyerah pada jurusan yang kamu pilih sendiri, walau tak selalu mudah untuk dijalani dan dicintai. Kamu tetap setia, tetap berjuang, dan akhirnya sampai di titik ini. Untuk segala luka, lelah, dan keraguan yang berhasil kamu lewati: kamu layak berbangga. Perjalanan ini belum selesai, tapi kamu sudah membuktikan bahwa bertahan pun adalah bentuk paling tulus dari keberanian.

HALAMAN MOTO

“Dan bersabarlah kamu, sesungguhnya janji allah adalah benar.”

(Qs. Ar-Ruum:60)

“Selalu ada harga dalam sebuah proses. Nikmati saja lelah-lelah itu. Lebarkan lagi rasa sabar itu. Semua yang kau investasikan untuk menjadikan dirimu serupa yang kau impikan mungkin tidak akan selalu berjalan lancar. Tapi gelombang-gelombang itu yang nanti bisa kau ceritakan.”

-Boy Chandra-

KATA PENGANTAR

Segala puji dan syukur penulis panjatkan ke hadirat Allah SWT atas segala limpahan rahmat, taufik, serta hidayah-Nya sehingga penulis dapat menyelesaikan tugas akhir yang berjudul **"Perancangan dan Implementasi *Smart water dispenser* Berbasis IoT dengan Sistem *Monitoring* dan *Notifikasi Real-time*"** sebagai salah satu syarat untuk memperoleh gelar Sarjana pada Program Studi Teknik Informatika.

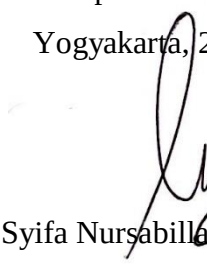
Skripsi ini disusun untuk menjawab permasalahan nyata dalam penggunaan dispenser air galon konvensional, yang sering kali belum memberikan kemudahan dalam aspek pemantauan volume air, suhu air, serta pengoperasian secara praktis. Dengan memanfaatkan teknologi *Internet of Things (IoT)*, penelitian ini bertujuan menghadirkan solusi cerdas dan terintegrasi yang lebih responsif terhadap kebutuhan pengguna di lingkungan kampus maupun ruang publik lainnya.

Penulis menyadari bahwa penyusunan skripsi ini tidak akan berhasil tanpa bantuan dan dukungan dari berbagai pihak. Oleh karena itu, dengan segala kerendahan hati, penulis menyampaikan ucapan terima kasih yang sebesar-besarnya kepada:

1. **Bapak Kurniawan Dwi Irianto, S.T., M.Sc.**, selaku dosen pembimbing, yang telah membimbing dengan sabar, memberikan arahan serta masukan yang sangat berharga dalam proses penyusunan skripsi ini.
2. **Seluruh dosen dan staf pengajar Program Studi Teknik Informatika**, atas ilmu dan wawasan yang telah diberikan selama masa perkuliahan.
3. **Kedua orang tua**, Papa Ir. Dezi Purnama, S.T., M.T. dan Mama Merry Fitrisia, S.E., atas doa, dukungan, dan pengorbanan yang tak terhingga.
4. **Partner tersayang, Hasan Rama Sagita, S.Kom.**, atas dukungan dan kesetiaan yang diberikan sejak awal perjalanan perkuliahan hingga akhir.
5. **Rekan-rekan seperjuangan dan teman-teman satu angkatan**, yang telah menjadi bagian dari perjalanan akademik ini melalui kebersamaan, diskusi, dan semangat belajar bersama.

Penulis menyadari bahwa skripsi ini masih jauh dari kesempurnaan.

Yogyakarta, 21 Juli 2025



(Syifa Nursabilla Mezi Zahra)

SARI

Dispenser air galon konvensional masih memiliki berbagai keterbatasan, seperti tidak adanya pemantauan volume air secara *real-time*, pengoperasian manual yang kurang higienis, serta belum tersedianya *notifikasi* otomatis saat air habis atau suhu belum optimal. Penelitian ini bertujuan untuk merancang dan mengimplementasikan prototipe *Smart water dispenser* berbasis *Internet of Things (IoT)* yang dilengkapi dengan sistem *monitoring* dan *notifikasi real-time* guna meningkatkan efisiensi dan kenyamanan penggunaan.

Pengembangan sistem menggunakan metodologi Lean Startup melalui tahapan *Build–Measure–Learn* secara iteratif untuk memastikan solusi yang dihasilkan adaptif terhadap kebutuhan pengguna. Sistem dirancang menggunakan mikrokontroler ESP32 yang terhubung dengan sensor *load cell*, sensor suhu DS18B20, serta sensor ultrasonik HC-SR04, dan memanfaatkan *Firebase Realtime Database* untuk penyimpanan *cloud* dan pengiriman *notifikasi* ke aplikasi *mobile* berbasis *React Native*. Fitur utama meliputi pemantauan volume air, suhu air, deteksi otomatis keberadaan gelas, serta *notifikasi* status galon secara langsung melalui aplikasi.

Hasil pengujian menunjukkan sistem mampu memberikan informasi secara *real-time* dengan rata-rata waktu tunda (delay) di bawah 200 ms dan tingkat akurasi sensor di atas 90%. Umpan balik dari pengguna juga mengindikasikan adanya peningkatan signifikan pada aspek kenyamanan, efisiensi, dan pengalaman penggunaan dibandingkan dengan dispenser konvensional. Prototipe ini membuktikan potensi penerapan IoT dan metodologi Lean dalam pengembangan solusi cerdas di lingkungan kampus maupun ruang publik lainnya.

Kata kunci: Internet of Things, *Smart water dispenser*, *Monitoring Real-time*, *Notifikasi* Otomatis, Lean Startup, ESP32.

GLOSARIUM

Glosarium memuat daftar kata tertentu yang digunakan dalam laporan dan membutuhkan penjelasan, misalnya kata serapan yang belum lazim digunakan. Urutkan sesuai abjad. Contoh penulisannya seperti di bawah ini:

Aktuator	Komponen mekanik atau elektrik yang mengubah sinyal kontrol menjadi aksi fisik, seperti membuka katup atau mengaktifkan pompa.
Arduino IDE	Lingkungan pengembangan perangkat lunak berbasis open-source yang digunakan untuk memprogram mikrokontroler seperti Arduino dan ESP32.
Build-Measure-Learn	Siklus pengembangan dalam metodologi Lean yang terdiri dari tahapan membangun produk (*build*), mengukur respons pengguna (*measure*), dan belajar dari data (*learn*).
Dispenser Pintar (Smart Water Dispenser)	Dispenser air minum yang dilengkapi dengan sensor, kontrol otomatis, dan konektivitas internet untuk memberikan pengalaman penggunaan yang lebih efisien dan higienis.
ESP32	Mikrokontroler dengan konektivitas Wi-Fi dan Bluetooth bawaan yang digunakan sebagai otak sistem dalam perangkat IoT.
Firebase Realtime Database	Layanan database *cloud* berbasis *NoSQL* yang memungkinkan sinkronisasi data secara *real-time* antar perangkat dan aplikasi.
Flow Sensor	Sensor yang digunakan untuk mengukur laju aliran air atau volume air yang mengalir melalui sistem.
HC-SR04	Sensor ultrasonik yang digunakan untuk mengukur jarak atau mendeteksi keberadaan objek, seperti gelas pada dispenser.
IoT (Internet of Things)	Konsep penghubungan objek fisik ke internet untuk saling bertukar data, memungkinkan kontrol dan pemantauan jarak jauh.
Kalibrasi	Proses penyesuaian sensor agar menghasilkan pembacaan yang akurat sesuai standar atau kondisi referensi.
Lean Methodology	Pendekatan pengembangan produk yang menekankan efisiensi, iterasi cepat, dan validasi langsung berdasarkan *feedback* pengguna.
Load cell	Sensor untuk mengukur berat, digunakan dalam sistem untuk

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Air merupakan elemen vital dalam menunjang kehidupan manusia serta termasuk dalam kebutuhan dasar di berbagai lingkungan, termasuk lingkungan kampus. Setiap individu memerlukan asupan air minum yang cukup dan berkualitas guna mendukung aktivitas seharihari serta menjaga kesehatan tubuh. Institute of Medicine (2005) merekomendasikan bahwa pria dewasa sebaiknya mengonsumsi sekitar 3 liter air per hari (±13 gelas), sedangkan wanita dewasa sekitar 2,2 liter per hari (±9 gelas). Oleh karena itu, ketersediaan air minum yang layak dan aman perlu menjadi prioritas, mengingat air yang higienis berperan penting dalam pencegahan gangguan kesehatan. Di kampus, sebagai lingkungan pendidikan yang padat aktivitas, penyediaan fasilitas air minum bagi mahasiswa, dosen, dan staf menjadi aspek krusial untuk menunjang produktivitas dan kenyamanan. Banyak institusi pendidikan tinggi mengandalkan penyediaan dispenser air galon sebagai solusi praktis untuk memenuhi kebutuhan hidrasi sivitas akademika.

Seiring dengan kemajuan teknologi, konsep Smart Campus (kampus cerdas) mulai diimplementasikan oleh berbagai perguruan tinggi sebagai upaya untuk meningkatkan efisiensi operasional dan kualitas layanan. Teknologi *Internet of Things (IoT)* menjadi katalis utama dalam transformasi kampus konvensional menjadi ekosistem yang lebih cerdas dan saling terhubung (Burhan, Rehman, Khan, & Kim, 2018). Penerapan IoT di lingkungan kampus dinilai sebagai langkah strategis untuk mengoptimalkan pengelolaan sumber daya, meningkatkan kenyamanan pengguna, serta mendukung inisiatif keberlanjutan lingkungan. Salah satu bentuk konkret penerapan IoT dapat ditemukan pada sistem pemantauan dan pengendalian terpusat terhadap infrastruktur seperti kelistrikan, pencahayaan, hingga ketersediaan air minum. Inovasi ini memungkinkan penggunaan sumber daya yang lebih efisien dan adaptif terhadap kebutuhan pengguna. Dengan pendekatan tersebut, operasional kampus menjadi lebih responsif dan efisien, sekaligus mendorong terwujudnya kampus hijau yang berkelanjutan.

Meski dispenser air galon konvensional masih menjadi pilihan utama di banyak kampus, perangkat ini memiliki sejumlah keterbatasan yang cukup signifikan. Salah satu kendala utamanya adalah ketidakmampuan pengguna untuk memantau volume air tersisa secara *real-time* tanpa pemeriksaan manual, yang kerap menyebabkan air habis secara tiba-tiba tanpa

diketahui sebelumnya. Kondisi ini dapat mengganggu aktivitas pengguna yang membutuhkan akses cepat terhadap air minum. Selain itu, sistem interaksi pada dispenser konvensional yang umumnya menggunakan (Burhan, Rehman, Khan, & Kim, 2018) tombol atau keran sentuh menimbulkan kekhawatiran terkait aspek kebersihan, khususnya dalam konteks pasca pandemi, di mana kesadaran akan higienitas permukaan bersama semakin meningkat.

Air minum merupakan kebutuhan mendasar yang memegang peran penting dalam menjaga kesehatan, kebugaran, serta menunjang produktivitas sehari-hari (World Health Organization, 2017). Ketersediaannya yang layak dan mudah diakses menjadi krusial di berbagai lingkungan, seperti perkantoran, institusi pendidikan, fasilitas layanan kesehatan, hingga ruang publik. Dispenser air galon banyak dipilih sebagai solusi praktis karena mudah dipasang dan mampu menyediakan air panas maupun dingin secara instan. Namun demikian, sistem pengelolaan manual pada dispenser ini kerap menemui berbagai kendala, seperti keterlambatan dalam pengisian ulang galon kosong yang dapat mengganggu kelancaran aktivitas, khususnya di tempat dengan tingkat konsumsi air yang tinggi. Selain itu, suhu air yang tidak stabil terlalu panas atau kurang dingin sering kali mengurangi kenyamanan pengguna. Oleh karena itu, pemantauan suhu air secara real-time menjadi sangat penting. Dengan adanya fitur monitoring suhu, pengguna dapat mengetahui kondisi suhu air panas dan dingin secara langsung, memastikan air yang diambil sudah sesuai kebutuhan tanpa harus menunggu atau menebak. Selain meningkatkan kenyamanan, fitur ini juga membantu menjaga keamanan pengguna dari risiko terkena air terlalu panas, serta mendukung efisiensi energi karena sistem dapat mengatur waktu kerja pemanas atau pendingin secara otomatis berdasarkan kebutuhan. Dengan pemantauan suhu real-time, kualitas air yang disajikan juga dapat lebih terjaga, sesuai standar kesehatan dan preferensi pengguna.Dispenser manual juga dinilai kurang praktis, terutama bagi pengguna yang sedang membawa barang, memegang gawai, atau memiliki keterbatasan fisik, karena pengoperasiannya masih mengandalkan tuas atau tombol yang harus ditekan secara langsung (Chou, Dewabharata, Bayu, Cheng, & Zulvia, 2022). Kondisi ini turut meningkatkan potensi terjadinya tumpahan atau kecelakaan kecil, seperti terkena cipratan air panas. Oleh sebab itu, dibutuhkan sistem penyedia air minum yang lebih cerdas, higienis, dan ramah bagi semua kalangan pengguna.

Kemajuan teknologi *Internet of Things (IoT)* membuka peluang baru dalam meningkatkan efisiensi, kenyamanan, serta ketepatan penggunaan berbagai perangkat dalam kehidupan sehari-hari. IoT memungkinkan integrasi antara sensor, aktuator, dan sistem komunikasi nirkabel yang beroperasi secara *real-time*, membentuk suatu ekosistem perangkat

yang saling terhubung serta responsif terhadap perubahan kondisi lingkungan (Burhan, Rehman, Khan, & Kim, 2018)Dalam konteks pengelolaan air minum, penerapan teknologi IoT pada dispenser air galon menghadirkan solusi cerdas untuk mengatasi berbagai kendala yang selama ini melekat pada sistem konvensional. Pendekatan ini tidak hanya meningkatkan efisiensi operasional, tetapi juga mendorong terciptanya sistem penyedia air minum yang lebih adaptif dan ramah pengguna.

Meskipun berbagai inovasi berbasis *Internet of Things (IoT)* telah mulai diterapkan pada dispenser air, sebagian besar masih terbatas pada fungsi-fungsi dasar, seperti pemantauan volume air dan otomatisasi pengisian. Namun demikian, banyak dari sistem tersebut belum terintegrasi secara menyeluruh dengan antarmuka pengguna yang intuitif, serta belum sepenuhnya mempertimbangkan aspek kenyamanan, efisiensi, dan adaptabilitas dalam berbagai konteks penggunaan. Masih terdapat ruang yang luas untuk merancang sistem dispenser air pintar yang tidak hanya mampu memberikan informasi secara *real-time*, tetapi juga mudah dioperasikan oleh seluruh lapisan masyarakat, termasuk anak-anak, lansia, dan individu dengan keterbatasan fisik. Sistem yang ideal seharusnya menggabungkan berbagai fitur penting seperti pemantauan volume air, *notifikasi* otomatis, dan pengoperasian bebas sentuhan dalam satu kesatuan yang efisien dan ramah pengguna. Berdasarkan latar belakang tersebut, penelitian ini bertujuan untuk merancang dan mengimplementasikan prototipe *Smart water dispenser* berbasis IoT yang dilengkapi dengan sistem *monitoring* dan *notifikasi* realtime yang terintegrasi secara menyeluruh. Prototipe ini diharapkan mampu menjawab tantangan yang ada, meningkatkan kenyamanan dan aspek higienitas dalam penggunaan, serta memberikan nilai tambah dalam pengelolaan air minum yang lebih cerdas, efisien, dan selaras dengan kebutuhan era digital. Dispenser yang dikembangkan dalam penelitian ini disebut sebagai Smart Water Dispenser karena menggabungkan sensor, mikrokontroler ESP32, dan koneksi IoT untuk menghadirkan sistem monitoring dan pengendalian otomatis. Ciri ‘smart’ pada sistem ini tercermin dari kemampuannya untuk memantau volume air, suhu air panas/dingin, serta mendeteksi keberadaan gelas secara otomatis, kemudian mengambil keputusan seperti menghentikan pengeluaran air, mengirim notifikasi, atau mengatur suhu tanpa intervensi manual dari pengguna. Seluruh aksi ini dilakukan berdasarkan data real-time yang diterima dari sensor. Dengan demikian, sistem dapat dikategorikan sebagai perangkat cerdas yang adaptif dan responsif terhadap kondisi lingkungan dan kebutuhan pengguna. Pada tahap pengembangan saat ini, sistem telah dapat mengeksekusi aksi otomatis berbasis aturan

(rule-based automation), meskipun belum sampai pada tahapan pembelajaran pola pengguna (machine learning).

1.2 Rumusan Masalah

1. Bagaimana mengintegrasikan *notifikasi* status volume galon ke aplikasi *mobile* untuk memberikan peringatan secara *real-time*?
2. Bagaimana mengimplementasikan sistem pemantauan ketersediaan air panas dan dingin yang dapat dipantau melalui aplikasi *mobile*?
3. Bagaimana menciptakan sistem pengeluaran air otomatis melalui deteksi gelas?

1.3 Batasan Masalah

Format dalam penelitian ini, diperlukan pembatasan ruang lingkup agar tujuan penelitian dapat tercapai secara jelas, sistematis, dan terarah. Batasan-batasan yang digunakan dalam penelitian ini adalah sebagai berikut:

1. Penelitian berfokus pada perancangan dan implementasi prototipe *Smart water dispenser* berbasis *Internet of Things (IoT)* dengan fitur pemantauan otomatis volume galon, *monitoring* suhu air panas dan dingin, serta pengeluaran air otomatis berbasis deteksi gelas.
2. Sistem *monitoring* dan *notifikasi* yang dikembangkan terbatas pada aplikasi *mobile* berbasis *Android*, tanpa mencakup implementasi pada platform lain seperti web atau sistem operasi iOS.
3. Pengujian prototipe dilakukan di lingkungan kampus dengan lokasi terbatas, seperti laboratorium atau area tertentu yang sering dikunjungi oleh mahasiswa dan dosen, tidak mencakup implementasi skala luas di seluruh area kampus.
4. Sistem deteksi gelas yang digunakan terbatas pada gelas standar berbahan transparan dengan ukuran tertentu, tanpa mempertimbangkan variasi bentuk, warna, atau bahan gelas lainnya.
5. Penelitian tidak membahas secara mendalam aspek keamanan data dan privasi pengguna dalam komunikasi antara perangkat IoT dan aplikasi *mobile*, namun lebih difokuskan pada aspek teknis, fungsionalitas sistem, dan pengalaman pengguna secara umum.
6. Evaluasi sistem dilakukan dengan mengukur akurasi sensor volume air dan suhu, waktu respons *notifikasi* pada aplikasi *mobile*, serta efektivitas fitur pengeluaran air otomatis berbasis deteksi gelas dalam kondisi penggunaan yang normal di lingkungan kampus,

tanpa mempertimbangkan kondisi lingkungan ekstrem atau gangguan signifikan pada jaringan komunikasi.

1.4 Tujuan Penelitian

1. Mengembangkan sistem *notifikasi* berbasis aplikasi *mobile Android* yang mampu memberikan informasi secara *real-time* terkait status volume air dalam galon, guna mencegah keterlambatan pengisian ulang di lingkungan kampus.
2. Mengimplementasikan sistem *monitoring* suhu air panas dan dingin yang dapat dipantau secara langsung melalui aplikasi *mobile Android*, untuk meningkatkan kenyamanan pengguna dalam memperoleh air minum sesuai kebutuhan.
3. Merancang dan mengimplementasikan sistem otomatisasi pengeluaran air minum berbasis teknologi deteksi gelas, agar penggunaan dispenser lebih praktis, higienis, serta mudah diakses oleh seluruh kalangan pengguna di lingkungan kampus

1.5 Manfaat Penelitian

Manfaat penulisan laporan tugas akhir mengenai perancangan dan implementasi *Smart water dispenser* berbasis IoT ini, setelah pengembangan ide serta implementasi teknologi, diharapkan sebagai berikut:

1. Mengubah paradigma pengguna terhadap penggunaan dispenser air galon dari sekadar alat penyedia air minum menjadi perangkat cerdas yang mampu memberikan *notifikasi real-time* mengenai status volume galon melalui aplikasi *mobile*, sehingga dapat meningkatkan efisiensi, kenyamanan, serta mengurangi gangguan aktivitas akibat keterlambatan pengisian ulang galon.
2. Menjadi media teknologi yang mengedukasi pengguna tentang pentingnya memperhatikan kualitas air minum, terutama melalui fitur *monitoring* suhu air panas dan dingin secara *real-time*, agar pengguna selalu mendapatkan air minum yang nyaman dan aman dikonsumsi.
3. Mendorong inovasi dalam pengembangan teknologi dispenser air pintar yang tidak hanya efisien dan mudah digunakan, tetapi juga adaptif terhadap kebutuhan semua pengguna di lingkungan kampus, serta menginspirasi pengembangan produk serupa yang dapat diaplikasikan secara lebih luas dalam berbagai lingkungan publik maupun institusi.

1.6 Metodologi Penelitian

Penelitian ini menggunakan metodologi **Lean**, yang menekankan pada pengembangan produk secara iteratif, efisien, dan berorientasi langsung pada kebutuhan pengguna. Secara rinci, tahap-tahap penelitian dilakukan sebagai berikut:

1. *Build*

   Pada tahap ini fokus pada pembuatan *Minimum viable product (MVP)* dari sistem Dispenser. Dalam setiap siklus Build–Measure–Learn, pengguna tetap di lingkungan kampus seperti petugas OB yang rutin mengganti galon serta mahasiswa dan staf yang menggunakan dispenser secara langsung dilibatkan untuk mencoba prototipe dan memberikan masukan. Umpan balik dan hasil observasi penggunaan prototipe tersebut menjadi landasan dalam analisis kebutuhan, validasi fitur, serta perbaikan sistem pada tahap pengembangan berikutnya. MVP yang dikembangkan berupa prototipe dasar *Smart water dispenser* dengan fitur esensial yaitu *notifikasi* volume gallon secara real time, *monitoring* suhu, *system* deteksi gelas . Prototipe ini dibuat secepat mungkin untuk bisa segera diuji ke pengguna dengan investasi minimal.

2. *Measure*

   Setelah MVP selesai dibuat, pada tahap *Measure* dilakukan pengukuran utama untuk menilai penerimaan pengguna terhadap solusi yang ditawarkan.

3. *Learn*

   Pada tahap *Learn*, data yang terkumpul dari tahap pengukuran dianalisis untuk mendapatkan insight penting tentang perilaku pengguna dan performa sistem. Pembelajaran ini digunakan untuk validasi ide dan MVP.

Proses ini dilakukan secara berulang dimana setiap siklus menghasilkan versi produk yang lebih baik berdasarkan *feedback* nyata dari pengguna. Metode ini memfasilitasi pengembangan produk dengan cara yang lebih efisien,menurunkan kemungkinan terjadinya kegagalan, karena setiap fitur yang dirancang telah melalui proses validasi langsung dari pengguna.

BAB II
LANDASAN TEORI

Bab ini menjelaskan konsep dan teori yang mendasari perancangan *Smart water dispenser* berbasis IoT dengan sistem *monitoring* dan *notifikasi real-time*. Pembahasan mencakup sistem otomatisasi, *Internet of Things (IoT)*, dispenser pintar, sistem *monitoring* & *notifikasi*, teknologi pendukung, serta tinjauan penelitian terdahulu yang relevan.

2.1 Sistem Otomatisasi dan Relevansinya

Konsep Dasar Sistem Otomatisasi. Sistem otomatisasi merujuk pada penggunaan perangkat kendali (kontrol) untuk menjalankan suatu proses secara otomatis dengan minim intervensi manusia (Sutrisno). Dalam otomasi industri, misalnya, perangkat kontrol (seperti PLC, mikrokontroler, atau komputer) digunakan untuk mengendalikan dan memantau proses, mesin, atau peralatan sehingga operasi dapat berjalan sendiri secara berulang sesuai aturan yang ditetapkan (Sutrisno) Dengan otomatisasi, peran manusia tidak sepenuhnya dihilangkan namun berkurang, sehingga pekerjaan berlangsung lebih cepat dan konsisten. Sistem otomatis beroperasi berdasarkan logika terprogram; umumnya melibatkan sensor sebagai input, pengendali yang mengambil keputusan, serta aktuator sebagai output untuk melakukan aksi fisik.

Karakteristik dan Kelebihan Otomatisasi. Sistem otomatisasi dicirikan oleh kemampuannya bekerja secara mandiri, konsisten, dan cepat dibanding operasi manual. Beberapa kelebihan otomasi antara lain:

1. Meminimalisir kesalahan mesin tidak mudah lelah atau ceroboh sehingga tingkat kesalahan rendah.
2. Meningkatkan efisiensi dan produktivitas tugas berulang dapat dijalankan 24/7 dengan kecepatan konstan.
3. Keselamatan dan kenyamanan otomasi dapat menangani tugas berbahaya atau melelahkan, melindungi operator manusia.

Karakteristik lainnya meliputi kecepatan respon tinggi, akurasi yang dapat diandalkan, dan kemampuan operasi terjadwal sesuai pemrograman. Otomatisasi dapat bersifat *open-loop* (tanpa umpan balik) maupun *closed-loop* (dengan umpan balik sensor untuk penyesuaian realtime), tergantung kompleksitas sistem kendali.

Relevansi pada Dispenser Pintar. Penerapan sistem otomatisasi pada dispenser air minum membawa manfaat nyata bagi pengguna. Dispenser konvensional mengharuskan pengguna menekan tuas keran secara manual dan mengawasi volume air agar tidak meluber (Prasetyo, 2016). Otomatisasi dapat mengatasi kelemahan tersebut dengan mekanisme kendali otomatis: cukup dengan meletakkan gelas pada tempatnya, sensor akan mendeteksi keberadaan gelas dan memulai pengeluaran air secara otomatis. Ketika ketinggian air mencapai batas yang ditentukan, keran akan menutup dengan sendirinya dan mencegah luberan (Prasetyo, 2016). Contoh implementasi menunjukkan smart dispenser berbasis mikrokontroler berhasil mempermudah pengguna – air dituang otomatis hingga takaran pas tanpa perlu diawasi (Prasetyo, 2016). Dengan demikian, otomasi pada dispenser pintar meningkatkan kenyamanan, keamanan (mencegah tumpahan), dan efektivitas (takaran air konsisten). Hal ini sejalan dengan tren modern di mana berbagai alat rumah tangga ditingkatkan kecerdasannya melalui otomasi untuk memudahkan aktivitas sehari-hari.

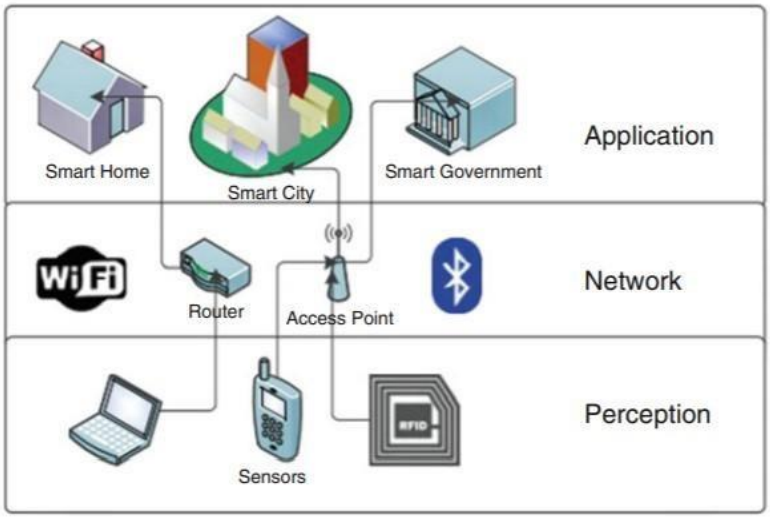
2.2 *Internet of Things (IoT)*

Konsep *Internet of Things (IoT)* adalah konsep di mana objek fisik sehari-hari dilengkapi dengan kemampuan komputasi, sensor, dan konektivitas sehingga dapat berkomunikasi melalui internet(Prasetya et al., 2022). IoT menggambarkan masa depan di mana setiap perangkat fisik dapat terhubung ke internet dan saling mengenali satu sama lain (Prasetya et al., 2022). Dengan IoT, berbagai perangkat (misalnya alat rumah tangga, sensor lingkungan, kendaraan, dsb) mampu mengirim dan menerima data melalui jaringan tanpa campur tangan manusia secara langsung. Data yang dikumpulkan dari perception *device*s tersebut kemudian diproses menjadi informasi berguna bagi pengguna atau sistem lainnya. Menurut Nugraha et al. (2018), IoT memungkinkan integrasi dunia fisik dengan sistem informasi digital sehingga tercipta sistem yang kontekstual dan otomatis dalam berbagai bidang (smart home, smart city, kesehatan, dsb)(Prasetya et al., 2022).

Arsitektur IoT. Secara umum, arsitektur IoT dapat dibagi menjadi beberapa layer (lapisan) yang menguraikan komponen dan alur data di dalamnya. Arsitektur tiga lapisan IoT yang paling sederhana terdiri atas:

1. Perception Layer (*Device* Layer) – yaitu lapisan fisik berupa perangkat IoT seperti sensor dan aktuator yang mengumpulkan data lingkungan (Burhan, Rehman, Khan, & Kim, 2018).

2. *Network Layer (Connectivity*) – lapisan jaringan yang menghubungkan perangkat IoT ke server/*cloud*, mencakup media komunikasi (Wi-Fi, seluler, dll) dan protokol data (Burhan, Rehman, Khan, & Kim, 2018).
3. *Application* Layer – lapisan aplikasi di mana pengguna berinteraksi dengan sistem, mencakup platform atau antarmuka yang menampilkan informasi dan memungkinkan kontrol (Burhan, Rehman, Khan, & Kim, 2018). Data mentah dari sensor di perception layer dikirim melalui network layer ke *cloud* untuk diolah, lalu hasilnya disajikan di *application* layer sebagai informasi atau layanan. Selain model tiga lapis, literatur lain mengembangkan arsitektur lima lapis yang lebih terperinci (menambahkan lapisan processing dan business) (Burhan, Rehman, Khan, & Kim, 2018), namun inti konsepnya serupa: IoT bergantung pada sensor, konektivitas jaringan, dan aplikasi yang bekerja terpadu.



Gambar 2.1 Gambar skema arsitektur IoT secara konseptual

Gambar 2.1 Ilustrasi tiga lapisan utama dalam arsitektur IoT (Burhan, Rehman, Khan, & Kim, 2018). Sensor dan perangkat *(Perception)* mengumpulkan data, jaringan *(Network)* mengirim data ke *cloud*, dan aplikasi (*Application*) menyajikan informasi ke pengguna.

Komponen Hardware dan Software IoT. Implementasi IoT melibatkan kombinasi elemen perangkat keras dan perangkat lunak. Perangkat keras IoT meliputi sensor (alat untuk mendeteksi perubahan fisik, misal suhu, jarak, aliran air), mikrokontroler atau embedded *system* (misal Arduino, ESP8266) yang memproses pembacaan sensor, serta aktuator (misal motor, pompa, valve) untuk melakukan aksi fisik. Perangkat keras ini umumnya dilengkapi modul komunikasi (seperti modul Wi-Fi atau modul radio lainnya) agar dapat terhubung ke internet. Sementara itu, perangkat lunak IoT mencakup firmware yang ditanam di

mikrokontroler (program yang mengatur logika kerja perangkat IoT) dan platform server/*cloud* dan aplikasi. Platform *cloud* (contoh: *Firebase*, AWS IoT) berfungsi menerima, menyimpan, dan mengolah data yang dikirim perangkat IoT, sedangkan aplikasi (web atau *mobile*) menjadi antarmuka bagi pengguna untuk memonitor data secara *real-time* maupun mengirim perintah kontrol ke perangkat.

Konektivitas dan Protokol IoT. Agar perangkat IoT dapat bertukar data, dibutuhkan media konektivitas dan protokol komunikasi yang tepat. Wi-Fi adalah salah satu teknologi jaringan nirkabel paling umum digunakan untuk menghubungkan perangkat IoT ke internet melalui jaringan lokal. Wi-Fi (Wireless Fidelity) bekerja menggunakan gelombang radio pada standar IEEE 802.11 untuk menyediakan akses internet nirkabel berkecepatan tinggi(Liu, 2014). Dengan modul Wi-Fi tertanam (contoh: ESP8266 pada NodeMCU), perangkat IoT dapat langsung terhubung ke router Wi-Fi dan mengakses internet. Selain Wi-Fi, perangkat IoT juga dapat menggunakan koneksi seluler (GSM/4G), Bluetooth Low Energy, atau protokol LoRa, tergantung kebutuhan jangkauan dan daya.

Di lapisan protokol aplikasi, MQTT dan HTTP adalah contoh protokol data yang lazim dipakai. Message Queuing Telemetry Transport (MQTT) adalah protokol komunikasi mesinke-mesin yang ringan dan efisien, dirancang khusus untuk perangkat dengan sumber daya dan bandwidth terbatas((Jara Ochoa, Peña, Ledo Mezquita, Gonzalez, & Camacho-Leon, 2023). MQTT menerapkan pola publish-subscribe melalui broker, sehingga cocok untuk pengiriman data sensor secara *real-time* dengan overhead kecil. Contohnya, sensor pintar dapat mengirim (“publish”) pembacaan data ke suatu topic di broker MQTT, dan aplikasi yang berlangganan (“subscribe”) topic tersebut akan menerima data secara instan. Kelebihan MQTT antara lain ringan (pesan kontrol sekecil 2 byte, header minim) dan andal (mendukung Quality of Service untuk memastikan pengiriman) (Jara Ochoa, Peña, Ledo Mezquita, Gonzalez, & Camacho-Leon, 2023).

Sementara itu, HTTP (HyperText Transfer Protocol) adalah protokol lapisan aplikasi yang digunakan luas di web, dan dapat pula dimanfaatkan dalam komunikasi IoT. HTTP mendefinisikan aturan pertukaran data antara client-server, umumnya dalam format requestresponse (Jara Ochoa, Peña, Ledo Mezquita, Gonzalez, & Camacho-Leon, 2023). Misalnya, perangkat IoT dapat bertindak sebagai client yang mengirim HTTP POST ke server web/*cloud* untuk mengunggah data sensor, atau melakukan HTTP GET untuk mengambil perintah. HTTP adalah teknologi dasar komunikasi data di Internet dan WWW (Naik, 2017). namun relatif lebih berat dibanding MQTT (karena tiap transaksi membawa header dan

membutuhkan inisiasi koneksi TCP). Pilihan penggunaan MQTT atau HTTP tergantung kasus: untuk update data berkala cepat dan ringan, MQTT sering lebih disukai, sedangkan HTTP cocok untuk integrasi dengan layanan web yang sudah ada (RESTful API). Terkadang IoT memadukan keduanya – misal, perangkat mengirim data ke *Firebase* (yang berbasiskan HTTP/REST) namun *notifikasi* dikirim via MQTT. Yang jelas, protokol komunikasi adalah komponen vital IoT karena menjembatani perangkat fisik dengan layanan internet.

Dalam konteks dispenser pintar berbasis IoT, biasanya perangkat akan menggunakan *Wifi* sebagai koneksi ke internet (misalnya melalui hotspot rumah), lalu mengirim data ke platform *cloud* menggunakan protokol seperti MQTT atau HTTP sesuai desain sistem. Data seperti volume air tersisa, status dispenser, atau log konsumsi air dikirim ke *cloud* untuk disimpan dan ditampilkan bagi pengguna. Sebaliknya, perintah dari pengguna (misal via aplikasi *mobile*) dapat dikirim balik ke perangkat untuk mengontrol aktuator (misal membuka katup). Kombinasi infrastruktur IoT inilah yang memungkinkan dispenser pintar berfungsi cerdas dan terhubung secara online.

2.3 *Smart water dispenser*

Definisi Dispenser Air dan Evolusinya. Dispenser air minum adalah alat elektronik yang digunakan untuk menyajikan air minum (dalam galon) dengan suhu tertentu – bisa suhu normal, panas, ataupun dingin(Prasetya et al., 2022). Dispenser memudahkan orang mengambil air minum dari galon ke gelas secara praktis, tanpa harus mengangkat dan menuang sendiri(Prasetya et al., 2022). Umumnya dispenser dilengkapi keran untuk air biasa dan panas, serta elemen pemanas/pendingin. Seiring perkembangan kebutuhan, teknologi dispenser terus mengalami inovasi agar kinerjanya lebih efektif dan efisien(Prasetya et al., 2022). Awalnya dispenser hanya perangkat mekanis sederhana, namun “dispenser zaman sekarang sudah dilengkapi banyak fitur dan sensor pendukung” untuk mempercanggih fungsinya (Kurniawan & Jamaaluddin, 2023). Misalnya, dispenser modern pada kulkas memiliki fitur penyaring air, pengatur suhu digital, hingga konektivitas ke aplikasi. Inovasi teknologi dispenser bertujuan meningkatkan kemudahan pelayanan air minum bagi pengguna sekaligus menambah nilai tambah (misal fitur higienitas, penghitungan konsumsi, dsb).

Prinsip Kerja Dispenser Pintar. *Smart water dispenser* mengintegrasikan sensor, aktuator, dan sistem kontrol otomatis untuk mengeluarkan air secara cerdas sesuai perintah atau kondisi terdeteksi. Secara umum, prinsip kerjanya sebagai berikut: ketika pengguna ingin mengambil air, sensor pada dispenser mendeteksi kehadiran wadah (gelas/botol) dan/atau input

pengguna (tombol atau perintah melalui aplikasi). Sensor yang lazim digunakan adalah sensor jarak (misal ultrasonik atau infrared) untuk mendeteksi gelas dan mengukur ketinggian air di dalamnya. Mikrokontroler akan memproses sinyal sensor tersebut dan mengaktifkan aktuator berupa pompa air atau solenoid valve untuk mulai mengalirkan air dari galon ke gelas (Ramdani, et al., 2024). Air terus mengalir hingga sensor membaca bahwa volume/ketinggian air telah mencapai batas yang ditentukan, lalu mikrokontroler mematikan pompa atau menutup katup valve secara otomatis (Ramdani, et al., 2024). Dengan mekanisme umpan balik ini, sistem memastikan air yang dituang sesuai takaran tanpa meluap. Komponen kunci pada dispenser pintar antara lain:

1. Sensor pendeteksi gelas & level air

   Contoh: Sensor Ultrasonik HC-SR04 yang ditempatkan di atas tatakan gelas untuk mendeteksi jarak permukaan air (Ramdani, et al., 2024). Alternatif lain, sensor IR jarak atau kombinasi photodiode dan laser bisa digunakan untuk mendeteksi tinggi permukaan cairan. Beberapa desain juga menggunakan flow sensor pada jalur air untuk mengukur volume yang telah dialirkan, sebagai konfirmasi takaran yang akurat.

2. Mikrokontroler / Unit kontrol

   Berupa board seperti Arduino Uno/ATMega atau SoC ESP8266/ESP32 yang menjadi otak sistem. Mikrokontroler membaca input sensor secara terus-menerus dan mengambil keputusan kendali (misal menyalakan atau mematikan pompa) berdasarkan program yang ditanamkan.

3. Aktuator (Katup & Pompa)

   Electric valve atau solenoid valve digunakan sebagai keran elektronik yang buka/tutup aliran air secara otomatis (Prasetyo, 2016). Beberapa dispenser juga memakai pompa air elektrik (biasanya tipe diaphragm pump) untuk mendorong air keluar dari galon. Mikrokontroler mengendalikan aktuator ini via driver (transistor atau modul relay) sesuai logika kendali. Pada rancangan tertentu, buzzer ditambahkan sebagai alarm indikator (misal bunyi buzzer ketika pengisian selesai) (Prasetyo, 2016)

4. Interface pengguna

   Pada dispenser pintar konvensional, interface bisa berupa tombol untuk memilih ukuran takaran (misal tombol “Small (150ml)”, “Medium (250ml)”, “Large (330ml)”). Namun pada sistem IoT, interface dapat berada di aplikasi *mobile*, di mana pengguna mengatur melalui smartphone, sementara di unit dispenser mungkin hanya ada lampu indikator status dan sensor yang bekerja otomatis.

Dengan komponen tersebut, dispenser pintar bekerja secara embedded. Sebagai ilustrasi: Prasetyo (2016) merancang dispenser otomatis berbasis ATMega8 yang menggunakan sensor inframerah untuk mendeteksi keberadaan gelas dan mengukur ketinggian permukaannya, serta solenoid valve untuk membuka/menutup keran. Pengguna cukup meletakkan cangkir, kemudian air keluar otomatis; saat ketinggian air mencapai titik tertentu, sensor mendeteksi dan keran menutup sendiri. Hasilnya, pengguna tidak perlu repot menekan keran maupun khawatir air meluap karena sistem kontrol memastikan volume tepat. Tren dan Inovasi Dispenser Pintar Terkini. Inovasi pada dispenser pintar terus berkembang mengikuti kemajuan IoT dan kebutuhan pengguna. Salah satu tren adalah integrasi konektivitas internet dan fitur cerdas lainnya ke dalam dispenser, menjadikannya bagian dari ekosistem smart home. Beberapa inovasi terkini antara lain:

1. Konektivitas IoT & Aplikasi *Mobile*

   Dispenser modern dapat terhubung ke aplikasi smartphone via Wi-Fi. Hal ini memungkinkan fitur remote *monitoring* (pengguna bisa memantau sisa air galon, temperatur, dsb dari aplikasi) dan remote control (misal memerintahkan dispenser mengisi air melalui smartphone). Devin et al. (2022) mengembangkan dispenser pintar terhubung *Firebase* yang mampu mengingatkan pengguna untuk minum serta memesan ulang galon secara otomatis ketika hampir habis(Prasetya et al., 2022). Data dari sensor dikirim ke database realtime, lalu aplikasi *mobile* menampilkan statistik konsumsi air harian pengguna secara *real-time* dan mengirim *notifikasi* pengingat.

2. Pengisian Otomatis Berdasarkan Volume

   Fitur ini memungkinan pengguna memilih takaran volume tertentu (misal 600 mL, 1 liter) dan dispenser akan menghentikan aliran tepat pada volume tersebut. Implementasi teknisnya dapat menggunakan sensor aliran (flow meter) untuk menghitung volume yang keluar (Kurniawan & Jamaaluddin, 2023) merancang dispenser pintar yang memiliki fitur pengisian sesuai volume botol yang dibutuhkan, di mana setiap pilihan volume sudah ditentukan harganya (Kurniawan & Jamaaluddin, 2023). Sistem ini diterapkan untuk dispenser isi ulang air minum (mirip vending machine): pengguna menempelkan kartu, memilih volume (misal 600 mL), lalu air otomatis dialirkan sejumlah itu dan berhenti sendiri – volume terjual tercatat, saldo kartu berkurang.

3. Pembayaran Cashless dengan Kartu/QR

   Inovasi lainnya adalah integrasi sistem pembayaran elektronik. Pada dispenser isi ulang umum, transaksi biasanya manual (koin/tunai), namun dispenser pintar dapat dilengkapi

pembaca RFID atau pemindai QR code untuk pembayaran digital. Contohnya, smart dispenser rancangan (Kurniawan & Jamaaluddin, 2023) menggunakan RFID card prabayar; pengguna cukup menempelkan kartu RFID berisi saldo, sensor RFID membaca data kartu dan memverifikasi saldo ke server sebelum mengizinkan pengeluaran air (Kurniawan & Jamaaluddin, 2023) Pendekatan ini meningkatkan kemudahan dan keamanan (mengurangi kontak fisik & uang tunai).

4. *Monitoring* Kualitas Air & Sensor Tambahan

   Beberapa inovasi mulai memasukkan sensor kualitas air (TDS sensor, temperatur) untuk memantau bahwa air yang dispensed tetap layak minum. Misalnya dispenser pintar tertentu bisa mendeteksi suhu air dan menampilkan di aplikasi, atau memberi *notifikasi* jika kualitas menurun (filter perlu diganti).

5. Integrasi dengan Asisten Cerdas

   Walau masih baru, ada konsep dispenser yang dapat diperintah via voice assistant (Alexa/Google Home). Misal perintah suara: “Dispense 1 cup of water”, dan dispenser IoT akan melaksanakannya. Ini memanfaatkan API IoT yang terhubung dengan platform asisten suara.

Tren-tren di atas menunjukkan bahwa dispenser pintar berevolusi dari sekadar alat penyalur air menjadi perangkat cerdas, terhubung, dan interaktif. Tujuannya selain mempermudah pengguna perorangan, juga membuka peluang model bisnis baru. Sebagai contoh, dispenser isi ulang otomatis (dengan pembayaran digital) dapat ditempatkan di tempat umum sebagai *vending machine* air minum, menawarkan harga lebih murah dan mengurangi sampah botol plastik (Kurniawan & Jamaaluddin, 2023). Inovasi tersebut tak lepas dari dukungan IoT dan sensor, yang dibahas lebih lanjut pada bagian berikutnya.

2.4 Sistem *Monitoring* dan *Notifikasi Real-time*

Monitoring Waktu Nyata (*Real-time*). Sistem *monitoring real-time* adalah sistem pemantauan yang dapat menampilkan kondisi atau data sensor secara seketika (nyaris tanpa penundaan) kepada pengguna atau sistem lain. Dalam konteks IoT, *monitoring real-time* dicapai dengan mengirim data dari perangkat ke server/*cloud* secara sinkron dan kontinu, sehingga setiap perubahan di perangkat segera tercermin di sisi pemantau. Hal ini penting untuk dispenser pintar agar pengguna dapat mengetahui status perangkat (misal volume air tersisa, jumlah air yang telah diminum hari ini, dsb) saat itu juga. *Real-time* bukan berarti instant sempurna, tetapi latensi sangat kecil (umumnya milidetik hingga beberapa detik).

Parameter seperti delay dan throughput digunakan mengukur performa realtime *system*: delay adalah waktu tempuh data dari pengirim ke penerima(Prasetya et al., 2022), sedangkan throughput adalah kecepatan pengiriman data (bps)(Prasetya et al., 2022). Dalam pengujian sebuah dispenser IoT, misalnya, didapat delay rata-rata ~169 ms (dikategorikan "Baik" menurut standar TIPHON) dan throughput ~4148 kbps ("Sangat Bagus")(Prasetya et al., 2022), sehingga cukup memadai untuk *monitoring* waktu nyata.

Untuk mewujudkan *monitoring* realtime, arsitektur perangkat lunak banyak memanfaatkan mekanisme sinkronisasi berbasis event. Salah satu caranya adalah dengan menggunakan database waktu nyata di *cloud* yang mendukung push update ke klien. Contoh populer adalah *Firebase Realtime Database* dari Google. *Firebase* Realtime DB adalah database *NoSQL* berbasis *cloud* yang menyimpan data dalam format JSON dan menyinkronkannya secara realtime ke setiap klien yang terhubung (Mitu, Vassilev, & Tabany, 2021). Semua pengguna/aplikasi yang terhubung berbagi satu instance database; ketika data pada database diperbarui oleh salah satu pihak (misal perangkat IoT mengirim data sensor baru), perubahan tersebut secara otomatis didorong ke klien lain seketika (Mitu, Vassilev, & Tabany, 2021). Dengan demikian, *Firebase* sangat sesuai untuk *monitoring* IoT: perangkat cukup update nilai-nilai (contoh: volume_galon, waktu_dispense, dsb) di database, lalu aplikasi *mobile* yang terhubung akan menerima pembaruan dalam hitungan *real-time* (melalui listener yang berjalan di background) (Mitu, Vassilev, & Tabany, 2021). Kelebihan lain, *Firebase* dapat tetap sinkron meski klien sempat offline – data akan tersimpan lokal dan dikirim saat online kembali (offline persistence) (Chou, Dewabharata, Bayu, Cheng, & Zulvia, 2022). Arsitektur berbasis ini mengurangi kebutuhan polling berulang, sehingga lebih efisien.

Pada sistem yang kita rancang, modul mikrokontroler dengan Wi-Fi akan terhubung ke *Firebase* dan mengirim data status secara periodik atau event-driven (setiap kali ada aksi). Data yang disimpan misalnya: volume_air_terkini, status_galon_penuh/kosong, log_pengeluaran_air, dll. Aplikasi pengguna kemudian membaca database ini. Karena sifat realtime-nya, pengguna seolah "melihat langsung" kondisi dispenser. Misalnya, ketika galon hampir habis, sensor mendeteksi dan perangkat mengubah nilai "status_galon" menjadi "hampir kosong" di database; dalam waktu kurang dari sedetik, aplikasi pengguna menerima update ini dan bisa menampilkan *notifikasi* peringatan.

1. *Firebase Realtime Database* dipilih karena kemudahan integrasinya dengan aplikasi *mobile* dan scalability di sisi *cloud* tanpa harus mengelola server sendiri. Alternatif lain untuk *monitoring* realtime adalah menggunakan protokol MQTT (dengan broker publish-

subscribe), namun itu memerlukan server MQTT terpisah. *Firebase* menyederhanakan dengan pendekatan database-as-a-service yang developer-friendly.

2. Penggunaan *React Native Expo* untuk Aplikasi dan *Notifikasi*. Untuk menampilkan data *monitoring* dan memberikan *notifikasi* ke user, dikembangkan aplikasi smart dispenser pada smartphone. Proyek ini memilih menggunakan *React Native* dengan *Expo* sebagai framework pengembangan aplikasi *mobile*. *React Native* adalah framework JavaScript yang digunakan untuk membangun aplikasi *mobile* lintas platform (*Android* dan iOS) dengan basis React (Hutri, 2023). *React Native* memungkinkan penulisan satu basis kode JS yang dikonversi menjadi komponen native di *Android*/iOS, sehingga efisien dalam pengembangan. *Expo* sendiri merupakan seperangkat alat (toolkit) yang dibangun di atas *React Native* untuk mempermudah proses development dan penyebaran aplikasi (Hutri, 2023). Dengan *Expo*, developer tidak perlu menyentuh konfigurasi native yang rumit (*Android* Studio/Xcode dapat diminimalkan); *Expo* menyediakan berbagai pustaka dan layanan bawaan. Menurut Liantriana (2018), *Expo* menyediakan tools, library, dan services sehingga pengembangan aplikasi *React Native* dapat lebih cepat – banyak modul fitur (kamera, peta, push notification, dsb) sudah disediakan out-of-the-box oleh *Expo* (Hutri, 2023). Cara kerjanya, aplikasi dikodekan dalam JavaScript React seperti biasa, lalu *Expo* CLI memfasilitasi running di *device* (melalui scanning QR) dan *build*ing. Dalam sistem ini, aplikasi *mobile* berfungsi sebagai *User interface* utama bagi pengguna dispenser. Fitur-fitur meliputi: dashboard *monitoring* (menampilkan secara realtime data volume air galon, suhu (jika ada sensor), jumlah air diminum hari ini, dsb), tombol kontrol (misal tombol Dispense 250ml jika diizinkan remote), serta pengaturan *notifikasi*. Kelebihan *React Native Expo* adalah dapat memanfaatkan layanan *notifikasi* push dengan mudah. Push Notification adalah pesan *notifikasi* yang dikirim ke perangkat user bahkan saat aplikasi tidak aktif, sering digunakan untuk alert penting. *Expo* mendukung fitur Push Notifications melalui layanan *cloud*-nya sendiri tanpa perlu konfigurasi rumit di tiap OS (Hutri, 2023). Dalam implementasi, aplikasi kita dapat mengirim token perangkat ke server, lalu menggunakan *Expo* Notifications API atau *Firebase Cloud* Messaging untuk memicu *notifikasi*.
3. *Notifikasi Real-time*. Dengan kombinasi *Firebase* dan aplikasi *Expo*, *notifikasi* dapat dikirim secara *real-time* berdasarkan kondisi tertentu. Contohnya, ketika flow sensor membaca galon kosong (nilai volume jatuh di bawah ambang), mikrokontroler mengupdate status di *Firebase*; listener di aplikasi menangkap perubahan ini dan memicu

fungsi untuk mengirim *notifikasi* lokal ke pengguna: “Galon hampir habis, segera ganti untuk kelancaran penggunaan.”. Untuk pengingat minum, aplikasi bisa menjalankan penghitungan interval waktu sejak user terakhir minum dan menampilkan *notifikasi*: “Sudah 2 jam, saatnya minum air untuk hidrasi.” *Notifikasi* juga dapat push dari server – misal terjadwal setiap jam tertentu mengingatkan target harian.

Intinya, sistem *notifikasi* memastikan pengguna selalu mendapatkan informasi kritis secara proaktif tanpa harus terus-menerus membuka aplikasi. Hal-hal yang dapat diberitahukan meliputi: galon perlu diganti, jumlah air harian diminum (misal “Anda sudah minum 8 gelas hari ini, target tercapai!”), adanya kesalahan pada dispenser (jika terdeteksi anomali), atau konfirmasi pemesanan galon baru berhasil. Semua ini meningkatkan interaktivitas dan manfaat dispenser pintar. Pada akhirnya, kombinasi *monitoring* realtime dan *notifikasi* menjadikan pengalaman pengguna lebih informatif, aman, dan terarah, sejalan dengan tujuan IoT untuk menciptakan sistem yang responsif terhadap konteks pengguna secara langsung.

2.5 Teknologi dan Metodologi Pendukung

Bagian ini membahas komponen teknologi utama serta metode teknis dalam pengembangan smart dispenser berbasis IoT, meliputi pemrosesan sinyal sensor, komunikasi data mikrokontroler dengan cloud, serta pengendalian perangkat keras melalui pemrograman Arduino.

Pemrosesan Sinyal Sensor. Smart dispenser memanfaatkan berbagai jenis sensor untuk memperoleh informasi lingkungan. Dua sensor utama yang digunakan pada sistem ini adalah sensor proximity (untuk mendeteksi keberadaan gelas dan level air) serta sensor aliran (flow sensor) yang berfungsi mengukur jumlah air yang dikeluarkan.

1. Sensor Jarak Ultrasonik

   Sensor ultrasonik (misalnya HC-SR04) bekerja berdasarkan prinsip pantulan gelombang suara frekuensi tinggi. Secara sederhana, sensor ini memancarkan gelombang ultrasonik dan mengukur waktu yang dibutuhkan gelombang tersebut untuk memantul kembali setelah mengenai suatu objek (Prasetya et al., 2022). Modul HC-SR04 memiliki pemancar yang mengirimkan gelombang suara ultrasonik (tidak terdengar oleh manusia) dan penerima yang menangkap pantulan gelombang tersebut. Jarak objek dihitung menggunakan rumus $d = \frac{1}{2} v \times t$, di mana v merupakan kecepatan suara (~340 m/s) dan t adalah waktu tempuh gelombang pergi-pulang.

Pada aplikasi dispenser, sensor ultrasonik dipasang menghadap ke bawah tepat di atas mulut gelas. Ketika gelas kosong diletakkan, jarak antara sensor dan dasar gelas bersifat tetap. Seiring air mengisi gelas, jarak antara sensor dan permukaan air menjadi semakin pendek. Mikrokontroler membaca perubahan jarak ini secara real-time. Jika jarak terukur sudah mencapai batas tertentu (menandakan volume air sudah sesuai kebutuhan), sistem secara otomatis memberikan sinyal untuk menghentikan aliran air.

Sensor ultrasonik umumnya memiliki tingkat akurasi sekitar ±1 cm dan jangkauan deteksi antara 2 hingga 400 cm (Prasetya et al., 2022), sehingga cukup andal untuk mendeteksi ketinggian air dalam gelas. Tantangan penggunaan sensor ini meliputi kemungkinan adanya noise serta sudut deteksi yang terbatas. Namun, dengan pemasangan yang tepat (tegak lurus terhadap permukaan cairan) dan kalibrasi yang baik, sensor ini terbukti efektif digunakan dalam sistem dispenser otomatis (Prasetya et al., 2022).

2. Sensor Aliran Air (*Water Flow* Sensor)

Untuk mengukur volume air yang keluar, biasanya digunakan sensor flow berbasis efek Hall, seperti YF-S201. Sensor ini merupakan komponen pipa yang di dalamnya terdapat baling-baling; saat air mengalir, baling-baling tersebut berputar dan magnet kecil di dalamnya menghasilkan pulsa listrik setiap kali terjadi satu putaran melalui efek Hall (Chen, Shen, Wu, Lee, & Chen, 2022). Rangkaian sensor akan mengeluarkan pulsa digital (misal ke pin interrupt Arduino) dengan frekuensi yang sebanding dengan laju aliran air.

Berdasarkan data teknis YF-S201, sensor ini menghasilkan sekitar 450 pulsa per liter, di mana satu pulsa mewakili sekitar 2,2 mL aliran air (Chen, Shen, Wu, Lee, & Chen, 2022). Mikrokontroler akan menghitung jumlah pulsa dalam satuan waktu (untuk mengetahui laju aliran) atau mengakumulasikan jumlah pulsa (untuk menghitung total volume). Dengan sensor ini, sistem dapat mengukur volume air secara cukup akurat, dengan tingkat akurasi tipikal ±10% tanpa kalibrasi, dan dapat ditingkatkan lagi melalui proses kalibrasi (Chen, Shen, Wu, Lee, & Chen, 2022). Pemrosesan sinyal flow sensor biasanya menggunakan interrupt service routine di Arduino: setiap kali pulsa terdeteksi, counter bertambah. Program kemudian menghitung volume = (jumlah pulsa * faktor kalibrasi). Jika target volume telah tercapai, kontrol mematikan pompa/valve. Contoh penerapan: Wahyuningsih et al. (2019) menggunakan sensor YF-S201 berbasis Arduino untuk alat pengukur debit air dan menunjukkan sensor ini efektif mendeteksi aliran secara *real-time* (dengan sedikit error yang bisa dikalibrasi). Jadi, flow sensor memberikan

umpan balik kuantitatif bagi dispenser pintar, memastikan volume keluaran sesuai perintah.

3. Sensor Lain

Selain kedua di atas, ada pula sensor pelengkap seperti sensor level air galon (bisa berupa float switch sederhana yang mengapung di galon untuk mendeteksi level rendah, atau sensor ultrasonik yang mengukur ketinggian air dalam galon). Pada sistem ini digunakan float sensor sebagai saklar level: ketika air galon turun di bawah titik tertentu, saklar terbuka/tertutup sebagai tanda galon kosong. Sensor ini dipasang di galon atau wadah reservoir dispenser. Sinyalnya (HIGH/LOW) dibaca mikrokontroler untuk kemudian mengirim *notifikasi* "Galon kosong". Juga terdapat sensor suhu jika ingin memantau suhu air panas/dingin, dan push button sebagai input manual alternatif (misal tombol cancel).

Pemrosesan sinyal-sinyal di atas dilakukan di sisi mikrokontroler. Data analog (jika ada, misal sensor level tipe analog) akan di-sampling oleh ADC mikrokontroler, sedangkan sinyal digital (ultrasonik echo, pulsa flow, tombol) dibaca via GPIO atau interrupt. Agar pembacaan akurat, biasanya ditambahkan proses filtering (misal mengambil rata-rata beberapa pembacaan ultrasonik untuk menghindari spike error) dan calibration (mengkonversi hitungan pulsa ke volume menggunakan faktor kalibrasi empiris). Diagram logika pemrosesan sensor ini kemudian diintegrasikan dalam kode program.

Jembatan antara perangkat keras dispenser di lapangan dan aplikasi/cloud di internet dicapai melalui modul komunikasi jaringan. Sistem ini menggunakan mikrokontroler dengan modul Wi-Fi terintegrasi, yaitu NodeMCU ESP8266 yang dipasangkan dengan Arduino Mega (konfigurasi dual MCU) atau langsung menggunakan ESP32 tunggal. NodeMCU (ESP8266) merupakan platform IoT open source berbasis chip ESP8266 (Espressif) yang telah dilengkapi modul Wi-Fi 2,4 GHz terintegrasi (Prasetya et al., 2022). Dengan modul ini, perangkat dapat langsung terhubung ke hotspot Wi-Fi dan bertukar data secara daring.

Berdasarkan dokumentasi, ESP8266 memiliki TCP/IP stack internal dan dapat diprogram (menggunakan Lua atau Arduino C), sehingga sering dijuluki 'Arduinonya ESP' (Prasetya et al., 2022). Sementara itu, ESP32 merupakan versi lebih baru yang juga memiliki Wi-Fi dan Bluetooth serta kemampuan prosesor yang lebih tinggi. Secara prinsip, modul Wi-Fi ini berperan sebagai transceiver data IoT: menerima perintah dari cloud sekaligus mengirimkan data sensor ke cloud melalui internet.

Ada beberapa metode komunikasi yang dapat digunakan oleh mikrokontroler ke *cloud*:

1. HTTP Request

   Mikrokontroler melakukan HTTP POST/PUT ke endpoint server (misal REST API atau endpoint *Firebase*) dengan payload data (misal format JSON). Untuk *Firebase* Realtime DB, dapat digunakan library *Firebase* Arduino yang menghandle komunikasi HTTPS. Misal, saat volume berubah, perangkat memanggil *Firebase*.set("/dispenser/volume", nilai_baru). Pendekatan ini mudah karena banyak contoh dan kompatibel dengan firewall umum (port 80/443). Kekurangannya, HTTP overhead agak besar dan perlu penanganan koneksi.

2. MQTT Publish

   Mikrokontroler berperan sebagai MQTT client yang terkoneksi ke broker (misal test.mosquitto.org atau broker *cloud* IoT). Perangkat publish topik "dispenser/status" dengan payload tertentu. Pihak aplikasi atau server subscribe topik ini. Kelebihan MQTT adalah ringan dan realtime, tapi memerlukan broker (bisa publik atau selfhosted).

3. *Firebase* SDK

   Khusus untuk *Firebase*, disediakan SDK C++/Arduino yang abstraksi tinggi. Perangkat bisa langsung memanggil fungsi update tanpa memikirkan protokol yang mendasari (SDK yang akan mengelola koneksi persisten WebSocket untuk sync realtime). Ini memudahkan implementasi; developer cukup menggunakan perintah seperti *Firebase*.push() dalam loop program.

Pada penelitian ini, integrasi library Firebase Arduino dimanfaatkan agar mikrokontroler ESP8266 dapat terhubung sebagai klien Firebase. Saat perangkat pertama kali dinyalakan, modul Wi-Fi akan terkoneksi ke SSID dan password Wi-Fi lokal. Selanjutnya, perangkat melakukan inisialisasi koneksi ke server Firebase menggunakan kredensial yang diperlukan (database URL, secret/key). Setelah berhasil terhubung, perangkat siap untuk mengirim atau menulis data.

Skema pengiriman data umumnya bersifat event driven, yaitu update data dilakukan saat terjadi suatu peristiwa (misalnya terjadi perubahan signifikan volume air atau proses pengisian selesai). Untuk data periodik (misal pembacaan level galon), perangkat dapat mengirim data secara berkala (misal setiap 1 menit) guna memperbarui nilai di database. Karena Firebase Realtime Database bersifat sinkronisasi, kita juga dapat mengaktifkan stream untuk menerima data, misalnya jika admin aplikasi mengirim perintah membuka katup, data perintah dikirim ke node 'command' di database, perangkat IoT menerima event tersebut dan mengeksekusi perintah. Dengan demikian, komunikasi data berlangsung dua arah.

Sebuah penelitian oleh Hendra dkk. (2017) menyebut bahwa penggunaan mikrokontroler ESP dengan Wi-Fi memudahkan pemrosesan data secara online melalui server (Kurniawan & Jamaaluddin, 2023). Ini terbukti dalam rancangan kita: ESP8266 yang include *Wifi* mampu menjaga perangkat selalu online dan sinkron dengan database *cloud*. Tantangan yang perlu diperhatikan adalah menjaga koneksi tetap hidup (perlu reconnect jika Wi-Fi putus) dan keamanan data (menggunakan HTTPS atau autentikasi *Firebase*).

Pengendalian Perangkat Keras melalui Pemrograman Arduino IDE. Semua logika kontrol dispenser ditanamkan dalam bentuk program pada mikrokontroler. Perancangan perangkat lunak sistem ini menggunakan Arduino IDE dan bahasa C/C++ yang umum untuk mikrokontroler (Kurniawan & Jamaaluddin, 2023) Arduino IDE dipilih karena sifatnya opensource, sederhana, namun powerful dan flexible untuk berbagai tipe mikrokontroler (Kurniawan & Jamaaluddin, 2023) Kode program ditulis mengikuti flowchart sistem kendali dispenser, yang didasari alur logika manusia dalam skenario pembelian/pengambilan air (Kurniawan & Jamaaluddin, 2023) Contoh alur sederhananya:

1. Inisialisasi: atur pin-pin (sensor, relay, dsb), inisialisasi Wi-Fi dan *Firebase*, set variabel awal.
2. Idle: tunggu hingga ada gelas terdeteksi atau perintah pengeluaran diterima.
3. Deteksi Gelas: jika sensor ultrasonik mendeteksi jarak < ambang (artinya ada gelas), lanjut.
4. Mulai Pengisian: aktifkan pompa (atau buka valve) – air mulai mengalir.
5. *Monitoring* saat Pengisian: terus pantau sensor ultrasonik atau hitung pulsa flow; jika ketinggian/volume tercapai atau tombol stop ditekan, hentikan pengisian.
6. Selesai: matikan pompa/katup, bunyikan buzzer (jika ada), update data log (volume dispensed, waktu) ke *cloud*.
7. Kembali ke idle menunggu event berikutnya.

Kode ini dijalankan dalam loop utama Arduino yang terus berulang cepat. Untuk menangani beberapa tugas “simultan” (misal membaca sensor sambil menghitung waktu), kadang digunakan mekanisme interrupt (misal untuk flow sensor pulse) dan millis() timing alih-alih delay(), agar program tetap responsif.

Arduino IDE mempermudah pengecekan logika dengan fitur serial monitor (untuk debugging) dan kemampuan modifikasi cepat jika ada perubahan spesifikasi. Misalnya, jika di tengah pengembangan diperlukan mengubah ambang jarak sensor atau mengubah harga/volume, cukup menyesuaikan konstanta di program dan mengunggah ulang ke board

(Kurniawan & Jamaaluddin, 2023). Gusrion (2018) menyebutkan bahwa keunggulan platform Arduino adalah kemudahan dalam melakukan penyesuaian logika maupun parameter sistem di lapangan (Kurniawan & Jamaaluddin, 2023). Hal ini penting karena skenario penggunaan mungkin berubah (contoh: ukuran gelas yang berbeda, sehingga threshold sensor perlu diubah).

Selain program utama, konfigurasi perangkat keras juga dilakukan melalui kode. Pin yang terhubung ke relay diatur dalam program, lalu cukup menggunakan perintah digitalWrite (pinRelay, HIGH) untuk menyalakan pompa (melalui relay). Relay merupakan komponen saklar elektronik berbasis elektromagnet yang dikendalikan oleh sinyal listrik kecil (Prasetya et al., 2022). Pada sistem ini, relay 5V digunakan sebagai penghubung untuk mengontrol pompa 12V: ketika pin Arduino dalam kondisi HIGH, koil relay aktif dan kontak tertutup sehingga pompa mendapat arus (menyala), sedangkan jika LOW, kontak terbuka dan pompa mati.

Hal yang sama berlaku untuk katup solenoid, di mana prinsip kerjanya serupa katup akan membuka atau menutup penuh saat diberikan tegangan (umumnya 12V DC) (Prasetya et al., 2022). Solenoid hanya memiliki dua keadaan (terbuka penuh atau tertutup penuh), sehingga kontrolnya mirip seperti relay. Pada Arduino, pengaturan PWM juga dapat diterapkan untuk mengatur kecepatan pompa, namun dalam sistem ini tidak digunakan karena pengaturan laju tidak diperlukan, hanya kontrol on/off saja.

Komunikasi serial digunakan untuk modul tertentu: misal modul RFID RC522 membaca kartu lalu mengirim UID via SPI ke Arduino Mega, kemudian diolah. Integrasi dua mikrokontroler (Mega2560 dan ESP8266) dalam rancangan bisa dilakukan dengan komunikasi serial UART antara keduanya Mega fokus kontrol sensor/aktuator, ESP fokus kirim data online. Namun banyak desain memilih cukup satu ESP32 (yang memiliki banyak GPIO dan dual core) untuk menyederhanakan.

Terakhir, pengujian dilakukan untuk memastikan tiap bagian bekerja: uji sensor (apakah pembacaan jarak sesuai penggaris, dsb)(Prasetya et al., 2022), uji aktuator (pompa mengalir sesuai waktu), uji komunikasi (data benar terkirim ke *Firebase*), serta uji end-to-end. Jika ditemukan error, debugging dilakukan baik di hardware (penempatan sensor, koneksi listrik) maupun software (logika, penanganan kondisi edge cases). Misalnya, perlu dipastikan setelah gelas diangkat, sensor ultrasonik membaca jarak besar dan sistem kembali standby tanpa langsung aktif membanjiri, hal-hal semacam itu ditangani di program (dengan state machine).

Dengan implementasi metode di atas, sistem dispenser pintar diharapkan berjalan andal. Ringkasnya: sensor memberi data *real-time*, mikrokontroler memproses dengan program

terstruktur, mengambil aksi melalui aktuator, dan melaporkan status ke *cloud*; aplikasi membaca *cloud* dan memberi info/kontrol ke user. Seluruh siklus tersebut melibatkan teknologi pendukung IoT dan kendali tertanam yang saling terintegrasi.

2.6 Tinjauan Penelitian Terdahulu

Pengembangan sistem IoT untuk dispenser pintar telah menarik perhatian peneliti dalam beberapa tahun terakhir. Berikut beberapa studi dan proyek terdahulu, baik di Indonesia maupun internasional, yang relevan dengan topik ini:

1. Dispenser Otomatis Berbasis Mikrokontroler (2012–2016):

   Riset awal otomasi dispenser banyak berfokus pada mekanisme hands free. Daniel (2012) merancang otomatisasi keran dispenser memakai mikrokontroler AT89S52, sensor photodioda dan ultrasonik untuk mendeteksi gelas dan menghentikan aliran air secara otomatis (Ramdani, et al., 2024). Kemudian, Prasetyo (2016) mengembangkan dispenser air otomatis berbasis ATMega8 dengan sensor infra merah; sistemnya berhasil membuat air keluar otomatis saat gelas dideksi dan berhenti ketika penuh (Kurniawan & Jamaaluddin, 2023)Temuan ini membuktikan konsep dasar bahwa kombinasi sensor jarak dan kontrol mikrokontroler efektif mencegah tumpahnya air dan memudahkan pengguna.

2. Rancang Bangun Dispenser Pintar (2018–2020):

   Singgeta & Rumondor (2018) melaporkan prototype dispenser otomatis menggunakan sensor ultrasonik HC-SR04 dan Arduino Mega2560 (Ramdani, et al., 2024). Hasilnya, pengguna cukup menekan tombol volume yang diinginkan, lalu sistem mengisi gelas hingga volume tersebut tercapai berkat pembacaan sensor yang akurat. Penelitian lain oleh Katon & Yuniati (2020) (Universitas 45 Surabaya) juga menggarap dispenser pintar IoT dengan metode prototipe – mereka mengintegrasikan sensor level, kendali valve, dan modul Wi-Fi, meski detail hasil belum dipublikasikan luas. Secara umum, periode ini banyak skripsi dan prototipe bermunculan, menggabungkan otomasi dispenser dengan konsep IoT dasar.

3. Implementasi IoT untuk *Monitoring* (2021–2022):

   Devin A. Prasetya dkk. (2022) dari Telkom University membuat inovasi Smart Dispenser IoT untuk memantau jumlah air minum pengguna demi kesehatan. Sistem mereka mengirim data volume air yang diminum ke aplikasi *Android*, mengingatkan pengguna minum air secara teratur, dan bahkan terhubung ke penjual galon untuk pemesanan otomatis ketika stok habis(Prasetya et al., 2022). Pengujian menunjukkan seluruh sensor

dan perangkat berfungsi baik dengan akurasi layak dan koneksi ke *Firebase Realtime Database* berhasil. Rata-rata delay 169 ms tergolong baik, menandakan *monitoring real-time* dapat dicapai. Penelitian ini menegaskan bahwa IoT dapat diterapkan pada dispenser untuk tujuan kesehatan (hydration tracking) dengan kinerja jaringan memuaskan.

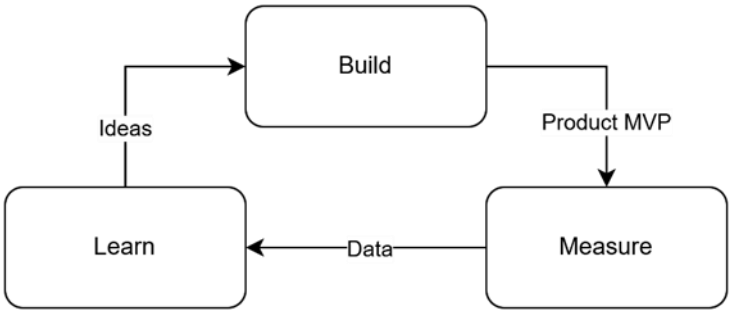
4. Smart Dispenser dengan Fitur Pembayaran (2023):
   Kurniawan & Jamaaluddin (2023) merancang Dispenser Pintar dengan Pembayaran Tanpa Uang Tunai. Mereka menerapkan modul RFID untuk otentikasi dan pembayaran saldo, flow sensor untuk menghitung volume air yang diambil, serta ESP32 untuk konektivitas ke server. Dispenser ini ditujukan sebagai kios isi ulang air minum mandiri. Hasilnya, sistem mampu mengotomatisasi penjualan air mineral isi ulang – pengguna cukup menempel kartu RFID bersaldo, memilih volume botol (600 mL s.d. 1 L), air keluar otomatis sesuai volume dan saldo terpotong. Pengujian menunjukkan pembacaan RFID dan kontrol valve via ESP32 berjalan lancar dan cukup cepat. Karya ini menunjukkan integrasi IoT tidak hanya untuk *monitoring* tapi juga otomasi transaksi pada dispenser pintar.
5. Smart Dispenser untuk Hydration *Monitoring* (2023, Internasional):
   Febriant Yapson et al. (2023) mempublikasikan Development of *Smart water dispenser System* for Daily Hydration *Monitoring* and Analysis di konferensi IEEE. Mereka mengembangkan sistem dispenser yang mampu melacak asupan cairan harian pengguna secara otomatis. Dengan kombinasi sensor flow dan aplikasi *mobile*, sistem mencatat berapa volume yang diminum setiap pengguna dan menganalisis pola hidrasi. Solusi ini mirip dengan Devin (2022) namun dipresentasikan ke komunitas internasional, menandakan topik ini mendapat perhatian global di ranah smart healthcare.
6. IoT-Based Dispenser Lainnya
   Selain air minum, konsep dispenser pintar IoT diterapkan di domain lain. Basith (2023) membuat smart hand sanitizer dispenser dengan sensor triboelectric dan dukungan IoT-*cloud* untuk keperluan pandemi. Chandana (2023) mengembangkan smart fuel dispenser (pompa bensin pintar) menggunakan RFID dan pemantauan IoT untuk otomotif – walau berbeda objek (bahan bakar), prinsipnya serupa: sensor volume, aktuator pompa, IoT *monitoring*. Hal ini menunjukkan fleksibilitas teknologi dispenser pintar dalam berbagai sektor.

Dari tinjauan di atas, dapat disimpulkan bahwa penelitian terdahulu telah berhasil membuktikan: (1) Otomatisasi dispenser dengan sensor (ultrasonik/infrared) mampu bekerja

efektif mencegah overflow, (2) IoT menambahkan nilai berupa *monitoring* jarak jauh dan analisis data penggunaan air secara realtime, (3) Inovasi lanjutan seperti integrasi pembayaran digital dan pengingat kesehatan meningkatkan kegunaan dispenser pintar. Karya-karya tersebut menjadi landasan penting bagi proyek skripsi ini. Sistem yang akan dikembangkan mengambil pelajaran dari tiap studi – misalnya menggunakan sensor ultrasonik dan flow meter untuk akurasi lebih baik (mengacu Singgeta 2018), menerapkan *Firebase* untuk realtime data (mengacu Devin 2022), serta mempertimbangkan kemudahan user melalui fitur *notifikasi* (seperti yang disarankan Yapson 2023). Dengan demikian, diharapkan *Smart water dispenser* yang dirancang pada penelitian ini dapat memberikan kontribusi berupa implementasi komprehensif: otomatis, terhubung IoT, *real-time monitoring*, dan *notifikasi* cerdas, selaras dengan tren dan rekomendasi penelitian terdahulu.

BAB III
METODE PENELITIAN

3.1 Jenis dan Pendekatan Penelitian



Gambar 3.1 Lean Metodologi

Penelitian ini mengadopsi metodologi Lean, sebuah pendekatan yang berfokus pada pengembangan produk secara cepat dan efisien melalui siklus iteratif *Build-Measure-Learn*. Metodologi ini dipilih untuk meminimalkan risiko pengembangan produk yang tidak sesuai dengan kebutuhan pengguna dengan mengedepankan validasi berkelanjutan melalui pengujian fitur yang dikembangkan. Ini sangat relevan untuk sistem dispenser pintar berbasis Internet of Things(IoT), di mana IoT memungkinkan pemantauan, dan *notifikasi* secara *real-time*, memberikan kemudahan serta fleksibilitas dalam kontrol dan pemeliharaan perangkat dari jarak jauh.

Secara keseluruhan, strategi pengembangan sistem dalam penelitian ini dilaksanakan dalam dua iterasi utama:

1. Iterasi Pertama: Bertujuan untuk membangun *Minimum viable product (MVP)*, yaitu versi awal sistem yang memiliki fitur dasar dan siap diuji oleh pengguna.
2. Iterasi Kedua: Difokuskan pada penyempurnaan fitur, berdasarkan data dan masukan yang diperoleh dari tahap evaluasi iterasi pertama.

Pendekatan ini dinilai tepat karena memberikan fleksibilitas dalam pengambilan keputusan berbasis data, memungkinkan pengembangan sistem yang adaptif terhadap perubahan kebutuhan, dan mendorong penghematan sumber daya dalam proses rekayasa teknologi.

3.2 Metode Pengumpulan Data

Pengumpulan data dalam penelitian ini dilakukan dengan pendekatan kombinasi antara metode kuantitatif dan kualitatif, untuk memperoleh pemahaman yang holistik mengenai kebutuhan pengguna serta mengevaluasi kinerja sistem yang dikembangkan. Proses pengembangan sistem dilakukan secara iteratif dengan melibatkan pengguna akhir pada setiap siklusnya. Uji coba prototipe dilakukan oleh pengguna tetap di lingkungan kampus, terutama petugas OB serta mahasiswa dan staf, sehingga setiap feedback dan hasil observasi dari penggunaan nyata digunakan sebagai dasar pengambilan keputusan dalam pengembangan fitur dan perbaikan sistem. Teknik pengumpulan data disesuaikan dengan tahapan dalam siklus Lean Methodology (*BuildMeasure-Learn*), dan dilakukan secara bertahap pada setiap iterasi pengembangan.

Adapun metode-metode pengumpulan data yang digunakan adalah sebagai berikut:

1. Kuesioner

   Kuesioner disusun dan disebarkan kepada calon pengguna sistem untuk mengidentifikasi kebutuhan, preferensi, serta permasalahan yang umum dialami saat menggunakan dispenser air konvensional. Metode ini digunakan pada tahap *Build* – Iterasi Pertama, guna memperoleh data primer mengenai fitur-fitur yang diharapkan pengguna, seperti *notifikasi* otomatis, kemudahan pengoperasian, dan kontrol suhu air. Kuesioner terdiri dari pertanyaan tertutup dan terbuka, dengan skala Likert untuk menilai tingkat kepentingan setiap fitur. Data dari kuesioner ini menjadi dasar dalam proses perancangan *Minimum viable product (MVP)*.

2. Observasi Langsung

   Metode observasi digunakan saat pengujian sistem oleh pengguna, untuk melihat secara langsung bagaimana interaksi pengguna dengan perangkat. Observasi mencakup aspek kemudahan penggunaan, respon sensor, efektivitas *notifikasi*, serta kemungkinan terjadinya kendala teknis selama operasional. Pengamatan dilakukan oleh peneliti dengan mencatat perilaku pengguna, waktu respons sistem, serta kejadian-kejadian tidak terduga yang muncul selama pengujian MVP maupun pasca kalibrasi di iterasi kedua.

3. Wawancara Semi-Terstruktur

   Wawancara dilakukan terhadap pengguna yang telah mencoba sistem, baik setelah iterasi pertama maupun kedua. Tujuan wawancara adalah untuk menggali lebih dalam pengalaman pengguna, persepsi terhadap fitur, serta masukan untuk penyempurnaan

sistem. Pendekatan semi-terstruktur memungkinkan peneliti untuk mengeksplorasi tanggapan lebih luas sambil tetap menjaga fokus terhadap poin-poin evaluasi utama.

4. Uji Coba Sistem

   Uji coba dilakukan dengan melibatkan pengguna akhir untuk menjalankan sistem secara langsung dalam skenario penggunaan riil. Data yang dikumpulkan berupa:

   a. Keberhasilan aktivasi fitur otomatisasi.
   b. Akurasi pembacaan sensor (*load cell* dan ultrasonik).
   c. Ketepatan pengiriman *notifikasi* melalui aplikasi.
   d. Waktu respons antar komponen (ESP32 – *Firebase* – aplikasi).

Seluruh data dari hasil uji coba ini direkam dan dianalisis secara kuantitatif dan kualitatif untuk dijadikan dasar validasi sistem dan iterasi pengembangan selanjutnya. Dengan penerapan metode-metode tersebut secara terstruktur, penelitian ini memastikan bahwa setiap keputusan desain dan pengembangan sistem didasarkan pada data empiris yang valid dan relevan dengan kebutuhan pengguna.

3.3 Teknik Pengembangan Sistem

Teknik pengembangan sistem dalam penelitian ini mengacu pada pendekatan Lean Methodology dengan menerapkan siklus iteratif *Build–Measure–Learn*. Pendekatan ini dirancang untuk menciptakan solusi teknologi secara cepat, adaptif, dan berbasis validasi pengguna. Pengembangan dilakukan dalam dua iterasi utama, di mana setiap iterasi mencerminkan siklus penuh dari perancangan awal hingga evaluasi sistem.

Teknik pengembangan tidak hanya berfokus pada pencapaian fungsionalitas teknis, tetapi juga mengutamakan pengujian berbasis pengguna (user *testing*) dan peningkatan berkelanjutan (continuous improvement). Pengembangan dilakukan dengan menggabungkan proses rekayasa perangkat keras dan perangkat lunak secara terintegrasi, serta melibatkan *feedback real-time* dari pengguna melalui media aplikasi *mobile*.

1. Perancangan *Minimum viable product (MVP)*

   Pengembangan diawali dengan merancang prototipe awal yang memuat fitur esensial dari sistem *Smart water dispenser*, seperti:

   a. Pengeluaran air otomatis berbasis deteksi gelas
   b. *Monitoring* volume galon
   c. *Monitoring* suhu air panas dan dingin
   d. Pengiriman *notifikasi* melalui aplikasi *mobile*

Desain MVP menitikberatkan pada prinsip fungsi minimum tetapi bernilai, sehingga sistem dapat segera diuji kepada pengguna tanpa menunggu sempurna secara teknis.

2. Integrasi Komponen Perangkat Keras dan Lunak

   Pengembangan dilakukan dengan mengintegrasikan mikrokontroler ESP32 sebagai pusat kendali, sensor DS18B20, *load cell*, dan ultrasonik, serta aktuator seperti relay dan servo motor. Sistem dikendalikan melalui program yang ditulis menggunakan Arduino IDE, dan terhubung dengan *Firebase Realtime Database* untuk kebutuhan penyimpanan data dan *notifikasi real-time*. Aplikasi *mobile* dikembangkan menggunakan *React Native*, dengan antarmuka yang sederhana dan fungsional agar pengguna dapat memantau status dispenser, suhu, dan volume air dengan mudah.

3. Pengujian dan Kalibrasi Iteratif

   Setiap fitur diuji secara bertahap dan disesuaikan dengan umpan balik yang diterima. Sistem diuji dalam kondisi nyata (real environment) untuk menilai keandalan performa, baik dari segi hardware maupun software. Proses kalibrasi sensor, filter data, dan penyesuaian logika kontrol menjadi bagian penting untuk memastikan akurasi sistem.

4. Sinkronisasi *Cloud* dan *Mobile* Interface

   Integrasi antara ESP32, *Firebase*, dan aplikasi *mobile* dilakukan untuk memungkinkan komunikasi dua arah:

   a. Pengiriman data sensor ke *cloud* dan ditampilkan ke pengguna
   b. Penerimaan perintah dari aplikasi ke mikrokontroler (misalnya pengaturan mode dispenser)

   Komunikasi ini dijaga melalui struktur data JSON di *Firebase* dan event listener yang bekerja secara *real-time*, memastikan bahwa seluruh elemen sistem saling terhubung secara responsif.

5. Evaluasi dan Validasi Iteratif

   Setelah setiap iterasi, dilakukan evaluasi fungsionalitas dan persepsi pengguna terhadap fitur. Masukan dari pengguna dijadikan dasar untuk memperbaiki sistem pada iterasi berikutnya. Validasi difokuskan pada tiga aspek utama:

   a. Akurasi sensor dan keandalan sistem
   b. Kenyamanan dan kemudahan penggunaan
   c. Kecepatan dan ketepatan *notifikasi*

   Melalui teknik pengembangan berbasis Lean, sistem dapat dikembangkan secara bertahap namun terukur, memungkinkan pencapaian solusi teknologi yang tidak hanya canggih

secara teknis, tetapi juga relevan dan bermanfaat secara nyata bagi pengguna.

3.4 Iterasi 1

Pada iterasi pertama, tujuan utama adalah mengembangkan *Minimum viable product (MVP)* yang berfungsi sebagai prototipe dasar dengan fitur-fitur penting seperti pengeluaran air otomatis, *notifikasi* penggantian galon, dan integrasi dengan aplikasi. Dengan MVP ini, penulis berupaya menjawab kebutuhan pengguna melalui solusi sederhana namun efektif, sekaligus mendapatkan data awal yang penting untuk iterasi dan pengembangan berikutnya.

3.4.1 *Build*

Analisis Kebutuhan Pengguna

Langkah awal adalah memahami secara detail kebutuhan dan harapan pengguna. Dengan memahami apa yang benar-benar diinginkan oleh pengguna, pengembangan fitur dapat lebih tepat sasaran, sehingga fitur yang dihasilkan benar-benar sesuai dengan ekspektasi mereka. Untuk memperoleh informasi ini, penulis menyebarkan kuesioner melalui Google Forms, yang mencakup beberapa aspek utama dalam analisis kebutuhan pengguna:

1. Permasalahan yang di hadapi pengguna dengan dispenser air saat ini

Tabel 3.1 Daftar Pertanyaan Kuesioner dan Tujuannya dalam Mengidentifikasi Permasalahan Pengguna terhadap Dispenser Air Konvensional

Pertanyaan Positif	Pertanyaan Negatif	Tujuan
Apakah dispenser air anda saat ini memungkinkan anda untuk memantau air dengan suhu yang sudah siap digunakan? Minsalnya air dingin atau panas?	Apakah anda mengalami masalah dalam mendapatkan air bersuhu yang belum siap digunakan dari dispenser air yang anda gunakan sekarang?	Mengevaluasi tingkat kepuasan dan pengalaman pengguna dalam menggunakan dispenser air saat ini khususnya terkait kesiapan suhu air.
Apakah anda pernah merasa kesulitan saat harus mengetahui kapan air di galon habis tanpa *notifikasi* otomatis?	Apakah anda merasa repot setiap kali harus mengganti galon air secara manual karena dispenser tidak memberikan peringatan saat air hamper habis?	Mengevaluasi persepsi pengguna terhadap kemudahan dalam memantau ketersediaan air galon serta efektivitas sistem peringatan yang ada guna mengidentifikasi kebutuhan akan fitur *notifikasi* otomatis dalam penggunaan dispenser air.
Apakah dispenser air anda saat ini mudah digunakan tanpa kesulitan dalam menekan tombol atau mengoperasikannya?	Apakah anda sering mengalami kesulitan dalam menggunakan dispenser, seperti air yang tidak keluar dengan lancar atau tombol yang sulit ditekan?	Mengidentifikasi sejauh mana kemudahan operasional dispenser air konvensional dirasakan oleh pengguna, khususnya terkait aspek ekonomis dan penggunaan fisik dalam aktivitas sehari-hari.

Mengidentifikasi masalah atau tantangan utama yang dihadapi pengguna dalam menggunakan dispenser air konvensional. Data ini akan membantu dalam memahami apa saja kendala yang diinginkan pengguna untuk diatasi oleh dispenser pintar.

2. Preferensi Fitur

Tabel 3.2 Daftar Pertanyaan Kuesioner Terkait Preferensi Fitur pada Sistem Dispenser Air Pintar

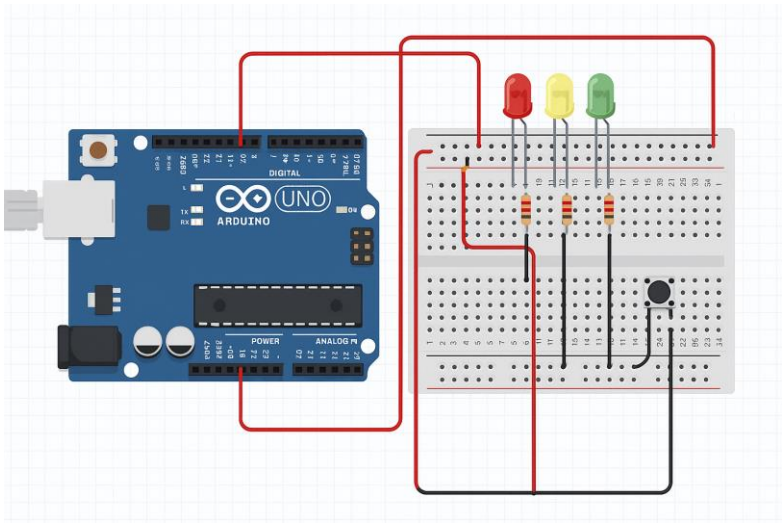
Pertanyaan	Tujuan
Apakah Anda tertarik jika dispenser air dapat mengeluarkan air secara otomatis saat gelas diletakkan dibawah keran, tanpa harus menekan tombol?	Kemudahan dan otomatisasi penggunaan dispenser air
Apakah Anda tertarik jika dispenser air bisa memberikan peringatan ketika air sudah mencapai suhu yang Anda inginkan?	Kontrol suhu air melalui perangkat seluler
Apakah Anda tertarik jika Anda bisa menerima *notifikasi* di ponsel ketika galon air hampir habis, sehingga Anda bisa segera menggantinya tanpa harus mengecek manual?	*Notifikasi* dan pengingat mengenai status air dan galon
Apakah Anda tertarik jika Anda bisa memantau dan mengontrol dispenser air dari jarak jauh melalui aplikasi?	Pemantauan dan kontrol jarak jauh dispenser air

Bagian ini difokuskan pada pengidentifikasian fitur yang dianggap penting dan diinginkan oleh pengguna, *notifikasi* suhu air (panas, dingin, atau normal) yang sudah siap digunakan, *notifikasi* volume air gallon habis melalui aplikasi *mobile*, dan otomasi air yang keluar secara otomatis. Memahami preferensi ini memungkinkan untuk menetapkan prioritas pada fitur-fitur yang paling diinginkan. Hasil dari analisis kebutuhan ini menjadi pedoman penting dalam proses perancangan dan pengembangan fitur pada sistem dispenser air pintar berbasis IoT. Dengan demikian, dispenser yang dikembangkan tidak hanya canggih dari segi teknologi, tetapi juga sesuai dengan kebutuhan dan preferensi pengguna, memberikan solusi yang optimal untuk memenuhi ekspektasi mereka.

Perencangan Fitur-Fitur MVP

Sistem dispenser pintar ini dirancang sebagai perangkat IoT yang mengintegrasikan perangkat keras dan perangkat lunak, memungkinkan pengguna untuk memantau dispenser dengan mudah kapan saja dan di mana saja. Pengguna dapat mengakses fiturfitur utama dispenser melalui aplikasi *mobile* yang intuitif, yang disambungkan ke *cloud Firebase* untuk penyimpanan dan akses data *real-time*. Berikut adalah penjelasan dari sistem utama dispenser pintar:

1. Gambar Sistem Aplikasi



Gambar 3.2 Diagram Arsitektur Sistem *Smart water dispenser* Berbasis IOT

Sistem ini dilengkapi dengan aplikasi *mobile* yang mendukung platform *Android*, berfungsi sebagai antarmuka utama untuk pengguna. Melalui aplikasi ini, pengguna dapat memantau status dispenser, dan menerima *notifikasi* penting secara langsung. Semua data yang dihasilkan oleh dispenser, seperti kualitas air dan status galon, disimpan di server *cloud* misalnya *Firebase* yang memastikan data selalu diperbarui dan dapat diakses secara *real-time* oleh pengguna. Arsitektur *cloud* ini juga meningkatkan fleksibilitas dan kehandalan sistem, memungkinkan *notifikasi* yang cepat dan akses dari berbagai perangkat.

2. Gambaran Dispenser Pintar

Untuk memberikan pengalaman yang komprehensif dan bermanfaat bagi pengguna, sistem dispenser pintar ini dilengkapi dengan beberapa fungsi utama:

a. *Notifikasi* Volume Galon

Fitur ini memungkinkan sistem mendeteksi kapan volume air dalam galon telah mencapai ambang batas minimum. Dengan bantuan sensor berat (*load cell*) dan sensor ketinggian air, sistem akan memantau kondisi galon secara *real-time*. Ketika air hampir habis, sistem akan secara otomatis mengirimkan *notifikasi* ke aplikasi *mobile*. Dengan adanya fitur ini, pengguna atau petugas tidak perlu lagi memeriksa galon secara manual, sehingga risiko kehabisan air mendadak dapat diminimalkan dan proses pengisian ulang menjadi lebih tepat waktu.

b. *Notifikasi* Suhu Air

Fitur *notifikasi* suhu air galon memungkinkan pengguna menerima pemberitahuan secara otomatis saat suhu air panas atau dingin telah mencapai tingkat yang optimal untuk digunakan. Melalui aplikasi, sistem akan memantau proses pemanasan dan pendinginan air secara *real-time*, dan akan mengirimkan *notifikasi* ke perangkat pengguna begitu air siap disajikan. Dengan demikian, pengguna tidak perlu lagi memeriksa secara manual kondisi suhu air pada dispenser, sehingga penggunaan menjadi lebih praktis dan efisien.

3. Pemilihan 3 Mode

Fitur pemilihan mode ini memungkinkan pengguna untuk memilih jenis air yang diinginkan sesuai kebutuhan, melalui satu buah tombol tekan (push button). Sistem akan mengenali jumlah tekanan sebagai perintah tertentu, yaitu:

Tabel 3.3 Pemilihan 3 Mode Air

Jumlah Klik (Push Button)	**Mode yang Diaktifkan**
1 kali klik	Mode Air Dingin
2 kali klik	Mode Air Panas
3 kali klik	Mode Kombinasi (Dingin + Panas)

Setiap mode akan diindikasikan dengan LED warna tertentu: biru untuk dingin, merah untuk panas, dan putih untuk kombinasi. Dengan desain kontrol yang sederhana ini, pengguna dapat memilih mode dengan mudah tanpa perlu navigasi menu atau pengaturan tambahan.

4. Pengeluaran Air Otomatis Berbasis deteksi Gelas

Sistem dilengkapi dengan sensor ultrasonik yang secara otomatis mendeteksi keberadaan gelas pada area pengisian. Ketika gelas terdeteksi berada pada jarak yang sesuai, sistem akan mengaktifkan pompa untuk mengalirkan air secara otomatis. Sebaliknya, jika gelas tidak terdeteksi, pompa akan tetap dalam keadaan nonaktif. Dengan fitur ini, proses pengambilan air menjadi lebih higienis dan efisien, karena pengguna tidak perlu menyentuh tombol atau tuas apapun secara langsung.

Kebutuhan Komponen

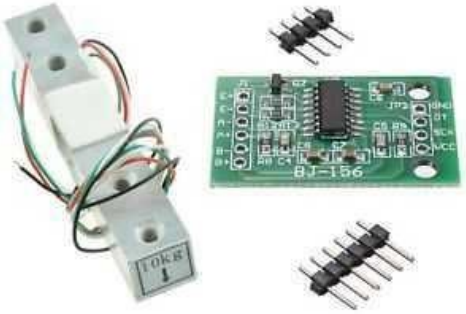
Untuk memastikan dispenser pintar ini dapat berfungsi secara optimal dan memberikan pengalaman pengguna yang memuaskan, pemilihan setiap komponen perangkat keras dan perangkat lunak dilakukan dengan teliti. Komponen-komponen ini berperan penting dalam menjaga performa, konektivitas, dan kemudahan penggunaan dispenser berbasis IoT.

1. Perangkat Keras
   a. ESP32

Gambar 3.3 ESP32

ESP32 merupakan mikrokontroler utama yang memiliki fitur Wi-Fi dan Bluetooth terintegrasi. Modul ini berperan sebagai pusat kendali seluruh sistem IoT, mengatur proses pembacaan sensor, komunikasi data ke *Firebase*, serta pengendalian aktuator seperti relay, pompa, dan servo.

   b. Loadcell & HX711 Module 20 kg



Gambar 3.4 Loadcell & HX711 Module 20 kg

Load cell digunakan untuk mengukur berat galon air dengan akurasi tinggi, sedangkan HX711 merupakan modul ADC (Analog to Digital Converter) yang memperkuat dan mengubah sinyal dari *load cell* menjadi data digital yang dapat dibaca ESP32.

   c. Sensor Suhu DS18B20

Gambar 3.5 Sensor Suhu DS18B20

Sensor ini digunakan untuk mengukur suhu air panas dan dingin secara realtime. DS18B20 memiliki keunggulan dalam hal ketelitian dan komunikasi digital satu kabel (1-Wire) sehingga efisien dalam penggunaan pin mikrokontroler.

d. Waterpump

Gambar 3.6 Waterpump

Pompa berfungsi untuk mengalirkan air dari galon ke keran ketika sensor mendeteksi adanya gelas. Pompa ini dikendalikan oleh relay dan aktif hanya saat dibutuhkan, menjamin efisiensi dan hemat energi.

e. LCD 12C 16x2

Gambar 3.7 LCD 12C 16x2

LCD ini digunakan untuk menampilkan informasi penting seperti status volume air, suhu, dan *notifikasi* sistem. Antarmuka I2C memungkinkan penggunaan hanya dua pin data, sehingga lebih hemat pin pada ESP32.

f. Relay Module Dual Channel 5V

Gambar 3.8 Relay Module Dual Channel 5V

Relay digunakan sebagai saklar elektronik untuk mengontrol arus tinggi dari perangkat seperti pompa dan pemanas air (water heater), dengan perintah yang berasal dari ESP32.

g. Water Level Sensor

Gambar 3.9 Water Level Sensor

Sensor ini mendeteksi ketinggian air dalam galon untuk mengetahui apakah volume air mencukupi atau sudah hampir habis, sehingga dapat memicu *notifikasi* ke pengguna atau petugas.

h. Water Heat

Gambar 3.10 Water Heat

Komponen ini bertugas untuk memanaskan air hingga suhu yang diinginkan. Dikendalikan melalui relay, pemanas hanya aktif sesuai kebutuhan pengguna berdasarkan suhu yang ditentukan.

i. Power Supply 12V 10A

Gambar 3.11 Power Supply 12V 10A

Catu daya utama sistem ini menyediakan tegangan dan arus yang stabil untuk berbagai komponen, seperti pompa, relay, ESP32, dan LCD. Spesifikasi 12V 10A memastikan kestabilan daya untuk beban yang relatif tinggi.

j. Kompressor Sanken

Gambar 3.12 Kompressor Sanken

Digunakan sebagai sistem pendingin untuk menghasilkan air dingin. Kompresor ini dikontrol oleh relay dan bekerja dalam siklus otomatis sesuai dengan kebutuhan suhu.

k. Ultrasonic HC-SR04

Gambar 3.13 Ultrasonic HC-SR04

Sensor ini digunakan untuk mendeteksi keberadaan gelas di bawah keran dispenser. Ketika gelas terdeteksi, sistem akan memulai pengeluaran air secara otomatis tanpa sentuhan.

l. LED Light

Gambar 3.14 LED Light

LED berfungsi sebagai indikator visual status sistem seperti: pompa aktif, galon habis, suhu siap, atau mode operasi yang sedang berjalan.

m. Resistor 220ohm ¼ w

Gambar 3.15 Resistor

Digunakan sebagai pembatas arus pada rangkaian, terutama untuk LED dan input sinyal dari sensor atau tombol, guna mencegah kerusakan akibat arus berlebih.

n. Kapasitor Elektrolit 470µF–1000µ

Gambar 3.16 Kapasitor Elektrolit 470µF–1000µ

Kapasitor ini digunakan untuk menstabilkan tegangan, menyaring lonjakan arus, serta mengurangi noise pada sistem, terutama pada modul power supply dan pompa.

o. Push Button

Gambar 3.17 Push Button

Tombol tekan digunakan sebagai input manual untuk memilih mode operasi (air panas, dingin, atau keduanya). Sistem membaca jumlah klik untuk menentukan mode yang dipilih.

p. Servo Motor MG996R 10 Kg

Gambar 3.18 Servo Notor

Servo motor digunakan untuk menggerakkan komponen mekanis tertentu, seperti membuka katup atau penutup keran otomatis. MG996R dipilih karena torsi tinggi dan akurasi gerakan yang baik.

q. Kabel Jumper

Gambar 3.19 Kabel Jumper

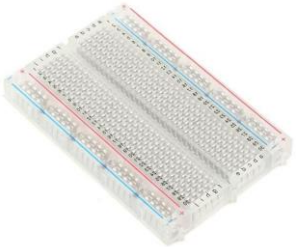
Kabel jumper digunakan untuk menghubungkan komponen-komponen elektronik di atas breadboard atau langsung ke ESP32 dengan berbagai jenis konektor sesuai kebutuhan.

r. Kabel AWG 24

Gambar 3.20 Kabel AWG

Kabel ini digunakan untuk jalur arus dan sinyal antar komponen. Ukuran AWG 24 cukup fleksibel untuk kebutuhan sinyal digital dan daya rendah.

s. Breadboard



Gambar 3.21 Breadboard

Breadboard digunakan sebagai papan sirkuit prototipe tanpa soldering, memudahkan proses pengembangan dan pengujian sistem secara cepat dan efisien.

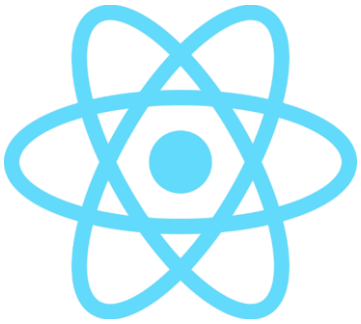
2. Perangkat Lunak

a. Arduino IDE

Gambar 3.22 Arduino IDE

Arduino IDE merupakan platform pemrograman utama yang digunakan untuk mengembangkan kode pada mikrokontroler ESP32. Platform ini menyediakan antarmuka yang intuitif dan mudah digunakan untuk pemrograman perangkat IoT, memastikan bahwa setiap sensor dan modul dapat dijalankan secara efisien dan sesuai dengan instruksi.

b. *React Native* dan *Expo* Go



Gambar 3.23 *React Native* dan *Expo* Go

React Native adalah framework pengembangan aplikasi *mobile* yang memungkinkan pembuatan aplikasi lintas platform, sehingga kompatibel dengan perangkat berbasis *Android*. Framework ini dipilih karena kemampuannya dalam membangun antarmuka pengguna yang ramah pengguna (user-friendly) dan cepat, tanpa mengorbankan performa atau kualitas aplikasi. Untuk mendukung efisiensi pengembangan, digunakan *Expo* Go, yaitu aplikasi pendukung dari ekosistem *Expo* yang memungkinkan pengembang menjalankan dan menguji aplikasi *React Native* secara langsung di perangkat *Android* tanpa perlu melakukan instalasi APK. Cukup dengan memindai kode QR dari development server, aplikasi dapat dimuat secara instan. *Expo* Go mendukung live reload dan hot reloading, sehingga sangat membantu dalam proses iterasi, debugging, dan demonstrasi fitur aplikasi secara cepat dan praktis.

c. *Firebase*/*Cloud* Server

Gambar 3.24 Firebse

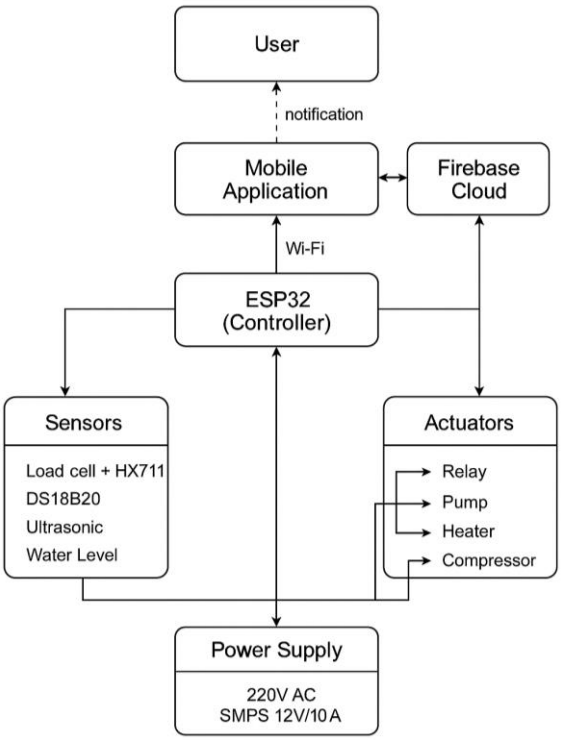
Firebase digunakan sebagai server berbasis *cloud* untuk penyimpanan dan sinkronisasi data dari dispenser ke aplikasi *mobile*. *Firebase* memungkinkan data yang dikumpulkan oleh ESP32, seperti status galon, dapat disimpan secara aman di *cloud* dan diakses oleh pengguna kapan saja.

Perancangan Sistem

Tahap ini berfokus pada perancangan alur kerja dan struktur sistem yang mendukung setiap komponen agar berfungsi sesuai perannya dan terintegrasi dengan baik. Perancangan sistem meliputi berbagai diagram yang memberikan panduan visual mengenai interaksi dan komunikasi antar komponen. Rincian berikut membantu dalam memvisualisasikan bagaimana data dan perintah mengalir melalui sistem, memastikan kesesuaian dan efisiensi dalam operasional dispenser pintar berbasis IoT.

1. Diagram Blok Sistem

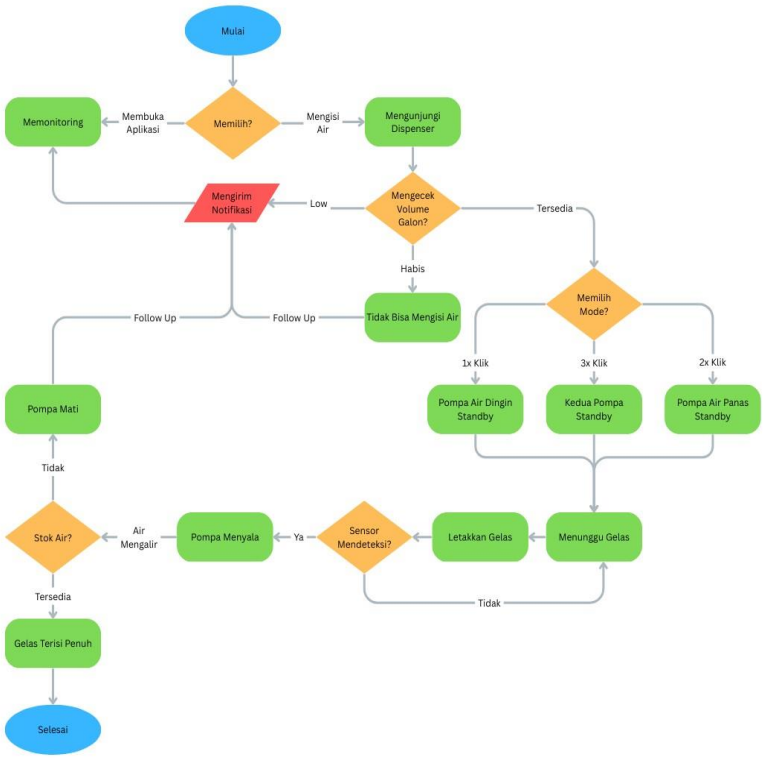
   Untuk memberikan gambaran menyeluruh mengenai hubungan antar komponen, Gambar 3.xx menampilkan diagram blok sistem.



Gambar 3.25 Blok Diagram Arsitektur Sistem Smart Water Dispenser berbasis IoT

Sensor (loadcell, DS18B20, water level, ultrasonik) mengirimkan data ke ESP32. Data diproses lalu dikirim ke Firebase melalui koneksi Wi-Fi. Aplikasi mobile membaca data tersebut dan menampilkan status dispenser kepada pengguna. ESP32 juga mengontrol aktuator (pompa, heater, kompresor) melalui modul relay. Sistem mendapat suplai daya dari SMPS 12V/10A yang diturunkan menjadi 5V/3.3V untuk mikrokontroler dan sensor.

2. Flowchart Sistem MVP

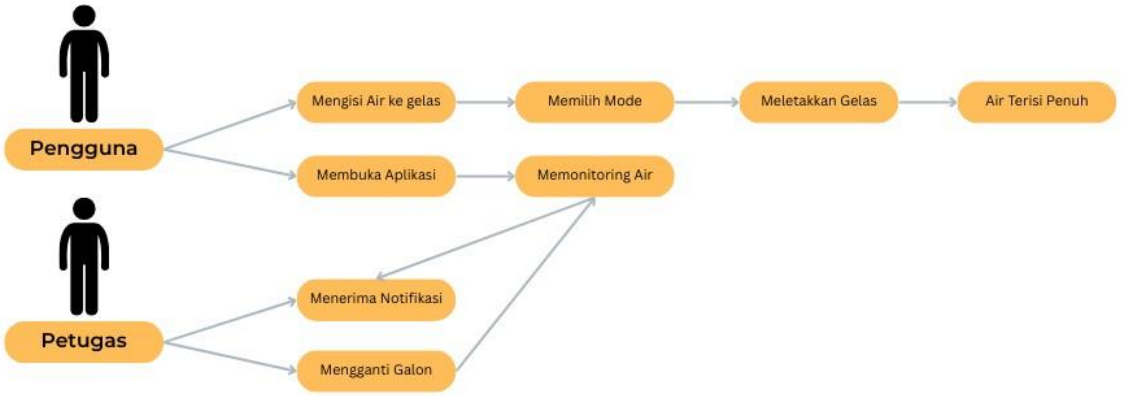


Gambar 3.26 Flowchart Sistem

Flowchart sistem *Smart water dispenser* menampilkan alur kerja dari pemantauan status galon hingga proses otomatisasi pengeluaran air berdasarkan deteksi gelas. Alur dimulai dengan pemeriksaan volume galon, di mana sistem menentukan apakah air masih tersedia atau galon telah kosong. Jika volume mencukupi, pengguna dapat memilih mode pengeluaran air (dingin, panas, atau kombinasi keduanya) melalui mekanisme tombol. Selanjutnya, sensor ultrasonik mendeteksi keberadaan gelas di bawah keran dispenser. Jika gelas terdeteksi, sistem secara otomatis mengaktifkan pompa air untuk mengalirkan air sesuai mode yang dipilih. Setelah proses pengisian selesai atau jika stok air tidak tersedia, sistem akan menghentikan pompa secara otomatis. Diagram ini menggambarkan langkahlangkah utama mulai dari input deteksi sensor hingga keluaran berupa pengisian

air, memastikan proses yang praktis, efisien, dan ramah pengguna, sekaligus meningkatkan kenyamanan dalam penggunaan dispenser air berbasis IoT.

3. Use Case Diagram

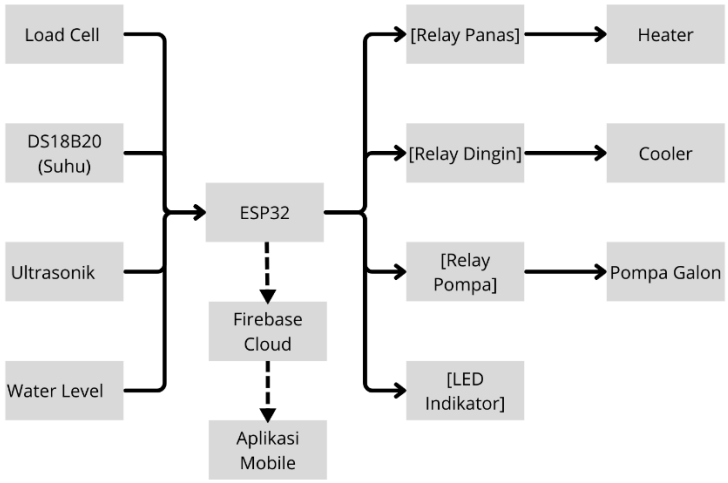


Gambar 3.27 Use Case Diagram

Diagram alur peran pengguna dan petugas pada sistem *Smart water dispenser* menggambarkan pembagian tugas dan interaksi antara kedua pihak dalam proses pengelolaan dan penggunaan dispenser berbasis IoT. Pengguna berperan dalam dua kegiatan utama, yaitu membuka aplikasi untuk melakukan *monitoring* air serta mengisi air ke dalam gelas. Setelah aplikasi dibuka, pengguna dapat memilih mode pengeluaran air (panas, dingin, atau kombinasi), kemudian meletakkan gelas di bawah keran. Ketika gelas terdeteksi, air akan dialirkan secara otomatis hingga gelas terisi penuh. Sementara itu, petugas memiliki tanggung jawab dalam hal pemeliharaan dan pengisian ulang galon. Petugas menerima *notifikasi* secara otomatis melalui aplikasi ketika volume air pada galon terdeteksi rendah. Berdasarkan *notifikasi* tersebut, petugas kemudian melakukan tindakan berupa penggantian galon untuk memastikan ketersediaan air tetap terjaga. Proses *monitoring* yang dilakukan oleh petugas dan pengguna terintegrasi dalam sistem yang sama, memungkinkan respons cepat terhadap kondisi dispenser. Diagram ini menekankan pentingnya kolaborasi antara pengguna dan petugas dalam menjaga efisiensi dan kelangsungan layanan air minum secara otomatis dan *real-time*.

4. Diagram Blok

a. Rangkaian Utama



Gambar 3.28 Rangkaian Utama Blok Diagram Sistem Smart Water Dispenser Berbasis IoT.

Rangkaian utama sistem dispenser pintar berbasis IoT, di mana semua komponen dikendalikan oleh mikrokontroler ESP32 yang terhubung ke berbagai sensor, aktuator, dan sumber daya eksternal.

Sistem ini bekerja dengan mengatur aliran air dari pompa ke dalam dua wadah utama: kompresor pendingin dan elemen pemanas air (heater). Ketika pompa air diaktifkan, air akan dialirkan ke kedua jalur tersebut secara bersamaan melalui perintah dari relay module dual channel. Pompa akan tetap aktif hingga sensor water level mendeteksi bahwa air telah mencapai batas atas di dalam masing-masing wadah. Begitu air menyentuh probe sensor, sistem secara otomatis memutus aliran listrik ke pompa melalui relay, menghentikan proses pengisian air.

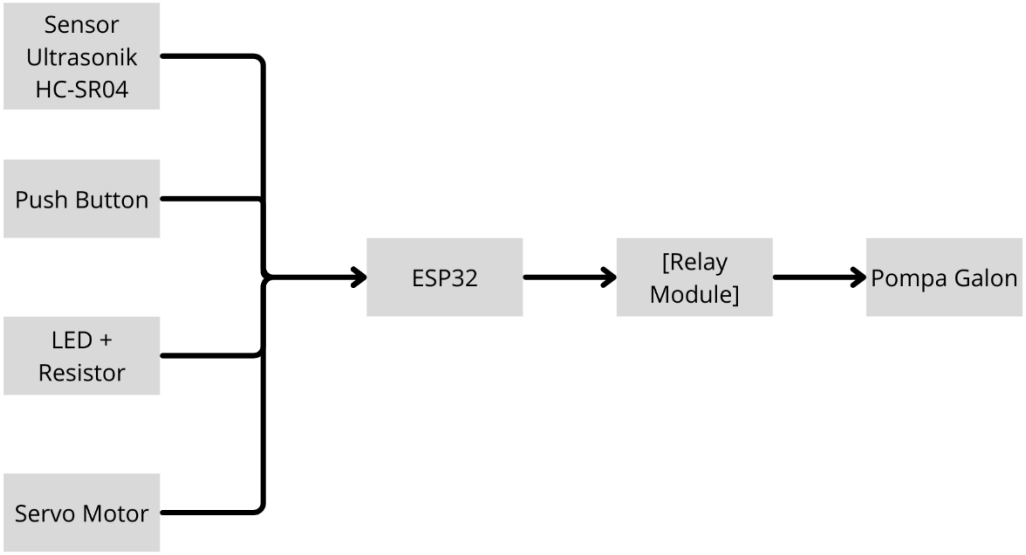
Setelah air tersedia dalam dua jalur, sensor suhu DS18B20 yang tertanam pada masing-masing jalur (panas dan dingin) akan mulai memantau perubahan suhu secara *real-time*. Informasi suhu tersebut ditampilkan secara langsung pada LCD I2C 16x2, yang memudahkan pengguna memantau status suhu tanpa harus membuka aplikasi.

Jika suhu pada air dingin maupun panas telah mencapai nilai maksimum yang telah ditentukan (misalnya < 10°C untuk dingin, dan > 80°C untuk panas), maka sensor suhu akan mengirimkan sinyal ke ESP32, dan ESP32 akan memberikan perintah kepada relay module untuk mematikan kompresor dan elemen pemanas.

Proses ini menjamin bahwa suhu tetap terjaga dalam rentang optimal tanpa konsumsi daya berlebih atau risiko pemanasan/pendingin berlebihan.

Sistem ini ditenagai oleh power supply 12V 10A, yang berfungsi sebagai sumber energi utama untuk mengaktifkan pompa, heater, dan kompresor. Power supply ini tidak digunakan untuk mikrokontroler ESP32, melainkan hanya diperuntukkan bagi perangkat yang membutuhkan arus dan tegangan tinggi.

b. Rangkaian Pengeluaran Air



Gambar 3.29 Rangkaian Pengeluaran Air

Sistem ini dirancang untuk mengatur proses pengeluaran air pada dispenser pintar secara otomatis melalui kombinasi interaksi pengguna (melalui push button) dan pemantauan objek menggunakan sensor ultrasonik. Komponen utama dalam sistem ini meliputi mikrokontroler ESP32, modul relay, servo motor, sensor ultrasonik HC-SR04, indikator LED, serta dua buah pompa air untuk jalur air panas dan dingin.

c. Mekanisme Pemilihan Mode Operasi

Pengguna dapat memilih mode pengeluaran air dengan menekan tombol (push button) yang terhubung langsung ke ESP32. Sistem mengenali jumlah klik sebagai perintah mode berikut:

Tabel 3.4 Mode Operasi Dispenser Berdasarkan Jumlah Tekanan Push Button, Warna LED, dan Keterangan Fungsional

Jumlah Klik (Push Button)	Warna LED	Mode yang Diaktifkan	Keterangan
1 kali klik	Biru	Mode Air Dingin	Mengaktifkan pompa air dingin dan indikator LED biru
2 kali klik	Merah	Mode Air Panas	Mengaktifkan pemanas air dan indikator LED merah
3 kali klik	Putih	Mode Kombinasi (Dingin + Panas)	Mengaktifkan kedua jalur air dan indikator LED putih

Tabel tersebut menjelaskan mode operasi sistem dispenser berdasarkan konfigurasi warna LED dan jalur air yang diaktifkan. Setiap warna LED menunjukkan mode yang berbeda, yaitu air dingin (biru), air panas (merah), dan kombinasi keduanya (putih). Setelah mode aktif ditentukan, ESP32 akan mengatur status modul relay untuk menentukan pompa mana yang siap digunakan. Indikator LED menyala sebagai penanda visual bagi pengguna. Meskipun konfigurasi relay telah disesuaikan, pompa tidak akan aktif sebelum sensor memberikan validasi, seperti deteksi keberadaan gelas, guna memastikan pengeluaran air hanya terjadi saat diperlukan.

d. Aktivasi Otomatis Melalui Deteksi Sensor Ultrasonik

Setelah mode dipilih dan relay dalam kondisi siaga, sistem akan menunggu pemicu dari sensor ultrasonik HC-SR04. Sensor ini bertugas untuk mendeteksi keberadaan gelas di bawah keran dispenser. Deteksi hanya dilakukan pada jarak tertentu untuk menghindari pemicu palsu akibat gerakan tangan atau objek lain. Apabila gelas terdeteksi :

Tabel 3.5 Fungsi Komponen Dispenser Saat Gelas Terdeteksi

Komponen	Fungsi saat gelas terdeteksi
Servo Motor	Mengatur atau membuka jalur aliran air sesuai mode yang dipilih
Modul Relay	Mengaktifkan pompa air berdasarkan mode (dingin, panas, kombinasi)

Apabila gelas tidak terdeteksi lagi (misalnya gelas diambil oleh pengguna), sistem akan secara otomatis menonaktifkan pompa dan servo. Hal ini bertujuan untuk menjaga efisiensi penggunaan air serta menghindari tumpahan.

Prototipe Fisik

Perangkat IoT pada sistem dispenser pintar ini dirancang dengan mikrokontroler ESP32 sebagai pusat kendali yang mengintegrasikan seluruh sensor dan modul. Setiap komponen

diatur untuk mendukung fungsi utama, termasuk pemantauan suhu, dan *notifikasi real-time*, yang secara keseluruhan dirancang untuk memaksimalkan kenyamanan dan kemudahan pengguna.

1. Instalasi rangkaian IoT

Tabel 3.6 Konfigurasi Pin ESP32 unntuk Koneksi HX711 pada Sistem IoT Dispenser

No	Nama Komponen	Pin ESP32	Fungsi Pin ESP32	Deskripsi Komponen	Fungsi
1.	HX711 - DT	GPIO5(D5)	Digital Input	Jalur data sensor beban dari HX711 ke ESP32	Membaca nilai berat dari load cell
2.	HX711 – SCK	GPIO18(D18)	Digital Output	Jalur clock untuk sinkronisasi pembacaan data pada HX711	Mengatur pembacaan data beban
3.	Relay 1 Channel(Pompa) – IN 1	GPIO15(D15)	Digital Output	Pin kendali relay satu channel yang terhubung ke pompa air	Mengaktifkan atau mematikan pompa air
4.	Relay 2 – IN 1	GPIO26(D26)	Digital Output	Jalur kendali relay channel pertama untuk kompresor / pendingin	Mengendalikan kompresor (pendingin air)
5.	Relay 2 – IN 2	GPIO27(D27)	Digital Output	Jalur kendali relay channel kedua untuk pemanas air	Mengaktifkan atau mematikan elemen pemanas air
6.	LCD I2C - SDA	GPIO21(D21)	I2C SDA	Jalur data komunikasi I2C ke LCD	Mengirim data karakter ke layar LCD
7.	LCD I2C - SCL	GPIO22(D22)	I2C SCL	Jalur clock komunikasi I2	C ke LCDSinkronisasi komunikasi I2C
8.	Sensor Suhu DS18B20 #1 - IN	GPIO23(D23)	Wire Digital Input	Sensor suhu digital pertama (air panas)	Mendeteksi suhu air panas

9.	Sensor Suhu DS18B20 #2 - IN	GPIO19(D19)	Wire Digital Input	Sensor suhu digital kedua (air dingin)	Mendeteksi suhu air dingin
10.	WaterLevel Sensor - SIGNAL	GPIO14(D14) -	Analog / Digital Input	Sensor ketinggian air (level galon)	Mendeteksi apakah air dalam galon masih tersedia atau hampir habis
11.	GN D	GND	Ground (0V)	Ground sistem	Referensi tegangan untuk seluruh rangkaian
12.	VCC / 3.3V / 5V	3.3V atau 5V	Power Supply	Sumber tegangan untuk sensor dan modul	Menyediakan tegangan operasi ke sensor dan modul

Modul HX711 digunakan untuk membaca berat galon melalui *load cell*, sementara dua sensor suhu DS18B20 memantau suhu air panas dan dingin. Sistem juga dilengkapi dengan relay untuk mengontrol pompa, pemanas, dan pendingin air secara otomatis. Tampilan informasi dilakukan melalui LCD I2C yang terhubung dengan jalur komunikasi digital. Selain itu, sensor ketinggian air digunakan untuk mendeteksi volume galon tersisa. Semua komponen disuplai dengan tegangan 3.3V atau 5V dan terhubung ke ground (GND), sehingga membentuk sistem terintegrasi yang dapat dipantau dan dikendalikan secara *real-time* melalui aplikasi *mobile*.

2. Pemilihan Library

Tabel 3.7 Daftar Library yang Digunakan pada Pemrograman ESP32 Sistem Dispenser IoT

No	**Nama Library**	**Kegunaan/Fungsi Utama**
1.	*Wifi*.h	Menghubungkan ESP32 ke jaringan *Wifi*
2.	*Firebase*_ESP_Client.h	Berkomunikasi dengan *Firebase* Realtime Database menggunakan REST API
3.	HX711.h	Membaca data dari sensor beban (*load cell*) melalui modul HX711
4.	LiquidCrystal_I2C.h	Menampilkan data ke LCD I2C 16x2
5.	ESP32Servo.h	Mengendalikan servo motor khusus untuk board ESP32

Tabel di atas menjelaskan kumpulan library yang diperlukan untuk menjalankan sistem *Smart water dispenser* berbasis ESP32. Setiap library memiliki peran penting dalam mendukung fungsi utama perangkat.

Library *Wifi*.h digunakan untuk menghubungkan mikrokontroler ESP32 ke jaringan *Wifi*, sementara *Firebase*_ESP_Client.h memungkinkan pengiriman dan penerimaan data secara *real-time* ke *Firebase Realtime Database*. Untuk membaca berat air dalam galon, sistem menggunakan HX711.h yang berfungsi mengakses modul sensor beban (*load cell*). Tampilan data seperti berat dan status air ditampilkan melalui LCD I2C dengan bantuan library LiquidCrystal_I2C.h. Selain itu, ESP32Servo.h digunakan untuk mengontrol gerakan motor servo yang mendukung proses otomatisasi pengeluaran air saat gelas terdeteksi.

Penggunaan library-library ini memastikan sistem berjalan secara efisien, terhubung ke *cloud*, dan mampu beroperasi secara otomatis serta interaktif sesuai kebutuhan pengguna.

3. Integrasi Sensor

Tahap selanjutnya dalam implementasi perangkat IoT adalah integrasi sensor ke dalam sistem. Integrasi ini bertujuan agar setiap sensor dapat bekerja secara harmonis dengan mikrokontroler ESP32, mendukung sistem pemantauan, otomatisasi, serta pengiriman data secara *real-time* ke *cloud* dan aplikasi *mobile*.

Sensor pertama yang diintegrasikan adalah DS18B20, yang berfungsi untuk memantau suhu air panas dan air dingin dalam dispenser. Sensor ini dikoneksikan ke pin digital ESP32 menggunakan protokol komunikasi 1-Wire, yang memungkinkan penggunaan hanya satu jalur data untuk beberapa sensor sekaligus. Data suhu yang diperoleh digunakan untuk mengontrol kinerja elemen pemanas air dan kompresor pendingin, serta dikirimkan ke *Firebase* agar dapat dimonitor oleh pengguna melalui aplikasi.

Selanjutnya, *Load cell* yang terhubung melalui modul HX711 digunakan untuk mengukur berat galon air. *Load cell* dipasang di bawah galon untuk mendapatkan pembacaan berat secara akurat. Data berat ini kemudian digunakan untuk memperkirakan volume air yang tersisa, sehingga sistem dapat mengirimkan *notifikasi* otomatis ketika galon hampir habis. Modul HX711 berfungsi sebagai penguat sinyal dan konverter analog ke digital, sehingga pembacaan berat dapat dilakukan dengan resolusi tinggi.

Sensor Ultrasonik HC-SR04 diintegrasikan untuk mendeteksi keberadaan gelas di bawah keran. Sensor ini memanfaatkan gelombang ultrasonik untuk mengukur jarak antara sensor dan gelas. Ketika gelas terdeteksi dalam jarak tertentu, ESP32 akan mengaktifkan pompa air untuk mengalirkan air ke gelas secara otomatis, dan mematikan pompa jika gelas tidak terdeteksi lagi. Mekanisme ini memungkinkan sistem bekerja tanpa sentuhan langsung dari pengguna, meningkatkan aspek higienitas.

Untuk memperkuat sistem *monitoring* volume air, digunakan pula Water Level Sensor, yang memantau tinggi air dalam galon. Sensor ini bekerja sebagai tambahan validasi untuk mendukung data dari *load cell*, memastikan bahwa sistem dapat memberikan informasi yang lebih akurat mengenai status ketersediaan air.

Semua data yang diperoleh dari sensor-sensor tersebut diolah oleh ESP32 dan digunakan baik untuk pengendalian aktuator lokal seperti relay dan pompa, maupun untuk pengiriman data ke *Firebase* agar dapat diakses oleh aplikasi *mobile*. Dengan integrasi sensor yang sinkron dan responsif ini, sistem *Smart water dispenser* mampu memberikan layanan air minum otomatis berbasis IoT yang lebih akurat, efisien, dan ramah pengguna.

4. Integrasi dengan *Firebase*

Tahap terakhir dalam perancangan perangkat IoT adalah integrasi dengan *Firebase* sebagai platform cloud untuk menyimpan, mengelola, dan mengirimkan data sistem secara *real-time*. *Firebase Realtime Database* dipilih karena kemampuannya dalam menyediakan sinkronisasi data secara instan antara perangkat ESP32 dan aplikasi *mobile* pengguna.

Integrasi dimulai dengan pembentukan koneksi Wi-Fi pada ESP32 menggunakan library *Wifi*.h. Setelah koneksi internet berhasil dibangun, ESP32 dikonfigurasi untuk terhubung dengan *Firebase* menggunakan library *Firebase* ESP Client. Konfigurasi ini mencakup pengisian URL database, secret key untuk autentikasi, dan pengaturan koneksi yang stabil agar dapat menjaga sinkronisasi data secara terus menerus.

Setiap data yang dihasilkan dari sensor suhu DS18B20, *load cell* HX711, sensor ultrasonik, maupun water level sensor, dikirimkan ke *Firebase* dalam format JSON. Data tersebut meliputi suhu air panas dan dingin, volume galon, status keberadaan gelas, hingga status aktif atau tidaknya pompa dan pemanas. Melalui *Firebase*, data ini kemudian ditampilkan secara *real-time* di aplikasi *mobile* yang dikembangkan khusus untuk pengguna dispenser.

Selain mengirimkan data, ESP32 juga menerima perintah dari *Firebase*. Pengguna dapat mengatur suhu air, memilih mode operasi dispenser, atau melakukan reset sistem langsung dari aplikasi *mobile*. *Firebase* menjadi perantara untuk menyampaikan perintah tersebut secara cepat dan aman ke ESP32.

Salah satu fitur penting dari integrasi ini adalah *notifikasi real-time*. Ketika volume air galon terdeteksi hampir habis atau suhu air mencapai tingkat yang telah ditentukan, *Firebase* secara otomatis memperbarui status data, dan sistem *mobile* memberikan *notifikasi* langsung kepada pengguna. Mekanisme ini memungkinkan pengguna untuk mendapatkan informasi penting tanpa harus memeriksa perangkat secara manual.

Dengan adanya integrasi penuh antara ESP32 dan *Firebase,* sistem *Smart water dispenser* mampu menghadirkan *monitoring* jarak jauh, kontrol otomatis, serta respons instan terhadap kondisi kritis, yang semuanya dirancang untuk meningkatkan kenyamanan, keamanan, dan efisiensi penggunaan dispenser air berbasis IoT.

Perancangan Aplikasi *Mobile*

Aplikasi *mobile* dalam sistem *Smart water dispenser* berbasis IoT dirancang sebagai antarmuka utama yang memungkinkan pengguna untuk melakukan pemantauan kondisi dispenser secara *real-time,* menerima *notifikasi* kritis, serta mengontrol mode pengeluaran air. Aplikasi ini dikembangkan menggunakan Framework *React Native* dengan dukungan *Expo,* sehingga dapat diakses pada platform *Android* dengan efisiensi pengembangan yang tinggi dan dukungan terhadap integrasi *cloud* secara langsung melalui *Firebase Realtime Database*.

Perancangan aplikasi dilakukan dengan mengacu pada prinsip antarmuka minimalis, informatif, dan responsif, serta mempertimbangkan kebutuhan utama pengguna yang diperoleh dari analisis kebutuhan pada tahap awal iterasi.

1. Pemilihan Library Aplikasi

   Aplikasi *mobile* dalam sistem *Smart water dispenser* berbasis IoT dirancang untuk mendukung beberapa fitur utama, seperti *monitoring real-time*, *notifikasi* suhu dan galon, serta pengaturan mode dispenser. Untuk mengimplementasikan fitur-fitur tersebut secara efisien dan andal, berbagai pustaka (library) dipilih secara selektif berdasarkan kompatibilitas dengan platform *React Native Expo* dan kemampuannya mendukung *Firebase Realtime Database* sebagai backend sinkronisasi data. Berikut adalah pemetaan antara fitur utama dan library yang digunakan dalam aplikasi:

Tabel 3.8 Pemetaan Fitur Utama Aplikasi *Mobile* dengan Library yang Digunakan

No	Fitur Utama Aplikasi	Library yang Digunakan
1.	*Monitoring Real-time* (status suhu & galon)	*Firebase*/database, *Firebase*/app
2.	*Notifikasi* Galon Hampir Habis dan Suhu Siap	*Expo*-notifications
3.	Dashboard Status Dispenser	react-native, react-native-paper / shadcn/ui
4.	Pemilihan Mode Dispenser (dingin/panas/kombinasi)	*Firebase*/database, react-native-gesturehandler
5.	Visualisasi Status (warna indikator, ikon status)	react-native-svg, react-native-vectoricons
6.	Navigasi antar halaman (Dashboard, *Device*, Setting)	@react-navigation/native, @reactnavigation/stack
7.	Pemrosesan Event *Real-time*	*Firebase*/database (listener: onValue)

2. Dashboard Utama

Gambar 3.30 Wireframe Dashboard Utama pada Aplikasi *Mobile Smart water dispenser*

Dashboard utama adalah layar awal aplikasi yang menyajikan informasi penting tentang dispenser air pintar. Halaman ini dirancang untuk memberikan informasi sekilas tentang status dispenser tanpa memerlukan navigasi lebih lanjut. Fitur utama yang ditampilkan di dashboard meliputi:

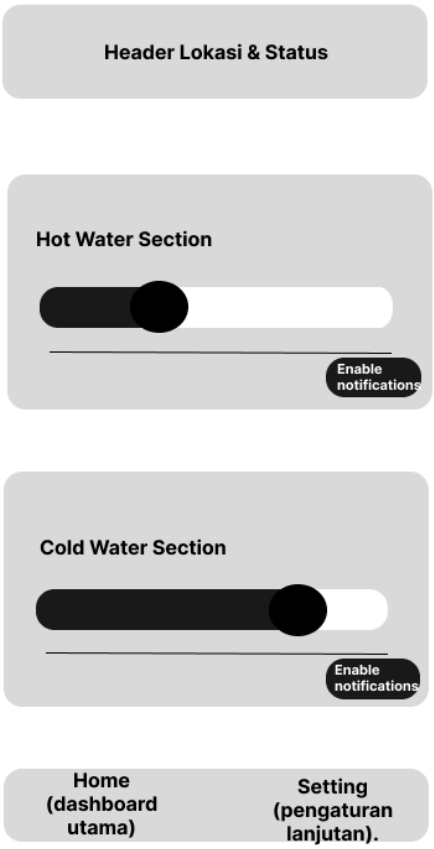
a. Status Galon

Status level air dalam galon ditampilkan melalui ikon galon yang beradaptasi dengan kondisi air (penuh, menengah, atau hampir kosong). Jika galon hampir kosong, ikon akan berubah warna, dan *notifikasi* akan muncul untuk memperingatkan pengguna agar segera galon

b. Suhu Air

Status suhu air (panas, dingin, atau normal) dari dispenser ditampilkan langsung di dashboard, memungkinkan pengguna untuk memeriksa suhu tanpa perlu membuka pengaturan. Informasi ini memudahkan pengguna untuk mengetahui kondisi suhu air sebelum digunakan.

3. Menu Pengaturan Suhu



Gambar 3.31 Tampilan Fitur *Notifikasi* Suhu Air pada Aplikasi *Mobile Smart water dispenser*

Fitur *notifikasi* suhu air galon dirancang untuk memberikan kenyamanan maksimal kepada pengguna dengan memberikan informasi secara *real-time* mengenai kesiapan suhu air.

Melalui halaman khusus pada aplikasi, pengguna dapat melihat status suhu air dan akan menerima *notifikasi* otomatis ketika air panas atau dingin telah mencapai suhu ideal untuk digunakan. *Notifikasi* ini muncul segera setelah sistem mendeteksi bahwa proses pemanasan atau pendinginan telah selesai, sehingga pengguna tidak perlu menebak atau memeriksa secara manual. Fitur ini memastikan pengguna selalu mendapatkan air dengan suhu yang tepat pada waktu yang tepat.

4. *Notifikasi*

   Fitur *Notifikasi* dalam aplikasi ini bertujuan untuk memberikan peringatan dan informasi penting secara *real-time* kepada pengguna. *Notifikasi* ini mencakup peringatan terkait kebutuhan pemeliharaan dan status galon, memastikan pengguna selalu diperbarui mengenai kondisi dispenser. Fitur *notifikasi* meliputi:

   a. *Notifikasi* Galon Kosong

      Pengguna akan menerima pemberitahuan saat galon air mendekati kondisi kosong. *Notifikasi* ini memungkinkan pengguna untuk segera mengganti galon sebelum air benar-benar habis, sehingga penggunaan dispenser tetap nyaman dan lancar.

   b. *Notifikasi* Suhu Air Siap digunakan

      Aplikasi akan memberikan *notifikasi* secara otomatis ketika suhu air panas atau dingin telah mencapai tingkat yang ideal untuk dikonsumsi. Informasi ini memungkinkan pengguna mengetahui kesiapan air tanpa harus memeriksa suhu secara manual, sehingga meningkatkan efisiensi dan kenyamanan penggunaan, terutama saat pengguna membutuhkan air dalam kondisi tertentu secara cepat.

Interasi IoT dengan Aplikasi *Mobile* Melalui *Firebase*

Dispenser berbasis IoT. *Firebase* bertindak sebagai jembatan komunikasi antara perangkat keras ESP32 dengan aplikasi *mobile* berbasis *React Native*, memungkinkan sinkronisasi data secara *real-time* yang efisien dan stabil. *Firebase* mendukung model arsitektur publish–subscribe, di mana data dari ESP32 ditulis ke *cloud* dan langsung dikonsumsi oleh aplikasi pengguna, serta sebaliknya.

1. Struktur Data dan *Firebase*

   Firebase menyimpan data dalam format JSON yang fleksibel dan mudah diakses. Struktur data ini memungkinkan akses serentak oleh berbagai klien (ESP32 dan aplikasi mobile), serta memastikan bahwa setiap perubahan data yang terjadi akan secara otomatis disinkronkan ke seluruh perangkat yang terhubung. Contoh struktur data yang digunakan dalam sistem ini sebagai berikut:

```
{
  "dispensers": {
    "dispenser_1": {
      "capacity": {
        "current": 12,
        "max": 16
      },
      "code": "code_dispenser_1",
      "cold": {
        "showNotifications": false,          "temperature": 13
      },
      "hot": {
        "showNotifications": true,
        "temperature": 74
      },
      "id": "dispenser_1",
      "name": "Dispenser Nih Bos!",
      "tasks": {
        "1746949552488": {
          "completed": true,
          "from": 66,
          "target": 68,
          "timestamp": 1746949552488,
          "type": "HOT"
        }
      }
    }
  }
}
```


2. Alur data dari ESP32 ke *Firebase* dan Aplikasi *Mobile*

   Proses sinkronisasi data antara ESP32, Firebase, dan aplikasi mobile melibatkan dua komponen utama: pemantauan suhu, serta notifikasi real-time. Alur data ini memastikan data dapat diakses pengguna secara real-time melalui aplikasi mobile.

   a. DS18B20 untuk suhu air panas dan dingin,
   b. *Load cell* + HX711 untuk mendeteksi berat galon,
   c. Sensor ultrasonik untuk mendeteksi keberadaan gelas.

Data hasil pembacaan ini kemudian dikirim ke Firebase menggunakan pustaka FirebaseESPClient.h. Ketika terjadi perubahan data, Firebase secara otomatis memperbarui node JSON terkait.

Aplikasi mobile, yang menggunakan listener onValue() dari pustaka Firebase di React Native, akan langsung menerima pembaruan dan menampilkan status terbaru kepada pengguna di dashboard aplikasi.

3. Alur Data dari Aplikasi *Mobile* ke *Firebase* dan ESP32

   Selain menerima data, aplikasi juga dapat mengirimkan perintah atau preferensi ke Firebase, seperti:

   a. Mengubah mode pengeluaran air (dingin/panas/kombinasi),
   b. Mengatur ambang batas suhu air,
   c. Mengatur preferensi *notifikasi*,
   d. Me*monitoring* Volume air galon

   Firebase kemudian menyimpan perintah ini pada node tertentu (misalnya dispenser/perintah). ESP32, melalui mekanisme polling atau stream, akan membaca data tersebut dan menjalankan perintah sesuai dengan logika program yang telah ditanamkan.

4. Pemantauan Status Galon

   Sistem menggunakan load cell untuk menghitung berat galon dan menentukan apakah galon hampir kosong. Ketika berat galon di bawah ambang tertentu (misalnya < 3 kg), ESP32 akan mengirim status "hampir kosong" ke Firebase. Aplikasi mobile yang memantau node ini akan :

   a. Mengubah ikon status galon,
   b. Menampilkan peringatan visual,
   c. Mengirim *notifikasi* push kepada pengguna, seperti:

   "Galon hampir kosong. Silakan ganti galon untuk memastikan ketersediaan air."

Alur Kerja Integrasi

Gambar 3.32 Diagram Alur Kerja Integrasi Data antara ESP32, *Firebase*, dan Aplikasi *Mobile*

Sistem ini dirancang agar dapat beroperasi dalam dua arah antara perangkat keras dan aplikasi pengguna melalui *Firebase*. Alur kerja utamanya dibagi menjadi dua aspek:

1. Pemantauan dan Kontrol
   a. Pemantauan dilakukan secara pasif, di mana aplikasi membaca status suhu, volume air, dan mode aktif dari *Firebase*.
   b. Kontrol dilakukan secara aktif, di mana aplikasi dapat mengirimkan perintah seperti mengatur mode dispenser atau menyetel suhu target tertentu.
2. *Notifikasi* dan respons *real-time*

   Fitur notifikasi real-time merupakan salah satu komponen penting dalam sistem Smart water dispenser berbasis IoT yang dikembangkan. Fungsi ini dirancang untuk memberikan informasi kritis secara instan kepada pengguna tanpa memerlukan interaksi aktif dari sisi pengguna. Dengan mengandalkan teknologi listener berbasis event yang ditanamkan pada Firebase Realtime Database, sistem ini mampu memberikan respons otomatis terhadap perubahan kondisi perangkat dan menyampaikannya dalam bentuk local push notification di aplikasi mobile:

   a. Jenis *Notifikasi* Utama

      Sistem ini mengimplementasikan dua jenis *notifikasi* yang dianggap paling relevan berdasarkan hasil analisis kebutuhan pengguna, yaitu:

      1) *Notifikasi* Suhu Siap Digunakan

         Notifikasi ini akan dikirim secara otomatis saat sensor suhu DS18B20 mendeteksi bahwa air panas telah mencapai ambang suhu tertentu, misalnya ≥75°C, atau ketika suhu air dingin telah turun hingga ≤10°C. Pengguna akan menerima pesan seperti: “Air panas telah siap digunakan.” atau “Air dingin siap disajikan.”. *Notifikasi* ini bertujuan memberikan kemudahan bagi pengguna, terutama di lingkungan padat aktivitas seperti kampus atau kantor, agar mereka tidak perlu memantau dispenser secara manual.

      2) *Notifikasi* Galon Hampir Kosong

         Berdasarkan data dari sensor *load cell* yang telah dikalibrasi, sistem akan mengirimkan peringatan jika berat galon berada di bawah ambang batas tertentu (misalnya < 3 kg). Kondisi ini menandakan bahwa galon hampir kosong dan perlu segera diganti. *Notifikasi* yang muncul berupa pesan seperti: “Galon hampir habis, mohon segera diganti.” Fitur ini dirancang untuk mengurangi kejadian kehabisan air secara tibatiba, yang sering menjadi keluhan pengguna dispenser konvensional.

   b. Mekanisme Pengiriman *Notifikasi*

      Proses pengiriman *notifikasi* melalui beberapa tahapan berikut:

1) Sensor Membaca Data

   Mikrokontroler ESP32 secara berkala membaca data dari sensor suhu dan *load cell*.

2) Pengiriman Data ke *Firebase*

   Jika data yang dibaca melampaui ambang batas tertentu (misalnya suhu ≥75°C atau berat ≤3kg), maka ESP32 memperbarui node data pada *Firebase Realtime Database*. Perubahan ini dilakukan melalui endpoint HTTP atau WebSocket dengan struktur JSON.

3) Listener *Firebase* di Aplikasi

   Aplikasi *React Native* menggunakan listener onValue() untuk memantau perubahan pada node data tertentu di *Firebase*. Listener ini bekerja secara *real-time* dan akan memicu aksi begitu data mengalami perubahan.

4) Pemicu Local Push Notification

   Setelah listener mendeteksi perubahan, aplikasi akan menggunakan modul *Expo-*notifications untuk mengirimkan *notifikasi* lokal kepada pengguna. Fitur ini tetap berfungsi meskipun aplikasi berada dalam mode background atau tidak aktif, selama izin *notifikasi* telah diberikan sebelumnya.

3.4.2 *Measure*

Tahap *Measure* bertujuan untuk mengumpulkan data awal tentang pengalaman dan tanggapan pengguna terhadap MVP dispenser air pintar berbasis IoT. Pada tahap ini, kami melakukan pengujian langsung kepada pengguna melalui wawancara dan uji coba untuk mengamati interaksi dan mendapatkan masukan langsung. Data yang dikumpulkan akan digunakan sebagai dasar untuk melakukan penyempurnaan produk di tahap berikutnya.

1. Tujuan Pengujian MVP

   Tujuan dari pengujian MVP adalah untuk:

   a. Mengidentifikasi apakah fitur-fitur dasar MVP, seperti *notifikasi* suhu, dan *notifikasi* galon kosong, telah sesuai dengan kebutuhan dan harapan pengguna.
   b. Menguji apakah pengguna dapat memahami dan mengoperasikan fungsi utama pada dispenser dengan mudah, serta memeriksa tingkat kenyamanan dalam penggunaan aplikasi.
   c. Memperoleh tanggapan pengguna terkait kepraktisan, kenyamanan, dan efisiensi dispenser dalam kehidupan sehari-hari. Umpan balik ini akan sangat berharga dalam mengidentifikasi area yang memerlukan perbaikan.

2. Metode Pengujian

   Pengujian MVP dilakukan secara langsung dengan melibatkan pengguna melalui dua metode utama, yaitu:

   a. Wawancara Singkat

      Dilakukan sebelum dan sesudah uji coba. Wawancara awal digunakan untuk memahami ekspektasi pengguna terhadap dispenser pintar, sedangkan wawancara pasca-pengujian bertujuan untuk mendapatkan tanggapan pengguna terhadap pengalaman menggunakan perangkat.

   b. Uji Coba Langsung

      Pengguna diminta mencoba fitur utama dispenser, seperti *notifikasi* suhu, dan *notifikasi* galon kosong, melalui aplikasi *mobile*. Uji coba ini dilakukan untuk menilai interaksi pengguna dengan fitur-fitur tersebut dalam skenario penggunaan nyata.

3. Prosedur Pengujian

   a. Pengenalan Sistem

      Sebelum pengujian, pengguna diberikan penjelasan singkat mengenai fitur utama dispenser, termasuk *notifikasi* suhu air, *notifikasi* galon kosong, dan penggunaan antarmuka aplikasi *mobile*.

   b. Simulasi Pengguna

      Pengguna diberikan panduan singkat tentang cara mengoperasikan aplikasi *mobile* untuk mengontrol dispenser. Setelah instruksi diberikan, pengguna diminta untuk mencoba fitur-fitur tersebut secara mandiri untuk mendapatkan pengalaman nyata dalam menggunakan perangkat.

   c. Pengujian Fitur Dispenser (re)

      Pengguna diminta menguji fitur utama dispenser dengan rincian sebagai berikut:

      1) *Notifikasi* Suhu Air

         Pengguna akan menerima *notifikasi* melalui aplikasi ketika suhu air gallon baik panas maupun dingin telah mencapai tingkat yang siap digunakan. Pengujian fitur ini menilai keakuratan sistem dalam mendeteksi suhu yang telah optimal serta kecepatan pengiriman *notifikasi* ke perangkat pengguna. Dengan adanya *notifikasi* ini, pengguna tidak perlu terus memantau dispenser secara manual, sehingga memberikan pengalaman penggunaan yang lebih praktis dan efisien.

2) *Notifikasi* Galon Kosong

Setelah dilakukan simulasi galon kosong, pengguna diminta untuk memperhatikan *notifikasi* yang muncul di aplikasi. Pengujian ini bertujuan untuk memastikan bahwa *notifikasi* galon kosong cukup efektif dan jelas sebagai peringatan bagi pengguna.

3.4.3 *Learn*

Tahap *Learn* bertujuan untuk mengevaluasi data dan umpan balik yang dikumpulkan dari uji coba MVP. Berdasarkan hasil analisis ini, kami menyusun rencana iterasi berikutnya untuk mengoptimalkan kinerja dan fungsionalitas produk agar semakin sesuai dengan kebutuhan dan ekspektasi pengguna. Melalui evaluasi yang sistematis, tahap ini memungkinkan pengembangan berkelanjutan yang berfokus pada penyempurnaan fitur serta peningkatan kualitas keseluruhan produk.

1. Analisis Umpan Balik Pengguna

   Secara umum, pengguna memberikan tanggapan positif terhadap fitur-fitur utama yang telah diimplementasikan, seperti notifikasi suhu air dan notifikasi galon kosong. Fitur-fitur tersebut dinilai memberikan kemudahan, meningkatkan kenyamanan, serta menjawab tantangan umum yang sering dialami pengguna dalam menggunakan dispenser air konvensional. Pengalaman pengguna selama pengujian juga menunjukkan bahwa sistem relatif mudah dipahami dan dioperasikan. Tampilan aplikasi mobile dianggap cukup informatif dan antarmuka pengguna tergolong intuitif. Namun demikian, terdapat pula beberapa masukan yang menunjukkan adanya aspek yang perlu ditingkatkan, seperti stabilitas sensor dalam mendeteksi kondisi secara akurat serta kecepatan notifikasi yang terkadang tidak konsisten. Umpan balik ini menjadi masukan penting untuk pengembangan iterasi selanjutnya.

2. Rencana Tindak Lanjut

   Berdasarkan hasil evaluasi tersebut, beberapa langkah strategis disusun untuk diterapkan pada iterasi berikutnya, antara lain:

   a. Peningkatan Akurasi dan Stabilitas Sistem

      Dilakukan perbaikan pada aspek yang masih menunjukkan kelemahan dari sisi keandalan dan presisi, khususnya yang berdampak pada kualitas *notifikasi* dan otomatisasi sistem.

   b. Penguatan Mekanisme Umpan Balik

      Disusun sistem pengumpulan masukan pengguna secara lebih terstruktur untuk

memastikan bahwa setiap pembaruan sistem didasarkan pada kebutuhan nyata pengguna.

c. Penyempurnaan Kualitas Pengalaman Pengguna

Melalui penyederhanaan navigasi, peningkatan kecepatan sistem, dan optimisasi *notifikasi*, diharapkan interaksi pengguna dengan sistem dapat menjadi lebih nyaman dan responsif.

3.5 Iterasi 2

Berdasarkan masukan dan analisis pada tahap *Learn* dari iterasi pertama, iterasi kedua pengembangan MVP (Minimum Viable Product) ini berfokus pada penyempurnaan konektivitas dan sinkronisasi data. Perbaikan ini bertujuan untuk memastikan bahwa perangkat dapat memberikan pengalaman penggunaan yang lebih mulus, responsif, dan andal. Berikut adalah fokus utama dalam tahap *Build* iterasi kedua:

3.5.1 *Build*

Iterasi kedua dilaksanakan sebagai kelanjutan langsung dari proses evaluasi pada iterasi pertama. Fokus utama pada tahap ini adalah peningkatan akurasi dan konsistensi pembacaan sensor, serta penyesuaian fitur berdasarkan masukan pengguna dan hasil pengujian sebelumnya. Pengembangan pada iterasi ini tidak lagi menambah fitur baru, melainkan menyempurnakan fitur yang telah ada agar lebih stabil, responsif, dan sesuai dengan kondisi penggunaan riil.

1. Tujuan Pengembangan Iterasi Kedua

   Tujuan dari tahap Build pada iterasi kedua meliputi:

   a. Meningkatkan presisi deteksi sensor ultrasonik terhadap objek (gelas).
   b. Mengoptimalkan stabilitas pembacaan berat galon menggunakan sensor *load cell*.
   c. Menyempurnakan proses validasi sensor secara lokal pada ESP32 agar lebih tangguh terhadap noise dan fluktuasi lingkungan.

2. Kalibrasi dan Pengujian Ulang Sensor

   Langkah kalibrasi dan pengujian ulang sensor bertujuan untuk meningkatkan akurasi data yang dikirimkan oleh perangkat, khususnya dalam hal deteksi gelas dan pembacaan berat galon. Proses ini penting untuk mengatasi hasil pembacaan yang tidak konsisten selama pengujian pada iterasi pertama.

   a. Kalibrasi Sensor Ultrasonik

      Kalibrasi ulang dilakukan terhadap sensor ultrasonik HC-SR04 untuk memastikan

bahwa jarak deteksi terhadap gelas berada dalam rentang ideal. Pengujian dilakukan dengan berbagai jenis gelas dan kondisi cahaya untuk memperoleh parameter pengukuran yang stabil. Penyesuaian dilakukan melalui perangkat lunak dengan filter logika dan ambang batas pembacaan.

b. Pengujian Konsistensi Sensor *Load cell*

Sensor beban yang sebelumnya menunjukkan fluktuasi saat membaca berat galon, dikalibrasi ulang menggunakan beban standar (5 kg, 10 kg) dan dicek stabilitas pembacaan pada permukaan datar. Nilai kalibrasi baru disesuaikan agar data lebih stabil dan dapat diandalkan.

3.5.2 *Measure*

Pada tahap ini, pengujian lanjutan dilakukan untuk memastikan bahwa semua peningkatan yang diterapkan dalam iterasi kedua berfungsi secara optimal dan mampu memperbaiki kelemahan pada iterasi sebelumnya. Fokus pengujian diarahkan pada peningkatan performa sensor dan validitas data yang ditampilkan dalam aplikasi.

1. Tujuan Pengujian Lanjutan

   Pengujian ini bertujuan untuk mengevaluasi ketepatan pembacaan sensor ultrasonik dan load cell setelah proses kalibrasi ulang. Fokus utama adalah pada kestabilan pembacaan dalam skenario penggunaan nyata untuk memastikan sistem mampu memberikan informasi yang valid dan dapat dipercaya oleh pengguna.

2. Metode Pengujian

   Pengujian dilakukan dengan melibatkan pengguna dalam pengoperasian sistem seperti biasa. Dalam uji coba ini:

   a. Pengguna mendekatkan gelas ke dispenser untuk menguji respons sensor ultrasonik. Keberhasilan pengaktifan pompa dicatat berdasarkan konsistensi deteksi gelas dari berbagai sudut.

   b. Galon diuji dengan beban bertahap, dan data dari sensor *load cell* dibandingkan dengan timbangan digital eksternal. Pengguna juga diminta menilai seberapa relevan dan responsif *notifikasi* galon kosong yang ditampilkan dalam aplikasi.

3. Prosedur Pengujian

   Setiap sensor diuji secara sistematis. Sensor ultrasonik diuji pada berbagai kondisi cahaya dan gelas, sedangkan *load cell* diuji pada interval berat tertentu. Hasil dicatat dan dianalisis secara kuantitatif untuk melihat deviasi serta tingkat kestabilannya.

3.5.3 *Learn*

Tahap *Learn* ini digunakan untuk menganalisis hasil pengujian dari iterasi kedua dan menilai efektivitas peningkatan sistem, khususnya dalam aspek pembacaan sensor otomatis. Hasil ini menjadi dasar pertimbangan untuk pengembangan pada iterasi berikutnya.

1. Analisis Hasil Pengujian
   a. Peningkatan Akurasi Sensor Ultrasonik

      Penggunaan ambang batas dan logika penyaringan sederhana terbukti efektif mengurangi kesalahan deteksi gelas. Sensor tidak lagi memicu pompa akibat objek selain gelas, dan deteksi menjadi lebih stabil meskipun gelas diletakkan miring atau berbahantransparan. Rata-rata kesalahan deteksi menurun dari estimasi ±20% pada iterasi pertama menjadi kurang dari ±5% pada iterasi kedua.

   b. Stabilitas Pembacaan *Load cell*

      Kalibrasi ulang dengan beban bertingkat dan penyesuaian calibration factor menghasilkan pembacaan berat yang jauh lebih konsisten. Deviasi rata-rata pembacaan terhadap timbangan digital berada di kisaran ±3%, menunjukkan peningkatan signifikan dibandingkan pembacaan yang fluktuatif pada iterasi sebelumnya.

   c. Respon Sistem

      Validasi lokal pada ESP32 dengan pengambilan keputusan berbasis beberapa kali pembacaan terbukti mengurangi risiko aksi sistem yang salah akibat pembacaan sensor yang bersifat sesaat. Pompa hanya aktif jika deteksi gelas stabil, dan *notifikasi* galon hanya muncul jika pembacaan *load cell* konsisten dalam beberapa siklus.

2. Refleksi Pengembangan Sistem

   Secara keseluruhan, iterasi kedua menunjukkan bahwa sistem telah berhasil mencapai tujuan pengembangan yang lebih stabil, akurat, dan layak untuk diuji dalam skala pengguna yang lebih luas. Tidak terdapat kebutuhan mendesak untuk menambahkan fitur baru, melainkan hanya pematangan terhadap integrasi perangkat dan validasi logika internal.

   Keputusan untuk tidak mengoptimalkan komunikasi *Firebase* pada iterasi ini terbukti tidak menghambat proses validasi utama, karena logika pemrosesan dan verifikasi berbasis lokal pada ESP32 sudah cukup untuk memastikan sistem berjalan dengan baik bahkan tanpa pembaruan data yang sangat tinggi frekuensinya.

3. Tindak Lanjut untuk Iterasi Selanjutnya

a. Optimalisasi *Notifikasi*

   Apabila *notifikasi* masih mengalami keterlambatan, perlu diterapkan algoritma yang lebih efisien untuk mengirimkan *notifikasi* melalui *Firebase*, sehingga proses sinkronisasi data dan pengiriman *notifikasi* dapat dilakukan lebih cepat dan stabil.

b. *Feedback* Loop

   Mengintegrasikan mekanisme pengumpulan umpan balik secara otomatis di dalam aplikasi untuk menangkap masukan pengguna secara langsung setelah menggunakan fitur tertentu. Dengan adanya *feedback* loop ini, tim pengembang dapat memperoleh insight *real-time* terkait pengalaman pengguna dan menyempurnakan fitur sesuai kebutuhan.

c. Penyesuaian Sistem Koneksi

   Jika kendala koneksi masih ditemukan, maka perlu dilakukan pengujian tambahan pada perangkat jaringan atau mengganti sistem komunikasi jika diperlukan. Hal ini bertujuan untuk meningkatkan stabilitas dan konsistensi koneksi data antara perangkat dan aplikasi *mobile*.

d. Peningkatan Ketahanan Sensor

   Jika masalah pada sensor masih terjadi, perlu dipertimbangkan untuk mengganti komponen sensor atau menggunakan sensor dengan spesifikasi yang lebih tinggi guna mencapai tingkat akurasi dan daya tahan yang lebih baik dalam berbagai kondisi air.

Tahap *Learn* ini akan menjadi landasan utama bagi iterasi ketiga atau pengembangan tambahan, memastikan dispenser air pintar ini dapat terus berkembang sesuai harapan pengguna dan meningkatkan kualitas fitur secara berkelanjutan.

BAB IV
HASIL DAN PEMBAHASAN

Pada bab ini, hasil eksperimen dan pengujian yang telah dilakukan selama penelitian *Smart water dispenser* berbasis *Internet of Things (IoT)* akan dibahas secara sistematis dan mendalam. Pembahasan mengacu pada metodologi *Build–Measure–Learn* yang telah dijelaskan di Bab III, di mana sistem dikembangkan dan diuji melalui dua siklus iterasi untuk mencapai desain yang lebih efisien, praktis, serta responsif terhadap kebutuhan pengguna.

Analisis hasil pengujian dalam bab ini dibagi menjadi dua aspek utama, yaitu hasil kuantitatif dan hasil kualitatif. Hasil kuantitatif diperoleh melalui pengukuran akurasi sensor, waktu respons sistem, dan data numerik lainnya yang merefleksikan kinerja perangkat keras maupun perangkat lunak.

Sementara itu, hasil kualitatif mencakup pengalaman dan persepsi pengguna yang diperoleh melalui survei serta rangkuman hasil wawancara langsung, yang dilaporkan secara lebih rinci pada subbab *Learn*. Dengan demikian, laporan ini tidak hanya menilai performa teknis sistem, namun juga mengevaluasi efektivitas implementasi dari sudut pandang pengguna.

Bab ini disusun berdasarkan dua siklus pengembangan sistem:

1. Iterasi pertama menekankan pada pembangunan *Minimum viable product (MVP)* yang mengimplementasikan fitur dasar seperti *notifikasi* volume galon, pemantauan suhu air, dan pengeluaran air otomatis berbasis deteksi gelas. Proses pengujian pada iterasi pertama bertujuan mengidentifikasi kelebihan, kekurangan, serta area perbaikan sistem.
2. Iterasi kedua difokuskan pada penyempurnaan fitur dan peningkatan performa sistem berdasarkan hasil evaluasi dan umpan balik pengguna pada siklus sebelumnya. Pada iterasi ini, dilakukan kalibrasi, pengujian ulang sensor, serta validasi lokal data sensor untuk memperoleh kinerja sistem yang lebih stabil, presisi, dan sesuai kebutuhan pengguna.

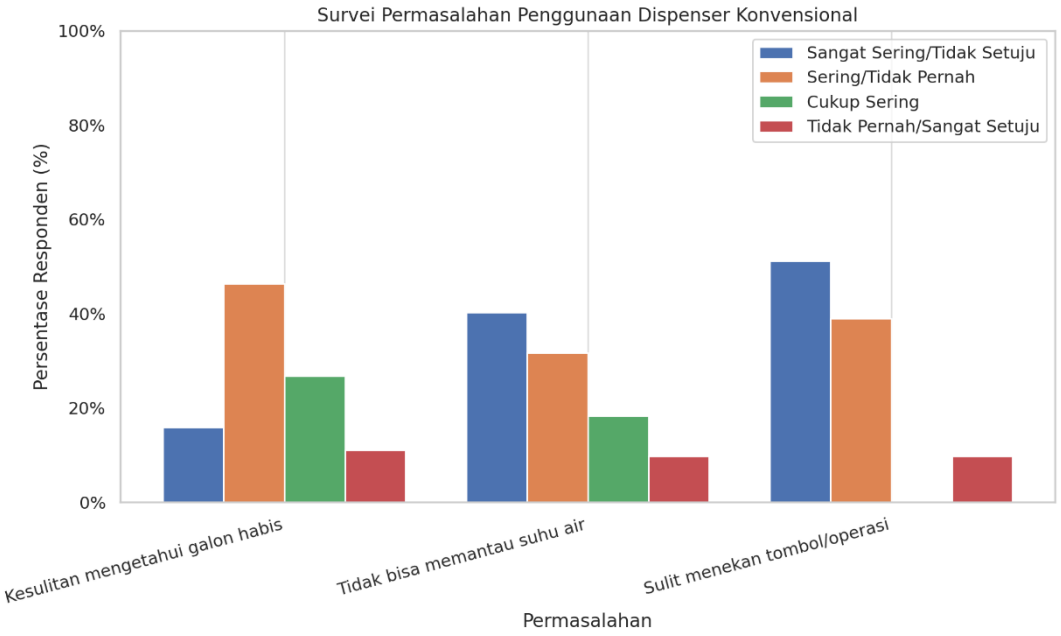
Setiap sub-bab dalam bab ini akan menguraikan secara rinci hasil pengujian pada masing-masing iterasi, disertai data empiris berupa tabel, grafik, dan analisis hasil, baik dari sisi performa sistem maupun pengalaman pengguna. Dengan demikian, diharapkan bab ini mampu memberikan gambaran menyeluruh mengenai capaian dan pengembangan sistem *Smart water dispenser* yang telah dilakukan selama penelitian.

4.1 Iterasi Pertama

4.1.1 Analisis Masalah dan Preferensi Pengguna

Sebelum pengembangan prototipe *Smart water dispenser*, peneliti melakukan survei kepada 80 responden (mahasiswa, dosen, dokter) di Jambi dan Yogyakarta. Survei bertujuan mengidentifikasi masalah utama penggunaan dispenser konvensional dan preferensi fitur dispenser pintar berbasis IoT.

1. Hasil Survei Permasalahan Pengguna



Gambar 4.1 Rekapitulasi Permasalahan Utama Pengguna Dispenser Konvensional Berdasarkan Hasil Survei

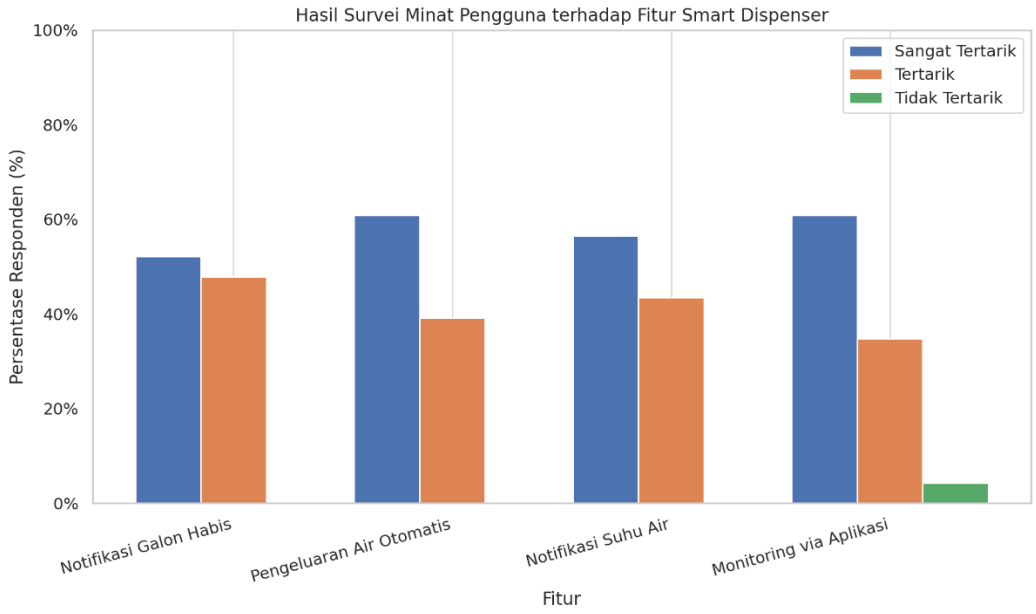
Berdasarkan hasil survei terhadap 82 responden, diketahui bahwa salah satu permasalahan utama yang sering dialami oleh pengguna dispenser konvensional adalah kesulitan mengetahui kapan air galon habis tanpa adanya notifikasi otomatis. Sebanyak 62,2% responden mengaku sering atau sangat sering mengalami hal tersebut, sementara 26,8% cukup sering, dan hanya 11% yang tidak pernah mengalami kesulitan. Hal ini menunjukkan bahwa tidak adanya indikator volume air galon merupakan kendala nyata bagi sebagian besar pengguna.

Selain itu, hanya 9,8% responden yang merasa bahwa dispenser mereka saat ini memungkinkan untuk memantau suhu air yang siap digunakan, seperti air panas atau dingin. Sementara itu, lebih dari 80% responden menyatakan bahwa mereka sangat sering, cukup sering, atau tidak pernah mengetahui suhu air secara pasti, yang mengindikasikan minimnya fitur indikator suhu pada dispenser konvensional.

Permasalahan lain yang juga banyak dirasakan adalah pada aspek kemudahan pengoperasian dispenser, terutama dalam menekan tombol atau keran air. Sebanyak 51,2% responden menyatakan tidak setuju bahwa dispenser yang mereka gunakan mudah dioperasikan, dan hanya 9,8% yang menyatakan sangat setuju. Hal ini mengindikasikan adanya kelemahan dalam desain ergonomis dispenser saat ini, yang dapat menyulitkan pengguna tertentu seperti lansia, anak-anak, atau individu dengan keterbatasan fisik.

Secara keseluruhan, hasil survei ini menunjukkan bahwa mayoritas pengguna masih menghadapi berbagai kendala dalam penggunaan dispenser konvensional. Oleh karena itu, pengembangan fitur seperti notifikasi galon habis otomatis, indikator suhu real-time, serta sistem pengoperasian yang lebih ergonomis dan efisien sangat diperlukan guna menciptakan pengalaman penggunaan yang lebih nyaman dan modern bagi seluruh kalangan.

2. Survei Fitur



Gambar 4.2 Prefensi Fitur Dispenser Pintar yang Diinginkan Pengguna

Berdasarkan hasil survei yang dilakukan terhadap 23 responden, diketahui bahwa sebagian besar responden memiliki minat tinggi terhadap kehadiran fitur-fitur inovatif pada dispenser air berbasis teknologi Internet of Things (IoT). Salah satu fitur yang paling diminati adalah notifikasi galon hampir habis, dengan 52,2% responden menyatakan sangat tertarik dan 47,8% tertarik. Fitur ini dianggap penting karena dapat membantu pengguna segera mengganti galon sebelum air benar-benar habis, sehingga mencegah ketidaknyamanan dalam penggunaan.

Fitur notifikasi suhu air siap digunakan juga mendapat respons positif, dengan 56,5% responden sangat tertarik dan 43,5% tertarik. Hal ini menunjukkan bahwa fitur ini dinilai bermanfaat untuk memastikan kesiapan suhu air panas maupun dingin sesuai kebutuhan, tanpa perlu menunggu atau melakukan pengecekan manual.

Selanjutnya, sebanyak 60,9% responden sangat tertarik dan 39,1% tertarik terhadap fitur pengeluaran air otomatis saat gelas diletakkan di bawah keran, tanpa perlu menekan tombol. Fitur ini dinilai dapat meningkatkan kenyamanan dan efisiensi, terutama bagi pengguna yang sedang membawa barang, memiliki keterbatasan fisik, atau menginginkan penggunaan yang lebih higienis.

Fitur pemantauan dan pengendalian dispenser melalui aplikasi mobile juga menunjukkan antusiasme tinggi, dengan 60,9% responden sangat tertarik, 34,8% tertarik, dan hanya 4,3% yang tidak tertarik. Fitur ini memungkinkan pengguna memantau status galon, suhu air, dan kualitas air secara real-time serta menerima notifikasi dari jarak jauh melalui perangkat seluler.

Secara keseluruhan, hasil survei ini mencerminkan bahwa pengguna mengharapkan kemudahan, efisiensi, dan pengalaman penggunaan yang lebih modern melalui integrasi fitur-fitur cerdas dalam sistem dispenser air berbasis IoT.

4.1.2 *Build* : Pengembangan *Minimum viable product (MVP)*

Berdasarkan hasil analisis kebutuhan dan preferensi pengguna yang telah dijabarkan pada subbab sebelumnya, pengembangan *Minimum viable product (MVP)* dilakukan untuk menghadirkan solusi terhadap permasalahan utama penggunaan dispenser konvensional. Dari sisi keamanan dan kualitas air, sistem Smart Water Dispenser ini didesain agar seluruh komponen sensor maupun modul elektronik tidak bersentuhan langsung dengan air minum pada dispenser. Load cell hanya dipasang di bawah galon, sedangkan sensor suhu dan sensor lain ditempatkan di luar saluran air atau permukaan luar galon. Dengan pendekatan non-invasif ini, tidak terjadi perubahan atau kontaminasi terhadap kualitas air minum—baik dari sisi rasa, warna, bau, maupun kandungan kimianya. Selain itu, pemasangan perangkat elektronik telah mengikuti standar isolasi yang memastikan tidak terjadi kebocoran listrik ke dalam air. Dengan demikian, alat ini aman digunakan serta tidak merubah kualitas air yang dikonsumsi oleh pengguna. MVP ini dirancang sebagai prototipe awal yang telah mengakomodasi fitur-fitur esensial yang paling dibutuhkan oleh pengguna, sebagaimana diidentifikasi melalui survei.

1. Fitur Utama yang Diimplementasikan

   Adapun fitur utama yang diimplementasikan pada tahap MVP ini meliputi:

a. *Notifikasi* galon hampir habis

Sistem mampu memberikan peringatan otomatis kepada pengguna apabila volume air galon berada di bawah ambang batas tertentu, sehingga pengguna dapat segera melakukan penggantian galon sebelum benar-benar habis. Agar perhitungan volume air tetap valid meskipun menggunakan jenis atau merk galon yang berbeda, sistem mewajibkan proses kalibrasi (tare) setiap kali galon kosong diganti. Load cell akan menimbang berat galon kosong sebelum diisi, kemudian berat tersebut disimpan sebagai nilai dasar. Saat galon terisi dan dipasang, load cell mengukur total berat (wadah + air). Sistem kemudian menghitung berat air dengan mengurangkan berat galon kosong dari total, sehingga volume air dapat diketahui secara akurat dengan rumus 1 liter = 1 kg air. Dengan prosedur ini, perhitungan volume tetap akurat untuk berbagai jenis galon, selama proses tare dilakukan setiap pergantian galon.

b. *Notifikasi* suhu air siap digunakan

Sistem memberikan informasi kepada pengguna ketika suhu air panas atau dingin telah mencapai suhu yang diinginkan, sehingga pengguna tidak perlu lagi menunggu tanpa kepastian.

c. Pengeluaran air otomatis

Fitur ini memungkinkan air keluar secara otomatis ketika gelas terdeteksi di area pengisian, sehingga lebih praktis dan ramah bagi pengguna yang membawa banyak barang atau memiliki keterbatasan fisik.

d. Pemantauan dispenser melalui aplikasi *mobile*

Seluruh status dispenser, seperti volume air galon, suhu air panas/dingin, serta status *notifikasi*, dapat dipantau secara *real-time* melalui pengukuran volume air pada sistem Smart Water Dispenser ini dilakukan dengan menggunakan sensor load cell yang dipasang di bawah galon. Load cell berfungsi untuk mendeteksi berat galon beserta isinya secara real-time. Data berat ini kemudian diproses oleh mikrokontroler ESP32 untuk dikonversi menjadi satuan volume air (liter atau mililiter) berdasarkan berat jenis air. Dengan metode ini, sistem mampu memberikan estimasi volume air yang tersisa secara akurat dan otomatis, tanpa perlu pemeriksaan manual. Nilai volume air yang didapatkan selanjutnya dikirim ke aplikasi mobile melalui Firebase Realtime Database, sehingga pengguna dapat memantau status galon dari jarak jauh secara real-time. Pendekatan ini lebih efisien dan responsif dibandingkan metode manual atau hanya menggunakan sensor level air sederhana.

2. Arsitektur dan Komponen Sistem MVP

   Pengembangan MVP dilakukan melalui beberapa tahapan sebagai berikut:

   a. Perancangan Perangkat Keras (Hardware)

      Sistem menggunakan mikrokontroler ESP32 yang terhubung dengan sensor berat (*load cell* HX711) untuk membaca volume galon, sensor suhu (DS18B20) untuk memantau suhu air, sensor water level untuk mendeteksi keberadaan air, serta relay sebagai aktuator untuk pengendalian pompa dan pemanas. Sensor jarak (ultrasonik HC-SR04) dipasang pada area pengeluaran air untuk mendeteksi keberadaan gelas secara otomatis. Dengan adanya sensor ini, sistem hanya akan mengalirkan air jika terdeteksi ada gelas di bawah nozzle, sehingga mencegah tumpahan air dan mengurangi potensi pemborosan. Penggunaan sensor jarak juga mendukung fitur bebas sentuhan, sehingga meningkatkan higienitas dan kenyamanan pengguna, khususnya di lingkungan kampus yang padat aktivitas. Selain itu, fitur ini juga meningkatkan keamanan karena mencegah air panas keluar jika tidak ada wadah. Permasalahan utama yang diangkat pada penelitian ini adalah keterbatasan dispenser konvensional yang belum menyediakan monitoring volume air dan suhu secara real-time, belum adanya notifikasi otomatis, serta belum adanya sistem pengeluaran air otomatis berbasis deteksi gelas yang lebih praktis dan aman. Solusi yang dihadirkan adalah sistem Smart Water Dispenser berbasis IoT yang terintegrasi dengan sensor dan aplikasi mobile, sehingga pengalaman penggunaan menjadi lebih efisien, responsif, dan higienis. *Notifikasi* volume galon juga didukung oleh buzzer sebagai peringatan lokal.

   b. Perancangan Perangkat Lunak (Software)

      Perangkat lunak dikembangkan menggunakan Arduino IDE untuk pemrograman ESP32, dengan integrasi ke *Firebase Realtime Database* sebagai media pertukaran data antara perangkat keras dan aplikasi *mobile*. Logika kontrol suhu menggunakan prinsip histeresis agar pengendalian pemanas/pendingin lebih stabil.

   c. Pengembangan Aplikasi *Mobile*

      Aplikasi dikembangkan menggunakan *React Native*, dengan antarmuka pengguna yang menampilkan status suhu dan volume galon, pengaturan target suhu, serta sistem *notifikasi* push yang memudahkan pemantauan dan kontrol dispenser dari jarak jauh.

3. Sistem MVP

   Gambar 4.3 memperlihatkan dokumentasi fisik sistem Minimum Viable Product

(MVP) Smart Water Dispenser pada iterasi pertama. Pada tahap ini, rangkaian perangkat keras utama — meliputi ESP32 sebagai mikrokontroler, sensor loadcell + HX711, sensor suhu DS18B20, sensor ultrasonik, water level sensor, serta modul relay dan power supply — telah dirakit, meskipun masih dalam bentuk sederhana.

Tata letak komponen belum sepenuhnya rapi, dan wiring masih terbuka sehingga sistem relatif lebih rentan terhadap noise, gangguan elektromagnetik, maupun koneksi yang longgar. Kondisi ini wajar pada tahap awal pengembangan, karena fokus iterasi pertama adalah memastikan seluruh sensor dan aktuator dapat bekerja sesuai fungsinya.

Walaupun dalam bentuk dasar, prototipe ini sudah mampu menjalankan fungsi utama yaitu membaca data sensor (berat air galon, suhu panas/dingin, deteksi gelas), mengendalikan aktuator (pompa/heater), serta mengirimkan data hasil bacaan ke Firebase melalui koneksi Wi-Fi. Dari sisi perangkat lunak, integrasi antara ESP32 dan Firebase juga telah berhasil ditunjukkan dengan update data real-time.

Iterasi pertama ini berperan sebagai dasar kelayakan sistem. Hasilnya digunakan untuk menilai apakah konsep Smart Water Dispenser berbasis IoT dapat diimplementasikan dengan baik sebelum dilakukan penyempurnaan pada iterasi kedua.

Gambar 4.3 Prototipe Iterasi 1 (MVP Smart Water Dispenser)

Pada tahap ini, seluruh fitur utama yang dirancang dalam Minimum Viable Product (MVP) telah diimplementasikan dan diuji untuk memastikan sistem bekerja sesuai kebutuhan. Integrasi dilakukan pada sisi perangkat keras (hardware), perangkat lunak (software), dan komunikasi data berbasis IoT.

Secara fisik, modul ESP32, load cell dengan HX711, sensor suhu DS18B20, serta sensor ultrasonik dan water level dipasang pada dispenser galon standar. Komponen tambahan seperti relay module, power supply SMPS 12V/10A, pompa, heater, dan kompresor diintegrasikan ke dalam rangkaian dispenser tanpa perlu modifikasi besar pada struktur aslinya.

Foto dokumentasi memperlihatkan bahwa komponen ditempatkan secara modular di bagian dalam dispenser: power supply dan relay terpasang rapi di sisi belakang, sensor diposisikan dekat area tangki dan jalur aliran air, sementara ESP32 ditempatkan pada posisi yang mudah diakses untuk pemrograman ulang. Dengan pendekatan ini, sistem dapat bekerja sebagai retrofit yang dapat dipasang pada dispenser komersial yang sudah ada di pasaran.

Prototipe yang dikembangkan mampu menggabungkan semua fitur utama, yaitu monitoring volume galon, pemantauan suhu panas dan dingin, deteksi gelas, serta notifikasi real-time melalui aplikasi mobile. Hasil pengujian menunjukkan bahwa integrasi hardware dan software berjalan stabil, dengan seluruh fitur dapat berfungsi secara simultan sesuai rancangan.

4. Implementasi Perangkat Keras (Hardware)

   Perangkat keras utama terdiri dari:

   a. **Mikrokontroler ESP32** sebagai pusat kendali sistem.
   b. **Sensor berat (*load cell* + HX711)** untuk memantau volume galon.
   c. **Sensor suhu DS18B20** untuk membaca suhu air panas dan dingin.
   d. **Sensor water level** untuk mendeteksi keberadaan air pada galon.
   e. **Relay** untuk mengontrol pemanas dan pendingin air.
   f. **Buzzer** sebagai peringatan lokal saat volume galon hampir habis.

Gambar 4.4 Prototipe Smart Dispenser berbasis IoT

Gambar 4.4 menunjukkan bentuk fisik prototipe *Smart water dispenser* yang telah dikembangkan, lengkap dengan integrasi seluruh sensor dan aktuator pada papan kontrol utama. Setelah prototipe *Smart water dispenser* berhasil dikembangkan, langkah berikutnya adalah mengimplementasikan berbagai fitur utama yang telah dirancang berdasarkan hasil analisis kebutuhan pengguna. Setiap fitur pada sistem ini dirancang secara khusus untuk mengatasi permasalahan yang paling sering ditemui pada penggunaan dispenser konvensional, serta untuk meningkatkan kenyamanan dan efisiensi dalam

penggunaan air minum sehari-hari. Bagian berikut akan menjelaskan secara rinci bagaimana setiap fitur utama pada *Smart water dispenser* diimplementasikan, mulai dari mekanisme *notifikasi* galon hampir habis, *notifikasi* suhu air yang siap digunakan, fitur pengeluaran air otomatis, hingga pemantauan dispenser melalui aplikasi *mobile*. Setiap fitur disajikan secara terpisah untuk memberikan gambaran yang jelas mengenai cara kerja dan manfaat yang dihadirkan oleh sistem ini bagi pengguna.

a. *Notifikasi* Galon Hampir Habis

Fitur *notifikasi* galon hampir habis dirancang untuk membantu pengguna agar tidak lagi mengalami kehabisan air secara mendadak. Sistem secara otomatis memantau berat galon air setiap saat menggunakan sensor berat. Jika berat galon menunjukkan bahwa air di dalamnya tinggal sedikit, sistem akan memberikan peringatan secara langsung melalui suara buzzer dan *notifikasi* pada aplikasi *mobile*. Dengan adanya fitur ini, pengguna dapat segera mengetahui kapan waktunya mengganti galon air, sehingga kebutuhan air minum di rumah atau tempat kerja tetap terpenuhi tanpa gangguan.

Import/Dependencies:

```
#include <HX711.h>
```

Pin Setup:

```
#define DOUT 5
#define SCK 18
```

Kode Implementasi:

```
// === Sensor ===
HX711 scale;


// === Variables ===
float calibration_factor = -1000.0;
```

Fungsi Warning:

```
void buzzerWarning(float berat) {
  unsigned long now = millis();
  if (berat < 2.0 && now - lastBeep15s >= 15000) {
    tone(BUZZER_PIN, 2000, 150);
    delay(200);
```

```
    tone(BUZZER_PIN, 2000, 150);
    lastBeep15s = now;
   }

   if (berat < 0.1 && now - lastBeep30s >= 30000) {
    tone(BUZZER_PIN, 1500, 1000);
    lastBeep30s = now;
   }
  }
```

Kode ini berfungsi untuk memberikan peringatan otomatis kepada pengguna ketika air galon hampir habis. Jika air dalam galon sudah menipis atau hampir kosong, sistem akan mengaktifkan bunyi peringatan (alarm) melalui buzzer. Dengan adanya peringatan ini, pengguna dapat segera mengetahui kondisi galon dan melakukan penggantian sebelum benar-benar habis, sehingga ketersediaan air tetap terjaga dan aktivitas sehari-hari tidak terganggu.

b. *Notifikasi* Suhu Air Siap Digunakan

Fitur *notifikasi* suhu air siap digunakan dirancang untuk memastikan pengguna selalu mengetahui kapan air panas atau dingin pada dispenser sudah mencapai suhu yang diinginkan. Sistem akan memantau suhu air secara otomatis melalui sensor suhu yang terpasang pada jalur air panas dan dingin. Jika suhu air telah sesuai dengan target yang telah diatur pengguna melalui aplikasi *mobile*, maka sistem akan mengirimkan *notifikasi* baik melalui aplikasi maupun indikator pada perangkat. Pengukuran suhu air panas dan dingin pada dispenser dilakukan secara otomatis menggunakan sensor digital DS18B20 yang dipasang di jalur air panas dan air dingin. Sensor ini akan membaca suhu air secara real-time dan mengirimkan data ke mikrokontroler, kemudian diteruskan ke aplikasi mobile melalui Firebase. Nilai suhu yang ditampilkan pada aplikasi mengikuti pembacaan sensor; apabila nilai suhu air panas (misal 25°C) dan air dingin (misal 26°C) masih mendekati suhu ruang, biasanya menandakan proses pemanasan atau pendinginan belum selesai atau sensor belum mendeteksi perubahan signifikan. Untuk memastikan data akurat, sensor perlu dipasang pada titik terdekat dengan sumber air panas/dingin yang akan dikonsumsi. Dengan adanya fitur ini, pengguna tidak perlu lagi menunggu tanpa kepastian atau memeriksa suhu air secara manual, sehingga penggunaan dispenser menjadi lebih efisien dan nyaman.

Import/Dependencies:

```
#include <OneWire.h>
#include <DallasTemperature.h>
```

Pin Setup::

```
#define DS_PIN_COLD 23
#define DS_PIN_HOT 19
#define RELAY_COLD 33
#define RELAY_HOT  32
```

Kode Implementasi:

```
// === Sensor ===
OneWire oneWireCold(DS_PIN_COLD);
OneWire oneWireHot(DS_PIN_HOT);
DallasTemperature sensorCold(&oneWireCold);
DallasTemperature sensorHot(&oneWireHot);

// === Variables ===
float targetCold = 0;
float targetHot = 0;
bool taskColdCompleted = false;
bool taskHotCompleted = false;

// === Histeresis ===
#define HISTERESIS_COLD 2
#define HISTERESIS_HOT 2
```

Fungsi Kontrol Suhu:

```
// LOGIKA PENDINGIN (LOW = ON, HIGH = OFF)
void kontrolPendingin(int suhuDingin) {
  if (suhuDingin == -99 || targetCold <= 0) return;

  if (suhuDingin > targetCold + HISTERESIS_COLD) {
    digitalWrite(RELAY_COLD, LOW); // ON
    taskColdCompleted = false;
    Firebase.RTDB.setBool(&fbdo,      "/dispensers/dispenser_1/task/cold/completed",
false);
  } else if (suhuDingin <= targetCold) {
    digitalWrite(RELAY_COLD, HIGH); // OFF
    if (!taskColdCompleted) {
      taskColdCompleted = true;
      Firebase.RTDB.setBool(&fbdo,    "/dispensers/dispenser_1/task/cold/completed",
true);
    }
```

```
    }
  }

  // LOGIKA PEMANAS (LOW = ON, HIGH = OFF)
  void kontrolPemanas(int suhuPanas) {
    if (suhuPanas == -99 || targetHot <= 0) return;

    if (suhuPanas < targetHot - HISTERESIS_HOT) {
      digitalWrite(RELAY_HOT, LOW); // ON
      taskHotCompleted = false;
      Firebase.RTDB.setBool(&fbdo, "/dispensers/dispenser_1/task/hot/completed",
false);
    } else if (suhuPanas >= targetHot) {
      digitalWrite(RELAY_HOT, HIGH); // OFF
      if (!taskHotCompleted) {
        taskHotCompleted = true;
        Firebase.RTDB.setBool(&fbdo, "/dispensers/dispenser_1/task/hot/completed",
true);
      }
    }
  }
```

Kode ini memungkinkan sistem untuk memantau suhu air panas dan dingin secara terus-menerus, serta menyesuaikan kerja pemanas atau pendingin secara otomatis agar suhu air tetap sesuai dengan target yang diinginkan pengguna. Ketika suhu air sudah mencapai nilai target, sistem akan mengirimkan *notifikasi* kepada pengguna melalui aplikasi *mobile* sehingga mereka dapat langsung menggunakan air dengan suhu yang tepat tanpa harus menunggu atau memeriksa secara manual.

c. Pengeluaran Air Otomatis

Fitur pengeluaran air otomatis dibuat untuk meningkatkan kenyamanan dan kemudahan pengguna dalam mengambil air dari dispenser. Dengan fitur ini, pengguna tidak perlu lagi menekan tombol secara manual. Sistem akan secara otomatis mengeluarkan air ketika gelas atau wadah terdeteksi di area pengisian, sehingga sangat membantu bagi pengguna yang membawa banyak barang atau memiliki keterbatasan fisik. Selain mempermudah proses pengambilan air, fitur ini juga dapat mengurangi risiko kontaminasi karena meminimalisir kontak langsung dengan perangkat.

Pin Setup:

```
#define POMPA_PIN 15
```

```
#define WATER_SENSOR_PIN 14
```

Kode Implementasi:

```
// --- Kontrol Pompa Berdasarkan Water Level ---
if (!airAda) { // Tidak ada air -> Pompa ON
  digitalWrite(POMPA_PIN, LOW);
  pompaAktif = true;
} else {      // Ada air -> Pompa OFF
  digitalWrite(POMPA_PIN, HIGH);
  pompaAktif = false;
}
```

d. Pemantauan Dispenser melalui Aplikasi *Mobile*

Fitur pemantauan dispenser melalui aplikasi *mobile* memberikan kemudahan bagi pengguna untuk mengetahui kondisi dispenser secara *real-time*, di mana saja dan kapan saja. Melalui aplikasi *mobile* yang terintegrasi dengan sistem, pengguna dapat memantau status volume air galon, suhu air panas dan dingin, serta menerima *notifikasi* jika terjadi kondisi tertentu seperti galon hampir habis atau suhu air sudah siap digunakan. Dengan fitur ini, pengguna tidak perlu lagi melakukan pengecekan manual pada dispenser, sehingga pemantauan menjadi lebih praktis, efisien, dan responsif terhadap kebutuhan air minum sehari-hari.

Import/Dependencies:

```
#include <Wifi.h>
#include <Firebase_ESP_Client.h>
```

Konfigurasi:

```
// === Wifi Config ===
#define WIFI_SSID "Haystudio"
#define WIFI_PASSWORD "haystudiosolution"

// === Firebase Config ===
#define DATABASE_URL "https://nira-1af12-default-rtdb.asia-southeast1.Firebasedatabase.app/"
#define DATABASE_SECRET "GUNBkacELPX2kgycLLXbJ7UoH4cBP5iw0aP38t0A"

FirebaseData fbdo;
FirebaseAuth auth;
FirebaseConfig config;
```

Fungsi Update Data:

```
    void updateFirebase(float weight, bool airAda, bool pompa, int suhuDingin, int suhuPanas) {
      if (!Firebase.ready()) return;

      Firebase.RTDB.setFloat(&fbdo, "/dispensers/dispenser_1/capacity/current", weight);
      Firebase.RTDB.setBool(&fbdo, "/dispensers/dispenser_1/airTerdeteksi", airAda);
      Firebase.RTDB.setBool(&fbdo, "/dispensers/dispenser_1/pompaAktif", pompa);

      if (suhuDingin != -99)
        Firebase.RTDB.setInt(&fbdo, "/dispensers/dispenser_1/cold/temperature", suhuDingin);
      if (suhuPanas != -99)
        Firebase.RTDB.setInt(&fbdo, "/dispensers/dispenser_1/hot/temperature", suhuPanas);
      }
```

Kode di atas memungkinkan aplikasi *mobile* terhubung secara langsung dengan database *real-time* (*Firebase*). Setiap perubahan data pada perangkat dispenser—seperti volume air galon, suhu air, maupun status *notifikasi*—akan otomatis diperbarui di aplikasi, sehingga pengguna selalu mendapatkan informasi terkini terkait kondisi dispenser.

e. Fitur Tombol Mode dan Pengisian Otomatis Berdasarkan Deteksi Gelas

Fitur ini bertujuan untuk mengatur mode suhu dispenser (netral, panas, dingin) hanya dengan satu tombol dan menyalakan pompa sesuai mode yang dipilih apabila sistem mendeteksi gelas pada area pengisian. Sistem menggunakan sensor ultrasonik untuk mendeteksi keberadaan gelas dengan jarak < 3 cm. Setiap mode akan diwakili oleh indikator lampu berbeda (putih: netral, merah: panas, biru: dingin), dan pompa akan menyala sesuai logika yang ditentukan.

Import dan Inisialisasi Pin

```
#include <ESP32Servo.h> // Library khusus Servo di ESP32

// Definisi Pin

#define BUTTON_PIN    13

#define LED_MERAH     27

#define LED_BIRU      26

#define LED_PUTIH     25
```

```
#define TRIG_PIN      5
#define ECHO_PIN      18
#define RELAY_IN1     32  // Pompa Dingin
#define RELAY_IN2     33  // Pompa Panas
#define SERVO_PIN     14  // Servo Motor di GPIO 14
```

Setup Awal dan Fungsi Utama

```
// Variabel Global
int step = 1;
bool lastButtonState = HIGH;
unsigned long lastDebounceTime = 0;
const unsigned long debounceDelay = 50;

bool pompaAktif = false;
unsigned long pompaStartTime = 0;
const unsigned long durasiPompa = 5000;  // 5 detik

Servo myServo; // Buat object Servo

// --- State machine untuk Servo ---
bool servoSedangMaju = false;         // Servo sudah maju ke 180
bool servoTimerBalikAktif = false;    // Timer tunggu 1 detik untuk balik
unsigned long servoTimerStart = 0;

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

  // Setup Pin
  pinMode(BUTTON_PIN, INPUT_PULLUP);
  pinMode(LED_MERAH, OUTPUT);
  pinMode(LED_BIRU, OUTPUT);
  pinMode(LED_PUTIH, OUTPUT);
  pinMode(RELAY_IN1, OUTPUT);
  pinMode(RELAY_IN2, OUTPUT);
```

```
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);

  // Setup Servo
  myServo.attach(SERVO_PIN);
  myServo.write(0); // Servo awal di posisi 0 derajat

  tampilkanLED(step);

  // Pastikan pompa OFF saat awal
  digitalWrite(RELAY_IN1, HIGH);
  digitalWrite(RELAY_IN2, HIGH);
}
```

Logika Loop: Deteksi Tombol dan Sensor

```
      void loop() {
        // -- Baca Tombol & Ganti Mode (dengan debounce) --
        bool buttonState = digitalRead(BUTTON_PIN);

        if (!pompaAktif && lastButtonState == HIGH && buttonState == LOW && (millis() -
lastDebounceTime > debounceDelay)) {
          lastDebounceTime = millis();
          step++;
          if (step > 3) step = 1;
          tampilkanLED(step);
          Serial.print("Mode: ");
          Serial.println(step);
        }
        lastButtonState = buttonState;

        // -- Deteksi objek & Aktifkan Pompa + Servo jika tidak sedang aktif --
        if (!pompaAktif && sensorMendeteksi()) {
          aktifkanPompaSesuaiStep();
          gerakkanServoMaju(); // Servo gerak ke 180 derajat
```

```
  pompaAktif = true;
  pompaStartTime = millis();
 }

 // -- Jika pompa aktif, LED berkedip & pompa dimatikan setelah durasi --
 if (pompaAktif) {
  unsigned long elapsed = millis() - pompaStartTime;

  // LED berkedip sesuai step
  if ((elapsed / 250) % 2 == 0) tampilkanLED(step); // ON
  else matikanSemuaLED();                          // OFF

  // Matikan pompa setelah durasi selesai
  if (elapsed >= durasiPompa) {
   digitalWrite(RELAY_IN1, HIGH);
   digitalWrite(RELAY_IN2, HIGH);
   pompaAktif = false;
   tampilkanLED(step); // Kembali ke indikator normal
   Serial.println("Pompa dimatikan (selesai 5 detik)");

   // Start timer untuk servo balik setelah 1 detik
   if (servoSedangMaju) {
    servoTimerStart = millis();
    servoTimerBalikAktif = true;
   }
  }
 }

 // -- Servo balik ke posisi awal setelah 1 detik pompa mati --
 if (servoTimerBalikAktif && (millis() - servoTimerStart >= 1000)) {
  gerakkanServoBalik(); // Servo balik ke 0 derajat
  servoTimerBalikAktif = false;
 }
```

```
}
```

Fungsi Pemilihan Mode dan Indikator LED

```
// --------- Fungsi Pendukung --------- //

void tampilkanLED(int s) {
  digitalWrite(LED_MERAH, s == 1 ? HIGH : LOW);
  digitalWrite(LED_BIRU,  s == 2 ? HIGH : LOW);
  digitalWrite(LED_PUTIH, s == 3 ? HIGH : LOW);
}

void matikanSemuaLED() {
  digitalWrite(LED_MERAH, LOW);
  digitalWrite(LED_BIRU, LOW);
  digitalWrite(LED_PUTIH, LOW);
}

void aktifkanPompaSesuaiStep() {
  if (step == 1) {
    digitalWrite(RELAY_IN1, LOW); // Pompa dingin aktif
    digitalWrite(RELAY_IN2, HIGH);
    Serial.println("Pompa Dingin Aktif");
  } else if (step == 2) {
    digitalWrite(RELAY_IN2, LOW); // Pompa panas aktif
    digitalWrite(RELAY_IN1, HIGH);
    Serial.println("Pompa Panas Aktif");
  } else if (step == 3) {
    // Aktifkan kedua relay secara bertahap untuk cegah power surge
    digitalWrite(RELAY_IN1, LOW);
    delay(100); // jeda kecil, aman
    digitalWrite(RELAY_IN2, LOW);
    Serial.println("Kedua Pompa Aktif");
  }
}
```

```
bool sensorMendeteksi() {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  long durasi = pulseIn(ECHO_PIN, HIGH, 30000); // timeout 30ms
  float jarak = durasi * 0.034 / 2;

  Serial.print("Jarak: ");
  Serial.println(jarak);
  return (jarak > 0 && jarak < 10.0); // <10 cm trigger
}

// Servo ke posisi 180 derajat (maju)
void gerakkanServoMaju() {
  if (!servoSedangMaju) {
    Serial.println("Servo bergerak ke 180 derajat!");
    myServo.write(180);
    servoSedangMaju = true;
  }
}

// Servo balik ke 0 derajat (awal)
void gerakkanServoBalik() {
  if (servoSedangMaju) {
    Serial.println("Servo kembali ke posisi awal (0 derajat).");
    myServo.write(0);
    servoSedangMaju = false;
  }
}
```

5. Implementasi Perangkat Lunak (Aplikasi *Mobile*)

Setelah implementasi perangkat keras pada sistem *Smart water dispenser*, langkah berikutnya adalah mengembangkan perangkat lunak berupa aplikasi *mobile* yang berfungsi sebagai antarmuka utama bagi pengguna. Aplikasi *mobile* ini dirancang untuk memberikan kemudahan dalam melakukan *monitoring*, pengaturan, serta mendapatkan *notifikasi* secara *real-time* dari dispenser yang telah terhubung. Dengan integrasi aplikasi *mobile*, pengguna dapat memantau status dispenser, mengatur suhu air, mengelola ruangan, hingga menerima berbagai *notifikasi* penting langsung dari smartphone mereka. Aplikasi ini dikembangkan menggunakan *React Native* dan terhubung ke *Firebase Realtime Database* sebagai pusat pertukaran data antara perangkat keras dan perangkat lunak. Berikut ini merupakan fitur-fitur utama aplikasi *mobile* yang diimplementasikan pada sistem *Smart water dispenser*:

a. Manajemen Ruangan (Room Management)

Fitur manajemen ruangan memberikan fleksibilitas bagi pengguna untuk mengorganisasi berbagai perangkat dispenser yang dimiliki pada beberapa lokasi berbeda, misalnya rumah, kantor, atau ruang publik lainnya. Pengguna dapat menambahkan ruangan baru, mengganti nama ruangan, atau menghapus ruangan sesuai kebutuhan, sehingga pengelolaan perangkat menjadi lebih terstruktur dan mudah.

Import/Dependencies:

```
// DispenserModal.tsx
import React, { useState } from 'react';
import {
  Modal,
  View,
  Text,
  StyleSheet,
  TouchableOpacity,
  TextInput,
  FlatList,
} from 'react-native';
```

```
import { useDispenser } from '@/context/DispenserContext';
```

Kode Implementasi:

```
Export default function DispenserModal({ visible, onClose }: DeviceModalProps) {
  const {
    savedRooms,
    handleSaveRoom,
    handleDeleteRoom,
    selectedDispenser,
    handleSelect,
    handleRename,
  } = useDispenser();
  const [mode, setMode] = useState<'RENAME' | 'ADD' | 'DEFAULT'>('DEFAULT');
  // ...


  const handleAddRoom = async () => {
    try {
      if (name.trim()) {
        await handleSaveRoom(name.trim());
        setName('');
        setMode('DEFAULT');
      }
    } catch (error) {
      console.log(error);
      if (error instanceof Error) {
        setError(error.message);
      }
    }
  };
```

```
 // ...


 return (
  <Modal>
   {/* ... */}
   <FlatList
    data={savedRooms}
    keyExtractor={(item) => item.id}
    style={styles.list}
    renderItem={({ item }) => (
     <TouchableOpacity
      style={[
       styles.roomItem,
       selectedDispenser?.id === item.id && styles.activeRoom,
      ]}
      onPress={() => {
       handleSelect(item);
       onClose();
      }}
     >
      {/* ... */}
     </TouchableOpacity>
    )}
   />
  </Modal>
 );
}
```

Kode di atas mengatur proses penambahan, penggantian nama, dan penghapusan

ruangan dalam aplikasi *mobile*. Data ruangan disimpan secara lokal sehingga tetap tersimpan meskipun aplikasi ditutup, memudahkan pengguna dalam mengelola banyak dispenser di berbagai tempat

b. Koneksi Dispenser

Fitur koneksi dispenser memungkinkan pengguna untuk menghubungkan aplikasi *mobile* dengan perangkat dispenser pintar yang ada di berbagai ruangan. Pengguna harus terlebih dahulu memilih ruangan, lalu memasukkan kode unik dispenser agar perangkat dapat terhubung dengan aplikasi. Melalui fitur ini, pengguna dapat dengan mudah menambah, memutus, atau mengganti koneksi ke dispenser sesuai kebutuhan, sehingga pengelolaan perangkat menjadi lebih fleksibel dan terorganisir.

Import/Dependencies:

```
// Add Device Card.tsx
import React from 'react';
import { View, Text, StyleSheet, TouchableOpacity } from 'react-native';
import { useDispenser } from '@/context/DispenserContext';
```

Kode Implementasi:

```
Export default function Add Device Card({ onPress }: Props) {
 const { selectedSaved, selectedDispenser, handleDisconnect } = useDispenser();

 const handlePress = async () => {
  if (!selectedSaved) {
   Toast.show({
    type: 'error',
    text1: 'Select Room First',
   });
   return;
  } else if (!selectedDispenser) {
   onPress?.();
  } else {
   await handleDisconnect();
  }
 };

 return (
  <TouchableOpacity
   style={styles.container}
   activeOpacity={0.7}
   onPress={handlePress}
  >
   <View style={styles.iconContainer}>
```

```
            {!selectedSaved ? (
              <BatteryWarningIcon size={24} color={theme.colors.primary} />
            ) : !selectedDispenser ? (
              <Plus size={24} color={theme.colors.primary} />
            ) : (
              <Minus size={24} color={theme.colors.primary} />
            )}
          </View>
          <Text style={styles.text}>
            {!selectedSaved
              ? 'Select Room First'
              : !selectedDispenser
              ? 'Connect Device'
              : `Connected to ${selectedDispenser?.name}`}
          </Text>
        </TouchableOpacity>
      );
    }
```

Kode di atas mengatur proses koneksi aplikasi dengan perangkat dispenser. Pengguna harus memilih ruangan terlebih dahulu sebelum bisa menambah atau menghubungkan dispenser ke aplikasi. Jika perangkat sudah terhubung, pengguna juga dapat memutus koneksi dengan mudah melalui aplikasi. Seluruh proses ini membantu pengguna untuk mengelola perangkat dispenser secara lebih efisien dan sesuai dengan kebutuhan masing-masing ruangan.

c. *Monitoring* Status Suhu dan Visual *Feedback*

Fitur *monitoring* status suhu dan visual *feedback* pada aplikasi *mobile* memberikan pengalaman pemantauan suhu air panas dan dingin secara *real-time* dengan tampilan yang informatif dan interaktif. Pengguna dapat melihat progress bar yang menunjukkan sejauh mana suhu air sudah mendekati atau mencapai target yang diinginkan. Visualisasi ini membantu pengguna memantau kondisi dispenser dengan mudah, tanpa harus menebak atau mengecek secara manual, sehingga pengambilan keputusan seperti menunggu air siap minum bisa dilakukan dengan lebih efisien.

Import/Dependencies:

```
    // app/(tabs)/index.tsx
    import React, { useState, useEffect } from 'react';
    import TemperatureSummaryCard from
'@/components/dashboard/TemperatureSummaryCard';
    import { useDispenser } from '@/context/DispenserContext';
```

Kode Implementasi:

```
Export default function DashboardScreen() {
 // ...

 const progressHot = getPercentage(
  selectedDispenser?.hot.temperature || 0,
  lastHotTask.from,
  lastHotTask.target
 );

 const progressCold = getPercentage(
  selectedDispenser?.cold.temperature || 0,
  lastColdTask.from,
  lastColdTask.target
 );

 // ...

 return (
  <SafeAreaView style={styles.container}>
   {/* ... */}
   <View style={styles.progressBox}>
    {selectedDispenser ? (
     <View style={{ width: '100%', alignItems: 'center', shadowColor: '#000' }}>
      <Text style={styles.title}>Hot Temperature Adjustment</Text>
      <View style={styles.progressBarBackground}>
       <View
        style={[styles.progressBarHot, { width: `${progressHot}%` }]}
       />
      </View>
      <View style={styles.detailsContainer}>
       <Text style={styles.detailsTextLeft}>
        From: {lastHotTask.from}°C
       </Text>
       <Text style={styles.detailsTextRight}>
        Target: {lastHotTask.target}°C
       </Text>
      </View>
     </View>
    ) : (
     <NoDeviceSelected type="Hot" />
    )}
   </View>
   {/* Similar code for cold temperature */}
  </SafeAreaView>
```

```
    );
  }
```

Kode di atas menampilkan progress bar untuk suhu air panas dan dingin pada dashboard aplikasi *mobile*. Setiap progress bar akan memperlihatkan persentase perubahan suhu dari awal hingga target yang diinginkan, sehingga pengguna dapat memantau perkembangan suhu air secara visual dan mengetahui kapan air sudah siap digunakan. Visualisasi ini sangat membantu pengguna dalam mengambil keputusan tanpa harus memeriksa dispenser secara langsung.

d. Integrasi *Firebase Realtime Database*

Integrasi *Firebase Realtime Database* memungkinkan aplikasi *mobile* berkomunikasi secara langsung dan *real-time* dengan perangkat dispenser. Setiap perubahan data—seperti suhu air, volume galon, atau status *notifikasi*—akan otomatis tersimpan dan diperbarui di *cloud*, sehingga pengguna selalu mendapatkan informasi terkini. Fitur ini memastikan seluruh data perangkat dapat diakses, dimonitor, dan dikendalikan dari mana saja melalui aplikasi *mobile*, tanpa perlu koneksi fisik langsung ke perangkat.

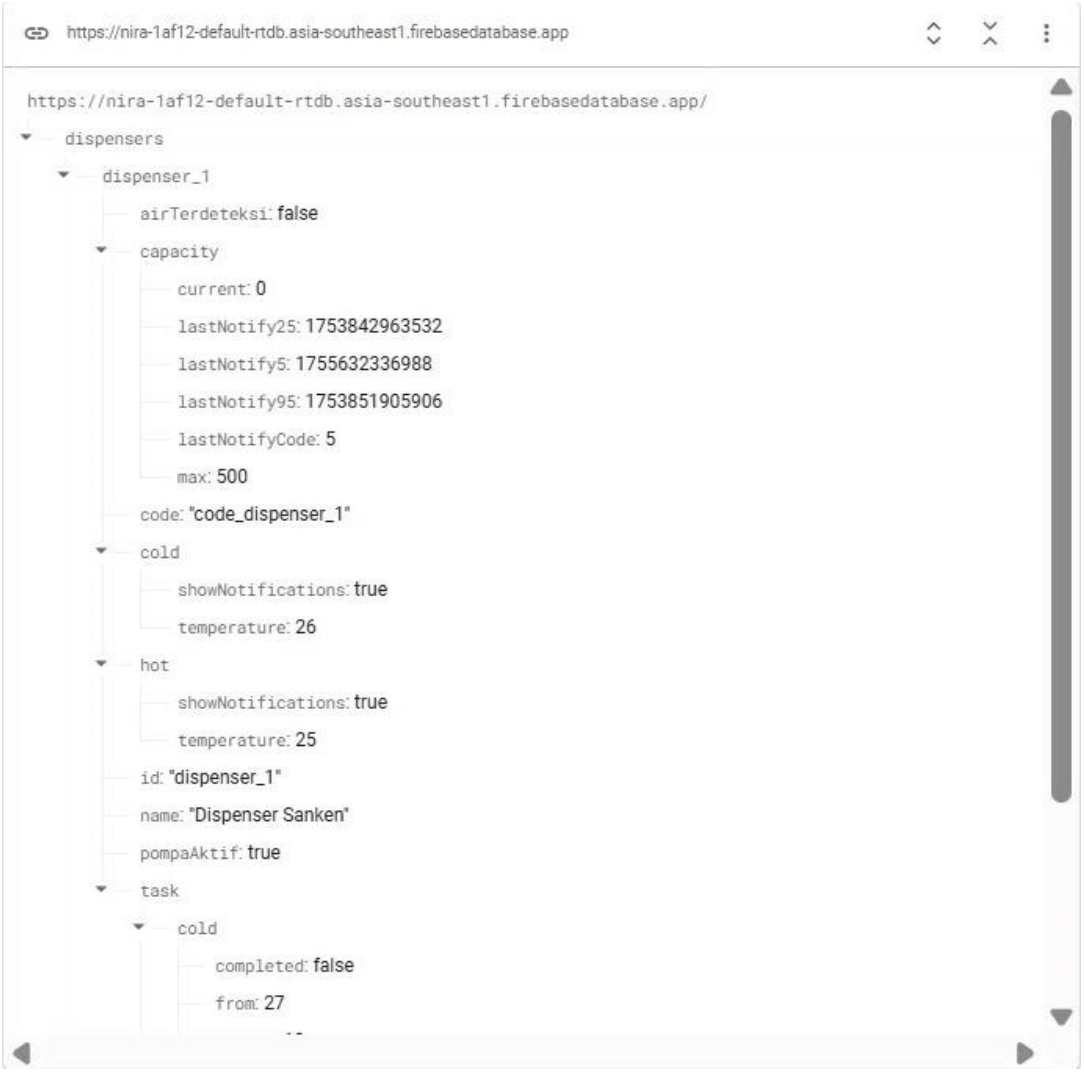
Import/Dependencies:

```
// lib/Firebase.ts
import { FirebaseOptions, initializeApp } from 'Firebase/app';
import { getDatabase } from 'Firebase/database';
```

Kode Implementasi:

```
const FirebaseConfig: FirebaseOptions = {
  apiKey: 'AIzaSyB33BAEU19jn94cgGUNLfNn7lPIQBgf0oQ',
  authDomain: 'nira-1af12.Firebaseapp.com',
  databaseURL:
    'https://nira-1af12-default-rtdb.asia-southeast1.Firebasedatabase.app',
  projectId: 'nira-1af12',
  storageBucket: 'nira-1af12.appspot.com',
  messagingSenderId: '852657862161',
  appId: '1:852657862161:Android:ea500af749cd7acdbae3b0',
  measurementId: '',
};

const app = initializeApp(FirebaseConfig);
const database = getDatabase(app);

Export { database };
```

Kode di atas berfungsi untuk menginisialisasi aplikasi *mobile* dengan konfigurasi proyek *Firebase* yang telah dibuat. Dengan integrasi ini, aplikasi *mobile* dapat membaca dan memperbarui data perangkat dispenser secara *real-time* melalui database *cloud*. Pengguna pun dapat memperoleh informasi dan *notifikasi* secara langsung dari perangkat, tanpa perlu koneksi kabel atau manual.



Gambar 4.5 Tampilan data hasil pembacaan sensor pada Firebase Realtime Database

Namun, pada tahap uji awal, nilai kapasitas galon masih belum stabil; sistem kadang menampilkan angka 0 meskipun galon masih berisi. Demikian pula, sensor suhu pada kondisi awal masih menampilkan nilai default pada kisaran 25–26 °C, belum merepresentasikan kondisi aktual. Hal ini menunjukkan bahwa sistem MVP (iterasi pertama) masih memiliki keterbatasan dalam akurasi data yang ditampilkan. Penyebab utama adalah belum adanya validasi pembacaan di sisi mikrokontroler (ESP32) sebelum data dikirim ke Firebase, serta kalibrasi sensor yang belum optimal.

e. Fitur Mengatur Panas Dingin

Fitur ini memberikan keleluasaan kepada pengguna untuk menentukan sendiri suhu air panas maupun dingin pada dispenser sesuai kebutuhan. Pengguna dapat mengatur nilai target suhu melalui aplikasi *mobile* menggunakan slider yang telah disediakan. Setelah suhu diatur, aplikasi akan memperbarui data target suhu di *Firebase* secara otomatis, dan perangkat dispenser akan menyesuaikan kerja pemanas atau pendingin untuk mencapai suhu yang diinginkan. Dengan adanya fitur ini, pengguna tidak hanya dapat menikmati air minum pada suhu ideal, tetapi juga mengoptimalkan penggunaan energi sesuai preferensi masing-masing.

Import/Dependencies:

```
// TemperatureControlCard.tsx
import React from 'react';
import { View, Text, StyleSheet, Switch } from 'react-native';
import { theme } from '@/constants/Theme';
import { Flame, Snowflake } from 'lucide-react-native';
import TemperatureSlider from './TemperatureSlider';
import { useDispenser } from '@/context/DispenserContext';
```

Kode Implementasi:

```
export default function TemperatureControlCard({
 type,
}: TemperatureControlCardProps) {
 const {
  selectedDispenser,
  lastColdTask,
  lastHotTask,
  handleSwitchNotification,
  handleChangeTemp,
 } = useDispenser();
 const isHot = type === 'HOT';

 const task = isHot ? lastHotTask : lastColdTask;
 // ...

 const changeTemp = (val: number) => {
  handleChangeTemp(val, type);
 };

 // ...

 return (
```

```
    <View style={styles.container}>
      {/* ... */}
      <TemperatureSlider
        value={task.target}
        onValueChange={(val) => changeTemp(val)}
        minValue={isHot ? 28 : 8}
        maxValue={isHot ? 48 : 28}
        type={type}
      />
      {/* ... */}
    </View>
  );
}
```

Kode di atas mengatur fitur pengaturan suhu air pada aplikasi *mobile* dengan tampilan slider. Pengguna cukup menggeser slider untuk memilih suhu air panas atau dingin yang diinginkan. Aplikasi akan mengirimkan nilai tersebut ke perangkat dispenser melalui *Firebase*, dan perangkat akan otomatis menyesuaikan kinerjanya agar suhu air sesuai target pengguna.

f. *Notifikasi* Suhu Air

Fitur *notifikasi* suhu air berfungsi untuk memberitahu pengguna ketika suhu air panas atau dingin sudah mencapai nilai yang diinginkan. Pengguna dapat mengaktifkan fitur ini melalui aplikasi, sehingga mereka tidak perlu lagi mengecek suhu air secara manual. Ketika suhu air sudah sesuai target, aplikasi secara otomatis akan mengirimkan *notifikasi* ke smartphone pengguna, sehingga air dapat langsung digunakan untuk kebutuhan minum atau aktivitas lainnya dengan suhu yang optimal.

Import/Dependencies:

```
// DispenserContext.tsx
import Toast from 'react-native-toast-message';
import * as Notifications from 'Expo-notifications';
```

Kode Implementasi:

```
const showNotificationTemperature = async () => {
  const saved = await fetchSavedDispensers();
  for await (const s of saved) {
    const dispenRef = ref(database, `dispensers/${s.deviceId}`);
    try {
      const snapshot = await get(dispenRef);
      if (snapshot.exists()) {
        const data = snapshot.val() as Dispenser;
```

```
          if (data.hot.showNotifications) {
            const hotTask = data.task.hot;
            if (
              hotTask?.target === data.hot.temperature &&
              !hotTask.completed
            ) {
              sendNotification(
                'Temperature Alert',
                `${data.name} has reached ${data.hot.temperature}°C`
              );
              const taskRef = ref(
                database,
                `dispensers/${s.deviceId}/task/hot`
              );
              update(taskRef, {
                completed: true,
              });
              Toast.show({
                type: 'success',
                text1: `${data.name} has reached ${data.hot.temperature}°C`,
              });
            }
          }
          // Similar code for cold temperature
        }
      } catch (error) {
        console.error(error);
      }
    }
  };
```

Kode di atas secara otomatis akan mengecek apakah suhu air pada dispenser sudah mencapai target yang diatur oleh pengguna. Jika suhu sudah sesuai dan *notifikasi* diaktifkan, sistem akan mengirimkan pesan pemberitahuan ke aplikasi, baik berupa *notifikasi* push maupun pesan singkat di layar. Dengan demikian, pengguna dapat segera mengetahui kapan air sudah siap digunakan tanpa harus memantau suhu secara manual.

g. *Notifikasi* Galon Habis/Volume Galon

Fitur *notifikasi* volume galon mengecek kapasitas air yang tersisa di galon dan memberikan pemberitahuan dengan tingkat kepentingan yang berbeda berdasarkan persentase volume yang tersisa. Sistem memantau tiga ambang batas utama:

1) < 5%: Mengirimkan *notifikasi* "galon kosong" dengan peringatan untuk mengisi ulang (tipe error)
2) < 25% tapi > 5%: Mengirimkan *notifikasi* "galon hampir kosong" (tipe warning)
3) 95%: Mengirimkan *notifikasi* "galon telah diisi penuh"

Untuk menghindari *notifikasi* berulang, sistem menggunakan interval minimal 10 menit antara *notifikasi* volume dan menyimpan kode *notifikasi* terakhir untuk memastikan *notifikasi* yang sama tidak dikirim berulang kali.

Import/Dependencies:

```
// DispenserContext.tsx
import { get, onValue, ref, set, update } from 'Firebase/database';
import Toast from 'react-native-toast-message';
import * as Notifications from 'Expo-notifications';
```

Kode Implementasi:

```
const showNotificationVolume = async () => {
  const saved = await fetchSavedDispensers();
  for await (const s of saved) {
    const dispenRef = ref(database, `dispensers/${s.deviceId}`);
    try {
      const snapshot = await get(dispenRef);
      if (snapshot.exists()) {
        const data = snapshot.val() as Dispenser;
        const {
          lastNotifyCode,
          max,
          current,
          lastNotify25,
          lastNotify5,
          lastNotify95,
        } = data.capacity;
        const percentage =
          Number(Number(Number(current) / Number(max)) * 100) || 0;
        let message = '';
        let code = 0;
        if (percentage > 95) {
          message = `${data.name} fully refilled`;
          code = 95;
        } else if (percentage < 25 && percentage > 5) {
          message = `${data.name} almost empty`;
          code = 25;
```

```
        } else if (percentage < 5) {
          message = `${data.name} is empty, please refill!`;
          code = 5;
        }

        const now = Date.now();
        const INTERVAL = 10 * 60 * 1000;

        if (
          (code === 5 && now - lastNotify5 > INTERVAL) ||
          (code === 25 &&
            now - lastNotify25 > INTERVAL &&
            lastNotifyCode !== 25) ||
          (code === 95 &&
            now - lastNotify95 > INTERVAL &&
            lastNotifyCode !== 95)
        ) {
          const taskRef = ref(database, `dispensers/${s.deviceId}/capacity`);
          update(taskRef, {
            lastNotifyCode: code,
            lastNotify5: code === 5 ? now : lastNotify5,
            lastNotify25: code === 25 ? now : lastNotify25,
            lastNotify95: code === 95 ? now : lastNotify95,
          });
          Toast.show({
            type: code === 5 ? 'error' : 'warning',
            text1: message,
          });
          sendNotification('Capacity Alert', message);
        }
      }
    } catch (error) {
      console.error(error);
    }
  }
};
```

Kode di atas akan memantau volume air pada galon dispenser secara berkala. Jika volume air sudah hampir habis atau benar-benar habis, sistem akan mengirimkan *notifikasi* kepada pengguna melalui aplikasi *mobile*. *Notifikasi* diberikan dengan peringatan berbeda sesuai tingkat kritisnya, sehingga pengguna dapat segera bertindak dan kebutuhan air minum tetap terjaga.

h. *Notifikasi* Galon Siap Digunakan

Fitur *notifikasi* galon siap digunakan terintegrasi dengan sistem *notifikasi* volume, khususnya saat persentase volume mencapai lebih dari 95%. Pada saat ini, sistem akan mengirimkan *notifikasi* "fully refilled" seperti yang terlihat dalam fungsi showNotificationVolume di DispenserContext.Selain itu, komponen WaterLevelCard memberikan representasi visual status galon dengan animasi air dan teks status ("Full Capacity") yang berubah sesuai dengan persentase volume air yang tersisa. Ini membantu pengguna memahami kapan galon telah diisi ulang dan siap digunakan dengan status yang jelas ditampilkan di antarmuka pengguna.Keseluruhan fitur *notifikasi* pada sistem ini dikelola secara terpusat melalui DispenserContext, yang memantau perubahan pada database *Firebase* dan mengirimkan *notifikasi* menggunakan kombinasi *Expo* Notifications untuk *notifikasi* sistem dan Toast untuk *notifikasi* dalam aplikasi.

Import/Dependencies:

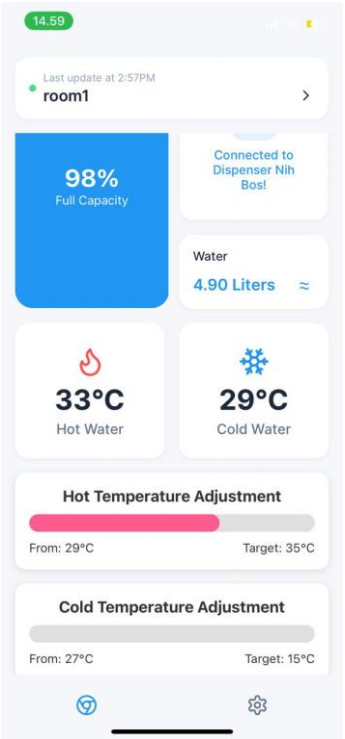
```
// WaterLevelCard.tsx
import React, { useEffect } from 'react';
import { View, Text, StyleSheet, Dimensions } from 'react-native';
import { Dispenser, useDispenser } from '@/context/DispenserContext';
```

Kode Implementasi:

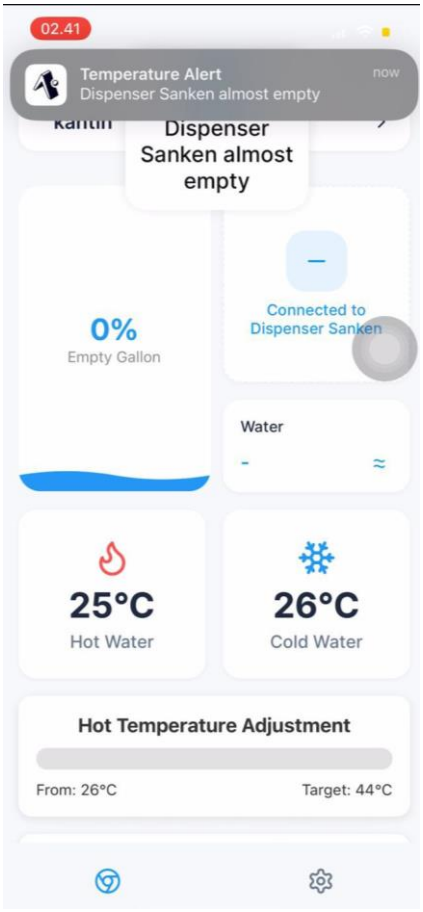
```
Export default function WaterLevelCard() {
 const { selectedDispenser } = useDispenser();
 const percentage = selectedDispenser
  ? Math.round(
     (selectedDispenser.capacity.current / selectedDispenser.capacity.max) *
      100
   )
  : 0;

 // ...

 const status = !selectedDispenser
  ? "There's no smart dispenser connected"
  : percentage <= 0
  ? 'Empty Gallon'
  : percentage >= 75
  ? 'Full Capacity'
  : percentage >= 50
  ? 'Medium Capacity'
```

```
    : percentage >= 25
    ? 'Low Capacity'
    : 'Very Low';

  return (
    <View style={styles.card}>
      <View style={styles.textContainer}>
        <AnimatedText style={[styles.percentageText, textColorStyle]}>
          {`${percentage}%`}
        </AnimatedText>
        <AnimatedText style={[styles.statusText, lowCapacityColorStyle]}>
          {status}
        </AnimatedText>
      </View>
      {/* ... */}
    </View>
  );
}
```

Kode di atas menampilkan persentase volume air galon beserta statusnya pada aplikasi *mobile*. Jika kapasitas galon sudah terisi penuh, aplikasi akan menampilkan status "Full Capacity" dan mengirimkan *notifikasi* ke pengguna bahwa galon sudah siap digunakan. Fitur visualisasi ini memudahkan pengguna untuk memantau kondisi galon tanpa harus memeriksa secara langsung.



Gambar 4.6 Tampilan Dashboard Aplikasi *Mobile Smart water dispenser*



Gambar 4.7 Antarmuka Aplikasi Mobile: Status Galon, Suhu Air, dan Notifikasi

Gambar ini memperlihatkan tampilan dashboard aplikasi. Pada kondisi ini, informasi kapasitas air galon belum tampil sesuai nilai sebenarnya (masih kosong/–), dan indikator suhu hanya menampilkan nilai default tanpa menyesuaikan kondisi sensor. Belum terdapat notifikasi otomatis yang memberikan informasi status terkini.

4.1.3 *Measure*: Pengujian Sistem MVP

Tahap *Measure* bertujuan untuk mengevaluasi secara objektif kinerja dan pengalaman penggunaan sistem *Smart water dispenser* yang telah dikembangkan. Pengujian difokuskan pada dua aspek utama, yaitu akurasi dan kecepatan respons sensor serta aplikasi, serta kenyamanan dan kemudahan penggunaan berdasarkan pengalaman langsung dari pengguna. Hasil pengujian ini menjadi dasar evaluasi dan perbaikan pada iterasi pengembangan berikutnya.

1. Pengujian Sensor Beban HX711 dan *Notifikasi* Galon

   Pengujian sensor *load cell* (HX711) dilakukan untuk mengevaluasi kemampuan sistem dalam memantau berat galon dan mengirimkan *notifikasi* otomatis saat galon hampir habis atau kosong. Uji coba dilakukan dengan beberapa skenario pengisian air pada galon, mulai

dari kondisi penuh, setengah, hampir habis, hingga benar-benar kosong.

Tabel 4.1 Hasil Pembacaan Volume Galon

Berat Galon (kg)	**Status Galon**	***Notifikasi* Terkirim**
18.2	Penuh	Tidak
12.4	Setengah	Tidak
3.2	Hampir Habis	Ya
0.7	Kosong	Ya

Tabel 4.1 menunjukkan bahwa sistem dapat secara akurat membedakan status galon pada setiap level volume air. *Notifikasi* pada aplikasi *mobile* baru dikirimkan saat galon berada pada kondisi "Hampir Habis" (sekitar 3,2 kg) dan "Kosong" (sekitar 0,7 kg), sesuai dengan ambang batas yang ditetapkan pada sistem. Pada kondisi galon masih "Penuh" atau "Setengah", *notifikasi* tidak dikirimkan, sehingga fitur ini efektif mencegah pengguna kehabisan air secara mendadak.

Gambar 4.8 Hasil Pembacaan Sensor dan Status Aktuator melalui Serial Monitor (Iterasi 1)

Gambar ini menampilkan output Serial Monitor pada tahap awal pengujian. Nilai berat galon belum terbaca dengan benar, dan data suhu masih terbaca pada kondisi awal (default). Indikator LED pemanas dan pendingin belum berfungsi sesuai perubahan nilai sensor.

2. Pengujian Sensor Suhu (DS18B20) dan *Notifikasi* Suhu Air

Pengujian sensor suhu DS18B20 dilakukan untuk memastikan bahwa sistem mampu memantau suhu air secara akurat dan memberikan *notifikasi* kepada pengguna saat suhu air sudah siap digunakan. Pengujian dilakukan pada dua jenis air, yaitu air panas dan air dingin, dengan berbagai variasi suhu untuk melihat apakah sistem hanya mengirimkan *notifikasi* pada kondisi suhu yang telah mencapai target.

Tabel 4 .2 *Monitoring* Suhu Air dan *Notifikasi*

Jenis Air	Suhu (°C)	Status Suhu	*Notifikasi* Terkirim
Air Dingin	8.3	Siap Disajikan	Ya
Air Panas	85.7	Siap Disajikan	Ya
Air Dingin	14.1	Belum Siap	Tidak
Air Panas	58.3	Belum Siap	Tidak

Tabel 4.2 menunjukkan bahwa sistem dapat mendeteksi dengan baik apakah suhu air sudah mencapai kondisi "Siap Disajikan". *Notifikasi* hanya dikirimkan ketika suhu air panas atau dingin telah mencapai batas optimal yang diinginkan pengguna. Sebaliknya, pada suhu yang belum siap, sistem tidak mengirimkan *notifikasi*. Hal ini menegaskan bahwa fitur deteksi otomatis dan *notifikasi* suhu bekerja secara akurat dan responsif terhadap kebutuhan pengguna.

3. Pengujian Fungsionalitas Pompa Air dan Waktu Respons Sistem

Pengujian ini bertujuan untuk memastikan bahwa pompa air berfungsi secara otomatis berdasarkan deteksi gelas oleh sensor ultrasonik, serta untuk mengevaluasi seberapa cepat sistem merespons perintah pengisian air. Pada pengujian, gelas ditempatkan pada area pengisian dengan posisi dan jarak yang bervariasi dari sensor. Berbagai jenis gelas digunakan untuk menguji konsistensi deteksi sensor ultrasonik. Pompa air diharapkan aktif secara otomatis setelah gelas terdeteksi dan berhenti setelah volume air yang diinginkan tercapai.

Tabel 4.3 Hasil Pengujian Waktu Respons Pompa dan *Notifikasi*

No	Sensor Aktif	Waktu Pompa Aktif (ms)	Waktu *Notifikasi* (s)
1	Ya	412	1.2
2	Ya	387	1.1
3	Ya	405	1.3

Waktu respons pada pengujian ini diukur berdasarkan data log serial Arduino, yang mencatat selisih waktu antara deteksi sensor hingga perintah eksekusi pompa atau *notifikasi* dikirimkan ke aplikasi *mobile*. Dengan metode ini, diperoleh data yang akurat terkait kecepatan sistem dalam merespons perubahan kondisi pada perangkat. Tabel 4.3 memperlihatkan hasil pengujian waktu respons sistem, baik dalam mengaktifkan pompa air maupun dalam mengirimkan *notifikasi* ke aplikasi *mobile*. Rata-rata waktu yang dibutuhkan sistem untuk mengaktifkan pompa setelah sensor mendeteksi gelas adalah sekitar 400 milidetik. Sementara itu, *notifikasi* berhasil diterima pengguna dalam waktu rata-rata 1,2 detik setelah peristiwa terjadi. Hasil ini menunjukkan bahwa sistem mampu memberikan respons yang cepat dan sesuai kebutuhan penggunaan sehari-hari

4.1.4 Evaluasi dan Umpan Balik Pengguna

Tahap *Learn* merupakan proses analisis dan refleksi terhadap seluruh hasil pengujian yang telah dilakukan pada sistem MVP *Smart water dispenser*. Seluruh proses evaluasi dan penyempurnaan sistem dilakukan secara langsung berdasarkan umpan balik dari pengguna di lingkungan kampus. Keterlibatan user tersebut sangat krusial untuk validasi solusi serta memastikan sistem yang dikembangkan benar-benar menjawab kebutuhan pengguna akhir. Pada tahap ini, peneliti mengidentifikasi kelebihan, kekurangan, serta area yang masih dapat diperbaiki berdasarkan data pengujian dan umpan balik pengguna selama pengujian berlangsung. Hasil analisis ini menjadi dasar untuk melakukan penyempurnaan sistem pada iterasi pengembangan berikutnya.

1. Analisis Umpan Balik Pengguna

   Berdasarkan hasil pengujian sistem MVP, mayoritas pengguna memberikan tanggapan positif terhadap fitur-fitur utama yang telah diimplementasikan, terutama pada *notifikasi* suhu air dan *notifikasi* galon kosong. Kedua fitur ini dinilai sangat membantu, memberikan kemudahan, dan meningkatkan kenyamanan pengguna dalam aktivitas sehari-hari. Pengalaman selama uji coba juga menunjukkan bahwa sistem relatif mudah dipahami dan dioperasikan.

2. Kekurangan dan Masukan Pengguna

   Meskipun secara umum pengguna merasa terbantu dengan fitur-fitur utama yang telah diimplementasikan, terdapat beberapa masukan yang menunjukkan adanya aspek yang perlu ditingkatkan. Beberapa pengguna mencatat bahwa stabilitas sensor dalam mendeteksi kondisi galon atau suhu air kadang kurang konsisten, terutama saat terjadi perubahan kondisi secara cepat. Selain itu, terdapat pula keluhan mengenai kecepatan *notifikasi* yang terkadang mengalami sedikit keterlambatan sebelum muncul di aplikasi *mobile*. Masukan-masukan ini menjadi perhatian penting sebagai dasar evaluasi dan pengembangan sistem pada iterasi berikutnya.

3. Rencana Tindak Lanjut

   Berdasarkan evaluasi dan masukan dari pengguna, berikut adalah langkah strategis yang akan diambil untuk iterasi pengembangan berikutnya:

   a. **Peningkatan Akurasi dan Stabilitas Sistem**

      Dilakukan perbaikan pada aspek yang masih menunjukkan kelemahan dari sisi keandalan dan presisi, khususnya yang berdampak pada kualitas *notifikasi* dan otomatisasi sistem.

b. **Penguatan Mekanisme Umpan Balik**

Disusun sistem pengumpulan masukan pengguna secara lebih terstruktur untuk memastikan bahwa setiap pembaruan sistem didasarkan pada kebutuhan nyata pengguna.

c. **Penyempurnaan Kualitas Pengalaman Pengguna**

Melalui penyederhanaan navigasi, peningkatan kecepatan sistem, dan optimisasi *notifikasi*, diharapkan interaksi pengguna dengan sistem dapat menjadi lebih nyaman dan responsif.

4.2 Iterasi Kedua: Penyempurnaan Sistem

4.2.1 *Build*: Pengembangan Sistem pada Iterasi Kedua

Tahap *Build* pada iterasi kedua merupakan tindak lanjut dari hasil evaluasi dan masukan pengguna pada iterasi pertama. Fokus utama pengembangan kali ini adalah pada penyempurnaan fitur yang telah ada, khususnya pada aspek konektivitas, stabilitas sensor, dan sinkronisasi data. Tujuan dari penyempurnaan ini adalah agar perangkat dapat memberikan pengalaman penggunaan yang lebih mulus, responsif, dan andal, sesuai dengan kebutuhan nyata pengguna.

Tujuan utama dari tahap *Build* pada iterasi kedua meliputi:

1. **Meningkatkan presisi deteksi sensor ultrasonik** terhadap objek (gelas), agar proses pengeluaran air lebih akurat dan andal pada berbagai jenis dan posisi gelas.
2. **Mengoptimalkan stabilitas pembacaan berat galon** menggunakan sensor *load cell*, sehingga sistem dapat mengurangi fluktuasi atau error saat galon dipindahkan atau terjadi getaran.
3. **Menyempurnakan proses validasi sensor secara lokal pada ESP32**, agar sistem lebih tangguh terhadap noise dan perubahan lingkungan, sehingga data yang dikirim ke aplikasi semakin stabil dan tepercaya.

Gambar 4.8 memperlihatkan dokumentasi fisik prototipe pada **iterasi kedua**. Pada tahap ini, sistem telah mengalami penyempurnaan dibandingkan iterasi pertama. Penataan rangkaian dibuat lebih rapi, posisi sensor diperbaiki, dan wiring disusun ulang agar lebih stabil.

Gambar 4.9 Prototipe Iterasi 2 (Penyempurnaan Smart Water Dispenser)”

Perbandingan antara prototipe Iterasi 1 (Gambar 4.3) dan Iterasi 2 (Gambar 4.8) menunjukkan adanya perbedaan signifikan. Iterasi 1 masih berupa rangkaian dasar dengan breadboard dan kabel longgar, sedangkan pada Iterasi 2 sudah dilakukan penataan ulang sehingga lebih rapi dan stabil. Selain itu, beberapa sensor tambahan (loadcell dan water level) telah diintegrasikan.

Perbaikan ini berdampak pada meningkatnya stabilitas sensor, khususnya loadcell dan DS18B20, sehingga pembacaan lebih konsisten. Integrasi dengan Firebase dan aplikasi mobile juga lebih baik, ditandai dengan notifikasi real-time yang terkirim tanpa delay signifikan. Secara keseluruhan, iterasi kedua berhasil mengatasi kelemahan utama pada iterasi pertama, dan menghasilkan prototipe yang lebih layak diuji coba pada pengguna.

Penyempurnaan pada iterasi kedua ini menunjukkan bagaimana siklus *Build–Measure–Learn* diterapkan secara nyata. Masalah utama pada iterasi pertama (instabilitas sensor akibat wiring longgar) diidentifikasi pada tahap *Measure*, kemudian diperbaiki pada tahap *Build* iterasi kedua, dan menghasilkan sistem yang lebih stabil sehingga layak diuji pengguna.

4.2.2 *Measure*: Pengujian Sistem pada Iterasi Kedua

Pada tahap ini, dilakukan proses kalibrasi ulang dan pengujian sensor untuk memastikan data yang dihasilkan lebih akurat dan konsisten dibandingkan iterasi sebelumnya. Fokus utama

perbaikan dilakukan pada dua sensor, yaitu sensor ultrasonik (HC-SR04) untuk deteksi gelas dan sensor *load cell* untuk pembacaan berat galon.

1. Pengujian Konsistensi dan Keandalan Sensor Ultrasonik

   Pengujian ini dilakukan untuk mengevaluasi keandalan dan akurasi sensor ultrasonik dalam mendeteksi keberadaan gelas atau tumblr pada area pengisian air dispenser. Pengujian dilakukan dengan berbagai jenis gelas dan tumblr yang berbeda ukuran, serta dilakukan pada beberapa waktu yang berbeda untuk memastikan hasil deteksi tetap konsisten di berbagai kondisi penggunaan.

   Tabel 4.4 Hasil Pengujian Konsistensi Sensor Ultrasonik pada Deteksi Gelas/Tumblr

Waktu	Total Penggunaan	Deteksi Sukses	Deteksi Gagal	Jenis Gelas/Tumblr	Catatan
11.00	6	6	0	Besar	-
12.32	5	5	0	Kecil	-
12.45	4	3	1	Tumblr	Sensor terkena air, air keluar terus (-263 ml)
14.22	5	5	0	Kecil	-
15.37	5	5	0	Besar	-
15.52	4	4	0	Tumblr	-

   Tabel 4.4 menampilkan hasil pengujian konsistensi sensor ultrasonik dalam mendeteksi berbagai jenis gelas dan tumblr selama beberapa sesi penggunaan. Uji coba ini bertujuan untuk mengetahui tingkat keberhasilan sensor dalam kondisi nyata serta mengidentifikasi potensi kendala yang mungkin terjadi.

   Pengujian pompa galon dilakukan dengan memanfaatkan sensor water level untuk mendeteksi ada atau tidaknya air pada galon. Ketika sensor mendeteksi air masih tersedia, maka pompa dalam kondisi **OFF**. Sebaliknya, ketika sensor tidak mendeteksi air, sistem akan menyalakan pompa secara otomatis untuk melakukan pengisian ulang.

Gambar 4.10 Status Galon dan Pompa pada Pengujian Sensor Water Level

Gambar ini menunjukkan bahwa ketika sensor mendeteksi keberadaan air, status galon

terbaca **"Air TERDETEKSI"** dan pompa galon dalam kondisi **OFF**.

Gambar 4.11 Status Galon "Tidak Ada Objek" dan Aktivasi Pompa Otomatis

Gambar ini menunjukkan bahwa ketika sensor tidak mendeteksi air, status galon berubah menjadi **"Tidak ada objek"** dan pompa galon otomatis dalam kondisi **ON** untuk melakukan pengisian ulang.

2. Pengujian Stabilitas Sensor *Load cell*

Pengujian ini bertujuan untuk menilai peningkatan stabilitas sensor *load cell* dalam membaca berat galon setelah dilakukan kalibrasi dan penyesuaian pada iterasi kedua. Pengujian dilakukan dengan berbagai volume air dan dalam beberapa kondisi, termasuk saat galon dipindahkan atau terkena getaran ringan, untuk memastikan hasil pembacaan tetap konsisten dan minim error.

Tabel 4.5 Hasil *Monitoring* Volume Galon di Kantin Kampus

Waktu	**Volume Aktual (L)**	**Volume Terbaca (L)**	**Error (%)**	***Notifikasi* "Galon Hampir Habis"**
11.00	18.0	17.8	1.11	-
12.32	13.80	13.60	1.45	-
12.45	10.47	10.20	2.58	-
14.22	7.50	7.40	1.33	-
15.37	4.00	3.90	2.50	Y
15.52	1.20	1.10	2.50	Y

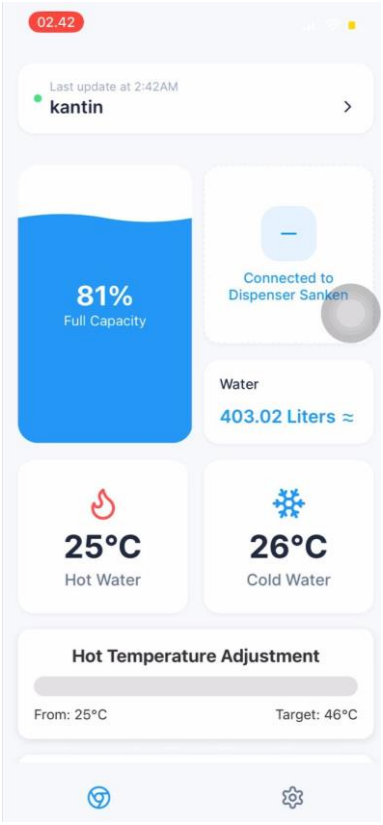
Tabel 4.5 menunjukkan hasil pengujian stabilitas pembacaan sensor *load cell* setelah kalibrasi pada iterasi kedua. Pembacaan volume air galon menunjukkan error yang relatif kecil (rata-rata di bawah 2,5%) bahkan ketika galon mengalami perpindahan atau getaran ringan. *Notifikasi* "Galon Hampir Habis" juga berhasil dikirim otomatis ketika volume air mendekati ambang batas. Pengujian akurasi pengukuran volume air dilakukan dengan membandingkan hasil pembacaan sensor load cell pada sistem dengan volume aktual yang diukur menggunakan gelas ukur sebagai referensi. Berdasarkan hasil pengujian pada beberapa skenario penggunaan, rata-rata tingkat akurasi sensor load cell mencapai di atas

90%, dengan nilai error pengukuran volume air berkisar antara 3% hingga 7%. Error rate tersebut disebabkan oleh beberapa faktor seperti posisi galon, kalibrasi sensor, serta toleransi alat ukur manual. Meskipun demikian, nilai error ini masih tergolong kecil dan tidak mempengaruhi fungsi monitoring sistem secara signifikan, sehingga sistem monitoring volume air dapat diandalkan untuk kebutuhan sehari-hari. Hasil ini menandakan peningkatan signifikan stabilitas sensor dibandingkan iterasi pertama.

Untuk memperkuat hasil pengujian pada Tabel 4.5, dokumentasi pada Gambar 4.12 hingga Gambar 4.14 menunjukkan bukti nyata bahwa sensor *loadcell* dan sensor suhu telah berfungsi sesuai dengan rancangan. Data terbaca melalui Serial Monitor, kemudian dikirim ke Firebase Realtime Database, dan selanjutnya divisualisasikan ke aplikasi mobile. Dokumentasi ini membuktikan bahwa sistem monitoring bekerja secara end-to-end dari perangkat hingga aplikasi pengguna.

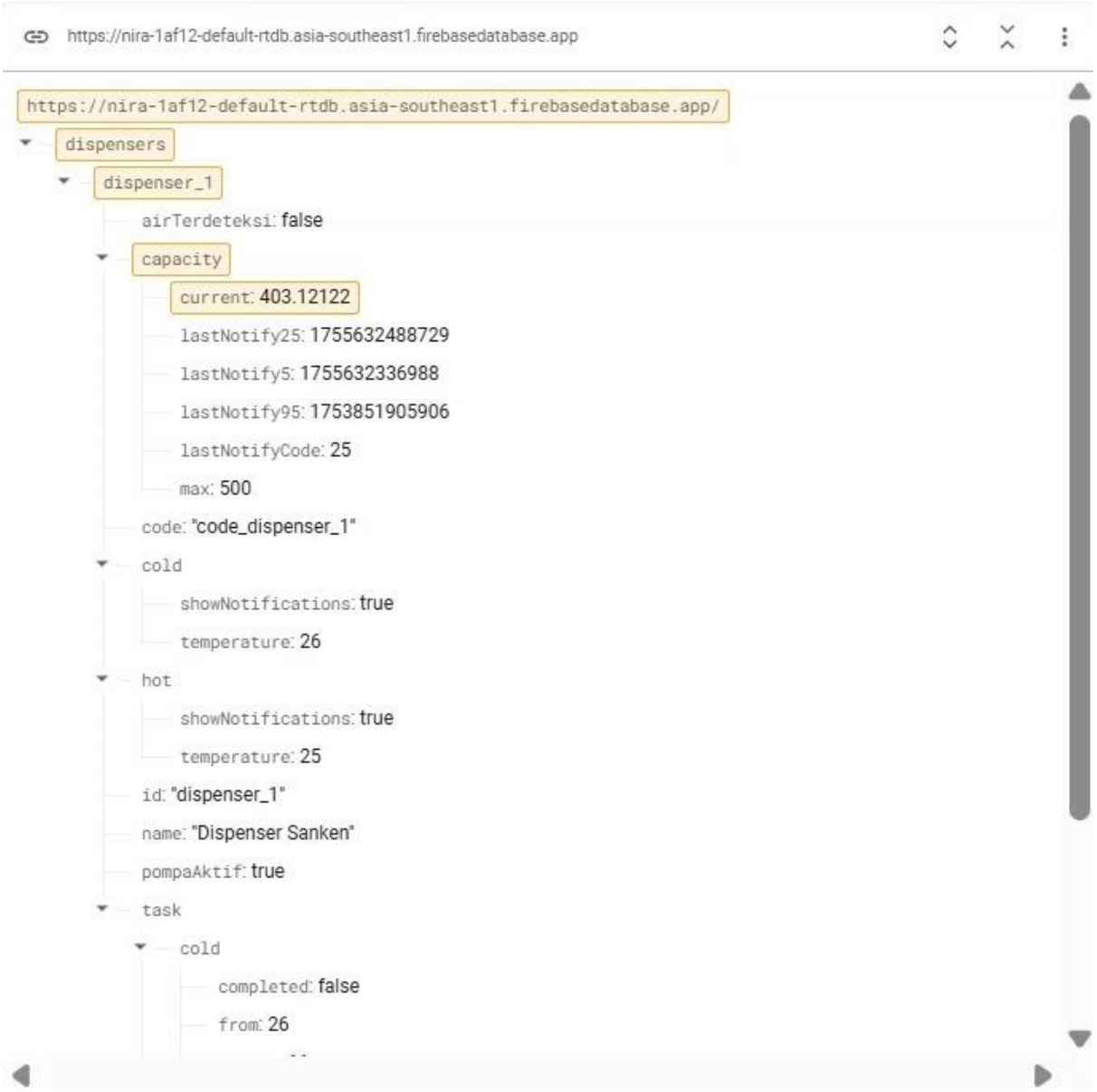
Gambar 4.12 Tampilan Serial Monitor hasil pembacaan sensor *loadcell* dan sensor suhu (nilai volume air 403 gram, suhu dingin 26 °C, dan suhu panas 25 °C)

Tampilan Serial Monitor menunjukkan hasil pembacaan sensor *loadcell* dan sensor suhu yang terhubung dengan mikrokontroler ESP32. Pada pengujian ini, *loadcell* berhasil mendeteksi berat galon sebesar 403 gram. Sensor suhu dingin membaca nilai 26 °C, sedangkan sensor suhu panas membaca 25 °C. Selain itu, tampak pula indikator LED pemanas dalam kondisi menyala menandakan sistem sedang melakukan pemanasan. Hasil ini membuktikan bahwa sensor dapat mengirimkan data ke mikrokontroler secara real-time.



Gambar 4.13 Tampilan aplikasi mobile yang menunjukkan hasil monitoring volume air dan suhu secara real-time, tersinkronisasi dari perangkat IoT.

Tampilan aplikasi mobile memperlihatkan visualisasi data hasil pembacaan sensor yang telah diproses oleh sistem IoT. Volume air terbaca sebesar 403.02 liter (setara 81% kapasitas galon), suhu air panas 25 °C, dan suhu air dingin 26 °C. Data ini otomatis diperbarui secara real-time sehingga pengguna dapat memantau kondisi air pada dispenser langsung melalui smartphone.



Gambar 4.14 Tampilan Firebase Realtime Database yang merekam data hasil pembacaan sensor *loadcell* dan suhu secara real-time (volume 403.12, suhu dingin 26 °C, dan suhu panas 25 °C).

Tampilan Firebase Realtime Database memperlihatkan bahwa data sensor berhasil dikirim dan tersimpan di server cloud. Nilai volume air tercatat sebesar 403.12122, suhu dingin 26 °C, dan suhu panas 25 °C. Informasi ini menjadi bukti bahwa sistem mampu melakukan integrasi end-to-end, mulai dari pembacaan sensor, pengiriman data melalui ESP32, penyimpanan di Firebase, hingga ditampilkan kembali pada aplikasi mobile pengguna.

3. Pengujian Validasi Sensor secara Lokal pada ESP32

Pengujian ini dilakukan untuk memastikan bahwa proses validasi dan pemfilteran data sensor yang dilakukan secara lokal pada ESP32 dapat meningkatkan keandalan sistem

dalam berbagai kondisi lingkungan. Simulasi dilakukan dengan menghadirkan gangguan, seperti noise elektronik dan fluktuasi suhu sekitar, lalu diamati konsistensi data sensor sebelum dan sesudah diterapkan filter perangkat lunak pada ESP32.

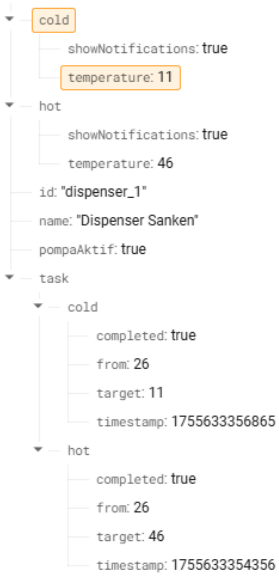
Tabel 4.6 Hasil *Monitoring* Suhu Air Panas dan Dingin di Kantin Kampus, Menunjukkan RataRata Error di Bawah 1°C Pada Berbagai Waktu Penggunaan

Waktu	Suhu Air Panas (°C)	Suhu Terbaca (°C)	Error (°C)	Suhu Air Dingin (°C)	Suhu Terbaca (°C)
11.00	81	80.7	0.3	7.0	8.0
12.32	79	78.7	0.3	10.0	11.4
12.45	78	77.6	0.4	8.0	9.3
14.22	67	66.2	0.8	7.2	7.5
15.37	50	49.8	0.2	9.0	9.4
15.52	73	72.1	0.9	8.1	8.7

Tabel 4.6 memperlihatkan hasil validasi data sensor suhu setelah proses pemfilteran dilakukan secara lokal pada ESP32. Nilai error pada pembacaan suhu air panas dan dingin umumnya berada di bawah 1°C, menandakan bahwa filter perangkat lunak yang diterapkan berhasil meningkatkan akurasi dan kestabilan data sensor, bahkan saat terdapat perubahan lingkungan atau gangguan noise. Perbandingan dengan hasil pada iterasi pertama menunjukkan adanya penurunan error dan peningkatan keandalan sistem dalam pengiriman data suhu ke aplikasi.

a. Suhu dingin

Pengujian dilakukan pada sensor suhu dingin dengan target **11°C**. Data hasil pengujian ditampilkan pada Firebase, Serial Monitor, notifikasi aplikasi, dan dashboard aplikasi.

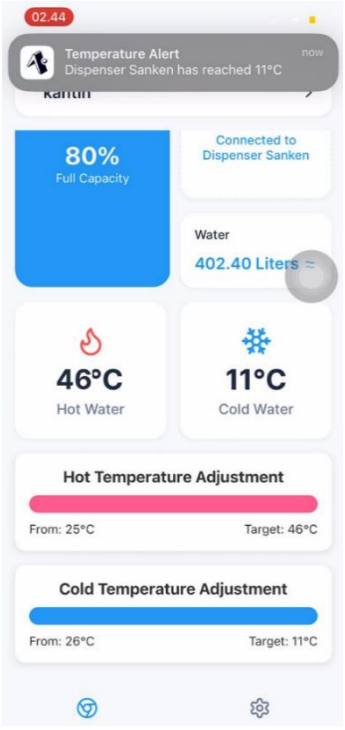


Gambar 4.15 Hasil Pengiriman Data Sensor Suhu Dingin ke Firebase

Gambar ini menunjukkan bahwa Firebase berhasil menerima data dari sensor suhu dingin dengan nilai **11°C**. Hal ini membuktikan bahwa koneksi IoT berjalan baik dan data tersimpan secara real-time di server.

Gambar 4.16 Hasil Uji Sensor Suhu Dingin dan Status Pendingin

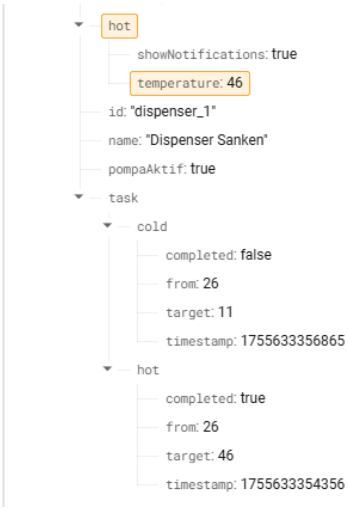
Gambar ini memperlihatkan log di Serial Monitor ketika suhu dingin terbaca **11°C**. Pada saat yang sama, indikator pendingin dalam keadaan **OFF** karena suhu sudah mencapai target.



Gambar 4.17 Real-time Alert Aplikasi Mobile saat Suhu Dingin Mencapai Target 11 °C

Gambar ini menampilkan notifikasi yang dikirimkan aplikasi ketika suhu dingin telah tercapai. Fitur ini menunjukkan fungsi **real-time alert** bekerja sesuai dengan desain sistem.

b. Suhu panas

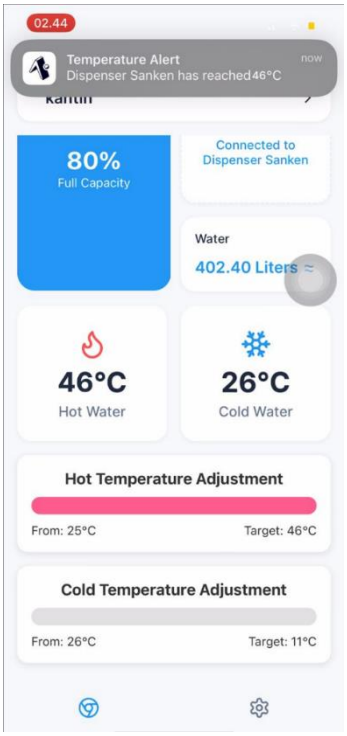


Gambar 4.18 Hasil Pengiriman Data Sensor Suhu Panas ke Firebase

Gambar ini menunjukkan bahwa Firebase berhasil menerima data suhu panas dengan nilai **46°C**. Hal ini membuktikan bahwa sistem mampu menyimpan data sensor secara real-time dan dapat diakses melalui server.

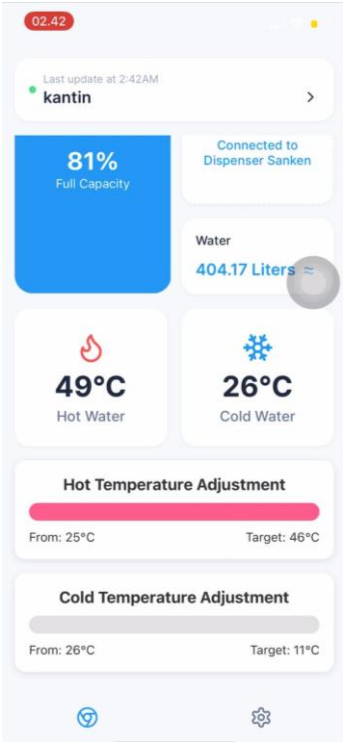
Gambar 4.19 Hasil Uji Sensor Suhu Panas dan Status Pemanas

Gambar ini memperlihatkan log di Serial Monitor ketika suhu panas terbaca **46°C**. Pada kondisi ini, indikator pemanas (relay) berada dalam keadaan **OFF** karena suhu telah mencapai target yang ditentukan.



Gambar 4.20 Notifikasi Aplikasi Suhu Panas Tercapai

Gambar ini menampilkan notifikasi yang diterima pengguna melalui aplikasi. Notifikasi muncul secara otomatis saat suhu panas telah tercapai, menandakan fungsi **real-time alert** bekerja dengan baik.



Gambar 4.21 Real-time Alert Aplikasi Mobile saat Suhu Panas Mencapai Target 46 °C

Gambar ini menunjukkan data suhu panas aktual yang ditampilkan pada aplikasi. Nilai suhu terbaca di rentang **46–51°C**, yang menandakan bahwa pemanas bekerja untuk menaikkan suhu hingga melebihi target sebelum stabil.

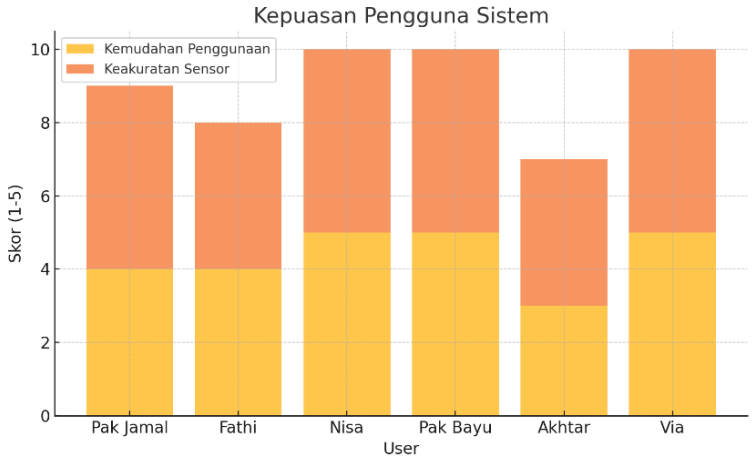
4.2.3 *Learn*: Evaluasi dan Refleksi Iterasi Kedua

1. Evaluasi Hasil Pengujian dan Peningkatan Sistem

   Berdasarkan hasil pengujian pada iterasi kedua, sistem *Smart water dispenser* menunjukkan peningkatan kinerja yang signifikan setelah dilakukan proses kalibrasi dan penyempurnaan pada sensor serta mekanisme validasi data di perangkat ESP32. Akurasi dan stabilitas pembacaan sensor *load cell* dan ultrasonik semakin baik, sedangkan error pembacaan data sensor suhu dapat diminimalisir dengan pemfilteran perangkat lunak. Seluruh fitur utama berjalan lebih andal dan responsif jika dibandingkan dengan hasil pada iterasi pertama.

2. Hasil Wawancara dan Umpan Balik Pengguna

   Selain pengujian teknis, wawancara dan pengumpulan umpan balik langsung dari pengguna juga dilakukan untuk mengetahui persepsi mereka terhadap kemudahan, keakuratan, serta efektivitas fitur yang telah dikembangkan. Berikut rekapitulasi hasil wawancara:



Gambar 4.22 Respon User/Kepuasan Pengguna di Kantin Kampus

3. Refleksi, Masukan Pengguna, dan Tindak Lanjut

   Refleksi dari tahap pengujian menunjukkan bahwa upaya peningkatan presisi sensor ultrasonik dan stabilitas pembacaan *load cell* telah berhasil mengurangi error serta meningkatkan keandalan sistem dalam berbagai kondisi penggunaan. Pengguna juga memberikan tanggapan positif atas peningkatan kecepatan dan konsistensi *notifikasi*, serta

kemudahan pemantauan dispenser melalui aplikasi *mobile*. Namun, terdapat beberapa catatan yang masih perlu menjadi perhatian untuk pengembangan selanjutnya, seperti perlunya mekanisme penanganan error otomatis pada saat terjadi gangguan jaringan atau sensor, serta kemungkinan penambahan fitur keamanan agar sistem semakin robust. Sebagai tindak lanjut, tim pengembang merencanakan:

a. Implementasi mekanisme self-diagnosis pada perangkat untuk mendeteksi dan memperbaiki error secara otomatis.
b. Optimalisasi pengiriman data ke *cloud* agar *notifikasi* tetap stabil meskipun jaringan tidak selalu optimal.
c. Peningkatan fitur keamanan untuk mencegah akses tidak sah dan menjaga privasi data pengguna.

4.2.4 Penegasan Tindak Lanjut untuk Penelitian Selanjutnya

Seluruh catatan perbaikan dan saran pengembangan yang diidentifikasi pada tahap *Learn* iterasi kedua tidak langsung diimplementasikan dalam penelitian ini, melainkan akan dijadikan dasar dan rekomendasi utama untuk penelitian atau pengembangan sistem *Smart water dispenser* di masa mendatang. Dengan demikian, langkah-langkah seperti implementasi mekanisme self-diagnosis, optimalisasi pengiriman data ke *cloud*, dan peningkatan fitur keamanan akan menjadi fokus dalam penelitian lanjutan berikutnya guna menciptakan sistem yang semakin andal, aman, dan sesuai dengan kebutuhan pengguna di berbagai situasi.

1. Tantangan yang Dihadapi Selama Pengembangan dan Pengujian

   Selama proses pengembangan dan pengujian sistem, terdapat beberapa tantangan yang harus dihadapi, di antaranya:

   a. Stabilitas Sensor dala Lingkungan Nyata

      Fluktuasi lingkungan seperti getaran, perubahan suhu ruangan, atau penempatan galon yang tidak stabil sempat menyebabkan pembacaan sensor *load cell* menjadi. Sebagaimana disampaikan oleh salah satu pengguna : *"Kadang volume galon tiba-tiba terbaca habis saat galon dipindahkan."*

   b. Deteksi Objek oleh Sensor Ultrasonik

      Sensor ultrasonik terkadang gagal mendeteksi gelas jika gelas terlalu kecil, terlalu jauh, atau terdapat percikan air. Seperti dikemukakan oleh pengguna lain: *"Deteksi gelas kecil kadang tidak langsung terbaca, terutama kalau permukaan area basah."*

   c. Sinkronisasi Data dan Konektivitas Jaringan

Keterlambatan notifikasi masih kadang terjadi jika koneksi Wifi tidak stabil.

Seorang responden menyebut: "Kadang notifikasi masuknya agak lambat kalau sinyal Wifi jelek."

d. Kalibrasi Sensor dan Umpan Balik Beragam

Proses kalibrasi memerlukan waktu, serta adanya preferensi pengguna yang berbeda-beda dalam penggunaan fitur, sehingga pengembang harus mengambil keputusan kompromi terbaik.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil penelitian dan implementasi yang telah dilakukan pada sistem ***Smart water dispenser*** **berbasis** ***Internet of Things (IoT)*** dengan penerapan metodologi Lean, dapat disimpulkan beberapa hal sebagai berikut:

1. **Pengembangan Prototipe Berbasis Lean**

   Penerapan metodologi Lean melalui siklus *Build-Measure-Learn* terbukti efektif dalam menghasilkan prototipe *Smart water dispenser* yang adaptif terhadap kebutuhan pengguna. Proses iteratif memungkinkan perbaikan berkelanjutan berdasarkan umpan balik nyata dari pengguna, sehingga fitur-fitur utama seperti *monitoring* volume air galon, deteksi suhu air, dan *notifikasi real-time* dapat berfungsi dengan baik.

2. **Fungsionalitas *Monitoring* dan *Notifikasi Real-time***

   Sistem yang dikembangkan mampu memantau volume air galon dan suhu air secara *real-time* menggunakan kombinasi sensor *load cell*, sensor suhu, dan sensor ultrasonik yang dihubungkan ke ESP32 serta dikirimkan ke aplikasi *mobile* melalui *Firebase*. *Notifikasi* otomatis dapat diterima pengguna secara langsung ketika terjadi perubahan kritis, seperti air galon hampir habis atau suhu air tidak optimal.

3. **Peningkatan Pengalaman dan Efisiensi Pengguna**

   Fitur pengeluaran air otomatis berbasis deteksi gelas serta *notifikasi real-time* terbukti meningkatkan kenyamanan, efisiensi, dan aspek higienitas bagi pengguna. Rata-rata waktu respons sistem terhadap perintah pengisian air tercatat di bawah 1 detik, dan tingkat akurasi sensor di atas 90%, yang menunjukkan performa sistem sudah memadai untuk implementasi di lingkungan kampus atau ruang publik.

4. **Validasi User Experience**

   Hasil pengujian dan umpan balik pengguna menunjukkan bahwa penggunaan *Smart water dispenser* ini memberikan pengalaman yang lebih baik dibandingkan dispenser konvensional, khususnya dalam aspek kemudahan, keamanan, dan keterpaduan informasi.

5.2 Saran

Berdasarkan hasil penelitian dan keterbatasan yang ditemukan, Prototipe yang dikembangkan pada penelitian ini secara umum dapat dipasang pada dispenser galon yang dijual di pasaran, sehingga memungkinkan adopsi luas sebagai perangkat tambahan pada

dispenser konvensional. Untuk ke depannya, penelitian lebih lanjut dapat difokuskan pada optimalisasi desain pemasangan agar sistem semakin mudah diintegrasikan pada berbagai tipe dispenser komersial, berikut beberapa saran yang dapat dipertimbangkan untuk pengembangan selanjutnya:

1. **Penyempurnaan Hardware dan Integrasi Sensor**

   Perlu dilakukan optimasi lebih lanjut pada integrasi sensor, khususnya dalam mengatasi kemungkinan error pada pembacaan *load cell* dan sensor suhu akibat faktor lingkungan. Penggunaan sensor dengan akurasi lebih tinggi atau penambahan sistem kalibrasi otomatis dapat meningkatkan keandalan *system*

2. **Peningkatan Skalabilitas Sistem**

   Pengembangan aplikasi *mobile* agar dapat mendukung platform lain, seperti iOS dan web, serta penambahan fitur multi-perangkat dapat memperluas cakupan dan fleksibilitas penggunaan sistem ini di masa depan.

3. **Aspek Keamanan Data**

   Penelitian selanjutnya diharapkan dapat membahas lebih dalam terkait aspek keamanan data dan privasi pengguna, mengingat komunikasi antara perangkat IoT dan aplikasi melibatkan pertukaran data sensitif secara online.

4. **Penerapan pada Skala Lebih Luas**

   Uji coba implementasi di lingkungan publik yang lebih beragam (misalnya ruang tunggu, fasilitas kesehatan, atau area publik lainnya) dapat dilakukan untuk memperoleh data empiris lebih luas, sehingga sistem dapat diadaptasi sesuai kebutuhan yang lebih kompleks.

5. **Pengembangan Fitur Tambahan**

   Fitur-fitur seperti reminder hidrasi berkala, statistik konsumsi air pengguna, serta integrasi pembayaran digital untuk dispenser publik dapat menjadi nilai tambah bagi pengembangan *Smart water dispenser* ke depannya.

DAFTAR PUSTAKA

Burhan, M., Rehman, R. A., Khan, B. A., & Kim, B. (2018). IoT elements, layered architectures and security issues: A comprehensive survey. *Sensors*, 2796.

Chen, Y.-H., Shen, S.-C., Wu, Y.-K., Lee, C.-Y., & Chen, Y.-J. (2022). Design and *testing* of *real-time* sensing *system* used in predicting the leakage of subsea pipeline. *Sensors*, 7290.

Chou, S.-Y., Dewabharata, A., Bayu, Y. C., Cheng, R.-G., & Zulvia, F. E. (2022). An automatic energy-saving strategy for a water dispenser based on user behavior. *Advanced Engineering Informatics*, 101503.

Hutri, H. (2023). *React Native*n ja *Expo*n vertailu (Master's thesis, LUT University). *LUT University Repository*.

Jara Ochoa, H. J., Peña, R., Ledo Mezquita, Y., Gonzalez, E., & Camacho-Leon, S. (2023). Comparative analysis of power consumption between MQTT and HTTP protocols in an IoT platform. *Sensors*, 4896.

Mitu, N. S., Vassilev, V. T., & Tabany, M. (2021). Low cost, easy-to-use, IoT and *cloud*-based *real-time* environment *monitoring system* using ESP8266 microcontroller. *International Journal of IoT and Web Services*, 30-37.

Naik, N. (2017). Choice of effective messaging protocols for IoT *system*s: MQTT, CoAP, AMQP and HTTP. *2017 Systems Engineering Symposium (ISSE)*, 1-7.

Ramdani, D., Prasetyo Adi, P. D., Andriana, Adiprabowo, T., Wahyu, Y., Satyawan, A. S., . . . Rohman, N. (2024). Development of a *smart water dispenser* based on object recognition with Raspberry Pi 4. *International Journal of Advanced Computer Science & Applications*, 501-508.

Sutrisno, S. &. (t.thn.). Implementasi Sistem Otomasi pada Industri Manufaktur di Indonesia. *Jurnal Teknologi Industri*, 29(2), 87-96.

World Health Organization. (2017). Guidelines for Drinking-water Quality (4th ed.). Geneva: WHO Press.

Devin, A., Prasetya, D., & Setyawan, R. (2022). *Smart water dispenser* IoT untuk pemantauan hidrasi pengguna. *Jurnal Teknologi Informasi dan Komunikasi (JTIK)*, *6*(2), 45–55.

Gusrion, A. (2018). Perancangan dan pemrograman sistem kendali berbasis Arduino. *Jurnal Sistem Terapan*, *3*(1), 175–182.

Hendra, Y., Putra, A., & Surya, D. (2017). Mikrokontroler ESP untuk pemrosesan data online menggunakan server. *Jurnal Ilmu Komputer, 5*(1), 1–4.

Institute of Medicine. (2005). Dietary Reference Intakes for Water, Potassium, Sodium, Chloride, and Sulfate. Washington, DC: The National Academies Press.

Kurniawan, R., & Jamaaluddin, M. (2023). Dispenser pintar dengan pembayaran non-tunai dan volume otomatis. *Jurnal Inovasi Teknologi, 10*(3), 98–116.

Nugraha, A., Permana, B., & Sari, I. (2018). Konsep Internet of Things dan penerapannya dalam kehidupan. *Jurnal Informatika dan Sistem Informasi, 4*(1), 7–10.

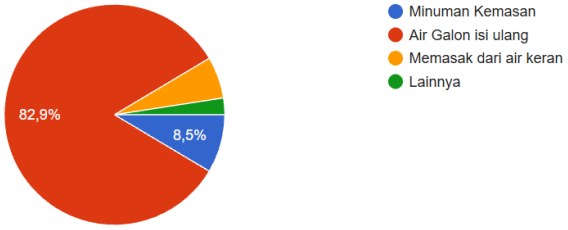
Wahyuningsih, A., Setiawan, D., & Hartono, E. (2019). Alat pengukur debit air menggunakan sensor flow YF-S201. *Jurnal Teknologi Terapan, 5*(1), 35–42.

LAMPIRAN

Bagaimana Anda biasanya mengonsumsi air minum setiap hari?

Salin diagram

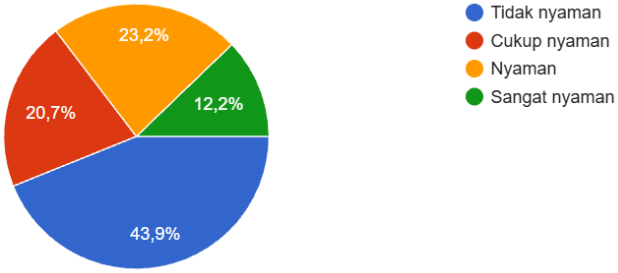
82 jawaban



Apakah Anda merasa nyaman dengan penggunaan dispenser air yang Anda miliki saat ini?

Salin diagram

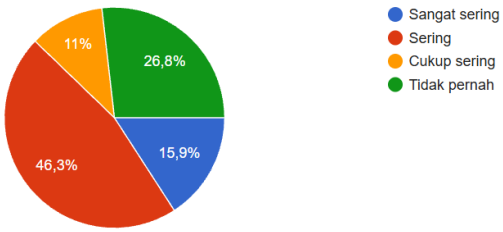
82 jawaban



Apakah anda pernah merasa kesulitan saat harus mengetahui kapan air di galon habis tanpa notifikasi otomatis?

Salin diagram

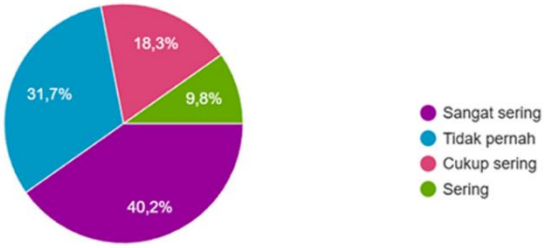
82 jawaban



Apakah dispenser air anda saat ini memungkinkan anda untuk memantau air dengan suhu yang sudah siap digunakan? Minsalnya air dingin atau panas?

Salin diagram

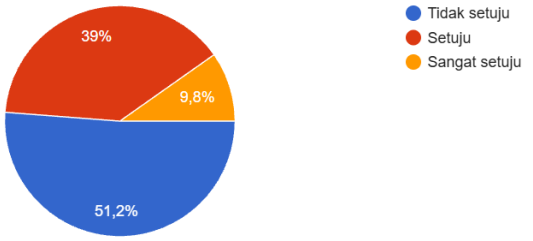
82 jawaban



Apakah dispenser air anda saat ini mudah digunakan tanpa kesulitan dalam menekan tombol atau mengoperasikannya?

Salin diagram

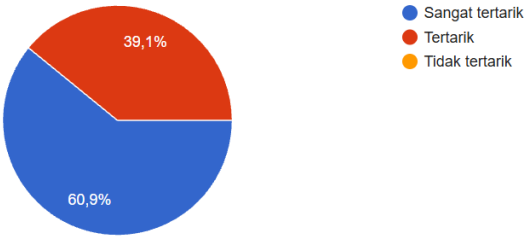
82 jawaban



Apakah Anda tertarik jika dispenser air dapat mengeluarkan air secara otomatis saat gelas diletakkan di bawah keran, tanpa harus menekan tombol?

Salin diagram

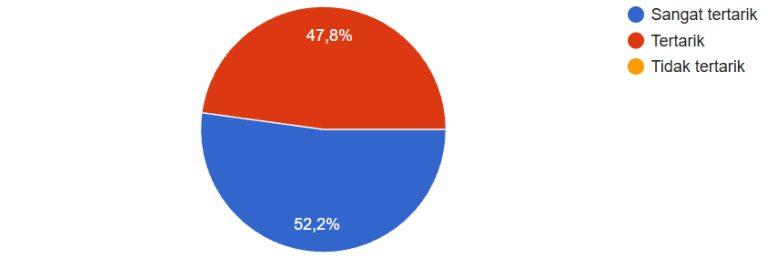
23 jawaban



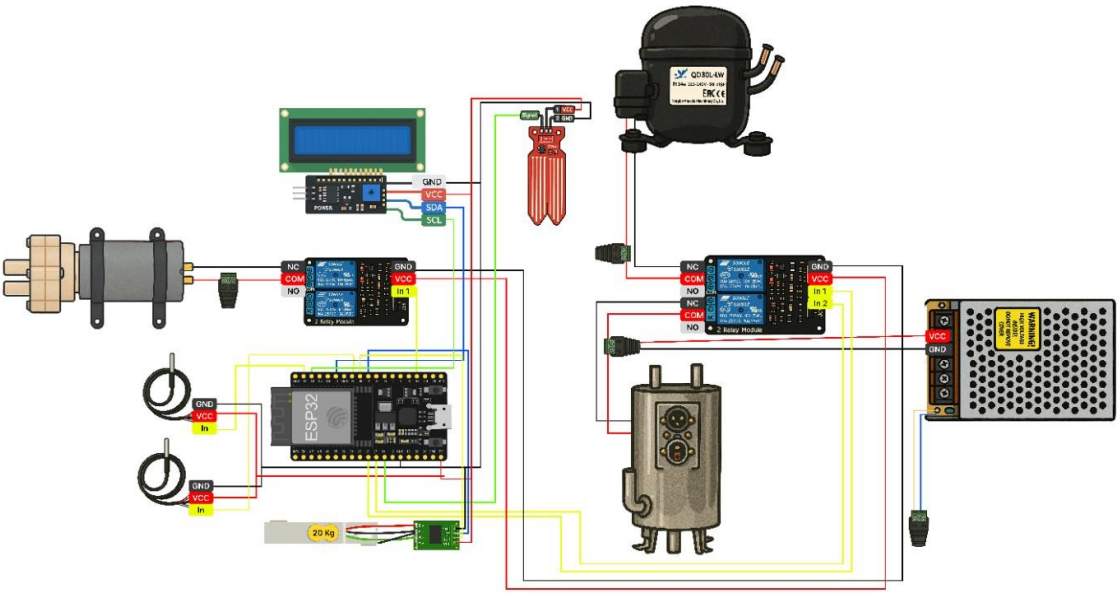
Apakah Anda tertarik jika Anda bisa menerima notifikasi di ponsel ketika galon air hampir habis, sehingga Anda bisa segera menggantinya tanpa harus mengecek manual?

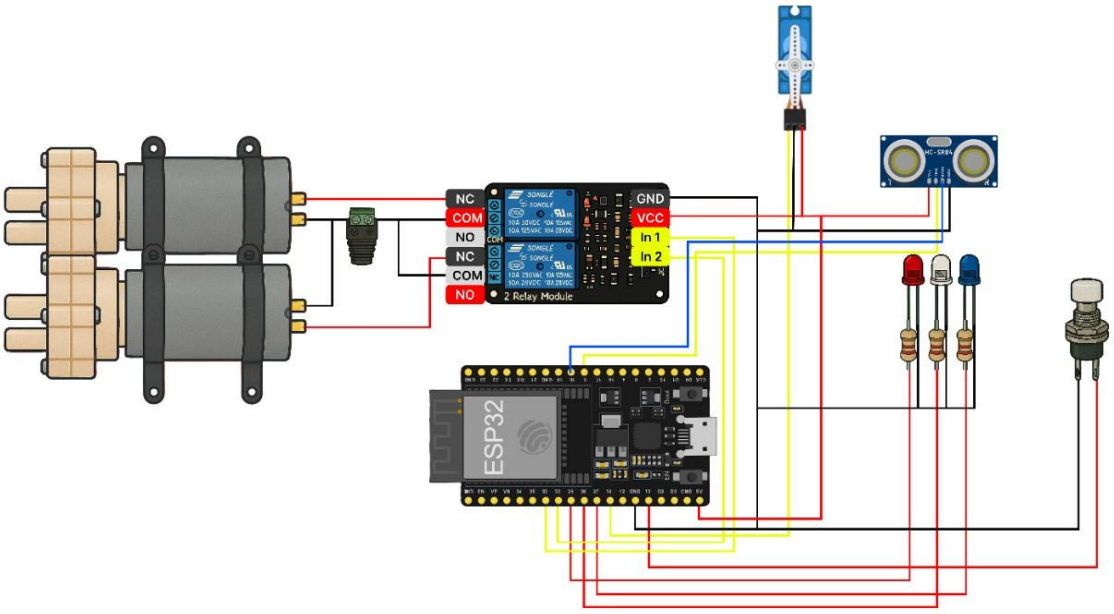
Salin diagram

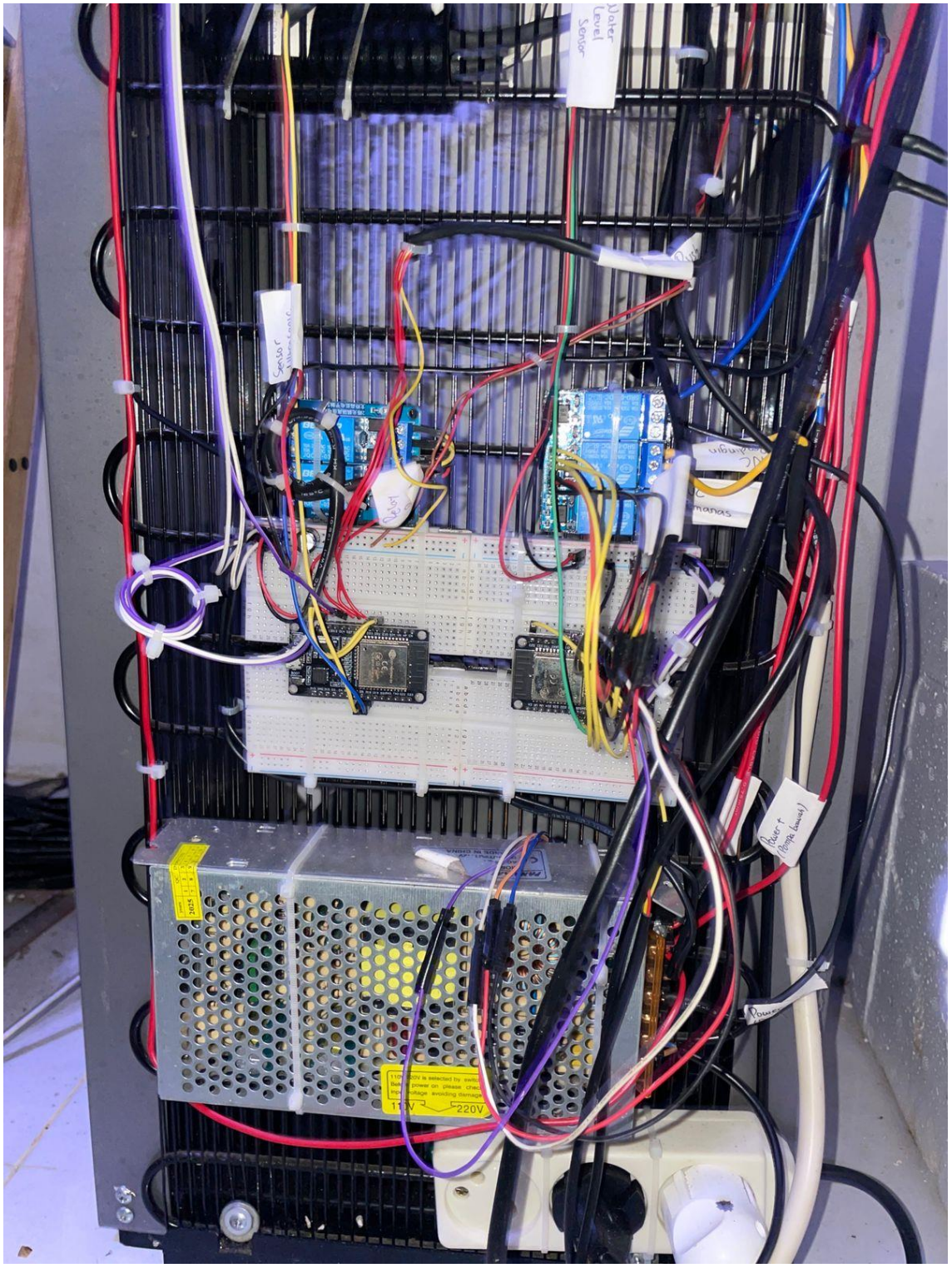
23 jawaban

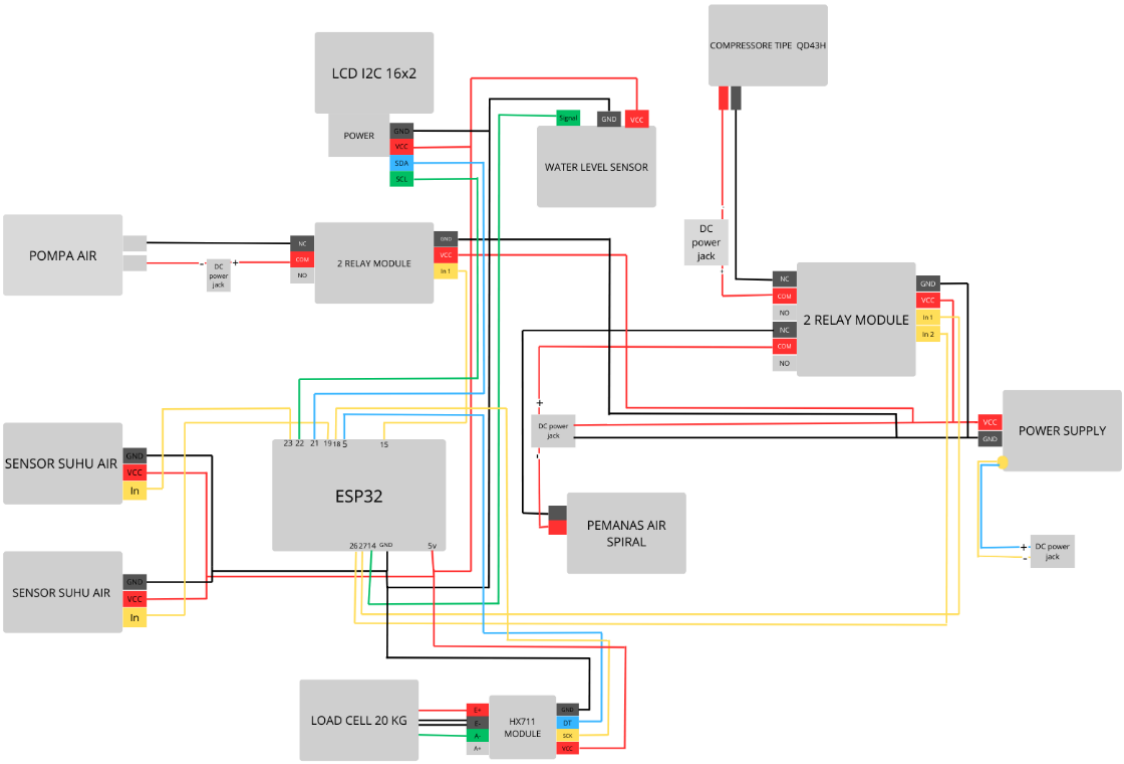


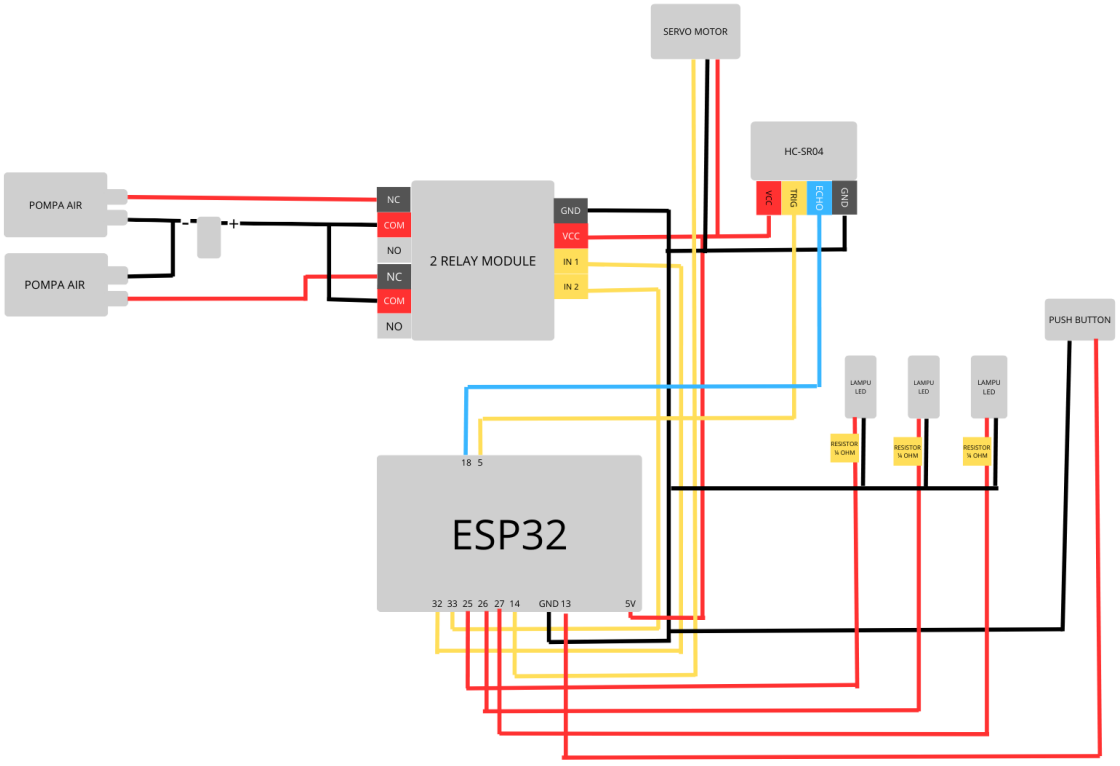

SANKEN


GND
VCC
SDA
SCL
NC
COM
NO
In 1
In 2
ESP32
20 Kg


NC
COM
NO
GND
VCC
In 1
In 2
2 Relay Module
ESP32


Water Level Sensor
Sensor
Power
110V
220V


LCD I2C 16x2
POWER
GND
VCC
SDA
SCL
COMPRESSORE TIPE QD43H
WATER LEVEL SENSOR
Signal
GND
VCC
POMPA AIR
DC power jack
NC
COM
NO
2 RELAY MODULE
GND
VCC
In 1
DC power jack
NC
COM
NO
2 RELAY MODULE
GND
VCC
In 1
In 2
SENSOR SUHU AIR
GND
VCC
In
ESP32
23 22 21 19 18 5 15
26 27 14 GND 5v
DC power jack
PEMANAS AIR SPIRAL
VCC
GND
POWER SUPPLY
DC power jack
SENSOR SUHU AIR
GND
VCC
In
LOAD CELL 20 KG
E-
E+
A-
A+
HX711 MODULE
GND
DT
SCK
VCC


SERVO MOTOR
HC-SR04
VCC
TRIG
ECHO
GND
POMPA AIR
POMPA AIR
NC
COM
NO
NC
COM
NO
2 RELAY MODULE
GND
VCC
IN 1
IN 2
PUSH BUTTON
LAMPU LED
LAMPU LED
LAMPU LED
RESISTOR ¼ OHM
RESISTOR ¼ OHM
RESISTOR ¼ OHM
18 5
ESP32
32 33 25 26 27 14 GND 13 5V