

BAB II

TINJAUAN PUSTAKA

2.1 Penelitian Terkait

Untuk menunjukkan posisi dan kebaruan penelitian, disajikan ringkasan enam penelitian terdahulu yang relevan dan disusun secara bertahap: mulai dari pemantauan jaringan distribusi air, monitoring air berbasis IoT, pemilihan LoRa, analisis manajemen daya, hingga implementasi *deep sleep* ESP32. Seperti yang tersaji dalam Tabel 2.1.

Tabel 2.1 Perbandingan Penelitian Terdahulu

Peneliti (Tahun)	Fokus Penelitian	Teknologi & Metode	Keterbatasan / <i>Research Gap</i>
Guezouli, dkk. (2024) [26]	Monitoring kualitas air & deteksi kebocoran real- time	QoW-Pro, sensor aliran & kualitas air, WiFi/GSM, Random Forest	Komunikasi boros daya (WiFi/GSM), fokus kualitas air, belum LoRa, tanpa <i>deep sleep</i> , belum tekanan
Mounce, dkk. (2012) [27]	Analisis interval sampling data tekanan & debit	Perbandingan interval 1, 5, dan 15 menit untuk deteksi abnormal	Interval 1 & 5 menit tidak signifikan meningkatkan

	pada jaringan distribusi		deteksi tetapi menambah beban daya dan komunikasi; belum IoT, belum LoRa, belum deep sleep
Samijayani, dkk. (2023) [28]	WSN bertenaga surya untuk kualitas air	Zigbee, panel surya, sensor pH/TDS/turbidity	Parameter kualitas air, bukan tekanan/debit; Zigbee; tanpa deep sleep
Yüksel, dkk. (2020) [15]	Analisis konsumsi daya perangkat IoT berbasis Wi-Fi	Pengukuran arus rinci: deep sleep (0,025 mA), light sleep (0,8 mA), aktif transmisi (180 mA)	Membuktikan Wi-Fi boros dan pentingnya mode tidur, tetapi belum menggunakan LoRa
Safira, dkk. (2025)	Monitoring daya listrik via LoRa	LoRa SX1278, ESP32, uji LoS/NLoS (150 m /	Fokus daya listrik, tanpa deep sleep,

		80 m), RSSI, packet loss, Node-Red	tanpa interval 15 menit
Penelitian Ini	Rancang Bangun Prototipe Komunikasi Lora Dengan Manajemen Daya Deep Sleep Berbasis ESP32-S3 Dan Uji Coba Klasifikasi Anomali	ESP32-S3, LoRa SX1278 (P2P), Deep Sleep, Interval Pengiriman 15 Menit	Implementasi dibatasi pada skala simulasi jaringan distribusi air dan belum diterapkan pada jaringan operasional Perumdam

Adapun tabel matriks perbandingan penelitian terdahulu yang tersaji dalam Tabel 2.2.

Tabel 2.2 Matriks Perbandingan Penelitian Terdahulu

Fitur Utama	Guezouli, dkk. (2024)	Mounce, dkk. (2011)	Samijayan, dkk. (2023)	Yüksekl, dkk. (2020)	Safira, dkk. (2025)	Penelitian Ini
-------------	-----------------------	---------------------	------------------------	----------------------	---------------------	----------------

Parameter Tekanan Air	Tidak	Ya	Tidak	Tidak	Tidak	Ya
Parameter Debit Air	Ya	Ya	Tidak	Tidak	Tidak	Ya
Komunikasi LoRa SX1278	Tidak	Tidak	Ya	Tidak	Tidak	Ya
Manajemen Daya Deep Sleep ESP32	Tidak	Tidak	Tidak	Ya	Tidak	Ya
Interval Transmisi 15 Menit	Tidak	Ya	Tidak	Tidak	Tidak	Ya
Fokus Distribusi Air	Ya	Ya	Tidak	Tidak	Tidak	Ya

2.2 Landasan Teoretis

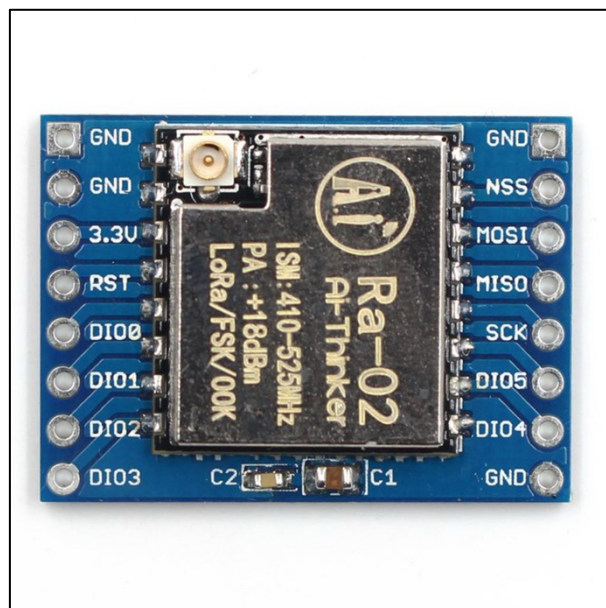
2.2.1 Internet of Things (IoT)

Internet of Things (IoT) merupakan konsep yang memungkinkan objek fisik saling terhubung dan bertukar data melalui jaringan internet tanpa interaksi manusia-ke-manusia secara langsung [29]. Arsitektur IoT secara generik terdiri atas empat lapisan: lapisan sensor/aktuator yang mengakuisisi data fisik, lapisan jaringan/*gateway* yang menghubungkan perangkat ke server, lapisan komputasi/*cloud* yang mengolah dan menyimpan data, serta lapisan aplikasi yang menyajikan informasi kepada pengguna [30]. Dalam konteks monitoring distribusi air bersih, IoT memungkinkan pemantauan parameter hidrolik seperti tekanan dan debit secara *real-time* untuk mendeteksi anomali jaringan secara dini.

2.2.2 LoRa (Long Range) SX1278

LoRa merupakan teknologi komunikasi nirkabel *Low Power Wide Area Network* (LPWAN) yang menggunakan modulasi *Chirp Spread Spectrum* (CSS) pada pita frekuensi ISM, seperti 868 MHz (Eropa), 915 MHz (Amerika/Indonesia), atau 433 MHz [31], [32]. Modul *transceiver* Semtech SX1278 adalah salah satu implementasi paling populer dari teknologi LoRa yang beroperasi pada rentang frekuensi 137–525 MHz dan mampu mencapai jangkauan hingga beberapa kilometer dengan konsumsi daya sangat rendah. Modul ini menawarkan sensitivitas penerimaan hingga -148 dBm, daya pancar

maksimum +20 dBm, serta konfigurasi parameter *spreading factor* (SF7–SF12), *bandwidth* (7,8–500 kHz), dan *coding rate* (CR 4/5–4/8) yang fleksibel, memungkinkan optimasi *trade-off* antara jangkauan, laju data (0,3–50 kbps), dan ketahanan interferensi [33], [34]. Dibandingkan dengan WiFi, SX1278 jauh lebih sesuai untuk aplikasi lapangan yang tersebar luas karena hanya perlu mengirim data berukuran kecil secara periodik, sehingga cocok untuk sistem monitoring distribusi air pedesaan atau perkotaan [35]. Dalam penelitian ini, modul SX1278 dikonfigurasi dalam mode *Point-to-Point* pada frekuensi 434 MHz untuk mengirimkan data tekanan dan debit dari menuju *gateway*. Gambaran disajikan pada Gambar 2.1.



Gambar 2.1 Module LoRa SX1278

Sumber: calcuttaelectronics.com

2.2.3 LoRa Point-to-Point

Konfigurasi LoRa *Point-to-Point* (P2P) adalah komunikasi langsung antara dua perangkat (*node* dan *gateway*) tanpa melalui infrastruktur server jaringan eksternal seperti pada arsitektur LoRaWAN. Dalam mode ini, *node* dapat mengirimkan data sensor secara langsung ke penerima (*gateway*) dengan pengaturan frekuensi, *bandwidth*, *spreading factor*, dan *coding rate* yang dikonfigurasi secara manual [36]. Kelebihan P2P adalah latensi rendah, kontrol penuh atas parameter radio, dan tidak memerlukan biaya berlangganan server, sehingga sangat sesuai untuk sistem dengan jumlah *node* terbatas seperti pada penelitian ini.

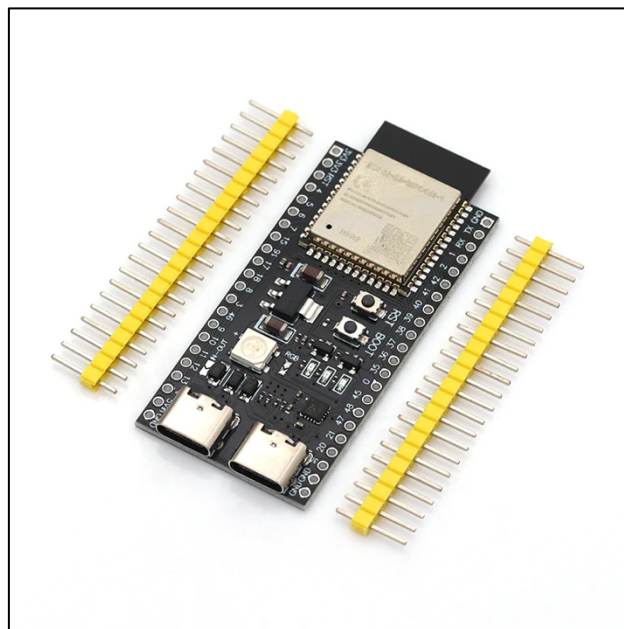
2.2.4 Gateway LoRa

Gateway dalam konteks jaringan LoRa berfungsi sebagai penerima data dari satu atau beberapa melalui radio frekuensi, kemudian meneruskan data tersebut ke jaringan internet atau server lokal melalui antarmuka seperti WiFi, Ethernet, atau serial [37]. Pada arsitektur LoRa *Point-to-Point*, *gateway* umumnya berupa perangkat tunggal yang terdiri atas modul transceiver SX1278 yang identik dengan yang digunakan *node*, dipadukan dengan mikrokontroler atau *single-board* computer (misalnya ESP32 atau Raspberry Pi) yang bertugas menangani penerimaan paket, pengecekan integritas data, serta penyajian atau penyimpanan data [34]. *Gateway* tidak hanya menerima data mentah, tetapi juga dapat melakukan pra-pemrosesan,

menghitung metrik seperti RSSI dan SNR per paket, dan meneruskan data ke platform pemantauan berbasis web atau *database* lokal [38].

2.2.5 Mikrokontroler ESP32-S3

ESP32-S3 merupakan *system-on-chip* (SoC) dari Espressif dengan prosesor *dual-core* Xtensa LX7, ADC 12-bit *successive approximation*, GPIO yang melimpah, serta *subsistem* manajemen daya rendah. Fitur yang relevan dengan penelitian ini mencakup RTC (*Real-Time Clock*), Mode *deep sleep* yang memungkinkan perangkat mengonsumsi arus sangat rendah saat tidak melakukan akuisisi atau transmisi data [39]. Dengan memanfaatkan mode operasi hemat daya ini, durasi operasi bertenaga baterai dapat diperpanjang secara signifikan. Gambaran disajikan pada Gambar 2.2.



Gambar 2.2 ESP32S3 DevKit

Sumber: s.alicdn.com

2.2.6 Deep Sleep pada ESP32-S3

Deep sleep adalah mode daya terendah pada ESP32-S3 di mana CPU utama, memori, dan sebagian besar periferal dimatikan, hanya menyisakan RTC yang tetap aktif. Dalam mode ini, chip dapat bangun (*wake-up*) melalui *timer*, *pin* GPIO eksternal, atau *touch pad* [39]. Durasi *deep sleep* diatur oleh *timer* RTC, sehingga dapat dijadwalkan untuk mengakuisisi data dan mengirimkannya dalam interval tertentu, lalu kembali tidur. Strategi ini menjadi inti pengurangan konsumsi daya total dan secara langsung memperpanjang masa pakai baterai.

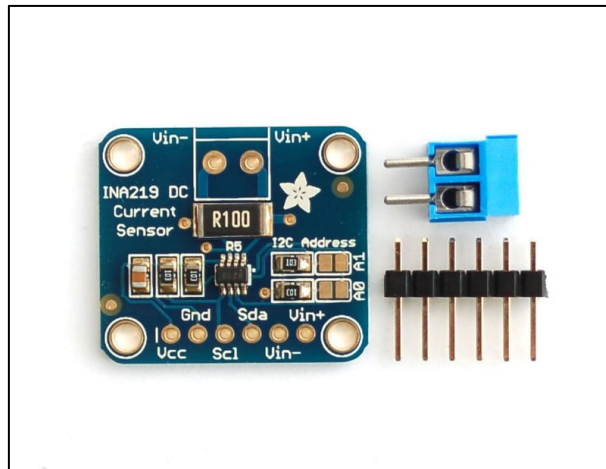
2.2.7 Manajemen Daya pada Sistem IoT

Manajemen daya menjadi aspek kritis dalam perancangan nirkabel bertenaga baterai. Konsep efisiensi energi dicapai melalui pengaturan *duty cycle*, yaitu rasio waktu aktif terhadap total periode operasi. Teknik *sleep scheduling* mematikan komponen yang tidak diperlukan dan hanya mengaktifkannya sesaat untuk akuisisi dan transmisi data. Penelitian [40] menekankan bahwa pada sistem monitoring air, interval pengiriman data dan pemilihan mode tidur harus dioptimalkan untuk menyeimbangkan kebutuhan *real-time* data dengan konsumsi energi. Fathurahman, dkk (2024) mendemonstrasikan bahwa penerapan manajemen daya berbasis *deep sleep* dan penjadwalan transmisi adaptif pada *node* LoRa dapat mengurangi konsumsi energi hingga 79,8% tanpa kehilangan signifikan pada *Packet Delivery Rasio* [41]. *Trade-off* antara

ketersediaan data dan umur sistem merupakan pertimbangan utama yang mendasari analisis konsumsi energi pada penelitian ini.

2.2.8 Sensor INA219

Sensor INA219 adalah modul pengukur arus dan tegangan berbasis antarmuka I²C yang diproduksi oleh Texas Instruments. Modul ini mampu mengukur tegangan bus hingga 26 V dan arus yang mengalir melalui resistor *shunt* internal dengan resolusi tinggi (12-bit ADC), serta secara otomatis menghitung daya dalam satuan watt [42]. INA219 mendukung mode pengukuran kontinu maupun *triggered*, dan menyediakan register untuk pembacaan tegangan *shunt*, tegangan bus, arus, dan daya yang dapat diakses melalui dua kabel antarmuka I²C. Akurasi pengukuran mencapai $\pm 1\%$ untuk arus dan $\pm 0,5\%$ untuk tegangan, dengan offset rendah yang memungkinkan pemantauan arus sangat kecil hingga arus puncak 3,2 A (dengan resistor *shunt* 0,1 Ω). Keunggulan sensor ini terletak pada kemudahan integrasinya ke mikrokontroler seperti ESP32-S3, dan kemampuannya mengukur baik sisi tegangan tinggi maupun rendah. Dalam penelitian ini, INA219 ditempatkan pada sisi *gateway* dan dipasang seri pada jalur catu daya, sehingga arus dan daya yang dikonsumsi selama pengujian dapat diukur secara langsung. Gambaran disajikan pada Gambar 2.3.



Gambar 2.3 sensor INA219

Sumber: cdn-learn.adafruit.com

2.2.9 Baterai Lithium-Ion

Baterai lithium-ion (Li-ion) dan lithium polymer (Li-Po) banyak digunakan pada perangkat portabel karena kepadatan energi yang tinggi, tegangan nominal 3,7 V per sel, dan siklus pengisian yang relatif panjang. Kapasitas baterai dinyatakan dalam miliampere-jam (mAh) yang menentukan jumlah muatan listrik yang dapat disimpan [43]. Salah satu jenis baterai isi ulang lithium-ion berbentuk silinder dengan diameter 18 mm dan panjang 65 mm yaitu Baterai 18650. Baterai ini memiliki tegangan nominal 3,7 Volt dengan kapasitas umum berkisar antara 1.200 mAh hingga 3.500 mAh. Gambaran baterai lithium-ion seperti yang disajikan pada Gambar 2.4.



Gambar 2.4 Baterai Lithium-Ion

Sumber: id.gnscomponent.com

2.2.10 Packet Delivery Rasio (PDR)

Packet Delivery Ratio (PDR) adalah metrik keandalan komunikasi yang didefinisikan sebagai perbandingan jumlah paket data yang berhasil diterima oleh *receiver* terhadap total paket yang dikirim oleh *sender*, dinyatakan dalam persen:

$$PDR = \frac{Packet\ Received}{Packet\ Sent} \times 100\%$$

[44]. PDR menjadi parameter pengujian utama untuk menilai kualitas *link* LoRa P2P pada berbagai jarak dan kondisi lingkungan, sehingga dapat memvalidasi bahwa data tekanan dan debit sampai dengan utuh di *gateway*.

2.2.11 RSSI (Received Signal Strength Indicator)

RSSI adalah indikator kekuatan sinyal yang diterima pada perangkat penerima, umumnya dinyatakan dalam dBm. Nilai RSSI yang lebih tinggi (kurang negatif) menunjukkan sinyal yang lebih

kuat, sementara nilai yang sangat rendah mendekati *sensitivity limit* perangkat. Dalam sistem komunikasi LoRa, RSSI digunakan untuk memperkirakan jarak dan kualitas kanal transmisi [45]. Data RSSI membantu analisis faktor eksternal (rintangan, interferensi) terhadap performa *link* pada pengujian lapangan.

2.2.12 Signal to Noise Rasio (SNR)

SNR adalah perbandingan antara daya sinyal informasi dan daya *noise* latar belakang pada penerima. SNR positif menunjukkan sinyal lebih kuat dari *noise*, sedangkan SNR negatif masih dapat didekode oleh LoRa berkat keunggulan modulasi CSS. Interpretasi SNR menjadi dasar penentuan *spreading factor* yang optimal—semakin rendah SNR, dibutuhkan *spreading factor* lebih tinggi yang juga meningkatkan konsumsi energi [46]. Dengan demikian, SNR menjadi parameter penting dalam optimasi *trade-off* antara jangkauan, keandalan, dan konsumsi daya.

2.2.13 Konsumsi Energi dan Estimasi Umur Baterai

Konsumsi energi total sistem dihitung dengan menjumlahkan energi yang digunakan pada setiap mode operasi (*active*, *deep sleep*, *transmission*). Energi (E) dinyatakan sebagai hasil kali tegangan (V), arus (I), dan durasi (t) tiap mode: $E = V \times I \times t$. Estimasi umur baterai (T) dapat dihitung dengan membagi kapasitas baterai (C) dengan arus rata-rata sistem (I_{avg}):

$$Battery\ Life = \frac{C}{I_{avg}}$$

dengan memperhatikan faktor efisiensi dan *depth of discharge* (DoD) yang aman [47]. Subbab ini menjadi dasar perhitungan dan analisis pada BAB V, terutama untuk menjawab rumusan masalah tentang pengaruh manajemen daya terhadap durasi operasi berbasis baterai.

2.2.14 Perancangan Sistem

Perancangan sistem adalah aktivitas mendefinisikan arsitektur, komponen, modul, antarmuka, dan data untuk suatu sistem agar dapat memenuhi kebutuhan yang telah ditentukan [48]. Dalam konteks sistem tertanam dan IoT, perancangan sistem meliputi perancangan perangkat keras (sensor, aktuator, pengolah), perancangan perangkat lunak (algoritma, struktur program), perancangan komunikasi data (protokol, format pesan), serta perancangan antarmuka pengguna [49]. Tujuan perancangan sistem adalah menghasilkan cetak biru yang jelas sehingga proses implementasi dapat berjalan terarah, risiko kesalahan terminimalkan, dan sistem akhir sesuai dengan spesifikasi kebutuhan. Pada penelitian ini, perancangan sistem menjadi dasar untuk menentukan arsitektur komunikasi LoRa *Point-to-Point*, alur pembacaan data tekanan dan debit, format paket data, serta strategi manajemen daya berbasis *deep sleep* pada ESP32-S3.

2.2.15 Diagram Blok Sistem

Diagram blok merupakan representasi grafis yang menggambarkan hubungan fungsional antarkomponen utama suatu

sistem melalui blok-blok yang saling terhubung oleh garis sinyal atau aliran data [50]. Setiap blok mewakili fungsi tertentu, misalnya penginderaan, pemrosesan, komunikasi, dan pencatu daya. Diagram blok memudahkan pemahaman struktur sistem secara keseluruhan tanpa harus masuk ke detail implementasi, sehingga sering digunakan pada tahap awal perancangan [51].

2.2.16 Flowchart Sistem

Flowchart adalah diagram yang merepresentasikan alur logika atau urutan proses dalam suatu sistem menggunakan simbol-simbol standar [52]. Simbol seperti oval (mulai/selesai), persegi panjang (proses), belah ketupat (keputusan), dan panah (aliran) memungkinkan pengembang memodelkan algoritma secara visual sehingga mudah ditelusuri dan diverifikasi sebelum ditulis menjadi kode program [53]. Gambaran disajikan pada Gambar 2.5.

	Flow Simbol yang digunakan untuk menggabungkan antara simbol yang satu dengan simbol yang lain. Simbol ini disebut juga dengan Connecting Line.		Input/output Simbol yang menyatakan proses input atau output tanpa tergantung peralatan.
	On-Page Reference Simbol untuk keluar - masuk atau penyambungan proses dalam lembar kerja yang sama.		Manual Operation Simbol yang menyatakan suatu proses yang tidak dilakukan oleh komputer.
	Off-Page Reference Simbol untuk keluar - masuk atau penyambungan proses dalam lembar kerja yang berbeda.		Document Simbol yang menyatakan bahwa input berasal dari dokumen dalam bentuk fisik, atau output yang perlu dicetak.
	Terminator Simbol yang menyatakan awal atau akhir suatu program.		Predefine Proses Simbol untuk pelaksanaan suatu bagian (sub-program) atau prosedur.
	Process Simbol yang menyatakan suatu proses yang dilakukan komputer.		Display Simbol yang menyatakan peralatan output yang digunakan.
	Decision Simbol yang menunjukkan kondisi tertentu yang akan menghasilkan dua kemungkinan jawaban, yaitu ya dan tidak.		Preparation Simbol yang menyatakan penyediaan tempat penyimpanan suatu pengolahan untuk memberikan nilai awal.

Gambar 2.5 Simbol Flowchart

Sumber: dicoding.com

2.2.17 Skema Rangkaian

Skema rangkaian merupakan representasi grafis dari suatu rangkaian elektronika yang menggunakan simbol-simbol standar untuk menggambarkan komponen serta hubungan kelistrikan antar komponen secara rinci [54]. Skema berbeda dari diagram blok karena menunjukkan koneksi *pin* demi *pin*, nilai komponen, tegangan, dan arah aliran sinyal, sehingga menjadi acuan utama dalam proses perakitan, pengawatan, dan pembuatan papan sirkuit cetak (PCB). Skema memungkinkan perancang dan teknisi menelusuri jalur sinyal, mengidentifikasi potensi konflik tegangan, dan memverifikasi kebenaran desain sebelum implementasi fisik dilakukan [55].

2.2.18 Metode Prototype

Metode *prototype* adalah pendekatan pengembangan sistem yang menekankan pembuatan *prototype* secara cepat dan bertahap untuk mendapatkan umpan balik pengguna sebagai dasar penyempurnaan [56]. Tahapan metode ini meliputi komunikasi awal, perencanaan cepat, pemodelan desain cepat, konstruksi prototipe, serta penyerahan dan evaluasi oleh pengguna [48]. Metode *prototype* cocok diterapkan pada sistem yang memerlukan integrasi perangkat keras dan perangkat lunak serta spesifikasi yang tidak seluruhnya dapat ditentukan di awal, seperti pada sistem monitoring IoT berbasis sensor dan komunikasi nirkabel.

2.2.19 Arduino IDE

Arduino IDE (*Integrated Development Environment*) adalah perangkat lunak sumber terbuka yang menyediakan lingkungan terpadu untuk menulis, mengompilasi, dan mengunggah kode program ke papan mikrokontroler, termasuk ESP32-S3 melalui dukungan *board package* tambahan [57]. IDE ini menggunakan bahasa pemrograman C/C++ yang telah disederhanakan, dilengkapi dengan pustaka inti, manajer pustaka eksternal, serta fitur serial monitor dan serial plotter yang memfasilitasi pemantauan data dan proses *debugging* secara *real-time* [58]. Kemudahan instalasi, konfigurasi *board*, serta ketersediaan pustaka komunitas menjadikan Arduino IDE pilihan yang efisien untuk pengembangan *prototype* sistem tertanam.

2.2.20 Klasifikasi Anomali dengan Logistic Regression

Integrasi model prediksi kebocoran merupakan proses menyematkan algoritma pembelajaran mesin (seperti *logistic regression*) ke dalam arsitektur sistem monitoring agar mampu mengklasifikasikan kondisi jaringan distribusi air secara otomatis. Dalam arsitektur IoT, model dapat ditempatkan pada dua lokasi, yaitu *edge computing*, di mana inferensi dilakukan langsung pada atau *gateway* lokal, dan/atau *cloud computing*, di mana data mentah dikirim ke server dan inferensi dilakukan di pusat data [59], [60]. Untuk sistem berdaya rendah seperti node LoRa, inferensi di edge (*gateway*) lebih disukai karena mengurangi latensi, menghemat energi

transmisi, dan tetap bekerja meskipun koneksi internet terputus [61]. Proses integrasi mencakup akuisisi data tekanan dan debit dari sensor, pra-pemrosesan (normalisasi, penanganan missing value), pengambilan keputusan oleh model yang telah dilatih dan diunggah ke dalam mikrokontroler atau komputer mini [62]. Pada penelitian ini, model logistic regression dilatih menggunakan dataset tekanan dan debit pada kondisi normal dan bocor, kemudian koefisien regresinya ditanamkan ke dalam kode C/C++ pada .