RANCANG BANGUN *SMART DISPENSER* BERBASIS *INTERNET OF THINGS* MENGGUNAKAN *XENDIT* PADA DEPOT AIR MINUM BLUEBUB

TUGAS AKHIR

Disusun Untuk Memenuhi Syarat Kelulusan Program Strata I
pada Sekolah Tinggi Manajemen Informatika dan Komputer
(STMIK) Palangkaraya

OLEH

DANIEL SETIAWAN PURNOMO
C2255201107
PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER
(STMIK) PALANGKARAYA
2026

RANCANG BANGUN *SMART DISPENSER* BERBASIS *INTERNET OF THINGS* MENGGUNAKAN *XENDIT* PADA DEPOT AIR MINUM BLUEBUB

TUGAS AKHIR

Disusun Untuk Memenuhi Syarat Kelulusan Program Strata I
pada Sekolah Tinggi Manajemen Informatika dan Komputer
(STMIK) Palangkaraya

OLEH

DANIEL SETIAWAN PURNOMO
C2255201107
PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER
(STMIK) PALANGKARAYA
2026

PERSETUJUAN

RANCANG BANGUN *SMART DISPENSER* BERBASIS *INTERNET OF THINGS* MENGGUNAKAN *XENDIT* PADA DEPOT AIR MINUM BLUEBUB

Tugas Akhir Ini Telah Disetujui dan Disahkan

Pembimbing I,

Herkules, S.Kom., M.Cs.
NIK. 198510042010106

Pembimbing II,

Elia Zakharia, MT.
NIK. 199205262016104

Mengetahui

Ketua STMIK Palangkaraya,

Ir. Hj. Siti Maryamah, M.M
NIK. 196902021995004

Mengesahkan

Ketua Prodi Teknik Informatika,

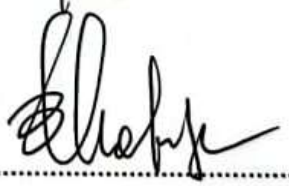
Ir. Abdul Hadi, S.T., M.Kom
NIK. 198505102021101

PENGESAHAN

RANCANG BANGUN *SMART DISPENSER* BERBASIS *INTERNET OF THINGS* MENGGUNAKAN *XENDIT* PADA DEPOT AIR MINUM BLUEBUB

Tugas Akhir ini telah Diujikan, Dinilai, dan Disahkan
Oleh Tim Penguji pada Tanggal 7 Agustus 2025

Tim Penguji Tugas Akhir :

1. Ir. Abdul Hadi, S.T., M.Kom ..... 
   Ketua
2. Dr. Maura Widyaningsih, S.Kom., M.Cs ..... 
   Sekretaris
3. Hotmian Sitohang, M.Kom ..... 
   Anggota
4. Herkules, S.Kom., M.Cs ..... 
   Anggota
5. Elia Zakharia, M.T ..... 
   Anggota

PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama Mahasiswa	:	Daniel Setiawan Purnomo
N I M	:	C2255201107

menyatakan bahwa Tugas Akhir dengan judul :

RANCANG BANGUN *SMART DISPENSER* BERBASIS *INTERNET OF THINGS* MENGGUNAKAN *XENDIT* PADA DEPOT AIR MINUM BLUEBUB

adalah hasil karya saya dan bukan merupakan duplikasi sebagian atau seluruhnya dari karya orang lain, kecuali bagian yang sumber informasi dicantumkan.

Pernyataan ini dibuat dengan sebenar-benarnya secara sadar dan bertanggungjawab dan saya bersedia menerima sanksi pembatalan Tugas Akhir apabila terbukti melakukan duplikasi terhadap Tugas Akhir atau karya ilmiah lain yang sudah ada.

Palangka Raya, 18 Juni 2026

Yang Membuat Pernyataan,



DANIEL SETIAWAN PURNOMO

MOTTO DAN PERSEMBAHAN

Nothing last forever, we can change the future.

Tugas Akhir ini dipersembahkan untuk:

1. Dengan rasa syukur untuk **Tuhan Yesus** yang telah memberi kekuatan dan hikmat dalam menyusun laporan Tugas Akhir ini.
2. Untuk **Kedua Orangtua**, yang telah memberikan dukungan, pengorbanan, kasih, dan doa selama menyusun Tugas Akhir ini.
3. **Para Dosen**, Bapak Herkules, S.Kom., M.Cs dan Bapak Elia Zakharia, MT. Selaku Dosen Pembimbing Tugas Akhir yang selalu memberikan arahan, semangat, kesabaran, dan masukan.
4. Kepada seluruh **Teman-teman**, yang selalu memberikan doa, dukungan, dorongan, dan semangat yang tidak bisa disebutkan satu per satu, yang saya yakini ini akan menjadi langkah menuju kesuksesan dan manfaat bagi banyak orang.
5. Untuk **Andri Permana**, yang telah memberikan izin untuk meneliti di lokasi dan memberikan tempat sehingga penelitian ini dapat diselesaikan.

INTISARI

Daniel Setiawan Purnomo, C2255201107, 2026. *Rancang Bangun Smart Dispenser Berbasis Internet of Things Menggunakan Xendit Pada Depot Air Minum Bluebub*, Pembimbing I Herkules, S.Kom., M.Cs. Pembimbing II Elia Zakharia, M.T.

Depot Air Minum Bluebub saat ini masih menggunakan pencatatan dan penakaran air secara manual. Keterbatasan tersebut memicu antrean fisik pada jam sibuk, waktu tunggu yang kurang efisien, serta rentannya selisih pembukuan transaksi keuangan. Penelitian ini bertujuan untuk merancang dan membangun *prototype* berbasis *Internet of Things* (*IoT*) yang menerapkan konsep bisnis *Online-to-Offline* (O2O) guna mengotomatisasi pemesanan jarak jauh dan transaksi pembayaran nirkontak melalui integrasi *QRIS Xendit*.

Metode pengembangan sistem yang digunakan dalam penelitian ini adalah model *Prototype*. Tahapan penelitian meliputi perancangan perangkat keras menggunakan mikrokontroler *ESP32*, sensor *waterflow* YF-S401, dan sensor *proximity* E18-D80NK, serta pengembangan aplikasi *web* responsif menggunakan *framework CodeIgniter 4* yang dihubungkan melalui sistem *tunnelling Ngrok*. Pengujian eksternal melibatkan dua kelompok responden, yaitu satu orang pemilik depot sebagai *administrator* dan sepuluh orang simulasi pelanggan untuk menguji antarmuka *web*.

Hasil pengujian menunjukkan bahwa sistem berhasil mengintegrasikan pemesanan *web* dengan otomasi mesin secara *real-time*. Hasil uji oleh pemilik depot mencapai persentase 100% (sangat layak), sementara pengujian antarmuka dengan Skala Likert pada pelanggan juga memperoleh kategori sangat layak. Evaluasi mekanis mencatat margin kesalahan penakaran air yang sangat rendah, yaitu antara 0,1% hingga 0,4%, yang dipicu oleh sisa air pada selang dan gelembung udara, namun hal tersebut berhasil diminimalisir melalui kalibrasi pulsa presisi pada perangkat lunak.

Kata Kunci: *CodeIgniter 4, ESP32, Internet of Things, Xendit API*

ABSTRACT

Daniel Setiawan Purnomo, C2255201107, 2026. *Design and Development of an Internet of Things-Based Smart Dispenser Using Xendit At Bluebub Drinking Water Depot*, Advisor I Herkules, S.Kom, M.Cs. Advisor II Elia Zakharia, M.T.

Bluebub Drinking Water Depot currently relies on manual water dispensing and transaction recording. These limitations cause physical queues during peak hours, inefficient waiting times, and potential discrepancies in financial bookkeeping. This research aims to design and build an Internet of Things (IoT)-based Smart Dispenser prototype implementing the Online-to-Offline (O2O) business concept to automate remote ordering and contactless payment transactions through Xendit QRIS integration.

The system development method used in this research is the Prototype model. The research stages include hardware design utilizing an ESP32 microcontroller, YF-S401 water flow sensor, and E18-D80NK proximity sensor, as well as responsive web application development using the CodeIgniter 4 framework connected via the Ngrok tunneling system. External testing involves two groups of respondents: one depot owner acting as the administrator and ten simulated customers testing the web interface.

Test results indicate that the system successfully integrates web ordering with real-time machine automation. Testing by the depot owner achieved a 100% feasibility score, while interface testing using the Likert Scale on customers also resulted in a highly feasible category. Mechanical evaluation recorded a very low water dispensing error margin of 0.1% to 0.4%, triggered by residual water in the hose and air cavitation, which was successfully minimized through precise pulse calibration in the software.

Keywords: *CodeIgniter 4, ESP32, Internet of Things, Xendit API*

PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR

Sebagai sivitas akademik Sekolah Tinggi Manajemen Informatika dan Komputer (STMIK) Palangkaraya, saya yang bertanda tangan dibawah ini:

Nama : Daniel Setiawan Purnomo

NIM : C2255201107

Program Studi : Teknik Informatika

Jenis Karya : Laporan Tugas Akhir

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Sekolah Tinggi Manajemen Informatika dan Komputer (STMIK) Palangkaraya, Hak Bebas Royalti Non Eksklusif (*Non-exclusive Royalty-Free Right*) atas karya ilmiah saya yang berjudul:

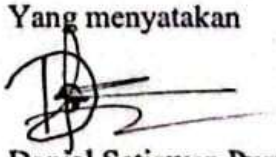
RANCANG BANGUN *SMART DISPENSER* BERBASIS *INTERNET OF THINGS* MENGGUNAKAN *XENDIT* PADA DEPOT AIR MINUM BLUEBUB

Dengan Hak Bebas Royalti Non Eksklusif ini Sekolah Tinggi Manajemen Informatika dan Komputer (STMIK) Palangkaraya berhak menyimpan, mengalih media/formatkan, mengelola dalam bentuk pangkalan data (database), merawat, dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Palangka Raya
Pada tanggal : 18 Juni 2026

Yang menyatakan



Daniel Setiawan Purnomo

KATA PENGANTAR

Puji syukur ke hadirat Tuhan Yang Maha Esa atas berkat, kasih dan penyertaan-Nya sehingga Laporan Tugas Akhir berjudul "Rancang Bangun *Smart Dispenser* Berbasis *Internet of Things* Menggunakan *Xendit* Pada Depot Air Minum Bluebub" ini dapat diselesaikan dengan baik dan benar.

Keberhasilan penyusunan skripsi ini tidak lepas dari bantuan banyak pihak. Oleh karena itu, penulis mengucapkan terima kasih kepada Ibu Ir. Hj. Siti Maryamah, MM., sebagai ketua STMIK yang telah memberikan dukungan akademik; kepada Bapak Ir. Abdul Hadi, S.T., M.Kom., selaku ketua prodi teknik informatika yang telah mendukung proses pembuatan skripsi ini. Ucapan terima kasih khusus penulis haturkan kepada Bapak Herkules, S.Kom., M.Cs. selaku Dosen Pembimbing I dan Bapak Elia Zakharia, M.T. selaku Dosen Pembimbing II dan Dosen Pembimbing Akademik atas segala arahan, kesabaran, dan bimbingannya. Penulis juga berterima kasih kepada Andri Permana selaku pemilik Depot Air Minum Bluebub atas izin dan bantuannya selama penelitian, serta kepada kedua orang tua, keluarga, dan teman-teman angkatan 2022 yang selalu memberikan dukungan dan doa.

Penulis menyadari bahwa skripsi ini masih memiliki kekurangan. Oleh karena itu, kritik dan saran yang membangun sangat penulis harapkan. Semoga purwarupa dan penelitian ini dapat memberikan manfaat bagi pengembangan teknologi ke depannya.

Palangka Raya, 18 Juni 2026

Penulis

DAFTAR ISI

PENGESAHAN iv
PERNYATAAN KEASLIAN v
MOTTO DAN PERSEMBAHAN vi
INTISARI vii
ABSTRACT viii
KATA PENGANTAR x
DAFTAR ISI xi
DAFTAR TABEL xiii
DAFTAR GAMBAR xiv
DAFTAR LAMPIRAN xvi
BAB I PENDAHULUAN 1
1.1 Latar Belakang Masalah 1
1.2 Perumusan Masalah 3
1.3 Batasan Masalah 3
1.4 Tujuan dan Manfaat 4
1.5 Sistematika Penulisan 6
BAB II TINJAUAN PUSTAKA 8
2.1 Kajian Teori 8
2.1.1 *Internet Of Things* (IoT) 8
2.1.2 Mikrokontroler ESP32 9
2.1.3 *Payment Gateway* 9
2.1.4 *Xendit* (*API*) 10
2.1.5 *QRIS* 11
2.1.6 Sensor *Flowmeter* (YF-S401) 13
2.1.7 Sensor *Infrared* (E18-D80NK) 13
2.1.8 *Modul Relay* 15
2.1.9 *Waterpump DC* 12V 16
2.1.10 *Modul LCD* 16x2 I2C 17
2.1.11 Catu Daya (*Adaptor* 12V) 17
2.1.12 *CodeIgniter 4* (CI4) 18
2.1.13 Dispenser 19
2.1.14 *PHP* 19
2.1.15 MySQL 20
2.1.16 *XAMPP* 20
2.1.17 *Webhook* 21
2.1.18 *JSON* 21
2.1.19 *Ngrok* (*Tunneling*) 22

2.1.20 Skala Likert 23
2.2 Penelitian Yang Relevan 24
BAB III METODOLOGI PENELITIAN 30
3.1 Tinjauan Umum 30
3.2 Jenis penelitian 31
3.3 Desain Penelitian 32
3.4 Instrumen Penelitian 34
3.5 Analisis Kebutuhan 35
3.5.1 Analisis Data 36
3.5.2 Analisis Proses 38
3.5.3 Analisis Kelemahan 39
3.6 Desain 41
3.6.1 Desain Proses 42
3.6.2 Desain Perangkat 48
3.6.3 Desain Basis Data 58
BAB IV HASIL DAN PEMBAHASAN 61
4.1 Hasil 61
4.1.1 Implementasi 61
4.1.2 Pengujian 85
4.2 Pembahasan 113
4.2.1 Pembahasan Program dan Analisis Jaringan 113
4.2.2 Analisis Akurasi dan Faktor Penyebab Error Penakaran 116
4.2.3 Pembahasan Basis Data 118
4.2.4 Pembahasan Antarmuka Program (*Interface*) 119
4.2.5 Pembahasan Pengujian Hasil Kusioner (*Usability Test*) 121
4.2.6 Pembahasan Pengujian Oleh Pemilik Depot 123
BAB V KESIMPULAN DAN SARAN 125
5.1 Kesimpulan 125
5.2 Saran 126
DAFTAR PUSTAKA

DAFTAR TABEL

2.1 Penelitian yang relevan 25
3.1 Kebutuhan Perangkat Keras 36
3.2 Kebutuhan Perangkat Lunak 36
3.3 Basis data pengguna 59
3.4 Basis data perangkat 59
3.5 Basis data transaksi 60
4.1 Konfigurasi Rincian Pin dan Pengkabelan (*Wiring*) pada ESP32 66
4.2 Nilai Kalibrasi 94
4.3 Percobaan Penakaran Air 600 mL 94
4.4 Percobaan Penakaran Air 1,5 Liter 96
4.5 Percobaan Penakaran Air 19 Liter 97
4.6 Pengujian Antarmuka Pelanggan (*Customer Side*) 100
4.7 Pengujian Antarmuka Administrator (*Admin Side*) 102
4.8 Integrasi Perangkat Keras (*Hardware Side*) 104
4.9 Aspek Dari Perangkat Lunak 107
4.10 Aspek Dari Perangkat Keras 107
4.11 Aspek Dari Manfaat Bisnis dan Efisiensi Operasional 108
4.12 Tabel Total Hasil Kusioner Dari Responder 110
4.13 Tabel Interval Skala Likert 112

DAFTAR GAMBAR

Gambar 2. 1 *NodeMCU ESP32* 9
Gambar 2.2 *Flowmeter* (YF-S401) 13
Gambar 2.3 *Sensor Infrared (E18-D80NK)* 15
Gambar 2.4 *Modul Relay* 16
Gambar 2.5 *Waterpump DC* 12V 16
Gambar 2.6 *Modul LCD* 16x2 I2C 17
Gambar 2.7 *Adaptor* 12V 2A 18
Gambar 3.1 Alur Penelitian 32
Gambar 3.2 *Signature System* 41
Gambar 3.3 *Flowchart* Sistem *Smart* Dispenser 43
Gambar 3.4 *Usecase* Diagram Sistem *Smart* Dispenser 44
Gambar 3.5 *Activity* Diagram Proses Pemesanan 46
Gambar 3.6 Desain Perangkat 48
Gambar 3.7 Halaman *Login* 51
Gambar 3.8 Pemesanan Air 52
Gambar 3.9 Pembayaran *Scan QRIS* 53
Gambar 3.10 Halaman Menunggu Pengisian 54
Gambar 3.11 Halaman Monitoring *Admin* 55
Gambar 3.12 Halaman Transaksi 56
Gambar 3.13 Desain *Database* Sistem 58
Gambar 4.1 Kabel VIN dan GND pada ESP32 ke *Breadboard* 63
Gambar 4.2 Koneksi Kabel dari Sensor *Infrared* ke *ESP32* 63
Gambar 4.3 Koneksi Kabel *Water Flow Sensor* Ke *ESP32* 64
Gambar 4.4 Koneksi Kabel *Relay* Ke *ESP32* 65
Gambar 4.5 Koneksi Kabel *Waterpump* DC 12V Ke *Relay* 65
Gambar 4.6 Koneksi Kabel *LCD* Ke *ESP32* 66
Gambar 4.7 Rangkaian Koneksi Kabel VCC dan GND Ke *Breadboard* 68
Gambar 4.8 Aktivasi *Apache* dan *MySQL* pada *XAMPP Control Panel* 69
Gambar 4.9 Konfigurasi *Tunnelling* Server Menggunakan *CMD Ngrok* 70
Gambar 4. 10 Tampilan *Dashboard Xendit* (*Testing Mode*) 71
Gambar 4.11 Pembuatan *Secret API Keys* 71
Gambar 4.12 Memasukan *Link Rute API Ngrok* ke *Webhook Xendit* 72
Gambar 4. 13 Halaman Register Pengguna 73
Gambar 4.14 Halaman *Login* Pengguna 73
Gambar 4.15 Halaman Pemesanan Air 74
Gambar 4.16 Tampilan *Barcode QRIS* 75
Gambar 4.17 Halaman Monitoring Pengguna Menunggu Wadah 76
Gambar 4.18 Halaman Monitoring Pengisian Air Sisi Pengguna 76
Gambar 4.19 Halaman Monitoring Pengguna Ketika Pengisian Selesai 77
Gambar 4. 20 Halaman *Dashboard Admin* 78
Gambar 4.21 Kode Program Pembuatan *Invoice* Pembayaran 79
Gambar 4.22 Kode Program Mengirim Notifikasi dari Server ke *Website* 80

Gambar 4.23 Kode Program Mengirimkan Data Format *JSON* 81
Gambar 4.24 Kode Program Permintaan Data ke Server dengan *HTTP GET* 82
Gambar 4.25 Kode Program Mendeteksi Keberaaan Wadah atau Galon 82
Gambar 4.26 Kode program menghitung jumlah pulse 83
Gambar 4.27 Kode Program Menyalakan atau Mematikan *Relay* 84
Gambar 4.28 Kode Program Memperbarui Status Transaksi di *Database* 84
Gambar 4.29 Kode Program Untuk Memperbarui Status Pesanan Selesai 85
Gambar 4.30 Halaman pemesanan 86
Gambar 4.31 Halaman *QRIS Xendit* 87
Gambar 4.32 Halaman Riwayat Pemesanan Status *Paid* 88
Gambar 4.33 Tampilan *JSON* Cek Status *ESP32* 88
Gambar 4.34 Modul *LCD* Menunggu Wadah 89
Gambar 4.35 Modul *LCD* Menampilkan Proses Pengisian Air 89
Gambar 4.36 Sisi Pelanggan Saat Air Sedang Proses Pengisian 90
Gambar 4.37 Tampilan Modul *LCD* Pengisian Sukses 90
Gambar 4.38 Sisi Pelanggan Ketika Proses Pengisian Selesai 91
Gambar 4.39 Status Mesin Pada *Dashboard Admin* 91
Gambar 4.40 *Sensor Proximity* Mendeteksi Keberadaan Wadah 92
Gambar 4.41 Status Serial Monitor Ketika Wadah Terdeteksi 92
Gambar 4.42 *Sensor Proximity* Tidak Mendeteksi Keberadaan Wadah 93
Gambar 4.43 Status Serial Monitor Ketika Tidak Ada Wadah Terdeteksi 93
Gambar 4.44 Hasil Pengujian Kalibrasi 600 mL 95
Gambar 4.45 Hasil Pengujian Kalibrasi 1500 mL 97
Gambar 4.46 Hasil Pengujian Kalibrasi 1500 mL 98
Gambar 4.47 Pengujian *Black Box Testing* 106
Gambar 4.48 Pengujian *Prototype* di Lokasi Depot 109
Gambar 4.49 Google Form Pertanyaan untuk *Usability* dan *User Experience* 112
Gambar 4.50 *Traffic* Data dan Catatan Waktu Respon *Ngrok* 116
Gambar 4.51 Rekaman Data Transaksi yang Berhasil Diperbarui 119
Gambar 4.52 Ringkasan Kontrol dan *Traffic* Data pada *Dashboard Admin* 121
Gambar 4.53 Kusioner *Usability* Pada *Google Form Spreadsheets* 123

DAFTAR LAMPIRAN

Lampiran 1. Surat tugas pembimbing tugas akhir
Lampiran 2. Lembar konsultasi bimbingan Tugas Akhir
Lampiran 3. Surat izin penelitian
Lampiran 4. Lembar observasi
Lampiran 5. Lembar wawancara
Lampiran 6. Lembar dokumentasi wawancara
Lampiran 7. Lembar pengujian *black box*
Lampiran 8. Lembar pengujian eksternal di lokasi studi kasus
Lampiran 9. Lembar dokumentasi pengujian dilokasi Studi Kasus
Lampiran 10. Form kusioner usability testing
Lampiran 11. Surat tugas penguji seminar
Lampiran 12. Listing Program
Lampiran 13. Berita Acara Sidang T.A

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Air mempunyai peranan sangat penting dalam keberlangsungan kehidupan makhluk di bumi, dan fungsinya tidak dapat digantikan oleh senyawa lain. Saat ini, pemenuhan kebutuhan air minum masyarakat banyak mengandalkan Depot Air Minum isi ulang karena harganya yang ekonomis dan mudah didapat. Di saat yang bersamaan, perkembangan zaman menuntut efisiensi di berbagai sektor usaha melalui pemanfaatan teknologi. Penerapan *Internet of Things* (*IoT*) yang dipadukan dengan analisis data *real-time* terbukti memberikan kesempatan bagi UMKM untuk meningkatkan efisiensi pengelolaan operasional dan keuangan (Wardana & Sitompul, 2026). Tren digitalisasi ini juga didukung oleh pergeseran kebiasaan masyarakat menuju pembayaran digital seperti *Quick Response Code Indonesian Standard* (*QRIS*) yang menawarkan kemudahan, keamanan, dan transparansi transaksi.

Namun, pemanfaatan teknologi tersebut belum sepenuhnya dirasakan pada sektor layanan air minum akar rumput. Pada Depot Air Minum Bluebub, proses operasional saat ini masih berjalan sangat konvensional dan menimbulkan berbagai masalah. Dari sisi pelayanan, sebagian pelanggan diwajibkan datang secara fisik membawa galon kosong dan menghabiskan waktu untuk antre, sebuah model yang mulai kurang relevan dengan tingkat mobilitas masyarakat modern yang serba cepat. Dari sisi operasional, pengisian air dilakukan secara manual dengan

membuka keran, yang rentan terhadap kelalaian manusia (*human error*) seperti air tumpah dan takaran volume yang tidak konsisten. Selain itu, transaksi yang sepenuhnya menggunakan uang tunai memunculkan rentetan kelemahan baru, mulai dari kesulitan petugas menyediakan uang kembalian, potensi kesalahan pencatatan pembukuan harian, hingga masalah higienitas karena petugas harus memegang uang dan galon secara bersamaan.

Mengatasi permasalahan lapangan tersebut, diperlukan sebuah sistem otomasi yang terintegrasi. Beberapa penelitian sebelumnya sebenarnya telah mencoba menerapkan otomasi pada depot air minum. Misalnya, pada penelitian yang dilakukan oleh Saputra et al. (2018), pemanfaatan teknologi difokuskan pada sistem monitoring stok tangki air menggunakan sensor ultrasonik. Demikian pula dengan beberapa penelitian terkait lainnya yang umumnya hanya berfokus pada otomasi sensor keluaran air semata, tanpa menyentuh aspek transaksi bisnisnya. Padahal, untuk memodernisasi UMKM seperti Depot Bluebub secara menyeluruh, otomasi perangkat keras (dispenser) saja tidaklah cukup. Dibutuhkan sebuah ekosistem yang tidak hanya mampu menakar air secara presisi, tetapi juga terhubung dengan layanan transaksi pembayaran digital guna memfasilitasi model bisnis pesan-antar dari jarak jauh secara aman.

Oleh karena itu, penelitian ini mengusulkan inovasi berupa rancang bangun *prototype* yang terintegrasi dengan sistem bisnis terotomatisasi berbasis pemesanan jarak jauh (*Online-to-Offline*). Melalui sistem ini, pelanggan dapat mengakses tautan *website* dari rumah untuk melakukan pemesanan dan pembayaran langsung menggunakan layanan *payment gateway Xendit* (*QRIS*). Setelah transaksi

terverifikasi otomatis melalui *webhook*, sistem server akan menampilkan notifikasi pesanan masuk secara *real-time* pada halaman *dashboard admin.* Mikrokontroler *ESP32* dan sensor *water flow* kemudian akan bekerja memompa dan menakar volume air secara presisi tanpa campur tangan manual. Galon yang telah terisi penuh selanjutnya akan diantarkan oleh petugas ke rumah pelanggan menggunakan sistem tukar galon kosong. Penelitian ini diharapkan tidak hanya menyempurnakan celah dari penelitian-penelitian sebelumnya, namun juga mampu mentransformasi layanan Depot Air Minum Bluebub menjadi lebih modern, praktis, dan menjangkau pasar yang lebih luas.

1.2 Perumusan Masalah

Berdasarkan latar belakang yang telah dipaparkan, maka rumusan masalah dalam penelitian ini dapat dirumuskan "Bagaimana merancang bangun *prototype* berbasis *Internet of Things* yang terintegrasi dengan sistem pemesanan *website* dan layanan *payment gateway Xendit* (*QRIS*), sehingga proses transaksi pembayaran dan penakaran air dapat berjalan secara otomatis dari jarak jauh?"

1.3 Batasan Masalah

Agar penelitian lebih terarah dan tidak meluas, maka ditetapkan batasan masalah sebagai berikut:

a. Sistem kendali utama pada *prototype* menggunakan mikrokontroler *ESP32* berbasis *Internet of Things* (*IoT*).
b. Metode pembayaran hanya berfokus pada penggunaan *QRIS* Dinamis melalui *API Payment Gateway Xendit*.

c. Otomatisasi perangkat keras hanya sebatas pada pembacaan keberadaan galon (menggunakan sensor *infrared*) dan penakaran debit air (menggunakan *water flow sensor*), sedangkan proses peletakan dan pengantaran galon tetap dilakukan secara manual oleh petugas depot.

d. Antarmuka sistem dibangun berbasis *website* (bukan aplikasi *mobile/APK*) sebagai *prototype* (*Proof of Concept*), menggunakan *CodeIgniter* 4, *MySQL*, dan *Ngrok*. Adapun keamanan privasi dan data transaksi pelanggan sepenuhnya ditangani oleh sistem enkripsi *Payment Gateway Xendit*.

e. Penelitian pada Depot Air Minum Bluebub ini bertindak semata-mata sebagai batasan studi kasus untuk menganalisis alur kebutuhan operasional konvensional.

f. Sistem yang dibangun murni sebatas *prototype* (*Proof of Concept*) untuk keperluan uji coba teknis, dan tidak dirancang untuk diimplementasikan secara komersial maupun operasional permanen di lokasi depot.

g. Sistem pembayaran digital yang digunakan pada penelitian ini menggunakan layanan *payment gateway Xendit* dalam mode pengujian (*sandbox/development mode*), sehingga transaksi yang dilakukan hanya digunakan untuk keperluan simulasi dan pengujian sistem.

1.4 Tujuan dan Manfaat

a. Tujuan

Tujuan penelitian ini adalah merancang bangun *prototype* berbasis yang terintegrasi dengan *website* pemesanan jarak jauh dan *payment gateway Xendit*, guna mengotomatisasi proses verifikasi pembayaran *QRIS* dan penakaran volume

air, sehingga mampu meningkatkan efisiensi operasional serta memperluas jangkauan pelayanan di Depot Air Minum Bluebub.

b. Manfaat

Berdasarkan tujuan penelitian di atas, manfaat yang diharapkan dari pengembangan sistem *prototype Smart* Dispenser ini adalah sebagai berikut:

1) Efisiensi Operasional dan Finansial UMKM

Sistem ini memodernisasi manajemen keuangan depot air minum melalui otomatisasi rekapitulasi data. Melalui integrasi *payment gateway Xendit* (*QRIS*), seluruh aliran dana tervalidasi secara *real-time* sehingga laporan pembukuan harian dipastikan seimbang (*balance*) dan terhindar dari risiko selisih uang akibat transaksi manual. Selain itu, beban kerja fisik operasional menjadi jauh lebih ringan karena proses penakaran volume air dieksekusi secara otomatis oleh alat .

2) Transformasi Layanan dan Kepraktisan Konsumen

Dari segi pelayanan, penelitian ini mentransformasi model bisnis konvensional menjadi ekosistem layanan pesan-antar (*Online-to-Offline*). Hal ini memberikan kepraktisan yang tinggi bagi masyarakat karena menghilangkan keharusan antre dan membawa galon kosong secara fisik ke lokasi depot. Konsumen juga mendapatkan jaminan takaran air minum yang pasti dan lebih higienis karena proses pengisian dikendalikan langsung oleh mesin bersensor.

3) Pengembangan Ilmu Pengetahuan dan Teknologi Terapan

Secara akademis, penelitian ini memberikan sumbangsih literatur terapan yang membuktikan bahwa perangkat keras *Internet of Things* (*IoT*) berbasis mikrokontroler dapat diintegrasikan secara utuh dengan layanan *Payment Gateway* jarak jauh. Hasil penelitian ini dapat menjadi landasan pijakan (*Proof of Concept*) bagi pengembangan penelitian selanjutnya dalam merancang sistem otomasi mesin berbasis pembayaran digital untuk sektor UMKM lainnya.

1.5 Sistematika Penulisan

Adapun sistematika penulisan dalam penulisan tugas akhir ini terdiri dari bebrapa bab dan masing-masing bab membahas dan menguraikan pokok permasalahan yang berbeda, sebagai gambaran disini penulis menyertakan garis-garis besarnya yaitu:

BAB I PENDAHULUAN

Bab ini berisikan latar belakang masalah, rumusan masalah, Batasan masalah, tujuan dan manfaat, serta sistematika penulisan.

BAB II LANDASAN TEORI

Bab ini berisikan tentang tinjauan pustaka yang diambil dari penelitian yang relevan beserta susunan kajian teori yang disesuaikan dengan tema Tugas Akhir.

BAB III METODOLOGI PENELITIAN

Bab ini berisikan tentang tahapan yang dilakukan peneliti dalam mengumpulkan informasi atau data yang dibutuhkan.

BAB II

TINJAUAN PUSTAKA

2.1 Kajian Teori

2.1.1 *Internet Of Things* (IoT)

Internet of things adalah sebuah istilah yang muncul dengan pengertian sebuah akses perangkat elektronik melalui media internet atau tanpa internet. Akses perangkat tersebut terjadi akibat hubungan melalui manusia dengan perangkat atau perangkat dengan perangkat dengan memanfaatkan jaringan *internet*. Akses perangkat tersebut terjadi karena keinginan untuk berbagi data, berbagi akses dan juga mempertimbangkan keamanan dalam aksesnya. *Internet of thing* dimanfaatkan sebagai media pengembangan kecerdasan akses perangkat di dunia industri, di rumah tangga dan beberapa sektor yang sangat luas dan beragam. *Internet of thing* dapat dikembangkan dengan media perangkat elektronika yang umum seperti Arduino untuk keperluan yang spesifik bisa disebut juga khusus (Bella, 2023).

Mengacu pada pernyataan penelitian sebelumnya, penerapan *Internet of Things* (*IoT*) pada *prototype Smart* Dispenser ini diimplementasikan sebagai media komunikasi nirkabel untuk mengelola akses dan pertukaran data secara *real-time*. Pemanfaatan ini menghadirkan kebaruan, dengan menghubungkan perangkat keras (mikrokontroler) secara langsung dengan *website* dan *payment gateway* jarak jauh. Melalui mekanisme ini, operasional *Smart* Dispenser tidak lagi terbatas pada *self-service* di lokasi fisik, melainkan mampu mendukung ekosistem *Online-to-Offline* (pesan-antar). Sistem dapat memproses transaksi pelanggan dan mengendalikan perangkat keras secara terotomatisasi, mandiri, dan terjamin tingkat keamanannya.

2.1.2 Mikrokontroler ESP32

NodeMCU ESP32 adalah modul pengembangan dengan mikrokontroler *ESP32* yang mendukung *Wi-Fi* dan *Bluetooth*, ideal untuk aplikasi . Fitur utama termasuk *dual core processor*, memori *RAM* 520 *KB*, dan *flash memory* hingga 4 *MB*. Digunakan untuk menghubungkan dan mengontrol perangkat melalui jaringan *internet* (Putri et al., 2024).

Dapat disimpulkan dari penelitian sebelumnya bahwa, mikrokontroler dari *NodeMCU ESP32* cocok untuk proyek *Smart* Dispenser karena keunggulan dari fitur *Wi-Fi* bawaan. Fitur tersebut menjadi syarat agar perangkat keras dapat terus terhubung dengan *internet* dan mampu merespon instruksi *webhook* dari *payment gateway* secara *real-time* untuk menyalakan pompa air. Dengan kata lain, ESP32 bertindak sebagai otak fisik yang menggantikan peran operator manusia untuk mengeksekusi perintah dari internet ke dalam bentuk kerja mekanis.



Gambar 2. 1 *NodeMCU ESP32*

2.1.3 *Payment Gateway*

Payment gateway adalah salah satu cara untuk memproses transaksi elektronik. *Payment gateway* menyediakan alat – alat untuk memproses pembayaran antara *customer*, *businesses* dan *banks*. *Payment gateway* merupakan

bagian terpenting dari suatu transaksi antar *customer*, *business,* dan lembaga – lembaga perbankan yang keduanya digunakan. *Payment Gateway* digunakan untuk memfasilitasi transaksi elektronik (Mikroskil, 2012).

Pada penelitian (Siahaan & Sianturi, 2024) perbandingan *payment gateway* ini dengan membuat daftar fitur dan fungsionalitas lalu membandingkan fitur setiap *payment gateway*. Semua *payment gateway* melakukan transaksi *real-time*. Pada pelaporan transaksi, *Prismalink* tidak dapat dilihat *dashboard* laporan transaksinya. Semua *payment gateway* dapat diintegrasikan pada *platform* bisnis. *Midtrans, iPaymu, iPay88, Xendit, Prismalink* dan Duitku mendukung metode pembayaran yang lengkap. *Payment gateway* yang mendukung konversi mata uang asing adalah *Doku, iPaymu, Xendit* dan Duitku. *Payment gateway* yang memiliki sistem keamanan yang lengkap adalah *Midtrans, Doku*, *Xendit* dan Duitku.

Mengacu pada teori penelitian sebelumnya, penerapan *payment gateway* pada proyek ini disimpulkan berfungsi sebagai alat pemroses transaksi elektronik yang aman dan terintegrasi. *Payment gateway* menggantikan peran verifikasi manual dengan memfasilitasi jalur komunikasi finansial antara pelanggan, sistem *website Smart* Dispenser, dan lembaga perbankan. Otomatisasi transaksi ini memungkinkan operasional dispenser berjalan secara mandiri dari jarak jauh setelah pembayaran tervalidasi secara sistem.

2.1.4 *Xendit* (*API*)

Xendit adalah infrastruktur pembayaran digital yang memungkinkan bisnis dan organisasi nirlaba menerima pembayaran dari berbagai saluran dengan satu

pintu integrasi. Dalam konteks sistem informasi, *Xendit* bertindak sebagai middleware yang mengamankan dan memvalidasi transaksi (Tafonao & Nadia Raslina, 2025).

Jadi, *Xendit* adalah perusahaan penyedia infrastruktur pembayaran digital (*Payment Gateway*), bukan sebuah entitas bank tunggal. *Xendit* bertindak sebagai fasilitator pihak ketiga yang menjalin kerja sama resmi dengan berbagai lembaga perbankan di Indonesia (seperti Bank Mandiri, BCA, BRI, BNI) serta berbagai penyedia *e-wallet*. Bagi UMKM, fungsi utama *Xendit* adalah memproses dan merekap seluruh transaksi *QRIS* secara otomatis dan *real-time*. Hal ini memastikan bahwa laporan keuangan harian depot menjadi rata (*balance*) antara takaran air yang keluar dengan uang yang masuk, sehingga terhindar dari risiko selisih hitung pada transaksi tunai manual.

Berdasarkan fungsi tersebut, penggunaan *Xendit* pada pengembangan ini berperan krusial sebagai *middleware*. Dalam ekosistem bisnis *Online-to-Offline* (O2O), *Xendit* menjadi jembatan yang menghubungkan verifikasi pembayaran di *website* pelanggan dengan mikrokontroler. Melalui validasi otomatis ini, sistem memastikan bahwa perangkat keras hanya akan memompa air ke dalam galon setelah dana dinyatakan berhasil masuk, menciptakan alur layanan jarak jauh yang aman dan terpercaya.

2.1.5 *QRIS*

QRIS adalah metode standar nasional dari Bank Indonesia (BI) dalam bentuk *QR Code* Nasional yang difungsikan untuk memberikan fasilitas merupakan standar kode *QR* Nasional untuk memfasilitasi pembayaran secara digital (*QRIS*, 2019).

Pembayaran digital melalui *QRIS* ditujukan untuk menggabungkan dan memberikan standar nasional pada dompet digital. (Suprapto & Kusuma, 2022).

Adapun dua jenis *QRIS* yang di tawarkan oleh BI (Bank Indonesia) menurut (Suprapto & Kusuma, 2022),yaitu *QRIS Static* dan *Dynamic*. *QRIS Static* menjadi jenis layanan yang paling sering digunakan oleh penjual saat ini karena dapat dicetak di kertas. Informasi yang ada di dalam *QRIS Static* hanya terdapat IDE Penjual saja. Penggunaan *QRIS* jenis *QRIS Static*, pembeli perlu memindai dan memasukkan nominal yang akan dibayarkan sebelum melakukan transaksi. Hal ini berbeda dengan *QRIS Dynamic* yang sudah terdapat informasi mengenai *ID* Penjual serta nominal transaksi didalamnya dan akan memberikan *QR Code* yang berbeda untuk nominalnya. *QRIS Dynamic* lebih cocok digunakan untuk sistem pembayaran yang terintegrasi dengan sistem verifikasi otomatis sehingga keluaran/produk yang diperjualbelikan dapat dieksekusi tanpa bantuan operator.

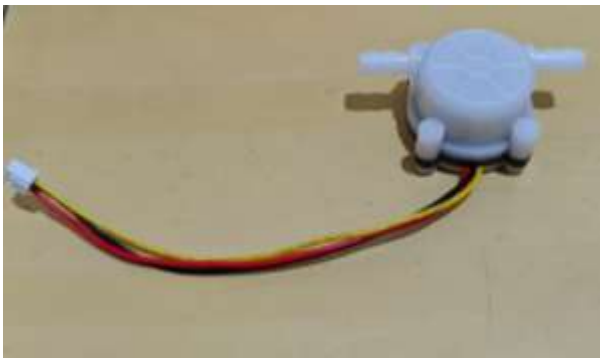
Berdasarkan tinjauan penelitian sebelumnya, penerapan *QRIS* pada sistem ini disimpulkan sebagai metode utama pembayaran digital terstandardisasi. Mengingat *Smart* Dispenser dirancang untuk beroperasi secara mandiri jarak jauh, sistem ini secara spesifik mengimplementasikan jenis *QRIS Dynamic*. Hal ini dikarenakan *QRIS Dynamic* mampu menyertakan informasi nominal transaksi secara otomatis ke dalam kode *QR*, sehingga sangat mendukung proses verifikasi pembayaran *real-time* yang terintegrasi langsung dengan mikrokontroler tanpa bantuan operator manusia.

2.1.6 Sensor *Flowmeter* (YF-S401)

Water flow sensor merupakan sebuah perangkat sensor yang digunakan untuk mengukur debit air. Biasanya *water flow* sensor adalah elemen (bagian) yang digunakan pada *flow* meter. Sebagaimana pada sebuah sensor, keakuratan mutlak pada pengukuran memerlukan fungsi untuk pengkalibrasian sensor (Hafizh Andifaisa et al., 2021).

Menurut (Pangestu *et al*., 2024) persentase kesalahan (*percentage error*) digunakan untuk mengetahui tingkat perbedaan antara nilai hasil pengukuran dan nilai acuan yang digunakan. Semakin kecil nilai persentase kesalahan yang diperoleh, maka semakin tinggi tingkat akurasi hasil pengukuran. Persentase kesalahan dapat dihitung menggunakan Persamaan berikut.

$$\text{Presentase Kesalahan} = \frac{\text{Nilai yang diterima} - \text{Nilai yang diukur}}{\text{Nilai yang diterima}} \text{ x } 100\%$$



Gambar 2.2 *Flowmeter* (YF-S401)

2.1.7 Sensor *Infrared* (E18-D80NK)

Sensor Jarak Inframerah *E18-D80NK* adalah sensor non-kontak ringkas yang dirancang untuk mendeteksi objek dan rintangan sederhana. Ia bekerja dengan memancarkan cahaya inframerah dan mendeteksi sinyal yang dipantulkan dari

objek terdekat. Ketika suatu objek memasuki rentang deteksi yang telah ditetapkan, sensor mengubah status keluaran digitalnya, membuatnya mudah untuk dihubungkan dengan mikrokontroler dan sirkuit kontrol. *LED* indikator internal memberikan sinyal visual yang jelas saat deteksi terjadi.

Menurut (Kamaluddin *et al*., 2025), sensor inframerah tipe *E18-D80NK* dimanfaatkan untuk menghitung jumlah siklus penggunaan dengan cara mendeteksi keberadaan suatu objek. Ketika sebuah objek berada didepan sensor dan berada dalam jangkauan deteksi, maka sensor akan menghasilkan sinyal keluaran bernilai "1" atau "*high*", yang menunjukkan bahwa objek tersebut berhasil dikenali oleh sensor. Sebaliknya, apabila objek berada di luar area deteksi, sensor akan memberikan output berupa "0" atau "*low*" yang berarti objek tersebut tidak terdeteksi. Sensor ini memiliki kemampuan deteksi jarak yang cukup jauh serta tingkat sensitivitas tinggi terhadap cahaya yang menghalangi jalurnya.

Berdasarkan penelitian sebelumnya, dapat disimpulkan bahwa sensor inframerah *E18-D80NK* pada *Smart* Dispenser berfungsi sebagai detektor keberadaan wadah (galon) secara non-kontak. Dalam sistem otomasi jarak jauh ini, sinyal digital dari sensor menjadi syarat mutlak bagi mikrokontroler guna memastikan pompa air hanya menyala ketika galon telah diletakkan dengan benar di area pengisian, sehingga mencegah tumpahnya air. Sensor ini bertindak sebagai sistem keamanan fisik (*safety*) yang menutupi kelemahan pembayaran *online* jarak jauh, sehingga air tidak akan tumpah terbuang sia-sia meskipun pelanggan menekan tombol bayar dari rumah.



Gambar 2.3 *Sensor Infrared (E18-D80NK)*

2.1.8 *Modul Relay*

Relay merupakan komponen elektronika berupa saklar atau *switch* elektrik yang dioperasikan secara listrik dan terdiri dari 2 bagian utama yaitu Elektromagnet (*coil*) dan mekanikal (seperangkat kontak Saklar/*Switch*). Komponen elektronika ini menggunakan prinsip elektromagnetik untuk menggerakan saklar sehingga dengan arus listrik yang kecil (*low power*) dapat menghantarkan listrik yang bertegangan lebih tinggi. Berikut adalah simbol dari komponen *relay* (Danindra Riski *et al*., 2019).

Mengacu pada penelitian di atas, penerapan modul *relay* disimpulkan sebagai komponen esensial untuk mengendalikan sirkuit daya tinggi menggunakan sinyal daya rendah. Dalam operasional *Smart* Dispenser, *relay* bertugas menerima sinyal digital dari mikrokontroler (setelah pembayaran terverifikasi) untuk kemudian menghubungkan arus listrik utama ke pompa air, menjaga agar mikrokontroler tetap aman dari lonjakan tegangan.



Gambar 2.4 *Modul Relay*

2.1.9 *Waterpump DC* 12V

Water Pump merupakan jenis pompa yang menggunakan motor dc dan tegangan searah sebagai sumber tenaganya. Dengan memberikan beda tegangan pada kedua terminal tersebut, motor akan berputar pada satu arah, dan bila polaritas dari tegangan tersebut dibalik maka arah putaran motor akan terbalik pula (Revivaldy Mucthar *et al*., 2020).

Jadi dari penelitian sebelumnya, dapat disimpulkan bahwa *water pump* berfungsi sebagai aktuator utama penggerak cairan pada sistem *Smart* Dispenser. Dalam alur operasional otomatisasi ini, pompa bertugas mengeksekusi pendistribusian air secara fisik ke dalam wadah pelanggan tepat setelah menerima pasokan tegangan listrik yang dipicu oleh mikrokontroler melalui modul *relay*.



Gambar 2.5 *Waterpump DC* 12V

2.1.10 *Modul LCD* 16x2 I2C

LCD I2C adalah modul tampilan berbasis *LCD* 16x2 (16 karakter, 2 baris) atau 20x4 yang terhubung ke *Arduino* menggunakan komunikasi *I2C*, yaitu hanya menggunakan 2 pin saja (*SDA* dan *SCL*) (Muhammad Mujab Fachni *et al*., 2025).

Merujuk pada penelitian sebelumnya, penerapan modul *LCD* I2C disimpulkan sebagai komponen krusial untuk memberikan *feedback* visual terkait status sistem. Dalam implementasi *Smart* Dispenser, layar ini secara efektif menampilkan informasi proses bisnis yang sedang berjalan kepada pengguna, sembari meminimalisir kompleksitas jalur kabel pada mikrokontroler berkat penggunaan protokol komunikasi I2C.



Gambar 2.6 *Modul LCD* 16x2 I2C

2.1.11 Catu Daya (*Adaptor* 12V)

Adaptor adalah rangkaian listrik yang mampu menurunkan tegangan dan menyearahkan arus AC menjadi DC. Hal ini dikarenakan *adaptor* terdiri atas banyak dioda, yakni komponen yang mampu mengubah arus bolak balik menjadi arus searah. *Adaptor* bisa digunakan sebagai alternatif sumber tegangan DC selain baterai, akumulator, dan catu daya (N Somayasa et al., 2024).

Pada penelitian sebelumnya, disimpulkan bahwa catu daya (*adaptor*) dimanfaatkan untuk menghantarkan masuk keluarnya arus dari bolak balik menjadi searah, yang dapat juga digantikan sebagai alternatif dari sumber tegangan lainnya.



Gambar 2.7 *Adaptor* 12V 2A

2.1.12 *CodeIgniter 4* (CI4)

Codeigniter merupakan sebuah aplikasi *open source* yang berupa kerangka kerja atau *framework* untuk membangun *website* menggunakan Bahasa pemrograman *PHP*. Tujuannya memungkinkan pengembangan proyek jadi lebih cepat daripada menuliskan kode dasar atau kode terstruktir, dengan menyediakan banyak *library* yang biasanya digunakan dalam pengerjaan (Hanayah et al., 2024).

Jadi dari penelitian sebelumnya, dapat disimpulkan bahwa penerapan *CodeIgniter* disimpulkan sebagai solusi efisiensi dalam pengembangan perangkat lunak pada sistem ini. *Framework* ini menyediakan struktur kode dan *library* bawaan yang memfasilitasi pembangunan *website* transaksi secara cepat, rapi, dan aman, yang bertugas menjembatani komunikasi data antara pengguna jarak jauh dengan perangkat keras dispenser.

2.1.13 Dispenser

Dispenser merupakan sebuah alat atau model benda yang digunakan untuk memudahkan masyarakat dalam minum, selain itu dispenser juga umum nya untuk memberikan kemudahan atau bagi para masyarkat dalam mengeluarkan air dalam jumlah tertentu (Wijaya & Ceng Giap, 2024).

Dari penelitian sebelumnya, dapat disimpulkan fungsi dasar dispenser sebagai fasilitas pengeluaran air disimpulkan tetap menjadi landasan dari alat ini. Namun, kebaruan proyek ini terletak pada integrasinya dengan sistem , di mana kemudahan pengeluaran air dalam takaran spesifik tidak lagi dilakukan secara fisik oleh manusia, melainkan dieksekusi secara otomatis oleh mesin berdasarkan verifikasi pesanan *online*.

2.1.14 *PHP*

PHP adalah bahasa pemrograman *side* yang menangani logika sistem dan proses di balik layar. Dalam konteks *fayucom.store*, *PHP* kemungkinan besar digunakan untuk memproses formulir pembelian, *login/logout* pengguna, mengelola akun *admin*, serta menghubungkan *website* dengan . *PHP* juga mengatur bagaimana data ditampilkan secara dinamis, seperti menampilkan produk berdasarkan kategori atau menampilkan status pesanan pelanggan (Tafonao & Nadia Raslina, 2025).

Berdasarkan peneltian sebelumnya, dapat disimpulkan bahwa *PHP* berfungsi sebagai bahasa pemrograman *server-side* yang mengendalikan seluruh logika *backend* pada *website Smart* Dispenser. Dalam ekosistem pesan-antar ini, skrip *PHP* berperan vital untuk memproses formulir pemesanan pelanggan,

memvalidasi status pembayaran dari *payment gateway*, serta mengelola alur komunikasi *database* agar perintah pengisian air dapat diteruskan ke perangkat mikrokontroler secara dinamis dan *real-time*.

2.1.15 MySQL

MySQL adalah suatu perangkat lunak relasi atau *Relational Management System* (RDBMS) yang didistribusikan gratis di bawah lisensi GPL (*General Public License*). Dimana setiap orang bebas menggunakan *MySQL*, namun tidak boleh dijadikan produk turunan yang dijadikan *closed source* atau komersial (Dinata et al., 2015, dalam Sains et al., 2022).

Mengacu pada penelitian sebelumnya, dapat disimpulkan bahwa *MySQL* berfungsi sebagai sistem manajemen basis data utama dalam pengembangan *website Smart* Dispenser. Dalam ekosistem pesan-antar ini, *MySQL* bertugas menyimpan dan mengelola seluruh relasi data secara terpusat—mulai dari informasi akun pelanggan, riwayat transaksi dari *payment gateway*, hingga pencatatan status operasional perangkat sehingga menjamin integritas dan kelancaran aliran informasi pada sistem.

2.1.16 *XAMPP*

XAMPP adalah sebuah *software* server *apache* yang didalamnya sudah tersedia server *MySQL* dan dapat mendukung pemrograman *PHP*. *XAMPP* merupakan *software* yang mudah digunakan, gratis dan mendukung instalasi di *Linux* dan *Windows*. Keuntungan lainnya adalah cuma menginstal satu kali sudah tersedia *Apache , MySQL, PHP Support* (*PHP 4* dan *PHP 5*) dan beberapa modul lainnya (Dinata *et al*., 2015, dalam Sains *et al*., 2022).

Mengacu pada penelitian tersebut, penerapan *XAMPP* disimpulkan sebagai solusi efisiensi dalam penyediaan infrastruktur server . Berkat kemampuannya merangkum *Apache*, *MySQL*, dan *PHP* dalam satu instalasi praktis, *XAMPP* memfasilitasi pengembang untuk mensimulasikan dan mengelola aliran data operasional *website Smart* Dispenser secara lokal tanpa memerlukan konfigurasi server yang rumit.

2.1.17 *Webhook*

Webhook adalah komunikasi server-ke-server yang hanya berfungsi ketika dipicu oleh peristiwa tertentu. Akibatnya, *webhook* berbagi beban kerja di antara berbagai server berdasarkan tanggung jawabnya, memastikan tidak ada server yang kelebihan beban (Nugraha & Sebastian, 2021, dalam Arsy Akbar & Nerisafitra, 2025).

Mengacu pada literatur di atas, penerapan *webhook* disimpulkan sebagai metode komunikasi *server-to-server* yang krusial untuk menjaga efisiensi sistem. *webhook* memastikan bahwa instruksi untuk mengaktifkan pompa air hanya akan dipicu oleh satu peristiwa spesifik yaitu validasi transaksi yang berhasil dari pihak perbankan, sehingga pendistribusian air berjalan sepenuhnya otomatis secara sinkron tanpa menyebabkan kelebihan beban kerja (*overload*) pada server operasional.

2.1.18 *JSON*

JSON adalah format data ringan yang digunakan dalam komunikasi antar sistem, biasanya digunakan dalam integrasi *API* (*Application Programming Interface*). Di *fayucom.store*, *JSON* digunakan saat *website* terhubung ke sistem

pembayaran seperti *Xendit*, di mana data transaksi seperti jumlah pembayaran, status, dan *ID* pesanan dikirimkan dalam format *JSON*. *API* memungkinkan pertukaran data ini berjalan otomatis dan aman. Dengan *JSON* dan *API*, proses seperti pembayaran, konfirmasi otomatis, dan pembaruan status pesanan bisa dilakukan tanpa perlu *input* manual dari *admin* (Tafonao & Nadia Raslina, 2025).

Berdasarkan penjelasan tersebut, disimpulkan bahwa penggunaan *JSON* dan *API* merupakan fondasi komunikasi data pada ekosistem *Smart* Dispenser. Dalam model pemesanan online ini, *API* berfungsi sebagai jalur integrasi antara *website* dan pihak ketiga seperti *Xendit*, sementara *JSON* berperan sebagai format standar yang ringan untuk membungkus data transaksi seperti nominal dan status pembayaran. Kombinasi keduanya memungkinkan pertukaran instruksi dari server ke mikrokontroler (*ESP32*) berjalan secara otomatis, cepat, dan aman tanpa intervensi manual.

2.1.19 *Ngrok* (*Tunneling*)

Ngrok merupakan aplikasi yang membuka lokal ke internet dan merupakan proxy dengan metode tunnelling (*ngrok*, 2021, dalam Hafizh Andifaisa *et al*., 2021). *Ngrok* bekerja dengan melewatkan akses lokal ke melalui proxy dan diberikan koneksi dari *Ngrok cloud service* sehingga dapat diakses dari mana saja.

Berdasarkan penelitian sebelumnya, dapat disimpulkan bahwa penggunaan *Ngrok* pada tahap pengembangan berfungsi sebagai jembatan penghubung sementara antara lingkungan server lokal (*XAMPP*) dan jaringan internet publik. Melalui metode *tunnelling*, *Ngrok* memungkinkan *website* yang masih berjalan secara lokal untuk menerima komunikasi *webhook* dari *payment gateway* (*Xendit*)

secara *real-time*, sehingga pengujian alur otomatisasi transaksi jarak jauh dapat dilakukan tanpa memerlukan *server hosting* berbayar terlebih dahulu.

2.1.20 Skala Likert

Metode Skala Likert merupakan metode yang digunakan untuk menggukur tingkat kepuasan pengguna menggunakan skala likert. Skala likert adalah skala pengukuran yang dikembangkan oleh Likert pada Tahun 1932. Skala likert memiliki empat atau lebih butir-butir pertanyaan yang dikombinasikan sehingga membentuk sebuah skor atau nilai yang mempersentasikan sifat individu, misalkan pengetahuan, sikap dan perilaku. Skala likert dapat juga dikatan sebagai skala psikometrik yang umum digunakan dalam kuesioner dan merupakan skala yang paling banyak digunakan untuk penelitian. (Setyawan & Walter, 2018).

Menurut (Septiana & Firdonsyah, 2025) metode Skala Likert dapat digunakan untuk mengetahui tingkat penerimaan pengguna terhadap suatu sistem berdasarkan hasil kuesioner yang telah diberikan kepada responden. Pada penelitian ini, setiap jawaban responden diberikan bobot nilai yang berbeda. Pilihan jawaban Sangat Setuju (SS) diberikan nilai 5, Setuju (S) bernilai 4, Netral (N) bernilai 3, Tidak Setuju (TS) bernilai 2, dan Sangat Tidak Setuju (STS) bernilai 1.

Perhitungan skor setiap jawaban dapat dilakukan menggunakan persamaan berikut.

$$\text{Skor } = T \; x \; Pn$$

Dengan keterangan yatu, T sebagai jumlah responden yang memilih jawaban tertentu dan Pn sebagai bobot jawaban.

Lalu, Setelah seluruh skor diperoleh, dilakukan perhitungan indeks persentase untuk mengetahui tingkat penerimaan responden terhadap sistem yang diuji. Persamaan yang digunakan adalah sebagai berikut.

$$\text{Indeks}(\%) = \frac{\text{Total Skor}}{\text{X}} \text{ x } 100$$

Dengan keterangan yaitu, *Total Skor* adalah jumlah seluruh skor jawaban responden dan *X* adalah *Skor Tertinggi* x *jumlah responden* x *jumlah pertanyaan*.

Kemudian untuk menentukan kategori hasil interpretasi, digunakan perhitungan interval Skala Likert sebagai berikut.

$$\text{Interval} = \frac{100}{\text{Jumlah Skala Likert}}$$

Karena penelitian ini menggunakan Skala Likert 5 poin, maka;

$$\text{Interval} = \frac{100}{5}$$

$$\text{Interval} = 20$$

Nilai interval sebesar 20 digunakan untuk menentukan kategori interpretasi hasil kuesioner.

2.2 Penelitian Yang Relevan

Penelitian terdahulu yang berkaitan dengan penelitian yang sedang dilakukan, sangat diperlukan sebagai referensi dan acuan dalam penlitian ini. Pada tabel 2.1 berikut, disajikan beberapa penelitian terdahulu yang relevan dengan topik penlitian ini.

Table 2.1 Penelitian yang relevan

No	Penulis / Tahun	Topik Penelitian	Metode	Pembahasan	Hasil
1	(Wahanie et al., 2023)	Pengembangan sistem meteran listrik prabayar berbasis *Internet of Things* (IoT) yang terintegrasi dengan *payment gateway* untuk transaksi *online* menggunakan QRIS.	*Prototyping*	Sistem ini dirancang untuk menyelesaikan masalah penagihan listrik *flat* (digabung biaya sewa kamar) yang sering kali merugikan pihak pengelola kost maupun penyewa yang hemat listrik. Prinsip kerja datanya adalah saat pembayaran via *QRIS* berhasil, *Xendit API* mengirim notifikasi ke *database*. ESP32 secara berkala meminta data saldo (*balance*) tersebut ke *server*, lalu mengonversi nominal rupiah (misal: Rp 1.500) menjadi takaran kuota kWh.	Pengujian akurasi membuktikan bahwa sensor PZEM-004T memiliki keandalan yang sangat baik, di mana pembacaan parameter energinya memiliki nilai yang mendekati sama persis dengan alat ukur pembanding (SDM 120M). Uji coba fungsional menunjukkan bahwa perangkat berhasil menerima *top-up* pulsa listrik yang dibayar menggunakan aplikasi OVO dan Bank Mandiri melalui pemindaian *QRIS*.
2	(Nurul et al., 2024)	Penelitian ini berfokus pada pengembangan dan implementasi purwarupa yang menggunakan teknologi *RFID-card* untuk mencatat, mendeteksi, serta memonitoring jumlah konsumsi air minum harian secara otomatis.	*Prototyping*	Pembahasan utama berfokus pada pengujian jarak optimal bagi *RFID reader* (MFRC522) dalam mendeteksi kartu identitas pengguna secara *contactless* (tanpa kontak fisik). Pengujian dilakukan dengan menempatkan kartu RFID pada variasi jarak mulai dari 2 cm hingga 12 cm dari sensor, yang diukur menggunakan mistar untuk akurasi.	Pengujian membuktikan adanya hubungan positif (berbanding lurus) antara jarak sensor ke kartu RFID dengan waktu respons (*response time*); di mana semakin jauh jarak kartu, semakin lama waktu respons yang dibutuhkan sensor untuk memproses data. Sensor RFID terbukti dapat mendeteksi kartu secara konsisten pada jarak 0-10 cm, namun mengalami kegagalan deteksi pada rentang jarak 11-12 cm akibat

No	Penulis / Tahun	Topik Penelitian	Metode	Pembahasan	Hasil
					penurunan kekuatan sinyal.
3	(Abadi et al., 2021)	Penelitian ini berfokus pada pengembangan smart water dispenser berbasis Internet of Things (IoT) yang terintegrasi dengan aplikasi Android dan *cloud server* (*Google Firebase*). Sistem ini bertujuan untuk memonitor, mencatat, serta menentukan jumlah kebutuhan air minum harian yang ideal bagi pengguna secara otomatis.	*Prototyping*	Untuk menentukan target kebutuhan air yang spesifik bagi setiap individu, sistem memanfaatkan basis pengetahuan berupa kuesioner *General Practice Physical Activity Questionnaire* (GPPAQ) dan formula *Estimated Energy Requirement* (EER). Aplikasi Android meminta pengguna memasukkan data usia, jenis kelamin, dan berat badan untuk memetakan tingkat aktivitas fisik.	Implementasi metode perhitungan berbasis GPPAQ dan EER berhasil berjalan dengan baik tanpa ada perbedaan hasil dengan perhitungan manual. Pengujian pada perangkat keras menunjukkan *water flow* sensor memiliki tingkat presisi/akurasi rata-rata sebesar 92,6% saat mengeluarkan air. Uji fungsionalitas mencatat bahwa perangkat memerlukan perbaikan fisik pada linear *motion track* untuk menghindari *overheat* pada komponen stepper motor.
4	(Pengendali Berbasis Android Asriyadi et al., 2021)	Penelitian ini berfokus pada perancangan smart water dispenser berbasis *Internet of Things* (IoT) yang terhubung dengan aplikasi smartphone. Sistem ini dirancang untuk menakar volume keluaran air secara presisi, menghitung target kebutuhan air minum harian pengguna secara otomatis, serta memantau riwayat konsumsinya secara real-time.	*Prototyping*	Sistem ini menggunakan kuesioner *General Practice Physical Activity Questionnaire* (GPPAQ) dan *formula Estimated Energy Requirement* (EER) pada aplikasi Androidnya untuk menghitung target cairan ideal berdasarkan usia, berat badan, dan aktivitas fisik pengguna. Alat fisik dispenser dibangun menggunakan mikrokontroler yang terhubung	Algoritma penghitungan target kebutuhan air harian pada aplikasi *Android* terbukti berhasil berjalan 100% sesuai dengan hasil hitungan matematis manual. Dari sisi perangkat keras, implementasi water flow sensor terbukti sangat andal dengan tingkat presisi dan akurasi pembacaan volume air mencapai rata-rata 92,6%. Pengujian antarmuka pengguna (*Cognitive Walkthrough*) menunjukkan

No	Penulis / Tahun	Topik Penelitian	Metode	Pembahasan	Hasil
				dengan *water flow* sensor untuk membaca debit air yang masuk ke dalam gelas.	bahwa fitur penuangan air otomatis sangat disukai dan mudah digunakan, meskipun disarankan adanya perbaikan mekanis pada stepper motor dispenser untuk menghindari panas berlebih (*overheat*).
5	(Ralda et al., 2022)	Merancang dan membuat sistem dispenser otomatis berbasis Arduino Uno. Tujuannya adalah mempermudah pengguna dalam mengambil air minum tanpa harus membuka dan menutup keran secara manual, sehingga lebih efektif dan efisien.	*Prototyping*	Arduino Uno dilekatkan pada dinding sisi bagian kiri dalam dispenser. Motor servo dipasang pada dudukan kayu di bagian depan dalam, lalu dihubungkan dengan kawat ke sisi tuas dispenser. Sensor ultrasonik dipasang di bagian depan luar dispenser (melalui dua lubang) agar posisinya pas untuk "melihat" gelas.	Penelitian menyimpulkan bahwa dispenser otomatis yang dirancang berhasil bekerja dengan baik sesuai tujuannya untuk mempermudah pengambilan air. Sensor dan sistem mampu merespons dengan presisi, di mana alat secara otomatis mengisi gelas saat diletakkan pada jarak antara 0 cm hingga 5 cm. Penggerak mekanisnya berhasil merespons perintah dengan membuka keran dispenser sejauh 1 cm saat air mengalir.

Sub bab ini menguraikan penelitian sebelumnya yang berkaitan dengan sistem , khususnya penggunaan *Internet of Things* (*IoT*), sensor penakar air (*water flow*), dan integrasi *payment gateway*. Penelitian-penelitian tersebut menjadi acuan dalam perancangan penelitian ini, yaitu:

a. Wahanie dkk. (2023) melakukan penelitian mengenai sistem meteran listrik prabayar berbasis yang terintegrasi dengan *payment gateway* untuk transaksi *online* menggunakan *QRIS*. Penelitian ini relevan karena menjadi acuan utama penerapan *API Xendit* dan *QRIS*, namun memiliki kelemahan karena berfokus pada kendali arus listrik, bukan pada otomasi pengisian fluida (air).
b. Nurul dkk. (2024) merancang menggunakan teknologi *RFID-card* untuk mencatat konsumsi air secara otomatis. Karya ini menjadi acuan dalam konsep otomasi mesin, namun memiliki kelemahan karena hanya beroperasi dalam sensor jarak dekat (*contactless* fisik) dan tidak mendukung pemesanan jarak jauh.
c. Abadi dkk. (2021) mengembangkan *smart water dispenser* berbasis yang terintegrasi dengan aplikasi *Android* untuk menakar kebutuhan air. Penelitian ini relevan dalam hal penggunaan sensor untuk menakar debit air secara presisi, namun kelemahannya belum terintegrasi dengan sistem transaksi atau *payment gateway*.
d. Asriyadi dkk. (2021) merancang *smart water dispenser* berbasis yang terhubung dengan aplikasi *smartphone* untuk memantau volume keluaran air. Acuan ini penting untuk logika penakaran volume air, namun memiliki kelemahan yang sama yaitu ketiadaan fitur pembayaran digital jarak jauh untuk ekosistem komersial.
e. Ralda dkk. (2022) melakukan penelitian mengenai sistem dispenser otomatis berbasis *Arduino Uno* menggunakan sensor ultrasonik untuk mempermudah pengambilan air. Penelitian ini relevan pada aspek otomasi aktuator perangkat

keras, namun kelemahannya hanya berfokus pada sensor fisik jarak dekat tanpa adanya integrasi jaringan internet () maupun pemesanan *online*.

Berdasarkan Tabel 2.1, perbedaan dan kebaruan (*novelty*) dari proposal ini adalah mengintegrasikan keunggulan dari masing-masing penelitian tersebut sekaligus menutupi kelemahannya. Penelitian ini menggabungkan sistem penakaran fluida otomatis dengan keandalan validasi pembayaran *QRIS* jarak jauh. Proposal ini tidak hanya sekadar membuat alat yang mengeluarkan air, melainkan membangun ekosistem bisnis *Online-to-Offline* (O2O) secara utuh, mulai dari pemesanan di *website*, pembayaran non-tunai via *Xendit*, hingga eksekusi pengisian air secara presisi oleh mesin.

BAB III

METODOLOGI PENELITIAN

3.1 Tinjauan Umum

Perancangan Smart Dispenser pada Depot Air Minum Bluebub dikembangkan sebagai *prototype* sistem bisnis *Online-to-Offline* (O2O) berbasis *Internet of Things* (*IoT*). Alur operasional dimulai saat pelanggan memesan dan membayar air dari rumah melalui pemindaian *QRIS* yang diotorisasi secara otomatis oleh *Payment Gateway Xendit*. Apabila berhasil, server lokal yang dijembatani oleh tunneling *Ngrok* akan menampilkan notifikasi pesanan di *dashboard admin* dan mengirimkan sinyal perintah ke mikrokontroler *ESP32* di depot untuk beralih ke mode bersiap (*standby*).

Merespons sinyal tersebut, proses selanjutnya dikendalikan sepenuhnya oleh perangkat keras (*hardware*). Mikrokontroler *ESP32* terlebih dahulu melakukan validasi keamanan fisik menggunakan sensor *Proximity Infrared* (*E18-D80NK*) untuk memastikan galon benar-benar berada tepat di bawah saluran air. Setelah terdeteksi, *ESP32* memicu modul *relay* untuk menyalakan *water pump* DC 12V. Selama pemompaan berlangsung, sensor *water flow* (*YF-S401*) bekerja menakar debit air berdasarkan prinsip *Hall Effect* secara presisi, lalu mematikan pompa seketika secara otomatis saat target volume tercapai. Tahap akhir diselesaikan dengan petugas mengantarkan galon terisi ke rumah pelanggan melalui sistem tukar wadah (*swap system*).

Meskipun telah terotomatisasi, *prototype* ini memiliki beberapa keterbatasan operasional beserta langkah mitigasinya. Kelemahan utama sistem adalah

ketergantungannya pada koneksi internet dan suplai listrik. Sebagai mitigasi risiko, alat ini dirancang memiliki mekanisme saklar manual (*manual override*) agar petugas tetap dapat melayani pengisian air secara konvensional jika terjadi gangguan jaringan (*downtime*) atau terdapat pelanggan yang datang langsung membawa uang tunai. Kelemahan kedua terletak pada aspek mobilitas fisik; otomasi alat murni berfokus pada sensor deteksi dan penakaran air, sehingga proses memindahkan dan mengantar galon tetap membutuhkan tenaga manusia. Terakhir, penggunaan layanan *tunneling Ngrok* pada server lokal membuat tautan *website* berubah dinamis setiap kali server dihidupkan ulang, di mana batasan ini merupakan ruang lingkup pengembangan yang dapat diselesaikan dengan *hosting cloud* berbayar di masa mendatang.

3.2 Jenis penelitian

Jenis penlitian yang digunakan oleh peneliti adalah *Research and Development* (R&D) atau Penelitian dan Pengembangan. Penelitian R&D merupakan metode penelitian yang bertujuan untuk menghasilkan produk tertentu dan menguji keefektifan produk tersebut. Dalam konteks penelitian ini, produk yang dikembangkan adalah *prototype Smart Dispenser* berbasis beserta perangkat lunak pengontrolnya, Pendekatan R&D dipilih karena fokus utama penelitian adalah pada proses pengembangan produk fisik yang fungsional dan validasi kinerjanya, bukan sekedar kajian teoritis.

Adapun model pengembangan sistem yang diterapkan pada penelitian ini tidak menggunakan pendekatan sekuensial tradisional, melainkan menggunakan metode *Prototyping*. Pemilihan metode ini sangat krusial dan didasarkan pada

tujuan penelitian yang berfokus pada pembuatan *prototype* (*Proof of Concept*) integrasi perangkat keras dengan layanan *API Payment Gateway*. Metode *Prototyping* memungkinkan peneliti untuk merakit, menguji, dan mengevaluasi fungsi teknis komponen (sensor, aktuator, mikrokontroler) secara iteratif. Pendekatan ini sangat sesuai untuk pengembangan sistem eksperimental, karena berfokus pada validasi kelayakan teknis (*technical feasibility*) tanpa harus terikat pada implementasi sistem secara final dan permanen di lapangan.

3.3 Desain Penelitian

Desain penelitian menggambarkan langkah-langkah terstruktur yang dilakukan oleh peneliti dari tahap awal hingga penyelesaian proyek. Secara garis besar, tahapan penelitian ini mencakup proses identifikasi awal melalui analisis kebutuhan, perancangan desain sistem, implementasi dan pengkodean, pengujian fungsionalitas, hingga dievaluasi. Rangkaian tahapan sistematis tersebut dapat diilustrasikan secara visual pada Gambar 3.1 berikut:



Gambar 3.1 Alur Penelitian

Pada Gambar 3.1 digambarkan alur penelitian yang dilakukan. Tahapan penelitian ini meliputi:

a) Analisis kebutuhan

Pada tahapan awal ini berfokus pada pengumpulan data dan identifikasi masalah pada Depot Air Minum Bluebub, terutama terkait sistem transaksi yang

masih manual serta kebutuhan untuk mengembangkan model bisnis pesan-antar jarak jauh. Hasil analisis ini adalah spesifikasi teknis untuk perangkat keras (mikrokontroler *ESP32*, sensor air *YF-S401*, sensor *infrared E18-D80NK*, *relay*) dan perangkat lunak (*framework CI4*, *Xendit API*, *MySQL*, *Ngrok* sebagai penjembatan akses publik) yang dibutuhkan, serta melakukan observasi dan wawancara kepada pemilik depot tentang masalah yang dialami.

b) Desain sistem

Berdasarkan kebutuhan yang teridentifikasi, dilakukan perancangan sistem sebelum implementasi fisik. Tahap ini menghasilkan Diagram Blok Sistem, *Flowchart* (yang mencakup alur pemesanan pelanggan dari rumah hingga pengantaran galon oleh petugas), skema rangkaian kelistrikan (*Wiring Diagram*), serta perancangan basis data dan antarmuka (mencakup halaman pemesanan publik dan *dashboard* notifikasi *admin*).

c) Implementasi dan Pengkodean

Tahap ini merupakan realisasi dari desain. Kegiatan meliputi perakitan komponen fisik dispenser, penulisan program kontrol untuk *ESP32* (membaca sensor, mengaktifkan *relay*), dan pembuatan kode aplikasi *CI4* (integrasi *webhook Xendit*, manajemen data, serta fitur notifikasi pesanan masuk secara *real-time* pada halaman *admin*).

d) Pengujian sistem

Setelah *prototype* terakit, dilakukan pengujian untuk memastikan *prototype* berfungsi tanpa error dan sesuai dengan rancangan awal. Pengujian mencakup

kalibrasi sensor air (untuk mencari persentase error takaran), uji coba transaksi *QRIS* jarak jauh, dan *Blackbox* testing fungsi (baik dari sisi pelanggan maupun *admin*).

e) Evaluasi dan Demonstrasi *Prototype*

Mengingat batasan ruang lingkup penelitian, tahap akhir pada pengembangan sistem ini tidak melibatkan penerapan atau pemeliharaan alat secara langsung di Depot Air Minum Bluebub. Tahap ini difokuskan pada demonstrasi fungsionalitas dan evaluasi *prototype Smart* Dispenser dalam lingkungan uji coba yang terkendali. Sistem dievaluasi berdasarkan tingkat keberhasilannya dalam menjawab rumusan masalah, yaitu kemampuan menjalankan alur pemesanan *online* dari jarak jauh, menampilkan notifikasi pesanan di *dashboard admin*, memproses pembayaran *QRIS* melalui *webhook Xendit* secara *real-time*, dan menakar keluaran air secara presisi sebelum galon diantarkan oleh petugas. Seluruh hasil evaluasi *prototype* ini kemudian didokumentasikan dan disusun ke dalam laporan penelitian skripsi.

3.4 Instrumen Penelitian

Adapun instrumen yang peneliti gunakan pada penelitian ini, akan dirincikan sebagai berikut:

a) Observasi

Instrumen observasi diterapkan dengan cara melakukan pengamatan langsung ke lokasi Depot Air Minum Bluebub. Proses observasi ini dicatat dalam bentuk lembar catatan lapangan yang berfokus pada kondisi fisik dan infrastruktur depot. Pengamatan dilakukan untuk mengidentifikasi ketersediaan jaringan internet yang stabil sebagai syarat utama operasional mikrokontroler *ESP32*, serta memetakan

tata letak area pengisian air untuk menentukan dimensi *prototype* alat. Selain itu, observasi juga difokuskan pada pengamatan alur kerja petugas dalam memproses pengisian air, ketersediaan ruang atau area penyimpanan untuk mendukung sistem pertukaran galon (*swap system*), hingga spesifikasi kelistrikan di lokasi guna memastikan *prototype Smart* Dispenser dapat dirancang secara proporsional dan terintegrasi dengan baik di lokasi aslinya.

b) Wawancara

Proses wawancara dilakukan secara langsung dengan pemilik sekaligus pengelola Depot Air Minum Bluebub menggunakan pedoman wawancara semi-terstruktur. Instrumen ini difokuskan untuk menggali informasi kualitatif secara mendalam terkait kendala operasional yang selama ini terjadi pada sistem pembayaran tunai, seperti risiko penerimaan uang palsu, kesulitan menyediakan uang kembalian, hingga masalah selisih pencatatan pembukuan manual. Melalui wawancara ini, peneliti juga menyelaraskan spesifikasi kebutuhan sistem dengan harapan pengelola depot, seperti penentuan takaran pasti volume air yang akan dijual, fitur pelaporan transaksi digital (*dashboard backend*) yang dibutuhkan untuk pembukuan, serta tanggapan dan kelayakan dari sisi pengelola terkait penerapan model bisnis pesan-antar jarak jauh di mana pelanggan memesan via *website* dan petugas melakukan pengantaran galon.

3.5 Analisis Kebutuhan

Berikut ini adalah kebutuhan yang peneliti gunakan untuk penelitian, sebagai berikut:

a) Perangkat Keras

Perangkat keras yang peneliti gunakan sebagai berikut:

Table 3.1 Kebutuhan Perangkat Keras

No	Jenis Perangkat Keras	Keterangan	Harga (Satuan)
1.	Mikrokontroler	*NodeMCU ESP32*	Rp. 75.500
2.	Sensor Debit Air	*Water Flow YF-S401*	Rp. 33.000
3.	Sensor Jarak	*Infrared Proximity E18-D80NK*	Rp. 23.500
4.	Modul Saklar	*Relay 2-Channel* 5V	Rp. 16.000
5.	Pompa Air	*Water Pump DC* 12V	Rp. 23.000
6.	Modul Penampil	*LCD* 16x2 dengan Modul I2C	Rp. 23.500
7.	Catu Daya	*Adaptor* 12V 2A	Rp. 12.500
Total			Rp. 207.000

b) Perangkat Lunak

Perangkat lunak yang peneliti gunakan sebagai berikut:

Table 3.2 Kebutuhan Perangkat Lunak

No	Jenis Perangkat Lunak	Keterangan
1.	Sistem Operasi	*Windows 11*
2.	*Text Editor*	*Visual Studio Code*
3.	*Framework*	*Codeigniter 4*
4.	*Database*	*MySQL* (XAMPP)
5.	Lingkungan Pengembangan IoT	Arduino IDE
6.	*Payment Gateway API*	*Xendit API* & *webhook*
7.	Layanan *Tunneling* Jaringan	*Ngrok*
8.	*Browser*	*Google Chrome*

3.5.1 Analisis Data

Analisis data pada penelitian ini dilakukan untuk mengevaluasi data hasil investigasi kebutuhan awal serta mengukur tingkat keberhasilan fungsional *prototype Smart* Dispenser dan sistem pemesanan jarak jauh yang dikembangkan. Proses analisis data dalam penelitian ini dibagi menjadi tiga pendekatan utama, yaitu:

a) Analisis Kebutuhan Data Sistem

Data kualitatif yang diperoleh dari hasil wawancara dengan pemilik depot dan observasi lapangan dianalisis secara deskriptif kualitatif. Hasil analisis ini digunakan untuk merumuskan landasan peralihan model bisnis dari operasional manual menjadi sistem pesan-antar mandiri (*Online-to-Offline*). Data ini juga menjadi acuan untuk menentukan spesifikasi *hardware* yang tepat serta merancang *software* berbasis *website* yang mampu menerima pesanan jarak jauh dan memberikan notifikasi admin.

b) Analisis Data Pengujian Perangkat Keras

Data kuantitatif dikumpulkan dari tahap pengujian fungsionalitas perangkat keras, khususnya pada tingkat akurasi pembacaan volume air oleh sensor *water flow* (*YF-S401*). Volume air yang dikalkulasi oleh mikrokontroler *ESP32* akan dianalisis dengan cara membandingkannya terhadap volume air aktual yang diukur secara manual menggunakan gelas ukur standar. Tingkat akurasi sistem penakaran dihitung menggunakan rumus persentase kesalahan (*error percentage*) sebagai berikut:

$$\text{Presentase Error} = \frac{\text{Volume Aktual} - \text{Volume Sensor}}{\text{Volume Aktual}} \text{ x } 100\%$$

1. Volume Aktual: Volume air nyata yang tertampung di dalam gelas ukur (mL).
2. Volume Sensor: Volume air yang tercatat pada sistem pembacaan *ESP32* (mL).

Analisis ini bertujuan untuk membuktikan bahwa *prototype* mampu menakar air dengan presisi sebelum galon diantarkan oleh petugas ke rumah pelanggan.

c) Analisis Data Pengujian Perangkat Lunak dan Integrasi

Data hasil pengujian sistem *website* dan integrasi *Internet of Things* (*IoT*) dianalisis menggunakan metode *Blackbox Testing*. Analisis dilakukan dengan membandingkan kesesuaian antara *Expected Result* (hasil yang dirancang) dengan *Actual Result* (hasil aktual yang berjalan pada sistem). Indikator utama yang dianalisis meliputi: kelancaran akses tautan publik *Ngrok* oleh pelanggan dari jarak jauh, kecepatan munculnya notifikasi pesanan di *dashboard admin* setelah *webhook Xendit* diterima, serta ketepatan instruksi server kepada perangkat keras untuk mengeksekusi pengisian air.

3.5.2 Analisis Proses

Analisis proses menguraikan bagaimana alur kerja dari sistem pemesanan *online* dan otomasi *Smart* Dispenser ini berjalan, mulai dari pemesanan oleh pelanggan hingga pemenuhan pesanan oleh petugas. Proses operasional pada sistem yang diusulkan ini dibagi menjadi tiga tahapan proses utama, yaitu:

a) Proses Pemesanan dan Pembayaran

Pelanggan mengakses *website* aplikasi melalui *browser smartphone* mereka menggunakan tautan publik yang dihasilkan oleh *Ngrok*. Pelanggan memilih menu pengisian air dan melakukan pembayaran secara non-tunai melalui *scan* kode *QRIS (Simulation Pay)*. Proses validasi transaksi dilakukan sepenuhnya oleh sistem *payment gateway Xendit*.

b) Proses Verifikasi dan Notifikasi

Setelah pembayaran dinyatakan berhasil oleh *Xendit*, server lokal (*CodeIgniter 4*) menerima notifikasi *webhook*. *Server* secara otomatis menyimpan data transaksi ke dalam *database MySQL*, memperbarui status pesanan menjadi "Lunas", dan seketika memunculkan notifikasi *real-time* pada halaman *dashboard admin* yang sedang dipantau oleh petugas depot.

c) Proses Pengisian dan Pemenuhan Pesanan

Bersamaan dengan notifikasi di *admin*, server mengirimkan sinyal perintah ke mikrokontroler *ESP32* untuk bersiap. Petugas meletakkan galon di alat . Sensor *infrared* (*E18-D80NK*) membaca keberadaan galon, kemudian *ESP32* memicu *relay* untuk menyalakan pompa air. Sensor *water flow* (*YF-S401*) menakar volume air secara presisi hingga penuh, lalu pompa otomatis mati. Setelah itu, petugas mengambil galon yang sudah terisi dan mengantarkannya ke rumah pelanggan (sistem tukar galon).

3.5.3 Analisis Kelemahan

Analisis kelemahan dilakukan untuk mengidentifikasi berbagai kendala dan kekurangan pada sistem operasional konvensional yang saat ini berjalan di Depot Air Minum Bluebub. Kelemahan-kelemahan ini menjadi landasan utama mengapa *prototype Smart* Dispenser bersistem *Online-to-Offline* (O2O) ini perlu dikembangkan. Beberapa kelemahan pada sistem lama meliputi:

a) Kelemahan pada Proses Transaksi

Sistem pembayaran yang masih menggunakan uang tunai sering kali menimbulkan kendala bagi petugas, seperti kesulitan mencari uang kembalian, tingginya risiko penerimaan uang palsu atau lecek, serta rawan terjadi selisih (kesalahan hitung) saat pencatatan pembukuan manual di akhir hari operasional.

b) Kelemahan pada Proses Pelayanan

Pelanggan diwajibkan untuk datang langsung ke lokasi depot membawa galon kosong, lalu menunggu proses antrean dan pengisian selesai. Hal ini dinilai kurang praktis dan memakan waktu bagi pelanggan yang memiliki mobilitas tinggi, sehingga membatasi jangkauan penjualan depot.

c) Kelemahan pada Proses Pengisian Air

Proses pengisian air ke dalam galon masih dilakukan secara manual dengan menekan atau memutar keran. Hal ini mengharuskan petugas untuk terus berjaga dan mengawasi takaran air. Kelalaian manusia (*human error*), seperti terlambat mematikan keran, sering kali menyebabkan air luber terbuang sia-sia dan volume air yang diberikan kepada pelanggan menjadi tidak konsisten.

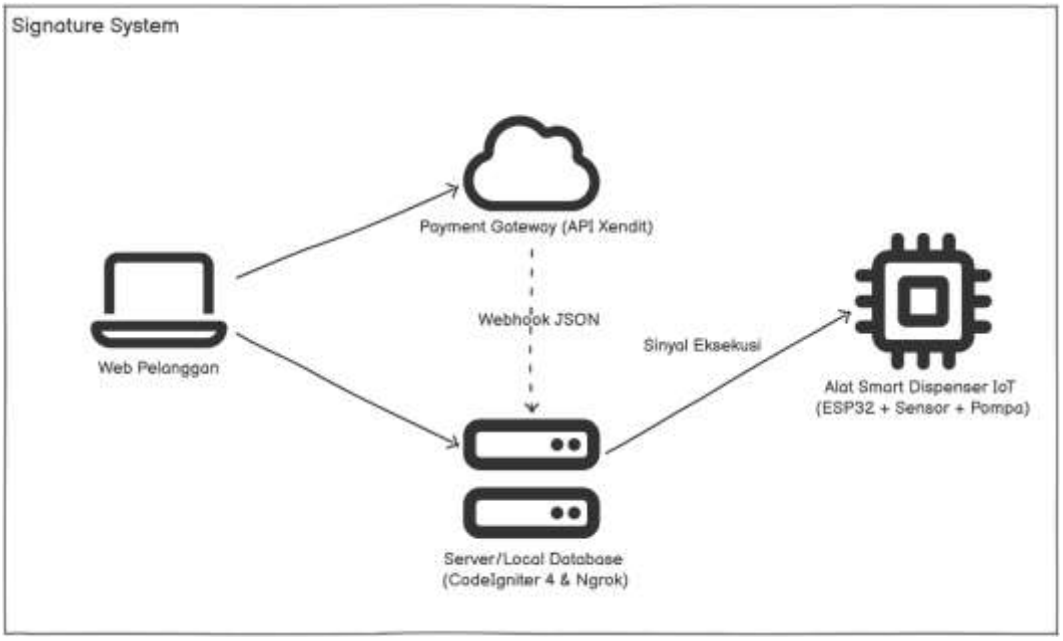
d) Kelemahan pada Pencatatan Data

Belum adanya sistem *database* yang terintegrasi membuat pemilik depot kesulitan untuk memonitor riwayat transaksi, menghitung total pendapatan harian secara *real-time*, dan mengaudit kesesuaian antara jumlah air yang terjual dengan uang yang masuk.

3.6 Desain

Tahap desain sistem merupakan kelanjutan dari tahapan analisis kebutuhan. Pada tahap ini, peneliti menerjemahkan spesifikasi kebutuhan dan model operasional yang telah dirumuskan ke dalam bentuk cetak biru (*blueprint*). Sebelum merincikan desain ke dalam tahapan proses dan perangkat, rancang bangun sistem ini secara keseluruhan dipetakan melalui Arsitektur Sistem (*Signature System*).

Signature System ini menjabarkan keterhubungan teknis antara ekosistem transaksi digital pelanggan dengan perangkat keras di lokasi depot (toko), yang dapat divisualisasikan pada Gambar 3.2 berikut:



Gambar 3.2 *Signature System*

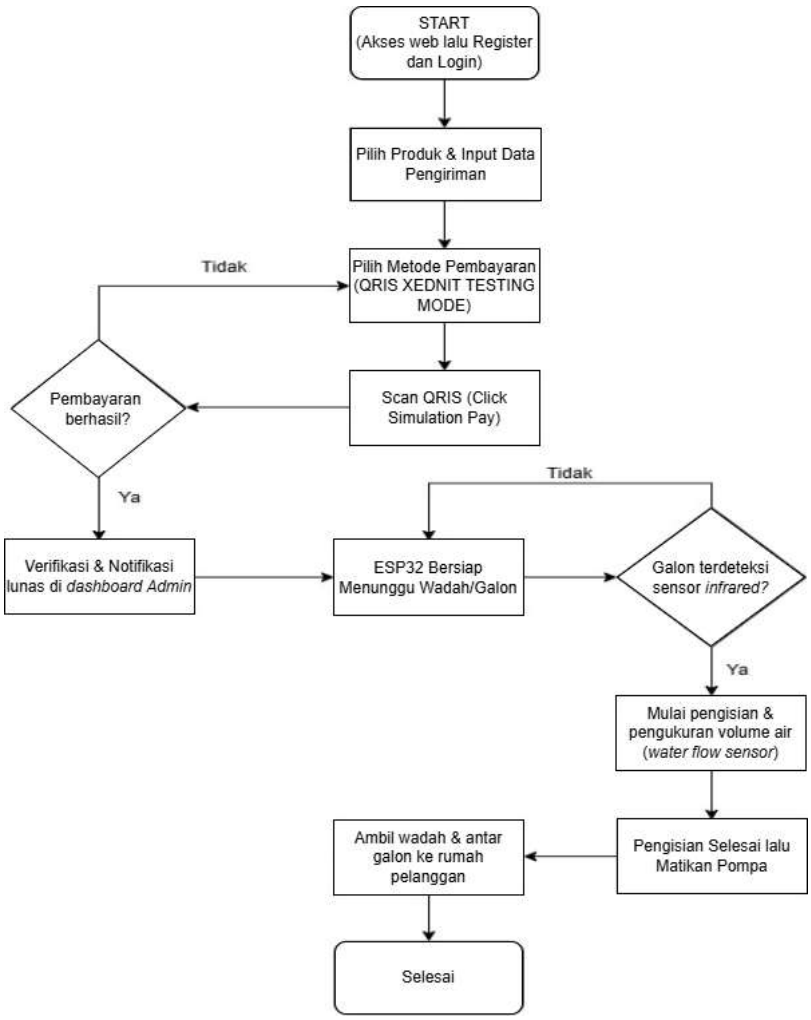
Berdasarkan Gambar 3.2 di atas, arsitektur sistem dibagi menjadi empat blok utama. Alur diawali dari sisi pelanggan yang melakukan *register* lalu *login* kemudian melakukan pemesanan dan pembayaran melalaui *website*. Transaksi tersebut kemudian diteruskan ke *Payment Gateway* (*API Xendit*) untuk proses

otorisasi. Setelah pembayaran dinyatakan valid, server awan *Xendit* mengirimkan sinyal *Webhook JSON* secara real-time ke Server Lokal/*Database* (*CodeIgniter 4* yang dijembatani *Ngrok*). Terakhir, server bertindak sebagai pusat komando yang mengirimkan sinyal eksekusi nirkabel kepada Alat Smart Dispenser (*ESP32*) di toko untuk menyalakan pompa air secara fisik.

Penjabaran arsitektur utama di atas kemudian diuraikan lebih rinci ke dalam dua bagian desain utama, yaitu desain proses operasional dan desain perangkat.

3.6.1 Desain Proses

Desain proses merupakan tahapan esensial untuk memetakan bagaimana seluruh komponen dalam ekosistem bisnis *Online-to-Offline* (O2O) ini saling berinteraksi secara dinamis. Pada bagian ini, alur kerja sistem diuraikan secara logis dan berurutan, mulai dari sisi pengguna (saat pelanggan mengakses *website*), proses validasi transaksi oleh pihak ketiga (*Payment Gateway Xendit*), hingga respons akhir berupa eksekusi fisik oleh mesin di lokasi depot. Untuk memberikan gambaran teknis yang komprehensif, rancangan alur operasional utama direpresentasikan melalui bagan alir (*flowchart*). Secara lebih rinci, alur logika operasional tersebut dapat dilihat pada Gambar 3.3 berikut:



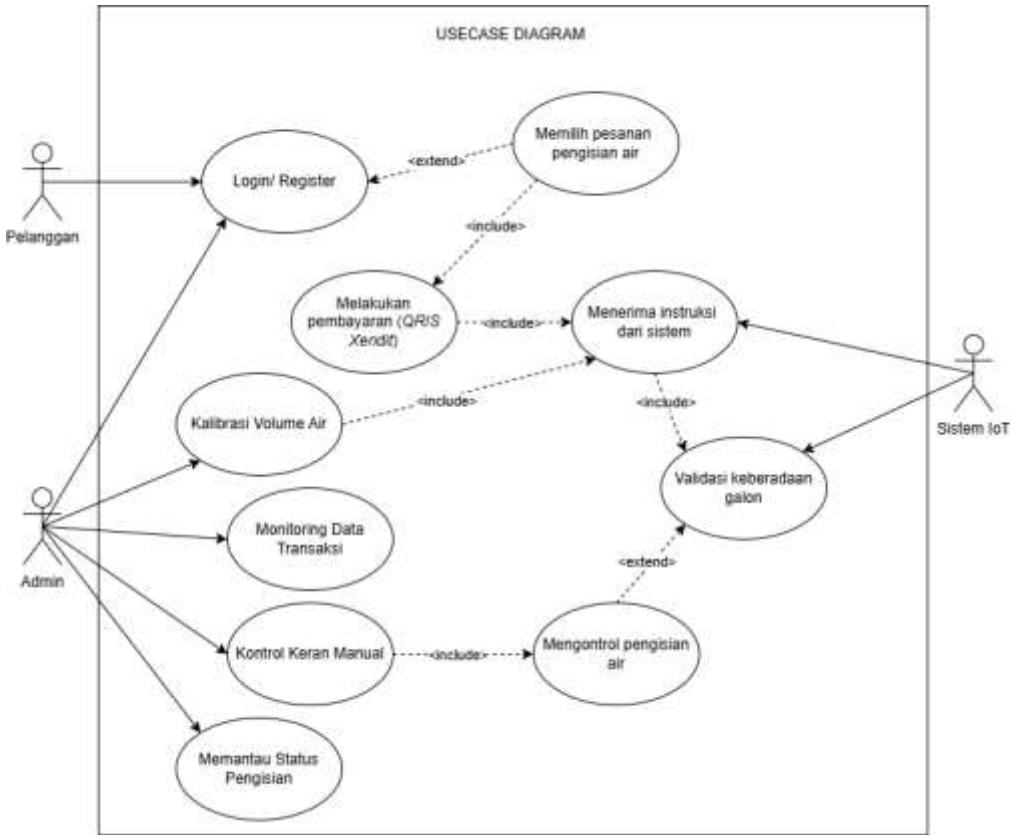
Gambar 3.3 *Flowchart* Sistem *Smart* Dispenser

Gambar 3.3 Proses dimulai saat pelanggan mengakses *website* melalui *smartphone*, lalu melakukan *register* dan *login* untuk memilih pesanan lalu memasukkan detail pesanan (nama, alamat, dan jumlah galon) serta memilih metode pembayaran *QRIS* via *Xendit* (*Simulation Pay*). Setelah pelanggan melakukan pembayaran, sistem secara otomatis memverifikasi transaksi.

Jika pembayaran berhasil, sistem akan memperbarui status pesanan menjadi lunas di *database* dan seketika memunculkan notifikasi pada *dashboard admin*. Bersamaan dengan itu, server mengirimkan instruksi ke mikrokontroler *ESP32*

untuk bersiap (*standby*). Petugas yang melihat notifikasi akan meletakkan galon kosong di area pengisian. Sensor *infrared* kemudian akan mendeteksi galon tersebut. Jika galon terdeteksi, pompa air akan menyala dan sensor *water flow* akan menakar air secara presisi hingga volume yang ditentukan tercapai. Setelah pengisian selesai, petugas akan mengambil galon tersebut dan mengantarkannya ke rumah pelanggan (sistem *swap*), menandai proses pesanan telah selesai.

Untuk menggambarkan interaksi antara pengguna (aktor) dengan sistem yang dibangun, digunakan *Use Case Diagram*. Diagram ini memetakan peran pelanggan, *admin*, dan sistem dalam serangkaian proses operasional, mulai dari pemesanan hingga penakaran air. Peta interaksi tersebut ditunjukkan pada Gambar 3.4 berikut:



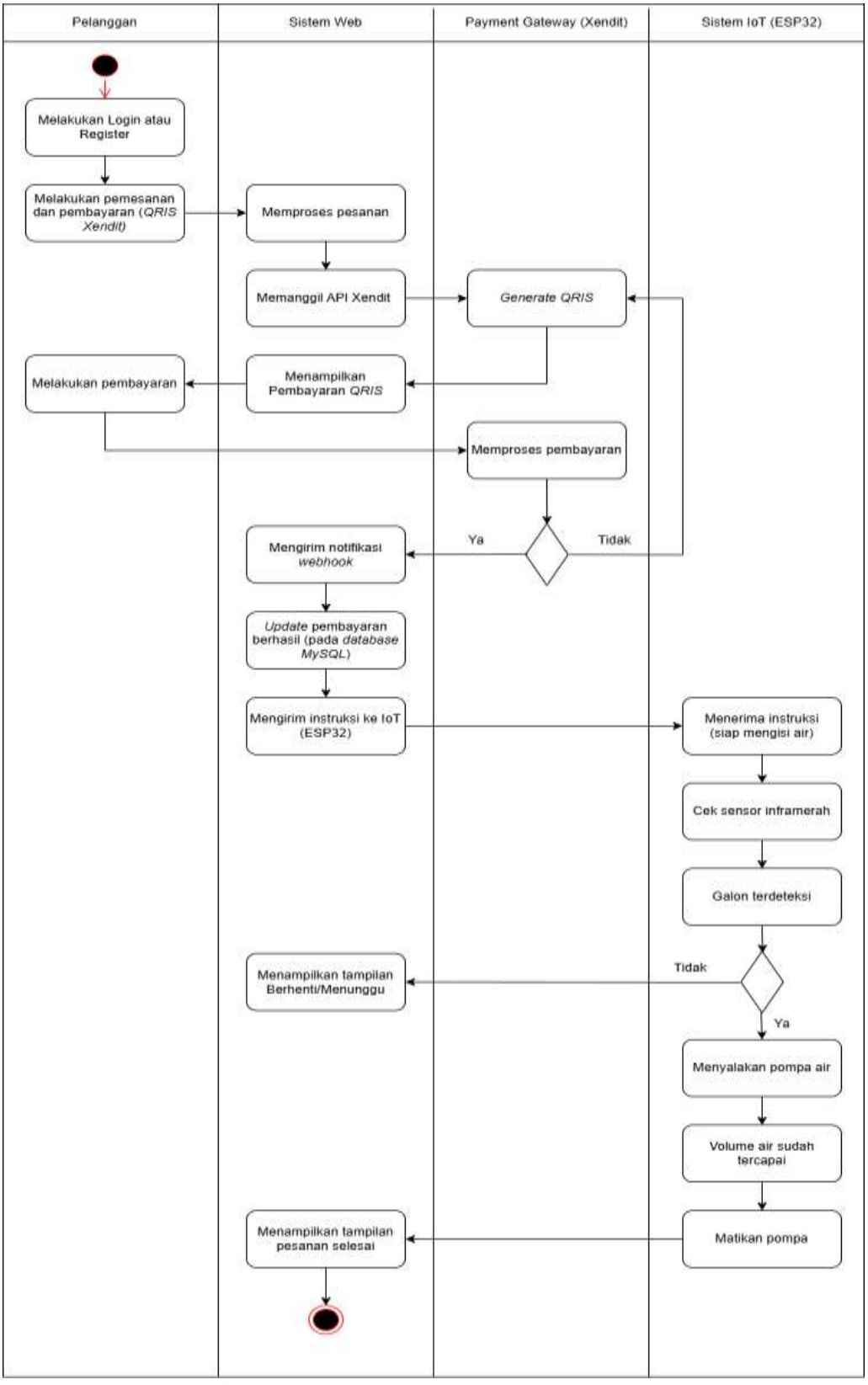
Gambar 3.4 *Usecase* Diagram Sistem *Smart* Dispenser

Berdasarkan Gambar 3.4, terdapat tiga aktor yang terlibat dalam sistem, yaitu pelanggan, *admin*, dan sistem *IoT*. Pelanggan dapat melakukan *login* atau registrasi, memilih pesanan air, mengisi data pengantaran, dan melakukan pembayaran menggunakan *QRIS Xendit*. Setelah pembayaran berhasil, sistem akan mengirimkan instruksi pengisian kepada perangkat *IoT*.

Admin berfungsi untuk melakukan pengawasan dan pengendalian sistem, seperti memantau status pengisian, melakukan monitoring data transaksi, mengontrol keran secara manual, serta melakukan kalibrasi volume air. Sementara itu, sistem *IoT* bertugas menerima instruksi pengisian, memvalidasi keberadaan galon menggunakan *sensor proximity*, dan menjalankan proses pengisian air sesuai data yang diterima dari *website*.

Relasi antar *use case* menggambarkan alur proses yang terjadi mulai dari pelanggan melakukan pemesanan, pembayaran, hingga sistem *IoT* menjalankan proses pengisian air secara otomatis berdasarkan data transaksi yang telah diterima.

Selain *Use Case*, alur kerja operasional sistem juga digambarkan lebih detail melalui *Activity* Diagram. Diagram ini memvisualisasikan interaksi berkesinambungan antara empat entitas utama, yaitu Pelanggan, Sistem , *Payment Gateway* (*Xendit*), dan Sistem (*ESP32*). Rincian aktivitas antar entitas tersebut dapat dilihat pada Gambar 3.5 berikut:



Gambar 3.5 *Activity* Diagram Proses Pemesanan

Gambar 3.5 merupakan alur kerja operasional sistem digambarkan melalui interaksi berkesinambungan antara empat entitas utama, yaitu Pelanggan, Sistem , *Payment Gateway* (*Xendit*), dan Sistem (*ESP32*). Alur ini diawali oleh pelanggan yang mengakses *website* lalu melakukan *register* dan *login* untuk melakukan pemesanan air sekaligus memilih metode pembayaran via *QRIS* (*Simulation Pay*). Data pesanan tersebut diproses oleh sistem, yang kemudian secara otomatis memanggil *API Xendit* untuk menghasilkan (*generate*) kode *QRIS*. Setelah kode tersebut ditampilkan pada layar perangkat pelanggan, tahapan dilanjutkan dengan proses pemindaian (*scan*) dan pembayaran oleh pelanggan. Tapi karena *Xendit* yang digunakan dalam penelitian ini dalam testing mode, maka dilakukan simulasi pembayaran saja, mode ini digunakan *Xendit* sama seperti akun *real* yang sudah terdaftar untuk usaha.

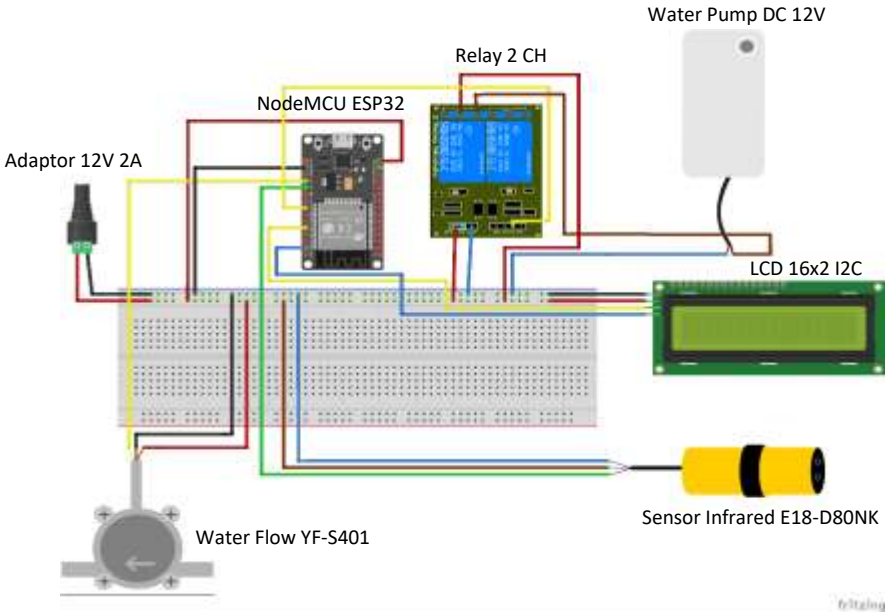
Setelah proses pembayaran dilakukan, *Xendit* bertugas memverifikasi status transaksi secara langsung. Pada tahap ini, sistem dihadapkan pada percabangan logika, apabila pembayaran terdeteksi gagal atau tidak valid, sistem akan menahan proses dan menunggu hingga pembayaran yang benar dilakukan. Sebaliknya, apabila pembayaran berhasil divalidasi, *Xendit* akan seketika mengirimkan notifikasi *webhook* kepada Sistem . Merespons notifikasi tersebut, Sistem secara otomatis memperbarui status transaksi menjadi berhasil di dalam *database MySQL*, lalu meneruskan instruksi eksekusi pengisian air kepada perangkat keras (*ESP32*) yang berada di lokasi mesin.

Memasuki fase eksekusi fisik, mikrokontroler *ESP32* yang menerima instruksi akan beralih ke mode siap beroperasi. Sebelum mengaktifkan aktuator,

sistem terlebih dahulu memvalidasi keamanan fisik dengan memeriksa sensor inframerah. Jika wadah galon belum terdeteksi, sistem akan mengirimkan status umpan balik untuk berhenti atau menunggu hingga galon diletakkan. Namun, jika galon telah terdeteksi pada posisinya, *ESP32* akan segera menyalakan pompa air. Selama proses pemompaan berlangsung, *water flow* sensor terus menakar debit air hingga volume keluaran mencapai target pesanan pelanggan. Begitu takaran terpenuhi, mikrokontroler mematikan pompa dan Sistem memperbarui antarmuka pengguna dengan status pesanan selesai, yang sekaligus mengakhiri seluruh rangkaian operasional alat.

3.6.2 Desain Perangkat

Bagian ini menguraikan perancangan fisik dan logika operasional dari *prototype Smart* Dispenser. Penjelasan mencakup perangkat keras yang digunakan, struktur diagram blok sistem, prinsip kerja utama, hingga alur kerja alat secara terperinci.



Gambar 3.6 Desain Perangkat

a) Desain *hardware*

Perancangan perangkat keras pada *prototype Smart* Dispenser ini dibagi menjadi tiga kategori utama berdasarkan alur kerja sistemnya, yaitu pengendali, perangkat masukan, dan perangkat keluaran:

1. *Controller*

   Mikrokontroler *ESP32*, bertindak sebagai pusat kendali atau "otak" utama dari seluruh sistem. Komponen ini bertugas memproses instruksi logika dan menyediakan konektivitas *Wi-Fi* terintegrasi agar mesin dapat berkomunikasi secara *real-time* dengan server *website* dan *payment gateway* jarak jauh.

2. Perangkat *input*

   a. Sensor Jarak Inframerah (*E18-D80NK*), berfungsi sebagai sistem keamanan dan validasi fisik. Sensor ini mendeteksi keberadaan galon di area pengisian, memastikan mesin hanya memompa air ketika wadah telah tersedia untuk mencegah tumpahan.

   b. *Water Flow* Sensor, berfungsi sebagai alat ukur debit cairan. Sensor ini mengirimkan data volume air yang mengalir ke mikrokontroler, memastikan takaran air yang didistribusikan presisi dan sesuai dengan pesanan pelanggan.

3. Perangkat *output*

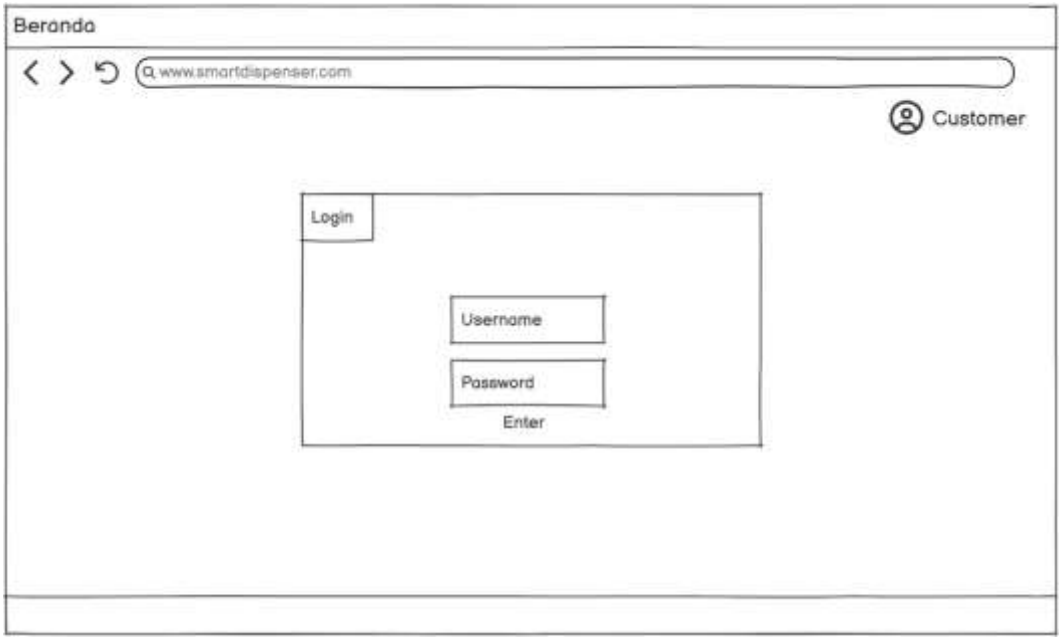
   a. Modul *Relay* 2 CH, berfungsi sebagai sakelar elektrik penghubung. Komponen ini menerima sinyal logika daya rendah dari *ESP32* untuk memutus atau mengalirkan arus listrik bertegangan lebih tinggi menuju pompa air.

b. *Water Pump DC* (Pompa Air), bertindak sebagai aktuator mekanis eksekutor. Pompa ini bertugas memindahkan air dari tangki penampungan utama ke dalam galon pelanggan berdasarkan instruksi dari mikrokontroler.
c. *LCD* 16x2 I2C, berfungsi sebagai antarmuka mesin dan manusia (*Human Machine Interface*). Layar ini menampilkan status operasional alat secara visual kepada pengguna di lokasi, seperti status siaga (*standby*), verifikasi transaksi, hingga proses pengisian air.

b) Desain antarmuka *website*

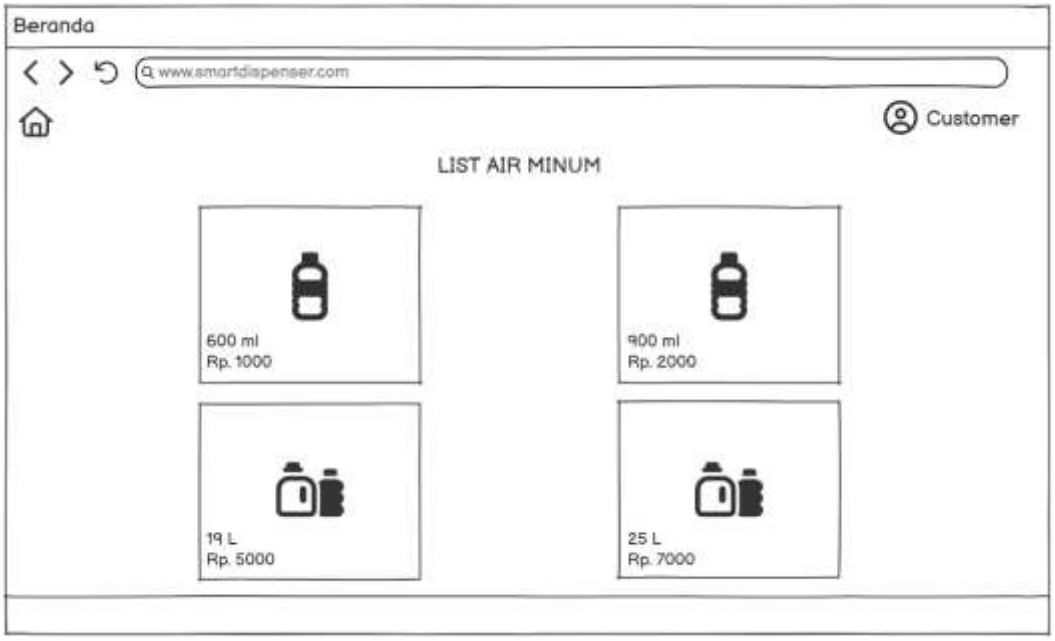
Desain antarmuka (*website*) dirancang sebagai media interaksi utama antara pelanggan dan sistem operasional depot. Perancangan ini berfokus pada kemudahan navigasi (*User Experience*) guna memastikan proses pemesanan dan pembayaran *QRIS* dapat dipahami dengan cepat oleh masyarakat luas. Berikut adalah penjabaran desain dari masing-masing halaman antarmuka sistem:

Halaman pertama pada antarmuka *website* adalah halaman *Login*. Halaman ini berfungsi sebagai pintu akses autentikasi utama bagi pelanggan. Antarmuka ini memuat form input *Username* dan *Password*, serta menyediakan tautan *Register* di bagian bawah bagi pengguna yang belum memiliki akun. Desain tata letak halaman *Login* dapat dilihat pada Gambar 3.7 berikut:



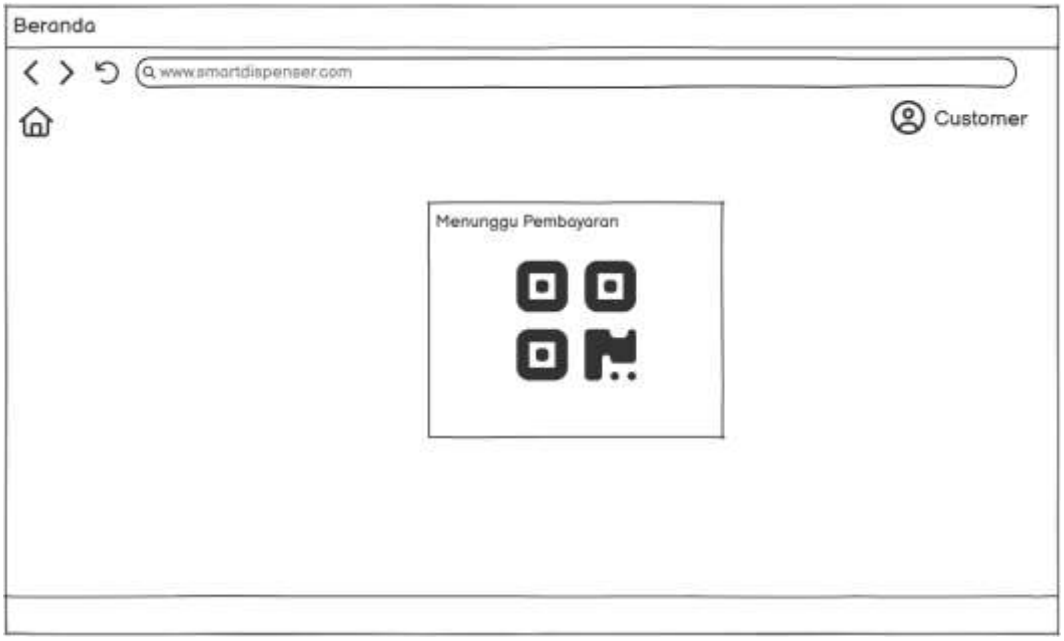
Gambar 3.7 Halaman *Login*

Setelah berhasil melakukan l*ogin*, pelanggan akan diarahkan ke halaman Pemesanan Air Minum. Halaman ini berfungsi sebagai katalog utama tempat pelanggan dapat memilih varian takaran air yang ingin dipesan. Tampilan dirancang menyajikan opsi produk secara terstruktur dalam bentuk grid, lengkap dengan ikon visual kemasan, keterangan volume air, beserta rincian harga untuk masing-masing pilihan. Desain halaman pemesanan tersebut diilustrasikan pada Gambar 3.8 berikut:



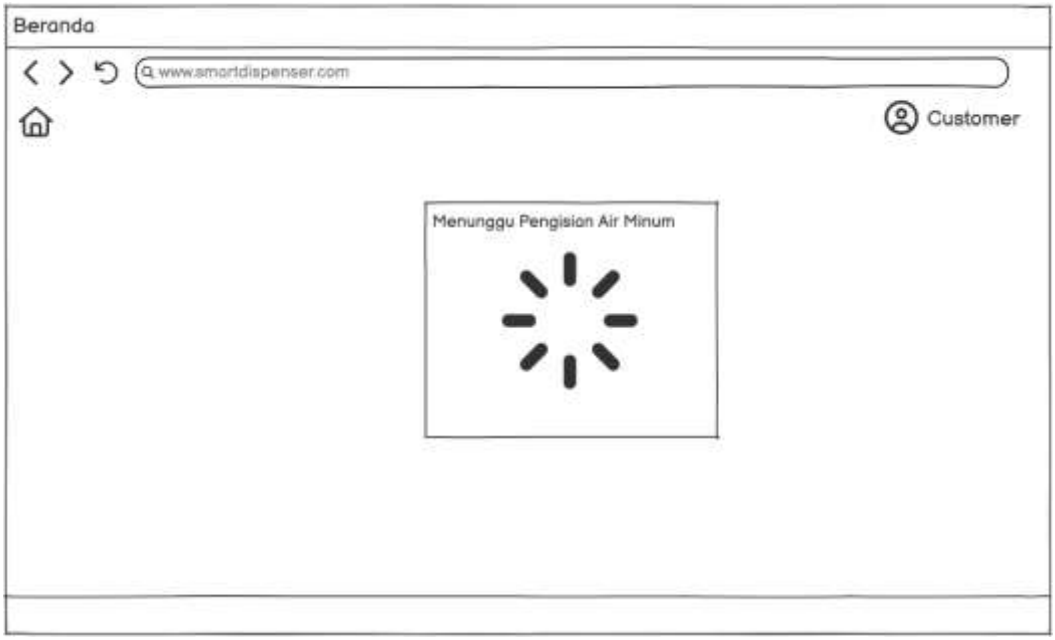
Gambar 3.8 Pemesanan Air

Apabila pelanggan telah memilih produk dan mengonfirmasi pesanan, sistem akan mengarahkan ke halaman *Scan* Pembayaran. Halaman ini bertugas menampilkan kode *QRIS* yang di-*generate* melalui *API Xendit* secara terpusat di layar perangkat pelanggan. Terdapat pula status indikator "Menunggu Pembayaran" yang menandakan bahwa sistem sedang bersiap memverifikasi transaksi sebelum melanjutkan ke proses pengisian air. Tampilan halaman pembayaran ini dapat dilihat pada Gambar 3.9 berikut:



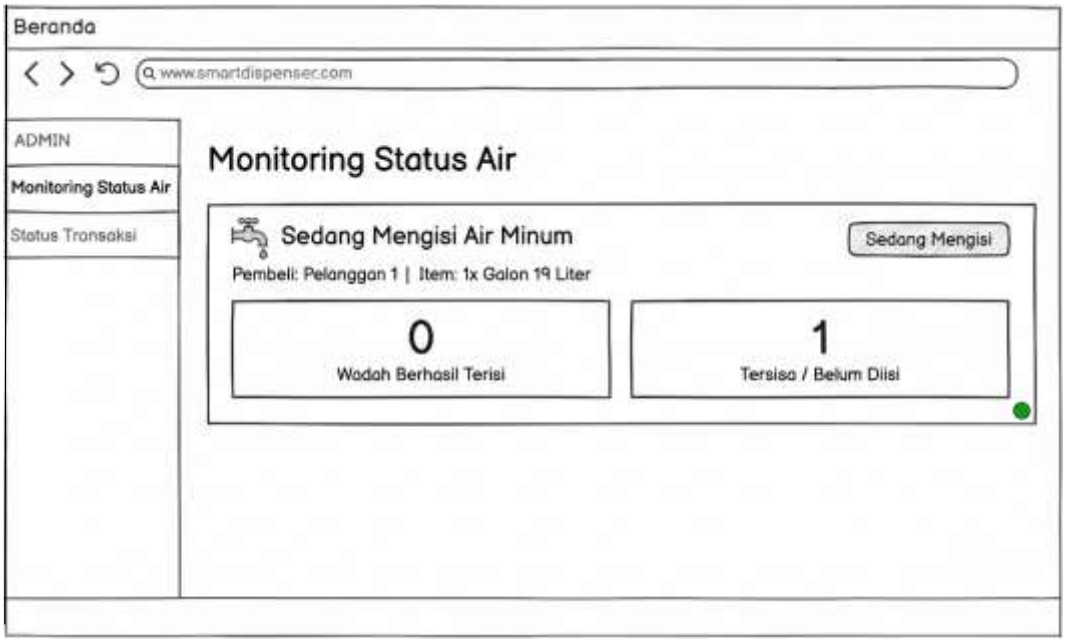
Gambar 3.9 Pembayaran *Scan QRIS*

Setelah transaksi divalidasi berhasil oleh *Payment Gateway*, halaman akan otomatis beralih memunculkan status "Menunggu Pengisian Air Minum". Tampilan ini berfungsi sebagai indikator visual *real-time* yang menginformasikan kepada pelanggan bahwa pesanan telah lunas dan sistem tengah menunggu perangkat keras (dispenser) mengeksekusi pengisian air secara fisik. Desain halaman status pengisian ini direpresentasikan pada Gambar 3.10 berikut:



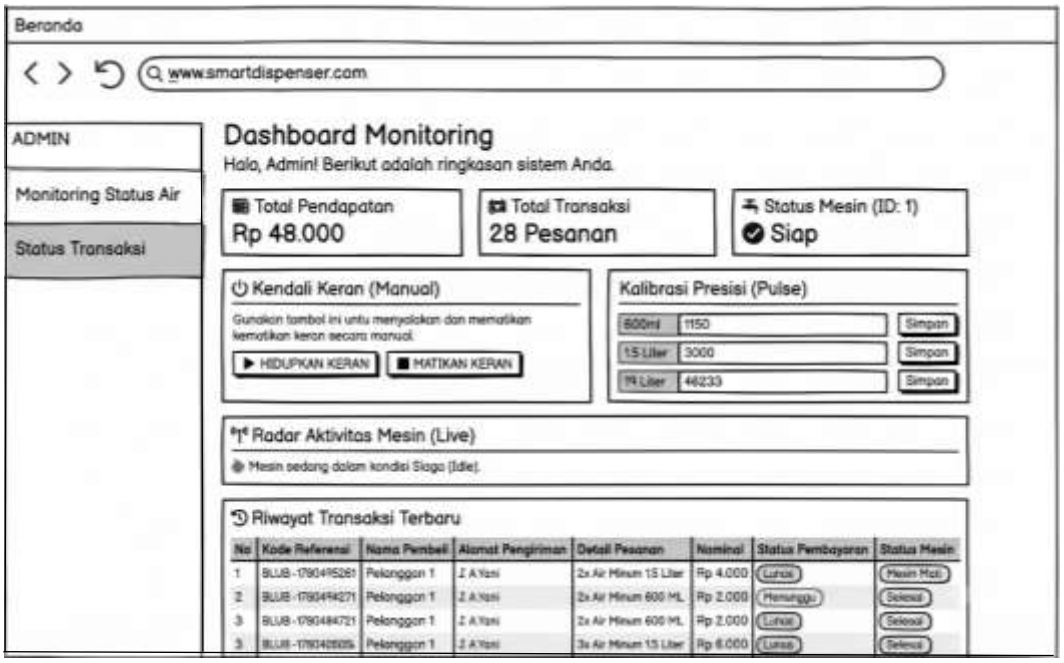
Gambar 3.10 Halaman Menunggu Pengisian

Beralih ke sisi pengelola yaitu petugas depot, antarmuka *Dashboard Administrator* dirancang dengan format tata letak desktop untuk memudahkan pemantauan. Menu utama pada *dashboard* ini adalah "Monitoring Status Air", yang memuat panel indikator untuk memantau status operasional mesin secara *real-time*. Halaman ini juga dilengkapi panel navigasi samping (*sidebar*) untuk berpindah antar menu pengelolaan. Desain *wireframe* halaman monitoring tersebut dapat dilihat pada Gambar 3.11 berikut:



Gambar 3.11 Halaman Monitoring *Admin*

Menu kedua pada *Dashboard Administrator* adalah halaman "Status Transaksi". Halaman ini berfungsi sebagai pusat informasi bagi *admin* untuk memantau riwayat pesanan yang masuk ke dalam sistem. Pada area konten utama, disajikan daftar transaksi pelanggan secara terstruktur, yang menampilkan informasi ringkas mengenai detail pesanan beserta status keberhasilan transaksinya dari sistem *Xendit*. Desain halaman riwayat transaksi ini ditunjukkan pada Gambar 3.12 berikut:



Gambar 3.12 Halaman Transaksi

c) Diagram Blok

Secara struktural, diagram blok *prototype* ini mengintegrasikan komponen fisik alat dengan ekosistem *Online-to-Offline* (pesan-antar). Blok masukan (*input*) mengirimkan data kondisi fisik (keberadaan galon dan debit air) ke blok kendali (*controller*) yaitu *ESP32*. Mikrokontroler ini berperan sebagai jembatan nirkabel yang terhubung langsung dengan server *website* dan *payment gateway* (*Xendit*). Ketika *ESP32* menerima validasi transaksi jarak jauh dari server, instruksi tersebut diteruskan ke blok keluaran (*output*) untuk mengeksekusi aktuator mekanis (pompa air) dan memperbarui tampilan informasi pada antarmuka layar.

d) Prinsip kerja alat

Prinsip kerja mekanis ini berpusat pada sistem kendali fluida terotomatisasi berbasis *Internet of Things* (*IoT*). Alat ini dirancang untuk mengubah data digital (status pembayaran) menjadi tindakan fisik (penuangan air) tanpa intervensi

operator. Mikrokontroler *ESP32* bertindak sebagai pusat komputasi fisik yang terus bersiaga menerima *trigger* (pemicu) dari server jarak jauh.

Dalam operasionalnya, prinsip kerja alat ini sangat mengedepankan protokol keamanan (*safety protocol*) dan presisi. Sebelum aktuator bekerja, sistem mewajibkan validasi fisik di mana sensor inframerah (*E18-D80NK*) harus mendeteksi pantulan objek (galon) dalam jarak dekat. Jika syarat fisik terpenuhi, mikrokontroler akan mengirimkan sinyal daya rendah untuk memicu modul *relay*, yang kemudian menyambungkan arus listrik bertegangan tinggi untuk mengaktifkan *water pump* DC 12V. Selama pemompaan berlangsung, air akan melewati *water flow* sensor (*YF-S401*) yang memutar turbin internal. Berdasarkan prinsip *Hall Effect*, putaran turbin tersebut diubah menjadi pulsa elektrik yang dihitung secara *real-time* oleh *ESP32* untuk menentukan volume presisi. Proses kerja fisik ini akan otomatis berhenti memutus arus listrik seketika setelah takaran air mencapai volume yang dipesan oleh pelanggan.

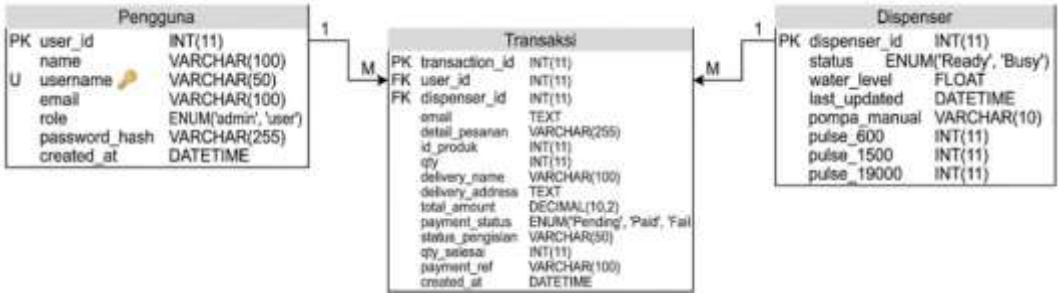
e) Alur kerja alat

1. Mesin menyala, terhubung ke jaringan internet, dan menunggu instruksi pesanan dari sistem *website*.
2. Pembayaran pelanggan di *website* divalidasi oleh *payment gateway*, dan mesin menerima instruksi eksekusi pengisian.
3. Sensor inframerah mendeteksi posisi galon. Jika galon belum ada, sistem menahan pengeluaran air agar tidak tumpah.
4. Mikrokontroler mengaktifkan *relay* untuk menyalakan pompa air. Layar *LCD* menampilkan indikator bahwa pengisian sedang berlangsung.

5. *Water flow* sensor secara *real-time* menghitung volume debit air yang mengalir ke dalam galon.
6. Pompa air otomatis dimatikan sesaat setelah volume air mencapai target pesanan, dan mesin kembali ke status siaga.

3.6.3 Desain Basis Data

Basis data merupakan sekumpulan data yang terorganisir secara sistematis di dalam sistem komputer untuk memudahkan pengelolaan, penyimpanan, dan pemanggilan kembali informasi. Dalam sistem ini, basis data berfungsi sebagai pusat penyimpanan riwayat transaksi, data pelanggan, serta status operasional perangkat secara terpusat. Perancangan struktur data ini direpresentasikan melalui *Entity Relationship Diagram* (ERD) pada Gambar 3.13 berikut:



Gambar 3.13 Desain *Database* Sistem

Guna memperjelas tipe dan fungsi setiap data di atas, maka desain basis data tersebut diuraikan ke dalam tabel kamus data sebagai berikut:

a. Tabel Pengguna

Tabel ini digunakan untuk menyimpan identitas pelanggan yang melakukan akses ke sistem pemesanan.

Table 3.3 Basis data pengguna

Field	*Type*	Keterangan
user_id	INT (PK)	ID unik pengguna
name	VARCHAR	Nama lengkap pelanggan
email	VARCHAR	Alamat untuk login
password_hash	VARCHAR	Kredensial keamanan akun
username	VARCHAR	Untuk melakukan login ke dalam sistem
role	ENUM	Untuk mengetahui role pengguna (*Admin* atau *User*).
created_at	DATETIME	Merekam waktu dan tanggal transaksi dibuat.

b. Tabel Perangkat

Tabel ini mencatat data inventaris fisik dan status operasional mesin .

Table 3.4 Basis data perangkat

Field	*Type*	Keterangan
dispenser_id	INT (PK)	ID untuk mesin dispenser
status	ENUM	Status operasional (*ready/busy*).
water_level	FLOAT	Menyimpan volume sisa air di dalam dispenser yang terbaca oleh sensor.
last_update	DATETIME	Merekaman waktu terakhir lalu *update* ke *database*.
pompa_manual	VARCHAR	Menyalakan pompa manual (*on/off*).
pulse_600	INT	Menyimpan nilai kalibrasi pulse 600mL ke *waterflow*.
pulse_1500	INT	Menyimpan nilai kalibrasi pulse 1500mL ke *waterflow*.
pulse_19000	INT	Menyimpan nilai kalibrasi pulse 19000mL ke *waterflow*.

c. Tabel Transaksi

Tabel ini merekam aktivitas bisnis dan integrasi verifikasi pembayaran.

Table 3.5 Basis data transaksi

Field	*Type*	Keterangan
transaction_id	INT (PK)	ID unik transaksi
user_id	INT (FK)	Relasi ke tabel pengguna
total_amount	DECIMAL	Nominal pembayaran (Rupiah)
payment_status	ENUM	Status dari Xendit (*Pending/Paid*)
payment_ref	VARCHAR	Kode referensi dari *Xendit*
delivery_address	TEXT	Alamat pengiriman
delivery_name	VARCHAR	Nama pemesan
detail_pesanan	VARCHAR	Jumlah pesanan yang dibeli oleh pelanggan
qty	INT	Menyimpan total kuantitas pesanan dalam satu transaksi.
qty_selesai	INT	Menyimpan jumlah galon yang sudah selesai diisi oleh mesin.
created_at	DATETIME	Merekam waktu dan tanggal transaksi dibuat.
email	TEXT	Menyimpan *email* pengguna untuk keperluan l*ogin*.
id_produk	INT	Menyimpan ID produk (*Foreign Key*) dari varian air yang dipesan.
status_pengisian	VARCHAR	Menyimpan status mesin secara *real-time* (seperti: "*Pumping*", "Selesai") untuk layar pemantauan..

BAB IV

HASIL DAN PEMBAHASAN

4.1 Hasil

Bab ini menguraikan hasil akhir dari rancangan dan pembangunan *prototype Smart Dispenser* berbasis *Internet of Things* (*IoT*) menggunakan ESP32, *website* berbasis *CodeIgniter 4*, dan integrasi *payment gateway Xendit*. Hasil yang diperoleh berupa sistem yang mampu menghubungkan proses pemesanan melalui *website* dengan proses pengisian air secara otomatis pada sistem *smart dispenser*.

4.1.1 Implementasi

Tahap implementasi merupakan perwujudan fisik dan digital dari seluruh rancangan sistem yang memadukan perakitan *prototype*, konfigurasi basis data relasional, penyusunan fitur program aplikasi, serta penulisan algoritma kode program.

a) Implementasi Perangkat Keras

Pada tahap ini, implementasi perangkat keras diwujudkan dengan merakit *prototype* di atas papan uji (*breadboard*) untuk diuji apakah sistem layak dan dapat memenuhi tujuan perancangan.

Otak pemrosesan logika dan penyedia konektivitas *Wi-Fi* dikendalikan oleh *NodeMCU ESP32*. Sebagai media interaksi fisik lokal di dekat mesin, dipasang *Modul LCD 16x2 I2C* untuk menampilkan teks informasi status alat secara langsung kepada *user*.

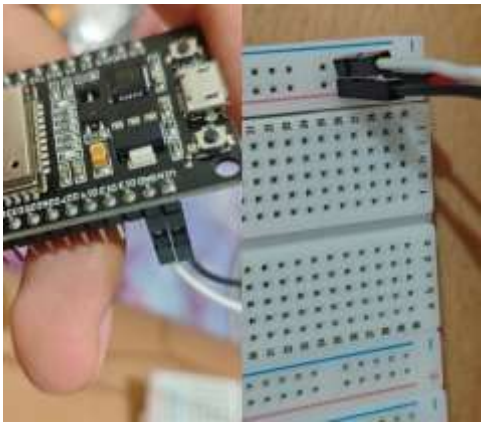
Protokol keamanan fisik dijalankan oleh sensor *Infrared Proximity E18-D80NK* yang mendeteksi keberadaan wadah secara *contactless*. Untuk mengukur volume cairan, sensor *Water Flow YF-S401* dipasang pada jalur selang pengeluaran fluida. Pergerakan pemompaan didorong oleh dinamo pompa air *DC 12V* yang dikendalikan melalui modul *relay 2-channel*. Rangkaian ini disuplai *adaptor 12V 2A* dengan pemisahan jalur tegangan guna menghindari interferensi gelombang elektrik (*noise*).

Untuk memastikan seluruh komponen berjalan dengan baik dan terhindar dari risiko korsleting, proses perakitan kabel (*wiring*) dilakukan secara bertahap dan terstruktur dengan langkah-langkah sebagai berikut:

1. Pendistribusian Daya Listrik

   Langkah pertama adalah mendistribusikan jalur tenaga. Penulis menghubungkan pin tegangan (VIN/5V) dan pin *Ground* (GND) dari ESP32 ke jalur rel positif dan negatif pada *breadboard*. Jalur ini disuplai oleh *adaptor* 12V yang arusnya diturunkan (*step-down*) agar aman bagi sensor.

   Berikut untuk perangkaian kabel yang saling terhubung dari pin VIN dan GND pada *ESP32* ke *breadboard* pada Gambar 4.1.

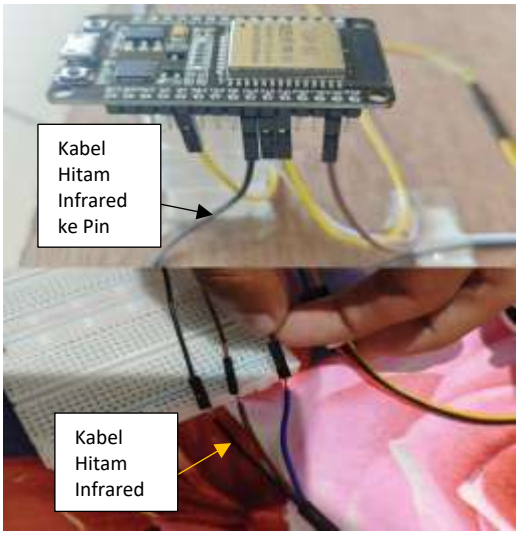


Gambar 4.1 Kabel VIN dan GND pada ESP32 ke *Breadboard*

2. Perakitan Sensor Keamanan Fisik

Selanjutnya, penulis memasang sensor *Infrared Proximity* (E18-D80NK) yang berfungsi mendeteksi keberadaan galon. Sensor ini memiliki tiga kabel yaitu; kabel VCC dan GND dihubungkan ke jalur daya, sedangkan kabel sinyal data (*Out*) dihubungkan ke Pin D18 pada *ESP32*.

Berikut untuk perangkaian kabel yang terhubung pada sensor *Infrared Proximity* (E18-D80NK) ke pin yang ada pada *ESP32* pada Gambar 4.2.

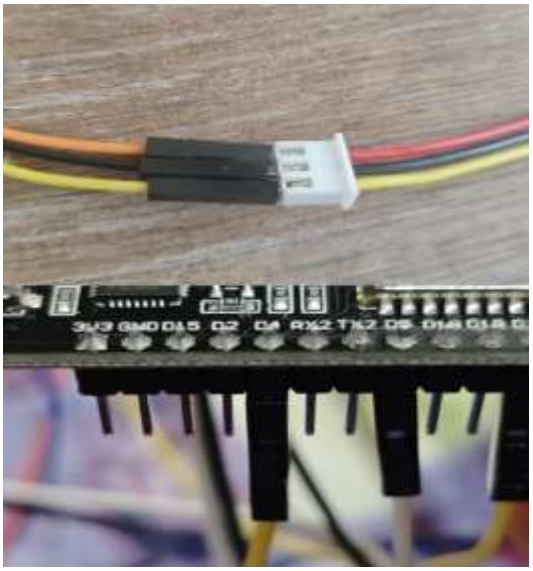


Gambar 4.2 Koneksi Kabel dari Sensor *Infrared* ke *ESP32*

3. Perakitan Sensor Penakar Volume

Untuk menakar debit air, penulis memasang *Water Flow Sensor* (YF-S401) pada jalur selang air. Kabel daya merah dan hitam dihubungkan ke rel listrik, sedangkan kabel sinyal data yang berwarna kuning secara spesifik dihubungkan ke Pin D4 *ESP32*. Pin ini dipilih karena mendukung fungsi pembacaan interupsi (*interrupt*) agar putaran kincir air terhitung dengan sangat presisi.

Berikut untuk perangkaian kabel yang terhubung dari *Water Flow Sensor* (YF-S401) ke pin yang ada pada *ESP32* pada Gambar 4.3.

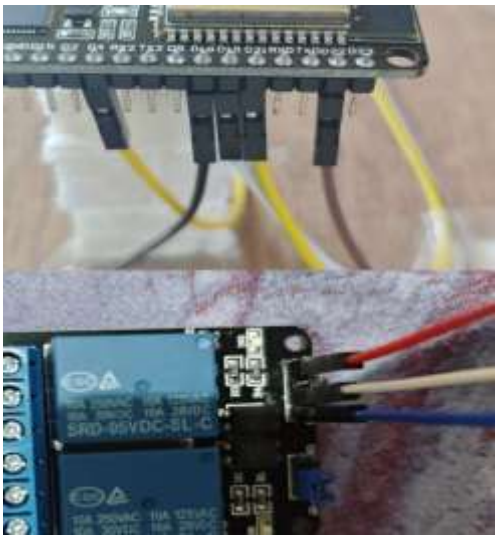


Gambar 4.3 Koneksi Kabel *Water Flow Sensor* Ke *ESP32*

4. Perakitan Pengendali Pompa

Sebagai aktuator pemompa air, penulis menggunakan *Waterpump* DC 12V yang dikendalikan oleh modul *Relay* 2-*Channel*. Kabel kendali (*IN*) dari *relay* berwarna putih disambungkan ke Pin D19 ESP32, yang bertindak sebagai saklar elektronik pemutus arus.

Berikut untuk perangkaian kabel dari *relay* yang terhubung ke *ESP32* pada Gambar 4.4.



Gambar 4.4 Koneksi Kabel *Relay* Ke *ESP32*

Kemudian dilanjutkan koneksi kabel positif (+) dari pompa air DC 12V ke *port* NO (*Normally Open*) pada *relay* pada Gambar 4.5.



Gambar 4.5 Koneksi Kabel *Waterpump* DC 12V Ke *Relay*

5. Perakitan Layar Indikator (*LCD* 16x2 *I2C)*

Terakhir, modul *LCD* 16x2 dengan modul tambahan *I2C* dipasang untuk memberikan informasi teks di lokasi mesin. Penggunaan modul *I2C*

sangat menghemat kabel, di mana pin data (*SDA*) kabel berwarna kuning pada modul *LCD* hanya perlu dihubungkan ke Pin D21, dan pin selaras waktu (*SCL*) kabel berwarna coklat pada modul *LCD* dihubungkan ke Pin D22 pada *ESP32*.

Berikut untuk perangkaian kabel yang terhubung dari Modul *LCD* I2C ke pin yang ada pada *ESP32* pada Gambar 4.6.



Gambar 4.6 Koneksi Kabel *LCD* Ke *ESP32*

Agar proses perakitan ini dapat direplikasi dengan mudah dan sistematis, rincian detail mengenai sambungan pin, warna kabel standar dari masing-masing modul, beserta fungsi spesifiknya diuraikan secara lengkap pada Tabel 4.1 berikut ini.

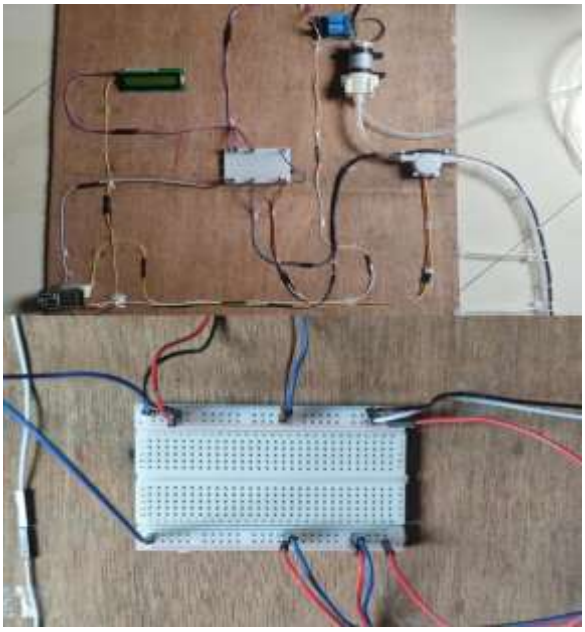
Table 4.1 Konfigurasi Rincian Pin dan Pengkabelan (*Wiring*) pada *ESP32*

No	Nama Komponen	Kabel/Pin Komponen	Terhubung ke (*ESP32*/Daya)	Keterangan Fungsi
1	*Proximity* (E18-D80NK)	Data/*Out* (Kabel Hitam/Kuning)	Pin 18 (*ESP32*)	Mengirim sinyal logika (*LOW*) saat mendeteksi keberadaan galon

No	Nama Komponen	Kabel/Pin Komponen	Terhubung ke (*ESP32*/Daya)	Keterangan Fungsi
2	*Proximity* (E18-D80NK)	VCC & GND (Kabel Coklat & Biru)	Jalur 5V & GND (*Breadboard*)	Suplai daya listrik untuk sensor inframerah.
3	*Water Flow* (YF-S401)	Data / Pulse (Kabel Kuning)	Pin 4 (*ESP32*)	Mengirim data pulsa kincir air untuk menakar volume
4	*Water Flow* (YF-S401)	VCC & GND (Kabel Merah & Hitam)	Jalur 5V & GND (*Breadboard*)	Suplai daya listrik untuk sensor debit air
5	Modul *Relay* 2-*Channel*	Sinyal (Pin IN1)	Pin 19 (*ESP32*)	Menerima instruksi saklar otomatis untuk menghidupkan pompa
6	Modul LCD 16x2 I2C	SDA (*Serial Data*)	Pin 21 (*ESP32*)	Jalur pengiriman data teks ke layar (*I2C Protocol*)
7	Modul LCD 16x2 I2C	SCL (*Serial Clock*)	Pin 22 (*ESP32*)	Jalur sinkronisasi waktu pengiriman data layar
8	Seluruh Modul Terkait	VCC / VIN & GND	*Adaptor* 12V (*Step-Down*)	Sumber tegangan arus searah utama bagi seluruh sistem

Pada bagian ini merupakan rangkaian pengkabelan untuk memperlihatkan arsitektur koneksi untuk distribusi daya *VCC* dan *GND* pada setiap masing-masing komponen dari Mikrokontroller *ESP32*, Sensor *Flowmeter YF-S401*, Sensor *Infrared E18-D80NK*, *Modul Relay*, *Waterpump DC* 12V, *LCD 16x2 I2C*, dan *Adaptor* 12V.

Berikut untuk rincian koneksi pengkabelan di atas *breadboard* yang memperlihatkan keseluruhan arsitektur pendistribusian daya *VCC* dan *GND* secara paralel pada Gambar 4.7.



Gambar 4.7 Rangkaian Koneksi Kabel VCC dan GND Ke *Breadboard*

b) Implementasi Perangkat Lunak

Pada bagian ini merupakan implementasi perangkat lunak pada sistem, mencakup pengaturan jaringan lokal ke publik, integrasi layanan pihak ketiga, hingga seluruh antarmuka (*User Interface*) yang menghubungkan interaksi antara pengguna dengan mesin. *Web* ini dibangun menggunakan *framework CodeIgniter 4* dan didesain responsif agar optimal saat diakses melalui *smartphone*.

Implementasi antarmuka dijabarkan secara berurutan berdasarkan alur penggunaan sistem, sebagai berikut.

1. Konfigurasi *XAMPP*

   Sebelum *website* di-*online*-kan menggunakan *Ngrok*, penulis perlu menyiapkan server lokal terlebih dahulu menggunakan aplikasi *XAMPP*. Pertama membuka *XAMPP Control Panel* lalu mengaktifkan modul *Apache*

sebagai web server dan *MySQL* sebagai *database* dengan menekan tombol *Start*.

Setelah indikator kedua modul tersebut menyala hijau, penulis memindahkan seluruh folder aplikasi *web CodeIgniter* 4 ke dalam direktori bawaan *XAMPP*, yaitu *xampp/htdocs/smart_dispenser*. Dengan langkah ini, *website* pemesanan sudah siap dijalankan di komputer lokal dan siap dihubungkan ke jaringan internet publik.

Berikut ini tampilan panel *XAMPP* yang sudah aktif dapat dilihat pada Gambar 4.8.



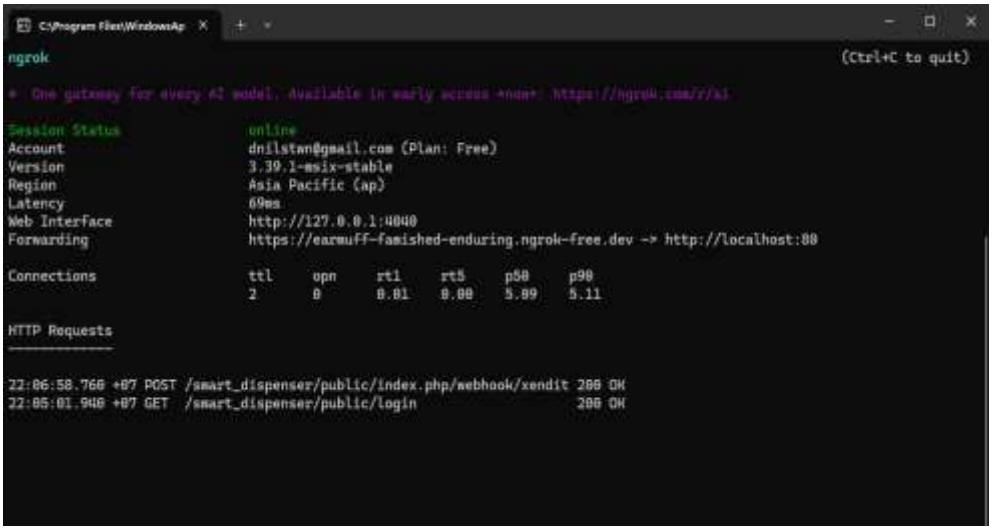
Gambar 4.8 Aktivasi *Apache* dan *MySQL* pada *XAMPP Control Panel*

2. Konfigurasi *tunneling* server lokal (*Ngrok*)

Aplikasi *web* dijalankan menggunakan server lokal *XAMPP*. Namun, agar sistem *web* ini dapat diakses secara *online* oleh alat *ESP32* dan layanan pihak ketiga (*Xendit*), penulis menggunakan layanan *tunnelling* dari *Ngrok*. Implementasinya dilakukan dengan membuka *Command Prompt* (*CMD*) pada sistem operasi, lalu menjalankan perintah *ngrok http 8080*. Perintah ini

akan menghasilkan sebuah tautan publik (*link public*) sementara (contoh: (*https://earmuff-famished-enduring.ngrok-free.dev*)) yang akan digunakan sebagai *Base URL* utama pada mikrokontroler dan *Webhook Xendit*.

Berikut merupakan tampilan eksekusi perintah *CMD Ngrok* dapat dilihat pada Gambar 4.9.



Gambar 4.9 Konfigurasi *Tunnelling* Server Menggunakan *CMD Ngrok*

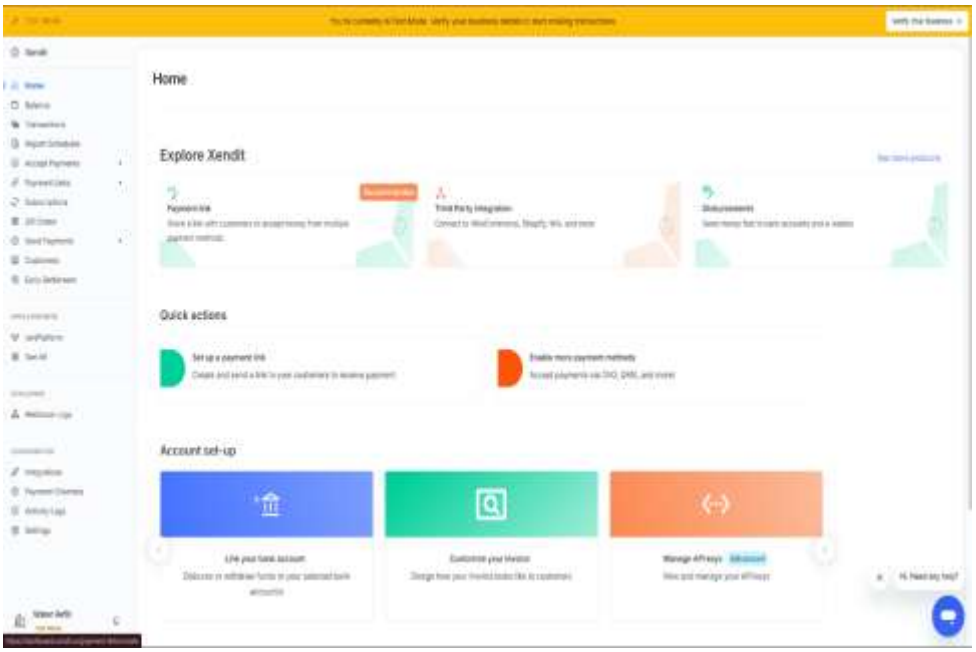
3. Integrasi *payment gateway* (*API Xendit*)

Pada bagian ini merupakan fitur pembayaran nirkontak otomatis, penulis mengintegrasikan layanan *payment gateway Xendit*. Karena sistem hanya berbentuk *prototype*, implementasi hanya difokuskan pada lingkungan *testing mode*. Berikut ini langkah-langkah konfigurasinya.

a. Registrasi akun *Xendit* (*Testing Mode*)

Bagian ini proses yang penulis lakukan adalah resgistrasi akun pengembang pada protal *Xendit*. Setelah masuk ke *dashboard*, *testing mode* langsung diaktifkan agar simulasi transaksi tidak memotong saldo asli.

Berikut untuk tampilan *dashboard* portal *Xendit* setelah masuk registrasi akun dapat dilihat pada Gambar 4.10.



Gambar 4. 10 Tampilan *Dashboard Xendit* (*Testing Mode*)

b. Pembuatan *API Keys*

Setelah itu, penulis membuat *API Key* dengan menavigasi ke menu *API Keys* untuk men-*generate secret API Key*. Kunci rahasia ini di salin ke dalam file *Checkout.php* pada *CodeIgniter 4* agar *website* memiliki izin resmi untuk menampilkan tagihan *QRIS*.

Berikut ini *source code* yang digunakan untuk menampilkan tagihan *QRIS* secara resmi ke *CodeIgniter 4* pada Gambar 4.11.

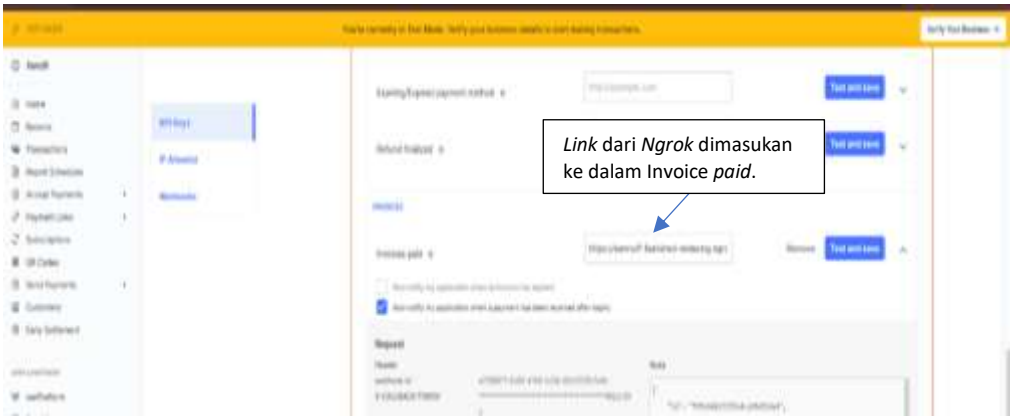


Gambar 4.11 Pembuatan *Secret API Keys*

c. Pengaturan *Webhook Xendit*

Pada bagian ini, penulis memasukan *link* rute *API web* yang didapat dari *Ngrok* sebelumnya (contoh: *https://earmuff-famished-enduring.ngrok-free.dev*) ke dalam pengaturan *webhook Xendit* pada bagian kolom *invoices paid*, tujuannya agar sistem *Xendit* dapat mengirimkan notifikasi "lunas" secara otomatis ke *website*.

Berikut ini tampilan pengaturan untuk memasukan *link* rute *API* yang di dapat dari *Ngrok* ke dalam pengaturan *webhook Xendit* pada Gambar 4.12.

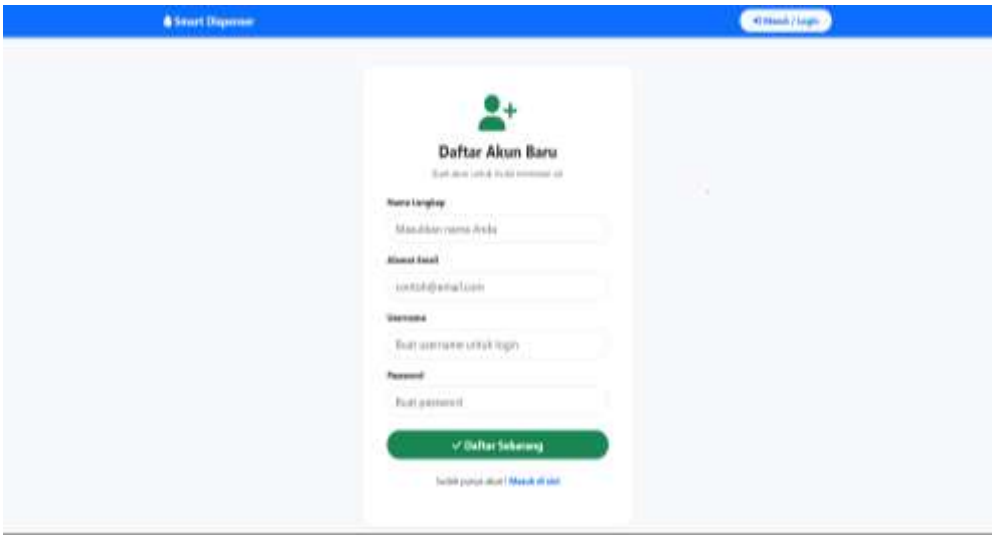


Gambar 4.12 Memasukan *Link Rute API Ngrok* ke *Webhook Xendit*

4. Antarmuka halaman registrasi dan *login*

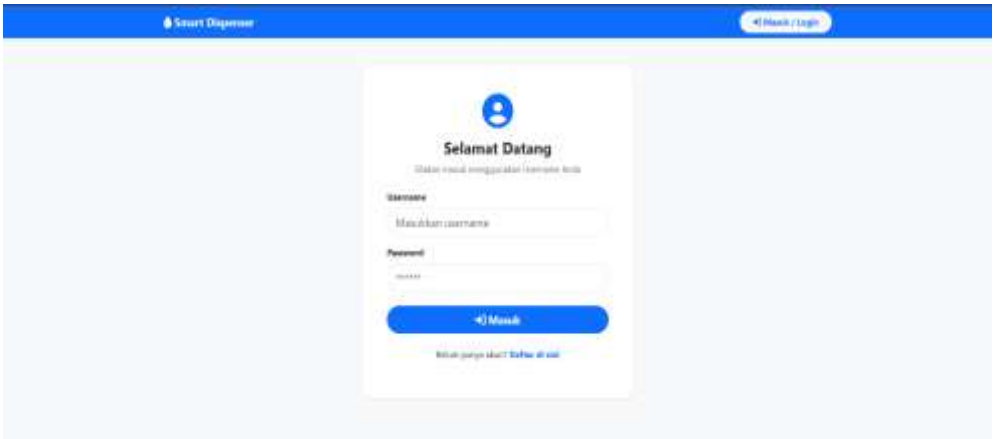
Halaman ini merupakan gerbang utama keamanan sistem. Pelanggan baru diwajibkan untuk membuat akun melalui halaman registrasi dengan memasukkan data diri dasar. Setelah akun terdaftar, pengguna dapat mengakses halaman *Login* untuk memverifikasi identitas (autentikasi) sebelum dapat melakukan pemesanan air.

Berikut ini merupakan tampilan dari antarmuka registrasi pengguna dapat dilihat pada Gambar 4.13.



Gambar 4. 13 Halaman Register Pengguna

Selanjutnya ketika pengguna sudah registrasi akun maka akan beralih ke tampilan *login* pengguna, dapat dilihat pada Gambar 4.14.



Gambar 4.14 Halaman *Login* Pengguna

5. Antarmuka halaman pemesanan dan data antar

Setelah berhasil masuk, pengguna akan diarahkan ke halaman pemesanan produk. Pada antarmuka ini, pengguna dapat memilih varian takaran air yang disediakan oleh depot (misalnya 600 mL, 1.5 Liter, atau 19 Liter). Selain itu, pada halaman ini juga disediakan formulir pengisian data antar, di mana pelanggan harus mencantumkan nama dan alamat pengiriman secara jelas.

Berikut merupakan tampilan antarmuka pemilihan produk dan formulir pengiriman dapat dilihat pada Gambar 4.15.



Gambar 4.15 Halaman Pemesanan Air

6. Antarmuka halaman pembayaran (Integrasi *API Xendit*)

Setelah data pesanan dikonfirmasi, sistem akan membawa pengguna ke halaman pembayaran. Implementasi pada halaman ini sangat krusial karena terintegrasi langsung dengan *Payment Gateway Xendit*. Sistem akan secara otomatis menerbitkan kode *Quick Response Code Indonesian Standard*

(*QRIS*) dinamis yang sesuai dengan total nominal tagihan pengguna, sehingga proses pembayaran menjadi nirkontak dan otomatis terverifikasi.

Berikut merupakan tampilan halaman pembayaran *QRIS* dapat dilihat pada Gambar 4.16.



Gambar 4.16 Tampilan *Barcode QRIS*

7. Antarmuka halaman monitoring (*Customer Side*)

Ketika pembayaran berhasil dikonfirmasi oleh sistem, antarmuka pengguna akan beralih ke halaman Monitoring atau layar tunggu. Halaman ini berfungsi sebagai indikator pemantauan secara *real-time* bagi pengguna. Layar ini akan menampilkan instruksi interaktif yang tersinkronisasi dengan mesin fisik, seperti peringatan "Menunggu Wadah", status "Sedang Mengisi", hingga informasi bahwa pesanan telah "Selesai" dituangkan.

Berikut merupakan antarmuka pemantauan proses pengisian air di sisi pengguna dari menunggu peletakan wadah dapat dilihat pada Gambar 4.17.



Gambar 4.17 Halaman Monitoring Pengguna Menunggu Wadah

Kemudian, ketika wadah sudah diletakan maka akan diproses. Berikut ini merupakan antarmuka pemantauan proses pengisian air di sisi pengguna dari menunggu peletakan wadah dapat dilihat pada Gambar 4.18.



Gambar 4.18 Halaman Monitoring Pengisian Air Sisi Pengguna

Selanjutnya, ketika pengisian telah selesai dilakukan, tampilan monitoring pada pengguna akan berubah. Berikut ini halaman monitoring pada pengguna ketika pengisian telah selesai dilakukan pada Gambar 4.19.

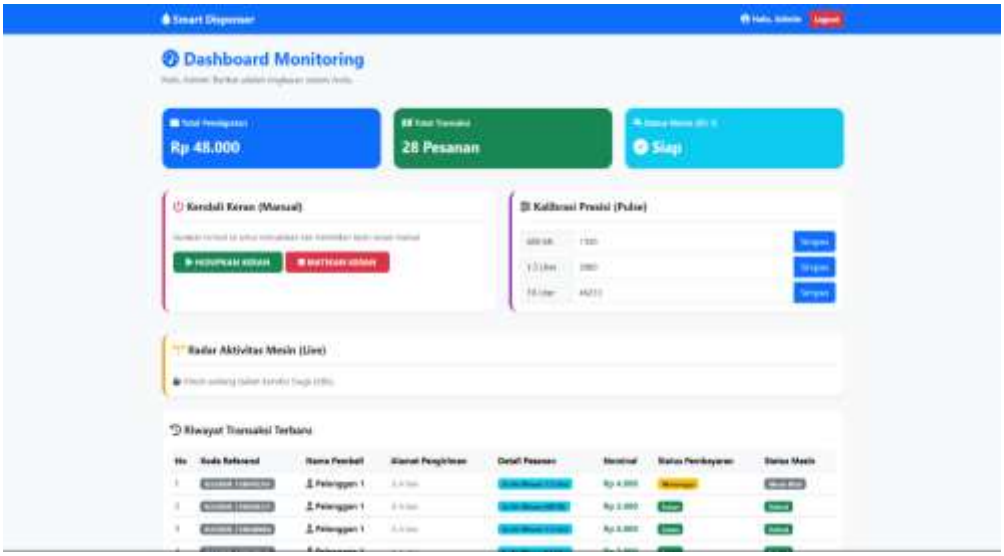


Gambar 4.19 Halaman Monitoring Pengguna Ketika Pengisian Selesai

8. Antarmuka halaman *dashboard* (*Admin Side*)

Halaman ini adalah pusat kendali (*Control Panel*) yang didesain khusus dan hanya dapat diakses oleh *admin* depot Bluebub. Pada implementasinya, halaman ini merangkum seluruh rekam jejak operasional, meliputi kartu indikator total pendapatan dan total transaksi. Di halaman ini juga disematkan fitur *Live Radar* untuk melacak status aktivitas mesin di lapangan, serta fitur kalibrasi pulse mesin yang dapat diatur dari jarak jauh.

Berikut merupakan antarmuka dari *Dashboard Admin* dapat dilihat pada Gambar 4.20.



Gambar 4. 20 Halaman *Dashboard Admin*

c) Implementasi Kode Program

Setelah proses perancangan sistem selesai dilakukan, tahap selanjutnya adalah implementasi program pada *website* dan perangkat . Implementasi ini bertujuan untuk menerapkan seluruh rancangan yang telah dibuat sehingga sistem dapat berjalan sesuai dengan fungsi yang diharapkan.

Pada penelitian ini implementasi program dilakukan pada *website* yang dibangun menggunakan *framework CodeIgniter 4* dan perangkat yang menggunakan mikrokontroler *ESP32*. Program yang dibuat berfungsi untuk mengelola transaksi pelanggan, menerima notifikasi pembayaran dari *Xendi*t, mengirim data pesanan ke perangkat dispenser, serta mengontrol proses pengisian air secara otomatis berdasarkan volume yang dipilih pelanggan.

Berikut ini penulis menjelaskan beberapa potongan kode program yang saling terhubung di dalam sistem ini antara satu sama lain.

1. Pembuatan *Invoiced* Pembayaran

Pada proses pemesanan, pengguna terlebih dahulu memilih produk yang ingin dibeli kemudian mengisi data pemesanan yang tersedia pada *website*. Setelah data berhasil diterima, sistem akan menghitung total pembayaran berdasarkan jumlah produk yang dipilih pelanggan.

Selanjutnya sistem akan membuat *invoice* pembayaran menggunakan layanan *Xendit*. *Invoice* ini digunakan sebagai media pembayaran yang nantinya akan ditampilkan kepada pelanggan. Melalui *invoice* tersebut pelanggan dapat melakukan pembayaran menggunakan *QRIS* maupun metode pembayaran lain yang disediakan oleh *Xendit*.

Sistem menghitung total pembayaran berdasarkan jumlah pesanan pelanggan kemudian membuat *invoice* pembayaran melalui *API Xendit*. Setelah *invoice* berhasil dibuat, pelanggan akan diarahkan ke halaman pembayaran yang disediakan oleh *Xendit* untuk menyelesaikan transaksi.

Berikut kode program yang digunakan untuk mengatur pembuatan invoice pembayaran pada Gambar 4.21



Gambar 4.21 Kode Program Pembuatan *Invoice* Pembayaran

2. *Webhook* Pembayaran

Setelah pengguna menyelesaikan pembayaran, sistem memerlukan mekanisme untuk mengetahui status transaksi tersebut. Untuk memenuhi kebutuhan tersebut digunakan fitur *webhook* dari *Xendit*.

Webhook berfungsi untuk mengirimkan notifikasi otomatis dari server *Xendit* ke *website* ketika terjadi perubahan status pembayaran sehingga sistem dapat memperbarui data transaksi secara otomatis.

Sistem menerima data notifikasi dari *Xendit* kemudian memeriksa status pembayaran yang diterima. Jika status transaksi bernilai *PAID*, maka status pembayaran pada *database* akan diperbarui menjadi *Paid* sebagai tanda bahwa pesanan telah berhasil dibayar.

Berikut kode program yang digunakan untuk mengirimkan notifikasi otomatis dari server ke *website* pada Gambar 4.22.



```
$data = $this->request->getJSON();

if ($data && isset($data->external_id) && isset($data->status)) {

    if ($data->status == 'PAID') {

        $db = \Config\Database::connect();
        $db->table('transaction')
           ->where('payment_ref', $data->external_id)
           ->update(['payment_status' => 'Paid']);
    }
}
```

Gambar 4.22 Kode Program Mengirim Notifikasi dari Server ke *Website*

3. Komunikasi *Website* dengan *ESP32*

Setelah pembayaran berhasil diverifikasi, *website* akan meneruskan informasi pesanan kepada perangkat melalui *REST API*. *API* berfungsi

sebagai penghubung antara *website* dan *ESP32* sehingga kedua sistem dapat saling bertukar data melalui jaringan internet.

Server mengirimkan data pesanan dalam format *JSON* yang berisi volume air, target pulsa sensor *water flow*, jumlah pesanan, dan nomor referensi transaksi. Informasi tersebut akan digunakan oleh *ESP32* untuk menjalankan proses pengisian sesuai pesanan pelanggan.

Berikut kode program yang digunakan untuk mengirimkan data pesanan format *JSON* pada Gambar 4.23.

```
return $this->response->setJSON([
    'perintah'     => 'NYALA',
    'volume'       => $volume,
    'target_pulse' => (int) $target_pulse,
    'payment_ref'  => $trx['payment_ref'],
    'qty'          => $trx['qty'],
    'qty_selesai'  => $trx['qty_selesai'],
    'manual'       => $manual
]);
```

Gambar 4.23 Kode Program Mengirimkan Data Format *JSON*

4. Pembacaan Data *API* pada *ESP32*

Agar dapat mengetahui adanya pesanan baru, *ESP32* melakukan pengecekan secara berkala ke server *website* melalui *API* yang telah disediakan.

Data yang diterima dari server berbentuk *JSON* sehingga perlu dilakukan proses parsing sebelum digunakan oleh program.

ESP32 mengirimkan permintaan data ke server menggunakan metode *HTTP GET*. Setelah data diterima, sistem melakukan parsing *JSON* dan menyimpan data tersebut ke dalam variabel yang akan digunakan selama proses pengisian berlangsung.

Berikut kode program yang digunakan untuk mengirimkan permintaan data ke server menggunakan *HTTP GET* pada Gambar 4.24.

```
HTTPClient http;
http.begin(baseUrl + "cek_mesin");
http.addHeader("ngrok-skip-browser-warning", "69420");
if (http.GET() == 200) {
    DynamicJsonDocument doc(1024);
    deserializeJson(doc, http.getString());
targetPulses = doc["target_pulse"].as<int>();
targetVolume = doc["volume"].as<int>();
```

Gambar 4.24 Kode Program Permintaan Data ke Server dengan *HTTP GET*

5. Deteksi Wadah Menggunakan *Sensor Proximity*

Sebelum pompa dijalankan, sistem harus memastikan bahwa wadah atau galon telah berada pada posisi pengisian. Untuk melakukan proses tersebut digunakan *sensor proximity infrared* (*E18-D80NK*).

Sensor akan mendeteksi keberadaan objek yang berada di depannya. Jika wadah terdeteksi maka sistem akan melanjutkan proses pengisian.

ESP32 membaca kondisi *sensor proximity* untuk mengetahui keberadaan wadah. Jika sensor mendeteksi objek, maka sistem akan masuk ke tahap persiapan sebelum proses pengisian dimulai.

Berikut kode program yang digunakan untuk medeteksi keberadaan wadah atau galon sebelum proses pengisian dilanjutkan pada Gambar 4.25.

```
bool gelasAda = (digitalRead(PROXIMITY_PIN) == LOW);
  if (gelasAda) {
    currentStatus = "PREPARING";
    prepareTimer = millis();
    lcd.clear();
  }
```

Gambar 4.25 Kode Program Mendeteksi Keberaaan Wadah atau Galon

6. Pengukuran Volume Air Menggunakan *Sensor Water Flow*

Untuk memastikan volume air yang keluar sesuai dengan pesanan pengguna, sistem menggunakan *sensor water flow YF-S401*. Sensor ini menghasilkan pulsa berdasarkan jumlah aliran air yang melewati sensor.

Jumlah pulsa yang terbaca digunakan untuk menghitung volume air yang telah terisi. Volume air dihitung menggunakan jumlah pulsa yang terbaca dibandingkan dengan target pulsa yang telah ditentukan sebelumnya. Nilai tersebut digunakan untuk memantau progres pengisian secara *real-time*.

Berikut kode program yang digunakan untuk menghitung target jumlah volume air menggunakan pulse yang melewati *sensor water flow* pada Gambar 4.26.

```
if (targetPulses > 0) {
    current_mL = (pulseCount * targetVolume) / targetPulses;
} else {
    current_mL = 0;
}
```

Gambar 4.26 Kode program menghitung jumlah pulse

7. Pengendalian Pompa Air Otomatis

Pompa air dikendalikan menggunakan modul *relay* yang terhubung dengan *ESP32*. *Relay* berfungsi sebagai saklar elektronik yang akan menghidupkan dan mematikan pompa secara otomatis sesuai kondisi pengisian.

Relay akan mengaktifkan pompa selama jumlah air yang keluar belum mencapai target. Ketika jumlah pulse telah memenuhi target yang

ditentukan, *relay* akan mematikan pompa sehingga proses pengisian berhenti secara otomatis.

Berikut kode program yang digunakan untuk mematikan atau menyalakan *relay* saat proses pengisian air pada Gambar 4.27.

```
digitalWrite(RELAY_PIN, LOW);
if (pulseCount >= targetPulses) {
  digitalWrite(RELAY_PIN, HIGH);
  delay(50);
  lcd.init();

  currentStatus = "FINISH";
  oldTime = millis();
}
```

Gambar 4.27 Kode Program Menyalakan atau Mematikan *Relay*

8. Pelaporan Status Pengisian

Setelah proses pengisian selesai dilakukan, *ESP32* akan mengirimkan laporan kembali ke server sebagai tanda bahwa pesanan telah berhasil diproses.

Informasi tersebut digunakan untuk memperbarui status transaksi pada *database* dan menampilkan perkembangan pesanan pada halaman monitoring.

Berikut kode program yang digunakan untuk memperbarui status transaksi pada *database* ke halaman monitoring pada Gambar 4.28.

```
int responCode = hitApiWeb("lapor_progress/" + currentPaymentRef);
if (responCode == 200 || responCode == 201) {
    isReported = true;
}
```

Gambar 4.28 Kode Program Memperbarui Status Transaksi di *Database*

9. Penerimaan Status Pengisian pada Server

Setelah semuanya berhasil dijalankan, *ESP32* mengirimkan informasi bahwa proses pengisian telah selesai dilakukan. Setelah laporan diterima, server akan memperbarui status transaksi pada *database* sehingga pesanan dapat dinyatakan selesai dan ditampilkan pada halaman monitoring *admin*.

Berikut kode program yang digunakan untuk memperbarui status transaksi pada *database* bahwa status pesanan telah diselesaikan dan di perbarui pada tampilan *dashboard* monitoring *admin* pada Gambar 4.29.

```
$qty_selesai_baru = $trx['qty_selesai'] + 1;

if ($qty_selesai_baru >= $trx['qty']) {
    $db->table('transaction')->where('transaction_id', $trx['transaction_id'])->update([
        'qty_selesai'      => $qty_selesai_baru,
        'status_pengisian' => 'Selesai',
        'payment_status'   => 'Completed'
    ]);
```

Gambar 4.29 Kode Program Untuk Memperbarui Status Pesanan Selesai

4.1.2 Pengujian

Pada tahap pengujian ini dilakukan untuk mengetahui apakah sistem yang telah dibangun dapat berjalan sesuai dengan kebutuhan dan tujuan penelitian. Pengujian dilakukan terhadap beberapa komponen utama sistem, meliputi layanan pembayaran *Xendit*, komunikasi data antara *website* dan *ESP32*, sensor yang digunakan pada perangkat, serta pengujian fungsionalitas *website*.

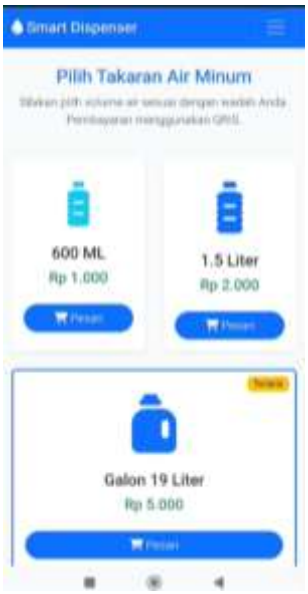
Selain itu, dilakukan pula pengujian terhadap pengguna untuk mengetahui tingkat kemudahan penggunaan *website* pemesanan yang telah dikembangkan. Hasil pengujian ini digunakan sebagai dasar dalam melakukan evaluasi terhadap kinerja sistem secara keseluruhan.

a. Pengujian *Payment Gateway Xendit*

Disini dilakukan pengujian *Payment Gateway Xendit* untuk memastikan bahwa sistem dapat membuat invoice pembayaran dan menerima konfirmasi pembayaran secara otomatis.

Proses pengujian diawali dengan melakukan pemesanan produk melalui *website* .

Berikut ini dapat dilihat pengguna melakukan pemesanan pada halaman pemesanan pada Gambar 4.30.



Gambar 4.30 Halaman pemesanan

Setelah pengguna memilih produk dan mengisi data pemesanan, sistem berhasil membuat *invoice* pembayaran menggunakan layanan *Xendit*. *Invoice* yang dihasilkan kemudian menampilkan kode *QRIS* yang dapat digunakan oleh pelanggan untuk melakukan pembayaran.

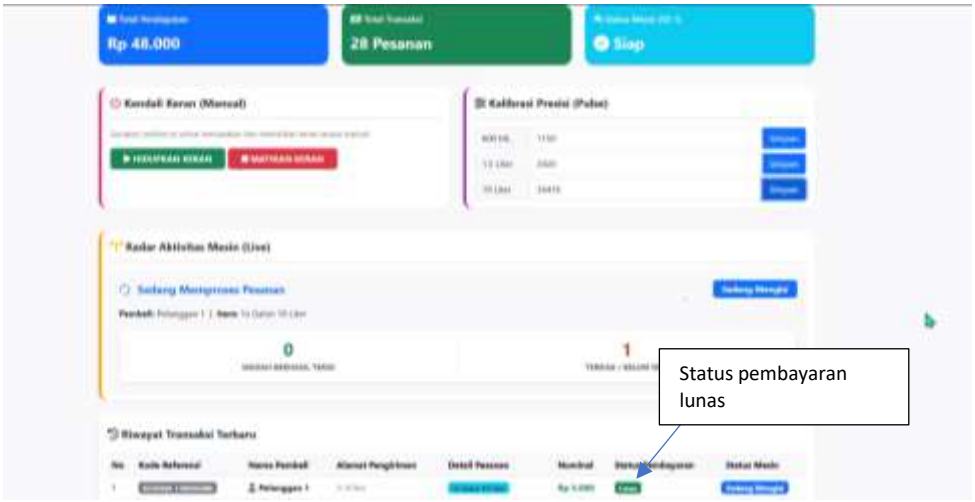
Berikut ini halaman *Xendit* menampilkan *barcode QRIS* untuk dilanjutkan ke pembayaran (*simulation mode*) pada Gambar 4.31.



Gambar 4.31 Halaman *QRIS Xendit*

Setelah pembayaran berhasil dilakukan, *Xendit* mengirimkan notifikasi pembayaran ke server melalui mekanisme *webhook*. Sistem kemudian memperbarui status transaksi dari *Pending* menjadi *Paid* secara otomatis. Perubahan status tersebut menandakan bahwa pesanan siap diproses oleh *ESP32* untuk di lanjutkan ke proses pengisian.

Berikut halaman *dashboard admin* menampilkan status *paid* (lunas) pada riwayat pemesanan pelanggan pada Gambar 4.32.



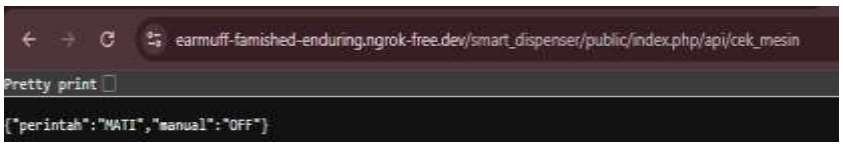
Gambar 4.32 Halaman Riwayat Pemesanan Status *Paid*

b. Pengujian Komunikasi *Website* dan *ESP32*

Kemudian pada tahap ini dilakukan pengujian komunikasi untuk memastikan bahwa *website* dapat mengirimkan data pesanan kepada *ESP32* melalui *API*.

Proses pengujian dimulai dengan melakukan pembayaran hingga status transaksi berubah menjadi *Paid*. Setelah status pembayaran diperbarui, *ESP32* secara otomatis melakukan permintaan data ke endpoint *API* yang tersedia pada *website*.

Berikut ini tampilan *JSON* cek_mesin atau mengecek status pada modul *ESP32* yang meminta data ke *endpoint API* yang tersedia di *website* pada Gambar 4.33.



Gambar 4.33 Tampilan *JSON* Cek Status *ESP32*

Data pesanan yang diterima berupa informasi volume air, target pulse *sensor water flow*, jumlah pesanan, dan nomor referensi transaksi. Data tersebut kemudian diproses oleh *ESP32* untuk menjalankan proses pengisian air sesuai dengan pesanan pelanggan.

Ketika pembayaran berhasil, maka tampilan *LCD* akan menampilkan status bahwa pembayaran telah berhasil lalu memerintahkan untuk menaruh wadah dibawah saluran pengisian air.

Berikut tampilan *LCD* ketika pembayaran telah berhasil dan alat sedang menunggu wadah ditaruh pada Gambar 4.34.



Gambar 4.34 Modul *LCD* Menunggu Wadah

Setelah itu sistem akan menyuruh pengguna untuk menaruh wadah di tempat saluran pengeluaran air, lalu tampilan LCD akan berubah untuk menampilkan proses air yang sedang keluar sesuai target pilihan menu pesanan.

Berikut tampilan pada modul *LCD* ketika pemabayaran sudah berhasil lalu dilanjutkan dengan proses pengisian air pada Gambar 4.35.



Gambar 4.35 Modul *LCD* Menampilkan Proses Pengisian Air

Berikut tampilan pada sisi pelanggan ketika pembayaran berhasil lalu dilanjutkan ke proses pengisian air pada Gambar 4.36.



Gambar 4.36 Sisi Pelanggan Saat Air Sedang Proses Pengisian

Kemudian, ketika proses pengisian sudah selesai, modul *LCD* akan berubah dan menampilkan tampilan pengisian telah sukses, lalu sistem menyuruh untuk mengambil wadah.

Berikut tampilan modul *LCD* ketika proses pengisian sudah berhasil dilakukan pada Gambar 4.37.



Gambar 4.37 Tampilan Modul *LCD* Pengisian Sukses

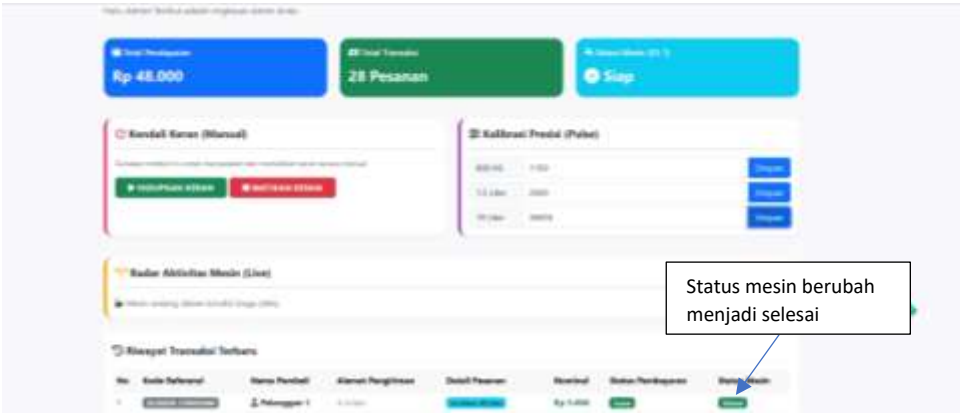
Setelah alat pada *ESP32* selesai memproses pengisian hingga selesai dan *LCD* menampilkan tempilan tersebut, maka pada sisi tampilan *website* pelanggan akan menampikan hasil monitoring pada Gambar 4.38.



Gambar 4.38 Sisi Pelanggan Ketika Proses Pengisian Selesai

Setelah pemesanan telah diselesaikan, maka *ESP32* akan memberitahu sistem kalau pesanan telah diselesaikan dan status pengisian pada *dashboard admin* juga akan berubah mejadi selesai.

Berikut tampilan *dashboard admin* ketika proses pemesanan telah diselesaikan pada Gambar 4.39.



Gambar 4.39 Status Mesin Pada *Dashboard Admin*

c. Pengujian *Sensor Proximity* (E18-D80NK)

Pada tahap ini dilakukan pengujian *sensor proximity* untuk mengetahui kemampuan sensor dalam mendeteksi keberadaan wadah pada area pengisian.

Pengujian dilakukan dengan menempatkan wadah di bawah saluran pengisian air. Ketika wadah berada dalam jangkauan sensor, sistem berhasil mendeteksi keberadaan wadah dan melanjutkan proses pengisian.

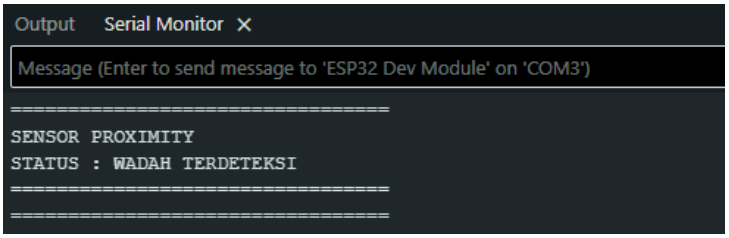
Berikut ini adalah ketika sensor mendeteksi keberadaan wadah pada Gambar 4.40.



Gambar 4.40 *Sensor Proximity* Mendeteksi Keberadaan Wadah

Lalu ketika sensor mendeteksi keberadaan wadah, respon dari serial monitor akan menampilkan status bahwa wadah terdeteksi.

Berikut tampilan status dari serial monitor ketika wadah terdeteksi oleh *sensor proximity* pada Gambar 4.41.



Gambar 4.41 Status Serial Monitor Ketika Wadah Terdeteksi

Sebaliknya, ketika wadah tidak berada pada area sensor, sistem tidak mengaktifkan pompa sehingga air tidak keluar.

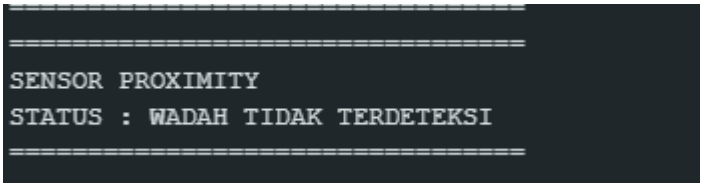
Berikut tampilan ketika *sensor proximity* tidak mendeteksi keberadaan wadah disekitarnya pada Gambar 4.42.



Gambar 4.42 *Sensor Proximity* Tidak Mendeteksi Keberadaan Wadah

Pada serial monitor pun begitu, ketika sensor tidak mendeteksi keberadaan wadah, status dari serial monitor akan menampilkan status bahwa wadah tidak terdeteksi.

Berikut tampilan status dari serial monitor ketika tidak ada wadah yang terdeteksi oleh sensor proximity pada Gambar 4.43.



Gambar 4.43 Status Serial Monitor Ketika Tidak Ada Wadah Terdeteksi

d. Pengujian *Sensor Water Flow* (YF-S401)

Pada tahap ini dilakukan pengujian untuk sensor water flow untuk mengetahui kemampuan sensor dalam mengukur volume air yang keluar selama proses pengisian. Selain itu, pengujian ini juga bertujuan untuk menentukan nilai pulse yang paling sesuai agar volume air yang dihasilkan mendekati target yang telah ditentukan.

Pada tahap kalibrasi, dilakukan beberapa kali percobaan untuk mencari jumlah pulse yang menghasilkan volume air yang paling akurat. Berdasarkan hasil kalibrasi yang telah dilakukan, diperoleh nilai pulse sebesar 1160 pulse

untuk volume 600 mL, 2940 pulse untuk volume 1,5 liter, dan 36525 pulse untuk volume 19 liter. Nilai tersebut kemudian digunakan sebagai acuan pada sistem untuk menghentikan pompa secara otomatis ketika target volume telah tercapai.

Berikut ini nilai kalibrasi pulse yang digunakan untuk mendapatkan volume air yang pas pada Tabel 4.2.

Table 4.2 Nilai Kalibrasi

No	Volume Target	Nilai Pulse
1.	600 mL	1160 Pulse
2.	1,5 Liter	2940 Pulse
3.	19 Liter	36525 Pulse

Setelah melakukan kalibrasi, penulis melakukan uji coba untuk volume air 600 mL dengan nilai kalibrasi sebanyak 1160 pulse, sehingga diperoleh rata-rata volume aktual sebesar 599,4 mL dengan rata-rata selisih sebesar 2,2 mL. Hasil tersebut menunjukkan bahwa sistem mampu menghasilkan volume air yang sangat mendekati target yang ditentukan.

Berikut uji coba yang dilakukan untuk mendapatkan nilai volume 600 mL pada Tabel 4.3.

Table 4.3 Percobaan Penakaran Air 600 mL

Percobaan	Target (mL)	Aktual (mL)	Selisih (mL)
Ke-1	600 mL	598 mL	2
Ke-2	600 mL	603 mL	3
Ke-3	600 mL	596 mL	4
Ke-4	600 mL	601 mL	1
Ke-5	600 mL	599 mL	1
Rata-rata	600 mL	599,4 mL	2,2

Selanjutnya dilakukan perhitungan persentase error menggunakan rumus yang telah dijelaskan sebagai berikut:

$$\text{Presentase Error} = \frac{\text{Volume Aktual} - \text{Volume Sensor}}{\text{Volume Aktual}} \text{ x } 100\%$$

$$\text{Presentase Error} = \frac{600 - 599{,}4}{600} \text{ x } 100\%$$

$$\text{Presentase Error} = 0{,}1\%$$

Berikut hasil pengukuran yang penulis dapatkan ketika pengujian untuk nilai volume 600 mL pada Gambar 4.44.



Gambar 4.44 Hasil Pengujian Kalibrasi 600 mL

Kemudian penulis melakukan uji coba untuk volume air 1,5 liter dengan nilai kalibrasi sebanyak 2940 pulse, sehingga diperoleh rata-rata volume aktual sebesar 1499,4 mL dengan rata-rata selisih sebesar 3,0 mL. Hasil tersebut menunjukkan bahwa sistem mampu menghasilkan volume air yang sangat mendekati target yang ditentukan.

Berikut uji coba yang dilakukan untuk mendapatkan nilai volume 600 mL pada Tabel 4.4.

Table 4.4 Percobaan Penakaran Air 1,5 Liter

Percobaan	Target (mL)	Aktual (mL)	Selisih (mL)
Ke-1	1,5 liter	1469 mL	4
Ke-2	1,5 liter	1502 mL	2
Ke-3	1,5 liter	1498 mL	2
Ke-4	1,5 liter	1504 mL	4
Ke-5	1,5 liter	1497 mL	3
Rata-rata	1,5 liter	1499,4 mL	3,0

Berdasarkan hasil pengujian volume 1,5 liter, diperoleh rata-rata volume aktual sebesar 1499,4 mL dengan rata-rata selisih sebesar 3 mL dari volume target yang ditentukan sistem.

Kemudian dilakukan perhitungan persentase error untuk voulme 1,5 liter dengan rumus sebagai berikut:

$$\text{Presentase Error} = \frac{\text{Volume Aktual} - \text{Volume Sensor}}{\text{Volume Aktual}} \text{ x } 100\%$$

$$\text{Presentase Error} = \frac{1500 - 1499{,}4}{1500} \text{ x } 100\%$$

$$\text{Presentase Error} = 0{,}4\%$$

Berikut hasil pengukuran yang penulis dapatkan ketika pengujian untuk nilai volume 1500 mL pada Gambar 4.45.



Gambar 4.45 Hasil Pengujian Kalibrasi 1500 mL

Selanjutnya dilakukan uji coba untuk volume air 19 liter dengan nilai kalibrasi sebanyak 36525 pulse, sehingga diperoleh rata-rata volume aktual sebesar 18996 mL dengan rata-rata selisih sebesar 18 mL. Hasil tersebut menunjukkan bahwa sistem mampu menghasilkan volume air yang sangat mendekati target yang ditentukan.

Berikut uji coba yang dilakukan untuk mendapatkan nilai volume 19 liter pada Tabel 4.5.

Table 4.5 Percobaan Penakaran Air 19 liter

Percobaan	Target (mL)	Aktual (mL)	Selisih (mL)
Ke-1	19 liter	18970 mL	30
Ke-2	19 liter	19015 mL	15
Ke-3	19 liter	18990 mL	10
Ke-4	19 liter	19020mL	20
Ke-5	19 liter	18985mL	15
Rata-rata	19 liter	18996 mL	18

Kemudian dilakukan perhitungan persentase error untuk voulme 19 liter dengan rumus sebagai berikut:

$$\text{Presentase Error} = \frac{\text{Volume Aktual} - \text{Volume Sensor}}{\text{Volume Aktual}} \text{ x } 100\%$$

$$\text{Presentase Error} = \frac{19000 - 18996}{19000} \text{ x } 100\%$$

$$\text{Presentase Error} = 0{,}2\%$$

Berikut hasil pengukuran yang penulis dapatkan ketika pengujian untuk nilai volume 19000 mL pada Gambar 4.46.



Gambar 4.46 Hasil Pengujian Kalibrasi 1500 mL

Berdasarkan seluruh pengujian yang telah dilakukan, *sensor water flow* (YF-S401) mampu bekerja dengan baik dalam mengukur volume air pada setiap variasi pengisian. Nilai pulse yang telah ditentukan melalui proses kalibrasi, yaitu 1160 pulse untuk volume 600 mL, 2940 pulse untuk volume

1,5 liter, dan 36525 pulse untuk volume 19 liter, mampu menghasilkan volume air yang mendekati target yang ditentukan oleh sistem.

Persentase *error* yang diperoleh juga relatif kecil, yaitu sebesar 0,1% pada volume 600 mL, 0,04% pada volume 1,5 liter, dan 0,02% pada volume 19 liter. Hasil tersebut menunjukkan bahwa *prototype* sistem pengisian air otomatis yang dikembangkan memiliki tingkat akurasi yang baik dan dapat digunakan untuk mendukung proses pengisian air secara otomatis pada berbasis di Depot Air Minum Bluebub.

e. Pengujian *black box*

Di tahap ini pengujian *Black Box* Testing dilakukan untuk memastikan bahwa seluruh fungsi yang terdapat pada sistem dapat berjalan sesuai dengan kebutuhan yang telah dirancang. Pengujian dilakukan dengan mengamati keluaran sistem berdasarkan masukan yang diberikan tanpa melihat proses atau struktur program yang berjalan di dalam sistem.

Pengujian ini dilaksanakan pada tanggal 25 Mei 2026 dengan pendampingan dosen penguji, yaitu Ir. Abdul Hadi, S.T., M.Kom. Pengujian dilakukan pada tiga bagian utama sistem, yaitu antarmuka pelanggan (*customer side*), antarmuka *administrator* (*admin side*), dan integrasi perangkat keras (*hardware side*).

1. Antarmuka pelanggan (*Customer Side*)

Pada tahap ini dilakukan pengujian terhadap fitur-fitur yang digunakan langsung oleh pelanggan, mulai dari proses autentikasi akun,

pemilihan produk, pembayaran menggunakan *QRIS Xendit*, hingga monitoring proses pengisian air secara *real-time*.

Table 4.6 Pengujian Antarmuka Pelanggan (*Customer Side*)

No	Fungsi Sistem	Skenario	Hasil yang Diharapkan	Hasil
1.	Pengguna (*Login*)	*Input Username* dan *Password* yang tidak terdaftar di *database* lalu klik	Sistem menolak hak akses, tetap di halaman *login*, dan menampilkan pemberitahuan *error*.	Sesuai [√] Tidak Sesuai [ ]
2.	Pengguna *Login Regist*	*Input Username* dan *Password* yang benar dan valid, sesuai degan data *register*.	Sistem berhasil melakukan autentikasi, menyimpan *session*, dan mengarahkan pengguna ke halaman pemenasan.	Sesuai [√] Tidak Sesuai [ ]
3.	Pemilihan Volume Air (Katalog)	Memilih salah satu volume air (600ml / 1,5 Liter / 19 Liter) pada katalog.	Sistem memproses varian, menampilkan detail nama produkm rincian nominal harga, serta mengarahkan ke formulir data pengiriman.	Sesuai [√] Tidak Sesuai [ ]
4.	Integrasi Pembayaran (*Xendit QRIS*)	Mengonfirmasi data pesaan, nama, alamat, lalu menekan tombol bayar (dengan *QRIS Xendit*).	Sistem memanggil *Callback API Xendit* secara *server-side* dan berhasil meng-*generate QRIS* Dinamis yang sesuai nominal di layar	Sesuai [√] Tidak Sesuai [ ]

No	Fungsi Sistem	Skenario	Hasil yang Diharapkan	Hasil
			(*Simulation Mode*).	
5.	Mekanisme Radar Pengisian *Live*	Melakukan simulasi pembayaran *QRIS* hingga status transaksi di *Payment Gateway* berubah menjadi '*Paid*'.	Halaman monitor di HP pelanggan berubah otomatis dari status 'Menunggu Pengisian' menjadi 'Air Sedang Diisi' via *AJAX* (tanpa *refresh* manual).	Sesuai [√] Tidak Sesuai [ ]
6.	Otomatisasi Selesai Transaksi	Menunggu aliran air diselesaikan secara fisik oleh perangkat dispenser pintar.	Begitu mendapat konfirmasi selesai dari server, halaman *web* otomatis menampilkan 'Pengisian Selesai!' lalu me-*redirect* balik pengguna ke menu utama dalam 3 detik.	Sesuai [√] Tidak Sesuai [ ]

Berdasarkan hasil pengujian yang telah dilakukan, seluruh fitur pada sisi pelanggan dapat berjalan sesuai dengan fungsi yang diharapkan. Sistem berhasil melakukan validasi *login*, menampilkan pilihan volume air, menghasilkan *QRIS* pembayaran melalui layanan *Xendit*, serta menampilkan status pengisian secara otomatis setelah pembayaran berhasil dilakukan.

2. Antarmuka *administrator* (*Admin Side*)

Selanjutnya dilakukan pengujian pada sisi *administrator* untuk memastikan bahwa seluruh fitur pengelolaan sistem dapat berfungsi dengan baik. Pengujian meliputi tampilan ringkasan statistik sistem, kendali pompa secara manual, pengaturan nilai pulse sensor, pemantauan aktivitas mesin, serta riwayat transaksi yang diperbarui secara otomatis.

Table 4.7 Pengujian Antarmuka *Administrator* (*Admin Side*)

No	Fungsi Sistem	Skenario	Hasil yang Diharapkan	Hasil
1.	Ringkasan Statistik Sistem (*Cards*)	Membuka halaman utama *Dashboard Administrator* setelah *login* berhasil.	Sistem secara dinamis menghitung dan menampilkan total pendapatan (rupiah), jumlah pesanan transaksi, dan status kesiapan mesin.	Sesuai [√] Tidak Sesuai [ ]
2.	Kendali Pompa Manual (*Override*)	Menekan tombol 'HIDUPKAN KERAN' pada *card* kendali keran manual.	Sistem mengirimkan parameter '*ON*' ke *database* secara latar belakang (*background AJAX*) secara senyap tanpa memunculkan *pop-up* alert yang mengganggu.	Sesuai [√] Tidak Sesuai [ ]
3.	Kendali Pompa Manual (*Override*)	Menekan tombol 'MATIKAN KERAN' pada *card* kendali keran manual setelah dihidupkan.	Sistem mengubah parameter menjadi '*OFF*' di *database* via *AJAX* tanpa *pop-up alert*, dan mesin merespons mati secara instan.	Sesuai [√] Tidak Sesuai [ ]

No	Fungsi Sistem	Skenario	Hasil yang Diharapkan	Hasil
4.	Kalibrasi Presisi Nilai *Pulse*	Mengubah angka batas pulsa sensor *Flowmeter* (misal: 600ml diganti ke 1350) lalu klik tombol 'Simpan'	Sistem memperbarui konfigurasi data pulse dispenser di *database MySQL* secara *real-time* dan memunculkan *log* sukses di *background*.	Sesuai [√] Tidak Sesuai [ ]
5.	Radar Aktivitas Mesin (*Live Alert*)	Sistem menerima *Callback webhook Xendit* saat membayar lunas.	*Card* radar aktivitas di *Dashboard Admin* otomatis berubah Memunculkan status '*REF*: XXX sedang mengisi..'	Sesuai [√] Tidak Sesuai [ ]
6.	Riwayat Transaksi Dinamis	Membiarkan Halaman *dashboard* menyala tanpa melakukan *refresh* manual.	Tabel transaksi memuat ulang data secara asinkron (*.load*) setiap beberapa detik untuk menampilkan penambahan data Nama Pembeli dan Alamat Pengiriman terbaru.	Sesuai [√] Tidak Sesuai [ ]

Berdasarkan hasil pengujian, seluruh fitur *administrator* berhasil dijalankan sesuai dengan skenario yang telah ditentukan. Sistem mampu menampilkan informasi transaksi secara *real-time*, mengendalikan pompa dari *dashboard*, memperbarui konfigurasi pulse sensor, serta menampilkan aktivitas mesin secara langsung tanpa memerlukan *refresh* halaman secara manual.

3. Integrasi perangkat keras (*Hardware Side*)

Pengujian integrasi perangkat keras dilakukan untuk memastikan komunikasi antara *website*, *database*, dan perangkat *ESP32* dapat berjalan dengan baik. Pengujian meliputi konektivitas jaringan , validasi keamanan pengisian menggunakan *sensor proximity* (E18-D80NK), serta eksekusi aktuator pompa air setelah transaksi berhasil diverifikasi.

Table 4.8 Integrasi Perangkat Keras (*Hardware Side*)

No	**Fungsi Sistem**	**Skenario**	**Hasil yang Diharapkan**	**Hasil**
1.	Konektivitas Jaringan	Menyalakan modul ESP32 dan menghubung kannya dengan gateway jaringan *WiFi* lokal.	*ESP32* berhasil mendapatkan IP lokal, melakukan *handshake* dengan server lokal yang dijembatani *Ngrok Tunneling*, dan masuk ke status *Standby*.	Sesuai [√] Tidak Sesuai [ ]
2.	Validasi Sistem Keamanan Fisik	Status transaksi web sudah ‘Paid’, namun petugas belum meletakkan galon di bawah dispenser pintar.	*Sensor Proximity Infrared E18-D80NK* Memberikan output ‘*LOW*’, *ESP32* menahan modul *Relay* agar tetap mati, mencegah air tumpah sia-sia.	Sesuai [√] Tidak Sesuai [ ]
3.	Eksekusi Aktuator Pompa Air	Meletakkan galon tepat di bawah sensor dispenser setelah transaksi terverifikasi.	*Sensor Infrared* mendeteksi pantulan objek (*Output HIGH*), *ESP32* memicu *Modul Relay* 5V menjadi aktif, dan *Water Pump DC*	Sesuai [√] Tidak Sesuai [ ]

No	Fungsi Sistem	Skenario	Hasil yang Diharapkan	Hasil
			12V menyala memompa air.	
4.	Penakaran Volume *Real-Time*	Air mengalir melewati komponen sensor *flowmeter YF-S401* menuju ke dalam galon.	Turbin internal *Flowmeter* berputar, prinsip *Hall Effect* mengubah putaran menjadi pulsa elektrik yang dihitung akumulasinya (*counting pulse*) secara *real-time* oleh *ESP32*.	Sesuai [√] Tidak Sesuai [ ]
5.	Otomatisasi *Cut-Off* Aliran Air	Akumulasi pulsa sensor *flowmeter* mencapai batas kalibrasi volume yang ditentukan (misal: 1350 *pulse* untuk 600 ML).	*ESP32* menghentikan pemicu *Relay* secara seketika, pompa air mati total tepat waktu, takaran presisi, dan status transaksi di-*update* menjadi 'Selesai'.	Sesuai [√] Tidak Sesuai [ ]
6.	Antarmuka Manusia dan Mesin (*HMI*)	Mengamati perubahan data teks pada layar *Modul LCD 16x2 I2C* fisik alat sepanjang siklus kerja.	Layar *LCD* fisik berhasil menampilkan informasi status penalaan secara sinkron: Teks '*Standby...*' saat *idle*, '*Pumping...*' saat air mengalir, dan 'Selesai!' saat tuntas.	Sesuai [√] Tidak Sesuai [ ]

Berikut ketika penulis melalukan pengujian bersama dosen penguji *black box testing* pada Gambar 4.47.



Gambar 4.47 Pengujian *Black Box Testing*

Hasil pengujian menunjukkan bahwa *ESP32* berhasil terhubung dengan jaringan *WiFi* dan server melalui layanan *tunneling* yang digunakan. Selain itu, *sensor proximity* mampu mendeteksi keberadaan wadah sebelum proses pengisian dilakukan sehingga dapat mencegah air keluar ketika wadah belum tersedia. Setelah transaksi dinyatakan berhasil dan wadah terdeteksi, pompa air dapat aktif secara otomatis sesuai dengan perintah yang diterima dari *server*.

f. Pengujian eksternal

Pada pengujian tahap selanjutnya adalah pengujian eksternal atau *User Acceptance Test* (*UAT*). Pengujian ini dilakukan 2 tahap yaitu secara langsung oleh Andri Permana selaku pemilik dan pengelola Depot Air Minum Bluebub dan juga dari sisi customer melalui 10 responden.

1. *User Acceptance Testing* (UAT) oleh Pemilik Depot

Pengujian tahap pertama dilakukan secara langsung oleh Andri Permana selaku pemilik Depot Air Minum Bluebub. Instrumen yang digunakan adalah lembar pengujian *checklist*, di mana responden

memberikan penilaian "Setuju" atau "Tidak Setuju" pada 10 pernyataan terkait antarmuka *web*, fungsi *prototype*, dan efisiensi bisnis.

Berikut tabel aspek dari perangkat lunak dan hasil dari pengujian yang dilakukan langsung dengan pemilik depot pada Tabel 4.9.

Table 4.9 Aspek Dari Perangkat Lunak

No	Pengujian	Setuju	Tidak Setuju
1	Antarmuka *website* pemesanan mudah dipahami dan digunakan oleh pelanggan melalui *smartphone*.	√	
2	Proses pembayaran nirkontak menggunakan *barcode QRIS* berjalan dengan sangat cepat dan praktis.	√	
3	Tampilan halaman *Dashboard Admin* menyajikan data pendapatan dan riwayat transaksi dengan jelas	√	
4	Fitur pemantauan jarak jauh (*Live Radar*) di *website* beroperasi dengan baik sesuai status mesin.	√	

Kemudian dilakukan pengujian dari aspek perangkat keras, berikut tabel pengujian dan hasil yang dilakukan langsung oleh pemilik depot pada tabel 4.10.

Table 4.10 Aspek Dari Perangkat Keras

No	Pengujian	Setuju	Tidak Setuju
1	Sensor pendeteksi wadah berfungsi dengan baik untuk mencegah air tumpah jika tidak ada galon	√	
2	Mesin memproses penakaran air secara otomatis tepat setelah pembayaran terverifikasi di *website*.	√	
3	Pompa air berhasil mati secara otomatis ketika takaran volume air telah sesuai dengan pesanan.	√	

Selanjutnya dilakukan pengujian dari aspek manfaat bisnis dan efisiensi operasional, berikut tabel pengujian dan hasil yang dilakukan langsung oleh pemilik depot.

Table 4.11 Aspek Manfaat Bisnis dan Efisiensi Operasional

No	Pengujian	Setuju	Tidak Setuju
1	Sistem otomasi ini secara nyata mampu memangkas waktu tunggu (*idle time*) pelanggan di depot.	√	
2	Fitur alamat pengantaran pada *website* berpotensi memperluas jangkauan pesanan pelanggan.	√	
3	Secara keseluruhan, sistem ini sangat membantu mempermudah operasional Depot Air Minum Bluebub.	√	

Berdasarkan tabel pengujian di atas, persentase kelayakan dari sisi pemilik depot dihitung menggunakan rumus sebagai berikut.

$$\text{Presentase Kelayakan} = \frac{\text{Total Jawaban "Setuju"}}{\text{Total Pertanyaan}} \text{ x } 100\%$$

$$\text{Presentase Kelayakan} = \frac{10}{10} \text{ x } 100\%$$

$$\text{Presentase Kelayakan} = 100\%$$

Hasil persentase 100% ini menunjukkan bahwa secara keseluruhan fungsi perangkat lunak, perangkat keras, maupun efisiensi bisnis, *prototype* ini memenuhi standar dan layak untuk mendukung operasional Depot Bluebub.

Berikut ini ketika dilakukan pengujian bersama pemilik depot bisa dilihat pada Gambar 4.48.



Gambar 4.48 Pengujian *Prototype* di Lokasi Depot

2. Pengujian oleh pengguna (*Usability Testing*)

Pengujian kedua yaitu pengujian *usability* dilakukan untuk mengetahui tingkat kemudahan penggunaan antarmuka *website* dari sudut pandang pengguna. Pengujian ini dilakukan dengan menyebarkan kuesioner kepada 10 responden yang telah mencoba fitur pemesanan air melalui *website*. Responden yang dilibatkan tidak hanya difokuskan pada pelanggan Depot Air Minum BlueBub, tetapi juga pengguna yang memahami penggunaan aplikasi berbasis *web* sehingga dapat memberikan penilaian terhadap aspek antarmuka dan pengalaman pengguna (*user experience*). Aspek yang dinilai meliputi kemudahan penggunaan *website*, kemudahan pemesanan, kemudahan pembayaran, kejelasan informasi, kecepatan respon sistem, hingga tingkat kepuasan pengguna secara keseluruhan.

Metode yang digunakan dalam pengolahan data kuesioner adalah Skala Likert dengan lima pilihan jawaban, yaitu Sangat Setuju (SS) bernilai 5, Setuju (S) bernilai 4, Netral (N) bernilai 3, Tidak Setuju (TS) bernilai 2, dan

Sangat Tidak Setuju (STS) bernilai 1. Perhitungan dilakukan berdasarkan teori yang dikemukakan oleh (Septiana & Firdonsyah, 2025).

Untuk mendapatkan skor pehitungan kusioner dari setiap jawaban dihitung menggunakan persamaan sebagai berikut.

$$\text{Skor } = T \; x \; Pn$$

Dengan keterangan, *T* yaitu Jumlah Responden yang memilih jawaba tertentu dan *Pn* adalah bobot nilai jawaban. Kemudian seluruh skor pada masing-masing pertanyaan di jumlahkan, dan diperoleh skor keseluruhan sebesar 396.

Berikut hasil rekapitulasi skor dari seluruh pertanyaan bisa dilihat pada Tabel 4.12.

Table 4.12 Tabel Total Hasil Kusioner Dari Responden

No	Pertanyaan	Total Skor	Rata-rata
1	Tampilan *website* mudah dipahami saat pertama kali digunakan.	46	4,6
2	Informasi yang ditampilkan pada *website* mudah dibaca dan dipahami.	45	4,5
3	Proses pemesanan air dapat dilakukan dengan mudah tanpa mengalami kesulitan.	42	4,2
4	*Website* memberikan respon yang cepat saat digunakan.	41	4,1
5	Informasi produk yang tersedia sudah ditampilkan dengan jelas.	44	4,4
6	Proses pembayaran melalui *QRIS* mudah dilakukan melalui *website*.	45	4,5
7	Status pesanan dan proses pengisian air mudah dipantau melalui *website*.	45	4,5

No	Pertanyaan	Total Skor	Rata-rata
8	*Website* membantu mempermudah proses pemesanan air minum dibandingkan cara manual.	43	4,3
9	Secara keseluruhan saya merasa puas menggunakan *website* ini.	45	4,5
Total		396	4,4

Berdasarkan Tabel 4.12, diperoleh total skor sebesar 396 dari seluruh jawaban responden. Kemudian untuk mendapatkan skor maksimum diperoleh dari perkalian jumlah responden, jumlah pertanyaan, dan skor tertinggi pada Skala Likert.

$$\text{X} = \text{Jumlah Responden x Jumlah Pertanyaan x Skor Tertinggi}$$

$$\text{X} = 10 \text{ x } 9 \text{ x } 5$$

$$\text{X} = 450$$

Selanjutnya dilakukan perhitungan persentase *usability* menggunakan rumus indeks presentase Skala Likert, persamaan yang digunakan sebagai berikut.

$$\text{Indeks}(\%) = \frac{\text{Total Skor}}{\text{X}} \text{ x } 100$$

$$\text{Indeks}(\%) = \frac{396}{450} \text{ x } 100$$

$$\text{Indeks}(\%) = 88{,}00\%$$

Kemudian, dilakukan perhitungan untuk menentukan kategori hasil pengujian dan digunakan perhitungan interval Skala Likert sebagai berikut.

$$\text{Interval} = \frac{100}{5}$$

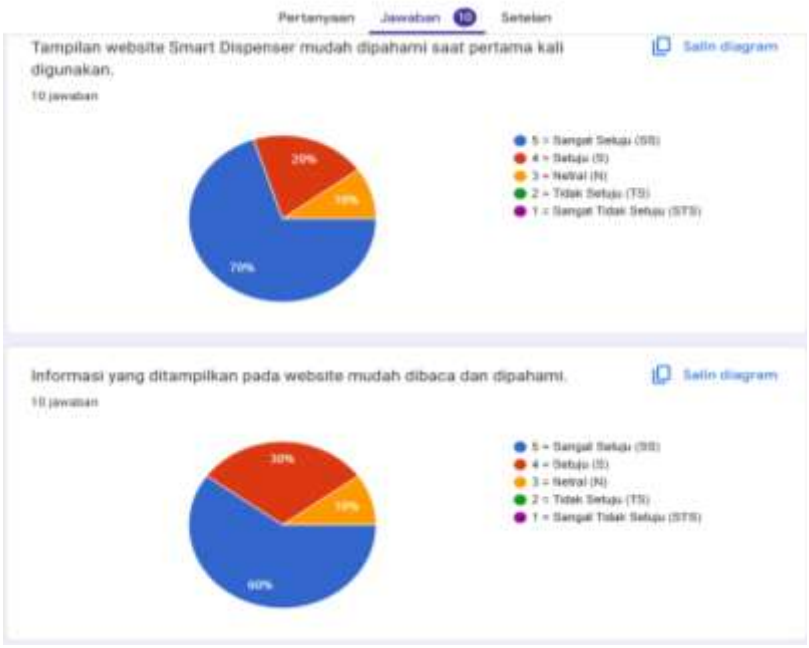
$$\text{Interval} = 20$$

Berdasarkan hasil interval sebesar 20, maka diperoleh interpretasi seperti pada Tabel 4.13.

Table 4.13 Tabel Interval Skala Likert

Presentase	Kategori
0% - 19,99%	Sangat Tidak Baik
20% - 39,99%	Tidak Baik
40% - 59,99%	Cukup
60% - 79,99%	Baik
80% - 100%	Sangat Baik

Berikut ini merupakan form beberapa respon pertanyaan yang diajukan untuk dijawab oleh responden dalam *usability* dan *user experience* dalam penggunaan *website* pada Gambar 4.49.



Gambar 4.49 Google Form Pertanyaan untuk penggunaan *website*

Berdasarkan hasil perhitungan menggunakan Skala Likert tersebut diperoleh nilai *usability* sebesar 88,00%. Nilai tersebut berada pada rentang

80%–100% sehingga termasuk dalam kategori Sangat Baik. Hasil ini menunjukkan bahwa *website* mudah digunakan, mudah dipahami, serta mampu membantu pengguna dalam melakukan pemesanan air minum secara digital. Selain itu, fitur pembayaran *QRIS* dan monitoring status pesanan yang tersedia pada *website* juga dapat digunakan dengan baik oleh pengguna dalam mengetahui status pesanan secara *real-time*.

4.2 Pembahasan

Pada sub-bab ini, penulis akan menguraikan analisis dan evaluasi secara mendalam terhadap hasil implementasi dan pengujian sistem yang telah dipaparkan pada sub-bab sebelumnya. Pembahasan ini berfokus pada evaluasi kinerja program dan jaringan *Internet of Things* (*IoT*), keandalan basis data, interaktivitas antarmuka sistem, serta penerimaan pengguna akhir terhadap sistem yang dibangun.

4.2.1 Pembahasan Program dan Analisis Jaringan

Berdasarkan hasil pengujian fungsionalitas, program keseluruhan telah terintegrasi dengan sangat baik. Namun, karena sistem ini mengadopsi model *Internet of Things* yang bergantung pada komunikasi *server-to-server* (via *Webhook Xendit*) dan *client-to-server* (via ESP32), kelancaran program sangat dipengaruhi oleh stabilitas koneksi dan lalu lintas jaringan (*traffic network*).

Pada penelitian ini, layanan *Ngrok* digunakan sebagai media *tunnelling* untuk menghubungkan server lokal yang berjalan pada *XAMPP* dengan jaringan internet. Melalui mekanisme tersebut, *website* dapat diakses oleh layanan *Xendit* untuk proses *webhook* pembayaran maupun oleh perangkat ESP32 untuk mengambil data

transaksi secara *real-time*. Penggunaan *Ngrok* memungkinkan sistem yang masih berjalan pada lingkungan localhost tetap dapat berkomunikasi dengan perangkat *IoT* dan layanan pihak ketiga tanpa harus menggunakan *hosting* publik.

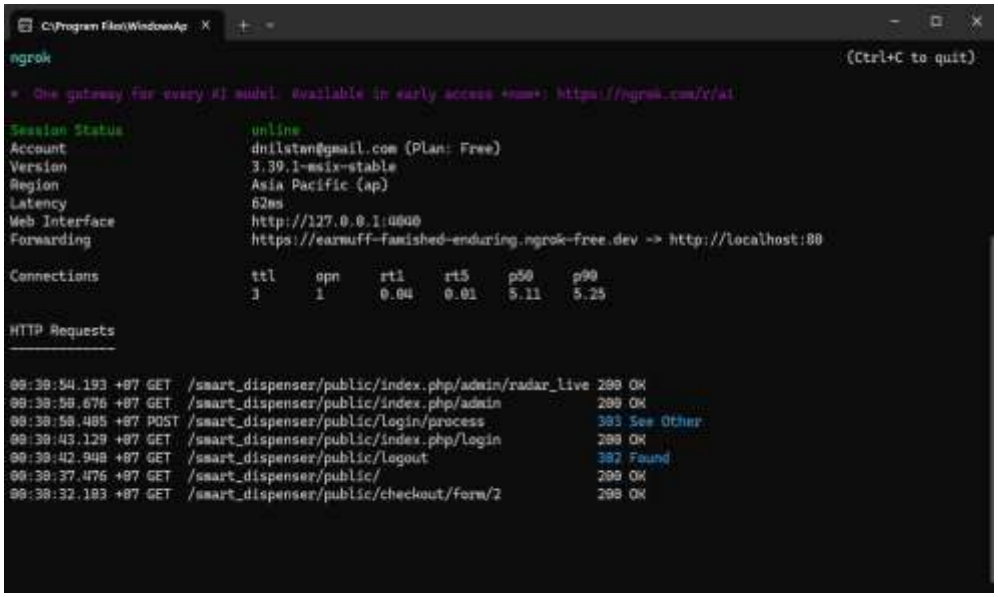
Dalam proses integrasi pembayaran, *Xendit* mengirimkan notifikasi pembayaran secara otomatis melalui mekanisme *webhook* ketika transaksi berhasil diselesaikan oleh pelanggan. Notifikasi tersebut digunakan oleh sistem untuk memperbarui status transaksi menjadi *paid* sehingga dapat diproses oleh perangkat *IoT*. Selain itu, *Xendit* juga menyediakan fitur *Webhook Verification Token* yang dapat digunakan untuk memvalidasi bahwa notifikasi pembayaran benar-benar berasal dari server *Xendit*. Pada penelitian ini, token verifikasi telah tersedia pada *dashboard Xendit* dan dapat diterapkan sebagai pengembangan keamanan sistem pada implementasi selanjutnya untuk meningkatkan keamanan komunikasi data antara *payment gateway* dan server aplikasi.

Dalam operasionalnya, penggunaan layanan *tunnelling Ngrok* terbukti efektif menjembatani *localhost XAMPP* ke internet publik dengan *response time* (latensi) yang cukup stabil di kisaran 60 ms hingga 100 ms. Evaluasi konektivitas menunjukkan bahwa mikrokontroler ESP32 mampu melakukan ping dan menarik data (*HTTP GET*) dari server setiap 3 detik tanpa mengalami *Request Timeout*. Keandalan logika program juga terbukti dari kemampuan ESP32 dalam menerapkan sistem *debounce* (toleransi kedipan sensor selama 3 detik) serta penguncian pelaporan (*anti-spam*), sehingga *traffic* ke *database* tetap terjaga dari beban permintaan (*request*) yang berlebihan.

Selain itu, sistem menerapkan mekanisme validasi sebelum proses pengisian dilakukan. Setelah pembayaran berhasil diverifikasi oleh sistem, proses pengisian tidak langsung dijalankan, melainkan menunggu hingga wadah atau galon terdeteksi oleh sensor *proximity*. Selama wadah belum terdeteksi, sistem akan tetap berada pada kondisi menunggu dan menampilkan instruksi kepada pengguna untuk menempatkan wadah pada area pengisian. Setelah wadah terdeteksi oleh sensor *proximity*, sistem memberikan waktu jeda selama kurang lebih 3 detik sebelum pompa diaktifkan dan proses pengisian dimulai. Mekanisme ini diterapkan untuk mencegah terjadinya pengisian tanpa wadah serta menyesuaikan kondisi operasional yang terjadi di lapangan.

Prioritas pengisian pada sistem ditentukan berdasarkan status pembayaran yang telah berhasil diverifikasi oleh sistem. Transaksi dengan status paid akan masuk ke dalam antrian pengisian dan diproses sesuai urutan transaksi yang telah berhasil menyelesaikan pembayaran.

Berikut proses pemantauan *traffic* data dan catatan waktu respon (*response time*) layanan *Ngrok* ini dapat dilihat pada Gambar 4.50.



Gambar 4.50 *Traffic* Data dan Catatan Waktu Respon dari *tunneling Ngrok*

4.2.2 Analisis Akurasi dan Faktor Penyebab Error Penakaran

Merujuk pada data hasil pengujian fungsionalitas *Water Flow Sensor* (YF-S401) yang telah dijabarkan sebelumnya, sistem penakaran otomatis ini secara umum memiliki tingkat akurasi yang sangat tinggi dengan persentase kesalahan (*error percentage*) yang relatif kecil, yaitu berkisar antara 0,1% hingga 0,4%. Walaupun angka selisih aktualnya hanya berada pada rentang beberapa mililiter saja, munculnya variasi hasil keluaran cairan tersebut tetap perlu dianalisis secara ilmiah.

Berdasarkan pengamatan penulis selama proses pengujian *prorotype* dilakukan, terdapat dua faktor teknis utama yang menjadi penyebab munculnya persentase *error* tersebut, yaitu;

a. Dinamika fluida dan sisa arus cairan pada selang (*Residual Fluid*)

Faktor pertama berasal dari karakteristik mekanis pergerakan air di dalam jalur selang pengeluaran. Sesaat setelah jumlah pulsa akumulasi dari sensor

kincir menyentuh target volume dan mikrokontroler memutuskan arus listrik ke modul *relay*, dinamo pompa air *DC 12V* memang langsung berhenti berputar seketika. Namun, air yang sudah terlanjur dipompa dan berada di dalam saluran selang setelah melewati sensor tidak langsung berhenti secara instan akibat adanya gaya inersia atau sisa tekanan dorong. Tetesan atau sisa aliran air yang tertinggal di ujung selang ini ikut jatuh ke dalam wadah galon tanpa terhitung oleh *sensor flowmeter*. Akibatnya, volume aktual yang diukur menggunakan gelas ukur manual sering kali menunjukkan angka sedikit lebih tinggi ataupun agak kurang dibandingkan volume yang terekam pada pembacaan sistem komputer.

b. Terbentuknya gelembung udara pada aliran sensor (*Air Cavitation*)

Faktor kedua disebabkan oleh aspek internal mekanika fluida yang memicu kesalahan pembacaan sensor turbin. Saat pompa air menghisap cairan dari penampungan utama dengan kecepatan tinggi, sering kali muncul gelembung-gelembung udara kecil di dalam aliran selang akibat adanya turbulensi atau pusaran udara yang ikut terhisap. Ketika kumpulan gelembung udara ini mengalir melewati badan sensor YF-S401, putaran kincir internal berbasis efek *Hall* (*Hall Effect*) dapat menjadi sedikit tidak stabil. Gelembung udara tersebut terkadang membuat kincir berputar lebih cepat tanpa membawa volume cairan yang padat, atau sebaliknya, menyelinap melewati sela-sela kincir turbin tanpa memicu sinyal pulsa elektrik. Inkonsistensi pembacaan sinyal pulsa inilah yang menyebabkan terjadinya deviasi naik-turun pada kalkulasi volume akhir air.

Sebagai langkah optimasi untuk menekan laju kesalahan dari kedua faktor fisik di atas, penulis telah melakukan proses kalibrasi secara berulang demi menemukan nilai batas konversi pulsa yang paling ideal untuk setiap variasi menu seperti, di angka 1160 pulse untuk takaran 600 mL, 2940 pulse untuk 1,5 Liter, dan 36525 pulse untuk 19 Liter. Skema penalaan empiris ini terbukti berhasil mereduksi dampak sisa air selang dan gelembung udara, sehingga prototype dispenser tetap mampu mendistribusikan volume air minum secara konsisten dan aman bagi kebutuhan operasional bisnis komersial.

4.2.3 Pembahasan Basis Data

Struktur basis data *MySQL* yang dirancang terbukti mampu mengelola aliran data operasional secara *real-time* tanpa mengalami *bottleneck*. Tabel utama yaitu *Transaction* secara efektif mengelola dan mengunci status pesanan pelanggan.

Kinerja basis data terlihat paling krusial pada saat menerima eksekusi perubahan status dari *Payment Gateway*. Ketika pelanggan selesai melakukan pembayaran *QRIS*, fungsi *Webhook* seketika memperbarui atribut *payment_status* dari *Pending* menjadi *Paid*. Relasi data ini bekerja sangat efisien, di mana mesin *ESP32* yang terus memantau *database* dapat langsung mendeteksi perubahan status tersebut pada baris data yang tepat berdasarkan *payment_ref*, lalu mengeksekusi aktuator fisik (pompa air) tanpa terjadi tumpang tindih dengan data pesanan pelanggan lainnya.

Berikut ini bentuk struktur rekaman data transaksi yang telah berhasil diperbarui melalui *callback* pihak ketiga ini dapat dilihat pada Gambar 4.51.



Gambar 4.51 Rekaman Data Transaksi yang Berhasil Diperbarui

4.2.4 Pembahasan Antarmuka Program (*Interface*)

Pembahasan pada antarmuka sistem adalah tingkat interaktivitas antara aplikasi web dengan aktivitas fisik di lapangan. Secara keseluruhan, *User Interface* (*UI*) yang dibangun menggunakan *CodeIgniter 4* berhasil menghadirkan *User Experience* (*UX*) yang mulus.

Kelebihan utama dari antarmuka ini adalah penggunaan sinkronisasi dinamis pada halaman Monitoring pelanggan dan *Dashboard Live Radar* milik *administrator*. Pelanggan tidak perlu melakukan *refresh* halaman secara manual setelah membayar; layar secara asinkron berubah menampilkan instruksi "Taruh Wadah" hingga "Selesai" selaras dengan pergerakan mesin pengisi air di depot. Antarmuka pembayaran yang memunculkan barcode *QRIS* dinamis juga dinilai sangat responsif karena mampu merepresentasikan total keranjang belanja tanpa memerlukan campur tangan perhitungan *admin*.

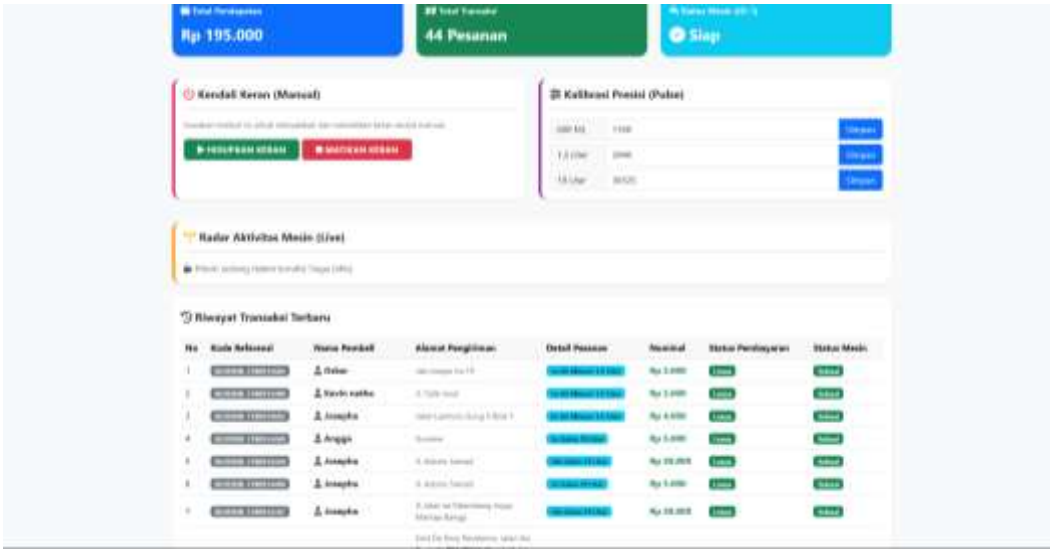
Di sisi lain, karena *Xendit* yang digunakan dalam *testing mode* jadi *barcode QRIS* hanya bisa untuk menampilkan saja tetapi tidak bisa di scan secara langsung,

karena jika ingin mendaftarkan akun *Xendit*, diperlukan syarat yang berstandart bisnis nyata agar akun bisa terverifikasi dengan benar.

Dashboard administrator tidak hanya berfungsi sebagai media pemantauan status mesin, tetapi juga sebagai pusat informasi operasional bagi pemilik depot. Melalui *dashboard* ini, *administrator* dapat memantau data transaksi yang masuk, melihat total pendapatan yang diperoleh, mengetahui status pengisian yang sedang berlangsung, serta melakukan kontrol manual dan kalibrasi volume air apabila diperlukan.

Informasi transaksi dan pendapatan yang ditampilkan pada *dashboard* membantu pemilik depot dalam melakukan pengawasan terhadap aktivitas pemesanan dan pembayaran pelanggan secara lebih terstruktur. Dengan adanya fitur monitoring tersebut, proses pencatatan transaksi tidak lagi dilakukan secara manual karena seluruh data yang berhasil diproses oleh sistem akan ditampilkan secara otomatis pada halaman *administrator*.

Berikut tampilan antarmuka *Dashboard Monitoring* utama pada sisi *administrator* yang menyajikan seluruh ringkasan kontrol dan *traffic* data secara terpusat dapat dilihat pada Gambar 4.52.



Gambar 4.52 Ringkasan Kontrol dan *Traffic* Data pada *Dashboard Admin*

4.2.5 Pembahasan Hasil Kusioner (*Usability Test*)

Berdasarkan hasil pengujian usability yang telah dilakukan terhadap 10 responden, diperoleh nilai *usability* sebesar 88,00%. Nilai tersebut termasuk ke dalam kategori Sangat Baik, yang menunjukkan bahwa *website* dapat diterima dengan baik oleh pengguna.

Hasil kuesioner menunjukkan bahwa sebagian besar responden memberikan penilaian Setuju dan Sangat Setuju terhadap tampilan maupun fungsi *website* yang telah dikembangkan. Pengguna menilai bahwa tampilan *website* mudah dipahami saat pertama kali digunakan, informasi yang ditampilkan dapat dibaca dengan jelas, serta proses pemesanan air dapat dilakukan tanpa mengalami kesulitan yang berarti.

Selain itu, fitur pembayaran menggunakan *QRIS* yang terintegrasi dengan *payment gateway Xendit* juga mendapatkan tanggapan yang positif dari responden. Pengguna dapat melakukan pembayaran secara langsung melalui *website* tanpa harus melakukan konfirmasi pembayaran secara manual kepada pihak depot. Hal

ini menunjukkan bahwa integrasi sistem pembayaran yang diterapkan mampu meningkatkan kemudahan dalam proses transaksi.

Pada aspek monitoring pesanan, responden juga memberikan penilaian yang baik terhadap fitur pelacakan status pengisian air. Melalui fitur tersebut, pengguna dapat mengetahui status pesanan secara *real-time* mulai dari proses pembayaran hingga proses pengisian air pada dispenser. Fitur ini memberikan transparansi informasi kepada pelanggan sehingga proses pemesanan menjadi lebih praktis dibandingkan dengan metode pemesanan konvensional.

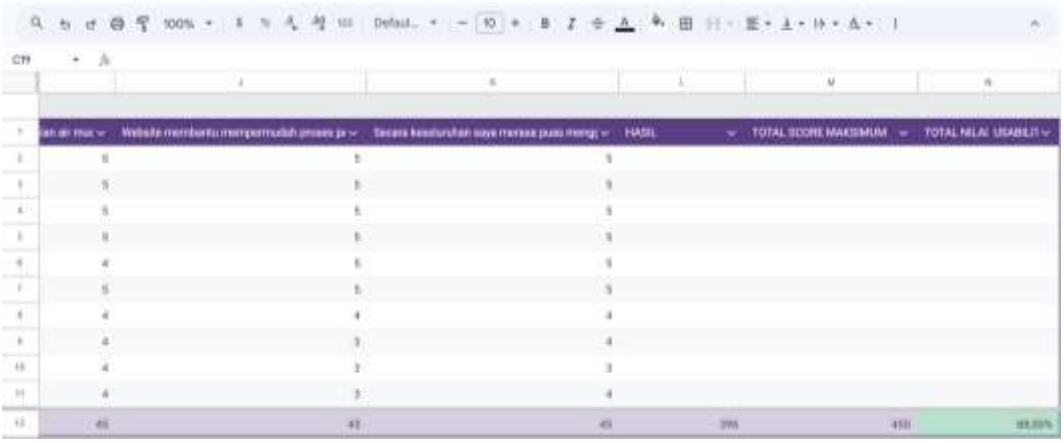
Meskipun memperoleh hasil yang sangat baik, masih terdapat beberapa responden yang memberikan nilai netral dan tidak setuju pada beberapa pertanyaan. Berdasarkan hasil evaluasi, penilaian tersebut umumnya berkaitan dengan waktu respons sistem yang masih bergantung pada kestabilan koneksi internet dan layanan *Ngrok*, serta perlunya penyesuaian pada beberapa bagian antarmuka agar lebih mudah dipahami oleh pengguna yang baru pertama kali menggunakan sistem. Namun demikian, jumlah penilaian netral dan tidak setuju relatif sedikit sehingga tidak mempengaruhi hasil akhir pengujian yang tetap berada pada kategori Sangat Baik dengan nilai sebesar 88,00%.

Pengujian *usability* pada penelitian ini difokuskan pada evaluasi antarmuka *website* dan pengalaman pengguna (*user experience*) selama proses pemesanan air minum. Oleh karena itu, responden yang dilibatkan tidak terbatas pada pelanggan aktif Depot Air Minum Bluebub, tetapi juga pengguna yang memahami penggunaan aplikasi berbasis *web* dan mampu memberikan penilaian terhadap aspek kemudahan penggunaan, navigasi, serta alur interaksi sistem. Pendekatan ini

dipilih karena tujuan utama pengujian adalah menilai tingkat *usability* antarmuka *website* yang telah dikembangkan, sehingga responden yang memahami cara kerja sistem berbasis *web* dinilai telah memenuhi kebutuhan pengujian pada aspek tersebut.

Secara keseluruhan, hasil pengujian *usability* menunjukkan bahwa *website* yang dikembangkan telah mampu memberikan pengalaman penggunaan yang baik, mudah dipahami, dan mendukung proses pemesanan air minum secara digital. Dengan nilai *usability* sebesar 88,00%, *website* dinilai layak digunakan sebagai media pemesanan dan monitoring pada Depot Air Minum Bluebub.

Berikut ini perhitungan menggunakan *Google Spreadsheet* berdasarkan hasil kuesioner yang telah diisi oleh 10 responden. Hasil perhitungan menunjukkan nilai usability sebesar 88,00%. Tampilan hasil pengolahan data dapat dilihat pada Gambar 4.53.



Gambar 4.53 Kusioner *Usability* pada *Google Form Spreadsheets*

4.2.6 Pembahasan Pengujian Oleh Pemilik Depot

Selain pengujian *usability* yang dilakukan terhadap 10 responden, sistem juga diuji secara langsung oleh pemilik Depot Air Minum Bluebub sebagai pengguna

utama dari sistem yang dikembangkan. Pengujian dilakukan dengan mencoba fitur-fitur utama yang tersedia pada *website* maupun perangkat , mulai dari proses pemesanan, pembayaran *QRIS*, monitoring transaksi, hingga proses pengisian air secara otomatis.

Berdasarkan hasil pengujian yang dilakukan, sistem dapat berjalan sesuai dengan kebutuhan operasional depot air minum. Pemilik depot dapat memantau transaksi yang masuk melalui *dashboard admin*, melihat status pembayaran pelanggan secara otomatis, serta melakukan pemantauan terhadap proses pengisian air yang sedang berlangsung. Selain itu, fitur kontrol pompa manual dan pengaturan nilai pulse juga dapat digunakan dengan baik melalui dashboard yang telah disediakan.

Dari sisi operasional, penerapan dinilai mampu membantu proses pelayanan kepada pelanggan karena sebagian proses yang sebelumnya dilakukan secara manual dapat dilakukan secara otomatis oleh sistem. Integrasi antara *website*, *payment gateway Xendit*, *database*, dan perangkat *ESP32* memungkinkan proses pemesanan hingga pengisian air berlangsung secara lebih praktis dan terintegrasi.

Secara keseluruhan, hasil pengujian oleh pemilik Depot Air Minum Bluebub menunjukkan bahwa sistem yang dikembangkan telah sesuai dengan kebutuhan pengguna dan dapat diterapkan sebagai solusi digital dalam proses pemesanan serta pengisian air minum.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan seluruh tahapan perancangan, implementasi, hingga proses pengujian yang telah dilakukan terhadap *prototype* berbasis *Internet of Things* (*IoT*) pada Depot Air Minum Bluebub, maka dapat ditarik beberapa kesimpulan sebagai berikut:

a. *Prototype Smart Dispenser* berbasis *Internet of Things* (*IoT*) pada Depot Air Minum Bluebub berhasil dirancang dan dibangun menggunakan mikrokontroler ESP32, sensor *waterflow*, sensor *proximity*, serta aplikasi web berbasis *CodeIgniter 4*. Seluruh komponen berhasil terintegrasi melalui komunikasi data berbasis HTTP dan format JSON sehingga mampu mendukung proses pemesanan dan pengisian air secara otomatis.
b. Integrasi sistem pembayaran menggunakan *payment gateway* Xendit berhasil diimplementasikan melalui penerbitan *QRIS* dan mekanisme *webhook* pembayaran. Sistem mampu menerima informasi pelunasan transaksi secara otomatis dan mengubah status pesanan menjadi aktif sehingga dapat diproses oleh perangkat *IoT* tanpa memerlukan konfirmasi manual dari administrator.
c. Berdasarkan hasil pengujian, sistem mampu menjalankan proses pengisian air secara otomatis sesuai volume yang dipilih pengguna. Hasil pengujian penakaran menunjukkan tingkat kesalahan yang rendah, yaitu berkisar antara 0,1% hingga 0,4%, sedangkan hasil pengujian eksternal menunjukkan tingkat

penerimaan sistem yang sangat baik dengan nilai 100% dari pemilik depot dan nilai *usability* sebesar 88,00% yang termasuk dalam kategori Sangat Baik.

5.2 Saran

Demi pengembangan dan penyempurnaan sistem *IoT* ini di masa yang akan datang, penulis menyarankan beberapa poin perbaikan sebagai berikut:

a. Pengembangan sistem selanjutnya disarankan untuk memindahkan arsitektur aplikasi *website* dari *local server* komputer ke layanan *Cloud Hosting* dan domain publik yang permanen, guna mengeliminasi ketergantungan terhadap tautan sementara dari *Ngrok*.
b. Disarankan untuk menggunakan komponen sensor *water flow* dan sensor deteksi objek bermutu tinggi dengan material standar industri (seperti kuningan atau *stainless steel standard food-grade*) agar struktur mekanis lebih kokoh dan memiliki ketahanan jangka panjang yang lebih baik terhadap korosi air.
c. Perlu adanya penambahan rancangan sistem proteksi fisik (kotak panel boks *waterproof*) yang lebih aman dan terisolasi untuk membungkus rangkaian mikrokontroler *ESP32* di area dispenser, guna menghindari risiko terkena cipratan air atau kelembapan tinggi yang dapat memicu hubungan arus pendek (*korsleting*).
d. Dapat dikembangkan fitur tambahan berupa modul notifikasi otomatis berbasis *WhatsApp API* atau *SMS Gateway* untuk mengirimkan pesan bukti nota digital secara berkala langsung ke nomor *handphone* pelanggan ketika pengisian air galon di depot telah selesai diproses oleh mesin.

e. Pada pengembangan selanjutnya, sistem dapat ditambahkan fitur pengelolaan transaksi yang lebih lengkap seperti pencarian transaksi, pengaturan prioritas pengisian, serta pembuatan laporan otomatis untuk membantu operasional depot air minum.

f. Untuk meningkatkan akurasi pengukuran volume air, penelitian selanjutnya dapat mempertimbangkan penggunaan komponen *air eliminator* atau perancangan jalur selang yang lebih landai guna meminimalkan terbentuknya gelembung udara (*air cavitation*). Dengan berkurangnya turbulensi udara pada aliran air, pembacaan sensor *waterflow* diharapkan menjadi lebih stabil dan akurat.

g. Penelitian selanjutnya juga dapat mempertimbangkan penggunaan pemberat atau sistem penyaring pada ujung selang hisap sebagaimana prinsip yang digunakan pada *water jet*, sehingga posisi selang tetap berada di dalam air dan dapat mengurangi kemungkinan masuknya udara ke dalam aliran. Dengan demikian, kestabilan aliran air menuju pompa dapat ditingkatkan dan potensi kesalahan pembacaan sensor dapat diminimalkan.

DAFTAR PUSTAKA

Abadi, D. A., Saputra, L. K. P., & Virginia, G. (2021). Smart Water Dispenser Terintegrasi untuk Monitoring Konsumsi Air Minum Harian. *Jurnal Media Informatika Budidarma*, *5*(2), 705. https://doi.org/10.30865/mib.v5i2.2950

Arsy Akbar, M., & Nerisafitra, P. (2025). Penerapan Payment Gateway Menggunakan Metode Webhook Dalam Pengembangan Bot Reservasi Jiot Gadget Solution. *Journal of Informatics and Computer Science*, *06*.

Bella, S. (2023). Implementasi Smart Akuarium Berbasis *Internet Of Things* (Iot) Pada Salma Akuarium Ikan Hias. *Jurnal Teknik Informatika Kaputama (JTIK)*, *7*(2).

Danindra Riski, M., Teknik Pesawat Udara, J., & Penerbangan Surabaya Jl Jemur Andayani, P. I. (2019). *Rancang Alat Lampu Otomatis Di Cargo Compartment Pesawat Berbasis Arduino Menggunakan Push Button Switch Sebagai Pembelajaran Di Politeknik Penerbangan Surabaya*.

Hafizh Andifaisa, M., Pramono, D., & Hanggara, B. T. (2021). *Pembangunan Sistem Informasi Manajemen Toko menggunakan Telegram Bot berbasis Web (Studi Kasus Noor Electric)* (Vol. 5, Number 9). http://j-ptiik.ub.ac.id

Hanayah, N., Ridho, F., Prodi,], Informatika, M., & Medan, G. (2024). Rancang Bangun Aplikasi Monitoring Telkomsel Orbit Witel Medan berbasis WEB menggunakan Framework Codeigniter. In *Jurnal SIKOM (Sistem Informasi Komputer)* (Vol. 1, Number 1).

Kamaluddin, A., Subiyanta, E., & Wachyudin, D. (2025). *Rancang Bangun Wastafel Otomatis Portabel*. *04*(01).

Mikroskil, M. (2012). *Perancangan Sistem Informasi Pembayaran Online Menggunakan Payment Gateway*.

Muhammad Mujab Fachni, Irsyad Rizqi Rochman, Muhammad Koma Amirudin, M.Zidan Al-Farisi, & Fahri Agung Ardiansyah. (2025). *Rancang Bangun Sistem Kontrol Alat Penyiram Tanaman Otomatis Berbasis Arduino Nano Dan Sensor Kelembaban Tanah*.

N Somayasa, D. M., Nursalam, M. N., Taslimah, I. I., Sutiari, D. K., & Muhammad Abidin, dan Z. (2024). *Bakti Cendekia Jurnal Nasional Hasil Penelitian Bidang Multidisiplin Volume 1 Nomor 1, Hal. 1-14*. https://doi.org/10.30598/jnbcxxxxxxxxxxxx

Nurul, Y., 1✉, A., & Mufthi, E. M. (2024). Deteksi Jumlah Konsumsi Air Minum Harian Dengan Smart Dispenser. *Innovative: Journal Of Social Science Research*, *4*, 2993–2999.

Pengendali Berbasis Android Asriyadi, dan, Tafrant, D., Ciksadan, dan, Teknik Elektro, J., Negeri Sriwijaya, P., & Srijaya, J. (2021). *Nomor 2 Oktober 2021 Spektral : Journal Of Communications, Antennas and Propagation Jurusan Teknik Elektro* (Vol. 2).

Putri, A. O., Tohir, T., Ariefka, F., Putra, S., Elektro, J. T., Bandung, N., & 40559, B. (2024). *Prosiding the 15 th Industrial Research Workshop and National Seminar Bandung*.

Ralda, M. I., Ode, L., Barata, A., Endriatno, N., & Kunci, K. (2022). *Piston: Jurnal Teknologi Dispenser Otomatis Berbasis Arduino Uno*. https://dx.doi.org/10.xx/xx.xyz

Revivaldy Mucthar, F., Wibowo, S. A., & Ariwibisono, F. X. (2020). Penerapan Iot (Internet Of Thing) Terhadap Rancang Bangun Sangkar Burung Pintar Untuk Burung Teriep. In *Jurnal Mahasiswa Teknik Informatika*.

Sains, J., Tekonologi, dan, Mahdiania, D., Alfitri Lubis, I., Taufik Al Afkari Siahaan, A., Kunci, K., & Sistem, P. (2022). *Yayasan Insan Cipta Medan Pendaftaran Wasit Berbasis Website Menggunakan Php Dan Mysql Pada Kantor Dinas Pemuda Dan Olahraga Kota Medan*.

Setyawan, R. A., & Walter, F. (2018). Pengukuran Usability *Website* E-Commerce Sambal Nyoss Menggunakan Metode Skala Likert. In *Atapukan* (Vol. 54, Number 1).

Siahaan, R. A., & Sianturi, R. A. (2024). Analisis Perbandingan Payment Gateway untuk Sistem Pembayaran Berbasis Aplikasi dengan Comparative Study. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *11*(2), 291–296. https://doi.org/10.25126/jtiik.20241127680

Suprapto, S. S., & Kusuma, V. A. (2022). Pengukuran Waktu Tunggu Verifikasi Transaksi Digital Otomatis Berbasis Quick Response Code Indonesian Standard (QRIS) Dynamic pada Rancang Bangun Vending Machine. *SPECTA Journal of Technology*, *6*(1). https://doi.org/10.35718/specta.v6i1

Tafonao, F., & Nadia Raslina, N. (2025). Perancangan *website* e-commerce untuk fayucom dengan integrasi payment gateway xendit menggunakan html, css, dan javascript. *Jurnal Sosial Dan Teknologi (SOSTECH), 5*.

Wahanie, G. D., Lim, R., & Sugiarto, I. (2023). *(Relevan 1) Pembayaran meteran Listrik berbasis* .

Wijaya, K., & Ceng Giap, Y. (2024). Dispenser Otomatis Berbasis Mikrokontroler Dengan Led Informasi Dan Sensor Suhu. In *Jurnal Mahasiswa Teknik Informatika* (Vol. 8, Number 6).

LAMPIRAN

SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER
(STMIK) PALANGKARAYA
Jl. G. Obos No.114 Telp.0536-3224593, 3225515 Fax.0536-3225515 Palangka Raya
email : humas@stmikplk.ac.id - website : www.stmikplk.ac.id

SURAT TUGAS PEMBIMBING TUGAS AKHIR

NO. 416/STMIK-3.C.2/AU/III/2026

Ketua Program Studi Teknik Informatika Sekolah Tinggi Manajemen Informatika dan Komputer (STMIK) Palangkaraya menugaskan nama-nama tersebut di bawah ini :

1. Nama : Herkules, S.Kom., M.Cs
   NIK : 198510042010106
   Sebagai Pembimbing I dalam **PEMBUATAN PROGRAM**
2. Nama : Elia Zakharia, M.T
   NIK : 199205262016104
   Sebagai Pembimbing II dalam **PENULISAN TUGAS AKHIR**

Untuk membimbing Tugas Akhir mahasiswa :

Nama	:	Daniel Setiawan Purnomo
NIM	:	C2255201107
Program Studi	:	TEKNIK INFORMATIKA (55201)
Tanggal Daftar	:	24/02/2026
Judul Tugas Akhir	:	Rancang Bangun Smart Dispenser Berbasis IoT Menggunakan Xendit Pada Depot Air Minum Bluebub

Demikian surat ini dibuat agar dapat dipergunakan sebagaimana mestinya, dan dilaksanakan dengan penuh tanggung jawab.

Palangka Raya, 02 Maret 2026

Ketua Program Studi,



02 Maret 2026

Ir. Abdul Hadi, S.T., M.Kom
NIK. 198505102021101

TEMBUSAN :
1. Pembimbing I dan II
2. Mahasiswa yang bersangkutan
3. Arsip Prodi TI

SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER
(STMIK) PALANGKARAYA
Jl. G. Obos No.114 Telp.0536-3224593, 3225515 Fax.0536-3225515 Palangkaraya
email : humas@stmikplk.ac.id - website : www.stmikplk.ac.id

KARTU KEGIATAN KONSULTASI TUGAS AKHIR

Nama Mahasiswa : Daniel Setiawan Purnomo
NIM : C2255201107
Tanggal Persetujuan Judul : 29 Februari 2026
Judul Tugas Akhir : Rancang Bangun Smart Dispenser Berbasis IoT Menggunakan Xendit Pada Depot Air Minum Blubub

No.	Tanggal Konsultasi		Uraian	Tanda Tangan
	Terima	Kembali		
1.	2/4/2026		Bimbingan judul dan latar belakang	
2.	6/4/2026		Penyesuaian pengisian teori dan konsepnya	
3.	8/4/2026		Pembahasan desain proses dan desain perangkat	
4.	11/4/2026	11/26	Acc Sumun	
5.	29/3/2026		Merapikan baris halaman Bab 1 dan 2	
6.	30/3/2026		Tambahkan gambar pada teori di Bab 2	
7.	9/4/2026		Tambahkan desain dan rapikan baris halaman pada Bab 3	
8.	11/4/2026	11/26	Acc Seminar	
9.	4/6/2026		Membahas Struktur Narasi pada Bab IV	
10.	5/6/2026		Menambahkan Gambar pada bab IV Sub bab implentasi	

Dipindai dengan CamScanner

No.	Tanggal Konsultasi		Uraian	Tanda Tangan
	Terima	Kembali		
11	8/6/2026		Sesuaikan desain pada Bab III dengan desain yang sudah jadi	
12	10/6/2026	10/6/2026	Acc Sidang	
13	4/6/2026		Pembahasan struktur pengisian narasi pada Bab 4	
14	6/6/2026		Penyesuaian teori pada bab II	
15	8/6/2026		Pada Bab III gambar desain kasar web di sesuaikan dengan desain web yang sudah jadi	
16	10/6/2026	10/6/2026	Acc Sidang	

Menyetujui :

Dosen Pembimbing I,

Hardiyo

Dosen Pembimbing II,

Eka

Catatan :
Mahasiswa wajib melaksanakan minimal 16 kali bimbingan, dengan ketentuan setiap pembimbing (I dan II) melakukan sedikitnya 4 kali bimbingan sebelum seminar proposal (jumlah 8 bimbingan) dan 4 kali bimbingan sebelum sidang tugas akhir (jumlah 8 bimbingan).

Dipindai dengan CamScanner

SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER
(STMIK) PALANGKARAYA
Jl. G. Obos No.114 Telp.0536-3224593, 3225515 Fax.0536-3225515 Palangkaraya
email : humas@stmikplk.ac.id - website : www.stmikplk.ac.id

Nomor : ...136/STMIK-.C2/...AK/III/2026
Lampiran : -
Perihal : Permohonan Izin Penelitian dan Pengumpulan Data untuk Tugas Akhir

Kepada
Yth. **Pemilik Depot Air Minum Bluebub**

Jl. Rajawali Km.3
Palangka Raya

Dengan hormat,

Sehubungan dengan penyusunan Tugas Akhir mahasiswa sebagai persyaratan kelulusan Program Studi Teknik Informatika (S1) pada STMIK Palangkaraya, maka dengan ini kami sampaikan permohonan izin penelitian dan pengumpulan data bagi mahasiswa kami berikut:

Nama	:	DANIEL SETIAWAN PURNOMO
NIM	:	C2255201107
Prodi (Jenjang)	:	Teknik Informatika (S1)
Thn. Akad. (Semester)	:	2025/2026 (8)
Lama Penelitian	:	13 Maret 2026 s.d 13 April 2026
Tempat Penelitian	:	Depot Air Minum Bluebub

Dengan judul Tugas Akhir:

RANCANG BANGUN SMART DISPENSER BERBASIS IOT MENGGUNAKAN XENDIT PADA DEPOT AIR MINUM BLUEBUB

Adapun ketentuan dan aturan pemberian informasi dan data yang diperlukan dalam penelitian tersebut menyesuaikan dengan ketentuan/peraturan pada instansi Bapak/Ibu.

Demikian permohonan ini disampaikan, atas perhatian dan kerja samanya diucapkan terima kasih.

Palangka Raya, 13 Maret 2026
Ketua,

Ir. Hj. Siti Maryamah, MM.
NIK. 196902021995004

Dipindai dengan CamScanner

LEMBAR OBSERVASI

Nama : Daniel Setiawan Purnomo

NIM : C2255201107

Lokasi Penelitian : Depot Air Minum Bluebub Palangka Raya

Observasi dilakukan langsung ketika penjaga depot air minum melayani pelanggan saat pembelian untuk pengisian air galon. Berikut ini merupakan hasil observasi dari objek yang diamati:

No	Objek Yang Diamati	Hasil Pengamatan
1	Sistem Pemesanan Air	Pemesanan masih bersifat konvensional, pelanggan harus datang langsung secara fisik ke lokasi depot.
2	Sistem Transaksi Pembayaran	Pembayaran sepenuhnya menggunakan uang tunai (*cash*), jadi masih sering muncul risiko masalah uang kembalian yang harus di tukar dulu ke bank ketika tidak ada uang kecil.
3	Proses Pengisian dan Penakaran Air	Pengisian air ke dalam galon masih dilakukan secara manual oleh petugas, sehingga volume takaran air tidak memiliki standar otomatis dan presisi.
4	Mekanisme Antrean Pelanggan	Pelanggan harus membawa galon kosong sendiri dan menghabiskan waktu untuk menunggu antrean di lokasi.

Palangka Raya. 9 Maret 2026

Pemiliki Depot Air Minum Bluebub,

Andri Permana

Dipindai dengan CamScanner

LEMBAR WAWANCARA

Narasumber

Nama : Andri Permana

Pewawancara

Nama : Daniel Setiawan Purnomo

NIM : C2255201107

Hari/Tanggal : Senin, 9 Maret 2026

Lokasi : Depot Air Minum Bluebub Palangka Raya

Berikut ini merupakan hasil wawancara dengan pemilik depot untuk mengetahui apa saja yang mejadi masalah yang terjadi:

No	Pertanyaan	Jawaban Narasumber
1	Bagaimana proses pencatatan penjualan air galon, apakah terkendala seperti selisih uang saat menyiapkan kembalian atau saat pengisian galon yang kurang presisi?	Pencatatannya menggunakan aplikasi di playstore dengan langganan tahunan karena bisnis ini sudah lama, untuk masalah uang terkadang cukul kerepotan juga jika kembaliannya tidak ada uang kecil jadi harus menukar dulu ke bank. Harapannya punya website pribadi sendiri untuk mengelola bisnis ini dari pencatatan uang dan bisa manage air agar tidak terbuang sia sia jika tidak di awasi.
2	Apakah pelanggan disini sering menanyakan opsi pembayaran non-tunai seperti transfer atau qris?	Lumayan banyak untuk pelanggan yang menanyakan hal itu, terkadang kalau ada yang tidak bawa uang tunai, mereka minta di transfer saja, tetapi kendalanya jika lagi banyak yang orderan lain cukup susah juga untuk melayani pelanggan yang minta ditransfer satu persatu karena harus memberikan nomor rekening lagi.
3	Bagaimana cara petugas untuk menakar air saat pengisian, apakah sering terjadi pengisian yang tidak presisi dan tumpah?	Ya, itu sering terjadi kalau sedang banyak orderan, kadang saat *handle* 1 galon untuk di cuci dan galon lain yang sedang diisi ditinggal, kadang suka tumpah.
4	Apakah tertarik kalau ada alat yang bisa melakukan pengisian air otomatis di depot ini, saat pelanggan melakukan order secara online dengan pembayaran qris?	Sangat tertarik jika ada alat pengisian secara otomatis seperti itu, karena memudahkan pekerjaan di depot sini juga, dengan alat seperti itu pengisian akan menjadi lebih

Dipindai dengan CamScanner

		presisi dan pelanggan juga tidak lagi menunggu lama jika memesan air.
5	Apa ke khawatiran jika ada alat otomatis seperti itu dipasang di depot ini?	Kekhawatirannya lebih ke SOP dan adaptasi pelanggan di awal. Walaupun pesannya bisa dari rumah lewat website, pelanggan pasti butuh diedukasi dulu cara pakainya, apalagi cara bayar pakai QRIS ini. Selain itu, saya tetap butuh karyawan yang standby untuk merespons pesanan masuk di website, kemudian meletakkan galon ke mesin dispenser, sampai mengantarkannya ke rumah pelanggan pakai sistem tukar galon.

Palangka Raya, 9 Maret 2026

Pemlik Depot Air Minum Bluebub,

Andri Permana

Dipindai dengan CamScanner

Lembar Dokumentasi Wawancara



LEMBAR BERITA ACARA PENGUJIAN *BLACK BOX TESTING*

Pada Senin, 25 Mei 2026 dilakukan uji coba *black box testing* dalam penyusunan tugas akhir.

Nama : Daniel Setiawan Purnomo

NIM : C2255201107

Jurusan : Teknik Informatika

Judul Tugas Akhir : Rancang Bangun *Smart Dispenser* Berbasis *IoT* Menggunakan *Xendit* Pada Depot Air Minum Bluebub.

1. PENGUJIAN ANTARMUKA WEB PELANGGAN (*CUSTOMER SIDE)*

No	Fitur/Fungsi Sistem	Skenario/Prosedur Pengujian	Hasil yang Diharapkan	Hasil
1.	Autentikasi Pengguna (*Login*)	*Input Username* dan *Password* yang tidak terdaftar di *database* lalu klik	Sistem menolak hak akses, tetap di halaman *login*, dan menampilkan pemberitahuan *error*.	Sesuai [✓] Tidak Sesuai [ ]
2.	Autentikasi Pengguna (*Login Regist*)	*Input Username* dan *Password* yang benar dan valid, sesuai degan data *register*.	Sistem berhasil melakukan autentikasi, menyimpan *session*, dan mengarahkan pengguna ke halaman pemenasan.	Sesuai [✓] Tidak Sesuai [ ]
3.	Pemilihan Volume Air (Katalog)	Memilih salah satu volume air (600ml / 1,5 Liter / 19 Liter) pada katalog.	Sistem memproses varian, menampilkan detail nama produkm rincian nominal harga, serta mengarahkan ke formulir data pengiriman.	Sesuai [✓] Tidak Sesuai [ ]
4.	Integrasi Pembayaran (*Xendit QRIS*)	Mengonfirmasi data pesaan, nama, alamat, lalu menekan tombol bayar (dengan *QRIS Xendit*).	Sistem memanggil *Callback API Xendit* secara *server-side* dan berhasil meng-*generate QRIS* Dinamis yang sesuai nominal di layar (*Simulation Mode*).	Sesuai [✓] Tidak Sesuai [ ]
5.	Mekanisme Radar Pengisian *Live*	Melakukan simulasi pembayaran *QRIS* hingga status transaksi di *Payment Gateway*	Halaman monitor di HP pelanggan berubah otomatis dari status 'Menunggu Pengisian' menjadi 'Air Sedang Diisi' via *AJAX* (tanpa *refresh* manual).	Sesuai [✓] Tidak Sesuai [ ]

Dipindai dengan CamScanner

No	Fitur/Fungsi Sistem	Skenario/Prosedur Pengujian	Hasil yang Diharapkan	Hasil
		berubah menjadi '*Paid*'.		
6.	Otomatisasi Selesai Transaksi	Menunggu aliran air diselesaikan secara fisik oleh perangkat dispenser pintar.	Begitu mendapat konfirmasi selesai dari server, halaman *web* otomatis menampilkan 'Pengisian Selesai!' lalu me-*redirect* balik pengguna ke menu utama dalam 3 detik.	Sesuai [V] Tidak Sesuai [ ]

2. PENGUJIAN ANTARMUKA DASHBOARD ADMINISTRATOR (*ADMIN SIDE*)

No	Fitur/Fungsi Sistem	Skenario/Prosedur Pengujian	Hasil yang Diharapkan	Hasil
1.	Ringkasan Statistik Sistem (*Cards*)	Membuka halaman utama *Dashboard Administrator* setelah *login* berhasil.	Sistem secara dinamis menghitung dan menampilkan total pendapatan (rupiah), jumlah pesanan transaksi, dan status kesiapan mesin.	Sesuai [V] Tidak Sesuai [ ]
2.	Kendali Pompa Manual (*Override*)	Menekan tombol 'HIDUPKAN KERAN' pada *card* kendali keran manual.	Sistem mengirimkan parameter '*ON*' ke *database* secara latar belakang (*background AJAX*) secara senyap tanpa memunculkan *pop-up* alert yang mengganggu.	Sesuai [V] Tidak Sesuai [ ]
3.	Kendali Pompa Manual (*Override*)	Menekan tombol 'MATIKAN KERAN' pada *card* kendali keran manual setelah dihidupkan.	Sistem mengubah parameter menjadi '*OFF*' di *database* via *AJAX* tanpa *pop-up alert*, dan mesin merespons mati secara instan.	Sesuai [V] Tidak Sesuai [ ]
4.	Kalibrasi Presisi Nilai *Pulse*	Mengubah angka batas pulsa sensor *Flowmeter* (misal: 600ml diganti ke 1350) lalu klik tombol 'Simpan'	Sistem memperbarui konfigurasi data pulse dispenser di *database MySQL* secara *real-time* dan memunculkan *log* sukses di *background*.	Sesuai [V] Tidak Sesuai [ ]

Dipindai dengan CamScanner

No	Fitur/Fungsi Sistem	Skenario/Prosedur Pengujian	Hasil yang Diharapkan	Hasil
5.	Radar Aktivitas Mesin (*Live Alert*)	Sistem menerima *Callback webhook Xendit* saat pelanggan membayar lunas.	*Card* radar aktivitas di *Dashboard Admin* otomatis berubah Memunculkan status '*REF*: XXX sedang mengisi...'.	Sesuai [V] Tidak Sesuai [ ]
6.	Riwayat Transaksi Dinamis	Membiarkan Halaman *dashboard* menyala tanpa melakukan *refresh* manual.	Tabel transaksi memuat ulang data secara asinkron (*.load*) setiap beberapa detik untuk menampilkan penambahan data Nama Pembeli dan Alamat Pengiriman terbaru.	Sesuai [V] Tidak Sesuai [ ]

3. PENGUJIAN INTEGRASI PERANGKAT KERAS *IOT* (*HARDWARE SIDE*)

No	Fitur/Fungsi Sistem	Skenario/Prosedur Pengujian	Hasil yang Diharapkan	Hasil
1.	Konektivitas Jaringan *IoT*	Menyalakan modul ESP32 dan menghubung kannya dengan gateway jaringan *WiFi* lokal.	*ESP32* berhasil mendapatkan IP lokal, melakukan *handshake* dengan server lokal yang dijembatani *Ngrok Tunneling*, dan masuk ke status *Standby*.	Sesuai [V] Tidak Sesuai [ ]
2.	Validasi Sistem Keamanan Fisik	Status transaksi web sudah 'Paid', namun petugas belum meletakkan galon di bawah dispenser pintar.	*Sensor Proximity Infrared E18-D80NK* Memberikan output '*LOW*', *ESP32* menahan modul *Relay* agar tetap mati, mencegah air tumpah sia-sia.	Sesuai [V] Tidak Sesuai [ ]
3.	Eksekusi Aktuator Pompa Air	Meletakkan galon tepat di bawah sensor dispenser setelah transaksi terverifikasi.	*Sensor Infrared* mendeteksi pantulan objek (*Output HIGH*), *ESP32* memicu *Modul Relay* 5V menjadi aktif, dan *Water Pump DC 12V* menyala memompa air.	Sesuai [V] Tidak Sesuai [ ]

Dipindai dengan CamScanner

No	Fitur/Fungsi Sistem	Skenario/Prosedur Pengujian	Hasil yang Diharapkan	Hasil
4.	Penakaran Volume *Real-Time*	Air mengalir melewati komponen sensor *flowmeter YF-S401* menuju ke dalam galon.	Turbin internal *Flowmeter* berputar, prinsip *Hall Effect* mengubah putaran menjadi pulsa elektrik yang dihitung akumulasinya (*counting pulse*) secara *real-time* oleh *ESP32*.	Sesuai [✓] Tidak Sesuai [ ]
5.	Otomatisasi *Cut-Off* Aliran Air	Akumulasi pulsa sensor *flowmeter* mencapai batas kalibrasi volume yang ditentukan (misal: 1350 *pulse* untuk 600 ML).	*ESP32* menghentikan pemicu *Relay* secara seketika, pompa air mati total tepat waktu, takaran presisi, dan status transaksi di-*update* menjadi 'Selesai'.	Sesuai [✓] Tidak Sesuai [ ]
6.	Antarmuka Manusia dan Mesin (*HMI*)	Mengamati perubahan data teks pada layar *Modul LCD 16x2 I2C* fisik alat sepanjang siklus kerja.	Layar *LCD* fisik berhasil menampilkan informasi status penalaan secara sinkron: Teks '*Standby...*' saat *idle*, '*Pumping...*' saat air mengalir, dan 'Selesai!' saat tuntas.	Sesuai [✓] Tidak Sesuai [ ]

Palangka Raya, 25 Mei 2026
Dosen Penguji,

Ir. Abdul Hadi, S.T., M.Kom.
NIK. 197502232009002

Dipindai dengan CamScanner

LEMBAR PENGUJIAN EKSTERNAL

Nama : Daniel Setiawan Purnomo

NIM : C2255201107

Lokasi Pengujian : Depot Air Minum Bluebub

Berilah tanda centang pada kolom penilaian yang paling sesuai dengan pendapat Pemilik Depot, setelah mencoba langsung sistem *website* dan *prototype* mesin *Smart Dispenser*.

Aspek Dari Perangkat Lunak (*Website* dan Pembayaran)

No	Pengujian	Setuju	Tidak Setuju
1	Antarmuka *website* pemesanan mudah dipahami dan digunakan oleh pelanggan melalui *smartphone*.	✓	
2	Proses pembayaran nirkontak menggunakan *barcode* QRIS berjalan dengan sangat cepat dan praktis.	✓	
3	Tampilan halaman *Dashboard* Admin menyajikan data pendapatan dan riwayat transaksi dengan jelas.	✓	
4	Fitur pemantauan jarak jauh (*Live Radar*) di *website* beroperasi dengan baik sesuai status mesin.	✓	

Aspek Dari Perangkat Keras (Fungsi *Prototype* Alat)

No	Pengujian	Setuju	Tidak Setuju
1	Sensor pendeteksi wadah berfungsi dengan baik untuk mencegah air tumpah jika tidak ada galon.	✓	
2	Mesin memproses penakaran air secara otomatis tepat setelah pembayaran terverifikasi di *website*.	✓	
3	Pompa air berhasil mati secara otomatis ketika takaran volume air telah sesuai dengan pesanan.	✓	

Dipindai dengan CamScanner

Aspek Manfaat Bisnis dan Efisiensi Operasional

No	Pengujian	Setuju	Tidak Setuju
1	Sistem otomasi ini secara nyata mampu memangkas waktu tunggu (*idle time*) pelanggan di depot.	✓	
2	Fitur alamat pengantaran pada *website* berpotensi memperluas jangkauan pesanan pelanggan.	✓	
3	Secara keseluruhan, sistem ini sangat membantu mempermudah operasional Depot Air Minum Bluebub.	✓	

Palangka Raya, 7 Juni 2026

Pemilik Depot Air Minum Bluebub

Andri Permana

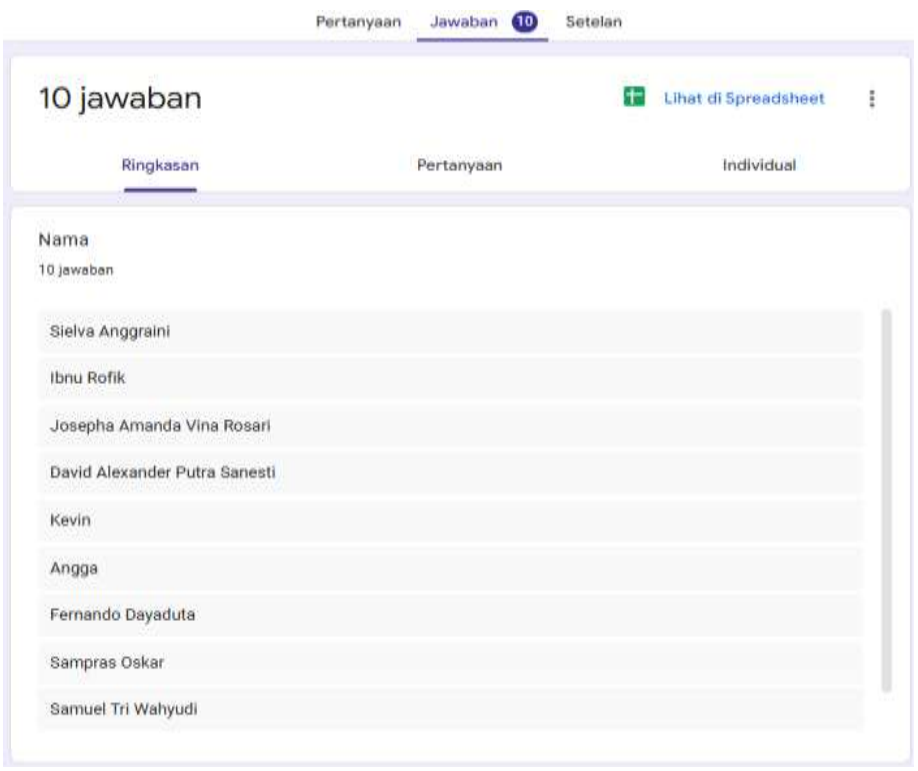
Dipindai dengan CamScanner

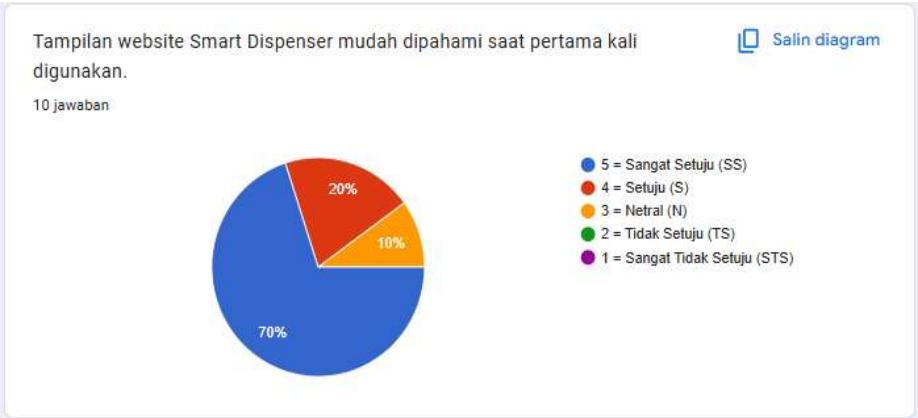
LEMBAR DOKUMENTASI PENGUJIAN EKSTERNAL (PEMILIK DEPOT)

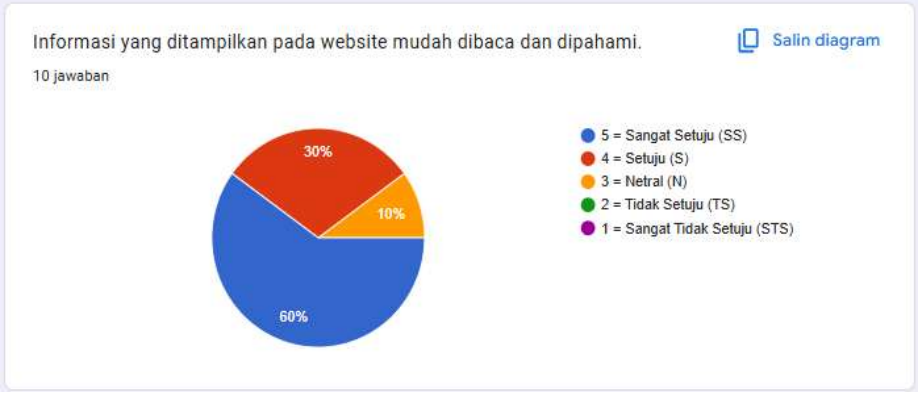




HASIL KUSIONER (USABILITY TESTING)



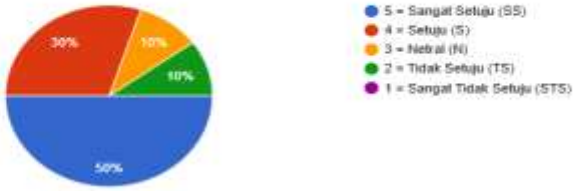




Proses pemesanan air dapat dilakukan dengan mudah tanpa mengalami kesulitan.

Salin diagram

10 jawaban



Website memberikan respon yang cepat saat digunakan.

Salin diagram

10 jawaban

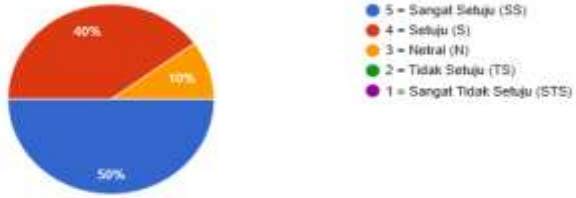


Pertanyaan Jawaban 10 Setelan

Informasi produk yang tersedia sudah ditampilkan dengan jelas.

Salin diagram

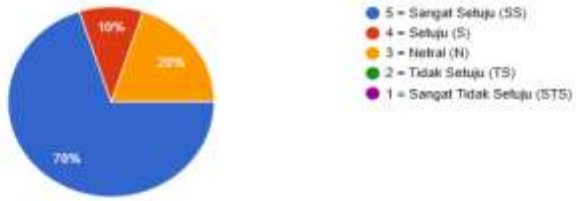
10 jawaban



Proses pembayaran melalui QRIS mudah dilakukan melalui website.

Salin diagram

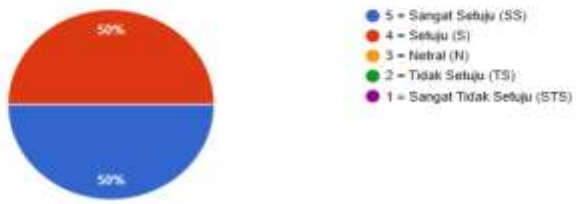
10 jawaban



Status pesanan dan proses pengisian air mudah dipantau melalui website.

Salin diagram

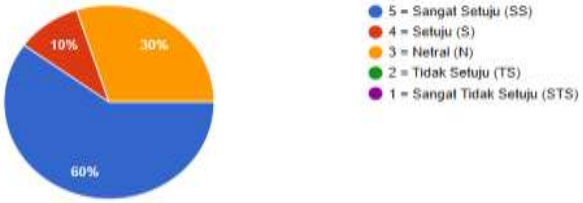
10 jawaban



Website membantu mempermudah proses pemesanan air minum dibandingkan cara manual.

Salin diagram

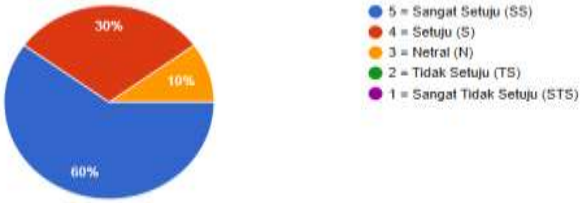
10 jawaban



Secara keseluruhan saya merasa puas menggunakan website Smart Dispenser ini.

Salin diagram

10 jawaban



SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER
(STMIK) PALANGKARAYA
Jl. G. Obos No.114 Telp.0536-3224593, 3225515 Fax.0536-3225515 Palangka Raya
email : humas@stmikplk.ac.id - website : www.stmikplk.ac.id

SURAT TUGAS
PENGUJI SEMINAR PROPOSAL TUGAS AKHIR

465/STMIK-3.C.2/AU/IV/2026

Ketua Program Studi Teknik Informatika Sekolah Tinggi Manajemen Informatika dan Komputer (STMIK) Palangkaraya menugaskan kepada nama-nama berikut :

1. Nama : Hotmian Sitohang, M.Kom.
   NIK : 198503282008002
   Sebagai Ketua
2. Nama : Herkules, S.Kom., M.Cs
   NIK : 198510042010106
   Sebagai Sekretaris
3. Nama : Elia Zakharia, M.T
   NIK : 199205262016104
   Sebagai Anggota

Untuk menguji seminar proposal tugas akhir mahasiswa :

Nama Mahasiswa	:	DANIEL SETIAWAN PURNOMO
N I M	:	C2255201107
Hari / Tanggal Ujian	:	Rabu, 22 April 2026
Waktu	:	08.00 WIB
Ruangan	:	Seminar 1
Judul TA	:	Rancang Bangun Smart Dispenser Berbasis IoT dengan Xendit pada Depot Air Minum Bluebub

Demikian surat ini dibuat agar dapat dipergunakan sebagaimana mestinya dan dilaksanakan dengan penuh tanggung jawab.

Palangka Raya, 13 April 2026

Ketua Program Studi Teknik Informatika,

13 April 2026

Ir. Abdul Hadi, S.T., M.Kom
NIK. 198505102021101

Tembusan :
1. Dosen Penguji
2. Mahasiswa yang Bersangkutan
3. Arsip

LISTING PROGRAM

app/Config/ App.php

```php
<?php

namespace Config;

use CodeIgniter\Config\BaseConfig;

class App extends BaseConfig
{
    /**
     * --------------------------------------------------------------------
     * Base Site URL
     * --------------------------------------------------------------------
     *
     * URL to your CodeIgniter root. Typically, this will be your base URL,
     * WITH a trailing slash:
     *
     * E.g., http://example.com/
     */
    public string $baseURL = 'https://earmuff-famished-enduring.ngrok-free.dev/smart_dispenser/public/';

    /**
     * Allowed Hostnames in the Site URL other than the hostname in the baseURL.
     * If you want to accept multiple Hostnames, set this.
     *
     * E.g.,
     * When your site URL ($baseURL) is 'http://example.com/', and your site
     * also accepts 'http://media.example.com/' and 'http://accounts.example.com/':
     *     ['media.example.com', 'accounts.example.com']
     *
     * @var list<string>
     */
    public array $allowedHostnames = [];

    /**
     * --------------------------------------------------------------------
     * Index File
     * --------------------------------------------------------------------
     *
     * Typically, this will be your `index.php` file, unless you've renamed it to
     * something else. If you have configured your web server to remove this file
     * from your site URIs, set this variable to an empty string.
     */
    public string $indexPage = 'index.php';
```

```
/**
 * ------------------------------------------------------------------
 * URI PROTOCOL
 * ------------------------------------------------------------------
 *
 * This item determines which server global should be used to retrieve the
 * URI string. The default setting of 'REQUEST_URI' works for most servers.
 * If your links do not seem to work, try one of the other delicious flavors:
 *
 *  'REQUEST_URI': Uses $_SERVER['REQUEST_URI']
 * 'QUERY_STRING': Uses $_SERVER['QUERY_STRING']
 *    'PATH_INFO': Uses $_SERVER['PATH_INFO']
 *
 * WARNING: If you set this to 'PATH_INFO', URIs will always be URL-decoded!
 */
public string $uriProtocol = 'REQUEST_URI';

/*
|--------------------------------------------------------------------
| Allowed URL Characters
|--------------------------------------------------------------------
|
| This lets you specify which characters are permitted within your URLs.
| When someone tries to submit a URL with disallowed characters they will
| get a warning message.
|
| As a security measure you are STRONGLY encouraged to restrict URLs to
| as few characters as possible.
|
| By default, only these are allowed: `a-z 0-9~%.:_-`
|
| Set an empty string to allow all characters -- but only if you are insane.
|
| The configured value is actually a regular expression character group
| and it will be used as: '/\A[<permittedURIChars>]+\z/iu'
|
| DO NOT CHANGE THIS UNLESS YOU FULLY UNDERSTAND THE REPERCUSSIONS!!
|
*/
public string $permittedURIChars = 'a-z 0-9~%.:_\-';

/**
 * ------------------------------------------------------------------
 * Default Locale
 * ------------------------------------------------------------------
 *
 * The Locale roughly represents the language and location that your visitor
 * is viewing the site from. It affects the language strings and other
```

```
     * strings (like currency markers, numbers, etc), that your program
     * should run under for this request.
     */
    public string $defaultLocale = 'en';

    /**
     * --------------------------------------------------------------------
     * Negotiate Locale
     * --------------------------------------------------------------------
     *
     * If true, the current Request object will automatically determine the
     * language to use based on the value of the Accept-Language header.
     *
     * If false, no automatic detection will be performed.
     */
    public bool $negotiateLocale = false;

    /**
     * --------------------------------------------------------------------
     * Supported Locales
     * --------------------------------------------------------------------
     *
     * If $negotiateLocale is true, this array lists the locales supported
     * by the application in descending order of priority. If no match is
     * found, the first locale will be used.
     *
     * IncomingRequest::setLocale() also uses this list.
     *
     * @var list<string>
     */
    public array $supportedLocales = ['en'];

    /**
     * --------------------------------------------------------------------
     * Application Timezone
     * --------------------------------------------------------------------
     *
     * The default timezone that will be used in your application to display
     * dates with the date helper, and can be retrieved through app_timezone()
     *
     * @see https://www.php.net/manual/en/timezones.php for list of timezones
     *      supported by PHP.
     */
    public string $appTimezone = 'UTC';

    /**
     * --------------------------------------------------------------------
     * Default Character Set
```

```
     * ---------------------------------------------------------------
     *
     * This determines which character set is used by default in various methods
     * that require a character set to be provided.
     *
     * @see http://php.net/htmlspecialchars for a list of supported charsets.
     */
    public string $charset = 'UTF-8';

    /**
     * ---------------------------------------------------------------
     * Force Global Secure Requests
     * ---------------------------------------------------------------
     *
     * If true, this will force every request made to this application to be
     * made via a secure connection (HTTPS). If the incoming request is not
     * secure, the user will be redirected to a secure version of the page
     * and the HTTP Strict Transport Security (HSTS) header will be set.
     */
    public bool $forceGlobalSecureRequests = false;

    /**
     * ---------------------------------------------------------------
     * Reverse Proxy IPs
     * ---------------------------------------------------------------
     *
     * If your server is behind a reverse proxy, you must whitelist the proxy
     * IP addresses from which CodeIgniter should trust headers such as
     * X-Forwarded-For or Client-IP in order to properly identify
     * the visitor's IP address.
     *
     * You need to set a proxy IP address or IP address with subnets and
     * the HTTP header for the client IP address.
     *
     * Here are some examples:
     *     [
     *         '10.0.1.200'     => 'X-Forwarded-For',
     *         '192.168.5.0/24' => 'X-Real-IP',
     *     ]
     *
     * @var array<string, string>
     */
    public array $proxyIPs = [];

    /**
     * ---------------------------------------------------------------
     * Content Security Policy
     * ---------------------------------------------------------------
```

```
     *
     * Enables the Response's Content Secure Policy to restrict the sources that
     * can be used for images, scripts, CSS files, audio, video, etc. If enabled,
     * the Response object will populate default values for the policy from the
     * `ContentSecurityPolicy.php` file. Controllers can always add to those
     * restrictions at run time.
     *
     * For a better understanding of CSP, see these documents:
     *
     * @see http://www.html5rocks.com/en/tutorials/security/content-security-policy/
     * @see http://www.w3.org/TR/CSP/
     */
    public bool $CSPEnabled = false;
}
```

app/Config/ Database.php

```
<?php

namespace Config;

use CodeIgniter\Database\Config;

/**
 * Database Configuration
 */
class Database extends Config
{
    /**
     * The directory that holds the Migrations and Seeds directories.
     */
    public string $filesPath = APPPATH . 'Database' . DIRECTORY_SEPARATOR;

    /**
     * Lets you choose which connection group to use if no other is specified.
     */
    public string $defaultGroup = 'default';

    /**
     * The default database connection.
     *
     * @var array<string, mixed>
     */
    public array $default = [
        'DSN'          => '',
        'hostname'     => 'localhost',
        'username'     => 'root',
        'password'     => '',
        'database'     => 'db_smartdispenser',
```

```
        'DBDriver'     => 'MySQLi',
        'DBPrefix'     => '',
        'pConnect'     => false,
        'DBDebug'      => true,
        'charset'      => 'utf8mb4',
        'DBCollat'     => 'utf8mb4_general_ci',
        'swapPre'      => '',
        'encrypt'      => false,
        'compress'     => false,
        'strictOn'     => false,
        'failover'     => [],
        'port'         => 3306,
        'numberNative' => false,
        'foundRows'    => false,
        'dateFormat'   => [
            'date'     => 'Y-m-d',
            'datetime' => 'Y-m-d H:i:s',
            'time'     => 'H:i:s',
        ],
    ];

    //    /**
    //     * Sample database connection for SQLite3.
    //     *
    //     * @var array<string, mixed>
    //     */
    //    public array $default = [
    //        'database'    => 'database.db',
    //        'DBDriver'    => 'SQLite3',
    //        'DBPrefix'    => '',
    //        'DBDebug'     => true,
    //        'swapPre'     => '',
    //        'failover'    => [],
    //        'foreignKeys' => true,
    //        'busyTimeout' => 1000,
    //        'synchronous' => null,
    //        'dateFormat'  => [
    //            'date'     => 'Y-m-d',
    //            'datetime' => 'Y-m-d H:i:s',
    //            'time'     => 'H:i:s',
    //        ],
    //    ];

    //    /**
    //     * Sample database connection for Postgre.
    //     *
    //     * @var array<string, mixed>
    //     */
```

```
//    public array $default = [
//        'DSN'        => '',
//        'hostname'   => 'localhost',
//        'username'   => 'root',
//        'password'   => 'root',
//        'database'   => 'ci4',
//        'schema'     => 'public',
//        'DBDriver'   => 'Postgre',
//        'DBPrefix'   => '',
//        'pConnect'   => false,
//        'DBDebug'    => true,
//        'charset'    => 'utf8',
//        'swapPre'    => '',
//        'failover'   => [],
//        'port'       => 5432,
//        'dateFormat' => [
//            'date'     => 'Y-m-d',
//            'datetime' => 'Y-m-d H:i:s',
//            'time'     => 'H:i:s',
//        ],
//    ];

//    /**
//     * Sample database connection for SQLSRV.
//     *
//     * @var array<string, mixed>
//     */
//    public array $default = [
//        'DSN'        => '',
//        'hostname'   => 'localhost',
//        'username'   => 'root',
//        'password'   => 'root',
//        'database'   => 'ci4',
//        'schema'     => 'dbo',
//        'DBDriver'   => 'SQLSRV',
//        'DBPrefix'   => '',
//        'pConnect'   => false,
//        'DBDebug'    => true,
//        'charset'    => 'utf8',
//        'swapPre'    => '',
//        'encrypt'    => false,
//        'failover'   => [],
//        'port'       => 1433,
//        'dateFormat' => [
//            'date'     => 'Y-m-d',
//            'datetime' => 'Y-m-d H:i:s',
//            'time'     => 'H:i:s',
//        ],
```

```
    //    ];

    //    /**
    //     * Sample database connection for OCI8.
    //     *
    //     * You may need the following environment variables:
    //     * NLS_LANG               = 'AMERICAN_AMERICA.UTF8'
    //     * NLS_DATE_FORMAT        = 'YYYY-MM-DD HH24:MI:SS'
    //     * NLS_TIMESTAMP_FORMAT    = 'YYYY-MM-DD HH24:MI:SS'
    //     * NLS_TIMESTAMP_TZ_FORMAT = 'YYYY-MM-DD HH24:MI:SS'
    //     *
    //     * @var array<string, mixed>
    //     */
    //    public array $default = [
    //        'DSN'        => 'localhost:1521/XEPDB1',
    //        'username'   => 'root',
    //        'password'   => 'root',
    //        'DBDriver'   => 'OCI8',
    //        'DBPrefix'   => '',
    //        'pConnect'   => false,
    //        'DBDebug'    => true,
    //        'charset'    => 'AL32UTF8',
    //        'swapPre'    => '',
    //        'failover'   => [],
    //        'dateFormat' => [
    //            'date'     => 'Y-m-d',
    //            'datetime' => 'Y-m-d H:i:s',
    //            'time'     => 'H:i:s',
    //        ],
    //    ];

    /**
     * This database connection is used when running PHPUnit database tests.
     *
     * @var array<string, mixed>
     */
    public array $tests = [
        'DSN'         => '',
        'hostname'    => '127.0.0.1',
        'username'    => '',
        'password'    => '',
        'database'    => ':memory:',
        'DBDriver'    => 'SQLite3',
        'DBPrefix'    => 'db_',  // Needed to ensure we're working correctly with prefixes live. DO NOT
REMOVE FOR CI DEVS
        'pConnect'    => false,
        'DBDebug'     => true,
        'charset'     => 'utf8',
```

```
        'DBCollat'    => '',
        'swapPre'     => '',
        'encrypt'     => false,
        'compress'    => false,
        'strictOn'    => true,
        'failover'    => [],
        'port'        => 3306,
        'foreignKeys' => true,
        'busyTimeout' => 1000,
        'synchronous' => null,
        'dateFormat'  => [
            'date'     => 'Y-m-d',
            'datetime' => 'Y-m-d H:i:s',
            'time'     => 'H:i:s',
        ],
    ];

    public function __construct()
    {
        parent::__construct();

        // Ensure that we always set the database group to 'tests' if
        // we are currently running an automated test suite, so that
        // we don't overwrite live data on accident.
        if (ENVIRONMENT === 'testing') {
            $this->defaultGroup = 'tests';
        }
    }
}
```

app/Config/ Routes.php

```
<?php

use CodeIgniter\Router\RouteCollection;

/**
 * @var RouteCollection $routes
 */
$routes->get('/', 'Home::index');

// ============================================================
// RUTE CHECKOUT DAN PEMBAYARAN
// ============================================================
$routes->get('/checkout/form/(:num)', 'Checkout::form/$1');
$routes->post('/checkout/proses/(:num)', 'Checkout::proses/$1');

// ============================================================
// RUTE WEBHOOK (Sinyal Xendit)
```

```
// ============================================================
$routes->post('/webhook/xendit', 'Webhook::xendit');

// ============================================================
// RUTE SISTEM LOGIN & REGISTER
// ============================================================
$routes->get('/login', 'Auth::index');
$routes->post('/login/process', 'Auth::process');
$routes->get('/logout', 'Auth::logout');
$routes->get('/register', 'Auth::register');
$routes->post('/register/process', 'Auth::registerProcess');

// ============================================================
// RUTE KHUSUS ADMIN & IOT DASHBOARD
// ============================================================
$routes->get('/admin', 'Admin::index');
$routes->post('/admin/update_iot', 'Admin::update_iot');
$routes->get('/admin/get_active_pumping', 'Admin::get_active_pumping');
$routes->get('admin/radar_live', 'Admin::radar_live');

// ============================================================
// RUTE API MESIN ESP32
// ============================================================
$routes->get('/api/cek_mesin', 'Api::cek_mesin');
$routes->get('/api/mulai_pompa/(:segment)', 'Api::mulai_pompa/$1');
$routes->get('/api/lapor_progress/(:segment)', 'Api::lapor_progress/$1');

// ============================================================
// RUTE MONITORING REAL-TIME PELANGGAN
// ============================================================
$routes->get('/monitoring/track/(:segment)', 'Monitoring::track/$1');
$routes->get('/monitoring/get_status/(:segment)', 'Monitoring::get_status/$1');
```

App/Config/ Routing.php

```
<?php

/**
 * This file is part of CodeIgniter 4 framework.
 *
 * (c) CodeIgniter Foundation <admin@codeigniter.com>
 *
 * For the full copyright and license information, please view
 * the LICENSE file that was distributed with this source code.
 */

namespace Config;

use CodeIgniter\Config\Routing as BaseRouting;
```

```php
/**
 * Routing configuration
 */
class Routing extends BaseRouting
{
    /**
     * For Defined Routes.
     * An array of files that contain route definitions.
     * Route files are read in order, with the first match
     * found taking precedence.
     *
     * Default: APPPATH . 'Config/Routes.php'
     *
     * @var list<string>
     */
    public array $routeFiles = [
        APPPATH . 'Config/Routes.php',
    ];

    /**
     * For Defined Routes and Auto Routing.
     * The default namespace to use for Controllers when no other
     * namespace has been specified.
     *
     * Default: 'App\Controllers'
     */
    public string $defaultNamespace = 'App\Controllers';

    /**
     * For Auto Routing.
     * The default controller to use when no other controller has been
     * specified.
     *
     * Default: 'Home'
     */
    public string $defaultController = 'Home';

    /**
     * For Defined Routes and Auto Routing.
     * The default method to call on the controller when no other
     * method has been set in the route.
     *
     * Default: 'index'
     */
    public string $defaultMethod = 'index';

    /**
```

```
     * For Auto Routing.
     * Whether to translate dashes in URIs for controller/method to underscores.
     * Primarily useful when using the auto-routing.
     *
     * Default: false
     */
    public bool $translateURIDashes = false;

    /**
     * Sets the class/method that should be called if routing doesn't
     * find a match. It can be the controller/method name like: Users::index
     *
     * This setting is passed to the Router class and handled there.
     *
     * If you want to use a closure, you will have to set it in the
     * routes file by calling:
     *
     * $routes->set404Override(function() {
     *    // Do something here
     * });
     *
     * Example:
     *  public $override404 = 'App\Errors::show404';
     */
    public ?string $override404 = null;

    /**
     * If TRUE, the system will attempt to match the URI against
     * Controllers by matching each segment against folders/files
     * in APPPATH/Controllers, when a match wasn't found against
     * defined routes.
     *
     * If FALSE, will stop searching and do NO automatic routing.
     */
    public bool $autoRoute = false;

    /**
     * If TRUE, the system will look for attributes on controller
     * class and methods that can run before and after the
     * controller/method.
     *
     * If FALSE, will ignore any attributes.
     */
    public bool $useControllerAttributes = true;

    /**
     * For Defined Routes.
     * If TRUE, will enable the use of the 'prioritize' option
```

```
     * when defining routes.
     *
     * Default: false
     */
    public bool $prioritize = false;

    /**
     * For Defined Routes.
     * If TRUE, matched multiple URI segments will be passed as one parameter.
     *
     * Default: false
     */
    public bool $multipleSegmentsOneParam = false;

    /**
     * For Auto Routing (Improved).
     * Map of URI segments and namespaces.
     *
     * The key is the first URI segment. The value is the controller namespace.
     * E.g.,
     *   [
     *       'blog' => 'Acme\Blog\Controllers',
     *   ]
     *
     * @var array<string, string>
     */
    public array $moduleRoutes = [];

    /**
     * For Auto Routing (Improved).
     * Whether to translate dashes in URIs for controller/method to CamelCase.
     * E.g., blog-controller -> BlogController
     *
     * If you enable this, $translateURIDashes is ignored.
     *
     * Default: false
     */
    public bool $translateUriToCamelCase = true;
}
```

App/Controllers/Admin.php

```
<?php

namespace App\Controllers;

class Admin extends BaseController
{
    public function index()
```

```
    {
        $session = session();
        if (!$session->get('logged_in') || $session->get('role') != 'admin') {
            return redirect()->to('/login');
        }

        $db = \Config\Database::connect();
        $queryUang = $db->query("SELECT SUM(total_amount) as pendapatan FROM transaction WHERE
payment_status IN ('Paid', 'Completed')");
        $pendapatan = $queryUang->getRow()->pendapatan ?? 0;

        $total_trx = $db->table('transaction')->countAllResults();
        $dispenser = $db->table('dispenser')->where('dispenser_id', 1)->get()->getRowArray();
        $transaksi = $db->table('transaction')
                        ->select('transaction.*, user.name as nama_pelanggan')
                        ->join('user', 'user.user_id = transaction.user_id', 'left')
                        ->orderBy('created_at', 'DESC')
                        ->get()->getResultArray();

        $data = [
            'pendapatan' => $pendapatan,
            'total_trx'  => $total_trx,
            'dispenser'  => $dispenser,
            'transaksi'  => $transaksi
        ];

        return view('pages/admin_dashboard', $data);
    }

    public function radar_live()
    {
        if (!session()->get('logged_in') || session()->get('role') != 'admin') {
            return $this->response->setJSON(['status' => 'unauthorized']);
        }

        $db = \Config\Database::connect();
        $trx = $db->table('transaction')
                  ->whereIn('payment_status', ['Paid', 'Completed'])
                  ->groupStart()
                    ->where('status_pengisian !=', 'Selesai')
                    ->orWhere('status_pengisian IS NULL')
                    ->orWhere('status_pengisian', '')
                  ->groupEnd()
                  ->orderBy('created_at', 'ASC')
                  ->get()->getRowArray();

        if ($trx) {
            $qty = (int)($trx['qty'] ?? 1);
```

```
            $qty_selesai = (int)($trx['qty_selesai'] ?? 0);
            $sisa = $qty - $qty_selesai;

            $statusTampil = $trx['status_pengisian'];
            if (empty($statusTampil)) {
                $statusTampil = "Menunggu Wadah/Botol";
            }

            return $this->response->setJSON([
                'status'          => 'aktif',
                'nama_pembeli'    => $trx['delivery_name'] ?? 'Pelanggan',
                'detail_pesanan'  => $trx['detail_pesanan'],
                'qty'             => $qty,
                'qty_selesai'     => $qty_selesai,
                'sisa'            => $sisa,
                'status_pengisian' => $statusTampil
            ]);
        } else {
            return $this->response->setJSON(['status' => 'idle']);
        }
    }

    public function update_iot()
    {
        $db = \Config\Database::connect();
        $tipe = $this->request->getPost('tipe');
        $nilai = $this->request->getPost('nilai');

        $db->table('dispenser')->where('dispenser_id', 1)->update([$tipe => $nilai]);
        return $this->response->setJSON(['status' => 'success']);
    }
}
```

App/Controllers/Api.php

```
<?php

namespace App\Controllers;

class Api extends BaseController
{
    public function cek_mesin()
    {
        $db = \Config\Database::connect();
        $setting = $db->table('dispenser')->where('dispenser_id', 1)->get()->getRowArray();
        $manual = $setting['pompa_manual'] ?? 'OFF';

        $trx = $db->table('transaction')
                ->groupStart()
```

```
                ->where('payment_status', 'Paid')
                ->orWhere('payment_status', 'Completed')
            ->groupEnd()
            ->where('status_pengisian !=', 'Selesai')
            ->orderBy('created_at', 'ASC')
            ->get()->getRowArray();

        if ($trx) {
            $detail = strtolower($trx['detail_pesanan'] ?? '');

            if (strpos($detail, '19') !== false) {
                $target_pulse = $setting['pulse_19000'] ?? 8550;
                $volume = 19000;
            } elseif (strpos($detail, '1.5') !== false || strpos($detail, '1500') !== false) {
                $target_pulse = $setting['pulse_1500'] ?? 675;
                $volume = 1500;
            } else {
                $target_pulse = $setting['pulse_600'] ?? 270;
                $volume = 600;
            }

            return $this->response->setJSON([
                'perintah'     => 'NYALA',
                'volume'       => $volume,
                'target_pulse' => (int) $target_pulse,
                'payment_ref'  => $trx['payment_ref'],
                'qty'          => $trx['qty'],
                'qty_selesai'  => $trx['qty_selesai'],
                'manual'       => $manual
            ]);
        } else {
            return $this->response->setJSON([
                'perintah' => 'MATI',
                'manual'   => $manual
            ]);
        }
    }

    public function mulai_pompa($payment_ref)
    {
        $db = \Config\Database::connect();
        $db->table('transaction')->where('payment_ref', $payment_ref)->update(['status_pengisian' =>
'Sedang Mengisi']);
        return $this->response->setStatusCode(200)->setJSON(['status' => 'success']);
    }

    public function lapor_progress($payment_ref)
    {
```

```
    $db = \Config\Database::connect();
    $trx = $db->table('transaction')->where('payment_ref', $payment_ref)->get()->getRowArray();

    if ($trx) {
        $qty_selesai_baru = $trx['qty_selesai'] + 1;

        if ($qty_selesai_baru >= $trx['qty']) {
            $db->table('transaction')->where('transaction_id', $trx['transaction_id'])->update([
                'qty_selesai'      => $qty_selesai_baru,
                'status_pengisian' => 'Selesai',
                'payment_status'   => 'Completed'
            ]);
        } else {
            $db->table('transaction')->where('transaction_id', $trx['transaction_id'])->update([
                'qty_selesai'      => $qty_selesai_baru,
                'status_pengisian' => 'Menunggu Botol Selanjutnya'
            ]);
        }
        return $this->response->setStatusCode(200)->setJSON(['status' => 'Laporan Diterima']);
    }
    return $this->response->setStatusCode(400)->setJSON(['status' => 'Gagal']);
}

public function radar_live()
{
    $db = \Config\Database::connect();
    $trx = $db->table('transaction')
            ->whereIn('payment_status', ['Paid', 'Completed'])
            ->groupStart()
                ->where('status_pengisian !=', 'Selesai')
                ->orWhere('status_pengisian IS NULL')
                ->orWhere('status_pengisian', '')
            ->groupEnd()
            ->orderBy('created_at', 'ASC')
            ->get()->getRowArray();

    if ($trx) {
        $qty = (int)($trx['qty'] ?? 1);
        $qty_selesai = (int)($trx['qty_selesai'] ?? 0);
        $sisa = $qty - $qty_selesai;

        $statusTampil = $trx['status_pengisian'];
        if (empty($statusTampil)) {
            $statusTampil = "Menunggu Wadah/Botol";
        }

        return $this->response->setJSON([
            'status'          => 'aktif',
```

```
            'nama_pembeli'     => $trx['delivery_name'] ?? 'Pelanggan',
            'detail_pesanan'   => $trx['detail_pesanan'],
            'qty'              => $qty,
            'qty_selesai'      => $qty_selesai,
            'sisa'             => $sisa,
            'status_pengisian' => $statusTampil
        ]);
    } else {
        return $this->response->setJSON(['status' => 'idle']);
    }
  }
}
```

App/Controllers/Auth.php

```
<?php

namespace App\Controllers;

class Auth extends BaseController
{
  public function index()
  {
    return view('pages/login');
  }

  public function process()
  {
    $session = session();

    $username = $this->request->getPost('username');
    $password = $this->request->getPost('password');

    $db = \Config\Database::connect();

    $user = $db->table('user')->where('username', $username)->get()->getRowArray();

    if ($user) {
      if (password_verify($password, $user['password_hash'])) {

        $session->set([
          'user_id'   => $user['user_id'],
          'name'      => $user['name'],
          'email'     => $user['email'],
          'role'      => $user['role'],
          'logged_in' => TRUE
        ]);
```

```
            if ($user['role'] == 'admin') {
                return redirect()->to('/admin');
            } else {
                return redirect()->to('/');
            }
        } else {
            $session->setFlashdata('error', 'Kata Sandi salah!');
            return redirect()->to('/login');
        }
    } else {
        $session->setFlashdata('error', 'Username tidak ditemukan!');
        return redirect()->to('/login');
    }
}

public function logout()
{
    session()->destroy();
    return redirect()->to('/login');
}

public function register()
{
    return view('pages/register');
}

public function registerProcess()
{
    $db = \Config\Database::connect();

    $name = $this->request->getPost('name');
    $email = $this->request->getPost('email');
    $username = $this->request->getPost('username');
    $password = $this->request->getPost('password');

    $cek_user = $db->table('user')->where('username', $username)->get()->getRowArray();
    if ($cek_user) {
        session()->setFlashdata('error', 'Username sudah dipakai, silakan cari yang lain!');
        return redirect()->to('/register');
    }

    $hashedPassword = password_hash($password, PASSWORD_BCRYPT);

    $data_baru = [
        'name'          => $name,
        'email'         => $email,
        'username'      => $username,
        'role'          => 'user',
```

```
            'password_hash' => $hashedPassword
        ];

        $db->table('user')->insert($data_baru);

        session()->setFlashdata('success', 'Akun berhasil dibuat! Silakan Login menggunakan Username Anda.');
        return redirect()->to('/login');
    }
}
```

App/Controllers/Checkout.php

```
<?php

namespace App\Controllers;

class Checkout extends BaseController
{
    // ========================================================
    // 1. FUNGSI MENAMPILKAN FORM
    // ========================================================
    public function form(string $id_produk)
    {
        $session = session();
        if (!$session->get('logged_in') || $session->get('role') != 'user') {
            $session->setFlashdata('error', 'Silakan Login atau Daftar akun terlebih dahulu untuk memesan air!');
            return redirect()->to('/login');
        }

        $nama_produk = "";
        if ($id_produk == 1) $nama_produk = "Air Minum 600 ML";
        elseif ($id_produk == 2) $nama_produk = "Air Minum 1.5 Liter";
        elseif ($id_produk == 3) $nama_produk = "Galon 19 Liter";
        else return redirect()->to('/');

        $data = [
            'id_produk' => $id_produk,
            'nama_produk' => $nama_produk
        ];

        return view('pages/order_form', $data);
    }

    // ========================================================
    // 2. FUNGSI PROSES MENGARAH KE HALAMAN INVOICE ASLI XENDIT
    // ========================================================
    public function proses(string $id_produk)
```

```
    {
        $session = session();
        if (!$session->get('logged_in') || $session->get('role') != 'user') {
            return redirect()->to('/login');
        }

        $user_id = $session->get('user_id');
        $email_pelanggan = $session->get('email');

        $delivery_name = $this->request->getPost('delivery_name');
        $delivery_address = $this->request->getPost('delivery_address');
        $qty = $this->request->getPost('qty');

        $harga_satuan = 0;
        $nama_produk = "";

        if ($id_produk == 1) {
            $harga_satuan = 1000;
            $nama_produk = "Air Minum 600 ML";
        } elseif ($id_produk == 2) {
            $harga_satuan = 2000;
            $nama_produk = "Air Minum 1.5 Liter";
        } elseif ($id_produk == 3) {
            $harga_satuan = 5000;
            $nama_produk = "Galon 19 Liter";
        } else {
            return redirect()->to('/');
        }

        $total_harga = $harga_satuan * $qty;
        $secret_key =
'xnd_development_t48VMv6S1GB7ESpUlaaoilaN4Xn8LXIZN4xGt1jEoKqgxNzhn1eJPP4G0PRI2';
        $external_id = 'BLUEBUB-' . time();

        $db = \Config\Database::connect();

        $data_transaksi = [
            'user_id'          => $user_id,
            'email'            => $email_pelanggan,
            'detail_pesanan'   => $qty . 'x ' . $nama_produk,
            'id_produk'        => $id_produk,
            'qty'              => $qty,
            'delivery_name'    => $delivery_name,
            'delivery_address' => $delivery_address,
            'total_amount'     => $total_harga,
            'payment_status'   => 'Pending',
            'status_pengisian' => 'Menunggu Mesin',
            'qty_selesai'      => 0,
```

```
            'payment_ref'      => $external_id
        ];
        $db->table('transaction')->insert($data_transaksi);

        $data_invoice = [
            'external_id' => $external_id,
            'amount'      => $total_harga,
            'description' => 'Pembelian ' . $qty . 'x ' . $nama_produk . ' (Diantar ke: ' . $delivery_name . ')',
            'currency'    => 'IDR',
            'success_redirect_url' => base_url('monitoring/track/' . $external_id),
            'failure_redirect_url' => base_url('dashboard')
        ];

        $client = \Config\Services::curlrequest();
        try {
            $response = $client->request('POST', 'https://api.xendit.co/v2/invoices', [
                'auth' => [$secret_key, ''],
                'json' => $data_invoice,
                'http_errors' => false
            ]);

            $body = json_decode($response->getBody());

            // Arahkan pelanggan langsung ke link invoice Xendit yang berakhiran xendit.co/...
            if (isset($body->invoice_url)) {
                return redirect()->to($body->invoice_url);
            } else {
                die('Gagal membuat Invoice Xendit. Keterangan: ' . json_encode($body));
            }
        } catch (\Exception $e) {
            die('Terjadi Kesalahan Koneksi: ' . $e->getMessage());
        }
    }
}
```

App/Controllers/Home.php

```
<?php

namespace App\Controllers;

class Home extends BaseController
{
    public function index()
    {
        $db = \Config\Database::connect();
        $terlaris = $db->query("SELECT id_produk, SUM(qty) as total_terjual FROM transaction WHERE
payment_status = 'Paid' GROUP BY id_produk ORDER BY total_terjual DESC LIMIT 1")->getRowArray();
```

```
        $id_terlaris = $terlaris ? $terlaris['id_produk'] : 3;

        $data = [
            'id_terlaris' => $id_terlaris
        ];

        return view('pages/home', $data);
    }
}
```

App/Controllers/Monitoring.php

```
<?php

namespace App\Controllers;

class Monitoring extends BaseController
{
    public function track($order_id)
    {
        $db = \Config\Database::connect();
        $trx = $db->table('transaction')->where('payment_ref', $order_id)->get()->getRowArray();

        // Kalau transaksi tidak ada, tendang kembali ke menu awal
        if (!$trx) return redirect()->to('/');

        // Cek durasi per 1 wadah
        $waktu_per_pcs = 0;
        if ($trx['id_produk'] == 1) $waktu_per_pcs = 10;
        elseif ($trx['id_produk'] == 2) $waktu_per_pcs = 15;
        elseif ($trx['id_produk'] == 3) $waktu_per_pcs = 35;

        $data = [
            'trx' => $trx,
            'waktu_per_pcs' => $waktu_per_pcs
        ];

        return view('pages/monitoring', $data);
    }
    public function get_status($order_id)
    {
        $db = \Config\Database::connect();
        $trx = $db->table('transaction')->where('payment_ref', $order_id)->get()->getRowArray();
        return $this->response->setJSON($trx);
    }
}
```

App/Controllers/Webhook.php

```php
<?php

namespace App\Controllers;

class Webhook extends BaseController
{
    public function xendit()
    {
        $data = $this->request->getJSON();

        if ($data && isset($data->external_id) && isset($data->status)) {

            if ($data->status == 'PAID') {

                $db = \Config\Database::connect();
                $db->table('transaction')
                   ->where('payment_ref', $data->external_id)
                   ->update(['payment_status' => 'Paid']);
            }
        }
        return $this->response->setStatusCode(200)->setJSON(['message' => 'Sinyal Diterima']);
    }
}
```

Views/layout/tempate.php

```php
<!DOCTYPE html>
<html lang="id">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Smart Dispenser</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
    <style>
        body {
            background-color: #f8f9fa;
        }
    </style>
</head>

<body>

    <nav class="navbar navbar-expand-lg navbar-dark bg-primary shadow-sm mb-4">
    <div class="container">
        <a class="navbar-brand fw-bold" href="<?= base_url('/'); ?>">
            <i class="fa-solid fa-droplet"></i> Smart Dispenser
        </a>
```

```
      <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarNav">
        <span class="navbar-toggler-icon"></span>
      </button>

      <div class="collapse navbar-collapse" id="navbarNav">
        <div class="ms-auto d-flex align-items-center mt-3 mt-lg-0">
          <?php if (session()->get('logged_in')): ?>

            <span class="text-white me-3 fw-bold">
              <i class="fa-solid fa-circle-user"></i> Halo, Admin
            </span>

            <a href="<?= base_url('logout'); ?>" class="btn btn-danger btn-sm fw-bold">Logout</a>

          <?php else: ?>
            <a href="<?= base_url('login'); ?>" class="btn btn-light fw-bold px-4 rounded-pill text-
primary">
              <i class="fa-solid fa-right-to-bracket"></i> Masuk / Login
            </a>
          <?php endif; ?>
        </div>
      </div>
    </div>
</nav>

    <div class="container">
      <?= $this->renderSection('content'); ?>
    </div>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
</body>

</html>
```

Views/pages/admin_dashboard.php

```
<?= $this->extend('layout/template'); ?>
<?= $this->section('content'); ?>

<div class="row mb-4 align-items-center">
    <div class="col-md-12">
      <h2 class="fw-bold text-primary"><i class="fa-solid fa-gauge-high"></i> Dashboard Monitoring</h2>
      <p class="text-muted">Halo, Admin! Berikut adalah ringkasan sistem Anda.</p>
    </div>
</div>

<div class="row mb-4">
```

```
    <div class="col-md-4 mb-3">
      <div class="card shadow-sm border-0 bg-primary text-white h-100 rounded-4">
        <div class="card-body">
          <h6 class="card-title"><i class="fa-solid fa-wallet"></i> Total Pendapatan</h6>
          <h3 class="fw-bold mt-3">Rp <?= number_format((float)($pendapatan ?? 0), 0, ',', '.'); ?></h3>
        </div>
      </div>
    </div>
    <div class="col-md-4 mb-3">
      <div class="card shadow-sm border-0 bg-success text-white h-100 rounded-4">
        <div class="card-body">
          <h6 class="card-title"><i class="fa-solid fa-money-bill-transfer"></i> Total Transaksi</h6>
          <h3 class="fw-bold mt-3"><?= $total_trx ?? 0; ?> Pesanan</h3>
        </div>
      </div>
    </div>
    <div class="col-md-4 mb-3">
      <div class="card shadow-sm border-0 bg-info text-white h-100 rounded-4">
        <div class="card-body">
          <h6 class="card-title"><i class="fa-solid fa-faucet-drip"></i> Status Mesin (ID: <?=
$dispenser['dispenser_id'] ?? '1'; ?>)</h6>
          <h3 class="fw-bold mt-3">
            <?php if (isset($dispenser['status']) && $dispenser['status'] == 'Ready'): ?>
              <i class="fa-solid fa-circle-check"></i> Siap
            <?php else: ?>
              <i class="fa-solid fa-spinner fa-spin"></i> Sibuk
            <?php endif; ?>
          </h3>
        </div>
      </div>
    </div>
</div>

<div class="row mb-4">
    <div class="col-md-6 mb-3">
      <div class="card shadow-sm border-0 rounded-4 h-100" style="border-left: 5px solid #e91e63
!important;">
        <div class="card-body">
          <h5 class="fw-bold text-dark"><i class="fa-solid fa-power-off text-danger"></i> Kendali Keran
(Manual)</h5>
          <hr>
          <p class="text-muted small">Gunakan tombol ini untuk menyalakan dan mematikan keran
secara manual.</p>
          <div class="d-grid gap-2 d-md-block">
            <button class="btn btn-success fw-bold px-4" onclick="setIot('pompa_manual', 'ON')"><i
class="fa-solid fa-play"></i> HIDUPKAN KERAN</button>
            <button class="btn btn-danger fw-bold px-4" onclick="setIot('pompa_manual', 'OFF')"><i
class="fa-solid fa-stop"></i> MATIKAN KERAN</button>
```

```
            </div>
          </div>
        </div>
    </div>

    <div class="col-md-6 mb-3">
        <div class="card shadow-sm border-0 rounded-4 h-100" style="border-left: 5px solid #9c27b0
!important;">
            <div class="card-body">
                <h5 class="fw-bold text-dark"><i class="fa-solid fa-sliders text-purple"></i> Kalibrasi Presisi
(Pulse)</h5>
                <hr>
                <div class="input-group mb-2">
                    <span class="input-group-text bg-light" style="width: 100px;">600 ML</span>
                    <input type="number" id="val_600" class="form-control" value="<?= $dispenser['pulse_600'] ??
270 ?>">
                    <button class="btn btn-primary" onclick="setIot('pulse_600',
$('#val_600').val())">Simpan</button>
                </div>
                <div class="input-group mb-2">
                    <span class="input-group-text bg-light" style="width: 100px;">1.5 Liter</span>
                    <input type="number" id="val_1500" class="form-control" value="<?= $dispenser['pulse_1500']
?? 675 ?>">
                    <button class="btn btn-primary" onclick="setIot('pulse_1500',
$('#val_1500').val())">Simpan</button>
                </div>
                <div class="input-group">
                    <span class="input-group-text bg-light" style="width: 100px;">19 Liter</span>
                    <input type="number" id="val_19000" class="form-control" value="<?=
$dispenser['pulse_19000'] ?? 8550 ?>">
                    <button class="btn btn-primary" onclick="setIot('pulse_19000',
$('#val_19000').val())">Simpan</button>
                </div>
            </div>
        </div>
    </div>
</div>

<div class="row mb-4">
    <div class="col-12">
        <div class="card shadow-sm border-0 rounded-4" style="border-left: 5px solid #ff9800 !important;">
            <div class="card-body">
                <h5 class="fw-bold text-dark"><i class="fa-solid fa-tower-broadcast text-warning"></i> Radar
Aktivitas Mesin (Live)</h5>
                <hr>
                <div id="liveRadarAdmin">
                    <p class="text-muted mb-0"><i class="fa-solid fa-mug-hot"></i> Mesin sedang dalam kondisi
Siaga (Idle).</p>
```

```
            </div>
          </div>
        </div>
      </div>
    </div>

    <div class="card shadow-sm border-0 rounded-4">
      <div class="card-header bg-white border-bottom-0 pt-4 pb-0">
        <h5 class="fw-bold"><i class="fa-solid fa-clock-rotate-left"></i> Riwayat Transaksi Terbaru</h5>
      </div>
      <div class="card-body">
        <div class="table-responsive" id="tabel-transaksi-container">
          <table class="table table-hover align-middle">
            <thead class="table-light">
              <tr>
                <th>No</th>
                <th>Kode Referensi</th>
                <th>Nama Pembeli</th>
                <th>Alamat Pengiriman</th>
                <th>Detail Pesanan</th>
                <th>Nominal</th>
                <th>Status Pembayaran</th>
                <th>Status Mesin</th>
              </tr>
            </thead>
            <tbody>
              <?php if (empty($transaksi)): ?>
                <tr>
                  <td colspan="8" class="text-center">Belum ada data.</td>
                </tr>
              <?php else: ?>
                <?php $no = 1;
                foreach ($transaksi as $t) : ?>
                  <tr>
                    <td><?= $no++; ?></td>
                    <td><span class="badge bg-secondary"><?= $t['payment_ref']; ?></span></td>

                    <td class="fw-bold text-dark"><i class="fa-solid fa-user text-muted"></i> <?=
$t['delivery_name'] ?? 'Guest/Anonim'; ?></td>

                    <td class="small text-muted" style="max-width: 200px; word-wrap: break-word;"><?=
$t['delivery_address'] ?? '-'; ?></td>

                    <td><span class="badge bg-info text-dark"><?= $t['detail_pesanan'];
?></span></td>

                    <td class="text-success fw-bold">Rp <?= number_format((float)$t['total_amount'], 0,
',', '.'); ?></td>
```

```
                        <td>
                            <?php if ($t['payment_status'] == 'Pending'): ?>
                                <span class="badge bg-warning text-dark">Menunggu</span>
                            <?php else: ?>
                                <span class="badge bg-success">Lunas</span>
                            <?php endif; ?>
                        </td>
                        <td>
                            <?php if ($t['payment_status'] == 'Pending'): ?>
                                <span class="badge bg-secondary">Mesin Mati</span>
                            <?php elseif ($t['status_pengisian'] == 'Sedang Mengisi'): ?>
                                <span class="badge bg-primary fa-beat">Sedang Mengisi</span>
                            <?php elseif ($t['status_pengisian'] == 'Selesai'): ?>
                                <span class="badge bg-success">Selesai</span>
                            <?php else: ?>
                                <span class="badge bg-info text-dark">Menunggu Botol</span>
                            <?php endif; ?>
                        </td>
                    </tr>
                <?php endforeach; ?>
            <?php endif; ?>
        </tbody>
    </table>
</div>
</div>
</div>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<script>
    function setIot(tipe, nilai) {
        if (nilai === '') return alert("Nilai tidak boleh kosong!");

        $.ajax({
            url: "/smart_dispenser/public/index.php/admin/update_iot",
            type: "POST",
            data: {
                tipe: tipe,
                nilai: nilai
            },
            success: function(res) {
                console.log("Perintah " + tipe + " sukses dikirim ke ESP32!");
            },
            error: function(xhr, status, error) {
                alert("GAGAL! Kode Error: " + xhr.status);
            }
        });
    }
```

```
$(document).ready(function() {

    function pantauRadarMesin() {
        $.ajax({
            url: "/smart_dispenser/public/index.php/admin/radar_live",
            type: "GET",
            dataType: "json",
            cache: false,
            success: function(data) {
                if (data.status === 'aktif') {
                    let htmlAktif = `
                        <div class="alert border-0 bg-light shadow-sm mb-0">
                            <div class="d-flex justify-content-between align-items-center mb-3">
                                <h5 class="fw-bold text-primary mb-0">
                                    <i class="fa-solid fa-spinner fa-spin me-2"></i> Sedang Memproses Pesanan
                                </h5>
                                <span class="badge bg-primary px-3 py-2 fs-6">${data.status_pengisian}</span>
                            </div>

                            <p class="mb-3 text-dark"><b>Pembeli:</b> ${data.nama_pembeli}   |  
<b>Item:</b> ${data.detail_pesanan}</p>

                            <div class="row text-center mt-2 bg-white rounded-3 py-3 shadow-sm mx-0 border">
                                <div class="col-6 border-end">
                                    <h2 class="text-success fw-bold mb-0">${data.qty_selesai}</h2>
                                    <span class="text-muted small fw-bold">WADAH BERHASIL TERISI</span>
                                </div>
                                <div class="col-6">
                                    <h2 class="text-danger fw-bold mb-0">${data.sisa}</h2>
                                    <span class="text-muted small fw-bold">TERSISA / BELUM DIISI</span>
                                </div>
                            </div>
                        </div>
                    `;
                    $('#liveRadarAdmin').html(htmlAktif);
                } else {
                    $('#liveRadarAdmin').html('<p class="text-muted mb-0"><i class="fa-solid fa-mug-hot"></i>
Mesin sedang dalam kondisi Siaga (Idle).</p>');
                }
            },
            error: function() {
                console.log("Menunggu data radar dari server...");
            }
        });
    }

    setInterval(pantauRadarMesin, 3000);
```

```
        function refreshTabelTransaksi() {
            $('#tabel-transaksi-container').load(window.location.href + ' #tabel-transaksi-container > *');
        }

        setInterval(refreshTabelTransaksi, 20000);
    });
</script>

<?= $this->endSection(); ?>
```

Views/pages/home.php

```
<?= $this->extend('layout/template'); ?>

<?= $this->section('content'); ?>

<?php $id_terlaris = $id_terlaris ?? 3; ?>

<div class="row text-center mb-4">
    <div class="col-12">
        <h2 class="fw-bold text-primary">Pilih Takaran Air Minum</h2>
        <p class="text-muted">Silakan pilih volume air sesuai dengan wadah Anda. Pembayaran menggunakan QRIS.</p>
    </div>
</div>

<div class="row justify-content-center">
    <div class="col-6 col-md-4 mb-4">
        <div class="card h-100 shadow-sm <?= ($id_terlaris == 1) ? 'border-primary border-2' : 'border-0'; ?>">
            <div class="card-body text-center mt-3">
                <i class="fa-solid fa-bottle-water text-info" style="font-size: 60px;"></i>

                <?php if($id_terlaris == 1): ?>
                    <span class="badge bg-warning text-dark position-absolute top-0 end-0 m-2">Terlaris</span>
                <?php endif; ?>

                <h4 class="card-title mt-4 fw-bold">600 ML</h4>
                <h5 class="text-success fw-bold">Rp 1.000</h5>
                <a href="<?= base_url('checkout/form/1'); ?>" class="btn btn-primary w-100 mt-3 rounded-pill"><i class="fa-solid fa-cart-shopping"></i> Pesan</a>
            </div>
        </div>
    </div>

    <div class="col-6 col-md-4 mb-4">
        <div class="card h-100 shadow-sm <?= ($id_terlaris == 2) ? 'border-primary border-2' : 'border-0'; ?>">
```

```
            <div class="card-body text-center mt-3">
                <i class="fa-solid fa-bottle-water text-primary" style="font-size: 70px;"></i>

                <?php if($id_terlaris == 2): ?>
                    <span class="badge bg-warning text-dark position-absolute top-0 end-0 m-
2">Terlaris</span>
                <?php endif; ?>

                <h4 class="card-title mt-4 fw-bold">1.5 Liter</h4>
                <h5 class="text-success fw-bold">Rp 2.000</h5>
                <a href="<?= base_url('checkout/form/2'); ?>" class="btn btn-primary w-100 mt-3 rounded-
pill"><i class="fa-solid fa-cart-shopping"></i> Pesan</a>
            </div>
        </div>
    </div>

    <div class="col-12 col-md-4 mb-4">
        <div class="card h-100 shadow-sm <?= ($id_terlaris == 3) ? 'border-primary border-2' : 'border-0';
?>">
            <div class="card-body text-center mt-3">
                <i class="fa-solid fa-jug-detergent text-primary" style="font-size: 80px;"></i>

                <?php if($id_terlaris == 3): ?>
                    <span class="badge bg-warning text-dark position-absolute top-0 end-0 m-
2">Terlaris</span>
                <?php endif; ?>

                <h4 class="card-title mt-4 fw-bold">Galon 19 Liter</h4>
                <h5 class="text-success fw-bold">Rp 5.000</h5>
                <a href="<?= base_url('checkout/form/3'); ?>" class="btn btn-primary w-100 mt-3 rounded-
pill"><i class="fa-solid fa-cart-shopping"></i> Pesan</a>
            </div>
        </div>
    </div>
</div>
<?= $this->endSection(); ?>
```

Views/pages/login.php

```
<?= $this->extend('layout/template'); ?>

<?= $this->section('content'); ?>
<div class="row justify-content-center mt-5">
    <div class="col-12 col-md-5">
        <div class="card shadow-sm border-0 rounded-4">
            <div class="card-body p-5">
                <div class="text-center mb-4">
                    <i class="fa-solid fa-circle-user text-primary" style="font-size: 60px;"></i>
                    <h3 class="fw-bold mt-3">Selamat Datang</h3>
```

```
            <p class="text-muted">Silakan masuk menggunakan Username Anda</p>
        </div>

        <?php if(session()->getFlashdata('success')): ?>
            <div class="alert alert-success text-center rounded-3">
                <?= (string) session()->getFlashdata('success'); ?>
            </div>
        <?php endif; ?>

        <?php if(session()->getFlashdata('error')): ?>
            <div class="alert alert-danger text-center rounded-3">
                <?= (string) session()->getFlashdata('error'); ?>
            </div>
        <?php endif; ?>

        <form action="<?= base_url('login/process'); ?>" method="POST">
            <?= csrf_field(); ?>

            <div class="mb-3">
                <label class="form-label fw-bold">Username</label>
                <input type="text" class="form-control form-control-lg rounded-pill" autocomplete="off"
name="username" placeholder="Masukkan username" required>
            </div>
            <div class="mb-4">
                <label class="form-label fw-bold">Password</label>
                <input type="password" class="form-control form-control-lg rounded-pill"
autocomplete="off" name="password" placeholder="******" required>
            </div>
            <button type="submit" class="btn btn-primary btn-lg w-100 rounded-pill fw-bold">
                <i class="fa-solid fa-right-to-bracket"></i> Masuk
            </button>
        </form>

        <div class="text-center mt-4">
            <p>Belum punya akun? <a href="<?= base_url('register'); ?>" class="text-decoration-none
fw-bold">Daftar di sini</a></p>
        </div>
    </div>
  </div>
</div>
<?= $this->endSection(); ?>
```

Views/pages/monitoring.php

```
<?php
/**
 * @var array $trx
 */
```

```
?>
<?= $this->extend('layout/template'); ?>
<?= $this->section('content'); ?>

<div class="row justify-content-center text-center mt-5 mb-5">
    <div class="col-12 col-md-6">
        <div class="card shadow-lg border-0 rounded-4 p-5" id="layarMonitor">

            <i class="fa-solid fa-hourglass-half text-warning fa-spin" style="font-size: 80px;" id="iconStatus"></i>
            <h2 class="fw-bold mt-4 text-warning" id="teksStatus">Menunggu Pengisian Air...</h2>
            <p class="text-muted" id="subTeks">Silakan letakkan wadah Anda di bawah keran.</p>

        </div>
    </div>
</div>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<script>
    let isSelesai = false;
    let intervalPolling = setInterval(cekStatusDatabase, 4000);

    function cekStatusDatabase() {
        if (isSelesai) return;

        $.ajax({
            url: "<?= base_url('monitoring/get_status/' . $trx['payment_ref']); ?>",
            type: "GET",
            dataType: "json",
            cache: false,
            success: function(data) {
                let statusSaatIni = data.status_pengisian || data.status;
                let qtyTotal = data.qty || 1;
                let qtySelesai = data.qty_selesai || 0;
                let botolKe = qtySelesai + 1;

                if (botolKe > qtyTotal) botolKe = qtyTotal;

                if(statusSaatIni === "Sedang Mengisi" || statusSaatIni === "pumping") {
                    $('#iconStatus').attr('class', 'fa-solid fa-faucet-drip text-primary fa-beat');
                    $('#teksStatus').attr('class', 'fw-bold mt-4 text-primary').text('Air Sedang Diisi...');
                    $('#subTeks').text('Mengisi wadah ke-' + botolKe + ' dari total ' + qtyTotal + ' pesanan.');
                }
                else if(statusSaatIni === "Menunggu Botol Selanjutnya") {
                    $('#iconStatus').attr('class', 'fa-solid fa-bottle-water text-info fa-bounce');
                    $('#teksStatus').attr('class', 'fw-bold mt-4 text-info').text(qtySelesai + ' Wadah Selesai!');
                    $('#subTeks').html('<b>Silakan letakkan wadah ke-' + botolKe + ' dari total ' + qtyTotal + ' di
bawah keran.</b>');
                }
```

```
            else if(statusSaatIni === "Selesai" || statusSaatIni === "selesai") {
                isSelesai = true;
                clearInterval(intervalPolling);
                $('#iconStatus').attr('class', 'fa-solid fa-circle-check text-success fa-bounce');
                $('#teksStatus').attr('class', 'fw-bold mt-4 text-success').text('Semua Pesanan Selesai!');
                $('#subTeks').html('<div class="alert alert-success mt-3 rounded-pill fw-bold"><i class="fa-solid
fa-spinner fa-spin"></i> Total ' + qtyTotal + ' wadah berhasil diisi. Kembali ke menu awal...</div>');

                setTimeout(function() {
                    window.location.href = '<?= base_url('/'); ?>';
                }, 3000);
            }
            else {
                $('#iconStatus').attr('class', 'fa-solid fa-hourglass-half text-warning fa-spin');
                $('#teksStatus').attr('class', 'fw-bold mt-4 text-warning').text('Menunggu Pengisian Air...');
                $('#subTeks').text('Silakan letakkan wadah ke-' + botolKe + ' dari ' + qtyTotal + ' di bawah
keran.');
            }
        }
    });
  }
</script>

<?= $this->endSection(); ?>
```

Views/pages/order_form.php

```
<?= $this->extend('layout/template'); ?>

<?= $this->section('content'); ?>
<div class="row justify-content-center mt-5 mb-5">
  <div class="col-12 col-md-6">
    <div class="card shadow-sm border-0 rounded-4">
      <div class="card-body p-5">
        <div class="text-center mb-4">
          <i class="fa-solid fa-truck-fast text-primary" style="font-size: 50px;"></i>
          <h3 class="fw-bold mt-3">Detail Pengiriman</h3>
          <p class="text-muted">Anda memesan: <strong class="text-primary"><?=
isset($nama_produk) ? $nama_produk : ''; ?></strong></p>
        </div>

        <form action="<?= base_url('checkout/proses/' . (isset($id_produk) ? $id_produk : '')); ?>"
method="POST" id="formPesanan">
          <?= csrf_field(); ?>

          <div class="mb-3">
            <label class="form-label fw-bold">Jumlah Pesanan (Pcs)</label>
            <input type="number" class="form-control form-control-lg rounded-3" name="qty"
value="1" min="1" required>
```

```
                </div>

                <div class="mb-3">
                    <label class="form-label fw-bold">Nama Penerima</label>
                    <input type="text" class="form-control form-control-lg rounded-3" name="delivery_name"
value="<?= session()->get('name'); ?>" required>
                </div>

                <div class="mb-4">
                    <label class="form-label fw-bold">Alamat Lengkap Pengiriman</label>
                    <textarea class="form-control rounded-3" name="delivery_address" rows="4"
placeholder="Masukkan nama jalan, nomor rumah, RT/RW, dan patokan..." required></textarea>
                </div>

                <button type="submit" class="btn btn-primary btn-lg w-100 rounded-pill fw-bold"
onclick="this.disabled=true; this.innerHTML='<i class=\'fa-solid fa-spinner fa-spin\'></i> Memproses ke
Xendit...'; this.form.submit();">
                    <i class="fa-solid fa-wallet"></i> Lanjut Pembayaran (Xendit)
                </button>
                <a href="<?= base_url('/'); ?>" class="btn btn-light btn-lg w-100 rounded-pill fw-bold mt-2">
                    Batal
                </a>
            </form>

        </div>
    </div>
  </div>
</div>
<?= $this->endSection(); ?>
```

Views/pages/register.php

```
<?= $this->extend('layout/template'); ?>

<?= $this->section('content'); ?>
<div class="row justify-content-center mt-5">
  <div class="col-12 col-md-5">
    <div class="card shadow-sm border-0 rounded-4">
      <div class="card-body p-5">
        <div class="text-center mb-4">
          <i class="fa-solid fa-user-plus text-success" style="font-size: 60px;"></i>
          <h3 class="fw-bold mt-3">Daftar Akun Baru</h3>
          <p class="text-muted">Buat akun untuk mulai memesan air</p>
        </div>

        <?php if(session()->getFlashdata('error')): ?>
          <div class="alert alert-danger text-center">
            <?= (string) session()->getFlashdata('error') ?>
          </div>
```

```
            <?php endif; ?>

            <form action="<?= base_url('register/process'); ?>" method="POST">
                <?= csrf_field(); ?>

                <div class="mb-3">
                    <label class="form-label fw-bold">Nama Lengkap</label>
                    <input type="text" class="form-control form-control-lg rounded-pill" autocomplete="off"
name="name" placeholder="Masukkan nama Anda" required>
                </div>
                <div class="mb-3">
                    <label class="form-label fw-bold">Alamat Email</label>
                    <input type="email" class="form-control form-control-lg rounded-pill" autocomplete="off"
name="email" placeholder="contoh@email.com" required>
                </div>
                <div class="mb-3">
                    <label class="form-label fw-bold">Username</label>
                    <input type="text" class="form-control form-control-lg rounded-pill" autocomplete="off"
name="username" placeholder="Buat username untuk login" required>
                </div>
                <div class="mb-4">
                    <label class="form-label fw-bold">Password</label>
                    <input type="password" class="form-control form-control-lg rounded-pill"
autocomplete="off" name="password" placeholder="Buat password" required>
                </div>
                <button type="submit" class="btn btn-success btn-lg w-100 rounded-pill fw-bold">
                    <i class="fa-solid fa-check"></i> Daftar Sekarang
                </button>
            </form>

            <div class="text-center mt-4">
                <p>Sudah punya akun? <a href="<?= base_url('login'); ?>" class="text-decoration-none fw-
bold">Masuk di sini</a></p>
            </div>
        </div>
    </div>
</div>
<?= $this->endSection(); ?>
```

SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER
(STMIK) PALANGKARAYA
Jl. G. Obos No.114 Telp.0536-3224593, 3225515 Fax.0536-3225515 Palangka Raya
email : humas@stmikplk.ac.id - website : www.stmikplk.ac.id

BERITA ACARA
UJIAN TUGAS AKHIR

Periode : April 2026

Pada hari ini Kamis, 18 Juni 2026 Pukul 10.15 WIB sampai dengan ..... WIB telah dilaksanakan Ujian Tugas Akhir mahasiswa atas nama :

Nama Mahasiswa	:	Daniel Setiawan Purnomo
N I M	:	C2255201107
Program Studi	:	Teknik Informatika
Tahun Angkatan	:	2022
Judul TA	:	RANCANG BANGUN SMART DISPENSER BERBASIS IOT MENGGUNAKAN XENDIT PADA DEPOT AIR MINUM BLUEBUB

Dosen Penguji :

	Nama		Nilai	Tanda Tangan
1.	Ir. Abdul Hadi, S.T., M.Kom	=	.....	(.....)
2.	Dr. Maura. Widyaningsih, S.Kom., M.Cs	=	.....	(.....)
3.	Hotmian Sitohang, M.Kom	=	.....	(.....)
4.	Herkules, S.Kom., M.Cs	=	.....	(.....)
5.	Elia Zakharia, M.T	=	.....	(.....)

Hasil Ujian : **LULUS** / ~~**TIDAK LULUS**~~ *) NILAI = 82,136

Revisi (~~Tidak~~ Ada / Minor / ~~Mayor~~ *)

Catatan Penting :
1. Lama Perbaikan : ..... 6 ..... hari
2. Jika lebih dari 1 (satu) bulan dikenakan sanksi berupa denda sebesar Rp. 600.000,- (Enam ratus ribu rupiah) per bulan dari tanggal ujian.
3. Jika lebih dari 3 (tiga) bulan dari tanggal ujian maka hasil ujian dibatalkan dan wajib mengajukan judul dan pembimbing baru.

Demikian Berita Acara ini dibuat, agar dapat digunakan sebagaimana semestinya.

Palangka Raya, Kamis, 18 Juni 2026

Ketua Penguji,

Ir. Abdul Hadi, S.T., M.Kom
NIK. 198505102021101

Mahasiswa,

Daniel Setiawan Purnomo
NIM. C2255201107

Mengetahui :
Ketua Program Studi Teknik Informatika,

Ir. Abdul Hadi, S.T., M.Kom
NIK. 198505102021101

Tembusan:
Mahasiswa yang bersangkutan.
Berita acara ini diberikan setelah menyelesaikan semua revisi dari semua penguji
**) Coret yang tidak perlu*