

PENGENDALIAN PENGISIAN AIR OTOMATIS MENGGUNAKAN ESP32, MQTT, DAN KODULAR DENGAN NOTIFIKASI *WHATSAPP* DAN *DATA LOGGING* MELALUI *GOOGLE SHEETS*

Aryo Wijoseno^{1*}, Henri P. Uranus², Julinda Pangaribuan³

^{1,2,3}Program Studi Teknik Elektro, Fakultas Sains dan Teknologi, Universitas Pelita Harapan; Jl. M. H. Thamrin Boulevard 1100 Lippo Village Tangerang 15811 Indonesia; Telp (021) 546 0901

Keywords:

IoT;
ESP32;
MQTT;
Kodular.

Correspondent Email:

01035230007@student.uph.edu



JITET is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Abstrak. Pengisian air toren secara konvensional yang mengandalkan pelampung mekanik memiliki keterbatasan dalam hal akurasi dan fleksibilitas pemantauan. Penelitian ini merancang sistem pengisian toren air otomatis berbasis Internet of Things (IoT) dengan mikrokontroler ESP32, sensor ultrasonik AJ-SR04M, dan katup motorized ball valve. Sistem dikendalikan melalui protokol MQTT dan antarmuka aplikasi Kodular untuk memungkinkan kontrol dan pemantauan jarak jauh secara real-time. Data ketinggian air dikirimkan ke pengguna, serta dicatat ke dalam Google Spreadsheet sebagai histori pemantauan. Hasil pengujian menunjukkan bahwa sistem mampu mendeteksi ketinggian air dengan akurasi stabil ($\pm 0,3$ cm), merespons otomatisasi pengisian dengan tingkat keberhasilan 100%, dan melakukan pencatatan data tanpa kehilangan paket. Sistem ini terbukti bekerja secara andal dan efisien, serta memberikan keunggulan dibandingkan metode konvensional.

Abstract. Conventional water tank filling systems that rely on mechanical float switches are limited in accuracy and remote monitoring capabilities. This study designs an automatic water tank filling system based on the Internet of Things (IoT) using the ESP32 microcontroller, AJ-SR04M ultrasonic sensor, and a motorized ball valve. The system is controlled via the MQTT protocol and utilizes a Kodular-based application interface to enable real-time remote monitoring and control. Water level data is sent to the user and logged into Google Spreadsheets for historical tracking. Testing results show that the system detects water level accurately (± 0.3 cm), responds reliably to automatic filling commands with a 100% success rate, and transmits data without packet loss. The system proves to be reliable, efficient, and superior to conventional methods.

1. PENDAHULUAN

Pengelolaan air secara efisien menjadi tantangan penting di berbagai sektor, termasuk rumah tangga. Pada sistem konvensional, pengisian toren air umumnya masih menggunakan pelampung mekanik yang tidak memberikan informasi ketinggian air secara

akurat dan tidak memungkinkan pemantauan jarak jauh [1]. Hal ini berisiko menyebabkan pemborosan akibat pengisian berlebih, serta menyulitkan pengguna dalam memastikan ketersediaan air secara *real-time*.

Perkembangan teknologi *Internet of Things* (IoT) membuka peluang untuk menciptakan sistem pengisian air otomatis yang lebih cerdas

dan responsif [2]. IoT memungkinkan integrasi antara sensor, mikrokontroler, dan aplikasi *mobile* untuk melakukan monitoring dan pengendalian dari jarak jauh secara otomatis [3]. Beberapa penelitian sebelumnya telah memanfaatkan NodeMCU dan sensor ultrasonik untuk kebutuhan serupa [4], namun keterbatasan fitur dan akurasi sistem masih menjadi isu yang perlu disempurnakan.

Penelitian ini merancang sistem pengisian toren otomatis menggunakan ESP32, sensor ultrasonik AJ-SR04M, dan katup *motorized ball valve*. Sistem ini dikendalikan melalui protokol komunikasi MQTT dan antarmuka *Kodular*, yang memungkinkan pemantauan kondisi toren secara *real-time* melalui perangkat *mobile* serta pencatatan data ke *Google Spreadsheet*. Sistem dilengkapi dengan notifikasi melalui *WhatsApp* untuk alarm. Penelitian ini bertujuan memberikan solusi yang efisien, fleksibel, dan lebih akurat dibandingkan metode konvensional.

2. TINJAUAN PUSTAKA

Beberapa penelitian sebelumnya telah mengembangkan sistem pengisian air otomatis berbasis IoT. Herdiana dan Triatna (2020) merancang sistem monitoring ketinggian air menggunakan NodeMCU dan sensor ultrasonik yang terintegrasi dengan aplikasi *Blynk*. Sistem tersebut berfungsi secara otomatis, namun kurang optimal pada skala tangki besar karena keterbatasan sensor [4]. Pamungkas dan Setyadjit (2023) menggunakan ESP32 dan sensor flow meter untuk sistem isi ulang galon otomatis, namun tidak menyediakan fleksibilitas pemantauan ketinggian air secara akurat [5].

Hakim dan Ashshidiq (2024) mengembangkan sistem kontrol level air berbasis ESP32 dengan aplikasi *Blynk*, tetapi sistem sangat tergantung pada kestabilan jaringan internet dan belum diuji untuk kondisi dinamis [6]. Sementara itu, Yazid *et al.* (2023) mengintegrasikan ESP32 dengan *Kodular* untuk pemantauan suhu dan tinggi air pada media akuaponik. Penelitian tersebut menunjukkan bahwa *Kodular* dapat menjadi alternatif yang efisien dan fleksibel dalam pengembangan antarmuka IoT [7].

Dalam konteks protokol komunikasi, MQTT banyak digunakan karena ringan, andal, dan

hemat bandwidth. Protokol ini memungkinkan pertukaran data yang cepat antara perangkat IoT dan aplikasi monitoring [8]. Sedangkan dari sisi aktuasi, katup *motorized ball valve* dinilai lebih tahan lama dibandingkan sistem pelampung karena mengurangi risiko kegagalan mekanik pada saklar tradisional.

Penelitian ini mengintegrasikan ESP32, sensor AJ-SR04M, *motorized ball valve*, dan *Kodular* melalui protokol MQTT untuk merancang sistem pengisian toren otomatis yang lebih stabil, akurat, dan dapat diakses dari jarak jauh.

2.1 Internet of Things (IoT)

Internet of Things (IoT) adalah suatu konsep yang memungkinkan perangkat elektronik berkomunikasi dan bertukar data melalui jaringan internet tanpa campur tangan langsung manusia. Dalam sistem otomatisasi pengisian air, IoT memberikan kemampuan monitoring dan pengendalian secara *real-time*, meningkatkan efisiensi serta memungkinkan pengambilan keputusan berbasis data [9]. Teknologi ini banyak diterapkan di berbagai bidang, termasuk manajemen air rumah tangga, pertanian, hingga industri.

2.2 Mikrokontroler ESP32

Mikrokontroler ESP32 adalah papan pengembangan yang dilengkapi dengan prosesor *dual-core* dan konektivitas Wi-Fi serta *Bluetooth* terintegrasi [10]. Dibandingkan dengan NodeMCU ESP8266, ESP32 memiliki kecepatan pemrosesan lebih tinggi dan lebih banyak pin GPIO, sehingga lebih fleksibel untuk integrasi multi-sensor [11]. Perangkat ini berfungsi sebagai pusat pemrosesan data sensor dan pengendali aktuator dalam sistem otomatisasi toren air berbasis IoT. Gambar 1 merupakan bentuk dari mikrokontroler ESP32.



Gambar 1. Mikrokontroler ESP32

2.3 Sensor Ultrasonik AJ-SR04M

Sensor ultrasonik AJ-SR04M, yang memiliki karakteristik tahan air dan cocok untuk aplikasi luar ruangan, digunakan untuk mendeteksi ketinggian air tanpa kontak langsung [12]. Sensor ini bekerja dengan prinsip pantulan gelombang ultrasonik, dan nilai jarak yang diperoleh digunakan untuk menentukan status volume air di dalam toren [13]. Gambar 2 menampilkan bentuk dari sensor ultrasonik AJ-SR04M.



Gambar 2. Sensor Ultrasonik AJ-SR04M

2.4 Protokol MQTT (Message Queuing Telemetry Transport)

Protokol MQTT (*Message Queuing Telemetry Transport*) merupakan protokol komunikasi ringan berbasis *publish-subscribe* yang sangat efisien untuk aplikasi IoT [14]. MQTT digunakan dalam sistem ini sebagai jembatan komunikasi antara ESP32 dan aplikasi antarmuka pengguna seperti *Kodular*, memungkinkan transmisi data yang cepat dan andal, bahkan dalam kondisi jaringan yang tidak stabil [8].

2.5 Aplikasi Kodular

Aplikasi *Kodular* adalah platform pengembangan visual yang memungkinkan pembuatan aplikasi Android tanpa perlu menulis kode secara manual [15]. Dalam sistem ini, *Kodular* digunakan untuk membuat antarmuka pengguna yang menampilkan status air, kontrol katup, dan pengaturan tinggi toren,

serta menerima data dari ESP32 melalui MQTT.

2.6 Relay

Modul Relay merupakan komponen elektronika yang berfungsi sebagai saklar elektrik yang dikendalikan oleh sinyal tegangan rendah dari mikrokontroler, seperti ESP32 [16]. Dalam sistem otomatisasi pengisian toren air, relay digunakan untuk mengendalikan katup motor listrik (*motorized ball valve*) yang bekerja pada tegangan 220V. Modul relay memiliki kemampuan isolation (isolasi) antara sirkuit kendali dan beban, sehingga melindungi mikrokontroler dari lonjakan arus. Relay yang digunakan bersifat *normally closed* atau *normally open*, tergantung dari kebutuhan sistem. Gambar 3 menampilkan bentuk dari modul relay yang akan digunakan.



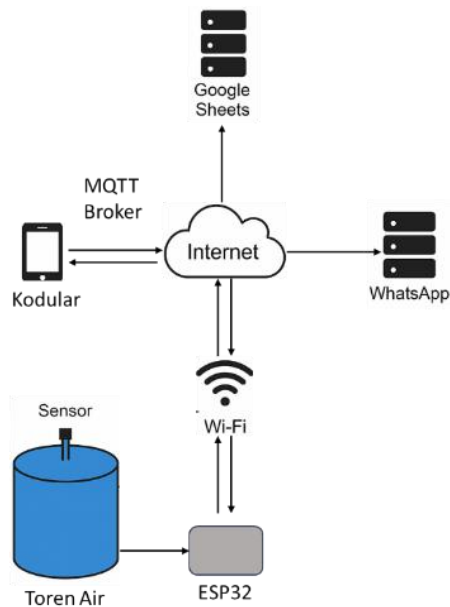
Gambar 3 Relay

2.7 Motorized Ball Valve

Katup elektrik *Motorized Ball Valve* digunakan sebagai aktuator untuk membuka atau menutup aliran air berdasarkan perintah dari ESP32. Katup jenis ini memiliki keunggulan dalam hal daya tahan dan kemampuan kontrol yang lebih presisi dibandingkan katup berbasis solenoid atau pelampung mekanik konvensional [17]. Gambar 4 menampilkan katup yang digunakan pada penelitian ini.



Gambar 4 Motorized Ball Valve



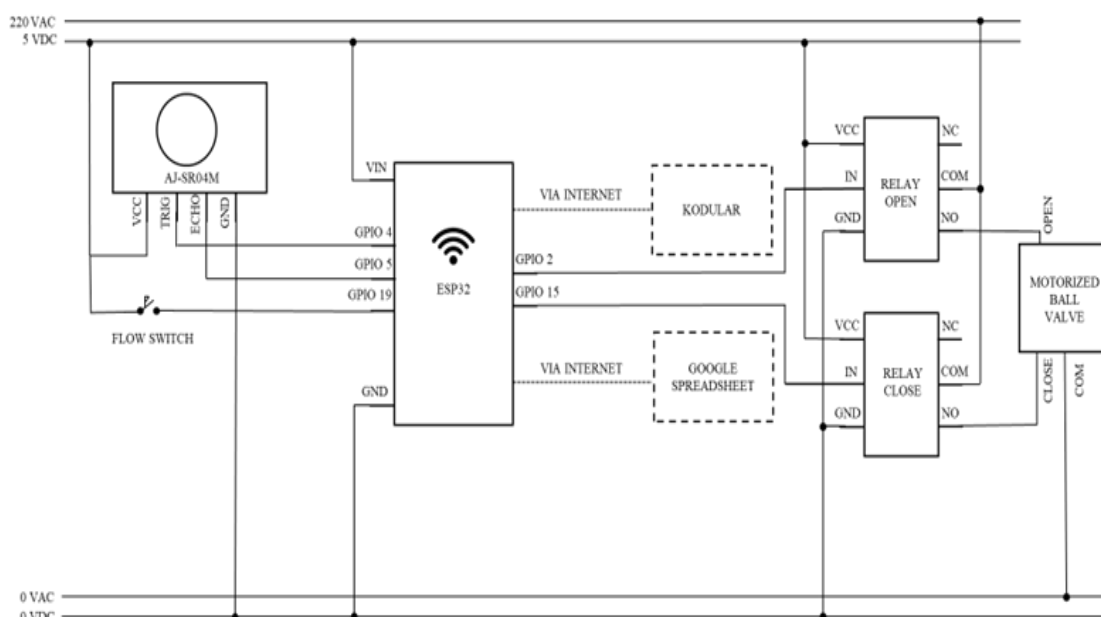
Gambar 6 Konsep Perancangan Sistem

3. METODE PENELITIAN

Penelitian ini diawali dengan studi literatur dari berbagai referensi seperti jurnal ilmiah, skripsi terdahulu, dan dokumen teknis perangkat IoT. Studi ini bertujuan untuk memahami konsep dasar sistem pengisian air otomatis berbasis IoT, termasuk pemanfaatan sensor ultrasonik, mikrokontroler ESP32, protokol MQTT, serta pemrograman aplikasi menggunakan *Kodular*.

Gambar 5 menunjukkan konsep sistem pengisian air toren otomatis berbasis IoT. Sensor ultrasonik mengukur ketinggian air dan mengirimkan data ke mikrokontroler ESP32. ESP32 kemudian meneruskan data tersebut ke aplikasi *mobile* melalui internet menggunakan protokol MQTT, mencatatnya ke *Google Sheets* sebagai data historis, dan mengirimkan notifikasi ke *WhatsApp* jika terjadi kondisi tertentu seperti air penuh atau habis. Sistem ini memungkinkan pemantauan dan pengendalian toren air secara *real-time*, otomatis, dan jarak jauh.

Tahapan selanjutnya adalah merancang sistem baik dari sisi perangkat keras (*hardware*) maupun perangkat lunak (*software*) seperti pada Gambar 6. Rancangan perangkat keras meliputi integrasi antara sensor ultrasonik AJ-SR04M, mikrokontroler ESP32, *flow switch*, modul relay, dan katup *motorized ball valve* 12V. Sedangkan perangkat lunak dirancang menggunakan *Arduino IDE* untuk

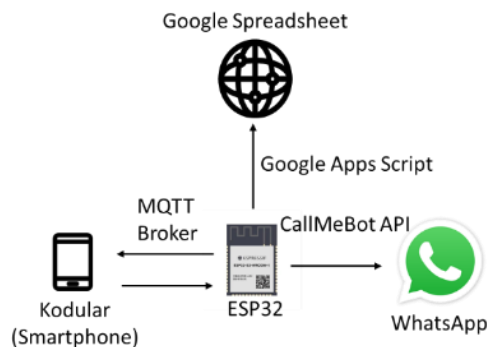


Gambar 5 Skema Rangkaian

pemrograman ESP32, serta *Kodular* sebagai antarmuka monitoring berbasis aplikasi *mobile*.

Setelah perancangan selesai, dilakukan proses uji coba sistem. Pengujian dilakukan dalam beberapa skenario, seperti pengujian pembacaan ketinggian air oleh sensor, keakuratan pembacaan, respon katup terhadap perintah dari mikrokontroler, serta kestabilan koneksi pengiriman data melalui protokol MQTT ke aplikasi *Kodular*.

Apabila hasil pengujian belum sesuai dengan tujuan, maka dilakukan perbaikan baik pada sisi pemrograman maupun perakitan *hardware*. Setelah sistem dinyatakan berjalan dengan baik dan sesuai dengan skenario yang diinginkan, tahap akhir adalah melakukan analisis data dari hasil pengujian.



Gambar 7 Alur Komunikasi Sistem

Berdasarkan Gambar 7, sistem ini menggunakan ESP32 sebagai pengendali utama dan protokol MQTT untuk komunikasi data dua arah antara perangkat dan aplikasi *Kodular*. Saat pengguna mengirim perintah dari aplikasi, data dikirim ke broker MQTT (broker.emqx.io), lalu diteruskan ke ESP32 yang mengeksekusi perintah. Sebaliknya, data seperti tinggi air dan status sensor dikirim kembali ke aplikasi secara *real-time*. Selain itu, ESP32 juga mengirimkan data secara berkala ke *Google Spreadsheet* melalui *Google Apps Script* untuk pencatatan historis. ESP32 juga mengirimkan notifikasi ke *WhatsApp* ketika terjadi hal-hal seperti air penuh, air habis, atau air tidak mengalir.

Dengan pendekatan ini, sistem tidak hanya mampu bekerja secara otomatis dan *real-time*, tetapi juga memberikan fleksibilitas pemantauan jarak jauh dan efisiensi dalam penggunaan air.

4. HASIL DAN PEMBAHASAN

4.1 PEMBAHASAN KODE PROGRAM ESP32

Proses pembacaan ketinggian air pada sistem dilakukan melalui sensor ultrasonik AJ-SR04M yang terhubung ke mikrokontroler ESP32. Sensor ini memanfaatkan prinsip pantulan gelombang ultrasonik untuk mengukur jarak antara permukaan air dengan sensor, lalu dikonversi menjadi nilai tinggi air dalam toren.

Potongan kode pada Gambar 8 menunjukkan fungsi *readUltrasonicSensor()* yang digunakan untuk membaca dan menghitung jarak.

```

void readUltrasonicSensor() {
  const int numReadings = 10;
  float totalDistance = 0.0;
  Sensor_Status = "Success";

  for (int i = 0; i < numReadings; i++) {
    digitalWrite(TRIG_PIN, LOW); delayMicroseconds(2);
    digitalWrite(TRIG_PIN, HIGH); delayMicroseconds(10);
    digitalWrite(TRIG_PIN, LOW);

    float duration = pulseIn(ECHO_PIN, HIGH);
    if (duration == 0) {
      Sensor_Status = "Fail";
      totalDistance = 0.0;
      break;
    } else {
      totalDistance += duration * 0.034 / 2;
    }
    delay(50);
  }

  if (Sensor_Status == "Success") {
    Distance = totalDistance / numReadings;
    Water_Level = Tank_Height - Distance;
    if (Distance < 0 || Distance > Tank_Height) {
      Sensor_Status = "Undetected";
      Water_Level = Distance = 0.0;
    }
  } else {
    Water_Level = Distance = 0.0;
  }
}

```

Gambar 8 Kode program pembacaan sensor

Fungsi ini menjalankan proses pengukuran sebanyak 10 kali dan menghitung nilai rata-rata jarak (dalam satuan cm) dari permukaan sensor ke permukaan air. Pembacaan jarak didapat dari waktu tempuh pulsa ultrasonik yang dipantulkan (melalui perintah *pulseIn()*), dikalikan dengan kecepatan suara di udara (0,034 cm/μs), dan dibagi dua (karena gelombang menempuh perjalanan bolak-balik).

Nilai jarak tersebut kemudian dikurangkan dari tinggi toren (*Tank_Height*) untuk memperoleh nilai aktual dari tinggi air. Untuk menjaga akurasi, sistem juga memeriksa validitas pembacaan: jika tidak ada pantulan (durasi = 0) atau nilai berada di luar batas logis, maka status sensor akan diberi label "Fail" atau

"Undetected", dan sistem akan mengabaikan pembacaan tersebut.

Agar sistem dapat mencatat data secara historis, ESP32 dirancang untuk mengirimkan informasi pembacaan sensor ke *Google Sheets* menggunakan layanan *Google Apps Script*. Komunikasi antara mikrokontroler dan *spreadsheet* dilakukan melalui permintaan HTTP GET yang ditujukan ke *endpoint Google Script* yang telah disiapkan sebelumnya.

Gambar 9 menampilkan program yang bertanggung jawab untuk melakukan pengiriman data setiap 60 detik.

```
void kirimDataTiapMenit() {
    static unsigned long lastSendTime = 0;
    if (millis() - lastSendTime > 60000) {
        String url = Web_App_URL + "?sts=write";
        url += "&sensor_status=" + Sensor_Status;
        url += "&water_level=" + String(Water_Level);
        url += "&distance=" + String(Distance);

        HTTPClient http;
        http.begin(url);
        int code = http.GET();
        Serial.println("Sheets status: " + String(code));
        http.end();

        lastSendTime = millis();
    }
}
```

Gambar 9 Kode program pengiriman data ke Google Sheets

Pada bagian ini, URL *endpoint* disusun secara dinamis dengan menyisipkan tiga parameter utama: *sensor_status*, *water_level*, dan *distance*. Masing-masing mewakili status pembacaan sensor ultrasonik, tinggi air aktual dalam toren, serta jarak dari sensor ke permukaan air. Data tersebut dikirimkan ke skrip *Google Apps Script* menggunakan metode GET yang ringan dan sesuai untuk mikrokontroler.

Jika permintaan berhasil, data akan otomatis masuk ke lembar kerja *Google Sheets* dan ditampilkan baik dalam bentuk log historis maupun ringkasan data terbaru. Nilai *code* dari respons HTTP akan ditampilkan melalui Serial Monitor, memungkinkan pengembang mengetahui status koneksi (misalnya 200 untuk sukses, 404 jika URL salah, atau 403 jika akses dibatasi).

Agar sistem dapat menjalankan fungsi pemantauan dan pengendalian jarak jauh secara *quasi real-time*, mikrokontroler ESP32 dikonfigurasi untuk terhubung dengan broker MQTT menggunakan *library* MQTT.h.

Protokol MQTT (*Message Queuing Telemetry Transport*) dipilih karena ringan dan cocok untuk komunikasi IoT dengan *bandwidth* rendah. Inisialisasi koneksi MQTT dilakukan seperti pada Gambar 10.

```
client.begin("broker.emqx.io", net);
client.onMessage(terimaData);
while (!client.connect("autofilltoreny001")) {
    Serial.print(".");
    delay(1000);
}
Serial.println("MQTT terhubung.");
client.subscribe("autofilltoren/#");
```

Gambar 10 Inisialisasi koneksi MQTT

ESP32 berperan sebagai *publisher* dan *subscriber*. Saat terhubung, ESP32 akan berlangganan ke seluruh topik dengan awalan *autofilltoren/*, termasuk:

- *autofilltoren/mode*
- *autofilltoren/keran*
- *autofilltoren/setlevel*

Topik tersebut digunakan untuk menerima perintah dari aplikasi *Kodular*. Fungsi *callback terimaData()* menangani logika setiap topik seperti yang ditampilkan pada Gambar 11.

```
void terimaData(String &topic, String &payload) {
    if (topic == "autofilltoren/keran" && !modeAuto) {
        if (payload == "nyala") bukaValve();
        else if (payload == "mati") tutupValve();
    } else if (topic == "autofilltoren/setlevel") {
        float newHeight = payload.toFloat();
        if (newHeight >= 30.0 && newHeight <= 107.0) Tank_Height = newHeight;
    } else if (topic == "autofilltoren/mode") {
        modeAuto = (payload == "auto");
        if (!modeAuto) matikanValve();
    }
}
```

Gambar 11 Fungsi terima data

Sementara itu, ESP32 juga mengirim data (*publish*) ke beberapa topik status agar dapat ditampilkan ke pengguna melalui *Kodular* seperti yang dapat dilihat pada Gambar 12.

```
void kirimMQTT() {
    client.publish("autofilltoren/waterlevel", String(Water_Level));
    client.publish("autofilltoren/tinggitoran", String(Tank_Height));
    client.publish("autofilltoren/sensorstatus", Sensor_Status);
    client.publish("autofilltoren/statusair", Status_Air);
}
```

Gambar 12 Publish data MQTT

Dengan skema ini, sistem mendukung komunikasi dua arah yang sinkron:

- *Kodular* sebagai *subscriber* menerima status sistem (tinggi air, status sensor, aliran air, dan indikator katup).
- *Kodular* sebagai *publisher* dapat mengirimkan perintah ke ESP32 untuk

membuka atau menutup katup secara manual, mengubah mode ke otomatis, atau menyesuaikan tinggi batas toren.

Salah satu fitur tambahan dari sistem adalah kemampuannya mengirimkan notifikasi otomatis ke aplikasi *WhatsApp*. Tujuan dari fitur ini adalah untuk memberikan peringatan dini kepada pengguna ketika sistem mendeteksi kondisi kritis, seperti air hampir habis atau air tidak mengalir meskipun katup dalam keadaan terbuka. Notifikasi ini dikirim melalui integrasi layanan pihak ketiga yaitu *CallMeBot*, menggunakan metode *HTTP request* dari ESP32.

```
void kirimPesanWA(String pesan) {
  if (WiFi.status() == WL_CONNECTED) {
    String url = "https://api.callmebot.com/whatsapp.php?phone=" + phone_number +
      "&text=" + urlencode(pesan) + "&apikey=" + apiKey;

    HTTPClient http;
    http.begin(url);
    int code = http.GET();
    Serial.println("Kirim WA: " + pesan + " | Status: " + String(code));
    http.end();
  }
}

void cekNotifikasiWA() {
  if (Water_Level >= tank_Height && !sudahPenuh && millis() > 100000) {
    if (millis() - lastWASentTime > 60000) {
      kirimPesanWA("Air sudah terisi penuh");
      sudahPenuh = true;
      hampirHabis = false;
      lastWASentTime = millis();
    }
  }

  if (Water_Level < 30.0 && !hampirHabis) {
    if (millis() - lastWASentTime > 50000) {
      kirimPesanWA("Air sudah mau habis");
      hampirHabis = true;
      sudahPenuh = false;
      lastWASentTime = millis();
    }
  }
}
```

Gambar 13 Pengiriman notifikasi ke WhatsApp

Gambar 13 menunjukkan implementasi kode untuk mengirim notifikasi ke *WhatsApp* menggunakan fungsi *kirimPesanWA()*. Fungsi ini akan dijalankan ketika kondisi tertentu terpenuhi, misalnya ketika toren telah penuh atau ketika air hampir habis, sebagaimana diperiksa dalam fungsi *cekNotifikasiWA()*. Pengiriman dilakukan melalui *HTTP GET* ke alamat API *CallMeBot* yang telah dirangkai bersama nomor telepon, pesan, dan API key. Fungsi ini hanya akan aktif bila koneksi internet tersedia dan memiliki jeda waktu minimal 60 detik antar notifikasi untuk mencegah spam pesan. Pendekatan ini menjadikan sistem lebih komunikatif dan proaktif dalam menjaga efisiensi penggunaan air dan mencegah gangguan operasional.

4.2 PENGUJIAN SENSOR ULTRASONIK

Pengujian sensor AJ-SR04M bertujuan untuk mengetahui akurasi pembacaan jarak

permukaan air. Sensor ini terhubung dengan mikrokontroler ESP32 dan data dikirimkan setiap 60 detik ke *Google Spreadsheet* secara otomatis. Pengujian dilakukan pada jarak tetap 25 cm untuk mensimulasikan tinggi air konstan. Hasil pengujian dapat dilihat pada Tabel 1.

Tabel 1 Pengujian Sensor Ultrasonik

Tanggal	Waktu	Status Sensor	Jarak (cm)
10/04/2025	20:07:23	Success	25.10
10/04/2025	20:08:25	Success	25.04
10/04/2025	20:09:27	Success	25.08
10/04/2025	20:10:29	Success	24.95
10/04/2025	20:11:31	Success	24.90
10/04/2025	20:12:33	Success	25.00
10/04/2025	20:13:35	Success	24.98
10/04/2025	20:14:37	Success	25.03
10/04/2025	20:15:39	Success	25.06
10/04/2025	20:16:41	Success	25.09
10/04/2025	20:17:43	Success	24.97
10/04/2025	20:18:45	Success	25.11
10/04/2025	20:19:47	Success	25.06
10/04/2025	20:20:49	Success	25.00
10/04/2025	20:21:51	Success	24.92
10/04/2025	20:22:53	Success	24.94
10/04/2025	20:23:55	Success	25.01
10/04/2025	20:24:57	Success	25.03
10/04/2025	20:25:59	Success	25.07
10/04/2025	20:27:01	Success	25.11
10/04/2025	20:28:03	Success	24.91
10/04/2025	20:29:05	Success	24.96
10/04/2025	20:30:07	Success	24.93
10/04/2025	20:31:09	Success	25.02
10/04/2025	20:32:11	Success	25.00
10/04/2025	20:33:13	Success	25.08
10/04/2025	20:34:15	Success	25.04
10/04/2025	20:35:17	Success	24.97

Dari hasil di atas, seluruh status sensor menunjukkan kondisi “*Success*”, tanpa ada gangguan sinyal atau kehilangan data. Rentang pembacaan sensor berkisar antara 24.98 cm hingga 25.11 cm, dengan rata-rata 25.04 cm. Hal ini menunjukkan bahwa sensor memiliki deviasi yang sangat kecil, yaitu ± 0.06 cm, dan cukup akurat untuk aplikasi monitoring air.

4.3 PENGUJIAN KATUP ELEKTRONIK

Pengujian katup *motorized ball valve* dilakukan untuk mengukur waktu respons saat menerima perintah dari ESP32. Katup ini bekerja berdasarkan perintah buka/tutup yang dikirim melalui relay, dikendalikan via aplikasi *Kodular*. Hasil pengujian dapat dilihat pada Tabel 2.

Tabel 2 Pengujian Katup Elektronik

Tanggal	Perintah	Status	Waktu Respons (detik)
07/06/2025	Buka	Berhasil	14.9
07/06/2025	Tutup	Berhasil	15.1
07/06/2025	Buka	Berhasil	15.0
07/06/2025	Tutup	Berhasil	15.2
07/06/2025	Buka	Berhasil	14.8
07/06/2025	Tutup	Berhasil	15.3
07/06/2025	Buka	Berhasil	15.0
07/06/2025	Tutup	Berhasil	14.9

Hasil menunjukkan bahwa rata-rata waktu buka/tutup adalah sekitar 15 detik, sesuai dengan spesifikasi pabrikan. Selama pengujian tidak ditemukan kebocoran atau kegagalan fungsi, sehingga katup ini dinyatakan stabil dan cocok untuk pengisian otomatis.

4.4 PENGUJIAN MIKROKONTROLER ESP32 DAN KONEKSI IOT

Pengujian mikrokontroler ESP32 dilakukan untuk memastikan bahwa seluruh fungsi kontrol dan komunikasi dalam sistem bekerja secara terintegrasi dan stabil. ESP32 berfungsi sebagai pusat kendali yang menerima input dari sensor ultrasonik, memproses data, dan mengirimkan perintah kepada relay untuk mengatur katup *motorized ball valve* secara otomatis. Selain itu, ESP32 juga menangani komunikasi data ke aplikasi *Kodular* melalui protokol MQTT serta pencatatan data ke *Google Spreadsheet* menggunakan *Google Apps Script*. Gambar 14 menampilkan sistem yang sedang diuji.



Gambar 14. Sistem dengan mikrokontroler ESP32 ditempatkan di dalam sebuah kotak plastik

Hasil pengujian menunjukkan bahwa ESP32 mampu membaca data dari sensor ultrasonik secara berkala dan stabil dengan interval pengiriman data ke server setiap 60 detik. Seluruh proses eksekusi perintah, baik otomatis maupun manual, berhasil dijalankan dengan tingkat keberhasilan 100% pada 8 kali percobaan siklus pengisian. Tidak ditemukan gangguan konektivitas selama 1 jam pengujian, dan data berhasil dikirim ke aplikasi *Kodular* dan *Google Spreadsheet* secara *real-time*.

Stabilitas sistem juga terlihat dari konsistensi pembacaan dan eksekusi logika program tanpa terjadi restart mendadak atau hang. Dengan performa tersebut, ESP32 dinilai mampu menjalankan tugas sebagai otak dari sistem otomatisasi pengisian toren air berbasis IoT secara andal dan efisien.

4.5 PENGUJIAN APLIKASI KODULAR

Aplikasi *Kodular* berfungsi sebagai antarmuka utama pengguna untuk memantau dan mengontrol kondisi toren air. Fitur utamanya meliputi: pemilihan mode otomatis/manual, status sensor, kontrol katup, dan set tinggi toren. Gambar 15 merupakan tampilan aplikasi sistem pengisian toren air otomatis.

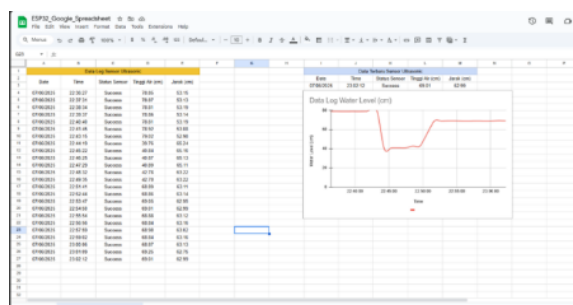


Gambar 15 Tampilan Antarmuka Kodular

Selama pengujian, semua elemen UI bekerja dengan baik. Data yang diterima dari ESP32 melalui MQTT tampil secara *real-time* tanpa lag. Respons terhadap perintah manual juga berlangsung instan, membuktikan bahwa aplikasi *Kodular* dapat menggantikan aplikasi IoT lain seperti *Blynk* dalam sistem *open-source* yang lebih fleksibel.

4.6 PENGUJIAN PENGIRIMAN DATA KE GOOGLE SPREADSHEET

ESP32 mengirimkan data sensor ke *Google Spreadsheet* menggunakan *Google Apps Script* Web App. Data dikirim menggunakan metode HTTP GET dengan interval 60 detik. Gambar 16 merupakan hasil data log yang ditampilkan pada *Google Sheets*.

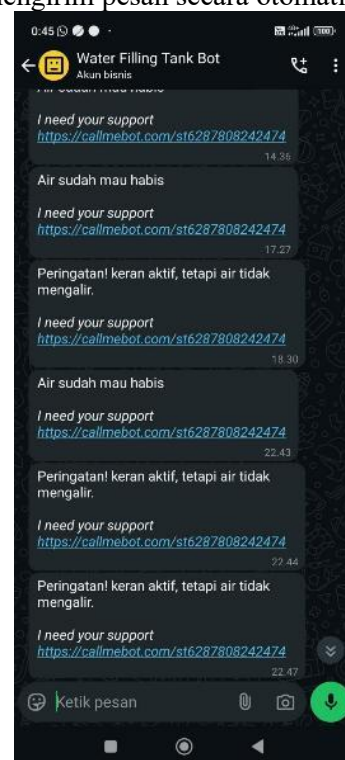


Gambar 16 Tampilan Data Log Google Sheets

Pengujian membuktikan bahwa sistem dapat mencatat data historis secara kontinu tanpa kehilangan data. Hal ini sangat membantu dalam pelacakan performa sistem dan pemantauan penggunaan air. *Google Sheets* memberikan fleksibilitas dalam visualisasi dan analisis data, yang penting untuk pengembangan sistem lebih lanjut.

4.7 PENGUJIAN PENGIRIMAN NOTIFIKASI KE WHATSAPP

Fitur notifikasi *WhatsApp* dirancang sebagai saluran komunikasi tambahan untuk memperingatkan pengguna ketika terjadi kondisi kritis, seperti air hampir habis atau aliran air tidak terdeteksi meskipun keran aktif. Sistem memanfaatkan layanan *CallMeBot* yang terhubung ke ESP32 melalui protokol HTTP untuk mengirim pesan secara otomatis.



Gambar 17 Tampilan Notifikasi WhatsApp

Seperti terlihat pada Gambar 17, sistem berhasil mengirim pesan yang relevan dan tidak berulang secara berlebihan. Pengujian menunjukkan bahwa fungsi ini berjalan stabil dan responsif, serta mampu memberikan informasi penting secara *real-time-like* meskipun pengguna tidak membuka aplikasi utama. Hal ini meningkatkan keandalan sistem dari sisi monitoring dan respons terhadap kondisi tidak normal.

5. KESIMPULAN

Berdasarkan hasil pengujian dan analisis terhadap sistem pengisian toren air otomatis berbasis IoT yang dikembangkan, dapat disimpulkan sebagai berikut:

- Sistem berhasil membaca ketinggian air secara *real-time* menggunakan sensor ultrasonik AJ-SR04M dengan rata-rata deviasi $\pm 0,06$ cm, menunjukkan akurasi yang tinggi untuk kebutuhan monitoring air rumah tangga.
- Proses pengendalian katup elektronik berjalan otomatis dan responsif, dengan waktu rata-rata aktivasi 14,8–15,3 detik pada 8 siklus pengujian, serta tingkat keberhasilan pengendalian sebesar 100%.
- Penggunaan mikrokontroler ESP32 dan komunikasi MQTT memungkinkan data dikirim dan diterima secara dua arah antara perangkat keras dan aplikasi *Kodular*, mendukung pemantauan dan kontrol jarak jauh dengan latensi rendah.
- Aplikasi *Kodular* memberikan antarmuka monitoring yang intuitif, lengkap dengan status sensor, level air, dan kontrol manual, serta berhasil menerima data dengan tingkat keakuratan tinggi.
- Data historis berhasil dicatat secara periodik ke *Google Spreadsheet* melalui layanan *Google Apps Script* setiap 60 detik tanpa adanya kehilangan data.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Allah SWT atas rahmat dan karunia-Nya sehingga penelitian ini dapat diselesaikan dengan baik.

DAFTAR PUSTAKA

- [1] A. R. Azhar, D. A. Setiawan, N. A. A. Yasmin, T. A. Putri, and G. F. Nama, "Sistem Monitoring Kapasitas Air dan Pengisian Otomatis Berbasis IoT Menggunakan Modul ESP8266," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 12, no. 1, Jan. 2024, doi: 10.23960/jitet.v12i1.3966.
- [2] I. Gunawan, T. Akbar, and M. G. Ilham, "Prototipe Penerapan Internet Of Things (Iot) Pada Monitoring Level Air Tandon Menggunakan Nodemcu Esp8266 Dan Blynk," *Jurnal Informatika dan Teknologi*, vol. 3, no. 1, pp. 1–7, 2020.
- [3] G. Rama Maulana and E. Junianto, "Perancangan Sistem Monitoring Prediksi Dini Banjir Berbasis IoT di Sungai Cibungur Buniwangi," *E-Prosiding Teknik Informatika*, Jan. 2024.
- [4] Y. Herdiana and A. Triatna, "Prototype Monitoring Ketinggian Air Berbasis Internet Of Things Menggunakan Blynk Dan Nodemcu ESP8266 Pada Tangki," *Jurnal Informatika-COMPUTING*, vol. 07, pp. 1–11, Jun. 2020.
- [5] D. Dwi Pamungkas and K. Setyadjit, "Rancang Bangun Sistem Kontrol Dan Monitoring Pengisian Air Minum Isi Ulang Otomatis Menggunakan ESP32 Berbasis IoT," *Prosiding Senakama*, Jan. 2023.
- [6] T. D. Hakim, A. Rizqi, and N. Ashshidiq, "Rancang Bangun Sistem Kendali Ketinggian Level Air Pada Groundtank Berbasis ESP32," *Jurnal Elektro*, vol. 12, no. 1, Jan. 2024.
- [7] A. Yazid, Weny Indah Kusumawati, and R. Febriliana, "Prototipe Sistem Monitoring Suhu, Ketinggian Air, dan Kontrol Otomatis pada Budikdamber Berbasis IoT," *Journal of Advances in Information and Industrial Technology*, vol. 5, no. 2, pp. 93–104, Nov. 2023, doi: 10.52435/jaiit.v5i2.371.
- [8] Z. B. Abilovani, W. Yahya, and F. A. Bakhtiar, "Implementasi Protokol MQTT Untuk Sistem Monitoring Perangkat IoT," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 2, no. 12, pp. 7521–7527, Dec. 2018, [Online]. Available: <http://j-ptiik.ub.ac.id>
- [9] D. M. Hartanto, D. Hartanti, and J. Maulindar, "Desain dan Implementasi Sistem Pemantauan Stok Barang Berbasis IoT untuk Produk Segar," *Jurnal Pendidikan dan Teknologi Indonesia*, vol. 4, no. 11, pp. 477–487, Dec. 2024, doi: 10.52436/1.jpti.504.
- [10] P. A. W. Widyatmika, N. P. A. W. Indrawati, W. W. A. Prastya, I. K. Darminta, Sangka. I Gde Nyoman, and A. A. N. G. Sapteka, "Perbandingan Kinerja Arduino Uno dan ESP32 Terhadap Pengukuran Arus dan Tegangan," *Jurnal Otomasi, Kontrol & Instrumentasi*, 2021.
- [11] Ankit Negi, "ESP32 Dev board Pinout, Specifications, datasheet and Schematic." Accessed: Dec. 28, 2024. [Online]. Available: <https://www.etechnophiles.com/esp32-dev-board-pinout-specifications-datasheet-and-schematic/>
- [12] Mantech Electronics, "Arduino Distance Measuring Transducer Sensor Model:AJ-SR04M User Manual Operating Mode." Accessed: May 12, 2025. [Online]. Available: <https://mantech.co.za/Datasheets/Products/AJ-SR04M-200925A.pdf>
- [13] B. A. Salsabilah, A. Adhamatika, D. Triardianto, D. A. Putri, and S. C. Yuansah,

- “Pengujian Akurasi Sensor Ultrasonik HC-SR04 dalam Mengukur,” *The 2nd National Conference on Innovative Agriculture 2024*, Nov. 2024, doi: 10.25047/nacia.v2i1.236.
- [14] M. S. Syamsudin *et al.*, “Rancang Bangun Alat Pendeteksi dan Monitoring Banjir Menggunakan ESP32,” *Prosiding Seminar Nasional Teknologi Informasi dan Bisnis (SENATIB)*, Jul. 2023.
- [15] H. Kurnia, “Implementasi IoT Pada Sistem Monitoring Suhu dan Kelembapan Menggunakan ESP32, Firebase, dan Kodular,” *Jurnal Mahasiswa Teknik Informatika*, vol. 9, no. 1, Feb. 2025.
- [16] T. Sulistyorini, N. Sofi, and E. Sova, “Pemanfaatan Nodemcu ESP8266 Berbasis Android (Blynk) Sebagai Alat Alat Mematikan Dan Menghidupkan Lampu,” *Jurnal Ilmiah Teknik*, vol. 1, no. 3, Sep. 2022.
- [17] Elektronikindo.com, “Mengenal Solenoid Valve: Fungsi, Bagian dan Cara Kerjanya.” Accessed: Dec. 29, 2024. [Online]. Available: <https://elektronikindo.com/mengenal-solenoid-valve/>